
Veronte Autopilot (Up to HW 4.5 / SW 6.4)

Release 6.4.90

Embention

2023-08-22

CONTENTS

| | | |
|-----------|--|-----------|
| 1 | Quick Start | 1 |
| 1.1 | First Steps | 1 |
| 1.1.1 | Kit Elements | 2 |
| 1.1.2 | Veronte Pipe | 2 |
| 1.2 | Model Simulation | 5 |
| 1.2.1 | Loading a Template to the Autopilot | 5 |
| 1.2.2 | X-Plane and Veronte Pipe Setup | 7 |
| 1.2.2.1 | Veronte Pipe | 7 |
| 1.2.2.2 | X-Plane | 7 |
| 1.2.3 | Executing a Simulation | 8 |
| 1.3 | Radio Pairing (LOS) | 15 |
| 1.3.1 | Microhard setup helper | 18 |
| 1.4 | Calibrating Servos | 20 |
| 1.4.1 | Servos Output | 20 |
| 1.4.2 | SU Matrix | 22 |
| 1.4.3 | System Trim | 24 |
| 1.5 | External Transmitter Configuration | 27 |
| 1.5.1 | PPM | 29 |
| 1.5.2 | Output | 30 |
| 1.6 | Pre-flight Checklist | 31 |
| 1.6.1 | Main Checklist | 31 |
| 1.6.2 | Other Items | 37 |
| 1.7 | PDI Mode | 37 |
| 2 | System Layout | 41 |
| 2.1 | Communication Layouts | 41 |
| 2.1.1 | Air Segment | 41 |
| 2.1.1.1 | Line-of-Sight (LOS) | 41 |
| 2.1.1.1.1 | Internal Radiolink | 41 |
| 2.1.1.1.2 | External Radiolink | 42 |
| 2.1.1.1.3 | Tethered | 42 |
| 2.1.1.2 | Beyond Line-of-Sight (BLOS) | 43 |
| 2.1.1.2.1 | Internal 4G + Veronte Cloud | 43 |
| 2.1.1.2.2 | External Internet access + Veronte Cloud | 43 |
| 2.1.1.2.3 | External Satellite communication | 44 |
| 2.1.1.2.4 | Remote GCS | 44 |
| 2.1.2 | Ground Segment | 45 |
| 2.1.2.1 | Serial Interface | 45 |
| 2.1.2.2 | External Radiolink (GS) | 46 |
| 2.1.2.3 | Ethernet | 46 |

| | | |
|----------|--------------------------------------|-----------|
| 2.1.2.4 | Wifi | 47 |
| 2.2 | Manual Control Layouts | 47 |
| 2.2.1 | PPM to Ground Unit | 48 |
| 2.2.2 | PPM to Air Unit | 48 |
| 2.2.3 | USB to PPM | 49 |
| 2.2.4 | USB to Pipe | 50 |
| 2.2.5 | Virtual Stick | 51 |
| 2.3 | Point to Multipoint Layouts | 51 |
| 2.3.1 | Point to Point with single Pipe | 51 |
| 2.3.2 | Point to Multipoint with single GS | 52 |
| 2.3.3 | Multipoint to Point with multiple GS | 52 |
| 3 | Hardware Installation | 55 |
| 3.1 | Veronte Autopilot | 55 |
| 3.1.1 | Aircraft Mounting | 55 |
| 3.1.1.1 | Enclosure | 55 |
| 3.1.1.2 | OEM | 56 |
| 3.1.1.3 | Mechanical Mounting | 57 |
| 3.1.1.4 | Vibration Isolation | 57 |
| 3.1.1.5 | Location | 58 |
| 3.1.1.6 | Orientation | 58 |
| 3.1.1.7 | Connector Layout | 58 |
| 3.1.1.8 | Mating Connector | 59 |
| 3.1.1.9 | Antenna Integration | 59 |
| 3.1.1.10 | Pressure lines | 60 |
| 3.1.2 | Electrical | 60 |
| 3.1.2.1 | Power | 60 |
| 3.1.2.2 | Veronte I/O Signals | 61 |
| 3.1.2.3 | Flight Termination System (FTS) | 63 |
| 3.1.2.4 | Joystick | 63 |
| 3.1.3 | Performances | 66 |
| 3.1.4 | Annex 1: Connector colour code | 66 |
| 3.1.5 | Annex 2: Connection examples | 68 |
| 3.1.5.1 | Ground Stations | 68 |
| 3.1.5.2 | Aircrafts | 72 |
| 3.1.6 | Operation | 75 |
| 3.1.7 | Platforms | 75 |
| 3.1.8 | Safety | 76 |
| 3.2 | 4x Veronte Autopilot | 76 |
| 3.2.1 | Aircraft Mounting | 76 |
| 3.2.1.1 | Enclosure | 76 |
| 3.2.1.2 | Vibration Isolation | 77 |
| 3.2.1.3 | Location | 78 |
| 3.2.1.4 | Orientation | 78 |
| 3.2.1.5 | Connector Layout | 79 |
| 3.2.1.6 | Mating Connectors | 80 |
| 3.2.1.7 | Antenna Integration | 81 |
| 3.2.1.8 | Pressure lines | 81 |
| 3.2.2 | Electrical | 82 |
| 3.2.2.1 | Power | 82 |
| 3.2.2.2 | Redundant Connector Pinout | 82 |
| 3.2.2.3 | Arbiter Connector Pinout | 87 |
| 3.2.3 | Performances | 95 |
| 3.2.4 | Annex 1: Connection Examples | 96 |

| | | |
|----------|---|------------|
| 3.2.4.1 | Ground Station | 96 |
| 3.2.5 | Annex 2: Connector colour code | 96 |
| 4 | Pipe Configuration | 101 |
| 4.1 | Installation | 101 |
| 4.1.1 | System Requeriments | 101 |
| 4.1.2 | Windows | 101 |
| 4.1.3 | MAC | 104 |
| 4.1.4 | Linux | 106 |
| 4.2 | Update | 110 |
| 4.3 | Preferences | 113 |
| 4.3.1 | General | 113 |
| 4.3.2 | Connections | 117 |
| 4.3.3 | Encryption | 122 |
| 4.3.3.1 | Remove Encryption | 125 |
| 4.3.4 | System Units | 125 |
| 4.3.4.1 | Creating Custom Units | 126 |
| 4.3.4.2 | Units | 127 |
| 4.3.5 | Map | 128 |
| 4.3.5.1 | Terrain Height | 128 |
| 4.3.5.2 | Download Altitude Information | 129 |
| 4.3.5.3 | Custom Terrain File | 130 |
| 5 | Users & access level | 133 |
| 5.1 | User Toolbar | 133 |
| 5.2 | User Manage | 133 |
| 5.3 | New User | 134 |
| 6 | Workspace | 137 |
| 6.1 | Workspace Toolbar | 137 |
| 6.1.1 | Edit Workspace | 138 |
| 6.1.2 | Add workspace | 139 |
| 6.1.3 | Import/Export Workspaces | 140 |
| 6.2 | Map Display | 140 |
| 6.2.1 | Map provider | 140 |
| 6.2.2 | Custom image | 141 |
| 6.3 | Widgets | 142 |
| 6.3.1 | Gimbal | 142 |
| 6.3.2 | Gauge Display | 144 |
| 6.3.2.1 | Gauge Selection | 144 |
| 6.3.2.2 | Gauge Configuration | 146 |
| 6.3.2.3 | Alarm Creation | 147 |
| 6.3.3 | Advanced Primary Flight Display (PFD) | 149 |
| 6.3.4 | Terrain | 152 |
| 6.3.5 | Autotune Tool | 153 |
| 6.3.6 | Line of Sight | 154 |
| 6.3.7 | ADS-B decoder | 155 |
| 6.3.8 | GNSS status | 156 |
| 6.3.9 | Telecommand | 157 |
| 6.3.9.1 | Stick | 157 |
| 6.3.9.2 | Change variable | 159 |
| 6.3.10 | Stick | 160 |
| 6.4 | Main Interface | 162 |
| 7 | Veronte Configuration | 165 |

| | | |
|-------------|--|-----|
| 7.1 | File Management | 165 |
| 7.1.1 | Import Configurations | 165 |
| 7.1.1.1 | Check PDI | 170 |
| 7.1.2 | Export Configurations | 172 |
| 7.1.3 | Offline Configurations | 174 |
| 7.1.4 | Version Control | 175 |
| 7.1.5 | Update Onboard Software | 175 |
| 7.1.6 | Maintenance Mode | 176 |
| 7.1.7 | PDI Files | 178 |
| 7.2 | Setup Toolbar | 178 |
| 7.2.1 | Veronte | 178 |
| 7.2.2 | Connections | 179 |
| 7.2.2.1 | ADC | 179 |
| 7.2.2.1.1 | Application example | 180 |
| 7.2.2.2 | Arbiter - SuC | 180 |
| 7.2.2.3 | CAN | 181 |
| 7.2.2.3.1 | CAN Configuration | 181 |
| 7.2.2.3.2 | CAN Telemetry | 183 |
| 7.2.2.3.2.1 | TX Messages | 184 |
| 7.2.2.3.2.2 | RX Messages | 186 |
| 7.2.2.3.3 | CAN Messages | 186 |
| 7.2.2.3.3.1 | Variable | 187 |
| 7.2.2.3.3.2 | Checksum | 188 |
| 7.2.2.3.3.3 | Matcher | 189 |
| 7.2.2.3.3.4 | Skip | 190 |
| 7.2.2.3.3.5 | Parse ASCII | 190 |
| 7.2.2.3.3.6 | Position | 190 |
| 7.2.2.3.4 | Arbiter CAN protocol | 191 |
| 7.2.2.3.4.1 | CAN parameters | 191 |
| 7.2.2.3.4.2 | CAN message structure | 191 |
| 7.2.2.3.4.3 | Single message structure | 193 |
| 7.2.2.3.4.4 | External autopilot simulation | 195 |
| 7.2.2.3.4.5 | Can protocol for 4-autopilot configuration | 195 |
| 7.2.2.4 | GPIO/PWM | 197 |
| 7.2.2.4.1 | GPIO | 197 |
| 7.2.2.4.2 | PWM | 199 |
| 7.2.2.4.3 | Add | 200 |
| 7.2.2.5 | Serial | 202 |
| 7.2.3 | Devices | 205 |
| 7.2.3.1 | Actuators | 205 |
| 7.2.3.1.1 | Physical | 205 |
| 7.2.3.1.1.1 | Calibration Interface | 205 |
| 7.2.3.1.1.2 | Configuration Wizard | 206 |
| 7.2.3.1.2 | Logical | 210 |
| 7.2.3.1.2.1 | <i>SU</i> & <i>US</i> Matrices | 210 |
| 7.2.3.1.2.2 | <i>SU</i> of a Flying Wing | 213 |
| 7.2.3.1.2.3 | <i>SU</i> of a V-Tail Aircraft | 213 |
| 7.2.3.1.2.4 | <i>SU</i> of a Quadcopter | 214 |
| 7.2.3.1.2.5 | <i>SU</i> of a VTOL Aircraft | 215 |
| 7.2.3.1.3 | Saturation | 215 |
| 7.2.3.1.3.1 | Standard Saturation | 216 |
| 7.2.3.2 | Communications | 221 |
| 7.2.3.2.1 | 4G | 221 |
| 7.2.3.2.1.1 | ESIM | 221 |

| | | |
|--------------|--------------------------|-----|
| 7.2.3.2.1.2 | SIM | 222 |
| 7.2.3.2.2 | Comstats | 223 |
| 7.2.3.2.3 | Iridium | 224 |
| 7.2.3.2.4 | Ports | 225 |
| 7.2.3.3 | Payload | 226 |
| 7.2.3.3.1 | Camera | 226 |
| 7.2.3.3.1.1 | Veronte Gimbal Wizard | 228 |
| 7.2.3.3.2 | Gimbal/Tracker | 231 |
| 7.2.3.3.3 | Transponder | 232 |
| 7.2.3.3.4 | ADS-B | 235 |
| 7.2.3.4 | Sensors | 240 |
| 7.2.3.4.1 | Accelerometer | 240 |
| 7.2.3.4.1.1 | Integer var sensor | 241 |
| 7.2.3.4.1.2 | Decimal var sensor | 242 |
| 7.2.3.4.1.3 | Internal | 243 |
| 7.2.3.4.2 | Altimeter | 245 |
| 7.2.3.4.3 | GNSS | 247 |
| 7.2.3.4.3.1 | GNSS 1 & 2 Configuration | 247 |
| 7.2.3.4.3.2 | Configuration | 247 |
| 7.2.3.4.3.3 | SBAS | 248 |
| 7.2.3.4.3.4 | Message Rate | 249 |
| 7.2.3.4.3.5 | Estimation Error | 250 |
| 7.2.3.4.3.6 | Advanced | 251 |
| 7.2.3.4.3.7 | GNSS Compass | 252 |
| 7.2.3.4.3.8 | RTK Configuration | 258 |
| 7.2.3.4.3.9 | GNSS External | 261 |
| 7.2.3.4.3.10 | Configuration | 262 |
| 7.2.3.4.3.11 | NMEA Parser | 263 |
| 7.2.3.4.3.12 | Estimation Error | 264 |
| 7.2.3.4.3.13 | NTRIP | 264 |
| 7.2.3.4.3.14 | Registration | 264 |
| 7.2.3.4.3.15 | Configuration | 264 |
| 7.2.3.4.4 | Gyroscope | 267 |
| 7.2.3.4.4.1 | Integer var sensor | 268 |
| 7.2.3.4.4.2 | Decimal var sensor | 269 |
| 7.2.3.4.4.3 | Internal | 270 |
| 7.2.3.4.5 | I2C Devices | 272 |
| 7.2.3.4.5.1 | Lidar Devices | 272 |
| 7.2.3.4.5.2 | Magnetometer Devices | 273 |
| 7.2.3.4.6 | Magnetometer | 274 |
| 7.2.3.4.6.1 | Calibration | 274 |
| 7.2.3.4.6.2 | Configuration | 275 |
| 7.2.3.4.6.3 | Integer var sensor | 276 |
| 7.2.3.4.6.4 | Decimal var sensor | 277 |
| 7.2.3.4.6.5 | Internal | 278 |
| 7.2.3.4.6.6 | External | 279 |
| 7.2.3.4.7 | Obstacle Detection | 280 |
| 7.2.3.4.7.1 | External | 280 |
| 7.2.3.4.7.2 | Obsens | 281 |
| 7.2.3.4.7.3 | Range sensors | 283 |
| 7.2.3.4.8 | Pressure | 284 |
| 7.2.3.4.8.1 | Dynamic | 284 |
| 7.2.3.4.8.2 | Sensor | 284 |
| 7.2.3.4.8.3 | Integer var sensor | 284 |

| | | |
|--------------|----------------------------------|-----|
| 7.2.3.4.8.4 | Decimal var sensor | 285 |
| 7.2.3.4.8.5 | Internal | 286 |
| 7.2.3.4.8.6 | Navigation | 287 |
| 7.2.3.4.8.7 | Static | 289 |
| 7.2.3.4.8.8 | Sensor | 289 |
| 7.2.3.4.8.9 | Integer var sensor | 289 |
| 7.2.3.4.8.10 | Decimal var sensor | 290 |
| 7.2.3.4.8.11 | Internal | 291 |
| 7.2.3.4.8.12 | Navigation | 292 |
| 7.2.3.4.8.13 | Ground Effect | 294 |
| 7.2.3.4.9 | RPM | 295 |
| 7.2.3.4.10 | Speed Down | 296 |
| 7.2.3.4.11 | Ultrasounds | 297 |
| 7.2.3.4.11.1 | Sensor | 297 |
| 7.2.3.4.11.2 | Base | 298 |
| 7.2.3.4.11.3 | Transmitter | 299 |
| 7.2.3.4.11.4 | Navigation | 299 |
| 7.2.3.5 | Stick Configuration | 301 |
| 7.2.3.5.1 | Arcade trim | 301 |
| 7.2.3.5.2 | Local Sources | 302 |
| 7.2.3.5.2.1 | Transmitter | 302 |
| 7.2.3.5.2.2 | PPM | 302 |
| 7.2.3.5.2.3 | Exponential | 304 |
| 7.2.3.5.2.4 | Trim | 304 |
| 7.2.3.5.2.5 | Output | 305 |
| 7.2.3.5.2.6 | Virtual Stick | 306 |
| 7.2.3.5.3 | Test Stick | 308 |
| 7.2.3.5.4 | Stick | 309 |
| 7.2.3.5.4.1 | Wizard Stick | 313 |
| 7.2.3.5.4.2 | Mask Servos | 314 |
| 7.2.3.5.4.3 | Transmitter Inputs | 316 |
| 7.2.3.6 | Others | 317 |
| 7.2.3.6.1 | Can Configuration | 317 |
| 7.2.3.6.1.1 | Configuration | 317 |
| 7.2.3.6.1.2 | CAN Telemetry | 319 |
| 7.2.3.6.1.3 | Custom Messages | 319 |
| 7.2.3.6.1.4 | Variable | 319 |
| 7.2.3.6.1.5 | Checksum | 321 |
| 7.2.3.6.1.6 | Matcher | 322 |
| 7.2.3.6.1.7 | Skip | 323 |
| 7.2.3.6.1.8 | Parse ASCII | 323 |
| 7.2.3.6.1.9 | Position | 323 |
| 7.2.3.6.1.10 | TX Messages | 326 |
| 7.2.3.6.1.11 | RX Messages | 327 |
| 7.2.3.6.1.12 | Mailboxes | 329 |
| 7.2.3.6.2 | I/O Manager | 330 |
| 7.2.3.6.2.1 | Serial Custom Messages | 330 |
| 7.2.3.6.2.2 | Message received | 331 |
| 7.2.3.6.2.3 | Tunnel | 332 |
| 7.2.3.6.3 | Digital Input Manager | 335 |
| 7.2.3.6.4 | Payload | 336 |
| 7.2.3.6.4.1 | Payload Operation | 337 |
| 7.2.4 | Control | 341 |
| 7.2.4.1 | Phases | 341 |

| | | |
|--------------|---------------------------------------|-----|
| 7.2.4.1.1 | Guidance | 341 |
| 7.2.4.1.1.1 | Guidance Variables | 341 |
| 7.2.4.1.1.2 | Taxi | 342 |
| 7.2.4.1.1.3 | VTOL | 344 |
| 7.2.4.1.1.4 | Guidance-generated Variables | 346 |
| 7.2.4.1.1.5 | Climb | 347 |
| 7.2.4.1.1.6 | Climbing guidance parameters behavior | 352 |
| 7.2.4.1.1.7 | Guidance-generated Variables | 354 |
| 7.2.4.1.1.8 | Cruise | 355 |
| 7.2.4.1.1.9 | Guidance-generated Variables | 357 |
| 7.2.4.1.1.10 | Hold | 358 |
| 7.2.4.1.1.11 | Landing | 360 |
| 7.2.4.1.1.12 | Guidance-generated Variables | 365 |
| 7.2.4.1.1.13 | Yaw | 366 |
| 7.2.4.1.1.14 | Rendezvous | 367 |
| 7.2.4.1.2 | Arcade | 370 |
| 7.2.4.1.2.1 | Arcade Mode Settings | 370 |
| 7.2.4.1.3 | Loop | 371 |
| 7.2.4.1.3.1 | PID Controller | 371 |
| 7.2.4.1.3.2 | Adaptative-Predictive Control | 374 |
| 7.2.4.1.3.3 | Gain Scheduling | 375 |
| 7.2.4.1.3.4 | Fixed Control | 378 |
| 7.2.4.1.3.5 | PID Settings | 379 |
| 7.2.4.1.3.6 | Exporting PIDs To Other Phases | 381 |
| 7.2.4.1.4 | TC Panel | 381 |
| 7.2.4.1.4.1 | Cruise Guidance | 382 |
| 7.2.4.1.5 | Automations | 384 |
| 7.2.4.2 | Runway | 386 |
| 7.2.4.2.1 | Runway | 386 |
| 7.2.4.2.2 | Spot | 388 |
| 7.2.4.3 | Smooth | 388 |
| 7.2.4.4 | Envelope | 390 |
| 7.2.4.5 | Modes | 394 |
| 7.2.4.6 | Arcade Axis | 395 |
| 7.2.5 | Navigation | 397 |
| 7.2.5.1 | Wind Estimation | 397 |
| 7.2.6 | Automations | 400 |
| 7.2.6.1 | Events | 400 |
| 7.2.6.1.1 | Alarm | 400 |
| 7.2.6.1.2 | Button | 401 |
| 7.2.6.1.3 | Mode | 401 |
| 7.2.6.1.4 | Phase | 402 |
| 7.2.6.1.5 | Polygon | 403 |
| 7.2.6.1.6 | Route | 405 |
| 7.2.6.1.7 | Timer | 406 |
| 7.2.6.1.8 | Variable | 407 |
| 7.2.6.2 | Actions | 409 |
| 7.2.6.2.1 | Arcade Trim | 409 |
| 7.2.6.2.2 | Atmosphere Calibration | 410 |
| 7.2.6.2.3 | Change Active Sensor | 410 |
| 7.2.6.2.4 | Command Block | 411 |
| 7.2.6.2.4.1 | Gimbal | 411 |
| 7.2.6.2.4.2 | ArcTrim | 413 |
| 7.2.6.2.5 | Custom CAN TX | 414 |

| | | |
|--------------|--------------------------------|-----|
| 7.2.6.2.6 | Custom Serial TX | 415 |
| 7.2.6.2.7 | DEM Calibration | 416 |
| 7.2.6.2.8 | Enable/Disable Wind Estimation | 416 |
| 7.2.6.2.9 | Entrance EKF | 417 |
| 7.2.6.2.10 | Envelope | 418 |
| 7.2.6.2.11 | FTS Activation | 419 |
| 7.2.6.2.12 | Feature | 420 |
| 7.2.6.2.13 | Format SD | 421 |
| 7.2.6.2.14 | Go To | 422 |
| 7.2.6.2.15 | Ground Effect | 423 |
| 7.2.6.2.16 | Mode | 424 |
| 7.2.6.2.17 | Navigation | 425 |
| 7.2.6.2.18 | Obstacle Avoidance | 426 |
| 7.2.6.2.19 | Output | 427 |
| 7.2.6.2.20 | Periodical | 429 |
| 7.2.6.2.21 | Phases | 430 |
| 7.2.6.2.22 | Ports | 430 |
| 7.2.6.2.23 | Run Block Program | 431 |
| 7.2.6.2.24 | Run Operation | 433 |
| 7.2.6.2.25 | Safety Bits | 434 |
| 7.2.6.2.26 | Select Arcade Axis | 434 |
| 7.2.6.2.27 | Stick Priority | 435 |
| 7.2.6.2.28 | Terrain Obstacle | 436 |
| 7.2.6.2.29 | Track | 437 |
| 7.2.6.2.29.1 | Position | 438 |
| 7.2.6.2.29.2 | Follow Leader | 439 |
| 7.2.6.2.30 | User Log | 440 |
| 7.2.6.2.31 | Variable | 440 |
| 7.2.6.2.32 | Yaw | 441 |
| 7.2.6.3 | Other Options | 445 |
| 7.2.7 | Variables | 447 |
| 7.2.7.1 | System Variables | 447 |
| 7.2.7.1.1 | Names | 447 |
| 7.2.7.2 | Telemetry | 448 |
| 7.2.7.2.1 | Data Link | 449 |
| 7.2.7.2.2 | Onboard Log | 451 |
| 7.2.7.2.3 | User Log | 451 |
| 7.2.7.2.4 | Fast Log | 452 |
| 7.2.7.2.5 | Compressing options | 453 |
| 7.2.7.3 | Sniffer | 456 |
| 7.2.7.4 | List of Variables | 458 |
| 7.2.7.4.1 | 32 VAR | 458 |
| 7.2.7.4.2 | BIT | 467 |
| 7.2.7.4.3 | 16 VAR | 469 |
| 7.2.8 | Panel | 471 |
| 7.2.8.1 | Sort Phases | 472 |
| 7.2.8.2 | Checklists | 472 |
| 7.2.8.3 | Confirmation | 475 |
| 7.2.8.4 | Notifications | 477 |
| 7.2.9 | HIL | 479 |
| 7.2.9.1 | Autopilot Configuration | 480 |
| 7.2.9.2 | Adding New variable | 481 |
| 7.2.9.3 | Veronte Pipe communications | 483 |
| 7.2.10 | Programs | 484 |

| | | |
|----------|--|------------|
| 7.2.10.1 | Control Blocks | 484 |
| 7.2.10.2 | Data Source/Sink Blocks | 490 |
| 7.2.10.3 | Execution Flow Blocks | 491 |
| 7.2.10.4 | Logic Blocks | 493 |
| 7.2.10.5 | Math Blocks | 493 |
| 7.2.10.6 | Mode/AP Selection Blocks | 496 |
| 7.2.10.7 | Position Blocks | 499 |
| 7.2.10.8 | Signal Blocks | 501 |
| 7.2.10.9 | Type Casting Blocks | 503 |
| 7.2.11 | Block PDIFs | 507 |
| 7.3 | Installation | 509 |
| 7.4 | License & Safe Mode | 510 |
| 7.4.1 | License | 510 |
| 7.4.2 | Safe Mode | 512 |
| 7.5 | Veronte LOS (Line Of Sight) | 513 |
| 7.5.1 | Microhard setup helper | 516 |
| 7.6 | Overview | 518 |
| 7.7 | Side Panel Options | 518 |
| 8 | Mission | 521 |
| 8.1 | Setup | 521 |
| 8.1.1 | Introduction | 521 |
| 8.1.2 | Terrain profile and magnetic field | 522 |
| 8.1.3 | Marks | 524 |
| 8.2 | Tools | 526 |
| 8.2.1 | New Waypoint | 526 |
| 8.2.2 | Segment | 528 |
| 8.2.3 | New Orbit | 529 |
| 8.2.4 | Intersect Lines | 529 |
| 8.2.5 | Fly By | 530 |
| 8.2.6 | Multiple Choice | 530 |
| 8.2.7 | Event Mark | 531 |
| 8.2.7.1 | Event creation and association | 533 |
| 8.2.8 | Obstacles | 534 |
| 8.2.9 | New Polygon | 535 |
| 8.2.10 | Circular Area | 539 |
| 8.2.11 | References | 539 |
| 8.2.12 | Mapping | 541 |
| 8.2.12.1 | Search & Rescue | 541 |
| 8.2.12.2 | Photogrammetry | 542 |
| 8.2.12.3 | Capsule Dropping | 548 |
| 8.2.13 | Import Route | 552 |
| 8.2.14 | Export Route | 553 |
| 8.2.14.1 | KML | 553 |
| 8.2.14.2 | Google Earth | 553 |
| 8.2.15 | Ruler | 554 |
| 8.2.16 | Remove unused points | 555 |
| 8.3 | Toolbar | 556 |
| 9 | Operation | 559 |
| 9.1 | Map | 559 |
| 9.2 | Telemetry | 561 |
| 9.3 | Operation Parameters | 563 |
| 9.4 | Mission | 567 |

| | | |
|--------------|---|------------|
| 9.4.1 | Loading a new mission | 569 |
| 9.4.2 | Modifying the current mission | 570 |
| 9.5 | Commands | 570 |
| 9.5.1 | Veronte Panel | 570 |
| 9.5.2 | Quick Commands | 573 |
| 9.5.3 | Additional Commands | 583 |
| 9.6 | Log Report | 586 |
| 10 | Post Flight | 591 |
| 10.1 | Tour Play | 591 |
| 10.2 | Data Export | 593 |
| 10.2.1 | Data link | 593 |
| 10.2.2 | LOG | 595 |
| 11 | Simulation | 599 |
| 11.1 | Professional HIL | 599 |
| 11.1.1 | Mounting | 599 |
| 11.1.1.1 | Cable Connection | 599 |
| 11.1.1.2 | Veronte Pipe Configuration | 600 |
| 11.1.1.3 | Autopilot Configuration | 600 |
| 11.1.1.4 | Veronte Pipe Communications | 603 |
| 11.1.2 | X-Plane 10 Settings | 603 |
| 11.1.2.1 | X-Plane 10 Configuration | 603 |
| 11.1.2.1.1 | Aircraft Model Installation | 604 |
| 11.1.2.1.2 | X-Plane 10 Setup | 604 |
| 11.1.2.2 | Low Performance Computer Configuration | 608 |
| 11.1.3 | X-Plane 11 Settings | 608 |
| 11.1.3.1 | X-Plane 11 Configuration | 608 |
| 11.1.3.1.1 | Aircraft Model Installation | 609 |
| 11.1.3.1.2 | X-Plane 11 Setup | 609 |
| 11.1.4 | Operation | 610 |
| 11.2 | SIL Simulink | 612 |
| 11.2.1 | Requisites | 613 |
| 11.2.2 | Autopilot Simulation | 614 |
| 11.2.3 | Sensors Simulation & Input Examples | 619 |
| 11.2.4 | Monitoring Telemetry with Simulink | 625 |
| 11.2.5 | Complete Simulation | 627 |
| 11.2.6 | Connecting Simulink & Veronte Pipe | 629 |
| 11.2.7 | Dealing with PDI files | 631 |
| 11.3 | 3D Simulation | 632 |
| 11.3.1 | Veronte Pipe Configuration | 633 |
| 12 | Examples | 637 |
| 12.1 | 4G Communication | 637 |
| 12.1.1 | 4G Communication with Veronte Autopilot | 637 |
| 12.2 | Configurations | 643 |
| 12.2.1 | Plataform Configurations | 643 |
| 12.2.1.1 | Fixed Wing-Mentor | 643 |
| 12.2.1.1.1 | Servo Configuration | 643 |
| 12.2.1.1.2 | Mission Phases | 646 |
| 12.2.1.1.2.1 | Takeoff | 646 |
| 12.2.1.1.2.2 | Climbing | 647 |
| 12.2.1.1.2.3 | Cruise | 649 |
| 12.2.1.1.2.4 | Hold | 651 |

| | | |
|--------------|-------------------------------|-----|
| 12.2.1.1.2.5 | Landing | 652 |
| 12.2.1.1.2.6 | Flare | 653 |
| 12.2.1.1.3 | Automations | 654 |
| 12.2.1.1.3.1 | Takeoff to Climb | 655 |
| 12.2.1.1.3.2 | Climbing to Cruise | 656 |
| 12.2.1.1.3.3 | Radio Error | 657 |
| 12.2.1.1.3.4 | Low Battery | 658 |
| 12.2.1.1.3.5 | Landing to Flare | 659 |
| 12.2.1.1.3.6 | Stick Auto | 660 |
| 12.2.1.2 | Flying Wing-W210 | 662 |
| 12.2.1.2.1 | Servo Configuration | 662 |
| 12.2.1.2.1.1 | Servos Output | 662 |
| 12.2.1.2.1.2 | SU Matrix | 664 |
| 12.2.1.2.1.3 | System Trim | 666 |
| 12.2.1.2.2 | Mission Phases | 668 |
| 12.2.1.2.2.1 | Standby | 668 |
| 12.2.1.2.2.2 | Hand-Launch | 669 |
| 12.2.1.2.2.3 | Catapult | 671 |
| 12.2.1.2.2.4 | Separation | 672 |
| 12.2.1.2.2.5 | Climbing | 673 |
| 12.2.1.2.2.6 | Cruise | 675 |
| 12.2.1.2.2.7 | Hold | 677 |
| 12.2.1.2.2.8 | Landing | 677 |
| 12.2.1.2.2.9 | Flare | 679 |
| 12.2.1.2.3 | L200 Catapult | 681 |
| 12.2.1.2.3.1 | Configuration | 681 |
| 12.2.1.2.3.2 | Actuator Trimming | 681 |
| 12.2.1.2.3.3 | Launch Automation | 682 |
| 12.2.1.3 | Quadcopter-M400 | 686 |
| 12.2.1.3.1 | Servo Configuration | 686 |
| 12.2.1.3.1.1 | Servos Output | 686 |
| 12.2.1.3.1.2 | SU Matrix | 688 |
| 12.2.1.3.1.3 | System Trim | 689 |
| 12.2.1.3.2 | Mission Phases | 690 |
| 12.2.1.3.2.1 | Standby | 690 |
| 12.2.1.3.2.2 | Takeoff | 690 |
| 12.2.1.3.2.3 | Hover | 692 |
| 12.2.1.3.2.4 | Cruise | 694 |
| 12.2.1.3.2.5 | Landing | 696 |
| 12.2.1.3.2.6 | VHOLD | 698 |
| 12.2.1.3.2.7 | Return to Home | 699 |
| 12.2.1.4 | Helicopter T-Rex600 | 703 |
| 12.2.1.4.1 | Servo Configuration | 703 |
| 12.2.1.4.1.1 | Servos Output | 703 |
| 12.2.1.4.1.2 | SU Matrix | 705 |
| 12.2.1.4.1.3 | System Trim | 707 |
| 12.2.1.4.2 | Mission Phases | 709 |
| 12.2.1.4.2.1 | Stanby | 709 |
| 12.2.1.4.2.2 | Engine Off | 710 |
| 12.2.1.4.2.3 | Ignite | 713 |
| 12.2.1.4.2.4 | Ready | 715 |
| 12.2.1.4.2.5 | Takeoff | 717 |
| 12.2.1.4.2.6 | Hover | 718 |
| 12.2.1.4.2.7 | Cruise | 719 |

| | | | |
|-----------|--------------|--|------------|
| | 12.2.1.4.2.8 | Landing | 721 |
| | 12.2.1.4.2.9 | Hold | 722 |
| | 12.2.1.5 | Hybrid | 725 |
| | 12.2.1.5.1 | Servo Configuration | 725 |
| | 12.2.1.5.1.1 | Quadcopter servos | 725 |
| | 12.2.1.5.1.2 | Servos Output | 725 |
| | 12.2.1.5.1.3 | SU Matrix | 727 |
| | 12.2.1.5.1.4 | Plane Servos | 728 |
| | 12.2.1.5.1.5 | Servos Output | 728 |
| | 12.2.1.5.1.6 | System Trim | 730 |
| | 12.2.1.5.2 | Mission Phases | 735 |
| | 12.2.1.5.2.1 | Standby | 735 |
| 12.3 | | Automations | 738 |
| 12.3.1 | | Veronte Panel Buttons | 738 |
| 12.3.1.1 | | Phase Change Buttons | 738 |
| 12.3.1.2 | | Generic Button | 740 |
| 12.3.2 | | Phase Changing | 743 |
| 12.3.2.1 | | Takeoff to Climb Change | 743 |
| 12.3.2.2 | | Climbing to Cruise Change | 744 |
| 12.3.2.3 | | Landing to Flare Change | 745 |
| 12.3.3 | | Failsafe | 746 |
| 12.3.3.1 | | Radio Error | 747 |
| 12.3.3.2 | | Low Battery | 749 |
| 12.3.3.3 | | GPS Signal lost | 749 |
| 12.3.4 | | Photogrammetry | 750 |
| 12.3.5 | | Others | 752 |
| 12.3.5.1 | | Stick Auto | 752 |
| 12.3.5.2 | | Automatic Landing (Multicopterss) | 753 |
| 12.4 | | Veronte Tracker | 755 |
| 12.4.1 | | Veronte Ground Configuration | 755 |
| 12.4.2 | | Veronte Air Configuration | 762 |
| 12.4.3 | | Tracking Operation | 763 |
| 12.5 | | Autotune | 767 |
| 12.5.1 | | Modes Configuration | 768 |
| 12.5.2 | | Workspace Configuration | 769 |
| 13 | | Troubleshooting | 773 |
| 13.1 | | Maintenance mode | 773 |
| 13.1.1 | | Entering maintenance mode | 774 |
| 13.1.2 | | Forcing maintenance mode | 775 |
| 13.1.3 | | Maintenance mode options | 775 |
| 13.1.3.1 | | Exit maintenance mode | 775 |
| 13.1.3.2 | | Remove Encryption | 775 |
| 13.1.3.3 | | Change setup | 775 |
| 13.1.3.4 | | Update | 775 |
| 13.1.3.5 | | Retry | 775 |
| 13.1.4 | | SD Loaded With errors | 776 |
| 13.1.4.1 | | Check PDI | 776 |
| 13.1.4.2 | | Try load PDI | 776 |
| 13.1.5 | | Communications failure | 776 |
| 13.2 | | Veronte can't be detected through USB port | 777 |
| 13.2.1 | | Check the COM port | 777 |
| 13.2.2 | | Check Pipe/Veronte Link | 777 |
| 13.2.3 | | Force Maintenance mode | 778 |

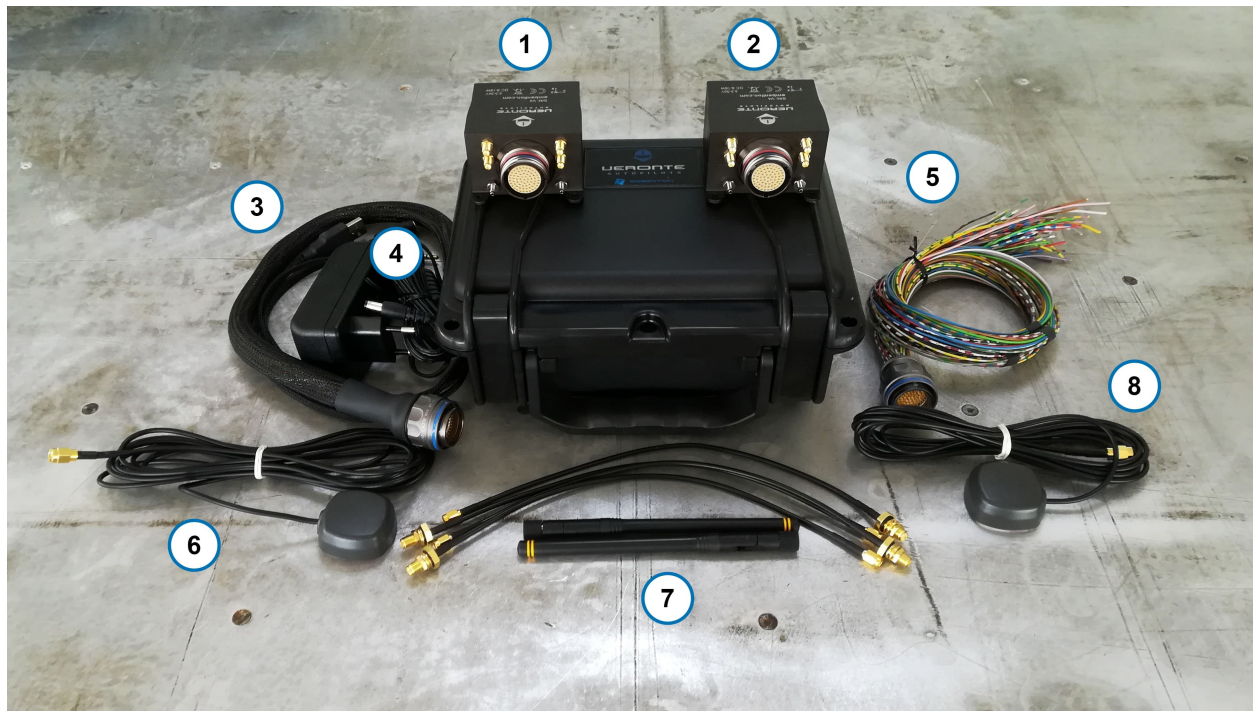
| | | |
|-----------|---|------------|
| 13.3 | Licensed Expired | 778 |
| 13.4 | Out of Georeferenced Area | 779 |
| 14 | Acronyms & Definitions | 781 |
| 14.1 | Acronyms | 781 |
| 14.2 | Definitions | 782 |
| 15 | Axes Convention | 783 |
| 16 | Feedback | 785 |
| 16.1 | Feedback Panel | 785 |
| 17 | Compatible devices | 787 |
| 17.1 | Servos and ESC | 787 |
| 17.1.1 | PWM Servos | 787 |
| 17.1.2 | Serial servos | 788 |
| 17.1.2.1 | Servo Volz DA26 | 789 |
| 17.2 | Altimeters | 793 |
| 17.2.1 | Reading altimeter measurement | 794 |
| 17.2.1.1 | ADC lidar | 794 |
| 17.2.1.2 | I2C lidar | 795 |
| 17.2.1.3 | CAN Radar | 795 |
| 17.2.2 | Using altimeter readings | 800 |
| 17.3 | External sensors | 801 |
| 17.3.1 | Magnetometer Honeywell HMR2300-232 | 801 |
| 17.4 | USB devices | 805 |
| 17.4.1 | Adding a USB joystick | 805 |
| 17.4.2 | Defining a virtual stick | 806 |
| 17.5 | Silvus radio (StreamCaster 4200E model) | 808 |
| 17.5.1 | System Layout | 808 |
| 17.5.2 | Hardware Installation | 809 |
| 17.5.3 | Silvus radio configuration | 810 |
| 17.5.3.1 | First Steps | 810 |
| 17.5.3.2 | Basic radio configuration | 816 |
| 17.5.4 | Silvus radio configuration in Veronte Pipe | 822 |
| 18 | 4x Redundant Autopilot - Arbiter | 831 |
| 18.1 | Arbitration | 831 |
| 18.1.1 | Scoring System | 831 |
| 18.1.1.1 | Absolute Arbitration Variables | 832 |
| 18.1.1.2 | Relative Arbitration Variables | 832 |
| 18.1.2 | Arbitration modes & parameters | 832 |
| 18.1.2.1 | Arbitration modes | 833 |
| 18.1.2.2 | Arbitration parameters | 833 |
| 18.2 | Operation | 833 |
| 18.2.1 | Arbiter Boot | 833 |
| 18.2.2 | Status message | 834 |
| 18.2.3 | Ready Status | 834 |
| 18.2.4 | Alive Status | 834 |
| 18.2.5 | Maintenance Mode | 835 |
| 18.3 | Software | 835 |
| 18.3.1 | Veronte Link | 835 |
| 18.3.2 | 4xVeronte PDIBuilder | 836 |
| 18.4 | Update | 838 |
| 18.4.1 | Updating 4xArbiter from Firmware version older than 6.0.0 | 840 |

| | | |
|-----------|-----------------------------------|------------|
| 18.5 | Configuration | 840 |
| 18.5.1 | Arbitration | 841 |
| 18.5.1.1 | Config | 842 |
| 18.5.1.2 | CAN Config | 843 |
| 18.5.2 | I/O Manager | 844 |
| 18.5.2.1 | Producers | 844 |
| 18.5.2.2 | Consumers | 844 |
| 18.5.3 | CAN I/O Manager | 845 |
| 18.5.3.1 | CAN - Producers | 845 |
| 18.5.3.2 | CAN - Consumers | 846 |
| 18.5.4 | SCI | 846 |
| 18.5.5 | CAN | 846 |
| 18.5.6 | PORTS | 846 |
| 18.5.7 | GPIO | 846 |
| 18.6 | CAN Protocol | 846 |
| 18.6.1 | From Arbiter | 847 |
| 18.6.1.1 | Status Message | 847 |
| 18.6.1.2 | Score Message | 847 |
| 18.6.2 | From Veronte | 848 |
| 18.6.2.1 | Ready Message | 848 |
| 18.6.2.2 | Arbitration Messages | 848 |
| 19 | 4x Redundant Autopilot | 849 |
| 19.1 | Introduction | 849 |
| 19.2 | Setup | 850 |
| 19.2.1 | CAN bus | 850 |
| 19.2.1.1 | Variable value message: | 850 |
| 19.2.1.2 | Status message: | 850 |
| 19.2.1.3 | Variable value message: | 850 |
| 19.2.1.4 | Status message: | 851 |
| 19.2.2 | Control | 851 |

QUICK START

1.1 First Steps

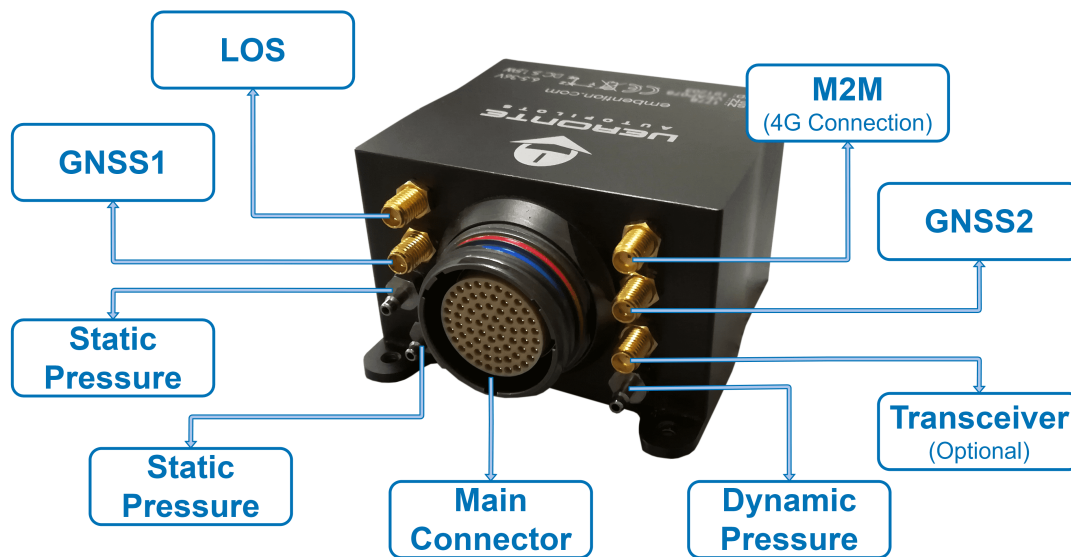
This section provides basic information of the connections of the autopilot and how to have it ready-to-use connected to the provided software **Veronte Pipe**.



Autopilot Kit

1.1.1 Kit Elements

1. Veronte Autopilot
2. Veronte Autopilot
3. Autopilot Harness CS, including USB & Joystick connectors and 50cm wire leads
4. 12V Power Supply (European plug)
5. Autopilot Harness including 50cm wire leads
6. GNSS Antenna (Standard, SSMA Male connector)
7. 2x LOS Antenna SMA Male connector + SMA/SSMA adapters
8. GNSS Antenna (Standard, SSMA Male connector)



Veronte 4.5 Connections

For more information about the autopilot's hardware (vibration isolation, orientation and location on the platform, pressure lines, etc) check [Hardware Installation](#).

1.1.2 Veronte Pipe

Once the Veronte is delivered, a shared folder between the Customer and Embention is automatically created. The user will receive an email from the Support Team containing the information needed to access. If the email is not received within 72h, please contact with support@embention.com and our Support Team will be happy to help you.

Sign in

<http://support.embention.com>

Your connection to this site is not private


Username

Password

Sign in

Cancel

FTP Access

 EMBENTION

[Products](#) [Services](#) [Projects](#) [Company](#) [HIL Broadcast](#) [News](#) [Contact](#)

FILES

Update

5.38.8 update

Veronte Pipe

VerontePipe_x64-v5.38.16.exe

Autopilots • Veronte • NMS • UAS / RPAS


Privacy • Work with us


Email [Subscribe](#)

Pl Las Alcañizas, C/ Chelín, 16, 03114

Alicante (Spain) | +34 965 115 421

embention@embention.com

 VERONTE

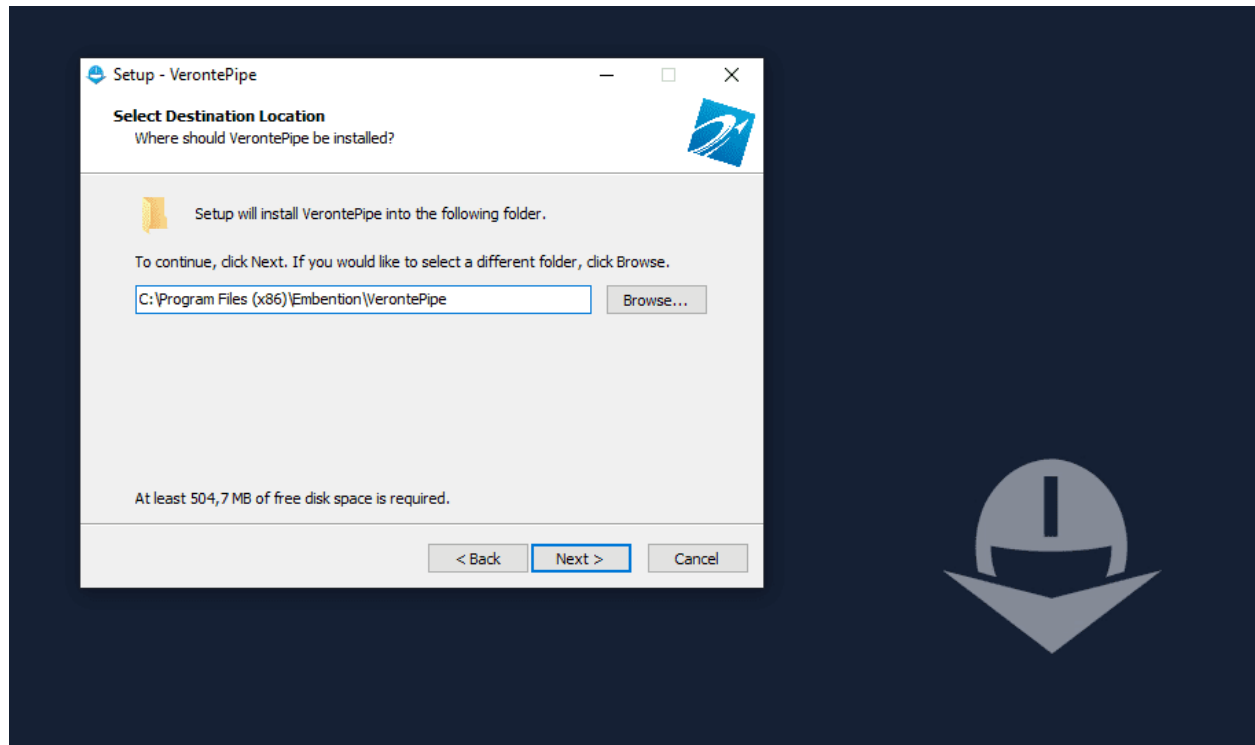
 NMS

AUTOPILOTS

AIRCRAFT

FTP folder content

The shared folder contains the last version of Veronte Pipe and firmware. Please, download and install the executable file provided. Veronte units are normally shipped with the last version of firmware.

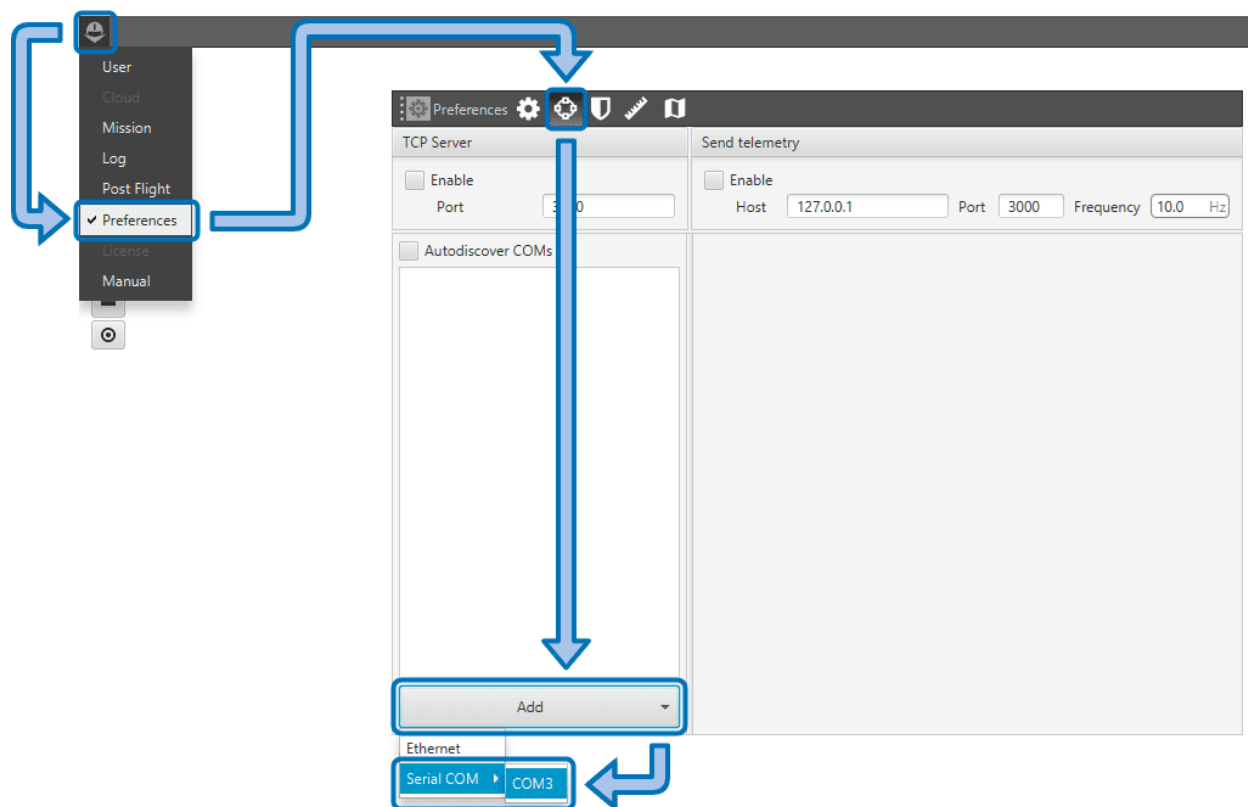


Pipe Installation

For more information about Veronte Pipe intallation process, system requirements and OS, please check the [Pipe Configuration](#) section.

Once Veronte Pipe is installed it is necessary to configure the connection to the autopilot.

Click on Main Menu and then select Preferences in the pull-down menu. Now click on Add -> Serial COM. If only one unit is connected to the computer via USB, only one COM port will be available.



Connections Menu


This step is only performed once with every new unit. Once the configuration is done, the Veronte autopilot will be automatically detected by Veronte Pipe and it will show up in the right bar.

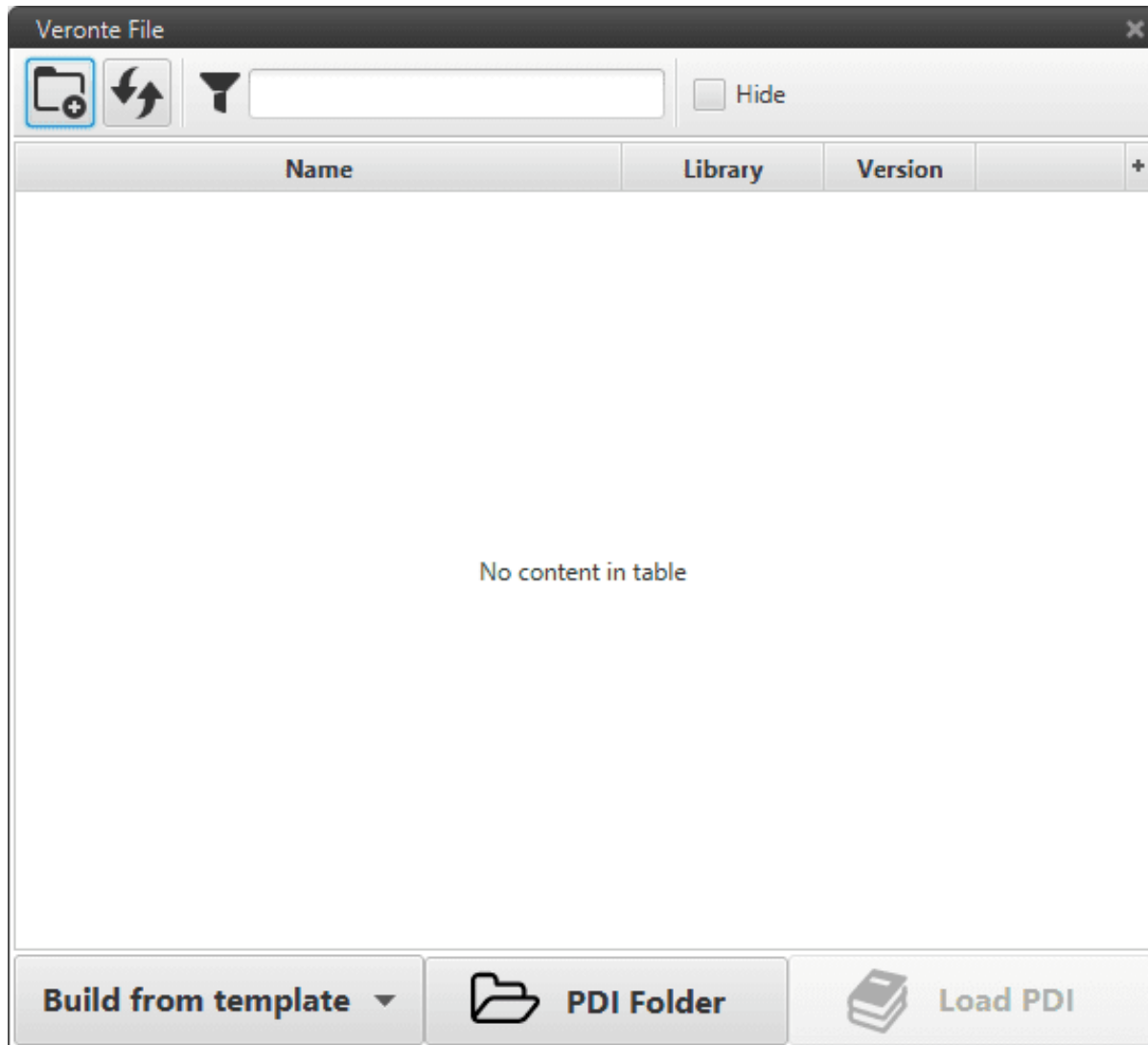
For more information about the Connection's menu check [Connections](#).

1.2 Model Simulation

This section will allow the user to load a pre-existing model/configuration to the autopilot and perform a mission simulation with the combined help of the software X-Plane. The content is organized in 3 sub-sections: how to load the template into Veronte autopilot, setting XPlane and Veronte Pipe so they can communicate, and how to run a simulation.

1.2.1 Loading a Template to the Autopilot

When the autopilot is connected, in the side panel click on  and then click on Import PDI.



Import a Configuration File

The **Import Configuration** option displays a window (shown in the previous figure) where the user can select a file to be uploaded. The option of interest for the scope of this section is **Build from template**. The software is fitted with a set of default configurations (templates) so the user can have reference of what a fixed wing, quadcopter or ground configuration should look like.

There are three other options to select a configuration file. For more information about the logical calibration of the servos check *Import Configurations*.

The configuration file is loaded in the software (Veronte Pipe), but it is not loaded on the autopilot until the Save button




is pressed.

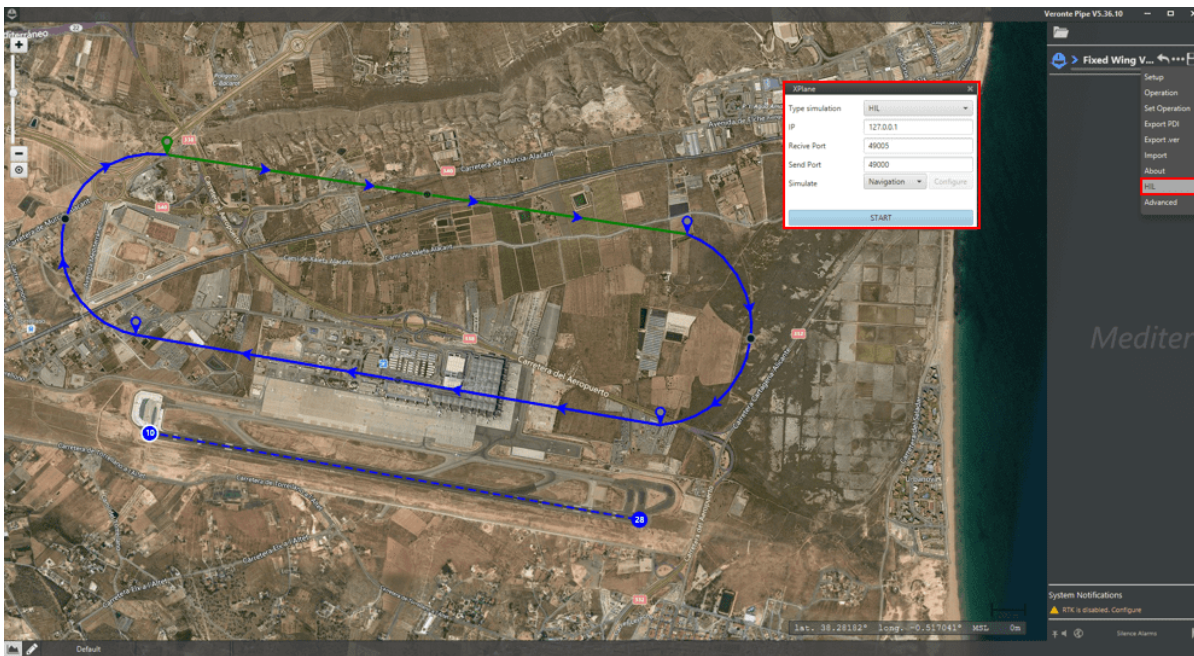
1.2.2 X-Plane and Veronte Pipe Setup

Integration of Veronte Autopilot within the X-Plane simulator enables the user to perform a Hardware-In-the-Loop (HIL) simulation. HIL simulation enables testing an effective model of the vehicle by adding the complexity of the plant under control.

1.2.2.1 Veronte Pipe

Once a template has been loaded into the autopilot, the HIL simulation parameters need to be set. On the side panel, click on  and then click on HIL. A pop-up window will appear (see the figure below) and it has to be filled in as follows:

| | |
|--------------|-----------|
| IP | 127.0.0.1 |
| Receive Port | 49005 |
| Send Port | 49000 |



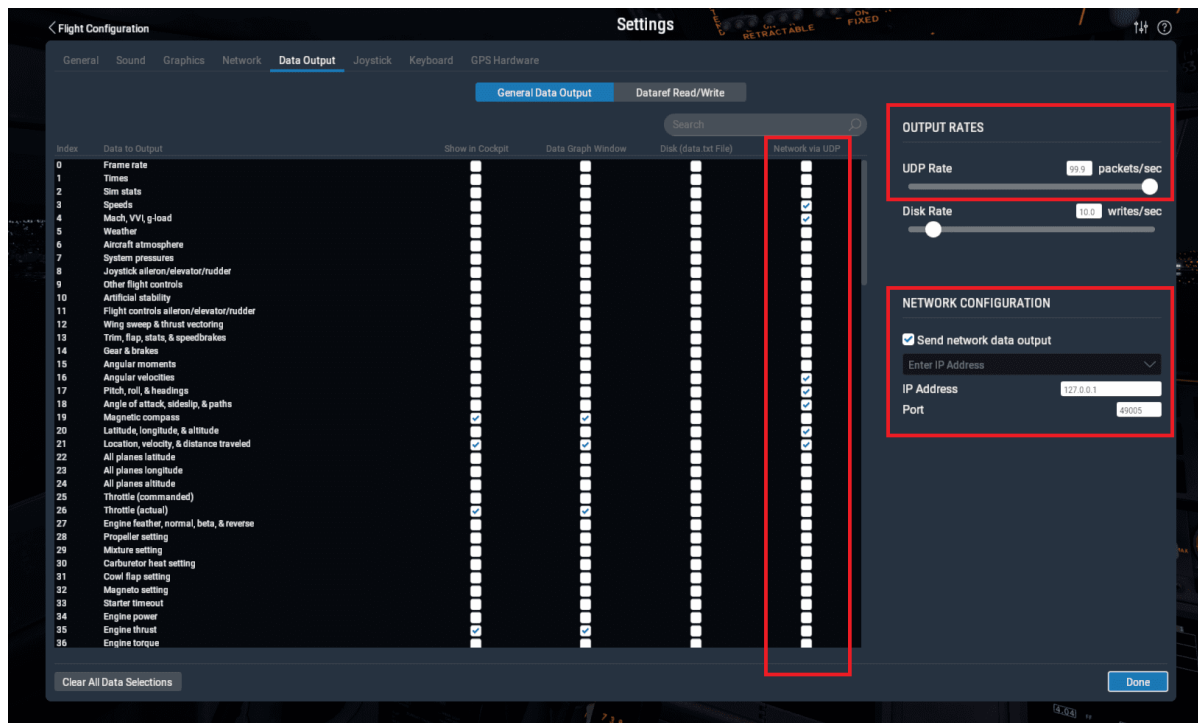
Veronte Pipe – HIL Configuration

1.2.2.2 X-Plane

X-Plane 10 or 11 are compatible with Veronte Pipe to perform HIL simulations. Here, necessary settings on X-Plane 11 to communicate with Veronte Pipe are detailed. If X-Plane 10 is used, the user can check [X-Plane 10 Settings](#).

On X-Plane's main menu, go to Settings and then to Data Output. Here, the data that will be shared with Veronte Pipe is defined:

- Variables 3, 4, 16, 17, 18, 20 and 21 should be selected on the column *Network via UDP*.
- UDP rate should be set to its maximum value of 99 packets/sec.
- Network configuration should be set to *IP Address 127.0.0.1 and Port 49005*.



X-Plane 11 I/O configuration

X-Plane 11 provides a tool called Plane Maker where the user can create the platform model. Once the vehicle model has been created, it can be integrated on the X-Plane 11 simulator by following next steps:

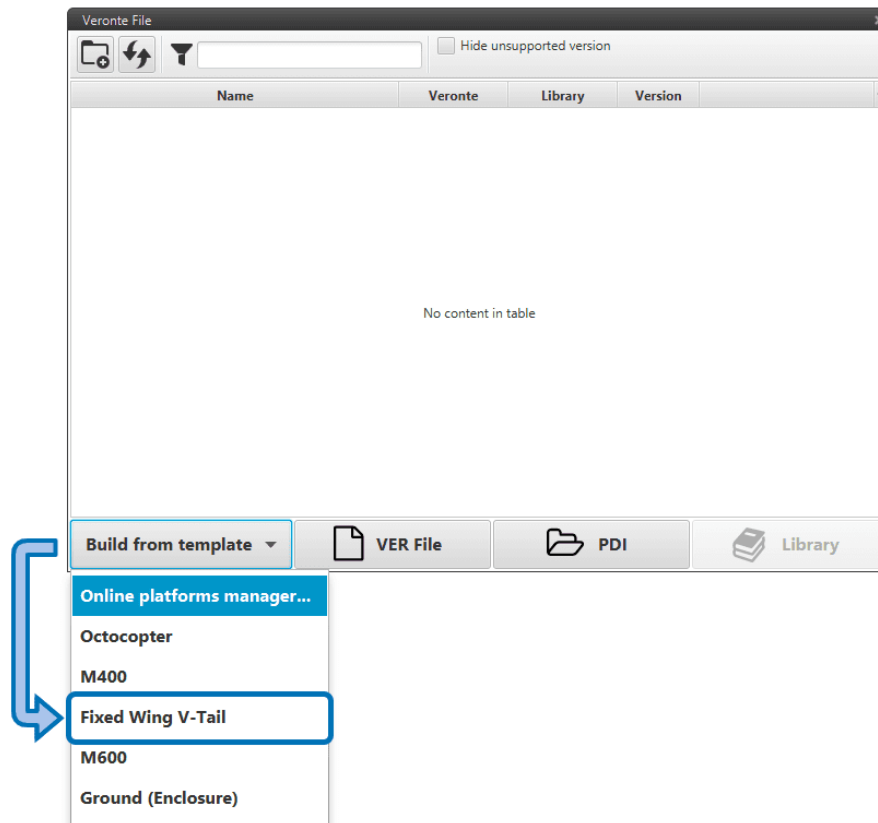
- Copy the model folder to the *Aircraft* folder within the X-Plane 11 installation directory.
- Copy the content of the *Airfoils* folder available on the aircraft model folder to the *Airfoils* directory within the X-Plane 11 installation directory.

1.2.3 Executing a Simulation

Before starting a simulation with one of the built-in templates of Veronte Pipe, the user needs to take into account the following:

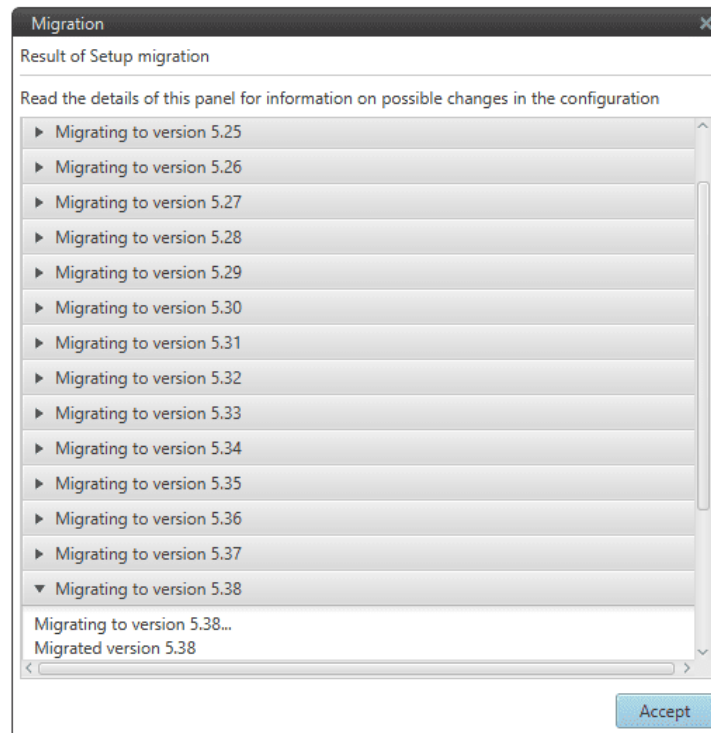
- The missions defined on this configurations are set at Alicante's airport (LEAL), with runway 10 as the default one.
- The flight envelope and all the flight control settings and transitions are tailored for the template model. Therefore, the user might need to modify some parameters in order to have their own model flying as desired.

Let us suppose the user has created a fixed wing model with a V-tail configuration on Plane Maker. Then, the most suitable template on Veronte Pipe would be Fixed Wing V-Tail:



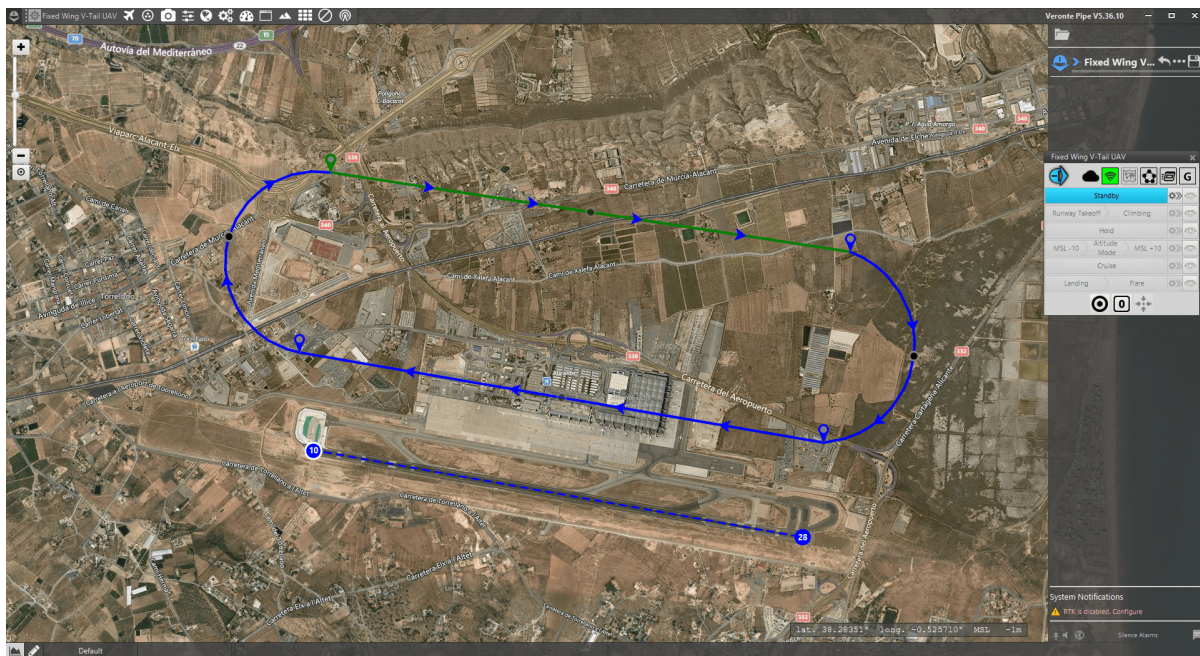
Fixed wing V-Tail template

A configuration merge and Migration windows will pop-up (see figure below). The user should accept both of them as they are. For more information on this topic go to [Import Configurations](#).



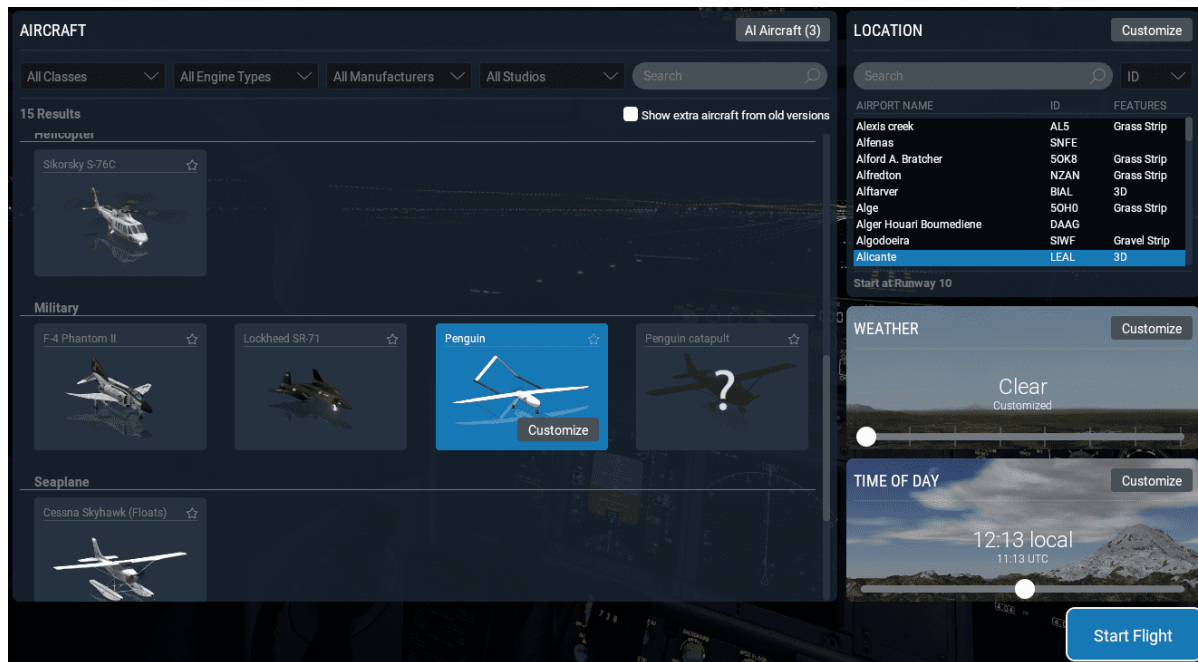
Pop-up window to end the import process

The Fixed Wing V-Tail template mission should look like this:



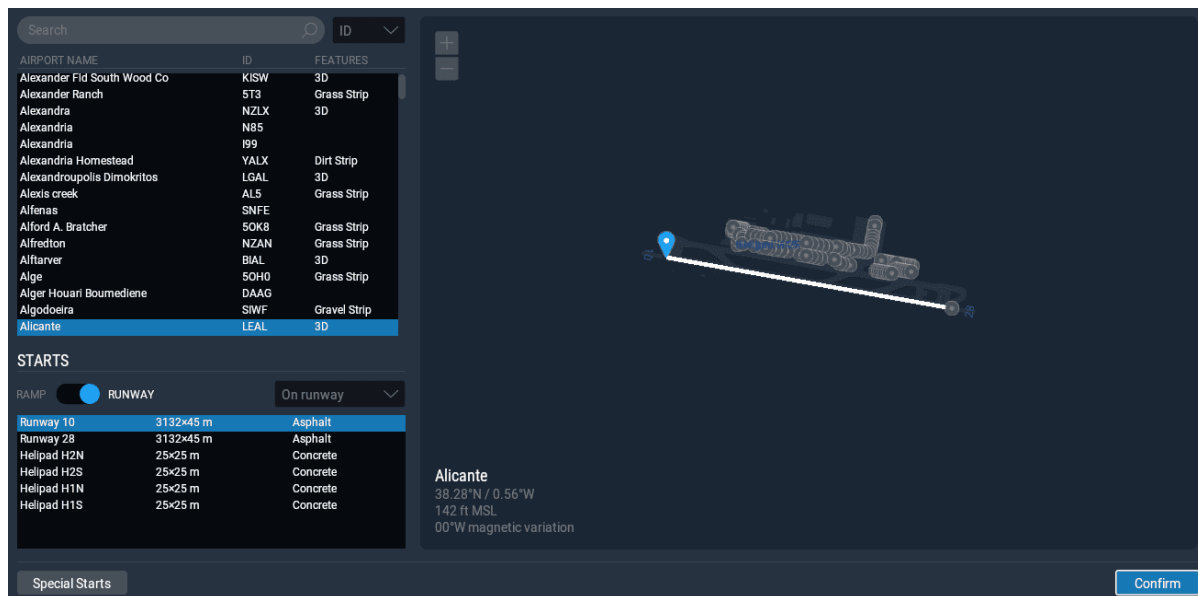
Template's mission

Now, the user needs to open X-Plane 11 and click on New Flight. On that menu the user will select the custom-made model and Alicante as the desired location (see the figure below):




X-Plane 11: new flight

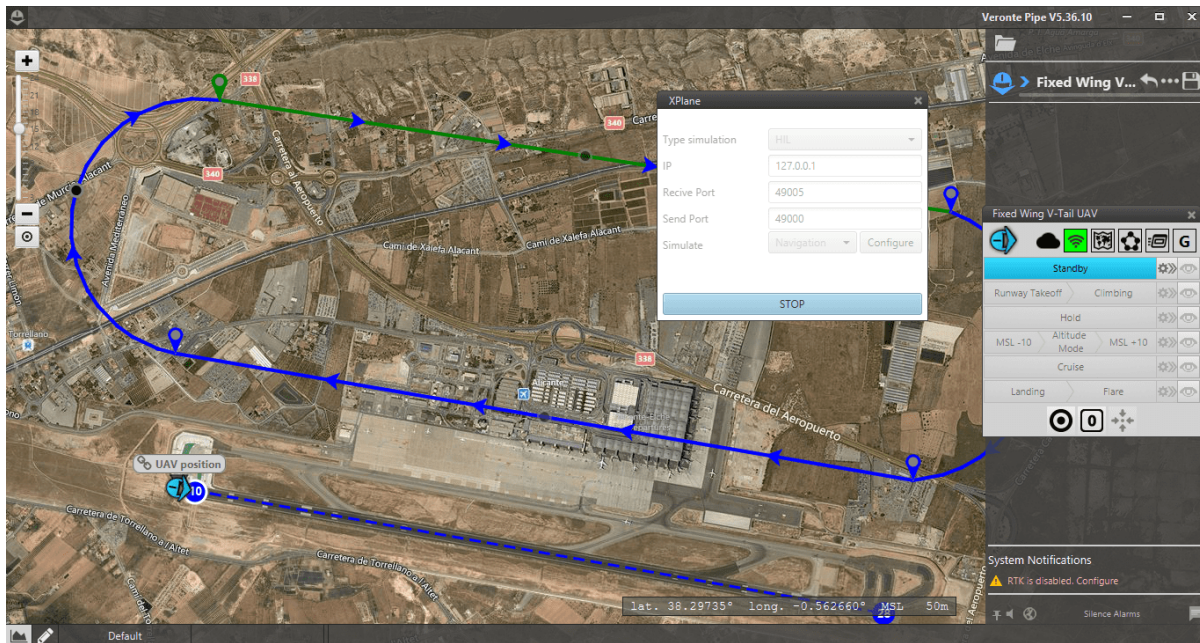
One should click on *Customize* in order to select the desired runway for take-off (see the figure below):



X-Plane 11: runway selection

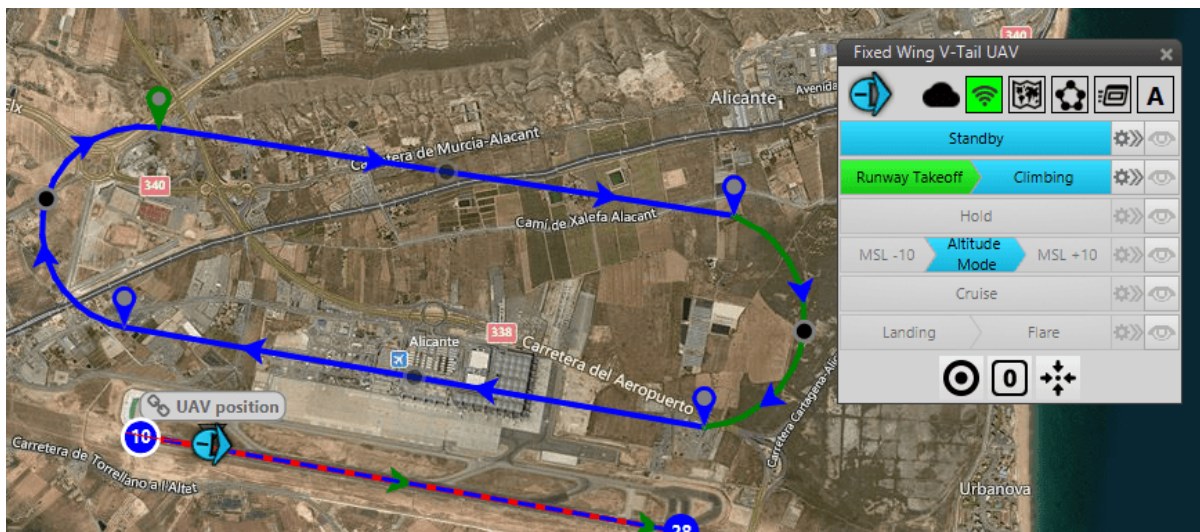
After that the flight can already be started (click on *Start Flight*). The aircraft will appear at the header of runway 10 of Alicante's airport. Once that is done, connection between X-Plane 11 and Veronte Pipe can be established. Go to Pipe,

click on , then HIL and click on Start. Autopilot must be in the initial phase. On the contrary, click on Advanced and then on Restart Veronte. If every step was correctly done, GPS is simulated (from X-Plane) and then the UAV must be visible on Veronte Pipe map.



Autopilot visible on Veronte Pipe

Now the user can pass the system to Standby phase to start the flight. To do so click on the panel and the word *Standby* will turn green. After that, click on *Runway Takeoff* and the aircraft will carry on autonomously until reaching a Cruise phase. The transition between phases is defined in the Automations menu. For more information regarding that subject check [Automations](#).



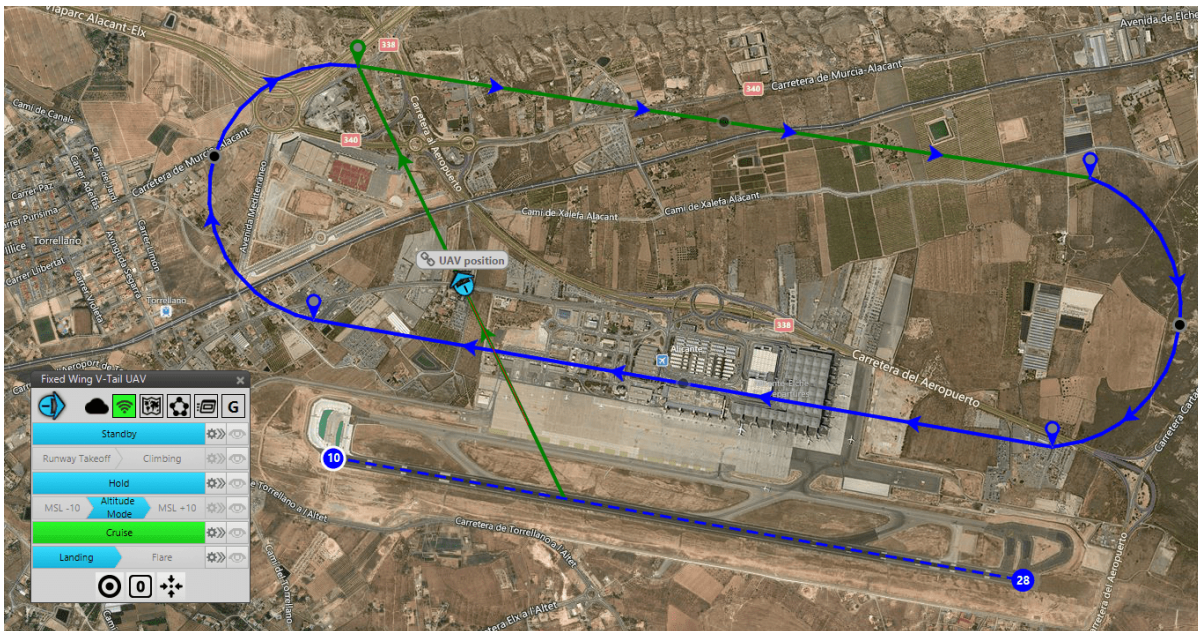
Take-off phase

The transition from take-off phase to climbing phase happens when the Indicated Air Speed (IAS) is greater than 20 m/s. The climbing phase (see figure below) automatically traces a climbing path.



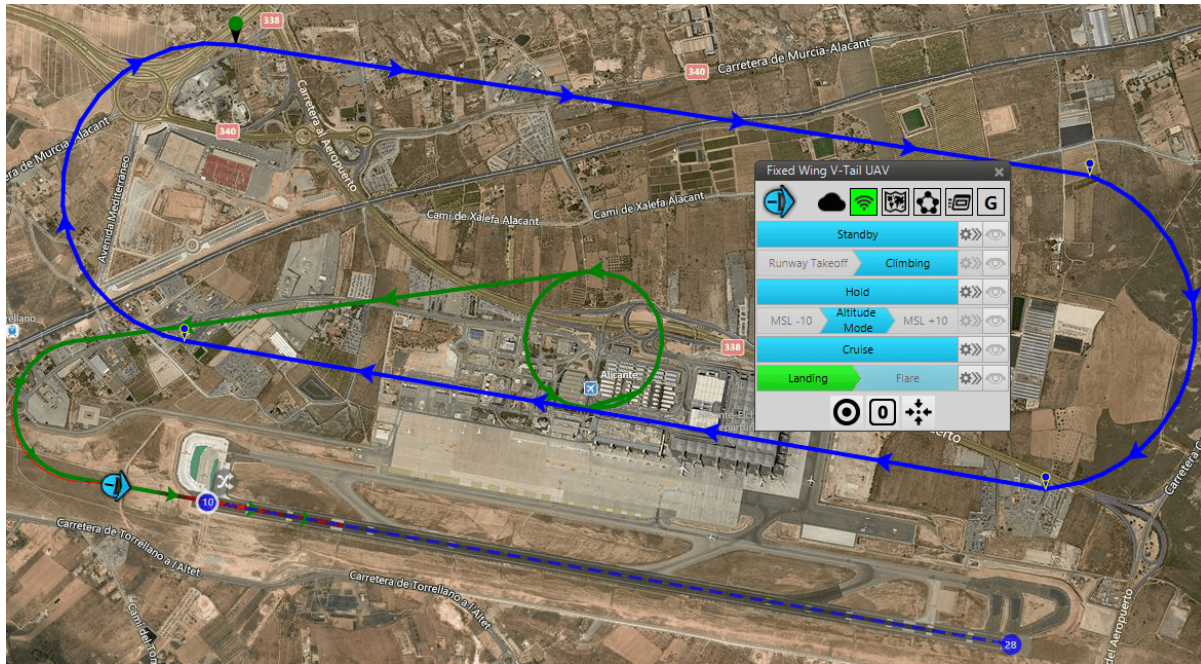
Climbing phase

But transitioning to cruise phase happens when only one condition is met: above ground level (AGL) altitude of more than 100 m.



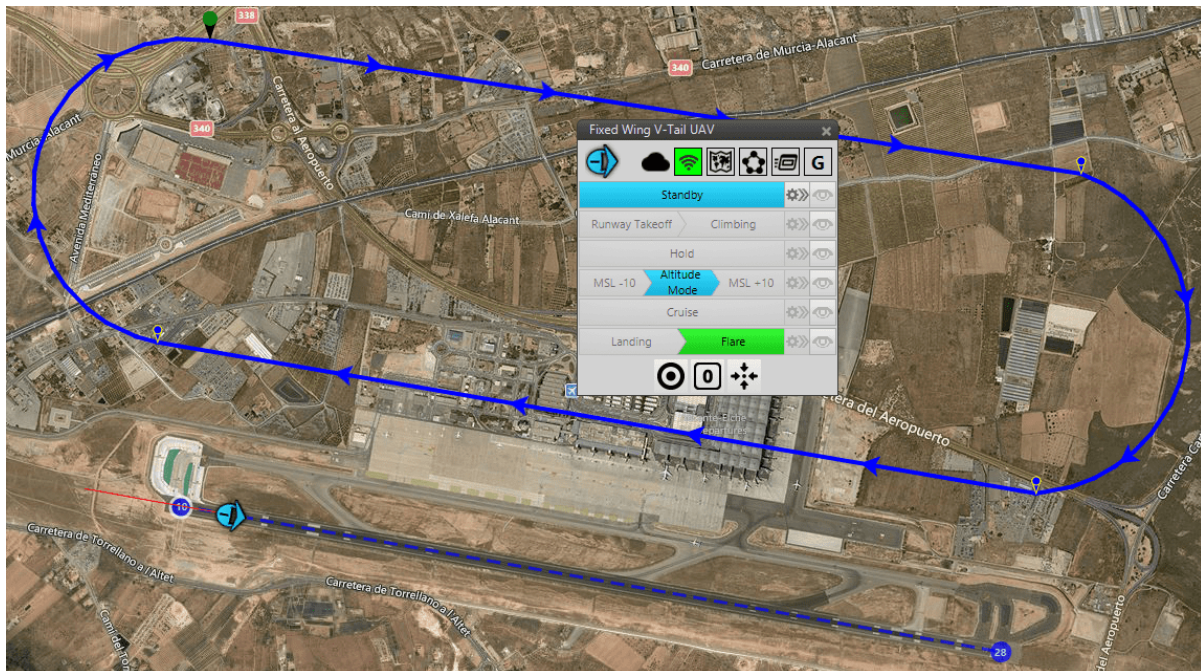
Cruise phase

Once in cruise mode the aircraft will follow the mission displayed above in blue, being in green the present route. Unless you define an event to motivate the entry of landing phase (a certain flight time, a certain altitude, etc), this template does not have that part automated. Upon clicking on *Landing*, a landing path is automatically traced.



Landing phase

The condition to change from landing to flare phase is double: having an AGL smaller than 10 m **and** having passed by a waypoint. For more information on mission planning/design check [Mission](#) chapter.



Flare phase

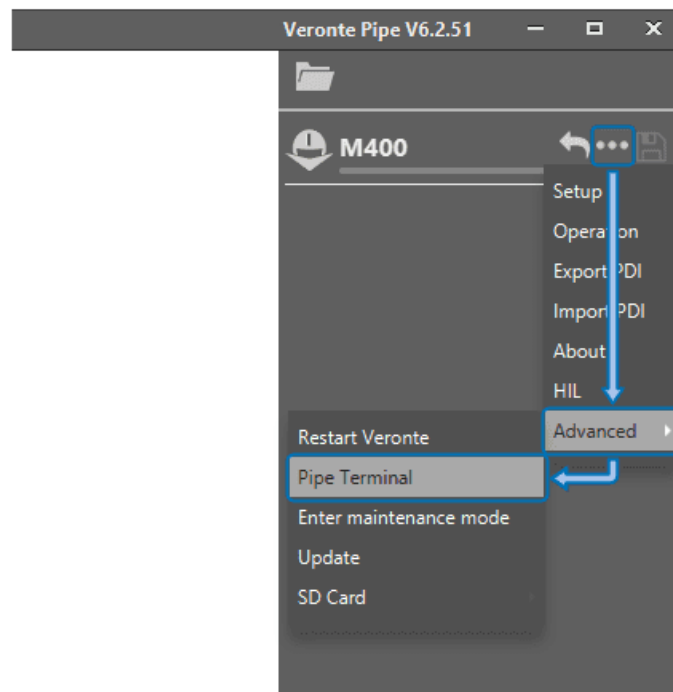
After that, the aircraft will have successfully landed and gone back to Standby phase. If the user need to adapt the flight phases to it's model, one needs to go to the **Control** menu, and then go to **Phases**. There, parameters as take-off speed, cruise speed, among others, can be modified. For more information please check [Phases](#) definition.

All the steps above should serve as a starting point for the user to become more acquainted with HIL simulations. With the latter he or she will improve the definition and operation of the mission and the performance and control of the


platform.

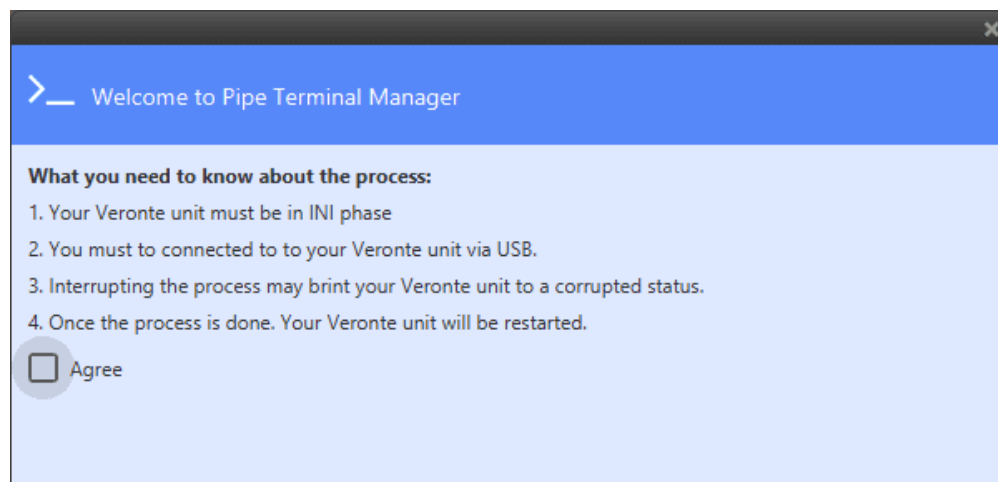
1.3 Radio Pairing (LOS)

This panel allows the user to easily access to the Pipe Terminal and modify the desired radio module settings. In the case of Veronte 4.0 and 4.5, they are integrated with Microhard Pico Series internal radio modules and their configuration can be modified by using the Microhard Setup Helper.

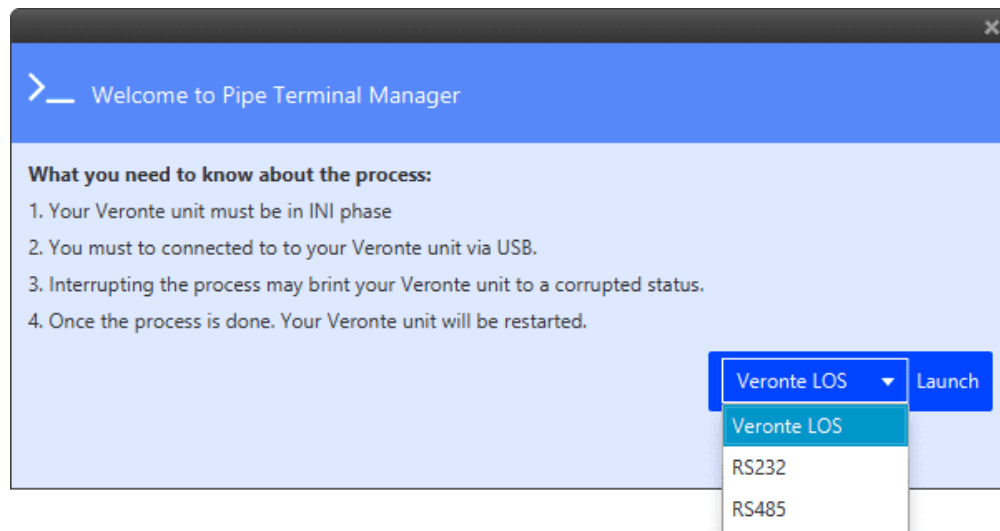


Veronte Pipe Terminal Setup

In order to start the configuration, the user should open the Setup by clicking on , then on Advanced and finally on Pipe Terminal. After this, the following window will show up.



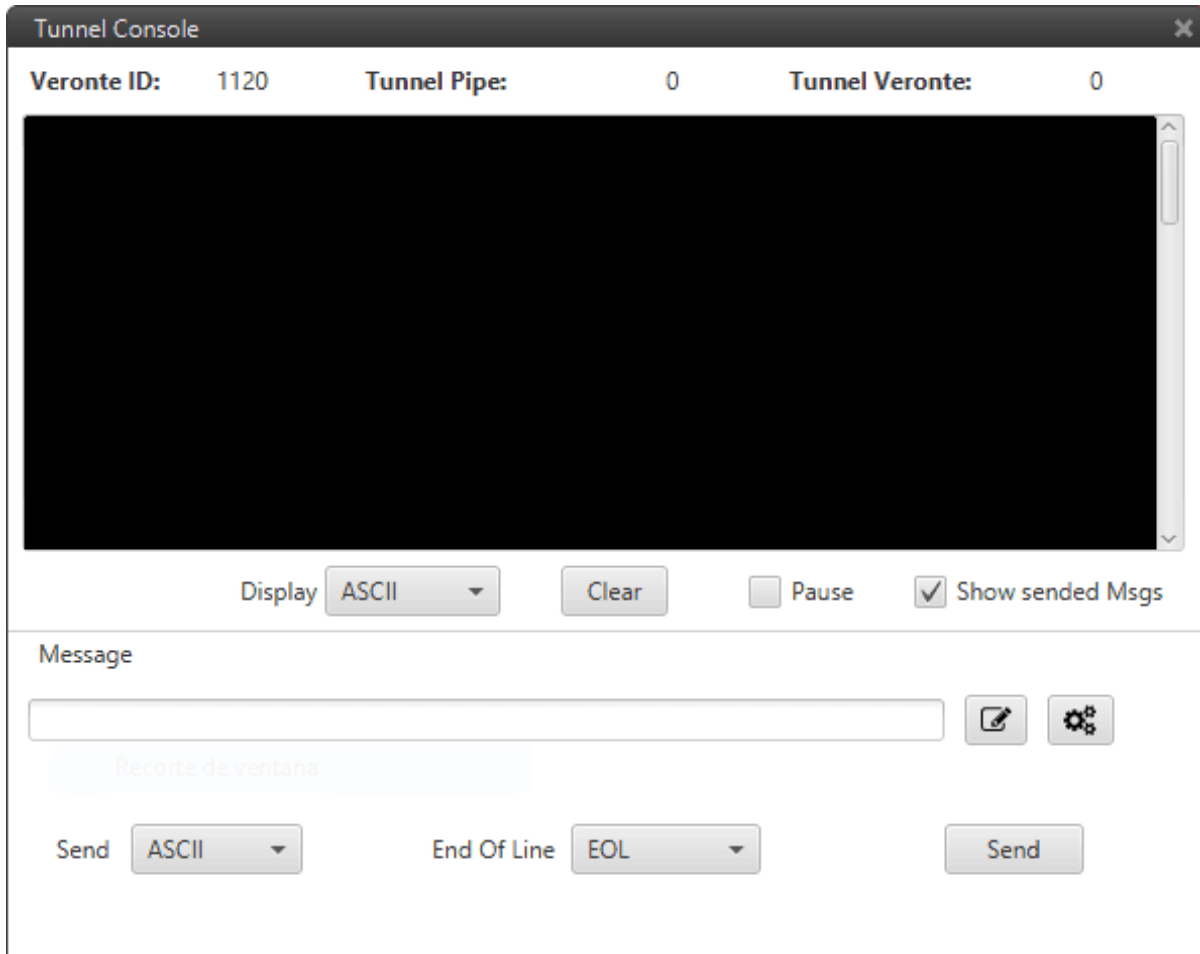
Veronte Pipe Terminal Manager agreement



Serial selection

Veronte Pipe is able to automatically set up the required tunnel over LOS, RS232 or RS485 depending on the user configuration and needs. In case of internal radio module configuration, the LOS option must be selected. If some external module is plugged over some external serial port, then RS232 or RS485 option must be used.

Once the process is launched, the Tunnel Console shows up.



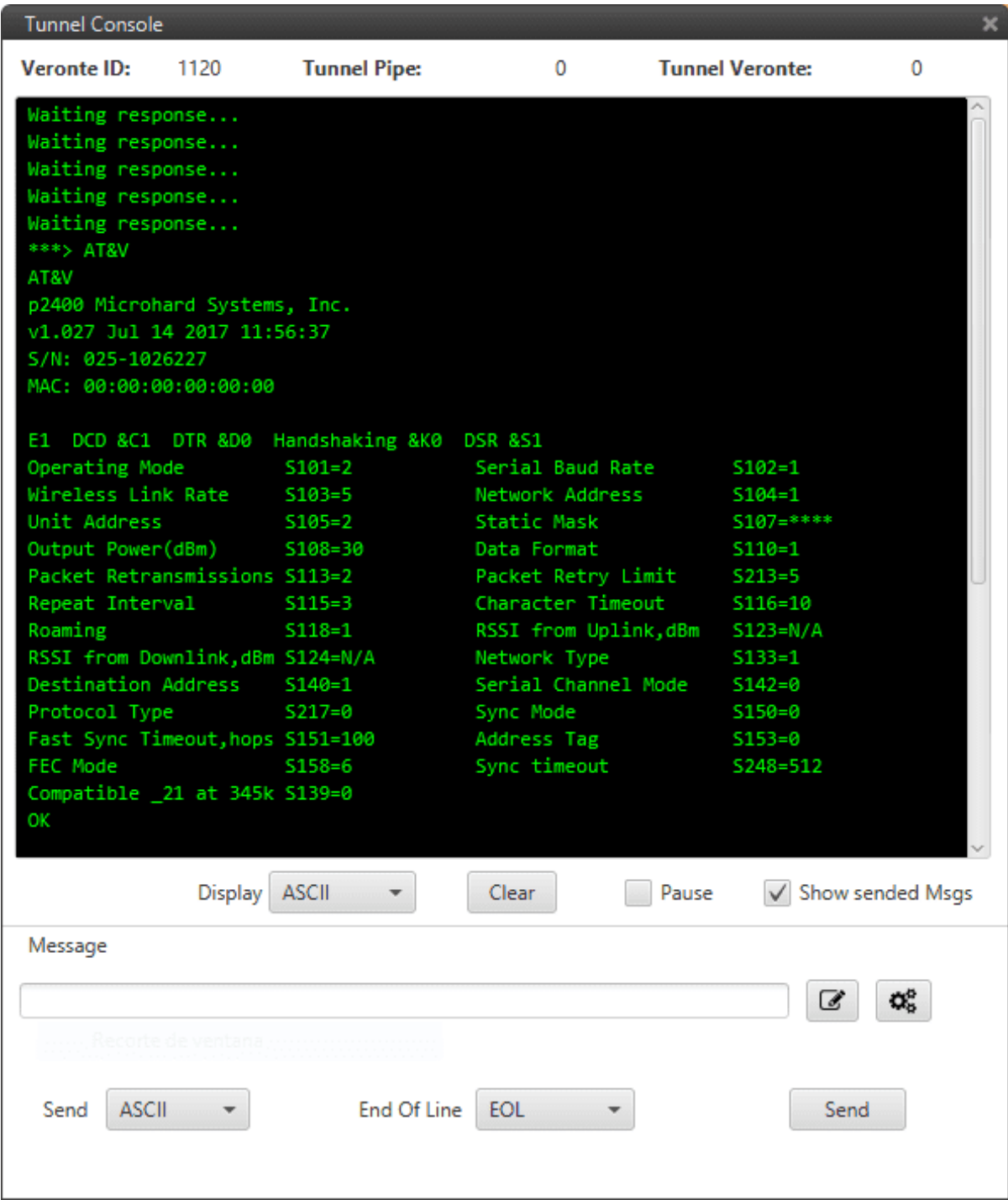
Tunnel Console

This console allows the user to communicate with the desired device (internal LOS or external device) by configuring the listed parameters:

- **Display:** Data can be displayed in Ascii or Hexa format.
- **Clear:** Clears the last console data.
- **Pause:** When the data stream is running, it is possible to pause it by enable this option.
- **Show sent messages:** If enabled, the sent messages are shown in the console.
- **Message:** Message content.
- **Microhard setup helper:** Check the related sub-section below.
- **Send:** Data sent in Ascii or Hexa format.
- **End Of Line:** Message “End of line” configuration.
- **Send:** Sending button. When clicked, the current message is sent.

1.3.1 Microhard setup helper

The **Microhard setup helper** can be used in order to easily configure the Microhard internal modules. In order to do this, the user should click on the dedicated button and the Wizard will automatically start the radio searching.



Terminal Console - Radio found

Once the radio is found, its model is shown in the upper-left corner of the window and the configured parameters in the console. The following Wizard menu is displayed.

Radio Wizard

Radio model **P2400**

Radio configuration

Out power: 30 dBm (1W)

Address: Master

Connection: PP

Network address: 1

Packet retransmission: 2 (0-254)

Commands:

```
AT&F6
ATS108=30
ATS104=1
ATS105=1
ATS102=1
ATS113=2
ATS103=5
AT&W
ATA
```

Reset

Licensed band configuration

Licensed Band Frequency: 430.0 Hz

Frequency hopping table

| Table 0 | | Table 1 | |
|---------|--------|---------|--------|
| #1 | 410.00 | #2 | 410.00 |
| #3 | 410.00 | #4 | 410.00 |
| #5 | 410.00 | #6 | 410.00 |
| #7 | 410.00 | #8 | 410.00 |
| #9 | 410.00 | #10 | 410.00 |

☒ Link values from Table1 to Table0

Send message

Radio Panel

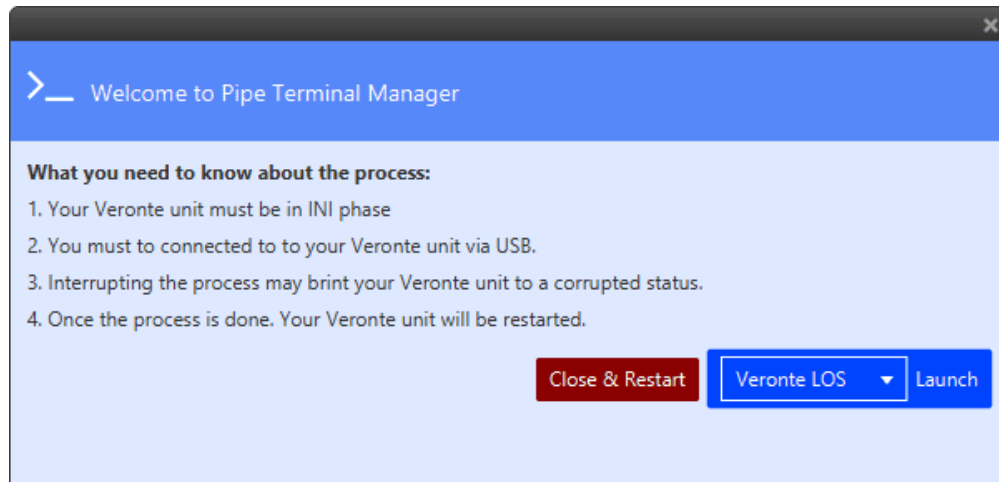
These parameters have to be set accordingly to the radio module installed in Veronte.

- **Radio Module:** Each Veronte has only one of the following radio modules P400, P900 and P2400. Select here the radio module installed.
- **Output power:** Sets the available power output.
- **Connection:** Point to Point or Point Multi Point. By default all units are paired with PP connection.
- **Address:** Master or Slave . Select the role for each Veronte by selectig Master or Slave. By default the air unit is defined as Slave and ground units as master.
- **Network address:** the network address is a number that must be equal between Verontes using the same network.
- **Packet retransmission:** Each data pack is sent as much times as defined here.

By pressing **Send message**, the user is able to send all these parameters to the Microhard module.

The commands can be also manually sent. For advanced Microhard configuration, please check the specific radio module User Manual [here](#).

Once the configuration is sent, the console is closed and the **Close & Restart** button appears. By clicking this button, the user can exit the configuration process and the Veronte unit is restarted. The Microhard radio module is now configured.



Veronte Pipe Terminal closing and Veronte restart

1.4 Calibrating Servos

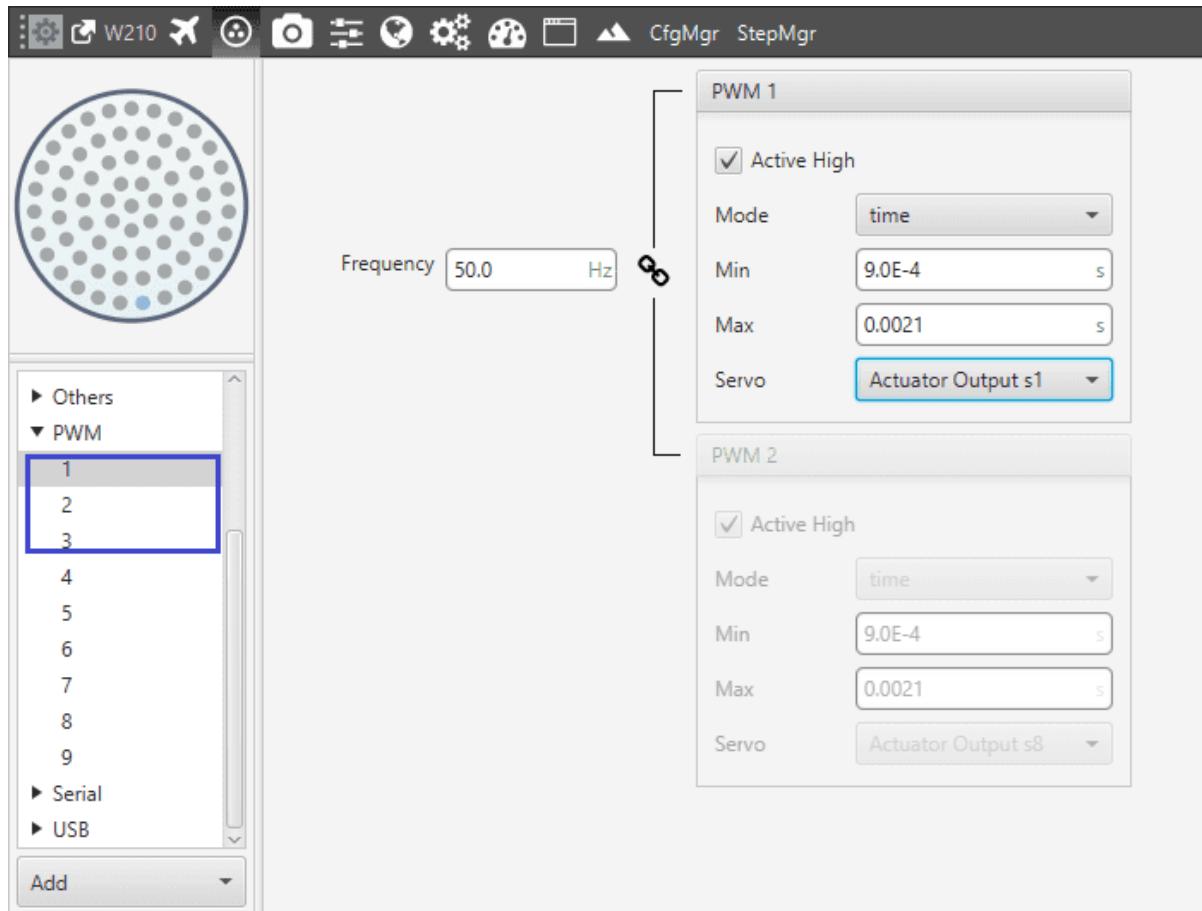
This section includes the information of how to calibrate the servos that control the attitude of the platform. Configuration of the servos includes: assignment of PWM pins to each servo, controller output to servo output relation, and trimming of the servos.

1.4.1 Servos Output

The first step of the process is the servos/actuators configuration. Let's consider a flying wing as our platform.

The controls in this case are the two control surfaces (elevons) and throttle. Each one of these control variables corresponds to a PWM pin of the connector and they must be positioned in the same order in an **S** vector representing the **Actuator/Servo Outputs**.

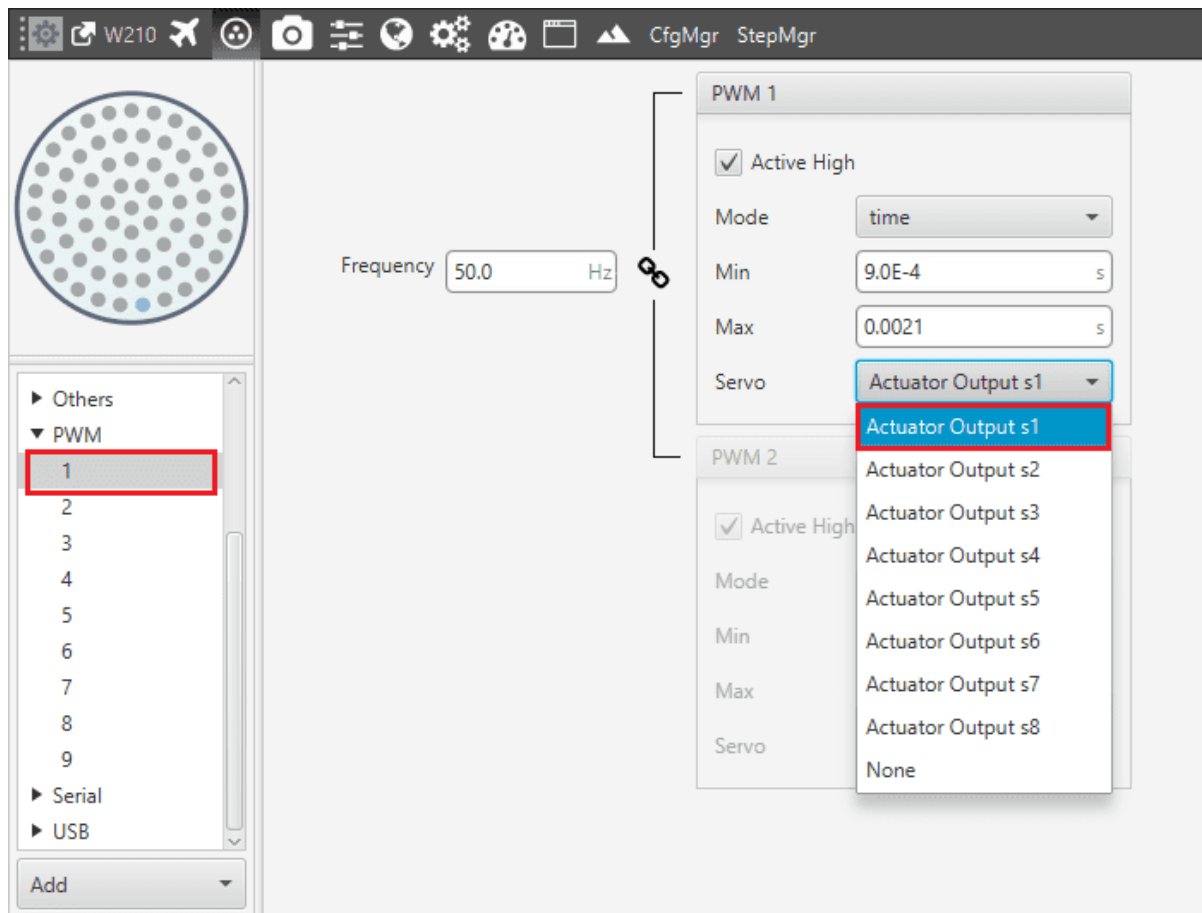
To assign the actuator outputs go the side panel , click on Setup, then go to Connections  and open the PWM tab.



Output-PWM pin links


In this case, only 3 pins are used:

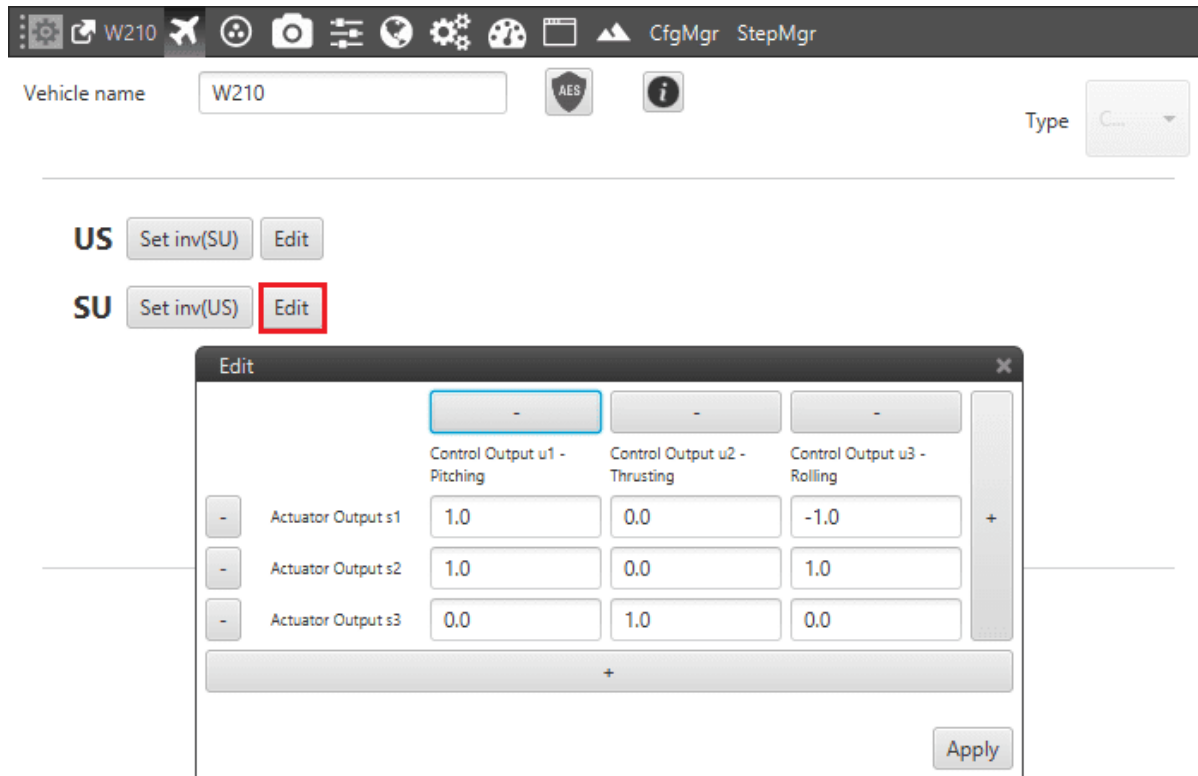
- Output 1 – Actuator Output s1 (**Elevon 1**)
- Output 2 – Actuator Output s2 (**Elevon 2**)
- Output 3 – Actuator Output s3 (**Throttle**)



Output 1 – PWM pin 1 configuration

1.4.2 SU Matrix

At this point, the **S** vector is defined and the **SU** matrix can be edited. At the Setup menu, go to Devices , then open the Actuators tab, and click on Logical. By clicking on **Edit** it is possible to configure the relation between the **Controller Outputs** (**U** vector) and the servo movements (**S** vector).



SU matrix editing


A flying wing is configured as follow:

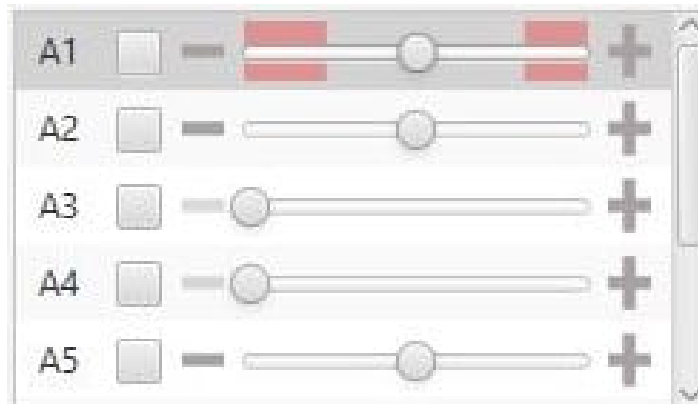
1. **Pitch Control:** control surfaces must be moved in the same direction to modify the pitching angle. The contribution of the actuators has same magnitude and direction.
2. **Thrust Control:** the Actuator Output 3 is the only one that allows a thrusting change.
3. **Roll Control:** in this case, the contribution of the actuators must be set with the same magnitude and reverse direction in order to perform a rotation around the body axis of the aircraft.

Warning: The panel above considers the reference system of the aircraft. It should match the Autopilot's one. In case it would not, it can be edited by clicking on the corresponding axis in order to reverse its direction.

For more information about the logical calibration of the servos check [Logical](#).

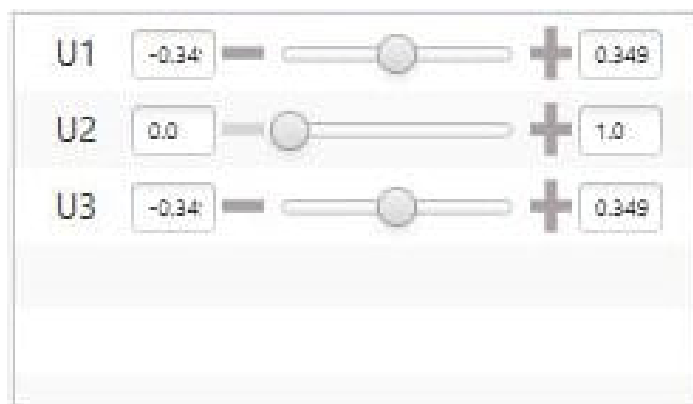
1.4.3 System Trim

As a final step, the system has to be trimmed. At the Setup menu, go to Devices , then open the Actuators tab, and click on Physical. The options that appear on the screen are presented as follows:



Servos Configuration Display

- **Servos:** this menu contains the servos of the platform. The signal to the system will only be sent if the checkbox next to the servo number (A1, A2...) is marked.



Control Signals Configuration Display

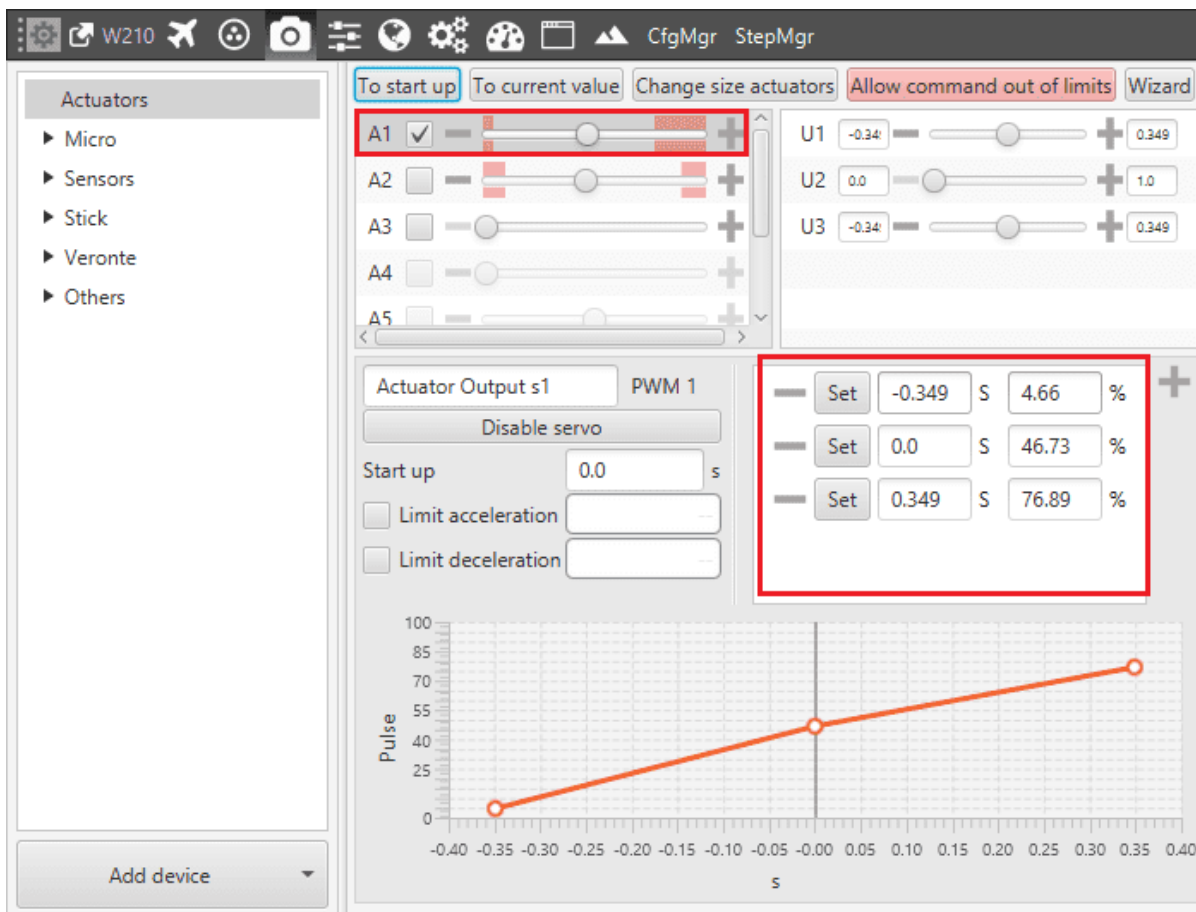
- **Control Signals:** this menu contains the variables (U vector) that represent the control signals generated by the system.



Servo PWM Configuration Display

- **Servo Position – PWM:** this option is used to set the transformation from a control position to a PWM signal. The example above is: a 20° degrees deflection (0.349 rad) of the right elevon corresponds to an 81'5 % pulse to be sent to the corresponding servo.
- The other options that appear on the screen are:
 - **To start up:** set initial values.
 - **To current value:** set the current value.
 - **Change size actuators:** change the number of actuators.
 - **Allow command out of limits:** allow manual control over the actuators limits established.
 - **Wizard:** recommended on first system configuration. It guides the user for configuring actuator limits and performance.
 - **Disable/Enable servos:** menu for enabling servos with their limits or just to disable them.
 - **Start up servo position:** set the pulse value for the Start up.
 - **Increasing/Decreasing Rate Limit:** set a limit for increasing/decreasing value of the servo.

The trim can be performed by moving the servos in three different positions: zero position, minimum and maximum deflection angle (angles are usually limited physically). These positions must be inserted and saved in the software by clicking on **Set** when the actuator is in the desired position. Otherwise, position can be introduced manually.



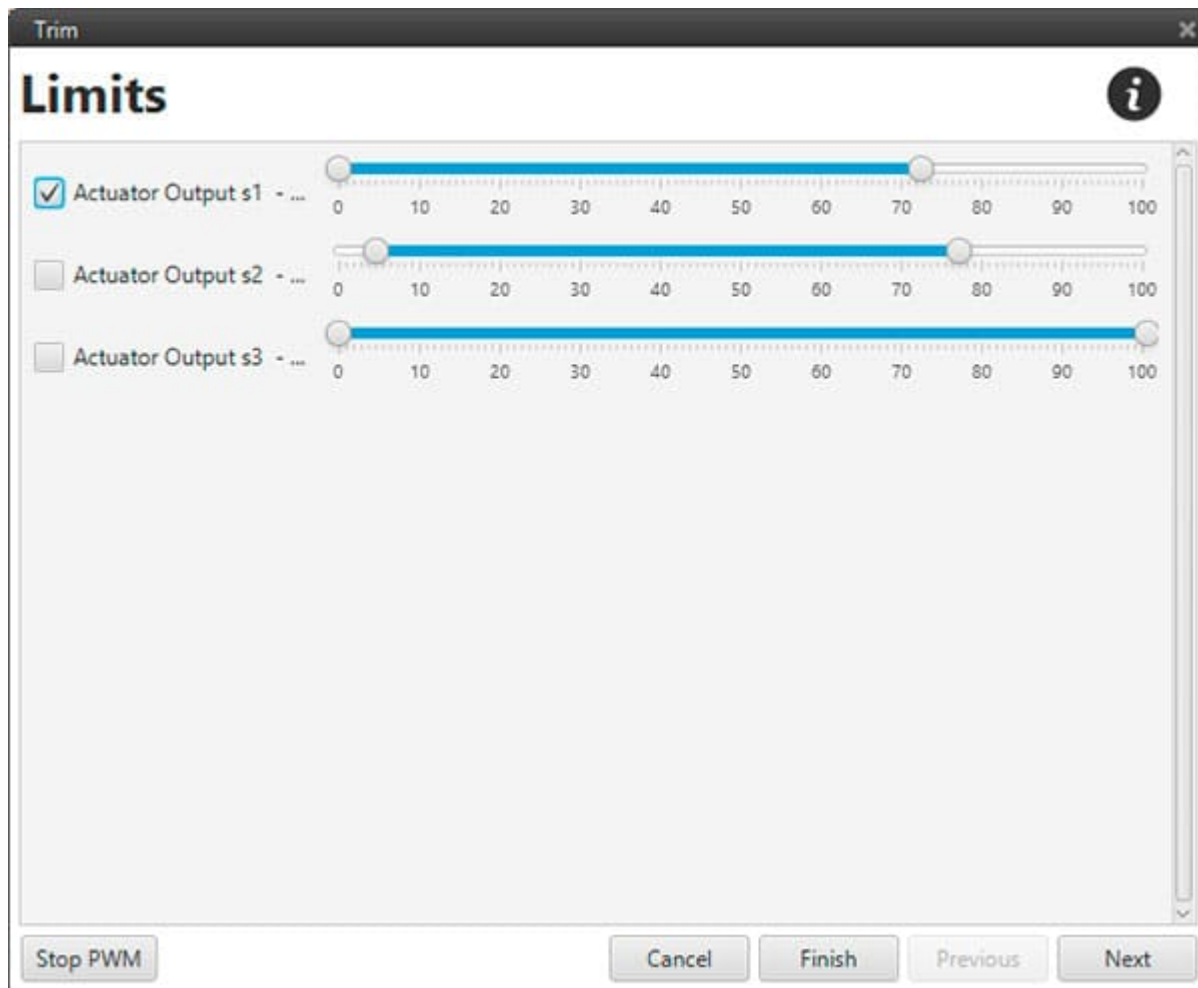
Actuator 1 Trimming

The picture above shows the setting of the elevon number 1:

- **Minimum:** -0.34906 [rad] deflection; 4.66% PWM output.
- **Zero position:** 0.0[rad] deflection; 46.73% PWM output.
- **Maximum:** 0.34906 [rad] deflection; 76.89% PWM output.

Warning: The actuators can be moved directly from Veronte Pipe only when the system is in an Initial phase (when there is no phase selected in Veronte Panel). During the actuator run, if the desired position is in the **Out of range** zone (red zone), it is possible to click on **Allow command out of limits** in order to move completely the actuator and find the correct position.

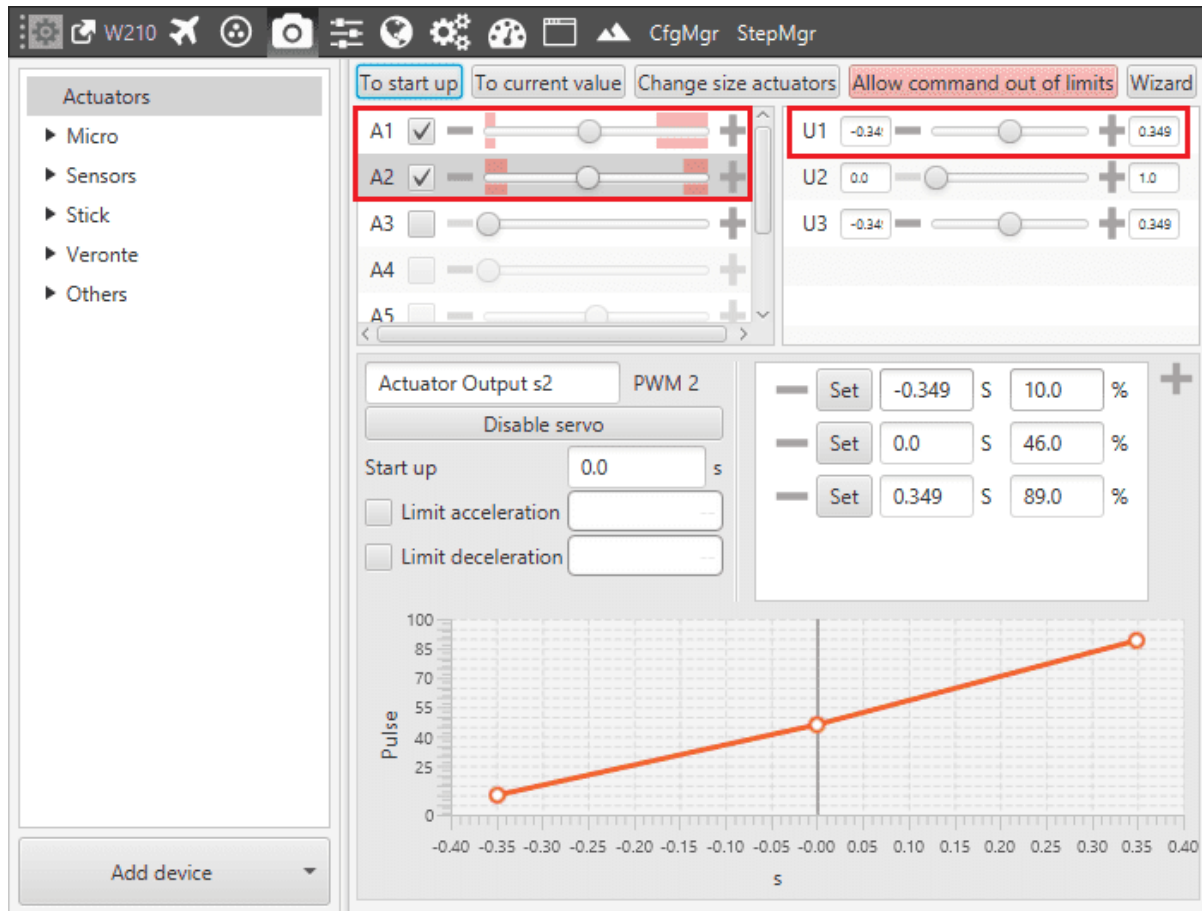
This procedure can be performed in the same way by using the **Wizard**. This tool allows moving actuators limits easily and finding the correct range, as shown below.



Trim wizard tool

In order to perform a final check, it is possible to select the desired channel and test pitch, roll and thrust control.

The image below shows a pitching output testing. By moving the U1 control, surfaces must change the position according to the reference system: positive corresponds to nose down and negative to nose up.



Pitching test

For more information about the physical calibration of the servos check [Physical](#).

1.5 External Transmitter Configuration

This section includes the information of how to connect an external PPM transmitter to control the platform fitted with the autopilot.

To use the joystick in the system, connect the PPMout of the trainer port to a digital input of Veronte and configure that digital input as the radio input in Pipe.

If the PPM level is 3.3V, pins 1-8, 10-17 and 55-58 pins can be used.

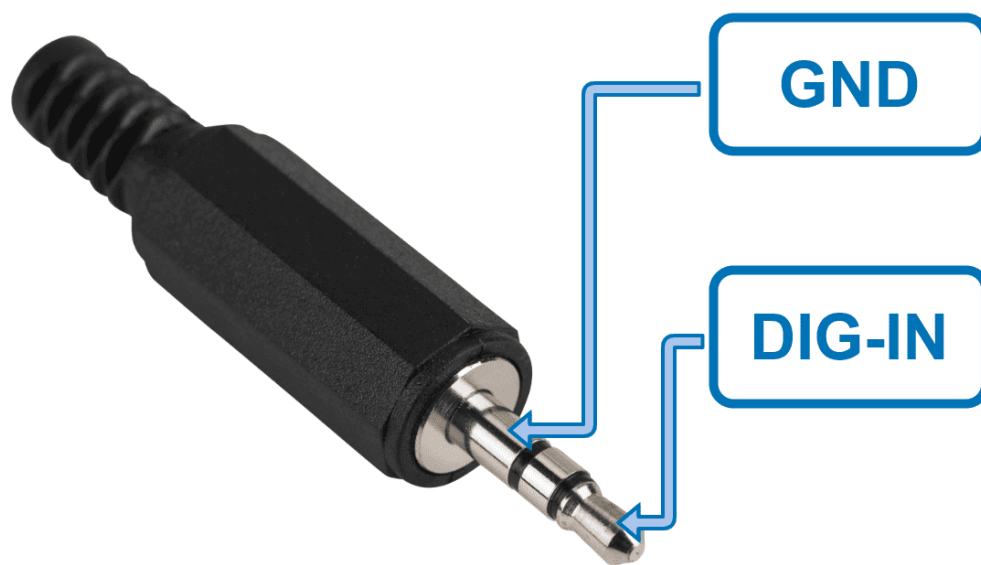
Veronte is compatible with standard Pulse Position Modulation (PPM) signals between 8 and 16 channels.





Jeti DC-16 Transmitter

Warning: Caution!! PPM signal must be into the Veronte voltage ranges. Some joysticks may need an adaptation board, please ask our team to check compatibility.

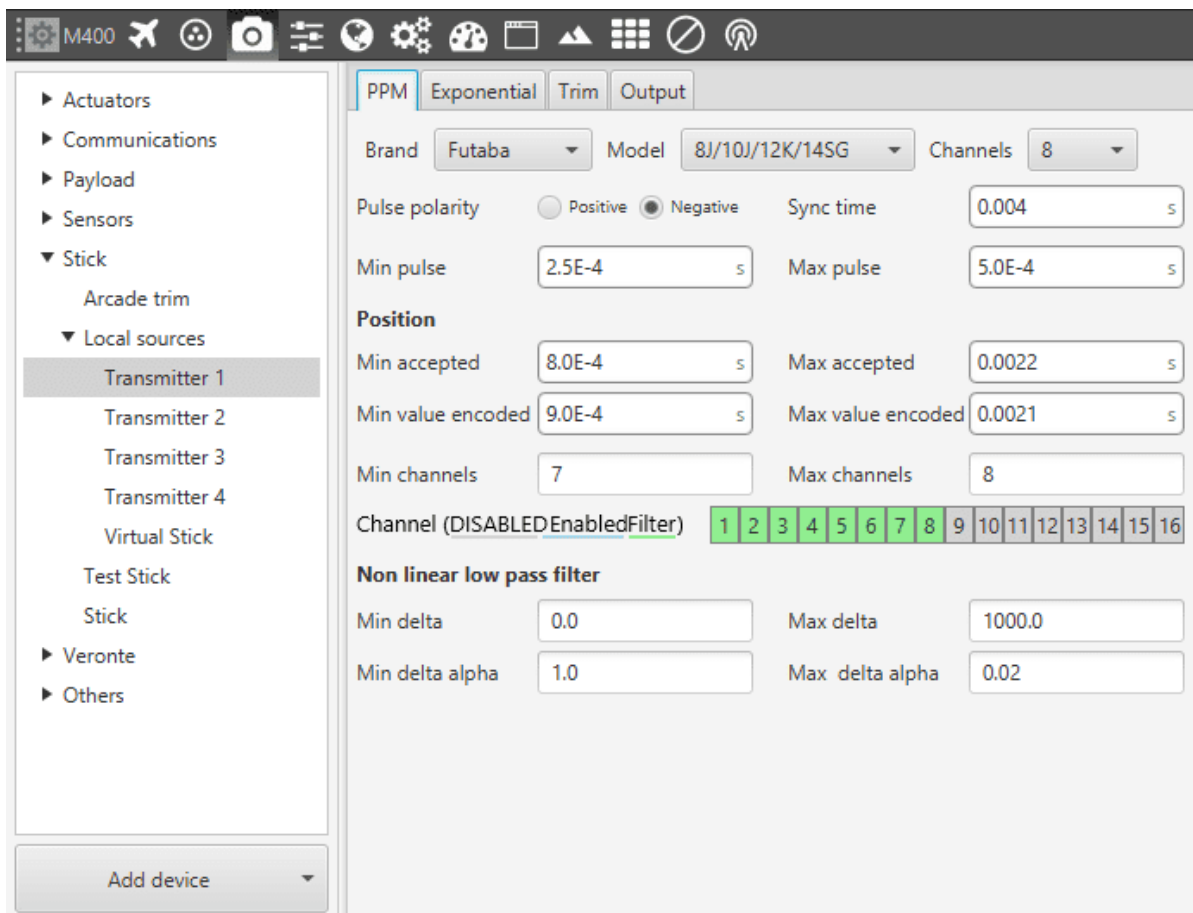
Veronte connector for CS is provided with 3.5mm stereo plug connector as follows:



Male Jack connector (3.5mm) pinout - CS Harness

In the side panel , click on Setup, go to Devices , then open the Stick tab, and finally go to the Local sources tab. A menu opens up with 4 tabs on top: PPM, Exponential, Trim and Output.

1.5.1 PPM



Stick Transmitter - PPM Configuration Parameters

Veronte Pipe can help the user by configuring the PPM panel in order to read the PPM output from the most famous transmitter brands:

- **Brand, Model & Channels:**
 - Futaba.
 - * Model 8J/10J/12K/14SG with 8 channels.
 - * Model 12K/14SG with 12 channels.
 - * Model T18SZ with 8 channels.
 - Jeti.
 - * DC 16 / DC 24 with 16 channels.

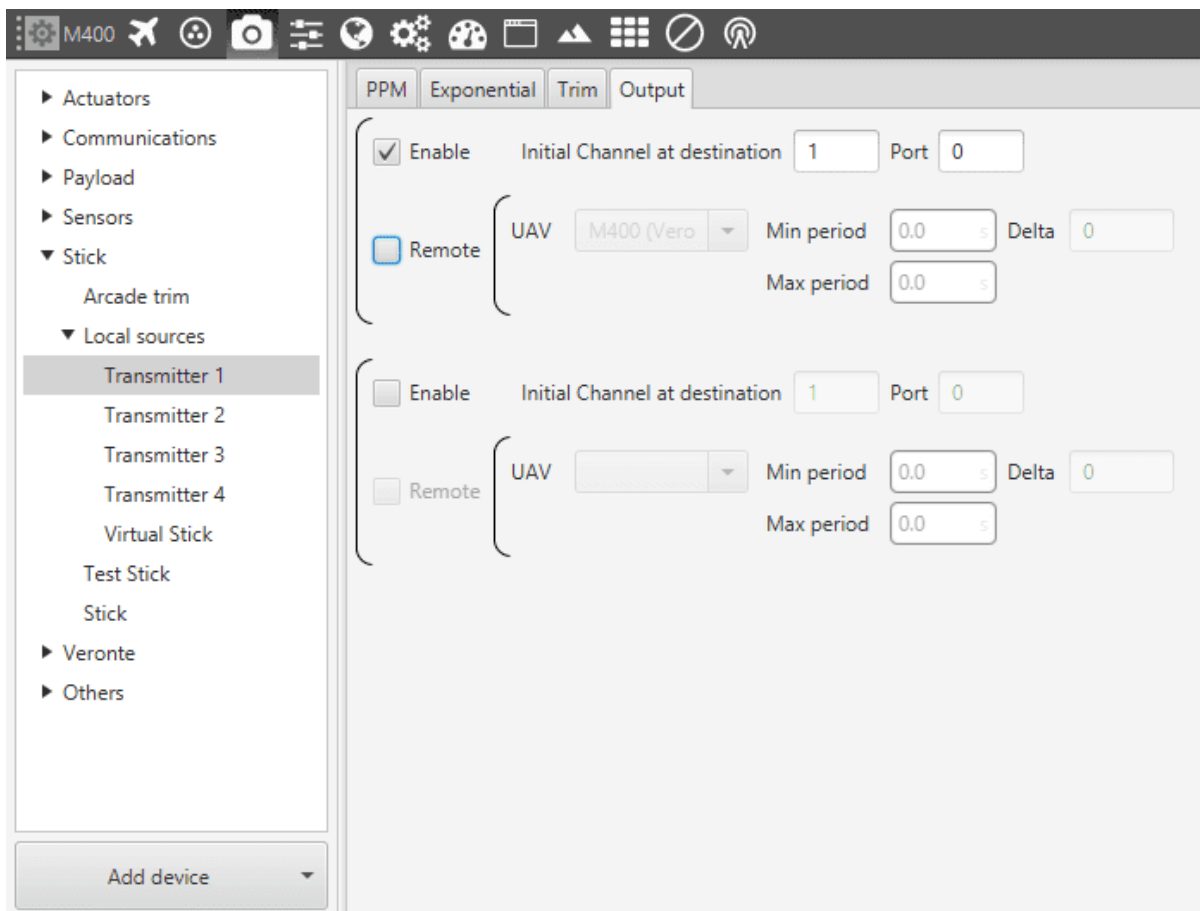
- FrSky.
 - * Taranis 9XD with 8 channels.
 - * X12S with 8 channels.

If the User's transmitter is not present in the previous table, it is also possible to configure the PPM parameters in order to match with the transmitter's PPM output.

Please check the [Stick Configuration](#) section for more information. In the same section, it's possible to find further information about Exponential and Trim configurations.

1.5.2 Output

Once the stick has been configured, the commands that arrive at the ground autopilot have to be sent to the air one.



Stick Transmitter - Output

Toggling the option *Enable*, the user indicates at which channel of the **AIR** autopilot will be sent the first channel received in the **GND** autopilot. The channels arrive at the platform by order and without spaces between them i.e, if the external transmitter is sending the **GND** the channels 1,2,3,4 and 6, then the **AIR** autopilot will arrive channels 1,2,3,4 and 5, where channel 5 of the **AIR** will actually be channel 6 of the external transmitter.

The alternate option *Remote* permits the delivery of the commands to the platform by indicating the address on UAV (when *Broadcast* is selected, the commands are sent to all the air autopilots linked to the ground one).

1.6 Pre-flight Checklist



This section provides the necessary information on how to implement/prepare a pre-flight checklist. The latter will be sensitive to every vehicle and its mission. It is fundamental for the user to know the set of sensors and/or events that must be evaluated before allowing the vehicle to enter a ready-to-fly phase.

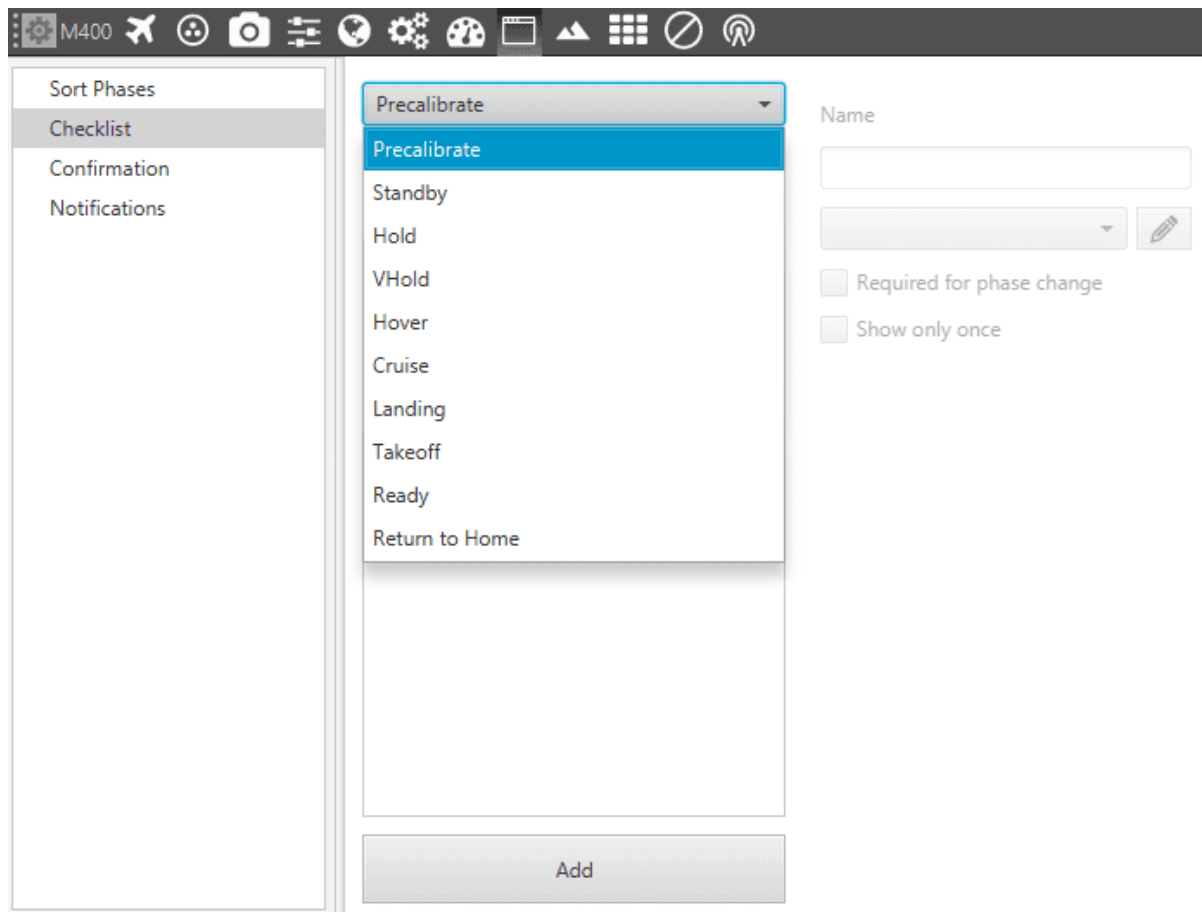
However, there is a set of critical parameters that should always be on the user's checklist. There are others worth-considering too. Both of them will be introduced hereafter with detail on the implementation in Veronte Pipe.

1.6.1 Main Checklist

Within the software, it is possible to define a checklist for every phase created. Such checklist can be tailored so that if one item is not satisfactorily fulfilled upon phase changing, the vehicle won't be able to swap to the desired/programmed phase.

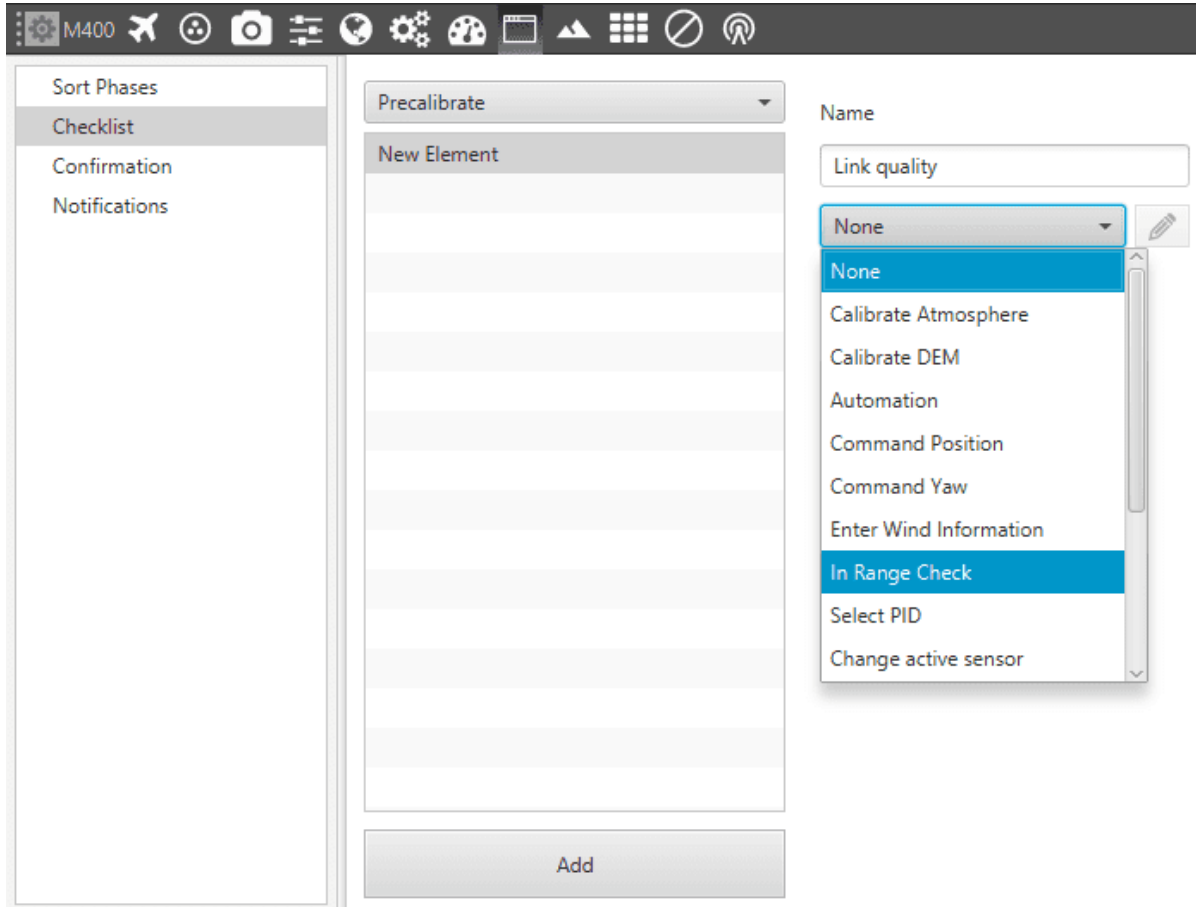
These checklist are also helpful for the user to check relevant variables when a phase change occurs. But for what it is concerned in this section, a checklist needs to be defined on the Precalibrate phase, i.e. the initial default phase of the autopilot.

Go to , then click on Setup. On the Setup toolbar then go to Panel . And finally click on Checklist. There the user should select the Precalibrate phase.



Panel Checklist – Precalibrate Phase

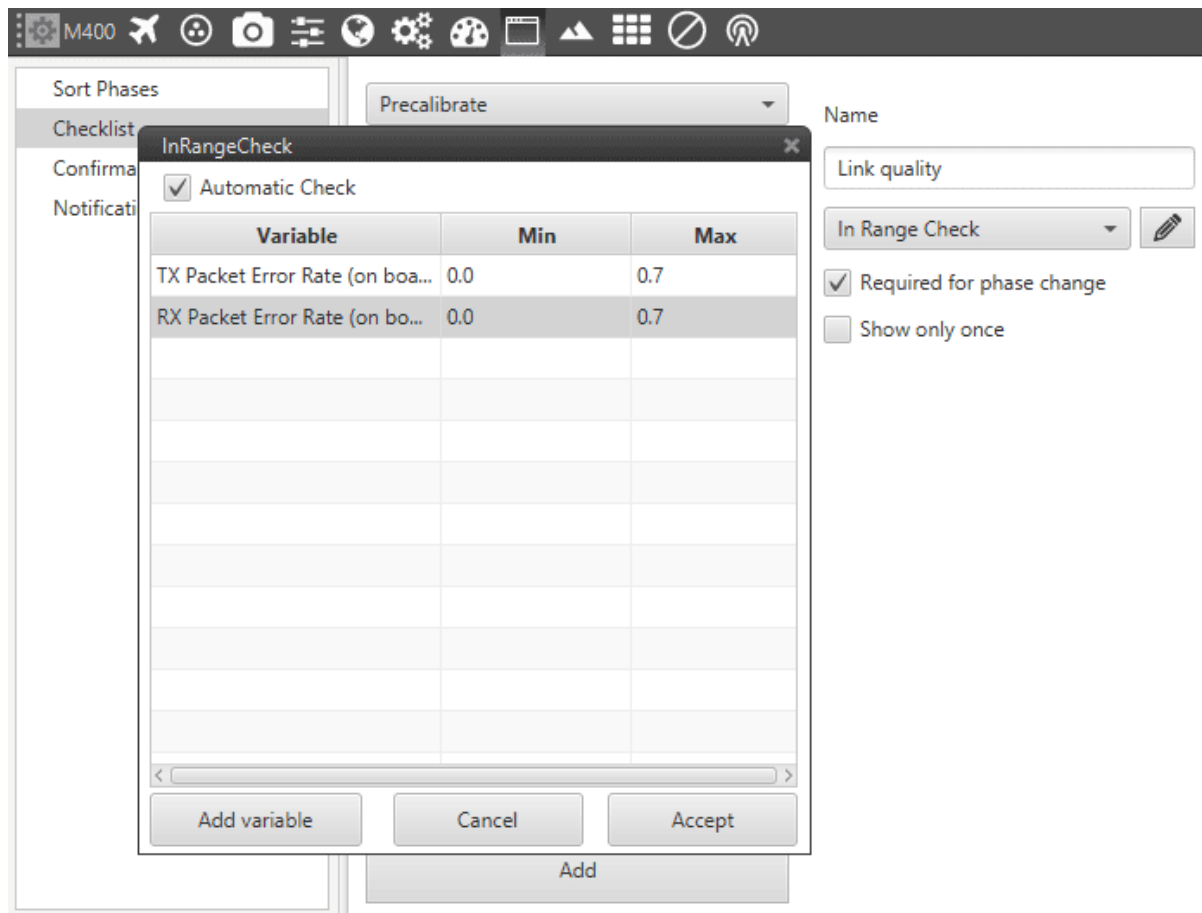
When creating a new element the user can select among many options: from commanding a certain action (position, yaw, etc), calibrating a sensor, to check that a variable or group of variables are within a range of values (see figure below). Then there are also the options “Required for phase change” and “Show only once”, which are self-explanatory. “Required for phase change” should be selected for every item as the user does not want to start the flight without this checklist completely verified.



Options for New Element

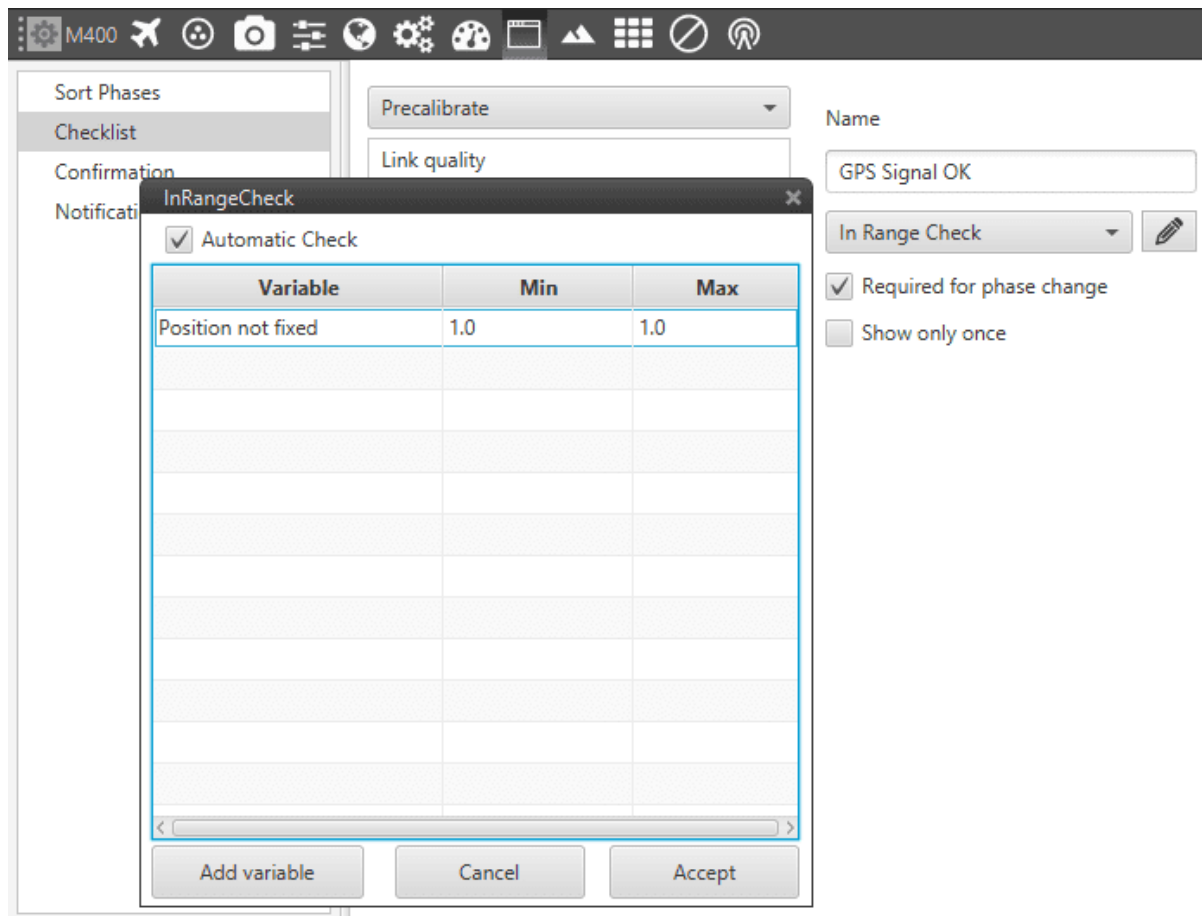
The main checklist is composed of:

- **Link quality:** radio connection from the **GND** unit to the **AIR** unit can be evaluated with 2 built-in variables. They measure the transmitting and receiving error rate, i.e. the percentage of messages that are not received or transmitted correctly. What is set is that this variables' values should not be bigger than 0.7 (option “In Range Check”) and the option “Automatic Check” should be toggled so this is reviewed automatically (see the figure below).



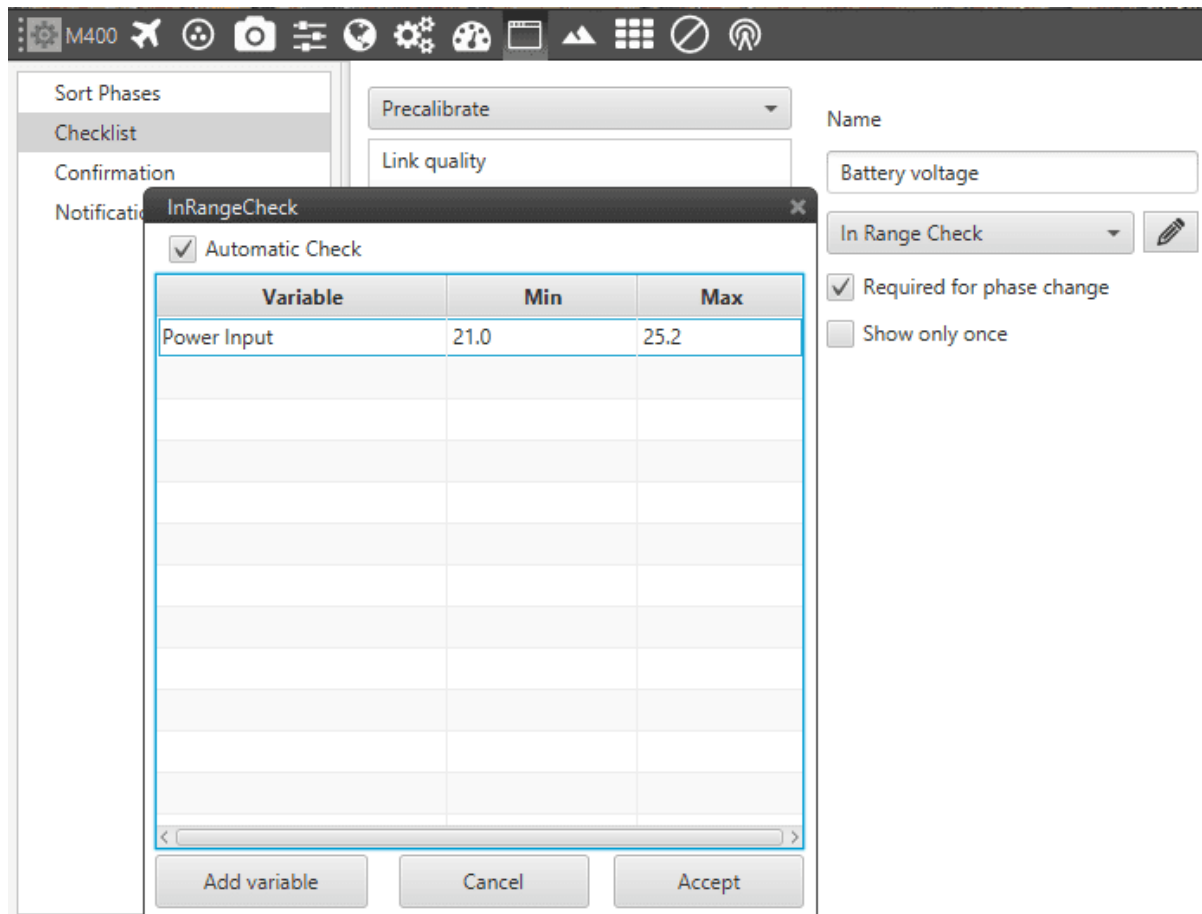
Link Quality Parameters

- **GPS signal OK:** if the GPS module is not working correctly there is a boolean variable called “Position not fixed” that will be turned to 0. Therefore option “In Range Check” is selected with a value between 1 and 1, i.e. the boolean variable should be *true*. Option “Automatic Check” should also be toggled.



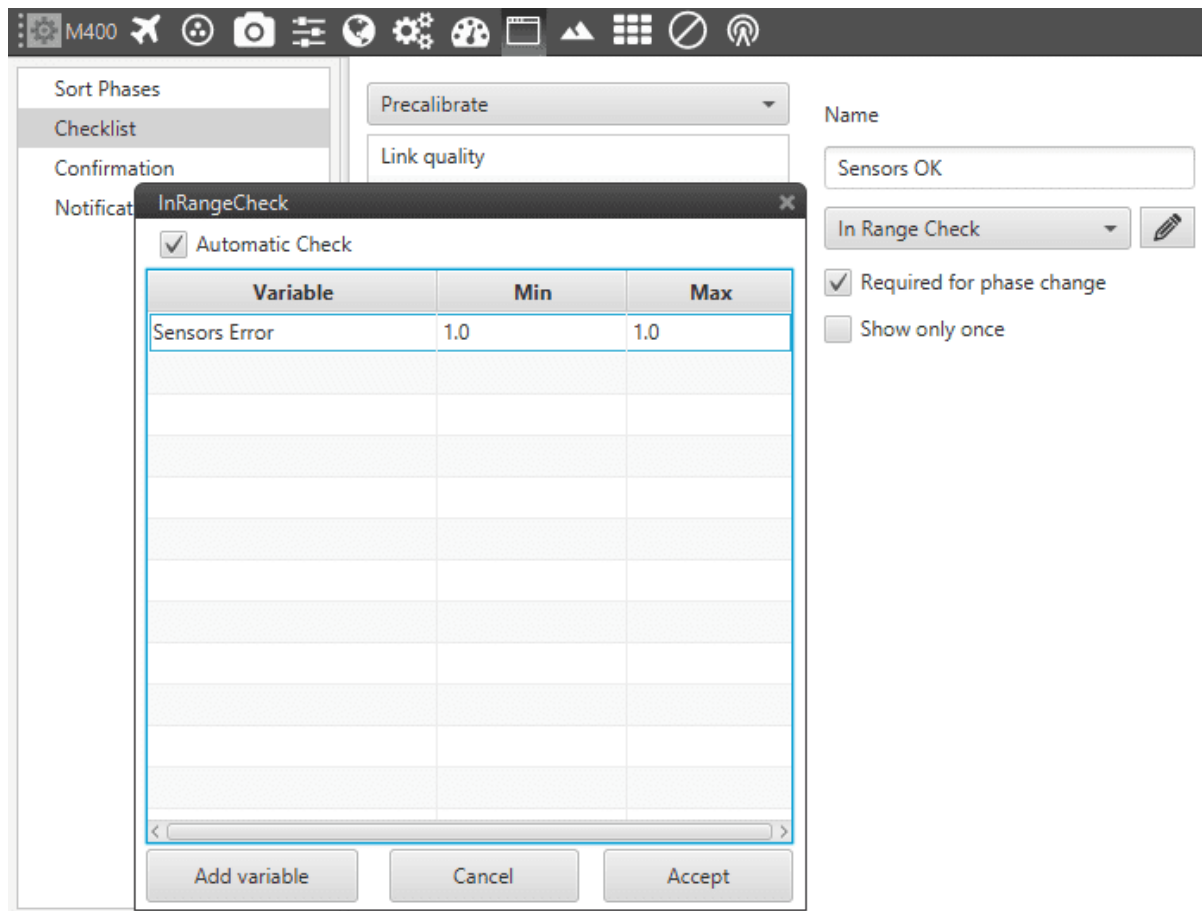
GPS Signal Parameters

- **Battery voltage / fuel volume:** a system variable should be used to monitor the voltage of the battery of the amount of fuel left on the tank of the vehicle. Then, option “In Range Check” should be selected setting the range of possible values, e.g. 21 V and 25.2 V for 6 Cell LiPo Battery. Option “Automatic Check” should also be toggled.



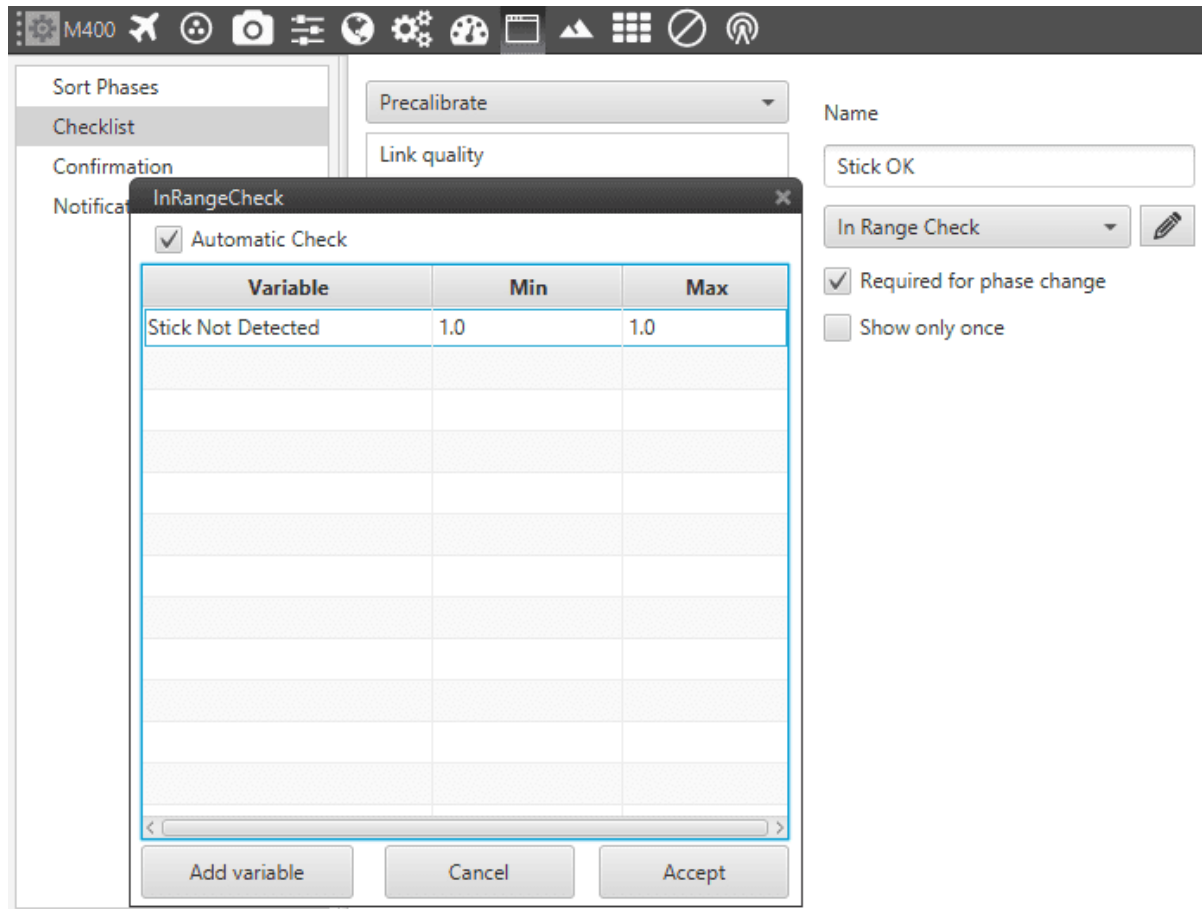
Battery Voltage Parameters

- **Sensors OK:** there is a boolean variable that checks all the autopilot's available sensors called "Sensors Error". If any of the sensors is not working then this variable will be 0. Therefore option "In Range Check" is selected with a value between 1 and 1, i.e. the boolean variable should be *true*. Option "Automatic Check" should also be toggled.



Sensors Checkup Parameters

- **Stick OK:** there is a boolean variable that checks that there is an external transmitter connected to the **GND** autopilot. The latter is essential so the mission can be carried in “Manual mode” if ever necessary. The variable is called “Stick not detected” and if there’s no external transmitter connected it will be 0. Therefore option “In Range Check” is selected with a value between 1 and 1, i.e. the boolean variable should be *true*. Option “Automatic Check” should also be toggled.




External Transmitter Parameters

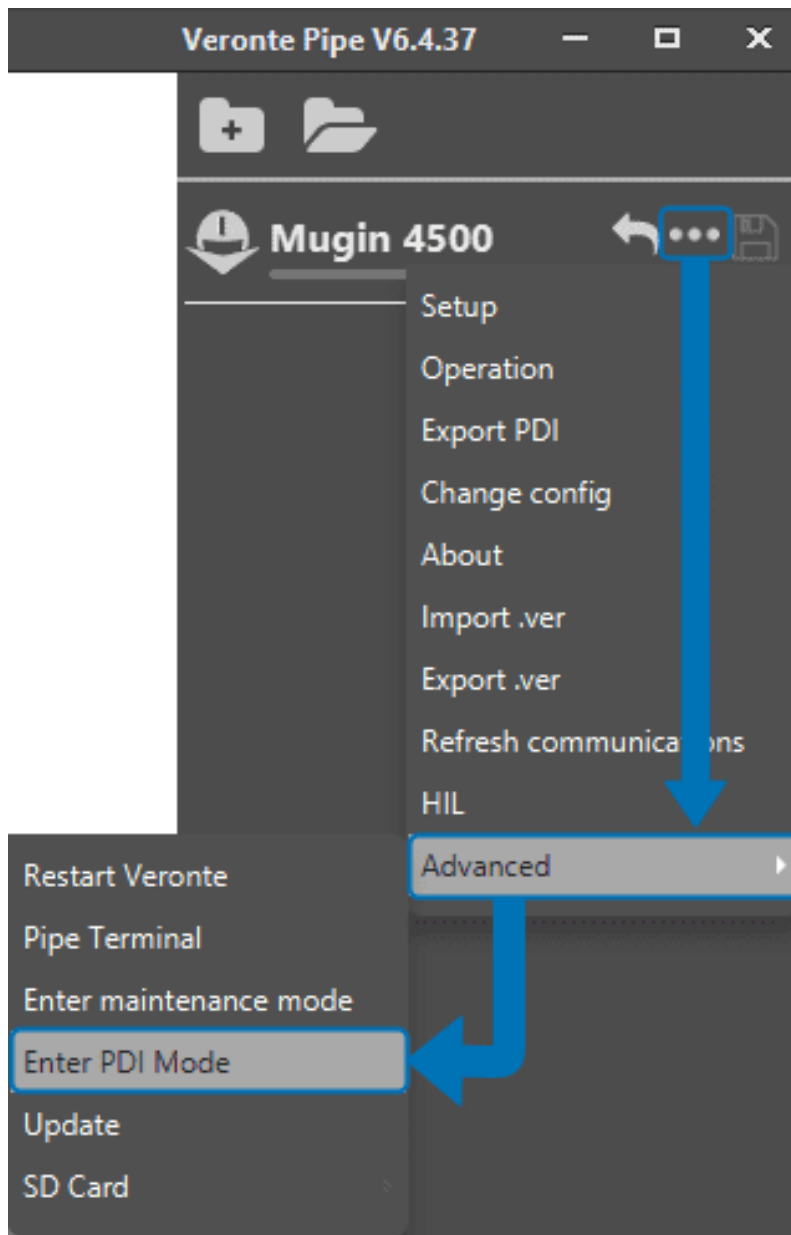
1.6.2 Other Items

Other parameters worth-considering to be on the user's checklist:

- **Terrain Mesh OK**
- **Mission OK**
- **Automations OK**

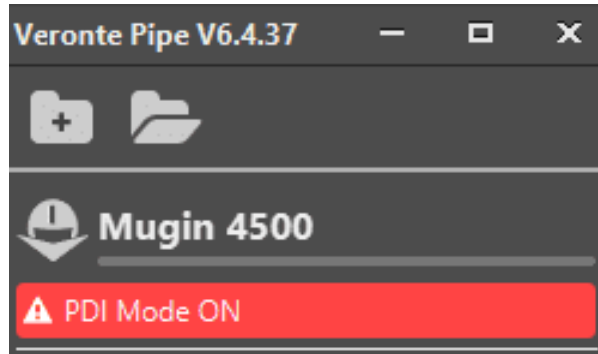
1.7 PDI Mode

This section will describe all the functionalities of the PDI mode. In order to access to this mode, the user should open the drop-down by clicking on , then on Advanced and finally on Enter PDI Mode as shown below.



Veronte Pipe Enter PDI Mode

After selecting PDI mode Veronte **will reboot** to enter this mode correctly. Once Veronte is in this mode, the following label will appear.



Veronte Pipe Inside PDI Mode

To exit this mode the user must follow the same procedure as to enter, and Veronte will have to restart.

PDI mode allow the user to change the setup if Veronte is not in INI phase (no phase in green in Veronte panel). Being normal mode, the user **cannot do** out of the INI phase:

- Reboot Veronte.
- Change Veronte setup (i.e. save to SD card).
- Enter in maintenance mode.

Warning: The following errors prevent normal mode operation:

- Memory allocation (Bit **8**): Problems with memory allocation for system variables. This error occurs when initializing the system.
- PDI version not compatible (Bit **11**): PDI version is higher than Veronte version. See [Pipe Configuration](#).
- System power up BIT Error (Bit **12**): Check that the whole system has started correctly. This error occurs when initializing the system.
- Stack Core 1 usage FAIL (Bit **16**): Memory overflow allocated for local variables of Core 1.
- Stack Core 2 usage FAIL (Bit **17**): Memory overflow allocated for local variables of Core 2.
- Main Power Error (Bit **117**): System power problems. The user should check that the power supply is in good condition.
- Those bits selected by the user in Setup - Devices - Veronte - Safety Bits.

If **any** of the above bits is false, the System Error (Bit **7**) will be set to false. PDI mode allows the user to operate even if there are any of the previous errors.

Tip: It is recommended to use normal mode when the configuration is finished, while PDI mode is useful during the development process.

This quick start guide provides the necessary information to have Veronte Autopilot set up and functional to operate its mission.



Veronte 4.5

This guide covers the following points:

- **First steps:** veronte autopilot connections and software installation.
- **HIL simulation:** how to perform a simulation with an existing configuration using veronte autopilot.
- **Internal radio module pairing:** how to pair of the ground segment (**GND**) and onboard (**AIR**) autopilots.
- **Outputs configuration:** how to calibrate the servos for a good control of the surfaces/devices that determine the attitude of the platform.
- **Transmitter:** how to set up an external transmitter to control the vehicle when necessary.
- **Pre-flight checklist:** set of sensors, communications, and other events that should be verified before starting the mission.

SYSTEM LAYOUT

2.1 Communication Layouts

Veronte allows for a wide variety of communication solutions in order to adapt to each mission and platform specifications

Tip: Most of the following communication solutions can be used independently, or in combination to create redundant systems or backup solutions.

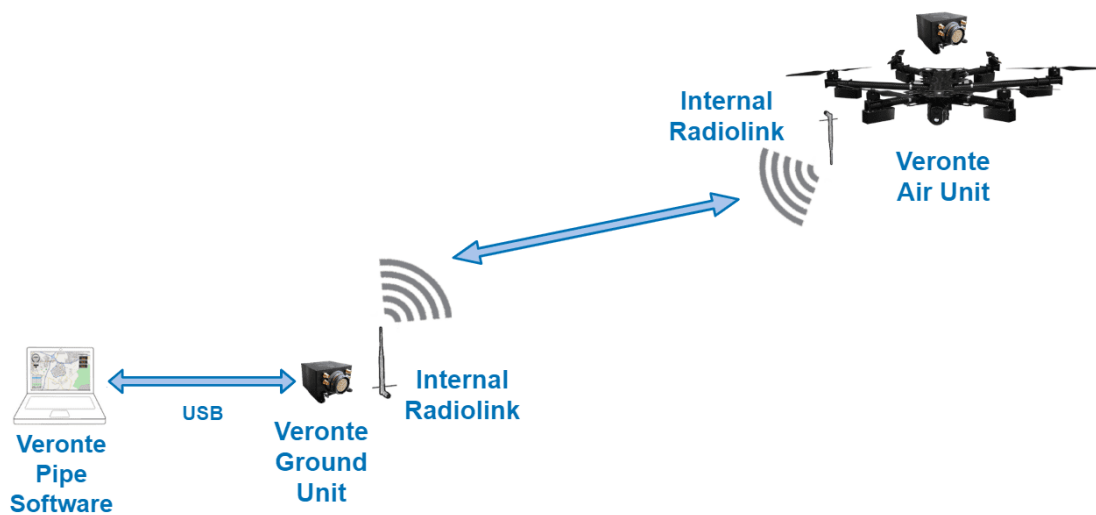
2.1.1 Air Segment

Communication solutions between the air and ground segments

2.1.1.1 Line-of-Sight (LOS)

2.1.1.1.1 Internal Radiolink

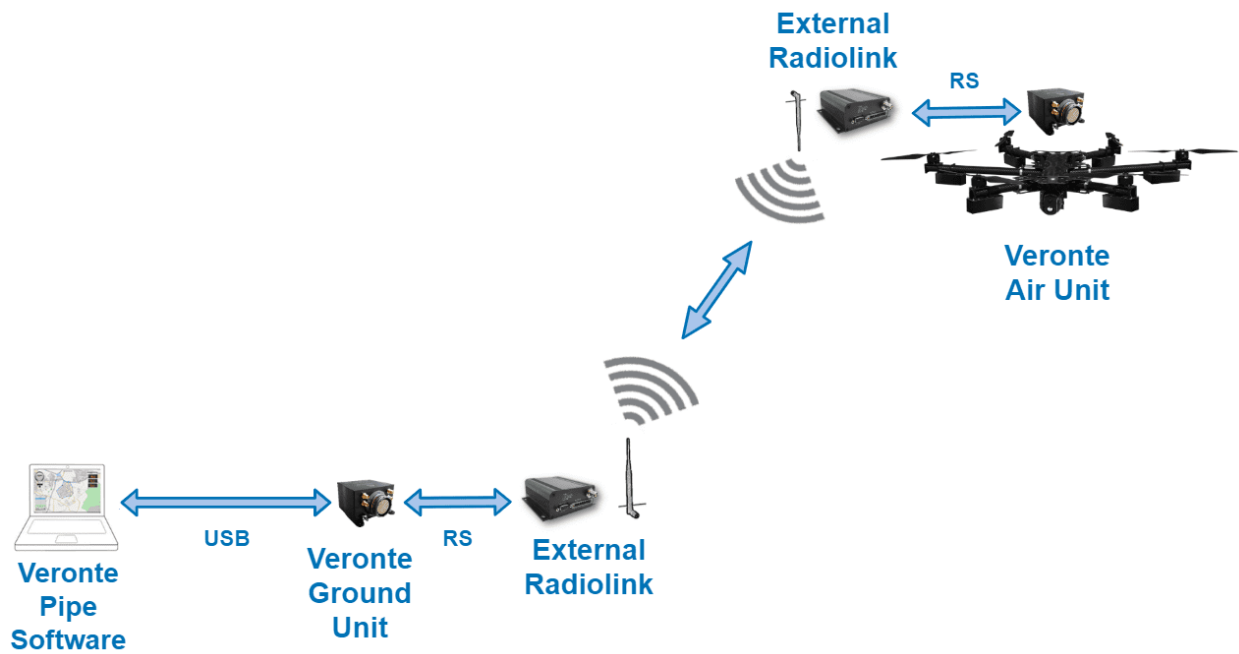
Standard setup



Internal Radiolink

2.1.1.1.2 External Radiolink

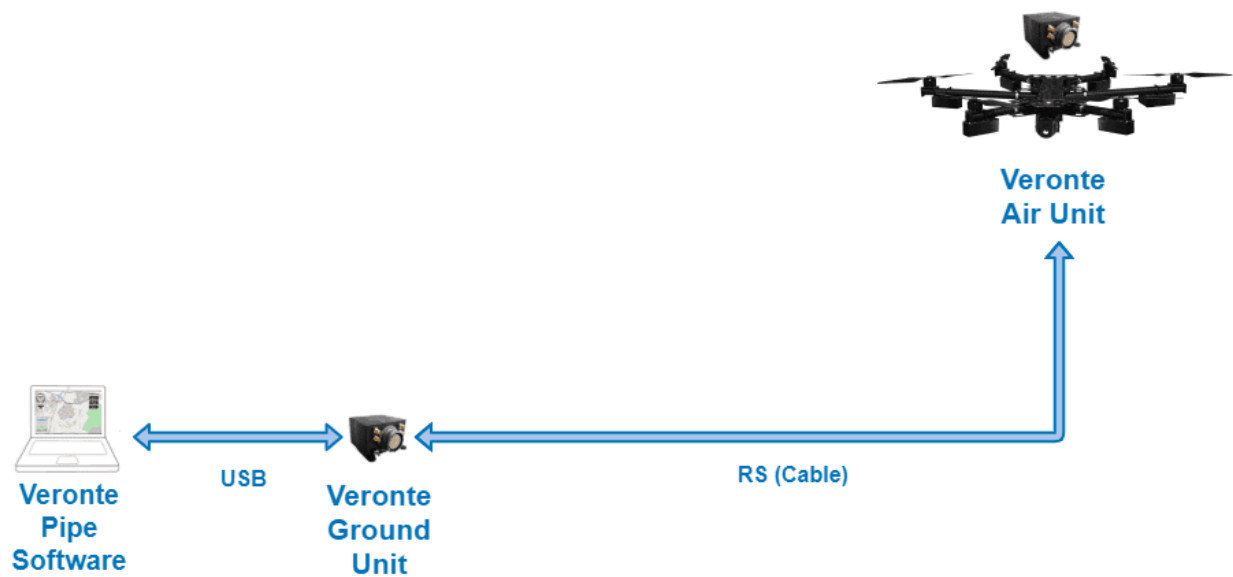
When increased range, bandwidth or channels are needed



External Radiolink

2.1.1.1.3 Tethered

For tethered solutions

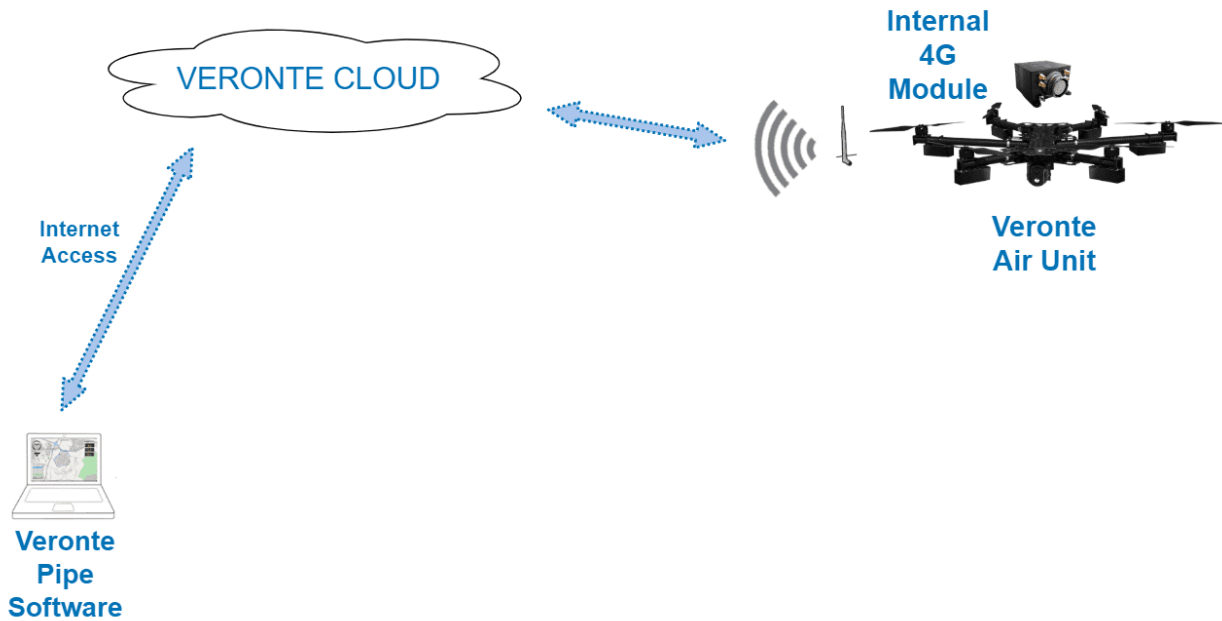


Tethered

2.1.1.2 Beyond Line-of-Sight (BLOS)

2.1.1.2.1 Internal 4G + Veronte Cloud

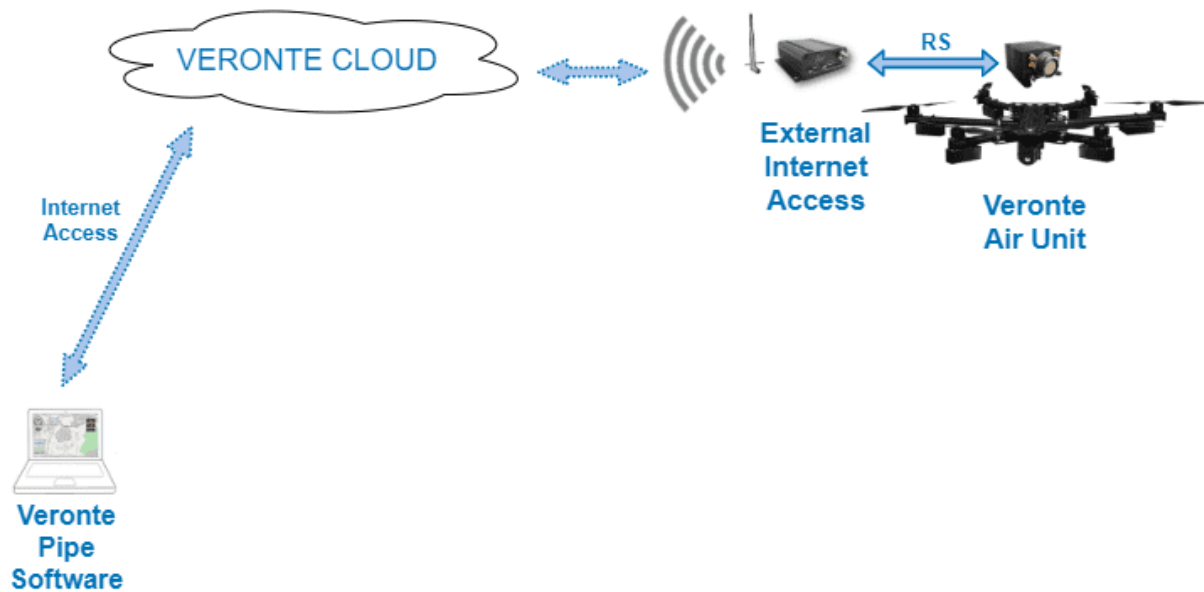
Standard BLOS



Internal 4G + Veronte Cloud

2.1.1.2.2 External Internet access + Veronte Cloud

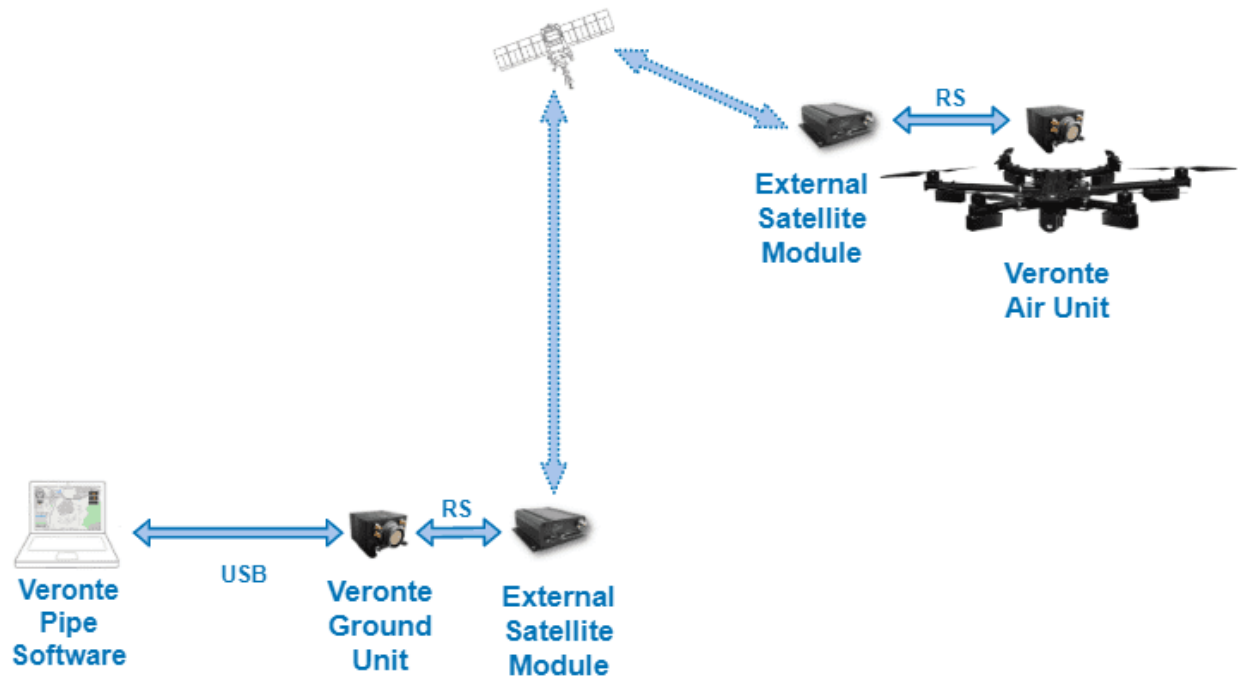
For alternative internet access



External Internet + Veronte Cloud

2.1.1.2.3 External Satellite communication

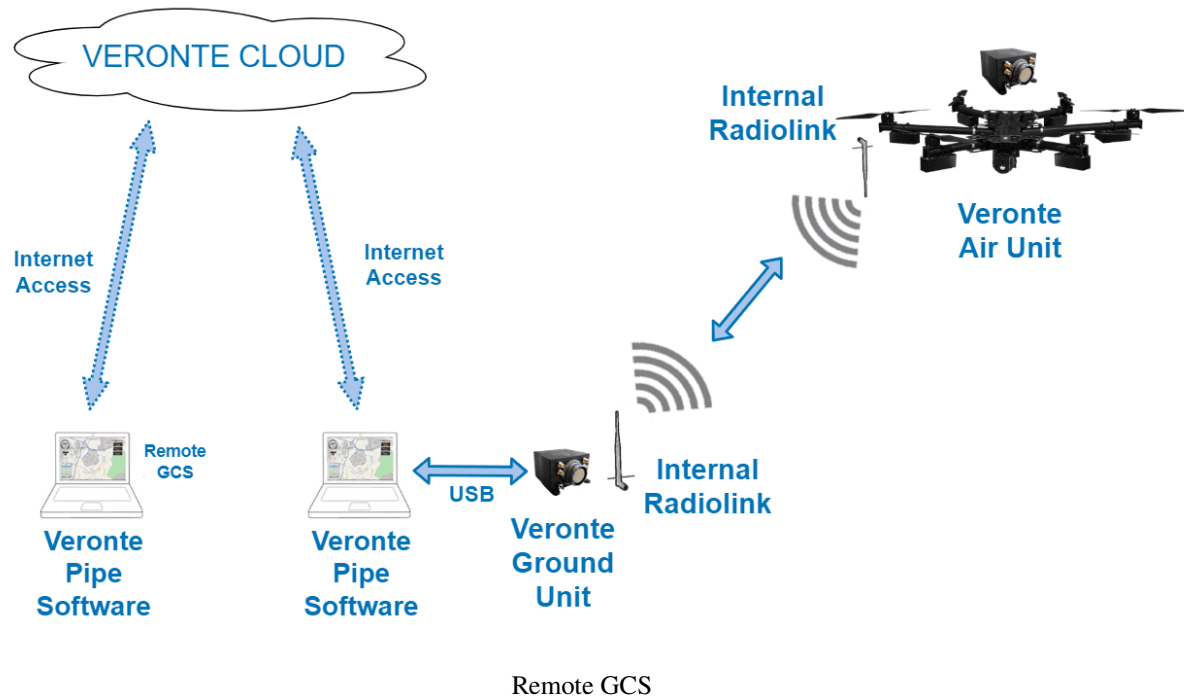
For maximum reliability



SATCOM

2.1.1.2.4 Remote GCS

For remote solutions with LOS backup operator

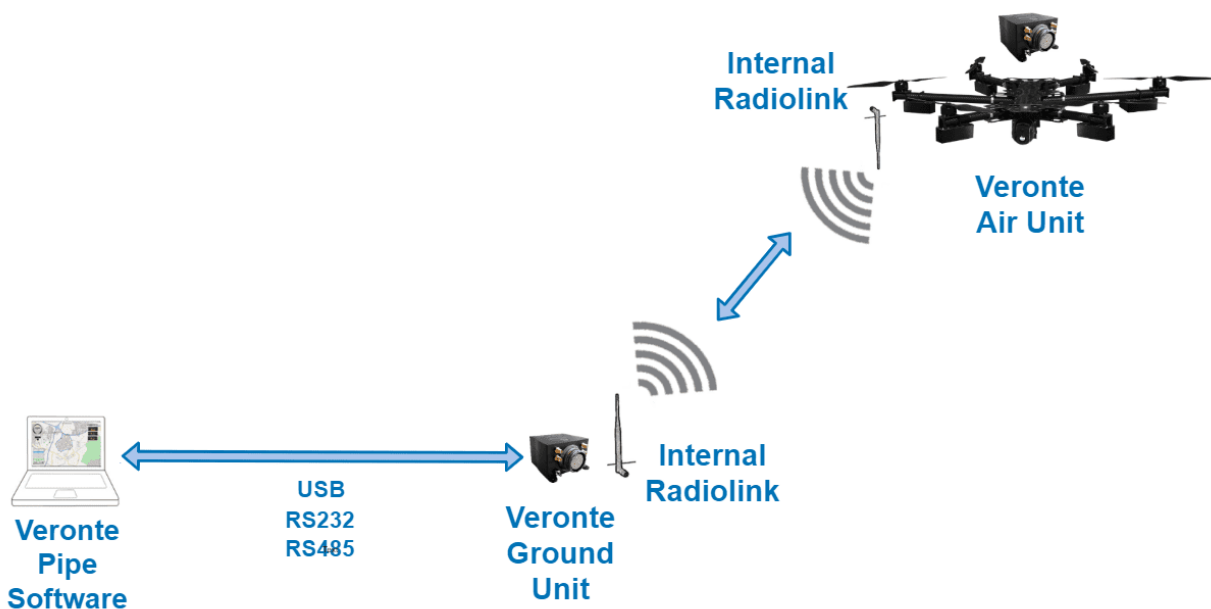


2.1.2 Ground Segment

Communication solutions between the different GS devices

2.1.2.1 Serial Interface

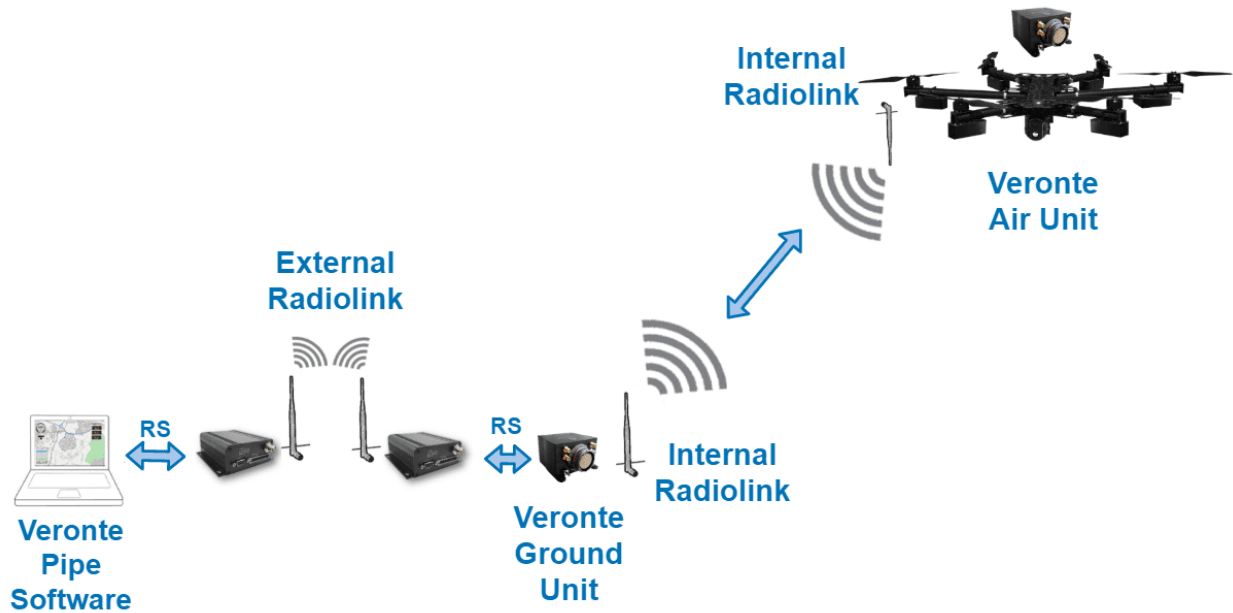
Standard setup



Serial GS

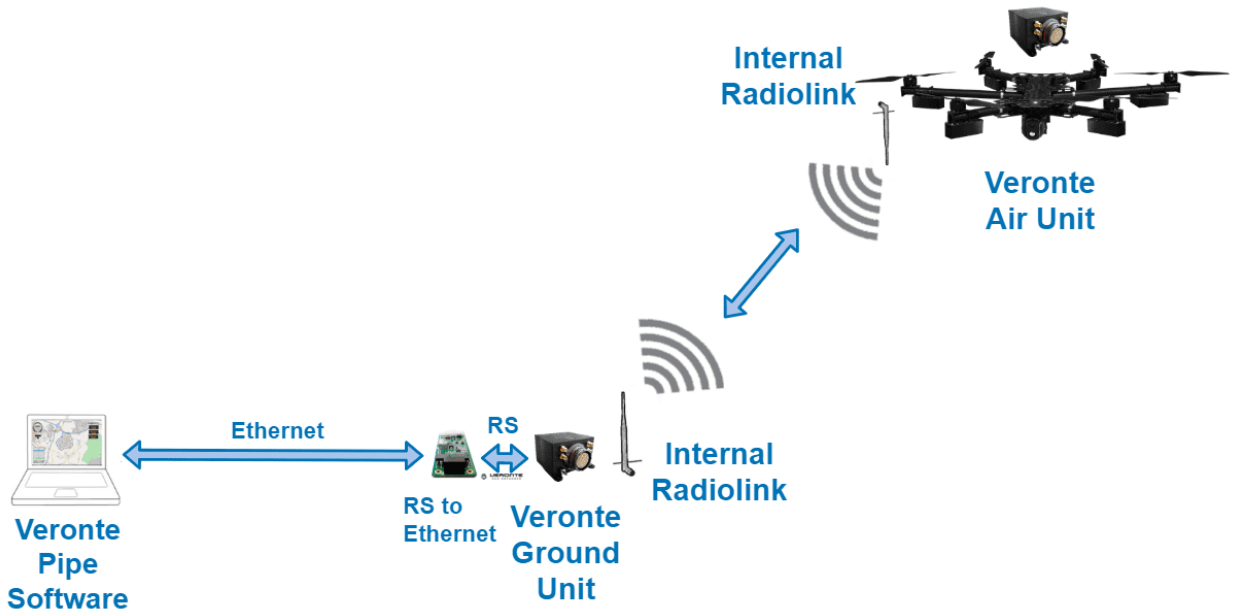
2.1.2.2 External Radiolink (GS)

For modular ground stations



2.1.2.3 Ethernet

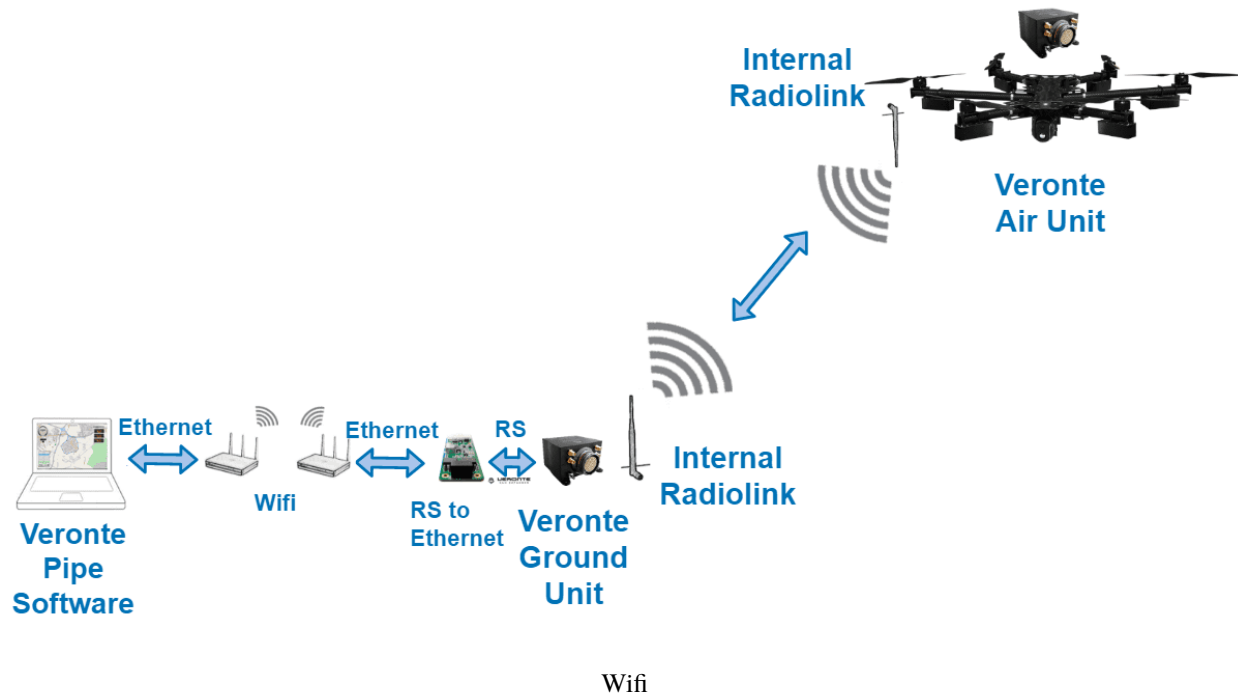
For complex Ground stations with several online devices



Ethernet

2.1.2.4 Wifi

For operation with laptops or tablets



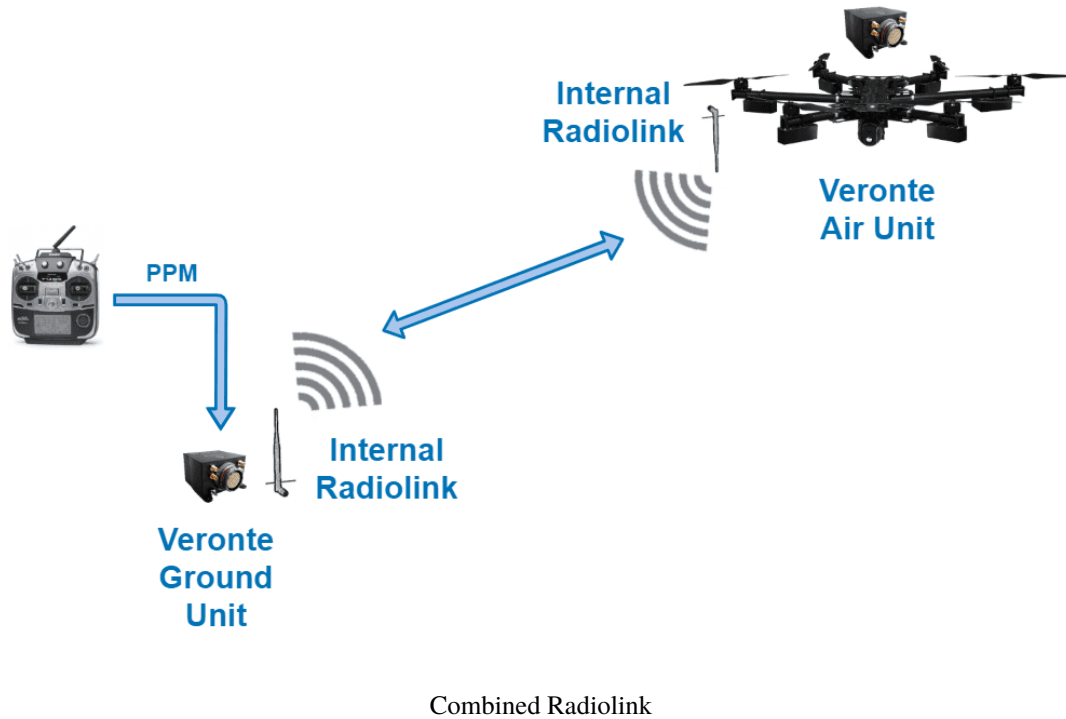
2.2 Manual Control Layouts

Veronte allows for a wide variety of pilot interface solutions in order to interact with manual flight modes, assisted flight modes (arcade) or payloads.

Tip: The following control solutions can be used independently, or in combination to create redundant systems, parallel channels (flight control + payload) or backup solutions.

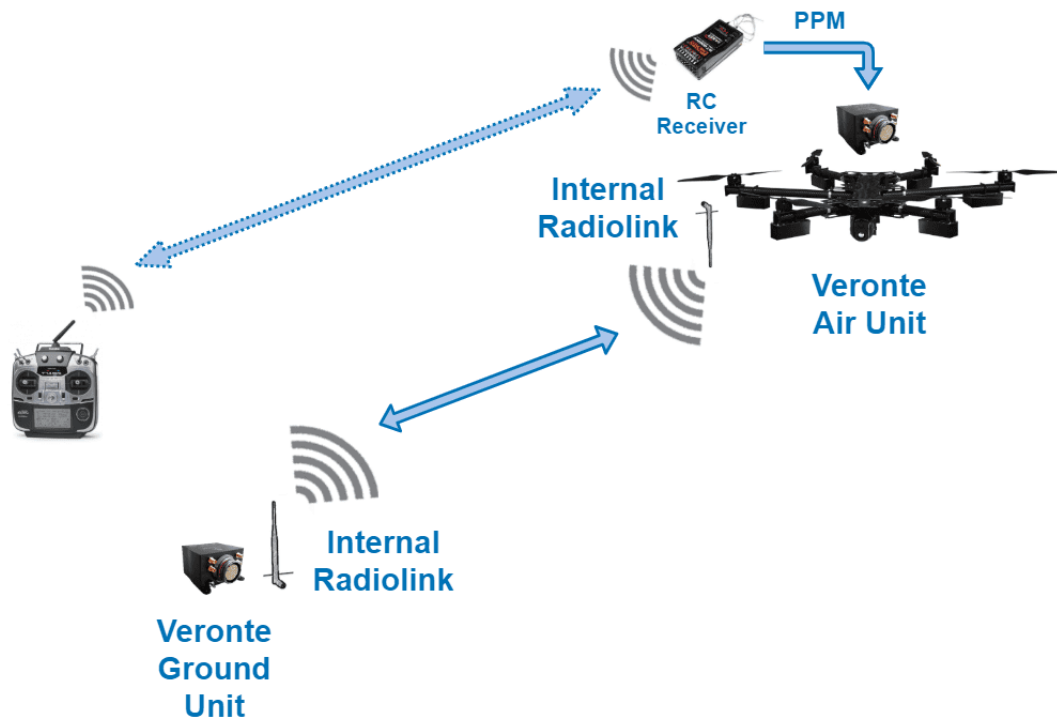
2.2.1 PPM to Ground Unit

Standard setup. Allows the usage of a single radio channel both for stick, control commands and telemetry, minimizing any potential interferences.



2.2.2 PPM to Air Unit

Allows for a backup manual channel in case there is a main channel loss and an emergency manual landing is needed. Recommended for initial development stages where automatic landing phases are not defined yet.

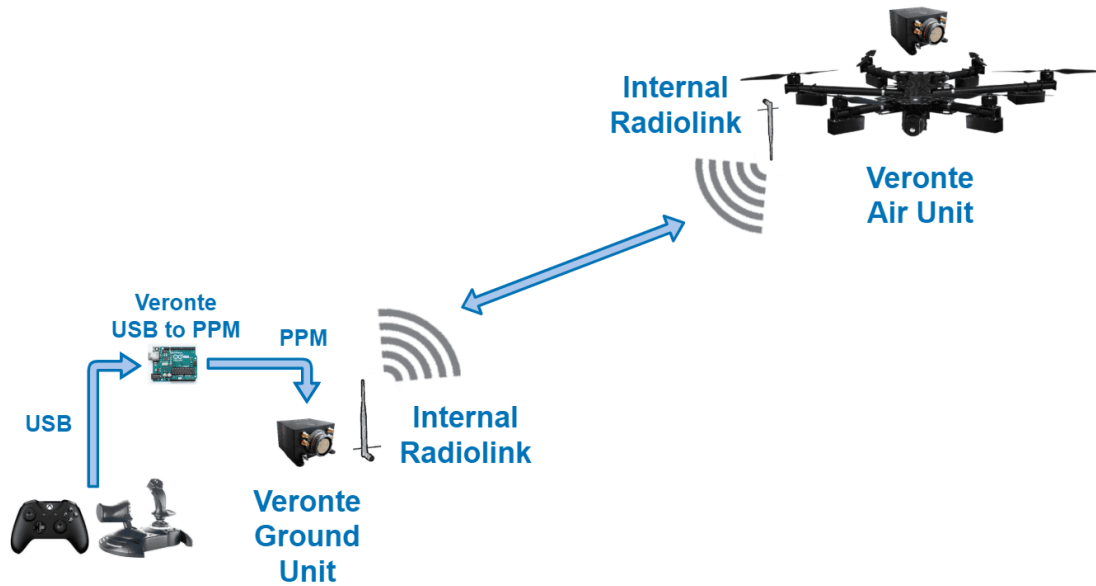


Parallel radiolink

2.2.3 USB to PPM

The USB to PPM solution allows for the integration of commercial flight station devices and remote controllers.

Note: Veronte PPM to USB device is sold separately. Please contact sales@embention.com for more information.

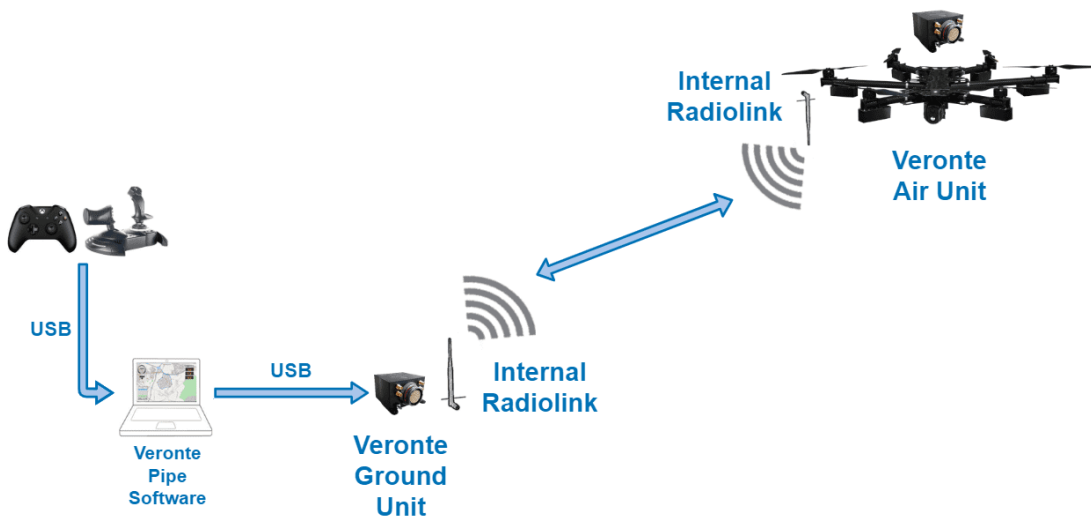


USB to PPM

2.2.4 USB to Pipe

Allows the use of any device that is detected as a remote controller by the operative system

Warning: This setup is dependent on external hardware and software (PC) and any potential issue will also affect the availability of the stick. It is recommended to use this solution for non-critical systems only. For critical systems (i.e. flight control), use the USB to PPM option instead.



USB to Pipe

2.2.5 Virtual Stick

The Virtual stick feature allows to integrate as a stick controller any device that can interface with Veronte Ground or Air unit (RS232, RS485, ADC, CAN,...) and can provide control reference values.

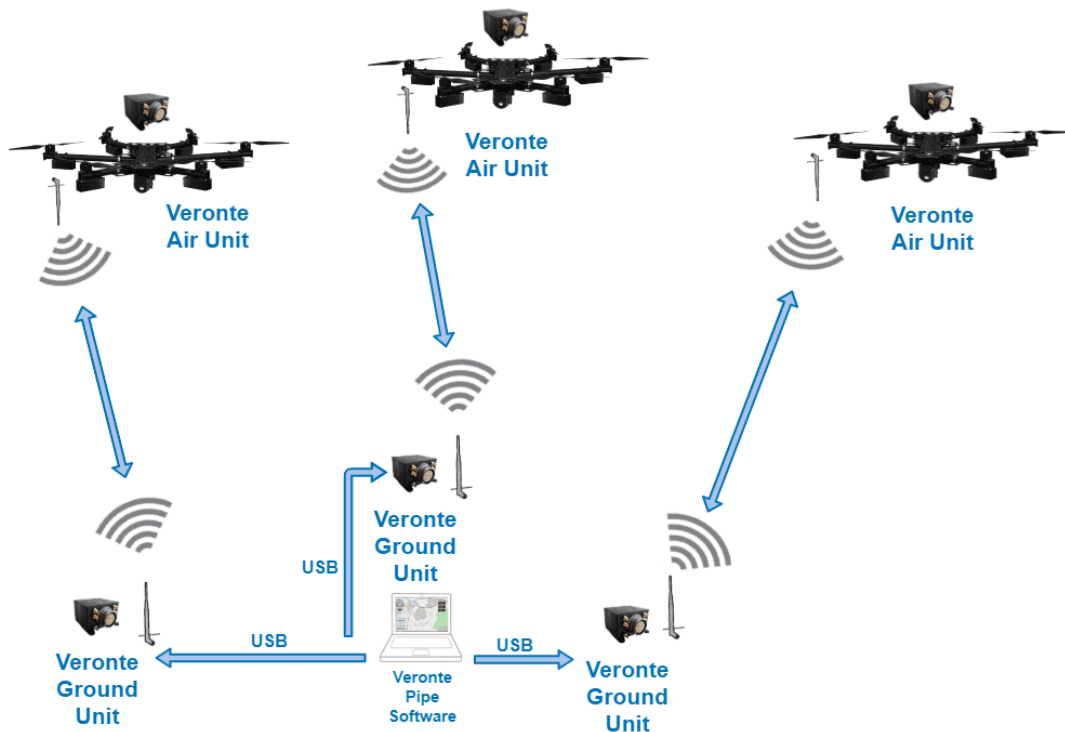
While the configuration will be slightly more complex, this feature allows using a wide variety of devices as flight control interfaces.

2.3 Point to Multipoint Layouts

Due to Veronte's modular configuration, it is possible to integrate several air and ground units within the same network.

2.3.1 Point to Point with single Pipe

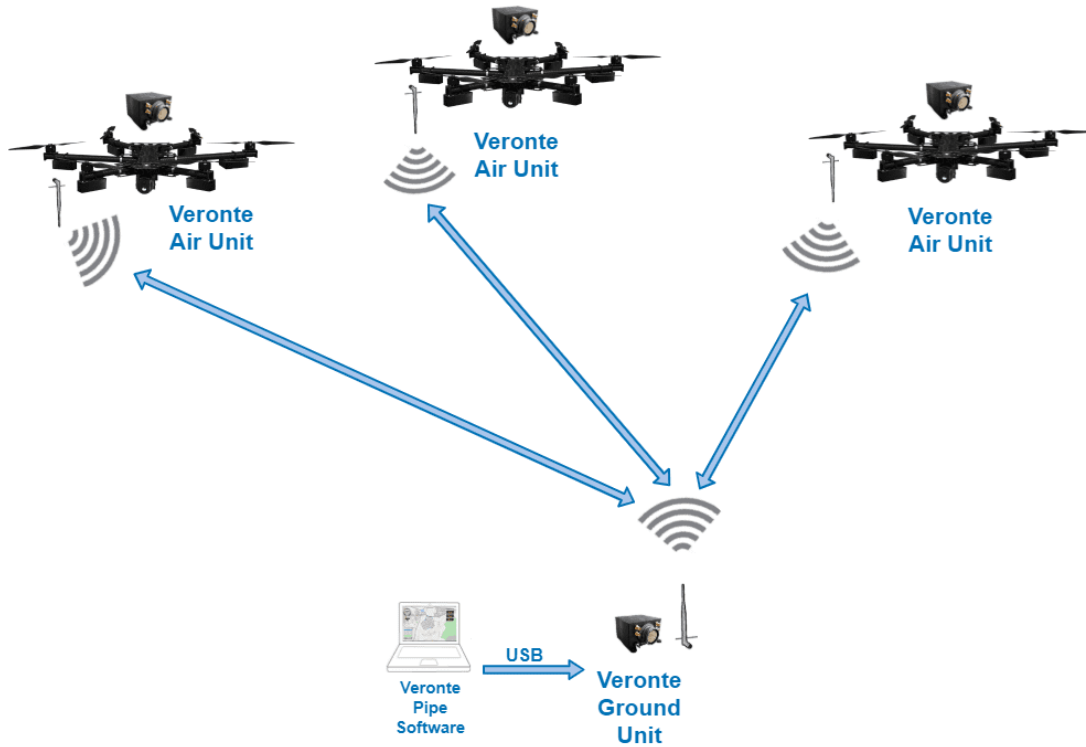
Standard multiplatform setup.



Multiple Air units - Point to Point

2.3.2 Point to Multipoint with single GS

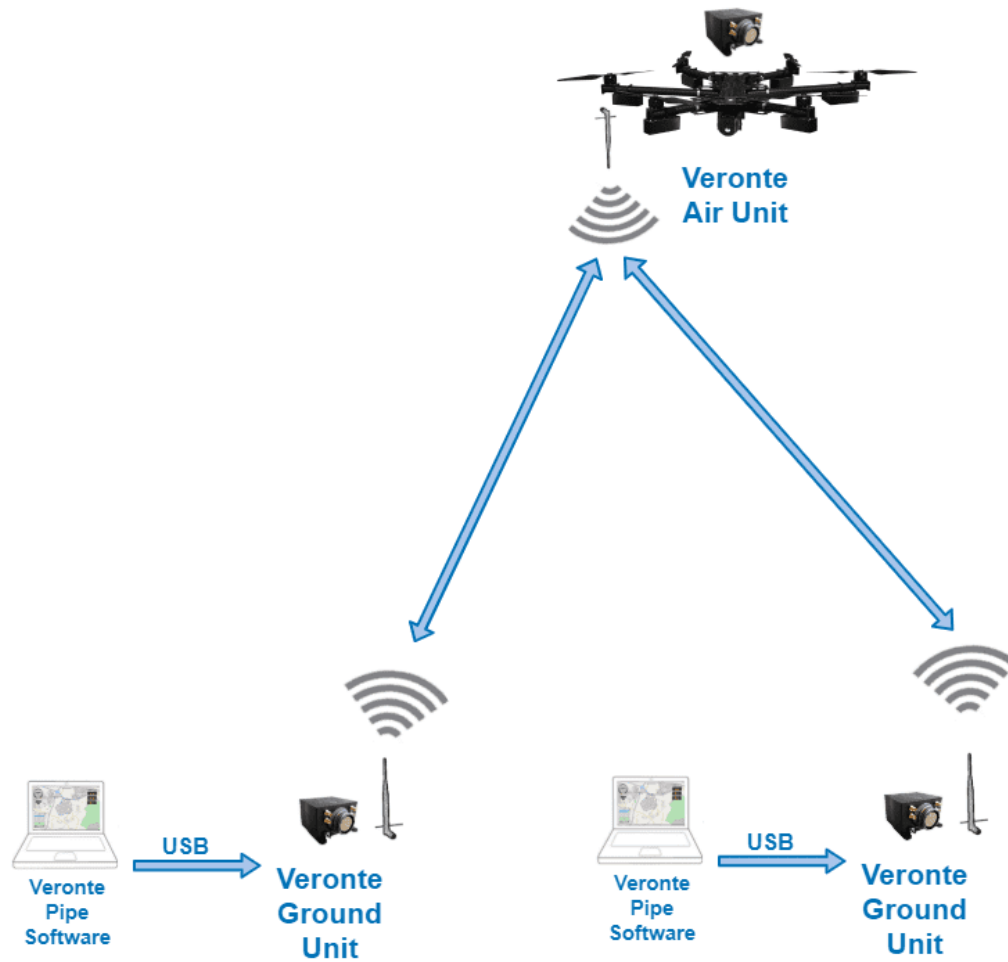
For managing several platforms with a single radiolink



Multiple Air units - Point to Multipoint

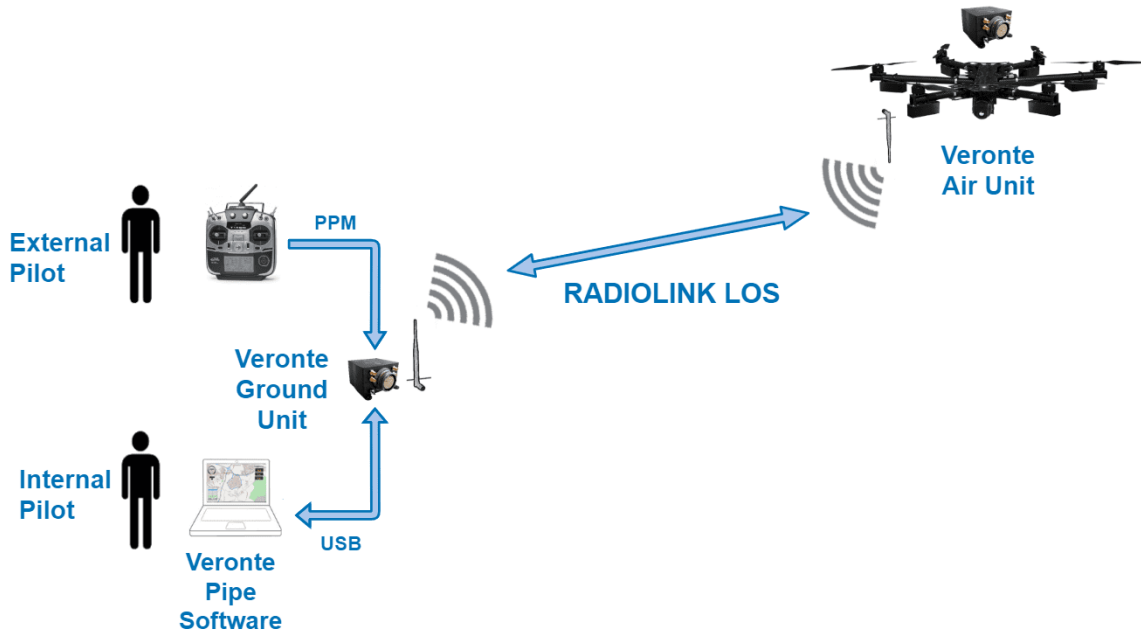
2.3.3 Multipoint to Point with multiple GS

For long range operations with several LOS stations



Multiple Ground Stations

The following image shows the **standard Veronte System Layout** for operation:



Veronte Standard Layout

In the standard layout, an **Operator (Internal Pilot)** controls the UAV from the **Ground Station** using **Veronte Pipe Software**.

Additionally, a **Safety Pilot (External Pilot)** is connected to the **Ground Station** using an RC Controller. The stick commands are read by the **Ground Unit** and re-routed to the **Air Unit**. The **Safety Pilot** is able to take control of the flight at any point using an *Automation*

While this is the most common setup, there is a wide variety of options, including:

- BLOS (Beyond Line-of-sight) communications
- Onboard RC receivers
- Point-to-Multipoint configurations

HARDWARE INSTALLATION

3.1 Veronte Autopilot

3.1.1 Aircraft Mounting

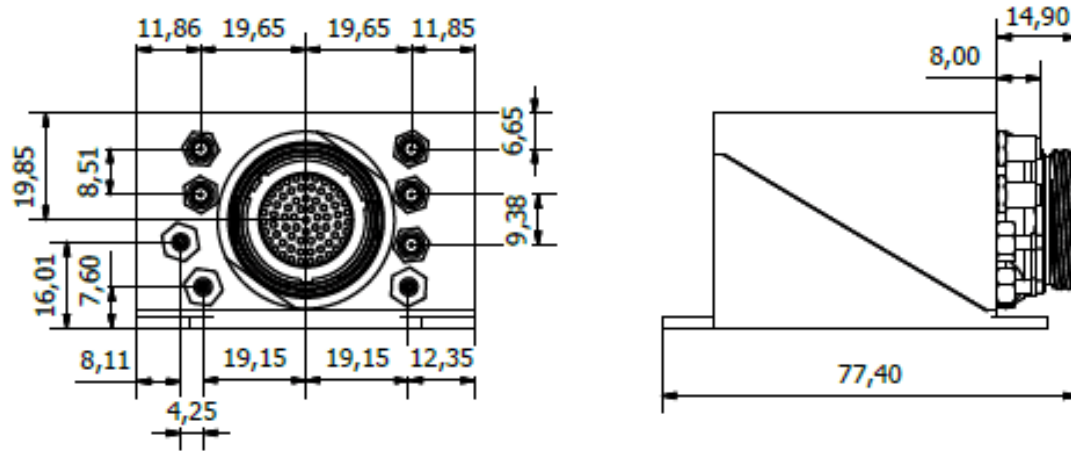
There are two versions of Veronte autopilot: with or without enclosure.

3.1.1.1 Enclosure

Veronte is provided using an anodized aluminium enclosure with enhanced EMI shielding and IP protection. A high reliability connector is also provided in this version. The total approximate weight is 190 g.



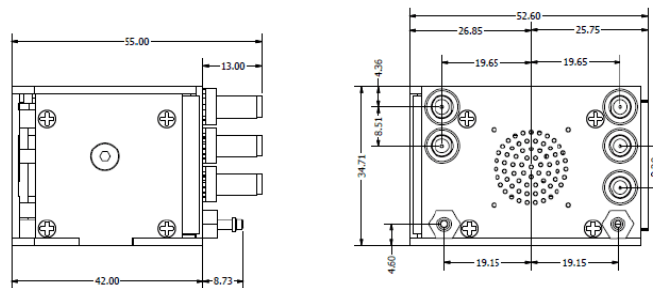
Veronte with aluminium enclosure



Veronte dimensions

3.1.1.2 OEM

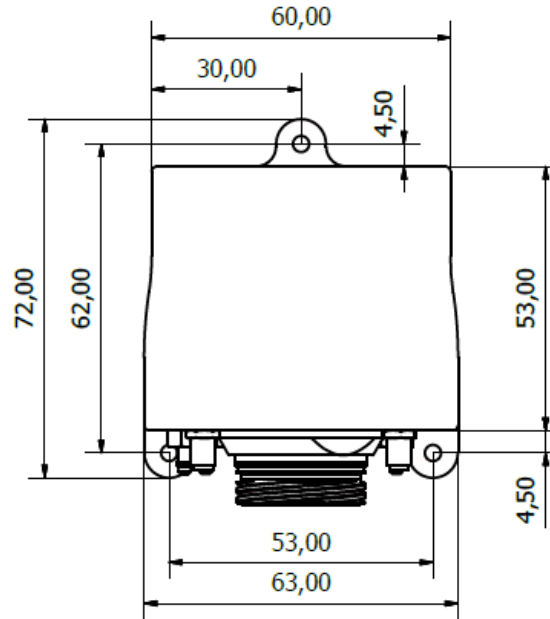
Veronte can be provided in OEM version too. The total approximate weight is 90g.



OEM dimensions

3.1.1.3 Mechanical Mounting

M3 screws are recommended for mounting. In saline environments such as coastal and oceanic, the screw material must be stainless steel.



Veronte Top View

3.1.1.4 Vibration Isolation

Although Veronte ultimately rejects noise and high-frequency modes of vibration with electronic filters and internal mechanical filters, there might be situations where external isolation components might be needed.

Veronte can be mounted in different ways in order to reject the airframe vibration. The simplest could be achieved by just using double-sided tape on the bottom side of Veronte. Other ways may use some external structure which could be rigidly attached to the airframe and softly attached to Veronte (e.g. foam, silent blocks, gel, etc)



Veronte mounts

The user should take into account that wiring should be loose enough so vibrations may not be transmitted to Veronte.

In cases where Veronte isolation is not viable, it is possible to use soft engine mounts. It is also recommended when there are other sensible payloads like video cameras or for high vibration engines.

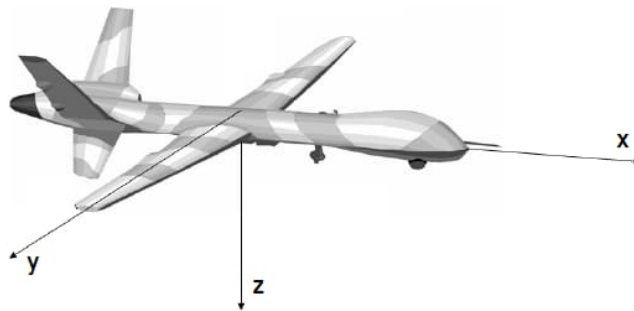
3.1.1.5 Location

The location of Veronte has no restrictions. You only need to configure its relative position with respect to the centre of mass of the aircraft and the GNSS antenna. The configuration of the location of Veronte can be easily configured using Veronte Pipe Software.

3.1.1.6 Orientation

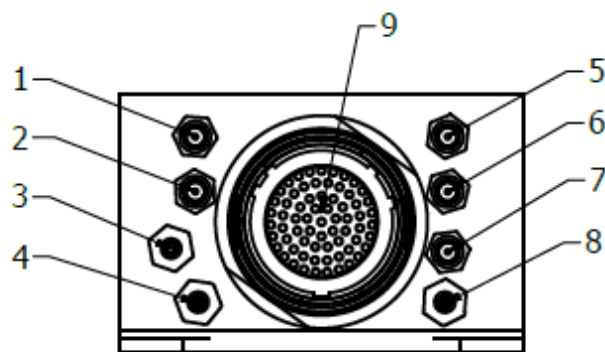
The orientation of Veronte has no restrictions either. You only need to configure Veronte axes with respect to the aircraft body axes by means of a rotation matrix or a set of correspondences between axes. The configuration of the orientation of Veronte can be easily configured using Veronte Pipe Software.

Veronte axes are printed on the box and aircraft coordinates are defined by the standard aeronautical conventions (see image below)



Aircraft Coordinates (Standard Aeronautical Convention)

3.1.1.7 Connector Layout



Veronte Connectors

| Index | Connector |
|-------|---|
| 1 | LOS SSMA connector |
| 2 | GNSS1 SSMA connector |
| 3 | Static pressure port (Fitting 5/64in) [for MS56 and DPS310 sensors] |
| 4 | Static pressure port (Fitting 5/64in) [for HSC sensor] |
| 5 | M2M SSMA connector |
| 6 | GNSS2 SSMA connector |
| 7 | TPDR (Transponder) SSMA connector |
| 8 | Dynamic pressure port (Fitting 5/64in) |
| 9 | 68-pin connector |

For the pressure ports, mating with clamped 2mm internal diameter flexible tubing is recommended.

The two static pressure ports must be used for sensor redundancy (Y tubing connection is strongly recommended).

3.1.1.8 Mating Connector

| Index | Connector | Mating Connector |
|-------|---------------------------------|---|
| 1 | RF antenna (SSMA Jack Female) | SSMA male Plug, low-loss cable is recommended. |
| 2,6 | GNSS antenna (SSMA Jack Female) | SSMA male Plug, low-loss cable is recommended. Active Antenna GNSS: Gain min 15dB (to compensate signal loss in RF Cable) max 50dB, maximum noise figure 1.5dB, power supply 3.3V max current 20 mA |
| 5 | M2M antenna (SSMA Jack Female) | SSMA male Plug, low-loss cable is recommended. |
| 7 | TPDR antenna (SSMA Jack Female) | SSMA male Plug, low-loss cable is recommended. |
| 9 | Connector HEW.LM.368.XLNP | Mating connector P/N: FGW.LM.368.XLCT Mating harness is available on demand. |

3.1.1.9 Antenna Integration

The system uses different kinds of antennas to operate that must be installed on the airframe. Here you can find some advice for obtaining the best performance and for avoiding antenna interferences.

| Antenna Installation |
|---|
| Maximize separation between antennas as much as possible. |
| Keep them far away from alternators or other interference generators. |
| Always isolate antenna ground panel from the aircraft structure. |
| Make sure the antenna is securely mounted. |
| Always use high-quality RF wires minimising the wire length. |
| Always follow the antenna manufacturer manual. |
| SSMA connections shall be tightened applying 1Nm of torque |
| For all-weather aircraft, insert SSMA lightning protectors. |

| |
|---|
| GNSS Antenna |
| Antenna top side must point the sky. |
| Install it on a top surface with direct sky view. |
| Never place metallic / carbon parts or wires above the antenna. |
| It is recommended to install it on a small ground plane. |
| For all-weather aircraft, insert SSMA lightning protectors. |

3.1.1.10 Pressure lines

Veronte has three pressure input lines, two for static pressure to determine the absolute pressure and one for pitot in order to determine the dynamic pressure.

| |
|--|
| Pressure Intake |
| Pressure intakes must be located in order to prevent clogging. |
| Never install pressure intakes on the propeller flow. |
| Design pressure tubing path in order to avoid tube constriction. |

| |
|---|
| Static Pressure |
| It is not recommended to use inside fuselage pressure if it is not properly vented. |

| |
|--|
| Pitot Tube |
| Pitot tube must be installed facing the airflow. |
| It is recommended to install it near the aircraft's x axis in order to avoid false measures during manoeuvres. |
| For low-speed aircraft it is recommended at least 6.3mm tubes to prevent any rain obstruction. |

3.1.2 Electrical

3.1.2.1 Power

Veronte can use unregulated DC (**6.5V to 36V**). Pins used for power and ground are the same for both Ground and Air configurations.

LiPo batteries between 2S and 8S can be used without regulation needs. Remaining battery level can be controlled by the internal voltage sensor and by configuring the voltage warnings on the Pipe software.

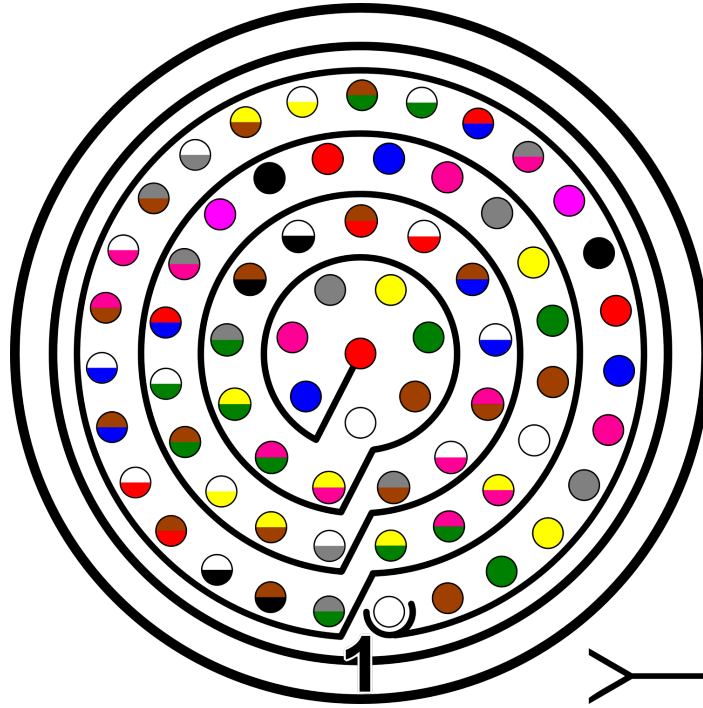
For higher voltage installations, voltage regulators must be used. For dimensioning voltage regulators take into account that a blocked servo can activate regulator thermal protection.

| |
|--|
| Warning: Caution!! Power Veronte out of the given range can cause irreversible damage to the system. Please read carefully the manual before powering the system. |
|--|

Veronte and servos can be powered by the same or different batteries. In case of having more than one battery on the system, a single point ground union is needed to ensure a good performance. The ground signal should be isolated from other noisy ground references (e.g. engines). If all grounds need to be connected, the connection should be made on the negative pole of the battery.

It is recommendable to use independent switches for autopilot and motor/actuators. During the system initialization, the PWM signal will be set to low level (0V), please make sure that actuators/motor connected support this behaviour before installing a single switch for the whole system.

3.1.2.2 Veronte I/O Signals



68 pin connector for Veronte Autopilot (frontal view)

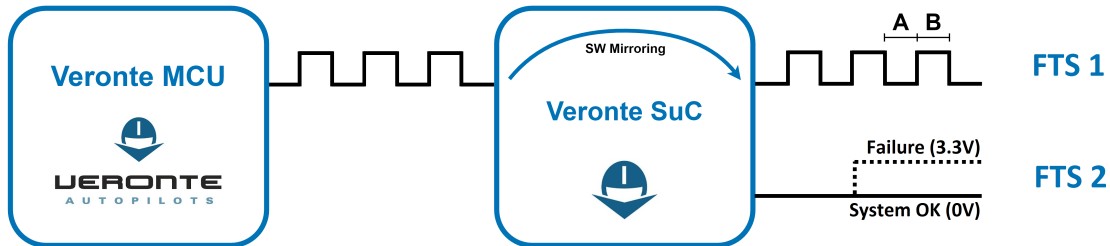
| Pin | Signal | Type | Comments |
|-----|-----------|--------------|--|
| 1 | I/O1 | I/O | PWM / Digital I/O signal (0-3.3V). Protected against ESD and short circuit |
| 2 | I/O2 | I/O | PWM / Digital I/O signal (0-3.3V). Protected against ESD and short circuit |
| 3 | I/O3 | I/O | PWM / Digital I/O signal (0-3.3V). Protected against ESD and short circuit |
| 4 | I/O4 | I/O | PWM / Digital I/O signal (0-3.3V). Protected against ESD and short circuit |
| 5 | I/O5 | I/O | PWM / Digital I/O signal (0-3.3V). Protected against ESD and short circuit |
| 6 | I/O6 | I/O | PWM / Digital I/O signal (0-3.3V). Protected against ESD and short circuit |
| 7 | I/O7 | I/O | PWM / Digital I/O signal (0-3.3V). Protected against ESD and short circuit |
| 8 | I/O8 | I/O | PWM / Digital I/O signal (0-3.3V). Protected against ESD and short circuit |
| 9 | GND | GROUND | Ground signal for actuators 1-8 |
| 10 | I/O9 | I/O | PWM / Digital I/O signal (0-3.3V). Protected against ESD and short circuit |
| 11 | I/O10 | I/O | PWM / Digital I/O signal (0-3.3V). Protected against ESD and short circuit |
| 12 | I/O11 | I/O | PWM / Digital I/O signal (0-3.3V). Protected against ESD and short circuit |
| 13 | I/O12 | I/O | PWM / Digital I/O signal (0-3.3V). Protected against ESD and short circuit |
| 14 | I/O13 | I/O | PWM / Digital I/O signal (0-3.3V). Protected against ESD and short circuit |
| 15 | I/O14 | I/O | PWM / Digital I/O signal (0-3.3V). Protected against ESD and short circuit |
| 16 | I/O15 | I/O | PWM / Digital I/O signal (0-3.3V). Protected against ESD and short circuit |
| 17 | I/O16 | I/O | PWM / Digital I/O signal (0-3.3V). Protected against ESD and short circuit |
| 18 | GND | GROUND | Ground signal for actuators 9-16 |
| 19 | RS 232 TX | Output | RS 232 Output (-13.2V to 13.2V Max, -5.4V to 5.4V Typical). Protected against ESD and short circuit |
| 20 | RS 232 RX | Input | RS 232 Input (-25V to 25V Max, -0.6V Low and 2.4V High Threshold). Protected against ESD and short circuit |
| 21 | GND | GROUND | Ground signal for buses |
| 22 | Analog 4 | Input Analog | Input 0-3V. Protected against ESD and short circuit |
| 23 | Analog 5 | Input Analog | Input 0-3V. Protected against ESD and short circuit |

Table 1 – continued from previous page

| Pin | Signal | Type | Comments |
|-----|--------------|--------|--|
| 24 | GND | GROUND | Ground signal for buses |
| 25 | CanA P | I/O | CANbus interface, up to 1Mbps (2.3V Typical, 1.2V-2.3V Differential). Protected against ESD |
| 26 | CanA N | I/O | Twisted pair with a 120 Z_0 recommended (2.3V Typical, 1.2V-2.3V Differential). Protected against ESD |
| 27 | GND | GROUND | Ground signal for buses |
| 28 | CANB_P | I/O | CANbus interface. It supports data rates up to 1 Mbps. Protected against ESD |
| 29 | CANB_N | I/O | Twisted pair with a 120 Z_0 recommended. Protected against ESD |
| 30 | GND | GROUND | Ground signal for buses |
| 31 | I2C_CLK | Output | Clk line for I2C bus (0.3V to 3.3V). Protected against ESD and short circuit |
| 32 | I2C_DATA | I/O | Data line for I2C bus (0.3V to 3.3V). Protected against ESD and short circuit |
| 33 | GND | GROUND | Ground for 3.3V power supply |
| 34 | 3.3V | POWER | 3.3V - 100mA power supply. Protected against ESD short circuit with 100mA resettable fuse |
| 35 | GND | GROUND | Ground for 5V power supply |
| 36 | 5V | POWER | 5V - 100mA power supply. Protected against ESD short circuit with 100mA resettable fuse |
| 37 | GND | GROUND | Ground for analog signals |
| 38 | ANALOG_1 | Input | Analog input 0-3V. Protected against ESD and short circuit |
| 39 | ANALOG_2 | Input | Analog input 0-3V. Protected against ESD and short circuit |
| 40 | ANALOG_3 | Input | Analog input 0-3V. Protected against ESD and short circuit |
| 41 | GND | GROUND | Ground for FTS signals |
| 42 | FTS1_OUT | Output | Deadman signal from comicro. Protected against ESD and short circuit |
| 43 | FTS2_OUT | Output | !SystemOK Bit. Protected against ESD and short circuit |
| 44 | GND | GROUND | Ground signal for safety buses |
| 45 | V_ARB_TX | Output | Veronte comicro UART output to activate safety mechanism. Protected against ESD and short circuit |
| 46 | V_ARB_RX | Input | Veronte comicro UART output to activate safety mechanism. Protected against ESD and short circuit |
| 47 | GND | GROUND | Ground signal comicro power supply |
| 48 | V_ARB_VCC | POWER | Veronte comicro power (6.5V to 36V). Protected against ESD and reverse polarity |
| 49 | FTS3_OUT_MPU | Output | MPU alive voting signal, to use with 4xVeronte. It is a Square Wave at [100,125] Hz. Protected against ESD |
| 50 | OUT_RS485_P | Output | Non-inverted output from RS485 bus (-7V to 12V Max, -2.3V to 2.3V Typical). Protected against ESD |
| 51 | OUT_RS485_N | Output | Inverted output from RS485 bus (-7V to 12V Max, -2.3V to 2.3V Typical). Protected against ESD |
| 52 | IN_RS485_N | Input | Inverted input from RS485 bus (-7V to 12V Max, -2.3V to 2.3V Typical). Protected against ESD |
| 53 | IN_RS485_P | Input | Non-inverted output from RS485 bus (-7V to 12V Max, -2.3V to 2.3V Typical). Protected against ESD |
| 54 | RS-485_GND | GND | Ground for RS-485 bus |
| 55 | EQEP_A | I/O | DIGITAL output / DIGITAL input / Encoder quadrature input A (0-3.3V). Protected against ESD |
| 56 | EQEP_B | I/O | DIGITAL output / DIGITAL input / Encoder quadrature input B (0-3.3V). Protected against ESD |
| 57 | EQEP_S | I/O | DIGITAL output / DIGITAL input / Encoder strobe input (0-3.3V). Protected against ESD |
| 58 | EQEP_I | I/O | DIGITAL output / DIGITAL input / Encoder index input A (0-3.3V). Protected against ESD |
| 59 | GND | GROUND | Ground for encoders |
| 60 | V_USB_DP | I/O | Veronte USB data line. Protected against ESD |
| 61 | V_USB_DN | I/O | Veronte USB data line. Protected against ESD |
| 62 | V_USB_ID | I/O | Veronte USB ID line. Protected against ESD and short circuit |
| 63 | FTS_OUT_MPU | Output | Abort mission voting signal from MPU, to use with 4xVeronte. Bit Low (0V) if mission OK |
| 64 | FTS2_OUT_MPU | Output | Abort mission voting signal 2 from MPU, to use with 4xVeronte. Bit Low (0V) if mission OK |
| 65 | GND | GROUND | Veronte ground input |
| 66 | ND | GROUND | Veronte ground input |
| 67 | VCC | POWER | Veronte power supply (6.5V to 36V). Protected against ESD and reverse polarity |
| 68 | VCC | POWER | Veronte power supply (6.5V to 36V). Protected against ESD and reverse polarity |

Warning: Remember!! All Veronte's GND pins are common.

3.1.2.3 Flight Termination System (FTS)



Flight Termination System

Veronte integrates two different FTS pins (42 and 43):

FTS1 - Deadman (Pin 42): On this pin, Veronte outputs a square wave with $A = \sim 5\text{ms}$ and $B = \sim 5\text{ms}$ (3.3V). Its frequency can be higher right after the rebooting (around 300-400Hz), but A and B must be always $< 8\text{ms}$.

FTS2 - !SystemOK (Pin 43): Its output is 0V when the system is working as expected and 3.3V when some error is detected. In detail, pin 43 goes high if $A > 8\text{ms}$ or $B > 8\text{ms}$ in the deadman signal sent by the Main Processor Unit (MPU).

3.1.2.4 Joystick

To use the joystick in the system, connect the PPMout of the trainer port to a digital input of Veronte and configure that digital input as the radio input in Pipe.

If the PPM level is 3.3V, pins 1-8, 10-17 and 55-58 pins can be used.

Veronte is compatible with standard Pulse Position Modulation (PPM) signals, Futaba radios between 8 and 12 channels are recommended.

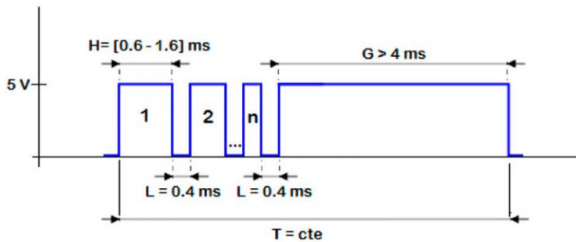


Futaba T10 Joystick

| Pin | Designation | Connector |
|--------|----------------------|-----------|
| SHIELD | GROUND | |
| 1 | V _{ENCODER} | |
| 2 | PPM _{OUT} | |
| 3 | PPM _{IN} | |
| 4 | V _{ENC2} | |
| 5 | V _{BATTERY} | |
| 6 | UNKNOWN | |

| Pin | Designation | Connector |
|-----|----------------------|-----------|
| 1 | NC | |
| 2 | GROUND | |
| 3 | PPM _{OUT} | |
| 4 | V _{BATTERY} | |
| 5 | V _{ENCODER} | |
| 6 | PPM _{IN} | |

Futaba T10 pinout

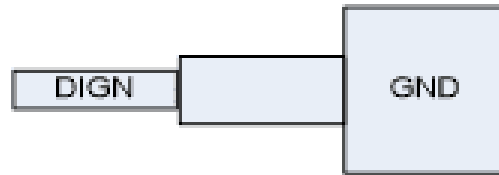


PPM signal

As default, channel 8 is reserved for manual / auto switch. High level is used for automatic flight and low level for manual control. This channel can be configured on Veronte Pipe.

Warning: Caution!! PPM signal must be into the Veronte voltage ranges. Some joysticks may need an adaptation board, please ask our team to check compatibility.

Veronte connector for CS is provided with 3.5mm stereo plug connector as follows:



PPM pinout

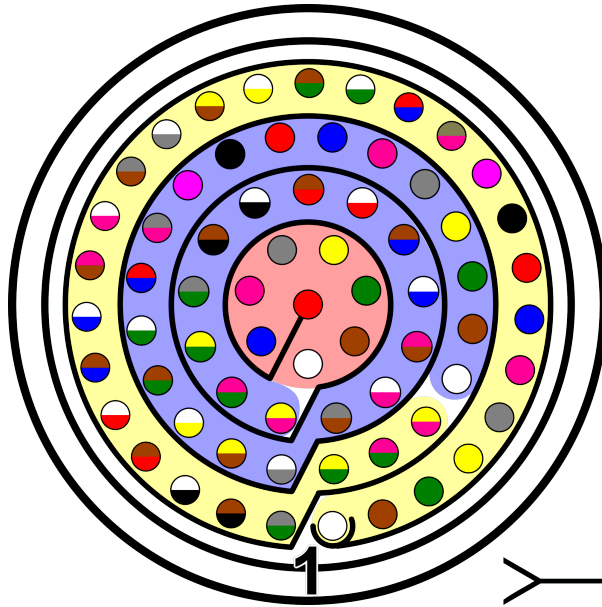


PPM connector

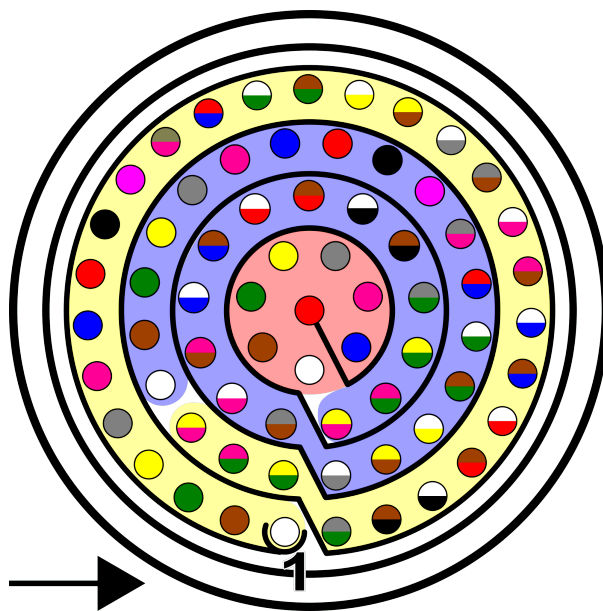
3.1.3 Performances

| Variable | Value |
|---------------------------------------|---|
| Weight (with enclosure and connector) | 190 g |
| Weight (OEM) | 90 g |
| Power Input | 6.5 V to 36 V |
| Minimum Temperature | -40 °C |
| Maximum Temperature | +55°C (No convection, ask for increased limits (up to 71°C)) |
| Max. Internal Temperature | +85°C |
| Minimum Pressure | 0 kPa |
| Maximum Pressure | 104 kPa |
| Maximum Dynamic pressure | 6 kPa (Ask for increased limits (up to 50kPa)) |
| Protection Rating | IP67 enclosure version |
| Acceleration Limits (3 axes) | ± 2 g to ± 16 g (for sustained maneuvers, transitional higher accelerations are possible (e.g. catapult launch). Ask for increased limits.) |
| Angular Velocity Limits (3 axes) | ± 125 deg/s to ± 2000 deg/s (for sustained maneuvers, transitional higher angular velocities are possible. Ask for increased limits.) |
| Magnetic Field Limits (3 axes) | ± 4 to ± 16 Gauss |
| GNSS | 72 channels, GPS L1C/A, GLONASS L1OF, BeiDou B1I |
| Datalink | 410 to 480 MHz licensed or FHSS/902-928MHz FHSS/2.4 to 2.483 GHz ISM Band/869.5-869.75 MHz ISM Band |

3.1.4 Annex 1: Connector colour code



Connector HEW.LM.368.XLNP



CS harness plug

Warning: Check the pin number before connecting. The colour code is repeated 3 times due to the amount of pins. First section (yellow) corresponds to pins 1-30, the second section (blue) to pins 31-60 and the third one (red) to pins 61-68. Pin number increases following the black line of the pictures above: counterclockwise for the connector and clockwise for the plug.

| PIN | Color code | PIN | Color code |
|-----|----------------|-----|----------------|
| 1 | White | 35 | Gray |
| 2 | Brown | 36 | Pink |
| 3 | Green | 37 | Blue |
| 4 | Yellow | 38 | Red |
| 5 | Gray | 39 | Black |
| 6 | Pink | 40 | Violet |
| 7 | Blue | 41 | Gray – Pink |
| 8 | Red | 42 | Red – Blue |
| 9 | Black | 43 | White – Green |
| 10 | Violet | 44 | Brown – Green |
| 11 | Gray – Pink | 45 | White – Yellow |
| 12 | Red – Blue | 46 | Yellow – Brown |
| 13 | White – Green | 47 | White – Gray |
| 14 | Brown – Green | 48 | Gray – Brown |
| 15 | White – Yellow | 49 | White – Pink |
| 16 | Yellow – Brown | 50 | Pink – Brown |
| 17 | White – Gray | 51 | White – Blue |
| 18 | Gray – Brown | 52 | Brown – Blue |
| 19 | White – Pink | 53 | White – Red |
| 20 | Pink – Brown | 54 | Brown – Red |
| 21 | White – Blue | 55 | White – Black |
| 22 | Brown – Blue | 56 | Brown – Black |

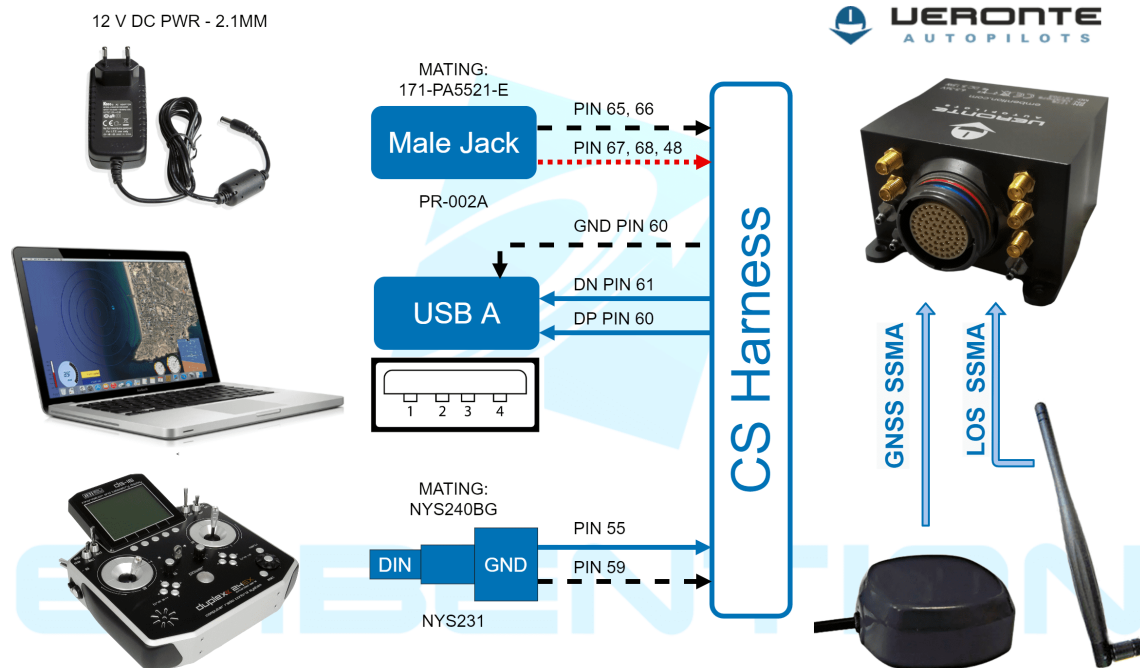
continues on next page

Table 2 – continued from previous page

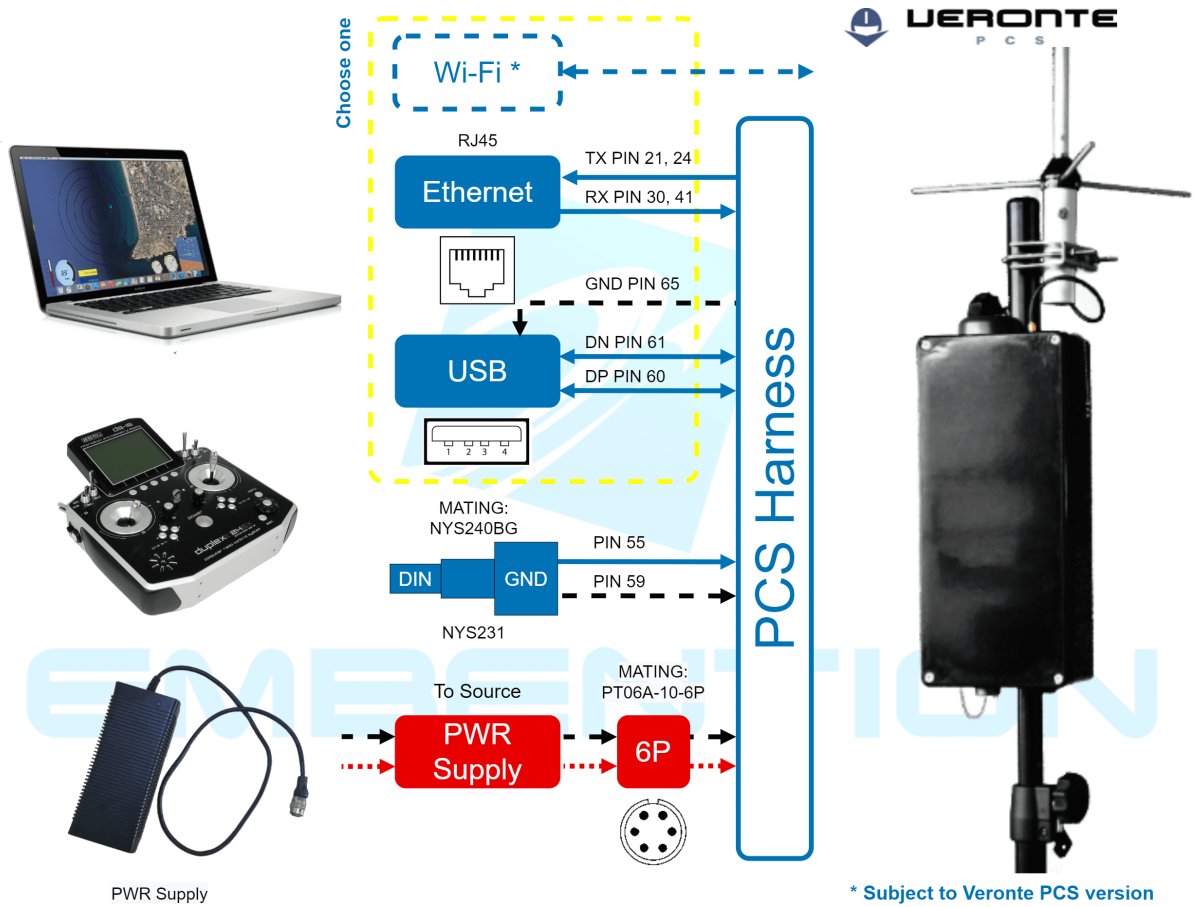
| PIN | Color code | PIN | Color code |
|-----|----------------|-----|----------------|
| 23 | White – Red | 57 | Gray – Green |
| 24 | Brown – Red | 58 | Yellow – Green |
| 25 | White – Black | 59 | Pink – Green |
| 26 | Brown – Black | 60 | Yellow – Pink |
| 27 | Grey – Green | 61 | White |
| 28 | Yellow – Green | 62 | Brown |
| 29 | Pink – Green | 63 | Green |
| 30 | Yellow – Pink | 64 | Yellow |
| 31 | White | 65 | Grey |
| 32 | Brown | 66 | Pink |
| 33 | Green | 67 | Blue |
| 34 | Yellow | 68 | Red |

3.1.5 Annex 2: Connection examples

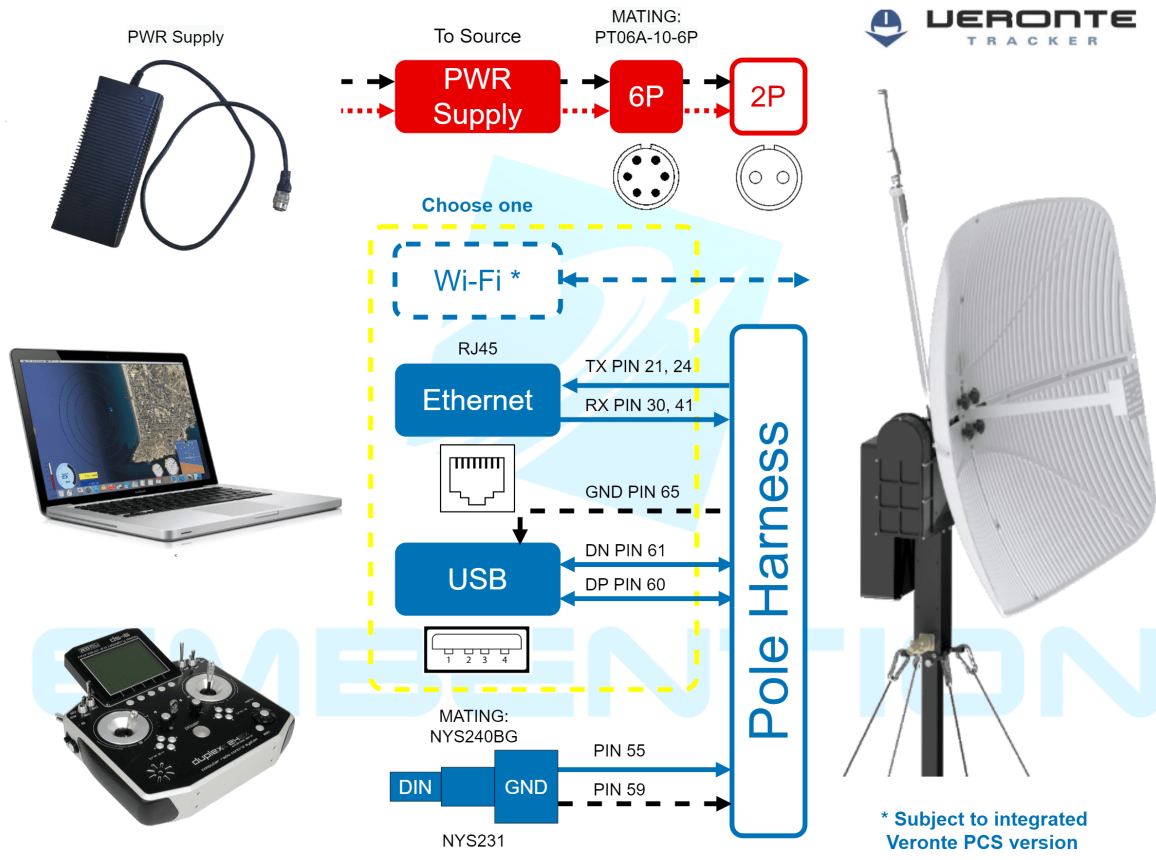
3.1.5.1 Ground Stations



Basic Veronte Autopilot Ground Station



Veronte PCS Ground Station



Veronte Tracker Ground Station



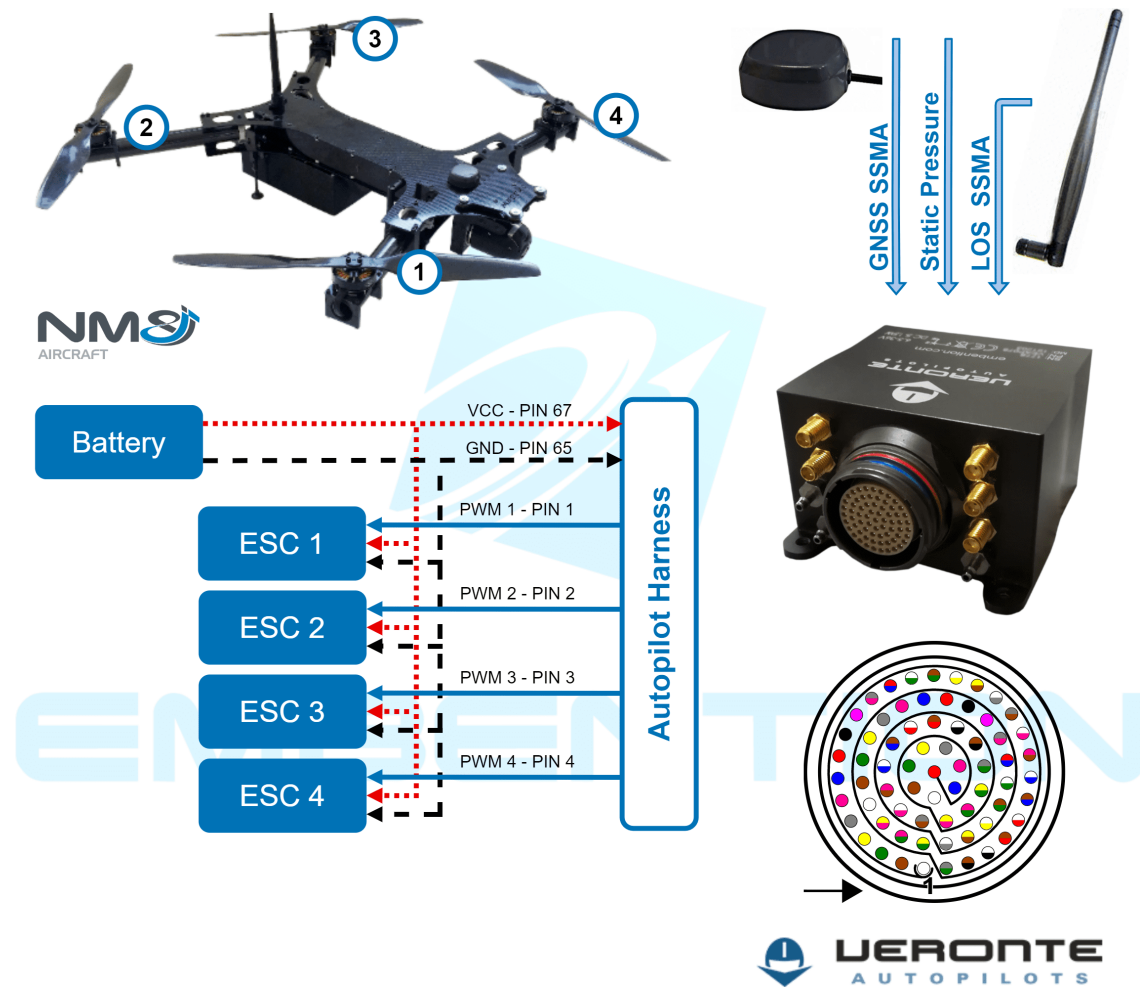
* Subject to hardware version

Check individual equipment diagrams for further details

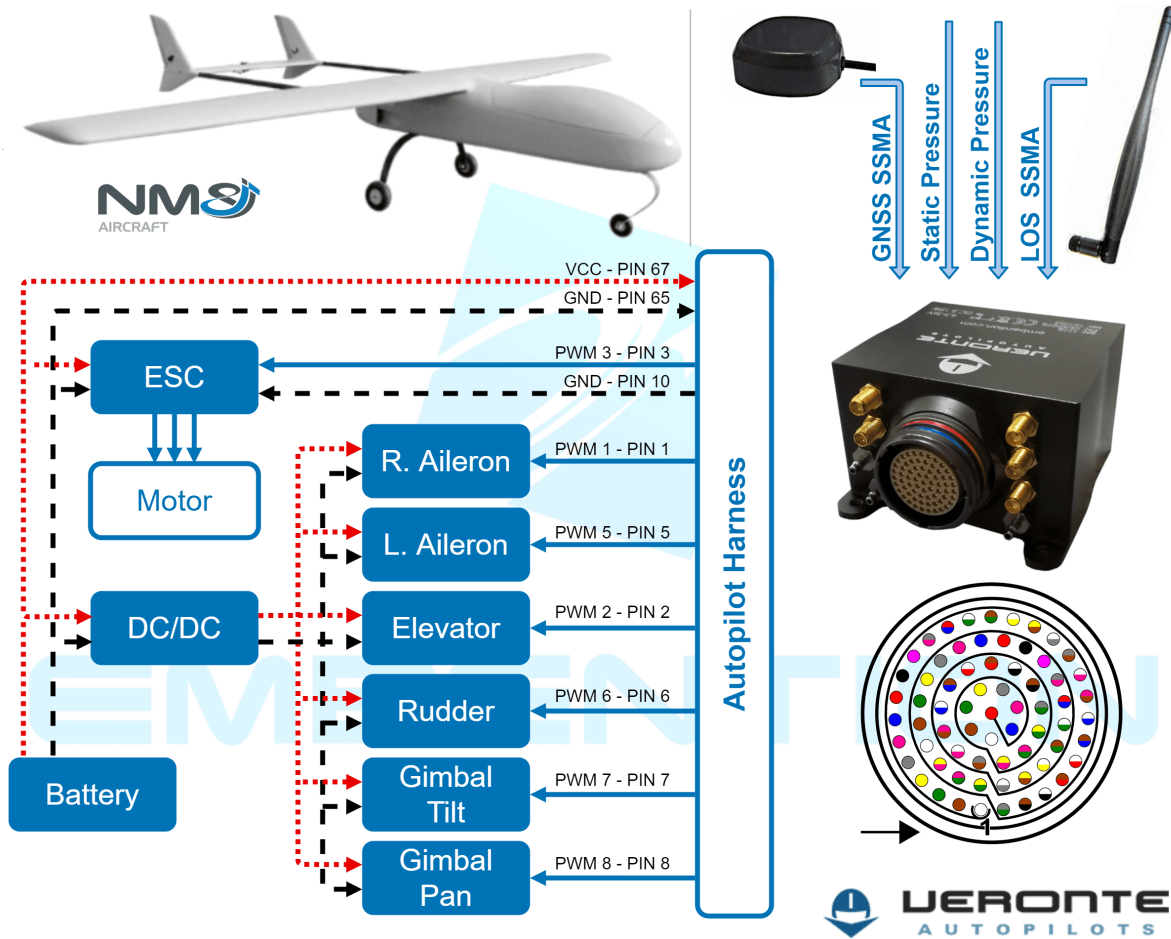
Veronte MCS Ground Station

Warning: Veronte equipment harnesses have specific pin layouts. Only use their own matting connectors, do NOT mix harnesses: misuse can lead to destruction.

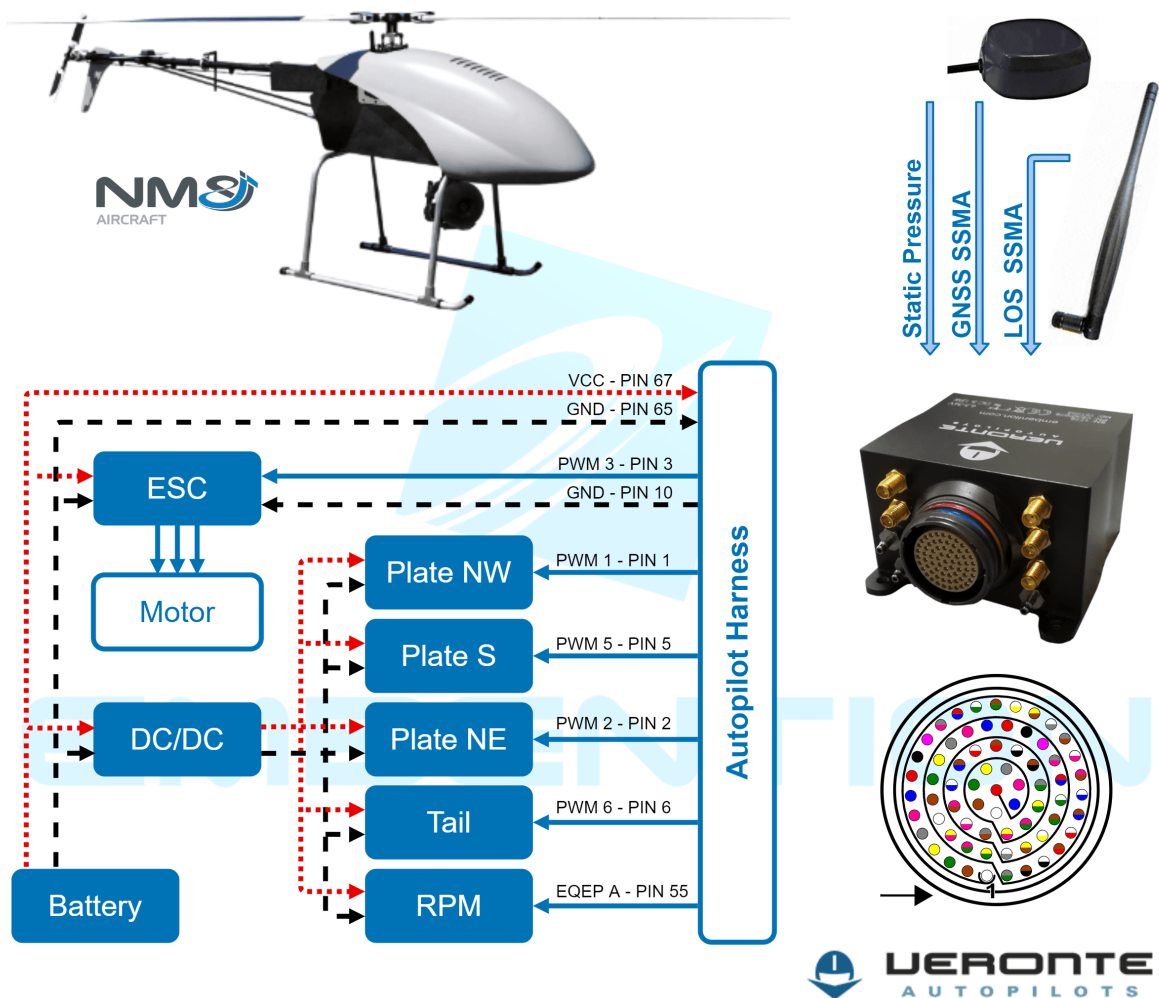
3.1.5.2 Aircrafts



Multicopter



Fixed Wing Airplane



Helicopter



Veronte Autopilot

Veronte Autopilot is a miniaturized high-reliability avionics system for advanced control of unmanned systems. This control system embeds a state-of-the-art suite of sensors and processors together with **LOS and BLOS M2M datalink** radio, all with reduced size and weight.

3.1.6 Operation

The unique **Plug 'n Fly** control system, Veronte Autopilot adds fully autonomous control capabilities to any unmanned system for complete operation, compatible with: UAV, Drone, RPAS, USV, UGV...

- **Highly configurable:** Veronte control system is fully configurable; payload, platform layout, control phases, control channels... even the user interface layout can be user-defined.
- **Custom routines:** User selectable automatic actions, activated on system event or periodically.
 - **Actions:** phase change, activate payload, move servo, go to, onboard log, parachute release...
 - **Events:** waypoint arrival, inside/outside polygon, alarm, variable range, button...
- **Telemetry & log:** Embedded datalink for system monitoring and telecommand and customizable user log in both onboard and control station, all with user-defined variables and frequency record.
- **External sensor:** Support for external sensor connection: magnetometer, radar, LIDAR, RPM, temperature, fuel level, battery level, weather...
- **Payload & Peripheral:** Transponder, secondary radios, satcom transceivers, camera gimbals, motor drivers, photo cameras, flares, parachute release systems, tracking antennas, pass through RS232, RS485 & CAN tunnel...

3.1.7 Platforms

The Veronte Autopilot is designed to control any unmanned vehicle, either aircraft such as: multirotors, helicopters, airplanes, VTOL, blimps... as well as ground vehicles, surface vehicles or many others. Custom flight phases and control channels provide support for any aircraft layout and performance by using the same software and hardware for: UAS, RPAS, Drone, USV / ASV, UGV...



Veronte FCS overview

Veronte contains all the electronics and sensors needed in order to properly execute all the functions needed to control the UAV. A Veronte-based FCS contains the following elements:

- **Veronte (Air):** it executes in real time all the guidance, navigation and control algorithms for the carrying airframe, acting on the control surfaces and propulsion system and processing the signals from different sensors: accelerometers, gyroscopes, magnetometer, static pressure, dynamic pressure, GPS (EGNOS/Galileo compatible).

- Veronte (Ground): apart from linking to other flying Veronte units and supporting manual and arcade modes with conventional joysticks, it can also control a directional antenna in order to expand the maximum range. It communicates to Veronte Pipe (software for ground segment mission management).
- Veronte Pipe: software for mission management at the ground segment. It monitors flying vehicles in real time and can also reproduce past missions in an offline manner. It is also the graphical user interface where commands and flight plans are produced.

3.1.8 Safety

Veronte autopilot includes the following features in order to provide your UAS with the best safety performances:

- Redundant IMU.
- Redundant GNSS receiver.
- Redundant Pressure sensor.
- Dual core principal microprocessor + dissimilar safety microcontroller (comicro).

Independent power supply for main system and safety microcontroller.

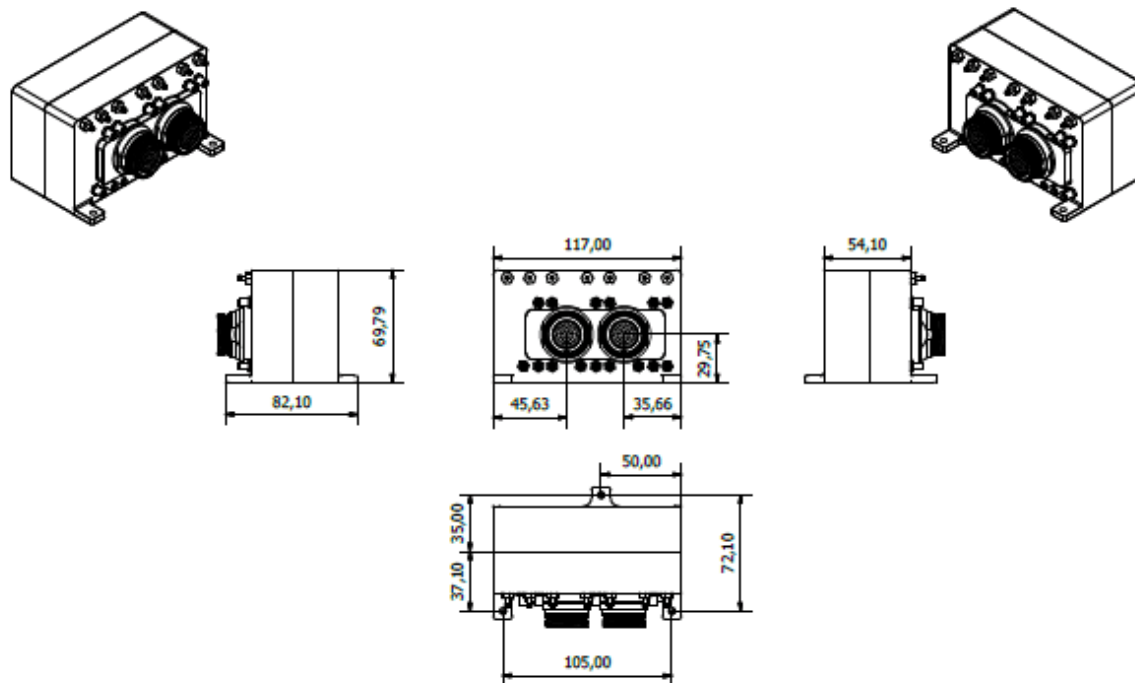
In case any malfunction occurs in the microprocessor, the comicro can activate different safety mechanism by means of 2 digital outputs and 1 serial port.

3.2 4x Veronte Autopilot

3.2.1 Aircraft Mounting

3.2.1.1 Enclosure

4xVeronte is provided using an anodized-aluminium enclosure with enhanced EMI shielding and IP protection. The approximate total weight including radio modules is 750g. The following figure show the dimensions of the enclosure. M4 screws are recommended for mounting.



4xVeronte dimensions (mm)

3.2.1.2 Vibration Isolation

Although Veronte rejects noise and modes of vibration with internal electronic and mechanical filters, an external vibration isolation might be needed depending of the vehicle.

Veronte can be mounted in different ways in order to reject the airframe vibration if needed. One way to avoid vibration would be the use of some external structure which could be rigidly attached to the airframe and softly attached to Veronte (e.g. foam, silent blocks, etc.)

The user should take into account that wiring should be loose enough so vibrations may not find another way to enter the aircraft system.

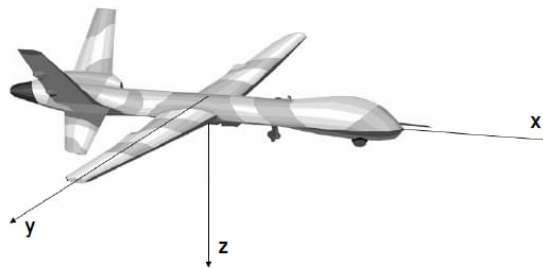
3.2.1.3 Location

The location of 4xVeronte has no restrictions. You only need to configure its relative position with respect to the centre of mass of the aircraft and the GNSS antenna. The configuration of the location of Veronte can be easily configured using Veronte Pipe Software.

3.2.1.4 Orientation

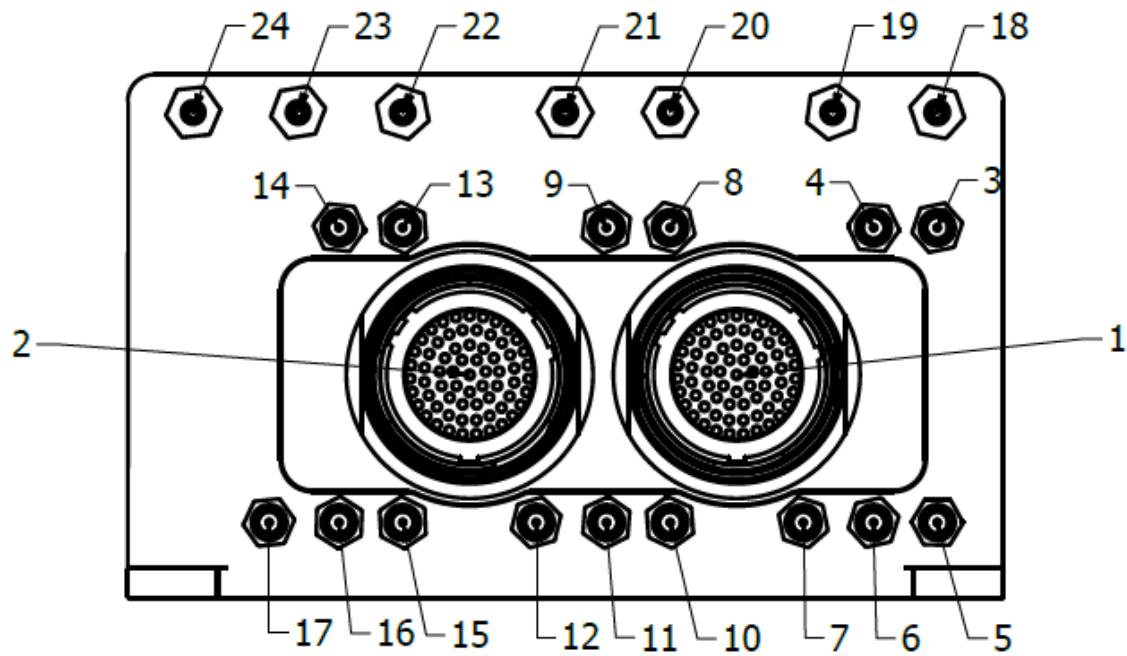
The orientation of 4xVeronte has no restrictions either. You only need to configure Veronte axes with respect to the aircraft body axes by means of a rotation matrix or a set of correspondences between axes. The configuration of the location of Veronte can be easily configured using Veronte Pipe Software.

Veronte axes are printed on the box and aircraft reference frame are defined by the standard flight dynamics conventions.



Aircraft Mounting

3.2.1.5 Connector Layout



4xVeronte Connectors

| Index | Connector |
|-------|--|
| 1 | Redundant (Critical) connector |
| 2 | Arbiter (Optional) connector |
| 3 | LOS SSMA connector for Veronte 3 |
| 4 | GNSS1 SSMA connector for Veronte 3 |
| 5 | M2M SSMA connector for Veronte 3 |
| 6 | GNSS2 SSMA connector for Veronte 3 |
| 7 | TPDR (Transponder) SSMA connector for Veronte 3 |
| 8 | LOS SSMA connector for Veronte 2 |
| 9 | GNSS1 SSMA connector for Veronte 2 |
| 10 | M2M SSMA connector for Veronte 2 |
| 11 | GNSS2 SSMA connector for Veronte 2 |
| 12 | TPDR (Transponder) SSMA connector for Veronte 2 |
| 13 | LOS SSMA connector for Veronte 1 |
| 14 | GNSS1 SSMA connector for Veronte 1 |
| 15 | M2M SSMA connector for Veronte 1 |
| 16 | GNSS2 SSMA connector for Veronte 1 |
| 17 | TPDR (Transponder) SSMA connector for Veronte 1 |
| 18 | Dynamic pressure port (Fitting 5/64in) for Veronte 3 |
| 19 | Static pressure port (Fitting 5/64in) for Veronte 3 |
| 20 | Dynamic pressure port (Fitting 5/64in) for Veronte 2 |
| 21 | Static pressure port (Fitting 5/64in) for Veronte 2 |
| 22 | Dynamic pressure port (Fitting 5/64in) for Veronte 1 |
| 23 | Static pressure port (Fitting 5/64in) for Veronte 1 |
| 24 | Static pressure port (Fitting 5/64in) for all Verontes |

For pressure ports, mating with clamped 2mm internal diameter flexible tubing is recommended.

The static pressure port for all Verontes sets the 4xVeronte internal cage pressure.

3.2.1.6 Mating Connectors

| Index | Connector | Mating Connector |
|--------------|---|---|
| 3,8,13 | RF antenna/(SSMA Jack Female) | SSMA male Plug , low-loss cable is recommended. |
| 4,9,14,16,22 | GNSS antenna/(SSMA Jack Female) | SSMA male Plug, low-loss cable is recommended./Active Antenna GNSS: Gain min 15dB (to compensate signal loss in RF Cable) max 50dB, maximum noise figure 1.5dB, power supply 3.3V max current 20 mA |
| 10,12,14 | M2M antenna/(SSMA Jack Female) | SSMA male Plug, low-loss cable is recommended. |
| 1 | Redundant Connector/Connector HEW.LM.368.XLNP | Mating connector P/N: FGW.LM.368.XLCT/Mating harness is available on demand. |
| 2 | Arbiter Connector/Connector HER.LM.368.XLNP | Mating connector P/N: FGR.LM.368.XLCT/Mating harness is available on demand. |
| 7,12,17 | TPDR antenna (SSMA Jack Female) | SSMA male Plug, low-loss cable is recommended. |

3.2.1.7 Antenna Integration

The system uses different kinds of antennas to operate that must be installed on the airframe. Here you can find some advice for obtaining the best performance and for avoiding antenna interferences.

| Antenna Installation |
|---|
| Maximize separation between antennas as much as possible. |
| Keep it far away from alternators or other interference generators. |
| Always isolate antenna ground panel from the aircraft structure. |
| Make that the antenna is securely mounted. |
| Always use high-quality RF wires minimising the wire length. |
| Always follow the antenna manufacturer manual. |
| SSMA connections shall be tightened applying 1Nm of torque |
| For all-weather aircraft, insert SSMA lightning protectors. |

| GNSS Antenna |
|---|
| Antenna top side must point the sky. |
| Install it on a top surface with direct sky view. |
| Never place metallic / carbon parts or wires above the antenna. |
| It is recommended to install it on top of a ground plane. |
| For all-weather aircraft, insert SSMA lightning protectors. |

3.2.1.8 Pressure lines

4xVeronte has 6 pressure input lines, 3 for static pressure to determine the absolute pressure and 3 for pitot in order to determine the dynamic pressure on each internal autopilot.

Absolute pressure connection on the aircraft is mandatory while pitot port can be obviated in some aircrafts. Pitot port absence must be configured on Veronte Pipe software.

| Pressure Intake |
|--|
| Pressure intakes must be located in order to prevent clogging. |
| Never install pressure intakes on the propeller flow. |
| Design pressure tubing path in order to avoid tube constriction. |

| Static Pressure |
|---|
| It is not recommended to use inside fuselage pressure if it is not properly vented. |

| Pitot Tube |
|--|
| It is recommended to install it near the aircraft axis in order to avoid false measures during manoeuvres. |
| For low-speed aircraft it is recommended at least 6,3mm tubes for preventing rain obstruction. |
| Pitot tube must be installed facing the airflow in the direction of the “x” axis of the aircraft. |

3.2.2 Electrical

3.2.2.1 Power

4xVeronte can use unregulated DC (6.5V to 36V) for the internal Veronte autopilots and also for the arbiter.

LiPo batteries between 2S and 8S can be used without voltage regulation. Remaining battery can be controlled by the internal voltage sensor and by configuring the voltage warnings on the PC application.

For higher voltage installations, voltage regulators must be used. For dimensioning voltage regulators take into account that a blocked servo can activate regulator thermal protection.

Warning: Caution!! Power Veronte out of the given range can cause irreversible damage to the system. Please read carefully the manual before powering the system.

Veronte and servos can be powered by the same or different batteries. In case there are more than one battery on the system, a single point ground union it is needed to ensure a good performance. The ground signal should be isolated from other system ground references (e.g. engines).

It is recommendable to use independent switches for autopilot and motor / actuators. During the system initialization, PWM signal will be fixed to low level (0V), please make sure that actuators / motor connected support this behaviour before installing a single switch for the whole system.

Despite the names, all GND connectors share the same line, meaning they can be used for any ground connection required.

Harness ended in Blue matting connector will refer to Redundant connector, and the one ended in Yellow (reverse polarity) will refer to Arbiter connector.

3.2.2.2 Redundant Connector Pinout

| PIN | SIGNAL | TYPE | INTERNAL POWER DOMAIN | COMMENTS |
|-----|--------|------|-----------------------|--|
| 1 | I/O1 | I/O | A | MUXED PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V) |
| 2 | I/O2 | I/O | B | MUXED PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V) |
| 3 | I/O3 | I/O | A | MUXED PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V) |

continues on next page

Table 3 – continued from previous page

| PIN | SIGNAL | TYPE | INTERNAL POWER DOMAIN | COMMENTS |
|-----|--------|--------|--------------------------|--|
| 4 | I/O4 | I/O | B | MUXED PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V) |
| 5 | I/O5 | I/O | A | MUXED PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V) |
| 6 | I/O6 | I/O | B | MUXED PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V) |
| 7 | I/O7 | I/O | A | MUXED PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V) |
| 8 | I/O8 | I/O | B | MUXED PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V) |
| 9 | GND | GROUND | • | GROUND SIGNAL FOR ACTUATORS 1-8 |
| 10 | I/O9 | I/O | A | MUXED PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V) |
| 11 | I/O10 | I/O | B | MUXED PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V) |
| 12 | I/O11 | I/O | A | MUXED PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V) |
| 13 | I/O12 | I/O | B | MUXED PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V) |

continues on next page

Table 3 – continued from previous page

| PIN | SIGNAL | TYPE | INTERNAL POWER DOMAIN | COMMENTS |
|-----|-----------|--------|-----------------------|--|
| 14 | I/O13 | I/O | A | MUXED PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V) |
| 15 | I/O14 | I/O | B | MUXED PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V) |
| 16 | I/O15 | I/O | A | MUXED PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V) |
| 17 | I/O16 | I/O | B | MUXED PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V) |
| 18 | GND | GROUND | • | GROUND SIGNAL FOR ACTUATORS 9-16 |
| 19 | RS_232_TX | OUTPUT | A | MUXED RS-232 OUTPUT |
| 20 | RS_232_RX | INPUT | A | REDUNDANT RS-232 INPUT |
| 21 | V2_USB_DP | I/O | • | VERONTE 2 USB DATA LINE |
| 22 | ANALOG_4 | INPUT | B | REDUNDANT ANALOG INPUT 0-36V |
| 23 | ANALOG_5 | INPUT | B | REDUNDANT ANALOG INPUT 0-36V |
| 24 | V2_USB_DN | I/O | • | VERONTE 2 USB DATA LINE |
| 25 | CANA_P | I/O | • | CANbus interface. It supports data rates up to 1 Mbps. Twisted pair with a 120 Z_0 recommended |
| 26 | CANA_N | I/O | • | |
| 27 | GND | GROUND | • | GROUND SIGNAL FOR BUSES |

continues on next page

Table 3 – continued from previous page

| PIN | SIGNAL | TYPE | INTERNAL POWER DOMAIN | COMMENTS |
|-----|-----------|----------|-----------------------|--|
| 28 | CANB_P | I/O | • | CANbus interface. It supports data rates up to 1 Mbps. Twisted pair with a 120 Z_0 recommended |
| 29 | CANB_N | I/O | • | |
| 30 | V2_USB_ID | I/O | • | VERONTE 2 USB ID LINE |
| 31 | I2C_CLK | OUTPUT A | • | MUXED CLK LINE FOR I2C BUS |
| 32 | I2C_DATA | I/O | A | MUXED DATA LINE FOR I2C BUS |
| 33 | GND | GROUND | • | GROUND FOR 3.3V POWER SUPPLY |
| 34 | 3.3V | POWER | B | 3.3V-100mA POWER SUPPLY |
| 35 | GND | GROUND | • | GROUND FOR 5V POWER SUPPLY |
| 36 | 5V | POWER | B | 5V-100mA POWER SUPPLY |
| 37 | GND | GROUND | • | GROUND FOR ANALOG SIGNALS |
| 38 | ANALOG_1 | INPUT | A | REDUNDANT ANALOG INPUT 0-36V |
| 39 | ANALOG_2 | INPUT | A | REDUNDANT ANALOG INPUT 0-36V |
| 40 | ANALOG_3 | INPUT | A | REDUNDANT ANALOG INPUT 0-36V |
| 41 | GND | GROUND | • | GROUND SIGNAL FOR BUSES |
| 42 | V3_USB_DP | I/O | • | VERONTE 3 USB DATA LINE |
| 43 | V3_USB_DN | I/O | • | VERONTE 3 USB DATA LINE |
| 44 | GND | GROUND | • | GROUND SIGNAL FOR BUSES |

continues on next page

Table 3 – continued from previous page

| PIN | SIGNAL | TYPE | INTERNAL POWER DOMAIN | COMMENTS |
|-----|-------------|--------|-----------------------------------|---|
| 45 | UART_TX | OUTPUT | B | MUXED UART OUTPUT |
| 46 | UART_RX | INPUT | B | REDUNDANT UART INPUT |
| 47 | GND | GROUND | • | GROUND SIGNAL FOR BUSES |
| 48 | GND | GROUND | • | GROUND SIGNAL FOR BUSES |
| 49 | V3_USB_ID | I/O | • | VERONTE 3 USB ID LINE |
| 50 | OUT_RS485_P | OUTPUT | B | NON-INVERTED OUTPUT FOR MUXED RS-485 BUS |
| 51 | OUT_RS485_N | OUTPUT | B | INVERTED OUTPUT FOR MUXED RS-485 BUS |
| 52 | IN_RS485_N | INPUT | • | INVERTED INPUT FOR MUXED RS-485 BUS |
| 53 | IN_RS485_P | INPUT | • | NON-INVERTED INPUT FOR MUXED RS-485 BUS |
| 54 | RS-485_GND | GROUND | • | GROUND FOR RS-485 BUS |
| 55 | EQEP_A | INPUT | A FOR VERONTE 1&2 B FOR VERONTE 3 | ENCODER QUADRATURE REDUNDANT INPUT A (0-5V) |
| 56 | EQEP_B | INPUT | | ENCODER QUADRATURE REDUNDANT INPUT B (0-5V) |
| 57 | EQEP_S | INPUT | | ENCODER STROBE REDUNDANT INPUT (0-5V) |
| 58 | EQEP_I | INPUT | | ENCODER INDEX REDUNDANT INPUT (0-5V) |
| 59 | GND3 | GROUND | • | VERONTE 3 GROUND INPUT |

continues on next page

Table 3 – continued from previous page

| PIN | SIGNAL | TYPE | INTERNAL POWER DOMAIN | COMMENTS |
|-----|-----------|--------|-----------------------|-------------------------------------|
| 60 | V1_USB_DP | I/O | • | VERONTE 1 USB DATA LINE |
| 61 | V1_USB_DN | I/O | • | VERONTE 1 USB DATA LINE |
| 62 | V1_USB_ID | I/O | • | VERONTE 1 USB ID LINE |
| 63 | GND | GROUND | • | GROUND SIGNAL FOR BUSES |
| 64 | VCC3 | POWER | • | VERONTE 3 POWER SUPPLY (6.5 to 36V) |
| 65 | GND2 | GROUND | • | VERONTE 2 GROUND INPUT |
| 66 | GND1 | GROUND | • | VERONTE 1 GROUND INPUT |
| 67 | VCC2 | POWER | • | VERONTE 2 POWER SUPPLY (6.5 to 36V) |
| 68 | VCC1 | POWER | • | VERONTE 1 POWER SUPPLY (6.5 to 36V) |

3.2.2.3 Arbiter Connector Pinout

Although being the same part, Arbiter connector and the Redundant connector are polarized differently to avoid wiring swapping.

| PIN | SIGNAL | TYPE | INTERNAL POWER DOMAIN | COMMENTS |
|-----|-------------------|------|-----------------------|---|
| 1 | EXTERNAL FCU I/O1 | I/O | A | EXTERNAL FCU PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V). Input current must be limited to 25mA. |

continues on next page

Table 4 – continued from previous page

| PIN | SIGNAL | TYPE | INTERNAL POWER DOMAIN | COMMENTS |
|-----|-------------------|------|-----------------------|---|
| 2 | EXTERNAL FCU I/O2 | I/O | B | EXTERNAL FCU PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V). Input current must be limited to 25mA. |
| 3 | EXTERNAL FCU I/O3 | I/O | A | EXTERNAL FCU PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V). Input current must be limited to 25mA. |
| 4 | EXTERNAL FCU I/O4 | I/O | B | EXTERNAL FCU PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V). Input current must be limited to 25mA. |
| 5 | EXTERNAL FCU I/O5 | I/O | A | EXTERNAL FCU PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V). Input current must be limited to 25mA. |
| 6 | EXTERNAL FCU I/O6 | I/O | B | EXTERNAL FCU PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V). Input current must be limited to 25mA. |
| 7 | EXTERNAL FCU I/O7 | I/O | A | EXTERNAL FCU PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V). Input current must be limited to 25mA. |
| 8 | EXTERNAL FCU I/O8 | I/O | B | EXTERNAL FCU PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V). Input current must be limited to 25mA. |

continues on next page

Table 4 – continued from previous page

| PIN | SIGNAL | TYPE | INTERNAL POWER DOMAIN | COMMENTS |
|-----|-----------------------------|--------|-----------------------|---|
| 9 | EXTERNAL FCU I/O9 | I/O | A | EXTERNAL FCU PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V). Input current must be limited to 25mA. |
| 10 | EXTERNAL FCU I/O10 | I/O | B | EXTERNAL FCU PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V). Input current must be limited to 25mA. |
| 11 | EXTERNAL FCU I/O11 | I/O | A | EXTERNAL FCU PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V). Input current must be limited to 25mA. |
| 12 | EXTERNAL FCU I/O12 | I/O | B | EXTERNAL FCU PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V). Input current must be limited to 25mA. |
| 13 | VCC2 | POWER | • | VERONTE 2 POWER SUPPLY (6.5 to 36V) |
| 14 | EXTERNAL FCU ANALOG INPUT 1 | OUTPUT | A | EXTERNAL FCU ANALOG INPUT (0-36V). This is the analog signal corresponding to Analog signal 1 on Redundant connector. |
| 15 | EXTERNAL FCU ANALOG INPUT 2 | OUTPUT | A | EXTERNAL FCU ANALOG INPUT (0-36V). This is the analog signal corresponding to Analog signal 2 on Redundant connector. |

continues on next page

Table 4 – continued from previous page

| PIN | SIGNAL | TYPE | INTERNAL POWER DOMAIN | COMMENTS |
|-----|-------------------------------------|--------|-----------------------|---|
| 16 | EXTERNAL FCU ANALOG INPUT 3 | OUTPUT | A | EXTERNAL FCU ANALOG INPUT (0-36V). This is the analog signal corresponding to Analog signal 3 on Redundant connector. |
| 17 | EXTERNAL FCU ANALOG INPUT 4 | OUTPUT | B | EXTERNAL FCU ANALOG INPUT (0-36V). This is the analog signal corresponding to Analog signal 4 on Redundant connector. |
| 18 | FLIGHT TERMINATION SIGNAL STAGE B | OUTPUT | B | OPEN DRAIN OUTPUT FROM VOTING STAGE B (Sensed by arbiter) |
| 19 | EXTERNAL FCU TO PAYLOAD UART SIGNAL | INPUT | A | EXTERNAL FCU UART OUTPUT. (0-3.3V) THIS SIGNAL WILL BE AN INPUT FOR THE RS_232 OUTPUT MULTIPLEXER. MULTIPLEXED OUTPUT ON REDUNDANT CONNECTOR PIN 19 |
| 20 | PAYLOAD TO EXTERNAL FCU UART SIGNAL | OUTPUT | A | EXTERNAL FCU UART INPUT. (0-3.3V) THIS SIGNAL WILL BE THE OUTPUT OF THE RS_232 MULTIPLEXER. MULTIPLEXED INPUT ON REDUNDANT CONNECTOR PIN 20 |

continues on next page

Table 4 – continued from previous page

| PIN | SIGNAL | TYPE | INTERNAL POWER DOMAIN | COMMENTS |
|-----|----------------------------------|--------|-----------------------|---|
| 21 | PAYLOAD TO EXTERNAL FCU RS-485_P | OUTPUT | • | RS-485_P OUTPUT SIGNAL FROM PAYLOAD (REDUNDANT CONNECTOR PIN 53) TO FCU RS-485_P INPUT |
| 22 | PAYLOAD TO EXTERNAL FCU RS-485_N | OUTPUT | • | RS-485_N OUTPUT SIGNAL FROM PAYLOAD (REDUNDANT CONNECTOR PIN 52) TO FCU RS-485_N INPUT |
| 23 | EXTERNAL FCU TO PAYLOAD RS-485_P | INPUT | B | RS-485_P OUTPUT SIGNAL FROM EXTERNAL FCU (REDUNDANT CONNECTOR PIN 50) TO PAYLOAD RS-485_P INPUT |
| 24 | EXTERNAL FCU TO PAYLOAD RS-485_N | INPUT | B | RS-485_N OUTPUT SIGNAL FROM EXTERNAL FCU (REDUNDANT CONNECTOR PIN 51) TO PAYLOAD RS-485_N INPUT |
| 25 | CANA_P | I/O | • | CANbus interface. It supports data rates up to 1 Mbps. Recommended cable is a twisted pair with a 120 Zo. |
| 26 | CANA_N | I/O | • | |
| 27 | VCC1 | POWER | • | VERONTE 1 POWER SUPPLY (6.5 to 36V) |
| 28 | CANB_P | I/O | • | CANbus interface. It supports data rates up to 1 Mbps. Recommended cable is a twisted pair with a 120 Zo. |
| 29 | CANB_N | I/O | • | |
| 30 | ARBITER_RS485_OUT | OUTPUT | ARBITER | NON-INVERTED OUTPUT FOR ARBITER'S RS-485 BUS |

continues on next page

Table 4 – continued from previous page

| PIN | SIGNAL | TYPE | INTERNAL POWER DOMAIN | COMMENTS |
|-----|------------------------|--------|-----------------------|---|
| 31 | ARBITER_RS485_OUT | OUTPUT | ARBITER | INVERTED OUTPUT FOR ARBITER'S RS-485 BUS |
| 32 | ARBITER_RS485_IN | INPUT | ARBITER | INVERTED INPUT FOR ARBITER'S RS-485 BUS |
| 33 | ARBITER_RS485_IN | INPUT | ARBITER | NON-INVERTED INPUT FOR ARBITER'S RS-485 BUS |
| 34 | ARBITER_ARINC_TX | OUTPUT | ARBITER | ARBITER'S ARINC POSITIVE OUTPUT |
| 35 | ARBITER_ARINC_TX | OUTPUT | ARBITER | ARBITER'S ARINC NEGATIVE OUTPUT |
| 36 | ARBITER_ARINC_RX | INPUT | ARBITER | ARBITER'S ARINC POSITIVE INPUT |
| 37 | ARBITER_ARINC_RX | INPUT | ARBITER | ARBITER'S ARINC NEGATIVE INPUT |
| 38 | GND | GROUND | • | GROUND SIGNAL FOR BUSES |
| 39 | ARBITER_I2C_SCL | OUTPUT | ARBITER | CLK LINE FOR ARBITER'S I2C BUS |
| 40 | ARBITER_I2C_DATA | I/O | ARBITER | DATA LINE FOR ARBITER'S I2C BUS |
| 41 | ARBITER_RS232B_RX | INPUT | ARBITER | ARBITER RS-232 INPUT B |
| 42 | ARBITER_RS232B_TX | OUTPUT | ARBITER | ARBITER RS-232 OUTPUT B |
| 43 | ARBITER_RS232A_RX | INPUT | ARBITER | ARBITER RS-232 INPUT A |
| 44 | ARBITER_RS232A_TX | OUTPUT | ARBITER | ARBITER RS-232 OUTPUT A |
| 45 | GND | GROUND | • | GROUND SIGNAL FOR ANALOG SIGNALS |
| 46 | ARBITER_ANALOG_INPUT_1 | I/O | ARBITER | ARBITER ANALOG INPUT (0-36V) |

continues on next page

Table 4 – continued from previous page

| PIN | SIGNAL | TYPE | INTERNAL POWER DOMAIN | COMMENTS |
|-----|-----------------------------------|--------|-----------------------|--|
| 47 | ARBITER ANALOG_INPUT_2 | I/O | ARBITER | ARBITER ANALOG INPUT (0-36V) |
| 48 | ARBITER ANALOG_INPUT_3 | I/O | ARBITER | ARBITER ANALOG INPUT (0-36V) |
| 49 | ARBITER ANALOG_INPUT_4 | I/O | ARBITER | ARBITER ANALOG INPUT (0-36V) |
| 50 | ARBITER ANALOG_INPUT_5 | I/O | ARBITER | ARBITER ANALOG INPUT (0-36V) |
| 51 | ARBITER ANALOG_INPUT_6 | I/O | ARBITER | ARBITER ANALOG INPUT (0-36V) |
| 52 | ARBITER ANALOG_INPUT_7 | I/O | ARBITER | ARBITER ANALOG INPUT (0-36V) |
| 53 | FLIGHT TERMINATION SIGNAL STAGE A | OUTPUT | A | OPEN DRAIN OUTPUT FROM VOTING STAGE A (Sensed by arbiter) |
| 54 | ARB_GPIO9 | I/O | ARBITER | ARBITER'S PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V) |
| 55 | ARB_GPIO10 | I/O | ARBITER | ARBITER'S PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V) |
| 56 | WD_EXT | I | ARBITER | WATCHDOG SIGNAL FROM EXTERNAL AUTOPILOT TO ARBITER (0-3.3V) |
| 57 | EXT_DETECT | I | ARBITER | CONNECT TO GND IF EXTERNAL FCU IS CONNECTED. OTHERWISE, LEAVE OPEN |
| 58 | GND | GROUND | • | GROUND SIGNAL FOR GPIO |

continues on next page

Table 4 – continued from previous page

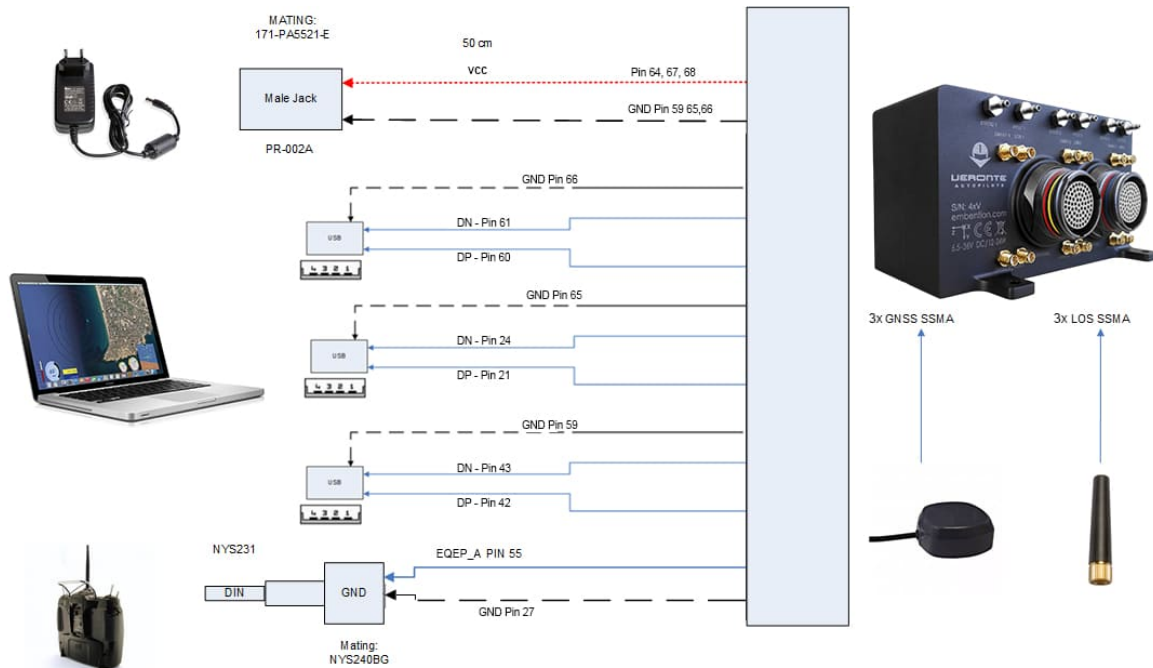
| PIN | SIGNAL | TYPE | INTERNAL POWER DOMAIN | COMMENTS |
|-----|-------------|--------|-----------------------|--|
| 59 | ARB_GPIO1 | I/O | ARBITER | ARBITER'S PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V) |
| 60 | ARB_GPIO2 | I/O | ARBITER | ARBITER'S PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V) |
| 61 | ARB_GPIO3 | I/O | ARBITER | ARBITER'S PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V) |
| 62 | ARB_GPIO4 | I/O | ARBITER | ARBITER'S PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V) |
| 63 | ARB_GPIO5 | I/O | ARBITER | ARBITER'S PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V) |
| 64 | ARB_GPIO6 | I/O | ARBITER | ARBITER'S PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V) |
| 65 | ARB_GPIO7 | I/O | ARBITER | ARBITER'S PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V) |
| 66 | ARB_GPIO8 | I/O | ARBITER | ARBITER'S PWM/DIGITAL OUTPUT/DIGITAL INPUT SIGNAL (0-3.3V) |
| 67 | GND_ARBITER | GROUND | • | ARBITER GROUND INPUT |
| 68 | VCC_ARBITER | POWER | • | ARBITER POWER SUPPLY (6.5 to 36V) |

3.2.3 Performances

| Variable | Value |
|----------------------------------|---|
| Weight (radio included) | 750 g |
| Power Input | 6.5 V to 36 V |
| Minimum Temperature | -40 °C |
| Maximum Temperature | +55°C (No convection, ask for increased limits (up to 71°C)) |
| Max. Internal Temperature | +85°C |
| Minimum Pressure | 0 kPa |
| Maximum Pressure | 104 kPa |
| Maximum Dynamic pressure | 6 kPa (Ask for increased limits (up to 50kPa)) |
| Protection Rating | IP67 enclosure version |
| Acceleration Limits (3 axes) | ± 2 g to ± 24 g (for sustained maneuvers, transitional higher accelerations are possible (e.g. catapult launch). Ask for increased limits.) |
| Angular Velocity Limits (3 axes) | ± 125 deg/s to ± 2000 deg/s (for sustained maneuvers, transitional higher angular velocities are possible. Ask for increased limits.) |
| Magnetic Field Limits (3 axes) | ± 4 to ± 16 Gauss |
| GNSS | 72 channels, GPS L1C/A, GLONASS L1OF, BeiDou B1I |
| Datalink | 410 to 480 MHz licensed or FHSS/902-928MHz FHSS/2.4 to 2.483 GHz ISM Band/869.5-869.75 MHz ISM Band |
| Special Datalinks on request | 920 – 925 MHz, Singapore regulation compliance/869.5-869.75 MHz ISM Band |

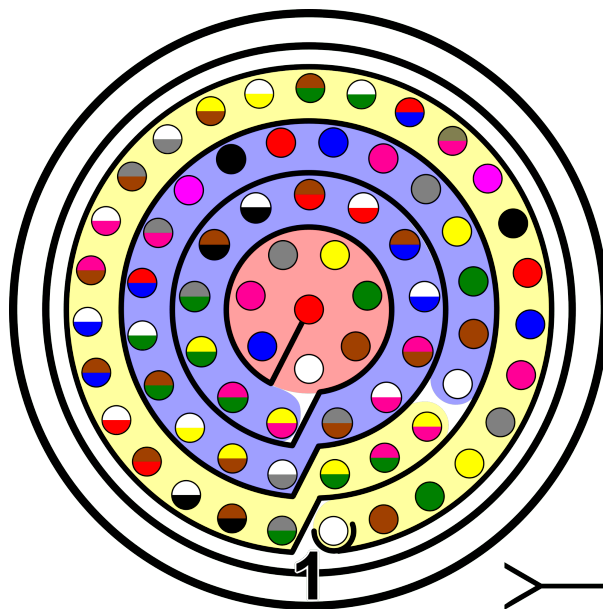
3.2.4 Annex 1: Connection Examples

3.2.4.1 Ground Station



Connection Example

3.2.5 Annex 2: Connector colour code

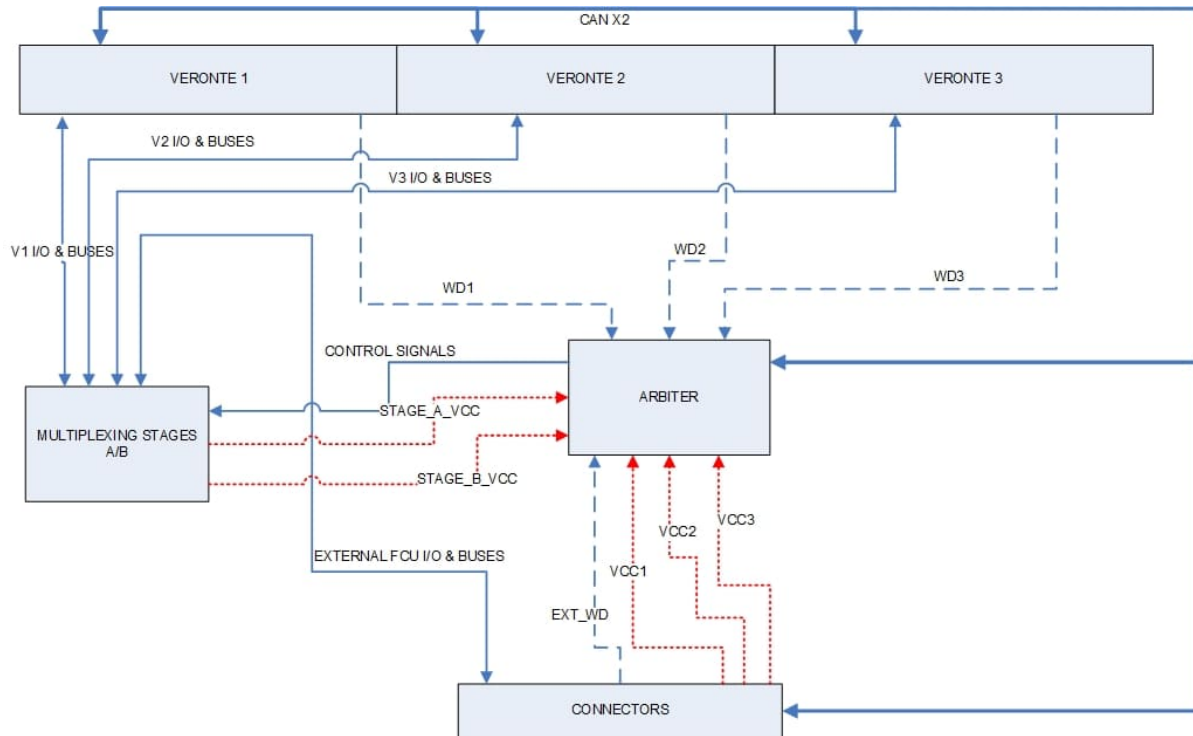


Connector HEW.LM.368.XLNP

| PIN | Color code | PIN | Color code |
|-----|----------------|-----|----------------|
| 1 | White | 35 | Gray |
| 2 | Brown | 36 | Pink |
| 3 | Green | 37 | Blue |
| 4 | Yellow | 38 | Red |
| 5 | Gray | 39 | Black |
| 6 | Pink | 40 | Violet |
| 7 | Blue | 41 | Gray – Pink |
| 8 | Red | 42 | Red – Blue |
| 9 | Black | 43 | White – Green |
| 10 | Violet | 44 | Brown – Green |
| 11 | Gray – Pink | 45 | White – Yellow |
| 12 | Red – Blue | 46 | Yellow – Brown |
| 13 | White – Green | 47 | White – Gray |
| 14 | Brown – Green | 48 | Gray – Brown |
| 15 | White – Yellow | 49 | White – Pink |
| 16 | Yellow – Brown | 50 | Pink – Brown |
| 17 | White – Gray | 51 | White – Blue |
| 18 | Gray – Brown | 52 | Brown – Blue |
| 19 | White – Pink | 53 | White – Red |
| 20 | Pink – Brown | 54 | Brown – Red |
| 21 | White – Blue | 55 | White – Black |
| 22 | Brown – Blue | 56 | Brown – Black |
| 23 | White – Red | 57 | Gray – Green |
| 24 | Brown – Red | 58 | Yellow – Green |
| 25 | White – Black | 59 | Pink – Green |
| 26 | Brown – Black | 60 | Yellow – Pink |
| 27 | Grey – Green | 61 | White |
| 28 | Yellow – Green | 62 | Brown |
| 29 | Pink – Green | 63 | Green |
| 30 | Yellow – Pink | 64 | Yellow |
| 31 | White | 65 | Grey |
| 32 | Brown | 66 | Pink |
| 33 | Green | 67 | Blue |
| 34 | Yellow | 68 | Red |

Warning: The colour code is repeated due to the amount of pins. Check the pin number before connecting. Pin number increases following the black line of the picture above.

4xVeronte Autopilot is a **triple redundant** version of the Veronte Autopilot. It includes three complete Veronte Autopilot modules together with a dissimilar arbiter for detecting system failures and selecting the module in charge of the control. The autopilot selected as the master will be the one controlling the actuators and communicating with the payloads, as seen in the following block diagram.



4xVeronte Overview

Each Veronte autopilot contains all the electronics and sensors in order to properly execute all the functions needed to control the UAV. Veronte executes in real time all the guidance, navigation and control algorithms for the carrying airframe, acting on the control surfaces and propulsion system and processing the signals from different sensors: accelerometers, gyroscopes, magnetometer, static pressure, dynamic pressure, GNSS and external sensors.

Additional I/O port is available for the connection of an **external control** system in case it is required and include it in the redundant scheme. It provides the system with full dissimilarity for high demanding environments as required by civil aviation authorities.

Datalink communications can be also redundant, being possible to install inside the autopilot 3 radios of different frequencies. For example, it allows you to have two radios working in the 900MHz frequency and one in the 2.4GHz, so in case there is any issue in the 900MHz band the module connected to the 2.4GHz band will take control. In addition, an external radio can be controlled as a critical device using the serial port in the redundant connector.

All three modules are managed by a dissimilar microprocessor. This arbiter includes voting algorithms for managing the module in charge of vehicle control. This microprocessor compares data from all modules in real time and processes it for discarding any autopilot module showing an undesired performance.

4xVeronte also includes two separate flight termination voting logics, completely dissimilar and implemented with simple hardware, with the purpose of giving the internal three Veronte Autopilots a way to decide by consensus if a flight termination signal should be activated or not.

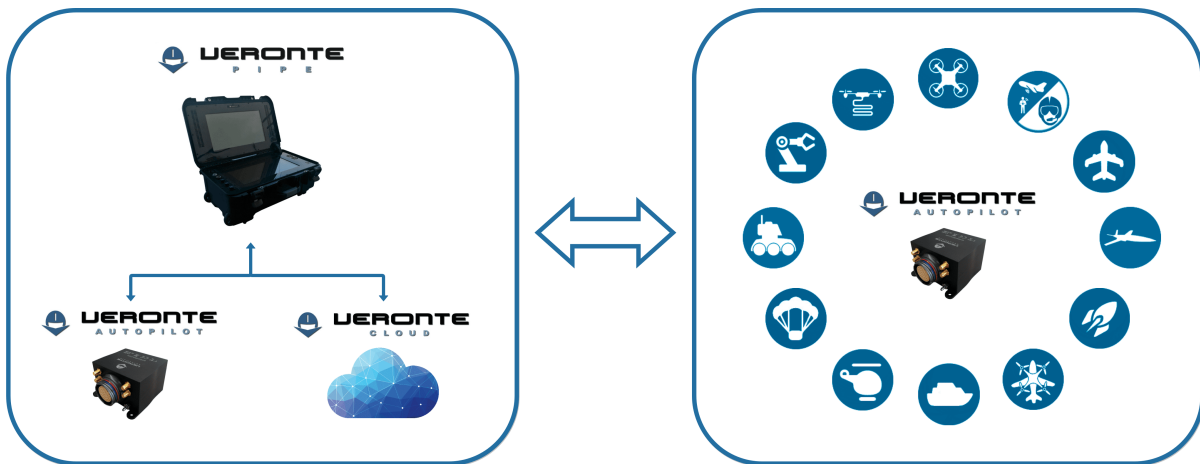
This redundant device allows platforms to perform **sensitive flight mission** and transport **valuable payload** with advanced safety conditions and high reliability. By installing a triple redundant core it is possible to extend the MTBF (Mean Time Between Failures) in the system. This control module is also suitable for both, fail-safe and fail-operational missions. Extending the operability of the system.



Veronte Autopilot Kit

This section provides the information about the Hardware of Veronte Autopilots. The content includes enclosure and mating connectors references, general informations about electrical connections and Veronte specifications.

Veronte is the main element in our Flight Control System for UAS. As shown in the following diagram, Veronte AP is used both in the ground segment (**GND**) and onboard (**AIR**).



System Overview

Veronte includes the required electronics and sensors in order to be able to properly execute all the needed functions for the UAS control. A Veronte-based FCS consists of the following components:

- **Veronte AIR** - It executes GNC algorithms in real time in order to accomplish the planned mission and handle the payload.
- **Veronte GND** - Link between Veronte Pipe and the AIR Units. It supports manual and arcade modes with conventional joysticks, it's capable to control a directional antenna in order to expand the maximum range and it

can be equipped with external physical interfaces in order to operate the system without Veronte Pipe.

- **Veronte Pipe** - Software dedicated to mission planning, configuration and operation. It allows the user to monitor the connected UAS in real time, to interact with them and to replay previous missions for post-flight analysis.

PIPE CONFIGURATION

4.1 Installation

4.1.1 System Requirements

Minimum Hardware Specification for installation:

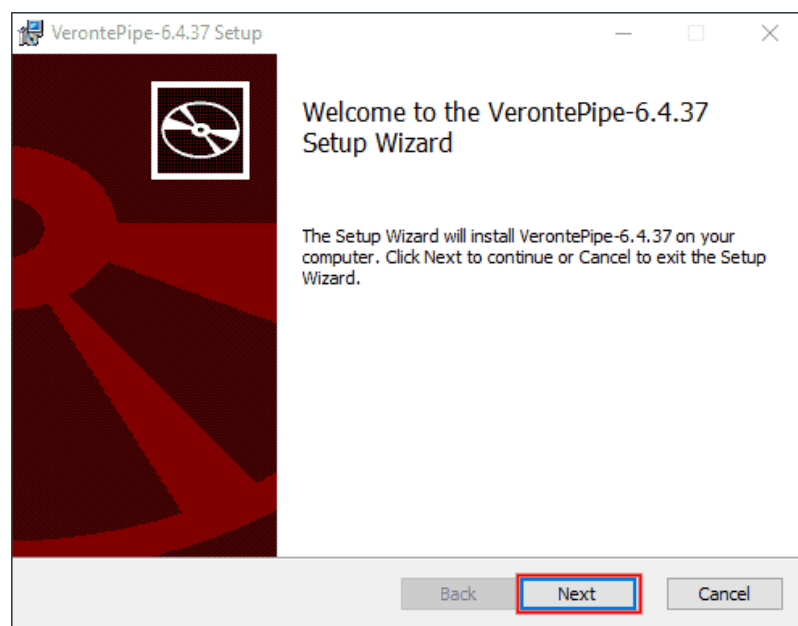
- **CPU:** Intel Core i5 6600K at 3.5 ghz or faster.
- **RAM:** 8 GB.
- **VIDEO CARD:** DirectX 11-capable video card.
- **FREE DISK SPACE:** 10 GB

Veronte supports the main Operating Systems (Windows, Linux and MacOS X). Contact Embention and we will provide you with the software that better fits your requirements. Also, you must have updated the [latest version of java](#).

4.1.2 Windows

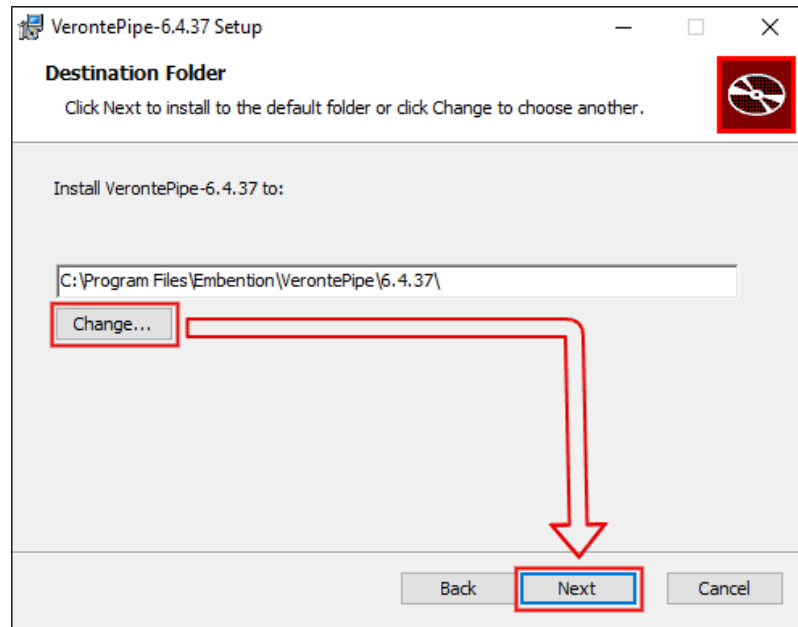
To install Veronte Pipe on Windows just execute “Veronte_Pipe.exe” and follow the indications.

1. Click on Next.



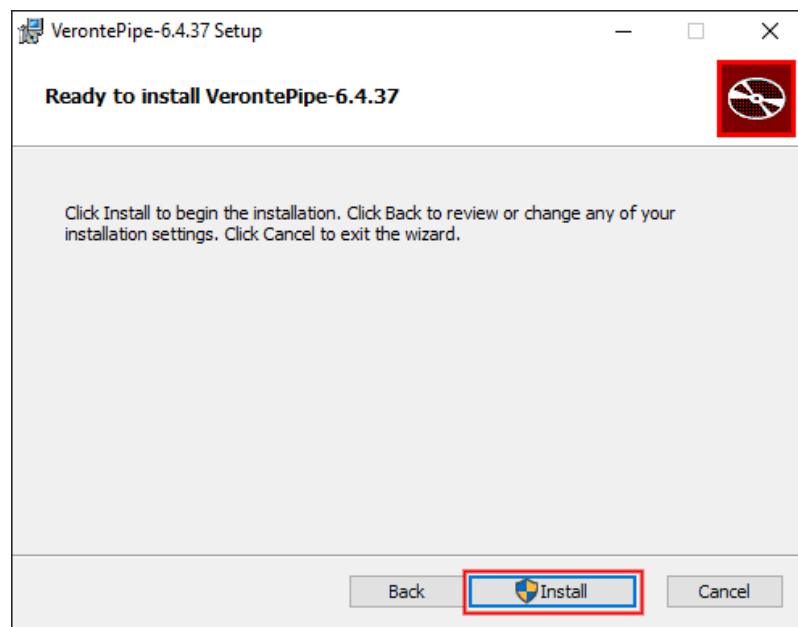
Windows Installation Step 01

2. Select the directory where you want to Install and click on next.



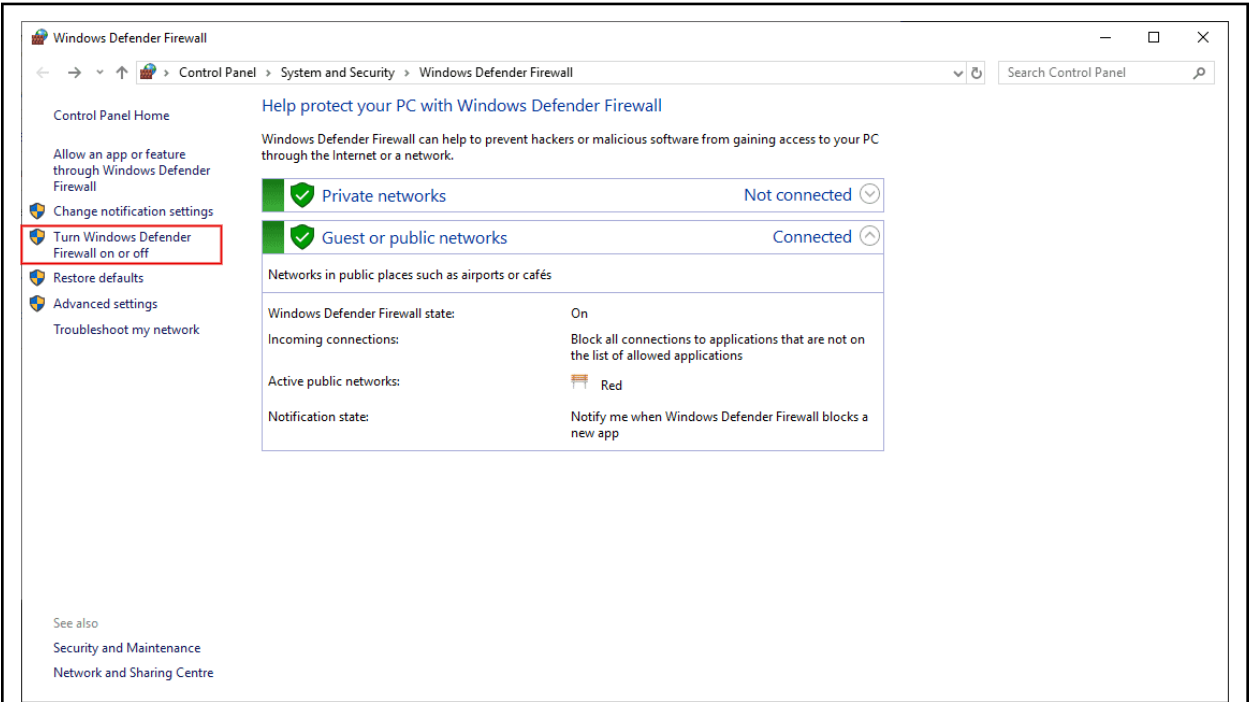
Windows Installation Step 02

3. Finally, click on Install (administrator rights are needed):

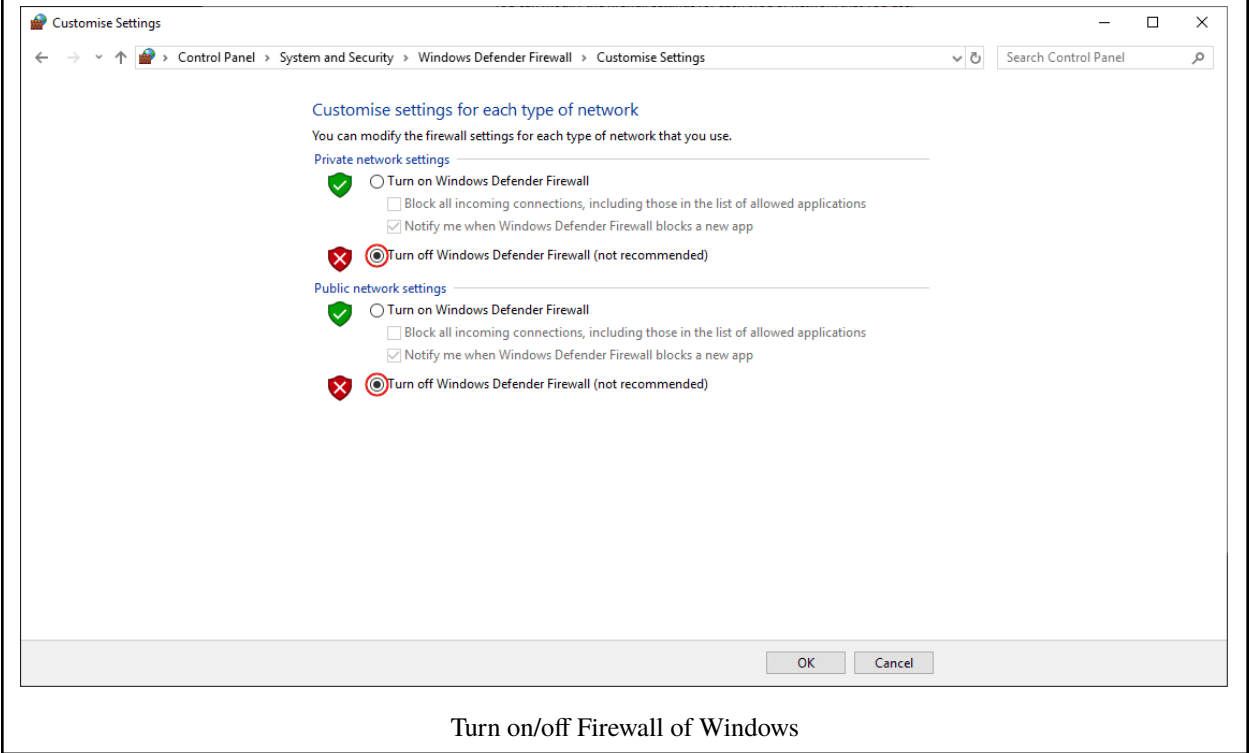


Windows Installation Step 03

Warning: If you have any problems with the installation, please disable antivirus and firewall of windows. Disabling the antivirus depends on your antivirus software. To disable the firewall, go to “Control Panel” and “Firewall of windows”, then click on Turn on and turn off windows firewall and finally click on.



Turn on/off Firewall of Windows

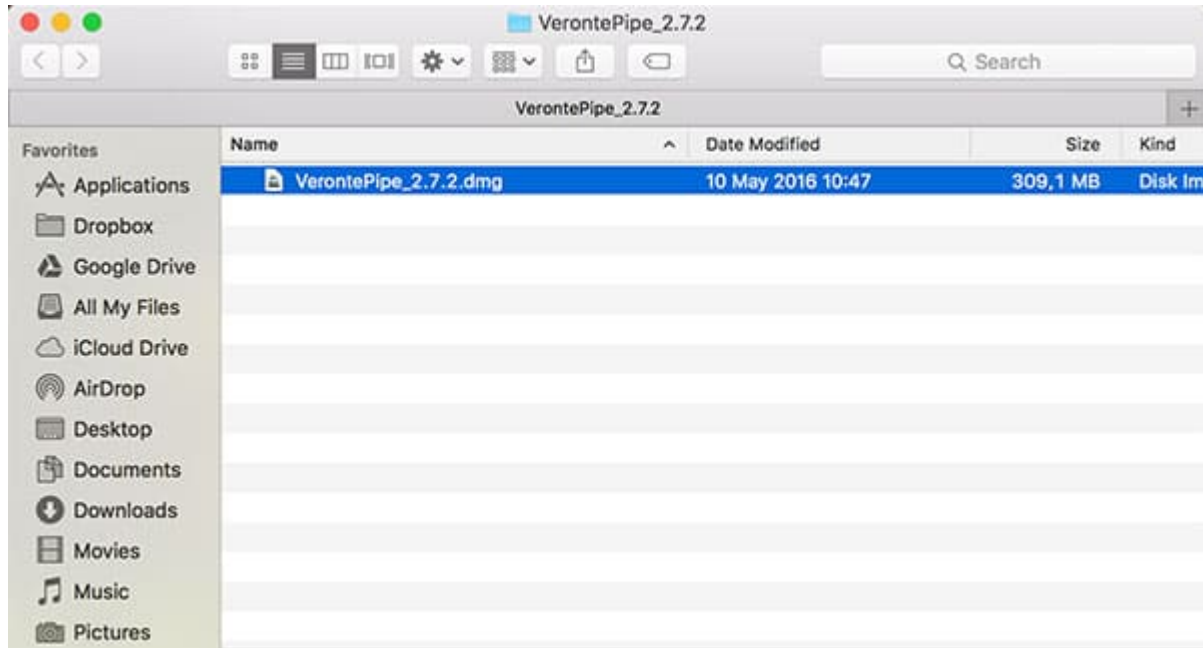


Turn on/off Firewall of Windows

4.1.3 MAC

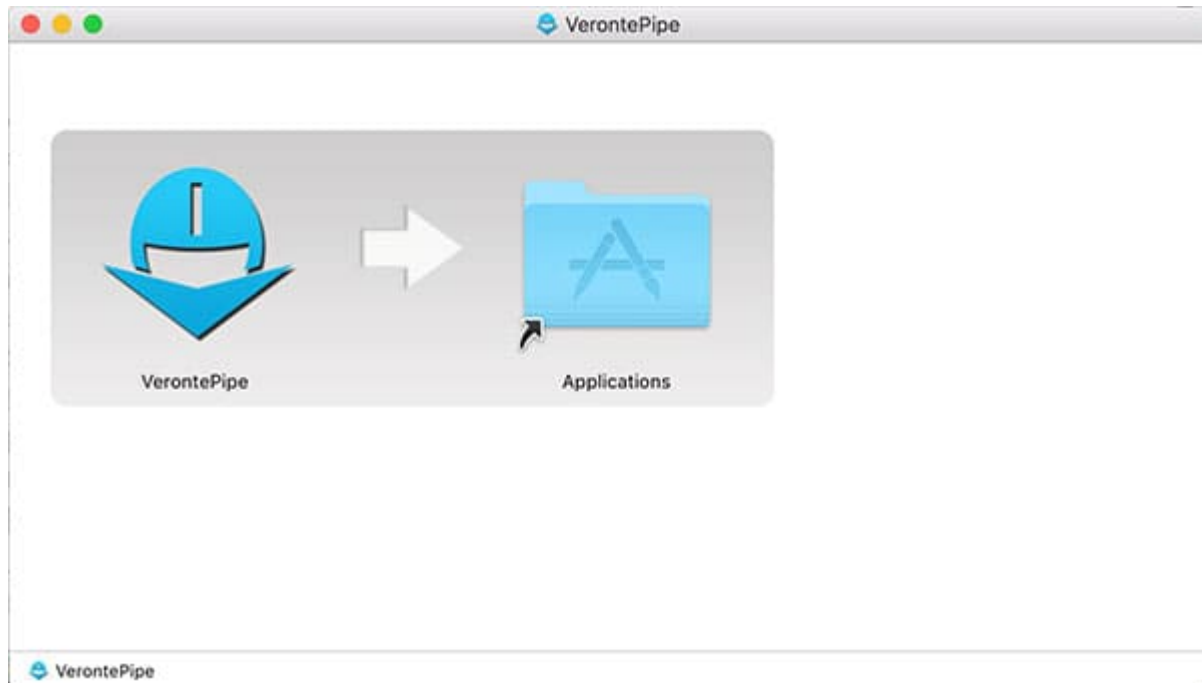
Attention: Pipe version for MAC is available on demand.

To install Veronte Pipe on Mac just double click on “VerontePipe_x.x.x.dmg” (where “x.x.x” is the version number), in order to mount the image.



Mac Installation Step 01

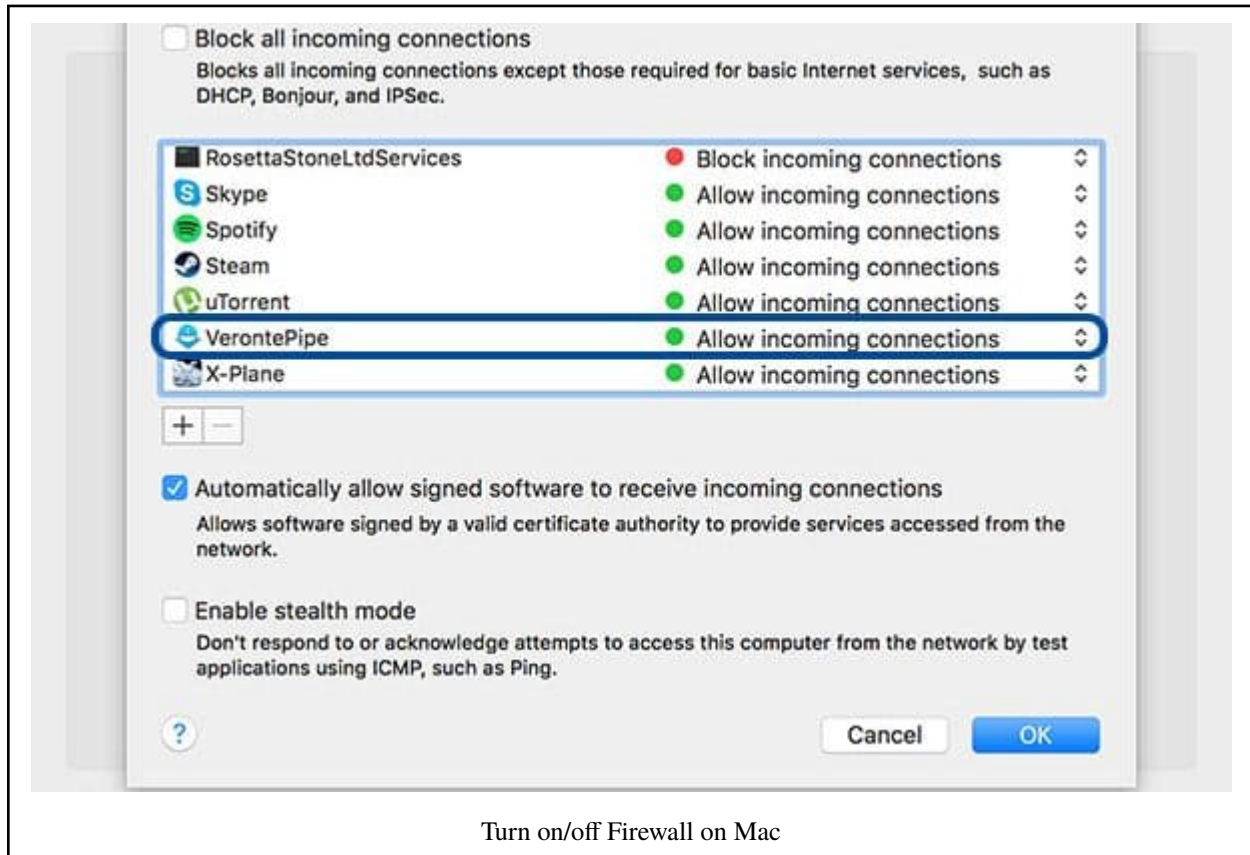
In the “VerontePipe” panel just follow the indications and move the Veronte Pipe application in the Applications folder to install it. The software is now ready to operate.



Mac Installation Step 02

Warning: If you have any problems during connection of VerontePipe, please check the firewall configuration of the application and disable it. To disable the firewall, go to “Security and Privacy” panel and be sure the program “allows incoming connections”.

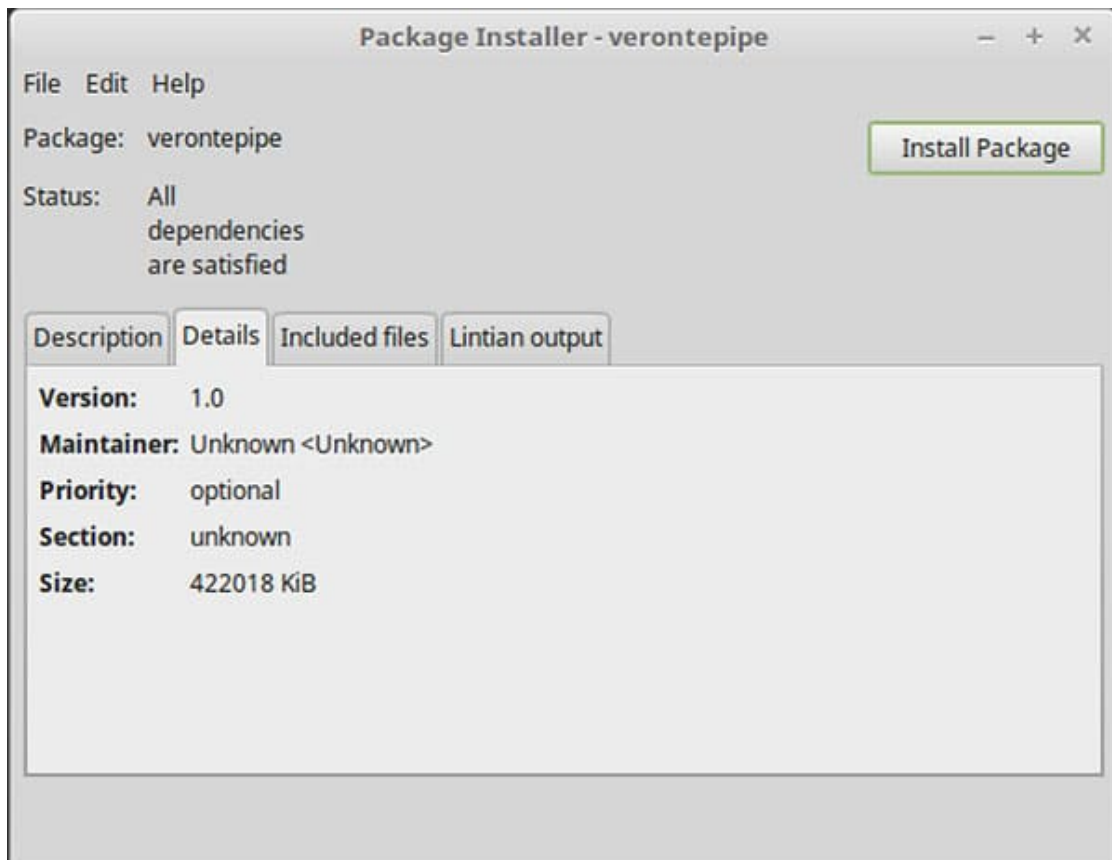
Deactivating the antivirus can be useful too. Disabling the antivirus depends on your antivirus software.



4.1.4 Linux

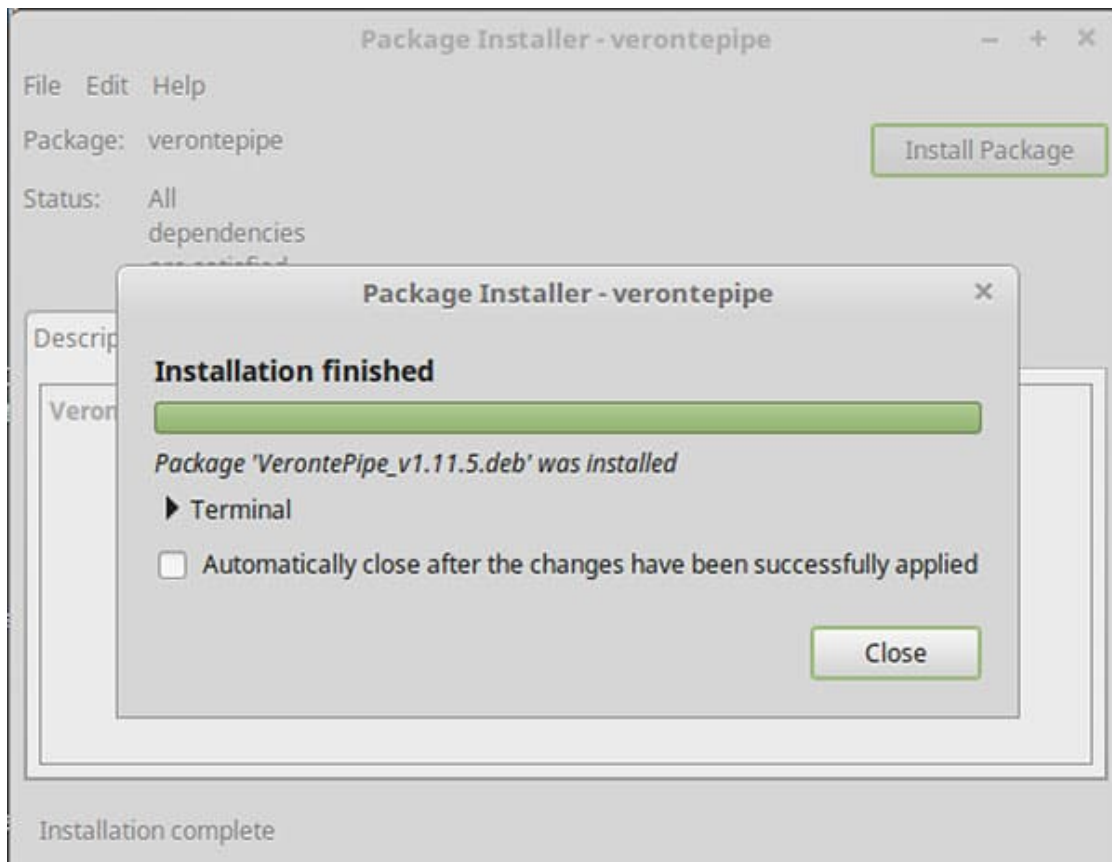
Attention: Pipe version for Linux is available on demand.

To install Veronte Pipe on Linux (Ubuntu in this case) just double click on “VerontePipe_x.x.x.deb” (where “x.x.x” is the version number), in order to open the Package Installer.



Linux install step 1

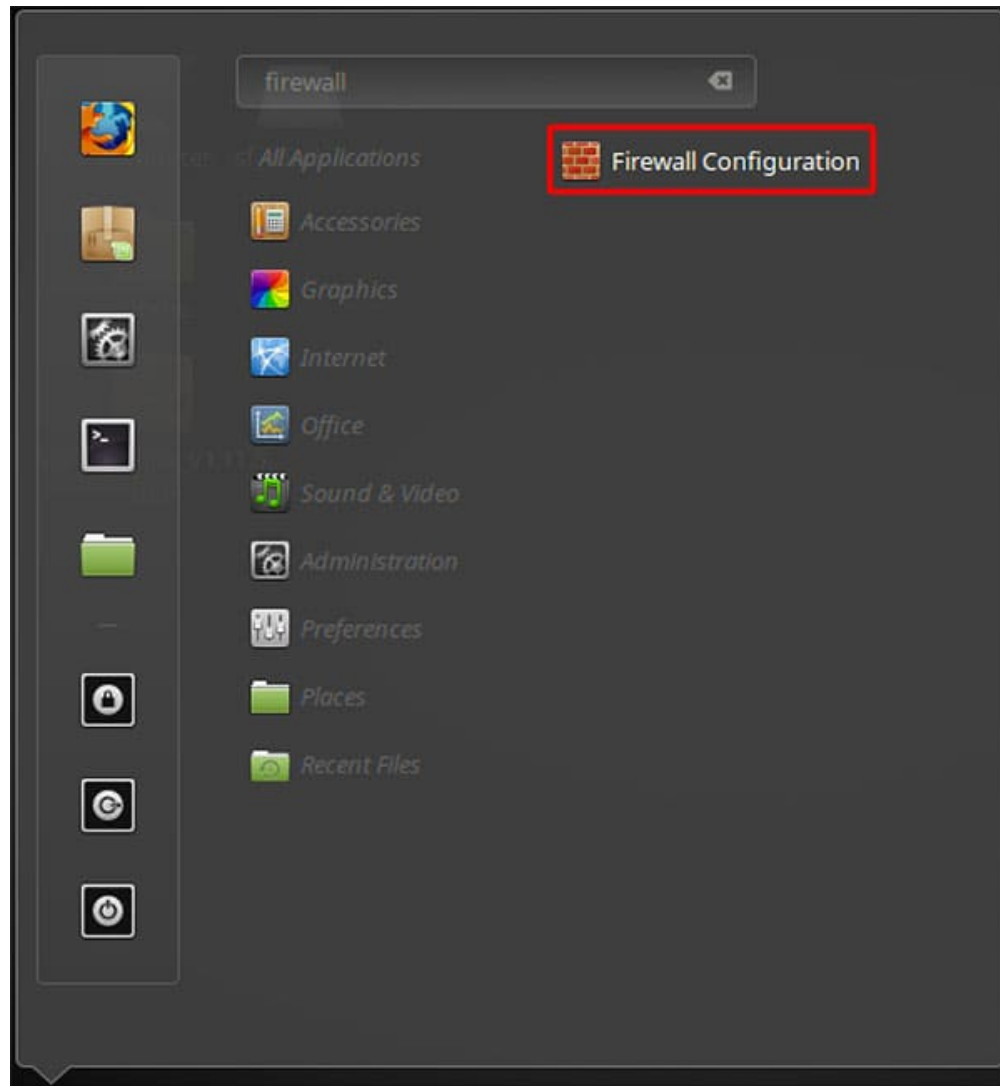
Then click on Install Package and wait the end of the installation process.



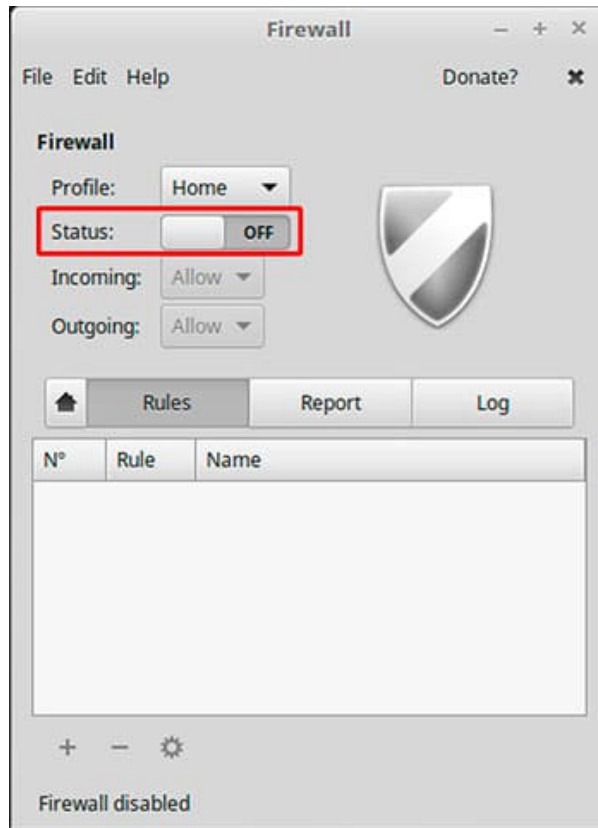
Linux install step 2

Warning: If you have any problems during connection of VerontePipe, please check the firewall configuration of the application and disable it. To disable the firewall, go to “Firewall Configuration” panel and be sure the firewall Status is set on “Off”.

Deactivating the antivirus can be useful too. Disabling the antivirus depends on your antivirus software.



Firewall Configuration step 1



Firewall Configuration step 2

4.2 Update

A setup wizard will be displayed in order to guide the user during the update process.

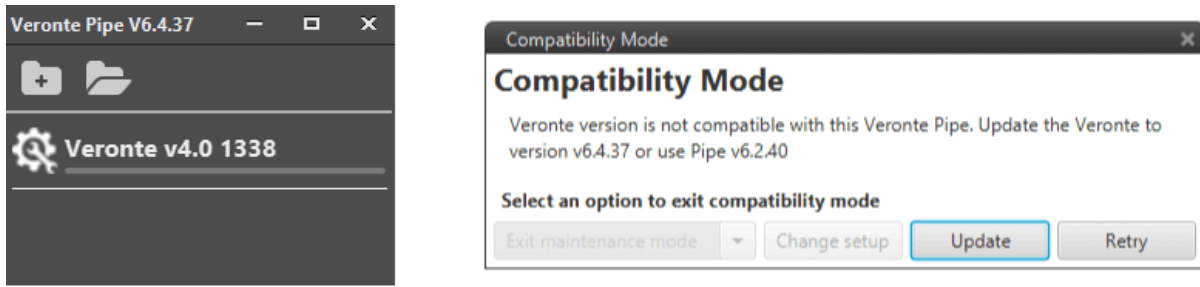
Warning:

- Although newer versions are usually compatible with older ones, when upgrading the system, updates must be done in the correct order. It is mandatory to update **Veronte Pipe** first, then **Veronte Onboard** and finally **Veronte on the Control Station**. Otherwise, part of the system could become unreachable.
- Never turn off Veronte during the update process. It could cause irreversible damage to the unit.
- When VerontePipe is updated be careful with managing .ver files. If the file was saved in a newer VerontePipe version, it will not be open in the actual one.

| Current VerontePipe | PDI last saving | Compatibility |
|---------------------|-----------------|---------------|
| 6.4.x | 6.4.x | OK |
| 6.4.x | 6.2.x | OK |
| 6.2.x | 6.4.x | NO |

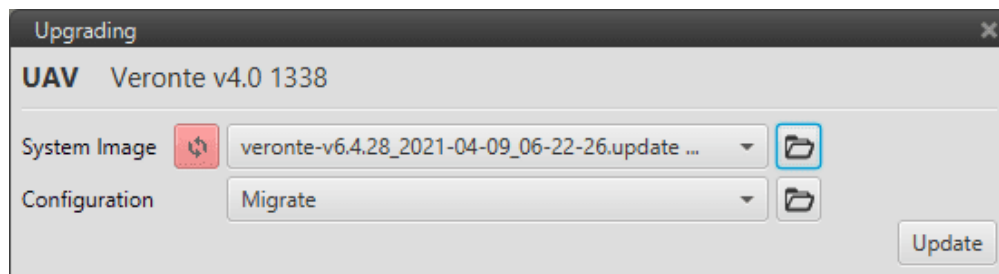
When having the last version of Veronte Pipe installed in the computer, now is the time to update the software of the

autopilots. In order to do that, click on the unit you wish to update from the sidebar. Then the window shown on the right of the following figure will appear.



Compatibility Mode

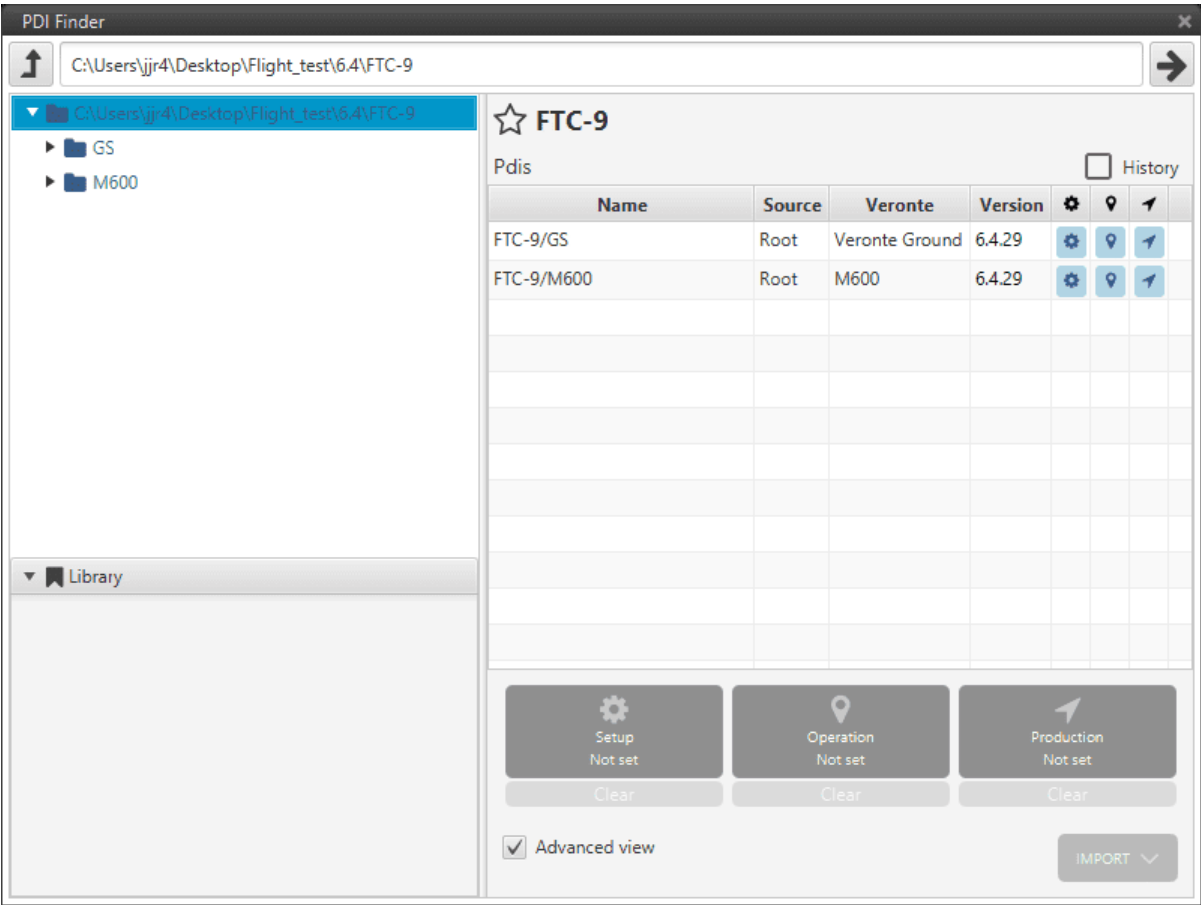
When clicking on Update, the next dialogue will appear on the screen.



Update Veronte Autopilot

User can select:

- **System image:** With internet connection, the upgrading process will download the last version of the onboard software (in this case v6.4.28). User can also select a different system image from the PC.
- **Setup:** Default option of the upgrading is the migration of the Setup. By clicking on the folder icon, user can open the following window and choose a way to upload a different Setup: Template, Select from PC or from Library.



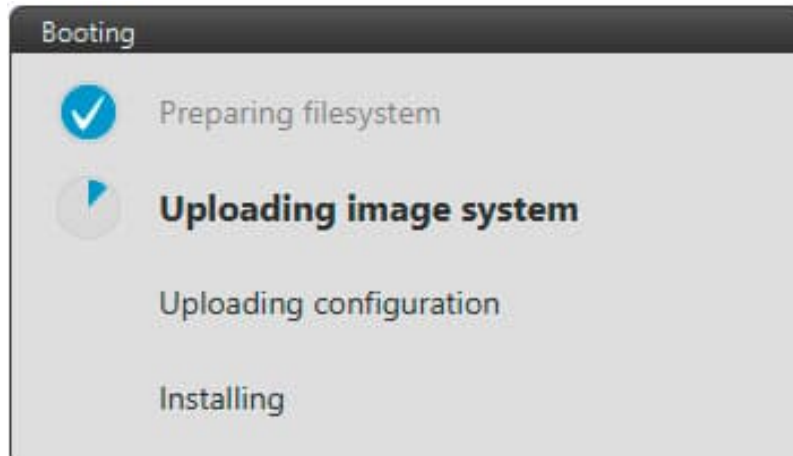
Veronte File Selection

The previous window shows info about Pipe version used to save the configuration.

When the booting is completely configured, the user can launch the Update by clicking on the correspondent button and waiting until the end of the process.

Warning:

- During the update, the system will reboot so never perform an update during an operation.
- Make sure you choose the right Veronte Update file for the selected Veronte Autopilot.



Booting Process

4.3 Preferences

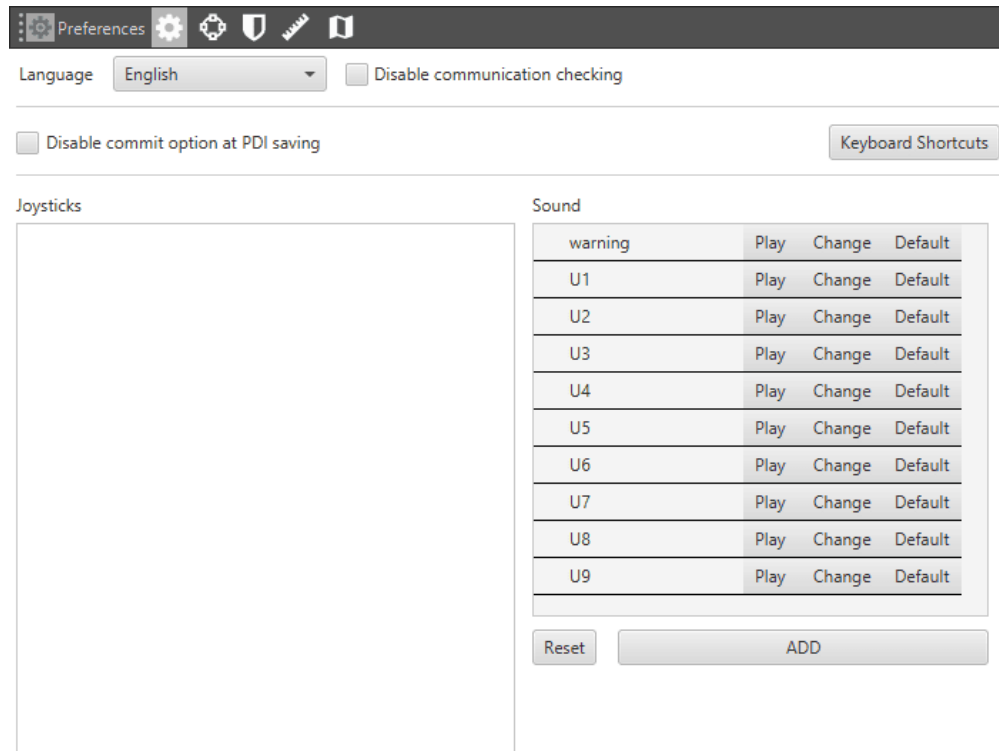
4.3.1 General

If you click in:



General Tool

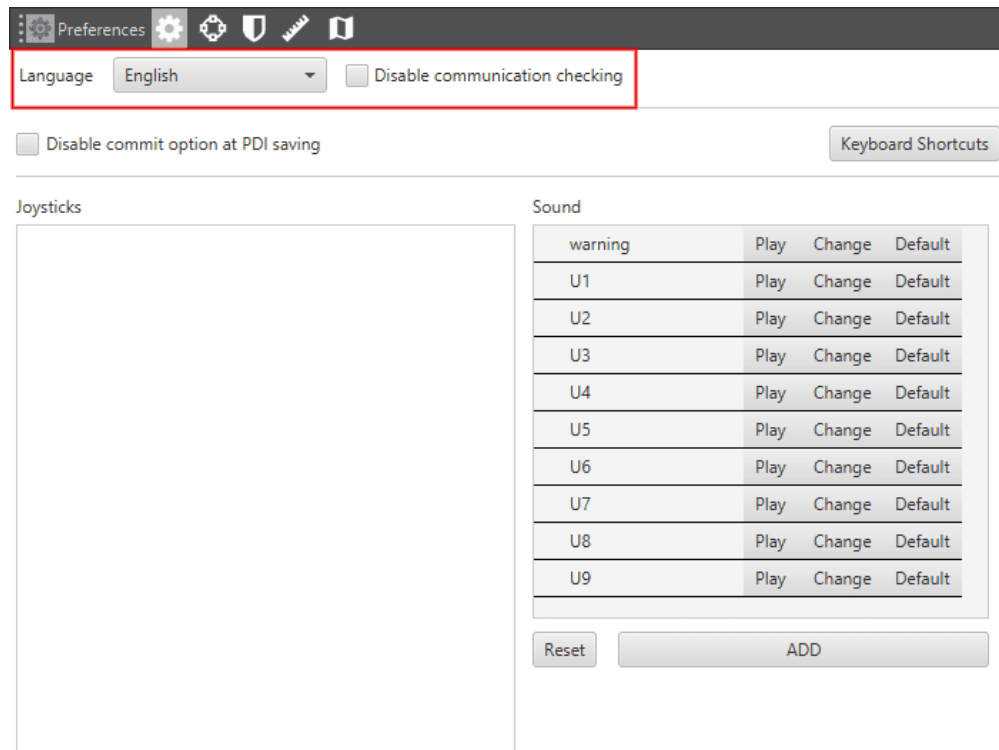
This menu allows the user to modify the general configuration for Veronte Pipe.



Software Setup

Where we will found:

- **Language:** The user can choose the language that he/she wants. If Disable communications checking option is selected, degraded communications mode is disabled.



Language

- **Disable commit option at PDI saving.**

Preferences

Language

English

☐ Disable communication checking

☐ Disable commit option at PDI saving

Keyboard Shortcuts

Joysticks

Sound

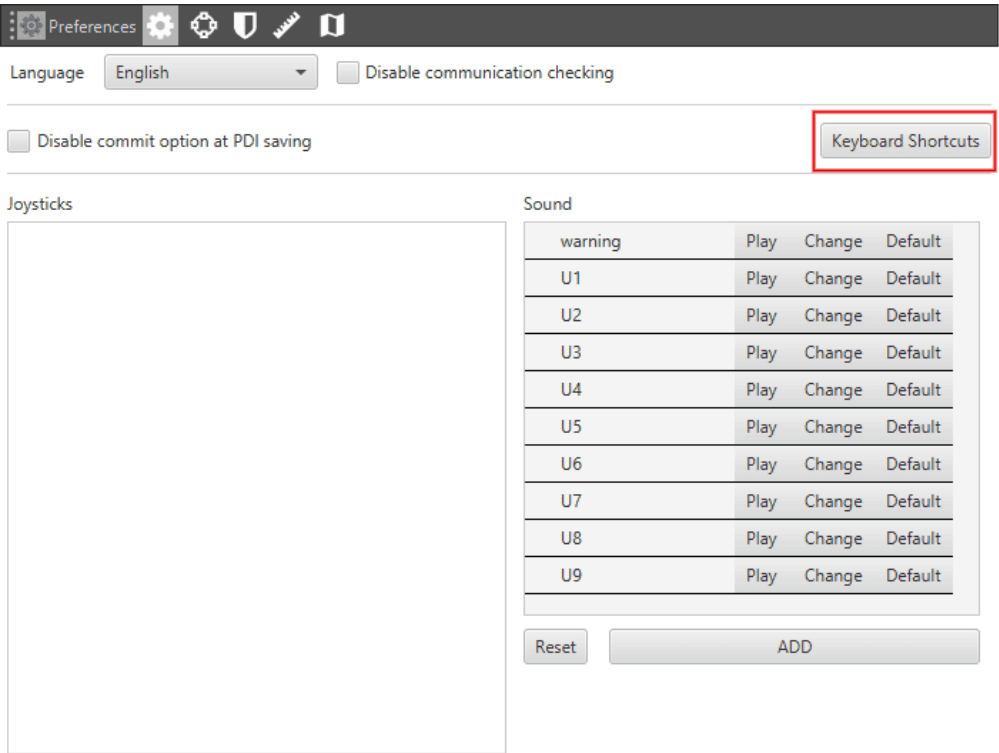
| | | | |
|---------|------|--------|---------|
| warning | Play | Change | Default |
| U1 | Play | Change | Default |
| U2 | Play | Change | Default |
| U3 | Play | Change | Default |
| U4 | Play | Change | Default |
| U5 | Play | Change | Default |
| U6 | Play | Change | Default |
| U7 | Play | Change | Default |
| U8 | Play | Change | Default |
| U9 | Play | Change | Default |

Reset

ADD

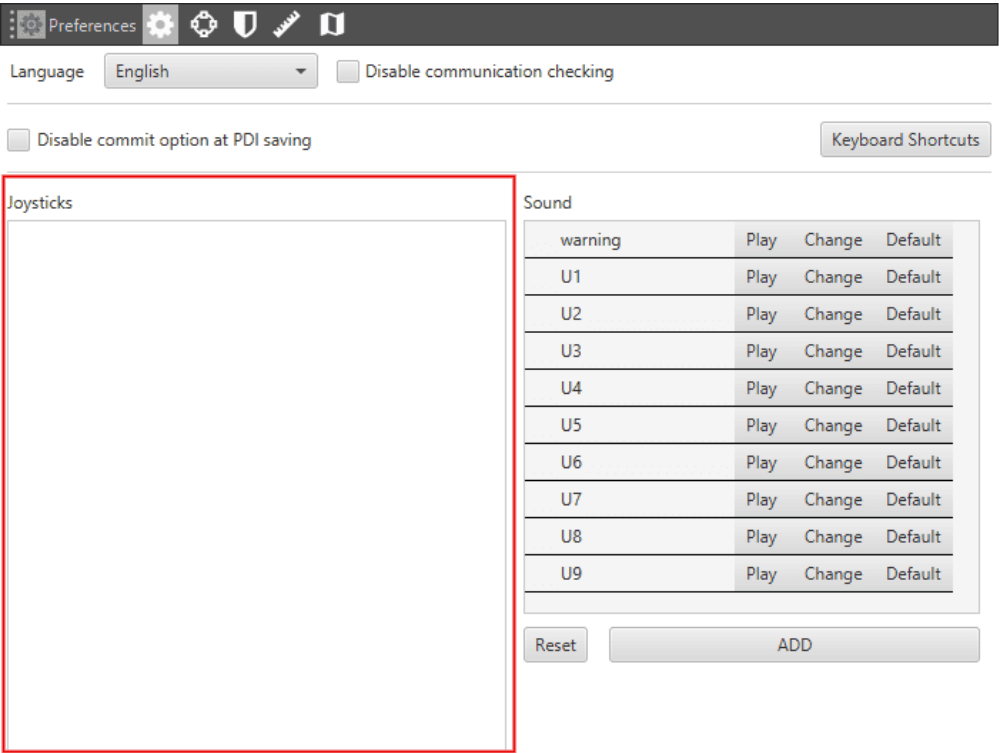
Disable commit option at PDI saving

- **Keyboard Shortcuts:** key combination to get quicker some pipe configurations.



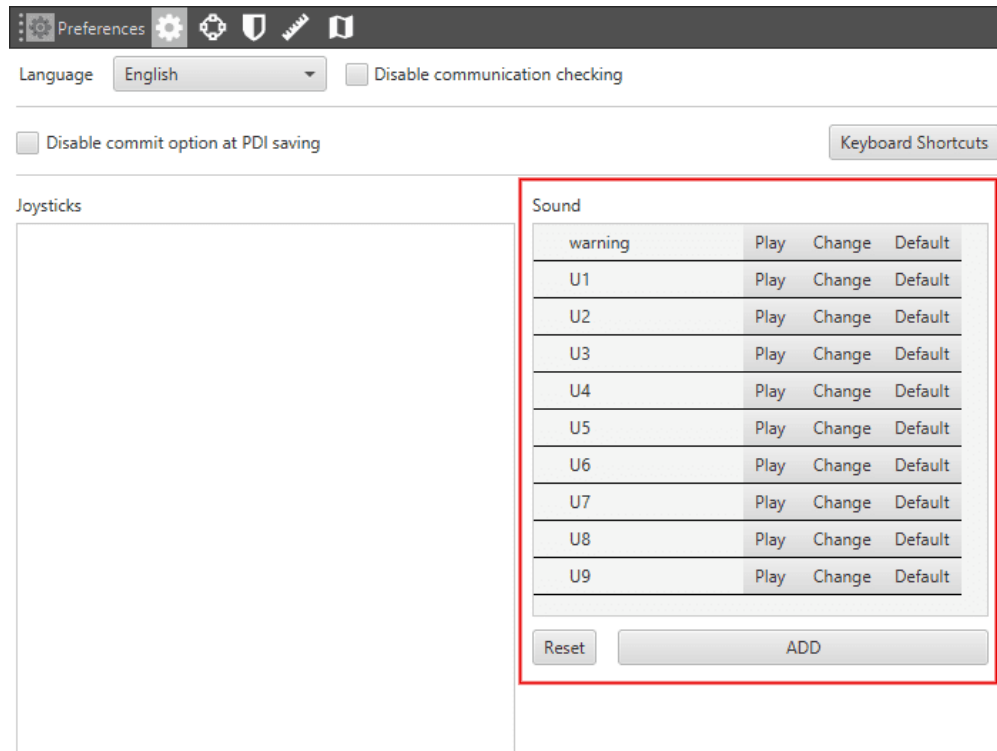
Keyboard Shortcuts

- **Joysticks:** this box shows you the joysticks that you have installed in your system.



Joysticks

- **Alert audioclips:** Alert Audioclips are used to manage audio files used on the application. They can be associated with system alerts on the Workspace configuration. There are nine audio files plus the alert_audioclip.
 - To substitute an audio file for another one, the Change button displays a browser to select a mp3/wav file stored in the computer.
 - It is possible to return to the standard alert by clicking Default.



Alert Audioclips

To configure the alerts on a certain scenario visit [Gauge Display](#) on Workspace section.

4.3.2 Connections

If you click in:



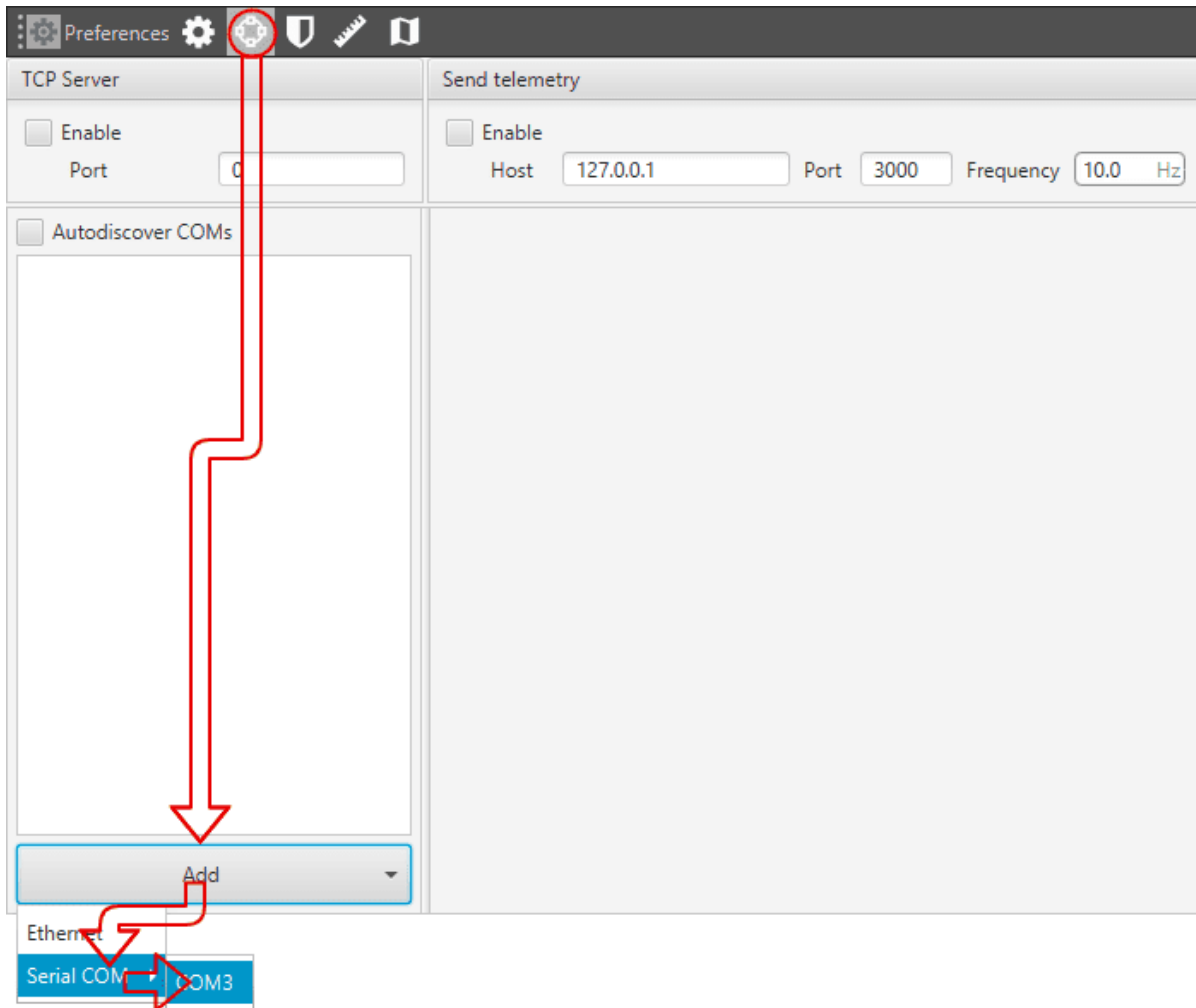
Connections Tool

You can open the connections menu:

The screenshot shows the 'Preferences' window of the Veronte Autopilot software. The window has a dark header bar with icons for 'Preferences', a gear, a network icon, a shield, a key, and a book. The main area is divided into two panels. The left panel is titled 'TCP Server' and contains an 'Enable' checkbox (unchecked), a 'Port' field with the value '0', and a section titled 'Autodiscover COMs' with a checked checkbox and an empty list box below it. At the bottom of this section is an 'Add' button. The right panel is titled 'Send telemetry' and contains an 'Enable' checkbox (unchecked), a 'Host' field with the value '127.0.0.1', a 'Port' field with the value '3000', and a 'Frequency' field with the value '10.0 Hz'.

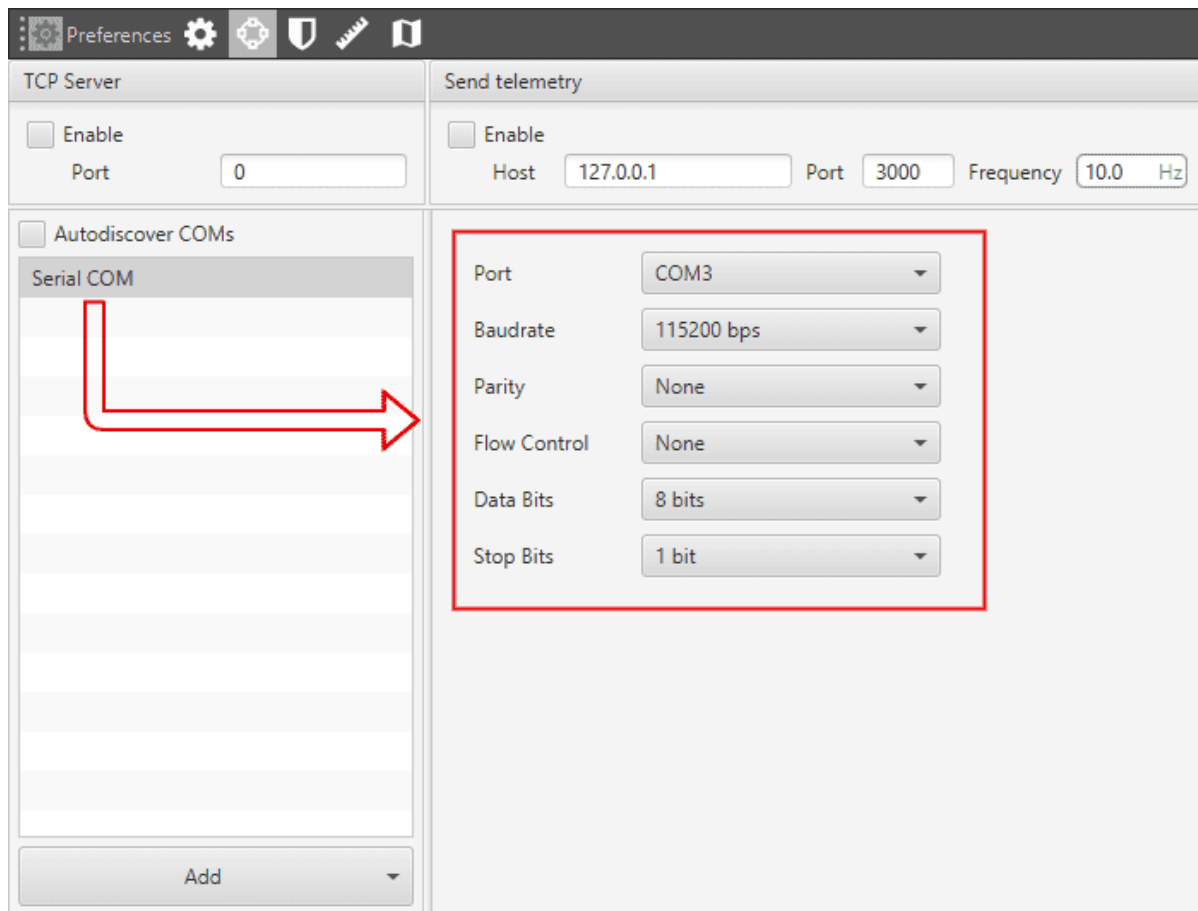
Connections Menu

By default, Autodiscover COMs option is selected to be able to recognize Veronte Units. However, they can be added manually by unchecking this option following the steps below.



New Veronte

In case of using a different interface of connection, it may require changing some parameters.



New Veronte Configuration

The following parameters can be edited in the menu shown above:

- **Baudrate:** this field specifies how fast data is sent.
- **Parity:** this field is a way of low-level error checking. It can be in odd or even.
- **Flow Control** RTS/CTS and XON/XOFF control can be configured if needed.
- **Data Bits:** this field defines the bits number of the message.
- **Stop:** number of the stop bit.

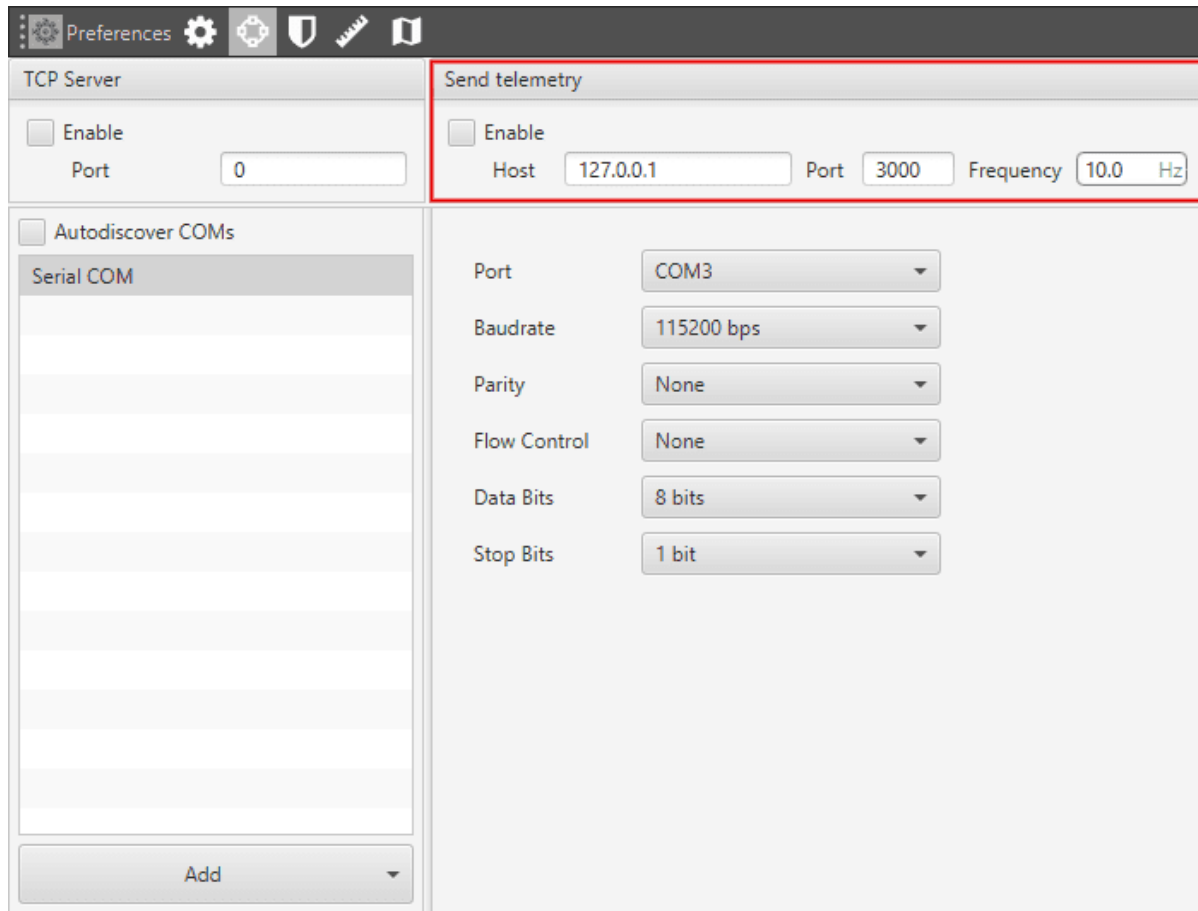
| Type | Bits |
|--------|------|
| Start | 1 |
| Data | 4-8 |
| Parity | 1-2 |
| Stop | 1-2 |

- **TCP Server:** Pipe gets information from the configured Port of the TCP (Transmission Control Protocol) Server.

The screenshot shows the 'Preferences' window of the Veronte Autopilot software. The 'TCP Server' section is highlighted with a red box. It contains an 'Enable' checkbox (unchecked) and a 'Port' field set to '0'. Below this is an 'Autodiscover COMs' section with a 'Serial COM' list and an 'Add' button. The 'Send telemetry' section contains an 'Enable' checkbox (unchecked), a 'Host' field set to '127.0.0.1', a 'Port' field set to '3000', and a 'Frequency' field set to '10.0 Hz'. Below these are several dropdown menus for 'Port' (set to 'COM3'), 'Baudrate' (set to '115200 bps'), 'Parity' (set to 'None'), 'Flow Control' (set to 'None'), 'Data Bits' (set to '8 bits'), and 'Stop Bits' (set to '1 bit').

TCP Server

- **Send-telemetry:** Pipe sends the telemetry document through the configured parameters.



Send Telemetry

4.3.3 Encryption

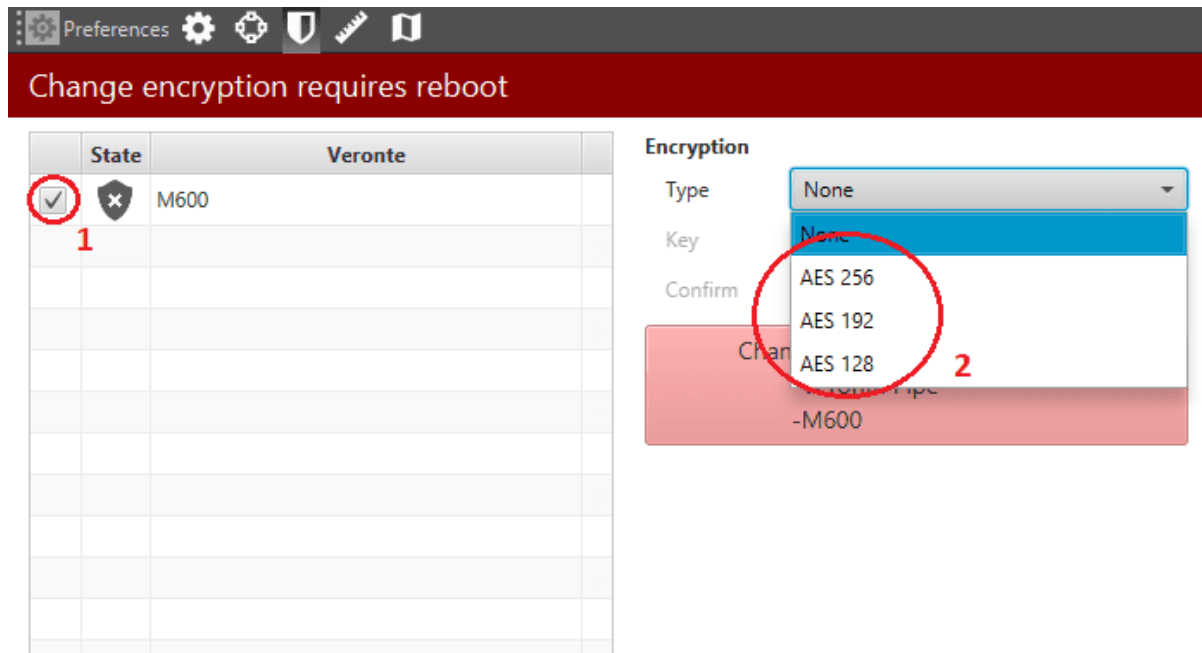
If you click in:



Encryption Tool

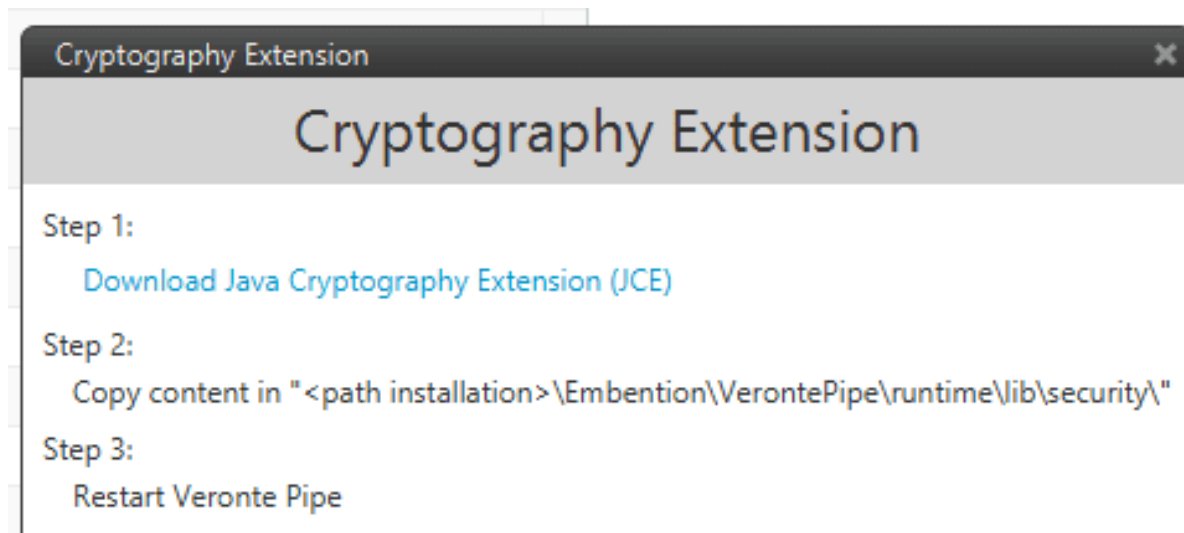
Veronte Pipe contains the option to encrypt the data transmitted from the autopilots and from Pipe itself. To configure this feature go to **Preferences** (Veronte Pipe) – **Encryption**.

1. Select the System that will be encrypted on the left side of the menu. By default, the unit set to encrypt is Veronte Pipe.
2. Select an Encryption Type. There exist three different encryption types, depending on the key size: 128, 192 and 256 bits. If one of the last two is selected, an additional step is required.



Encryption Menu

Warning: with 192 and 256 an additional step is required. Press The Shield to open the window that provides the instructions to ensure the functioning of AES (Advanced Encryption Standard).



Advanced Encryption Standard

1. Specify a Key, this will be used to encrypt the data and can be used later to get the information back to a readable format.
2. Press the button Change the default encryption, a pop-up window will appear.
3. Restart the encrypted autopilots.

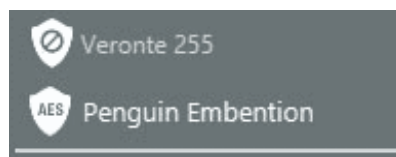


Veronte Restarting

Once autopilots are restarted they will show up in the right bar of Veronte Pipe. If both autopilot and Pipe are encrypted nothing will change in the view. If only the autopilots are encrypted (or the autopilots are connected to different encrypted Veronte Pipe), they will show up as shown in the following figure.



Both autopilots encrypted

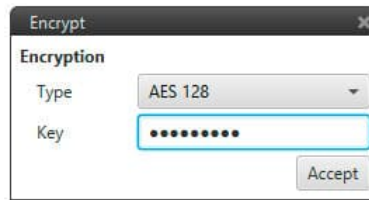


Only Ground autopilot encrypted

If the autopilot icon appears with the label “Veronte X” (where X is the ID of the autopilot), it is not possible accessing to its configuration and user can not use it.

4.3.3.1 Remove Encryption

To remove encryption temporarily and be able to use the autopilot it is necessary to left-click on the autopilot label and insert the encryption type and password in the following window.

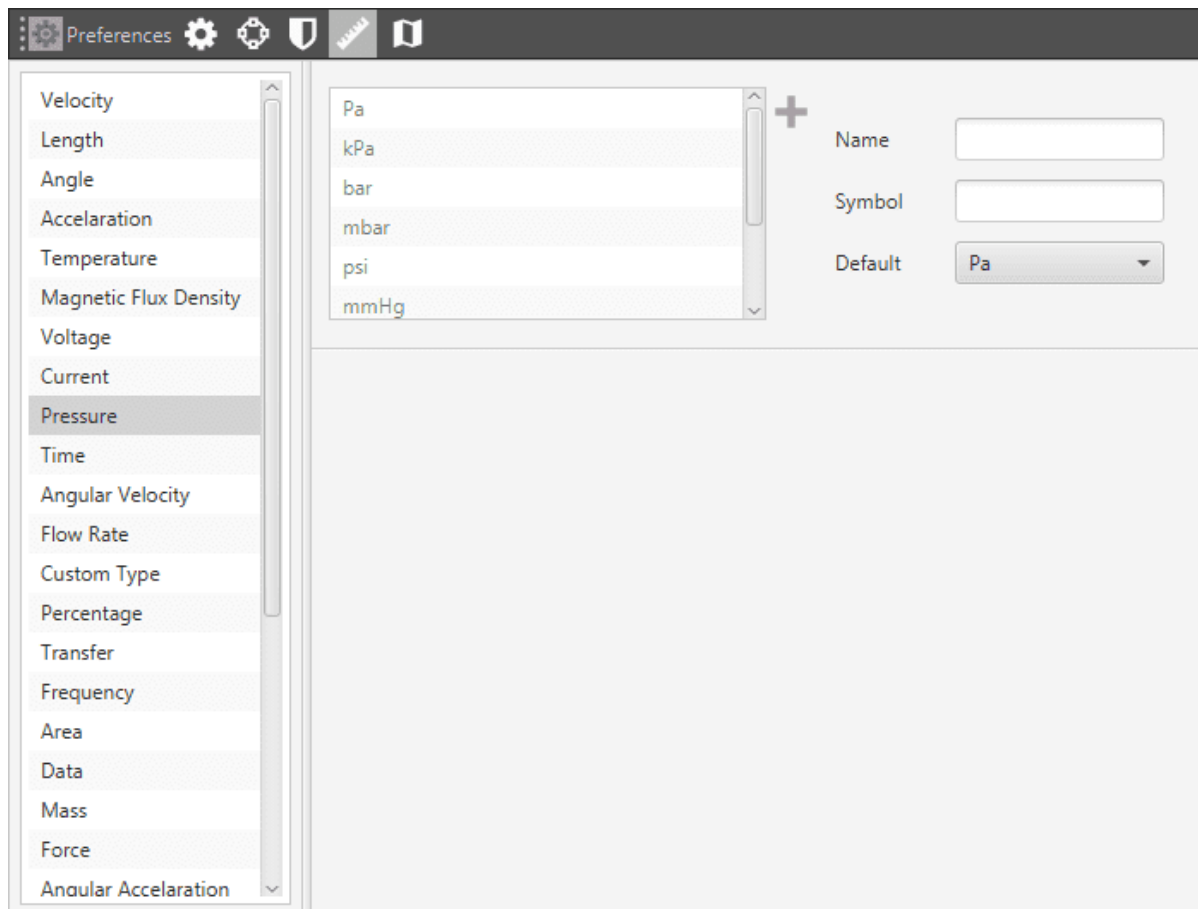


Remove temporarily encryption

To remove definitely the encryption it is sufficient to go in the Encryption Panel and selecting the encryption type None. The software will ask a last time for the password and, after the autopilots restarting, the encryption will be removed.

4.3.4 System Units

This panel shows all the system units available for the system variables. They are arranged according to the type of variable (velocity, acceleration, temperature, etc). It can be found in **Preferences – Units**.

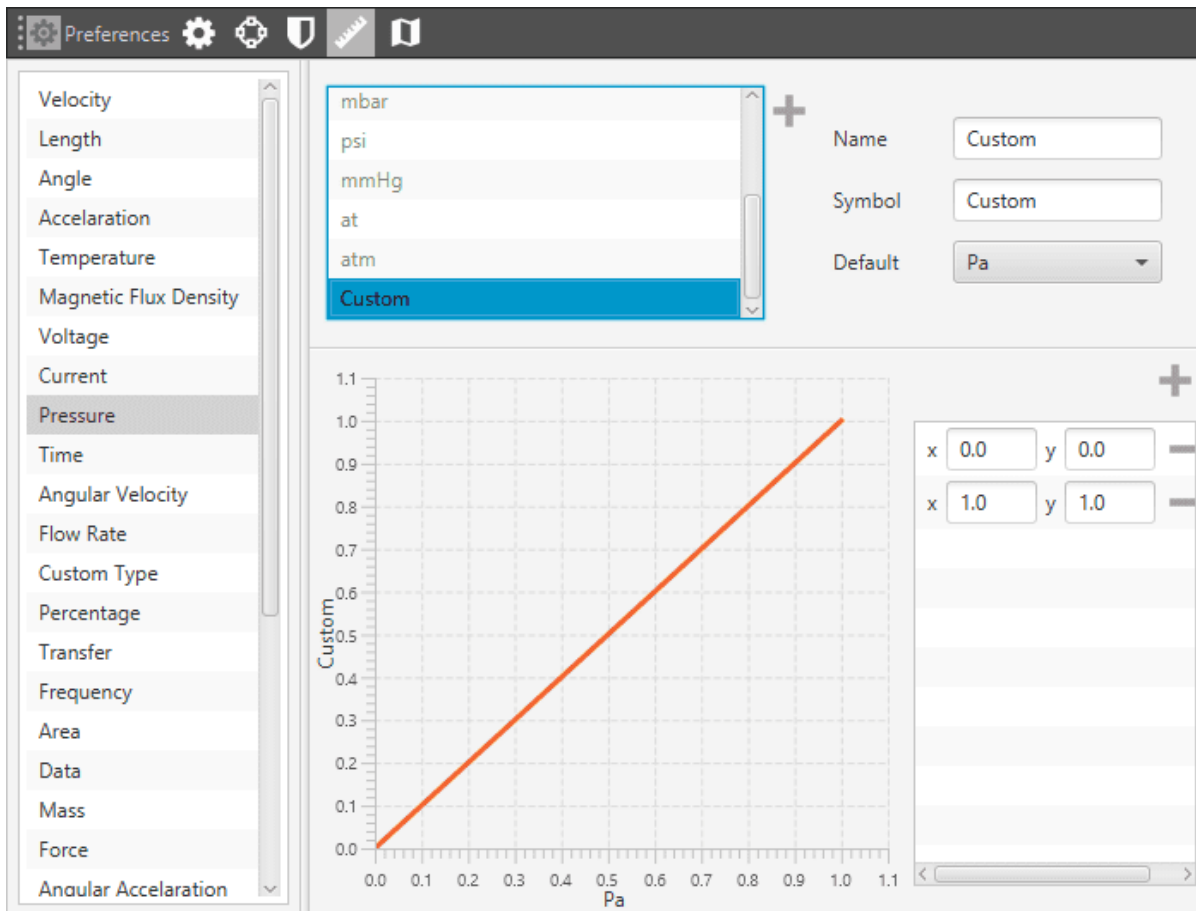


System Units

Veronte Pipe works by default with units of the International System, but these ones can be changed by selecting others from the ones shown when pressing Default option.

4.3.4.1 Creating Custom Units

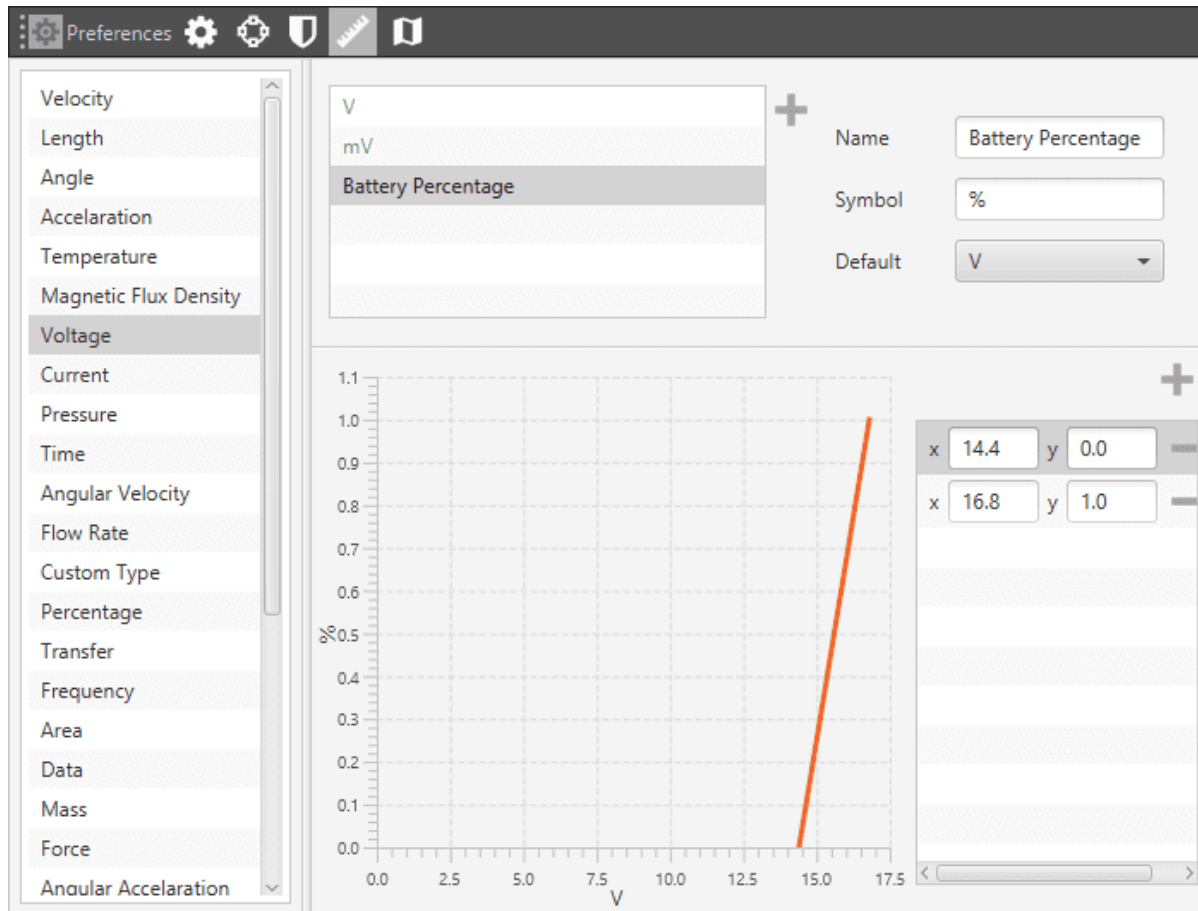
1. Click on “+” button, and a new variable will appear in the menu.
2. Introduce the conversion from the default unit to the new one. The conversion formula is created by introducing as many points as desired in the graph shown in the next figure.



Custom Unit Definition

An example is added to improve the understanding:

Considering the voltage of a LiPo battery, it could be useful to have a unit that indicates the remaining percentage of battery. For example, if the battery used is a 4S, the fully charged voltage is 16.8 V, while 14.4 will be the one when the battery does not provide the amount of energy needed to fly. So 16.8 V will correspond to 100% and 14.4 to 0%. Taking that into account the conversion graph will be the one shown in the next figure.



Voltage To Percentage Conversion

Warning: In this panel, it can be selected the default system units. It means the first displayed unit will be the selected one, but it will be possible to change it temporarily in all Pipe windows.

4.3.4.2 Units

The table below shows all the available units in VerontePipe.

| Variable Type | Units |
|-----------------------|--|
| Velocity | [m/s] [kt] [km/h] [mph] [ft/s] [mm/s] [ft/m] |
| Length | [m] [km] [mi] [NM] [yd] [ft] [in] [cm] [mm] |
| Angle | rad[-;]°[-180;180] rad[0;2] °[0;360] [° ‘ “] [° ‘ “](N/S) [° ‘ “](E/W) |
| Acceleration | [m/s ²] [ft/s ²] [in/s ²] [g] |
| Temperature | [K] [°C] [°F] |
| Magnetic Flux Density | [T] [mG] [gauss] [nT] |
| Voltage | [V] [mV] |
| Current | [A] [mA] |
| Pressure | [Pa] [kPa] [bar] [mbar] [psi] [mmHg] [at] [atm] |
| Time | [s] [min] [h] [s] [ms] [Time] |
| Angular Velocity | [rad/s] [rad/m] [rad/h] [rps] [rpm] [rph] [°/s] |

continues on next page

Table 1 – continued from previous page

| Variable Type | Units |
|----------------------------|---|
| Flow Rate | [m ³ /s] [gal/s] [gal/h] [l/s] [l/h] |
| Custom Type | [- -] |
| Percentage | [x1] [%] |
| Transfer | [pkts/s] |
| Frequency | [Hz] [mHz] [kHz] |
| Area | [m ²] [cm ²] [mm ²] [km ²] [mile ²] [ft ²] [yd ²] |
| Data | [bit] [byte] [KB] [GB] [bytes/s] |
| Mass | [kg] [g] [tonnes] [lbs] [oz] |
| Force | [N] [kN] [lbf] [pdl] |
| Angular Acceleration | [rad/s ²] [rad/m ²] [rad/h ²] [°/s ²] [°/m ²] [°/h ²] |
| Baudrate | [Bd] [kBd] [MBd] |
| Pressure Variance | [Pa ²] |
| Magfield Variance | [T ²] |
| Velocity Variance | [(m/s) ²] [(cm/s) ²] [(mm/s) ²] |
| Numeral System | [bin] [octal] [dec] [hex] |
| Pressure Square Error Rate | [Pa ² /s] |
| Centimeters/Pixels | [cm/pixel] |
| Jerk | [m/s ³] |
| Power | [W] [kW] [Kgm/s] [erg/s] [CV] |

4.3.5 Map

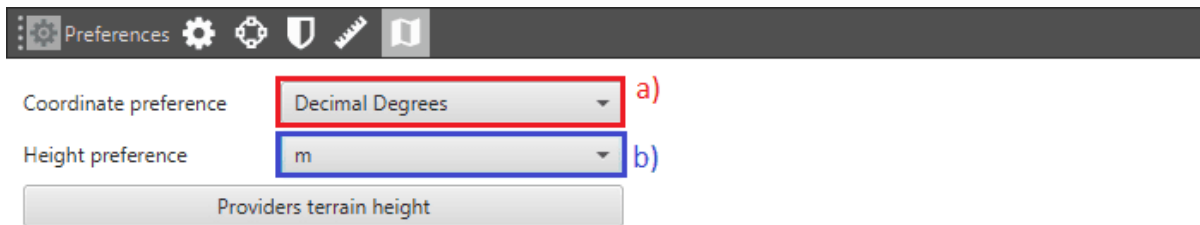
If you click in:



Map Tool

In the map menu, the user can choose the coordinate preference (from MGRS, Decimal Degrees and Degrees) and download the Terrain Height.

4.3.5.1 Terrain Height



Map Menu

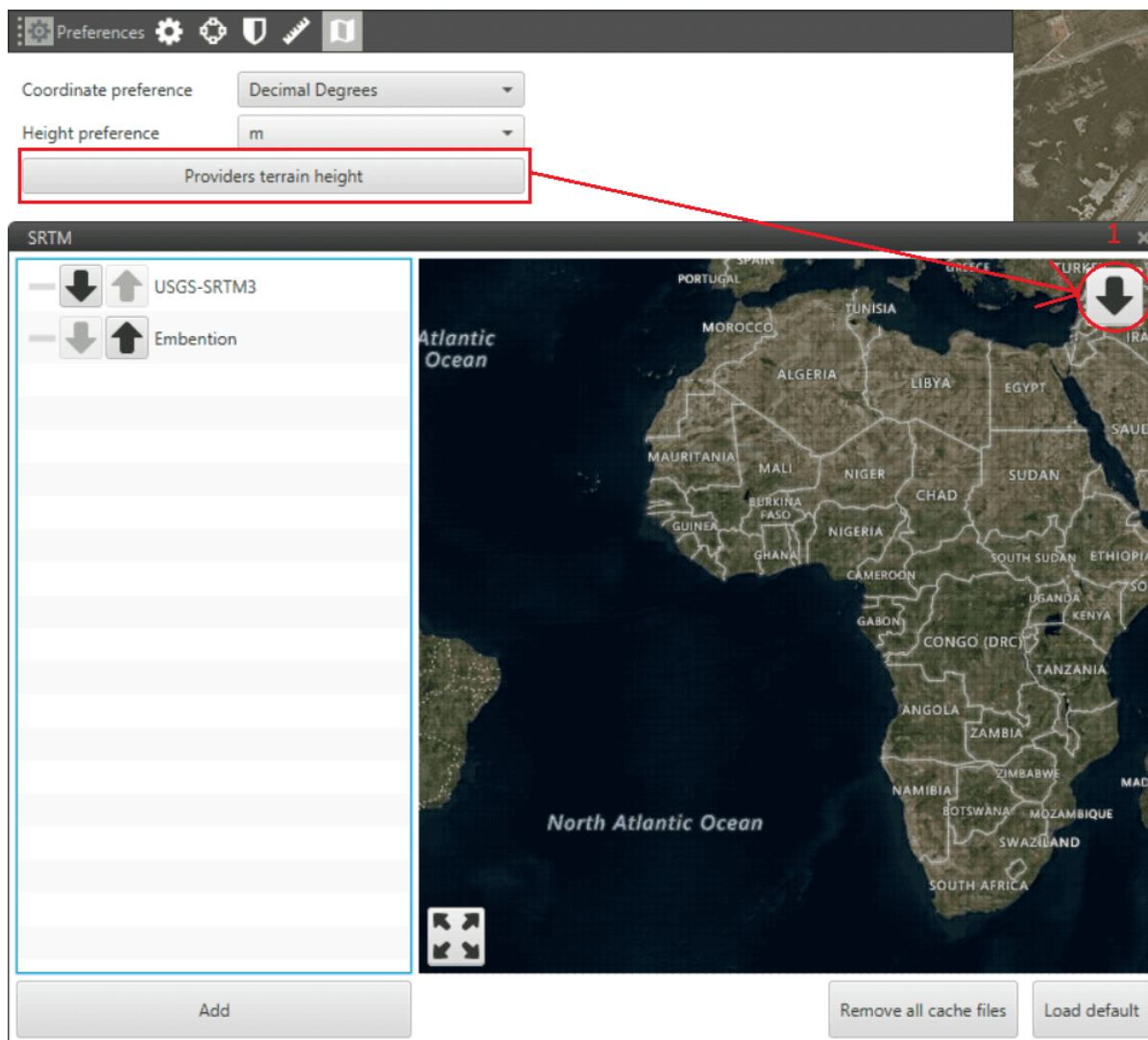
You can select the units of the coordinates clicking on **a)**. You can also select the units of the height on clicking on **b)**.

In order to perform the altitude estimation, Veronte installs a GIS (Geographic Information System) consisted of two meshes. When creating a mission, the terrain altitude on the mission area is automatically downloaded from the internet.

4.3.5.2 Download Altitude Information

If the flight operation is carried out in an area where there is no internet connection, the altitude information can be downloaded a priori. To download the information:

1. Select **Providers terrain height** and press button **1**.



Terrain Height

1. Click on a quadrant to download the information of that zone into the system.

There are two providers of altitude information in Pipe:

- The first one is the **SRTM3** (shuttle radar topographic mission 3) which contains the altitude information for the territories outside the United States (the number 1 is for USA).
- The other one is a server created by **Embention** with altitude information of areas that do not appear on the SRTM such as Iceland.

4.3.5.3 Custom Terrain File

Pipe also provides the option of introducing a custom terrain file.

Clicking on **Add** appears a menu with two options to charge a terrain file in the system:


- **Local:** allows the user to select a file from the computer.
- **Remote:** gives the option to introduce a URL (Un*iform Resource Identifier) of an altitude provider from both a web or FTP (File Transfer Protocol) server.

The 'Add' dialog box contains the following elements:

- Name:** A text input field.
- Local:** A radio button that is selected, with a 'Folder' button next to it.
- Remote:** A radio button that is unselected, with a text input field containing 'http/' next to it.
- Type:** Two radio buttons labeled 'Web' (selected) and 'FTP'.
- Save:** A button at the bottom right.

Custom Terrain Altitude Source






This section contains all preferences settings for Veronte Pipe (software). To access them, the user has to use the Preferences Toolbar.

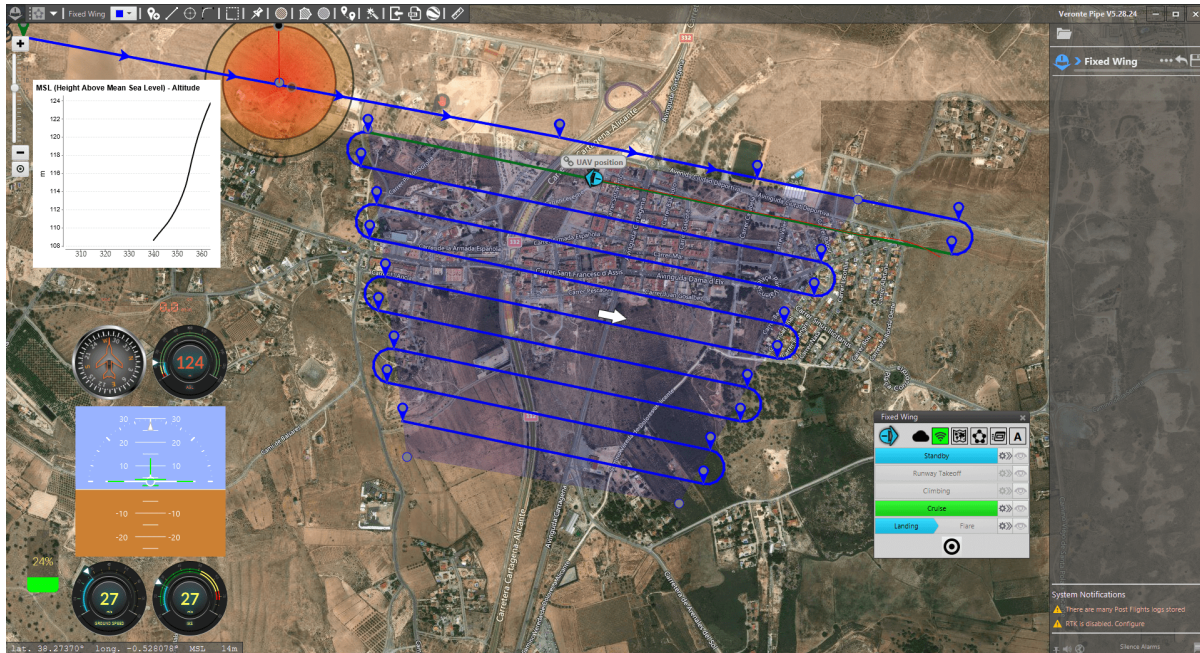
Click on  (Main Menu), and then select Preferences in the pull-down menu, a new toolbar with different options will appear.



Preference Toolbar

The different elements of the setup toolbar are detailed in the following table.

| | Item | Description |
|---|-------------|--|
|  | General | Displays general configurable fields |
|  | Connections | Open connections configuration |
|  | Encryption | Displays encryption configuration |
|  | Units | Open system units configuration |
|  | Map | Coordinate preference and height terrain providers |



Veronte System Overview

Veronte Pipe software is designed to operate any unmanned vehicle using Veronte as the onboard autopilot. With Veronte Pipe, users have an easy-to-use application with real-time response and telemetry with safety features.

It has been developed using software standard model of IEEE STD 830-1998, Recommended Practice for Software Requirements Specifications (SRS) and STANAG 4671 documentation, subpart about UAV Control Stations adapted to Veronte system.

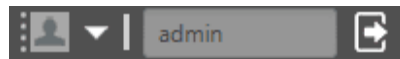
Veronte Pipe includes:

- **Telemetry:** Real-time onboard UAV metrics, such as sensors, actuators and control states.
- **Telecommand:** Support for all synchronous operator control commands that can be sent to the flight segment, e.g. operational mode switch, mission management, payload control.
- **Mission design:** User defined, predefined mapping and drop missions configuration as well as inflight mission edit.
- **Mission analysis:** Post-flight viewer, reproduce all recorded data from a previous flights and generate plots and reports.
- **Configuration:** Edit vehicle settings, such as servo trim, interface/port management, modes and automations.
- **Multiple Users:** One or more operators can work simultaneously.


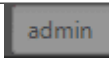

USERS & ACCESS LEVEL

5.1 User Toolbar

The user toolbar displays the options related to the configuration of the user account.



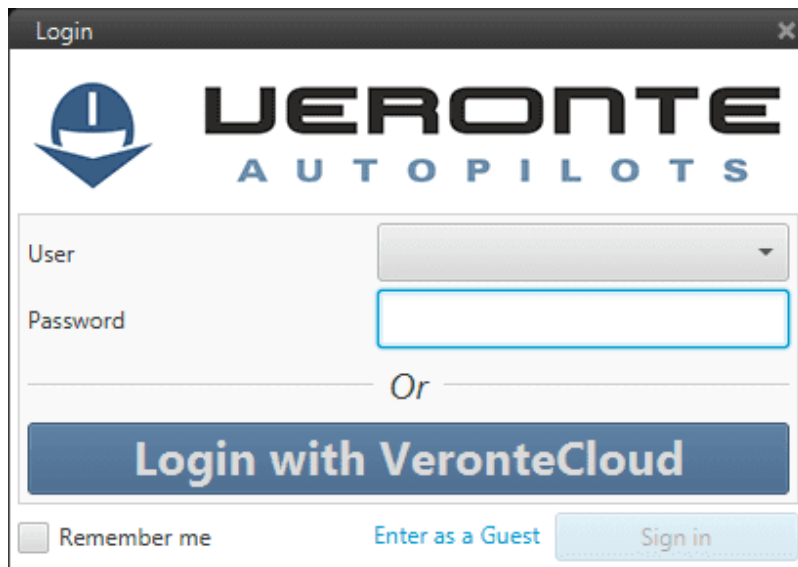
User toolbar

| | | |
|---|---------|-------------------------------|
|  | Details | Displays configurable fields. |
|  | User | Logged In user. |
|  | Log Out | Log Out from Veronte Pipe. |

5.2 User Manage

On the first software run, it will start with the default user:

- User: admin
- Password: admin

A screenshot of the Veronte Autopilot login window. The window has a title bar that says "Login" with a close button. Inside, there's a logo on the left consisting of a blue circle with a white 'I' and a downward arrow. To the right of the logo is the text "VERONTE" in large, bold, black letters, and "AUTOPILOTS" in smaller, blue, spaced-out letters below it. Below the logo and text are two input fields: "User" with a dropdown arrow and "Password" with a text box. Below these fields is a horizontal line with the word "Or" in the center. Underneath is a large blue button with the text "Login with VeronteCloud" in white. At the bottom, there are three elements: a checkbox labeled "Remember me", a blue link that says "Enter as a Guest", and a light blue button that says "Sign in".

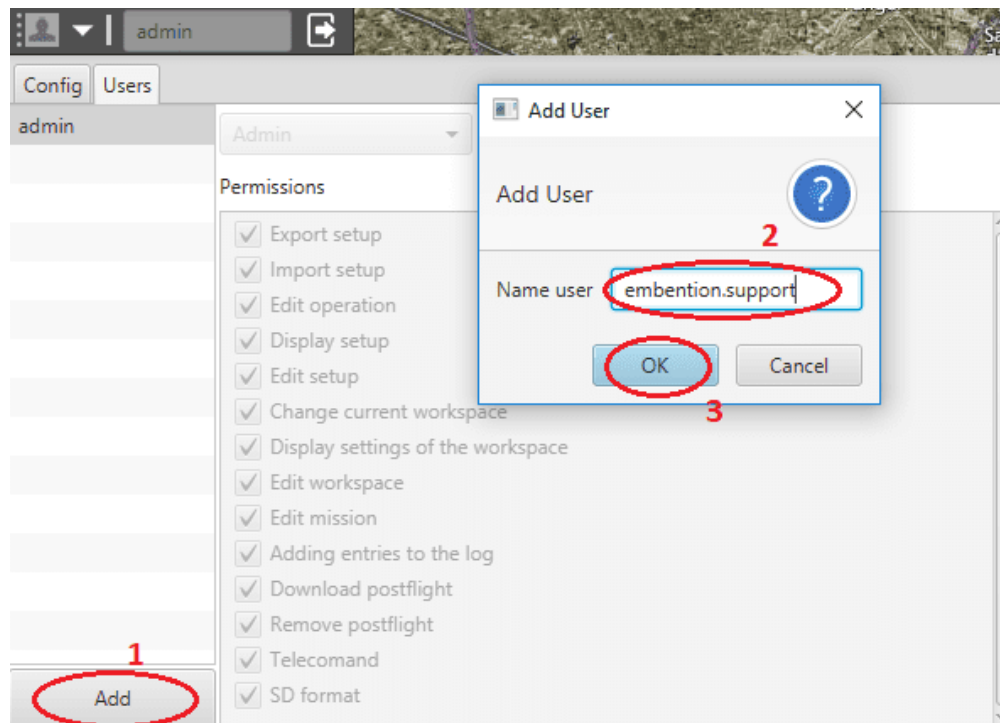
Default user

After the first use of Veronte Pipe, it is possible to change the username and the password. Moreover, Pipe gives the option of limiting the use that each user has over the software, what could be considered as a safety measure to limit the thing that can be changed for each person according to its knowledge over the different parameters. That is done through the permissions, where the options that can be changed by the user can be selected.

Warning: The user “admin” always will have all the permissions.

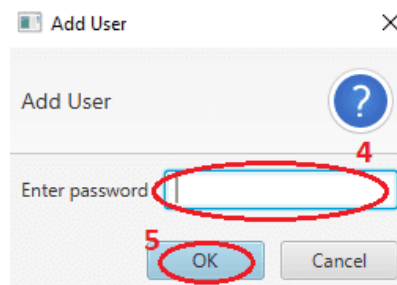
5.3 New User

You can create a new user and give him the permissions that you want.



New User

1. Click in "Add".
2. Write the new user name.
3. Click "Ok".
4. Write the password.
5. Click Ok.



New User Password

When the new user is created, you can give him the permissions that you want as a admin.





WORKSPACE

6.1 Workspace Toolbar

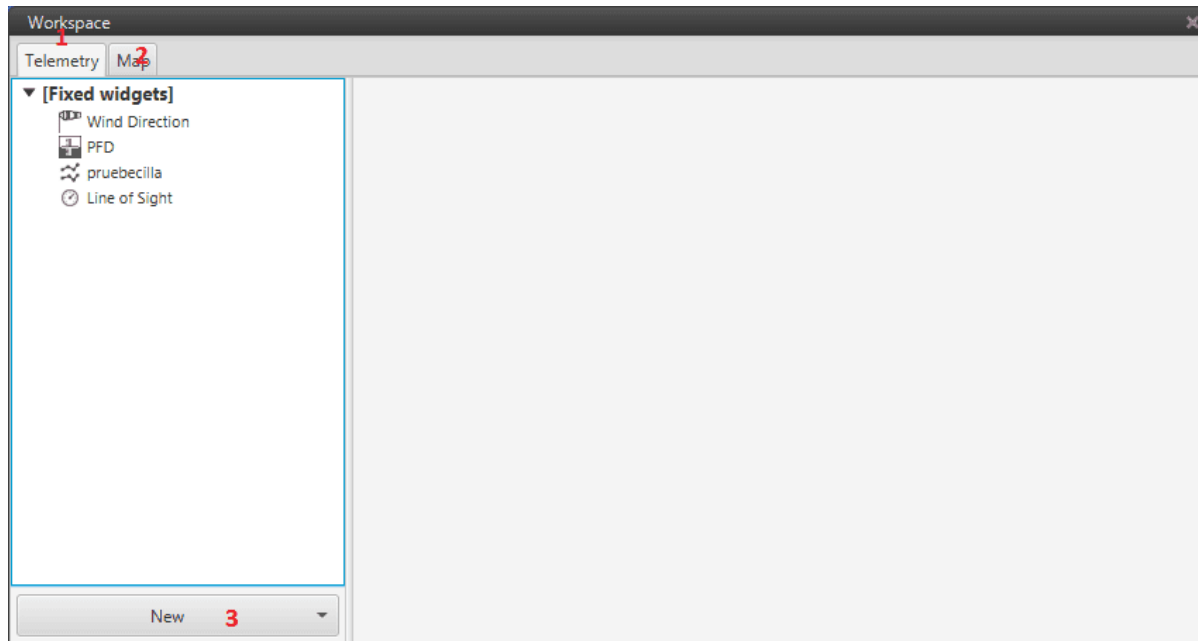
Workspace settings allow the user to customize any information to be displayed on the screen for monitoring the operation. Custom workspaces can be created, set any workspace as default in order to open it automatically on system start. All these options are available in the Workspace Toolbar, which is shown on the image below.



Workspace Toolbar

| | |
|---|-------------------------------------|
|  | Show/Hide Workspace |
|  | Enable/Disable Worspace Telecommand |
|  | Edit Workpace |
|  | Many options to manage workspace |

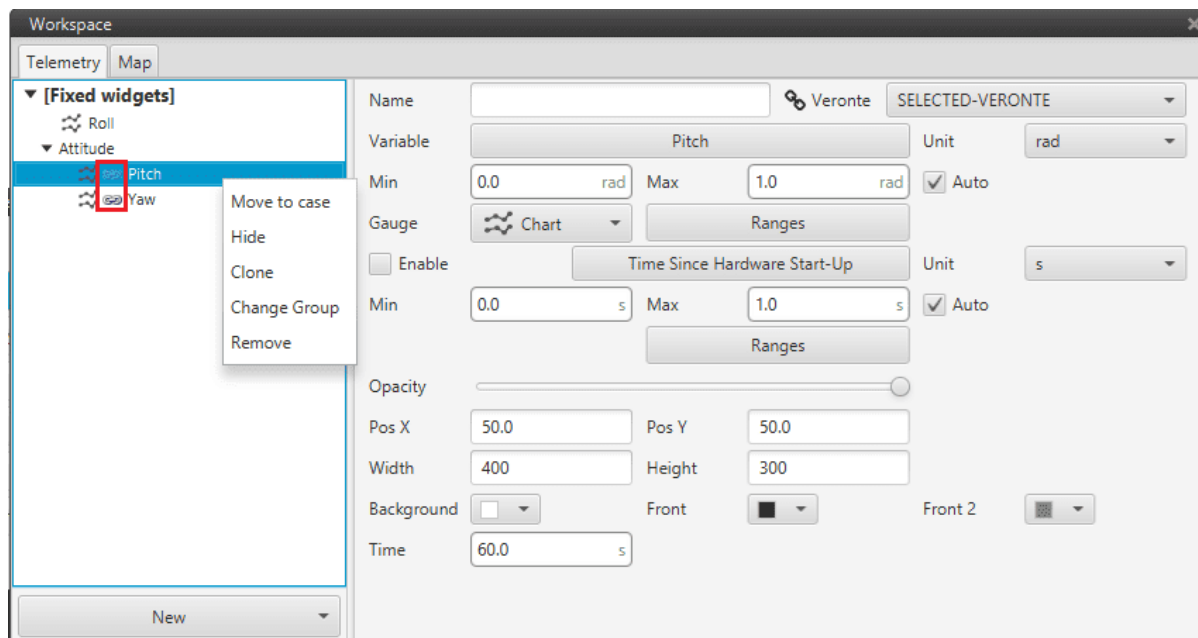
6.1.1 Edit Workspace



‘Open Details’ Menu

1. **Telemetry:** Displays a list with all the displays superposed to the map (widgets).
2. **Map:** Allows to set a map or background color to the workspace.
3. **New:** Allows the addition of new displays which can be configured in **Widgets** and **Main**.

By right clicking on an item in **Telemetry** or **Map**, the following dropdown list appears:



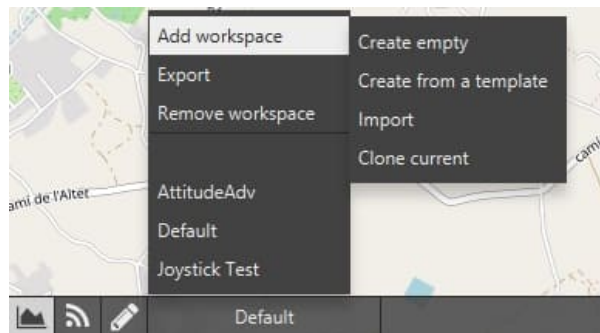
- **Move to case.**
- **Hide:** Occults the selected element.

- **Copy:** Stores in the clipboard the selected element.
- **Change Group:** Opens a window to type the group to place the element in. If there is no group created or the name typed does not belong to any group, this option will create one and store the element in it.
- **Remove:** Deletes the selected element.

The option highlighted in red is the option to lock the chart. If all the elements of the group are marked they will move together, if one is not marked it could be moved with respect to the other group elements.

6.1.2 Add workspace

By clicking on the fourth icon or space (Previously on 'Edit' icon) a dropdown list appears with the current workspaces and some options. Workspaces can be created by clicking on **Add workspace**



New Workspace

The workspaces can be created from **Empty**, **Create from template**, **Import** or **Clone current**.

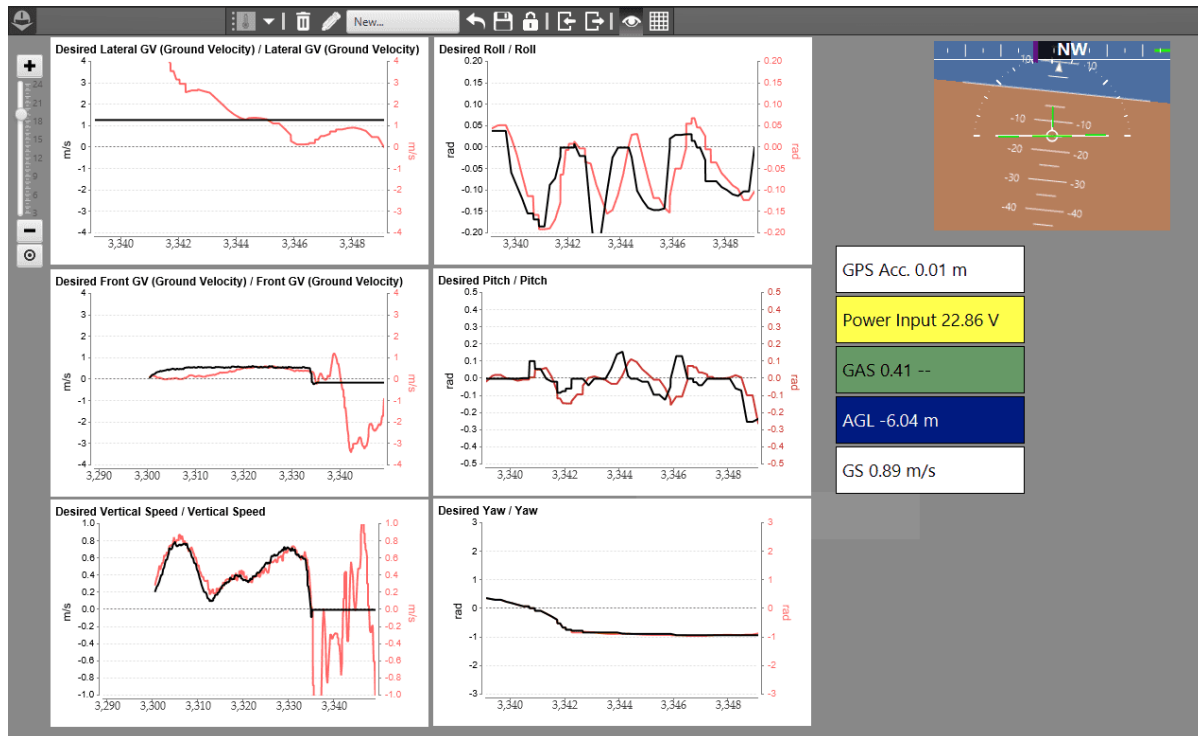
Workspaces can be modified too. The system allows the user to change the appearance and when it is completed, it can be exported.

Warning: Please note that Workspaces are not stored in Veronte Autopilot. Workspaces are located in the computer in which Veronte Pipe is installed.

In VerontePipe there are some templates available. When the workspace toolbar is open, the following templates can actually be opened:

- Attitude_Adv (Advanced)
- Attitude_PLANE
- Attitude_VTOL
- Default
- Joystick Test
- Map
- Output Test
- Sensors
- Stick Inputs

Following a picture of one of them:



Attitude Adv template

To do this is sufficient to open the toolbar, then the workspace selection and click on **New**.

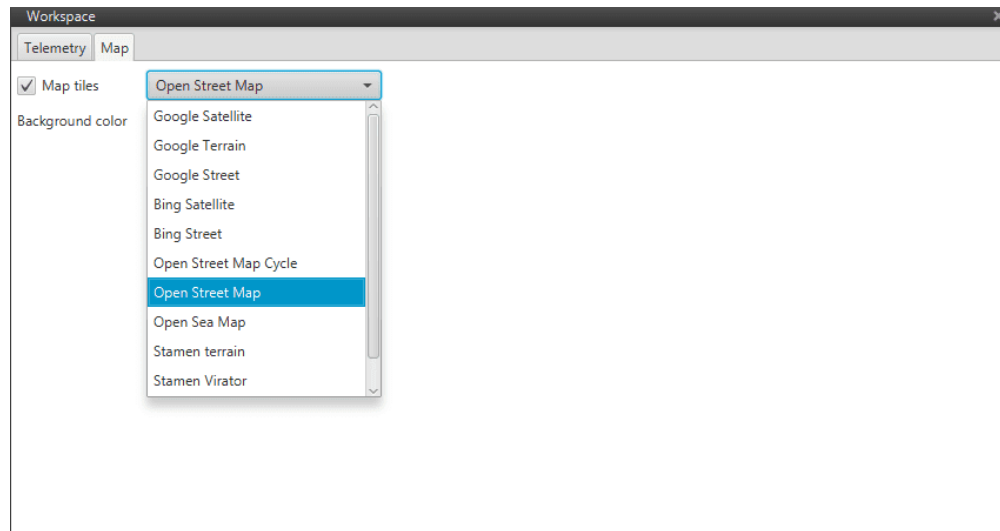
6.1.3 Import/Export Workspaces

It is possible to save the current workspace on a **.wsn** file, that can be loaded later in a different configuration.

6.2 Map Display

6.2.1 Map provider

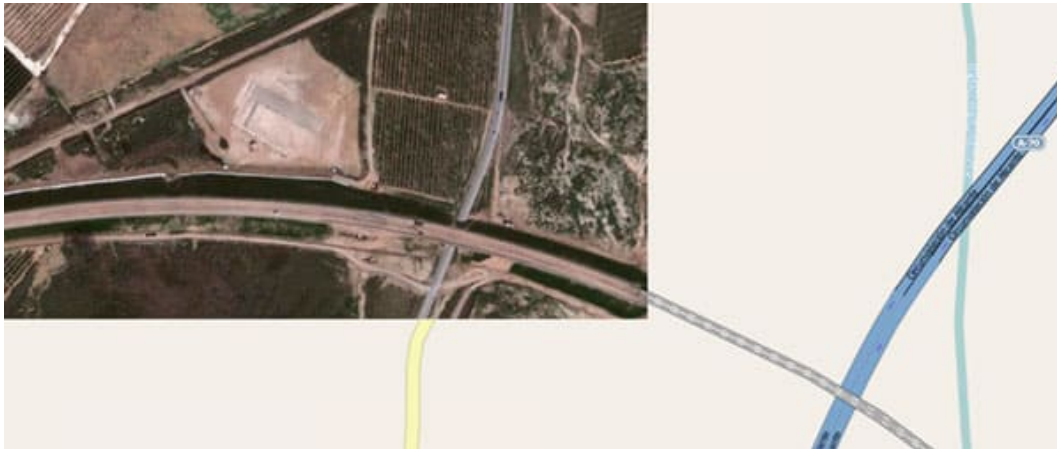
The map widget configures the background map that appears in Veronte screen. In the list shown in the following figure, it is possible to select the map provider from diverse options.



Custom Map Example

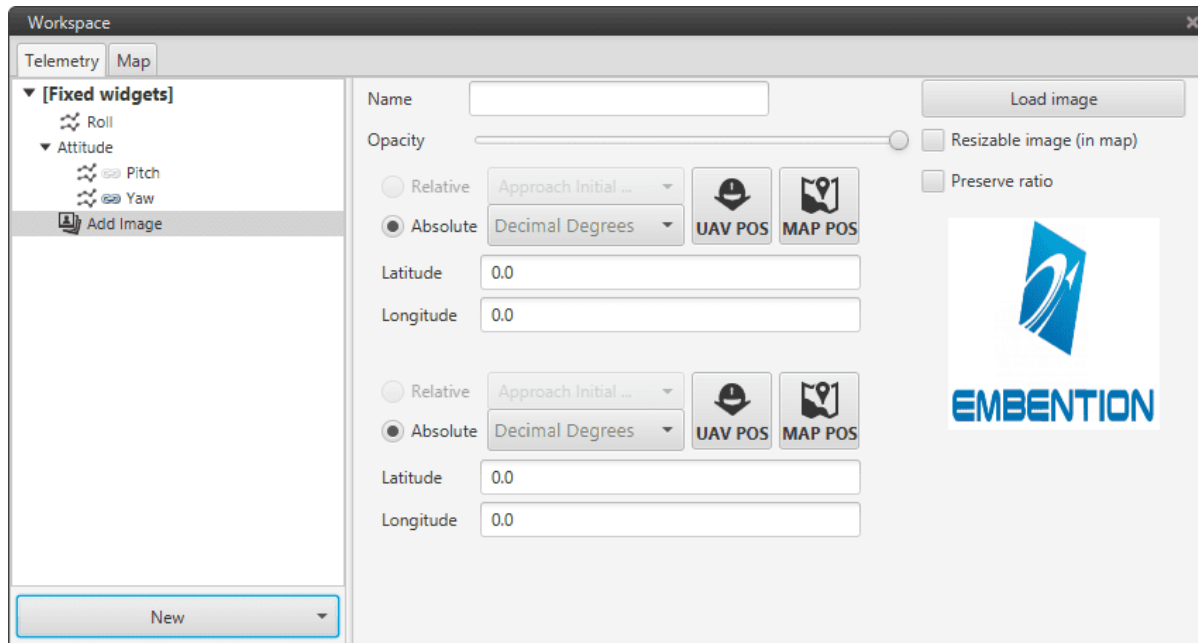
6.2.2 Custom image

Custom maps can be displayed in Veronte Pipe. It allows to include as many images as desired that will be displayed over the map.



Background Image Example

To insert an image within the map click on **New, Add images** and then select the desired image file. Once the image has been loaded, it is possible to configure its position and appearance in the image manager.



Background Image Positioning and Manager

| Item | Description |
|--------------------------|---|
| Name | Define widget's custom name. |
| Opacity | Change widget opacity. |
| Relative | Define image's relative position coordinates. |
| Absolute | Define image's absolute position coordinates (UTM, MGRS, Decimal degrees...). |
| UAV POS | Changes image's position to the current selected UAV position. |
| MAP POS | Allows the user to click on the map and change image's position to the position clicked |
| Load image | Opens a window to select and place an image on the map. |
| Resizable image (in map) | When selected, allows change image dimensions. |
| Preserve ratio | When selected, image dimensions' ratio is conserved. |

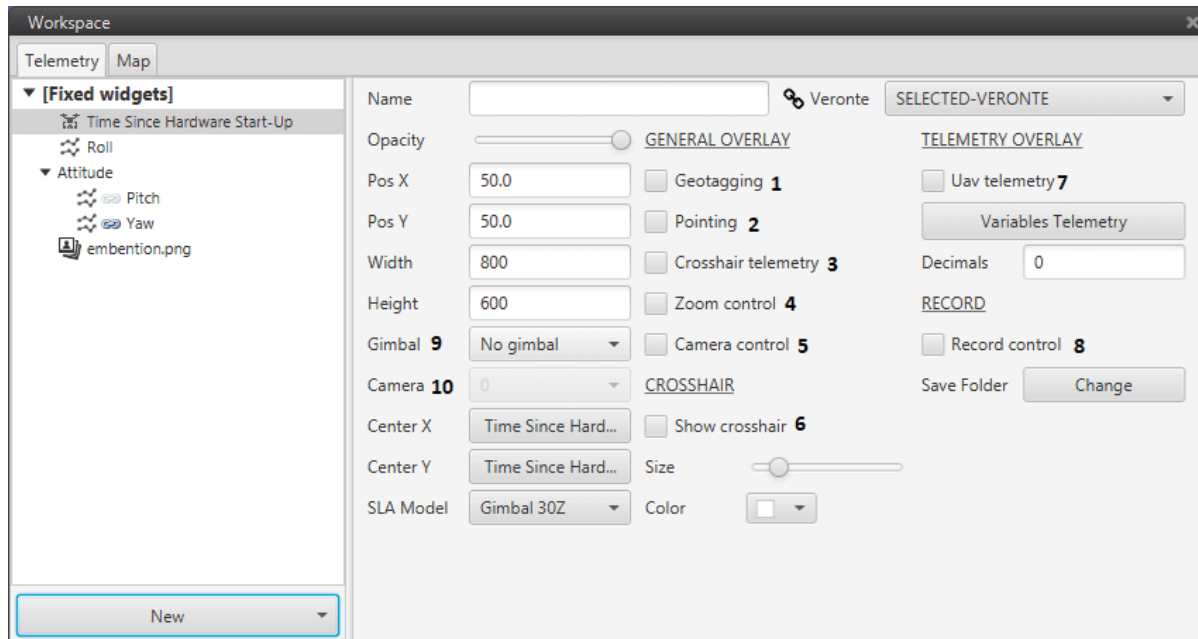
6.3 Widgets

6.3.1 Gimbal

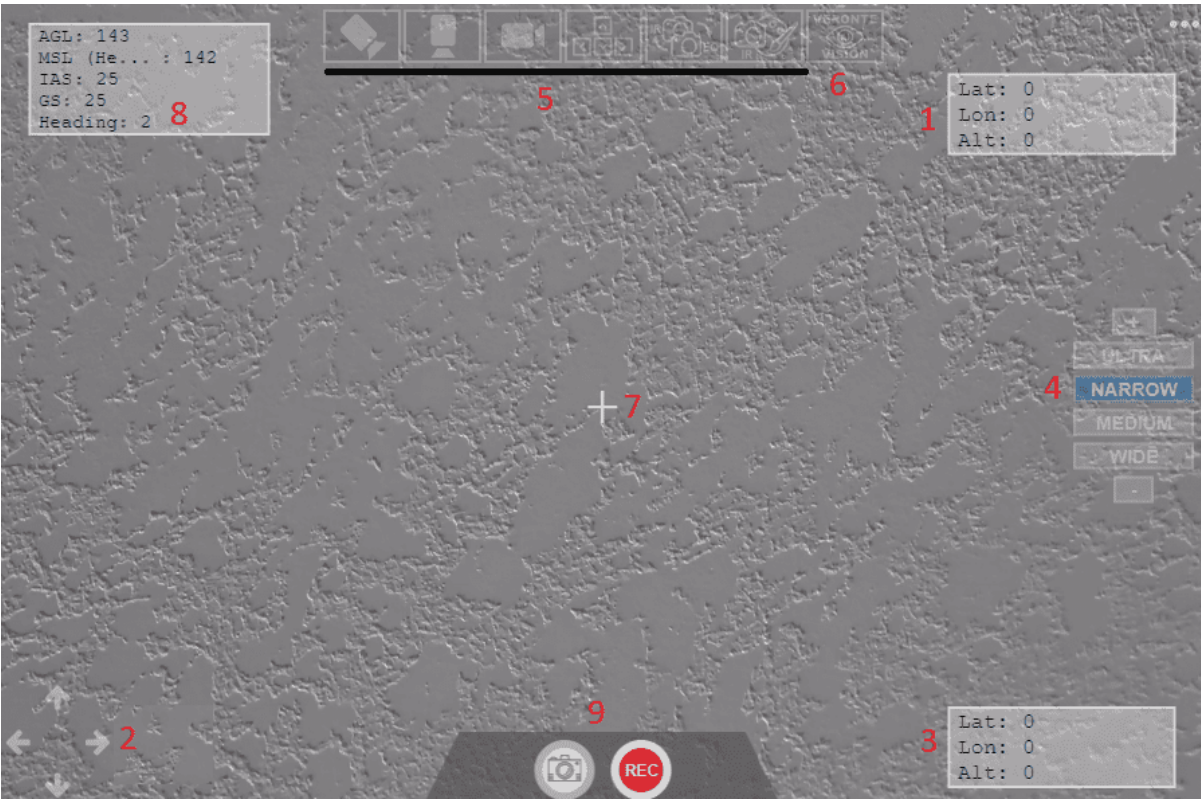
This widget allows the user to display a camera on screen and command a Gimbal if there is any device connected. The options configurable on this widget are the following ones:

1. **Geotagging.** Coordinates and height from mouse point selected. If Veronte Vision is in use, coordinates and height are from the tracking point.
2. **Pointing.** Manually move the camera through a digital stick.
3. **Crosshair.** Shows the coordinates from the picture central pixel (crosshair).
4. **Zoom control.** Top and bottom buttons for gradual zooming (+ and – icons) and 4 predefined zoom levels: wide, medium, narrow and ultra.
5. **Camera control.** Buttons placed on the top part of the widget for:
 - Three predefined positions (45°, Zenithal, Frontal – Navigation)

- Allow keyboard camera control
 - Camera switch EO – IR
 - Colour palette switch for IR camera.
6. **Crosshair.** Show/hide, resize and colour modification.
 7. **Telemetry Overlay.** Enables a screen with telemetry information into Gimbal Widget.
 8. **Record control.** Photography and video recording, choosing in which folder to be stored.
 9. **Gimbal.** Choose controlled Gimbal.
 10. **Camera.** Choose controlled Camera.



Gimbal Widget Configuration Menu



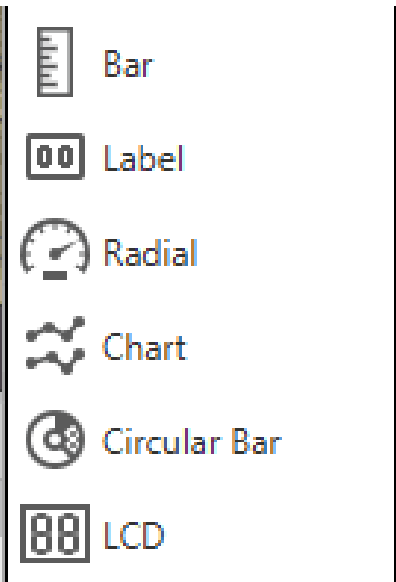
Gimbal Widget Display

6.3.2 Gauge Display

6.3.2.1 Gauge Selection

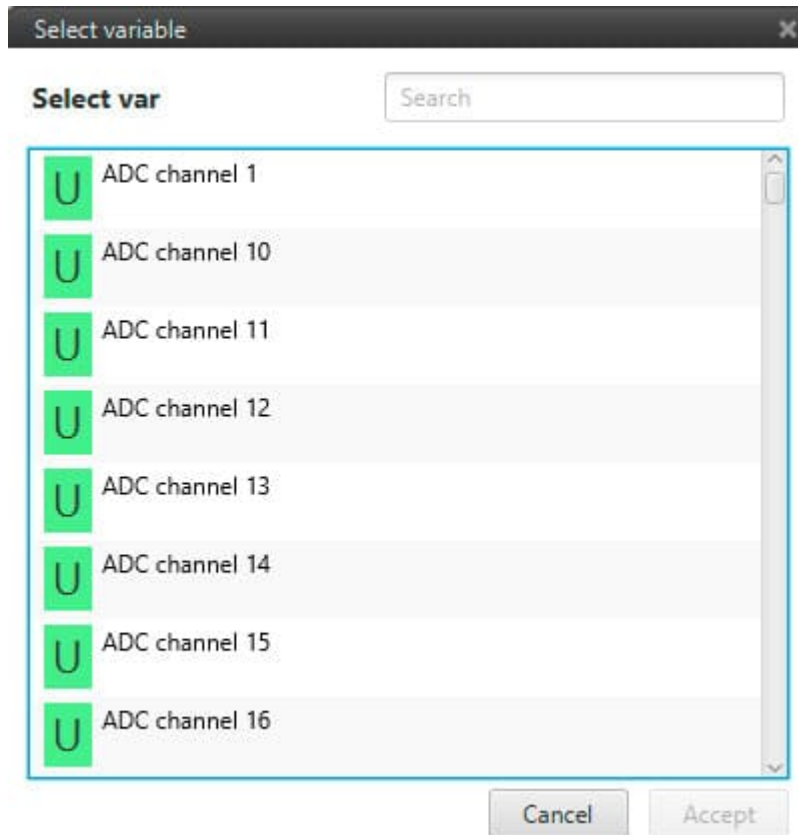
This option is used to display a variable on the workspace. In order to display a new variable:

1. Go to **Workspace** and click on **New**.



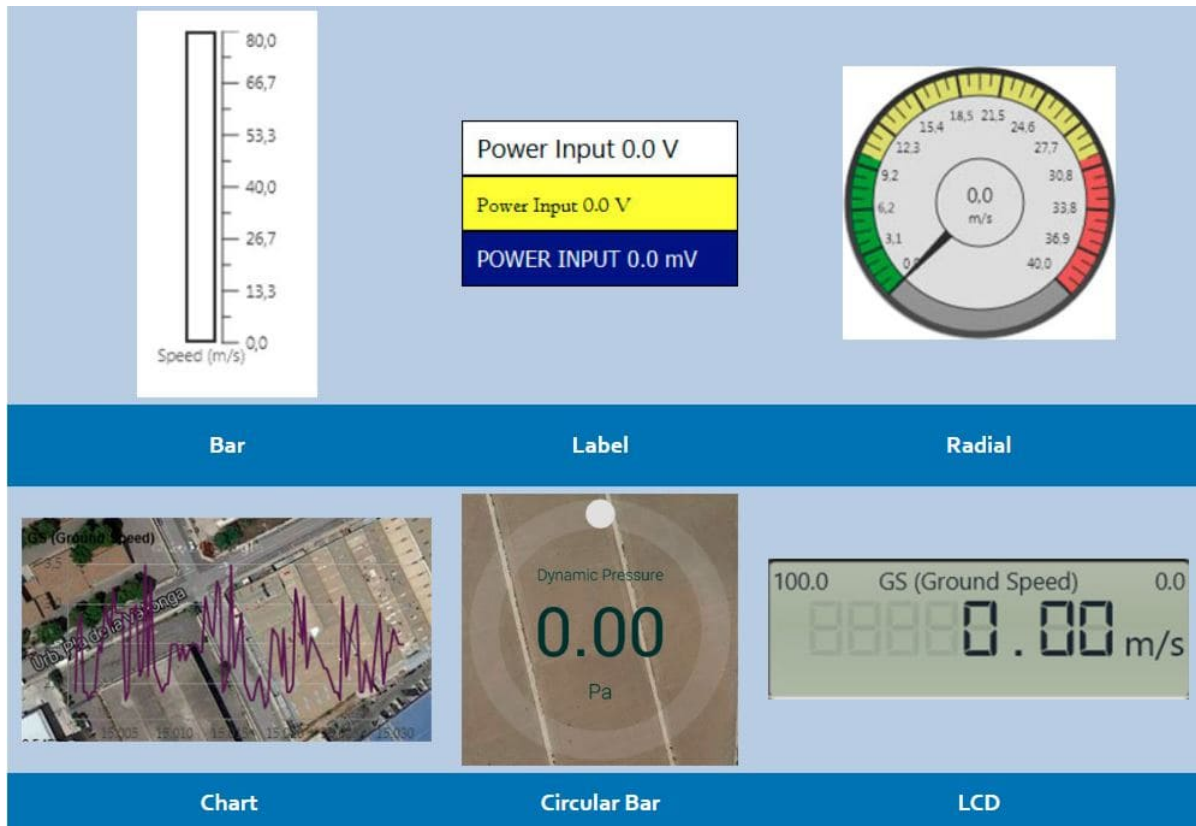
Gauge Selection Menu

- When a gauge is selected, the system will ask for a variable to display. The variable selected can be changed easily by clicking on **Variable**, appearing then the following window.



Variable Selection

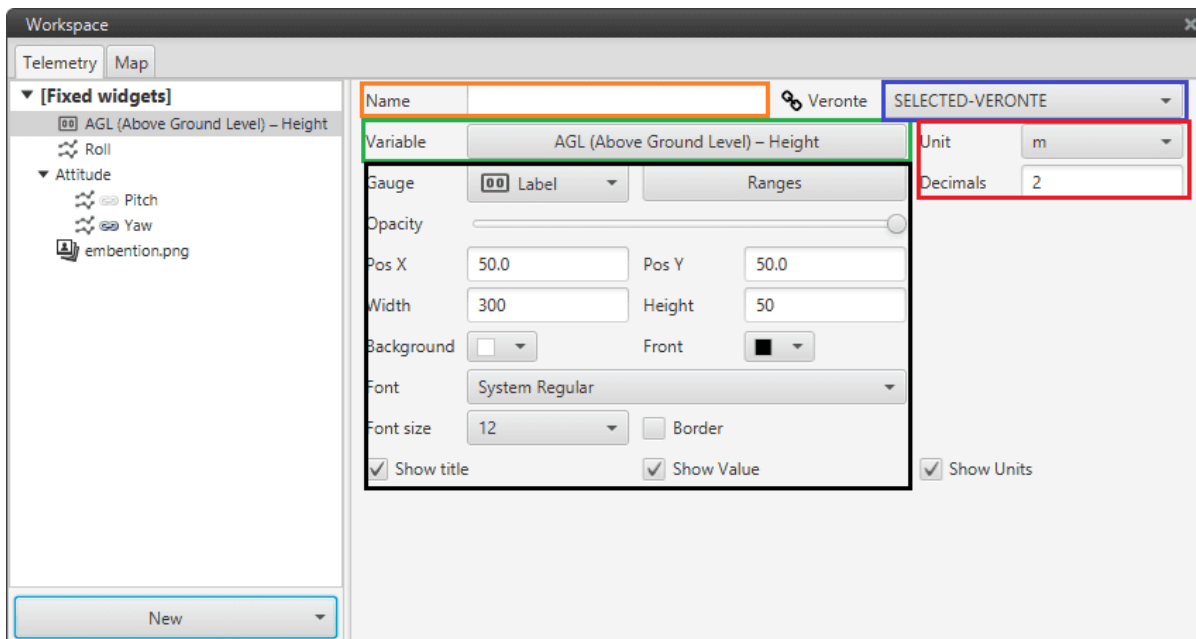
Once the variable to display has been selected, the way it is shown in the interface is also configurable. Here are some of the different gauge types that can be chosen.



Gauge Types

6.3.2.2 Gauge Configuration

Each gauge type can be configured in its panel and some parameters can be changed:




Gauge configuration

- Gauge name. It can be changed by typing a new one. (Orange)
- Selected Veronte. The dropdown menu allows to select another linked Veronte unit. (Blue)
 - If Selected Veronte is the option taken, the gauge will display the information corresponding to the autopilot which has been clicked on the Side Panel.
- Variable units and decimals. (Red)

Warning: Changing the unit of measure of the variable does not imply that the selected Veronte will gather the corresponding data in the selected unit. It only converts the unit.

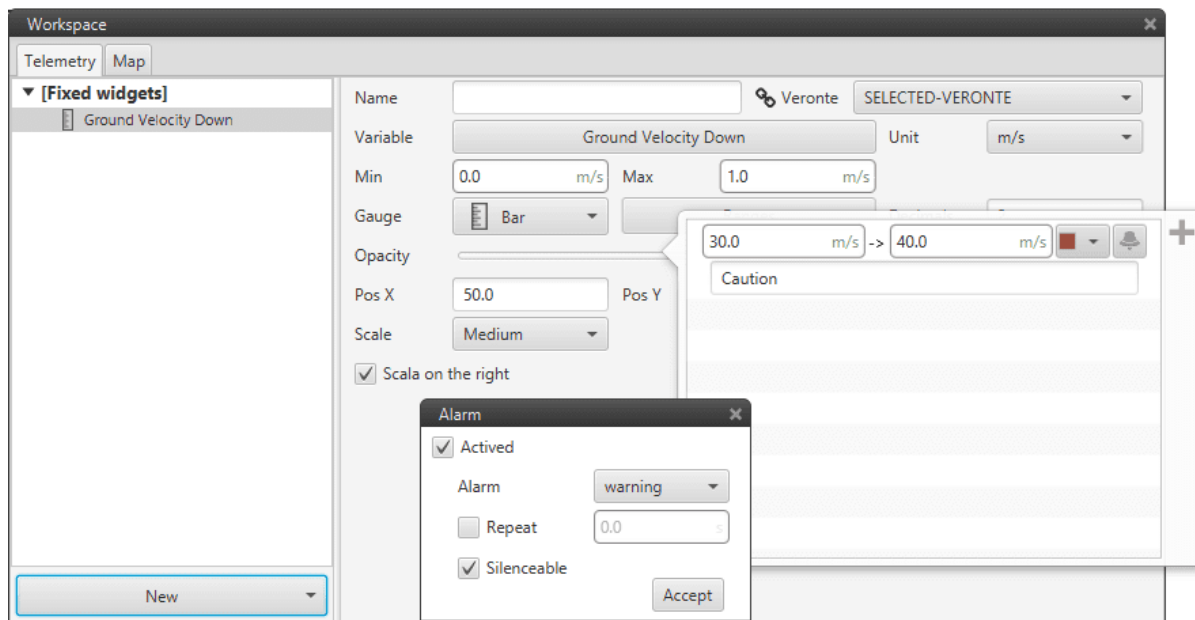
- Displayed variable. (Green)
- Gauge type, dimensions and position. (Black)
 - **Gauge:** Allows the user to change the gauge type (Bar, Label, Radial...)
 - **Ranges:** Allows the user to define an interval by clicking on the '+' button. For this interval several customizations can be made:
 - * Colour: Change the color of the interval which will be shown graphically in the gauge (does not apply to graph gauges)
 - * Alarm: By clicking the bell icon an alarm can be created. For more information about alarm customization please refer to 'Alarm Creation' section.
 - * Text: Add a text to be displayed (only in the gauge types that allow this addition of text)

One of the options that appear for all the kind of gauges is the customization of the colour according to the value of the variable shown.

Warning: Variables will not be collected from the autopilot unless  button is pressed or selected from the *Telemetry* panel.

6.3.2.3 Alarm Creation

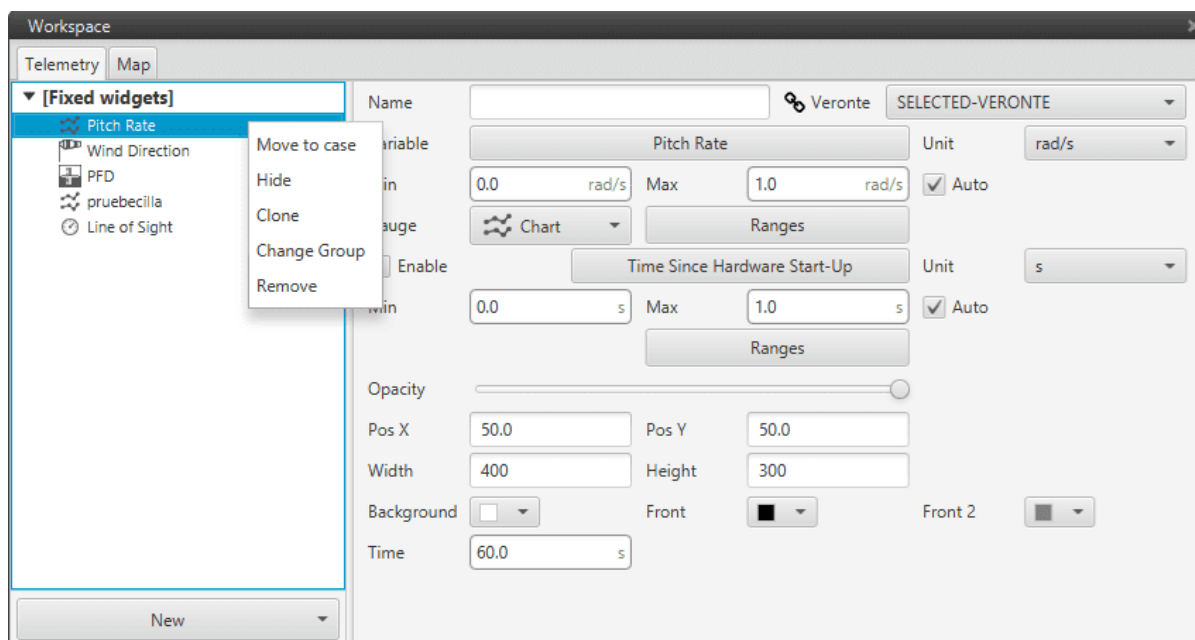
1. Select the option **Range** and a new window will be displayed.
2. In the new window, the user can create different ranges, each one having a certain colour and an associated alarm of the ones configured on **Pipe Configuration-Preferences-General-Alert_Audioclips**.
3. Click on the Alarm icon and configure it.



Gauge – Range

Warning: In the case of displaying a boolean on the workspace (GNSS Navigation Down for example), in the Range menu will appear automatically the two states, each one with a colour (green, red), and also with the possibility of selecting an audio clip for each one of them.

When displaying a chart, it is useful to overlap with transparencies two or more in the same figure to observe their evolution. To do that, Pipe provides the *Change Group* option. Right clicking on the graph and selecting this item displays a window where the group name can be introduced. If two charts are in the same group, their relative position will be kept unchanged (they will move together).

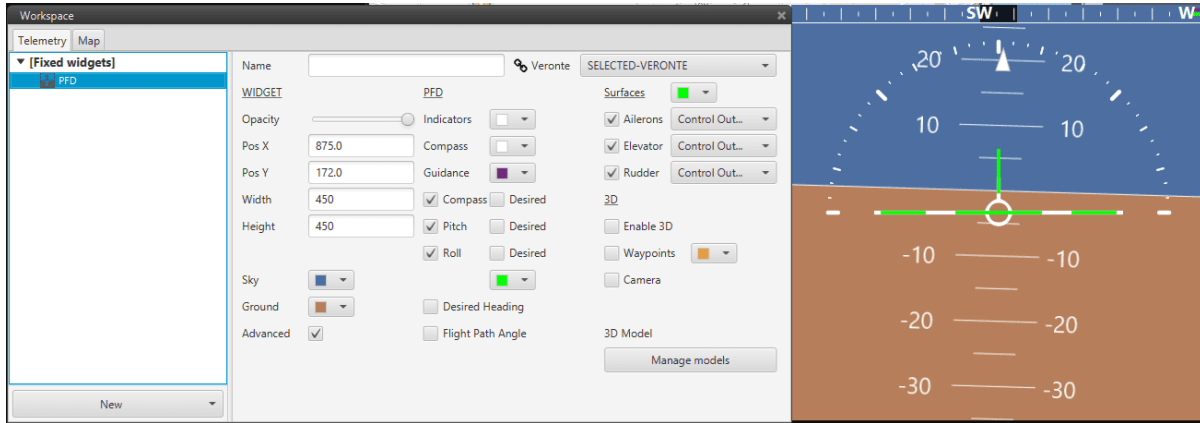


Variable – Group

The option highlighted in red is the option to lock the chart. If all the elements of the group are marked they will move together, if one is not marked it could be moved with respect to the other group elements.

6.3.3 Advanced Primary Flight Display (PFD)

The **primary flight display (PFD)**, also known as “artificial horizon”, represents graphically the attitude of the aircraft (roll, pitch and yaw). This display is highly configurable in colours and size. It is possible to select between the 2D and 3D visualization, and also the surface deflection can be represented.



PFD Configuration

| Item | Description |
|------------------|---|
| Name | Define widget's custom name. |
| Bring to front | Places the current widget above every other display. |
| SELECTED-VERONTE | Dropdown menu to select which Veronte will the widget gather data from. |

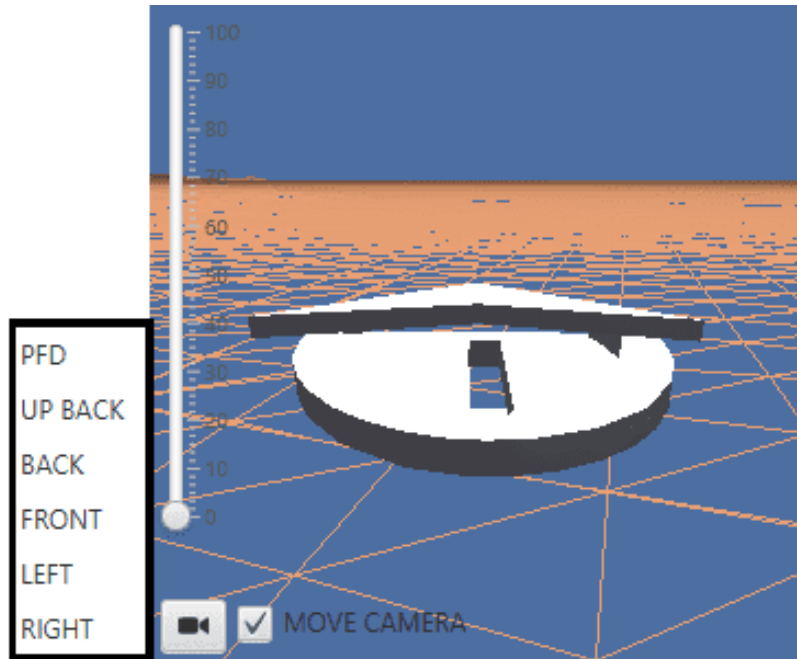
| Widget | |
|--------------|--|
| Opacity | Change widget opacity. |
| Pos X/Y | Define widget's position in the screen's X/Y axis. |
| Width/Height | Define widget's width/height. |
| Sky/Ground | Select sky's/ground's color. |
| Advanced | When marked, displays all the configuration options available. |

| PFD | |
|-----------------------------------|--|
| Indicators/Compass/Guidance | Change display color of the selected feature. |
| Compass/Pitch/Roll | When marked, displays the selected feature. |
| Desired | When marked, it displays the desired Compass/Pitch/Roll, a color change option is available. |
| Desired Heading/Flight Path angle | When marked, displays the selected feature. |

| Surfaces | |
|--------------------------|---|
| Ailerons/Elevator/Rudder | When marked it displays the selected control surface, the control channel can also be selected. |

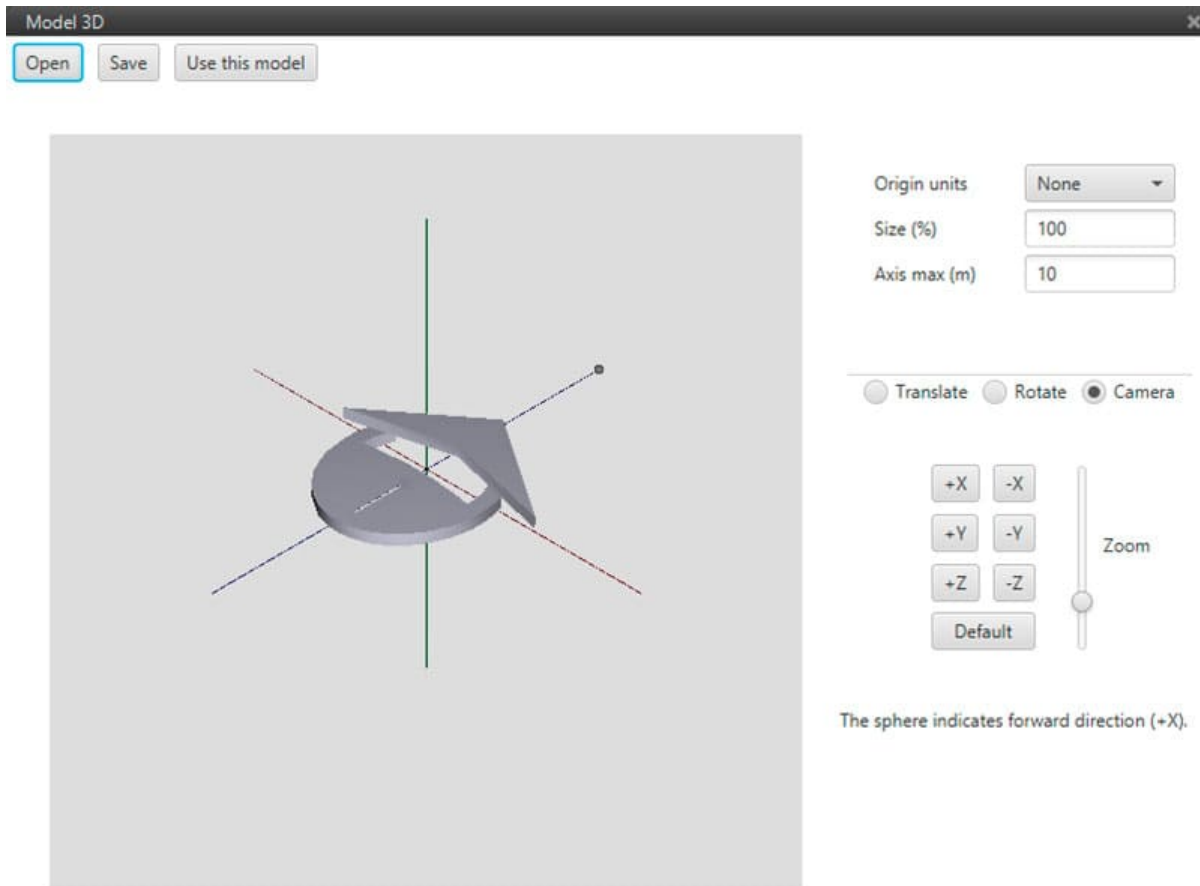
| | |
|-----------|--|
| 3D | |
| Enable 3D | When marked, it enables the 3D feature |
| Waypoints | Displays mission waypoints in the PFD. |
| Camera | Displays camera view if 'Enable 3D' feature is marked. |

- By clicking the camera icon, the camera view can be selected: *PFD* (internal view at the cabin), *UP BACK*, *BACK*, *FRONT*, *LEFT*, *RIGHT*.
- When an external view is selected, it will be displayed a slide to change the distance between the camera and the aircraft, and a *MOVE CAMERA* checkbox, which allows to move around the view by dragging the mouse on the widget.



3D PFD visualization

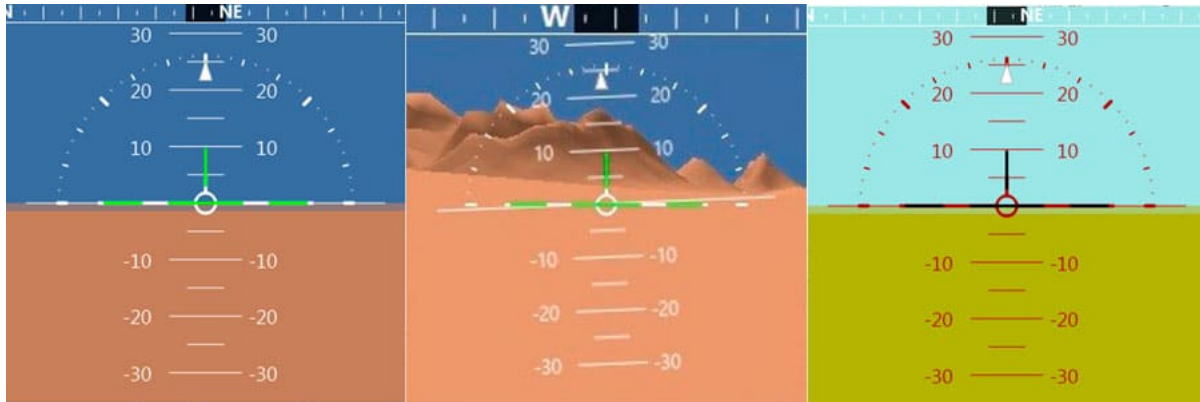
| | |
|---------------|-------------------------------|
| 3D Model | |
| Manage models | Changes the 3D model display. |



3D Model

- **Open:** it allows to select an STL file to be loaded.
- **Save:** save user changes in an STL file.
- **Use this model:** select the current model to be displayed on the widget.
- **Origin units:** user can select origin file units. The model is real-scaled according to the terrain shown on the widget.
- **Size:** percentage scale factor.
- **Axis max:** axis length in meters.
- **Translate/rotate/camera:** translate and rotate modify model orientation. Camera changes the view displayed in Model 3D window. X,Y,Z buttons produce changes according to this axis. Default button resets any modification.

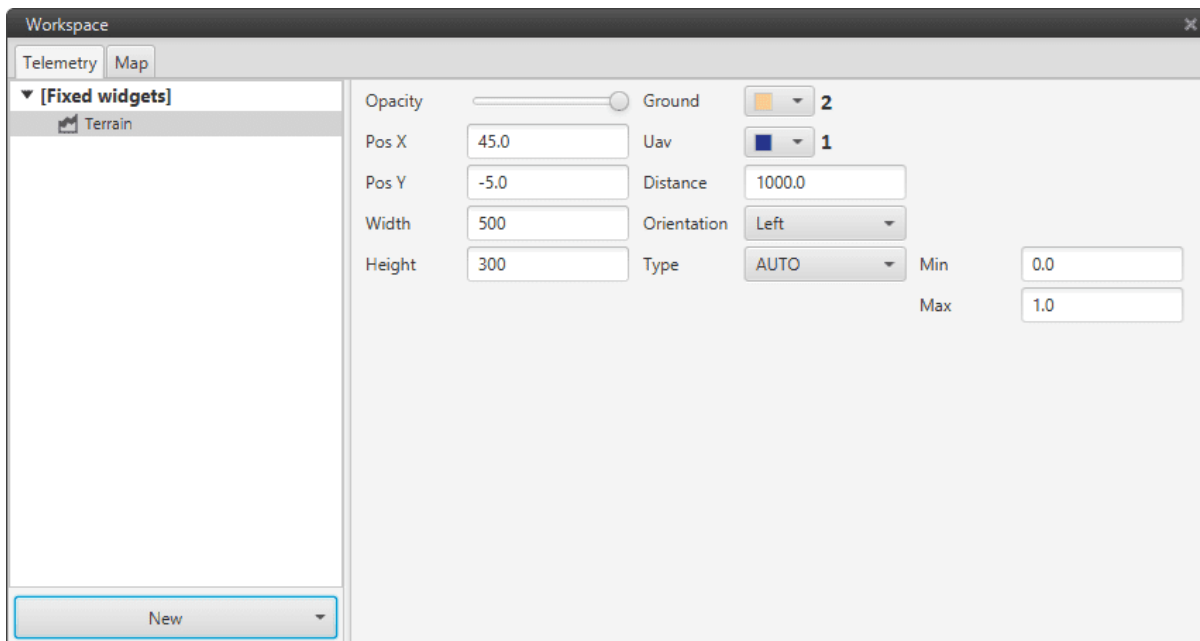
Some PFD display configurations are shown as an example:



PFD Example

6.3.4 Terrain

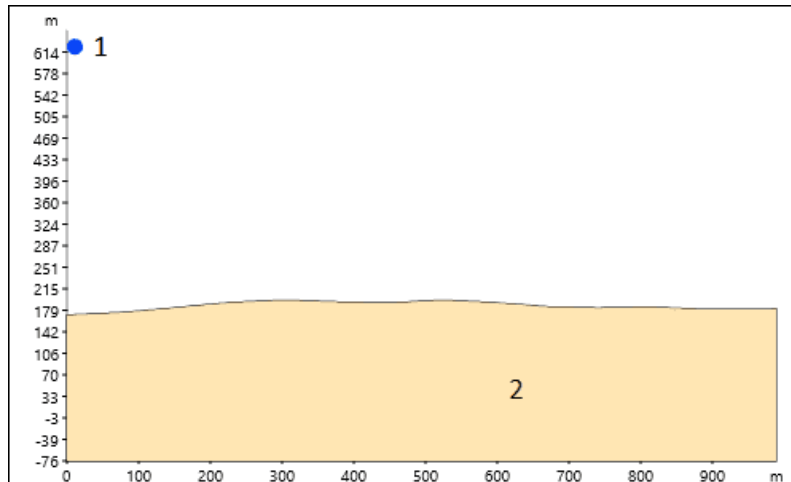
Terrain display shows the terrain profile on the platform direction. Visualization configuration options are as follows:



Terrain Profile Configuration

In the dropdown on the top right corner, the user can select the Veronte to be used in this widget.

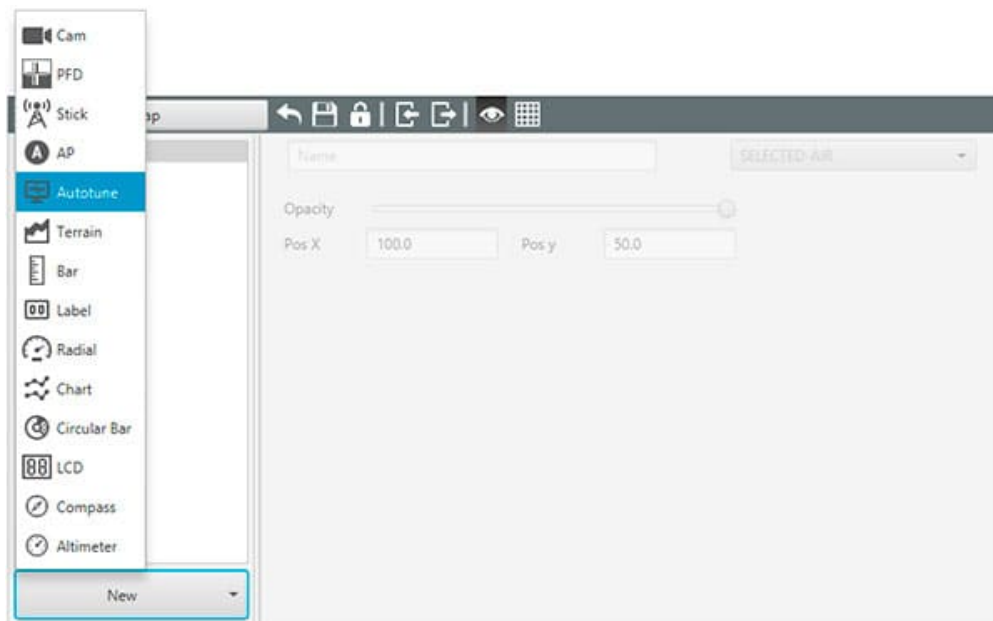
| Item | Description |
|--------------|---|
| Opacity | Change widget opacity. |
| Pos X/Y | Define widget's position in the screen's X/Y axis. |
| Width/Height | Define widget's width/height. |
| Ground | Select ground's color. |
| Uav | Select UAV's color. |
| Distance | Change distance range. |
| Orientation | Define UAV ahead orientation (from left to right or the opposite) |
| Type Min/Max | Select where the program catch the variables. |



Terrain Profile Widget Display

6.3.5 Autotune Tool

In the Workspace panel, it is possible to select the Autotune Tool. It is necessary to configure this window in order to be able to find automatically the gains of the PID controllers.



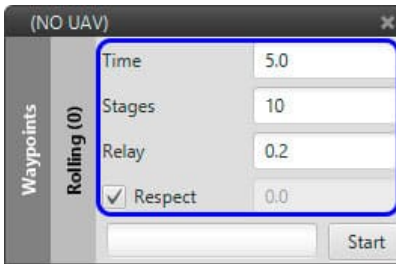
Autotune tool in the Workspace

When the window is opened, it is possible to select the loop for the autotuning.



Autotune loop selecting

This window shows the actual flight phase in the left bar and the control names. For each control, the available loops are showed and they are enumerated from 0 to 2 (from the intern one to the extern one). If some control loop does not have a PID controller configured, it appears with transparency and it can not be autotuned (red).



Autotune parameters

When the loop is selected, the window changes and some parameters (blue) are showed and they can be edited:

- **Time:** It is the period of time in which the Autotune is performed [s].
- **Stages:** The number of stages of the Fast Fourier Transform (a value between 5 and 10 is allowed).
- **Relay:** This is the amplitude of the Relay function (R). The value has to be chosen in accordance with the proportional gain in the PID of the autotuned variable.
- **Respect:** A mean value of the variable has to be selected. If the respect is select, the autotune will start from the last value of the variable. If not, the value can be edited by the user and the Autotune Relay function will start from this value and it will go from -R to R.

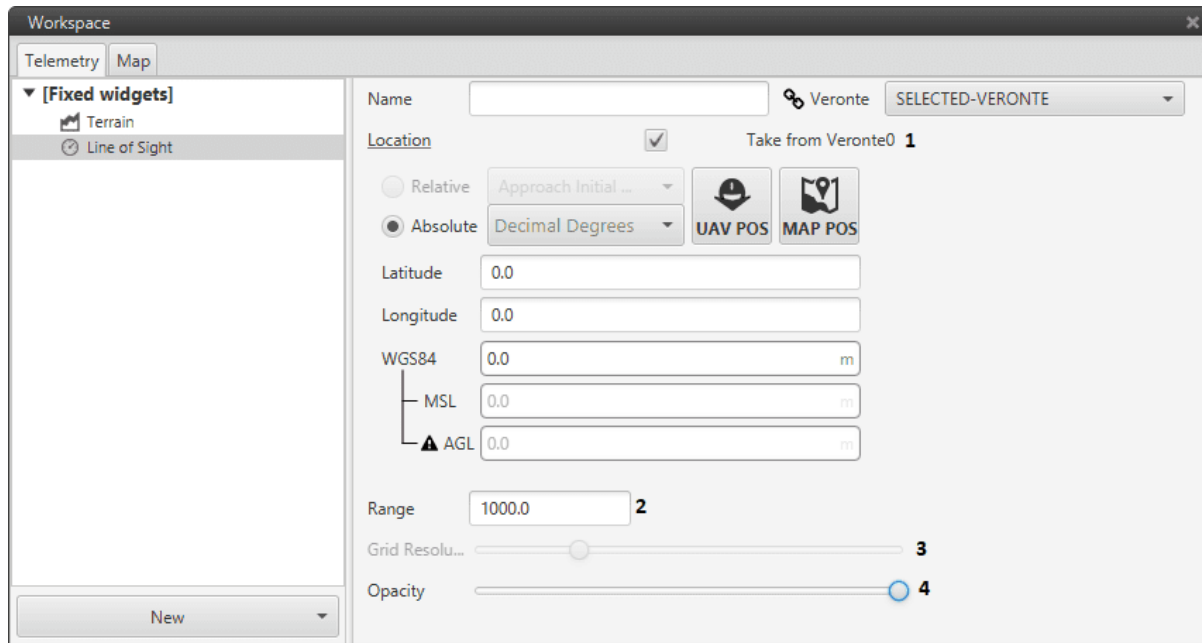
When all parameters are set, it is possible to click on Start and the autotuning process will begin and the blue bar will move until the end of the process. The left bar of the window allows checking the flight Phase and the selected loop.

In the section *Examples - Autotune* of this manual, an autotune example is explained in detail.

6.3.6 Line of Sight

Line of Sight is a widget with which the user can display a grey area around Veronte, showing the Line of Sight it has on every moment at 360°. The configurable parameters are:

1. **Location.** Choose from showing the LOS area from the UAV (checkbox) or any particular location which is fixed (menu).
2. **Range.** The radius of the area covered in meters.
3. **Grid resolution.** A slider that increases/decreases the number of points calculated in the grid. More points equal more resolution but more calculations as well.
4. **Opacity.** A slider that allows the change in LOS area opacity.

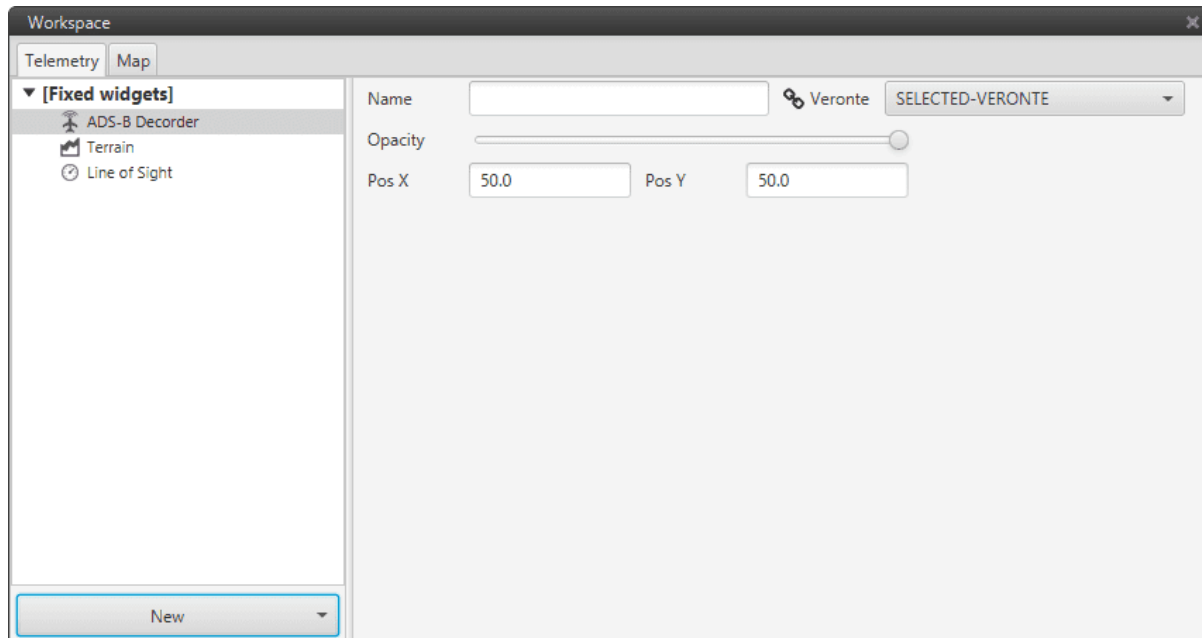


Line of Sight Configuration Menu

Line of Sight Simulation

6.3.7 ADS-B decoder

The ADS-B Decoder widget is a visual information menu page where the user can get all ADS-B information displayed in one window while also having real-time traffics on the screen.



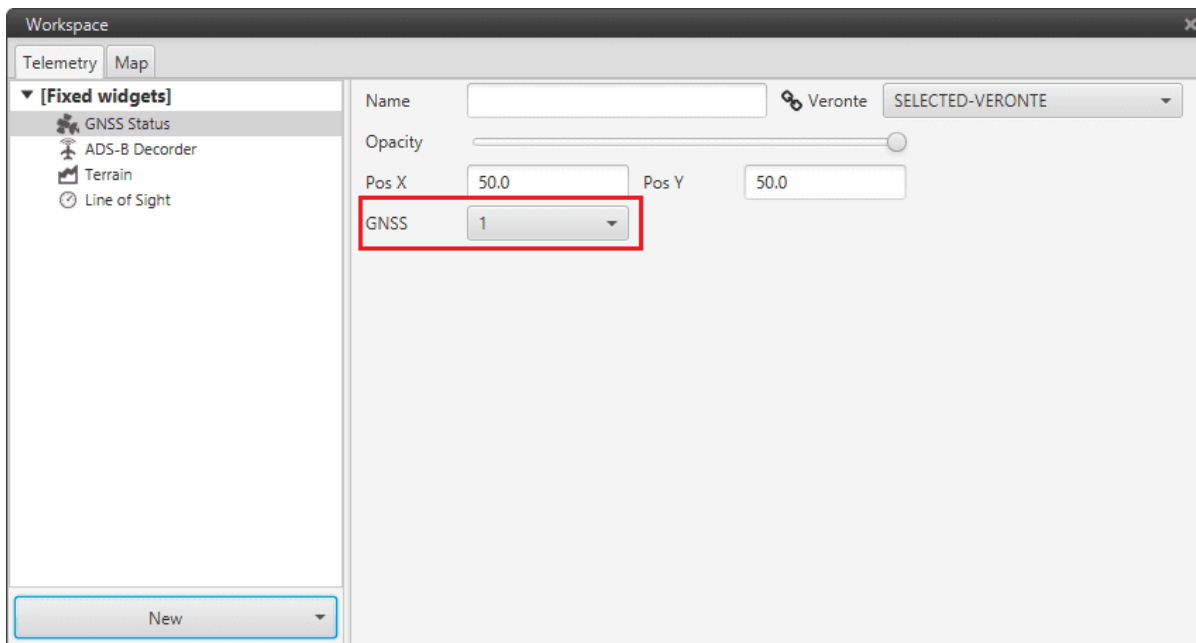
ADS-B Decoder Menu

| Item | Description |
|------------------|---|
| Name | Define widget's custom name. |
| SELECTED-VERONTE | Dropdown menu to select which Veronte will the widget gather data from. |
| Opacity | Change widget opacity. |
| Pos X/Y | Define widget's position in the screen's X/Y axis. |

ADS-B Real Session

6.3.8 GNSS status

The GNSS Status widget is a visual information menu where the user can get all information from the GNSS configured in one same window. It can be chosen which GNSS is displayed:



GNSS Status Configuration Menu

| Item | Description |
|------------------|---|
| Name | Define widget's custom name. |
| SELECTED-VERONTE | Dropdown menu to select which Veronte will the widget gather data from. |
| Opacity | Change widget opacity. |
| Pos X/Y | Define widget's position in the screen's X/Y axis. |
| GNSS | Select GNSS to be displayed. |

The screenshot shows a 'GNSS Status' window with a satellite icon and three status buttons: 'GNSS 2' (green), 'SURVEY IN' (grey), and 'RTK' (grey). The window is divided into three sections: POSITION, VELOCITY, and STATUS.

| POSITION | | VELOCITY | |
|----------------|-------------------------------|----------|------------------|
| Latitude | 0.66915107 rad $[-\pi, \pi]$ | East | 0.39600003 m/s |
| Longitude | -0.00995665 rad $[-\pi, \pi]$ | North | -0.065000005 m/s |
| Altitude (MSL) | 125.05601 m | Down | 0.062000003 m/s |

| STATUS | |
|--------------|---------------|
| Accuracy | 63.499073 m |
| H. Accuracy | 51.786003 m |
| V. Accuracy | 36.747 m |
| S. Accuracy | 1.3610001 m/s |
| #SVs Used | 4 -- |
| Time of Week | 487146.53 s |
| PDOP | 14.179999 -- |

A 'DGNSS' button is located at the bottom right of the window.

GNSS Status Display

The information shown in the widget is as follows:

- **GNSS 1/2.** Navigation bit status (red/green).
- **Survey In.** RTK conditions being accomplished if green. Grey if finished or not achieved.
- **RTK.** Becomes green when SurveyIn bit goes grey (off) and RTK is enabled. Grey otherwise.
- **DGNSS.** Green when Differential Positioning is enabled (GNSS Compass). Grey if not.

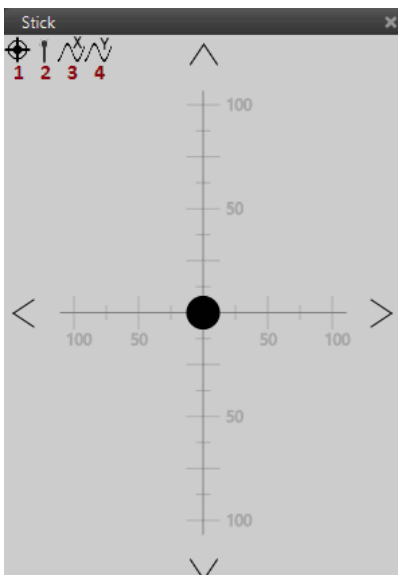
6.3.9 Telecommand

6.3.9.1 Stick

Virtual sticks are created to simulate a radio controller that controls the channels of the aircraft directly from the computer.

In the stick display, there are four icons and their function are:

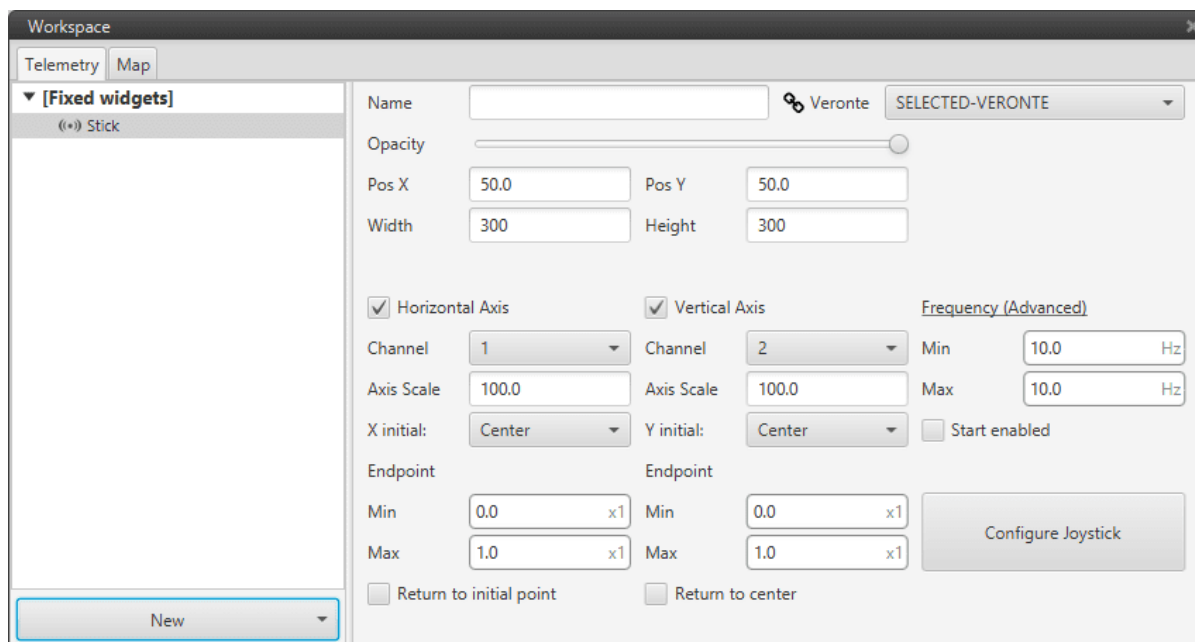
1. Return the stick to the centre
2. Activate/Deactivate the virtual stick
3. Move the stick in the X direction in a sinusoidal way
4. Move the stick in the Y direction in a sinusoidal way



Stick Display Configuration

Last two are normally used in order to test the stick or servos.

Warning: If the Stick is not active it will make no effect on the system. Activate it by clicking on the second icon (some waves will appear near the icon and the stick will be activated). In order to deactivate it, only click on the antenna icon one more time.



Stick Configuration Menu

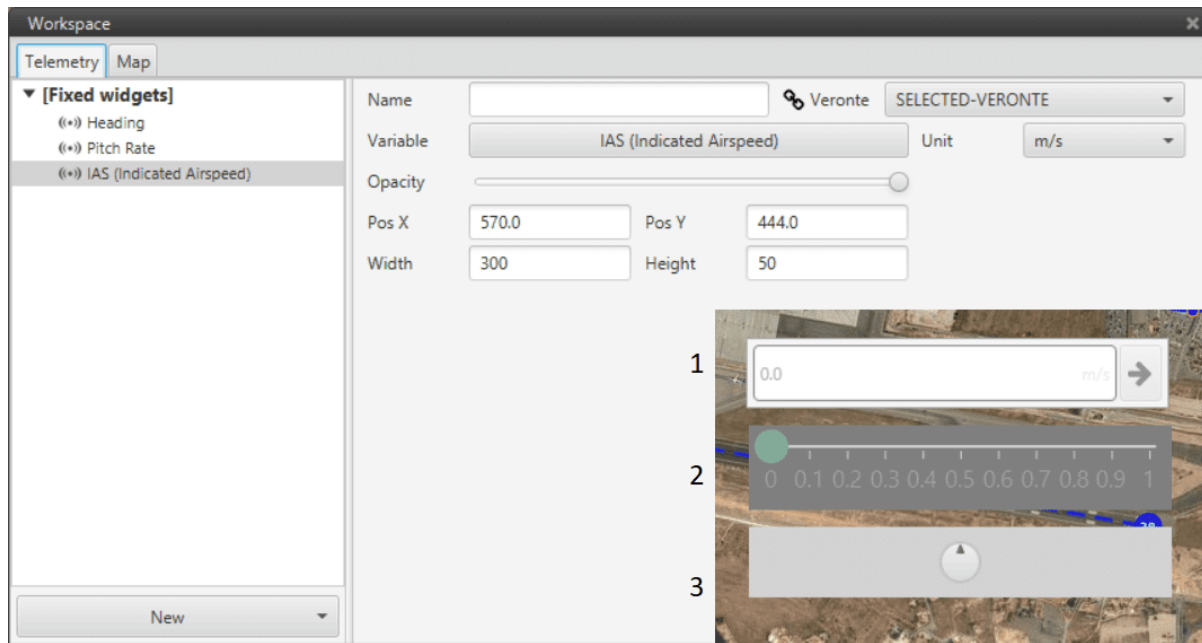
In the dropdown on the top right corner, the user can select the Veronte to be used in this widget.

| Item | Description |
|--------------------|--|
| Name | Define widget's custom name. |
| Opacity | Change widget opacity. |
| Pos X/Y | Define widget's position in the screen's X/Y axis. |
| Width/Height | Define widget's width/height |
| Channel | Select which channel is controlled by each axis. |
| Axis Scale | Scale to show in the axis of the stick. |
| Endpoint Min & Max | Establish the minimum and maximum values reached by the stick. |
| Frequency Min | Sets the minimum quantity of messages sent when there is no movement. |
| Frequency Max | Sets the maximum quantity of messages that can be sent. |
| Return center | When it is selected the stick automatically returns to middle position on stick release. |
| Configure Joystick | Configure external USB joystick for camera control. |

6.3.9.2 Change variable

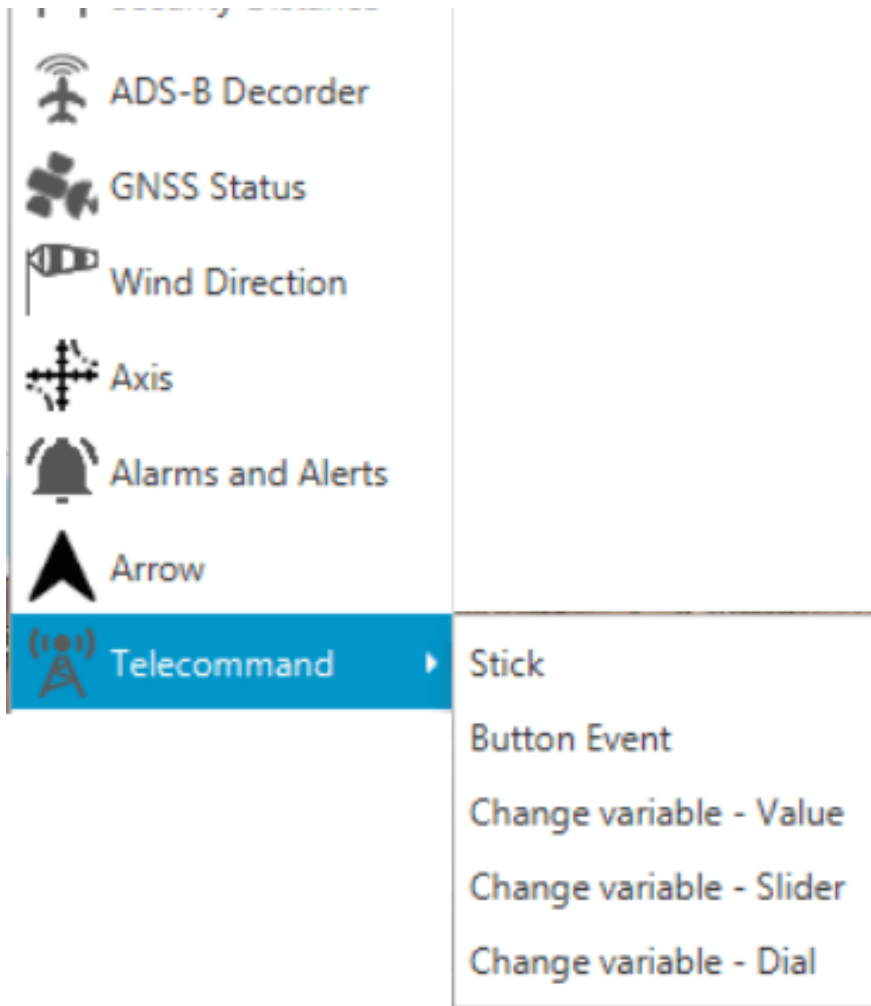
The user can choose certain variable and change its value from Workspace during a flight. There are 3 ways or widgets to implement this action:

1. Value
2. Slider
3. Dial



Change variable

Veronte pipe allows the user modify some variables during a flight to test some parameters or to simulate a stick controller. Click on *New* in the widget menu and then on *Telecommand* and some options appear.



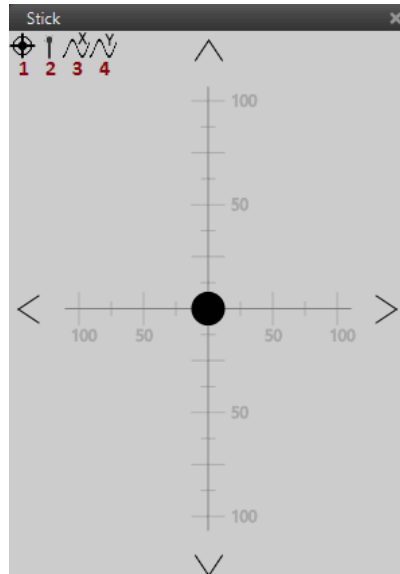
Stick Display Configuration

6.3.10 Stick

Virtual sticks are created to simulate a radio controller that controls the channels of the aircraft directly from the computer.

In the stick display, there are four icons and their function are:

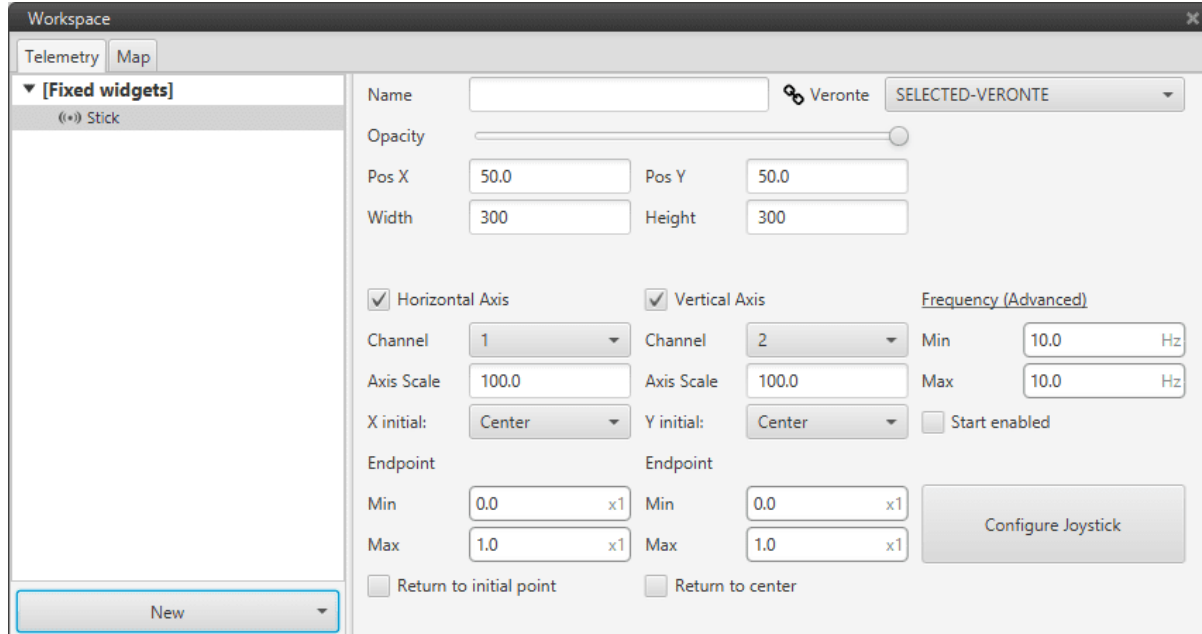
1. Return the stick to the centre
2. Activate/Deactivate the virtual stick
3. Move the stick in the X direction in a sinusoidal way
4. Move the stick in the Y direction in a sinusoidal way



Stick Display Configuration

Last two are normally used in order to test the stick or servos.

Warning: If the Stick is not active it will make no effect on the system. Activate it by clicking on the second icon (some waves will appear near the icon and the stick will be activated). In order to deactivate it, only click on the antenna icon one more time.



Stick Configuration Menu

In the dropdown on the top right corner, the user can select the Veronte to be used in this widget.



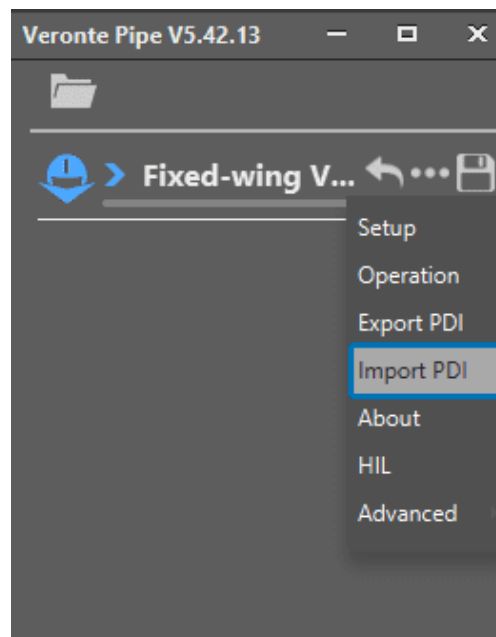
- **User:** Manage user preferences.
 - **Cloud:** Provides access to the Cloud options.
 - **Workspace:** Select the way flight information is displayed.
 - **Mission:** Create and edit missions.
 - **Log:** View operation data log and introduce custom events.
 - **Post Flight:** Tools for recorded data analysis.
 - **Preferences:** Configure Pipe and Veronte autopilot.
 - **License:** Manage license preferences.
 - **Manual:** Shows help information available.
3. **Veronte Panel:** Veronte information and telecommand buttons.
 4. **Veronte Position:** Veronte location on the map.
 5. **Mission:** Defined mission on Veronte.
 6. **Telemetry:** Configurable drag & drop flight information displays.
 7. **Side Panel:** Shows linked Veronte information.

VERONTE CONFIGURATION


7.1 File Management

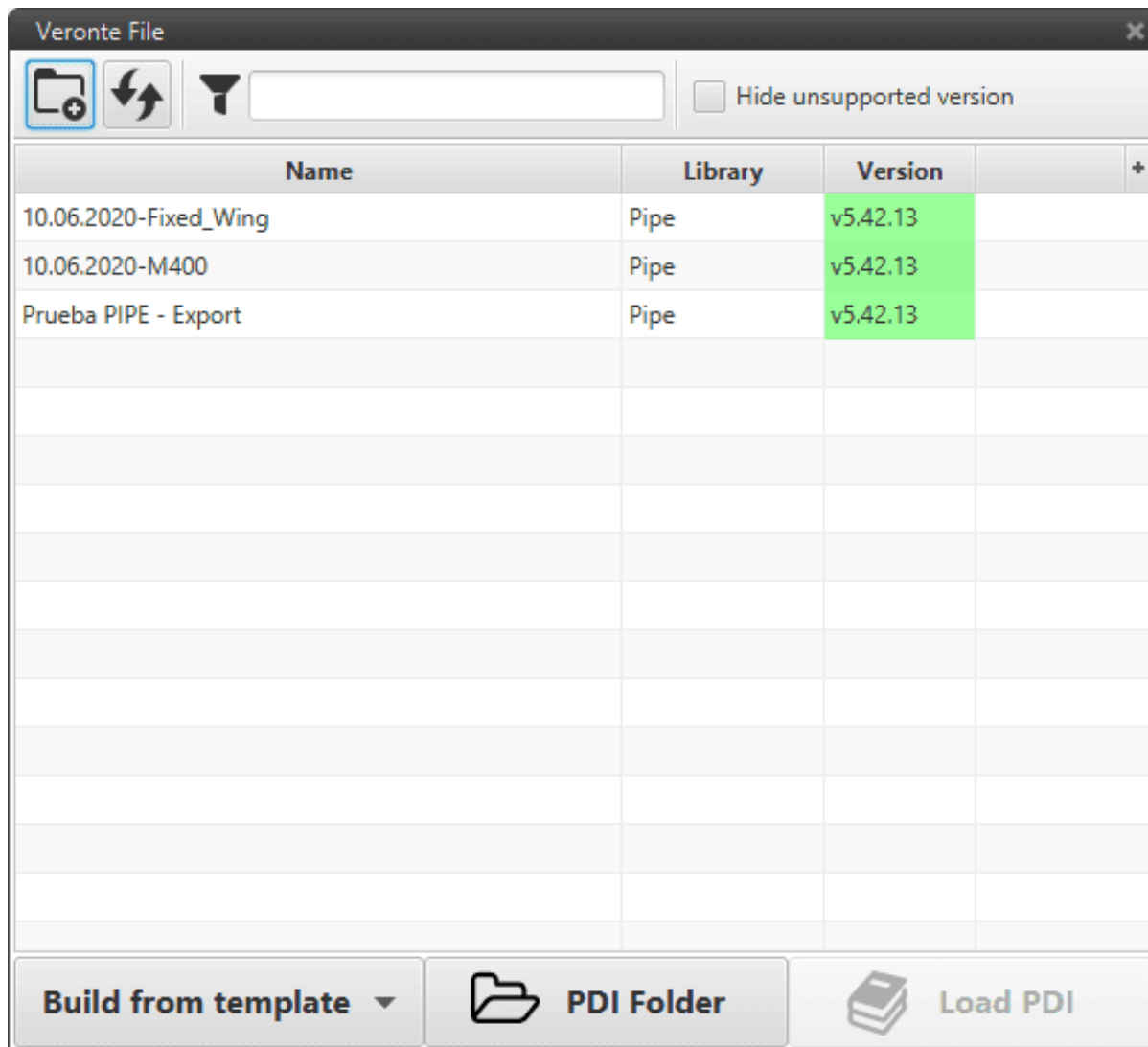
7.1.1 Import Configurations

In this section is explained the process to load configuration files into a Veronte Unit.



Import Configuration

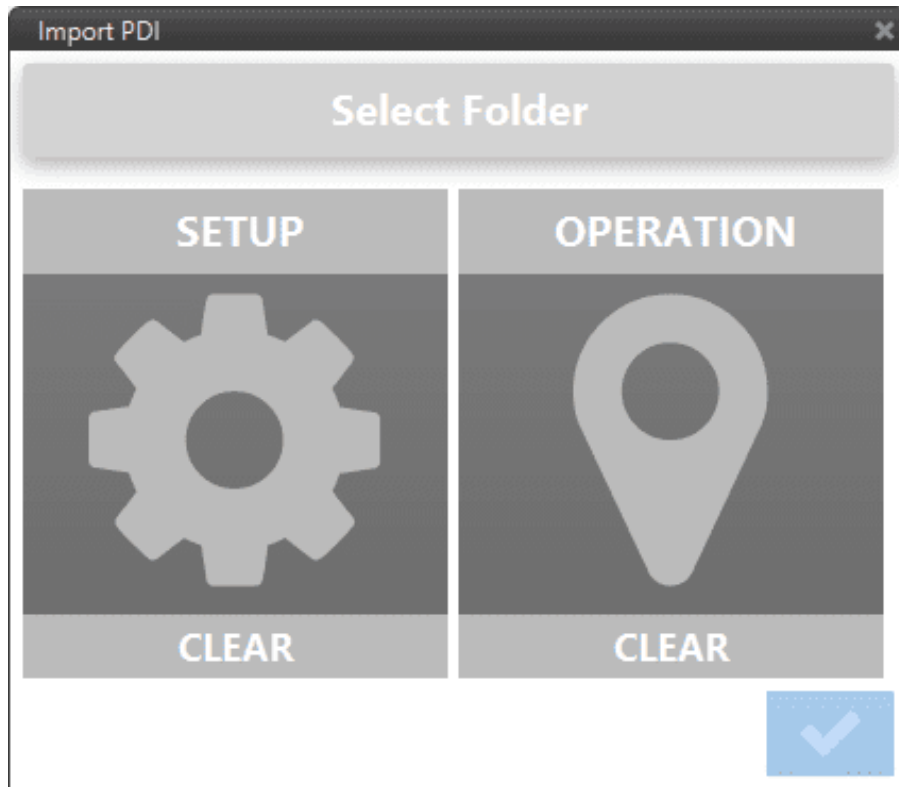
First, power the autopilot and connect it to Veronte Pipe. In the side panel, click on  and then select **Import PDI**. The **Import Configuration** option displays a window used to select the file to be uploaded.



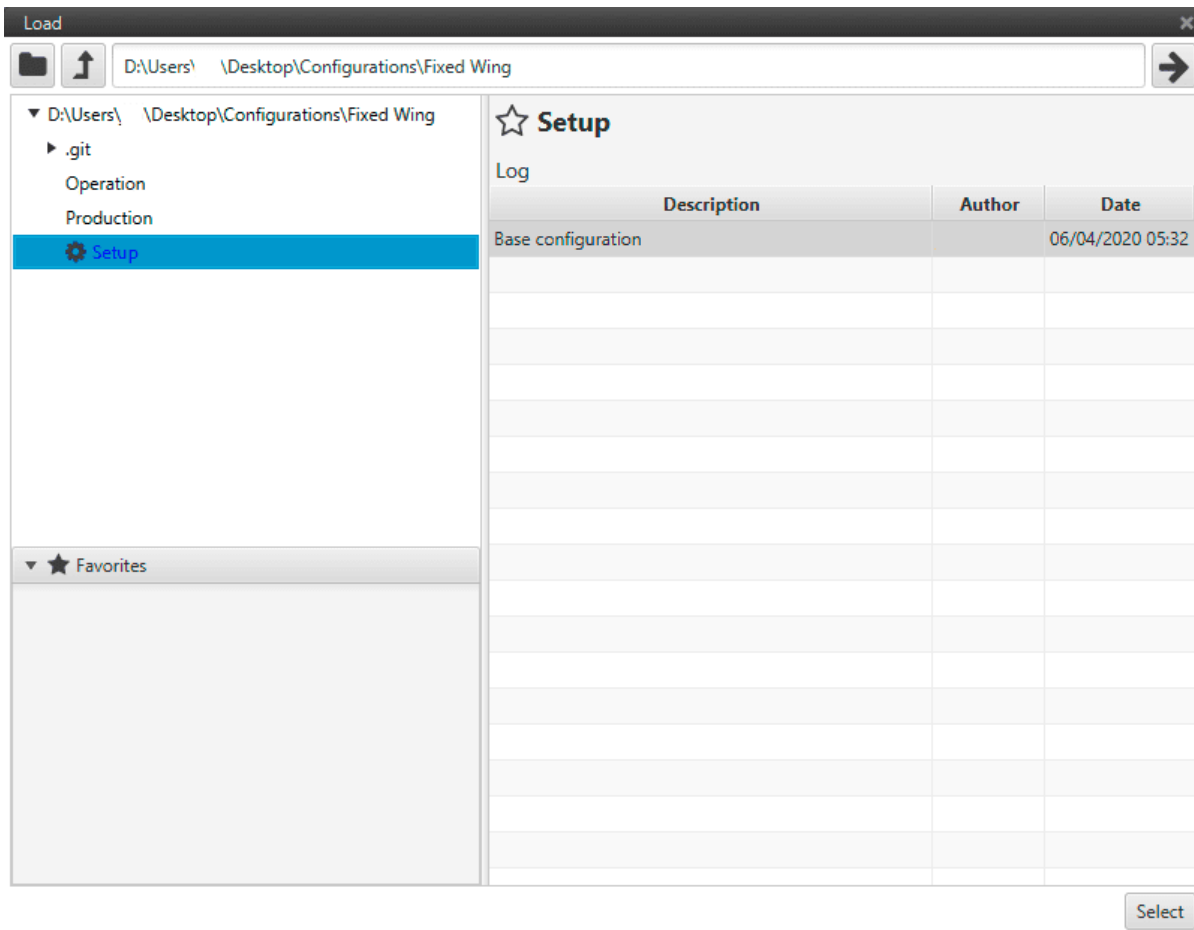
Import Configuration File Menu

There are three options within this window to select a configuration file:

- **Build from template:** By selecting this option the user can import a configuration file from a set of default configurations that can be used as a starting point. There are templates for multicopters, fixed wings, Ground station among others.
- **PDI folder:** By selecting PDI The user will be able to select the locations of each folder and load them.




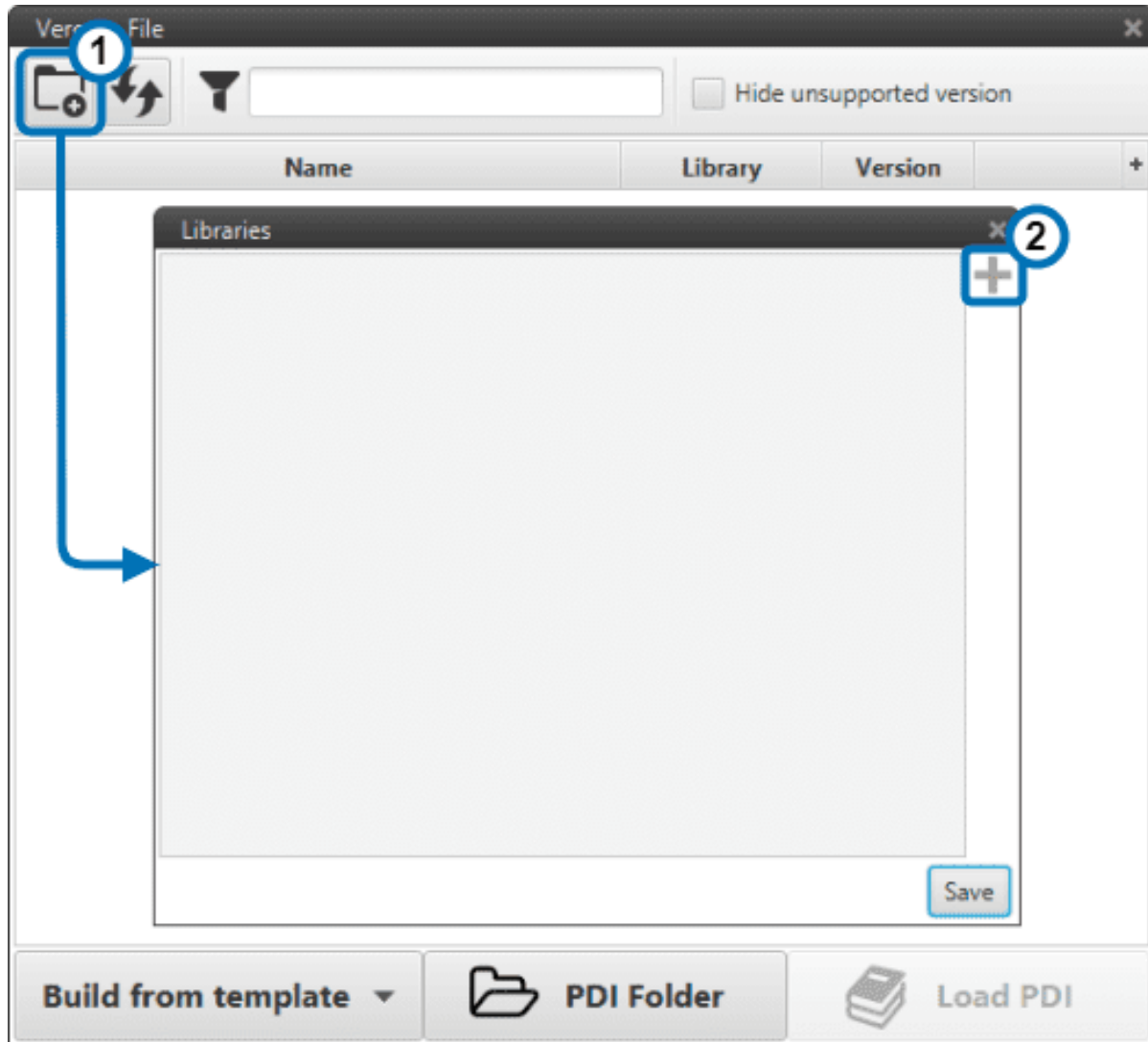
Import a Configuration File



Import a Configuration File


- **Load PDI:** Loads a configuration from the library.

In order to include a folder as a part of the library select **Manage Libraries** icon . By including a folder to the library, any configuration files in that folder will be displayed.



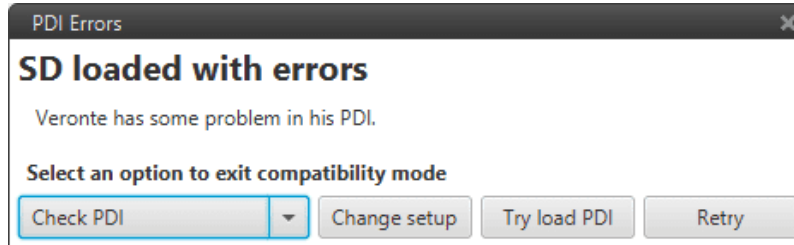
Library Management Menu

Warning: Each configuration in the library has a Version. It is only possible to load configuration files whose version is the same or lower than the one of the software embedded on the autopilot. As shown in the pictures above, colours are the way to show the compatibility: **Green** means fully compatible, **Yellow** means migration/PN adaptation and **Red** means not compatible.

Warning: Press the **Save** button in the right Side Panel,  , to load the configuration on the autopilot. Otherwise, it will be only loaded to Pipe software.

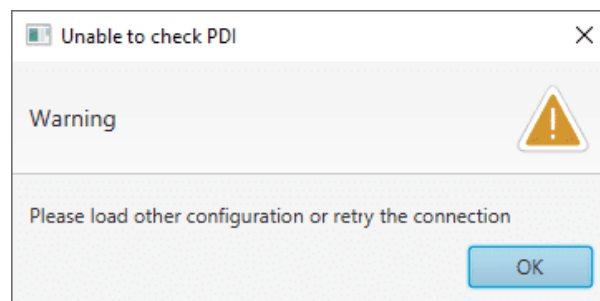
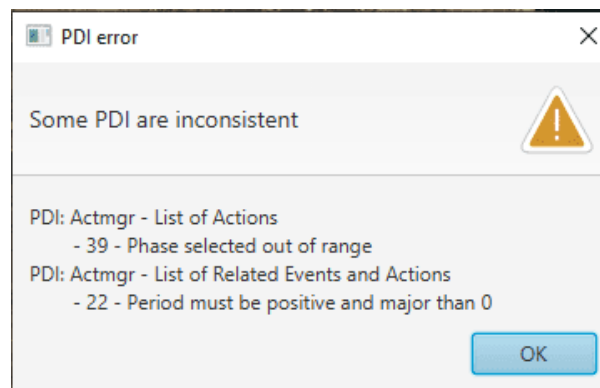
7.1.1.1 Check PDI

After Importing PDI files or migrating a configuration to a newer version, Veronte might enter in “SD loaded with error” state, which is a similar state to *Maintenance Mode*. When clicking on the unit affected it can be seen the following window appearing.



PDI Error Window

The options provided to the user are “Check PDI”, which generates an error display window, “Change Setup”, explained below, “Try load PDI” and “Retry”. When selecting “Check PDI”, the system will try to report the errors. It may or may not find them:



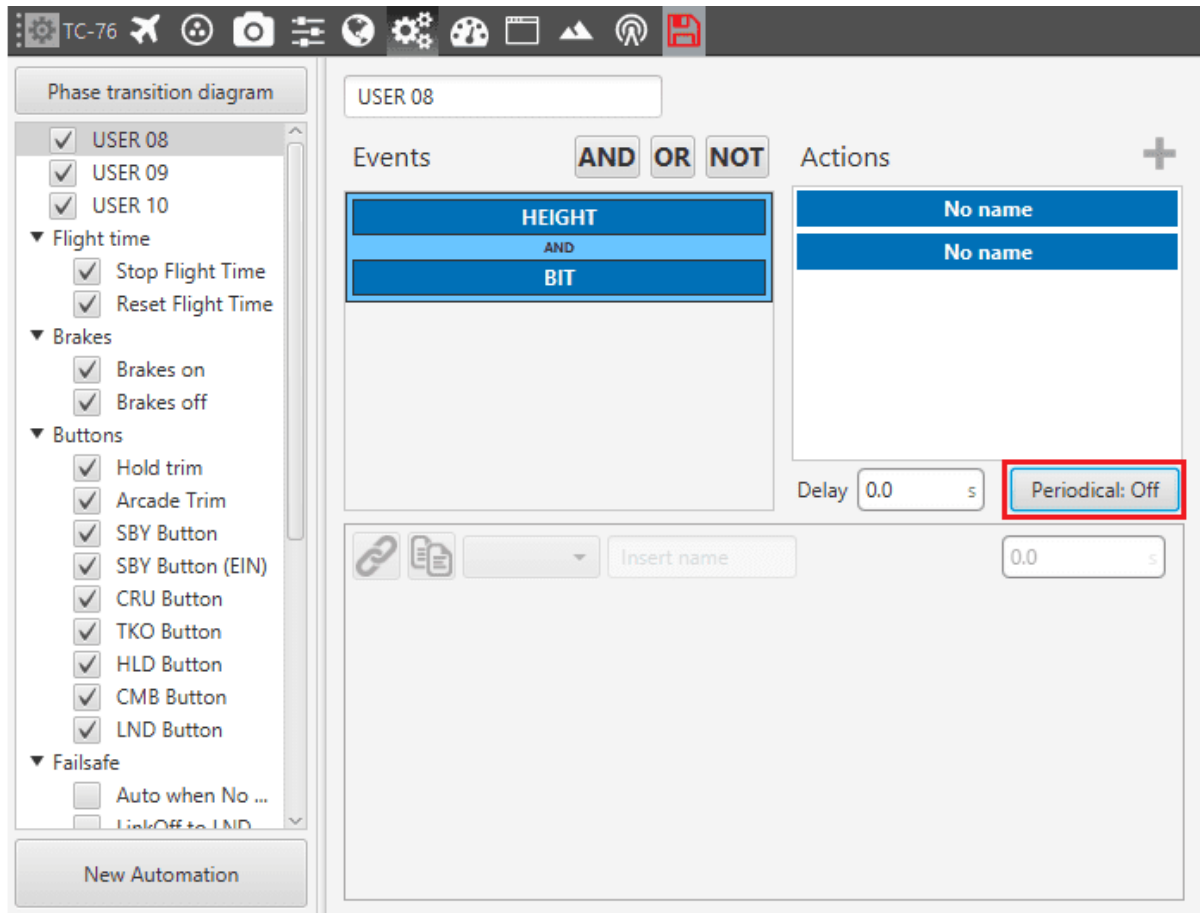
Check PDI windows

Now the user must find and correct the errors shown (or not) in the list displayed. In order to do that, click on “Try load PDI” in the PDI Error window.



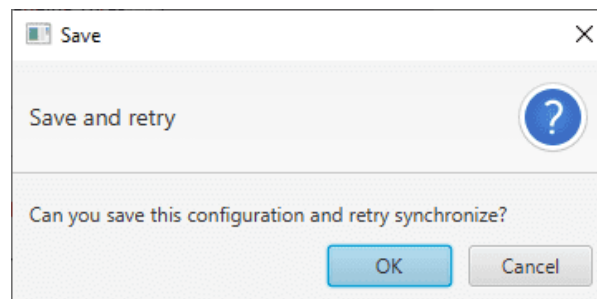
Check PDI setup

In this particular example, one of the errors reported was a Period in Events/Action (Automations tab) which has been disabled.



Correction done in PDI Check

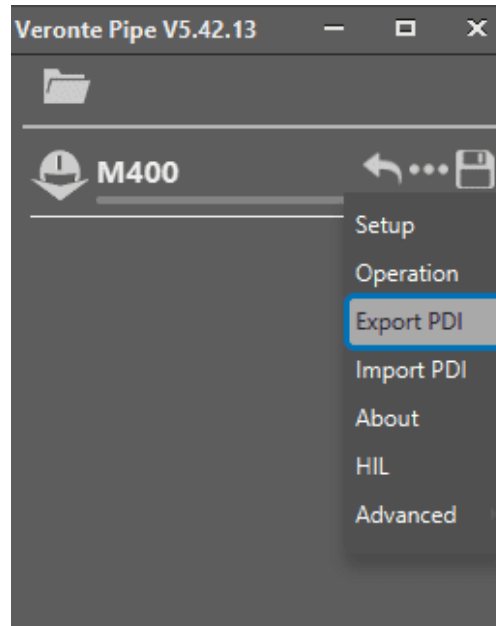
Once all reported errors have been solved, press on the Red Save button on the Setup toolbar to finish the PDI Check.




Save Setup after PDI Check

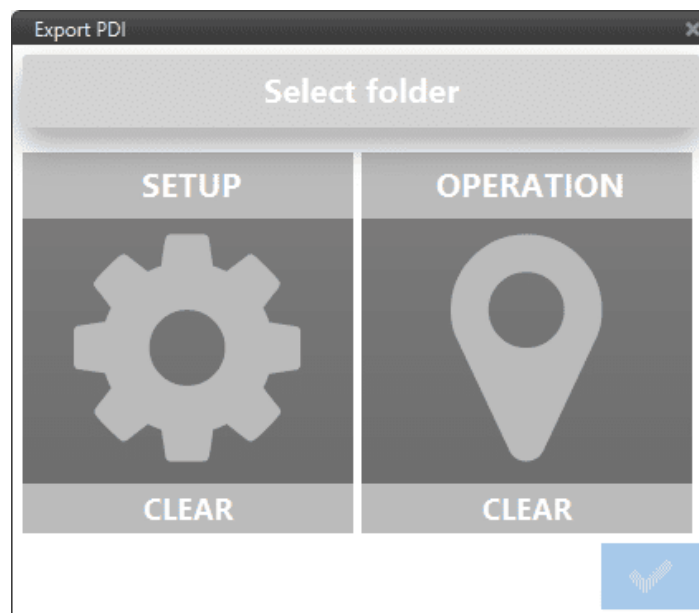
7.1.2 Export Configurations

Exporting configurations or PDI files is very useful. It allows the user to share a configuration between several autopilot units or to store different missions that can be loaded when needed. Currently, two folders are exported, **Setup** and **Operation**. Each folder contains specific files with all this information. Setup holds the configuration of the aircraft or vehicle and Operation all the files related with the mission. For more details about this folders and its files, go to [File Management](#).



Side Panel Option

To export a configuration, go to the side panel, click on  and then click on **Export PDI**:



Export Configuration Menu

Once the Export PDI menu opens, the user has several options. The general workflow is to click on “Select Folder” and select a folder where the Setup and Operation ones will be created and stored. Once selected, the panels of the export menu will turn blue:

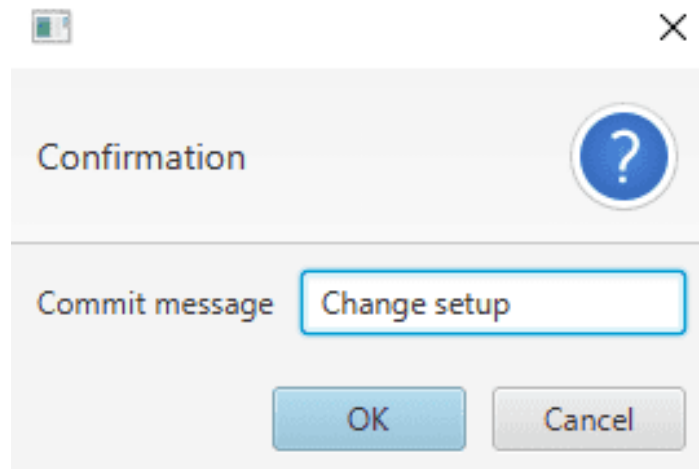


Confirmation and choosing

The user may decide not to store one of them. In that case, right-clicking on the panel, or clicking the “clear” button below it, will prevent the export of those files.

It is recommended to work with a repository for version control of the configurations so there is track of the changes done to them and the modifications go together with a commit message for later reviews.

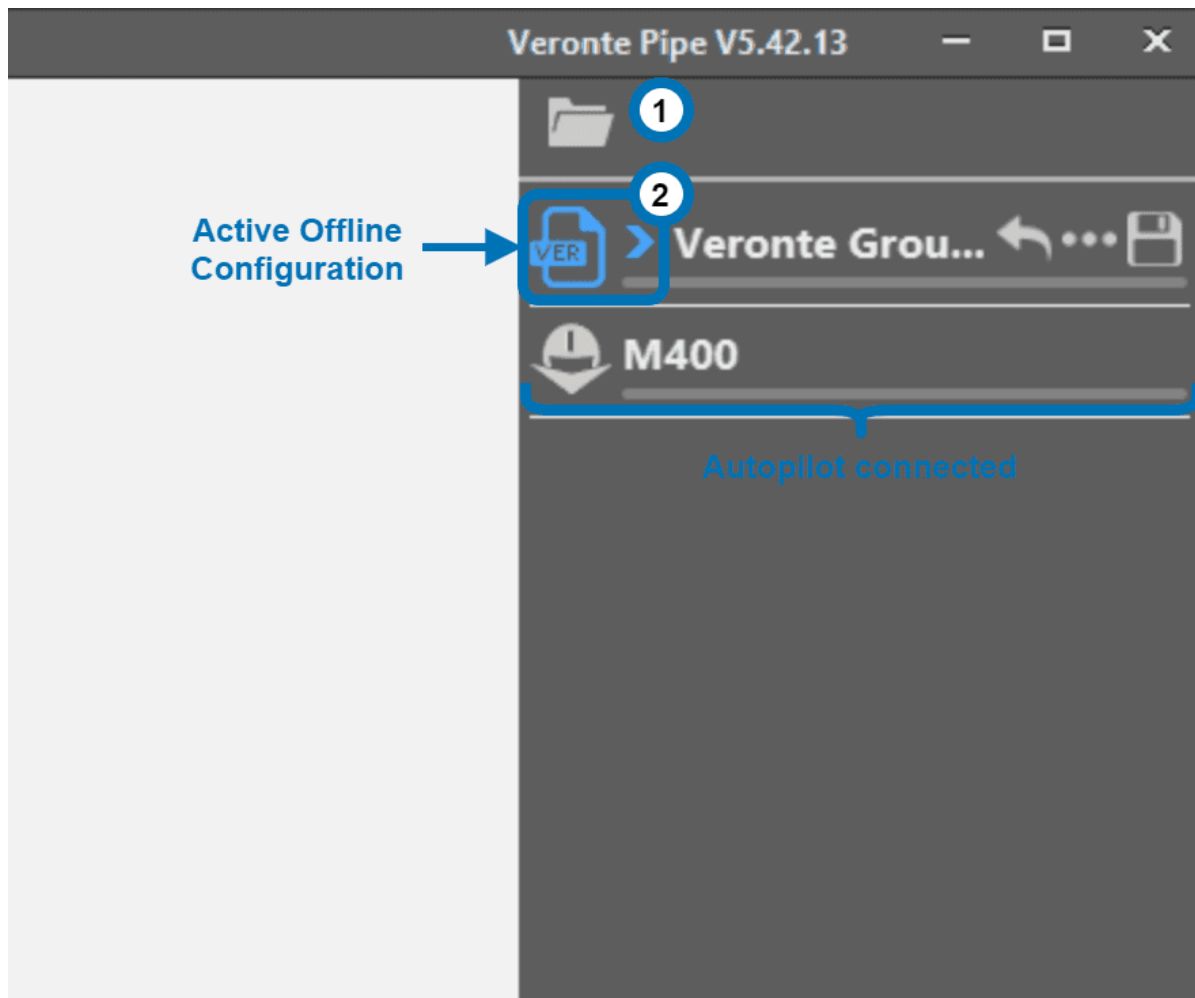
If you have a repository set up, Pipe makes version control easy. The software will detect when you are going to save changes to the repository and, automatically, a window appears to write a commit message.




Confirmation and choosing

7.1.3 Offline Configurations

It is possible to load and modify a configuration when an autopilot/s is not connected. That configuration file can be changed and exported, and it is also possible to overwrite it importing PDI files.



Offline Configuration

First, click on  (1). The file management menu will pop up and the user can select the configuration to edit or build a new one from the templates. After uploading it, click on the configuration (2) to activate it. The icon will turn blue. Several configurations can be open simultaneously in Veronte Pipe.

The main features available when working with a configuration file are the **Mission** and **Setup** configuration menus. The user can modify them and save them.

If the user is working on a new configuration starting from a template, it is recommended to **export** the PDI files to a new folder, as explained in the previous section [Export Configuration](#).

Once finished, this configuration file can be uploaded to the Autopilot following the steps described in the section [Import Configurations](#).

7.1.4 Version Control

Embention offers the possibility to have a tool for version control over a customer's configuration. It is performed through the platform GitHub, where it is necessary to sign in before accessing the service. The steps to do it are:

1. Access [GitHub](#).
2. Proceed through Step 1 with user data.
3. In Step 2, choose Free plan and decide upon the below options

☐ **Help me set up an organization next**

Organizations are separate from personal accounts and are best suited for businesses who need to manage permissions for many employees.

[Learn more about organizations](#)


☐ **Send me updates on GitHub news, offers, and events**

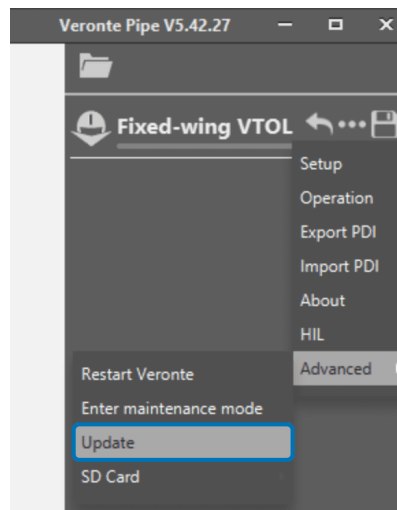
Unsubscribe anytime in your email preferences. [Learn more](#)

Step 2 Options

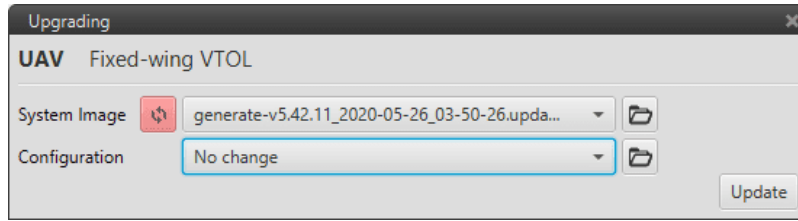
4. During Step 3, we recommend to “skip this step”, but GitHub can be configured through these options freely.
5. Finally, send your Username and email of access to Embention so we can provide you with the relevant permissions for the correspondent repositories.

7.1.5 Update Onboard Software

To display the Update dialogue click on  and choose **Advanced – Update** on the side menu.



Advanced - Update Menu



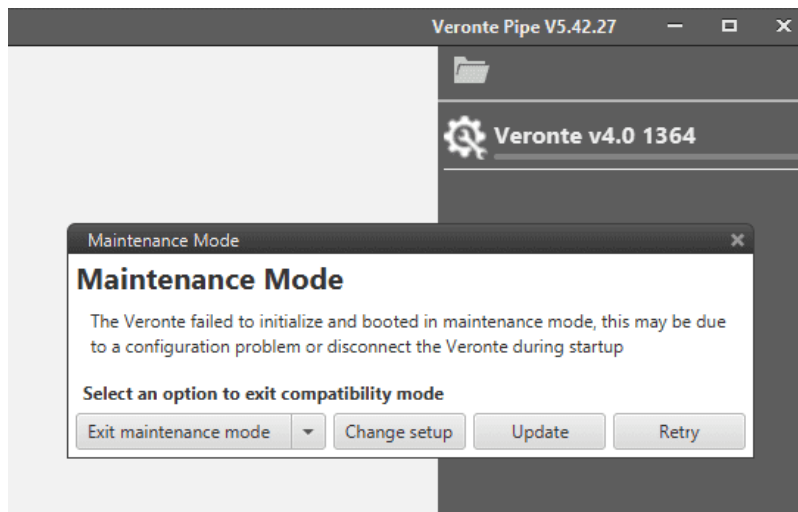
Update Veronte Autopilot

The different parameters to configure when updating the onboard software are detailed here:

- **System image:** here is where the version to be updated is selected. For instance, using *Pipe 5.42* the onboard version will be *5.42.X* (with 'X' being the last option available). Pipe will try to download the latest version but you can manually upload the .update file if is stored in the computer by clicking on the folder icon.
- **Configuration:** this option is used to decide which configuration will be loaded when the update process ends. There are different options: "No changes" will keep the current configuration for the new version. If you want to change it, click on the folder icon. It will display the *Import Configurations* window. Select the desired configuration and press "Update".

7.1.6 Maintenance Mode

Whenever there is an issue with connection, powering or configuration, the unit enters in Maintenance Mode. The autopilot icon changes and the following window is displayed:



Maintenance Mode

The options available are:

- **Exit Maintenance Mode.** The Veronte AP tries to initialise again in order to exit Maintenance Mode. Whatever option the user chooses of the ones described below to fix the issue, this button will be the last step.
- **Change setup:** this option is used to load a configuration file into Veronte. The user has to select a configuration file, once selected, a new window will be displayed in Veronte Pipe showing the version and identification of the configuration file and the autopilot where the file is going to be loaded.

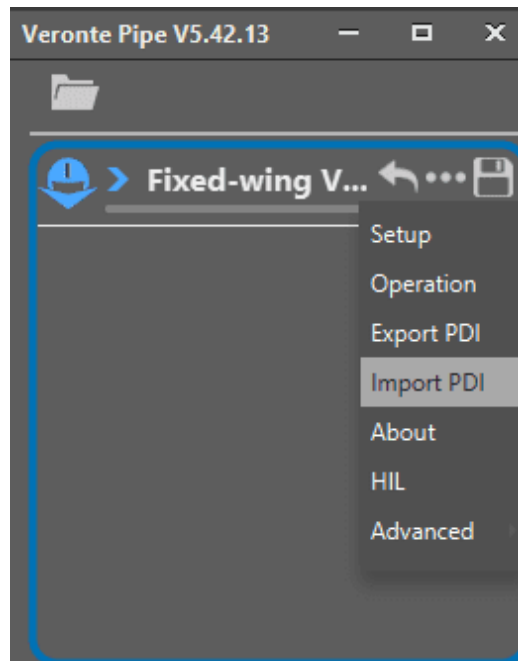


Upload Configuration - Maintenance Mode

Press “Start” and this option will upload the configuration file directly in the autopilot. Now Pipe works a tool to load the file from the computer to the autopilot and the configuration parameters will not be shown in the software window before being loaded on the autopilot.

- **Update:** this option allows the user to Update the Veronte Unit. This is explained in [Update Onboard Software](#).
- **Retry:** the unit will try to boot again.

This section explains the management of configuration files. Veronte Pipe allows users to import or export configuration files and Update Veronte Units in the Side Panel.



Side Panel Options

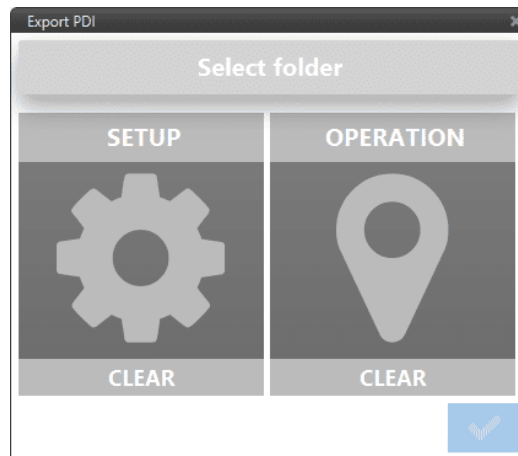
The Setup Toolbar of Veronte Pipe contains all the parameters to configure the system, which includes the autopilots (Air and Ground) and the software (Veronte Pipe).

7.1.7 PDI Files

PDI files are Veronte configuration files. This files allows for modular control with improved version management. PDI files are split in two folders. Each folder hold several .xml files:

- **Setup.** The Setup folder contains the configuration of the aircraft or vehicle. All the control loops and their parameters, the definition of the flight phases and guidance commands, and the automations defined are stored here.
- **Operation.** This folder holds all the files related with the operations defined: waypoints, routes, operative parameters, runaways, etc.

It is recommended to work with a repository for version control so each time the user exports a configuration there is track of the changes done and the saving goes together with a commit message for later revision. Embention offers the possibility to have a tool for version control over a user's configuration. Check [Version Control](#) for more information.



PDI Files management menu

7.2 Setup Toolbar

7.2.1 Veronte

This tab consists of the identification of the autopilot, where the user can define the name of the configuration in **Vehicle Name**.

Another option available is **Encryption**, the user can click on the shield icon, see section [Encryption](#) for more information.



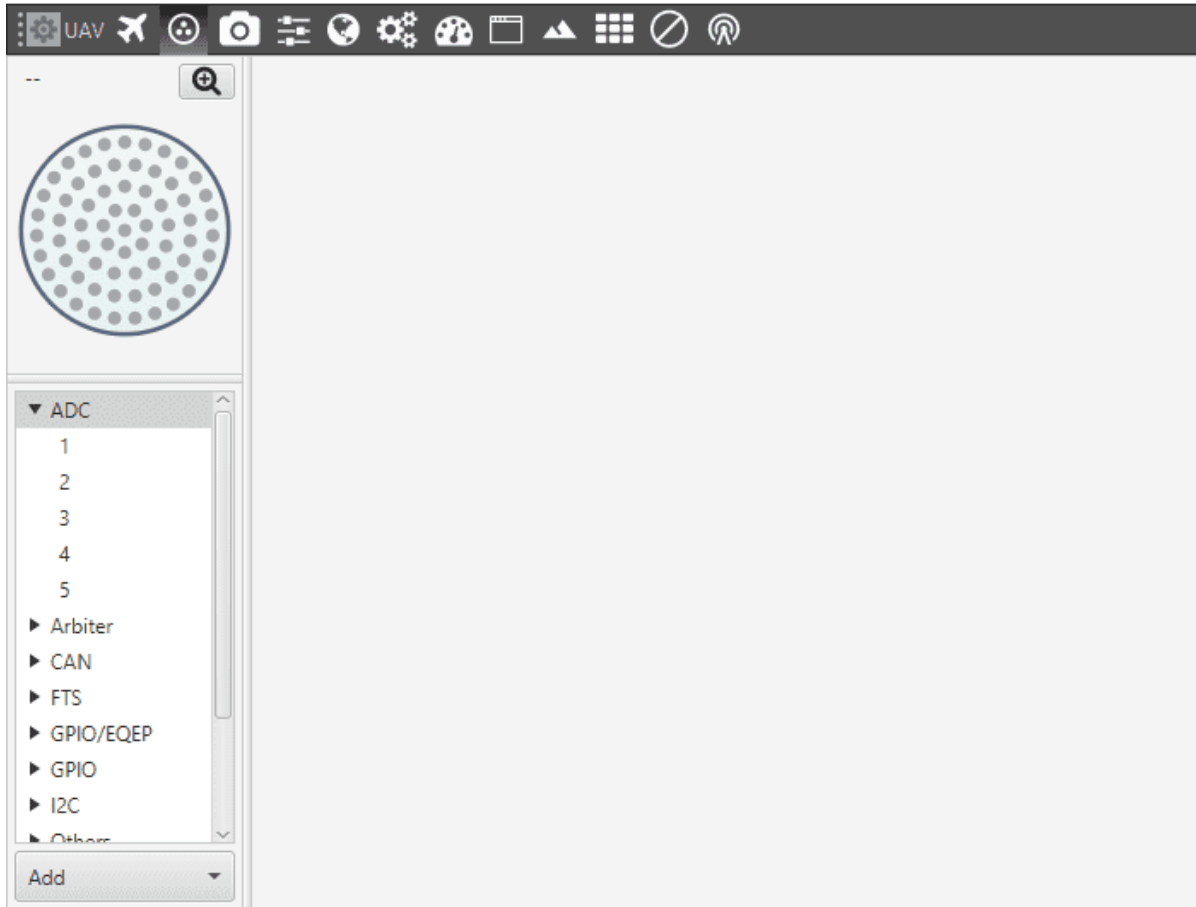
Autopilot Identification Data

7.2.2 Connections

7.2.2.1 ADC

ADC stands for Analog-to-Digital Converter. This connection is used by analog sensors. These sensors provide a voltage readout that needs to be converted into the actual measured variable, e.g. temperature, fuel volume, etc.

Veronte autopilots are equipped with **5 connections** of this kind. To set them up, the user needs to go to the *Connections menu* and click on *ADC* on the left-hand side panel. Every *ADC* connection that is set requires an integer variable associated where the voltage readout will be stored. The maximum voltage of the ADC connection is 3 V.



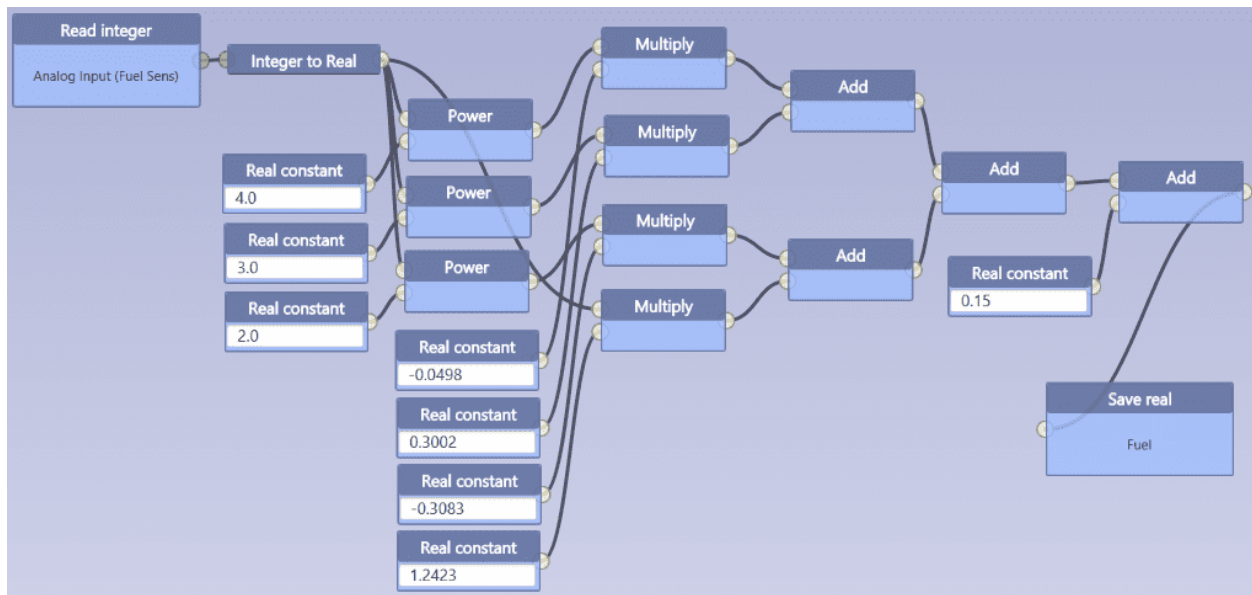
ADC Menu

To convert the input ADC value to the physical variable it represents the user needs to create a new program – see more information in [Blocks](#).

7.2.2.1.1 Application example

Let us consider a Fuel Level Sensor whose datasheet provides a direct relation of the voltage readout and the fuel volume (in L) through the polynomial $y = -0.0498x^4 + 0.3002x^3 - 0.3083x^2 + 1.2423x + 0.15$, where y is the fuel volume and x is the voltage of the sensor.

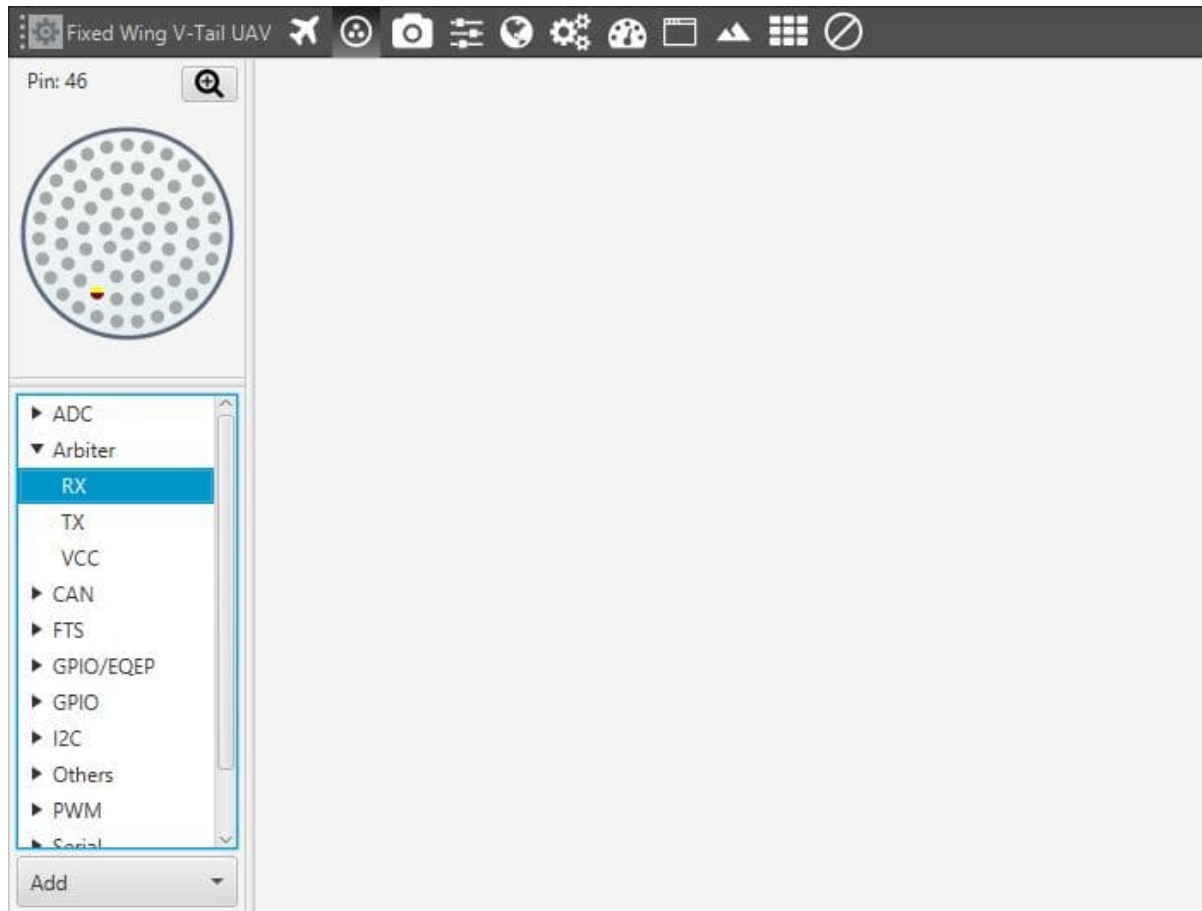
Creating a new program in the *Blocks menu*, the above can be reproduced – see Figure below. Note that the ADC variable is first converted from integer to real, and then the polynomial is applied. The fuel remaining in the tank is saved in a *user variable*, which can be used for displaying or warning pupropses.



ADC conversion for a Fuel Level Sensor

7.2.2.2 Arbiter - SuC

Pins 45 and 46 are dedicated pins to allow the uart communication with the Safety micro Controller (SuC). This microcontrollers is in charge of monitoring the state of the main microcontroller and providing the Flight Termination Signals (FTS).



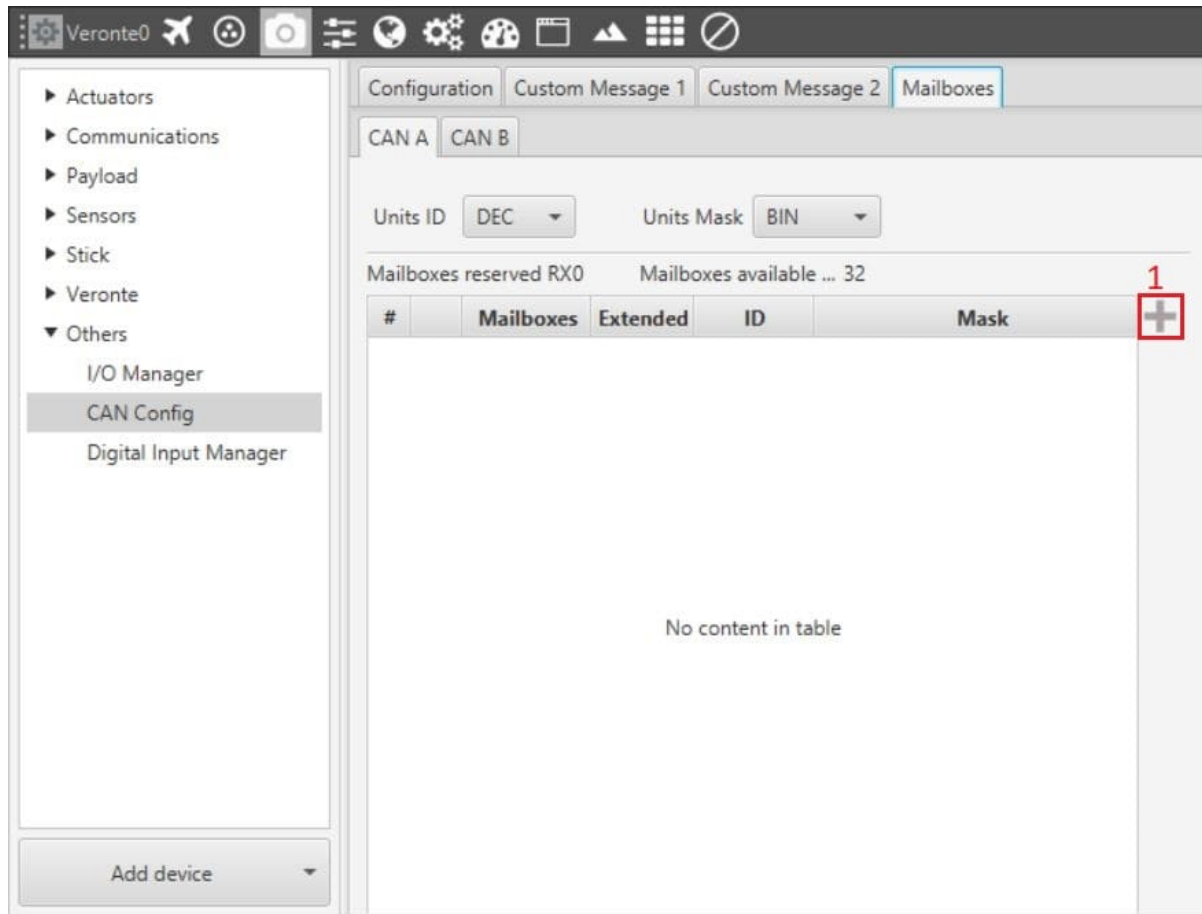
Connections – Safety microcontroller (SuC)

7.2.2.3 CAN

7.2.2.3.1 CAN Configuration

In this menu, it is possible to modify the Baudrate and the Mailboxes from the CAN bus.

When the user has to configure Veronte to receive data from the CAN Bus, it is mandatory configuring a Mailbox. In order to add a mailbox select (1).



CAN – Mailboxes

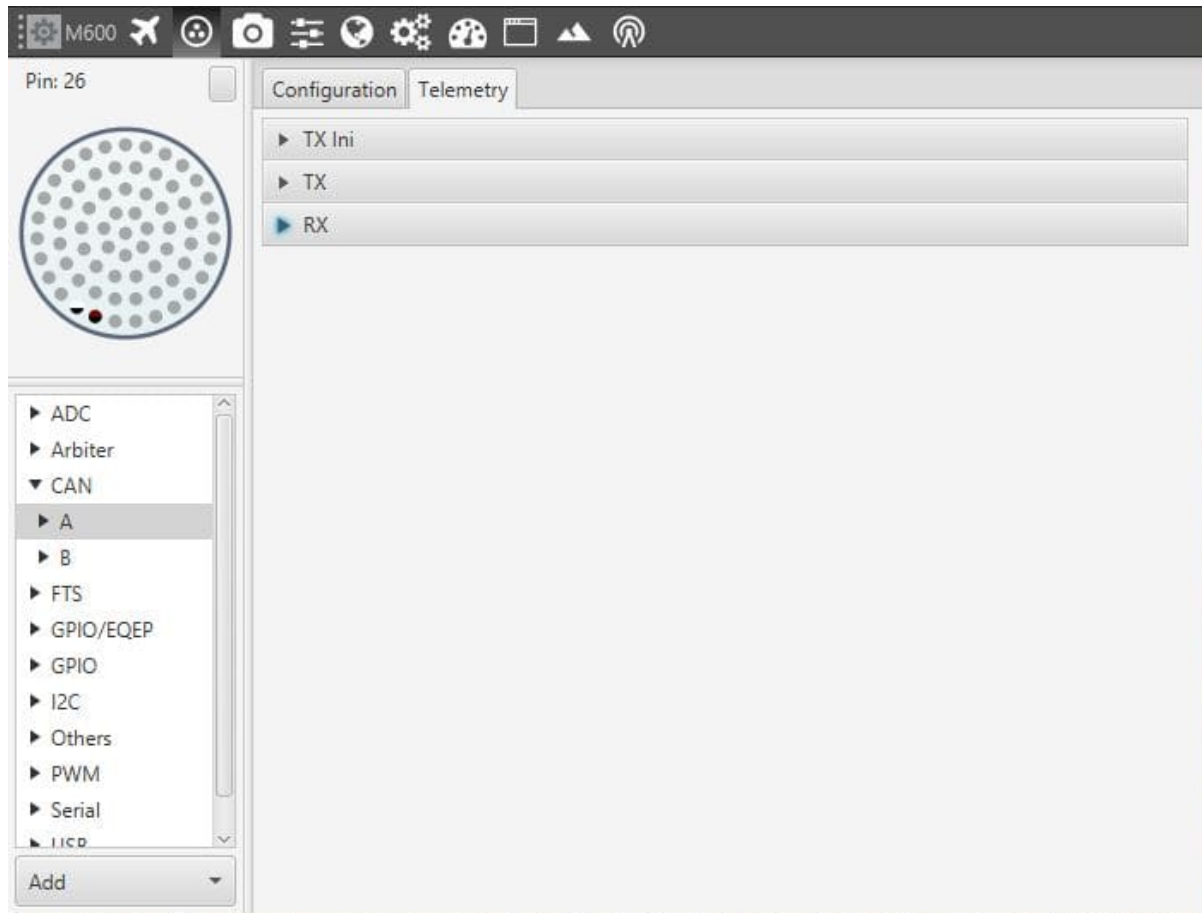
Mailboxes are used to save the data received, being possible to configure more than one. Veronte allows up to 32 mailboxes. The options available when adding a new mailbox are:

- **Mailboxes:** allows the user to identify a Mailbox with a determined configuration.
- **Extended:** frame format with 29 identifier bits.
- **ID:** identifier 11 or 29-bits (Extended), used to identify RX messages. The value set has to be decimal.
- **Mask:** This filter is configured for reception messages; received data will be stored on mailboxes where message ID coincides with mailbox ID. Mask adds some flexibility on the reception, when comparing message with mailbox data, only the value of binary digits configured as 1 on the mask will be taken into account. (e.g, for a configuration MASK: **11** 000 and ID:**10** 110 all incoming messages addressed to **10** XXX will be received in this mailbox).

7.2.2.3.2 CAN Telemetry

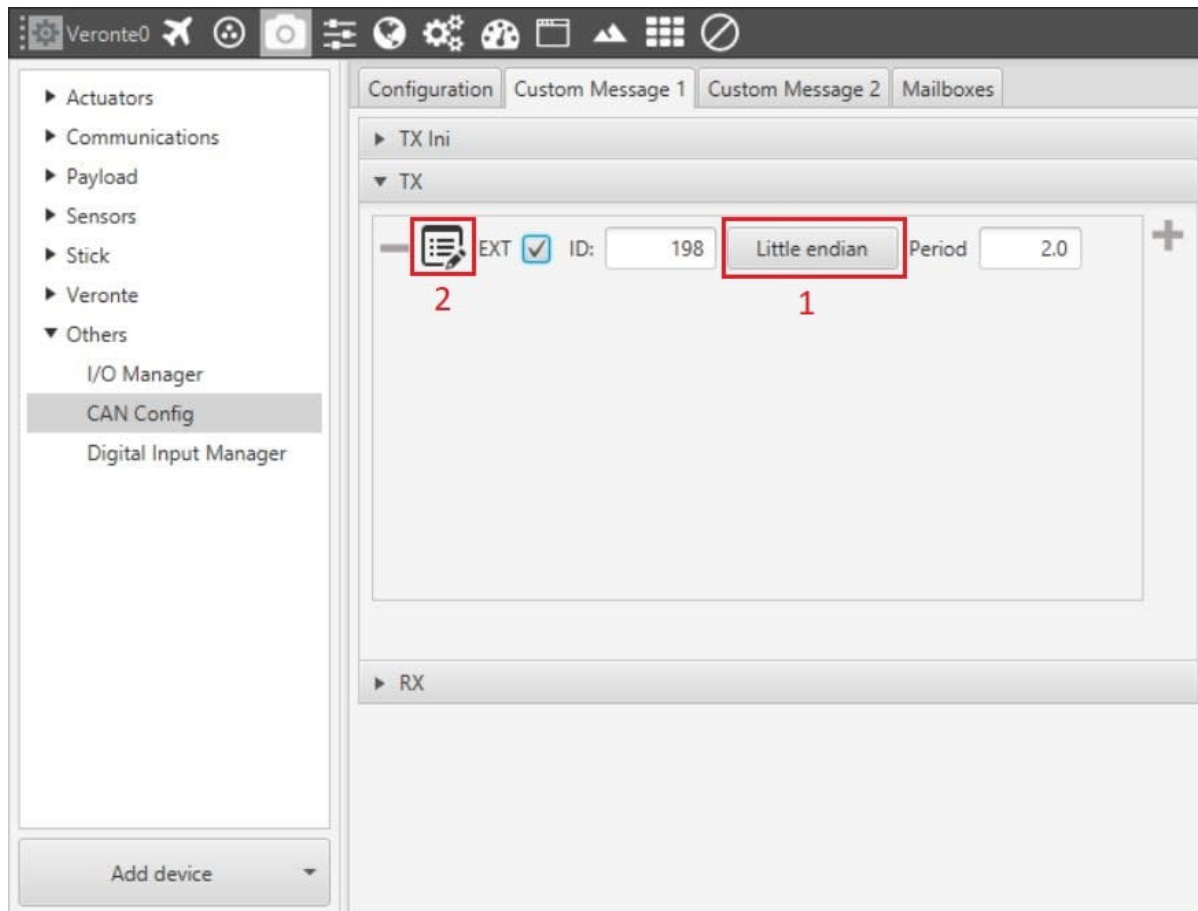
Interactions between system variables and CAN variables are managed from the telemetry configuration. In this section the following elements can be configured:

- **TX Ini:** used to configure transmitted messages that are only sent once at the beginning of the operation
- **TX:** used to configure transmitted messages through CAN bus.
- **RX:** used to configure the interaction between the variables read from the CAN bus and the variables of the system i.e, how they are stored.



CAN – Telemetry

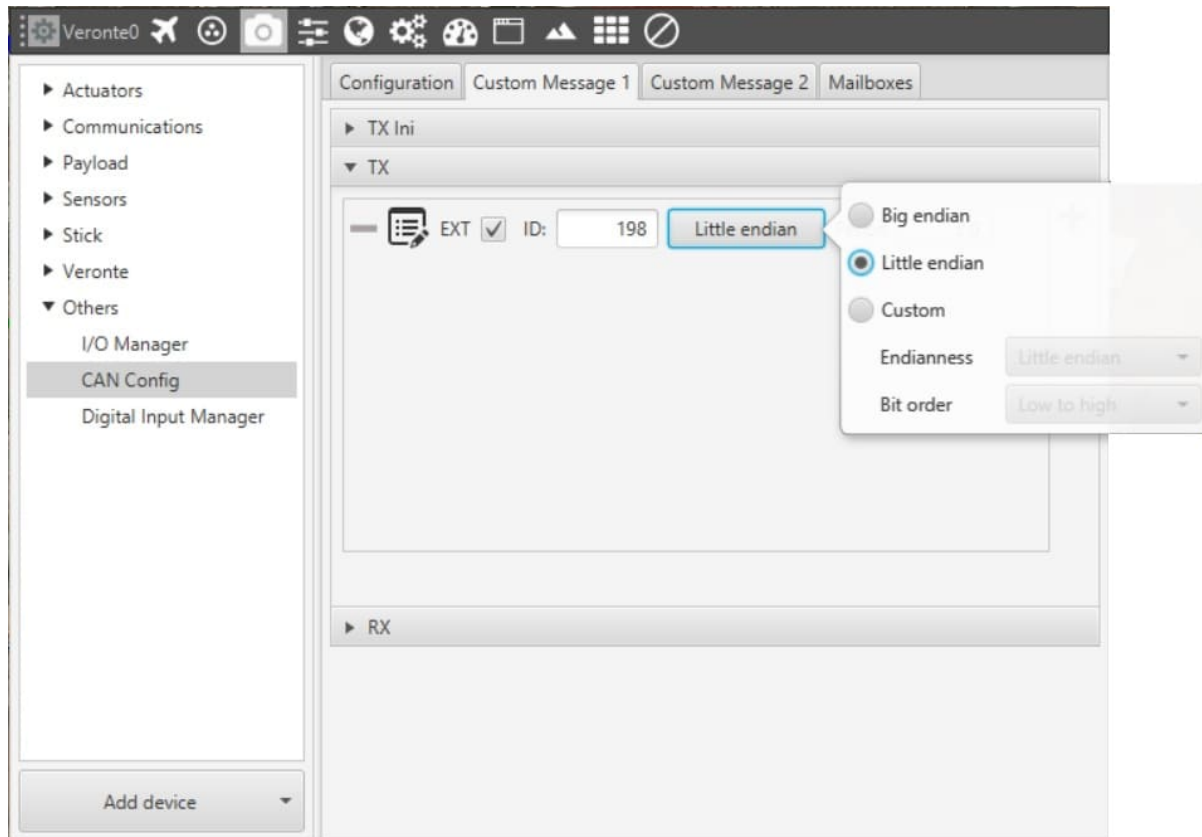
7.2.2.3.2.1 TX Messages



CAN – Telemetry - TX

In order to add a message to be sent press “+” and a new element will be added into the panel.

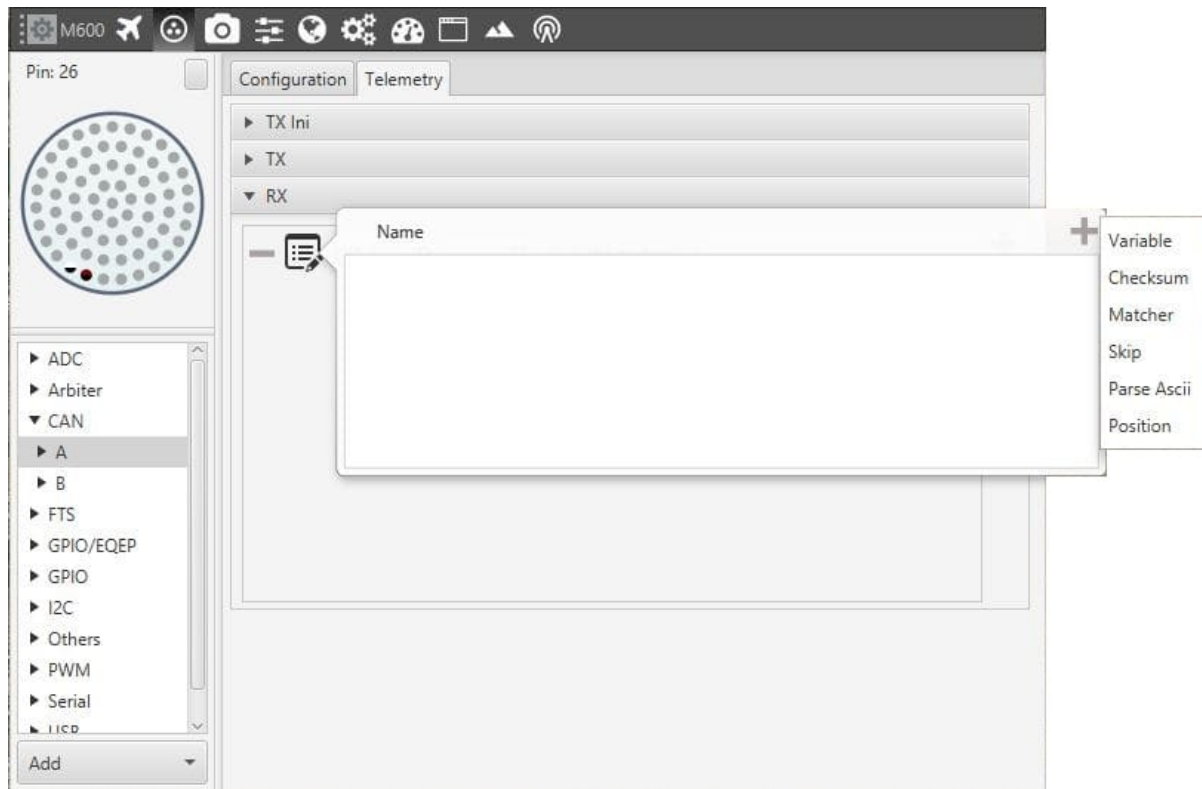
- **EXT:** enables the frame format with 29 identifier bits.
- **ID:** identifier 11 or 29-bits (Extended), used to identify TX messages. The value set has to be decimal as it was written on Configuration.
- **Period:** time in seconds between TX messages delivery.
- **Button 1:** open a new window to configure the endianness of the message, which indicates how the bytes that it contains are written:
 - **Big endian:** set the value from left to right
 - **Little endian:** set the value from right to left
 - **Mixed endian:** Any devices have this format. If you need to configure, please contact us.



CAN – Telemetry – RX Endianness

- **Button 2:** displays the menu to configure how the bits of the message are divided and sent.

There are six different elements that can be added when setting up a CAN message: *Variable*, *Checksum*, *Matcher*, *Skip*, *Parse Ascii* and *Position*.



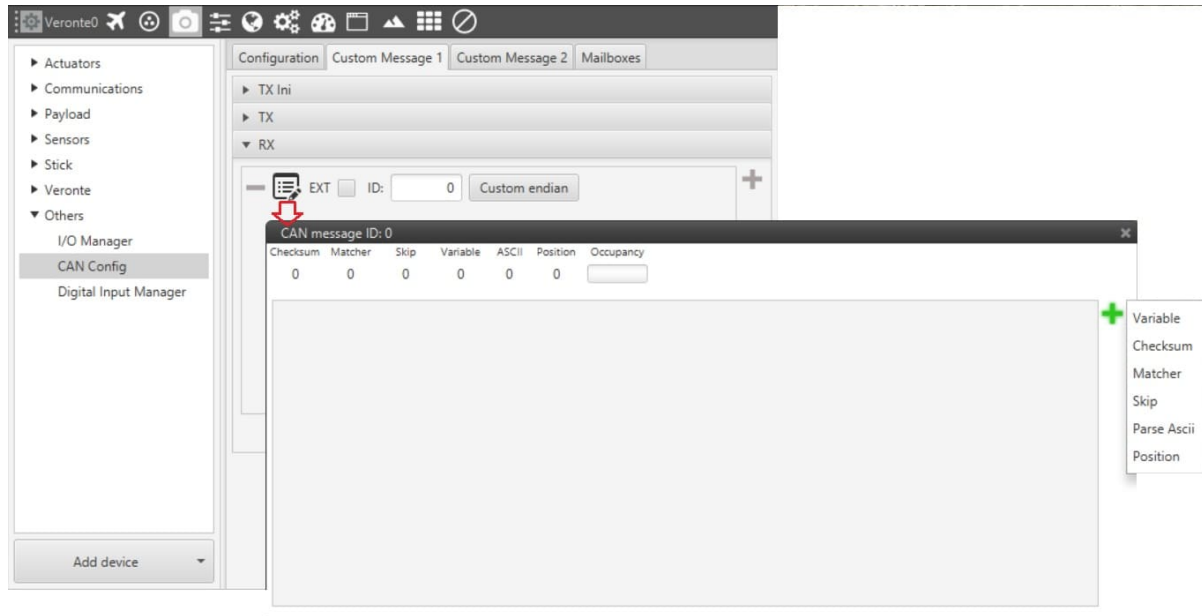
CAN – Telemetry – RX Message Configuration

7.2.2.3.2 RX Messages

Once a message is “captured” by Pipe (tab Configuration) and stored in the mailbox, its information can be read and saved in the software. To perform that, press “+” to create a new message and follow the steps explained for TX Messages, except by Period which is exclusive for TX.

7.2.2.3.3 CAN Messages

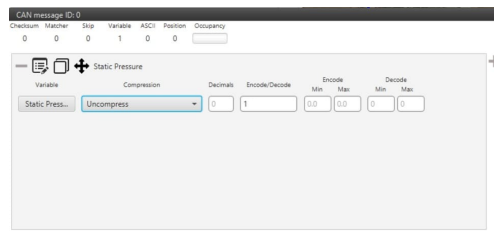
When using and configuring CAN messages either TX (transmitted) or RX (received), the following types of message are available.



Telemetry Panel

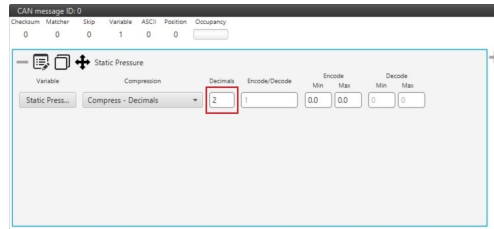
7.2.2.3.3.1 Variable

Used to store certain bits in a system variable (RX) or to send a certain variable through the CAN bus (TX).



Variable Configuration Menu

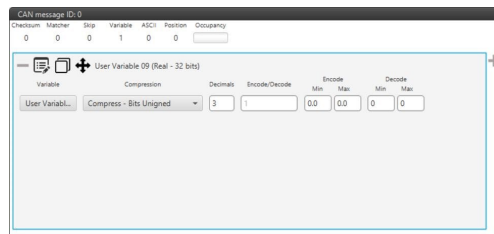
- **Type of Compression.** The first step is to configure which kind of compression will be used for this variable.
 - **Uncompressed.** The variable is taken in its full length, with no value modification. If it is a 32bits variable, it will take all 32bits from the CAN message directly.
 - **Compress.** The three subcategories of compressed Variable allow the user to input a digit, which is explained in the following lines:
 - * **Compress (Decimals).** Sets the number of decimals to use i.e, if the AGL has a value of 500.43453, it will be compressed to 500.43.
 - * **Compress (Bits signed).** Sets the number of bits, and these can be signed i.e, negative values are admitted. It is necessary that the user configures Encode/Decode options.
 - * **Compress (Bits unsigned).** Sets the number of bits which will not accept negative values. It is necessary that the user configures Encode/Decode options.



Compress Decimal Selection

- **Encode/Decode.** These values are used to apply a scaling factor after the transformation from binary to decimal value.

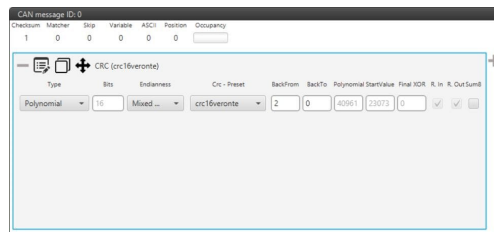
When reading data from a CAN bus it is common to have information about the message layout. In that case, let's consider that the first 3 bits correspond to a certain variable. If that variable is always positive, unsigned has to be selected, if it could be positive or negative select signed. The fields Encode/Decode are used to make the transformation from binary to decimal, corresponding Encode to the decimal value and Decode to the binary one. Considering the example shown in the following figure, the binary number is divided by 10 to make the transformation to decimal (and decimal multiplied by 10 to go to binary), so the variable represented here will go from 0 to 0.7 (as 3 unsigned bits can represent numbers from 0 to 7).



Variable Identification Example

7.2.2.3.3.2 Checksum

Sometimes, control codes are needed for preventing random errors in transmission, where a bits frame is operated and the result is sent to the receiver to check it. To do so the CheckSum option is used.



CheckSum

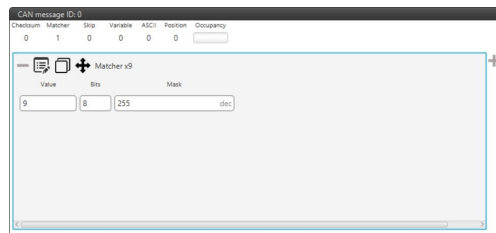
- **Back From:** Indicates that the CRC will be computed from the indicated byte (inclusive).
- **Back To:** Indicates that the CRC will be computed to the indicated byte (exclusive).

Note: Byte 0 it is referred to the first byte of the Checksum block. i.e for a message with 6 bytes where 2 bytes are the CRC. Back from = 4 and Back To = 0

- **Endianness:** Indicates how the bytes that it contains are read:
 - **Big endian:** Set the value from left to right
 - **Little endian:** Set the value from right to left
 - **Mixed endian:** For some special devices.
- **Type:** User can choose the type of CRC that will be applied.
- **CRC type.**
 - **Polynomial.** Select from a list of predefined Embention Veronte CRC. Last option is Custom, where fields as nº Bits, Polynomial, Start Value, Final Xor, Reflect In and Out can be defined. Check Polynomial CRC online for more information.
 - **Module.** Applies Module CRC.
 - **Mavlink.** Embention has implemented the Mavlink checksum, used only for Mavlink protocol communications.
- **Sum 8.** The Sum 8 checksum is the **8 bit modulo 256 checksum**. It consists of adding all bytes to take only the less significant bits whenever the sum exceeds 8 bits (255). The algorithm will be:
 - $SUM = \text{Sum on all bytes}$
 - $CHECKSUM = \text{MOD}(SUM, 256)$

7.2.2.3.3.3 Matcher

This option is used to send a constant value through the CAN bus in TX or wait for a particular value in RX.



Matcher

- **Nº Bits:** number of bits in which the matcher is performed.
- **Value:** sent/received value for the above nº of bits.
- **Mask:** it is automatically set when the nº of bits is assigned.

For example, a matcher of 8 bits with a value of 9 will be reading/sending: 0000 1001 .

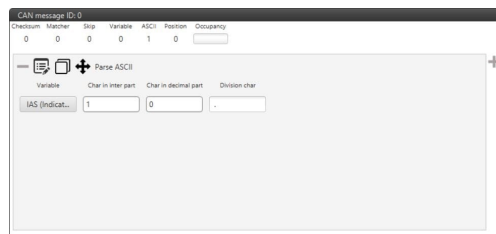
7.2.2.3.3.4 Skip

This option is used to discard a certain number of bits from the message (the maximum number of bits that can be skipped with a single “Skip” are 32). This tool can be used when there are variables incoming that are from no interest for the user, not loading unnecessary information into the system.

7.2.2.3.3.5 Parse ASCII

Parsing ASCII is used when the protocol required is of this kind. Only for RX. The variable set at the top is where the ASCII will be saved.

ASCII protocol is used for transforming a character array into decimal values. For such task, the user needs to define the number of characters in the integer and the decimal parts, as well as defining which is the division character. See how to introduce all this information in the picture below.

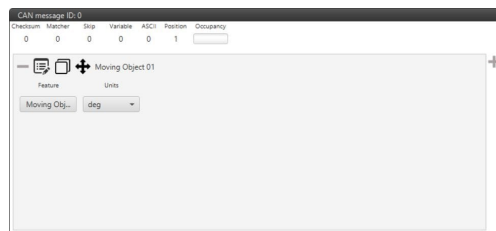


Parse ASCII Menu

7.2.2.3.3.6 Position

Position is used to input/output a data set with a particular format. When created, the user can only choose from Moving Object variables.

The window display below is the configurable menu, where it can be chosen between degrees and radians as units. The information stored is the WGS84 coordinates in the following order: Latitude, Longitude and Height. All of them are stored with double precision.



Position Menu

7.2.2.3.4 Arbiter CAN protocol

This section describes the CAN Bus protocol actually in use for the communication between the **Arbiter** and the Autopilots in **Redundant Veronte Autopilot**. Veronte is able to use both Standard CAN and Extended CAN messages. In this first integration step, the messages are configured following the Standard CAN communication protocol.

7.2.2.3.4.1 CAN parameters

The following Standard CAN parameters are described according to the Veronte Pipe configuration:

- **Baudrate:** 1 [Mbps].
- **TX Ini:** in this section is set the Start Message. **Period** has a value of 1 [s].
- **TX:** this section has to be configured as it is shown below. **Period** has to be set to 0.1 [s]
- **RX:** None.

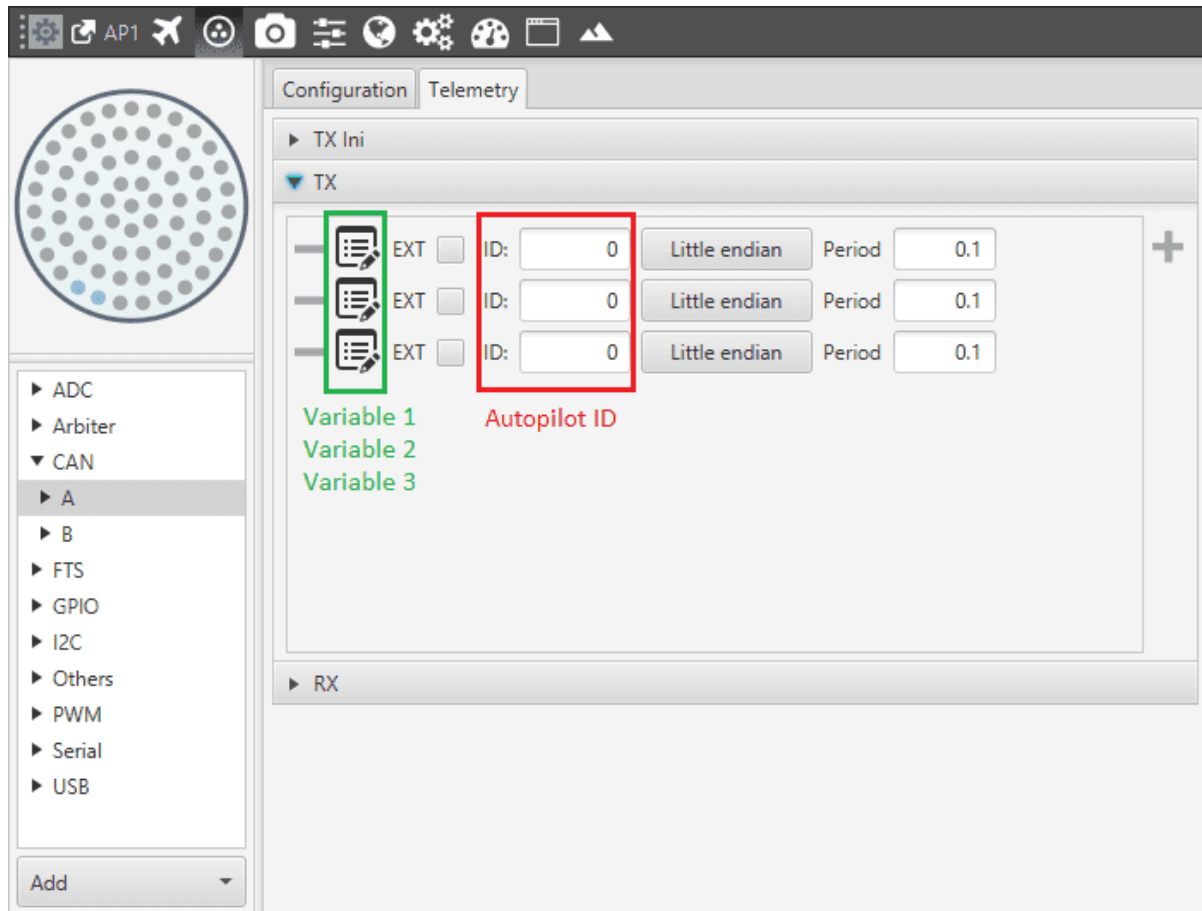
7.2.2.3.4.2 CAN message structure

The message that each Autopilot (Veronte or External) connected to the Arbiter must send through CAN Bus has to be as follows:

Table 1: Can Message Structure

| Name | Variable 1 | Variable 2 | Variable 3 | Start Message (TX Ini) |
|------------|---------------|---------------|---------------|------------------------|
| Endianness | Little Endian | Little Endian | Little Endian | Little Endian |
| ID | Autopilot ID | Autopilot ID | Autopilot ID | Autopilot ID |

- **Variable X:** variable sent to the Arbiter. In this first integration phase, they are Roll, Pitch and Yaw angles.



Variable X – Veronte AP 1

- **Start Message:** the arbiter will start to arbitrate only when it reaches the start message from all Autopilots. The start message must be like the following:

Table 2: Start Message Structure

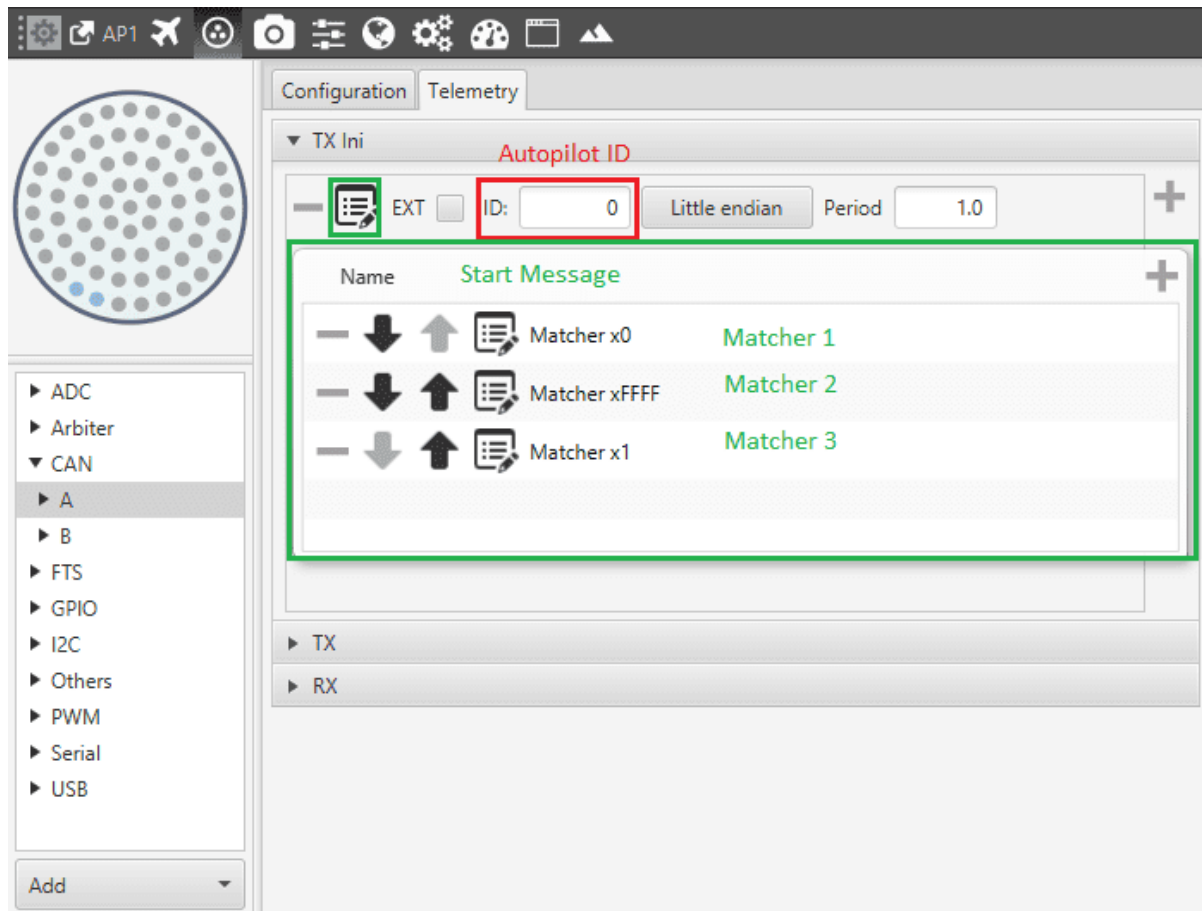
| Name: | Matcher 1 | Matcher 2 | Matcher 3 |
|-------------|--------------|-----------|-----------|
| Bits Number | 16 bits | 16 bits | 1 bit |
| Value | Autopilot ID | 65535 | 1 |

Where:

Table 3: Autopilot ID Number

| Name: | Veronte AP 1 | Veronte AP 2 | Veronte AP 3 | External AP |
|--------------|--------------|--------------|--------------|-------------|
| Autopilot ID | 0 | 1 | 2 | 3 |

The following image shows this configuration using Veronte Pipe and Veronte AP 1.



Start Message – Veronte AP 1

- **Endianness:** In this case, all the messages are configured with a Little Endian structure.
- **ID:** The ID of the Autopilot that is sending the message (value from 0 to 3).

7.2.2.3.4.3 Single message structure

The structure of the messages including the single variables sent from the Autopilots must be configured like the following:

Table 4: Variable Message Structure

| Matcher 1 | Matcher 2 | Variable |
|-----------|-----------|----------|
| 16 bits | 16 Bits | 32 bits |

Where:

- **Matcher 1:** Autopilot Identifier. In this case, the value must be from 0 to 3.

Table 5: Autopilot ID Number

| Veronte Autopilot 1 | Veronte Autopilot 2 | Veronte Autopilot 3 | External Autopilot |
|---------------------|---------------------|---------------------|--------------------|
| 0 | 1 | 2 | 3 |

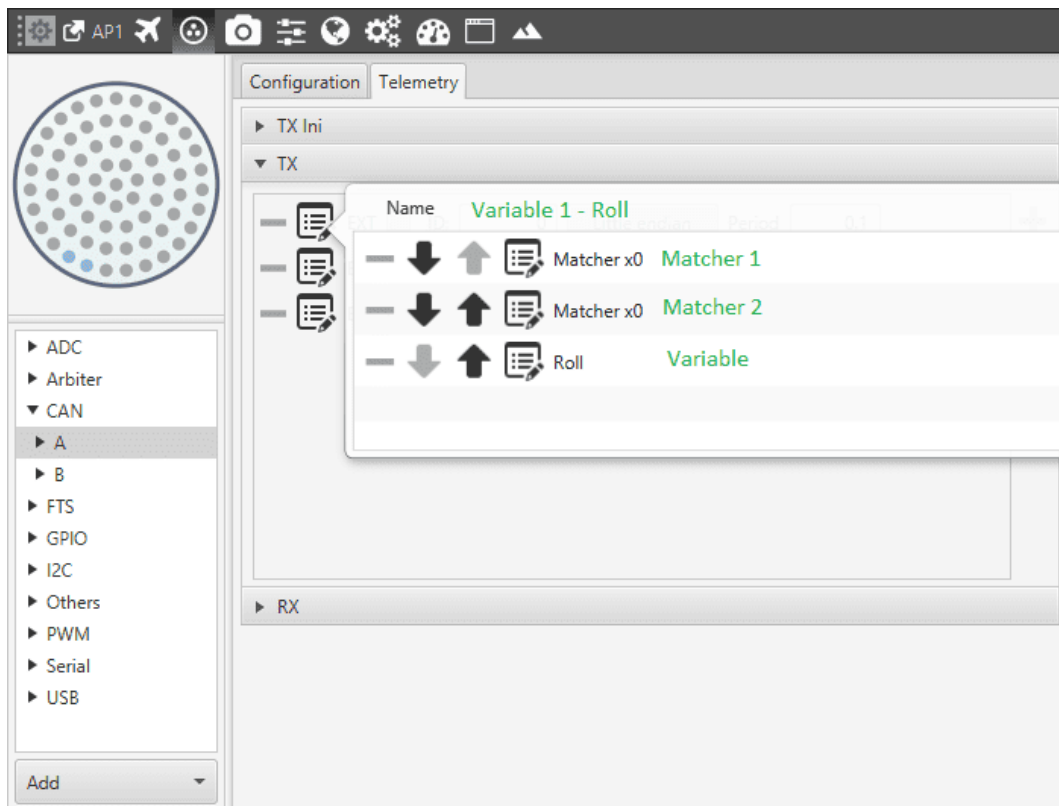
- **Matcher 2:** Value ranging from 0 to 2 meaning the variable ID.

- **Variable:** the variable that the Arbiter will receive and monitor. In this first integration step the variables are:

Table 6: Variable ID

| Variable | Description | Range [rad] | Variable ID | Bits | Type |
|----------|--|-------------|-------------|------|-------|
| Roll | Roll Angle measured from Veronte Autopilot in real time in Body Axes. | (-,) | 0 | 32 | Float |
| Pitch | Pitch Angle measured from Veronte Autopilot in real time in Body Axes. | (-,) | 1 | 32 | Float |
| Yaw | Yaw Angle measured from Veronte Autopilot in real time in Body Axes. | (0, 2) | 2 | 32 | Float |

The following figure shows the configuration for Variable 1 – Roll, using Veronte AP 1.



Variable 1 – Roll

The variables can be modified in the Autopilot CAN configuration using Veronte Pipe. To do this it is sufficient to change the variables in the single message in both the Autopilot configurations.

7.2.2.3.4.4 External autopilot simulation

When the External Autopilot is not connected to the Redundant network, it is possible to simulate its presence allowing the Arbiter to start the arbitration. To do this, it is sufficient to add a message sent from one of the active autopilots through the CAN Bus that contains the Start message of the External Autopilot (ID: 3). The following example represents this kind of message sent from Veronte Autopilot 1 (ID: 0):

| Message | Autopilot | ID | Variable ID | Type | Value | Description |
|---------|-----------|----|-------------|---------------|-------------|--------------------------------------|
| 1 | 0 | 0 | Variable | Roll Angle | Roll Angle | variable of Veronte Autopilot 1 |
| 2 | 0 | 1 | Variable | Pitch Angle | Pitch Angle | variable of Veronte Autopilot 1 |
| 3 | 0 | 2 | Variable | Yaw Angle | Yaw Angle | variable of Veronte Autopilot 1 |
| 4 | – TX Ini | 0 | 65535 | Start Message | 1 | Start message of Veronte Autopilot 1 |
| 5 | – TX Ini | 3 | 65535 | Start Message | 1 | Start message of External Autopilot |

External Autopilot Structure

Once the Arbiter starts to arbitrate, the External Autopilot will be automatically discarded and the process will be carried out using the Veronte Autopilots only.

7.2.2.3.4.5 Can protocol for 4-autopilot configuration

When the system is configured integrating four Autopilots (including then: three Veronte Autopilots, one Arbiter and one External Autopilot) the CAN Protocol structure does not change. In this case, the arbiter will expect to receive the Start Message from 4 Autopilots with different ID value that will go from 0 to 3.

The user must take into account that:

- the ID order is not important (Veronte 1 can be configured using ID:2 and vice-versa).
- the Arbiter does not need any information about the Autopilot type (External or Veronte).
- when the number of the Autopilots changes, the Arbiter software has to be modified.
- when the number of the monitored variables changes, the Arbiter software has to be modified

The following tables represent a possible configuration of the messages sent from four Autopilots using three monitored variables.

- Veronte Autopilot 1:

| Message | Autopilot | ID | Variable ID | Type | Value | Description |
|---------|-----------|----|-------------|---------------|-------------|--------------------------------------|
| 1 | 0 | 0 | Variable | Roll Angle | Roll Angle | variable of Veronte Autopilot 1 |
| 2 | 0 | 1 | Variable | Pitch Angle | Pitch Angle | variable of Veronte Autopilot 1 |
| 3 | 0 | 2 | Variable | Yaw Angle | Yaw Angle | variable of Veronte Autopilot 1 |
| 4 | – TX Ini | 0 | 65535 | Start Message | 1 | Start message of Veronte Autopilot 1 |

Message Structure AP1

- Veronte Autopilot 2:

| Message | Autopilot | ID | Variable ID | Type | Value | Description |
|---------|-------------|----|-------------|---------------|-------------|--------------------------------------|
| 1 | 1 | 0 | Variable | Roll Angle | Roll Angle | variable of Veronte Autopilot 2 |
| 2 | 1 | 1 | Variable | Pitch Angle | Pitch Angle | variable of Veronte Autopilot 2 |
| 3 | 1 | 2 | Variable | Yaw Angle | Yaw Angle | variable of Veronte Autopilot 2 |
| 4 | – TX Ini | 1 | 65535 | Start Message | 1 | Start message of Veronte Autopilot 2 |

Message Structure AP2

- Veronte Autopilot 3:

| Message | Autopilot | ID | Variable ID | Type | Value | Description |
|---------|-------------|----|-------------|---------------|-------------|--------------------------------------|
| 1 | 2 | 0 | Variable | Roll Angle | Roll Angle | variable of Veronte Autopilot 3 |
| 2 | 2 | 1 | Variable | Pitch Angle | Pitch Angle | variable of Veronte Autopilot 3 |
| 3 | 2 | 2 | Variable | Yaw Angle | Yaw Angle | variable of Veronte Autopilot 3 |
| 4 | – TX Ini | 2 | 65535 | Start Message | 1 | Start message of Veronte Autopilot 3 |

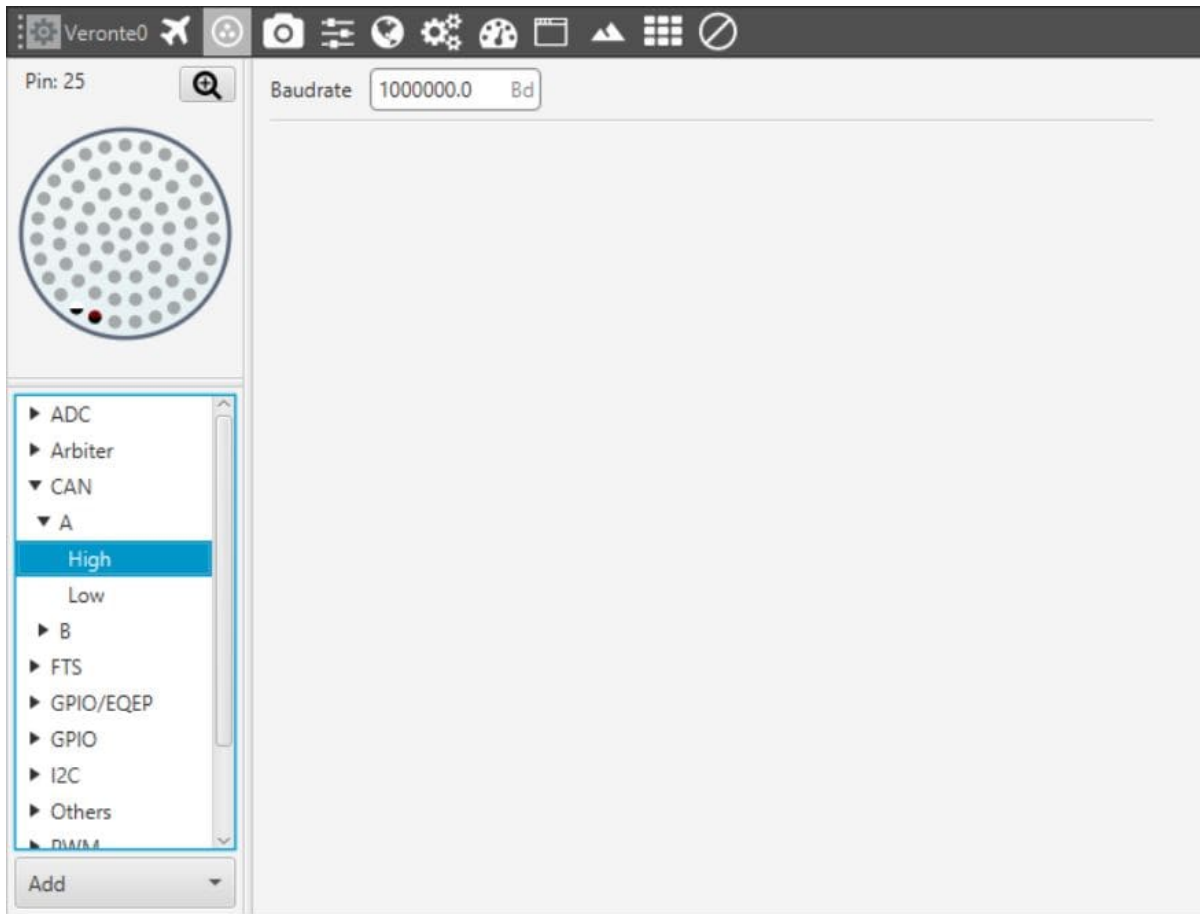
Message Structure AP3

- External Autopilot:

| Message | Autopilot | ID | Variable ID | Type | Value | Description |
|---------|-------------|----|-------------|---------------|-------------|-------------------------------------|
| 1 | 3 | 0 | Variable | Roll Angle | Roll Angle | variable of External Autopilot |
| 2 | 3 | 1 | Variable | Pitch Angle | Pitch Angle | variable of External Autopilot |
| 3 | 3 | 2 | Variable | Yaw Angle | Yaw Angle | variable of External Autopilot |
| 4 | – TX Ini | 3 | 65535 | Start Message | 1 | Start message of External Autopilot |

Message Structure External

Controller Area Network is a message-based protocol. Veronte includes two CAN bus connections (A-B), being both configurable.



Connections – CAN

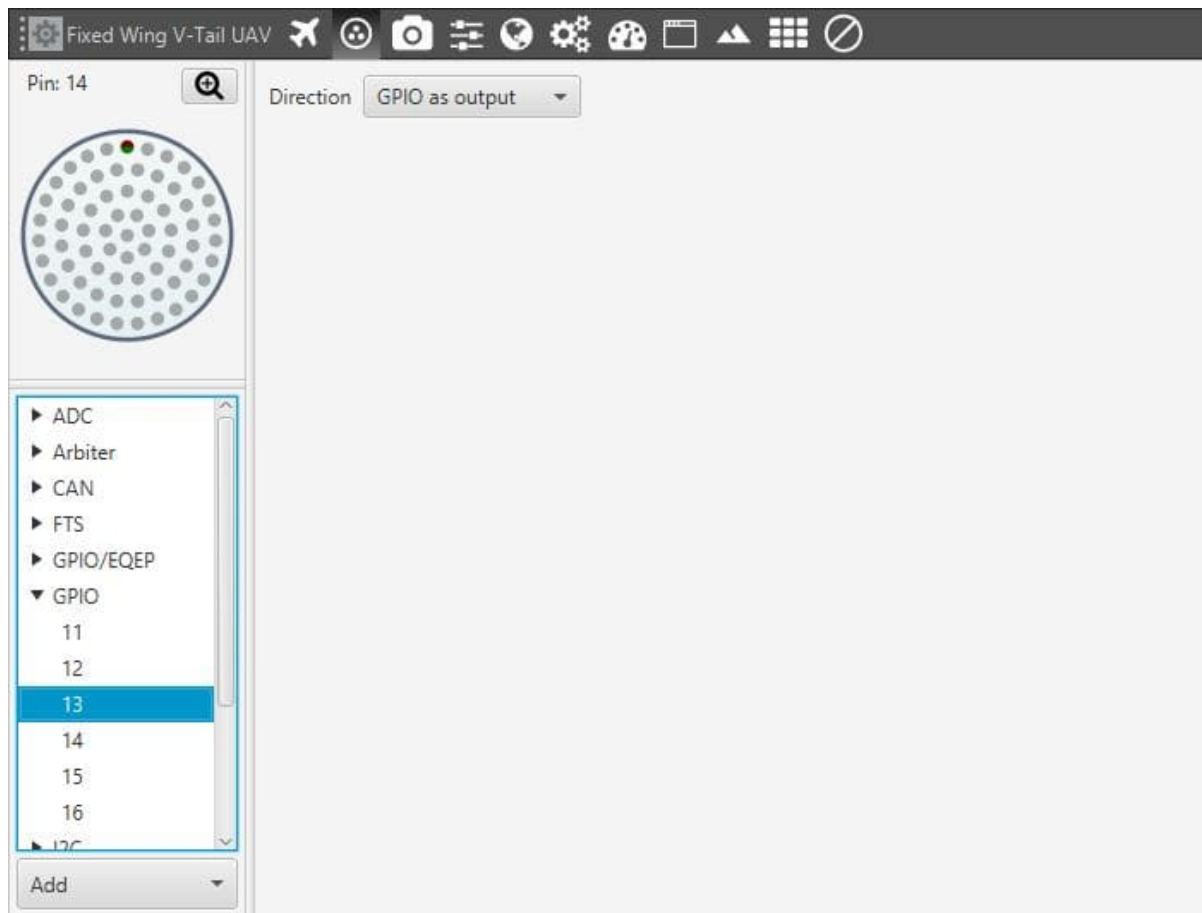
This connection allows the user to communicate with other devices. Each of them will be on the same bus, being possible to configure Veronte to transmit (TX) or receive (RX) data from other devices. In this menu you can only modify the Baudrate. To configure messages go to [Guidance](#)

7.2.2.4 GPIO/PWM

Output pins produce PWM or GPIO signals that are used to move the different servos and actuators of the platform.

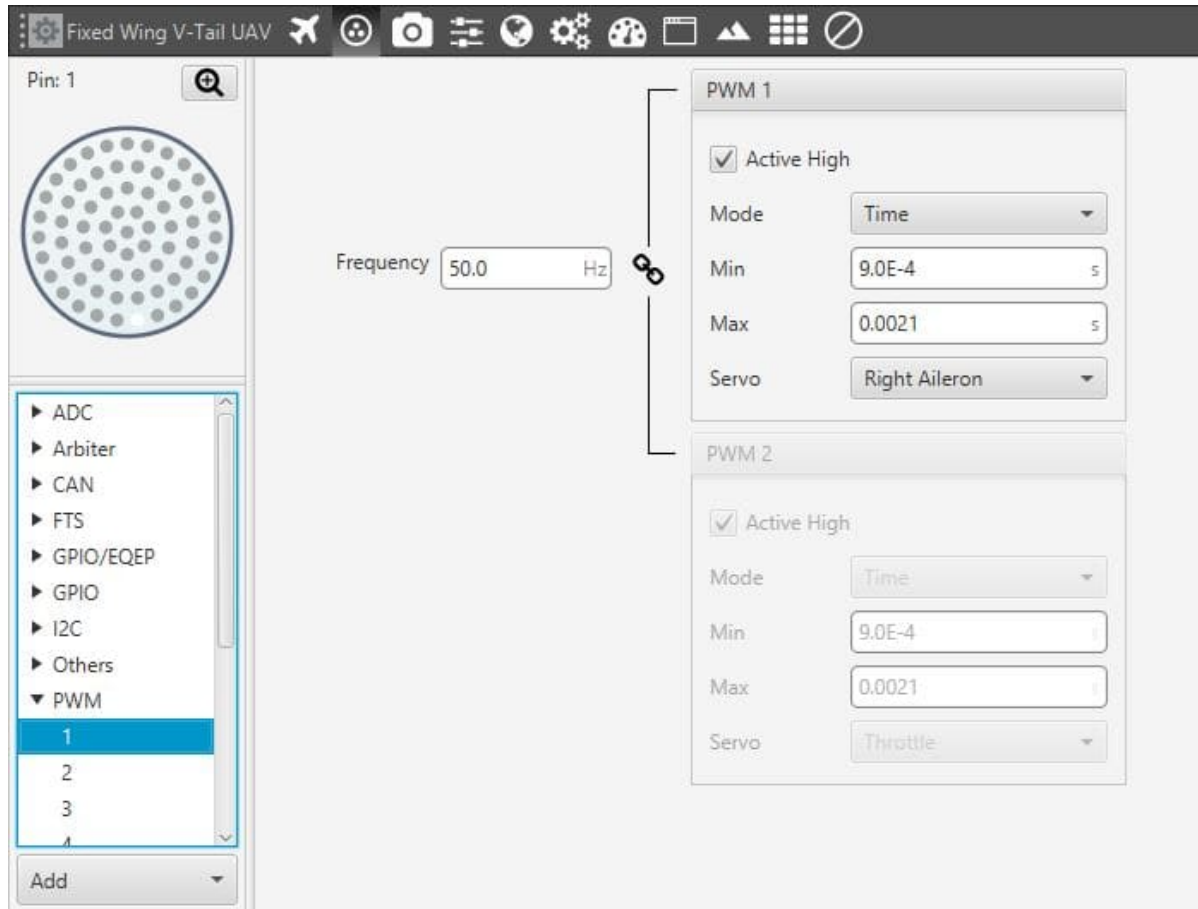
7.2.2.4.1 GPIO

A GPIO (General Purpose Input/Output) is a generic pin that can be configured as an input or output pin. When this option is configured as an output pin, the value sent will be different from the one sent if it was a PWM. GPIO pins admit up to 4 different states: ON (a continuous signal of value 1, made by 3.3V), OFF (a continuous signal of value 0, made by Ground), PULSE ON (a single pulse of value 1, with a width specified in seconds) and PULSE OFF (a single pulse of value 0, with a width specified in seconds). The configuration of the pin output value is done with an action *Output* in [Automations](#).



GPIO Menu

7.2.2.4.2 PWM

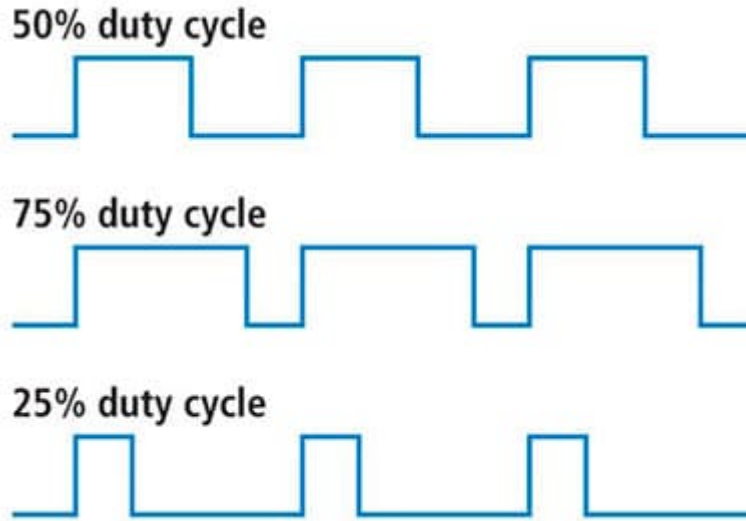


PWM Menu

The acronym PWM corresponds to Pulse Width Modulation. Veronte sends a pulse with a certain width that is received by the servo/actuator, and according to the width of such pulse, it changes its behaviour. A wide pulse will correspond to a big movement and a narrow one to a small movement.

The **Min** and **Max** parameters are the pulse width values that will make the servo/actuator go to its lowest and highest position. As an example let's consider the servo of an aircraft elevator, a pulse sent by Veronte of 0.9 ms will correspond with the lowest point of the servo range (-30 degrees for example). On the other hand, a pulse of 2.1 ms will make the servo go to its top position (for example 30 degrees).

The option **duty cycle** (select mode/duty cycle) is a different way of indicating the pulse width. Now the value indicated is a percentage which corresponds to the relation between the pulse width over the total period of the sent signal. So a 100% duty cycle will correspond to a signal with a constant value of 1, while a 0% duty cycle implies a constant signal with value 0. Between this two extremes, the pulse width can vary as in the examples shown in the following figure.



Duty Cycle

The option **Servo** is used to select which servo of the platform will be wired to that pin of Veronte connector, so the signal sent through that pin will go to it. The name of the “Actuator Output *S*” can be changed to other more identifiable according to the real actuator such as right or left aileron, see section *System Variables*.

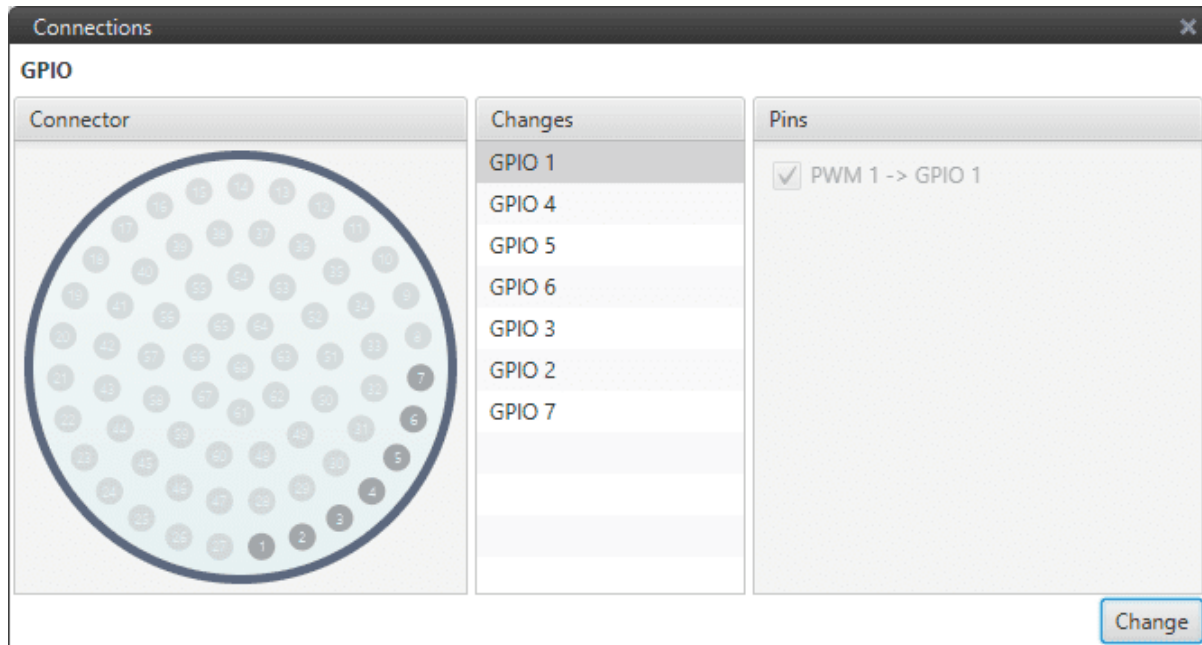
The option **Frequency** determines the period of the pulses sent by the autopilot. The PWM is built in pairs inside the autopilot, and that is why the frequency is indicated in pairs, i.e when the frequency of PWM 1 is changed, the one of PWM 2 also changes. The following table shows the PWM pairs as configured in Veronte autopilot.

| PWM Pairs | |
|-----------|--------|
| PWM 1 | PWM 2 |
| PWM 3 | PWM 4 |
| PWM 5 | PWM 6 |
| PWM 7 | PWM 8 |
| PWM 9 | PWM 10 |
| PWM 11 | PWM 12 |
| PWM 13 | PWM 14 |
| PWM 15 | PWM 16 |

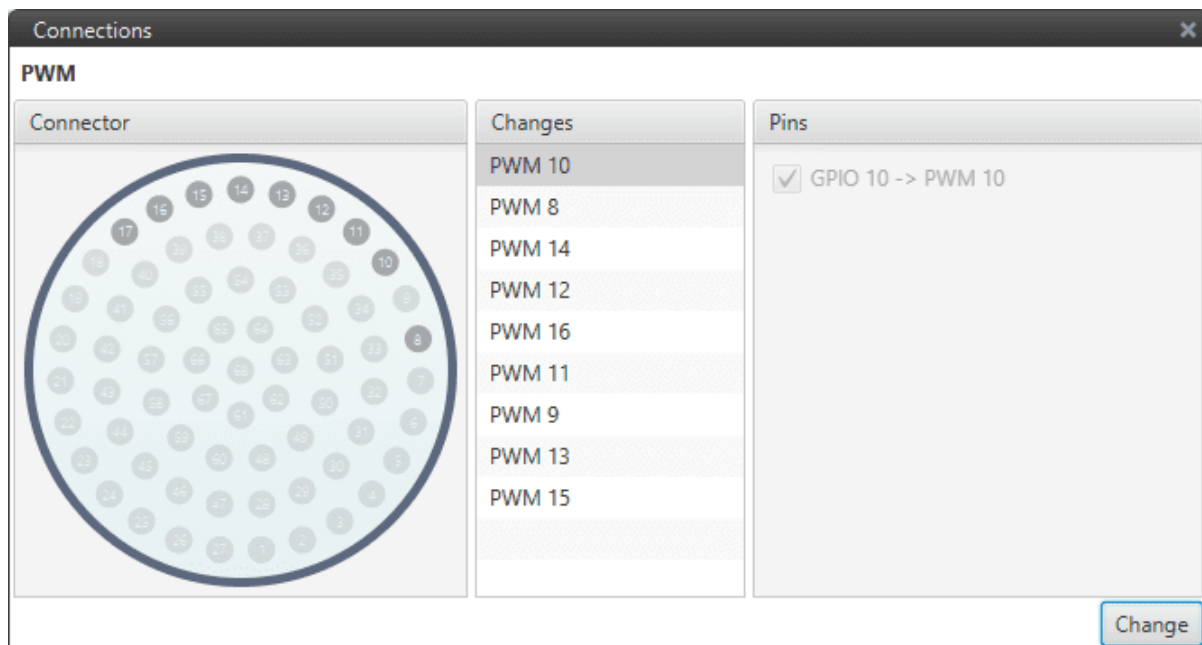
To sum up, a PWM is a signal which consists of a series of pulses having a width determined by a percentage over a range specified by the parameters **Min** and **Max**. On the other hand, the GPIO is a signal with a constant value (1,0) or with a single pulse (1,0).

7.2.2.4.3 Add

Veronte admits up to 16 I/O PWM/GPIO signals. To configure a pin as PWM or GPIO, click on **Add** and select **PWM** or **GPIO**. The following windows will be displayed according to the option selected.



GPIO Selection

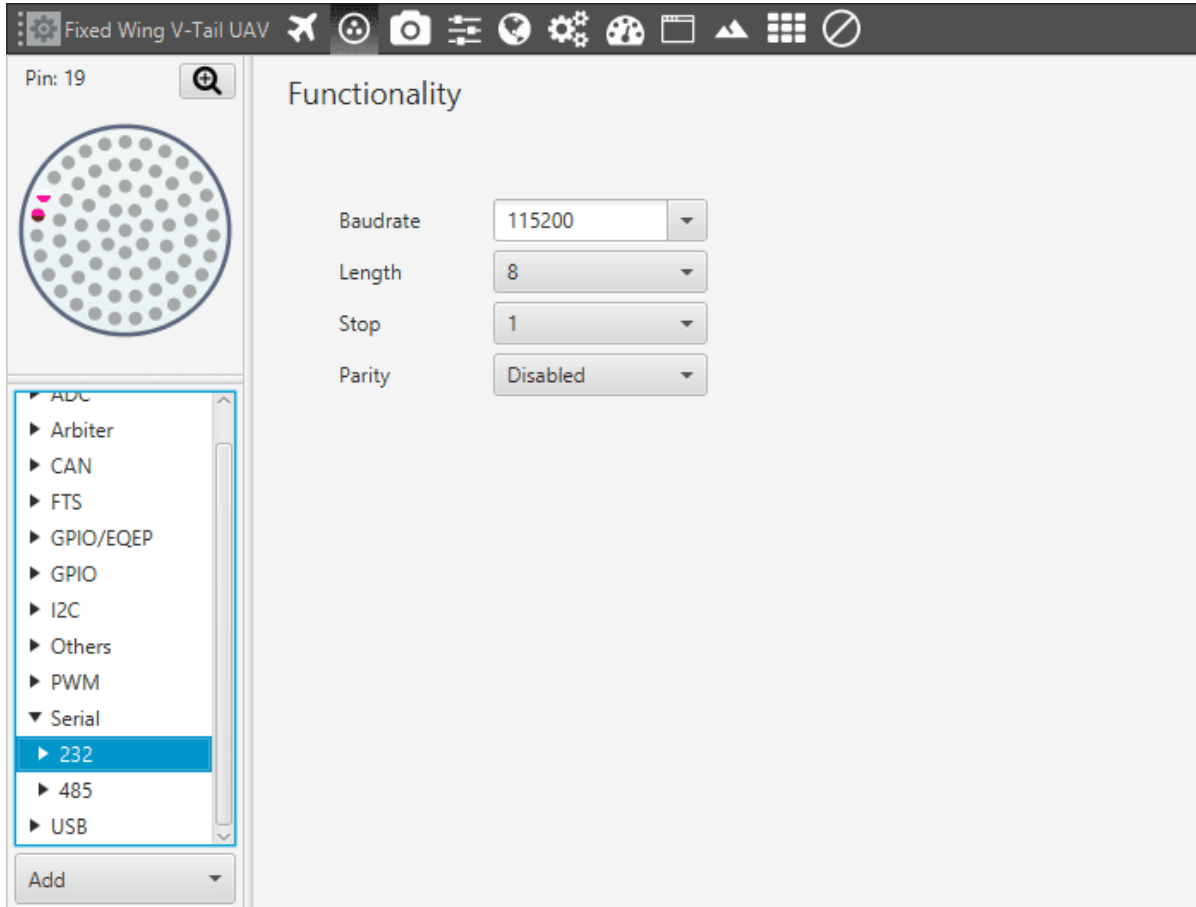


PWM Selection

As can be seen, pins are interchangeable. Once selected the desired pin, click on **Change**.

7.2.2.5 Serial

Two serial interfaces are available with Veronte Autopilots, RS-232 and RS-485, and more can be added by using the Can Expander Board. Each one of the serial interfaces is associated with a set of pins, which are displayed when an interface is selected.



Connections – Serial

The following fields can be configured:

- **Baud rate:** This field specifies how fast data is sent over a serial line.
- **Length:** This field defines the number of data bits in each character.
- **Stop:** Stop bits sent at the end of every character.
- **Parity:** is a method of detecting errors in transmission. When parity is used with a serial port, an extra data bit is sent with each data character, arranged so that the number of 1 bits in each character, including the parity bit, is always odd or always even.

| Type | Start | Data | Parity | Stop |
|------|-------|------|--------|------|
| Bits | 1 | 4-8 | 1-2 | 1-2 |

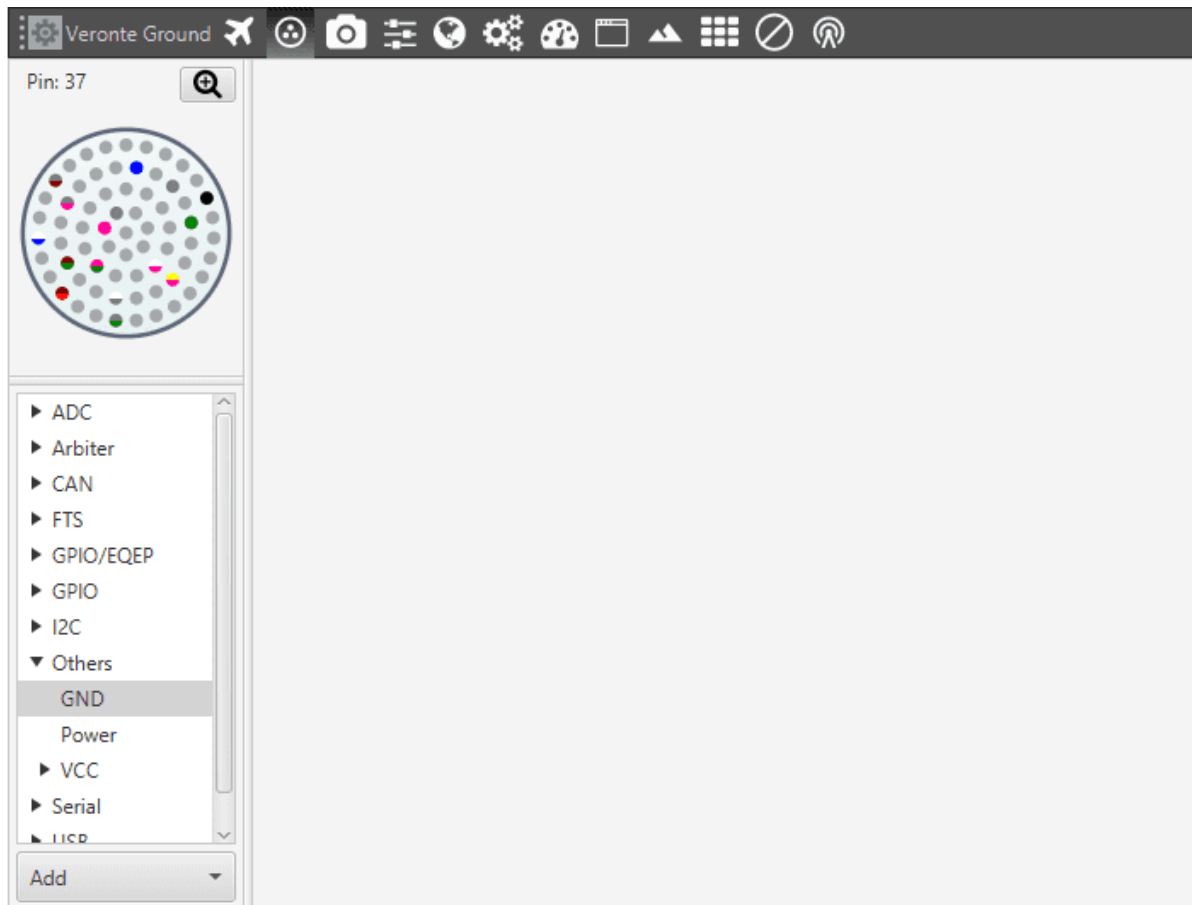
Note: All these settings are already specified for a given device, therefore, Veronte should match with them in order to be able to communicate.

Compatibility table:

| Port name | RS-232 | RS-422 RS-485 | |
|-----------------------|-----------------------|--------------------------------|-------------------------|
| Transfer Type | Full duplex | Full duplex Half/Full duplex | |
| Maximum distance | 15 meters at 9600 bps | 1200 meters at 9600 bps | 1200 meters at 9600 bps |
| Topology | Point to point | Point to point | Multi point |
| Max number of devices | 1 | 1-10 in receive mode | 32 |

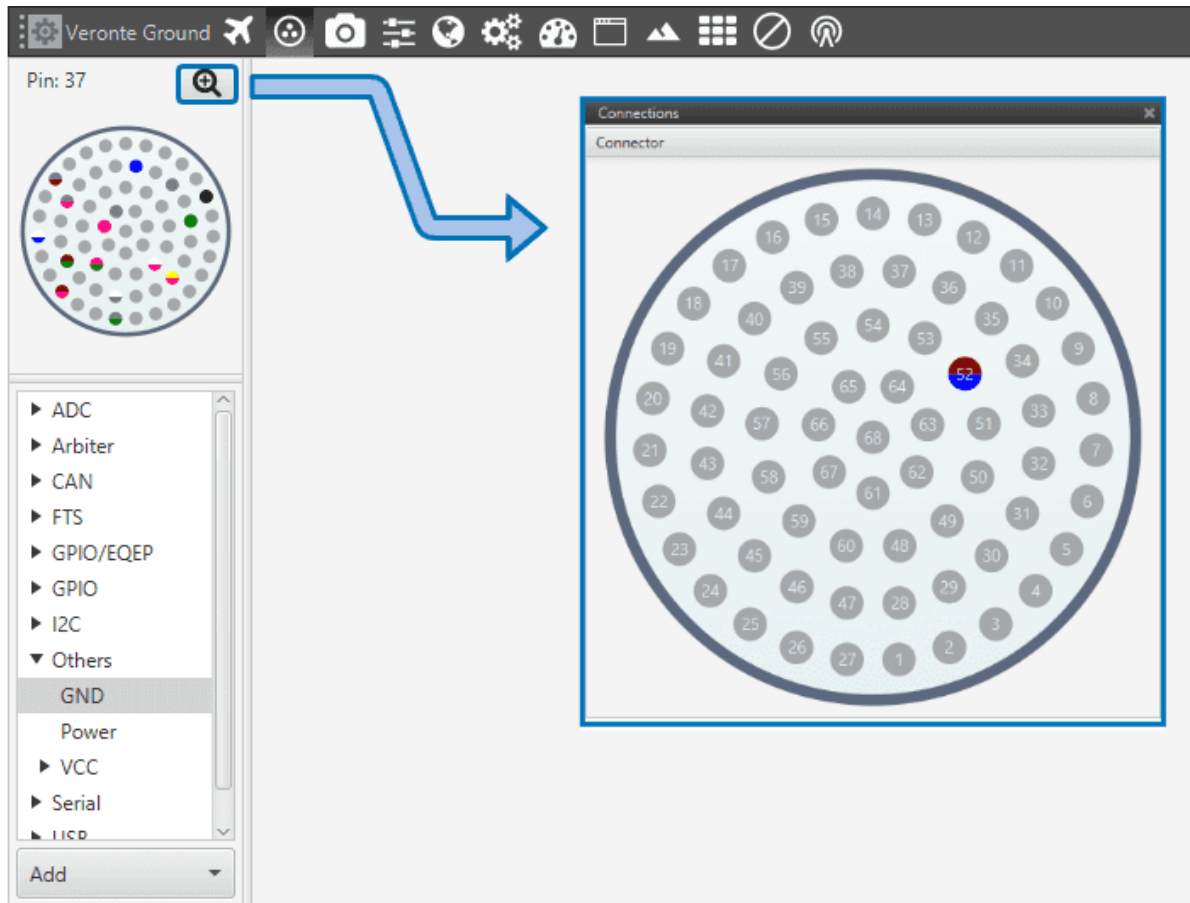
Refer to [examples](#) for a complete implementation of a serial device communication.

Here the user can configure the Input/Output ports of the autopilot. Veronte Autopilots have 68 pins, which are shown on the Pipe menu **Setup -> Connections** – see the Figure below. When selecting a port its pin is displayed on the panel.



Setup – Connections – GND Pins Displayed

To know about a particular pin's colour scheme, the user should click the button on the top right corner above the 68 pin connector picture – see the Figure below.



Connection 68 Pin Colour Scheme

Warning: The colour code is referred to the single Veronte. For 4x Veronte refer to *4x Hardware Installation - Electrical*.

Finally, depending on the configurable port selected the user will need to provide different parameters. The following table shows the type of connections available for configuration in Veronte Autopilot.

| Field | Description |
|-------------|---|
| ADC | Analog-to-Digital Converter. |
| Arbiter-SuC | Safety micro controller |
| CAN | Configurable Controller Area Network bus A & B. |
| GPIO/EQEP | Enhanced Quadrature Encoder Pulse Input. |
| GPIO | Veronte input/output signals. |
| I2C | I2C (Inter-Integrated Circuit) bus. |
| PWM | Pulse Width Modulation configuration. |
| Serial | Configurable Serial ports RS232 & RS485. |

Each connection is associated with a specific pin number. For more details see the section *Hardware Installation - Electrical*.

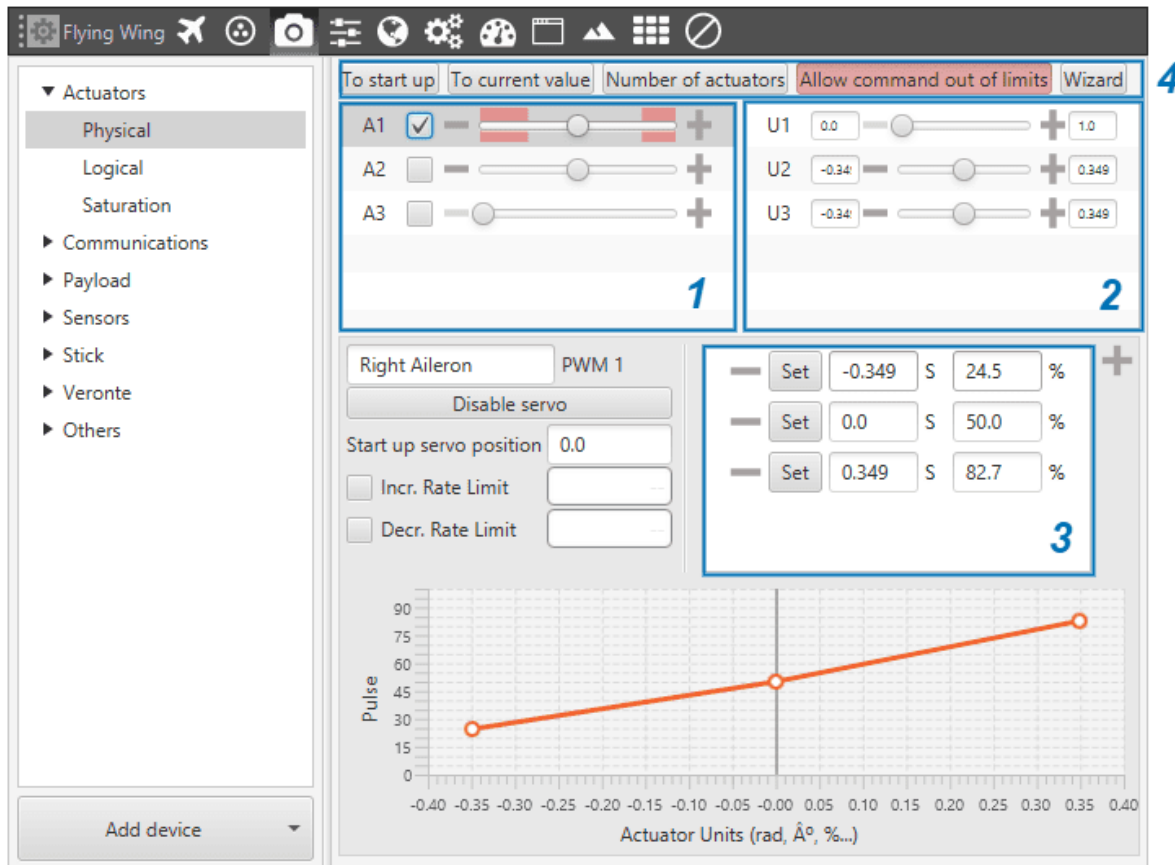
7.2.3 Devices

7.2.3.1 Actuators

7.2.3.1.1 Physical

7.2.3.1.1.1 Calibration Interface

The menu available on **Devices -> Actuators -> Physical** allows the calibration of all connected actuators. On this panel it is possible to set actuator position for each control signal/output, allowing to configure the maximum and minimum values and its **custom** performance. On the Figure below the configuration of a tailless aircraft , i.e. flying wing , is used as an example.



Physical Actuators Menu

The parameters of each one of the servos can be configured in two different ways: directly on the menu shown in the previous Figure, or through the *wizard button*. This last option is the recommended upon first configuration, leaving the other option for adjustments and advanced settings.

The options of the calibration menu are presented next:

1. **Servos (actuators):** this menu contains the servos of the platform. Moving the scrolling bar will change the servo position, but the signal to the system will only be sent if the checkbox next to the servo number (A1, A2...) is marked. The manual movement of the servos can only be done in the "Initial" phase (when there is no phase selected in Veronte Panel).
2. **Control signals:** this menu contains the variables that represent the control signals/outputs U generated by the

system. Considering the flying wing, the controls are thrusting (U1), pitching (U2) and rolling (U3), so there are 3 different controls in total. The mapping from the controls to the servo positions is indicated within the **SU matrix**, which is set in *Devices - Actuators - Logical*. When moving the scrolling bar of one of the control channels, the corresponding movement on the servos will also be represented.

3. **Servo position – PWM:** this option is used to set the mapping from servo position S to PWM signal. For example, a 20° degrees deflection (0.349 rad) of the right elevon corresponds to an 82.7 % pulse to be sent to the corresponding servo. The mapping is expressed through the graph, where the user can introduce as many points as desired.
4. The other tools that appear on the menu are:
 - **To start up:** set initial values of the actuators.
 - **To current value:** set actuators to the current value – requires the physical platform to be connected.
 - **Change size actuators:** change the total number of available actuators.
 - **Allow command out of limits:** allow out-of-limits motion for the selected actuator.
 - **Wizard:** explained below.
 - **Disable/Enable servos:** allows the user to enable or disable a configured actuator.
 - **Start up servo position:** sets the pulse value for the *start up configuration*.
 - **Increasing/Decreasing Rate limit:** sets a rate limit for increasing/decreasing motions of the servo.

The procedure to follow when configuring a servo/actuator is:

- Firstly, set the scrolling bar to its middle position.
- Secondly, click on Enable: this avoids the problem of enabling a servo when it is in an extreme position, which can produce a malfunction.
- Set the desired maximum S_{max} and minimum S_{min} values of the actuator S (a restrictive approximate value works).

Note: S_{max} and S_{min} can be chosen to match the actual physical value they represent, e.g. the angle of deflection of an aileron. But they can also be chosen not to match a physical variable, and it won't affect the autopilot's behaviour as these S values are the representation of PWM signals. And the latter are the ones that correspond to an actual control surface/device movement.

Take a motor, for instance. The minimum PWM signal corresponds to a 0 rpm state, but a maximum PWM signal corresponds to a finite number of rpms. In this case it's easier to assume $S_{max} = 1$ and $S_{min} = 0$. The same could be done for the ailerons. If they are deflected the same amount in both directions then $S_{max} = 1$ and $S_{min} = -1$.

- Finally, connect the physical actuator and adjust its minimum and maximum values.

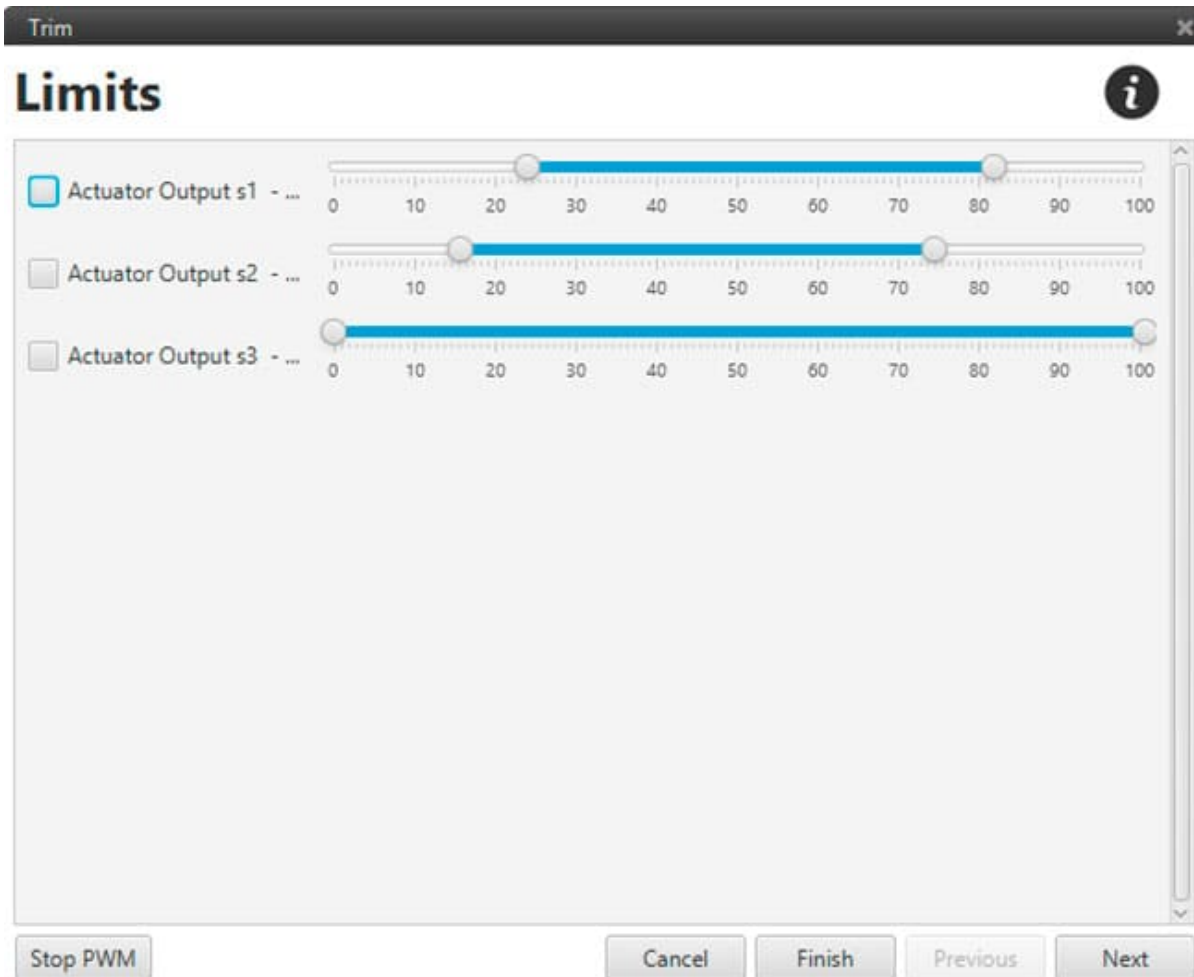
7.2.3.1.1.2 Configuration Wizard

Configuration wizard is recommended on first system configuration. It guides the user for configuring actuator limits and performance. **This calibration should be performed in order to define the relationship between the autopilot output and real servo position.**

In order to set servo positions, it is possible to enable servo movement from the user interface. Check the box on the left of each servo in order to enable servo movement, once enabled, servos will move according to the bar. On the bottom left part of the windows, there is the Stop PWM, with which all checkboxes are disabled.

If the user wants to finish the wizard and keep the results configured up to that point, just click on Finish on the bottom right part.

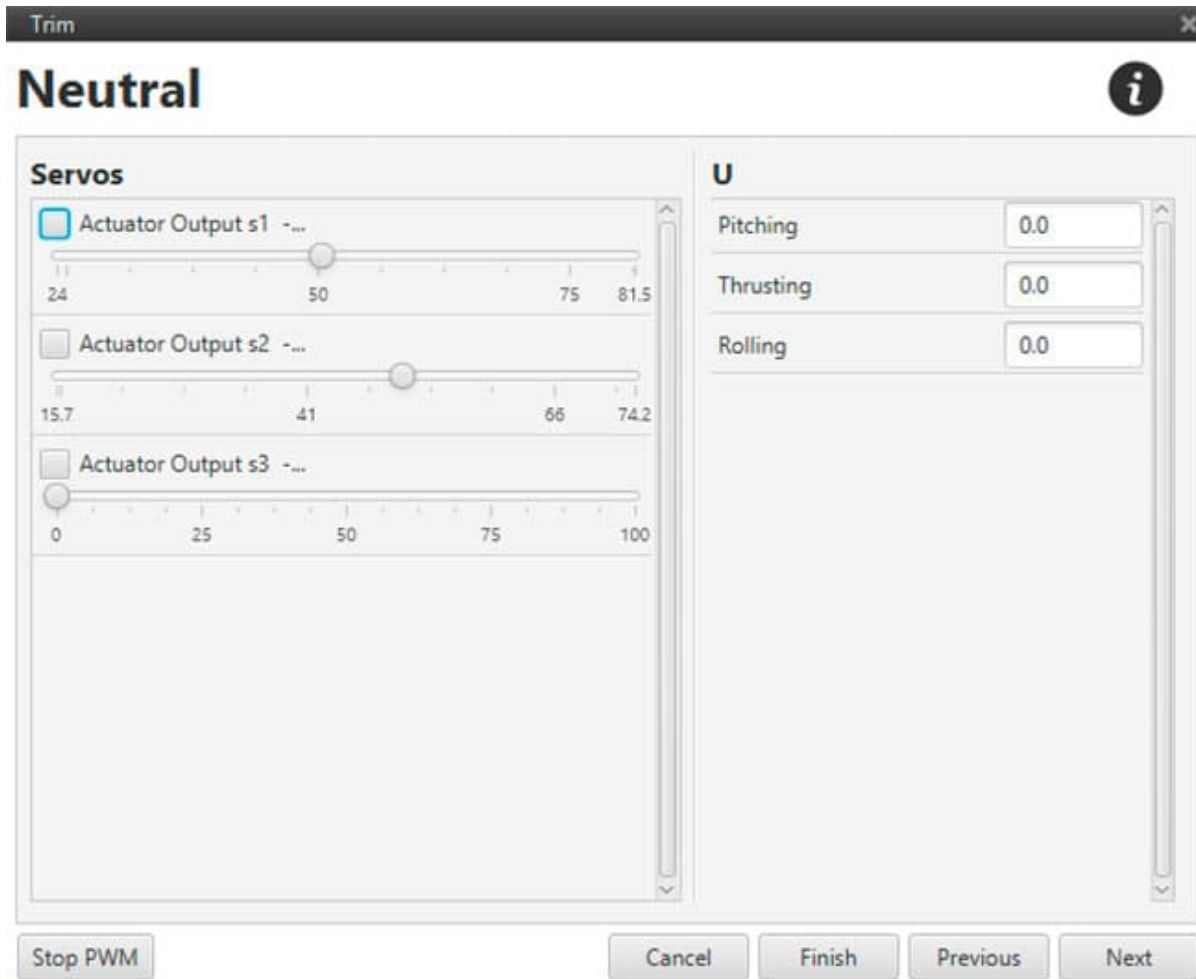
- **Limits:** Set maximum and minimum pulse value for the mechanical limits on the actuator.



Trim Wizard – Limits

The autopilot will never command the actuator outside these limits, preventing mechanical damages in the aircraft.

Neutral: Set neutral position for actuators and control channels. This position is set as a reference in the system and will remain as the default position on system startup. On the right, there is the column for U, where the neutral value for that particular control output is chosen.



Trim Wizard – Trim Wizard – Neutral

- **Select:** Select a servo to trim. The servos are sorted depending on which is affected by each control input.

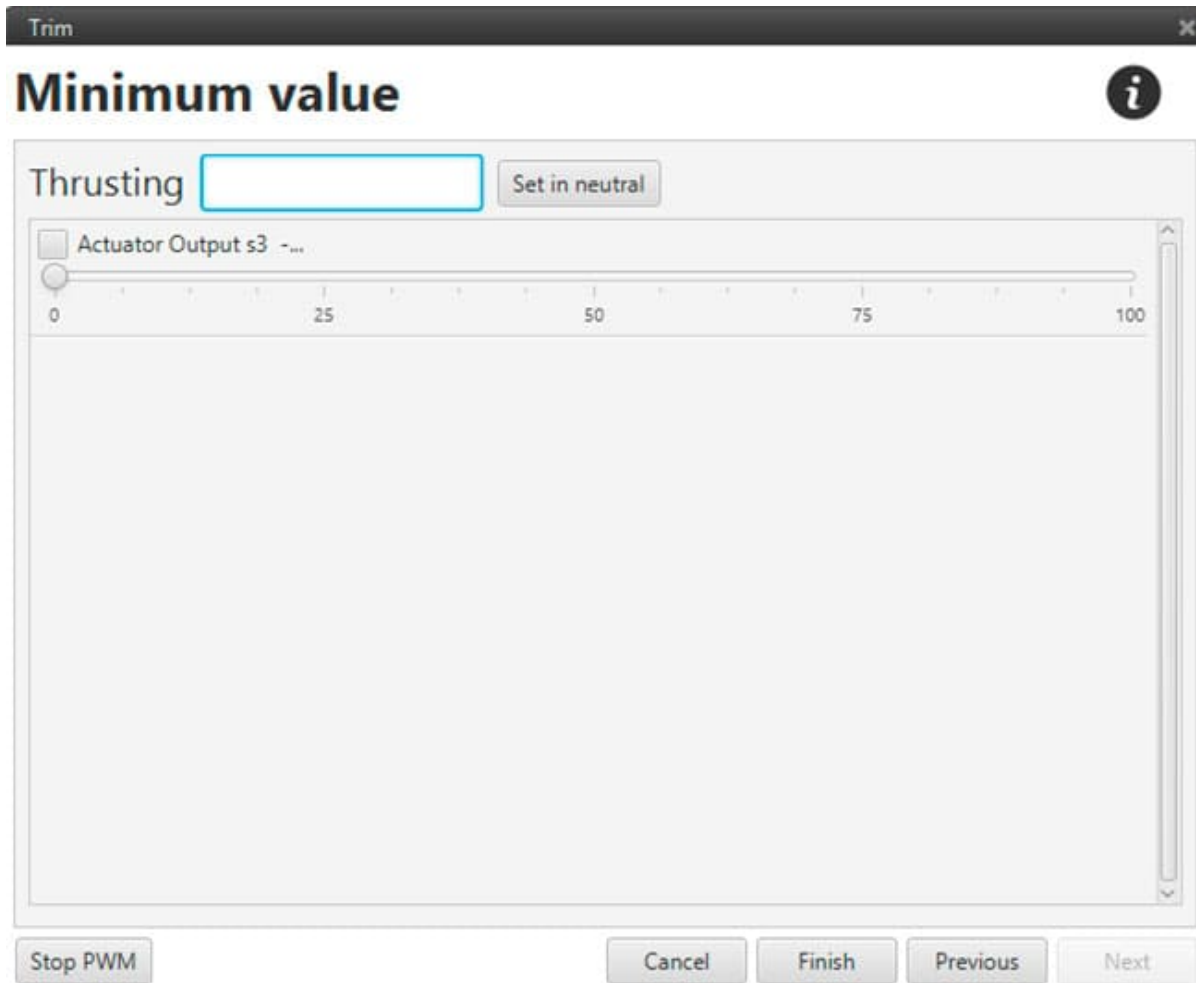
| Control Channel | Actuator Output |
|-----------------|----------------------------|
| Pitching | Actuator Output s1 - PWM 0 |
| | Actuator Output s2 - PWM 1 |
| Thrusting | Actuator Output s3 - PWM 2 |
| Rolling | Actuator Output s1 - PWM 0 |
| | Actuator Output s2 - PWM 1 |

Buttons: Stop PWM, Cancel, Finish, Previous, Next

Trim Wizard – Trim Wizard – Select Output

Once a servo is trimmed for a defined control channel, it will not be displayed for trim in other control channels.

- **Value:** Enter the maximum and minimum actuator position for the control variable. These values correspond to the components of S, for example, a maximum deflection of 20 degrees will correspond to a 75 % pulse and a minimum of -20 a 25%. These limits don't have to be the same as the mechanical limits of the actuator establish in the first screen of the wizard.



Trim Wizard – Trim Wizard – Min/Max Values

When the process is finished, a summary display appears, showing the user the graphs relating all the information provided, as the on the bottom part of the main display.

7.2.3.1.2 Logical

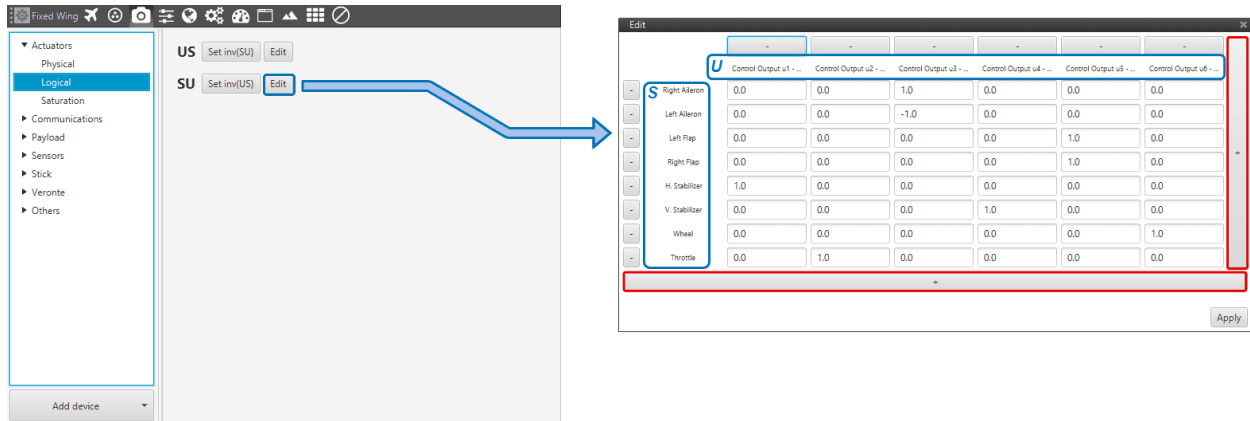
7.2.3.1.2.1 SU & US Matrices

SU and US are two matrices (inverse of one another, respectively) which contain the relationship between actuator outputs S and control outputs U , i.e. the influence of each control channel on each actuator output. The option of having a configurable SU matrix allows Veronte autopilots to control any type of vehicle, independently of how its control surfaces/devices are set and adjusted.

U is a vector which contains the control outputs of the platform, e.g. **pitch**, **roll**, **yaw**, **throttle**, etc. The values of U do not represent a physical variable. They are instead fictitious variables which are used in the control algorithm. What is actually applied to the system are the actuators movements, i.e. the PWM signals sent to the servos, which are mapped in the S vector – see [Devices - Actuators - Physical](#) for more information. The relation between S and U is essential for the right attitude control of the platform.

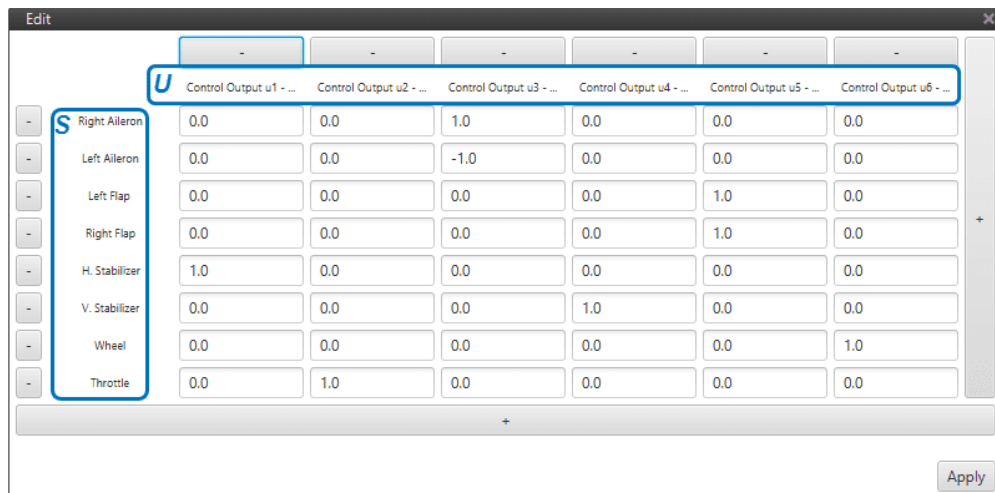
In order to define SU , the user needs to go to **Devices -> Actuators -> Logical** and click on *Edit* – see Figure below. A pop-up window will open containing the matrix. Control outputs U are placed on the columns and actuator outputs

S on the rows. Clicking on the '+' sign allows the user to add a new U or S , adding a new column or row will appear, respectively.



SU Matrix Menu

Definition of US is also available, but it is recommended to go with the first as it is more intuitive to define. Let's consider the SU of a fixed-wing aircraft – see the Figure below. On the S vector there are 8 entries, i.e. 8 actuators; and on the U vector there are 6 control outputs.



SU Matrix for a Fixed-Wing aircraft

As we see in the Figure above, the user defines which actuators influence which control channels. For instance, only the actuator **throttle**, e.g. the engine – influences the control output U_2 : **thrusting**; and only the **horizontal stabilizer** influences U_1 : **pitching**.

Warning: Regarding the selection of the parameters of SU matrix:

The order of magnitude of the parameters should be respected at least for every row, i.e. every control channel, as long as there are no coupled control channels U .

Note: Good practice recommendations:

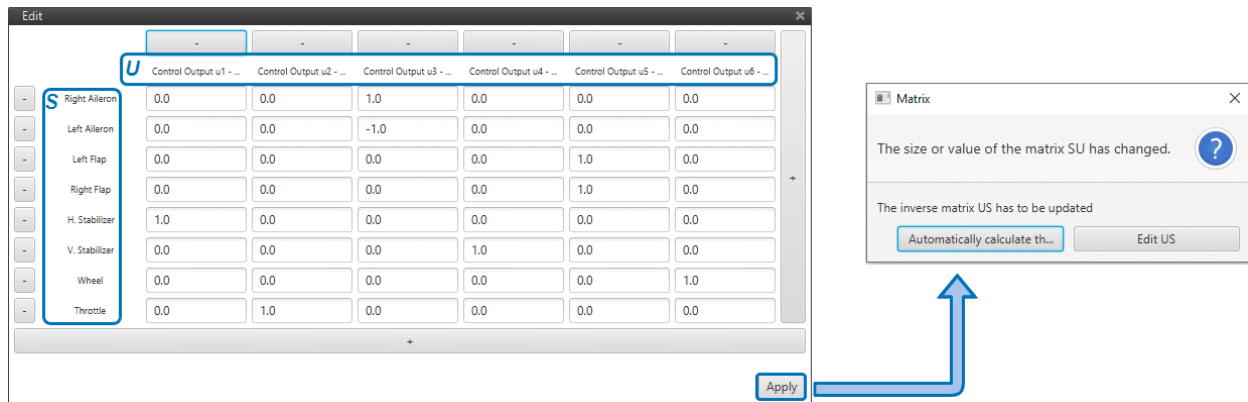
- Unitary values are recommended. Doing so, U will be equal to S . And if S has been defined according to a physical value – e.g. deflected angle, then control outputs can be easier to understand.

- The order of magnitude and the value of the SU parameters will not influence control algorithm calculations. But it will affect the control parameters, i.e. the control gains.
- It is recommended to keep the same order of magnitude for the whole matrix. That will allow an easier set up of a scaled version of the platform. Keeping the same SU and knowing the scaling factor, then the new control gains should be the old ones multiplied by that scaling factor. This practice can also be useful for transition to similar platforms.
- The SU vector should be defined accordingly in order to follow the sign convention for aerial navigation: a positive roll lowers the right wing, a positive pitch moves the nose up and a positive yaw moves the nose the right.

Important: Regarding SU parameters of opposite sign:

In the above Figure one can see that for control output U_3 : **rolling**, the right aileron indicates 1 and the left aileron -1. In other words, one moves opposite to the other. But this is a mere convention. Everything depends on the ailerons PWM to servo movement curves – defined in *Devices -> Actuators -> Physical*.

Coming back to the matrix SU menu, the user should click on *Apply* once it has finished. A pop-up window will appear (see Figure below) warning the user about the US matrix needing to be updated: it can be automatically updated from the SU , or introduced manually. It is advised to go with the first option.



SU to US

The US matrix of the fixed-wing aircraft is shown in the Figure below:

| | Right Aileron | Left Aileron | Left Flap | Right Flap | H. Stabilizer | V. Stabilizer | Wheel | Throttle |
|-------------------------------|---------------|---------------|-----------|------------|---------------|---------------|-------|----------|
| Control Output u1 - Pitching | -0.0 | -0.0 | -0.0 | -0.0 | 1.0 | -0.0 | -0.0 | -0.0 |
| Control Output u2 - Thrusting | -0.0 | -0.0 | -0.0 | -0.0 | -0.0 | -0.0 | -0.0 | 1.0 |
| Control Output u3 - Rolling | 0.5 | -0.5 | -0.0 | -0.0 | -0.0 | -0.0 | -0.0 | -0.0 |
| Control Output u4 - Yawing | -0.0 | -0.0 | -0.0 | -0.0 | -0.0 | 1.0 | -0.0 | -0.0 |
| Control Output u5 - Flaps | -7.850462E-17 | 1.1775693E-16 | 0.5 | 0.5 | -0.0 | -0.0 | -0.0 | -0.0 |
| Control Output u6 - Wheel | -0.0 | -0.0 | -0.0 | -0.0 | -0.0 | -0.0 | 1.0 | -0.0 |

US Matrix for a Fixed-Wing aircraft

Note that the fixed-wing example shown above has only one S only. Now, some more practical examples of real configurations – with more complex SU matrices – will be presented in order to complement the above explanation.

7.2.3.1.2.2 SU of a Flying Wing

Flying wings are tailless aircrafts, i.e. their only control devices are ailerons and a motor. So both the longitudinal (**pitching**) and lateral (**rolling**) control of the platform have to be controlled only with the ailerons – known as elevons in this kind of configurations. Therefore, two actuator channels S (left and right ailerons) will have to influence two control outputs U . When working like elevators, the elevons move together; when working like ailerons, one moves up as the other one moves down – this dual behaviour is highlighted in red in the Figure below.

| | Control Output u1 - Thrusting | Control Output u2 - Pitching | Control Output u3 - Rolling |
|---------------|-------------------------------|------------------------------|-----------------------------|
| Right Aileron | 0.0 | 1.0 | 1.0 |
| Left Aileron | 0.0 | 1.0 | -1.0 |
| Throttle | 1.0 | 0.0 | 0.0 |

SU Matrix of a Flying Wing

7.2.3.1.2.3 SU of a V-Tail Aircraft

The V-tail aircraft presented next has the same platform structure as the one shown above except for one difference: the tail of the aircraft is at an angle (v-shaped). That implies that **pitching** and **yawing** are controlled by two control surfaces, while for a regular configuration they are controlled by one separate control surface each. These control surfaces are known as ruddervators. When working like elevators, the ruddervators move together; when working like rudders, one moves up as the other one moves down – this dual behaviour is highlighted in red in the Figure below.

Regarding the rest of the actuators S , each of them only influence one single control output U – see the SU matrix below.

| | Control Output u1 - Pitching | Control Output u2 - Thrusting | Control Output u3 - Rolling | Control Output u4 - Yawing | Control Output u5 - Flaps | Control Output u6 - Wheel |
|------------------|------------------------------|-------------------------------|-----------------------------|----------------------------|---------------------------|---------------------------|
| Right Aileron | 0.0 | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 |
| Left Aileron | 0.0 | 0.0 | -1.0 | 0.0 | 0.0 | 0.0 |
| Left Flap | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 | 0.0 |
| Right Flap | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 | 0.0 |
| Right Stabilizer | 1.0 | 0.0 | 0.0 | -1.0 | 0.0 | 0.0 |
| Left Stabilizer | 1.0 | 0.0 | 0.0 | 1.0 | 0.0 | 0.0 |
| Wheel | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 |
| Throttle | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 |

SU Matrix of a V-Tail Aircraft

7.2.3.1.2.4 *SU* of a Quadcopter

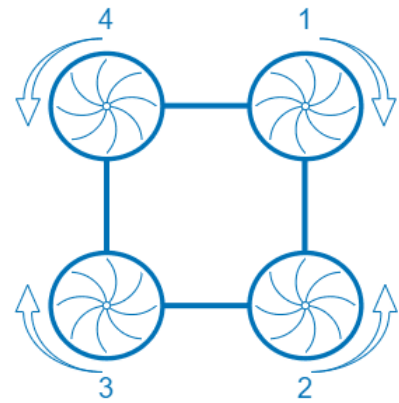
When it comes to a multicopter the *SU* matrix definition is not straightforward. Here we consider a quadcopter because it is the simplest of cases. Therefore, as its name suggests, it has 4 motors. The convention of the motor order in a quadcopter is the following: motor 1 is placed on the front right of the quadcopter, and the rest of them are placed following a clock-wise order – see the Figure below.

When defining the *SU* matrix of a quadcopter with symmetrical geometry:

- Control output *U*: **thrusting** will require all 4 motors to increase their rpms the same amount.
- Control output *U*: **pitching** will require for motors 1 and 4 to increase their rpms one particular amount, and motors 2 and 3 to decrease their rpms that same amount.
- Control output *U*: **rolling** will require for motors 1 and 2 to increase their rpms one particular amount, and motors 3 and 4 to decrease their rpms that same amount.
- Control output *U*: **yawing** will require for motors 1 and 3 to increase their rpms one particular amount, and motors 2 and 4 to decrease their rpms that same amount.

In multicopters, specially for symmetrical geometries, it is interesting to introduce the same influence values for the **pitching** and **rolling** control channels. That is so because when tuning the control gains (control parameters) they are usually very similar in both control channels.

| | Control Output u1 - Pitching | Control Output u2 - Thrusting | Control Output u3 - Rolling | Control Output u4 - Yawing |
|---------|------------------------------|-------------------------------|-----------------------------|----------------------------|
| Motor 1 | -1.0 | 1.0 | -1.0 | 1.0 |
| Motor 2 | 1.0 | 1.0 | -1.0 | -1.0 |
| Motor 3 | 1.0 | 1.0 | 1.0 | 1.0 |
| Motor 4 | -1.0 | 1.0 | 1.0 | -1.0 |



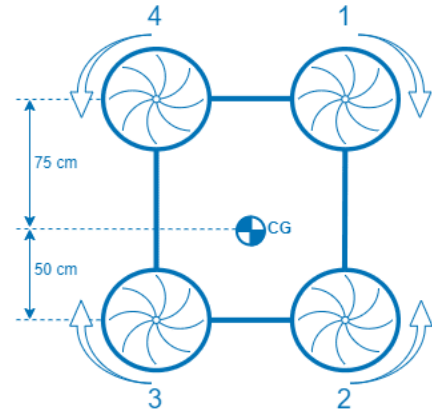
SU Matrix of a Quadcopter

If the considered quadcopter wouldn't have symmetrical geometry, the *SU* matrix needs to account for the lack of symmetry. In the Figure below, one can see that motors 1 and 4 are further ahead of the center of gravity (CG) of the platform than motors 2 and 3.

For a hover manoeuvre, the desired **pitching** moment is 0. Assuming the latter, i.e. the sum of all existing moments equal to 0, one finds the following relation $T_{back} = 1.5 \cdot T_{front}$, where T represents the force generated by the motors.

Therefore, one needs to define an influence value for the motors in the front (1 and 4) and then, multiply that same value by 1.5 for the motors in the back (2 and 3). One could choose the values 1 and 1.5, but in order for the **pitching** and **rolling** values to be similar, 0.8 and 1.2 are a better option – and they also satisfy the above relation.

| | Control Output u1 - Pitching | Control Output u2 - Thrusting | Control Output u3 - Rolling | Control Output u4 - Yawing |
|---------|------------------------------|-------------------------------|-----------------------------|----------------------------|
| Motor 1 | -0.8 | 1.0 | -1.0 | 1.0 |
| Motor 2 | 1.2 | 1.0 | -1.0 | -1.0 |
| Motor 3 | 1.2 | 1.0 | 1.0 | 1.0 |
| Motor 4 | -0.8 | 1.0 | 1.0 | -1.0 |



SU Matrix of a Quadcopter (non-symmetrical geometry)

7.2.3.1.2.5 *SU* of a VTOL Aircraft

Finally, a vertical take-off and landing (VTOL) aircraft configuration is covered. Main features of the aircraft are: V-shaped tail, a wing with ailerons only, one motor for plane mode, and 4 motors for quadcopter mode (VTOL). The recommended strategy for this kind of platform is to create 4 control channels *U* for the plane mode and 4 control channels *U* for the quadcopter modes.

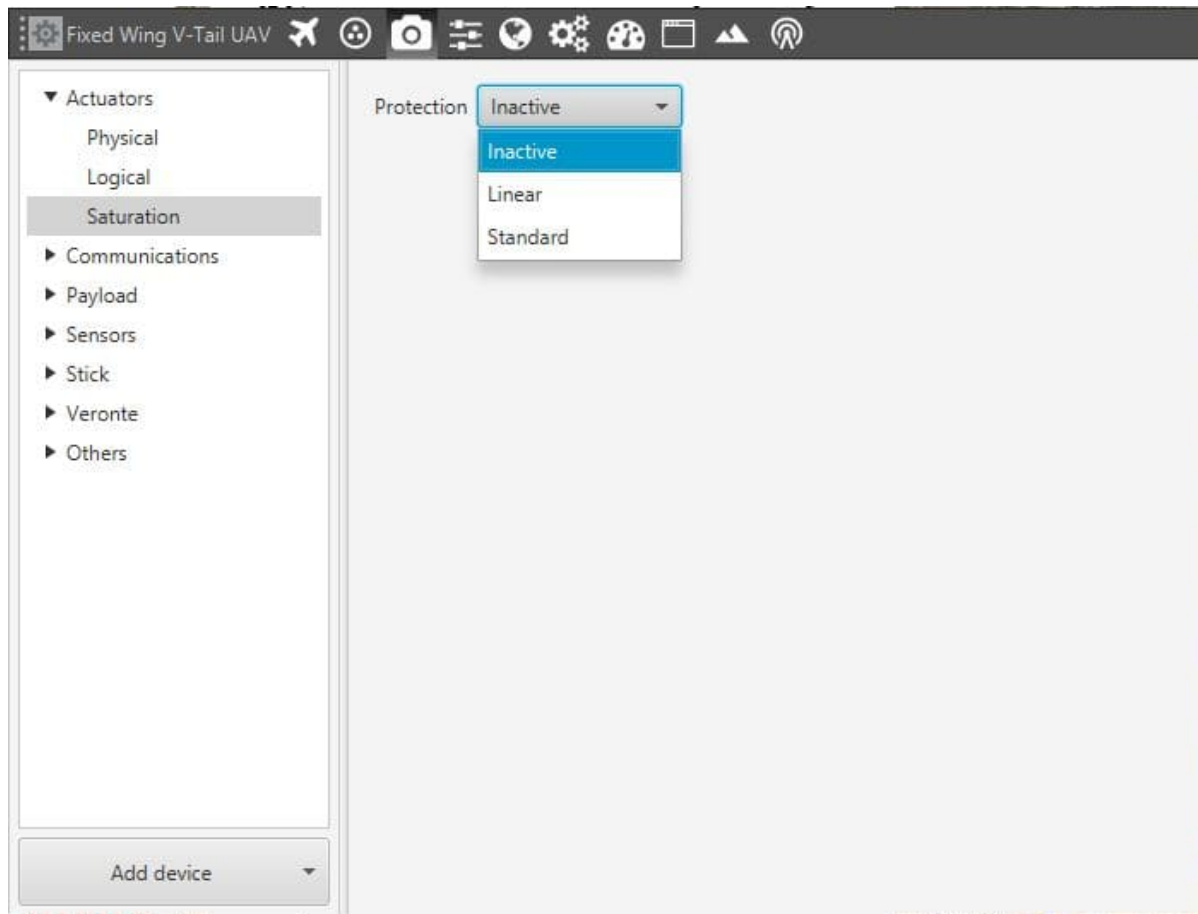
The *SU* matrix of the VTOL aircraft looks like:

| | Control Output u1 - Pitching (P) | Control Output u2 - Rolling (P) | Control Output u3 - Yawing (P) | Control Output u4 - Pitching (Q) | Control Output u5 - Throttle (Q) | Control Output u6 - Rolling (Q) | Control Output u7 - Yawing (Q) | Control Output u8 - Throttle (P) |
|----------------|----------------------------------|---------------------------------|--------------------------------|----------------------------------|----------------------------------|---------------------------------|--------------------------------|----------------------------------|
| Left Aileron | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| Right Aileron | 0.0 | -1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| Throttle | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 |
| Left Elevator | 1.0 | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| Right Elevator | 1.0 | 0.0 | -1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| Motor 1 | 0.0 | 0.0 | 0.0 | -1.0 | 1.0 | -1.0 | 1.0 | 0.0 |
| Motor 2 | 0.0 | 0.0 | 0.0 | 1.0 | 1.0 | -1.0 | -1.0 | 0.0 |
| Motor 3 | 0.0 | 0.0 | 0.0 | 1.0 | 1.0 | 1.0 | 1.0 | 0.0 |
| Motor 4 | 0.0 | 0.0 | 0.0 | -1.0 | 1.0 | 1.0 | -1.0 | 0.0 |

SU Matrix of a VTOL Aircraft

7.2.3.1.3 Saturation

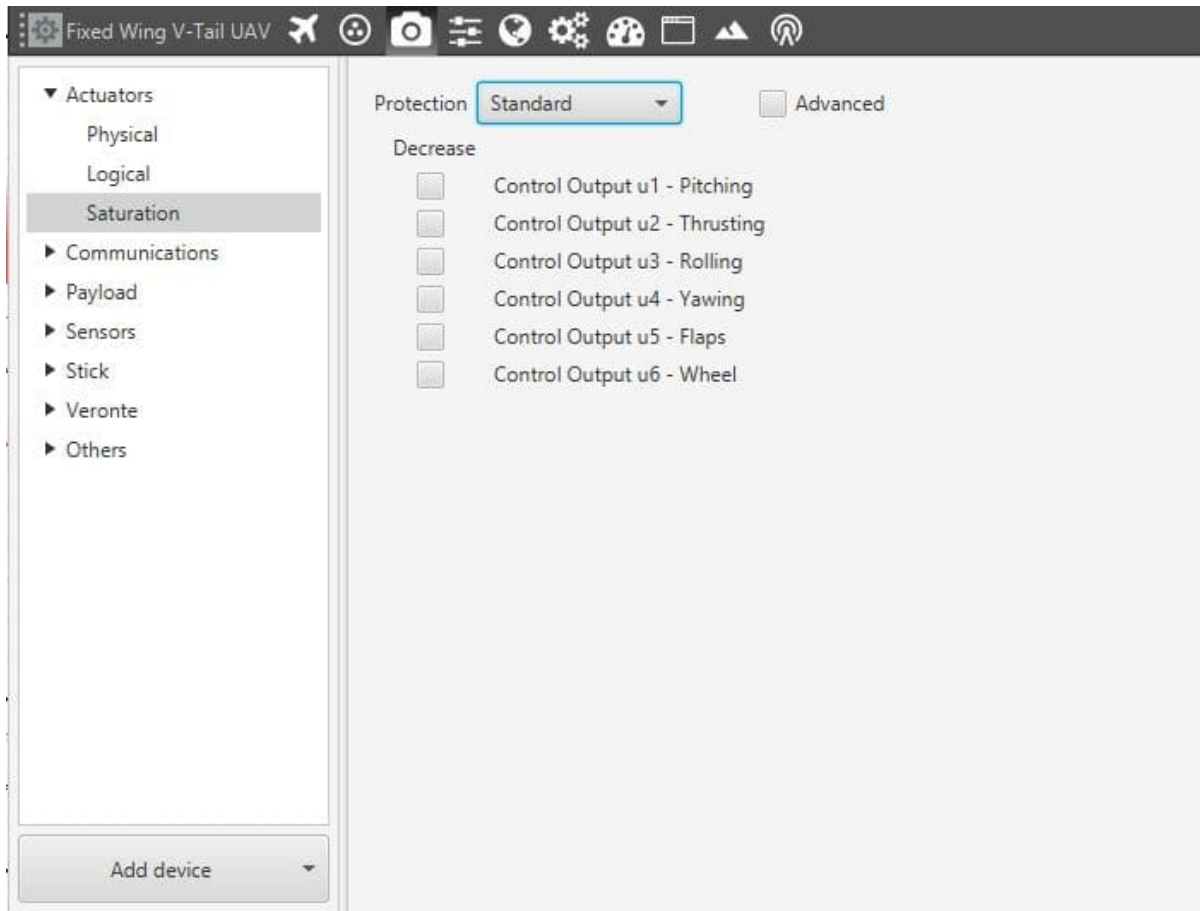
In this menu, the user can configure the behaviour of the platform when one or more of its actuators is/are in saturation state. The three options available are **Inactive** (the system does not respond to saturation), **Linear** (system affects all the actuators on the same way) and **Standard**, which can be found below.



Saturation Option for Actuators

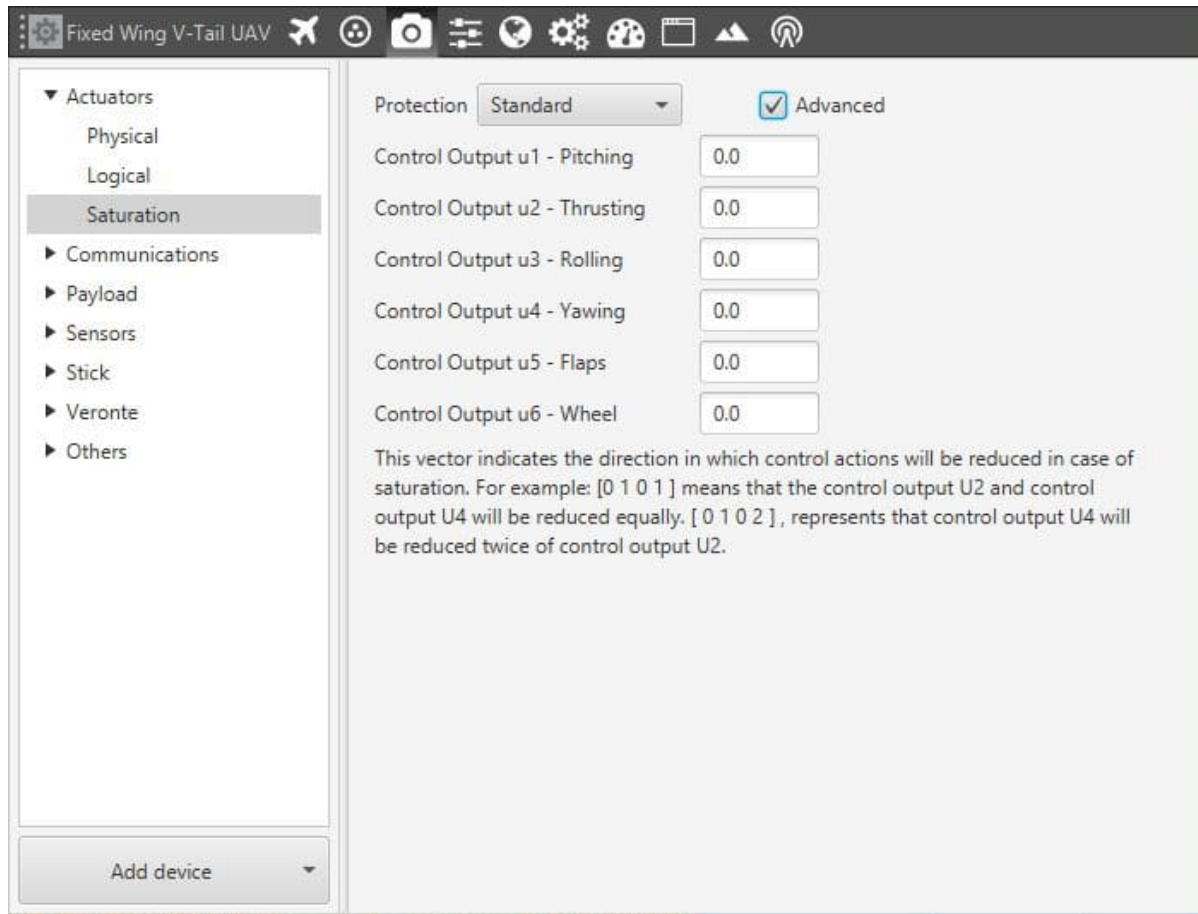
7.2.3.1.3.1 Standard Saturation

The system allows the user to choose which of the Controls is going to be affected if saturation is reached at any actuator. It can be chosen from 1 to all of them (which will be linear action).



Standard Saturation Menu

If clicking in the advance checkbox, a vector including all control outputs is generated, allowing for proportional control over the system when saturation happens. This tool is set for the user to have wider control over this feature if needed.

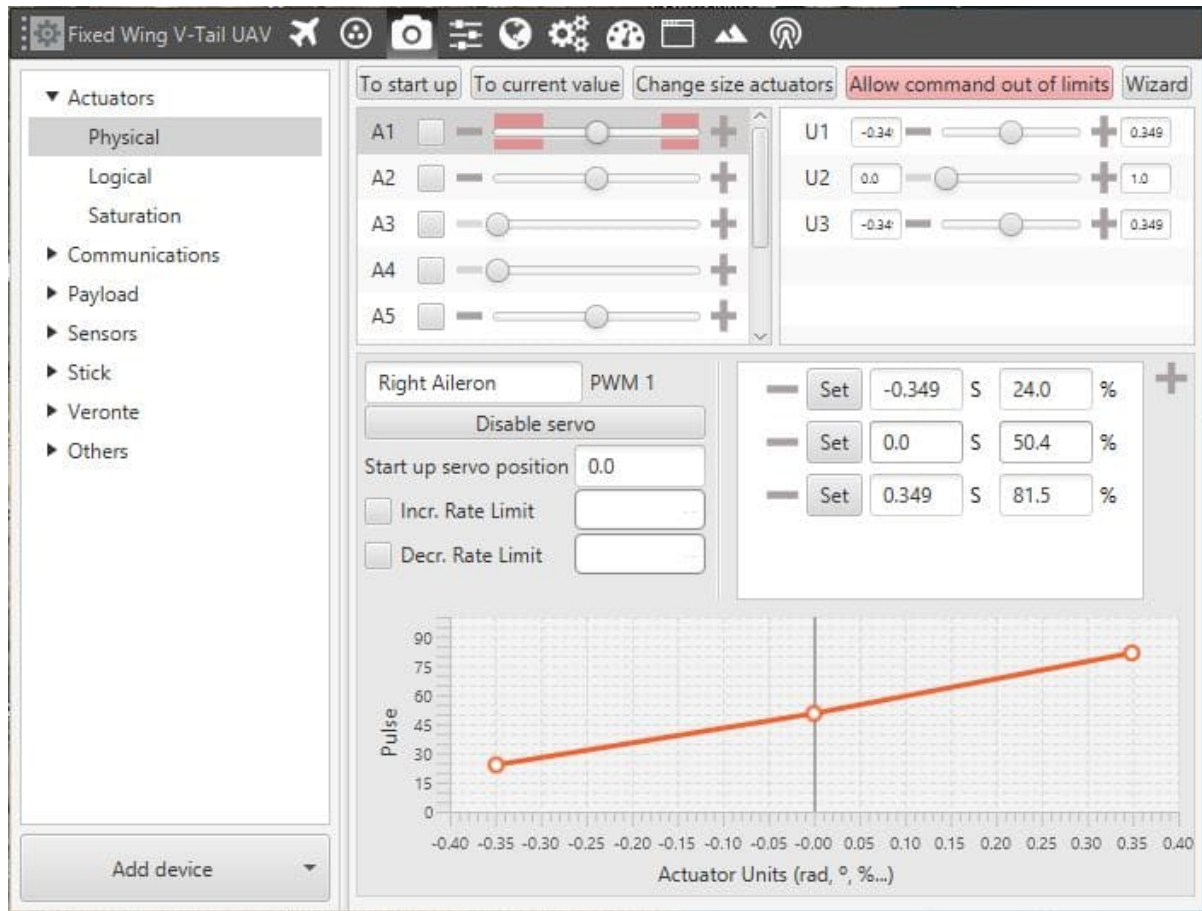


Advance Standard Saturation Mode

Inside this menu, the user can access the configuration for the servos.

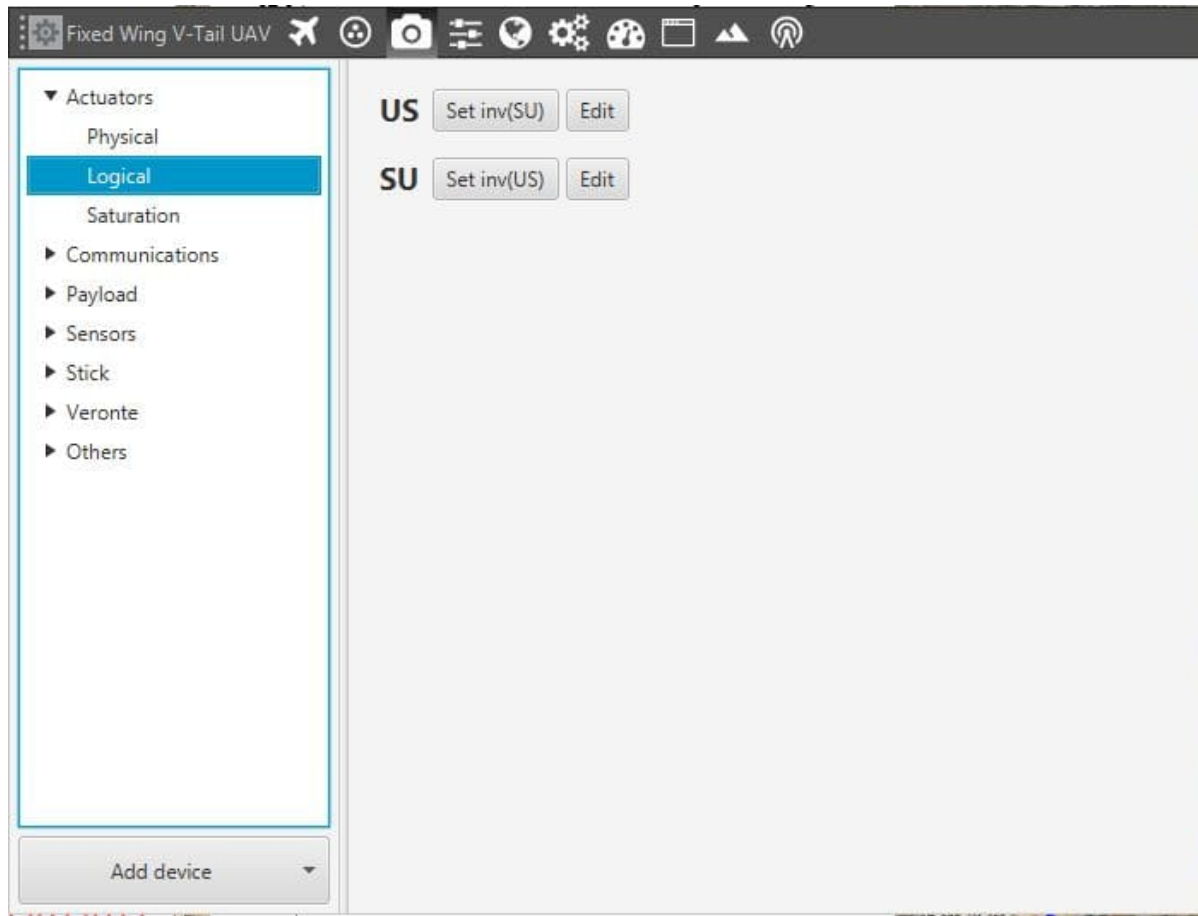
The first menu allows for actuators physical configuration, establishing maximum and minimum values, enabling or disabling actuators, giving them a corresponding angle associated to a pulse or using a Wizard display to do everything at once.

Note: Veronte allows configuring up to 32 actuators at the same time.



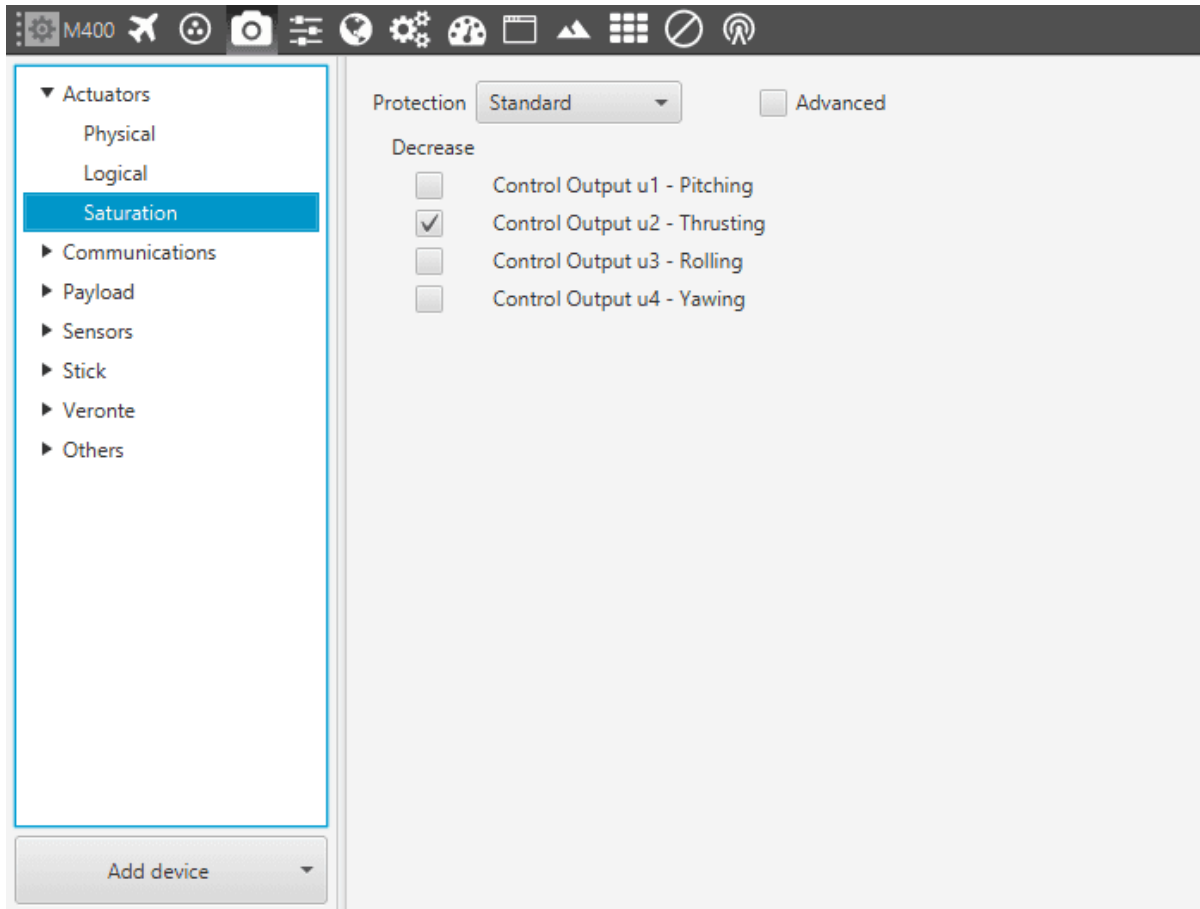
Physical Actuators Configuration

The second menu is the logical, where the US/SU matrices are configured. This defines the influence of each control channel on each control actuator.



Logical Actuators Configuration

The last configuration menu available is the Saturation one, where the user can choose between Saturation kind of protection and some more internal parameters.



Saturation Actuators Configuration

7.2.3.2 Communications

Communications menu covers the configuration for alternative communication channels, statistics and routing.

7.2.3.2.1 4G

Checking the **Enable** box will enable the use of 4G communication through the **Veronte LTE** Consumer/Producer in the *I/O Manager*.

7.2.3.2.1.1 ESIM

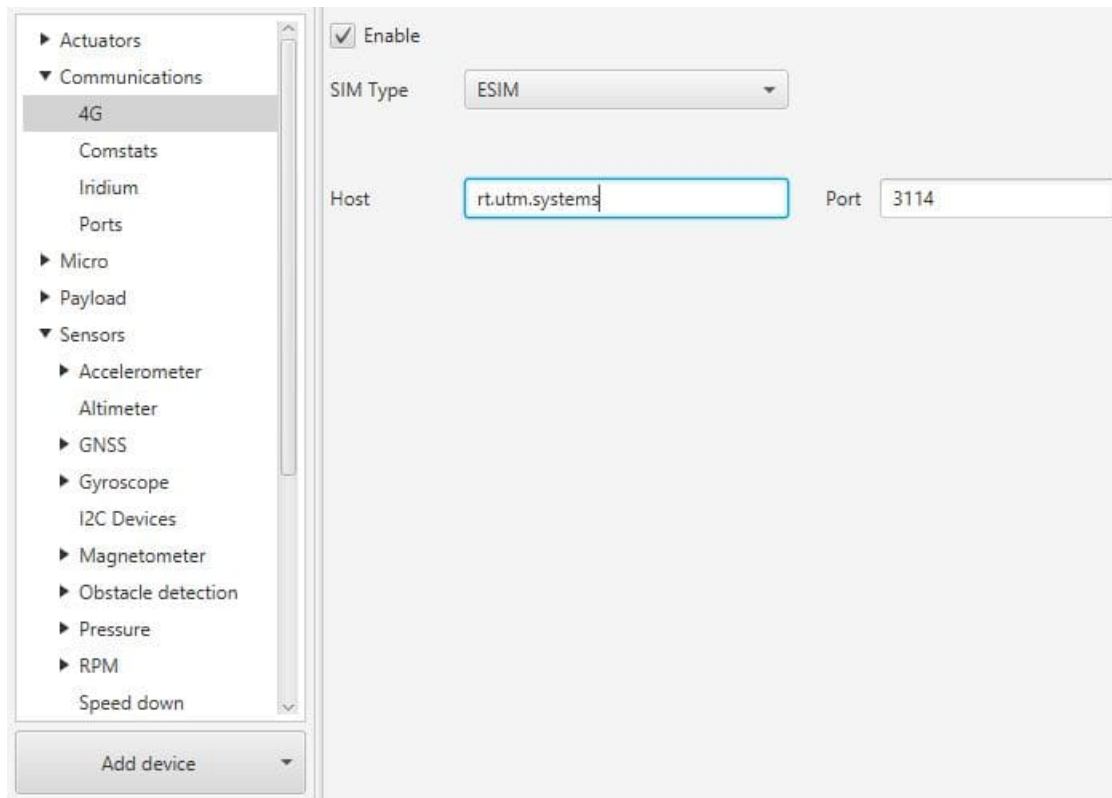
The embeded **ESIM** in Veronte allows the user to send and receive telemetry using a commercial data provider.

The connection between the air unit and the ground station is stablished through the **Veronte Cloud** server.

To connect with **Veronte Cloud** the following parameters have to be set:

- **Host:** rt.utm.systems
- **Port:** 3114

Host and Port can be changed if the used server differs from **Veronte Cloud**, but the communication protocol does not change.



The screenshot shows the configuration interface for the Veronte Autopilot. On the left, a sidebar menu lists various components: Actuators, Communications (expanded), 4G (selected), Comstats, Iridium, Ports, Micro, Payload, Sensors (expanded), Accelerometer, Altimeter, GNSS, Gyroscope, I2C Devices, Magnetometer, Obstacle detection, Pressure, RPM, and Speed down. At the bottom of the sidebar is an 'Add device' button. The main configuration area on the right has an 'Enable' checkbox checked. Below it, 'SIM Type' is set to 'ESIM' in a dropdown menu. The 'Host' field contains 'rt.utm.systems' and the 'Port' field contains '3114'.

ESIM Configuration for 4G

Note: In order to use the embedded **SIM** card, the contract with the data supplier needs to be done through **Embention**. Please contact sales@embention.com for more information on availability, coverage, suppliers and prices in your country.

7.2.3.2.1.2 SIM

If needed it is also possible to install a custom **SIM** card on Veronte. **PIN** number and **APN** (Access Point Name) of the **SIM** card provider must be defined before enabling the 4G communication.

Warning: Introducing the wrong **PIN** number may block the **SIM** card.

Warning: The installation of the SIM card must be done by Embention during the production of the unit. Please make sure to indicate the interest on using a Custom SIM card when ordering new Veronte units.

The screenshot shows the Veronte Autopilot configuration window. On the left is a sidebar menu with categories: Actuators, Communications (expanded), 4G (selected), Comstats, Iridium, Ports, Micro, Payload, Sensors (expanded), Accelerometer, Altimeter, GNSS, Gyroscope, I2C Devices, Magnetometer, Obstacle detection, Pressure, RPM, and Speed down. At the bottom of the sidebar is an 'Add device' button. The main configuration area on the right includes:

- An 'Enable' checkbox that is checked.
- A 'SIM Type' dropdown menu set to 'SIM'.
- An 'APN' text field containing 'em'.
- A 'PIN' field with four digits, all set to '0'.
- A 'Host' text field containing 'rt.utm.systems'.
- A 'Port' text field containing '3114'.

SIM Configuration for 4G

7.2.3.2.2 Comstats

The Comstats feature allows Veronte to make an estimation of the overall quality of the communication channel.

Veronte will send periodically (If enabled) a message with its current communication statistics (Packets sent and received per second). Then, any other Veronte unit can receive this information and compare against its own statistics to estimate the average amount of packets lost in the communication.

The results of this estimation can be monitored in variables **2000 (RX Packet Error Rate)** and **2001 (TX Packet Error Rate)**. These variables can be used to enable, for example, failsafe actions in case of degradation or loss of communications.

It is possible to configure the source or destination of the statistics, as well as the frequency at which the Comstats message is sent:

- **RX Auto:** Enabling this option will use the first remote AP found. If this option is disabled, the user must choose manually the address of the unit used for Comstats calculation.
- **TX:** When enabled, the unit will periodically send its Comstats message. The message can be sent to a specific unit or to all units on the network (Broadcast).

► Actuators

▼ Communications

4G

Comstats

Iridium

Ports

► Micro

► Payload

▼ Sensors

► Accelerometer

Altimeter

► GNSS

► Gyroscope

I2C Devices

► Magnetometer

► Obstacle detection

► Pressure

► RPM

Speed down

Add device

☒ RX Auto Veronte v4.5 1844

☒ TX

Address Broadcast

Period 0.1 s

Comstats Configuration Menu

Note: Enabling **Tx** will enable Veronte to send its Comstats message, but in order to compute Packet Error rate it's necessary to **receive** the Ts_x message from a different unit.

Warning: **Packet error rate** is a good indicator of the status of the communication, but it is not representative of the radiolink status. For monitoring the status of the radiolink RSSI Variables (820-822) shall be used instead. Depending on the configuration it is possible to have bad Error rates with good RSSI (overloaded radiolink) or good Error rates with bad RSSI (degraded communication with low load on radiolink). For the best results, it is recommended to use a combination of both statistics for failsafe automations.

7.2.3.2.3 Iridium

Checking the **Enable** box will enable the use of Iridium communication through the **Iridium** Consumer/Producer in the *I/O Manager*.

- **Synchronization time:** Transmission period.
- **Destination address:** Address of the destination Iridium module.

The screenshot shows the 'Iridium' configuration menu. On the left, a sidebar contains a tree view with categories: Actuators, Communications (expanded), 4G, Comstats, Iridium (selected), Ports, Micro, Payload, Sensors, Stick, Veronte, and Others. Below the sidebar is an 'Add device' button. The main configuration area on the right includes a checked 'Enabled' checkbox, a 'Synchronization time' input field set to '120.0' with a unit 's', and a 'Destination address' input field set to '0'.

Iridium Configuration Menu

7.2.3.2.4 Ports

Ports configuration allows the user to configure which communication ports (Commgr Ports in *I/O Manager*) will be used for communication.

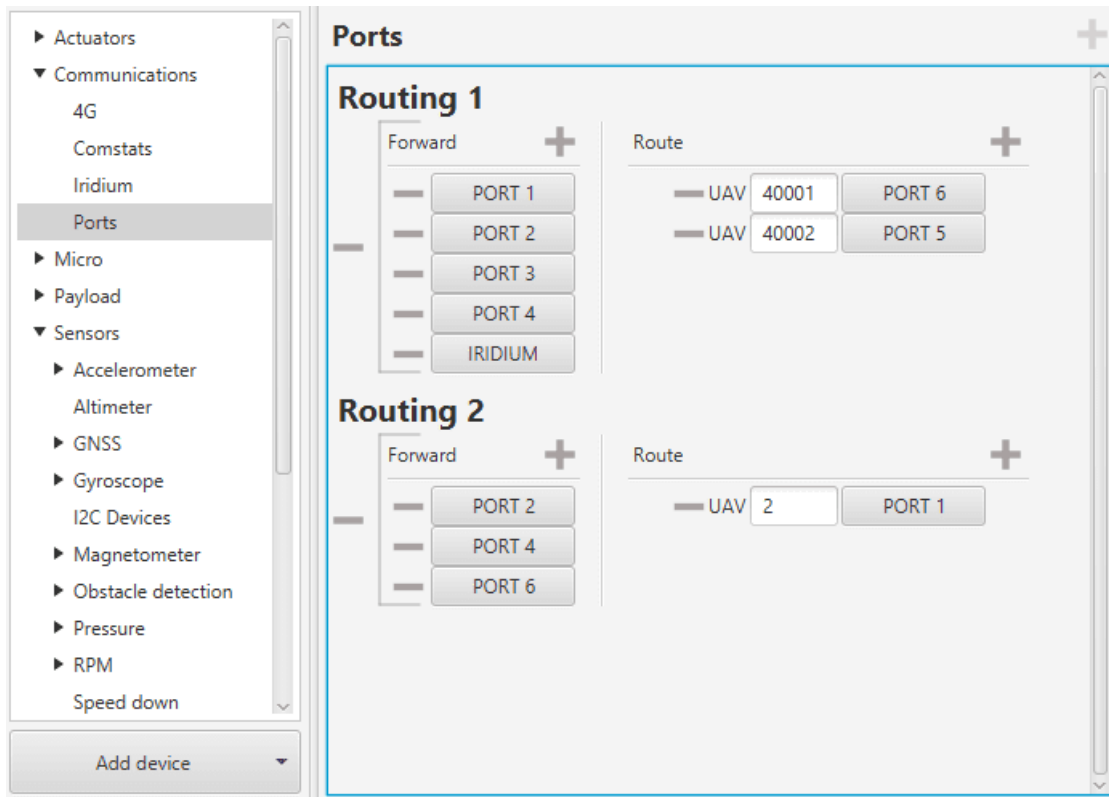
By using the **Route** feature, it is also possible for Veronte to act as a **Router**, redirecting any messages received (with a specific Address destination) through a specific Port.

Each of the different ports can be configured as either of the following options:

- **Forward:** Any messages generated by this unit (i.e. Telemetry or response messages to certain commands) will be sent through these ports.
- **Route:** Any messages received at any Commgr Port with the defined address will be re-sent through the defined port. It is possible to route several addresses through the same port, but is **not possible** to route the same address through several ports. Only the first configured port will be used. Routing also applies to messages generated by the unit for the defined address.

Note that the same **Port** cannot be used as **Forward** and **Route** at the same time.

It is possible to define up to 2 routing setups, which can be switched using the **Ports** action in the **Automations** menu. **Routing 1** will always be selected by default when booting Veronte.



Ports Menu

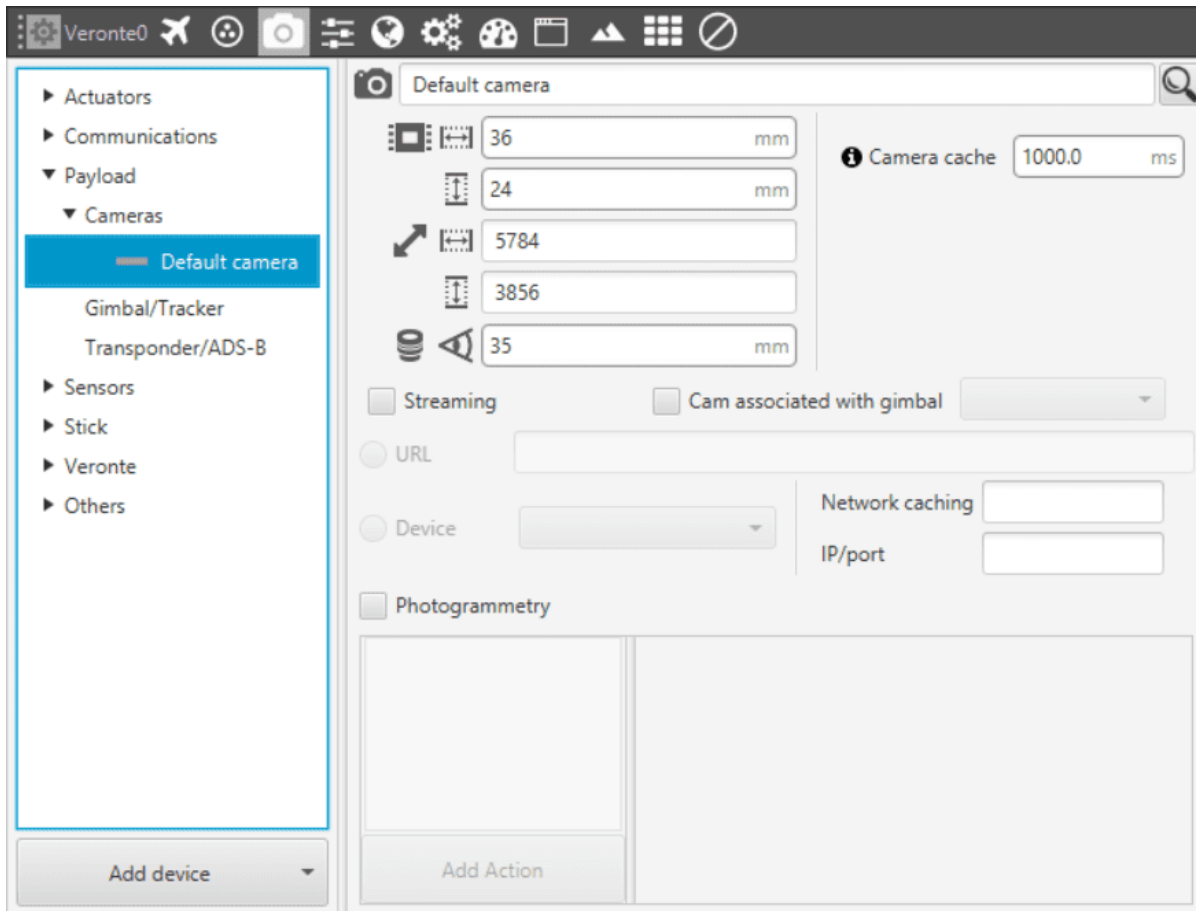
A practical example of the use of this menu are Veronte Ground configurations. These configurations will have configured a **Routing** of Address **2 (Pipe)** through **Port 1**. This way, any messages that are received through a Commgr Port (i.e. through Veronte LOS), will be re-routed through **Port 1 (USB)** and received by **Pipe Software**, including any messages generated by **Veronte Ground** itself.

Warning: An incorrect Port configuration can disable USB communication. If this happens, Veronte will not be able to be detected through **PIPE Software**. If this is the case, please visit our Troubleshooting section on how to **recover USB communication**.

7.2.3.3 Payload

7.2.3.3.1 Camera

When adding a Device, the user can choose adding a Camera. This will create a default Camera under **Devices - Payload - Camera**.

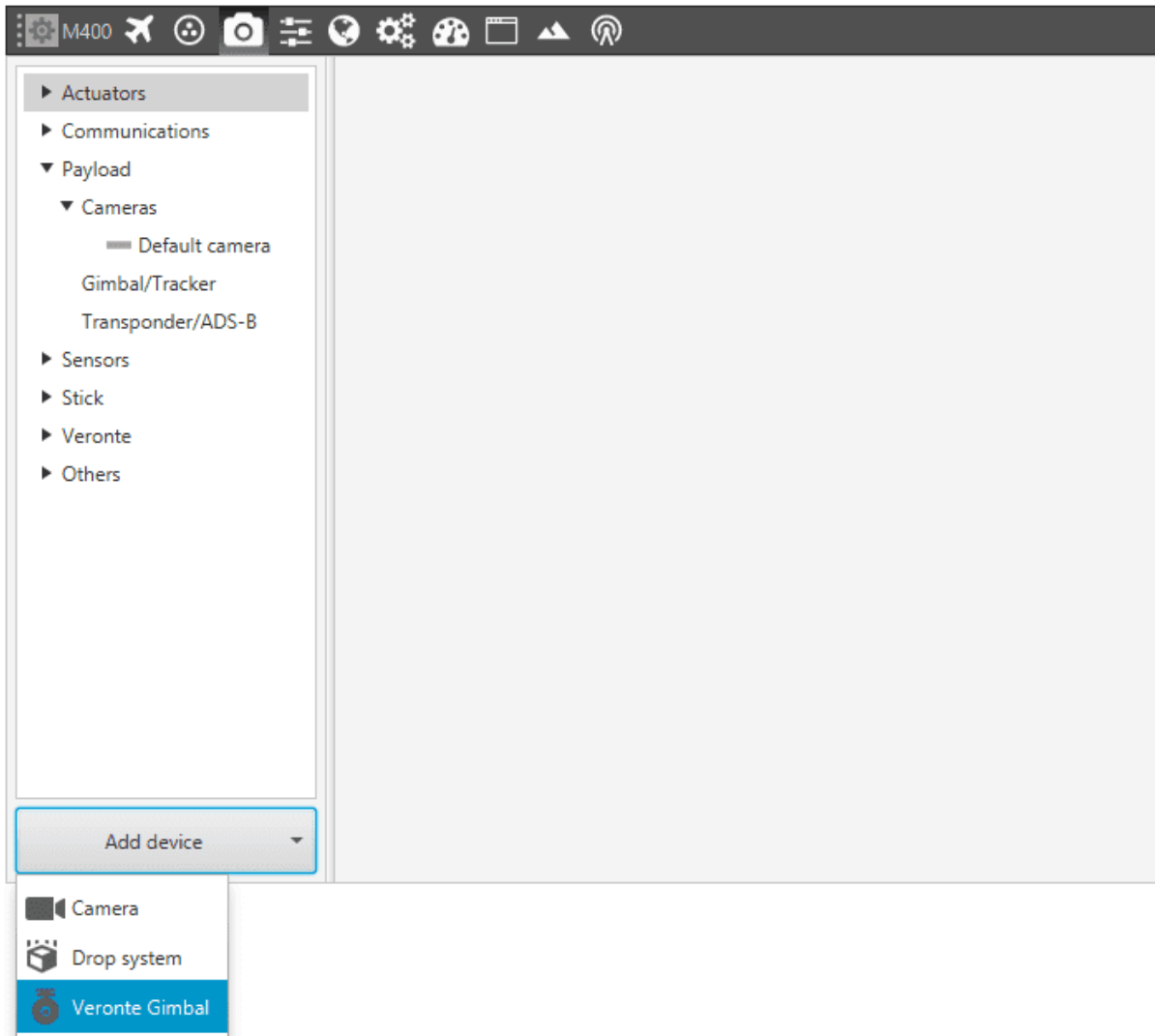


Camera Menu - Configuration Parameters

- **Name.** Put a name to the camera define or use the lens on the right to choose from the predefined list of cameras. Making this will automatically establish the following values (which in other way would have to be define manually):
 - **Sensor.** Defines the camera sensor width and height in mm.
 - **Resolution.** Defines the camera resolution width and height.
 - **Lens.** Defines the focal length from the camera in mm.
 - **Camera cache.** Defines the cache time used to play the selected camera on the gimbal widget.
- **Streaming.** Section used to define a streaming service with which to configure a payload system to be able to have its video in Veronte Pipe, particularly in Workspace - Gimbal widget.
 - **URL.** Cameras whose protocol goes through Ethernet are configured introducing its URL in this field.
If it is from a Gimbal device, it is important to be sure to configure the field *Cam associated with gimbal* if the user wants to move the Gimbal from the Widget.
 - **Device.** If the Camera is a system connected directly to the computer it will be selected from here. Generally speaking, Device 0 is linked to the Screen, Device 1 to the internal camera (if portable computer).
 - **Photogrammetry.** The actions undertaken in a Photogrammetry mission can be defined here, following the same possibilities as in Automations - Actions. A maximum of 4 Actions can be defined.

7.2.3.3.1.1 Veronte Gimbal Wizard

Veronte Pipe has an option which allows the user to configure from a basic menu all needed parameters for a Veronte Gimbal (in any of its four models).



Camera Menu - Veronte Gimbal Wizard

The parameters to be configured are:

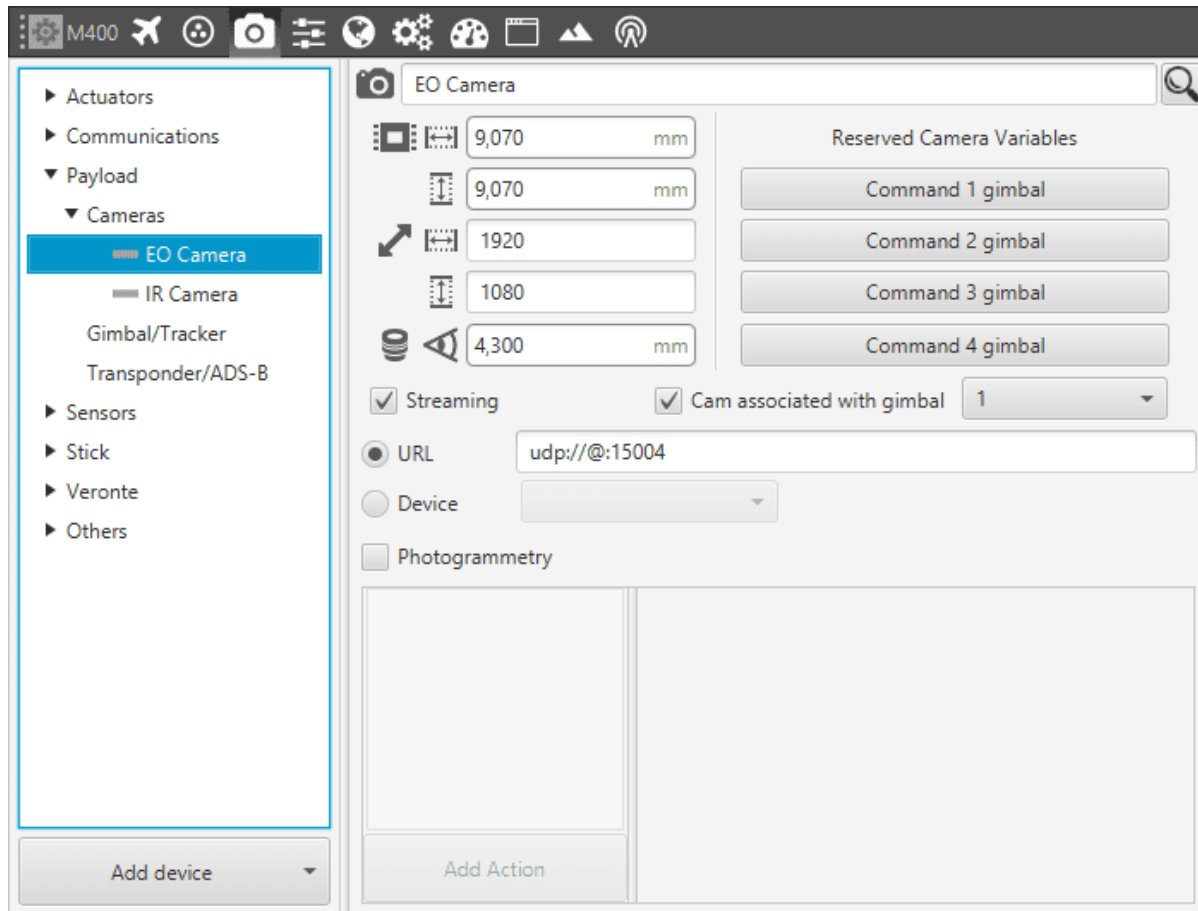
- **Type.** Defines the Veronte Gimbal model which is being configured:
 - Veronte Gimbal
 - Veronte Gimbal SC
 - Veronte Gimbal PRO
 - Veronte Gimbal PRO SC
- **Gimbal axis type.** Defines the angles that Veronte will be controlling from the payload system from a Combination of Pan (Z axis, the same as Yaw), Tilt (Y axis, the same as Pitch) and Roll. The two options are as seen below:

- Pan & Tilt.
- Roll & Tilt.
- **CAN ID.** Chooses the CAN which will be configured for using the Gimbal between A and B.
- **Camera URL.** Set the computer IP for the wizard to create the corresponding links between Gimbal and PC.
- **Automations.** Defines the name of the group of automations which will be created after the Wizard.

The screenshot shows a window titled "Veronte gimbal" with a close button in the top right corner. The main title "Veronte Gimbal" is displayed at the top. The configuration is divided into two columns. The left column contains three dropdown menus: "Type" set to "Veronte Gimbal SC", "Gimbal axis type" set to "Pan Tilt", and "CAN ID" set to "B". The right column, under the heading "Camera URL", contains a "Computer IP" field with an "@" symbol, a "Primary url" field with the text "udp://@:8008", and an "Automations" section with a "Group name" field set to "Veronte Gimbal". An "Apply" button is located at the bottom right of the window.

Camera Menu - Veronte Gimbal Wizard - Configuration Parameters

When clicking on apply, the wizard will automatically generate the CAN Messages, the cameras needed (either EO or IO depending on the model), the Gimbal configuration and a set of Automations for its use.



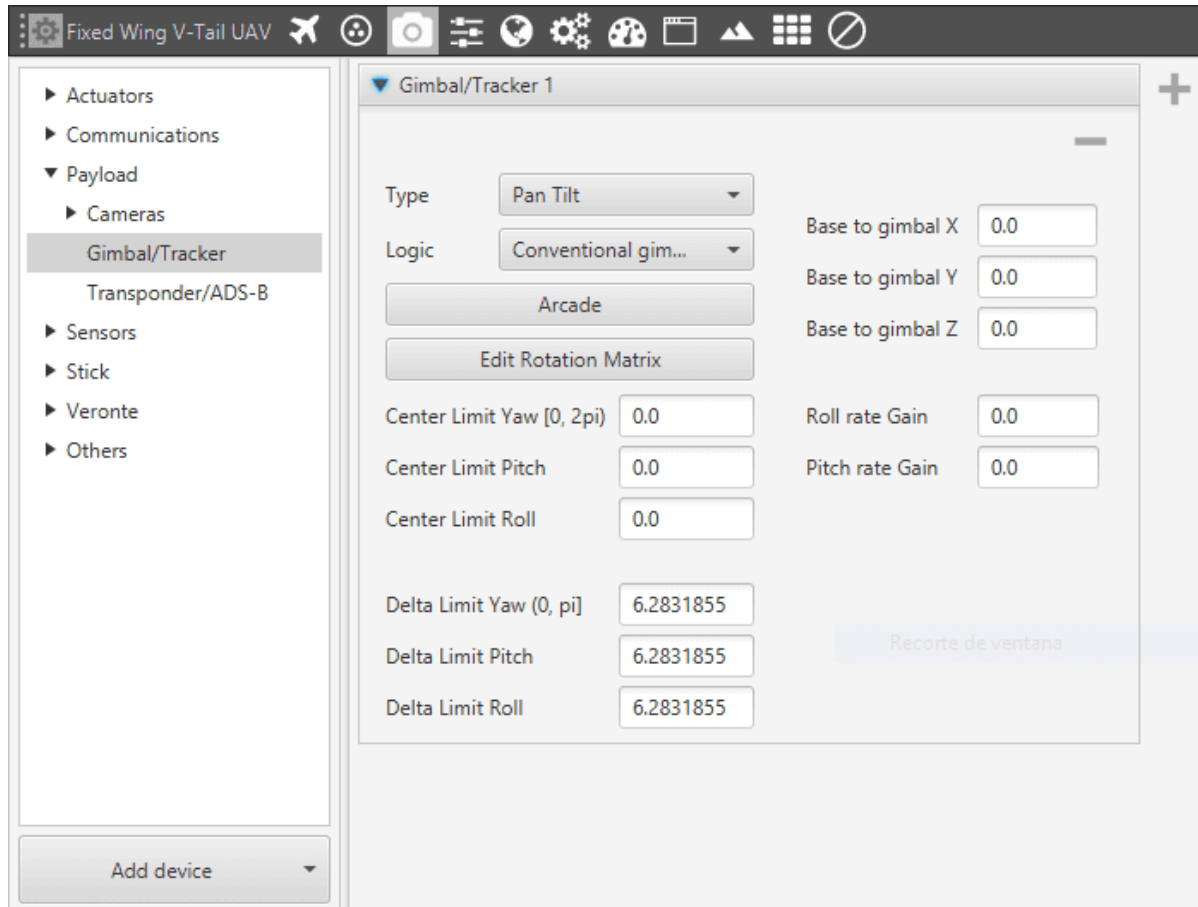
Camera Menu - Veronte Gimbal Wizard Results

Warning: If the camera has not been properly configured or the user wants to delete the previous configured model there are 4 menus to clear:

- CAN A/B - Telemetry - TX
- Devices - Payload - Cameras & Gimbal/Tracker
- Automations - Group Defined

7.2.3.3.2 Gimbal/Tracker

This menu allows the user to configure a Gimbal Camera or a Tracker Antenna. From here the user only needs to define the movements the system has (from predefined combinations of Pan, Tilt and Roll), its logic, a distance vector and the number of systems connected (up to a maximum of 2).



Gimbal/Tracker Menu - Configuration Parameters

- **Type.** Defines the angles that Veronte will be controlling from the payload system from a Combination of Pan (Z axis, the same as Yaw), Tilt (Y axis, the same as Pitch) and Roll. The three options are as seen below:
 - Pan & Tilt.
 - Pan, Roll & Tilt.
 - Roll & Tilt.
- **Logic.** Defines the kind of payload system configured.
 - **Conventional gimbal.** This option writes over the variables Joint 1-3 Gimbal 1-3 which are used later to configure the control and stabilization of the camera from Veronte.
 - **Self-stabilized gimbal.** The payload system only needs movement inputs and the variables mentioned will have no output.
- **Base to gimbal Vector.** Defines the vector that joins the Veronte Autopilot controlling the payload system and the payload system itself, in Veronte body axes.

- **Arcade.** Configures the Arcade control of the payload system. For a new variable to be controlled, click on the top right “+” icon.
 - **Var.** Displays the variable over which the Arcade control will have its effect.
 - **Gain.** Value which multiplies the Var. to obtain the control value.
 - **DBand X.** Creates a non-effect area in which the Joystick won’t input control values. DBand X is applied to both left and right on the X axis. Value used to stop the stick from inputting control when at rest.
 - **DBand Y.** Same as DBand X but applied to the Y axis.
 - **Control Output.** Defines the value for the control output.

| Var | Gain | DBand X | DBand Y | Control Output |
|---------------------|------|---------|---------|----------------|
| Joint 1 of Gimbal 1 | 1.0 | 0.02 | 0.0 | 1 |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |

Gimbal/Tracker Arcade - Configuration Parameters

Any further configuration will strongly depend on the payload system selected and its control signals. Check [Veronte's Gimbal manual](#) as a reference.

7.2.3.3.3 Transponder

Veronte is compatible with:

- Sagetech
- uAvionix
 - Mode S Transponder ping 200Sr/200S
 - Mode S Transponder ping 20S

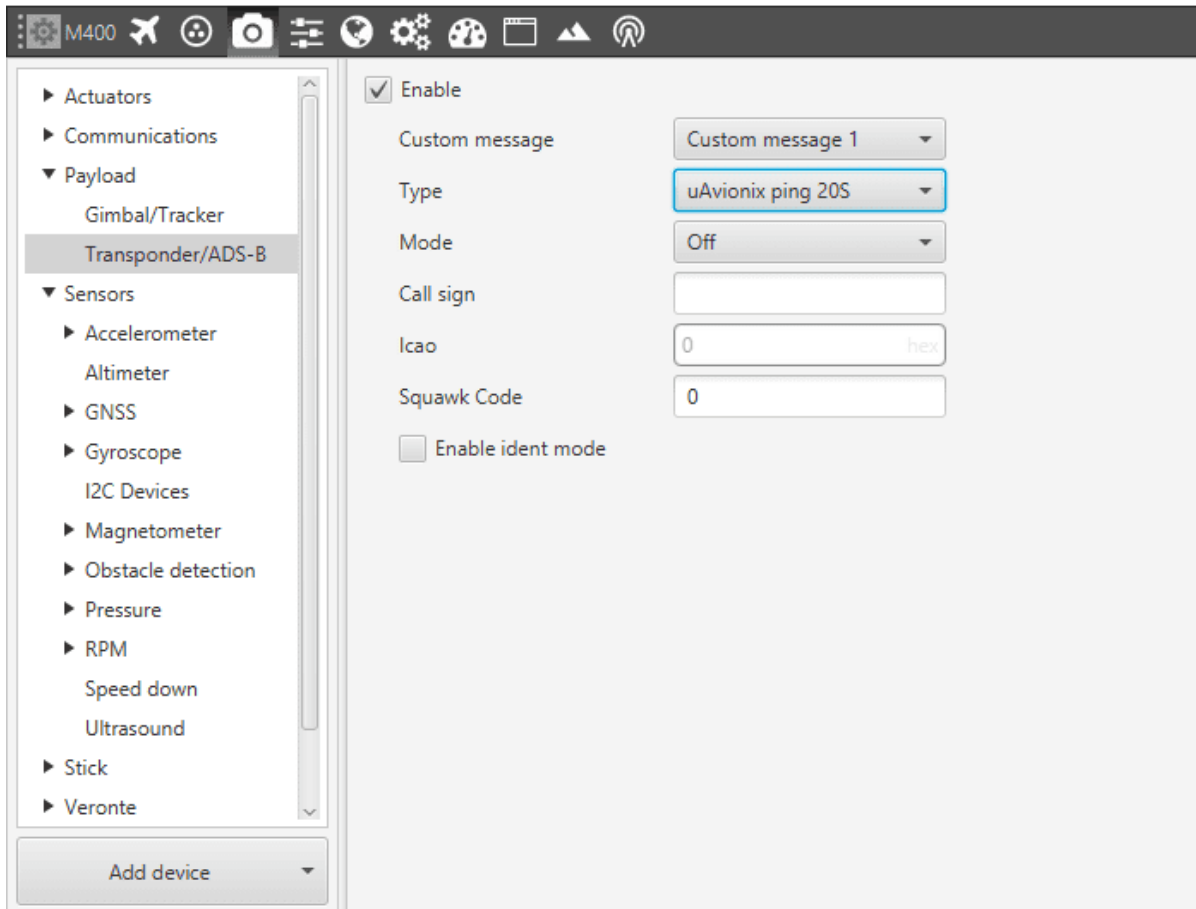
The configuration displayed below is for a Transponder.

1. The first step to configure it is to go to **Device – Payload – Transponder/ADS-B** and check the enable button. Once there, select the **Custom message** to be used for the information exchange, since it will automatically be filled with the information that the Transponder needs.
2. Choose the model already mentioned (uAvionix ping20S)
3. Switch to the desired **Mode**
 - **OFF:** Transponder switched off

- **STANDBY:** Transponder will not respond to interrogation.
- **ON:** Replies to interrogations with 4-digit squawk code.
- **ALT:** Replies to interrogation with altitude information.
- **Enable Ident Mode:** UAV identification under ATC request (only available for ON and ALT modes)

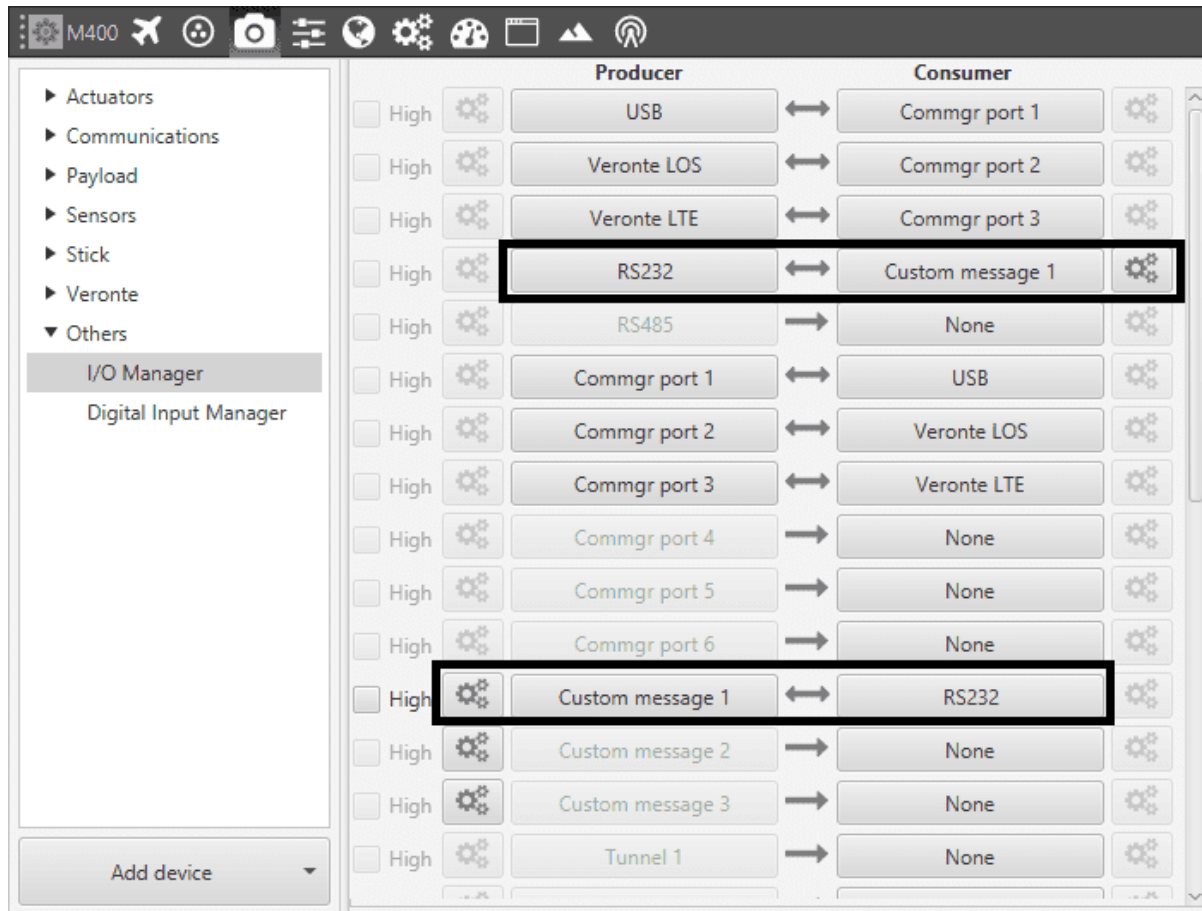
4. Introduce the **Squawk Code** provided.

There is an example configuration in the following picture from uAvionix ping 20S.



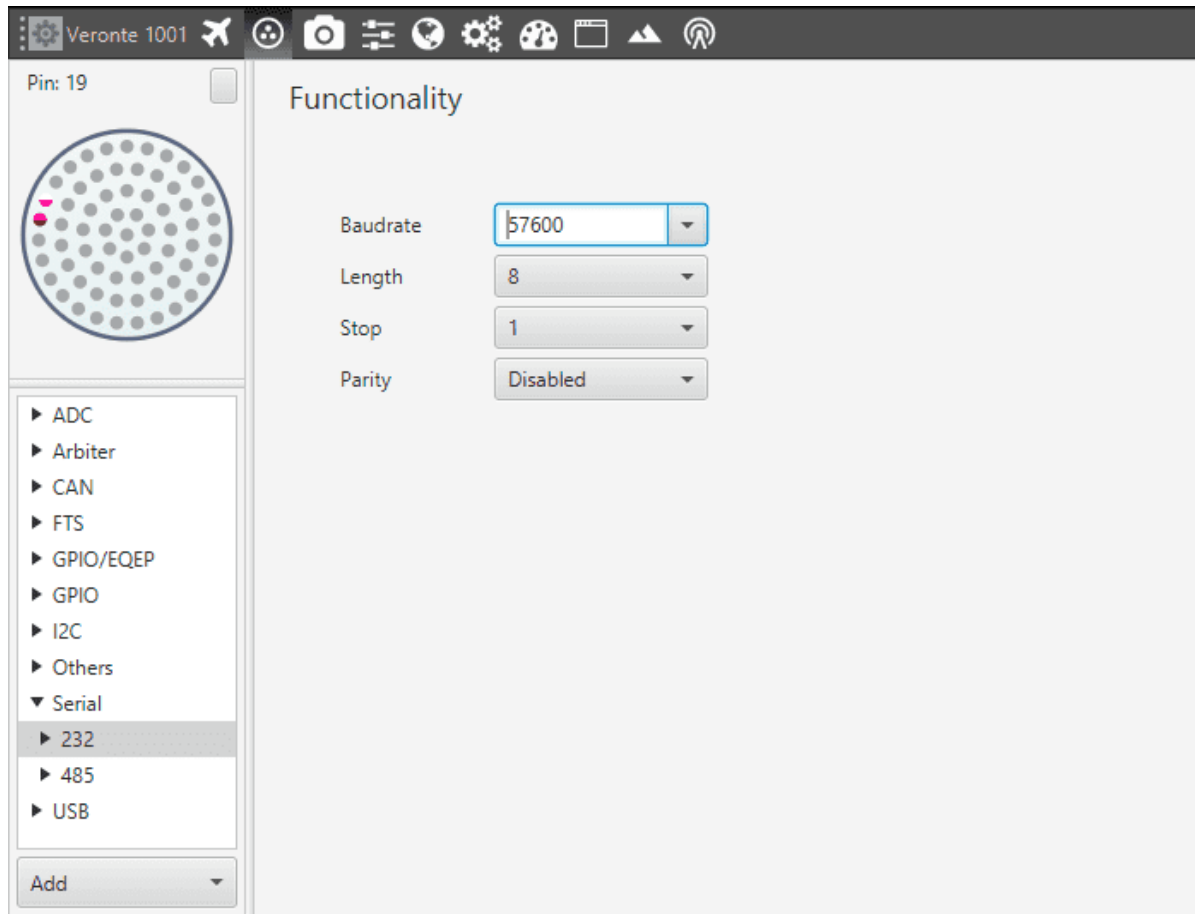
Transponder Configuration Parameters

Once the transponder is configured, go to **Devices – Others – I/O Manager**. Create a bind-bidirectional communication RS232/RS485 (in the example, through RS232) and assign to it the Custom Message selected before.



RS232 Message Reception

Note that either RS232 or RS485 must be configured with the corresponding Baudrate from the user transponder in **Connections – Serial – 232/485**.



RS232 Configuration Menu

7.2.3.3.4 ADS-B

Veronte is compatible with:

- Sagetech
- uAvionix
 - ADS-B Transceiver ping 1090/1090i
 - ADS-B Transceiver ping 2020/2020i
 - ADS-B Receiver ping Rx

The configuration displayed below is for an ADS-B.

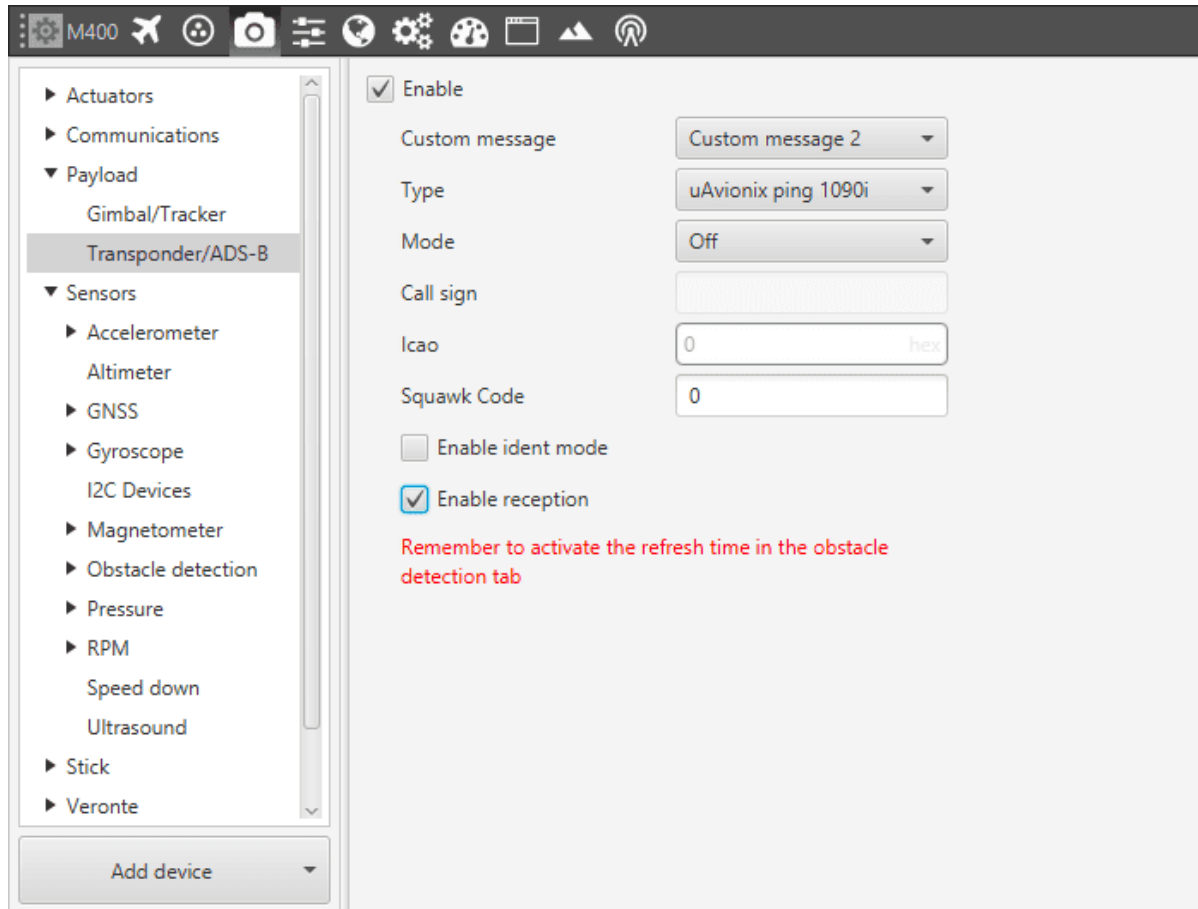
1. The first step to configure it is to go to **Device – Payload – Transponder/ADS-B** and check the enable button. Once there, select the **Custom message** to be used for the information exchange, since it will automatically be filled with the information that the ADS-B needs.
2. Choose the model already mentioned (uAvionix ping20S)
3. Switch to the desired **Mode**
 - **OFF:** Transponder switched off
 - **STANDBY:** Transponder will not respond to interrogation.

- **ON:** Replies to interrogations with 4-digit squawk code.
- **ALT:** Replies to interrogation with altitude information.
- **1090ES:** ADS-B transmit is always enabled when a 6-digit ICAO code is entered.
- **Enable Ident Mode:** UAV identification under ATC request (only available for ON and ALT modes)

4. Introduce the **Squawk Code** provided.

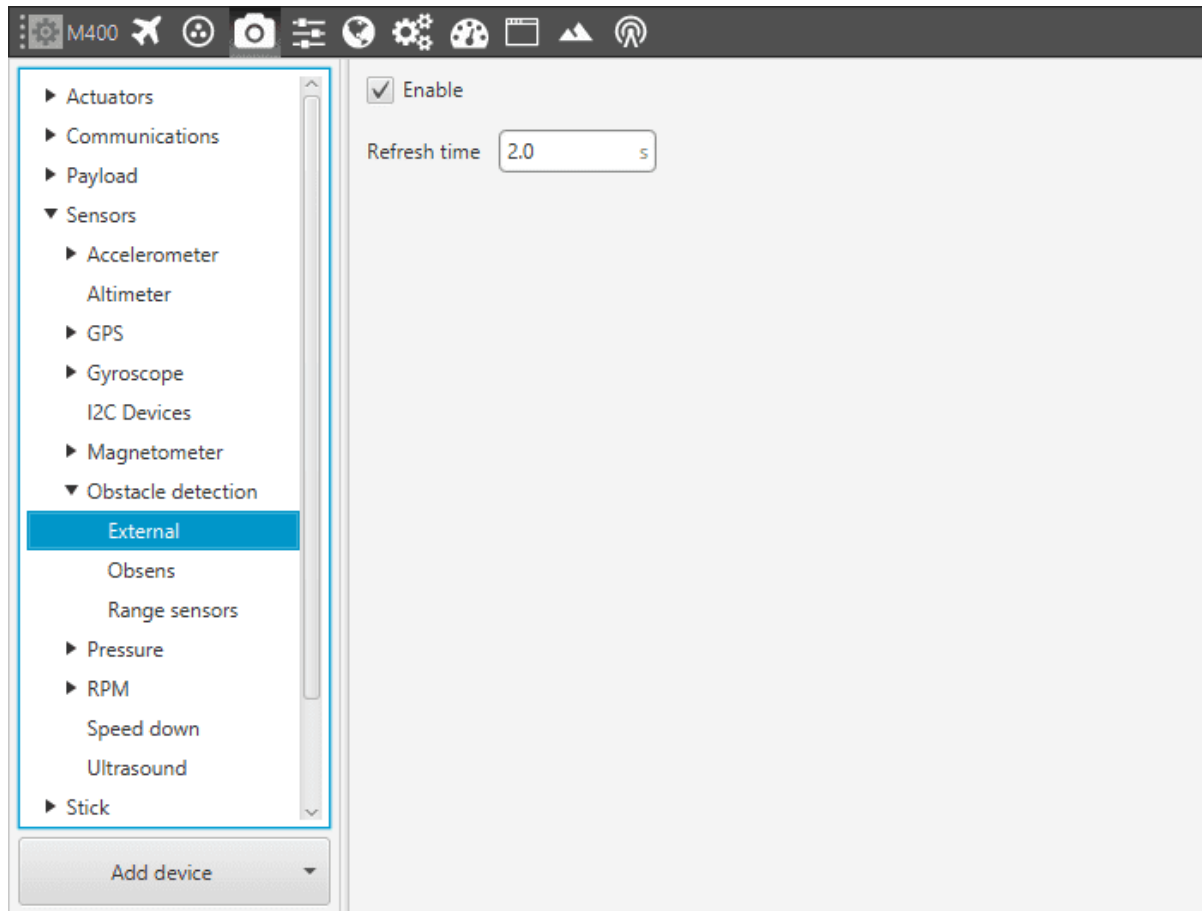
5. Enable reception.

There is an example configuration in the following picture from uAvionix ping 1090/1090i.



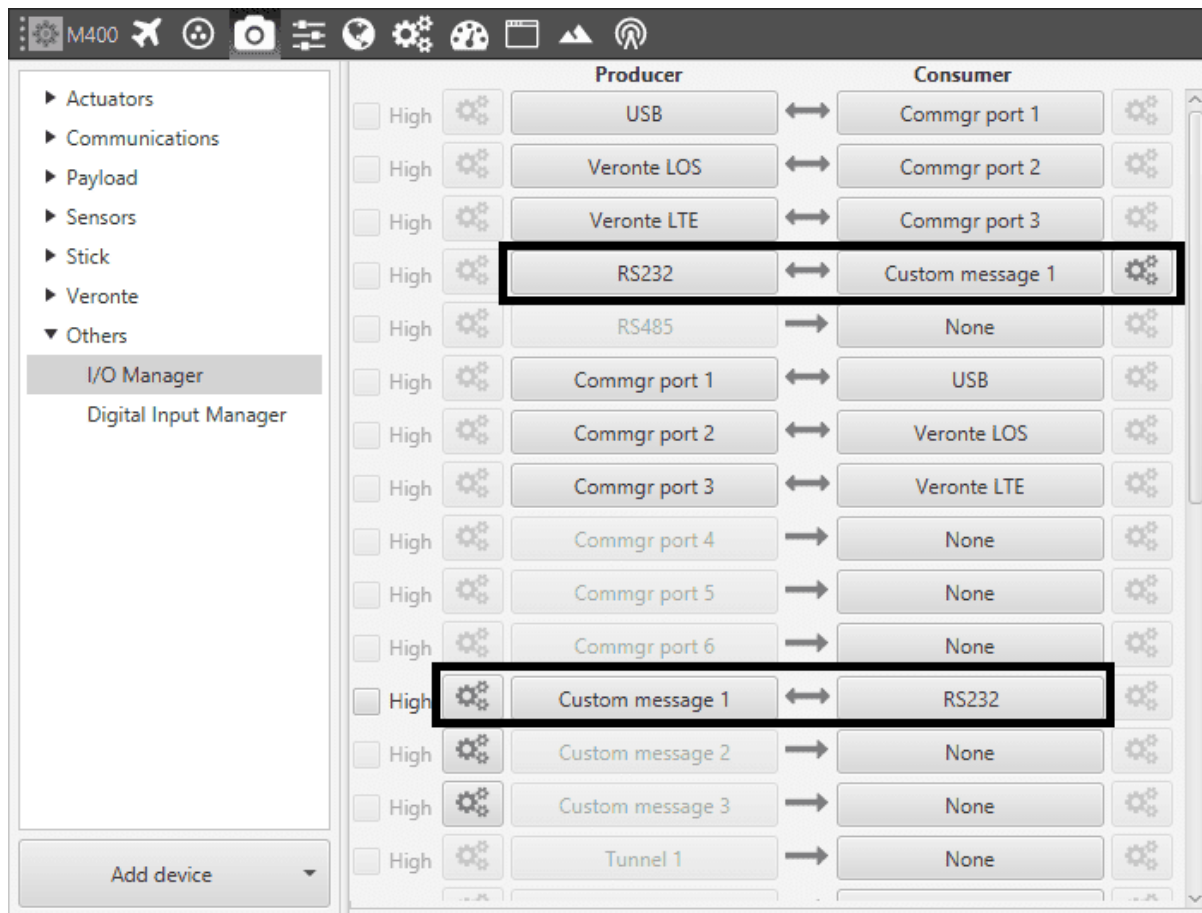
ADS-B Configuration Parameters

The last step you need to do is to enable the External Object Detection in Devices – Sensors – Obstacle Detection.



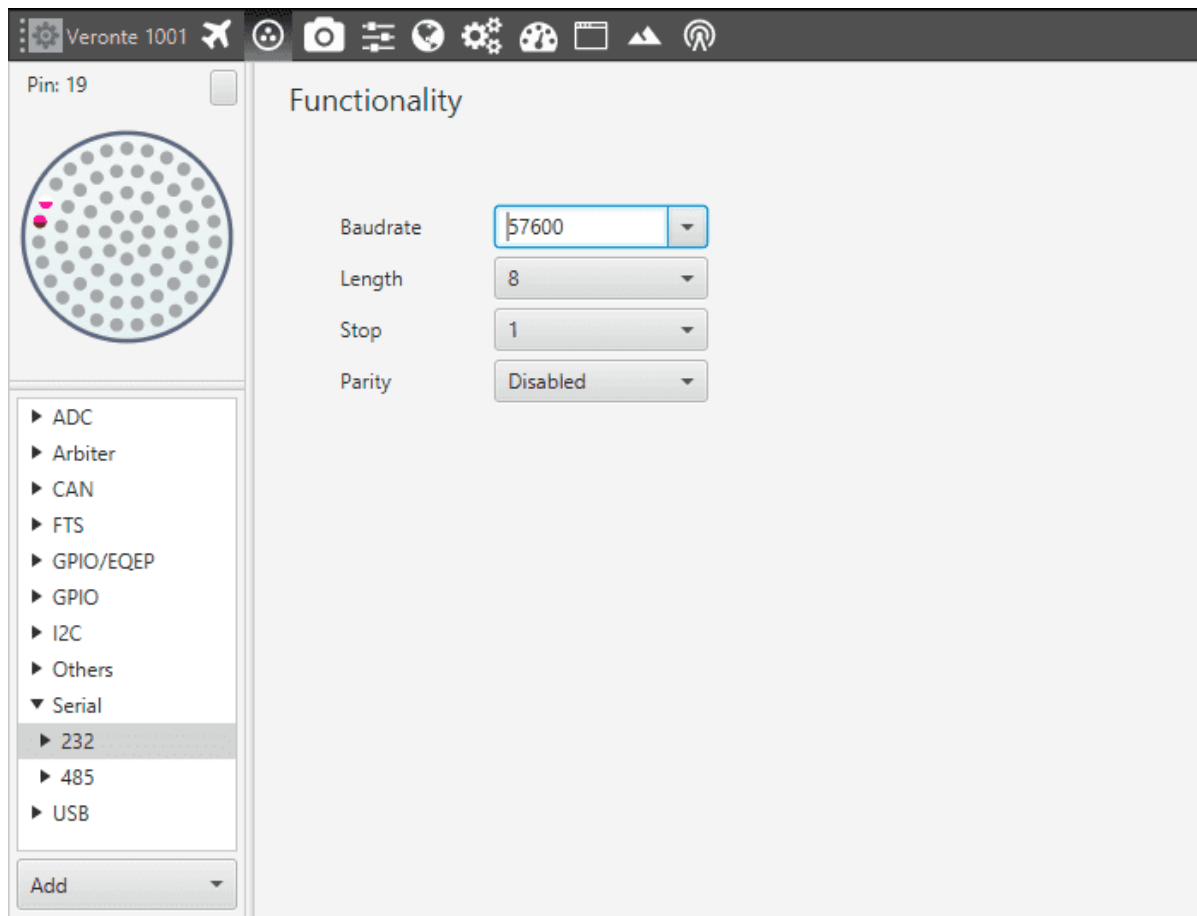
External Obstacle Detection Menu

Once the transponder is configured, go to **Devices – Others – I/O Manager**. Create a bind-bidirectional communication RS232/RS485 (in the example, through RS232) and assign to it the Custom Message selected before.



Transponder Configuration - Serial Communication

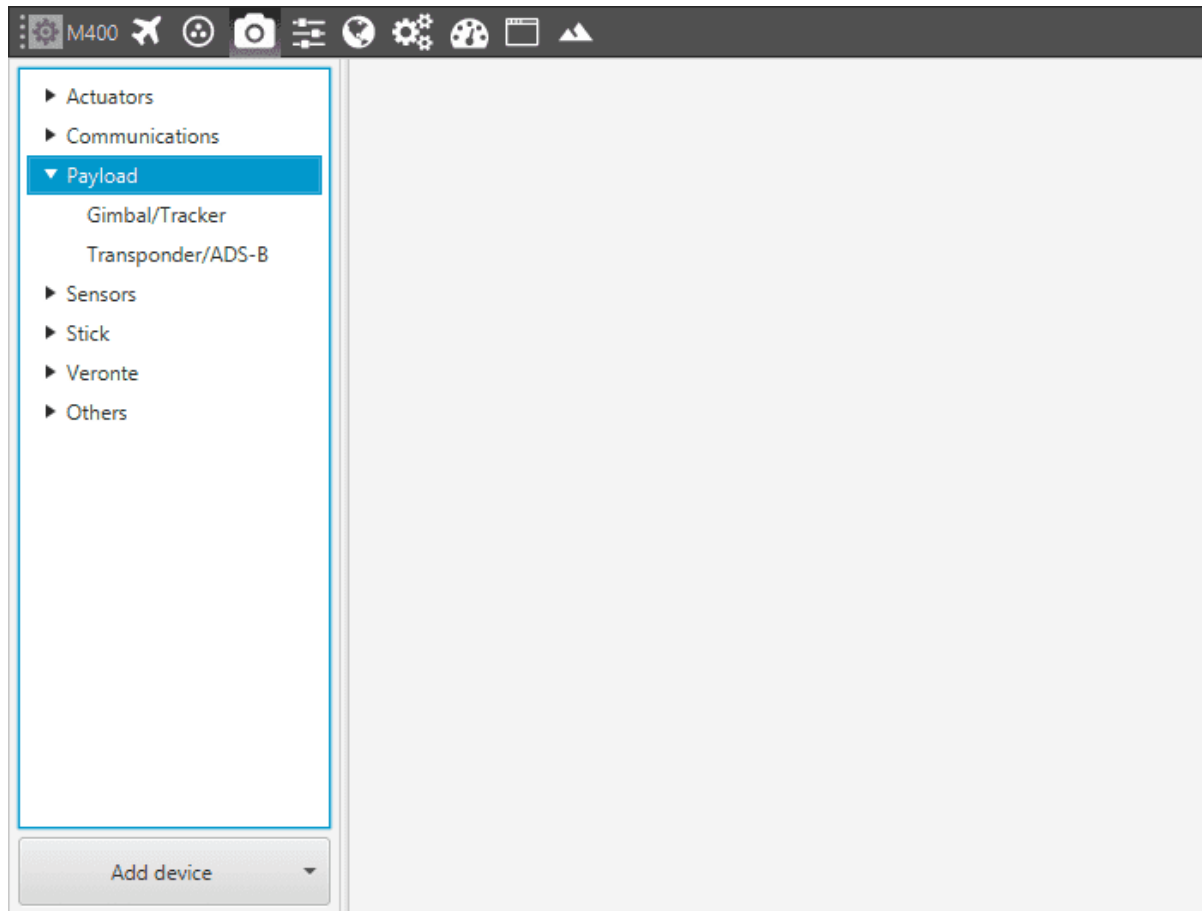
Note that either RS232 or RS485 must be configured with the corresponding Baudrate from the user transponder in **Connections – Serial – 232/485**.



Transponder Configuration - Serial Configuration

In order to see the information from the ADS-B, aside from having visual objects on the screen, the user can use a Widget (refer to Workspace – [ADS-B Decoder](#))

The following menu displays the possible Payload that can be configured with Veronte. Each window will allow the user to configure different parameters from the available variety of payloads.



Payload Menu

7.2.3.4 Sensors

7.2.3.4.1 Accelerometer

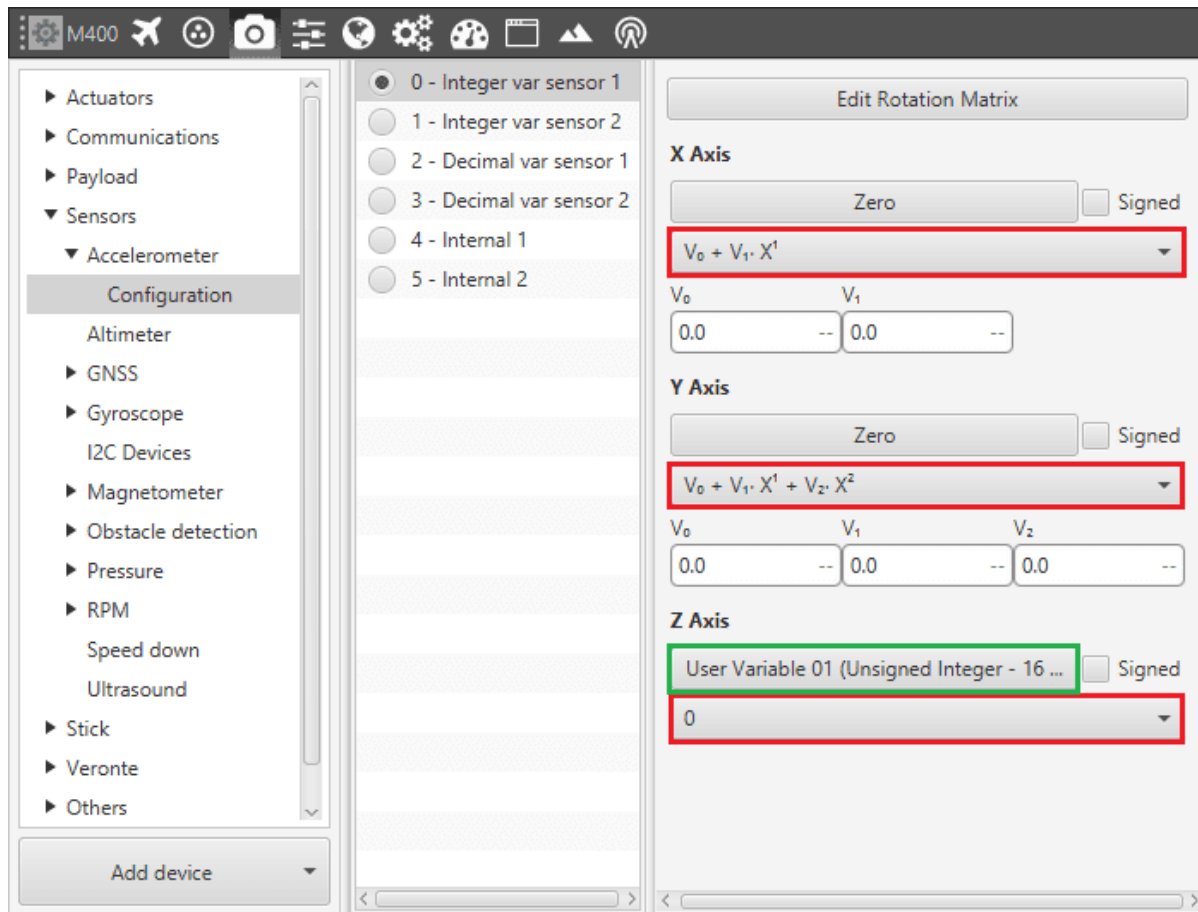
Veronte incorporates two Inertial Measurement Units (IMUs) that allows the Veronte System to measure different variables and that are the main navigation data source. From the IMU, the user can configure the Accelerometer and Gyroscope. The first one is explained below.

The user can choose between 3 types of source for the accelerometer.

- **Integer var sensor 1-2.** Veronte uses a integer value provided by an external sensor.
- **Decimal var sensor 1-2.** Veronte uses a decimal value provided by an external sensor.
- **Internal 1-2 (Main/Secondary).** Veronte uses the internal sensor.

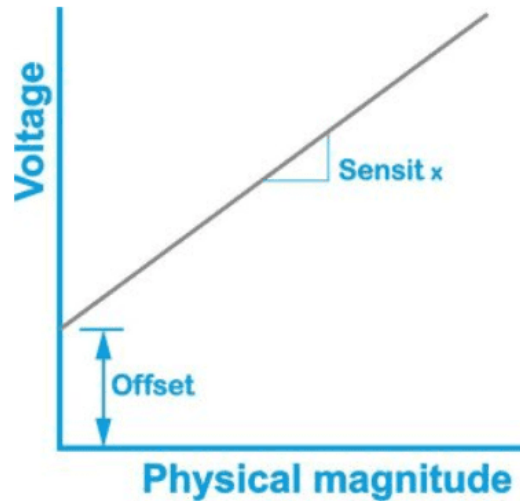
7.2.3.4.1.1 Integer var sensor

In this menu it is possible to configure integer variables provided by an external sensor.



Integer var Accelerometer Menu - Configuration Parameters

When Integer var sensor 1 or 2 are selected, the previous panel will be shown. In this panel, the user selects the variable that has been stored in a user variable (Green Box) and the operations that will be carried on (Red Box). It is possible to use the signal through a linear or quadratic relation. The following image shows an example of a linear relation.

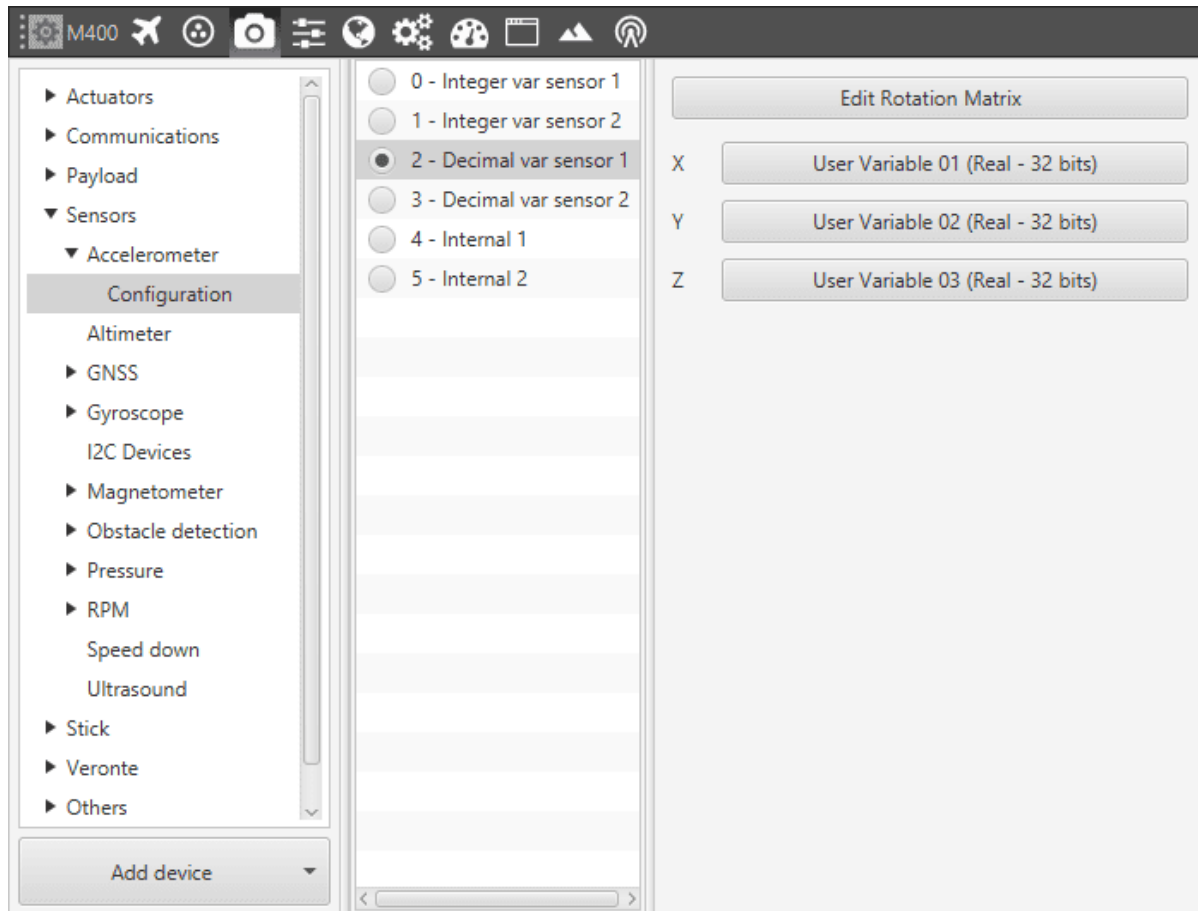


Linear relation of 2 Variables

The process of configuration has to be done using **Custom Messages**. This is to be configured in **Devices - Others - Digital - I/O Manager**. The configuration will depends on the device in use and its communication protocol.

7.2.3.4.1.2 Decimal var sensor

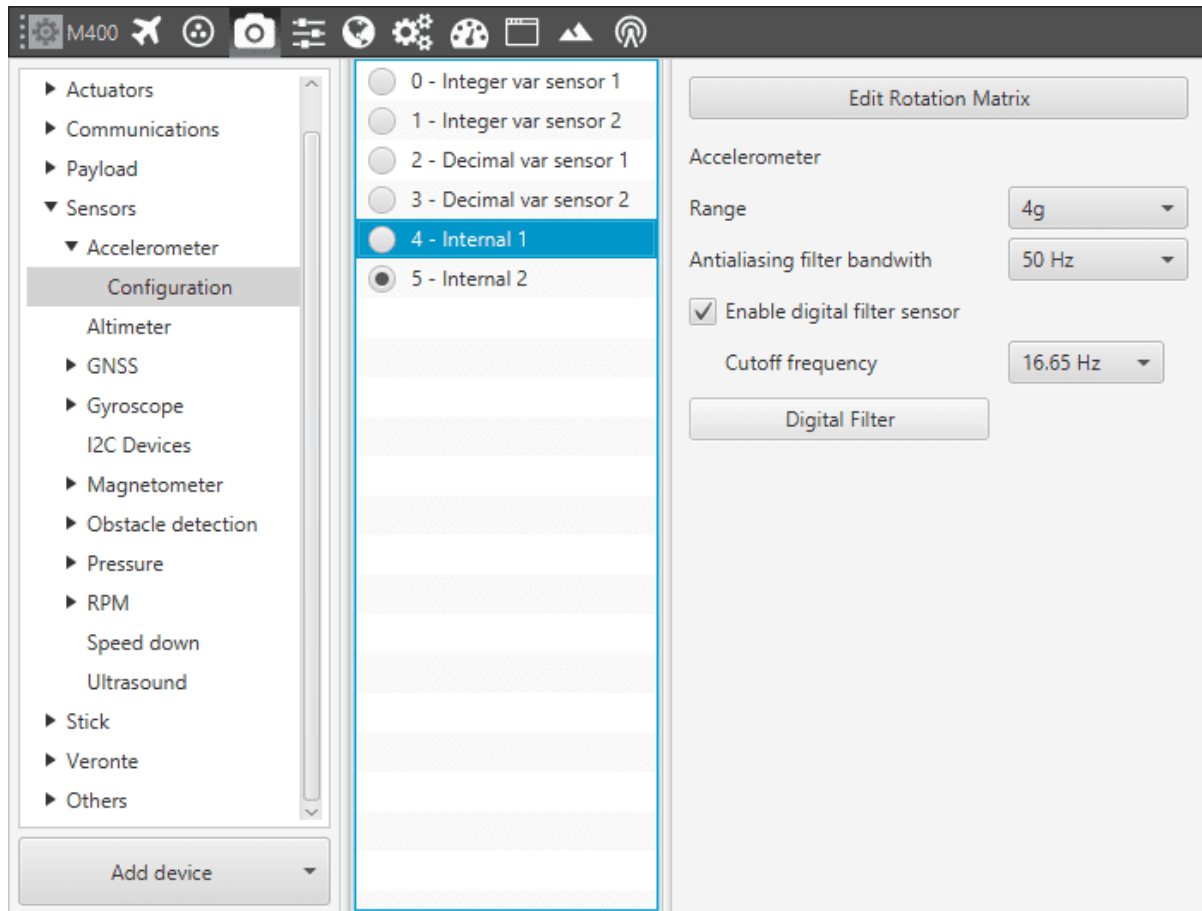
In this menu, the user selects real variables for each axis (X,Y,Z), these do not requiere a signal treatment. The process of configuration is similar to the one carried out when configuring a Integer Variable.



Decimal var Accelerometer Menu - Configuration Parameters

7.2.3.4.1.3 Internal

This last menu available displays the possible parameters that can be configured for the internal Accelerometer.



Accelerometer Menu - Configuration Parameters

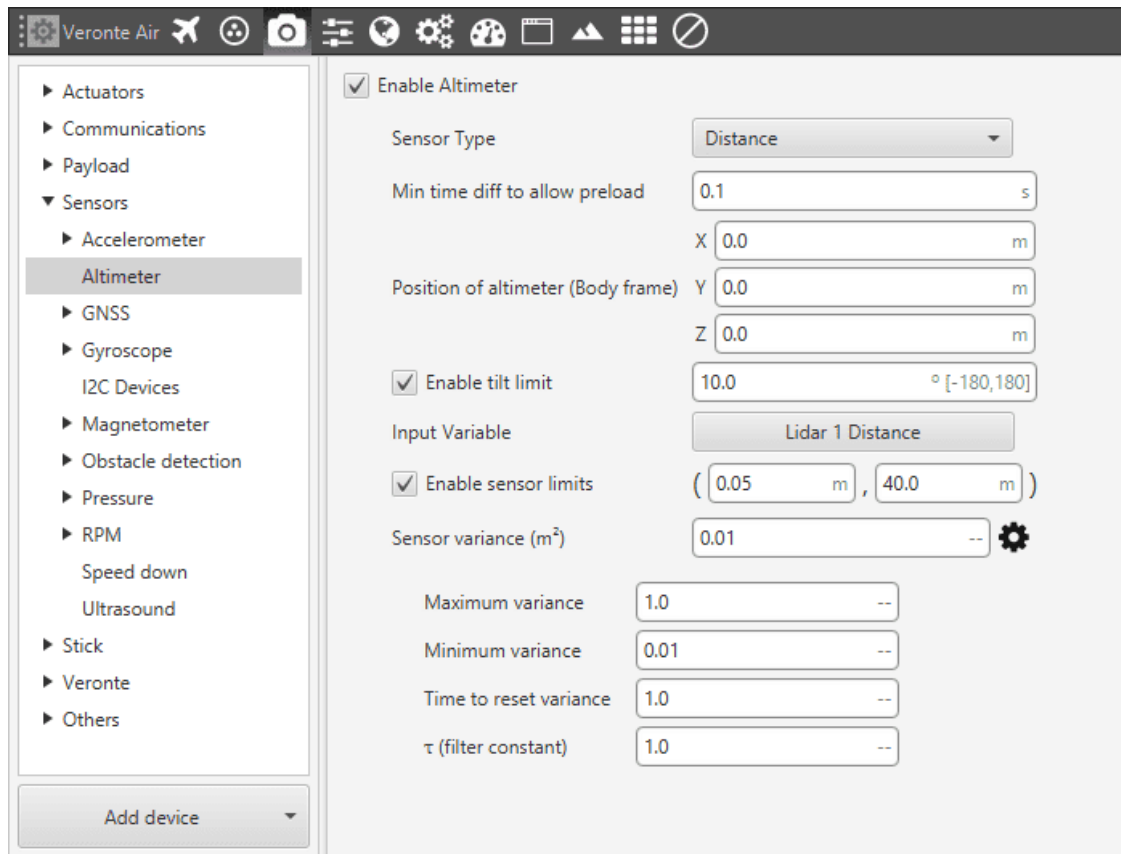
Warning: **Edit Rotation Matrix** brings the position of the accelerometer inside the Veronte Autopilot, it must **NOT** be changed under any circumstance.

In this menu it is possible to set different options regarding range and filters from the accelerometer. The parameters that can be modified are:

- **Range.** Sets the maximum range of performance, high ranges implies less precision while small ranges might mean the system saturates. Values allowed are 2g, 4g, 8g and 16g.
- **Antialiasing filter bandwidth.** It is the bandwidth of the antialiasing **low pass filter**. The options available are 50Hz, 100Hz, 200Hz and 400Hz, the greater the value selected the worse the filtering will be.
- **Enable digital filter.** Enables a low pass filter which its cutoff frequency is configured from the options 16.65Hz, 66.6Hz, 133.2Hz and 740.0Hz. This is a **hardware filter**, included directly in the Accelerometer.
- **Digital filter.** Enables a low pass filter which its cutoff frequency is configured manually, allowing the user to input any desired value in Hz. It is a **software filter**, applied after the hardware filter from the point before.

7.2.3.4.2 Altimeter

The following figure shows the configuration menu for an external altimeter like LIDAR, Sonar, etc.

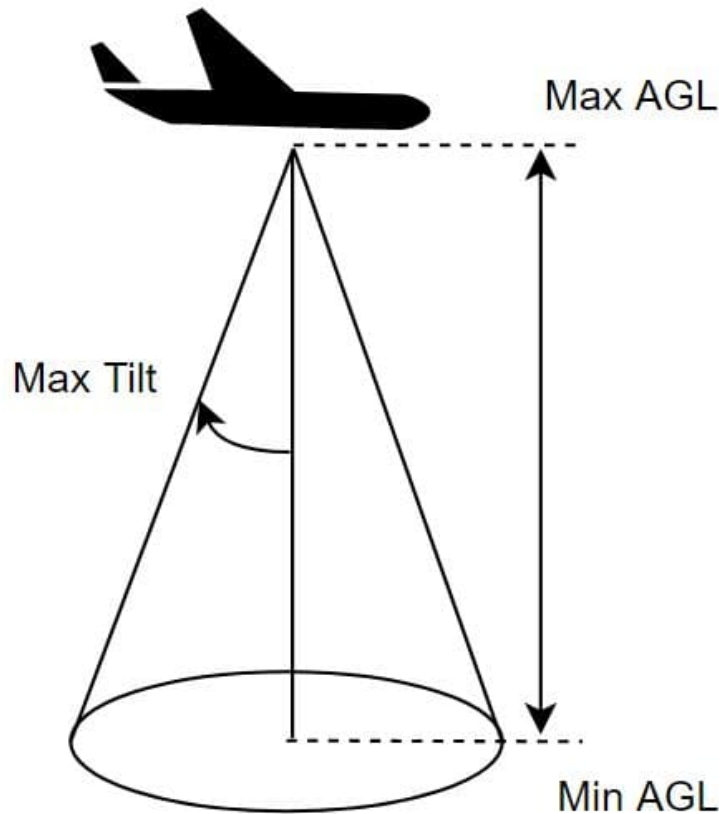


Altimeter Menu - Configuration Parameters

- **Sensor Type.** This drop-down list allows the user to choose between two possible configurations:
 - **AGL.** AGL stands for Height Above Ground Level. This options is intended for use with altimeters that do not depend on the inclination of the platform (e.g. barometric) or when this inclination is or will be negligible.
 - **Distance.** It is meant to be used with radar altimeter or Lidar systems. The distance to the ground (or to a certain object of interest) is corrected taking into account the inclination angles and, also, the X, Y and Z offset between the altimeter position and the origin of the body frame.
- **Min time diff to allow preload.** Establishes the maximum frequency at which this sensor values will enter the Navigation filter. I.e. if it is executed at 50 Hz and we set (as in the picture 0.1 s) 10 Hz, there will be loops when the value is the one stored before.
- **Position of altimeter (Body frame).** Parameter to indicate the distance from the altimeter to the centre of gravity of the platform. This is used to take into account the weight of the altimeter in the aircraft control.
- **Enable tilt limit:** The altimeter is normally installed in a fixed position having a constant direction with respect to the platform. Taking a LIDAR as an example, it is used to measure altitude so it has to point towards the ground, in a direction parallel to the Z body axis. When the vehicle is not level on its X or Y axis (has a pitch or roll angle different from zero), the LIDAR will not point in a direction perpendicular to the ground, and the measurement taken will not be the real altitude of the aircraft. This option is a safe condition to discard the measure of an altimeter when its tilt angle exceeds a certain value.

- **Sensor variance.** It is a measurement related to sensor error, whose units are squared meters. It indicates the weight that this measure will have in the sensor fusion algorithm that combines the information of different sensors to generate an altitude estimation.
- **Input Variable.** Input variable to which the parameters defined here will be applied.
- **Enable sensor limits.** It is the range in which the sensor measurement is taken to be processed by Veronte Pipe. Any outer value will be discarded by the system.

The following figure shows a diagram with the values of maximum and minimum sensor limits altitude, and the maximum tilt angle.



Altimeter Menu - Sensor Limits

- **Maximum variance:** Maximum variance applied to the measurement after measurement lost.
- **Minimum variance:** Minimum variance applied to the measurement after recovering from a measurement lost.
- **Time to reset variance:** Time before considering measurement lost.
- **Tau (Filter constant):** Smoothing parameter for the transition from maximum to minimum variance.

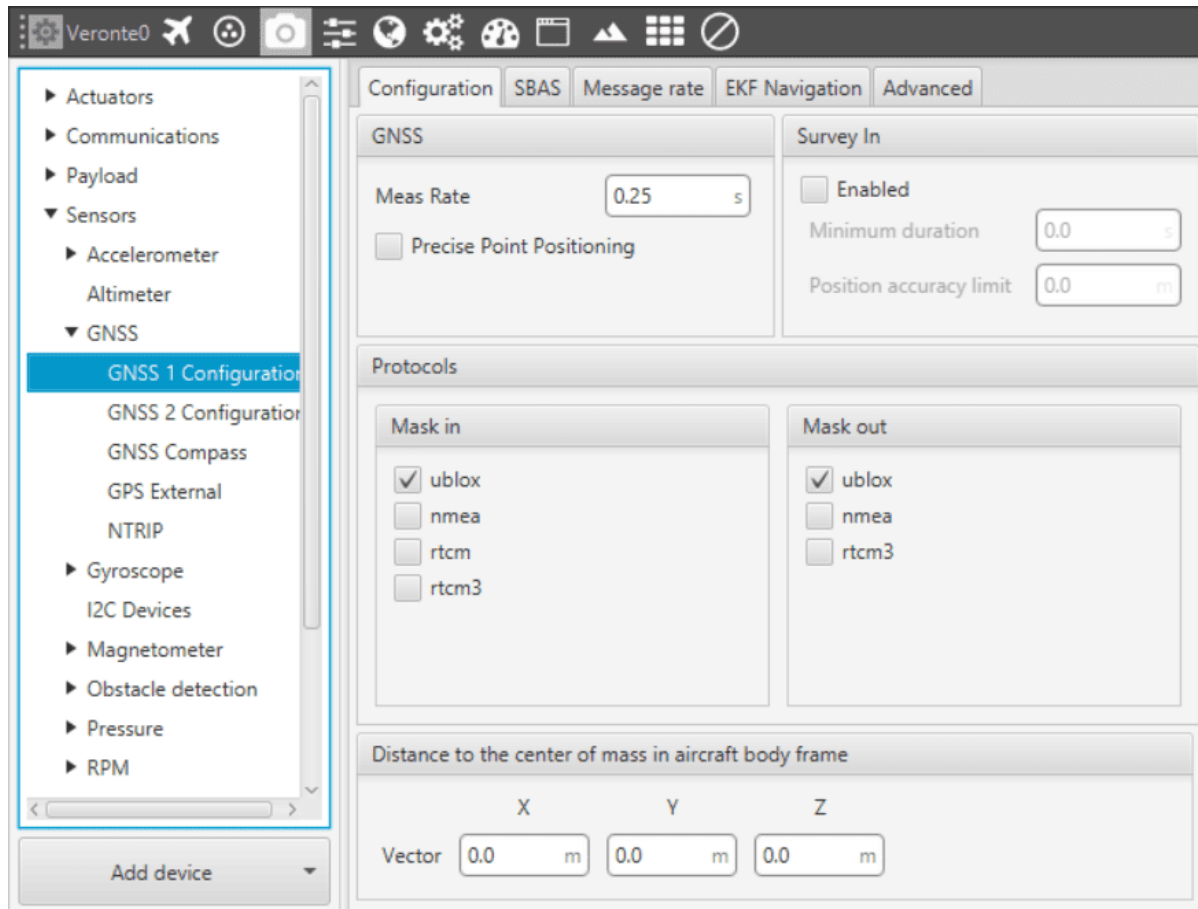
Common implementations of lidar can be found in [Lidar Integration](#).

7.2.3.4.3 GNSS

7.2.3.4.3.1 GNSS 1 & 2 Configuration

7.2.3.4.3.2 Configuration

This menu contains the parameters needed to configure the GNSS receiver 1-2 located in Veronte Autopilot.



GNSS - Configuration Menu

The following parameters are configurable in this menu:

- **GNSS.** Data values that can be configured.
 - **Meas Rate.** Defines the minimum time between data acquisition.
 - **Precise Point Positioning (PPP).** This option is a global precise positioning service, PPP is able to provide position solutions at centimetre to decimetre level after a few minutes with unobstructed sky view.
- **Survey in.** Determines a stationary receiver's position by building a weighted mean of all valid 3D position solutions. This mode should be activated in a Veronte Ground to enable GNSS Differential mode and send corrections to Veronte Air. Two requirements for stopping the procedure must be specified. Survey in procedure will end when both requirements are met.
 - **Minimum duration.** Defines a minimum amount of observation time regardless of the actual number of valid fixes that were used for the position calculation. Reasonable values range from one day for high

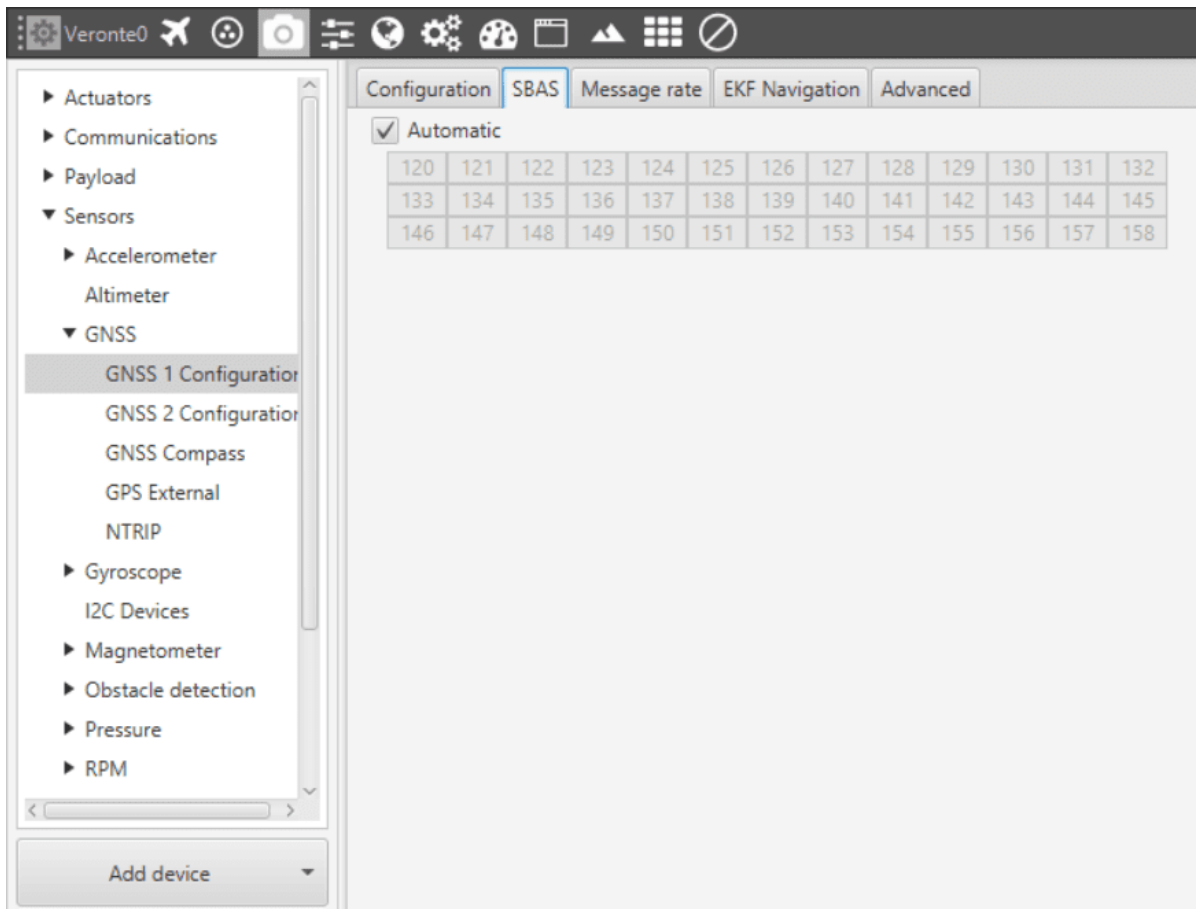
accuracy requirements to a few minutes for coarse position determination

- **Position accuracy limit.** Defines a limit on the spread of positions that contribute to the calculated mean.
- **Protocols.** Allows the user to select the different communication protocols as input or output. One port can handle several protocols at the same time (e.g. NMEA and UBX).
 - Protocols will be changed when using the GNSS Compass Wizard (see in Devices - GNSS - GNSS Compass).
- **Distance to center of mass.** It is used to set the relative distance to the center of mass from the GNSS antenna in aircraft body axis. This parameter has to be set correctly in order to get a correct value when using GNSS Compass.

More information about protocols and configuration can be found [here](#).

7.2.3.4.3.3 SBAS

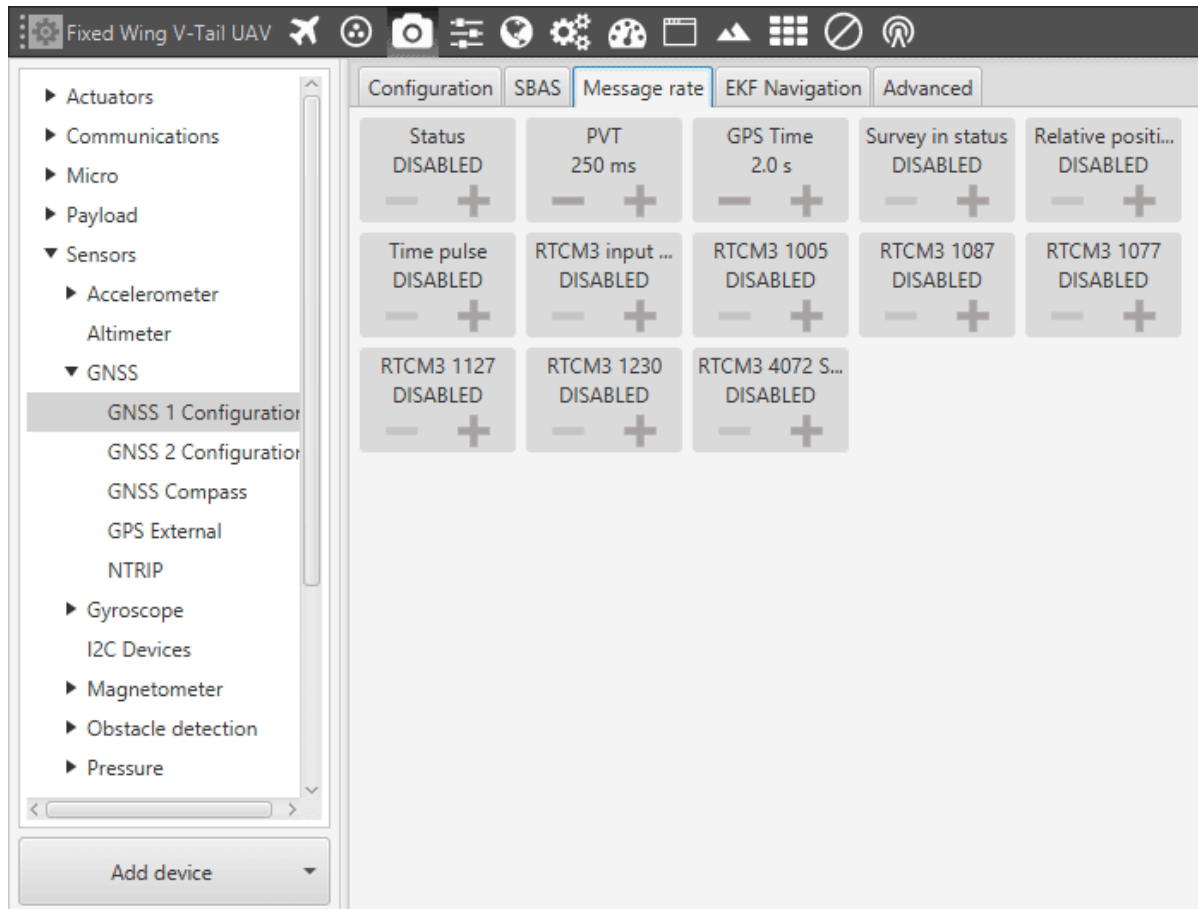
SBAS stands for Satellite Based Augmentation System. It is a set of geostationary satellites that are used to check the status of the signals sent by the GPS Satellites, and to improve the tracking via the correction of atmospheric disturbances, orbit deviations, clock errors and so on. In Veronte Pipe, it is possible to select the satellites that will be used for this purpose (selecting the number that appears in the table of the following figure), or make the software to automatically choose them according to the location of the platform.



GNSS - SBAS Menu

7.2.3.4.3.4 Message Rate

The **Message rate** options are used to set the time between the messages received at the autopilot. Each one of the different messages can be configured separately: ECEF (earth centred, earth fixed reference system), LLH (latitude, longitude and height), Speed, GPS Time, SV Status (state of the GPS satellite) and so on. This messages will be changed when using the GNSS Compass Wizard (see in Devices - GNSS - GNSS Compass).



GNSS - Message Rate

For information about RTCM 3 message list can be found [here](#) .

7.2.3.4.3.5 Estimation Error

Configuration SBAS Message rate **EKF Navigation** Advanced

☒ Enable GNSS in EKF Navigation

☐ Use position measures in the attitude calculation

☐ Use speed measures in attitude calculation

Square error on strong acceleration for position m^2

Square error on strong acceleration for speed $(m/s)^2$

Acceleration m/s^2

Duration of effect (disappears linearly with time) s

| | Square error | Use receiver value if present |
|---------------------|--|-------------------------------------|
| GNSS North Position | <input type="text" value="10.0"/> m^2 | <input checked="" type="checkbox"/> |
| GNSS East Position | <input type="text" value="10.0"/> m^2 | <input checked="" type="checkbox"/> |
| GNSS Down Position | <input type="text" value="10.0"/> m^2 | <input checked="" type="checkbox"/> |
| GNSS North Velocity | <input type="text" value="0.1"/> $(m/s)^2$ | <input checked="" type="checkbox"/> |
| GNSS East Velocity | <input type="text" value="0.1"/> $(m/s)^2$ | <input checked="" type="checkbox"/> |
| GNSS Down Velocity | <input type="text" value="0.1"/> $(m/s)^2$ | <input checked="" type="checkbox"/> |

Add device

GNSS - Estimation Error

- **Enable check:** Enables/Disables the usage GNSS solution in the Extended Kalman Filter. Veronte will keep receiving GNSS information from that module, but it will not be considered.
- **Use position measures in the attitude calculation:** When enabled, the position data from the GNSS solution is considered for the attitude estimation.
- **Use speed measures in the attitude calculation:** When enabled, the speed data from the GNSS solution is considered for the attitude estimation.

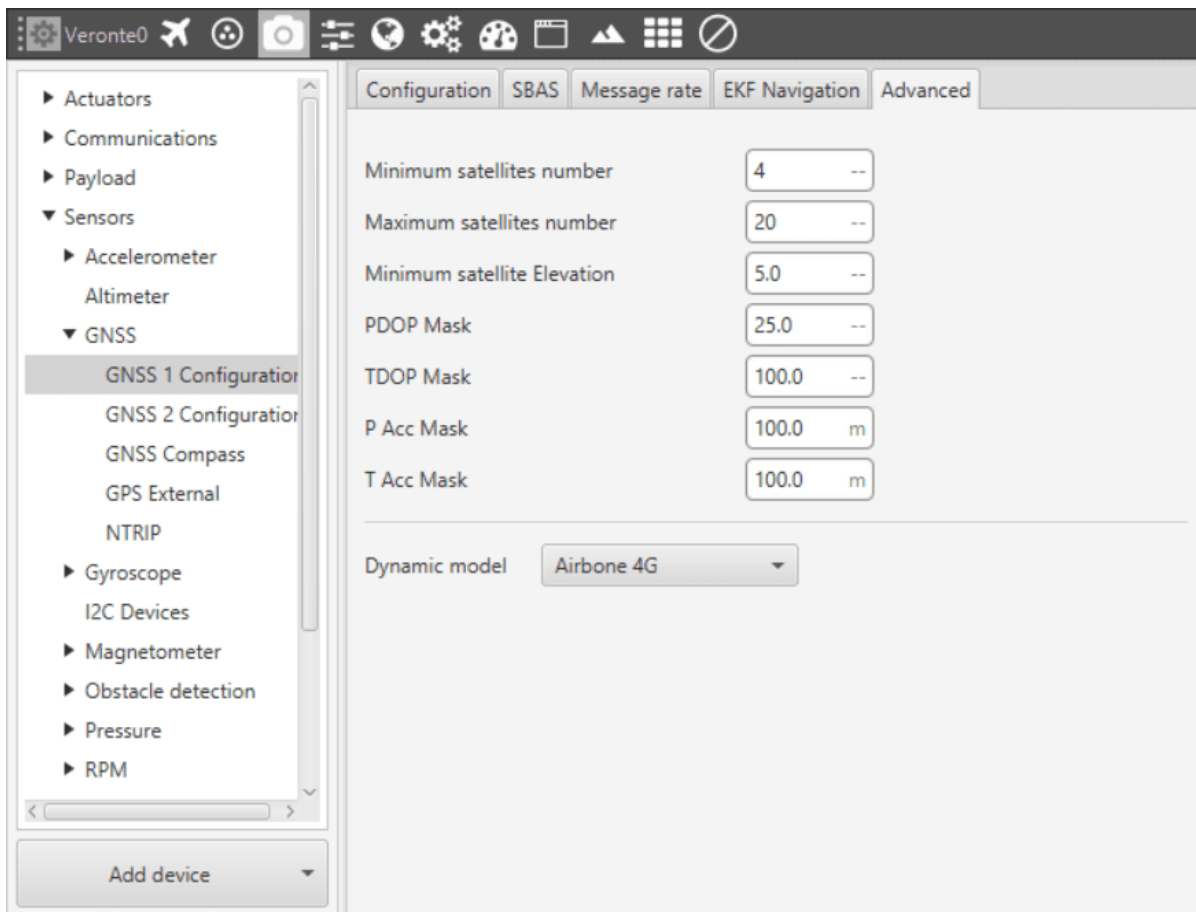
The variances considered in the EKF for the GNSS solution are by default the values provided by the GNSS receiver but can be modified for more complex scenarios.

- **GNSS North Position:** Variance for the North component of the position solution.
- **GNSS East Position:** Variance for the East component of the position solution.
- **GNSS Down Position:** Variance for the Down component of the position solution.
- **GNSS North Velocity:** Variance for the North component of the velocity solution.
- **GNSS East Velocity:** Variance for the East component of the velocity solution.
- **GNSS Down Velocity:** Variance for the Down component of the velocity solution.

Under strong accelerations the information provided by the GNSS does not represent the fast change of position and velocity therefore different variance must be considered in the EKF.

- **Square error on strong acceleration for position:** Under a strong acceleration the variance for the position solution it is changed to the specified value.
- **Square error on strong acceleration for speed:** Under a strong acceleration the variance for the speed solution it is changed to the specified value.
- **Acceleration:** Threshold definition. When this threshold is exceeded, strong acceleration variances are considered.
- **Duration of effect:** Time needed to restore the default variances of the GNSS solution .

7.2.3.4.3.6 Advanced



GNSS - Estimation Error

Warning: Modifying these parameters can cause problems during the acquisition of GNSS positioning.

- **Minimum satellites number:** Minimum number of satellites needed to have position fixed.
- **Maximum satellites number:** Maximum number of satellites needed to have position fixed

- **Minimum satellite elevation:** Minimum elevation of a satellite to be considered. Value in degrees.
- **PDOP mask:** Maximum Position dilution of precision to consider the solution.
- **TDOP mask:** Maximum Time dilution of precision to consider the solution.
- **P Acc mask:** Maximum Position accuracy to consider the solution.
- **T Acc mask:** Maximum Time accuracy to consider the solution.

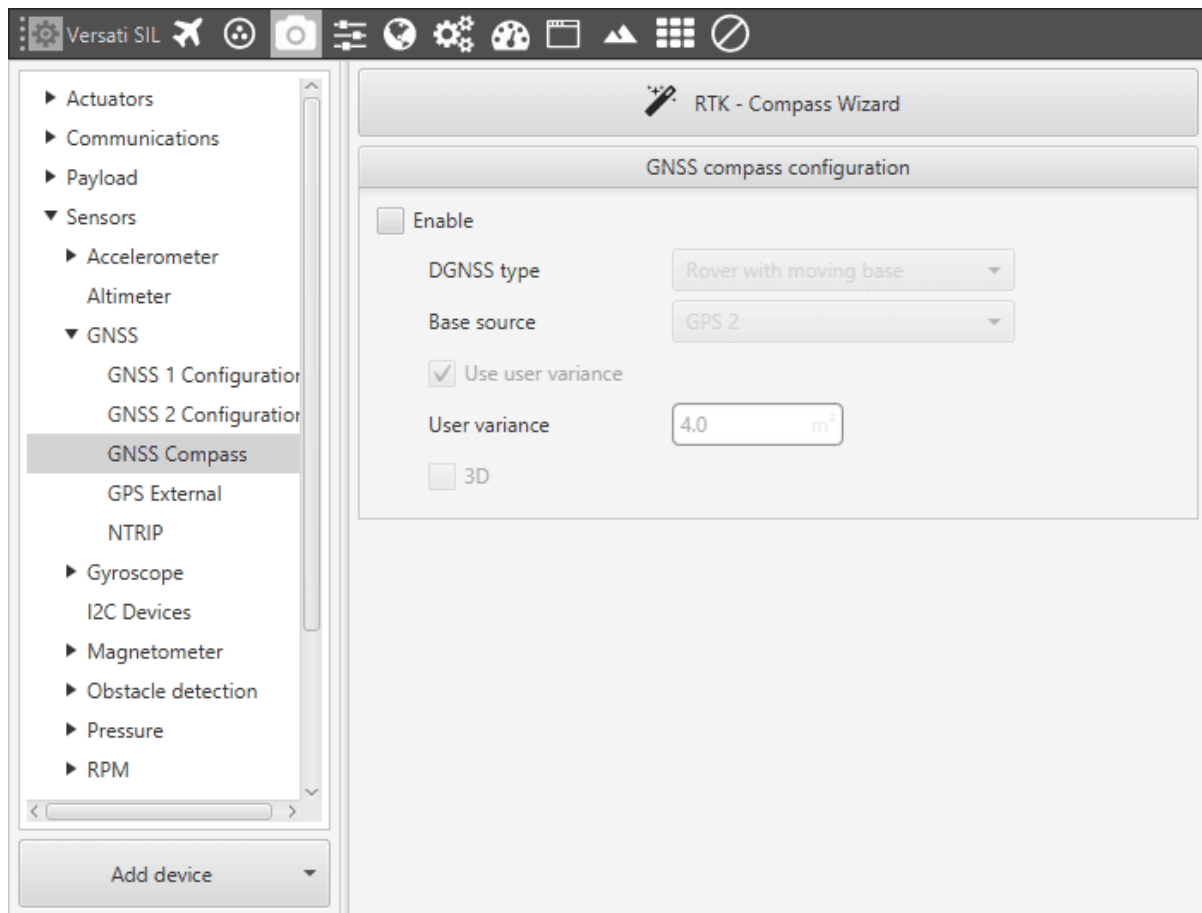
Dynamic Model

The embedded receiver supports different dynamic platform models to adjust the GNSS navigation engine to the expected application environment. The settings improve the receiver's interpretation of the measurements and thus provide a more accurate position output. Setting the receiver to an unsuitable platform model for the given application environment is likely to result in a loss of receiver performance and position accuracy.

| Platform | Description |
|-------------|--|
| Portable | Applications with low acceleration. |
| Stationary | Stationary applications. Velocity restricted to 0 m/s. Zero dynamics assumed |
| Pedestrian | Applications with low acceleration and speed. Low acceleration assumed. |
| Automotive | Used for applications with equivalent dynamics to those of a car. Low vertical acceleration assumed. |
| Sea | Recommended for applications at sea, with zero vertical velocity. Zero vertical velocity assumed. Sea level assumed. |
| Airborne 1G | Used for applications with a higher dynamic range and greater vertical acceleration than a car. |
| Airborne 2G | Recommended for typical airborne environments. |
| Airborne 4G | Recommended for extremely dynamic environments. |

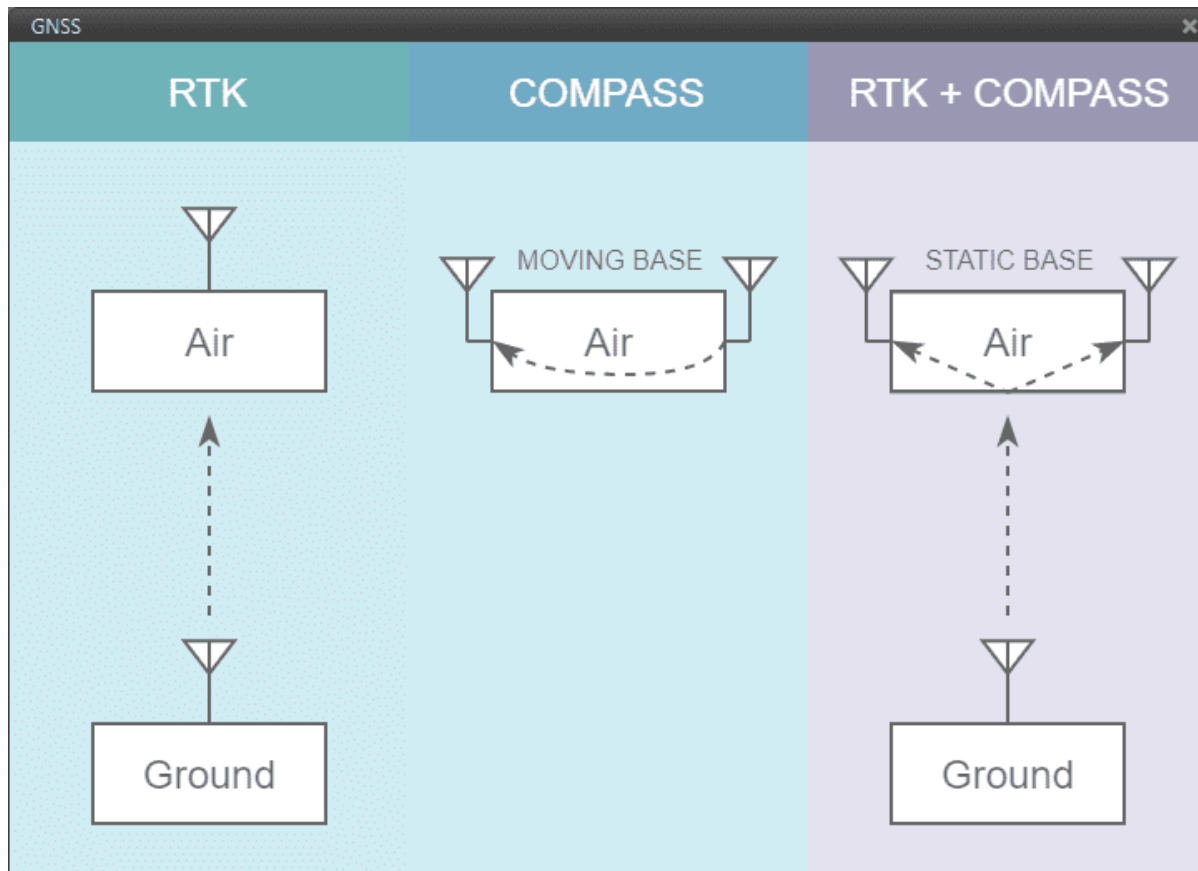
7.2.3.4.3.7 GNSS Compass

GNSS Compass menu allows the user to access to the RTK - Compass Wizard (an interface which helps the user configuring everything related to RTK - GNSS Compass) and some other parameters which could be changed afterwards or if the user has knowledge enough.



GNSS Compass - Configuration Menu

By clicking on RTK - Compass Wizard button, the user will access the configuration menu where three options will be displayed.

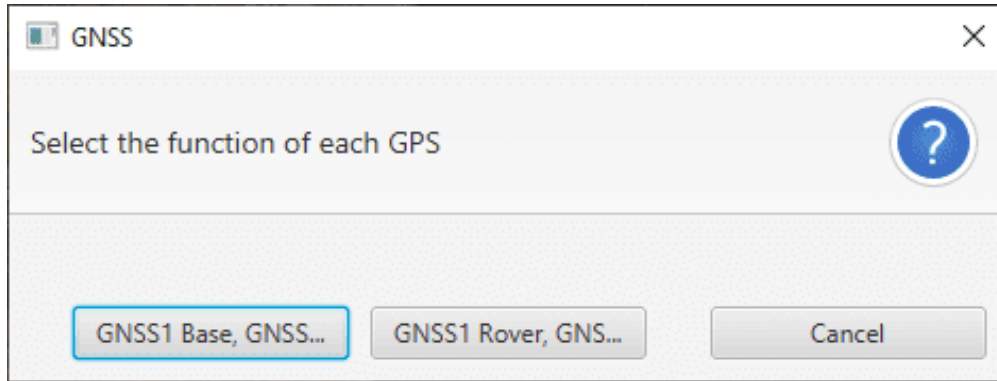


GNSS Compass - RTK & Compass Wizard

- **RTK.** Stands for Real Time Kinematics and it is a satellite navigation technique used to **enhance the precision** of position data derived from satellite-based positioning systems. It requires from **2 GNSS receivers** placed in **different autopilots** to work.
- **Compass.** The GNSS compass provides accurate dual antenna **GNSS based heading** that is not subject to magnetic interference. It requires from **2 GNSS receivers** placed in **the same autopilot** to work.
- **RTK + Compass.** A hybrid combination where both tool are employed at the same time in a system where the **AIR unit** must have **2 GNSS receivers** and the **GND**, at least, **must have 1**.

When accessing the GNSS Compass feature, the user will be asked which configuration is preferred in accordance to the function of each receiver:

- GNSS 1 Base and GNSS 2 Rover.
- GNSS 1 Rover and GNSS 2 Base.



GNSS Compass - Compass Base/Rover

Warning: It's possible that Compass options are blocked in Wizard if the position of GNSS antennas is not defined. To avoid this problem, change the distance to the center of mass in aircraft body frame in Devices/Sensors/GNSS.

These same parameters can be manually changed afterwards in GNSS Compass Configuration. After using the wizard, the Enable check will be marked. The options enabled this way are:

- **DGNSS Type.** Where the option available are “Rover with moving base” and “Rover with rover (static base)”. The first option references to GNSS Compass, while the second is used in RTK with Compass. Consider the parameter established will be the one coming from the Wizard configuration.
- **Base source.** Where option are GPS 1 or 2.

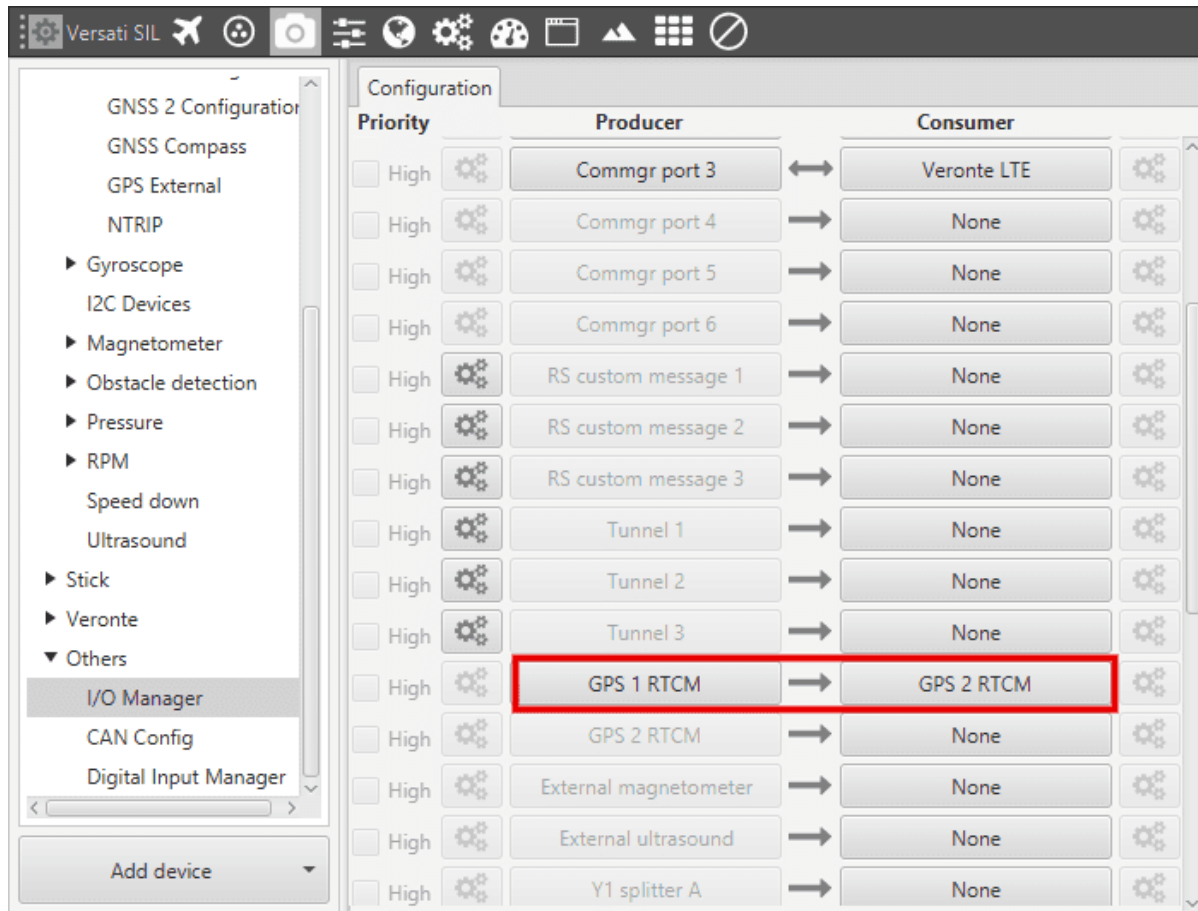
To configure the rest of parameters in Compass mode, user has to modify the GNSS configuration and message rate. First, set the MEAS rate to 0.5. Then configure the protocols. RtcM3 is enabled in *mask in* for the GNSS configured as rover, and in the *mask out* for the GNSS base. The message rates are different in each GNSS. The essential messages are shown below:

| Configuration | SBAS | Message rate | EKF Navigation | Advanced | BASE |
|-------------------------------|------------------------------------|----------------------------------|-------------------------------------|-------------------------------------|------|
| Status 500 ms — + | PVT 500 ms — + | GPS Time 4.0 s — + | Survey in status DISABLED — + | Relative posi... DISABLED — + | |
| Time pulse DISABLED — + | RTCM3 input ... DISABLED — + | RTCM3 1005 DISABLED — + | RTCM3 1087 500 ms — + | RTCM3 1077 500 ms — + | |
| RTCM3 1127 DISABLED — + | RTCM3 1230 1.0 s — + | RTCM3 4072 S... 500 ms — + | | | |

| Configuration | SBAS | Message rate | EKF Navigation | Advanced | ROVER |
|-------------------------------|------------------------------------|------------------------------------|-------------------------------------|-----------------------------------|-------|
| Status 500 ms — + | PVT 500 ms — + | GPS Time 4.0 s — + | Survey in status DISABLED — + | Relative posi... 500 ms — + | |
| Time pulse DISABLED — + | RTCM3 input ... DISABLED — + | RTCM3 1005 DISABLED — + | RTCM3 1087 DISABLED — + | RTCM3 1077 DISABLED — + | |
| RTCM3 1127 DISABLED — + | RTCM3 1230 DISABLED — + | RTCM3 4072 S... DISABLED — + | | | |

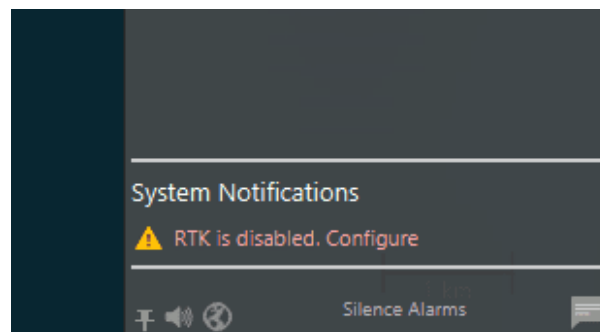
GNSS Compass - Compass Message Rate

In the I/O Manager menu (**Devices - Others**), base GNSS (producer) sends RTCM information to rover GNSS (consumer).



GNSS Compass - Compass I/O Manager

The RTK - Compass Wizard does also appear as a **red notification** on the bottom right side of Veronte Pipe.

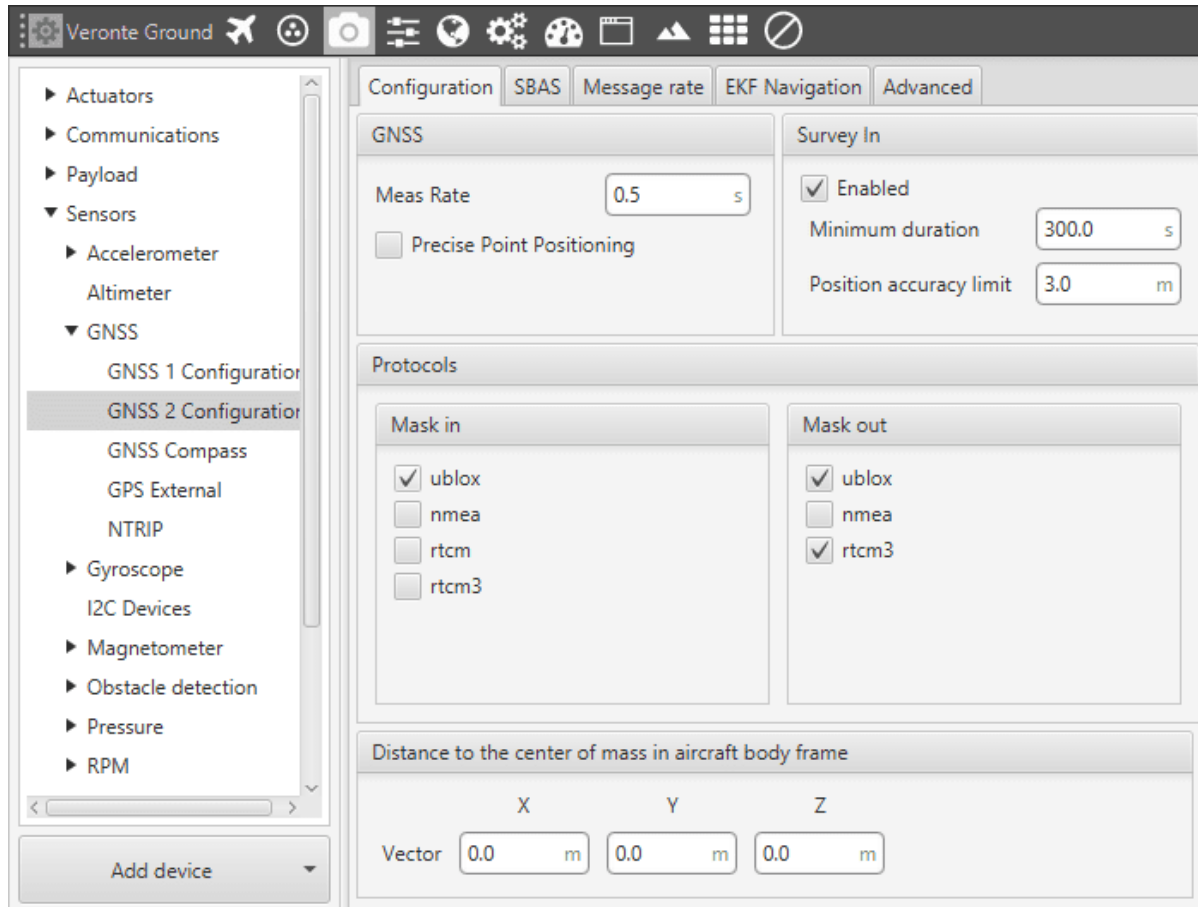


GNSS Compass - Configuration Notification

By clicking on it, the Wizard will ask the user which AP is the AIR unit and which one is the GND (in case RTK functionality is expected) and afterwards, the GNSS Compass wizard will be displayed and use in the same way.

7.2.3.4.3.8 RTK Configuration

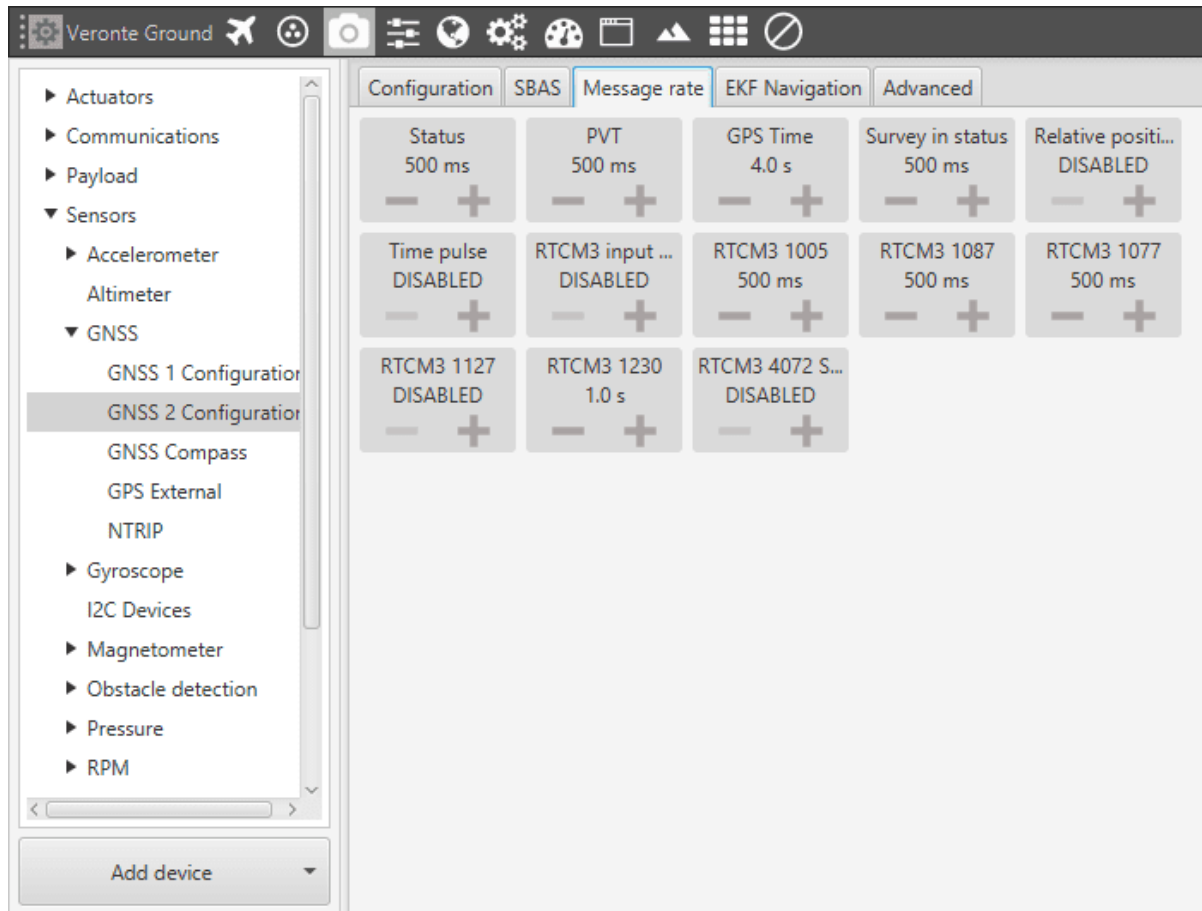
Apart from the wizard option, it is possible to configure RTK following the next guide. First, it is needed to configure Ground Station setup. Ground station has two GNSS ports. User can choose one of them to carry out the RTK technique (Using Wizard tool the first GNSS is chosen by default). The main parameters are shown in the picture below.



GNSS Compass - RTK configuration (Ground Unit)

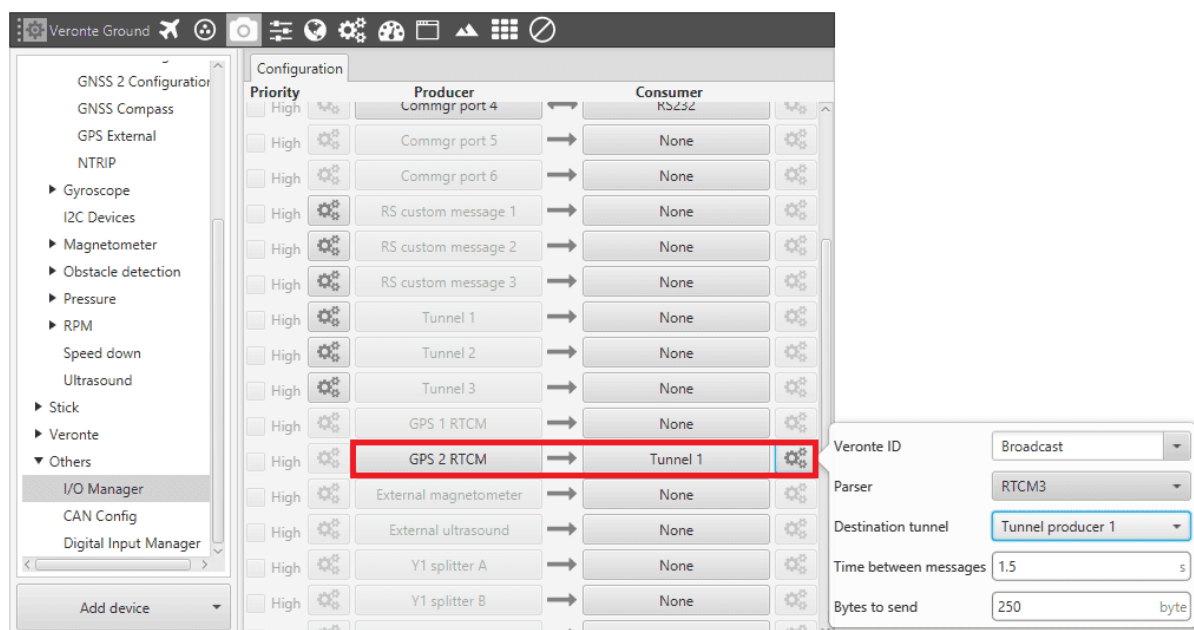
- **Meas Rate:** measurement rate. The optimal is fixed in 0.5.
- **Protocols:** rctm3 has to be enabled in Mask out.
- **Survey in:** to configure the **minimum** 'Survey In' accuracy and for how long the accuracy needs to be held within this range in order to enable the RTK feature.

Then, the following messages need to be configured in the 'Message Rate' tab. They are properly selected in the next picture.



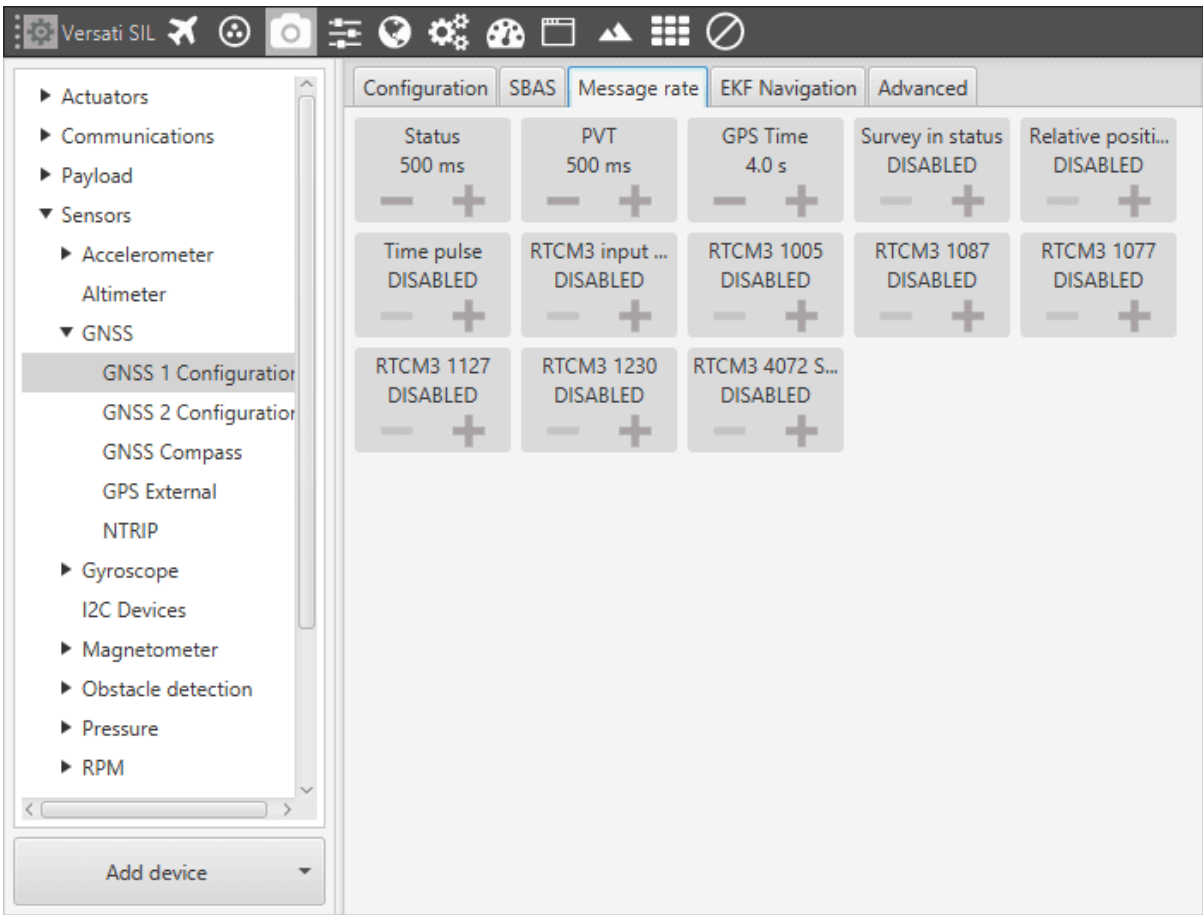
GNSS Compass - RTK Message Rate (Ground Unit)

Finally, go to the I/O Manager, and add a message by a tunnel with RTCM information. This will be sent to AIR UNIT to correct their GNSS measurements and improve their accuracy.



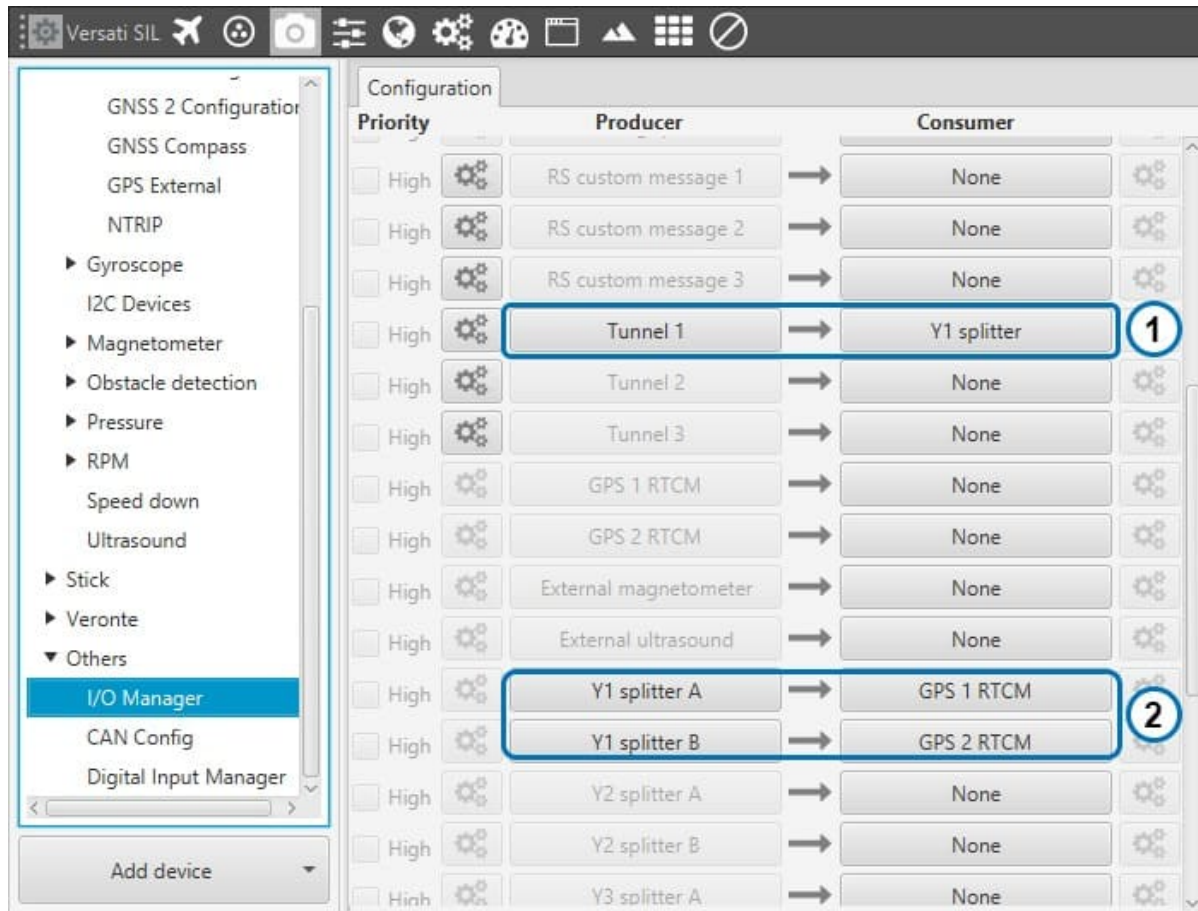
GNSS Compass - RTK I/O Manager (Ground Unit)

Now, it's time to configure the AIR Unit. Meas Rate to 0.5, and rtm3 enabled in *Mask in* (Protocols). The message rate has to be configured as follows:



GNSS Compass - RTK Message Rate (Air Unit)

The last step consists of configuring I/O Manager to receive RTCM message sent by the tunnel (Ground unit). If user wants to receive position corrections in both GNSS devices the tunnel has to be splitted in 2 channel (1). Then, user has to link these splitted channels to GPS RTCM (2).

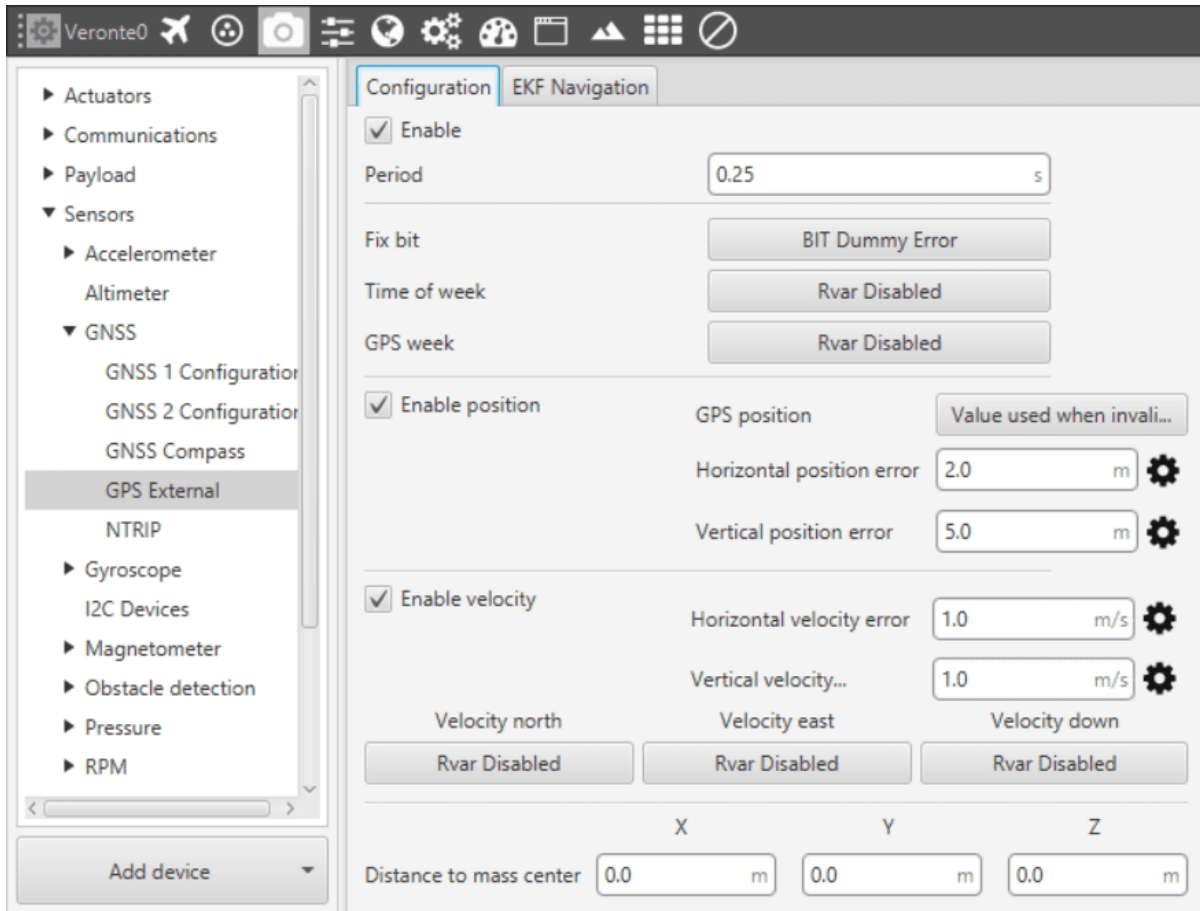


GNSS Compass - RTK I/O Manager (Air Unit)

7.2.3.4.3.9 GNSS External

If the reception of GNSS information is done through an external system, the user must configure it in this menu, so that system can be included in the navigation filters. After properly configuring the communication protocol on the corresponding channel (RS 232, RS485, CAN...) the GNSS External variables of interest must be filled in this interface:

7.2.3.4.3.10 Configuration



GNSS External - Configuration Menu

Check GNSS External device communication protocol before filling this menu.

The user should create a Custom Message according to the communication protocol used by the external sensor, so that its readings are stored in system variables. Then, the user can select this variables to configure the following parameters:

- **Period.** Defines the period of incoming information from the external system.
- **Fix bit.** Data provided by the external device which is important to know the status of the positioning.
- **Time of week.** Variable extracted from the communication protocol defining the time of the week.
- **GPS Week.** Variable extracted from the communication protocol defining the week.
- **GPS Position.** Variable defining latitude, longitude and height from GNSS. Usually Moving Object variables are used in Pipe.
- **Horizontal/Vertical Position Error.** Defined by the GNSS External device provider.
- **Velocity north/east/down.** Variables extracted from the communication protocol defining GNSS velocity measured.
- **Horizontal/Vertical velocity error.** Defined by the GNSS External device provider.

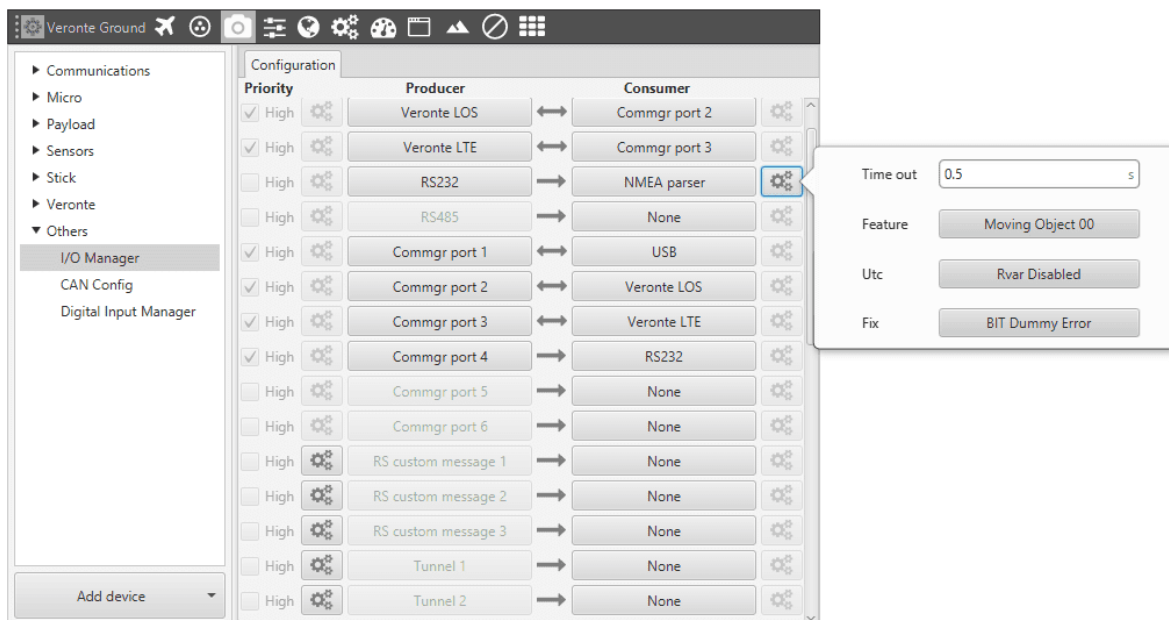
- **Distance to mass center.** Define the distance between the uav mass center and the position of the external GNSS antenna.

7.2.3.4.3.11 NMEA Parser

Another way to add an external GNSS device is to use the **NMEA Parser** Consumer (*I/O Manager*). This consumer allows to receive NMEA 0183 messages and parses them directly. The NMEA Parser configuration menu includes the following parameters:

- **Time out.** Defines the period of incoming information from the external system.
- **Feature.** Variable extracted from the message defining the GNSS position. Usually Moving Object variables are used in Pipe.
- **Utc.** Variable extracted from the message defining the UTC.
- **Fix.** Data provided by the external device which is important to know the status of the positioning.

Once the NMEA message has been parsed, the variables used for **Fix** and **Feature** can be selected in the GPS External configuration menu as **Fix Bit** and **GPS Position**. The variable used for **Utc** can be selected in the GPS External configuration menu as **Time of week**. The other parameters of the GPS External configuration must be defined according to their definitions shown above.



NMEA Parser - Configuration Menu

7.2.3.4.3.12 Estimation Error

See section Setup - Devices - Sensors - GNSS - *GNSS 1 & 2 Configuration*.

7.2.3.4.3.13 NTRIP

RTCM functionality provides precise positioning through NTRIP (Networked Transport of RTCM via Internet Protocol). This protocol is a standard for streaming differential GPS (DGPS) data over the Internet.

7.2.3.4.3.14 Registration

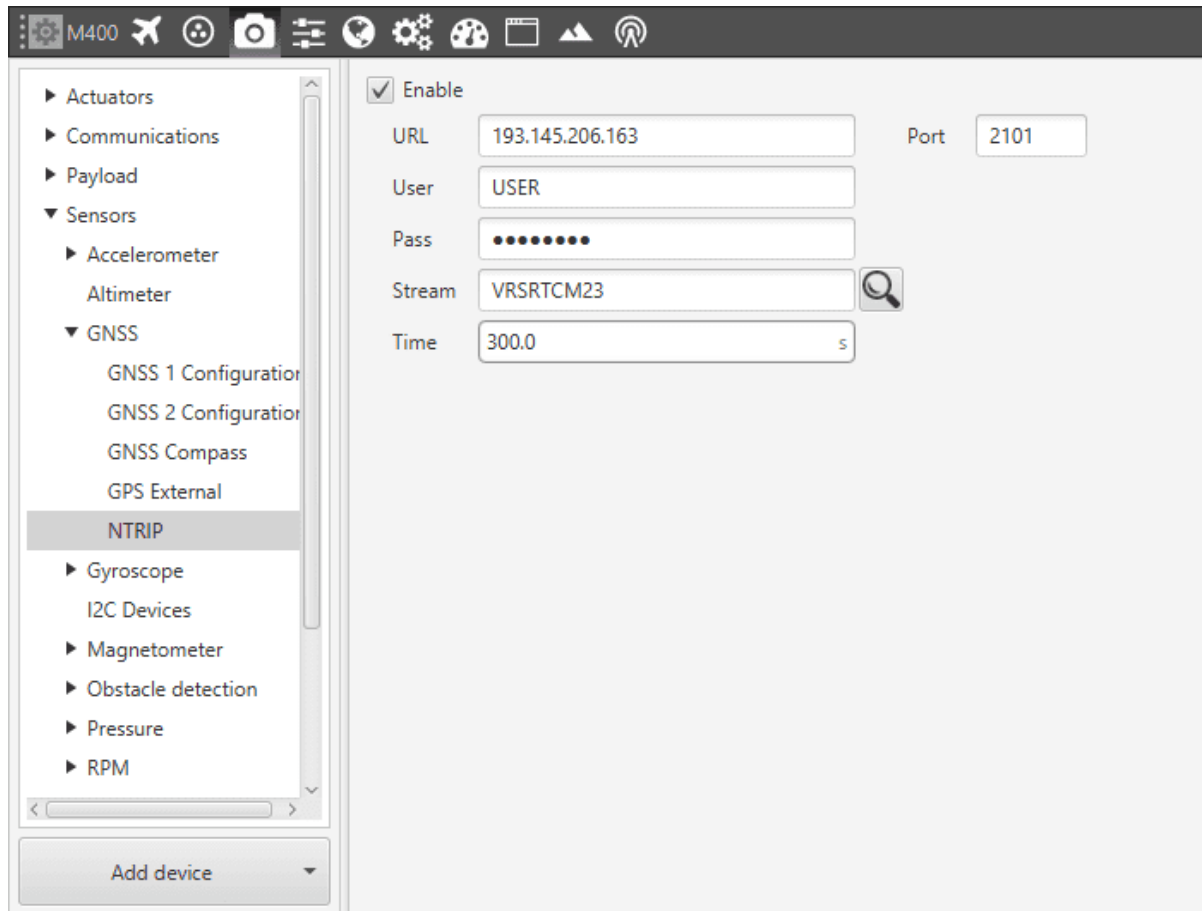
RTCM activation requires from a previous registration in one of the NTRIP stream providers available. Select the stream provider according to the location of your operation. Some examples of NTRIP providers are IGS.IP or EUREF.

Once registered, you will receive the following data from the NTRIP provider: URL, Port, User, Password

7.2.3.4.3.15 Configuration

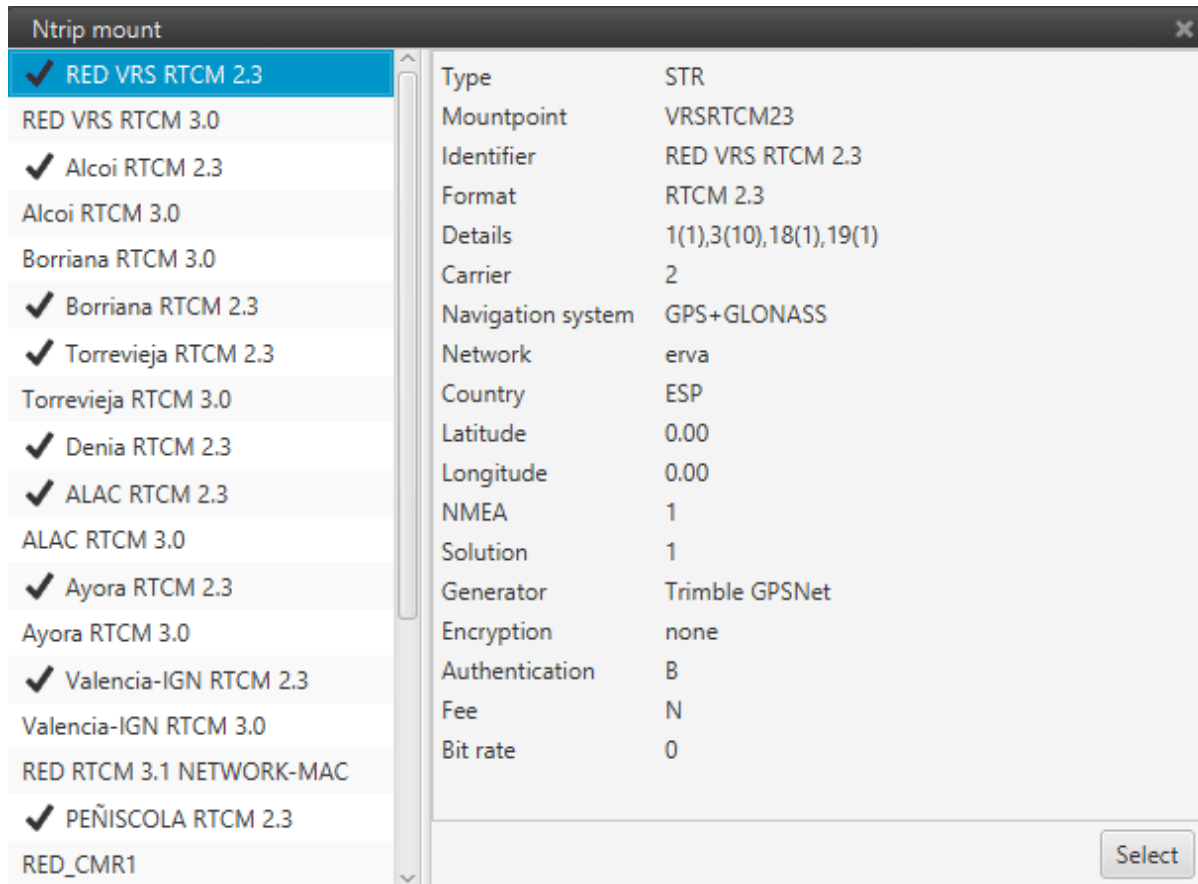
In order to activate precise positioning functionality in Veronte, NTRIP user data must be loaded in THE Veronte Autopilot. Go to the Setup menu in the Veronte Autopilot unit (Setup - Devices - Sensors - GNSS - NTRIP) and follow these steps:

1. Enable NTRIP
2. Enter NTRIP server data and user as follows



NTRIP - Configuration Menu

1. Click on the **Lens Icon** for connecting to the server
2. From the displayed list select the desired stream and click on select.

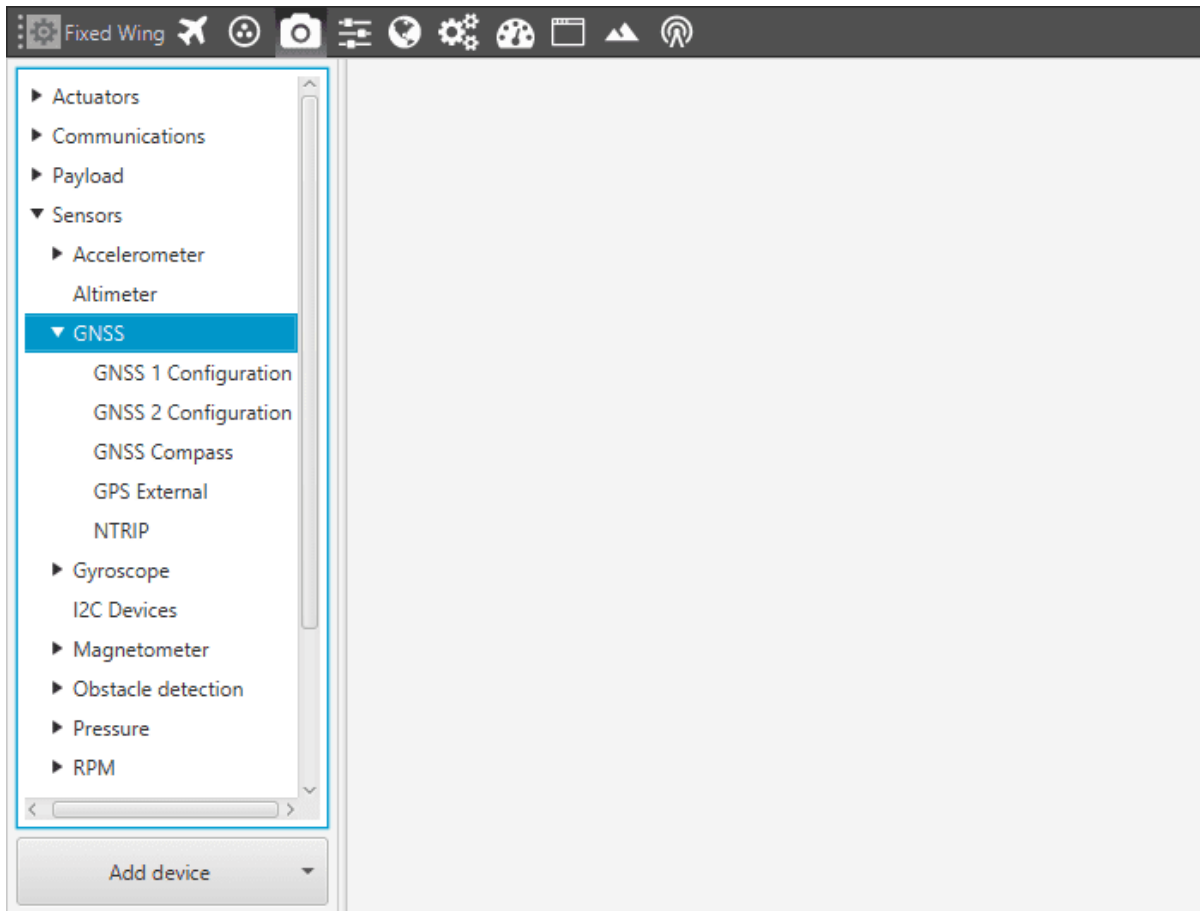


NTRIP - Stream List

Warning:

- The system is compatible with RTCM protocol 2.3. Compatible streams are marked with **Check**. It is recommended to select a virtual stream covering the area of operation.
- RTCM positioning requires an internet connection.

In this menu the user can modify all parameters which are relevant to GNSS sensors.



GNSS - Options available

Veronte has embedded 2 GNSS modules which are configured through GNSS 1-2 Configuration tabs. They can be configured as well for precision applications, such as RTK (high precision with correction from a base) or GNSS Compass (Compass information from differential positioning). External positioning such as GPS External or NTRIP have their own configuration tab as well.

7.2.3.4.4 Gyroscope

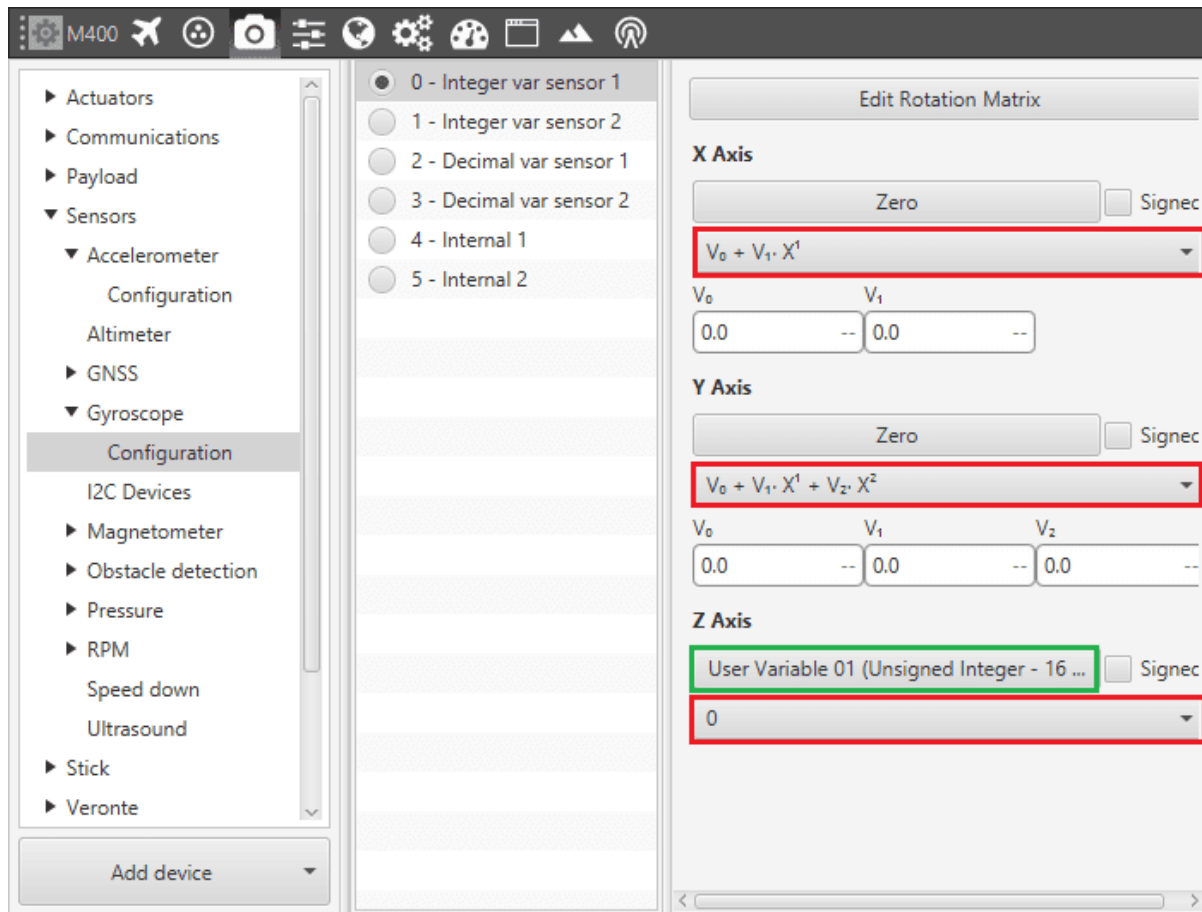
The gyroscope from the IMU can be configured as explained in the menus shown below.

The user can choose between 3 types of source for the gyroscope.

- **Integer var sensor 1-2.** Veronte uses a integer value provided by an external sensor.
- **Decimal var sensor 1-2.** Veronte uses a decimal value provided by an external sensor.
- **Internal 1-2 (Main/Secondary).** Veronte uses the internal sensor.

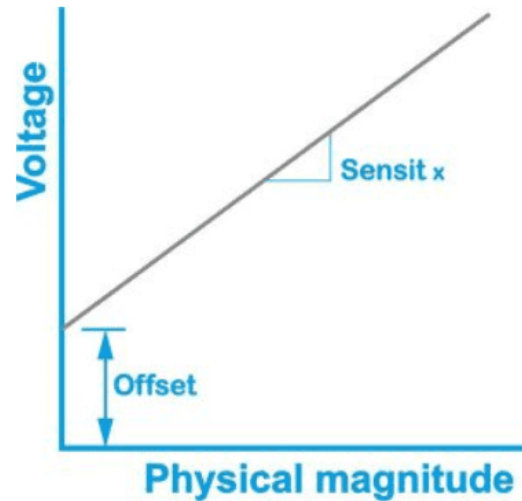
7.2.3.4.4.1 Integer var sensor

In this menu it is possible to configure integer variables provided by an external sensor.



Integer var Gyroscope Menu - Configuration Parameters

When Integer var sensor 1 or 2 are selected, the previous panel will be shown. In this panel, the user selects the variable that has been stored in a user variable (Green Box) and the operations that will be carried on (Red Box). It is possible to use the signal through a linear or quadratic relation. The following image shows an example of a linear relation.

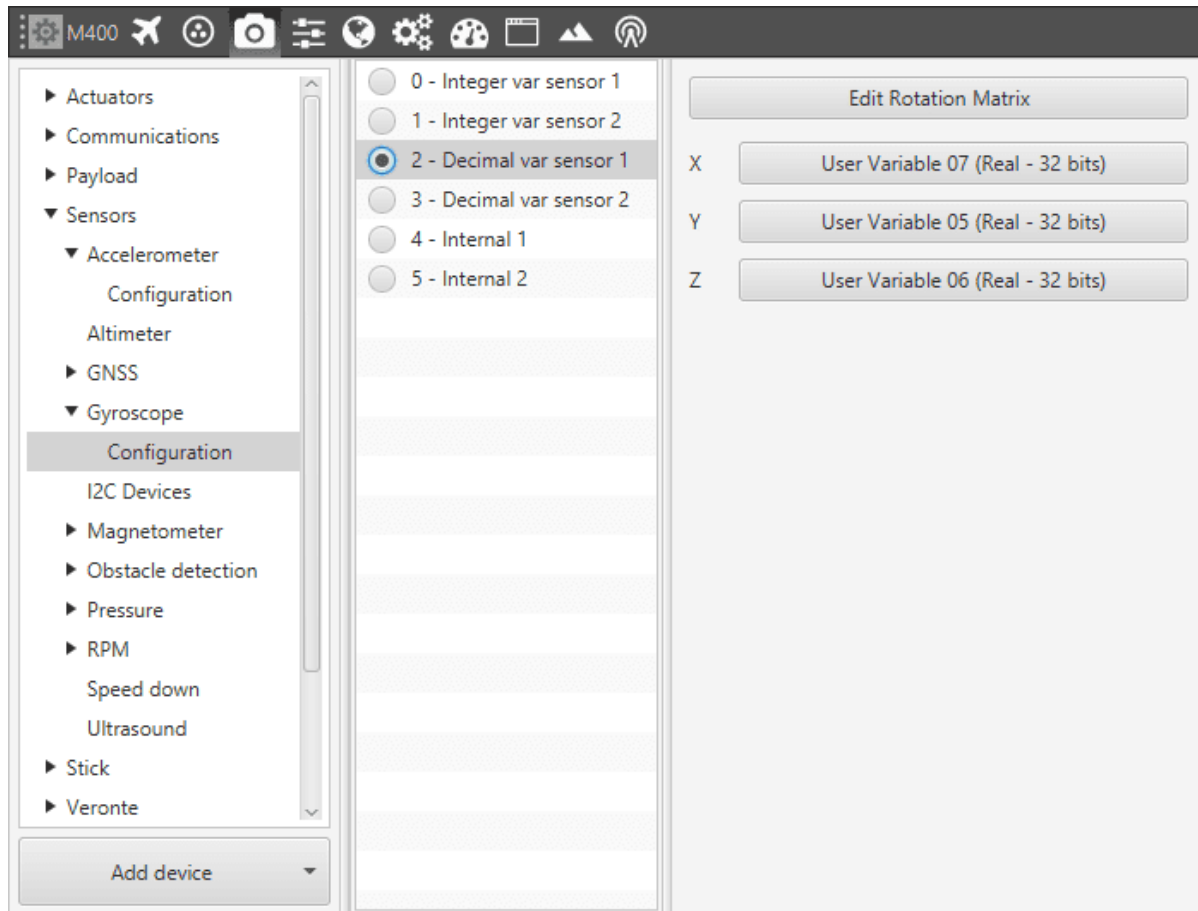


Linear relation of 2 Variables

The process of configuration has to be done using **Custom Messages**. This is to be configured in *Devices - Others - Digital - I/O Manager*. The configuration will depends on the device in use and its communication protocol.

7.2.3.4.4.2 Decimal var sensor

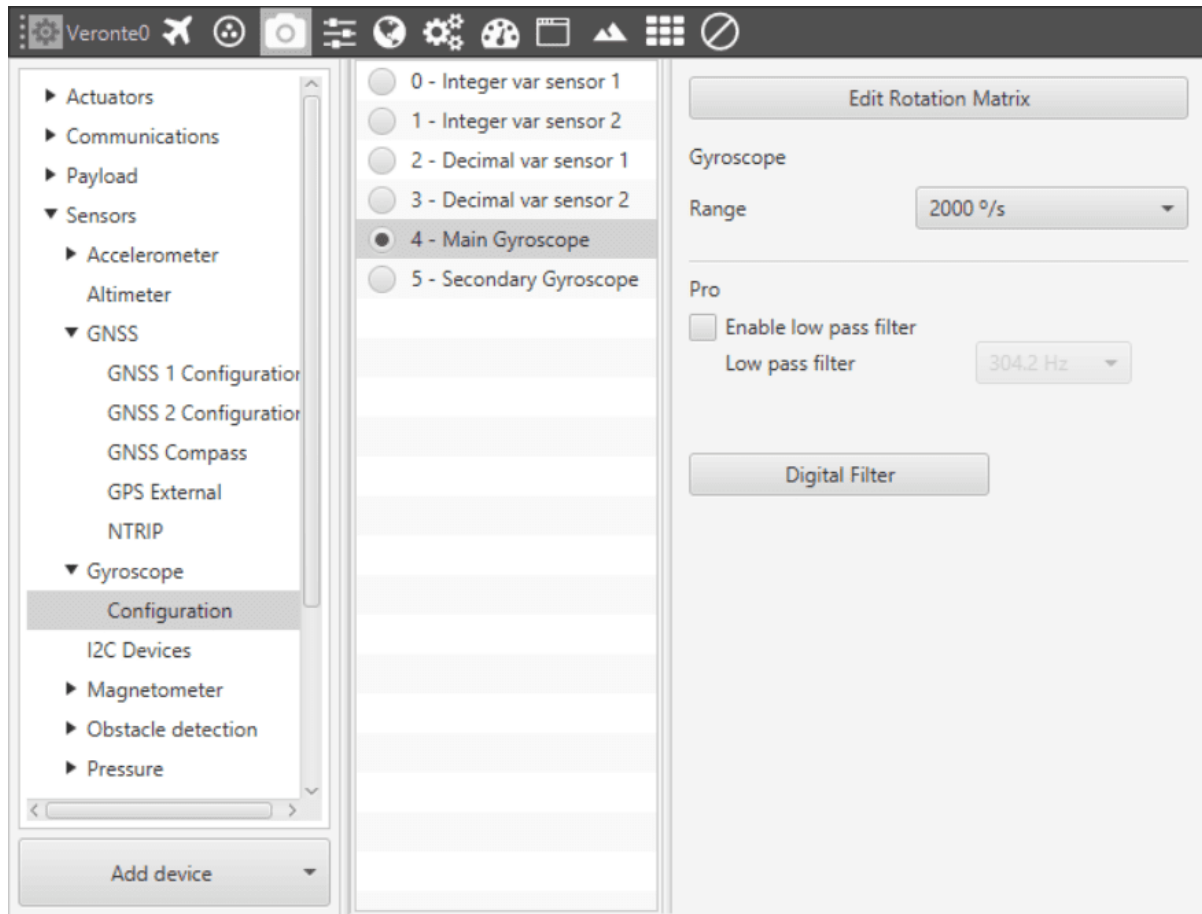
In this menu, the user selects real variables for each axis (X,Y,Z), these do not requiere a signal treatment. The process of configuration is similar to the one carried out when configuring a Integer Variable.



Decimal var Gyroscope Menu - Configuration Parameters

7.2.3.4.4.3 Internal

This last menu available displays the possible parameters that can be configured for the internal Gyroscope.



Gyroscope Menu - Configuration Parameters

Warning: **Edit Rotation Matrix** brings the position of the gyroscope inside the Veronte Autopilot, it must **NOT** be changed under any circumstance.

In this menu it is possible to set different options regarding range and filters from the gyroscope. The parameters that can be modified are:

- **Range.** Sets the maximum range of performance, high ranges implies less precision while small ranges might mean the system saturates. Values allowed are 125°/s, 250°/s, 500°/s, 1000°/s and 2000°/s.
- **Digital filter.** Enables a low pass filter which its cutoff frequency is configured manually, allowing the user to input any desired value in Hz. It is a **software filter**.

7.2.3.4.5 I2C Devices

Warning: Be aware that custom messages are not available for I2C. Only the devices listed below can be integrated with Veronte Autopilot using the I2C interface.

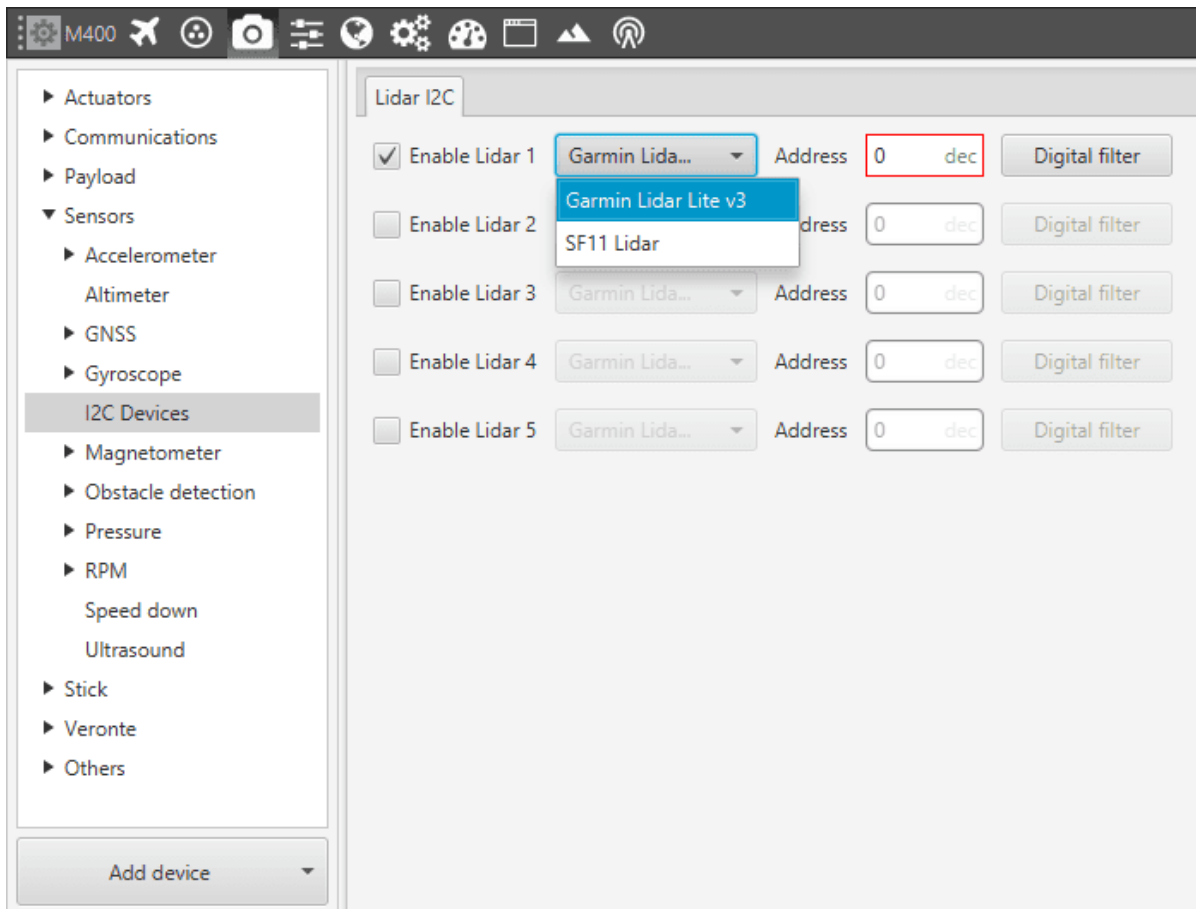
The I2C bus allows the connection of several devices with different addresses to the same line via master-slave communication. At this moment, Veronte supports the following devices:

- **Garmin LIDAR-Lite v3.** Optical distance measurement sensor with a range of 5cm to 40m.
- **SF11 Lidar.** Long range laser altimeter. Supported SF11/B and SF11/C with a maximum range of 50m and 120m respectively.
- **Magnetometer LIS3MDL.** Three-axis magnetic sensor with a very small package.
- **Magnetometer HSCDTD008A.** Three-axis terrestrial magnetism sensor of the digital output.

7.2.3.4.5.1 Lidar Devices

Both Lidar must be configured in *I2C Devices* menu. In our case, Veronte allows up to 5 Lidar devices to be connected to the system at the same time.

The configuration menu can be seen below:



I2C - Configuration Menu

After enabling the needed number of Lidar devices, configurable parameters are:

- **Type of Lidar.** Veronte is compatible with Garmin Lidar Lite v3 and SF11 Lidar.
- **Address.** With an accepted value between 16 - 239, this is the origin address from the Lidar being configured.
- **Digital filter.** Enables a low pass filter which its cutoff frequency is configured manually, allowing the user to input any desired value in Hz. It is a **software filter**.

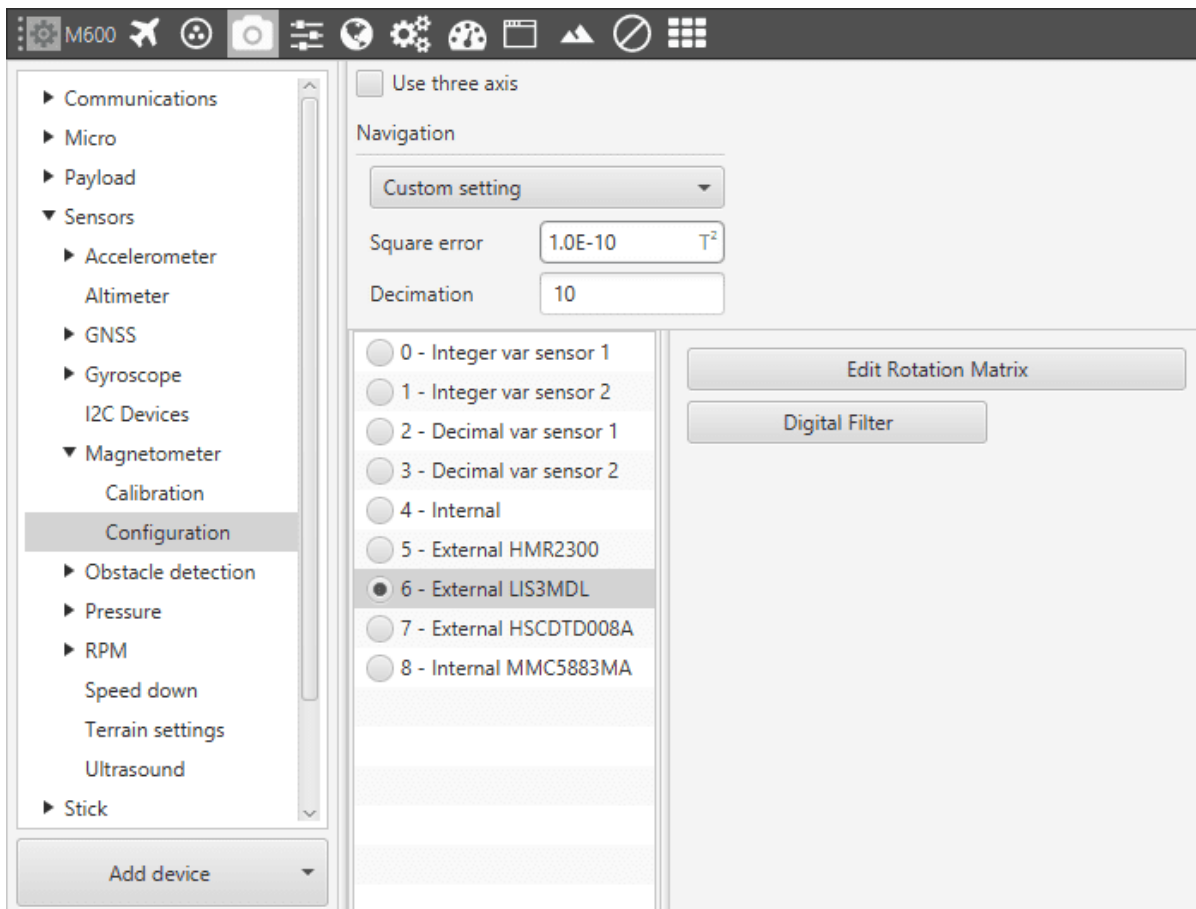
The Lidar number need to be kept in order to properly configure the Altimeter menu afterwards in Setup - Devices - Sensors - [Altimeter](#).

7.2.3.4.5.2 Magnetometer Devices

The magnetometers are configured in Setup - Devices - Sensors - [Magnetometer](#). It is **very important** to know that **address cannot be chosen in the software** and must be as follows:

- **Magnetometer LIS3MDL:** 0x1C.
- **Magnetometer HSCDTD008A:** 0x0C.

In order to select a magnetometer connected by I2C bus, the user simply has to select it in the configuration menu (see the figure below):



I2C - Magnetometer Configuration

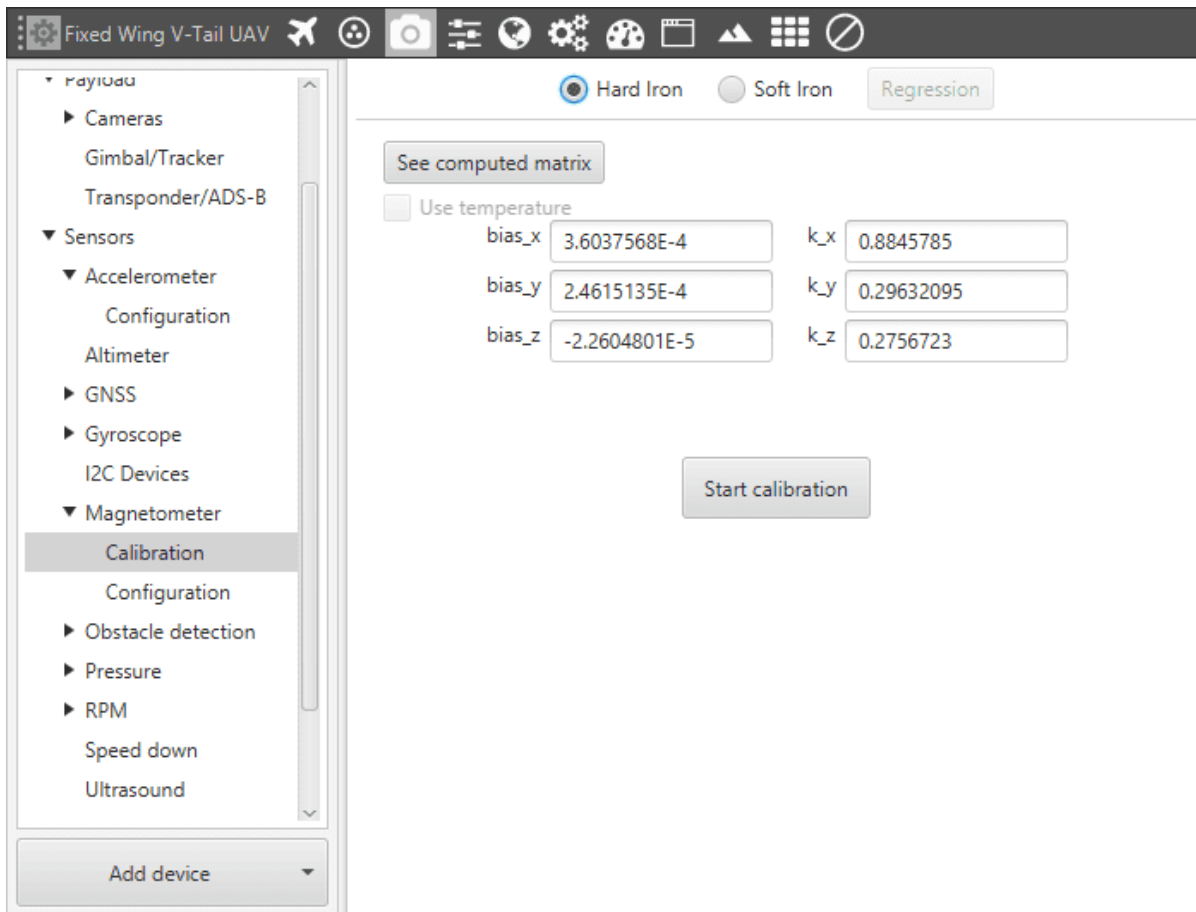
Once the magnetometer is chosen, the user can modify the following parameters:

- **Rotation matrix.** It must be modified in the case that the axes of the magnetometer do not coincide with those of the aircraft.
- **Digital filter.** Enables a low pass filter which its cutoff frequency is configured manually, allowing the user to input any desired value in Hz. It is a **software filter**.

7.2.3.4.6 Magnetometer

7.2.3.4.6.1 Calibration

Magnetometer calibration should be performed once Veronte has been installed on the platform so the magnetic field during the operation is similar to the one measured during the calibration. Hard calibration includes distortions created by objects that produce a magnetic field (these objects are the uav or their components). Soft calibration also includes the alterations in the existing magnetic field.



Magnetometer - Calibration Menu

In order to start calibration, press on the **Start Calibration** button so the system can capture magnetometer data. During the calibration, the system must be oriented in all possible directions so enough data can be captured. Once enough data has been captured, **Compute Data** sets the calibration.

The procedure for acquiring enough data for performing the calibration is:

1. Rotate Veronte around one of its axis (e.g. the x-axis) and make at least three turns around this axis.


2. Repeat the same process for another of its axis (e.g. y-axis).
3. Finish the calibration by repeating this process for the remaining axis (e.g. z-axis).

The different rotations that Veronte must be subjected can be visualized in the figure below.

Magnetometer - Calibration Procedure

Once three circles have been drawn on the screen, captured data will be enough for saving the calibration data. The following image shows an example of the calibration result:

| | | | | |
|-------------------|--------|----------------------|-----|---------------------|
| Start calibration | bias_x | 0.29702584814227595 | k_x | 0.586923625767523 |
| Compute | bias_y | -0.12203297925805784 | k_y | 0.2973864362346599 |
| | bias_z | 0.304061118583464 | k_z | 0.36988663216152706 |



| | | | | | | |
|-------------------|---|------------|---|-------------|---|------------|
| Precalibrate: | X | 0.16959064 | Y | -0.39181286 | Z | 0.14473684 |
| Actual calibrate: | X | 0.16959064 | Y | -0.39181286 | Z | 0.14473684 |
| New calibrate: | X | 0.09799161 | Y | -0.3247512 | Z | 0.5023342 |

Magnetometer - Calibration Values

7.2.3.4.6.2 Configuration

Veronte incorporates an internal magnetometer that allows the Veronte System to measure the magnetic field.

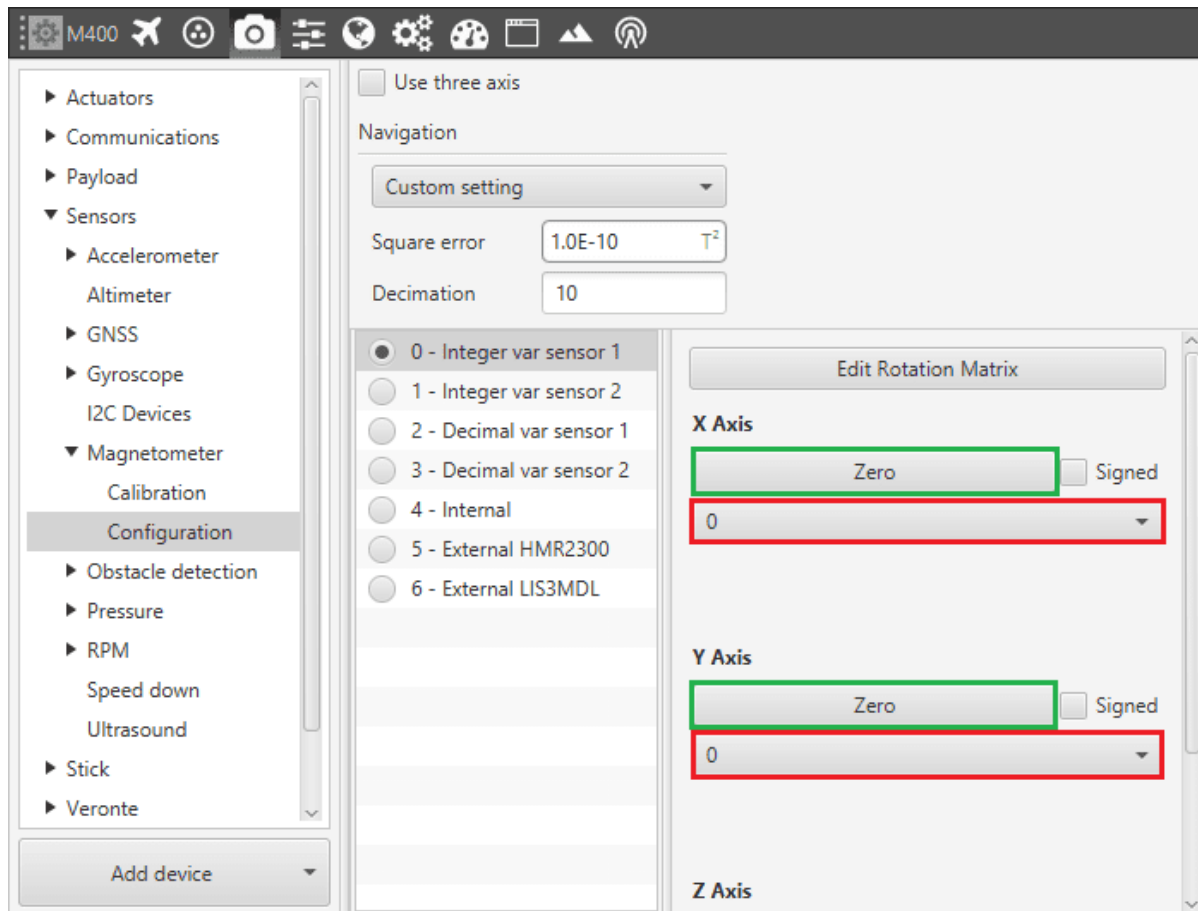
The user can choose between 4 types of source for the accelerometer.

- **Integer var sensor 1-2.** Veronte uses a integer value provided by a no-integrated external sensor.
- **Decimal var sensor 1-2.** Veronte uses a decimal value provided by a no-integrated external sensor.
- **Internal** Veronte uses the internal sensor.
- **External HMR2300/LIS3MDL/HSCDTD008A.** Veronte uses the information from one of the compatible external magnetometers.

There are 3 parameters that are configured independently from the Magnetometer selected. These are the one in the Navigation section: - **State**. User can choose from **Disabled** for magnetometer not entering the Navigation filters (not being used but not turning it off) or **Custom settings**, which will take into account all the following parameters. - **Square error**. Means the influence of the parameter on the Navigation filters. The greater the less effect it will have. - **Decimation**. Defines the bunch of data from which 1 value will be stored. For example, if decimation is 10, every 10 measurements 1 will be taken into account. This procedure is used to reduce the number of samples.

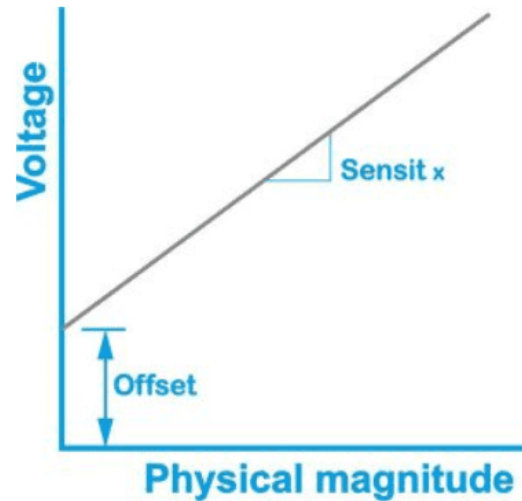
7.2.3.4.6.3 Integer var sensor

In this menu it is possible to configure integer variables provided by an external sensor.



Integer var Magnetometer Menu - Configuration Parameters

When Integer var sensor 1 or 2 are selected, the previous panel will be shown. In this panel, the user selects the variable that has been stored in a user variable (Green Box) and the operations that will be carried on (Red Box). It is possible to use the signal through a linear or quadratic relation. The following image shows an example of a linear relation.

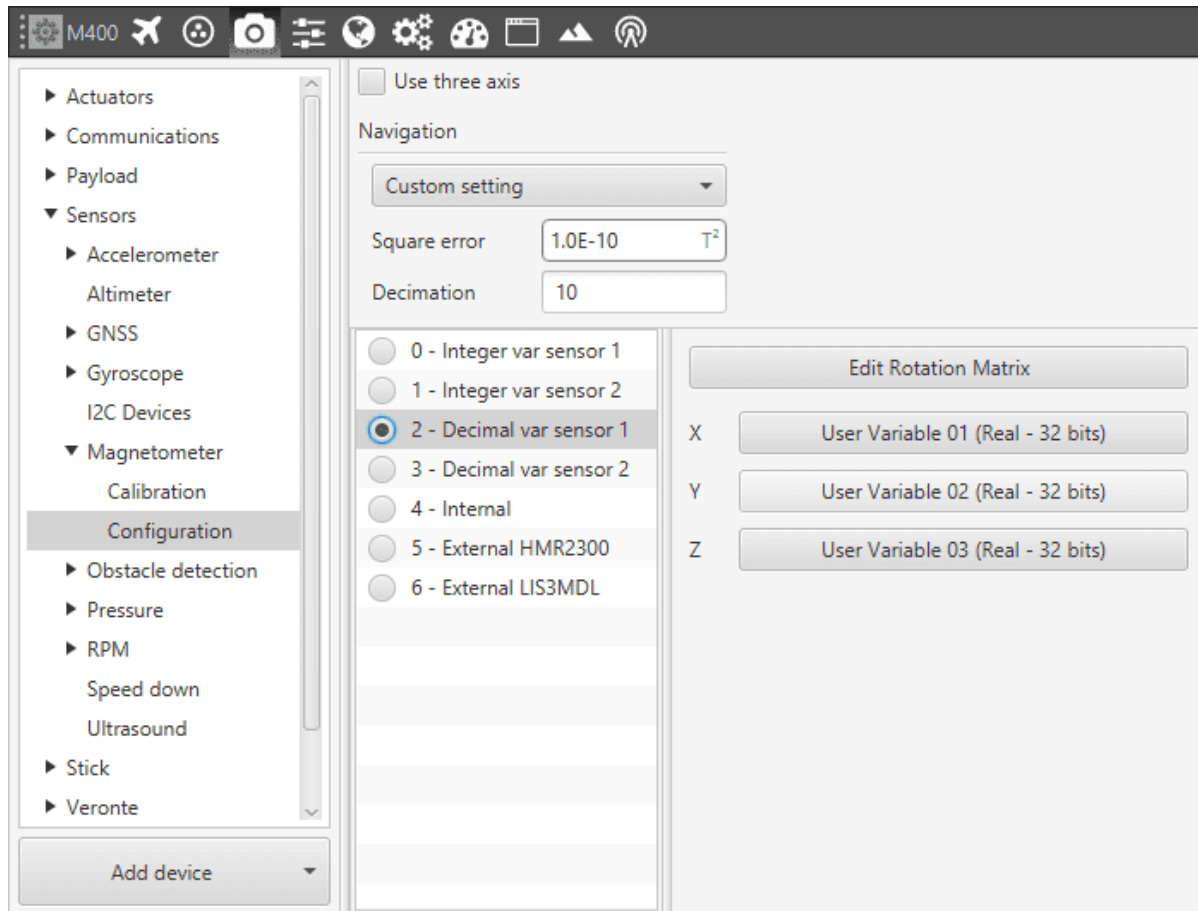


Linear relation of 2 Variables

The process of configuration has to be done using **Custom Messages**. This is to be configured in *Devices - Others - Digital - I/O Manager*. The configuration will depends on the device in use and its communication protocol.

7.2.3.4.6.4 Decimal var sensor

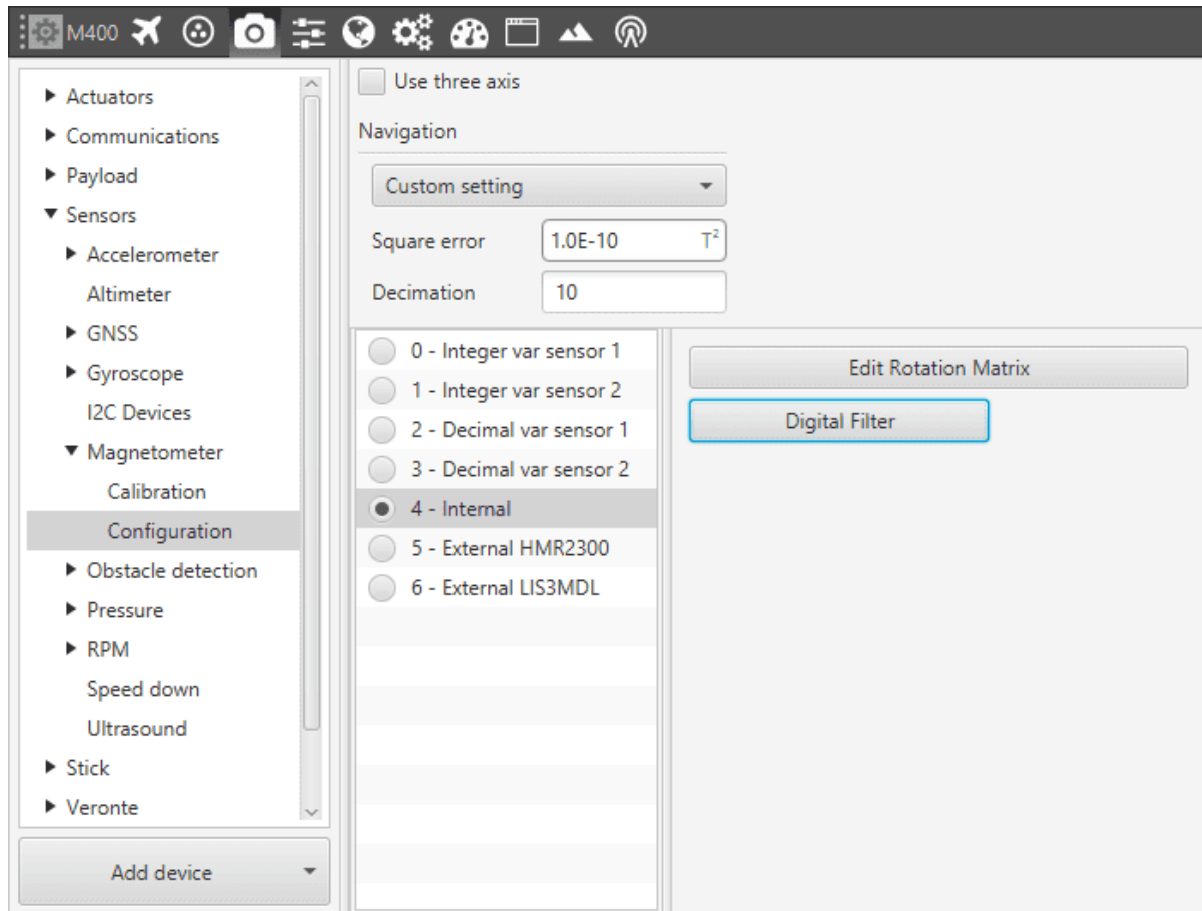
In this menu, the user selects real variables for each axis (X,Y,Z), these do not requiere a signal treatment. The process of configuration is similar to the one carried out when configuring a Integer Variable.



Decimal var Magnetometer Menu - Configuration Parameters

7.2.3.4.6.5 Internal

This menu available displays the possible parameters that can be configured for the internal Magnetometer.



Internal var Magnetometer Menu - Configuration Parameters

Warning: **Edit Rotation Matrix** brings the position of the magnetometer inside the Veronte Autopilot, it must **NOT** be changed under any circumstance.

In this menu it is possible to set different options regarding filters.

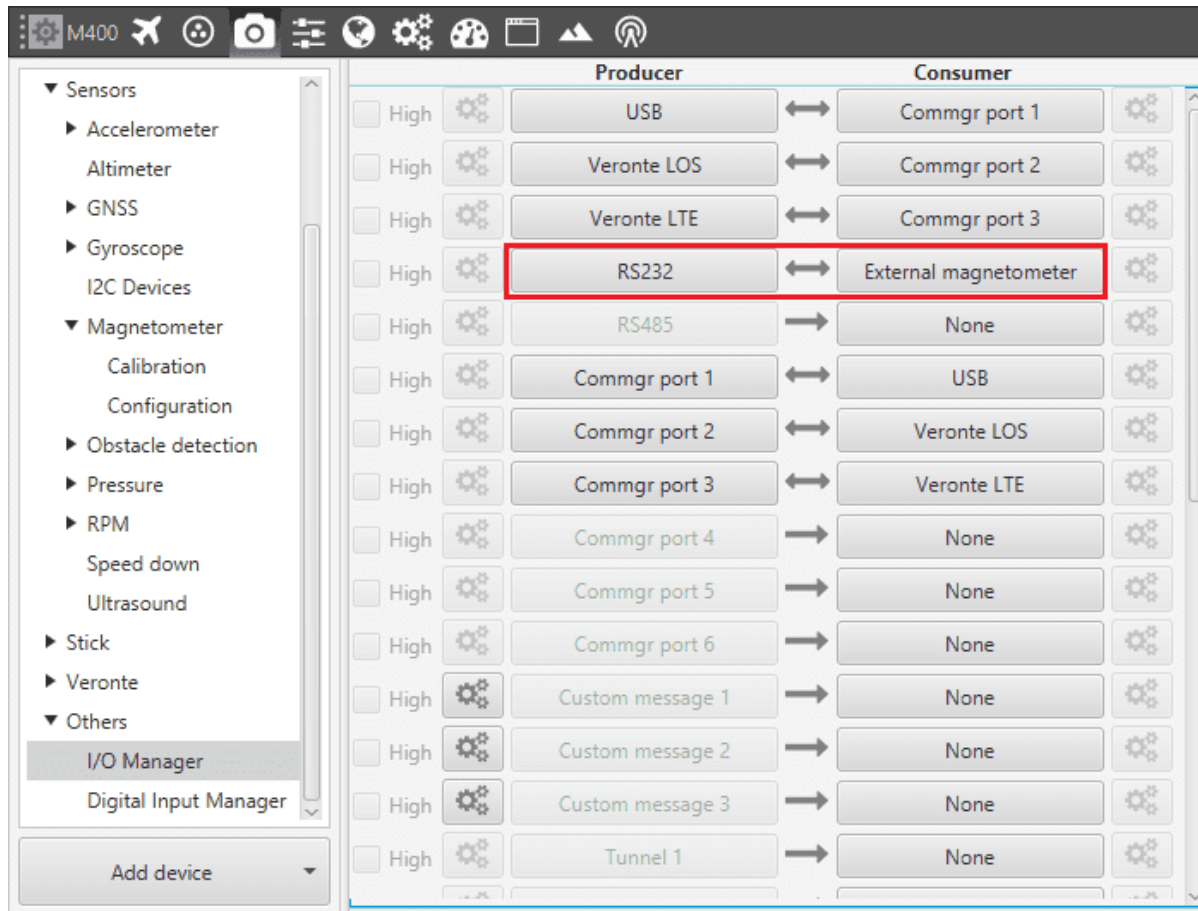
- **Digital filter.** Enables a low pass filter which its cutoff frequency is configured manually, allowing the user to input any desired value in Hz. It is a **software filter**.

7.2.3.4.6.6 External

Veronte has been designed to have compatibility with the external magnetometers HMR200 and LIS3MDL. The first one has no filters configurable, only the option Edit Rotation Matrix to set the orientation of it.

The LIS3MDL option is the sensor Veronte has inside it, but mounted externally to avoid the possible interferences from being close to electronic components. It has the Edit Rotation Matrix and the Digital Filter.

On the other hand, the connection to the serial port is configured in *Devices - Others - I/O Manager*



External Magnetometer - Channel Configuration

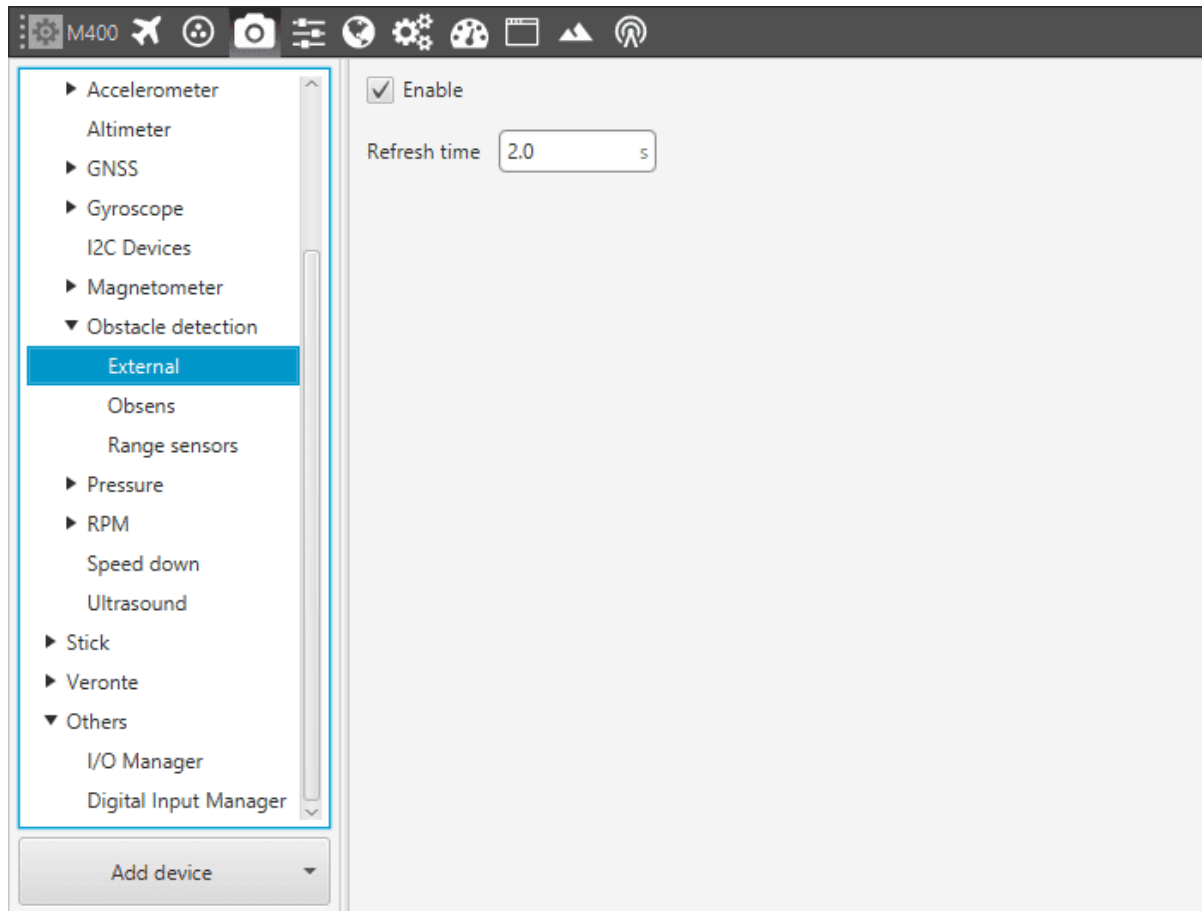
7.2.3.4.7 Obstacle Detection

Obstacle detection allows the creation of obstacles that the platform will consider for its Navigation. It could be an external source of obstacles or the sensors connected to Veronte with the selection of the sensors range.

7.2.3.4.7.1 External

If a user connects an Obstacle provider system to Veronte, it has to be enabled from this menu.

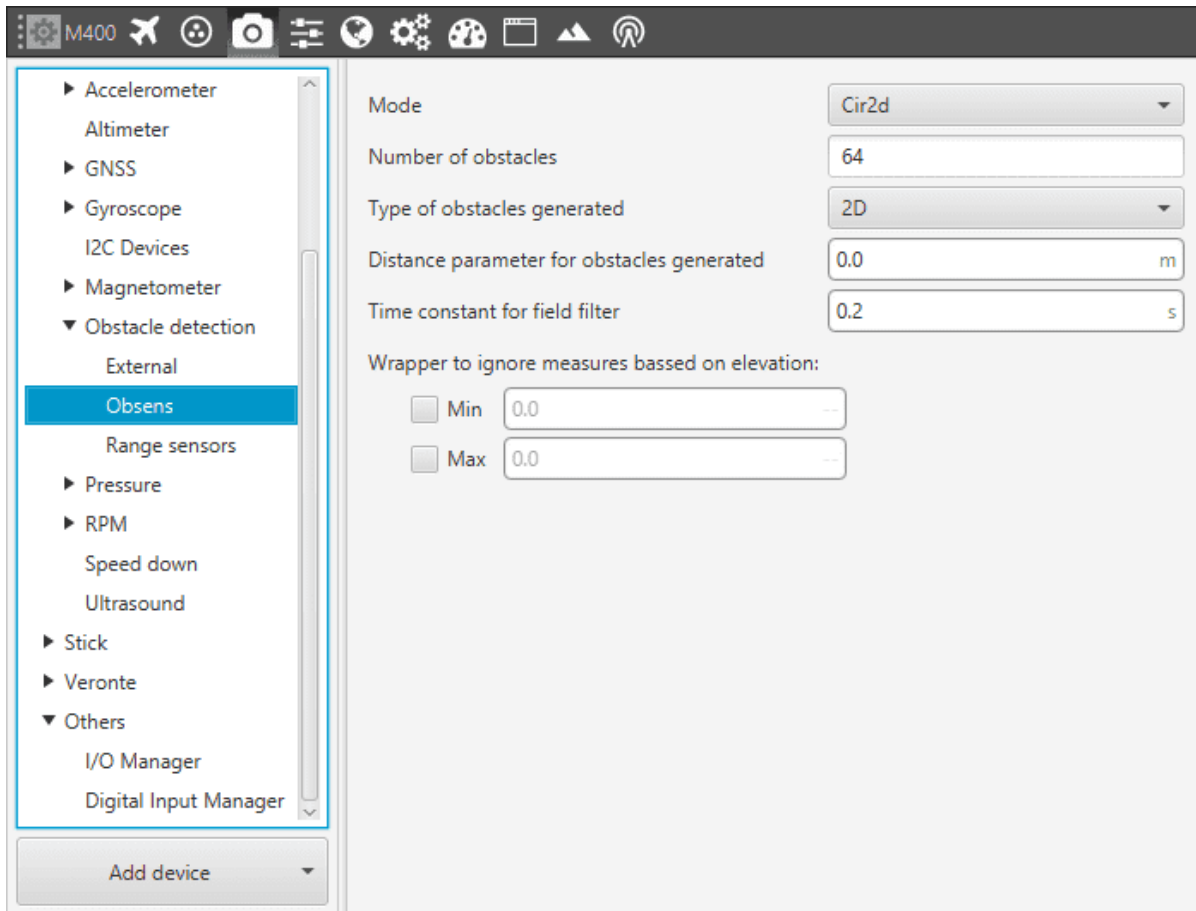
The **Refresh time** is the time it takes Veronte to check again the obstacle status from the external device. This is used to know when an obstacle has disappear from range or a new obstacle is inside it.



Obstacle Detection - External

7.2.3.4.7.2 Obsens

This panel contains the configuration menu to set the obstacles creation.



Obstacle Detection - Obsens

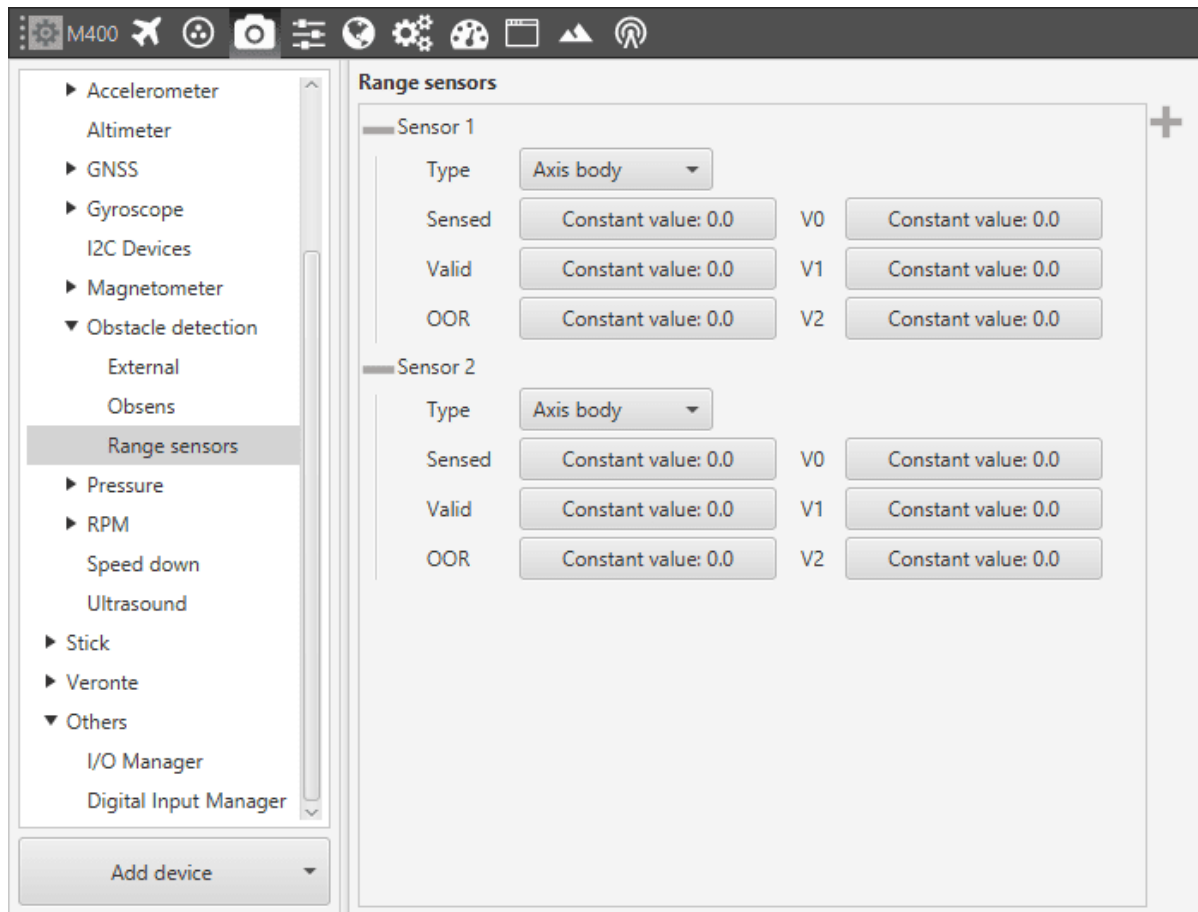
In this panel, it can be configured the following Obsens parameters:

- **Mode:** Cir2d (circular – 2 dimensions), Raw and off mode can be selected.
 - **Raw.** The sensors create the obstacle where it is located. At next step, if the obstacle is not present, the area is cleared.
 - **Cir2d.** Allows to input the number of divisions the area around the platform will have (considering a circular area). If one area is blocked due to an obstacle presence, even the sensors lost that obstacle, the area will remain blocked until enough distance has been reached.
 - **Off.** The obstacle detection is not enabled.
- **Type of obstacles generated.** 2D or 3D obstacles can be generated.
- **Distance parameter for obstacle generated.** Defines de radius of the obstacles generated with Obsens.
- **Time constant for field filter.** Filter applied to smooth the transition when obstacles are affecting the control.
- **Wrapper.** A minimum and a maximum elevation wrapper can be activated in order to limit the obstacle generation in a limited cone angle.

When configuring this sensor, obstacles will be created automatically, working similarly to the ones created in the mission toolbar.

7.2.3.4.7.3 Range sensors

The following figure shows the menu to configure the sensors which are used during the obstacle detection.



Obstacle Detection - Range Sensors

In this menu can be configured the following parameters:

- **Type.** Allows the user to select the kind of orientation which will be used: Axis body, Angles body or Angles NED
- **V0, V1,V2.** Once an orientation is selected, it is possible to edit where the sensor is pointing to by editing these values.
- **Sensed.** To select the variable which is measured.
- **Valid.** Usually sensors incorporates a variable to know if the data measured is valid. If that variable has been stored, it is possible to select it.
- **OOR.** Allows the user to select the variable that indicates if the data measured is out of range.

7.2.3.4.8 Pressure

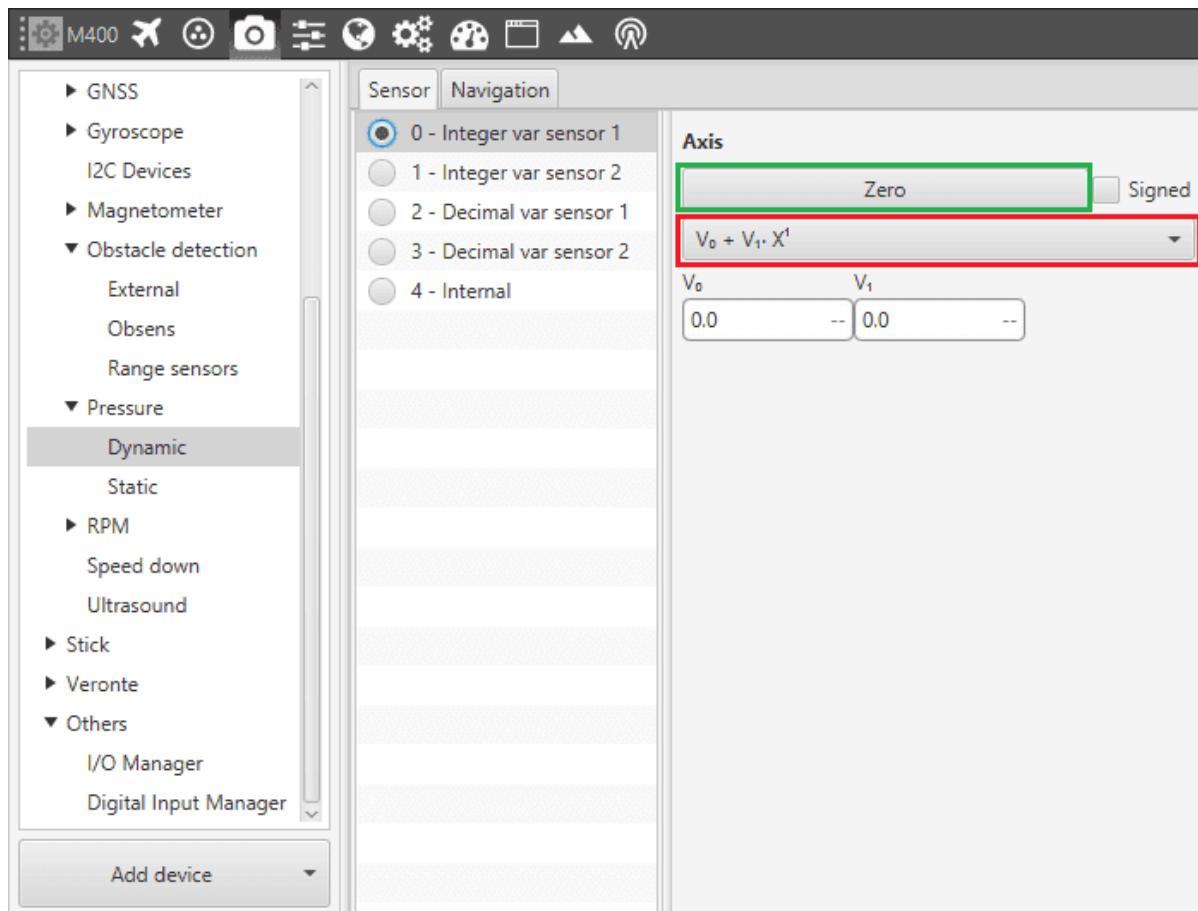
7.2.3.4.8.1 Dynamic

This menu allows the user to configure a dynamic pressure sensor input in Veronte.

7.2.3.4.8.2 Sensor

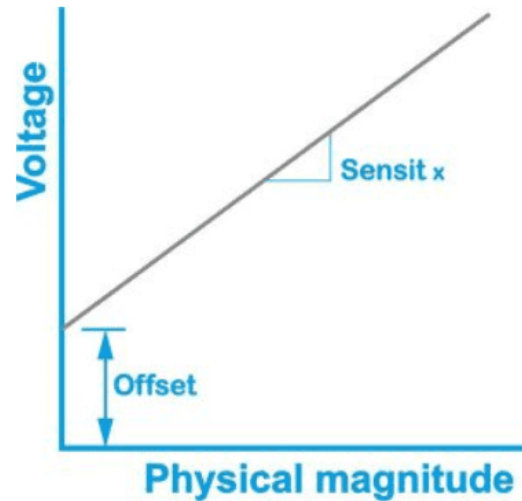
7.2.3.4.8.3 Integer var sensor

In this menu it is possible to configure integer variables provided by an external sensor.



Integer var Dynamic Pressure Menu - Configuration Parameters

When Integer var sensor 1 or 2 are selected, the previous panel will be shown. In this panel, the user selects the variable that has been stored in a user variable (Green Box) and the operations that will be carried on (Red Box). It is possible to use the signal through a linear or quadratic relation. The following image shows an example of a linear relation.

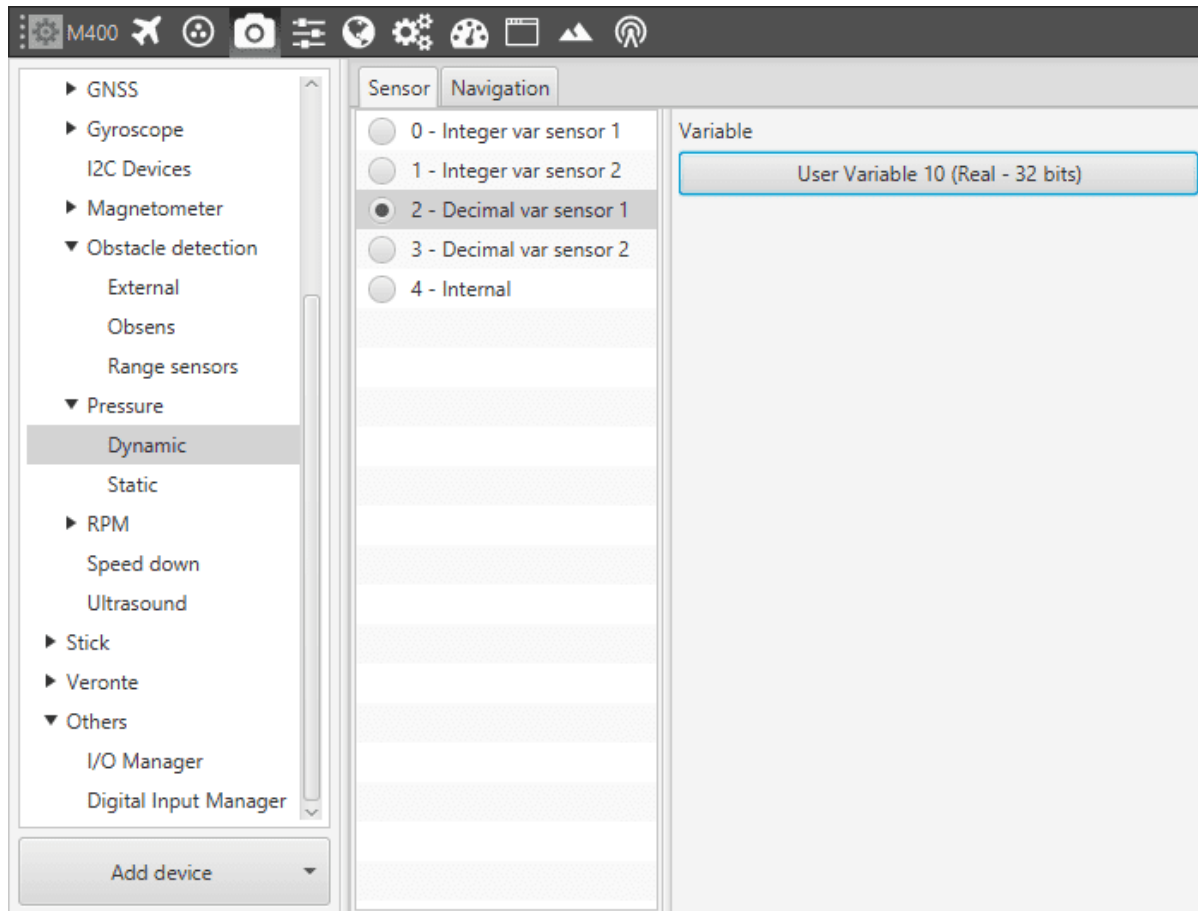


Linear relation of 2 Variables

The process of configuration has to be done using **Custom Messages**. This is to be configured in *Devices - Others - Digital - I/O Manager*. The configuration will depend on the device in use and its communication protocol.

7.2.3.4.8.4 Decimal var sensor

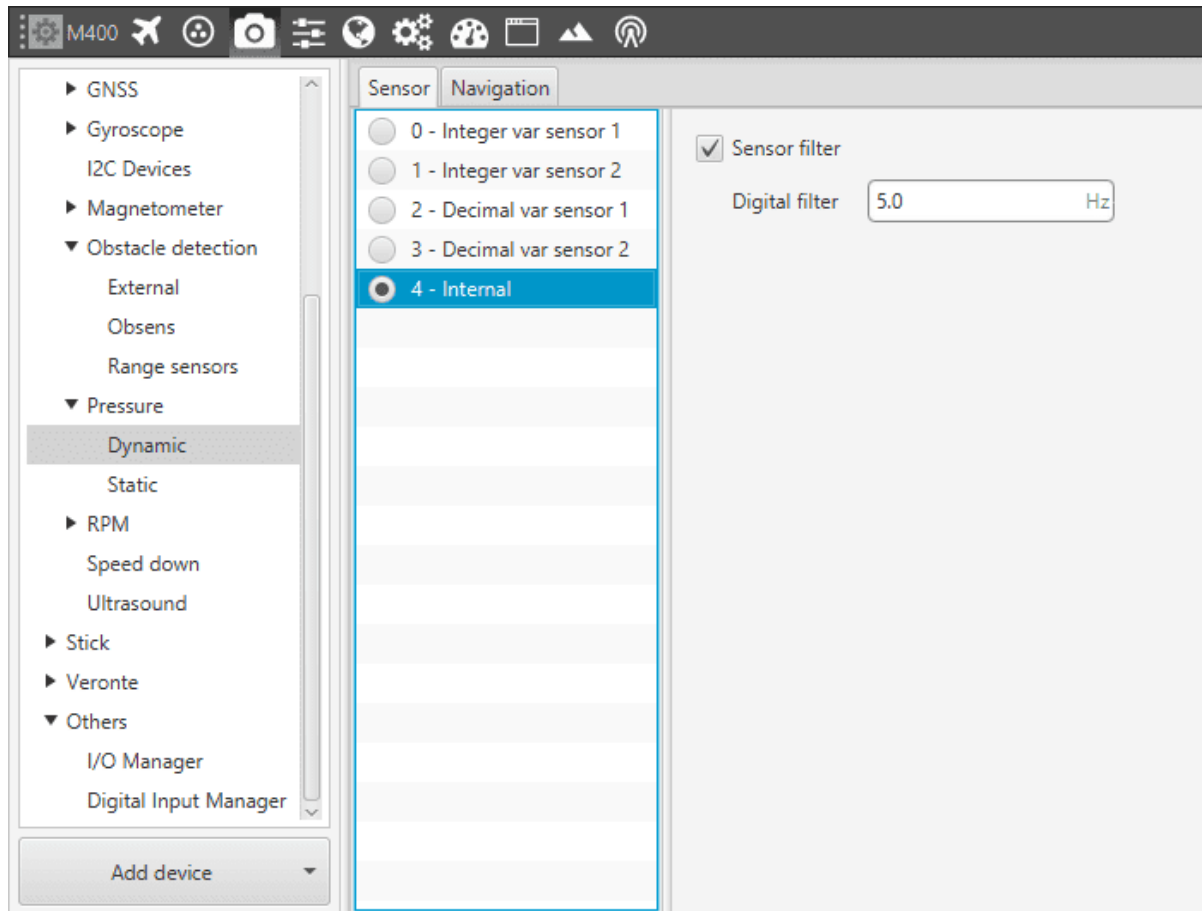
In this menu, the user selects a real variable, this does not require a signal treatment. The process of configuration is similar to the one carried out when configuring a Integer Variable.



Decimal var Dynamic Pressure Menu - Configuration Parameters

7.2.3.4.8.5 Internal

This menu displays the possible parameters that can be configured for the internal Dynamic Pressure sensor.



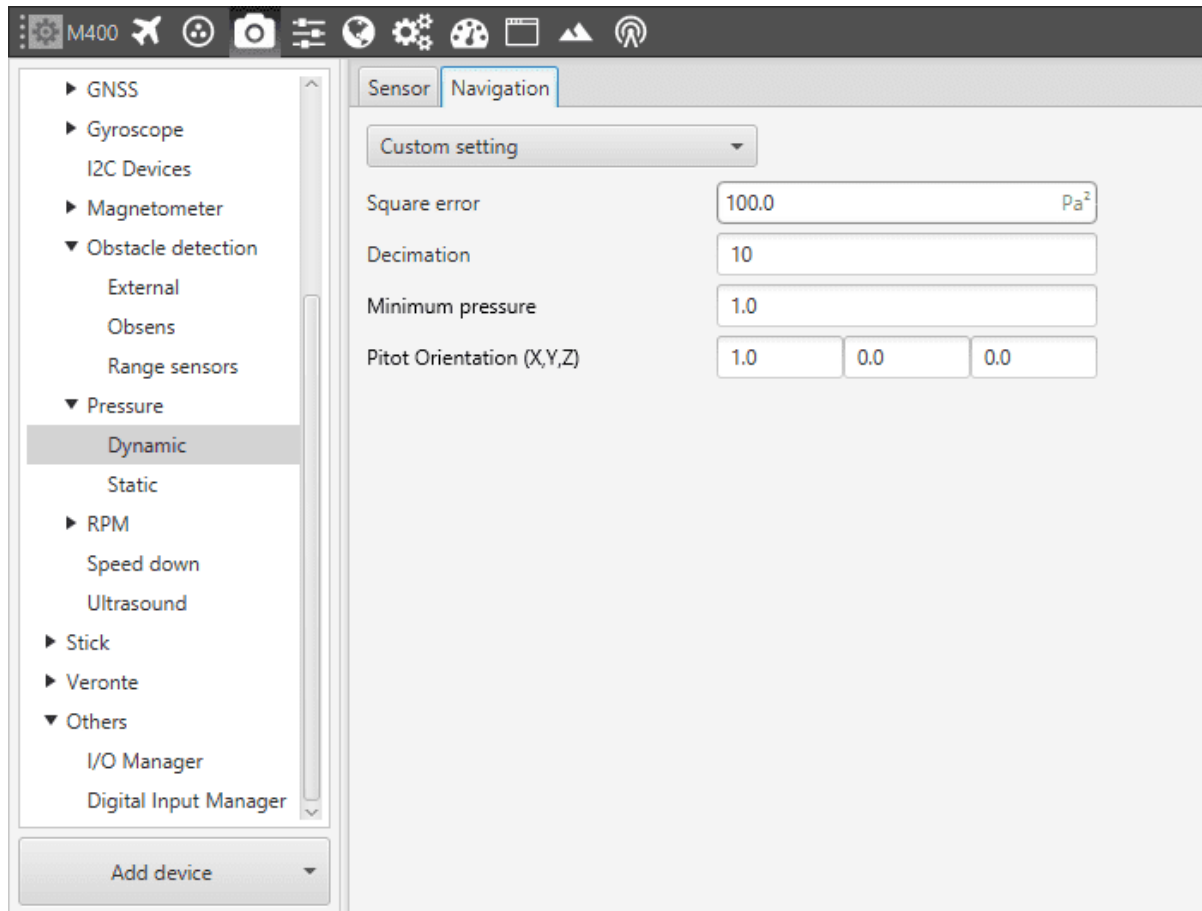
Internal Dynamic Pressure Menu - Configuration Parameters

In this menu it is possible to set different options regarding filters.

- **Digital filter.** Enables a low pass filter which its cutoff frequency is configured manually, allowing the user to input any desired value in Hz. It is a **software filter**.

7.2.3.4.8.6 Navigation

This menu allows the user to configure the navigation parameters from a dynamic pressure sensor input in Veronte.



Dynamic Pressure Menu - Navigation Configuration Parameters

- **Enable.** Choose the dynamic sensor pressure use on the system. Values from the sensor will be received, but they will not enter the Navigation Filter.
- **Square Error.** Sensor error square, which defined the weight if the measurement into the Navigation filter.
- **Decimation.** Defines the bunch of data from which 1 value will be stored. For example, if decimation is 10, every 10 measurements 1 will be taken into account. This procedure is used to reduce the number of samples.
- **Minimum pressure.** Minimum pressure readable from the sensor.
- **Pitot Orientation.** Vector defining the Pitot orientation on the platform.

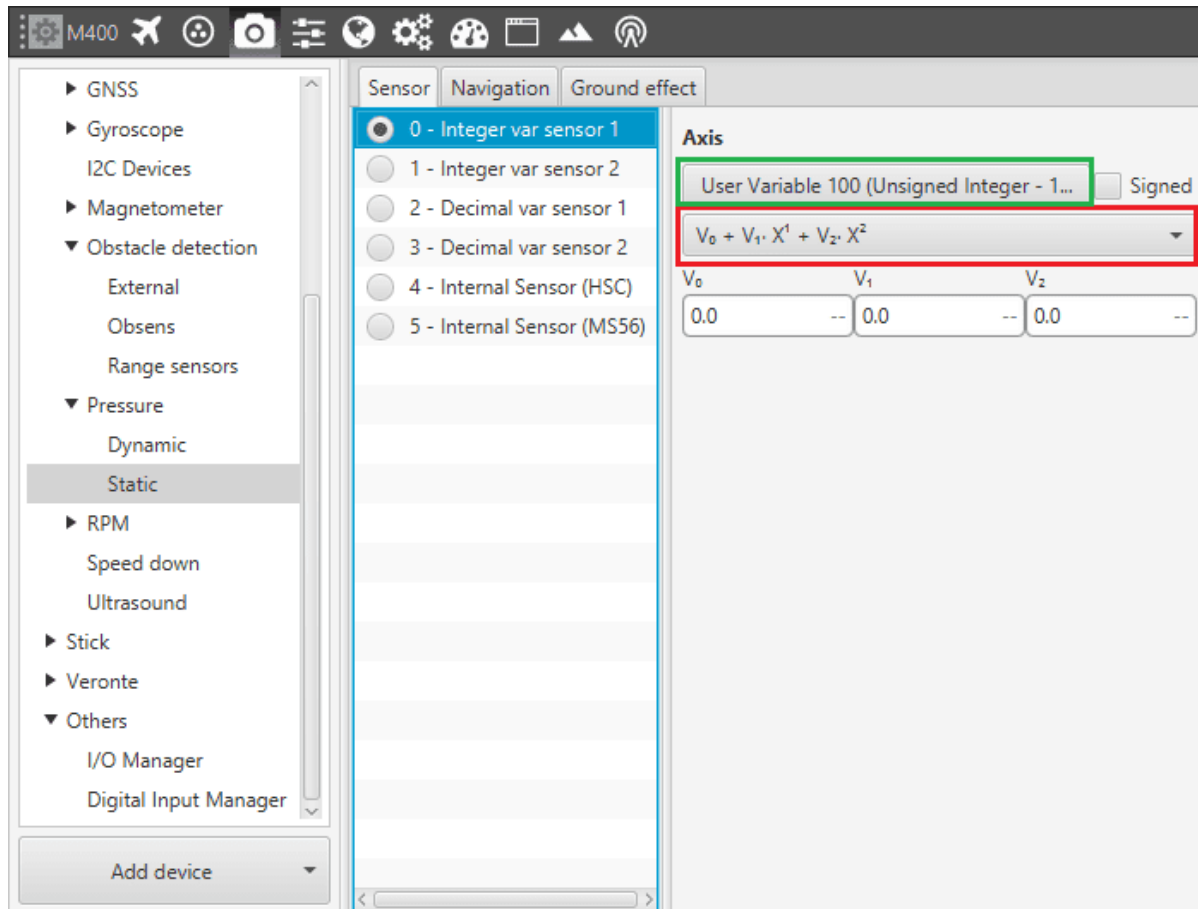
7.2.3.4.8.7 Static

This menu allows the user to configure a static pressure sensor input in Veronte.

7.2.3.4.8.8 Sensor

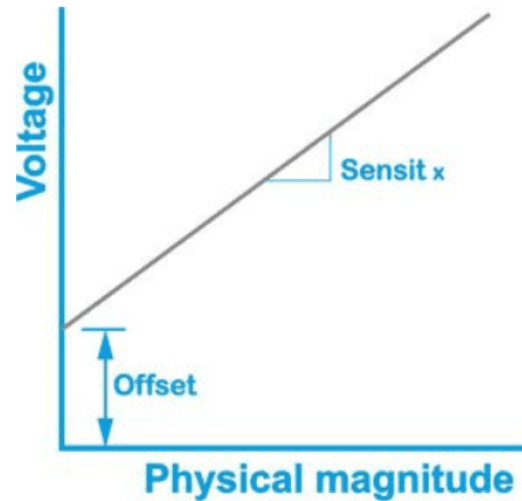
7.2.3.4.8.9 Integer var sensor

In this menu it is possible to configure integer variables provided by an external sensor.



Integer var Static Pressure Menu - Configuration Parameters

When Integer var sensor 1 or 2 are selected, the previous panel will be shown. In this panel, the user selects the variable that has been stored in a user variable (Green Box) and the operations that will be carried on (Red Box). It is possible to use the signal through a linear or quadratic relation. The following image shows an example of a linear relation.

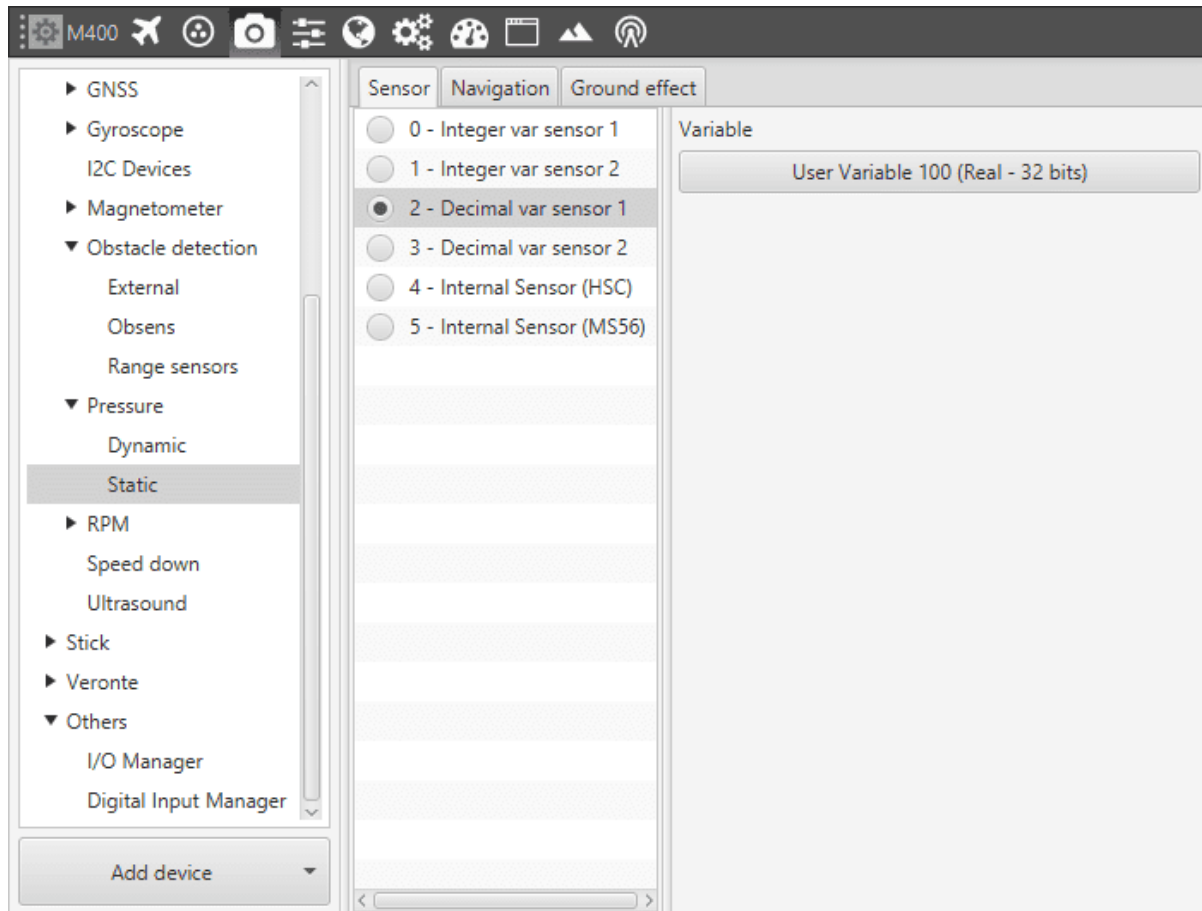


Linear relation of 2 Variables

The process of configuration has to be done using **Custom Messages**. This is to be configured in *Devices - Others - Digital - I/O Manager*. The configuration will depends on the device in use and its communication protocol.

7.2.3.4.8.10 Decimal var sensor

In this menu, the user selects a real variable, this does not requiere a signal treatment. The process of configuration is similar to the one carried out when configuring a Integer Variable.

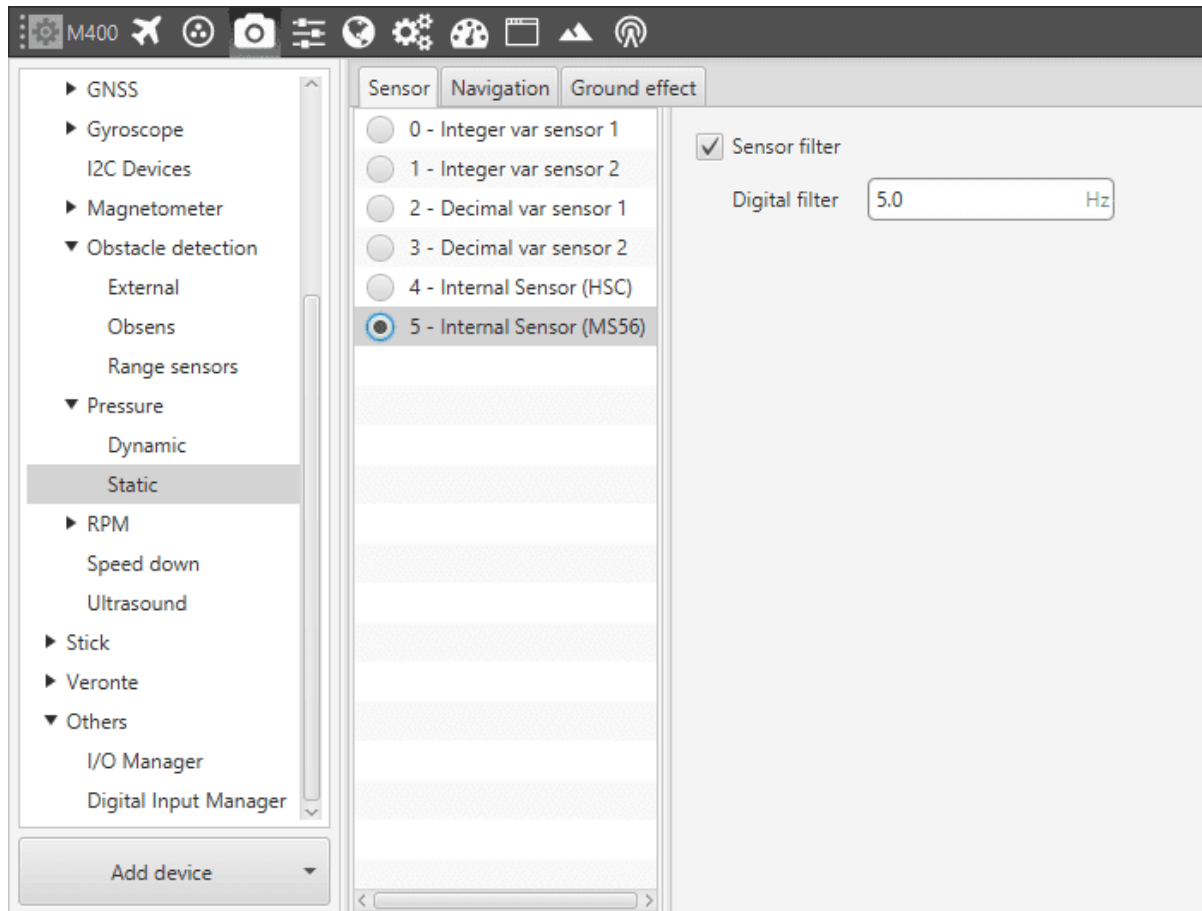


Decimal var Static Pressure Menu - Configuration Parameters

7.2.3.4.8.11 Internal

This menu displays the possible parameters that can be configured for the one of the internal Static Pressure sensors.

Veronte has embedded 3 digital static pressure sensors: the **DPS310** (only available in Veronte 4.5 hardware version), the **MS56** and the **HSC**. See more information on the pressure ports in [Connector Layout](#).



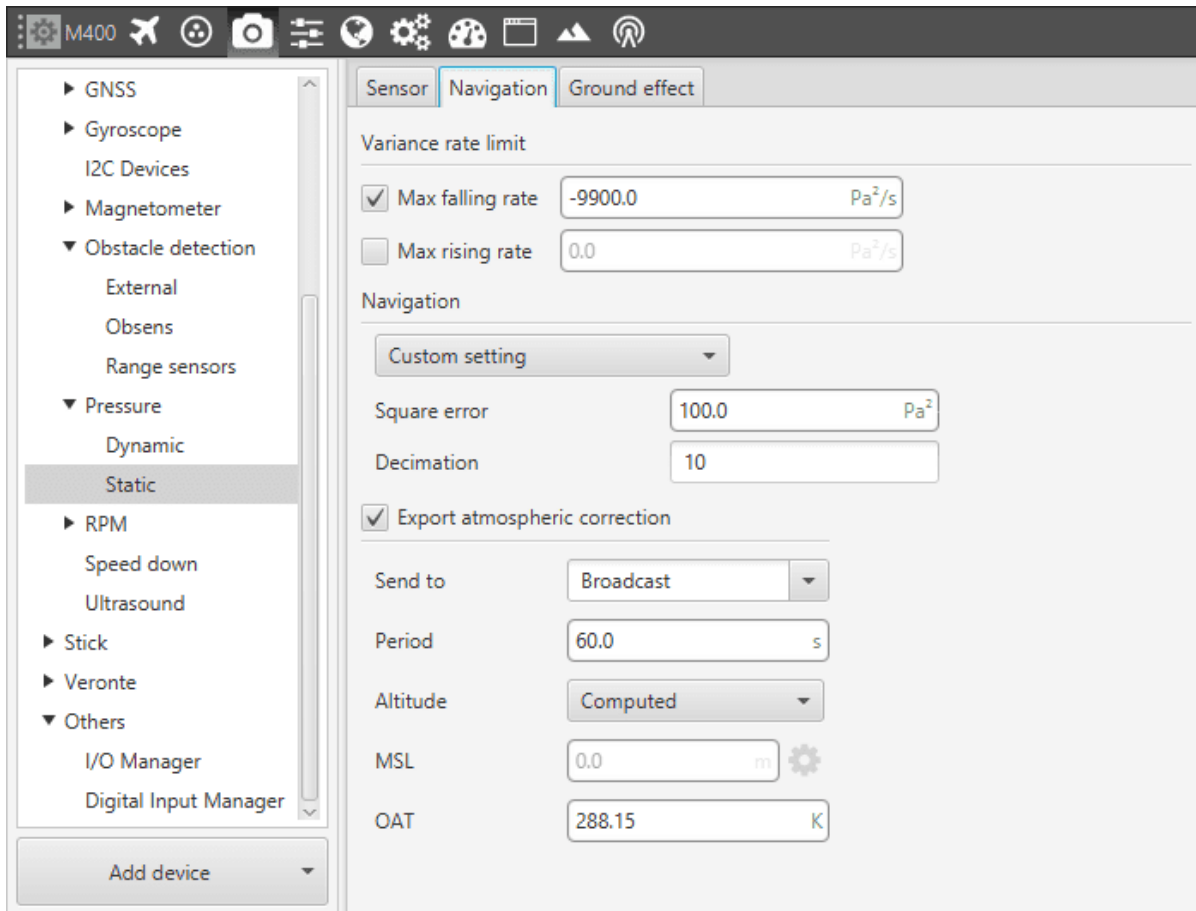
Internal Static Pressure Menu - Configuration Parameters

In this menu it is possible to set different options regarding filters.

- **Digital filter.** Enables a low pass filter which its cutoff frequency is configured manually, allowing the user to input any desired value in Hz. It is a **software filter**.

7.2.3.4.8.12 Navigation

This menu allows the user to configure the navigation parameters from a static pressure sensor input in Veronte.



Static Pressure Menu - Navigation Configuration Parameters

- **Variance rate limit.** Defines the maximum falling and/or rising rate from the system.
- **Navigation.**
 - **Enable.** Choose the static sensor pressure use on the system. Values from the sensor will be received, but they will not enter the Navigation Filter.
 - **Square Error.** Sensor error square, which defined the weight if the measurement into the Navigation filter.
 - **Decimation.** Defines the bunch of data from which 1 value will be stored. For example, if decimation is 10, every 10 measurements 1 will be taken into account. This procedure is used to reduce the number of samples.
- **Export atmospheric correction.** This feature allows, in a standard GND-AIR configuration, to send continuously information regarding configured platform static pressure to the desired platform. The parameters that need to be configured are:
 - **Send to.** Points where the correction will be sent. It can be another Veronte Unit (*Name-of-platform (Veronte SN)*), Broadcast or Pipe Fixed.
 - **Period.** Period of time spent between sending each correction.
 - **Altitude.** Can be computed by the system or input by the user manually if known.
 - **MSL.** Option enabled from the previous field. Can be input manually or by a system variable.
 - **OAT.** Outside Atmospheric Temperature, to be defined by the user.

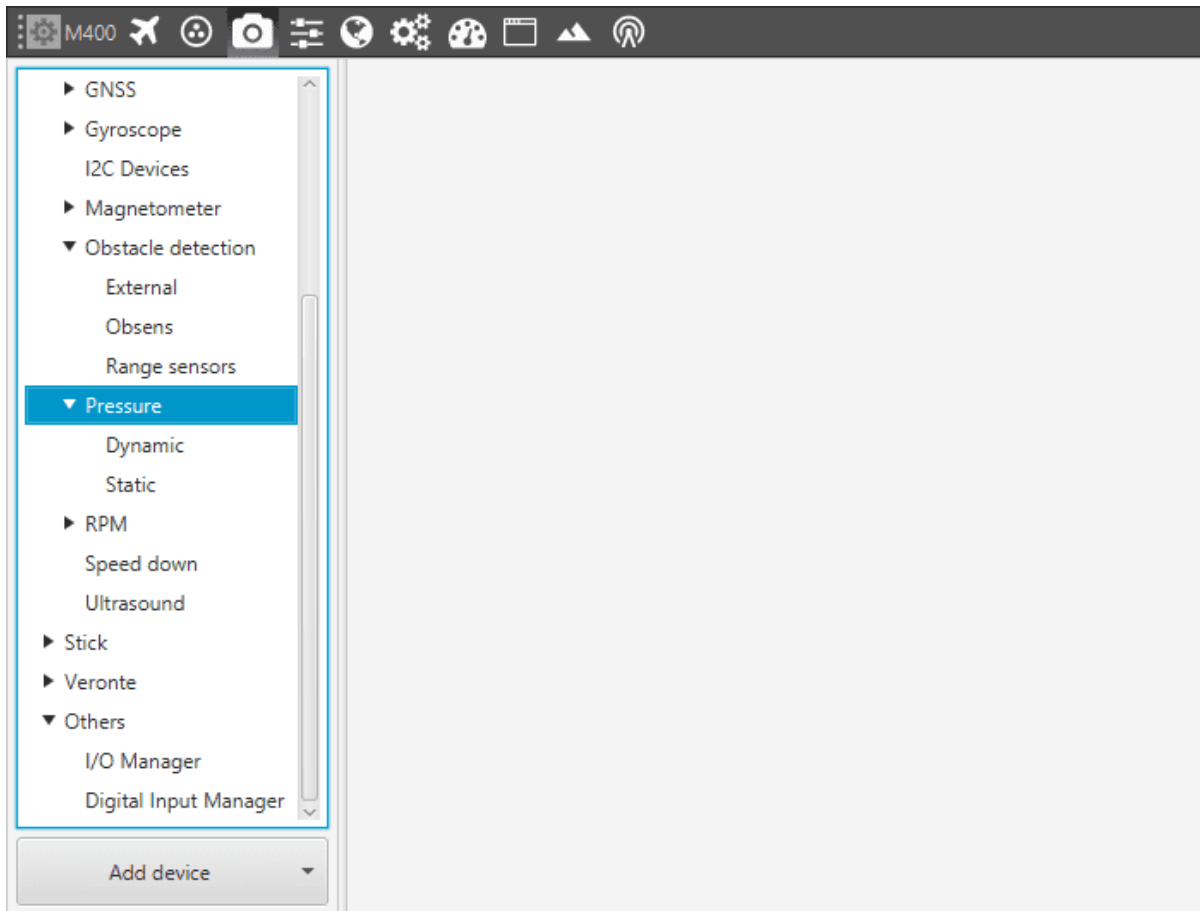
7.2.3.4.8.13 Ground Effect

Veronte is configured to apply a correction to the Ground Effect for landing purposes, which needs to be configured here.

- **Square error.** Values which automatically calculated from Navigation Square Error (see previous tab).
- **Altitude correction threshold.** The user has to define an Altitude Difference for the system to apply the Ground Effect. While landing, the aircraft will feel a decrease in the static pressure due to the Ground Effect, and this pressure difference (transformed into meters) is the Threshold that can be configured here. If set to 0, whenever Ground Effect is enabled, it will make its effect.

Note: Ground effect is configured as an Action in the Automations Tab.

In this menu the user can modify all parameters which are relevant to Pressure sensors.



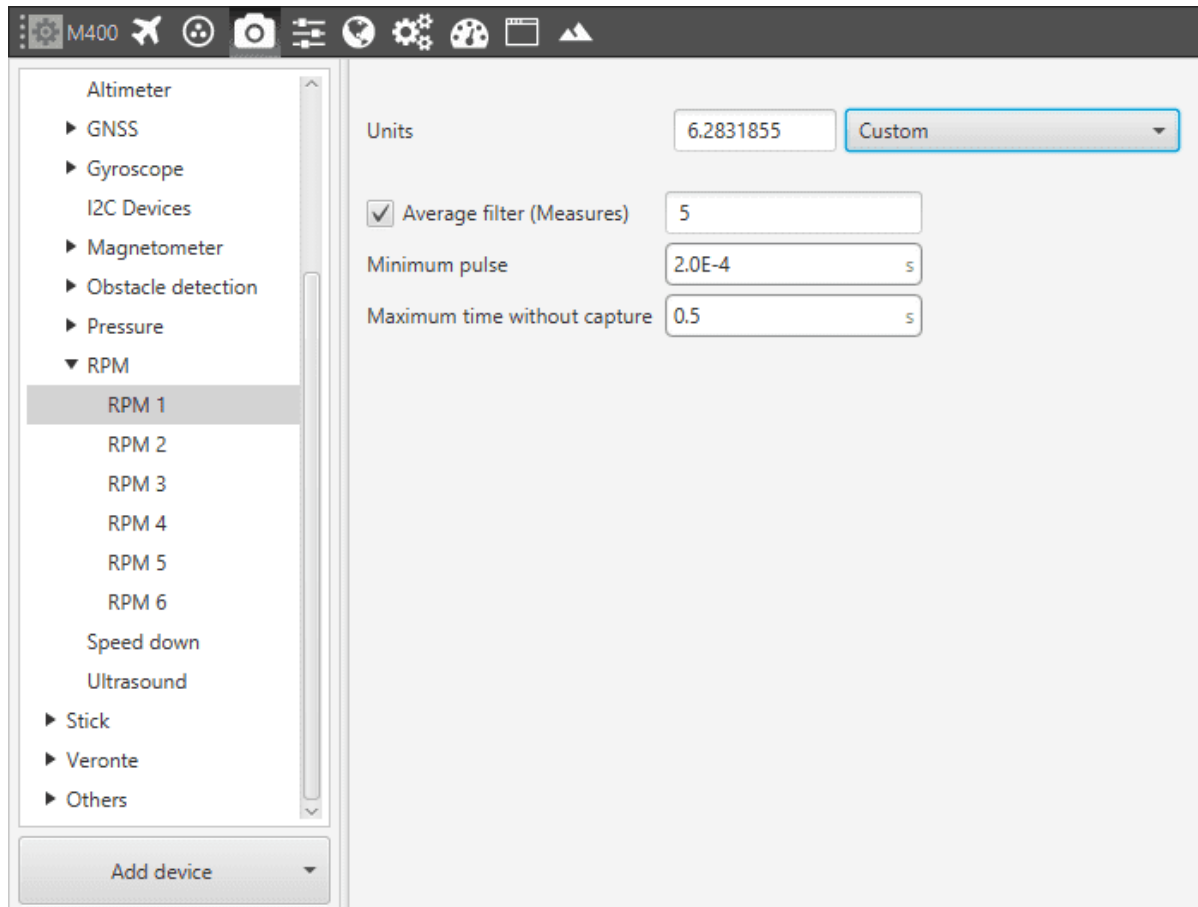
Pressure - Options available

Veronte has three pressure input lines, two for static pressure to determine the absolute pressure and one for pitot in order to determine the dynamic pressure.

7.2.3.4.9 RPM

Users can connect in *Devices - Other - Digital Input Manager* up to 6 RPM signal.

The following menu allows the user to configure these sensors.

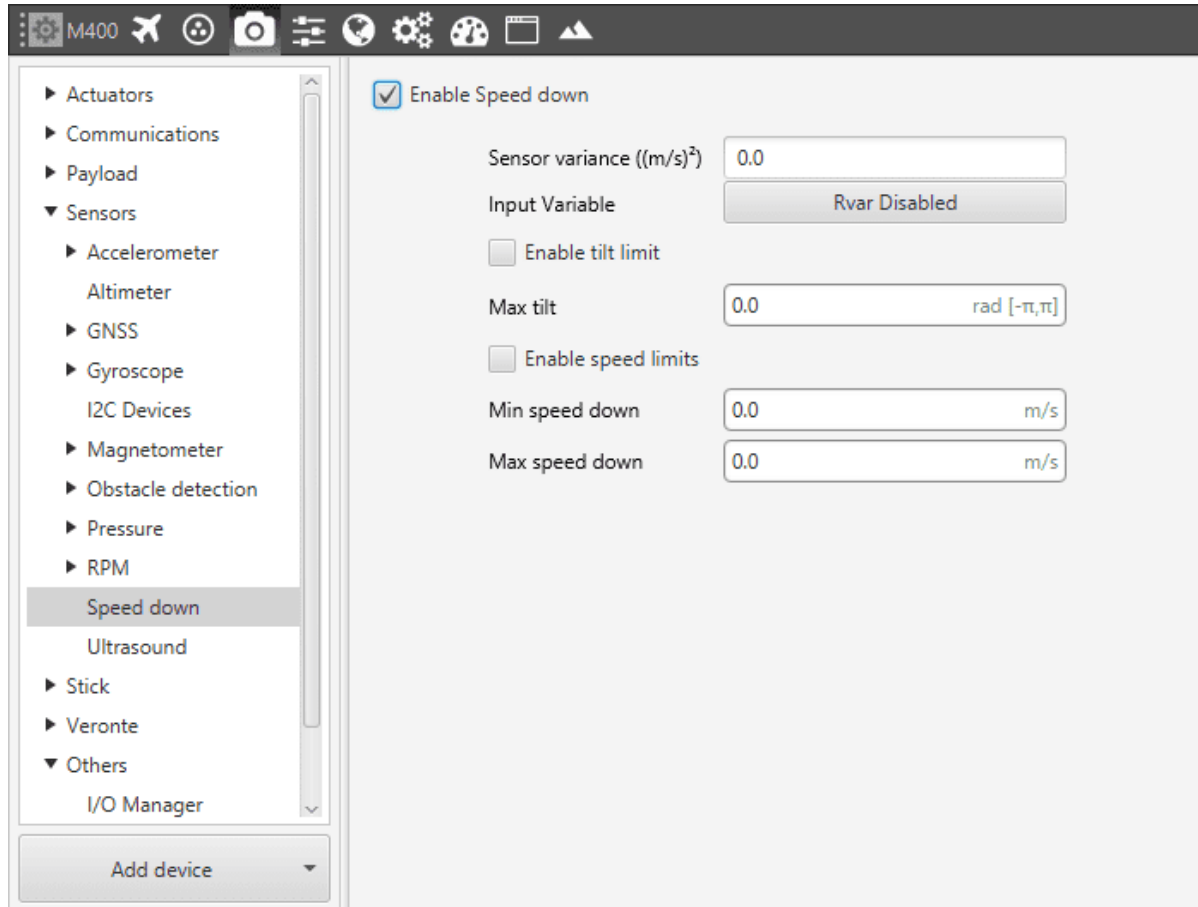


RPM - Menu Parameters

- **Units.** Sensor conversion factor. It can be Custom, Radians per pulse, Pulse per cycle.
- **Average filter (Measures).** It is a filter to avoid voltage spikes.
- **Minimum pulse.** Minimum time to detect a lap.
- **Maximum time without capture.** The maximum period of time allowed without capturing.

7.2.3.4.10 Speed Down

The speed down sensor measures the velocity in the Z axis of the platform. Works in the same way as the altimeter, but in this case, instead of a position reading, the magnitude measured is velocity.



Speed Down - Menu Parameters

- **Sensor variance.** Variance of the error of the sensor in meters per second squared.
- **Input variable.** It indicates the variable that contains the information from the sensor.
- **Tilt limits.** The sensor measures the variable in a direction perpendicular to the longitudinal axis of the platform, so when it is tilted the reading will not be reliable. This option allows the definition of a tilt limit, so that if the limit is reached, the sensor reading will be discarded.
- **Speed limits.** Defines the limits of the speed measured by the sensor.

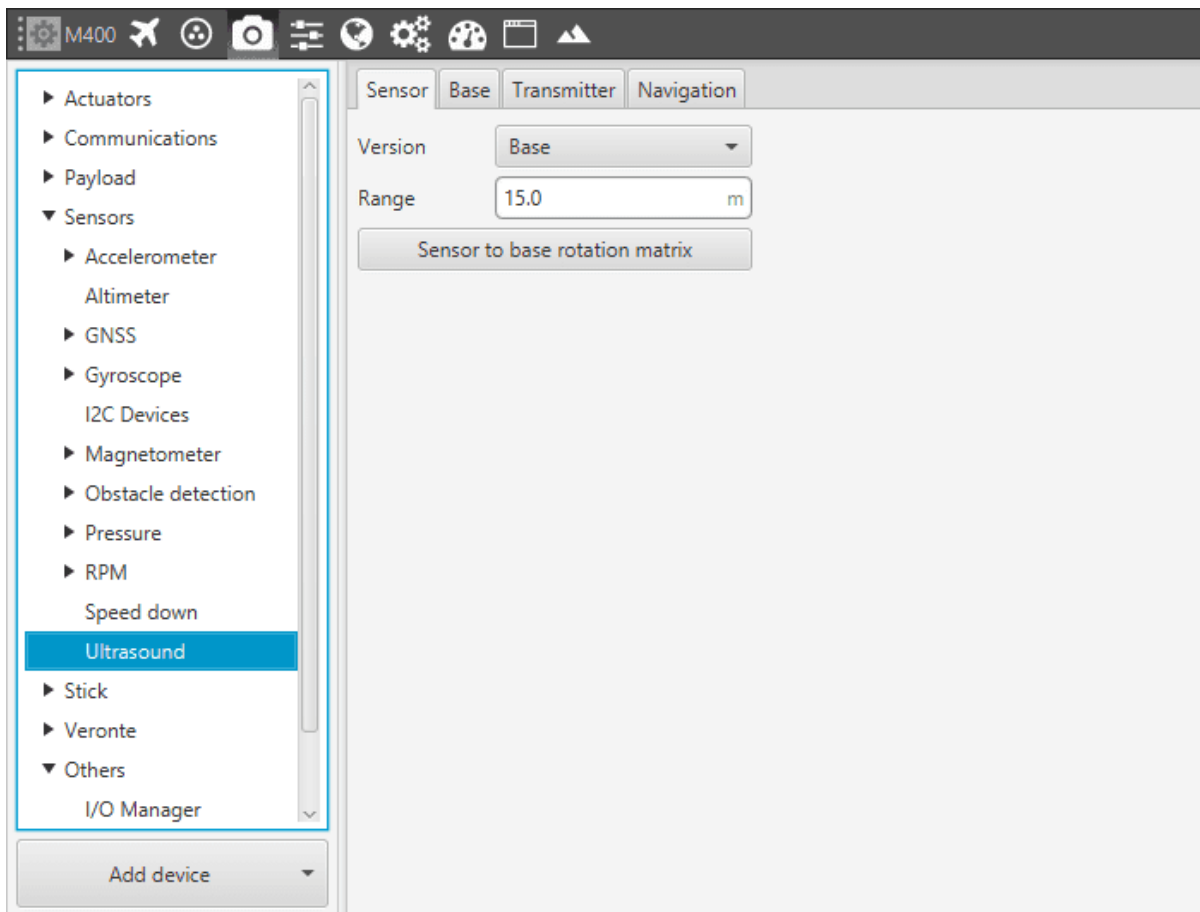
7.2.3.4.11 Ultrasounds

An ultrasound sensors computes Veronte position by measuring the time the signal sent out takes to return. The following panel allows the user to configure this sensor. This panel is used to configure an **Interneet** system with Veronte Autopilot.

7.2.3.4.11.1 Sensor

In this menu the user is allowed to choose which Internest version is to be used, its range and the rotation matrix:

- **Version.** To be chosen between *Internest Base* and *Internest Explorer*.
- **Range.** Defines de distances at which Internest values will start to be valid.
- **Sensor to base rotation matrix.** Matrix to rotate the system and make it coincident with the Veronte Autopilot.

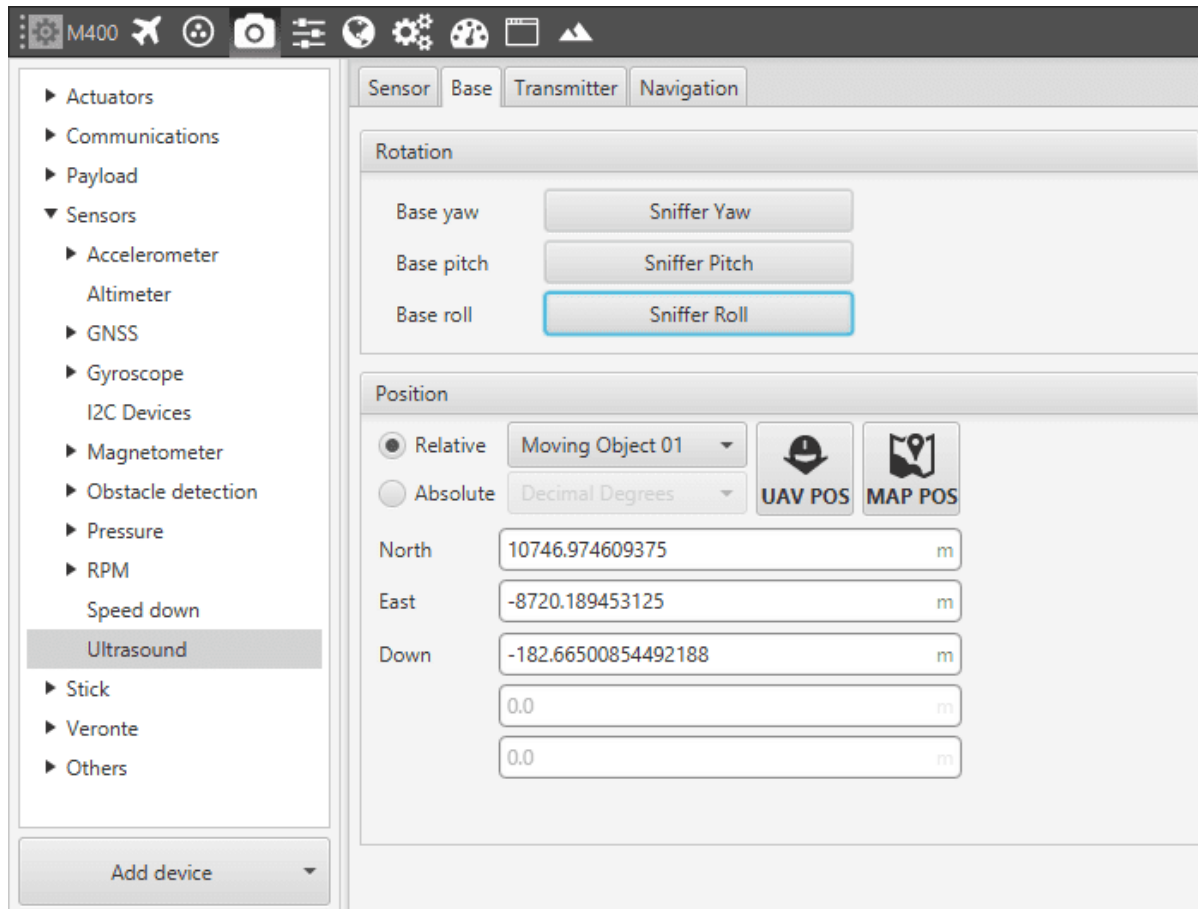


Ultrasound - Sensor Menu

7.2.3.4.11.2 Base

In the base menu the information *Sniffed* from the Base system is placed.

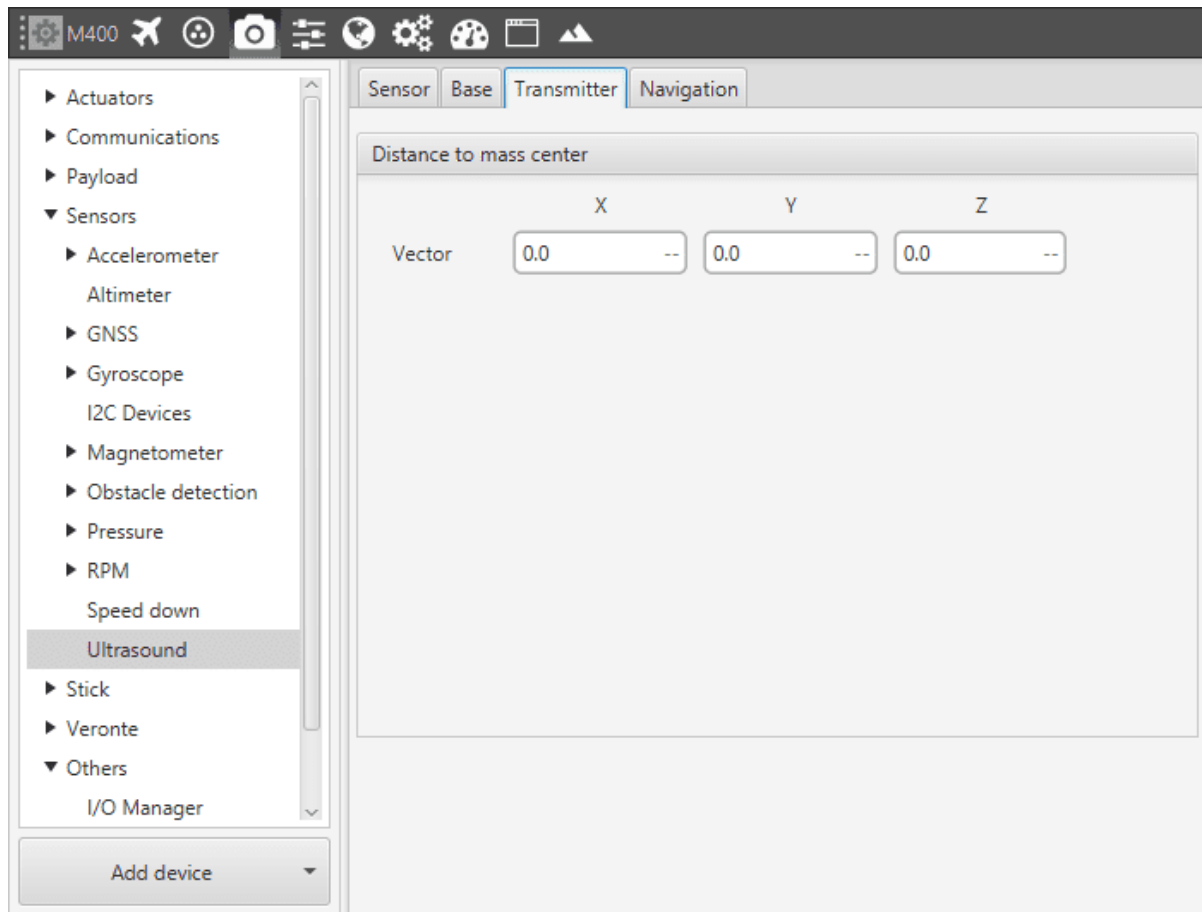
- **Rotation.** Input the 3 navigation angles from the Base platform.
- **Position.** Define it as Relative to Moving Object, which is the Base position sniffed.



Ultrasound - Base Menu

7.2.3.4.11.3 Transmitter

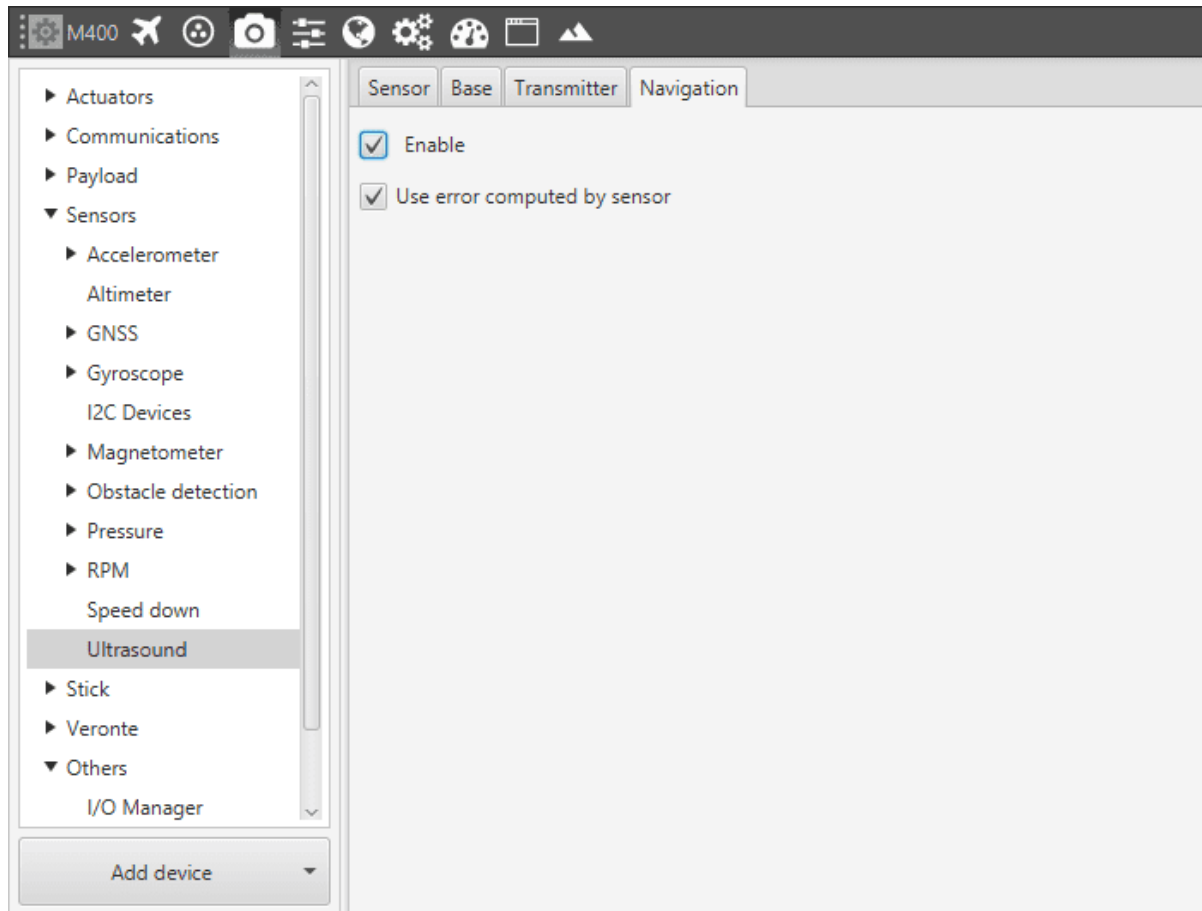
Defines the distance between the Internet system and the mass center from the platform.



Ultrasound - Transmitter Menu

7.2.3.4.11.4 Navigation

It has two checks configurable, one for enabling/disabling the Internet effect in navigation and a second one to use the error computed by the sensors if it affects the navigation.



Ultrasound - Navigation Menu

Sensors section permits to configure any sensor connected or internal from Veronte.

| | |
|---------------------------|--|
| Accelerometer | Configure the accelerometer parameters |
| Altimeter | Enable the Altimeter into navigation if a sensor is connected to Veronte |
| GNSS | Configure GNSS receivers, RTK, Compass, External source and NTRIP |
| Gyroscope | Configure the gyroscope parameters |
| I2C devices | Configure Lidar sensors connected through I2C |
| Magnetometer | Configure the internal magnetometer or an external one |
| Obstacle Detection | Enables the obstacle detection and configured it |
| Pressure | Menu including both dynamic and static pressure configurations |
| RPM | Configure RPM sensors parameters |
| Speed Down | Enable the navigation using a Speed down sensor |
| Ultrasound | Menu to configure Internest with Veronte |

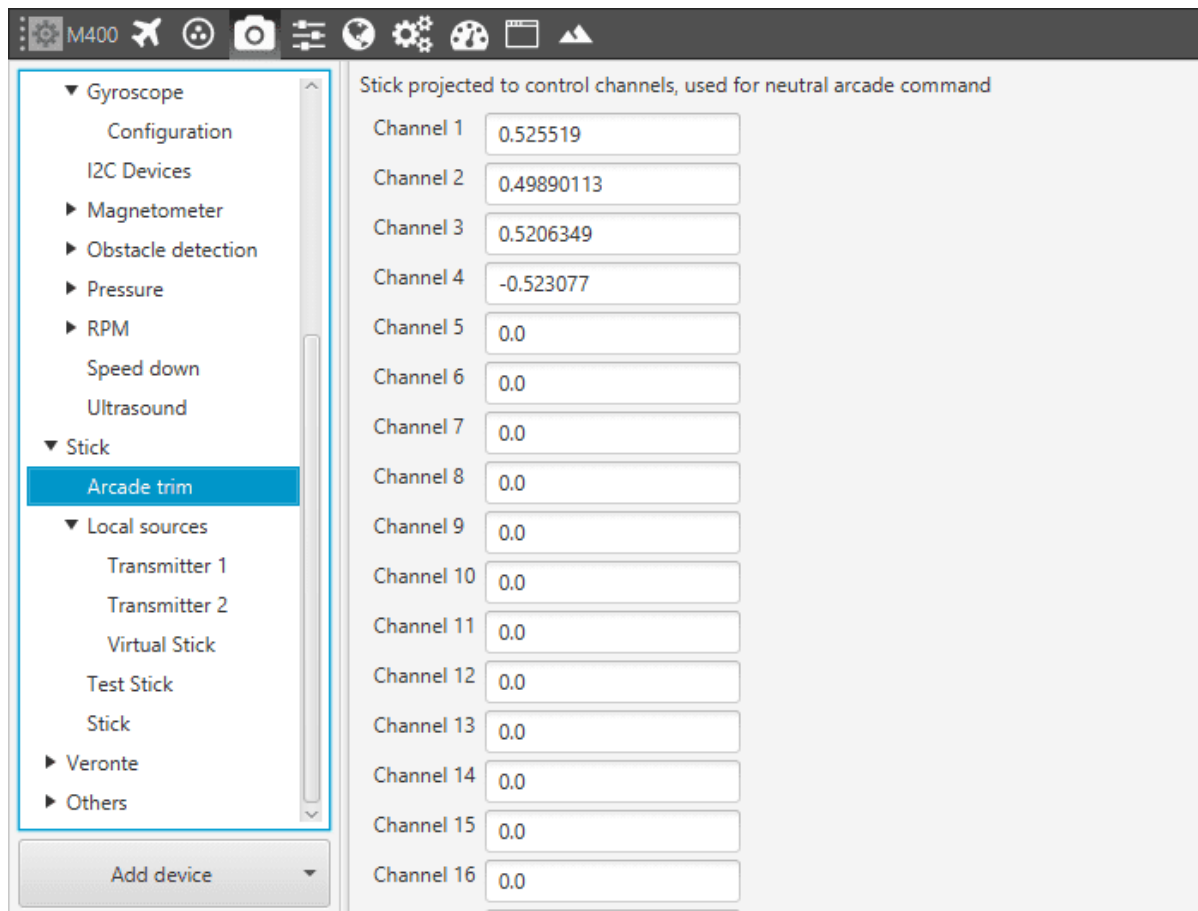
7.2.3.5 Stick Configuration

7.2.3.5.1 Arcade trim

The arcade trim is used to set the **zero-stick position** for the *Arcade Mode*.

In such mode, when obtaining the *Control output* U from *Stick input* R the final value is not the one that enters the navigation algorithm. The variable that is otherwise employed is D , called “*Stick input d*”, which is computed as $D = U - U_0$, being U_0 the arcade trim. In this way, when the sticks are trimmed at a certain position, the movement from that point will be the value of R that - after transformations - will generate the U .

The values of the trim vector U_0 can be introduced in **Devices - Actuators - Stick - Arcade trim** as shown in the figure below.



Stick Arcade Trim - Configuration Parameters

Another way to perform the arcade trim is to create the automation as described in *Automations - Actions - Arcade trim*. Once that is done, it is sufficient to move the stick main levers to their center position and to click on the *Trim Arcade button*. Then wait for 2-3 seconds and return the levers to their default positions.

Warning: The Arcade mode has to be trimmed before flight. If not trimmed, the zero level will be different from the desired one.

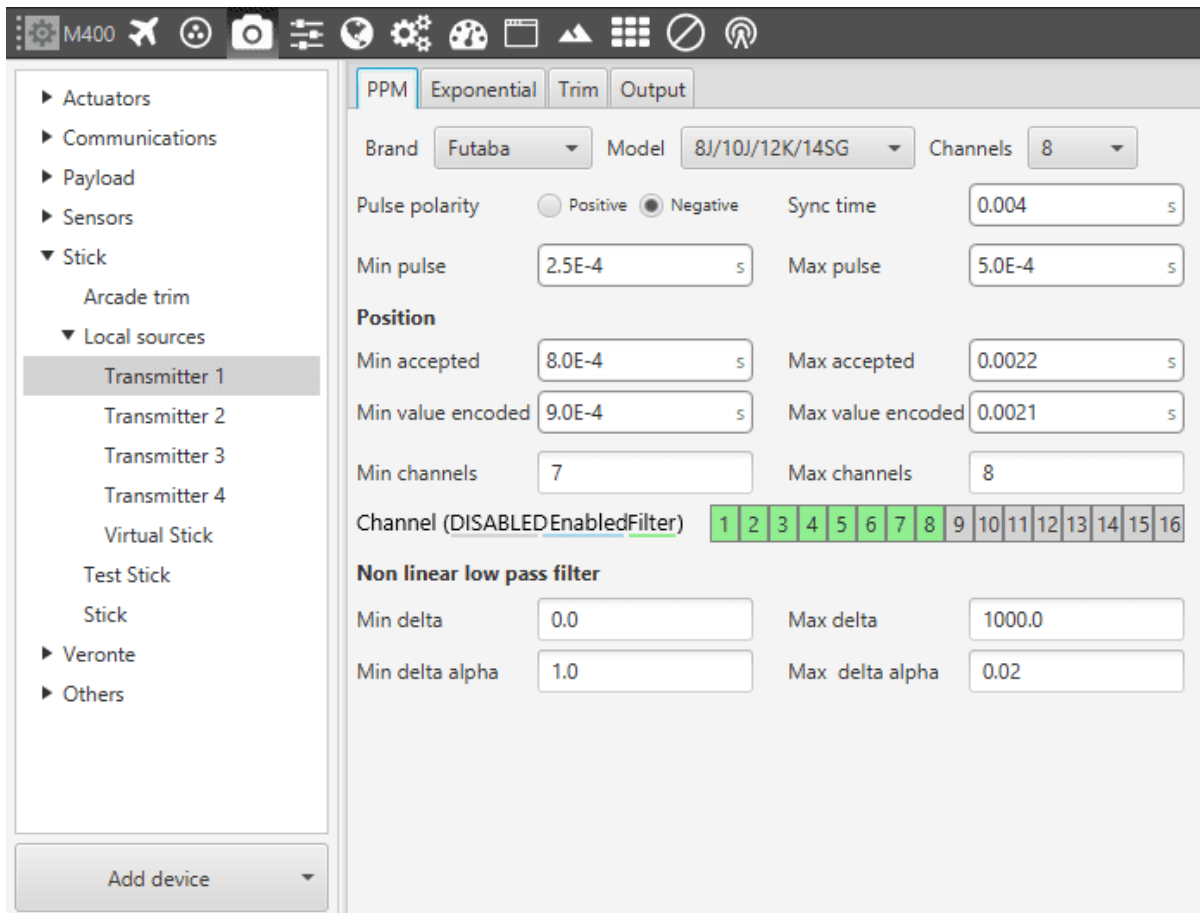
7.2.3.5.2 Local Sources

7.2.3.5.2.1 Transmitter

The wired connected transmitters are configured through the following menus.

7.2.3.5.2.2 PPM

This tab provides the options to configure a Pulse Position Modulation (PPM) radio controller to control the platform fitted with the autopilot.

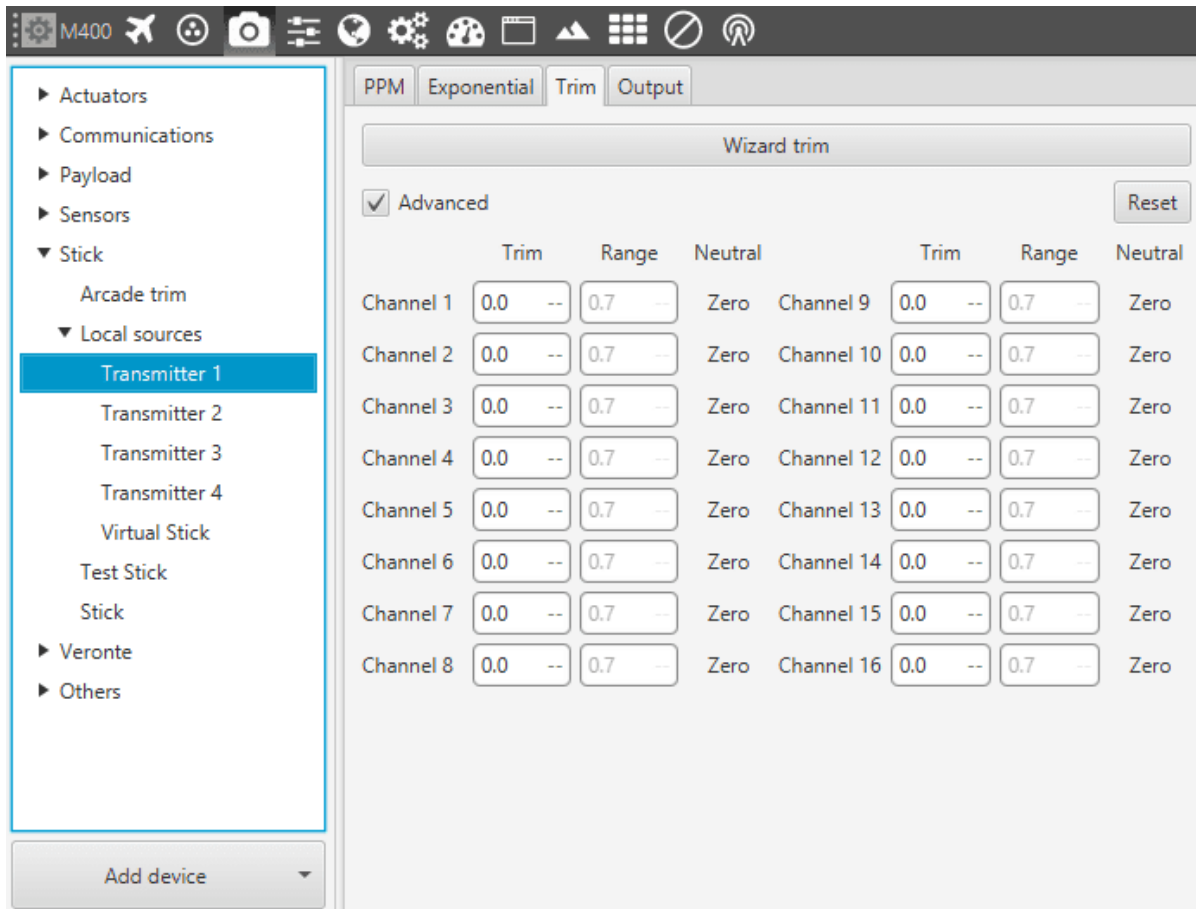


Stick Transmitter - PPM Configuration Parameters

- **Brand, Model & Channels.** Veronte Pipe has been configured to provide the user with the expected parameters to configure different transmitters models:
 - Futaba.
 - * Model 8J/10J/12K/14SG with 8 channels.
 - * Model 12K/14SG with 12 channels.
 - * Model T18SZ with 8 channels.
 - Jeti.

- * DC 16 / DC 24 with 16 channels.
- FrSky.
 - * Taranis 9XD with 8 channels.
 - * X12S with 8 channels.
- **Polarity.** Indicates the pulse polarity. The image below shows a positive signal.
- **Sync time.** Minimum time on the PPM output till the next frame. It tells the receiver to reset its channel counter.
- **Minimum/Maximum pulse.** Pulse length, it depends on the system and it is a constant value (usually 0.2-0.5 ms).
- **Position – Minimum/Maximum accepted.** Pulse length accepted for each channel. Standard for R/C servos uses a pulse of 1 ms for the maximum position at one end, 1.5 ms for the midpoint and 2 ms for the maximum position at the opposite end.
- **Minimum/Maximum encoded.** If there is noise and the signal is varying around the minimum/maximum values accepted, Veronte will encode those values to the ones set here. For instance, a pulse length between 0.8-0.9 ms will be considered as one of 0.9 ms.
- **Channels.** Sets the number of channels accepted (minimum and maximum). Besides, it is possible to Disable/Enable/Filter each channel individually.
- **Non linear low pass filter.** Default parameters are recommended.

When some data is out of the established ranges, Veronte will discard that frame to avoid a possible malfunction. The next image clarifies the parameters introduced previously.

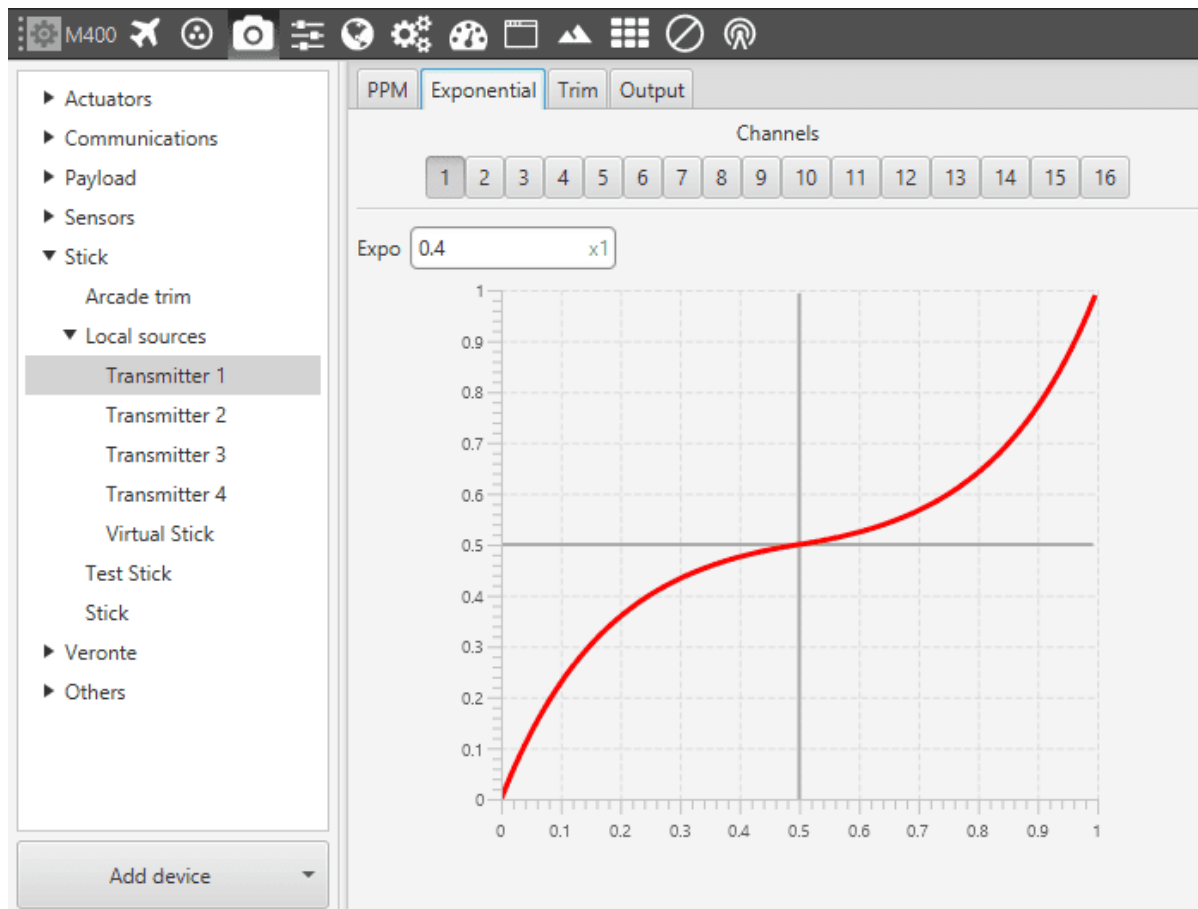


Transmitter - PPM Signal

7.2.3.5.2.3 Exponential

The second tab in allows the user to define an exponential stick response for every channel.

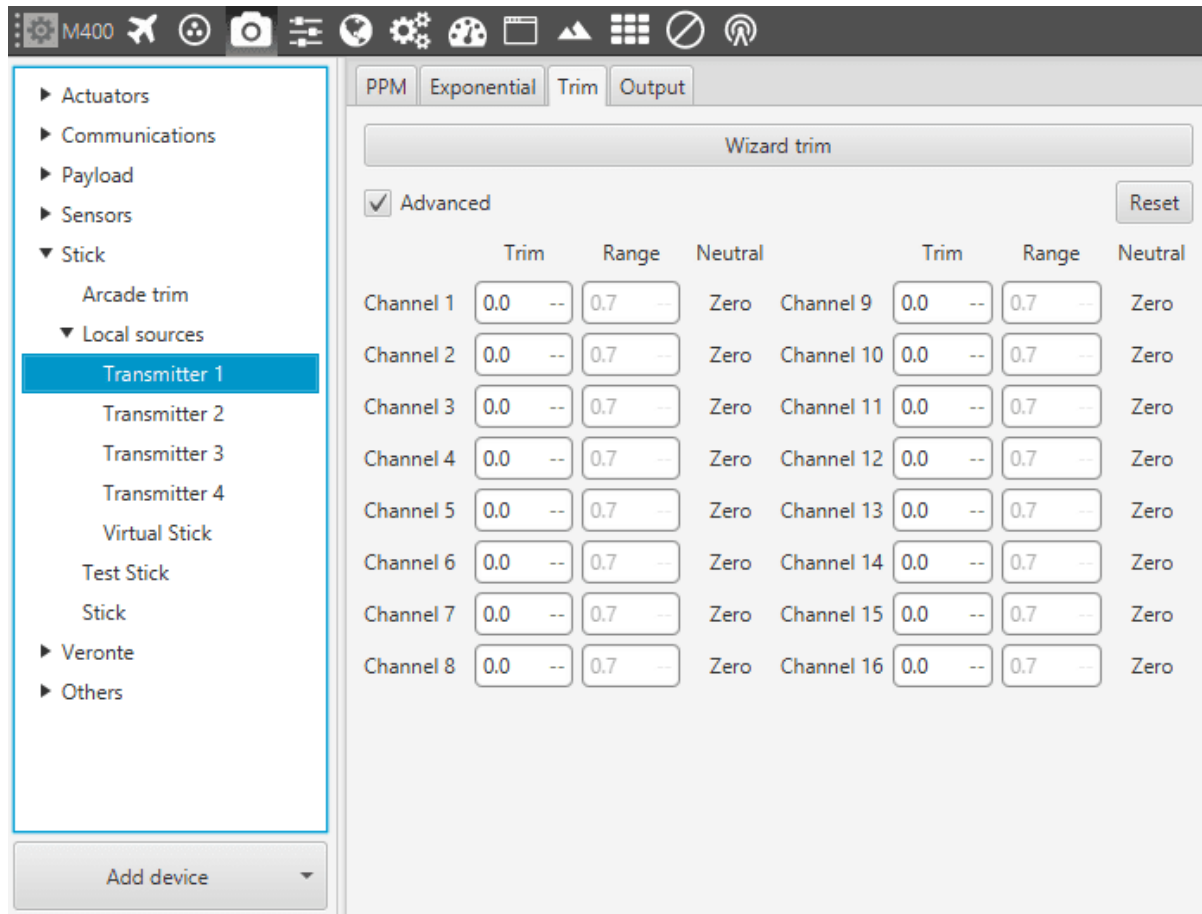
Allowed inputs go from 0 to 1 and there is a graph showing the generated response curve, as can be seen in the following figure.



Transmitter - Exponential

7.2.3.5.2.4 Trim

The third tab available is the Trim option. On the upper part, there is a *Wizard trim button*. The latter will guide the user through some pop-up windows.

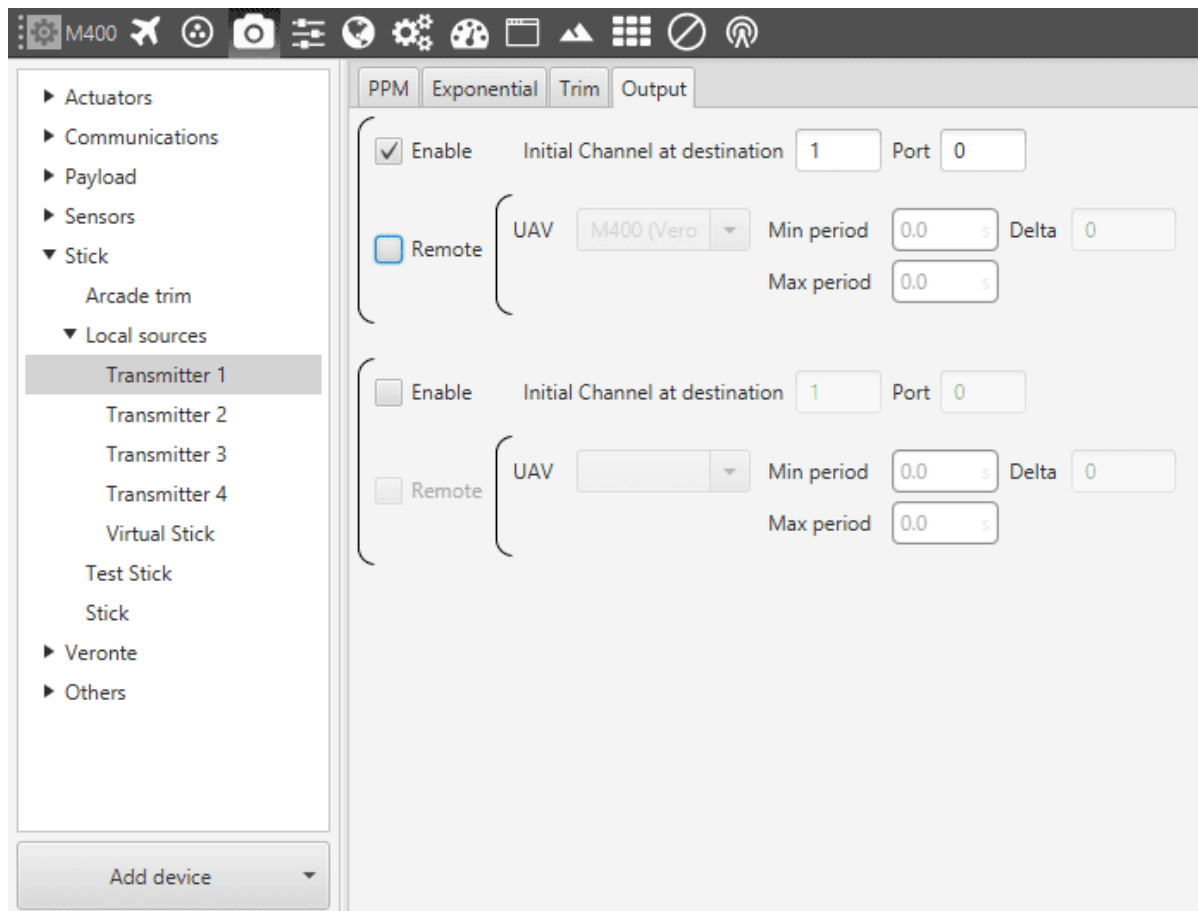


Transmitter - Trim

If that option is not chosen, the user can toggle the Advanced option and set the expected trim values manually. The user should have a deep knowledge on its transmitter if this option is selected. Finally, on the right hand side, the *Reset* button puts every parameter back to 0.

7.2.3.5.2.5 Output

Lastly there is the Output tab. Once the stick has been configured, the commands that arrive at the ground autopilot have to be sent to the air one. Here the user sets the receiving port and process the incoming commands.



Transmitter - Output

Once enable is flagged, the user indicates to which channel of the air autopilot (**AIR**) will be sent the first channel received in the ground segment (**GND**). The channels arrive at the platform in order and without spaces between them i.e, if at the **GND** channels 1,2,3,4 and 6 are enabled, the **AIR** will receive channels 1,2,3,4 and 5. Therefore channel 6 of the stick will be channel 5 in the **AIR** configuration.

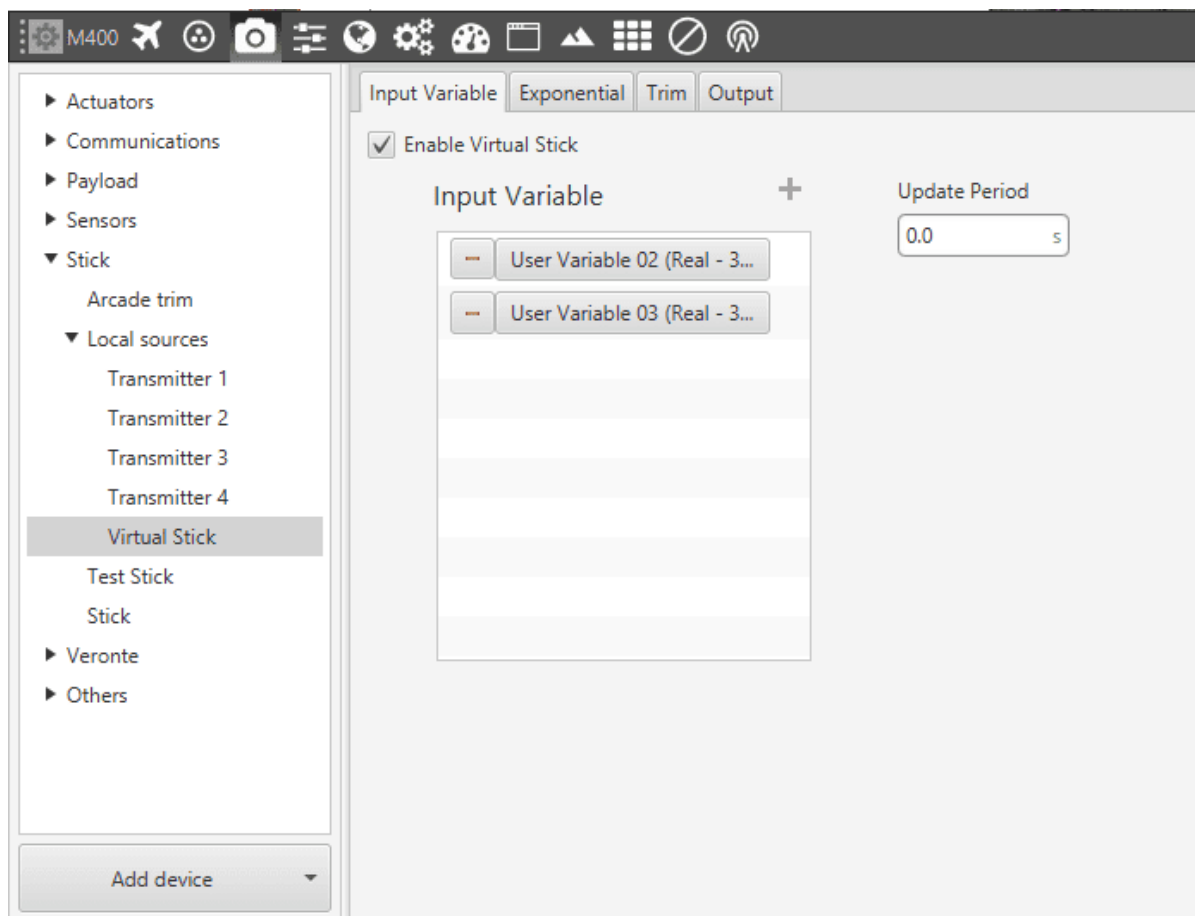
The option remote allows the delivery of the commands to the platform by indicating the address of the UAV. There is also the option *Broadcast*, where the commands are sent to all the air autopilots linked to the **GND**.

7.2.3.5.2.6 Virtual Stick

If the information of a stick is received through a different channel than PPM, the user can create a virtual stick to process that information and input it to the system.

In order to do so, enable the Virtual Stick, place the variables containing the stick information and configure the update frequency required.

The tabs Exponential, Trim and Output are the same as the Transmitter ones, so refer to *Transmitter* for more information.



Virtual Stick - Configuration Parameters

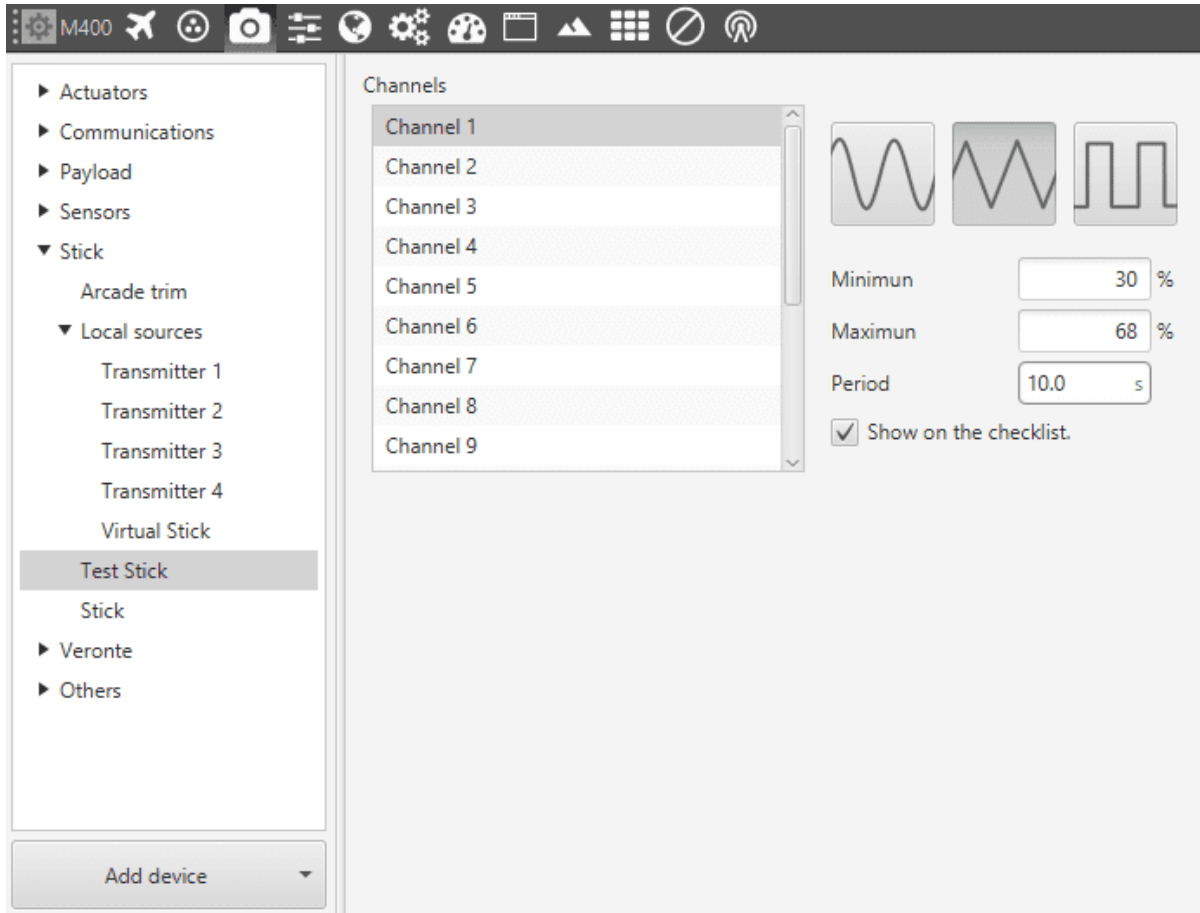
This part of the configuration toolbar allows the user to set **up to four transmitters** and **one virtual stick**. The autopilot's capabilities allows it to receive information from four different transmitters at the same time plus transforming some values into a virtual stick.

The content presented in the next two sections covers:

- Setting of the transmitter's parameters.
- Definition of exponential response-curves for the desired channels.
- Trimming of the channels' neutral position.
- Setting of the data receiving port on the autopilot.
- Definition of a virtual stick.

7.2.3.5.3 Test Stick

This menu is used to generate stick inputs that are introduced in the system. This is a way to check how the system behaves when a stick commands enters in the autopilot. For each channel, the user can set a continuous movement formed for example by a sine wave, spike-like function or a rectangular function. There are a set of parameters to be modified for the signals.



Test Stick - Configuration Parameters

- **Minimum & Maximum:** limit values that will be sent to the system for the stick test.
- **Period:** function period.
- **Show on the checklist:** configured parameters can be shown on the checklist in order to test the system prior to change flight phase - more information on [Checklists](#). To activate the automatic movement, activate the virtual stick configured - see [Workspace - Widgets - Stick](#).

Warning: This test should be performed carefully if control surfaces/devices are engaged because the stick input should be carefully defined.

7.2.3.5.4 Stick

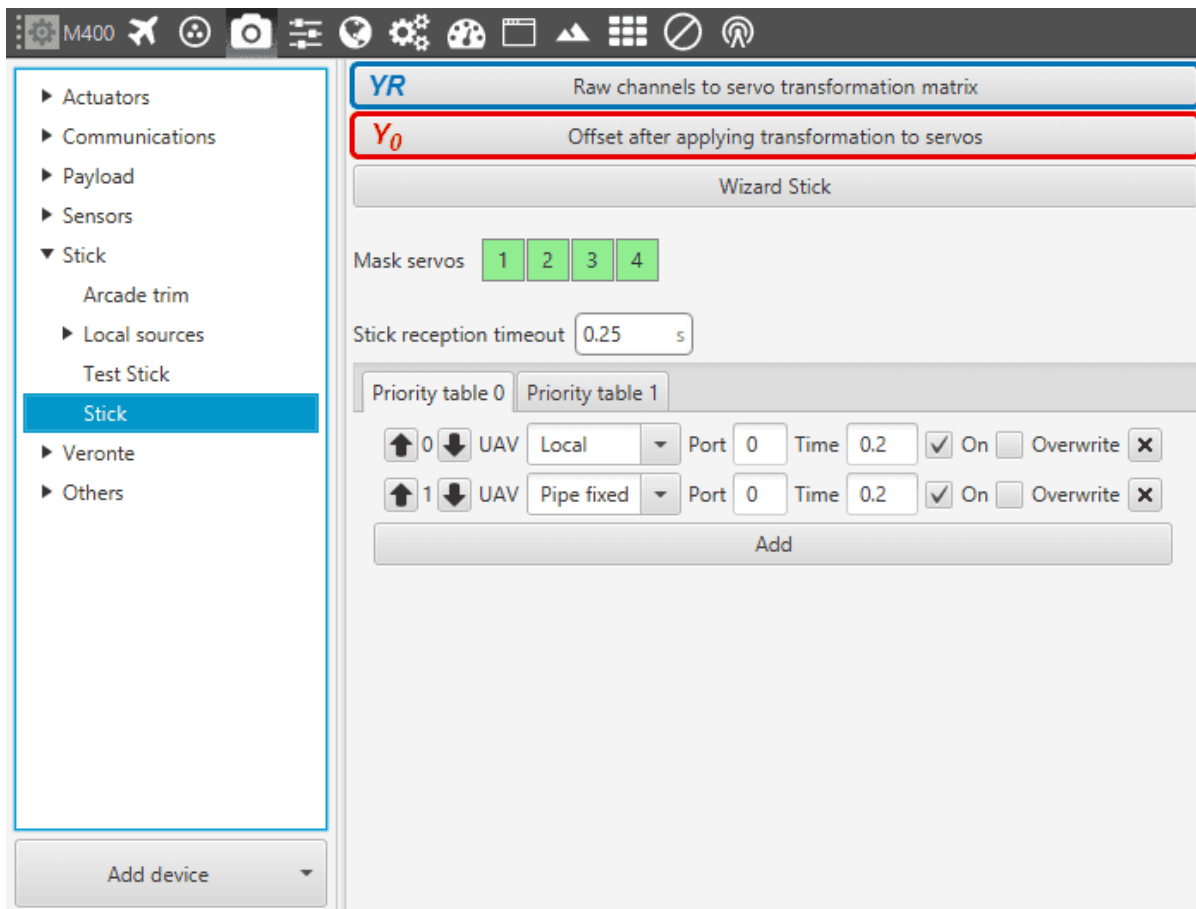
The following content will help the user to configure the stick parameters for manual and arcade modes – more information about these modes in [Modes](#).

The movement that the pilot makes on the stick produces variations on a vector called R of length n , where n goes from 1 to the total number of employed transmitter channels. The values reached by the components of R are limited between 0 and 1. These stick movements need to be processed to produce the input signals that will go into the control algorithm, in the case of arcade mode; or directly into the servos for manual mode.

The process begins by mapping each one of the sticks inputs to PWM signals into a vector called Y of length m , where m goes from 1 to the total number of actuators. The full definition of Y is $Y = YR \cdot R + Y_0$, where :

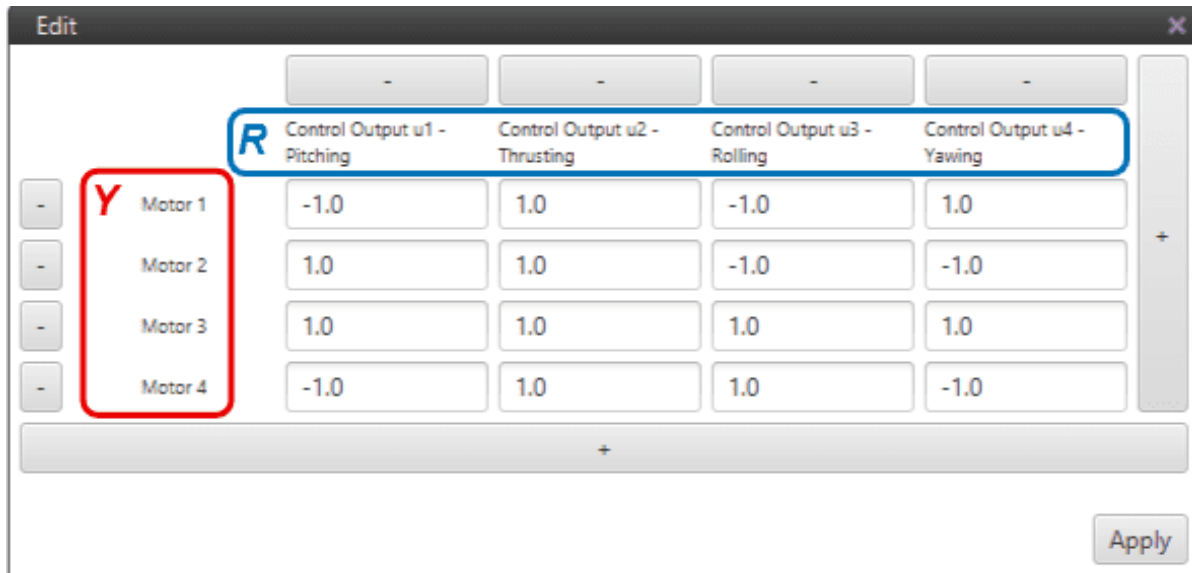
- YR is a matrix that transforms raw stick inputs R to PWM signals Y .
- Y_0 is an offset vector, which corrects the Y vector.

The above parameters are defined at beginning of the **Stick** menu (as seen in the Figure below). There is a *Wizard Stick* button that sets YR and Y_0 , which has a dedicated section below.

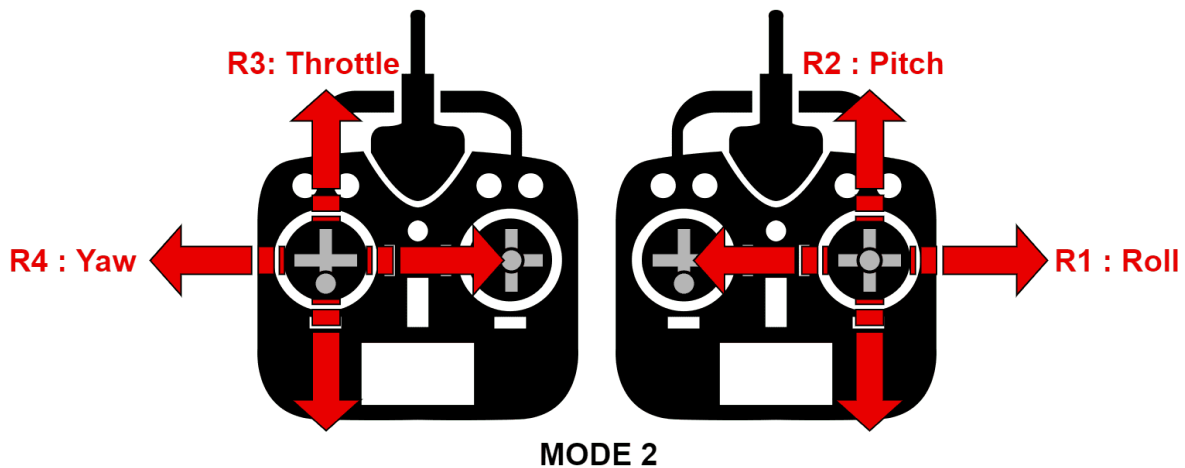


Stick - Configuration Parameters

Let's consider a quadcopter. With such platform applying thrust produces an increase in the rotation speed of the four engines, and only one stick channel (throttle) is required to achieve that. So here the mapping will be that an increase in one component of R has to produce an increase in four components of Y . Doing the same with the rest of stick channels (pitch, yaw, roll) will generate matrix YR :

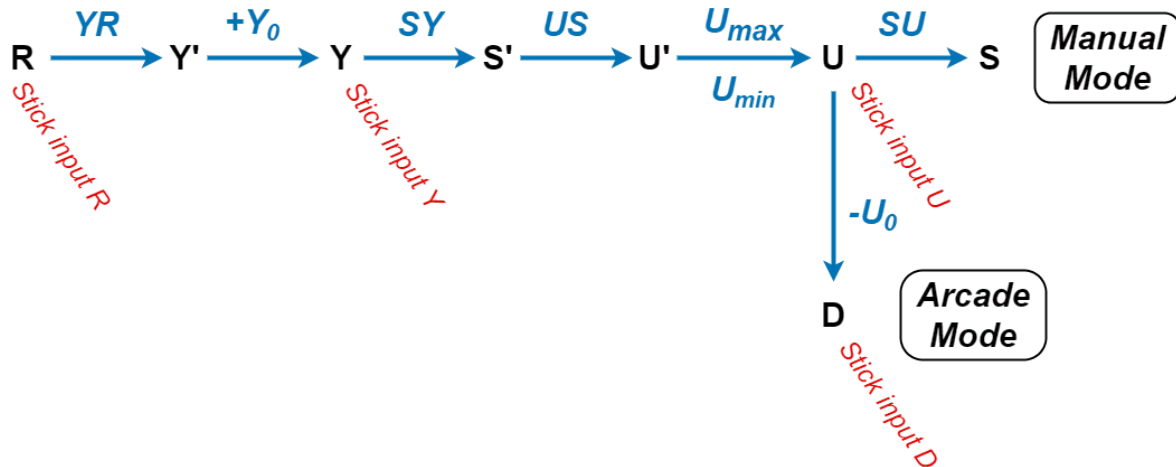
Stick - YR Matrix for a Quadcopter

Keep in mind that each stick channel (throttle, roll, yaw, pitch, etc) depends on the *mode* of the transmitter, i.e. the sticks' physical layout. There are 4 different modes and there is no prevailing type, it really depends on the pilot. In the following Figure a **Mode 2** channel layout is depicted:



Stick - Channels Layout

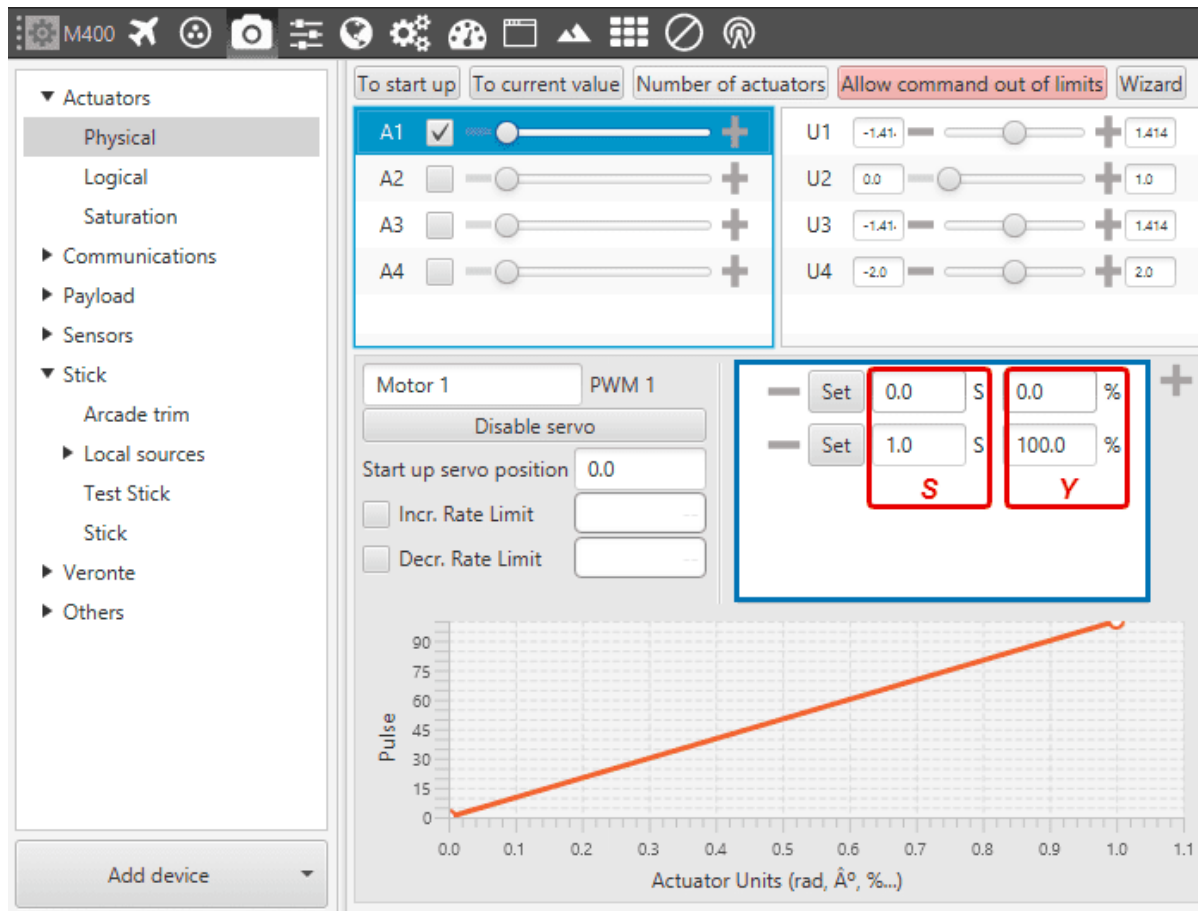
The offset Y_0 is calculated by setting R and Y with a specific value. Take the neutral position of a quadcopter: the sticks are $R_{neut} = [0.5; 0.5; 0; 0.5]$ (the third value is 0 as it corresponds to the neutral position of channel 3, i.e. the throttle) and the actuators PWM signals $Y_{neut} = [0; 0; 0; 0]$. Then, the solution for the offset vector will be: $Y_0 = Y_{neut} - YR \cdot R_{neut}$. The neutral position information will be asked when using the *Wizard Stick button*.



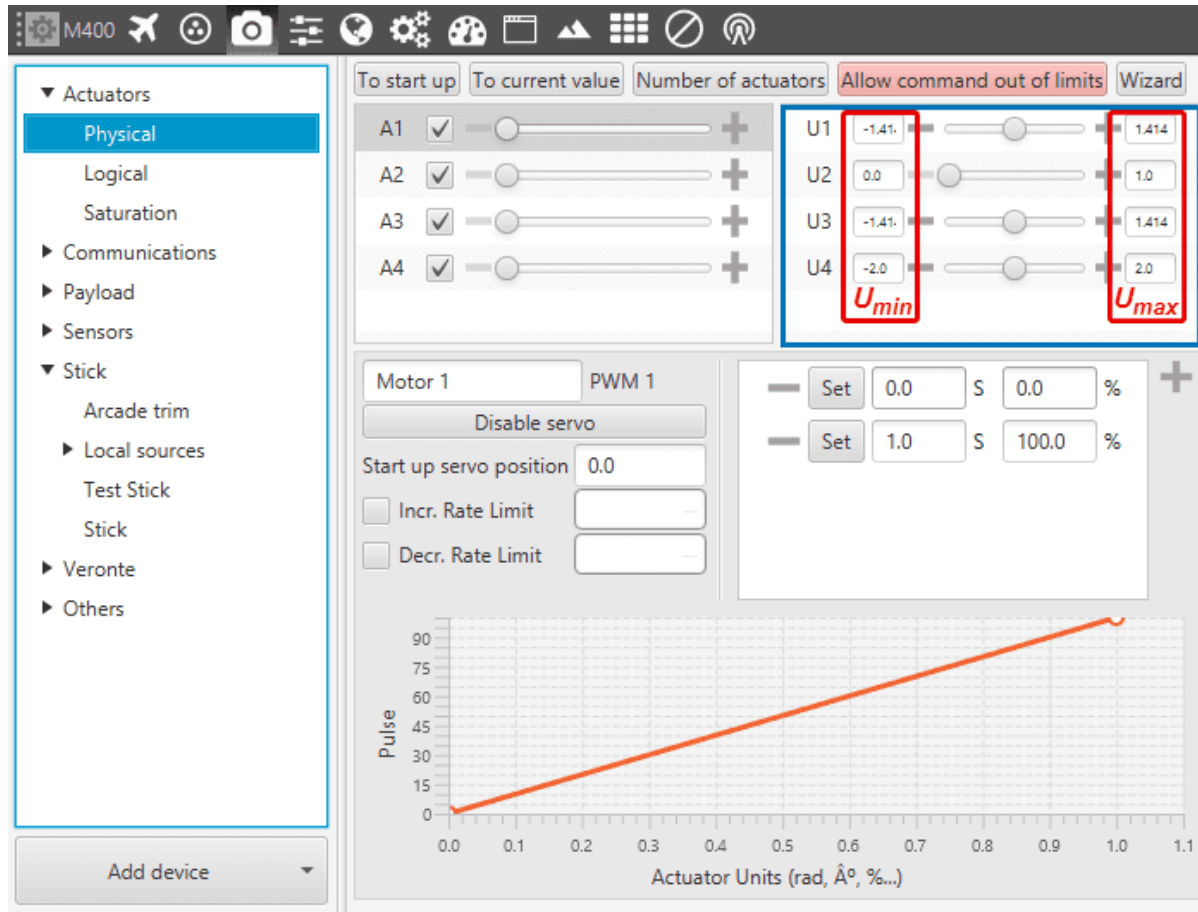
Stick - Transformation Diagram

The above diagram depicts the whole transformation process of the incoming raw input R vector. It also shows in red the naming of some important variables in Veronte Pipe. Step by step, the process is:

1. From R one gets Y' using YR , as defined above.
2. From Y' one gets Y adding Y_0 .
3. From Y , using the trim of the servos, one gets the actuator output S' . PWM to actuator output mapping SY is defined in [Devices - Actuators - Physical](#). The mapping consists in a piecewise-linear function relating each actuator output S to a certain amount of PWM signal S :

Linear Mapping SY of a quadcopter engine

4. From S' , using the logical transformation matrix US (defined in **Devices -> Actuators -> Logical**), one gets U' .
5. From U' , and delimiting its values if they are higher or lower than the limit values U_{max} and U_{min} , one gets U . The limit values of U are found in **Devices - Actuators - Physical**.

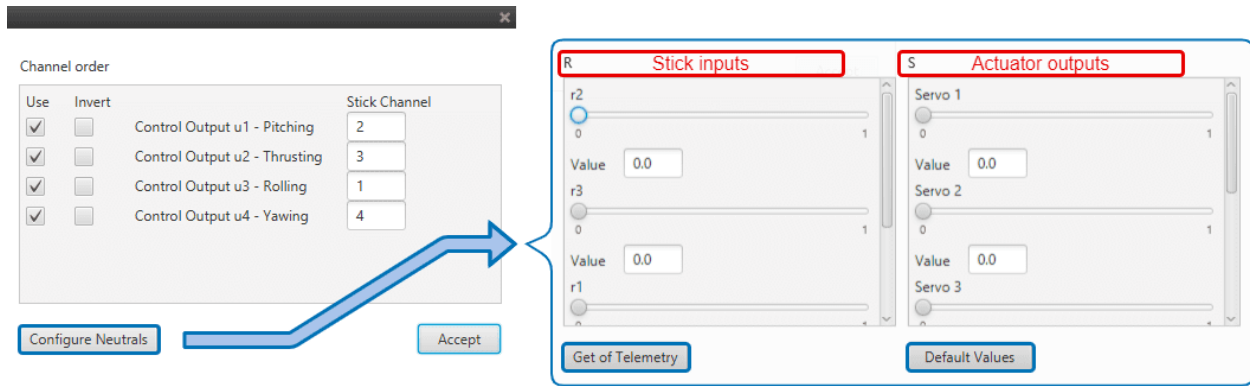
Limit values of U

6. From U , subtracting the arcade trim of the transmitter U_0 one gets vector D , which goes into the guidance control of the modes where arcade mode is enabled. More information on the arcade trim on [Devices - Stick - Arcatedtrim](#).
7. Lastly, if manual mode is selected, one last transformation is required. From U , applying SU , one gets the final S : actuator outputs.

7.2.3.5.4.1 Wizard Stick

To make use of this assistant tool, logical matrices US and SU must have been defined, otherwise it will not work.

When clicking on the button (see Figure below), the user is asked to associate each control output U of the platform with a stick channel R . Doing so, matrix YR is automatically calculated.



Stick - Wizard Stick Configuration Parameters

In addition, to obtain Y_0 the user needs to click on *Configure Neutrals* – otherwise it is considered as $Y_0 = 0$. There, the neutral values of the actuator outputs S_{neut} (i.e. servo positions, engine rpms, etc) and the neutral values of the stick R_{neut} can be introduced manually. If *Default Values* button is clicked, S_{neut} is taken from *Devices - Actuators - Physical*, where its start-up position is defined.

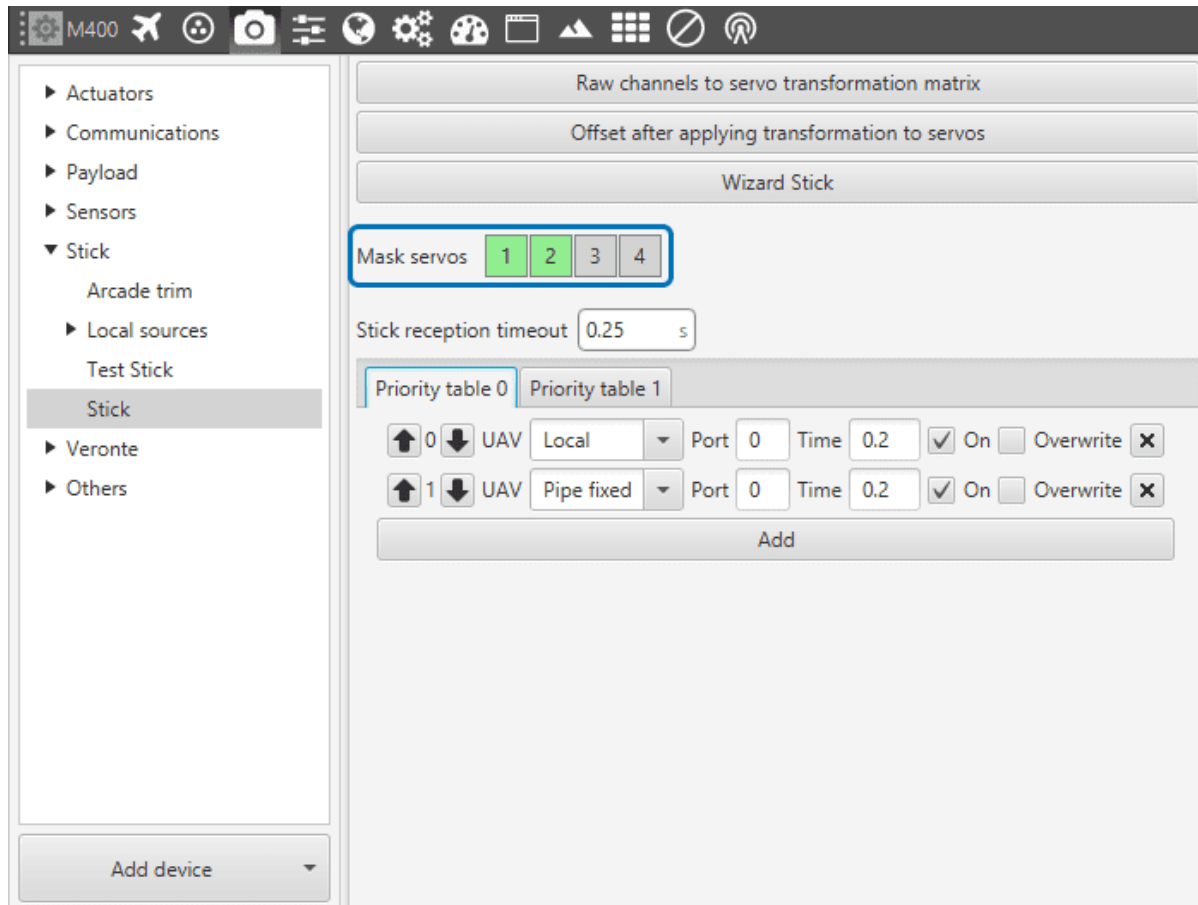
If *Get of Telemetry* is clicked, R_{neut} is taken from the connected transmitter. The user should move the sticks to their neutral position and then click on the button. Finally, to exit the assistant tool click on *Accept*.

Note: The assistant tool might not work for some configurations, such as **hybrid platforms**.

For those platforms where a same channel is used both for the plane and the quadcopter configuration, the stick configuration has to be done manually.

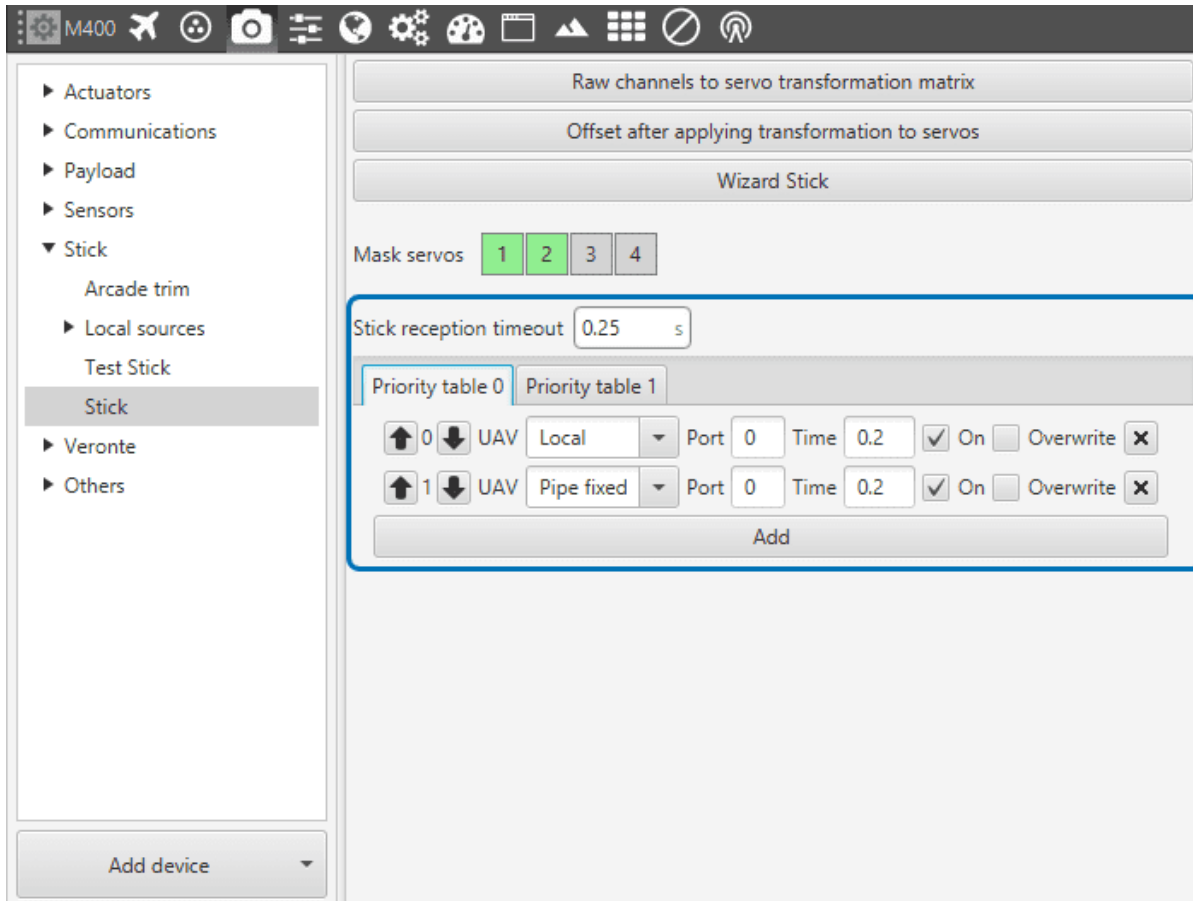
7.2.3.5.4.2 Mask Servos

This option is used to select which servos will “listen” to the commands sent by the stick. If the servo is selected (green box), it means the servo will be moved upon stick command. On the other side, if a servo is not selected (grey box), it will ignore the commands sent from the stick.



Stick - Mask Servos

7.2.3.5.4.3 Transmitter Inputs



Stick - Transmitter Inputs

It is possible to set multiple transmitter inputs with the respective priority, from top to bottom.

- **UAV:** defines the source of where the stick information is taken from. *Local* represents the actual selected autopilot (i.e. the transmitter is connected to it); *Pipe fixed* means the information is coming from the virtual stick; *Broadcast* means the information comes from all linked autopilots; and one particular autopilot, which needs to be visible in Pipe.
- **Port:** from each source it is possible to have more than one stick information, e.g. two transmitters can be connected to the same autopilot. The port is an identifier to distinguish them.
- **Time:** defines the time to consider the source inactive. Therefore the incoming stick information will be always the one from the source with higher priority and active. Once it is considered inactive the following active source will send its stick information.
- **ON:** enables the source.
- **Overwrite:** if marked, the stick information will be overwritten by this source when it becomes the active source.

In this section, the stick configuration on Veronte Pipe is explained. The next table details what is going to be covered:

| | |
|----------------------|---|
| Arcade trim | Neutral position value for the sticks |
| Local sources | External transmitter configuration (up to 2) and 1 virtual stick |
| Test Stick | Testing of the transmitter configured channels |
| Stick | Configuration of the stick matrices, offsets and transmitter priority |

All these menus help the user in configuring the transformation of stick inputs into control outputs. Summarized, the following process is followed:

- Stick inputs R are named “*Stick input r_n* ”, where n goes from 1 to the maximum number of employed channels of the transmitter. R values go through the YR transformation matrix (see **Devices -> Stick -> Stick** menu), which converts raw stick inputs to servo PWM signals. And become Y , which is named “*Stick input y_m* ”, where m goes from 1 to the total number of actuators of the platform.
- Y PWM signals are corrected by adding an offset matrix Y_0 . After that, Y values are converted into S values, known as “*Actuator outputs S_m* ”. The transformation is performed according to the relation defined in **Devices -> Actuators -> Physical** for each actuator.
- S values then go through the matrix US , defined in **Devices -> Actuators -> Logical**, and become U , the “*Control outputs U_n* ”.

Note: If the YR matrix has not been properly defined, Y values will be greater than 1 but the actuator output will have the maximum S available.

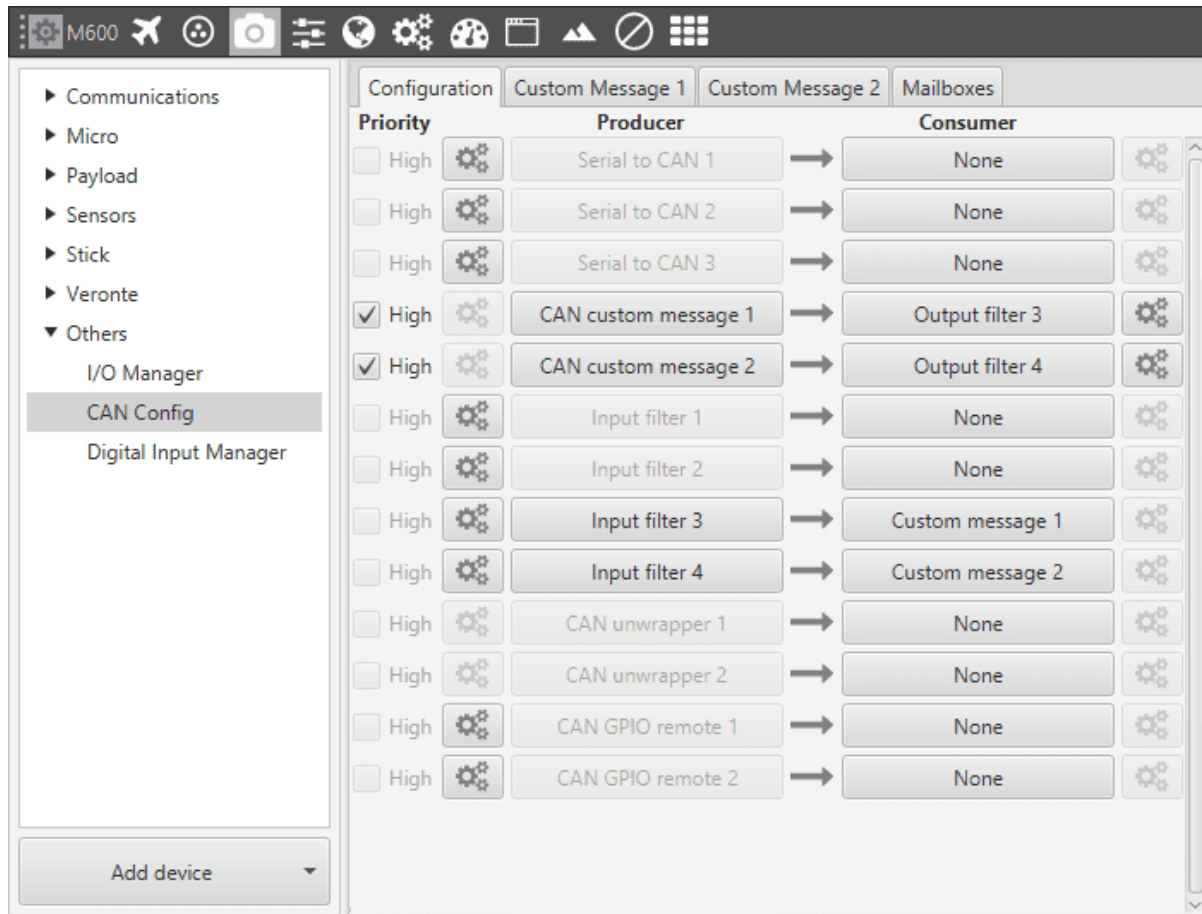
A broader and more detailed explanation of the above can be found in [Stick](#).

7.2.3.6 Others

7.2.3.6.1 Can Configuration

7.2.3.6.1.1 Configuration

This menu allows the configuration of communications between different devices.



CAN configuration

Attention: Make sure that the mask is set properly to be able to receive the desired CAN messages.

On the one hand, Veronte has the producers shown below:

- **Serial to CAN:** Serial messages over CAN output and it has to be connected to I/O Manager consumer. ID can be configured in the settings button.
- **CAN custom message:** CAN custom messages transmission.
- **Input filter:** CAN input filters. A mask can be set to filter messages in the settings button. Those CAN messages received in one filter **can no longer be received** in subsequent filters.
- **CAN unwrapper:** CAN messages over serial output and it has to be connected to I/O Manager consumer.
- **CAN GPIO remote:** CAN messages from GPIO peripherals such as CEX and Arbiter.

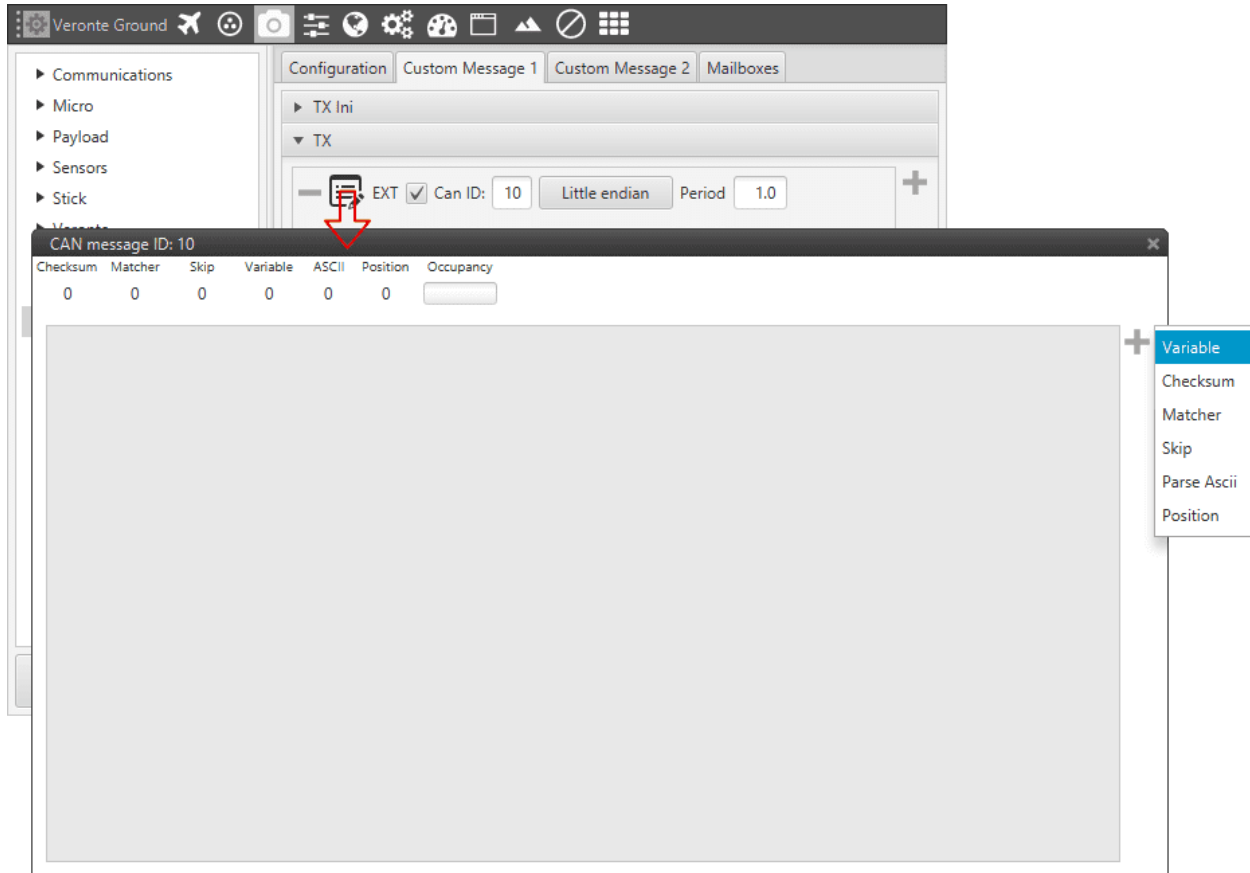
On the other hand, the consumers are the following:

- **CAN to serial:** Serial messages over CAN output and it has to be connected to I/O Manager producer.
- **Custom message:** CAN custom messages reception.
- **Output filter:** CAN output filters. CAN bus can be selected in settings button.
- **CAN wrapper:** CAN messages over serial output and it has to be connected to I/O Manager producer.

7.2.3.6.1.2 CAN Telemetry

7.2.3.6.1.3 Custom Messages

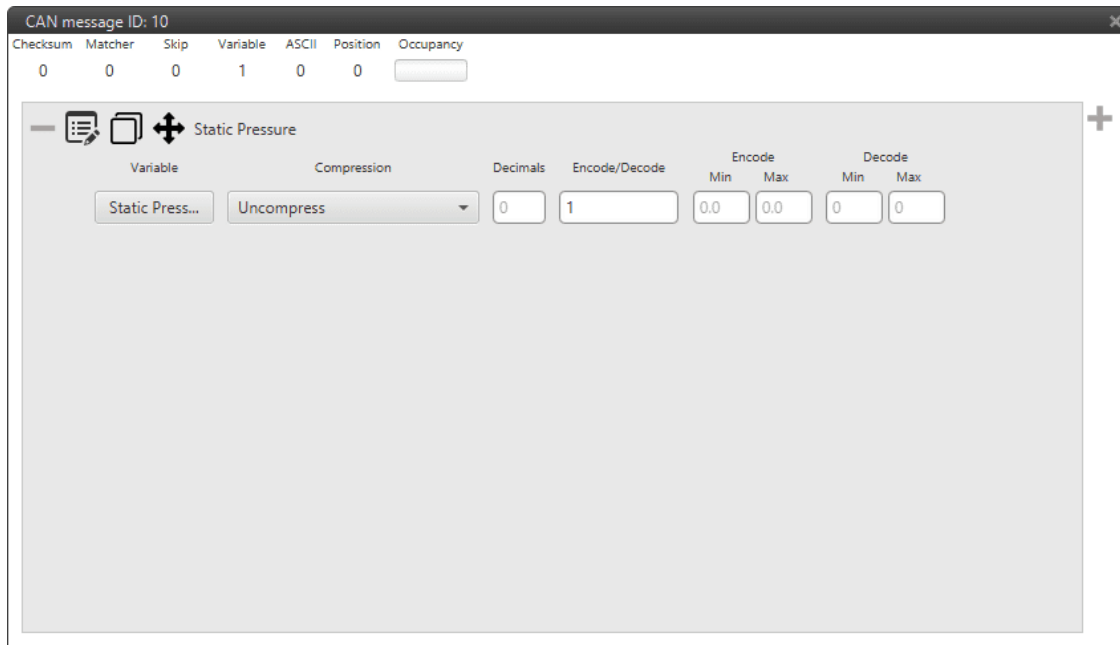
When using and configuring custom messages, the following types of elements can be defined within.



Telemetry Panel

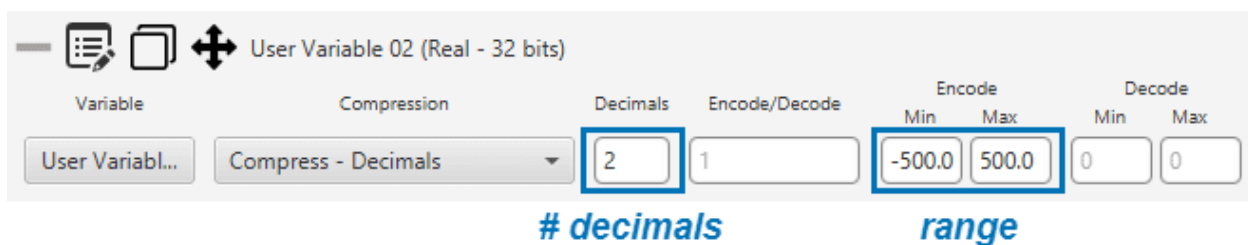
7.2.3.6.1.4 Variable

Used to store certain bits in a system variable (RX) or to send a certain variable (TX).



Variable Configuration Menu

- **Type of Compression.** The first step is to configure which kind of compression will be used for this variable.
 - **Uncompressed:** the variable is taken in its full length, with no value modification.
 - **Compressed:**
 - * **Compress (Bits signed):** specify the number of bits to be compressed to (negative values accepted). It is necessary that the user configures Encode/Decode options.
 - * **Compress (Bits unsigned):** specify the number of bits to be compressed to (no negative values accepted). It is necessary that the user configures Encode/Decode options.
 - * **Compress (Decimals):** the variable is compressed according to the number of decimals specified and the range specified (max and min values). The resultant compression (number of bits) follows the relation $(max - min) \cdot 10^{decimals}$ - which yields the encoding of the maximum value of the range (and the number of bits necessary for that). The range needs to be specified on the **Encode - Min/Max** field.



Compress Decimal Selection

- **Encode/Decode:** these values are used to apply a scaling factor after the transformation from binary to decimal value, or before the transformation from decimal to binary value.

In the example shown below, a real user variable (32 bits) is being used to receive data from an external device. This data corresponds to the heading angle of the aircraft (which goes from 0 to 359 degrees). The device is sending this

information in a 16-bit data frame and the angle value times 100 (hence why the **Decode** section goes from 0 to 35900). This needs to be saved in Veronte in the user variable in the range 0 to 359 (**Encode**).

Variable Identification Example

7.2.3.6.1.5 Checksum

Sometimes, control codes are needed for preventing random errors in transmission, where a bits frame is operated and the result is sent to the receiver to check it. To do so the CheckSum option is used.

Checksum

- **Back From:** Indicates that the CRC will be computed from the indicated byte (inclusive).
- **Back To:** Indicates that the CRC will be computed to the indicated byte (exclusive).

Note: Byte 0 it is referred to the first byte of the Checksum block. i.e for a message with 6 bytes where 2 bytes are the CRC. Back from = 4 and Back To = 0

- **Endianness:** Indicates how the bytes that it contains are read:
 - **Big endian:** Set the value from left to right
 - **Little endian:** Set the value from right to left
 - **Mixed endian:** For some special devices.

- **Type:** User can choose the type of CRC that will be applied.
- **CRC type.**
 - **Polynomial.** Select from a list of predefined Embention Veronte CRC. Last option is Custom, where fields as n° Bits, Polynomial, Start Value, Final Xor, Reflect In and Out can be defined. Check Polynomial CRC online for more information.
 - **Module.** Applies Module CRC.
 - **Mavlink.** Embention has implemented the Mavlink checksum, used only for Mavlink protocol communications.
- **Sum 8.** The Sum 8 checksum is the 8 bit modulo 256 checksum. It consists of adding all bytes to take only the less significant 8 bits.
 - $SUM = \text{Sum on all bytes}$
 - $CHECKSUM = MOD(SUM, 256)$

7.2.3.6.1.6 Matcher

This option is used to send a constant value through the bus in TX or wait for a particular value in RX.

The screenshot shows a window titled "CAN message ID: 10". At the top, there is a row of checkboxes: Checksum (0), Matcher (1), Skip (0), Variable (0), ASCII (0), Position (0), and Occupancy (empty). Below this, the "Matcher" tab is selected, showing a configuration area for "Matcher x9". It includes three input fields: "Value" (9), "Bits" (8), and "Mask" (255 dec). There are also icons for list, copy, and a plus sign for adding more matchers.

Matcher

- **N° Bits:** number of bits in which the matcher is performed.
- **Value:** sent/received value for the above n° of bits.
- **Mask:** it is automatically set when the n° of bits is assigned.

For example, a matcher of 8 bits with a value of 9 will be reading/sending: 0000 1001 .

7.2.3.6.1.7 Skip

This option is used to discard a certain number of bits from the message (the maximum number of bits that can be skipped with a single “Skip” are 32). This tool can be used when there are variables incoming that are from no interest for the user, not loading unnecessary information into the system.

7.2.3.6.1.8 Parse ASCII

Parsing ASCII is used when the protocol required is of this kind. Only for RX. The variable set at the top is where the ASCII will be saved.

ASCII protocol is used for transforming a character array into decimal values. For such task, the user needs to define the number of characters in the integer and the decimal parts, as well as defining which is the division character. See how to introduce all this information in the picture below.

CAN message ID: 10

| Checksum | Matcher | Skip | Variable | ASCII | Position | Occupancy |
|----------|---------|------|----------|-------|----------|-----------|
| 0 | 0 | 0 | 0 | 1 | 0 | |

Parse ASCII

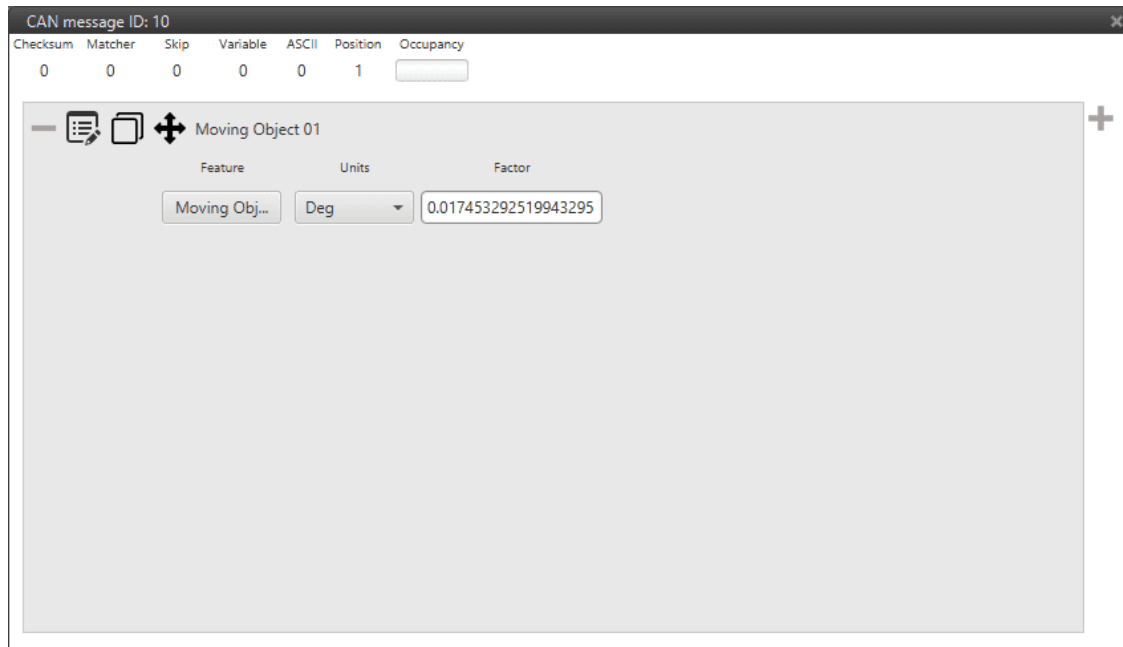
| Variable | Char in inter part | Char in decimal part | Division char |
|-----------------|--------------------|----------------------|---------------|
| IAS (Indicat... | 1 | 0 | . |

Parse ASCII Menu

7.2.3.6.1.9 Position

Position is used to input/output a data set with a particular format. When created, the user can only choose from Moving Object variables.

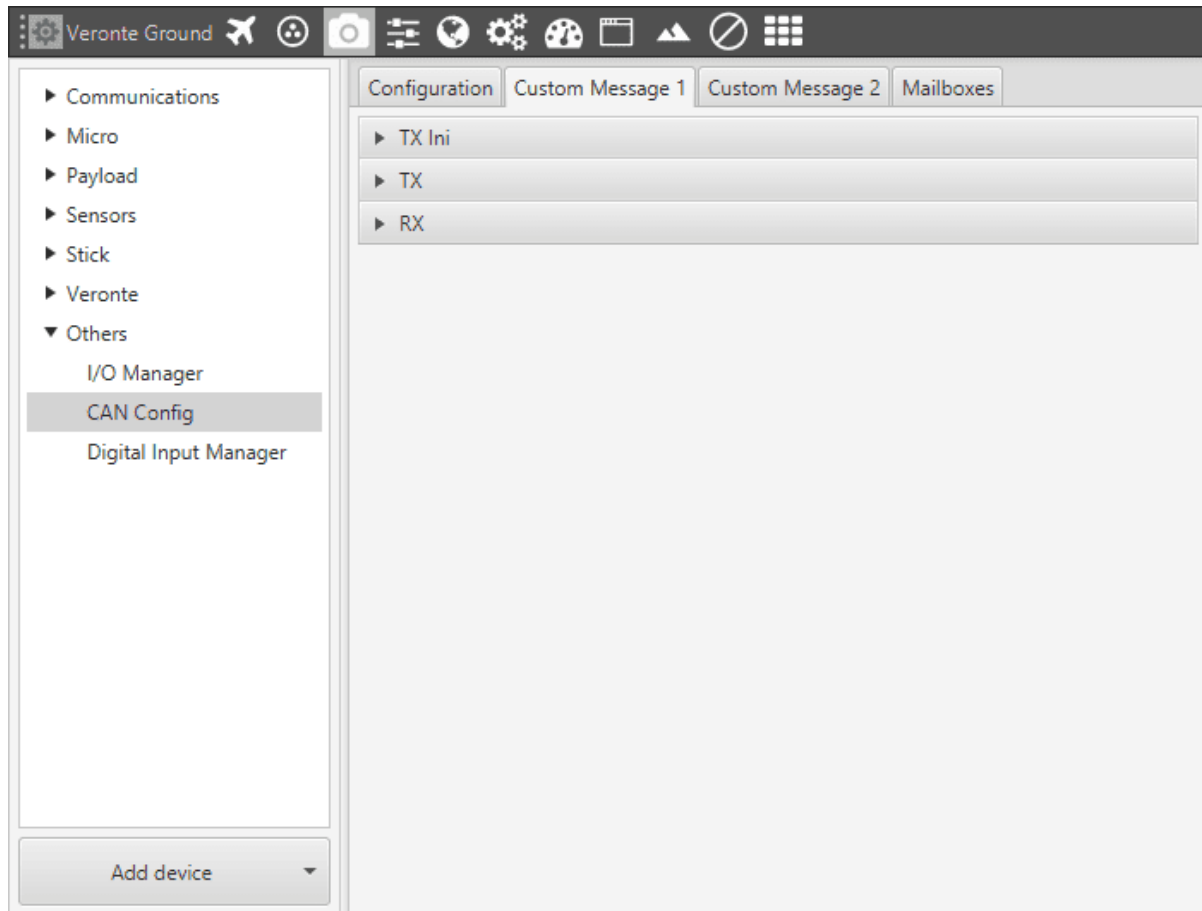
The window display below is the configurable menu, where it can be chosen between degrees and radians as units. The information stored is the WGS84 coordinates in the following order: Latitude, Longitude and Height. All of them are stored with double precision.



Position Menu

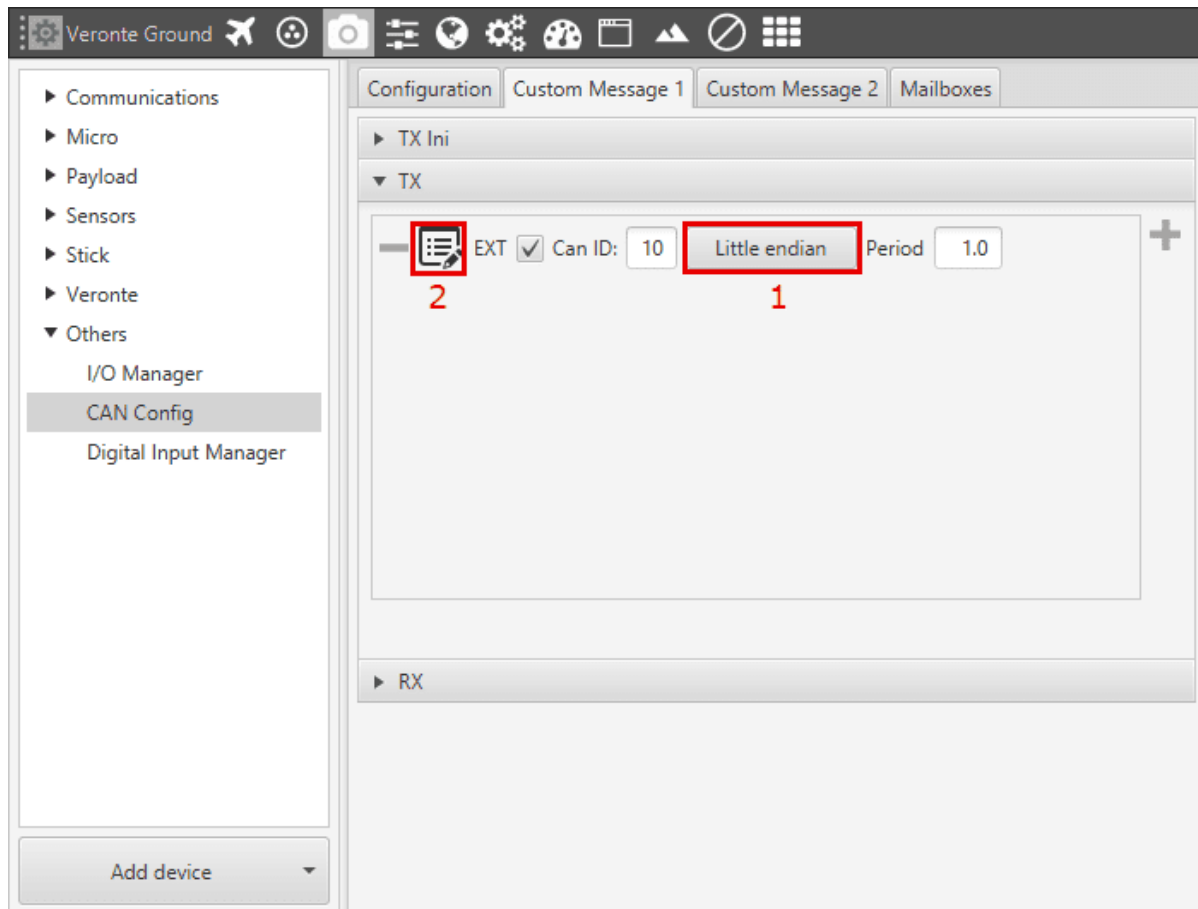
On the custom message tabs, the user chooses the variables to be sent/received over the CAN buses. The following elements can be configured:

- **TX Ini:** used to configure transmitted messages that are only sent once at the beginning of the operation (sent when the autopilot boots up). They can be used to initialize some devices.
- **TX:** used to configure transmitted messages.
- **RX:** used to configure the reception messages (where they are stored).



CAN – Telemetry

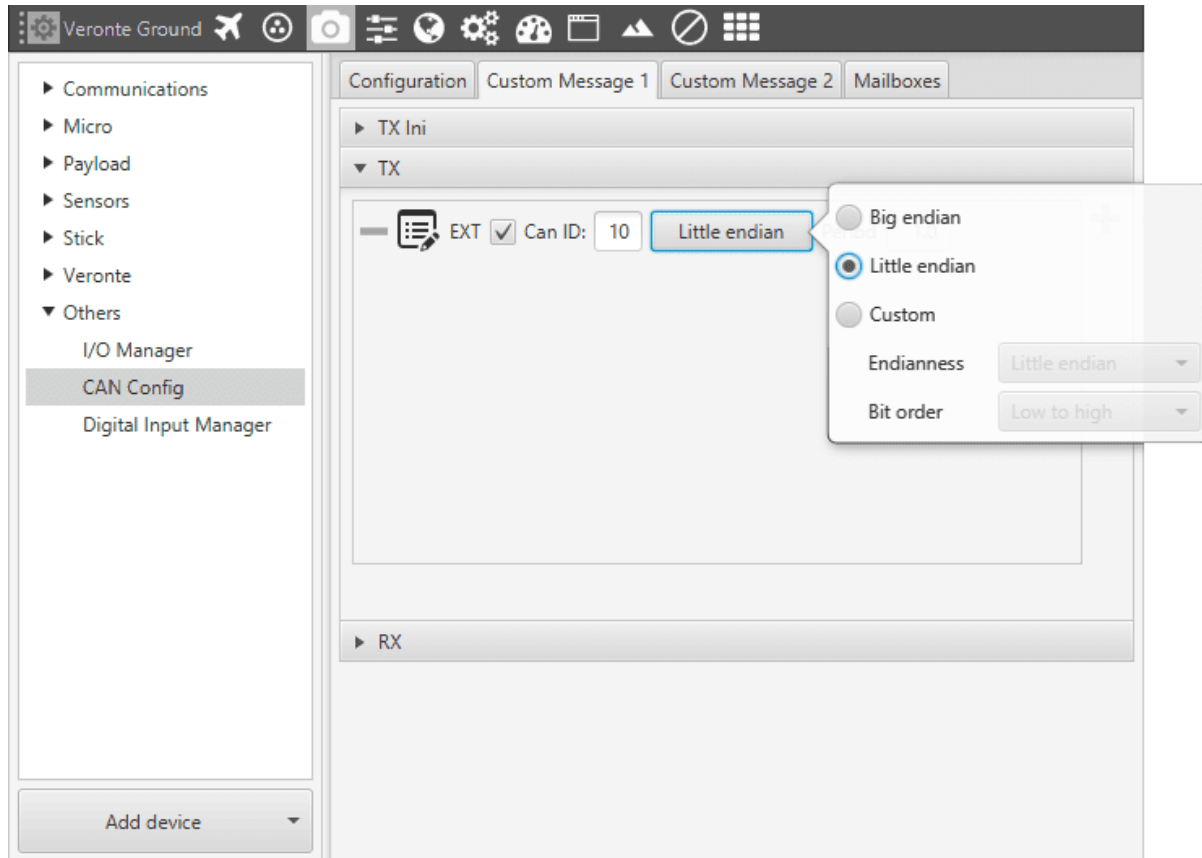
7.2.3.6.1.10 TX Messages



CAN – Telemetry - TX

In order to add a new custom message the user needs to press “+” and a new element will be added into the panel.

- **EXT:** enables the frame format with a 29-bit identifier.
- **ID:** 11 or 29-bits (Extended) ID used to identify TX messages. The value set has to be decimal format.
- **Period:** time in seconds between TX messages delivery.
- **Button 1:** open a new window to configure the endianness of the message, which indicates how the bytes that it contains are sent/read:
 - **Big endian:** set the value from left to right
 - **Little endian:** set the value from right to left
 - **Mixed endian:** custom endiannes.



CAN – Telemetry – TX Endianness

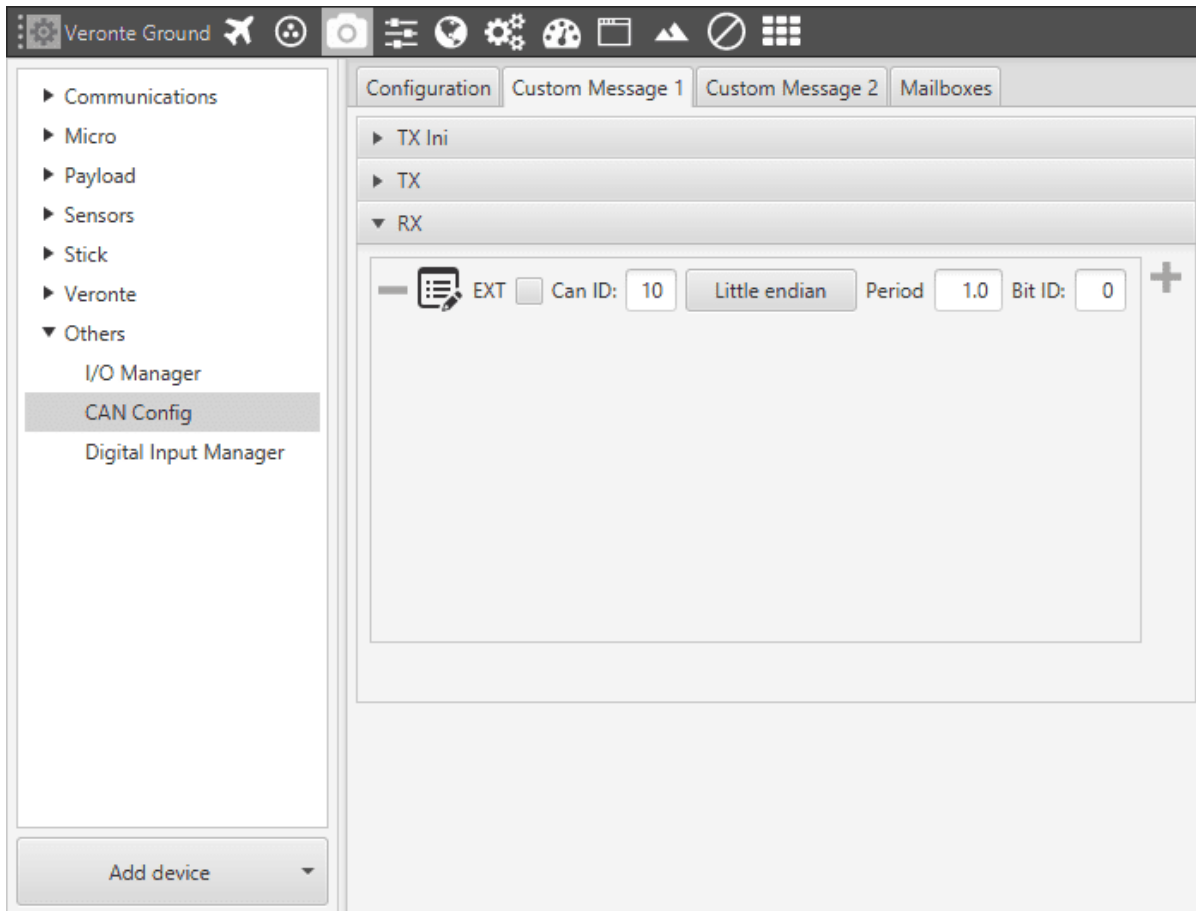
- **Button 2:** displays the menu to configure how the bits/bytes of the message are divided and sent.

There are six different elements that can be added when setting up a custom message: *Variable*, *Checksum*, *Matcher*, *Skip*, *Parse Ascii* and *Position*. This is covered in section [Custom Messages](#).

The **maximum capacity** of a CAN message is 64 bits (8 bytes), so to send more information it must be divided into several messages.

7.2.3.6.1.11 RX Messages

The procedure is similar to the one followed in TX messages.



CAN – Telemetry - RX

The custom message needs to have the expected ID with which the external device/sensor is going to be sending information.

Attention: It is important to configure a mailbox for every single reception ID. See [Mailboxes](#) for more information.

The custom message structure needs to match the reception data-format. User variables (real - 32 bits , integer - 16 bits or boolean - 1 bit) need to be used to store that data.

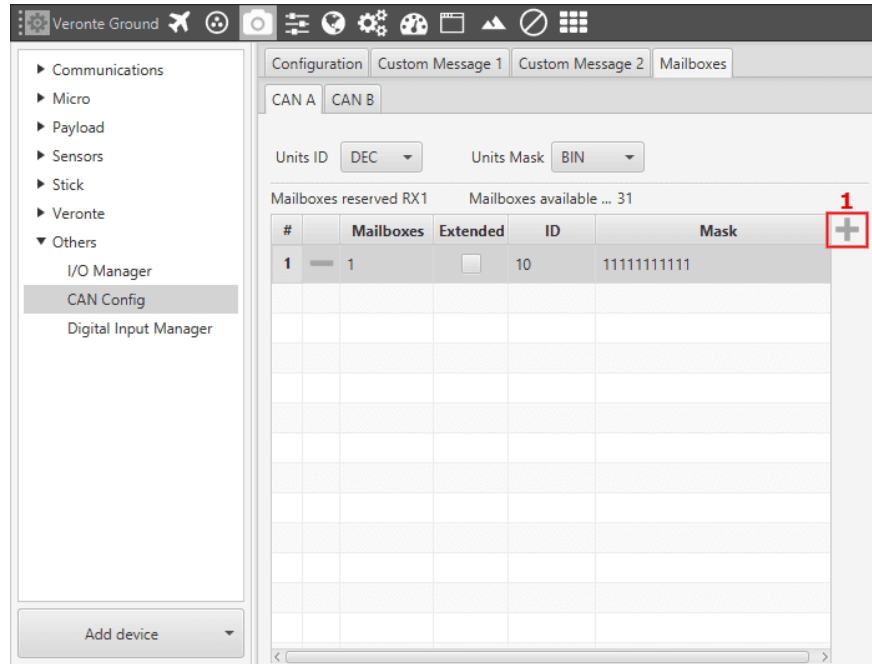
Unlike TX messages there are two additional variables per message, **Bit ID** and **Period**. The user bit selected in **Bit ID** box will be true if the message is being received correctly. **Period** is the time between receptions to consider that it is not being received correctly. For example, if period is set to 1s and it passes more than 1s since the last reception, the bit ID will be set to false.

Warning: Pay attention that the user bit selected in **Bit ID** is not in use for another task.

7.2.3.6.1.12 Mailboxes

Here the user can modify the mailboxes from the CAN bus.

When Veronte is going to receive data on the CAN Bus, it is mandatory to configure a certain number of mailboxes. In order to add a mailbox one needs to click on (1).



CAN – Mailboxes

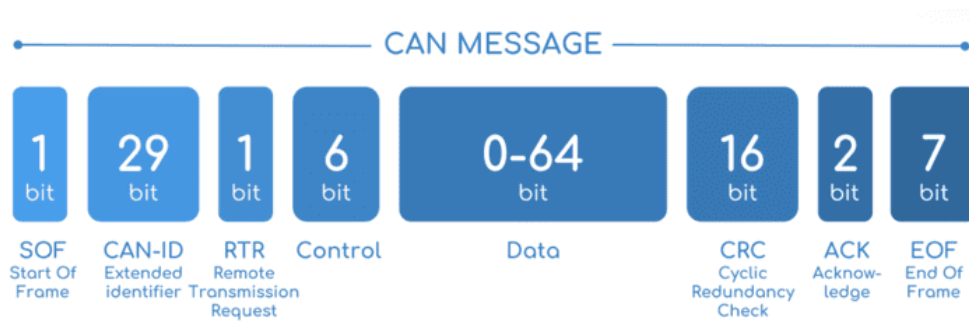
Mailboxes are required to store the data received until Veronte reads it and it is **necessary at least one mailbox** per RX message. Veronte allows up to 32 mailboxes. The options available when adding a new mailbox are:

- **Mailboxes:** number of mailboxes assigned to that ID.
- **Extended:** select this option for 29-bit IDs.
- **ID:** 11 or 29-bits (Extended) ID used to identify RX messages. The value set can be defined in decimal, hexadecimal or binary form.
- **Mask:** This filter is configured for reception messages; received data will be stored on mailboxes where message ID coincides with mailbox ID. Mask adds some flexibility on the reception, when comparing message with mailbox data, only the value of binary digits configured as 1 on the mask will be taken into account. (e.g, for a configuration MASK: 11 000 and ID:10 110 all incoming messages addressed to 10 XXX will be received in this mailbox).

Warning: If any mailbox is full and another message arrives, the **new** message is discarded.

A Controller Area Network (CAN Bus) is a robust vehicle bus standard widely used in the aviation sector. Veronte is fitted with two CAN buses that can be configured independently.

The structure of a CAN message can be seen in the following image:



CAN message structure

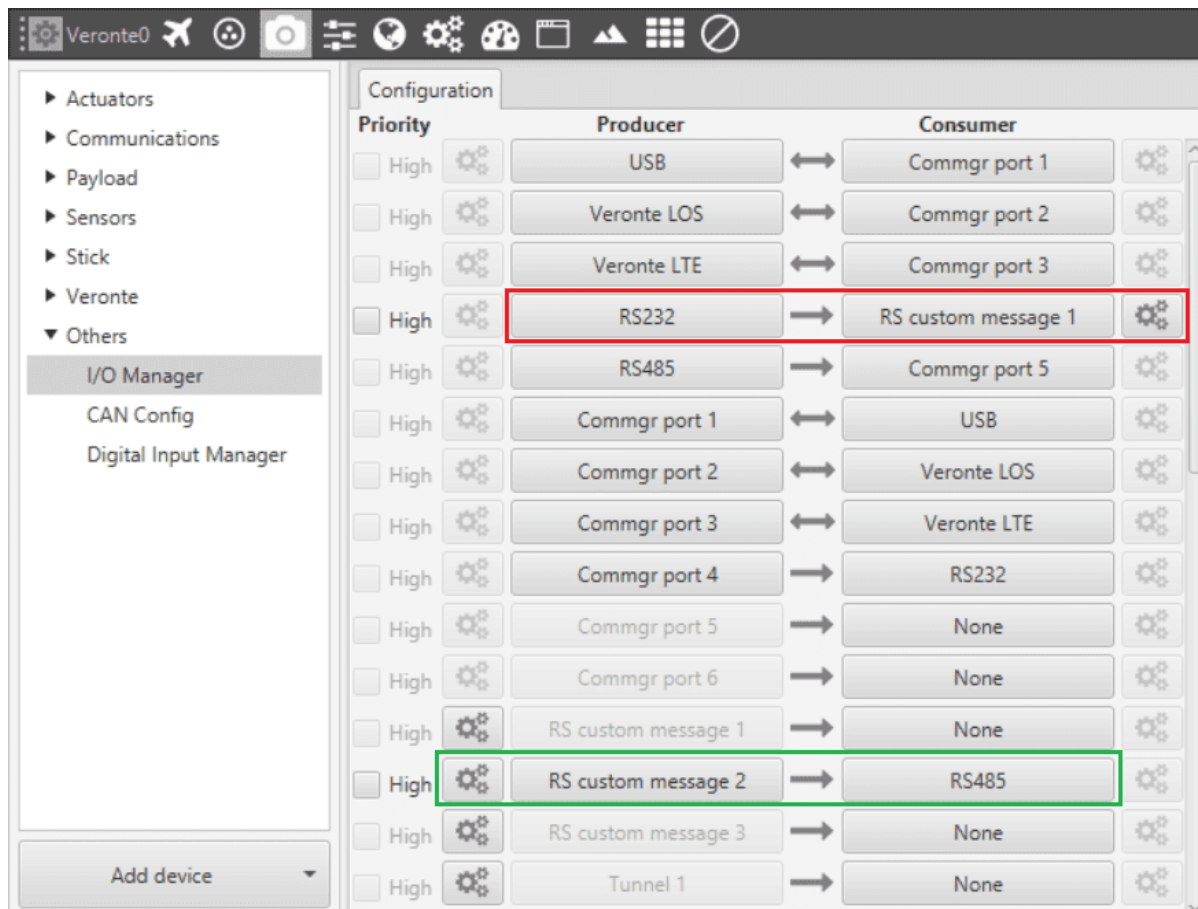
Only the ID is introduced in the system, the rest of the message layout is already coded. The data field is the one build by the user to send, and parsed when received.

The baud rate of both CAN buses can be configured in Setup - Connections - *CAN*.

7.2.3.6.2 I/O Manager

7.2.3.6.2.1 Serial Custom Messages

It is possible to configure the messages sent/received through the serial port and its conversion to system variables by selecting the option Custom message and configuring the I/O port.



XPC Unit8 Panel – Custom Message

In the image above can be seen two possible configurations using a Custom Message. The first one is configured to receive a determined message from a RS-232 serial port and the second is used to send a Custom Message through a RS-485 serial port. It is possible to use the same Custom Message for both tasks if Bind Bidirectional is used (the arrow indicates this). Custom Message configuration is covered in section *Custom Messages*.

The options available when configuring a custom message are:

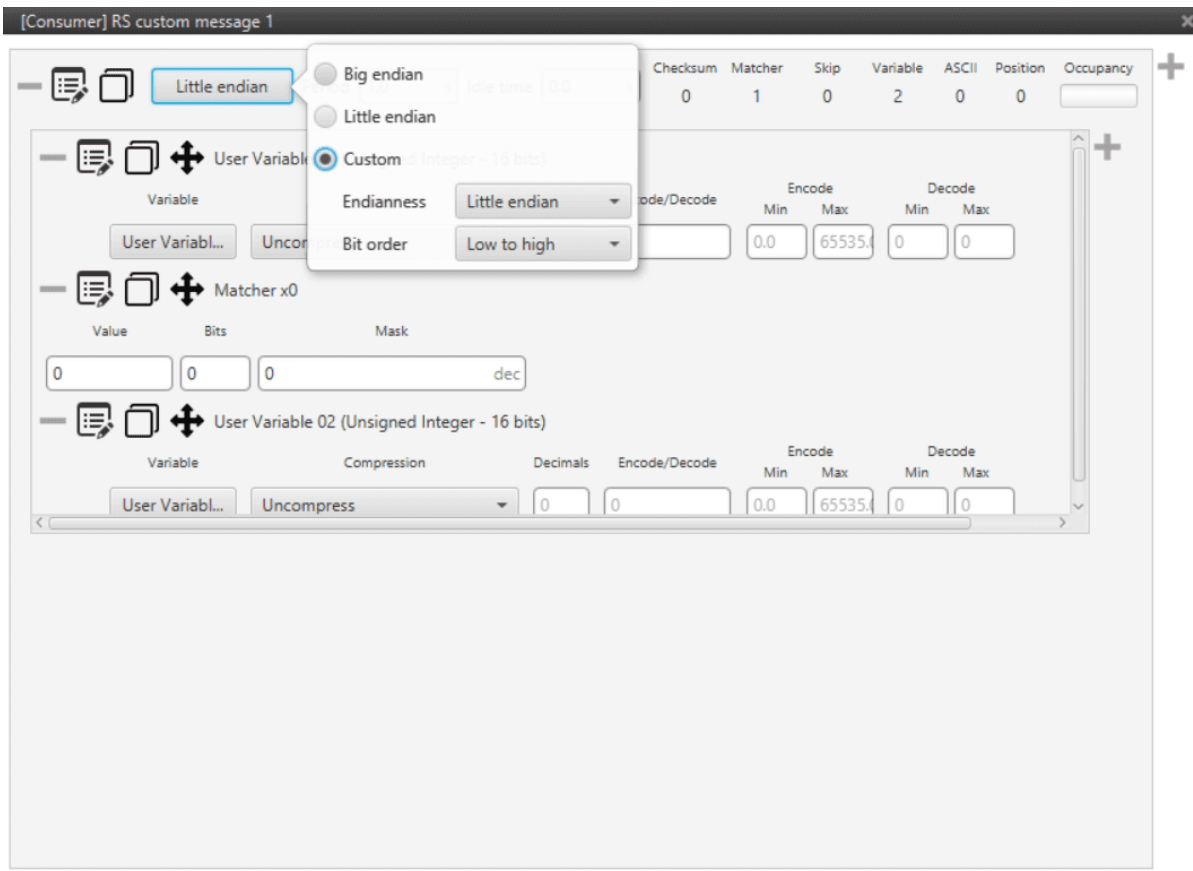
- **Period:** this option has a dual role depending on if it is used to transmit or receive data.
 - For transmitting, it is the inverse of the send frequency.
 - For receiving, it is the time Veronte wait for the message.
- **Endianness:** depending on the order in which the device issue the message, it is possible to select:
 - **Big endian:** set the value from left to right
 - **Little endian:** set the value from right to left
 - **Mixed endian:** Any devices have this format. If you need to configure, please contact us.

The following type of messages are available to configure a structure: **Variable, Checksum, Matcher and Skip**.

Before configuring any message, user has to know the structure it has to have according to the device that is connected to the port. Each device may have a different message structure when it sends or receives information.

7.2.3.6.2.2 Message received

In this section is shown a possible structure when receiving a message.



Custom Messages Configuration Menu

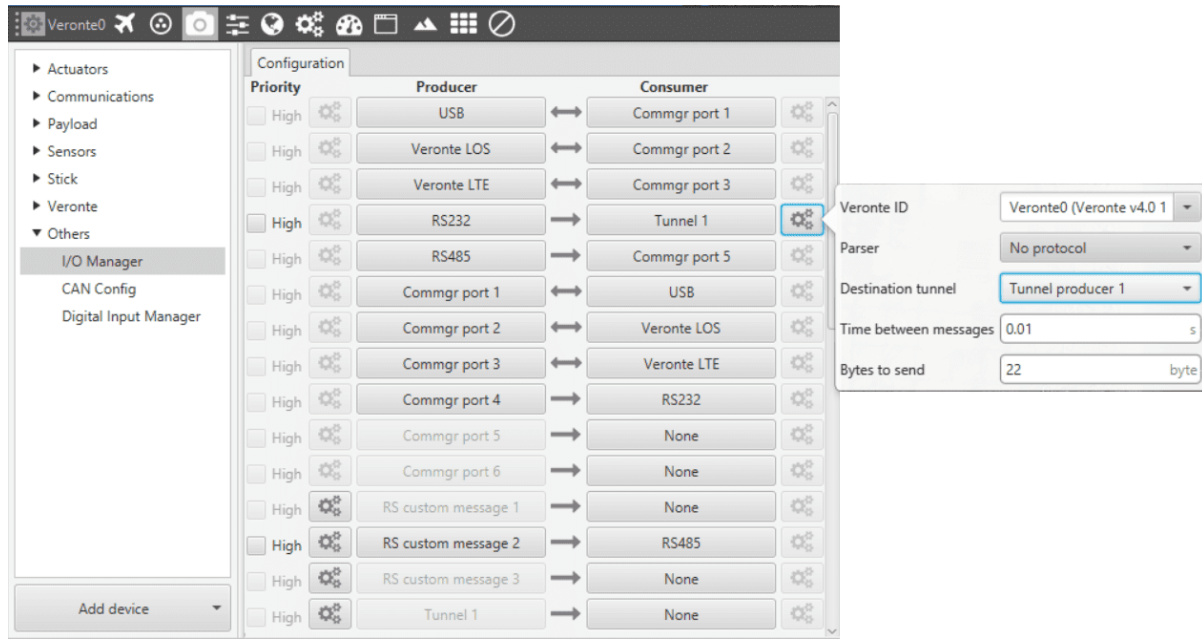
1. Select the desired **I/O** port and then select **Custom Message**.
2. Configure the custom message. Press **the configuration button** and another window will be displayed. In this window press “+” to add a message (system variables or ADSB Vehicle), when it is already added, select an **Endianness** and set a value for **Period**.
3. Create the message structure by adding fields to it. In this example, the structure of the message received consists of a **Matcher** (to identify the message) and two **User Variable** (where the information is stored).
4. Check the number of bits of each element. User has to know previously the structure of the message received and its size, each element has to be configured according to that.

7.2.3.6.2.3 Tunnel

It is possible to configure a Tunnel which is a bidirectional bridge between Veronte Units that communicate to each other sharing information about an external device connected to the Serial or Digital port.

Imagine that we would like to have a button connected to the air autopilot to launch a parachute. It is not possible to physically connect the button because the air autopilot is in the flying platform, so we need a different option. Here is where the tunnel becomes useful. We could connect the button through the Serial or Digital port to the Ground Autopilot, and then with the tunnel send the signal to the air one. With this configuration it would be like if the button were physically connected to the aircraft.

Let's consider the following image.



XPC Uint8 Configuration Menu

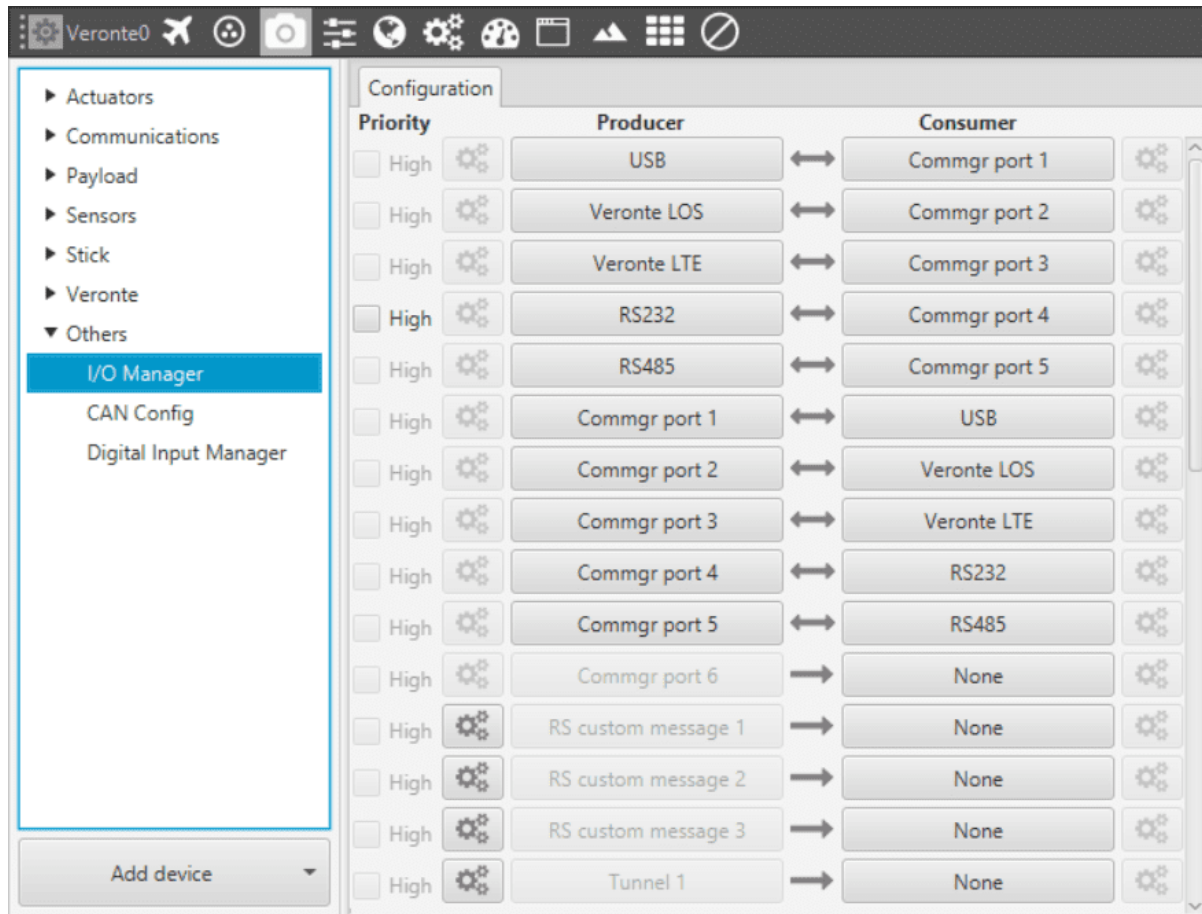
In the image above there is a device connected to the SCID Connector 232 (Producer) and there is a Tunnel (Consumer) which sends that information to other Veronte with a determined ID. On the other hand, Veronte Air has to be configured to receive the signal sent by other device. In that case the Producer will be Tunnel and Consumer will be the port or destination tunnel where the device is connected.

The option available when configuring Tunnel as Consumer are:

- **Veronte ID:** identifies the Autopilot which will receive the information.
- **Parser:** chooses protocol to parse message data.
- **Destination tunnel:** number of port is used to avoid mistakes and identify a Tunnel when using more than one.
- **Time between messages.**
- **Bytes to send:** sets the message size to send.

When configuring the unit that receives the information, only is necessary to configure the “Consumer” of that a information, generally a serial port.

In this panel the user can establish the relationship between a determined signal with a I/O port. This allows users to configure an external sensors, messages between Veronte Units (Tunnel) and custom messages.



XPC Uint8 Configuration Menu

Firstly, users have to configure the **Producer** selecting the I/O port or information to use. Later, users have to configure the Consumer by clicking on an element, a new window will be displayed to select an item. The relationship between them can be unidirectional (Blind) or bidirectional (Blind Bidirectional), the last enables a port to receive or send information.

The following I/O ports are available:

| Field | Description |
|-------------|------------------|
| USB | USB Port |
| SCIB | Radio |
| SCIA | 4G Connection |
| SCID | Serial Port 232 |
| SCIC | Serial Port 485 |
| Commgr port | COM Manager Port |

It is worth noting that only SCID/SCIC ports can be used to configure an element, the other ports have to be maintained by default.

Finally, it is possible to configure the next elements:

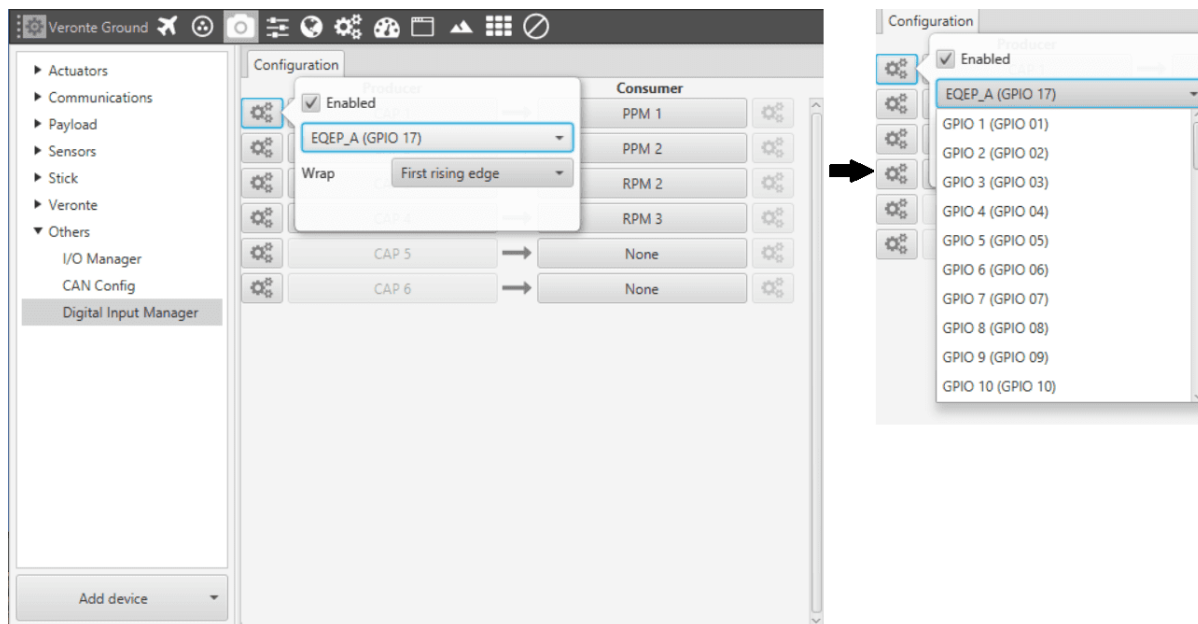
| Field | Description |
|----------------|--|
| Tunnel | Creates a bidirectional bridge between two devices, see section Tunnel |
| Custom Message | This allows user to send/receiver a custom message.see section Custom Messages |
| GPS RTCM | Differential GPS. |
| Magneto | External magnetometer sensor, see section Magnetometer |
| Ultrasound | External ultrasound sensor, see section Ultrasounds |
| Splitter | Used to split a signal into 2. |
| NMEA Parser | NMEA 0183 messages parser, see section GNSS External |

More information about each element can be found in the links provided before.

7.2.3.6.3 Digital Input Manager

Pins 1-8 and 10-16 can work as Digital Input or Output as well as PWM. In order to configure some custom sensors such as a Stick PPM, RPM Sensor or a Pulse Sensor Enhanced Quadrature Encoder Pulse Inputs (EQEP) pins are reserved, which correspond to pins 55-58. These pins can also be used as Digital I/O.

Sensors using a Digital In are configured in this menu.



XPC CAP Configuration Menu

The process to configure a device can be done as follows:

- Select a **Producer**.
- Click on a button to select the type of **Consumer**, it is possible to choose among a Stick PPM, RPM Sensor or Pulse Sensor.
- Configure the **Producer**, press on the configuration button and a new pop-up window will show.
- In the pop-up window it is possible to select the pin where the device is connected. Pins available are **GPIO** 1 to 16, and **EQEP** A, B, S and I. When using the harness provided by Embention the transmitter Digital Input is connected to the pin 55 (EQEP_A) with pin 49 as Ground.

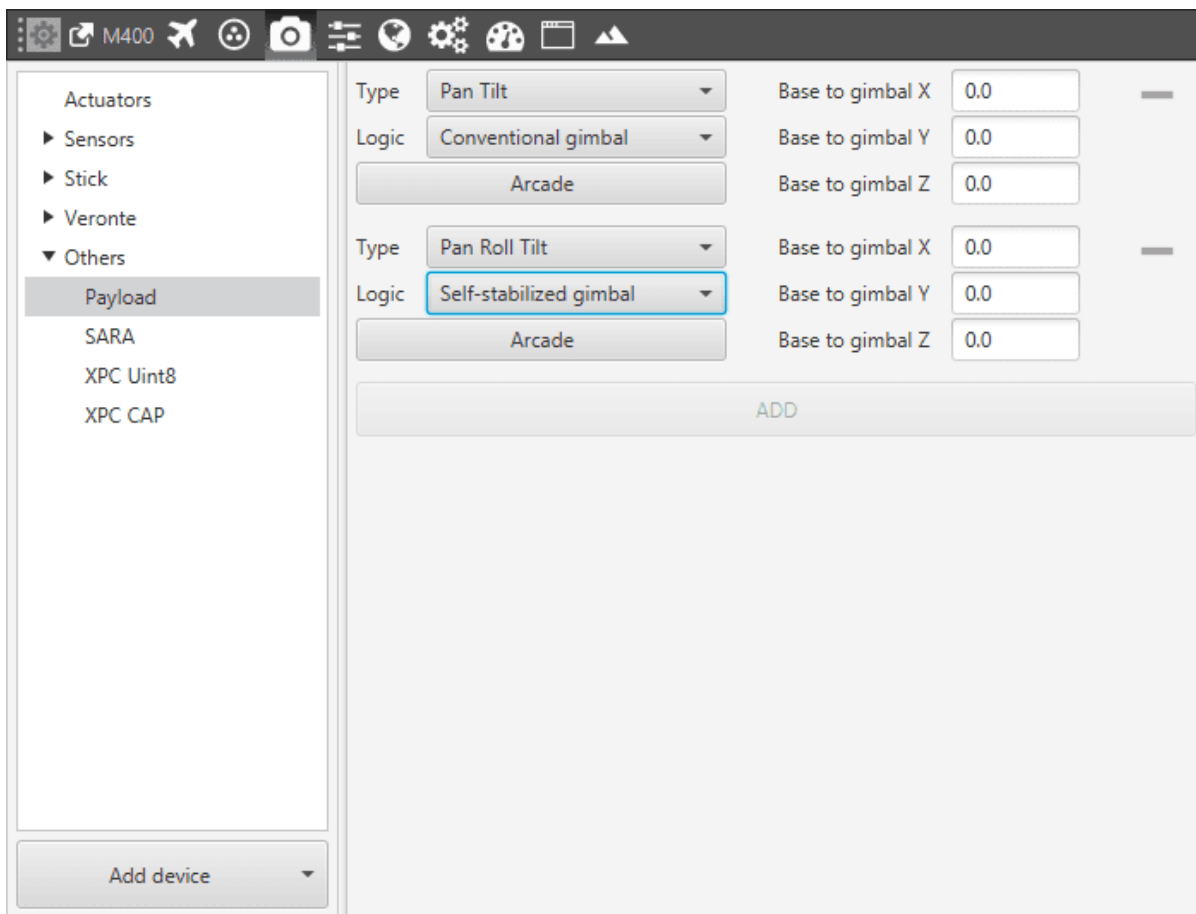
- Wrap options allows to configure how the information is treated: for example if it will read the first rising edge or the first falling one. Clicking on the arrows it can be configured as desired.

Warning: Do not use EQEP_B to read PPM, RPM or Pulse with Veronte units of **Hardware version 4.5 or lower**. It can only be used as digital I/O.

For more information on the stick configuration and how to set the RPMs correctly, visit : [Stick](#) and [RPM](#).

7.2.3.6.4 Payload

This panel allows users to configure the gimbal settings, including the number of gimbal axis, servos used for gimbal connection and distance from the autopilot to the Gimbal.



Payload Configuration Menu

- **Type:** user can install a Pan -Tilt, Roll-Tilt or Pan-Roll-Tilt Gimbal, depending on the degrees of freedom of the Gimbal Device.
- **Logic:** Conventional gimbal or Self-stabilized gimbal.
- **Base to gimbal X, Y, Z:** relative position between the Gimbal and Veronte.

It is possible to configure two gimbals at the same time, press **ADD** to incorporate one into the Panel.

7.2.3.6.4.1 Payload Operation

In order to configure a complete Payload Operation, the following steps have to be done. All steps are based on a Conventional Gimbal with two axes of rotation (Pan and Tilt).

- 1. Configure the US matrix selecting the control outputs (U) and actuator outputs (S). This has to be done according to the Gimbal used.

Edit

-

-

-

-

-

-

Control Output u1 - Pitching

Control Output u2 - Thrusting

Control Output u3 - Rolling

Control Output u4 - Yawing

Control Output u5 - Pan G1

Control Output u6 - Tilt G1

-

Motor 1

-

Motor 2

-

Motor 3

-

Motor 4

-

Actuator Output s5

-

Actuator Output s6

-0.35355338

1.0

-0.35355338

0.25

0.0

0.0

0.35355338

1.0

-0.35355338

-0.25

0.0

0.0

0.35355338

1.0

0.35355338

0.25

0.0

0.0

-0.35355338

1.0

0.35355338

-0.25

0.0

0.0

0.0

0.0

0.0

0.0

1.0

0.0

0.0

0.0

0.0

0.0

0.0

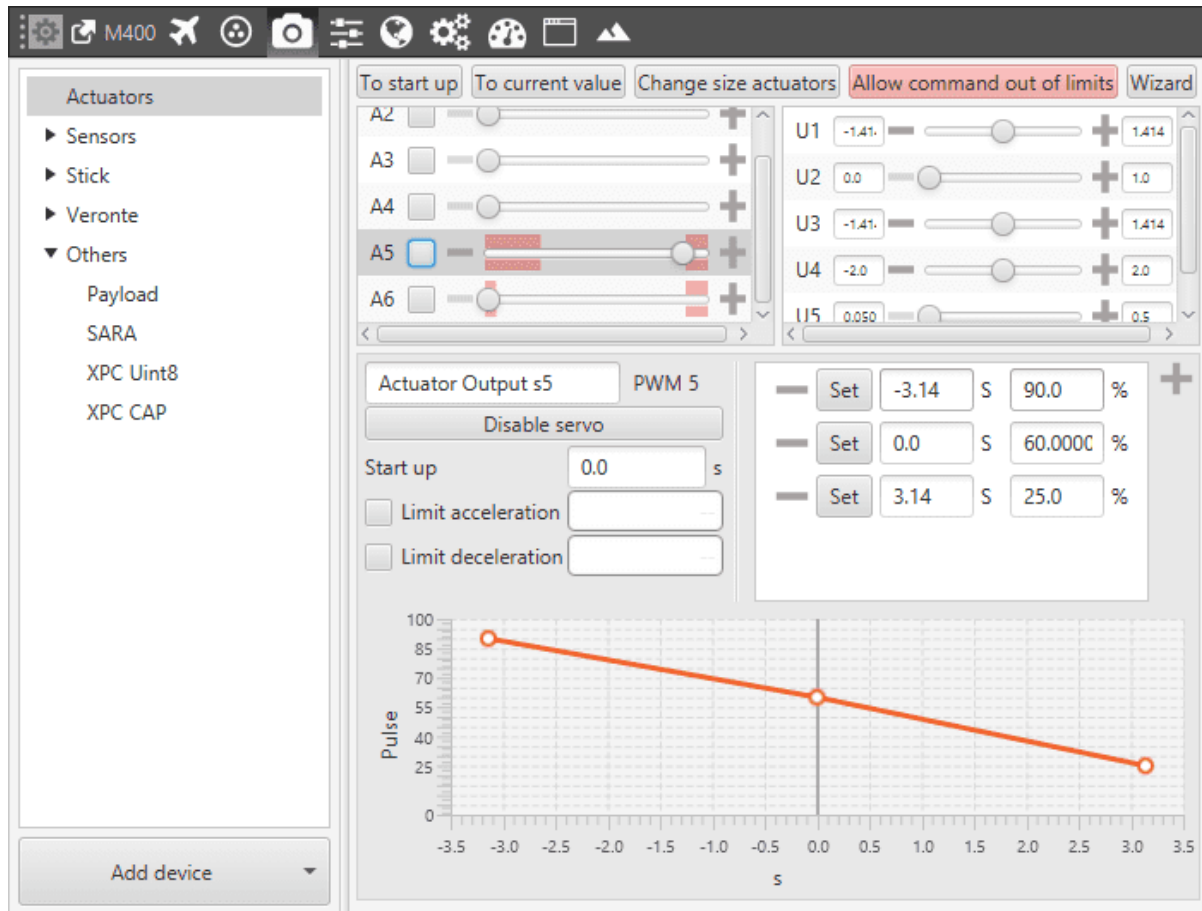
1.0

+

Apply

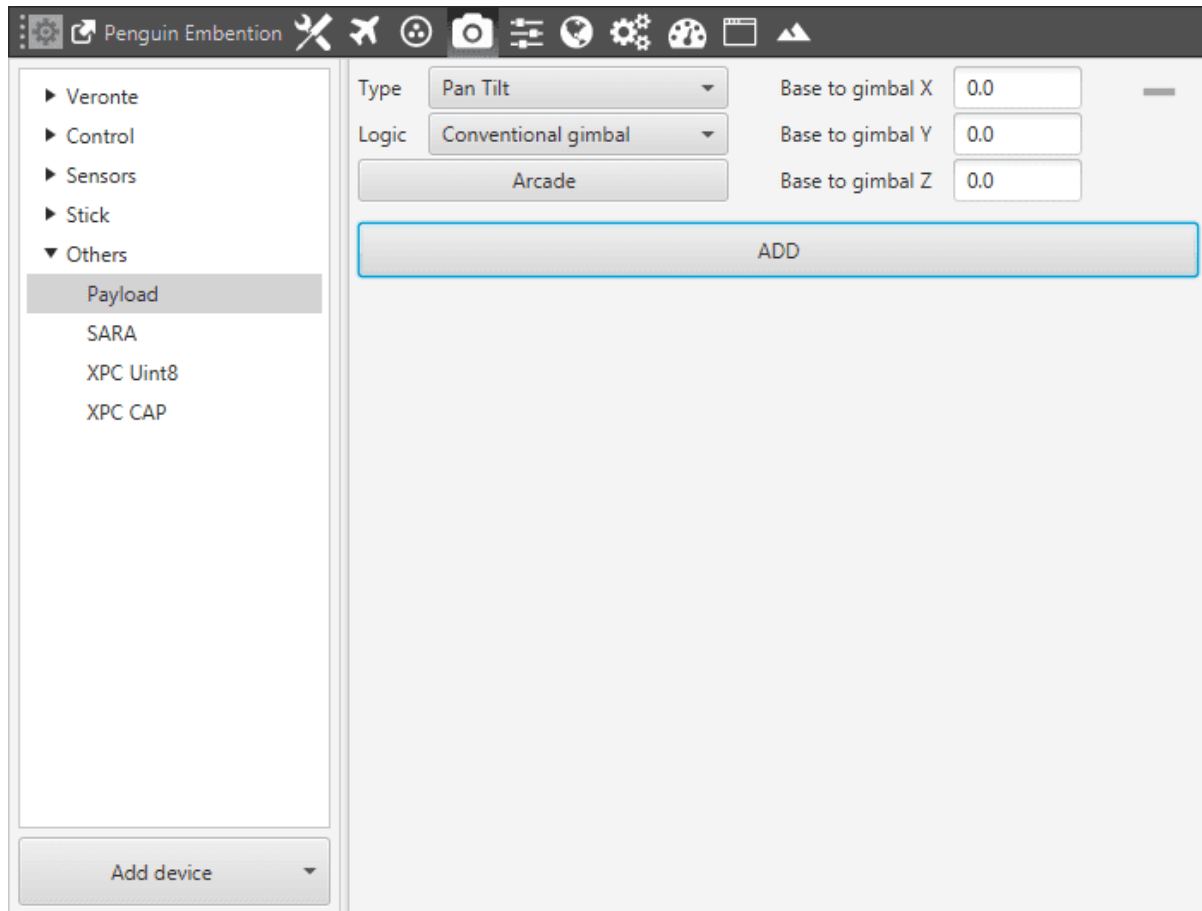
SU Matrix

- 2. Configure the relationship between PWM signals and actuator outputs (S), it depends on the device used, prior to configure this check performances. It may requiere to add PWM Outputs, in its correspondent panel, Connections.



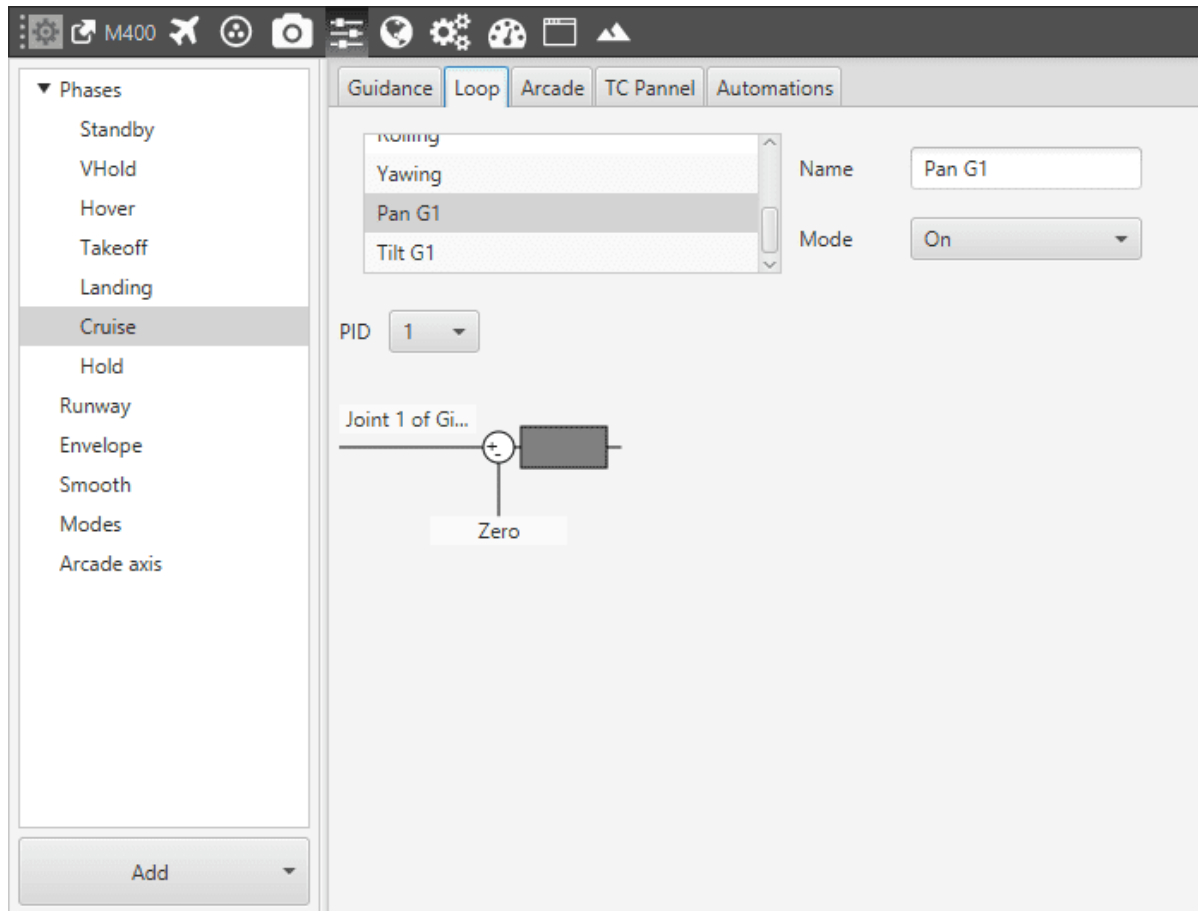
SU Matrix

3. Configure the Payload Device.



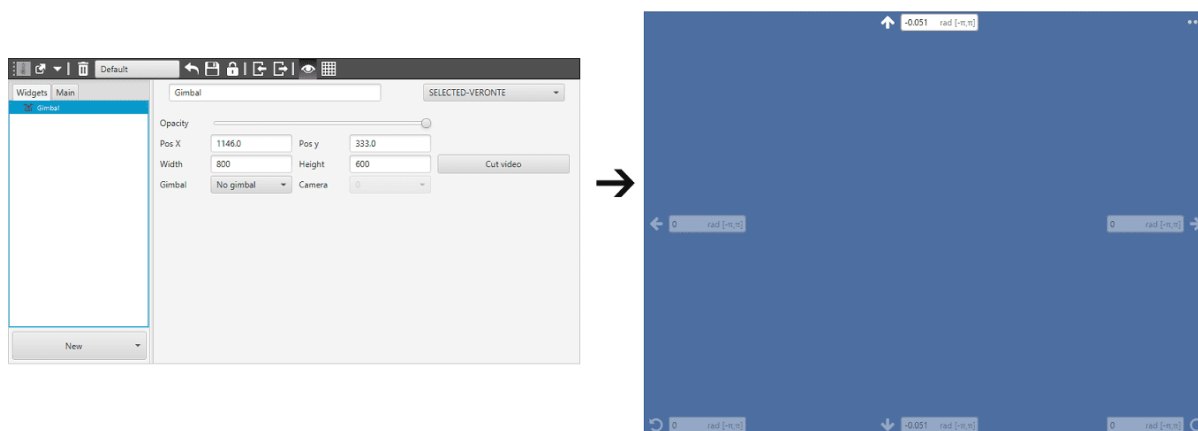
Payload Configuration

4. Next step is the configuration of the control loops for each axe: Pan and Tilt. Control loops can be either close loop or open loop. It depends on if there is any variable available for feedback.
 - Pan control: select **Joint 1 of gimbal 1** as entry variable (desired Pan angle) and if exists a feedback variable select it, otherwise select the variable **Zero**, this sets a open loop.
 - Tilt control: select **Joint 2 of gimbal 1** as entry variable (desired Tilt angle) and if exists a feedback variable select it, otherwise select the variable **Zero**, this sets a open loop.



Control Loop

- A typical setting for the control loop incorporates: a proportional and integral gain.
5. The last step consists in configuring the widget that displays all the information related to the gimbal. Generally, this device incorporates a Camera being possible to stream the signal in this widget. To configure this widget go to the Workspace Panel and add the widget of Gimbal. In this panel is possible to select a **Gimbal** and **Camera** if there is any device available.



Widget Configuration

When configuring this widget the window above will be shown, this allows the user to control the gimbal. The movements are controlled with the arrows shown on the window, the user can press on them to move the gimbal or introduce a value.

This section of the manual contains the information about external sensors/devices configuration. These devices are configured on the different ports available in Veronte (CAN buses, Serial buses, digital I/O, etc):

| Port / Manager | Description |
|-----------------------|--|
| CAN Configuration | Configuration of the two CAN buses (A and B) |
| I/O Manager | Configuration of serial ports, LOS & BLOS connection , and others. |
| Digital Input Manager | Configuration of PPM signals, pulses or rpm sensors. |

As Custom Messages need to be defined for both serial and CAN communication, there will be a section after the **CAN Configuration** and **I/O Manager** sections.

Devices Panel permits to configure any device (payload, sensors...) connected to Veronte and the internal Veronte ones.

| Type | Description |
|----------------|---|
| Actuators | To configure the actuators limits, their logic and saturation |
| Communications | 4G, Comstats and Iridium configurations |
| Payload | Menus for different Cameras/Transponder configurations |
| Sensors | To set sensor device |
| Stick | To configure Stick |
| Veronte | To set the orientation and mass centre of the aircraft |
| Others | To set other devices and I/O signals |

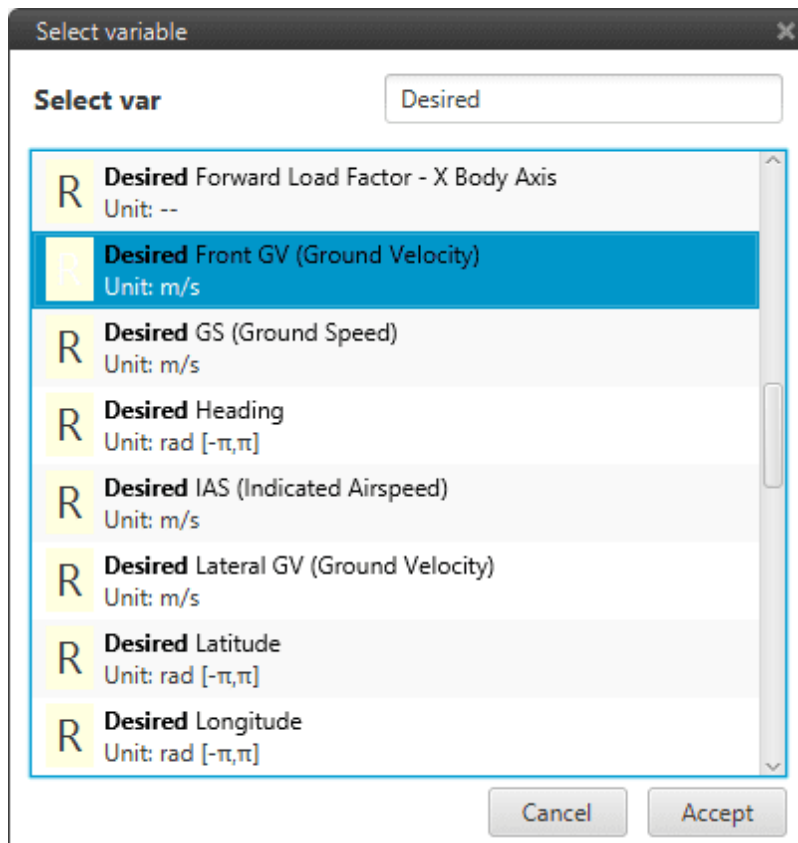
7.2.4 Control

7.2.4.1 Phases

7.2.4.1.1 Guidance

7.2.4.1.1.1 Guidance Variables

The guidances contained within Veronte generate a series of variables that are later used in the control system as the input of the PIDs. In the Hold guidance is the user the one that selects the desired variables to be generated. These variable can be added by clicking on the “+” button, there are different variables to select. Generally, variables named as Desired are used in this Guidance.

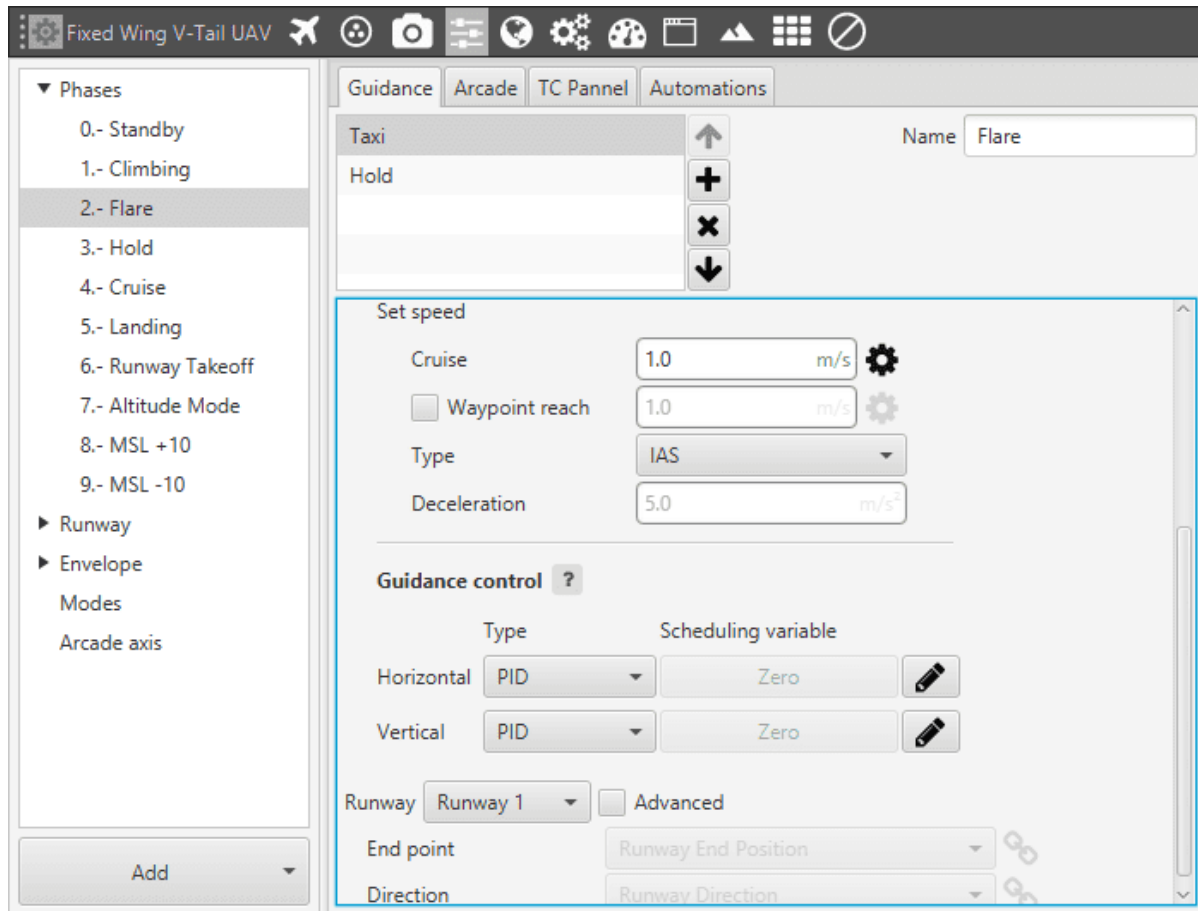


Variables Selection

After that these variables are used in the Control Loops.

7.2.4.1.1.2 Taxi

Taxi guidance is used to create a linear path along the runway that is followed by the aircraft. This command is normally used in the take-off phase, where the airplane is wanted to keep the direction of the runway while is accelerating until the lift-off point.

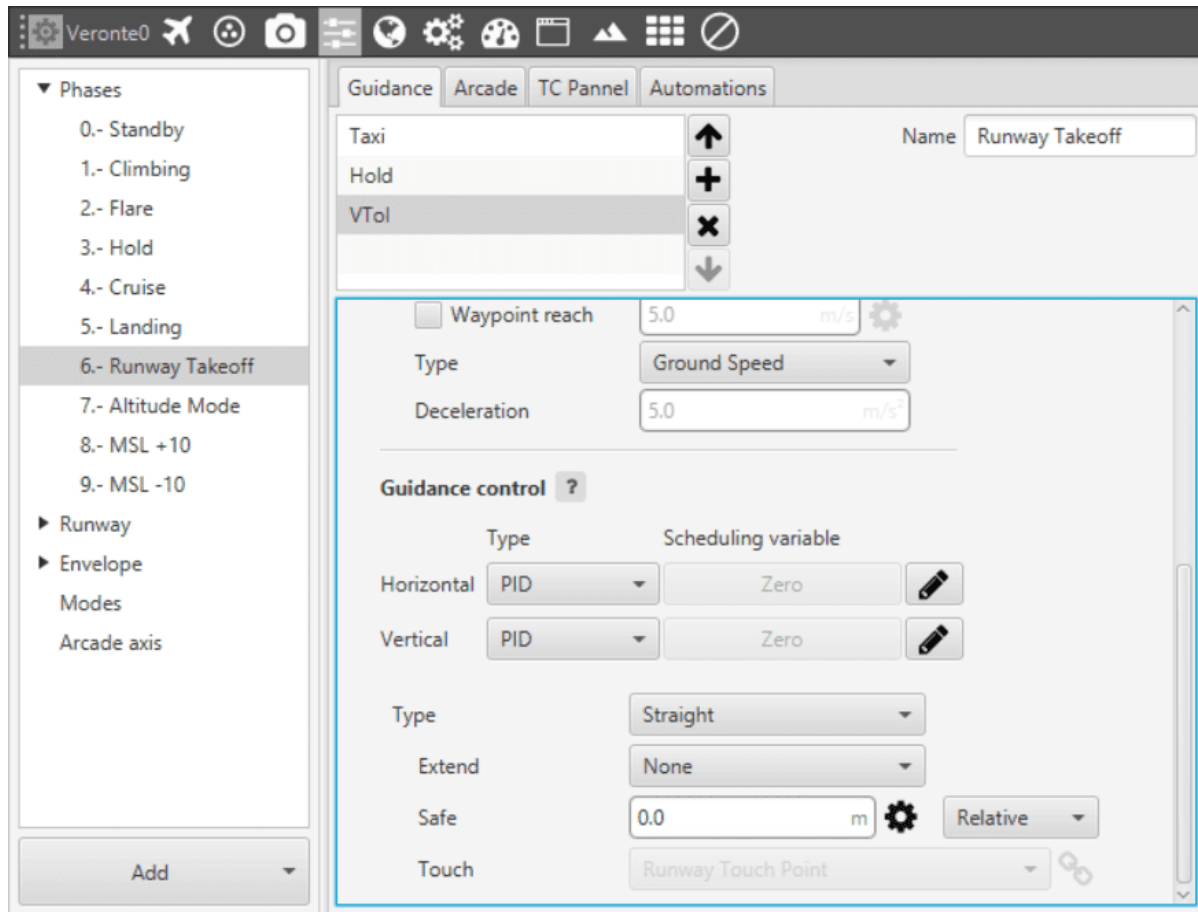


TAXI Guidance

- **Set height mode:** the height mode indicates how the aircraft will perform the route. Height mode in Taxi guidance is fixed to 2.5D type. This means that vertical flight is not allowed, so horizontal guidance has priority over vertical guidance.
- **Arcade position/speed transition** In Arcade mode the trajectory generated (*position*) is not followed and instead the aircraft moves according to the commanded *speed*. The *horizontal* and *vertical speed* parameters serve as the upper-thresholds for when the aircraft's guidance should be *position-based*, even in Arcade mode. This parameters are mainly useful for platforms like multicopters.
- **Set speed:** this option sets the speed that the vehicle will have during this phase. It can be **IAS** (indicated airspeed) or **Speed** (Ground Speed). Normally, IAS is used for airplanes and Speed for multicopters. The option **Waypoint reach** is used to indicate the speed at which the platform will reach the waypoints, so it will travel along the path with the speed indicated in the option **Cruise**, then it will decelerate or accelerate to the speed indicated in Waypoint Reach and then it will go back to the cruise speed.
- **Guidance control:** vertical and horizontal PIDs (with constants gains or variable that control this phase (see [Blocks](#))).
- **Runway:** here it is selected a runway previously configured, see section [Runway](#). Besides, it is possible to use the advanced mode and select a different end point or direction.

7.2.4.1.1.3 VTOL

VTOL guidance (vertical take-off and landing) is used in multicopters for the take-off and landing operations. This guidance consists on the creation of a vertical line that starts at the point where the platform enters in this guidance.



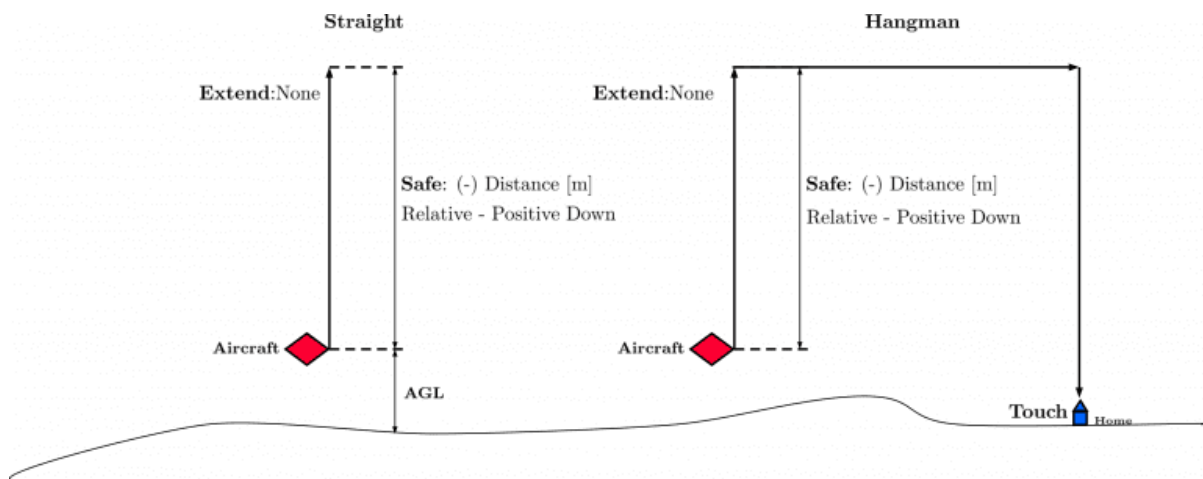
VTOL Guidance

- **Set height mode.** Height mode indicates how the aircraft will perform the route. Height mode in VTOL guidance is fixed to 3D type. This means that vertical flight is allowed, so both horizontal and vertical guidance have the same priority level.
- **Arcade position/speed transition** In Arcade mode the trajectory generated (*position*) is not followed and instead the aircraft moves according to the commanded *speed*. The *horizontal* and *vertical speed* parameters serve as the upper-thresholds for when the aircraft's guidance should be *position-based*, even in Arcade mode. This parameters are mainly useful for platforms like multicopters.
- **Set Speed:** this option sets the desired aircraft speed during the manoeuvre.
 - **Cruise:** sets the velocity modulus of the commanded velocity.
 - **Waypoint reach** indicates the speed at which the platform will reach the waypoints of the path, i.e. it will travel along the path with the speed indicated in the option *Cruise* and then it will decelerate or accelerate to the speed indicated in here.
 - **Defined speed type** can be IAS (indicated airspeed) or Ground speed. Normally, IAS is used for airplanes and Ground speed for multicopters.
 - **Deceleration:** maximum allowed acceleration/deceleration to meet the desired velocity.

- **Guidance control:** vertical and horizontal PIDs (with constant gains or variable gains based on a schedule or a table) that control this phase. The structure of PID is explained in [Blocks](#).
- **Type:** this parameters are used to indicate the route followed by the multicoter during the take-off and landing. The path Straight consists on a vertical line from the point where the vehicle enters in this phase. In the case of a take-off, the line goes from the ground to an altitude indicated by the user. The second option, Hangman, the path consists on a vertical and horizontal line.
 - **Extend:** when Up or Down are selected, the value set in Safe will be discard, and the platform will ascend or descend, until a next change.
 - **Safe:** this parameter defines the altitude the aircraft reach, it can be Relative (starting from the initial point of the route – current platform position) or an Absolute altitude (MSL,AGL or WGS84). As an example, in a take-off operation, an altitude of -10000 meters can be indicated as the final point of the route, so it is sure that the multicopter will keep climbing until another phase is commended (via automation or manually). The same procedure is done in the landing, indicating a big relative distance (for example 100 meters from the starting point), so it is sure that the vehicle reaches the ground, and an automation is set to stop the platform when it touches the surface.
 - **Touch:** additional parameter to be configured when the type Hangman is selected. It defines a point that the aircraft has to reach. For instance, after go Up/Down the value set, the aircraft will perform an horizontal movement according to the point defined. Finally, when the aircraft is over the point, it will descend till reach that point. Usually, this option is used when there are obstacles in the area and by performing this movement the platform can avoid them and land safely.

Note: when the option relative is selected, a positive value will made the aircraft descend. Therefore, this value is Positive down.

The following image gives an overview of some parameters introduced:



Parameters Overview

7.2.4.1.1.4 Guidance-generated Variables

VTOL guidance generates a vertical straight trajectory. It also generates a set of variables which can be used in the control loops. The list of variables is the following:

- Desired position.
- Track position.
- Track state (current patch, last patch).
- Desired latitude, desired longitude, desired WGS84, desired MSL, desired AGL.
- Desired velocity.
- Desired front groundspeed, desired lateral groundspeed, desired velocity down.
- Desired tangential acceleration.
- Desired IAS.
- Guidance north error.
- Guidance east error.
- Guidance down error.
- Desired body velocities.
- Desired velocities north, east, down.
- Desired heading, FPA and bank.
- Route-guidance distance - tangential component.
- Route-guidance distance - horizontal component.
- Route-guidance distance - perpendicular component.

7.2.4.1.1.5 Climb

Climb guidance is used to make the aircraft climb from the start of the phase to another altitude. Commonly, this guidance is used after the take-off to fly from the ground to cruise altitude through a loiter point, but it can be employed for other purposes.

Fixed Wing V-Tail UAV

Phases

- 0.- Standby
- 1.- Climbing
- 2.- Flare
- 3.- Hold
- 4.- Cruise
- 5.- Landing
- 6.- Runway Takeoff
- 7.- Altitude Mode
- 8.- MSL +10
- 9.- MSL -10

Runway

Envelope

Modes

Arcade axis

Guidance

Climbing

Name Climbing

Section 0

Set height mode 2.5D

Arcade position/speed transition

Horizontal 1.0 m/s

Vertical 1.0 m/s

Set speed

Cruise 20.0 m/s

Waypoint reach 20.0 m/s

Type IAS

Deceleration 0.5 m/s²

Axes controller

Horizontal PID Zero

Vertical PID Zero

Runway Runway 1 Advanced

Loiter point Runway Loiter

Direction Runway Direction

Loiter pos is center

Taxi extension 1800.0 m

Horizontal distance (dxy) 1800.0 m

Radius head turn (R3) 250.0 m


Radius loiter (R1) 250.0 m

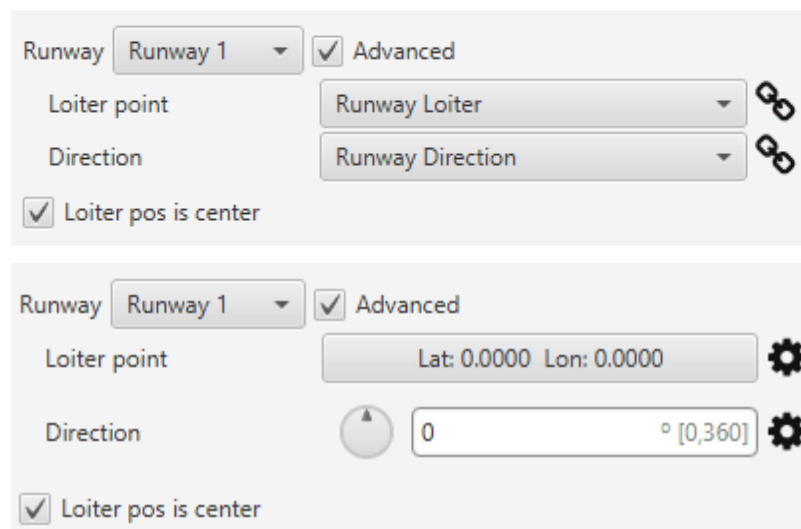
Flight path angle 0.1 rad [-π,π]

Climbing Guidance Parameters

The climbing path is automatically generated and is not directly shown to the user until the aircraft enters this phase. This is due to the algorithm recalculating the path each time to take into account the aircraft's actual flight conditions

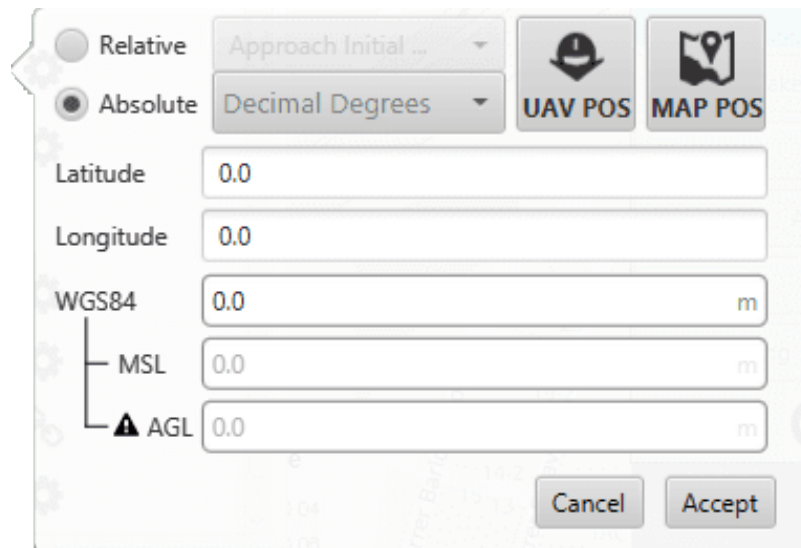
and the user's indicated parameters. Below, the parameters shown above are going to be described. Later, a brief description of the algorithm and its behavior in different possible situations will be presented:

1. **Section.** This option allows the user to select the first path to be flown by the aircraft in cruise mode. Section selection is disabled in this mode. As already said, in this phase the path can not be directly indicated by the user as it is done in cruise mode (see [Mission menu](#) for more details).
2. **Set height mode.** Height mode indicates how the aircraft will perform the route. Height mode in Climb guidance is fixed to 2.5D type. In 2.5D, the aircraft will fly from current altitude at a desired flight path angle, FPA. Go below for more information about this topic.
3. **Arcade position/speed transition** In Arcade mode the trajectory generated (*position*) is not followed and instead the aircraft moves according to the commanded *speed*. The *horizontal* and *vertical speed* parameters serve as the upper-thresholds for when the aircraft's guidance should be *position-based*, even in Arcade mode. This parameters are mainly useful for platforms like multicopters.
4. **Set speed:** this option sets the desired aircraft speed during the manoeuvre.
 - **Cruise:** Cruise Lets the user set the velocity modulus of the guidance. This velocity can be slightly modified by the autopilot control algorithms.
 - **Waypoint reach** indicates the speed at which the platform will reach the waypoints of the path, i.e. it will travel along the path with the speed indicated in the option *Cruise* and then it will decelerate or accelerate to the speed indicated.
 - **Defined speed** can be IAS (indicated airspeed) or Ground speed. Normally, IAS is used for airplanes and Ground speed for multicopters.
 - **Deceleration:** maximum allowed acceleration/deceleration to meet the desired velocity.
5. **Guidance control:** Horizontal and Vertical PIDs controllers to guarantee stability of guidance loop are defined here. Their purpose are to adjust the platform's actual trajectory to the desired one. When clicking on *Horizontal* or *Vertical*, a pop-up window will appear where the control parameters should be introduced – for more information on the latter subject check [Blocks](#).
6. **Loiter and runway position:** By default, loiter and runway positions and direction are defined in [Runway](#) menu and the user selects the runway. However, this parameters can be modified clicking on “Advanced”. Then, the user can choose between the two available option definitions by clicking on :



Runway and loiter position options and alternative configurations

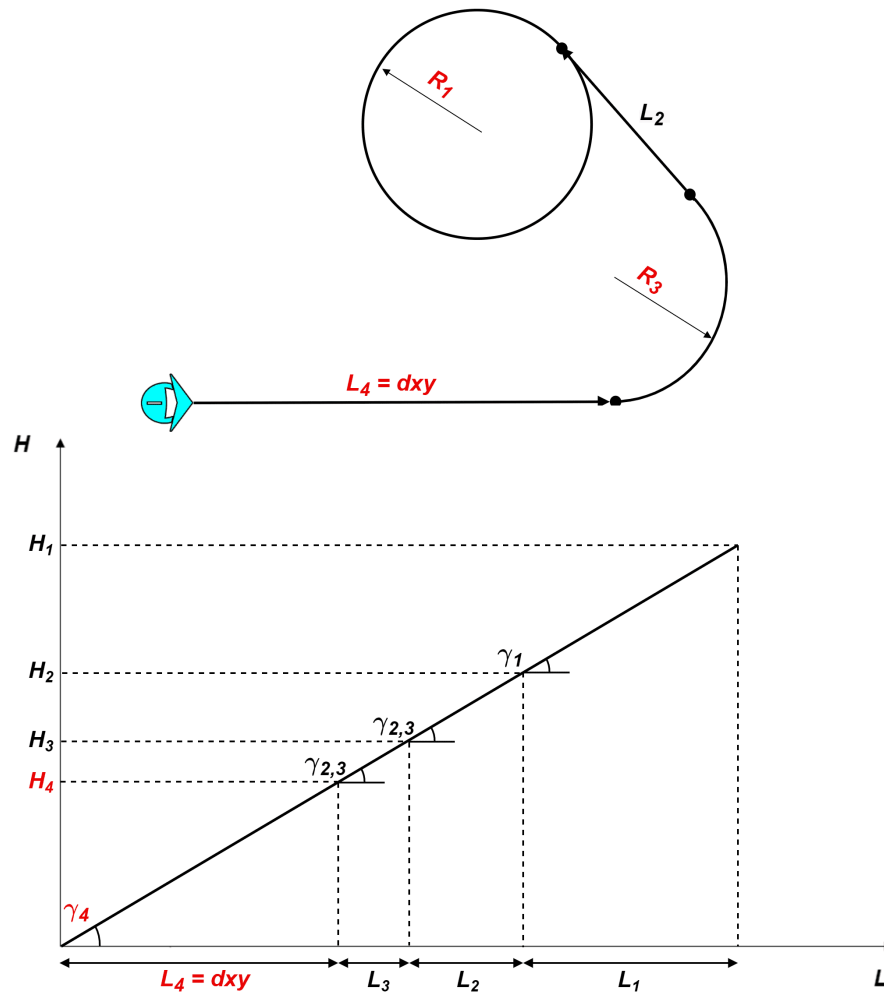
- **Loiter point:** defines the loiter point. By default, this point is the runaway's loiter. But the user can select in the drop-down list any other previously defined point. This includes waypoints defined in Mission, among many others. Alternately, the user can manually define the Loiter point.



The dialog box for selecting a loiter point. It features two radio buttons at the top: 'Relative' (unselected) and 'Absolute' (selected). To the right of these is a dropdown menu currently showing 'Approach Initial ...'. Further right are two buttons: 'UAV POS' with a drone icon and 'MAP POS' with a map icon. Below the radio buttons, there is another dropdown menu showing 'Decimal Degrees'. The main section contains five input fields, each with '0.0' and a unit 'm' on the right: 'Latitude', 'Longitude', 'WGS84', 'MSL', and 'AGL' (which has a warning triangle icon to its left). At the bottom right are 'Cancel' and 'Accept' buttons.

Loiter Point Selection Menu

- **Runway direction:** defines the runway direction. Again, by default, is the same as the selected runway. It can be also chosen from a list of options including runway direction, tailwind direction, etc. Alternately, it can also be defined as an angle with respect to the magnetic north.
- If the box **Loiter pos is center** is ticked, the defined loiter point will be the center of the loiter circular trajectory. In case of not, the circular loiter trajectory will pass through that point.




Climb route top and front views with parameter identification

7. **Route:** here is where the user can set some of the the climbing path parameters (those highlighted in red on the above diagram). First, the user defined parameters are described and, after, some considerations about the climb algorithm behavior will be explained:

- **Taxi extension:** this parameter does not apply to this algorithm.
- **Horizontal extension (dxy):** absolute ground distance of the first path, L_4 : from the star of the climb to the start of the turn that faces the loiter path. This distance will remain fix always and it will also fix L_4 path's final point height, H_4 . More information below.
- **Radius Head Turn (R3):** radius of the turn to head the platform towards the loiter.
- **Radius loiter (R1):** radius of the ascending helix path to reach the loiter height.
- **Flight Path Angle.** The *FPA* is the angle at which the aircraft will climb. Before the algorithm execution, all Flight Path Angles, γ_i , are equal: $FPA = \gamma_4 = \gamma_{2,3} = \gamma_1$. The algorithm can modify $\gamma_{2,3}$ and γ_1 . In that case, the Flight Path Angle option will serve as the upper threshold.

Each one of this parameters can be disabled, enabled or linked to an Operation Guidance defined by

the user clicking on .

7.2.4.1.1.6 Climbing guidance parameters behavior


The climbing track is not fix, the algorithm recalculates the paths each time to take into account the aircraft position and the user's parameters. The trajectory usually has 4 paths, excluding the final loiter path:

- **General trajectory description:**

- L_4 is the first path. The user can set the horizontal length, dxy , the direction (in Runway direction) and the path's final point height, H_4 with the defined Flight Path Angle. This is very relevant, as seen later. The path length can not be zero.

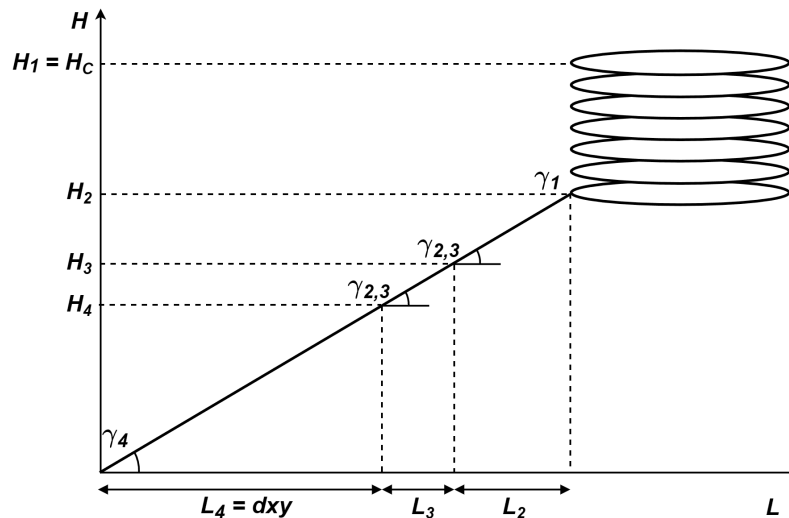
$$\gamma_4 = FPA$$

$$H_4 = dxy * \tan(\gamma_4)$$

- L_3 : Circular turn to head the platform towards the loiter. The user can set the radius, R_3 . Is possible to set this path to zero and disabled it clicking on . The FPA, $\gamma_{2,3}$, can be modified by the algorithm.
- L_2 : Straight path that reaches the climbing loiter point. This path is completely automatic generated. Its FPA, $\gamma_{2,3}$, can be modified by the algorithm.
- L_1 : Ascending helix path to reach the loiter height. The user can set the radius, R_1 .

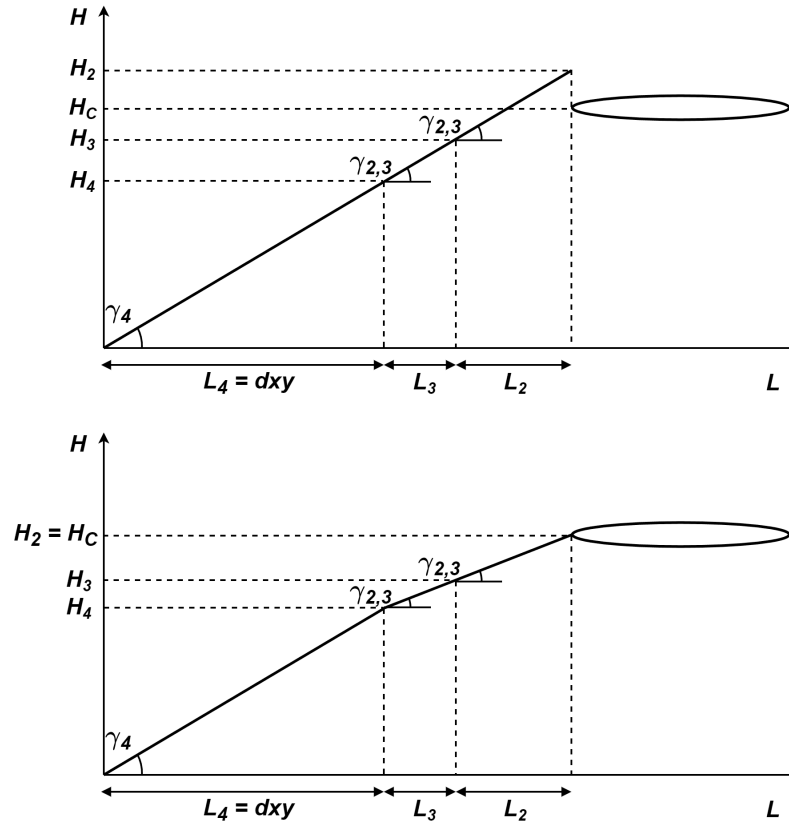
- **Loiter height effect:** Loiter height's, H_c , modifies the algorithm general behavior. Depending on whether H_c is bigger or smaller than H_2 or smaller than H_4 the algorithm will modify some parameters, in particular the flight path angles:

1. $H_2 < H_c$. Is the general case. In this situation, no corrections will be applied as shown below and $\gamma_{2,3} = \gamma_1 = FPA$.



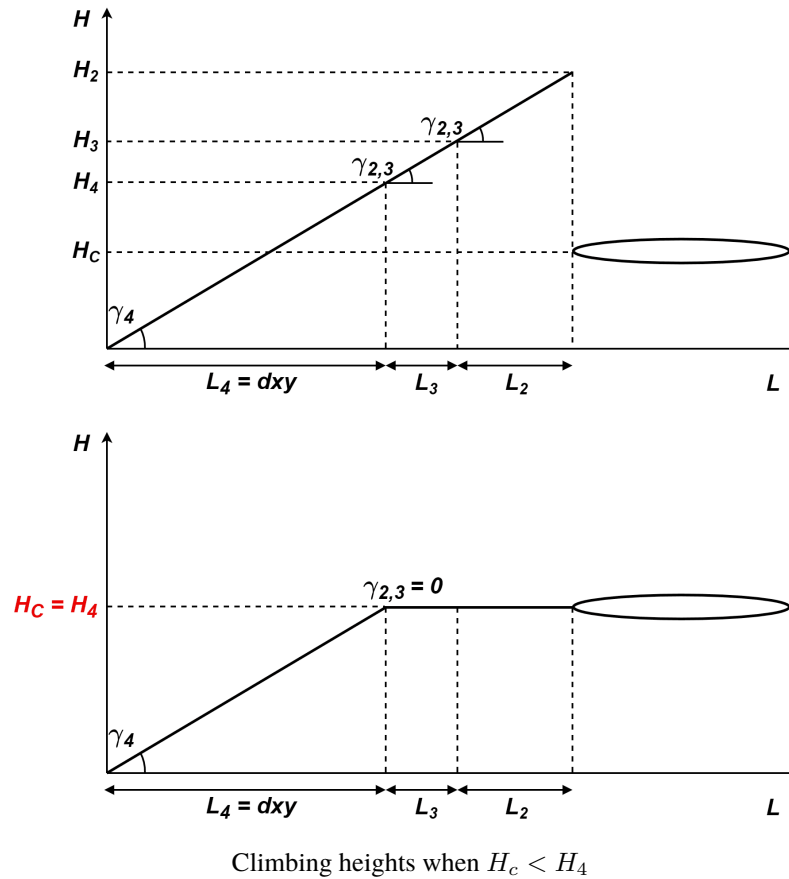
Climbing heights when $H_2 < H_c$

2. $H_4 < H_c < H_2$: In this case, the algorithm will compute a new $\gamma_{2,3}$ to avoid surpassing the loiter's height and γ_1 will be zero.



Climbing heights when $H_4 < H_c < H_2$

3. $H_c < H_4$: In this case, the algorithm will force $H_c = H_4$. So H_4 **will be the new loiter height** keeping $\gamma_4 = FPA$ and the other flight paths angles equal to zero, $\gamma_{2,3} = \gamma_1 = 0$.



7.2.4.1.1.7 Guidance-generated Variables

Climbing guidance generates a three-dimensional trajectory, as well as a set of variables which can be used in the control loops.

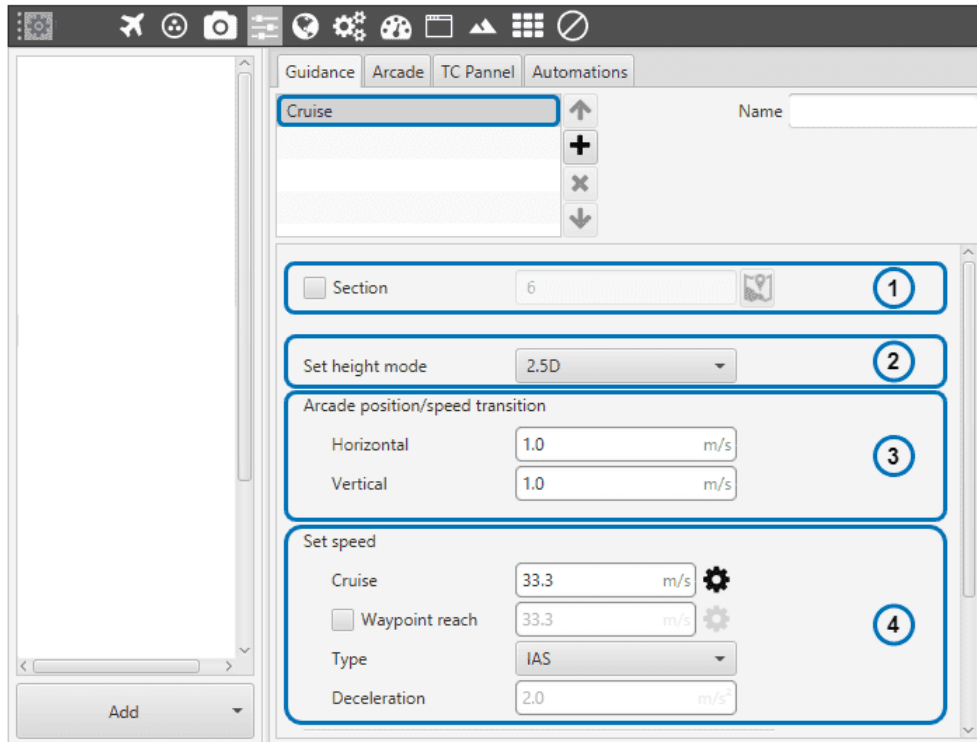
The list of variables is the following:

- Desired position.
- Track position.
- Track state (current patch, last patch).
- Desired latitude, desired longitude, desired WGS84, desired MSL, desired AGL.
- Desired velocity.
- Desired front groundspeed, desired lateral groundspeed, desired velocity down.
- Desired tangential acceleration.
- Desired IAS.
- Guidance north error.
- Guidance east error.
- Guidance down error.
- Desired body velocities.


- Desired velocities north, east, down.
- Desired heading, FPA and bank.
- Route-guidance distance - tangential component.
- Route-guidance distance - horizontal component.
- Route-guidance distance - perpendicular component.

7.2.4.1.1.8 Cruise

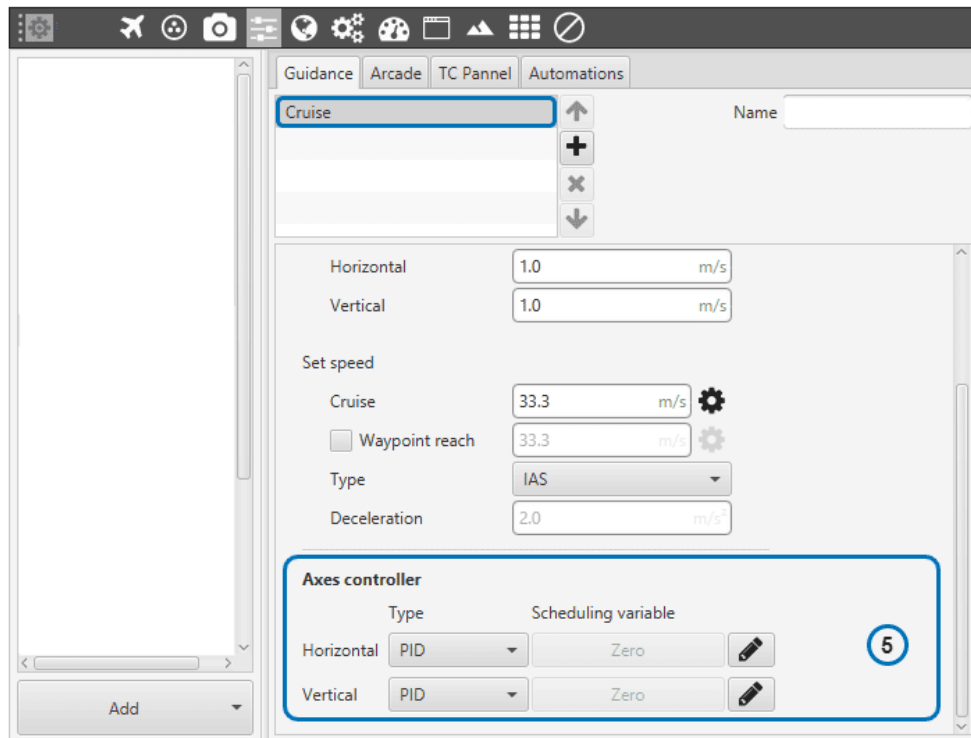
This phase is used to make the aircraft follow a position-based route created by the user in the Mission menu. This is the principal use of this guidance algorithm, but it can also be used to make the aircraft go to a certain location (e.g waypoint) without indicating the full route, thus being a guidance used to command a movement by position. All the parameters that define the cruise guidance are detailed.



Cruise Guidance Menu (I)

1. **Section.** This option allows the user to select the first path to be flown to. The user should first enable this option, click on  and then click on the desired waypoint to be the first-of-the-route.
2. **Set height mode.** Height mode indicates how the aircraft will perform the route when it comes to the waypoints' altitude. There are three possible height modes:
 - **2D mode:** if this mode is selected, the platform will follow the predefined route without taking into account the altitude of the waypoints, it will keep the altitude that it has at the moment it enters in the cruise guidance.

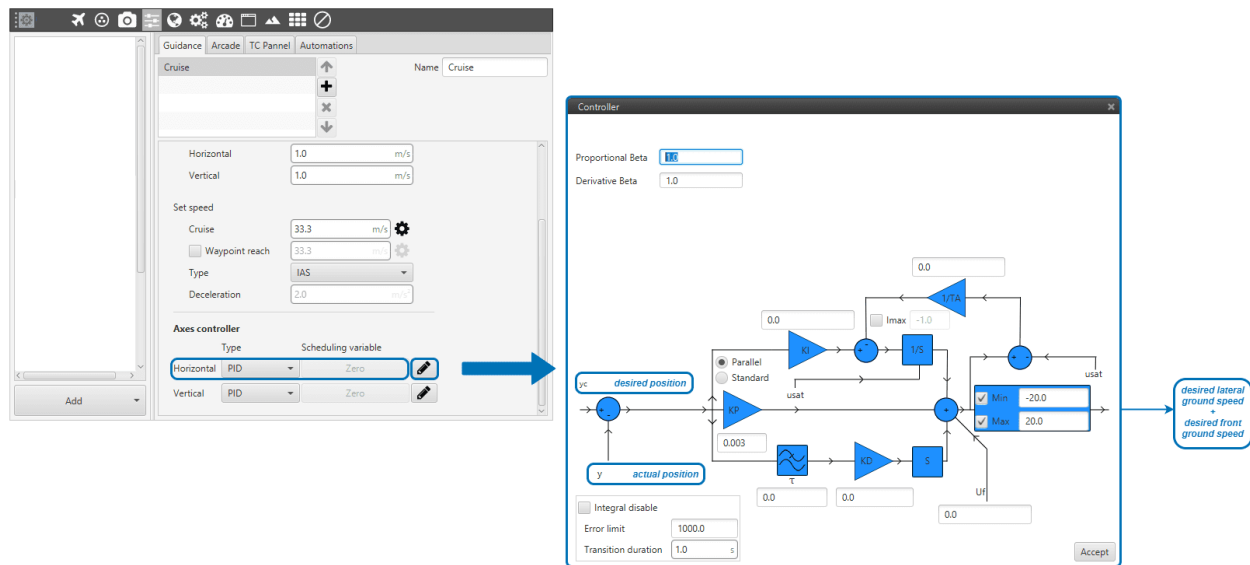
- **2.5D mode:** the vehicle will follow a 3D trajectory that connects both waypoints. However, it will give priority to horizontal guidance. Veronte will try to adjust its position and altitude to the path (both horizontally and vertically), but if for any reason it cannot reach the altitude of the final waypoint, it will be considered that it has been reached if its position matches the position of the waypoint.
 - **3D mode:** the vehicle will follow a 3D trajectory that connects both waypoints. In this case, horizontal and vertical guidance have the same priority level. This means that Veronte will not consider that a waypoint has been reached until its position and altitude match the waypoint's ones. As this type of guidance may result in a vertical flight, it is reserved for multicopters or hybrid platforms.
3. **Arcade position/speed transition** In Arcade mode the trajectory generated (*position*) is not followed and instead the aircraft moves according to the commanded *speed*. The *horizontal* and *vertical speed* parameters serve as the upper-thresholds for when the aircraft's guidance should be *position-based*, even in Arcade mode. This parameters are mainly useful for platforms like multicopters.
4. **Set speed: this option sets the desired aircraft speed during the manoeuvre.**
- **Cruise:** sets the velocity modulus of the commanded velocity.
 - **Waypoint reach** indicates the speed at which the platform will reach the waypoints of the path, i.e. it will travel along the path with the speed indicated in the option *Cruise* and then it will decelerate or accelerate to the speed indicated in here.
 - **Defined speed type** can be IAS (indicated airspeed) or Ground speed. Normally, IAS is used for airplanes and Ground speed for multicopters.
 - **Deceleration:** maximum allowed acceleration/deceleration to meet the desired velocity.



Cruise Guidance Menu (II)

5. **Guidance control:** a horizontal and vertical position PID controller is defined here. Its purpose is to adjust the platform's actual trajectory to the desired one. When clicking on *Horizontal* or *Vertical*, a pop-up window will open where the control gains should be introduced – for more information on the latter subject check the [Blocks](#). In the horizontal-position PID (see image below), North-East current position of the aircraft is compared

to the desired position. The output of the PID controller is going to be a ground speed in the North-East plane, which translates into a desired lateral and front ground speeds in body axes. The same logic applies for the vertical-position PID.



Horizontal Position PID

7.2.4.1.1.9 Guidance-generated Variables

Cruise guidance generates a three-dimensional trajectory, as well as a set of variables which can be used in the control loops.

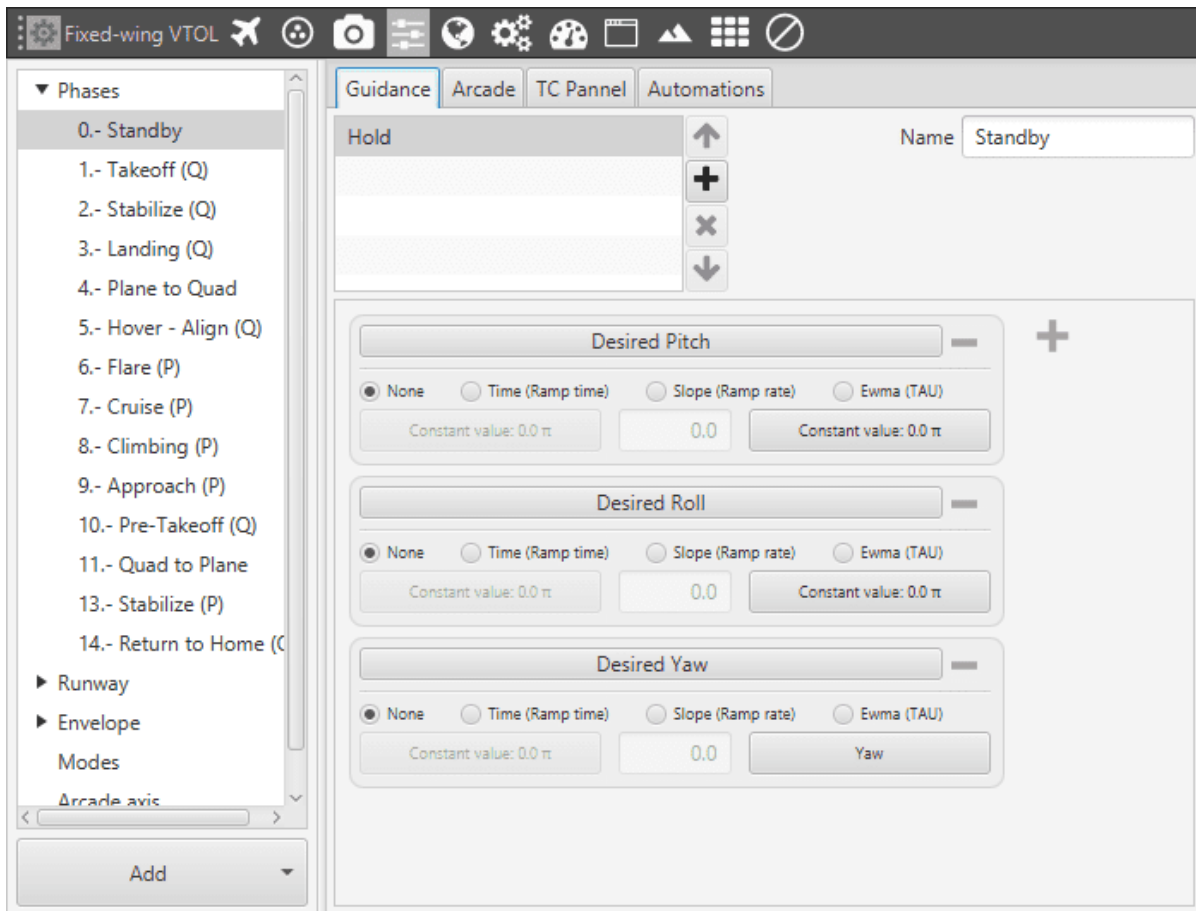
The list of variables is the following:

- Desired position.
- Track position.
- Track state (current patch, last patch).
- Desired latitude, desired longitude, desired WGS84, desired MSL, desired AGL.
- Desired velocity.
- Desired front groundspeed, desired lateral groundspeed, desired velocity down.
- Desired tangential acceleration.
- Desired IAS.
- Guidance north error.
- Guidance east error.
- Guidance down error.
- Desired body velocities.
- Desired velocities north, east, down.
- Desired heading, FPA and bank.
- Route-guidance distance - tangential component.

- Route-guidance distance - horizontal component.
- Route-guidance distance - perpendicular component.

7.2.4.1.1.10 Hold

Hold guidance algorithm allows the user to introduce a desired variable value that has to be followed or held by the platform. It differs from the other guidances where the user sets a path to follow and the aircraft follows the track commanding the desired heading, velocities, etcétera, which is determined by the control algorithm. Here, the aircraft will hold a velocity (IAS,TAS,...), a heading, a flight path angle, a roll, pitch and yaw angle, ... or a constant distance to a moving Object of Interest, for example.



Hold Guidance Example in Standby phase

There are 4 ways in Pipe to determine how the variable is introduced in the controller in a Hold Guidance:

- **None.** A simple value is specified. As soon as the aircraft enters in the phase that has this hold guidance, the desired variable will take the value specified in this option. It is possible to define a constant value for the desired variable, or take the value that another variable has at the instant when the aircraft enters in that phase. Lets see some examples:
 - To make the aircraft keep the heading that it has when changing phase, select Desired Heading as the hold variable, and choose heading. This is useful, for example, when the aircraft loses GNSS signal. You can set an automation that jumps to this phase to hold the flight path.

Hold Heading

- Hold is also useful in the **Standby phase** to check the correct setting of the servos of the aircraft. During this phase, if this is configured, the user can manually move the aircraft and the autopilot will try to counteract this change of attitude commanding movement to the corresponding servos:

Standby phase Hold guidance

- Another common application of this Hold guidance is the *Arcade Mode* for multicopters. Multicopters need assistance to fly as they are inherently unstable. Here are presented two strategies: hold attitude or hold position.
 - * **Hold attitude** will correct the angles of the multicopter to keep it stable when the stick is released, but the aircraft will continue moving due to inertia.

Hold multicopter attitude

- * To **hold aircraft's position**, a good strategy is to set a Velocity Hold (or VHold). This adds another layer to the control algorithms. Setting velocities to zero will stop the vehicle in place when the stick is released.

The image shows a configuration window for the Veronte Autopilot. It contains three vertically stacked sections, each for a different type of Ground Velocity (GV):

- Desired Down GV (Ground Velocity)**: The 'None' radio button is selected. The central value field is set to 0.0.
- Desired Lateral GV (Ground Velocity)**: The 'None' radio button is selected. The central value field is set to 0.0.
- Desired Front GV (Ground Velocity)**: The 'None' radio button is selected. The central value field is set to 0.0.

Each section also includes two 'Constant value: 0.0 m/s' labels on either side of the central value field.

Hold multicopter position

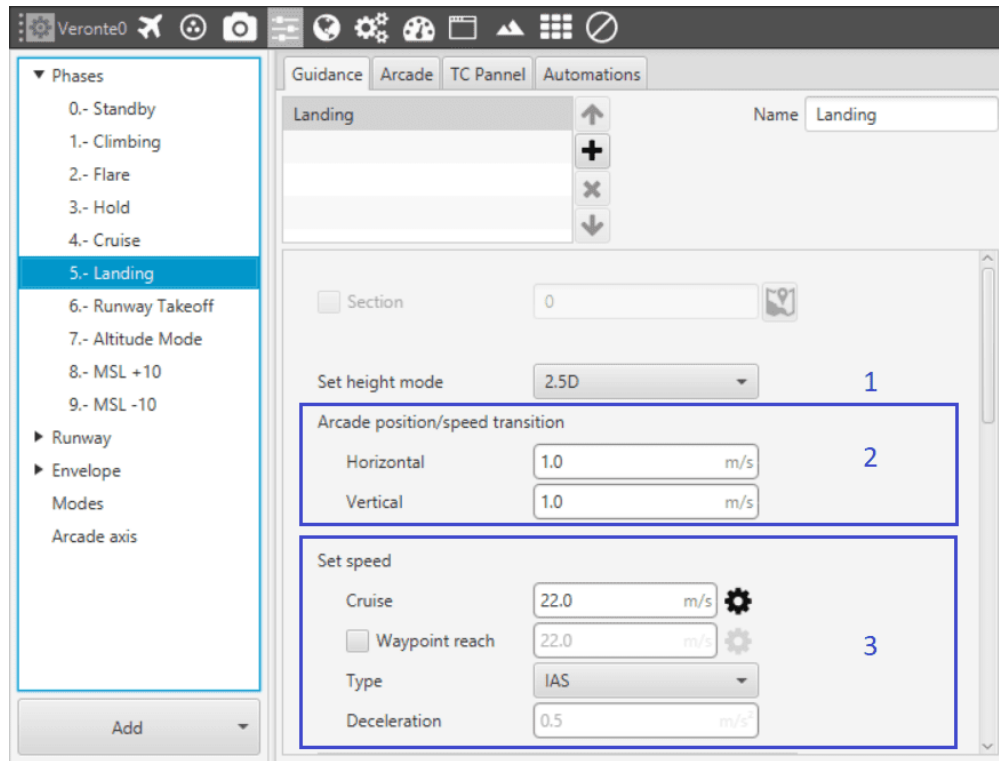
..note: This Hold guidances examples lack of Yaw control as it is included in the phase with other guidance control.

- **Time (Ramp Time)**. In this case the desired variable introduced in the controller will take first the value (or the variable) introduced. After a certain time established, the variable will reach the value indicated.
- **Slope (Ramp Rate)**. The same as before, but now instead of the time between the two values of the ramp, the slope of the linear ramp can be indicated.
- **Ewma (TAU)**. The variation between the start and final values is exponential.

These type of transitions are useful to avoid sending aggressive control commands to the aircraft that can damage the structure or set the aircraft in a compromising situation. For example, this can be used in the flare maneuver before the aircraft's touchdown during landing. The aircraft, which is flying at a very low speed height must reduce its vertical down velocity. A time ramp can be set to reduce it from the actual velocity at which the plane enters this phase to the desired one.

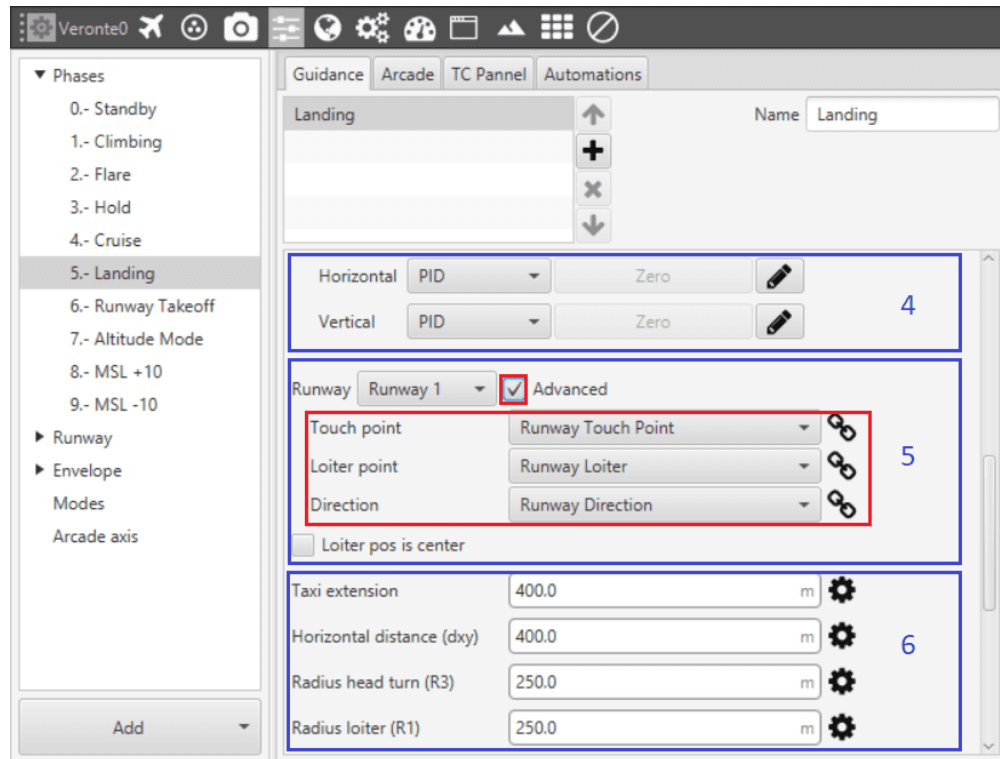
7.2.4.1.1.11 Landing

Landing guidance is used to generate the flying path the aircraft will follow when landing on a certain runway. The generated path is not directly indicated by the user as in the cruise guidance (which is defined in the [Mission menu](#)), instead a trajectory is generated based on the parameters detailed later in this section. Below, find all the information to be defined by the user accompanied by the corresponding Figures showing the location of these parameters on the menu.



Landing Guidance menu (I)

1. **Set height mode:** the height mode indicates how the aircraft will perform the landing path. The only one available for landing is 2.5D mode: the vehicle goes from the altitude at which it enters the phase with landing guidance to the beginning of the route in a diagonal trajectory (it follows a 3D trajectory that connects the two points) with a flight path angle, giving priority to horizontal over vertical guidance.
2. **Arcade position/speed transition:** if Arcade mode is configured, then the trajectory generated (*position*) is not followed and instead the aircraft moves according to the commanded *speed*. The parameters here (*horizontal* and *vertical speed*) serve as an upper-threshold for when the aircraft's guidance should be *position-based*, even in Arcade mode. These parameters are mainly useful for platforms like multicopters.
3. **Set speed:** this option sets the speed that the vehicle will have during the landing manoeuvre.
 - In *Cruise* the user sets the velocity modulus.
 - The option *Waypoint reach* is used to indicate the speed at which the platform will reach the waypoints, i.e. it will travel along the path with the speed indicated in the option *Cruise* and then it will decelerate or accelerate to the speed indicated in *Waypoint reach*.
 - The defined speed can be IAS (indicated airspeed) or Ground speed. Normally, IAS is used for airplanes and Ground speed for multicopters.
 - *Deceleration*: maximum allowed acceleration/deceleration in order to meet the desired velocity.



Landing Guidance menu (II)

4. **Guidance control:** a horizontal and vertical PID controller is defined here. Its purpose is to adjust the platform's actual trajectory to the desired one. When clicking on *Horizontal* or *Vertical*, a pop-up window will open where the control parameters should be introduced – for more information on the latter subject check the [Blocks](#).
5. **Loiter and runway position:** here the user defines the loiter and runway positions. The default option is to have the loiter and runway positions defined in [Runway](#). If the *Advanced* option is chosen, then the user needs

to define three parameters. By clicking on  different options will be displayed:

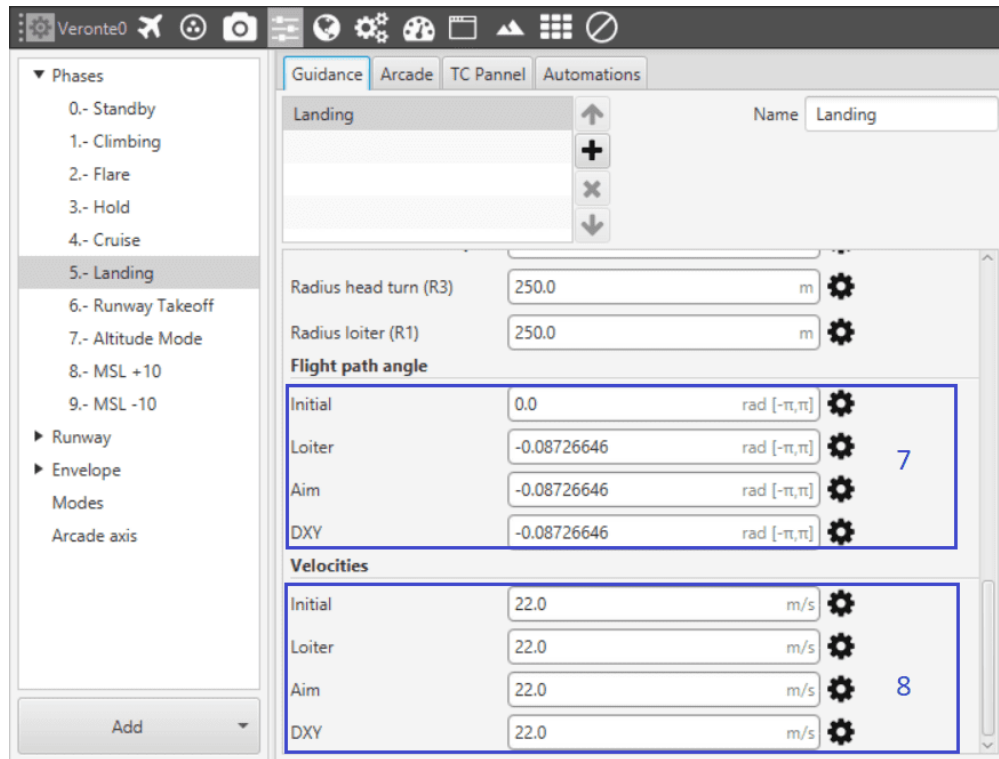
- *Touchpoint:* defines the touchpoint of the runway. This point can be defined by introducing its latitude, longitude and altitude; or else it can be chosen from a list of options that includes the waypoints defined in Mission, among many others.
- *Loiter point:* defines the loiter point. This point can be defined by introducing its latitude, longitude and altitude; or else it can be chosen from a list of options that includes the waypoints defined in Mission, among many others.
- *Runway direction:* defines the runway direction. It can be defined as an angle with respect to the magnetic north; or else it can be chosen from a list of options like runway direction, tailwind direction, etc.

If the box *Loiter pos is center* is ticked the defined loiter point will be the center of the loiter circular trajectory. On the contrary, the loiter trajectory can pass through that point.

6. **Trajectory distances:** here the user defines some of the trajectory distances. These distances match the trajectory patches lengths L or are proportional to them. See the explanation below for more information on every patch.
 - *Radius loiter R1:* radius of the descending loiter for the aircraft to reach an altitude suitable to perform the landing manoeuvre ($L_1 \propto \pi R_1$).
 - *Radius Head Turn R3:* radius of the last turn in order to face the runway direction ($L_3 \propto \pi R_3$).

- *Horizontal extension (dxy)*: distance before the head of the runway. At the end of this length, touchdown is expected.
- *Taxi extension*: distance from touchdown to where the aircraft is brought to a full stop.

Some patches don't have an associated user-defined distance, and are automatically calculated by the landing guidance algorithm: they depend on some of the above distances and other parameters defined below.



Landing Guidance menu (III)

7. **Trajectory flight path angles**: here the user defines the desired trajectory flight path angles for each of the patches of the trajectory. See the explanation below for more information on every patch.
 - *Initial*: desired flight path angle γ_0 of patch 0.
 - *Loiter*: desired flight path angle γ_1 of patch 1.
 - *Aim*: desired flight path angle $\gamma_{2,3}$ of patches 2 and 3.
 - *DXY*: desired flight path angle γ_4 of patch 4.
8. **Trajectory velocities**: here the user defines the desired trajectory velocities for each of the patches of the trajectory. See the explanation below for more information on every patch.
 - *Initial*: desired velocity v_0 of patch 0.
 - *Loiter*: desired velocity v_1 of patch 1.
 - *Aim*: desired velocity $v_{2,3}$ of patches 2 and 3.
 - *DXY*: desired velocity v_4 of patch 4.

The generated trajectory of the landing guidance defines the route that the aircraft follows from the point when the phase with this guidance is entered, to the point where it touches the ground – see the Figure below. The landing route has two parts, being decomposed into 6 patches.

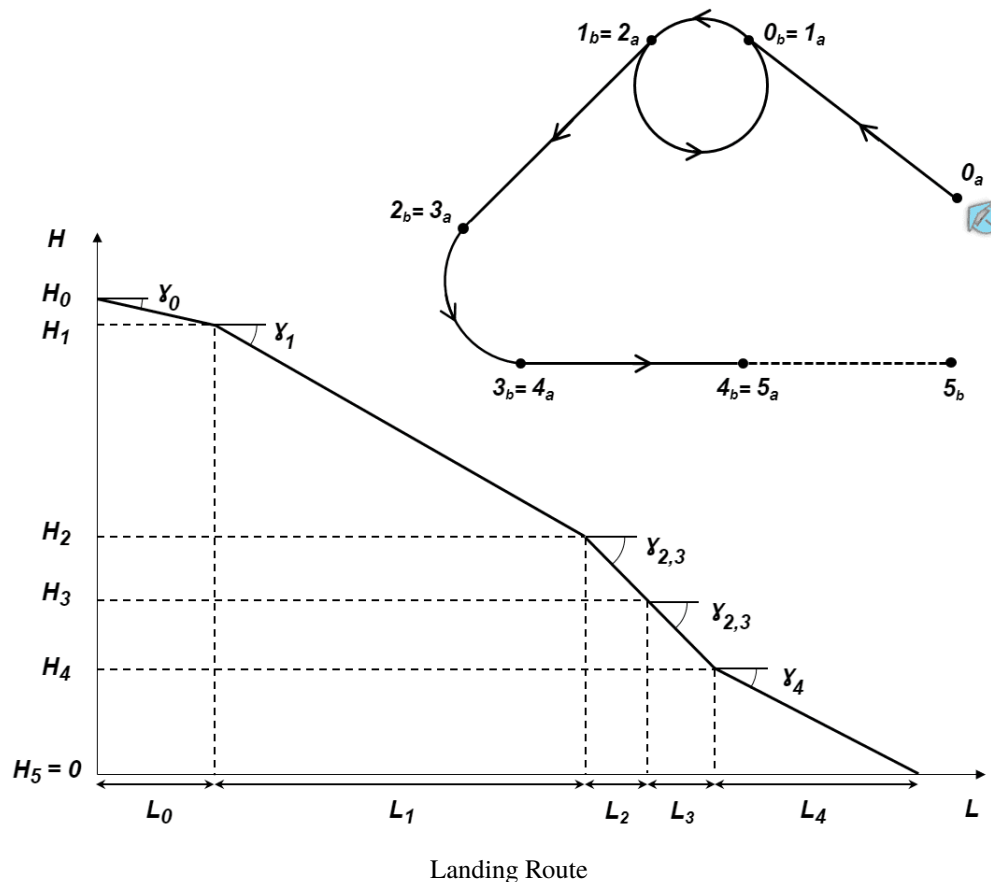
- **First part: descending loiter used to descend from the cruise altitude to an altitude where the heading manoeuvre towards**

- Patch 0: this patch is generated from the point the landing phase is entered to where the loiter is located. Variables that influence this patch are γ_0 , v_0 , altitudes H_0 and H_1 , and *Loiter point* position.
- Patch 1: the patch length (L_1) will depend on the amount of loops on the loiter. The latter can go from 0 to more than 1 loop, depending on the altitude necessary to descend/ascend. Variables that influence this patch are γ_1 , v_1 , altitudes H_1 and H_2 , and *Radius loiter* $R1$.

The loiter exiting point altitude is computed so that patches 2 to 5 can be performed following their desired v and γ . So it exists the possibility of starting the landing manoeuvre at a lower altitude than the exiting point of the loiter. In that case, the loiter would be used to ascend. If the aircraft starts the landing phase at an altitude similar to the one of the loiter (defined in point 6), then the loiter patch is simplified into a turn – during the turn the altitude can still be adjusted, and the turn's length will depend on the latter.

- **Second part: final approach of the landing, which consists on turning, facing the runway and touchdown.**

- Patch 2: patches 3 and 4 need to match the distances defined above. Patch number 2 will connect the exit of the loiter patch with the beginning of patch 3. Variables that influence this patch are $\gamma_{2,3}$, $v_{2,3}$, altitudes H_2 and H_3 , and *Loiter point* and *Touchpoint* positions.
- Patch 3: turning of the aircraft to face the runway. Variables that influence this patch are $\gamma_{2,3}$, $v_{2,3}$, altitudes H_3 and H_4 , and *Radius Head Turn* $R3$.
- Patch 4: at the end of the patch the aircraft lands. Variables that influence this patch are γ_4 , v_4 , altitude H_4 , and *Horizontal extension* (d_{xy}).
- Patch 5: taxi extension for the aircraft to slow down.



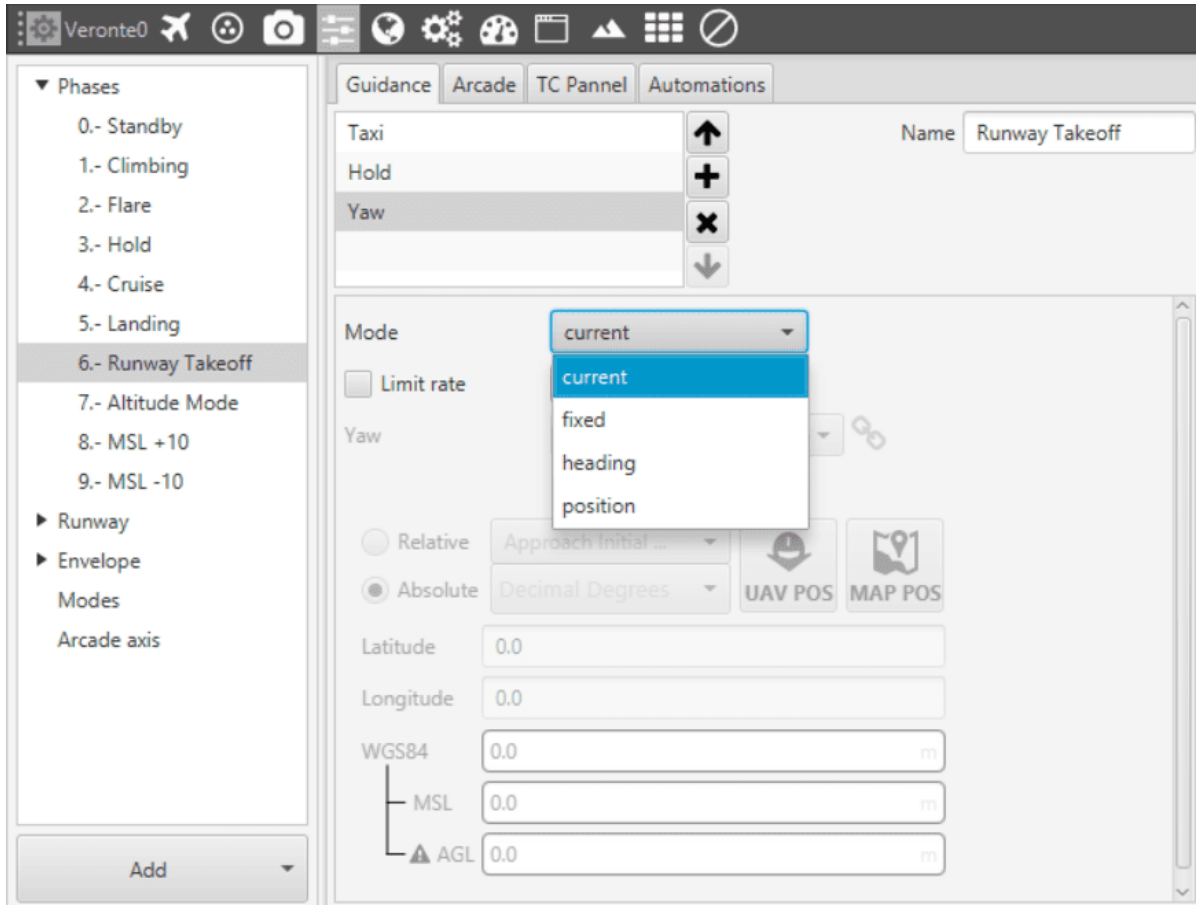
7.2.4.1.1.12 Guidance-generated Variables

Landing guidance generates a three-dimensional trajectory, as well as a set of variables which can be used in the control guidance. The list of variables is the following:

- Desired position.
- Track position.
- Track state (current patch, last patch).
- Desired latitude, desired longitude, desired WGS84, desired MSL, desired AGL.
- Desired velocity.
- Desired front groundspeed, desired lateral groundspeed, desired velocity down.
- Desired tangential acceleration.
- Desired IAS.
- Guidance north error.
- Guidance east error.
- Guidance down error.
- Desired body velocities.
- Desired velocities north, east, down.
- Desired heading, FPA and bank.
- Route-guidance distance - tangential component.
- Route-guidance distance - horizontal component.
- Route-guidance distance - perpendicular component.

7.2.4.1.1.13 Yaw

Yaw guidance is used in multicopters to indicate the behaviour of the platform in the yaw axis. This option is normally used during the cruise phase of the multicopters, because the route can be carried out with the aircraft without rotating in the yaw axis, or rotate it to point its longitudinal axis parallel to the path.



Yaw Guidance

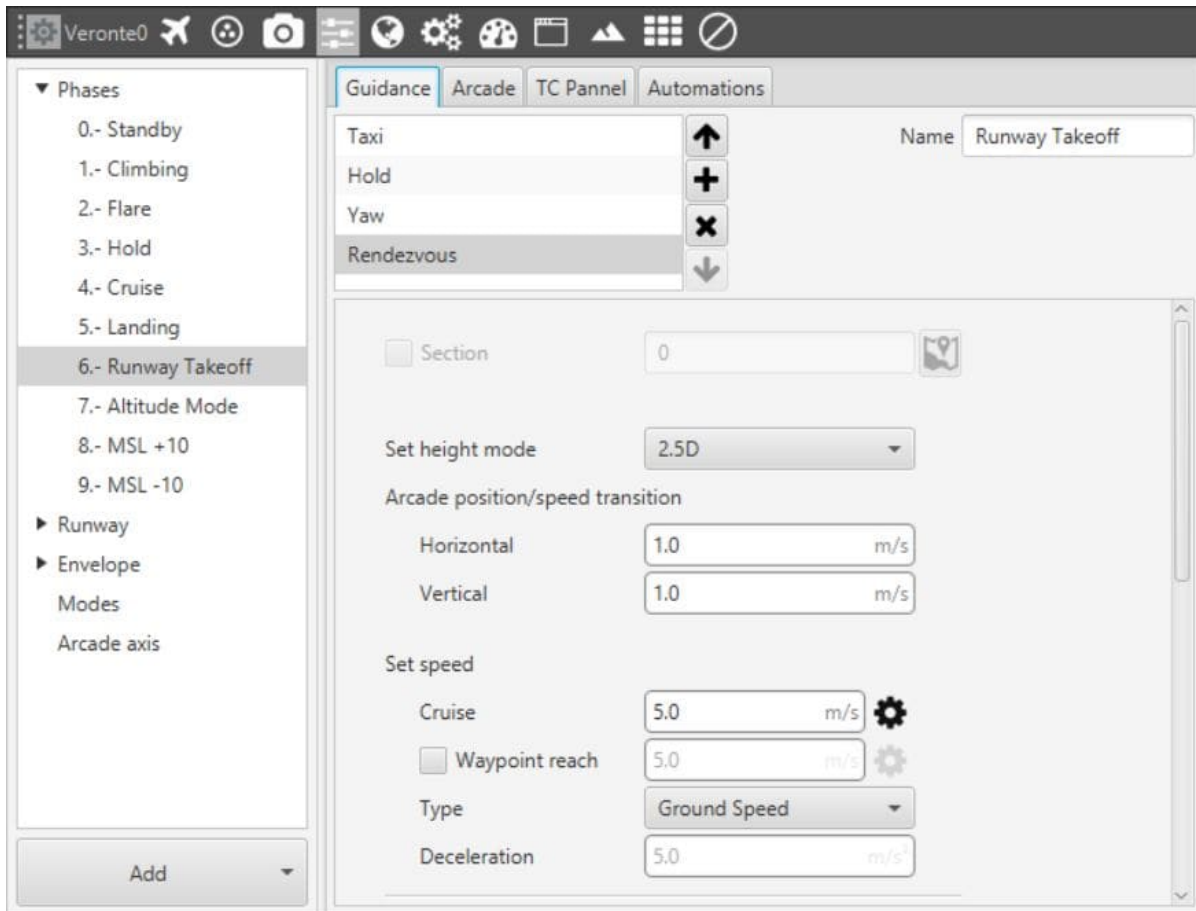
The modes available in the yaw guidance are:

- **Current:** the multicopter will keep the yaw angle it has when entering in the phase that contains this guidance. Desired Yaw = Current Yaw
- **Fixed:** the yaw is kept at a constant value indicated by the user.
- **Heading:** Heading represents velocity vector direction and, when it is very small, its estimation is more complex and direction is changing constantly. Because of this, the approximation $\text{Yaw} = \text{Heading}$ is introduced when the estimated velocity is near 0.
- **Position:** the yaw of the multicopter will be rotated so its longitudinal axis is always focusing a point defined by the user in the menu shown in the previous figure (absolute or relative).

It's possible to fix a limit in the yaw rate and a constant value (depending on the selected mode).

7.2.4.1.1.14 Rendezvous

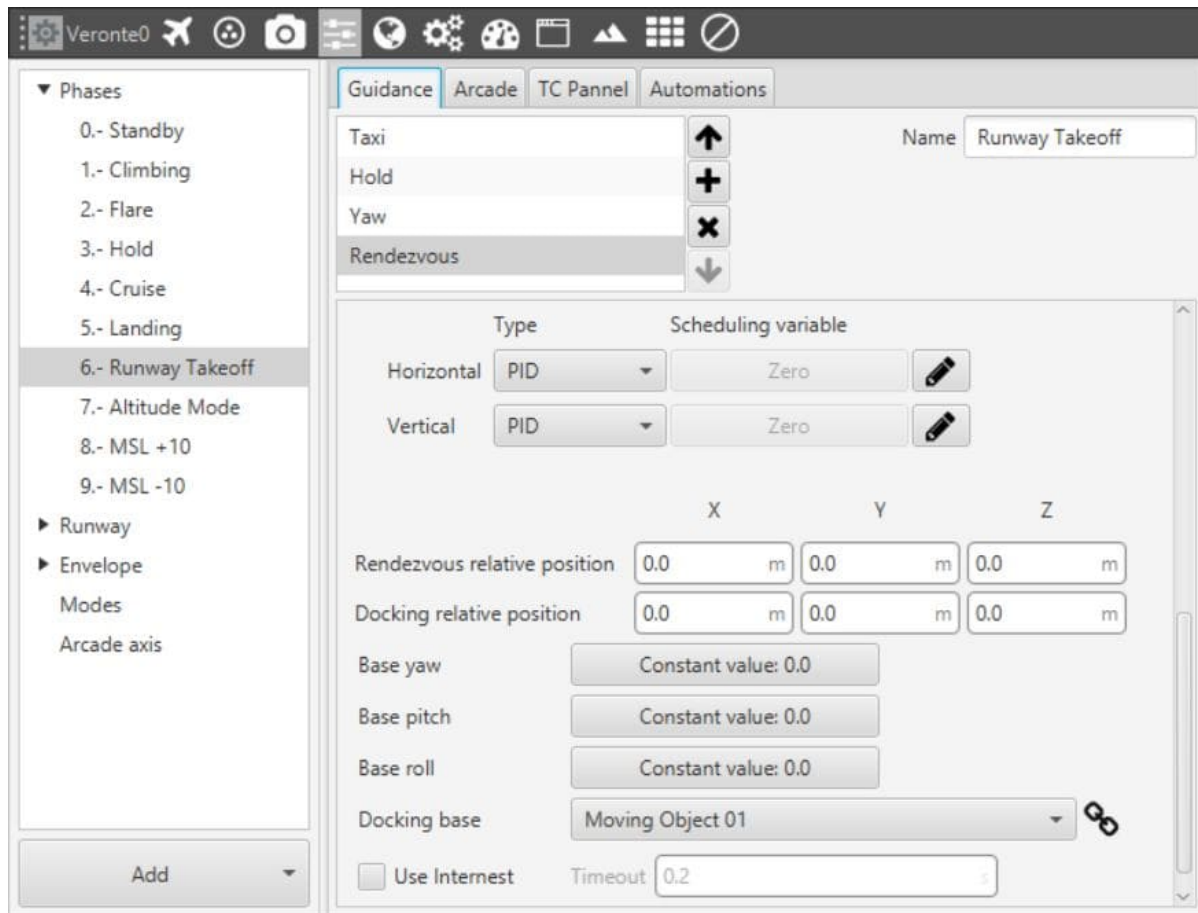
Rendezvous guidance is used to create a meeting point where the Air unit will approach a second unit (either Air or Ground) within a determined offset. This guidance updates constantly the vehicle attitude in order to track, with the shortest path, the position of the second unit (named as Ground hereafter). This guidance works for both static and moving Ground. Rendezvous navigation is ready for taking Internet input to improve the precision from its guidance, being the most suitable kind for Internet integration.



Rendezvous Guidance – Menu 1

- **Set height mode:** the height mode indicates how the aircraft will perform the route. 2D height mode is not allowed in this mode.
 - **2.5D mode:** the vehicle will follow a 3D trajectory that connects both waypoints. However, it will give priority to horizontal guidance. Veronte will try to adjust its position and altitude to the path (both horizontally and vertically), but if for any reason it cannot reach the altitude of the final waypoint, it will be considered that it has been reached if its position matches the position of the waypoint.
 - **3D mode:** the vehicle will follow a 3D trajectory that connects both waypoints. In this case, horizontal and vertical guidance have the same priority level. This means that Veronte will not consider that a waypoint has been reached until its position and altitude match the waypoint's ones. As this type of guidance may result in a vertical flight, it is reserved for multicopters or hybrid platforms.
- **Set speed:** this option sets the speed that the vehicle will have during the phase. It could be IAS (indicated airspeed) or Speed (Ground Speed). Normally, the IAS is used for airplanes and the Speed for multicopters. The option Waypoint reach is used to indicate the speed at which the platform will reach the waypoints, so it will

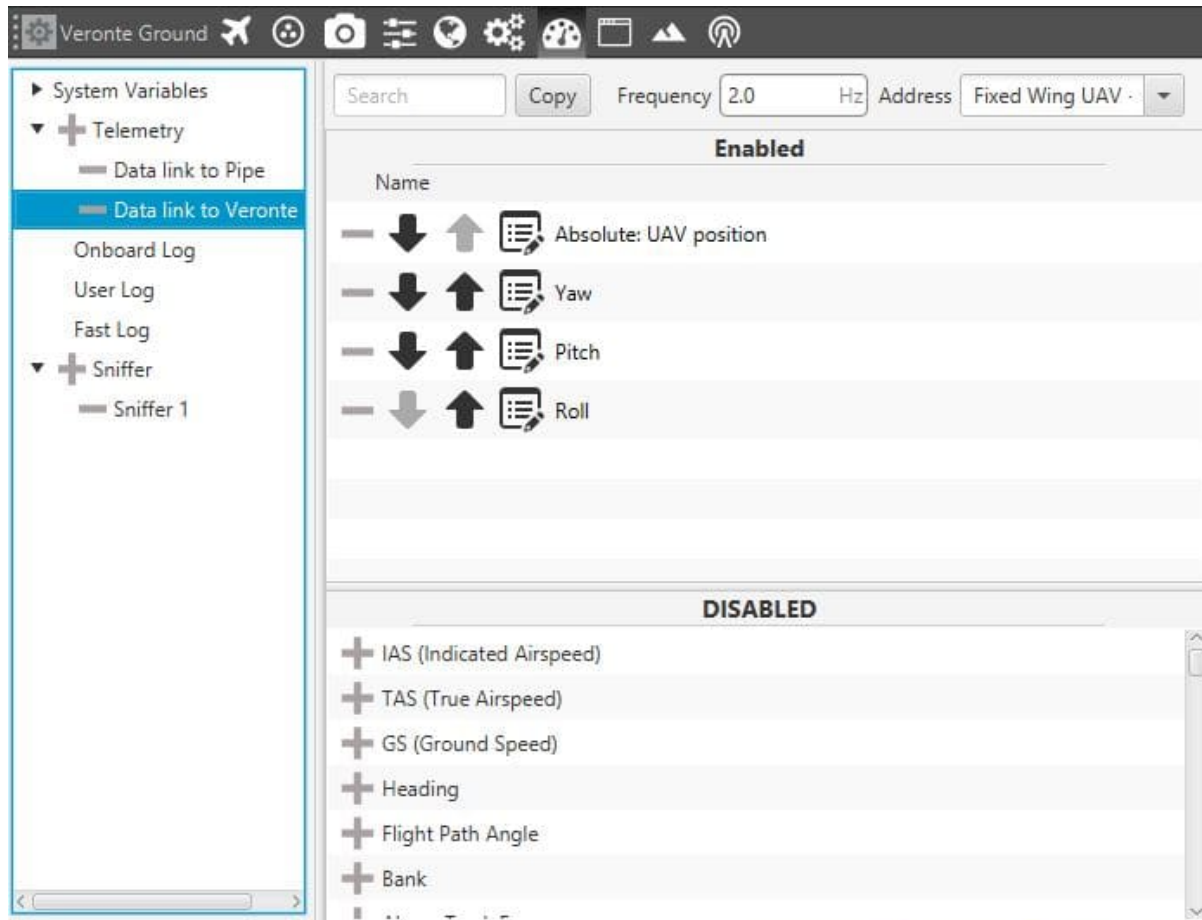
travel along the path with the speed indicated in the option Cruise, then it will decelerate or accelerate to the speed indicated in Waypoint Reach and then it will go back to the cruise speed.



Rendezvous Guidance – Menu 2

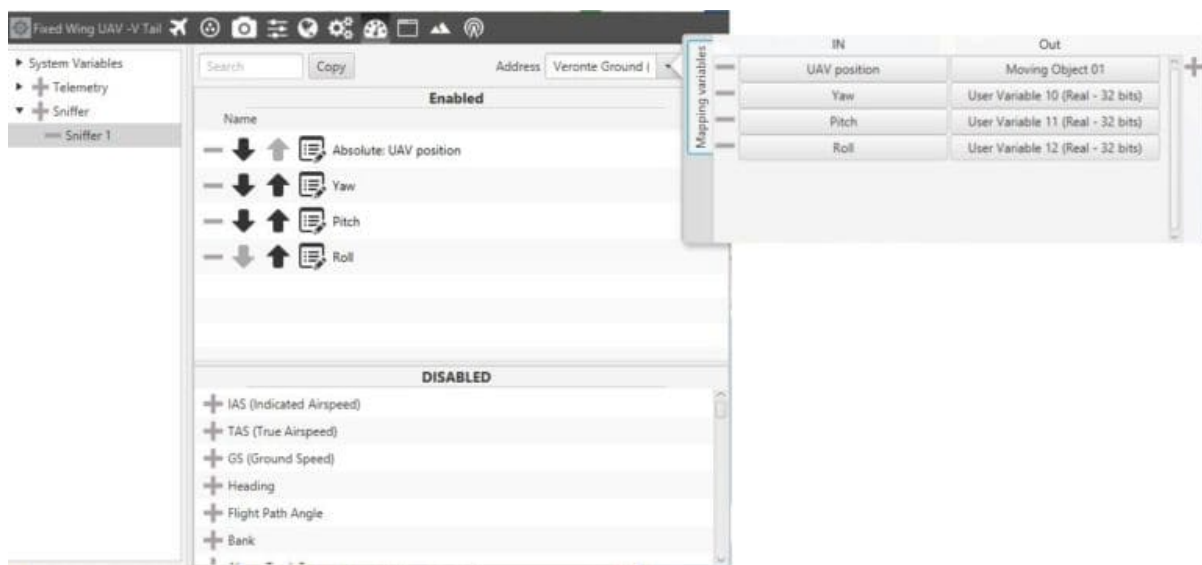
- **Guidance control (Horizontal and Vertical):** configuration of the PIDs from the system for both axes (see *Blocks*).
- **Docking base:** defines the position of the GNSS antenna connected to Veronte Ground.
- **Base yaw, pitch & roll:** defines the attitude from the base body. These values affect the navigation by orienting the Air unit to be equal to the attitude from the Ground one. To be configured with telemetry (example below).

In order to configure the Moving Object which is assigned to Docking Base and the Base attitude, the next procedure must be followed.



Ground Telemetry

In the Ground unit, go to Variables *Telemetry* – Data link to Veronte and add the variables Absolute: UAV position, Yaw, Pitch & Roll. A recommended frequency is 10 Hz. On address, point to the Air Unit (it is needed to have both units connected through the radio in order to be able to see them on the menu).



Air Sniffer

For the Air unit, go to *Sniffer* – Add a new Sniffer and configure the same Variables (keeping the same order) than in the Ground unit (in this case Absolute: UAV position, Yaw, Pitch & Roll). Address the Ground unit and in the gear next to it, configure the 4 incoming variables as System Variables. Assign UAV Position to Moving Object and the 3 variables from attitude to 3 different User Variables (keeping the same order as well).

In order to be able to see the Moving Object position in Pipe interface, go to Air unit configuration ref Variables – Telemetry – Data Link to Pipe and add Moving Object to the list of variables.

- **Docking relative position:** 3D point used to configure the offset for the approaching vehicle to the Docking base. This will be the difference from GNSS position that defines the landing point.
- **Rendezvous relative position:** 3D point used to configure the meeting point for the Air Veronte. This point will be tracked by the vehicle and, once reached, it will start travelling to Docking relative position. For VTOL, X and Y components must be equal. For both Docking and Rendezvous the axes are set according to Veronte orientation.

When defining a guidance system, we refer to a set of commands sent to the platform controller in order to make it carry out a certain task. This task could be follow a line, climb, land, hold one of its states at a certain value and so on.

In Veronte Pipe, it is possible to combine a series of guidances to create custom flight phases that will make the aircraft perform in a given way. For example, to create a Take Off phase, the guidances to be included could be a **Runway**, which defines a line over the runway to make the aircraft follow it, and a **Hold** that will keep the roll and pitch angle at zero to keep the aircraft level when it is accelerating on the runway.

Each Guidance contains a different set of parameters to be configured. All of them are presented as follows and its parameters are explained in detail.

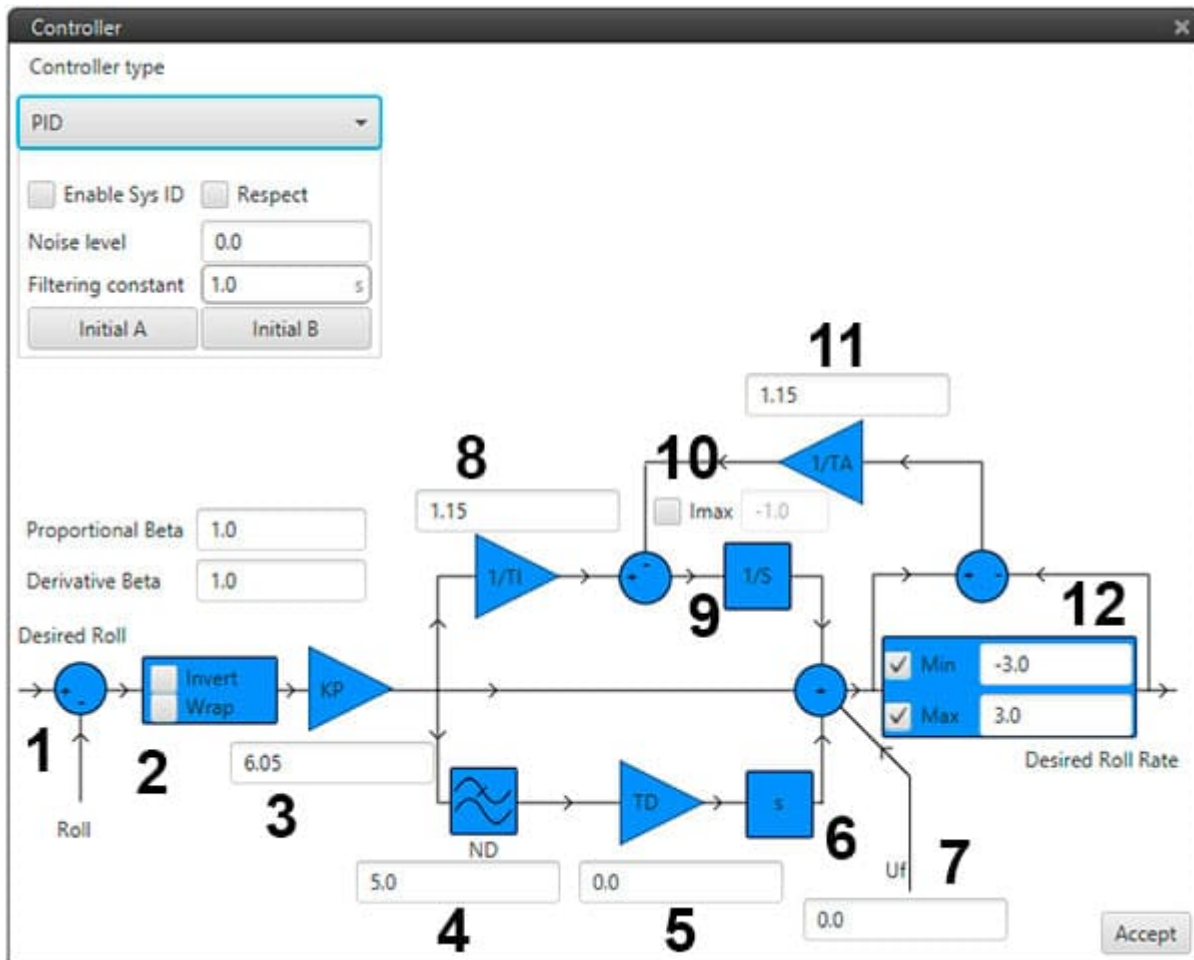
| Value | Description |
|---------|---|
| Taxi | Creates a linear path along the runway that is followed by the aircraft. |
| VTOL | Vertical take-off and landing. |
| Climb | Makes the aircraft climb from the start of the phase to another altitude. |
| Hold | Introduces in the control system a desired variable with a value specified by the user. |
| Cruise | Makes the aircraft follow a determined route created by the user. |
| Landing | Creates the route that the airplane will follow to land. |
| Yaw | Indicates the behaviour of the platform in the yaw axis. |

Note: in section *Configurations* there are some examples of how to combine these guidances to create phases for common platforms (Plane, Quadcopter, etc.).

7.2.4.1.2 Arcade

7.2.4.1.2.1 Arcade Mode Settings

The arcade mode could be considered as an aided manual control where the pilot sends a control command and not a PWN signal for the actuators. This control command is used by the Hold or Yaw guidance algorithm to provide the input for the controller. So, in this case, is not directly the autopilot which generates the PID inputs, but an optimal combination of its commands and the ones from the pilot.



PID Architecture

| Value | Description |
|-------|--|
| 1 | Measure |
| 2 | Invert: Change error sign/Wrap: Wrap to pi [-,]/It is used in some angular variables (radians) for avoiding numerical errors on the – to change and keep continuity of the error signal |
| 3 | Proportional gain |
| 4 | Discrete filter parameter |
| 5 | Derivative time parameter |
| 6 | Derivative gain |
| 7 | Constant value added to output (Feedforward Control) |
| 8 | Integral gain |
| 9 | Inverse integral time parameter |
| 10 | The maximum value of integral admitted |
| 11 | Anti-windup parameter |
| 12 | Output bounds |

Output values for PID controller refer to virtual control channels, units must coincide with servo trim configuration settings.

PID diagram represents the following PID model:

$$C = K_p \left(1 + \frac{1}{T_i} IF(z) + \frac{T_d}{\frac{T_d}{N} + DF(z)} \right)$$

PID Mathematical Model

- K_p=proportional gain
- T_i=Integrator time
- T_d=Derivative time
- N=Derivative filter constant

For the derivation and integration models, Backward Euler and Trapezoidal (respectively) models have been integrated:

- Backward Euler:

$$DF(z) = T_s \frac{z}{z - 1}$$

PID Mathematical Model - Backward Euler

- Trapezoidal

$$IF(z) = \frac{T_s}{2} \frac{z + 1}{z - 1}$$

PID Mathematical Model - Trapezoidal

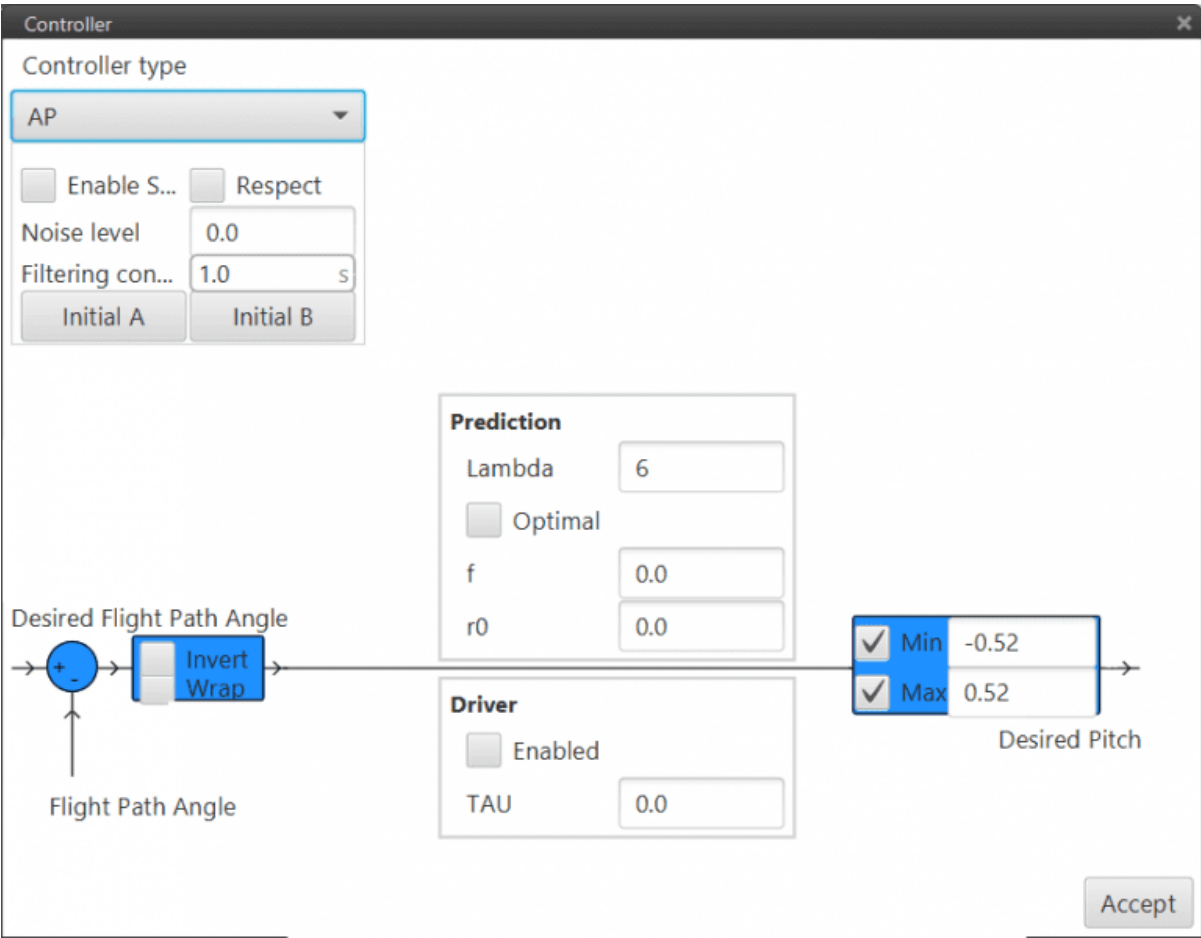
= T_d/N where is the time constant on a first order low pass filter (LPF). In Laplace notation:

$$LPF(s) = \frac{1}{\tau s + 1}$$

PID Mathematical Model - Laplace

7.2.4.1.3.2 Adaptive-Predictive Control

This controller incorporates an algorithm for self-tuning i.e, the controller settings are ajusted automatically.



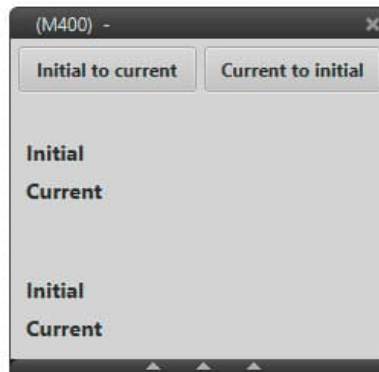
PID Mathematical Model - Adaptive Predictive controller configuration

The next table describes all the parameters available and what means each one of them.

| Field | Description |
|-----------------------|--|
| Enable Sys ID | Activates the option of system identification. The plant is modified continuously by predicting a new one according to a set of parameters. This option must be activated when working in Adaptive – Predictive control. |
| Noise Level | Supposed noise level during System Identification |
| Filtering Constant | Constant value for the filter |
| Initial A & Initial B | Used to establish the initial plants constants for the system identification process |
| Lambda | Parameter that defines the control aggressiveness. High value means less aggressive control, low lambda means more aggressive control |
| Optimal | Select this option to use the Optimal AP algorithm |
| f | Sampling input period |
| ro | Control output period |
| Enable/Disable Driver | Mark this option to use the Driver Block |
| TAU | Time constant for the desired default trajectory |

In addition, it is possible to display an AP panel in the workspace that allows the user to perform the following actions:

- **Initial to Current:** to set the initial platform model to the current one.
- **Current to Initial:** to save the actual model to the initial one



PID Mathematical Model - AP Tool in Workspace

7.2.4.1.3.3 Gain Scheduling

This approach uses a group of linear controllers for different operation points. Each controller adjusts its gains automatically according to scheduling variables.

Veronte incorporates 4 approaches for this type of control:

- **Table Scheduler:** gain scheduling controller with variation of the parameters according to a table interpolation. In this option, the values of the PID controller vary according to a variable that is selected when this option is marked. For example, if the IAS is selected as the scheduling parameter, depending on the velocity of the aircraft the parameters of the PID will change.

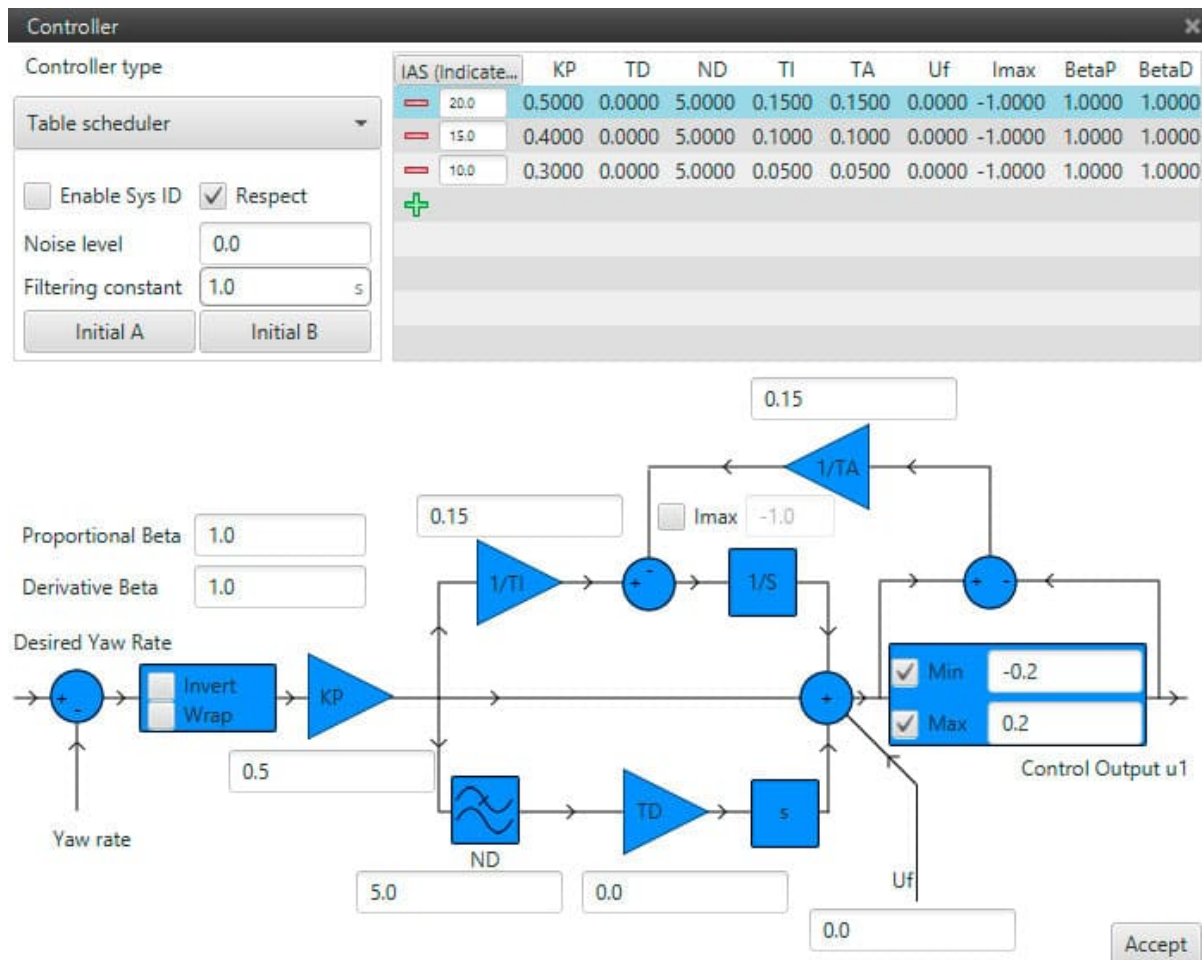


Table Scheduler controller

- **Inverse Scheduler:** gain scheduling controller with variation of the parameters according to the selected variable using inverse proportionality.
- **Proportional Scheduler:** gain scheduling controller with variation of the parameters according to the selected variable using direct proportionality.
- **Quadratic Scheduler:** gain scheduling controller with variation of the parameters according to the selected variable using quadratic proportionality.

For the last three approaches, the only gain that changes according to a certain variable is the proportional one. Now there is not a table interpolation but a mathematical expression, that can be inverse ($KP1/V1=cte$), proportional ($KP1 * V1=cte$) and quadratic ($KP1 * V1^2=cte$). The nominal point given by the expressions will remain fixed. Therefore, the proportional gain will change when $V1$ varies in order to maintain the same value.

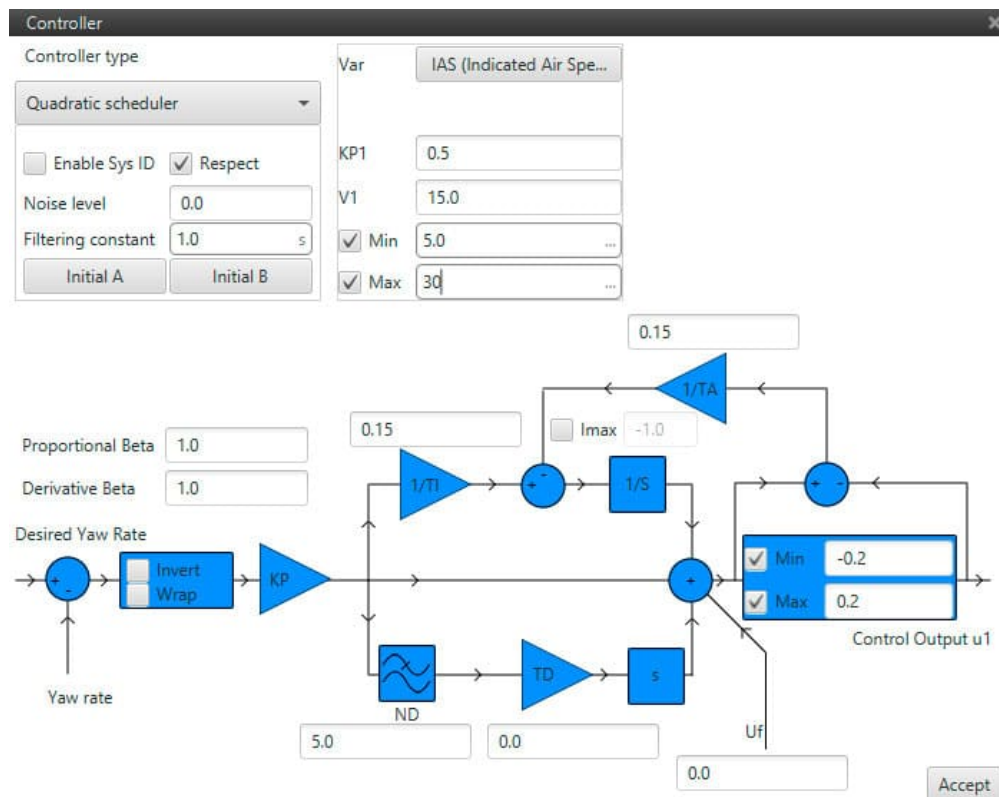
| | C | Kp | Result (Kp) |
|--------------|--------------------|---------------------------|--------------------------|
| Inverse | $\frac{Kp}{V_i}$ | $C * V$ | $Kp_i \frac{V}{V_i}$ |
| Proportional | $Kp_i * V_i$ | $\frac{c}{V} (V > 0)$ | $Kp_i \frac{V_i}{V}$ |
| Quadratic | $Kp_i * V_i * V_i$ | $\frac{c}{V+V} (V^2 > 0)$ | $Kp_i \frac{V_i^2}{V^2}$ |

Scheduler table

In the previous expressions, shall be clear that limits are applied to the variable (V) and not to the gain (Kp).

The following figure shows the quadratic case, where the proportional gain changes according to the IAS, having as nominal point (KP1 and V1) 0.5 and 15 m/s respectively.

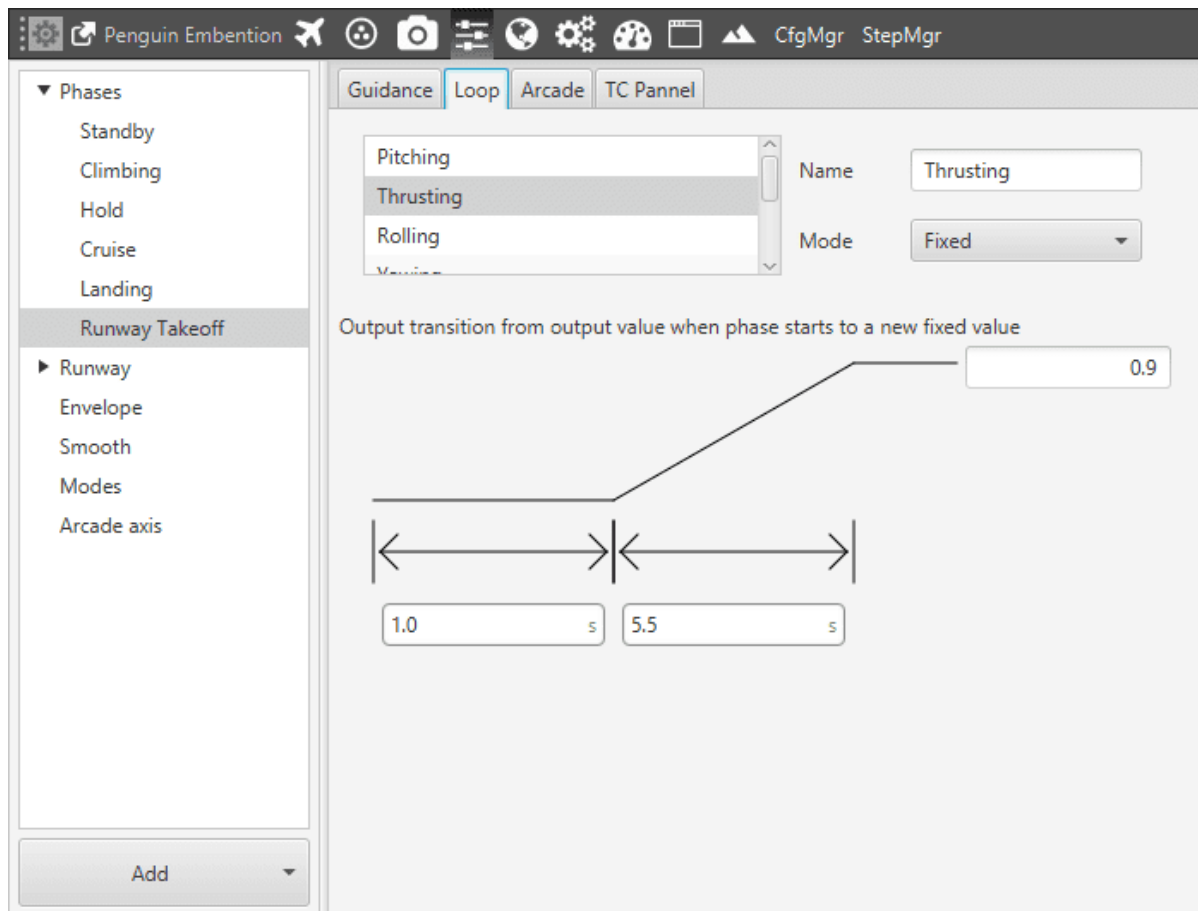
The values Min and Max are used to establish a limit of the scheduler. Below and above those limits, the system works as a conventional PID with the gain indicated in KP1.



Quadratic Scheduler controller

7.2.4.1.3.4 Fixed Control

When **Fixed** mode is selected, the Open Loop Control parameters are set to a fixed value.



Fixed Value Settings

Three values must be entered:

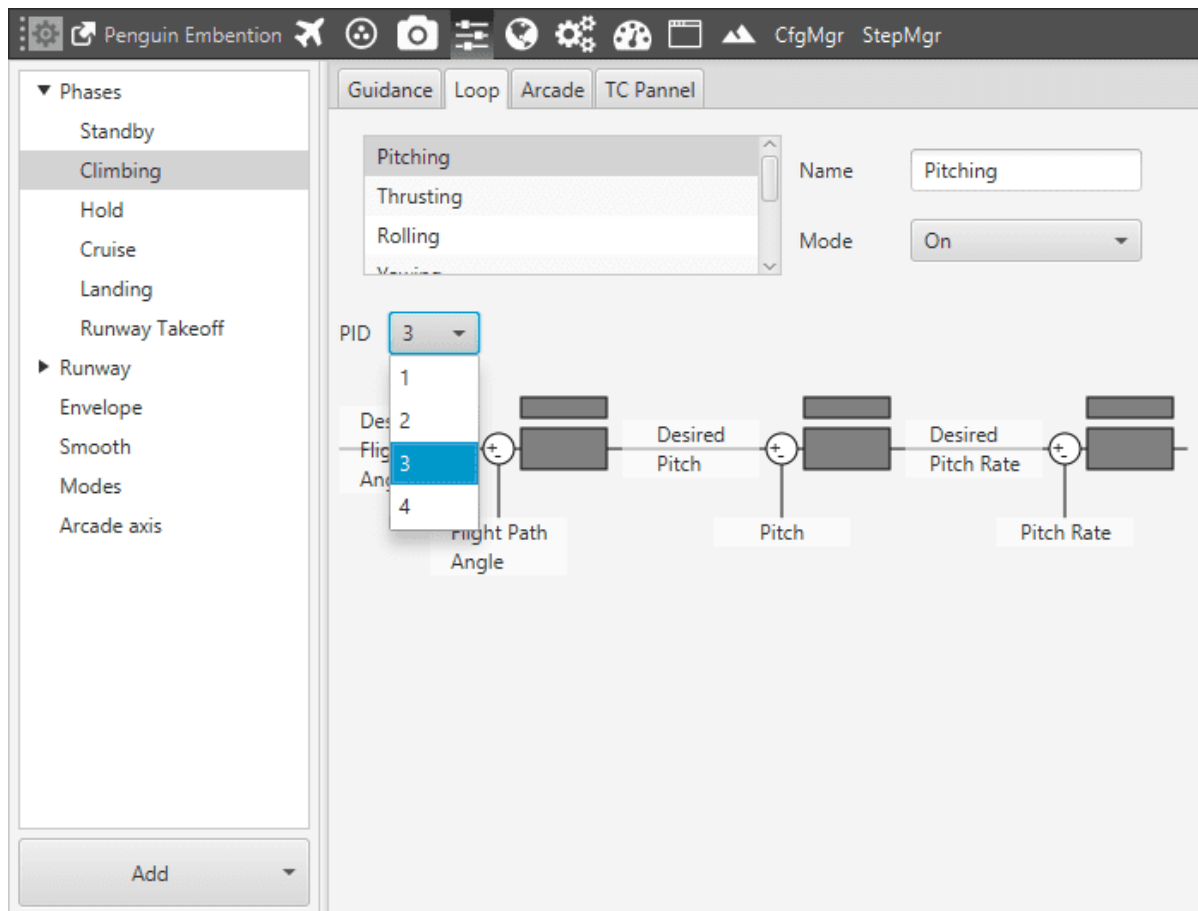
- Remaining time in the starting conditions.
- Transition time
- Variable final value.

In each one of the mission phases, it is possible to configure a controller for each control channel defined on Veronte Configuration. There are three different options for the control status.

| Value | Description |
|-------|---|
| Off | Disables the controller. |
| On | Enables the Closed Loop Control. |
| Fixed | Sets the Open Loop Control parameters to a fixed value. |

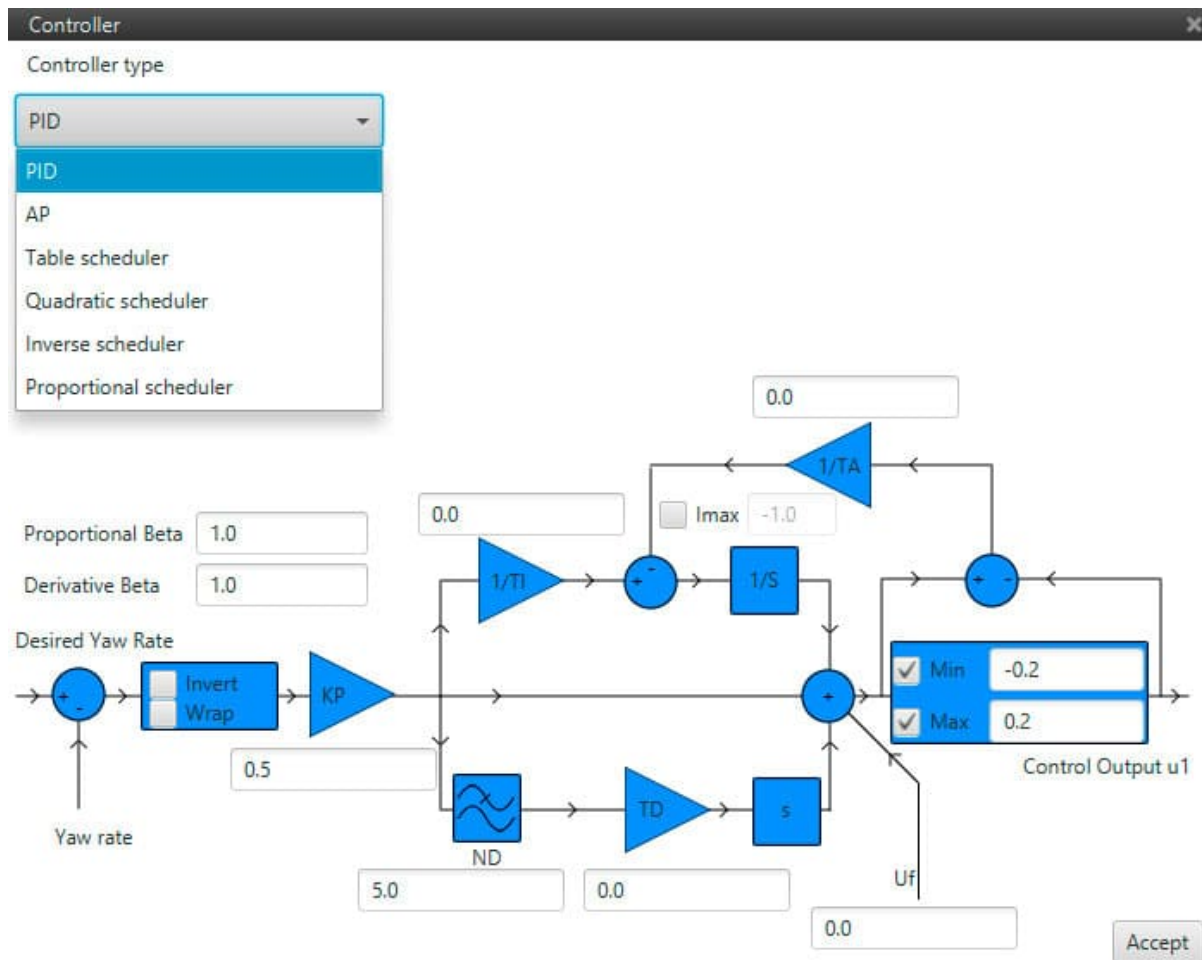
7.2.4.1.3.5 PID Settings

When creating the controller for a channel, it is possible to select up to four different loops connected in series.



Loops Diagram

For each block, it is possible to configure the controller type and its parameters by selecting it on the pull-down menu shown in the following figure.



Loop Configuration

There are six different options that can be implemented for a controller on Veronte Pipe:

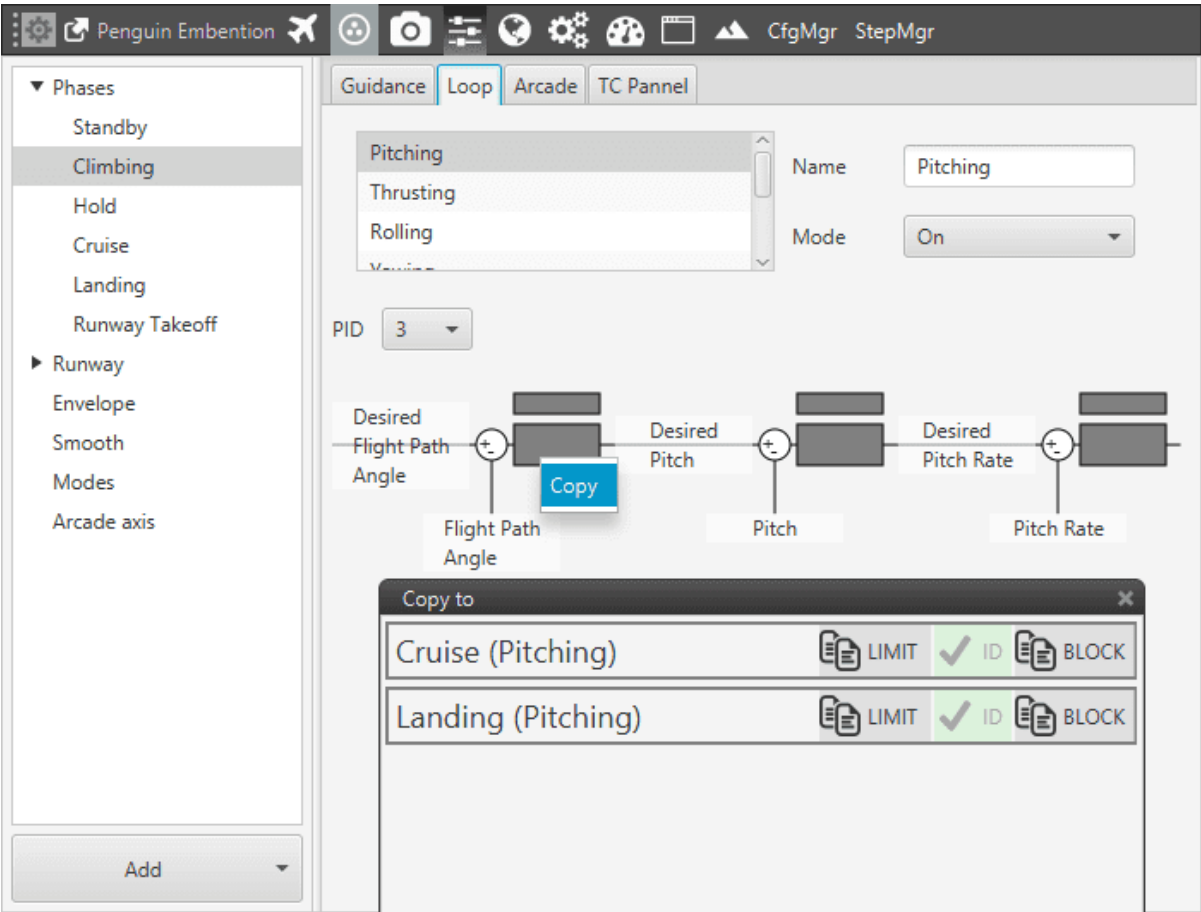
| | |
|----------------------------|--|
| PID | Classical PID controller. |
| AP | Adaptative-Predictive control. |
| Gain scheduling controller | Gain scheduling controller with variation of the parameters: Table Scheduler, Inverse Scheduler, Proportional Scheduler, Quadratic Scheduler |

In addition to the controller type, there are another set of parameters that can be changed in the window of each block.

- **Enable Sys ID:** activates the option of system identification. The plant is modified continuously by predicting a new one according to a set of parameters. This option must be activated when working in Adaptative-.Predictive control.
- **Respect:** this option is used for transition between phases. When activated the output of the controller is respected (is kept) when going from one phase to another. Normally, this is used in the internal control loops (rates) only to avoid great steps in the control parameters during transitions. If Respect is active in the external loops, the control will maintain attitude angles and heading/fly path angles during phase changing (for example), and this kind of control could be too much aggressive for the platform (depending on phases configuration). The Respect option **can be activated in PID controllers only**.
- **Noise Level & Filtering constant:** parameters used for the AP controller.
- **Initial A & B:** these buttons are used to establish the initial plants for the system identification process.

7.2.4.1.3.6 Exporting PIDs To Other Phases

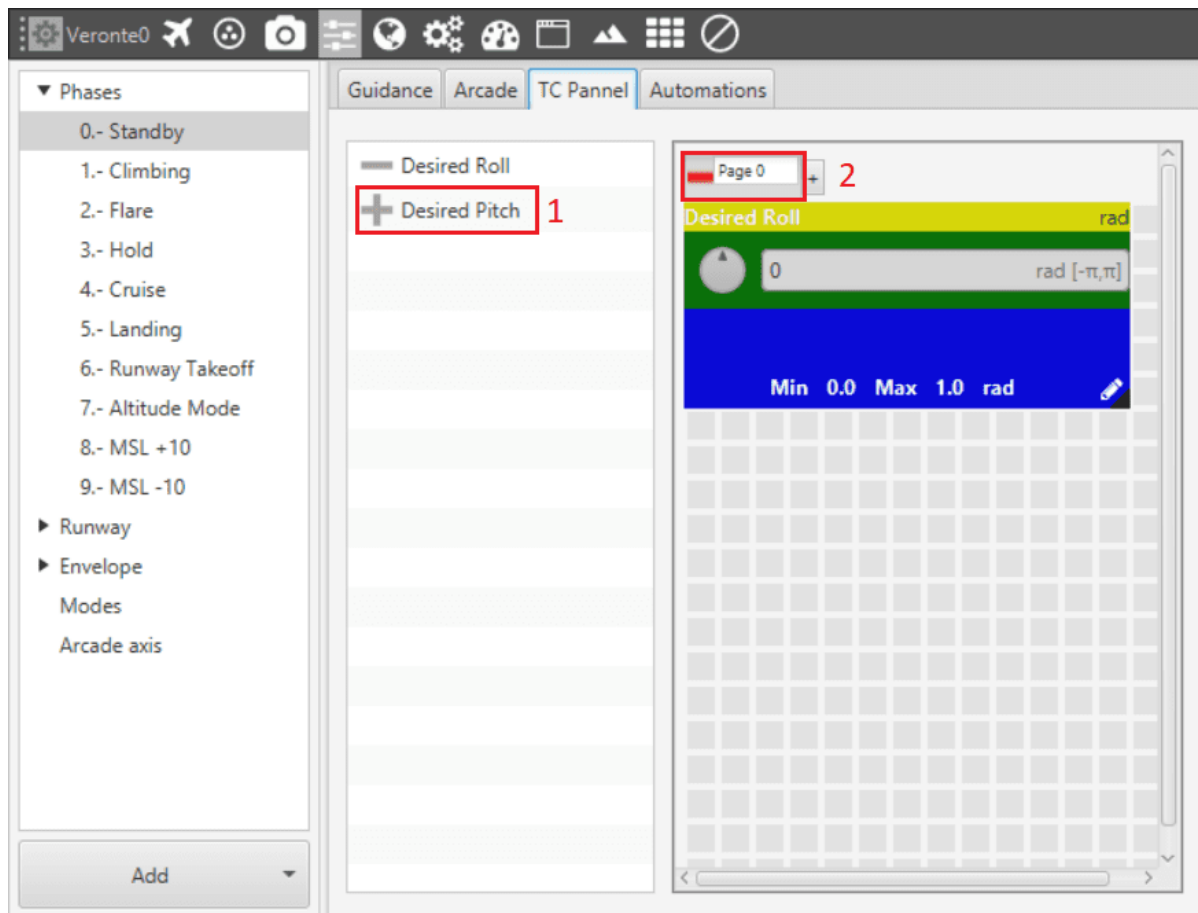
Once it is considered that the PID is tuned, the user can easily export that PID in order to use it in other phases. To do so, just select **Copy** by right-clicking on the desired PID and click on the **Block** cell of the desired phase. That would copy only the gains of the PID (P, D and I), if the user also wants to copy the **Limits**, that column has to be checked.



PID Exporting

7.2.4.1.4 TC Panel

This menu sets the variables that will be accessible during the flight from the Veronte Panel. Considering as an example a typical navigation phase, the parameters that can be changed from the TC Panel appear in the following figure. In recent versions of Veronte pipe, the most of these variables are managed by Workspace widgets.



TC Panel

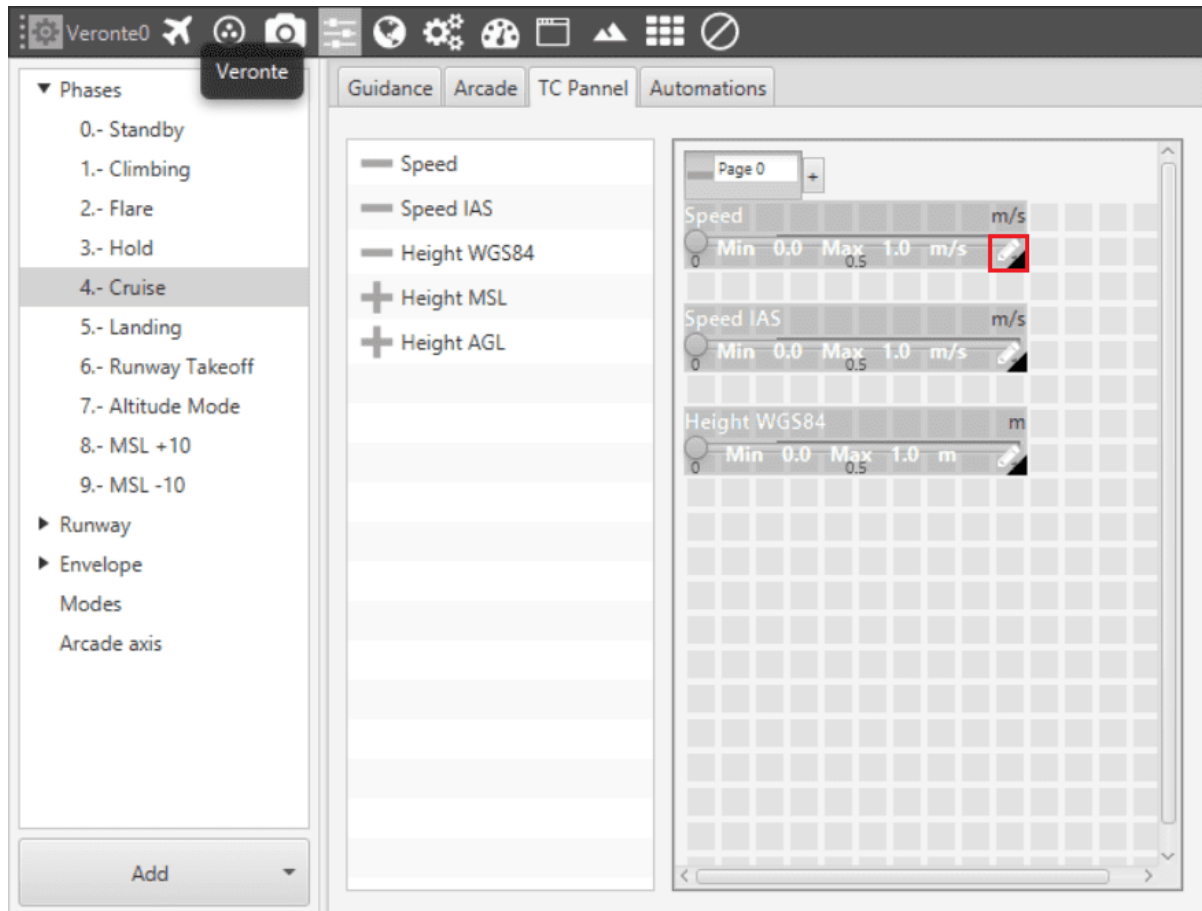
- Number **1** is the button used to add a variable to the TC Panel of a phase (the “-” button is used to remove an element).
- Number **2** indicates the TC Panel page where the current element is being added. Pipe allows user to set more than one page in order to arrange all the variables in an easily accessible way.

7.2.4.1.4.1 Cruise Guidance

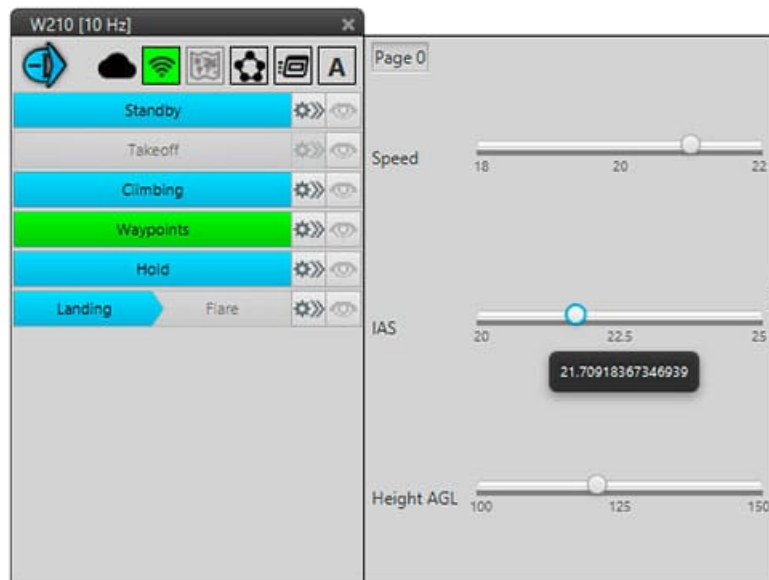
In addition, if the Guidance of the phase is Cruise, the TC Panel can also hold the option of creating different types of interactions with the UAV:

- **Desired variable change:** Speed, Speed IAS, Height AGL, Height MSL, Height WGS 84.

The variable changing allows the user to modify the desired variable in real time during the flight. In the configuration of the TC Panel, a maximum and a minimum of the variable has to be set (red box).




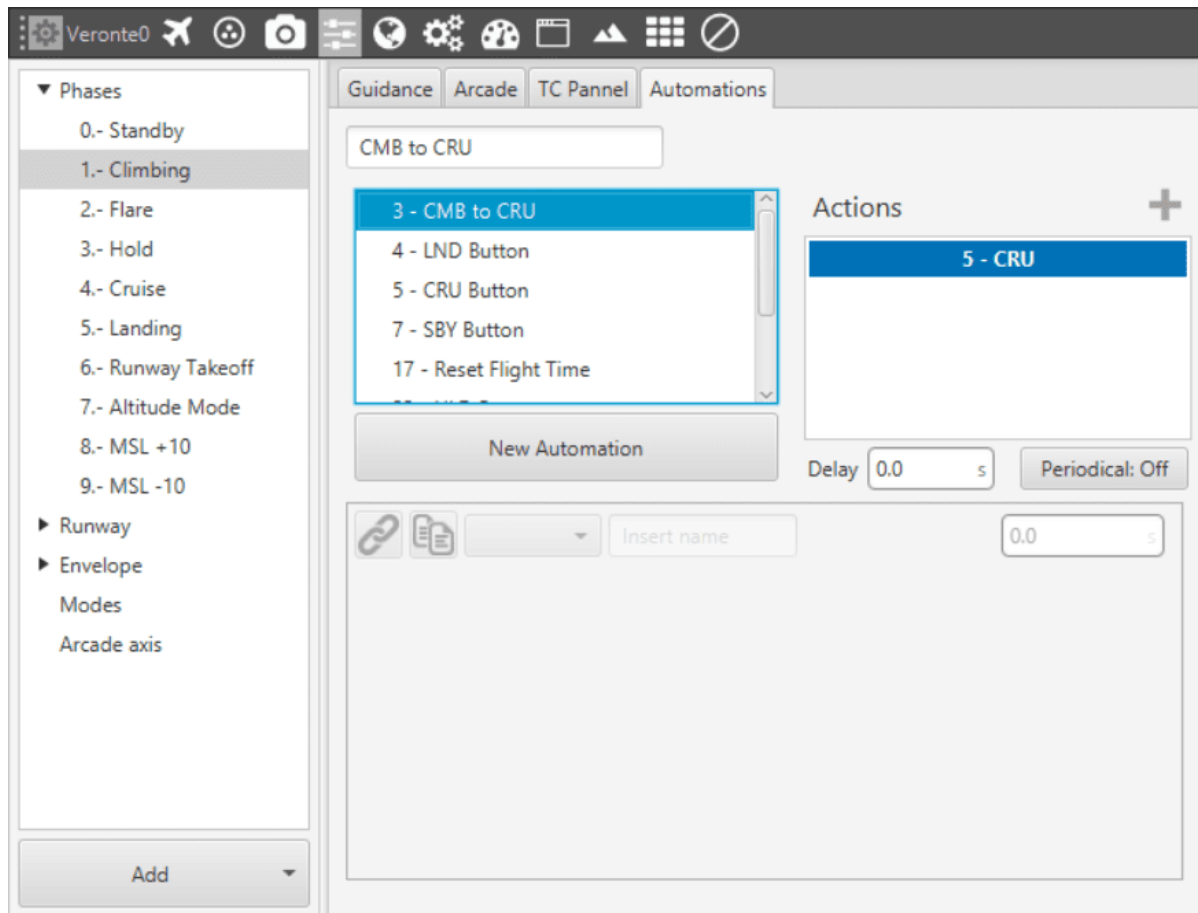
TC Panel variables change set



TC Panel during Waypoint phase (Variables)

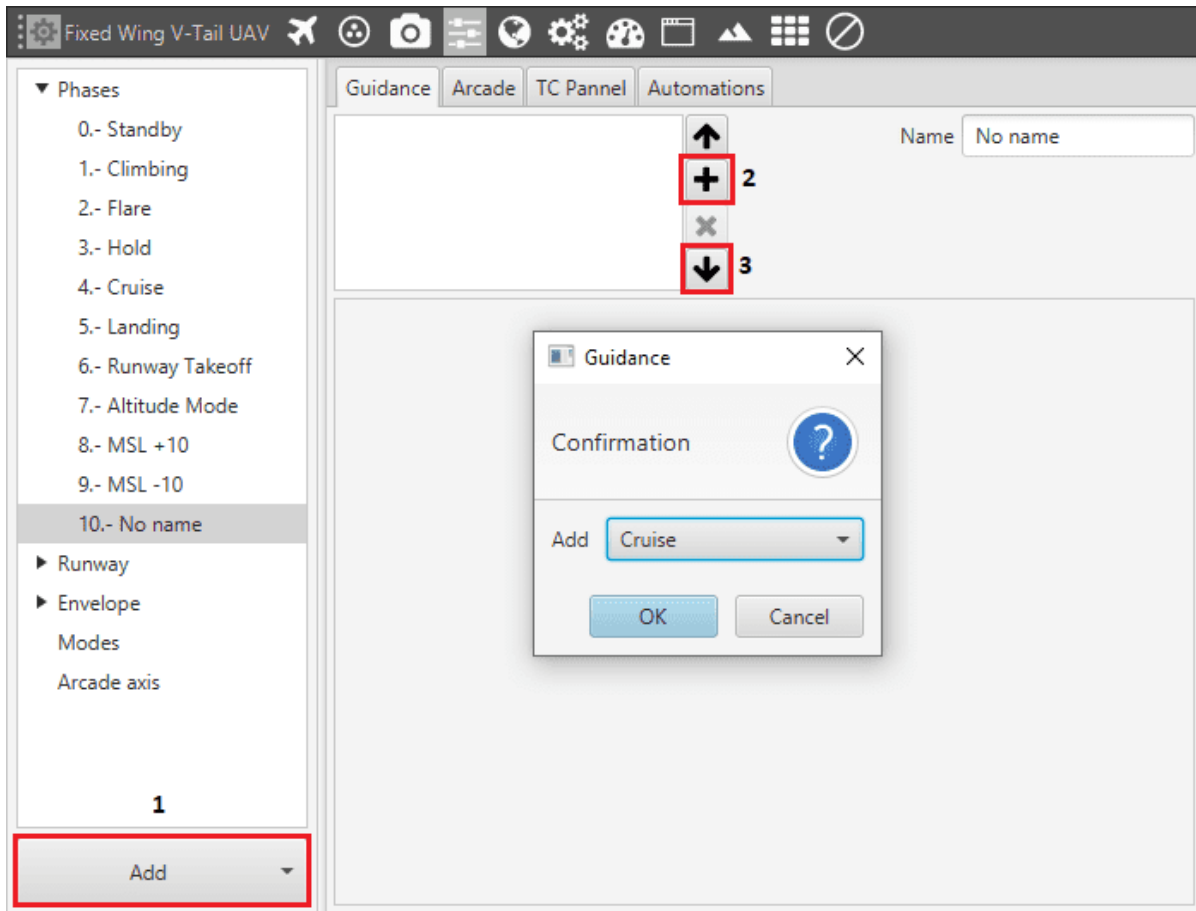
7.2.4.1.5 Automations

The last tab in phases menu is Automations. Here, we can find the automations that take part in each flight phase. From this menu you can modify or create new automations. Another option is clicking on  in the set-up toolbar (the changes are update in both menus). For more information, see [Automations](#).



Automations

The **Phases** menu contains the guidance and control commands used to rule the vehicle. It is possible to create different phases, each one of them with a series of guidance commands and with different control systems. Commonly, the phases correspond with different stages of the flight operation, for example, in an airplane the phases could be: Standby, Take Off, Climb, Cruise and Land. So it is clear from that in each one of them the commands generated to control the vehicle have to be different.



Control Phases Configuration Menu

When pulling down the phases option, the different phases created appear.

- To create a new phase click on **Add**, and select **Phase (1)**. The new phase appears by default with the label **No Name** and with any guidance defined.
- Pressing the **Add** button (2), a new Guidance is added to the current phase. The section *Guidance* contains the information about all the guidances available on Pipe.
- Lastly, the button (3) is used to order the guidances when there is more than one of the same type.

Arcade refers to the aided mode of controlling a platform, where the control system aids the pilot to improve the piloting experience. All the information about this option is available in section *Arcade*.

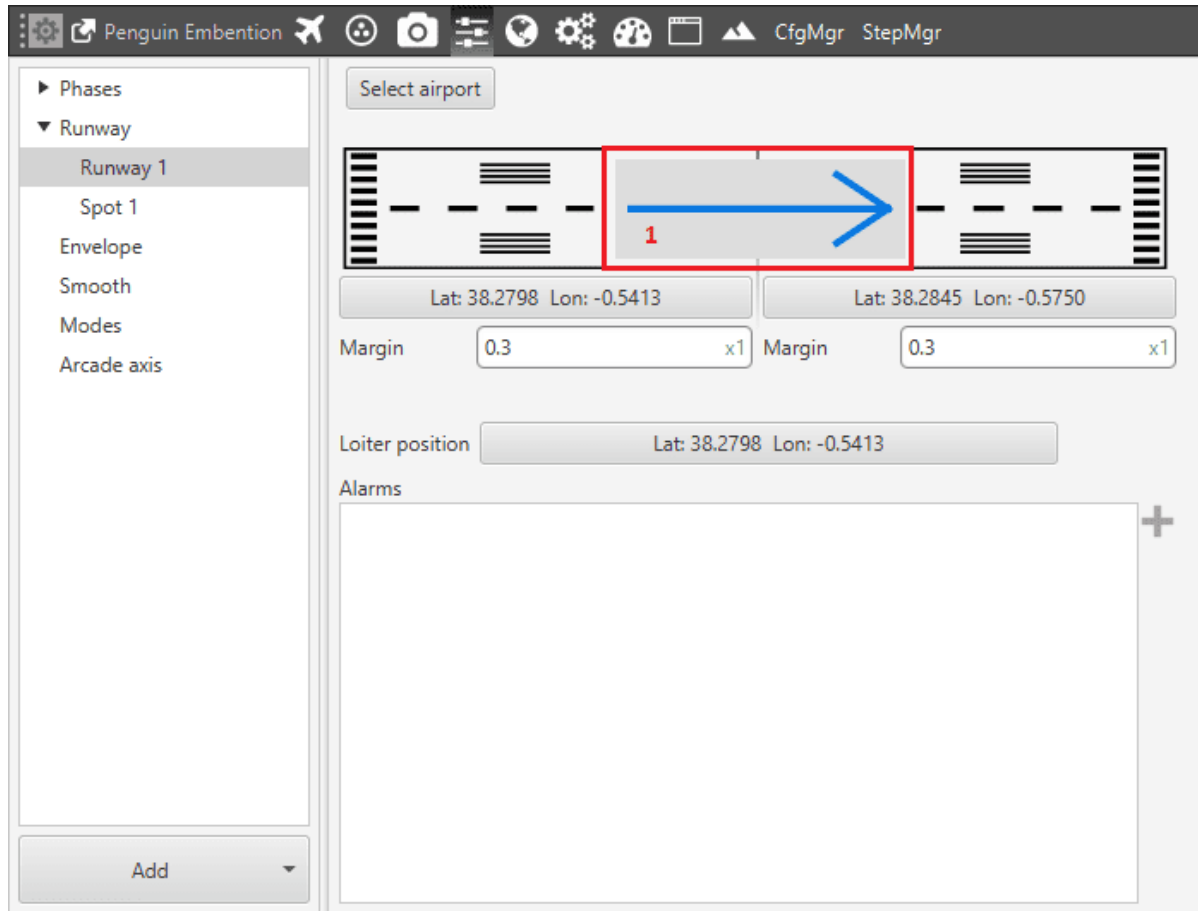
The TC Panel tab is used to include in Veronte Panel a series of variable to be changed in real time (while the platform is flying). Refer to *TC Panel* for more information about this.

Finally, Automations includes automatic changes or action buttons that can occur in each phase. In *Automations* there is more information about this.

7.2.4.2 Runway

7.2.4.2.1 Runway

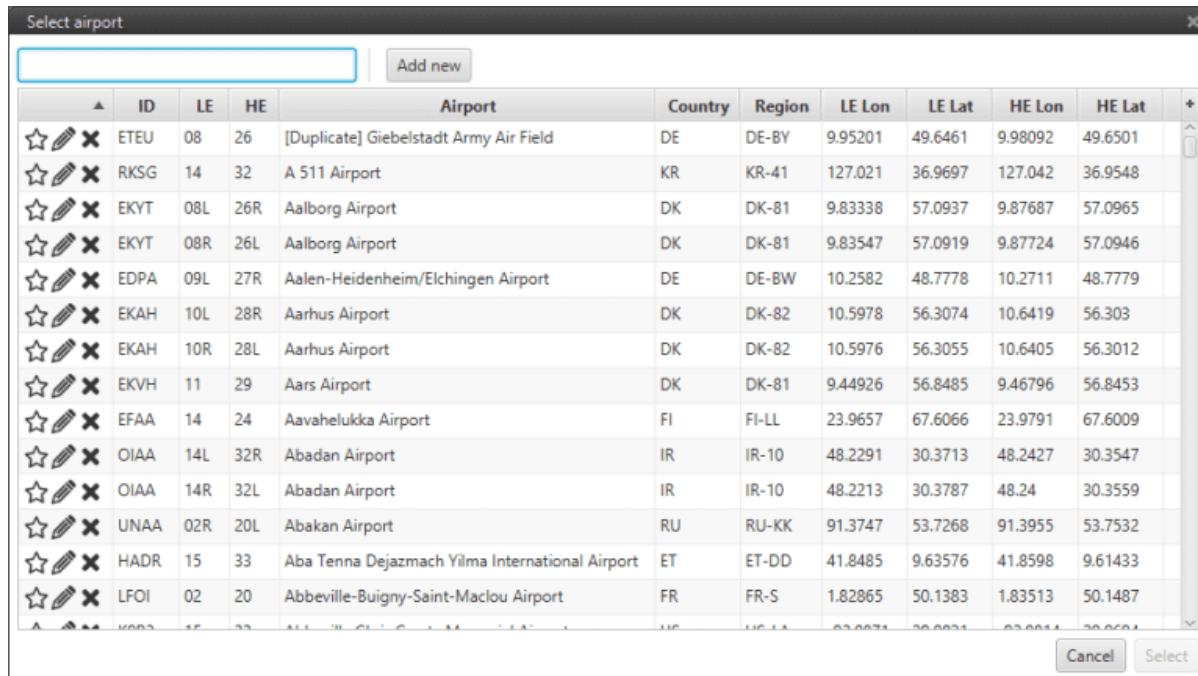
This menu allows the user to configure a **Runway** and **Spot** which are used during the **Phases** configuration. It is possible to configure more than one previously.



Runway Configuration Menu

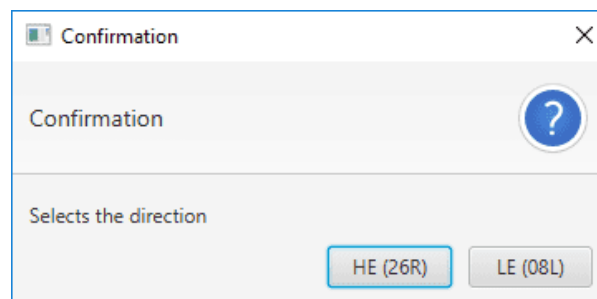
- **Margin:** percentage of the runway distance at which the airplane will try to touch the ground.
- **Loiter position:** this option defines the loiter point of reference (Runway Loiter) and the altitude that the aircraft will reach during the climb.
- **Alarms:** When an alarm is selected, e.g fuel lost. The aircraft will perform the actions associated to that alarm in the selected runway.

Veronte Pipe incorporates a database with different Airports. When the option **Select Airport** is selected the following panel is displayed.



Airport Selectiong Menu

Once the user selects the desired airport, the different coordinates will be introduced automatically. In addition, the direction of the runway is asked.



Runway Confirmation

On the other hand, it is possible to introduce the coordinates manually and the direction of the runway can be changed in the configuration menu by pressing 1.

7.2.4.2.2 Spot

This option refers to a kind of runway where a initial point and its azimuth is defined by **Select airport** option or the user information. Besides, it is necessary to define a **Delta** angle as is shown in the image. The aircraft will land or take off using the best orientation computed inside the area limited by the parameters introduced.

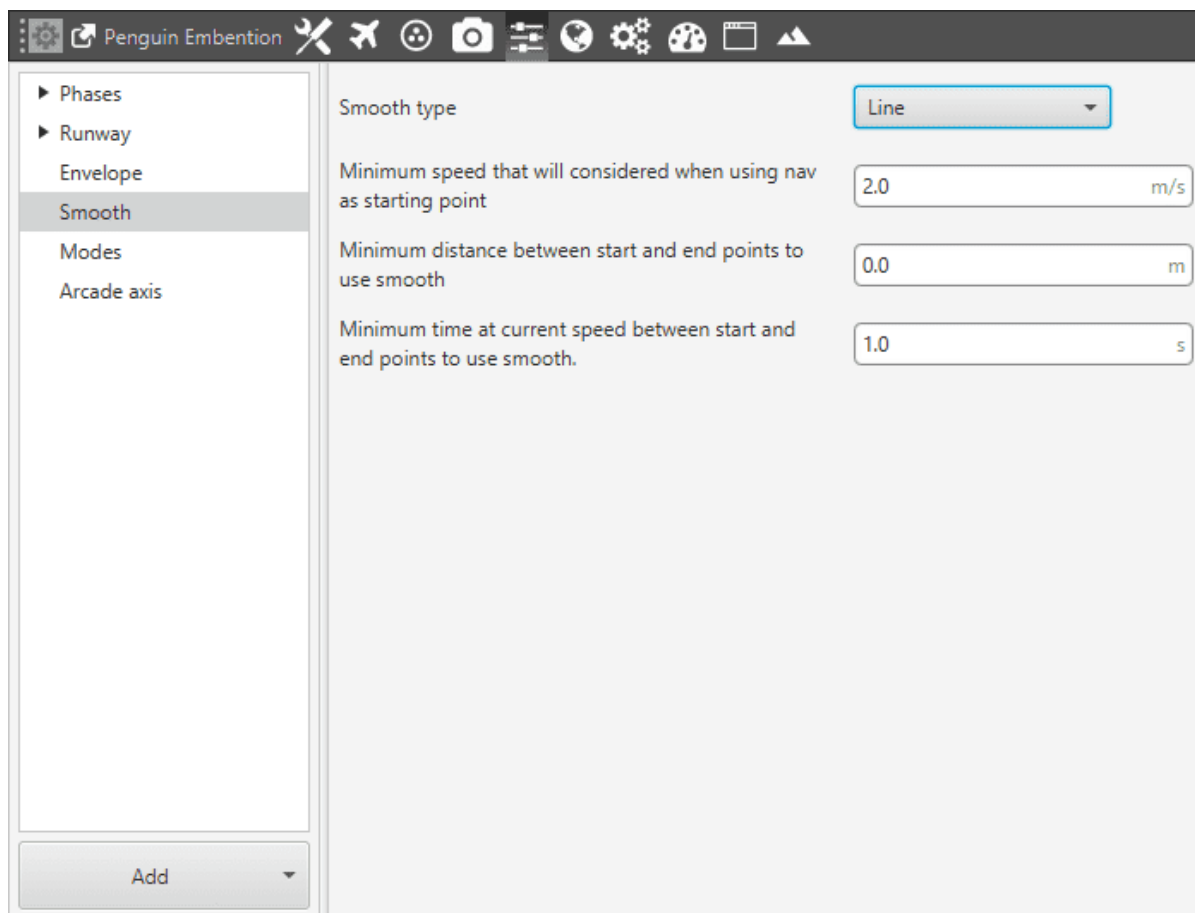
Spot Configuration Menu

7.2.4.3 Smooth

The Smooth menu establish the parameters for the smoothing of the aircraft trajectory when changing from a phase to another, i.e to go from the final point of a phase to the start of the new one.

There are two different ways to smooth the path:

- **Line** creates a linear track between the two points of the smoothed trajectory being possible only to set the position of the start and final points.
- **OGH** creates a cubic curve where is possible to set the position and velocity that the aircraft will have at those points.



Smooth Configuration Menu

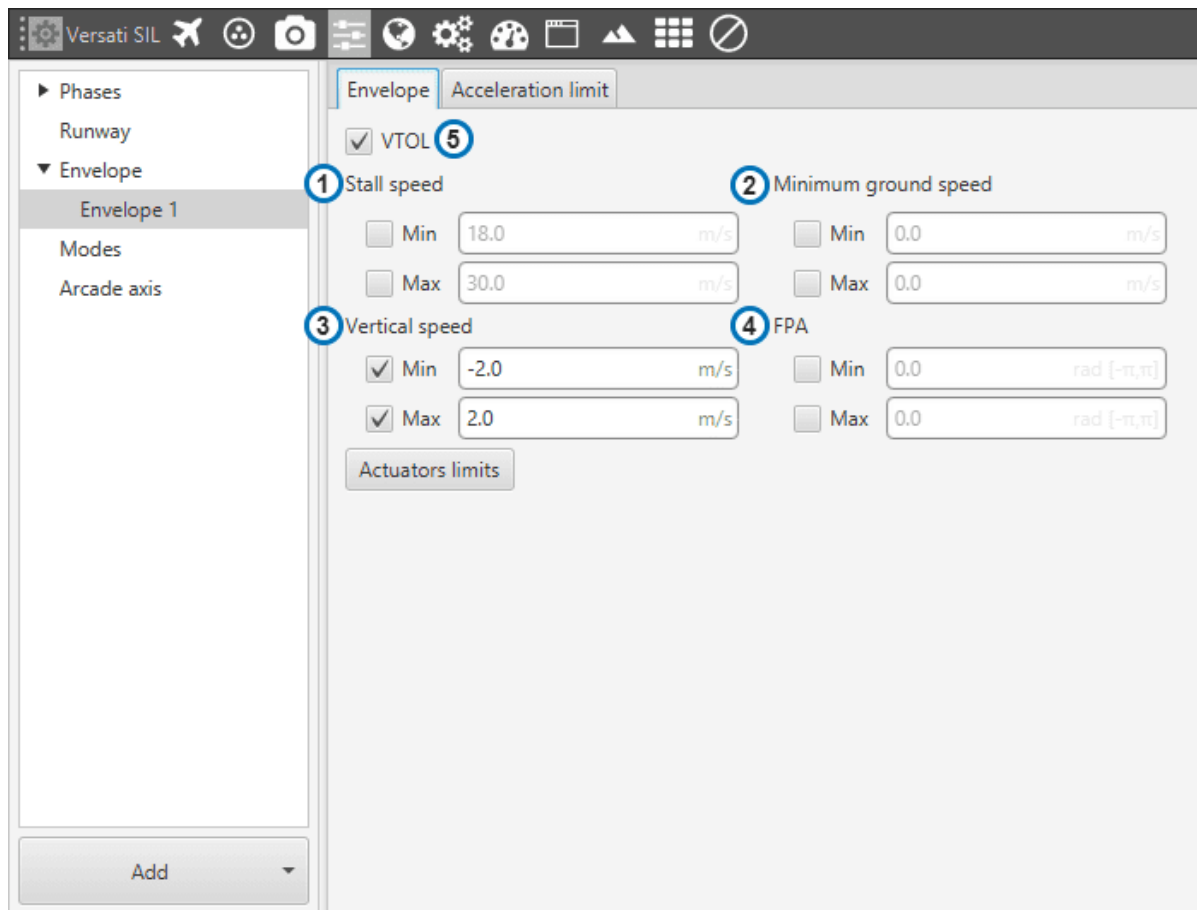
The other options that appear in this menu are parameters to limit when the smooth will be used or not.

- The first one establish the minimum velocity at which the smooth will start working,
- The second one is the minimum distance between the start and final points of the trajectory to begin with the smoothing.
- The last option is the minimum time between points to use smooth, considering that the aircraft flies at the current speed.

All these options are safety features to avoid a malfunction of the system when the final and start points of the phases are close to each other.

7.2.4.4 Envelope

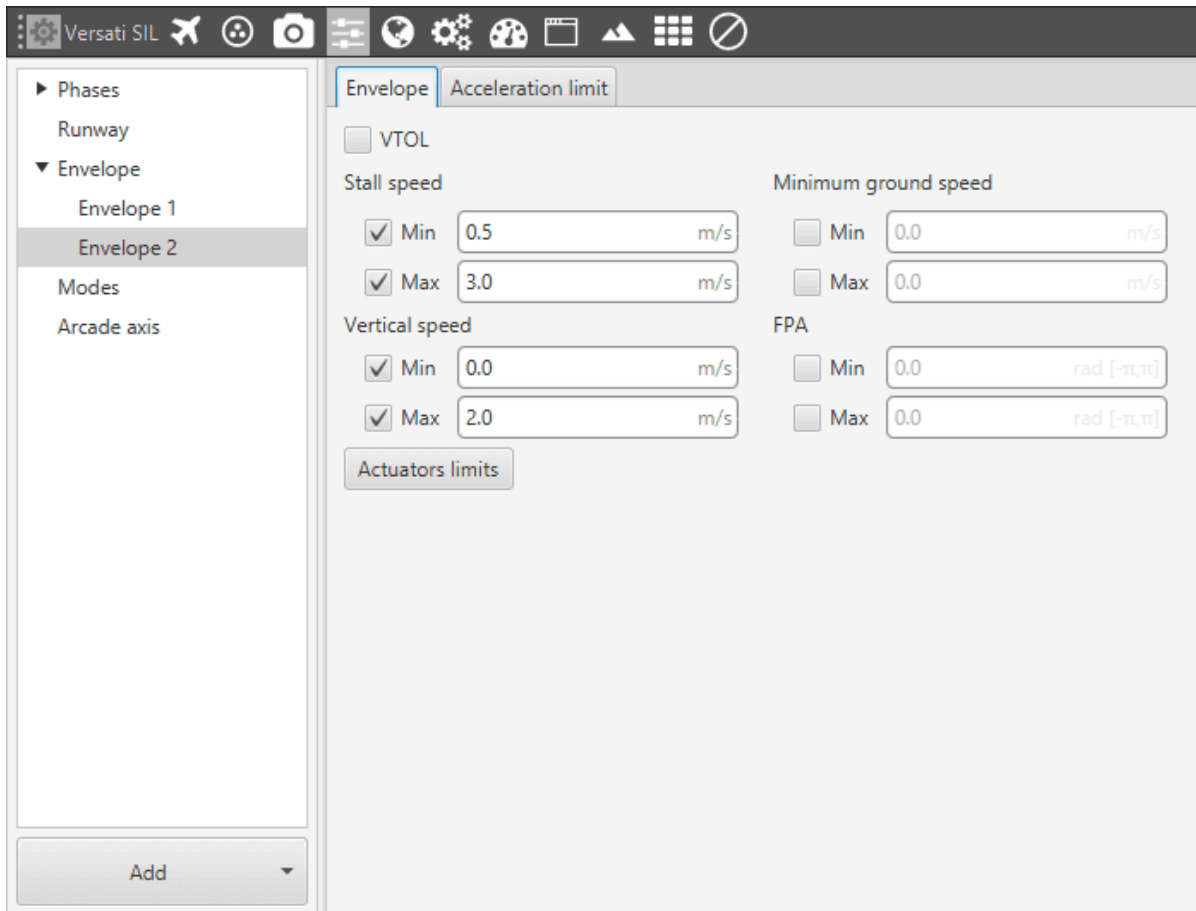
Menu to configure the flight envelope of the aircraft. Here the limits that will not be exceeded during the operation are set.



Envelope

1. **Stall speed:** limit for the indicated airspeed (IAS). The value indicated here has effect over the “Cruise” guidance, but is overridden if there is a Hold command on the IAS, so the user must be careful with the velocity commands. Besides, there is an option name as stall speed max that refers to the limit the uav shouldn’t exceed.
2. **Ground speed:** minimum and maximum ground speed of the platform. In case of strong wind, these parameters set the minimum GS that the aircraft can reach, for lower values than this one the thrust will be automatically increased to gain speed and avoid a point where the platform is stopped in the air.
3. **Vertical Speed:** similar to the previous limit. It sets the minimum and maximum value for the vertical speed of the platform.
4. **FPA:** maximum and minimum values for the flight path angle (angle of climb or descent).
5. **VTOL:** If this option is enabled, when the vehicle reaches the last waypoint of its route (and it is an open path), a hover is done instead of a loiter point.

It’s possible to insert multiple envelopes (useful for hybrid configurations). The change between envelopes can be performed using [Automations](#).

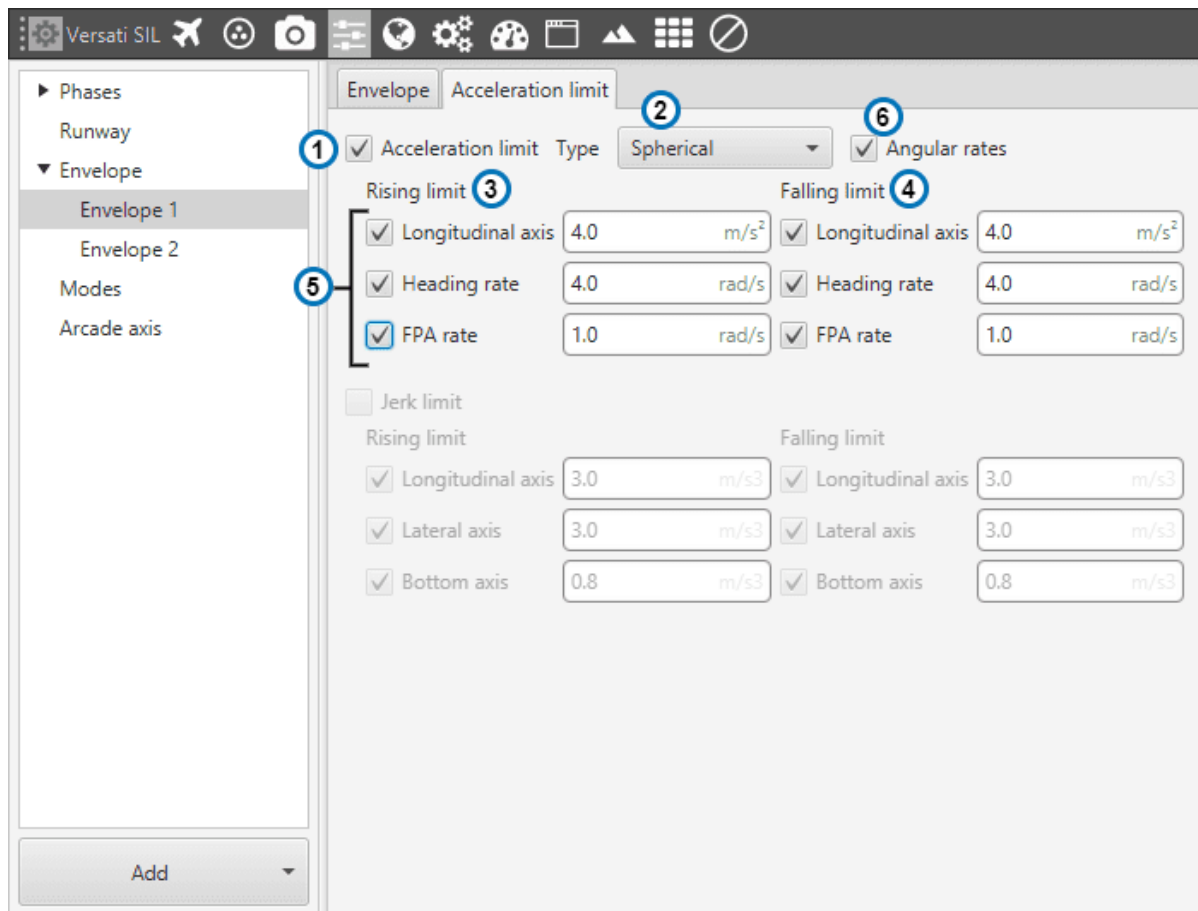


Multiple envelope configuration

In the second tab there are more options to fix the limits (rising and falling) of acceleration and jerk in SI units.

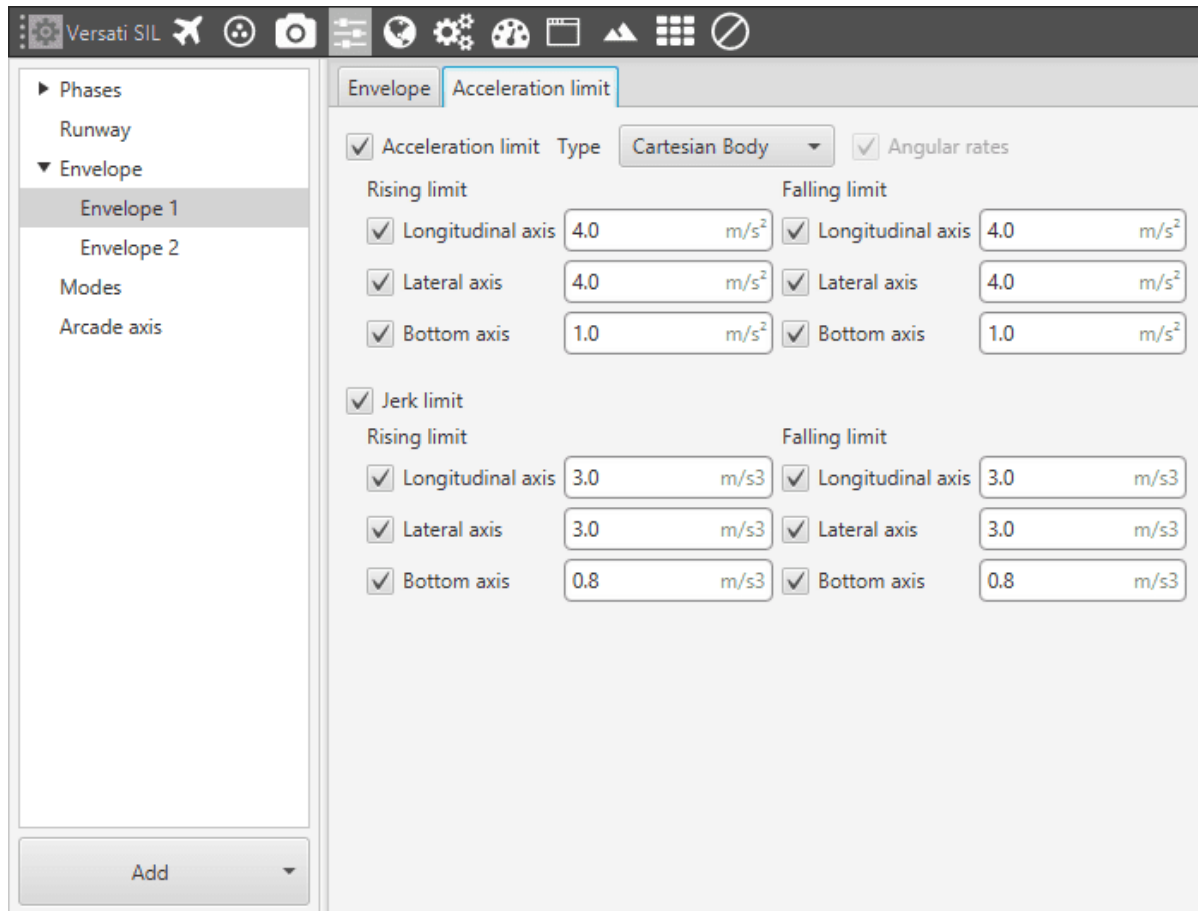
Acceleration limits are applied in any phase with position guidance. They modify the desired velocity. The algorithm compares the current desired velocity with that stored in previous step. There are six values to define. The configurable options in this menu are:

1. Enable/Disable the acceleration limit.
2. **Type** : User can choose between cartesian frame or spherical frame.
3. **Rising limit** : The limit for acceleration. If desired velocity has the same sign as in the previous step, and it's is lower (in absolute value) this limit is applied. For example, if a multicopter is flying in negative X direction, and it has to increase desired velocity in same direction this limit will be applied.
4. **Falling limit** : The limit for deceleration. In the case rising limit is not used.
5. **Axes** : In cartesian mode they reference body frame (Longitudinal X, lateral Y, and Bottom Z). In spherical type, the algorithm internally applies limits in module, inclination and azimuth to maintain the limits set by the user in body frame (body frame limits will be turn into spherical limits). The user also can selected directly **angular rates** option (6) which allows him set limits in Heading rate and Flight Path angles rates.



Acceleration limit (Spherical frame)

The second derivative of velocity (jerk) imposes another limit in acceleration. It modifies the behaviour of the vehicle. Depending on values, it's possible to get more smoothness during guidance. Algorithm ensures that when desired velocity is reached the acceleration value is near 0. As acceleration limits, user can set 6 values (3 for rising limit and 3 for failing limit).



Acceleration limit (Cartesian frame)

The effects of these limitations are explained below.

First, the acceleration limits are disabled. The stick that controls Thrust is moved and desired velocity change according to this stick command. In Desired velocity chart we can see this effect and in bottom acceleration chart is shown how acceleration is not limited (high values reached).

Only velocity limit (Thrust)

Now, a limit in acceleration bottom axis is set to 0.1. Now the desired speed grows with a lower slope due to the imposed limitation. Also in the acceleration bottom chart we can see how the value oscillates within the imposed limit.

Acceleration limit (Thrust)

To compare acceleration and jerk, roll axis is chosen. In the first gif below only the first limit is applied.

Acceleration limit (Cartesian frame)

When the jerk limit is also enabled we can see how acceleration (in Y Body axis) does not show peaks, and changes in desired velocity are smoother.

Acceleration limit (Cartesian frame)

7.2.4.5 Modes

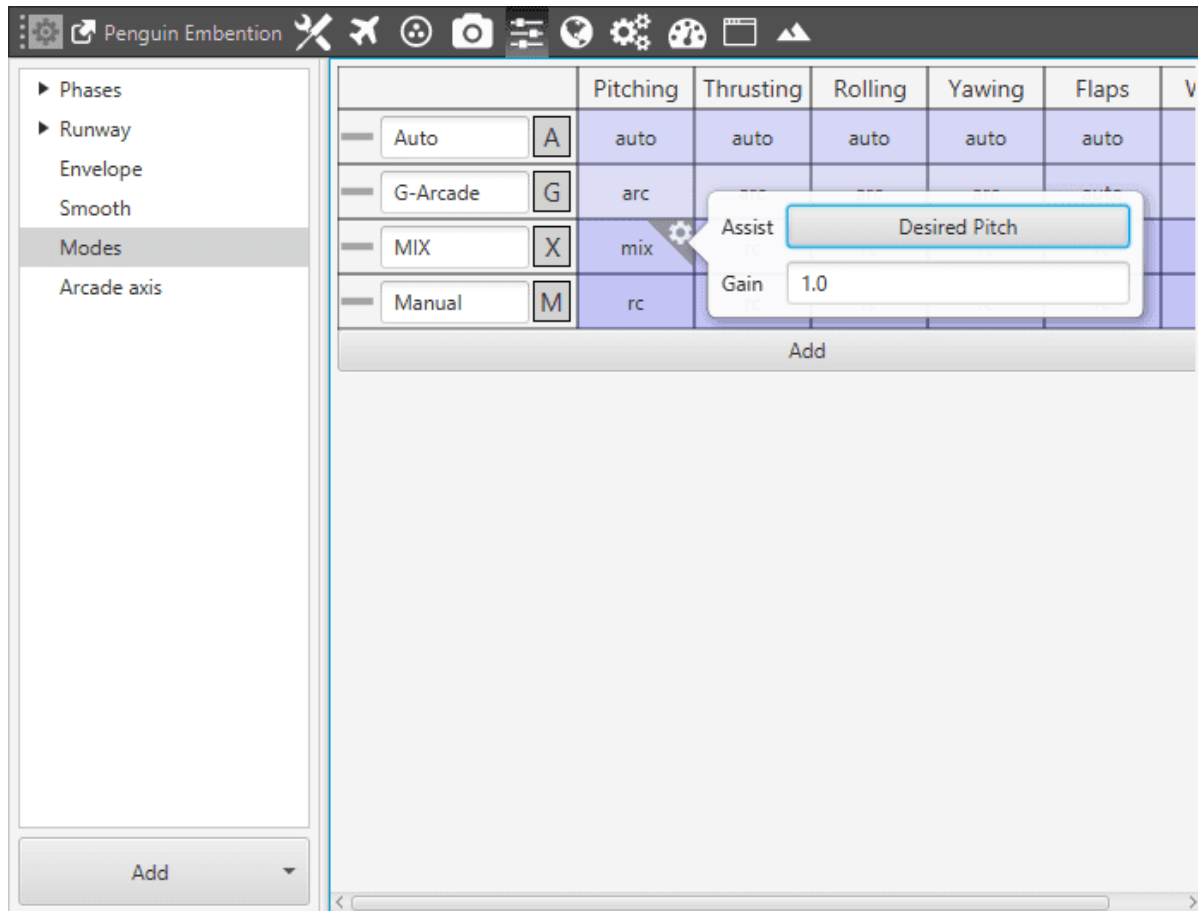
The flight modes determine who is in charge of controlling each one of the aircraft control channels. There are 5 different control modes and it is possible to combine them to create custom flight modes.

| | | Pitching ... | Thrustin... | Rolling (...) | Yawing (...) | Pitching ... | Thrustin... | Rolling (...) | Yawing (...) |
|-----|----------------------|--------------|-------------|---------------|--------------|--------------|-------------|---------------|--------------|
| — | Auto A | auto | auto | auto | auto | auto | auto | auto | auto |
| — | Manual M | rc | rc | rc | rc | auto | auto | auto | auto |
| — | G-Arcade G | auto | auto | auto | auto | arc | arc | arc | arc |
| — | Semi-Arcade S | auto | auto | auto | auto | arc | rc | arc | arc |
| Add | | | | | | | | | |

Modes editing panel

The options available are:

- **Automatic:** the control channel is controlled totally by the autopilot.
- **RC:** the control is totally carried out manually. The movements on the pilot stick imply directly movements on the servo linked to that control channel.
- **ARC:** the autopilot aids the radio controller during the flight, i.e it could be considered as a mix between automatic and manual. The movements on the pilot stick are the input values on the control system, so the pilot commands a desired pitch, roll, IAS, heading and so on, and is the control system who is in charge of making the platform follow those commands.
- **Mix:** in this mode, it is possible to select in which step of the controller will enter the pilot command. For example, the pitching of an aircraft is commonly controlled with 3 PID being: flight path angle, pitch and pitch rate. In the arcade mode the pilot command will be a desired flight path angle that enters as input of the whole control system, but in the Mix mode is possible to select where we want the command to enter, so the pilot command could be pitch (entering in the second PID directly) or pitch rate (entering directly on the third PID). The control system will take this input as a disturbance that it wants to discard because the final objective is to match the input of the first PID (a desired flight path angle in this case), so the Mix mode can be used to make small corrections when the aircraft is following a route for example, where we want it to move slightly towards a certain direction by introducing a value directly on the roll PID. The following figure shows how to select the Mix mode and the parameters that have to be configured: the variable to control (Desired Pitch in this case) and the gain applied on the stick command (it goes from -0.5 to 0.5 so in this case, a gain of 1 implies a Desired Pitch that can go from -0.5 rad to 0.5 rad).



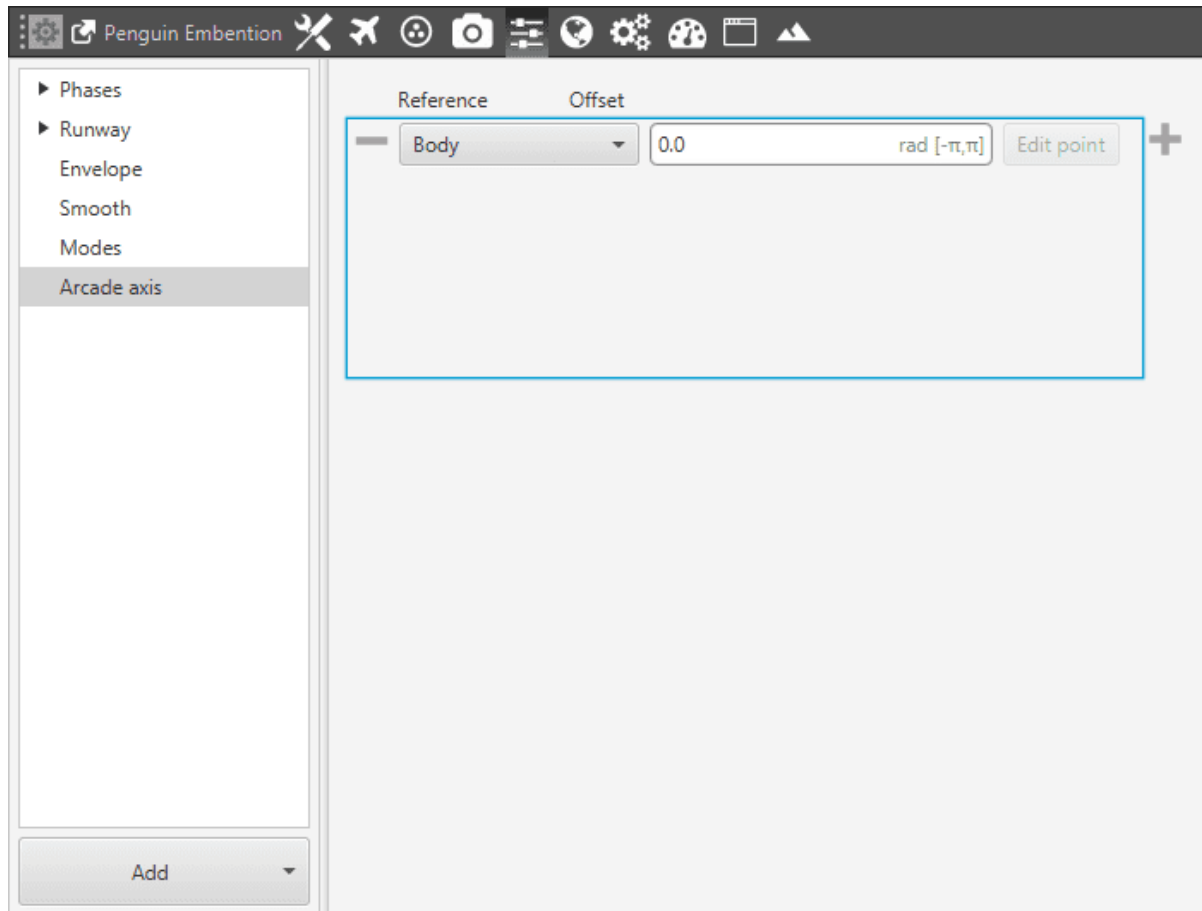
MIX Mode

- **Autotune:** this mode has to be select when the user wants to find control gains automatically.

It is every common to find an automatic mode where all the dynamics are controlled by the autopilot. Likewise, the manual mode is completely controlled by the remote controller (**rc**). To change any of this options, click on the cell you would like to change and the next option will be set.

7.2.4.6 Arcade Axis

The Arcade Axis menu enables the option of changing the center of the system axes. This option is useful, for example, when the pilot wants all movement to be made with respect him (Ground axes). In this way, if the pilot command a turn right, the aircraft will turn to the right of the pilot, instead the right of the aircraft (Body axes).



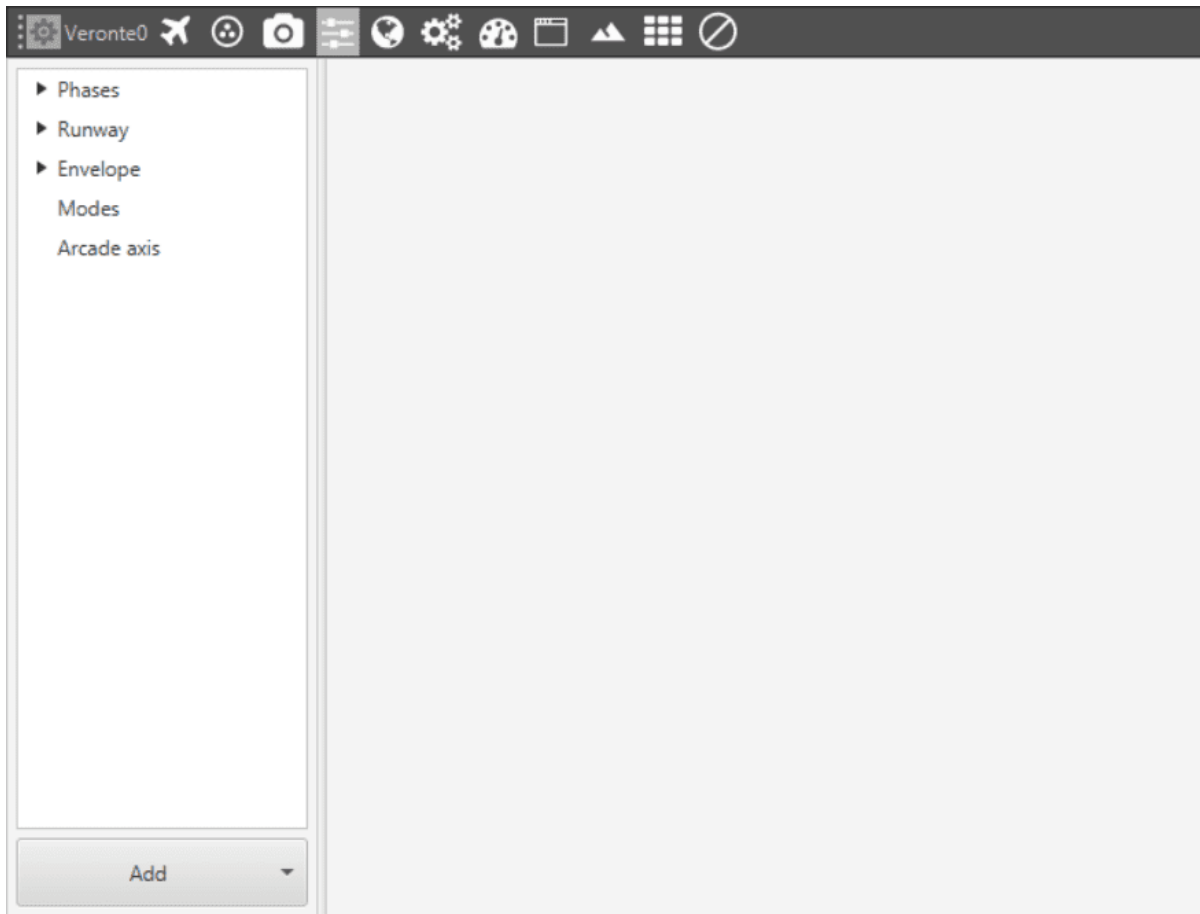
Arcade Axis Configuration Menu

It is possible to add as many axes system as desired, being able to choose between the following types:

- **Body:** fix the axes in the UAV. It is standard for the pilot.
- **Ground:** fix the axes in the Veronte GND.
- **Point:** fix the axes in a point that user defines.

An automation can be used to select an Arcade Axis in flight, see section [Select Arcade Axis](#).

In this panel all the parameters related to the control of the platform can be found. There are 5 sections, each one showing a different menu of configuration.



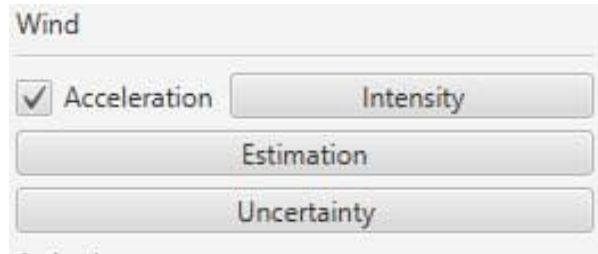
Control Configuration Menu

- **Phases:** in this section are created the Flight Phases that will control the aircraft at different stages of the operation. This section includes the Guidances that are necessary, the type of Control used for each phase, the Arcade configuration when is used and the TC Panel for configuring in-flight parameters.
- **Runway:** this section allows user to configure a Runway or Spot used during the flight operation.
- **Envelope:** here are defined the flight limits, stall speed and the maximum climb and descent angles.
- **Modes:** allows the creation of custom flight modes, where each one of the control channels can be assigned a mode individually.
- **Arcade Axis:** this option is used to create axis systems referred to a certain point or direction.

7.2.5 Navigation

7.2.5.1 Wind Estimation

From the Navigation window can be configured the parameters that affects the wind estimation algorithm.



Wind

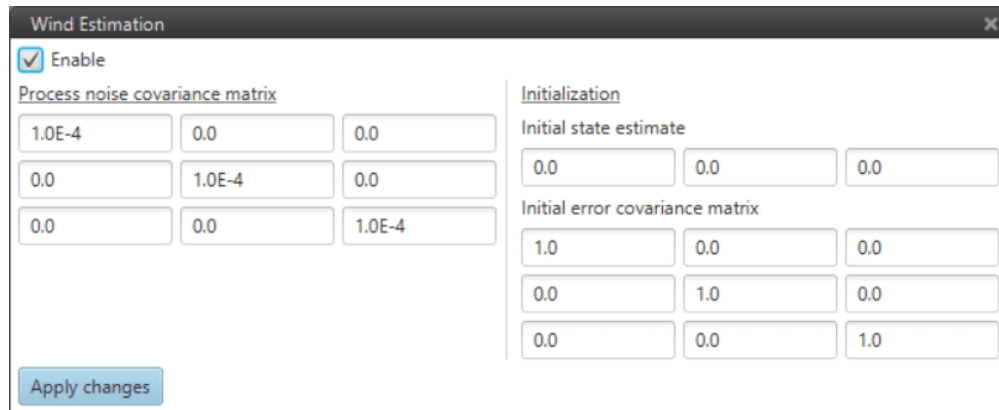
☒ Acceleration Intensity

Estimation

Uncertainty

Wind estimation parameters

For proper estimation, the system needs to gather as much information of the wind as possible so missions with a trajectory involving changes on the of directions will result in better wind estimation compared with a straight trajectory mission.



Wind Estimation

☒ Enable

Process noise covariance matrix

| | | |
|--------|--------|--------|
| 1.0E-4 | 0.0 | 0.0 |
| 0.0 | 1.0E-4 | 0.0 |
| 0.0 | 0.0 | 1.0E-4 |

Initialization

Initial state estimate

| | | |
|-----|-----|-----|
| 0.0 | 0.0 | 0.0 |
|-----|-----|-----|

Initial error covariance matrix

| | | |
|-----|-----|-----|
| 1.0 | 0.0 | 0.0 |
| 0.0 | 1.0 | 0.0 |
| 0.0 | 0.0 | 1.0 |

Apply changes

Wind Estimation (Process covariance matrix, Init state estimate, Init error covariance matrix)

The computed result is displayed in the variables: **Wind Velocity Down, Wind Velocity East, Wind Velocity North.**

Warning: The values that appear here should only be changed by advanced users.

This menu contains the parameters used in the Kalman Filter algorithm to fuse the information provided by the different sensors. This data is used in the navigation system to generate the commands sent to the aircraft.

The screenshot displays the Veronte Autopilot configuration window. The top bar shows the name 'Sandra' and several icons. The main interface is divided into several sections:

- State Vector:** Includes a 'GPS ok time to restore restricted mask' field set to 5.0. Below it are checkboxes for Position, Speed, Attitude, Bias Accelerometer, Bias Gyroscope, and Qdem. A 'Wind' section has a checked 'Acceleration' box and an 'Intensity' button. Below that are 'Estimation' and 'Uncertainty' buttons.
- Attitude:** Features a 'Magnetometer' dropdown set to '3D'.
- Filter parameters:** Contains fields for Beta (0.025 rad/s), Zeta (0.003 rad/s²), Initial (10.0 rad/s), and Initial (0.0 rad/s²). There are 'Advanced' and 'Steps' (50) buttons, and an 'Initial rains covariance' button.
- Navigation:** A dropdown menu set to 'Internal'.
- Accelerometer:** Fields for Qnfb (12.0) and Qdfb (4.0E-5) in three columns.
- Gyroscope:** Fields for Qnwb (0.12) and Qdwb (4.0E-6) in three columns.
- Qdem:** A 'Value' dropdown set to 'Irregular'.
- Angular speed estimation filter:** A 'Custom' section with a list containing 1.0 and -1.0, and an 'ADD' button.

Navigation Parameters

Veronte integrates a navigation system which can operate with GPS and without GPS coverage. In the navigation with GPS, the system uses it to make the aircraft fly a route or towards a certain waypoint. It is possible to control the aircraft position (longitude and latitude) and the altitude. This is the navigation used by default, the one that the system uses when everything is working properly.

In case the GPS signal is lost, the navigation can easily measures the attitude angles with a greater precision than using a simple IMU. With these measures, it is possible for the system to control the pitch, roll and yaw and then keep a safe attitude when the GPS signal is lost, avoiding any possible malfunction. It is recommended to create an automation to change to a phase where the attitude angles are controlled, in the case of a loss of GPS signal. For more information visit [Automations](#).

Note: the yaw can be measured in the navigation without GPS only if the magnetometer is activated in the navigation window.

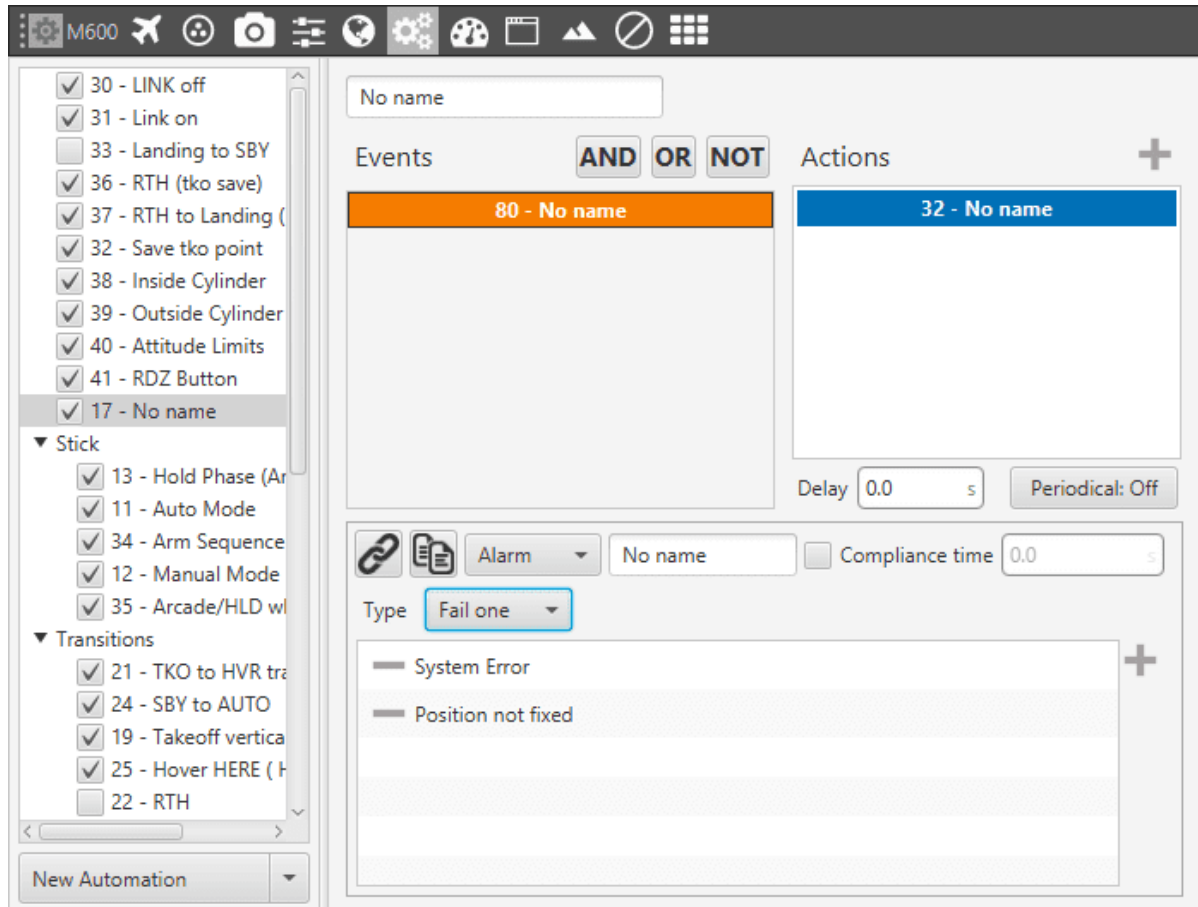
Warning: The values that appear here should only be changed by advanced users. If you are not familiar with the Kalman Filter algorithm and Sensor Fusion do not change the default parameters.

7.2.6 Automations

7.2.6.1 Events

7.2.6.1.1 Alarm

This kind of automation allows the user to add any system bit. Depending on the mode in which it is configured, it will be triggered in one way or another.



Event – Alarm

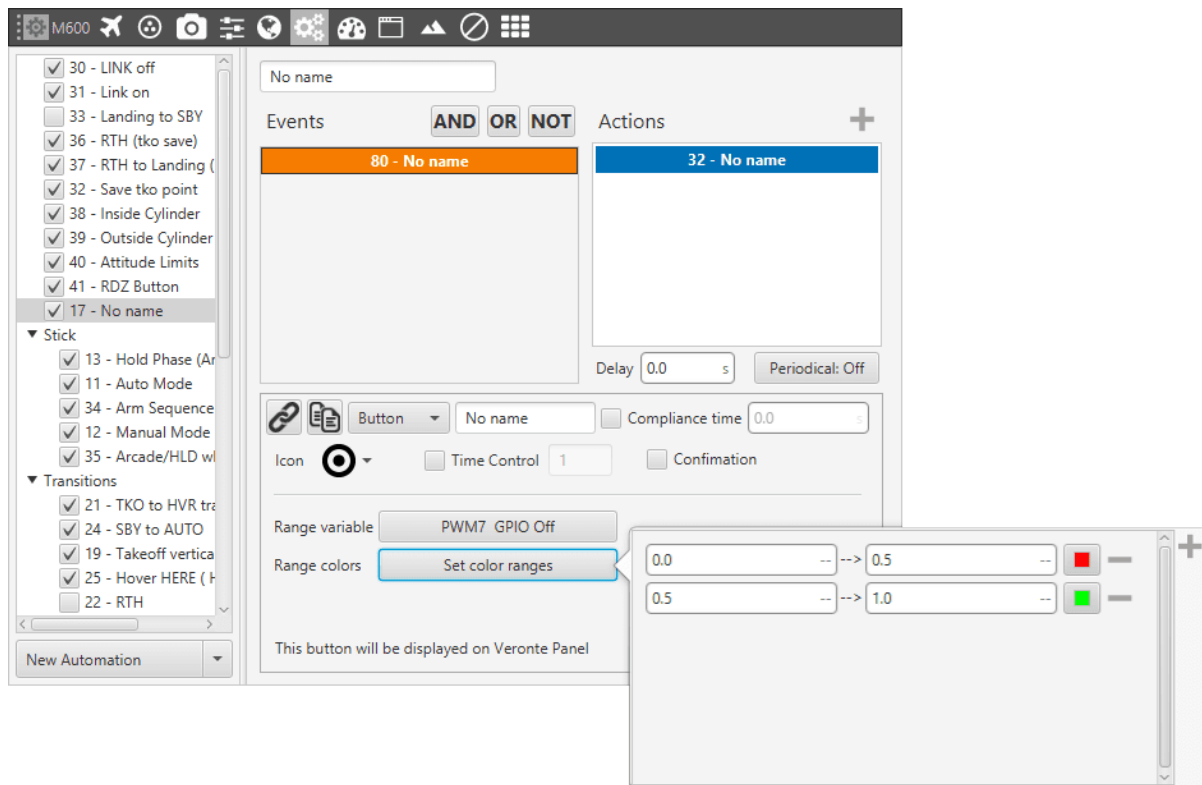
The two possible modes are the following:

- **Fail one:** it is triggered when one of the bits is set to false.
- **All ok:** it is triggered when all bits are set to true.

A common alarm event is the **Position not fixed** in fail one mode, which is triggered when there is not GPS signal in the autopilot.

7.2.6.1.2 Button

This option creates a button that will trigger the event when it is clicked.



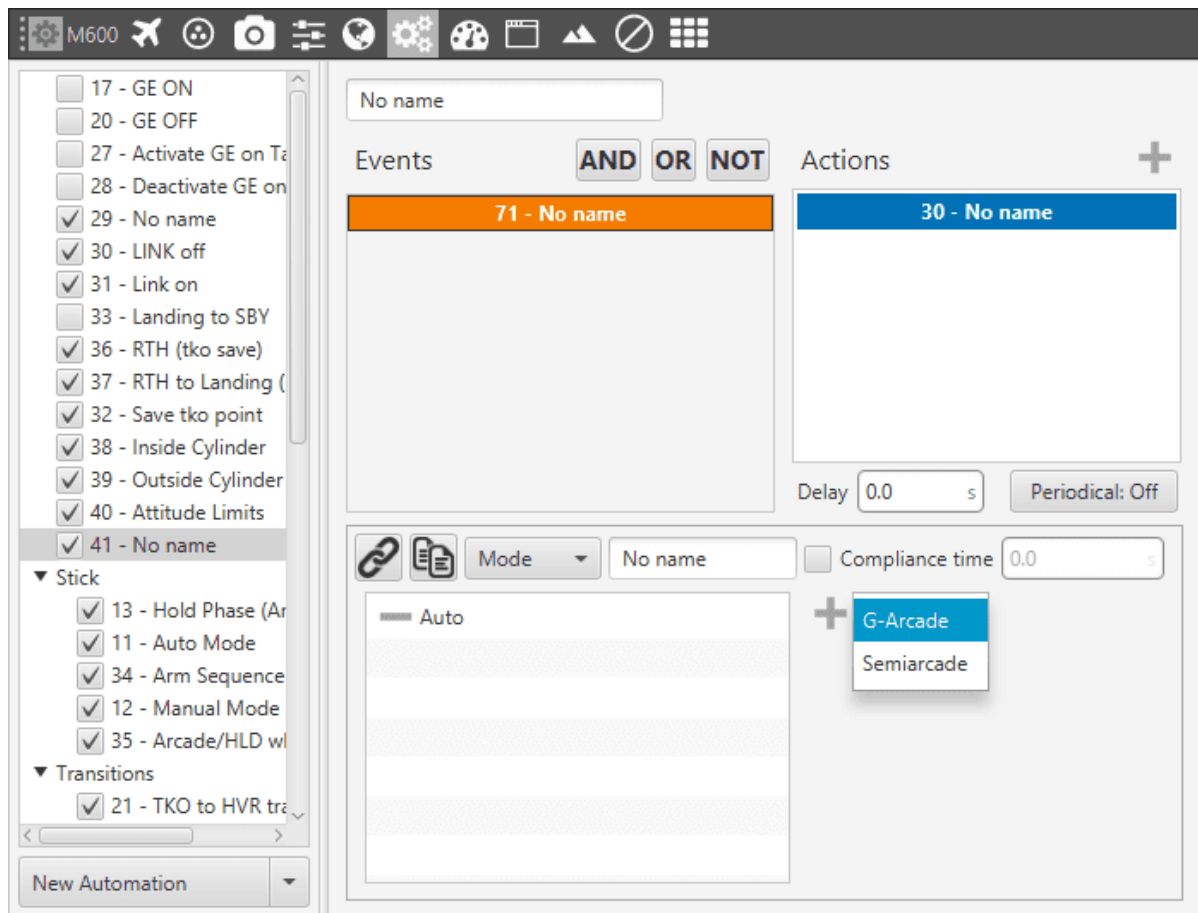
Event – Button

- **Range variable** and **range colors** options are used to make the button change its color according to the value of a variable. To do that, select a variable and then indicate as many points as desired, each one with its corresponding value and color.
- **Confirmation** option will display a pop-up window that asks for confirmation after pushing the button, so it is a safety measure.
- **Time Control** option is used to trigger the action when the button is being pushed during the time specified in this option.

If a button event triggers an action that consists of a change to a determined phase, the button will be the one of the Veronte Panel with the name of that phase on it. If the button event is linked to a different action (servo movement, variable...), it will appear in the lower part of Veronte Panel with the Icon selected by the user.

7.2.6.1.3 Mode

The event is triggered when the aircraft is in one of the modes selected.

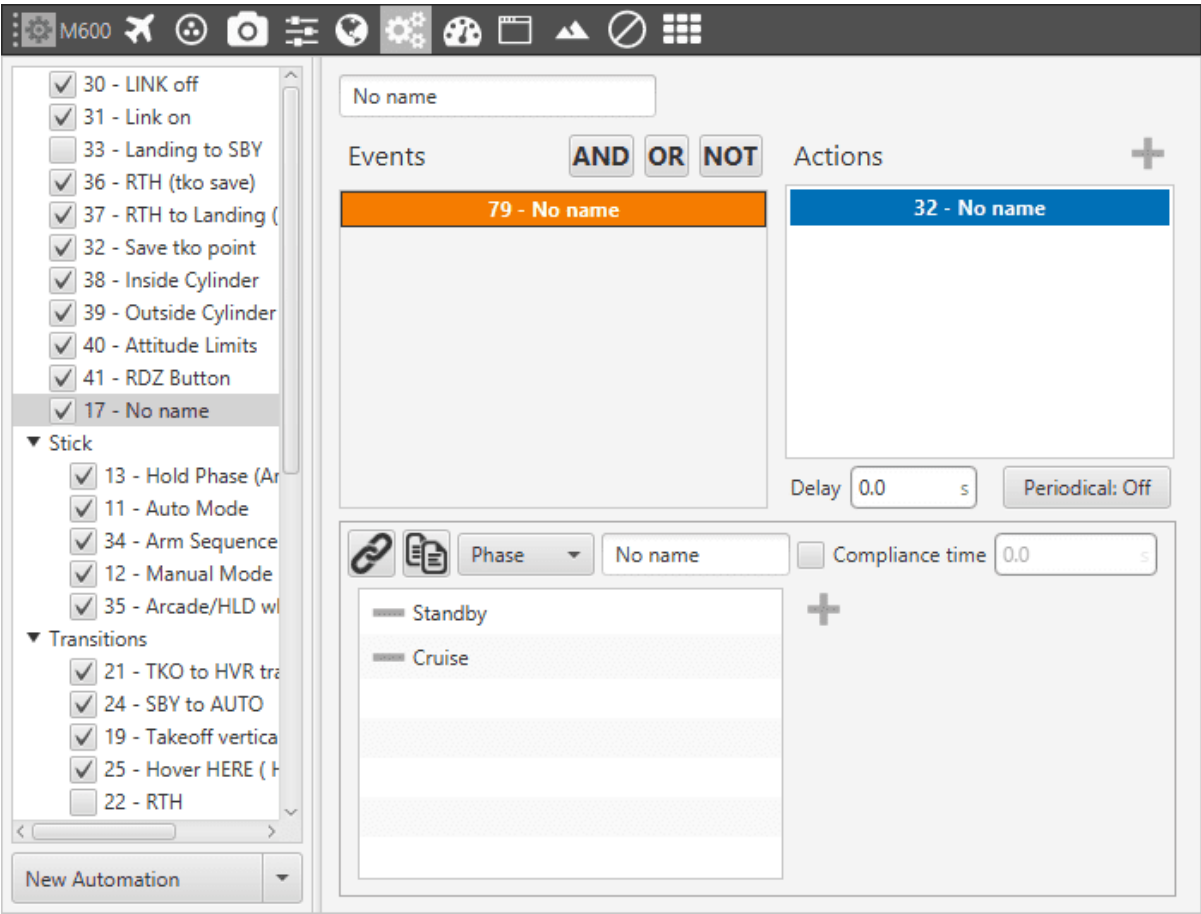


Event – Mode

These modes have been created previously. See section [Modes](#), for more information about creating modes. A **Compliance time** can be set for this kind of event.

7.2.6.1.4 Phase

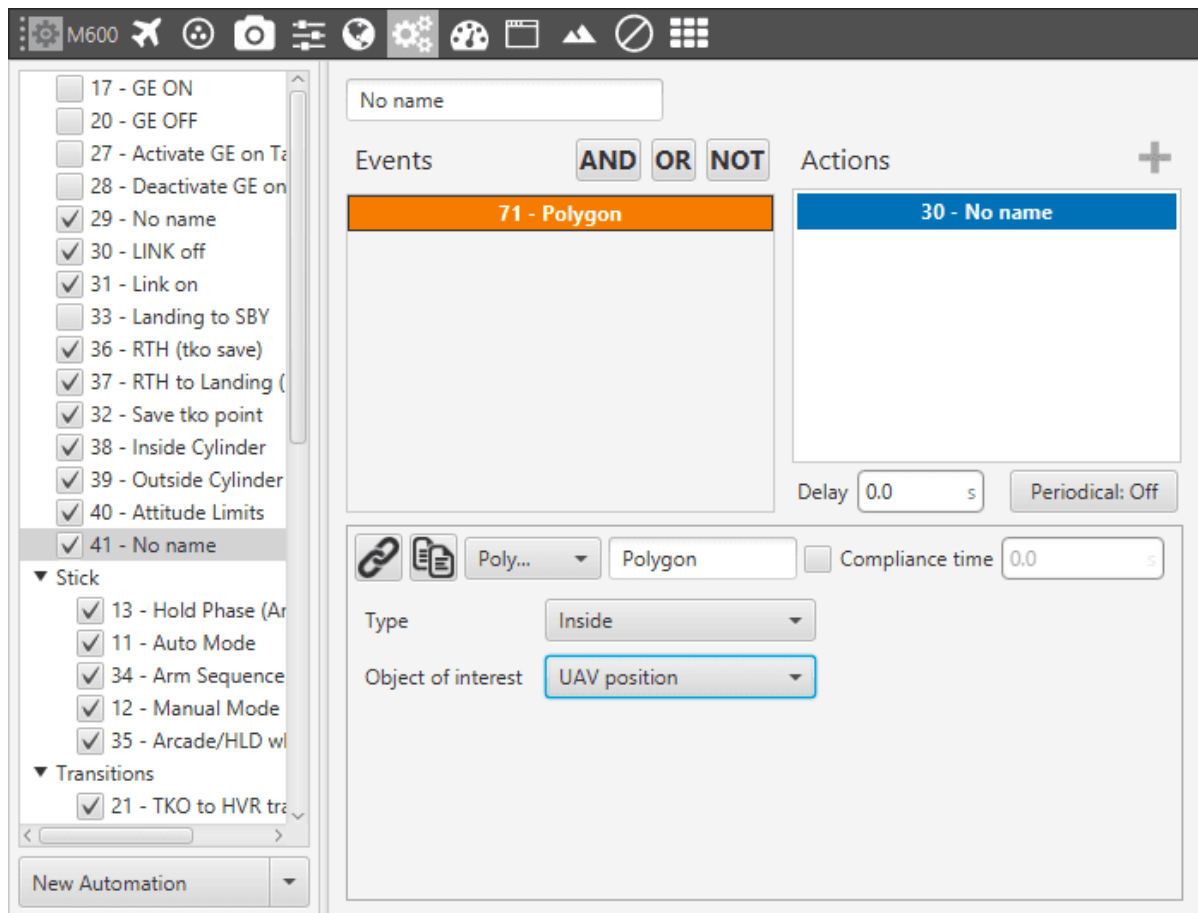
The event is triggered when the aircraft is in the phases selected by clicking on the “+” button, not in all of them at the same time, being in one of them is enough to trigger the action.



Event – Phase

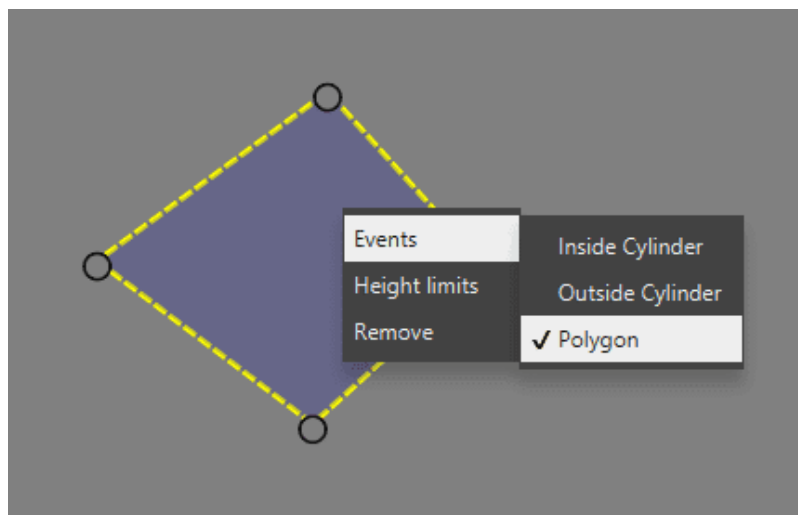
7.2.6.1.5 Polygon

The event is triggered when the aircraft is inside or outside a polygon defined in the *Mission menu*.



Event – Polygon

- When the event has been labeled (“Event Name (Polygon)” in this case) and saved, it is then possible to link it to a polygon drawn on the map in the *Mission menu*.
- Right-clicking on the polygon will open a menu, and in *Events* will appear all the polygon-type events defined on the system. Click on it to link the polygon and the event.



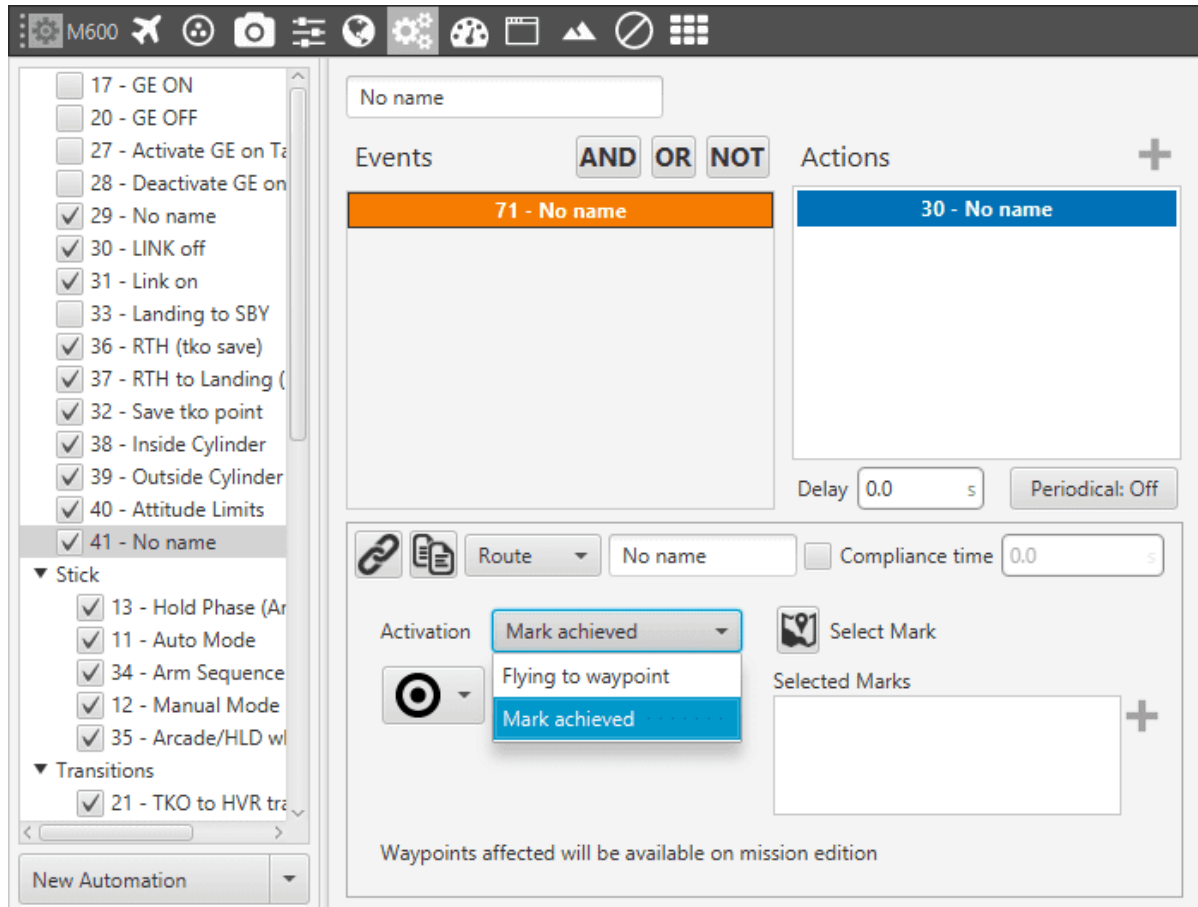
Event – Polygon Selection

Object of interest allows the user to select which object is the one that should fulfill the event.

An example about using this automation can be found in [Circular Area](#).

7.2.6.1.6 Route

This event is related with the waypoints defined by the user in the [Mission menu](#) and the marks defined in [Mission setup](#).



Event – Route

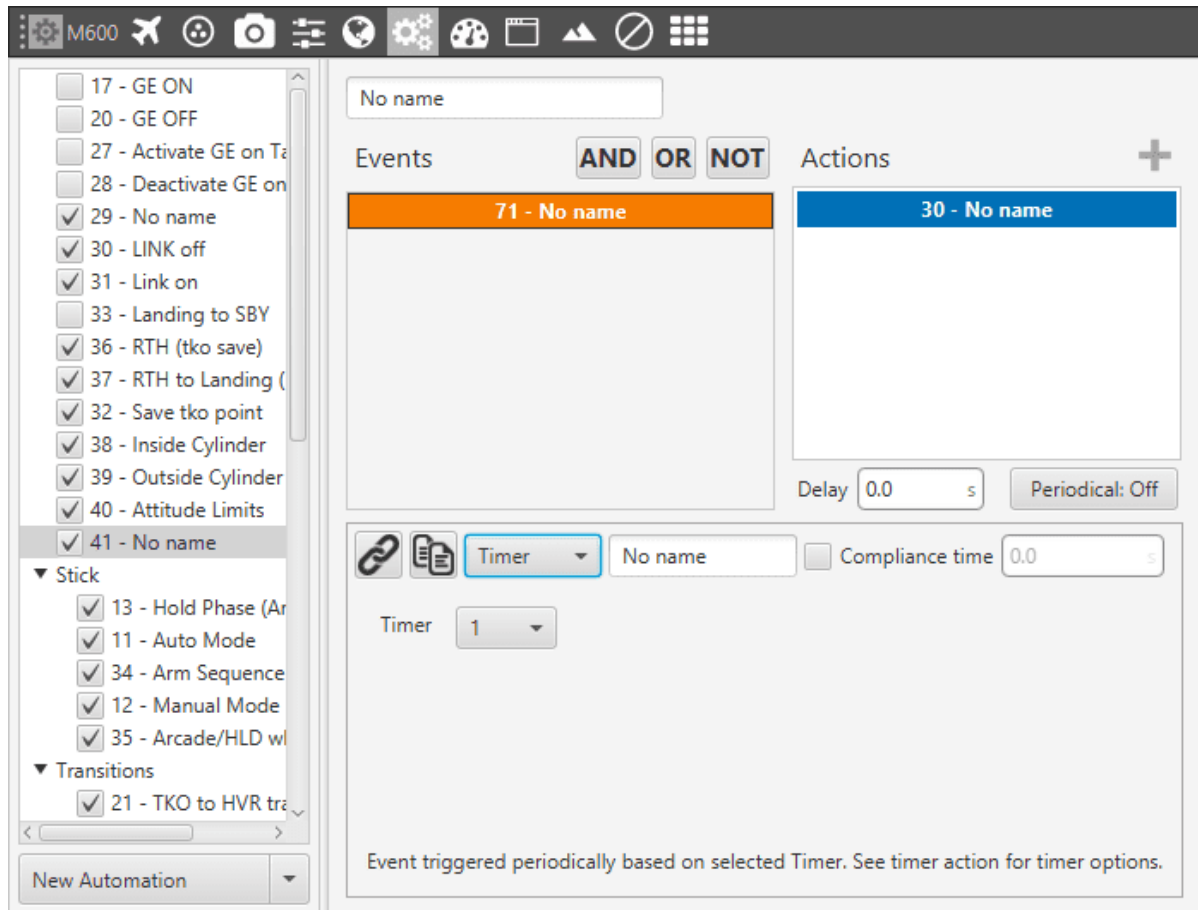
There are two modes in the event type.

- **Mark achieved:** triggers the action when the vehicle has reached the selected mark.
- **Flying to waypoint:** triggers the action when the platform is flying towards that waypoint (is the next waypoint of the route).

Clicking on **Select Mark/Waypoint** allows the selection of the mark/waypoint among the ones created by the user. It is possible to change the appearance of the waypoint to an image selected in the **icon** option, so the user can identify easily the waypoint linked to that automation.

7.2.6.1.7 Timer

This event will check the status of the timer selected in the menu. That timer should have been configured before on the action side of another automation ref (Action type *Periodical*).

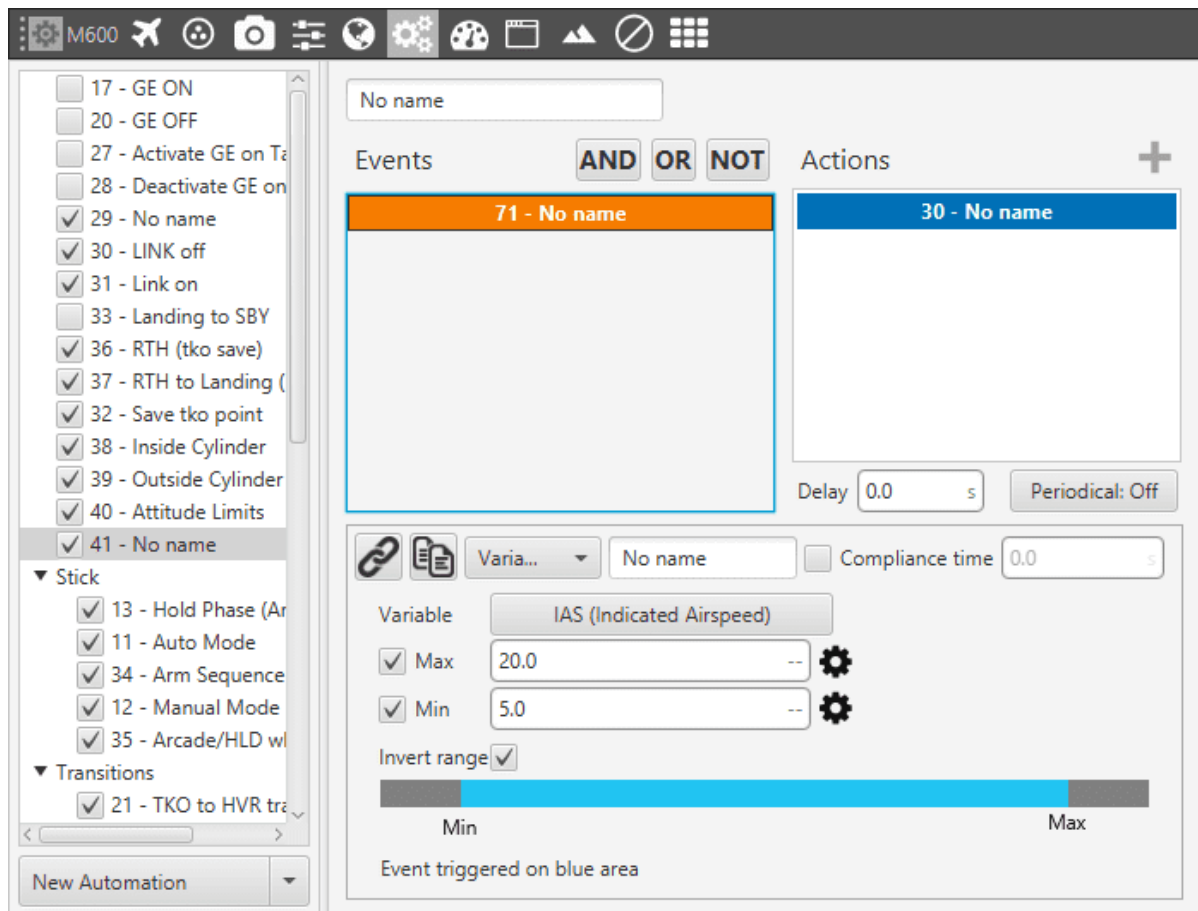


Event – Timer

For example: if it is desired to take a photo 10 seconds after the takeoff, one automation should have the event of Phase take off, with the correspondent *Periodical* action that will start a timer that lasts 10 seconds. Then with another automation, indicating in the event the timer created, an action is created to take a photo when the timer event is triggered. In Timer is selected the number that identifies the timer that is evaluated in this event.

7.2.6.1.8 Variable

This event is triggered when a variable selected is between a range established.



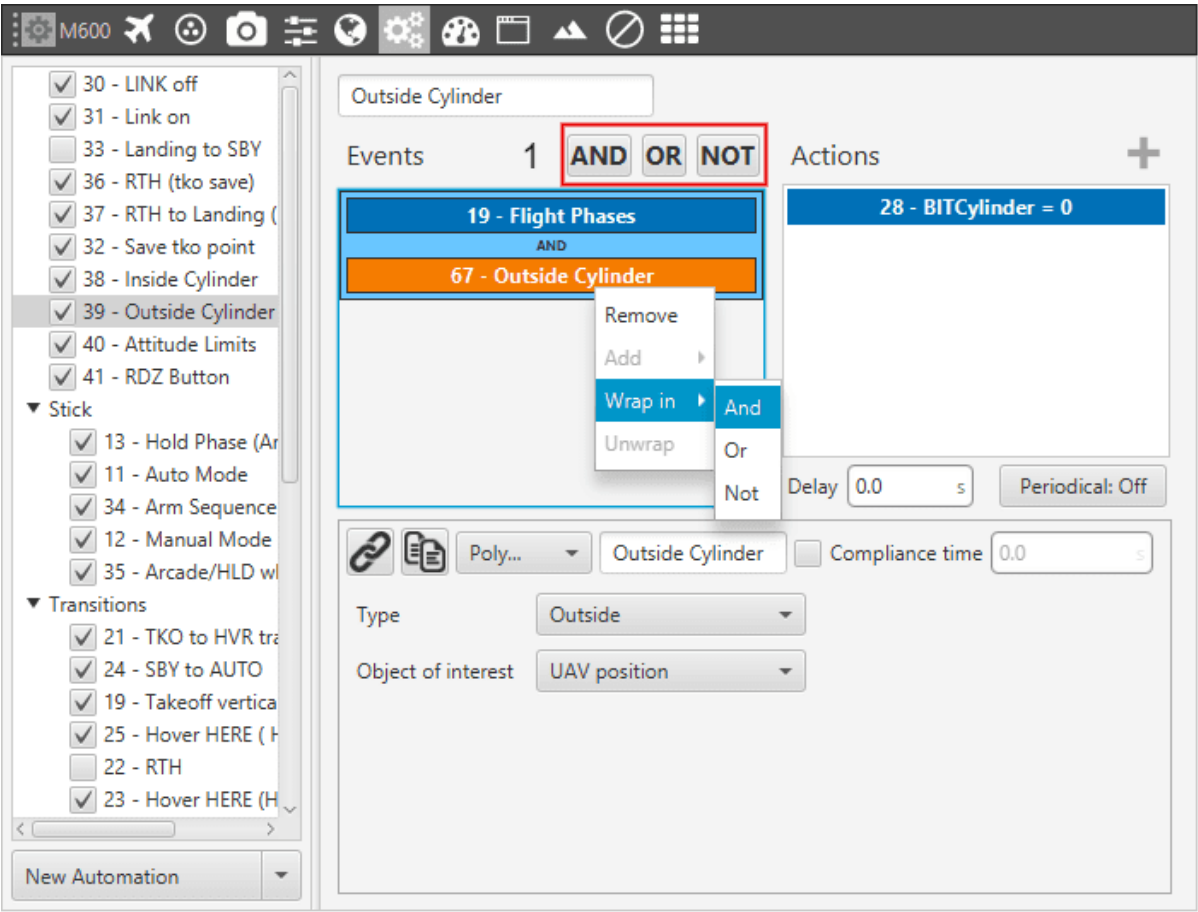
Event – Variable

- The variable to be evaluated is selected on **Variable**.
- **Max-Min**: maximum and minimum values of the threshold are established here. Custom threshold can be established by clicking on the gear icon.
- The option **invert range** will change the interval (the blue area will be gray, and the gray one will be blue).

As an example consider the event of the figure. With that parameters, the event is triggered when the IAS is between 5 and 20 meters per second. If the **invert range** option is unchecked, the event will be triggered when the IAS is lower than 5 m/s or greater than 20 m/s.

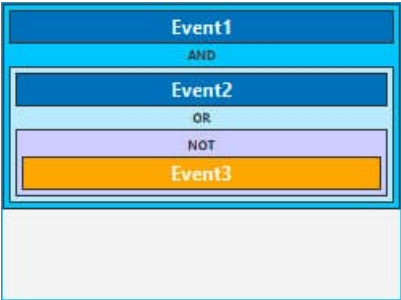
An event is something that has to be accomplished to trigger the actions. All the events can be combined to create a custom event, using the boolean operations provided by the software (AND, OR, NOT).

When entering a new Event or Action it is possible to choose from one of the previously created on the system or to create a new one.



Automations Panel – Events

When there is only one event, clicking on the boolean command (1) will create another event linked to the other one according to that operation. Right-clicking on an event and selecting **Wrap in** allows the creation of an operation as if it was inside brackets, i.e it will be evaluated first. Let's consider the following event group as an example.



Events

The first operation that is evaluated is the NOT, then the OR between Event2 and the result of the NOT, and finally the AND between Event1 and the result of the OR.

The following table depicts the meaning of each one of the boolean operators.

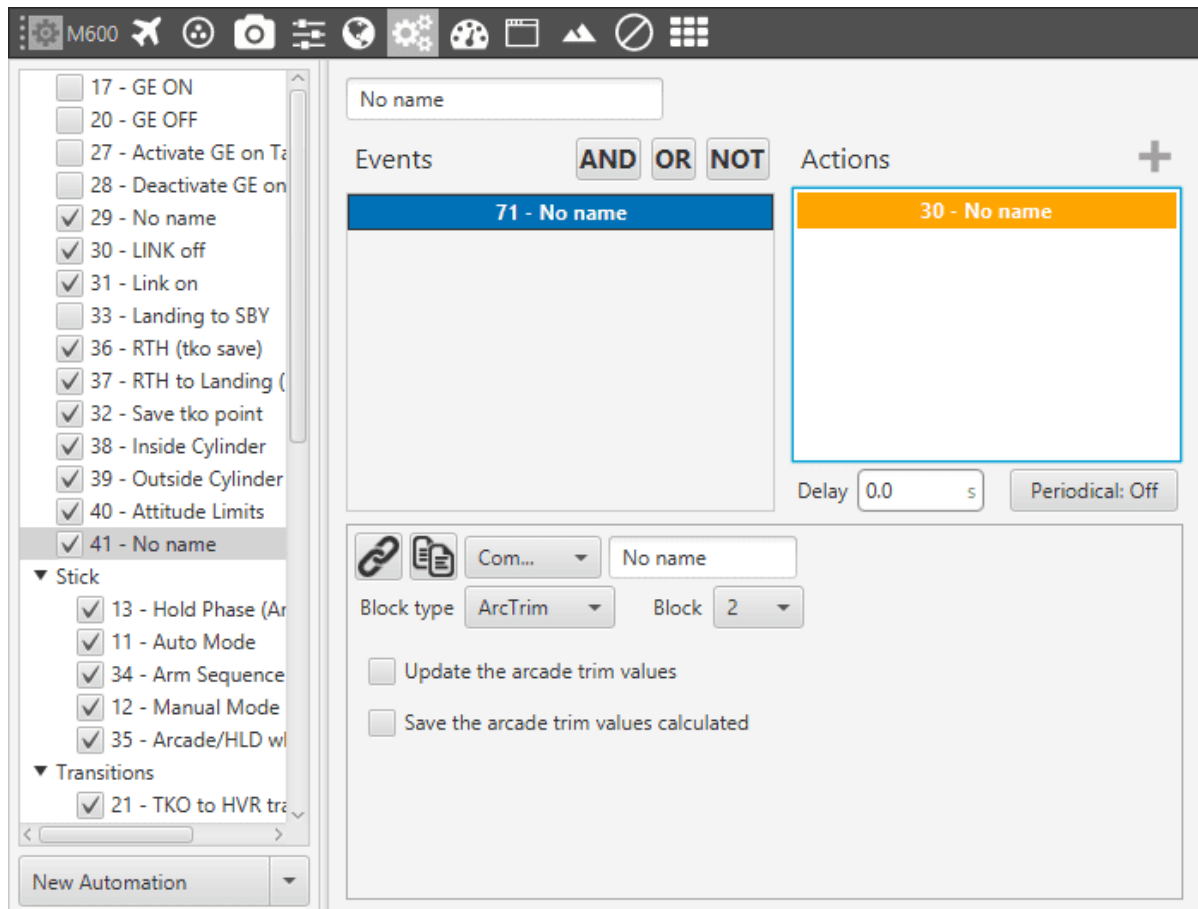
| Logics | Description |
|--------|---|
| AND | All events grouped on an AND should be accomplished simultaneously in order to activate the automation. |
| OR | One of the events in the group should be accomplished for activating the automation. |
| NOT | The event will be active meanwhile the event or event group is not accomplished. |

When creating a new event, it is possible to select different types of events, these are explained in the next sections.

7.2.6.2 Actions

7.2.6.2.1 Arcade Trim

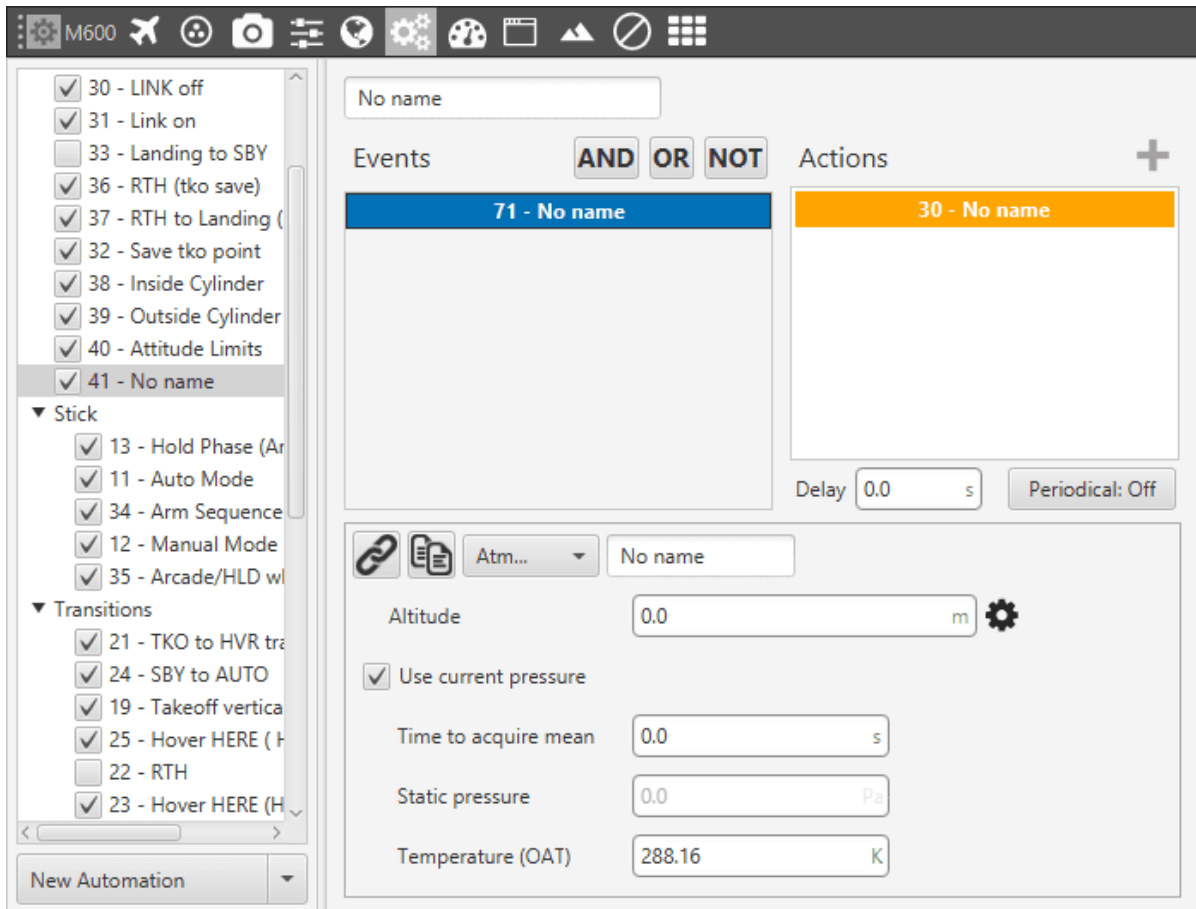
This action trims the radio controller, i.e. sets as zero the current sticks positions. As seen in the picture below, it consists of simply creating a **Button** on the **Events** side and adding an **Action** which is called *Arcade trim*. The latter action is already configured to copy the current stick position into the trim vector.



Action – Arcade Trim

7.2.6.2.2 Atmosphere Calibration

This action allows the atmosphere calibration in the same way as shown in *Quick Commands*.



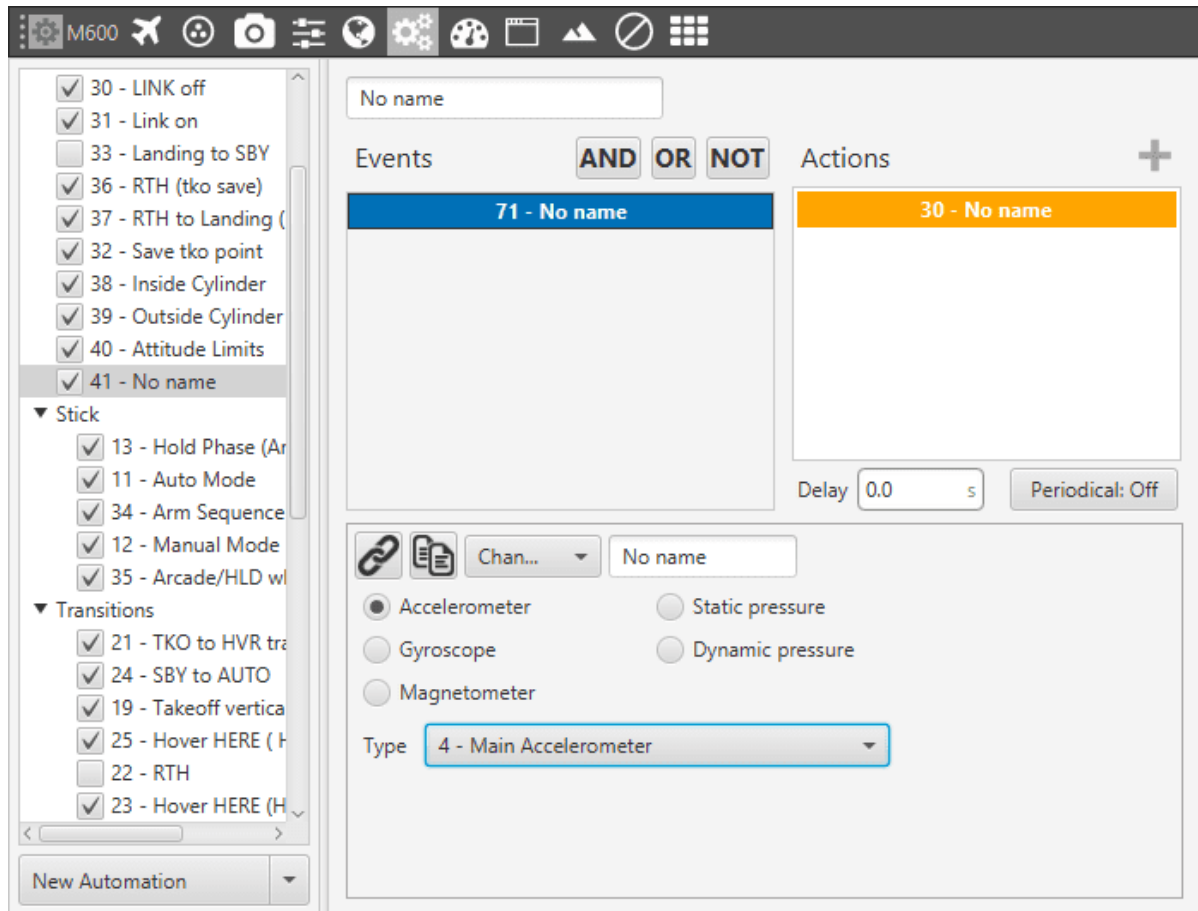
Action – Atmosphere Calibration

The following options can be configured:

- **Altitude:** actual MSL altitude.
- **User current pressure:** if this box is ticked, static pressure is read from the static pressure sensor during the specified time. Otherwise, actual static pressure should be specified manually.
- **Temperature (OAT):** outside air temperature.

7.2.6.2.3 Change Active Sensor

This option allows the change of the actual sensor used as accelerometer, gyroscope, magnetometer, static pressure and dynamic pressure; between the internal and external ones.



Action – Change Active Sensor

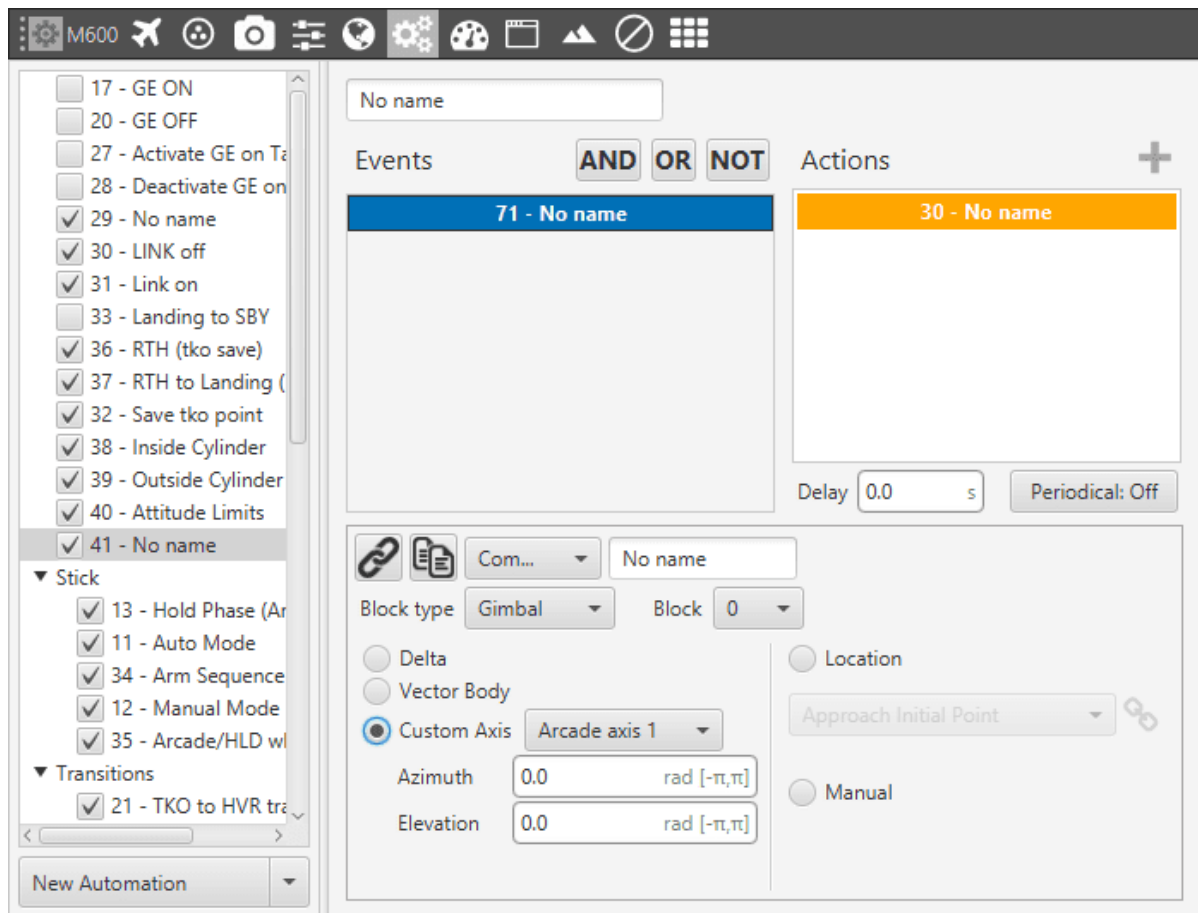
7.2.6.2.4 Command Block

This action allows the user to configure gimbal control or trim the radio controller.

7.2.6.2.4.1 Gimbal

Note: This action is disabled by default when Veronte is started. To activate it, the user have to run the action.

When this action is triggered, the gimbal control is enabled. There are several control modes that are explained below.

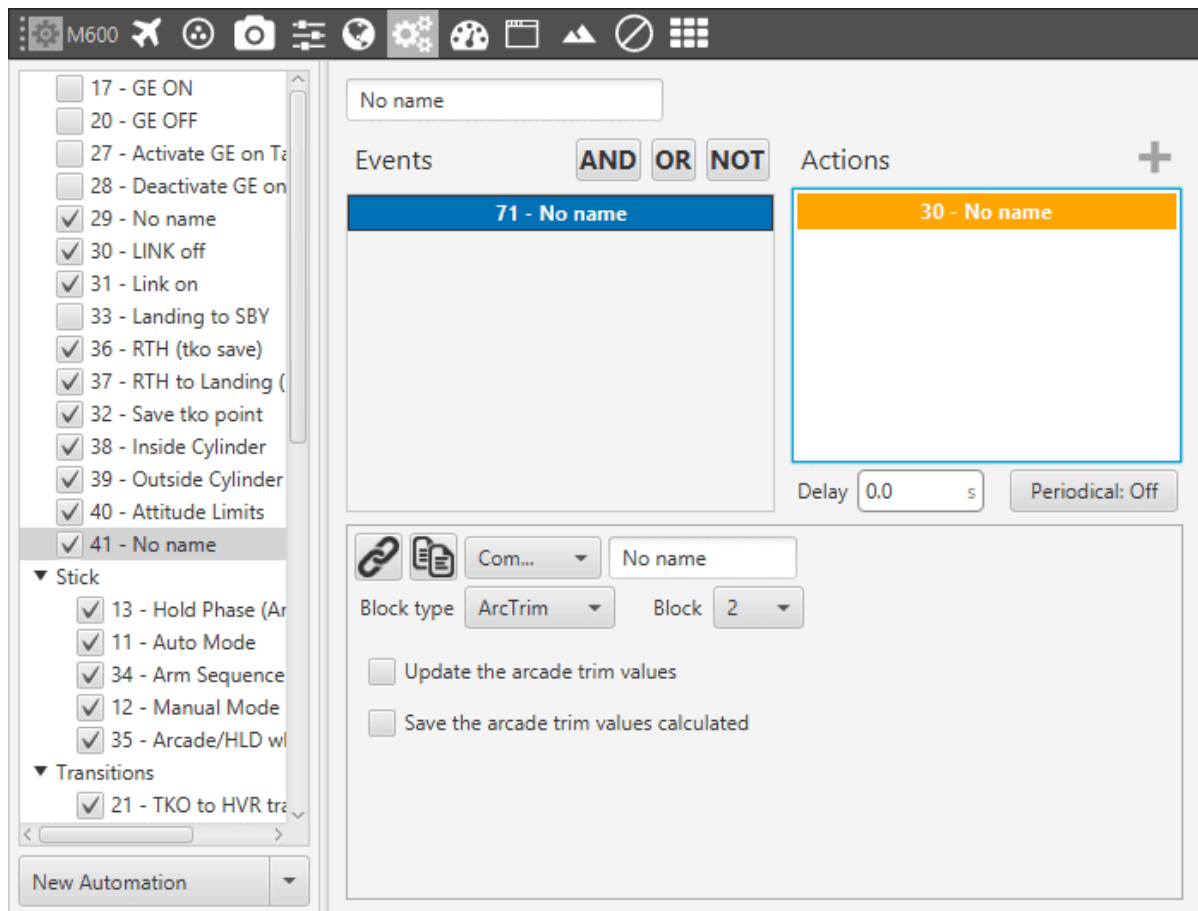


Action – Gimbal

- **Delta:** control using NED axis, defining the initial position through azimuth and elevation.
- **Vector body:** this control uses aircraft body axis. The initial position is defined through roll and tilt.
- **Custom Axis:** in this case the axis should have been defined in *Arcade Axis*.
- **Location:** the gimbal will point towards the projection on the ground at the specified point.
- **Manual:** gimbal control is done externally, e.g. via VCP commands.

7.2.6.2.4.2 ArcTrim

This action trims the radio controller, i.e sets as zero the current sticks positions.

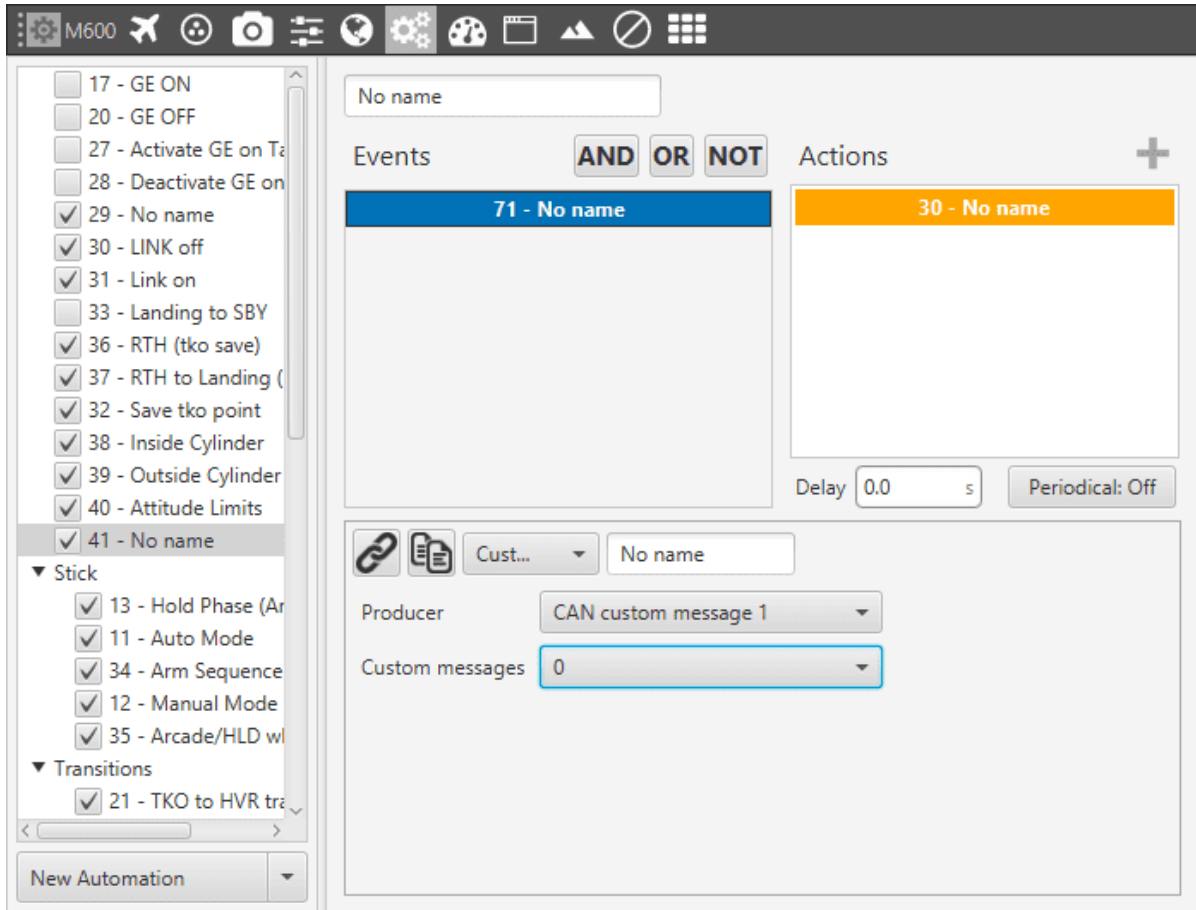


Action – Arcade Trim

- **Update the arcade trim values:** if this option is checked, the stick is trimmed but not saved in the configuration. This means that if Veronte is restarted the trimming is lost.
- **Save the arcade trim values calculated:** trim values are stored for future flights.

7.2.6.2.5 Custom CAN TX

When this action is triggered, a previously configured CAN message is sent through the CAN bus. The message has to be configured in *Custom messages* section.



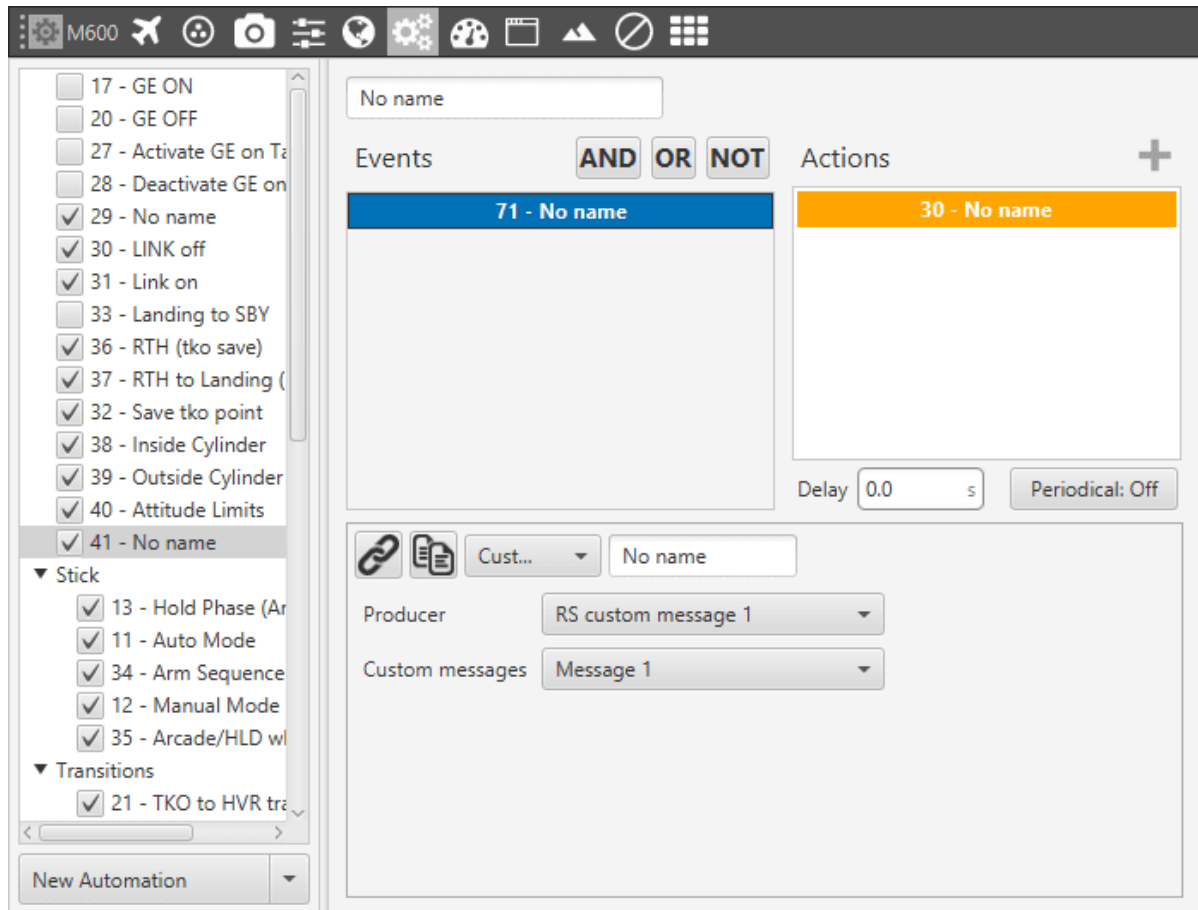
Action – Custom CAN TX

The two parameters to configure in this action are:

- **Producer:** where is located the custom message: CAN custom message 1 or 2.
- **Custom message:** the one that will be sent.

7.2.6.2.6 Custom Serial TX

When this action is triggered, a previously configured serial message is sent through the serial port (RS232 or RS485). The message has to be configured in [Custom messages section](#).



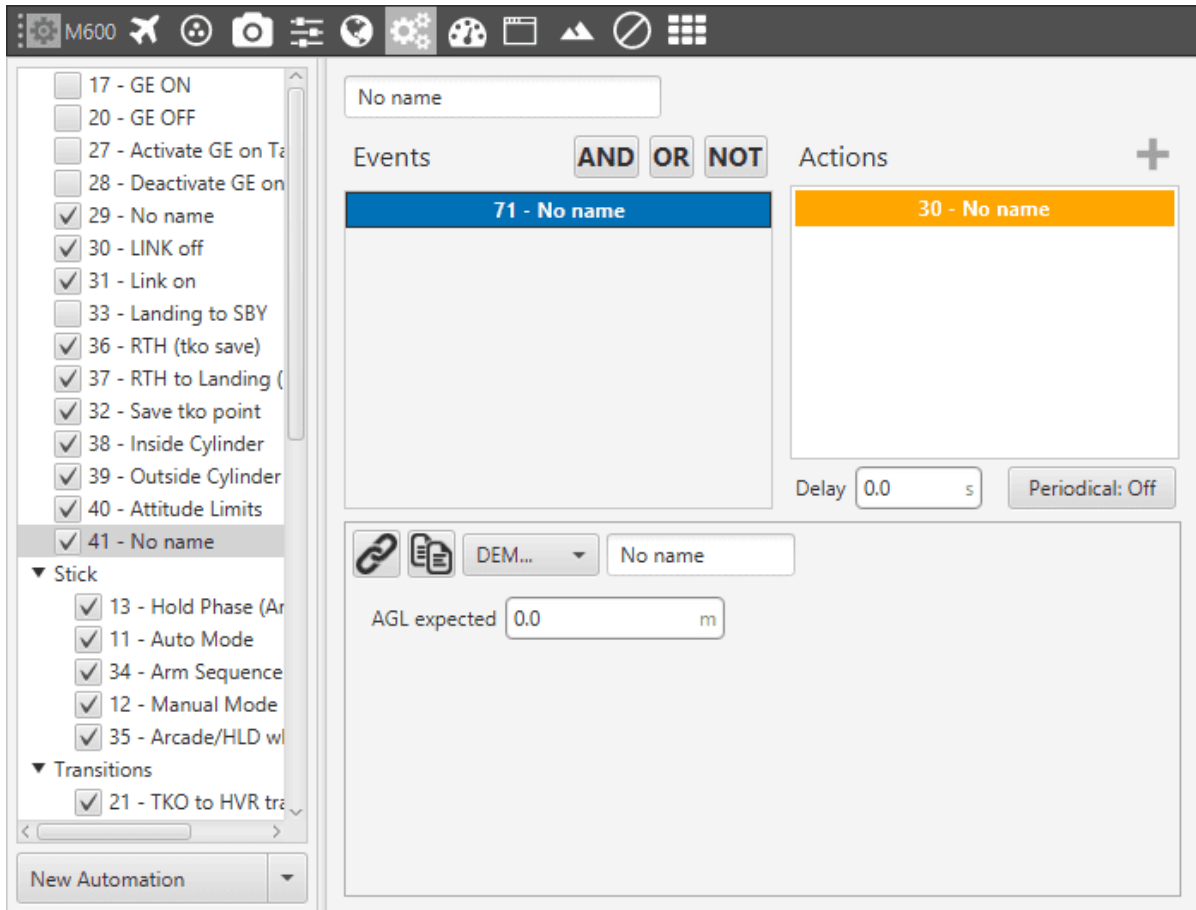
Action – Custom Serial TX

The two parameters to configure in this action are:

- **Producer:** where is located the custom message: RS custom message 1, 2 or 3.
- **Custom message:** the one that will be sent.

7.2.6.2.7 DEM Calibration

This option allows the calibration of the digital elevation model by setting the actual AGL value in the same way as shown in *Quick Commands*.

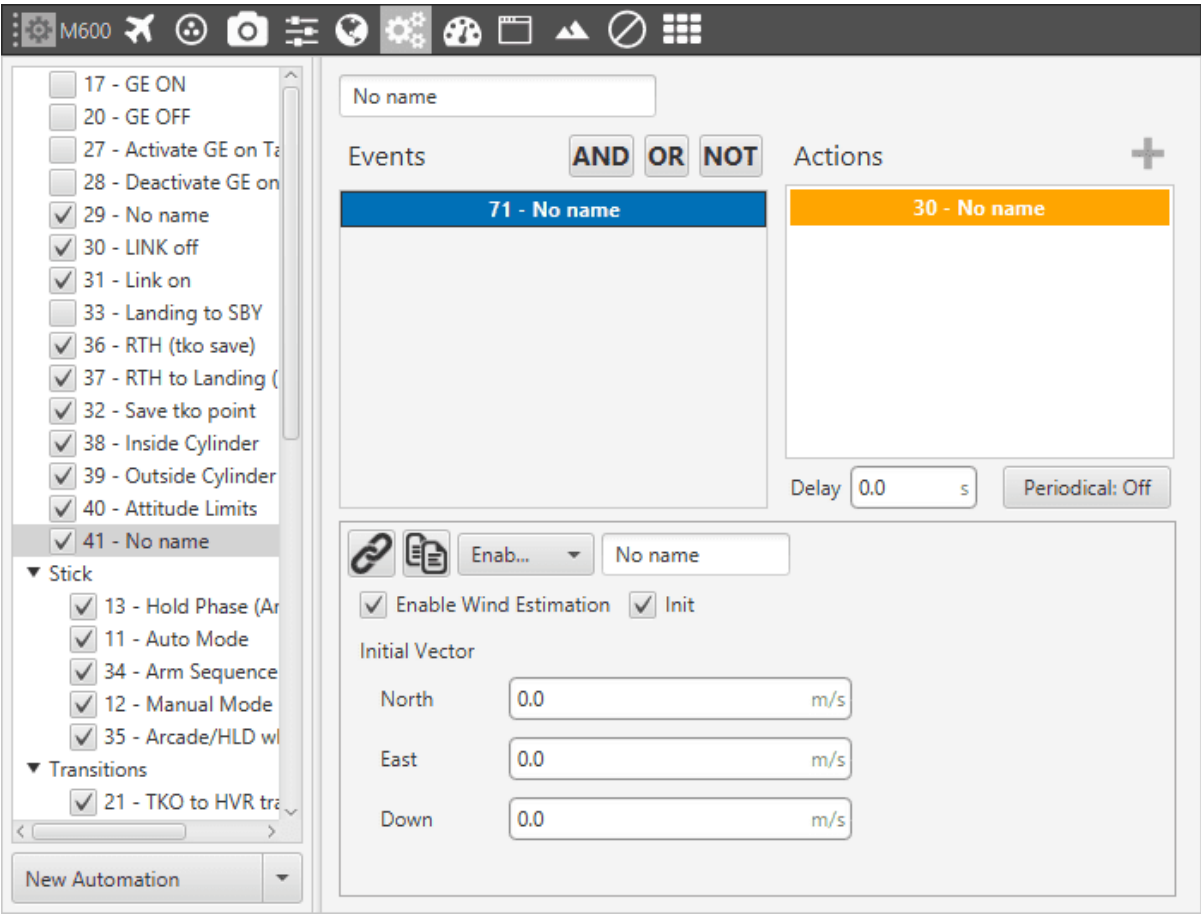


Action – DEM Calibration

7.2.6.2.8 Enable/Disable Wind Estimation

Note: This action is disabled by default when Veronte is started. To activate it, the user have to run the action.

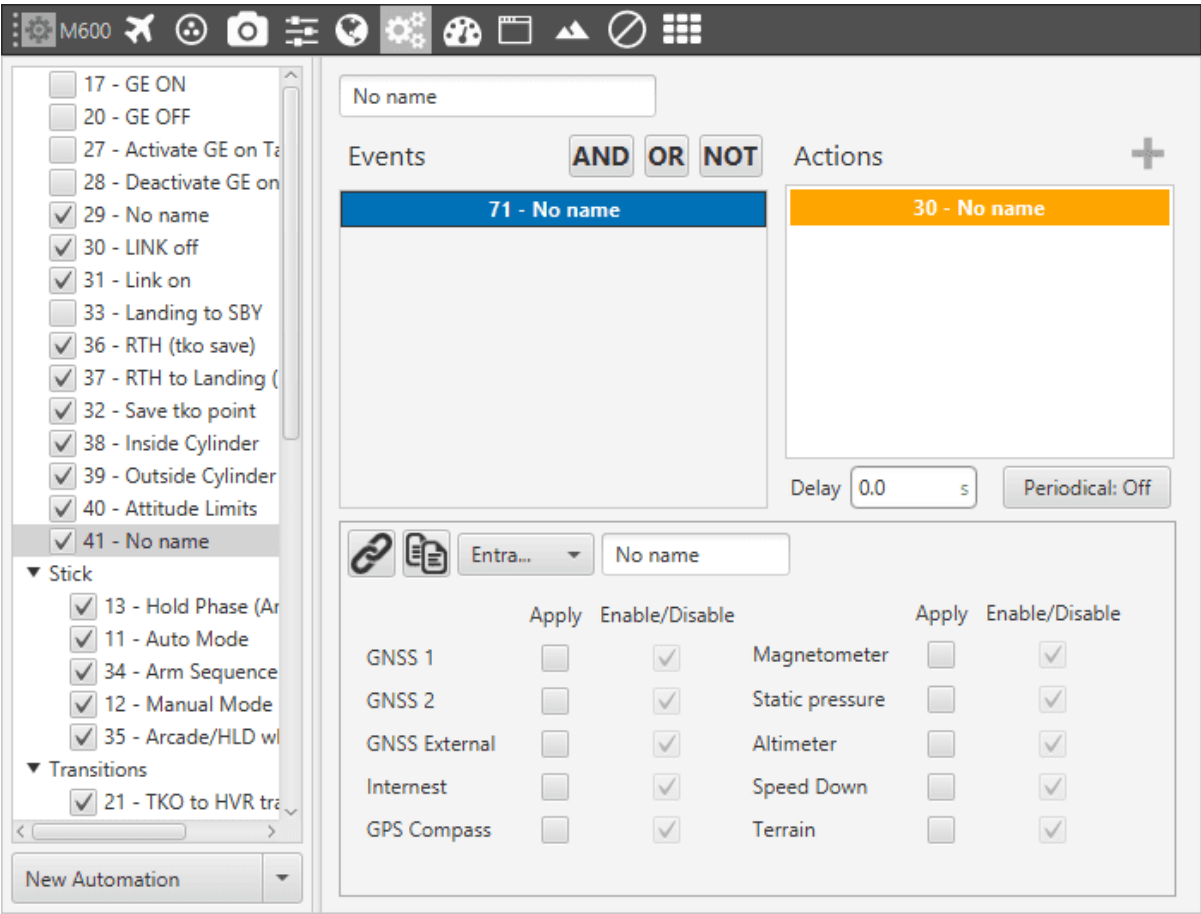
This action allows the user to enable or disable the wind estimation. Besides, an initial wind vector can be set to a faster convergence of the estimation.



Action – Enable/Disable Wind Estimation

7.2.6.2.9 Entrance EKF

This action modify the entrances to the Kalman filter. It can be configured so that a sensor enter or exits the Kalman filter.

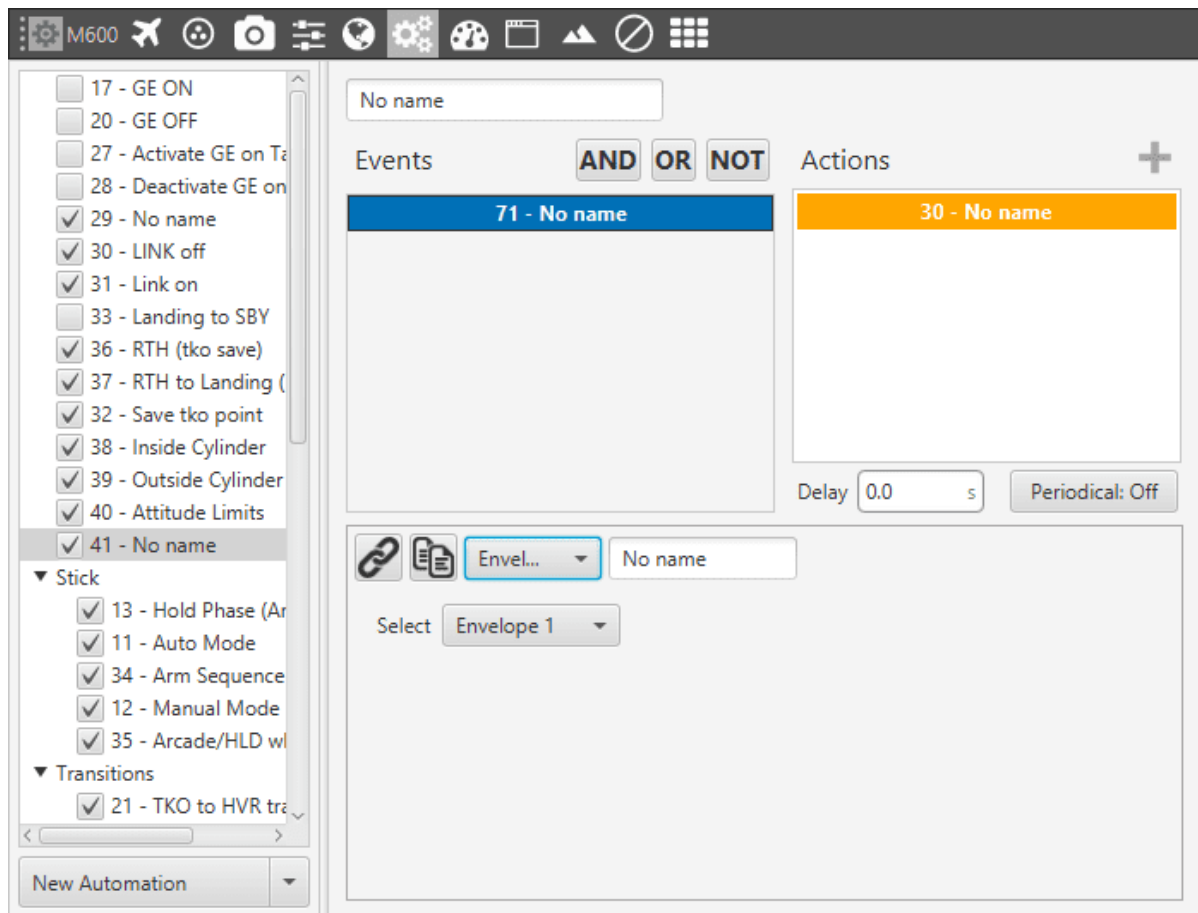


Action – Entrance EKF

Only those sensors that have their apply box checked are modified. Once this box is checked, the enable or disable box of the Kalman filter can be accessed.

7.2.6.2.10 Envelope

This action is used to change the envelope during the flight mission. Envelopes are created in the Control Menu, see section *Envelope*.



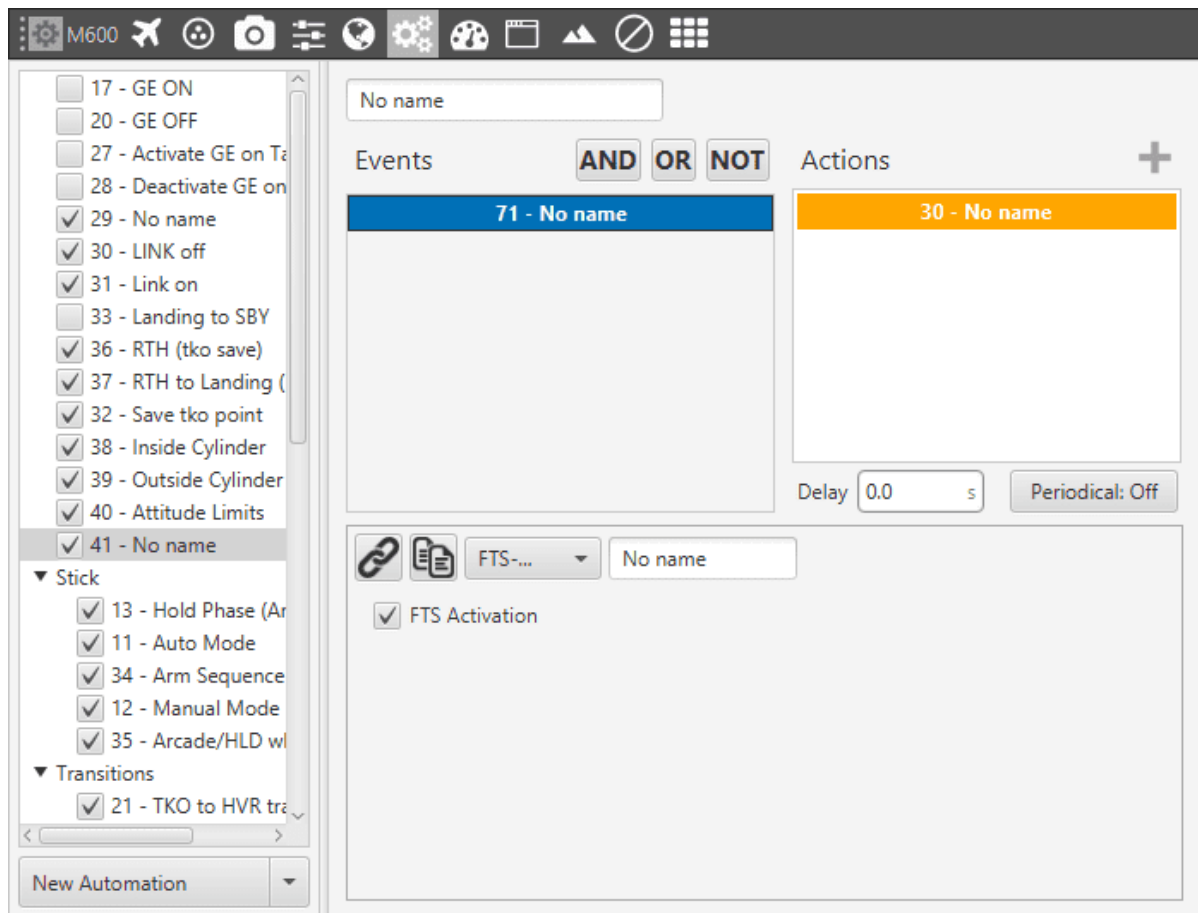
Action – Envelope

This is useful with a hybrid platform, being possible to change the envelope when the aircraft changes its configuration. Envelopes are selected from those created previously.

7.2.6.2.11 FTS Activation

Note: This action is useful for 4xVeronte. In 1xVeronte this action will have no effect.

This action activate the flight termination system (FTS) bit useful for arbitration.

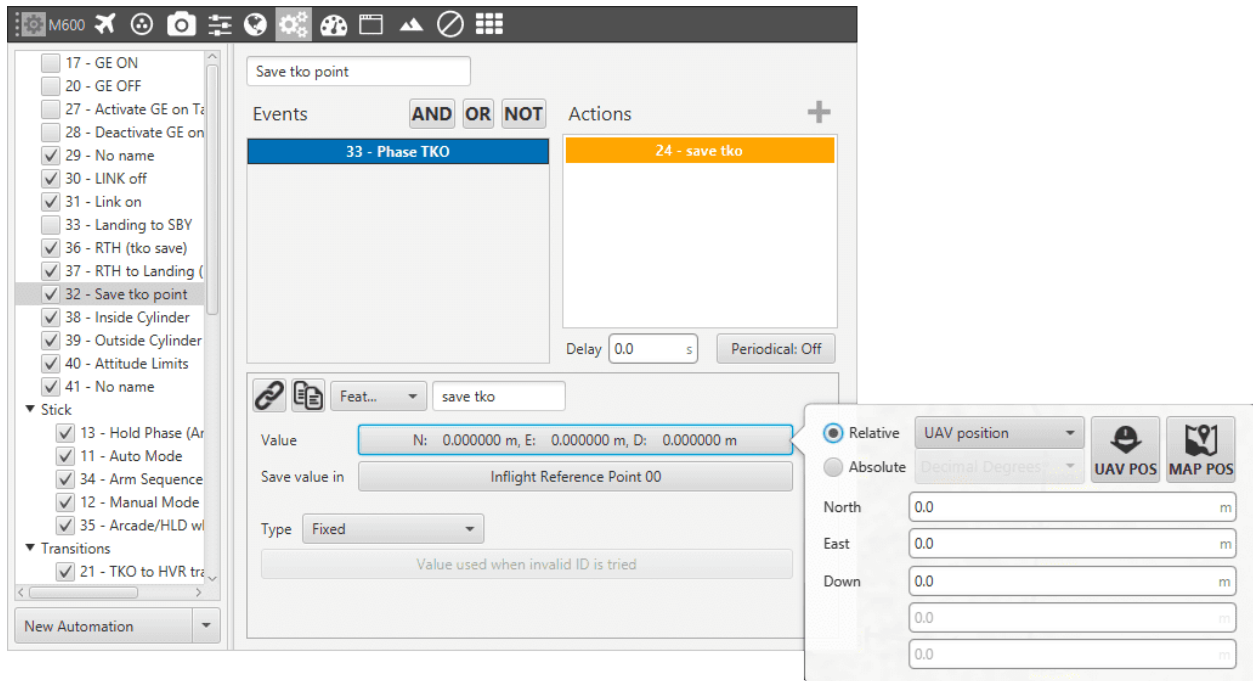


Action – FTS Activation

When two or more autopilots activate their FTS the arbiter can activate a safe system such as a parachute.

7.2.6.2.12 Feature

When this action is triggered, a position is stored in the desired variable. This position can be absolute or relative (in the figure below the current position of the aircraft would be saved):



Action – Feature

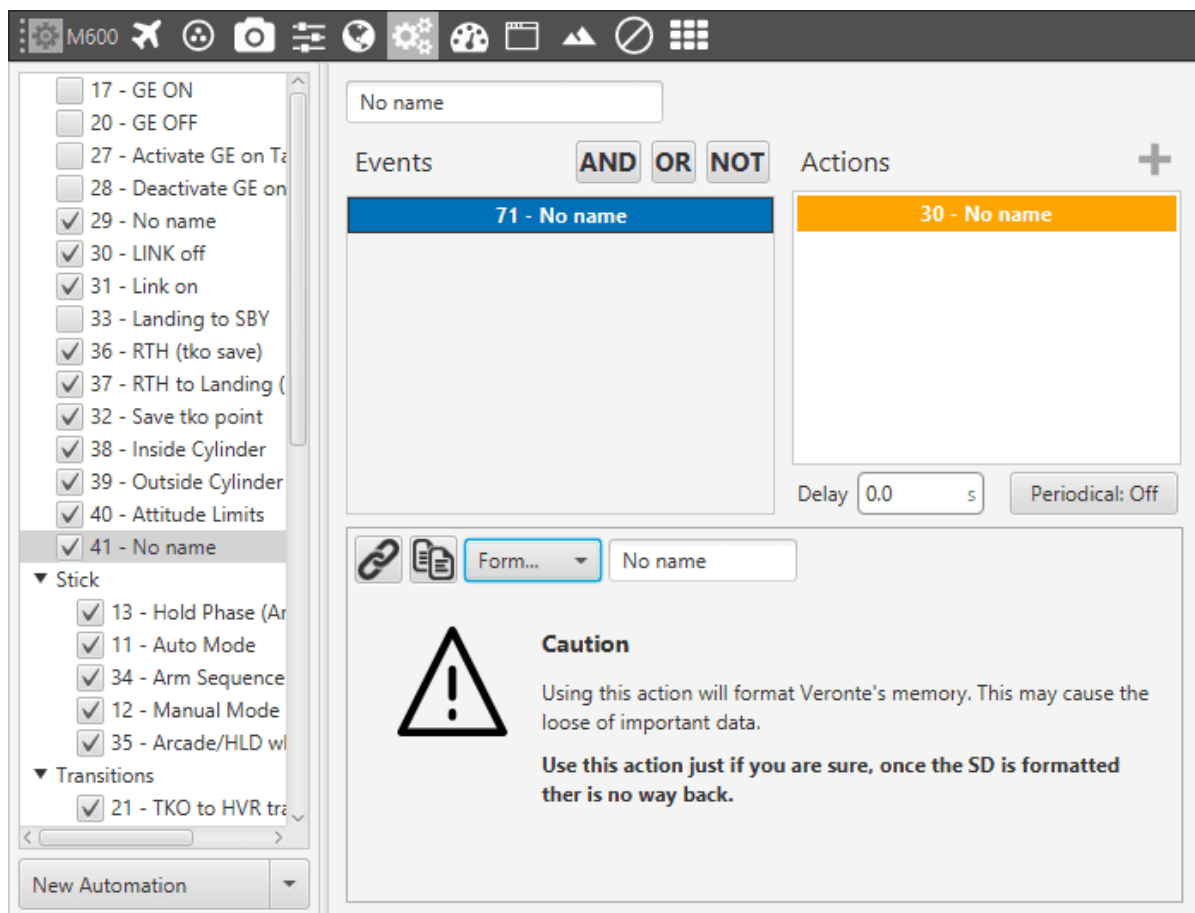
Besides, there are two types of features:

- **Fixed:** once the point has been generated it remains fixed.
- **No change:** if the point has been created relative, it remains relative all the time.

7.2.6.2.13 Format SD

Warning: This action will have irreversible effects on your Veronte. Formatting the SD card will delete important and mandatory files for the correct functioning of Veronte. In order to recover a formatted Veronte the customer should contact with Embention support.

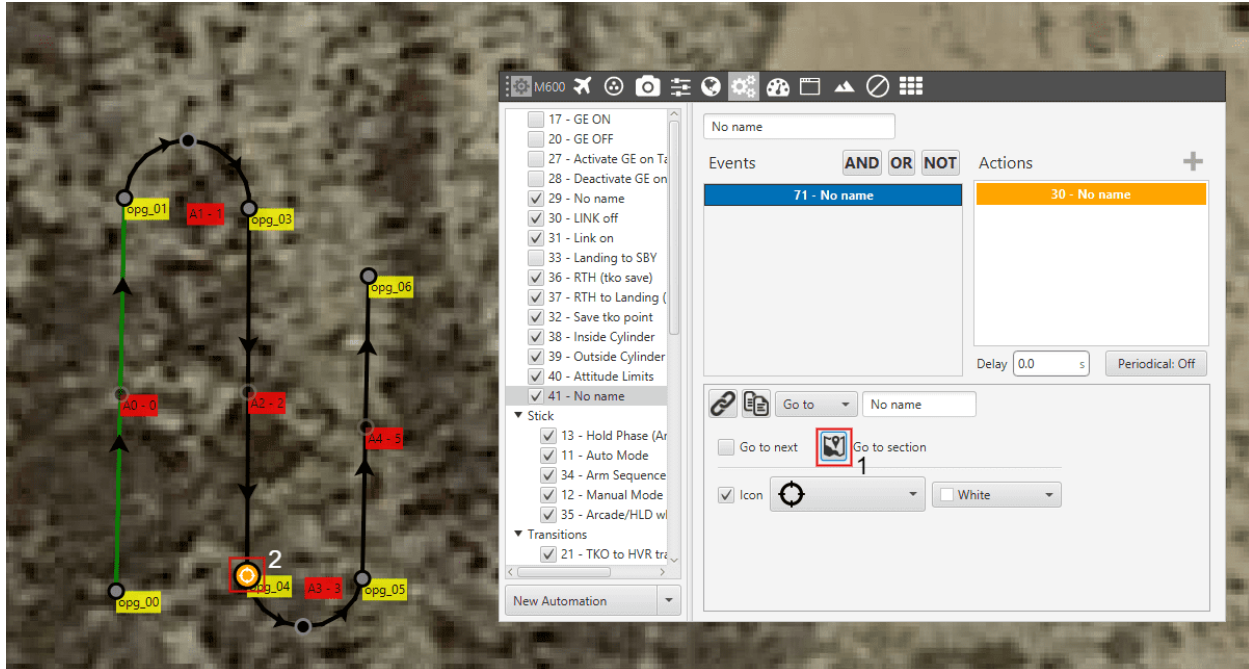
This action will format the SD card, deleting the configuration and flight logs from it.



Action – Format SD

7.2.6.2.14 Go To

This action is used to make the aircraft go to a path (or waypoint if it is alone) of the route created by the user with the *Mission toolbar*.



Action – Go To

- Clicking in **1** allows the user to select a waypoint on the map (2 for example).

Once the action is triggered, the vehicle will go to that patch (or waypoint). If the patch is on a route, the vehicle will follow the selected patch and then it will continue the route going to its adjacent. On the other hand, if the option **Go to next** is selected, once the event is accomplished the aircraft will forget about the current path that is following and it will go to the next one. If the event happens again it will “jump” another patch and go to the following.

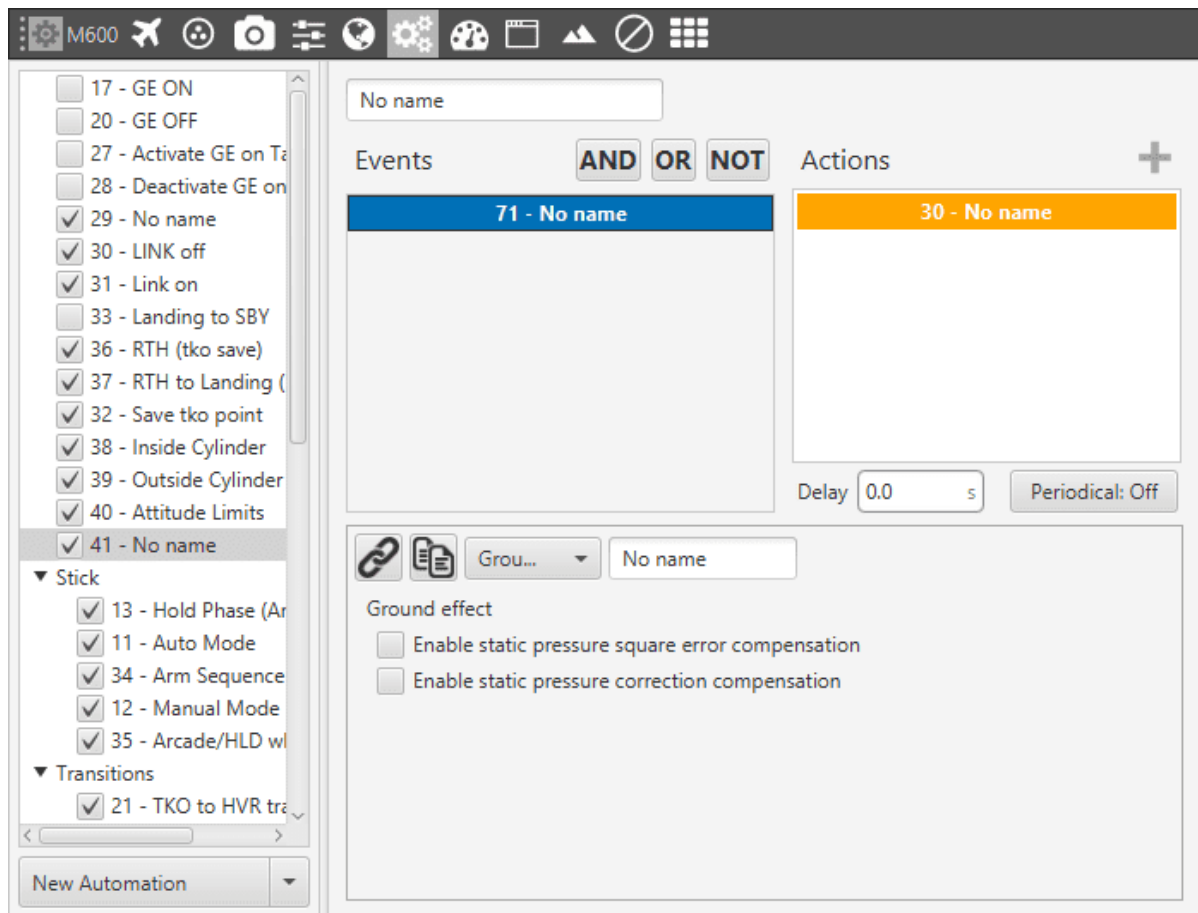
For example, considering the route that appears in the previous figure, if the automation **Go to next** is triggered by a button, when the aircraft is in the first patch (the one that starts on the green waypoint) and the button is pressed it will go to the next curved patch. If the button is pressed again before reaching that patch, the aircraft will go to the next straight line (parallel to the first one) without going over the curved patch.

It is possible to change the appearance of the waypoint to an image selected in the **icon** option, so the user can identify easily the waypoint linked to that automation.

7.2.6.2.15 Ground Effect

Note: This action is disabled by default when Veronte is started. To activate it, the user have to run the action.

This action activate the groun effect compensations previously configured in *Static pressure sensor*.



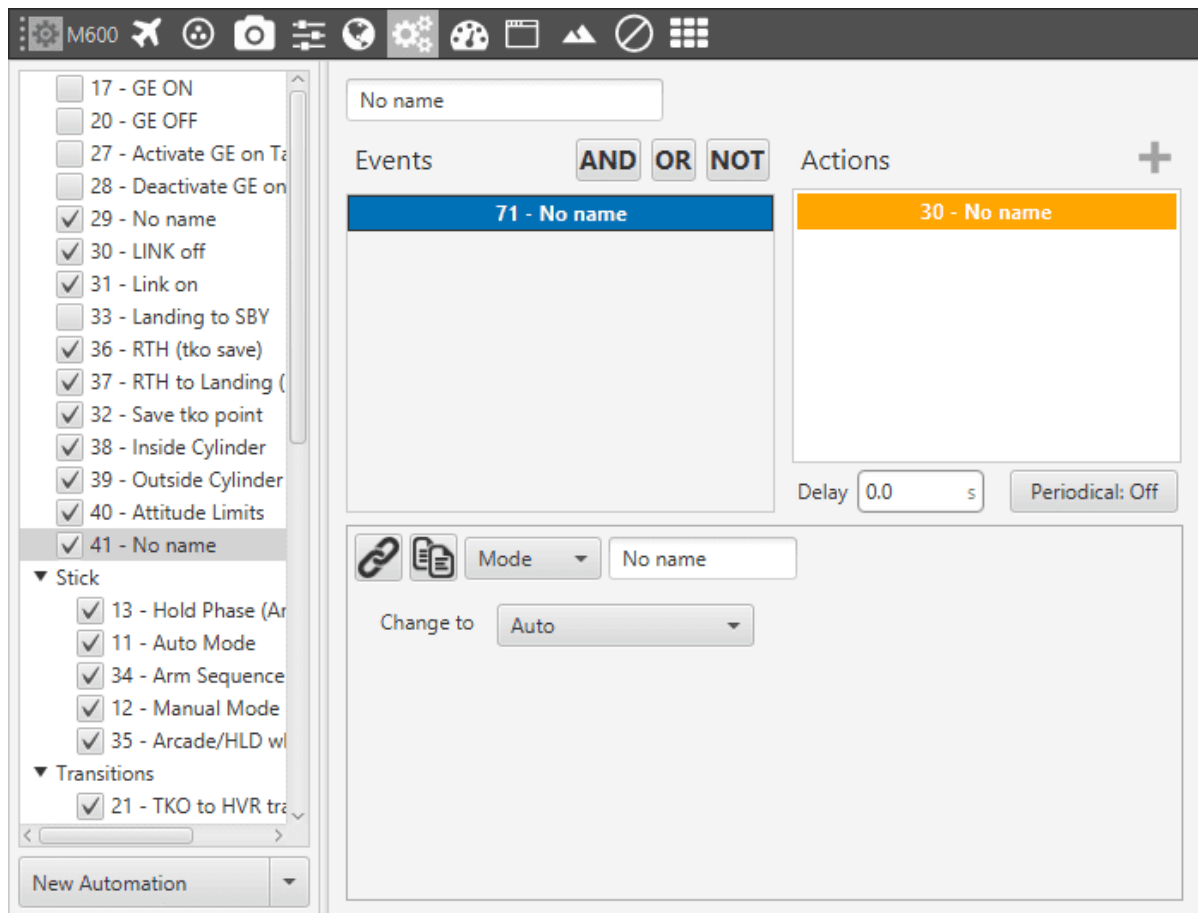
Action – Ground Effect

There are two types of ground effect compensation:

- **Enable static pressure square error compensation:** compensations in the pressure square error when Veronte is close to the ground.
- **Enable static pressure correction compensation:** compensation in the pressure reading from the table configured in *Static pressure sensor*.

7.2.6.2.16 Mode

The flight mode is changed to the one specified in this option.

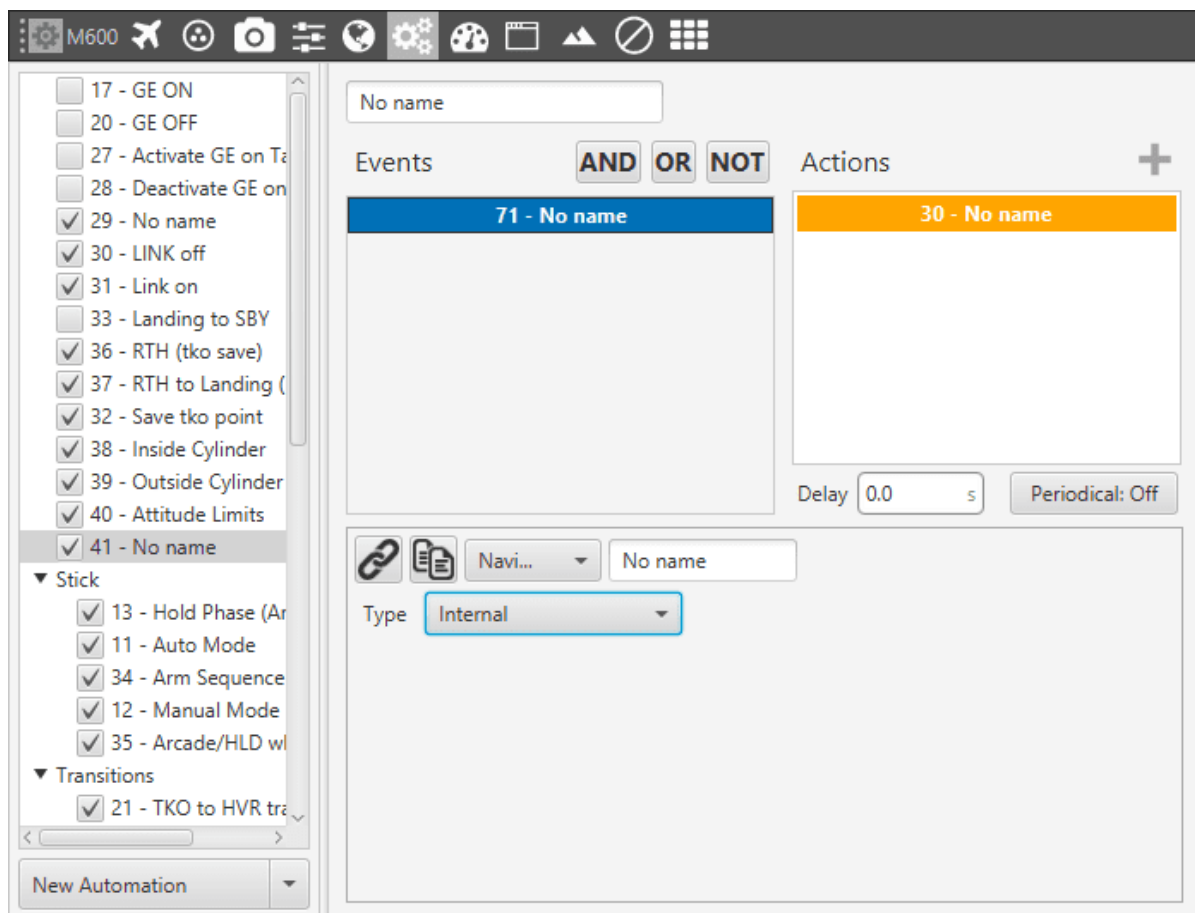


Action – Mode

These modes have been created previously. See [Modes section](#) for more information about creating modes.

7.2.6.2.17 Navigation

This action is used to change the navigation mode used by the aircraft. By default, the UAV uses a sensor fusion Internal algorithm, but for example, if the GPS falls, this algorithm produces bad results so it would be convenient to change to another type if that happens (External). The navigation without GPS will make the aircraft fly stable but it will not be possible to command a path to follow during that time, so it can be used as a safety mode to avoid a malfunction of the system when the GPS signal is lost.



Action – Navigation

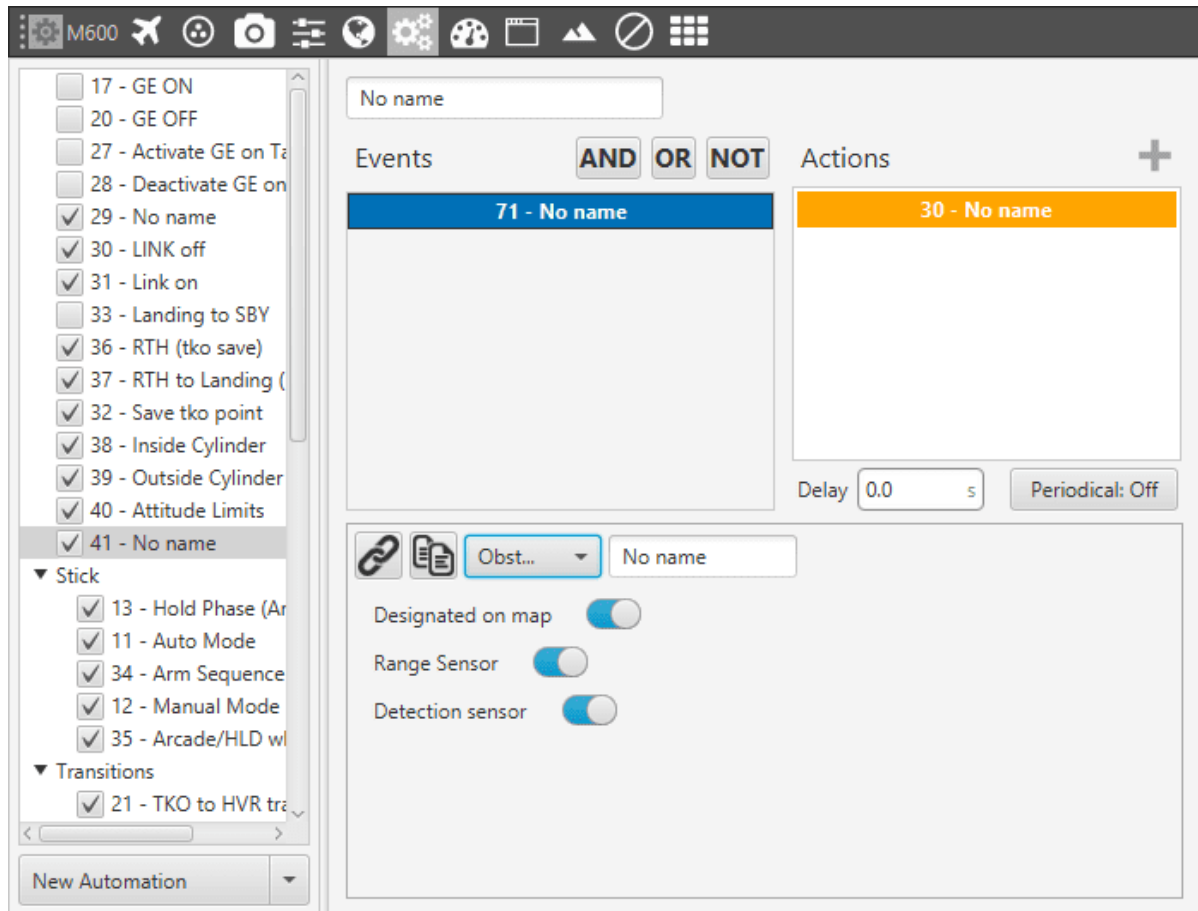
The options available are:

- **Internal:** uses internal data for navigation. Data (position, attitude, etc.) is processed into Veronte Unit from sensor measures.
- **External VCP:** uses external data for navigation. Data (position, attitude, etc.) is provided by Veronte Communication Protocol (VCP).
- **External Var:** uses external data for navigation. Data (position, attitude, etc.) is provided by Var.
- **Vectornav VN-300:** uses external data for navigation. Data (position, attitude, etc.) is provided by Vectornav VN-300.

7.2.6.2.18 Obstacle Avoidance

Note: This action is disabled by default when Veronte is started. To activate it, the user have to run the action.

This action enables the obstacle avoidance predefined in *Obstacle Detection*.



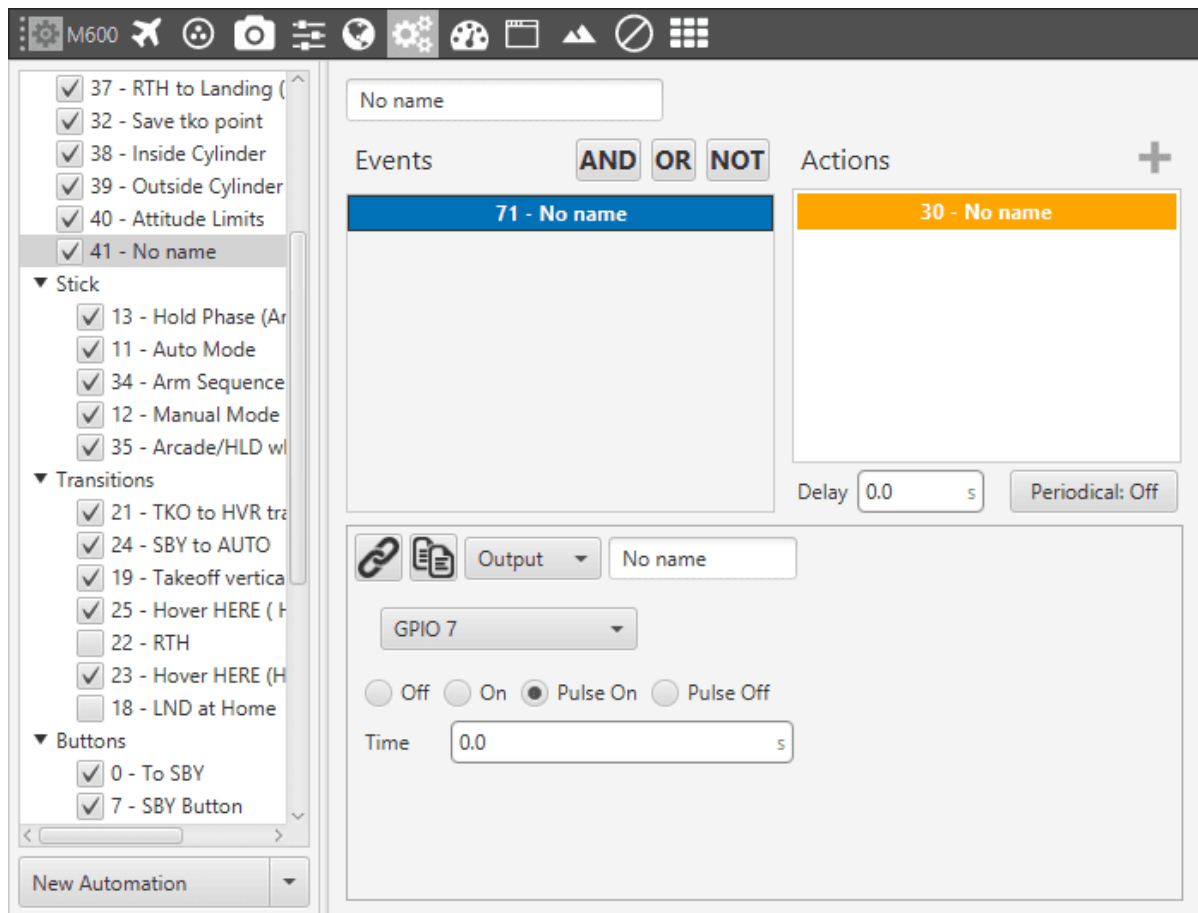
Action – Obstacle Avoidance

Three types of obstacles can be enabled:

- **Designated on map:** those obstacles that appears on the map, both obstacle zones and aircraft received by ADS-B.
- **Range sensor:** obstacles defined in *Range sensors* section.
- **Detection sensor:** obstacles defined in *Obsens* section.

7.2.6.2.19 Output

This action is used to set an output value in a GPIO pin. The output pin should have been configured as GPIO (visit section *GPIO*).



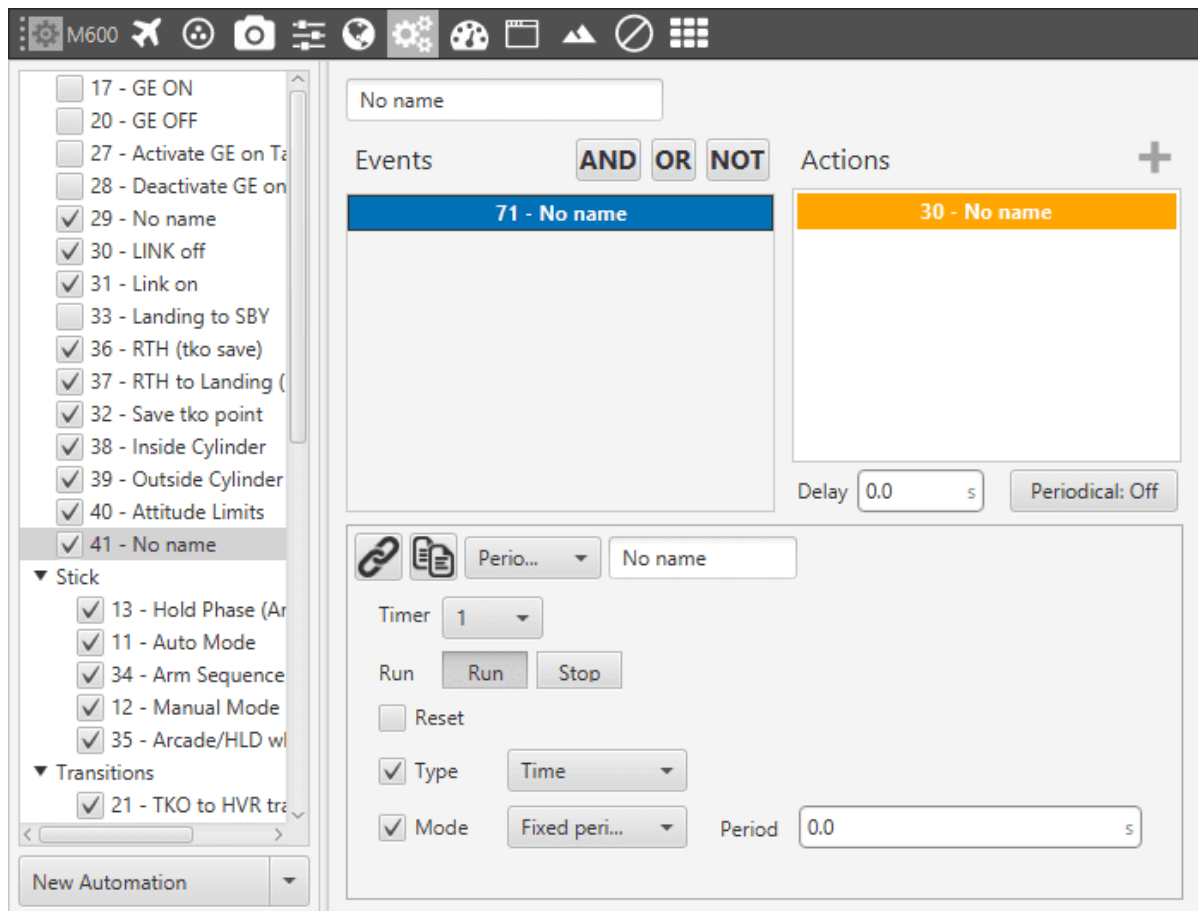
Action – Output

There are four possible output signals:

- **Off**: provides continuous 0V output.
- **On**: provides continuous 3.3V output.
- **Pulse ON**: provides 3.3V for the specified time and after that 0V.
- **Pulse OFF**: provides 0V for the specified time and after that 3.3V.

7.2.6.2.20 Periodical

This action is used to set a timer during a flight operation.



Action – Periodical

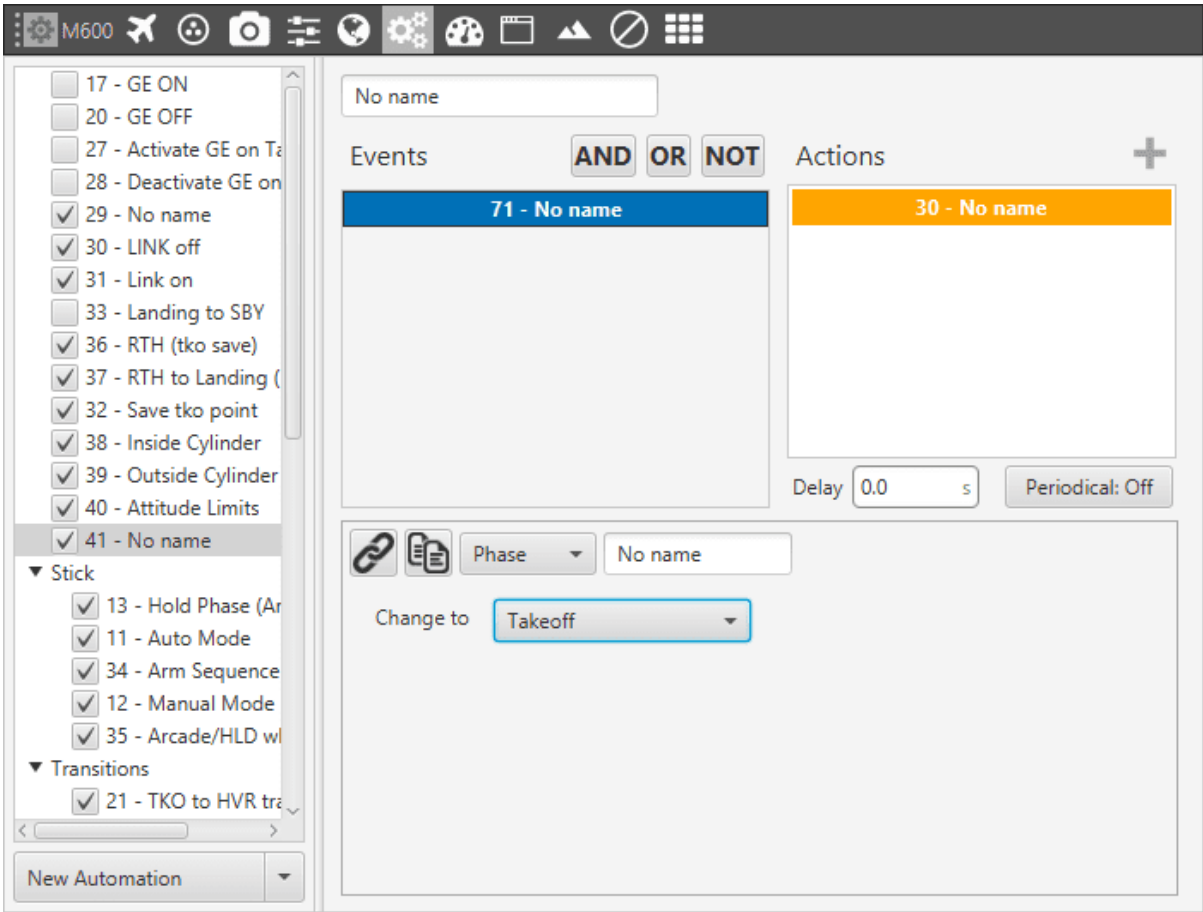
The first parameter is an identifier for the timer, so it can be used in an event for another automation. In order to explain the other parameters of the timer action, a set of examples will be detailed, each of them with different options.

- **Run + Distance/Time + Continuous:** when the action is triggered, the timer will be started and will measure distance/time from that instant until the moment when the autopilot is turned off (or until another automation acts on the same timer).
- **Run + Distance/Time + Fixed Delay/Period:** once the action has been triggered, the timer will start to measure a distance/time. Each time the value indicated in Period is reached, the event linked to this timer will be triggered. For example, if the user wants to take a photo each 25 meters, the timer should have Distance in the Type option and 25 meters in Period, then in another automation, an event of type Timer is created, so each time the timer reaches 25 meters the event will be triggered and the action will be carried out
- **Distance + Vector:** the distance is measured in the direction indicated by the vector.
- **Stop:** the timer will be stopped. Another automation should be created to run it again.
- **Reset:** when this action is active, the timer is reset to zero before starting to measure. If the reset is used with Stop, the timer will be stopped and set back to zero.

The difference between fixed delay and fixed period has been explained in [Automations](#).

7.2.6.2.21 Phases

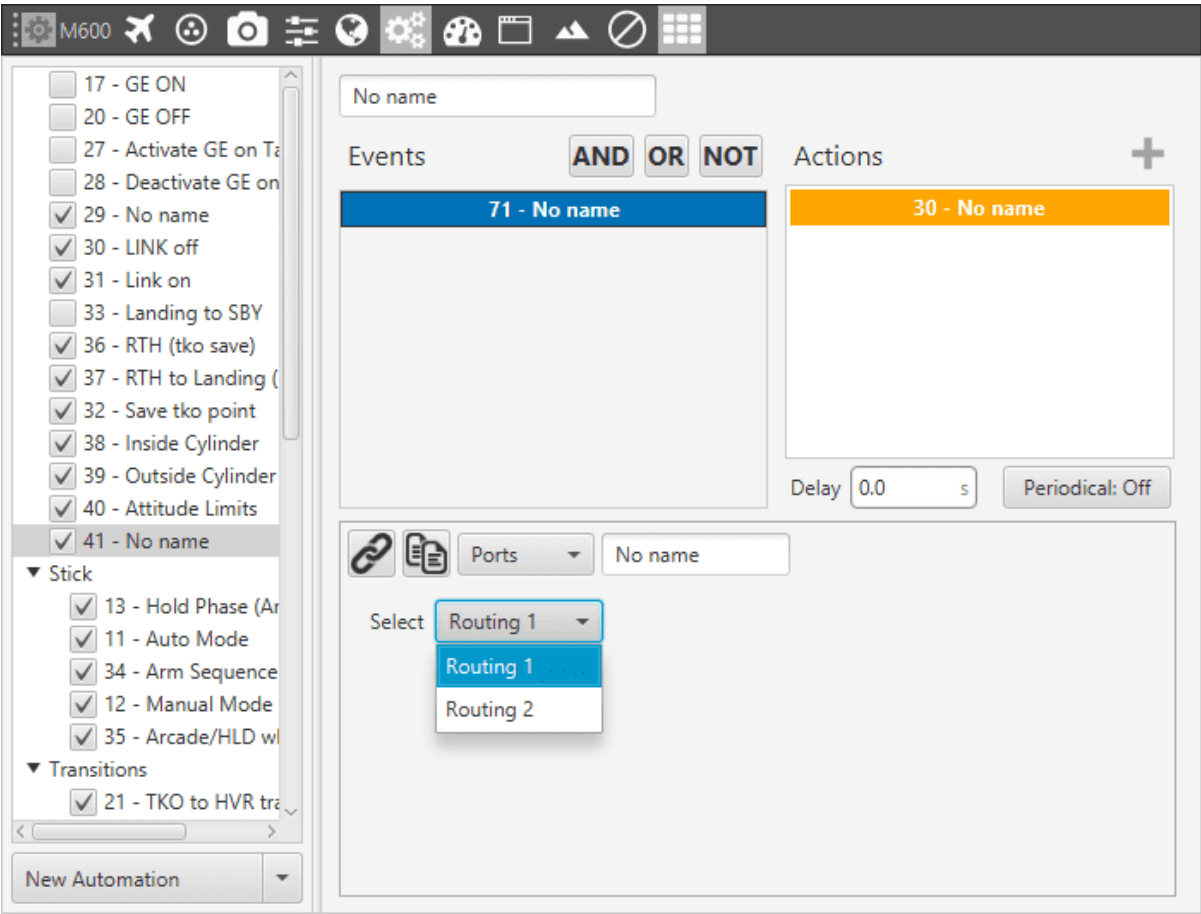
The flight phase is changed to the one selected in this action.



Action – Phases

7.2.6.2.22 Ports

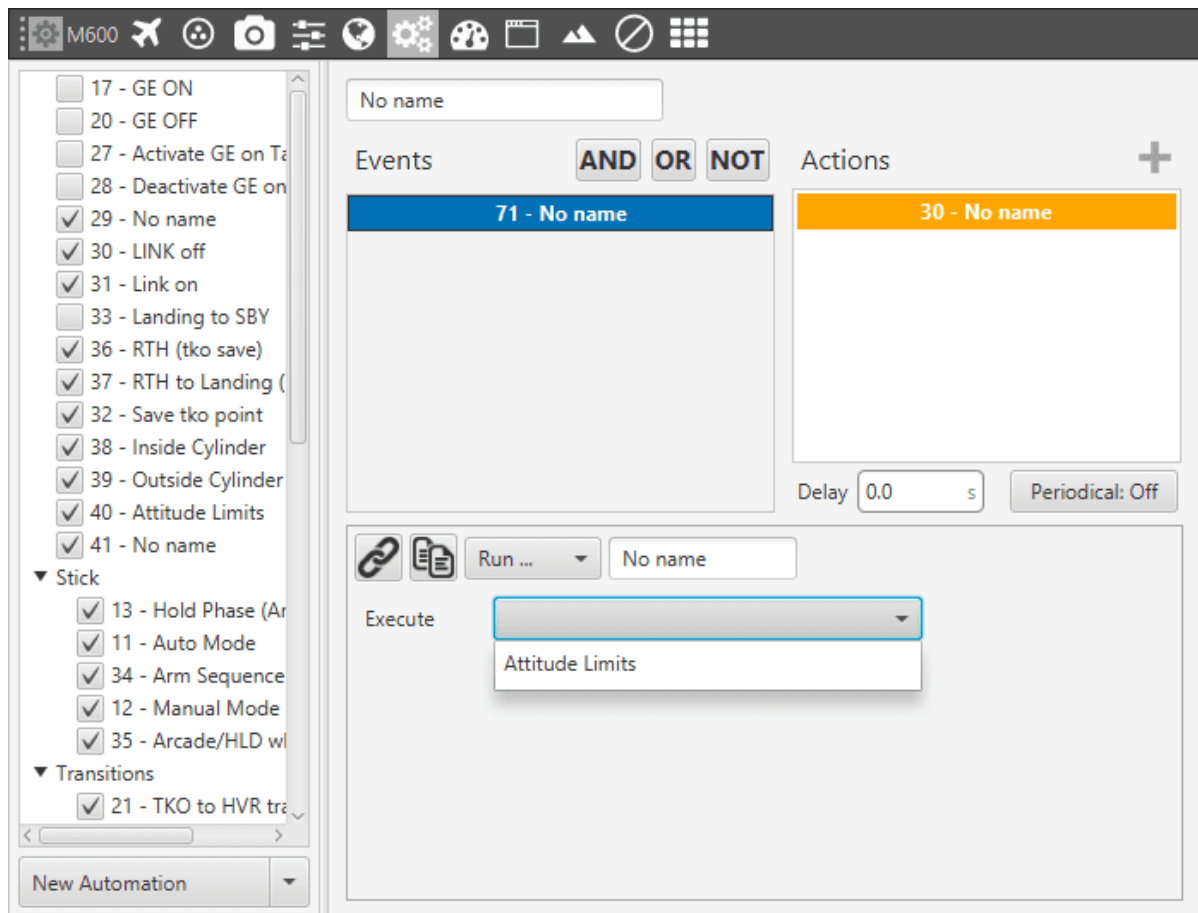
This action allows the user to switch between two pre-set configurations defined in *Ports*.



Action – Ports

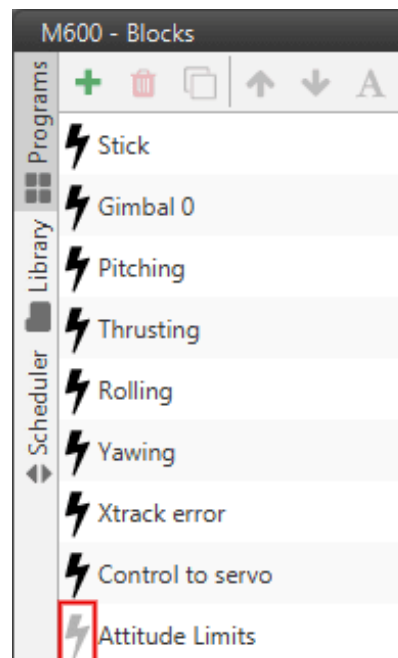
7.2.6.2.23 Run Block Program

When this action is triggered, the block program specified in the “Execute” label is executed.



Action – Run Block Program

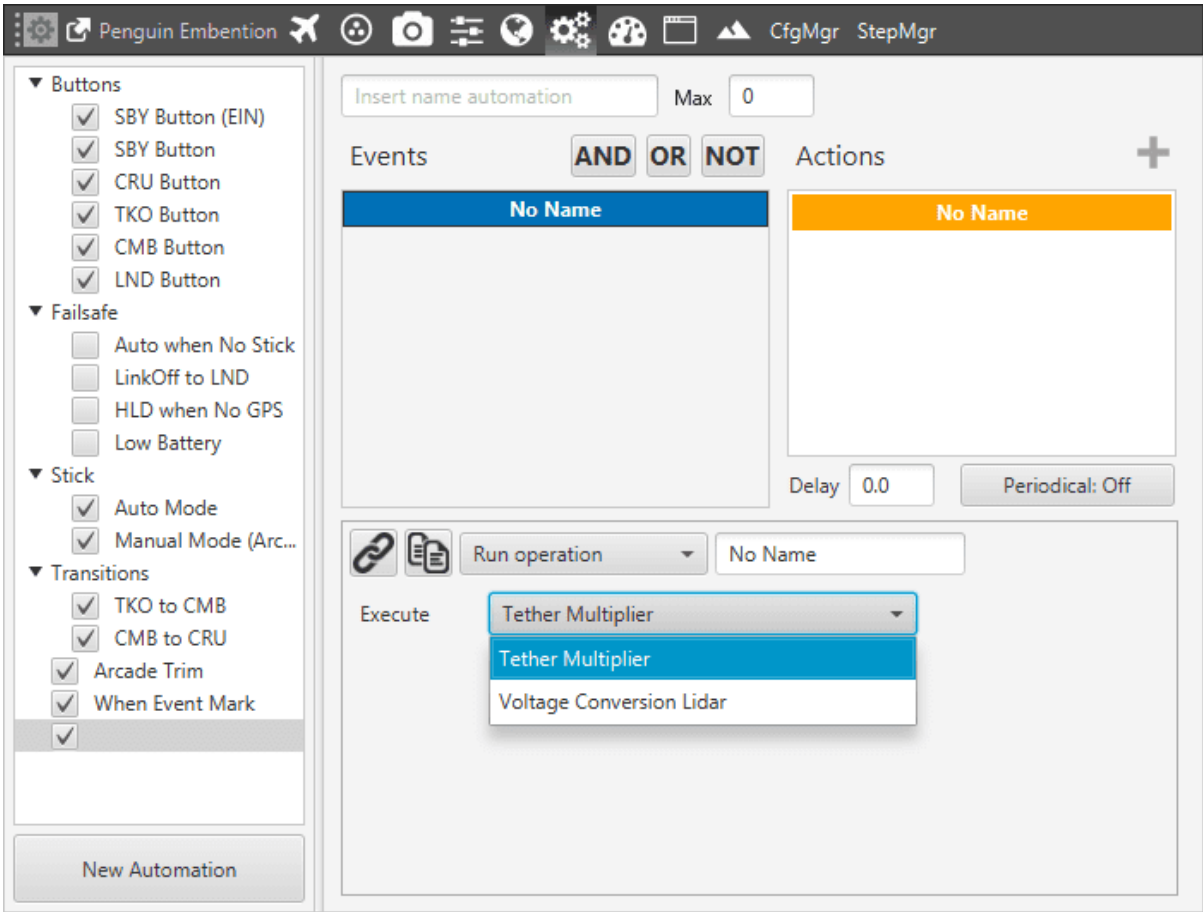
This action can run those programs that have the lightning icon in grey colour as those programs with the lightning icon in black color run continuously.



Action – Lightning Icon

7.2.6.2.24 Run Operation

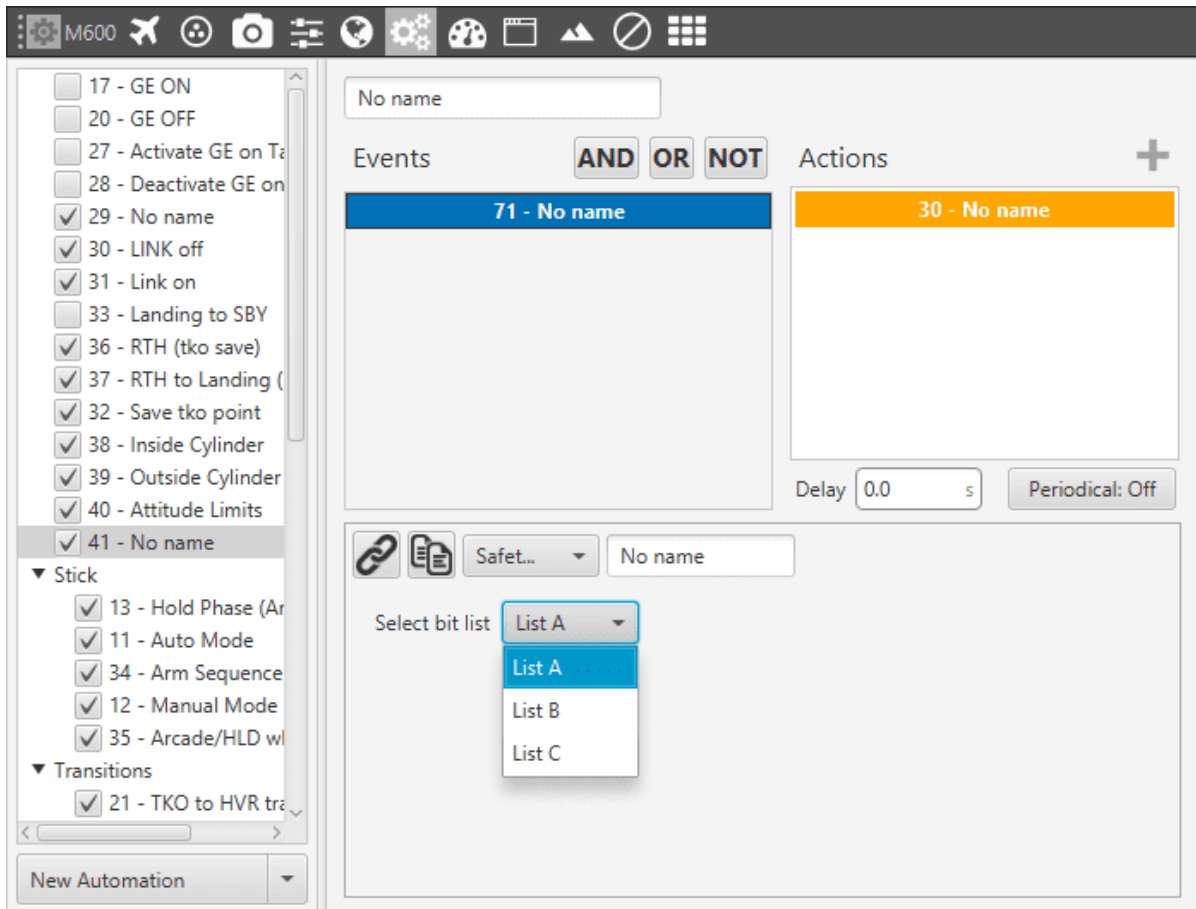
An operation previously defined in **Variables/System Variables** is carried out, see section *System Variables*.



Action – Run Operation

7.2.6.2.25 Safety Bits

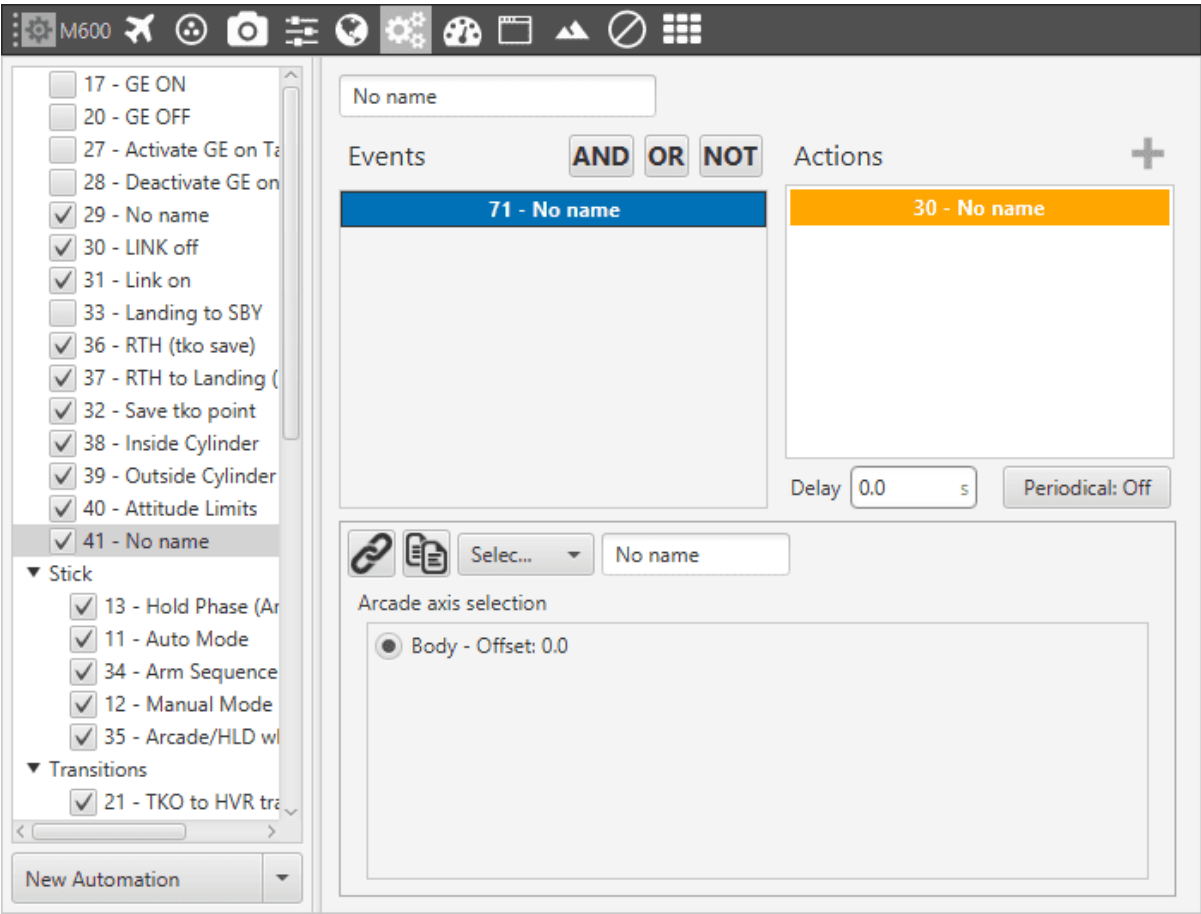
This action selects a predefined safety bits list which must have been previously configured.



Action – Safety Bits

7.2.6.2.26 Select Arcade Axis

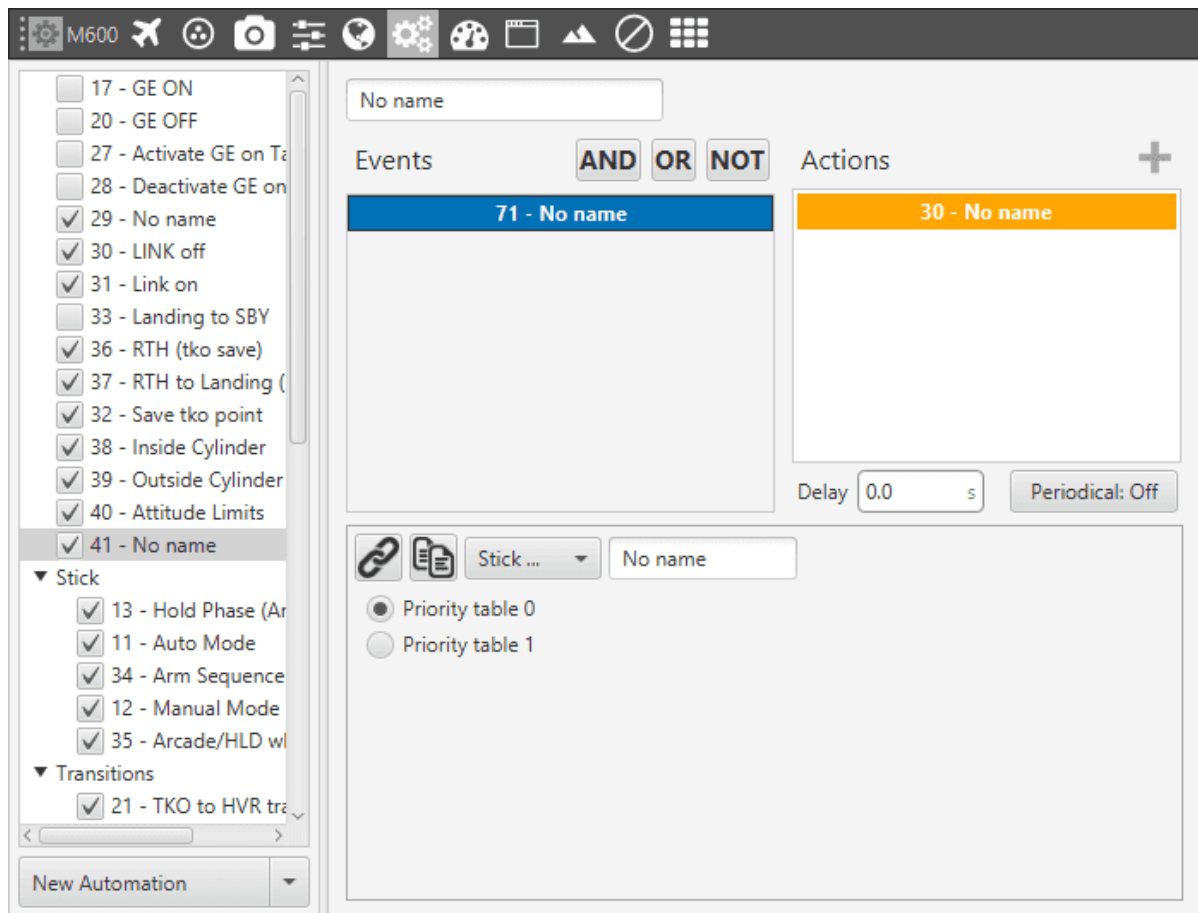
The axes system of the aircraft is changed to one that has been previously created in the Arcade Axis option inside the Control Panel, see section [Arcade Axis](#)



Action – Select Arcade Axis

7.2.6.2.27 Stick Priority

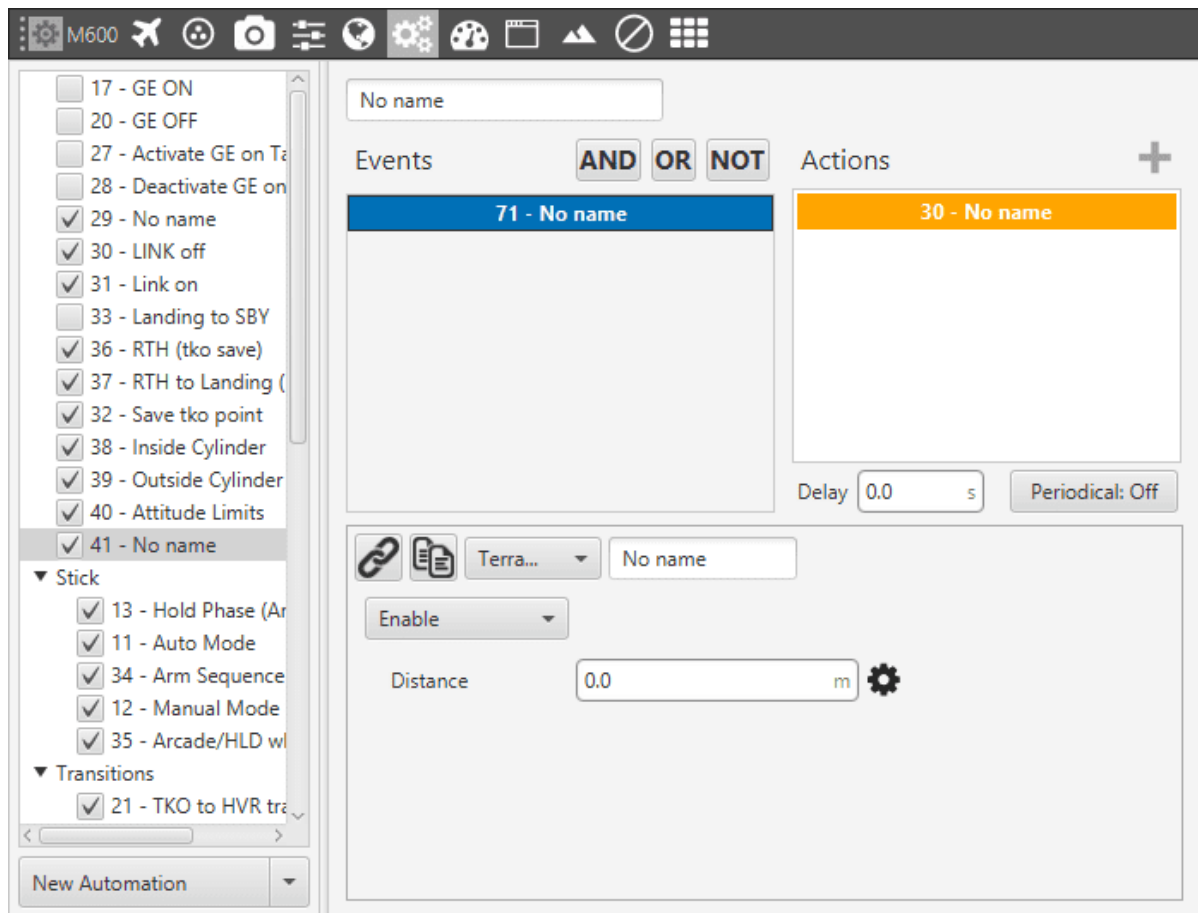
The stick priority table is changed to one that has been previously created in Stick Configuration inside the Devices menu, see section *Stick*. By default priority table 0 is selected when Veronte starts.



Action – Stick Priority

7.2.6.2.28 Terrain Obstacle

This option is used to make the aircraft climb when is reaching an altitude of zero meters, for example, when flying towards a mountain. This option is not activated all the time because it will not allow the aircraft to land.

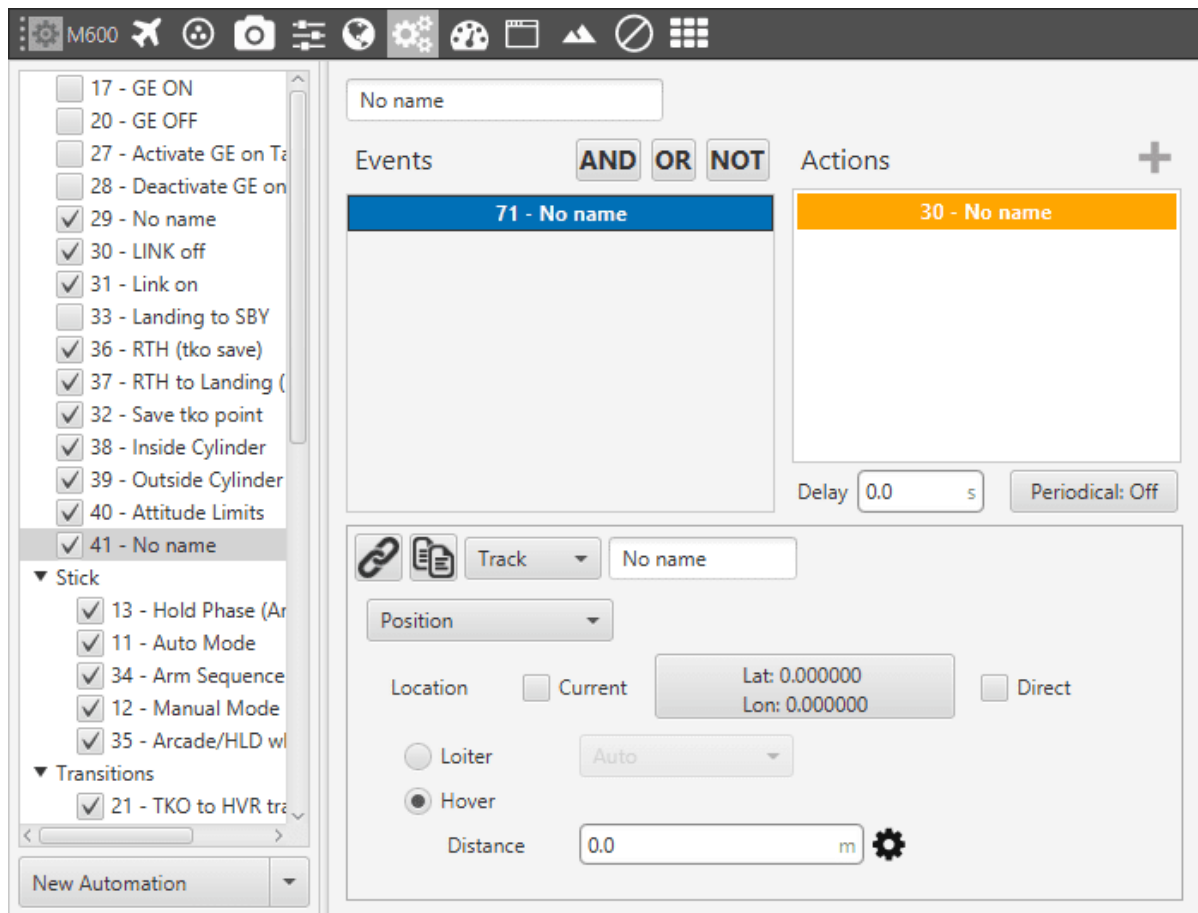


Action – Terrain Obstacle

- **Distance:** establish how the aircraft will climb, it can be said to be a repulsion value. High values made the platform ascent quickly. This effect is more noticeable when the aircraft is close to the ground.

7.2.6.2.29 Track

This action is used to configure a hover/loiter route (depending if it is a multicopter or an airplane) for the platform. Besides, there exists an option to follow a moving object.



Action – Track

There are our different options for the Track action, selecting **Disabled** no action will have effect on the guidance. The others are explained below.

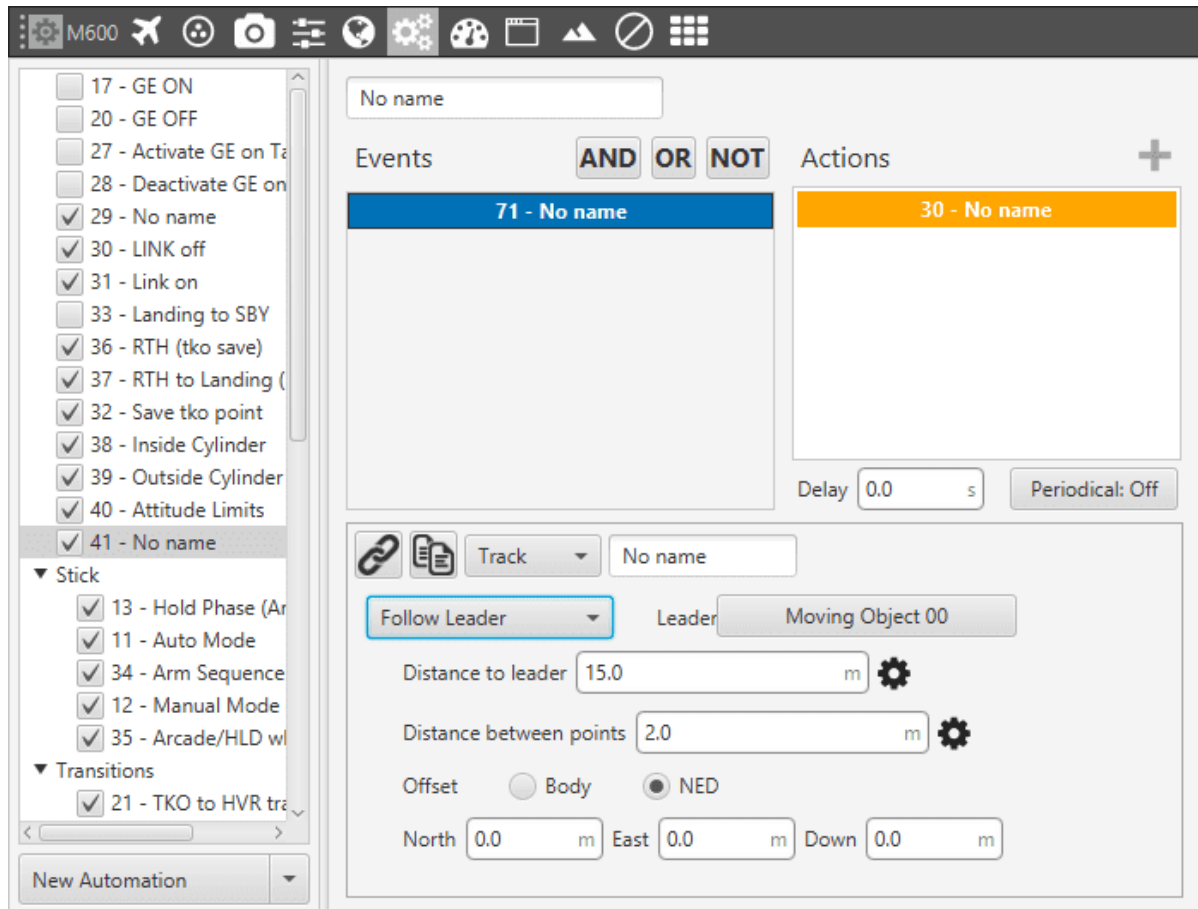
7.2.6.2.29.1 Position

The aircraft will loiter/hover in a selected point.

- Selecting **Current** will make the platform to hover over the position that the vehicle has when this action is triggered, or loiter around that point in a circular route with a radius indicated in **Distance**.
- It is also possible to select the direction of the loiter (**Auto**, **Clockwise** and **Anticlockwise**).
- On the other hand, the box (Longitude, Latitude) in the figure allows the user to select the point where the hover/loiter will be performed.

7.2.6.2.29.2 Follow Leader

The platform (Multicopter) will follow an moving object.



Follow Leader

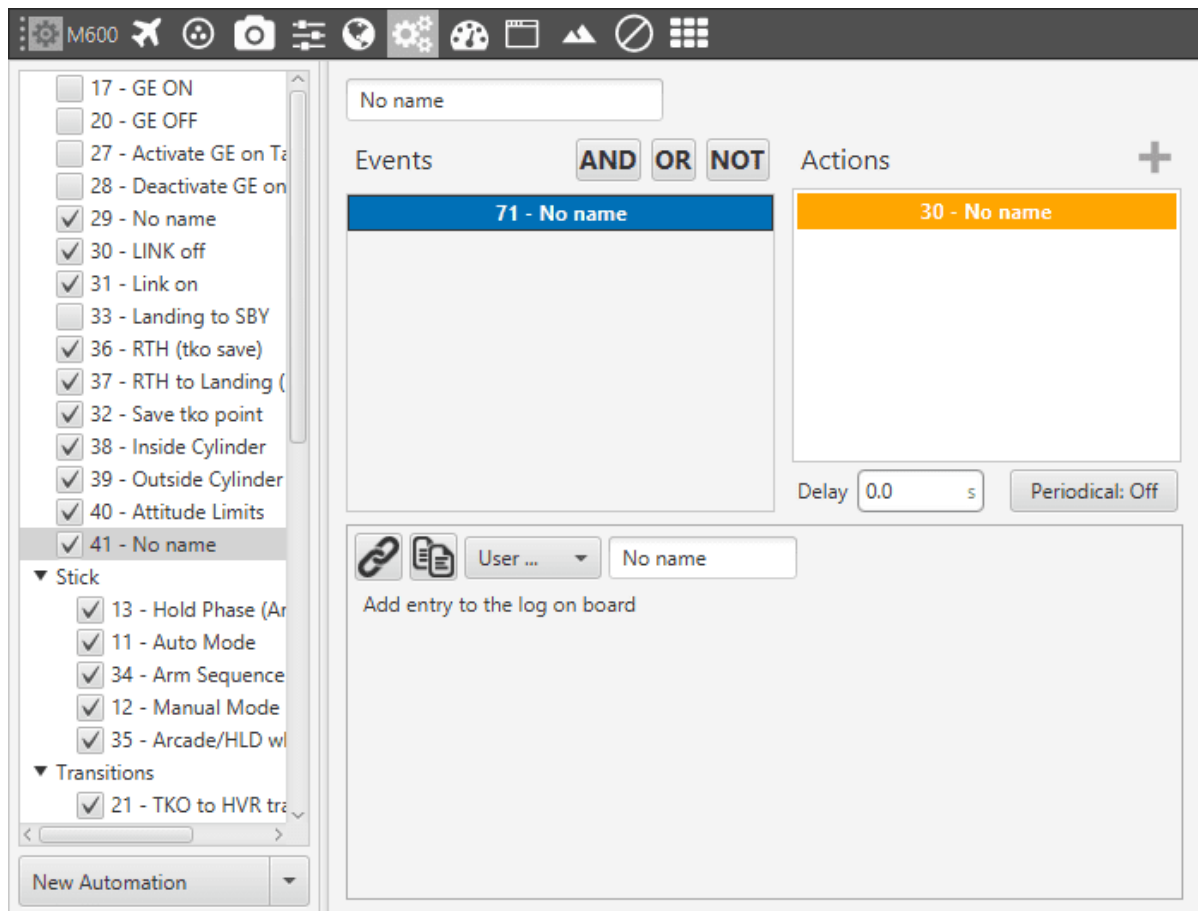
- **Leader:** here is selected the moving object i.e, the object to follow.
- **Distance to leader:** distance to leader over trajectory.
- **Distance between points:** leader route is generated by points separated by the distance specified here.
- **Offset:** user can establish offset parameters related to trajectory in Body or NED coordinates.

To configure correctly this automation, user has to follow the next steps:

- Configure Telemetry Air and Ground.
- Configure the automation as desired.

7.2.6.2.30 User Log

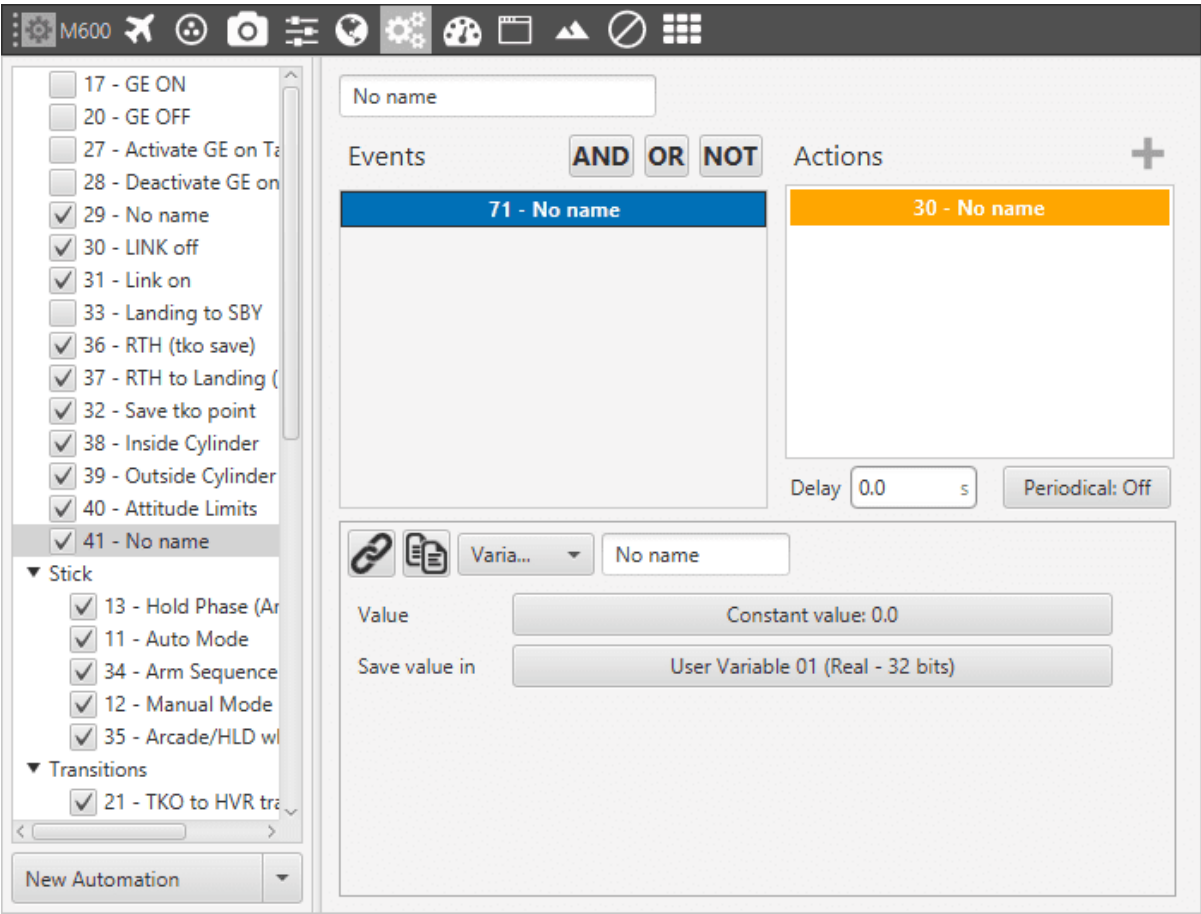
An entry is added to the log on board.



Action – User Log

7.2.6.2.31 Variable

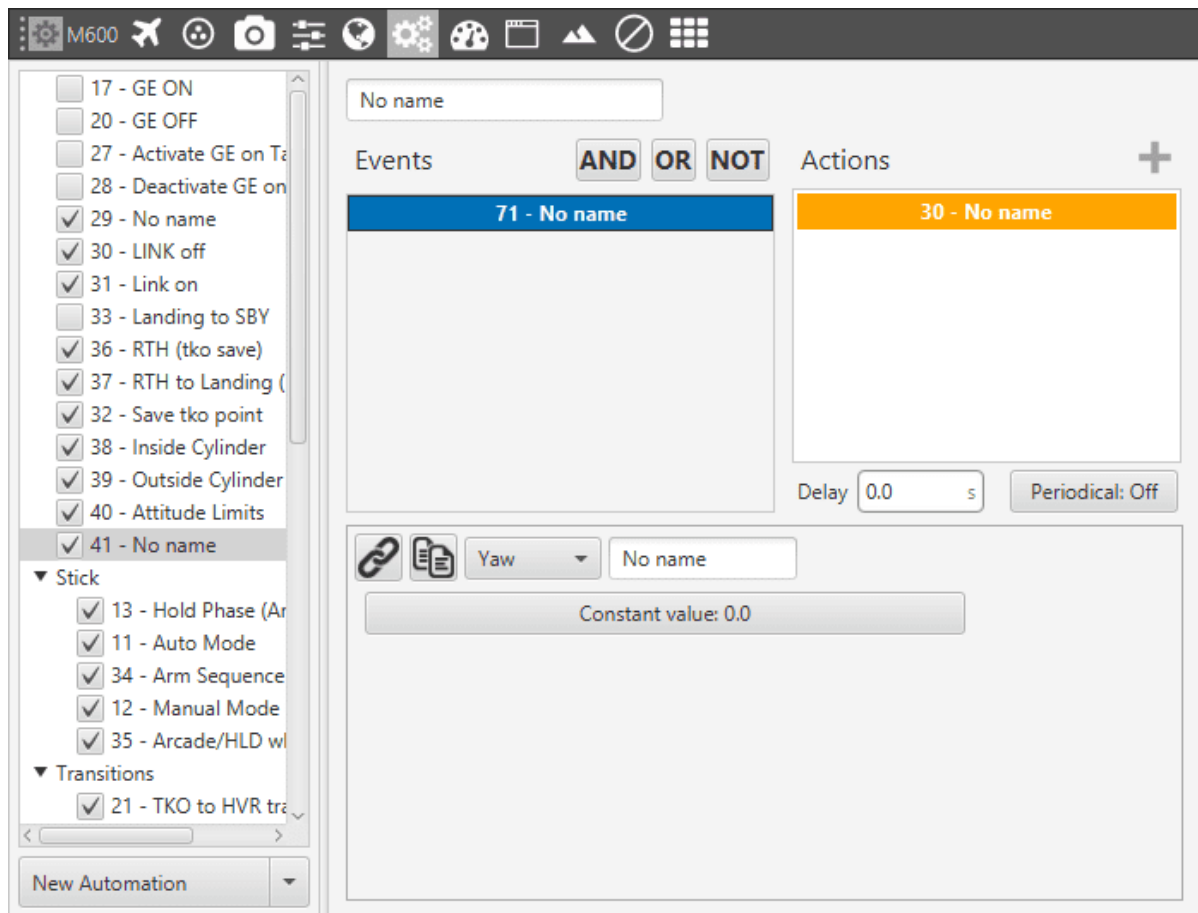
Allows the user to select a variable and save it in an user variable.



Action – Variable


7.2.6.2.32 Yaw

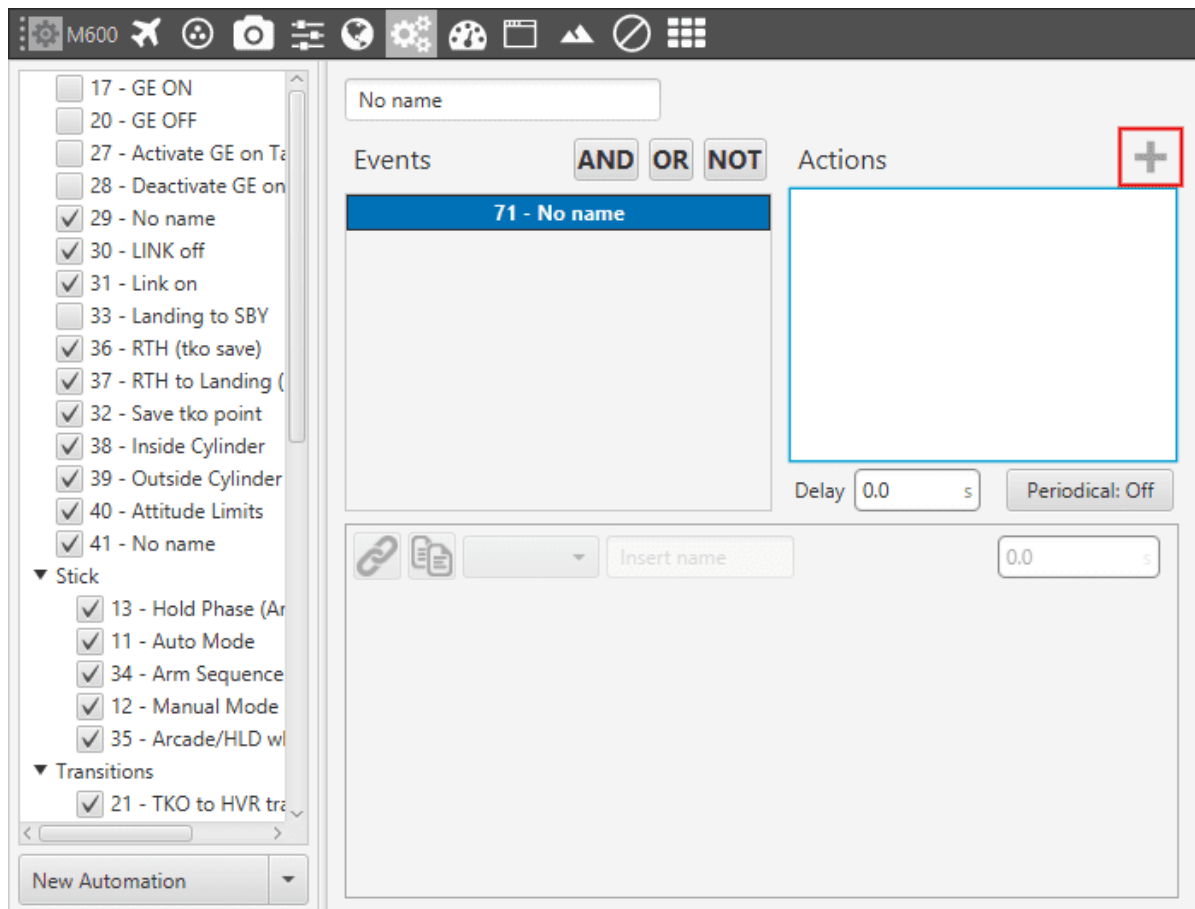
When this action is triggered the actual yaw can be commanded. This action is useful when flying without a magnetometer, as the user can establish the current yaw value when it is known.



Action – Yaw

The **Actions** box contains all the actions that will be performed when the event (or group of events) has been accomplished.

To create a new action press . When entering a new Event or Action it is possible to choose from one of the previously created on the system or to create a new one.

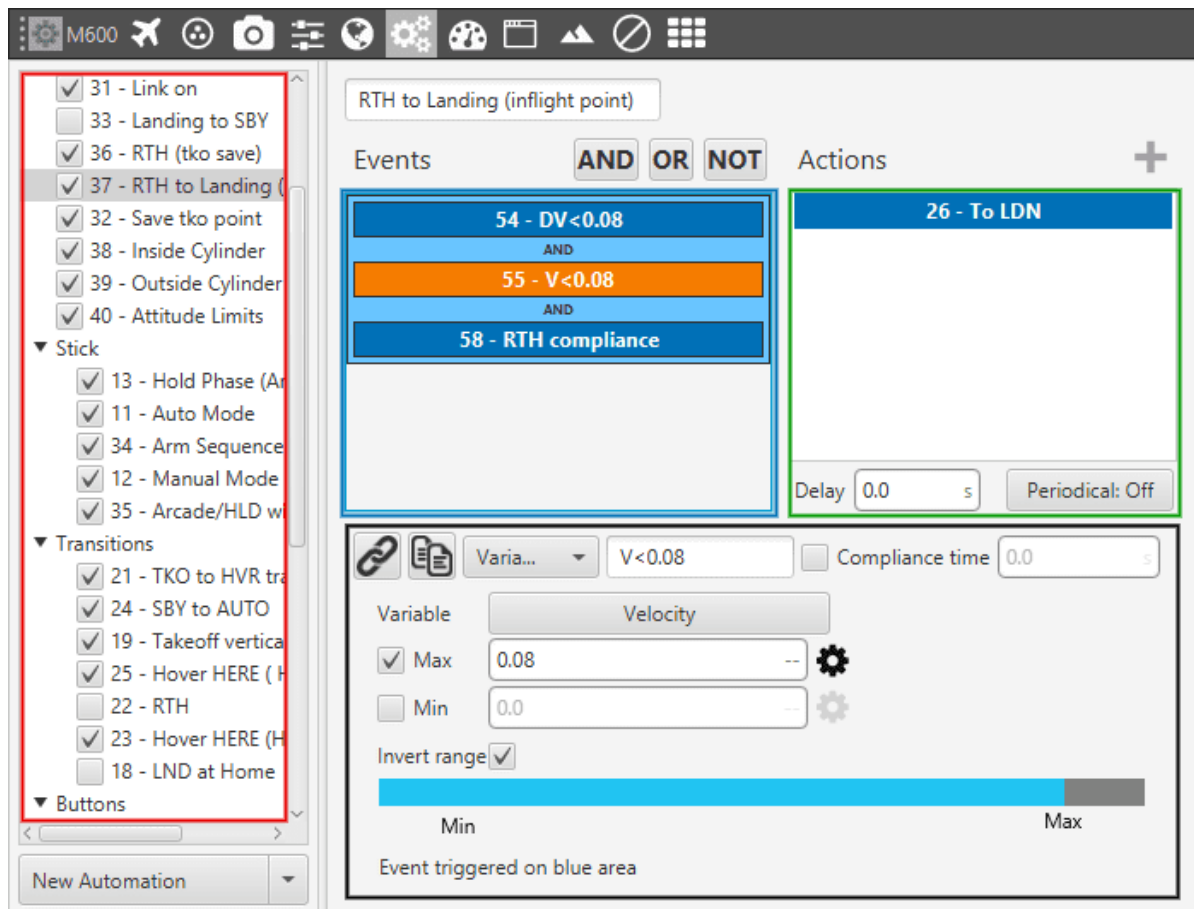


Automations Panel – Actions

When creating a new action, it is possible to select different types of actions, these are explained in the next sections.

Automations are actions that are carried out when a combination of events happen, i.e. when the events are accomplished the action is done. An example of what an automation could be is a change of phase when reaching a certain altitude and speed, moving a servo when a button is clicked and many other possible combinations. In this section all the possible events and actions will be explained in detail, so the user can combine them to create the automations that best suit their needs.

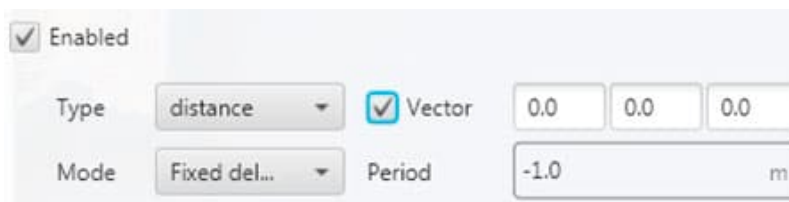
The following figure shows the layout of the automations menu, with a column for the events and another for the actions linked to these events.



Automation Display

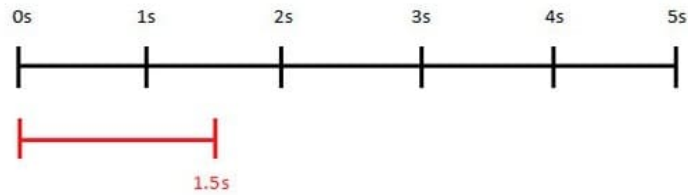
Automations (red) are a combination of events (blue) and actions (green). All actions will be performed on event or an event combination triggering. There are some parameters that can be configured in the events and actions menu and which are applicable independently of the type of event/action configured (black).

- **Delay:** time between the triggering of the event and the beginning of the action.
- **Compliance Time:** is a value related to the automations. Indicates how much time the event has to be accomplished in order to trigger the action. For example, if an event is to be above 100 meters and the value of Time is 3 seconds, the platform has to be above 100 meters during at least 3 second to trigger the action.
- **Periodical:** this menu is used to configure actions to take place periodically during the time that the events are active.



Automation Selection

The action can be configured to take place each certain distance or time. When using distance, the option Vector makes it possible to measure that distance along a direction specified by that vector. The two modes available for both time and distance are fixed delay and period. In order to explain the difference between them, the following figure is presented as an aid to the user.



Automation Selection

Let's consider that the system evaluates the automations each second (black line), and the automation that contains the periodical option is wanted to execute each 1.5 seconds (red line). In that case, the first action will be triggered at the second 1.5 but will be evaluated at second 2. The second time that the action is evaluated will depend on the mode: if fixed delay is selected, the evaluation of the action will be done 1.5 seconds after it was evaluated the first time, so that will be at second 3.5. On the other hand, if the mode is fixed period, the action will be evaluated 1.5 seconds after the first triggering (not evaluation) so that would be at the second 3. In the real praxis, the evaluation time for the automation is much lower than 1 second so the difference between the modes is much smaller.

Finally, automations can be grouped



7.2.6.3 Other Options

To create a new automation press **New Automation**, a new window will be displayed, users can select a previous one (if exists) or **Create new**.

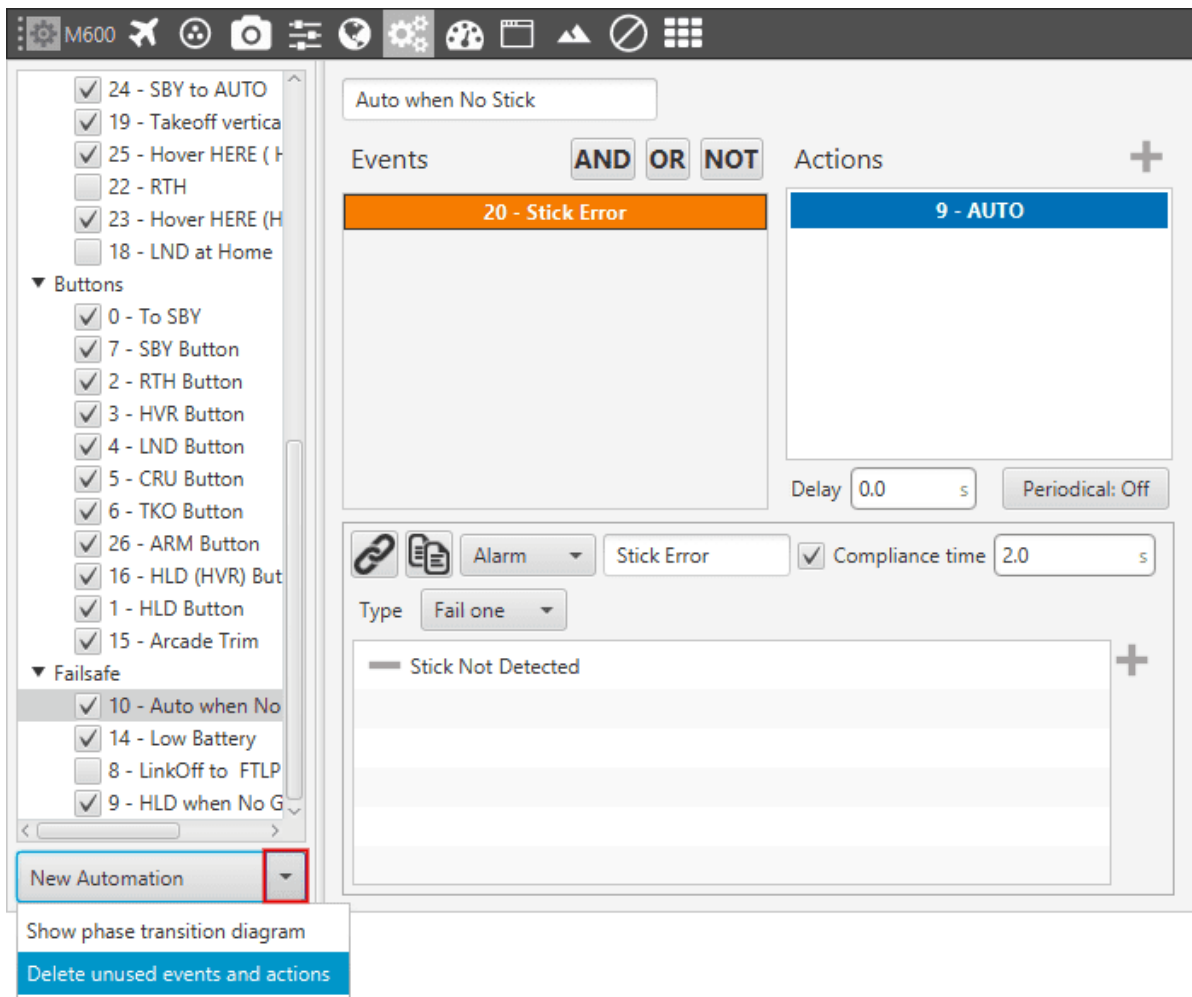
Right-clicking on an automation makes it possible to **Remove** it or to **Clone** it. When a clone of an automation is created, the changes made in the event panel will be applied to the other one and vice versa, while the actions can be different in each automation.

Automations also can be grouped by right-clicking in an automation and selecting the option **Change Group**. When a group is created, the rest of automations that the user wants to add to the group can be done by drag and drop.

Common configuration options are:

| Button | Description |
|---|--|
|  | Select an action or event from the available in the system. When modifying an action or event it will be modified in all automations where it is in use. |
|  | Clone an existing action or event creating a new one with same parameters configured on the start point. |

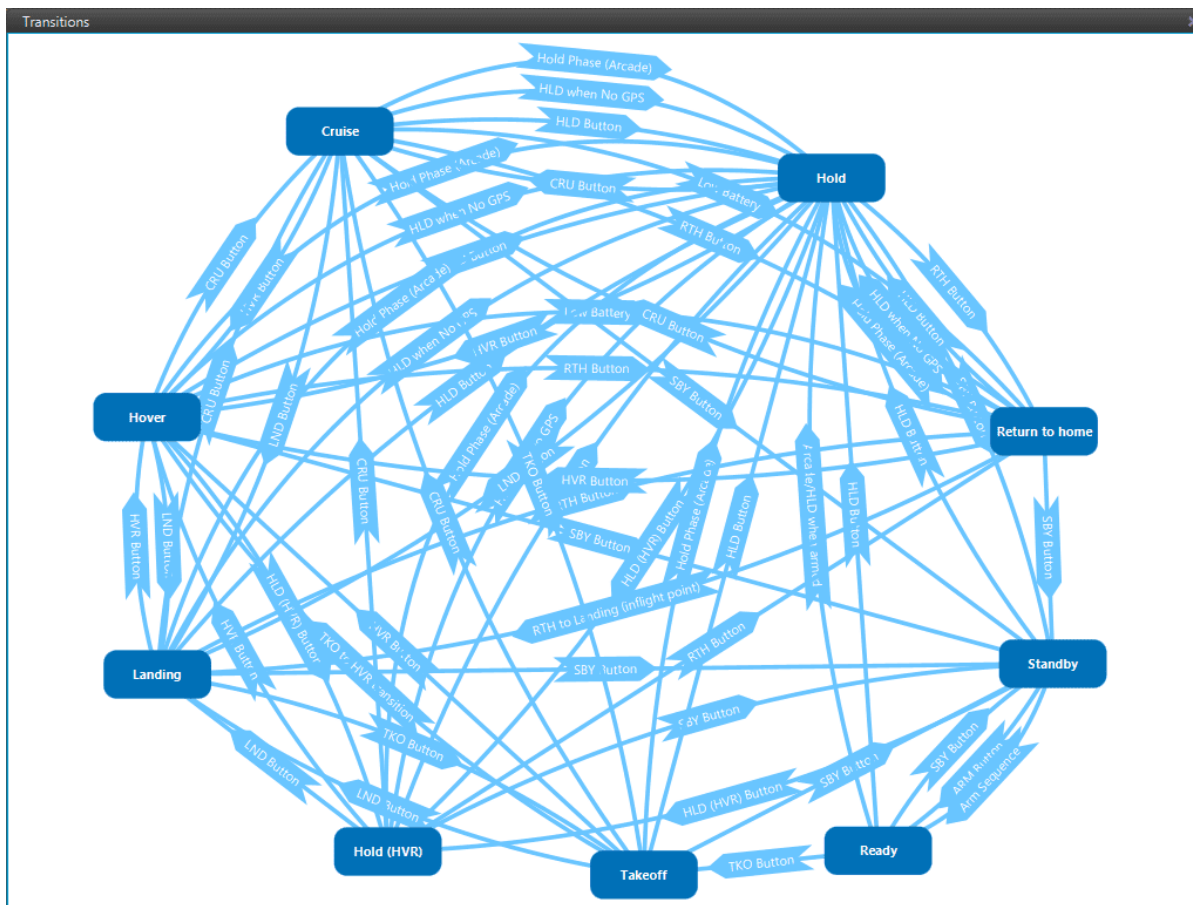
By pressing the drop down list next to **New Automation** button two additional options can be found.



Automation Drop Down List

Delete unused events and actions option deletes those events or actions that has been created but are not in use in any automation.

Show phase transition diagram option generates a diagram in which the phase changes through the different automations can be visualized. An example of such diagrams is shown in the figure below.



Phase Diagram

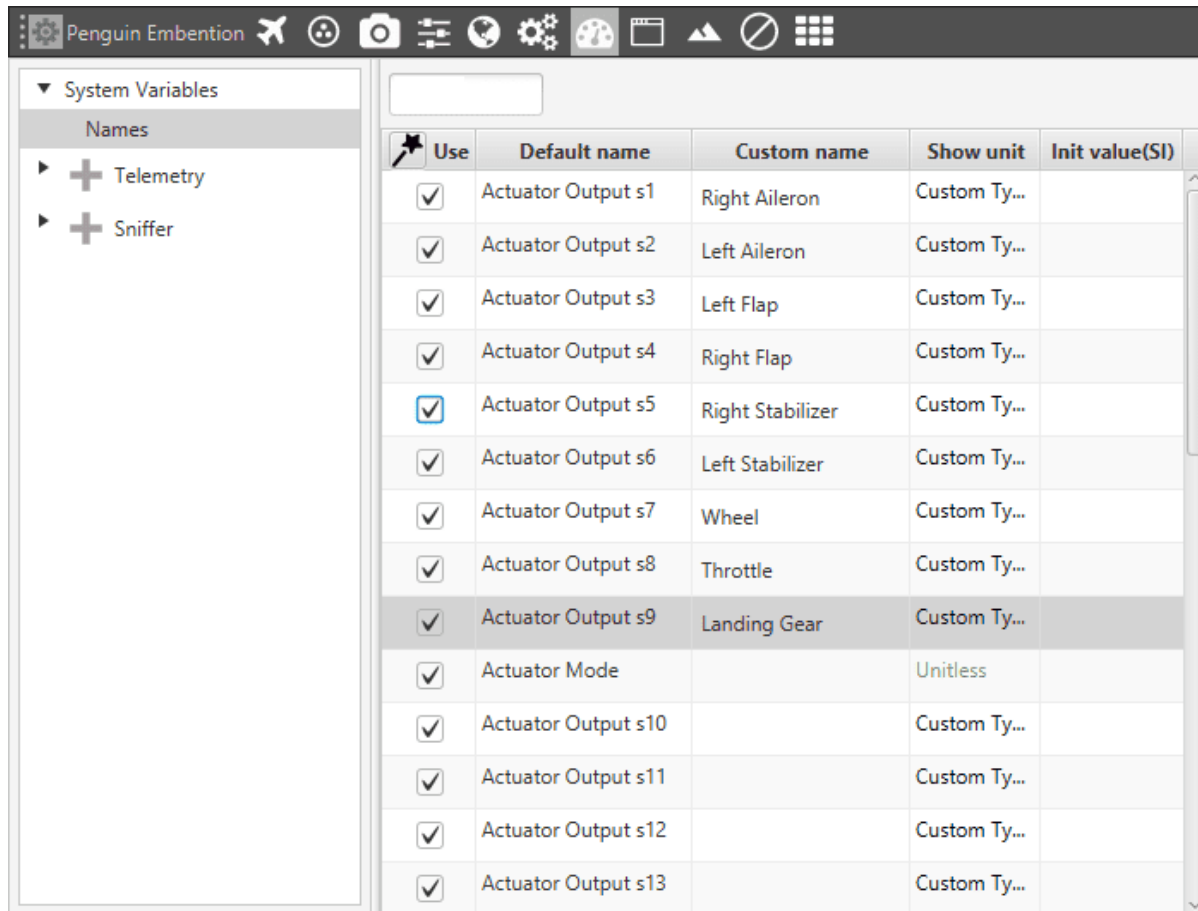
7.2.7 Variables

7.2.7.1 System Variables

7.2.7.1.1 Names

This menu is used to set a custom name for one of the system variables.

1. Click on the **Custom Name** cell of the desired variable and **introduce the new name** for it.
2. When the name is introduced press **Enter** to store the name on the system.
3. Press **Save** to save all changes.



Variable Name Customization

| Name | Type | Size |
|------|--------------|------|
| U | Unsigned int | 16 |
| R | Float | 32 |
| B | Bit | 1 |

Besides changing their name, the user can also configure the measurement units of the variables, as well as the initial value (expressed in SI units) they will have each time the system (re)starts, using the **Show Units** and the **Initial Value (SI)** cells.

7.2.7.2 Telemetry

Telemetry controls permit to configure data to be stored or transmitted on the system. There are 4 main items that can be configured within this panel:

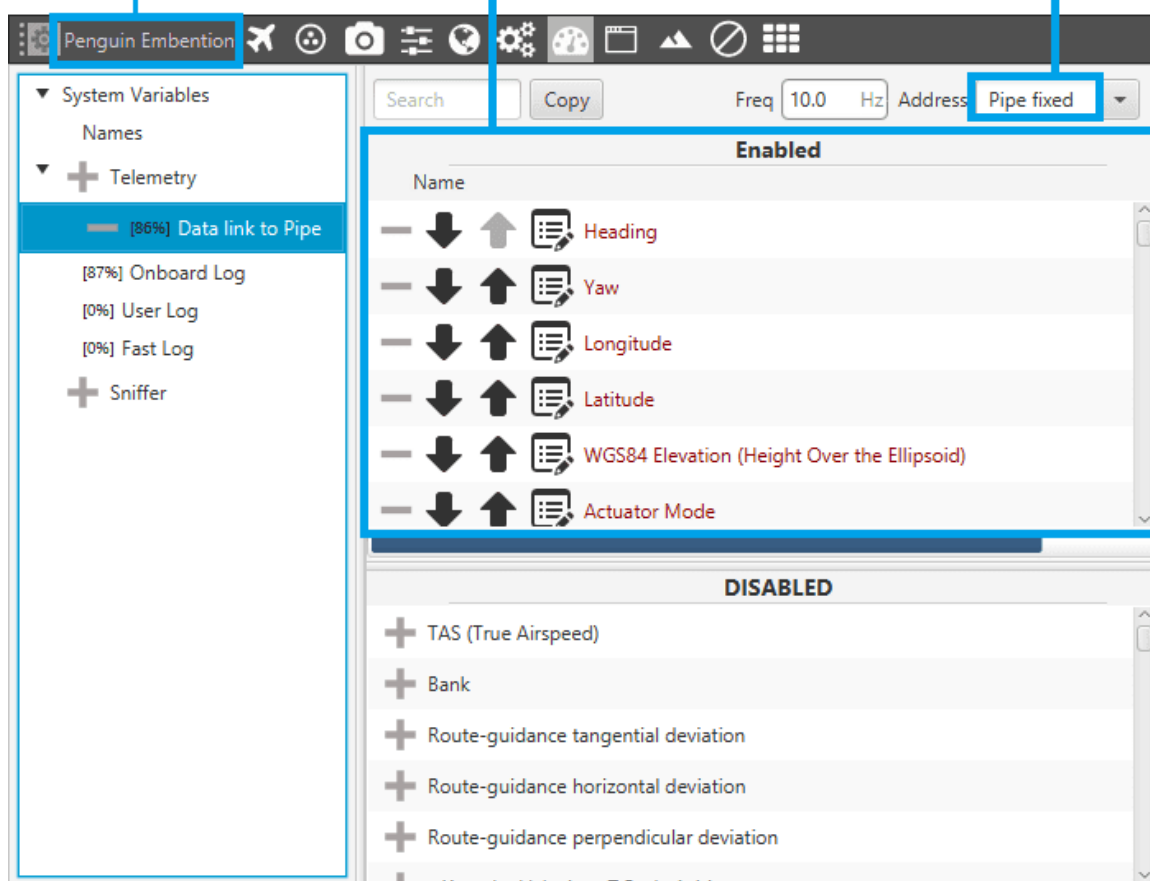
| Type | Description |
|-------------|--|
| Data Link | Configures the variables to send throughout the data link channel. |
| Onboard Log | Sets the variables to be stored on system Log. (on Veronte SD Card) |
| User Log | User Log for custom applications. |
| Fast Log | Saves data at the maximum frequency available on the system. Recording time depends on the selected variables. |

Configuration display permits to enable the desired variables for each telemetry file and to set the maximum and minimum values together with precision for each one.

7.2.7.2.1 Data Link

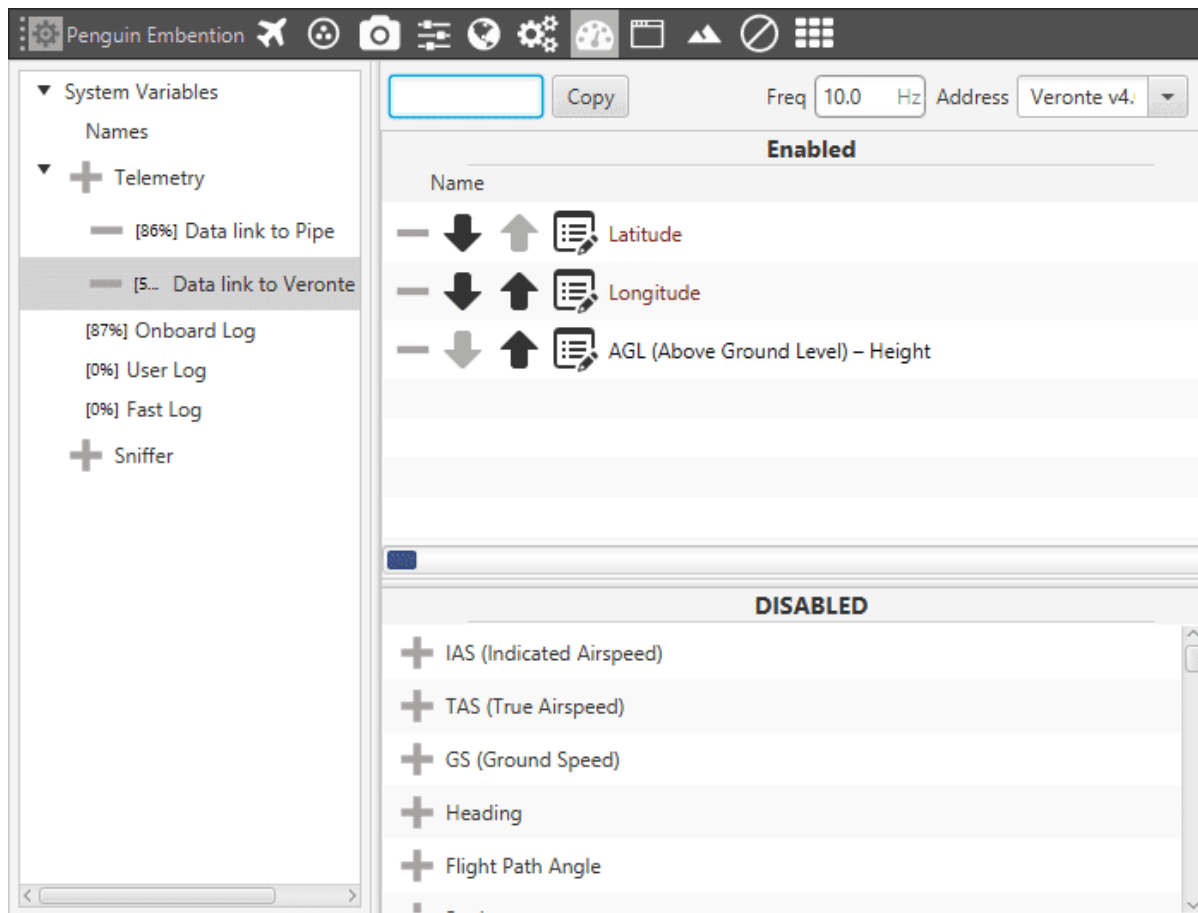
The Data Link contains the variables sent between Veronte Units and Veronte Pipe. By default, the system provides one Data Link that represents the connection between the air autopilot and the software (Pipe). Veronte Air sends the variables to Veronte Ground, being processed when they arrive there by Veronte Pipe. The variables indicated in red in a Data Link to Pipe are required for correct operation of Pipe.

Veronte will send all these variables to the Veronte with the corresponding Address



Telemetry Configuration Menu

Pipe permits the creation of more Data Links too. The user has to configure which unit sends the information and which receives it and the sending rate. As an example, another possible data link could be set between the Air and Ground autopilots directly (without Pipe) and used to send the position of the UAV to the Ground autopilot for the configuration of a tracker. This Data Link example is presented in the following figure.



Data Link (Ground/Air)

The autopilot **Penguin Embention** will send the Latitude, Longitude and AGL to the autopilot with address **Veronte v4.0 1085**. The sending rate of these data links can be modified by the user. In this example, the sending rate is 10 Hz. The unit that receives the telemetry has to configure its sniffer in order to store the data.

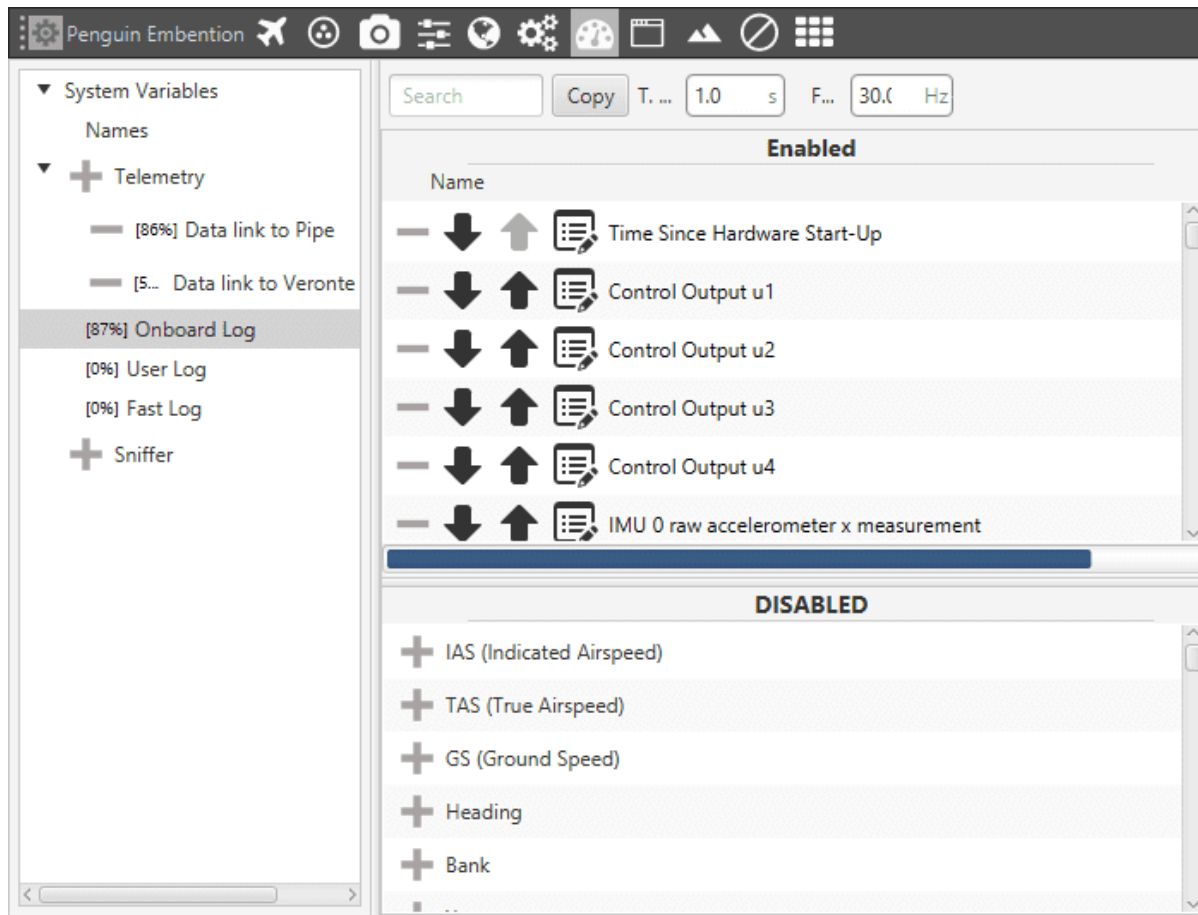
There is a special address called Broadcast. This means that every Veronte could receive the information of this data link (if they are configured to do so).

Warning: If the number of variables enabled for telemetry communication are higher than the maximum supported by the system, the latest variables will not be sent, so they will display a zero value if shown in the workspace.

Note: It is possible to create more than one data link associated to the same receiver address, and they can also have different sending rates. It could be useful in case one of the data links is almost full.

7.2.7.2.2 Onboard Log

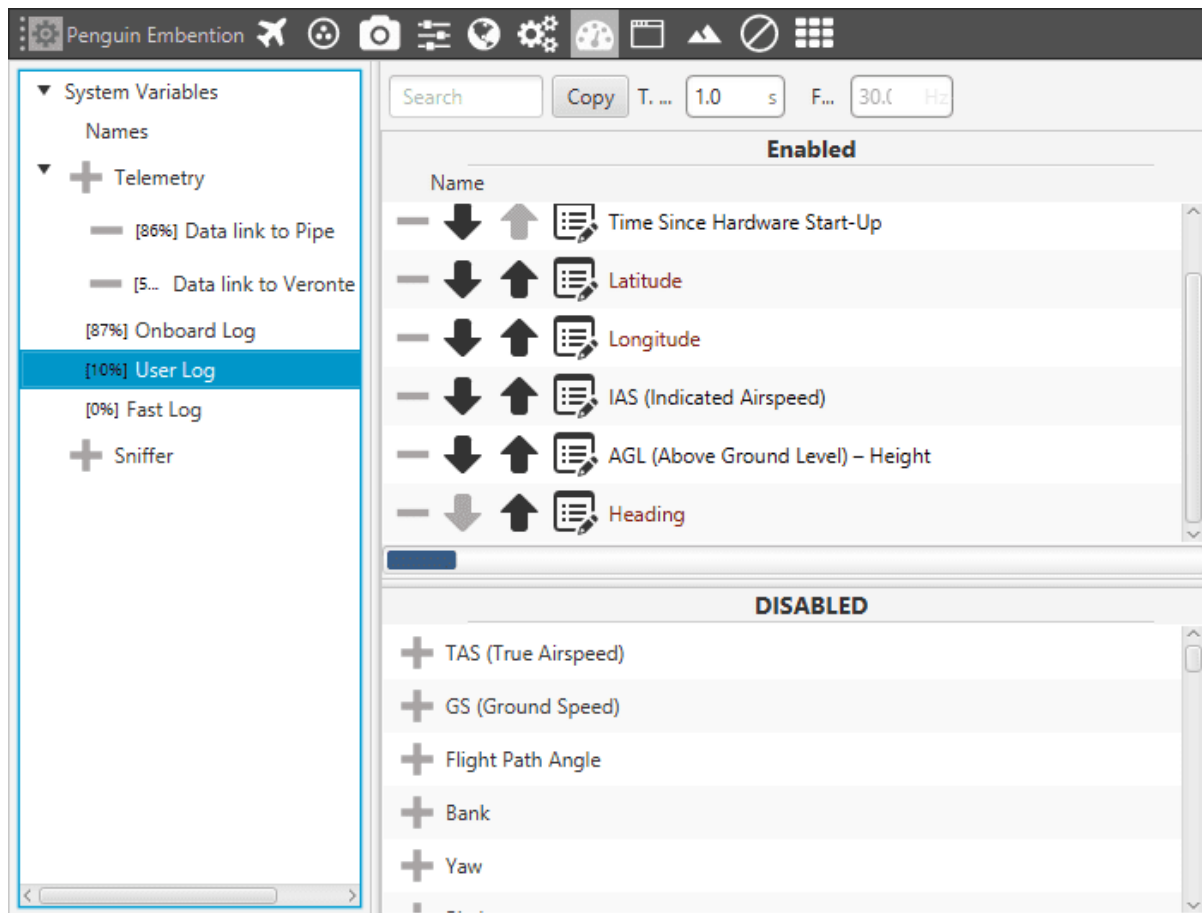
The Onboard Log determines the variables that are being stored on the autopilot SD Card. In this case, there are not sending/receiving units, so the only thing to configure here is the list of variables that will be saved on the autopilot internal memory for a further download and processing, as well as the writing frequency (30 Hz in the example of the image below).



Onboard Log

7.2.7.2.3 User Log

The user log contains the variables that are stored according to an automation created by the user. Considering an example, in a photogrammetry mission it is important to record the aircraft location when the photo is taken, so a user log could be used to record a certain set of variables (position, speed, direction...) each time a photo is taken.



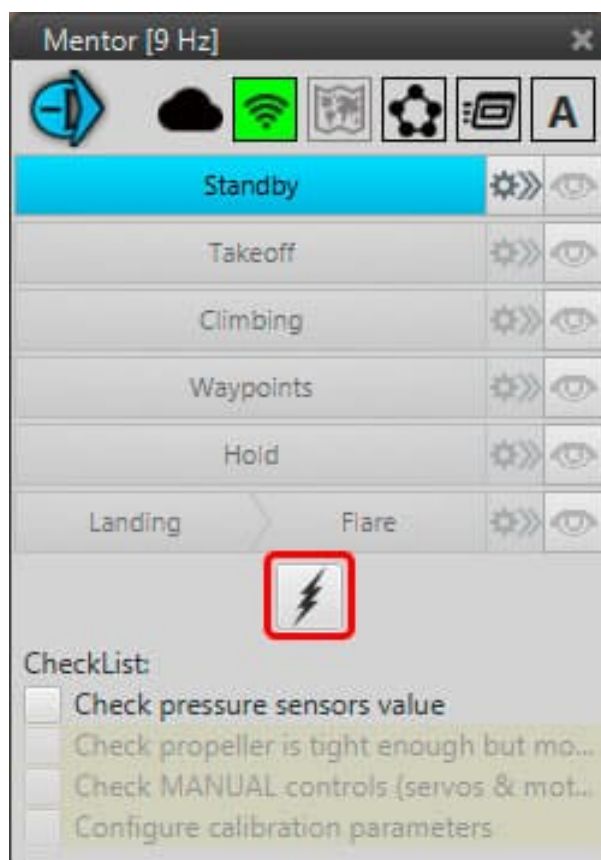
User Log

In order to create a User Log action where an entry is added to the log when a certain set of events are accomplished check *Automations*.

7.2.7.2.4 Fast Log

The fast log store the specified variables at the maximum rate available on the system. This tool could be used to save information in an operation that happens extremely fast, such as missile launching. The time that this logging process lasts depends on the number of variables being saved.

When a variable is enabled in the Fast Log, a new button will appear in the Veronte Panel. The user must click on it when he wants to activate the fast log tool.

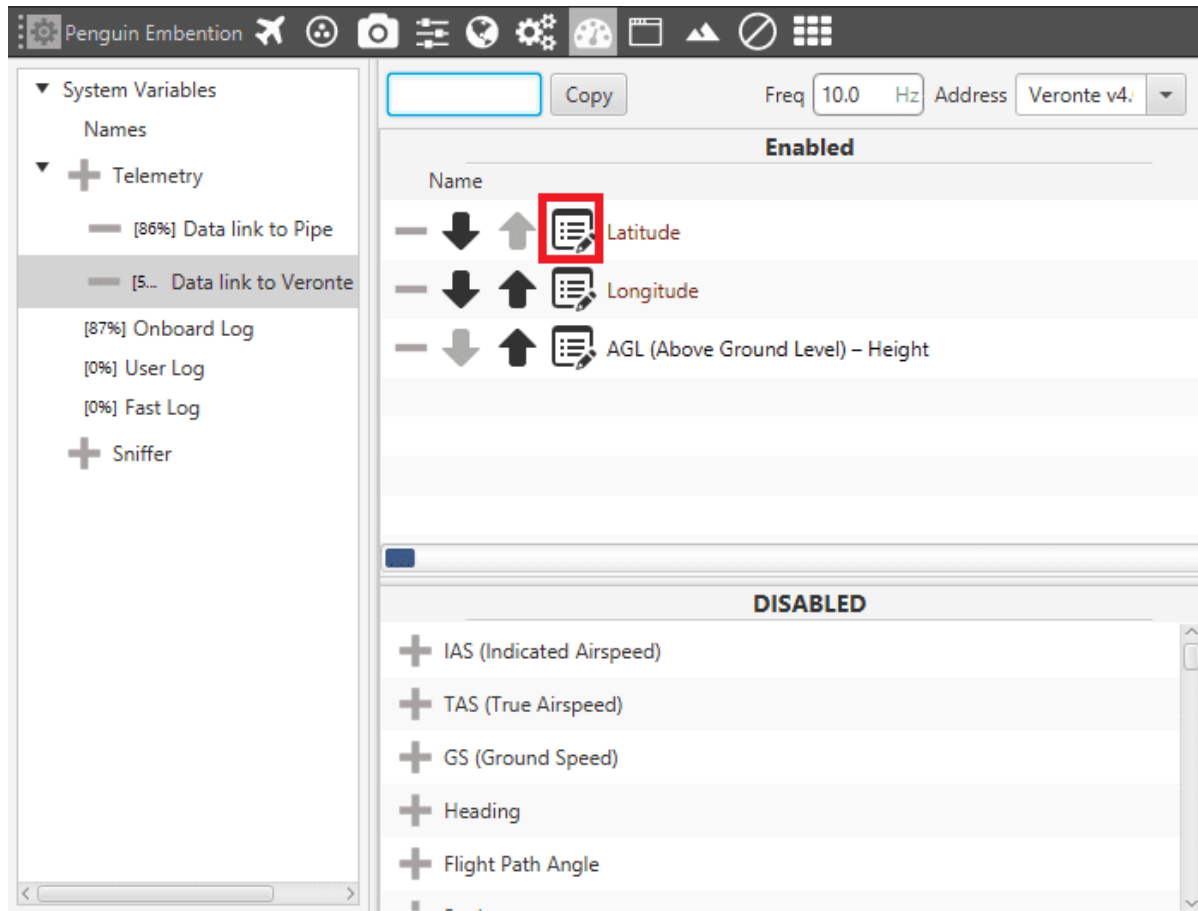


Fast Log button

The downloading of the information of an operation depends on how it has been stored, i.e depends on the type of log (data link, onboard, user or fast). Visit section 10 of the manual to see the information related to the postflight tools contained in Veronte Pipe.

7.2.7.2.5 Compressing options

Veronte includes some compression tools that may be useful for increasing the amount of information transmitted in a certain bandwidth or stored in a log. Each variable can be compressed separately in each log.



Compression options icon

There are different types of compression available:

The screenshot shows a configuration window for a variable named "Latitude". The "Type" is set to "Variable". Below the variable name, there are five radio button options for compression: "Uncompressed" (selected), "Compress", "Compress (Decimals)", "Compress (Bits signed)", and "Compress (Bits unsigned)". A bracket groups the last three options, with a text input field next to it containing the value "0". Below these options, the "Encode" section is labeled "DISABLED" in red. It contains two input fields for "Min" and "Max", both containing the value "0". The "Decode" section also contains two input fields for "Min" and "Max", both containing the value "0". At the bottom, there is an "Encode / Decode" input field containing the value "1.0".

Compression options

- **Uncompressed:** the variable is taken in its full length, with no value modification.
- **Compress (Bits signed):** specify the number of bits to be compressed to (negative values accepted). It is necessary that the user configures Encode/Decode options.
- **Compress (Bits unsigned):** specify the number of bits to be compressed to (no negative values accepted). It is necessary that the user configures Encode/Decode options.
- **Compress (Decimals):** the variable is compressed according to the number of decimals specified and the range specified (max and min values). The resultant compression (number of bits) follows the relation $(max - min) \cdot 10^{decimals}$ - which yields the encoding of the maximum value of the range (and the number of bits necessary for that). The range needs to be specified on the **Encode - Min/Max** field.
- **Encode/Decode:** these values are used to apply a scaling factor after the transformation from binary to decimal value, or before the transformation from decimal to binary value.

In the example shown below, the Heading variable with 3 decimals will be compressed, so instead of using 32 bits, it will only require 19 bits.

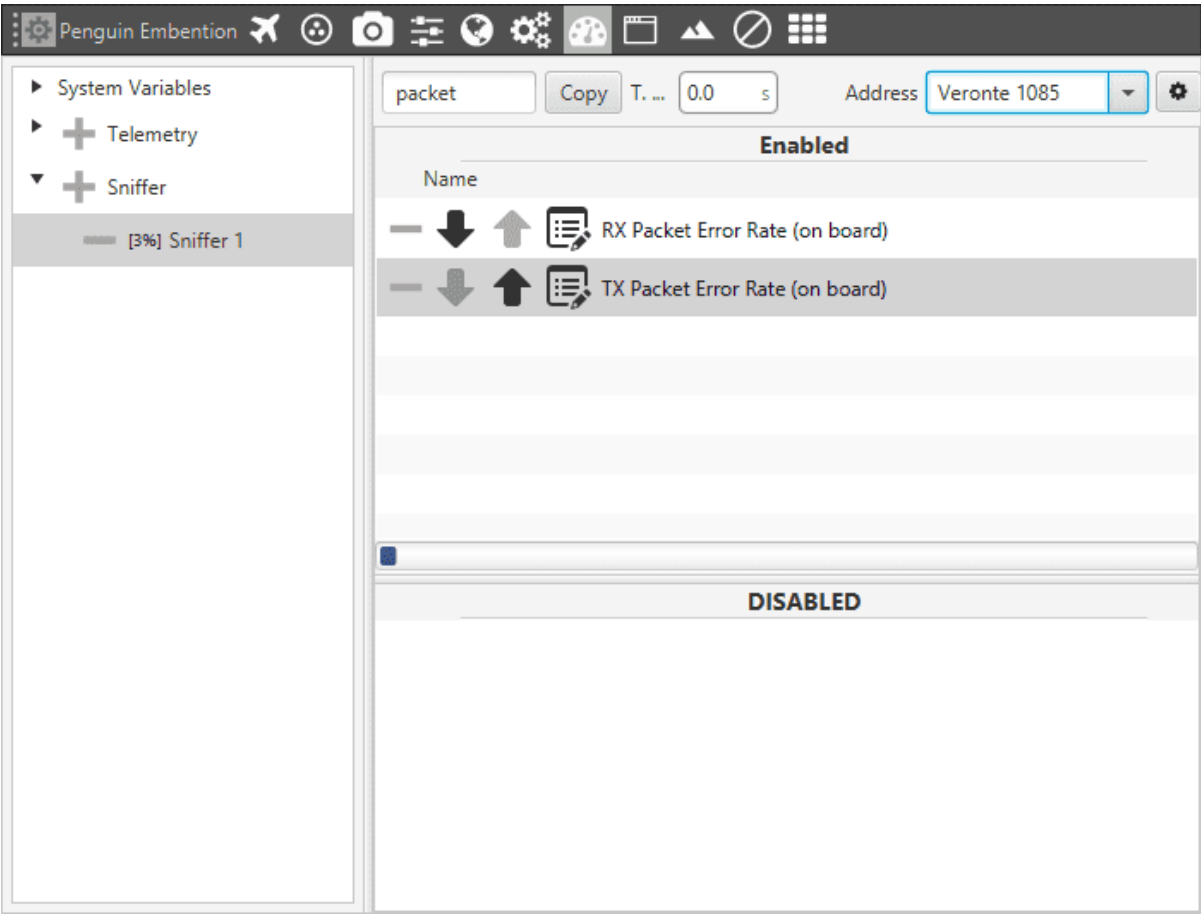
The screenshot shows a configuration window for a 'Variable' type. The variable is named 'Heading'. Under the 'Compression' section, the 'Compress (Decimals)' option is selected, and a value of '3' is entered in the adjacent text box. The 'Encode' section has 'Min' set to '0' and 'Max' set to '359.999'. The 'Decode' section has 'Min' set to '0' and 'Max' set to '0'. At the bottom, the 'Encode / Decode' ratio is set to '1.0'.

| Section | Option | Value |
|-----------------|--------------------------|------------------------------------|
| Compression | Uncompressed | <input type="radio"/> |
| | Compress | <input type="radio"/> |
| | Compress (Decimals) | <input checked="" type="radio"/> 3 |
| | Compress (Bits signed) | <input type="radio"/> |
| | Compress (Bits unsigned) | <input type="radio"/> |
| Encode | Min | 0 |
| | Max | 359.999 |
| Decode | Min | 0 |
| | Max | 0 |
| Encode / Decode | | 1.0 |

Compression example

7.2.7.3 Sniffer

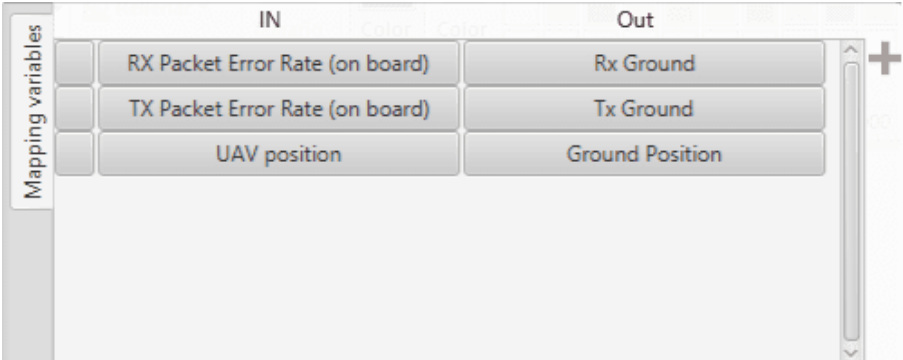
This menu is used to establish a telemetry communication between two autopilots. The autopilot being configured will “listen” the variables indicated in the window **Enabled**, from another autopilot whose address is indicated in **Address**. The sniffer is commonly used to make the aircraft listen the position of the ground station and the link quality.



Sniffer Configuration Menu

The source UAV, in this case, is the ground station (1085), which communicates to the Air UAV its position and some variables related to link quality (Rx and Tx Packet Error Rates). In **Mapping Variables** (Select **Configure**), the ones send by the ground UAV are indicated in the column **IN**, and they are stored in the variables indicated in **Out** for its later use by the air autopilot.

The sniffer is configured so that the air autopilot has information about the state of the communications, and it could perform an action when the link is lost. The aerial platform also receives information about the ground station position, so it can perform a mission in relation to that point.



Mapping Variables

The Veronte Unit that sends the data has to be configured as well, in the Telemetry Panel. That unit will send telemetry through a Data Link.

7.2.7.4 List of Variables

Warning: Bit Variables displayed on Labels (see ref Workspace – Gauge Display) will be shown as Red/Green depending on its state. Red stands for 0 and Green for 1, changing the name displayed accordingly to the BIT value.

7.2.7.4.1 32 VAR

| ID | Name | Units/Values | Description |
|----|--|--------------------|---|
| 0 | IAS (Indicated Air Speed) | m/s | Pitot-static measurement speed |
| 1 | TAS (True Air Speed) | m/s | Speed relative to the airmass in which the vehicle is moving (I |
| 2 | GS (Ground Speed) | m/s | Horizontal speed, relative to the ground |
| 3 | Heading | rad | Direction in which the vehicle velocity vector is pointing |
| 4 | Flight Path Angle | rad | Angle between velocity vector and local horizontal line |
| 5 | Bank | rad | Velocity vector lateral component |
| 6 | Yaw | rad | Angle around the Vertical axis of the vehicle |
| 7 | Pitch | rad | Angle around the Transverse axis of the vehicle |
| 8 | Roll | rad | Angle around the Longitudinal axis of the vehicle |
| 9 | Route-guidance tangential deviation | m | Tangential distance to the desired position (guidance) |
| 10 | Route-guidance horizontal deviation | m | Horizontal distance to the desired position (guidance) |
| 11 | Route-guidance perpendicular deviation | m | Perpendicular distance to the desired position (guidance) |
| 12 | p (Angular Velocity - X Body Axis) | rad/s | Angular velocity around longitudinal axis |
| 13 | q (Angular Velocity - Y Body Axis) | rad/s | Angular velocity around lateral axis |
| 14 | r (Angular Velocity - Z Body Axis) | rad/s | Angular velocity around vertical axis |
| 15 | Forward Acceleration – X Body Axis | m/s ² | Acceleration in the X-axis |
| 16 | Right Acceleration – Y Body Axis | m/s ² | Acceleration in the Y-axis |
| 17 | Bottom Acceleration – Z Body Axis | m/s ² | Acceleration in the Z-axis |
| 18 | RPM | rad/s (RDS) | Revolutions per minute configurable for external sensor |
| 19 | Front GV (Ground Velocity) | m/s | GV vector X component |
| 20 | Lateral GV (Ground Velocity) | m/s | GV vector Y component |
| 21 | Velocity | m/s | Velocity vector module |
| 22 | Forward Load Factor – X Body Axis | customType | G-force in X body axis |
| 23 | Right Load Factor – Y Body Axis | customType | G-force in Y body axis |
| 24 | Bottom Load Factor – Z Body Axis | customType | G-force in Z body axis |
| 25 | Tangential Acceleration | m/s ² | Absolute acceleration for tangential direction |
| 28 | Co-yaw | rad | Acrobatic Yaw with Body Z' axis pointing to X |
| 29 | Co-pitch | rad | Acrobatic Pitch with Body X' axis pointing to -Z |
| 30 | Co-roll | rad | Acrobatic Roll with Y' keeping same as Y |
| 31 | Angular Acceleration - X Body Axis | rad/s ² | Acceleration around the longitudinal axis |
| 32 | Angular Acceleration - Y Body Axis | rad/s ² | Acceleration around the lateral axis |
| 33 | Angular Acceleration - Z Body Axis | rad/s ² | Acceleration around the vertical axis |
| 34 | Body to NED quaternion qs | customType | First component of body to NED orientation quaternion |
| 35 | Body to NED quaternion qi | customType | Second component of body to NED orientation quaternion |
| 36 | Body to NED quaternion qj | customType | Third component of body to NED orientation quaternion |
| 37 | Body to NED quaternion qk | customType | Fourth component of body to NED orientation quaternion |
| 40 | RSSI | percentage | Received Signal Strength Indicator |
| 42 | SCI-A Rx rate (4G) | bytes/s | 4G link reception byte rate |
| 43 | SCI-A Tx rate (4G) | bytes/s | 4G link transmission byte rate |
| 44 | SCI-B Rx rate (LOS) | bytes/s | Radio link reception byte rate |

Table 7 – continued from previous page

| ID | Name | Units/Values | Description |
|-----|--|------------------|--|
| 45 | SCI-B Tx rate (LOS) | bytes/s | Radio link transmission byte rate |
| 46 | SCI-C Rx rate (RS485) | bytes/s | RS485 communication reception byte rate |
| 47 | SCI-C Tx rate (RS485) | bytes/s | RS485 communication transmission byte rate |
| 48 | SCI-D Rx rate (RS232) | bytes/s | RS232 communication reception byte rate |
| 49 | SCI-D Tx rate (RS232) | bytes/s | RS232 communication transmission byte rate |
| 50 | CAN-A Tx rate | pkts/s | CAN-A transmission packet rate |
| 51 | CAN-B Tx rate | pkts/s | CAN-B transmission packet rate |
| 52 | CAN-A Tx skip rate | pkts/s | CAN-A messages delayed because no mailbox is available for |
| 53 | CAN-B Tx skip rate | pkts/s | CAN-B messages delayed because no mailbox is available for |
| 56 | Yaw rate | rad/s | Rate of change of the yaw angle |
| 57 | Pitch rate | rad/s | Rate of change of the pitch angle |
| 58 | Roll rate | rad/s | Rate of change of the roll angle |
| 100 | Desired IAS (Indicated Air Speed) | m/s | Commanded IAS from guidance |
| 101 | Desired TAS (True Air Speed) | m/s | Commanded TAS from guidance |
| 102 | Desired GS (Ground Speed) | m/s | Commanded GS from guidance |
| 103 | Desired Heading | rad | Commanded Heading from guidance |
| 104 | Desired Flight Path Angle | rad | Commanded Flight Path Angle from guidance |
| 105 | Desired Bank | rad | Commanded Bank from guidance |
| 106 | Desired Yaw | rad | Commanded Yaw from guidance |
| 107 | Desired Pitch | rad | Commanded Pitch from guidance |
| 108 | Desired Roll | rad | Commanded Roll from guidance |
| 112 | Desired p (Angular Velocity - X Body Axis) | rad/s | Commanded angular velocity around longitudinal axis |
| 113 | Desired q (Angular Velocity - Y Body Axis) | rad/s | Commanded angular velocity around lateral axis |
| 114 | Desired r (Angular Velocity - Z Body Axis) | rad/s | Commanded angular velocity around vertical axis |
| 115 | Desired Foward Acceleration – X Body Axis | m/s ² | Commanded Forward Acceleration from guidance |
| 116 | Desired Right Acceleration – Y Body Axis | m/s ² | Commanded Right Acceleration from guidance |
| 117 | Desired Bottom Acceleration – Z Body Axis | m/s ² | Commanded Bottom Acceleration from guidance |
| 118 | Desired RPM | rad/s | Commanded RPM from guidance |
| 119 | Desired Front GV (Ground Velocity) | m/s | Commanded Front GV from guidance |
| 120 | Desired Lateral GV (Ground Velocity) | m/s | Commanded Lateral GV from guidance |
| 121 | Desired Velocity | m/s | Commanded Velocity from guidance |
| 122 | Desired Forward Load Factor – X Body Axis | customType | Commanded Forward Load Factor from guidance |
| 123 | Desired Right Load Factor – Y Body Axis | customType | Commanded Right Load Factor from guidance |
| 124 | Desired Bottom Load Factor – Z Body Axis | customType | Commanded Bottom Load Facto from guidance |
| 125 | Desired Tangential Acceleration | m/s ² | Commanded Tangential Acceleration from guidance |
| 126 | Energy Rate Error | customType | Rate of change of the Total System Energy |
| 127 | Energy Distribution Error | customType | Distribution of system energy between kinetical and geopoten |
| 128 | Desired co-yaw | rad | Commanded co-yaw from guidance |
| 129 | Desired co-pitch | rad | Commanded co-pitch from guidance |
| 130 | Desired co-roll | rad | Commanded co-roll from guidance |

| | | | |
|-----|--|-----|--|
| 200 | Desired North GV (Ground Velocity) | m/s | Commanded North (NED Coordinates system) GV from |
| 201 | Desired East GV (Ground Velocity) | m/s | Commanded East (NED Coordinates system) GV from g |
| 202 | Desired Down GV (Ground Velocity) | m/s | Commanded Down (NED Coordinates system) GV from |
| 203 | Desired 2D MSL (Heigh Above Mean Sea Level) | m | Commanded MSL from guidance in 2D height mode |
| 204 | Desired 2D AGL (Above Ground Level) – Height | m | Commanded AGL from guidance in 2D height mode |
| 205 | Desired 2D WGS84 Elevation (Height Over The Ellipsoid) | m | Commanded WGS84 Elevation from guidance in 2D heig |
| 206 | Desired Longitude | rad | Commanded Longitude from guidance |

con

Table 8 – continued from previous page

| | | | |
|-----|--|------------------|--|
| 207 | Desired Latitude | rad | Commanded Latitude from guidance |
| 208 | Desired WGS84 Elevation (Height Over The Ellipsoid) | m | Commanded WGS84 Elevation from guidance |
| 209 | Desired MSL (Height Above Mean Sea Level) – Altitude | m | Commanded MSL Altitude from guidance |
| 210 | Desired AGL (Above Ground Level) – Height | m | Commanded AGL Altitude from guidance |
| 250 | Guidance north position error | m | Difference from Desired and actual north position |
| 251 | Guidance east position error | m | Difference from Desired and actual east position |
| 252 | Guidance down position error | m | Difference from Desired and actual down position |
| 253 | Guidance PID north desired velocity | m/s | Difference from Desired and actual PID north velocity |
| 254 | Guidance PID east desired velocity | m/s | Difference from Desired and actual PID east velocity |
| 255 | Guidance PID down desired velocity | m/s | Difference from Desired and actual PID down velocity |
| 256 | Desired velocity X body axis | m/s | Commanded velocity in X-axis from guidance |
| 257 | Desired velocity Y body axis | m/s | Commanded velocity in Y-axis from guidance |
| 258 | Desired velocity Z body axis | m/s | Commanded velocity in Z-axis from guidance |
| 259 | External yaw | rad | Yaw from external navigation source |
| 260 | External pitch | rad | Pitch from external navigation source |
| 261 | External roll | rad | Roll from external navigation source |
| 262 | External Roll Rate | rad/s | Roll rate from external navigation source |
| 263 | External Pitch Rate | rad/s | Pitch rate from external navigation source |
| 264 | External Yaw Rate | rad/s | Yaw rate from external navigation source |
| 265 | External Velocity North | m/s | Velocity North from external navigation source |
| 266 | External Velocity East | m/s | Velocity East from external navigation source |
| 267 | External Velocity Down | m/s | Velocity Down from external navigation source |
| 268 | External acceleration x body axis | m/s ² | Acceleration x body axis from external navigation source |
| 269 | External acceleration y body axis | m/s ² | Acceleration y body axis from external navigation source |
| 270 | External acceleration z body axis | m/s ² | Acceleration z body axis from external navigation source |
| 271 | External GPS Time of Week | s | GNSS Time of week from external navigation source |
| 300 | Time since Hardware Start-Up | s | Time spent since power-on of the system |
| 301 | Used Memory Space | byte | SD used memory space |
| 302 | Free Memory Space | byte | SD free memory space |
| 303 | Dynamic Pressure | Pa | Physical measurement from Pitot (dynamic preasure) |
| 304 | Static Pressure | Pa | Physical measurement from Pitot (static preasure) |
| 305 | Internal Temperature | K | Physical measurement from internal sensors |
| 306 | External Temperature | K | Physical measurement from Veronte sensors |
| 307 | Accelerometer – X Body Axis | m/s ² | Accelerometer measurement for X axis |
| 308 | Accelerometer – Y Body Axis | m/s ² | Accelerometer measurement for Y axis |
| 309 | Accelerometer – Z Body Axis | m/s ² | Accelerometer measurement for Z axis |
| 310 | Gyroscope – X Body Axis | rad/s | Gyroscope measurement for X axis |
| 311 | Gyroscope – Y Body Axis | rad/s | Gyroscope measurement for Y axis |
| 312 | Gyroscope – Z Body Axis | rad/s | Gyroscope measurement for Z axis |
| 313 | Magnetometer – X Body Axis | T | Magnetometer measurement for X axis |
| 314 | Magnetometer – Y Body Axis | T | Magnetometer measurement for Y axis |
| 315 | Magnetometer – Z Body Axis | T | Magnetometer measurement for Z axis |
| 322 | Internal magnetometer raw X in SI | T | Magnetometer raw measurement for X axis |
| 323 | Internal magnetometer raw Y in SI | T | Magnetometer raw measurement for Y axis |
| 324 | Internal magnetometer raw Z in SI | T | Magnetometer raw measurement for Z axis |
| 325 | Internal magnetometer temperature | K | Magnetometer temperature |
| 326 | External LIS3MDL magnetometer raw X in SI | T | LIS3MDL external Magnetometer raw measurement for X axis |
| 327 | External LIS3MDL magnetometer raw Y in SI | T | LIS3MDL external Magnetometer raw measurement for Y axis |
| 328 | External LIS3MDL magnetometer raw Z in SI | T | LIS3MDL external Magnetometer raw measurement for Z axis |
| 329 | External LIS3MDL magnetometer temperature | K | LIS3MDL external Magnetometer temperature |

con

Table 8 – continued from previous page

| | | | |
|-----|---|------------------|---|
| 330 | IMU 0 raw accelerometer x measurement | m/s ² | IMU 0 raw accelerometer x measurement |
| 331 | IMU 0 raw accelerometer y measurement | m/s ² | IMU 0 raw accelerometer y measurement |
| 332 | IMU 0 raw accelerometer z measurement | m/s ² | IMU 0 raw accelerometer z measurement |
| 333 | IMU 0 raw gyroscope x measurement | rad/s | IMU 0 raw gyroscope x measurement |
| 334 | IMU 0 raw gyroscope y measurement | rad/s | IMU 0 raw gyroscope y measurement |
| 335 | IMU 0 raw gyroscope z measurement | rad/s | IMU 0 raw gyroscope z measurement |
| 336 | IMU 0 temperature measurement | K | IMU 0 temperature measurement |
| 337 | IMU 1 raw accelerometer x measurement | m/s ² | IMU 1 raw accelerometer x measurement |
| 338 | IMU 1 raw accelerometer y measurement | m/s ² | IMU 1 raw accelerometer y measurement |
| 339 | IMU 1 raw accelerometer z measurement | m/s ² | IMU 1 raw accelerometer z measurement |
| 340 | IMU 1 raw gyroscope x measurement | rad/s | IMU 1 raw gyroscope x measurement |
| 341 | IMU 1 raw gyroscope y measurement | rad/s | IMU 1 raw gyroscope y measurement |
| 342 | IMU 1 raw gyroscope z measurement | rad/s | IMU 1 raw gyroscope z measurement |
| 343 | IMU 1 temperature measurement | K | IMU 1 temperature measurement |
| 344 | Static pressure sensor (MS56) raw measurement | Pa | Static pressure sensor MS56 raw measurement |
| 345 | Static pressure sensor (MS56) temperature | K | Static pressure sensor MS56 temperature |
| 346 | Dynamic pressure sensor raw measurement | Pa | Dynamic pressure sensor raw measurement |
| 347 | Dynamic pressure sensor temperature | K | Dynamic pressure sensor temperature |
| 348 | Static pressure sensor (HSC) raw measurement | Pa | Static pressure sensor 0 raw measurement |
| 349 | Static pressure sensor (HSC) temperature | K | Static pressure sensor 0 temperature |
| 350 | Vectornav Message Frequency | Hz | Frequency at which external navigation source VectorNav |
| 351 | Vectornav Raw Acc x measurement | m/s ² | Raw accelerometer X measurement from external navigation |
| 352 | Vectornav Raw Acc y measurement | m/s ² | Raw accelerometer Y measurement from external navigation |
| 353 | Vectornav Raw Acc z measurement | m/s ² | Raw accelerometer Z measurement from external navigation |
| 354 | Vectornav Raw Gyr x measurement | m/s ² | Raw gyroscope X measurement from external navigation |
| 355 | Vectornav Raw Gyr y measurement | m/s ² | Raw gyroscope Y measurement from external navigation |
| 356 | Vectornav Raw Gyr z measurement | m/s ² | Raw gyroscope Z measurement from external navigation |
| 357 | External HSC magnetometer raw X in SI | T | HSCDTD008A external Magnetometer raw measurement |
| 358 | External HSC magnetometer raw Y in SI | T | HSCDTD008A external Magnetometer raw measurement |
| 359 | External HSC magnetometer raw Z in SI | T | HSCDTD008A external Magnetometer raw measurement |
| 360 | External HSC magnetometer temperature | K | HSCDTD008A external Magnetometer temperature |
| 361 | IMU 2 raw accelerometer x measurement | m/s ² | IMU 2 raw accelerometer x measurement |
| 362 | IMU 2 raw accelerometer y measurement | m/s ² | IMU 2 raw accelerometer y measurement |
| 363 | IMU 2 raw accelerometer z measurement | m/s ² | IMU 2 raw accelerometer z measurement |
| 364 | IMU 2 raw gyroscope x measurement | rad/s | IMU 2 raw gyroscope x measurement |
| 365 | IMU 2 raw gyroscope y measurement | rad/s | IMU 2 raw gyroscope y measurement |
| 366 | IMU 2 raw gyroscope z measurement | rad/s | IMU 2 raw gyroscope z measurement |
| 367 | IMU 2 temperature measurement | K | IMU 2 temperature measurement |
| 368 | Static pressure sensor (DPS310) raw measurement | Pa | Static pressure sensor DPS310 raw measurement |
| 369 | Static pressure sensor (DPS310) temperature | K | Static pressure sensor DPS310 temperature |
| 370 | Magnetometer 4 raw measure X converted to SI | T | Magnetometer 4 raw measurement for X axis converted to SI |
| 371 | Magnetometer 4 raw measure Y converted to SI | T | Magnetometer 4 raw measurement for Y axis converted to SI |
| 372 | Magnetometer 4 raw measure Z converted to SI | T | Magnetometer 4 raw measurement for Z axis converted to SI |
| 373 | Magnetometer 4 temperature | K | Magnetometer 4 temperature |
| 400 | Power Input | V | Power received by Veronte |
| 401 | Power Comicro 3.3V | V | Power received by Veronte through 3.3V port |
| 402 | Power 5V | V | Power received by Veronte through 5V port |
| 403 | SUC Power Input | V | Power received by Veronte SUC |
| 404 | Power 3.6V | V | Power received by Veronte through 3.6V port |
| 405 | CPU Temperature | K | Internal computer temperature |

| | | | |
|-----------|---|------------|--|
| 500 | Longitude | rad | East-West geographic coordinate |
| 501 | Latitude | rad | North-South geographic coordinate |
| 502 | WGS84 Elevation (Height Over the Ellipsoid) | m | Elevation over WGS84 reference frame |
| 503 | MSL (Height Above Mean Sea Level) – Altitude | m | Altitude over the Mean Sea Level |
| 504 | AGL (Above Ground Level) – Height | m | Height Above Ground Level – Dependent on external |
| 505 | Ground Velocity North | m/s | Ground Velocity component in the North direction (N) |
| 506 | Ground Velocity East | m/s | Ground Velocity component in the East direction (NE) |
| 507 | Ground Velocity Down | m/s | Ground Velocity component in the resultant axis from |
| 508 | Sensor IAS (Indicated Air Speed) | m/s | Pitot-static measurement speed |
| 509 | Angle of Attack – AoA | rad | Angle between reference body line and flow direction |
| 510 | Sideslip | rad | Angle between the flow direction vector and the long |
| 600-603 | Temperature 1-4 | K | Variables to be configured with external Temperature |
| 650 | Gimbal command yaw | customType | Yaw sent to the gimbal |
| 651 | Gimbal command pitch | customType | Pitch sent to the gimbal |
| 652 | Gimbal stick yaw | customType | Yaw received from the joystick controlling the gimba |
| 653 | Gimbal stick pitch | customType | Pitch received from the joystick controlling the gimba |
| 654 | Gimbal pitch correction 1 | customType | Correction calculated by the gimbal for the pitch con |
| 655 | Gimbal pitch correction 2 | customType | Correction calculated by the gimbal for the pitch con |
| 656 | Gimbal old joint 1 | customType | Auxiliar variable for Gimbal control configuration |
| 657 | Gimbal old joint 2 | customType | Auxiliar variable for Gimbal control configuration |
| 658 | Cos (gimbal yaw) | customType | Auxiliar variable for Gimbal control configuration |
| 659 | Sin (gimbal yaw) | customType | Auxiliar variable for Gimbal control configuration |
| 660 | Gimbal yaw rad | customType | Auxiliar variable for Gimbal control configuration |
| 661 | Veronte Gimbal yaw output | customType | Yaw value the gimbal is outputting |
| 662 | Veronte Gimbal pitch output | customType | Pitch value the gimbal is outputting |
| 663 | Gimbal phi(z) | customType | Auxiliar variable for Gimbal control configuration |
| 664 | Gimbal theta(y) | customType | Auxiliar variable for Gimbal control configuration |
| 665 | Gimbal psi(x) | customType | Auxiliar variable for Gimbal control configuration |
| 666 | Veronte Gimbal roll output | customType | Roll value the gimbal is outputting |
| 700-705 | RPM 1-6 | rad/s | RPM associated to pulse captured 1-6 |
| 750 | Selected controller time step | s | PID selected time step |
| 751 | Selected controller derivative filtered error | customType | PID selected derivative filtered error |
| 752 | Selected controller proportional action | customType | PID selected proportional action |
| 753 | Selected controller derivative action | customType | PID selected derivative action |
| 754 | Selected controller integral input | customType | PID selected integral input |
| 755 | Selected controller integral action | customType | PID selected integral action |
| 756 | Selected controller anti-windup input | customType | PID selected anti-windup input |
| 757 | Selected controller derivative error | customType | PID selected derivative error |
| 800-815 | PWM 1-16 | customType | Pulse Width Modulation signal |
| 900-915 | Stick Input r1 – r16 | customType | Raw stick measurement |
| 1000-1031 | Stick Input y1 – y32 | customType | Servo position commanded from stick |
| 1050-1069 | Control Output u1-20 before servo saturation | customType | Commanded control output before saturation correcti |
| 1100-1104 | Lidar 1-5 Distances | m | Variable configurable for Lidar distances |
| 1105-1109 | External range sensor 1-5 measurements | m | Variable configurable for external range sensors |
| 1200 | Route-Guidance Distance | customType | Shortest distance to desired path (perpendicular dista |
| 1201 | Radar AGL (Above Ground Level) – Height | m | Radar altimeter measure |
| 1202 | Radar Speed Down | m/s | Radar speed |
| 1203 | External Rotation for Follow Route | rad | Relative vector rotation when using Follow Route |
| 1204 | Time to Impact with Obstacles | s | Time calculated with Distance to Obstacle and travel |
| 1300-1309 | Clock 1-10 | s | Configurable timers for automations – Clock 1 corres |

Table 9 – continued from previous page

| | | | |
|-----------|-------------------------------------|------------|--------------------------------------|
| 1320-1321 | ADC 3.3V input 1-2 | V | CEX ADC 3.3 V inputs |
| 1322-1323 | ADC 5.0V input 1-2 | V | CEX ADC 5.0 V inputs |
| 1324-1325 | ADC 12.0V input 1-2 | V | CEX ADC 12.0 V inputs |
| 1326-1327 | ADC 36.0V input 1-2 | V | CEX ADC 36.0 V inputs |
| 1328-1329 | ADC vIn 1-2 | V | CEX External power supplies |
| 1330 | PCB Temperature | K | CEX PCB Temperature (from ADC input) |
| 1350-1356 | 4XV ADC1-6 converted value | V | 4xVeronte ADC 1-6 converted value |
| 1357-1359 | 4XV internal ADC7-9 converted value | V | 4xVeronte ADC 7-9 converted value |
| 1360 | 4XV Vcc for arbiter | V | 4xVeronte Arbiter power supply |
| 1361-1362 | 4XV Vcc A-B 3.3V | V | 4xVeronte Redundant power supplies |
| 1363-1365 | 4XV Vcc 1-3 | V | 4xVeronte Autopilots power supplies |
| 1366-1368 | 4XV Autopilot 0-1 Score | customType | 4xVeronte Autopilots scores |
| 1369 | 4XV Autopilot External score | decimal | 4xVeronte External autopilot score |

| | | | |
|-----------|--|------------|--------------------------------|
| 1400 | Velocity – X Body Axis | m/s | Velocity vector X component |
| 1401 | Velocity – Y Body Axis | m/s | Velocity vector Y component |
| 1402 | Velocity – Z Body Axis | m/s | Velocity vector Z component |
| 1403 | Estimated Dynamic Pressure | Pa | Dynamic pressure sensor raw |
| 1404 | Barometric Pressure at Sea Level (QNH) | Pa | Introduced value for QNH |
| 1450-1453 | Captured pulse 1-4 | customType | Input values from pulses |
| 1490 | Interneer raw x distance | m | Raw measurements for X-axis |
| 1491 | Interneer raw y distance | m | Raw measurements for Y-axis |
| 1492 | Interneer raw z distance | m | Raw measurements for Z-axis |
| 1493 | Interneer raw angle | rad | Raw measurements for interneer |
| 1494 | Interneer raw xy standard deviation | m | Raw measurements for XY a |
| 1495 | Interneer raw z standard deviation | m | Raw measurements for Z-axis |
| 1496 | Interneer raw angle standard deviation | rad | Raw measurements for interneer |
| 1497 | Interneer position update frequency | Hz | Frequency at which Interneer |
| 1500 | GNSS1 Time of Week | s | Data from GNSS1 module: T |
| 1501 | GNSS1 ECEF Position X | m | Data from GNSS1 module: I |
| 1502 | GNSS1 ECEF Position Y | m | Data from GNSS1 module: I |
| 1503 | GNSS1 ECEF Position Z | m | Data from GNSS1 module: I |
| 1504 | GNSS1 Longitude | rad | Data from GNSS1 module: I |
| 1505 | GNSS1 Latitude | rad | Data from GNSS1 module: I |
| 1506 | GNSS1 Height Above Ellipsoid (WGS84) | m | Data from GNSS1 module: I |
| 1507 | GNSS1 Mean Sea Level (MSL) | m | Data from GNSS1 module: I |
| 1508 | GNSS1 Above Ground Level (AGL) | m | Data from GNSS1 module: A |
| 1509 | GNSS1 PDOP (Dilution of Precision of Position) | customType | Data from GNSS1 module: I |
| 1510 | GNSS1 Accuracy | m | Data from GNSS1 module: A |
| 1511 | GNSS1 Horizontal Accuracy Estimate | m | Data from GNSS1 module: I |
| 1512 | GNSS1 Vertical Accuracy Estimate | m | Data from GNSS1 module: V |
| 1513 | GNSS1 Velocity North | m/s | Data from GNSS1 module: V |
| 1514 | GNSS1 Velocity East | m/s | Data from GNSS1 module: V |
| 1515 | GNSS1 Velocity Down | m/s | Data from GNSS1 module: V |
| 1516 | GNSS1 Speed Accuracy Estimate | m/s | Data from GNSS1 module: S |
| 1517 | GNSS1 Related Base Longitude | rad | Data from GNSS1 module: I |
| 1518 | GNSS1 Related Base Latitude | rad | Data from GNSS1 module: I |
| 1519 | GNSS1 Related Base WGS84 Altitude | m | Data from GNSS1 module: I |
| 1520 | GNSS1 Related Base to Rover Azimuth | rad | Data from GNSS1 module: I |

Table 10 – continued from previous page

| | | | |
|-----------|--|-----------------------------|------------------------------|
| 1521 | GNSS1 Related Base to Rover Elevation | rad | Data from GNSS1 module: I |
| 1522 | GNSS1 Related Base to Rover Distance | m | Data from GNSS1 module: I |
| 1523 | GNSS1 Related Base to Rover Accuracy | m | Data from GNSS1 module: I |
| 1524 | GNSS1 Survey in Accuracy | m | Data from GNSS1 module: I |
| 1525 | GNSS1 Related Base to Rover North | m | Data from GNSS1 module: I |
| 1526 | GNSS1 Related Base to Rover East | m | Data from GNSS1 module: I |
| 1527 | GNSS1 Related Base to Rover Down | m | Data from GNSS1 module: I |
| 1528 | GNSS1 Position Frequency | Hz | Data from GNSS1 module: I |
| 1600 | GNSS2 Time of Week | s | Data from GNSS2 module: I |
| 1601 | GNSS2 ECEF Position X | m | Data from GNSS2 module: I |
| 1602 | GNSS2 ECEF Position Y | m | Data from GNSS2 module: I |
| 1603 | GNSS2 ECEF Position Z | m | Data from GNSS2 module: I |
| 1604 | GNSS2 Longitude | rad | Data from GNSS2 module: I |
| 1605 | GNSS2 Latitude | rad | Data from GNSS2 module: I |
| 1606 | GNSS2 Height Above Ellipsoid (WGS84) | m | Data from GNSS2 module: I |
| 1607 | GNSS2 Mean Sea Level (MSL) | m | Data from GNSS2 module: I |
| 1608 | GNSS2 Above Ground Level (AGL) | m | Data from GNSS2 module: I |
| 1609 | GNSS2 PDOP (Dilution of Precision of Position) | customType | Data from GNSS2 module: I |
| 1610 | GNSS2 Accuracy | m | Data from GNSS2 module: I |
| 1611 | GNSS2 Horizontal Accuracy Estimate | m | Data from GNSS2 module: I |
| 1612 | GNSS2 Vertical Accuracy Estimate | m | Data from GNSS2 module: I |
| 1613 | GNSS2 Velocity North | m/s | Data from GNSS2 module: I |
| 1614 | GNSS2 Velocity East | m/s | Data from GNSS2 module: I |
| 1615 | GNSS2 Velocity Down | m/s | Data from GNSS2 module: I |
| 1616 | GNSS2 Speed Accuracy Estimate | m/s | Data from GNSS2 module: I |
| 1617 | GNSS2 Related Base Longitude | rad | Data from GNSS2 module: I |
| 1618 | GNSS2 Related Base Latitude | rad | Data from GNSS2 module: I |
| 1619 | GNSS2 Related Base WGS84 Altitude | m | Data from GNSS2 module: I |
| 1620 | GNSS2 Related Base to Rover Azimuth | rad | Data from GNSS2 module: I |
| 1621 | GNSS2 Related Base to Rover Elevation | rad | Data from GNSS2 module: I |
| 1622 | GNSS2 Related Base to Rover Distance | m | Data from GNSS2 module: I |
| 1623 | GNSS2 Related Base to Rover Accuracy | m | Data from GNSS2 module: I |
| 1624 | GNSS2 Survey in Accuracy | m | Data from GNSS2 module: I |
| 1625 | GNSS2 Related Base to Rover North | m | Data from GNSS2 module: I |
| 1626 | GNSS2 Related Base to Rover East | m | Data from GNSS2 module: I |
| 1627 | GNSS2 Related Base to Rover Down | m | Data from GNSS2 module: I |
| 1528 | GNSS2 Position Frequency | Hz | Data from GNSS2 module: I |
| 1700-1731 | Actuator Output s1 – s32 | customType | Configurable variable from a |
| 1800-1895 | Distance, azimuth and elevation to Object of Interest 1-32 | m, rad and m (respectively) | Spherical coordinates to Obj |

| | | | |
|------|-------------------------------------|----------|----------------------|
| 2000 | RX Packet Error Rate (on board) | decimal | Value relating RX p |
| 2001 | TX Packet Error Rate (on board) | decimal | Value relating TX p |
| 2002 | Computed RX pkt/s used for RX PER | messages | Packages per second |
| 2003 | Remote RX pkt/s used for TX PER | messages | Same as Computed |
| 2004 | Computed TX pkt/s used for TX PER | messages | Packages per second |
| 2005 | Remote TX pkt/s used for RX PER | messages | Same as Computed |
| 2019 | Stick RX rate | Hz | Stick messages recei |
| 2020 | Position fix Time | s | Time spend with GN |
| 2040 | Tunnel producer receive frequency 1 | Hz | Frequency at which |

Table 11 – continued from previous page

| | | | |
|-----------|--|------------------|-----------------------|
| 2041 | Tunnel producer receive frequency 2 | Hz | Frequency at which |
| 2042 | Tunnel producer receive frequency 3 | Hz | Frequency at which |
| 2043 | Tunnel consumer send frequency 1 | Hz | Frequency at which |
| 2044 | Tunnel consumer send frequency 2 | Hz | Frequency at which |
| 2045 | Tunnel consumer send frequency 3 | Hz | Frequency at which |
| 2046 | Max duration of step in CIO | s | Longest time duration |
| 2047 | Acquisition task timestep | s | Average period for e |
| 2048 | Acquisition task maximum timestep | s | Maximum period fo |
| 2049 | Cross core message queue CPU ratio | percentage | % of time of CPU th |
| 2050 | Acquisition task average CPU ratio | percentage | Average % of time c |
| 2051 | Acquisition task maximum CPU ratio | percentage | Maximum % of time |
| 2052 | Acquisition task average time | s | Average time acquis |
| 2053 | Acquisition task maximum time | s | Maximum time acqu |
| 2054 | CIO Max time | s | Maximum time in ac |
| 2055 | CIO average time | s | Average time in acqu |
| 2094 | GNC task average CPU ratio | percentage | Average % of time c |
| 2095 | GNC task maximum CPU ratio | percentage | Maximum % of time |
| 2096 | GNC task average time | s | Average time spent c |
| 2097 | GNC task maximum time | s | Maximum time spent |
| 2098 | GNC task maximum timestep | s | Maximum execution |
| 2099 | Max duration of step in GNC | s | Maximum duration o |
| 2100 | Gyroscope Based on Accelerometer – X Body Axis | rad/s | Gyroscope measure |
| 2101 | Gyroscope Based on Accelerometer – Y Body Axis | rad/s | Gyroscope measure |
| 2102 | Gyroscope Based on Accelerometer – Z Body Axis | rad/s | Gyroscope measure |
| 2103 | Acceleration North | m/s ² | Acceleration in the N |
| 2104 | Acceleration East | m/s ² | Acceleration in the E |
| 2105 | Acceleration Down | m/s ² | Acceleration in the D |
| 2112 | Estimated Dem | m | Altitude given by th |
| 2200 | Curve Length Covered | m | Total distance from c |
| 2201 | Curve Length | m | Total distance from c |
| 2202 | Curve Length Pending | m | Total distance from c |
| 2203 | Curve Parameter Covered | customType | Total amount from c |
| 2204 | Curve Parameter Range | customType | Total distance from c |
| 2205 | Curve Parameter Pending | customType | Total distance from c |
| 2250-2259 | Reserved 1-10 | customType | System reserved var |
| 2300-2302 | Joint 1-3 of Gimbal 1 | rad | Variables for Gimbal |
| 2303-2305 | Joint 1-3 of Gimbal 2 | rad | Variables for Gimbal |
| 2303-2305 | Joint 1-3 of Gimbal 2 | rad | Variables for Gimbal |
| 2330 | VMC control loop period | s | Veronte Motor Cont |
| 2331 | VMC control loop maximum period | s | Veronte Motor Cont |
| 2332 | VMC control loop duration | s | Veronte Motor Cont |
| 2333 | VMC control loop maximum duration | s | Veronte Motor Cont |
| 2334 | VMC control CPU usage ratio | percentage | Veronte Motor Cont |
| 2335 | VMC control loop maximum CPU usage ratio | percentage | Veronte Motor Cont |
| 2336-2338 | VMC U-W phase current | customType | Veronte Motor Cont |
| 2339 | VMC electrical angle | rad | Veronte Motor Cont |
| 2340 | VMC mechanical angle | rad | Veronte Motor Cont |
| 2341 | VMC mechanical angular speed | rad/s | Veronte Motor Cont |
| 2342 | VMC desired mechanical angle | rad | Veronte Motor Cont |
| 2343 | VMC position controller output | rad/s | Veronte Motor Cont |

Table 11 – continued from previous page

| | | | |
|-----------|---|------------|-----------------------|
| 2344 | VMC desired mechanical angular speed | rad/s | Veronte Motor Cont |
| 2345 | VMC desired mechanical angular speed after limiter | rad/s | Veronte Motor Cont |
| 2346 | VMC speed controller output | customType | Veronte Motor Cont |
| 2347-2348 | VMC clarke alpha-beta current | customType | Veronte Motor Cont |
| 2349-2350 | VMC park direct-quadrature current | customType | Veronte Motor Cont |
| 2351-2352 | VMC desired park direct-quadrature current | customType | Veronte Motor Cont |
| 2353-2354 | VMC park direct-quadrature current controller output | customType | Veronte Motor Cont |
| 2355-2356 | VMC clarke alpha-beta current from park controller output customType Veronte Motor Controll beta current from park controller output | | |
| 2357-2358 | VMC desired clarke alpha-beta current | customType | Veronte Motor Cont |
| 2359-2361 | VMC U-W phase space vector generator output | customType | Veronte Motor Cont |
| 2362-2364 | VMC U-W phase PWM output | percentage | Veronte Motor Cont |
| 2365 | VMC encoder raw angle | rad | Veronte Motor Cont |
| 2366 | VMC stepper output frequency | Hz | Veronte Motor Cont |
| 2367 | VMC mechanical angle error | rad | Veronte Motor Cont |
| 2368-2370 | VMC U-W phase BEMF | V | Veronte Motor Cont |
| 2400-2419 | Control Output u1-20 | customType | Control output 1-20 |
| 2500-2519 | Stick Input u1-u20 | customType | Intermediate values |
| 2600-2619 | Stick Input d1-d20 | customType | Intermediate values |
| 2700-2731 | Operation Guidance 1-32 | customType | Configurable values |
| 2800 | Wind Velocity North | m/s | Wind velocity vecto |
| 2801 | Wind Velocity East | m/s | Wind velocity vecto |
| 2802 | Wind Velocity Down | m/s | Wind velocity vecto |
| 2803 | Wind Velocity North Estimation Covariance | m/s | Wind velocity vecto |
| 2804 | Cross North-East Wind Velocity Estimation Covariance | m/s | Wind velocity vecto |
| 2805 | Wind Velocity Estimation Uncertainty (Element 2-0) | m/s | 2-0 element from co |
| 2806 | Wind Velocity Estimation Uncertainty (Element 0-1) | m/s | 0-1 element from co |
| 2807 | Wind Velocity Estimation Uncertainty (Element 1-1) | m/s | 1-1 element from co |
| 2808 | Wind Velocity Estimation Uncertainty (Element 2-1) | m/s | 2-1 element from co |
| 2809 | Wind Velocity Estimation Uncertainty (Element 0-2) | m/s | 0-2 element from co |
| 2810 | Wind Velocity Estimation Uncertainty (Element 1-2) | m/s | 1-2 element from co |
| 2811 | Wind Velocity Estimation Uncertainty (Element 2-2) | m/s | 2-2 element from co |
| 2812 | Wind Azimuth Angle | deg | Wind estimated azin |
| 2813 | Wind Velocity in North-East plane | m/s | Wind velocity vecto |
| 2900 | MSL Right from Actual QNH and Pressure Measurement | m | Mean Sea Level obt |
| 2901 | MSL for ISA and Pressure Measurement | m | Mean Sea Level calc |
| 2902 | Time Since Entering Current Phase | s | Time-lapse consider |
| 2903 | GNC Timestep | s | Task execution perio |
| 2904 | Total Flight Time | s | Time-lapse since the |
| 2905 | Total Flight Distance | m | Distance covered by |
| 2906 | Reception Frequency of Simulated Navigation Data | Hz | Frequency at which |
| 2907 | Reception Frequency of External Navigation Data | Hz | Frequency at which |
| 2908-2927 | Time Spent Within Phase 1-20 | s | Time-lapse spent by |
| 3000-3031 | Simulation Variable 1-32 | customType | Variables used for S |
| 3100-3399 | User Variable 1-300 | customType | Free variables for th |
| 4100 | Zero | customType | Constant value 0 |
| 4101 | Rvar disabled | customType | Disabled variable |

7.2.7.4.2 BIT

| ID | Name | Units/Values | Description |
|-------|---|--------------|---|
| 0-2 | Initialisation values | – | Bit for fail, ok and license – 0 for error, 1 for running |
| 3 | System Not Ready to Start | – | System is ready to start operating – 0 for not ready, 1 for ready |
| 4 | No Writing Telemetry | – | Telemetry is properly sending/receiving – 0 for no, 1 for yes |
| 5 | Reserved | – | Reserved variable |
| 6 | File System Error | – | System file manager is working – 0 for error, 1 for ok |
| 7 | System Error | – | System is working – 0 for error, 1 for running |
| 8 | Memory Allocation | – | Memory allocation is working – 0 for error, 1 for ok |
| 9 | PDI Error | – | PDI files are working – 0 for error, 1 for running |
| 10 | CIO Low or C2 Error | – | All threads running at specified frequencies – 0 for error, 1 for ok |
| 11 | PDI Version Not Compatible | – | PDI files compatible with current version – 0 for error, 1 for ok |
| 12 | System Power Up Bit Error | – | Power up Built in Test result – 0 for fail, 1 for ok |
| 13 | Reset and Write Disabled | – | When true, indicates that reset and non-operation |
| 14-15 | FTS-1-2 Feedback (V4.5) | – | Flight Termination System feedback – 0 for error, 1 for ok |
| 16-17 | Stack Core 1-2 Usage Fail | – | Stack CPU 1-2 usage - 0 for fail, 1 for ok |
| 20 | 4XV System | – | 4xVeronte system wide bit (Arbiter OK) - 0 for fail, 1 for ok |
| 21 | 4XV System Power up Bit | – | 4xVeronte power up Built in Test result – 0 for fail, 1 for ok |
| 22 | 4XV PDI | – | 4xVeronte PDI files are working – 0 for error, 1 for ok |
| 23 | 4XV Memory Allocation | – | 4xVeronte Memory allocation is working – 0 for error, 1 for ok |
| 24-25 | 4XV CAN-A-B Bus Off | – | 4xVeronte CAN A-B bus is working – 0 for error, 1 for ok |
| 26 | 4XV C1 Arbiter | – | 4xVeronte C1 Main Task – 0 for error, 1 for running |
| 27 | 4XV Acquisition Arbiter | – | 4xVeronte Acquisition task in real time – 0 for error, 1 for ok |
| 28 | 4XV Power A | – | 4xVeronte Power A Status - 0 for error, 1 for running |
| 29 | 4XV Not in Maintenance Mode | – | 4xVeronte Arbiter in Maintenance Mode – 0 for error, 1 for ok |
| 30-32 | 4XV Alive 0-2 | – | 4xVeronte Autopilot 0-2 Alive – 0 for false, 1 for true |
| 33 | 4XV Alive 3 External | – | 4xVeronte External Autopilot Alive – 0 for false, 1 for true |
| 34-36 | 4XV Ready 0-2 | – | 4xVeronte Autopilot 0-2 Ready – 0 for false, 1 for true |
| 37 | 4XV Ready 3 External | – | 4xVeronte External Autopilot Ready – 0 for false, 1 for true |
| 38 | 4XV Arbitrating | – | 4xVeronte Arbiter Ready – 0 for false, 1 for true |
| 39 | 4XV File Open Error | – | 4xVeronte System file manager is working – 0 for error, 1 for ok |
| 40 | 4XV PDI Version Not Compatible | – | 4xVeronte PDI files compatible with current version – 0 for error, 1 for ok |
| 41 | 4XV Stack Usage Fail | – | 4xVeronte Stack CPU usage - 0 for fail, 1 for ok |
| 42-46 | 4XV PWM1-5 GPIO Off | – | 4xVeronte GPIO/PWM 1-5 Read Value - 0 for fail, 1 for ok |
| 50 | Sensors error | – | Selected sensors are working - 0 for error, 1 for ok |
| 51 | Sensors-Main IMU | – | 0 for not enable, 1 for enable |
| 52 | Sensors-Secondary IMU | – | 0 for not enable, 1 for enable |
| 53 | Sensors-Magnetometer | – | Internal magnetometer – 0 for not enable, 1 for enable |
| 54 | Sensors-External magnetometer (HMR2300) | – | HMR2300 External magnetometer – 0 for not enable, 1 for enable |
| 55 | Sensors-External Magnetometer (LIS3MDL) | – | LIS3MDL External magnetometer – 0 for not enable, 1 for enable |
| 56 | Sensors-Static pressure (HSC) | – | HSC Static Pressure Sensor – 0 for not enable, 1 for enable |
| 57 | Sensors-Static pressure (MS56) | – | MS56 Static Pressure Sensor – 0 for not enable, 1 for enable |
| 58 | Sensors-Dynamic pressure (HSC) | – | HSC Dynamic Pressure Sensor – 0 for not enable, 1 for enable |
| 59 | Sensors-External I2C devices | – | 0 for not enable, 1 for enable |
| 60-64 | Sensors-External I2C 0-4 | – | External communication I2C – 0 for not enable, 1 for enable |
| 65-72 | SCI A-D Transmitting-Receiving | – | SCI A-D Transmitting-Receiving during last check – 0 for error, 1 for ok |
| 73-74 | CAN A-B Error | – | CAN A-B Bus communication – 0 for error, 1 for ok |
| 75-76 | CAN A-B Warning | – | CAN A-B deteriorated (but still working) - 0 for error, 1 for ok |
| 77 | Vectornav GPS fix | – | Vectornav GPS fix state - 0 for error, 1 for running |

Table 12 – continued from previous page

| ID | Name | Units/Values | Description |
|---------|---|--------------|--|
| 78 | Vectornav IMU Error | – | Vectornav IMU state - 0 for error, 1 for running |
| 79 | Vectornav Mag/Press Error | – | Vectornav Magentometer and Pressure sensors state - 0 for error, 1 for running |
| 80 | Vectornav GPS Error | – | Vectornav GPS state - 0 for error, 1 for running |
| 81 | Vectornav Navigation Error | – | Vectornav navigation state - 0 for error, 1 for running |
| 82 | Sensor External Magnetometer (HSCDTD008A) | – | External Magnetometer HSCDTD008A state - 0 for error, 1 for running |
| 83 | Sensor 3rd IMU BMI088 | – | IMU BMI088 state - 0 for error, 1 for running |
| 84 | Sensor Static Preassure 2 (DPS310) | – | Static preassure sensor DPS310 state - 0 for error, 1 for running |
| 85 | Magnetometer 4 (MMC5883MA) | – | Magnetometer MMC5883MA state - 0 for error, 1 for running |
| 100 | Position not Fixed | – | GNSS data reception – 0 for not receiving, 1 for running |
| 101 | Out of Georeferenced Area | – | 0 for being outside a Georeferenced area, 1 for being inside |
| 102-103 | CAN A-B Receiving | – | CAN A or B communication – 0 for not receiving, 1 for running |
| 104-105 | Stick PPM 1-2 detection | – | Stick PPM (1 or 2) – 0 for not detecting, 1 for detecting |
| 106 | Magnetic Field Out of Bounds | – | 0 for Magnetic Field Out of Bounds, 1 for Magnetic Field in Bounds |
| 107 | INSS Navigation Off | – | INS Navigation state - 0 for off, 1 for on |
| 108-109 | Stick PPM 3-4 detection | – | Stick PPM (3 or 4) – 0 for not detecting, 1 for detecting |
| 110 | Stick Not Detected | – | Stick signal detected – 0 for not detecting, 1 for detected |
| 111-112 | CAN A-B Transmitting | – | CAN A or B communication – 0 for not transmitting, 1 for transmitting |
| 113 | Iridium Ready | – | Iridium ready state – 0 for not ready, 1 for ready |
| 114 | EKF: Cholesky Inverse Error | – | Cholesky Inverse Error in Kalman filter - 0 for error, 1 for running |
| 115 | EKF: Condition Number Error | – | Matrix inversion error in Kalman filter - 0 for error, 1 for running |
| 116 | Radar Altimeter CAN-RX Error | – | Radar Altimeter State – 0 for error, 1 for running |
| 117-118 | Main-SuC Power Error | – | Main or SuC Power State – 0 for error, 1 for running |
| 120-123 | Pulse 1-4 Not Detected | – | Pulse 1-4 signal detected – 0 for not detecting, 1 for detected |
| 124 | 4XV Vcc for Arbiter CPU Error | – | 4xVeronte Arbiter Power State – 0 for error, 1 for running |
| 125-126 | 4XV Vcc-A-B Error | – | 4xVeronte Redundant Power State – 0 for error, 1 for running |
| 127-129 | 4XV Vcc Vcc-1-3 Error | – | 4xVeronte Autopilots Power State – 0 for error, 1 for running |
| 130 | EKF Navigation State | – | Extended Kalman Filter Navigation State – 0 for error, 1 for running |
| 150 | External VCP Navigation Error | – | External VCP State – 0 for error, 1 for running |
| 160 | External var Navigation Error | – | External Navigation State – 0 for error, 1 for running |
| 180 | External Attitude | – | Kind of attitude calculation – 0 for external, 1 for internal |
| 181 | Reserved Off | – | System Bit |
| 182 | FTS Activation (V4.5) | – | FTS Activation Bit - 0 for not activated, 1 for activated |
| 190 | Interneer ultrasound position status | – | 0 for error, 1 for running |
| 191 | Interneer ultrasound angle status | – | 0 for error, 1 for running |
| 200-206 | (D)GNSS1 | – | (Differential) GNSS1 Navigation, Input, Survey Mode |
| 207 | DGNSS1 Not Moving Baseline Mode | – | Moving Baseline Mode - 0 for deactivated, 1 for activated |
| 230-293 | 4XV Custom Msg 0-63 Error | – | 4xVeronte Arbiter custom message timeout - 0 for error, 1 for running |
| 300-306 | (D)GNSS2 | – | (Differential) GNSS2 Navigation, Input, Survey Mode |
| 307 | DGNSS2 Not Moving Baseline Mode | – | Moving Baseline Mode - 0 for deactivated, 1 for activated |
| 330 | Jetibox COMM Error | – | Jetibox is communicating properly - 0 for error, 1 for running |
| 400 | C1 Low Frequency | – | 0 for CPU main task error, 1 for CPU main task ok |
| 401 | GNC Step Missed | – | 0 for GNC Step Missed, 1 for GNC Task ok |
| 402 | Acquisition Step Missed | – | 0 for Acquisition Step Missed, 1 for Acquisition Task ok |
| 403 | CIO Hi Overload Warning | – | 0 for Acquisition Task overload, 1 for Acquisition Task ok |
| 404-405 | Reserved | – | System bits |
| 480 | VMC Stepper Direction Output | – | Veronte Motor Controller stepper inversion - 0 for normal, 1 for inverted |
| 481 | VMC Brushless Driver Fault | – | Veronte Motor Controller PWM driver input error - 0 for ok, 1 for error |
| 500 | Ground effect compensation variance disabled | – | Ground effect compensation effect – 0 for disabled, 1 for enabled |
| 501 | Ground effect compensation measurement disabled | – | Ground effect compensation effect – 0 for disabled, 1 for enabled |

Table 12 – continued from previous page

| ID | Name | Units/Values | Description |
|-----------|----------------------------------|--------------|--|
| 600 | Wind Estimation Off | – | Wind estimation - 0 for disabled, 1 for enabled |
| 700-731 | Servo 1-32 Off | – | Servos state – 0 for saturated, 1 for Ok |
| 800-815 | PWM1-16 GPIO Off | – | PWM GPIO 1-16 communication State – 0 for Off, 1 for On |
| 816-819 | EQEP A-D Off | – | Input/Output State – 0 for Off, 1 for On |
| 820-822 | RSSI LED 1-3 Off | – | Received Signal Strength Indicator led state – 0 for Off, 1 for On |
| 900-931 | Virtual GPIO 1-32 Off | – | Virtual GPIO 1-32 - 0 for Off, 1 for On |
| 1000-1009 | Simulation BIT 1-10 State | – | Variables used for Simulation data – 0 for error, 1 for ok |
| 1010-1113 | Custom Msg 0-103 Rx Error | – | Custom message timeout - 0 for error, 1 for ok |
| 1120-1121 | Entrance EKF GNSS0-1 Off | – | GNSS1-2 information considered in EKF Navigation |
| 1122 | Entrance EKF GNSS EXT Off | – | External GNSS information considered in EKF Navigation |
| 1123 | Entrance EKF Internet Off | – | Internet information considered in EKF Navigation |
| 1124 | Entrance EKF GPSCOMPASS Off | – | GNSS Compass information considered in EKF Navigation |
| 1125 | Entrance EKF Magnetometer Off | – | Magnetometer information considered in EKF Navigation |
| 1126 | Entrance EKF Static Press Off | – | Static Pressure sensor information considered in EKF Navigation |
| 1127 | Entrance EKF Altimeter Off | – | Altimeter information considered in EKF Navigation |
| 1128 | Entrance EKF Radar-Altimeter Off | – | Radar Altimeter information considered in EKF Navigation |
| 1129 | Entrance EKF DEM Off | – | DEM information considered in EKF Navigation |
| 1180-1181 | Sniffer Msg 0-1 Rx Error | – | Error in received sniffer message - 0 for error, 1 for ok |
| 1200-1499 | User BIT 1-300 | – | Free bits for the user to use – 0 for error, 1 for ok |
| 2200 | BIT Dummy | – | Bit for configurable checks – 0 for error, 1 for ok |

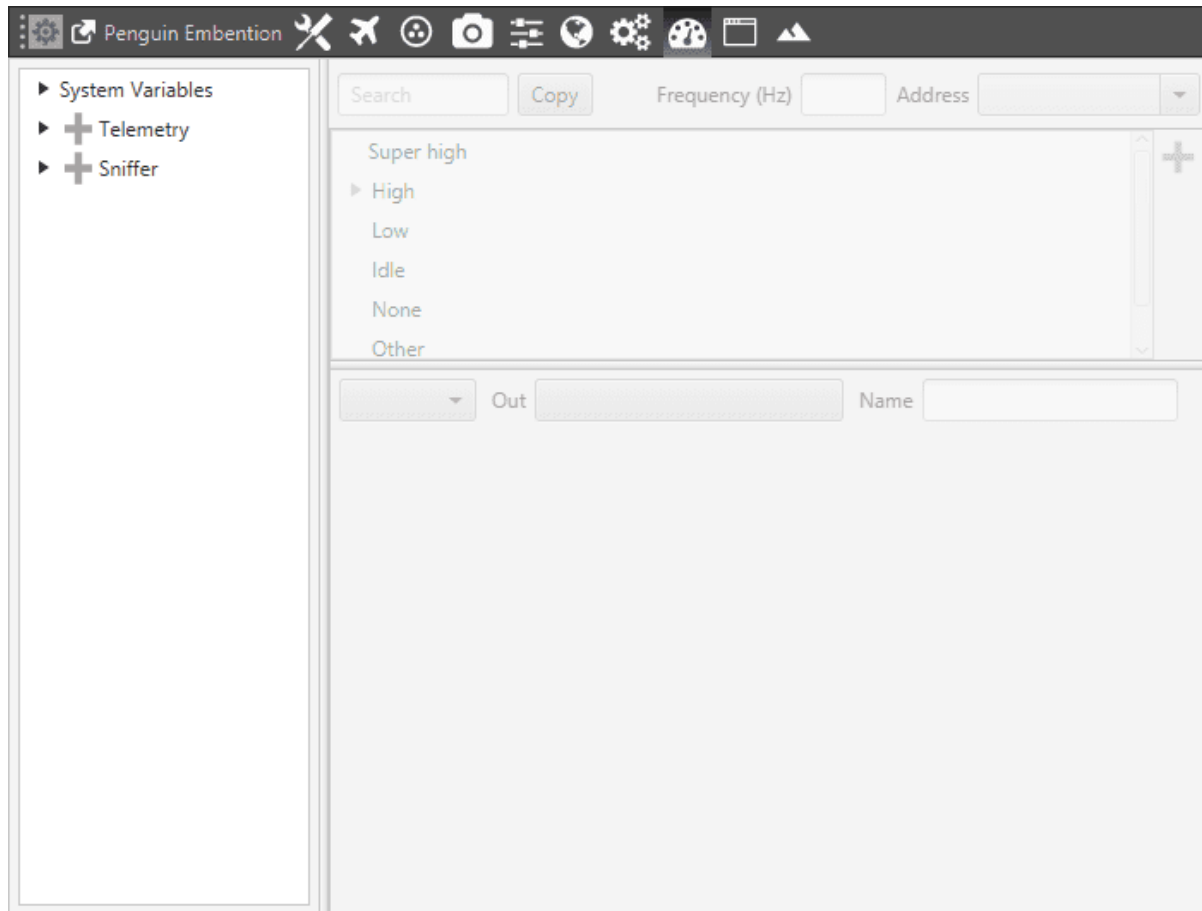
7.2.7.4.3 16 VAR

| ID | Name | Units/Values | Description |
|---------|---|--------------|---|
| 0 | Actuator mode | – | Index pointing to the flight mode in use |
| 1 | Phase identifier | – | Index pointing to the active phase |
| 2-18 | ADC Channel 1-17 | – | Internal ADC 1, 7-8 ADC 1-5 |
| 19 | Current envelope | – | Index pointing to the envelope in use |
| 20 | Counter for C2 system BIT | – | Index for number of cycles from Core 2 |
| 50 | PDI Error Source | – | Index for error source identification |
| 55 | 4XV Veronte Selected | – | 4xVeronte selected autopilot |
| 56 | 4XV Config manager status | – | 4xVeronte Configuration manager status (flash, ram) |
| 57 | 4XV File System Status | – | 4xVeronte State error for DFS2 file system |
| 58-59 | 4XV CAN to Serial 0-1 frames dropped | – | 4xVeronte Lost messages during CAN to Serial |
| 60-69 | 4XV Internal ADC Channel 1-10 | – | 4xVeronte ADC Converters input signals |
| 70 | 4XV VCC arbiter | – | 4xVeronte Arbiter power supply |
| 71-72 | 4XV VCC A-B | – | 4xVeronte Redundant power supplies |
| 73-75 | 4XV VCC 1-3 | – | 4xVeronte Autopilots power supplies |
| 80 | Detour calculation identifier | – | Index for a route change |
| 90 | Version Major | – | Major software version |
| 91 | Version Minor | – | Minor software version |
| 92 | Version Revision | – | Revision software version |
| 95 | UAV Address | – | UAV address |
| 96 | File system status | – | State error for DFS2 FS |
| 100 | GNSS1 Number of Satellites Used in Solution | – | Number of Satellites Used in Solution |
| 101-112 | GNSS1 rejected/accepted RTCM | – | Number of RTCM rejected by wrong CRC/correction |
| 113 | GNSS1 rejected RTCM unknown type | – | Number of RTCM unknown rejected by wrong |

Table 13 – continued from previous page

| ID | Name | Units/Values | Description |
|-----------|---|--------------|--|
| 114 | GNSS1 week | – | GNSS1 week |
| 150 | GNSS2 Number of Satellites Used in Solution | – | Number of Satellites Used in Solution |
| 151-162 | GNSS2 rejected/accepted RTCM | – | Number of RTCM rejected by wrong CRC/corr |
| 163 | GNSS2 rejected RTCM unknown type | – | Number of RTCM unknown rejected by wrong |
| 164 | GNSS2 week | – | GNSS2 week |
| 200 | Radar Altimeter State | – | Index for the radar altimeter state |
| 201 | Current Section | – | Index showing section |
| 202 | Last Achieved Section | – | Index showing sections achieved |
| 203 | Track Stage | – | Index showed when a route change happens |
| 204 | Current patchset ID | – | Index showing the patchset |
| 303-305 | HMR2300 Magnetometer Raw Measurement X-Y-Z | – | External HMR2300 Magnetometer Raw Measu |
| 310-311 | Iridium sent-received | – | Number of packets succesfully sent/received |
| 398 | VectorNav Mode | – | Index showing external source VectorNav mode |
| 399 | Identifier of max duration step in acquisition | – | Identifier of maximum duration step in acquisiti |
| 400 | Interneer raw status | – | Interneer raw status |
| 401 | Navigation source | – | Index pointing to the primary navigation source |
| 402 | Raw position source identifier | – | GPS identifier selected as main |
| 403-410 | Sensor selection | – | Static & Dynamic Pressure, Primary accelerom |
| 425 | Identifier of max duration step in GNC | – | Step with maximum duration |
| 426 | Group of user bits selected for CBIT | – | Step with maximum duration |
| 450-453 | CAN A-B Tx-Rx errors | – | CAN A-B communication errors in transmission |
| 454-456 | CAN to Serial 1-3 frames dropped | – | Lost messages during CAN to Serial transforma |
| 460-461 | First-Last file Periodic log | – | First-Last file of the periodic log |
| 462-463 | First-Last file Event log | – | First-Last file of the event log |
| 464-465 | First-Last file Fast log | – | First-Last file of the fast log |
| 490 | Number of moving objects detected | – | Number of moving objects detected |
| 491-492 | Veronte static cfg CRC (no Op.) of files | – | Veronte static cfg CRC (no Op.) of files - High |
| 493-494 | Veronte static cfg CRC (no Op.) of memory | – | Veronte static cfg CRC (no Op.) of memory - H |
| 495-496 | Global configuration state (crc) files-memory | – | Global configuration state (crc) files-memory - |
| 497 | Config manager status (flash/sd/maintenance m.) | – | Config manager status |
| 498-499 | Global configuration state (crc) files-memory | – | Global configuration state (crc) files-memory |
| 500 | Transponder sequence number | – | Value of the transponder sequence number |
| 501 | System Reserved | – | System variables not configurable |
| 550-555 | Reserved 1-8 | – | System reserved variables |
| 600-615 | PPM channel 0-15 output | – | CEX PPM channel outputs |
| 900-909 | Simulation variables | – | Variables used for Simulation data |
| 1000-1299 | User Variables | – | Free variables for the user to use |
| 2000 | Uvar Disabled | – | Disabled variable |
| 2001 | Zero | – | Uvar with constant 0 value |

The following figure shows the menu to manage the variables in the system.



Variables Configuration Menu

The three options available on the **Variables** menu are:

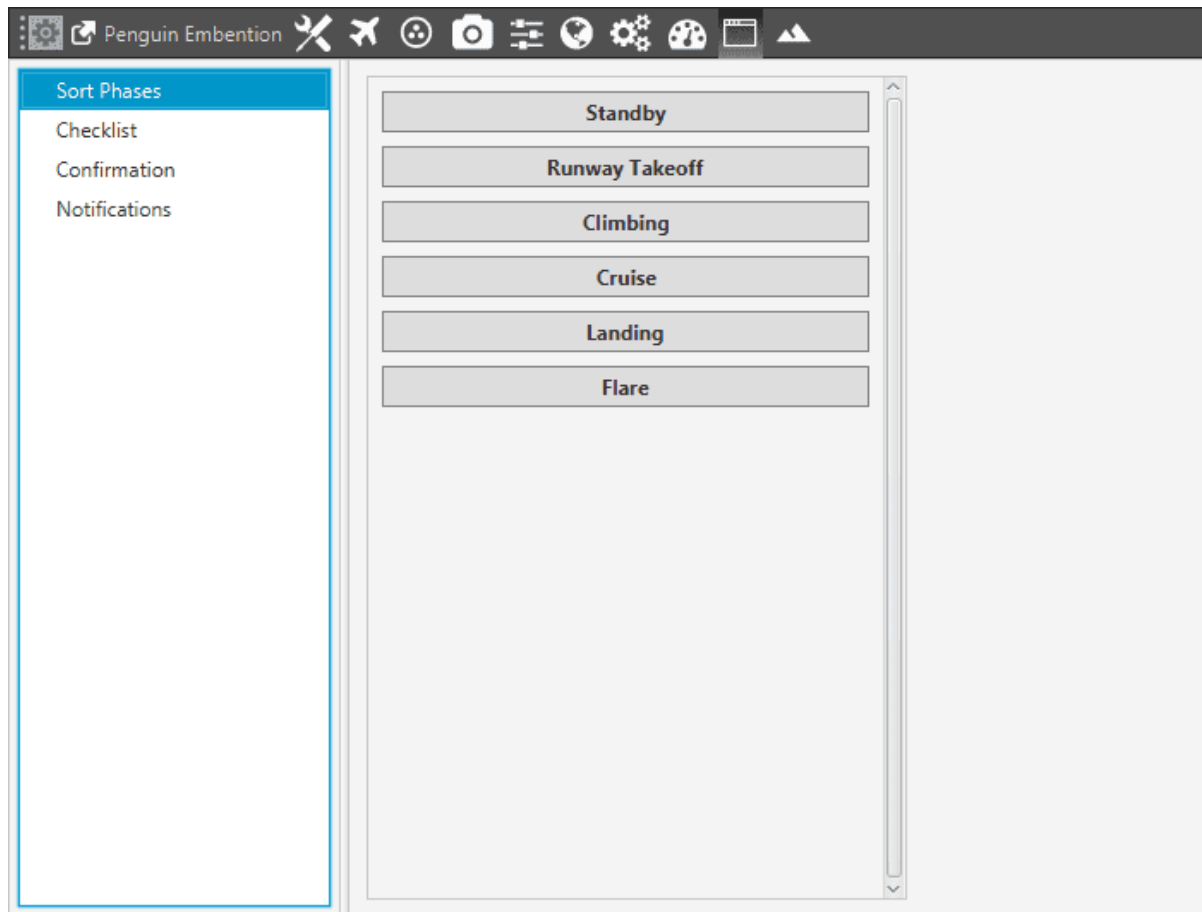
| Value | Description |
|------------------|---|
| System Variables | In this menu, the user can find the name of all system variables, as well as their units and initial values |
| Telemetry | In this menu, the user can set the priority of the system variables to be saved and sent |
| Sniffer | Option used to configure what messages user wants to receive |

7.2.8 Panel

In this menu the user can customize how the phases will look in Veronte Panel, create checklists for each phase, create manual check boxes for particular phase changes and configure some notifications that will appear behind the Veronte Unit (Side Panel) upon activation.

7.2.8.1 Sort Phases

The first menu gives the option to arrange the phases that will appear on the panel. Just dragging a phase and moving it wherever is desired will change its location in the panel. More than one phase can be arranged in the same line.



Phases Disposition

7.2.8.2 Checklists

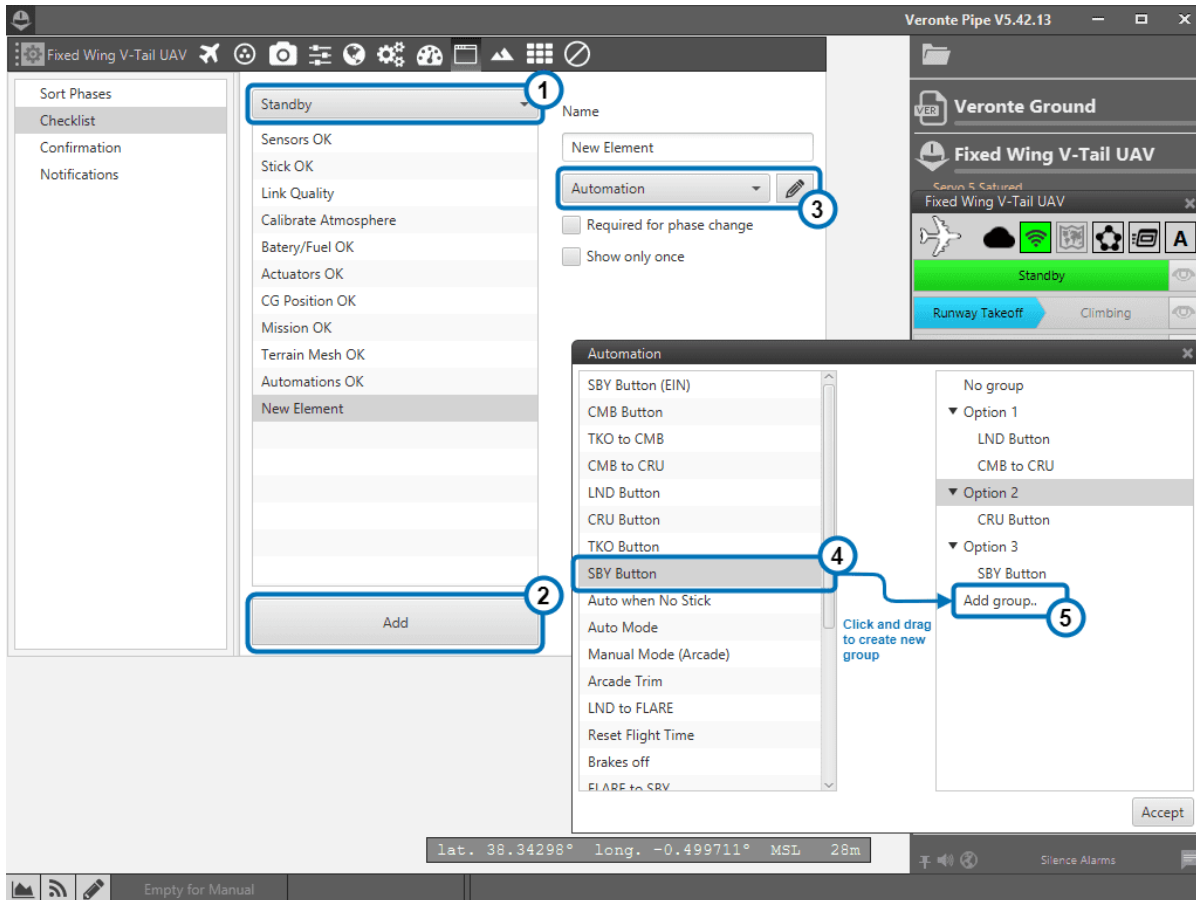
This feature is used to make sure that some requirements have been accomplished, for example, prior to a phase change or to avoid a possible malfunction.

In this menu, the checklist that will appear in the Panel is configured. In (1), the user will find all the phases configured for the operation. In each one of them, new elements for the checklist can be added with the button **Add** (2). The user can modify the checklist order of the phase by selecting and dragging elements in the list to the desired position.

The configurable parameters for each element are:

- **Name:** the name that will identify the element.
- **Type:** the element chosen from the checklist can be one of the following types.
 - **None:** any action is performed, been just a check for the user to do something external to Pipe
 - **Calibrate:** the user can request the calibration of the atmosphere model, the DEM or other internal sensors.
 - **Command:** send to the UAV a position, a yaw angle or the wind velocity in the three axes.

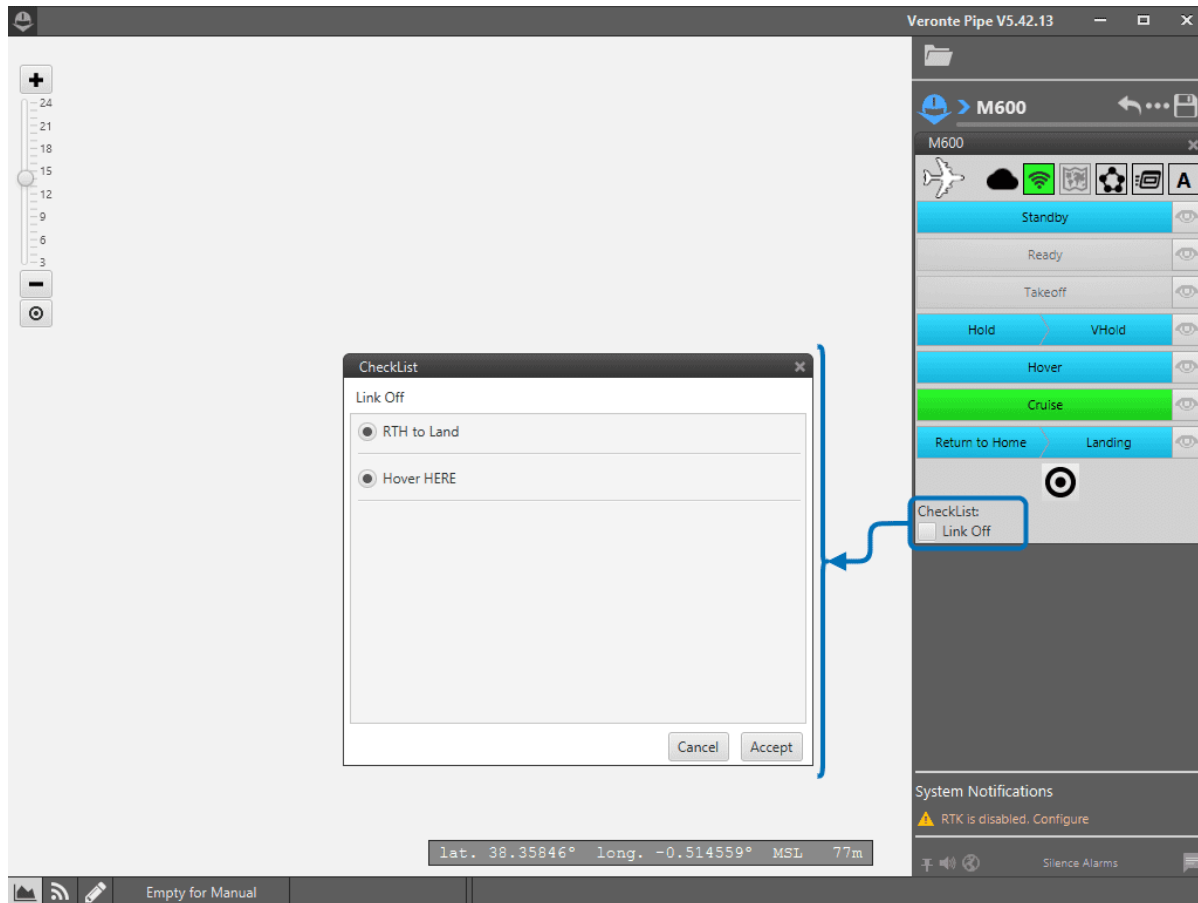
- **In Range Check:** Allows checking if a variable is between the range selected.
- **Atmosphere:** set the atmosphere parameters (temperature...).
- **Automation:** selection between a set of automations, in order to only activate one of them for the phase that the aircraft is entering in.



Checklist Elements

Click and drag the automations into “Add group” to create a new selection. It is possible to have more than one option.

For example, the following figure shows a window with an automation selection on the checklist. The user has to select between two automations previously included in a group. Once one of them is selected, the other one will be automatically deactivated:



Checklist Verification

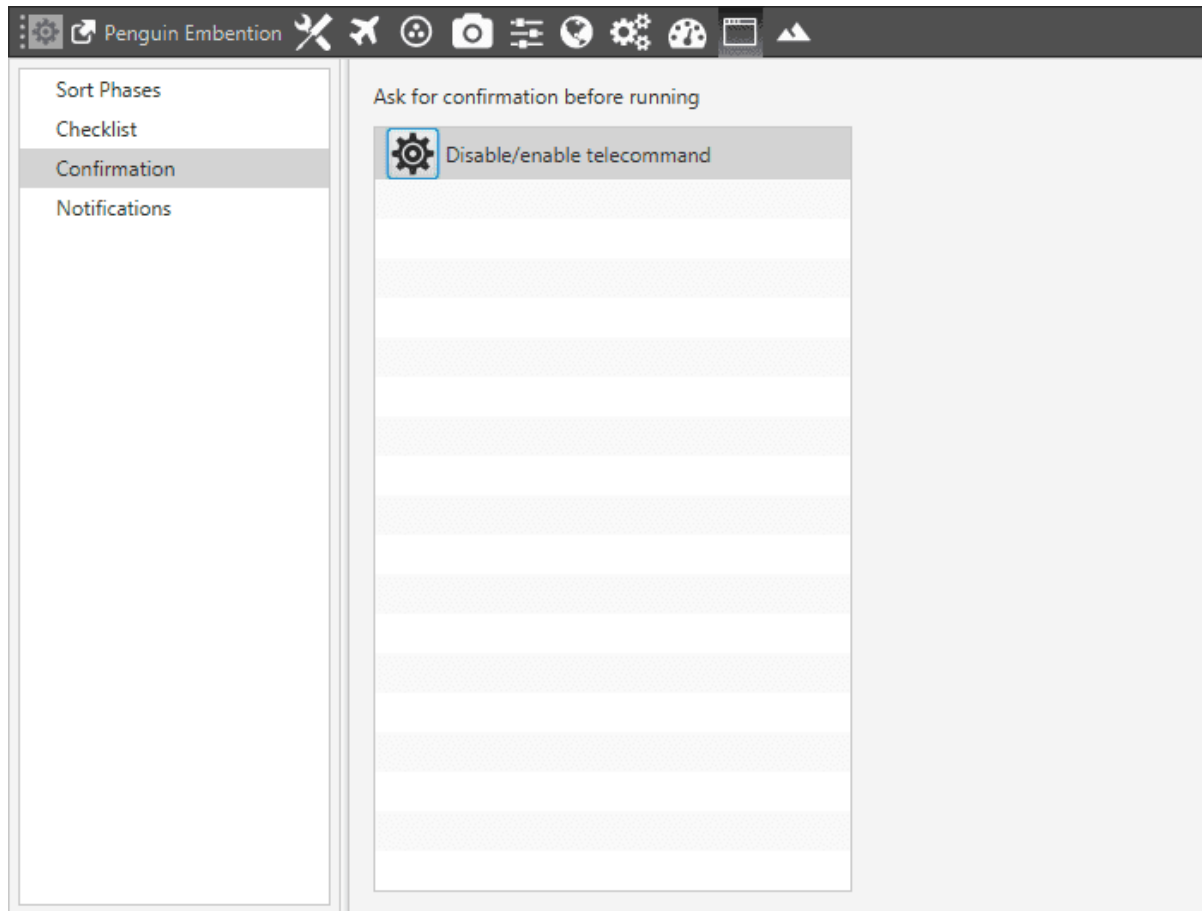
- **Required for phase change:** the element must be checked to go to another phase.
- **Show only once:** the check will only appear the first time its phase is executed.

The screenshot shows the 'Checklist Configuration' window. On the left is a sidebar with a menu containing 'Sort Phases', 'Checklist' (which is highlighted), 'Confirmation', and 'Notifications'. The main area is divided into three sections. The top section has a dropdown menu currently set to 'Standby'. Below this is a list of checklist items: 'Mode AUTO and check control in SBY', 'Check mission (First and last waypoint)', 'Check Veronte battery V(>15V)', and 'Arm catapult' (which is highlighted). There are several empty rows below the last item. At the bottom of this list is an 'Add' button. The right section is titled 'Name' and contains a text input field with 'Arm catapult', a dropdown menu set to 'None' with an edit icon, and two checkboxes: 'Required for phase change' and 'Show only once', both of which are currently unchecked.

Checklist Configuration

7.2.8.3 Confirmation

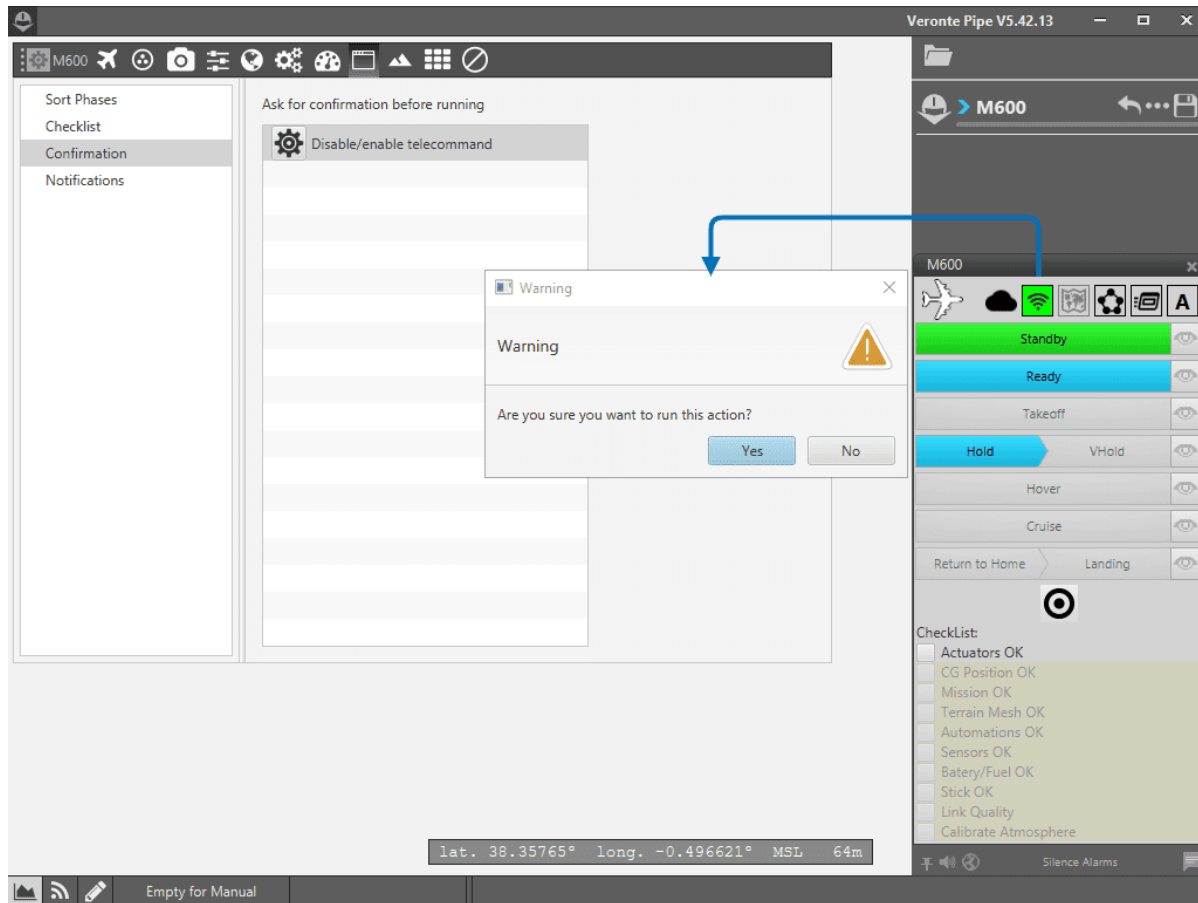
This menu manages the “Disable/Enable telecommand” confirmation message that appear when the radio link button is pressed in Veronte Panel. It is a safety system to avoid pressing this button by mistake.



Panel Confirmation

For the other custom buttons added by the user in the Veronte panel, the confirmation option is also available. When configuring the Button event, the system provides a check to include a confirmation when selecting that option. For more information check [Automations](#).

When the user is trying to activate an action (previously selected), Veronte Pipe will show a warning message like the following, asking us confirmation:

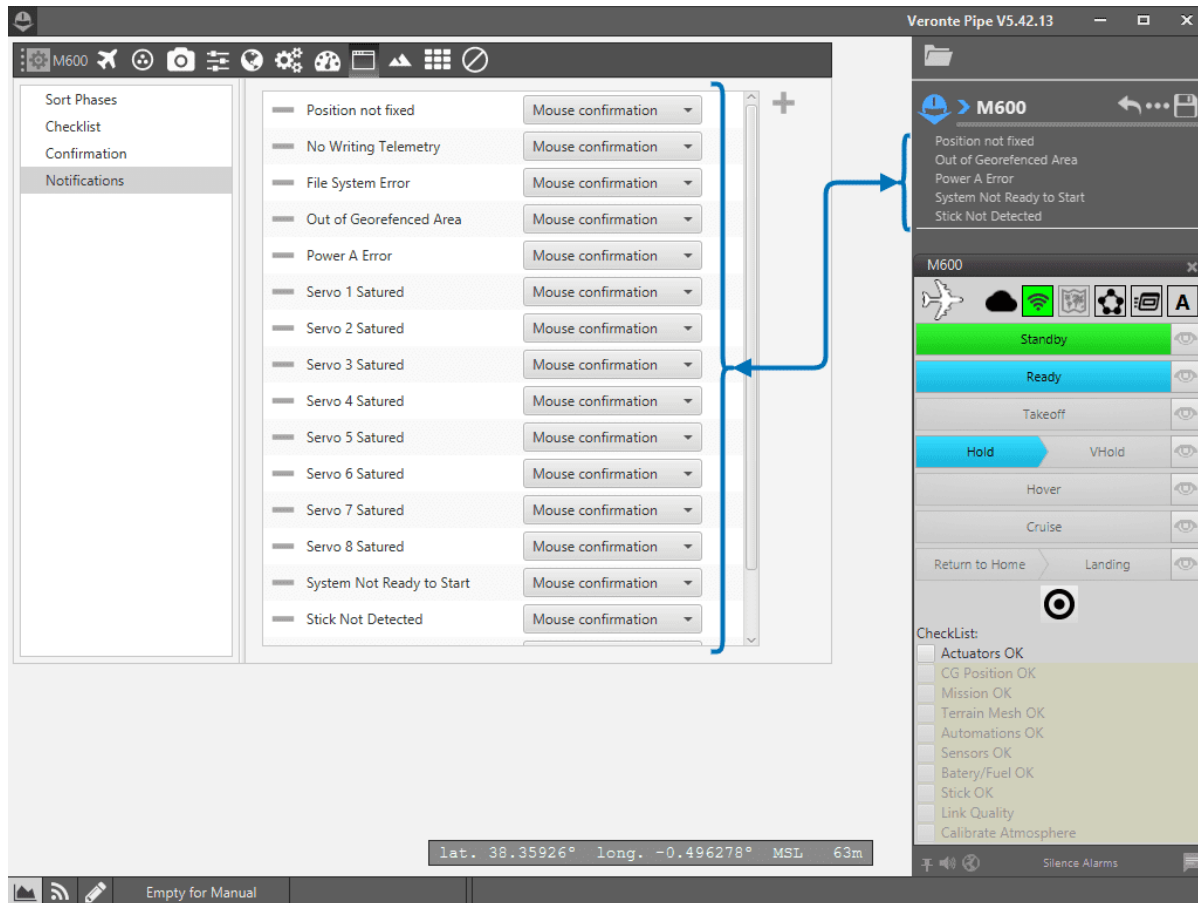


Confirmation – Example

7.2.8.4 Notifications

This menu is used to select which notifications will appear in the Side Panel and the way in which the user wants to hide them. There are three options:

- **Mouse confirmation** – It is necessary to pass with the mouse on notifications in order to hide them.
- **Auto-hide** – Notification will auto hide in few seconds.
- **No show** – The notifications will not show up.



Panel – Notifications

Besides the notifications that appear by default on this menu, it is possible to include some other system alarms or even custom ones created by the user. Click on “+” to display a pop-up window that allows the selection of a new variable. The window will allow the user to select any BIT variable of the system, including all the custom User BITs. See [variables](#) and [list of variables](#) for more information.

It is also possible to delete the notifications shown by default, click on the “-” button at the left side of any notification to remove it from the error list.

In the following table, it will be described the error types of the notification panel.

| Notification | Cause | Possible solution |
|---------------------------|---|--|
| Position not fixed | The system is not connected to GPS signal | Check the GPS signal in the area and the GPS antenna connection |
| No writing telemetry | System is not writing telemetry in the onboard log | Check the Air micro SD. It could be corrupted or without memory available |
| File system error | System has one or more corrupted files | Try to repeat the operation or check the micro SD files |
| Out of georeferenced area | The GPS position is out of the areas with terrain altitude information | Go to Terrain Profile in Mission Panel and move manually Coarse and Fine areas or choose Auto to make the system doing it automatically and save |
| Power A error | Comicro (pin 48) power supply out of range | Go to “Electrical” section of the manual to check admitted voltages |
| Servo X saturated | The servo number X is actually saturated | Check the servos panel to set the moving range |
| System not ready to start | No GPS connection/Licence expired | See “GPS navigation down”/Update the licence expiration |
| Task X real time error | System is not able to complete all tasks: 1 means High level, 5 means Low level | Check system variables in Variables Panel and move the operations to a lower level |
| Stick Not Detected | The stick is not connected to autopilot | Check radio signal and stick connection (in case of HIL simulation it will be simulated) |
| Sensors Error | Sensor malfunction | |
| PDI Error | Some PDI files are inconsistent | Review errors or import files again |

Warning: In some cases, it will be necessary to restart the autopilot once the solution of one of this problems has been applied (“Out of georeferenced area” for example).

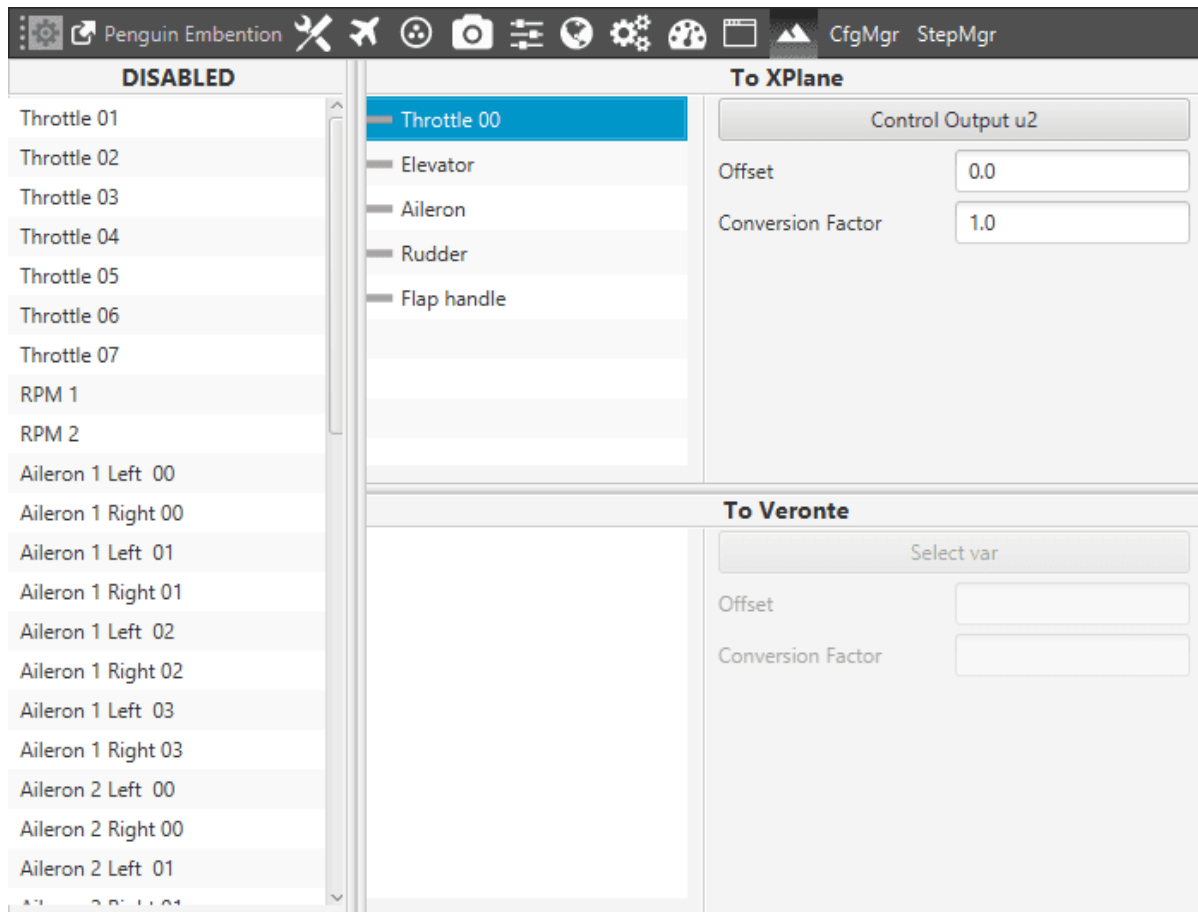
7.2.9 HIL

Professional Hardware In the Loop (HIL) Simulator package is a powerful tool for Veronte Autopilot integration, development and operator training; permitting to extensively operate the system in a safe environment, prior to conducting real flight operations

There are 2 configuration items on Veronte Pipe, one relating communications between the application and the X-Plane simulator and another one refereeing the autopilot configuration. Users can find the complete process including X-Plane settings in [Professional HIL](#).

7.2.9.1 Autopilot Configuration

HIL simulation tab is available within Veronte Autopilot setup toolbar. The user can link the variables on Veronte Autopilot with the corresponding ones in X-Plane simulator.



Veronte Pipe – HIL Setup

In this panel, X-Plane variables are available on the left side (**Disables**). In addition, it can be seen two section more **To XPlane** and **To Veronte**.

In order to configure the simulation variables, users have to:

1. Enable the ones that have been configured in the aircraft model (**Plane Marker**). Just drag and drop them into **To Xplane** section.

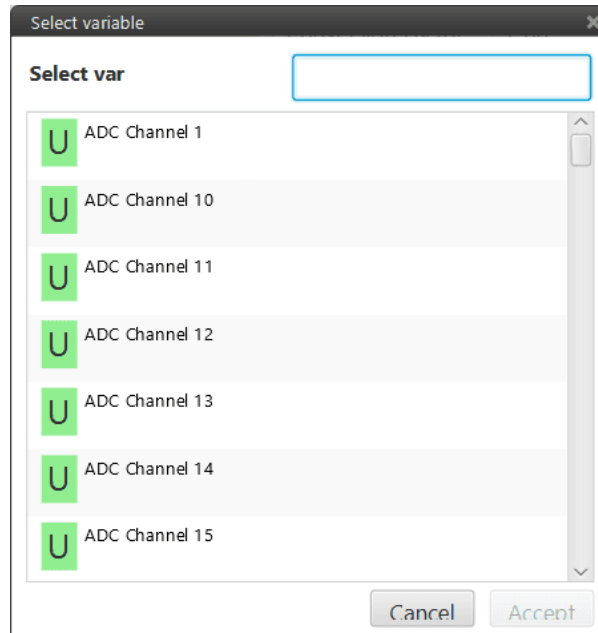
Warning: Always make sure that surfaces are moving in the right direction and with the correct deflection angle.

To avoid mistakes is possible to set a positive fixed deflection (Control Tab) in Standby phase for all surfaces and control surface deflections in the X-Plane model.

Surface control variables are of two types:

- Radians measure (variables with numbers)
- Degrees measure (variables with no numbers)

- Once X-Plane variables have been enabled, select the actuator variable (Control Output) that matches with the ones in Veronte autopilot. A new window will be displayed for each variable.



HIL Panel

- Set a **Conversion Factor** or **Offset**, if it is necessary. Conversion factor multiplies the Veronte output signal and can be used in case units on Veronte and the X-Plane simulator do not match (Surfaces in X-Plane move normally in a [-1,1] range). The following operation allows to converting an angle [deg] measure to the X-Plane form:

$$(\text{Angle} + \text{Offset}) \text{Conv.Factor} = \dots$$

When Angle and Offset are measured in [rad] and the Conversion factor is a constant (normally it can be calculated as $1/(\text{deflection angle in [rad]})$).

In the case of [rad] measures, the Conversion factor must be set in 57,29578 ([deg]-[rad] conv. factor).

Finally, it is necessary to configure the communication between Veronte Pipe and X-Plane, see section below.

7.2.9.2 Adding New variable

When there is any variable that doesn't appear in Veronte Pipe, it is possible to add a new one and linked it with the one correspondent in X-Plane. To do this, user have to edit the file "XplaneDATAGroupConfig.xml".

This file contains the following information:

```
<DATAGroup name="controls">
  <Entries>
    <DATA name="elv" desc="" convertTo="" unit="" xplaneName="" readOnly="false" parameter="0" message="11" nameUI="Elevator"/>
    <DATA name="ail" desc="" convertTo="" unit="" xplaneName="" readOnly="false" parameter="1" message="11" nameUI="Aileron"/>
    <DATA name="rud" desc="" convertTo="" unit="" xplaneName="" readOnly="false" parameter="2" message="11" nameUI="Rudder"/>
    <DATA name="srvAil1L0" desc="" convertTo="" unit="" xplaneName="" readOnly="false" parameter="0" message="70"
      nameUI="Aileron 1 Left 00"/>
    <DATA name="srvAil1R0" desc="" convertTo="" unit="" xplaneName="" readOnly="false" parameter="1" message="70"
      nameUI="Aileron 1 Right 00"/>
    <DATA name="srvAil1L1" desc="" convertTo="" unit="" xplaneName="" readOnly="false" parameter="2" message="70"
```

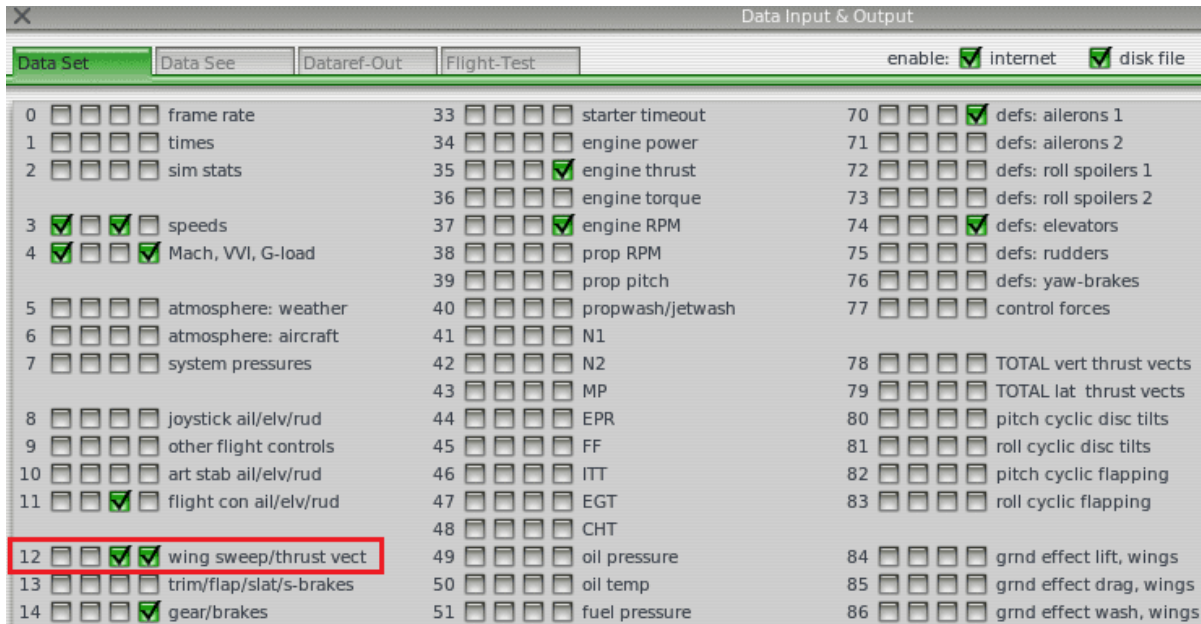
XplaneDATAGroupConfig.xml

- Data name:** variable's name in X-Plane.
- Parameter and message:** these values must be set correctly to link the variables.

- **nameUI:** variable's name in Veronte Pipe. User can type the name that will appear in the list.

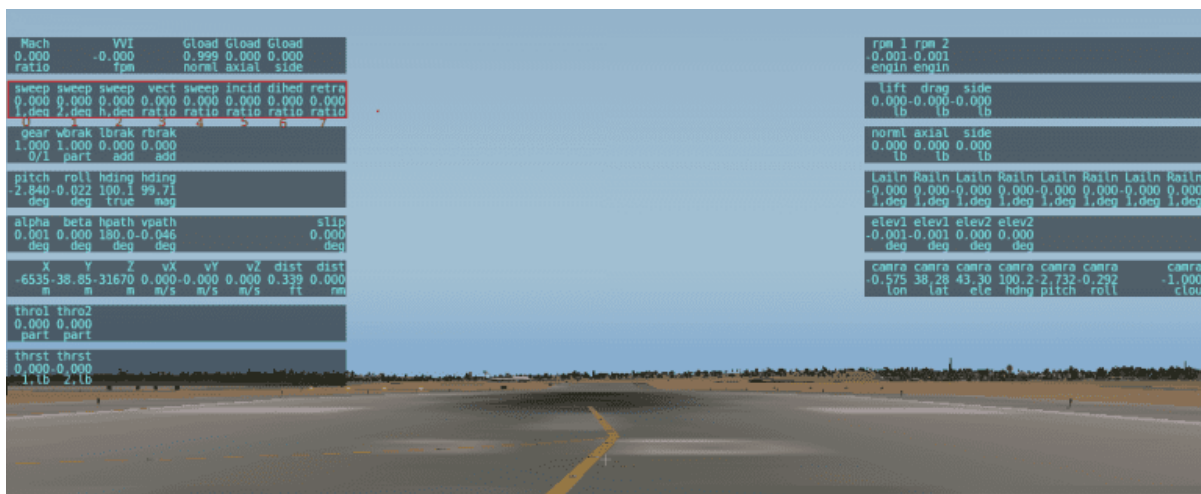
Consider the following example that explains how to add a new variable. Imagine you have a wing with variable incidence (configurable in Plane Maker) and this parameter doesn't appear in Veronte Pipe.

First, user have to check the variable's name, parameter and message in X-Plane. Open X-Plane and go to Setting/Data Input & Output.



Data Input & Output

The message is the number that appears next to each variable. In this case, the data related to a wing with variable incidence is “wing sweep/thrust vect” (see X-Plane manual), so message has to be set with 12. Variable's name and parameter can be found in the following image:



X-Plane

Variables's name is **incid** and parameter is **5**. To find out the parameter's number, user have to count starting from “0” and the left. Once all data is known, it is possible to edit the .xml file and add the new variable with a text editor. The result is shown in the following image:


```

- <DATAGroup name="controls">
  - <Entries>
    <DATA name="elv" desc="" convertTo="" unit="" xplaneName="" readOnly="false" parameter="0" message="11" nameUI="Elevator"/>
    <DATA name="ail" desc="" convertTo="" unit="" xplaneName="" readOnly="false" parameter="1" message="11" nameUI="Aileron"/>
    <DATA name="rud" desc="" convertTo="" unit="" xplaneName="" readOnly="false" parameter="2" message="11" nameUI="Rudder"/>
    <DATA name="incld" desc="" convertTo="" unit="" xplaneName="" readOnly="false" parameter="5" message="12" nameUI="Canard"/>
  
```

XplaneDATAGroupConfig.xml

The nameUI has been set as **Canard** and this will be displayed in the variables list. Now, user can simulate this variable in X-Plane from Veronte Pipe.

7.2.9.3 Veronte Pipe communications

In order to start the simulation, select the HIL option (1) on the Veronte Unit, this is available on the side menu . Popup screen will be displayed (2) for selecting the type of simulation and configuring the parameters.



Veronte Pipe – HIL Communications

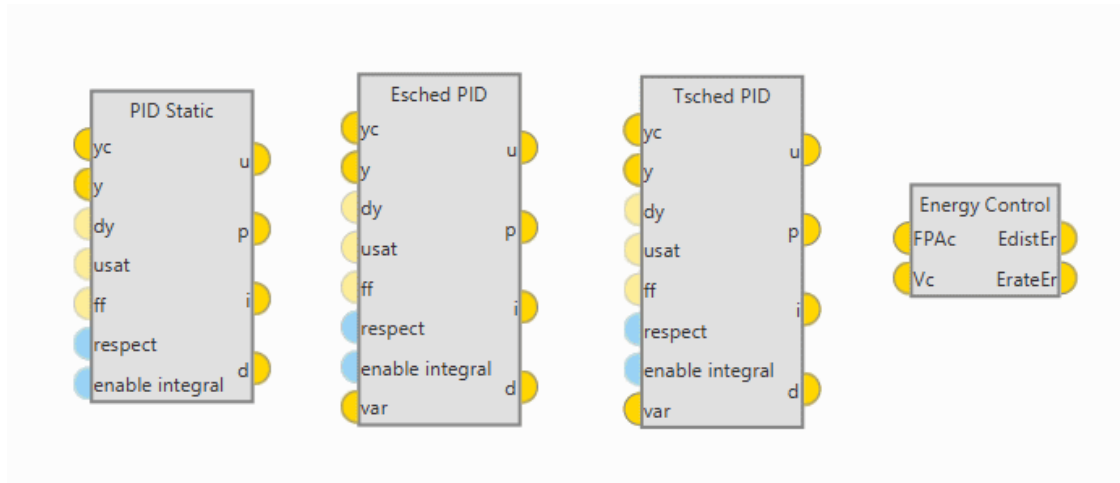
Changing the default values is only recommended for advanced users. Press start in order to start the data transfer between X-Plane and Veronte Autopilot.

Warning: The simulation must be started when the aircraft is in the Initial phase (the one that gets in once is powered). In this phase the X-Plane will simulate the GPS signal to locate the autopilot in the place indicated by the airport on X-Plane.

7.2.10 Programs

7.2.10.1 Control Blocks

Control blocks are those related to the creation of control loops:



Control Blocks

PID blocks

PID blocks allow the user to build a PID controller.

The PID mathematical implementation in **Veronte** is the following:

$$C = Kp + \frac{1}{Ti} IF(z) + \frac{Td}{\tau + DF(z)}$$

PID Model

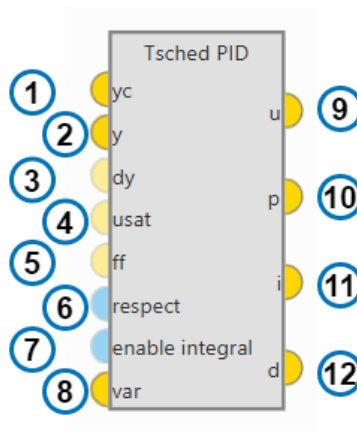
Where:

$$DF(z) = T_s \frac{z}{z-1}$$

$$IF(z) = \frac{T_s}{2} \frac{z+1}{z-1}$$

$$LPF(s) = \frac{1}{\tau s + 1}$$

Inputs and outputs

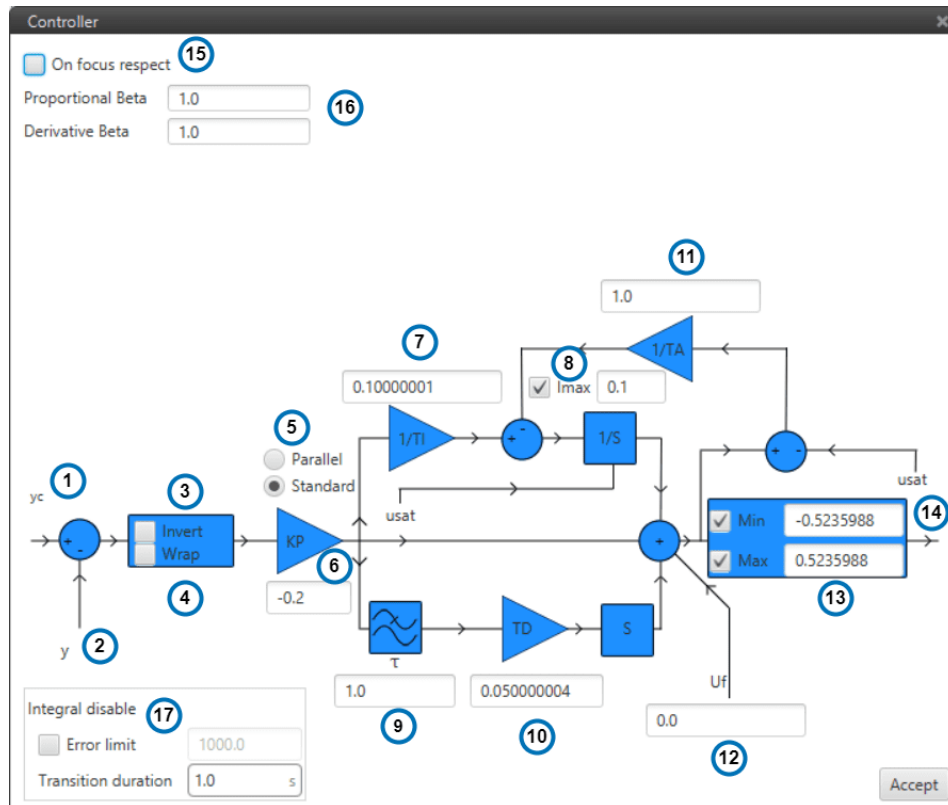


PID Block

1. **yc**: Target value, desired state that the controlled variable must acquire.
2. **y**: Closed loop feedback, current value for the controlled variable.
3. **dy (optional)**: Time derivative for feedback. If provided, this value will replace discrete derivative of y in the derivative term calculation.
4. **usat (optional)**: initial output value. Needed for the 'respect' feature.
5. **ff (optional)**: feedforward value. Offset applied at PID output.
6. **respect (optional)**: enable/disable the respect feature. Default is configured in the PID menu.
7. **enable integral (optional)**: enable/disable intergal term. Default is enabled.
8. **var (only scheduling blocks)**: scaling variable for gain scheduling.
9. **u**: pid output
10. **p**: proportional term
11. **i**: integral term
12. **d**: derivative term

Configuration

Double click on a PID block to open its configuration menu



PID Configuration

1. **yc**: Input variable.
2. **y**: Feedback variable.
3. **invert**: Apply a -1 gain .
4. **wrap**: Perform a $[-\pi, \pi]$ wrap.
5. **parallel/standard**: In *parallel* mode, PID gains are independent. In *standard* mode, I & D gains are scaled by P gain.
6. **KP**: Proportional gain.
7. **1/TI**: Integral gain.
8. **Imax**: Maximum value for integral term. Value must be positive and the limit applied is symmetrical $([-Imax, Imax])$.
9. **tau**: Time constant for the derivative term first order LPF.
10. **TD**: Derivative gain.
11. **TA**: Anti-windup gain. Recommended value around $\times 10$ KI. Unloads integral term if output is saturated.
12. **Uf**: Output offset. **Feedforward** value is also applied at this point.
13. **Min/Max**: Output limits.
14. **u**: PID output.
15. **On focus respect**: If respect is enabled, when the PID is first executed, an initial I value will be applied so that '**u**' = '**usat**' for the first iteration.

16. **Proportional/Derivative Beta:** y_c scaling for proportional and derivative terms. Unless necessary, value should always be **1**.
17. **Integral disable:** disables integral term if $(y_c - y) > \text{Error limit}$.

Tip: Remember to always use **'wrap'** for direction controllers, such as 'Heading' or 'Yaw' PIDs. This will allow the UAV to always turn in the right direction.

Danger: Applying changes to ANY **Program** will **RESET** all PIDs **Integral** terms. Make sure that this is not dangerous to your operation or make a proper use of the **'Respect'** feature if you are planning to make **in-flight** program changes.

Scheduling

Scheduling allows to adapt PID gains and parameters depending on flight conditions:

Proportional Scheduling

ESched block allows to scale **KP** gain using an external variable. This control block works in Standard mode, so integral and derivative gains are changed in the same proportion.

| | Result (Kp) |
|--------------|--------------------------|
| Inverse | $Kp_i \frac{V}{V_i}$ |
| Proportional | $Kp_i \frac{V_i}{V}$ |
| Quadratic | $Kp_i \frac{V_i^2}{V^2}$ |

PID Proportional Scheduler

Min and **Max** values make reference to the **Scaling Variable**.

If the variable is out of bounds the value of Kp for the closest limit will be applied.

Table Scheduling

TSched block allows to scale most PID parameters using an external variable.

| | KP | KD | τ | KI | TA | Uf | Imax | BetaP | BetaD |
|-----|--------|--------|--------|--------|--------|--------|---------|--------|--------|
| 0.0 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | -1.0000 | 1.0000 | 1.0000 |
| 2.0 | 1.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | -1.0000 | 1.0000 | 1.0000 |
| 3.0 | 3.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | -1.0000 | 1.0000 | 1.0000 |
| 4.0 | 7.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | -1.0000 | 1.0000 | 1.0000 |
| 5.0 | 3.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | -1.0000 | 1.0000 | 1.0000 |
| 6.0 | 4.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | -1.0000 | 1.0000 | 1.0000 |

PID Table Scheduler

If the variable is out of bounds the values for the closest point will be applied.

Values between points are **linearly interpolated**.

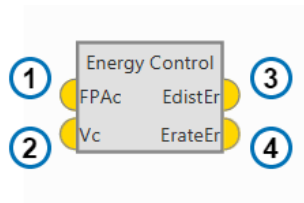
Total Energy Control

Total Energy Control is a strategy for the control of **Fixed Wing** aircrafts.

The aim of this strategy is to decouple **speed** and **altitude** controls.

The **Total Energy Control** block will provide two errors that must be minimized in order to obtain the desired speed and flight path:

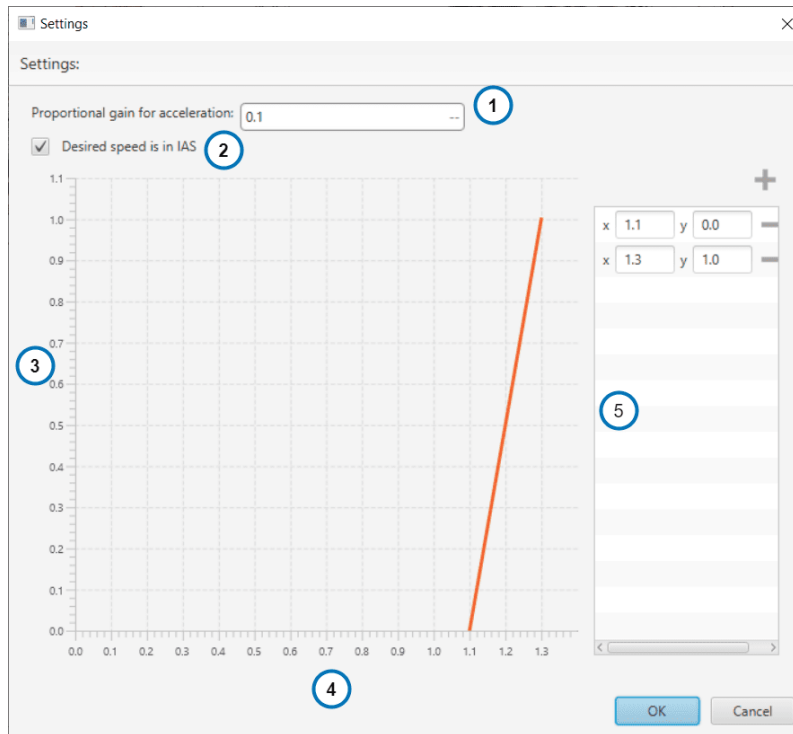
- **Energy Distribution Error:** Distribution of system energy between kinetical and geopotential energy. This error should be minimized using the **Elevator**.
- **Energy Rate Error:** Rate of change of the Total System Energy. This error should be minimized using **Throttle**.



Energy Control Block

1. **FPAc:** Desired Flight Path Angle.
2. **Vc:** Desired Speed.
3. **EdistEr:** Error in the required Energy Distribution.
4. **ErateEr:** Error in the required Energy rate.

Some parameters of the Energy algorithm can be modified by *Double clicking* on the block:

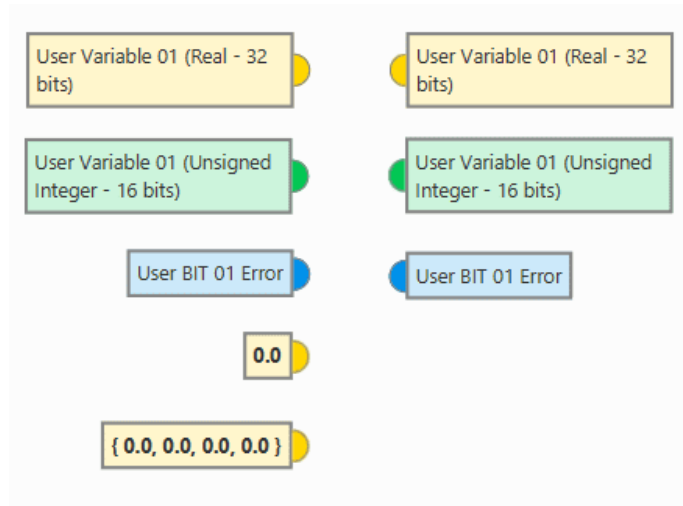


Energy Control Menu

1. **Proportional Gain for Acceleration:** in short, its an indication of how aggressive the algorithm is when trying to gain speed. The higher the value, the faster the algorithm will try to 'dive' in order to gain speed. A typical recommended value is around 0.1. Higher values are only recommended for fast maneuvering platforms.
2. **IAS/GS:**If checked, the block will use variable '**0 - IAS (Indicated Airspeed)**'. If unchecked, variable '**2 - GS (Ground speed)**' will be used as a reference instead. Use of Ground Speed is not recommended unless Airspeed measurement is not available.
3. **Stall correction coefficient:** If 1, energy control is balanced for altitude and speed. If 0 only speed control is taken into account.
4. **Speed/Stall ratio:** Ratio between current speed and minimum speed.
5. **Stall correction interpolation function:** Define how the relation function between the stall correction coefficient and the Speed/Stall ratio.

Note: The Stall correction coefficient is a **Safety** tool that can be used to sacrifice altitude control in order to improve speed control when speed gets close to the minimum speed selected in the *Envelope*

7.2.10.2 Data Source/Sink Blocks



Source/Sink Blocks

Source Blocks

Source blocks allow to import into the program any variable available in the system.

Additionally the **Const Real/Vector** allows to create a constant variable or vector.

Sink Blocks

Sink blocks allow to overwrite any variable in the system.

Variables that can be written using **Sink blocks** are:

- User Variables
- **Desired** variables (Variables whose name starts with 'Desired')

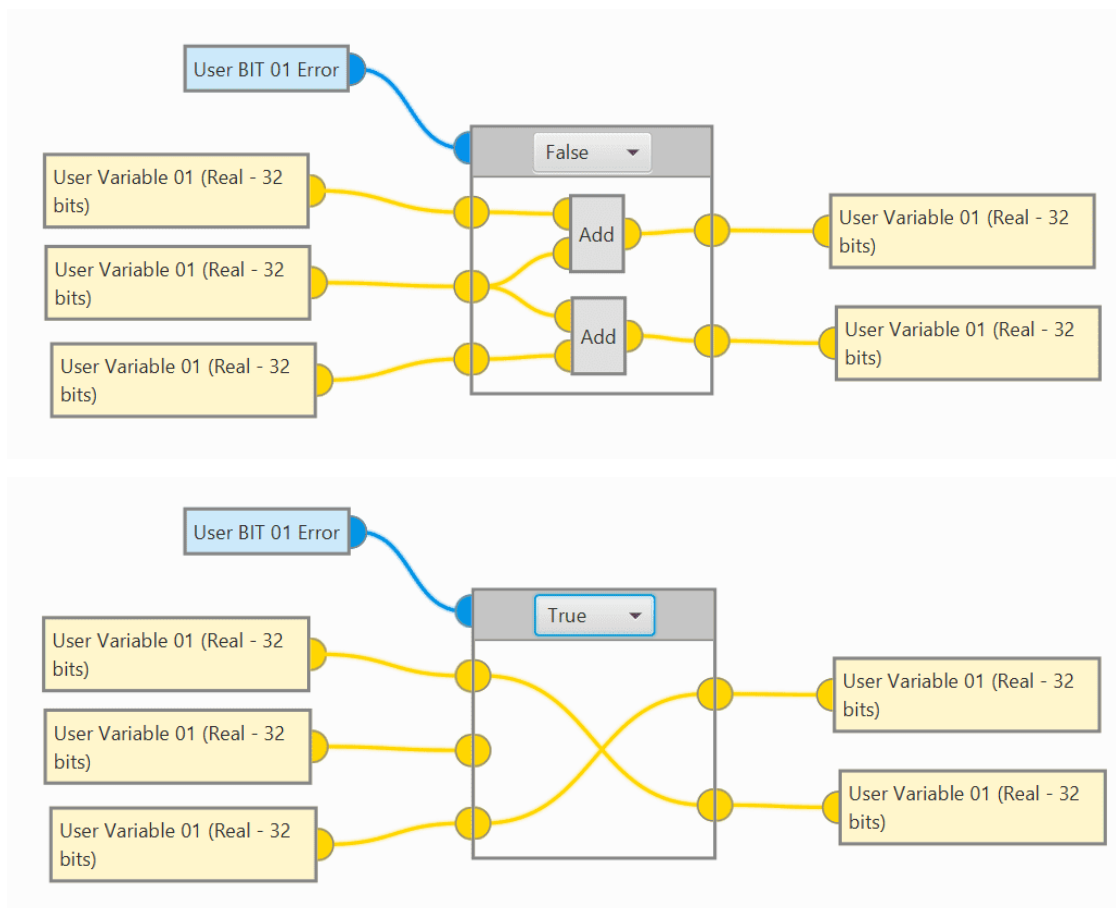
Note: **Desired** variables are naturally written by Veronte *Guidances*. If an active **Guidance** is writing a certain **Desired** variable, writing it with a **Sink** block should be avoided.

Warning: Using Sink blocks to overwrite **System Variables** usually results in the change not taking effect, but in some cases could end up causing Veronte to malfunction.

Avoid using **Sink** blocks to write any variable that does not belong to one of the groups listed above.

7.2.10.3 Execution Flow Blocks

Execution Flow blocks allow to switch sections of a program during its execution among a set of pre-configured options.



Execution Flow Blocks

Right click inside a Switch block in order to create a new block.

Drag into a **Switch block** to create **Inputs** and **Outputs**. To remove them, *right click* and select **Remove Input/Output**

Note: The size of a switch block depends on the blocks it contains. A switch block will always have the size of the **biggest** of its existing cases.

Switch Blocks

If/Else

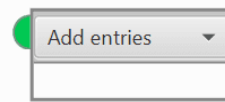
Choose between two cases based on the state of a boolean variable.



If/Else Switch

Integer case

Choose a case based on the value of an integer variable.



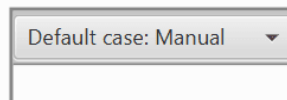
Integer Switch

Right click the **Case Block** to access Case configuration options

- **Add Case:** Create a new empty case.
- **Copy Case:** Create a copy of the current case.
- **Delete Case:** Delete the current case.
- **Add Entry:** Add a new **entry** to the current case. An **entry** is a condition under which the case will be selected. The same **entry** can on only be in one case at a time. Adding an **entry** that already exists will move said entry to the current case.
- **Delete Entry:** Remove an **entry** from the current case.
- **Set as Default case:** the Default case will be executed whenever the switch condition does not match any of the existing **entries**.

Phase case

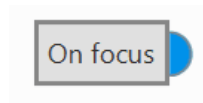
Same as **Integer case**, but using Flight Phases as the switch condition.



Phase Switch

Warning: **Case Blocks** will report a '**PDI ERROR**' if they don't have at least 2 cases with entries.

On Focus



On Focus Block

The **On Focus** block outputs a boolean value, which is only **True** the first time the block is executed.

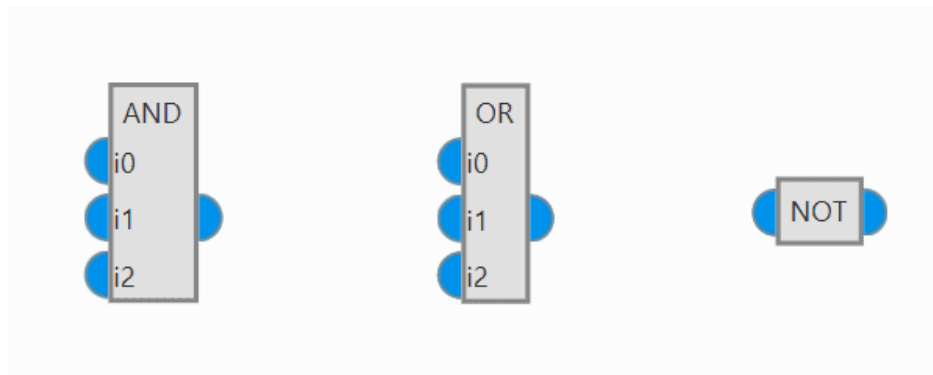
If used inside a **Switch Block**, the value will be **True** each time the case is selected.

On Focus can be used to trigger actions or initialize variables whenever a case is switched.

The following example would initialize **User Variable 01** to **7** whenever **Landing** phase is selected:

On Focus Example

7.2.10.4 Logic Blocks



Logic Blocks

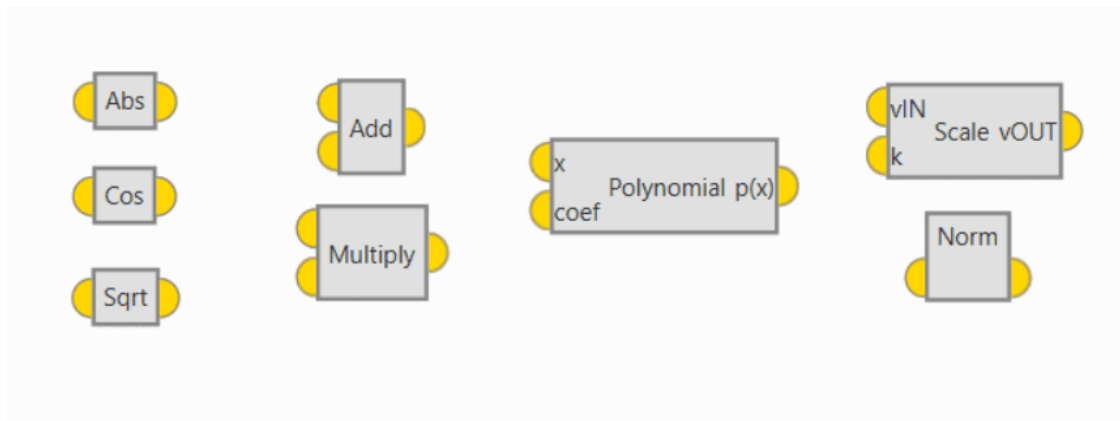
Logic gates for operating with boolean variables:

- **AND Gate**
- **OR Gate**
- **NOT Gate**

AND and **OR** number of inputs is configurable.

7.2.10.5 Math Blocks

Math blocks allow to perform a wide variety of mathematical operations.



Math Blocks

1R → 1R

1-input 1-output blocks

- **Abs**: returns absolute value of the input.
- **Sin**: returns the sine of an angle (in *rad*).
- **Cos**: returns the cosine of an angle (in *rad*).
- **Tan**: returns the tangent of an angle (in *rad*).
- **Arcsin**: inverse function of the sine. It returns an angle (in *rad*).
- **Arccos**: inverse function of the cosine. It returns an angle (in *rad*).
- **Arctan**: inverse function of the tangent. It returns an angle (in *rad*).
- **Sqrt**: returns the square root of the input.
- **Exp**: returns e powered to the input.
- **Log**: returns the natural logarithm of the input.
- **Ceil**: returns the input, rounded up.
- **Floor**: returns the input, rounded down.
- **[pi,-pi]Unwrap**: return the unwrap of the input. That is, if a jump higher than π is detected, an offset of $(-)2\pi$ is added to the result.
- **[pi,-pi]Wrap**: returns an angle (in *rad*) wrapped between π and $-\pi$.
- **[0,2pi]Wrap**: returns an angle (in *rad*) wrapped between 0 and 2π .

2R → 1R

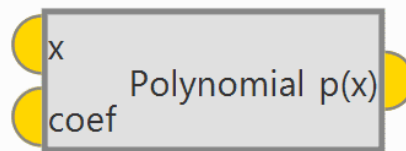
2-input 1-output blocks

- **Add**: returns the sum of the inputs.
- **Subtract**: returns the subtraction of the inputs.
- **Multiply**: returns the product of the inputs.

- **Divide:** returns the division of the inputs.
- **Power:** returns *input 1* to the power of *input 2*.
- **Max:** returns highest of the inputs.
- **Min:** returns the lowest of the inputs.
- **Remainder:** returns the remainder of the division of the inputs.
- **Atan2:** variation of the *atan* function, that allows to avoid .

Polynomial

Returns the value of the polynomial defined by the coefficients *coef* for the value of *x*.

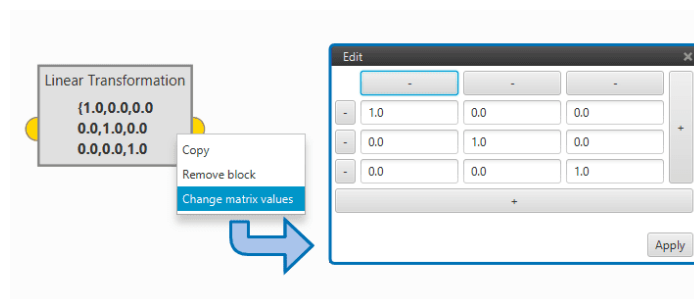


Polynomial Block

Vectors

Perform operations with vectors

- **Add:** return the sum of the input vectors.
- **Add elements:** returns the sum of input vector components.
- **Bundle:** returns a vector whose components are the inputs to the block.
- **Dot product:** returns the dot product of the input vectors.
- **Linear transformation:** returns the input vector multiplied by the transformation matrix.
In order to edit the transformation matrix, double click on the block.

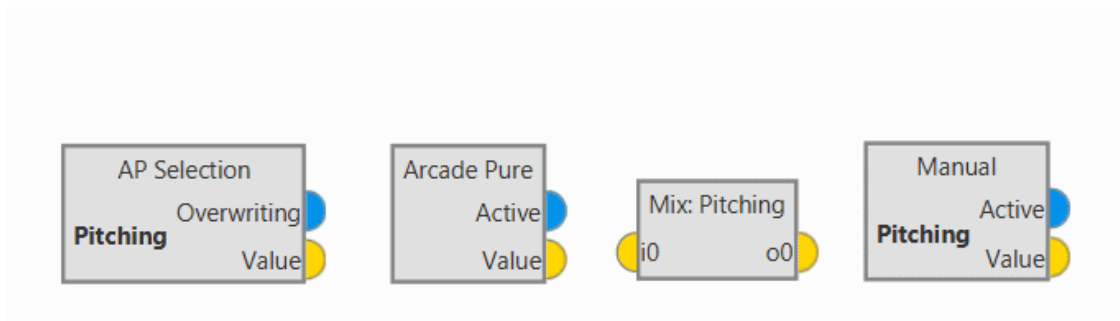


Linear Transformation Block

- **Multiply elements:** returns the product of the vector input components.

- **Norm**: returns the norm of the input vector.
- **Max**: returns the value and position of the highest component of the input vector.
- **Min**: returns the value and position of the lowest component of the input vector.
- **Scale**: returns vector v/N multiplied by scalar k .
- **Split**: returns each of the components of the input vector. The number of outputs must match the input vector length.
- **azeld \rightarrow xyz / xyz \rightarrow azeld**: transform from spherical coordinates (*azeld*) to cartesian coordinates (*xyz*) and viceversa. Input must be 3×1 .
- **Body \rightarrow NED / NED \rightarrow body**: rotate from body axis NED axis and viceversa. Input must be 3×1 .

7.2.10.6 Mode/AP Selection Blocks



Mode Blocks

AP selection

The **AP selection** block is meant to be used along **4xVeronte**, but can also have other applications.

An **AP Selection** block must be linked to one of the **Control loops**.

Whenever the current autopilot is selected, **Overwriting** will be **False**, and **Value** will be equal to the value of the control loop.

If the current autopilot is **not selected**, then **Overwriting** will be **True** and **Value** will be equal to the control loop from the **selected autopilot**.

Arcade

The arcade block allows the user to detect if the selected channel is in 'arcade' mode. This block includes the following output and configurable parameters:

- **Outputs:**

Active: BIT status that is HIGH when the mode is arcade and the stick is sending a command.

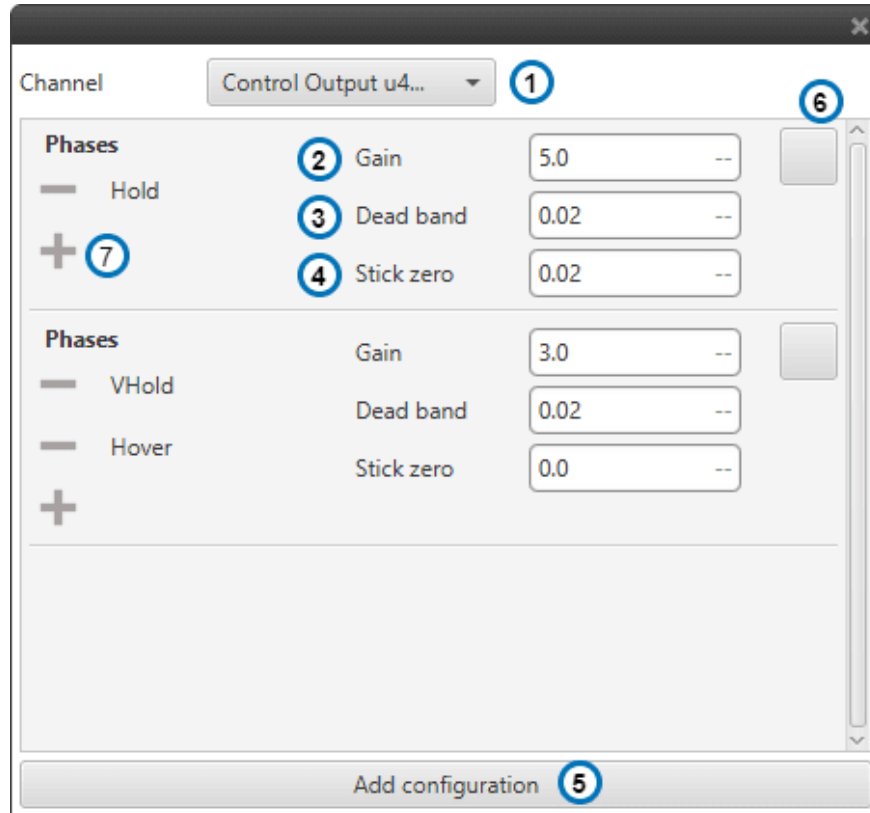
Value: will output the Arcade value for the selected channel. It is obtained as a result of applying the gain to the stick input *d*. For more information about the different stick inputs see [Stick](#)

- **Parameters:**

1. Channel: control output controlled by this block. It selects the stick input *d* used to calculate the output.
2. Gain: the output value is the result of multiplying the stick input *d* by this gain.

3. Dead band: creates a zone where the movement of the stick is not sent to the system.
4. Stick zero: output value when the value os the stick input d is 0.
5. Add configuration: add a new group of phases affected by this block.
6. Delete a group of phases.
7. Add a phase to a group.

All these parameters are shown in the image below:



Parameters Arcade block

There are three similar Arcade blocks:

Pure

Normal arcade mode. Its parameters are those explained previously.

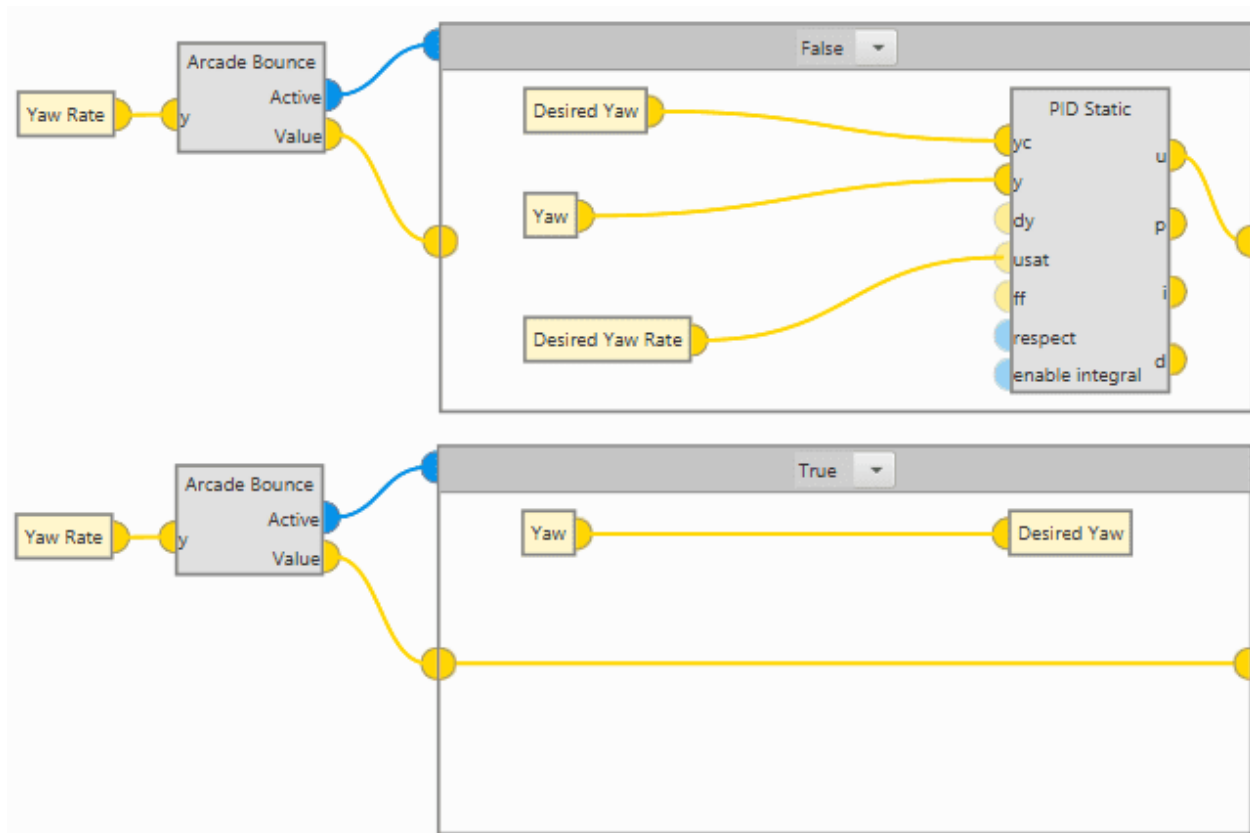
Bounce

Arcade will be also activate after applying a command, when the value of the **input** (y) changes its sign (value closest to 0) in order to modify the desired value to avoid a bounce in this variable. Normally, the value to enter is the derivate of the output variable.

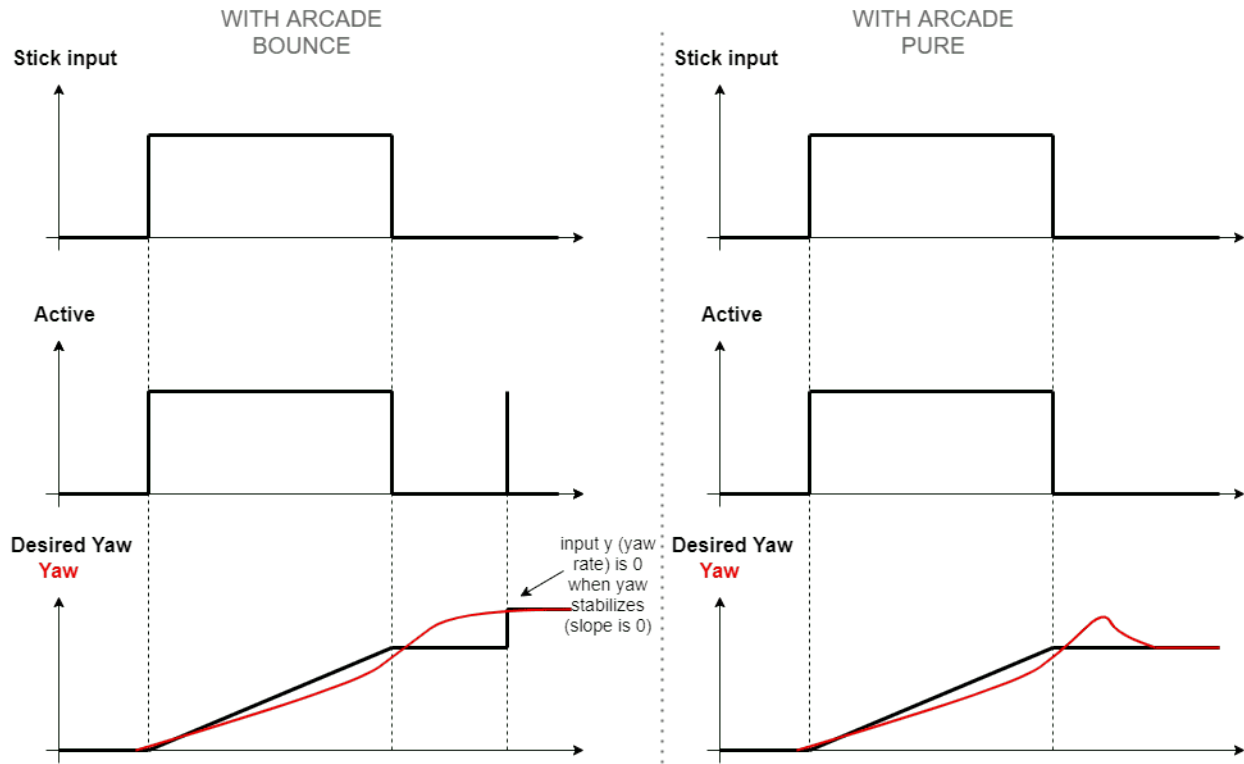
In the picture below there is an example controlling the yaw: when the stick in its zero position, the command sent is 0, the status of Active BIT is FALSE (low level), and the desired yaw is the last yaw saved when the status was TRUE (high level). However, the platform can still have a yaw rate and in a normal arcade block it could experiment this bounce. Therefore, when the yaw rate is close to 0 (changes its sign), this BIT is TRUE again (in spite of not being commanding a new yaw rate with the stick), and the desired yaw is updating with the current yaw.

Extended

Similar to Bounce. In this case Arcade will stay activated if **input** (y) is higher than the defined threshold.



Arcade bounce block example I



Arcade bounce block example II

Mix

Mix allows to apply a variable offset over the input using one of the stick channels.

Manual

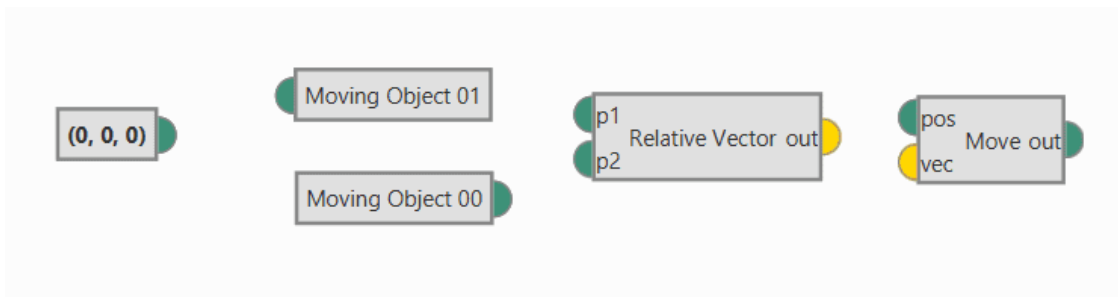
The manual block allows the user to detect if the selected channel is in 'RC' mode.

Value will output the Stick value for the selected channel.

7.2.10.7 Position Blocks

Position blocks allow to operate with position variables.

Note: In **Veronte** position variables are also referred to as **Features**.



Position Blocks

Const Position

Create a constant position.



Constant Position

Read/Write Feature

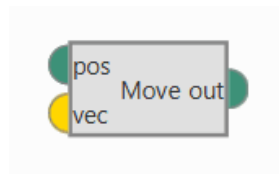
Import or export an existing **feature**.



position Blocks

Move

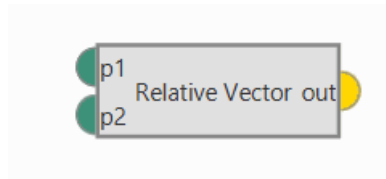
Displace a position *pos* based on an input vector *vec*. Input vector must be a 3x1 vector in meters and NED axis.



position Blocks

Relative vector

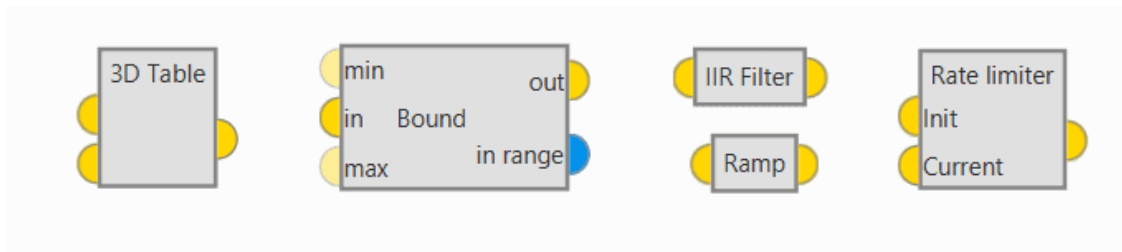
Calculate the relative vector from *p1* to *p2*.



position Blocks

7.2.10.8 Signal Blocks

Signal Blocks include functions for processing and filter signals, control inputs and outputs, etc.



Signal Blocks

3D Table Interpolation

Returns the value obtained by interpolating the configured table with the input variables. First input represents *rows*, second input represents *columns*.

If **out of range**, the value for the closest limit will be taken.

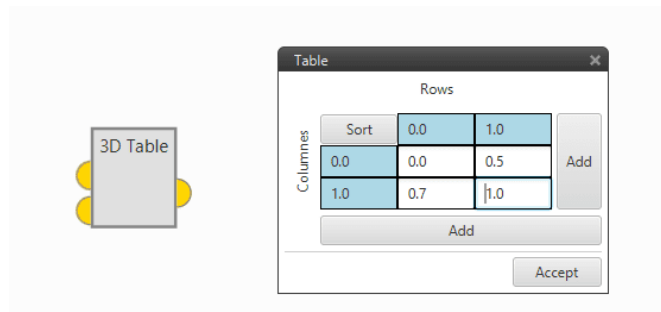


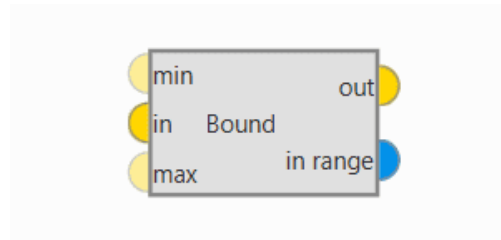
Table Interpolation

Note: **IIR Block** will return an error if the *row* or *column* values are not sorted from lowest to highest. In order to solve this, just click on the **Sort** button.

Bound

Returns the input *in*. If the input is out of range, the **limit** is returned instead. If limits are not defined, they are assumed to be **infinity**.

In range will be **true** if the input is within the limits and **false** if it is not.

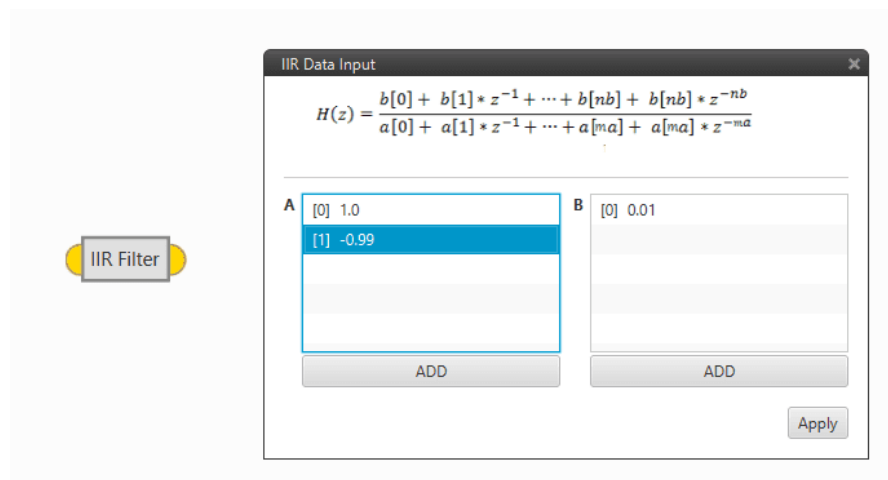


Bound Block

Tip: Use the **Bound** block to monitor that critical parameters inside your system are within operational limits (i.e. voltage levels). If **not OK**, **in range** can be used to activate an alarm.

IIR Filter

Allows the user to define an IIR Filter



IIR Filter Block

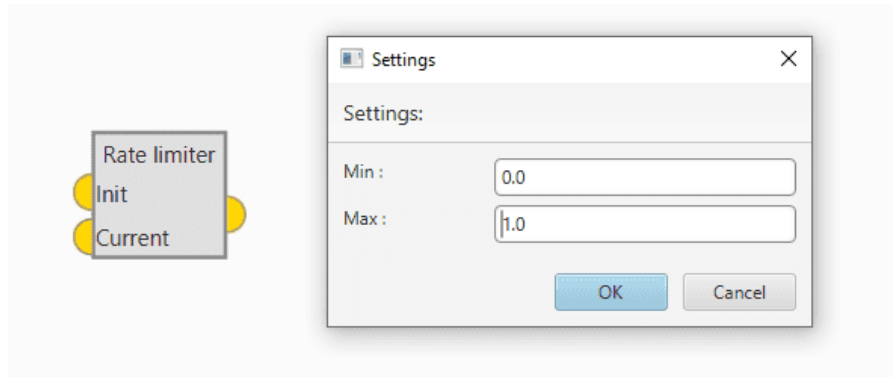
Rate Limiter

Returns the *Current* input signal, but limiting its maximum rate of change.

The limits are defined in **Input units/s**.

If the rate of change of the input is higher than the maximum, the output will try to converge to the input, but respecting the imposed maximum rate of change.

The first time the block is executed the output will be equal to *Init*.

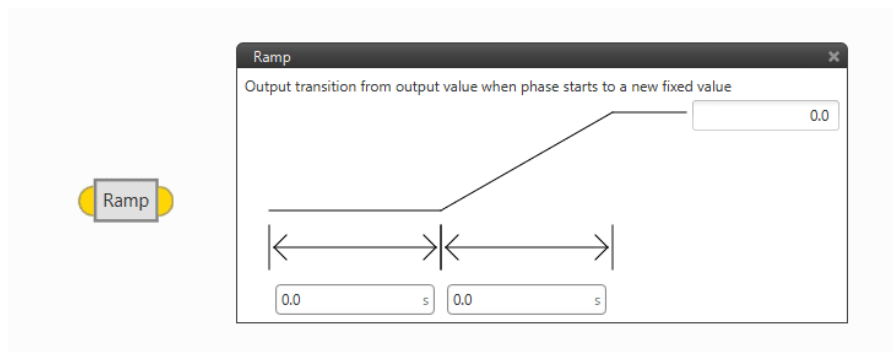


Rate limiter Block

Tip: The **Rate limiter** block can be used to avoid steps and instant spikes in the control signals, effectively reducing control noise and smoothing the flight.

Ramp

The Ramp block will ramp up to the defined value, starting from the *Input*, and respecting the **Delay** and **Ramp Time** parameters.

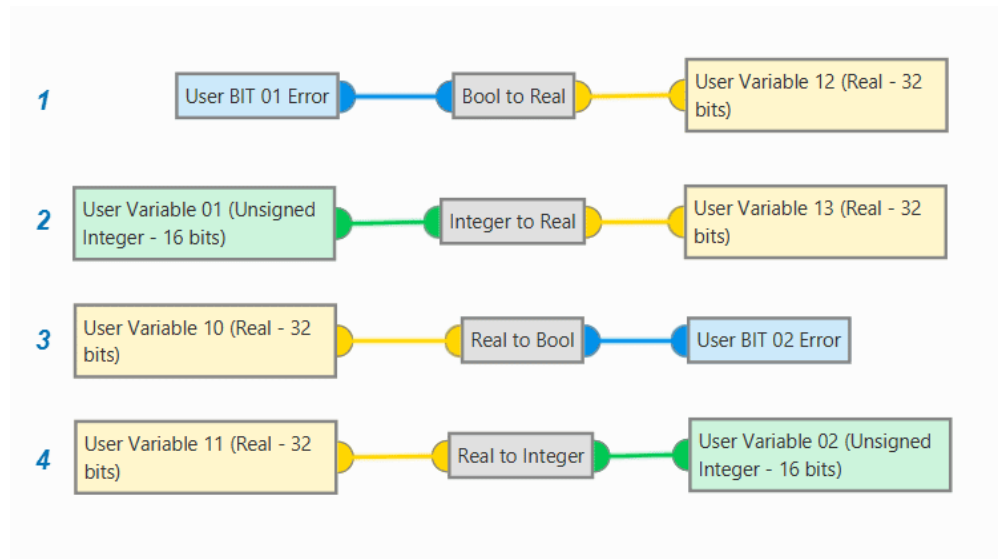


Ramp Block

7.2.10.9 Type Casting Blocks

These blocks allow to change from one data type to another. There are four different blocks available:

1. **Bool to real:** it transforms a boolean variable to a real variable.
2. **Integer to real:** it transforms an integer variable to a real variable.
3. **Real to bool:** it transforms a real variable to a boolean variable. Any number but 0 will be transformed to **true**; the rest (negative numbers included) will yield **false**.
4. **Real to integer:** it transforms a real variable to an integer variable.

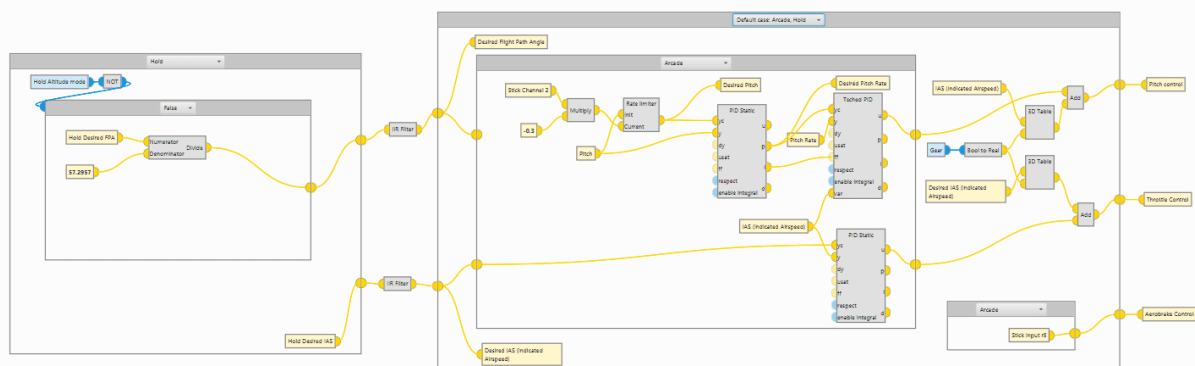


Type Castings Blocks

Programs are the core of Veronte Autopilot.

A **Program** is a custom algorithm executed by Veronte. While their main purpose is the control of the aircraft, **Programs** can be used to develop a wide variety of applications, from simple math operations to complex estimation filters.

Programs provide the user with a **block programming interface** that Veronte will then execute at core frequency:



Block Program Example

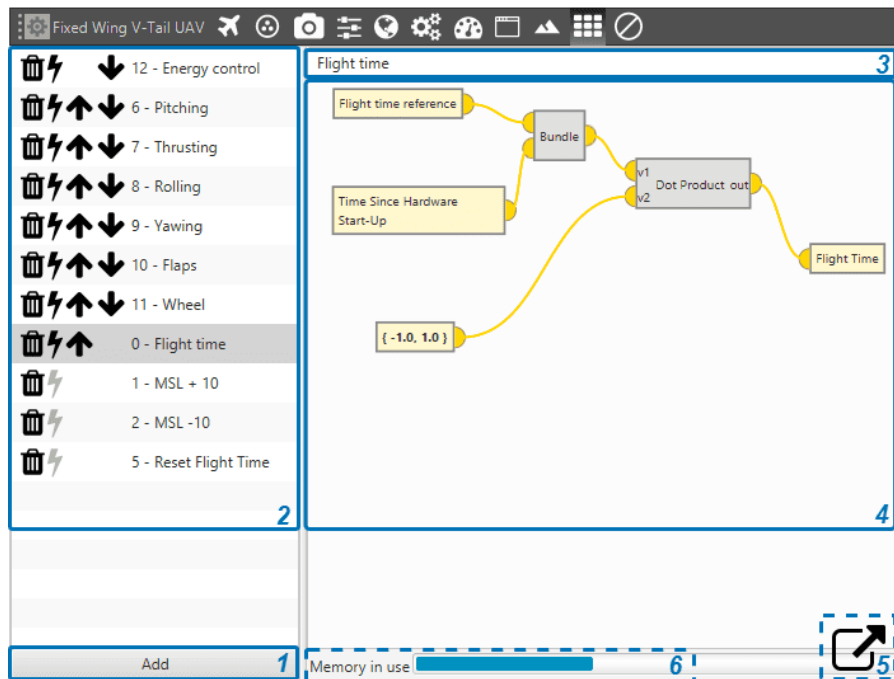
The different types of blocks available in Veronte **Programs** are:

- **Control:** control-related blocks (simple PID, scheduling PID,...).
- **Data source/sink:** Input/Output blocks. **Programs** can have access to any variable available within Veronte system. Results can then be stored in Custom variables for display, use as a different program input, feedback,...
- **Execution flow:** programming-like blocks for operation flow control. These blocks allow to alter parts of a program depending on a condition (If-Else, Integer Case, Phase Case,...)
- **Logic:** logical gates to operate with boolean variables.
- **Math:** mathematical blocks, which include a variety of mathematical operators: basic (sum, multiply, square root,...), trigonometric (sine, cosine, tangent,...), vectors (norm, dot product, rotations,...).


- **Mode/AP selection:** blocks that allow to interact with flight modes and redundancy (4x Veronte).
- **Positions:** blocks for operating with position-type variables (create position, read position, relative position,...).
- **Signals:** blocks for signal processing (IIR filter, rate limiter,...)
- **Type casting:** blocks for variable conversion (Real to BIT, Integer to real,...)

Block Program Interface


Accessible from the *Setup* menu, allows for the creation and edition of block programs:



Programs Menu

1. Add a new empty **Block Program**
2. Program list. From here programs can be selected for edition or deleted. Clicking on  will delete an existing **Program**.
3. Program name.
4. Program definition.
5. 'Pop up' the program edition window. This allows the user to have a clearer view of the operation while editing.
6. Estimation of the remaining memory available. If no more memory is available, no new blocks will be allowed to be created. The allocated memory for each block depends on the block type.

Execution

The icon  indicates that the program will be executed periodically, **at core frequency**.

Use  to determine execution order of existing programs. **Programs** are executed from top to bottom.

Clicking on  will toggle execution mode. Inactive  blocks will not execute periodically, but can be executed using *Run Program Actions*.

Danger: Applying changes to programs during an operation can cause changes in the behavior of some blocks.

For example, the **Integral** terms in *PID Blocks* will be reset.

Applying changes during an operation is **highly unrecommended**. If done, please make sure that operational safety is not compromised.

Blocks






Right click on the edition screen to open the block list. Select a block to import it into the program.

Click and drag on blocks to re-locate them:

Adding Blocks

Double click on a block to open its configuration menu.

Block inputs and outputs use a colour code in order to indicate variable types:


-  BIT Boolean variables
-  16-bit Integer variables
-  32-bit Real variables.  connectors can also be **Arrays** of real variables.
-  position variables

An input and an output can be connected directly using the mouse:

Linking Blocks

An input and output with different variable types cannot be connected without a *Type Cast*.

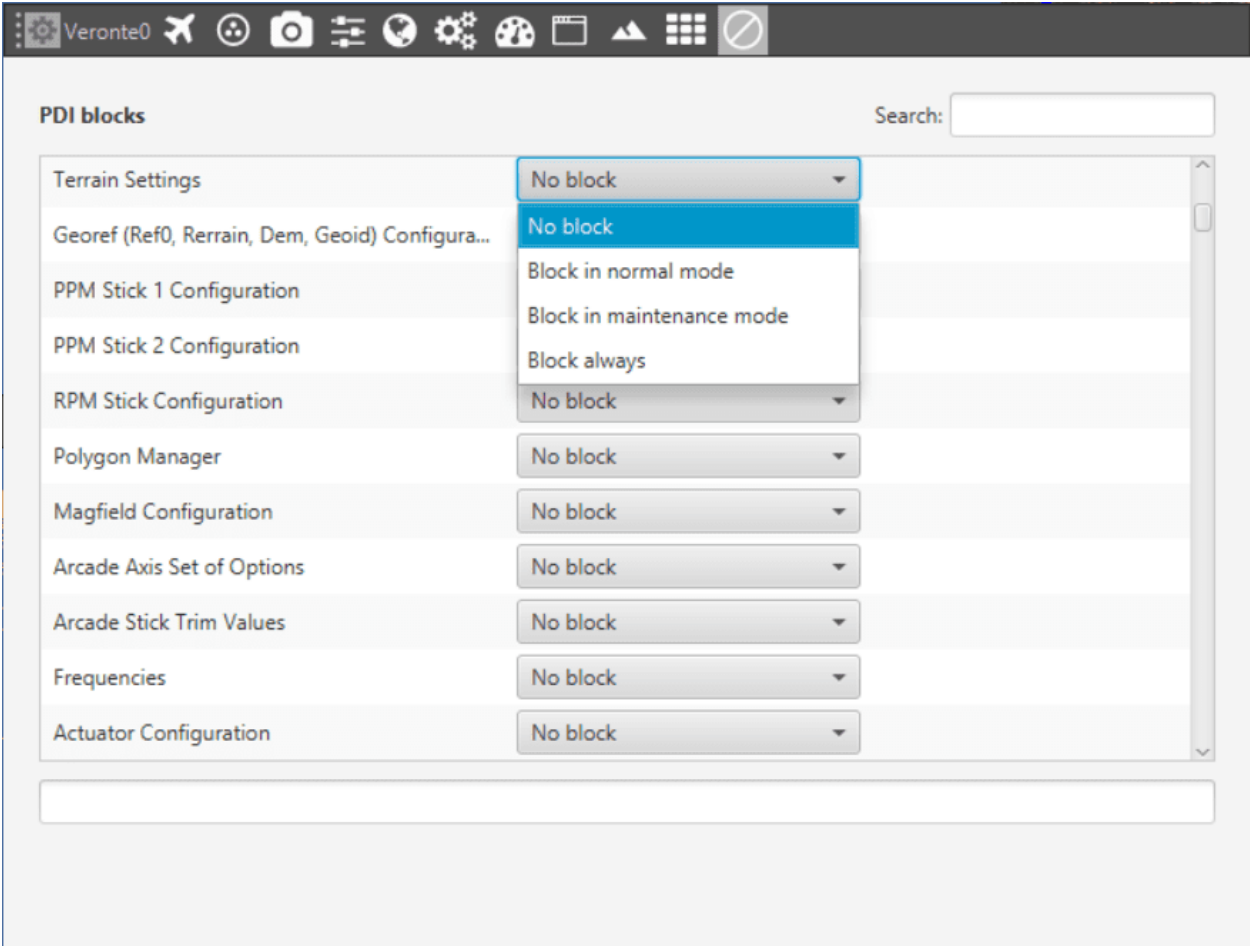
All inputs of each block **must** be connected, otherwise Veronte will report an **error**.

An exception to this rule are translucent inputs , which are optional. These inputs will have a default value if not linked.


Outputs do not need to be linked.

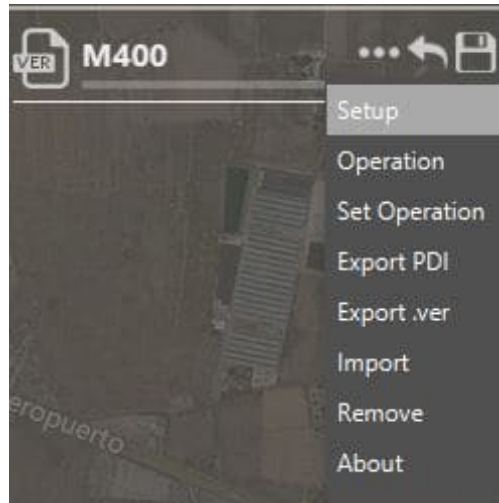
7.2.11 Block PDIFs

Block PDIFs avoid the user changing certain parameters, settings or programs of Veronte. It is shown in the picture below:



Block PDIFs

You can choose when theses options are disabled (in normal mode, in maintentance mode or always).
To display Veronte Setup Toolbar, when the autopilot is connected or an offline configuration is opened, in the side panel, click on  and then click Setup. This toolbar allows the user to modify the main features of the Veronte Units.



Setup Toolbar

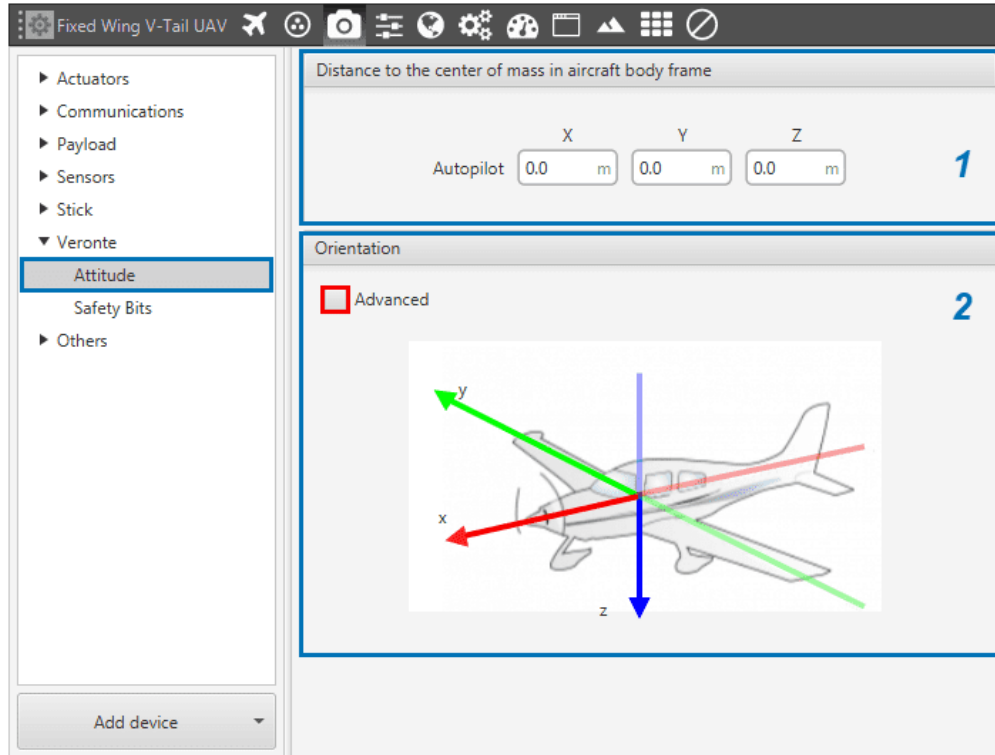
The different elements of the setup toolbar are detailed in the following table.

| Icon | Item | Description |
|------|--------------------|--|
| | <i>Veronte</i> | Introduce Veronte information. |
| | <i>Connections</i> | Configure I/O connections on Veronte. |
| | <i>Devices</i> | Configure any connected devices: servo, radio, camera... |
| | <i>Control</i> | Introduce control variables and phase configuration. |
| | <i>Navigation</i> | Configure navigation parameters on the system. |
| | <i>Automation</i> | Configure automatic actions on event detection (go home, change phase...). |
| | <i>Variables</i> | Customize variable names and traffic: log, telemetry... |
| | <i>Panel</i> | Configure Veronte Panel layout. |
| | <i>HIL</i> | Configure parameters for Xplane Simulator. |
| | <i>Programs</i> | Customize algorithms executed by Veronte. |
| | <i>Block PDIFs</i> | Block user control in PDI configuration. |

Each option will be explained in detail in the next sections.

7.3 Installation

The menu found in **Devices -> Veronte -> Attitude** (see the Figure below) allows the user to define the orientation of the autopilot with respect to the platform once it is installed. Aircraft axes are defined according to international aviation convention. Veronte axes are drawn on the autopilot's external case as defined in the *Hardware Installation*.



Veronte Attitude Menu

The menu consists in:

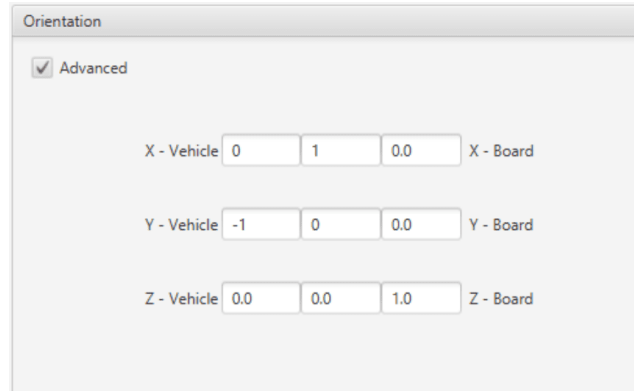
1. Distance to the center of mass in aircraft body frame.

The autopilot's distance to the centre of mass must be defined. This distance is entered in meters and accordingly to aircraft axes. This information is used to take into account the moment produced by the weight of the autopilot.

2. Orientation.

It is not compulsory to install the autopilot aligned with the aircraft axes. In order to indicate the autopilot's relative position inside the platform, select the advanced option. A matrix relating vehicle axes and autopilot axes is needed to be filled in.

The case of a non-orthogonal installation can be covered. If a simple rotation is introduced, for example, the autopilot's case having its x axis directed to the right wing (y - vehicle) the matrix should be completed as follows:



Orientation

☒ Advanced

X - Vehicle X - Board


Y - Vehicle Y - Board

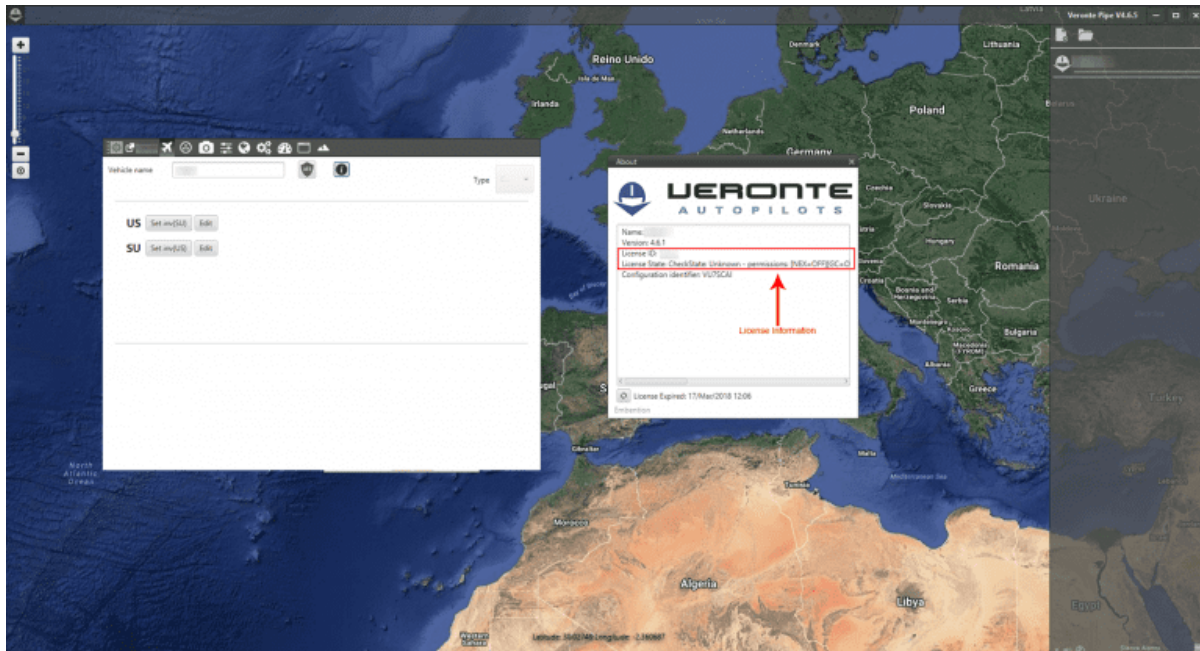
Z - Vehicle Z - Board

Veronte Advanced Orientation

7.4 License & Safe Mode

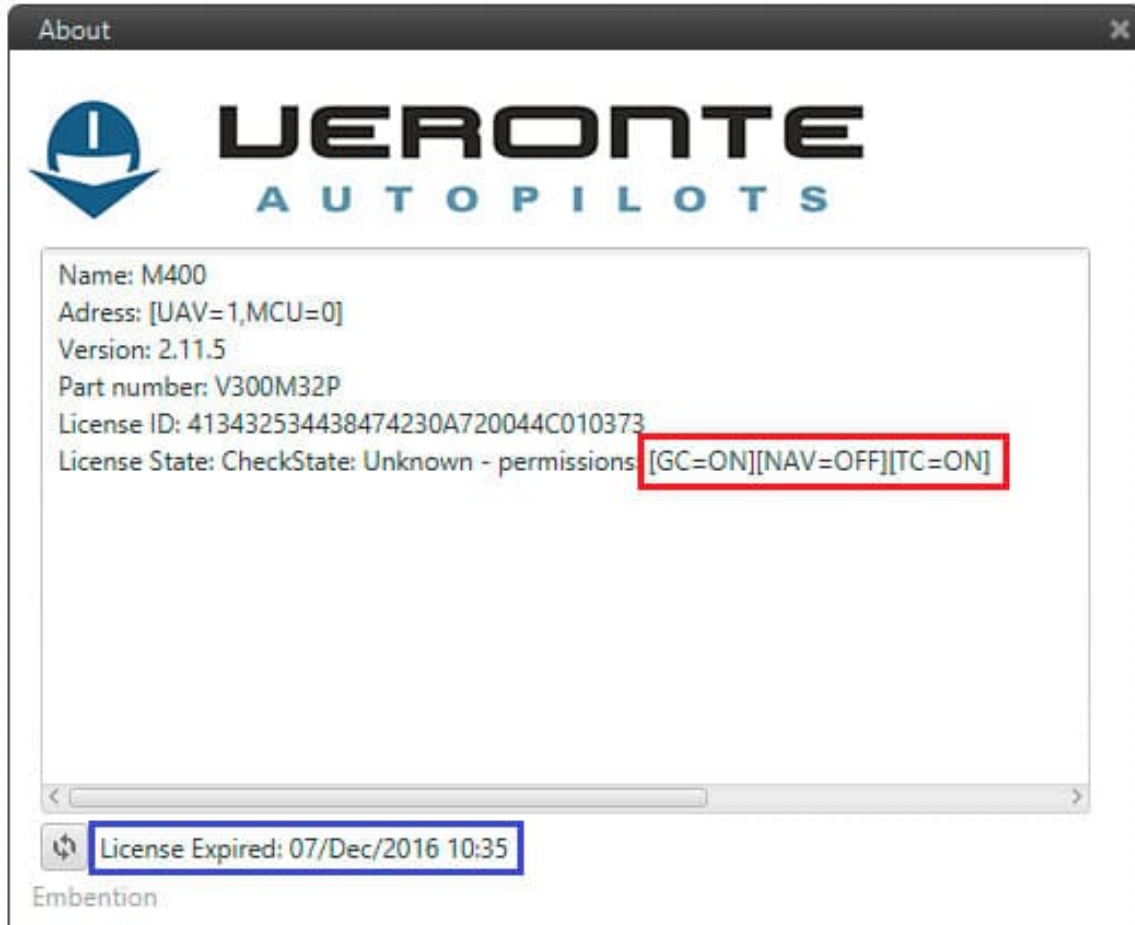
7.4.1 License

To access the License you have to open setup, then select Veronte and click on . In this tab you can check you License of your Veronte. You will have a different License Id for each Veronte (so, License Id Veronte Air is different from License Id Veronte GND). To renovate Veronte License, you must connect Veronte to Internet biweekly.



License

The license panel is useful to check the following information:



License panel

The panel is useful to check the following information:

1. UAV name
2. UAV Address
3. Software Version
4. Autopilot Part Number
5. Software version
6. License ID
7. License State (red)
 - GC: Guidance and Control
 - NAV: Navigation
 - TC: Transmission (RC)
8. License Expiration Date and Time (blue)

The license state shows the possible action for the selected autopilot (ON=Active, OFF=Not Active). The Ground Autopilot license state for a normal configuration is **ON: OFF: ON**. The Air state is **ON: ON: ON**.

7.4.2 Safe Mode

The safe mode is used to avoid malfunctioning when the autopilot is trying to charge a corrupt file.

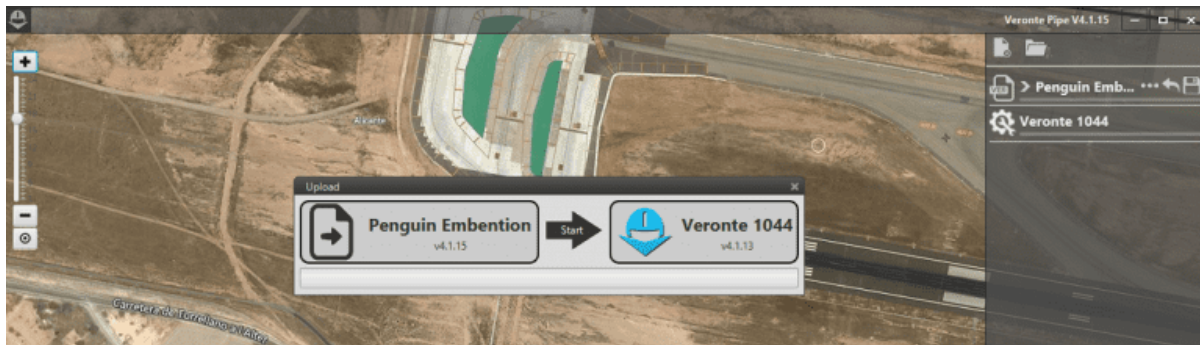
When Veronte is trying to charge a configuration file in its SD card and there is some problem, the safe mode avoids that the autopilot tries to charge the corrupt file again and again entering then in a loop that would not allow communicating with it. Instead of that, the system charges a default configuration stored in its flash memory and enters in what is called the safe mode, allowing the operator to change the configuration and send it again to Veronte.

It is also possible to force the enter in the safe mode by turning the autopilot off and on quickly. Safe mode displays the following window.



Safe Mode

It is possible to **Change setup** and select the configuration file which will be loaded on the autopilot. When a file is selected, a new window will be displayed in Veronte Pipe showing the version and identification of the configuration file and the autopilot where the file is going to be loaded.



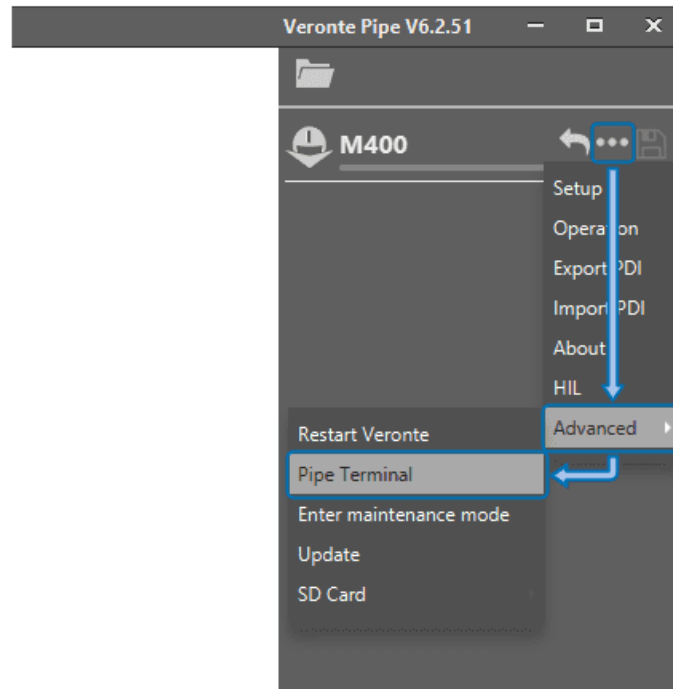
Upload Configuration – Safe Mode

With this tool, the configuration file is loaded directly on the autopilot. Now Pipe is only a tool to load the file from the computer to the autopilot and the configuration parameters will not be shown in the software window before being loaded on the autopilot.


The other option available allow the user update the Veronte Unit, this is explained in **Update Onboard Software**.

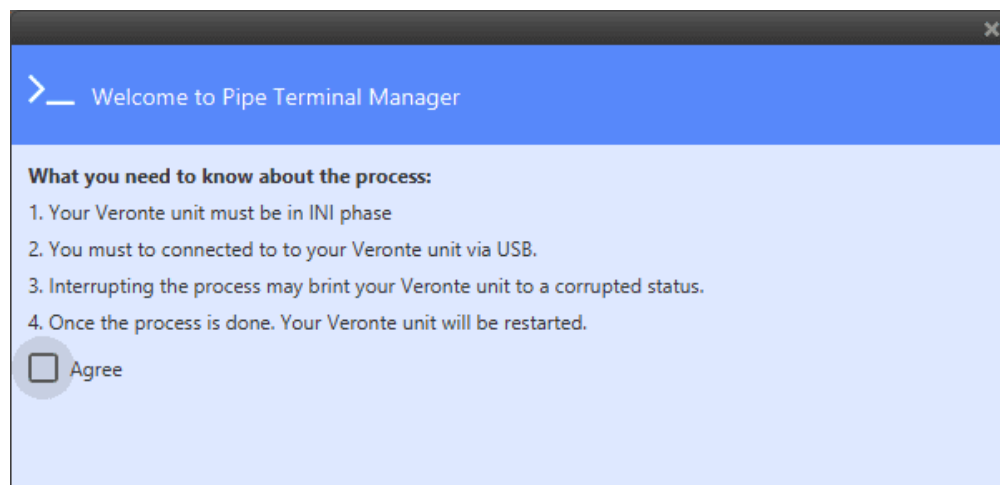
7.5 Veronte LOS (Line Of Sight)

This panel allows the user to easily access to the Pipe Terminal and modify the desired radio module settings. In the case of Veronte 4.0 and 4.5, they are integrated with Microhard Pico Series internal radio modules and their configuration can be modified by using the Microhard Setup Helper.

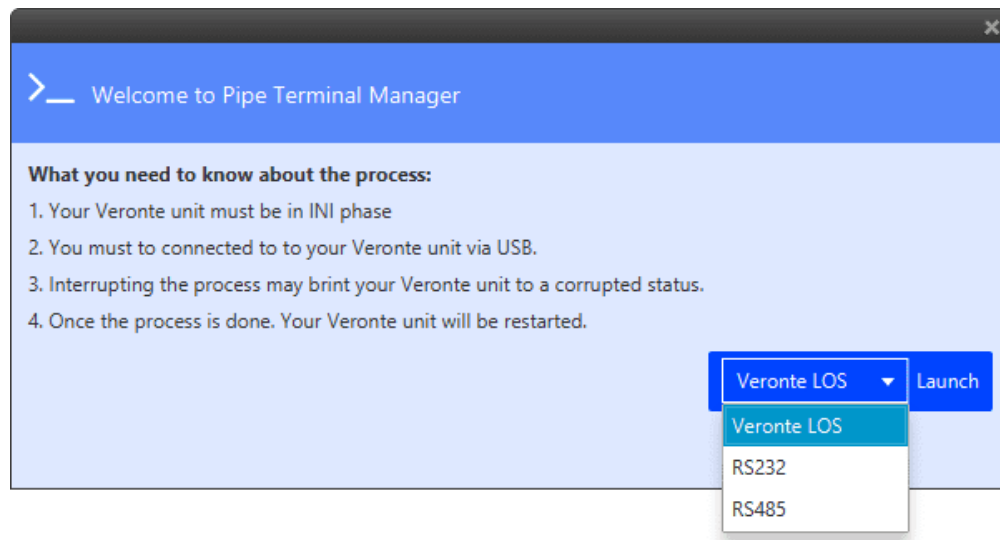


Veronte Pipe Terminal Setup

In order to start the configuration, the user should open the Setup by clicking on , then on Advanced and finally on Pipe Terminal. After this, the following window will show up.



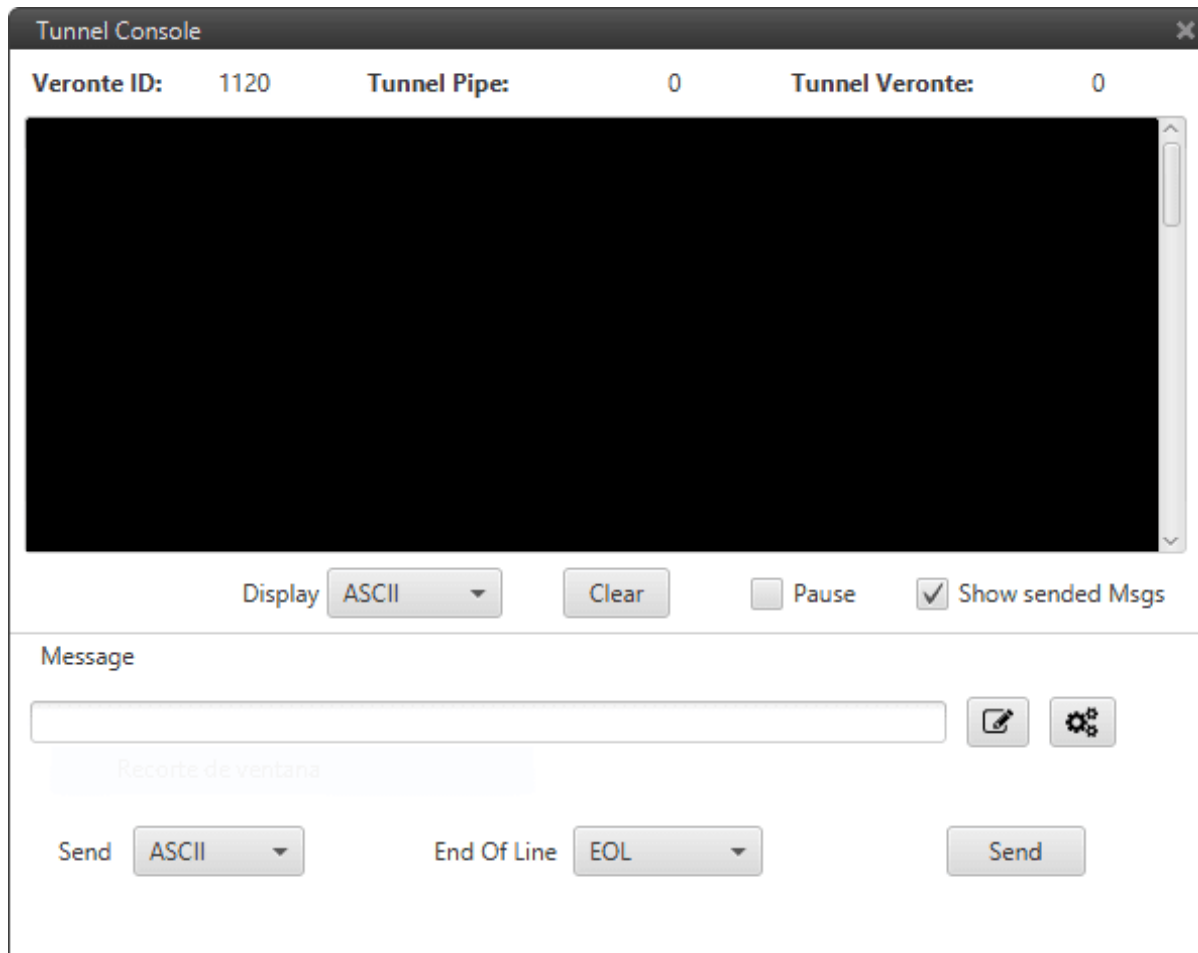
Veronte Pipe Terminal Manager agreement



Serial selection

Veronte Pipe is able to automatically set up the required tunnel over LOS, RS232 or RS485 depending on the user configuration and needs. In case of internal radio module configuration, the LOS option must be selected. If some external module is plugged over some external serial port, then RS232 or RS485 option must be used.

Once the process is launched, the Tunnel Console shows up.



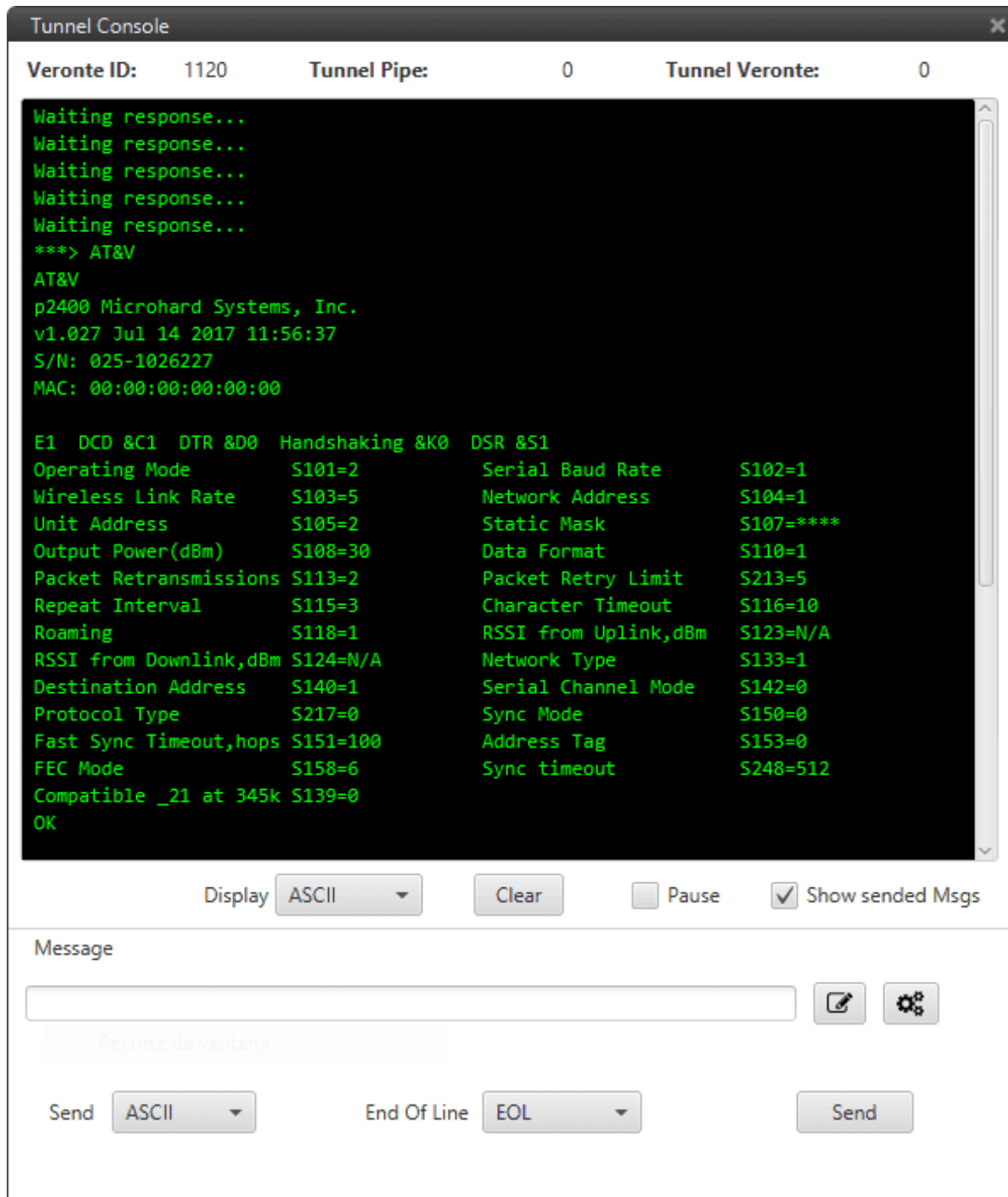
Tunnel Console

This console allows the user to communicate with the desired device (internal LOS or external device) by configuring the listed parameters:

- **Display:** Data can be displayed in Ascii or Hexa format.
- **Clear:** Clears the last console data.
- **Pause:** When the data stream is running, it is possible to pause it by enable this option.
- **Show sent messages:** If enabled, the sent messages are shown in the console.
- **Message:** Message content.
- **Microhard setup helper:** Check the related sub-section below.
- **Send:** Data sent in Ascii or Hexa format.
- **End Of Line:** Message “End of line” configuration.
- **Send:** Sending button. When clicked, the current message is sent.

7.5.1 Microhard setup helper

The **Microhard setup helper** can be used in order to easily configure the Microhard internal modules. In order to do this, the user should click on the dedicated button and the Wizard will automatically start the radio searching.



Terminal Console - Radio found

Once the radio is found, its model is shown in the upper-left corner of the window and the configured parameters in the console. The following Wizard menu is displayed.

Radio Wizard

Radio model **P2400**

Radio configuration

Out power: 30 dBm (1W)

Address: Master

Connection: PP

Network address: 1

Packet retransmission: 2 (0-254)

Commands:

```
AT&F6
ATS108=30
ATS104=1
ATS105=1
ATS102=1
ATS113=2
ATS103=5
AT&W
ATA
```

Reset

Licensed band configuration

Licensed Band Frequency: 430.0 Hz

Frequency hopping table

| Table 0 | | Table 1 | |
|---------|--------|---------|--------|
| #1 | 410.00 | #2 | 410.00 |
| #3 | 410.00 | #4 | 410.00 |
| #5 | 410.00 | #6 | 410.00 |
| #7 | 410.00 | #8 | 410.00 |
| #9 | 410.00 | #10 | 410.00 |

☒ Link values from Table1 to Table0

Send message

Radio Panel

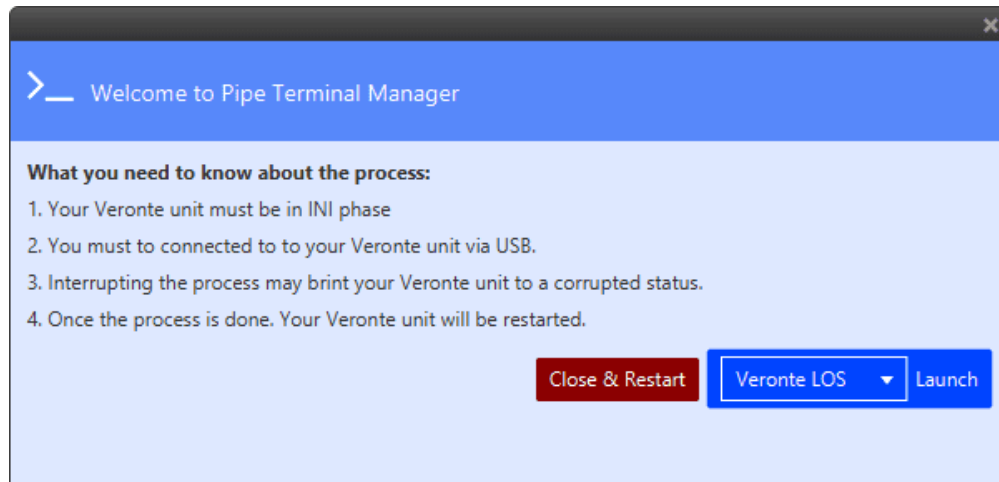
These parameters have to be set accordingly to the radio module installed in Veronte.

- **Radio Module:** Each Veronte has only one of the following radio modules P400, P900 and P2400. Select here the radio module installed.
- **Output power:** Sets the available power output.
- **Connection:** Point to Point or Point Multi Point. By default all units are paired with PP connection.
- **Address:** Master or Slave . Select the role for each Veronte by selectig Master or Slave. By default the air unit is defined as Slave and ground units as master.
- **Network address:** the network address is a number that must be equal between Verontes using the same network.
- **Packet retransmission:** Each data pack is sent as much times as defined here.

By pressing **Send message**, the user is able to send all these parameters to the Microhard module.

The commands can be also manually sent. For advanced Microhard configuration, please check the specific radio module User Manual [here](#).

Once the configuration is sent, the console is closed and the **Close & Restart** button appears. By clicking this button, the user can exit the configuration process and the Veronte unit is restarted. The Microhard radio module is now configured.



Veronte Pipe Terminal closing and Veronte restart

7.6 Overview

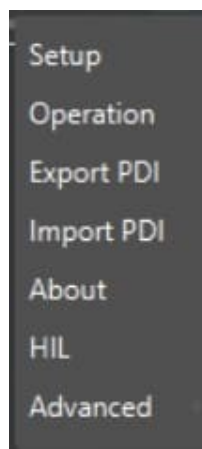
In this section, it is explained how to configure Veronte Units using Veronte Pipe. Each feature available is described in detail in following sub-sections.

Firstly, the user will find how to manage configuration files: import, export, update, etc. Later, this manual will focus on the setup toolbar, this allows the user to change the main settings of the Veronte Units according to the aircraft (Control Loops, Installation, etc.) and mission (Phases, Guidance, etc.) to perform.

Finally, the manual explains how to check Veronte's license, where the user can see the license ID and its status.

7.7 Side Panel Options

The Side Panel allows the user to manage Veronte Units and Configuration files. In this panel each item is displayed, when an autopilot is connected it will appear automatically. The following image shows both items and the options available.



Side Panel Options

The next table gives a brief overview of each option.

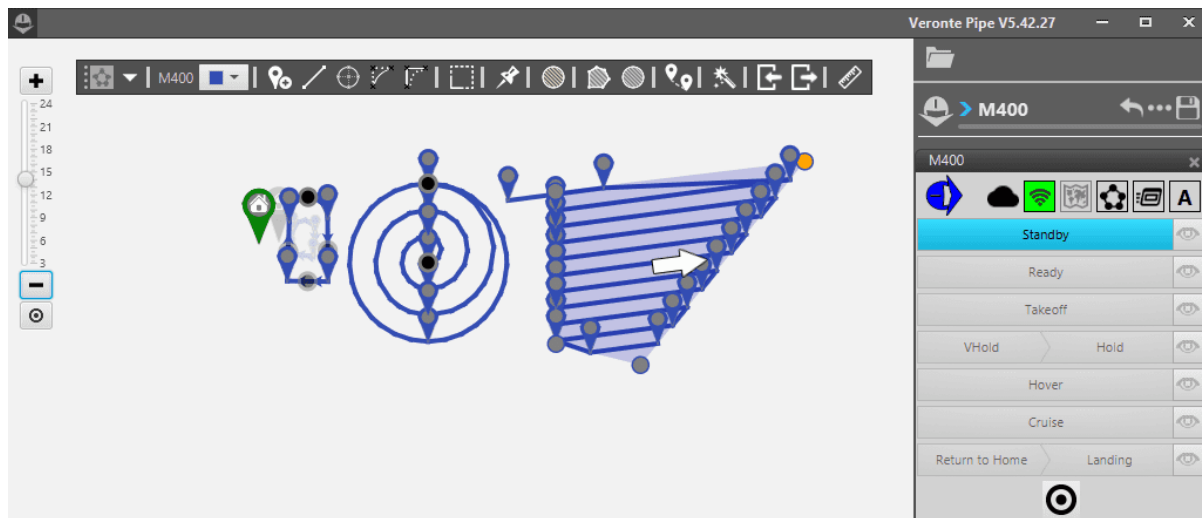
| Item | Description |
|----------------------|--|
| Setup | Displays the setup toolbar |
| Operation | Displays the operation toolbar |
| Set Operation | Choose one operation to be loaded |
| Export | Export configuration on Veronte to disk in .ver or PDI files |
| Import | Import a configuration from disk |
| About | Displays information on Veronte SN, License and Status (only available for connected Veronte configurations) |
| HIL | Hardware in the loop simulation |
| Advanced | Update Veronte Unit and Reboot it |

MISSION

8.1 Setup

8.1.1 Introduction

A Mission can be configured making use of the Mission Menu. Missions can be created and managed through the Mission menu. First, make sure the autopilot unit where the mission has to be upload is selected, i.e. the Air unit. Right after, the aircraft trajectory can be drawn using the graphical tool of this menu, along with other auxiliary elements such as polygons, event markers or obstacles markings.




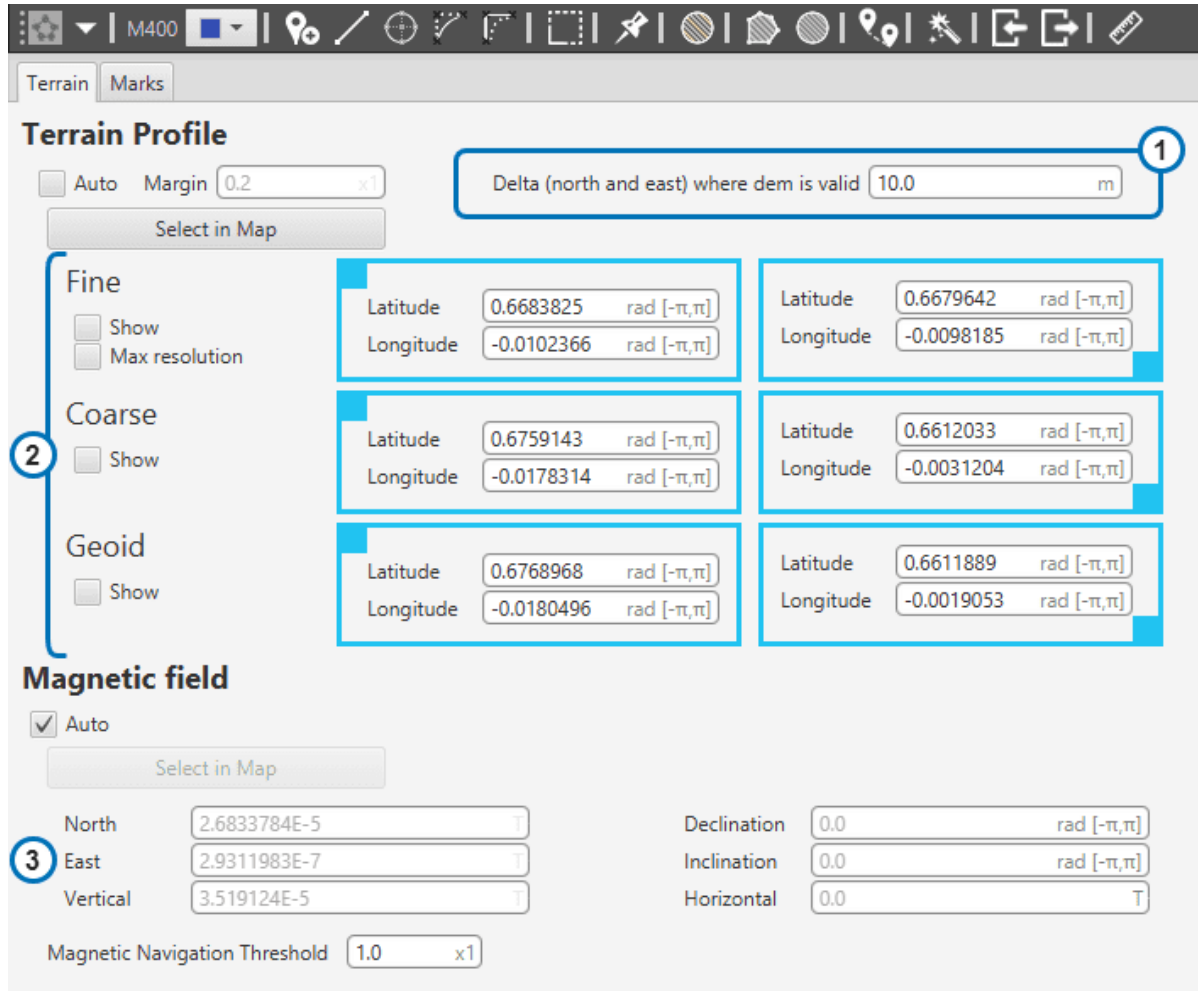
Mission Setup Examples

The user can manage two or more missions simultaneously open in Pipe, but be careful not to mix configurations. The paths created will be directly linked to the Cruise guidance mode, so each time the aircraft is in a phase with cruise guidance, it will start to follow the track created with the mission menu.

8.1.2 Terrain profile and magnetic field

Before starting the configuration of a new mission, the terrain profile and the magnetic field, Magfield, have to be set.

Click on Open details button, , to access the settings.



Terrain Profile

☐ Auto Margin x1

Delta (north and east) where dem is valid m

Select in Map

Fine

☐ Show ☐ Max resolution

Coarse

☐ Show

Geoid

☐ Show

Magnetic field

☒ Auto

Select in Map

North T Declination rad [- π , π]

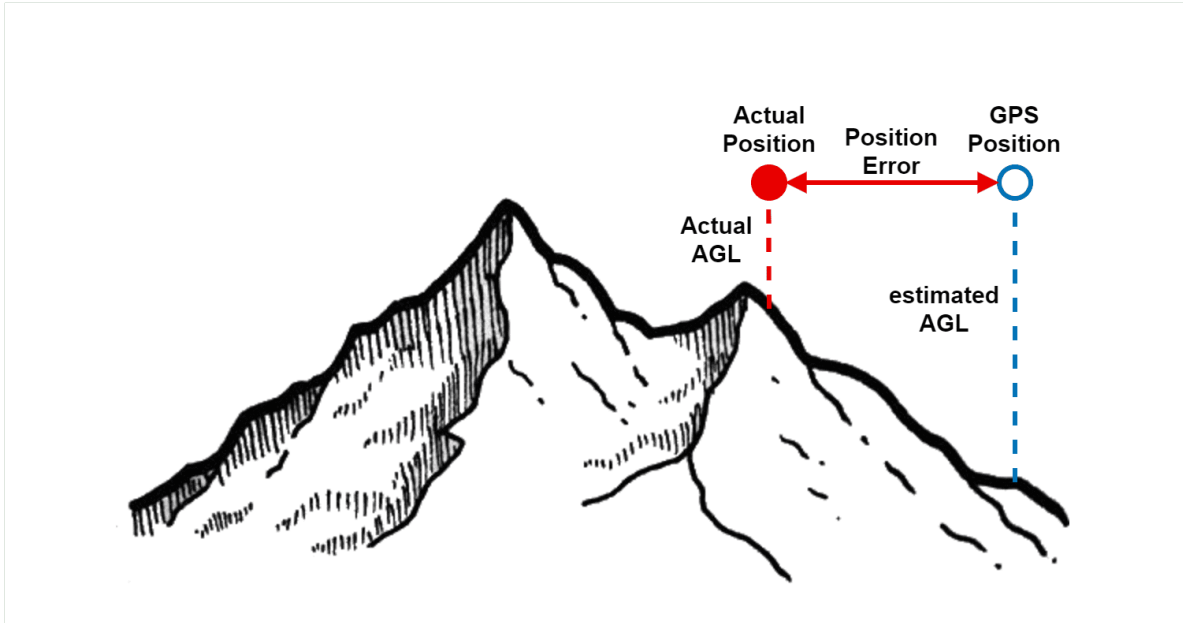
East T Inclination rad [- π , π]

Vertical T Horizontal T

Magnetic Navigation Threshold x1

Terrain profile and magnetic field settings

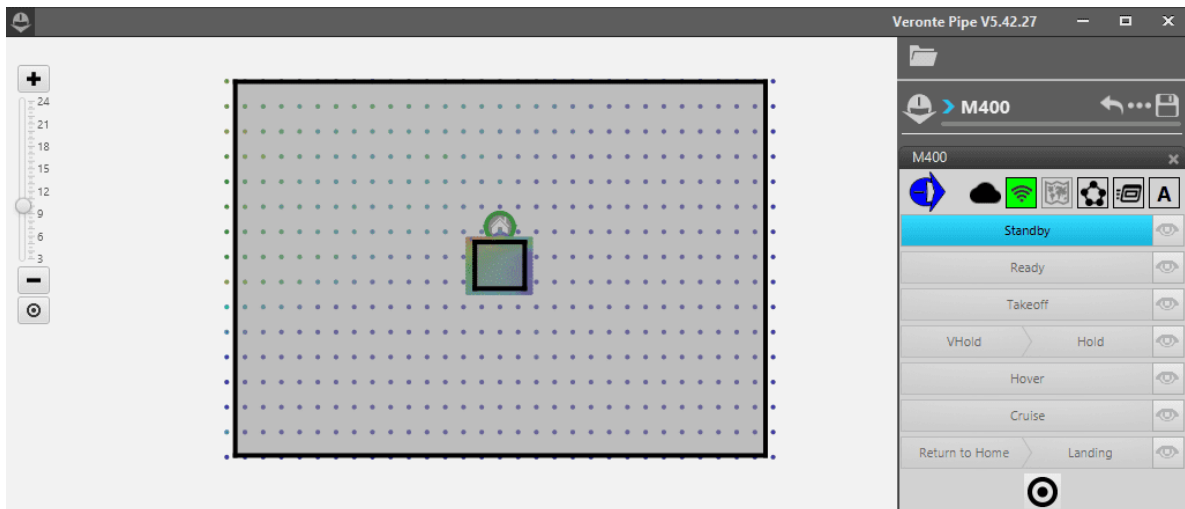
1. **Delta parameter:** The distance to the ground (AGL) may be measured through two or more systems. Usually, an altimeter like a LIDAR in conjunction with the GPS signal and the meshes information are used. During flight, it is possible that the GPS position error is large enough that the height provided by the meshes does not correspond with the actual position. In order to avoid problems, the Delta parameter can be defined. This parameter defines a circumference radius where both systems will be used. If the estimated position error is bigger than the delta parameter, only LIDAR data will be used.



Delta Parameter

2. **Mesheres of terrestrial mode:** three different are established: Fine, Coarse and Geoid.

- **Fine:** Is the smaller mesh. Contains detailed information about terrain altitude.
- **Coarse:** Medium mesh with not so much detail.
- **Geoid:** World mesh which provides the geoid altitude.



Fine and Coarse meshes setup

If “Auto” is enabled, the system will automatically place and adjust the meshes to optimal start dimensions. The user can modify them manually, also. First, deselect the “Auto” option. The coordinates of each upper left corner and lower right corner of the meshes’ rectangle can be introduced. Also, they can be set by hand by clicking and dragging the meshes and its edges around the map. In order to do that, click on “Show” to display the meshes on the screen.

In general, increasing meshes size will mean lower area definition. And greater resolutions smaller meshes, because it implies heavier data files.

“Margin” is the percentage at which the system will recalculate the mission if the route is displaced. In other words, if the mission is displaced 60% out of the area (Coarse or Fine) and the margin is set to 80%, the mission will be not recalculated. If the mission is 81% (or more) away from the previous one, the system will recalculate the mission. A low level (or zero) margin means more terrain profile precision but the system will have to recalculate the meshes more times (or each time) when the mission is modified.

3. **Magnetic vector:** the magnetic vector of the mission’s area should be introduced. As far as the Magfield is concerned, it is recommended to select the “Auto” option to take the magnetic declination information of the mission area.

Warning: Check that meshes position are over the mission area before flying, especially if carrying out an operation in mountainous terrain.

8.1.3 Marks

Marks are an important feature to Pipe and the missions. Marks can trigger actions when the aircraft passes through them. Actions can be configured in the [Automations menu](#).

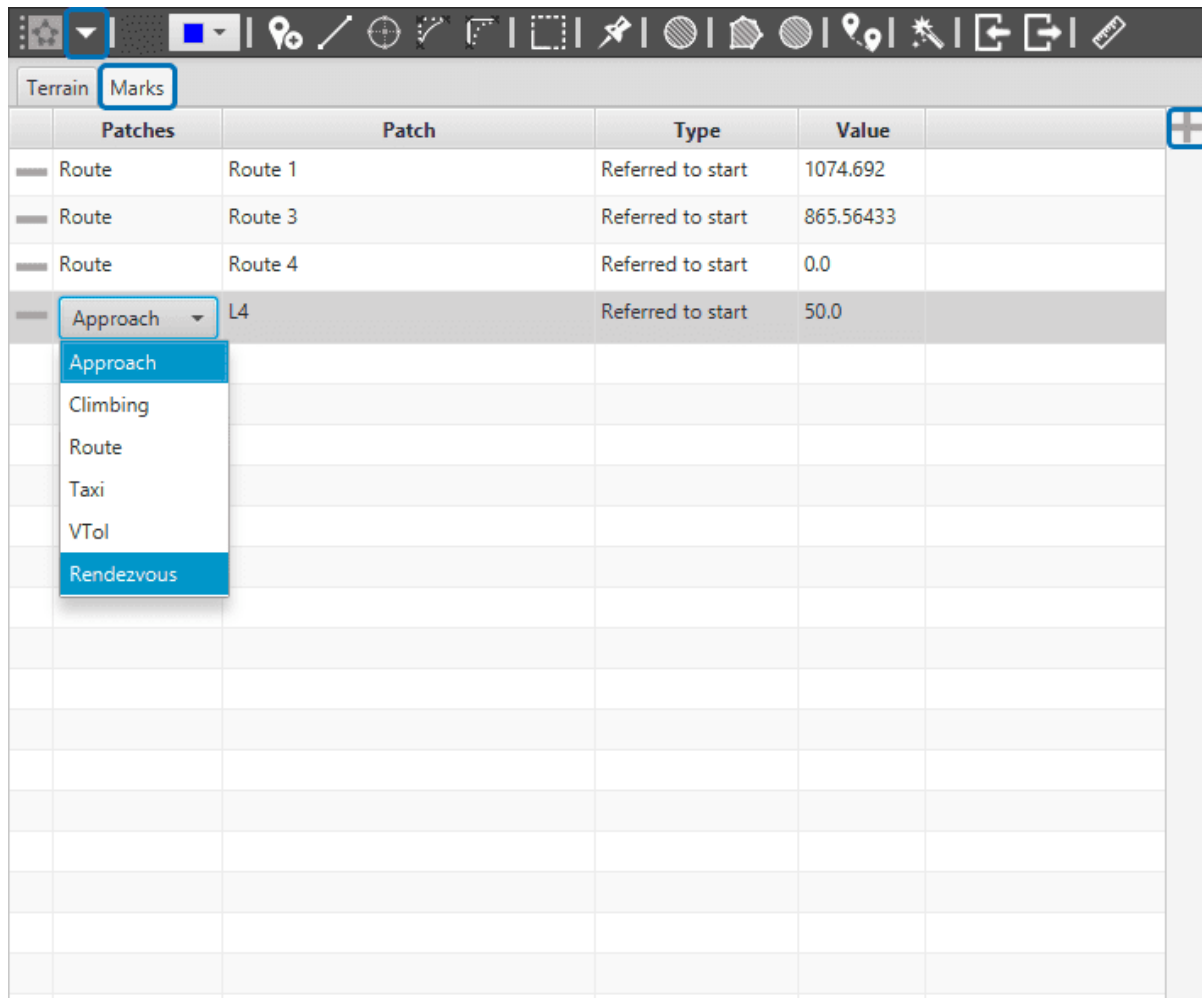
There are two possible options to add a new mark. One is through the “Event Mark” button on the Mission toolbar



, which allows the user to create a new mark on a **route** track directly or attached to an individual waypoint, or



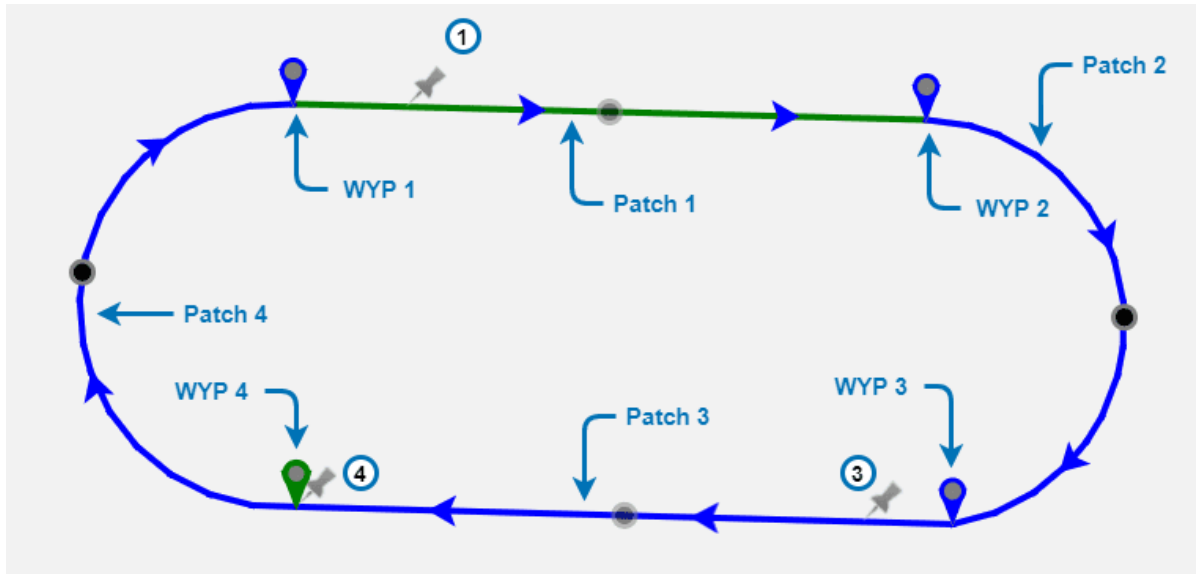
via the Marks flange on the Open Details by clicking on “+” at the upper right corner of the table. In both cases, all the marks created will appear on this menu:



Marks Menu and Guidance's patch selection

There are several options to configure the Marks:

- **Patches:** by double-clicking on "Route", a dropdown list allows the user to select different flight phases where the marks will be placed: Approach, Climbing, route... Except for Route patches, the rest of the phases are generated when the user, or an automation, activates them. As the user can not select this patches, they do not already exists, this option will create the mark automatically in the patch selected.
- **Patch:** Most Flight phases have predefined patches with specific names. The user can select where the mark will be placed on those patches. At the end of this section there is a table that summarizes available options. For more information about them, check the [Control menu](#) in Setup Toolbar.
- **Type:** right now the only possible option is "Referred to Start" of the selected patch.
- **Value:** Mark's distance to the start of the patch.



Route Patches

At the examples above there are four marks created. Three are attached to different route patches: 1,3 and 4. Route 1 and Route 3 marks were added by hand. Route 4 mark was introduced using this menu and is overlapping with waypoint 4 (value equals zero). Last mark was added to the approach L4 patch. **This mark could be linked to an automation that deploys landing flaps**, for example.

| | |
|------------|-------------------------------------|
| Approach | L0/A1/L2/A3/L4/L5 |
| Climbing | L4/A3/L2/A1 |
| Route | 512 patches |
| Taxi | Taxi1/Taxi2 |
| VTol | VTol1/VTol2/VTol3 |
| Rendezvous | Rendezvous1/Rendezvous2/Rendezvous3 |

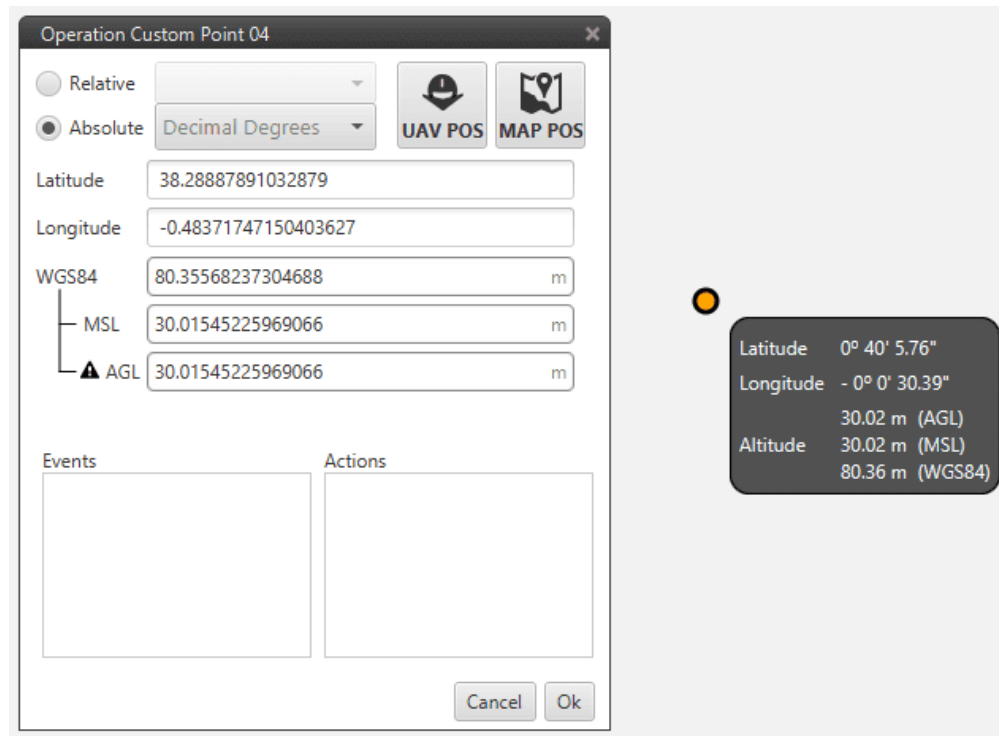
8.2 Tools

8.2.1 New Waypoint



New Waypoint Tool

Use the New Waypoint tool and press on the map for creating a waypoint. For moving waypoints, simply drag it to the desired position. If user wants to manually change the coordinates of the waypoint, it is possible to click twice the waypoint and the following display will appear for entering custom parameters. In order to only check them, a resume display can be obtained by simply passing mouse on the point.



Waypoint Settings

There exist two way of defining the position of a waypoint:

- **Absolute:** the coordinates of the waypoint are indicated trough the latitude, longitude and altitude (being possible to define this last one with respect to the ellipsoid, WGS84, to the sea level, MSL or to the ground, AGL). To introduce the point coordinates the user can select 3 and click on the map, introducing later the altitude manually, or press 2 that allows selecting the current position of the air or ground autopilot as the waypoint coordinates.
- **Relative:** in this case, the position of the waypoint is relative to another point. That point could be any platform fitted with a Veronte autopilot. It is possible to select up any waypoint, home or the UAV. It is common to select a waypoint with reference to the position of the ground station. In order to link an object of interest with the position of the ground autopilot please go to section [Sniffer](#).

The other two windows that appear on the waypoint menu are related with the automations of the system. There will appear all the events and actions related with the mission waypoints (Route, Track, Go To), so it is possible to indicate which waypoints will trigger the event or perform the action. To have a deep understanding on the system automations visit [Automations](#).

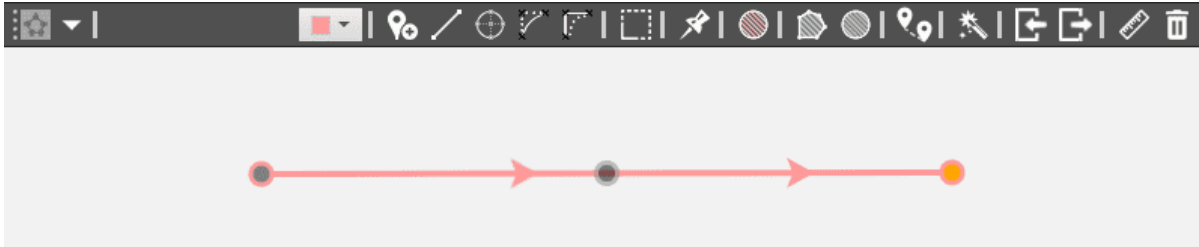
If you click the right button of the mouse yo can:

- Change the name of the waypoint
- Remove the waypoint
- Set the waypoint as start of the route


8.2.2 Segment

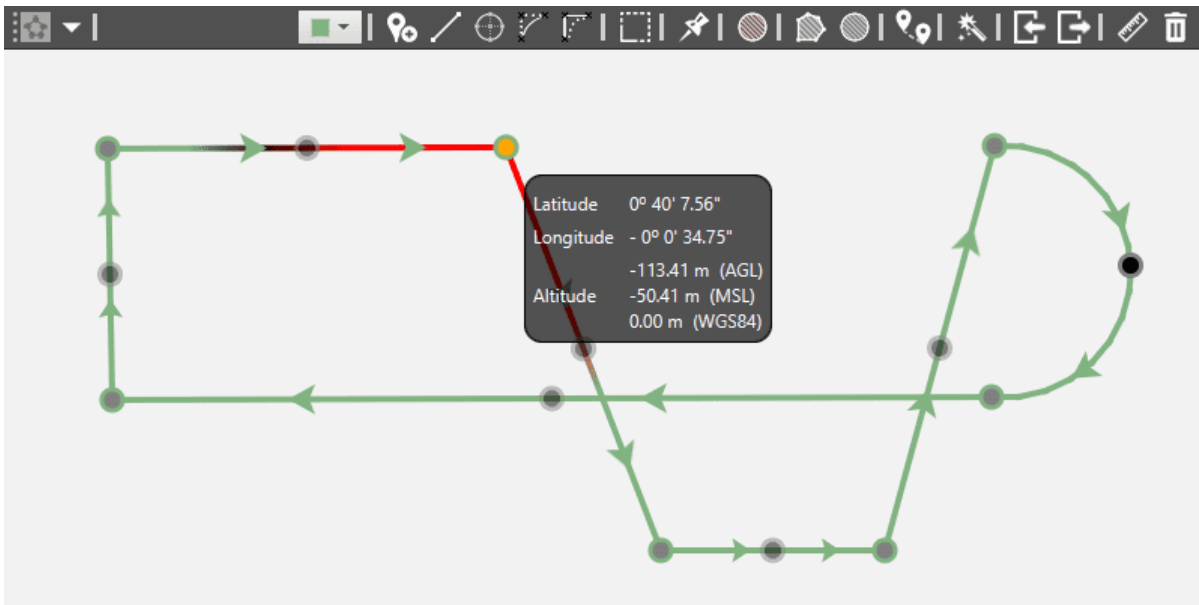


New Segment tool



New Segment

To create a new segment, click on  first. Then, select in the map the point where you want the segment to start. To finish the track click first the left button and then the right button. You can concatenate two or more segments clicking on several points in the maps with the left button, as you can see in the picture below:



Concatenated Segments with one waypoint below ground level

To delete a segment, remove one of the waypoints of its extremes. You can modify the concatenated segments moving the waypoint between them or create circular paths clicking and dragging the middle point of the path.

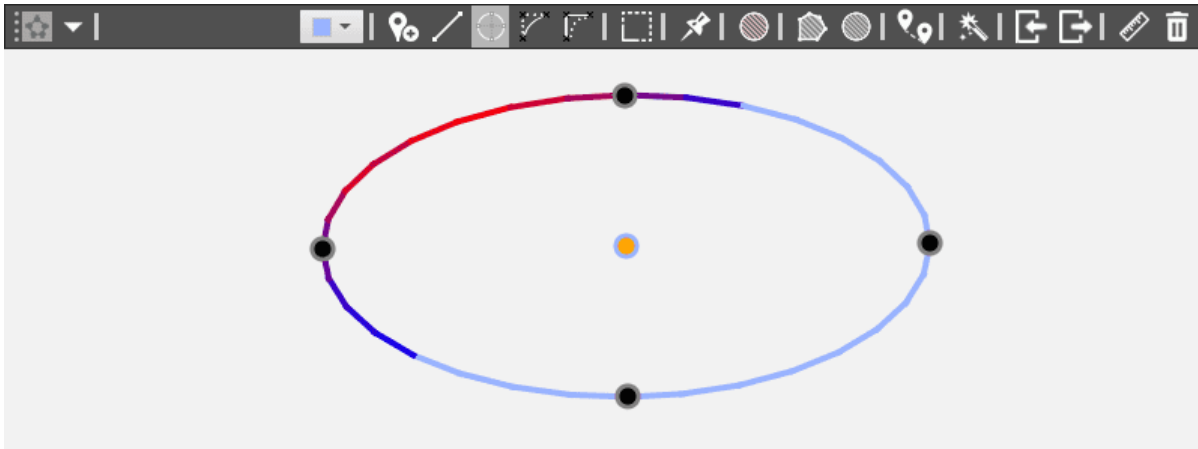
Warning: When you create a segment, you have to be careful with the height of the waypoints. **Please, remember to check their altitude.** If some of them are under the ground altitude indicated by the meshes, the segment will change its colors to red. To resolve this, change the altitude of the waypoints clicking on them twice.

8.2.3 New Orbit



New Orbit Tool

This tool allows the user to create a new orbit on the map. First, select a point on the map which will be the center of the new orbit. To move the orbit, drag the center point to the desired position. It is also possible to set the precise coordinates of the central waypoint by double-clicking on it.



New Orbit, transformed to an ellipse and partially below ground level

The initial circular orbit can be converted to an ellipse modifying the length of the axes the extremes. Click and drag on any of the four points to your preferences

To delete a orbit, remove the central waypoint.

Warning: When creating a orbit, **be careful with its altitude and the ground level** (all the points of the orbit will be at the same altitude) , if any part of orbit is below the terrain level indicated by the meshes, it will be coloured in red. Please, modify the altitude of the waypoint by clicking twice on the central waypoint.

8.2.4 Intersect Lines

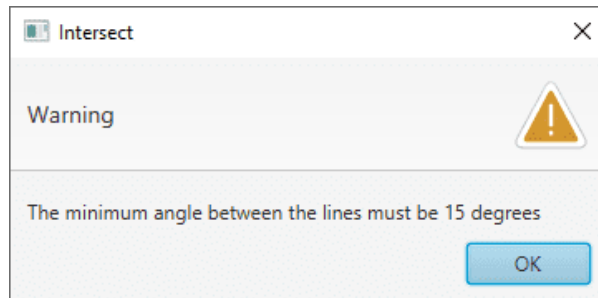


Intersect Lines Tool

This tool eliminates circular paths between linear paths by extending them until they intersect. To apply it, select the tool first. Then, click on the circular path you want to delete. Intersect Lines performs well reversing the fly-by tool:

Intersect Lines example

The angle between linear paths must be greater than 15 degrees to correctly undo the turn. In case the angle is smaller, a warning window will pop up. This feature prevents the creation of very long linear paths that intersect in the infinity.



Warning Message

Warning: Always check the altitude of the waypoint at the intersection

8.2.5 Fly By



Fly-By tool

This tool has been developed to improve turns of fixed-wings aircrafts. To explain better the utility of the tool we will use this example:

Fly-By example

To configure a Fly-By waypoint click on the tool's icon. Once it is selected, click on a waypoint. A new window will pop up asking to introduce the desired turn radius. When the radius is introduced the route will change and now the aircraft won't fly over the waypoint.

8.2.6 Multiple Choice



Multiple Choice Tool

Multiple Choice allows to move mission's path from one place on the map to another. Select the tool and create a rectangle by clicking and dragging until the waypoints you want to move fall into it. With the points selected, click and drag the selected path to the desired location. After that, click again in the tool's icon to deselect.

The tool can be used to remove multiple waypoints or paths too. This action can be performed by right-clicking on one of selected waypoints and clicking on Remove.

Drag and delete example

It is also possible to rotate the selection by clicking on the arrow symbol and dragging it.

Rotation example

Warning: Always remember to check the altitude of the waypoints by double clicking on them or in Operation menu, in the Waypoints tab.

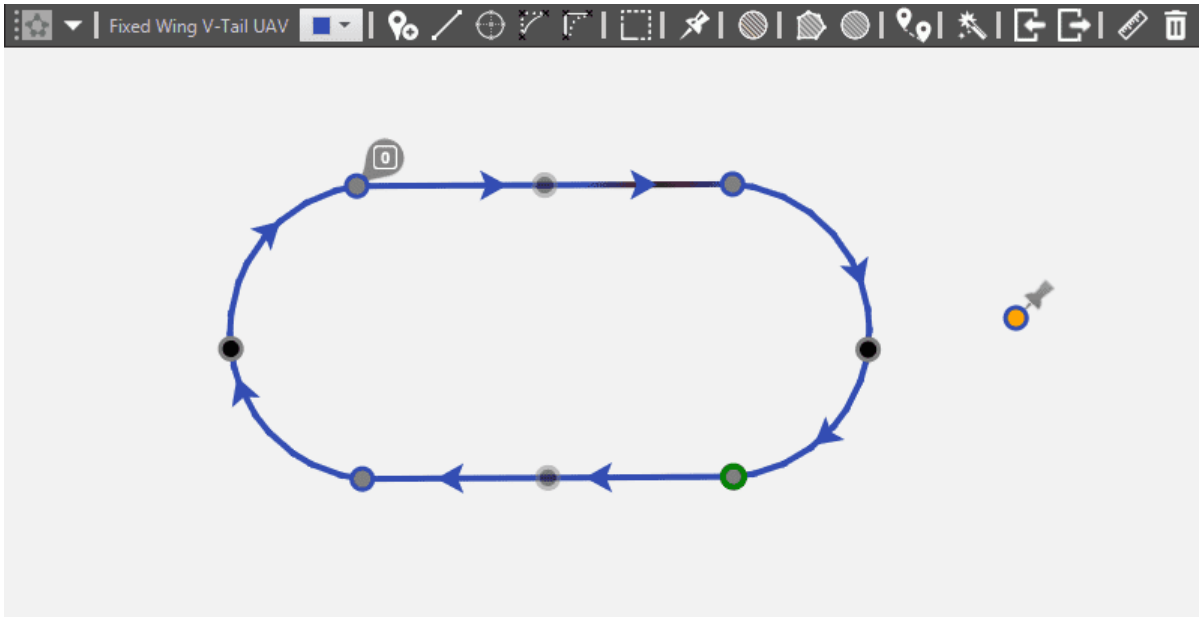
8.2.7 Event Mark



Event Mark Tool

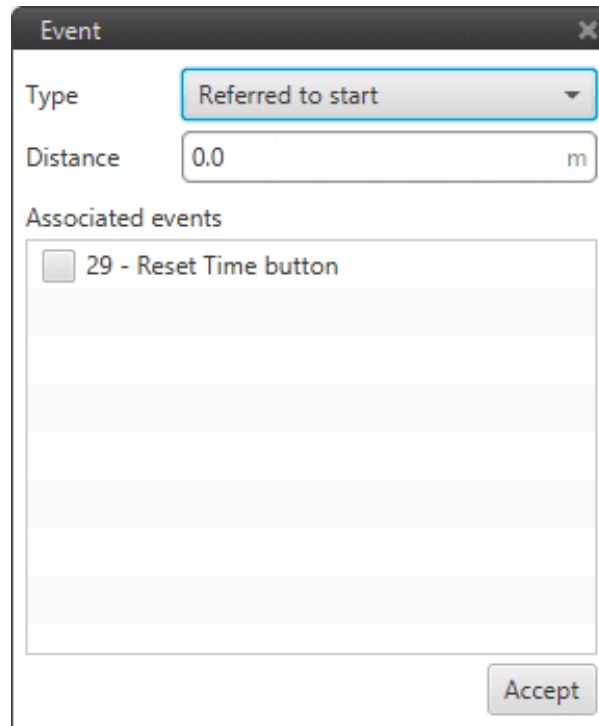
This tool allows to set an Event Mark over a track or attached to a waypoint. Marks are useful to start automations. When the aircraft flies over it, an event is triggered and it can be used as a condition to start a set of actions: add a lap to a counter, payload launch, take a photo, start video recording,...

To add an Event Mark, select the tool and click on the desired path point or waypoint. Initially, new marks appear as a pushpin. If you link an action to one of them, it will show the icon selected to this action.



Event Mark examples

To access the properties of the mark, make double click on it:



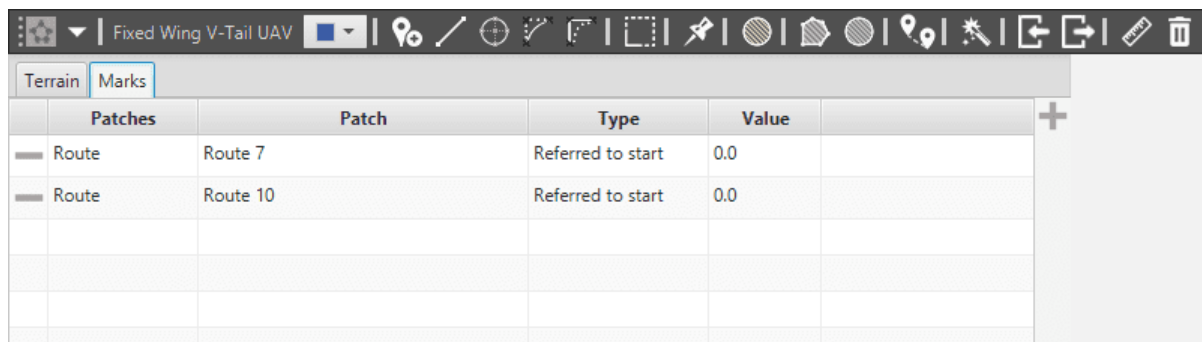
The dialog box titled "Event" contains the following fields:

- Type:** A dropdown menu with "Referred to start" selected.
- Distance:** A text input field with "0.0" and a unit "m" (meters).
- Associated events:** A list box containing one item: "29 - Reset Time button".
- Accept:** A button at the bottom right.

Event Mark properties

- **Type.** Allows the user to select the type of mark reference. Right now all the marks are “Referred to start”, that means the position of the mark is relative to the starting point of the path in which the mark is. I
- **Distance.** Is the horizontal distance along the path to the starting waypoint. If the mark is attached to a waypoint, distance will be zero.
- **Associated Events:** List of events that can be linked to the mark.

Remember to save the changes to make them effective. After saving, all the marks created will appear in the Marks tab of the Open Details menu:



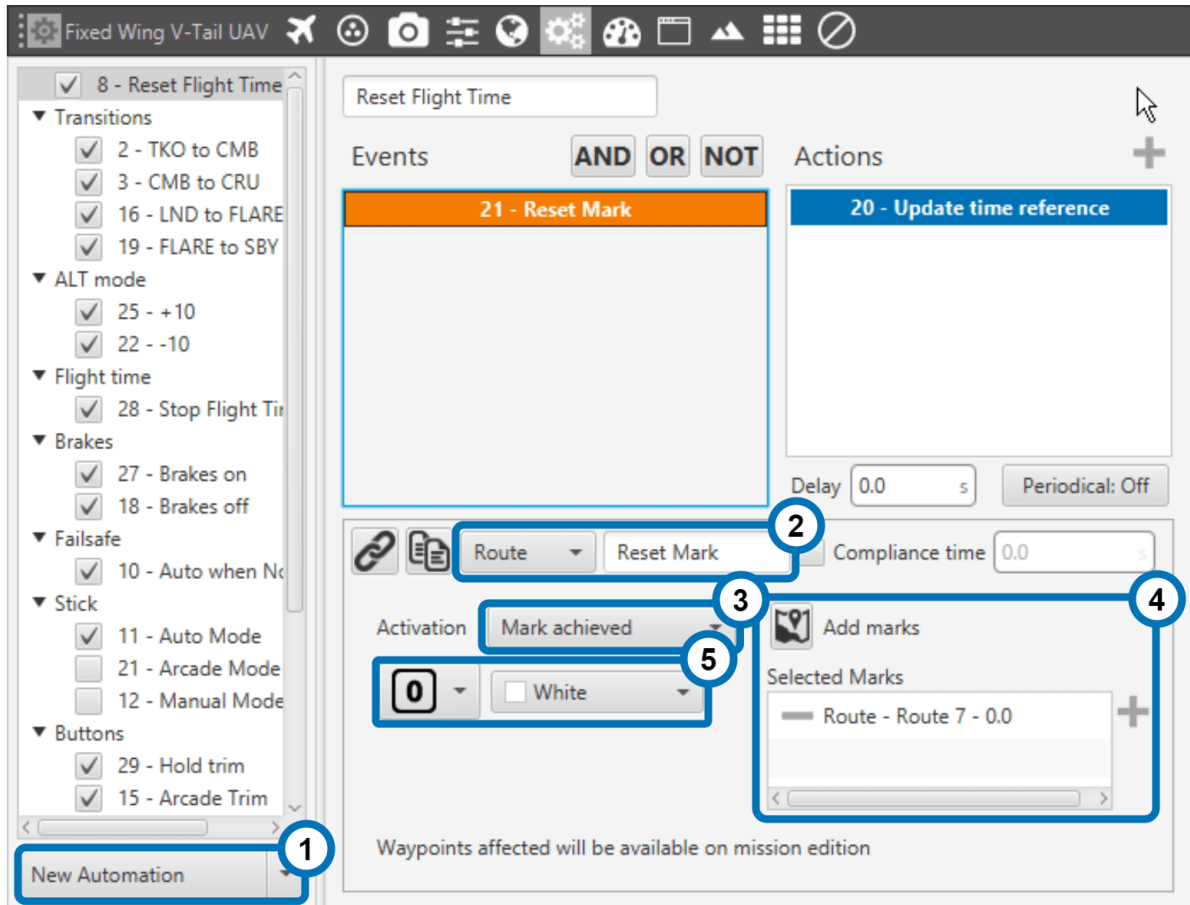
The screenshot shows the mission editor interface with the "Marks" tab selected. The table below represents the data shown in the Marks tab.

| Patches | | Patch | Type | Value |
|---------|----------|-------------------|------|-------|
| Route | Route 7 | Referred to start | 0.0 | |
| Route | Route 10 | Referred to start | 0.0 | |
| | | | | |
| | | | | |
| | | | | |

Marks List



8.2.7.1 Event creation and association

For this example we have created a new automation, Reset Flight Time, and created and associated an event to the mark in route 7 to trigger it:



Event creation and mark association

Clicking in Setup, in the subsection of automations, you can enter in the automations menu tab to link one event to one mark following the next steps:

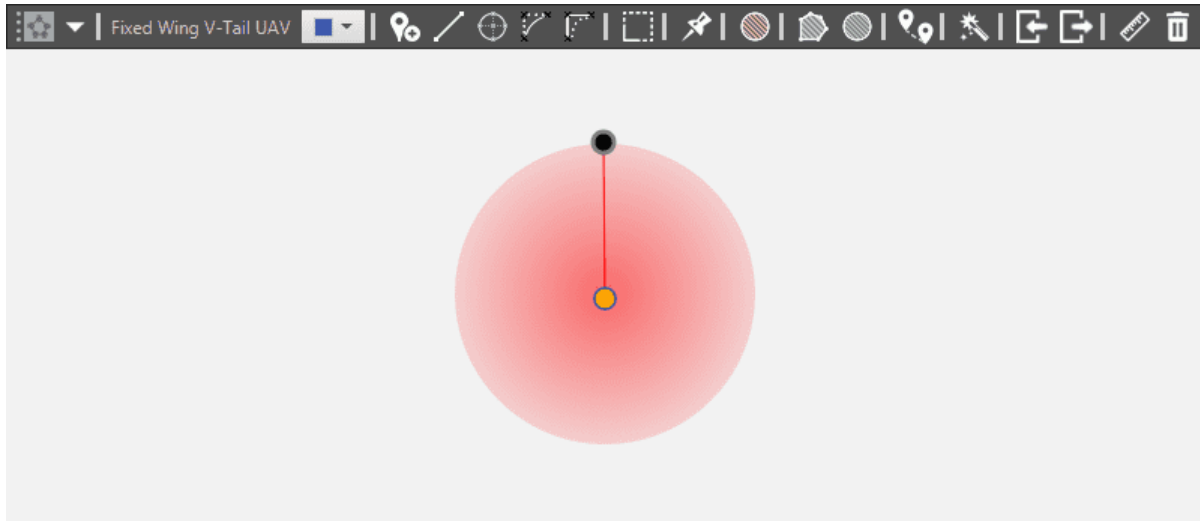
1. **New Automation.** The first step is to choose or create a new automation. In this example we have created a new one: Reset Flight Time. After its creation, a list with events will show up. We select “New Event”
2. **Route.** We have changed its name to “Reset Mark” and selected the “Route” type in the drop down list.
3. **Activation.** In the drop-down list menu, we select “Mark Achieved”.
4. **Add/Selected Marks.** Click on , or , to select a mark in the map. We’ve selected the mark in Route 7. The user can select multiple marks for the same event. All the marks associated to this event will appear in this list. This is useful if some action needs to be repeated several times during a mission, for example taking photos at the desired locations or dropping payloads.
5. **Icon and color.** The final step is changing the icon and color of the mark to remember easily the automation linked to the mark. We have selected a zero icon, as it can be seen in previous photos. This step is optional but highly recommendable.

Remember to save the changes to make them effective.

8.2.8 Obstacles



Obstacle Tool



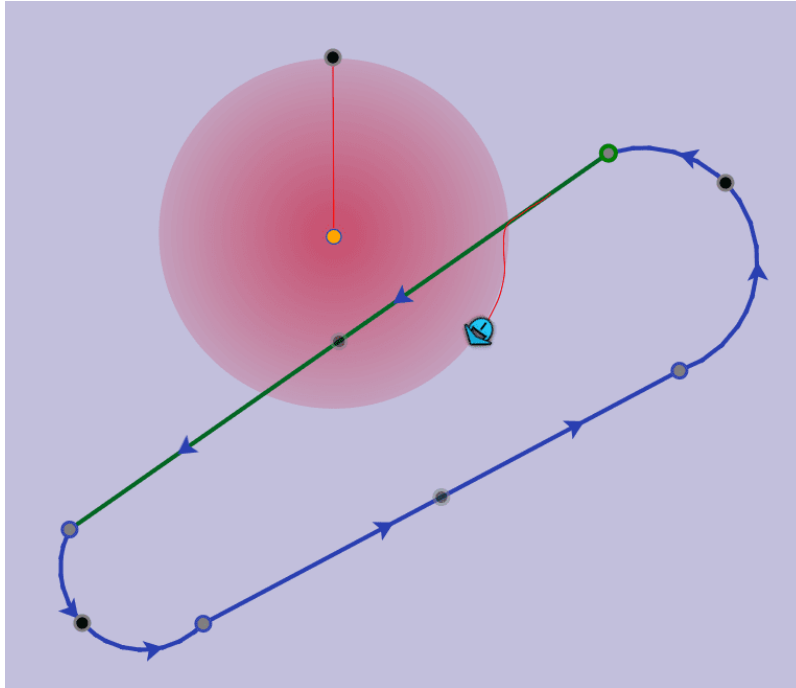
Obstacle

This tool permits to set an exclusion area on the map that can not be crossed by the RPAS. To change the area position, drag its central waypoint to the desired position. The radius can be extended or contracted directly by dragging the point in its circumference. It is also possible to set the coordinates of the central waypoint by double-clicking on it.

Main functions of the tool:

- Avoid collisions with obstacles as for example buildings, trees or antenna towers.
- Avoid flying restricted access areas.

This area is a repulsion potential field for the aircraft, so depending on the moving speed, the plane will cross the black line (high speed) or it will remain completely out of the red zone.



Obstacle avoidance

What can happen with high speed aircrafts (Airplanes) ? Its possible that because their velocity, the RPAS will sign in the obstacle area, but immediately it will apply the corrections to leave of the obstacle area and return to the path. The solutions for this problem its configure a bigger obstacle area in order to avoid the physical obstacle

What can happen with mow speed aircrafts (Multirotor) ? It can happen that a multirotor enter in a obstacle area (staying very near of its center), in this moment *Ground Speed Vector* and *Field Repulsion Vector* have the same direction but reverse sign. This phenomenon causes a conflict and until the directions are sufficiently different to allow the multirotor moving, it will stay in an indecision situation. To solve this problem you can configure the obstacle area letting the obstacle center away from the route line.

There are some options available for each obstacle. To access them right-click on the center waypoint of the obstacle.

- **Set 3D:** this option enables the aircraft to avoid an obstacle passing over it, taking account the height of the obstacle. The height can be edited as any waypoint, to access into the configuration menu double-click on it.
- **Set 2D:** this option only takes account 2 dimensions for the obstacle. So, the platform won't be able to avoid the obstacle passing over it
- **Remove:** deletes the selected obstacle.

8.2.9 New Polygon



New Polygon Tool

This tool is used to determinate an area in which an action is wanted to be performed. The way of working is very similar to Event Mark.

New Polygon

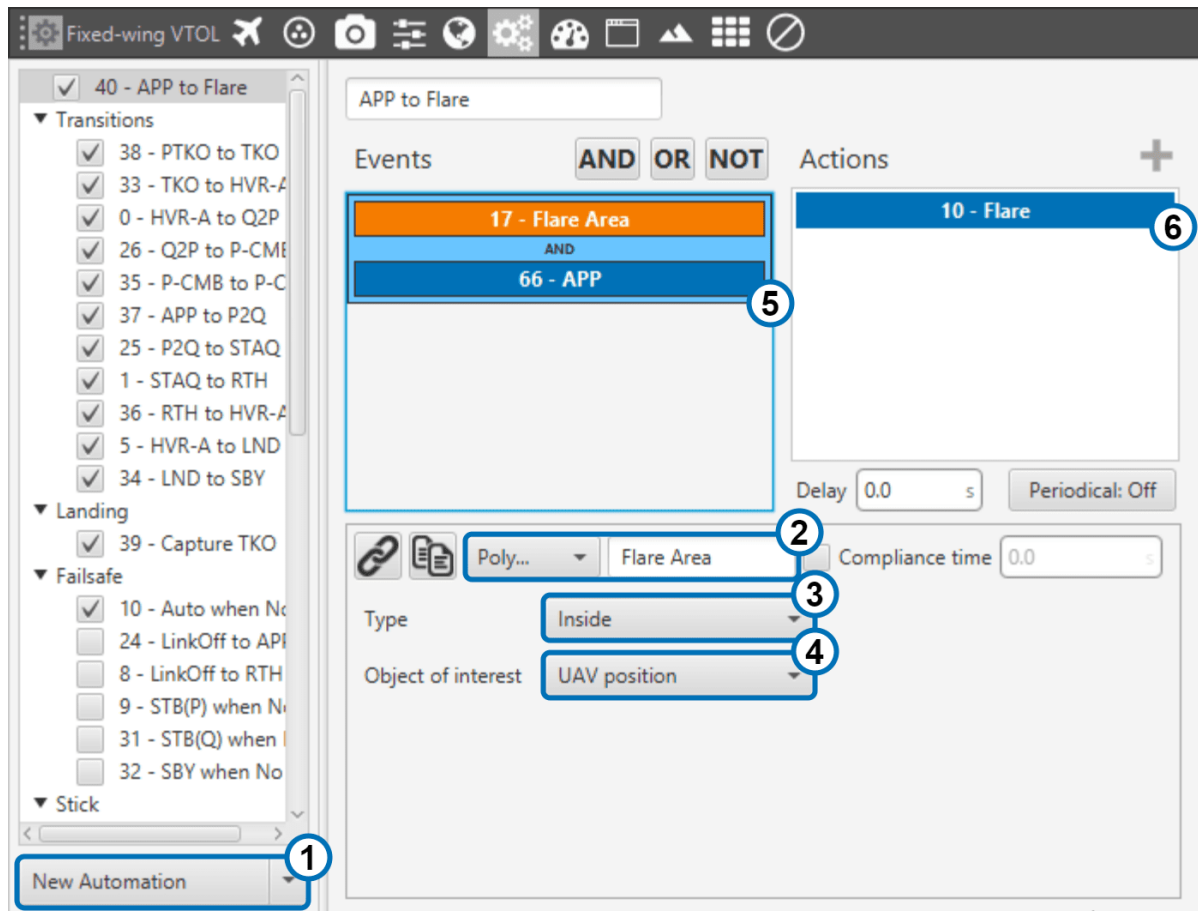
When the aircraft enters or leaves the polygon an event may be triggered and it can be used to start an automation.

To easy understand this tool, lets do an example with the action *Landing to flare*.

Note: *Landing to flare* is a maneuver made by an aircraft just before the touchdown. It Consists on a rise of the nose to decelerate the descent rate and set a proper attitude before touching the ground.

Following the same steps as in the Event Marks, we have created a new automation, “APP to Flare”, and a new event, “Flare Area”. Its purpose is to automatically change from the Approach maneuver (APP) to Flare before touchdown. To avoid problems, we have added as an additional condition that the plane should be in the Approach phase to trigger this automation:

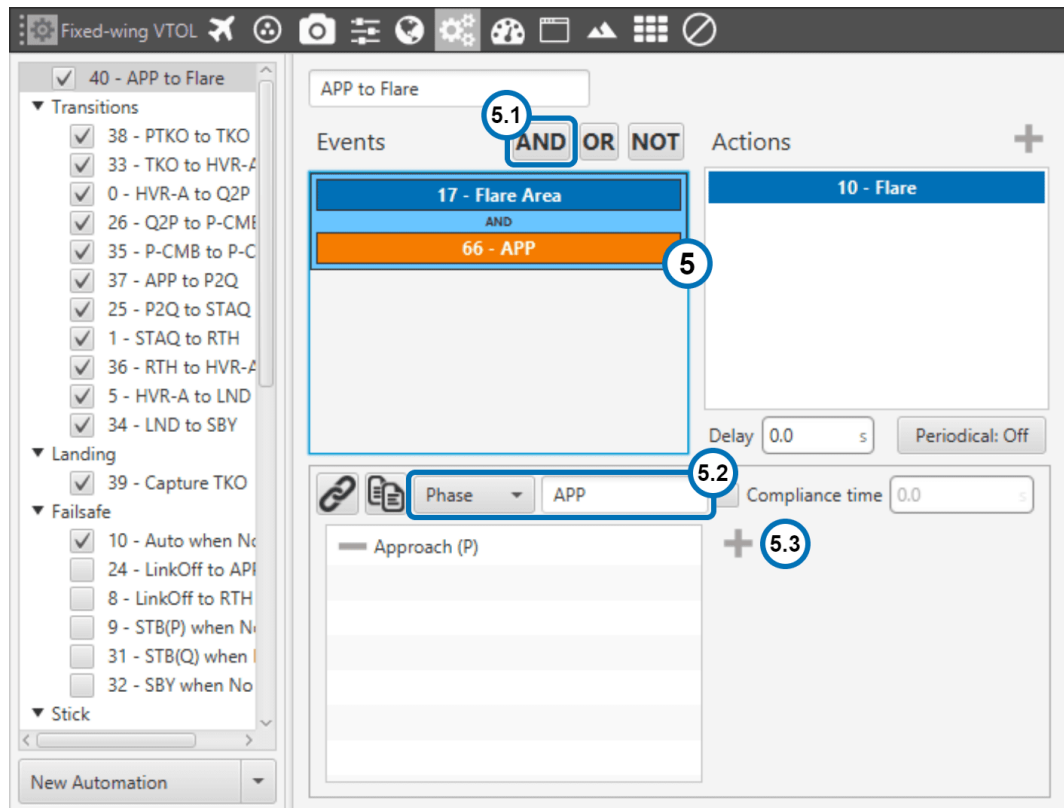
The steps followed to configure this automations were:



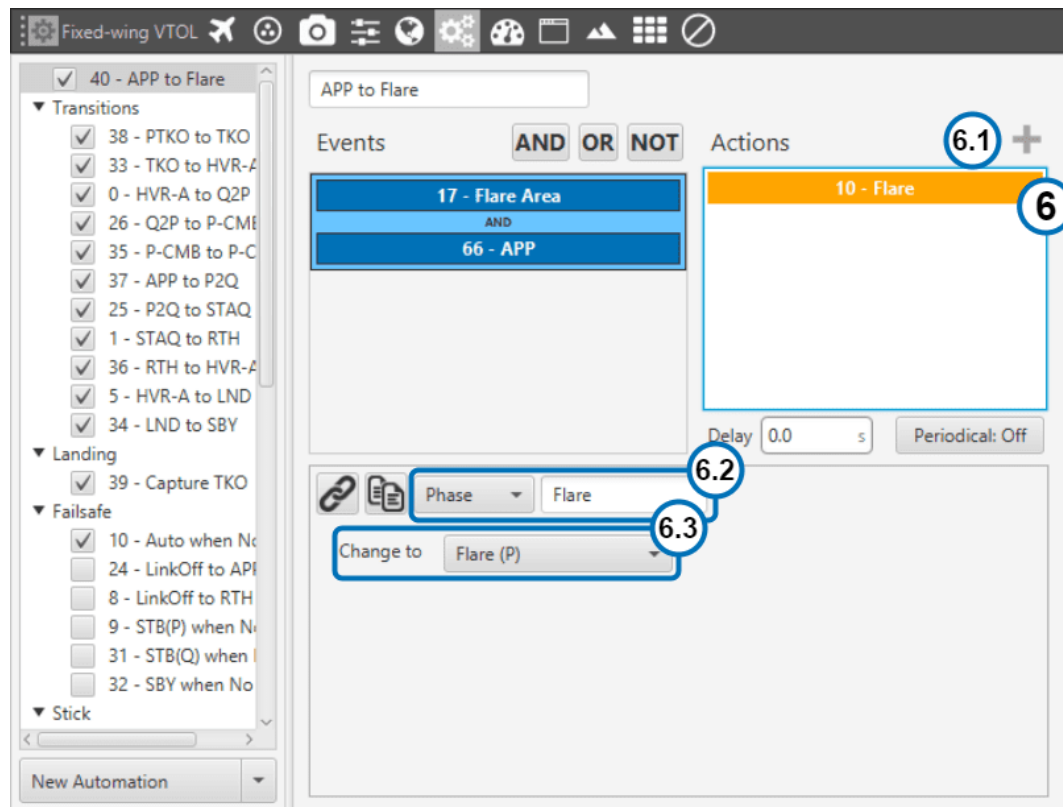
Flare Event Example

1. **New Automation.** First, create a new automation and change its name to “APP to flare”.
2. **Event Type.** Then, select the “Polygon” event type in the drop-down list and change its name to “Flare Area”.
3. **Type.** The Polygon event type can be defined as:
 - **Inside polygon.** The event will be carried out when the aircraft enters in the polygon.
 - **Outside polygon.** The event will be carried out when the aircraft leaves the polygon.
 Select “Inside”.
4. **Object of interest:** UAV position.

5. **Extra condition.** As previously said, to avoid problems another condition has to be met to trigger the automation. The plane must be in the Approach phase:



1. **Add Event.** Click on the “AND” button. An Event list Menu will pop-up, select a “Create Event”.
 2. **Event Type.** Select “Phase” in the drop-down list. Change its name to “APP” (approach).
 3. **Add Phase.** Click on the cross at the right of the list, search the “Approach” phase (the phase would have been previously created in the Control tab of the Setup menu. Check [Phases](#) for more information).
6. **New Action.** To end this automation, add a new action:



1. **Add Action.** Pressing the cross icon and “Create New” phase (if it doesn’t already exists).
2. **Action Type.** Select “Phase” in the dropdown list.
3. **Change to.** And select the “Flare” phase. This phase would have been previously created in the Phase menu.
7. **Link Polygon to event.** Finally, after the event creation, press the save button. Then, right click on the polygon and choose *Events > Name of the event*, to link the polygon to the new event.



8.2.10 Circular Area



Circular Area Tool

This tool is used to determinate an area in which an action is wanted to be performed. The way of working is the same as New polygon and very similar to Event Mark.

Circular Area

When the aircraft enters or leaves the circular area, an event is triggered and it can be used to start an automation.

Check Event Mark or New Polygon to learn how to create a Polygon Event.

To access the parameters, right button click on the circle:

- **Events.** Enter the Events tab to assign previously created events to the circular area.
- **Set Radius.** To accurately set the radius of the circle. You can also set the radius clicking and dragging the outer point.
- **Height Limits.** Yu can set the upper and lower heights of the circle volume. By default, circular areas have infinite lower and upper boundaries.



Circle Heights

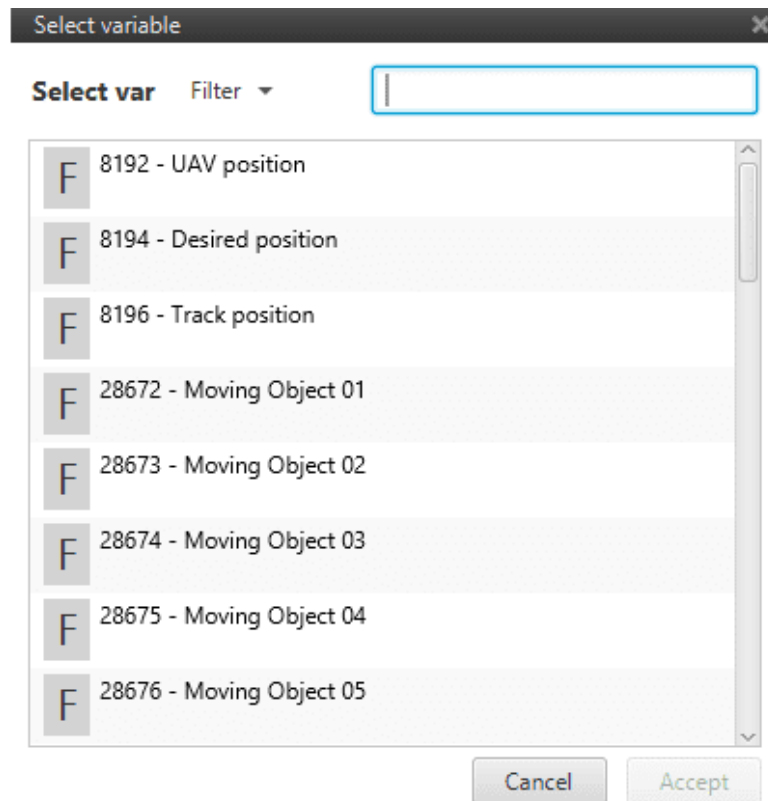
- **Remove.** Deletes the Circular Area.

8.2.11 References



References Tools Symbol

This tool has a wide range of capabilities, it is used to set references during the creation of a mission. Once selected, click on the map and a new window will be displayed to select the type of reference.



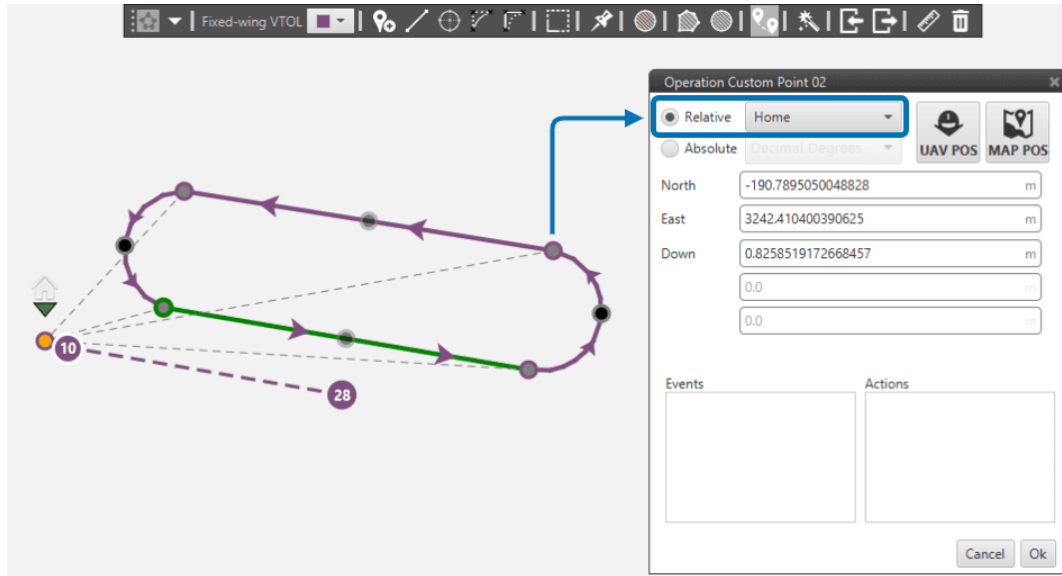
List of References

In this list is possible to select between 4 types of elements:

- **Home:** select a waypoint as Home. For example, it can be used to made a multicopter return to that point using the VTol guidance.
- **UAV position:** waypoint(s) can be linked to this reference and they will move relative to this reference.
- **Moving Object:** it works in the same way that UAV position, but now it is possible to use any moving object. They have to be configured and associated to a Moving Object in the sniffer. You can create until 32 moving objects.
- **Desired position:** it allows to define a fixed reference (it can be moved on the map, but not automatically). This reference can be used when there are a huge number of points. If users refer each point to a reference, user only has to move the reference to move all points or to perform a Phase (if it has the option) in a determined point.

How to add a reference:

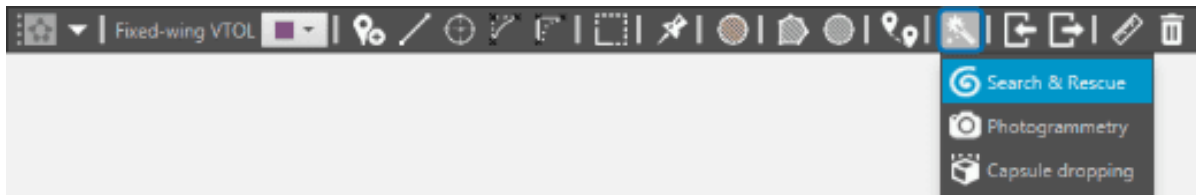
1. Select Reference on the Mission Toolbar and click on New Reference.
2. Click on the desired position on the map and select the type of element.
3. Double-click on a waypoint to acces its properties. Select the Relative drop-down list and then select the element of interest.



Waypoint Reference

4. Finally, set the relative position of the waypoint, this can be done introducing offsets or just moving the waypoint onto the desired place. A dashed line will appear to indicate that the waypoint is linked correctly.

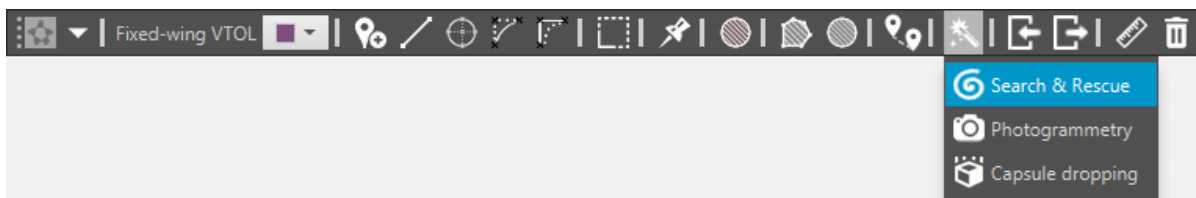
8.2.12 Mapping



Mapping Tool Symbol

The Mapping tab includes a series of missions automatically generated from a set of parameters introduced by the user. So far, Veronte Pipe includes the following mission wizards:

8.2.12.1 Search & Rescue



Search & Rescue

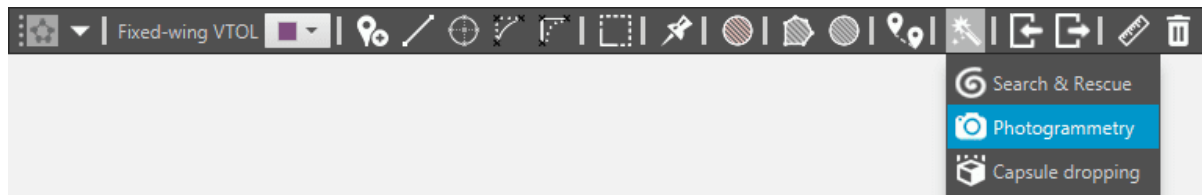
Search & Rescue mission draws a spiral that can cover the target area. The spiral can be modified using the two black-gray points or opening the Search & Rescue Menu (double click on the route waypoints) and changing parameters:

- **Radius:** spiral maximum radius.
- **Rounds:** the number of spiral rounds.

- **Altitude:** altitude of the waypoints. It can be expressed in AGL, MSL or WGS84.
- **Angle:** inclination angle of the spiral axes on the map.
- **Length:** length of the entire route.
- **Rotation:** you can choose between clock-wise or counter clock-wise.

Search & Rescue Example

8.2.12.2 Photogrammetry



Photogrammetry

This tool allows users to draw a polygon on the map and configure camera parameters to automatically generate a mapping mission. Photogrammetry can be set from the mission tab by clicking on the Mapping icon and selecting photogrammetry from the list.

Photogrammetry Example

The process to configure a *Photogrammetry* mission:

1. **Draw the polygon** in which the drone will cover flying. If no camera is configured, a new window will appear when you close the polygon.
2. **Camera specifications.** Define the camera (or select it from the database included in Veronte Pipe by clicking on the hand lens icon). Veronte can stream the video signal selecting the Streaming option. If the camera is associated with a gimbal, it can also be defined here.

Create camera

Default camera

Width Sensor: 36 mm

Height Sensor: 24 mm

Width resolution: 5784

Height resolution: 3856

Focal length: 35 mm

Camera cache: 1000.0 ms

☐ Streaming ☐ Cam associated with gimbal

☐ URL

☐ Device

Network caching

IP/port

☒ Photogrammetry

Action 0

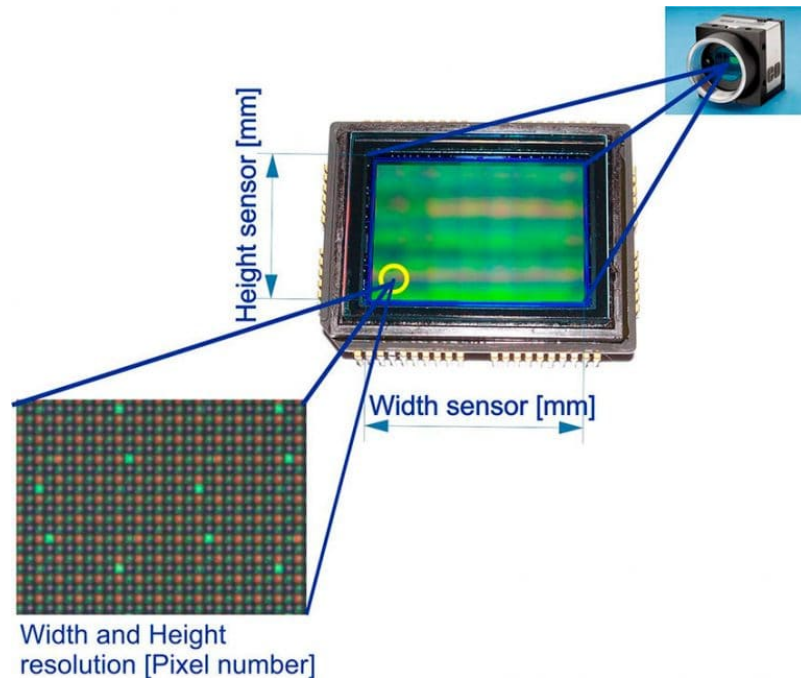
Add Action

Accept

Camera Configuration

Five parameters appear below the camera name:

- Width Sensor in millimeters.
- Height Sensor in millimeters.
- Width resolution in pixels.
- Height resolution in pixels.
- Focal length.



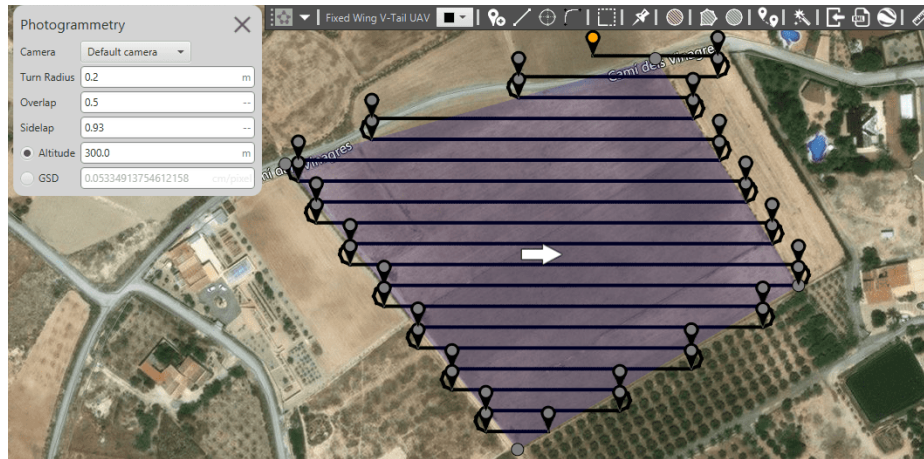
Sensor Parameters

Photogrammetry actions can be also added here.

3. **Photogrammetry configuration.** When your camera is created, it is time to configure your Photogrammetry process.

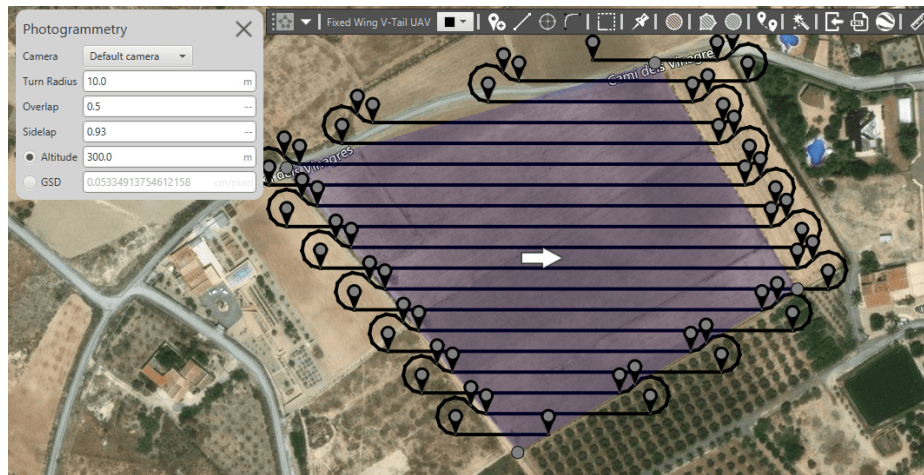
Photogrammetry Configuration

- **Camera.** Select which camera you are using in the process.
- **Turn radius.** radius of the route turns. Depending on this value and the distance between passes, there are three cases for this part of the route (the radius has to be established according to the minimum reachable by the platform):
 - **Radius 0**, there are not curves between passes, but straight lines. This option is used with multicopters, which are able to perform this kind of paths.
 - **Radius is smaller than half the distance between passes ($R < d/2$).** Pipe generates a semicircle with the diameter equal to the distance between parallel tracks.



Smaller Radius

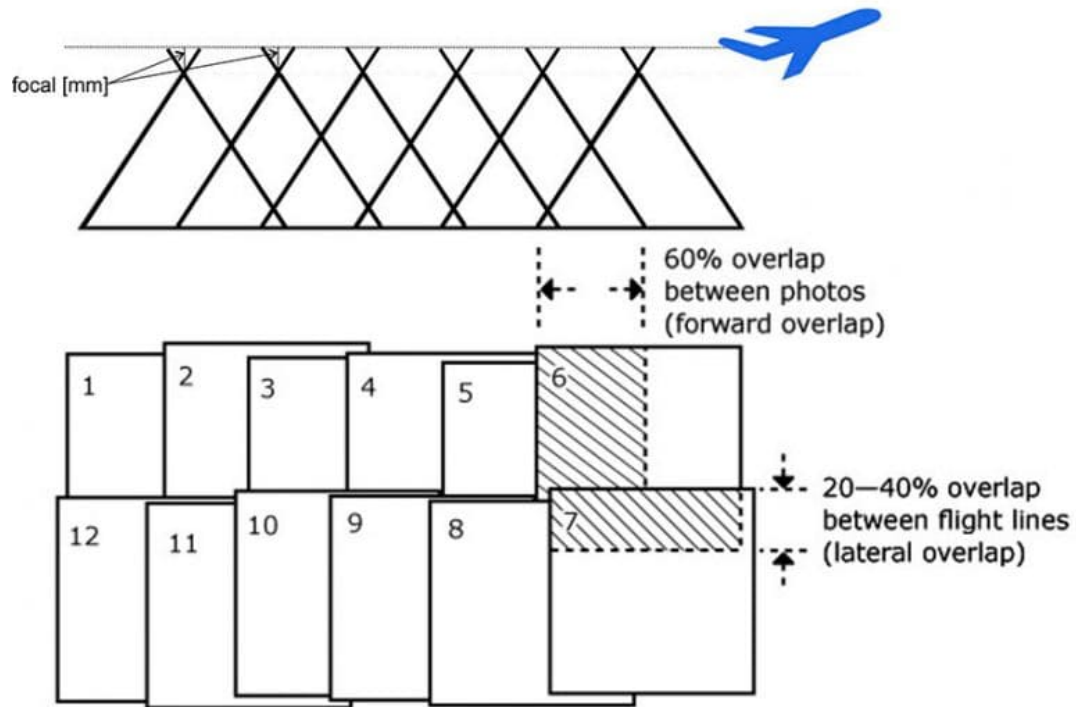
- **Radius bigger than half the distance between phases ($R > d/2$).** The path between straight lines is made up of two curves and a line.



Bigger Radius

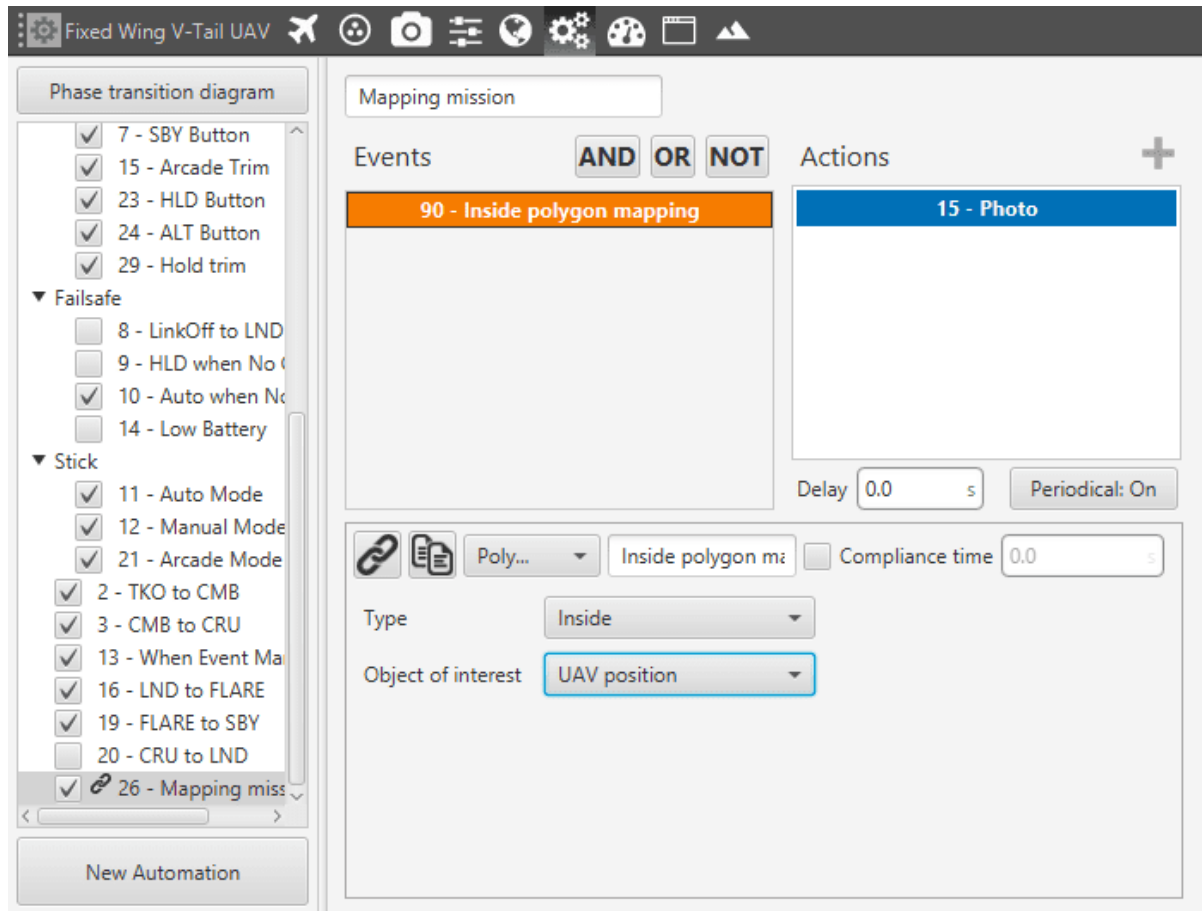
- **Overlap.** Desired overlap of the photos in the route direction. Its effect will be evident in the photos period changes.
- **Sidelap.** Desired photos overlap in the different passes. Increasing this value will reduce the distance between the passes.
- **Altitude.** Waypoints altitude in AGL.
- **GSD.** Ground Sampling Distance in photogrammetry is the distance between two consecutive pixel centres measured on the ground.
- **Main Direction of the mission** In the mission creation, at the centre of the created polygon, an steerable white arrow is present. When the arrow is turned, the mission will turn its main direction consequently. this is useful to avoid excessive wind exposition, for example.

The following image represents the mapping configuration variables in Veronte Pipe:



Overlap and Sidelap

4. When defining a Photogrammetry mission **an automation will be created automatically**. To complete the mission, in the lower part of the previous menu the option Photogrammetry has to be selected and the user must define the action that will be carried out, in this case taking a photo. The action (usually Output) has to be defined according to the hardware configuration on board. Once completed all the configuration, the automation will be created, by default its name is Mapping mission as showed in the two pictures below.

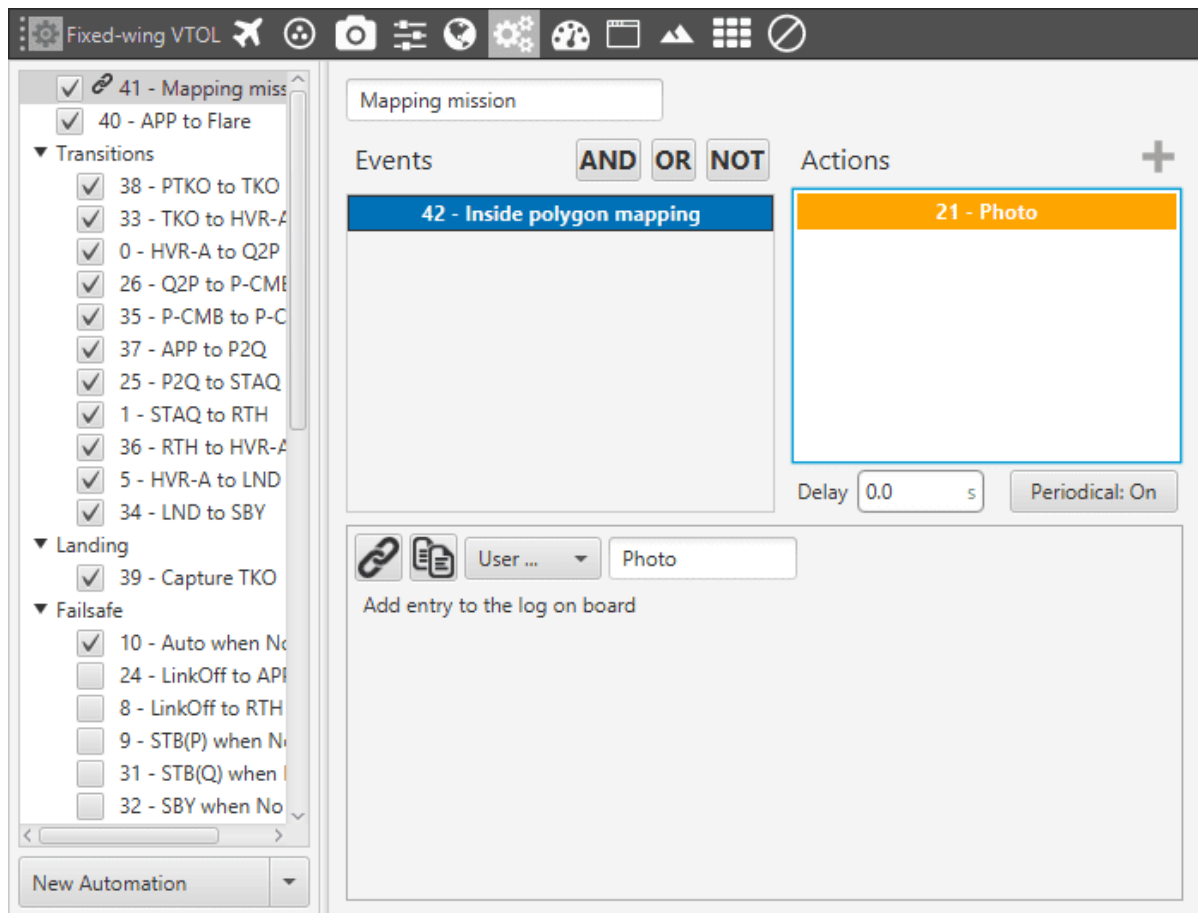


Mapping Automation Event

This automation tool offers you define automation as:

- **Inside area:** the event will be carried out when the aircraft enter in the area.
- **Outside area:** the event will be carried out when the aircraft leave the area.

Object of interest defines the object that have to enter or leave the area.



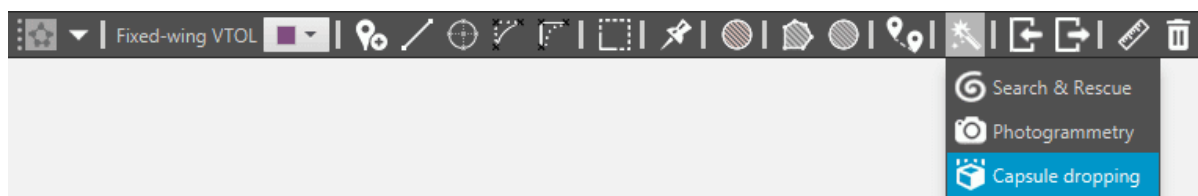
Mapping Automation Action

The action is carried out periodically in order to cover all the required area. Hence, the Period is computed in accordance with this and the camera specifications.

The configuration of the Photogrammetry mission is now complete. If the user wants to modify it, it can be done by changing the automation, the mission or the camera settings. The created camera can be found in the Devices panel as showed in the following figure.

It is also possible to configure a camera previously and select it after during the mission creation. How to add a new camera and configure it, is explained in [Camera](#) section.

8.2.12.3 Capsule Dropping



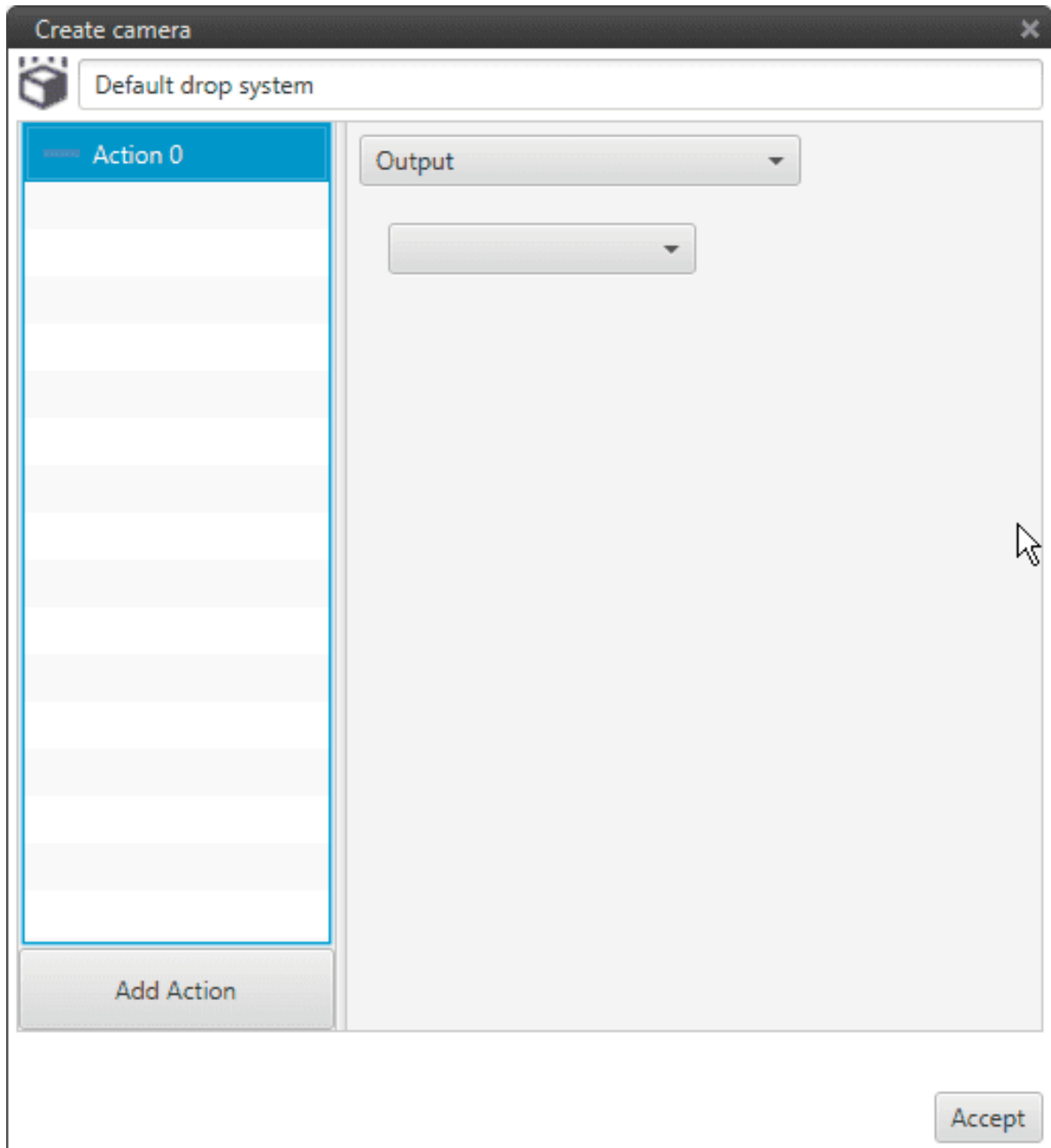
Capsule Dropping

This tool can be used when the user wants to create automatically a mission which has the scope of periodically dropping payload when the platform is following the route.

To do this, open the Capsule Dropping tool from the Mission toolbar. The configuration of this kind of mission is exactly the same of the Photogrammetry but menu and automation are different.

The process to configure a *Capsule Dropping* mission:

1. Clicking on the map, draw the polygon which indicates the zone the drone will cover with its flight. When you close the polygon, a new window will display.

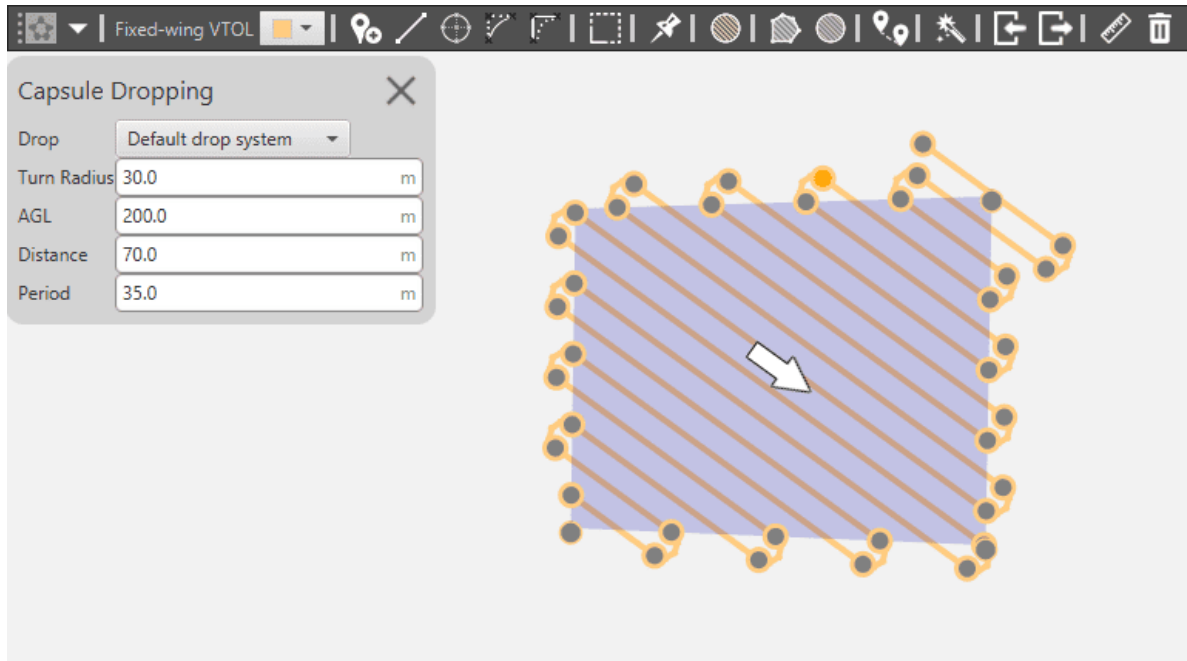


Capsule Dropping Configuration

2. When your camera is created, it is time to configure the *Capsule Dropping* process.

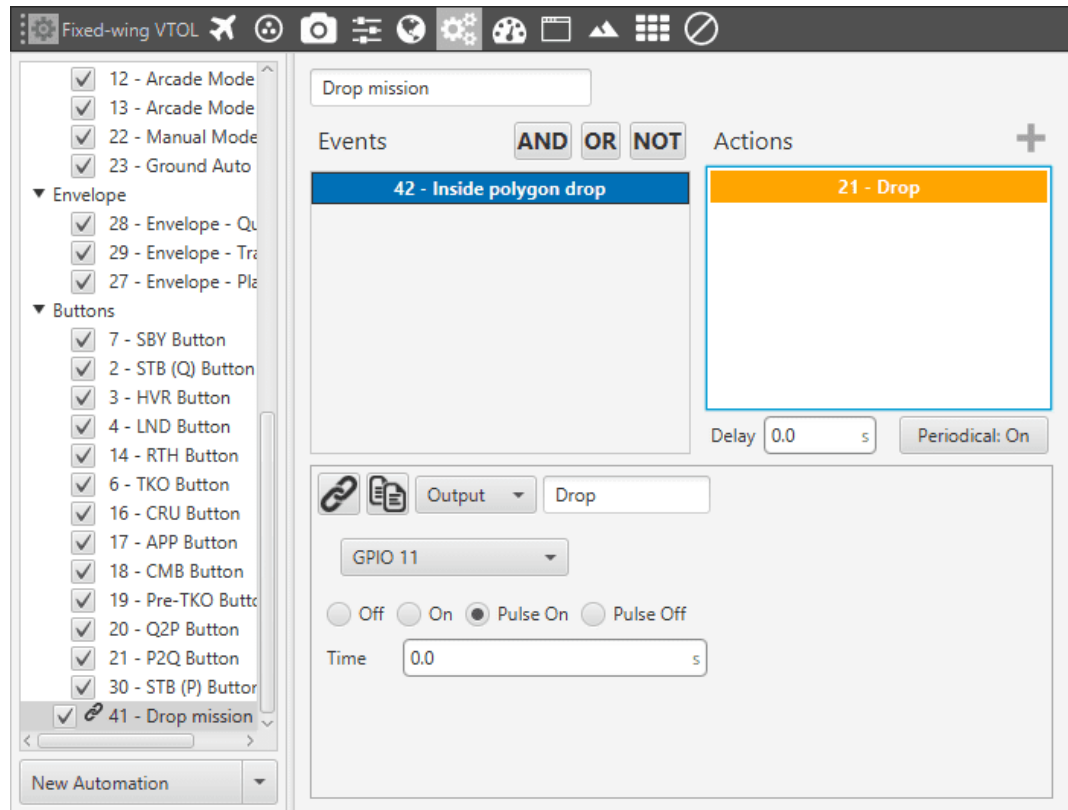
The routes can be modified by changing the parameters in the menu which shows up when a waypoint of the mapping mission is clicked. The following parameters can be set:

- **Drop:** the Dropping system can be configured in the tool menu and switched from this list.
- **Turn Radius:** radius of the route turns. In this case, the default configuration has the turn radius set to 0, but the effect of this parameter on the curve form is the same as in the case of photogrammetry mission.
- **AGL:** waypoints Above Ground Level altitude.
- **Distance:** distance between two parallel routes of the mission.
- **Period:** dropping period in [s].



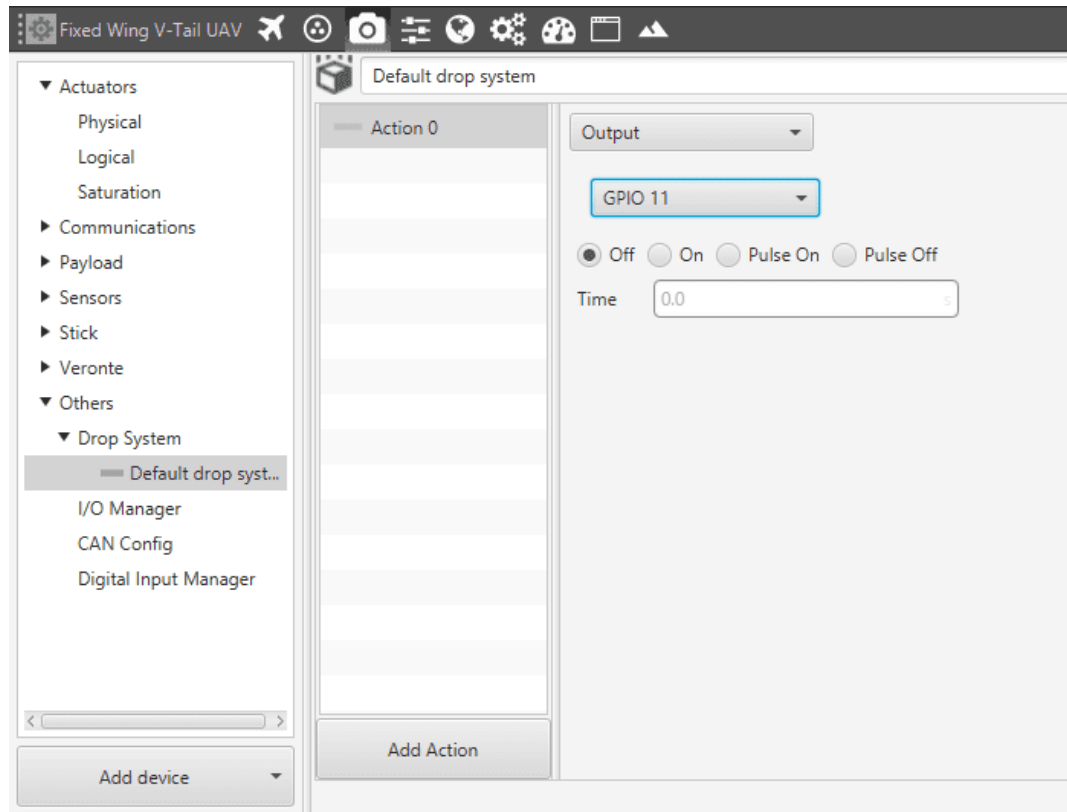
Capsule Dropping Configuration

3. As in the photogrammetry case, when the mission is defined, the previous menu will show up allowing the user to define the Capsule Dropping mission actions. The automation is then created as showed in the following image.



Capsule Dropping Automation Event

- The configuration of the Capsule Dropping mission is now complete. If the user wants to modify it, it can be done by changing the automation, the mission or the dropping system settings. The created dropping device can be found in the Devices panel as showed in the following figure.



Capsule Dropping device

It is possible to create a Drop System before the mission and select it after, this can be done at the Devices panel.

8.2.13 Import Route



Import Route Tool

Veronte Pipe has integrated the possibility to import routes from an external source software in NMEA RTE/WPL format, that is Route (RTE) or Waypoint (WPL) format.

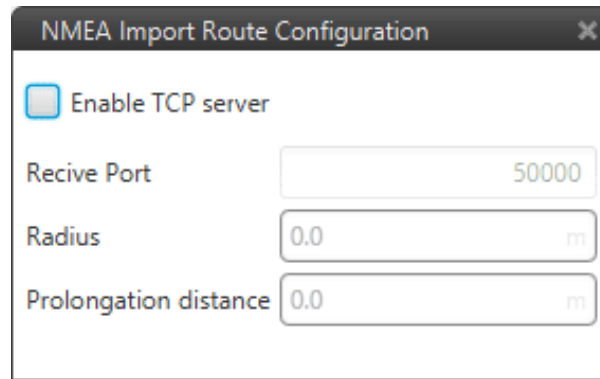
NMEA is a communication standard developed by the *National Marine Electronics Association*. For example, GPS receiver communication is defined within this specification. Most computer programs that provide real time position information understand and expect data to be in NMEA format. NMEA handles different standard format data sentences for multiple purposes including Routes definitions and waypoints positioning.

If you click in:



Import NMEA Protocol

The next window will appear:



NMEA Import Route Configuration

In this window you can configure the data reception via TCP connection. First, open a server socket. This socket must have the same address as the external software. Pipe will expect that the input format follows the MNEA RTE/WPL format to correctly display all the info.

8.2.14 Export Route



Export Route Tool

Pipe software allows to export the mission in two different ways: KML and Google Earth files.

8.2.14.1 KML



Export to KML

This tool facilitates to export the mission in KML format.

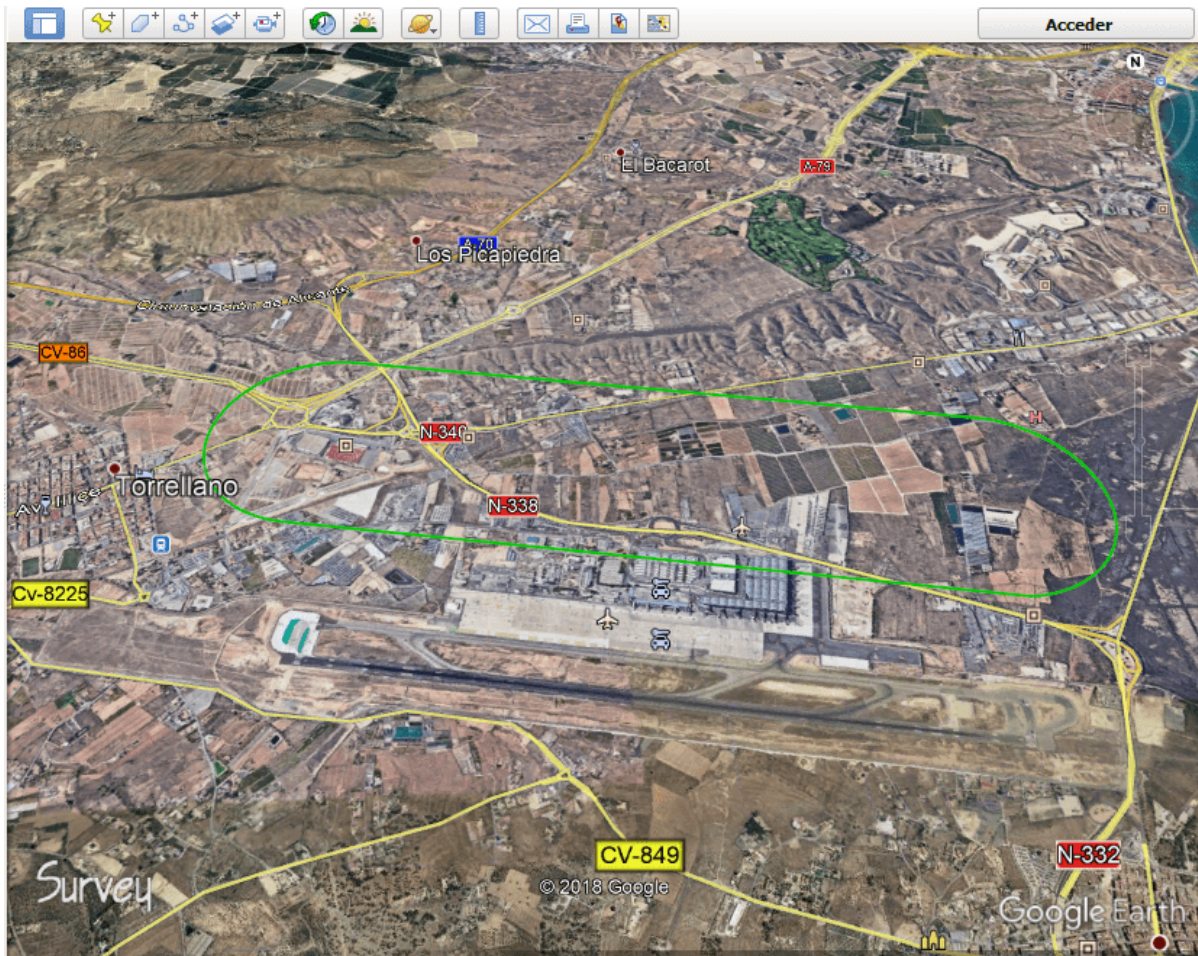
KML, *Keyhole Markup Language*, is a code language, based in XML, used to represent geographical data in three dimensions.

8.2.14.2 Google Earth



Google Earth Tool

This tool allows to export the mission to **Google Earth**, it can be very useful depending the objective of the mission.

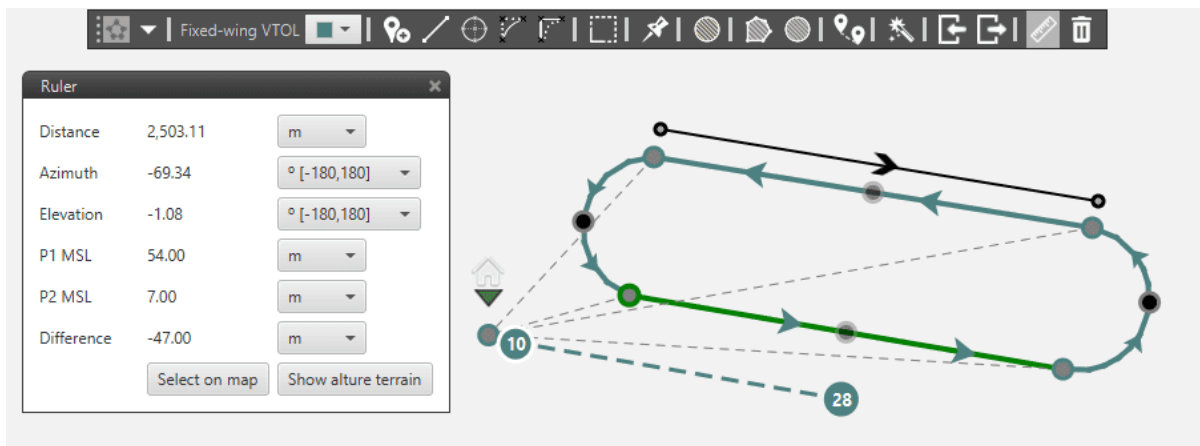


Veronte Mission in Google Earth

8.2.15 Ruler



Ruler



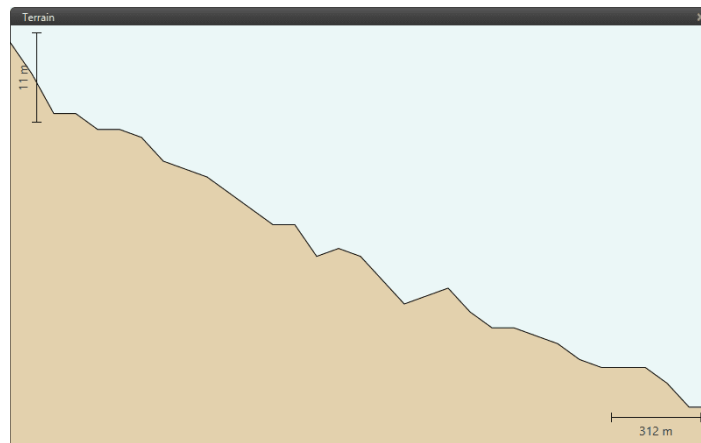
Ruler Example

Using the ruler, you can measure:

- **Distance:** distance between two points.
- **Azimuth:** angle of the segment with the North.
- **Elevation:** shows you the angle of the segment respect to a horizontal plane.
- **P1/P2 MSL:** altitude respect to the the *Main Sea Level* of the points P1 and P2.
- **Difference:** height difference of the points P1 and P2.

You can choose the units of the measures.

- **Select on Map.** Restarts the measurement.
- **Show alture terrain.** Shows the terrain profile between the ruler points



Terrain Profile

8.2.16 Remove unused points

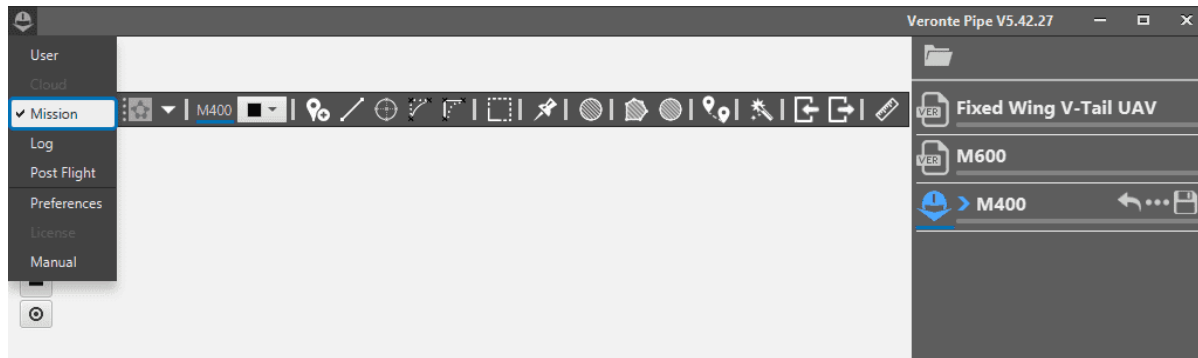


Remove Unused Points


This tool allows to clean the map of the mission of unused points. Those points can appear after paths modifications. This tool searches unconnected points with no dependencies and deletes them from the workspace.

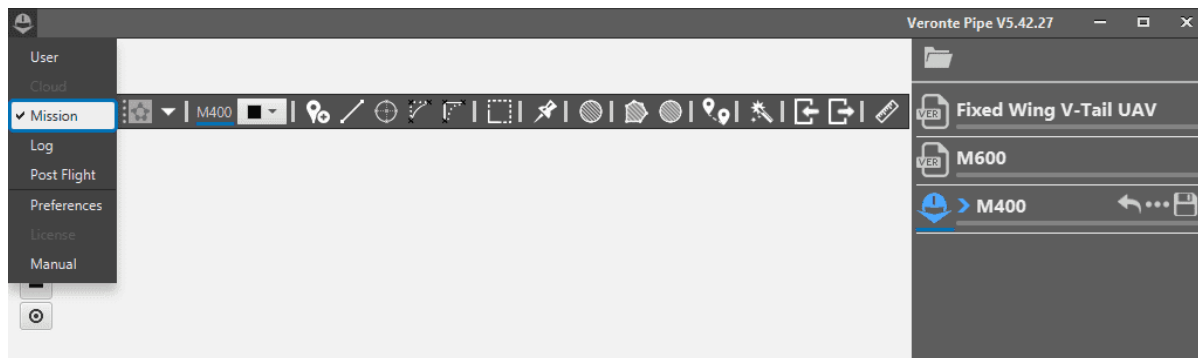
This section of the manual contains the information about the different tools available in Veronte Pipe to create the map elements used during the mission. Some of those are: waypoints, lines, circles and so on.

In order to activate the mission tools bar, a Veronte unit or a configuration has to be select in the right menu. Once selected, the toolbar will be activated with its corresponding name.



Mission Menu Access

The common way to work with Veronte is to create missions. Missions can be created and managed through the Mission menu. To access this menu, click on  at the upper left corner and select **Mission** in the pull-down menu. The Mission toolbar will appear as shown at the image below:



Mission toolbar menu access

The mission toolbar provides graphical tools to create the path that the aircraft follows while it is in **cruise phase**. Multiple missions can be created for different purposes. The missions generated are stored into the **Autopilot Air unit** and into the **Operation** folder of the PDI configuration files when exported.

Next, a detailed description of the toolbar is given. And an example of how to set up a mission will be given in the following section.

8.3 Toolbar



Mission toolbar

The controls found in the toolbar are detailed in the following table.

| Icon | Item | Description |
|---|----------------------|---|
|  | Open Detail | Displays Terrain and Marks configuration. |
|  | Colours | Line colour. |
|  | New Waypoint | Add new waypoint on click position. |
|  | Segment | Add a straight line. |
|  | New Orbit | Add a orbit. |
|  | Fly By | Tool to adapt straight path corners into turns. |
|  | Intersect Lines | Undoes the Fly-by command. |
|  | Multiple Choice | Selection tool. |
|  | Event Mark | Add a mark to trigger an event. |
|  | Obstacle | Add an obstacle signal the aircraft has to avoid. |
|  | New Polygon | Draw areas on the map for association with polygon events. |
|  | Circular Area | Draw a circular area on the map. For association with polygon events. |
|  | References | Create points of reference. |
|  | Mapping | Predefined tasks helper: Search & Rescue, Photogrametry,... |
|  | Import Route | Import NMEA route configuration. |
|  | Export Route | Export to a .kml or a .csv file the route created. |
|  | Ruler | Measure on map. |
|  | Remove Unused Points | Removes unused points |

OPERATION

9.1 Map

The map is always displayed on the background and provides visual information about the area.

If known, the position and direction of any connected units will also be plotted on the map automatically.

There are several provider and map type options. Choose the one that fits better the nature of the mission. It is also possible to use **Custom Maps**.

Veronte Pipe automatically downloads map information when needed. This information gets stored on the computer, and will then be accessible without the need of an internet connection.

Downloaded map tiles are stored under the following route: C:\Users\<Your User>\AppData\Roaming\VerontePipe\cache\map

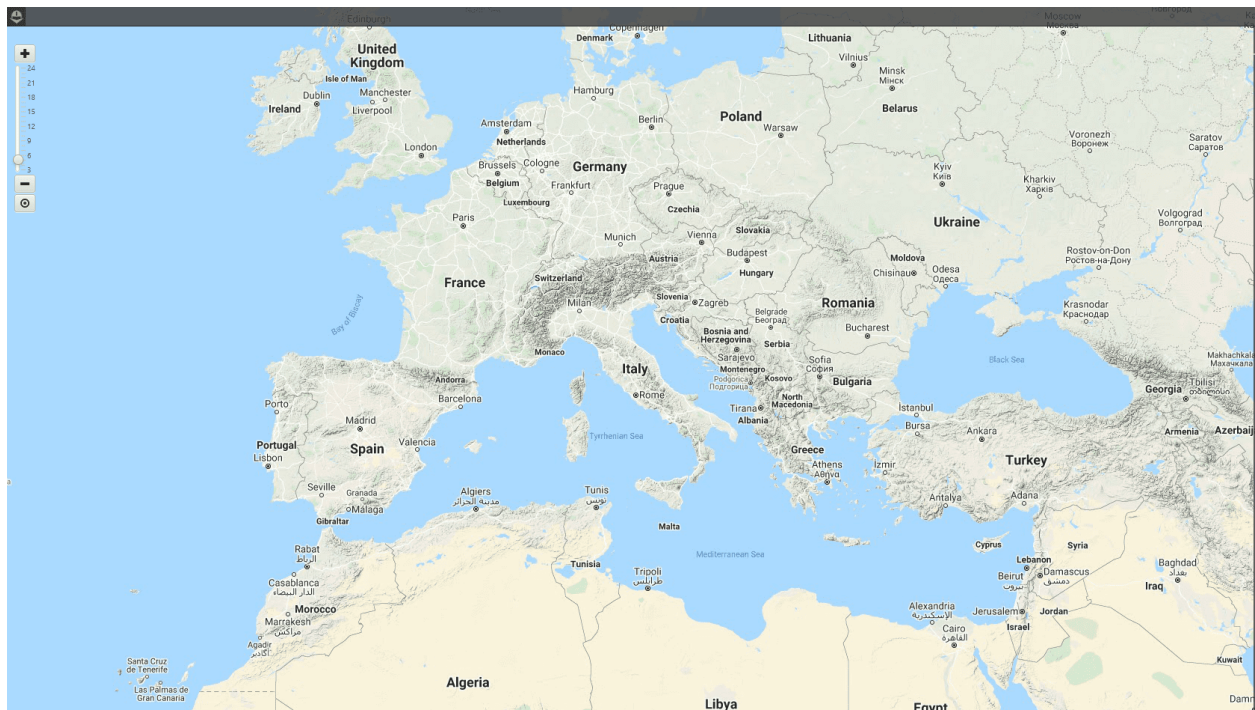
For more information on how to configure the different map options, please visit the [Workspace](#) section.

Warning: Veronte Pipe can only download map information using Internet access. If there is no internet access available during your operation, make sure to download any map tiles that may be needed preemptively.

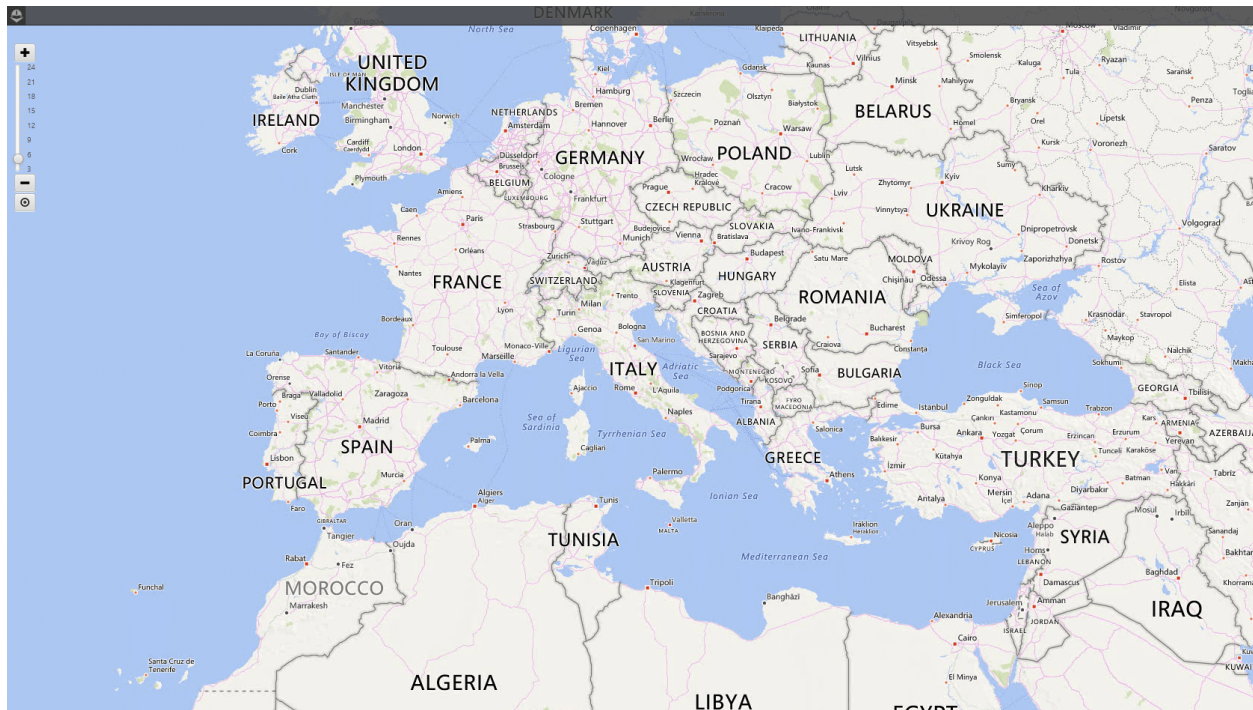
Each zoom level in Veronte Pipe counts as an independent map tile. When downloading map information, make sure that any relevant zoom levels are also downloaded.



Satellite Map



Terrain Map



Street Map

9.2 Telemetry

Telemetry involves any data that is received from the UAV in order to monitor the status of the mission.

Aside from position, which is always displayed, a typical operating workspace may include:



Example of telemetry setup with Veronte

1. **Attitude and heading:** Allows to know the status of the aircraft in the air. In case of External pilot takeover out of line-of-sight, it can work as a reference for flying if no other references are available.
2. **Battery/Fuel/Flight time:** Indication of the remaining energy resources. Indicates the operator the remaining operation time.
3. **GPS data:** Includes GPS status (GPS Ok/No OK), accuracy and satellite coverage.
4. **Wind information:** Wind direction and intensity.
5. **Speed information:** Airspeed and groundspeed.
6. **Alarms & Warnings:** Any alarms from the system that the operator should pay attention to.
7. **Altitude information:** Including different reference systems such as MSL and AGL.
8. **Actuator information:** Position of each of the system actuators.

Other relevant telemetry data could include:

- Link status
- Payload status
- Engine telemetry: RPM, temperature

Veronte Pipe workspace is fully configurable, and has a wide variety of widgets available to help the user build an operation screen that meets its requirements. For more information about how to build a workspace please visit the [Workspace](#) section.

Tip: It is recommended that the operating workspace includes only information relevant to the current mission, avoiding any irrelevant data on screen.

Warning: The fact that a certain variable is displayed on the workspace does not guarantee that the value is being received. Some widgets will warn you whenever the displayed variable is not being updated, but others may not.

Make sure to always verify that all relevant data is correctly included in the telemetry vector.

Different missions may require different workspaces. For instance, in the early stages of development, the most common type of mission is a Tuning mission, where different parameters of the system are adjusted in order to find the proper values or validate a new set of parameters.

In this type of mission, it is common to have more crowded workspaces like the one displayed below:



Example of tuning setup with Veronte

Tip: Make sure to take advantage of the workspace CASES and GROUPS to hide an show certain widgets only when they are relevant. With the CASE feature, this can even be done automatically!

9.3 Operation Parameters

Operation parameters are configurable values, positions and directions that can vary depending on the mission.

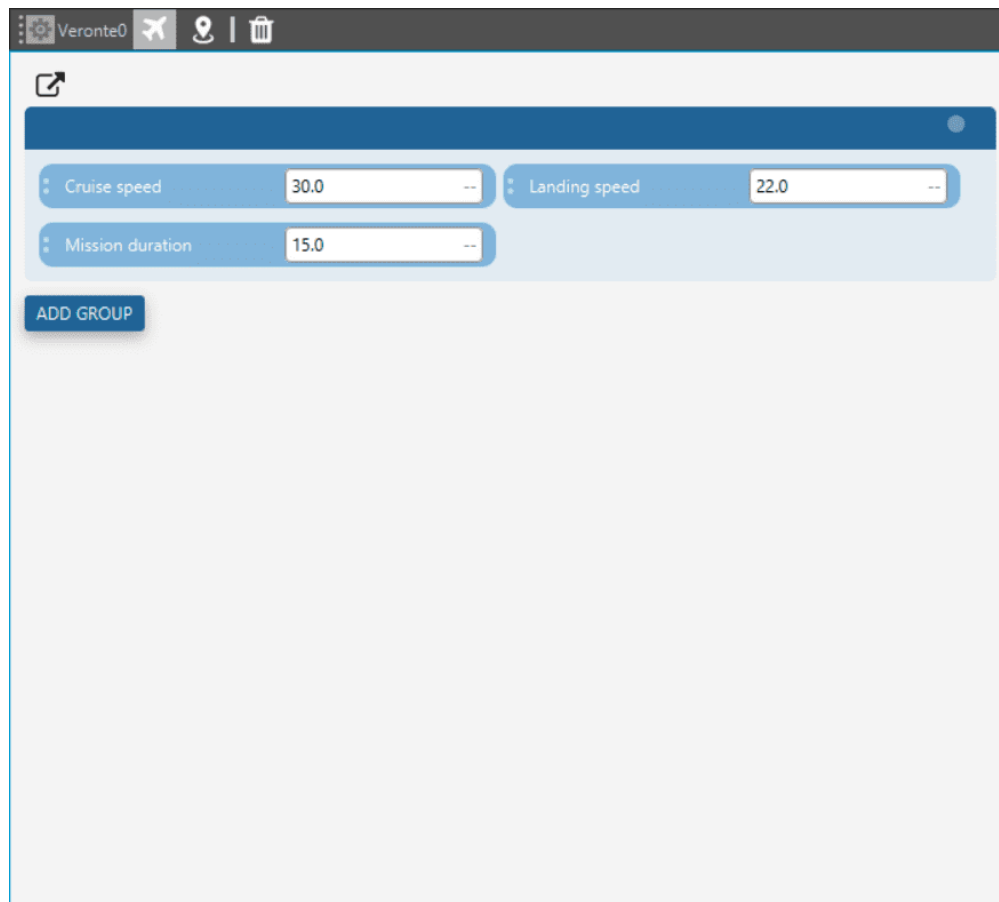
Examples of operation parameters can be:

- Mission duration
- Cruise speed
- Flight level

- **Takeoff and landing direction**
- **Home point**
- **Start of route**

The main advantage of Operation parameters is that there is no need to access **Veronte configuration** in order to change them. This way, the Operator is able to modify certain parameters without the need of having access to the whole configuration.


Defined operation parameters will be saved into Veronte configuration, and will be kept even if the system is rebooted.

The screenshot shows a web interface for 'Veronte0'. At the top, there is a dark header bar with icons for settings, a plane, a location pin, and a trash can. Below the header, there is a light blue sidebar with a share icon. The main content area has a dark blue header bar. Below this, there are three input fields: 'Cruise speed' with a value of 30.0, 'Landing speed' with a value of 22.0, and 'Mission duration' with a value of 15.0. Each field has a small gear icon on the left and a '--' button on the right. Below these fields is a blue button labeled 'ADD GROUP'.

Operation Parameters Example

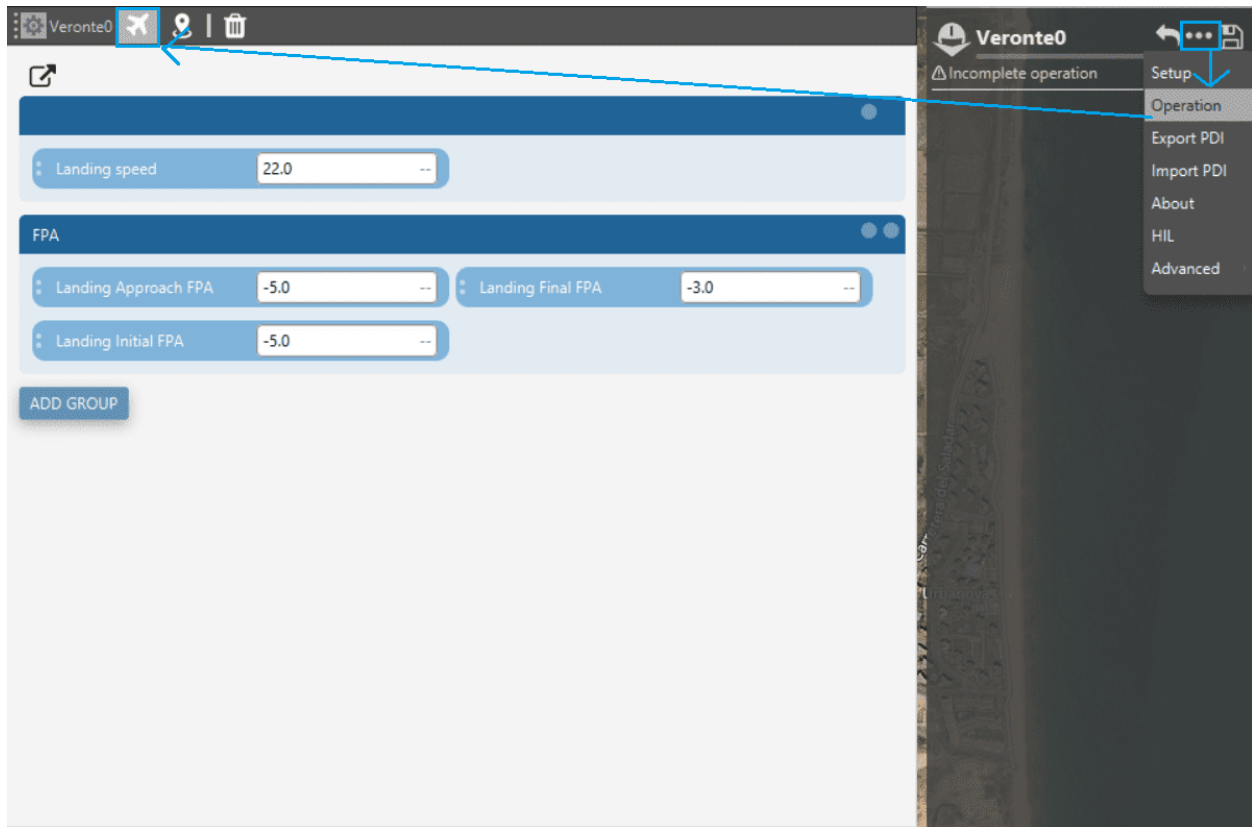
How to use operation parameters

Operation parameters are created whenever one of the following actions is performed while building a Veronte configuration:


- An **Operation Guidance** or **User direction** variable is used at any point within the configuration (e.g. in control clicking  on or in Blocks menu)
- A **Waypoint** on the map is given a 'Name'.
- A specific **Waypoint** is assigned to the **Guidance** of one of the **Phases**.

Whenever one of the above cases is true, a new field will appear in the **Operations menu**.

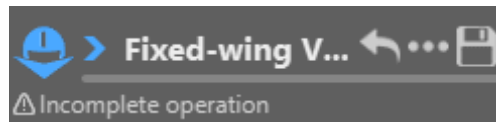
The Operations menu for each unit can be accessed from its respective dropdown list on the top-right corner of the screen:



Operation Menu

After that, the values for these parameters can be defined using this menu. Remember that, in order to apply the new parameters, it is necessary to save  the configuration.

If any of these parameters is not defined at any moment, the user will receive an **Operation Incomplete** warning message. Clicking this message will also open the Operation menu.







Parameter not defined warning

Warning: Although it is possible to modify Operation Parameters during the flight, this practice is not recommended.

Whenever changing values during an operation, make sure that no potential risk to flight safety is involved.

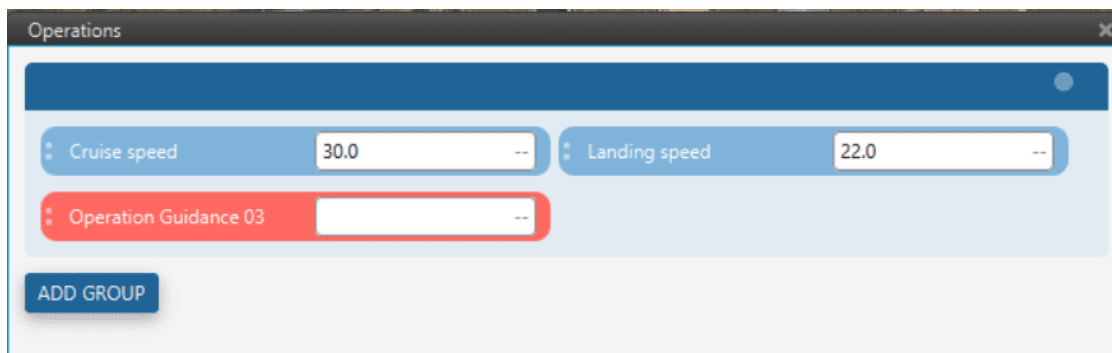
Example

Lets say it is required to define Flight path angles for the different Landing segments from the Operation menu. The **First step** should be going to the **Landing Phase Guidance**, and define these parameters as **Operation Guidance** variables.

| Flight path angle | |
|-------------------|---|
| Initial | Operation Guidance 04  |
| Loiter | Operation Guidance 05  |
| Aim | Operation Guidance 06  |
| DXY | 0.0 rad $[-\pi, \pi]$  |

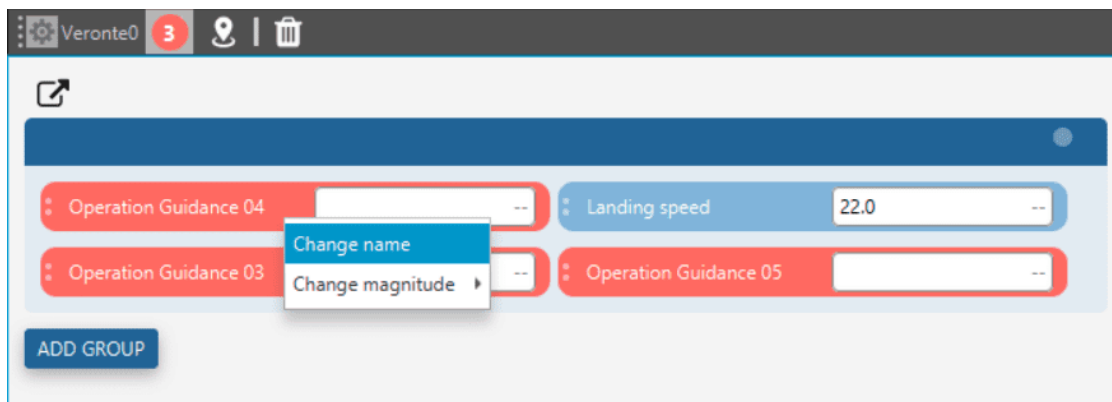
Example - Step 1

Now, the selected values will appear on the Operation menu.




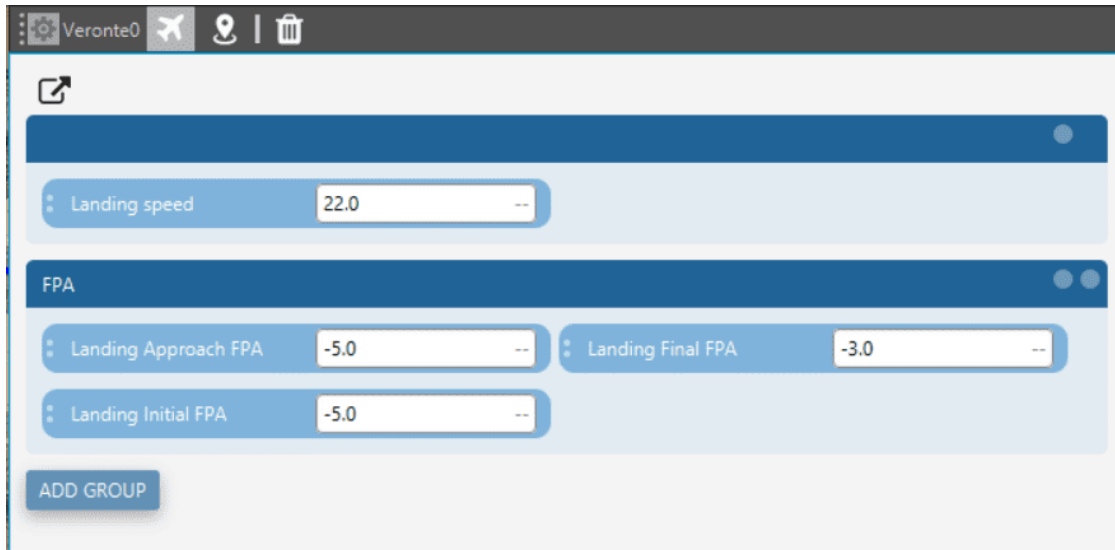
Example - Step 2

Operation Guidance variables can be directly renamed from the Operation Menu.



Example - Step 3

Finally, giving values to each of the parameters and saving  will apply the changes.



The screenshot shows the Veronte Autopilot configuration interface. At the top, there is a header bar with the text 'Veronte0' and three icons: a plane, a location pin, and a trash can. Below the header, there is a main content area with a light blue background. On the left side of this area, there is a vertical sidebar with a blue header and a list of items. The first item is 'Landing speed' with a value of '22.0'. Below it is a section titled 'FPA' with three items: 'Landing Approach FPA' with a value of '-5.0', 'Landing Final FPA' with a value of '-3.0', and 'Landing Initial FPA' with a value of '-5.0'. At the bottom of the sidebar, there is a button labeled 'ADD GROUP'. The main content area has a blue header and a light blue body. The 'Landing speed' item is highlighted in blue. The 'FPA' section is also highlighted in blue. The 'Landing Approach FPA' and 'Landing Final FPA' items are highlighted in blue. The 'Landing Initial FPA' item is highlighted in blue. The 'ADD GROUP' button is highlighted in blue.

Example - Step 4

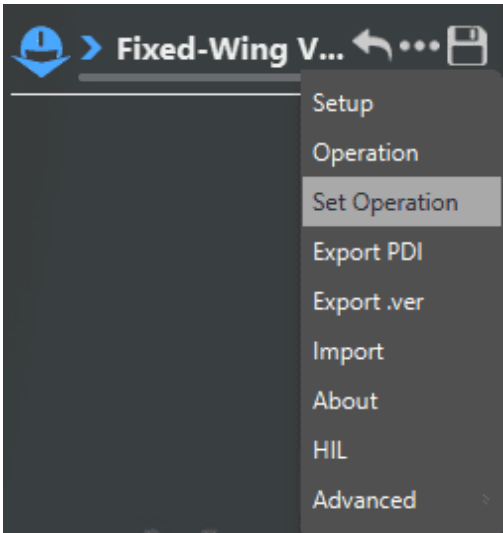
9.4 Mission

Even though it is recommended that the mission is already defined before the operation starts, it is possible to load a new mission or modify the current mission before and during the operation.

Within the Operation menu, the **Waypoint** tab can be accessed. In here it is possible to find a list of all elements that belong to the current mission, including **Waypoints**, **Polygons** and **Obstacles**.

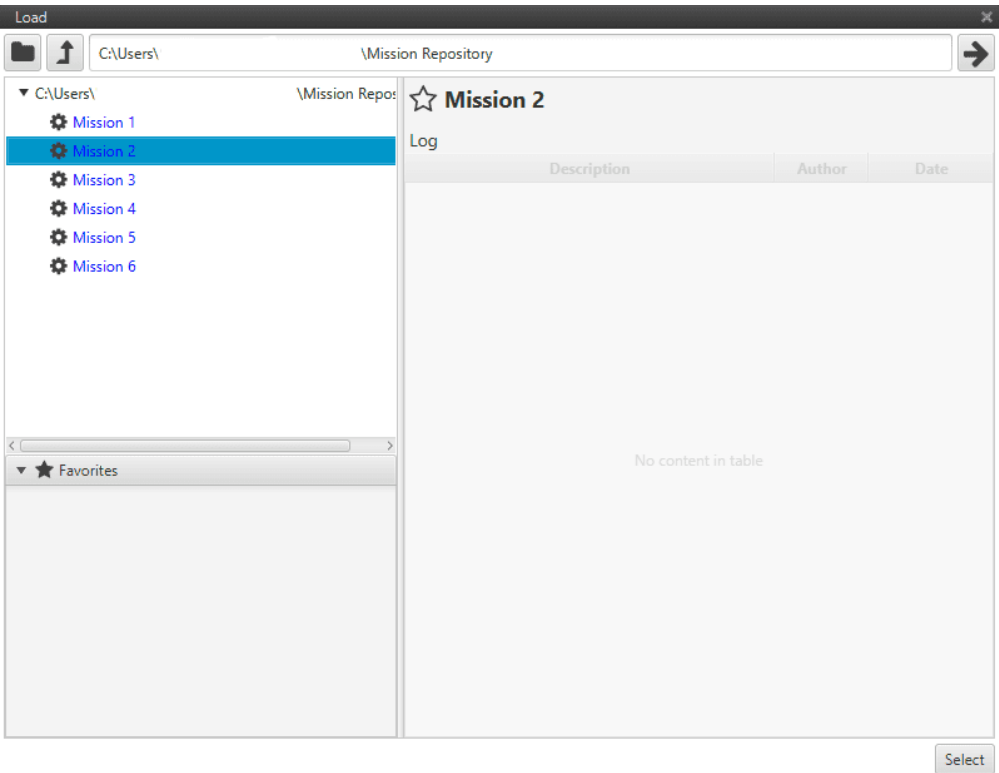
9.4.1 Loading a new mission

Using the feature **Set Operation** it is possible to replace the current mission by another one that was previously exported.




Set Operation

This allows the user to build a mission library that can then be used to load an operation from a set of pre-defined missions.



Operation Library

9.4.2 Modifying the current mission

The current mission can be modified at any point using the options described above, or by means of the Mission Building Tools *mission*, and then clicking on the Save  button.

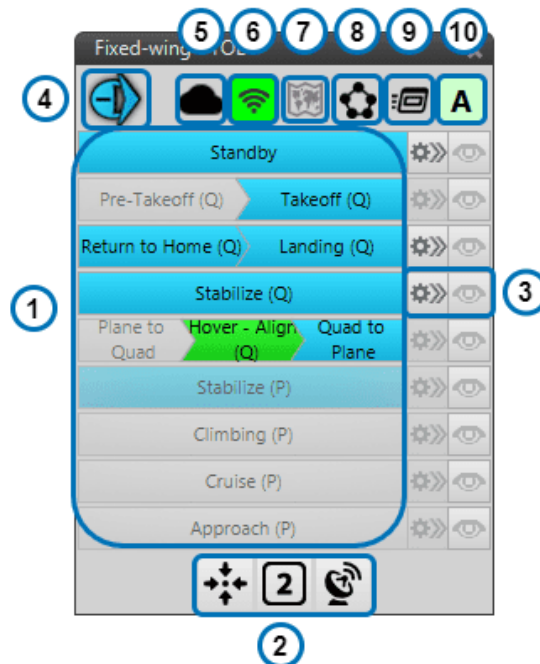
It is also possible to make temporary changes to the current mission by using *Detour commands*.

9.5 Commands

9.5.1 Veronte Panel

Veronte Panel is the main tool for operating Veronte. It includes all basic commands and information needed during a standard mission.

Note: Closing the Veronte Panel will hide it for the current workspace. In order to display it again, double click on the unit you want Veronte Panel displayed for.



Veronte Panel

1. Phase selection:

Display of the existing flight phases. The colour code indicates which phase is currently selected, and also the different phase transitions available.

- **Green:** This is the phase currently selected.

- **Blue:** This phase can be accessed from the current phase.
- **Blue (Translucid):** This phase can be accessed from the current phase, but only through an *Automation*
- **Grey:** This phase cannot be accessed from the current phase.

The distribution of phases on the Veronte Panel can be edited in the *Panel* Menu.

Note: If no phase is highlighted in **Green**, then the selected unit is currently in **INITIAL** Phase

2. Command Buttons:

Command buttons can be used to manually trigger certain actions, that can be programmed using *Automations*

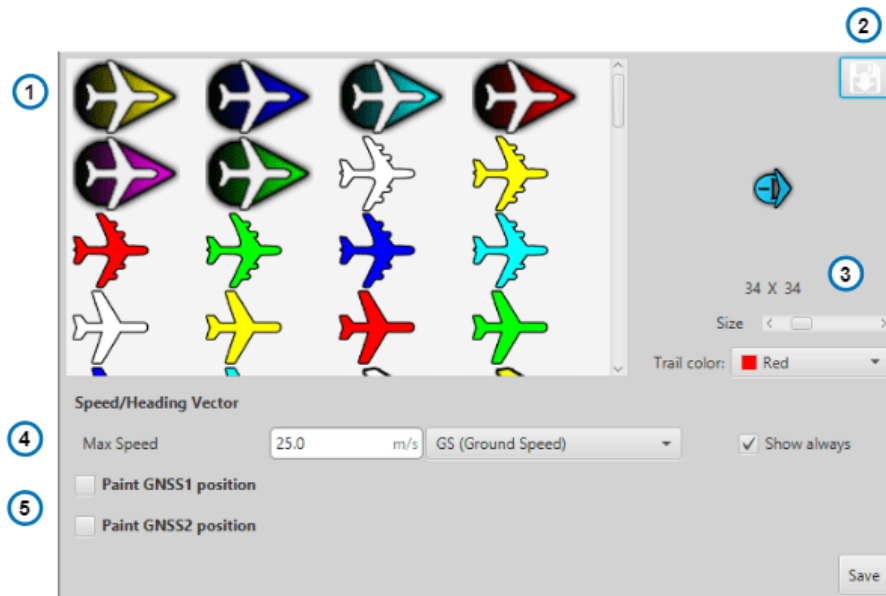
3. Guidance options:

Allow the user to interact with phase *Guidances*

- The **Gear** button allows to make changes to the guidance of the **Current phase**. These changes will only last for the duration of current phase and will disappear upon phase switch.
- The eye button allows displaying on the map the route for each phase, even if the phase is not currently selected.

4. UAV Icon:

This is the icon that will be used to represent this units' position on the map. Clicking on it will open the icon menu:



Icon Menu

1. **Icon List:** List of available icons
2. **Import Icon:** Import a custom icon to veronte Pipe
3. **Icon customization:** Choose icon size and the colour of the movement trail

4. **Speed display:** Allows to display an arrow indicating speed direction and magnitude




Speed arrow



5. **GNSS Position:** Display raw GNSS positions on the map.

5. **Veronte Cloud:**

Indicator of Cloud connection status.

6. **Radiolink status:**

Indicator of radiolink status. The colour of the  icon represents quality of the link, with **green** representing a good link connection, then changing to yellow-orange-red as the quality gets worse.

Clicking on the  icon will disable all telecommands to the unit .

7. **Center on map:**

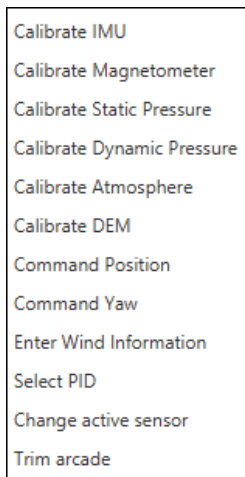
Center the map on the unit position

8. **Mission Edit menu:**

Quick access to the *mission* menu

9. **Quick Commands:**


Open the quick command list.




Quick Commands

10. **Flight Mode:**

Displays the current selected flight mode. Clicking on it allows to manually change the selected mode.

When a stick is correctly detected, the icon will have a green background . Otherwise, it will be

grey .

9.5.2 Quick Commands

Quick commands are common actions that are performed during a standard operation and are already predefined. This way, there is no need of modifying the configuration or building a specific *Automation*.

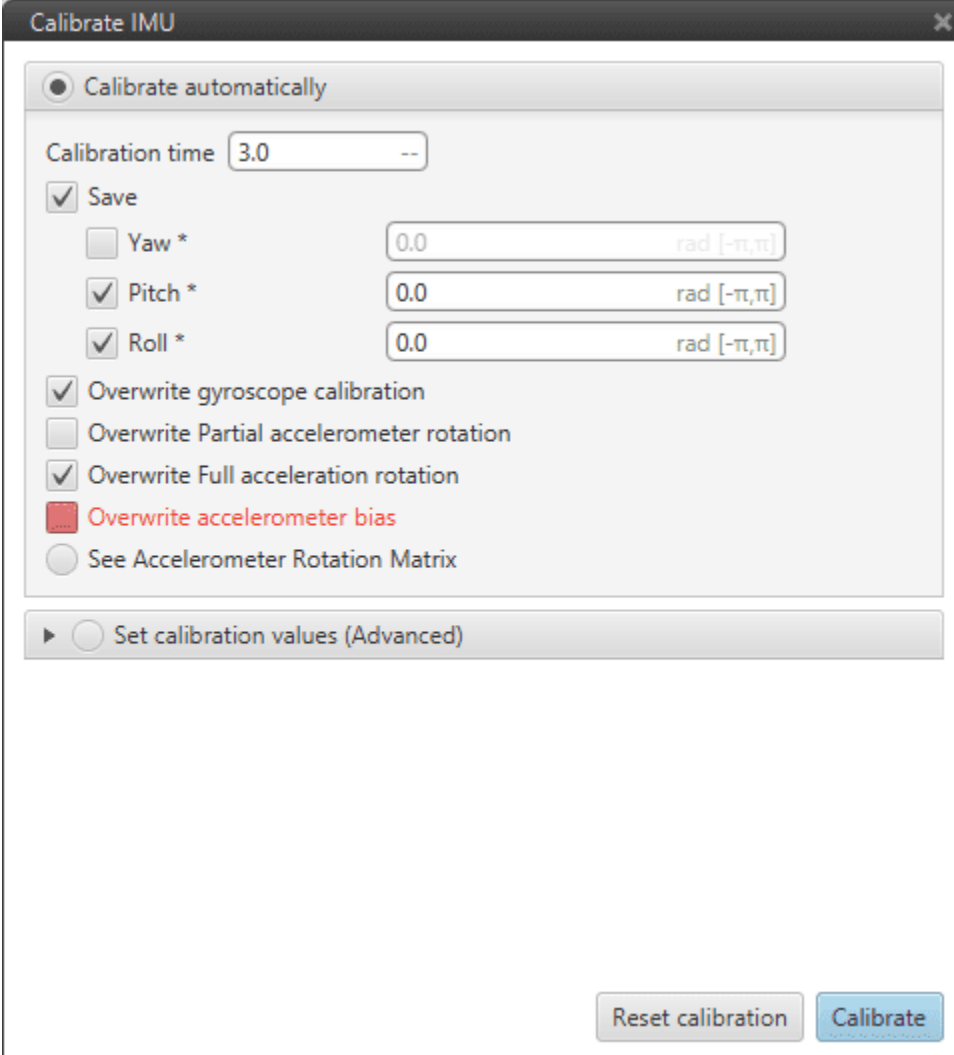
Note: Some of the following quick commands can also be triggered automatically using *Automations*.

Panel Quick Commands

These actions can be accessed from the specific drop-down list located on the Veronte Panel.

- **Calibrate IMU:**

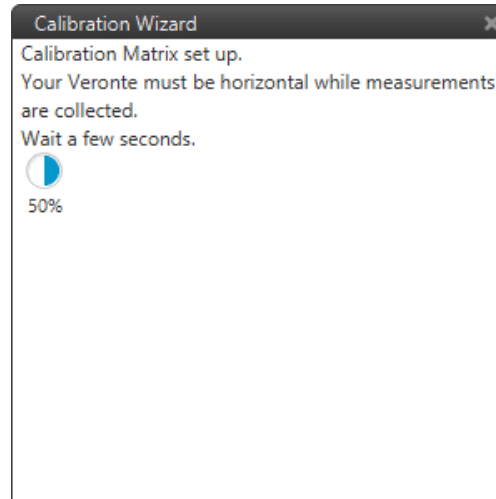
Correct any biases on the IMU calibration, including mounting offsets.



IMU Calibration

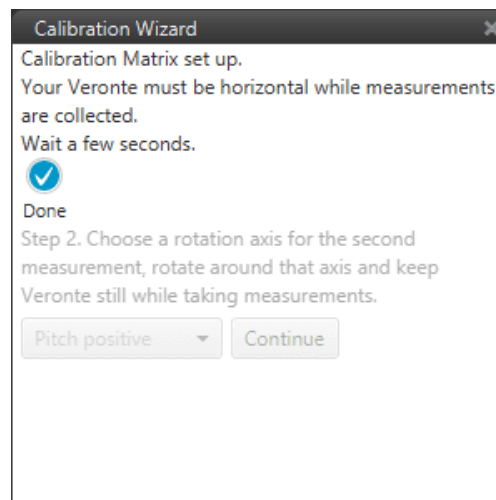
There are three calibration options available in the automatic calibration:

- **Overwrite Partial accelerometer rotation:** only calibrates pitch and roll angles. The autopilot must be horizontal during the calibration process:



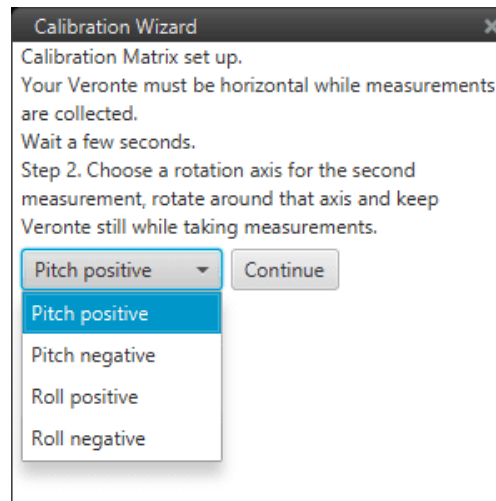
IMU Partial accelerometer rotation

- **Overwrite Full accelerometer rotation:** calibrates pitch, roll and yaw angles and it needs to take measurements in two orientations. During the first measurement the autopilot must be horizontal:



IMU Full accelerometer rotation first measurement

The second measurement requires to rotate the autopilot around the selected axis. The options are: pitch positive, pitch negative, roll positive, roll negative. The autopilot must be turned **more than 10 degrees** in the selected axis.



IMU Full accelerometer rotation second measurement

- **Overwrite accelerometer bias:** calibrates the accelerometer bias in each axis. The autopilot must be horizontal during the calibration process.

It is also possible to input calibration values manually if known:

Calibrate IMU

☐ Calibrate automatically

Calibration time

☒ Save

☐ Yaw * rad $[-\pi, \pi]$

☒ Pitch * rad $[-\pi, \pi]$

☒ Roll * rad $[-\pi, \pi]$

☒ Overwrite gyroscope calibration

☐ Overwrite Partial accelerometer rotation

☐ Overwrite Full acceleration rotation

☒ Overwrite accelerometer bias

☐ See Accelerometer Rotation Matrix

Set calibration values (Advanced)

Accelerometer 1

Accelerometer 2

Accelerometer 3

Accelerometer 4

Accelerometer 5

Accelerometer 6

| | Bias | Scale | Variance |
|---|----------------------------------|----------------------------------|----------------------------------|
| X | <input type="text" value="0.0"/> | <input type="text" value="0.0"/> | <input type="text" value="0.0"/> |
| Y | <input type="text" value="0.0"/> | <input type="text" value="0.0"/> | <input type="text" value="0.0"/> |
| Z | <input type="text" value="0.0"/> | <input type="text" value="0.0"/> | <input type="text" value="0.0"/> |

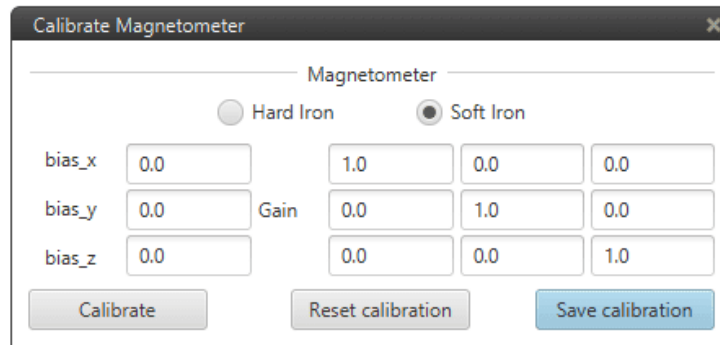
Reset calibration Save

IMU Advanced Calibration

- **Calibrate Magnetometer:**

Perform magnetometer calibration. Both Soft and Hard Iron calibrations are available. Soft Iron calibration will perform Hard Iron calibration also.

Tip: To achieve the best calibration, try to perform the calibration under operational conditions, including nominal currents and other magnetic disturbance sources that may be present during the operation



Calibrate Magnetometer

Magnetometer

☐ Hard Iron ☒ Soft Iron

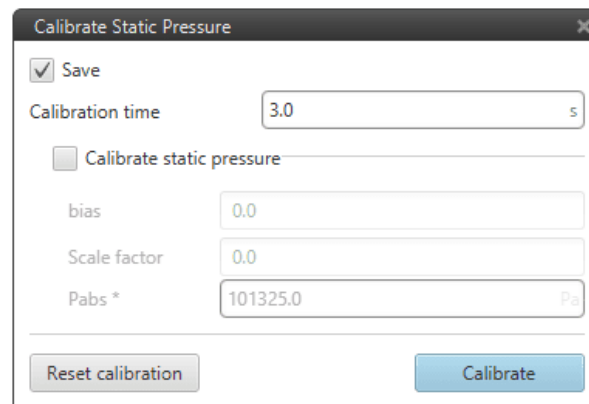
| | | | | | |
|--------|-----|------|-----|-----|-----|
| bias_x | 0.0 | Gain | 1.0 | 0.0 | 0.0 |
| bias_y | 0.0 | | 0.0 | 1.0 | 0.0 |
| bias_z | 0.0 | | 0.0 | 0.0 | 1.0 |

Calibrate Reset calibration Save calibration

Magnetometer Calibration

- **Calibrate Static Pressure:**

Calibration for the static pressure calculation.



Calibrate Static Pressure

☒ Save

Calibration time: 3.0 s

☐ Calibrate static pressure

bias: 0.0

Scale factor: 0.0

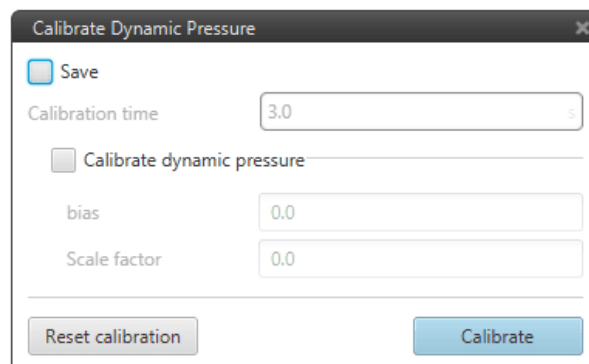
Pabs *: 101325.0 Pa

Reset calibration Calibrate

Static Pressure Calibration

- **Calibrate Dynamic pressure:**

Calibration for Airspeed calculation



Calibrate Dynamic Pressure

☐ Save

Calibration time: 3.0 s

☐ Calibrate dynamic pressure

bias: 0.0

Scale factor: 0.0

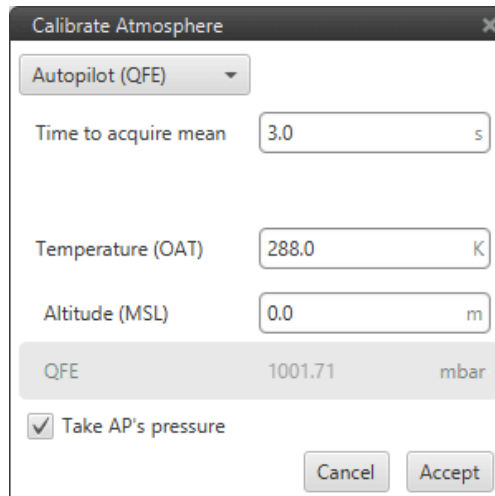
Reset calibration Calibrate

Dynamic Pressure Calibration

Warning: Dynamic pressure calibration is very sensitive to pressure disturbances. Make sure that the pressure intake is not disturbed while performing this calibration.

- **Calibrate Atmosphere:**

Calibration for MSL calculation with barometric pressure. Both QNH and QFE options are available. Static pressure can be selected from the autopilot measurement or it can be written by the user if known.

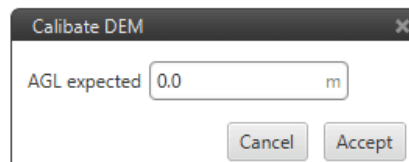


Atmosphere Calibration

- **Calibrate DEM:**

Calibrate any offset that the current terrain model may have for your current position.

Warning: Always perform this action on the ground, unless an accurate estimation of current AGL is available

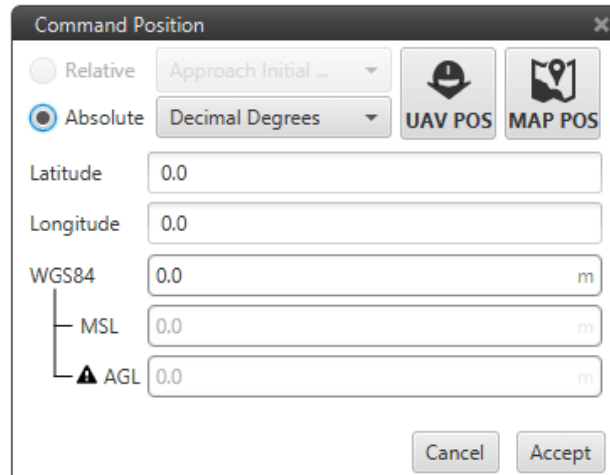


DEM Calibration

- **Command Position:**

Allows to manually modify the Position Navigation state.

Warning: If there is any absolute positioning sensor active (i.e. GNSS), this command will not work since it will be automatically overridden.



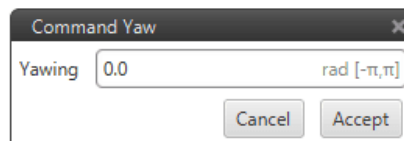
The 'Command Position' dialog box has a title bar with a close button. It contains two radio buttons: 'Relative' (unselected) and 'Absolute' (selected). To the right of the 'Absolute' button is a dropdown menu showing 'Decimal Degrees'. Further right are two buttons: 'UAV POS' with a UAV icon and 'MAP POS' with a map icon. Below these are input fields for 'Latitude' (0.0), 'Longitude' (0.0), and 'WGS84' (0.0 m). A tree view on the left shows 'WGS84' expanded, with 'MSL' and 'AGL' (marked with a warning triangle) as sub-items, each with its own input field (0.0 m). At the bottom are 'Cancel' and 'Accept' buttons.

Command Position

- **Command Yaw:**

Allows to manually modify the Yaw Navigation state.

Warning: If there is any Yaw sensor active (i.e. Magnetometer), this command will not work since it will be automatically overridden.

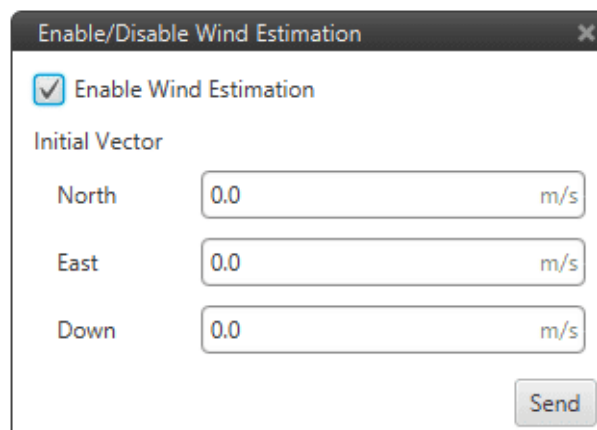


The 'Command Yaw' dialog box has a title bar with a close button. It contains a label 'Yawing' followed by an input field with '0.0' and a unit dropdown showing 'rad [-π,π]'. At the bottom are 'Cancel' and 'Accept' buttons.

Command Yaw

- **Enter Wind Information:**

Enter initial values for wind state and start wind estimation algorithm.

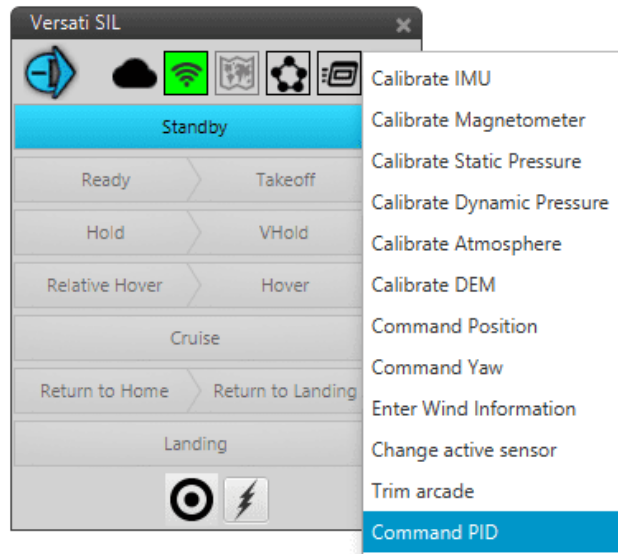


The 'Enable/Disable Wind Estimation' dialog box has a title bar with a close button. It contains a checked checkbox labeled 'Enable Wind Estimation'. Below is the section 'Initial Vector' with three input fields: 'North' (0.0 m/s), 'East' (0.0 m/s), and 'Down' (0.0 m/s). At the bottom right is a 'Send' button.

Wind Parameters

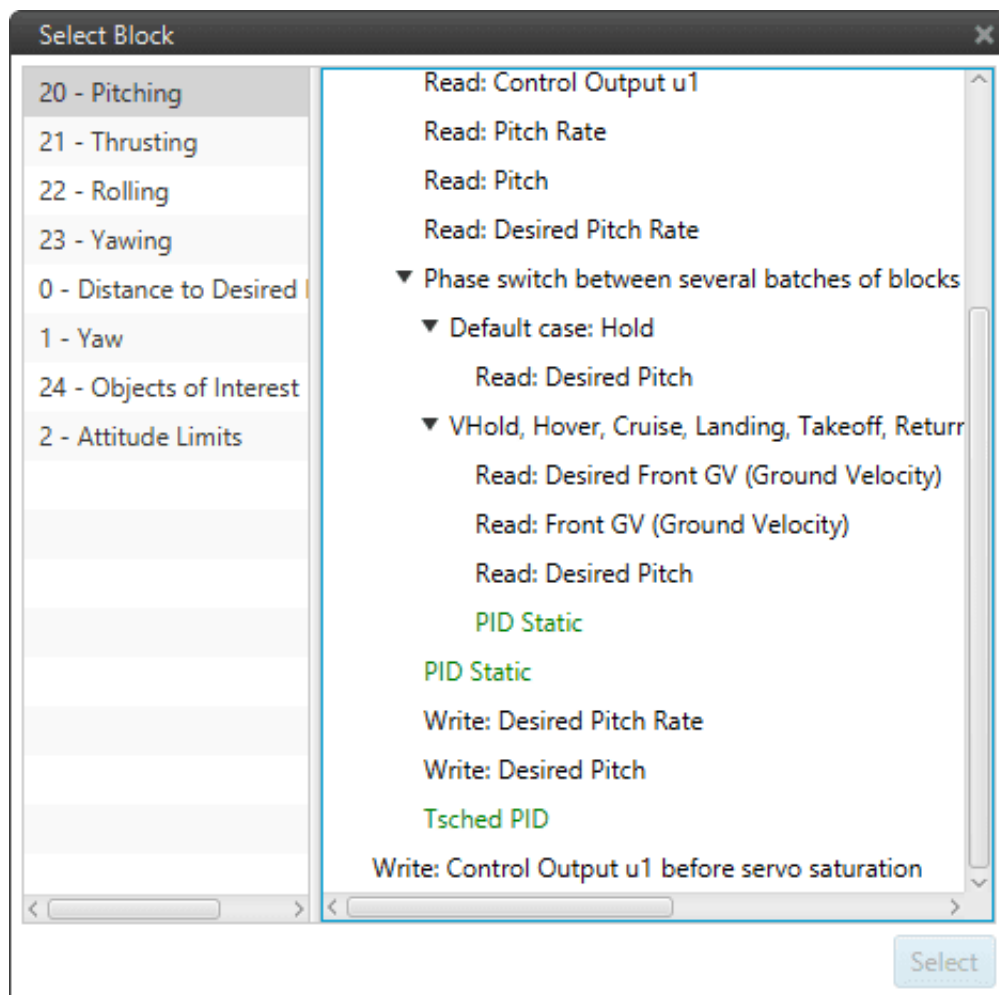
- **Command PID:**

Allows to modify PID gains during operation.



PID Command

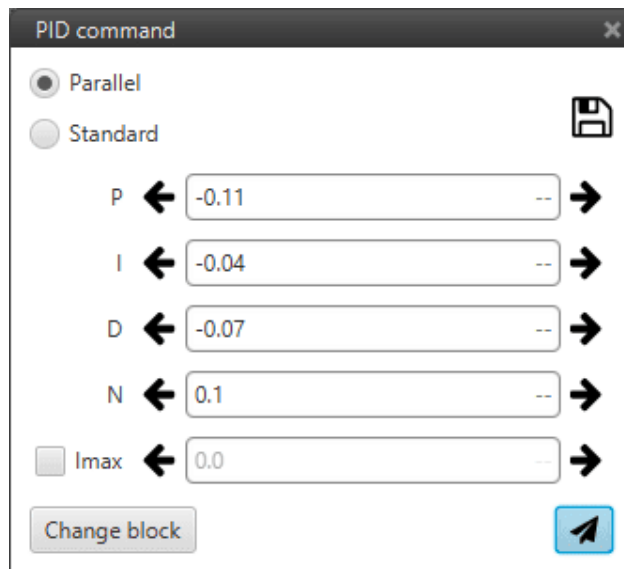
Click on Command PID, a new window will show up. Then select a valid PID block from the tree (they are shown in green). This tree represents how control blocks are organised in Programs.



Select PID

Behind this window there is another where users can modify the values of PID gains and type (standard or parallel). Here, users can either increase or decrease each of the gains P/I/D or N(derivative filter)/Imax. Modifying and clicking on **send**, updates the values of your vehicle. Clicking on **Change block** returns to the previous menu where you have to select again a new 'green' block.

Warning: All changes applied are volatile and will be lost after the system is reset. After the desired values are found, the save button must be clicked but restart is required.



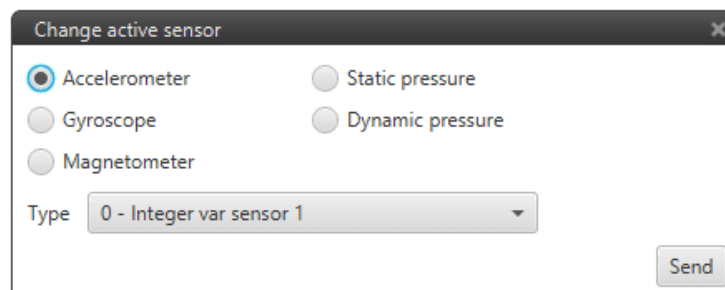
The 'PID command' window contains two radio buttons: 'Parallel' (selected) and 'Standard'. To the right of these is a save icon. Below are five input fields with left and right arrow buttons: 'P' with value -0.11, 'I' with value -0.04, 'D' with value -0.07, 'N' with value 0.1, and 'Imax' (checked) with value 0.0. At the bottom left is a 'Change block' button, and at the bottom right is a blue arrow button.

PID parameters

- **Change Active Sensor:**

Allows to change one of the currently selected sensors.

Warning: Remember that these changes are volatile. After the system is restarted, the selected sensor set will be the one defined on the configuration

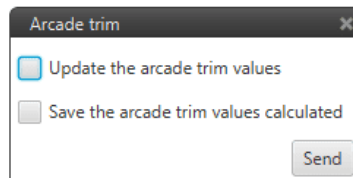


The 'Change active sensor' window shows five radio buttons: 'Accelerometer' (selected), 'Static pressure', 'Gyroscope', 'Dynamic pressure', and 'Magnetometer'. Below them is a 'Type' dropdown menu set to '0 - Integer var sensor 1'. A 'Send' button is at the bottom right.

Calibrate IMU

- **Trim Arcade:**

Calibrate current stick for arcade commands



The 'Arcade trim' window has two checkboxes: 'Update the arcade trim values' (checked) and 'Save the arcade trim values calculated' (unchecked). A 'Send' button is at the bottom right.

Trim Arcade

Detour Quick Commands:

The following commands are only accessible if the current phase has an active *Cruise* Guidance. They allow to make temporary changes to the execution of the current mission.

- **Fly to Hover:**

Create a volatile waypoint and change the current route. The created reference will disappear whenever this command is overridden.

Fly to Hover

- **Fly to Loiter:**

Create a volatile loiter and change the current route. The created reference will disappear whenever this command is overridden.

Fly to Loiter

- **Fly to Waypoint:**

Fly to an existing Waypoint. If the Waypoint belongs to a mission, the UAV will continue that mission after the waypoint is reached.

Fly to Waypoint

9.5.3 Additional Commands

Other available commands are:

- **Custom & Automatic Commands:**

By using *Automations*, the user is able to define any kind of command using the *actions* available in Veronte.

These commands can be triggered manually (buttons on Veronte Panel, physical switches, external devices, etc) or automatically (i.e. deploy a parachute if IAS is below stall speed).

- **Stick commands:**

Sticks and Flight boxes allow the configuration of **manual flight modes**. By using custom commands it is also possible to map additional actions to switches on the manual control devices, such as landing gear/flap position, lighting toggle, Auto/Manual toggle, parachute deployment, etc.

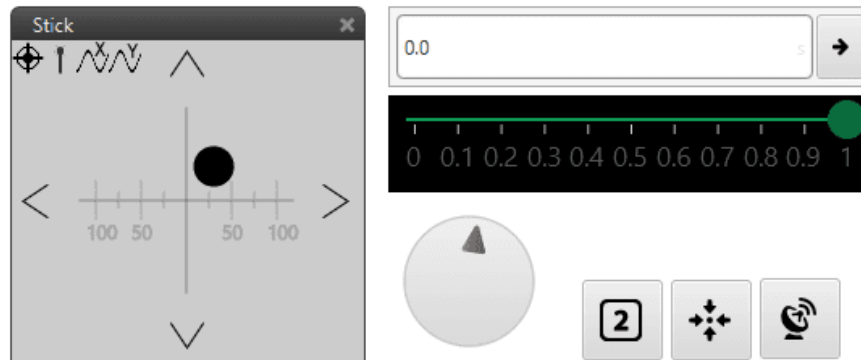




Stick commands

- **Workspace Command Widgets:**

Command widgets allow to update the values of certain variables in Veronte, that can then be used in flight modes and phases (i.e. desired speed, desired altitude...)



Workspace commands

- **VCP Commands:**

VCP stands for **Veronte Control Protocol**. By using VCP, it is possible to send commands to Veronte without the need of using Veronte Pipe.

VCP is designed for those users that wish to develop their own software applications, either to use them along Veronte Pipe or even replace it completely.

If you are interested in the use of VCP, please contact support@embention.com for more information.

A command involves any external action that causes a change on the system during the operation.

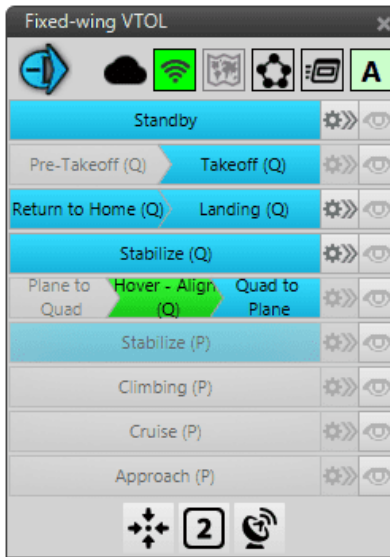
Note: Commands will never modify the autopilot's current configuration: the changes are volatile, and will disappear once the system is rebooted. On the other hand, sensor calibration can be changed with these commands and will be a permanent change.

Usual sources of commands are:

- **Operator/Internal Pilot:** phase changes, panel buttons, etc.
- **External Pilot:** override autonomous command, manual controls, etc.
- **Automatic actions:** triggered upon phase change, upon mode change, etc.
- **Payloads:** payload activation under certain circumstances.


The different command types available can be classified into:

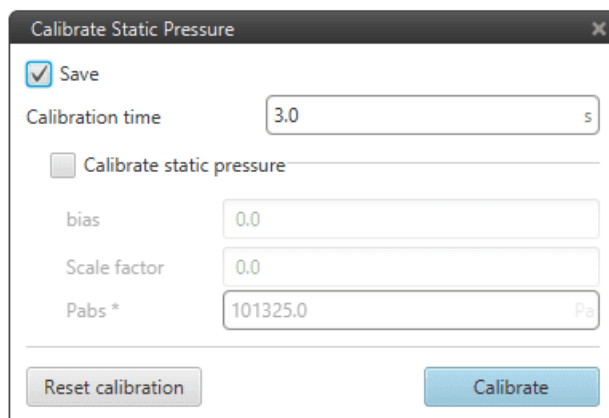
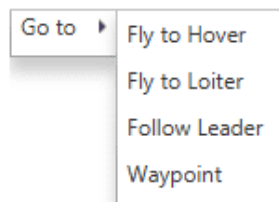
- **Veronte Panel:** this pannel is the basic Operator tool, and includes the use of Phase buttons and Command buttons. These commands can be triggered with a single click, and/or automatically.



Veronte Panel

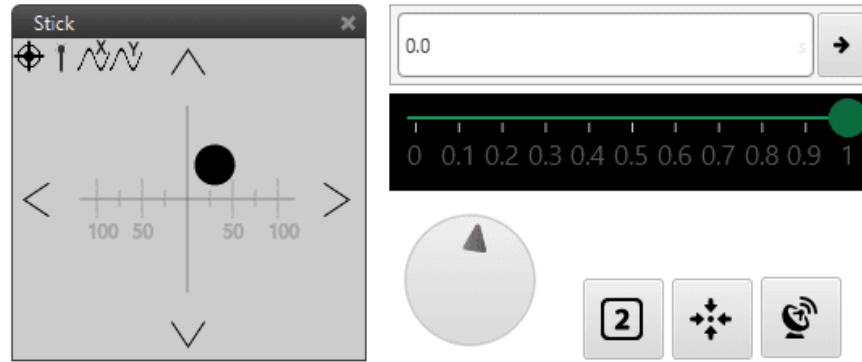
- **Quick commands:** these commands are also triggered by the operator, but usually require a couple more steps.

They can be accessed by either clicking the button  on the Panel, or by right clicking on the map. Examples of quick commands are internal sensor calibrations and mission detours.



Quick commands

- **Additional commands:** virtual sticks, external flight computers, command widgets, VCP messages, etc.



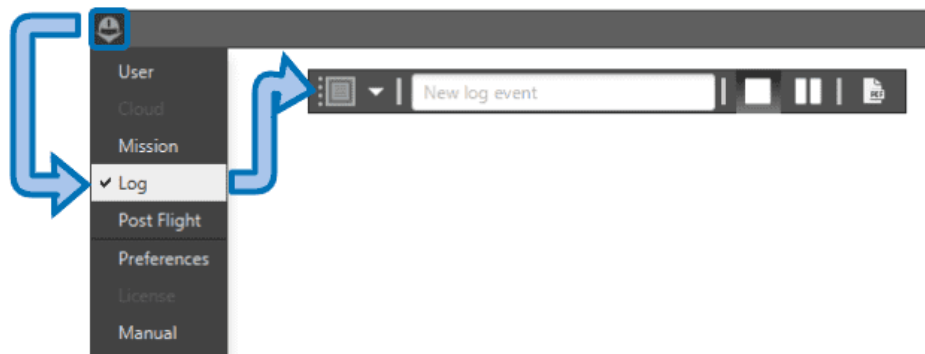
Additional commands


9.6 Log Report

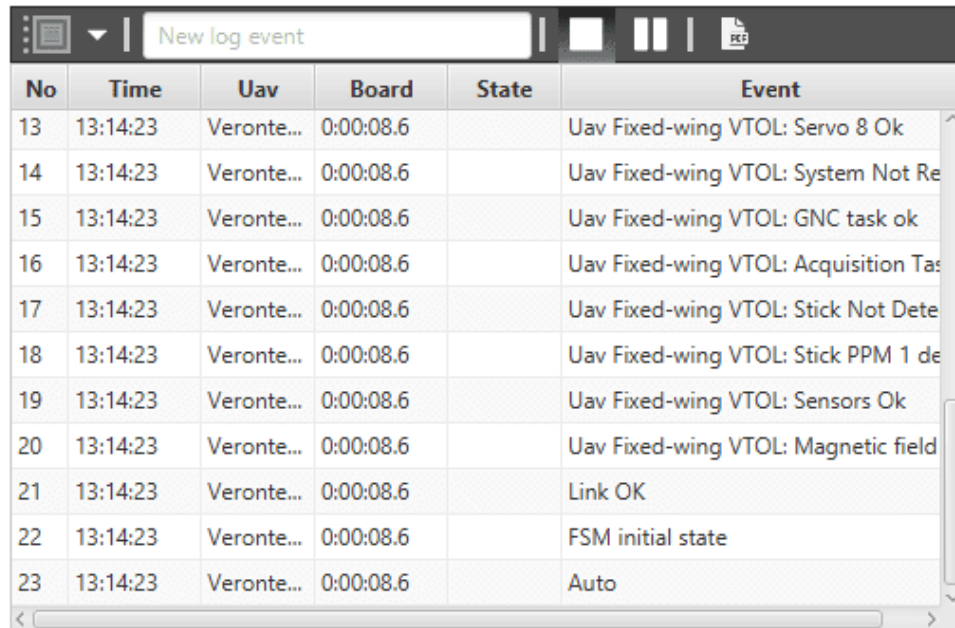
The Log feature in Pipe allows the user to create mission reports, that contain basic information (like mission duration, UAV ID, configuration name, date, etc.) and any relevant events, along with a timestamp.

If you are interested in logging of variables for post-flight analysis, please visit the [Post Flight](#) section.

The log menu can be accessed from the drop down list on the top left of the screen:



By default, Pipe starts logging when started, and will save the log when closed. It is also possible to stop/pause the log recording using the  buttons.



The screenshot shows a software window titled "New log event" with a table of log entries. The table has columns for No, Time, Uav, Board, State, and Event. The entries are numbered 13 through 23, all with a time of 13:14:23 and a board ID of 0:00:08.6. The events include various status reports for a Uav Fixed-wing VTOL, such as Servo 8 Ok, System Not Ready, GNC task ok, Acquisition Task, Stick Not Detected, Stick PPM 1 detected, Sensors Ok, Magnetic field, Link OK, FSM initial state, and Auto.

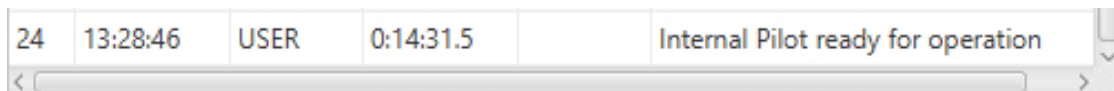
| No | Time | Uav | Board | State | Event |
|----|----------|------------|-----------|-------|---|
| 13 | 13:14:23 | Veronte... | 0:00:08.6 | | Uav Fixed-wing VTOL: Servo 8 Ok |
| 14 | 13:14:23 | Veronte... | 0:00:08.6 | | Uav Fixed-wing VTOL: System Not Ready |
| 15 | 13:14:23 | Veronte... | 0:00:08.6 | | Uav Fixed-wing VTOL: GNC task ok |
| 16 | 13:14:23 | Veronte... | 0:00:08.6 | | Uav Fixed-wing VTOL: Acquisition Task |
| 17 | 13:14:23 | Veronte... | 0:00:08.6 | | Uav Fixed-wing VTOL: Stick Not Detected |
| 18 | 13:14:23 | Veronte... | 0:00:08.6 | | Uav Fixed-wing VTOL: Stick PPM 1 detected |
| 19 | 13:14:23 | Veronte... | 0:00:08.6 | | Uav Fixed-wing VTOL: Sensors Ok |
| 20 | 13:14:23 | Veronte... | 0:00:08.6 | | Uav Fixed-wing VTOL: Magnetic field |
| 21 | 13:14:23 | Veronte... | 0:00:08.6 | | Link OK |
| 22 | 13:14:23 | Veronte... | 0:00:08.6 | | FSM initial state |
| 23 | 13:14:23 | Veronte... | 0:00:08.6 | | Auto |

Pipe Log

By default, Pipe registers the following events:

- Phase and Mode changes
- Position Fix
- File System OK
- Out of georeferenced area
- Power OK
- Servo 1-8 Saturation
- System Ready to start
- GNC step lost
- Acquisition step lost
- Stick not detected
- PPM not detected
- Sensors error
- Link Error


It is possible to add entries to the log manually. They will be registered with the appropriate timestamp.

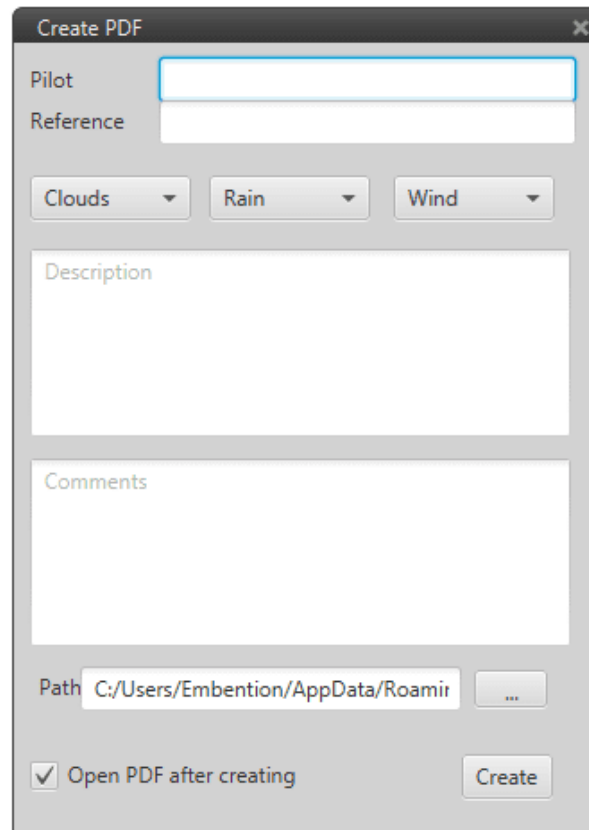


The screenshot shows a single manual log entry in the table. The entry number is 24, the time is 13:28:46, the user is USER, the board ID is 0:14:31.5, and the event description is "Internal Pilot ready for operation".

| | | | | | |
|----|----------|------|-----------|--|------------------------------------|
| 24 | 13:28:46 | USER | 0:14:31.5 | | Internal Pilot ready for operation |
|----|----------|------|-----------|--|------------------------------------|

User Entry

Once the mission is complete, clicking on the  button will generate the document. Additional data can be included when generating the document, such as the pilot identifier, weather conditions, extra comments,...



Create PDF

Pilot

Reference

Clouds Rain Wind

Description

Comments

Path C:/Users/Embention/AppData/Roamir ...

☒ Open PDF after creating

Create

Generate Log


Once all units are set up and the mission has been loaded, the system is ready to fly.

Before starting the operation, it is very important to check that the operator(s) will have access on the screen to all the information and commands that will be needed during the flight.

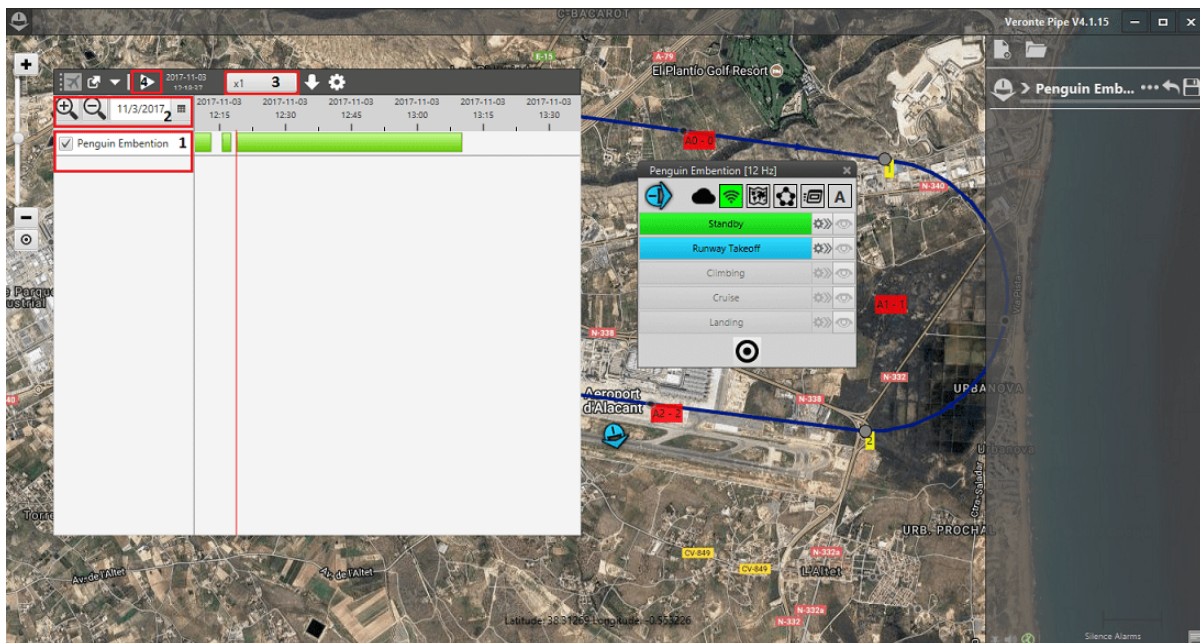
Below can be observed an example of an operation setup with Veronte Pipe:

POST FLIGHT

The post flight menu allows the user to check all the information stored during a flight operation. There exist different options to display the telemetry of a certain operation, and they are presented in this section.

To access the menu, click on  (Main Menu), and then select Post Flight in the pull-down menu.

10.1 Tour Play



Post Flight Panel

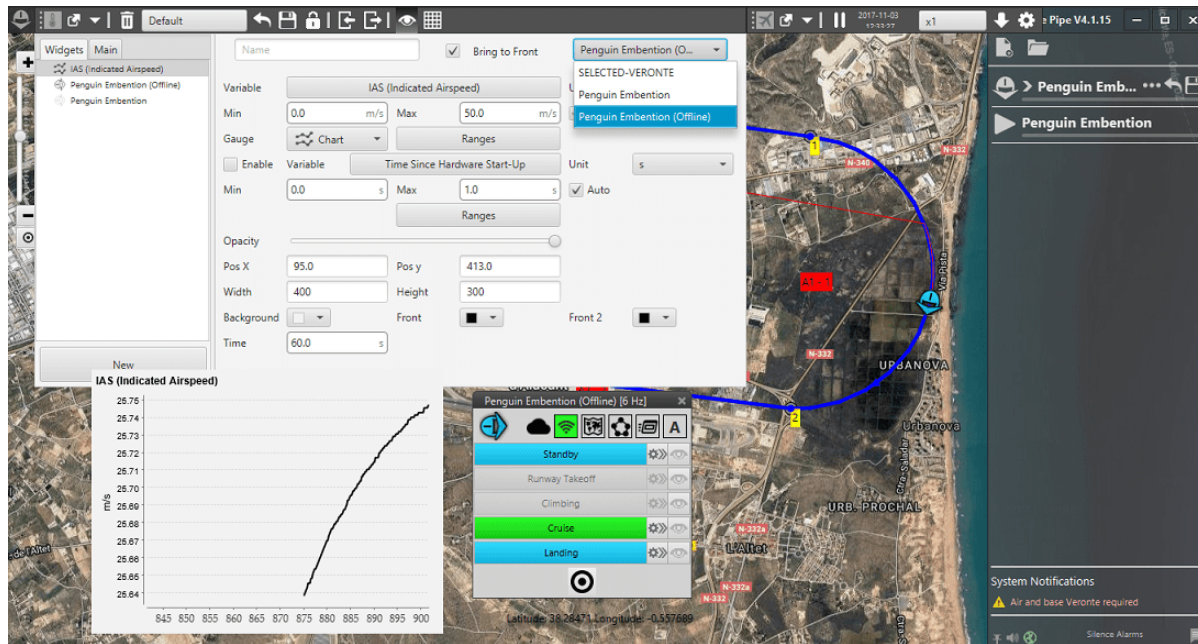
When opening the post flight menu, there are a series of options to configure in order to display the tour:

1. The autopilot, whose information wants to be played on the tour, has to be selected among the ones that appear on the list on the left side.
2. The user here can select the date of the flight that is going to be replayed.
3. The speed of the tour can be configured in this option (x1 means normal speed).

4. The play button starts the tour.

When clicking the play button, a new autopilot will appear in the right sidebar with the name of the unit plus Offline. As it can be seen in the previous figure, a new Veronte Panel appears on the screen showing the phase changes as they occurred in the real operation.

Regarding the telemetry, all the variables that were in the Telemetry Link during the real flight can be displayed in the tour.



Workspace Panel

For example, to display the IAS that the aircraft had during the flight played, just select the offline autopilot in the pull-down menu, and the gauge will show the indicated airspeed of that operation.

Warning: If a variable was not in the telemetry link during a flight, it can not be later shown in the tour of that flight.

10.2 Data Export

The telemetry generated during an operation is stored in different ways (see section [Telemetry](#)). According to how each one of the telemetry variables has been stored (data link, onboard log, user log and fast log) it has to be downloaded in a different way.

10.2.1 Data link

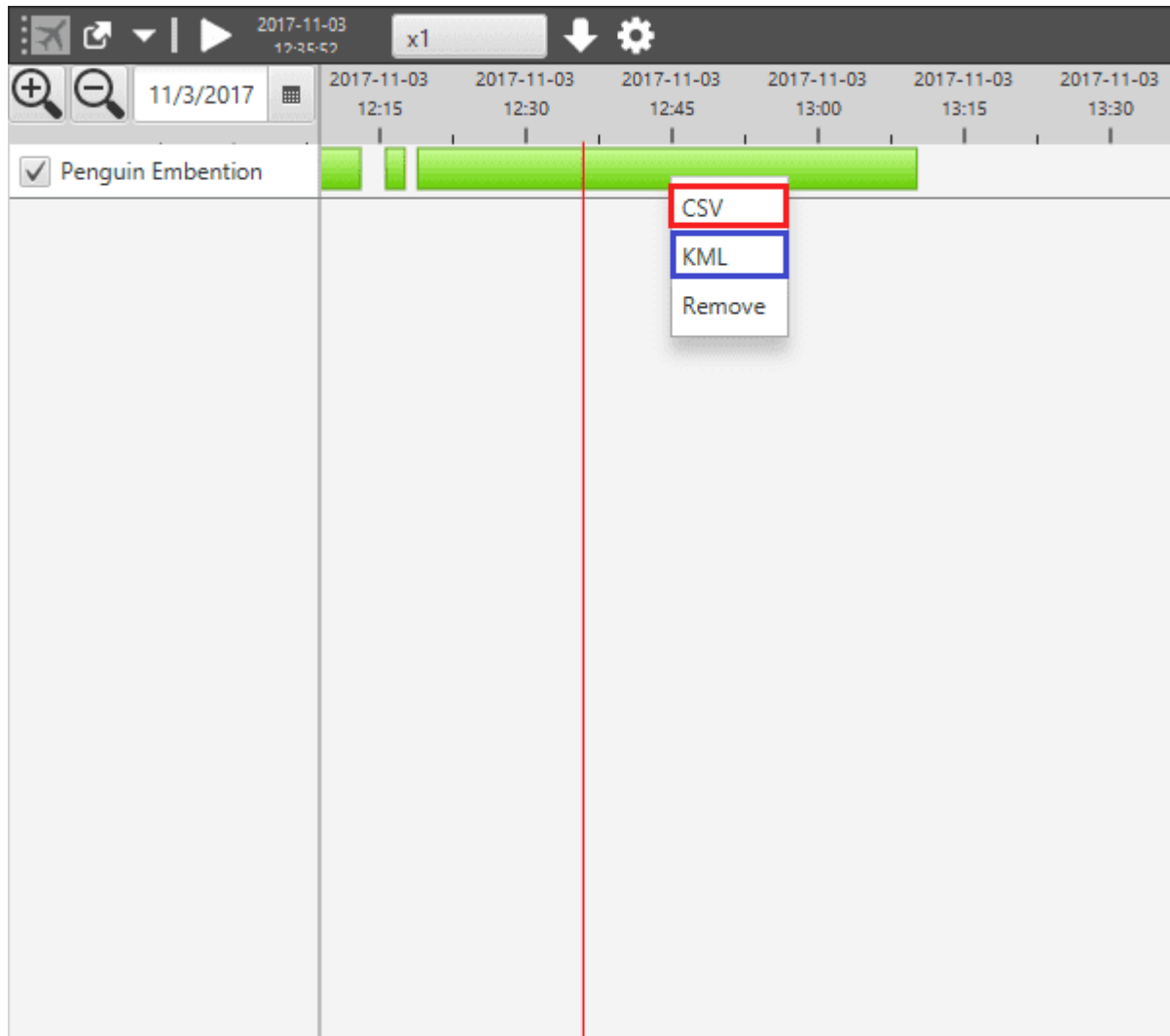
The information obtained directly from the **Data Link** is saved at the “appdata” folder of the operative system. This data is the telemetry sent by the air autopilot to Veronte Pipe, so it is stored in the computer that has the software installed. The exact path (in version 4.X) to obtain it is:

C:\Users\<Username>\AppData\Roaming\VerontePipe\<Veronte Pipe version>\session

In the folder that appears in this route, there are several folders each one corresponding to one operation. To identify the desired one check the date on the folder name. In order to play a tour of a flight in a different computer, just copy the folder to the same route in the other computer and the information will appear in the post flight menu as was explained in the previous section.

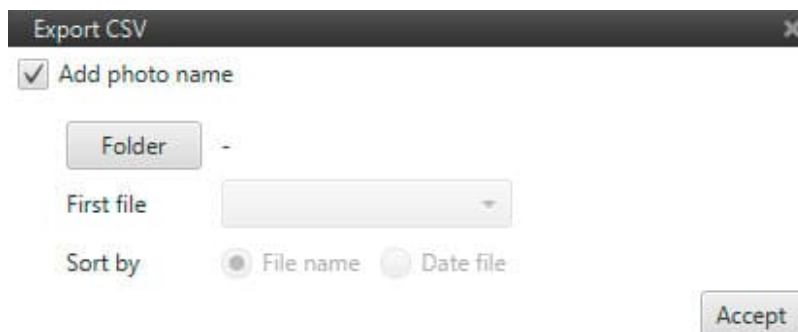
It is also possible to download the information of the Data Link for post-processing purposes. Right-clicking on the green bar of the desired flight will open a menu where it is possible to download that information in:

- **CSV file** – This file uses “;” for data separation and “,” for decimal indication.
- **KLM file** – It can be opened using Google Earth and allows to check the aircraft route in 3D.



Post-flight data export

When downloading a CSV file, Pipe provides the user with an option to automatically name the photos taken during a photogrammetry mission.



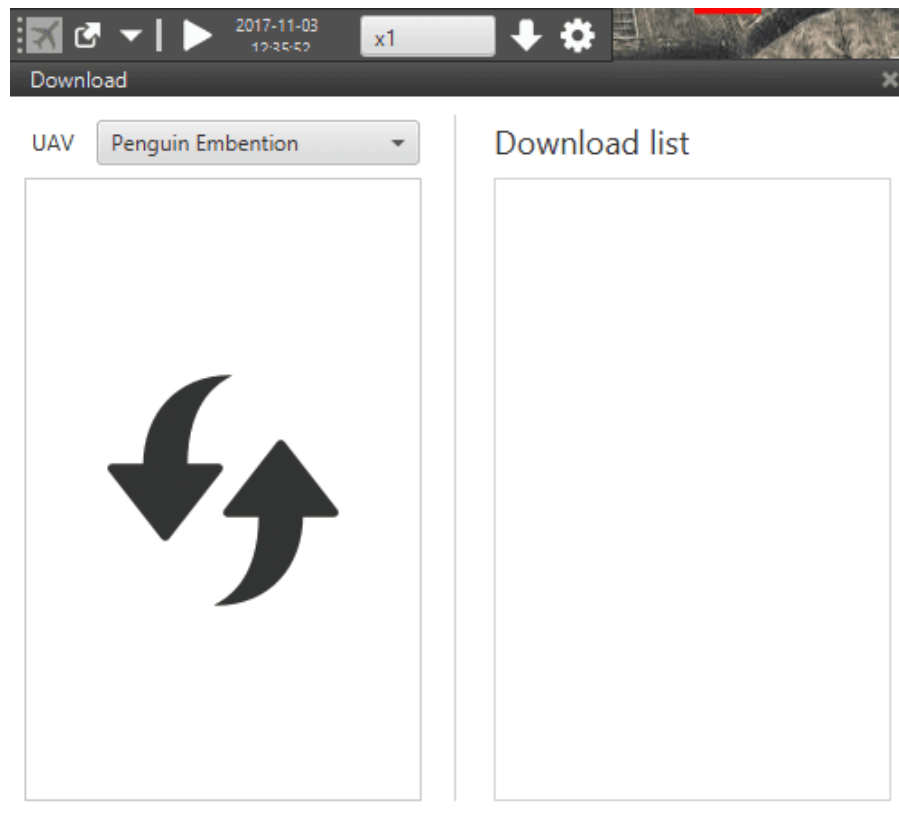
Export CSV

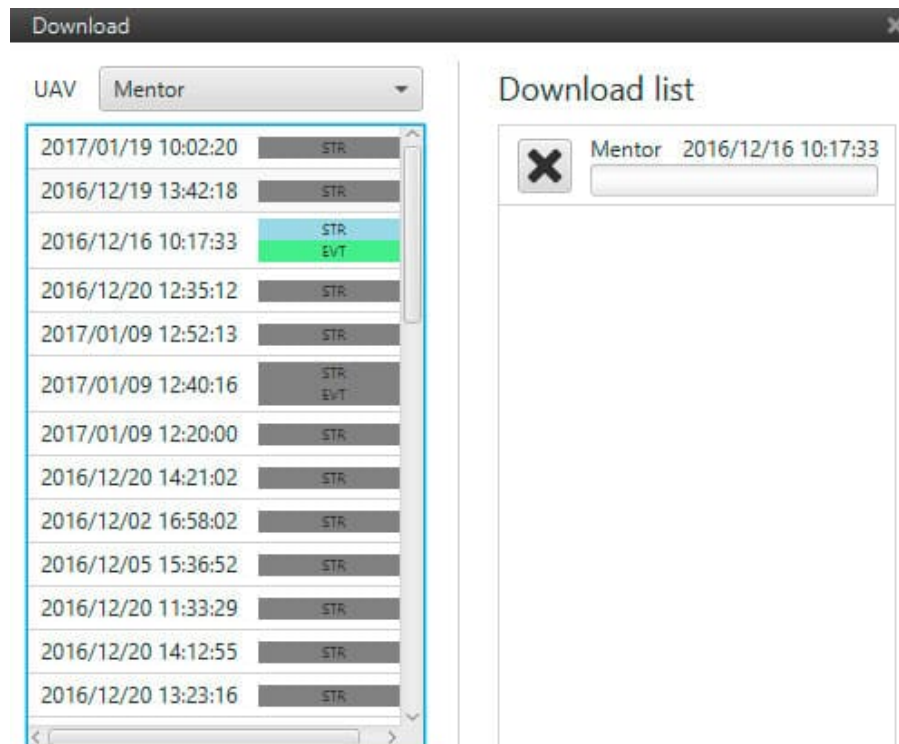
The button Folder allows the selection of the folder that contains the photos taken during the mission. Then, in First file will appear all the files of that folder and the user has to select the file that corresponds to the first photo. Finally,

Pipe permits to sort the photo files by name or date.

10.2.2 LOG

The data of the other three type of logs is stored in the SD Card of the air autopilot, so it has to be downloaded to the computer in order to obtain it. To do this, it's sufficient to open Postflight bar and click on the down-directed arrow, then select the Veronte unit for data downloading and refresh the page in order to choose the flight files to be download (checking the flight date and time).





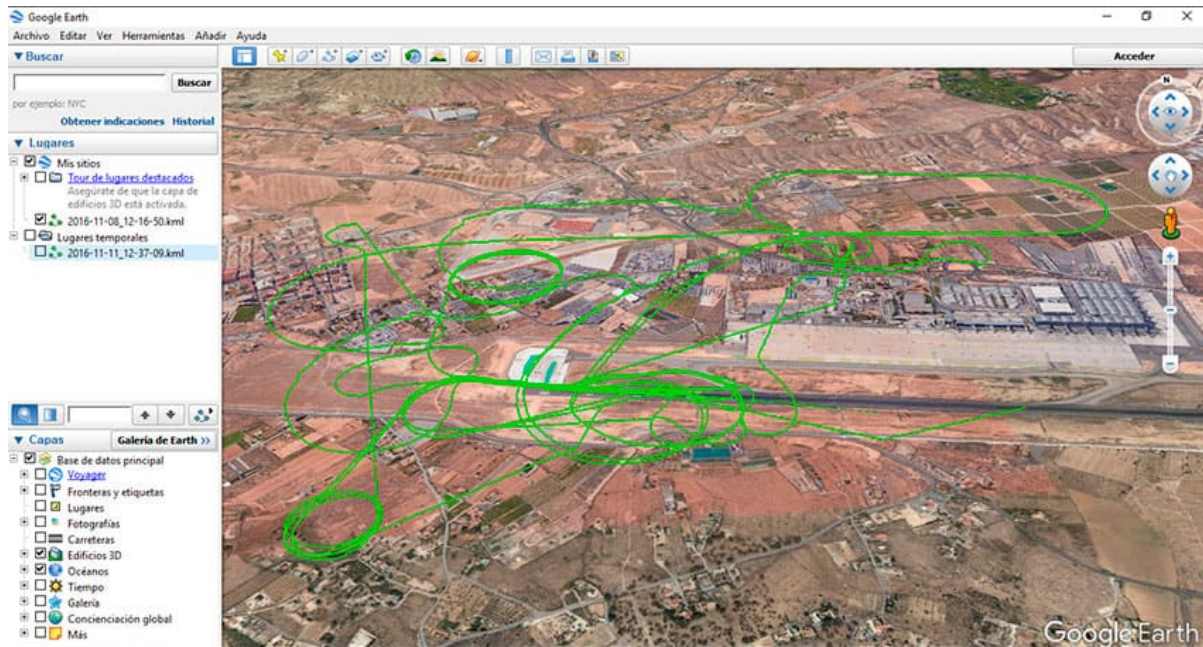
Data Export

The bar STR corresponds with the information of the Onboard Log and EVT is for the User Log. Once the file has been downloaded, the bar will turn green and right-clicking over it (SRT or EVT) will allow the user to download a CSV file with that information.

The following two images show a .CSV and a .KML files opened respectively with Microsoft Excel and Google Earth.

| | J | K | L | M | N | O | P | Q | R | S |
|-------|-------------|-----------|----------|------------------|--------------|-----------------|---------------------|-----------|-----------|-----------|
| 52852 | GS - Ground | Longitude | Latitude | Magnetometer - Y | AGL - Height | WGS84 Elevation | GPS ECEF Position Z | Heading | Yaw | Pitch |
| 52853 | 0.000000 | -0.009955 | 0.669152 | 0.008506 | 15.839.272 | 157.641.953 | 3.935.194.250.000 | 2.915.398 | 2.915.398 | -0.010472 |
| 52854 | 0.099976 | -0.009955 | 0.669152 | -0.101984 | 15.780.312 | 157.583.496 | 3.935.194.250.000 | 2.911.209 | 2.911.209 | -0.010472 |
| 52855 | 0.099976 | -0.009955 | 0.669152 | -0.002543 | 15.735.466 | 157.538.986 | 3.935.194.250.000 | 2.917.493 | 2.917.493 | -0.010472 |
| 52856 | 0.099976 | -0.009955 | 0.669152 | -0.107508 | 15.769.096 | 157.572.784 | 3.935.194.250.000 | 2.921.681 | 2.921.681 | -0.011519 |
| 52857 | 0.000000 | -0.009955 | 0.669152 | -0.101984 | 15.829.353 | 157.632.874 | 3.935.194.250.000 | 2.922.728 | 2.922.728 | -0.013614 |
| 52858 | 0.000000 | -0.009955 | 0.669152 | -0.096459 | 15.765.465 | 157.569.153 | 3.935.194.250.000 | 2.923.776 | 2.923.776 | -0.013614 |
| 52859 | 0.099976 | -0.009955 | 0.669152 | -0.090935 | 15.718.254 | 157.522.095 | 3.935.194.250.000 | 2.920.634 | 2.920.634 | -0.013614 |
| 52860 | 0.000000 | -0.009955 | 0.669152 | -0.107508 | 15.758.446 | 157.562.958 | 3.935.194.250.000 | 2.923.776 | 2.923.776 | -0.011519 |
| 52861 | 0.099976 | -0.009955 | 0.669152 | -0.113033 | 15.678.627 | 157.483.475 | 3.935.194.250.000 | 2.931.106 | 2.931.106 | -0.011519 |
| 52862 | 0.000000 | -0.009955 | 0.669152 | -0.107508 | 15.722.450 | 157.527.466 | 3.935.194.250.000 | 2.925.870 | 2.925.870 | -0.013614 |
| 52863 | 0.099976 | -0.009955 | 0.669152 | -0.008067 | 15.675.270 | 157.480.453 | 3.935.194.250.000 | 2.929.012 | 2.929.012 | -0.013614 |
| 52864 | 0.000000 | -0.009955 | 0.669152 | -0.101984 | 15.716.087 | 157.521.271 | 3.935.193.750.000 | 2.932.153 | 2.932.153 | -0.013614 |
| 52865 | 0.099976 | -0.009955 | 0.669152 | -0.101984 | 15.652.443 | 157.457.794 | 3.935.193.750.000 | 2.923.776 | 2.923.776 | -0.013614 |

Post flight in Microsoft Excel (.csv)



Post flight in Google Earth (.kml)

SIMULATION

11.1 Professional HIL

α

(11.1)

11.1.1 Mounting

11.1.1.1 Cable Connection

Package content includes the following cable. One connector must be connected to Veronte Autopilot and the other one to the CS Cable or the Aircraft connector in case of HIL Simulation using the complete platform system.



HIL Simulator Cable

11.1.1.2 Veronte Pipe Configuration

There are 2 configuration items on Veronte Pipe, one relating communications between the application and the X-Plane simulator and another one refereeing the autopilot configuration.

Veronte Pipe acts as a “bridge” between Veronte Autopilot and XPlane. Hence, these are the variables that the Autopilot receives from Pipe and how they are processed:

- **Heading, pitch, roll:** received in [deg], converted to [rad]
- **Roll rate, pitch rate, yaw rate:** received in [rad/s]
- **Longitude, latitude:** received in [deg], converted to [rad]
- **Height:** received MSL height in [ft], converted to ellipsoidal height in [m]

$$H_{ellipsoidal} = Ellipsoid(lon, lat) + h_{MSL}$$

- **X velocity, Y velocity, Z velocity:** received in [m/s]
- **Accelerations (G-loads):** received in g's, converted to [m/s²]
- **Incidence angle (Alpha) and sideslip angle (Beta):** received in [deg], converted to [rad]
- **Indicated Air Speed:** received in [kts], converted to [m/s] and projected to body axes

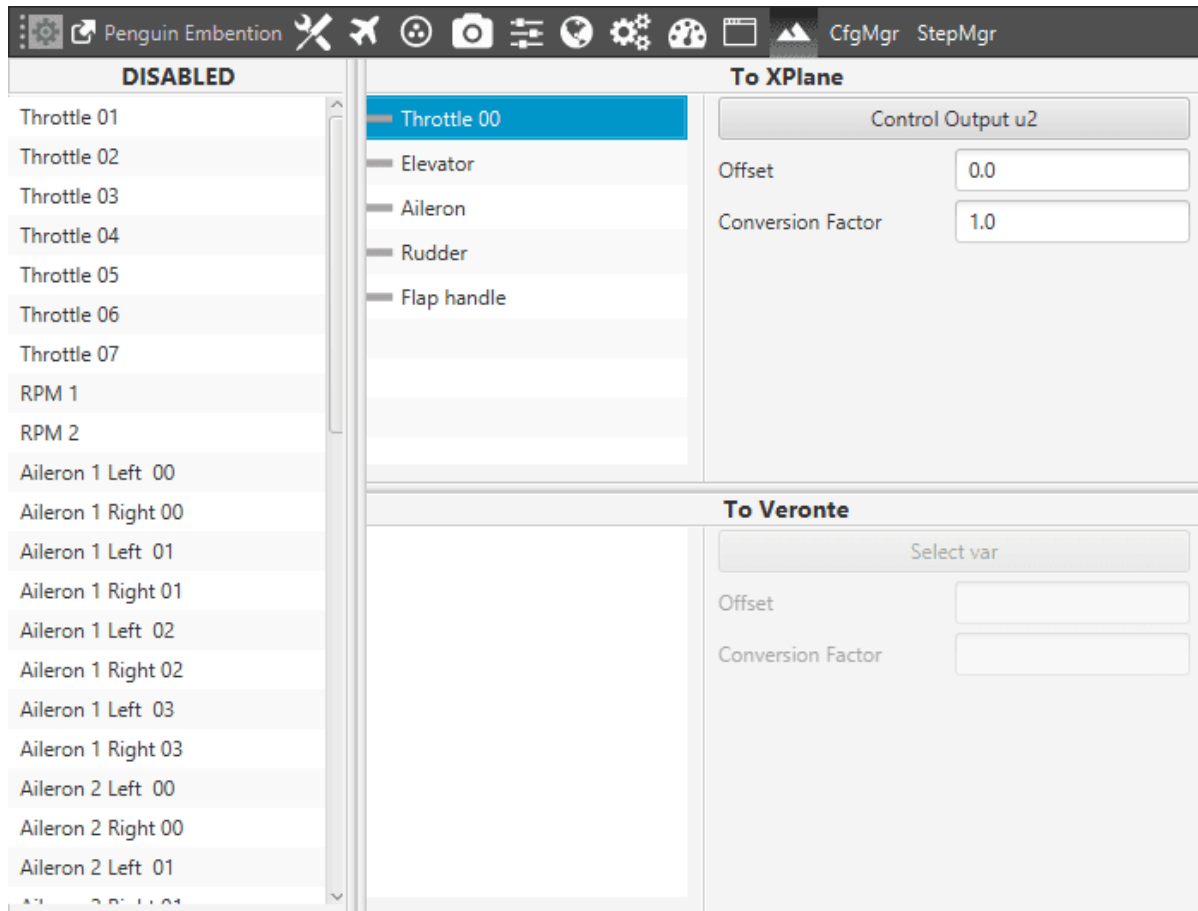
$$IAS_{BA} = IAS_{WA} \cos(\alpha) \cos(\beta)$$

- **Height above ground level:** received in [ft], converted to [m]

On the other hand, as it is explained in the following section, Veronte Pipe sends XPlane the commands received from the Autopilot according the variables of the aircraft configuration in XPlane.

11.1.1.3 Autopilot Configuration

HIL simulation tab is available within Veronte Autopilot setup toolbar. The user can link the variables on Veronte Autopilot with the corresponding ones in X-Plane simulator.



Veronte Pipe – HIL Setup

In this panel, X-Plane variables are available on the left side (Disables). In addition, it can be seen two section more To XPlane and To Veronte.

In order to configure the simulation variables, users have to:

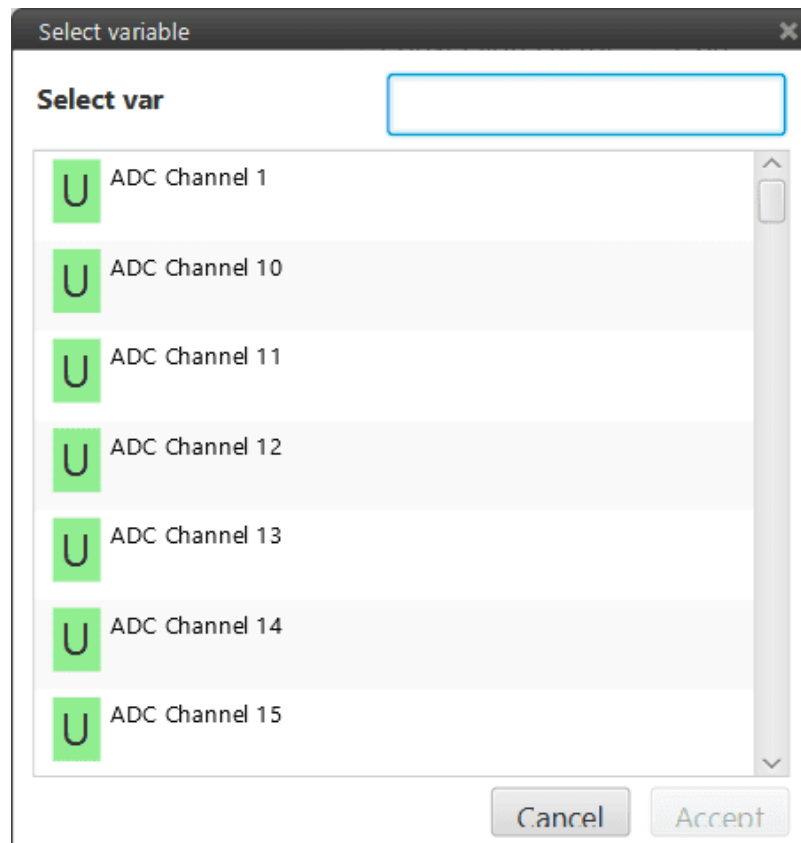
1. Enable the ones that have been configured in the aircraft model (Plane Marker). Just drag and drop them into To Xplane section.

Warning: Always make sure that surfaces are moving in the right direction and with the correct deflection angle.

To avoid mistakes is possible to set a positive fixed deflection (Control Tab) in Standby phase for all surfaces and control surface deflections in the X-Plane model.

Surface control variables are of two types:

- Radians measure (variables with numbers)
 - Degrees measure (variables with no numbers)
2. Once X-Plane variables have been enabled, select the actuator variable (Control Output) that matches with the ones in Veronte autopilot. A new window will be displayed for each variable.



3. Set a Conversion Factor or Offset, if it is necessary. Conversion factor multiplies the Veronte output signal and can be used in case units on Veronte and the X-Plane simulator do not match (Surfaces in X-Plane move normally in a [0,1] range). The following operation allows to converting an angle [deg] measure to the X-Plane form:

$$(\text{Angle} + \text{Offset}) \text{Conv.Factor} = \dots$$

When Angle and Offset are measured in [rad] and the Conversion factor is a constant (normally it can be calculated as $1/(\text{deflection angle in [rad]})$).

In the case of [rad] measures, the Conversion factor must be set in 57,29578 ([deg]-[rad] conv. factor).

Finally, it is necessary to configure the communication between Veronte Pipe and X-Plane, see section below.

11.1.1.4 Veronte Pipe Communications

In order to start the simulation, select the HIL option (1) on the Veronte Unit, this is available on the side menu . Popup screen will be displayed (2) for selecting the kind of simulation and configuring the parameters.



Veronte Pipe – HIL Communications

Changing the default values is only recommended for advanced users. Press start in order to start the data transfer between X-Plane and Veronte Autopilot.

Warning: The simulation must be started when the aircraft is in the Initial phase (the one that gets in once is powered). In this phase the X-Plane will simulate the GPS signal to locate the autopilot in the place indicated by the airport on X-Plane.

11.1.2 X-Plane 10 Settings

11.1.2.1 X-Plane 10 Configuration

X-Plane 10 demo version can be downloaded from this [link](#).

X-Plane 10 communications settings must be edited in order to have communication with Veronte system. Follow the next steps in order to make a proper configuration.

For low-performance computers, it may be needed to reduce the graphics quality on the simulator, as described below.

11.1.2.1.1 Aircraft Model Installation

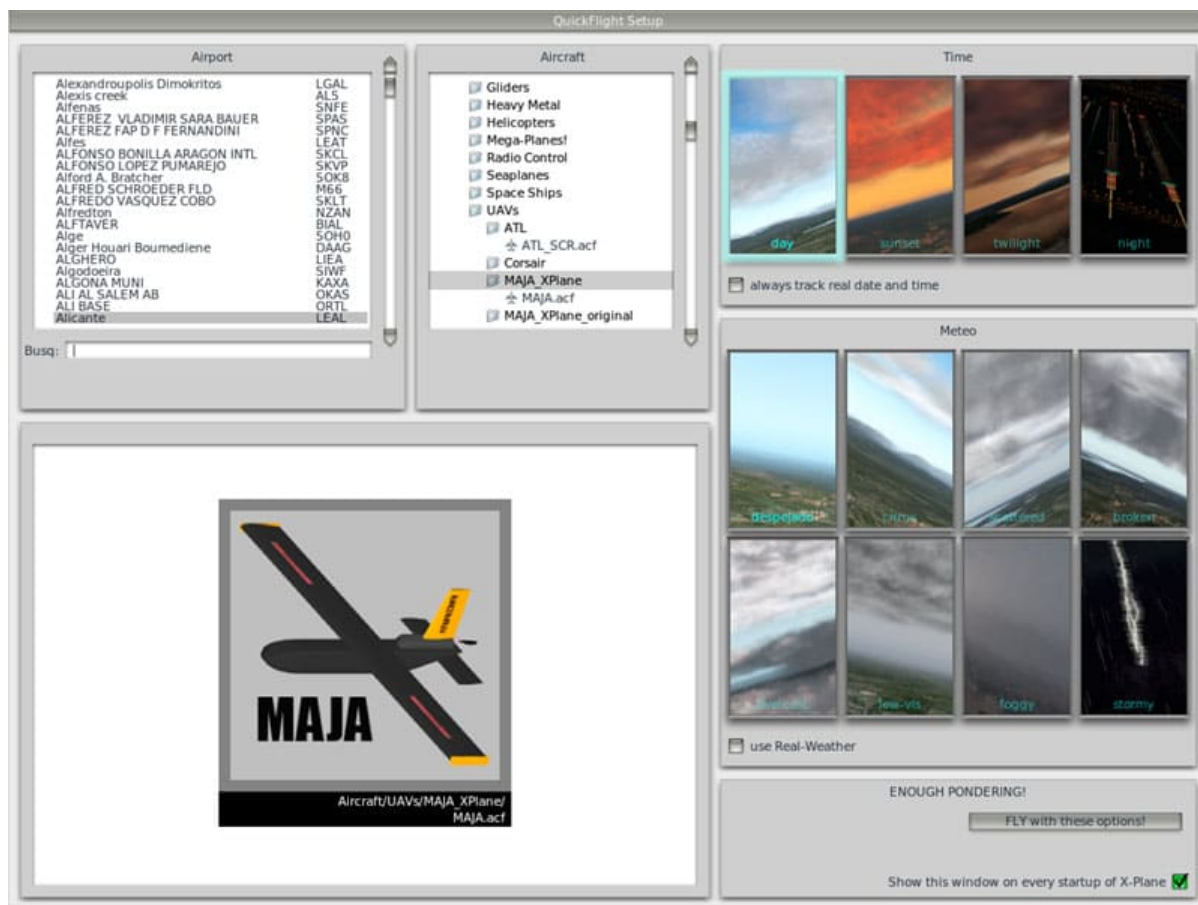
X-Plane 10 simulator is compatible with a wide variety of platforms: airplane, helicopter, multicopter, surface vehicle.... In order to create the platform model, Plane Maker tool provided by X-Plane 10 must be used.

Once the aircraft model has been created, it can be integrated on the X-Plane 10 simulator by following next steps:

- Copy the model folder to the “Aircraft” folder within the X-Plane 10 installation directory.
- Copy the content in the “Airfoils” folder, available on the aircraft model folder, to the “Airfoils” directory within the X-Plane 10 installation directory.

11.1.2.1.2 X-Plane 10 Setup

On X-Plane 10 execution, **Quick Flight Setup** window will be displayed; select which aircraft to use, the starting airport and weather conditions to be simulated during the flight.

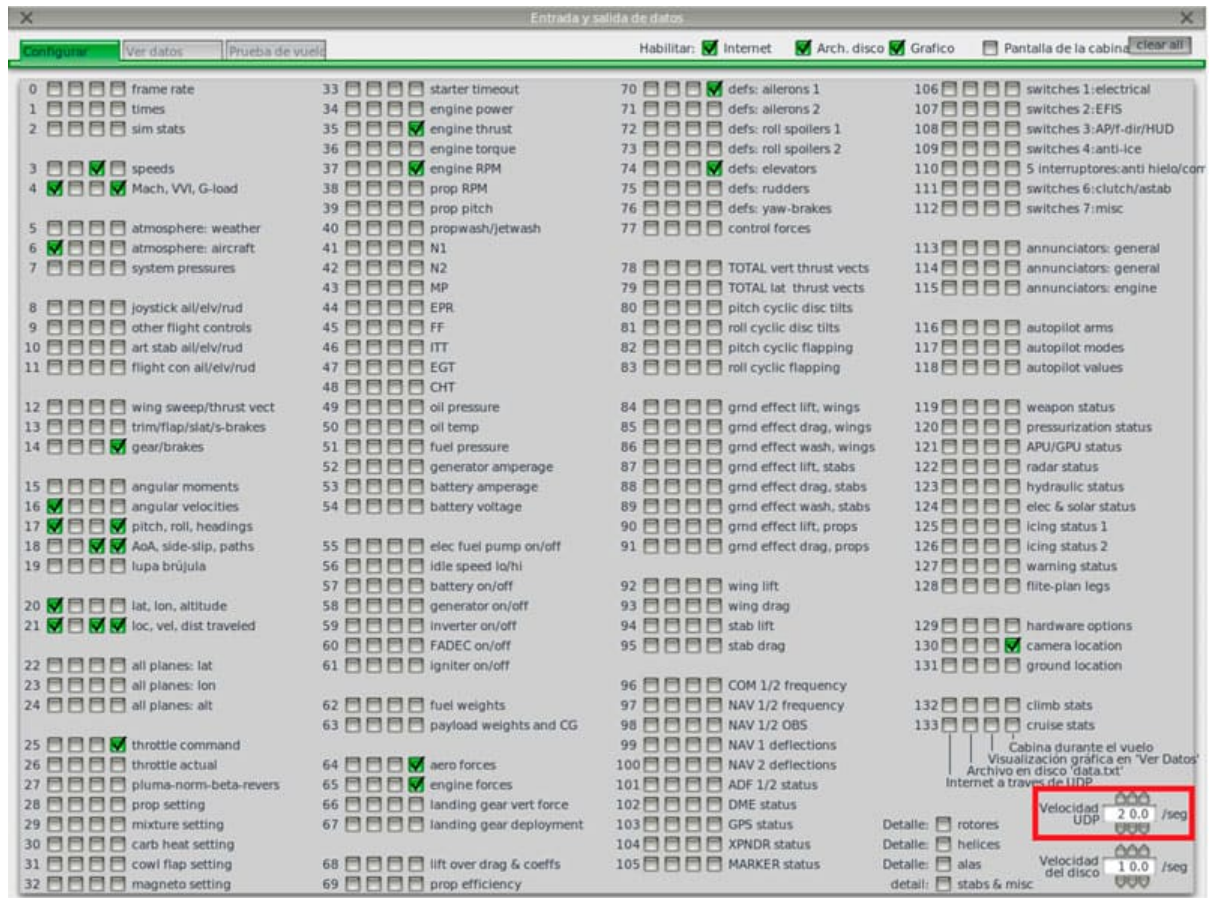


Quick Flight Setup

Data transmission settings must be edited on the settings tab. Select the input and output data option and edit the UDP speed. This speed must be set to **50/s**.



X-Plane 10 Setup



X-Plane 10 Input and Output Data

gSide Y acceleration – Body axis
 16 Angular Velocities pitchRate Pitch Rate
 rollRate Roll Rate

23 yawRate Yaw Rate
 17 Pitch, Roll, Headings pitch Pitch
 roll Roll

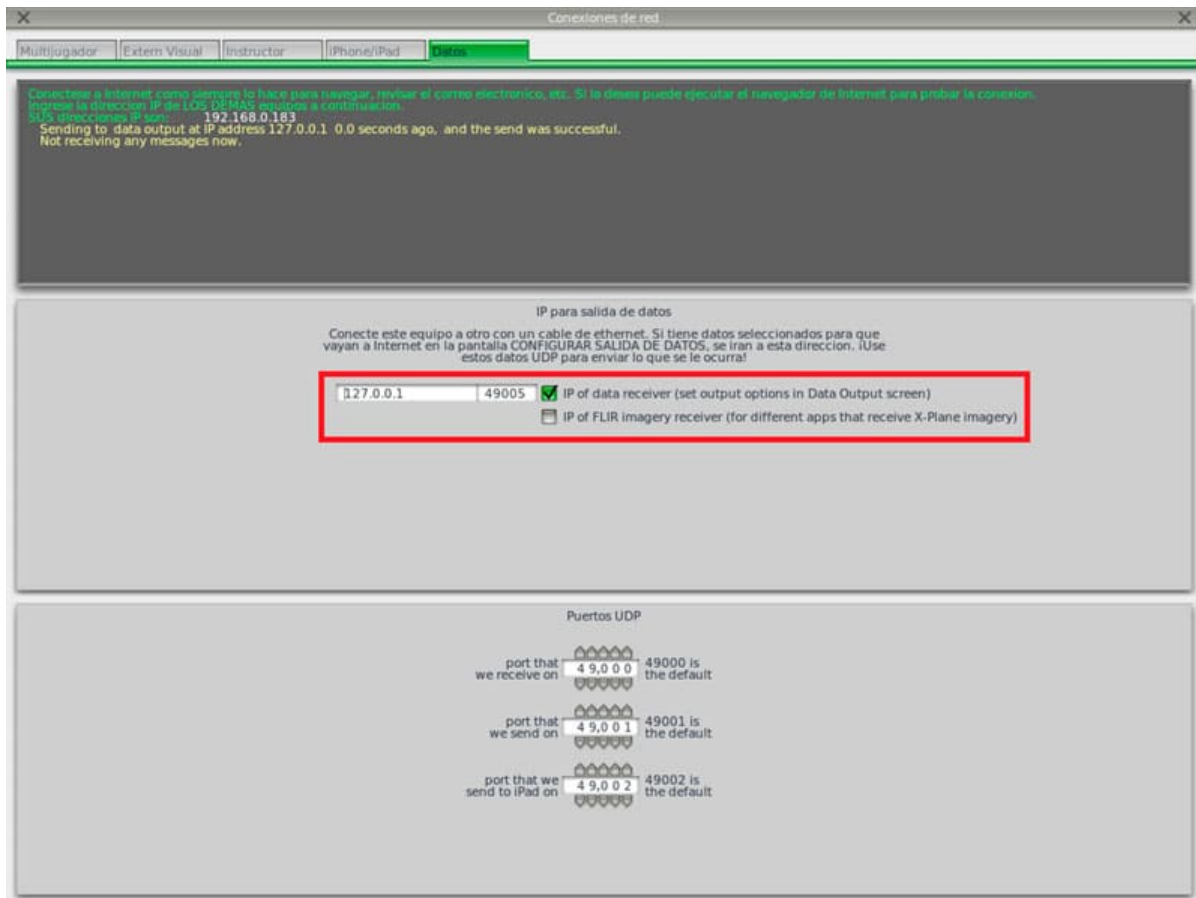
32 headingTrue Yaw
 18 AoA, side-slip, paths alpha used for IAS
 beta used for IAS
 20 lat,lon, altitude lat Latitude
 lon Longitude

| | | | |
|----------------------------|--|-----------|------------------------|
| | | 48 altMsl | Mean sea level |
| | | 48 altAgl | Above ground level |
| 21 loc, vec, dist traveled | | Vx | X velocity – Body Axis |
| | | Vy | Y velocity – Body Axis |
| | | 59 Vz | Z velocity – Body Axis |

11.1. Professional HIL

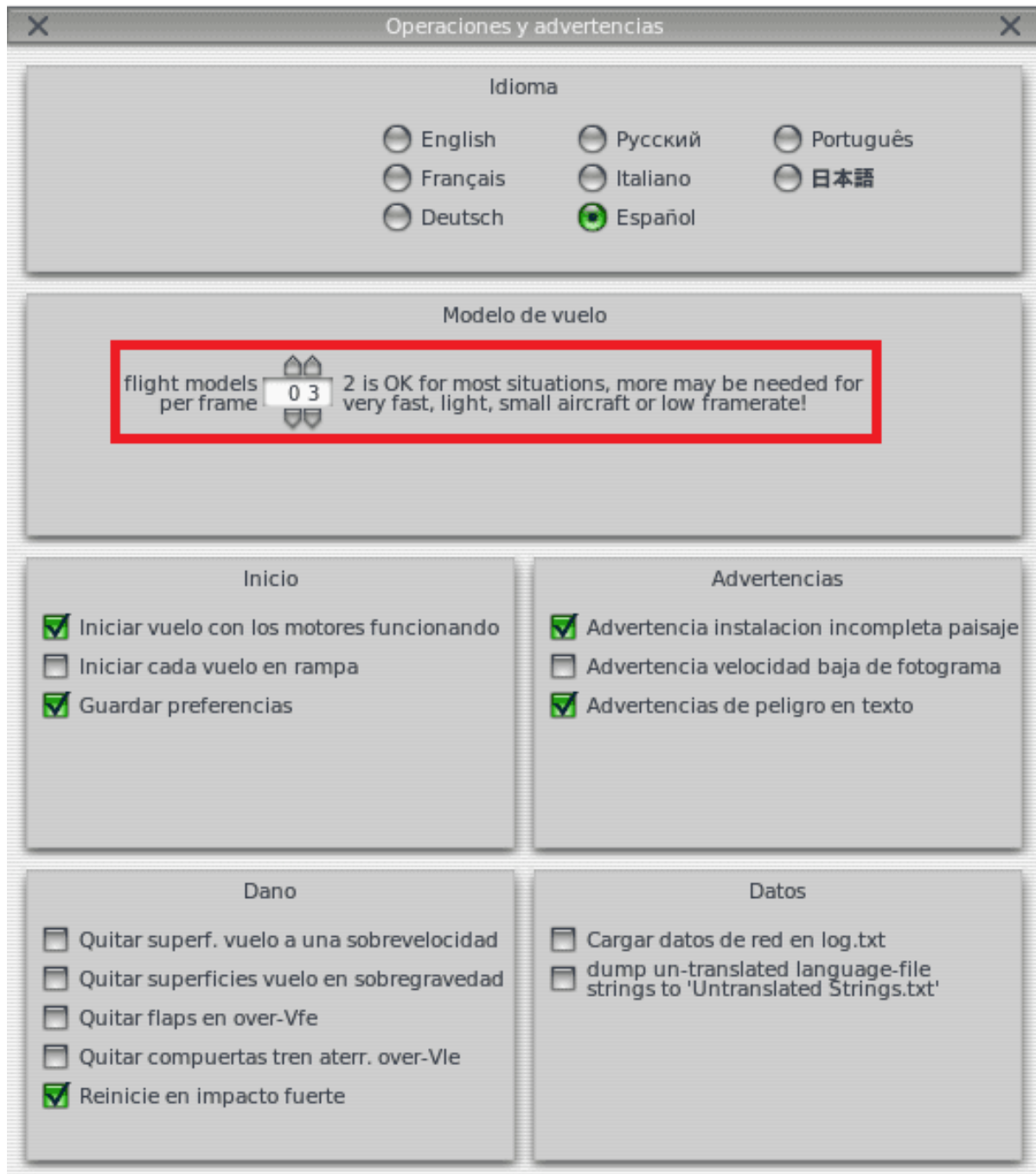
On the **settings** tab, enter into the **network configuration** and select the **data** menu. There user must edit the IP configuration as follows:

- IP: 127.0.0.1 49005
- Check the **IP of data receiver** option
- Uncheck the **IP of Flir Imagery receiver**



X-Plane 10 Network Configuration

On **settings** tab, **General options and warnings**, set the flight models per frame to a minimum of 3. It is needed for small aircraft simulation within X-Plane 10. If your model is vibrating in XPlane 10 you can increase this value but higher PC performances are required.

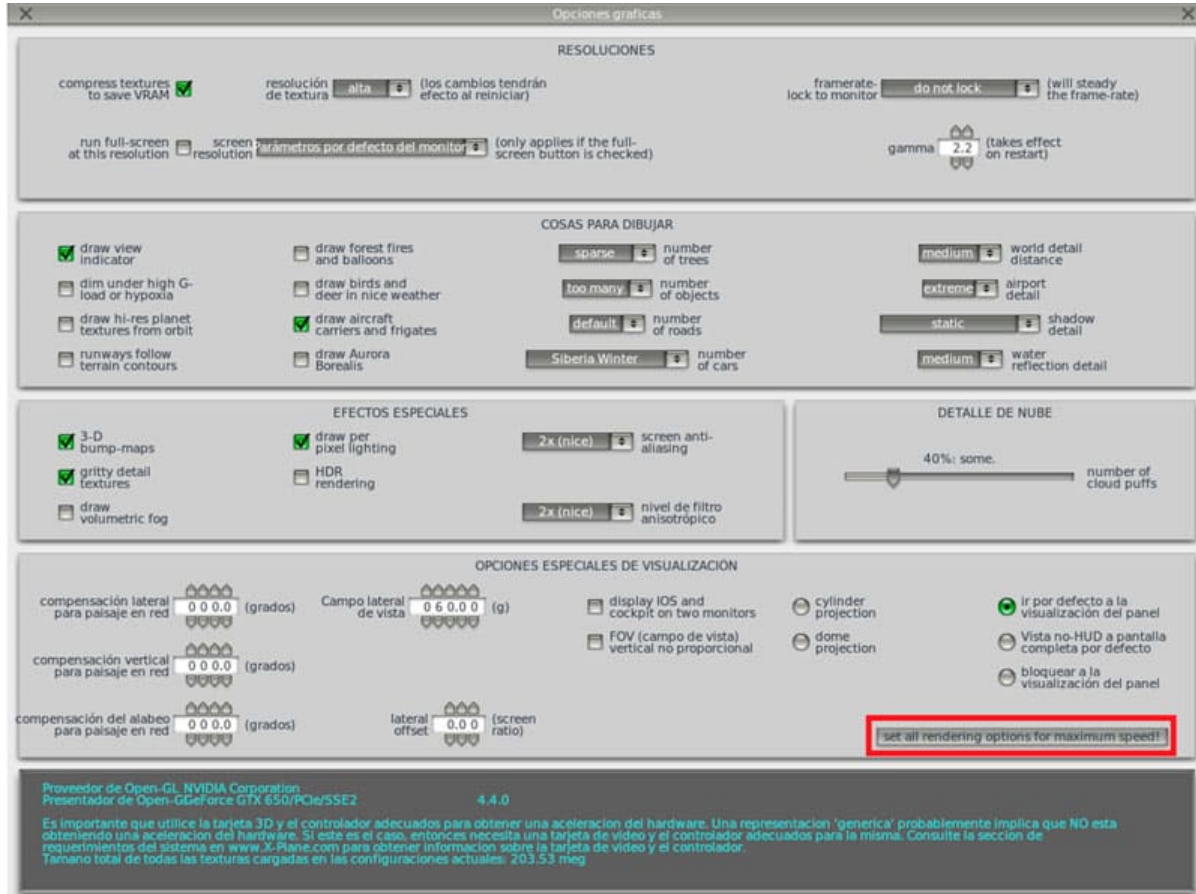


X-Plane 10 General Options and Warnings

11.1.2.2 Low Performance Computer Configuration

In case of using a low performance computer with the HIL Simulator the simulation reliability can decrease. In this case, it is recommended to reduce the graphic quality on the X-Plane 10 simulator.

On the **settings** tab, enter the **graphics option** menu and press on **set all rendering options for maximum speed**.



X-Plane 10 Graphics Option

11.1.3 X-Plane 11 Settings

11.1.3.1 X-Plane 11 Configuration

Since now, Veronte system is compatible with X-Plane 11 for HIL simulation. The demo version of the program can be downloaded from this [link](#).

X-Plane 11 communications settings must be edited in order to have communication with Veronte 4 system. Follow the next steps in order to make a proper configuration.

For low-performance computers, it may be needed to reduce the graphics quality on the simulator, as described below.

11.1.3.1.1 Aircraft Model Installation

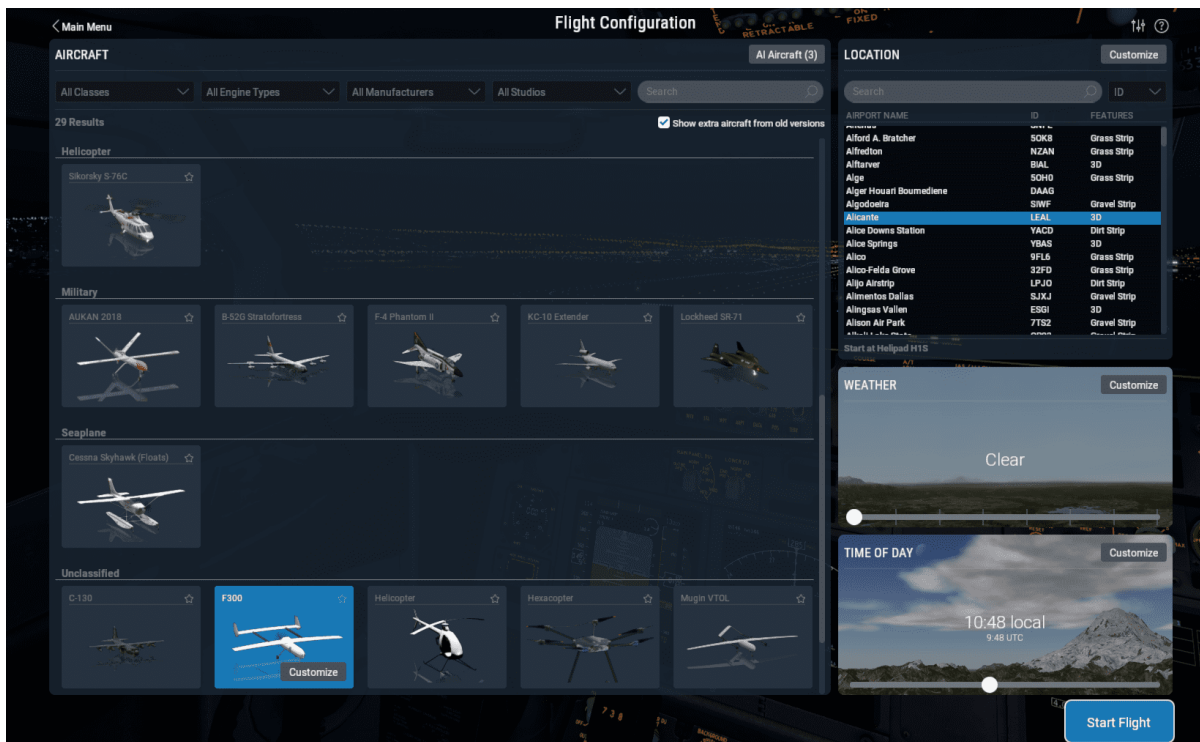
X-Plane 11 simulator is compatible with a wide variety of platforms: airplane, helicopter, multicopter, surface vehicle.... In order to create the platform model, **Plane Maker** tool provided by X-Plane 11 must be used.

Once the aircraft model has been created, it can be integrated on the X-Plane 11 simulator by following next steps:

- Copy the model folder to the “Aircraft” folder within the X-Plane 11 installation directory.
- Copy the content in the “Airfoils” folder, available on the aircraft model folder, to the “Airfoils” directory within the X-Plane 11 installation directory.

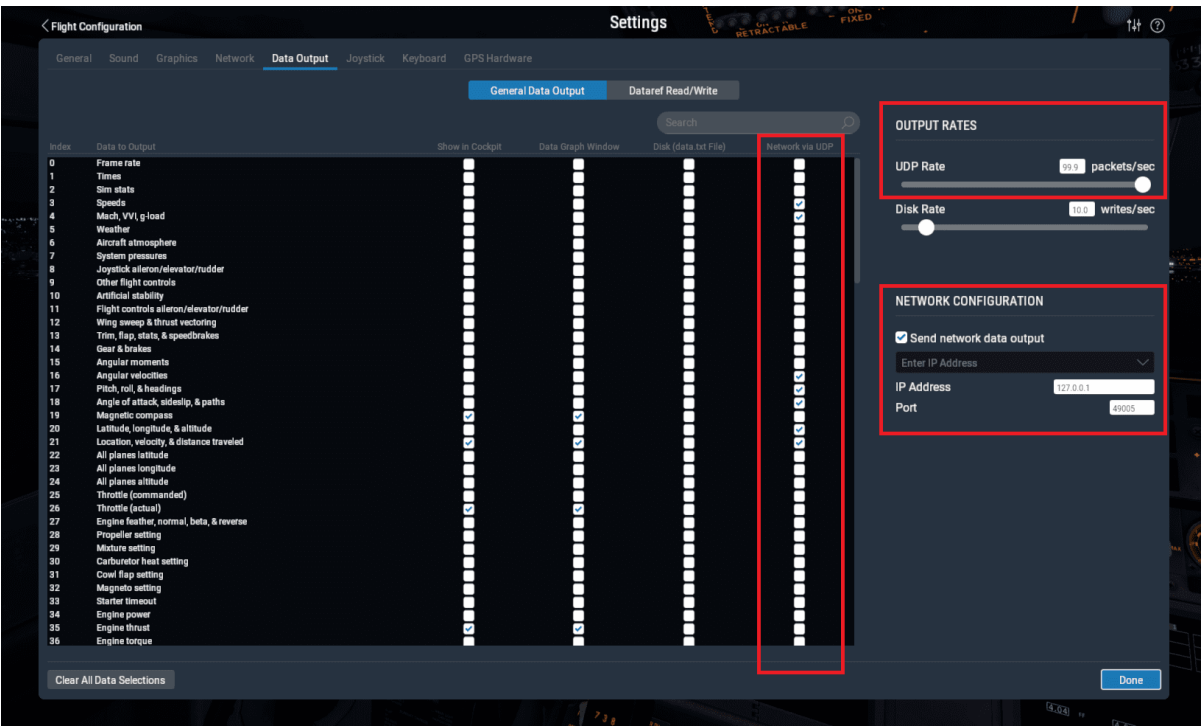
11.1.3.1.2 X-Plane 11 Setup

On X-Plane 11 execution, **Quick Flight Setup** window will be displayed; select which aircraft to use, the starting airport and weather conditions to be simulated during the flight.



Quick Flight Setup

Data transmission settings must be edited on the **Data Output** tab. Select all the variables in order to be sent through the UDP Network and set the UDP rate at maximum speed. The network configuration must be configured as shown in the following picture.



X-Plane 11 I/O configuration

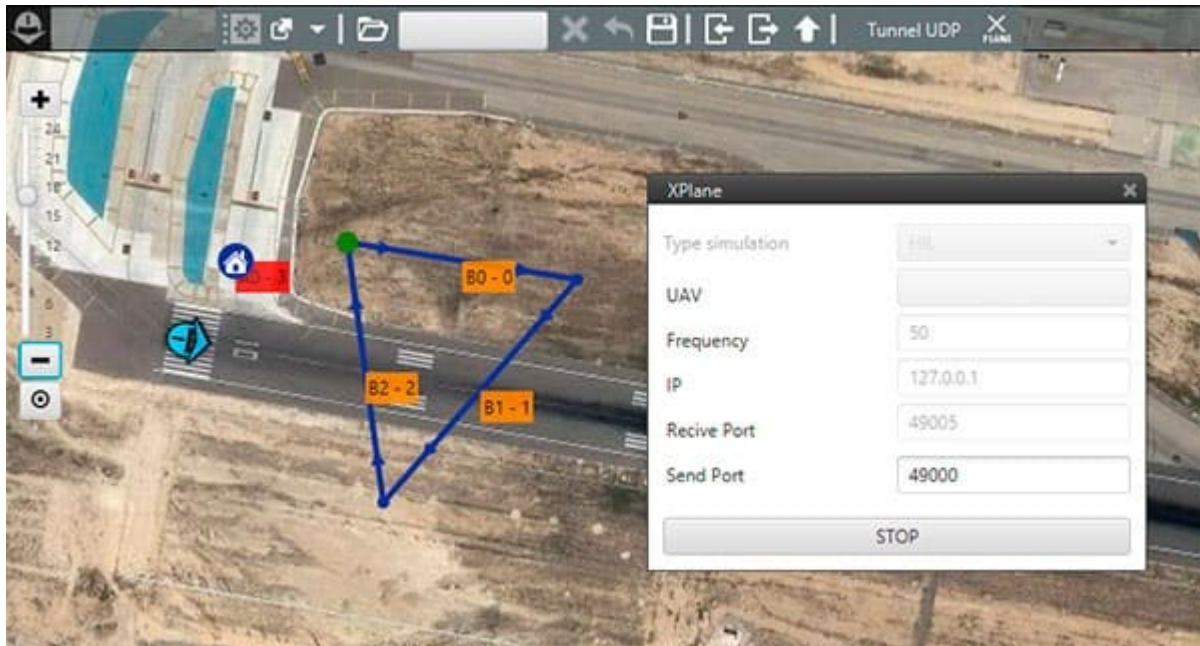
11.1.4 Operation

Once the hardware has been connected and Veronte Pipe and X-Plane have been configured, operation can start and the system can be operated as if it was on a real flight.



M400 Flight Simulation

When the X-Plane model is uploaded, GPS is simulated and the UAV must be visible on VerontePipe map. With GPS signal it is possible to pass the system to Standby phase and to start flight. X-Plane flight starts from an airport; a custom airport must be defined for simulating out of available airports.



Start GPS simulation

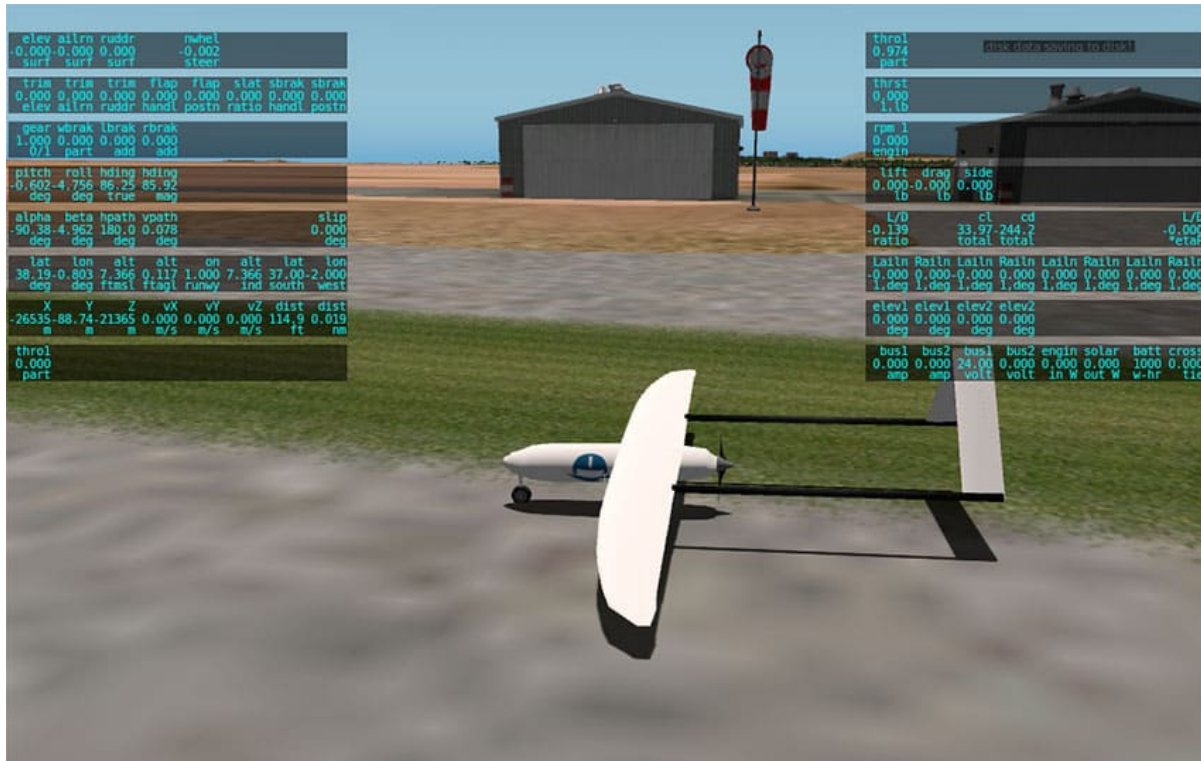
Moreover, it is possible to create a new airport. In order to do that, the user should follow the X-Plane tutorial presented in the next [link](#) . Once the aircraft has taken off from the airport on X-Plane the automatic control will start and the aircraft will fly to the defined mission on Veronte Pipe.

Warning: The flight is simulated in all real case aspects. Always make sure that the mission is well configured and check the terrain profile is in the correct position.

The operator can fly the system as on real flight, being compatible with main Veronte features: real-time mission edit, in-flight automatic to manual control, flight data recording... Sometimes is possible, during an edition saving, that the simulation fails because the simulation link suffers a little interruption (this fact does not exist in real flights).

When using the HIL simulator connected to the platform, control actuators will move as if it were flying. In order to avoid damaging the system or personnel, make sure that the motor is disconnected and there is no shock risk due to the actuators movement.

Warning: Always make sure that motor has been disconnected before starting a simulation. Otherwise the motor will run as if it were flying.



HIL Simulation

Veronte HIL simulator integrates Veronte Autopilot control system within the X-Plane simulator for highly realistic simulation within a safe environment.

Professional Hardware In the Loop (HIL) Simulator package is a powerful tool for Veronte Autopilot integration, development and operator training; permitting to extensively operate the system in a safe environment, prior to conducting real flight operations. Veronte HIL Simulator has been designed for accomplishing with requirements of most demanding unmanned system operators. Some of the main uses of the simulator are:

- Pilot training.
- Veronte configuration for unmanned platform control.
- PID setting.
- Mission configuration.
- Aircraft performance validation.

11.2 SIL Simulink

A software in the loop simulation consists of a Simulink model that simulates the behaviour of the system formed by the autopilot and a vehicle, without having the physical devices connected to the computer, in contrast to the HIL which has both the autopilot and (optionally) vehicle connected to the PC. This option has several advantages when it is compared with a HIL setup:

- Complete simulations without any hardware.
- Possibility of using your own vehicle model: no need to stick to XPlane models. You can add as much physics/complexity as desired.
- Possibility of simulating different kinds of sensors even if they are not fitted in Veronte.

- All results can be exported/visualized to MATLAB workspace simultaneously.
- Veronte Block runs faster than real time, allowing the user to execute a series of simulations in a short time.
- Light computational load.

11.2.1 Requisites

The following MATLAB Toolboxes are needed for a full exploitation of Veronte SIL capacities:

- **MATLAB + Simulink** (basic package).
- **Aerospace toolbox**: contains sensor blocks, flight instruments and environment blocks.
- **Simulink Real-Time**: contains useful blocks to be used with buses: UDP/RS232/CAN...

In addition, it is required to install and select Microsoft Visual Studio 2015 (or later) as your MEX compiler.

1. First, get Microsoft Visual Studio from [here](#).
2. Follow the onscreen steps, please make sure that C++ tools are selected (they may appear as an optional item).
3. When finished, select it as your default MEX compiler by typing in MATLAB console **mex -setup c++**.
4. If everything is correct, it should be configured as shown:

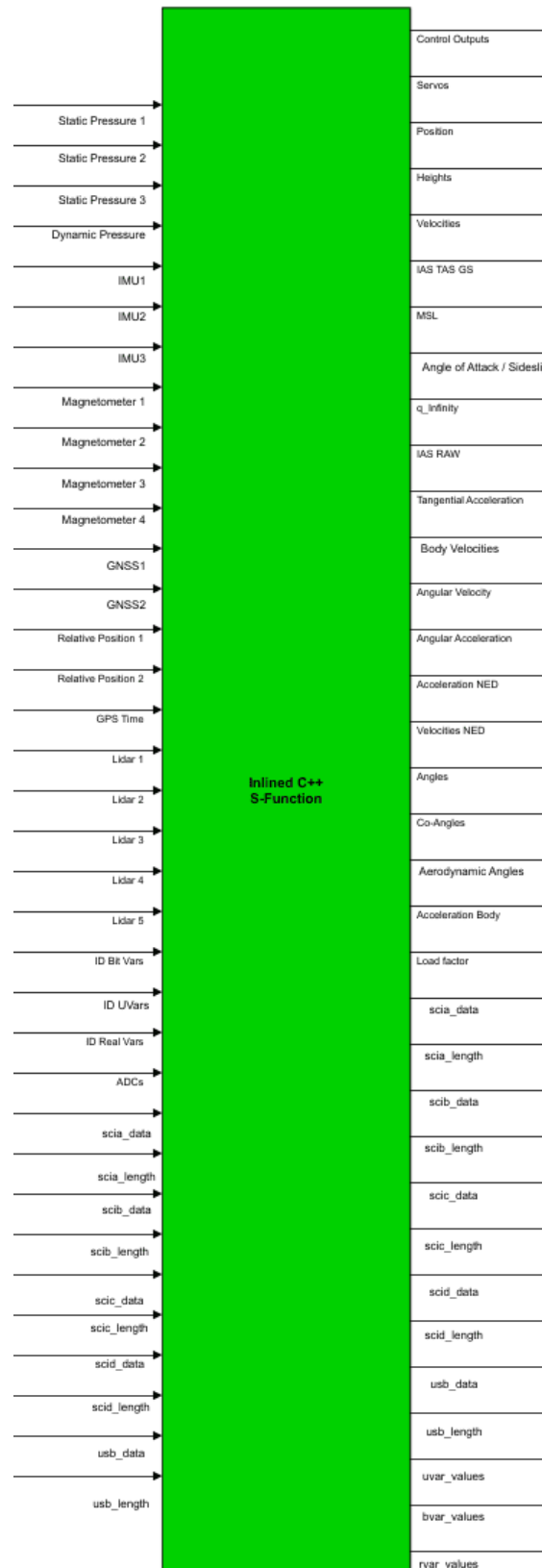
```
MEX configured to use 'Microsoft Visual C++ 2015' for C++ language compilation.
Warning: The MATLAB C and Fortran API has changed to support MATLAB
variables with more than 2^32-1 elements. You will be required
to update your code to utilize the new API.
You can find more information about this at:
https://www.mathworks.com/help/matlab/matlab\_external/upgrading-mex-files-to-use-64-bit-api.html.

To choose a different C++ compiler, select one from the following:
MinGW64 Compiler (C++) mex -setup:'C:\Program Files\MATLAB\R2018a\bin\win64\mexopts\mingw64_g++.xml' C++
MinGW64 Compiler with Windows 10 SDK or later (C++) mex -setup:'C:\Program Files\MATLAB\R2018a\bin\win64\mexopts\mingw64_g++_sdk10+.xml' C++
Microsoft Visual C++ 2015 mex -setup:C:\Users\mfg\AppData\Roaming\MathWorks\MATLAB\R2018a\mex_C++_win64.xml C++
```

Visual C++ 2015 Selected as Compiler for MEX functions.

11.2.2 Autopilot Simulation

The autopilot is implemented in Simulink with an S-Function. This kind of block takes a C, C++, Fortran or even Matlab code, and implements it in a block containing a certain number of inputs and outputs. A typical Veronte s-function is shown below.



S-Function containing the autopilot embedded code

Inputs are described in the next table:

| PIN | Signal Type | Description | Form | Size | Units |
|-----|-------------|---------------------|--|------|----------|
| 1 | Input | Static Pressure 1 | [pressure_measurement; alt; sensor temperature] | 1x1 | Pa / K |
| 2 | Input | Static Pressure 2 | [pressure_measurement; alt; sensor temperature] | 1x1 | Pa / K |
| 3 | Input | Static Pressure 3 | [pressure_measurement; alt; sensor temperature] | 1x1 | Pa / K |
| 4 | Input | Dynamic Pressure | [pressure_measurement; alt; sensor temperature] | 1x1 | Pa / K |
| 5 | Input | IMU 1 | [acc_x;acc_y;acc_z;gyr_x;gyr_y;gyr_z;temp] | 7x1 | m/s / K |
| 6 | Input | IMU 2 | [acc_x;acc_y;acc_z;gyr_x;gyr_y;gyr_z;temp] | 7x1 | m/s / K |
| 7 | Input | IMU 3 | [acc_x;acc_y;acc_z;gyr_x;gyr_y;gyr_z;temp] | 7x1 | m/s / K |
| 8 | Input | Magnetometer 1 | [mag_x;mag_y;mag_z;temp] | 4x1 | T |
| 9 | Input | Magnetometer 2 | [mag_x;mag_y;mag_z;temp] | 4x1 | T |
| 10 | Input | Magnetometer 3 | [mag_x;mag_y;mag_z;temp] | 4x1 | T |
| 11 | Input | Magnetometer 4 | [mag_x;mag_y;mag_z;temp] | 4x1 | T |
| 12 | Input | GNSS 1 | [1;3;lon;lat;alt;hor_acc;vert_acc;v_north;v_east;v_down;v_acc] | 10x1 | m / mm/s |
| 13 | Input | GNSS 2 | [1;3;lon;lat;alt;hor_acc;vert_acc;v_north;v_east;v_down;v_acc] | 10x1 | m / mm/s |
| 14 | Input | Relative Position 1 | [1;x_rel;y_rel;z_rel;distance_x;distance_y;distance_z;acc_x;acc_y;acc_z] | 10x1 | m |
| 15 | Input | Relative Position 2 | [1;x_rel;y_rel;z_rel;distance_x;distance_y;distance_z;acc_x;acc_y;acc_z] | 10x1 | m |
| 16 | Input | GPS Time | [week_number;seconds_of_week] | 2x1 | s |
| 17 | Input | Lidar 1 | [lidar_measurement] | 1x1 | cm |
| 18 | Input | Lidar 2 | [lidar_measurement] | 1x1 | cm |
| 19 | Input | Lidar 3 | [lidar_measurement] | 1x1 | cm |
| 20 | Input | Lidar 4 | [lidar_measurement] | 1x1 | cm |
| 21 | Input | Lidar 5 | [lidar_measurement] | 1x1 | cm |
| 22 | Input | ID Bit Var | [Var_IDs] | 50x1 | m |
| 23 | Input | ID Unsigned Var | [Var_IDs] | 50x1 | m |
| 24 | Input | ID Real Var | [Var_IDs] | 50x1 | m |
| 25 | Input | ADCs | [adc(1-17)] | 17x1 | . |

continues on next page

Table 1 – continued from previous page

| PIN | Signal Type | Description | Form | Size | Units |
|-----|-------------|-------------|-----------------|--------|-------|
| 26 | Input | SCIA Data | [serial_data] | 1024x1 | • |
| 27 | Input | SCIA Length | [serial_length] | 1x1 | • |
| 28 | Input | SCIB Data | [serial_data] | 1024x1 | • |
| 29 | Input | SCIB Length | [serial_length] | 1x1 | • |
| 30 | Input | SCIC Data | [serial_data] | 1024x1 | • |
| 31 | Input | SCIC Length | [serial_length] | 1x1 | • |
| 32 | Input | SCID Data | [serial_data] | 1024x1 | • |
| 33 | Input | SCID Length | [serial_length] | 1x1 | • |
| 34 | Input | USB Data | [serial_data] | 1024x1 | • |
| 35 | Input | USB Length | [serial_length] | 1x1 | • |

Outputs are the following:

| PIN | Signal Type | Description | Form Size | Units |
|-----|-------------|----------------------------|---|----------------|
| 1 | Output | Control Outputs | [control_outputs(1-20)] 20x1 | • |
| 2 | Output | Servo Values | [servos(1-32)] 32x1 | • |
| 3 | Output | Position | [lat;lon;alt] | 3x1 rad / m |
| 4 | Output | Heights | [msl,agl] | 2x1 m |
| 5 | Output | Velocities | [longitudinal_v; lateral_v; velocity(module)] | 3x1 m/s |
| 6 | Output | IAS TAS GS | [ias,tas,gs] | 3x1 m/s |
| 7 | Output | MSL | [msl_from_qnh;msl_from_ISA] | 2x1 m |
| 8 | Output | Angle of Attack / Sideslip | [angle_of_attack;sideslip] | 2x1 rad |
| 9 | Output | Q_Infinity | [dynamic_pressure] | 1x1 Pa |
| 10 | Output | IAS RAW | [ias_raw] | 1x1 m/s |

continues on next page

Table 2 – continued from previous page

| PIN | Signal Type | Description | Form Size | Units |
|-----|-------------|-------------------------|------------------------------------|--------|
| 11 | Output | Tangential Acceleration | [tangential_acceleration] | m/s^2 |
| 12 | Input | Body Velocities | [lon_v;lat_v;vertical_v] | m/s |
| 13 | Output | Angular Velocities | [roll_rate;pitch_rate;yaw_rate] | rad/s |
| 14 | Output | Angular Acceleration | [acc_z_axis;acc_y_axis;acc_x_axis] | rad/^2 |
| 15 | Output | Acceleration NED | [acc_north;acc_east;acc_down] | m/s^2 |
| 16 | Output | Velocity NED | [v_north;v_east;v_down] | m/s |
| 17 | Output | Angles | [Yaw;Pitch;Roll] | rad |
| 18 | Output | Co-Angles | [co-Yaw;co-Pitch;co-Roll] | rad |
| 19 | Output | Aerodynamic Angles | [heading,flight_path,bank_angle] | rad |
| 20 | Output | Acceleration Body | [acc_x,acc_y,acc_z] | m/s^2 |
| 21 | Output | Load factor | [nx;ny;nz] | . |
| 22 | Output | SCIA Data | [serial_data] | . |
| 23 | Output | SCIA Length | [serial_length] | . |
| 24 | Output | SCIB Data | [serial_data] | . |
| 25 | Output | SCIB Length | [serial_length] | . |
| 26 | Output | SCIC Data | [serial_data] | . |
| 27 | Output | SCIC Length | [serial_length] | . |
| 28 | Output | SCID Data | [serial_data] | . |
| 29 | Output | SCID Length | [serial_length] | . |
| 30 | Output | USB Data | [serial_data] | . |

continues on next page

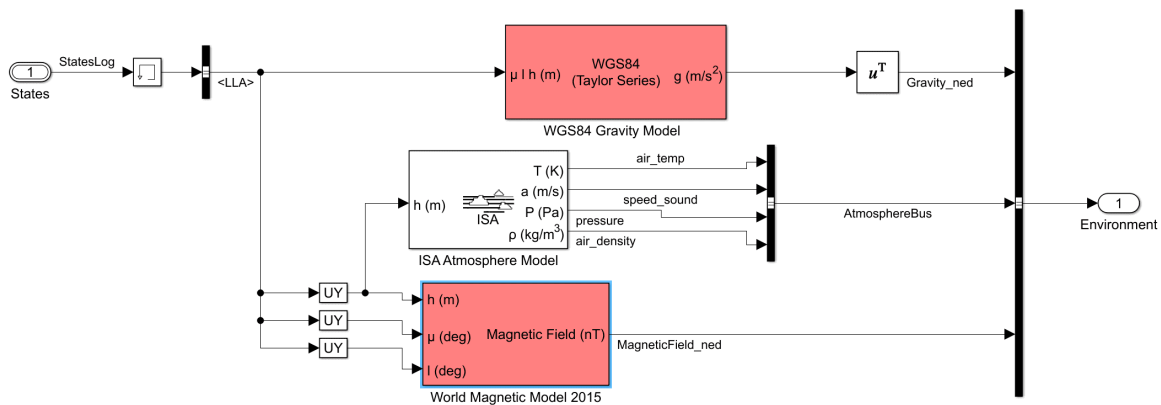
Table 2 – continued from previous page

| PIN | Signal Type | Description | Form Size | | Units |
|-----|-------------|--------------------|----------------------------|------|-------|
| 31 | Output | USB Length | [serial_length]) | 1x1 | • |
| 32 | Output | Unsigned Variables | [selected variables(1-50)] | 50x1 | • |
| 33 | Output | Bit Variables | [selected variables(1-50)] | 50x1 | • |
| 34 | Output | Real Variables | [selected variables(1-50)] | 50x1 | • |

11.2.3 Sensors Simulation & Input Examples

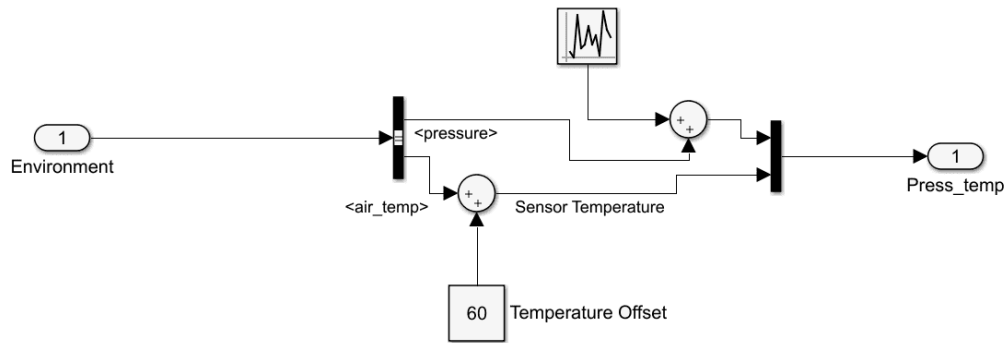
This section aims to illustrate how to implement the inputs described in the previous section. The structures that are shown here are orientative and, of course, can be adapted by the user.

A basic subsystem that must be built in every flight simulation is an environment model. This model, groups the atmospherical properties and it changes according to different variables as well as the magnetic field at certain coordinates on earth. This block will be the basis of most the sensors that will be shown later on:



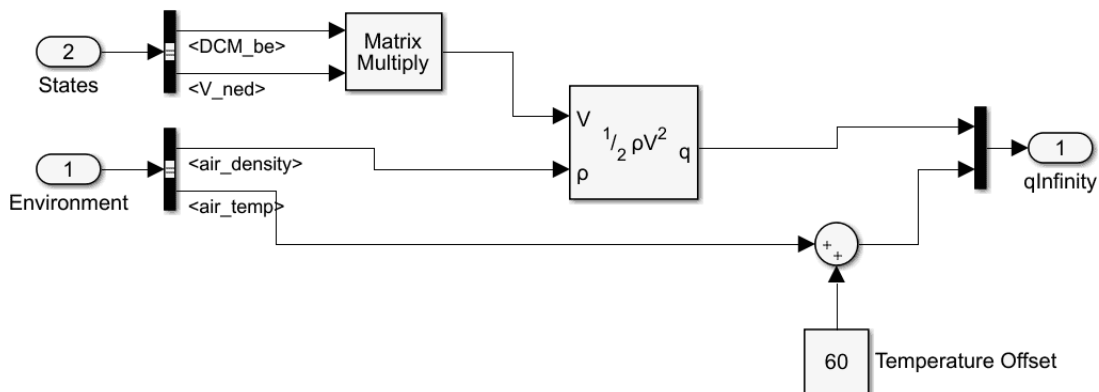
Environment Example Block

The static pressure sensor block can be easily derived by taking the pressure from the environment model. The only parameter that must be added is the temperature of the sensor (in this case is the OAT + 60):



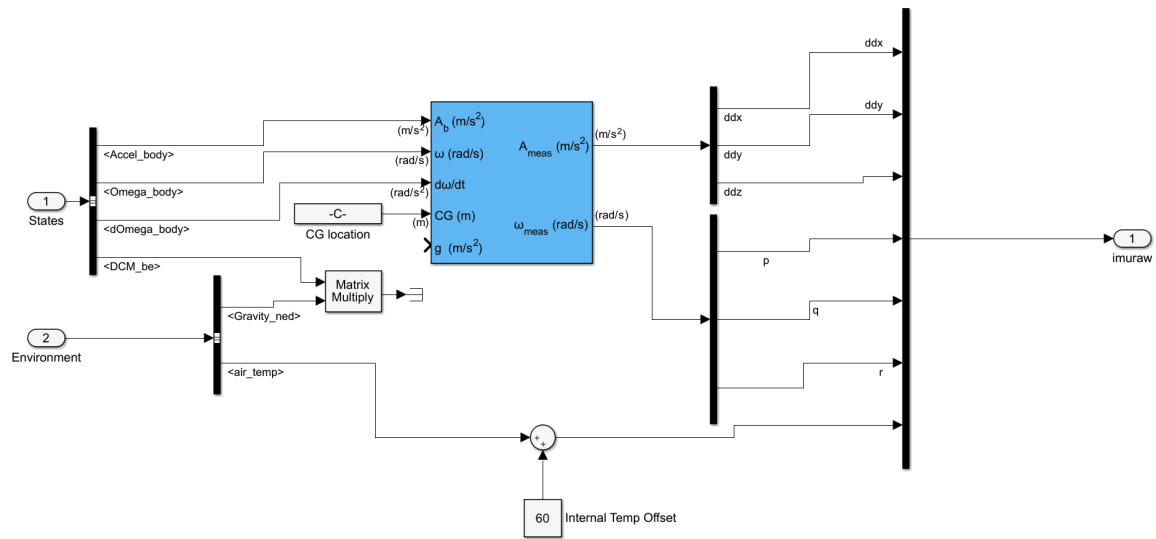
Static Pressure Example Block

The dynamic pressure input can be also obtained by using the standard dynamic pressure simulink block. The inputs are the speed (body axis) and the density:



Dynamic Pressure Example Block

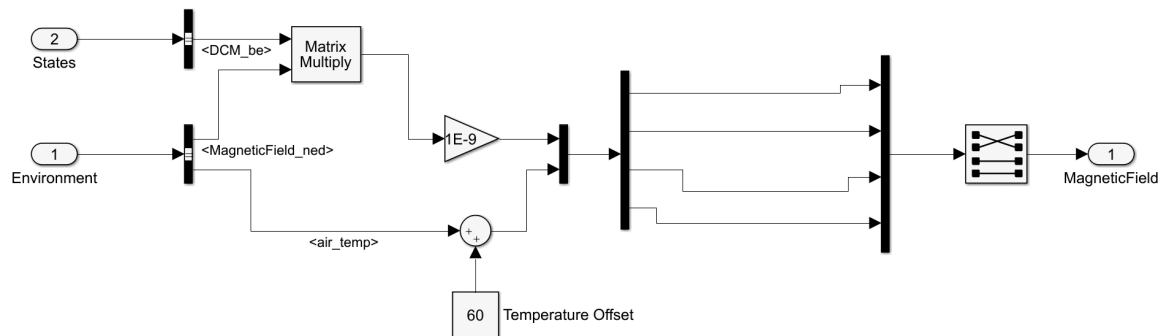
There is also a dedicated simulink block in the aerospace blockset that models a set of accelerometers and gyroscopes. The inputs of this block are the acceleration and angular velocities coming out of dynamics of the vehicle.



IMU Example Block

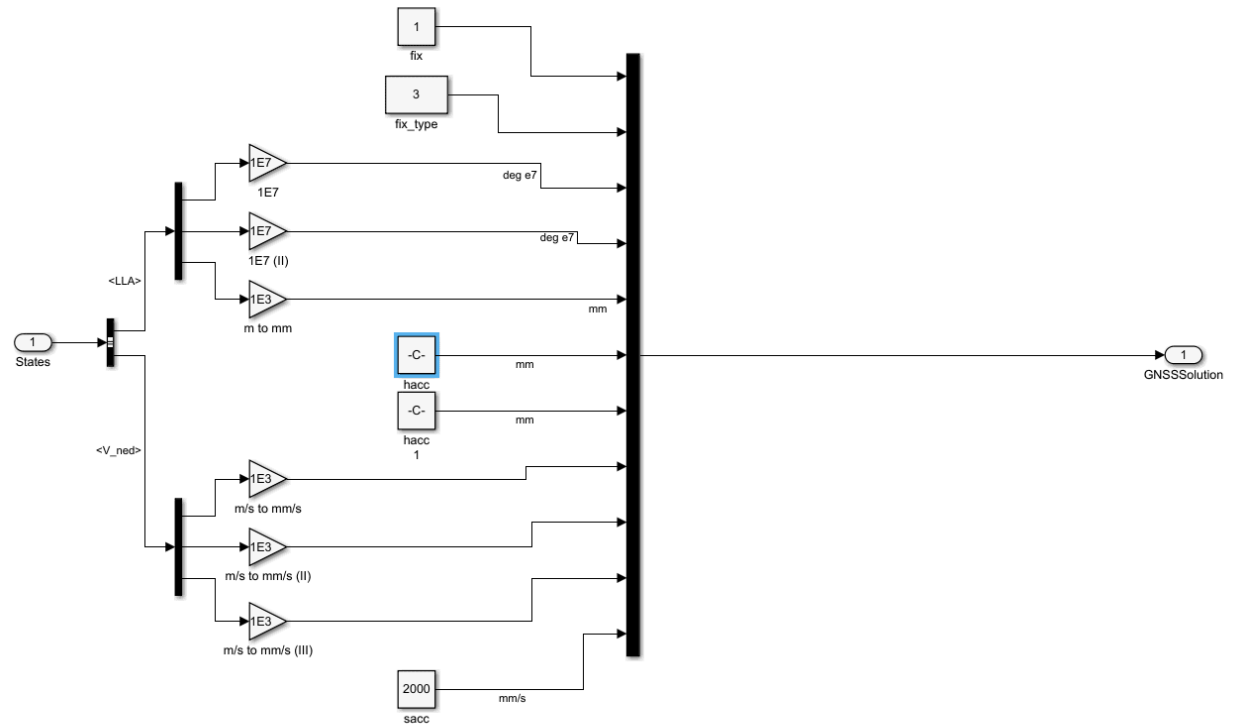
Warning: Do not connect the gravity port to the IMU block.

The magnetometer block is simply a rotated environment magnetic field where the temperature of sensor has been added (same as before OAT + 60). The reason why there is a selector crossing the signals is because there is a rotation matrix pre-configured in each PDI (the real magnetometer is not aligned with the autopilot axis):

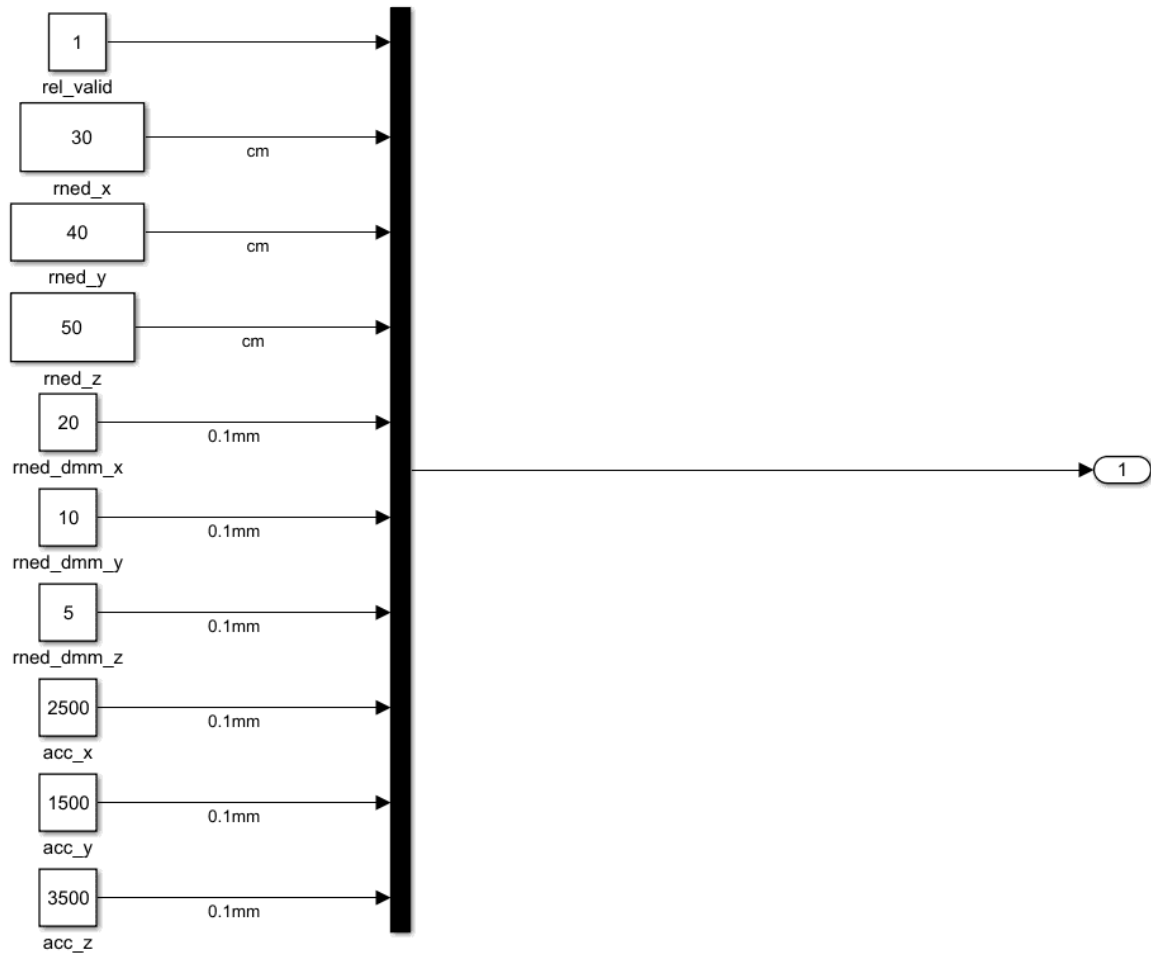


Magnetometer Example Block

The GNSS receiver and the relative position input (RTK) are only a multiplexor that creates an array. The position and the velocity are outputs of the vehicle model:

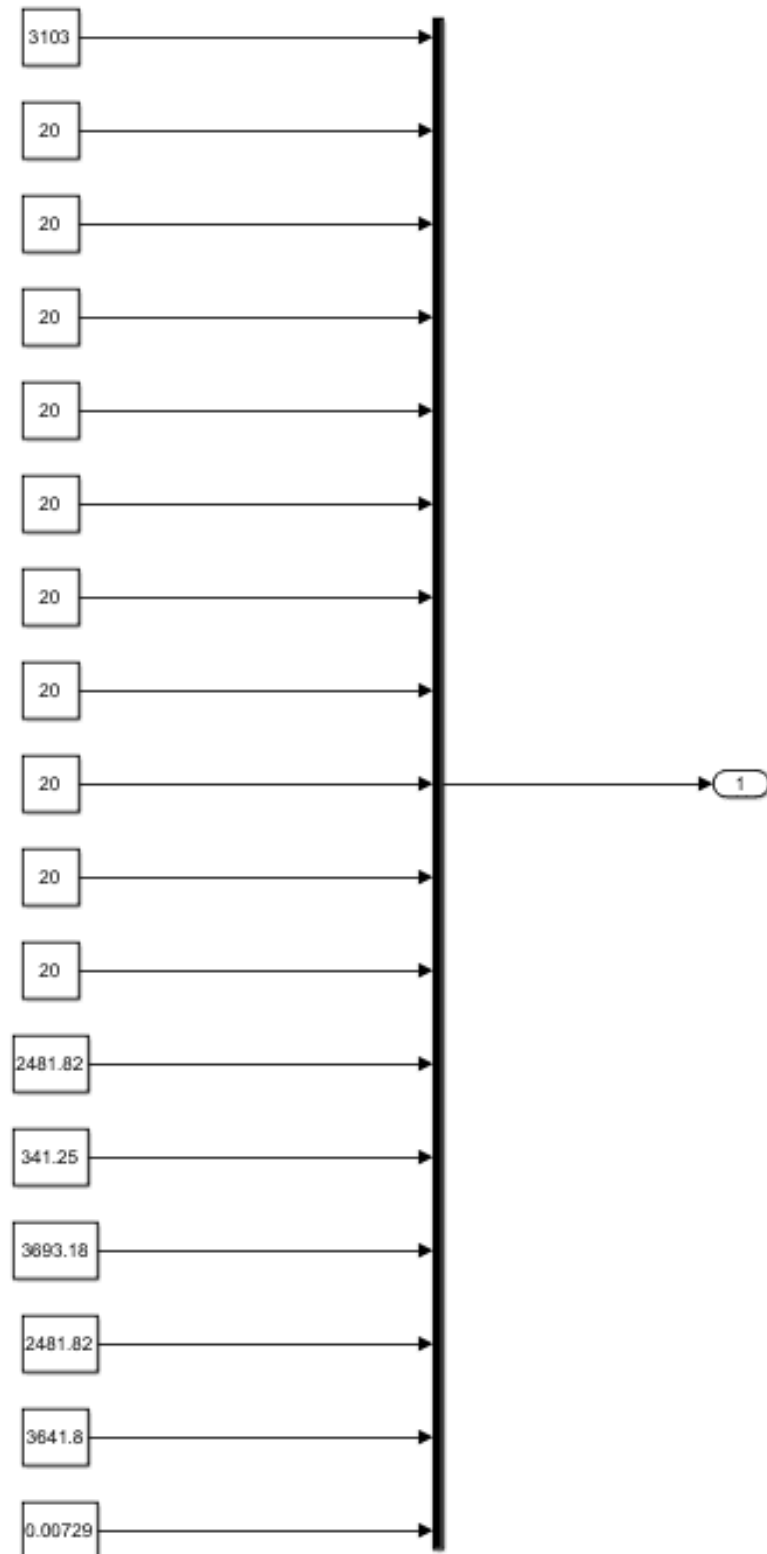


GNSS Receiver Example Block



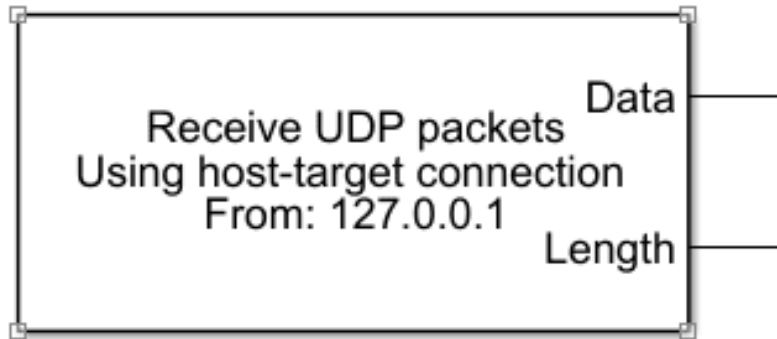
Relative Position (RTK) Example Block

The analogue inputs follow the same reasoning, the user must add the desired values to the mux:



ADCs Example Block

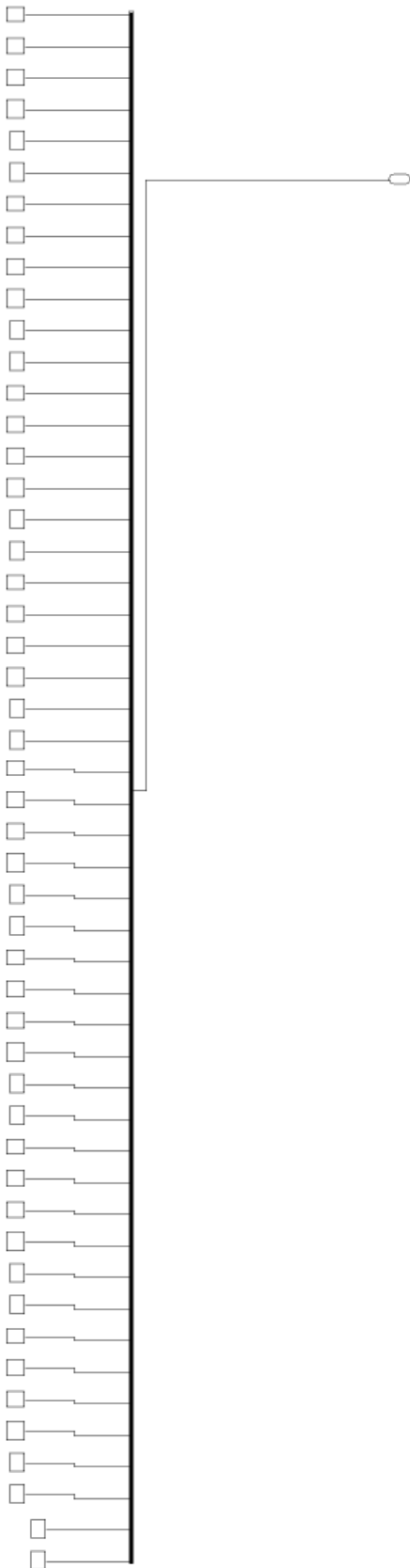
Veronte can manage input and output serial ports as explained [here](#). An easy way to create serial frames (data en length wires) is by using the simulink UDP block. Therefore, the data coming in to veronte should be sent though UDP (if this approach is taken):



SCI Example Block

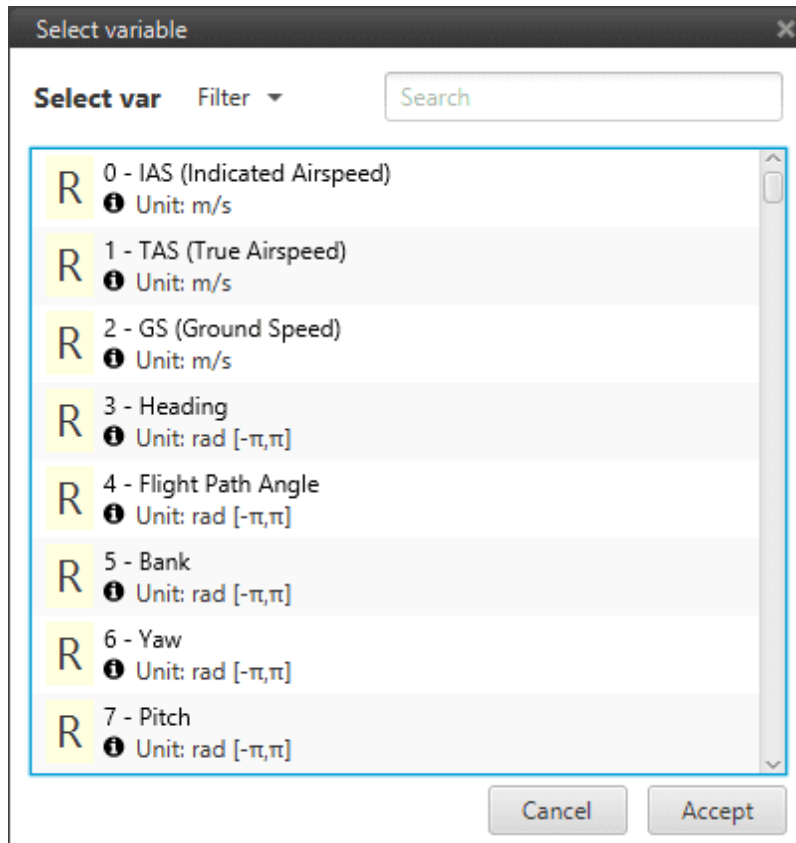
11.2.4 Monitoring Telemetry with Simulink

As explained before, there are three inputs specially dedicated to select custom telemetry (pins 22,23 and 24). The input structure of those is fixed and must be of size 50, as illustrated here:



Telemetry ID Mux

The ID of each variable available in Veronte can be easily found in Veronte Pipe by adding a new workspace widget. The ID is labelled right before the variable name:

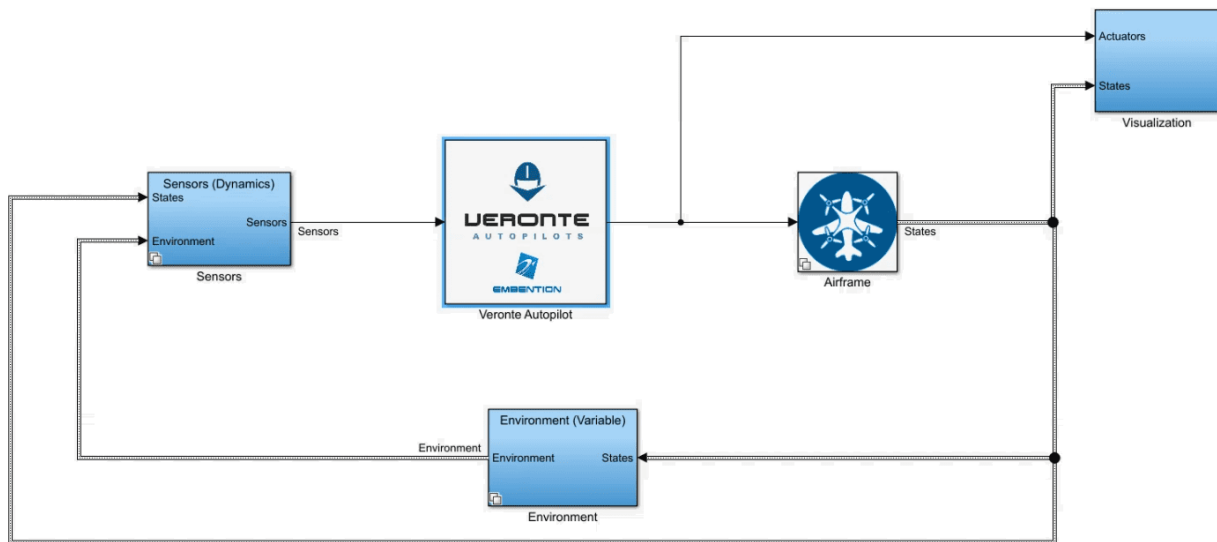


ID indicator

Lastly, to see their values, just place a *scope* connected to the matching output (pins 32, 33 or 34).

11.2.5 Complete Simulation

After setting the main blocks, the result should look like this:

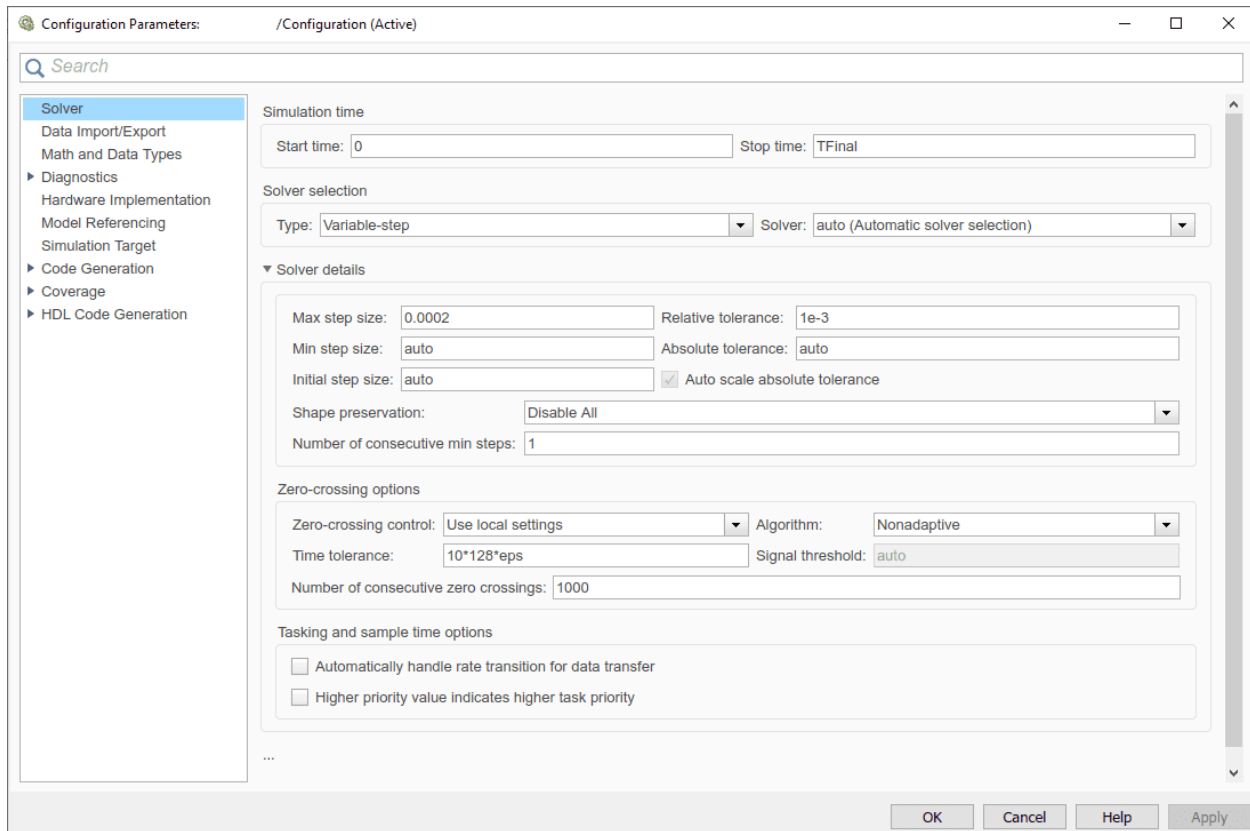


Complete Setup Example

The main systems are:

- **Veronte Autopilot:** It contains our flight control software.
- **Airframe:** a model of the flight dynamics.
- **Environment:** a model of the atmosphere, magnetic field, WGS84...
- **Sensors:** it contains individual blocks of all the sensors that veronte needs as input.
- **Visualization:** It contains, scopes, flight instruments...

The time step should be set to 0.0002 as shown in the next figure in order to guarantee a good GNC/Adquisition frequency:

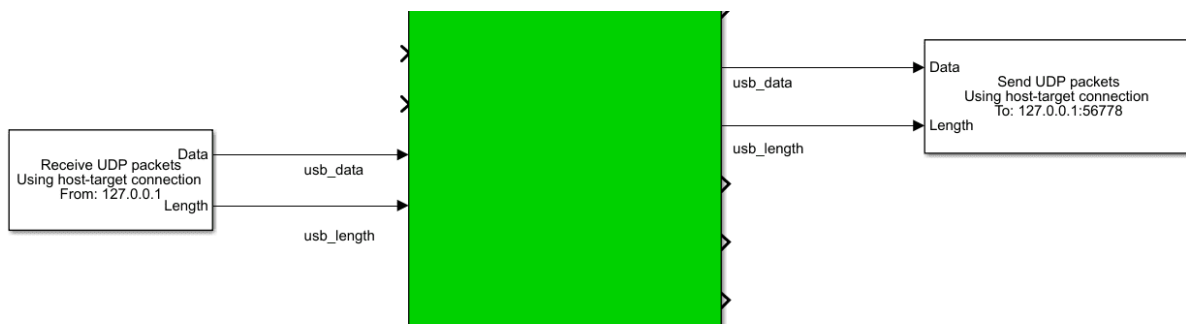


Time step settings

11.2.6 Connecting Simulink & Veronte Pipe

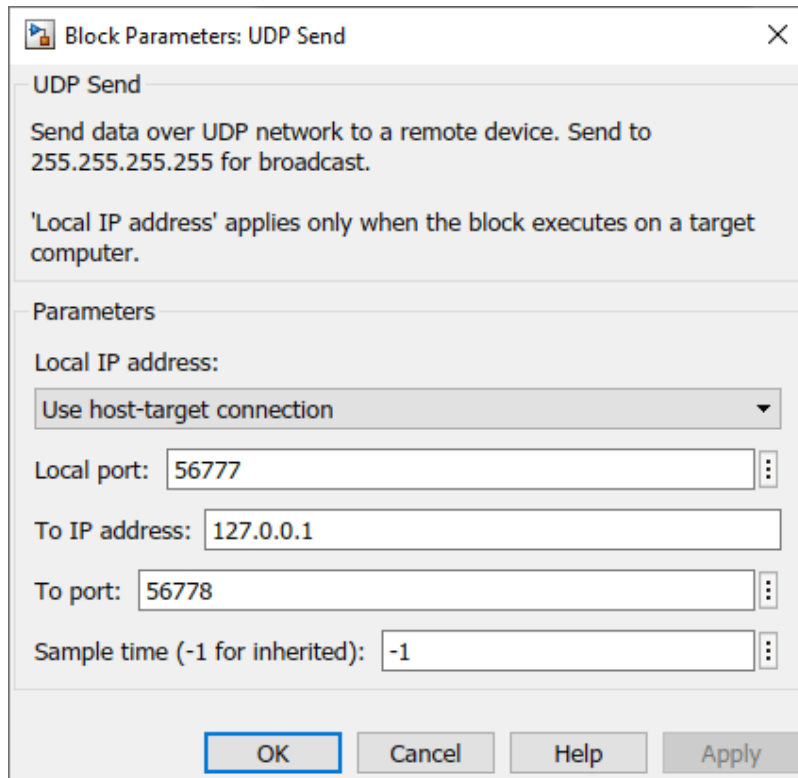
Our Software-in-the-loop simulator can be connected and used alongside Veronte Pipe software. To do it, Follow this steps:

1. Add a UDP serial communication block and connect it to USB data and length.
2. Add a second UDP serial communication block and connect it to the USB output of veronte.



UDP Blocks

3. Configure your destination port.



The image shows a dialog box titled "Block Parameters: UDP Send". It contains the following sections:

- UDP Send**: Send data over UDP network to a remote device. Send to 255.255.255.255 for broadcast.
'Local IP address' applies only when the block executes on a target computer.
- Parameters**:
 - Local IP address: A dropdown menu with "Use host-target connection" selected.
 - Local port: A text input field containing "56777".
 - To IP address: A text input field containing "127.0.0.1".
 - To port: A text input field containing "56778".
 - Sample time (-1 for inherited): A text input field containing "-1".

At the bottom, there are four buttons: "OK", "Cancel", "Help", and "Apply".

Destination UDP Port

4. Set an ethernet network in Preferences as shown using the destination port selected before.

The screenshot shows the 'Preferences' window for the Veronte Autopilot. It is divided into two main sections: 'TCP Server' and 'Send telemetry'.

TCP Server Section:

- ☐ Enable
- Port: 3000
- ☐ Autodiscover COMs
- Under 'Ethernet', there is a list of network interfaces. At the bottom of this list is an 'Add' button.

Send telemetry Section:

- ☐ Enable
- Host: 127.0.0.1
- Port: 3000
- Frequency: 10.0 Hz
- Multicast IP: 239.0.0.1
- Port: 56778 (highlighted with a blue border)
- Network Interface: Qualcomm Atheros AR8171/8175 PCI-E Gigabit Ethe... (dropdown menu)
- Local IP Address: 0.0.0.0
- Local Subnet Mask: 0.0.0.0

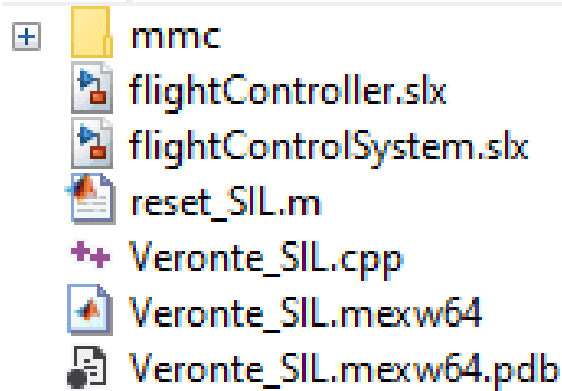
Destination UDP Port (Pipe)

11.2.7 Dealing with PDI files

PDI are uploaded exactly the same way that would be uploaded to a physical autopilot. Once connected to Veronte Pipe, a *virtual* autopilot will be detected in safe mode, then these [steps](#) can be followed.

Warning: Restart functionalities do no restart the simulation and must be done manually.

Veronte S-function deals with a direct copy of the binary files of a veronte autopilot, if a manual setup is needed, these files will be found in the same folder as the simulink block in a subfolder called *mmc*:



Folder structure (matlab)

11.3 3D Simulation



Veronte 3D Simulation

Veronte PFD works using a flight simulator for representing the worldwide geographical scenarios: *lands, seas, mountains, cities, airports, airfields, heliports, etc...* In addition, unlike other 3D PFDs an internet connection is not necessary so it can be operated from any location and environment without any delays in scenario loading.

This feature displays a 3D view of the aircraft which is being piloted, while it permits to use it as a 3D PFD (Primary Flight Display) when using the 1st camera view. This system permits to display custom aircraft models in the virtual

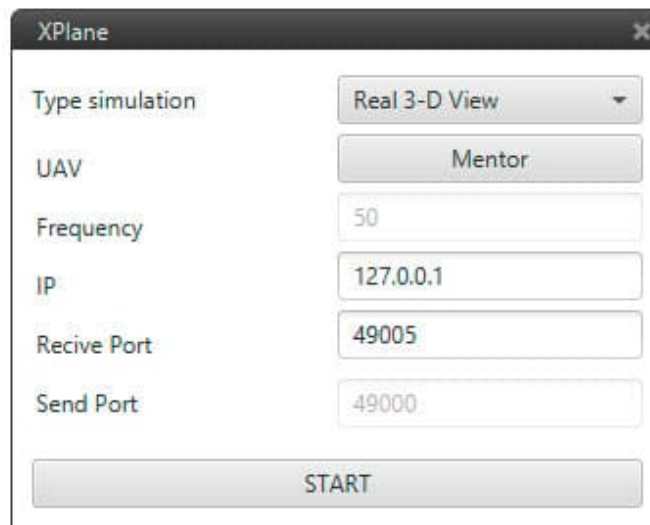
environment. Planemaker tool is available for creating custom models, thereby the operator can see in the interface aircraft model which is being flown.

Video: [Veronte Situational Awareness for UAVs](#).

11.3.1 Veronte Pipe Configuration

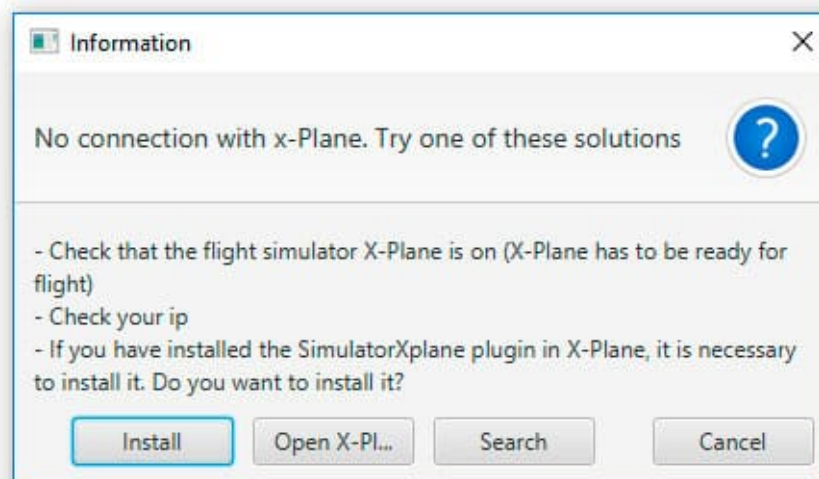
To configure the 3D Simulation in Veronte Pipe it is necessary to complete some steps. First of all, the user must connect Veronte Autopilots to the PC and run Veronte Pipe and XPlane simulator.

When the system is ready, it is possible to open the XPlane connection on Pipe following the path Setup > XPlane. In this panel, the user must select the Real 3-D View from the menu, complete the editable fields as described in section [Mounting](#) and configure X-Plane following the same section.

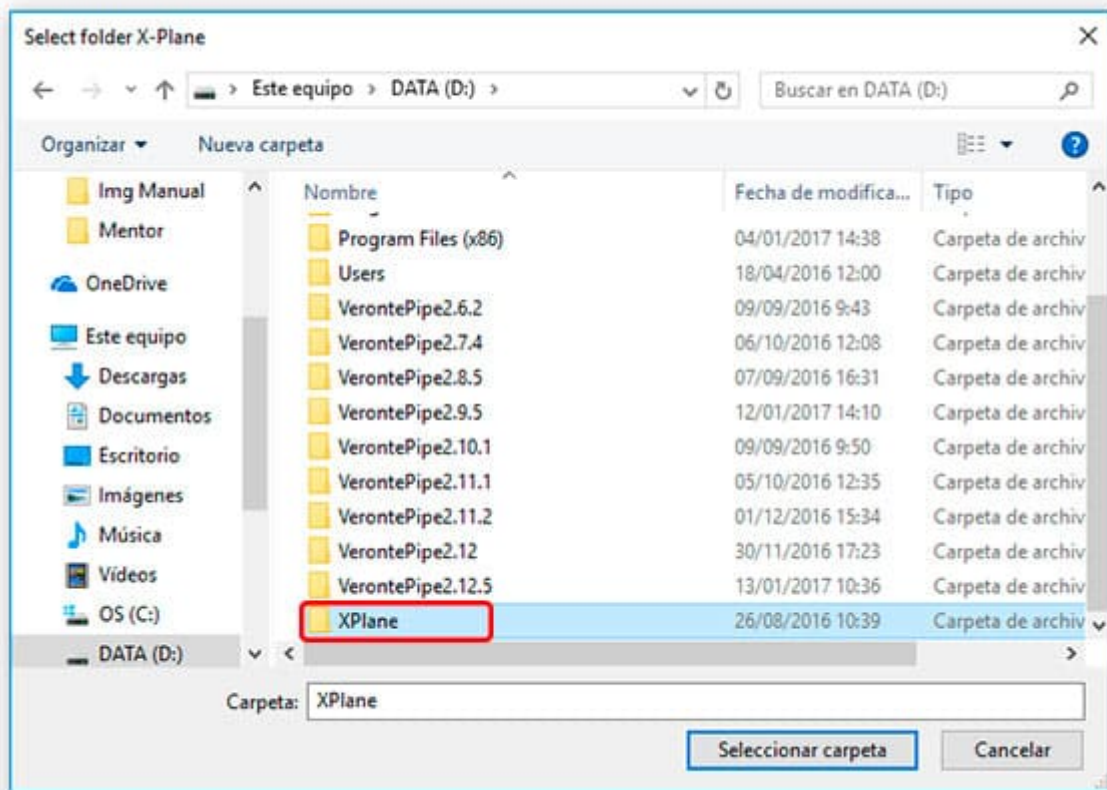


Veronte 3D Simulation

When the connection with XPlane is not possible, an information window shows up. In order to admit the input connection, XPlane requires a plugin installation. By clicking on “Install”, it is possible to select the XPlane folder on the PC in order to install it.



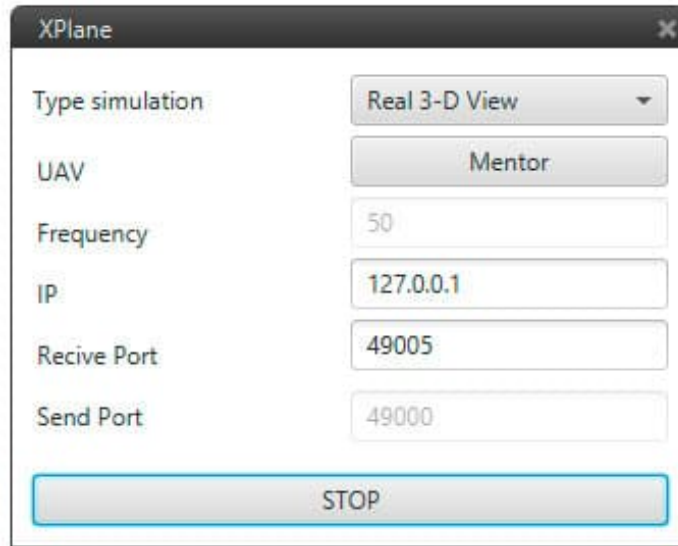
Plugin installation request



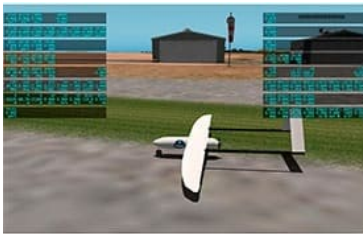
X-Plane folder selection

When the XPlane connection is correctly configured, it is possible to click on “Start” in the XPlane connection panel and the 3D Simulation will start. XPlane will take position on the map and attitude datas from Veronte Pipe and will reproduce the aircraft movement in the simulation environment following the real one.

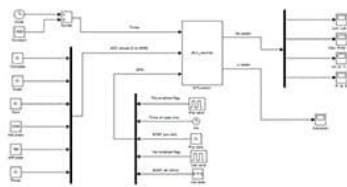
When the flight is over and the simulation is not required, in order to stop the connection with XPlane, it is sufficient to open the XPlane connection panel and click on “Stop”.



Stop XPlane connection



Professional HIL



SIL Simulink



3D Simulation

This section of the manual presents the three possible ways to simulate using Veronte System:

- **Hardware In the Loop simulation (HIL):** This kind of simulation shows Veronte Pipe software performing as it does during a real operation. Meanwhile running the simulation Veronte Autopilot “thinks” it is flying, taking simulator input as real sensor data. The whole flight is controlled by Veronte Autopilot in the virtual environment, making it the perfect tool for vehicle development and for training in the use of Veronte Autopilot.
- **Software In the Loop simulation (SIL):** The software in the loop simulation consists in the creation of a Simulink model that simulates the behaviour of the system formed by the autopilot and platform, without having the physical devices connected to the computer, in contrast to the HIL which has both the autopilot and platform connected to the PC. The SIL allows the user to simulate faster than real time, which allows the user to simulate a system several times.
- **3-D Simulation:** When Veronte Air is flying, it is possible to connect the system with X-Plane simulator through the 3-D Simulator. In this way, the user can see a virtual reproduction of the flight in real time and use it in a FPV-like mode.

EXAMPLES

12.1 4G Communication



12.1.1 4G Communication with Veronte Autopilot

Embention integrates its own Veronte Autopilot with a cloud service that allows the user controlling its own platform around the world with real time telemetry.

1. Add an Autopilot to cloud

Enter to cloud.

Please sign in

| | |
|---|---|
|  | <input type="text" value="User Email"/> |
|  | <input type="password" value="Password"/> |

☐ Remember me

[Sign in](#)

[Login problems?](#)

[Register](#)

Login

Sign in or create an account to be able to access to the cloud services . From your personal space the user is able to add new Veronte Autopilots by clicking on the top right corner.

| | | | | | | |
|------------------------------------|---------|--|-----------------|---|---|--|
| Veronte Cloud Dashboard Autopilots | | | Autopilots list | | + | |
| 1025 | V4 1025 | Version: Unknown Last connection: Unknown | ↶ | ↷ | i | |
| 1043 | V4 1043 | Version: Last connection: Unknown | ↶ | ↷ | i | |

Autopilots List

Introduce the information required in the list and a new autopilot will show up on the previous list.

Add new autopilot to your list

ID of the autopilot

Registration code

Custom name

+

Add Autopilot

Cancel

Add new Autopilot

Contact support@embention.com for further details

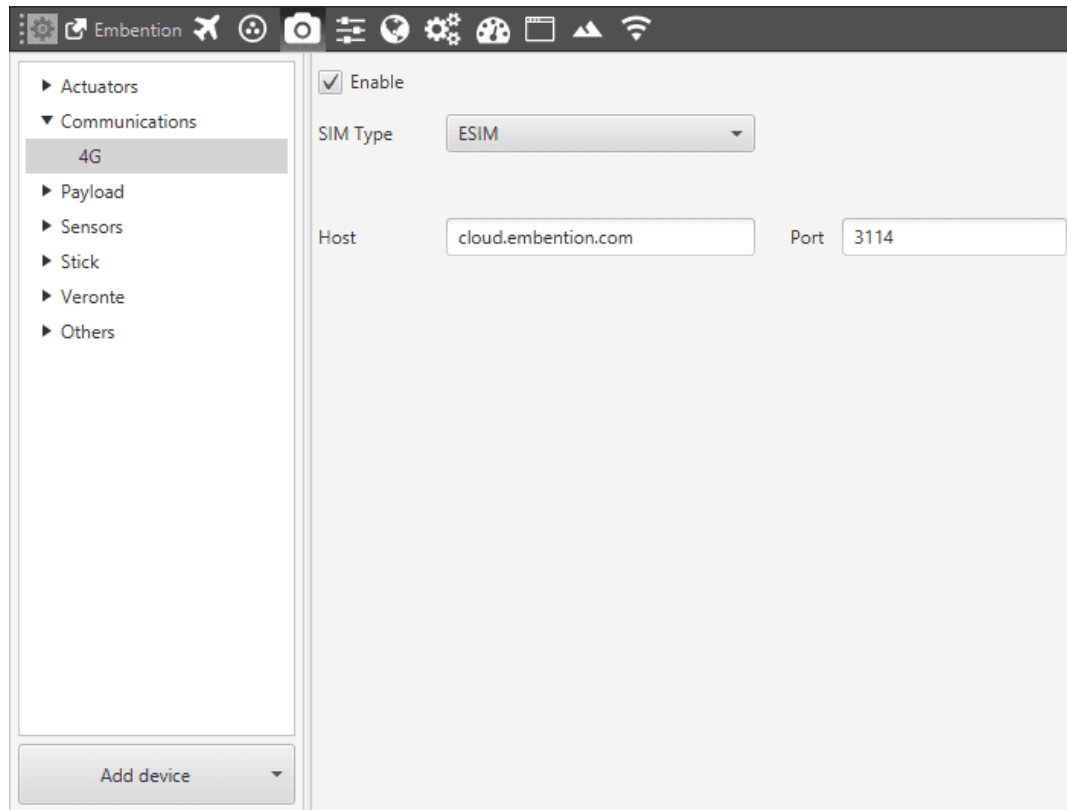
2. Activate 4G Communication in Veronte Autopilot

In order to activate 4G communications open the 'Set up' menu and select 'Devices'. By default the 4G communication is disabled. Select ESIM or SIM accordingly to your set up.

ESIM

For all Veronte Autopilots with ESIM activated the following fields should be filled.

- Host: cloud.embention.com
- Port: 3114



ESIM

SIM

For Veronte Autopilots with a physical SIM card installed in Embention facilities the following parameters have to be set up:

- Host: cloud.embention.com
- Port: 3114
- APN: APN given by the 4G provider.
- PIN: PIN code of the SIM card installed. It should be defined before enabling the communication.

The screenshot shows the Veronte Autopilot configuration interface. On the left is a sidebar with a tree view containing: Actuators, Communications (expanded), 4G (selected), Payload, Sensors, Stick, Veronte, and Others. Below the sidebar is a button labeled 'Add device'. The main panel on the right has a top bar with various system icons. The configuration area includes: an 'Enable' checkbox that is checked; a 'SIM Type' dropdown menu set to 'SIM'; an 'APN' text field with the value 'em'; a 'PIN' field with four digits, all set to '0'; a 'Host' text field with the value 'cloud.embention.com'; and a 'Port' text field with the value '3114'.

SIM

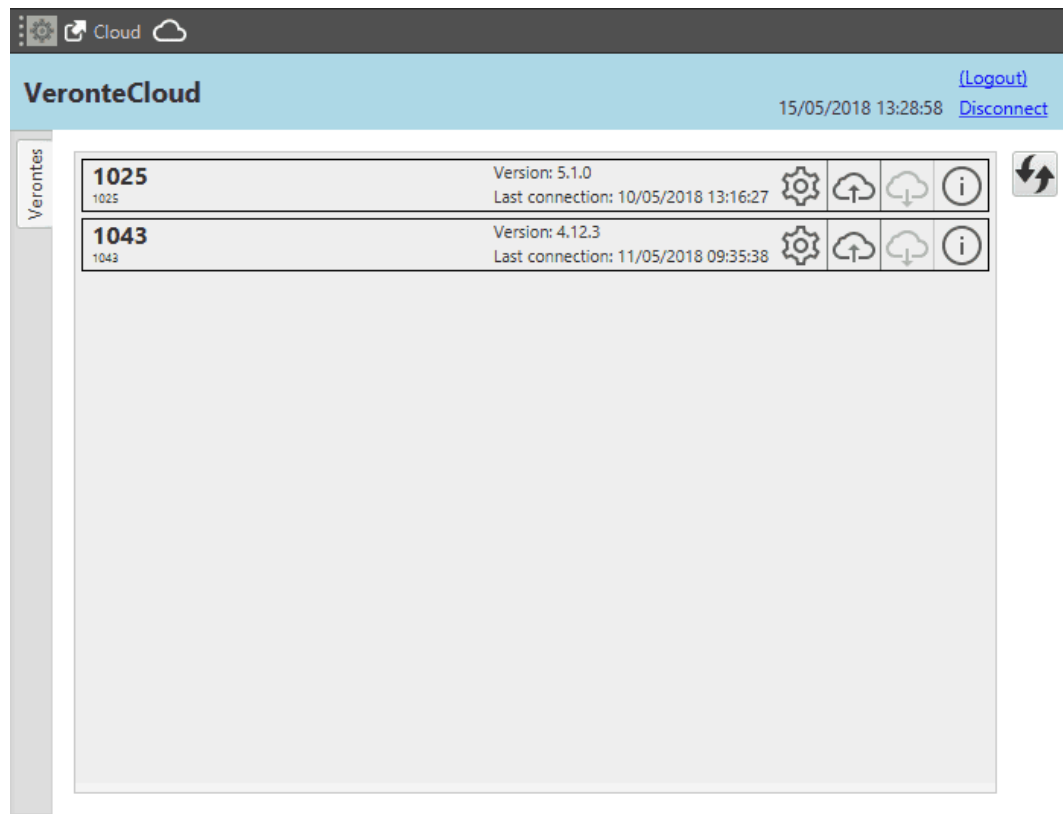
3. Add a Veronte Autopilot from cloud

Open Veronte Pipe and Select the Cloud menu. The user will be requested to enter its own cloud account:

The screenshot shows a login/register form for the Veronte Cloud. It has a light gray background. The form contains: an 'Email' label above a text input field with the placeholder 'user@domain.com'; a 'Password' label above a text input field with the placeholder 'Password'; a 'Remember me' checkbox that is checked; two radio buttons labeled 'Login' (which is selected) and 'Register'; and a large 'Login' button at the bottom.

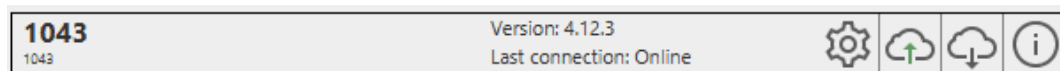
Cloud

All veronte autopilots registered in Veronte Cloud will appear showing the **Name**, **Version** and **Last connection information**.



Cloud

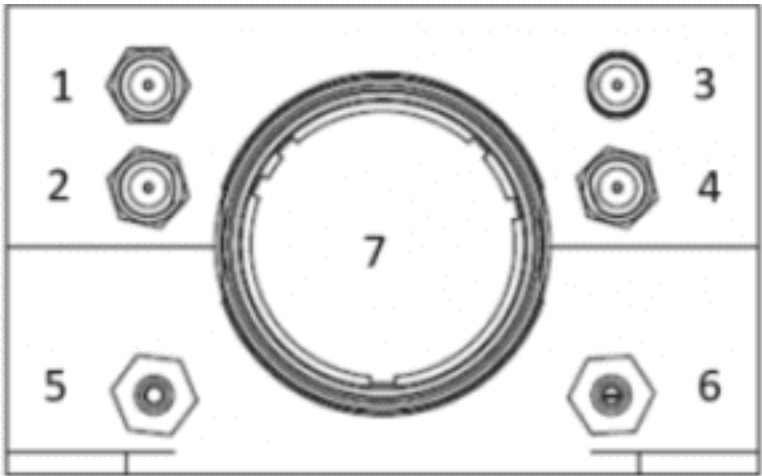
When a Veronte is **ONLINE** the user is able to download the information of the cloud by clicking



A new Veronte will appear in Pipe with exactly the same capabilities compared with a veronte connected by USB.

4. Antenna Connection

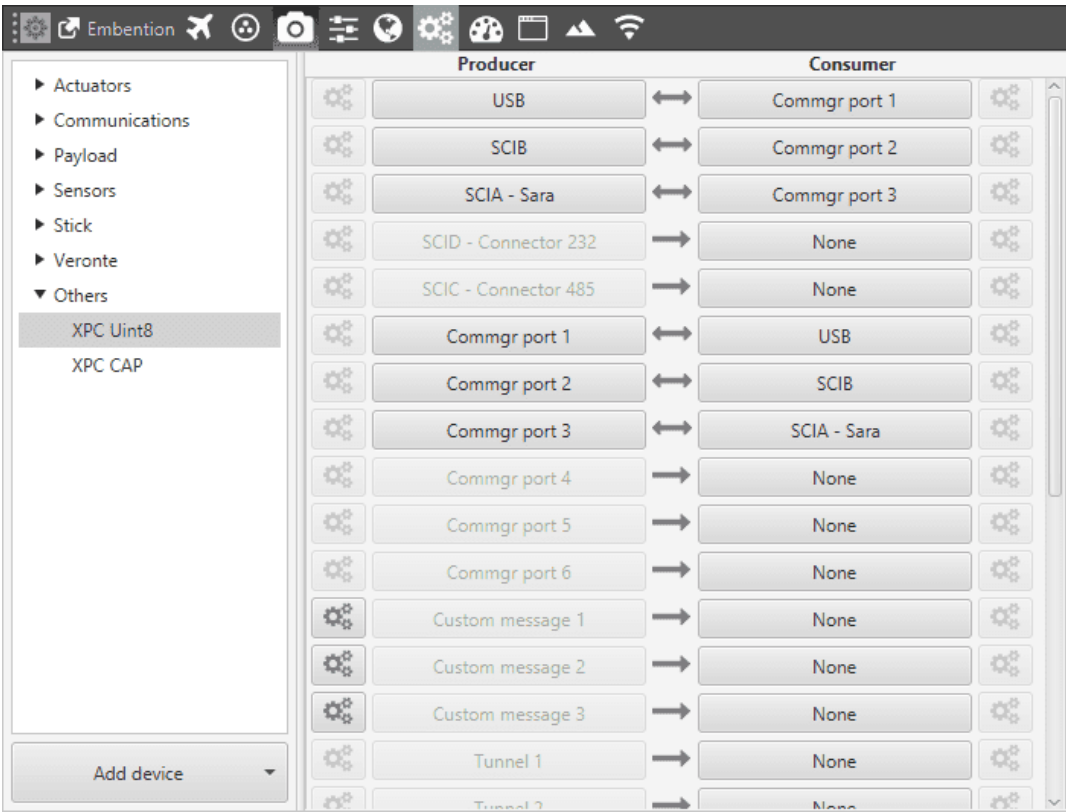
Connect the 4G antenna in 3 (SSMA connector).



Antenna Connector

5. XPC UINT8 Configuration

Check SCIA-Sara configuration match the following scheme:



XPC Uint8

12.2 Configurations

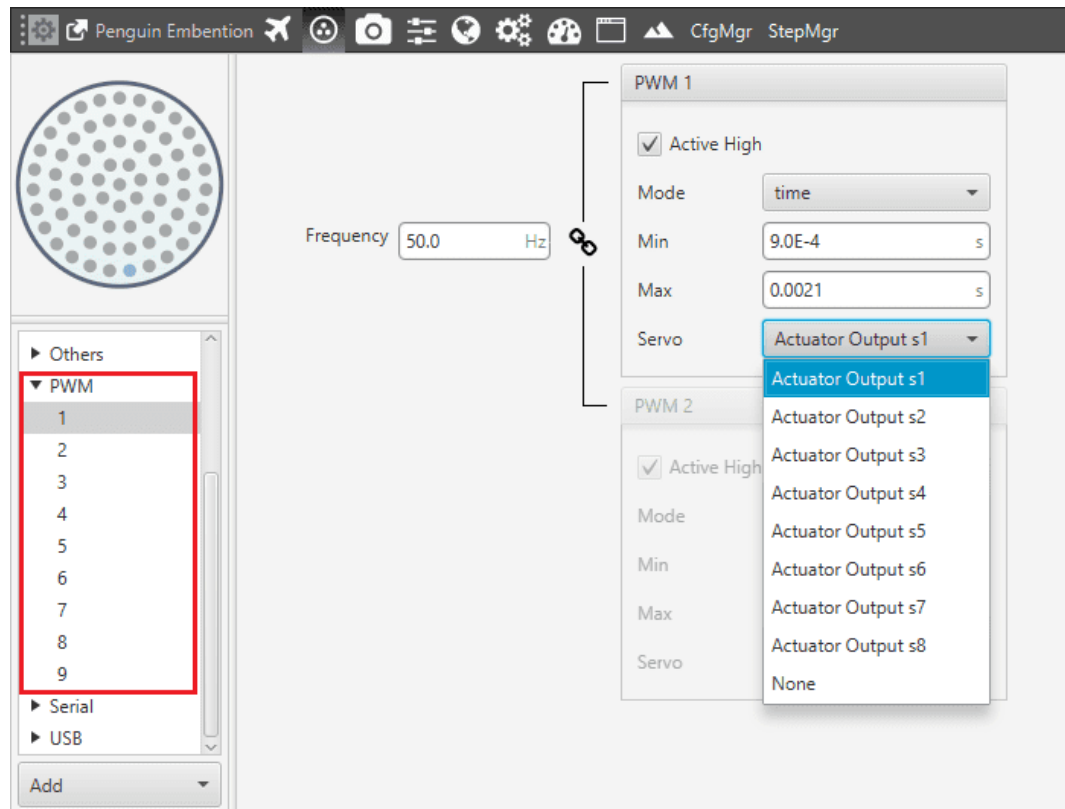
12.2.1 Platform Configurations

12.2.1.1 Fixed Wing-Mentor

12.2.1.1.1 Servo Configuration

Once the Autopilot has been installed and connected according to the guidelines that appear in the section [Hardware Installation](#), the first step of the configuration process is the adjustment and trimming of the servos. The next list contains the instructions to follow in order to trim the platform servos.

1. Set which servo number will have each one of the controls. **Mentor** has a conventional aircraft configuration, so there are five controls: elevator, two ailerons, rudder and throttle. Each one of these controls is assigned to a value of the “Actuator Output” vector (s) according to the pin where it is connected. It is possible to assign any value to any control, i.e the control connected to Pin1 can be s1, s2, s3, s4 or s5.

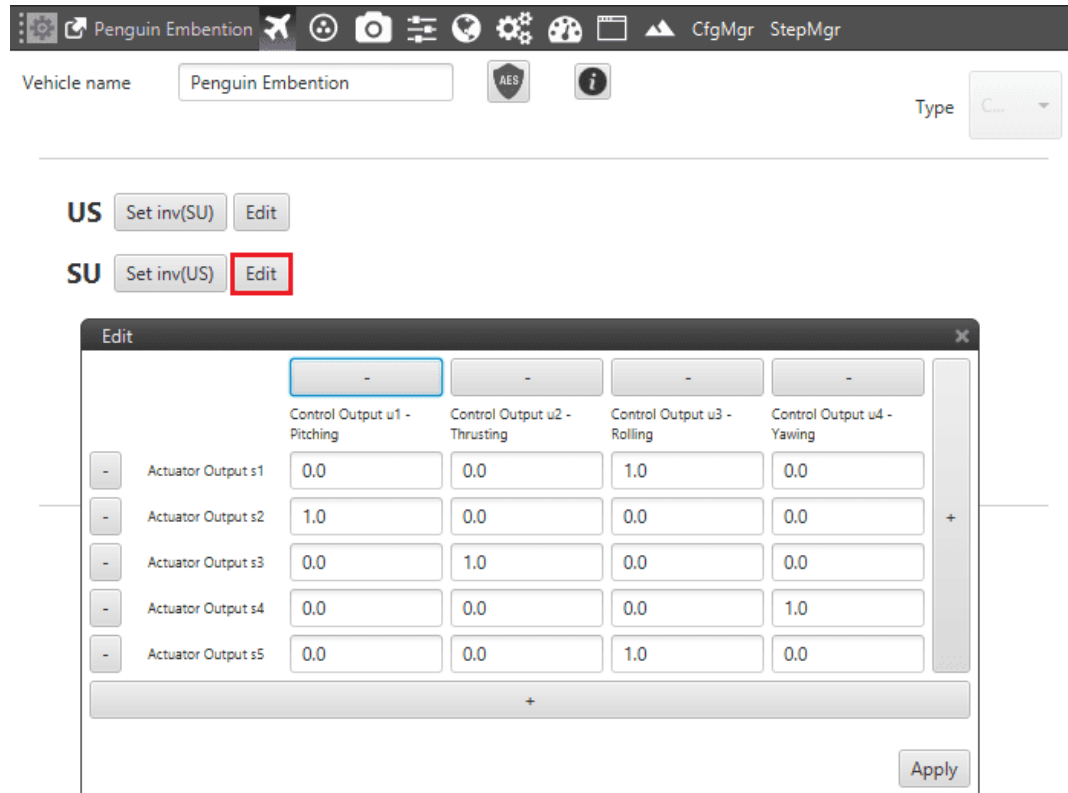


Actuator Output

- **Right Aileron:** Actuator Output s1
- **Elevator:** Actuator Output s2
- **Throttle:** Actuator Output s3
- **Rudder:** Actuator Output s4
- **Left Aileron:** Actuator Output s5

Note: Name **Actuator Output s** can be changed to other more identifiable according to the real actuator such as right or left aileron, see section *System Variables*.

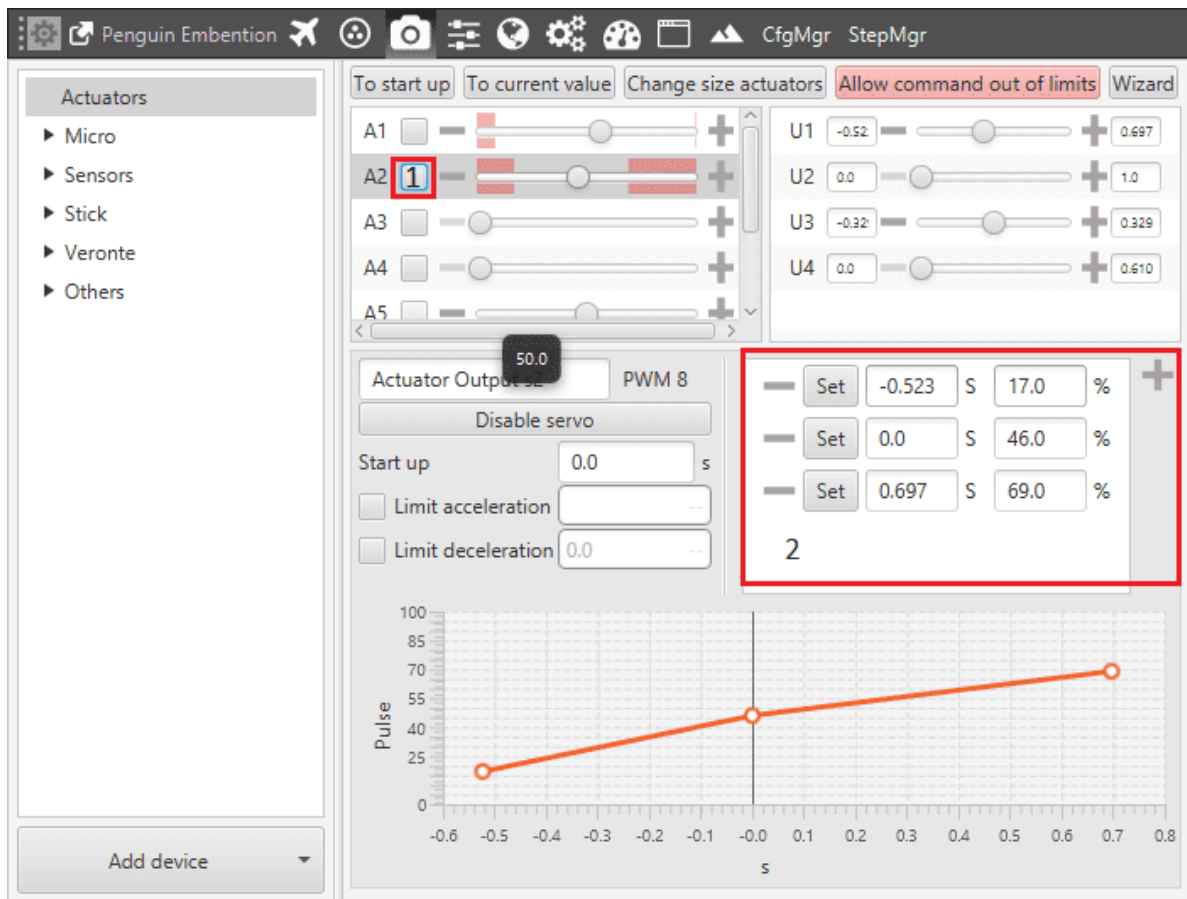
- With the five controls now having a value of the S vector, the **SU** matrix can now be edited. Here the user configures the relation between the controller outputs (“u”) and the servo movements (“s”). In the case of **Mentor**, each control channel is related with only one servo: the pitching with the elevator, the rolling with the ailerons, the yawing with the rudder and the thrusting with the throttle, so the **SU** matrix will be as follows, taking into account which s corresponds which control according to the connections.



SU Matrix Configuration

- **Pitching (u1):** Actuator Output s2 (Elevator)
- **Thrusting (u2):** Actuator Output s3 (Throttle)
- **Rolling (u3):** Actuator Output s1 and s5 (Ailerons)
- **Yawing (u4):** Actuator Output s4 (Rudder)

- Finally, the servo position has to be linked with a real variable, for example, the angle of the control surface or the throttle level. Besides, here are set the physical limits of the servo in the case when it is not possible for it to move in the full range (0-100%).



Servos Configuration

Servos can be moved from the actuators menu when being in INI phase (no phase in green in Veronte Panel).

- When **1** is marked, the slide bar allows the movement of that servo.
- The curve points are added in **2**.

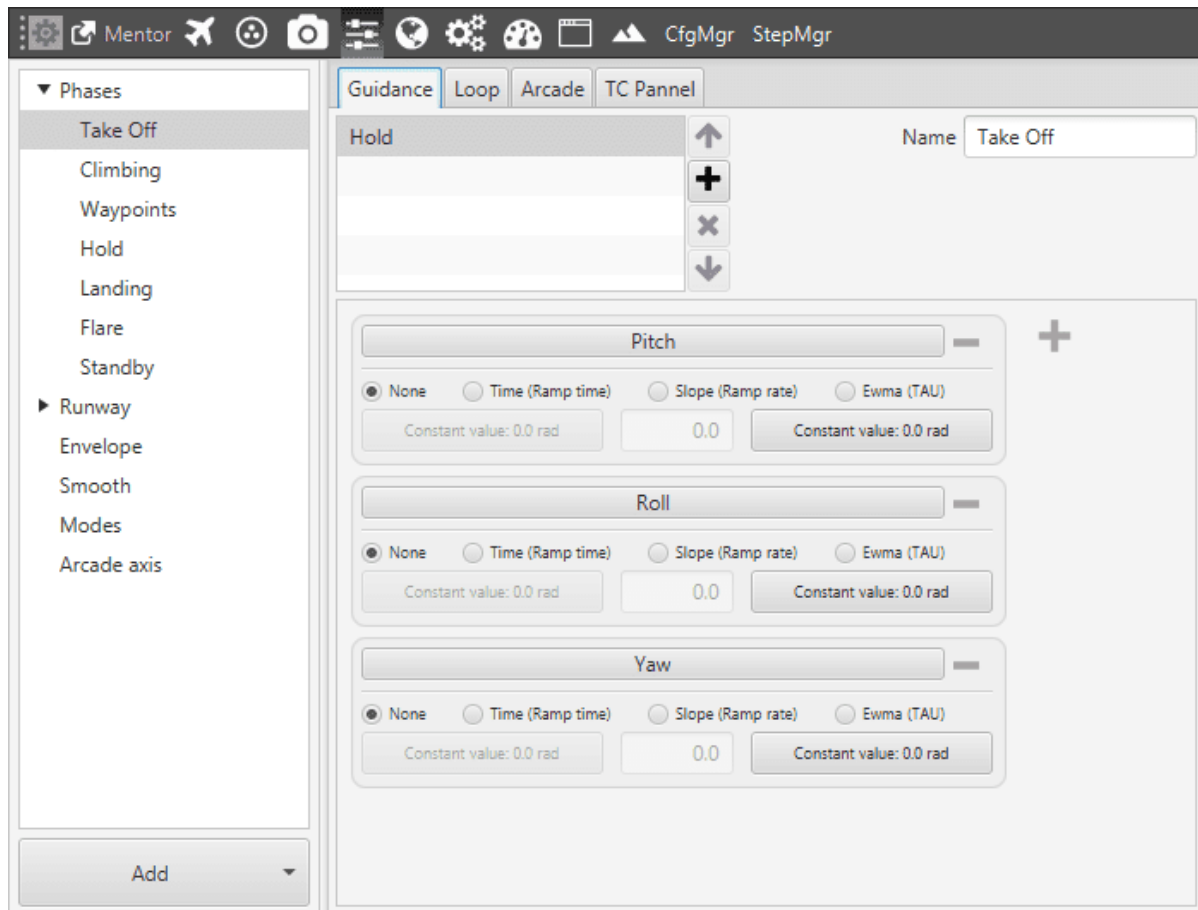
Let's consider the elevator (s2,A2) as an example. In Mentor, the elevator can not be moved totally down because it hits the landing gear, and neither totally up because it hits the rudder. Taking note of the pulse value at this points, the limits of the servo are obtained: 17% and 69% in the example of the previous figure. Measuring also the surface deflection at those points (-0.523 and 0.697 radians) the servo curve is totally defined (including also the point where the deflection is zero).

12.2.1.1.2 Mission Phases

In this section, it will be explained how to configure a whole mission for an airplane, in this case, **Mentor**. The typical phases will be detailed and the guidance for each one of them will be presented.

12.2.1.1.2.1 Takeoff

The takeoff phase will be the one where the aircraft goes from the initial to the lift off point.

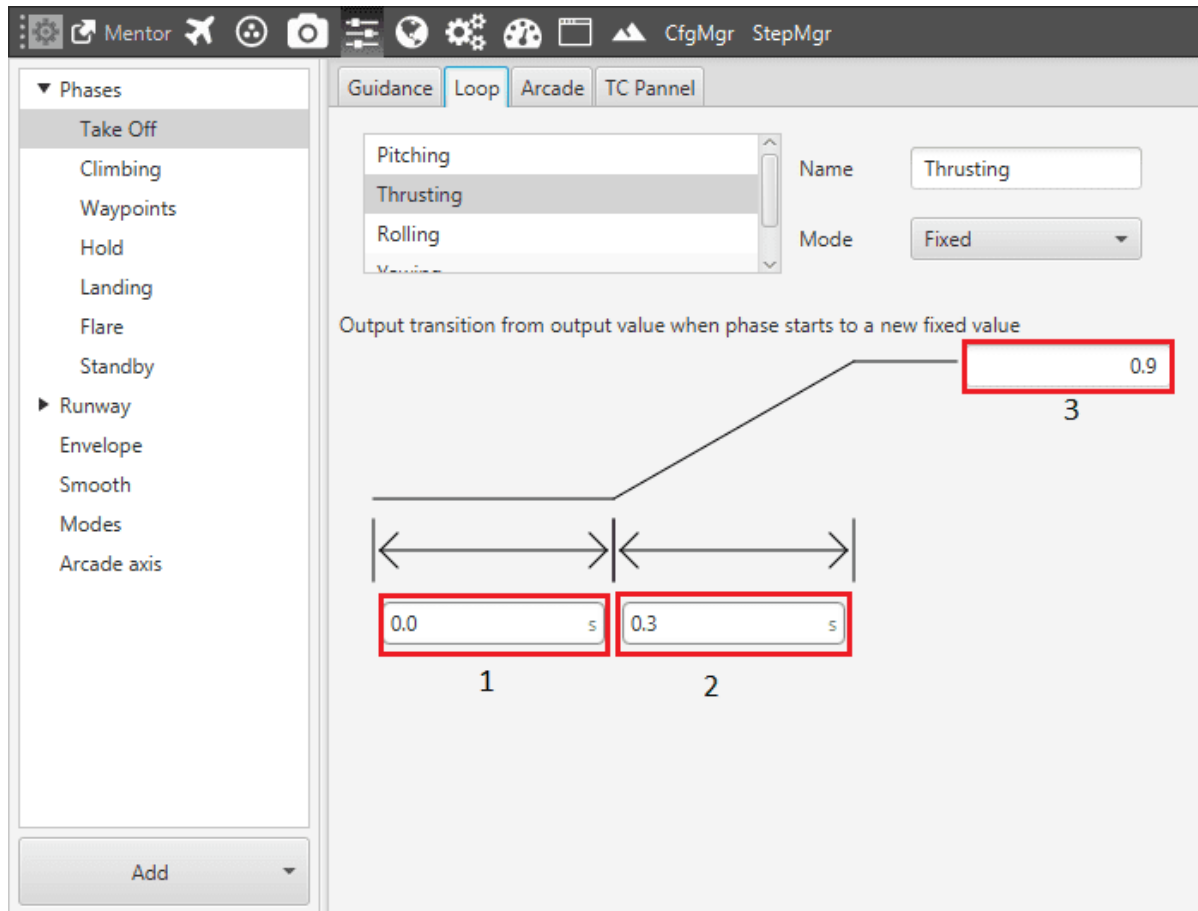


Take Off Phase – Guidance

The guidance for this phase will be a **Hold** of three variables:

- **Pitch and Roll Angle:** kept at 0 [rad].
- **Yaw Angle:** kept at the value that the aircraft has when is set on the runway i.e, the variable selected will be *Yaw*.

Regarding the control loop, there are PIDs in the pitch, roll and yaw control channels, while the throttle has a ramp as defined in the following figure.

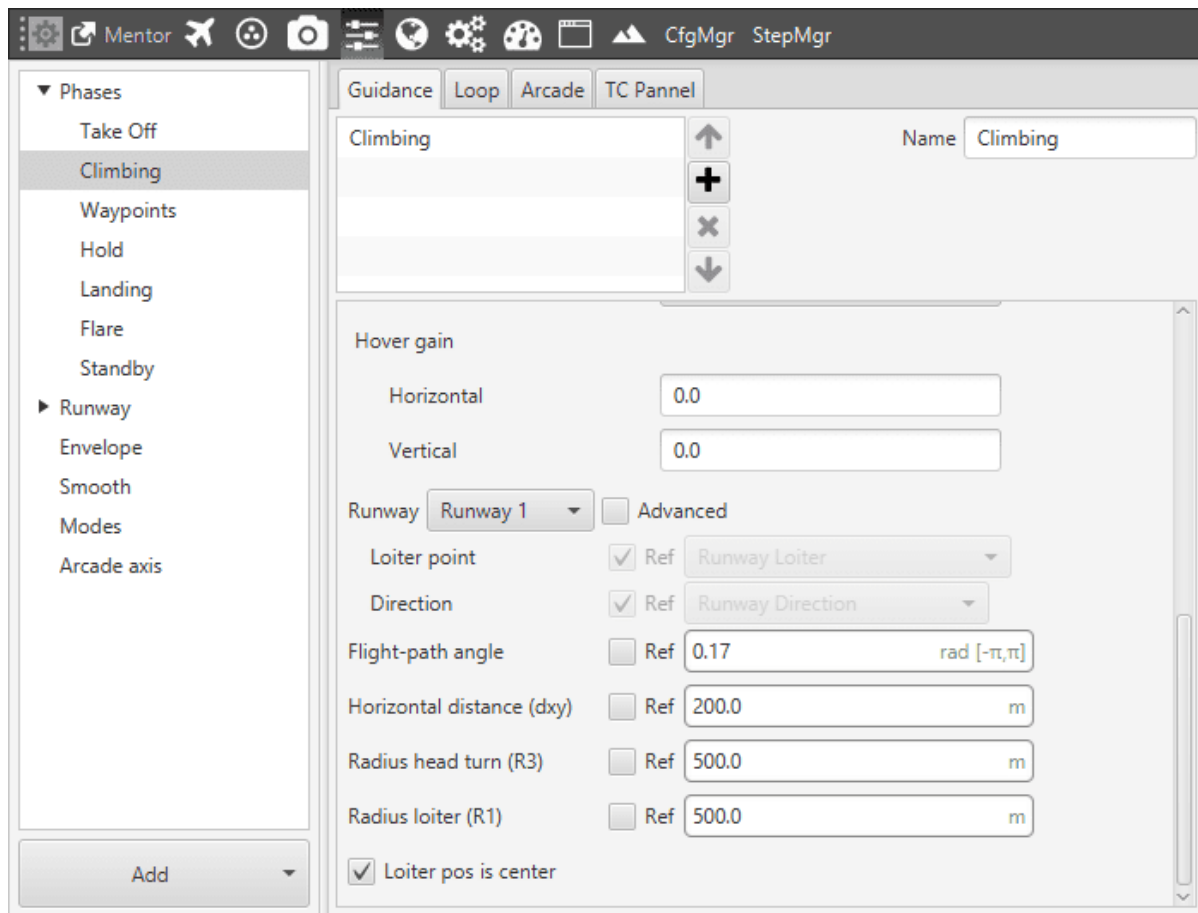


Take Off Phase – Loop

1. Here the initial value is set, in this case 0 because the aircraft starts from a standing point.
2. Time of transtion, 0.3 seconds, the throttle is increased from the initial to the final value.
3. Final value, 0.9 .

12.2.1.1.2.2 Climbing

The climbing phase is used to make the airplane reach the cruise altitude after the takeoff.



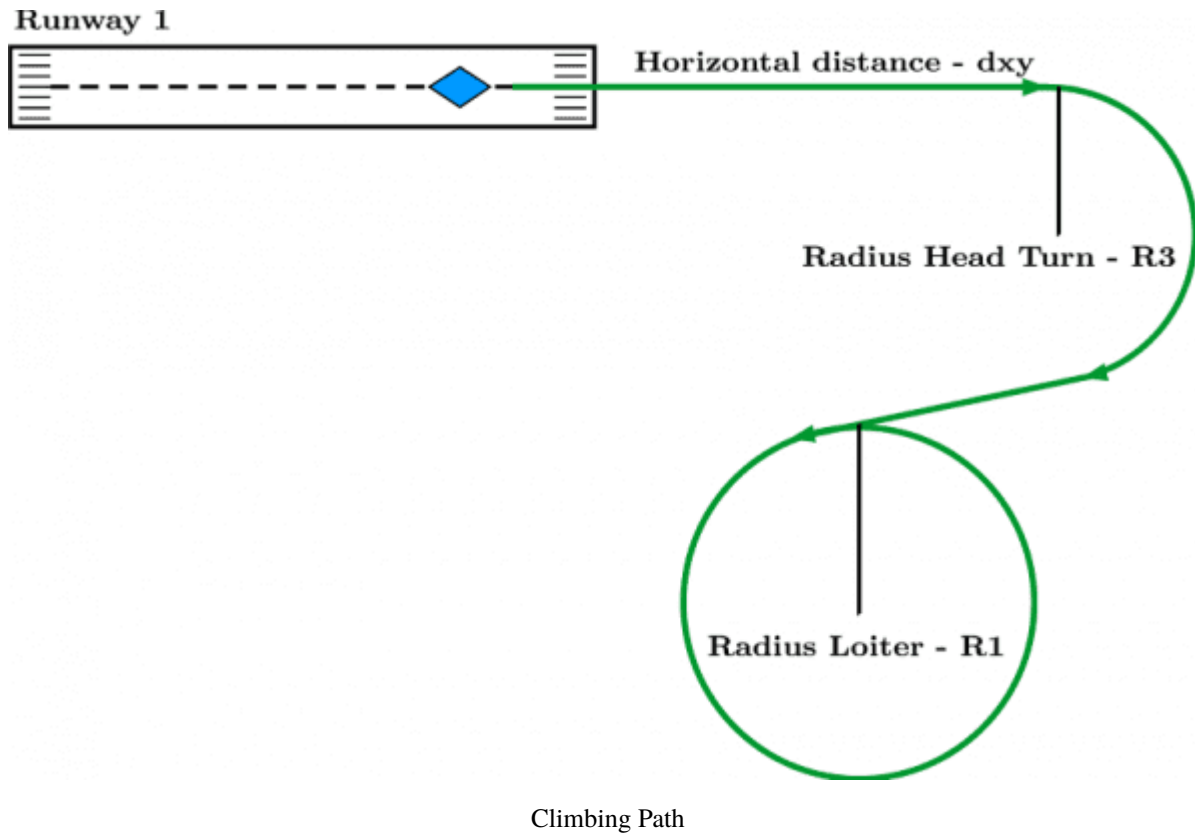
Climbing Phase – Guidance

This phase consists of Climbing Guidance:

- **Line attraction:** value related to how strongly the aircraft tries to reach a path (climb path in this case). This value is commonly between 20 and 40 for airplanes.
- **Set speed:** speed that will have the airplane during the climb, 15 [m/s] .

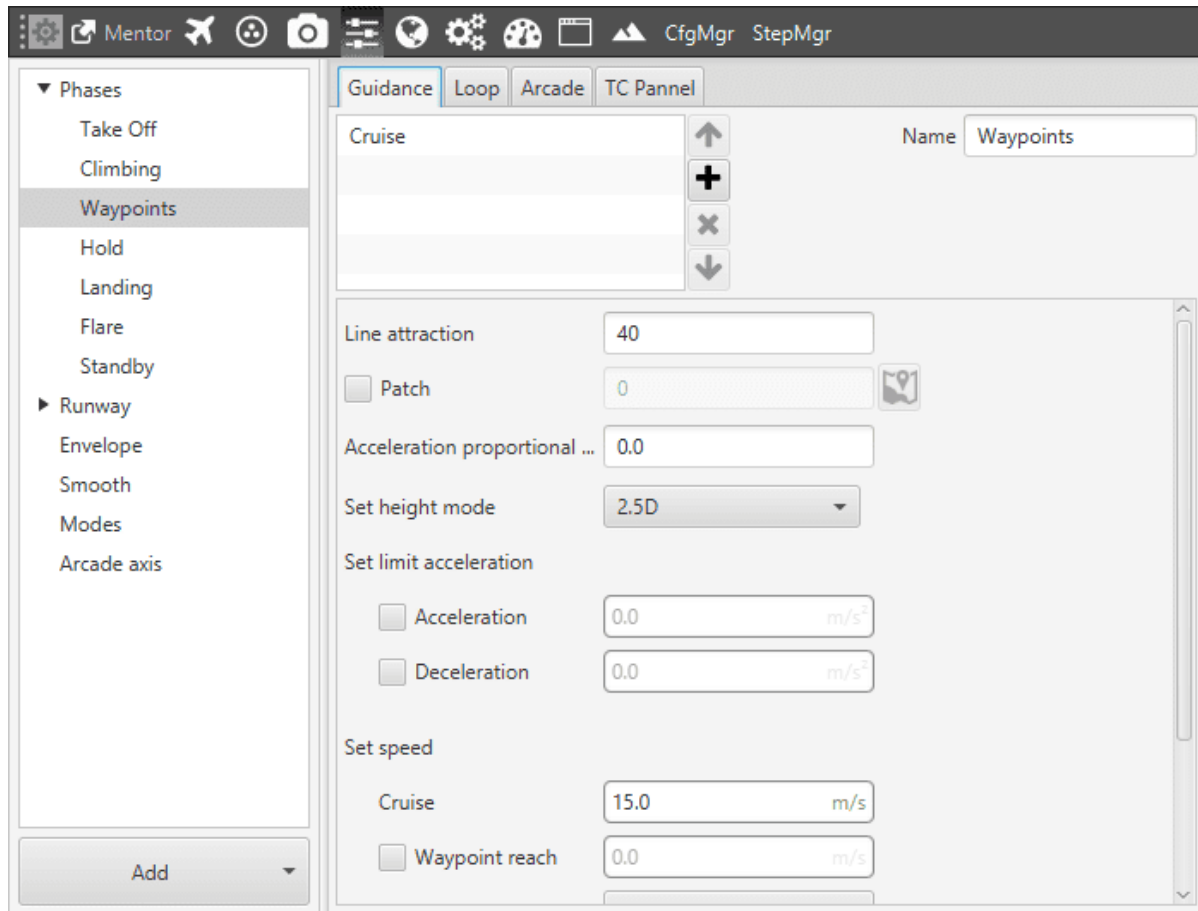
In this phase, the aircraft reaches the mission altitude by performing a spiral route. In order to determine the correct path, these parameters must be set:

- **Runway:** here is selected the runway which previously has been edited in its configuration menu.
- **Flight Path Angle:** angle at which the aircraft will climb, 0.17.
- **Horizontal Distance:** is the distance from the point where the aircraft enters in the phase which contains the climbing guidance, to the start of the circular climbing path, 200 [m].
- **Radius Head Turn R3:** radius of the turn made to head the airplane towards the loiter direction, 500 [m].
- **Radius Loiter R1:** radius of the loiter ascending made by the aircraft to reach an altitude suitable, 500 [m].



12.2.1.1.2.3 Cruise

The cruise phase is where the aircraft follows a route marked by a set of waypoints, which are defined by the user in the **Mission** menu.



Cruise Phase – Guidance

This phase consists of a Cruise Guidance. Parameters set here are the same as the ones from the climbing phase.

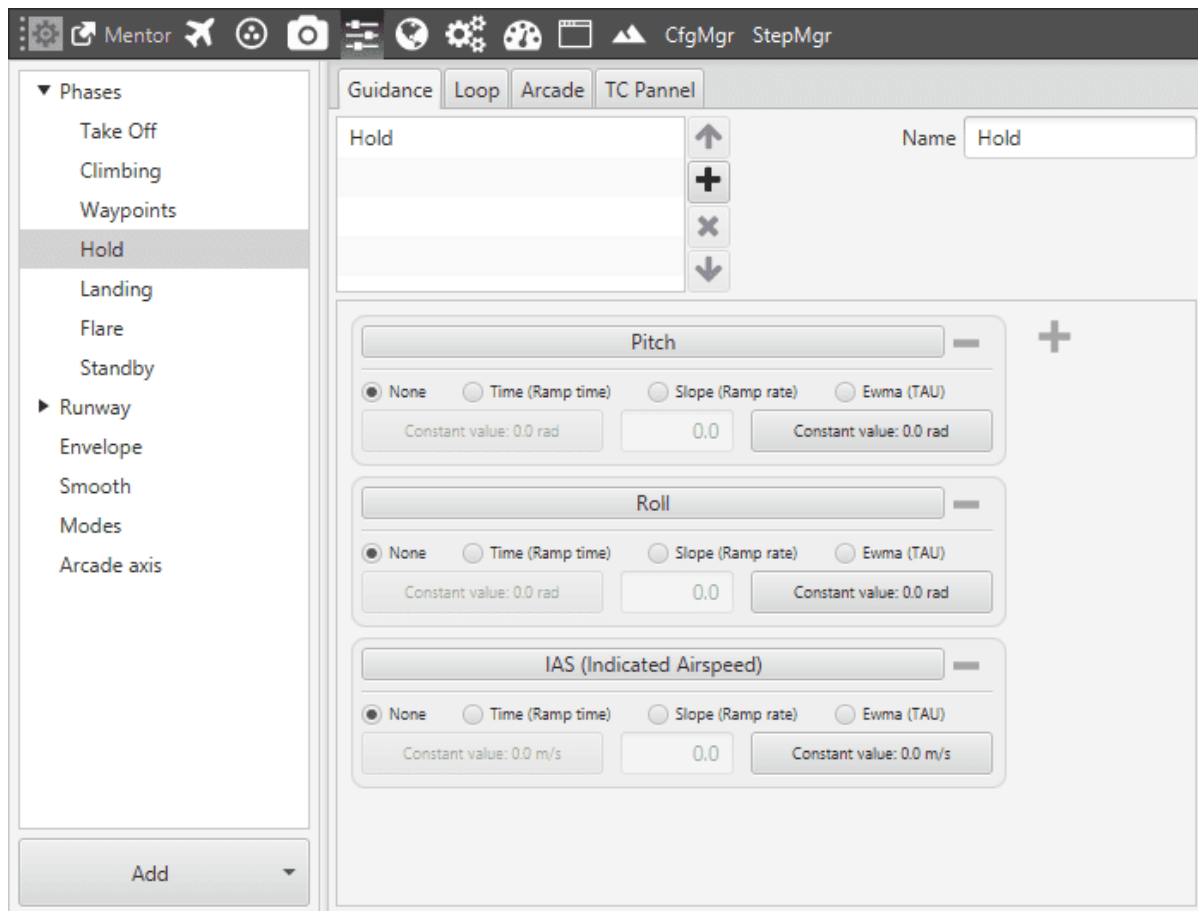
- **Line attraction:** 40.
- **Set Speed:** 15 [m/s].

In this guidance, there is an option related to the gains used to recover the hover point in a multicopter, **Hover Gain**. In this case, the platform configured is an airplane, so this option will not be used.

Warning: When using the Cruise phase, the aircraft will automatically follow the waypoint route. An automation has to be created to make the platform perform in a different way.

12.2.1.1.2.4 Hold

This phase is used to keep the aircraft at a constant heading, for example, when the radio connection is lost.



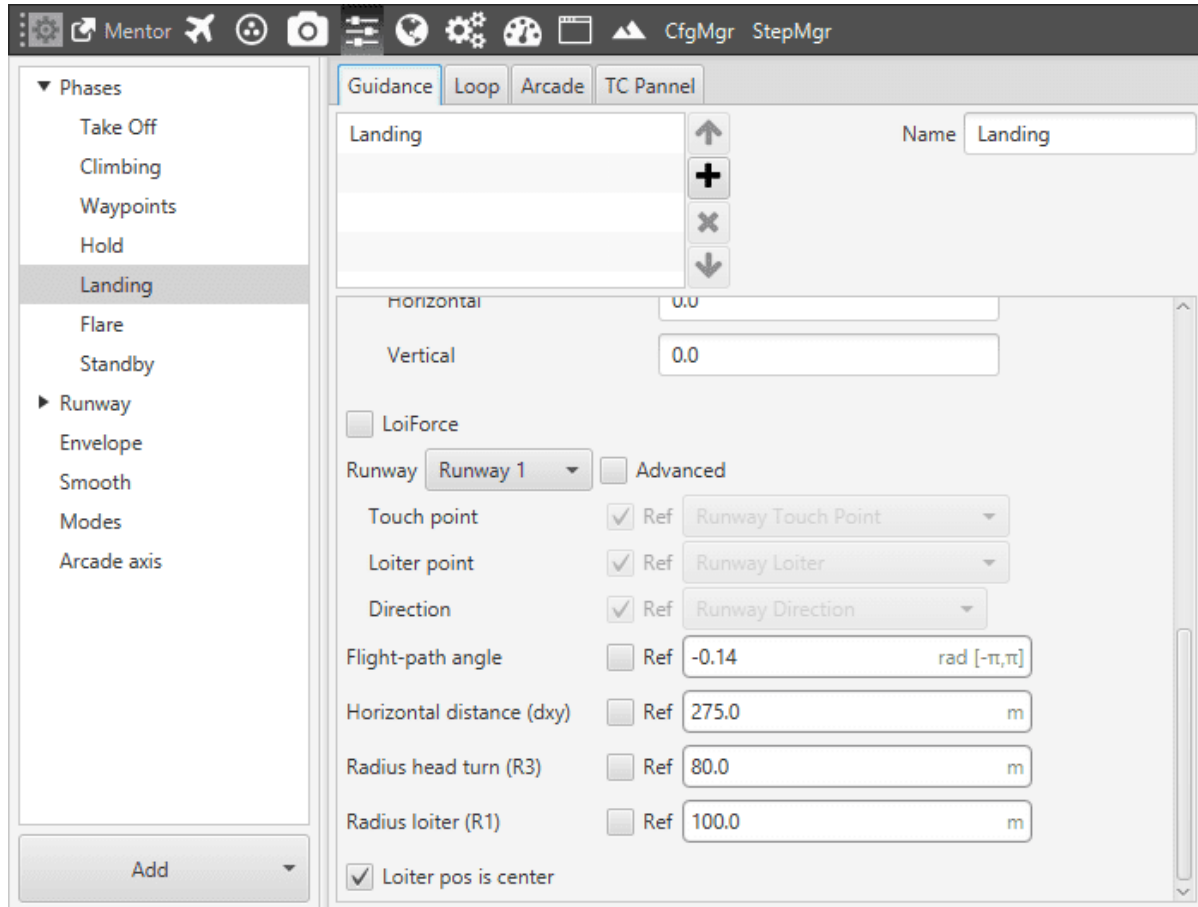
Hold Phase – Guidance

When this phase is active, the aircraft flies in a straight line, until another phase is commanded.

- **Pitch, Roll and IAS:** kept at a constant value 0.

12.2.1.1.2.5 Landing

This phase is used to make the aircraft land at a certain airport. When the flight altitude is too big, this phase contains the parameters which define the route performed by the platform to descent until an altitude where it can line up with the runway.



Landing Phase – Guidance

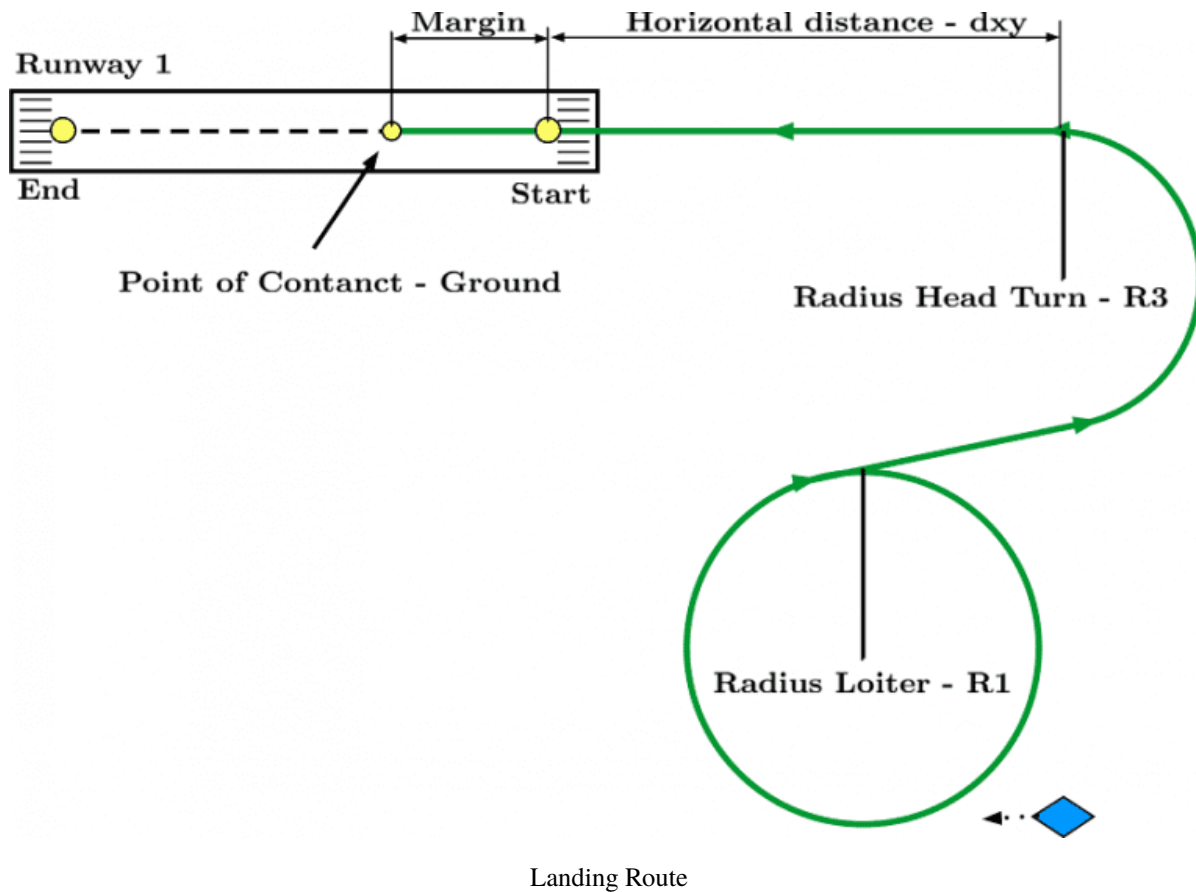
The guidance to be configured is Landing:

- **Line Attraction:** 20.
- **Set Speed–Cruise:** 14 [m/s].

Finally, the following parameters define the path during this phase.

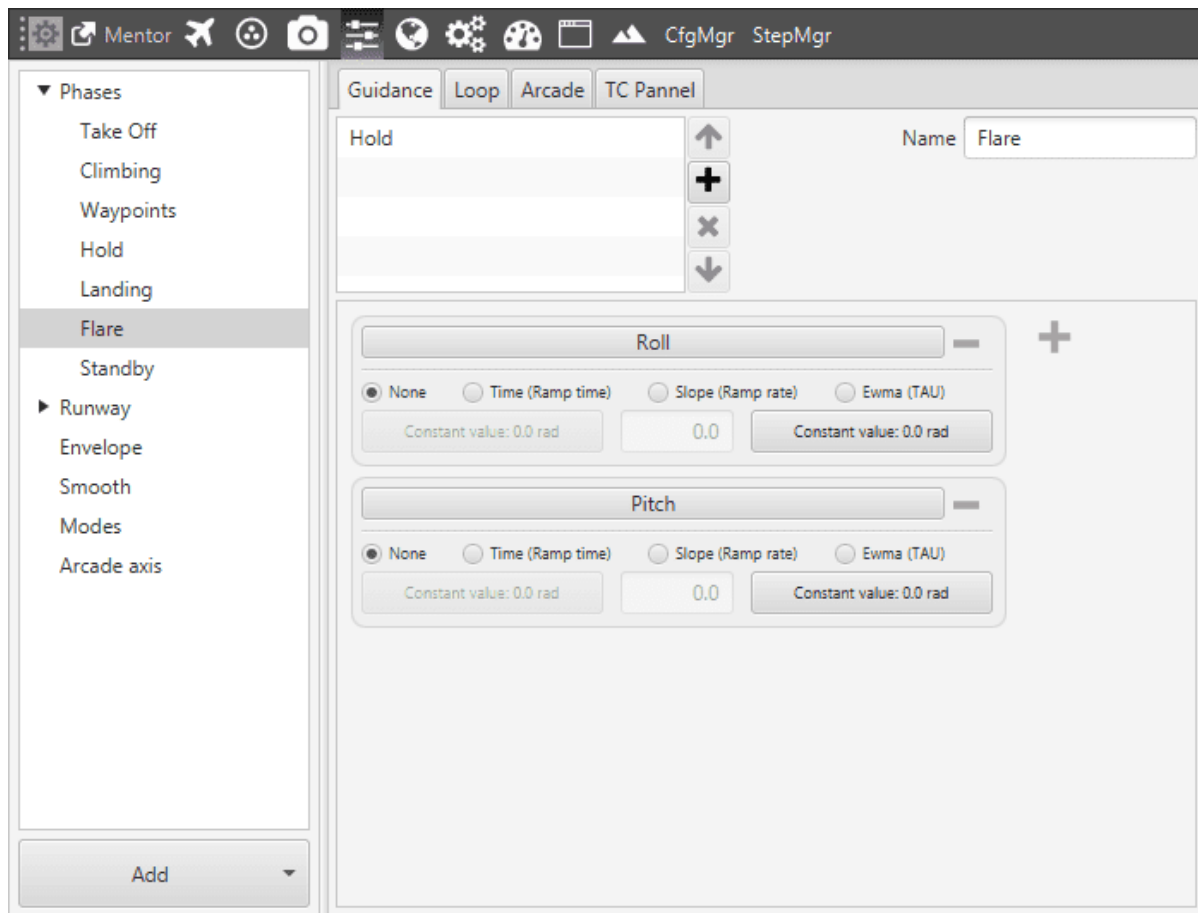
- **Runway:** here is selected the runway which previously has been edited in its configuration menu.
- **Flight Path Angle:** angle at which the aircraft will descend, -0.14 [rad].
- **Horizontal Distance:** is the distance from the point where the aircraft enters in the phase which contains the climbing guidance, to the start of the circular climbing path, 275 [m].
- **Radius Head Turn R3:** radius of the turn made to head the airplane towards the runway direction, 80 [m].
- **Radius loiter R1:** radius of the loiter descending made by the aircraft to reach an altitude suitable to perform the landing manoeuvre, 100 [m].

The route of this phase is shown in the following figure, where each one of the parameters that define it are defined.



12.2.1.1.2.6 Flare

The flare is a maneuver made by an aircraft just before the touchdown. Consists on a rise of the nose to decelerate the descent rate and set a proper attitude before touching the ground.



Flare Phase – Guidance

Considering what is wanted in this phase, the guidance to command is a **Hold**.

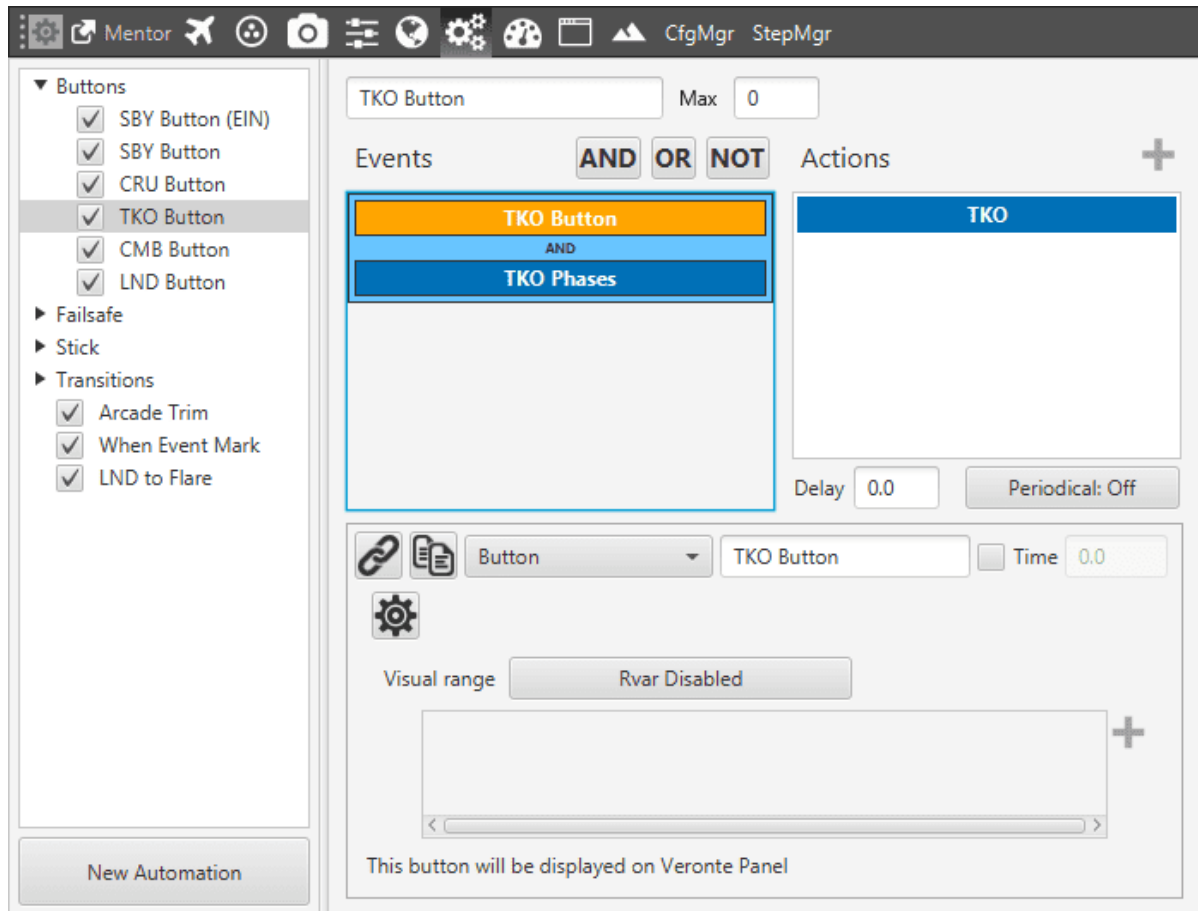
- **Roll – Pitch Angle:** kept at a constant value 0 [rad].

Regarding the thrust in this phase, the engines are shut off, so the mode of the controller is **off**.

12.2.1.1.3 Automations

Automations are the mechanisms used to perform an action when some event is triggered. These actions could be a change from one phase to another, taking a photo, dropping a payload and so on. In this section, the conventional automations used in a flight of the Mentor airplane will be detailed.

The first automations to be created are the ones linked to the buttons of Veronte Panel. When clicking one of these buttons the phase is changed to the one shown on the button label.



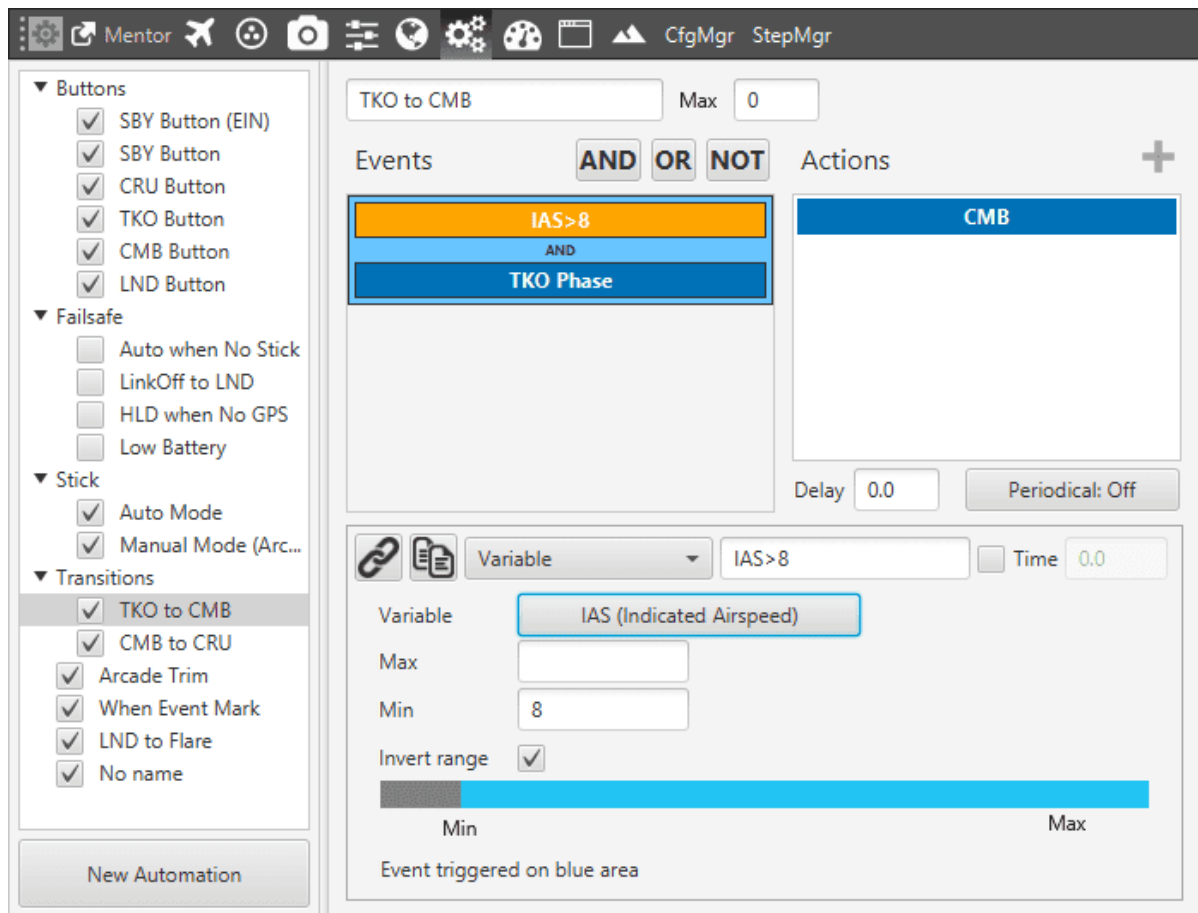
Automation Panel

Normally, besides the button event, a safety condition is added to the automation, which consists on letting only the system to change to a phase when being in a certain set of phases. For example, only change to cruise when being in climbing or change to landing when being in cruise or climbing. It is only necessary to create an AND condition with the phases to change from. In the previous figure, the automation is the change to takeoff when clicking on Veronte panel, being on Standby. This process is repeated for the rest of phases.

12.2.1.1.3.1 Takeoff to Climb

The change from the take off to the climbing phase occurs when the IAS of Mentor on the runway is greater than 8 m/s. There are two events to be configured:

- **Variable:** $IAS > 8 \text{ [m/s]}$.
- **Phases:** Standby.



Take off to Climb – Automation

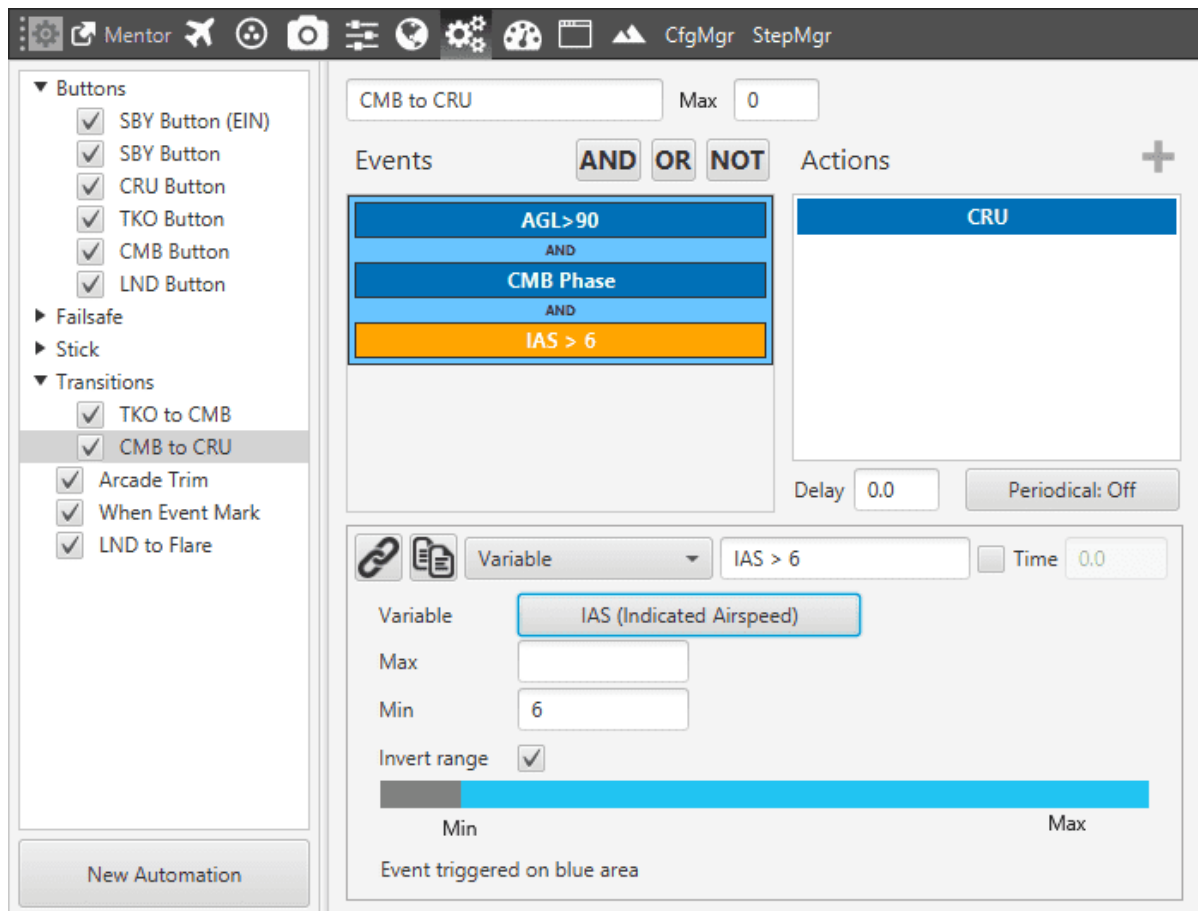
The action will be a change of Phase:

- **Phase:** Climbing.

12.2.1.1.3.2 Climbing to Cruise

When a certain altitude is reached, Mentor changes to the Cruise phase where it starts to follow the path determined by the user. As a safety condition, the change will only happen when having a speed greater than 6 meters per second. There are three events to be configured:

- **Variable:** AGL > 90 [m].
- **Variable:** IAS > 6 [m/s].
- **Phases:** Climbing.



to Cruise - Automation

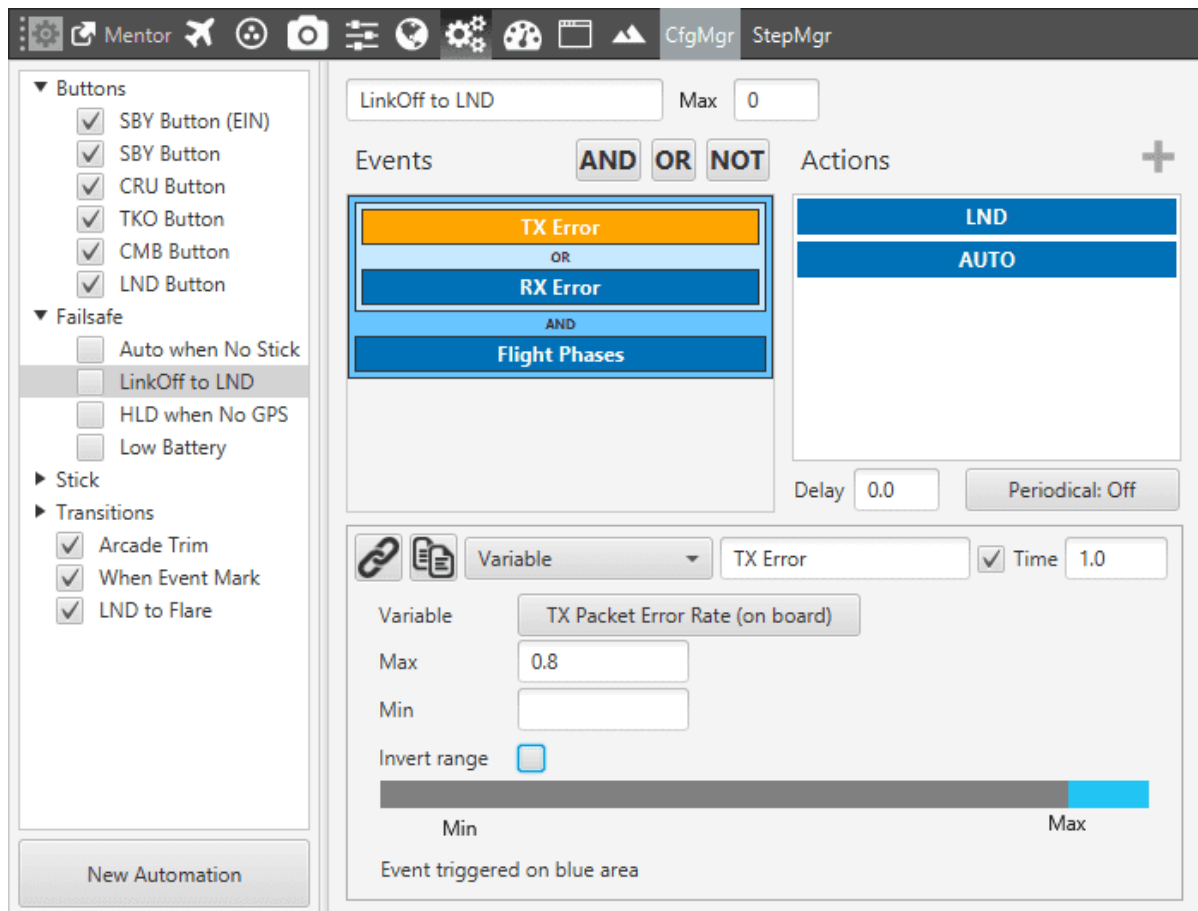
Finally, the action will be a change of Phase:

- **Phase:** Cruise.

12.2.1.1.3.3 Radio Error

When the radio connection from the ground station to Mentor is lost, the aircraft is forced to change to auto mode and land. It is necessary to configure two events:

- **Variable:** TX or RX error (TX /RX Packet Error Rate (on board)), greater than 0.
- **Phases:** Climbing or Cruise.



Radio Error – Automation

Two actions are configured:

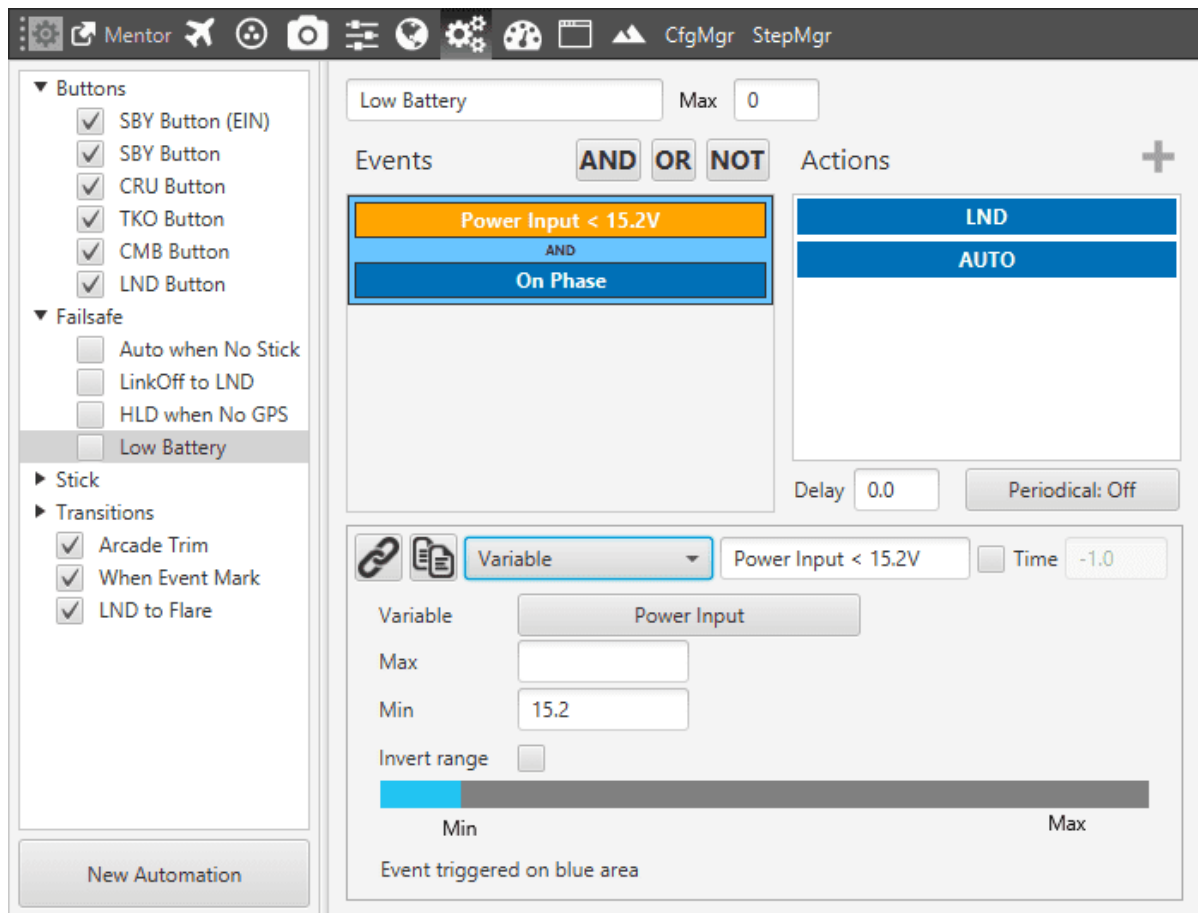
- **Phase:** Landing.
- **Mode:** Auto.

When the value of the flag that evaluates the TX or RX error (“TX Packet Error Rate (on board)”) has a value greater than 0.8, being the aircraft in climb or cruise phases, it is forced to land. The value 0.8 means that 80 percent of the packets sent through the radio link have been lost.

12.2.1.1.3.4 Low Battery

When the battery is below a certain level (15.2 Volts in this case), and the aircraft is in climb or cruise, it is automatically commanded to land. The events to configure are:

- **Variable:** Power Input < 15.2 [V].
- **Phases:** Climbing or Cruise.



Low Battery – Automation

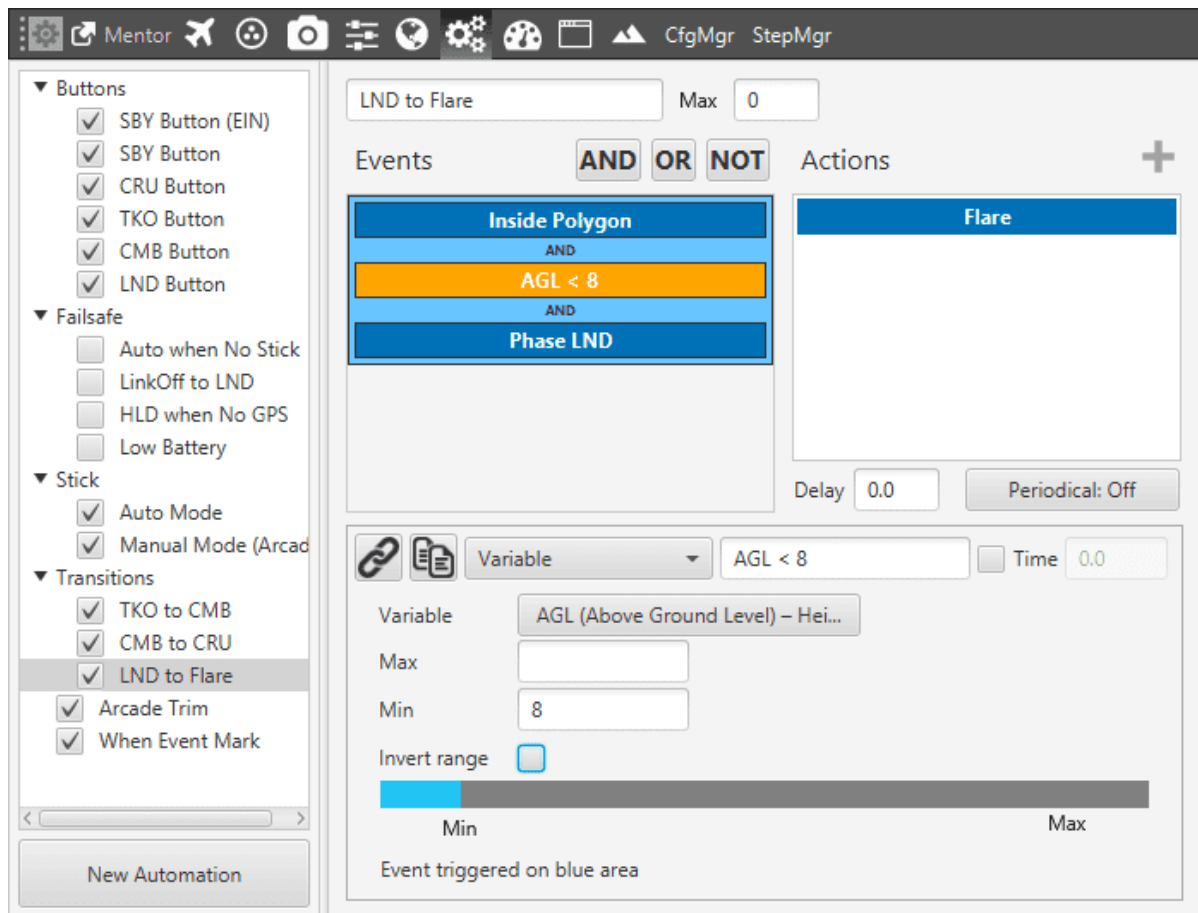
Two actions are configured:

- **Phase:** Landing.
- **Mode:** Auto.

12.2.1.1.3.5 Landing to Flare

The change from the landing to flare phase is triggered when the aircraft is above the runway and about to touch the ground. This idea is implemented with a set of three events:

- **AGL:** the aircraft is below a certain altitude, in this case, 8 meters.
- **Inside Polygon:** Mentor enters in a polygon defined in the map on the runway head.
- **Landing Phase:** the aircraft is landing phase.



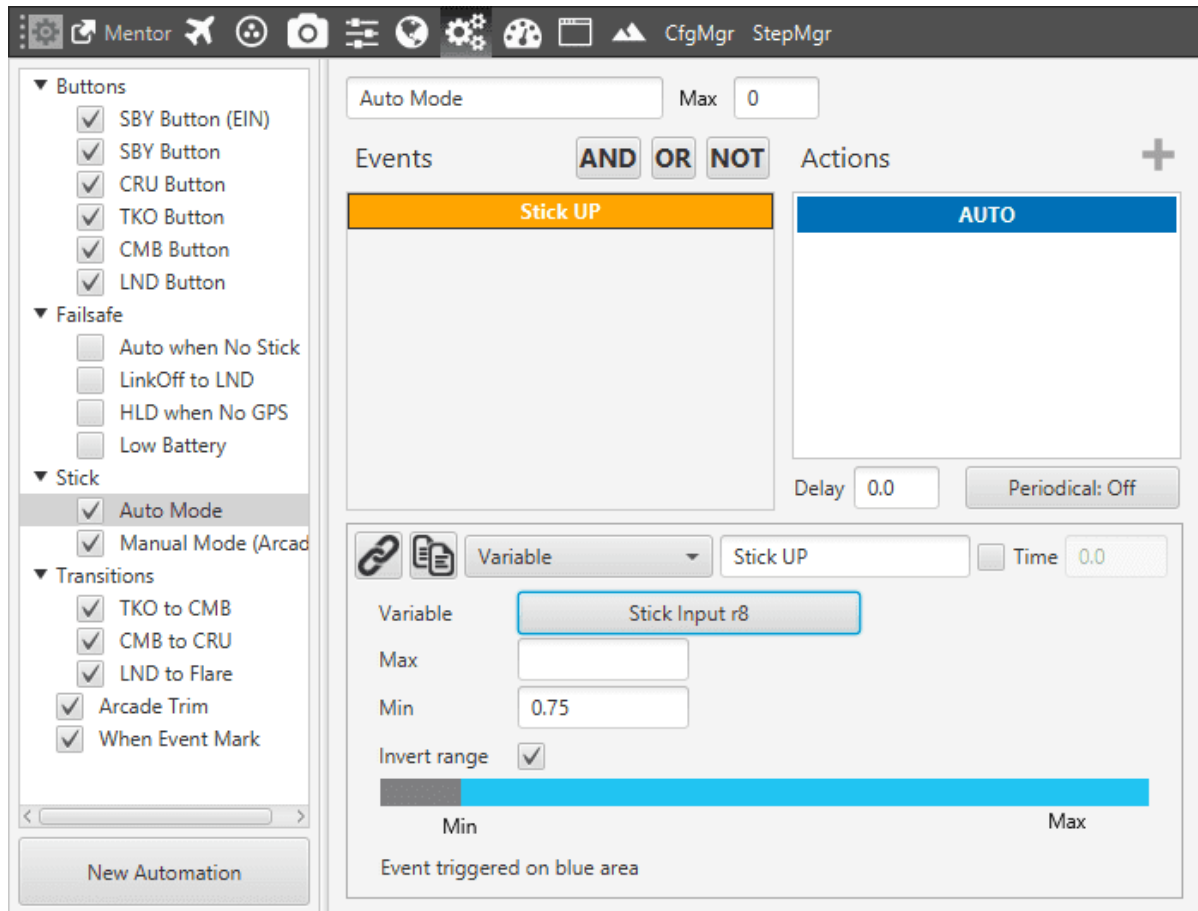
Landing to Flare

When these events are fulfilled, Mentor enters in the Flare phase and lands.

12.2.1.1.3.6 Stick Auto

This automation changes the control mode according to the command sent by the radio controller. Only an event is necessary:

- **Variable:** Stick Input rX, greater than 0.75.



Stick Auto

The action is:

- **Mode:** Auto

As it can be seen, the channel that controls the mode is the 8, so according to its value the mode is changed.

The process is the same when creating manual mode, but now the value of Stick Input r8 has to be lower than 0.25.

The Mentor UAV is a radio controlled, electric powered trainer aircraft. Made of special foam, is one of the largest RC aircrafts made of this material.



Mentor

It is a platform with the conventional controls of an airplane: elevator, ailerons, rudder and throttle, so it can be a good example of how to configure a typical airplane in Veronte Pipe.

12.2.1.2 Flying Wing-W210

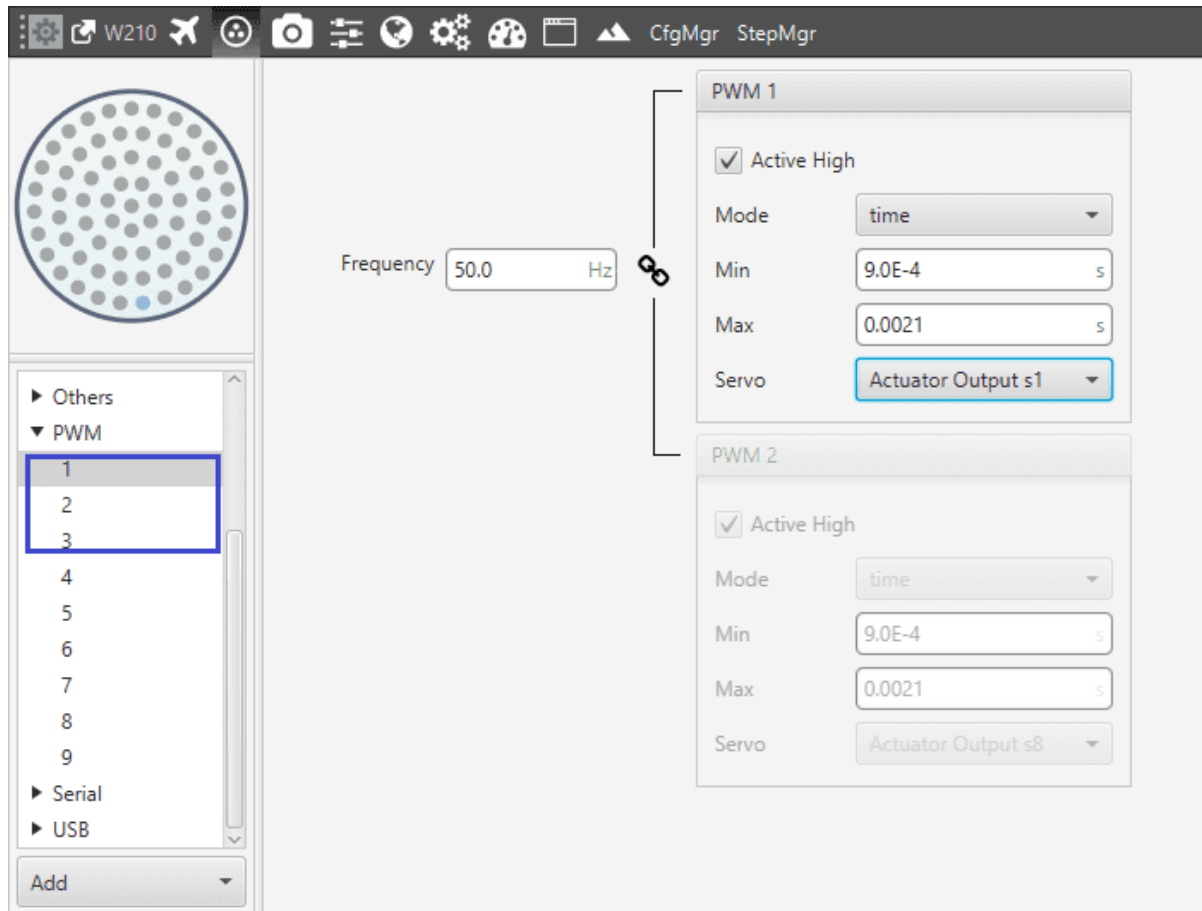
12.2.1.2.1 Servo Configuration

W210 configuration process can be performed by using a VerontePipe software version connected with the hardware system and the Autopilot as explained in section *Aircraft Mounting* of the manual.

12.2.1.2.1.1 Servos Output

The first step of the process is the servos configuration.

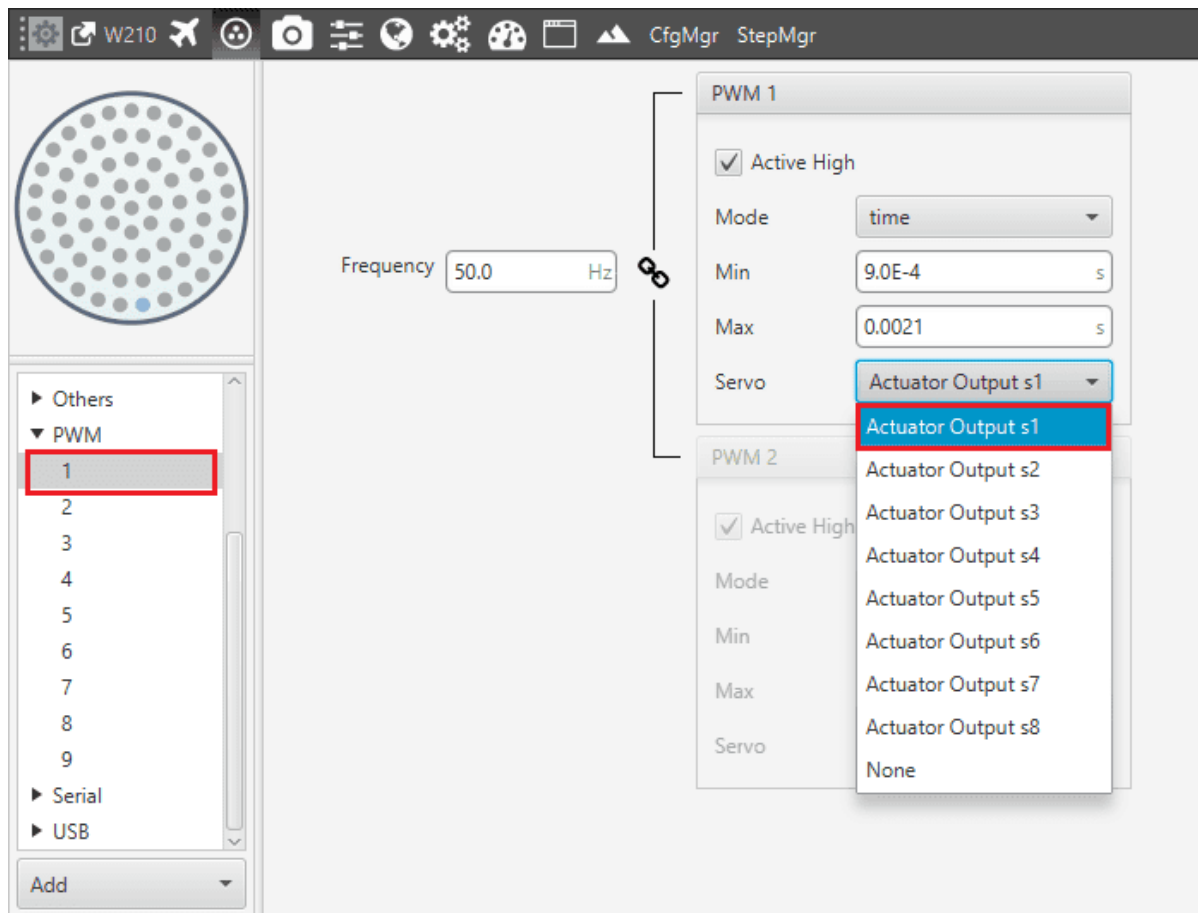
The controls of the airplane are the two control surfaces and throttle. Each one of these control corresponds to a pin of the connector and they must be positioned in the same order in an **S** vector who represents the **Actuator Output**. It is possible to connect any pin to any control surface or command but the easiest way to perform it and avoid confusion is by following the pins number.



Output-pin links

In this case, it is possible to use only 3 pins:

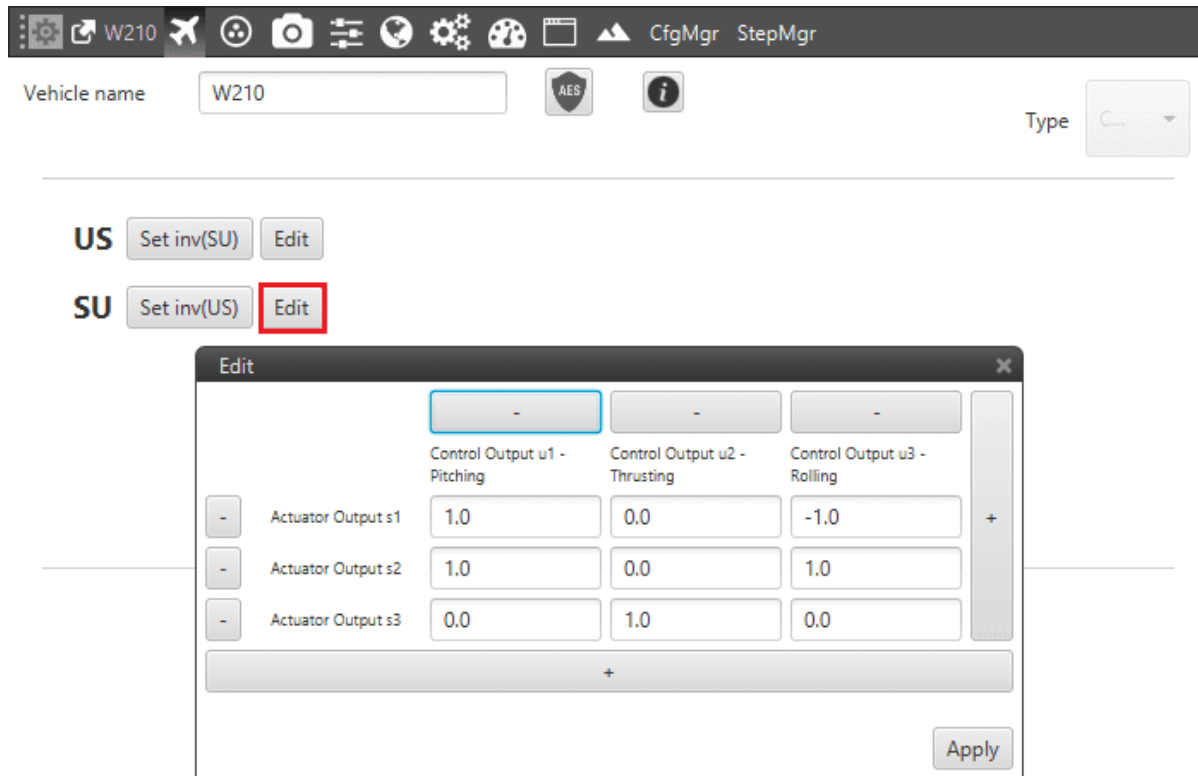
- Output 1 – Actuator Output s1 (**Aileron 1**)
- Output 2 – Actuator Output s2 (**Aileron 2**)
- Output 3 – Actuator Output s3 (**Throttle**)



Output 1 – pin 1 configuration

12.2.1.2.1.2 SU Matrix

At this point, the **S** vector is defined and the **SU** matrix can be edited. By clicking on **Edit** it is possible to configure the relation between the controller outputs (**U** vector) and the servo movements (**S** vector).



SU matrix editing

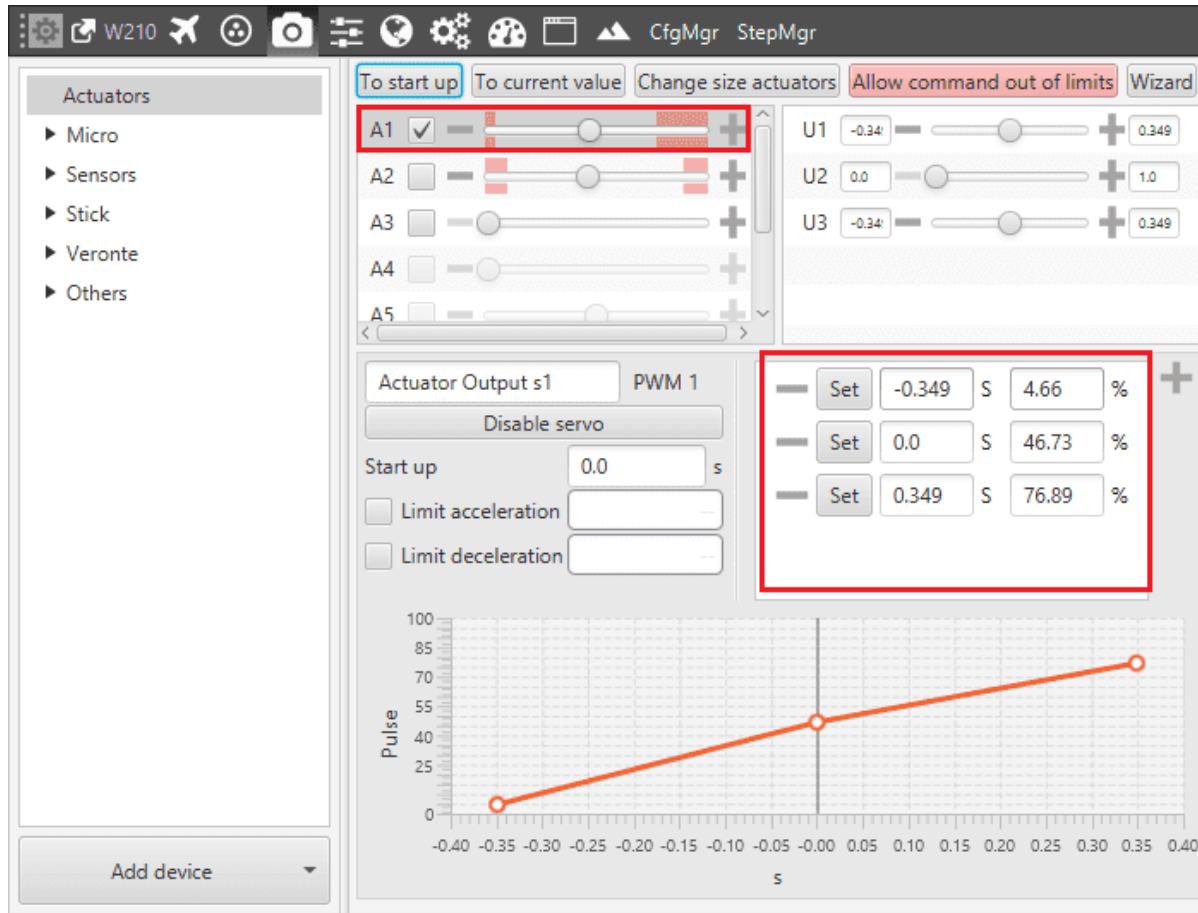
The W210 is configured as follow:

1. **Pitch Control:** control surfaces must be moved in the same direction to modify the pitching angle. The contribution of the actuators has same magnitude and direction.
2. **Thrust Control:** the Actuator Output 1 is the only one that allows a thrusting change.
3. **Roll Control:** in this case, the contribution of the actuators must be set with the same magnitude and reverse direction in order to perform a rotation around the X axis.

Warning: This panel shows the reference system of the aircraft. It must be positioned in the same way of the Autopilot's one. If it results different, it can be edited by clicking on the corresponding axis in order to reverse its direction.

12.2.1.2.1.3 System Trim

As a final step, the system has to be trimmed. This can be performed by moving servos in three different positions: zero position, minimum and maximum deflection angle (angles are usually limited physically). These positions must be inserted and saved in the software by clicking on **Set** when the actuator is in the desired position. Otherwise, position can be introduced manually.



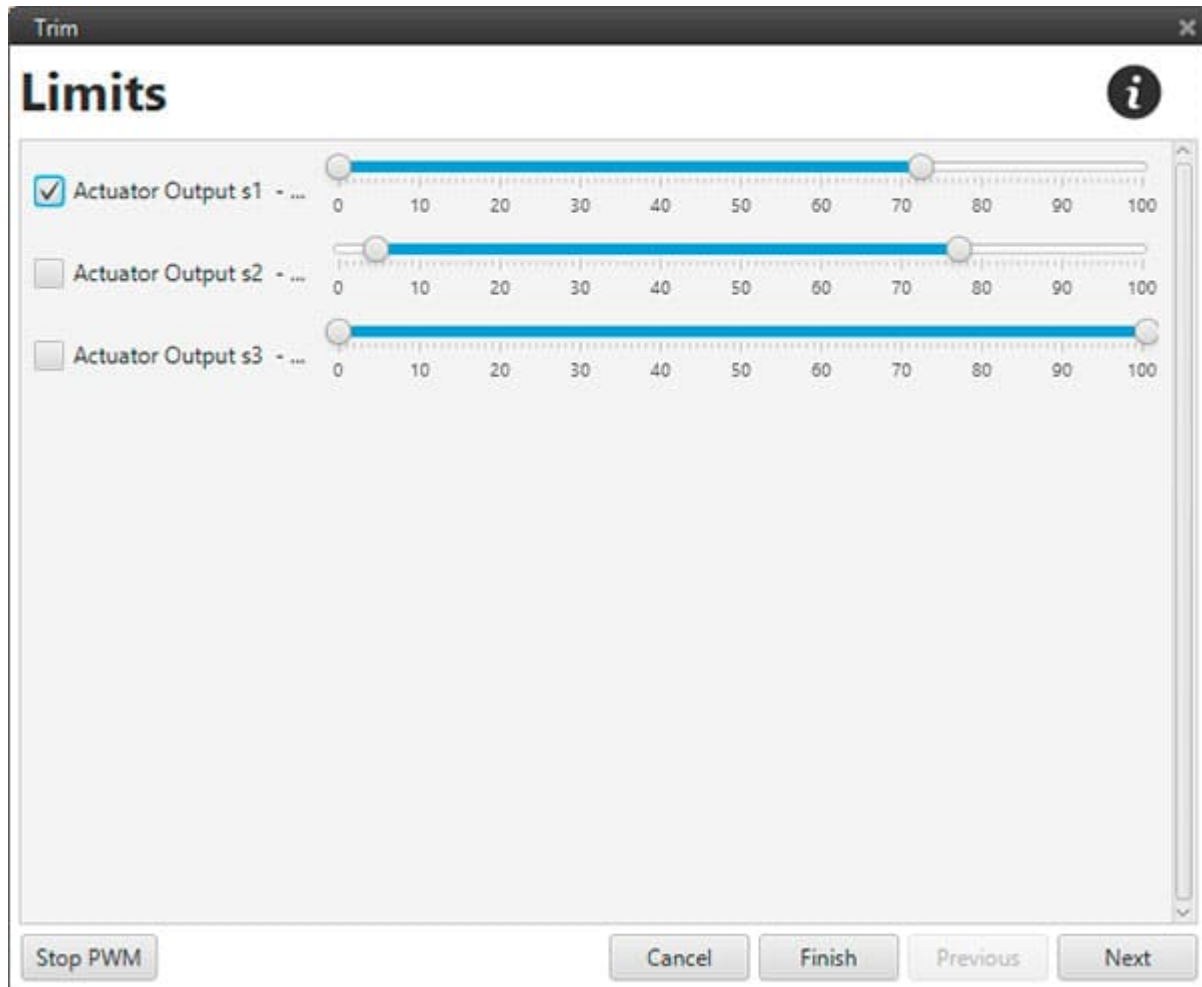
Actuator 1 Trimming

The picture shows the setting of the elevator number 1:

- **Minimum:** -0.34906 [rad] deflection; 4.6682%
- **Zero position:** 0.0[rad] deflection; 46.7317%
- **Maximum:** 0.34906 [rad] deflection; 76.8904%

Warning: The actuators can be moved directly from VerontePipe only when the system is in an Initial phase. During the actuator run, if the desired position is in the **Out of range** zone (red zone), it is possible to click on **Allow command out of limits** in order to move completely the actuator and find the correct position.

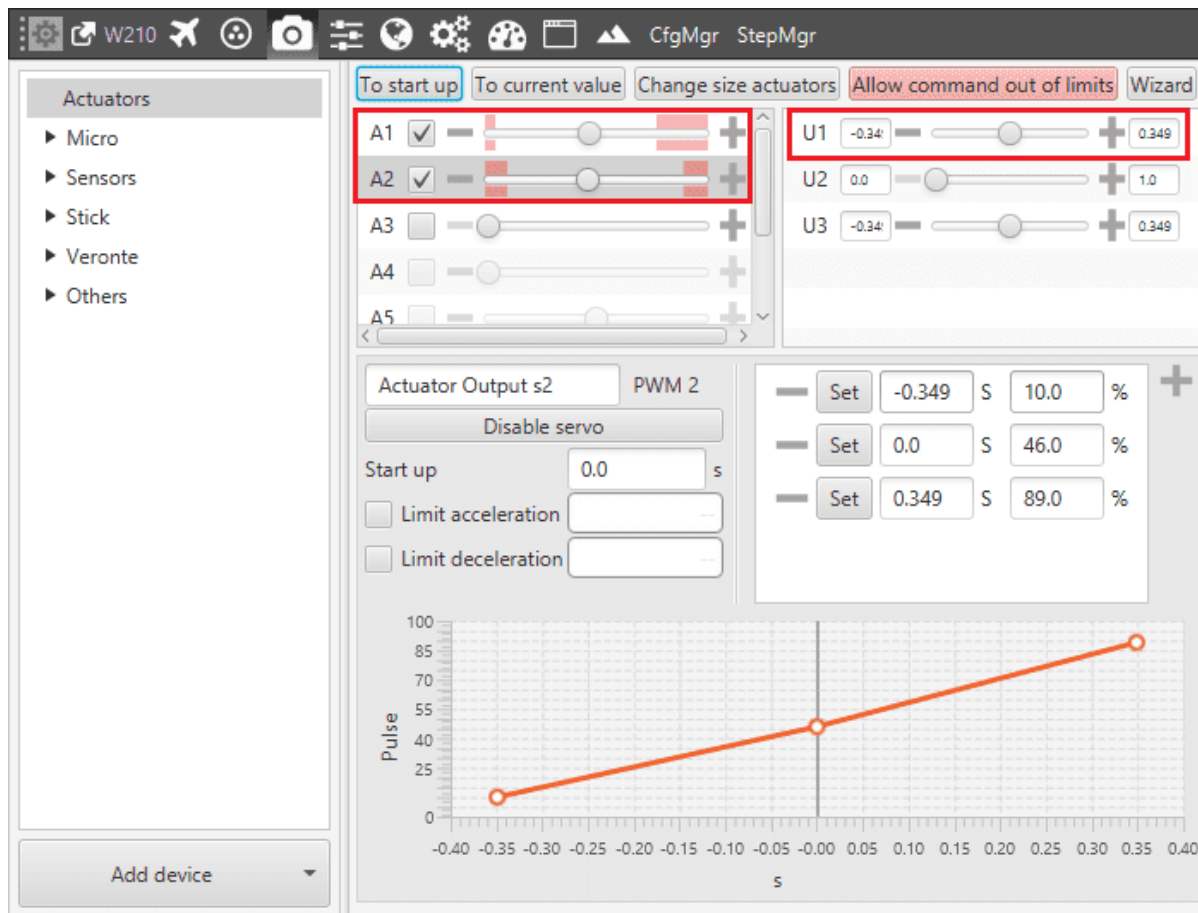
This procedure can be performed in the same way by using a **Wizard**. This tool allows moving actuators limits easily and finding the correct range.



Trim wizard tool

In order to perform a final checking, it is possible to select the desired channel and testing pitch, roll and thrust controls.

The image below shows a pitching output testing. By moving the U1 control, surfaces must change the position according to the reference system: positive corresponds to nose down and negative to nose up.



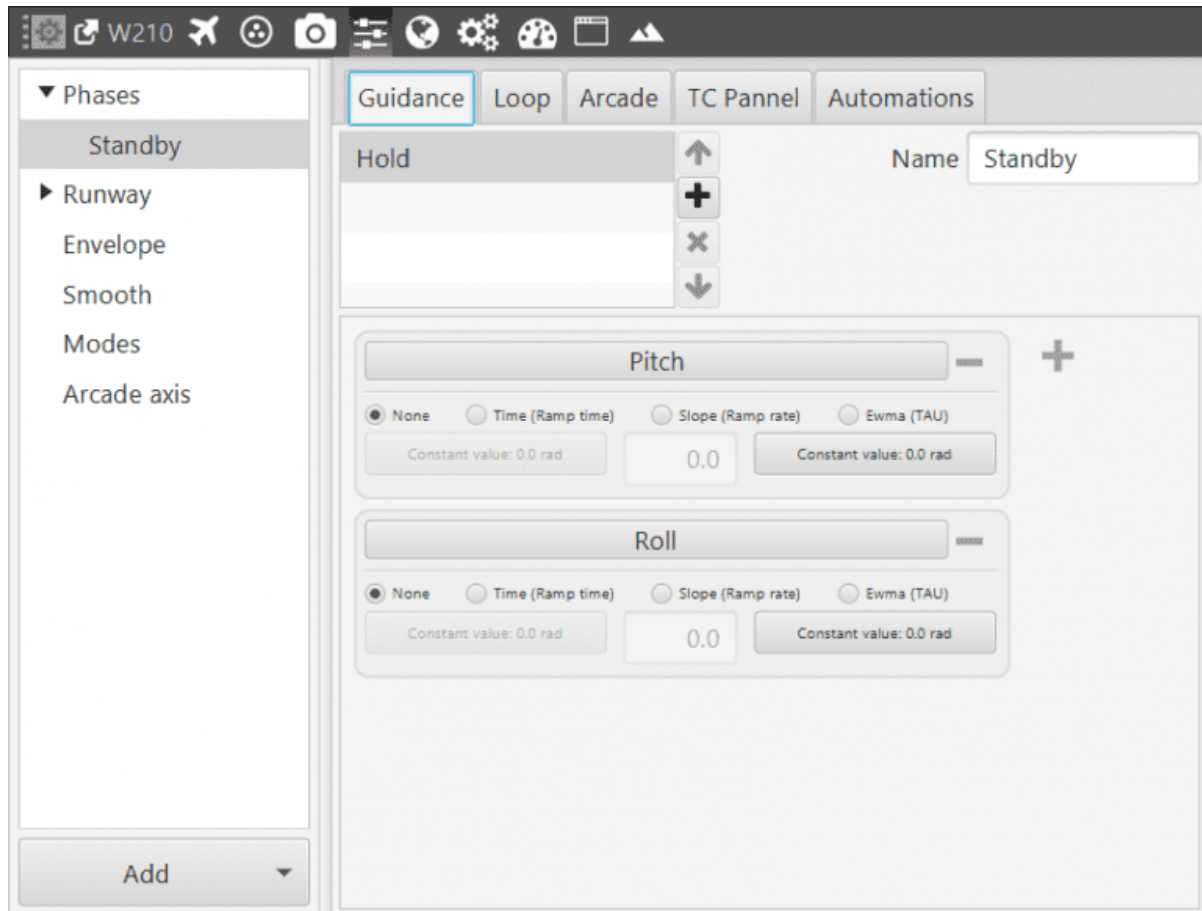
Pitching test

12.2.1.2.2 Mission Phases

In this section, it will be explained the W210 typical mission profile and the mission phases will be detailed.

12.2.1.2.2.1 Standby

Standby phase is a preliminary phase of the operation. During this phase is possible to check, for example, if the aircraft is correctly controlling the attitude by change it and seeing the control surfaces moving.



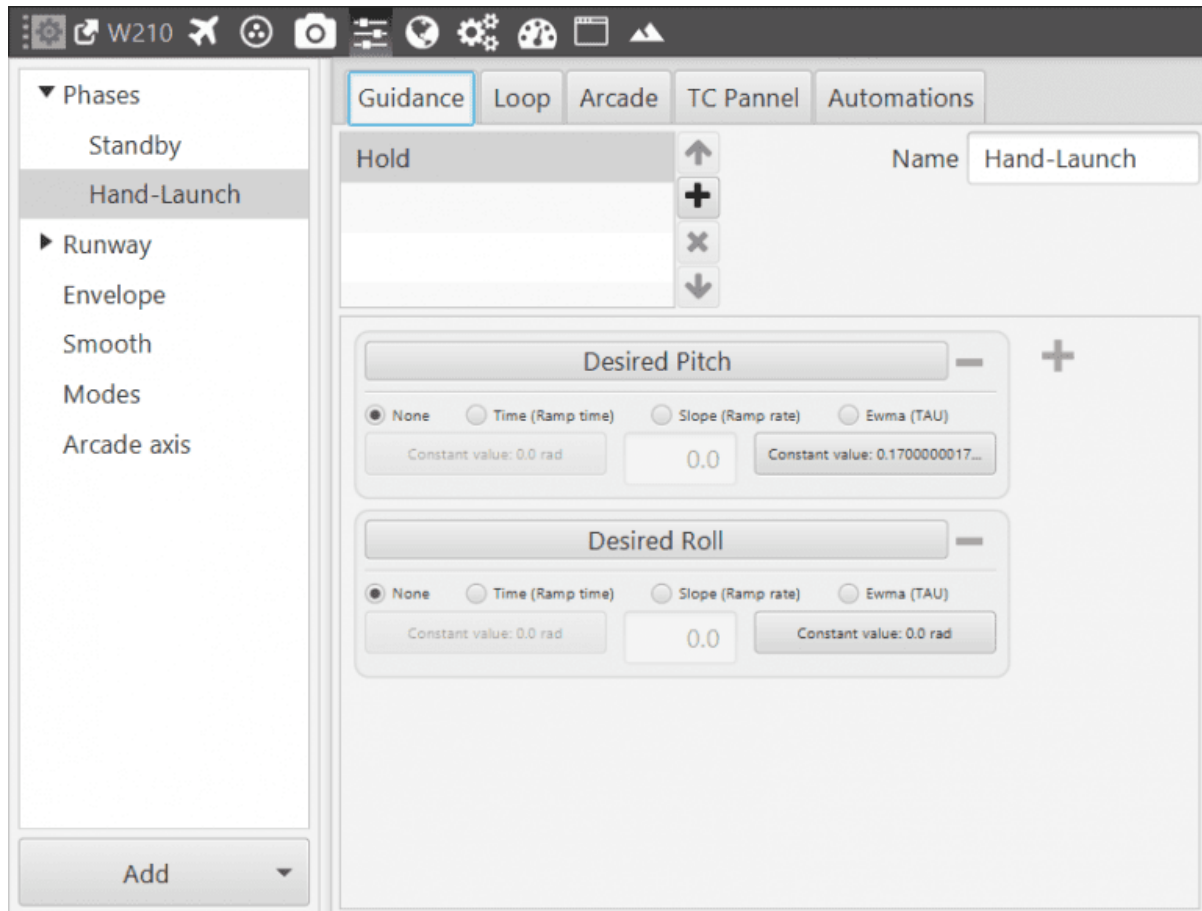
Standby phase panel

The guidance is a Hold of two angles:

- **Pitch and Roll:** kept at 0 [rad].

12.2.1.2.2.2 Hand-Launch

Hand-Launch phase is the first launch modality. In this case, the aircraft is launched “manually” by a person who has to increase its velocity trying to maintain the platform in a correct attitude. The image below shows the Hand-Launch phase configuration panel.



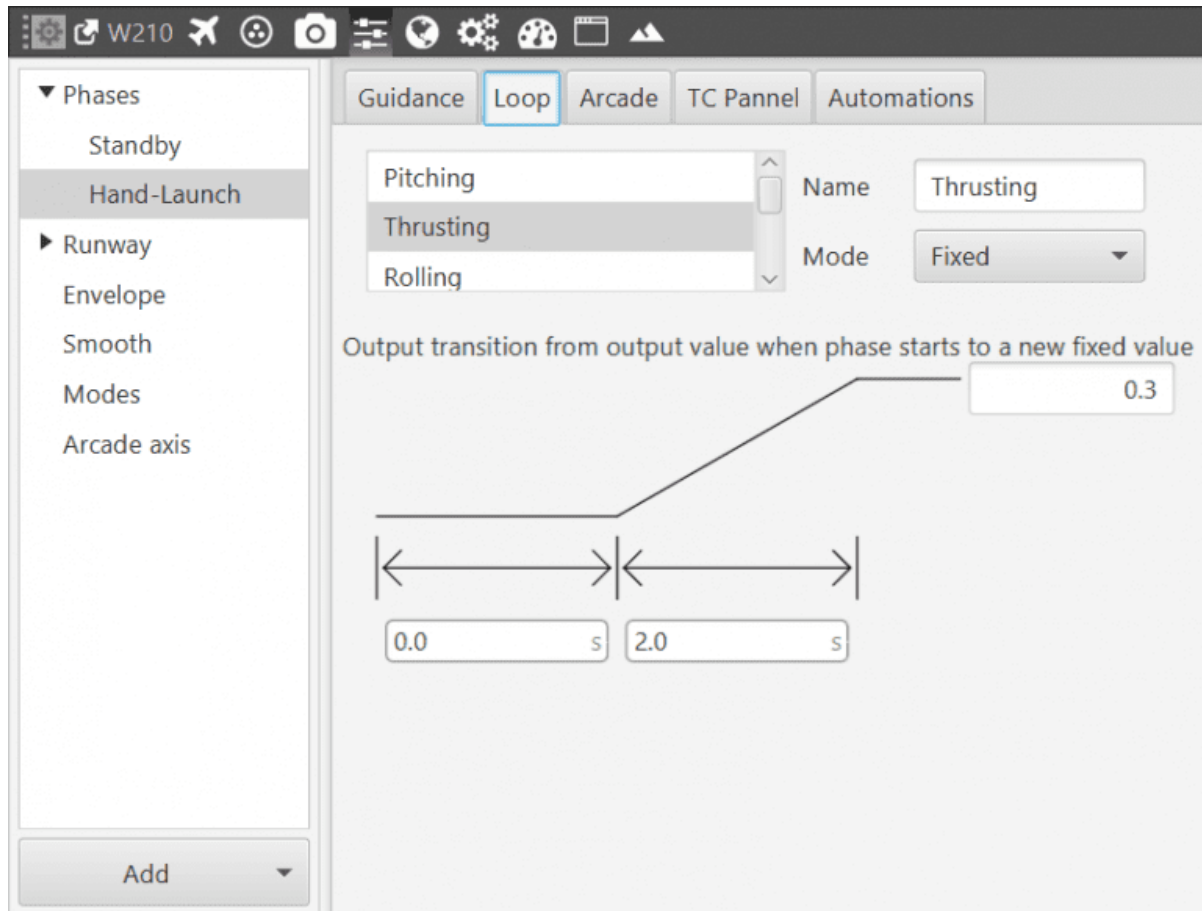
Hand-Launch phase panel

In this phase, the Guidance is a Hold of two variables:

- **Desired pitch:** at 0.17 [rad].
- **Desired roll:** at 0.0 [rad].

It means the platform control system will keep to zero the roll angle and the pitch angle to 10° degrees.

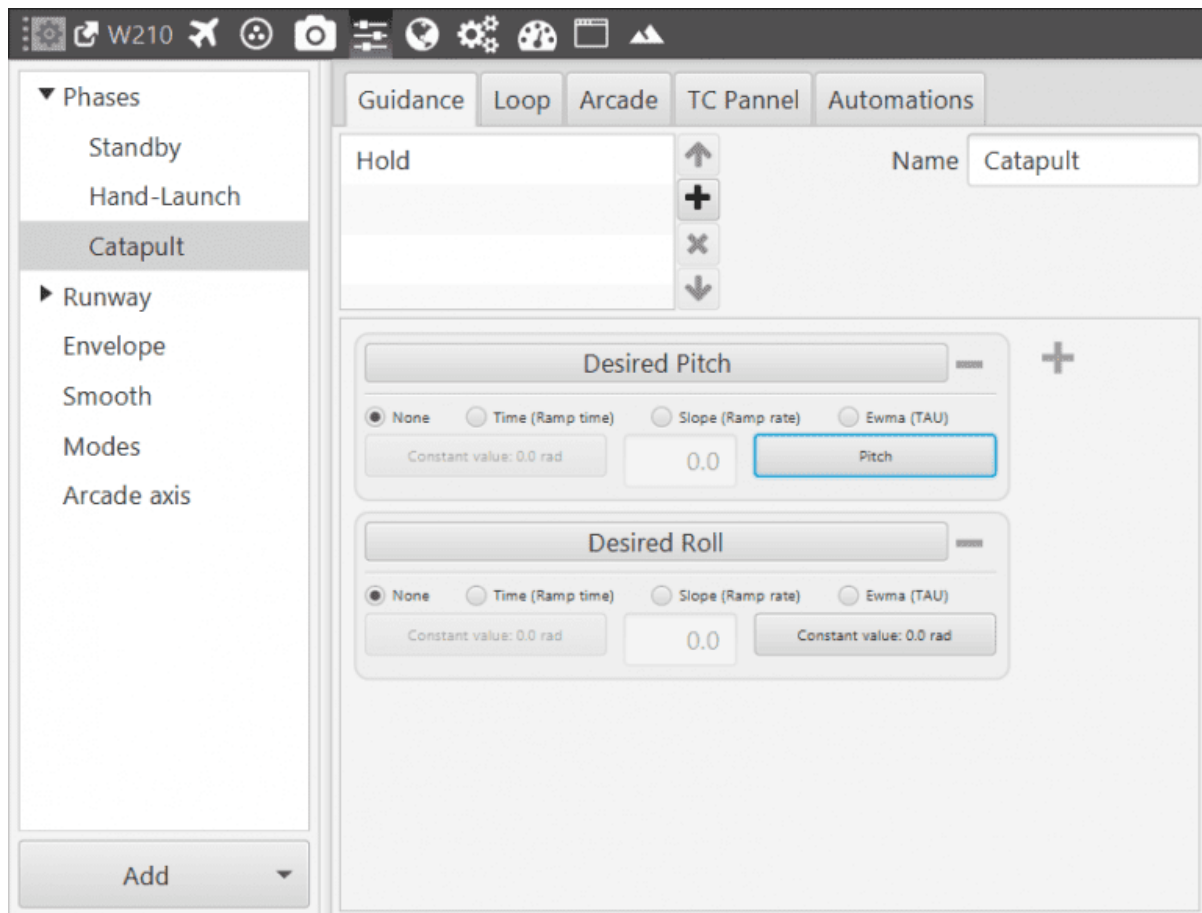
Furthermore, during this phase the motor starts. The image below shows the thrust behavior: during the Hand-Launch phase, thrust starts from zero and reaches the fixed value of 0.3 in 2 seconds.



Hand-Launch phase thrust

12.2.1.2.2.3 Catapult

Catapult phase is the second launch modality. The takeoff is performed using a catapult which allows reaching the desired speed and maintaining the correct attitude during the launch. The image below shows the Catapult phase configuration panel.



Catapult phase panel

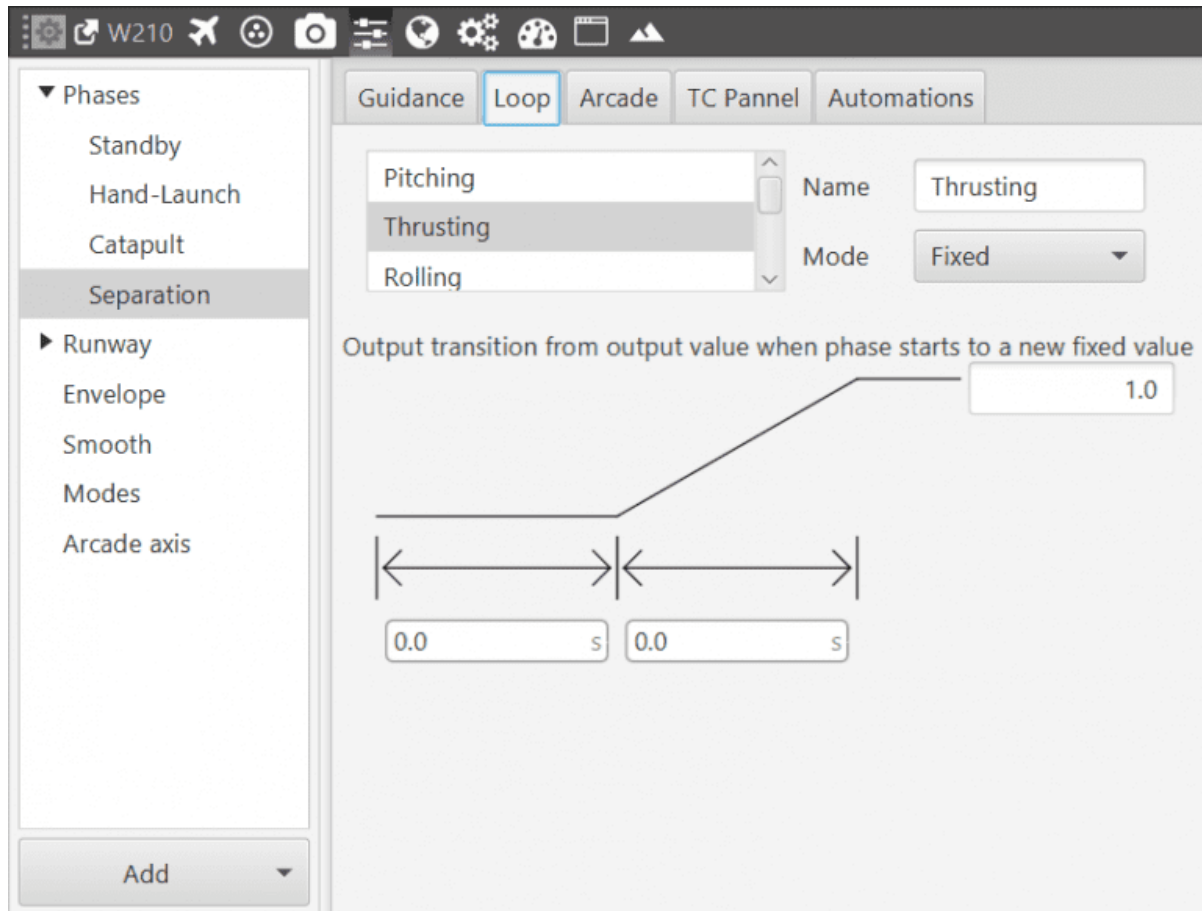
As happens in the Hand-Launch phase, the Guidance is a Hold of twop variables:

- **Desired Pitch angle:** at the current pitch angle.
- **Desired Roll angle:** at 0.0 [rad].

Thrusting, in this phase, is kept at zero.

12.2.1.2.2.4 Separation

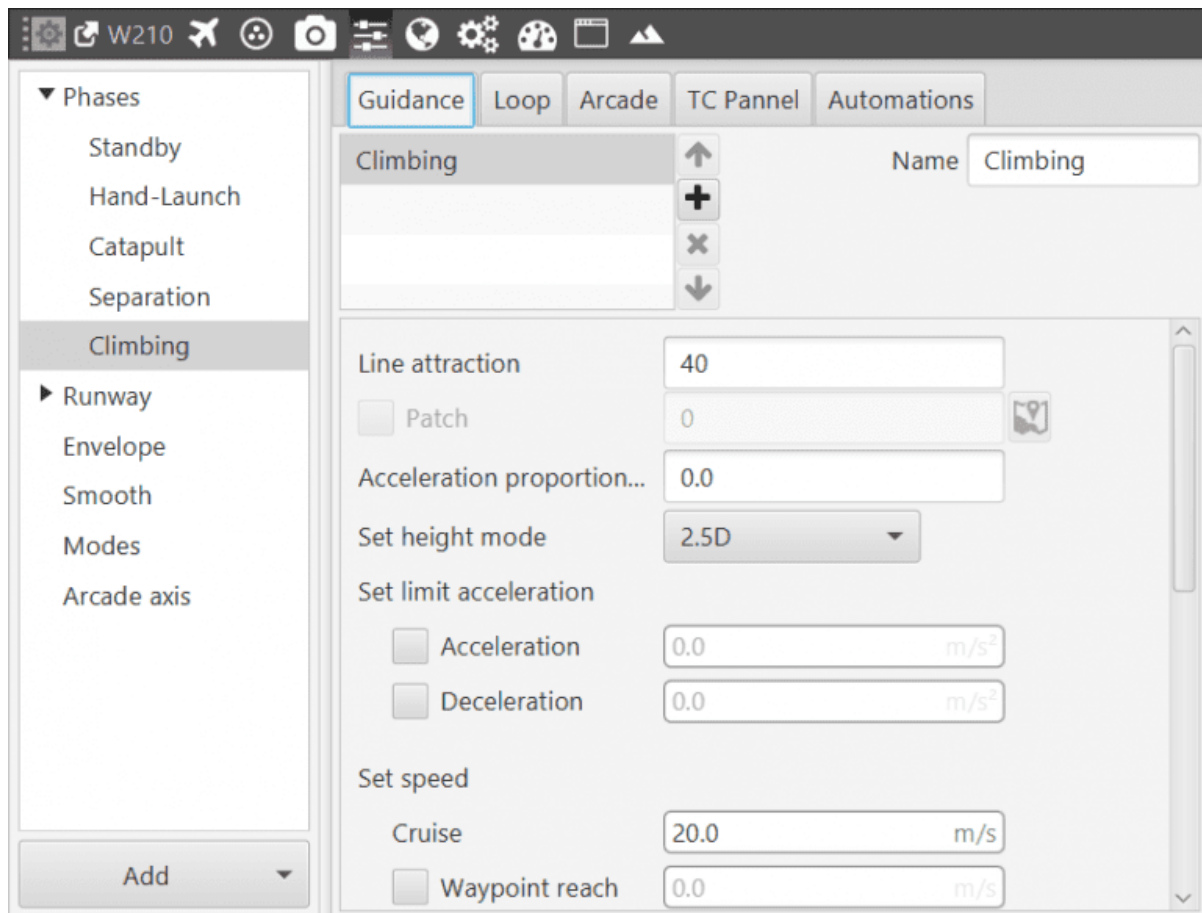
Separation is the second phase of this takeoff modality. Guidance is the same **Hold** configuration of the Catapult phase but Thrusting changes. In fact, Thrusting is an instantaneous step from 0 to 1 in order to switch on the motor only when the fan can not be able to hit the catapult.



Separation thrusting

12.2.1.2.2.5 Climbing

Climbing phase is configured to make the airplane reach the mission altitude after the takeoff.



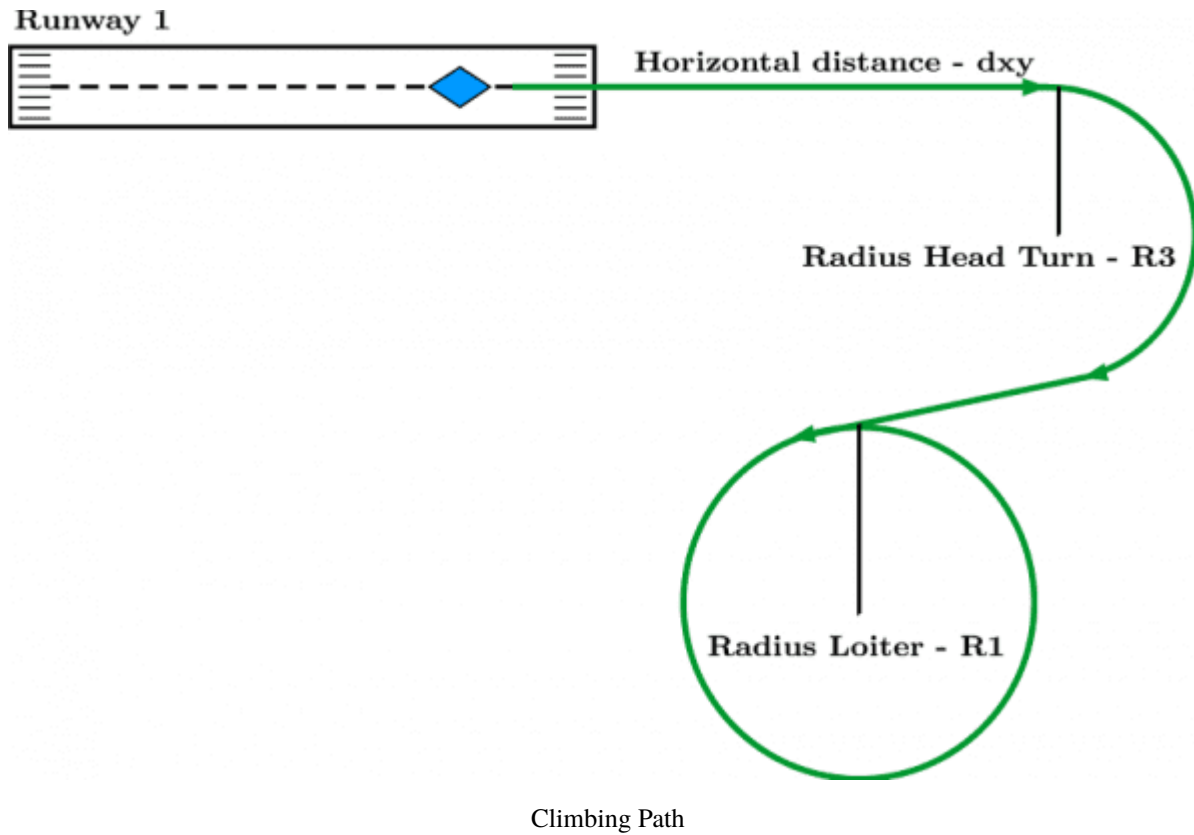
Climbing phase panel

This phase consists of a **Climbing** Guidance:

- **Line attraction:** value related to how strongly the aircraft tries to reach a path (climb path in this case). This value is commonly between 20 and 40 for airplanes. In this case, the value is set at 40.
- **Set speed:** speed that will have the airplane during the climb, in this case 20 [m/s] is a good value.

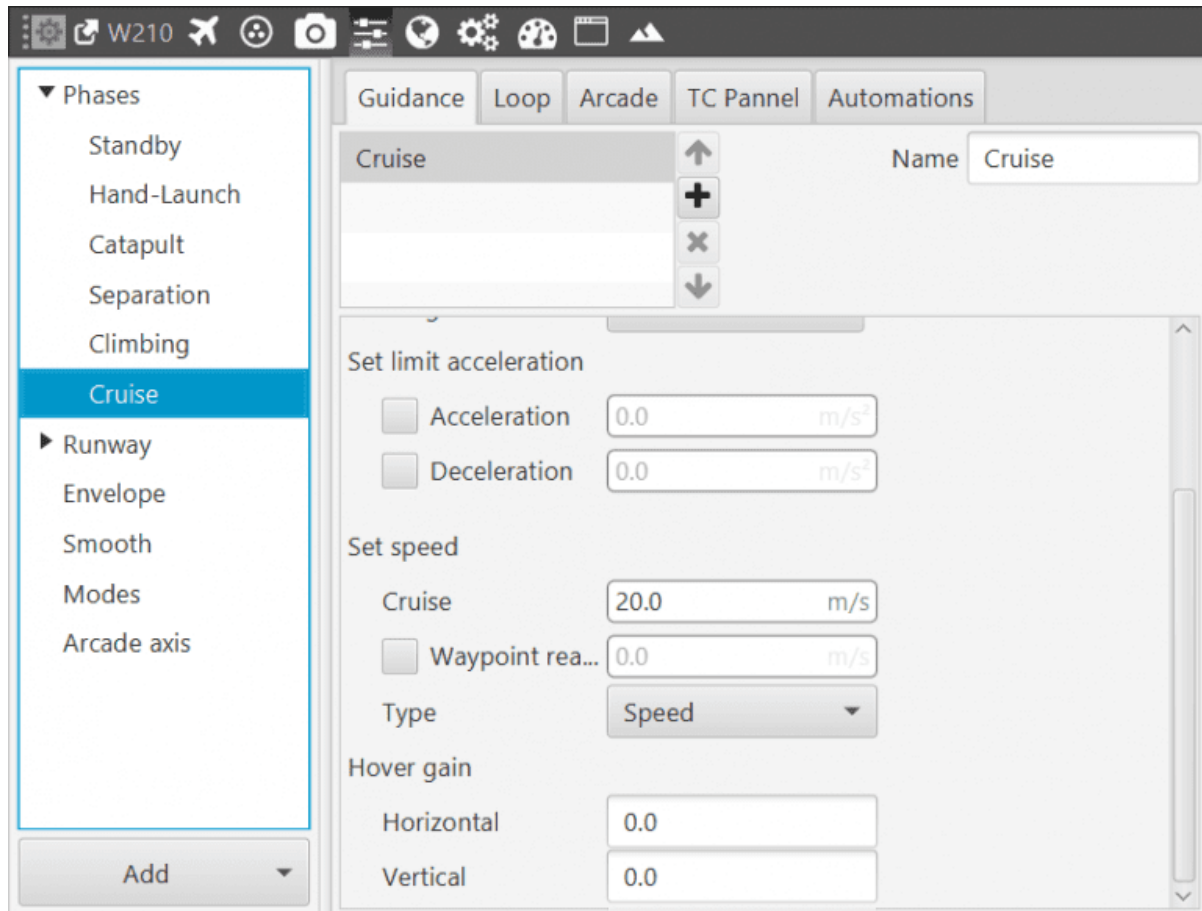
In this phase, the aircraft reaches the mission altitude by performing a spiral route. In order to determine the correct path, these parameters must be set:

- **Runway:** here is selected the runway which previously has been edited in its configuration menu.
- **Flight Path Angle:** angle at which the aircraft will climb, 0.1396 [rad].
- **Horizontal Distance:** is the distance from the point where the aircraft enters in the phase which contains the climbing guidance, to the start of the circular climbing path, 100 [m].
- **Radius Head Turn R3:** radius of the turn made to head the airplane towards the loiter direction, 100 [m].
- **Radius loiter R1:** radius of the loiter ascending made by the aircraft to reach an altitude suitable, 100 [m].



12.2.1.2.2.6 Cruise

In this phase, the Guidance is **Cruise**. The aircraft follows a route marked by a set of waypoints, which are defined by the user in the **Mission** menu.



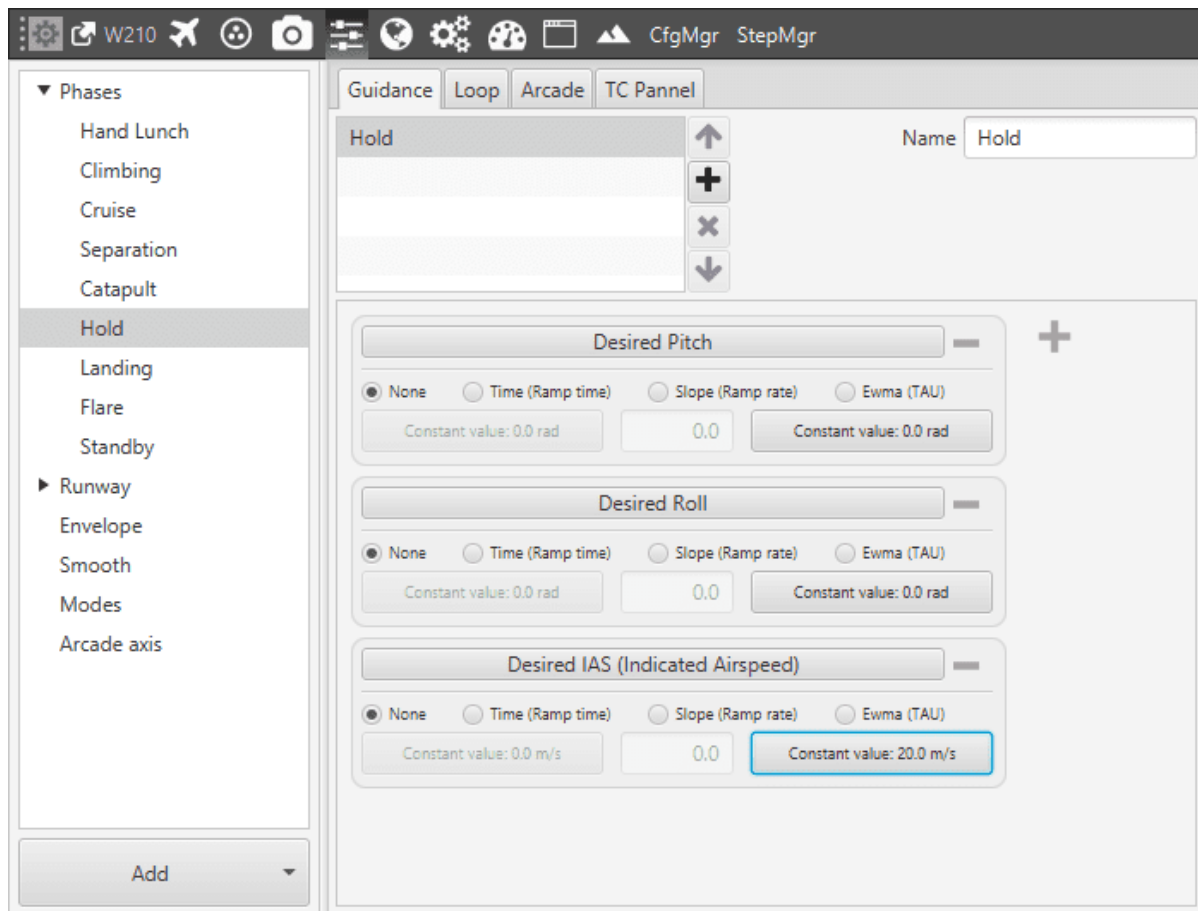
Cruise phase panel

Parameters set here are the same as the ones from the climbing phase.

- **Line attraction:** 40.
- **Set Speed:** 20 [m/s].

In this guidance, there is an option related to the gains used to recover the hover point in an multicopter, **Hover Gain**. In this case, the platform configured is an airplane, so this option will not be used.

12.2.1.2.2.7 Hold



Hold phase panel

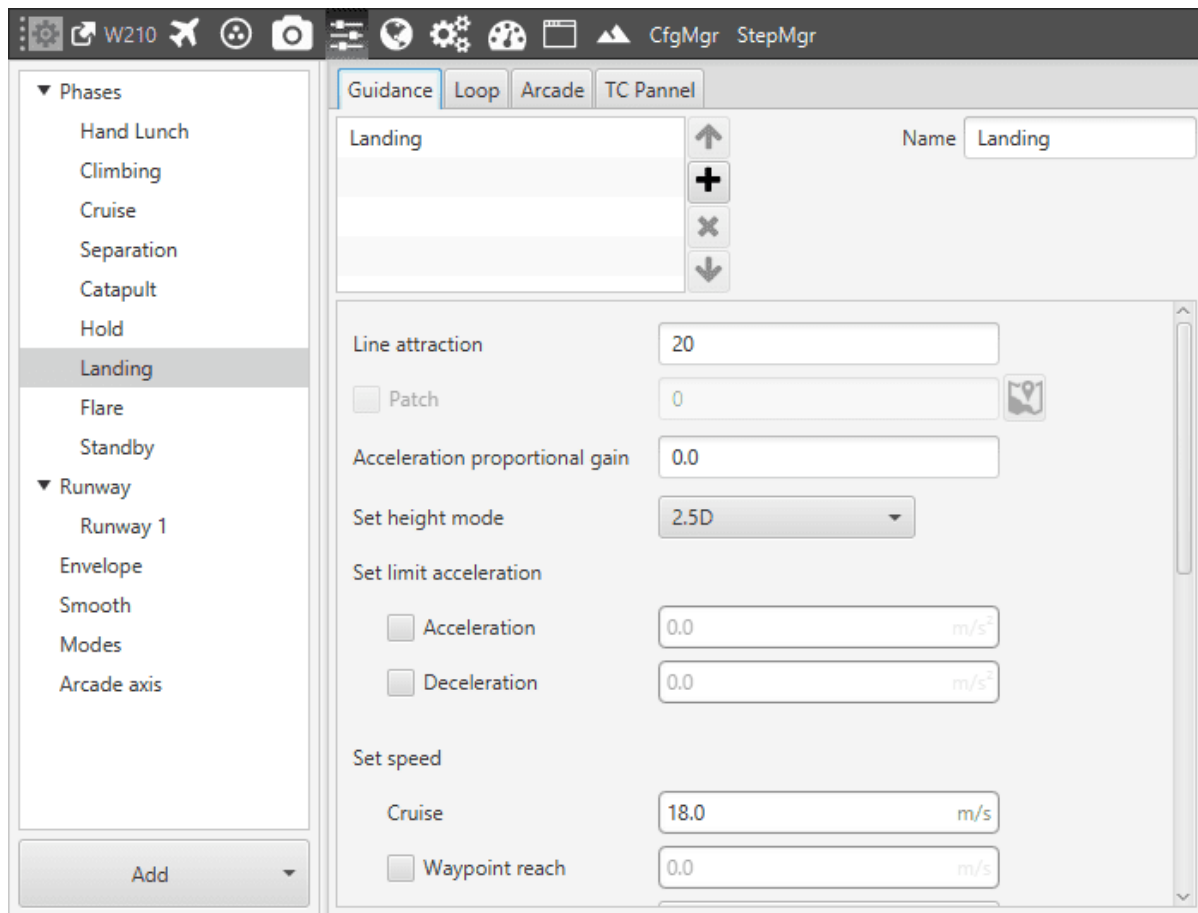
In this phase, the Guidance is a **Hold** of three variables:

- **Desired Pitch:** at 0 [rad].
- **Desired Roll:** at 0 [rad].
- **Desired I.A.S:** (Indicated Air Speed) kept at a constant value of 20 [m/s]

It means the aircraft will maintain this attitude until the next phase change.

12.2.1.2.2.8 Landing

This phase is used to make the aircraft land at a certain airport. Also, when the flight altitude is too big, this phase contains the parameters which define the route performed by the platform to descent until an altitude where it can line up with the runway.



Landing phase panel

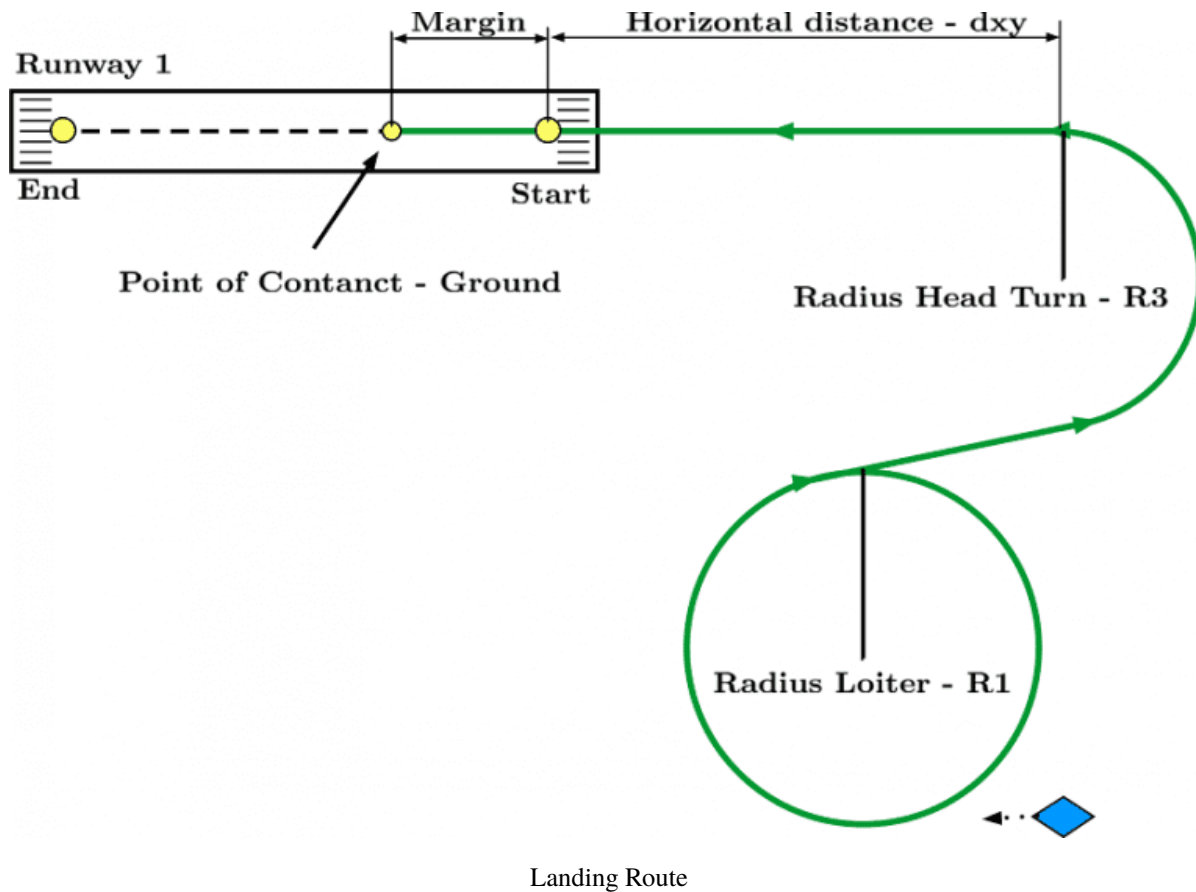
The guidance to be configured is Landing:

- **Line Attraction:** value related to how strongly the aircraft tries to reach a path, kept at 20.
- **Set Speed–Cruise:** 18 [m/s]

Finally, the following parameters define the path during this phase.

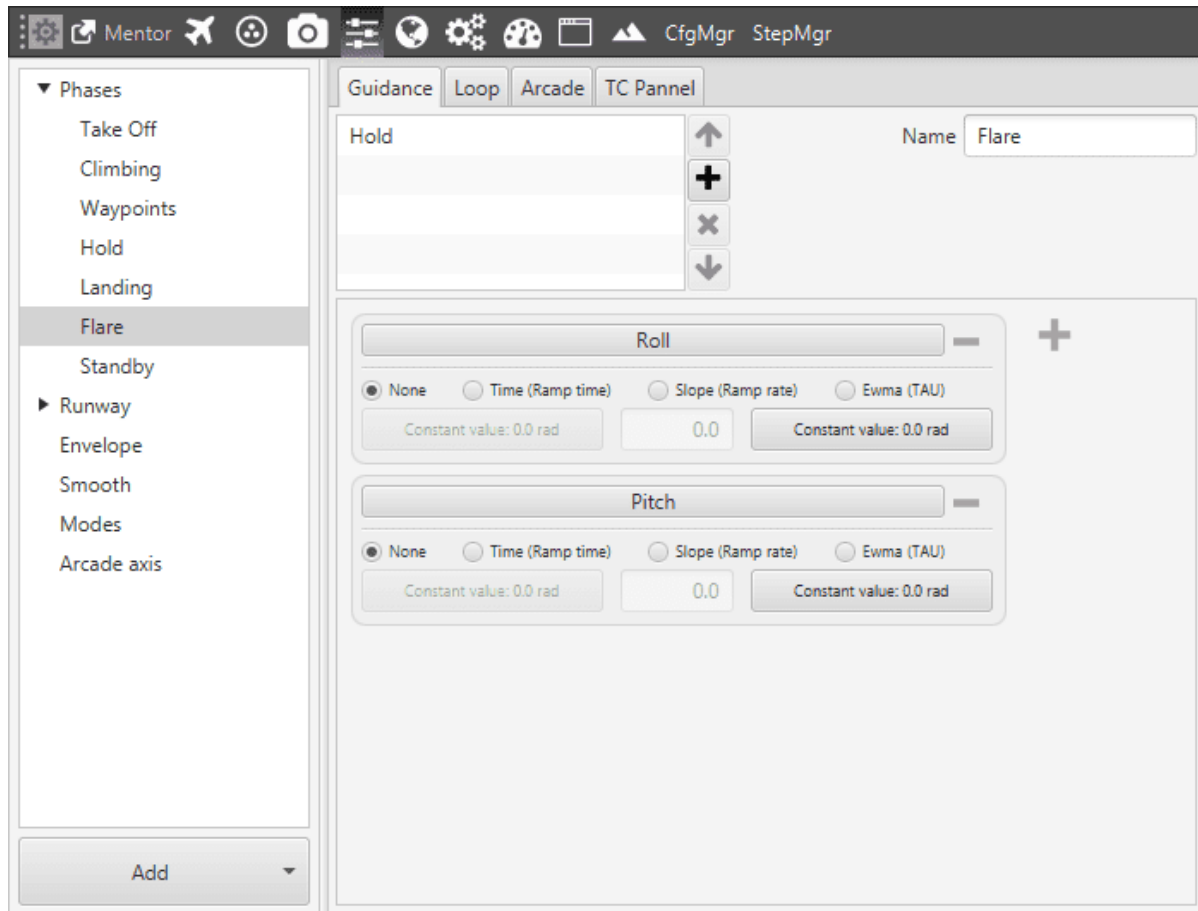
- **Runway:** here is selected the runway which previously has been edited in its configuration menu.
- **Flight Path Angle:** angle at which the aircraft will descend, -0.14 [rad].
- **Horizontal Distance:** is the distance from the point where the aircraft enters in the phase which contains the climbing guidance, to the start of the circular climbing path, 300 [m].
- **Radius Head Turn R3:** radius of the turn made to head the airplane towards the runway direction, 150 [m].
- **Radius loiter R1:** radius of the loiter descending made by the aircraft to reach an altitude suitable to perform the landing manoeuvre, 120 [m].

The route of this phase is shown in the following figure, where each one of the parameters that define it are defined.



12.2.1.2.2.9 Flare

Flare phase allows defining a point or a zone near the runway where the aircraft will perform a pitch angle change (nose-up) in order to modify its attitude before the touchdown and avoid a possible crash due to a nose-ground direct contact.



Flare phase panel

- **Roll – Pitch Angle:** kept at a constant value 0 [rad].

Regarding the thrust in this phase, the engines are shut off, so the mode of the controller is **off**.

12.2.1.2.3 L200 Catapult



L200 Catapult

From the Embention website it is possible to download [User Manual](#) and the [Datasheet](#) of the L200 Catapult.

12.2.1.2.3.1 Configuration

In order to configure correctly the launch system, it is necessary to perform two different actions:

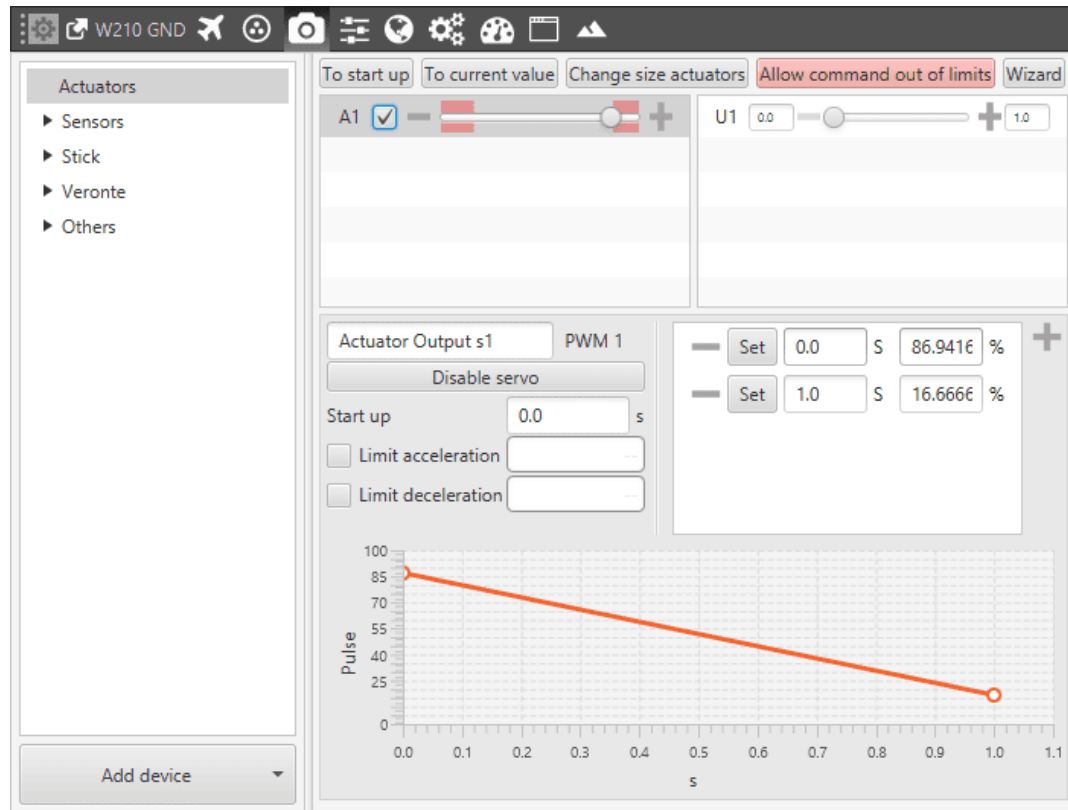
- Trimming of the launch actuator.
- Create the automation for the launch button.

To set them, it is necessary to connect the catapult to Veronte Ground using the catapult wire and completing the system with a PC connection with Veronte Pipe.

12.2.1.2.3.2 Actuator Trimming

To trim the actuator it is necessary to create the catapult actuator by following the path **Devices>Control>Actuators**.

1. Click on **Change size actuators** and select the actuator number (1) from the window.

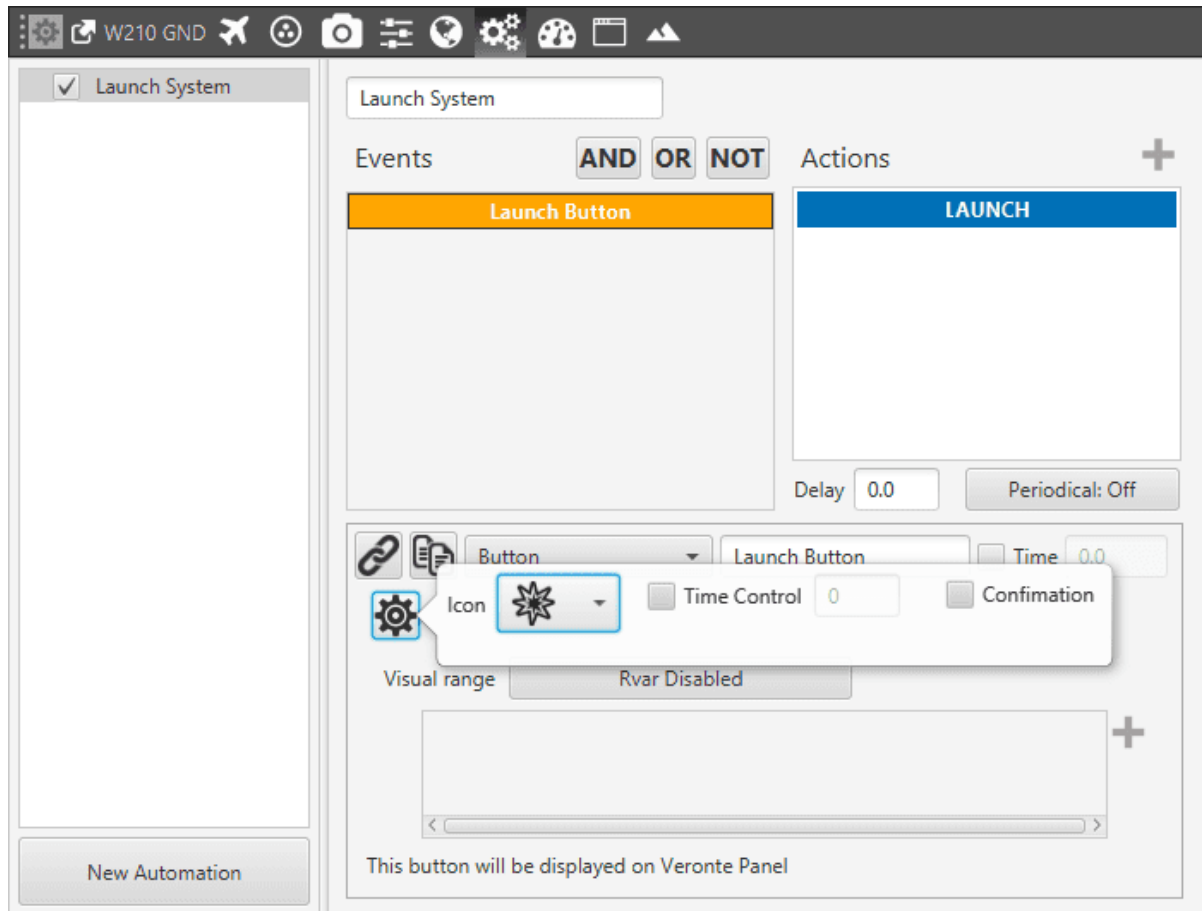


Catapult actuator trimming

2. Once created an actuator, it is possible to trim it. Default values are shown in the previous figure, but user must be sure the servo can move covering all useful range, from bottom to top (0-1).

12.2.1.2.3.3 Launch Automation

The automation must be created in the Ground autopilot. The typical Event of the automation is a Button which will appear in the Ground Veronte Panel.

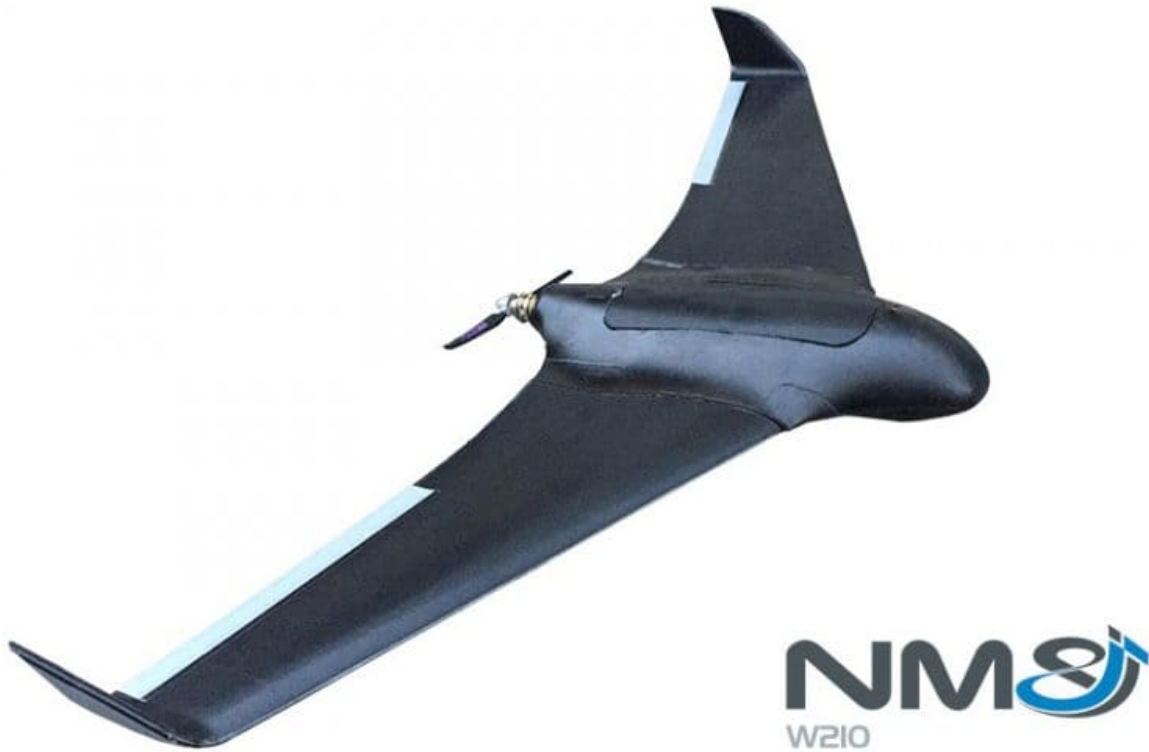


Automation Event

The Action is the change from 0 to 1 of the actuator value. A second value of the actuator should be set in order to restore the initial null value (in this case, after 2 seconds).

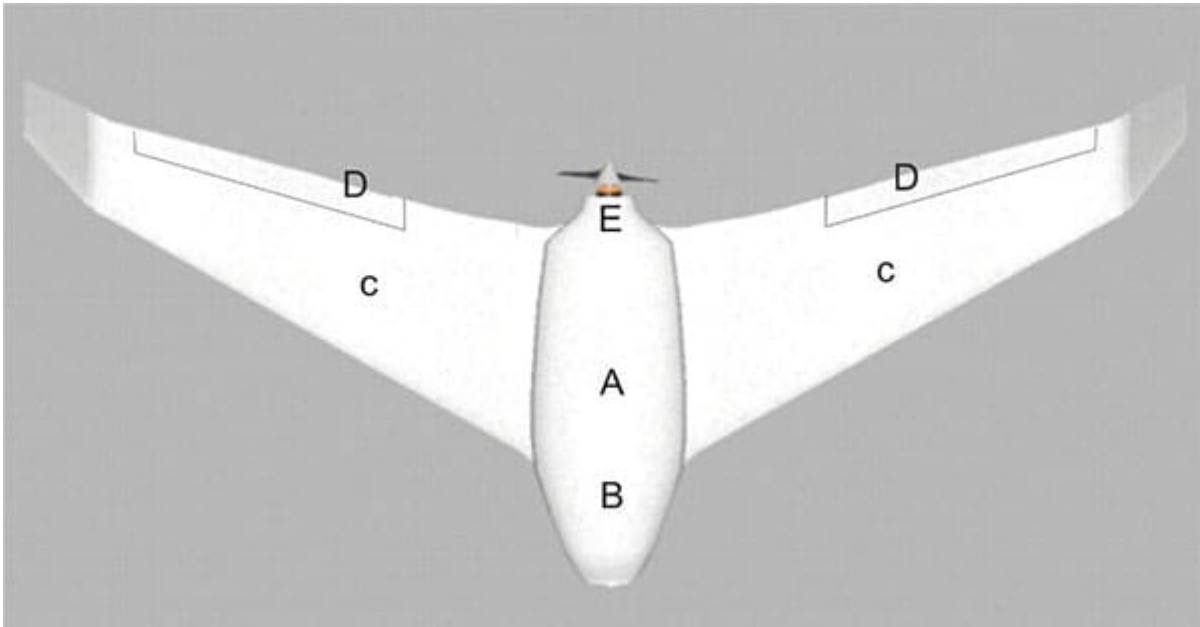
The screenshot shows the 'Automation Action' configuration window in the Veronte Autopilot software. The window has a dark grey header bar with various icons and the text 'W210 GND'. On the left, there is a sidebar with a checked box labeled 'Launch System' and a 'New Automation' button at the bottom. The main area is divided into two columns: 'Events' and 'Actions'. The 'Events' column contains a blue button labeled 'Launch Button'. The 'Actions' column contains an orange button labeled 'LAUNCH'. Between these columns are logical operators: 'AND', 'OR', and 'NOT'. Below the 'Events' column, there is a 'Delay' field set to '0.0' and a 'Periodical: Off' button. At the bottom, there is a section for 'Output' with a dropdown menu showing 'Output' and a text field containing 'LAUNCH'. Below this is a dropdown for 'Actuator Output s1'. There is also an 'Interpolate' checkbox. A table with two rows is visible, showing 'Time' and 'Value' pairs: the first row has 'Time' 0.2 and 'Value' 1.0, and the second row has 'Time' 2.0 and 'Value' 0.0. A plus sign is next to the table.

Automation Action



W210

This airplane has a “flying wing” structure and can be piloted by using only two symmetrical control surfaces. They can be moved to control the aircraft attitude in a symmetrical way to perform a pitch variation and in an unsymmetrical way to control the roll.



W210 parts in XPlane model

- Power battery

- Control electronics
- Detachable wings
- Control surfaces (ailerons)
- Power motor and propeller

12.2.1.3 Quadcopter-M400

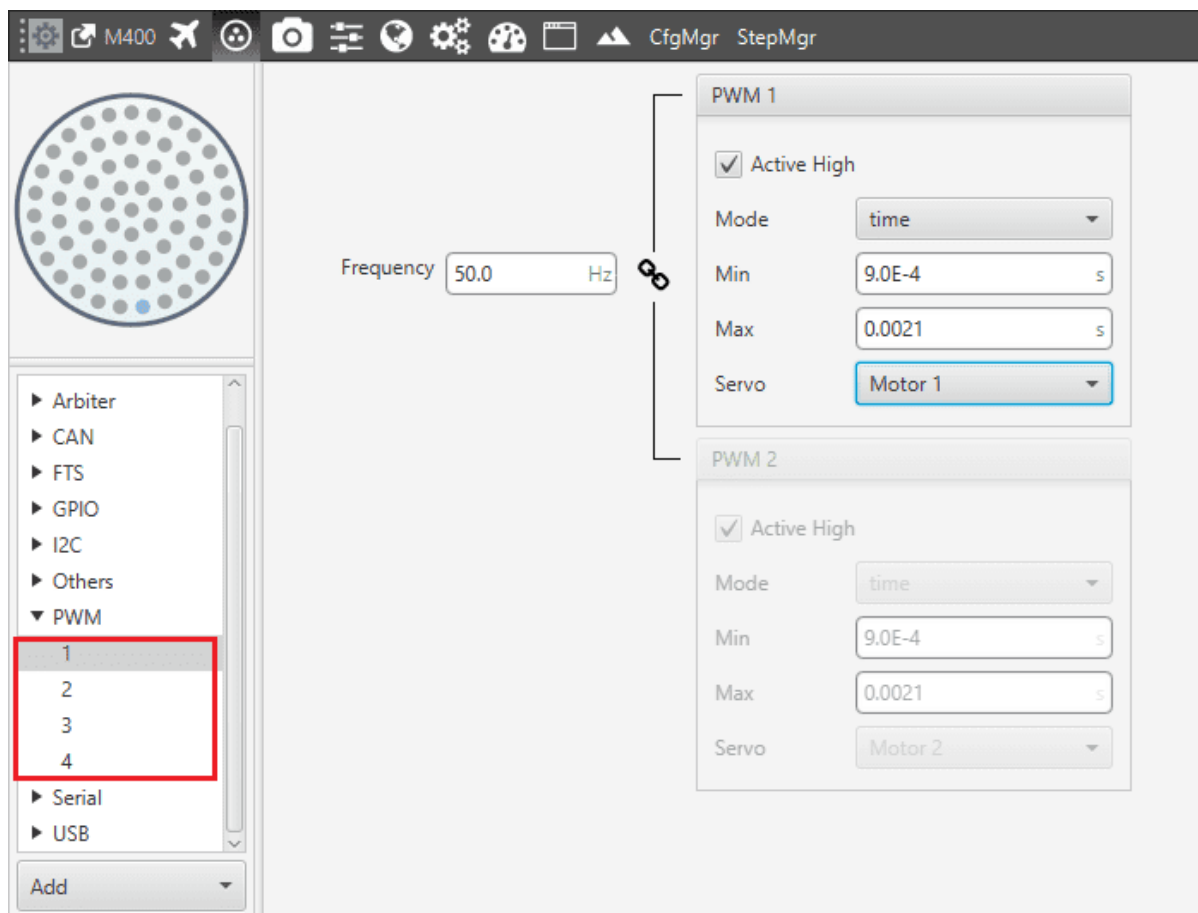
12.2.1.3.1 Servo Configuration

The M400 configuration process can be performed by using a VerontePipe software version connected with the hardware system and the Autopilot as explained in the section [Hardware Installation](#) of the manual.

12.2.1.3.1.1 Servos Output

The first step of the process is the servos configuration.

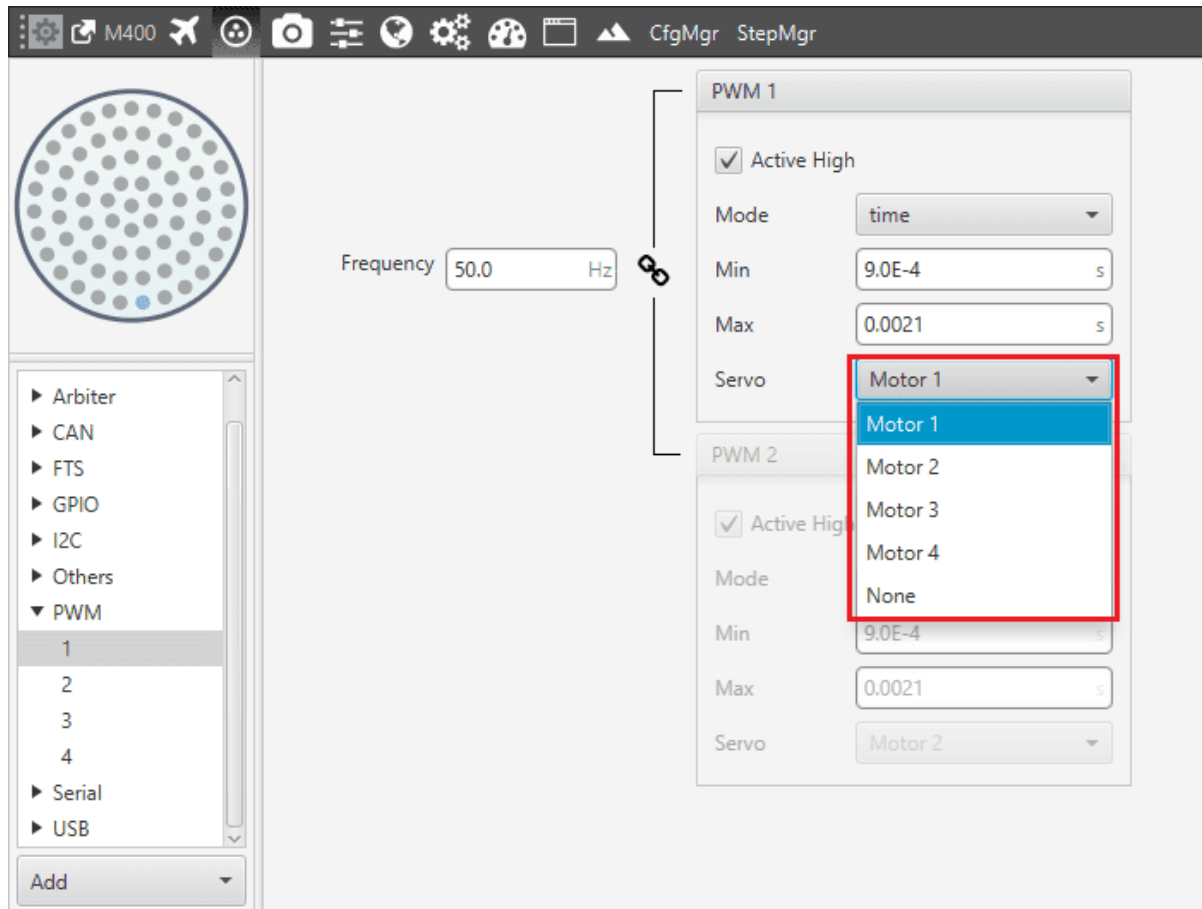
In this case, the controls of the airplane are the four electric motors. Motors can be controlled by changing their thrust, so each one of them corresponds to a pin of the connector and they must be positioned in the same order in an S vector who represents the **Actuator Output**. It is possible to connect any pin to any command but the easiest way to perform it and avoid confusion is by following the pins number.



Output-pin connections

In this case, it is possible to use 4 pins:

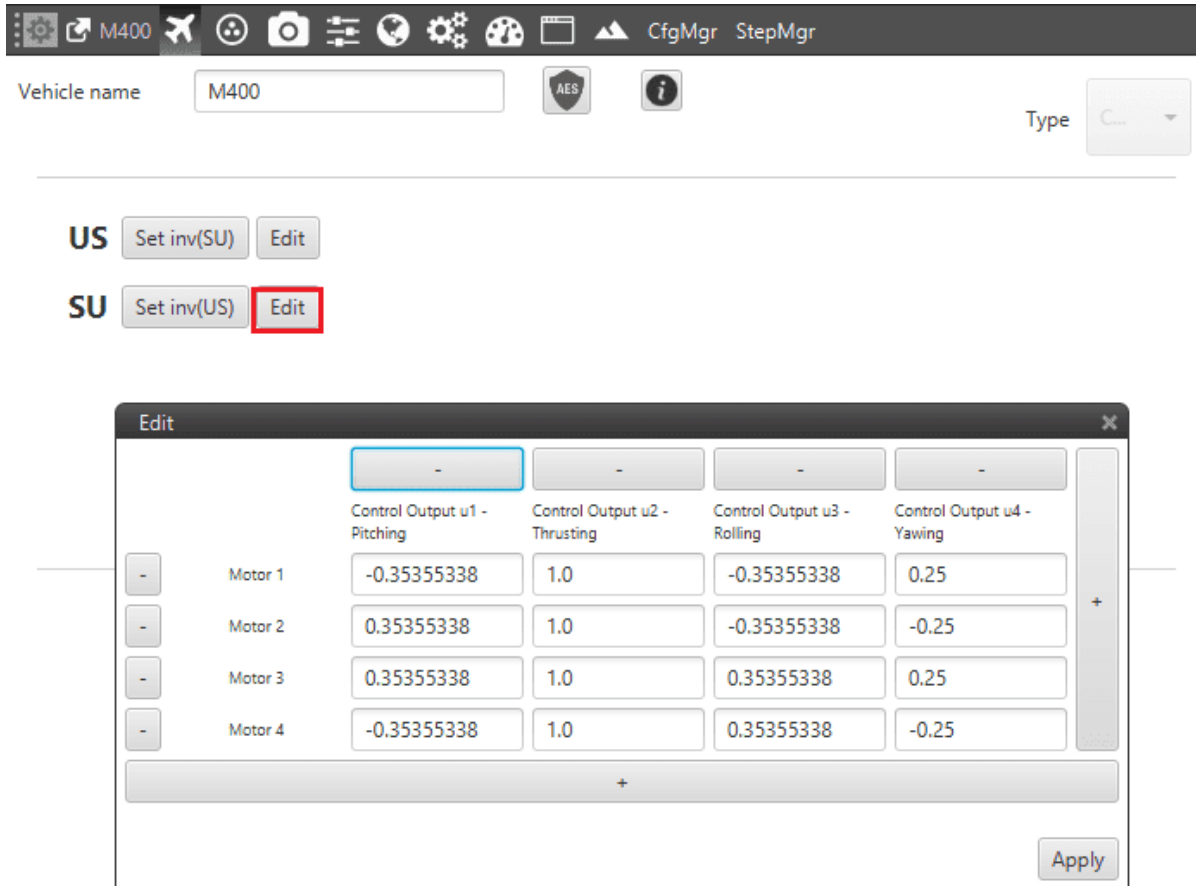
- Output 1 – Actuator Output s1 (Motor 1)
- Output 2 – Actuator Output s2 (Motor 2)
- Output 3 – Actuator Output s3 (Motor 3)
- Output 4 – Actuator Output s4 (Motor 4)



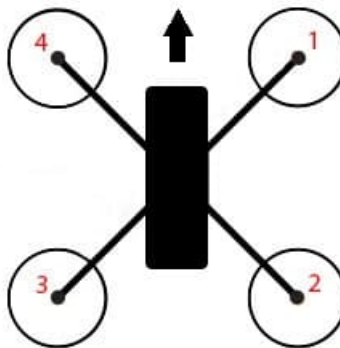
Output 1 – pin 1 configuration

12.2.1.3.1.2 SU Matrix

At this point, the **S** vector is defined and the **SU** matrix can be edited. By clicking on **Edit** it is possible to configure the relation between the controller outputs (**U** vector) and the servo movements (**S** vector).



Output 1 – SU matrix edit



SU matrix and motors numbers

The M400 is configured as follow:

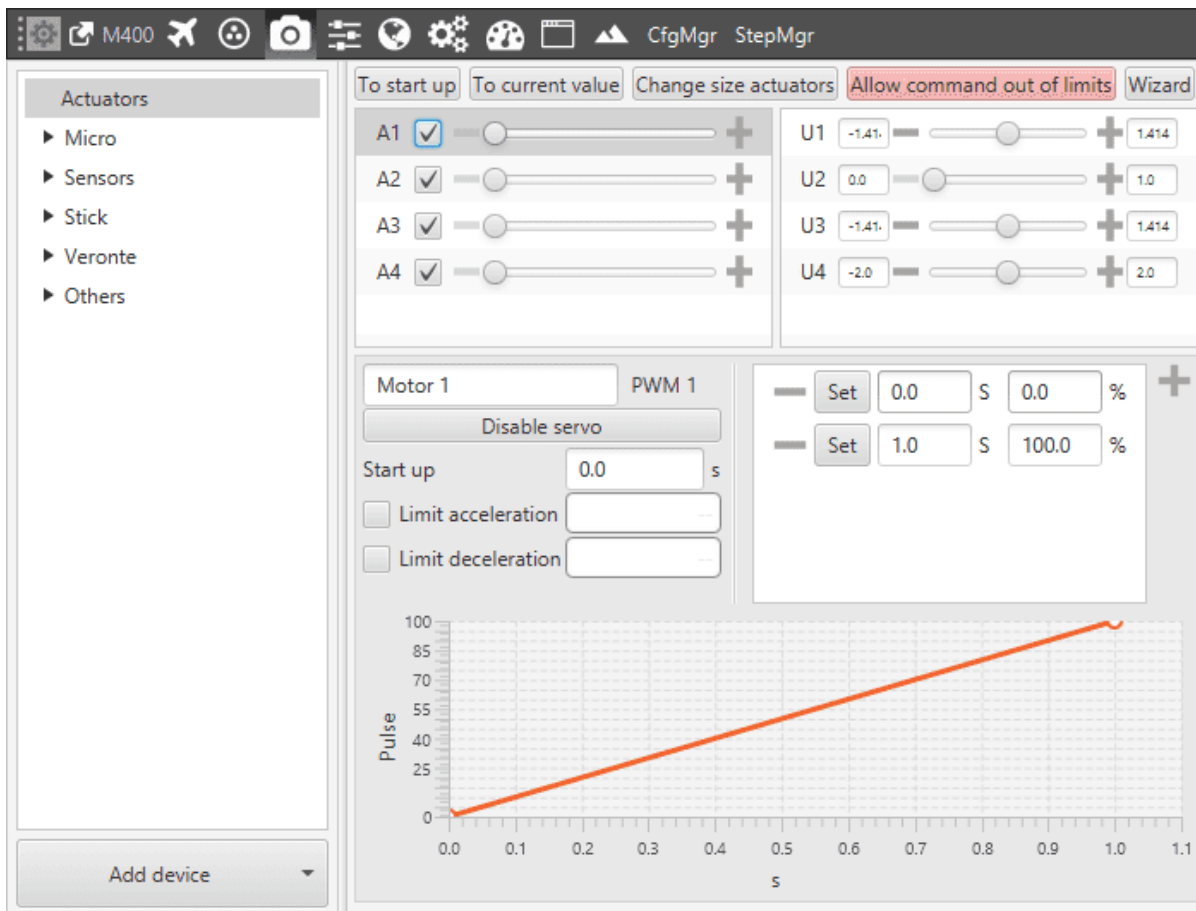
1. **Pitch Angle Control:** Control Output 1 is configured to perform a positive pitch angle change when motors 1-4 decrease their RPMs and motors 2-3 increase RPM value.

2. **Thrust Control:** the Control Output 2 is the one that allows a thrusting change (0-1).
3. **Roll Angle Control:** Control Output 3 is configured to perform a positive roll angle change when motors 1-2 decrease their RPMs and motors 3-4 increase RPM value.
4. **Yaw Angle Control:** Control Output 4 is configured to perform a positive yaw angle change when motors 2-4 decrease their RPMs and motors 1-3 increase RPM value, in this case with different proportions.

Warning: This panel shows the reference system of the aircraft too. It must be positioned in the same way of the Autopilot's one. If it results different, it can be edited by clicking on the corresponding axis in order to reverse its direction.

12.2.1.3.1.3 System Trim

As a final step, the system has to be trimmed. In this case, each motor will have a minimum and maximum value as shown in the picture.



M400 trim

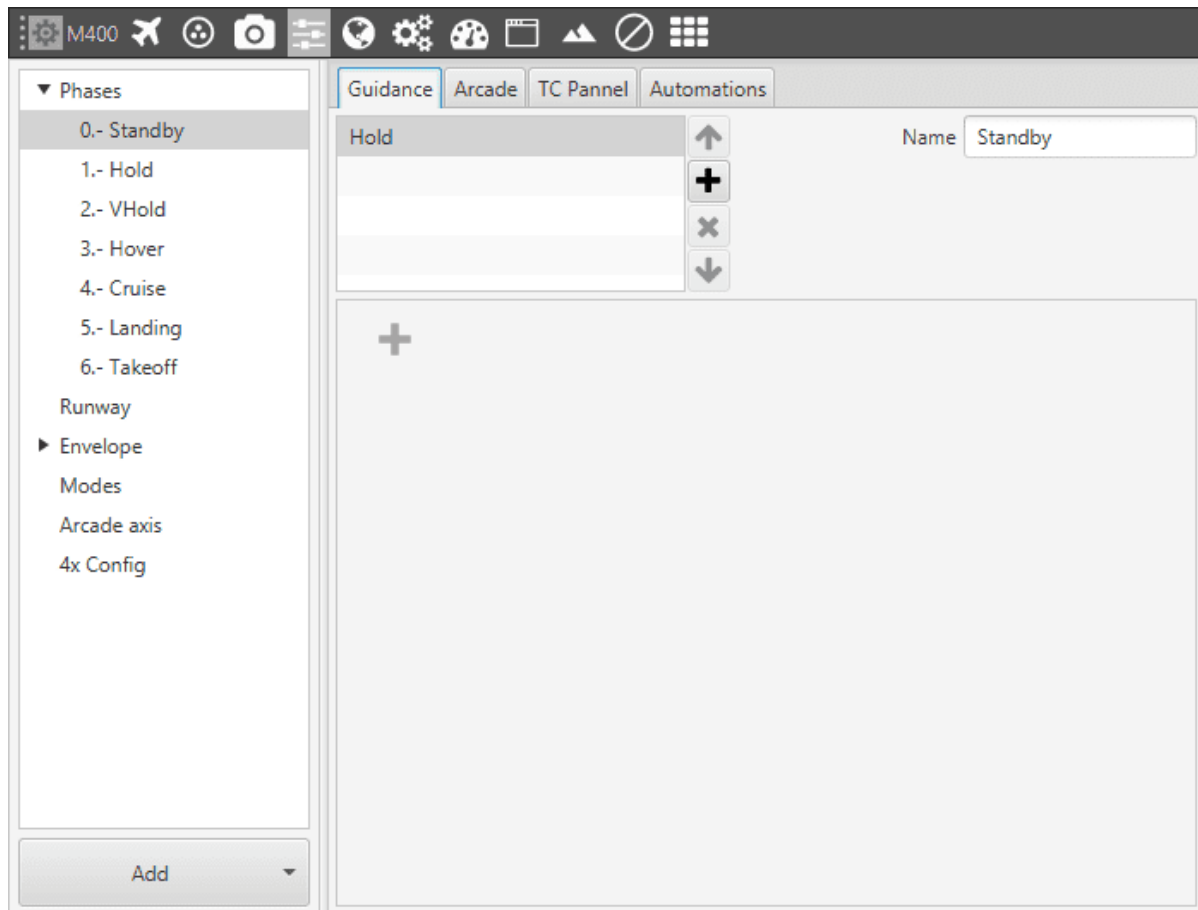
- **Minimum:** 0.
- **Maximum:** 1.

12.2.1.3.2 Mission Phases

In this section, it will be explained the M400 typical mission profile and the mission phases will be detailed.

12.2.1.3.2.1 Standby

Standby phase is a preliminary phase of the operation. The guidance is simply a **Hold** with no change. Normally, the automation which allows the system to pass to Standby phase requires the GPS signal.



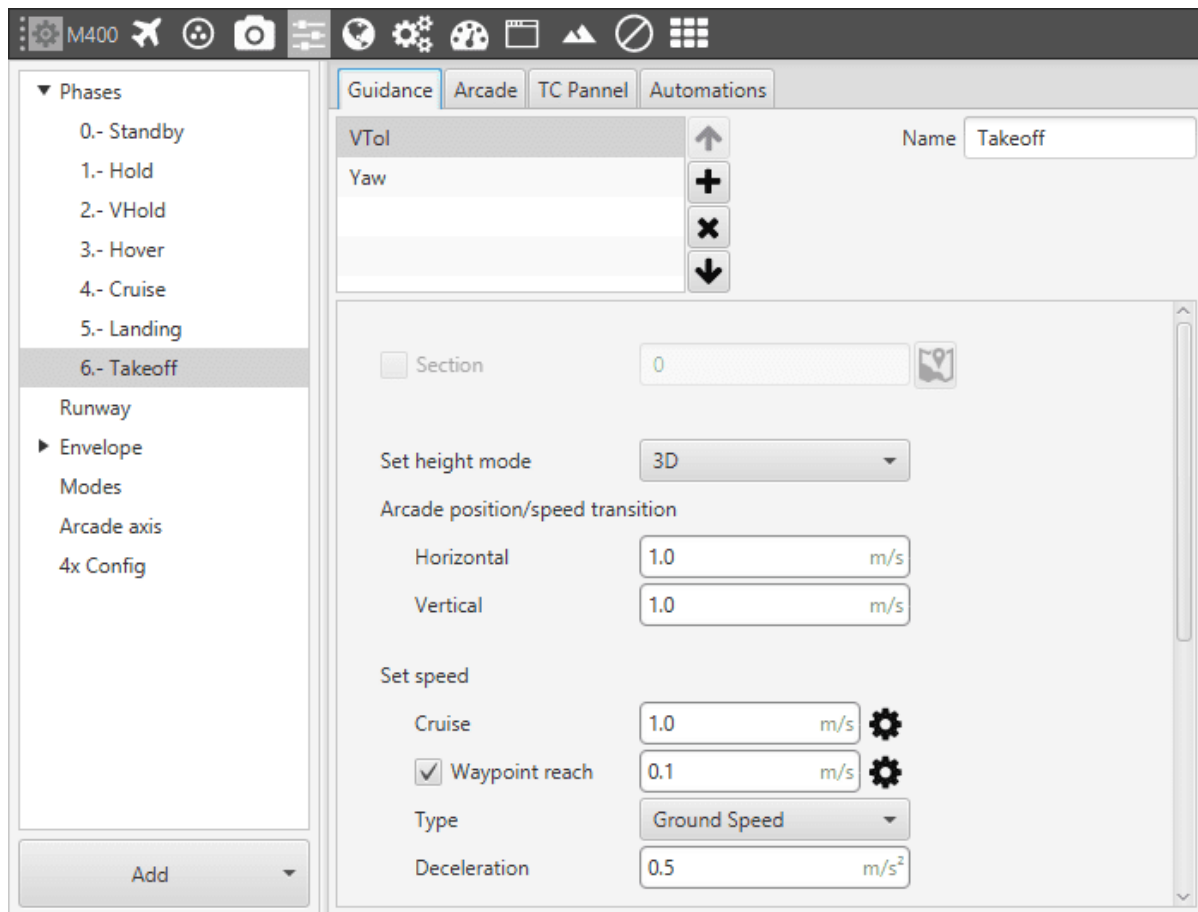
Standby phase panel

12.2.1.3.2.2 Takeoff

Take-off phase is composed of two Guidances: VTol and Yaw. The first one is a vertical guidance and where following parameters are set:

- **Arcade position/speed transition** (horizontal and vertical): 1.0 [m/s]
- **Set Speed:** 1.0 [m/s].
 - **Wypoint Reach:** 0.1 [m/s]
 - **Type:** Ground Speed
 - **Deceleration:** 0.5 [m/s]

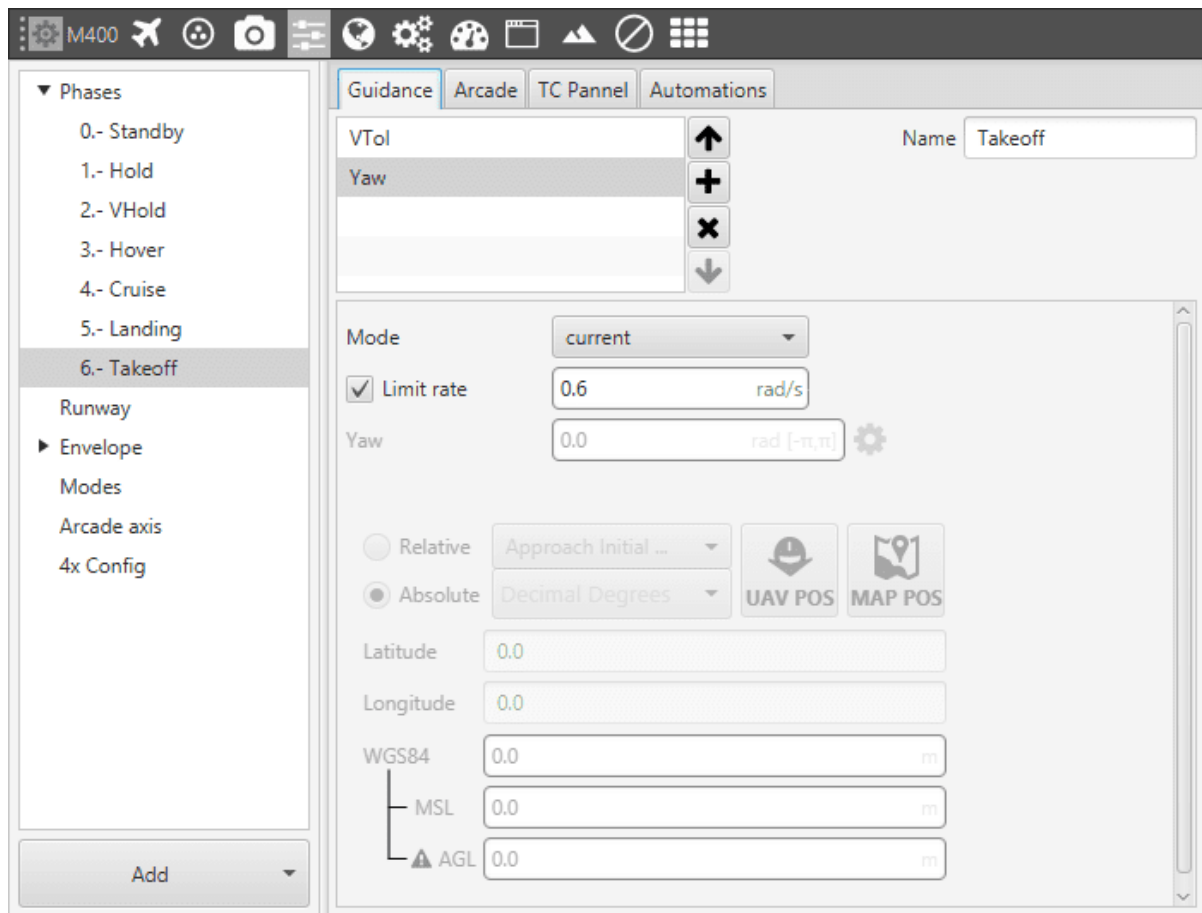
- **Type:** Straight
 - **Extend:** none
 - **Safe Distance:** -20.0 [m] and Relative (Positive down). With this value the platform will ascend.



VTol panel

Yaw guidance allows choosing the M400 orientation during takeoff:

- **Mode:** set to **Current**.
- **Limit rate:** limited rotation speed of 0.6 [rad/s].

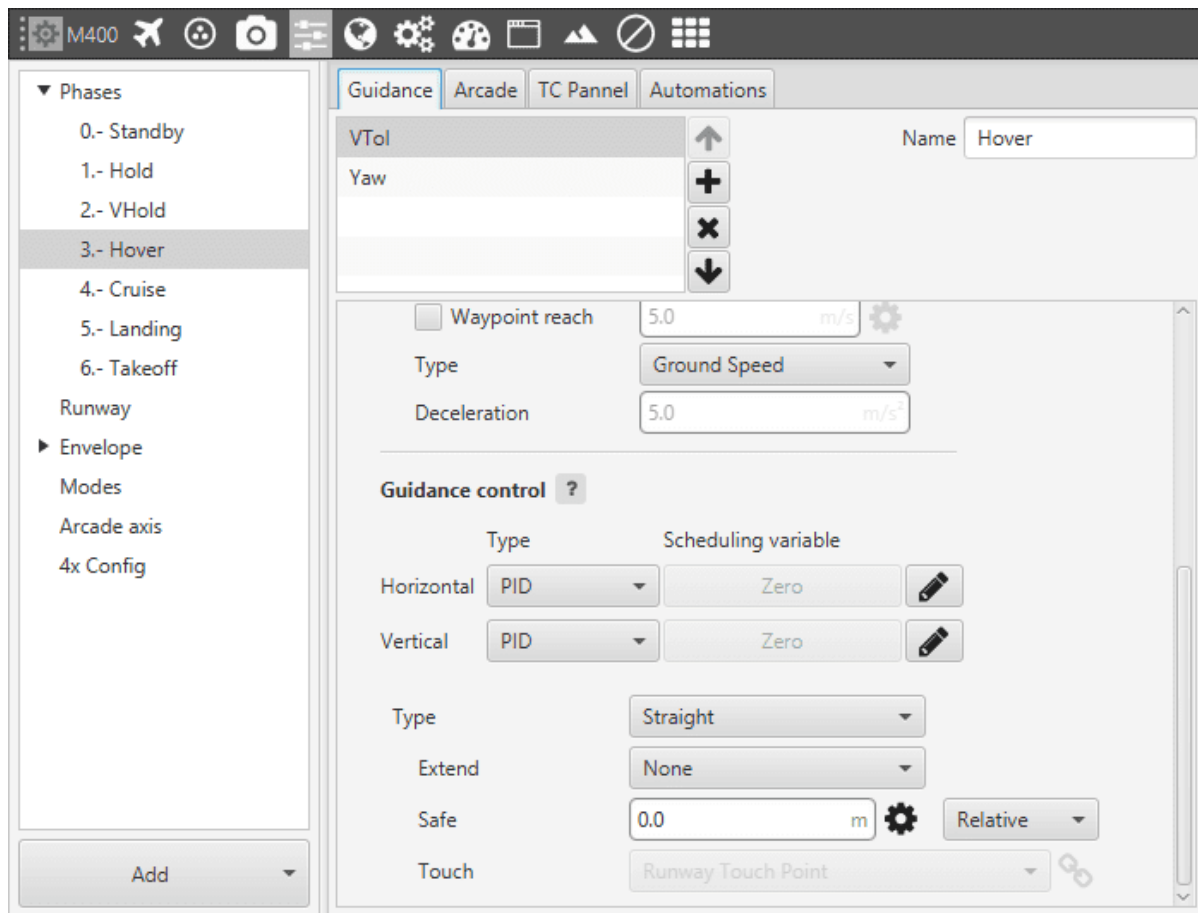


Yaw panel

12.2.1.3.2.3 Hover

Hover phase is configured to allow the multicopter maintain the position in the air. There are several ways to configure it: one of them is based on guidance specifications, while the other is related to automations.

The first way to configure the hover phase is by setting the guidances in order to maintain the position. The required guidances are **VTol** and **Yaw**.

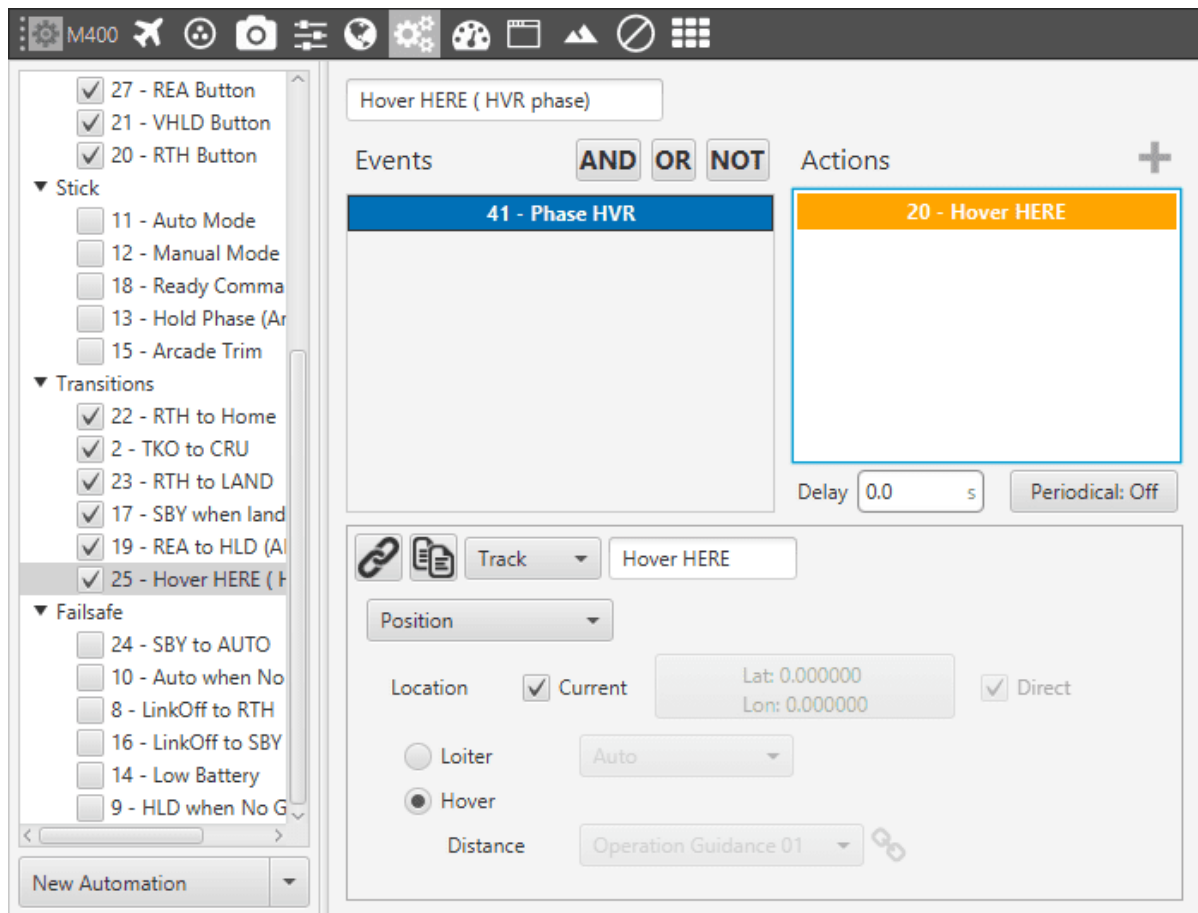


Hover phase panel

The **VTol** is configured as the one in Take Off phase, but in this case **Safe distance** is set at 0 [m] and Relative.

The **Yaw** guidance is the same of the Take Off phase.

The second option has the same effect, but is configured differently. There are also 2 guidances, but instead of **VTol** it is possible to use **Cruise**. The cruise guidance parameters are not relevant, as the automation will be the responsible of performing the hover. The **Yaw** guidance can be the same as the Takeoff one. Once the phase is configured, it is necessary to add an automation. This automation will have a phase event that will be activated when the UAV is in hover phase. The action will be a Track type action, so that the UAV tracks the current position performing a hover over it.



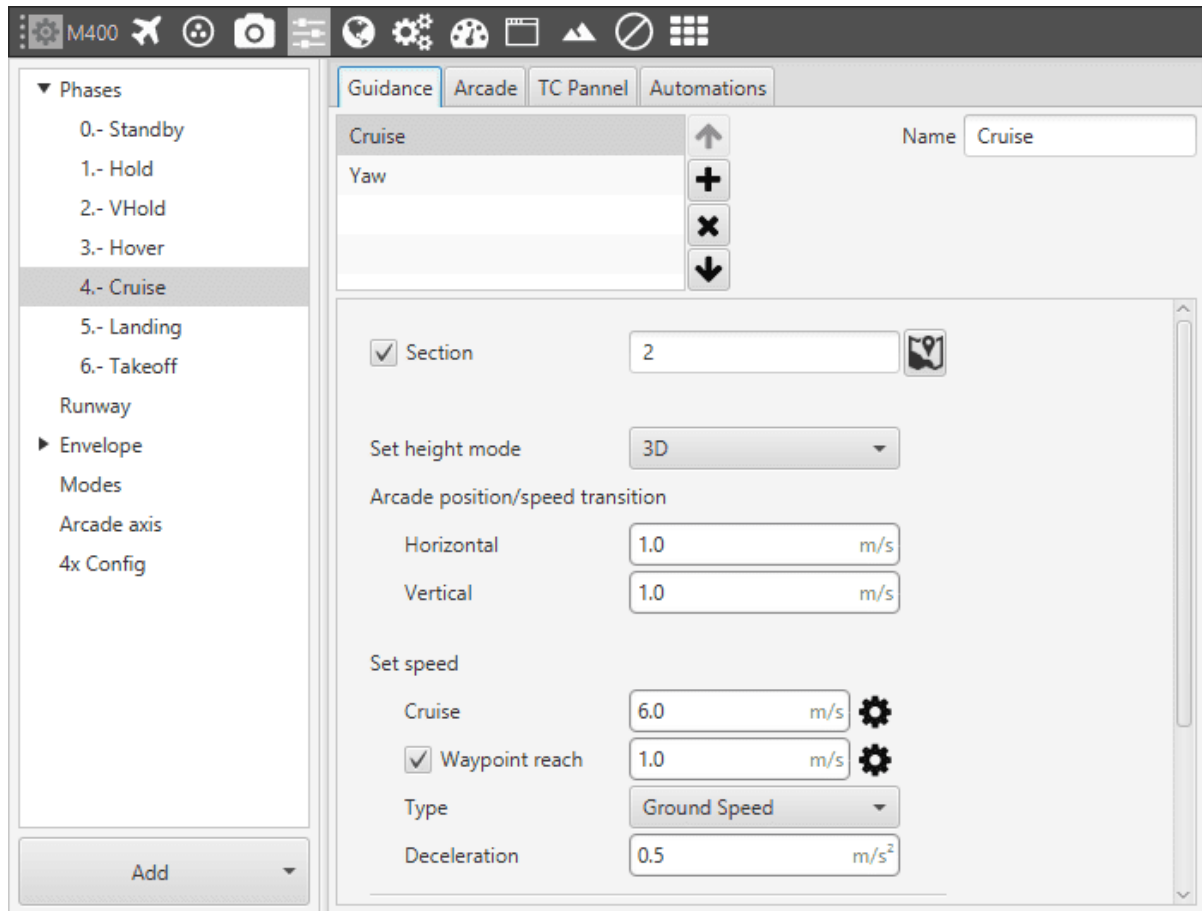
Hover automation

12.2.1.3.2.4 Cruise

In this phase, the Guidances are Cruise and Yaw.

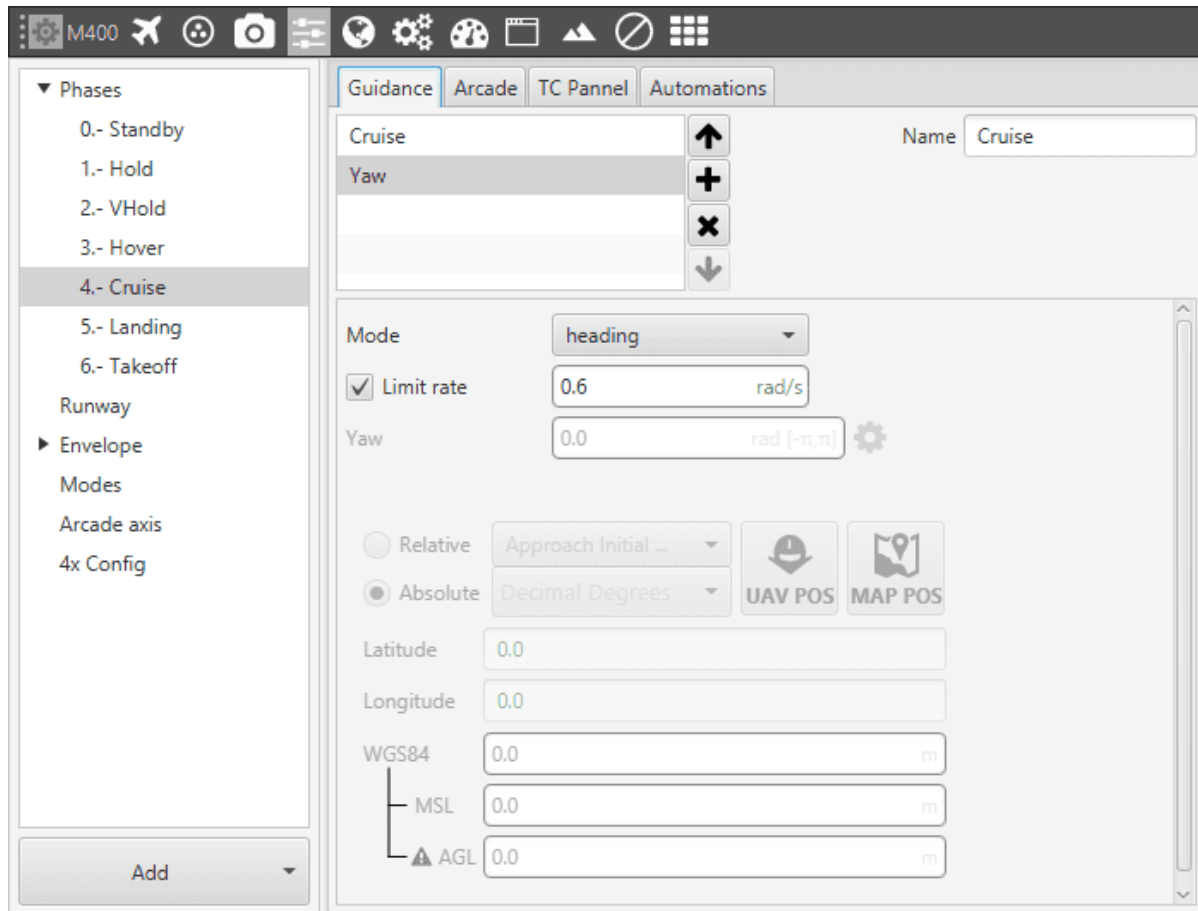
The main parameters to set are the Cruise Speeds (Cruise and Waypoint reach) and Acceleration limits.

- **Arcade position/speed transition** (horizontal and vertical): 1.0 [m/s]
- **Set Speed:** 6.0 [m/s].
 - **Waypoint Reach:** 1.0 [m/s]
 - **Type:** Ground Speed
 - **Deceleration:** 0.5 [m/s]



Cruise panel

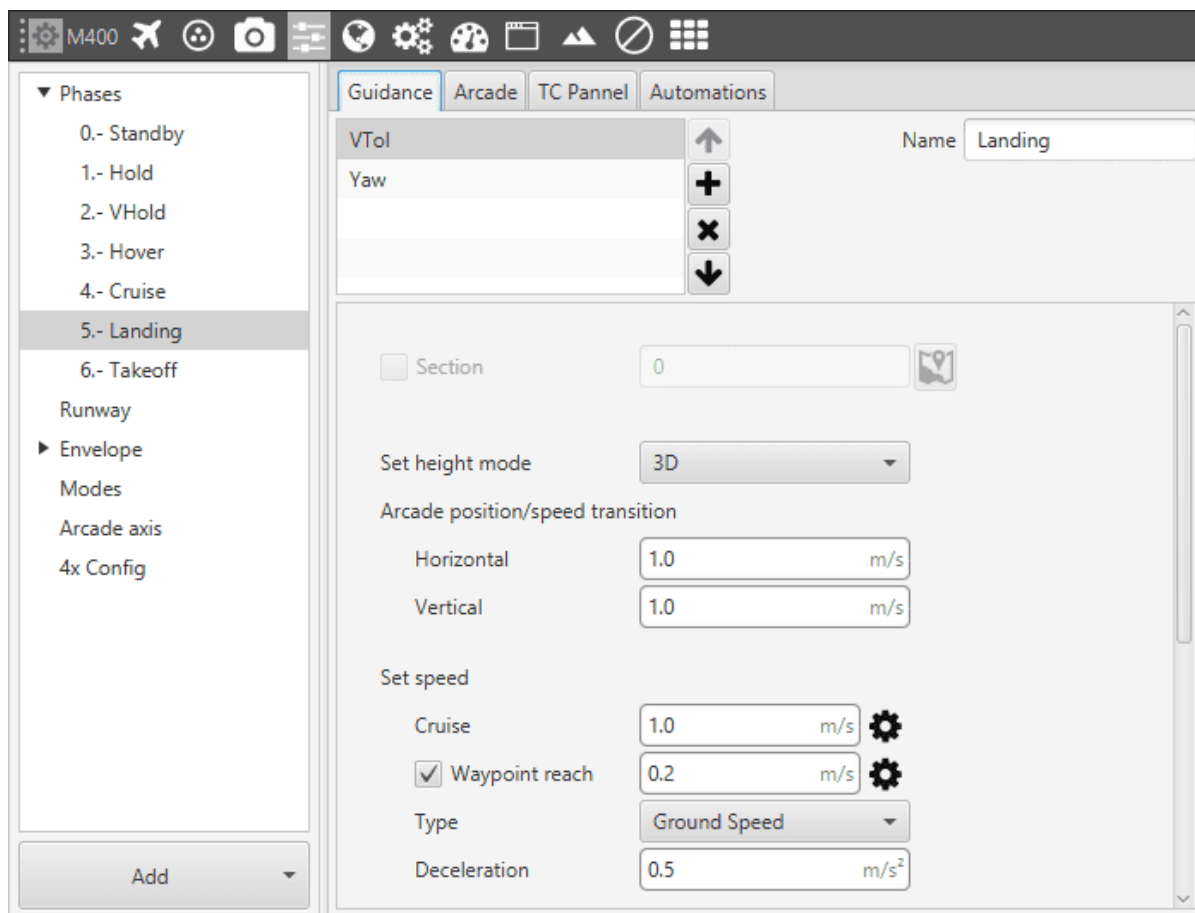
Yaw Guidance is set in order to make the multicopter controlling yaw angle following the heading angle.



Yaw Guidance panel

12.2.1.3.2.5 Landing

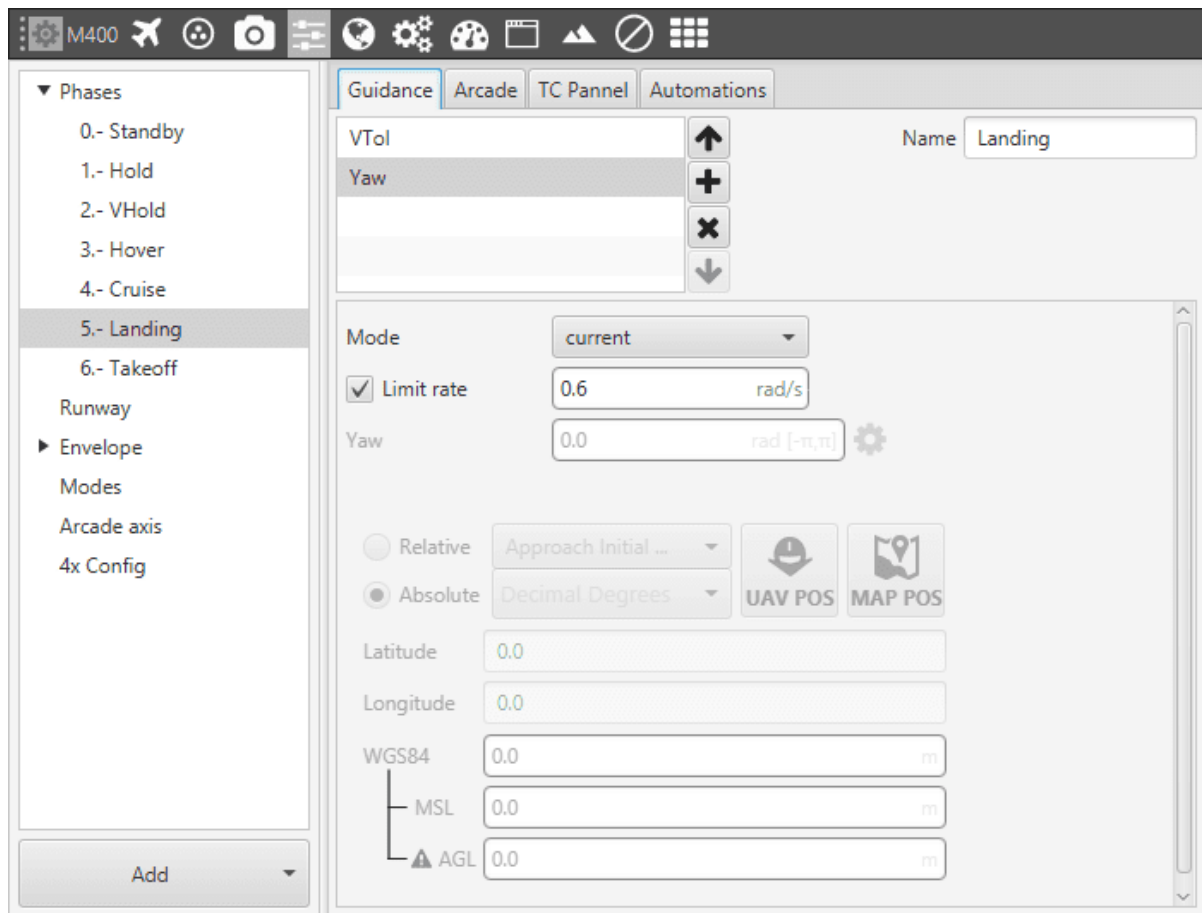
This phase has the same settings of the Take-off with some changes.



Landing phase panel

- **Arcade position/speed transition** (horizontal and vertical): 1.0 [m/s]
- **Set Speed:** 1.0 [m/s].
 - **Waypoint Reach:** 0.2 [m/s]
 - **Type:** Ground Speed
 - **Deceleration:** 0.5 [m/s]
- **Type:** Straight
 - **Extend:** none
 - **Safe Distance:** 100.0 [m] and Relative (Positive Down). In this case, we are supposing that at the instant when the Landing phase starts, the multicopter is flying at AGL<100m. It is possible select Extend – Down and the aircraft will descend till user desire.

Yaw Guidance is configured in order to let the quadcopter with the Current yaw angle and limiting the yaw rate at 0.6 [rad/s].



Yaw Guidance panel

12.2.1.3.2.6 VHOLD

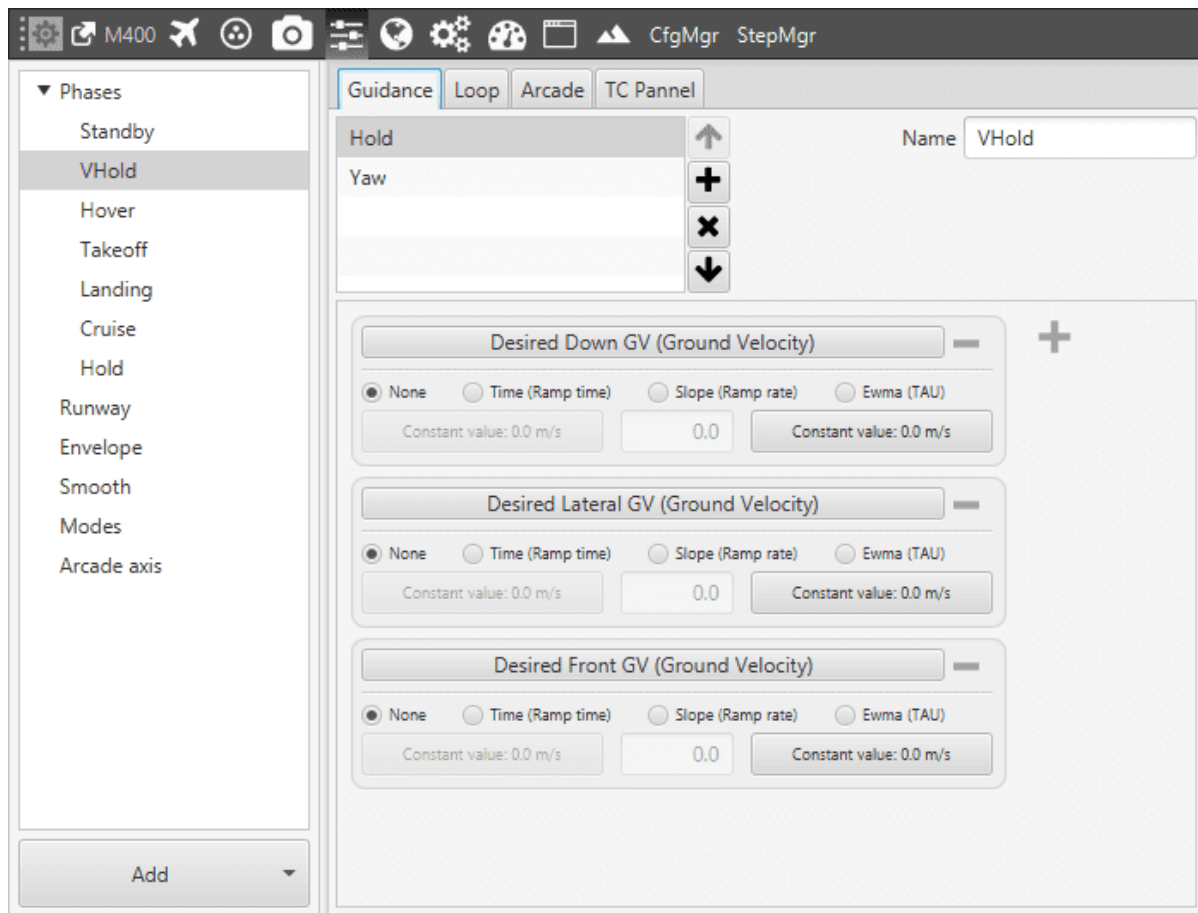
VHold phase is used to perform a Stick controlled flight. The phase guidance is composed of **Hold** and **Yaw** guidances.

Hold guidance is used to keep the following values at a specific value:

- **Desired Down GV:** at 0.0 [m/s].
- **Desired Lateral GV:** at 0.0 [m/s].
- **Desired Front GV:** at 0.0 [m/s].

In this way, if the stick control is not active, the multicopter will keep its position in the air without the need of phase changing.

Yaw Guidance is set on Current. In this phase, the Arcade allows controlling the platform depending on the set gains.

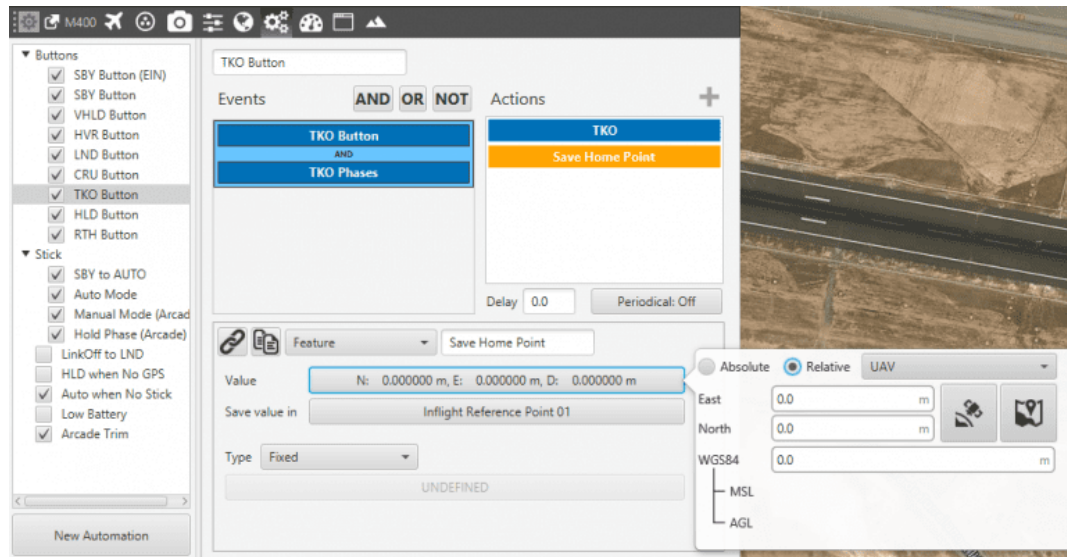


Manual phase panel

12.2.1.3.2.7 Return to Home

This operation is used to make a multicopter return to a point, considering it as Home. In order to perform this, is necessary to create two automations and a new phase, all process is explained below.

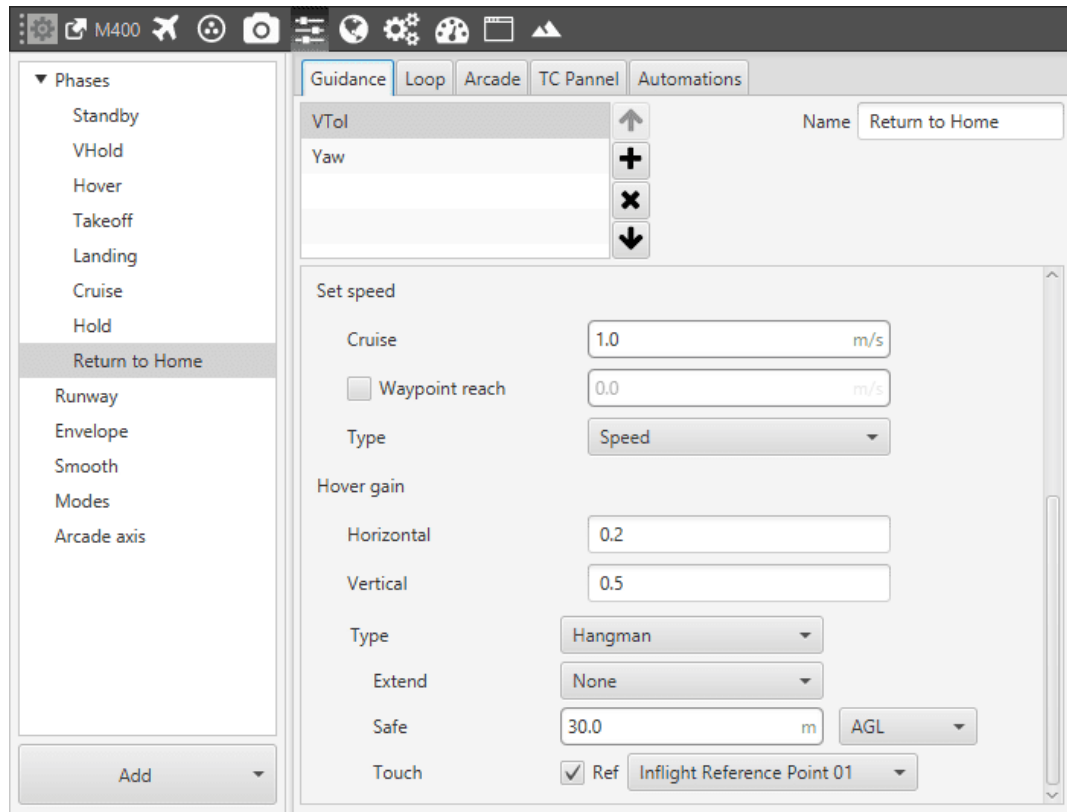
1. First create an Automation to save the current position, this has to be the one where the aircraft take-off (Home). This action can be added when creating the Take Off Button. See section [Veronte Panel Buttons](#) for more information about creating a button.



Automation – TKO Button

It is only necessary to add the action Feature and configure it as shown below:

- **Value:** select Relative and UAV. The parameters of position have to be set to 0, it means that the value to save is the UAV position.
 - **Save Value in:** select Inflight Reference Point.
 - **Type:** fixed.
2. The following step consists in creating a new phase. This phase can be named as **Return to Home** and it is necessary to include two guidances **VTol** and **Yaw**. VTol guidance has to be configured as follow:



Phase – Return to Home

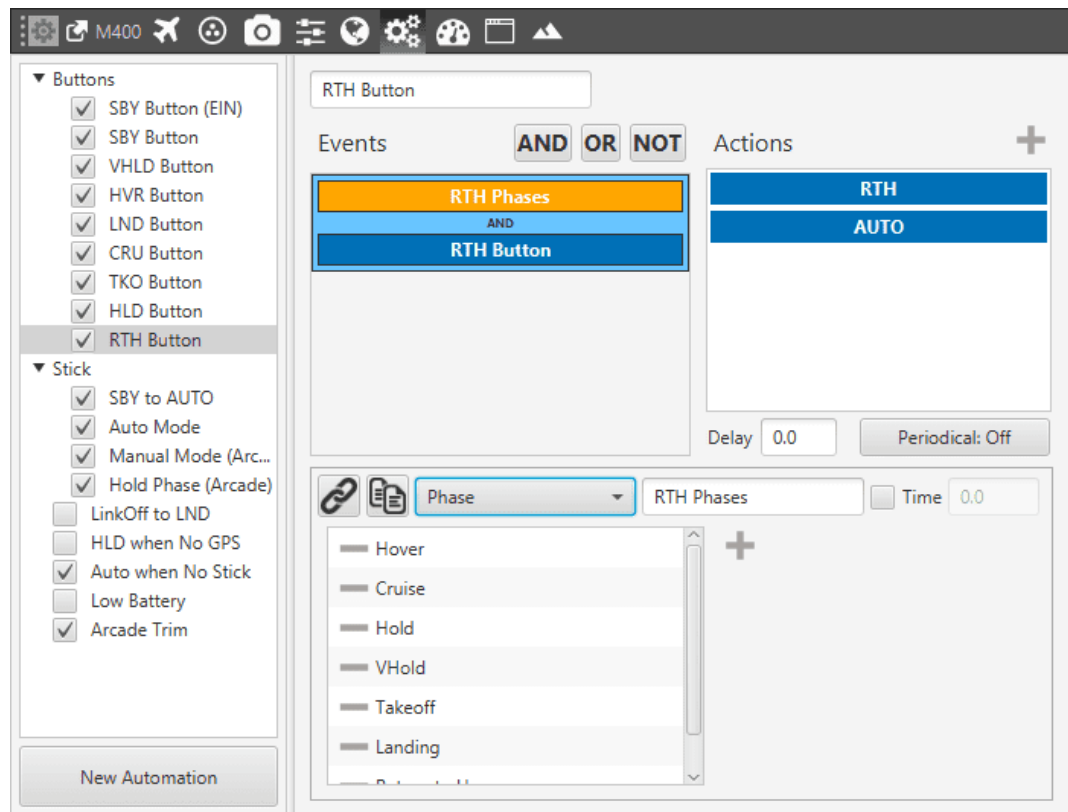
See section [VTOL Guidance](#) for more information about configuring this guidance. It is worth noting the following parameters:

- **Type:** Hangman is selected.
- **Extend:** None
- **Safe:** 30 [m] and AGL, this means that the aircraft will perform the Hangman path reaching an altitude of 30 [m] AGL.
- **Touch:** here is defined the point where the platform will touch ground. This has been defined in the first step (Inflight Reference Point) and now it is possible to select it.

Yaw guidance can be set to **current** and user may want to limit the rate.

3. The last step consists in the creation of a Button on the Veronte panel. Go to Automation Panel and create the following Events and Actions.

Two events have to be created: **Phase** and **Button**.



Automation

The actions configured are:

- **Phase:** here is selected the phase that the platform will enter in i.e, Return To Home.
- **Mode:** AUTO is selected.

Finally, user only have to click on the Button which appear on Veronte Panel to perform a Return Home.



M400

This M400 multirotor has a quadcopter structure. Its control is performed by using a differential thrust for each motor depending on the desired attitude change:

- ALTITUDE
- PITCH ANGLE
- ROLL ANGLE
- YAW ANGLE

12.2.1.4 Helicopter T-Rex600

12.2.1.4.1 Servo Configuration

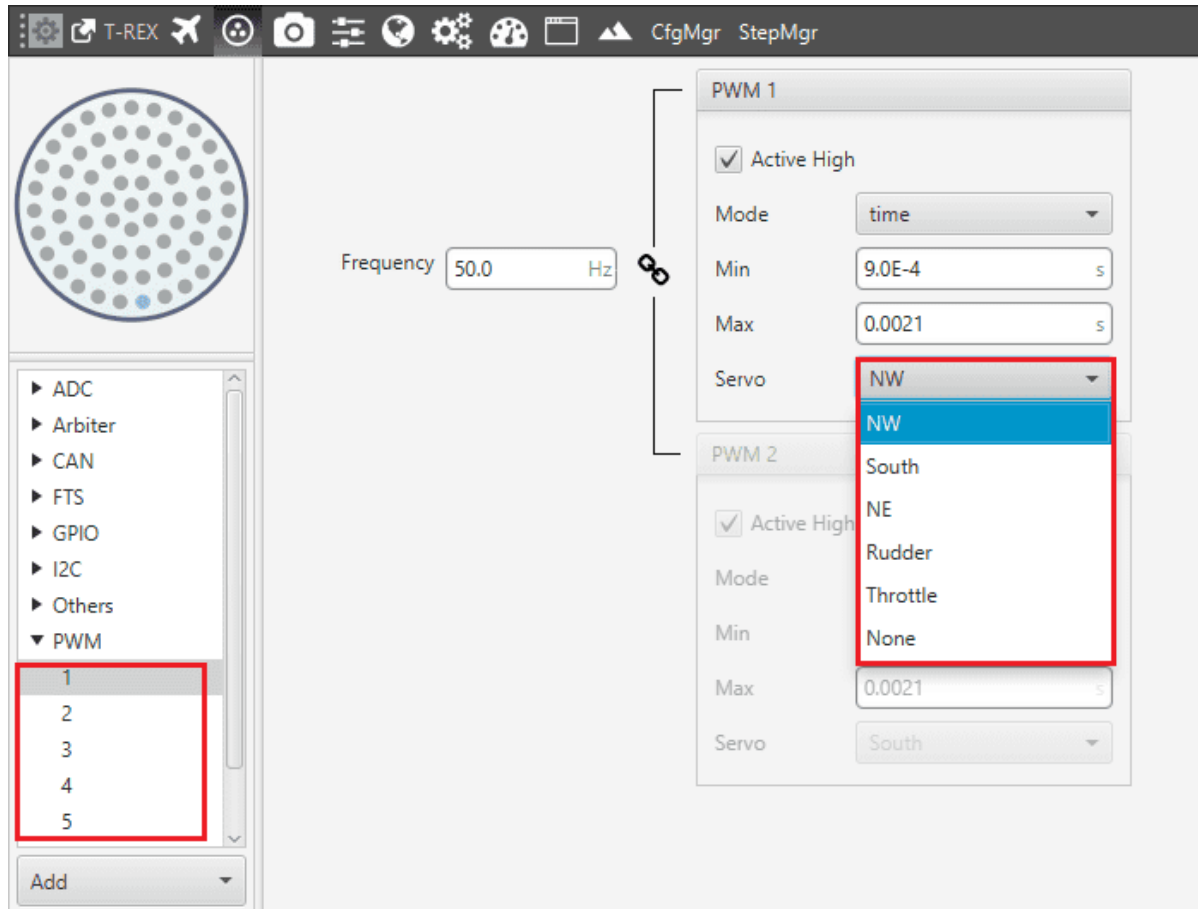
The T-Rex integration process can be performed by using Veronte Pipe connected with the hardware system and the Autopilot as explained in section *Aircraft Mounting* of the manual.

12.2.1.4.1.1 Servos Output

The first step of the process is the servos configuration. In this section will be explained how to set the servo-output matrix.

In this case, the controls of the helicopter are the cyclic, collective, RPM (main rotor) and blades pitch angle changing (tail rotor). To control the main rotor moving it is necessary to configure correctly the three servos that allow controlling the blades pitch angle and the one which changes the main rotor RPMs. In order to have the tail rotor control and to be able to change the blades pitch, another servo must be configured.

All the servos correspond to a pin of the connector and they must be positioned in the **S** vector who represents the **Actuator Outputs**. It is possible to connect any pin to any command but the easiest way to perform it and avoid confusion is by following the pins number.



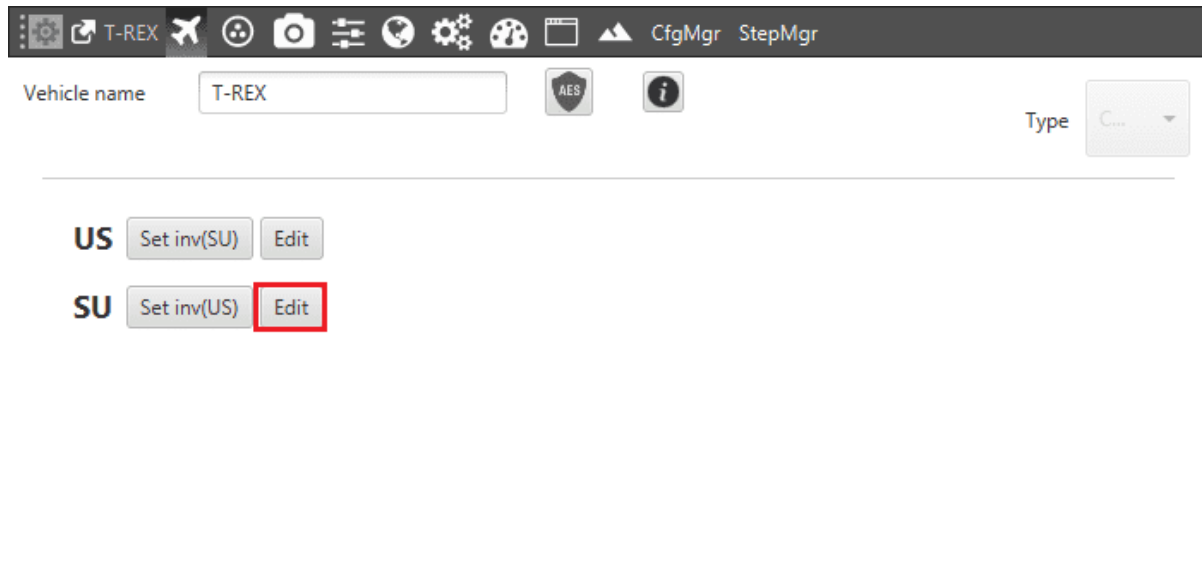
T-Rex Outputs

In this case, it is necessary to use 5 pins:

- **Output 1** – NW main rotor cyclic/collective
- **Output 2** – South main rotor cyclic/collective
- **Output 3** – NE main rotor cyclic/collective
- **Output 4** – Main rotor RPMs
- **Output 5** – Tail rotor blades angle changing

12.2.1.4.1.2 SU Matrix

At this point, the **S** vector is defined and the **SU** matrix can be edited. By clicking on **Edit** it is possible to configure the relation between the controller outputs (**U** vector) and the servo movements (**S** vector).



SU matrix edit

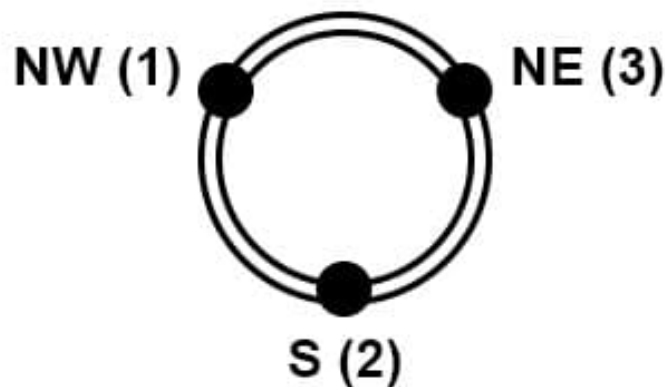
Warning: This panel shows the reference system of the aircraft too. It must be positioned in the same way of the Autopilot's one. If it results different, it can be edited by clicking on the corresponding axis in order to reverse its direction.

| | | Control Output u1 - Pitching - Cyclic | Control Output u2 - RPMs | Control Output u3 - Rolling - Cyclic | Control Output u4 - Yawing | Control Output u5 - Collective |
|---|----------|--|-----------------------------|---|-------------------------------|-----------------------------------|
| - | NW | 0.16666667 | 0.0 | -0.28867513 | 0.0 | -0.33333334 |
| - | South | -0.33333334 | 0.0 | 0.0 | 0.0 | -0.33333334 |
| - | NE | 0.16666667 | 0.0 | 0.28867513 | 0.0 | -0.33333334 |
| - | Rudder | 0.0 | 0.0 | 0.0 | 1.0 | 0.0 |
| - | Throttle | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 |

+

Apply

SU matrix



T-Rex swashplate actuators

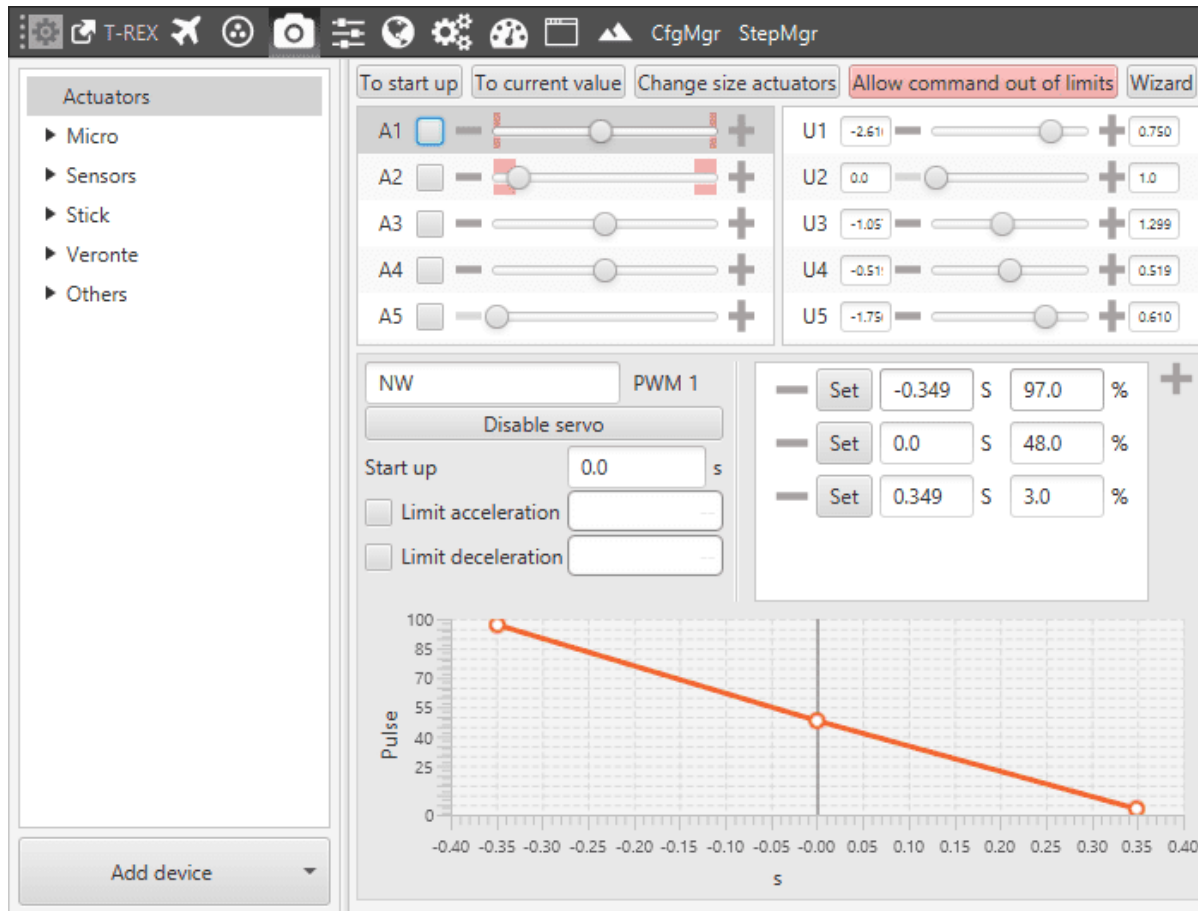
The previous image shows the swashplate of the T-Rex with the 3 actuators able to modify its rotation plane (X axis is positive up). As showed in the SU matrix image, controls are performed as follow:

1. **Pitch Angle Control:** Control Output 1 is configured to perform a positive pitch angle change when servos 1-3 move following Z axis positive and servo 2 moves in the opposite direction with double magnitude.
2. **RPM Control (Main rotor):** the Control Output 2 (servo number 5) is the one that allows an RPM change with the throttle moving (0-1).
3. **Roll Angle Control:** Control Output 3 is configured to perform a positive roll angle change when servos 1-3 move with an opposite direction (servo 2 is in the longitudinal plane of the aircraft, so it has not any influence).
4. **Yaw Angle Control:** Control Output 4 is configured to perform a positive yaw angle change when tail rotor decreases the blades pitch angle.

5. **Collective Control:** The altitude control is performed using the Control Output 5. A positive change of the pitch angle blade is performed when servos 1-2-3 move contemporaneously in the same direction.

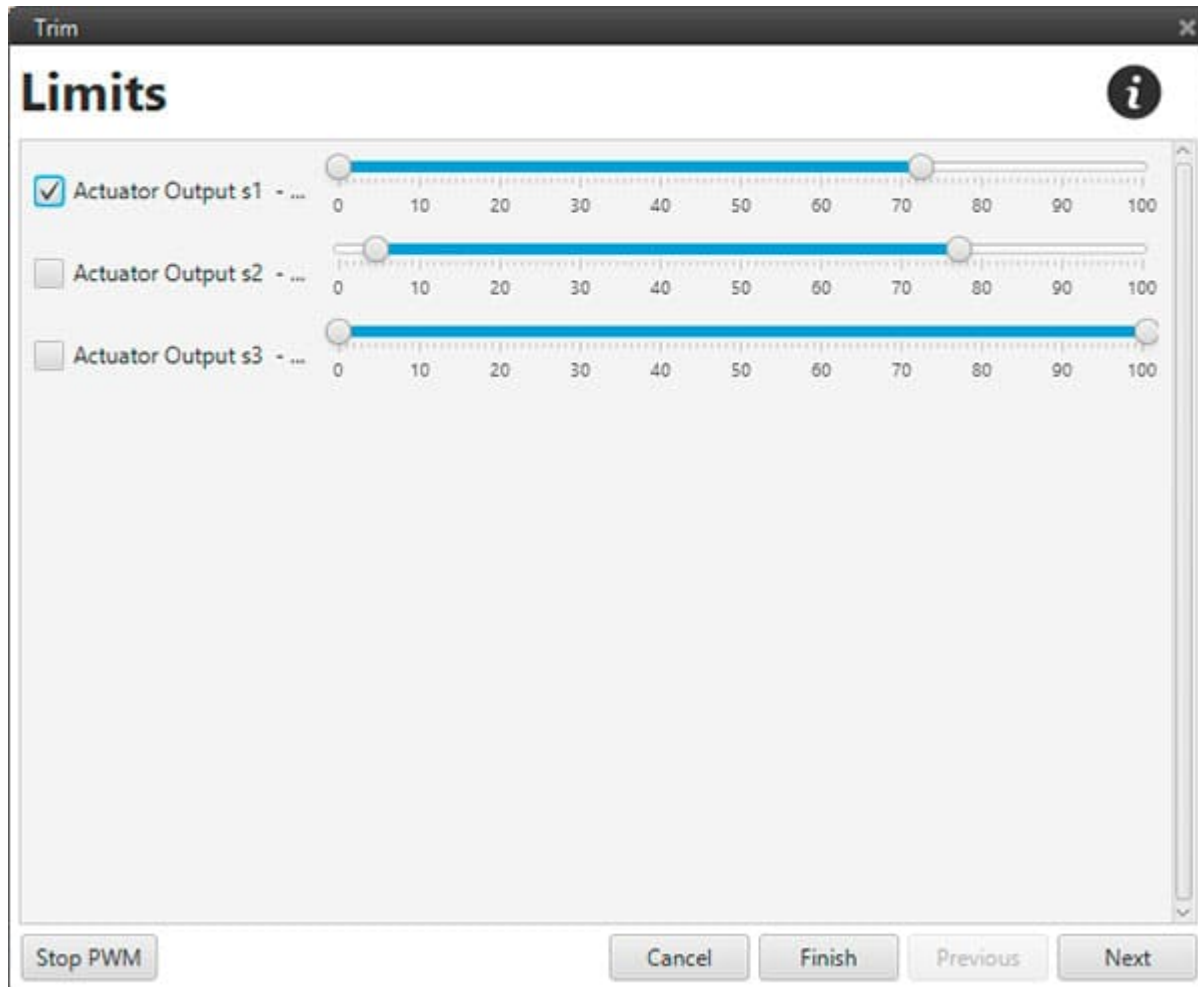
12.2.1.4.1.3 System Trim

As a final step, the system has to be trimmed. This can be performed by moving servos in three different positions: zero position, minimum and maximum (blade angles are usually limited physically). These positions must be inserted and saved in the software by clicking on Set when the actuator is in the desired position or introducing them manually.



Actuator 1 trimming

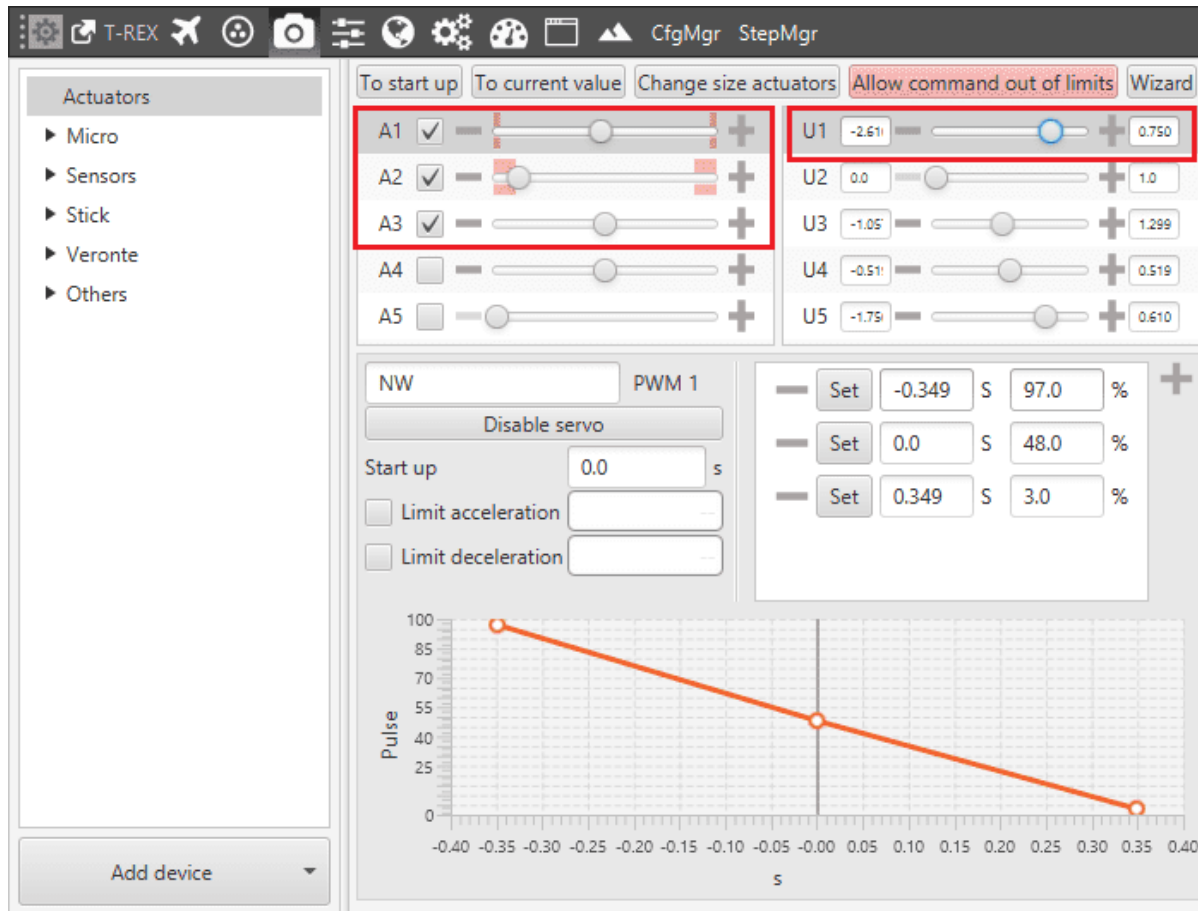
This procedure can be performed in the same way by using a **Wizard**. This tool allows moving actuators limits easily and finding the correct range.



Trim wizard tool

In order to perform a final checking, it is possible to select the desired channel and testing pitching, rolling, yawing and thrusting controls.

The image below shows a pitching output testing. By moving the U1 control (Pitching – Cyclic), main rotor servos change the position according to the reference system: positive corresponds to nose down and negative to nose up.



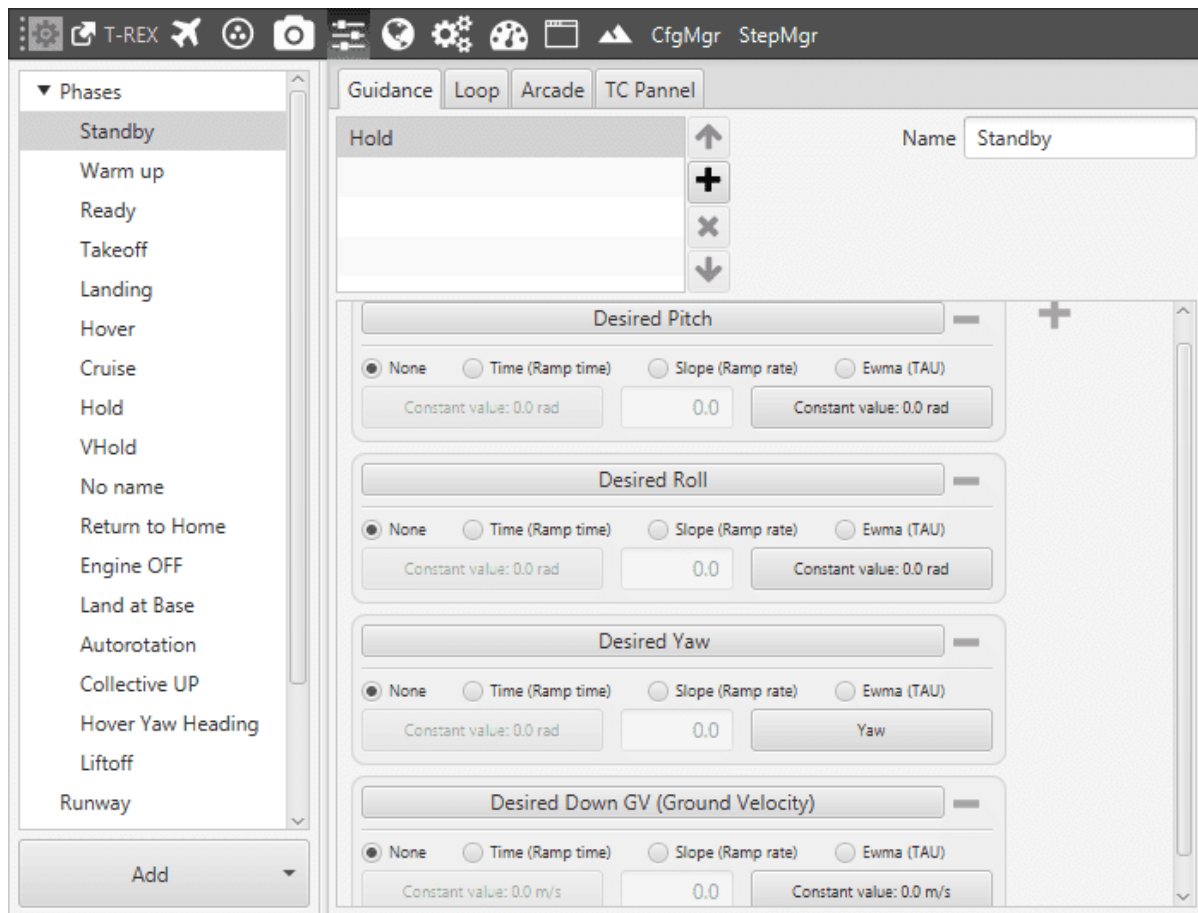
Pitching test

12.2.1.4.2 Mission Phases

In this section, it will be explained the T-Rex typical mission profile and the mission phases will be detailed.

12.2.1.4.2.1 Standby

Standby phase is a preliminary phase of the operation. During this phase is possible to check, for example, if the aircraft is correctly controlling the attitude by moving it and watching the servos positions changing.



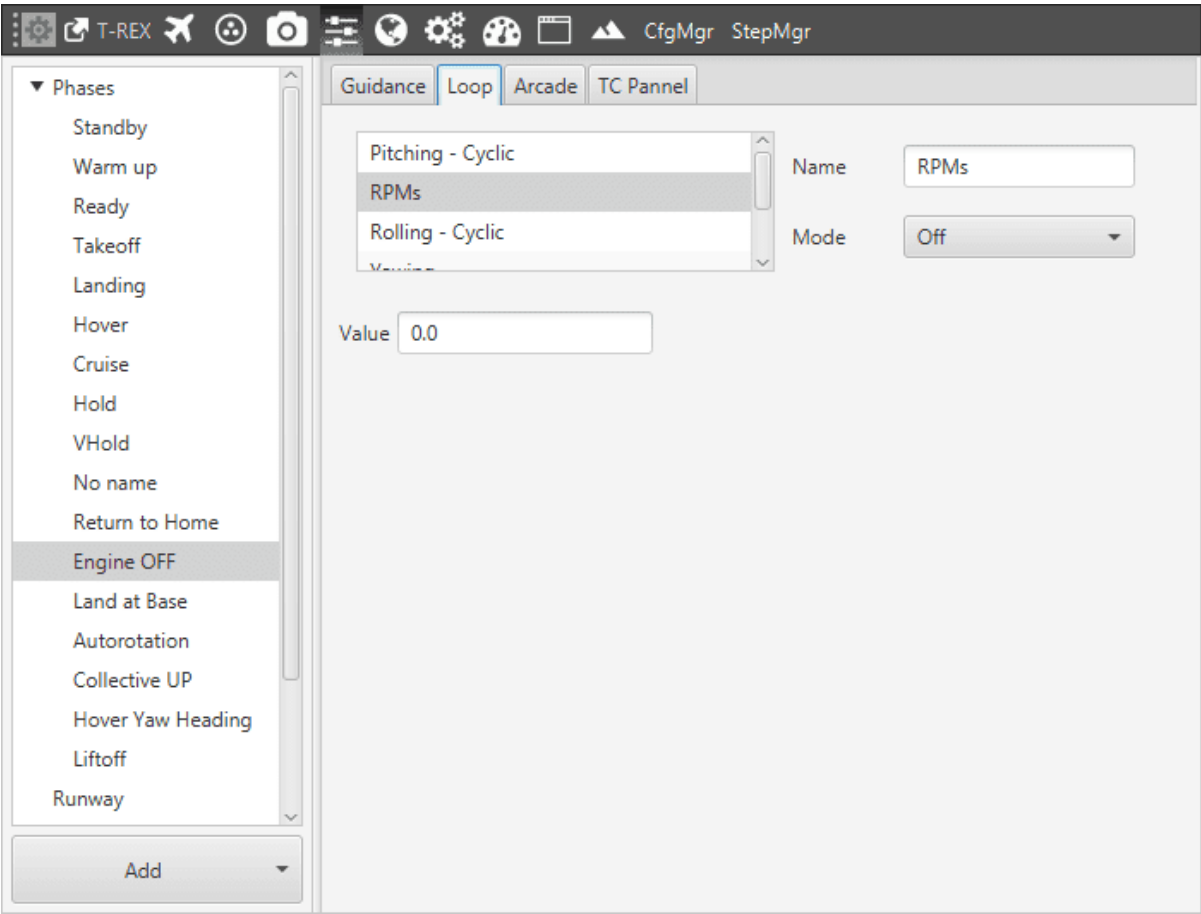
Standby phase panel

The guidance is a **Hold** of four variables:

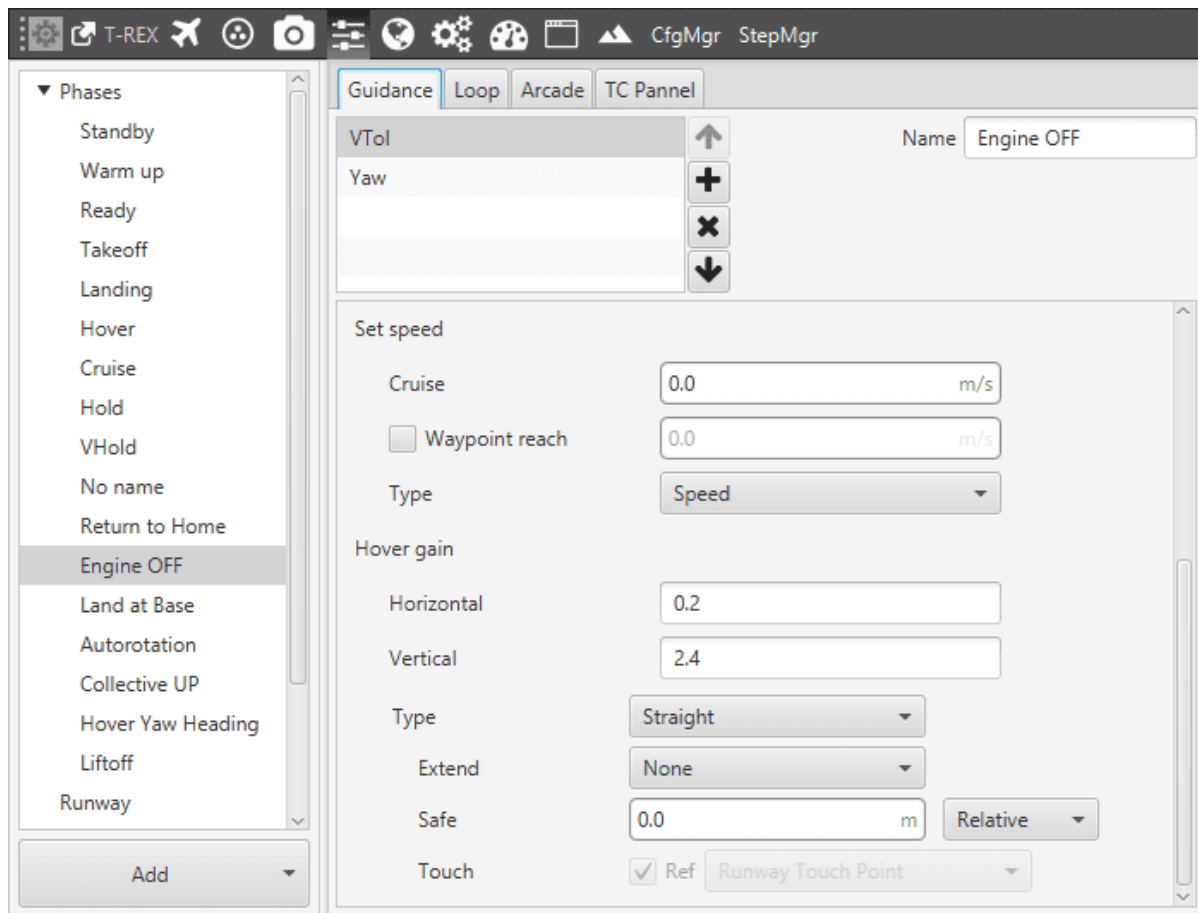
- **Pitch and Roll angles:** at 0 [rad].
- **Ground Speed:** at 0 [m/s].
- **Yaw angle:** set at the current.

12.2.1.4.2.2 Engine Off

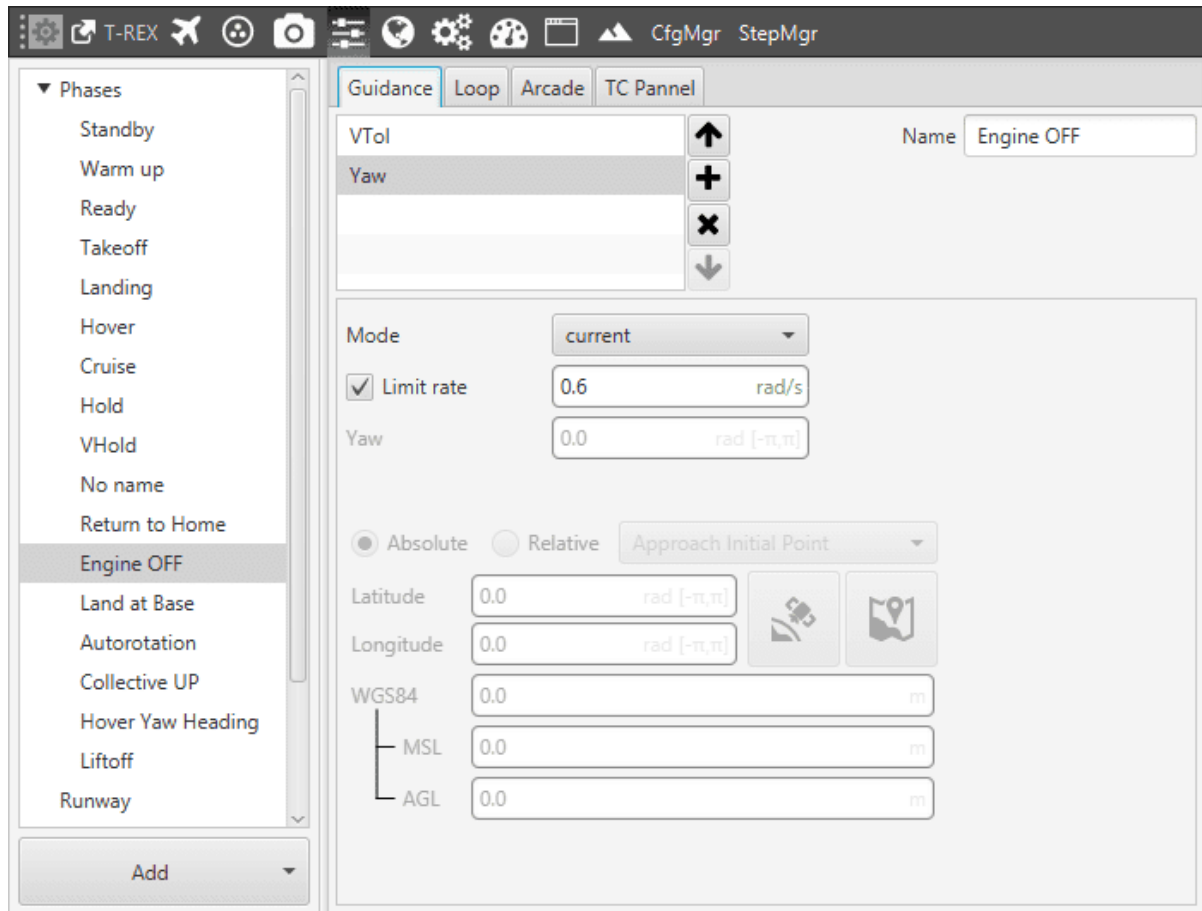
Engine OFF is the phase which allows the platform decreasing main rotor RPMs without attitude changes. In fact, the decreasing phase is not instantaneous and the autopilot needs to keep the control until the RPMs would not be able to make attitude changes. Following this fact, this phase has a **thrusting control** set at zero and VTol and **Yaw** Guidances.



Thrusting fixed control



VTol Guidance settings



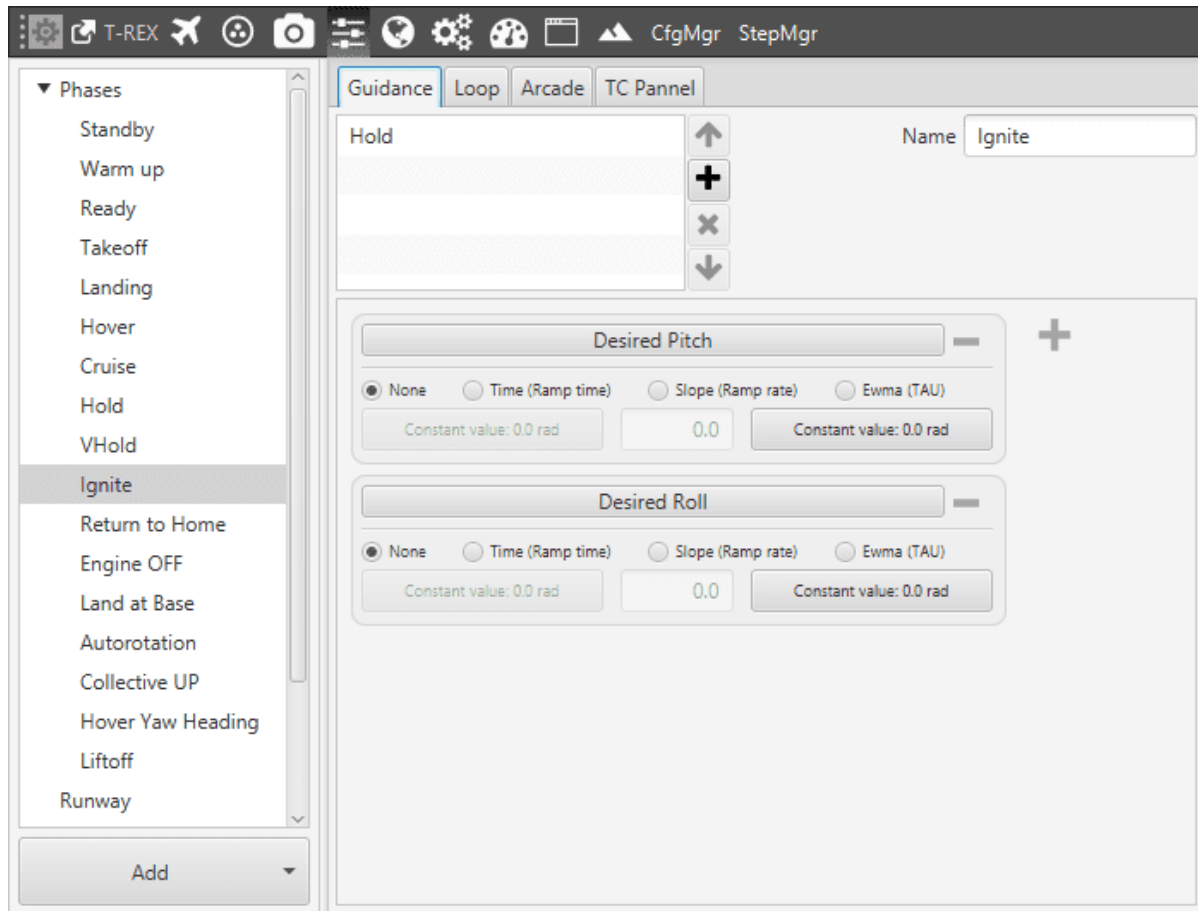
Yaw Guidance settings

12.2.1.4.2.3 Ignite

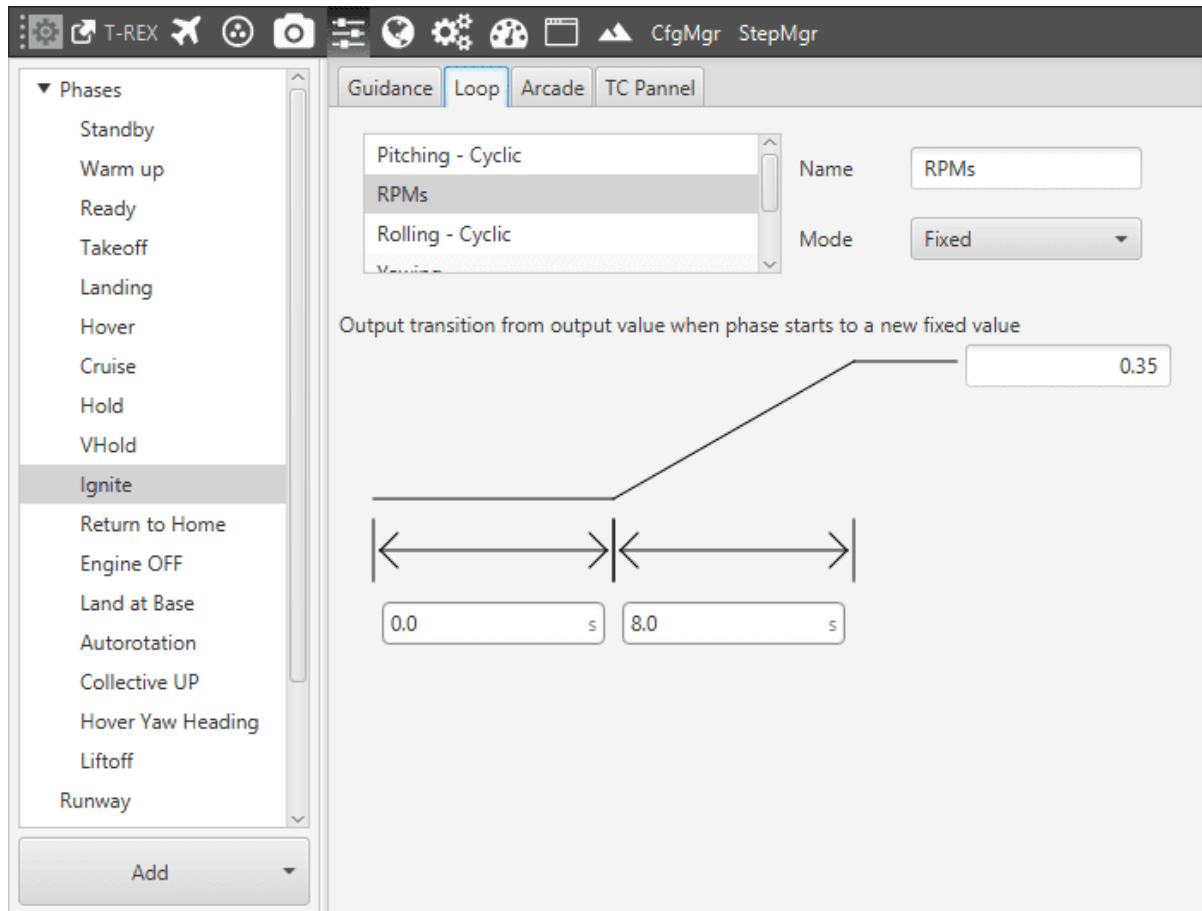
The Guidance of this phase is a **Hold**, which allows to maintain:

- **Desired Roll and Pitch angles:** at 0 [rad].

During this phase RPM of the engine will increase until 35 % of thrust.



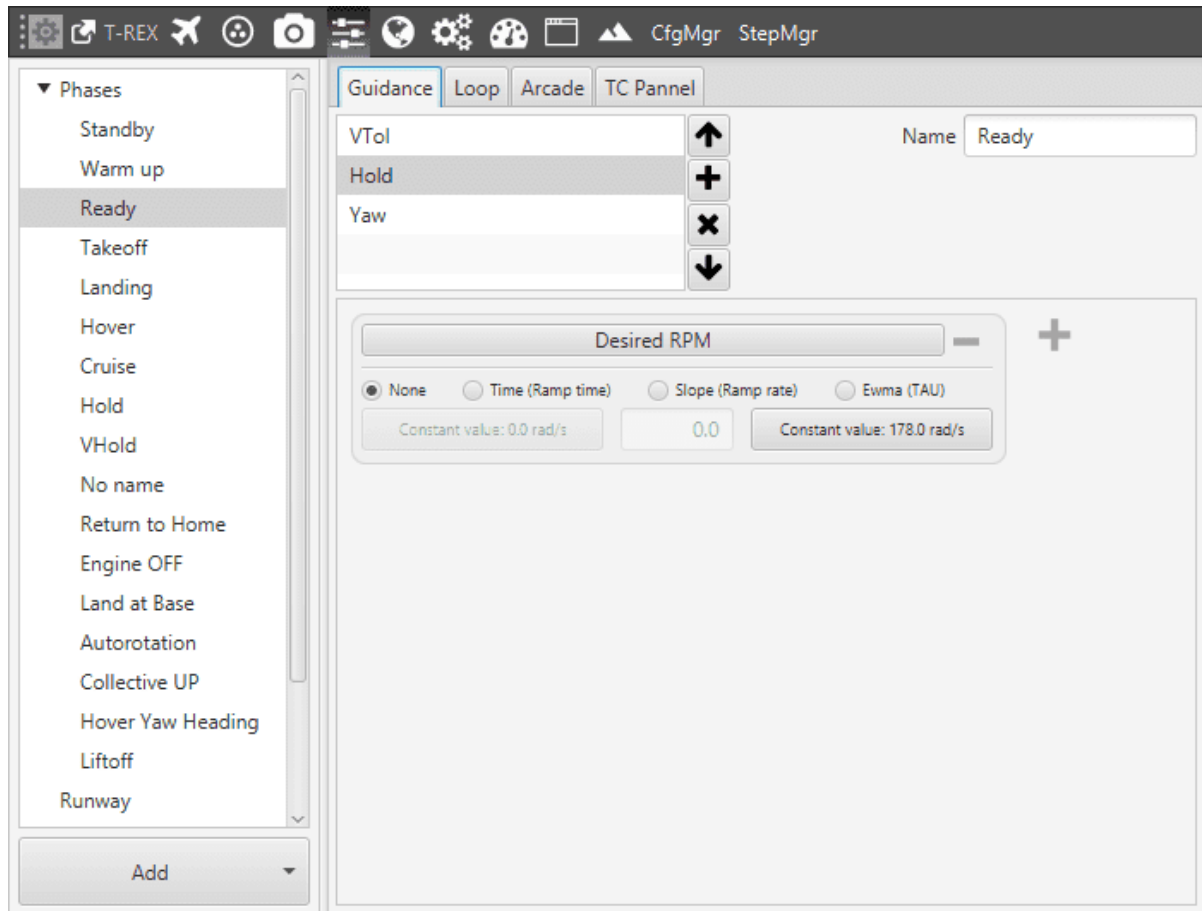
Hold Guidance configuration



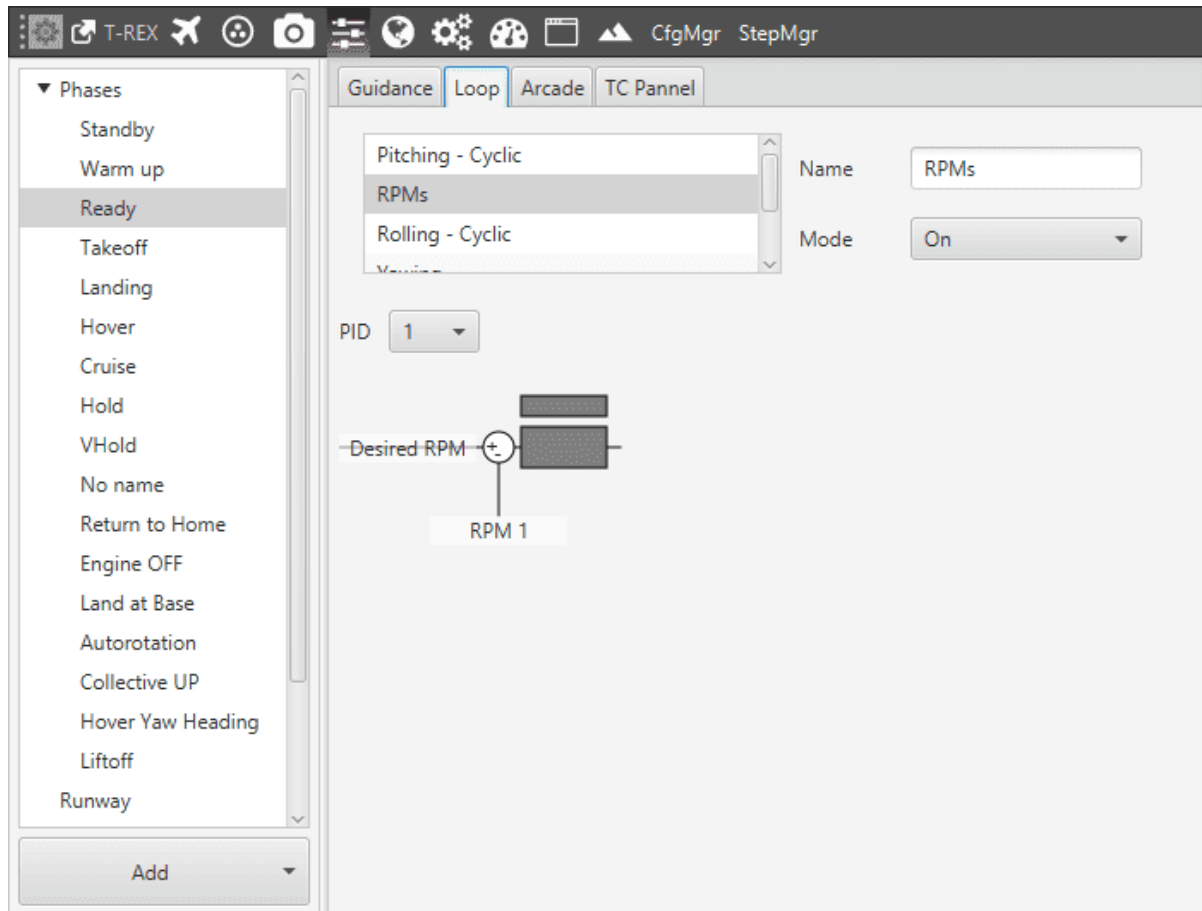
Engine RPM increasing settings

12.2.1.4.2.4 Ready

This phase is the one that starts to control RPM value with PID controller to maintain it at the desired value (Desired RPM). The Guidance are a **Hold** of RPMs (this guidance will be present in all flight phases), and the same **VTol** and **Yaw** of the Standby phase.



Desired RPM value

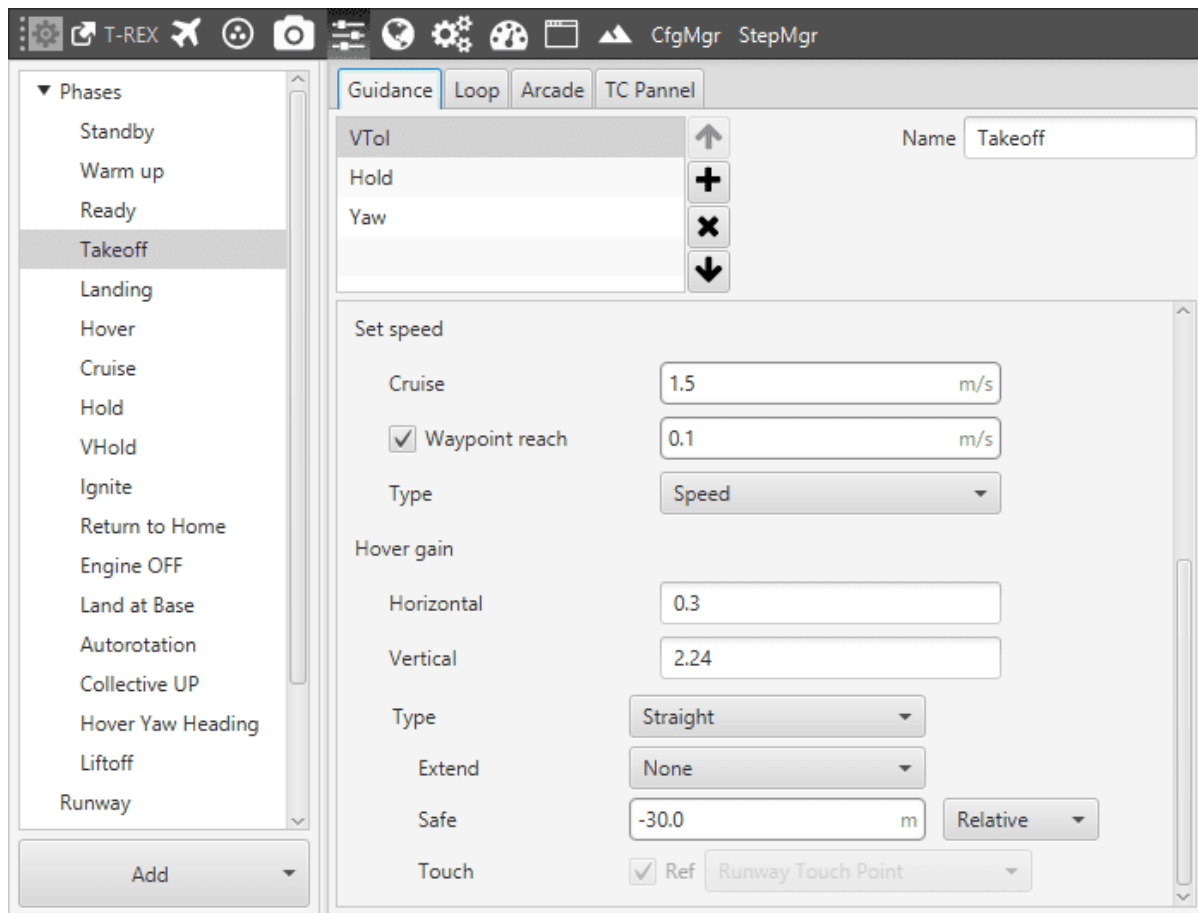


RPM control loop

12.2.1.4.2.5 Takeoff

Take-off phase is composed by 3 Guidances: **VTol**, **Hold** (RPM) and **Yaw** (Current). The first one is a vertical guidance and following parameters are set:

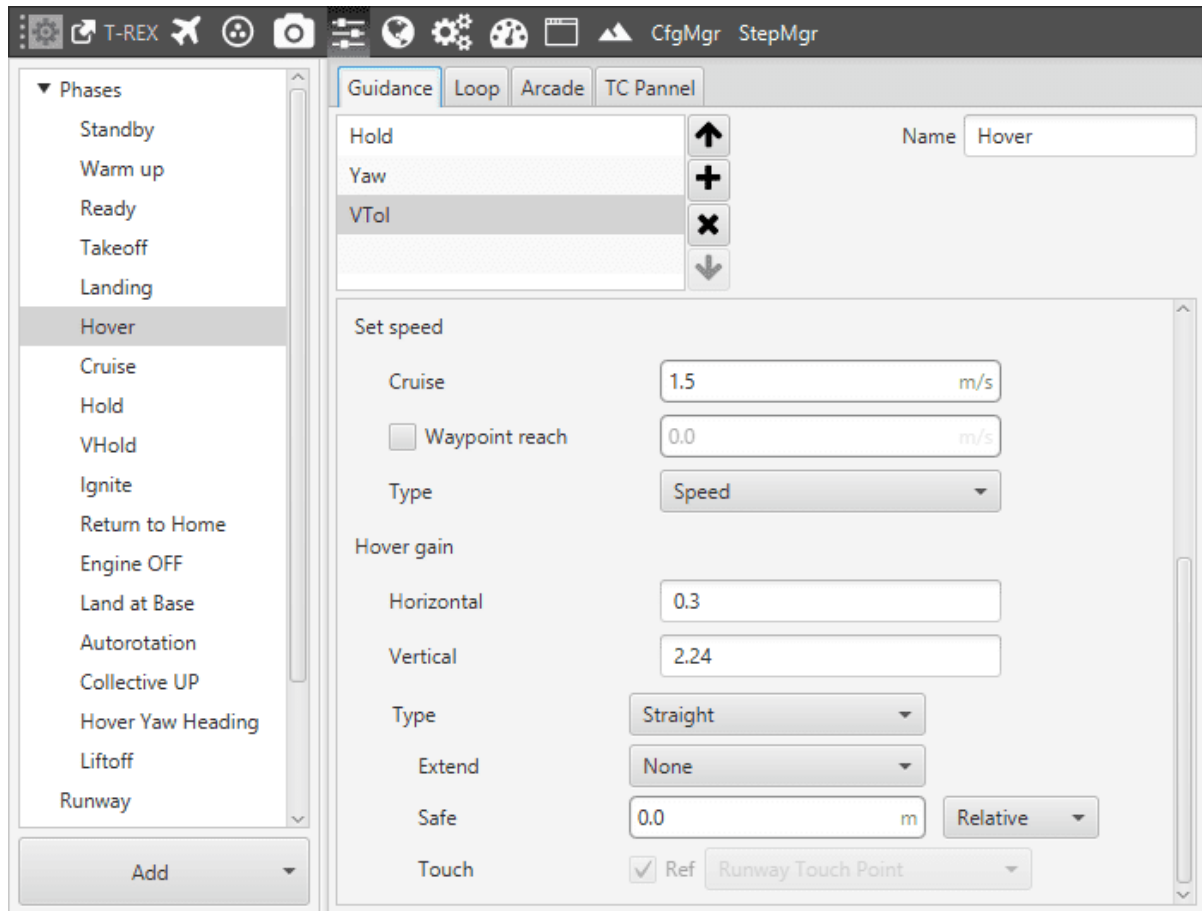
- **Line attraction:** 5.0 [m].
- **Set Speed – Cruise:** 1.5 [m/s].
- **Horizontal Hover Gain:** 0.3.
- **Vertical Hover Gain:** 2.24.
- **Safe Distance:** -30 [m] and Relative (Positive down).



VTol panel

12.2.1.4.2.6 Hover

Hover phase is configured to allow the helicopter maintain the position in the air. The only difference between this phase and the Take Off phase is the **Safe distance** value which is set at 0 [m] in this case.

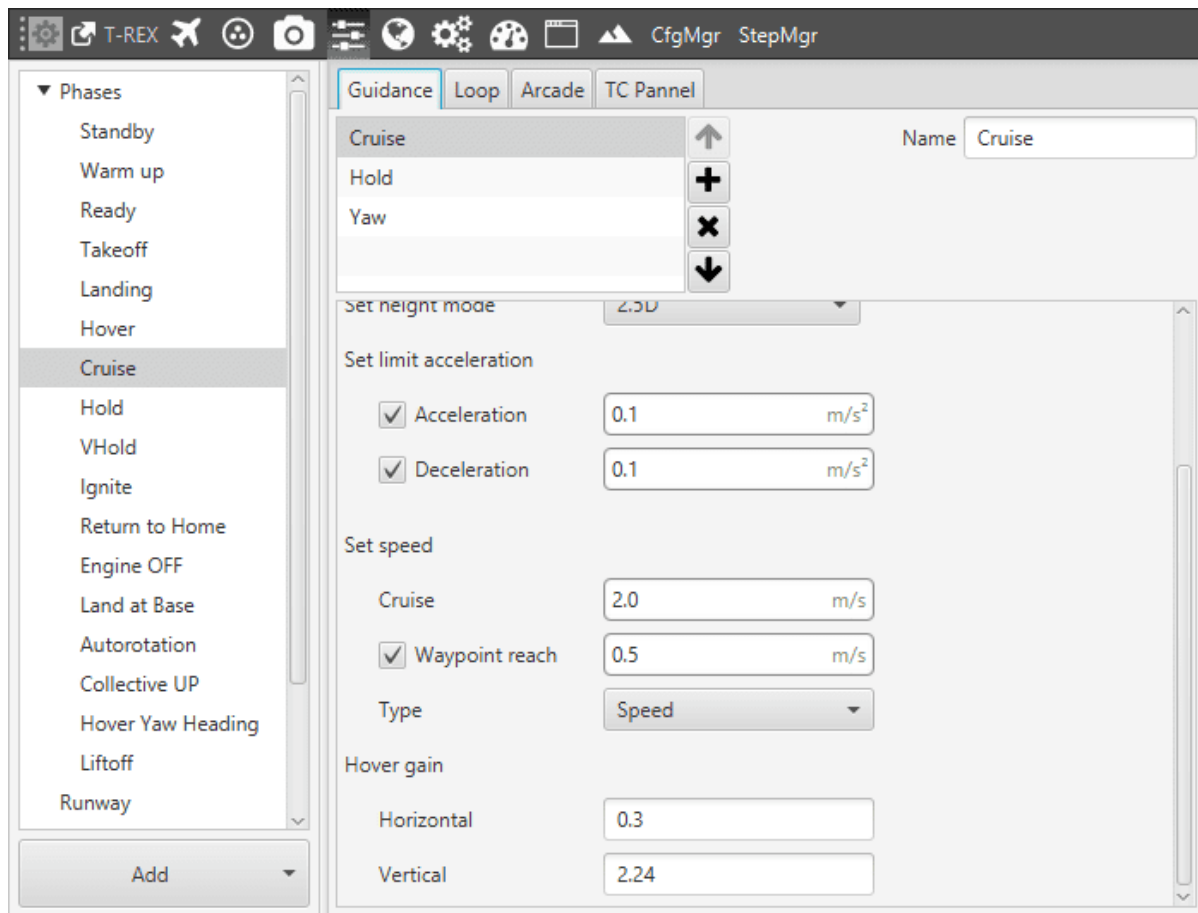


Hover phase panel

12.2.1.4.2.7 Cruise

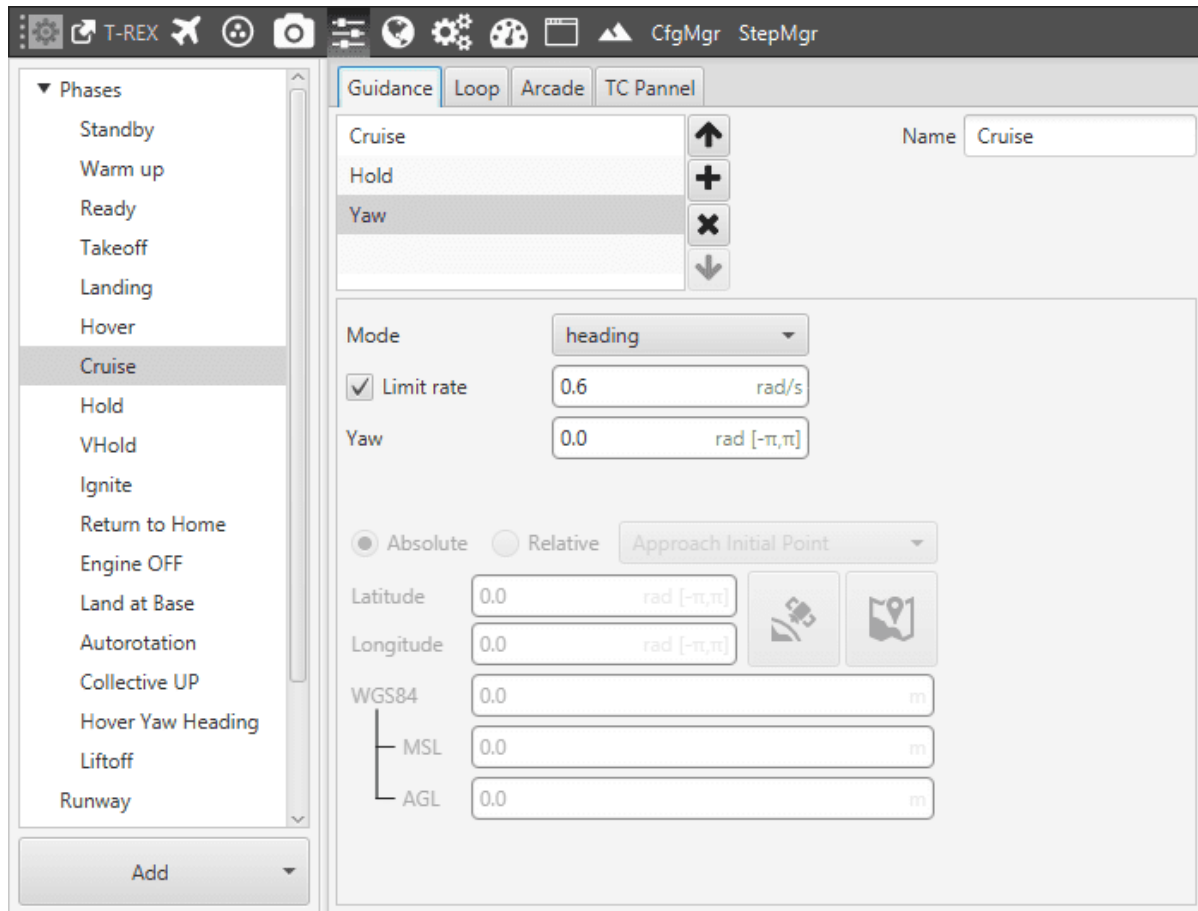
In this phase, the Guidance is Cruise. The parameters to set are the Cruise Speeds (Cruise and Waypoint reach) and Acceleration limits.

- **Acceleration limit:** 0.1.
- **Deceleration limit:** 0.1.
- **Set Speed – Cruise:** 2.0 [m/s].
- **Waypoint reach:** 0.5 [m/s].



Cruise panel

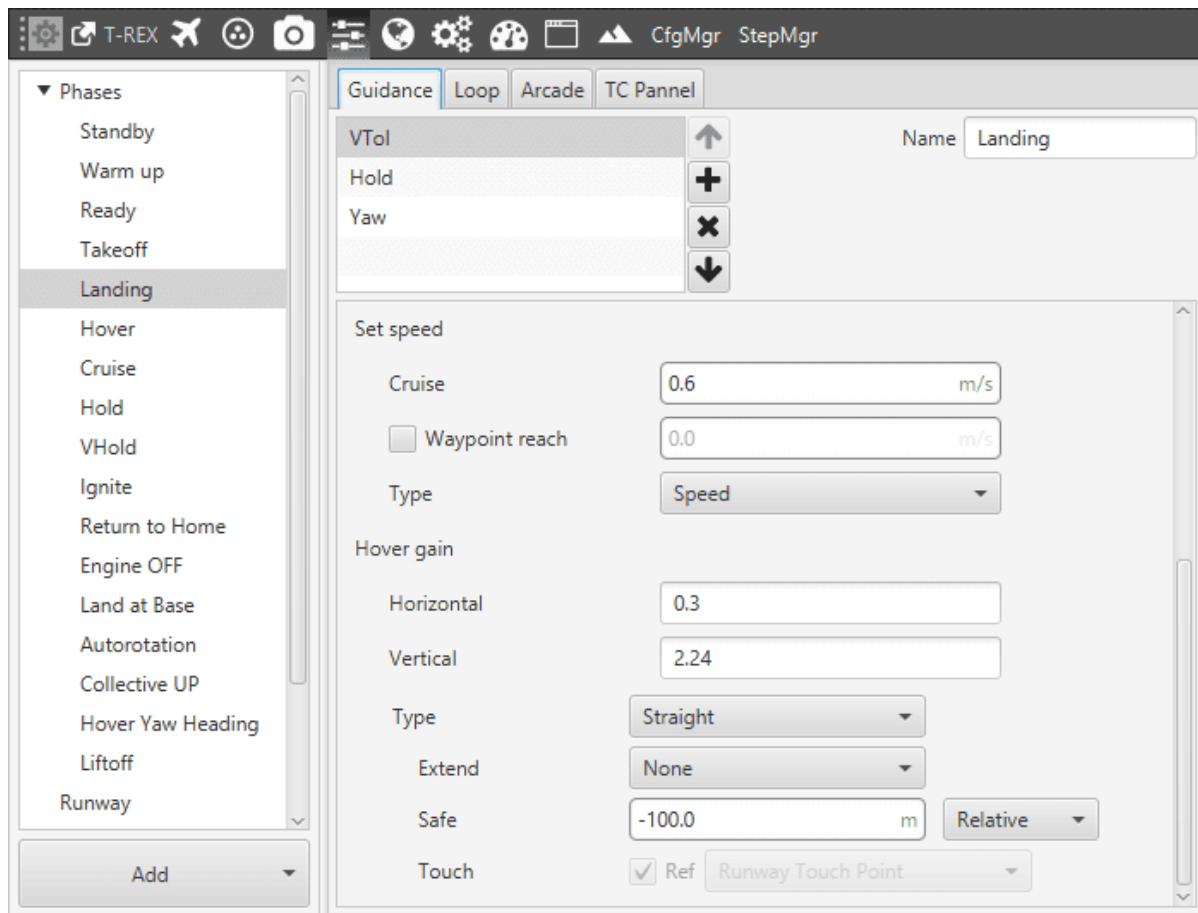
Yaw Guidance is set in order to make the helicopter controlling yaw angle following the heading angle.



Yaw Guidance panel

12.2.1.4.2.8 Landing

This phase has the same settings of the Take-off with a positive value of Safe distance (positive down).



Landing phase panel

In this phase are set the same parameters of the Take Off phase. The only 2 changes are:

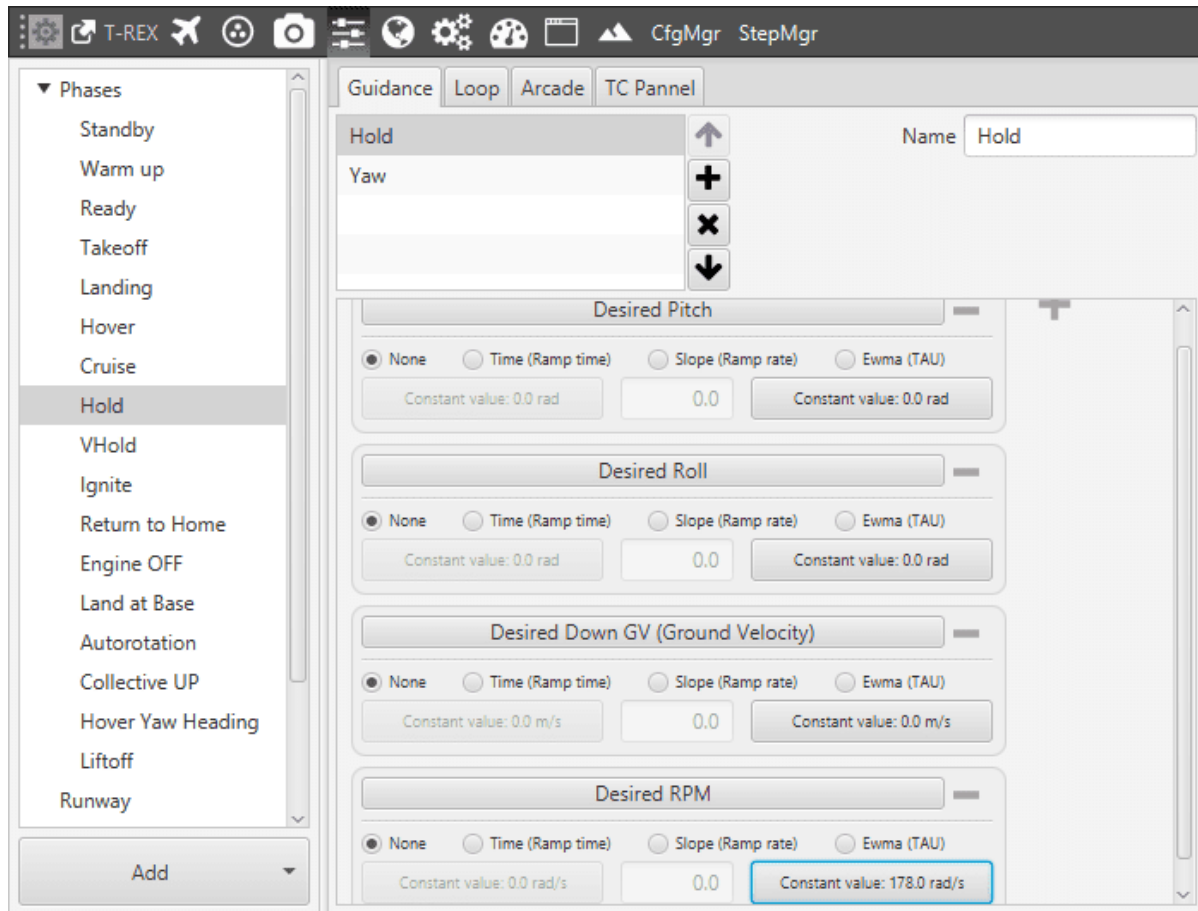
- **Set Speed – Cruise:** 0.6 [m/s].
- **Safe Distance:** -100.0 [m] and Relative (Positive down). In this case, we are supposing that at the instant when the Landing phase starts, the helicopter is flying at AGL < 100 [m].

Yaw Guidance is configured in order to let the helicopter with the Current yaw angle and limiting the **yaw rate** at 0.6 [rad/s].

12.2.1.4.2.9 Hold

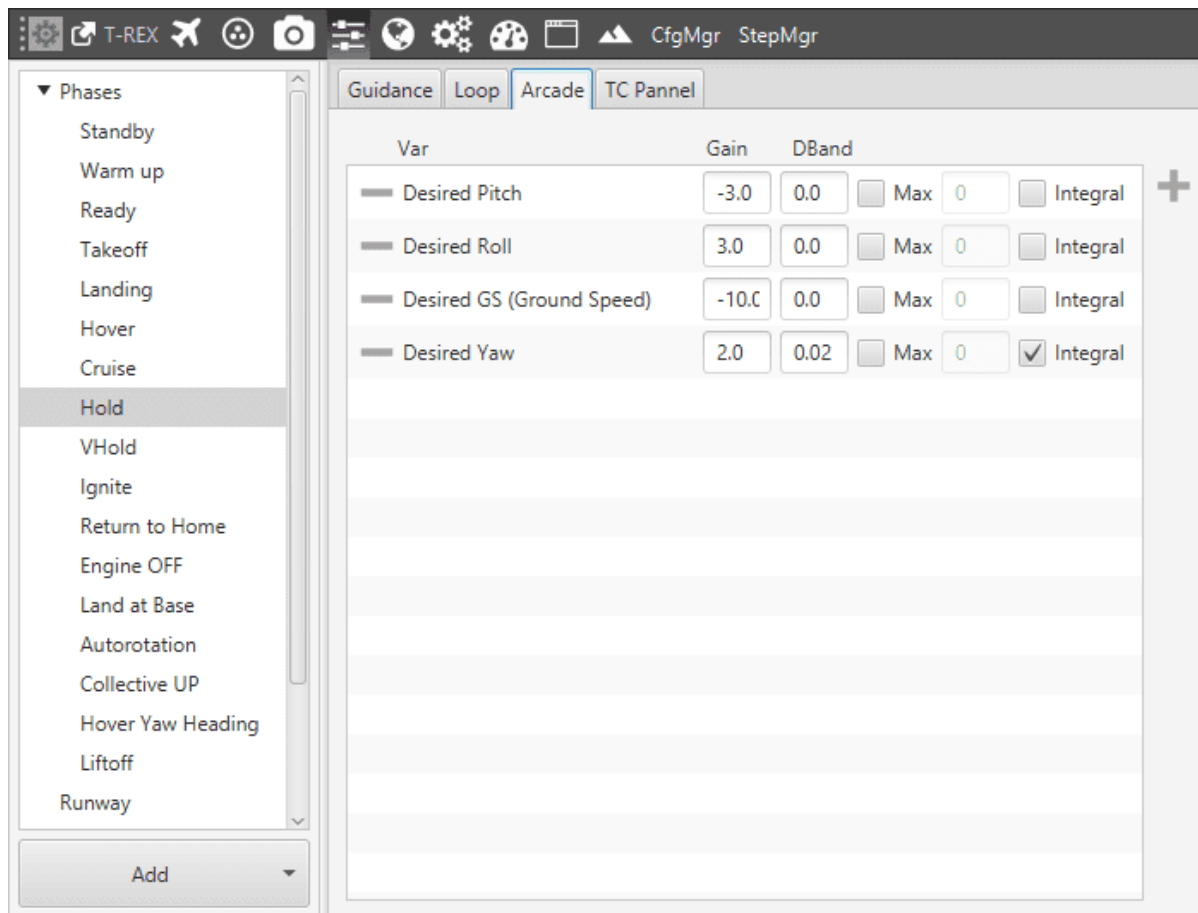
Hold phase is used to perform a Stick controlled flight. The phase guidance is composed of Hold and Yaw guidances. It is a Hold which allows maintaining:

- **Desired pitch:** at 0.0 [rad].
- **Desired roll:** at 0.0 [rad].
- **Desired Down Ground Velocity:** at 0.0 [m/s].
- **Desired RPMs:** at 178.0 [rad/s].



Hold phase panel

In this way, if the stick control is not active, the helicopter will keep its position in the air without the need of phase changing. The **Yaw** Guidance is set on **Current**. In this phase, the Arcade allows controlling the platform depending on the set gains.



Arcade panel

The T-Rex has a helicopter structure. Its control is performed by using the pitch changing of the main rotor blades (cyclic and collective control) and the tail rotor blades depending on the desired attitude change:

- ALTITUDE – Collective control
- PITCH ANGLE – Cyclic control
- ROLL ANGLE – Cyclic control
- YAW ANGLE – Tail blades pitch angle control



T-Rex 600

12.2.1.5 Hybrid

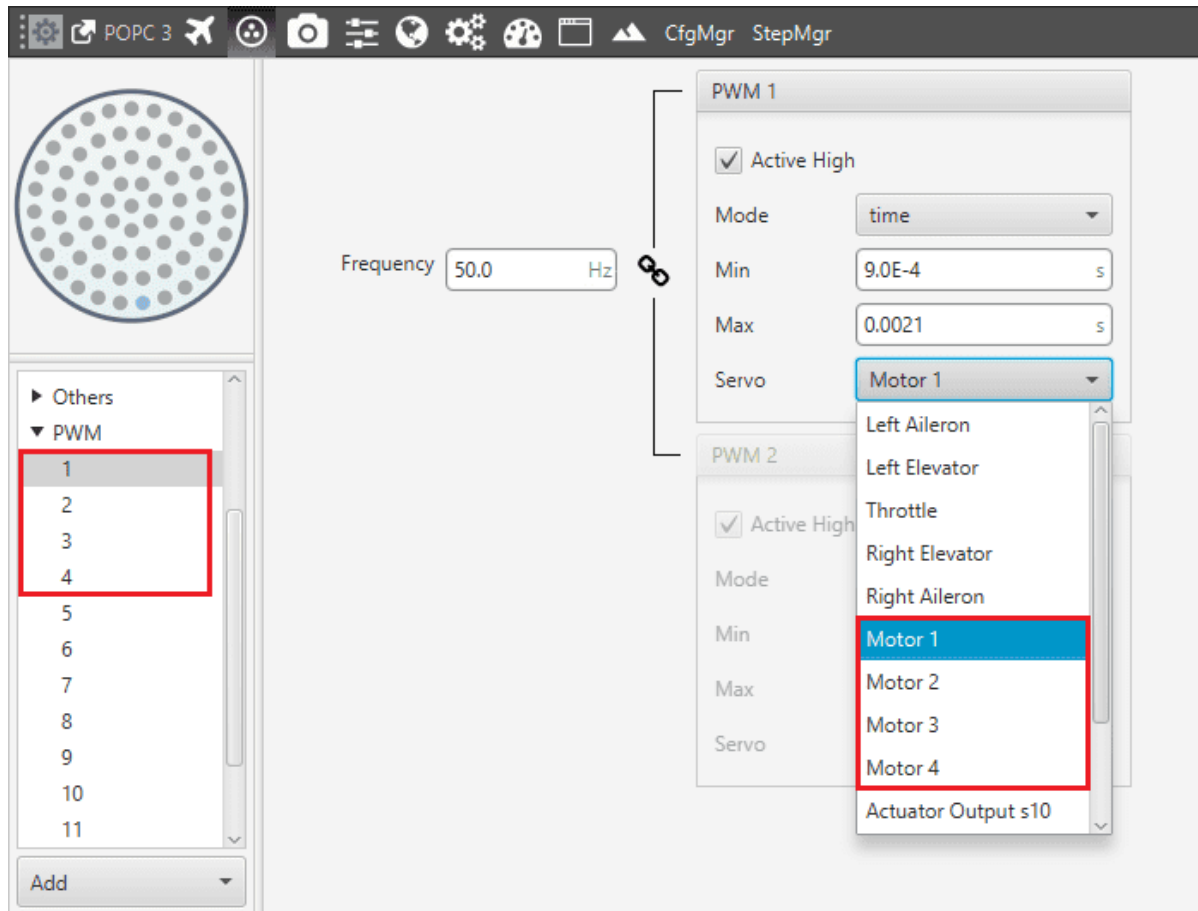
12.2.1.5.1 Servo Configuration

Once the Autopilot has been installed and connected according to the guidelines that appear in section 2.1 of this manual, the first step of the configuration process is the adjustment and trimming of the servos. This section contains the instructions to follow in order to trim the platform servos using Veronte Pipe.

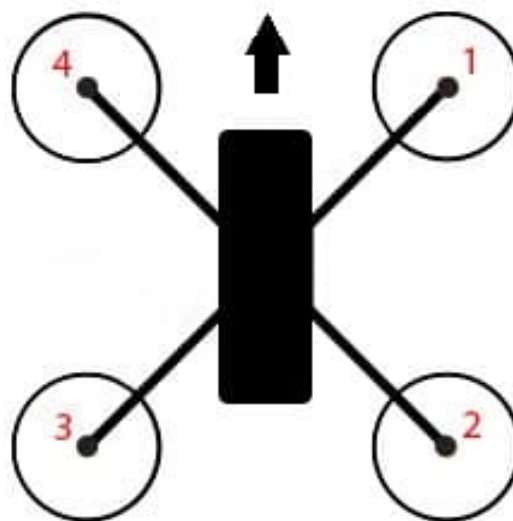
12.2.1.5.1.1 Quadcopter servos

12.2.1.5.1.2 Servos Output

In this case, the controls of the airplane are the four electric motors. Motors can be controlled by changing their thrust (from 0 to 1), so each one of them corresponds to a pin of the connector and they must be positioned in the same order in an S vector who represents the **Actuator Output**. It is possible to connect any pin to any command but the easiest way to perform it and avoid confusion is by following the pins number.



Quadcopter motors connections



Motors numbers

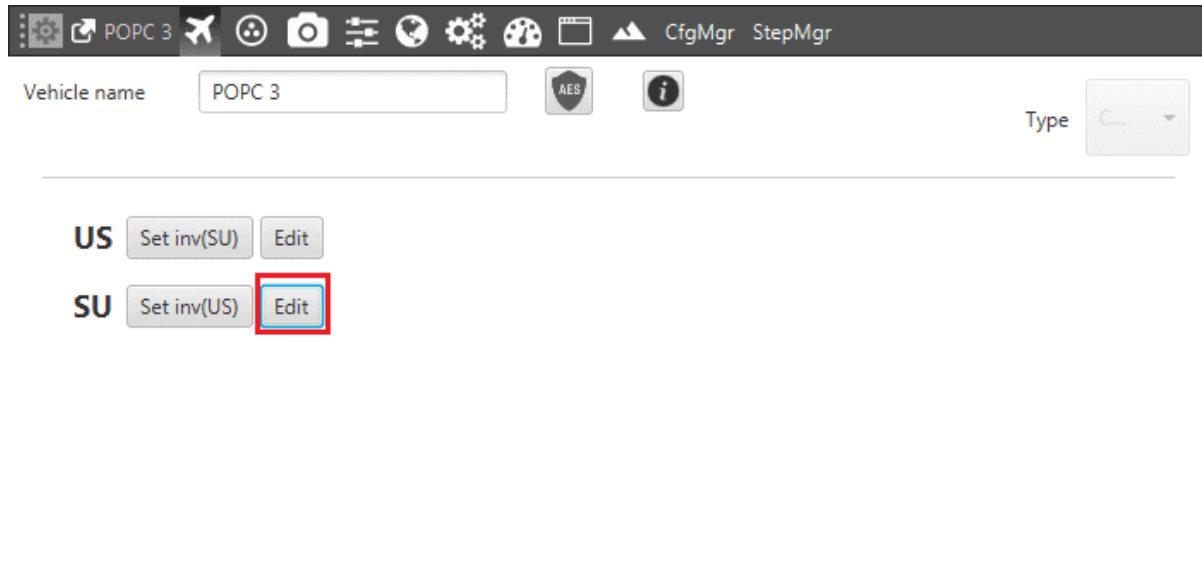
In this case, it is necessary to use 4 pins:

- **Motor 1:** Actuator Output s7.

- **Motor 2:** Actuator Output s8.
- **Motor 3:** Actuator Output s9.
- **Motor 4:** Actuator Output s10.

12.2.1.5.1.3 SU Matrix

At this point, the S vector is defined and the SU matrix can be edited. By clicking on **Edit** it is possible to configure the relation between the controller outputs (**U** vector) and the servo movements (**S** vector).



SU matrix edit

Warning: This panel shows the reference system of the aircraft too. It must be positioned in the same way of the Autopilot's one. If it results different, it can be edited by clicking on the corresponding axis in order to reverse its direction.

| | Control Output u1 - Pitching (Plane) | Control Output u2 - Rolling (Plane) | Control Output u3 - Yawing (Plane) | Control Output u4 - Pitching (Quad) | Control Output u5 - Throttle (Quad) | Control Output u6 - Rolling (Quad) | Control Output u7 - Yawing (Quad) | Control Output u8 - Throttle (Plane) |
|---------------|--------------------------------------|-------------------------------------|------------------------------------|-------------------------------------|-------------------------------------|------------------------------------|-----------------------------------|--------------------------------------|
| Right Aileron | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| Rudder | 0.0 | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| Throttle | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 |
| Elevator | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| Left Aileron | 0.0 | -1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| Motor 1 | 0.0 | 0.0 | 0.0 | -0.35355338 | 1.0 | 0.35355338 | -0.25 | 0.0 |
| Motor 2 | 0.0 | 0.0 | 0.0 | -0.35355338 | 1.0 | -0.35355338 | 0.25 | 0.0 |
| Motor 3 | 0.0 | 0.0 | 0.0 | 0.35355338 | 1.0 | 0.35355338 | 0.25 | 0.0 |
| Motor 4 | 0.0 | 0.0 | 0.0 | 0.35355338 | 1.0 | -0.35355338 | -0.25 | 0.0 |

SU Matrix (blue – airplane, red – quadcopter)

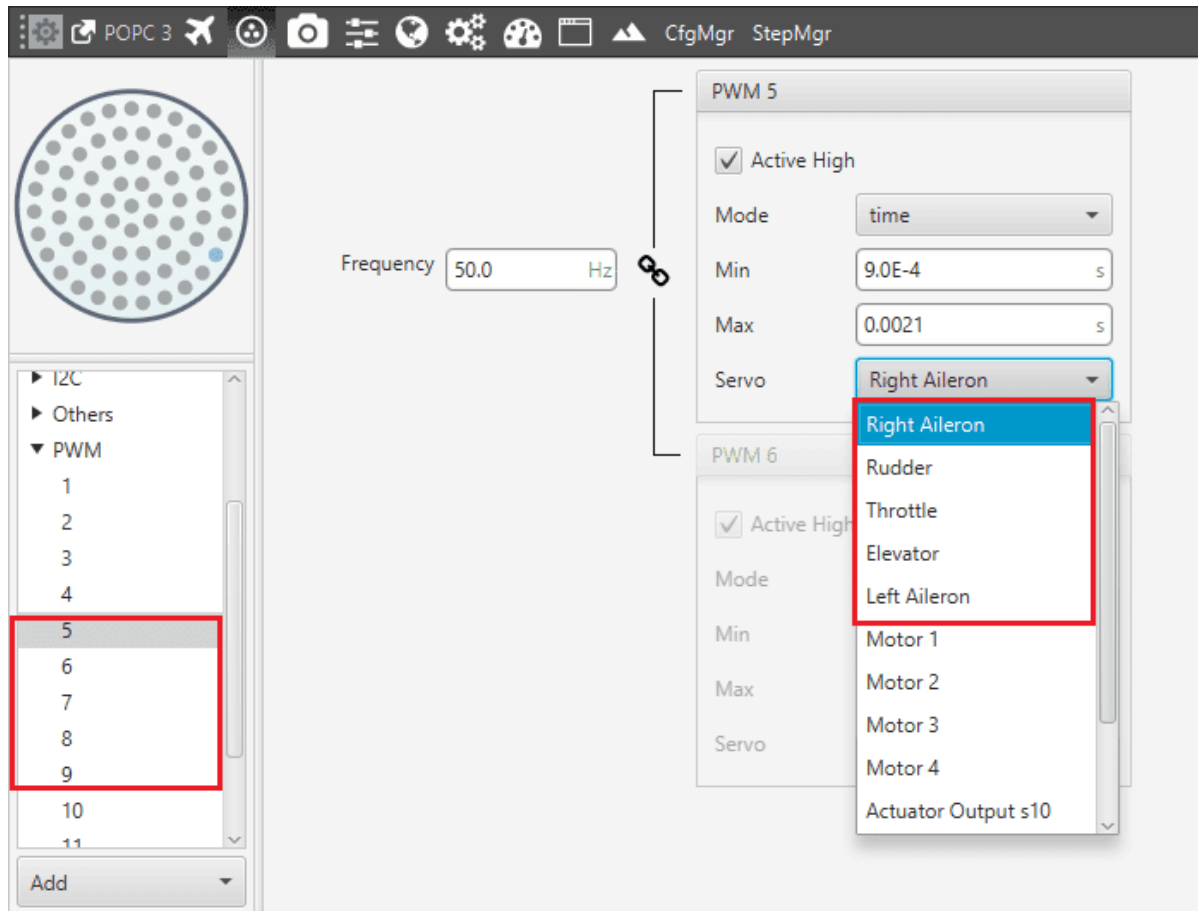
The Hybrid during quadcopter phases is configured as follow:

1. **Pitch Angle Control:** Control Output 5 is configured to perform a positive pitch angle change when motors 1-4 decrease their RPMs and motors 2-3 increase RPM value.
2. **Thrust Control:** the Control Output 6 is the one that allows a thrusting change (0-1).
3. **Roll Angle Control:** Control Output 7 is configured to perform a positive roll angle change when motors 1-2 decrease their RPMs and motors 3-4 increase RPM value.
4. **Yaw Angle Control:** Control Output 8 is configured to perform a positive yaw angle change when motors 2-4 decrease their RPMs and motors 1-3 increase RPM value, in this case with different proportions.

12.2.1.5.1.4 Plane Servos

12.2.1.5.1.5 Servos Output

In this case, it is necessary to configure a conventional aircraft configuration, so there are five controls: elevator, two ailerons, rudder and throttle. Each one of these controls is assigned to a value of the **Actuator Output** vector (s) according to the pin where it is connected and avoiding to use the quadcopter connection pins.



Plane servos connections

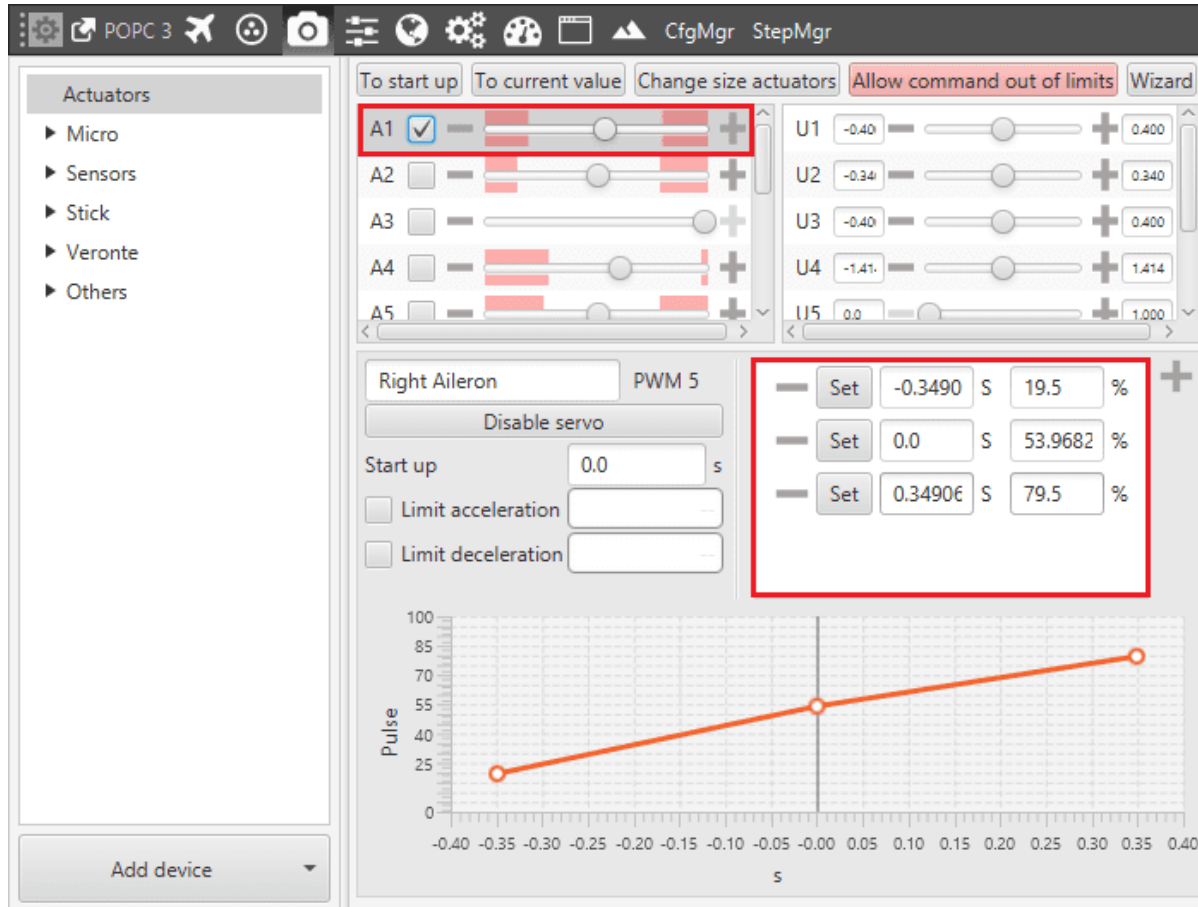
1. **Right Aileron:** Actuator Output s1
2. **Elevator:** Actuator Output s4
3. **Throttle:** Actuator Output s3
4. **Rudder:** Actuator Output s2
5. **Left Aileron:** Actuator Output s5

When the five controls are defined, the SU matrix plane part (blue part) can now be completed. In the case of this Hybrid, each control channel is related with only one servo: the pitching with the elevator, the rolling with the ailerons, the yawing with the rudder and the thrusting with the throttle:

- **Pitching (u1):** Actuator Output s4 (Elevator)
- **Thrusting (u2):** Actuator Output s3 (Throttle)
- **Rolling (u3):** Actuator Output s1 and s5 (Ailerons)
- **Yawing (u4):** Actuator Output s2 (Rudder)

12.2.1.5.1.6 System Trim

As a final step, the system has to be trimmed. This action can be performed by moving servos in three different positions: zero position, minimum and maximum deflection angle (angles are usually limited physically). These positions must be inserted and saved in the software by clicking on “Set” when the actuator is in the desired position.



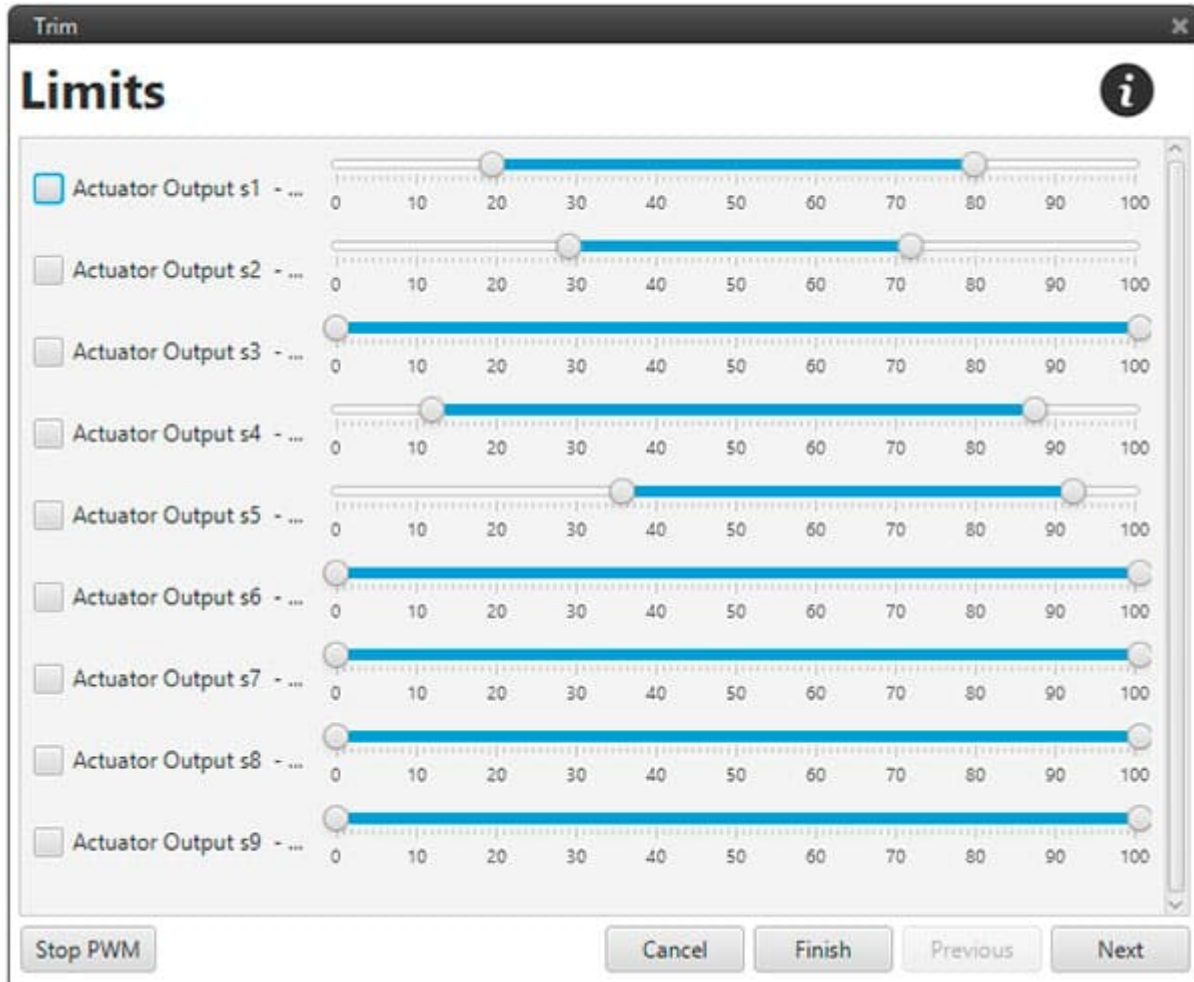
Actuator 1 trimming

The picture shows the setting of the elevator number 1:

- **Minimum:** -0.34906 [rad] deflection ; 19.5%.
- **Zero position:** 0.0[rad] deflection ; 53.9682%.
- **Maximum:** 0.34906 [rad] deflection; 79.5%.

Warning: The actuators can be moved directly from Veronte Pipe only when the system is in an **Initial** phase. During the actuator run, if the desired position is in the “out of range” zone (red zone), it is possible to click on “Allow command out of limits” in order to move completely the actuator and find the correct position.

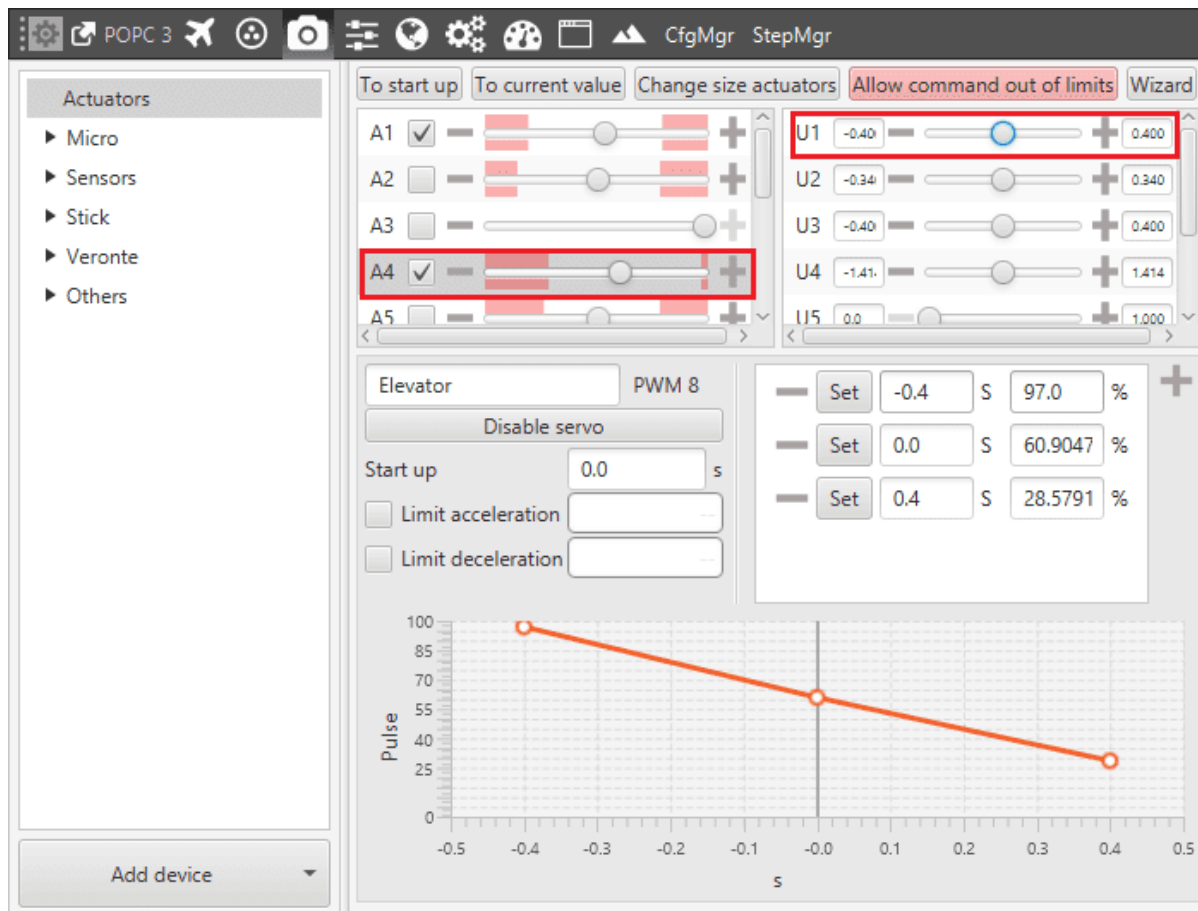
This procedure can be performed in the same way by using a “wizard”. This tool allows moving actuators limits easily and finding the correct range.



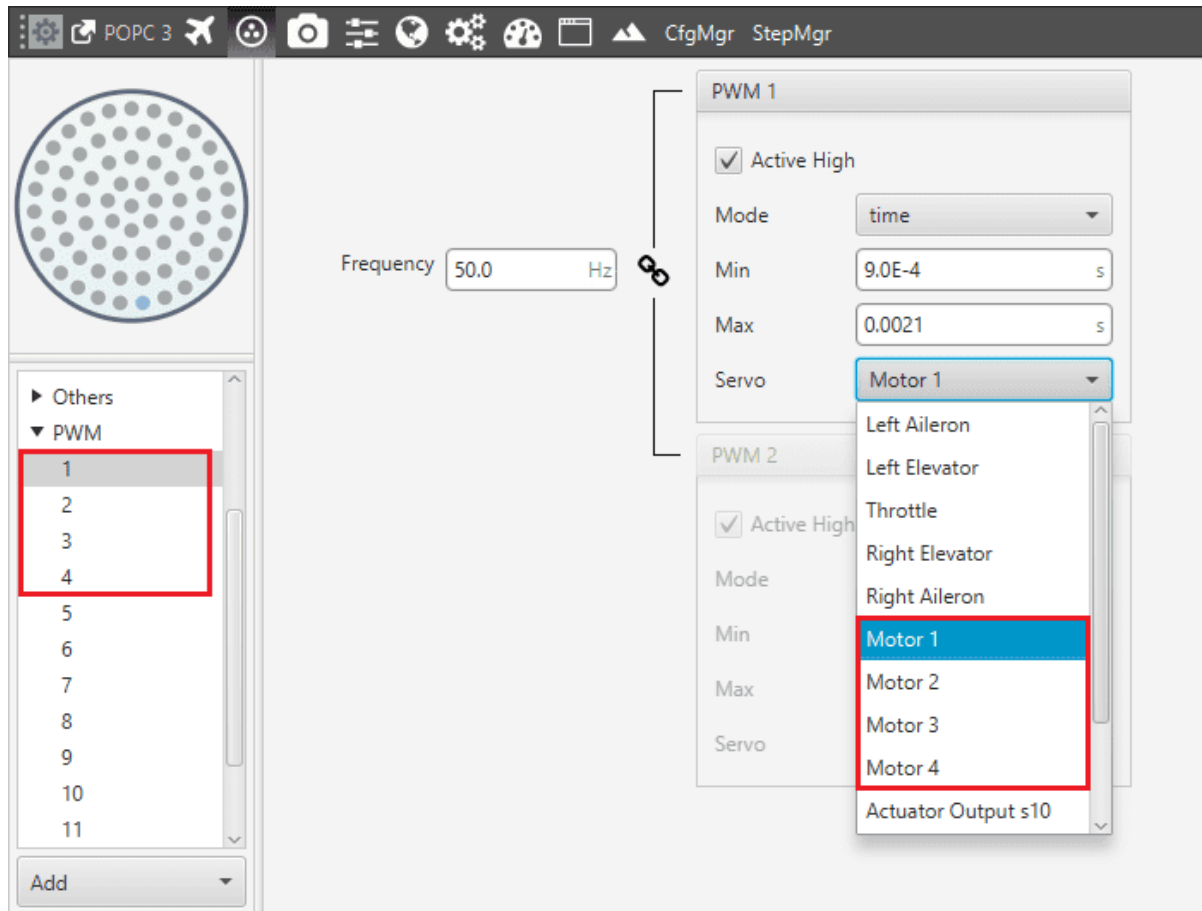
Trim wizard tool

In order to perform a final checking, it is possible to select the desired channel and testing pitch, roll, yaw and thrust controls.

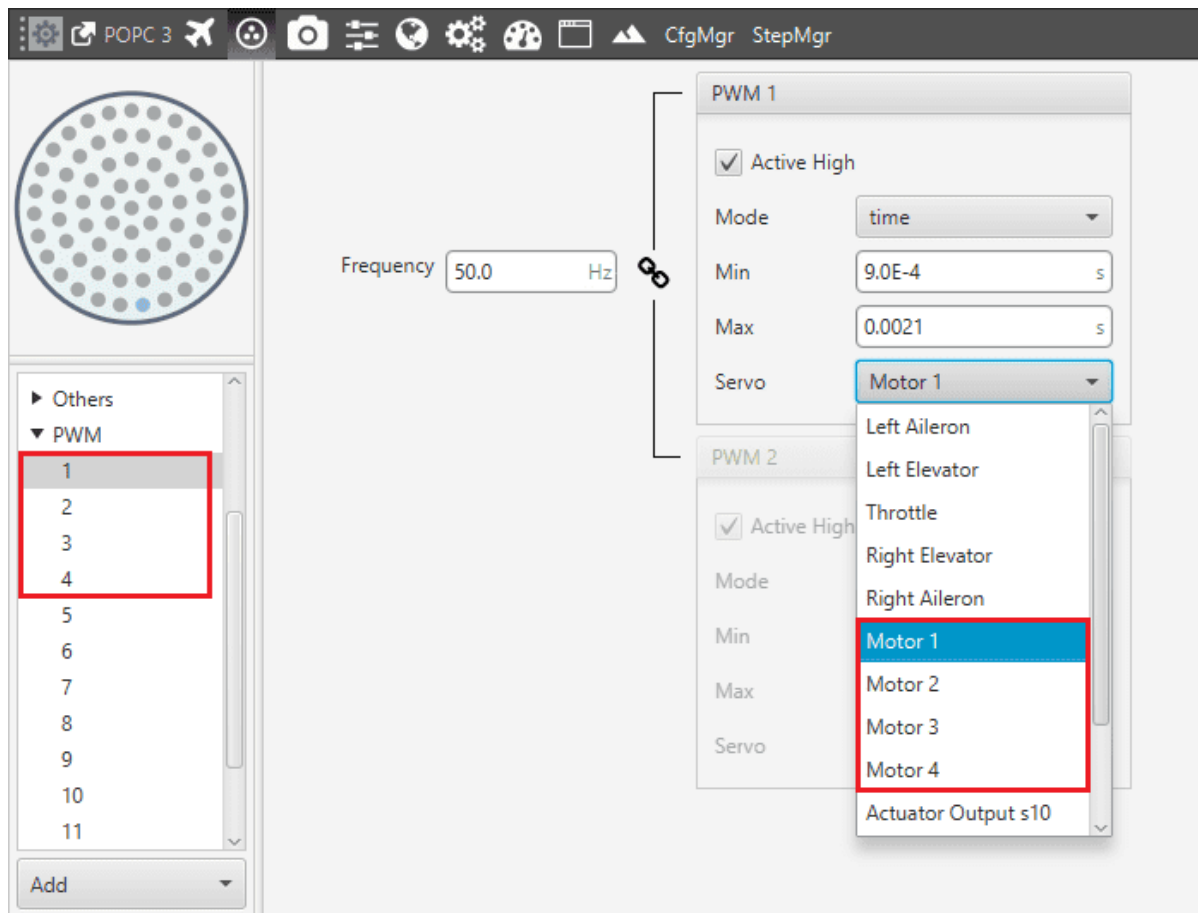
The image below shows a pitching output testing. By moving the U1 control, surfaces must change the position according to the reference system: positive corresponds to nose down and negative to nose up.



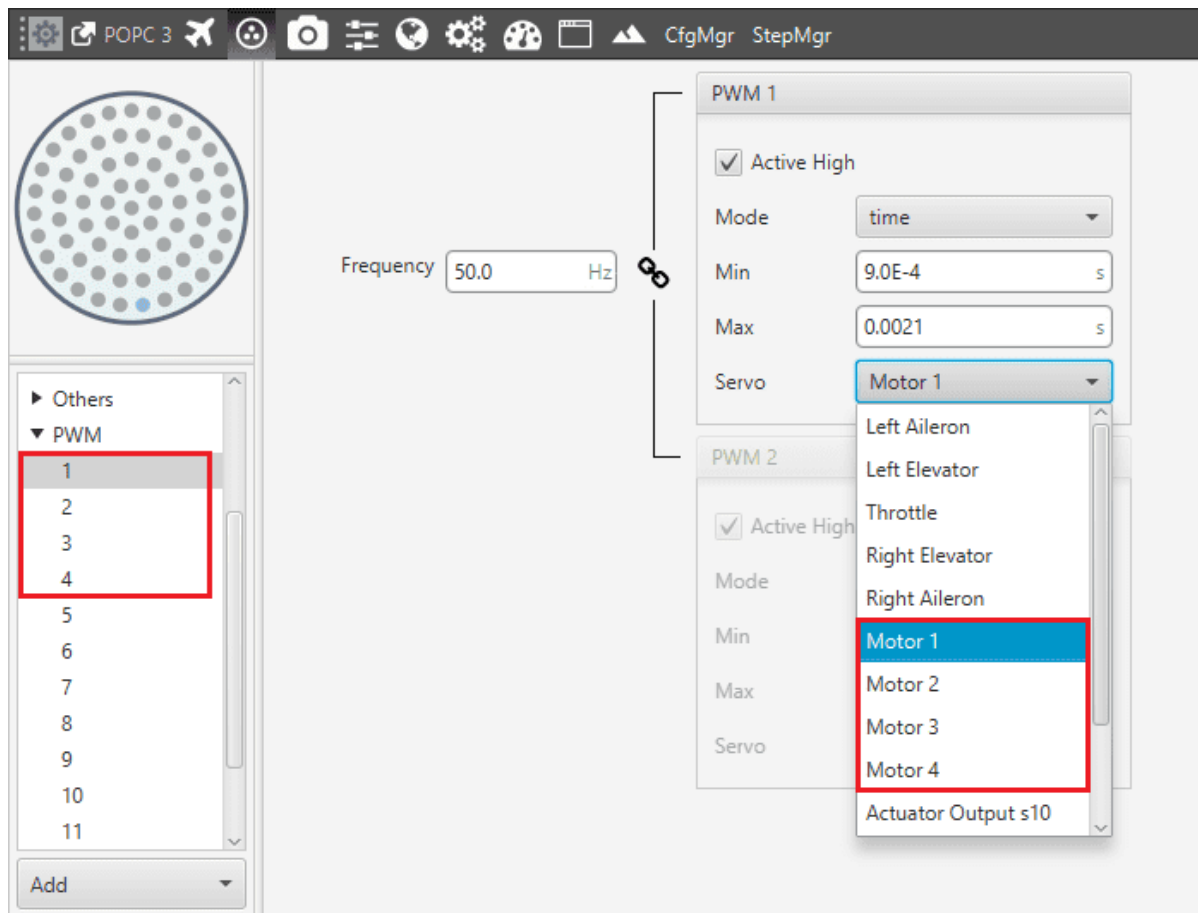
Pitching test



Quadcopter motors connections



Quadcopter motors connections



Quadcopter motors connections

12.2.1.5.2 Mission Phases

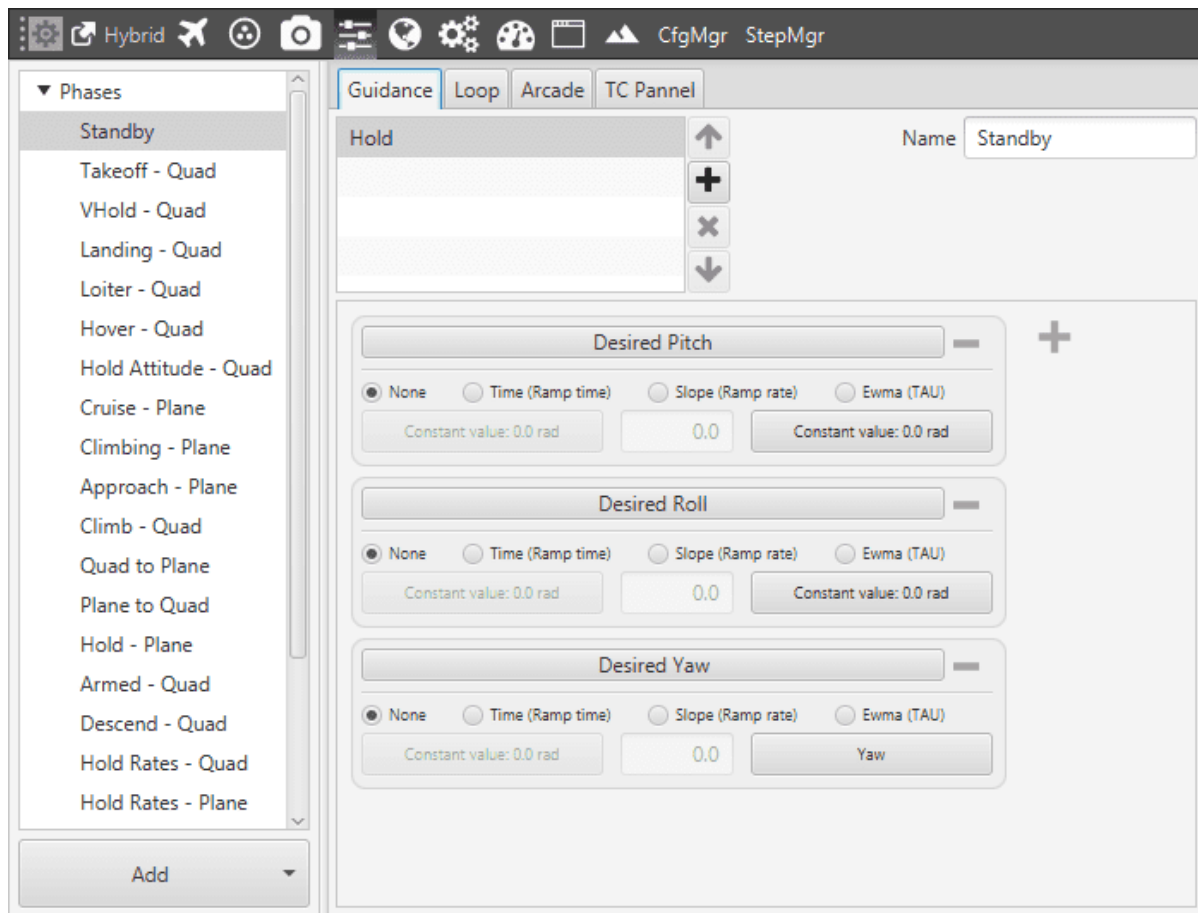
In this section, it will be explained the Hybrid typical mission profile and the mission phases will be detailed. The common phases of Quadcopter and Plane are:

- Standby
- Transition Quad to Plane
- Transition Plane to Quad

For all the other phases, the flight type will be specified.

12.2.1.5.2.1 Standby

Standby phase is a preliminary phase of the operation. Normally, the automation which allows the system to pass to Standby phase requires the GPS signal.



Standby phase panel

The guidance is a **Hold** of three variables:

- **Desired Pitch and Roll angles:** at 0 [rad].
- **Desired Yaw angle:** set at the current.



Hybrid

This hybrid aircraft has an airplane – quadcopter integrated structure. Its control is performed by using typical surfaces control for an airplane configuration during plane phases:

- PITCH
- ROLL
- YAW
- THRUST

And a differential thrust for each quadcopter motor depending on the desired attitude change during quadcopter phases.

The two transition phases are the most critical, in fact, it is important to keep the aircraft control during plane-quadcopter changing avoiding command overlaps.

This section contains the whole configuration process of four different platforms in Veronte Pipe. All the different menus that appear in the software will be covered so the final user of the system gains a fully understanding of how to totally configure its platform through some real examples.

The platforms available are:

- Aircraft: **Mentor.**
- Flying Wing: **W210.**
- Multicopter: **M400.**
- Helicopter: **T-Rex.**

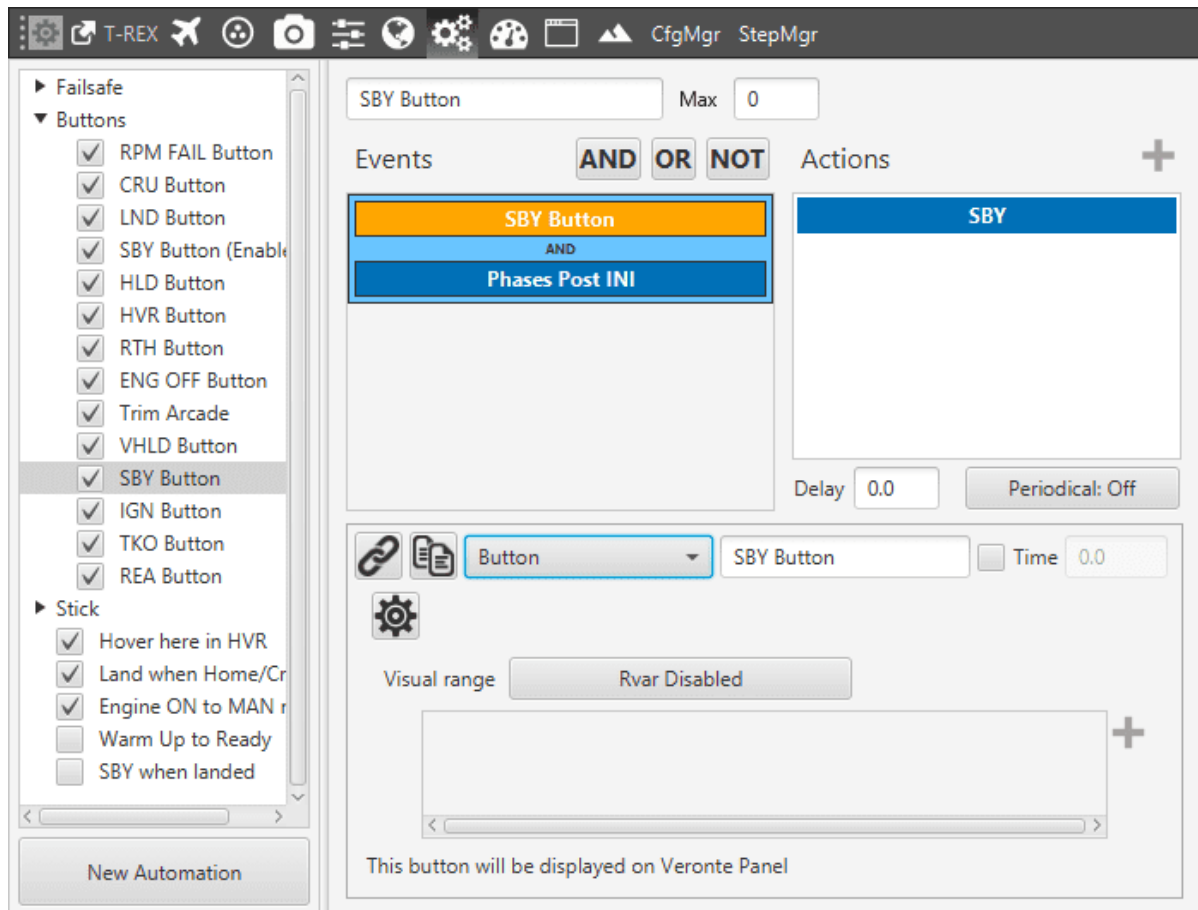
In this section we will explain the configuration of some devices and platforms to improve the understanding of the user.

12.3 Automations

12.3.1 Veronte Panel Buttons

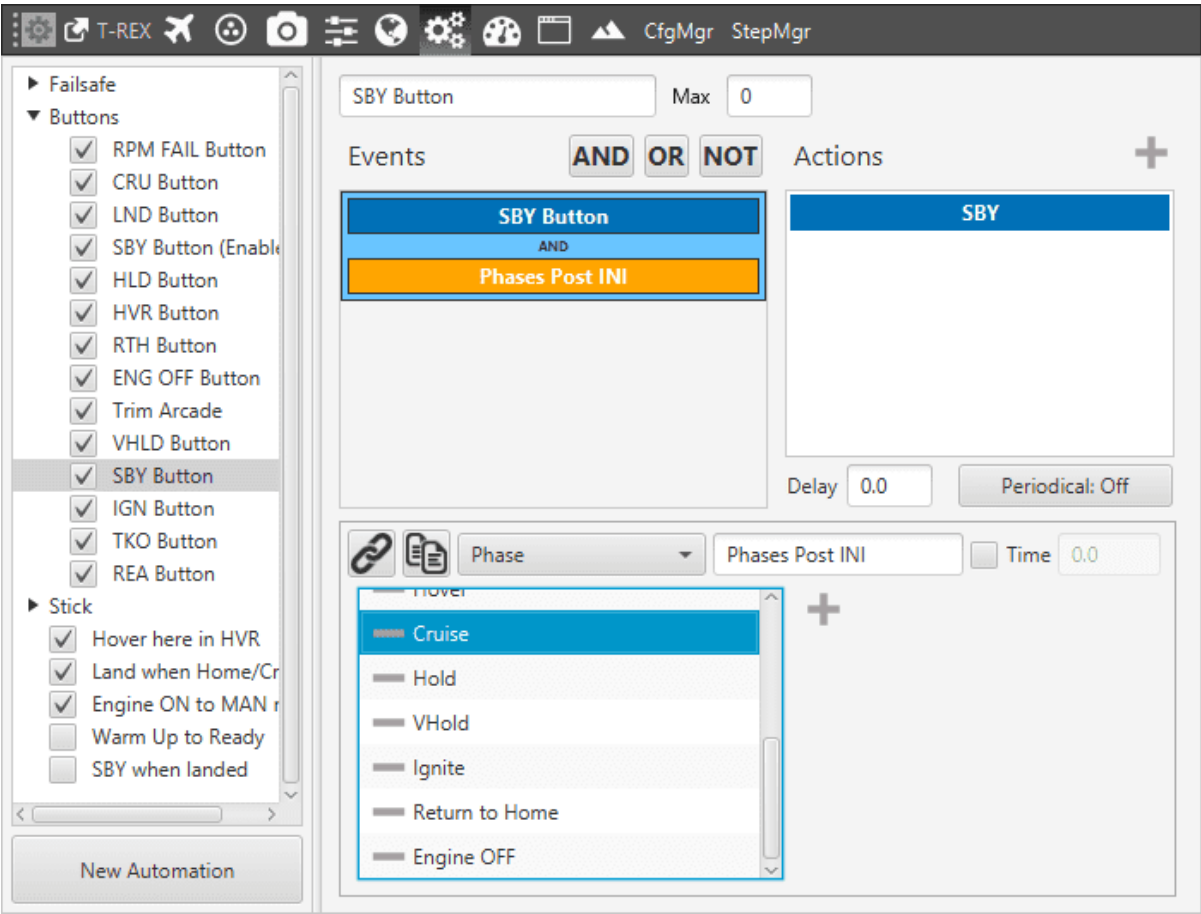
12.3.1.1 Phase Change Buttons

The first automations to be created are the ones linked to the buttons of Veronte Panel. When clicking one of these buttons the phase is changed to the one shown on the button label. The image below shows the Standby Button automation.

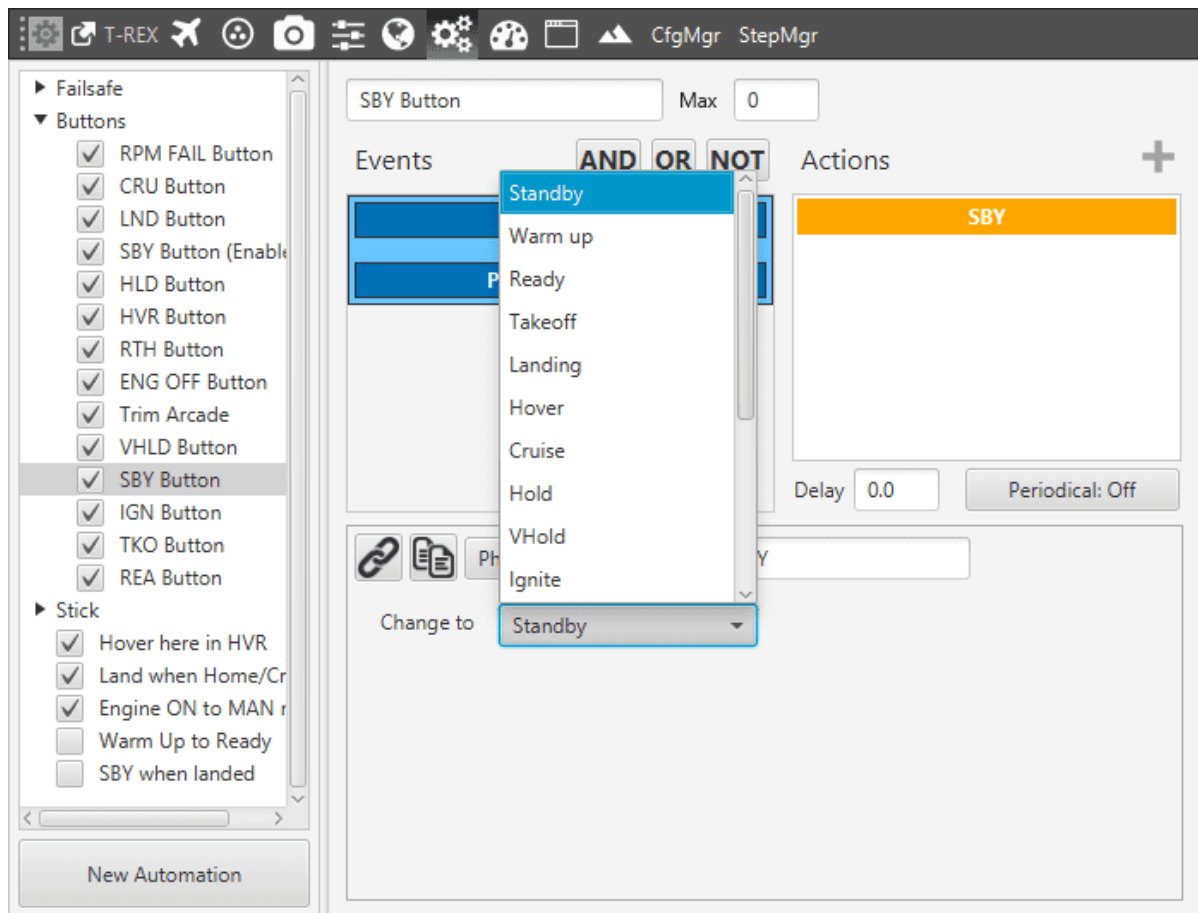


Standby button automation

Normally, besides the button event, a safety condition is added to the automation, which consists on letting only the system to change to a phase when being in a certain set of phases. It is only necessary to create an **AND** condition with the phases to change from. For example, in this case, the set of phases is shown in the following image.



Phases set

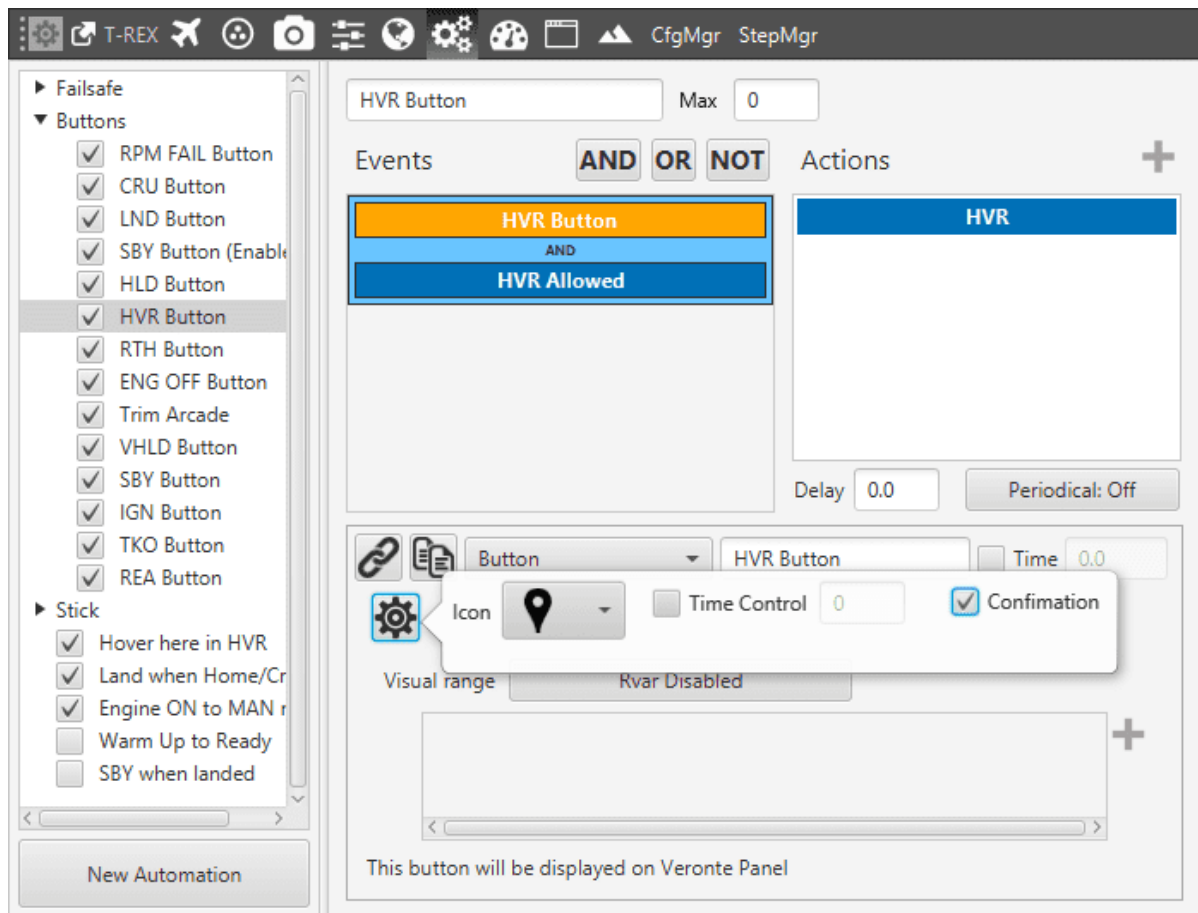


Standby action

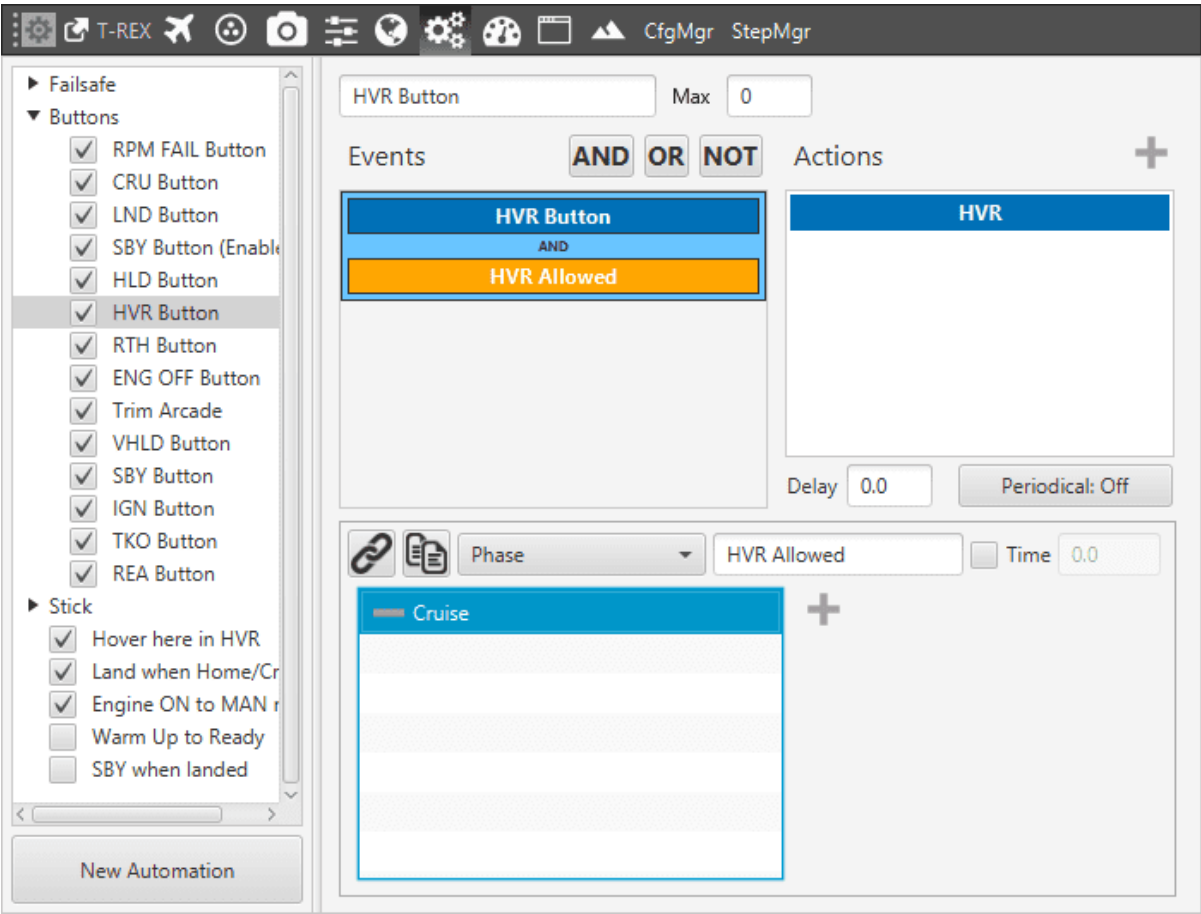
In the previous figure, the automation is the change to StandBy when clicking on Veronte panel, being on any phase selected in the event. This process is normally repeated for the rest of phases.

12.3.1.2 Generic Button

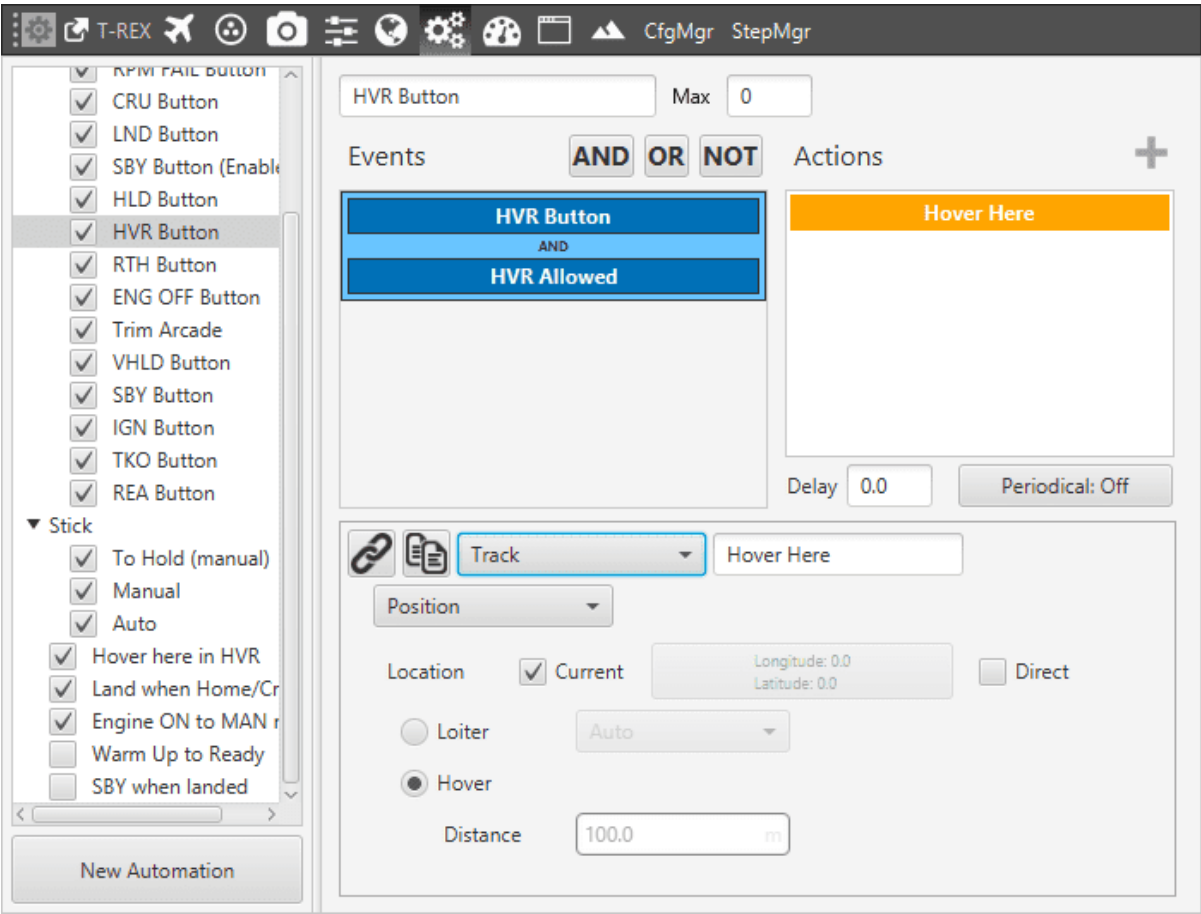
There are other types of button which can be created on the Veronte Panel. In this case will be detailed the Hover Button which allows performing a hover (only for multicopters, for airplanes could be set a loiter point) on the point of the map exactly when the linked button is pushed. The images below show the automation set.



Button Icon selection



Phases set

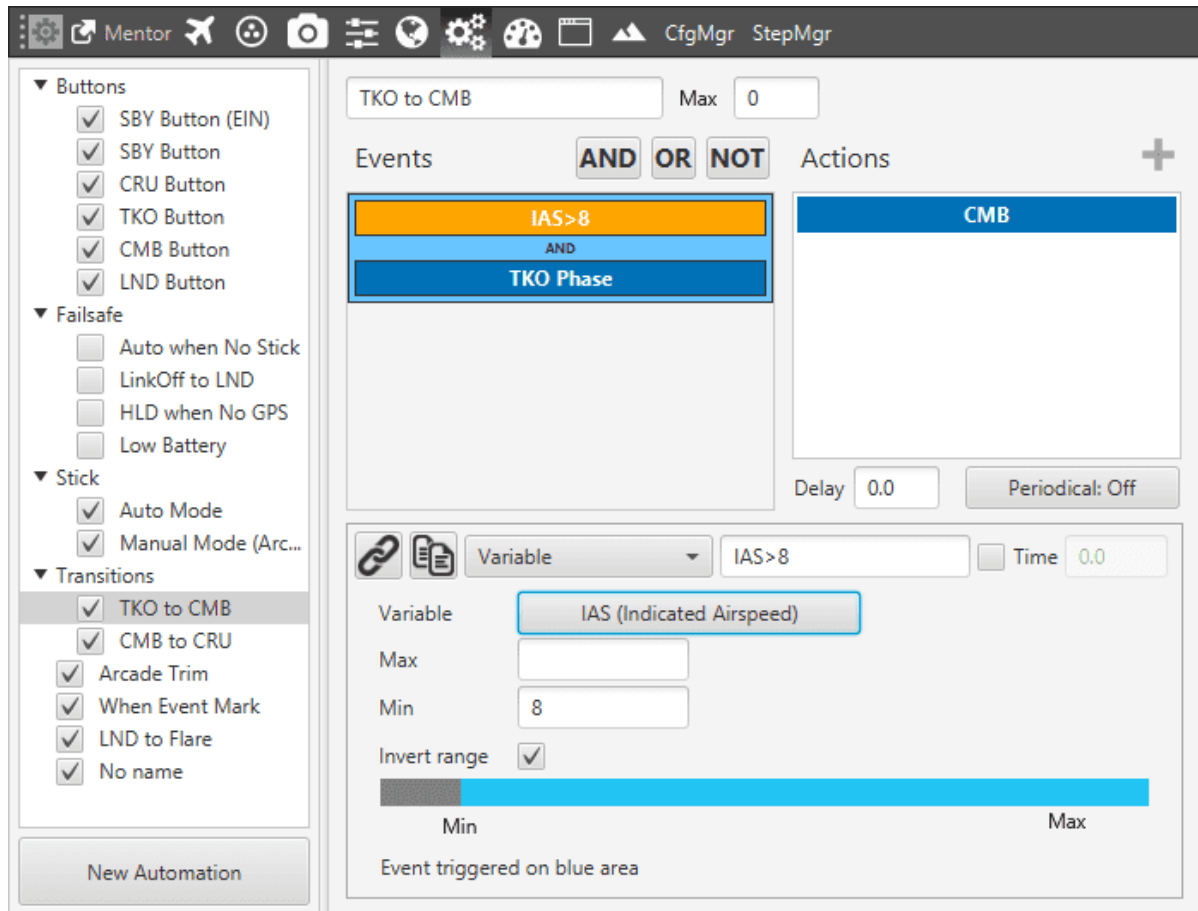


Hover button defining

12.3.2 Phase Changing

12.3.2.1 Takeoff to Climb Change

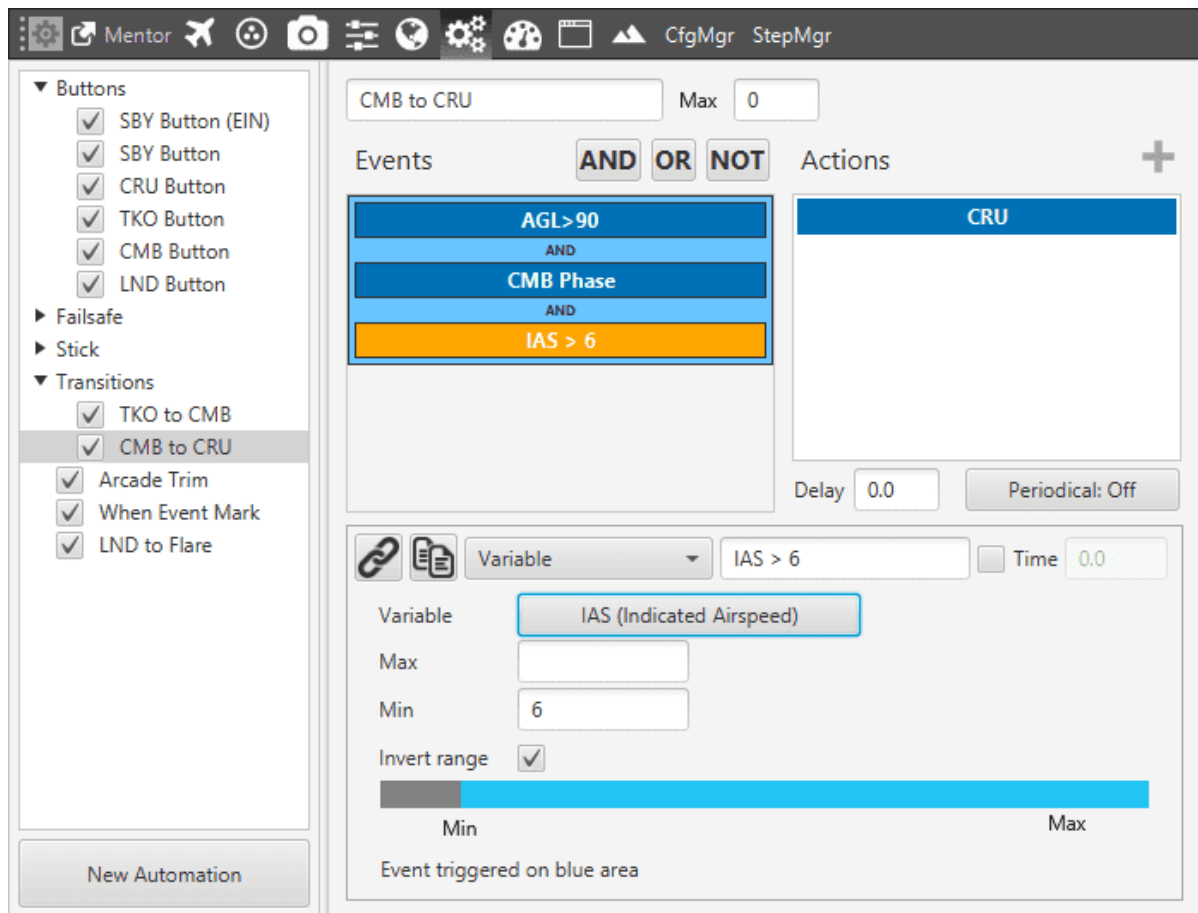
The change from the take off to the climbing phase, in this case, occurs when the IAS of the airplane on the runway is greater than 8 m/s. The phase condition is set on Takeoff phase and the action is defined to pass in Climbing phase.



Takeoff to climb automation

12.3.2.2 Climbing to Cruise Change

This kind of phase change is normally performed when a certain altitude is reached. In this case, the airplane changes to the “Cruise” phase where it starts to follow the path determined by the user. As a safety condition, the change will only happen when having a speed greater than 6 meters per second.



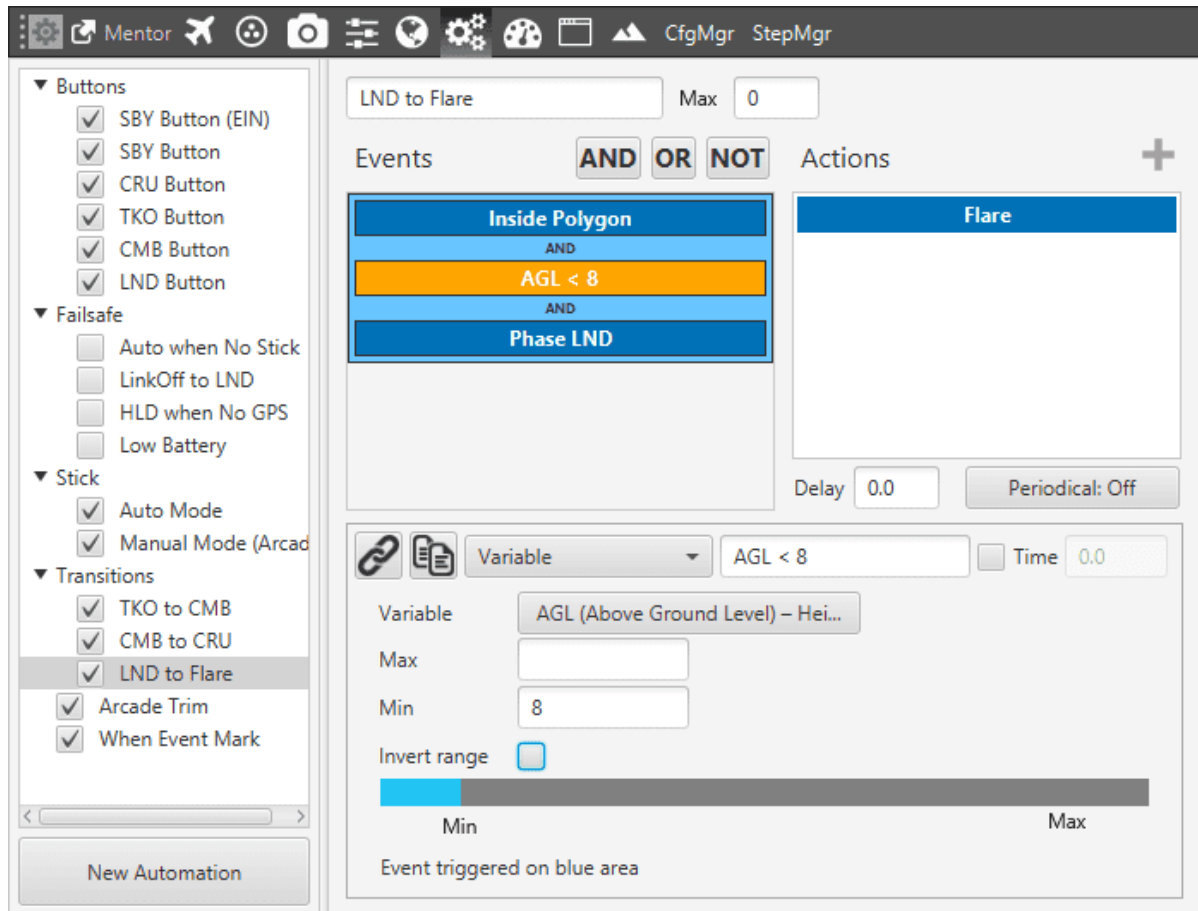
Climbing to cruise change definition

The AGL to the change of phase is 90 meters. Both the velocity and altitude are specified through an event of type “Variable”.

12.3.2.3 Landing to Flare Change

The change from the landing to flare phase is performed when the aircraft is above the runway and about to touch the ground. This idea is implemented with a set of three events:

- **AGL:** the aircraft is below a certain altitude, in this case, 8 meters.
- **Inside Polygon:** Mentor enters in a polygon defined in the map on the runway head.
- **Landing Phase:** the aircraft is landing phase.



Landing to Flare

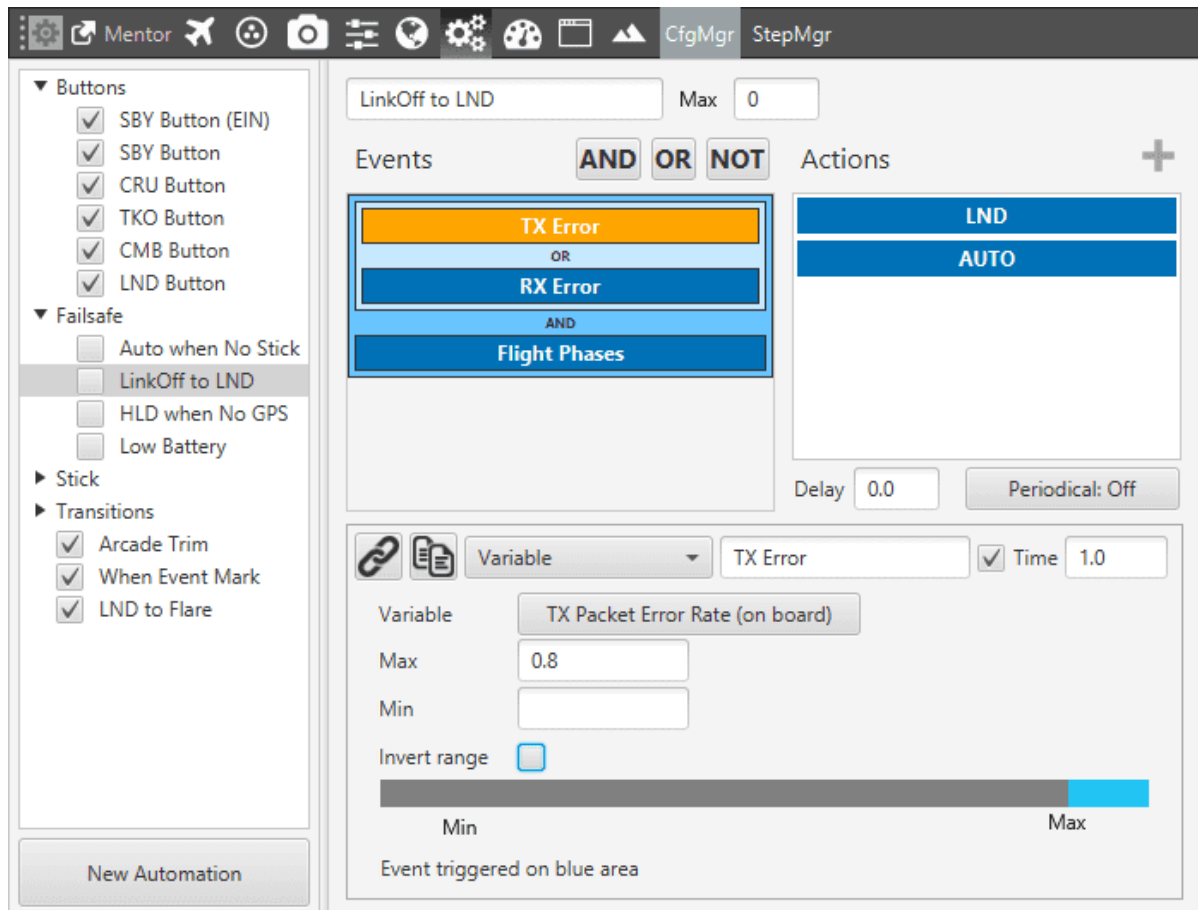
When these events are fulfilled, airplane enters in the Flare phase and lands.

12.3.3 Failsafe

Veronte allows users to create different types of Failsafe automations according to they need combining different Actions and Events. The more common automations are shown below.

12.3.3.1 Radio Error

When the radio connection from the ground station to Veronte Air is lost, the aircraft is forced to change to auto mode and landing phase.

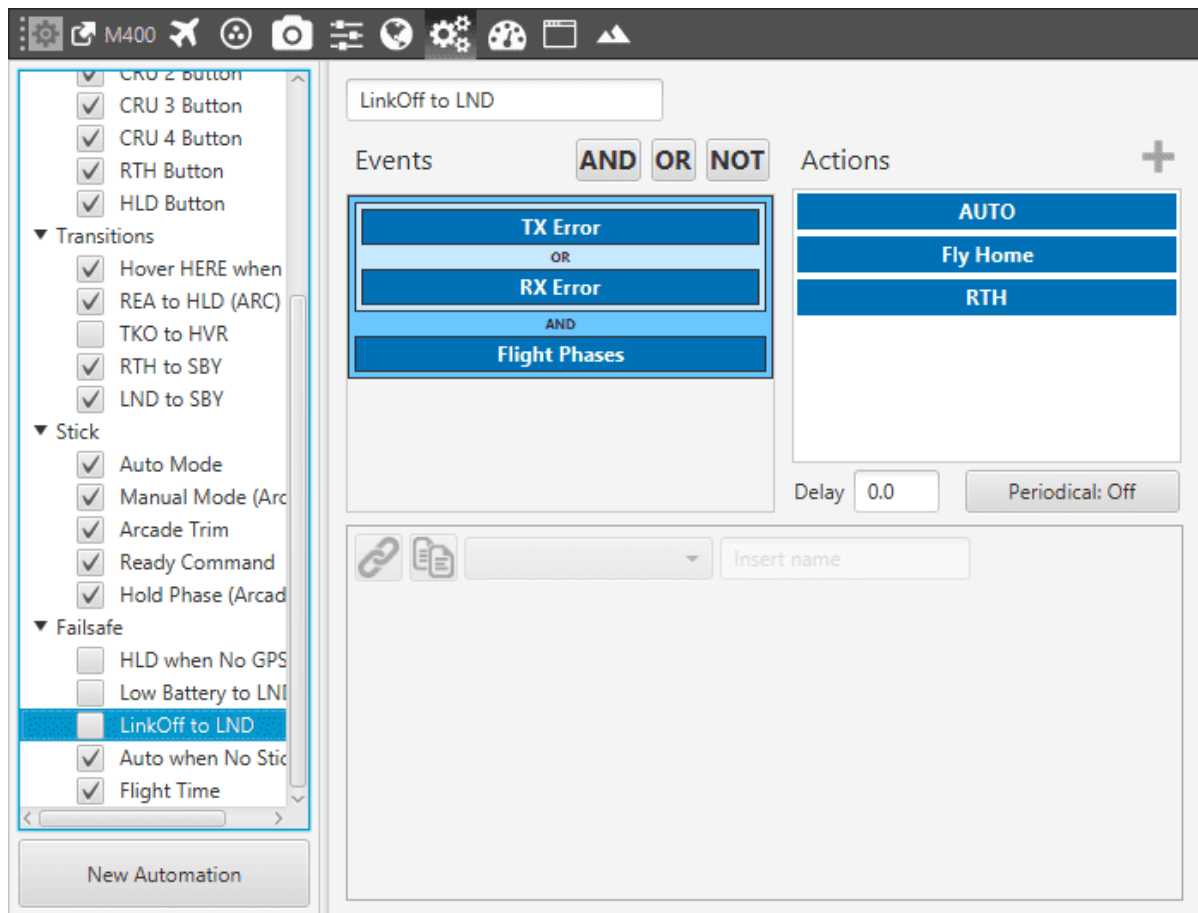


Failsafe for radio link-off – Plane

When the value of the flag that evaluates the TX or RX error (“TX/RX Packet Error Rate (on board)”) has a value of 1 (0.8 is established for safety), and the aircraft is in a certain phase (climb and cruise), it is forced to land.

Video Tutorial: <https://www.youtube.com/watch?v=mmzRw9V9OCs>

When using a multicopter the same process is done with some changes.



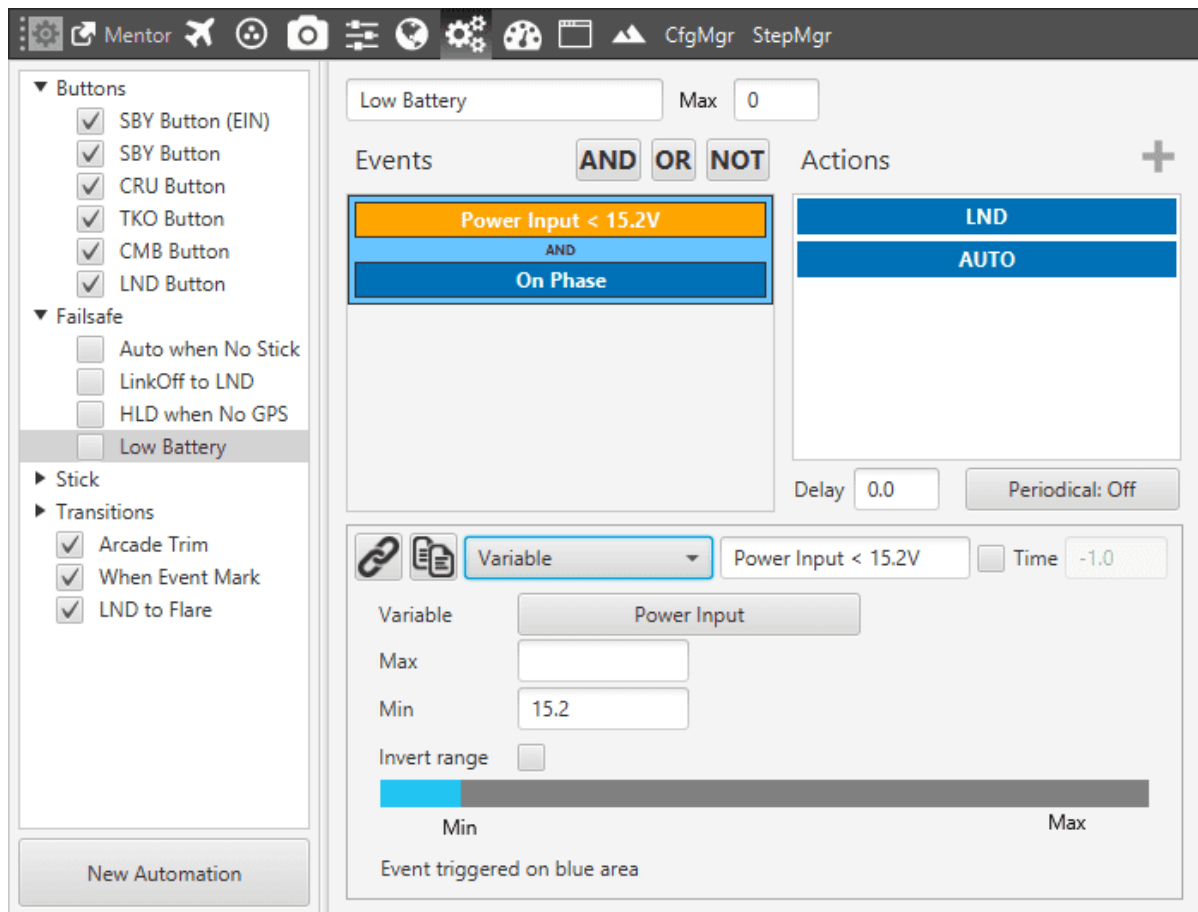
Failsafe for radio link-off – Multicopter

The actions carried out now are:

- **Mode:** select a change to AUTO mode.
- **Phase:** change to the phase Return to Home, in this phase is defined the Home point. See the example of this phase for the quadcopter M400 in section *Mission Phases*.
- **Go to:** as safety condition, this action is added and the Home point is selected.

12.3.3.2 Low Battery

When the battery is below a certain level (15.2 Volts in this case), and the aircraft is in climb or cruise, it is automatically commanded to land.

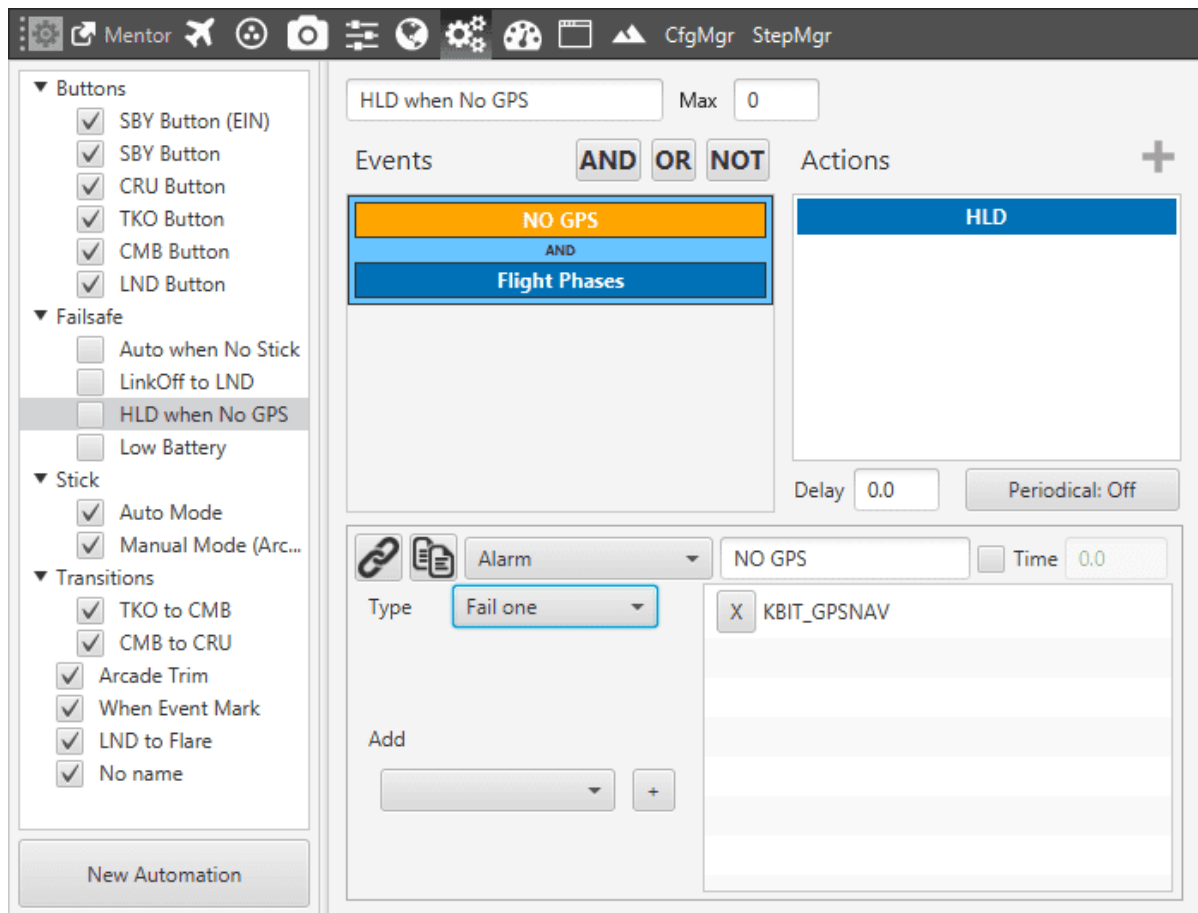


Failsafe for low battery

12.3.3.3 GPS Signal lost

When the UAV is not receiving the GPS signal, the system automatically adapts the navigation algorithm to avoid a possible accident by obtaining the aircraft attitude through the AHRS reference system. Besides that, it is also needed that the aircraft changes to a flight phase where the control is made over the attitude angles instead of position variables (heading, flight path). This phase is commonly known as HOLD, which has a guidance to keep pitch, roll and IAS at a certain value, and with PIDs in pitch and roll having two blocks for that angles and its rates.

The event of this fail safe automation will be the loss of GPS signal (type alarm), while the action is type phase (Hold).



Failsafe for GPS signal lost

12.3.4 Photogrammetry

Veronte Pipe incorporates the Photogrammetry mission, see section [Photogrammetry](#). It is possible to create it without using the tool incorporated within the mission toolbar.

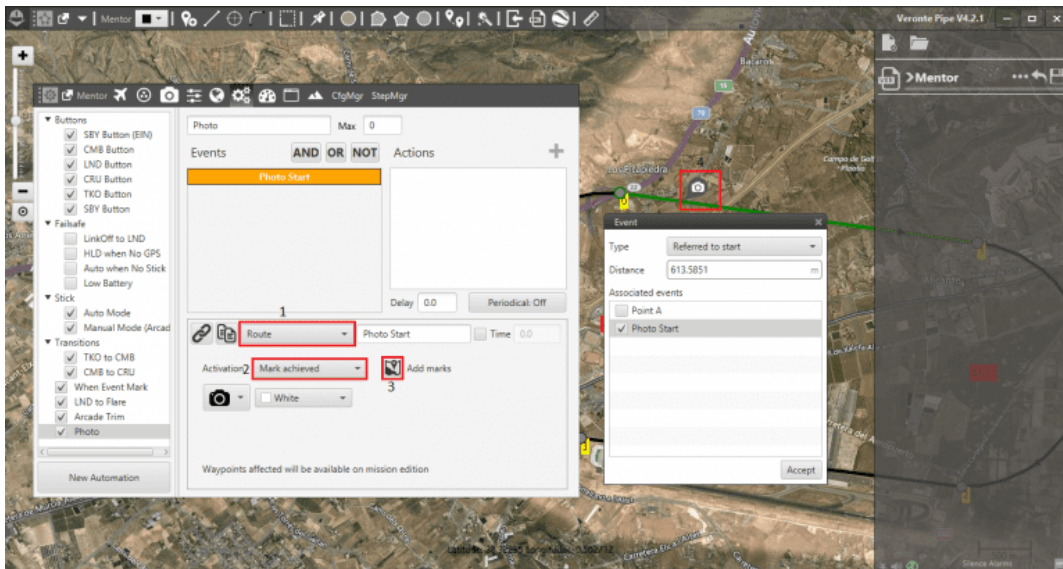
In the photogrammetry missions, the platform is wanted to take photos during the time that it is following a path previously defined. Commonly, the photo taking process begins at a certain location over the route and is repeated each certain distance. To create this automation the process detailed now has to be followed.

1. Create an **Event Marker** at the point where the photogrammetry mission has to start.



Photogrammetry Automation 1

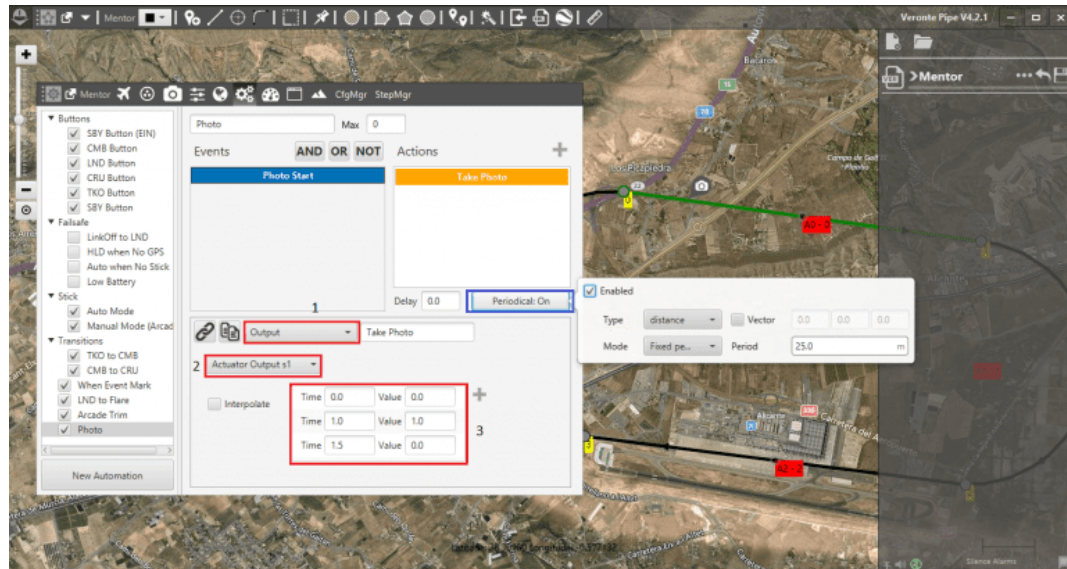
2. That **Event Marker** has then to be linked with an event that will trigger the process. Go to Automation in order to create it.



Photogrammetry Automation 2

The event is type **Route** (1) and will be triggered when the point is reached (2) (the other option is when flying towards the point). If the event is linked with a waypoint instead of an Event Marker, the point is selected in the map with button (3). Finally, with the event created in the automation menu, double-clicking on the event marker allows to link this automation to it.

3. The last step is to create the actions to be performed when the event is triggered.



Photogrammetry Automation 3

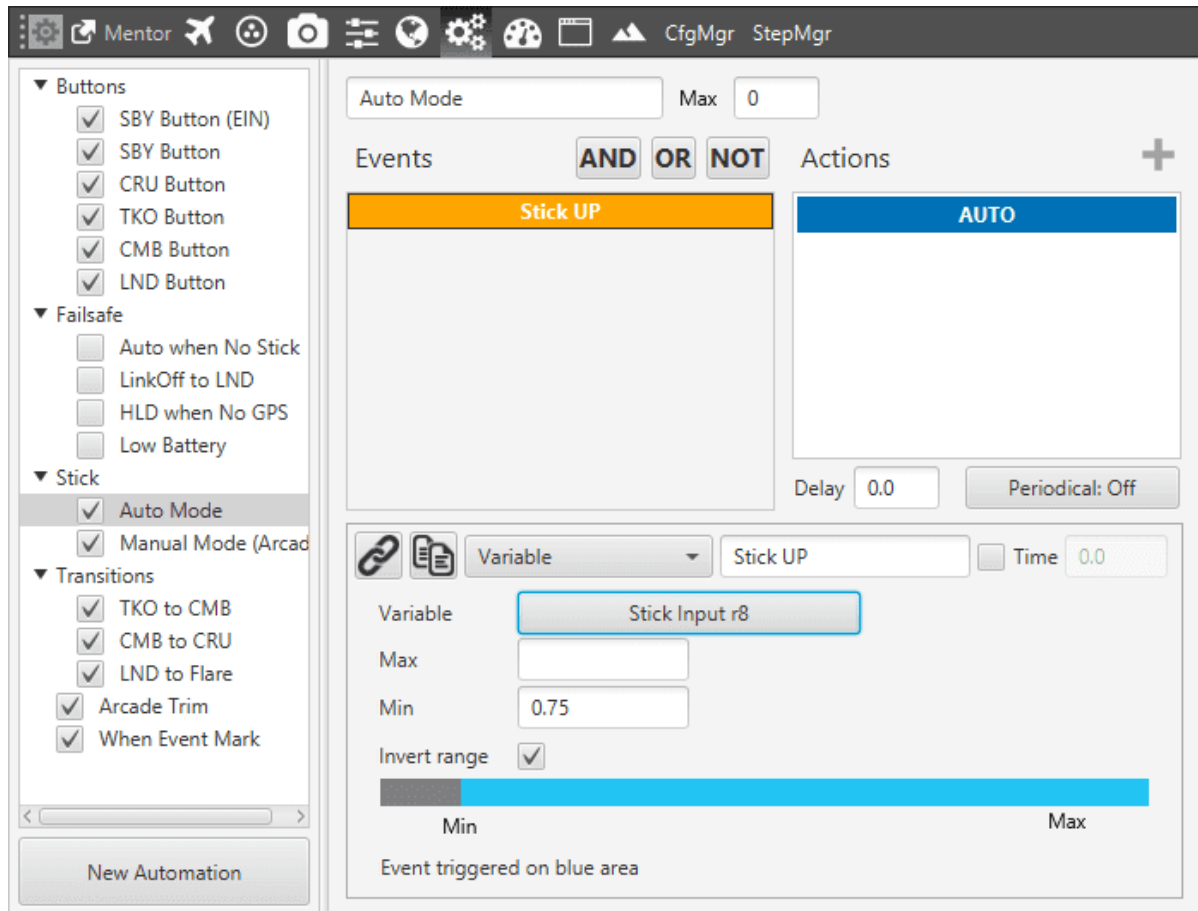
When the point is reached, a timer starts to measure the distance between the locations at which a photo will be taken. **Periodical** (blue box) has to be enabled, users have to indicate that the timer measures distance (the other option is time), and finally indicate that the distance measured is a fixed value (25 meters in this case).

The action type (1) is **Output**, which is used to send a signal trough one of the output pins of Veronte. In (2) is selected the actuator that is connected to the camera (or with the device that is controlled with this automation). Finally, in (3) are indicated the values of the PWM signal at each time instant.

12.3.5 Others

12.3.5.1 Stick Auto

This automation changes the control mode according to the command sent by the radio controller.



Stick Auto automation

In this case, the channel that controls the mode is the 8, so according to its value the mode is changed. With an event of type Variable the automation is defined. The process is the same for the manual mode, but when the r8 variable has a value lower than 0.25.

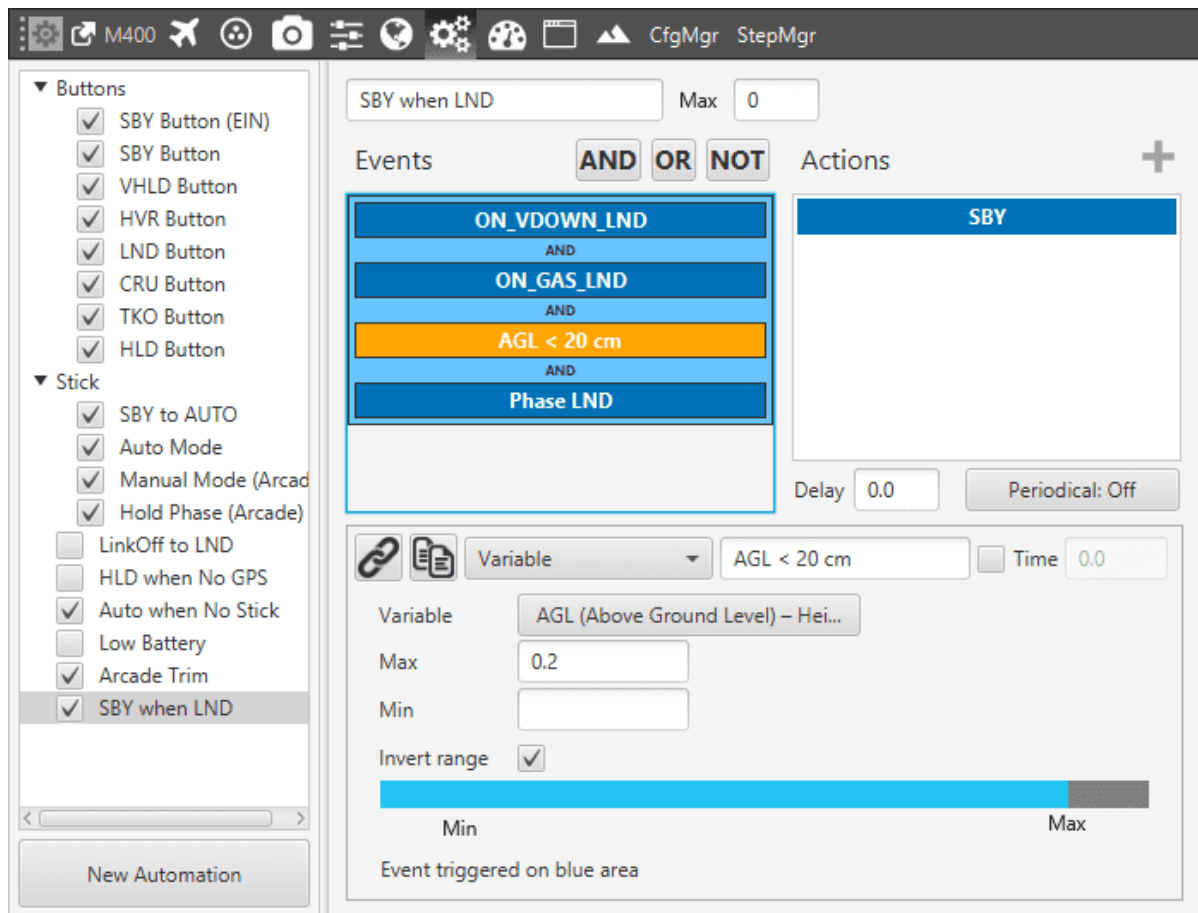
Video Tutorial: <https://www.youtube.com/watch?v=C3A6P1jgFV4&feature=youtu.be>

12.3.5.2 Automatic Landing (Multicopters)

If the user wants the multicopter stops the motors when landed (pass to Standby automatically), it is possible to configure an automation which performs this action.

Normally, to control the automatic landing, the following variables are used:

- Ground Speed Down
- AGL (Above ground level altitude)
- Thrust control
- Landing current phase



Landing auto configuration

Variable values must be chosen depending on multicopter (rotor number, dimensions, altitude sensors, etc...) and an integral control increase can be taken into account in order to perform a faster RPM descending and a better landing.

The automations are the mechanisms used to perform an action when some event is verified (using the Boolean Logic AND, OR and NOT tools). These actions could be a change from one phase to another, taking a photo, dropping a payload and so on. In this section, the following automation types will be detailed:

- Veronte Panel buttons
- Phase changing
- Failsafe (link-off, power low, GoHome point)
- Photogrammetry
- Others

12.4 Veronte Tracker

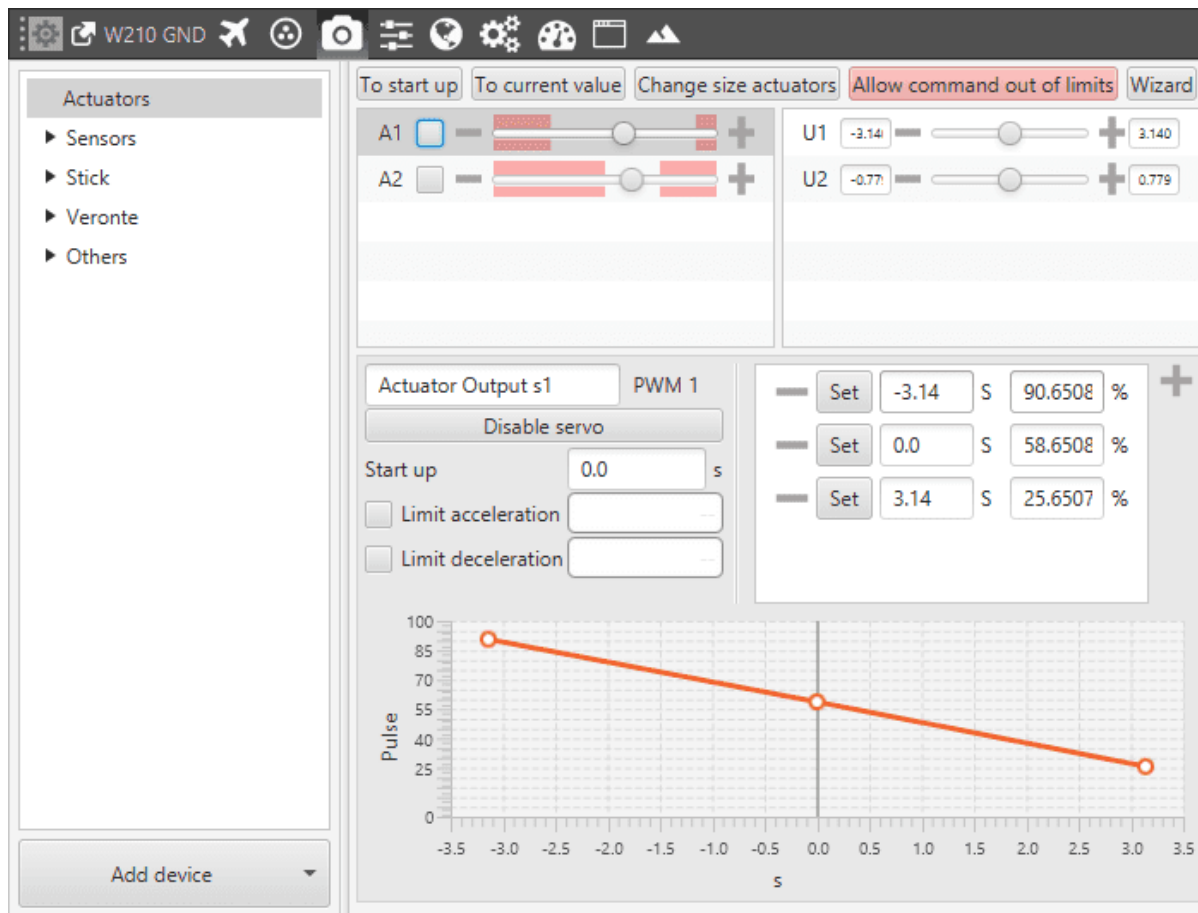
12.4.1 Veronte Ground Configuration

The first step of the configuration is the creation of the SU matrix including 2 controls (PAN-TILT) and 2 servos, the ones installed on Veronte Tracker in order to move Pan and Tilt axes.

The screenshot shows a software window titled "Edit" with a close button (X) in the top right corner. The window contains a configuration interface for an SU Matrix. At the top, there are two input fields, both containing a hyphen (-). Below these, the labels "Control Output u1 - Pan G1" and "Control Output u2 - Tilt G1" are visible. The main area features two rows of actuator outputs. The first row is labeled "Actuator Output s1" and has input fields with values "1.0" and "0.0". The second row is labeled "Actuator Output s2" and has input fields with values "0.0" and "1.0". To the left of each row is a small button with a minus sign (-). To the right of the input fields is a vertical slider with a plus sign (+) at the top. At the bottom of the window, there is a large button with a plus sign (+) and an "Apply" button in the bottom right corner.

SU Matrix

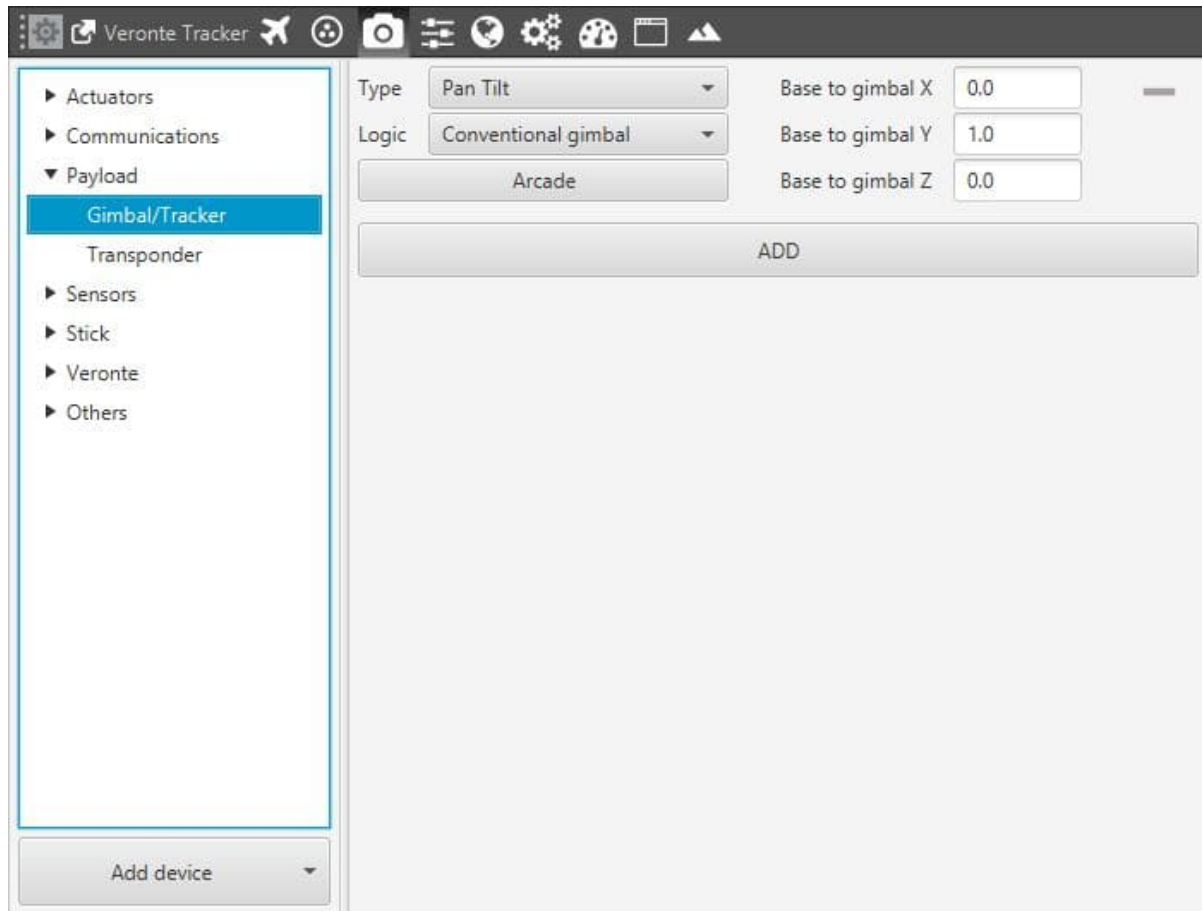
Once the SU matrix is complete, it is possible trimming both servos with the same procedure of aircraft servos in the **Devices > Actuators panel**. The user must set the center of the antenna and then maximum and minimum angles. The procedure has to be repeated for both servos.



Servos Trimming

When servos are trimmed, the tracker has to be configured in the Devices >Payload panel as:

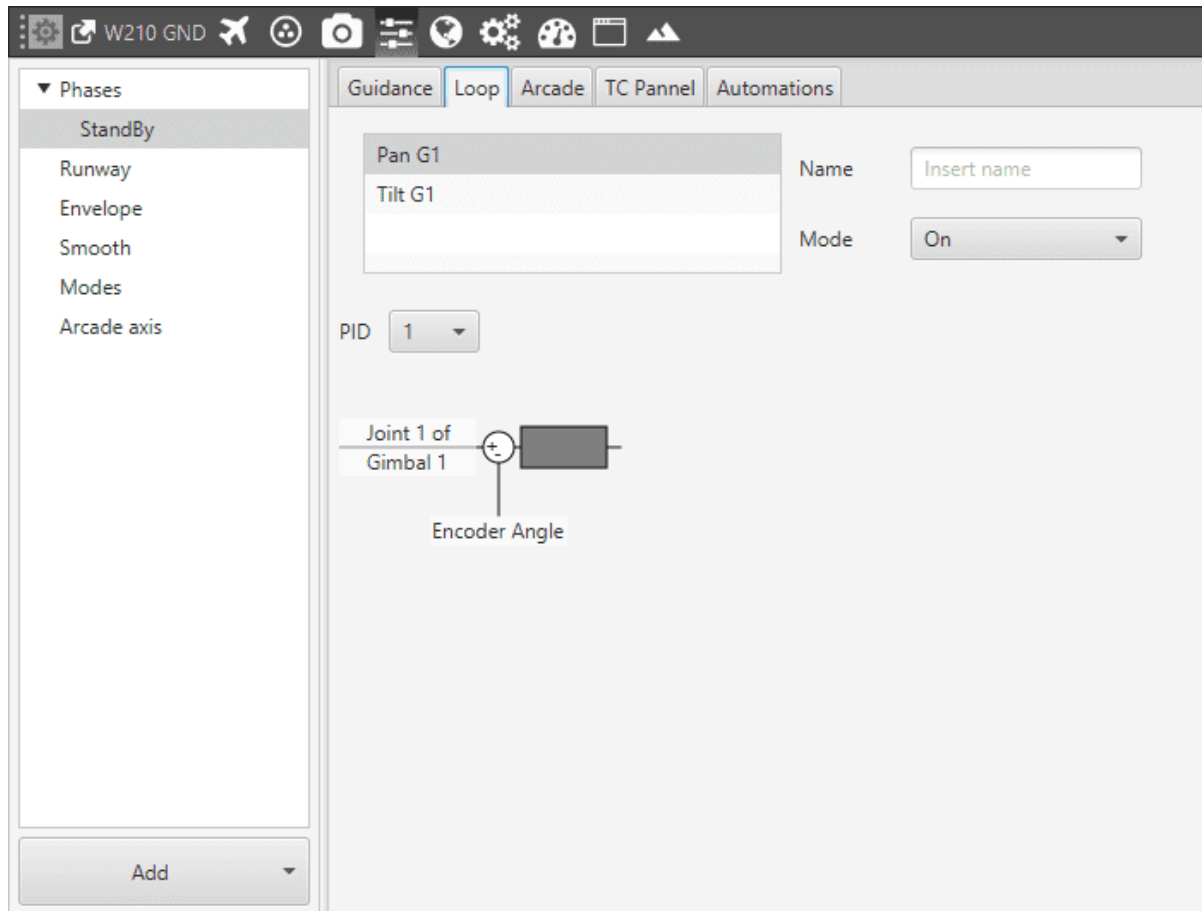
- **Type:** Pan Tilt.
- **Logic:** Conventional Gimbal, with Horizontal (Pan) and vertical (Tilt) axes.



Tracker Payload configuration

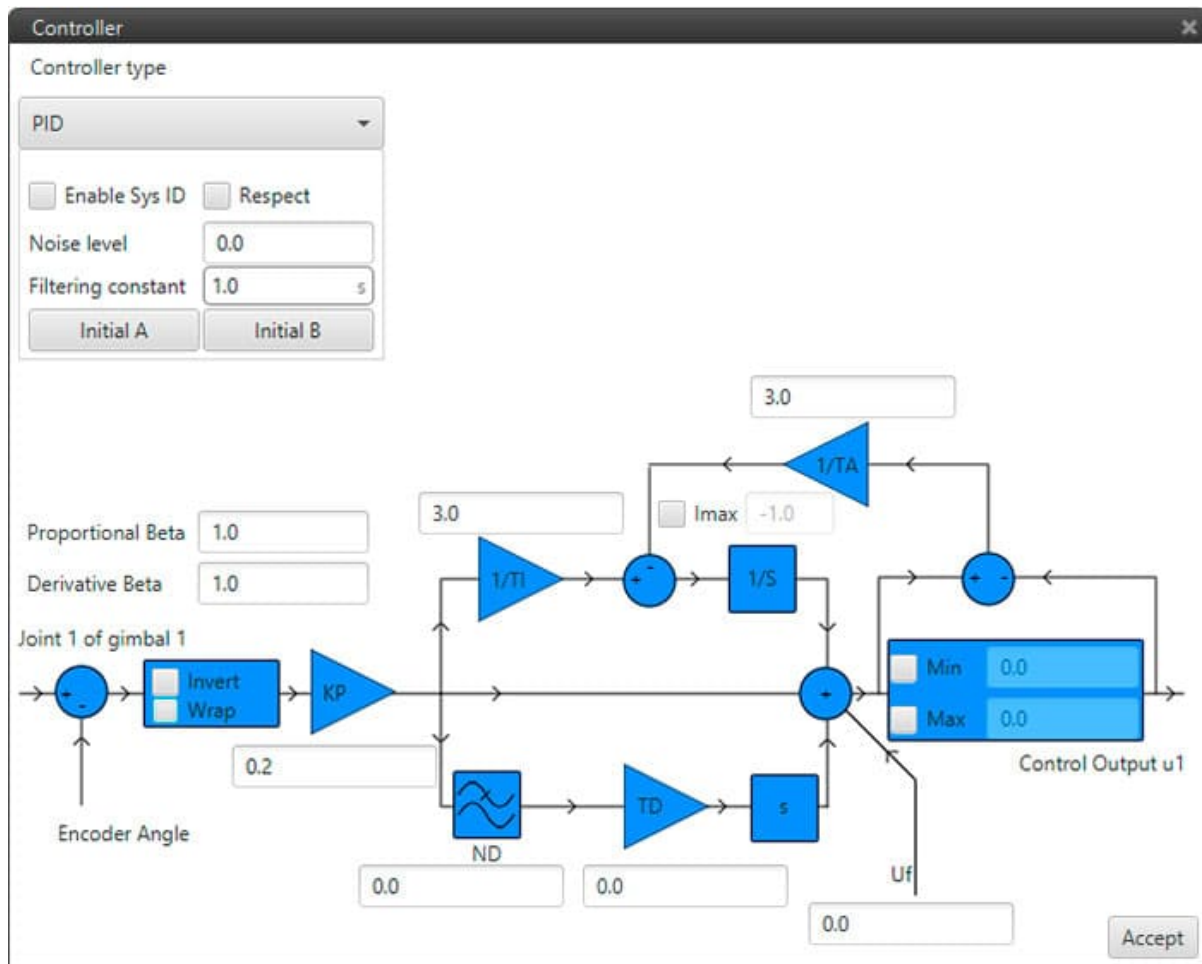
Next step is the configuration of the control loops for each axe: Pan and Tilt.

- Pan control loop is a closed loop with Joint 1 of gimbal 1 entry variable (the desired Pan angle) and the Encoder Angle variable in the feedback loop. The output is configured as the SU matrix: Control Output u1.



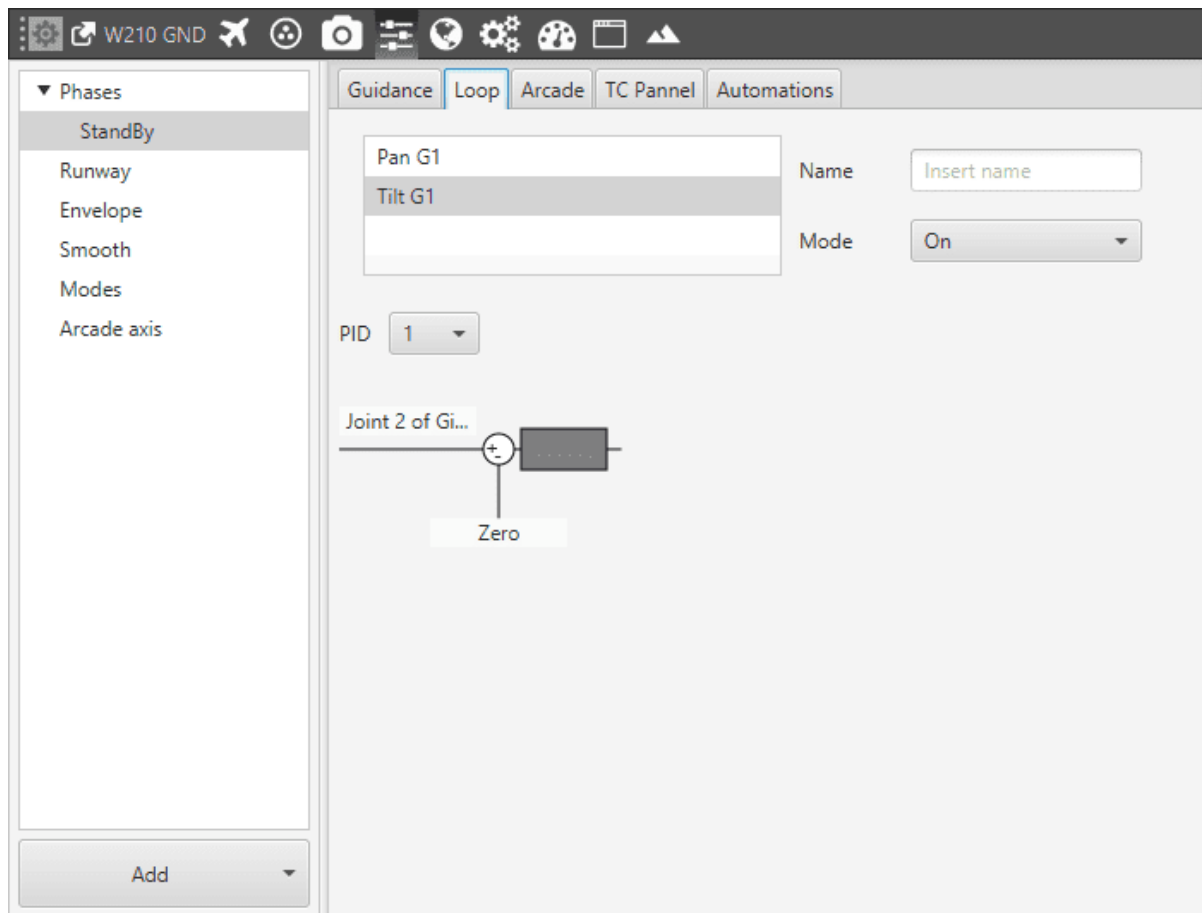
Pan Loop Configuration

A typical setting for the loop is presented in the following figure. The proportional gain is set at 0.2 and the integral gain at 3.0.

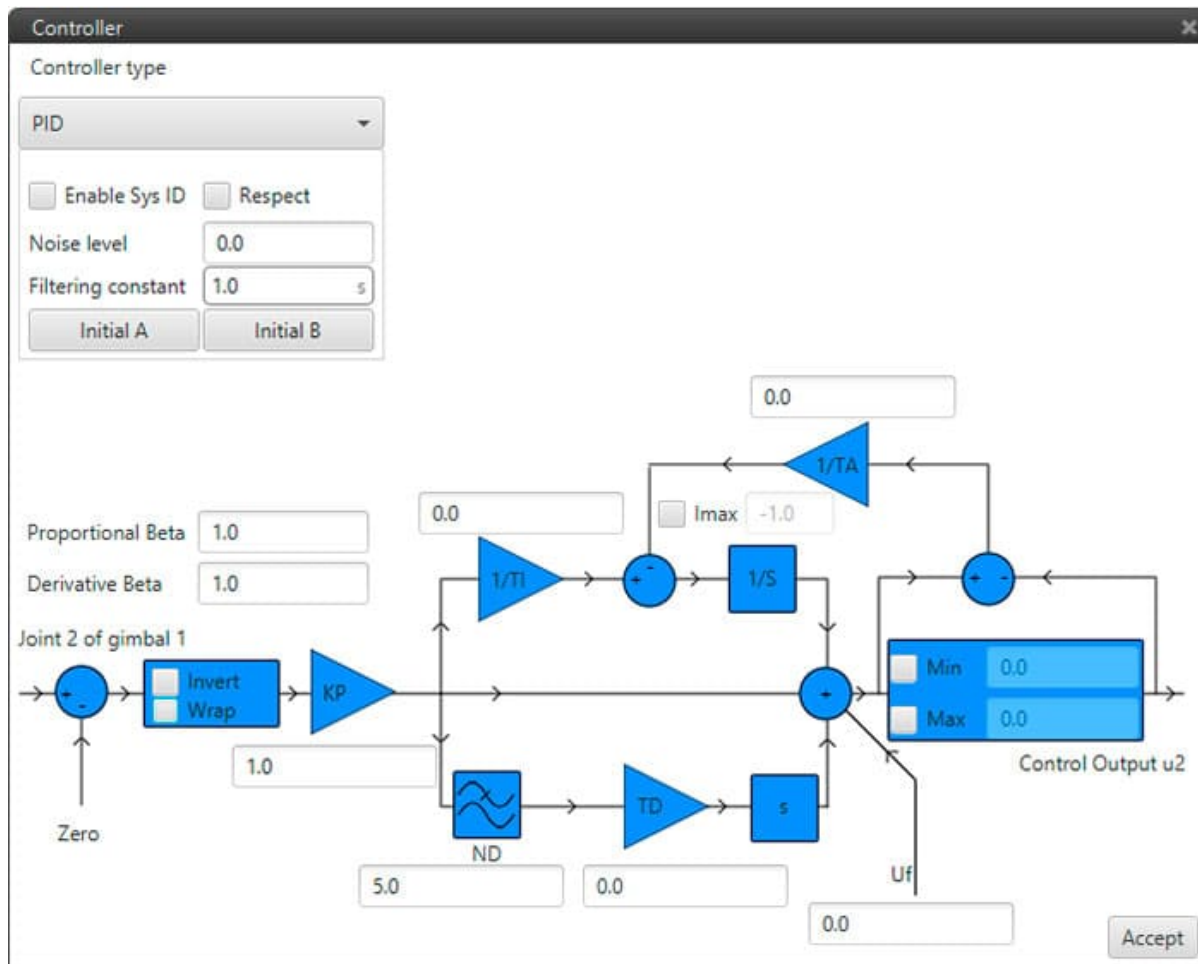


Pan Control Loop

- For the Tilt control, an open control loop is sufficient. **Joint 2 of gimbal 1** is the entry variable (the desired Tilt angle) and the **Zero** variable is configured for the feedback loop. For those cases where an **encoder** is present it should replace **Zero** as feedback for Tilt angle. The output is configured as the SU matrix: **Control Output u2**.

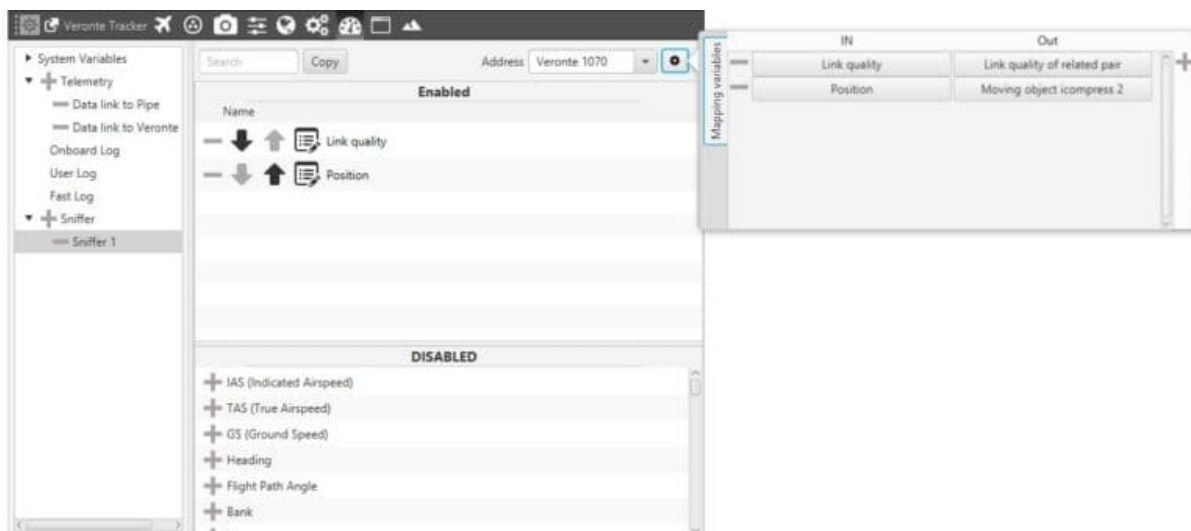


Tilt Loop Configuration



Tilt Control Loop

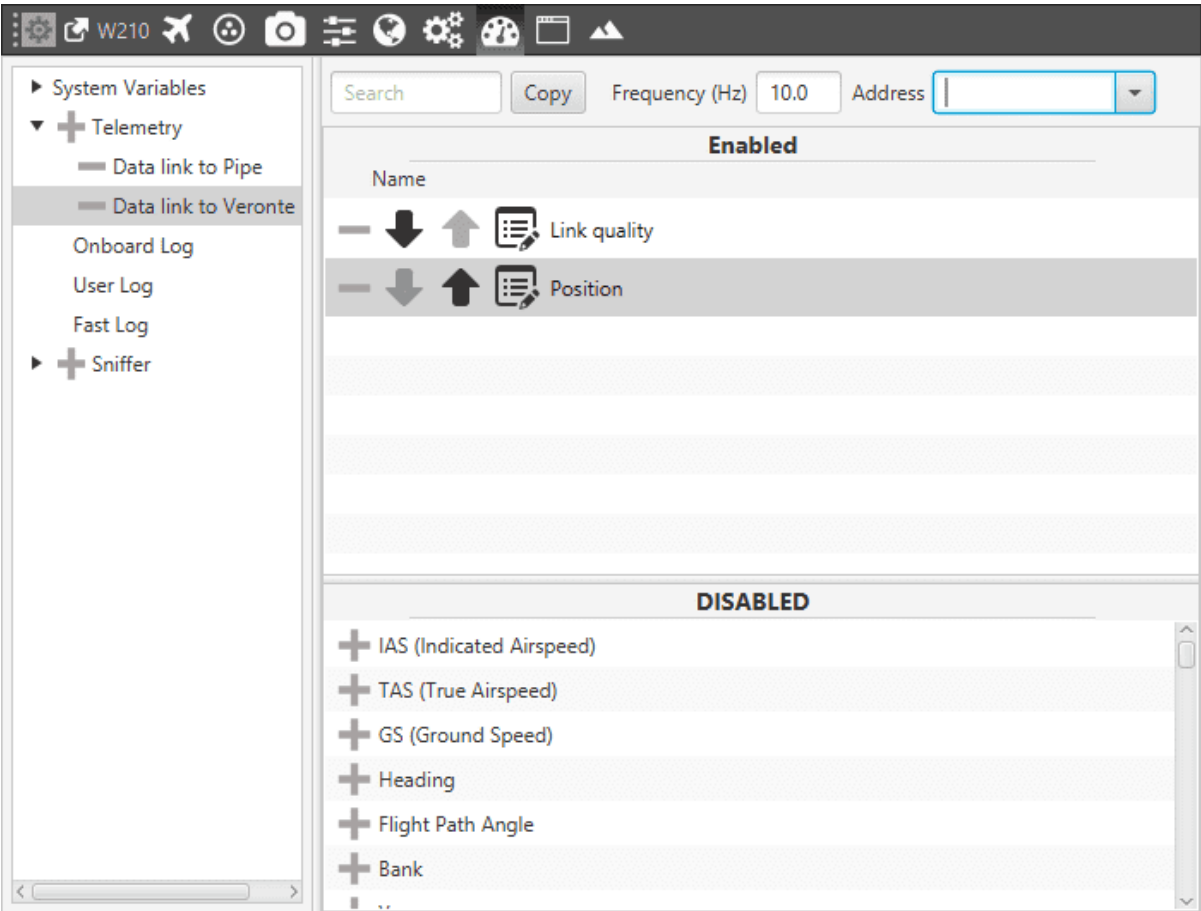
The last step of the Veronte Ground configuration is the Sniffer setting. Configuring a new Sniffer means allowing the Ground Station “knowing” the Veronte Air (UAV Address) position during flight and associating it with a Moving object. Veronte Air has to be configured to send its position to Veronte Ground. On **Address** it has to be selected the Veronte Air to be tracked.



Sniffer Ground Configuration

12.4.2 Veronte Air Configuration

Configuration of Veronte Air includes only one step which is the Data Link configuration. The new Data Link has to be defined as showed in the following image: the variable to send is the Position and **Address** has to be set according Veronte Ground (Unit that will receive this data) with a frequency of 10.0 Hz.



Data Link Veronte Air Configuration

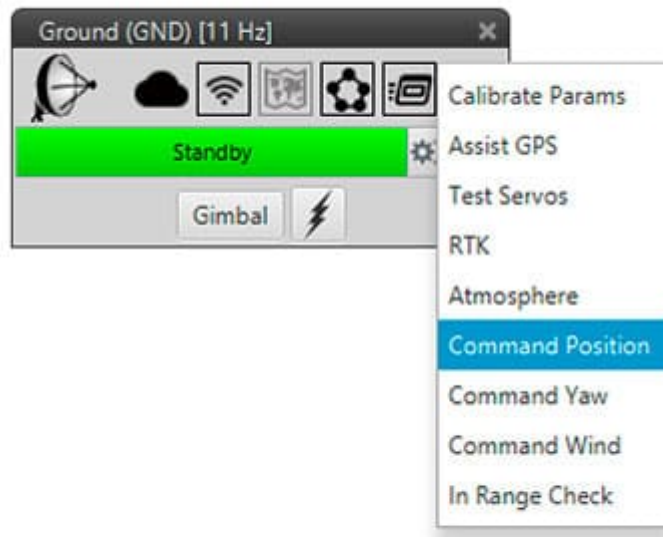
12.4.3 Tracking Operation



Tracking Operation

In order to activate the tracking, the user must follow some steps.

1. First of all, it is necessary to configure the Tracker (Veronte Ground) position on the map. To do this it is sufficient to go in the **Veronte Ground Panel**, open the **Run Task** and then the **Command Position** window.

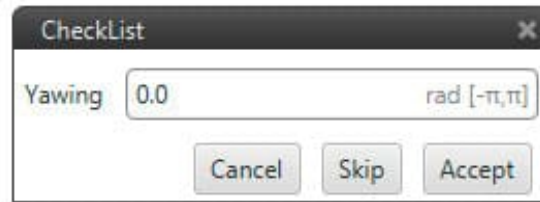


 The image shows a "CheckList" dialog box. It has two radio buttons: "Absolute" (selected) and "Relative". Below these are input fields for "Latitude" (0.6699061267967821 rad $[-\pi, \pi]$) and "Longitude" (-0.012143301342789 rad $[-\pi, \pi]$). There are also two map icons. Below these are fields for "WGS84" (316.7396599671141 m), "MSL" (266.2 m), and "AGL" (1.20 m). At the bottom are "Cancel", "Skip", and "Accept" buttons.

Tracker Position configuration

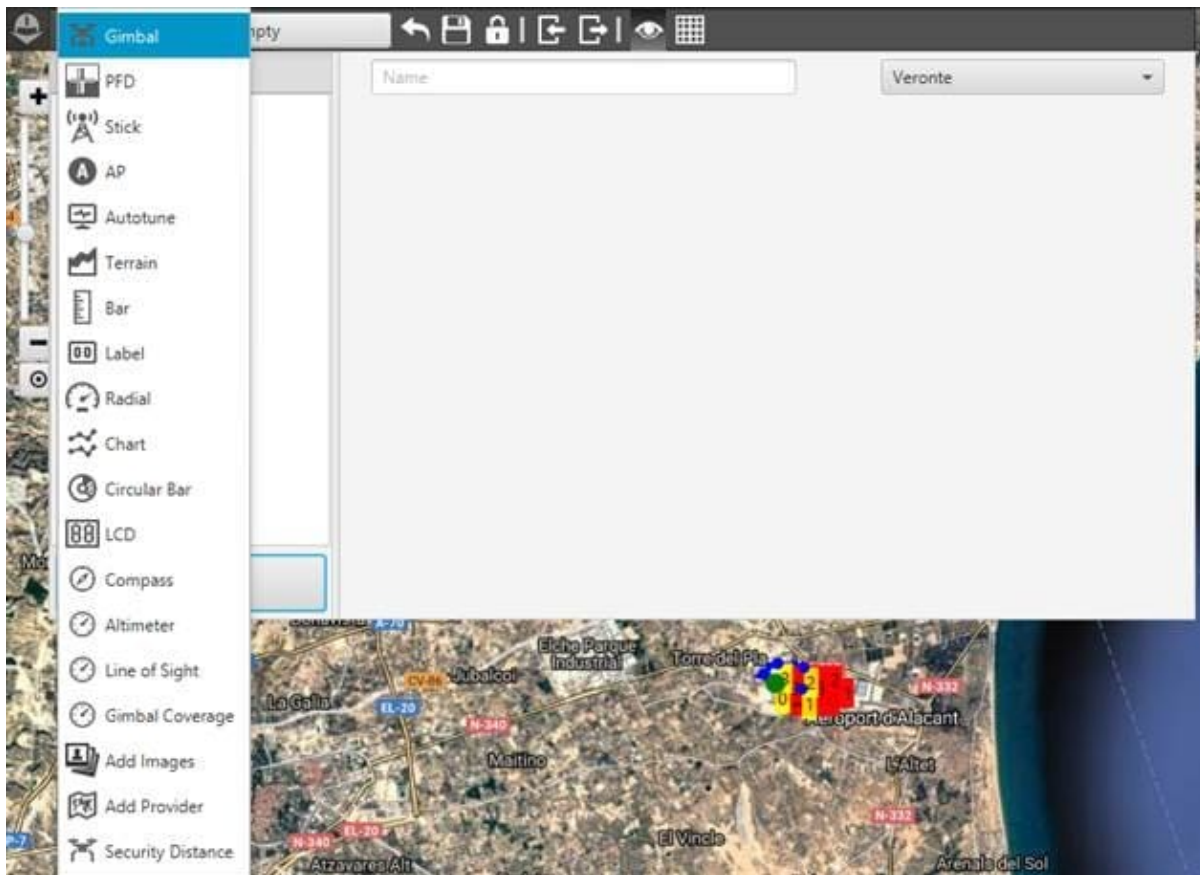
In this panel, it is necessary to insert the tracker position. The user can perform this action inserting Latitude and Longitude values manually or directly clicking on the Map location. The tracker altitude has to be defined too: normally, the tracker tripod is set at 1.20/1.40 [m] so this value has to be inserted in the AGL altitude.

2. Introduce **Yaw initial orientation** of the tracker. Go into **Veronte Ground Panel**, open the **Run Task** and then select **Command Yaw**.



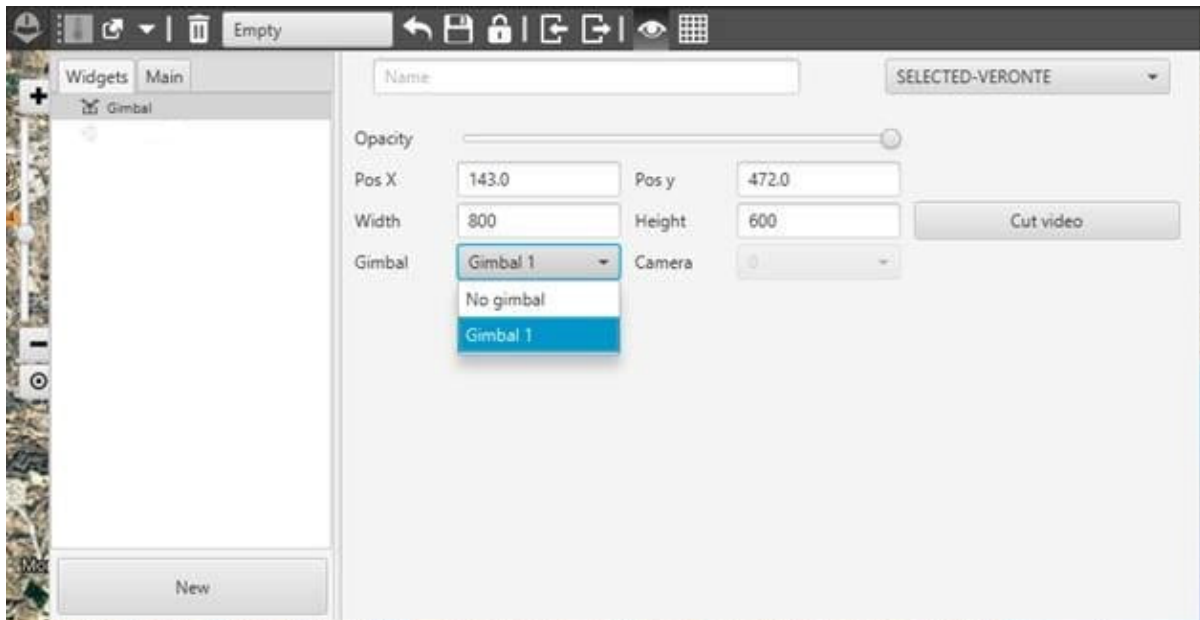
Tracker Orientation configuration

When Veronte Tracker is completely positioned, it is possible to activate the tracking. To do this it is necessary to add a Gimbal widget on the Workspace.



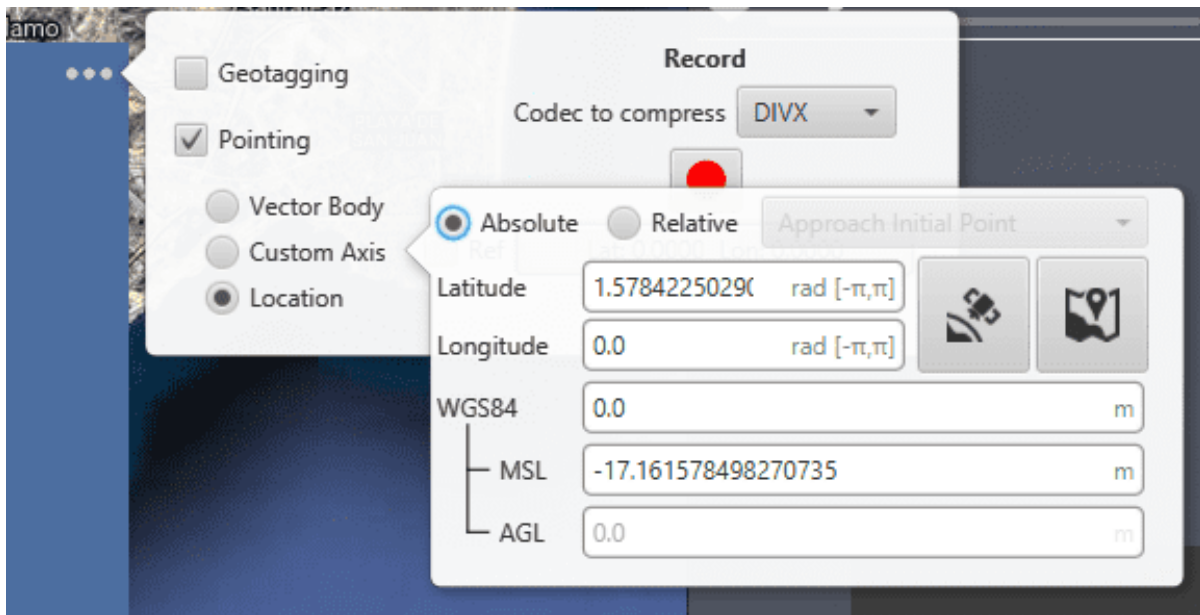
Workspace: Gimbal

Select the Payload as the one in the Veronte Ground Devices configuration.



Workspace: Gimbal widget

After doing it click on '...' placed on the top right corner of the widget Gimbal.



Gimbal widget: Pointing

In this window it is possible to set where the tracker will point. It can be a absolute position, like the one from the image, or a **Relative position**. For example the **Moving object incompress 2** (Veronte Air in this case).

It is possible to test the correct tracking by moving the Veronte Air (or the platform if it is possible) and checking the tracker correspondent rotation in Pan and Tilt. To perform this test it is useful to maintain the UAV at > 20 [m] from the Ground Station because of the bad transmission link in case of small distances.



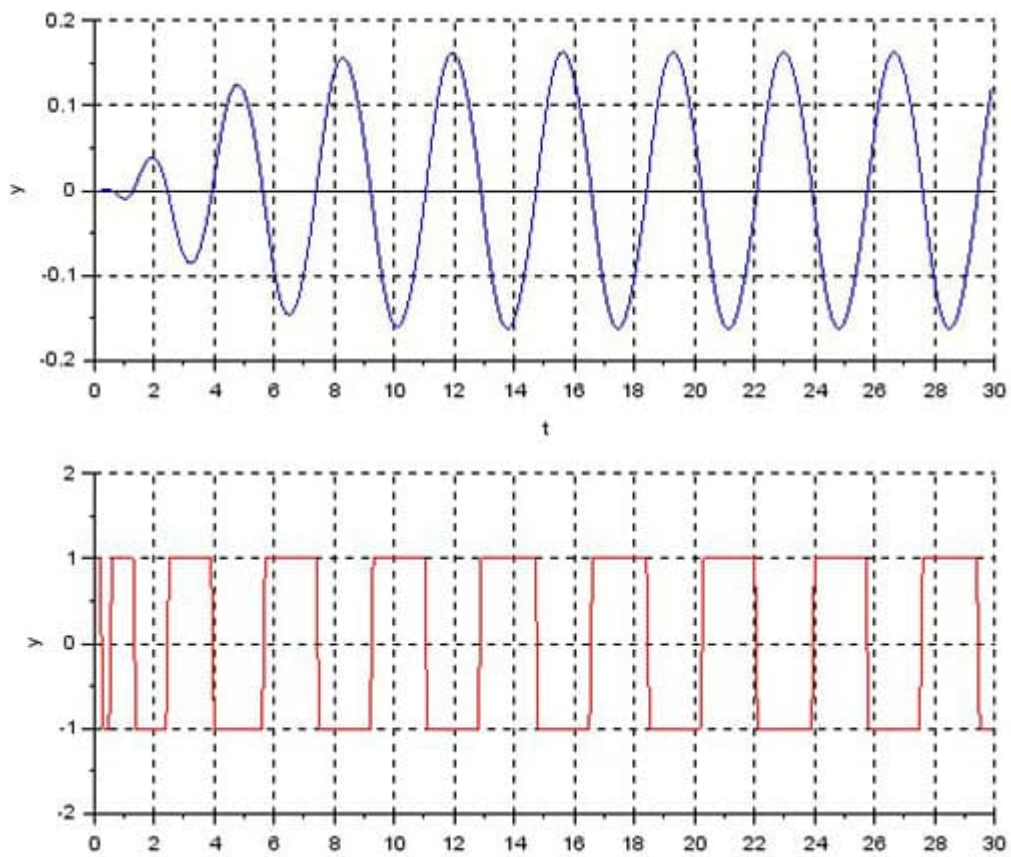
Veronte Tracker

Veronte Tracker is a high performance tracking antenna specifically designed for most demanding applications. The system can install any directional antenna for maximizing system operation capabilities. Embedded control actuators and installed encoders permit to automatically point the antenna with unique precision. Height (Tilt control) and orientation (Pan control) given to the antenna makes the device perfect for long range operations.

Video: [Tracking Antenna – Veronte Tracker](#)

12.5 Autotune

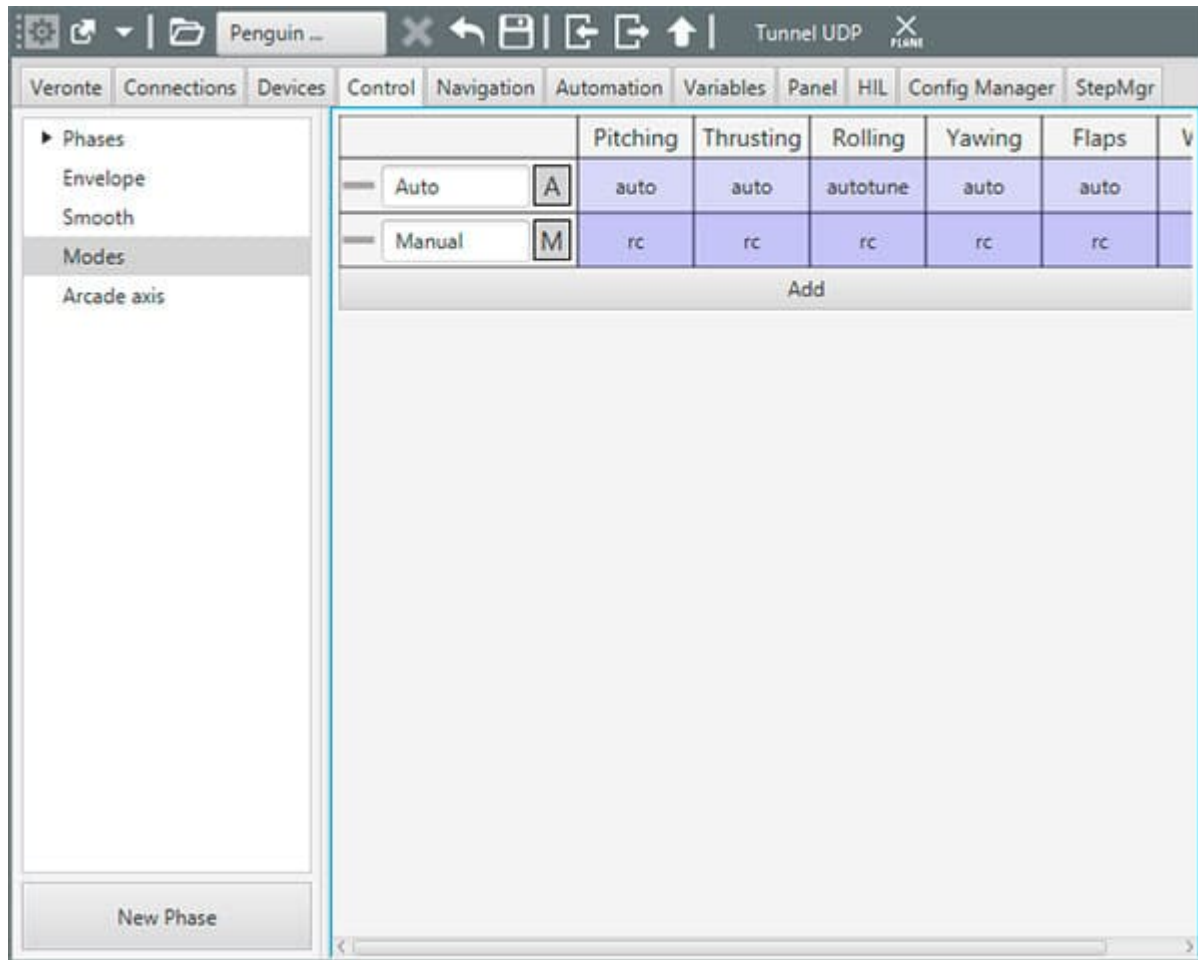
The Autotune allows finding the PID controller gains automatically. During the process, the PID controller is replaced with a relay function and the controller parameters are then determined from the period and the amplitude of the oscillation of the system by using FFTs.



Autotune output (blue) and input (red)

12.5.1 Modes Configuration

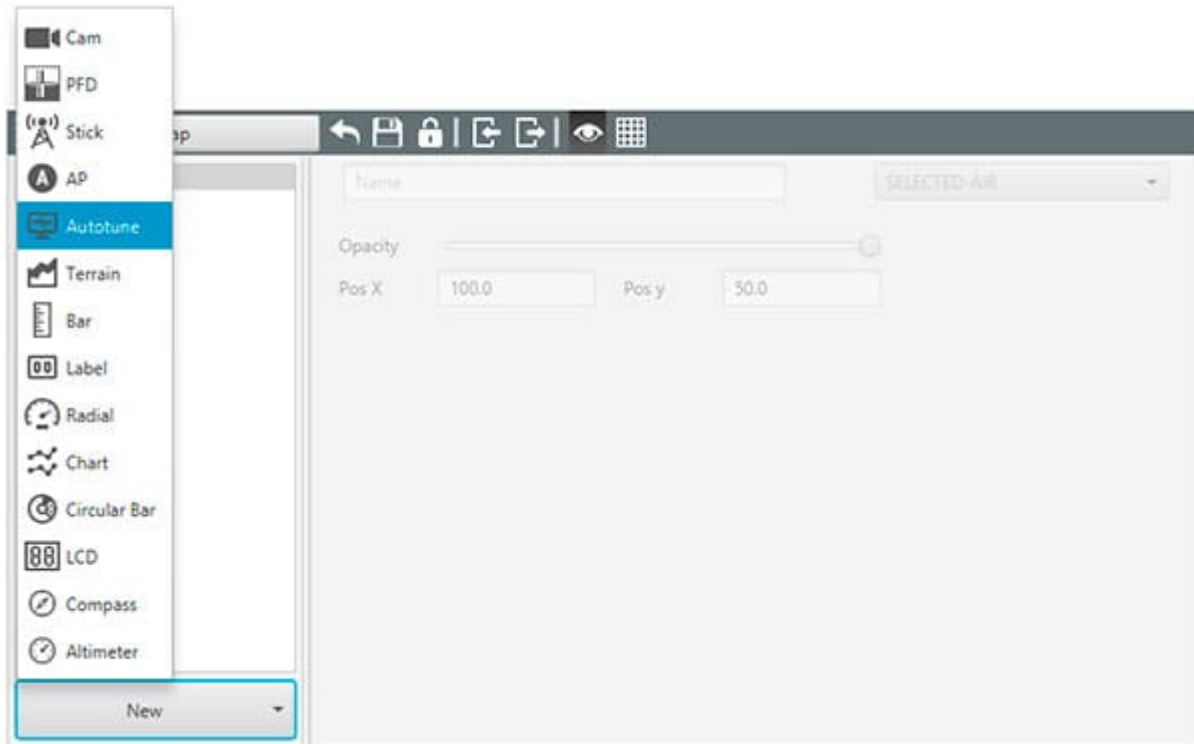
To be able to use the autotuning tool, the Autotune Mode has to be selected in the Modes menu for the control output which must be autotuned. In the following image, for example, for the rolling control is set the Autotune mode.



Autotune mode

12.5.2 Workspace Configuration

In the Workspace panel, it is possible to select the Autotune Tool.



Autotune tool in the Workspace

When the window is opened, it is possible to select the loop for the autotuning. If some control loop does not have a PID controller configured, it appears with transparency and it can not be autotuned (red).



Autotune loop selecting

When the loop is selected, the window changes and some parameters are showed and they can be edited:

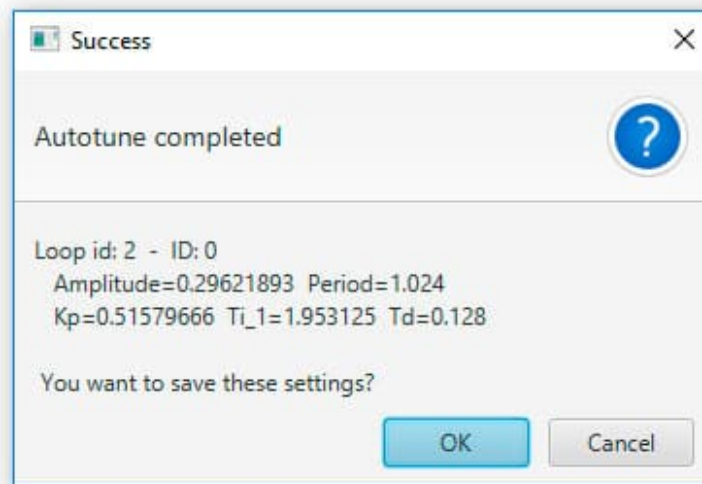
- **Time:** It is the period of time in which the Autotune is performed [s].
- **Stages:** The number of stages of the Fast Fourier Transform (a value between 5 and 10 is allowed).
- **Relay:** This is the amplitude of the Relay function (R). The value has to be chosen in according with the proportional gain in the PID of the autotuned variable.
- **Respect:** A mean value of the variable has to be selected. If the respect is select, the autotune will start from the last value of the variable. If not, the value can be edited by the user and the Autotune Relay function will start from this value and it will go from -R to R.

When all parameters are set, it is possible to click on Start and the autotuning process will begin and the blue bar will move until the end of the process. The left bar of the window allows to check the flight Phase and the selected loop.



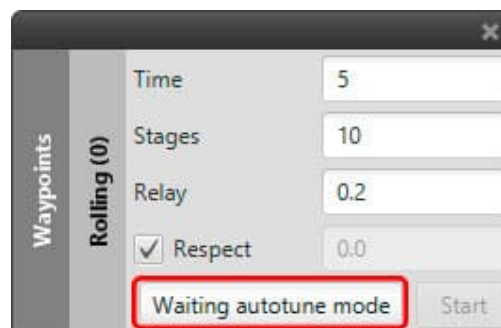
Autotune process

If the process is performed successfully, a new window will show up showing the PID gains found in the Autotune process and the output wave amplitude and period. From this window, it is possible to save the new values in the configuration or cancel them.



Autotune completed

Warning: Always be sure the gain values are in the correct range. A bad value, if saved in the configuration, could cause a control loss of the platform.



Process error

When the Autotune is started, it is possible to see a message in the blue bar: **“Waiting for autotune mode”**. In this

case, the autotune mode for the selected control is not correctly set. The user must change it in the control panel.

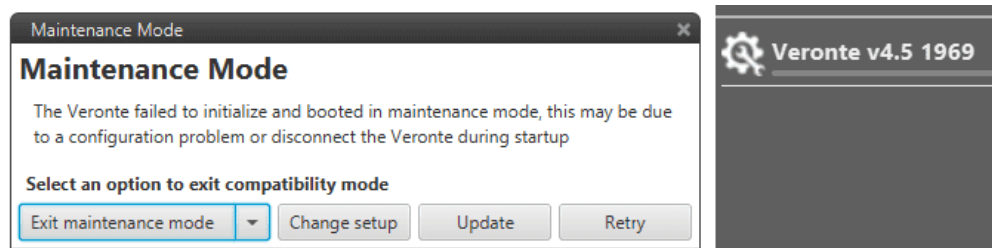
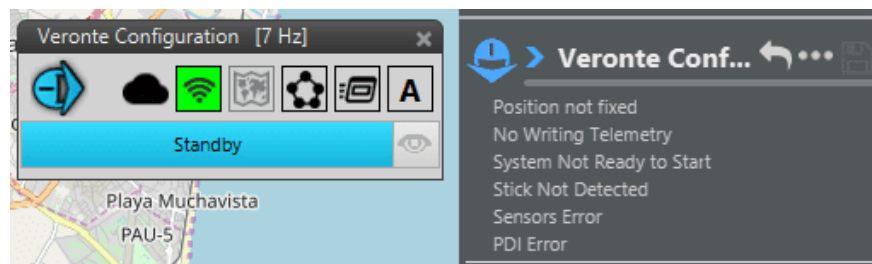
In this section, a series of different examples will be presented in order to improve the understanding that the final user will have of Veronte Pipe.

The different topics cover in this section are:

- **Configurations:** it will be explained how to configure some different platforms in Veronte Pipe, starting with the servo trimming process and the creation of the mission phases of a simple flight operation.
- **Automations:** this section will be focused on the creation of the common automations that are used for operating a UAV. Presenting this ones will make it easier for the user to develop its own automations for more specific operations.
- **Autotune:** the complete example of the autotune configuration and operation.
- **4G Communication:** Steps to configure 4G communication with Veronte Autopilots.

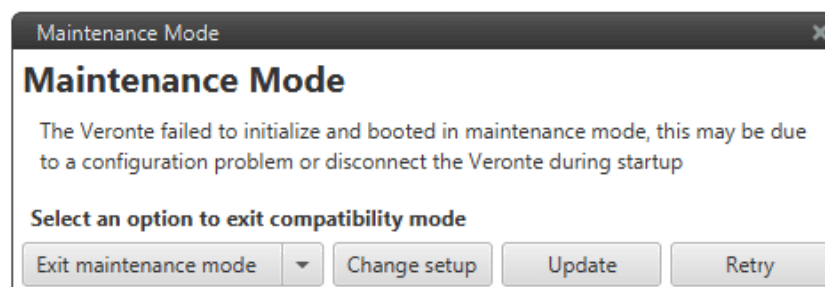
TROUBLESHOOTING

In the following section there are examples of issues that can be found while integrating **Veronte Autopilot** and how to troubleshoot them



Troubleshooting Veronte

13.1 Maintenance mode



Maintenance mode tab

Maintenance mode is the main troubleshooting tool that **Veronte** puts at the user disposal. While in **Maintenance mode**, all communication channels are enabled by default, so it is possible to connect with **Veronte** through any of its configuration interfaces no matter its current configuration.

While in **Maintenance mode**, it is possible to perform actions such as force the load of a **new configuration** file or **format the SD card**.

The main use of maintenance mode is to solve issues related to the current configuration, mainly related with communication or memory writting issues.

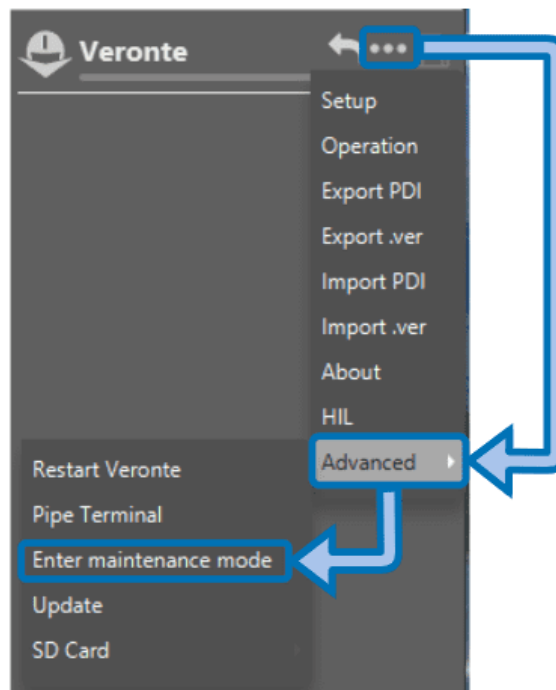
If at some point the communication with **Veronte** is lost, it is possible to use **Maintenance mode** to go back to a previous state of the configuration (as long as it was exported previously), **format the SD card** to start over or update the unit's **firmware**.

Tip: If you need to recover a **recent configuration**, but it has been some time since the configuration was **manually exported**, try using the *Post Flight* tool to recover your most recent configuration.

Important: It is heavily recommended to always use **Maintenance mode** to load a new configuration that is very different from the current one.

13.1.1 Entering maintenance mode

Maintenance mode can be entered by clicking the corresponding command, which can be found in Options -> Advaced -> Enter **Maintenance mode**



Maintenance mode access

13.1.2 Forcing maintenance mode

In the situation that communication is lost with the unit, it is also possible to activate **maintenance mode** using the power input.

In order to activate **Maintenance mode**, power cycle the unit **repetitively**, with a **period of 1 second**.

Maintenance mode will be entered after **30 cycles** are performed.

It is also possible that Veronte enters **Maintenance mode** if a problem with the power source (out of voltage/current range) is detected upon boot up.

Tip: Forcing a **Veronte PCS** unit into **maintenance mode** can be more difficult than average. For each power cycle, make sure that the Power LED is turned off before turning the unit on again.

13.1.3 Maintenance mode options

The following options are available while in **Maintenance mode**.

13.1.3.1 Exit maintenance mode

Exit **Maintenance mode** and **restart** the unit.

13.1.3.2 Remove Encryption

Remove **encryption** from the unit.

13.1.3.3 Change setup

Overwrite the **current configuration** with a new set of files. This option will force the load of the files, now matter if there are **PDI errors** on the new files.

13.1.3.4 Update

Update the unit's **firmware**.

13.1.3.5 Retry

Retry to boot **Veronte** system.

13.1.4 SD Loaded With errors

If upon booting, errors are found on **Veronte** configuration (be it an incorrect configuration or the absence of it), the boot will fail and **Veronte** will enter the '**SD loaded with errors**' mode.

Veronte will enter this mode if it detects an '**Illegal**' setting within the configuration, that could potentially affect the normal behaviour of **Veronte**. In order to escape **SD loaded with errors** mode, load an **Older** or **Default** configuration. If the configuration problem is **known**, it is also possible to use the **Try load PDI** feature to solve the issue directly.

This mode is very similar to **Maintenance mode**, but some of the troubleshooting actions are different



SD Loaded with errors tab

13.1.4.1 Check PDI

Re-attempt the PDI files check.

13.1.4.2 Try load PDI

Pipe will try to download the configuration from the unit. If the download succeeds, the configuration will be displayed. If the errors in the configuration are found and fixed, it is possible to save the changes and reboot the unit to exit **SD loaded with errors** mode.

13.1.5 Communications failure

This is not an actual **Maintenance mode**.

If this warning is displayed, it means that **Veronte Pipe** detected a **Veronte** unit, but was not able to stablish communication with it.

The only option available is attempting the connection again.

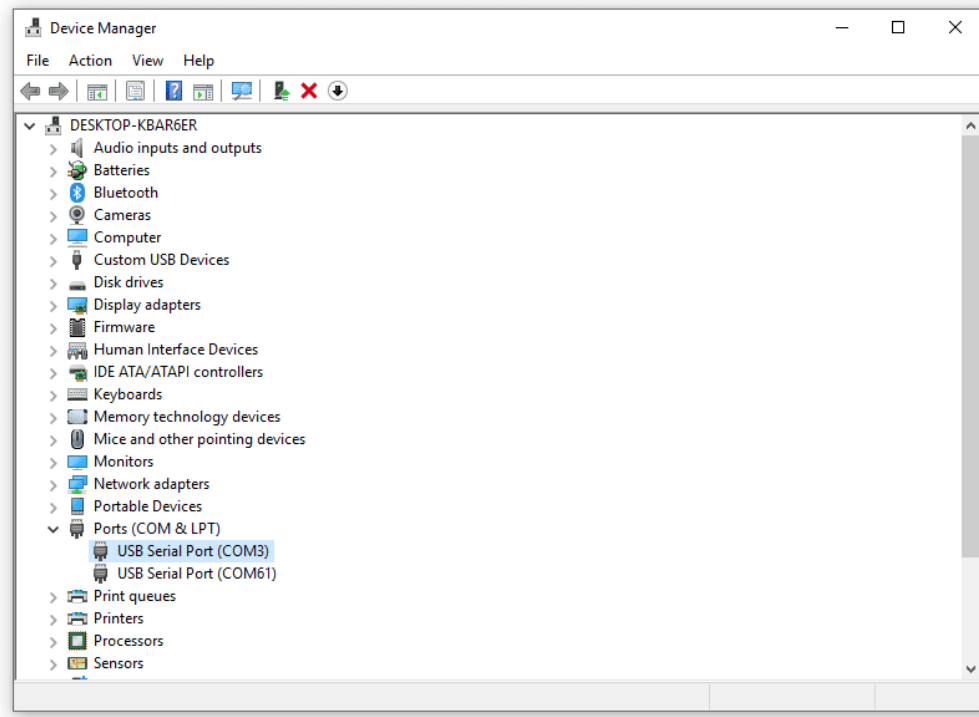
If unsuccessful, please review that all communication channels are correct and try again.

13.2 Veronte can't be detected through USB port

13.2.1 Check the COM port

If your **Veronte** unit cannot be detected through the USB port, please check if the PC is able to correctly detect the **COM port**.

On **Windows OS** systems this is done in the **Device Manager**:

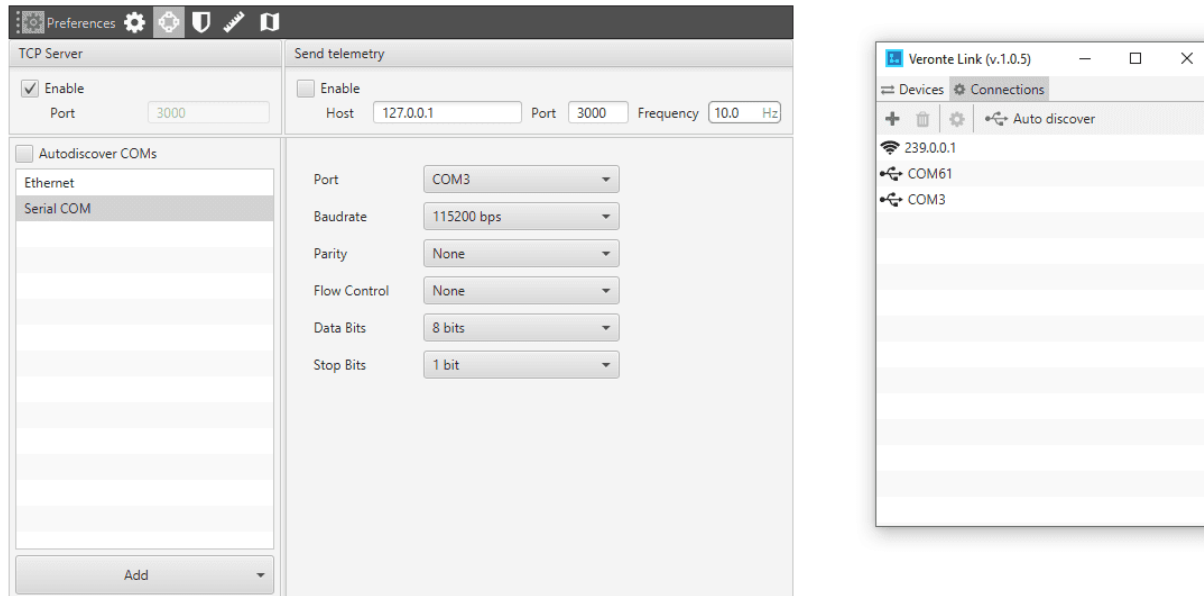


Windows Device Manager

Warning: If the **COM port** is not detected, this will indicate a hardware issue. Please verify the integrity and continuity of all harnesses and wirings. If the problem persists please contact support@embention.com, one of our engineers will connect with you to diagnose the problem.

13.2.2 Check Pipe/Veronte Link

If the **COM Port** is correctly detected, the next step is to check that **Veronte Pipe/ Veronte Link** is correctly configured to read that **PORT**. Verify that either the individual **PORT** is configured or that the **Autodetect COMs** option is checked:



COM Port Configuration

13.2.3 Force Maintenance mode

If the **COM Port** is correctly configured but **Veronte** still cannot be detected, its is likely that the USB communication has been disabled in the configuration.

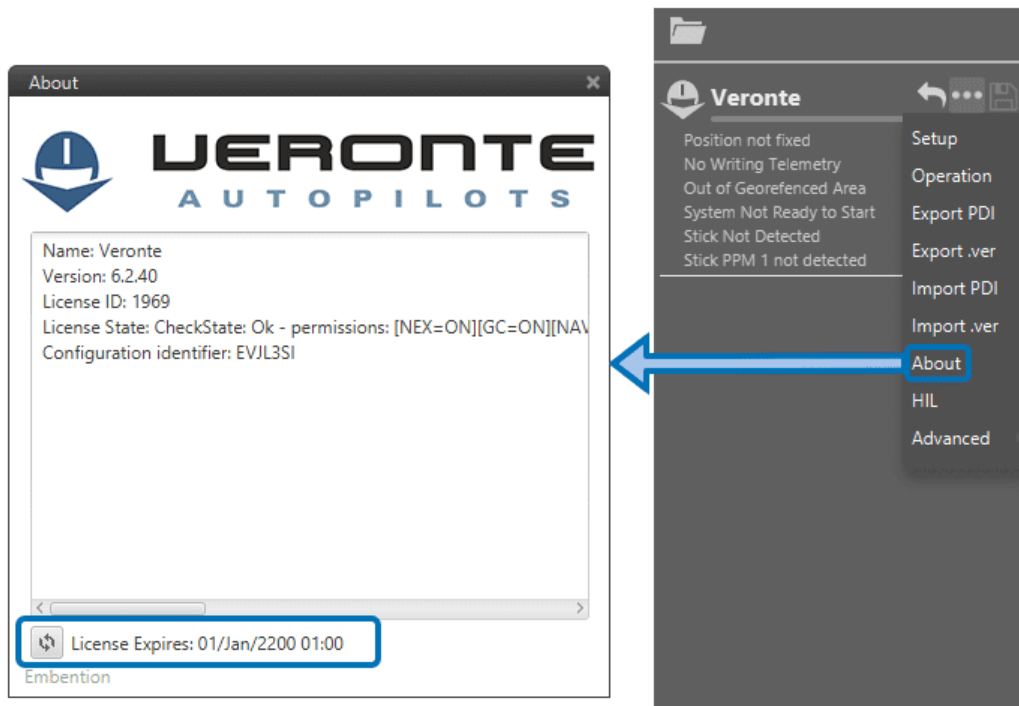
If thats the case, it is necessary to force *Maintenance mode* in order to recover the communication and load a new configuration.

13.3 Licensed Expired

Since 2019, it is **not necessary** to refresh the Veronte firmware license **periodically** anymore for **lifetime licenses**.

Still, if your **Veronte** unit has an **Old License** file, or its **SD Card** was **formatted** recently, it is possible to get a **License error** when connecting the unit to **Veronte Pipe** software.

If that is the case, just connect **Veronte** to a PC with **Internet access**, open its **About** tab and then click on the **refresh** button.



License check

if this does not solve the issue, please contact support@embention.com.

13.4 Out of Georeferenced Area

This warning will appear whenever **Veronte** has no terrain information for its **current position**.

In order to solve it, go to the **Mission** menu and verify that your current position (as well as your whole **mission area**) is covered by at least one of the **Terrain Meshes**:

Terrain Marks

Terrain Profile

Auto

Margin

0.2

x

Delta (north and east) where dem is valid

10.0

m

Select in Map

Fine

Show

Max resolution

Coarse

Show

Geoid

Show

Latitude

0.00274386

rad [-π,π]

Longitude

-0.0028295

rad [-π,π]

Latitude

-0.00242304

rad [-π,π]

Longitude

0.00301430

rad [-π,π]

Latitude

0.01058022

rad [-π,π]

Longitude

-0.0070706

rad [-π,π]

Latitude

-0.0077458

rad [-π,π]

Longitude

0.00732830

rad [-π,π]

Latitude

-0.0120694

rad [-π,π]

Longitude

-0.0137582

rad [-π,π]

Latitude

0.01567150

rad [-π,π]

Longitude

0.01030543

rad [-π,π]

Magnetic field

Auto

Select in Map

North

2.6948315E-5

East

4.33883537E-7

Vertical

3.5114364E-5

Declination

0.0

rad [-π,π]

Inclination

0.0

rad [-π,π]

Horizontal

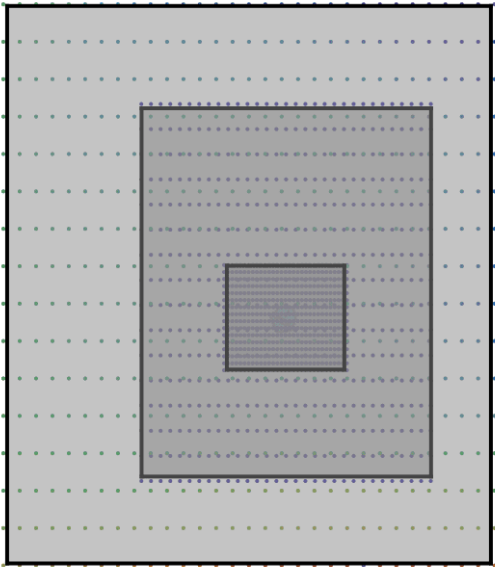
0.0

T

Magnetic Navigation Threshold

1.0

x1



License check

ACRONYMS & DEFINITIONS

14.1 Acronyms

| | |
|--------|---|
| 16 VAR | 16 Bits variables (Integers) |
| 32 VAR | 32 Bits variables (Reals) |
| ADC | Analog to Digital Converter |
| AGL | Above Ground Level |
| AoA | Angle of Attack |
| ARC | Arcade Mode |
| AUTO | Automatic Mode |
| BIT | Bit Variables |
| CAN | Controller Area Network |
| CAP | Capture Module |
| CMB | Climb Phase |
| CRU | Cruise Phase |
| DC | Direct Current |
| DGPS | Differential GPS |
| ECAP | Enhanced CAP |
| ECEF | Earth Centered – Earth Fixed |
| EGNOS | European Geostationary Navigation Overlay Service |
| EKF | Extended Kalman Filter |
| FCS | Flight Control System |
| FHSS | Frequency Hopping Spread Spectrum |
| FLR | Flare Phase |
| FTS | Flight Termination System |
| GIS | Geographical Information System |
| GND | Ground |
| GNSS | Global Navigation Satellite Systems |
| GPIO | General Purpose Input Output |
| GPS | Global Positioning System |
| GS | Ground Speed |
| GS | Ground Segment |
| HLD | Hold Phase |
| HUM | Hardware User Manual |
| I2C | Inter-Integrated Circuit |
| IAS | Indicated Air Speed |
| ID | Identification |
| ISM | Industrial Scientific and Medical |

continues on next page

Table 1 – continued from previous page

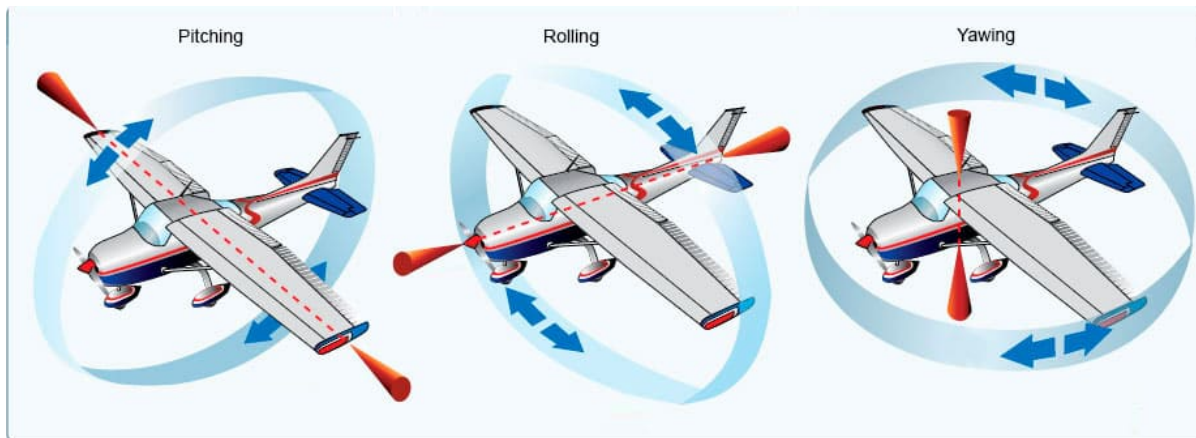
| | |
|--------|---|
| LED | Light-Emitting Diode |
| LND | Landing Phase |
| MSL | Mean Sea Level |
| PFD | Primary Flight Display |
| PID | Proportional Integral Derivative |
| PWM | Pulse Width Modulation |
| QNH | Barometric atmospheric pressure adjusted to sea level |
| RC | Radio Control Mode |
| RF | Radio Frequency |
| RPAS | Remotely Piloted Aircraft System |
| RPM | Revolutions Per Minute |
| RS 232 | Recommended Standard 232 |
| RS 485 | Recommended Standard 485 |
| RX | Reception |
| SMA | SubMiniature Version A Connector |
| SSMA | Miniature-Sized Connector |
| STB | Standby Phase |
| SU | Servo-Output matrix |
| SUM | Software User Manual |
| TAS | True Air Speed |
| TKO | TakeOff Phase |
| TX | Transmission |
| UAS | Unmanned Aerial System |
| UAV | Unmanned Aerial Vehicle |
| US | Output-Servo matrix |
| VTOL | Vertical TakeOff and Landing |
| WGS 84 | World Geodetic System 84 |
| WP | Waypoint |

14.2 Definitions

- **Control Phase:** The operation is divided into phases in which the UAV has a specific performance. Each of this phases is called a control phase.
- **Control Channel:** It is each of the signals used to control a behaviour or action.
- **Control Mode:** It is possible to make a manual control of the UAV by stick, assisted control and fully automatic control.
- **Actuator:** It is a mechanic device to provide force to move or “act” another mechanical device.

AXES CONVENTION

All signs in the system are managed according to the international aeronautical axes convention: It is considered positive any deflection that generates positive rotational forces or moments about the aerodynamic centre of the aircraft, except for “y” axis (elevator) where it is considered negative a positive moment.



Sign Convention

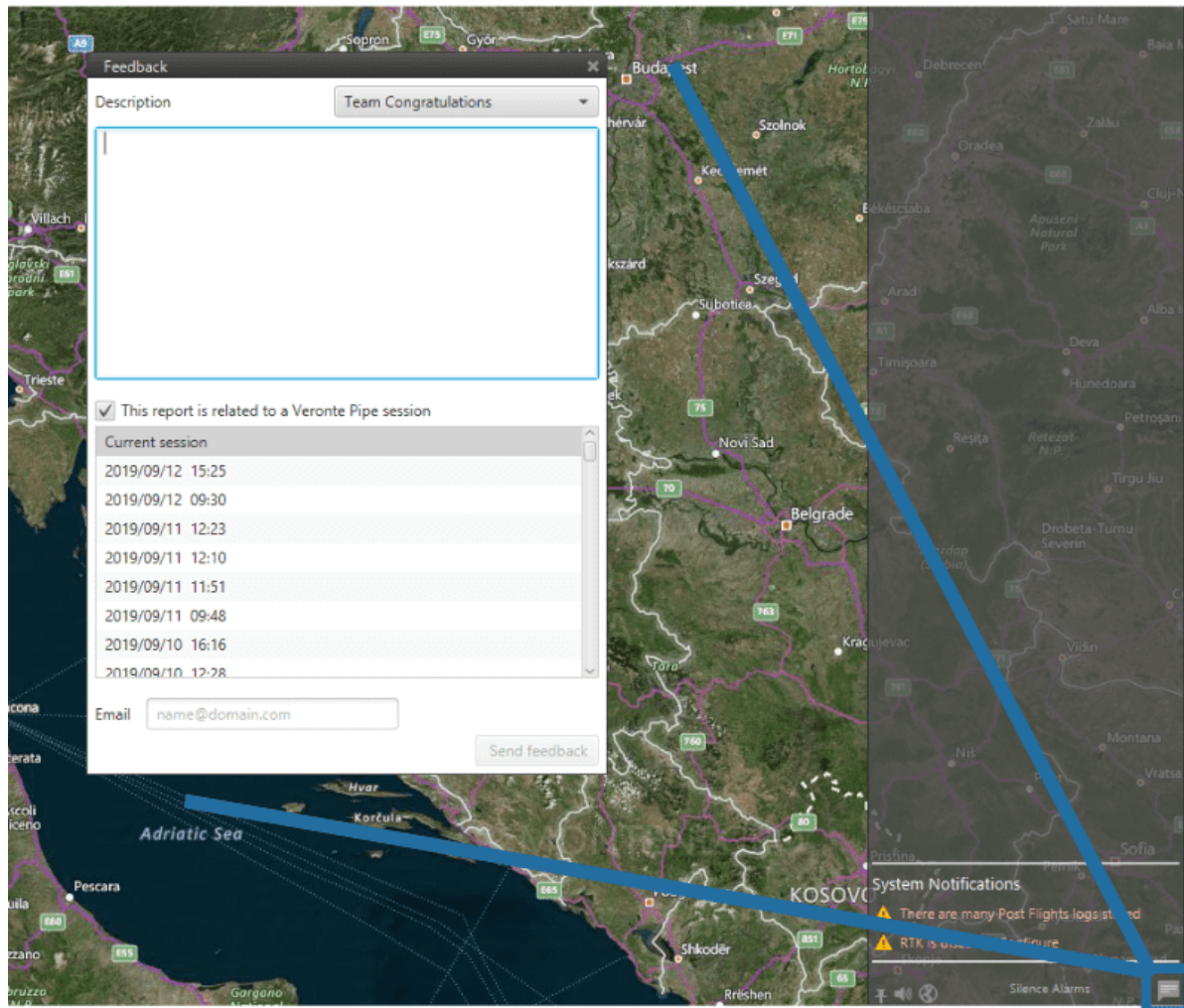
Example, an elevator down position will generate a positive pitch so the elevator is considered positive on down position.
Main actuators rules:

| Actuator | Positive | Negative |
|---------------|----------|----------|
| Elevator | Down | Up |
| Rudder | Right | Left |
| Right Aileron | Up | Down |
| Left Aileron | Down | Up |
| Tail Rotor | Right | Left |

FEEDBACK

16.1 Feedback Panel

If you'd like to provide us with feedback on future releases, issue report or even team congratulations, please provide us with all possible details about your Veronte experience and we will be happy to listen and to help you.



Feedback Panel

We have created the tool shown in the previous figure in order to allow the user to directly give us feedback about:

- **Team Congratulations:** Give us your positive feedback about your Veronte experience.
- **Suggestion for Next Releases:** All your comments and suggestions would be greatly appreciated.
- **Configuration Support:** Your feedback will be sent directly to our configuration team. They will help you with all your configuration troubles.
- **Failure Report:** It would be extremely useful to help our developing team reaching all possible information about Veronte System bugs or failures.

The screenshot shows a 'Feedback' window with a close button (X) in the top right corner. Below the title bar, there is a 'Description' label and a dropdown menu currently set to 'Team Congratulations'. A large text area for the description is below the dropdown. Underneath the text area is a checked checkbox labeled 'This report is related to a Veronte Pipe session'. Below the checkbox is a list box titled 'Current session' containing several date and time entries: 2019/09/12 15:25, 2019/09/12 09:30, 2019/09/11 12:23, 2019/09/11 12:10, 2019/09/11 11:51, 2019/09/11 09:48, 2019/09/10 16:16, and 2019/09/10 12:28. Below the list box is an 'Email' label and a text input field containing 'name@domain.com'. At the bottom right of the window is a 'Send feedback' button.

Feedback Panel Options

Once the feedback type is selected, the user can add a **Description** to detail the content. Furthermore, the report can be sent adding the reference to the **Veronte Pipe session**, which could be the actual one or the one selected from the list. If the user is interested in having an answer directly to his e-mail address, an **Email** can be inserted in the corresponding field.

When the window is entirely completed and an internet connection is available, it is necessary to click on Send feedback and the report will be sent.

COMPATIBLE DEVICES

17.1 Servos and ESC

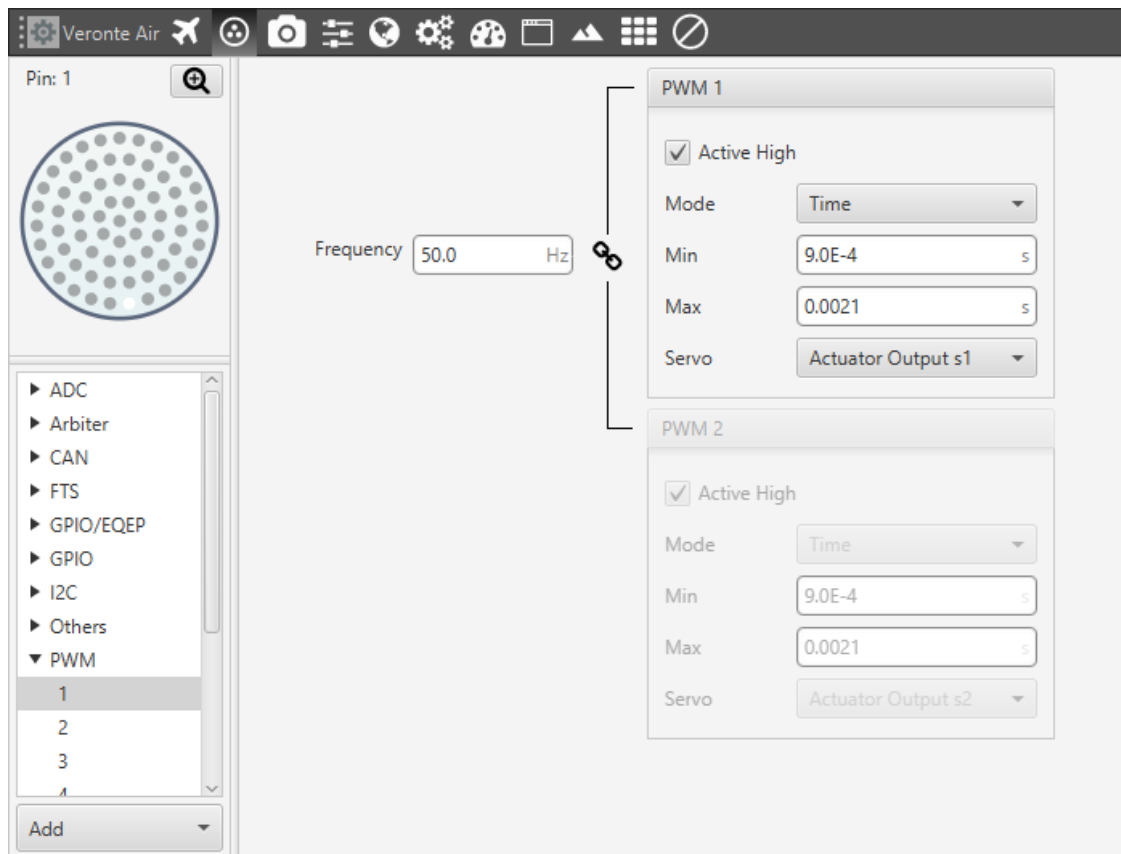
The user can configure any actuator compatible with the communication interfaces

17.1.1 PWM Servos

1. Connect the servo following the manufacturer documentation and follow the *electrical chapter* connect it to Veronte.

| |
|---|
| Warning: 4x Redundant Veronte autopilot has different pinout specifications. |
|---|

2. In the setup menu, go to **Connections/PWM** assign a given actuator to the PWM and set a frequency according to the manufacturer. In this example *Actuator Output 1* is assigned to PWM1



PWM servo - connections

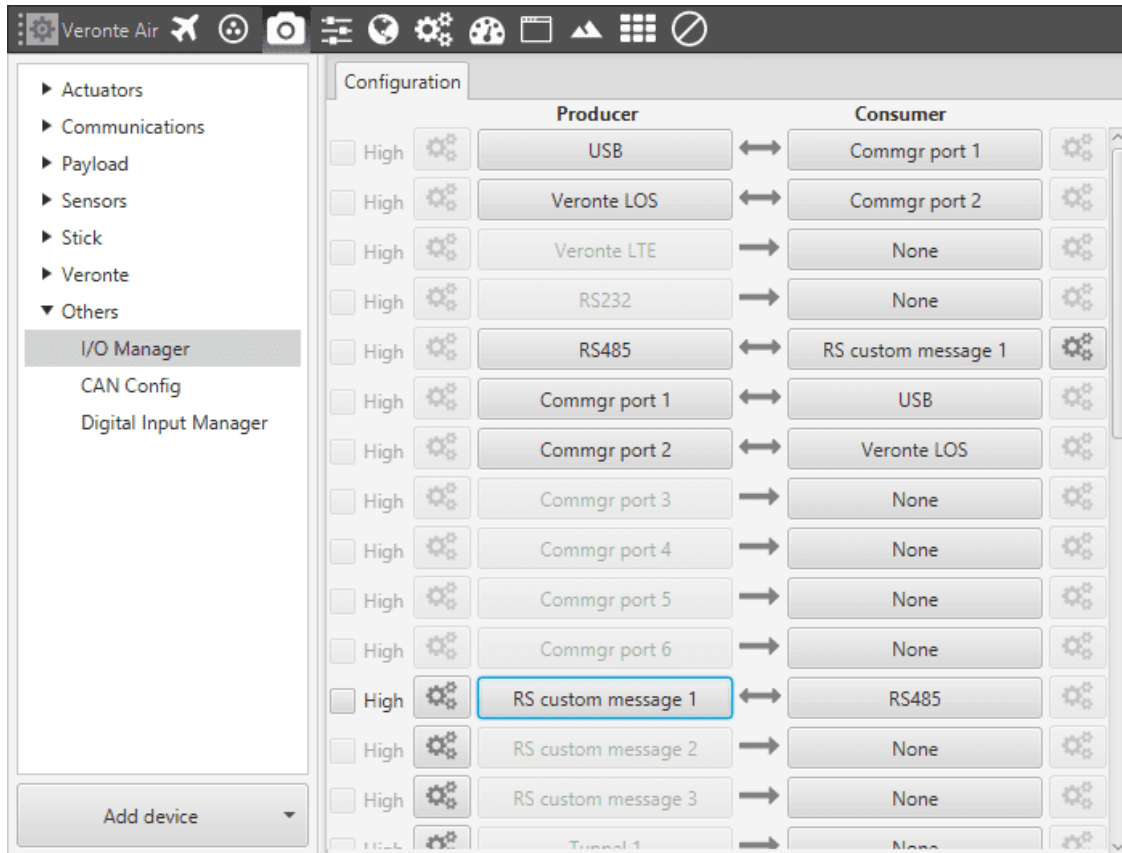
Note: Check the [actuators menu](#) for further details.

17.1.2 Serial servos

Serial servos are configured different compared to PWM servos since the protocol of a serial device must be defined in [custom messages chapter](#). In this case an *Actuator Output* variable must be sent through a serial interface as the protocol for the devices specifies.

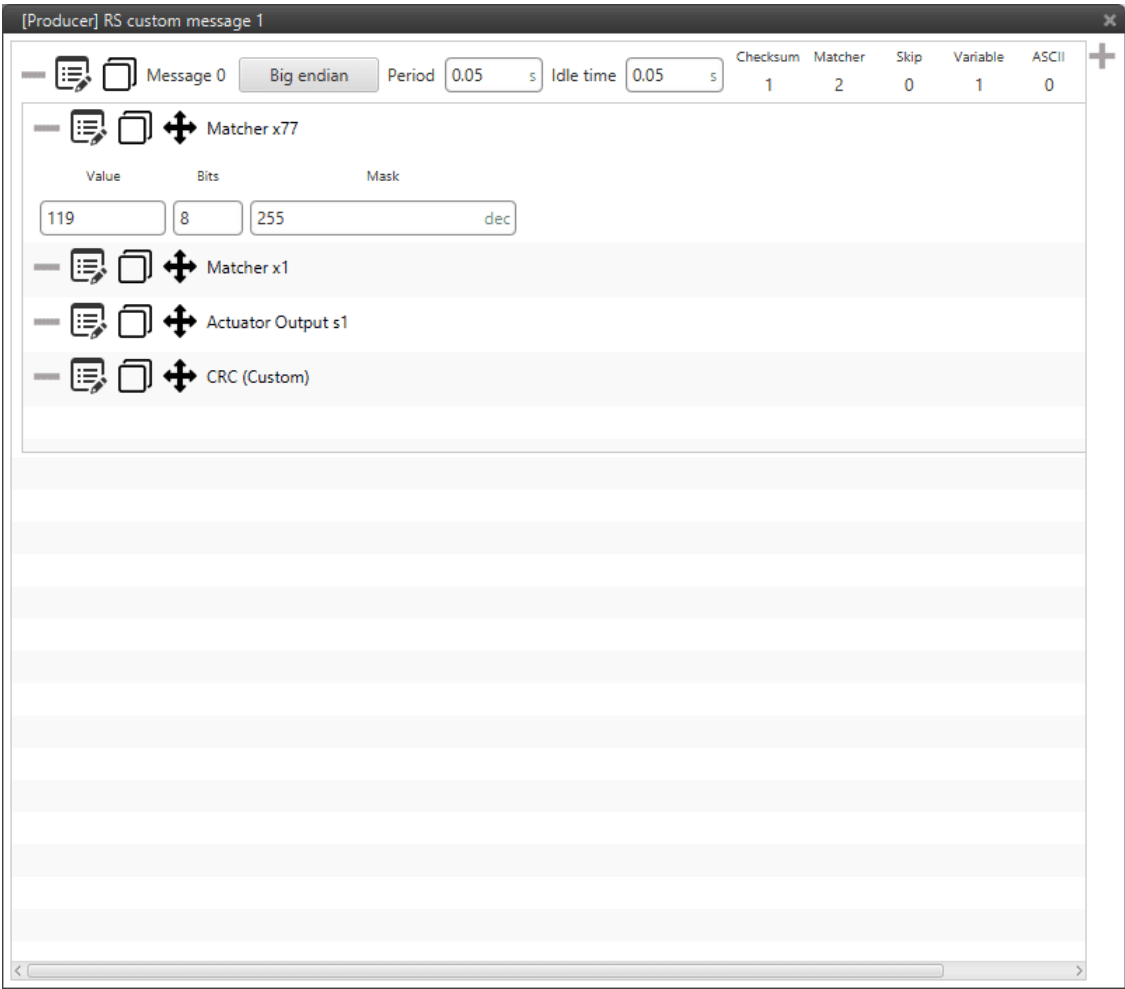
17.1.2.1 Servo Volz DA26

1. In the setup menu go to **Devices/Other/I/O Manager**. In this example the routing is set to send *RS custom message 1* to use RS485 bus. As it can be seen for the RS485 consumer and producer configuration.



I/O manager configuration

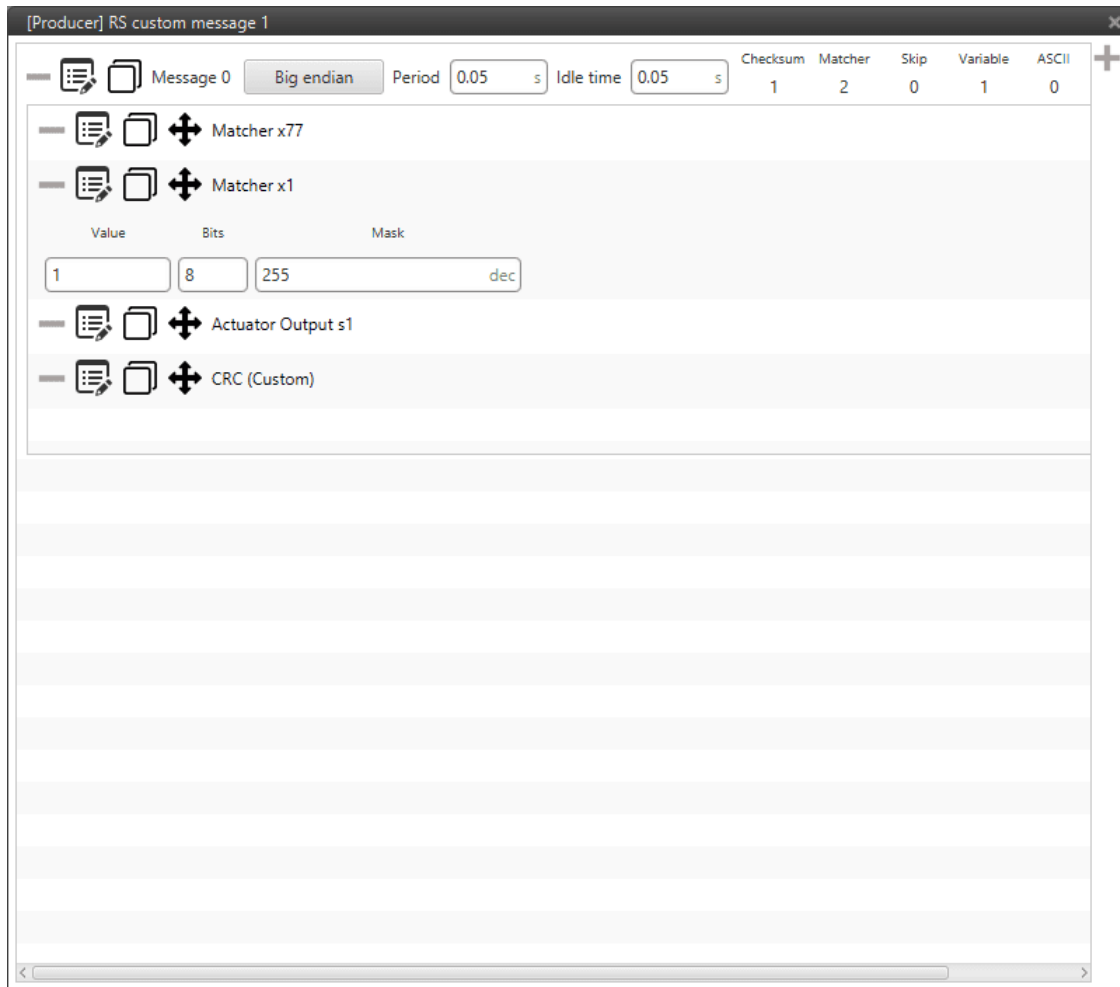
2. Next step is to define the protocol specified by the manufacturer.
 - In this case a silent mode command (0x77).



0x77 Silent mode command

- Servo interface Id = 1.

Note: For Volz servos the Id will be different for each servo and/or interface.



Servo Id

- *Actuator Output* is the variable that carries the information that has to be applied to the servo. Therefore it must be included in the message.

Note: For other actuators/devices the *Actuator Output* variable will be needed but it will be sent according to the specific protocol.

[Producer] RS custom message 1

Message 0 Big endian Period 1.0 s Idle time 0.0 s Checksum 1 Matcher 2 Skip 0 Variable 1 ASCII 0

Matcher x77

Matcher x1

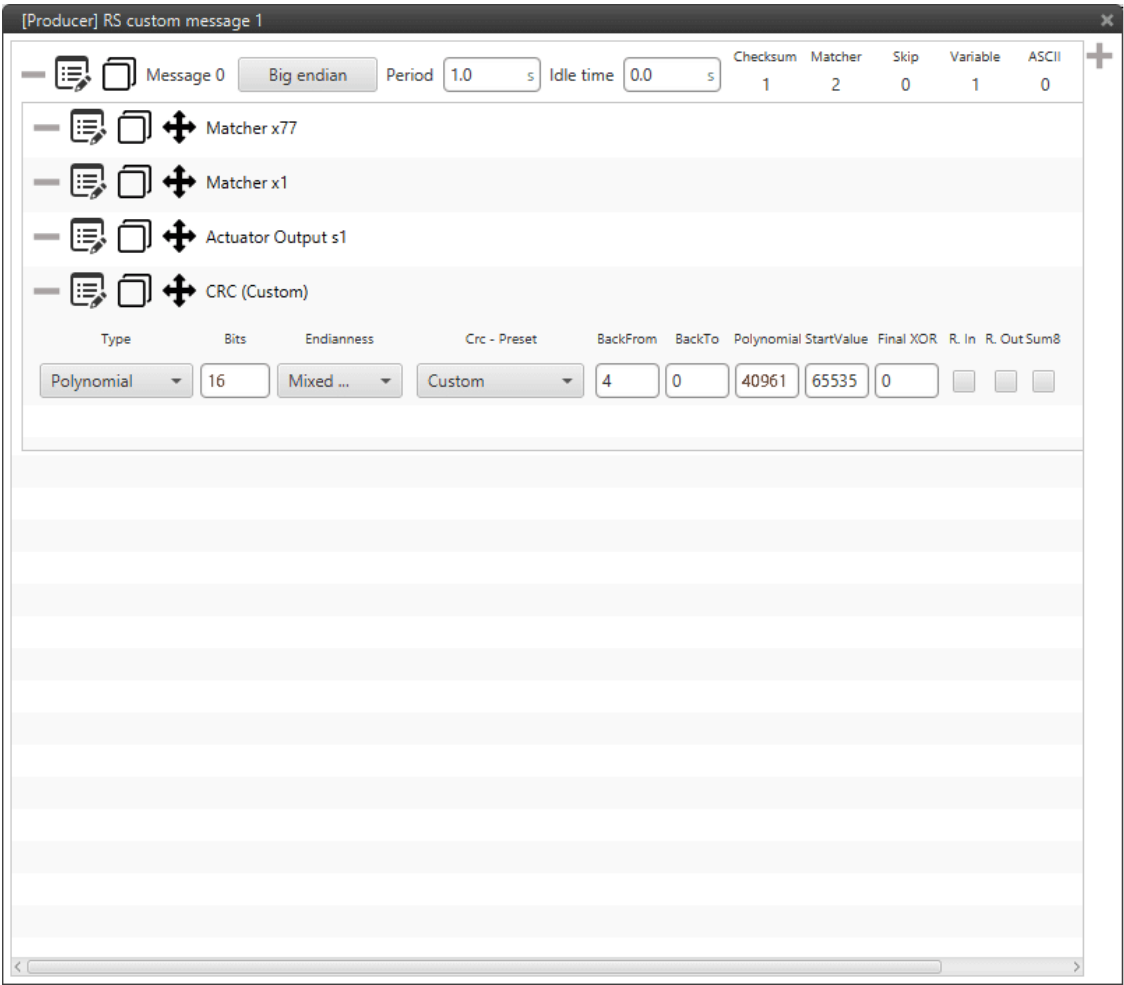
Actuator Output s1

| Variable | Compression | Decimals | Encode/Decode | Encode Min | Encode Max | Decode Min | Decode Max |
|---------------|-------------------------|----------|---------------|------------|------------|------------|------------|
| Actuator O... | Compress - Bits Unigned | 16 | 1 | -0.436 | 0.436 | 1568 | 2527 |

CRC (Custom)

Actuator Output

- Finally a *Checksum* is needed to complete the communication protocol.



Checksum

Note: For more information about checksum take a look [here](#).

17.2 Altimeters

The integration between Veronte and a lidar is performed using a variety of interfaces depending on the lidar device. The most common interfaces are I2C or analog although serial or CAN bus can also be used if the lidar is compatible.

17.2.1 Reading altimeter measurement

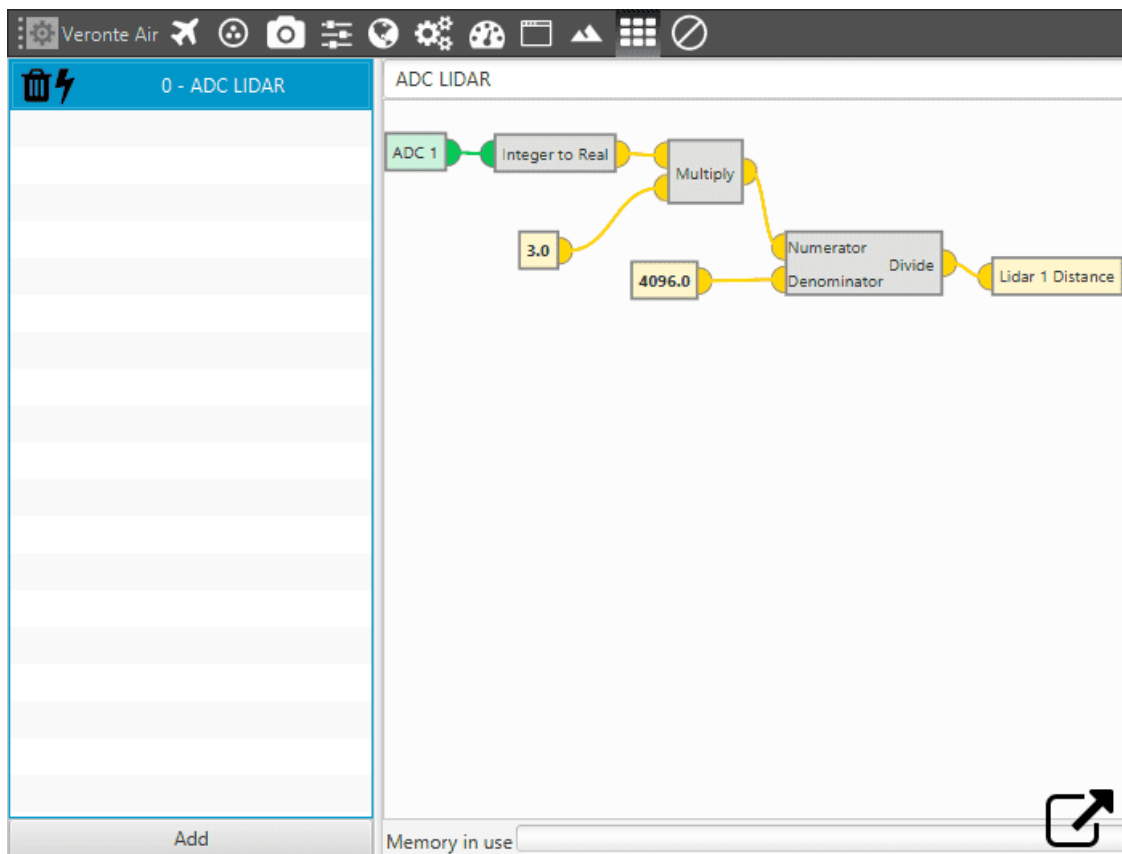
17.2.1.1 ADC lidar

An ADC lidar changes the voltage according to the distance measured and therefore the connection to Veronte is performed using the ADC pins(see section [Electrical](#)).

Warning: 4x Redundant Veronte autopilot has different ADC pins.

Once connected the value can be monitored in Veronte Pipe by using the variables ADC1 to ADC5. For pin *ANALOG_1* the correspondent ADC variable in Veronte Pipe is *ADC1*.

1. Go to **Programs**.
2. Configure the following operation:



Lidar operation

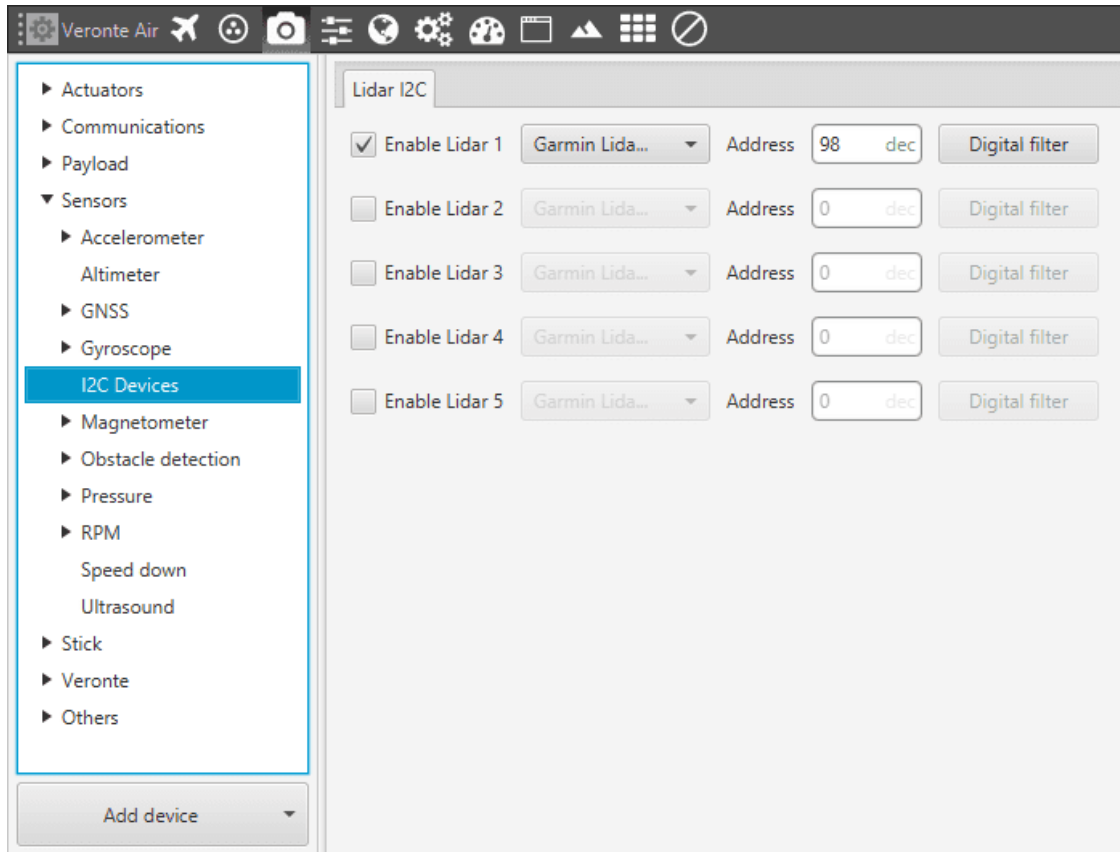
After implementing the operation the variable *Lidar 1 Distance* will represent the distance measured by the lidar.

Note: 4x Redundant Veronte can read up to 36 V for each ADC thus 3 must be changed to 36 if it is the case.

17.2.1.2 I2C lidar

I2C lidars are configured slightly different. Connect the lidar following the pinout provided by the manufacturer and connect it to the I2C bus of Veronte by following the section [Electrical](#). In this case it is not needed to transform the readings from the lidar, the readings will be directly reported in the selected lidar distance variable..

1. Go to **Devices/Sensors/I2C devices**.



I2C Lidar

Note: I2C address will be different for different devices make sure to define it properly by checking the manufacturer documentation.

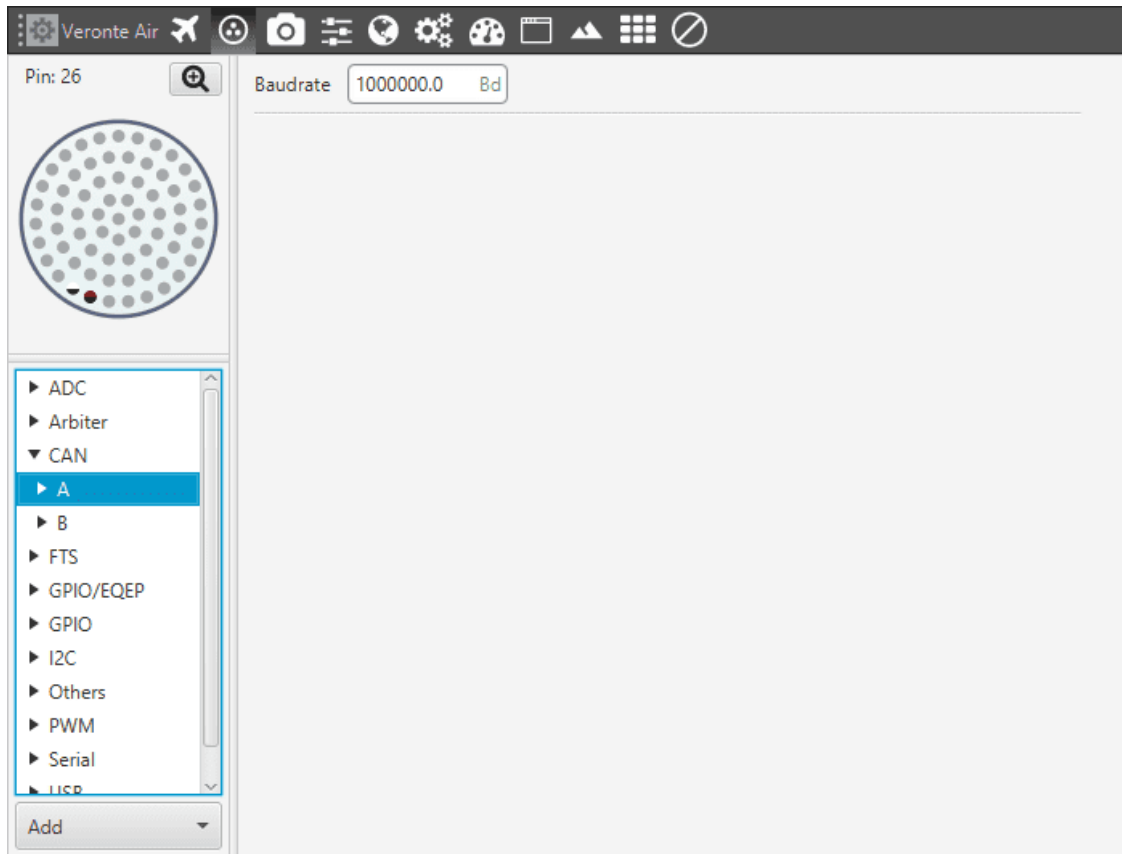
17.2.1.3 CAN Radar

Radar altimeters are common devices for aircrafts. The following pictures correspond to the integration of the Smartmicro Radar. To get more information of the Smartmicro Radar - Altimeter datasheet you can go to:

- [Micro Radar Altimeter Web](#)
- [Micro Radar Altimeter Data Sheet](#)

This settings will allow Veronte to read from CAN A readings from the radar altimeter. In particular AGL and vertical speed. In the datasheet the user can access to the complete protocol of the device.

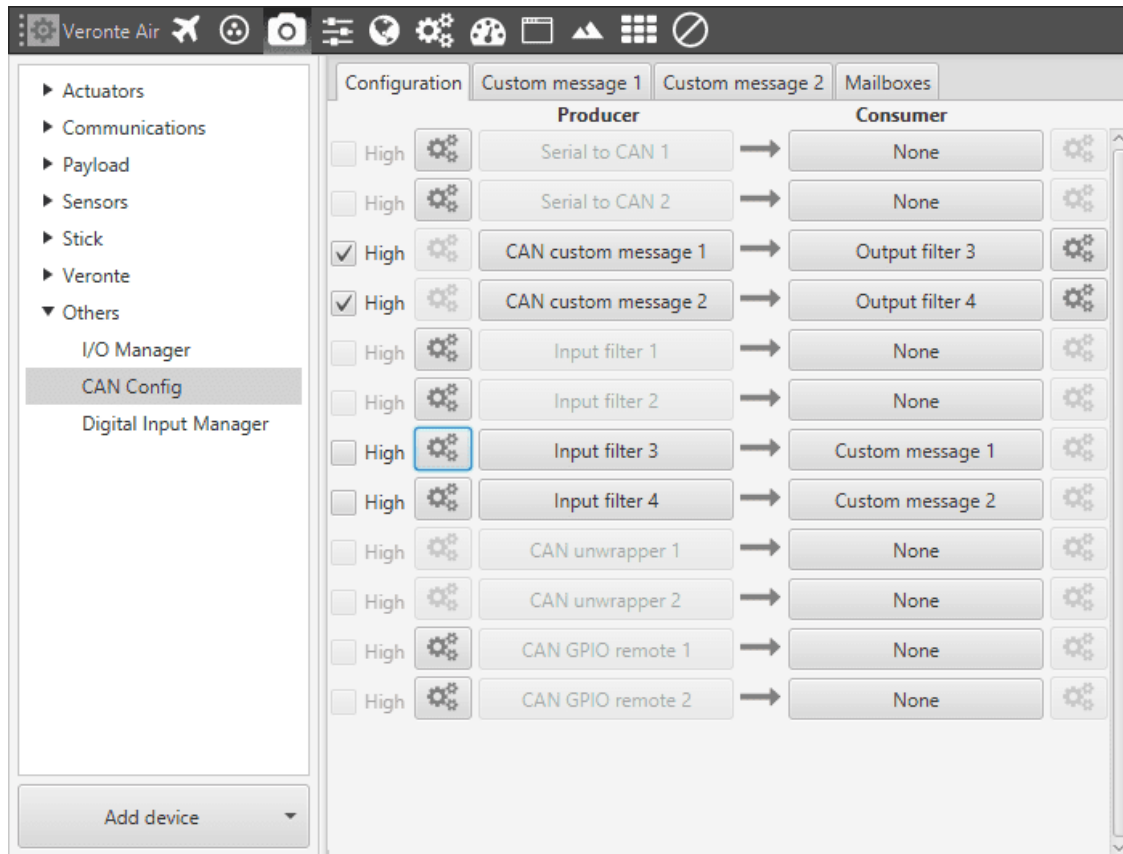
1. Go to **Connections/CAN/A** in the setup toolbar and configure the baudrate.



Baudrate configuration

2. Once the baudrate is set, go to **Devices/Others/CAN Config/Configuration**.

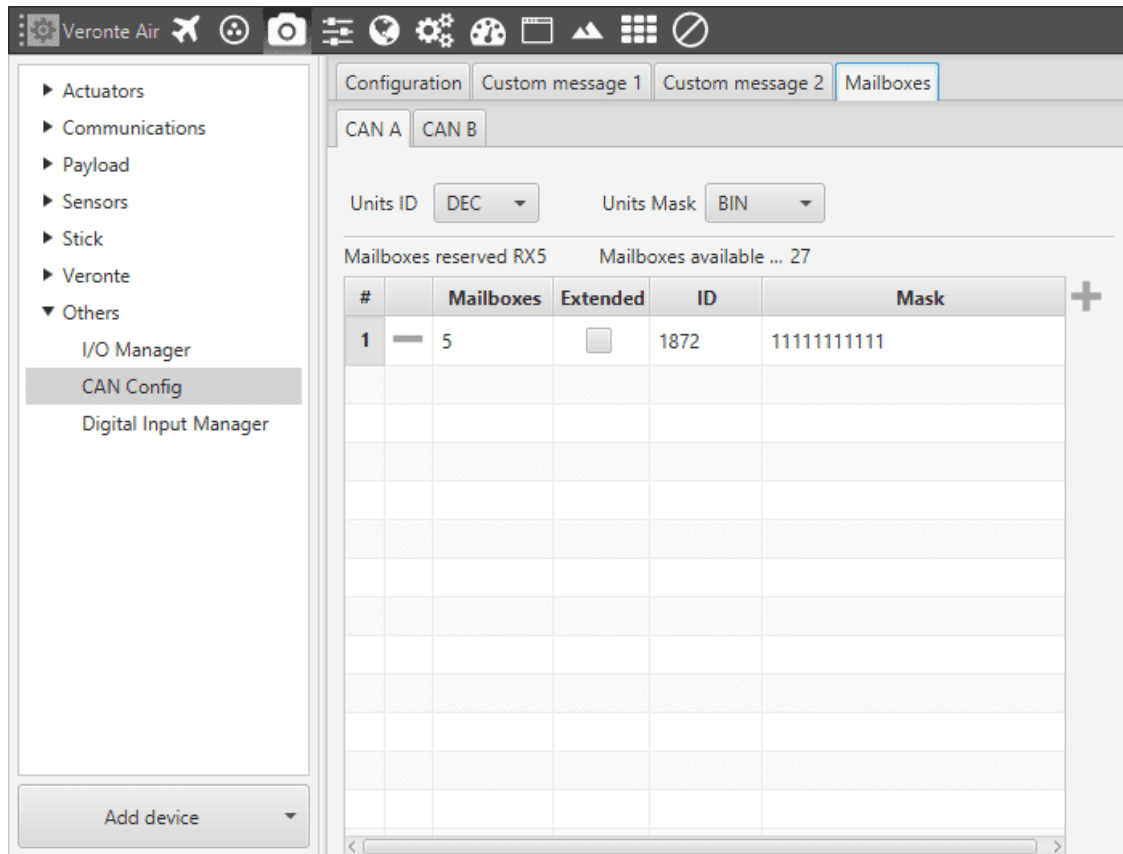
Set one input filter to read from CAN A in *custom message 1*, in this case Input Filter 3. On the next steps CAN custom message 1 will be configured.



CAN routing configuration

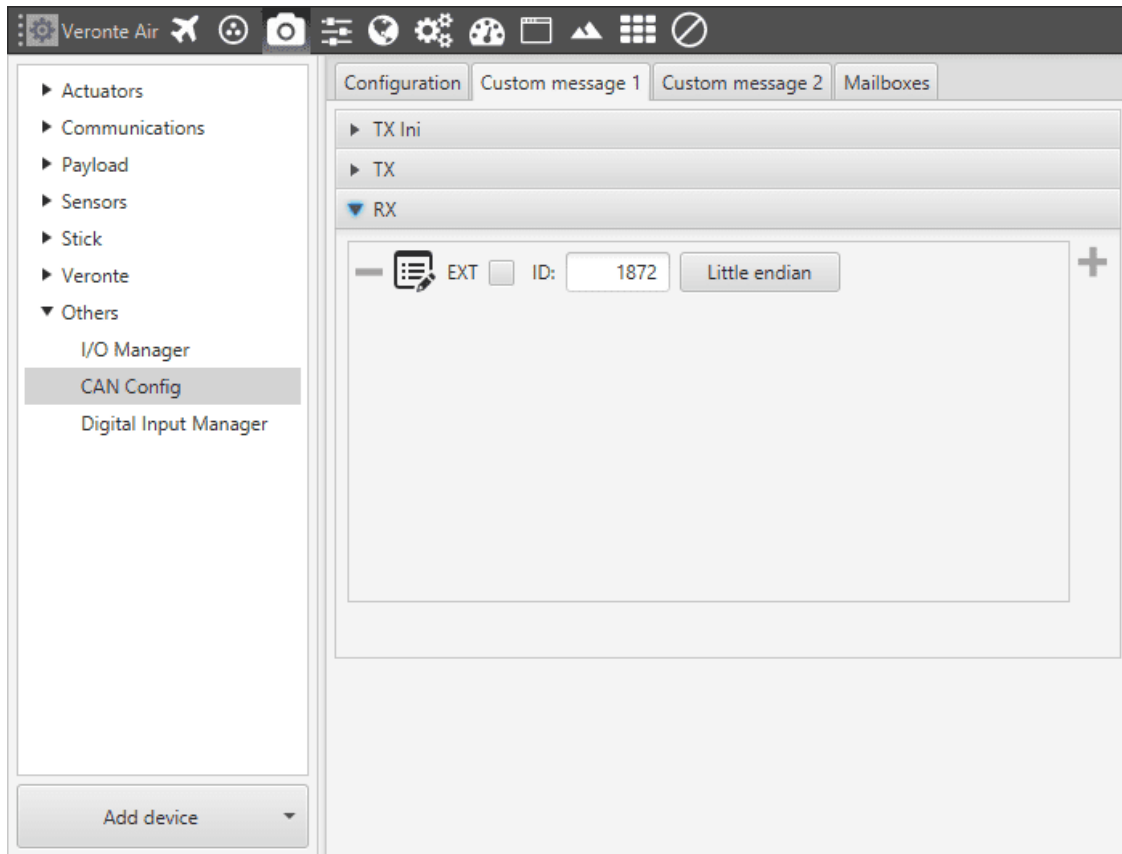
- After specifying that *custom message 1* will receive the data from CAN A, go to **Devices/Others/CAN Config/Mail Boxes**.

Set the mailboxes for the CAN message id: 1872



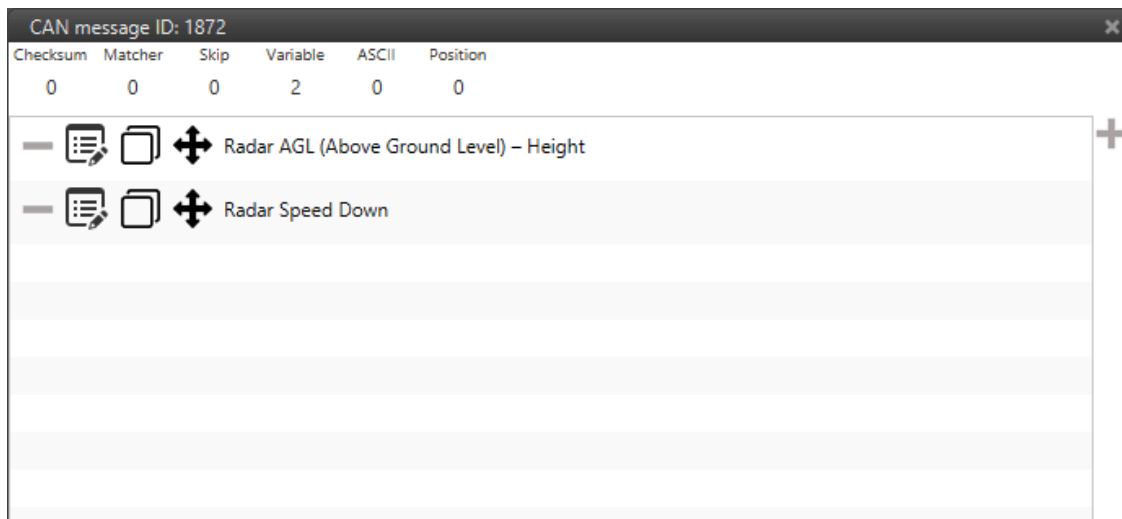
Mail boxes setting

4. Move to the tab Custom message 1 and add new message under RX with the ID 1872.



Rx message definition: Id and endianness

5. Define the content of the incoming message:



Rx message definition: message content

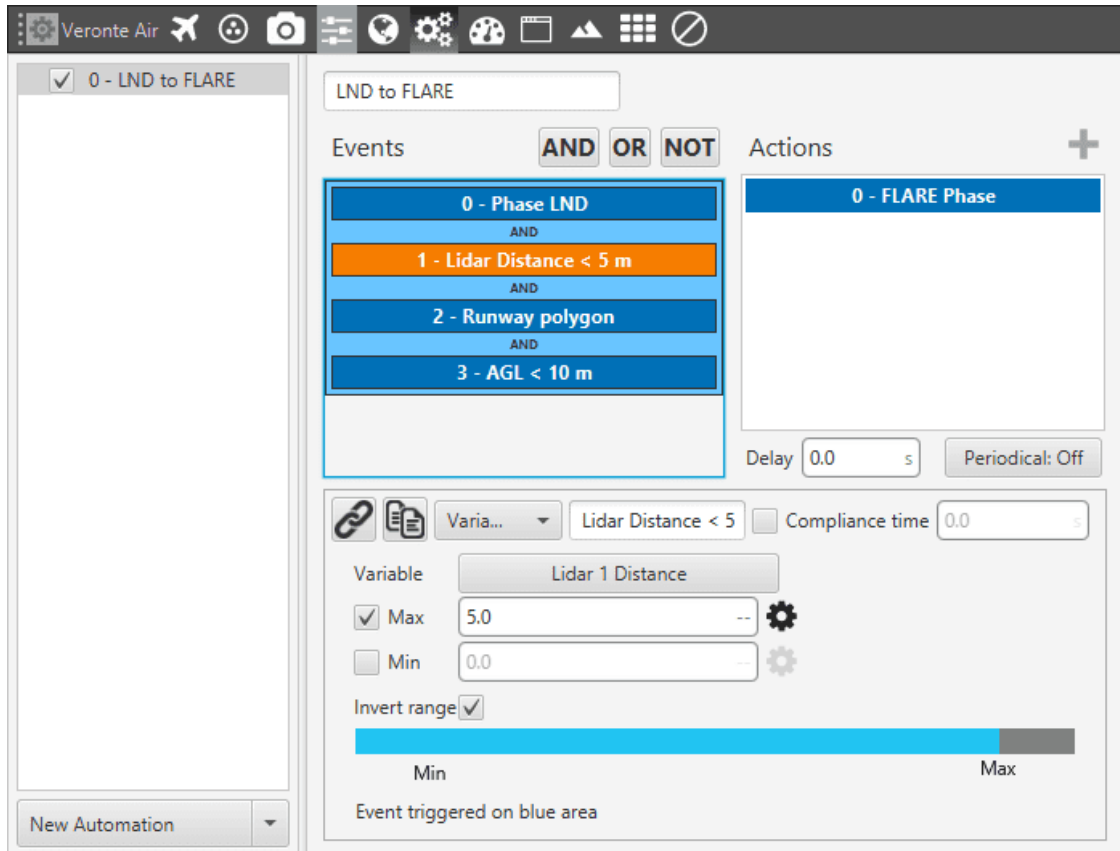
For more details about CAN configuration see [Can messages](#).

Note: CAN ID messages and messages content will change for different Radar altimeters. Check the documentation of your device for further details.

17.2.2 Using altimeter readings

Once the information provided by a lidar sensor is saved in a system variable as Lidar Distance by means of an ADC reading, I2C, serial or CAN. The user has to set how this data will be considered. Common usages are: triggering an action based on a predefined event or considering the lidar data as external sensor which will be considered as an input to the EKF.

- **Automation:**

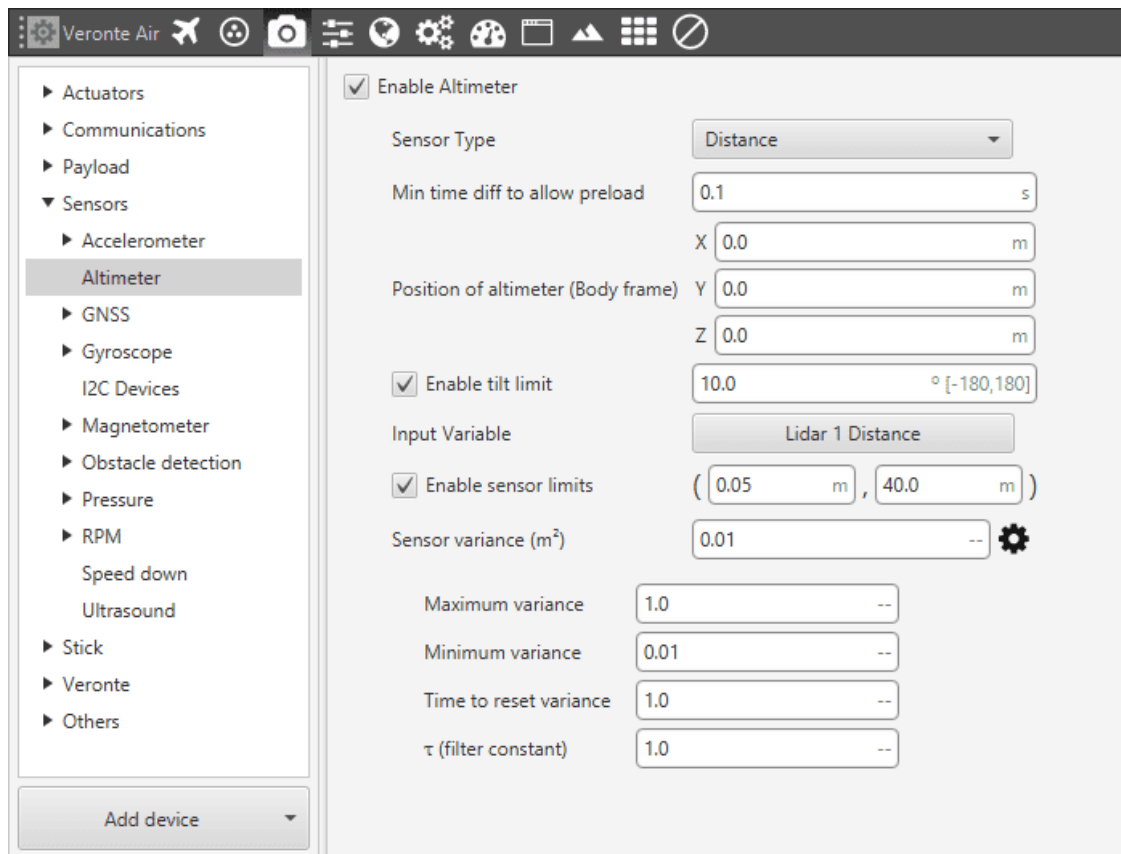


Lidar Automation example

This automation will be triggering a change of phase, flare phase, when the aircraft is landing and at 5 m AGL.

- **External sensor:**

Further information about this menu can be found in [altimeter section](#).



Altimeter configuration

17.3 External sensors

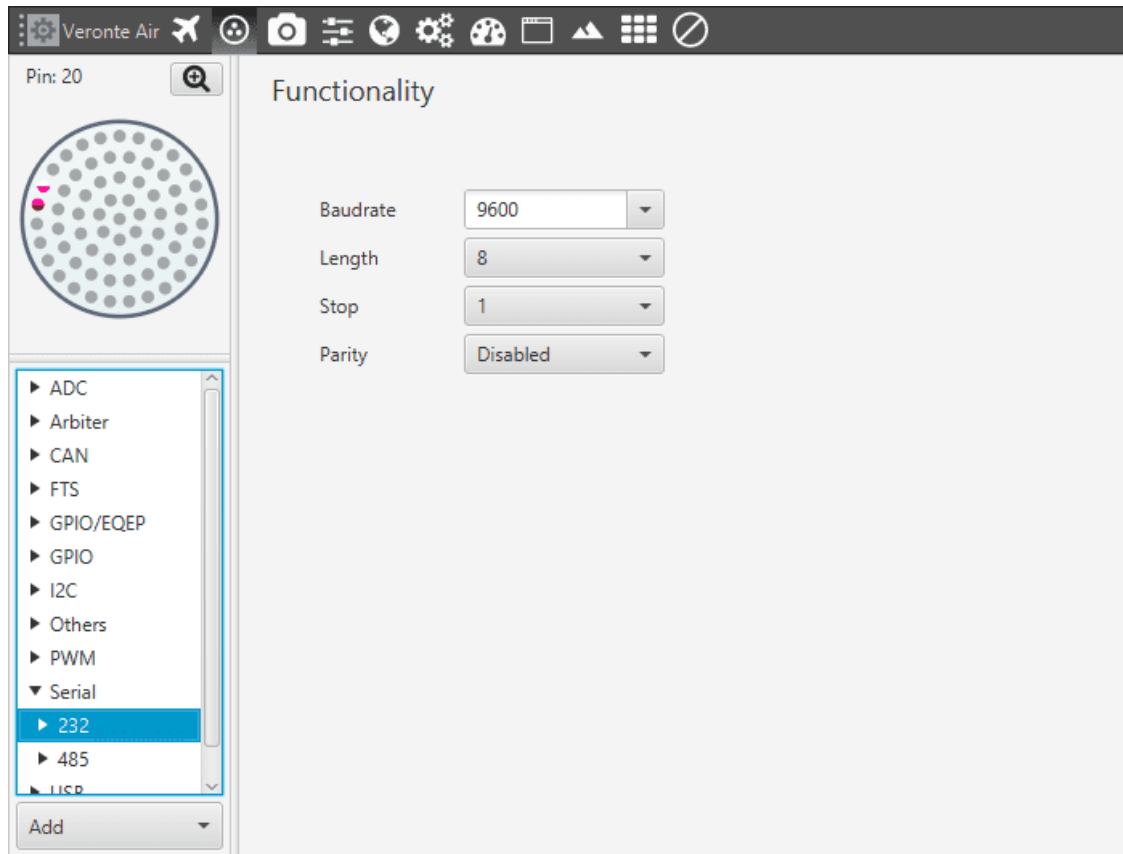
Veronte can be integrated with any external sensor that shares the communication interface. External sensors can be configured to be considered as part of the sensors fusion.

17.3.1 Magnetometer Honeywell HMR2300-232

Connect the serial interface according to the manufacturer specifications and following the *electrical chapter* of this document.

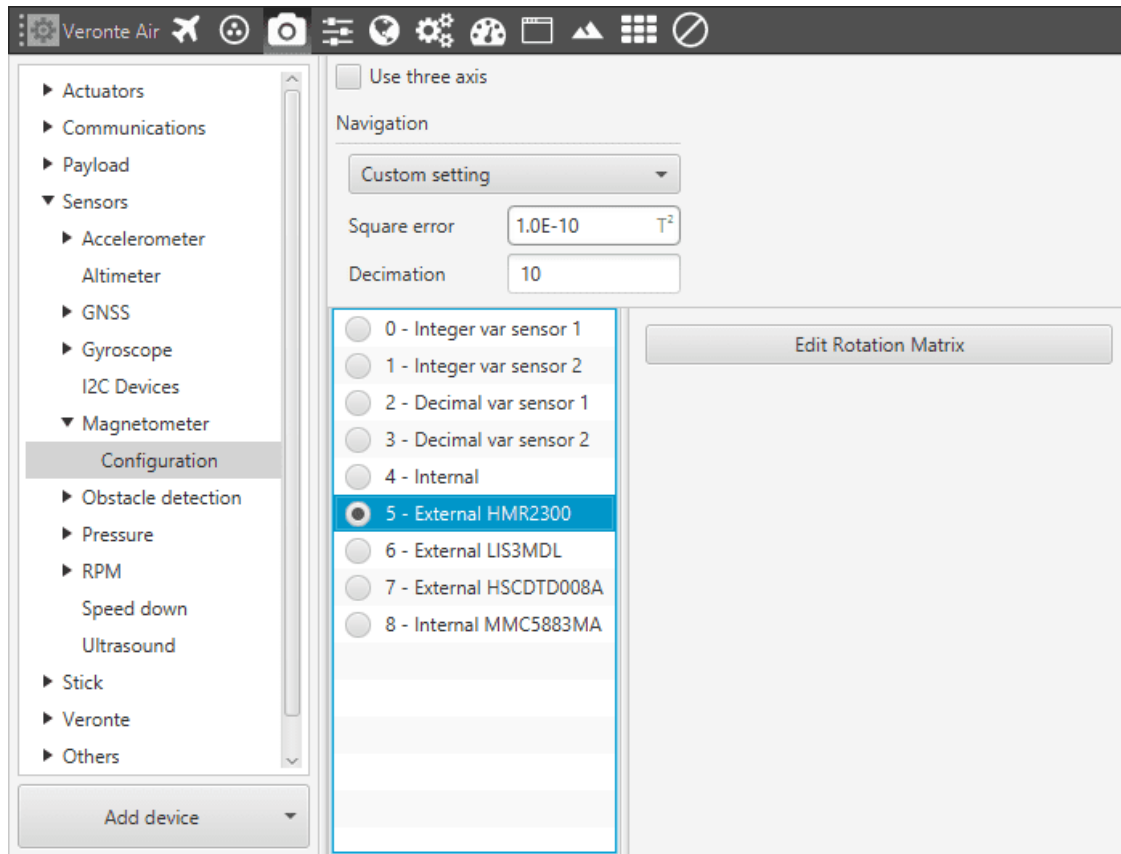
Warning: 4x Redundant Veronte autopilot has different pinout specifications.

1. Open the Setup Toolbar and go to **Connections/Serial/232** and configure the serial communication settings.



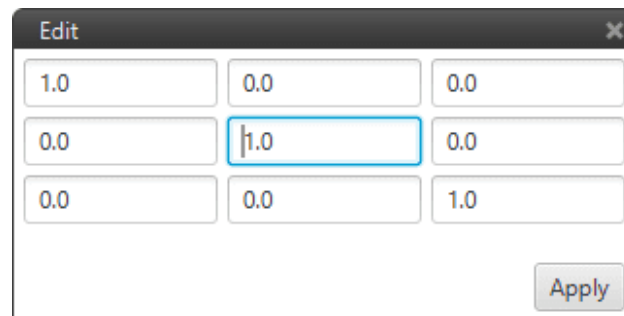
Serial settings

2. Select **Devices/Sensors/Magnetometer/Configuration** and choose *External HMR2300*



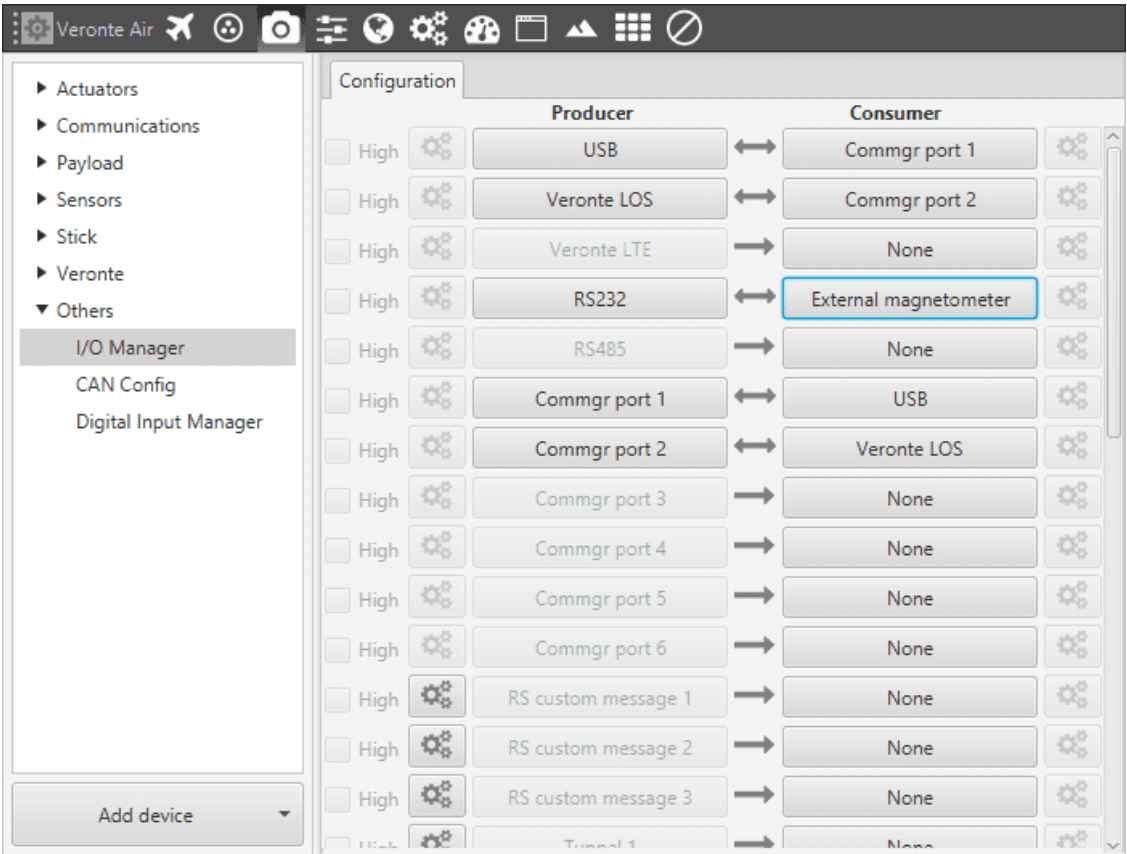
Magnetometer selection

3. Set the rotation matrix according to the installation for the sensor by clicking in *Edit Rotation Matrix*:

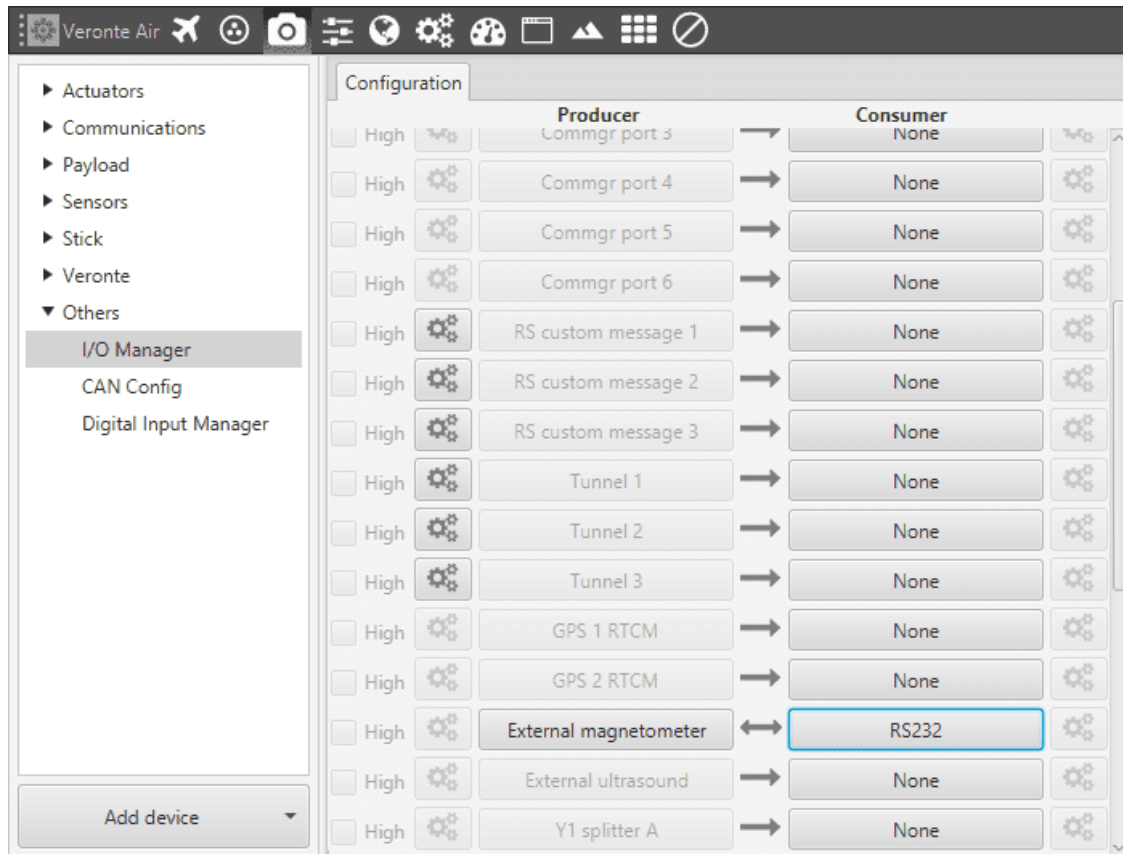


Rotation matrix

4. In **Devices/Other/ I/O Manager/** select the RS232 to receive *External magnetometer* information.



Producer



Consumer

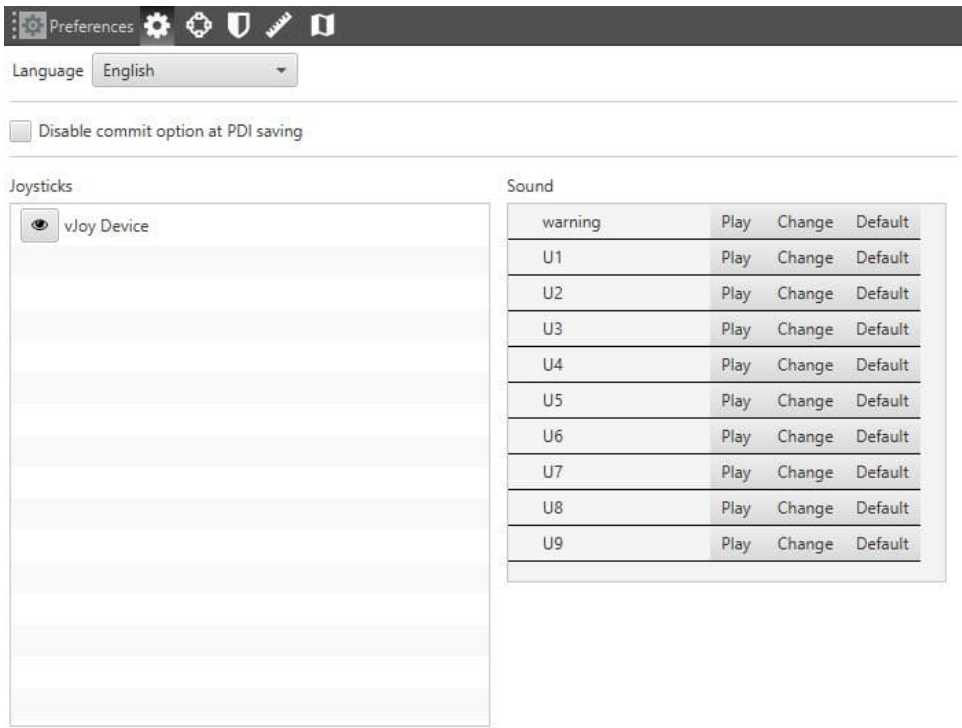
To get more information about the Magnetometer Honeywell HMR2300-232 datasheet check out: [Smart Digital Magnetometer HMR2300](#).

17.4 USB devices

17.4.1 Adding a USB joystick

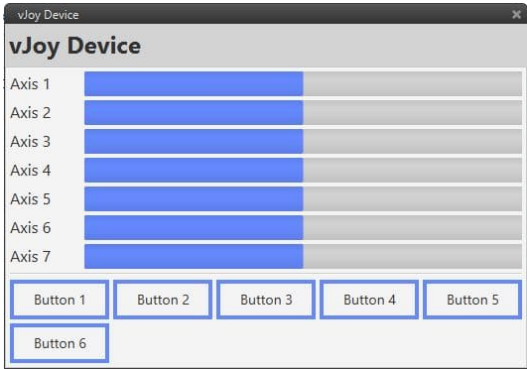
Veronte Pipe is able to detect USB devices such as joysticks. The buttons and axis of these devices can be read and configured to send stick information to Veronte Autopilot.

1. Connect your USB device to the computer and open Veronte Pipe.
2. Go to **Preferences/General** in Veronte Pipe. The USB device will be automatically detected and displayed.



Joystick device

Clicking  the readings of the USB device channels will be visualized.

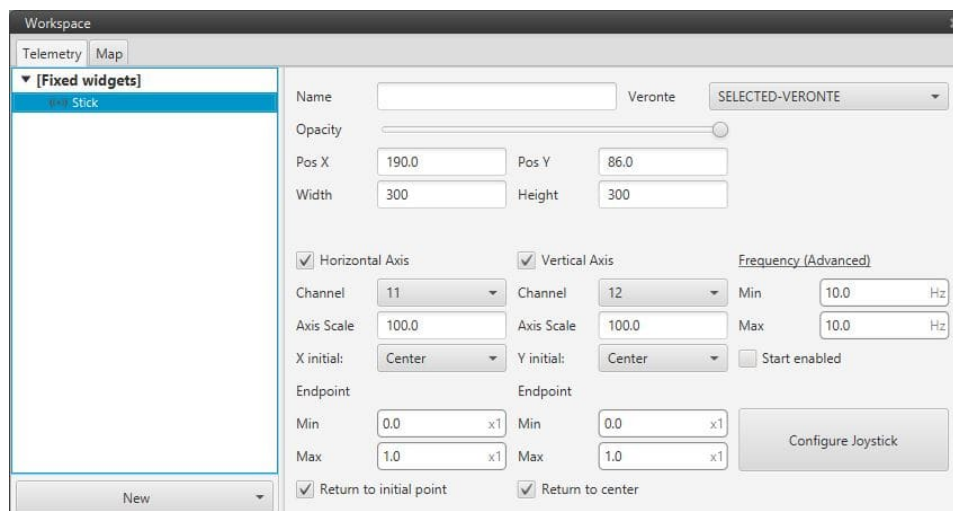


Usb device channels

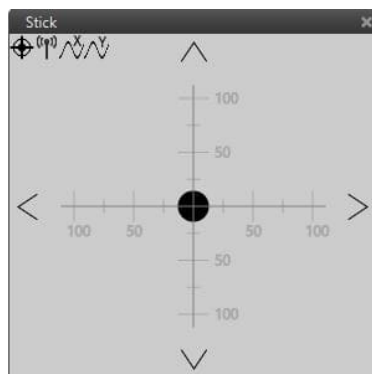
17.4.2 Defining a virtual stick

Once the usb device is recognized:


1. Add a virtual Stick: On the workspace window, select ,  Telecommand and then stick.



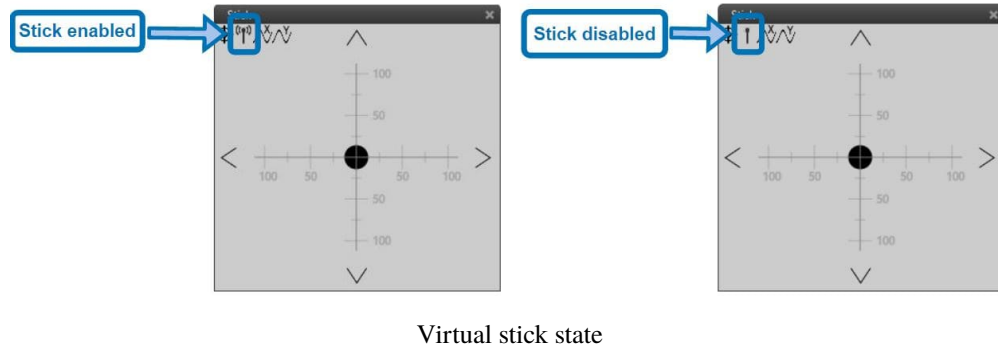
Joystick workspace menu



Joystick widget

2. Assign a USB device channel to a stick input variable.
 1. For each axis specify which channel will be considered by choosing one of the channels listed.
2. Select  and follow the instructions.
3. Verify that the movement of the virtual stick is associated with the movement of the USB device.
4. Check that the stick input variables are changing with the movement of the virtual stick. i.e for channel 1, stick input r1.

Note: The virtual stick will be active by clicking on the antenna button.



Note: It is recommended disabling the axis that are not used.

Warning: A virtual stick can command on the same channels as a physical stick box. Make sure that the channels are not interfering to each other.

17.5 Silvus radio (StreamCaster 4200E model)

17.5.1 System Layout

The following image shows the standard connection between Silvus radios and 1xVeronte for operation:



Fig. 1: Silvus and Veronte connection

17.5.2 Hardware Installation

A wiring configuration of the PRI cable connected to the PRI port of the radio is required in order to connect to the power supply, ethernet and RS-232.

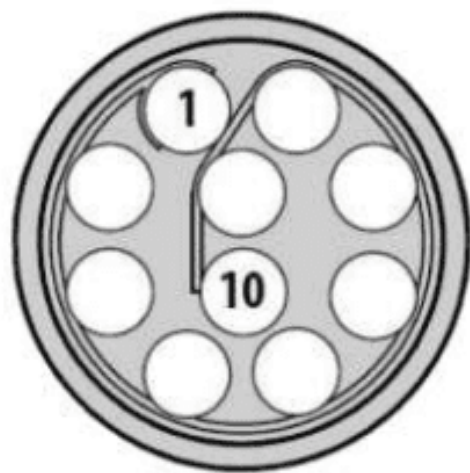


Fig. 2: PRI port connector (mounted in radio)

- Power supply



Fig. 3: Female DC Power Jack connector

| PRI port connector - Silvus radio | | Power connector |
|-----------------------------------|--------|-----------------|
| PIN N° | Signal | Signal |
| 2 | GND IN | Power - |
| 3 | VCC IN | Power + |

- Ethernet

Fig. 4: RJ45 pinout T-568B

| PRI port connector - Silvus radio | | RJ45 Connector (T-568B) | | |
|-----------------------------------|-----------------|-------------------------|--------|--------------|
| PIN N° | Signal | PIN N° | Signal | Color |
| 4 | ETH0_MX2N (RX-) | 6 | RX- | Green |
| 5 | ETH0_MX2P (RX+) | 3 | RX+ | Green-White |
| 6 | ETH0_MX1P (TX+) | 1 | TX+ | Orange-White |
| 10 | ETH0_MX1N (TX-) | 2 | TX- | Orange |

- RS-232

The RS-232 from the PRI cable should be connected to the RS-232 of Veronte harness.

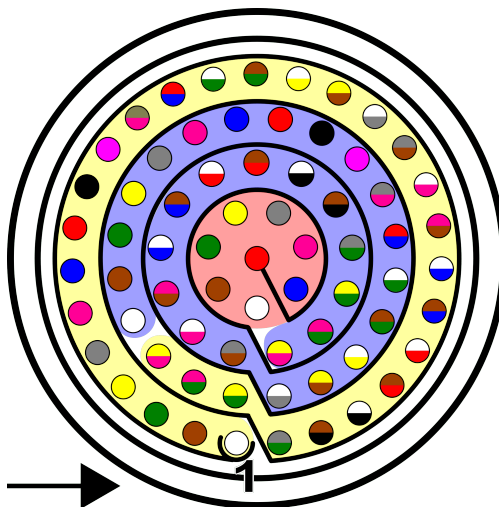


Fig. 5: CS harness plug

| PRI port connector - Silvus radio | | CS harness - Veronte | | |
|-----------------------------------|-----------|----------------------|-----------|------------|
| PIN N° | Signal | PIN N° | Signal | Color |
| 7 | RS232_RXD | 19 | RS 232 TX | White-Pink |
| 8 | RS232_TXD | 20 | RS 232 RX | Pink-Brown |
| 9 | GND | 21 | GND | White-Blue |

17.5.3 Silvus radio configuration

This section shows a basic configuration of the Silvus radio.

17.5.3.1 First Steps

Follow the steps below to set up the radio

1. Connect antennas (or attenuators) with male TNC ends to 2 RF ports.
2. Connect power supply to power port on PRI cable.
3. Connect non-forked female side of PRI cable to radio's PRI port.

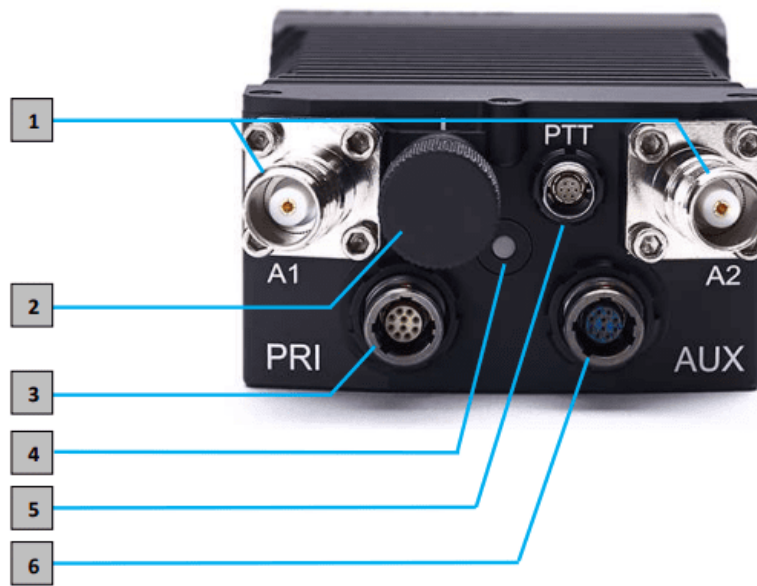


Figure 3 StreamCaster 4200E Ruggedized Enclosure

- 1 RF Channels 1-2 Connectors [TNC Female]
- 2 Power Switch [15-Position Rotating]
- 3 Power (EB Version Only, 9-20V), Ethernet, and Serial Port Connector [ODU GK0YAR-P10UC00-000L]
- 4 Bi-Color Status LED
 - Red – Radio is in the process of booting up
 - Flashing Green – Radio is fully booted but not wirelessly connected to any other radio
 - Green – Radio is wirelessly connected to at least one other radio

Fig. 6: Silvus connectors

4. When looking at the rotary multi position switch from the top, pull the knob towards you while rotating the knob towards the 1 position. This turns radio on. LED indicator will turn to fix red.
5. In order to access the StreamScape graphical user interface (GUI), connect Ethernet (RJ45) connector of PRI cable to Ethernet port of laptop/computer.
6. Make sure computer is set to static IP address on same subnet as radio.
 - a. Open network and sharing menu and click change adapter settings.

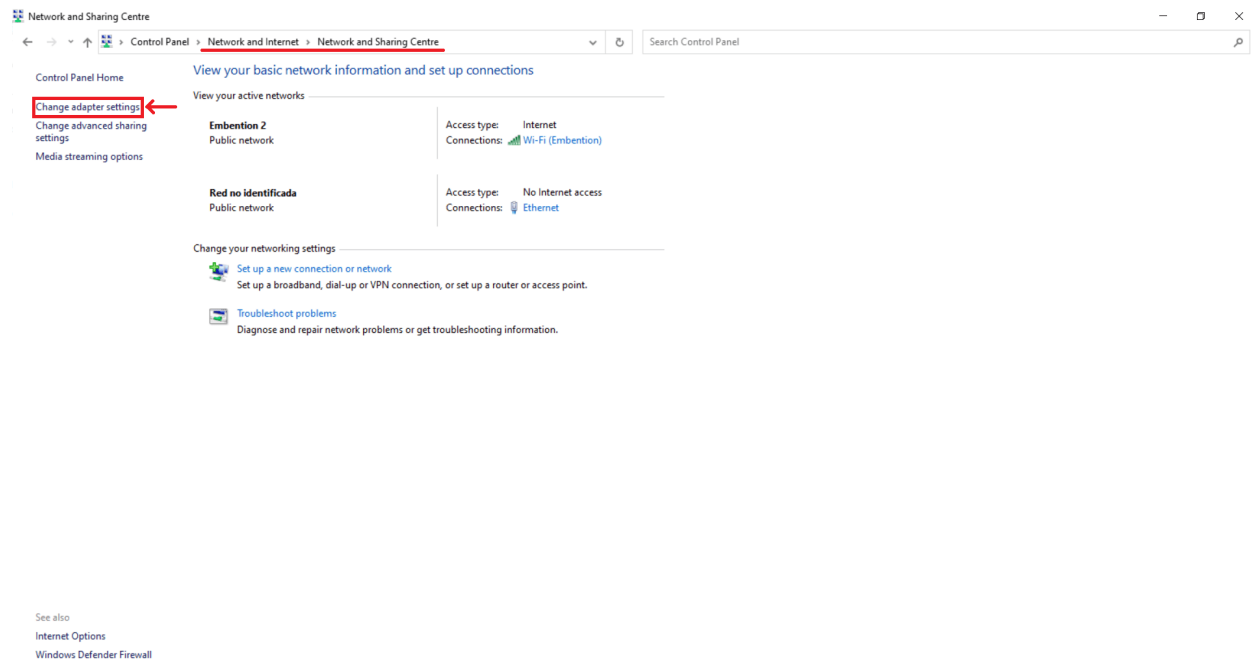


Fig. 7: Ethernet connection 1

- b. Select Local Area Connection, right click, and select properties.

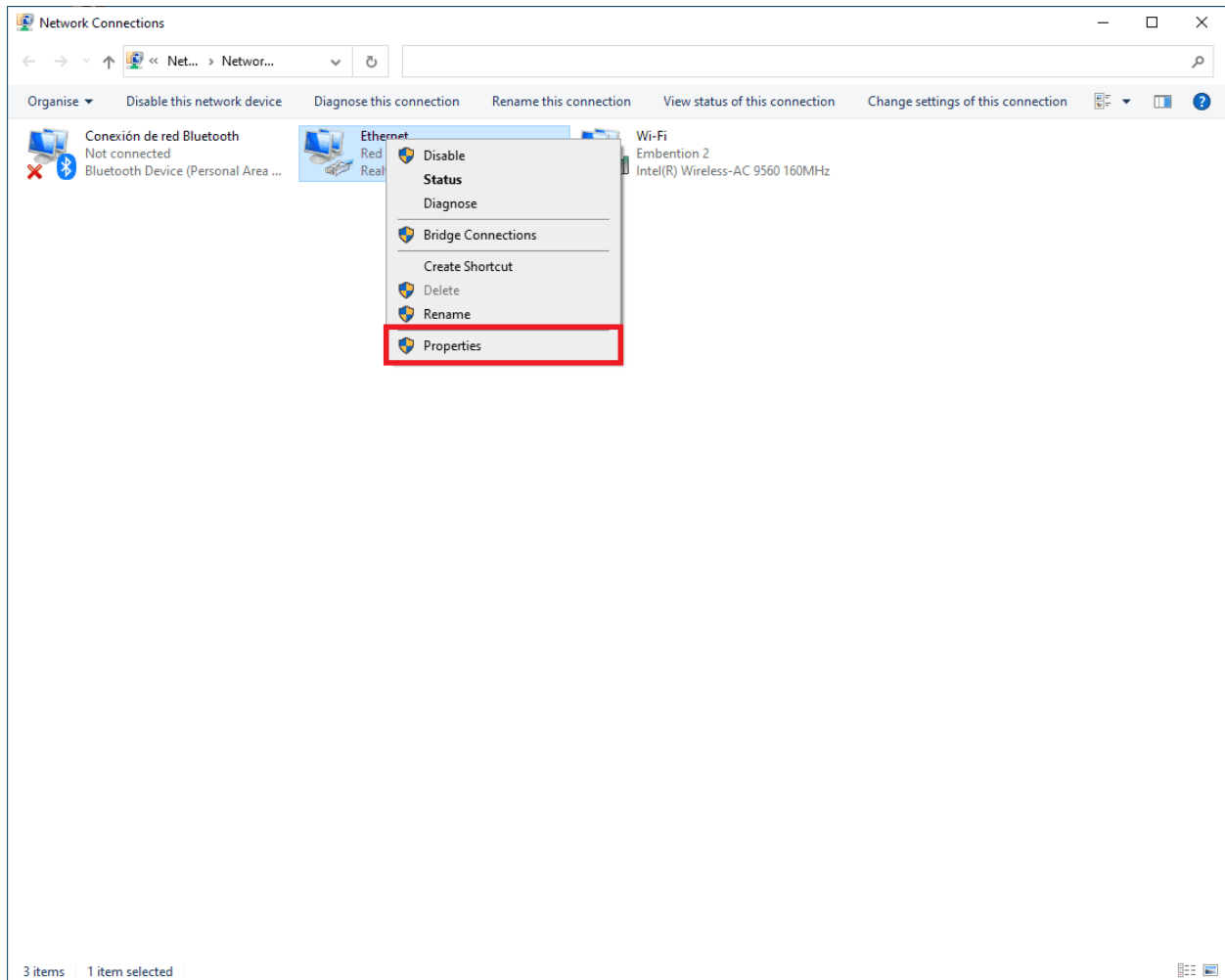


Fig. 8: Ethernet connection 2

c. Select IPv4 and click properties.

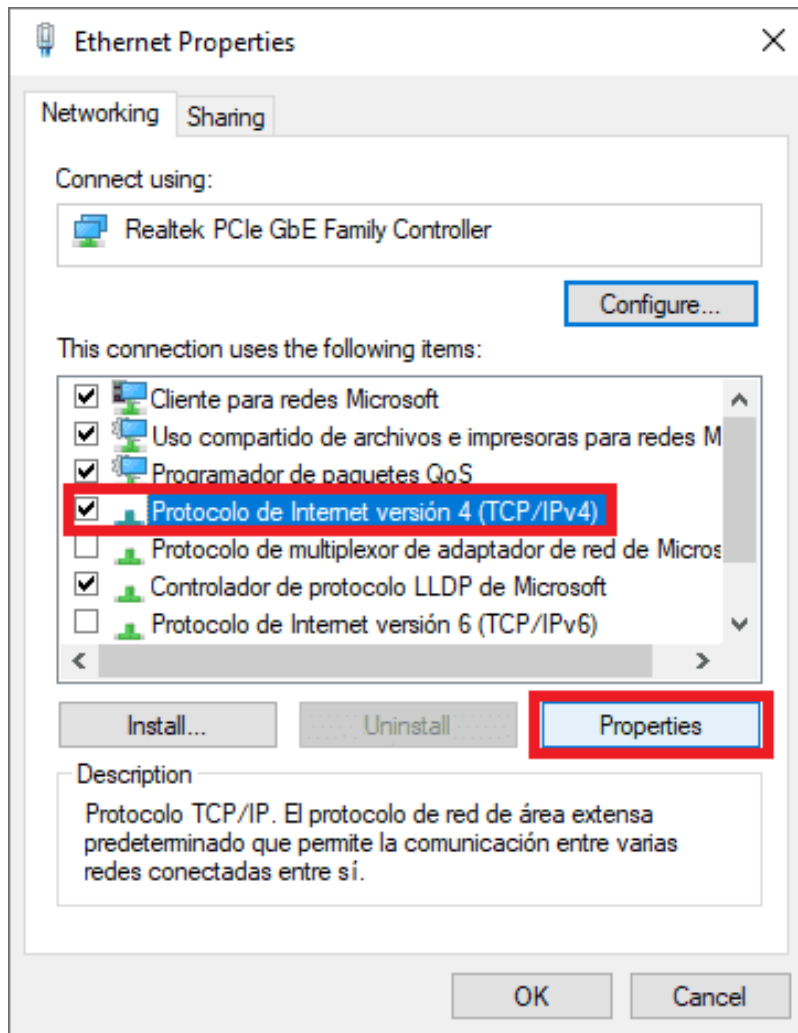


Fig. 9: Ethernet connection 3

- d. Set IP address to 172.20.XX.YY (e.g. if the IP of the radio is 172.20.178.203, set the IP 172.20.178.200) and subnet mask to 255.255.0.0. Click OK.

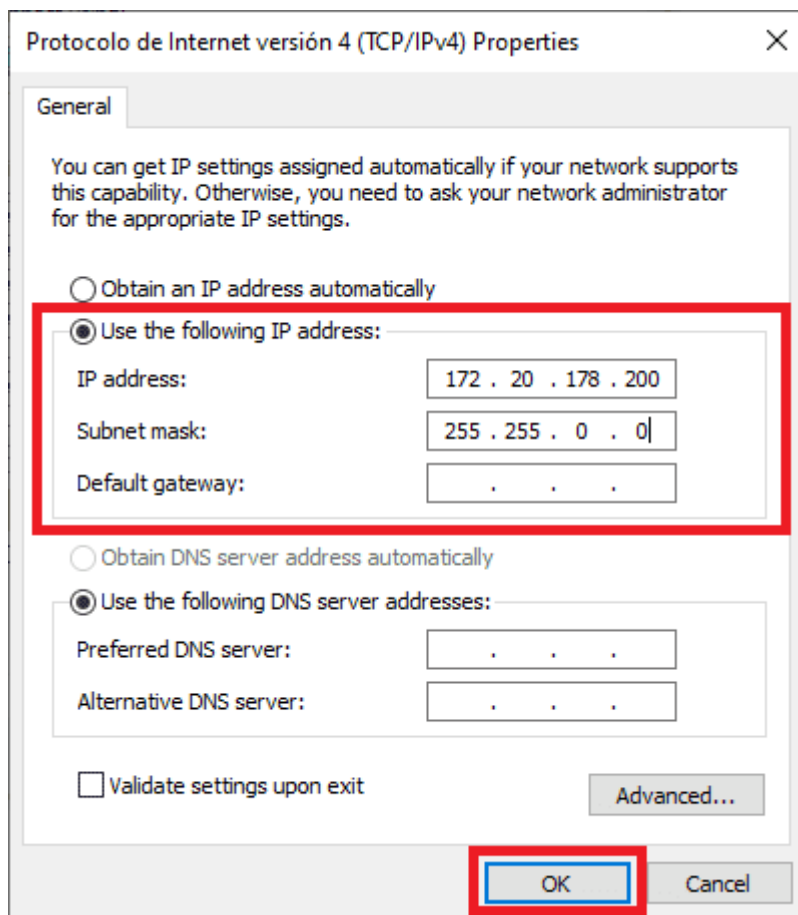


Fig. 10: Ethernet connection 4

7. Wait for LED indicator to turn to blinking green.
8. Access StreamScape GUI in web browser. To access, enter IP address of radio into web browser and press enter.

Note: Latest version of Firefox or Google Chrome preferred. IE or others not recommended.



Fig. 11: Silvus initial menu

9. User manual can be accessed by clicking the book icon in the GUI (Next to “Basic Configuration” in above screenshot.)

17.5.3.2 Basic radio configuration

Once the website has been accessed, follow the steps below which show the parameters that need to be modified for correct operation and pairing of the radios.

Note: This is an example of the radio configuration linked to a Veronte air unit.

Note: After making changes to each window, it is important to click on “Save and apply”.

1. Basic Configuration

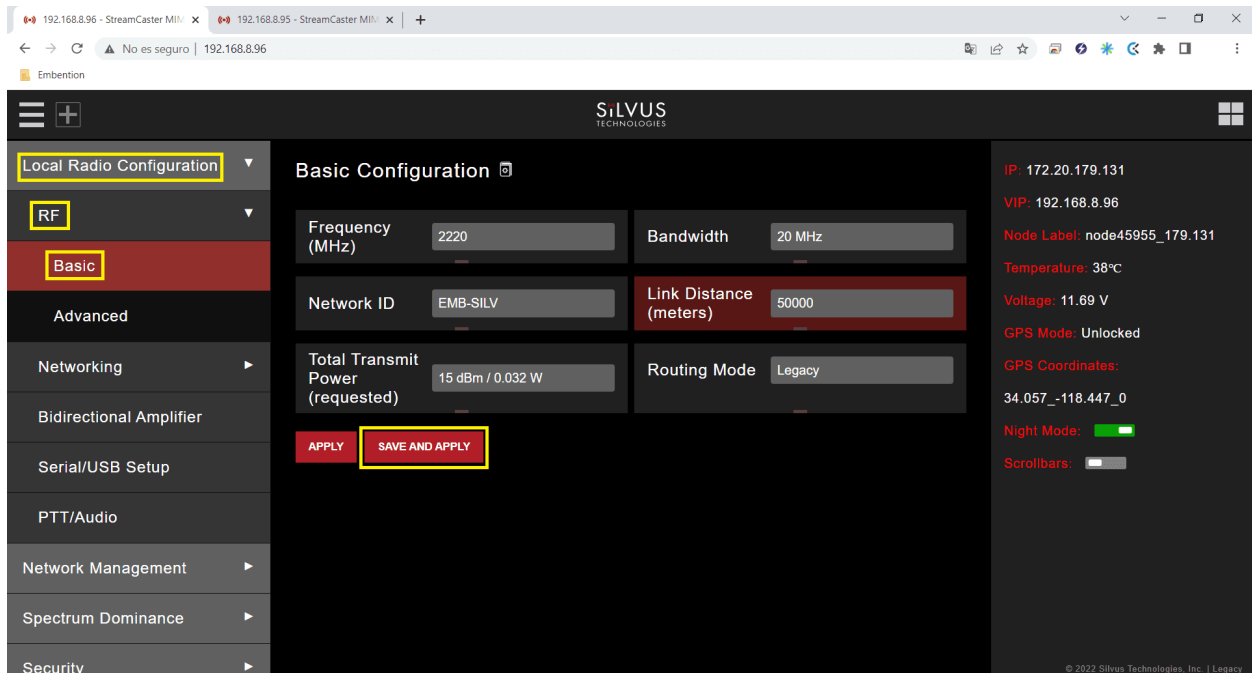


Fig. 12: Basic configuration panel

- **Frequency (MHZ):** This defines the frequency of the signal. There is a drop-down menu for frequency selection. We recommend 2220 MHz.

Warning: Be careful when choosing the frequency. The user may see interference with the Wifi frequency band, consult the radio spectrum.

- **Bandwidth:** This defines the RF bandwidth of the signal. Default value.
- **Network ID:** Network ID allows for clusters of radios to operate in the same channel, but remain independent. **A radio with a given Network ID will only communicate with other radios with the same Network ID.**
- **Link Distance (meters):** Set to an approximate maximum distance between any two nodes in meters. It is important to set the link distance to allow enough time for packets to propagate over the air. **It is recommended to set the link distance 10-15% greater than the actual maximum distance.**
- **Total Transmit Power (requested):** This defines the total power of the signal (power is divided equally between the radio antenna ports). Set the appropriate power for each application. The power that has been set is small, as

it is sufficient for our tests.

- **Routing Mode:** As *Large Network* mode requires a license and is not available outside USA, we set *Legacy* mode.

2. Advanced configuration

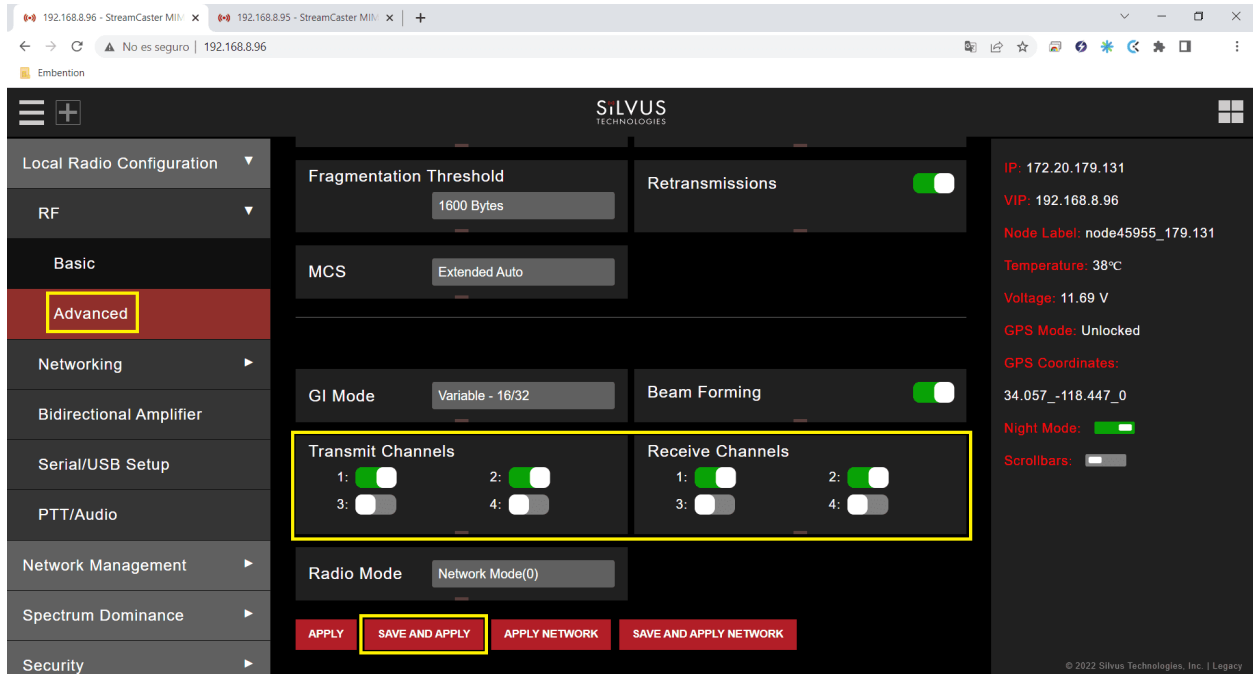


Fig. 13: Advanced configuration panel

- **Transmit/Receive Channels:** Allows user to Enable or Disable each channel on the radio for TX/RX (each RF port is a channel). We have enabled both channels.

3. Networking. Multicast

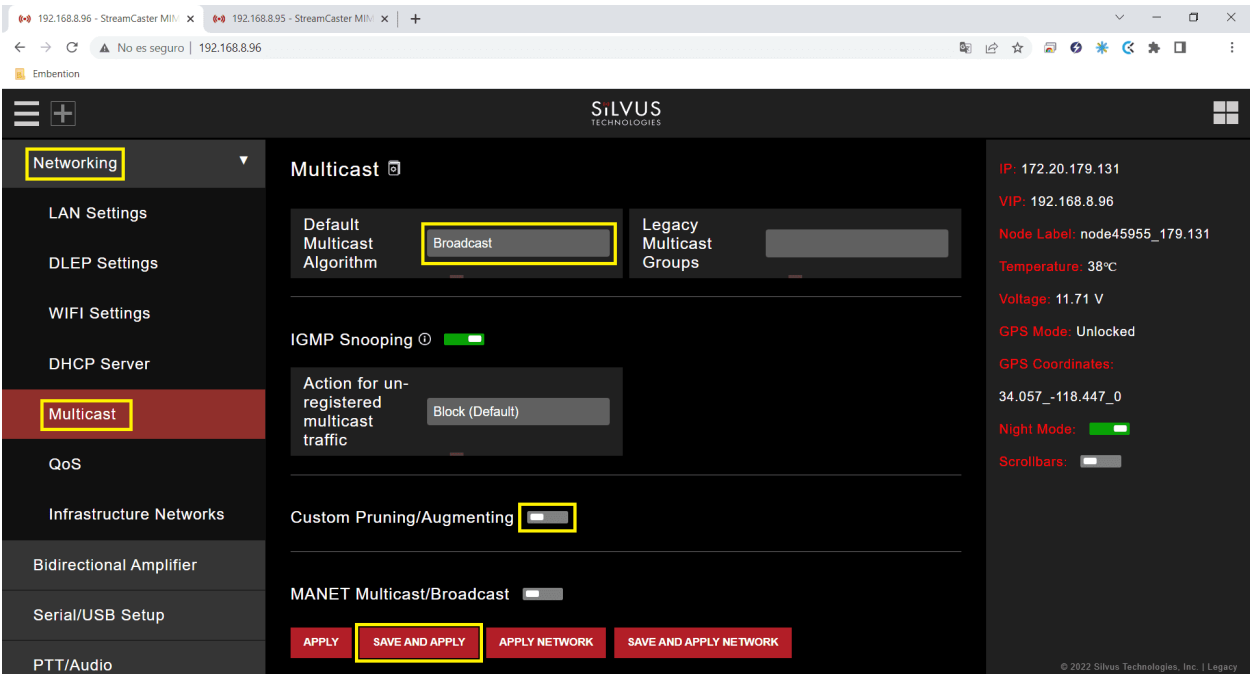


Fig. 14: Multicast panel

- **Default Multicast Algorithm:** Broadcast.
- **Custom Pruning/Augmenting:** Disable.

4. Serial/USB Setup

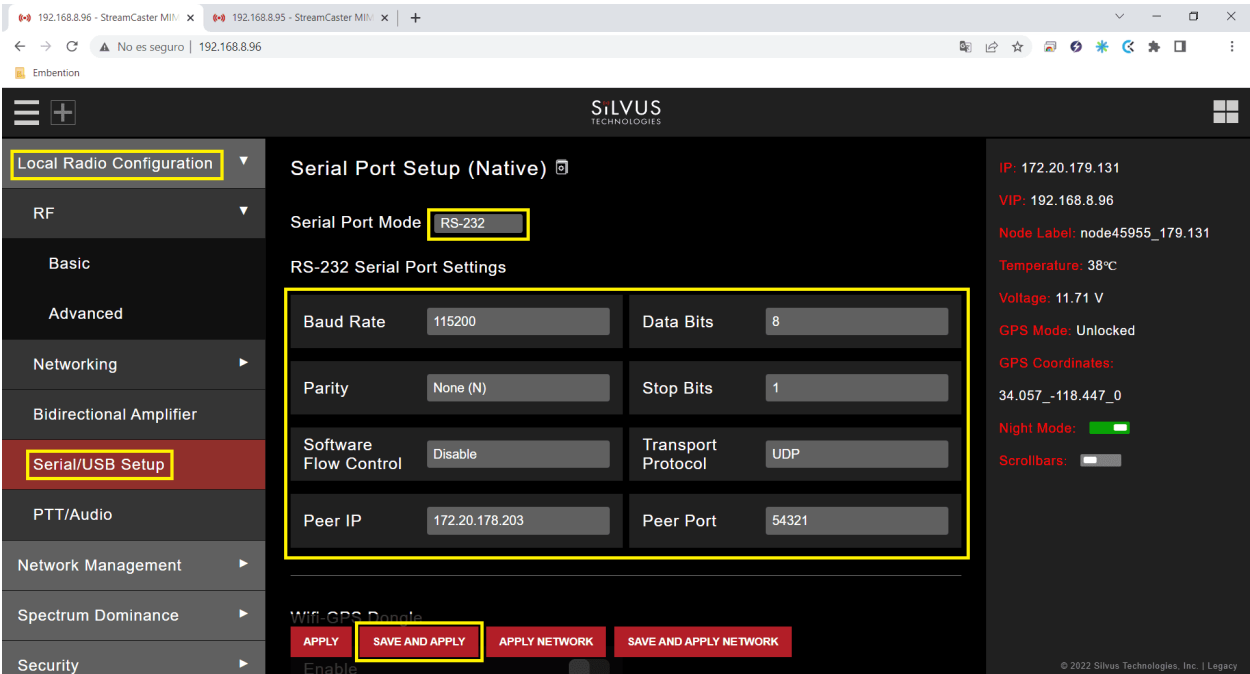


Fig. 15: RS-232 settings

- **Serial Port Setup:** RS-232.

- RS-232 Serial Port Settings

- The value of the **Baudrate**, **Data Bits**, **Parity** and **Stop Bits** parameters must be the same as those configured in Veronte Pipe.
- **Software Flow Control**: Disable.
- **Transport Protocol**: We recommend **UDP**. If no data loss can be tolerated, change this setting to TCP on the radio corresponding to the Veronte **air** unit.
- **Peer IP**: This should be the IP address of the radio on the other end of the RS-232. In this example, we must set the IP address of the radio linked to the GND unit.

Note: Both radios (the one connected to the GND unit and the one connected to the AIR unit) have the same configuration except for the **Peer IP**.

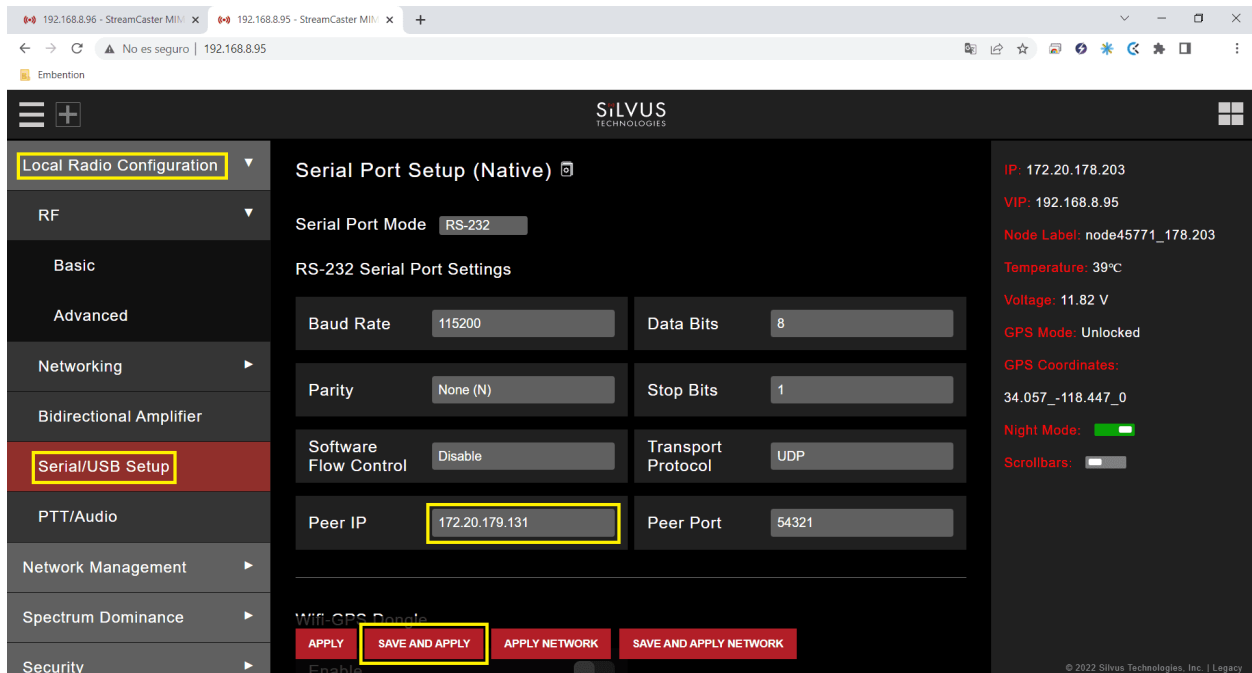


Fig. 16: Peer IP in radio linked to the GND unit

In addition to these settings, different configurations can be stored in the same radio, on the **Multi-Position Switch** panel. The user can select the one that will work, with the radio's switch position.

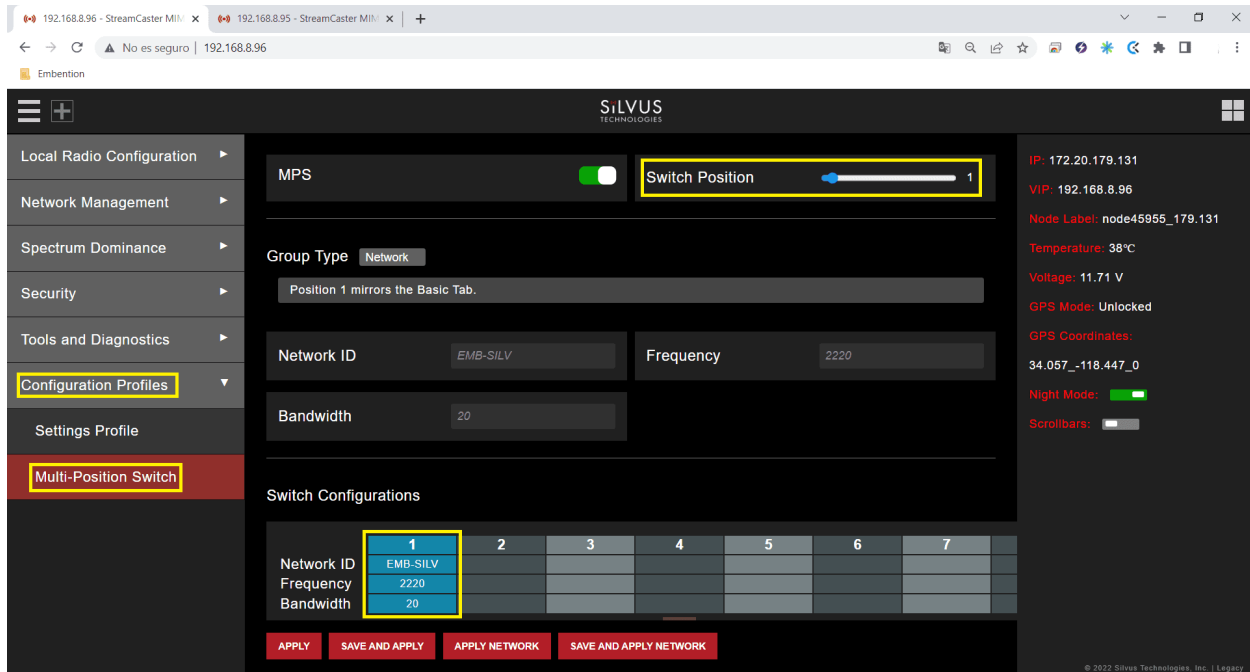


Fig. 17: Multi-Position Switch panel

In this example only one configuration has been created.

With the above settings the configuration is finished. Furthermore, this configuration can be saved and downloaded in the **Settings Profile** window of the Configuration Profiles section.

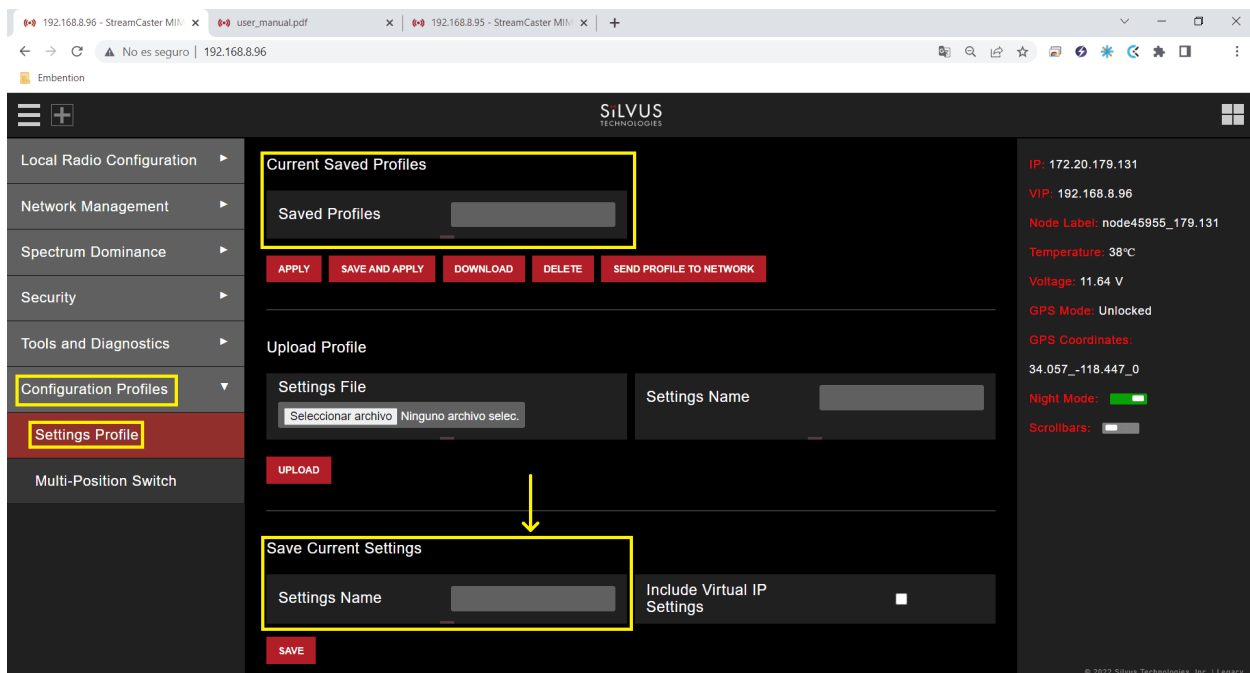


Fig. 18: Settings Profile panel

Before downloading the configuration it is necessary to save it.

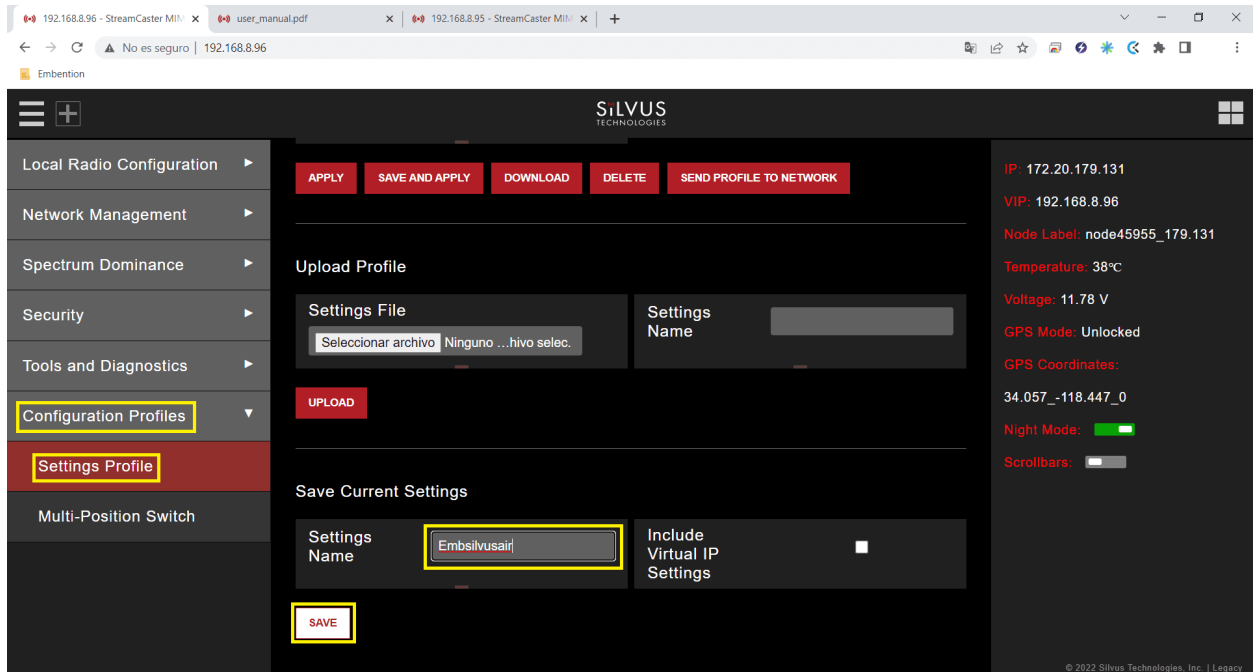


Fig. 19: Save settings

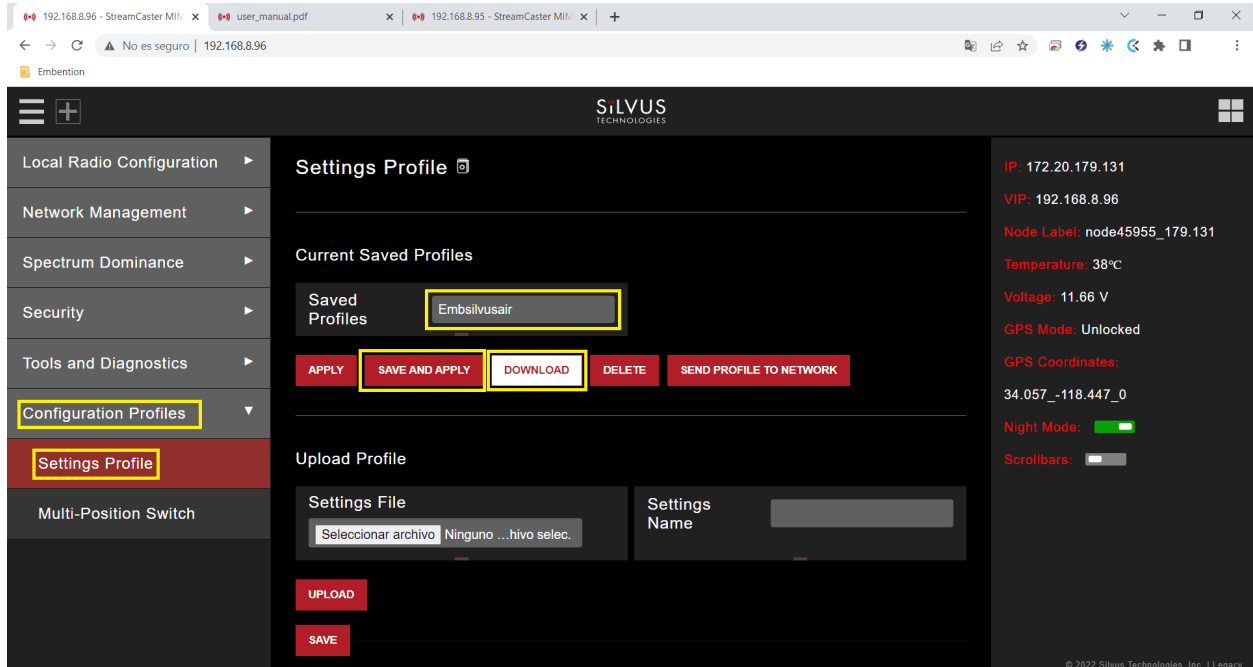


Fig. 20: Download settings

After configuring both radios with these settings they should be paired. Therefore, if we connect them to the power supply, when we switch them on, the LED will turn from fix red to fix green, this indicates that it is connected to at

least one radio. Also, if we connect only one of them to the computer, we can access the StreamScape GUI of both of them.

And, in the **Network Topology** window of the Network Management section, we can see the link between them.

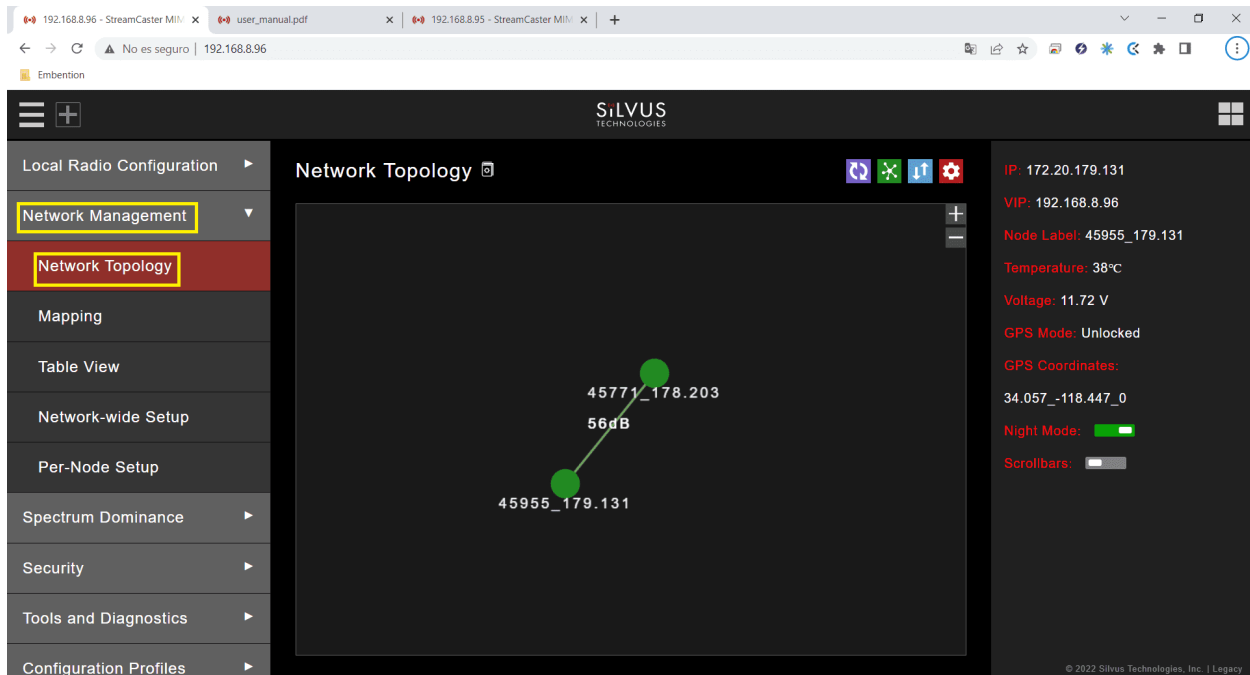


Fig. 21: Connection between radios

17.5.4 Silvus radio configuration in Veronte Pipe

The only configurations to be made in Veronte Pipe are the following:

1. Go to Connections -> Serial -> **232**

Check that they are the same as the parameter values previously set in the Silvus radio.

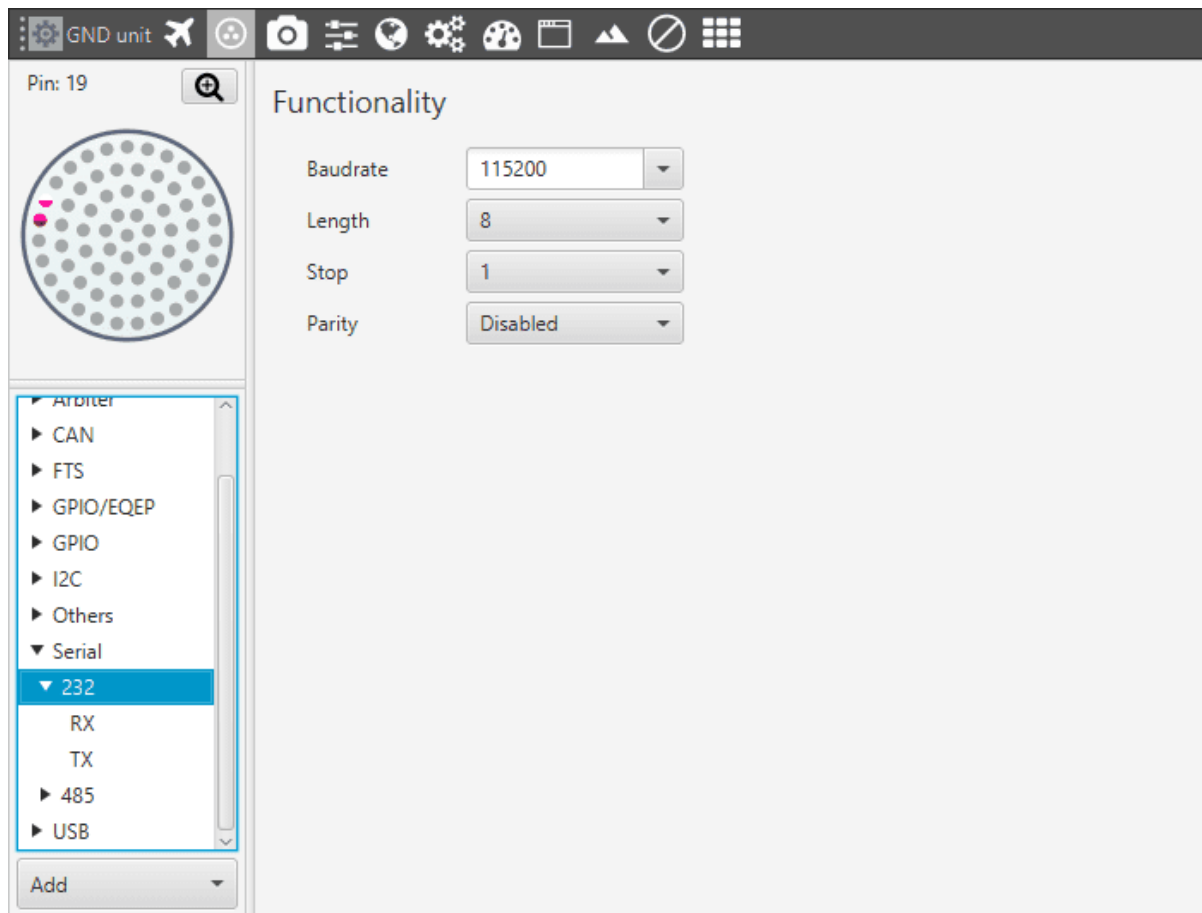


Fig. 22: RS-232 connection configuration

2. Go to Devices -> Others -> **I/O Manager**

RS-232 has to be configured as a bidirectional commgr port.

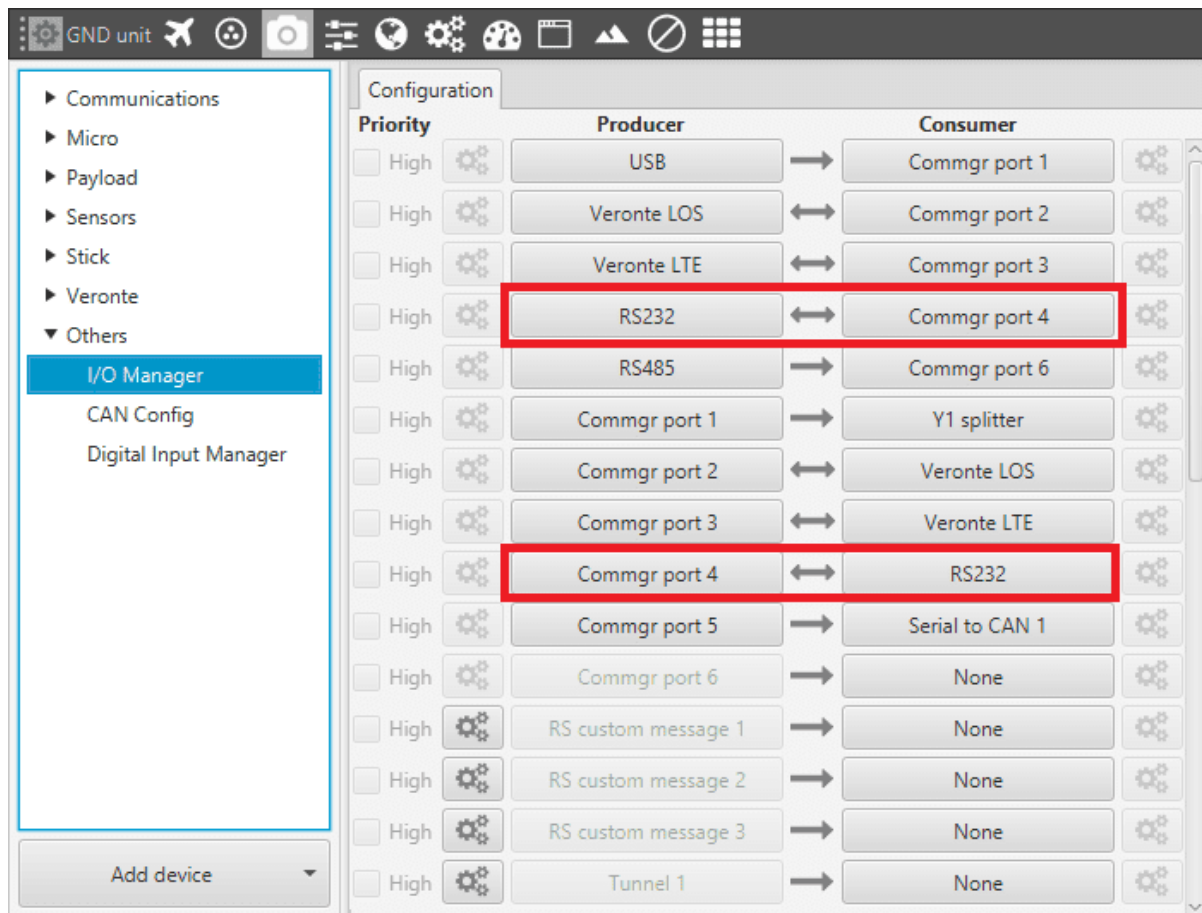


Fig. 23: RS-232 connection configuration

Note: These settings have to be made in both Veronte units (GND and AIR).







GARMIN®



SONY





4X REDUNDANT AUTOPILOT - ARBITER

18.1 Arbitration

The arbitration algorithm in **Veronte 4x** is based on a **scoring system**.

Each autopilot must send continuously a set of **Arbitration Variables** that will be used by the arbiter in order to calculate the score for each unit.

Then, based on the scores and the current **Arbitration mode**, the arbiter will choose to keep the current **Selected autopilot**, or switching to one of the other units.

Arbitration

18.1.1 Scoring System

A **Score** is a 32 bit, single precision, floating-point value with a range between **0** and **0xFFFFFFFF**, where **0** is a **Perfect score**. That is, **the lower the value, the better the score**.

Scores are calculated using the **Arbitration variables** received from each autopilot at their dedicated addresses. After receiving the value, each variable is multiplied by their respective **Arbitration weight**, and then it is used to compute the score for their respective unit.

Tip: The Arbitration weights should be used to increase or decrease the relevance that a certain arbitration variable has over the calculation of the score.

Any Variable in Veronte can be used as an arbitration variable. Depending on the platform, operation, application etc. the more relevant variables can be selected for its use as arbitration references.

Currently, the maximum amount of arbitration variables supported is **6**.

18.1.1.1 Absolute Arbitration Variables

Absolute arbitration variables are indicators that are inherently **good** or **bad**, and so they are added directly to the score.

Examples of absolute arbitration variables are **Link Quality**, **GNSS accuracy** or warnings such as **Sensors error** or **Position not fixed**.

18.1.1.2 Relative Arbitration Variables

Relative arbitration variables are not inherently good or bad, and hence need to be compared against the other autopilots in order to calculate its score contribution.

The contribution to the score from a relative arbitration variable will be its **Deviation** from the **Average** of the same variable from each autopilot.

Examples of relative arbitration variables are **Attitude**, **Position**, measurements from sensors, etc.

Lets see a practical example:

| Autopilot | Var. N° | Veronte Variable | Type | Value | Mean | Deviation | Weight | Score | Total Score |
|-----------|---------|------------------|----------|-------|-------|-----------|--------|-------|-------------|
| 1 | 1 | Roll | Relative | 0.12 | 0.096 | 0.024 | 1 | 0.024 | 0.058 |
| | 2 | Pitch | Relative | 0.30 | 0.283 | 0.017 | 1 | 0.017 | |
| | 3 | GNSS Accuracy | Absolute | 1.7 | | | 0.01 | 0.017 | |
| 2 | 1 | Roll | Relative | 0.10 | 0.096 | 0.004 | 1 | 0.004 | 0.026 |
| | 2 | Pitch | Relative | 0.28 | 0.283 | 0.003 | 1 | 0.003 | |
| | 3 | GNSS Accuracy | Absolute | 1.9 | | | 0.01 | 0.019 | |
| 3 | 1 | Roll | Relative | 0.07 | 0.096 | 0.026 | 1 | 0.026 | 0.054 |
| | 2 | Pitch | Relative | 0.27 | 0.283 | 0.013 | 1 | 0.013 | |
| | 3 | GNSS Accuracy | Absolute | 1.5 | | | 0.01 | 0.015 | |

Arbitration Example

In the above example, AP2 is considered to be the **best**, since it has the **lowest score**. Vven though its **GNSS accuracy** is the worst of all 3, its values for pitch and roll are the ones with the lower deviation from the mean.

18.1.2 Arbitration modes & parameters

The following parameters will determine the selected autopilot at each moment depending on the scores

18.1.2.1 Arbitration modes

The main arbitration modes are the following:

- **Always best:** The selected autopilot is always the one with the best score.
- **Change if worst:** The arbiter will only switch if the currently selected autopilot has the worst score. In that case, it will switch to the one with the best score.
- **Fixed while ok:** This mode does not take into account scores. In this mode, the **Preferred** autopilot will be selected by default. A switch will only happen if the current autopilot is considered **Dead** (See **Alive Status** in *Arbiter Operation*).

Additionally, the following modes are also available:

- **Round Robin Control:** The arbiter will periodically switch between autopilots. This mode is meant for testing purposes only.
- **Fixed:** Arbitration is disabled and one autopilot is selected arbitrarily. In this mode 4x Veronte will behave as a 1x Veronte.

18.1.2.2 Arbitration parameters

- **Preferred autopilot:** The preferred autopilot will be chosen in case of a draw. **Fixed while ok** mode will always select this autopilot first.
- **Holding CAP time:** Amount of time needed from last switch in order to allow a new switch.
- **Hysteresis:** When comparing scores, the difference between them needs to be bigger than this value in order to consider a score better or worse than other. The difference is proportional to the score of the **selected autopilot**.

i.e. If current selected autopilot is AP1, arbitration mode is **Always best**, hysteresis is **0.5** and score for AP1 is **0.3**, AP2 will need a score lower than **0.15** in order to be selected.

18.2 Operation

18.2.1 Arbiter Boot

When the arbiter is booted, it enters an **Idle** state, trying to verify the status of the system.

While **Idle**, the **Status Message** is not sent.

If all checks success, the arbiter will enter **Normal mode**.

If, after **30 seconds**, system errors are still present, arbiter will enter **Maintenance mode**.

18.2.2 Status message

Once in **Normal mode**, the Arbiter will start sending the **Status** and **Scores** telemetry messages.

The frequency of these messages is configurable. They can also be disabled.

The status message contains information such as:

- Arbitration ON/OFF
- Selected autopilot
- System BITs
- etc.

While in **Maintenance Mode**, the Arbiter will send the **Status** message, even if it is disabled, but will not send the **Scores** messages.

18.2.3 Ready Status

After the Arbiter enters **Normal mode**, it will wait until a **Ready Message** from each autopilot is received. Only then the Arbitration will start.

Normal Arbitration start

If the Arbiter is in maintenance mode, the Arbitration **will not start**. Even if the **Ready messages** are received.

Failed Arbitration start

18.2.4 Alive Status

Once arbitration starts, all autopilots are declared as alive by default, but it is possible that they are declared as **Dead** if a critical error is found.

APs being declared Dead

The arbiter will declare an autopilot dead if one of the following incidences is found:

- One of the arbitration messages (including the Ready message) is not received for 0.1 seconds.
- A **Not Ready** message is received.
- A **System Not OK** error is raised on any of the autopilots, activating the **FTS signal**.
- The **watchdog** signal for any autopilot is not ok.

Attention: Make sure to configure the sending of arbitration messages as **High priority**. Otherwise, the sending of the messages could be shortly interrupted by a higher priority task and the autopilot will be declared **Dead**.

APs being declared Dead when arbitration starts due to missed messages

A **Dead** autopilot can never be selected again as long as arbitration persists.

The **Dead** status is not reversible, it is necessary to reboot the whole system in order to recover a **Dead** autopilot.

If the number of **Alive** autopilots is **2 or less**, relative arbitration variables are disabled (since at least 3 autopilots are needed).

If **only one** autopilot is **Alive**, it will be selected no matter the score.

If **all** autopilots are **Dead**, the **Preferred** autopilot will be selected.

18.2.5 Maintenance Mode

Maintenance mode is used for changing the Arbiter configuration.

Arbitration is disabled while in maintenance mode.

Arbiter will also enter maintenance Mode after a failed boot.

The reason of the failed boot can be checked in the **Status message**.

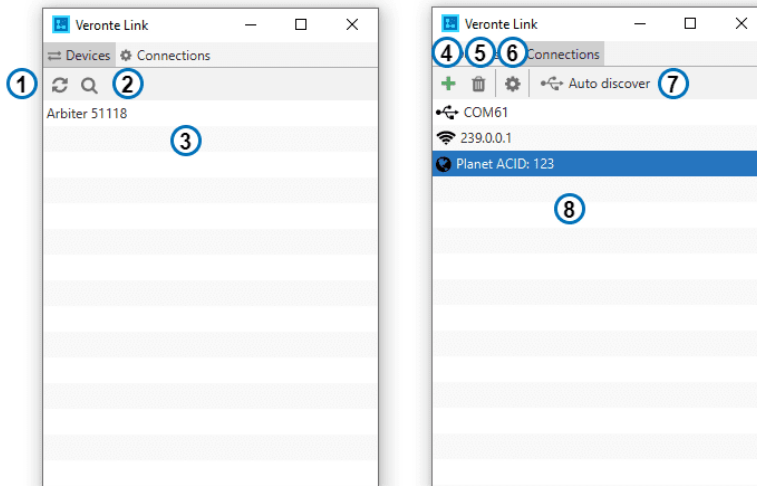
18.3 Software

18.3.1 Veronte Link

Veronte Link is the equivalent to the **Preferences/Connections** tab in **Veronte Pipe**.

It is used to open the **COM Ports** so that the 4x Arbiter can be detected by **4xVeronte PDI Builder**.

Attention: If **Veronte Pipe** has a certain **COM Port** open, Veronte Link will not be able to use it. If this happens, close Veronte Pipe and then restart Veronte Link



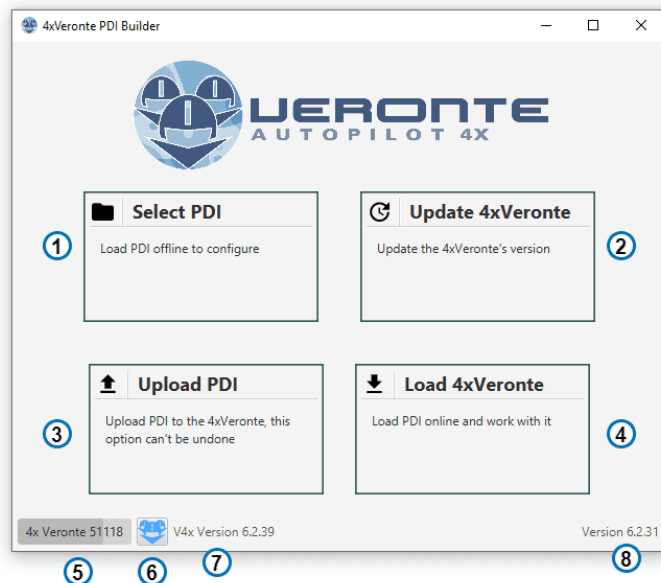
Veronte Link

| | |
|----|---------------------------------|
| 1. | Refresh devices |
| 2. | Add device |
| 3. | Connected devices |
| 4. | Add new connection |
| 5. | Remove selected connection |
| 6. | Edit selected Connections |
| 7. | Auto-Discover connections |
| 8. | Currently open connections list |



18.3.2 4xVeronte PDIBuilder

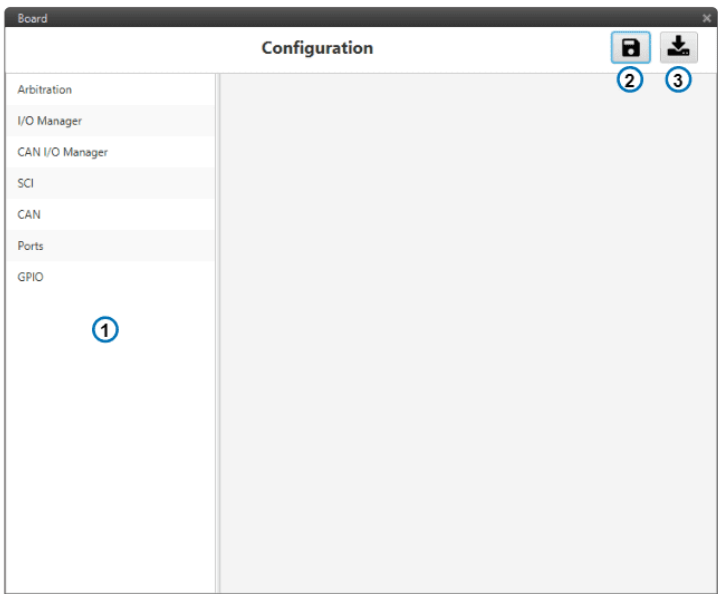
4xVeronte PDI Builder is the configuration interface for 4xVeronte Arbiter.

It can be used to perform firmware updates, load new configurations, or modify current configurations, both Offline and Online.



4x Veronte PDI Builder

| | |
|----|--|
| 1. | Open Arbiter PDI files Offline |
| 2. | Update selected Arbiter |
| 3. | Upload PDI Files to selected Arbiter |
| 4. | Open selected Arbiter PDI files |
| 5. | Connect to 4x Arbiter |
| 6. | Selected Arbiter mode. Toggle between  Normal and  Maintenance modes |
| 7. | Selected Arbiter Firmware version |
| 8. | 4xVeronte PDI Builder Software version |



4x Veronte PDI Builder - Configuration

| | |
|----|------------------------------|
| 1. | Configuration Tabs |
| 2. | Save PDI files to 4x Arbiter |
| 3. | Save PDI files to PC |

18.4 Update

In order to update 4x Arbiter:

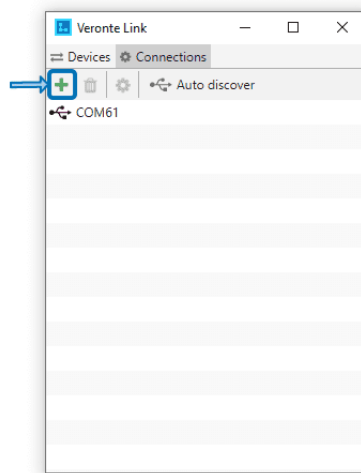
1. Open Veronte Link

Make sure to close first any Veronte Pipe instances.

2. Connect 4x Arbiter to the PC

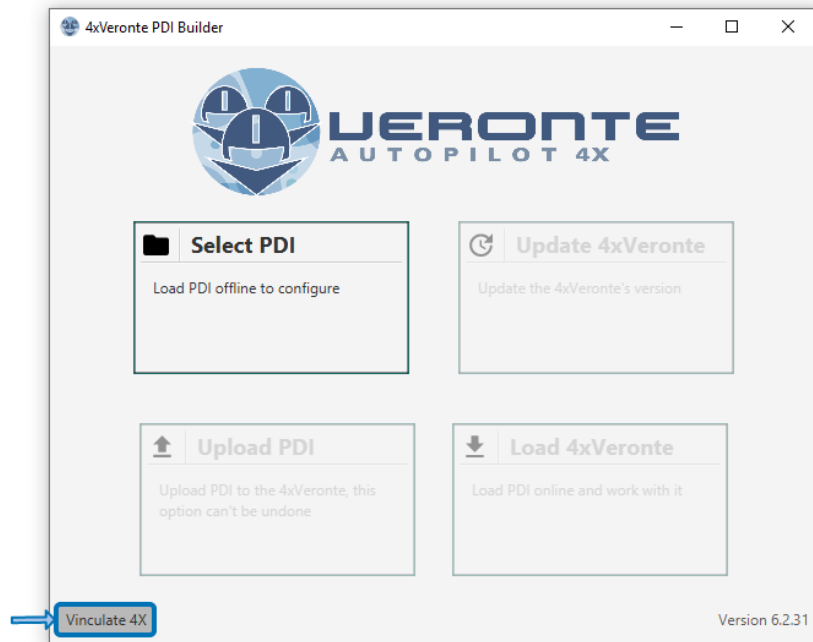
Connect it using the provided **4x Arbiter Harness** (Yellow connector) plus the 232 to USB converter. Make sure that 4x Veronte is correctly powered and that the COM Port is detected.

3. Identify the COM Port and add it to Veronte Link



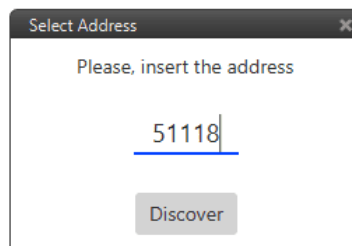
Veronte Link - Add COM

4. Open 4xVeronte PDI Builder and connect to the Arbiter



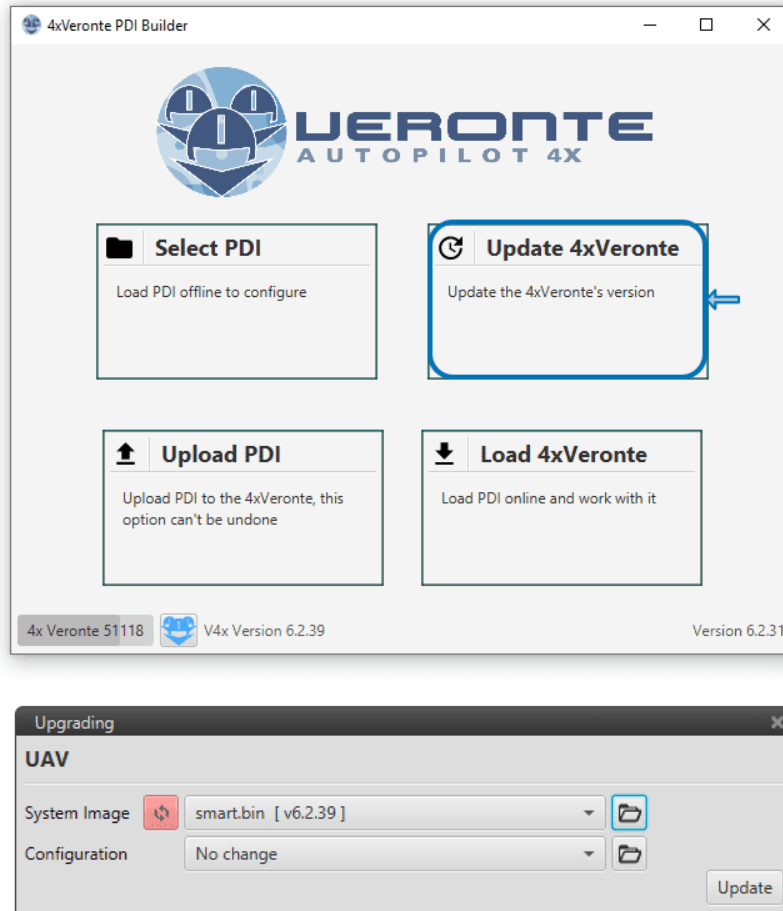
4xVeronte PDI Builder - Vinculate 4x

The **ID** for 4x Arbiter is its corresponding **4xVeronte S/N + 50000**



4xVeronte PDI Builder - Connect to Veronte 4x1118

5. Select Update 4xVeronte and load the new firmware version



Update 4x Veronte Arbiter

18.4.1 Updating 4xArbiter from Firmware version older than 6.0.0

After an update from a version older than **6.0.0**, the **Arbiter PDI files** will be empty.

Once the update is finished, it is necessary to, at least, load a set of **Default PDI Files**.

18.5 Configuration

In order to configure the **4x Arbiter**, connect it to **4xVeronte PDI Builder** and select **Load 4xVeronte**.

Alternatively, it is possible to modify PDI files Offline, and then upload them using **Upload PDI**.

Attention: If Veronte is in **Normal Mode**, it is possible that the **Load 4xVeronte** process takes longer than usual if arbitration is active.



Update 4x Veronte Arbiter

18.5.1 Arbitration

Configuration of the arbitration algorithm

18.5.1.1 Config

Update 4x Veronte Arbiter

| | |
|----|--|
| 1. | Preferred autopilot. Selected autopilot in case of a score draw |
| 2. | Arbitration method |
| 3. | Number of arbitration variables |
| 4. | Time delay before a new switch is possible after an AP switch |
| 5. | Score differential needed for a switch. This value represents a proportion of selected autopilot's score |
| 6. | Enable arbitration of external autopilot |
| 7. | Number of absolute variables. The asignation starts from the bottom. i.e. If there are 4 arbitration variables and 2 absolute variables, variables 0-1 will be relative and variables 2-3 will be absolute |
| 8. | Arbitration weights for each variable |

18.5.1.2 CAN Config

Board

Configuration

Arbitration

I/O Manager

CAN I/O Manager

SCI

CAN

Ports

GPIO

Config

CAN Config

☒ Send status

Period

0.2

☒ Send score

Period

1.0

Status message ID

255

Autopilot 0

☐ Extended

ID

8

Autopilot 1

☐ Extended

ID

9

Autopilot 2

☐ Extended

ID

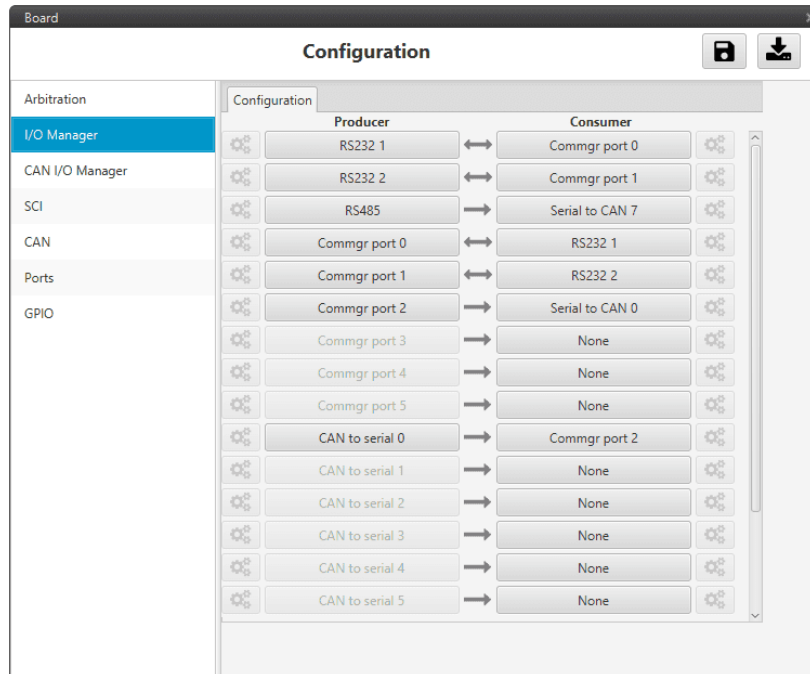
10

Update 4x Veronte Arbiter

| | |
|----|--|
| 1. | Enable and frequency of Status Message |
| 2. | Enable and frequency of Score messages |
| 3. | CAN ID at which Status and Score messages are sent |
| 4. | CAN ID at which Arbitration messages for AP1 must be sent |
| 5. | CAN ID at which Arbitration messages for AP2 must be sent |
| 6. | CAN ID at which Arbitration messages for AP3 must be sent |

18.5.2 I/O Manager

Configuration for serial interfaces



Update 4x Veronte Arbiter

18.5.2.1 Producers

| | |
|--------------------------|--|
| RS 232 1-2 | Serial RS232 Rx physical interfaces |
| RS 485 | Serial RS485 Rx physical interfaces |
| Commgr Port 1-5 | Configuration ports |
| CAN to Serial 0-7 | RS message over CAN output (Connected to CAN I/O Consumer) |
| CAN Wrapper 0-1 | CAN message over RS output (Connected to CAN I/O Consumer) |

18.5.2.2 Consumers

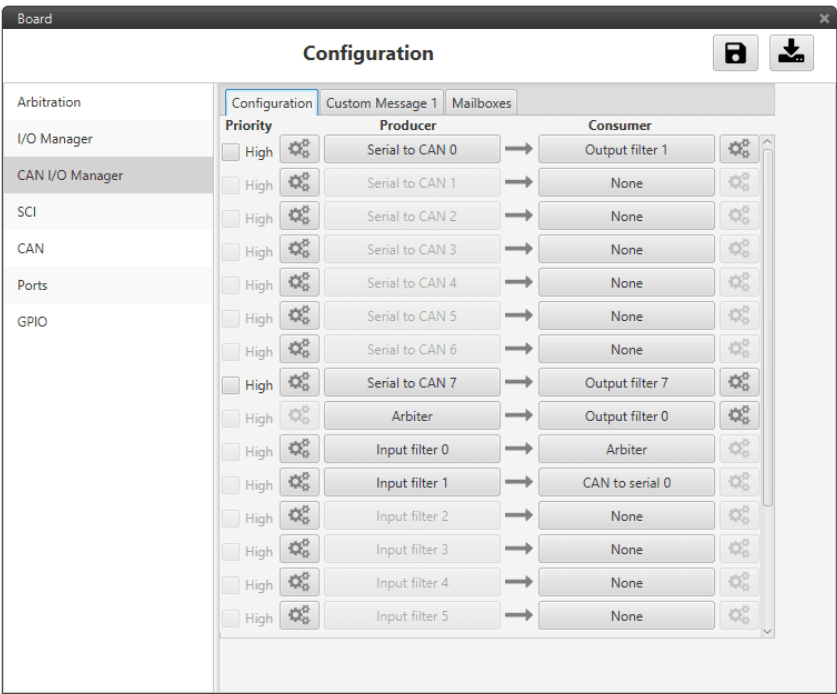
| | |
|--------------------------|--|
| RS 232 1-2 | Serial RS232 Tx physical interfaces |
| RS 485 | Serial RS485 Tx physical interfaces |
| Commgr Port 1-5 | Configuration ports |
| Serial to CAN 0-7 | RS messages over CAN input (Connected to CAN I/O Producer) |
| CAN unWrapper 0-1 | CAN messages over RS input (Connected to CAN I/O Producer) |

Warning: By default, **RS232 1-2** are set up for Arbiter configuration. If these connections are removed, it will not be possible to communicate with the arbiter in **Normal mode**. In this situation, force the Arbiter into **Maintenance mode** in order to recover the communication.

18.5.3 CAN I/O Manager

Configuration for CAN interfaces.

Mailbox and **Custom message** configuration are identical to Veronte Autopilot. For more information please check the related sections.



Update 4x Veronte Arbiter

18.5.3.1 CAN - Producers

| | |
|-------------------|---|
| Arbiter | Status and Score messages Output |
| Input Filter | CAN Input Filters |
| Serial to CAN 0-7 | RS messages over CAN output (Connected to I/O Manager Consumer) |
| CAN unWrapper 0-1 | CAN messages over RS output (Connected to I/O Manager Consumer) |
| Custom Message | CAN Custom Message 1 Tx |

18.5.3.2 CAN - Consumers

| | |
|--------------------------|---|
| Arbiter | Input for Arbitration messages |
| Output Filter | CAN Output Filters |
| CAN to Serial 0-7 | RS messages over CAN output (Connected to I/O Manager Producer) |
| CAN Wrapper 0-1 | CAN messages over RS output (Connected to I/O Manager Producer) |
| Custom Message | CAN Custom Message 1 Rx |
| GPIO Consumer | Consumer for GPIO control Messages |

| |
|--|
| Attention: The Input Filter connected to the Arbiter consumer must have a mask that allows all Arbitration CAN IDs |
|--|

18.5.4 SCI

Configuration of RS interfaces

18.5.5 CAN

Configuration of CAN Bus baudrates

18.5.6 PORTS

Commgr Port routing configuration

18.5.7 GPIO

GPIO pin configuration

18.6 CAN Protocol

CAN Payload protocol for arbitration messages.

Endianness is **Little Endian** for all messages.

CAN IDs are configurable for all messages.

18.6.1 From Arbiter

18.6.1.1 Status Message

| Byte | Position | Value | Description |
|------|----------|-------|--|
| 0 | 0 - 7 | 0x00 | Header |
| 1 | 0 - 7 | 0xFF | Status message Header |
| 2 | 0 - 6 | 0 - 3 | Selected AP (0 = AP1, 1 = AP2, 2 = AP3, 3 = External AP) |
| | 7 | 0 - 1 | Arbitration OFF/ON |
| 3 | 0 | 0 - 1 | AP1 Alive |
| | 1 | 0 - 1 | AP2 Alive |
| | 2 | 0 - 1 | AP3 Alive |
| | 3 | 0 - 1 | External AP Alive |
| | 4 | 0 - 1 | AP1 Ready |
| | 5 | 0 - 1 | AP2 Ready |
| | 6 | 0 - 1 | AP3 Ready |
| | 7 | 0 - 1 | External AP Ready |
| 4 | 0 | 0 - 1 | CBIT. System error. Will fail if any of the below items fail. |
| | 1 | 0 - 1 | PBIT. System Boot Ok. |
| | 2 | 0 - 1 | PDI Ok. Will fail if there is an error in configuration files. |
| | 3 | 0 - 1 | Memory Allocation OK |
| | 4 | 0 - 1 | CAN A Ok |
| | 5 | 0 - 1 | CAN B Ok |
| | 6 | 0 - 1 | CIO Low Task Ok |
| | 7 | 0 - 1 | CIO High Task Ok |
| 5 | 0 | 0 - 1 | Power OK. All power indicators are OK. |
| | 1 | 0 - 1 | A Bus Voltage Ok |
| | 2 | 0 - 1 | B Bus Voltage Ok |
| | 3 | 0 - 1 | Arbiter Voltage Ok |
| | 4 | 0 - 1 | AP1 Voltage Ok |
| | 5 | 0 - 1 | AP2 Voltage Ok |
| | 6 | 0 - 1 | AP3 Voltage Ok |
| | 7 | 0 - 1 | Arbiter Mode. 1 = Normal, 0 = Maintenance |

18.6.1.2 Score Message

| Byte | Position | Value | Description |
|-------|----------|--------------------|---|
| 0 | 0 - 7 | 0x00 | Header |
| 1 | 0 - 7 | 0 - 3 | Autopilot ID (0 = AP1, 1 = AP2, 2 = AP3, 3 = External AP) |
| 2 - 5 | 0 - 31 | 0 - 0xFFFF FFFF | Autopilot Arbitration Score |

18.6.2 From Veronte

Send to its corresponding CAN ID address.

Recommended addresses 8,9,10 & 11 (AP1,AP2,AP3, External AP).

18.6.2.1 Ready Message

| Byte | Position | Value | Description |
|------|-----------|-------|----------------------|
| 0 | 0 - 7 | 0x00 | Header |
| 1 | 0 - 7 | 0xFF | Ready Message Header |
| 2 | 0 (1 bit) | 0 - 1 | Ready/Not Ready |

18.6.2.2 Arbitration Messages

| Byte | Position | Value | Description |
|-------|----------|--------------------|-----------------------------|
| 0 | 0 - 7 | 0x00 | Header |
| 1 | 0 - 7 | 0 - 5 | Arbitration Variable Number |
| 2 - 5 | 0 - 31 | 0 - 0xFFFF FFFF | Arbitration Variable Value |



4x Redundant Autopilot

Note: This section makes reference to the **Arbiter** device in **4x Veronte**. For information about **4x Veronte** and its hardware please check [this section](#).

4xVeronte Autopilot is a triple redundant version of the Veronte Autopilot. It includes three complete Veronte Autopilot modules together with a dissimilar arbiter for detecting system failures and selecting the module in charge of the control.

All three modules are managed by a dissimilar microprocessor, the **Arbiter**. The **Arbiter** works in paralel to the Veronte units, monitors their status, and decides which one has access to the 4x connector based on a scoring algorithm.

The 4x Arbiter also includes a range of extra interfaces including **RS232**, **RS485**, **GPIO**, **ADC** and **CAN**, as well as the possibility of connecting a fourth autopilot to the arbitrating system.

4X REDUNDANT AUTOPILOT



4x Redundant Autopilot

19.1 Introduction

4xVeronte Autopilot is a triple redundant version of the Veronte Autopilot. It includes three complete Veronte Autopilot modules together with a dissimilar arbiter for detecting system failures and selecting the module in charge of the control. The autopilot selected as the master will be the one controlling the actuators and communicating with the payloads.

Each Veronte autopilot contains all the electronics and sensors in order to properly execute all the functions needed to control the UAV. Veronte executes in real time all the guidance, navigation and control algorithms for the carrying airframe, acting on the control surfaces and propulsion system and processing the signals from different sensors: accelerometers, gyroscopes, magnetometer, static pressure, dynamic pressure, GNSS and external sensors.

All three modules are managed by a dissimilar microprocessor. This arbiter includes voting algorithms for managing the module in charge of vehicle control. This microprocessor compares data from all modules in real time and processes

it for discarding any autopilot module showing an undesired performance.

19.2 Setup

19.2.1 CAN bus

4xVeronte Autopilot includes three complete Veronte Autopilot modules that have to be configured independently in order to work properly. 4xVeronte Autopilot is compatible with all configuration files from **Veronte Autopilot**. The communication channel between Veronte Autopilots modules and arbiter is done by using CAN bus so in order to start arbitring, Veronte Autopilots modules have to send the following information.

| Configuration | Value |
|---------------|---------------|
| Baudrate | 1 Mbps |
| Endianess | Little endian |

Each Veronte shall send using its CAN-TX ID and in little endian format.

19.2.1.1 Variable value message:

| | | |
|---------|----------------|-----------------------------------|
| Byte0 | 0 | Telemetry message |
| Byte1 | variable id | Variable id sent (0 to 127) |
| Byte2-5 | variable value | Variable value as Float (32 bits) |

19.2.1.2 Status message:

| | | |
|-------|--------|----------------------|
| Byte0 | 0 | Telemetry message |
| Byte1 | 0xFF | Status flag |
| Byte3 | bit0 | 1:Ready, 0:Not ready |
| | bit7-1 | Reserved |

By default the arbiter will send the following CAN messages.

19.2.1.3 Variable value message:

| | | |
|---------|-----------------|-----------------------------------|
| Byte0 | 0 | Telemetry message |
| Byte1 | N, Autopilot id | Autopilot [0, 3] |
| Byte2-5 | value | Autopilot score as Float(32 bits) |

19.2.1.4 Status message:

| | | |
|-------|--------|-----------------------------|
| Byte0 | 0 | Telemetry message |
| Byte1 | 0xFF | Status flag |
| Byte3 | bit6-0 | Selected autopilot |
| | bit7 | Arbitrating 0:false, 1:true |
| Byte4 | bit0 | AP0 Alive |
| | bit1 | AP1 Alive |
| | bit2 | AP2 Alive |
| | bit3 | AP3 Alive (external) |
| | bit4 | AP0 Ready |
| | bit5 | AP1 Ready |
| | bit6 | AP2 Ready |
| | bit7 | AP3 Ready (external) |
| Byte5 | bit0 | system ok |
| | bit1 | start |
| | bit2 | pdi |
| | bit3 | memory alloc |
| | bit4 | cana bus on |
| | bit5 | canb bus on |
| | bit6 | Reserved |
| | bit7 | Reserved |
| Byte6 | bit0 | system ok |
| | bit1 | Vcc A |
| | bit2 | Vcc B |
| | bit3 | Vcc arb |
| | bit4 | Vcc1 |
| | bit5 | Vcc2 |
| | bit6 | Vcc3 |
| | bit7 | Reserved |

19.2.2 Control

One modification has to be implemented in order to have smooth transition between Veronte Autopilots modules. This modification involves sending the control outputs of Veronte Autopilots modules to the rest of autopilots so it is considered in the transition. The user can refer to the configuration examples for this implementation.



Veronte autopilot

Veronte Autopilot is a miniaturized high reliability avionics system for advanced control of **unmanned systems**. This control system embeds a state-of-the-art suite of sensors and processors together with LOS and BLOS M2M datalink radio, all with reduced size and weight.

4X Veronte Autopilot is a **triple redundant** version of **Veronte Autopilot**. It includes three complete Veronte Autopilot modules fully integrated with a dissimilar arbiter for detecting system failures and selecting the module in charge of the control.

Veronte Pipe is the software designed for operating any Veronte powered platform. Users achieve a combination of easy-to-use application, real-time response and, firstly, safe operations.

The sections that follow this introduction explain in detail all the features of Veronte Pipe, the software used to control, configure and operate the platform fitted with the autopilot. Besides, the reader will find in [Hardware Installation](#) information about how to install both Veronte Autopilots in the aircraft (or the selected type of vehicle).