Veronte Ops

Release 7.8/1.2

Embention Sistemas Inteligentes, S.A.

2026-09-15

Contents

Scope of Changes 7
Quick Start 8
System requirements 8
Minimum 8
Recommended 8
Software Requirements 9
Download and Installation 9
Executable 10
Firewall Configuration 10
Additional apps 11
Veronte Terrain Provider 11
Download and Installation 11
Configuration 11
Operation 12
Web Converter 13
Download 13
Installation 13
Configuration 11
Operation 12
Panels 16
1. Veronte Ops configuration 16
Status bar 17
Notification 18
More options 20
Changelog 21
Light/Dark mode 22
Status bar settings 23
Setup 23
Preferences 23
Manage 35
Terms and Conditions 50
Log in 51
2. Mission 53

Operation 56
Mission Point 56
Waypoint 59
Segment 70
Orbit 74
Shape 77
Prism 80
Sphere 85
Cylinder 87
Geocaging area 90
Runway 99
Spot 102
Marks 106
Generate route 108
Spiral 108
Photogrammetry Mission 111
Figure 116
Mission panel 118
Edit mission 125
Turn on multiselection 126
Remove mission 128
Import features 129
Export features 129
Ruler 130
Turn on ruler 130
Paint 132
Enable paint 132
Delete paint 134
Offline 135
Show downloaded tiles 135
Save tiles 137
Remove tiles 138
3. Operation 139
Operation Panel 140
Customize 141

Operation Variables 142
Mission Points 145
Events Volumes 147
Events Waypoints 149
Action Waypoints 150
Events Marks 150
Runways 154
Spots 156
Initial position UAV 158
Operator position 159
Calibrations 160
Calibrate Atmosphere 161
Calibrate Wind 162
Advanced Calibrations 163
Complementary Telemetry 166
License 167
Stanag Scheduled Updates 168
Operations 169
Operation actions 171
4. Platform 172
Platform name 173
Platform connection status 174
Platform license 177
Platform actions 179
Autopilot 4x features 181
5. Workspace 183
Map options 184
Workspace options 192
Workspace toolbar 196
Workspaces 199
Widgets 200
Main 205
Map 205
Veronte Panel 206
Checklist 211

Group Conditional 215
Group button 222
Script 223
Safety 224
Veronte alerts 225
Flight Termination 232
Alerts 232
Electrical Report 233
Display 234
Text 234
Image 236
Iframe 238
Video 241
Timer 244
Telemetry 249
Label 249
Chart 253
Scatter 261
Gauge 268
Bar 277
Roll Tape 287
Label Script 294
Commands 296
Slider 296
Action Button 298
Input Data 301
Stick 303
Dial Button 308
Stick Buttons 311
Knob 313
Vehicle 320
Lights 320
Engine Command 321
Battery Report 322
Tanks Status 322

Vehicle Auxiliary 323
Flight instruments 324
Attitude 324
Heading 329
ADS-B 332
GPS Report 335
Wind 336
Payload 337
Gimbal Buttons 337
Gimbal Setup 339
Integration examples 343
Gimbal 343
Veronte Gimbal 345
Stick widget 349
USB joystick 350
Remote ID 352
Troubleshooting 354
Error when changing phase 354
Installed Veronte Ops app does not load 355
Navigation does not start 356
Sounds Notification Language 356
Appendix 357
Script Widget 357
Getting Started 357
Importing custom libraries 358
Stopping a Script 359
Available Scripting Functions 359
Example Use Cases 362
Error list 373
Notes and Tips 374
Safety Notes 374
📎 Summary Table 374
Software Changelog 376
7.8.24 376

Scope of Changes

- Version 1.0
  - Added:
    - First version issued
- Version 1.1
  - Added:
    - Explanation of the **Local (Default)** option for the "Network Interface" parameter in the CUCS - Setup menu
    - Warning message in Platform actions
    - Script widget explanation in **Appendix** section
    - Table with the URLs used by each available tile provider in Map options
- Version 1.2
  - Changed:
    - Workspace export/import file format from `.json` to `.vws` in Workspaces section

Quick Start

Veronte Ops is an application for the operation and monitoring of the vehicle during the mission.

Veronte Ops includes:

- **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**: Configuration of user-defined, pre-defined mapping and launch missions, as well as in-flight mission editing.
- **Operation parameters**: Configuration of operation parameters, such as theirs values, positions and directions, which may vary depending on the mission and during the mission.
- **Workspace**: User can get feedback on autopilot variables and status through widgets.
- **Multiple users**: One or more operators can work simultaneously.

System requirements

Before executing this software, users should check the following sections with the minimum and recommended PC hardware requirements.

Minimum

- **CPU**: Intel Core i5-8365UE
- **RAM**: 8 GB DDR4
- **STO**: 256 GB SSD

Recommended

- **CPU**: Intel Core i5-1235U (10C/DecaCore 1.3 / 4.4GHz, 10MB)
- **RAM**: 8 GB DDR4
- **STO**: 512 GB SSD M.2 NVMe PCIe

Software Requirements

Compatible browsers

The following are the functional, optional and unsupported browsers for use with **Veronte Ops** software:

- **Functional**: Chrome
- **Optional**: Firefox, Edge

Window Size

In order for the application and all its features to display correctly, check the following minimum and maximum sizes for the application window:

- **Minimum**: 1024x768
- **Maximum**: No limit

Operating System

- **Recommended**: Windows 11
- **Supported**: Windows 10

> **Important**
>
> Some of the latest features of this application may not be compatible with Windows 10.

Download and Installation

There are 2 ways to work with this application: as a **website app** or installing it through an **executable**.

> **Hint**
>
> The executable version might be useful for some users because:
>
> - No internet connection is required, i.e. for users who need to work offline.
> - No automatic updates are carried out.
> - Users do not need to have a browser installed.
> - The application does not depend on the browser.

Executable

If users wish to work with the executable app, **Veronte Ops** software is available on the **Veronte Toolbox** platform for downloaded and installation. For more information, please refer to the Veronte Toolbox user manual.

A personal account is required to access **Veronte Toolbox**; create a Ticket in the user's **Joint Collaboration Framework** and the support team will create it for you.

Firewall Configuration

Users must enable **UDP ports 51017 and 51018** (default ports) in the firewall to ensure the correct operation of the connection between **Veronte Ops** and **Veronte Link**.

> **Note**
>
> This configuration might need to be performed by the cybersecurity manager of your company.

Additional apps

Veronte Terrain Provider

Veronte Terrain Provider

Veronte Terrain Provider is an additional application that provides SRTM terrain data to **Veronte Ops**.

Thanks to this, **Veronte Ops** allows the user to visualize terrain heights on the created route. This makes it possible to foresee possible collisions of the aircraft with the terrain along the route.

Download and Installation

Veronte Terrain Provider software is available in the **Veronte Toolbox** platform. From there, users can download and install the application. For more information, please refer to the Veronte Toolbox user manual.

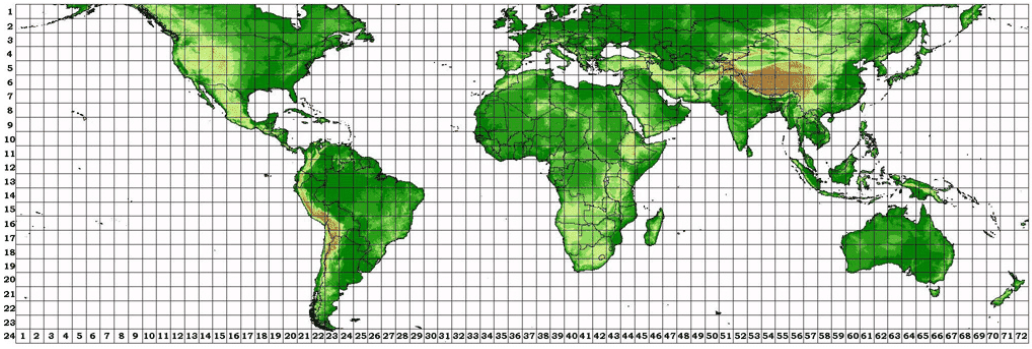
A personal account is required to access **Veronte Toolbox**; create a Ticket in the user's **Joint Collaboration Framework** and the support team will create it for you.

Configuration

It is possible to import external hgt files, which contain terrain information, and use them for mission preparations. These files follow the SRTM (Shuttle Radar Topography Mission) standard, with 3 arc-seconds (approximately 90 meters) of resolution. To import one of them, simply copy it into the directory:

`C:\Users\user\AppData\Roaming\VeronteTerrainProvider\terrainprovider`

Each hgt contains terrain information about one area of the world. The following image shows how these areas are distibuted, so each cell corresponds to one hgt:



Terrain information distribution

Operation

Once installed, users just need to run it. When it runs, it will always remain open in the background, which users can check in the PC's Windows taskbar.

Veronte Terrain Provider in task bar

It is **only necessary to run it the first time it is used**, then it should **always remain open** so that when the computer is switched on, it starts up as well.

> ⓘ **Note**
>
> If users close it, i.e. remove it from the taskbar, just run it again to start it.

Web Converter

Web Converter

Web Converter is a tool able to receive video in RTSP and serve it in webRTC and/or receive udp messages and send them via websocket.

Veronte Ops needs this application to **visualize video** and to work with **udp content** after it has been converted to websocket.

Therefore, it must be used when working with **gimbals** to visualize their video and click on them to send the **tracking command** to an object.

Download

Once a **Veronte Autopilot** has been purchased, a GitHub release should be created for the customer with the application.

To access to the release and download the software, read the Releases section of the **Joint Collaboration Framework** user manual.

Installation

To install **Web Converter** on Windows just execute "WebConverter.msi" and follow the indications of the setup wizard.

Configuration

Once the **Web Converter** software is downloaded and installed, configure it.

The configuration of this application is simply a `config.json` file with parameters for video display and command that need to be modified:

```
{
  "server": {
    "http_port": ":8083",
    "ice_servers": ["stun:stun.l.google.com:19302"]
  },
  "streams": {
    "veronte_Gimbal_Camera": {
      "on_demand": false,
      "url": "rtsp://192.168.8.94/net0"
    }
  },
  "udp_config": {
    "ip": "192.168.8.94",
    "port": 14003,
    "websocket_url": "localhost",
    "websocket_port": "1997"
  }
}
```

- `server` Users can modified the localhost port (http_port).
  The generated URL will look like this: `http://localhost"http_port"`

- `streams` Parameters to modify to **view the video recording** of the desired camera in **Veronte Ops**
  - `"camera name"`: Is to identify the gimbal camera. By default veronte_Gimbal_Camera is set.
    As the user can enter as many cameras to display the video as desired, each gimbal camera is configured here:
    - **url**: URL of the gimbal video. This is just an example, the URL (rtsp) depends on the user's gimbal.

- `udp_config` To **command** from **Veronte Ops** to the gimbal. Set the following parameters:
  - **ip**: **IP address** configured as the udp connection of the camera.
  - **port**: **Port** configured as the udp connection of the camera.
  - **websocket_url**: Websocket URL to which users want to send the converted udp messages.

- **websocket_port**: Websocket port to which users want to send the converted udp messages.

> ⓘ **Note**
>
> Even if users only want to view the video recording in **Veronte Ops**, they can leave the default configuration parameters of **upd_config**, as they do not affect the video configuration.
>
> However, they can also remove these lines from the configuration if they wish.

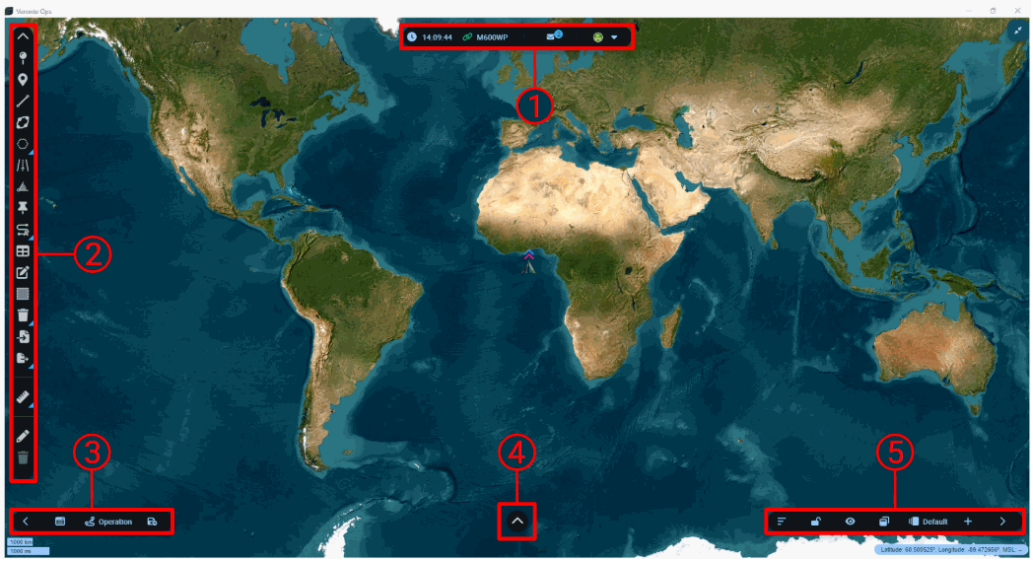
Operation

Once the **Web Converter** application has been configured according to the user's needs, follow the steps below to work with:

1. Just run the application and **always keep it open**.
2. Copy the link generated by the tool into the corresponding **Veronte Ops** configuration:
   - For **video recording**, copy the URL generated into the Iframe widget. For example, with the default **http_port**, the URL generated will be: `http://localhost:8083`.
   - For **command**, the user must copy the **websocket_url** and **websocket_port** previously configured into the **URL** and **Port** fields, respectively, of the **WebSocket** connection configuration of the Gimbal panel.

Panels

Panels on **Veronte Ops** are distributed as shown in the following figure:



Veronte Ops panels

1. Veronte Ops configuration: Edit **Veronte Ops** settings.
2. Mission: Create and edit **missions**.
3. Operation: Edit **operations** defined in **1x PDI Builder** and perform **calibrations**.
4. Platform: Veronte Autopilot information.
5. Workspace: Customize everything related to the **Veronte Ops** workspace, such as telemetry (widgets), map and workspace settings.

These panels are described in detail in the following sections.

In case of having any question about this **Joint Collaboration Framework**, please read its user manual.

1. Veronte Ops configuration

This section describes all the menus, options and actions that allow the user to **modify the configuration and appearance** of this application. These are:

- Status bar
- Feedback menu
- Platform icon

Status bar

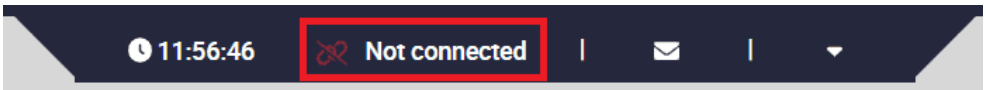
In the status bar, the user can view mission and operation relevant information, as well as modify general **Veronte Ops** settings.

Status bar

This bar is divided into 3 'parts':

- Status bar items: **Clock** and **Platform selected** information items are grouped here.
  The user can choose to show/hide each of these elements from the Status bar settings. By default, they are displayed as shown in the image above.

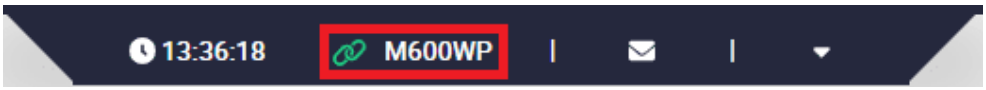
The figures below show all **Autopilot 1x** states:

- **Not connected**:
  

  Platform selected - Autopilot not connected

- **Not selected**:

  Platform selected - Autopilot not selected

- **Connected**:
  

  Platform selected - Autopilot connected

- **Maintenance mode**:

  Platform selected - Autopilot in maintenance mode

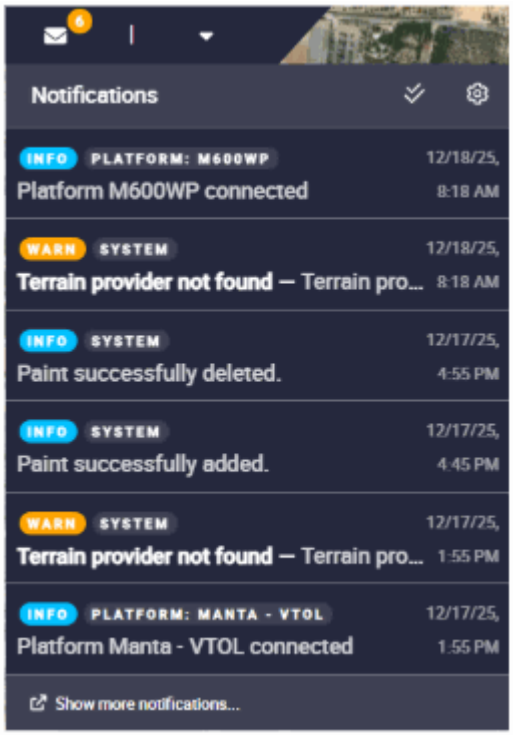
- **Disconnected**:

Platform selected - Autopilot disconnected

- Notifications: **Veronte Ops** notifications for users. This is always displayed in the status bar.
- More options: By clicking on this drop-down menu, users will find options for configuring this application and its appearance.

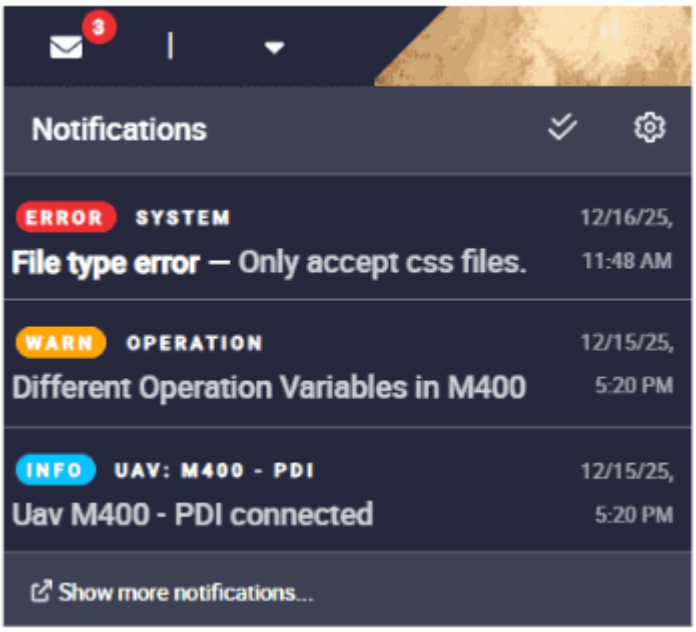
Notification

Veronte Ops will notify the user of any changes made with notifications (visual and audible, the latter is optional), as shown in the figure below:



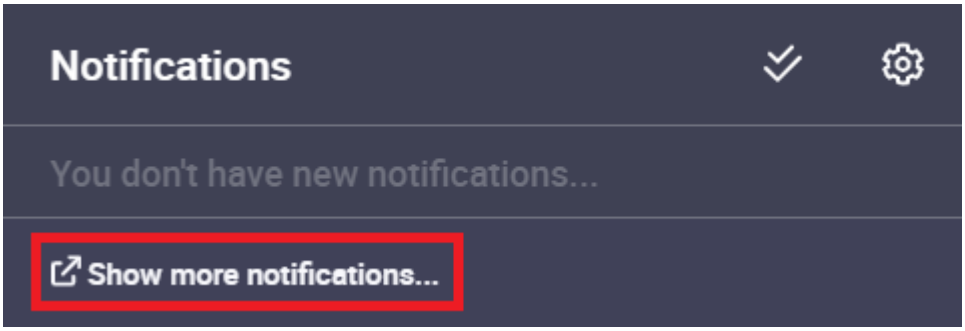
Notifications menu

In addition, notifications are classified into 3 groups (**Information**, **Warning** and **Error**) with a color code (**blue**, **orange** and **red** respectively):



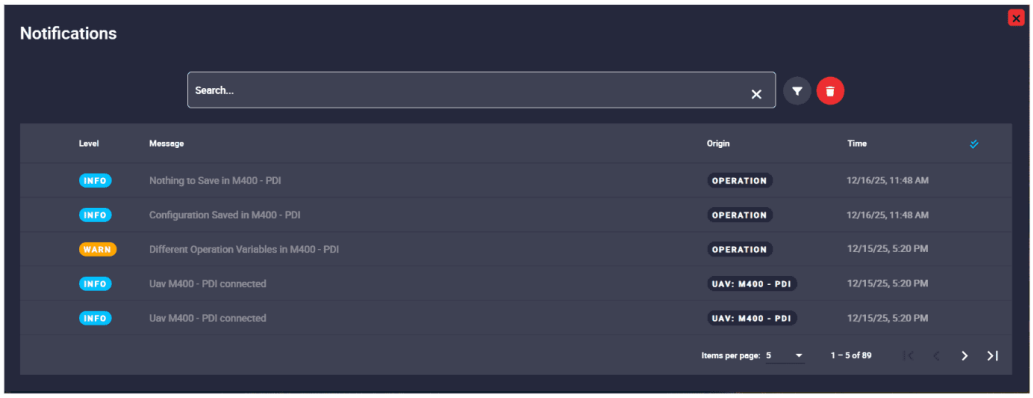
Notifications menu - Classification

These notifications can be **marked as read** by clicking on the icon at the top of this menu. So, they will no longer be displayed in this menu.



Notifications readed

However, if the user wishes to consult them, simply click on '**Show more notification...**' and a new window will appear with the entire history of notifications. Here the user can search for notifications or delete them:



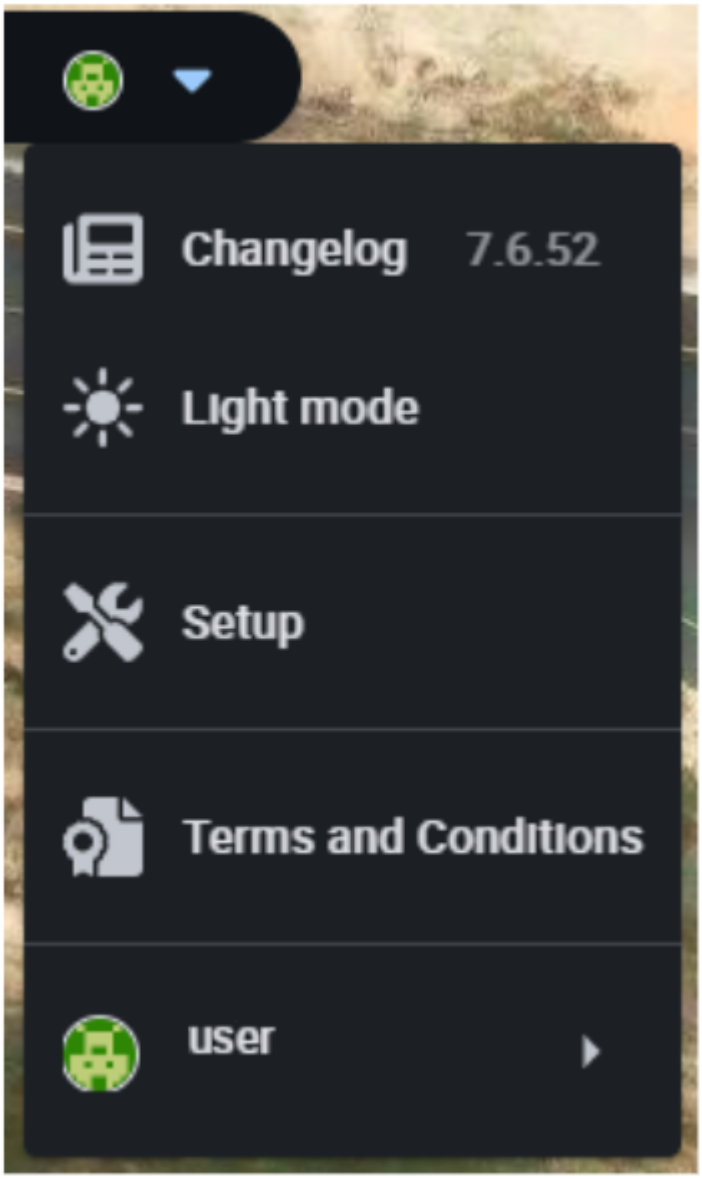
History of notifications

Besides, in the notification configuration users can choose, to some degree, which groups of notifications are shown or not. This is to a certain degree because, as the classification described above has been defined according to importance, **error notifications will always be shown** (as they are the most important ones) but warning and information notifications can be chosen to be seen/heard or not.

More options

Moreover, by clicking on the drop-down menu on the right, users will find more options, such as:

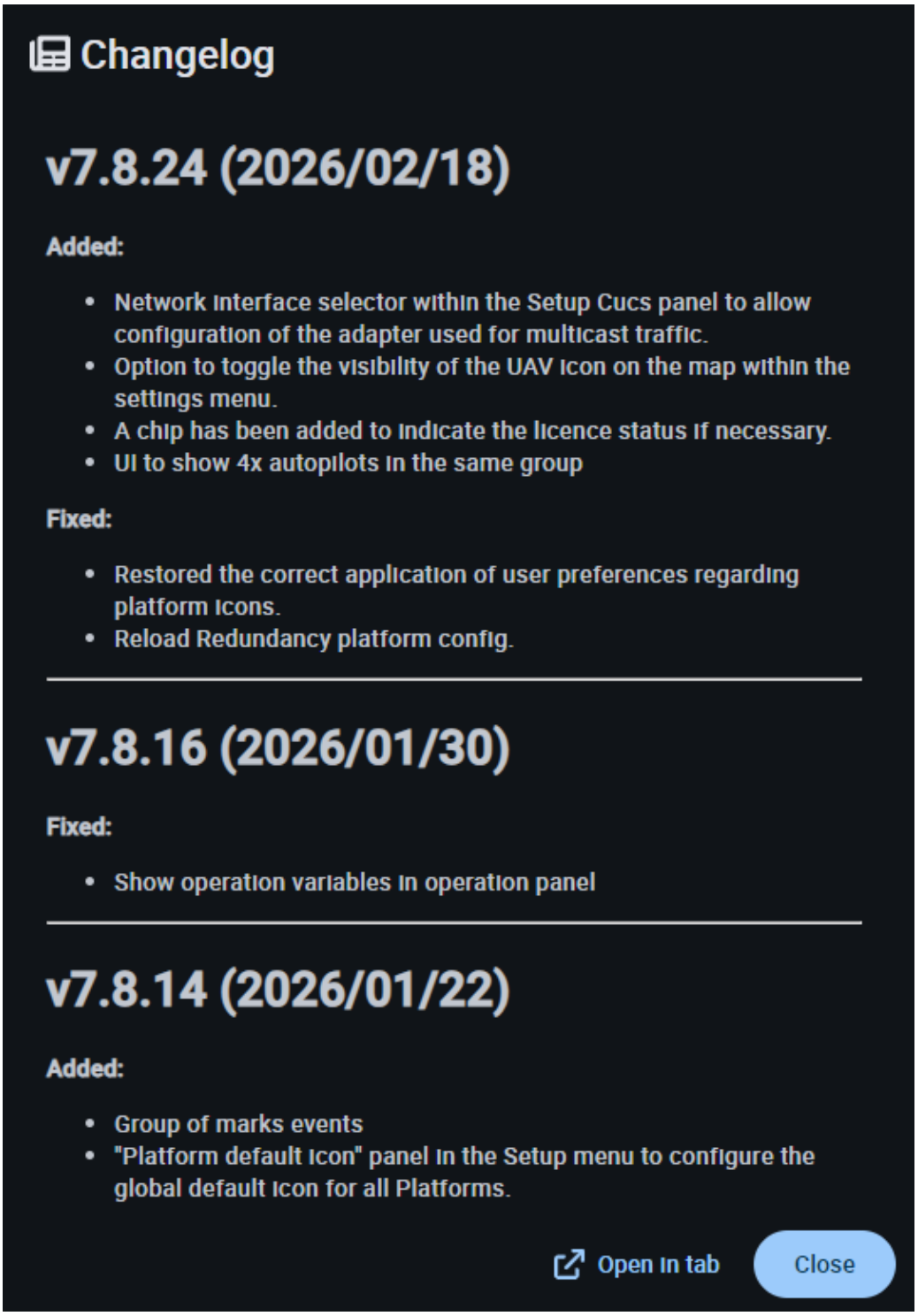
- Changelog
- Light/Dark mode
- Status bar settings
- Setup
- Terms and Conditions
- Log in



Drop-down menu

Changelog

Users can consult the changelog, which shows all updates made to the application.



Changelog

Light/Dark mode

Change the interface display mode: Light/Dark mode.

Light/Dark mode

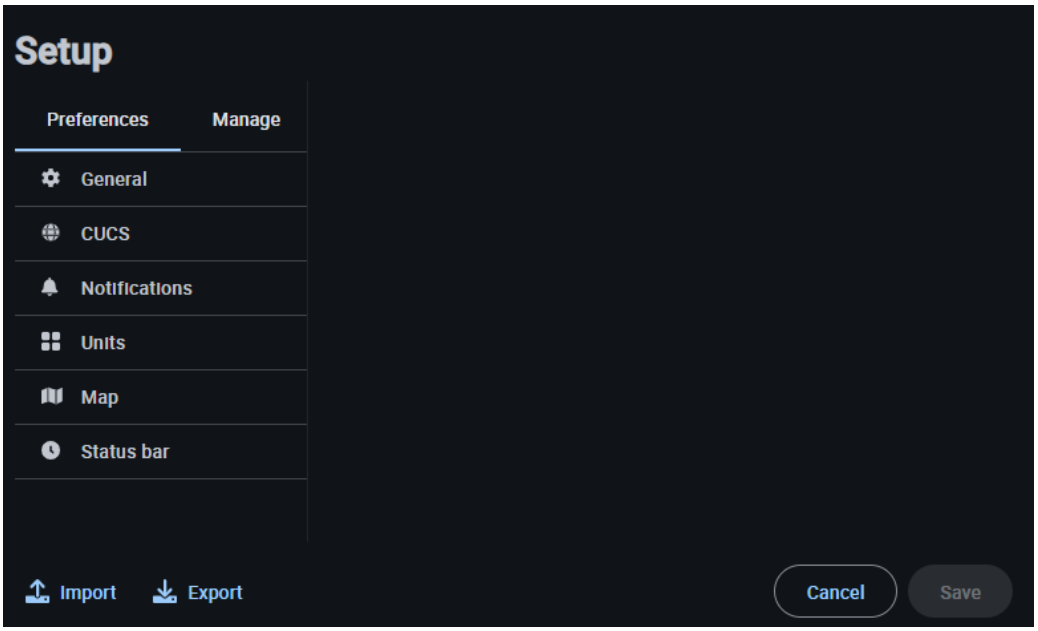
Status bar settings

This menu allows the user to **enable/disable** the information items in the status bar.

> ⓘ **Note**
>
> To save any changes, click the "Save" button in the bottom right corner of the menu.

Setup

This menu allows to edit the general settings of **Veronte Ops**:

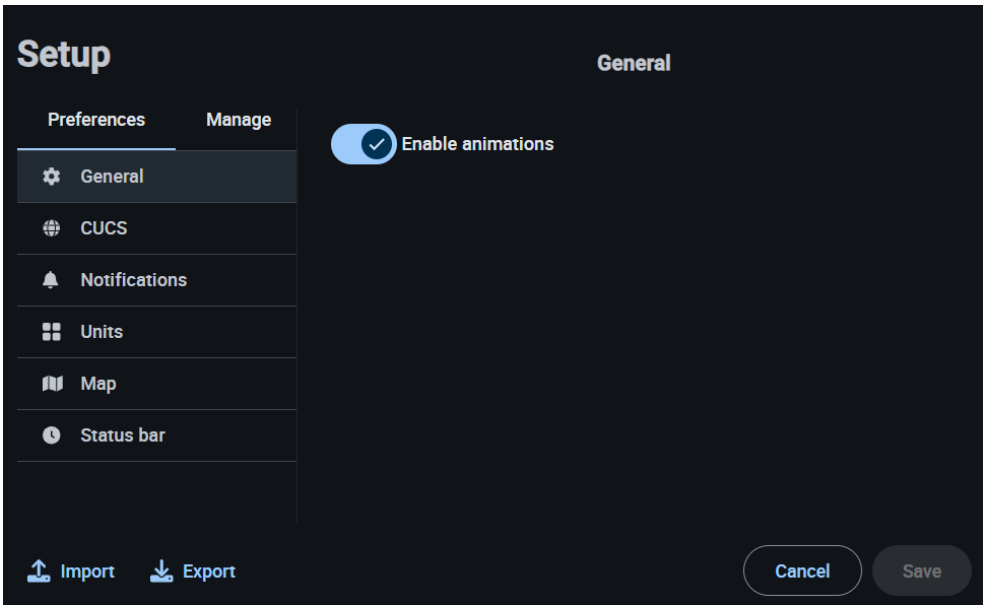


Status bar - Setup

> ⓘ **Note**
>
> Clicking the '**Close**' button, in the bottom right corner of the menu, will **close** this window **and save** any changes made.

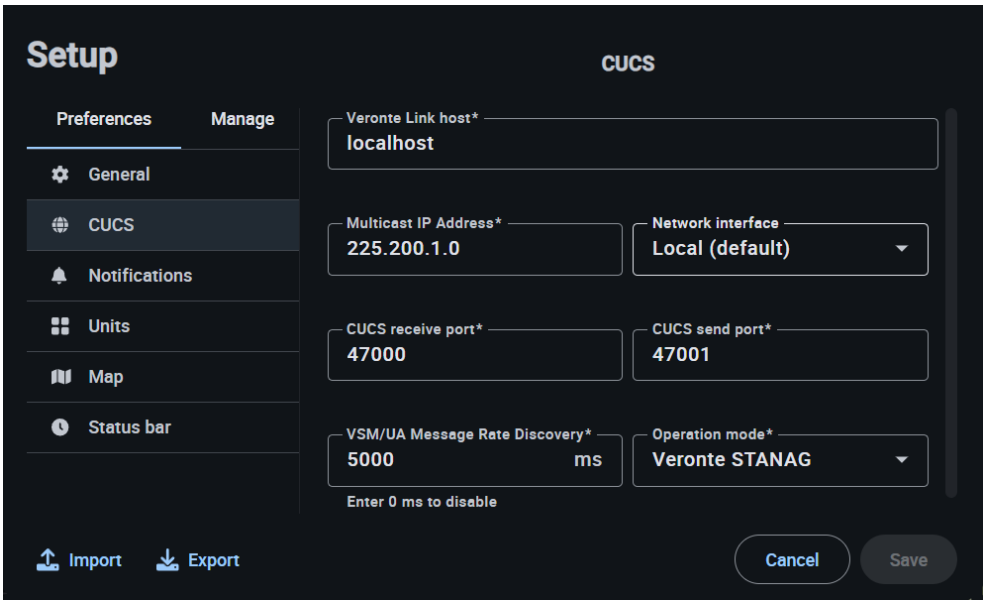
Preferences

- **General**
  Allows enabling/disabling application animations.



Status bar - General

- **CUCS** The CUCS allows the integration of the STANAG protocol, that enables different control stations to interact with the same UAV.



Status bar - CUCS

  - **Veronte Link Host**

    As Veronte Link is the Embention application that stores, reads and manages all the configurables of the products, the user can choose which Veronte Link to use to access the data, to access certain products. By default, the IP address 'localhost' is set, which is the Veronte Link on the same PC as **Veronte Ops**.

    However, it is also possible to connect **Veronte Ops** to a Veronte Link running on a different computer, simply by entering the IP address of

that computer, so **Veronte Ops** will be able to access the devices connected to it.

If the user has any problems when trying to connect **Veronte Ops** to Veronte Link, see Connecting to Veronte Link - Troubleshooting section of this manual.

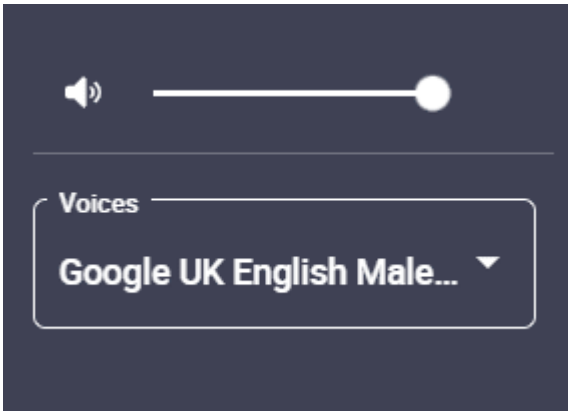
- **Multicast IP Address**: Address used for group transmission of data.
- **Network Interface**: Junction point that allows a device to connect to a network and exchange data.

  Selecting **Local (Default)** as Network Interface standardizes internal communication between VeronteLink and VeronteOps under a unified name, replacing the legacy loopback. This configuration is active by default to enable automatic interconnection between applications on the same computer without manual adjustments.
- **CUCS Receive/Send Port**: UDP ports dedicated respectively to the reception and sending of data.
- **VSM/UA Message Rate Discovery**: Frequency with which the system scans the network in search of new VSM modules.
- **Operation Mode**: Defines the operational protocol.

- **Notification**

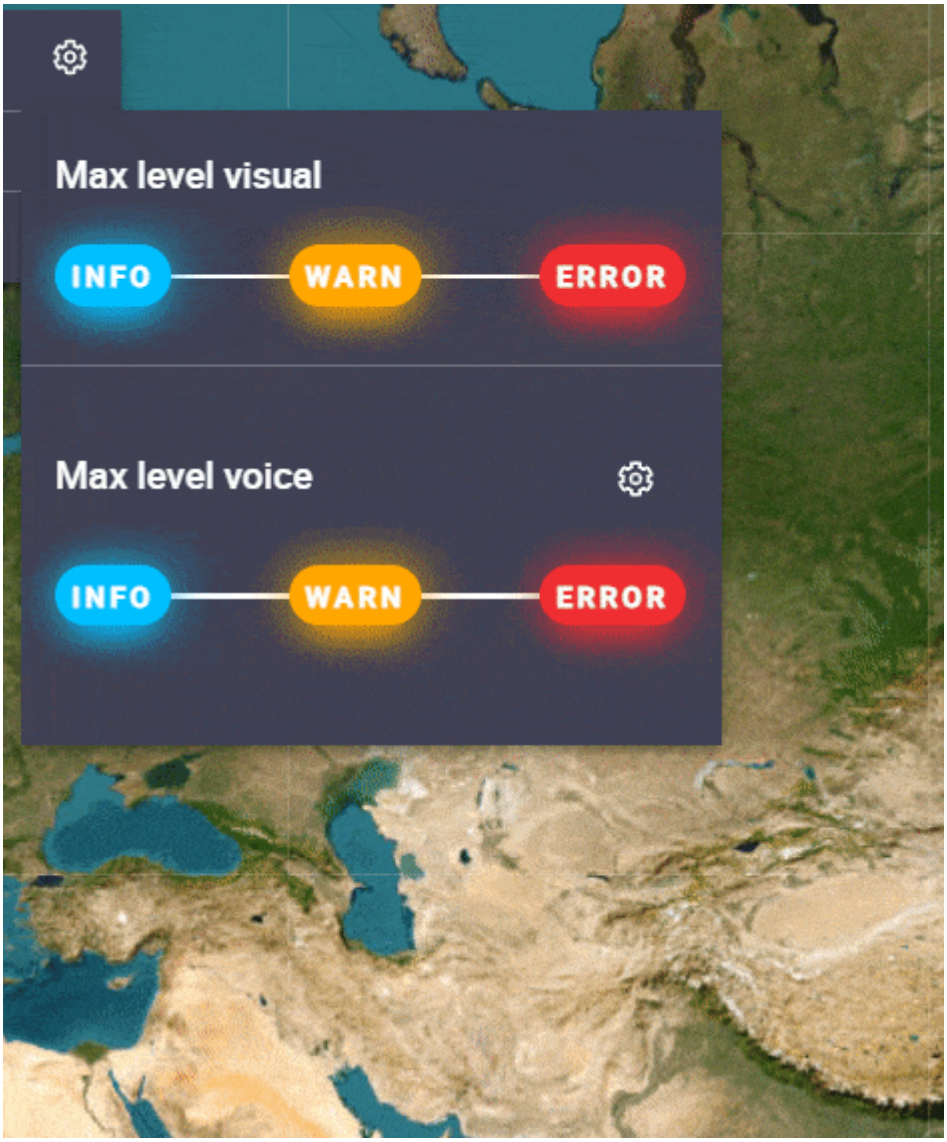
Notifications menu - Configuration

Besides, the user can also choose the language of the sound notifications by clicking on the settings icon:



Sound notifications language

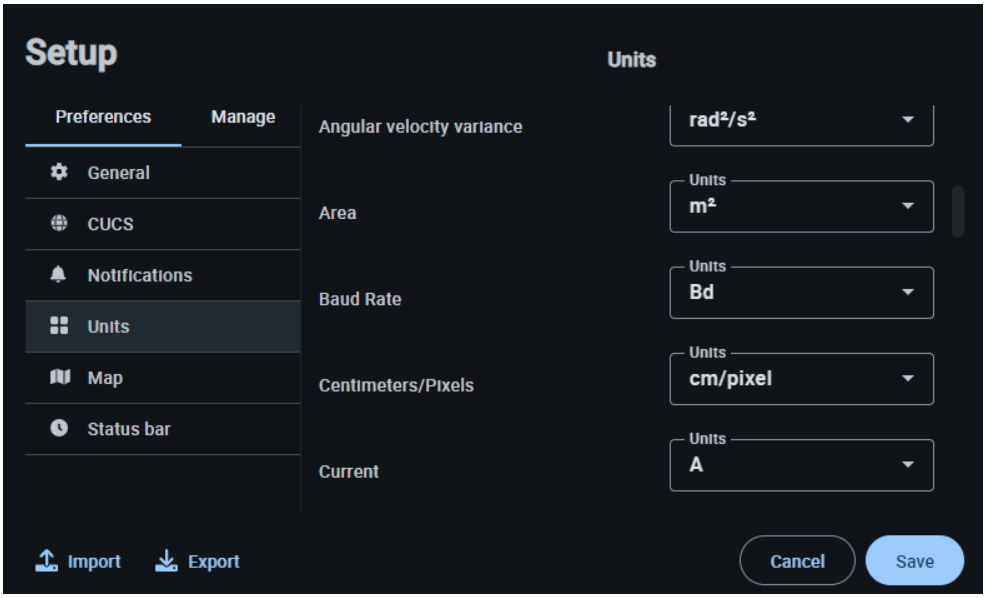
Below is an example of how to customize the notification settings:



Modified notifications

- **Units**

  This panel shows all the system units available for the system variables. They are sorted by variable type in alphabetical order: acceleration, temperature, velocity, etc.



Units settings panel

The following table shows all the units available in **Veronte Ops**:

Variable Type	Units
Acceleration	[m/s²] [ft/s²] [in/s²] [g]
Angle	[rad] rad[-π;π] rad[0;2π] [°] °[-180;180] °[0;360] [° ' “] °'"(N/S) °'"(E/W)
Angular Acceleration	[rad/s²] [rad/m²] [rad/h²] [°/s²] [°/m²] [°/h²]
Angular Velocity	[rad/s] [°/s] [rad/m] [rad/h] [rps] [rpm] [rph] [°/s]
Area	[m²] [cm²] [mm²] [km²] [mile²] [ft²] [yd²]
Baudrate	[Bd] [kBd] [MBd]

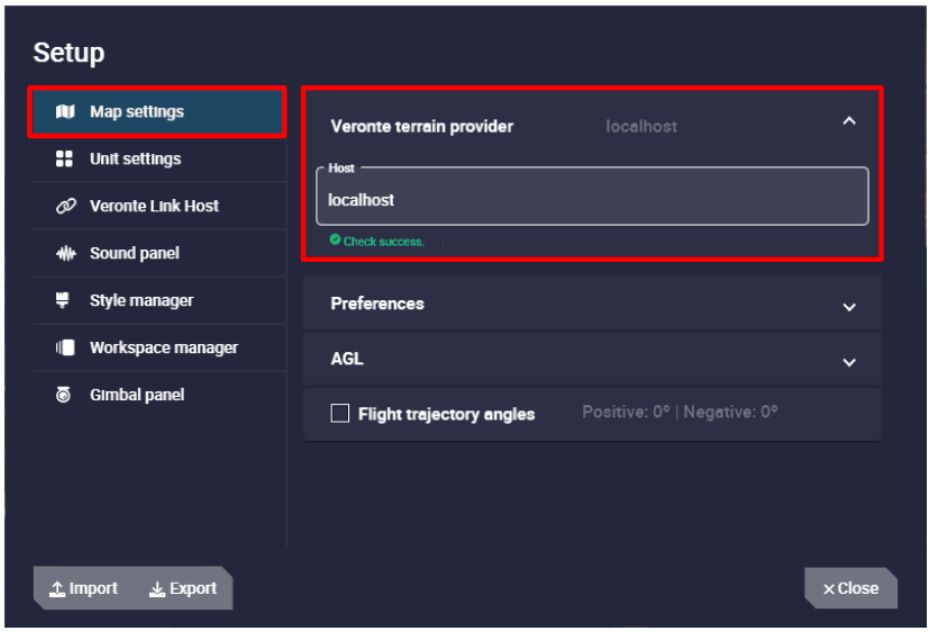
Variable Type	Units
Centimeters/Pixels	[cm/pixel]
Current	[A] [mA]
Data	[bit] [byte] [KB] [GB] [bytes/s]
Decibel	[db]
Density	[kg/m³]
Flow Rate	[m³/s] [gal/s] [gal/h] [l/s] [l/h]
Force	[N] [kN] [lbf] [pdl]
Frequency	[Hz] [MHz] [kHz]
Jerk	[m/s³]
Length	[m] [km] [mm] [cm] [mi] [NM] [yd] [ft] [in]
Magfield Variance	[T²]
Magnetic Flux Density	[T] [nT] [gauss] [mG]
Mass	[kg] [g] [tonnes] [lbs] [oz]
Numeral System	[bin] [octal] [dec] [hex]
Percentage	[x1] [%]
Power	[W] [kW] [Kgm/s] [erg/s] [CV]
Pressure	[Pa] [kPa] [bar] [mbar] [psi] [mmHg] [inHg] [ft] [at] [atm]
	[Pa²/s]

Variable Type	Units
Pressure Square Error Rate	
Pressure Variance	[Pa²]
Resistence	[Ω]
Temperature	[K] [°C] [°F]
Time	[s] [Time] [min] [h] [µs] [ms]
Transfer	[pkts/s]
Velocity	[m/s] [kt] [km/h] [mph] [ft/s] [ft/m] [mm/s]
Velocity Variance	[(m/s)²] [(cm/s)²] [(mm/s)²]
Voltage	[V] [mV]
Volume	[m³] [dm³] [mm³] [L] [mL]

- **Map**

  In this menu, it is possible to edit the terrain data settings, such as:

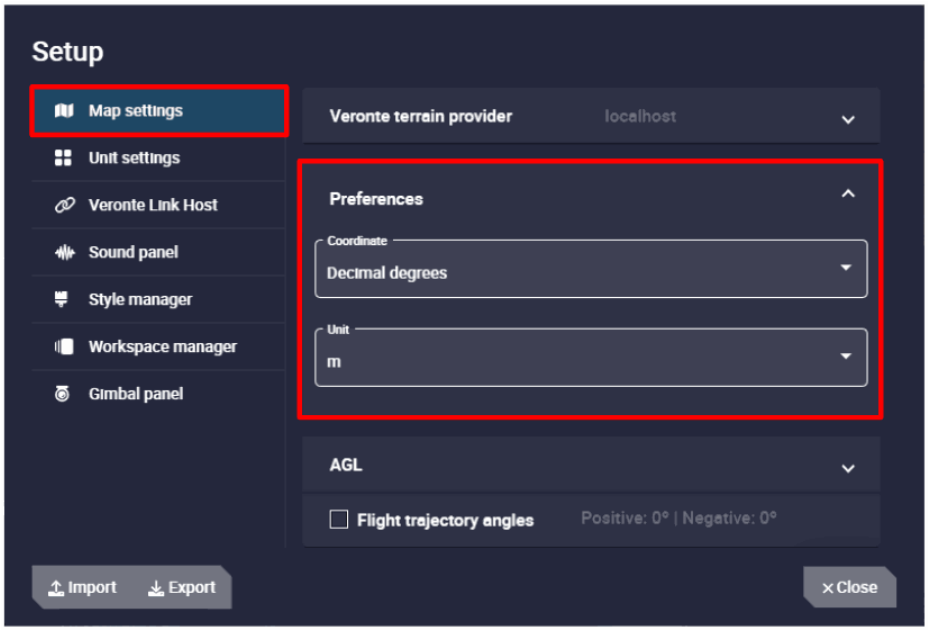
  - **Veronte Ops SRTM data provider**:
    - **Host**: The default SRTM data provider configured is the Veronte Terrain Provider application (i.e., localhost), but users can enter another SRTM data provider that is located on the PC.



Map settings panel - Veronte Terrain Provider

- **Preferences**

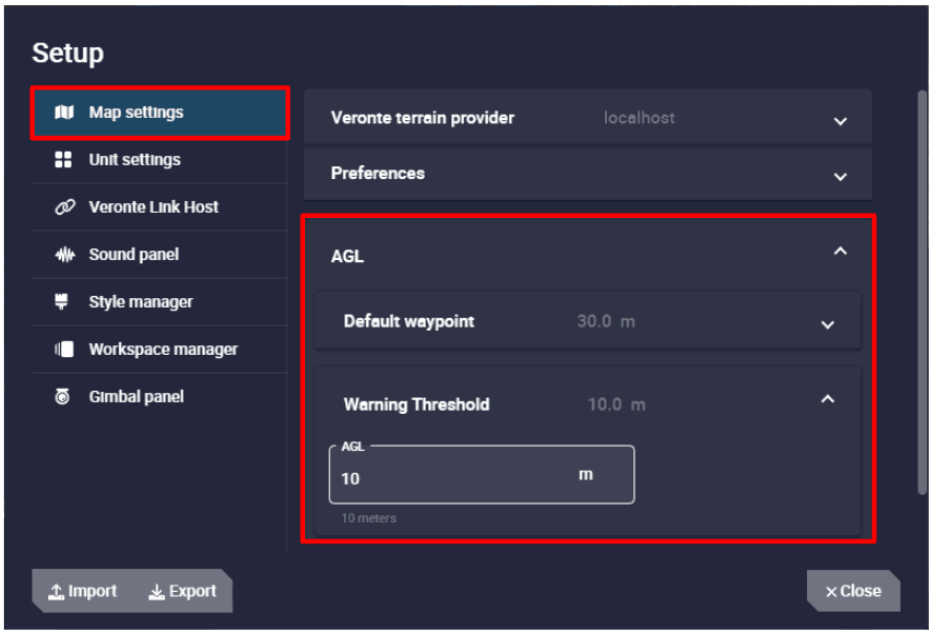
  Users can customize the coordinates and height units of the map.



Map settings panel - Preferences

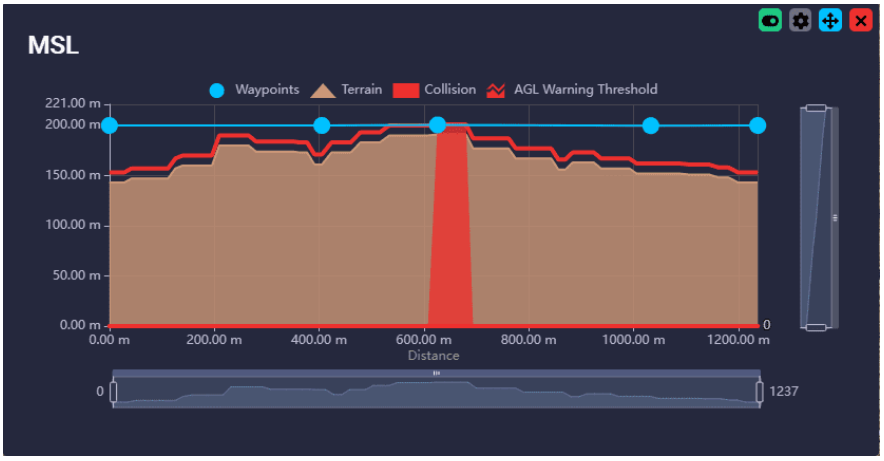
  - **Coordinate**: Select the units of the coordinates. The available options are **Decimal degress**, **Degrees**, **UTM** and **MGRS**.
  - **Unit**: It is possible to select the units of the height from the drop-down menu.

- **AGL**



Map settings panel - AGL

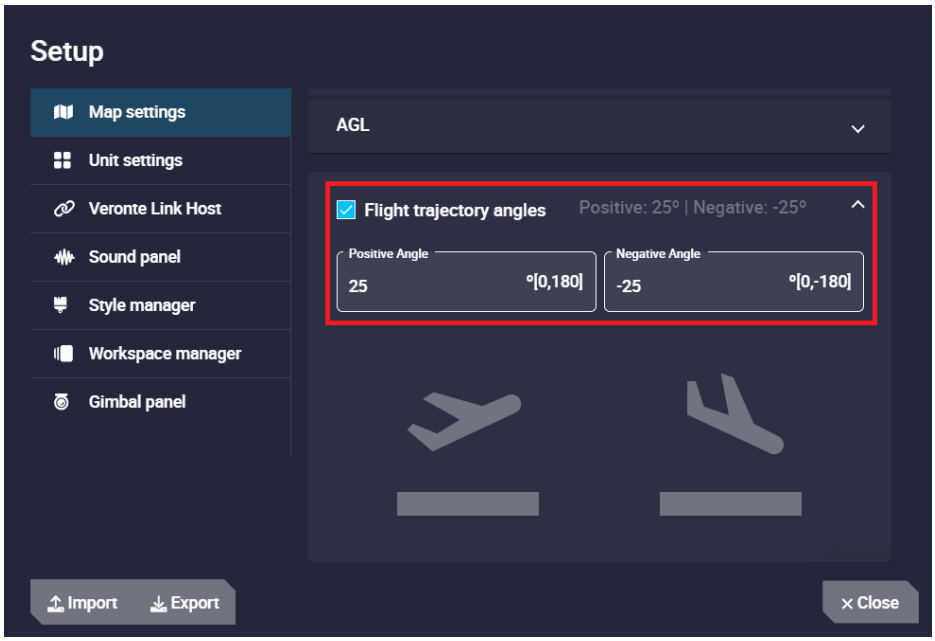
- **Default waypoints**: Enter the default altitude, in AGL, of the waypoints created.
- **Warning Threshold**: It is possible to add a warning threshold to **avoid collision with terrain**. This will appear as a line when users open the '**elevations**' option in the created route.
  For example, if the warning threshold is set to 10 metres, a **red warning line** will be drawn in the 'elevations' menu **10 metres above each terrain point**. Also, if the **mission path intersects this line**, **Veronte Ops** will interpret this as the route **colliding with the terrain**:



Map settings panel - Warning

- **Flight trajectory angles**: Defines the angle between the aircraft's velocity vector and the horizontal plane. A positive angle indicates a climb, while a negative angle indicates a descent.

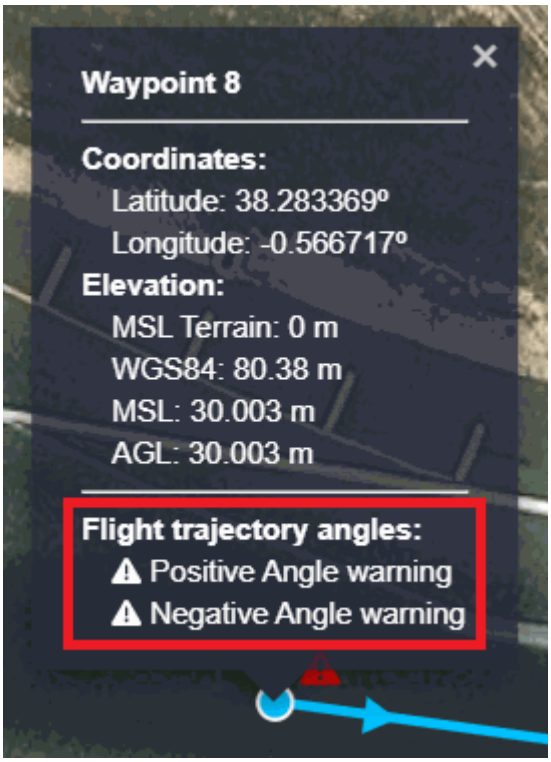
  Enabling this option allows the user to set the operational limits for these angles, which Veronte Ops uses to visually validate the mission trajectory against the aircraft's capabilities.



Map panel - Flight trajectory angles

  - **Positive Angle**: Sets the maximum angle for climbing in degrees.
  - **Negative Angle**: Sets the maximum angle for descending in degrees.

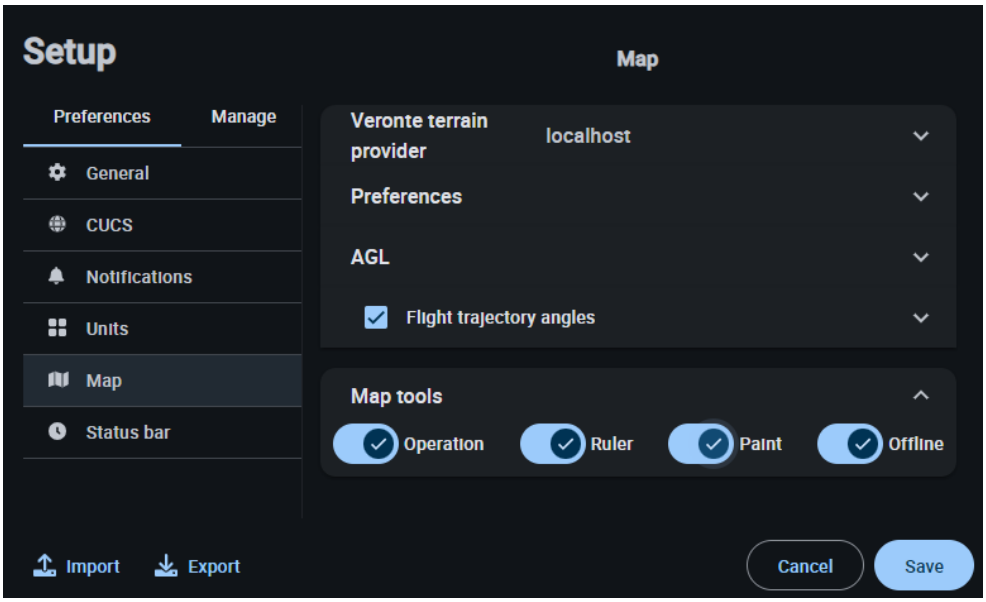
If the configured Positive/Negative Angles result in a climb or descent capability that is insufficient to meet the target altitude, a Warning will be triggered.



Map panel - Flight trajectory angles warning

- **Map tools**

Here, the user can choose, by enabling or disabling them, the group of functionalities to be displayed in the mission toolbar.



Map panel - Map tools

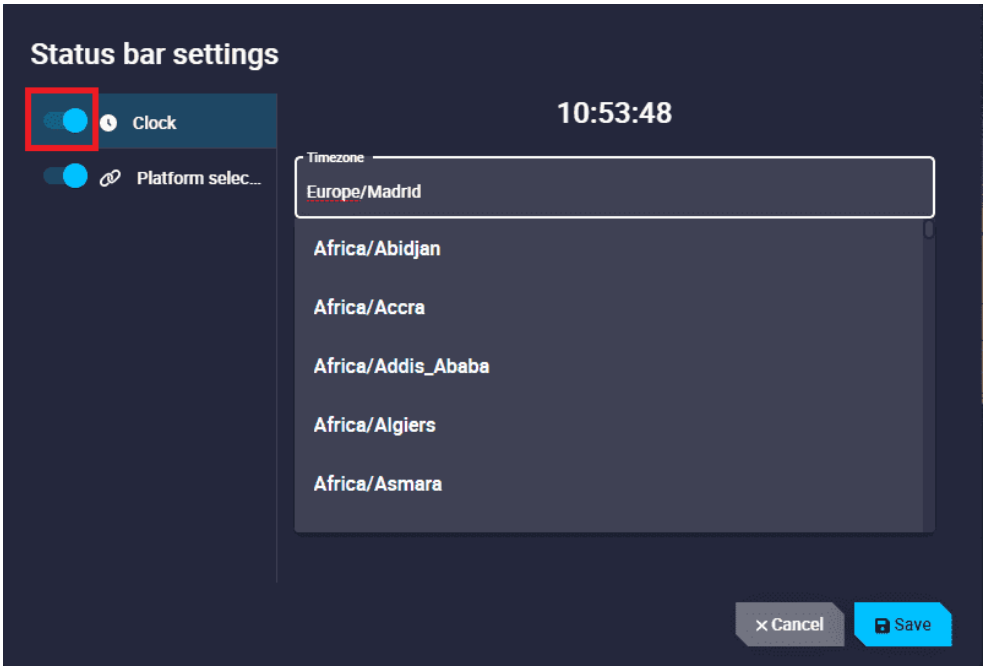
To save the changes made, click on the '**Save**' button.

Each group of functionalities has been described in the previous sections.

- Operation
- Ruler

  - Paint
  - Offline
- **Status bar**
  - **Show Clock**

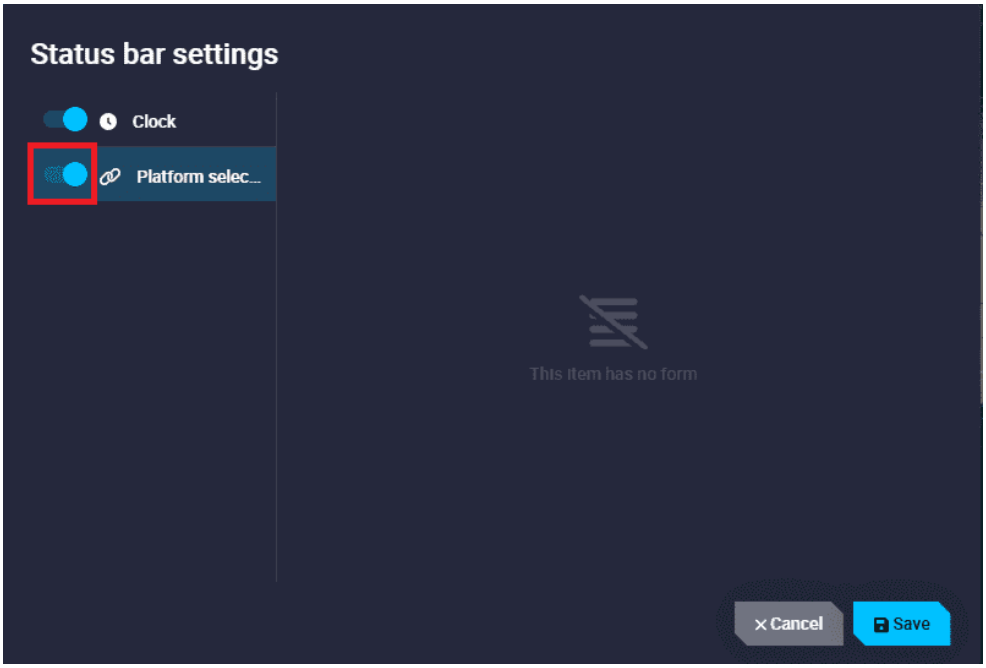
It is possible to customize the timezone where users are working.



Clock panel

  - **Show platform selected**

This element informs the user whether a **Veronte Autopilot 1x** is **not connected**, **not selected**, **connected**, **disconnected** or **in maintenance mode**, and the name of this device.



Platform selected panel

Manage

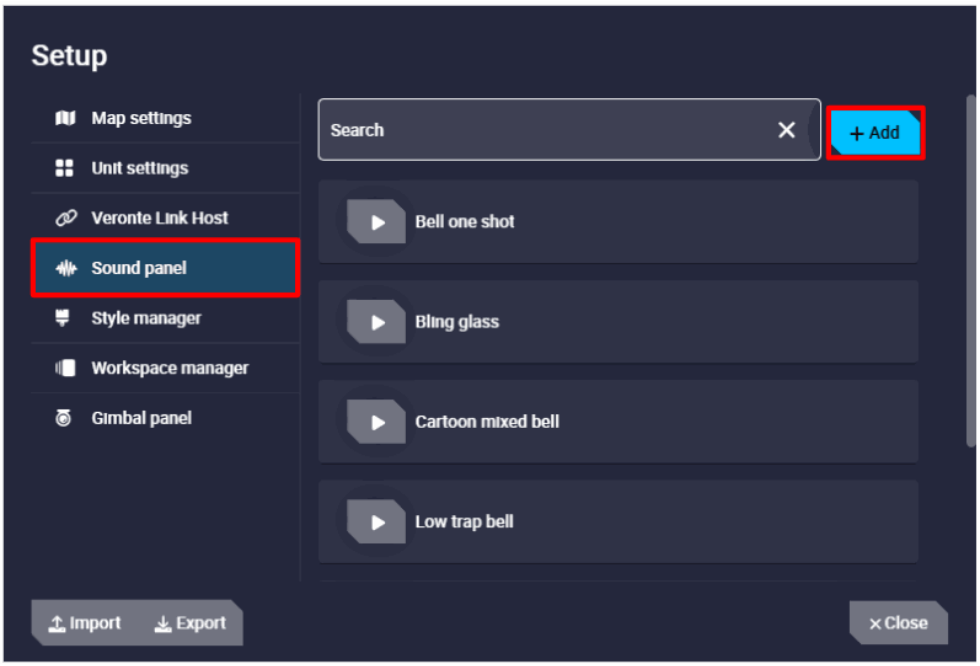
- **Sounds**

  This section lists the available sounds that can be selected through the editing/configuration of the mission.

  In addition, users can also add their own audio files from local storage. For this, click **Add** and select the desired audio file.

  Supported formats

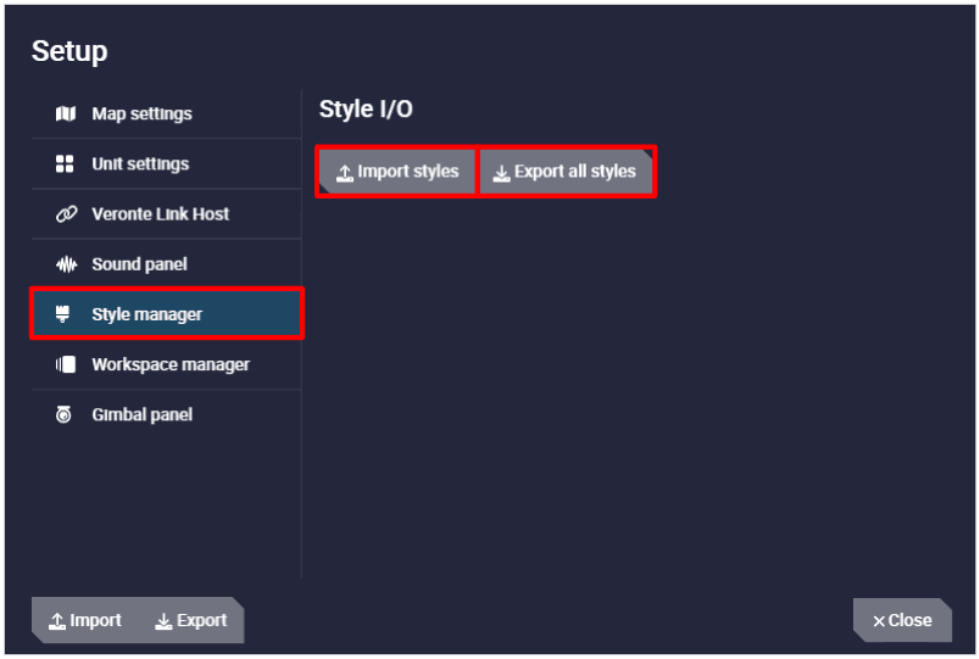
  are .opus, .flac, .webm, .weba, .wav, .ogg, .m4a, .oga, .mid, .mp3, .aiff, .wma, .au.



Sound panel

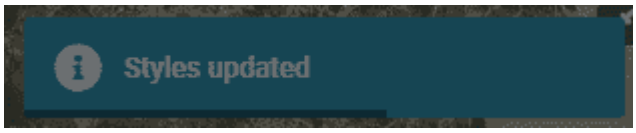
- **Styles**

  This section allows the user to import and export **custom widget styles**.



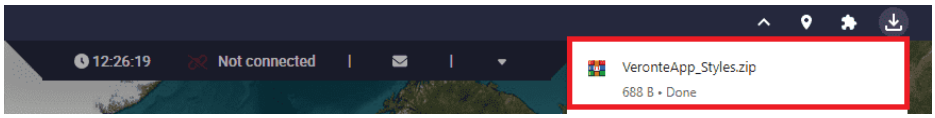
Style manager panel

- **Import styles**: By clicking here, users can import custom styles as a **zip folder** or by directly selecting all desired custom styles `.css` files. When the styles are imported, a notification will appear:



Style manager panel - Import notification

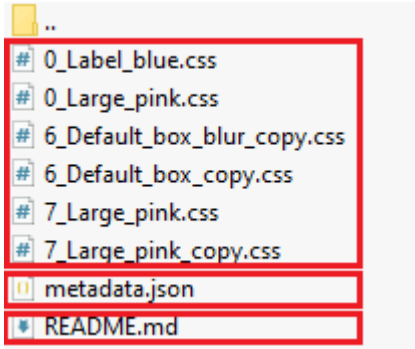
- **Export all styles**: By clicking here, all **custom** styles will be exported as a zip folder:



Style manager panel - Export

> ⓘ **Note**
>
> **Default** widget styles will **not** be **exported**.

The zip folder will contain all the `.css` files with custom styles for the different widgets, a `metadata.json` file and a `README` file:



Style manager panel - Exported files

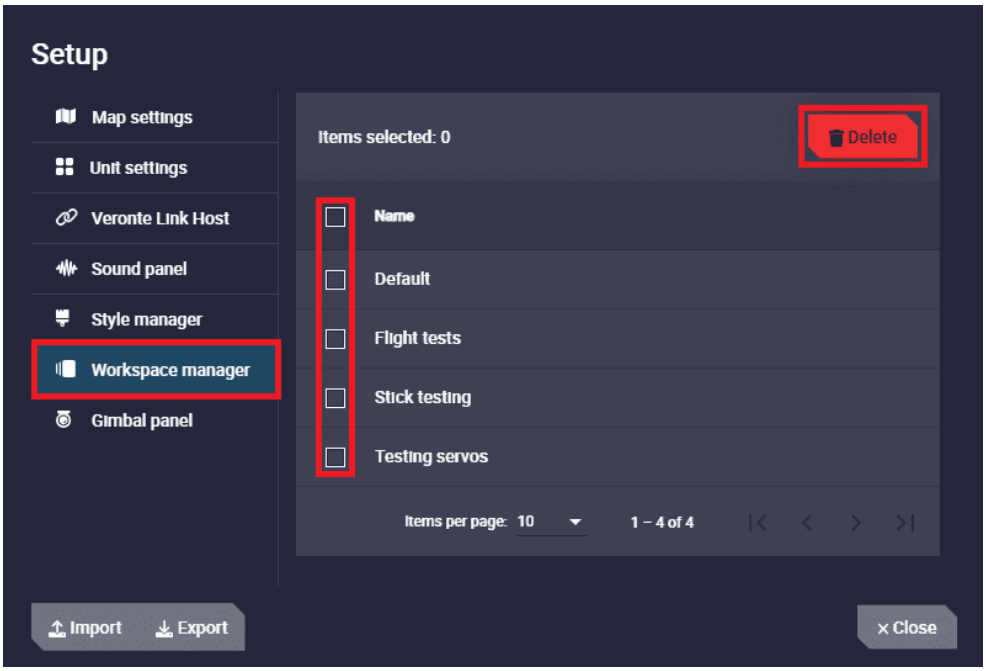
- **metadata.json** file: This file contains the IDs of the styles that are assigned to the workspaces.

  > **Danger**
  > It is advisable **not to modify** this file.

- **README** file: Pay special attention to this file to find out how to create or modify a style.

- **Workspaces**

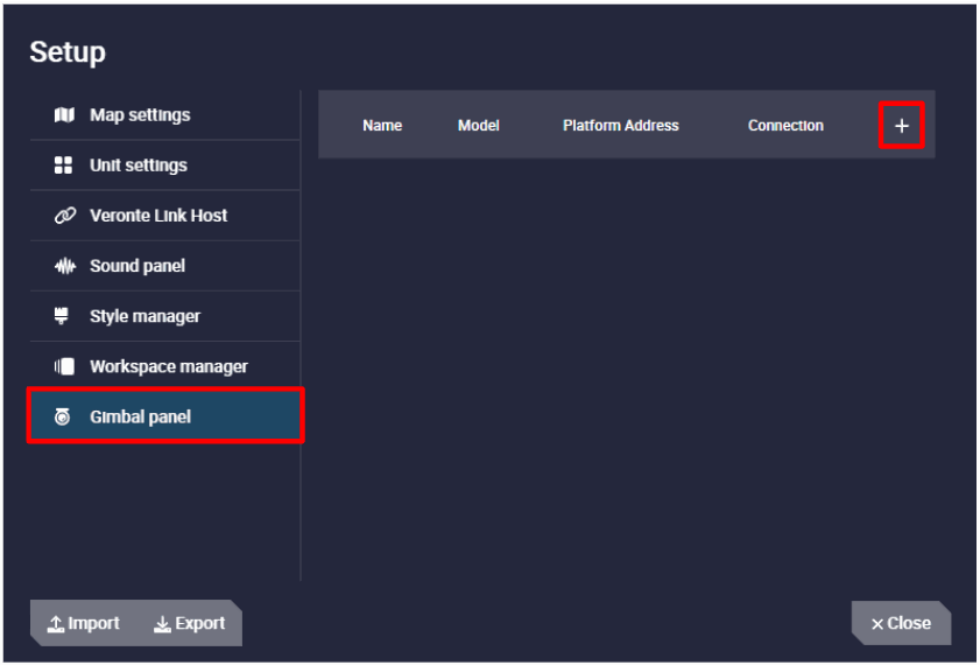
  From this panel users can **delete** one, several or all **workspaces** created. To do this, simply **select** the **workspaces** to be deleted and then click on the '**Delete**' button.



Workspace manager panel

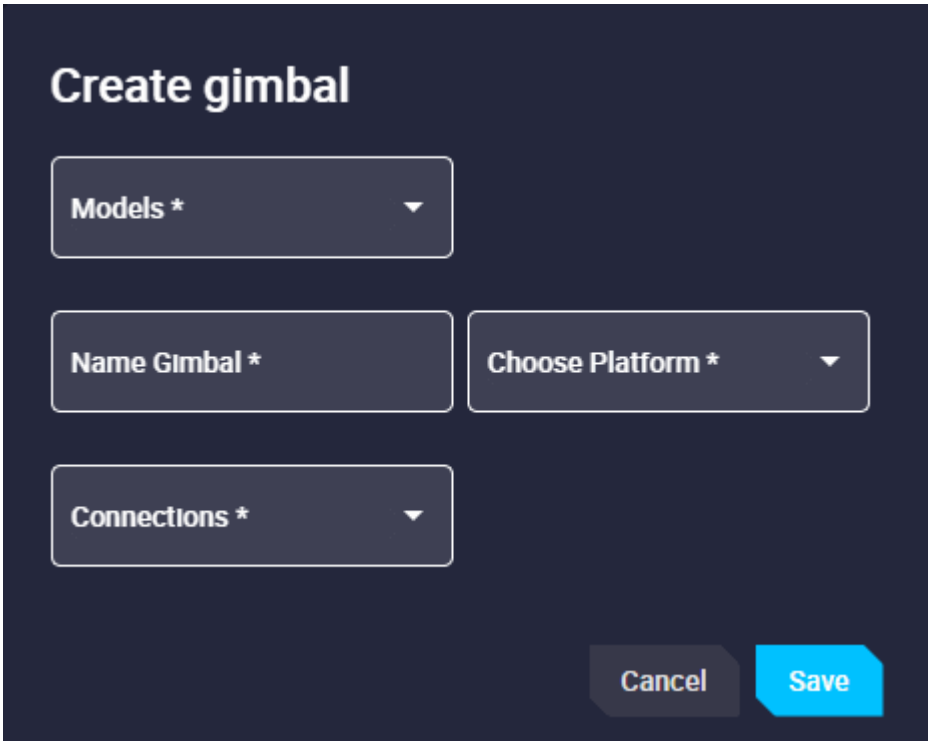
- **Gimbals**

Users can add a predefined gimbal from this panel:



Gimbal panel

Clicking the + icon displays the following configuration panel to add a new gimbal:



Gimbal panel - Create gimbal

- **Models**: Select a gimbal model from the list. The available options are **NextVision Trip2/Trip5**, **Veronte Gimbal 10z** and **Veronte Gimbal 30z**.

- **Name Gimbal**: Enter the desired name for the gimbal to be added.
- **Choose Platform**: Users must select the platform for which the gimbal is configured.
  The available options will always be the IDs of the connected Autopilots 1x and '**Selected platform**', i.e. the platform that is selected.
- **Connections**: The type of connection through which **Veronte Ops** sends commands to the gimbal must be selected:
  - **Veronte Tunnel**: When choosing this connection type, **Veronte Autopilot 1x** tunnel is used to send commands to the gimbal. Users must also specify the **Veronte tunnel port** used: 1, 2 or 3.

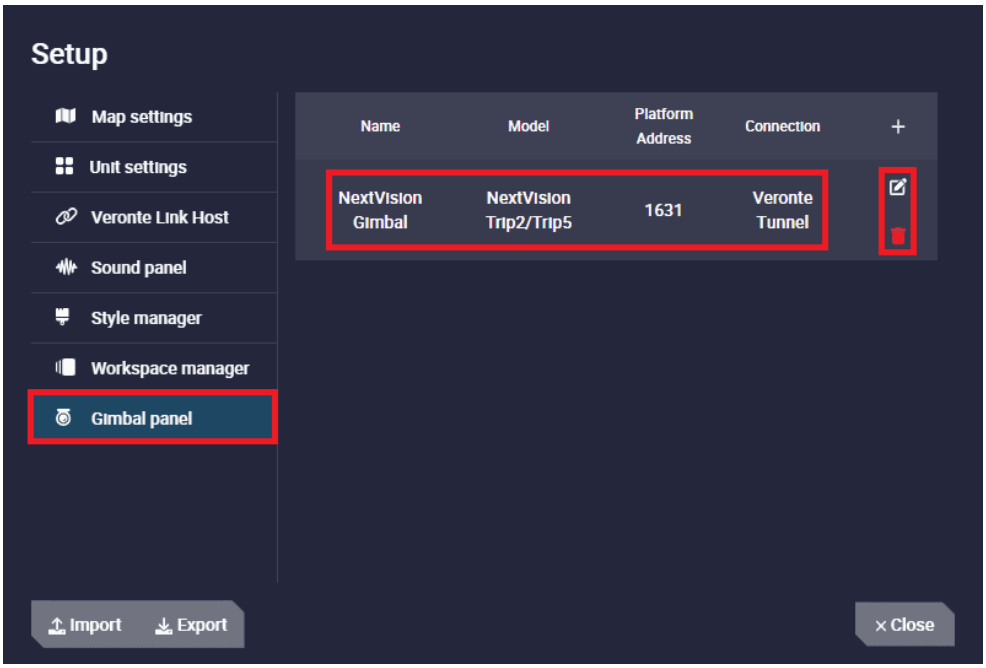
    > ⚠ **Warning**
    > Be careful! This port must match the Tunnel port configured in the **1x PDI Builder** software.

  - **WebSocket**: In order to send commands via the Web Converter application, it is required to enter the **websocket_url** configured in that tool in the **URL** field and copy the **websocket_port** configured as the camera's udp connection in the **port** field.
- **Advanced**: Depending on the gimbal model chosen, some additional options can be configured which are characteristic of each gimbal.

  > ⓘ **Note**
  > This option will only appear after the selection of the gimbal model.

  - NextVision Trip2/Trip5 model: It is possible to enable/disable the sending of some specific messages.
  - Veronte Gimbal 10z/30z models: The **Id** of the **gimbal block** that has been configured in the **Veronte Autopilot 1x** setup can be entered. For more information on this, please refer to the Gimbal - Devices blocks of the **Block Programs** section in the **1x PDI Builder** user manual.

Then, the created gimbal will appear on the panel:

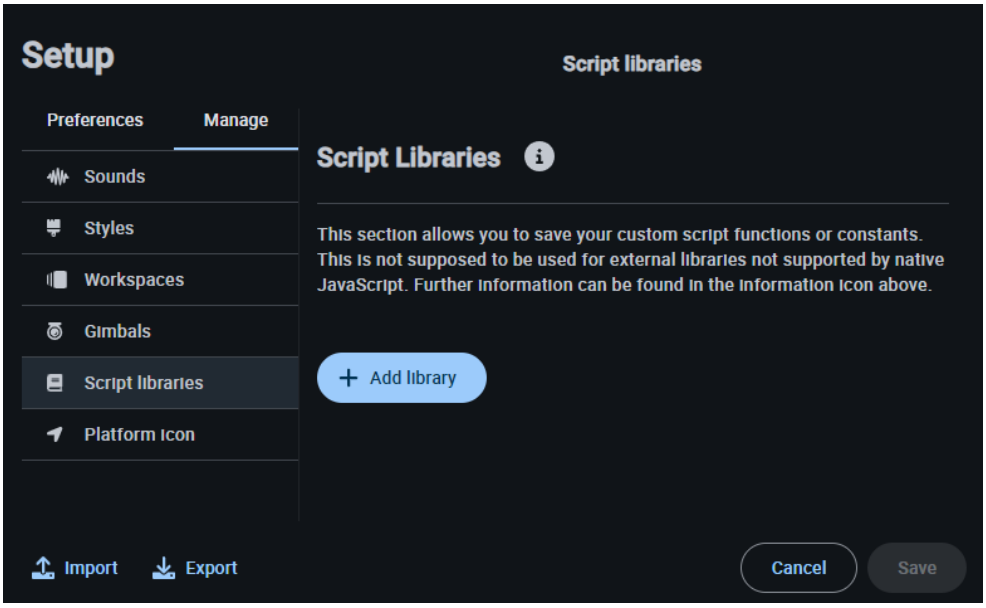


Gimbal panel - Gimbal added

- **Edit gimbal**: Allows the user to access again to the configuration menu described above.
- **Remove Gimbal**: Removes this gimbal.

- **Script Libraries**

  The Script Libraries section allows you to create and manage custom libraries in native JavaScript directly within Veronte Ops. It is used to define reusable custom functions or constants to automate processes or perform complex calculations not provided by default.



Script Libraries

Click on + Add library to insert custom code.

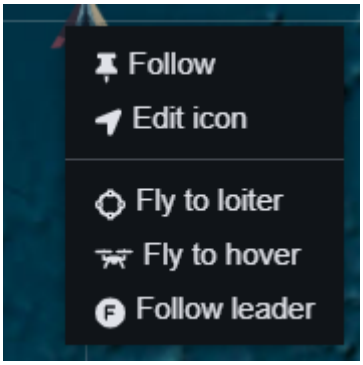
> ⚠ **Warning**
> It does not support loading external libraries that are incompatible with native JavaScript.

Further technical details are available by clicking the information icon.

- **Platform icon**

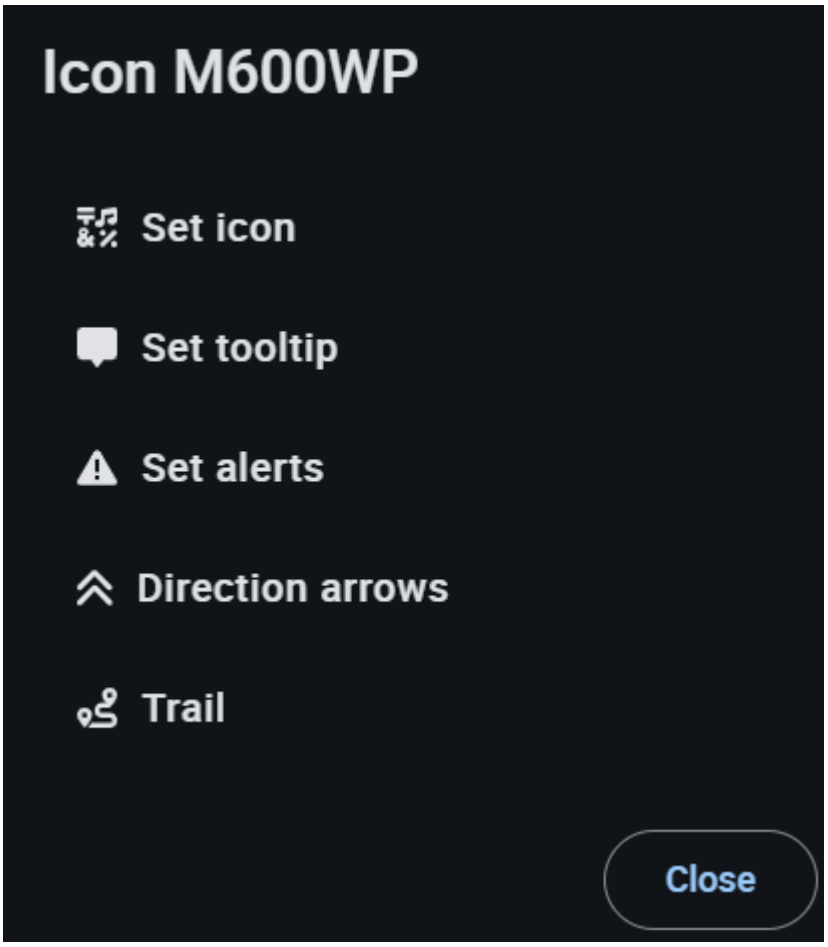
This icon allows the user to locate and follow the position of the platform at any time during the operation.

By **right-clicking** on it, users will access its **options**:



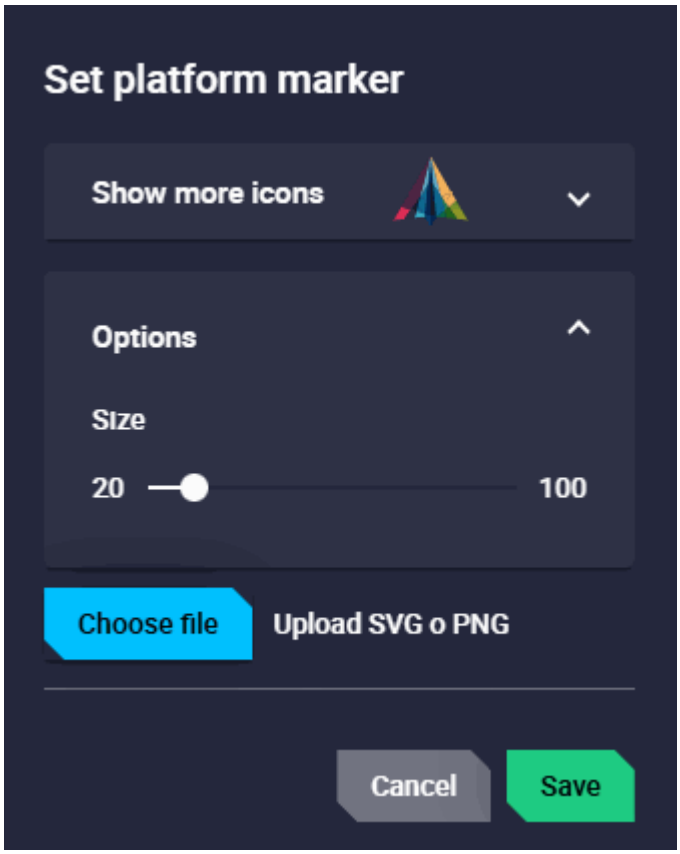
Platform icon options

- **Follow**: Keeps the platform always centered on the map during operation.
- **Edit icon**: Users will access to the settings menu:



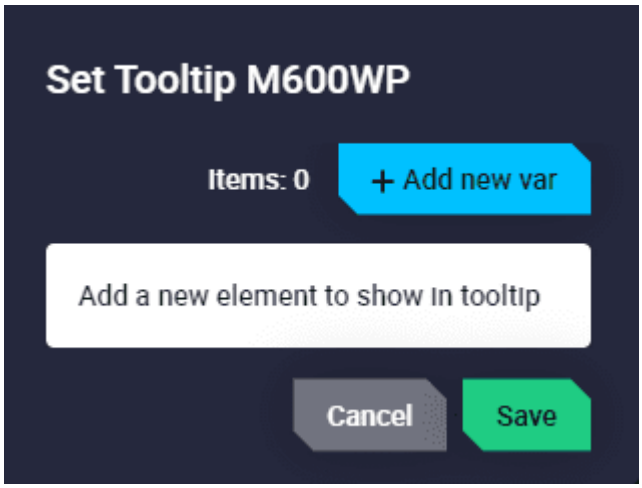
Platform icon settings

- **Set icon**: Users can customize the icon that will be used to represent the position of this unit on the map. They can choose one of the default icons or upload one from the laptop, as well as set the icon size.



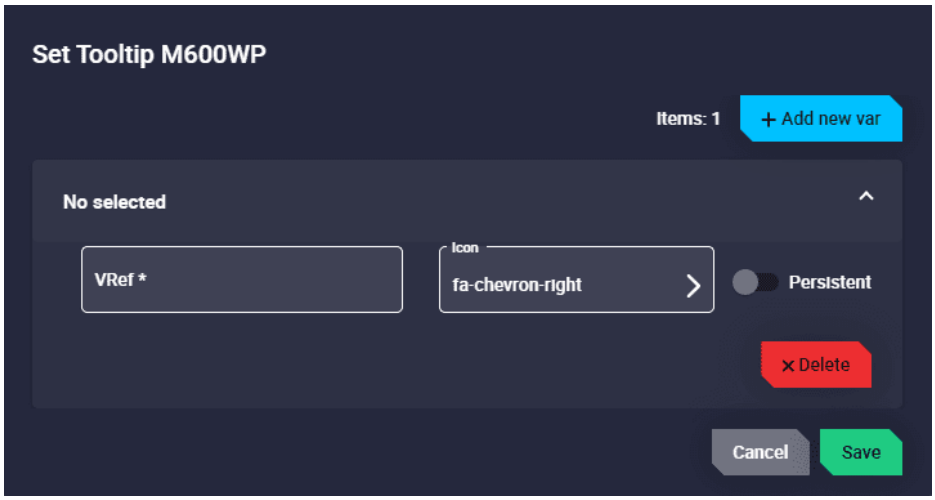
Platform icon settings - Set icon

- **Set tooltip**: Allows the user to display a variable next to the platform icon.



Platform icon settings - Set tooltip

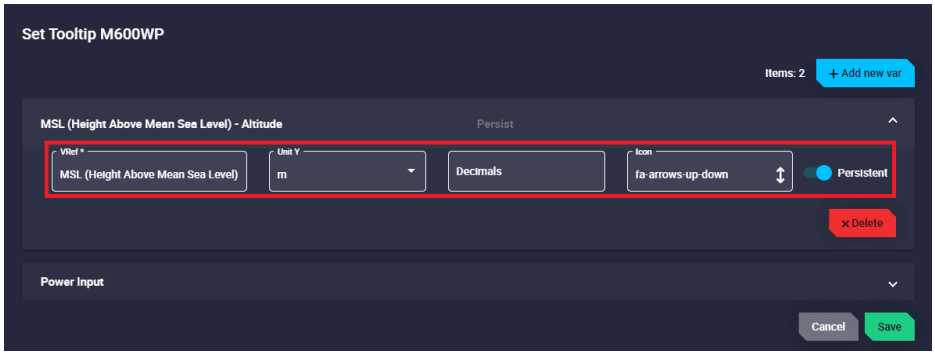
The following parameters must be configure by clicking on '**Add new var**':



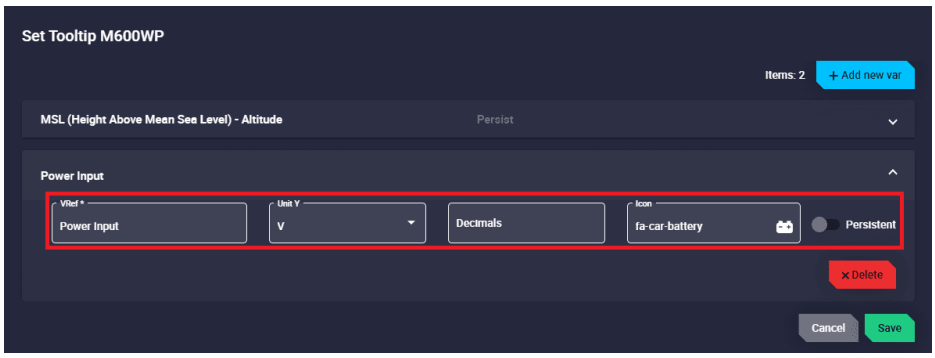
Platform icon settings - Set tooltip configuration

- **VRef**: Select the desired variable to be displayed.
- **Unit** and **Decimals**: The units and decimals can also be modified. These parameters will appear depending on the variable chosen.
- **Icon**: Choose the icon to be shown next to the variable.
- **Persistent**: When **enabled**, this tooltip will **remain displayed during the operation**. **Otherwise**, it will **only** appear **when clicking** on the icon.

An example is shown here:



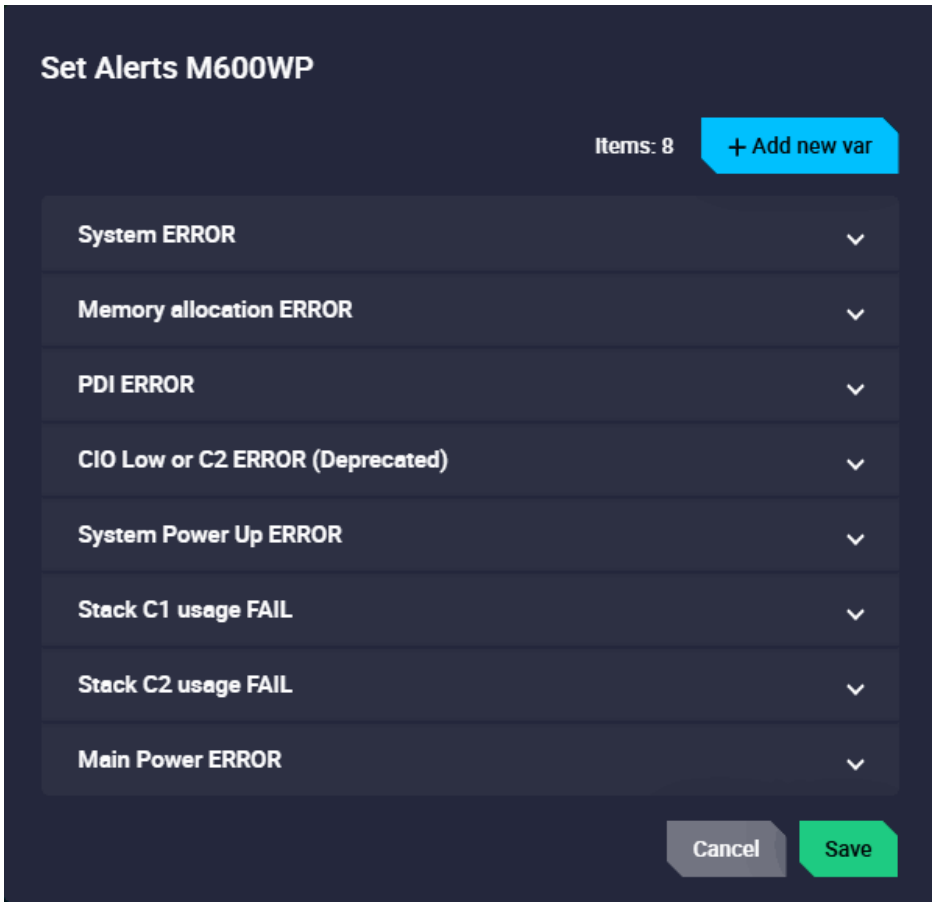
Platform icon settings - Set tooltip example



Platform icon settings - Set tooltip example

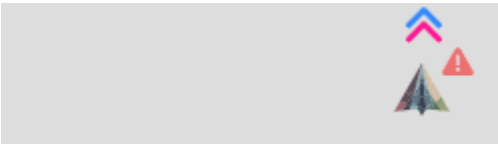
Platform icon settings - Set tooltip example

- **Set alerts**: These alerts are **bit** variables that appear next to the platform icon when in 'error mode'.
  By default, some bits that are sent in the autopilot 'Extended Status Message' are configured as alerts. The user can add as many alerts as desired.



Platform icon settings - Set alerts

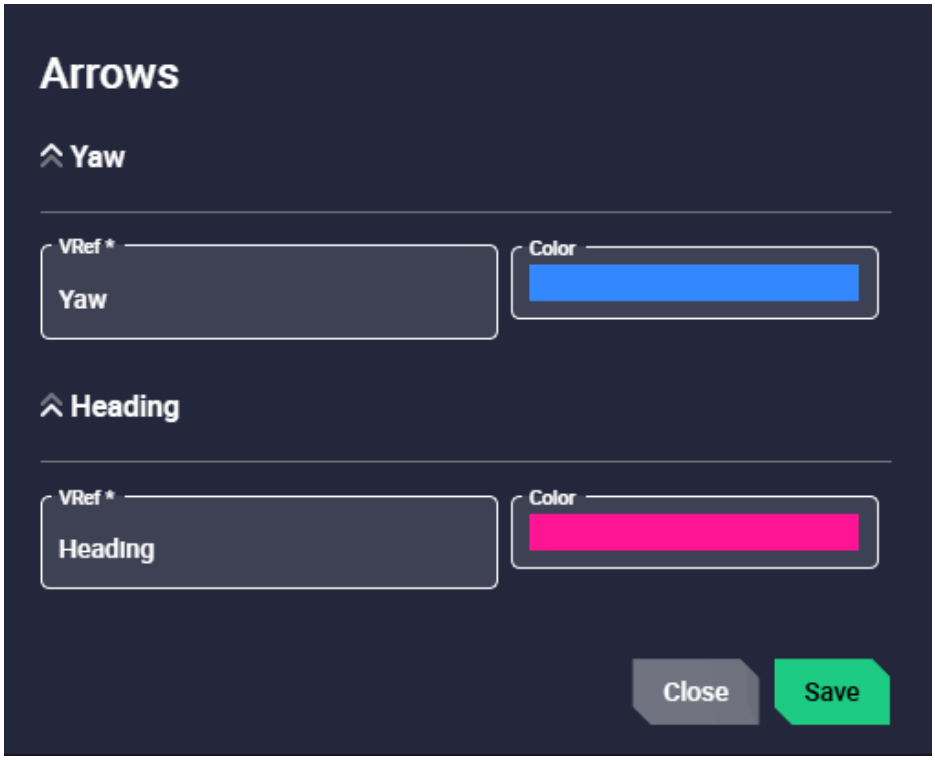
They are displayed with the **icon**, which has been configured with them, **flashing red** when in 'error mode'. When the platform icon is clicked on, the name of the variable that has triggered this alert will be displayed. An example is shown below:



Platform icon settings - Set alerts

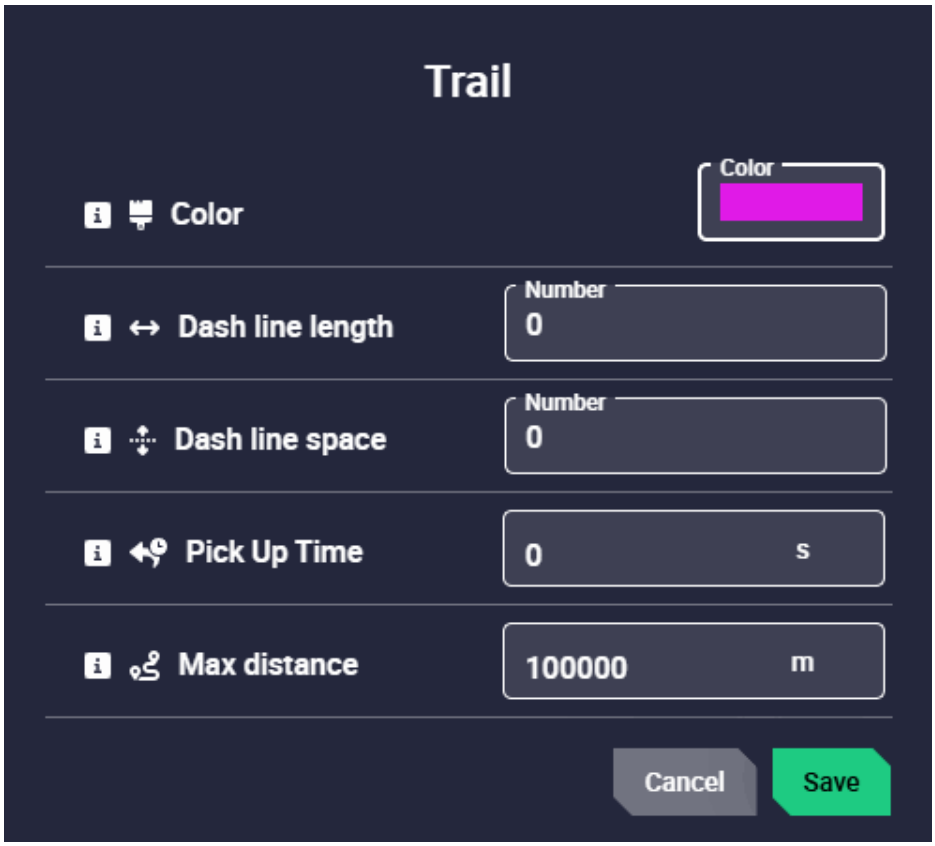
- **Direction arrows**: Users can choose variables from the **real angular variables** of the system to be displayed as the **yaw** and **heading arrows**.

  By default, the variables displayed are Yaw and Heading respectively. In addition, users can change the color of each arrow as desired.



Platform icon settings - Direction arrows

- **Trail**: The trail path of the platform can be customized.

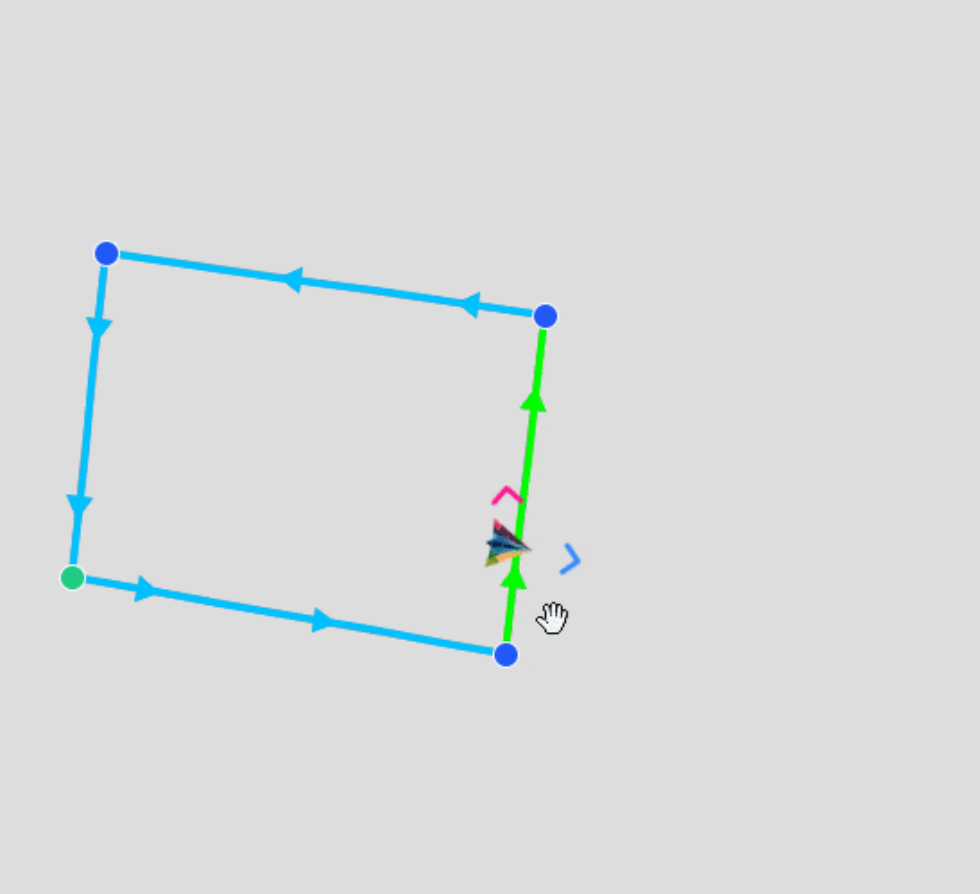


Platform icon settings - Trail

- **Color**: Set the color trail.
- **Dash line length**: Sets the length of the dash line. This change is **visualized** when a **number other than of 0 is set**.
- **Dash line space**: Sets the space between dash lines. This change is **visualized** when a **number other than of 0 and greater than** dash line length is set.
- **Pick Up Time**: Sets the time for the first point of the trial to be removed. This change is **visualized** when the time is **other than 0 s**.
- **Max distance**: Sets the maximum length of the path to be drawn.

- **Fly to loiter**: Create a volatile loiter and change the current route.

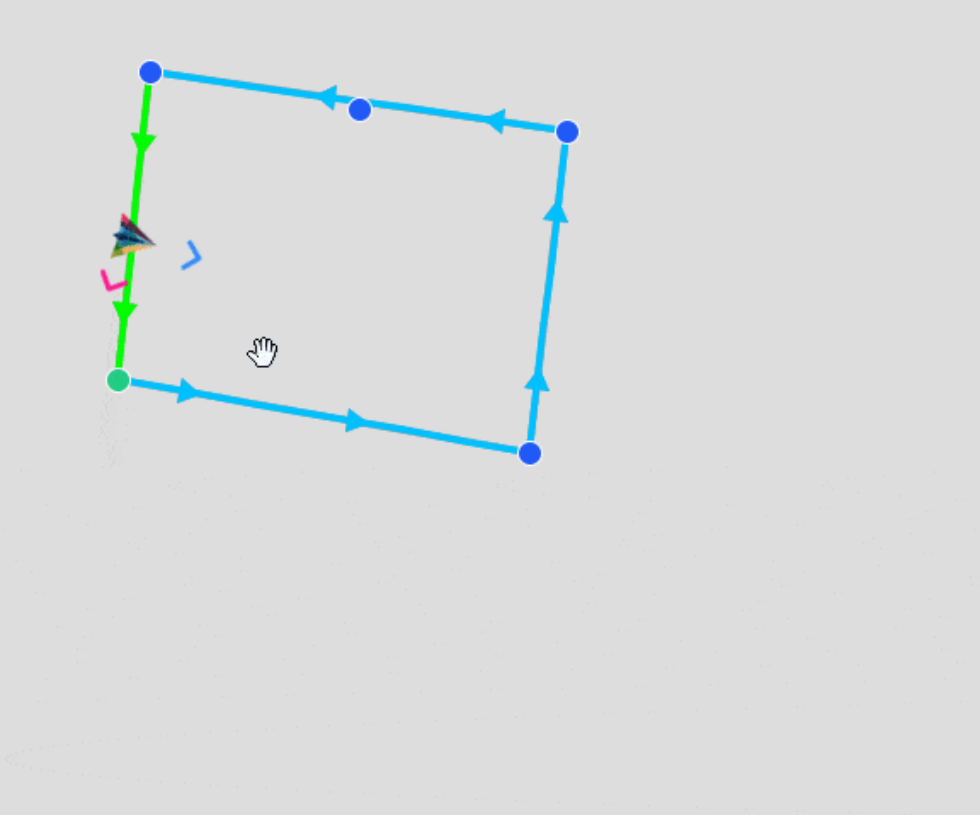
> ⓘ **Note**
>
> The platform will remain the altitude it had before this command.



Platform icon - Fly to loiter

- **Fly to hover**: Create a volatile waypoint and change the current route.

> ⓘ **Note**
>
> The platform will remain the altitude it had before this command.



Platform icon - Fly to hover

- **Follow leader**: With this action the platform will follow a moving object (it can be another platform). For more information on this, see Follow Leader automation of the **1x PDI Builder** manual.

The platform icon changes its appearance depending on the situation:

- **Platform disconnected**:
  - When **Autopilot 1x** is not connected, the platform icon is '**translucent grey**'
  - By default, it is placed in the world coordinates (0,0).

Platform icon - Translucent grey

- **Platform connected**:
  - **During operation/configuration creation**:
    - The platform icon is '**translucent colored**', indicating that GPS data is not being received, i.e. **Position not fixed** status.

- It is placed in the Initial position UAV defined in the **Operation Panel**.

Platform icon - Translucent colored

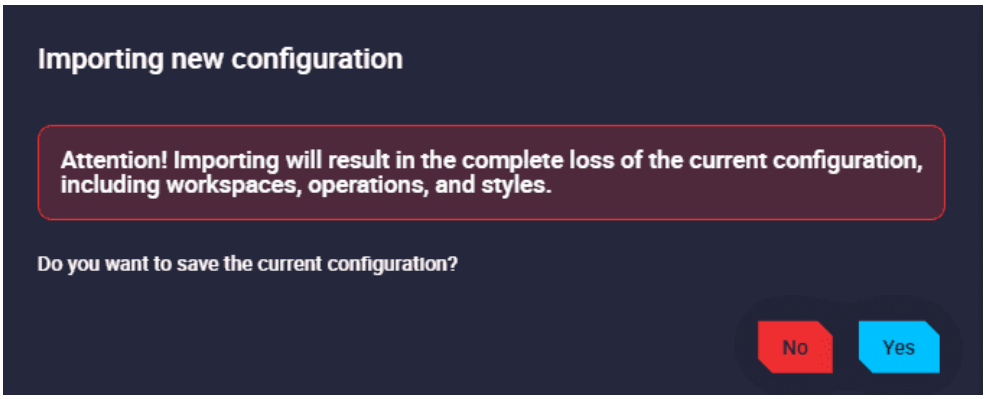
- **During operation/simulation**:
  - The platform icon is '**colored**', indicating that GPS data is being received/simulated.
  - It is placed in the actual position/simulation position.

Platform icon - Colored

- **Import/Export**

  Furthermore, all these settings can be **exported and imported** from one **Veronte Ops** to another, for example to move them from one PC to another.

  When the user tries to import a new **Veronte Ops** configuration, the following confirmation message will appear:



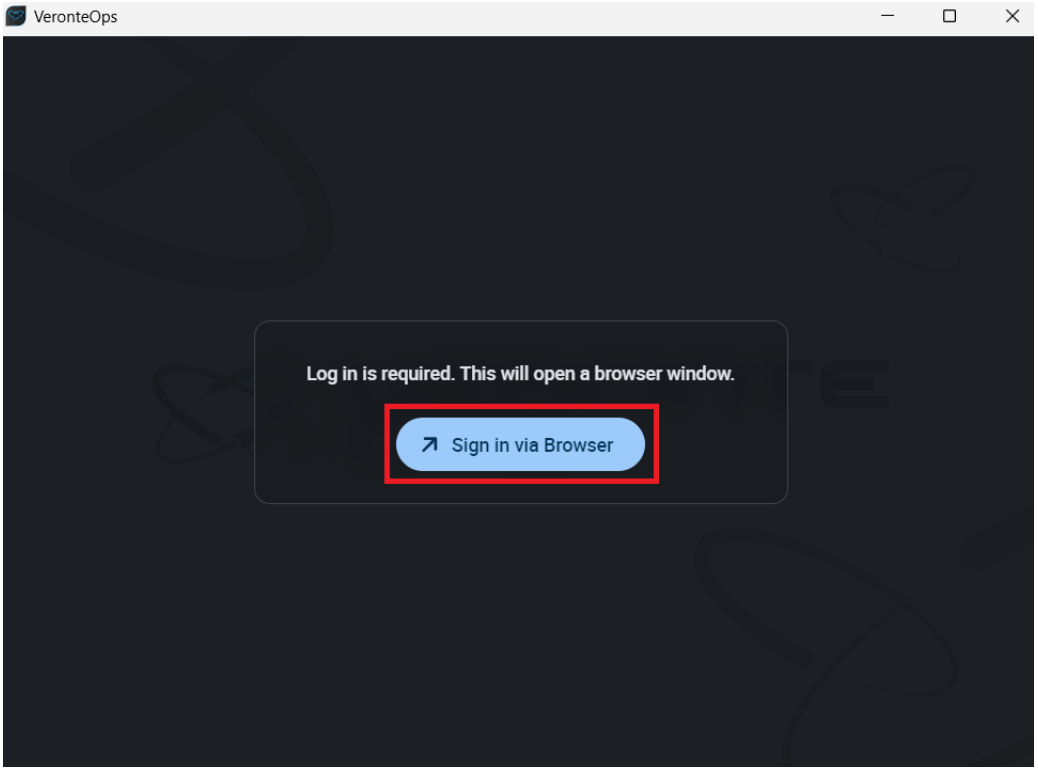
Import Veronte Ops configuration

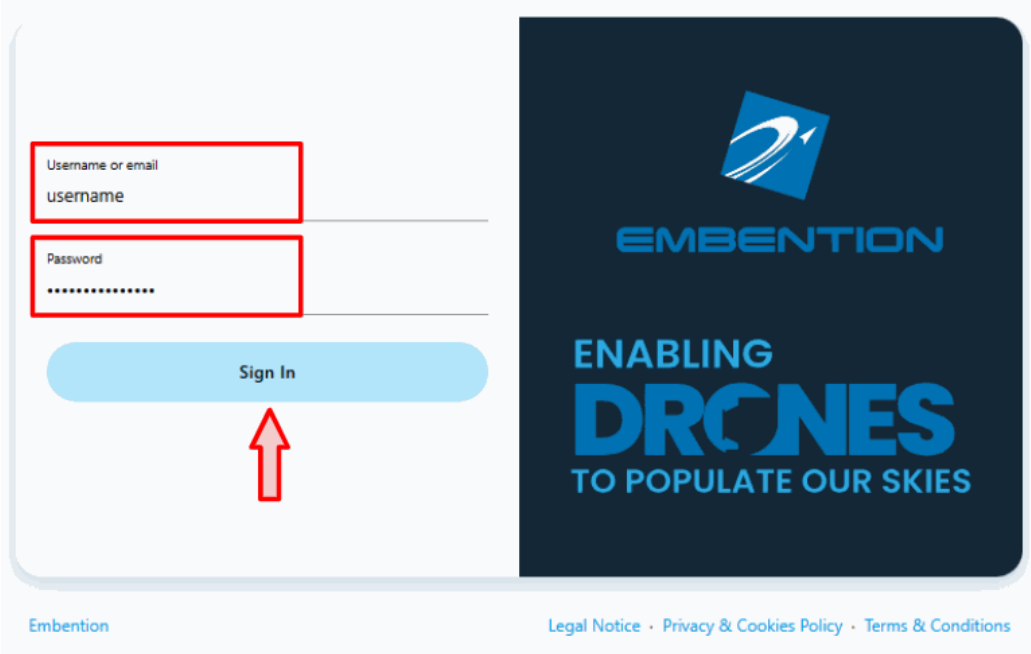
Terms and Conditions

Users can consult the 'End User License Agreement (**EULA**)' by simply clicking on this button.

Log in

Authentication is required before any operation.





Login

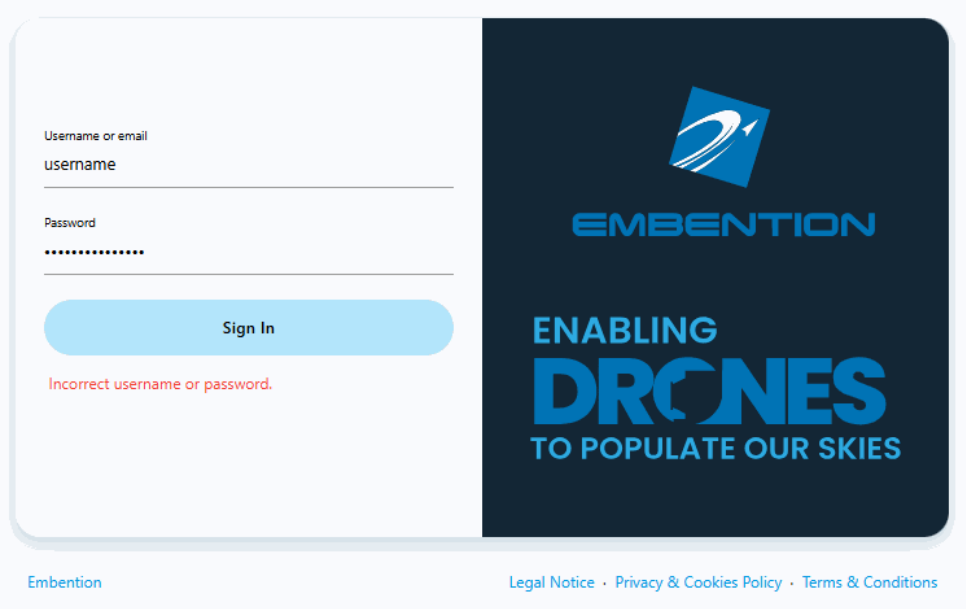
After clicking **Sign in**, a new window will open for users to enter their credentials. If users do not have credentials, do not know what they are or have any problems logging in with them, please contact the support team via the

Joint Collaboration Framework opening a Ticket or contact sales@embention.com.

If the login is successful, the screen will change as shown below:

Login successful

> ⚠ **Warning**
>
> If incorrect credentials are entered, the system will display the error message Incorrect username or password.



2. Mission

The common way to work with **Veronte Autopilot** is to create missions.

Missions can be created and managed through the 'mission toolbar' (left side of the menu (2)).

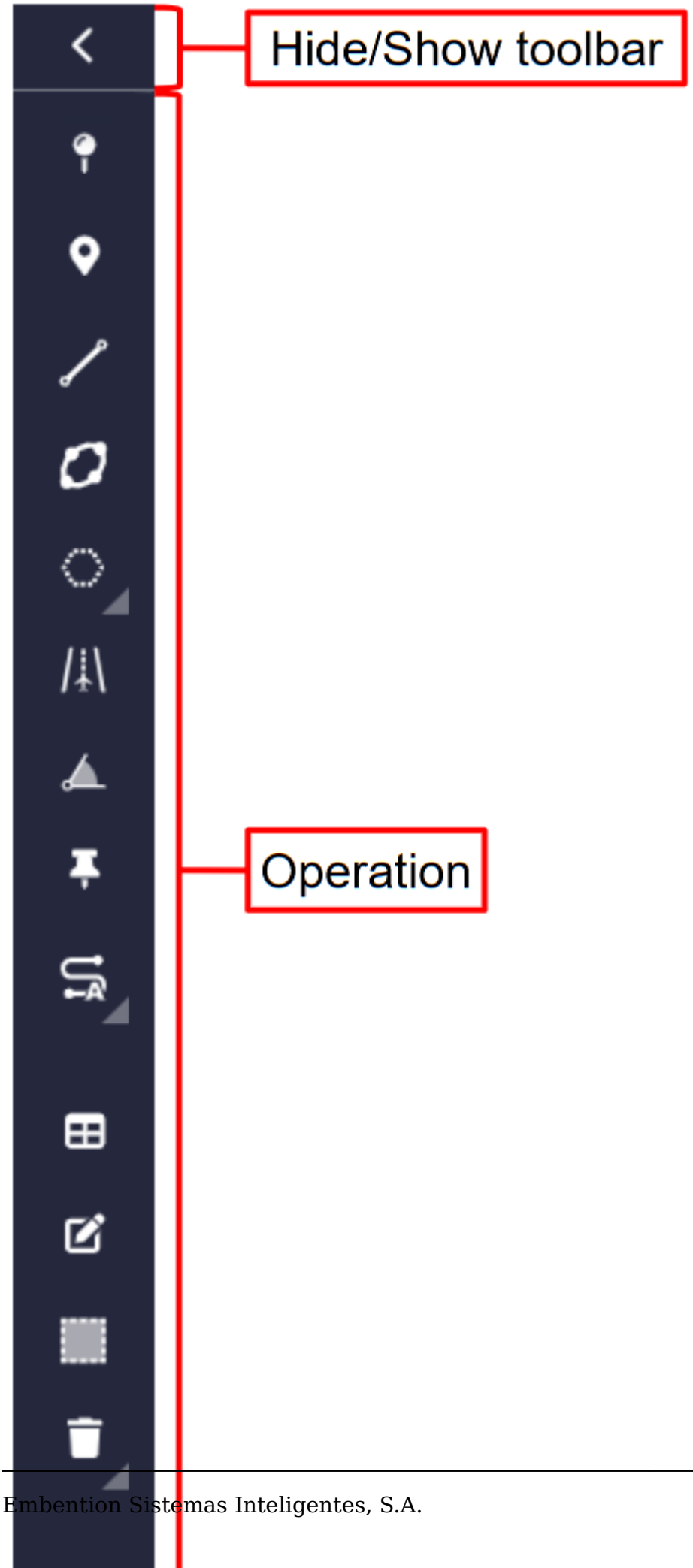
Mission toolbar

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 in the **Operation folder of the PDI configuration files** when the user saves the changes. For more information on the folders containing the PDI files, see the initial menu of the Configuration section of the **1x PDI Builder** user manual.

> ⓘ **Note**
>
> First, make sure the **Autopilot 1x** unit where the mission has to be uploaded is selected (Platform connection status - Platform section of this manual).

Next, a detailed description of the toolbar is given. It is divided into different functionalities:


Hide/Show toolbar
Operation

Mission toolbar functionalities

Operation

All the tools in this 'section' allow the user to design the mission, such as waypoints, segments, marks, runways, etc.

> **Important**
>
> Whenever any of these elements are created/modified, it is necessary to **Save** and **Upload** them into the Autopilot 1x configuration.
>
> For more information about these actions, refer to the Operation actions - Operation section of this manual.

Mission Point

Mission Points are points with absolute position to which a relative route can be associated.

Use the Mission Point tool and click (left-click) on the map to create new Mission Points. Then, when users have created all the desired Mission Points, **right-click on the map** to close/finish the 'creation event'.

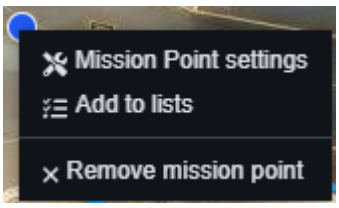
To move Mission Points, it is first necessary to activate mission editing by clicking on the icon. Then, simply drag it to the desired position.

In relation to Mission Points, the following actions can be performed with the mouse:

- **Clicking once** on the Mission Point will display a summary of the **Mission Point information**: Mission Point ID, coordinates and elevation:

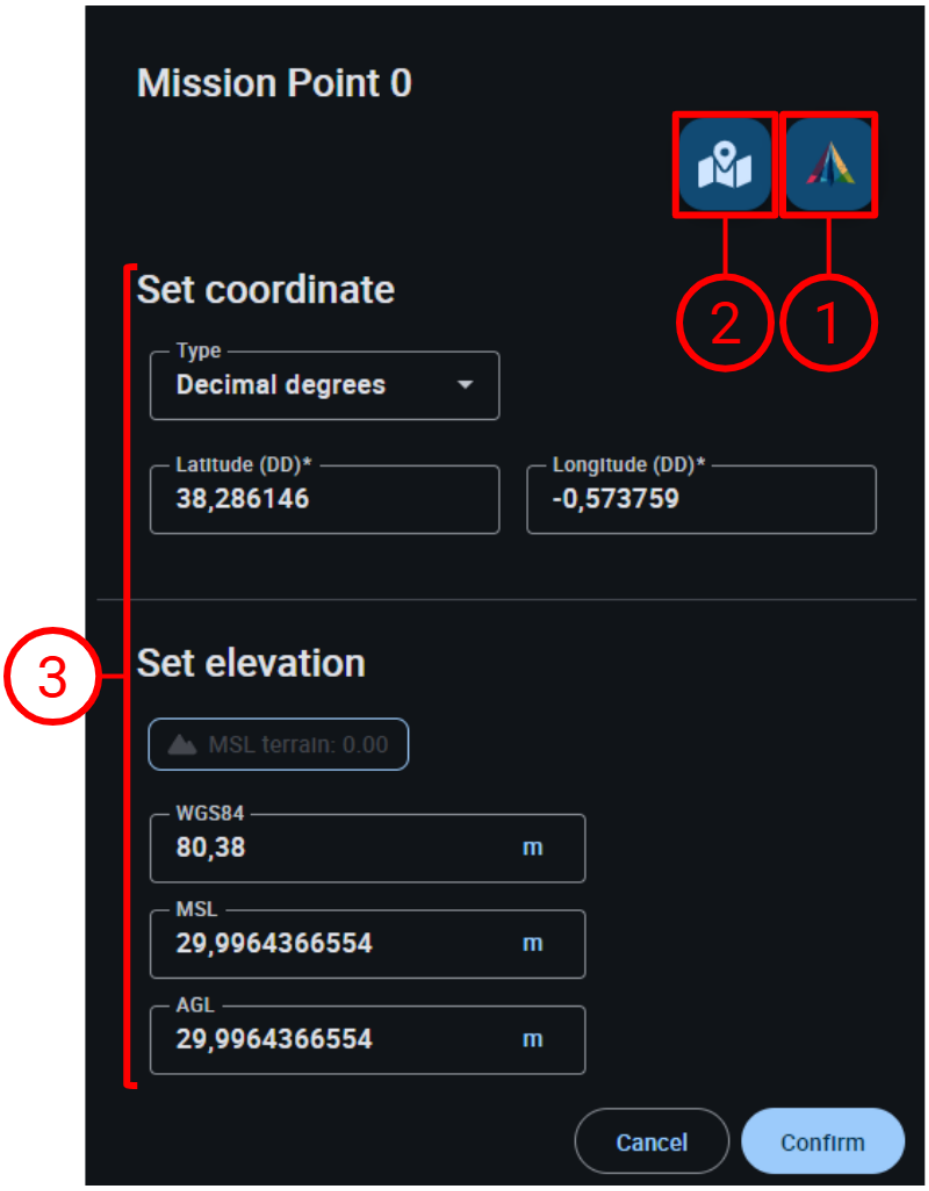
Mission Point information

- **Right-clicking** on the Mission Point, users will access its **options**:



Mission Point options

  - **Mission Point Settings**: The user can change the configuration of the Mission Point (coordinates and elevation) in the menu displayed here:



Mission Point configuration

There are 3 ways of defining the position of an Mission Point manually:

1. **Matching it to the platform position**. By selecting the icon, the Mission Point coordinates and elevation will be automatically updated with those of the actual platform's actual position.
2. **Selecting a position in the map**. Simply click on this icon , then click on the desired point on the map and the Mission Point coordinates and elevation will be updated automatically.

3. Entering the exact coordinates and elevation here:
    - **Set coordinate**: The coordinates can be set in:
        - Decimal Degrees → Latitude (DD) and Longitude (DD)
        - Degrees → Latitude (DMS) and Longitude (DMS)
        - UTM → x (Easting), y (Northing), Zone and Hemisphere
        - MGRS (Military Grid Reference System)
    - **Set elevation**: Whichever way the user defines the **altitude** (WGS84, MSL and AGL), **Veronte Ops** will calculate the other 2 ways.

      That is, if the user defines a waypoint at X m in AGL, **Veronte Ops** will automatically calculate what that value is in WGS84 and MSL.

- **Remove Mission Point**: Deletes the Mission Point.

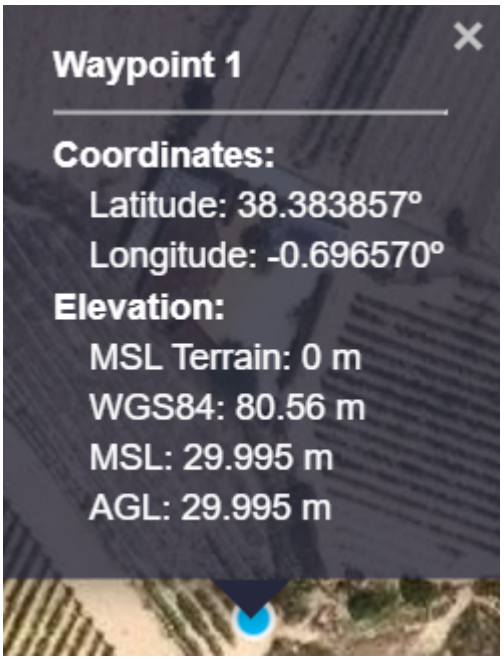
Waypoint

Use the Waypoint tool and click (left-click) on the map to create new waypoints. Then, when users have created all the desired waypoint, **right-click on the map** to close/finish the 'creation event'.

To move waypoints, it is first necessary to activate mission editing by clicking on the icon. Then, simply drag it to the desired position.

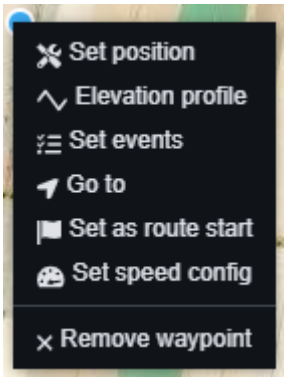
In relation to waypoints, the following actions can be performed with the mouse:

- **Clicking once** on the waypoint will display a summary of the **waypoint information**: Waypoint ID, coordinates and elevation:



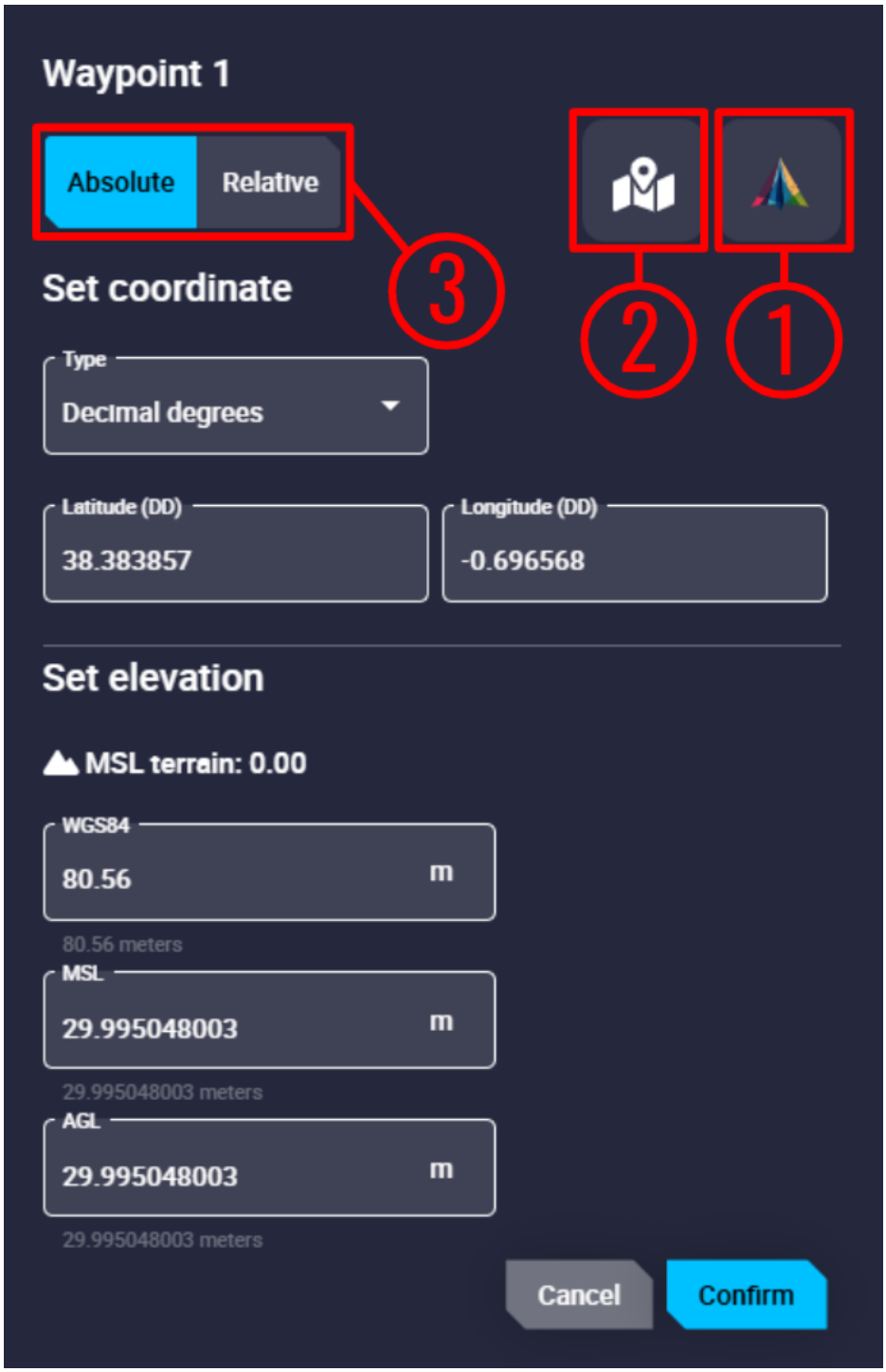
Waypoint information

- **Right-clicking** on the waypoint, users will access to its **options**:



Waypoint options

  - **Set position**: The user can change the configuration of the waypoint (coordinates and elevation) in the menu displayed here:

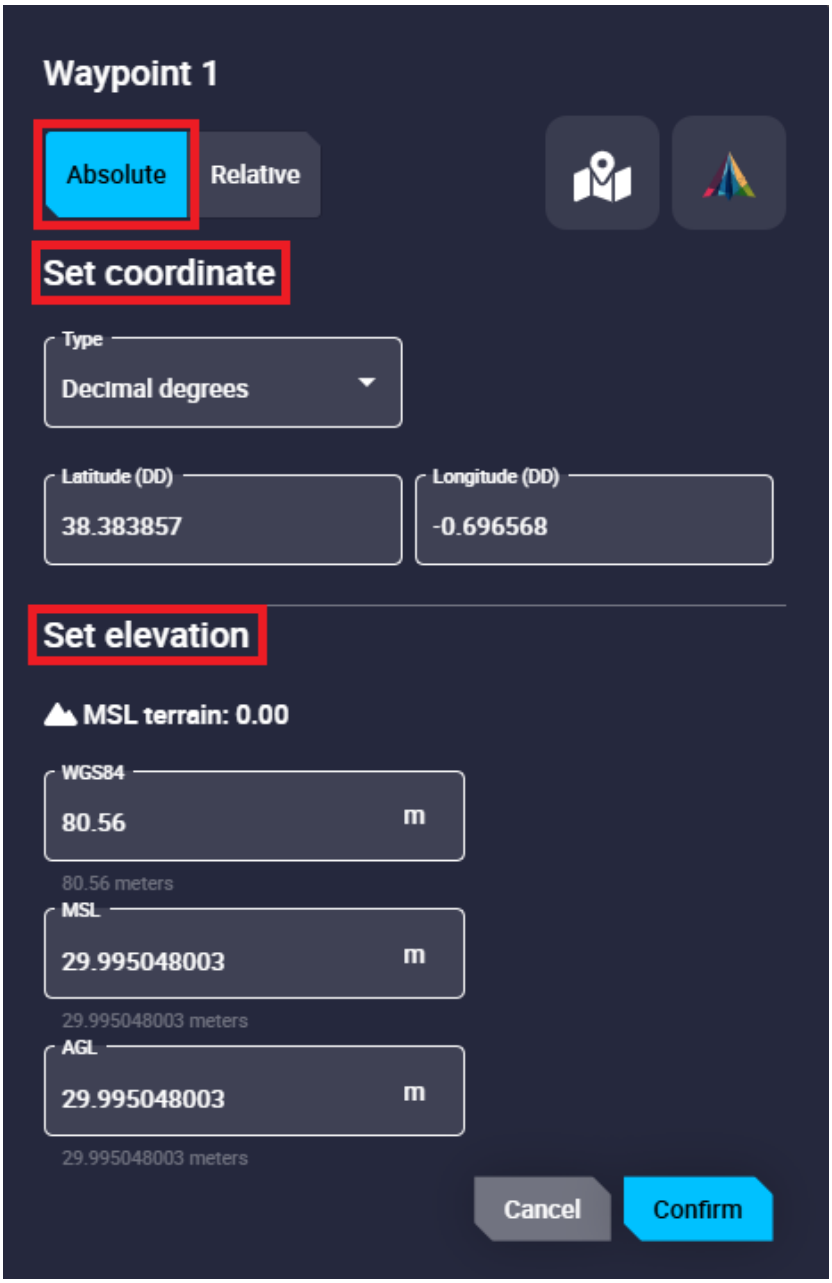


Waypoint configuration

There are 3 ways of defining the position of a waypoint manually:

1. **Matching it to the platform position**. By selecting the icon, the waypoint coordinates and elevation will be automatically updated with those of the actual platform's actual position.

2. **Selecting a position in the map**. Simply click on this icon , then click on the desired point on the map and the waypoint coordinates and elevation will be updated automatically.
3. Entering the exact coordinates and elevation here:
   - **Absolute**:



Absolute waypoint position

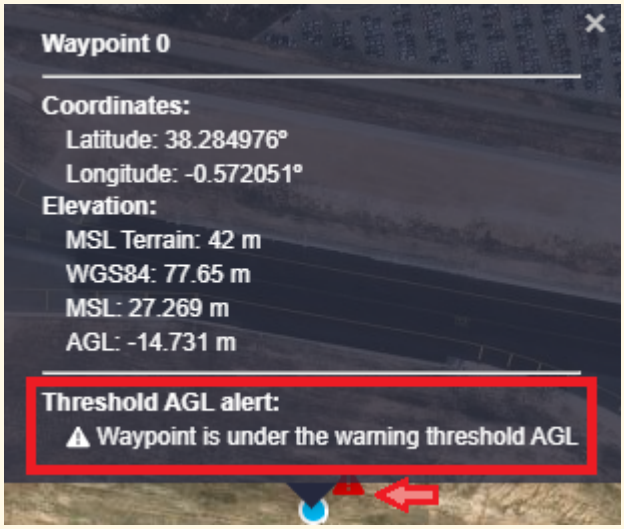
   - **Set coordinate**: The coordinates can be set in:
     - Decimal Degrees → Latitude (DD) and Longitude (DD)
     - Degrees → Latitude (DMS) and Longitude (DMS)
     - UTM → x (Easting), y (Northing), Zone and Hemisphere

        - MGRS (Military Grid Reference System)
    - **Set elevation**: Whichever way the user defines the **altitude** (WGS84, MSL and AGL), **Veronte Ops** will calculate the other 2 ways.

      That is, if the user defines a waypoint at X m in AGL, **Veronte Ops** will automatically calculate what that value is in WGS84 and MSL.
- **Relative**: In this case, the position of the waypoint is relative to another point.

  That point could be the **UAV position**, a **Desired position**, a **Track position** or an **Mission Point**. They are indicated through North, East and Down.

Relative waypoint position

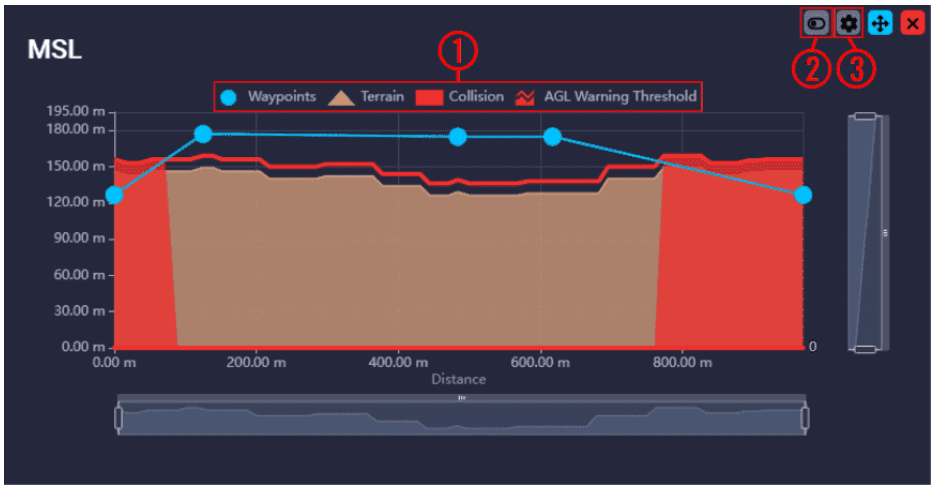
> ⚠ **Warning**
>
> If the elevation of a waypoint is set below the terrain altitude, the user will be able to detect it quickly as a warning symbol will appear next to the waypoint. In addition, it will also be indicated in the waypoint information as "Threshold AGL alert":
>
> 
>
>
> **Waypoint - Warning below terrain**

- **Elevation profile**: To be able to consult the elevation of the route, it is required to have installed the **Veronte Terrain Provider** application (for more information about this, see Additional apps section of this manual).

  Here users can see an estimate of the height of the terrain and the height of the route to be taken as well as the collisions that could occur.

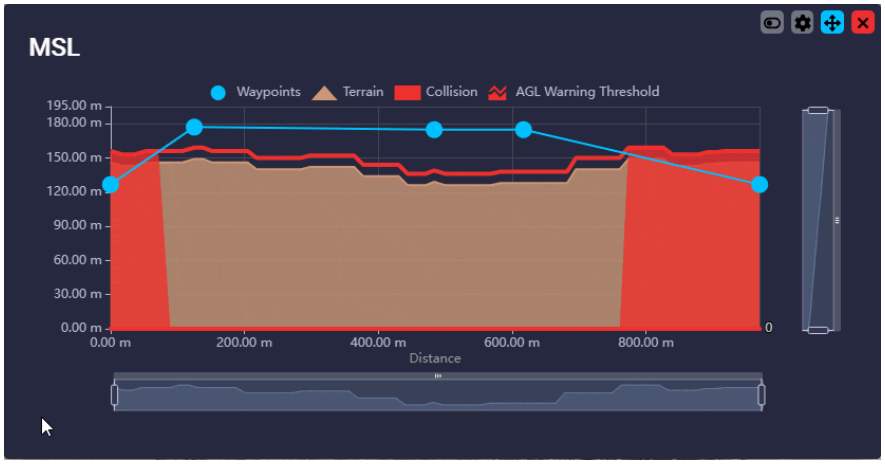
  In addition, users can set up a **Warning Threshold**, which will be represented as a red warning line over the terrain. Fore more information, see Map settings - Veronte Ops configuration section of this manual.

  By clicking on it, the following menu will appear:



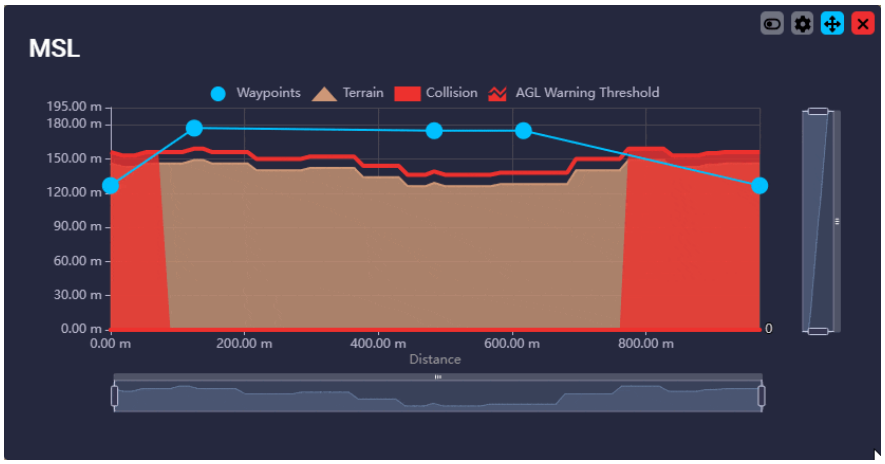
Elevation (MSL) menu

1. This legend showing waypoints, terrain, collision and warning threshold is an 'interactive' legend, i.e. users can choose which of these elements are displayed or not by simply clicking on them:



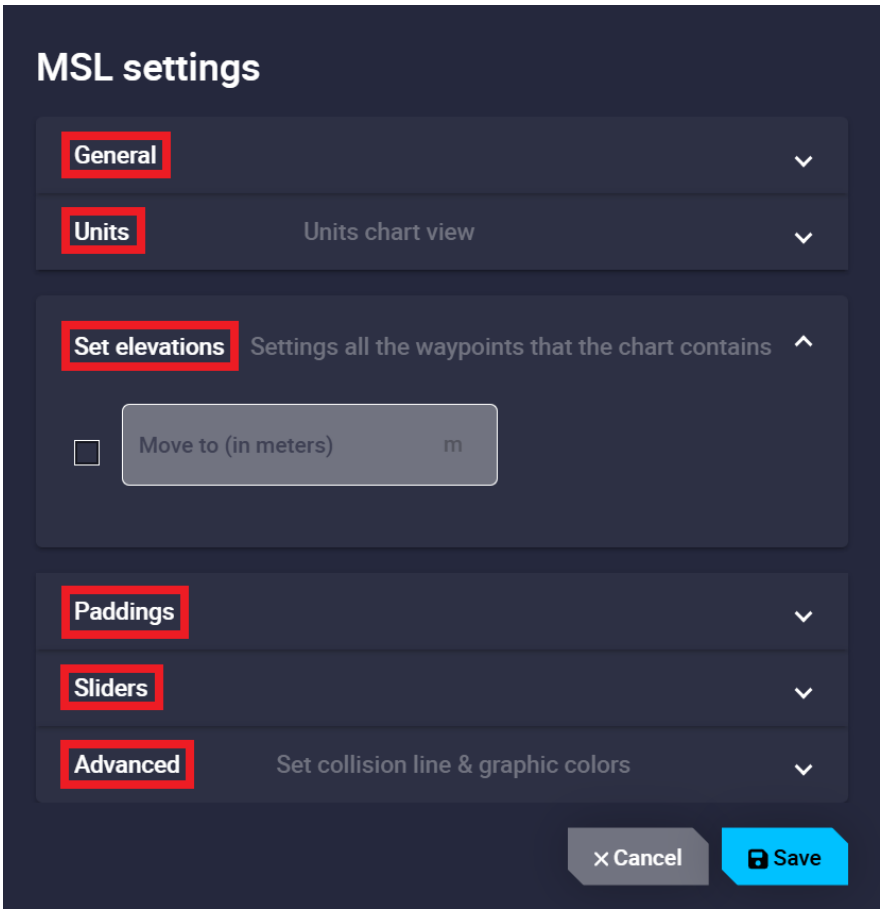
Elevation (MSL) menu - Legend

2. **Drag** button: When enabled ( ), users can modify the altitude of the waypoints by simply clicking and dragging them. If it is disabled ( ), it is not possible to move them:



Elevation (MSL) menu - Dragging waypoints

3. **Settings** button: Displays a new window with some parameters that can be customized:



Open elevation menu - Settings

- **General**: An offset can be set for Y axis.
- **Units**: Units of the elevation and distance of the chart view can be set.
- **Set elevations**: The altitude of all waypoints on the route can be modified.
- **Paddings**: The size of the grid can be adjusted as desired.
- **Sliders**: X and Y axes sliders can be hidden by disabling them.
- **Advanced**: The collision line and chart colors can be modified:

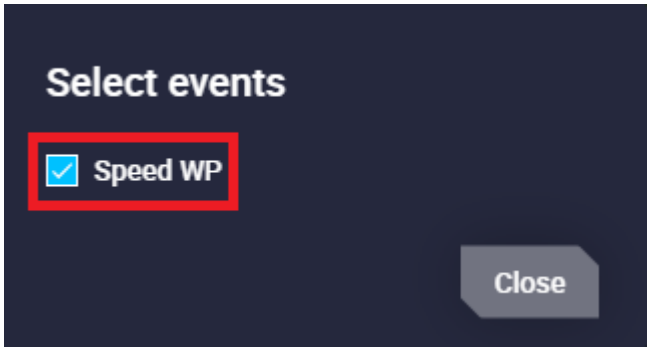
Elevation (MSL) menu - Advanced settings

- **Set Events**: This option allows to associate this waypoint to a specific event.

  This works as follows, in this panel the user links this waypoint to an "Event Waypoint". This Event Waypoint is the one that has to be linked to the desired event in the Automations menu of the **1x PDI Builder software**.

  > **Important**
  > To do this, it is first necessary to have defined (renamed) an Event Waypoint in the **UI menu of the 1x PDI Builder software**, as described and explained in the corresponding section of the **1x PDI Builder** manual, click here to access it.

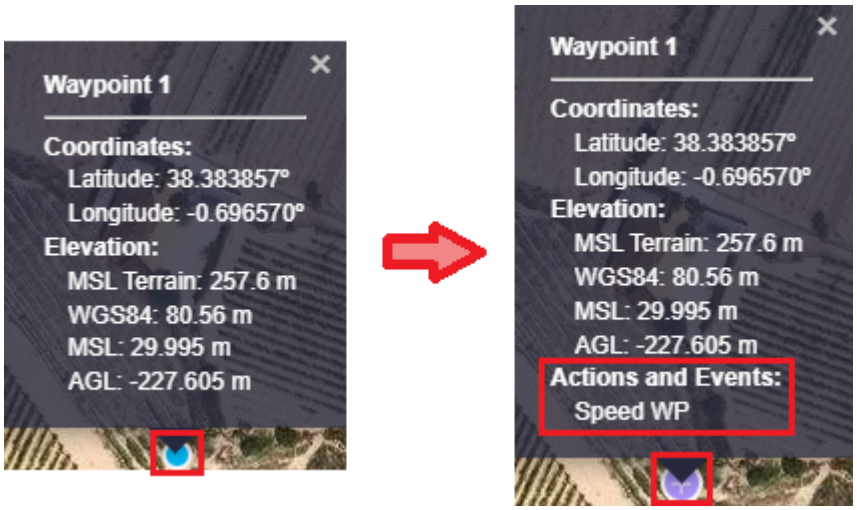
  Simply check the desired Event Waypoint to be associated:



Waypoint options - Set Events

> **Note**
> - Users can associate a waypoint to as many Event Waypoints as there are created (a maximum of 16 Event Waypoints can be defined in the system).
> - Different waypoints can be linked to the same Event Waypoint.

Moreover, if the Event Waypoint that is linked to the waypoint has an associated icon, the icon will appear at this waypoint. In addition, when clicking on it, the name of the Event Waypoint defined by the user will appear in its description. An example is shown below:

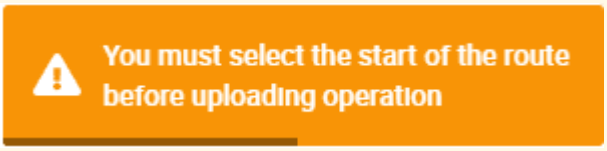


Waypoint information - Set Events

- **Go to**: By clicking here, the platform will fly to this waypoint. If the waypoint belongs to a mission, the platform will continue that mission after reaching it.

> **Note**
> This option only appears when a Veronte Autopilot 1x is connected and the operation, which the user is working on, is stored in the autopilot.

- **Set Start Route**: Users must define where they want the cruise route to start by defining the waypoint as such with this option. The waypoint will turn of green color.

> ⚠ **Warning**
>
> If users do not define any waypoint as the 'Start Route', the operation will not be uploaded to the autopilot configuration and the following warning notification will be displayed:
>
> 
>
>
> **Set start route - Warning notification**

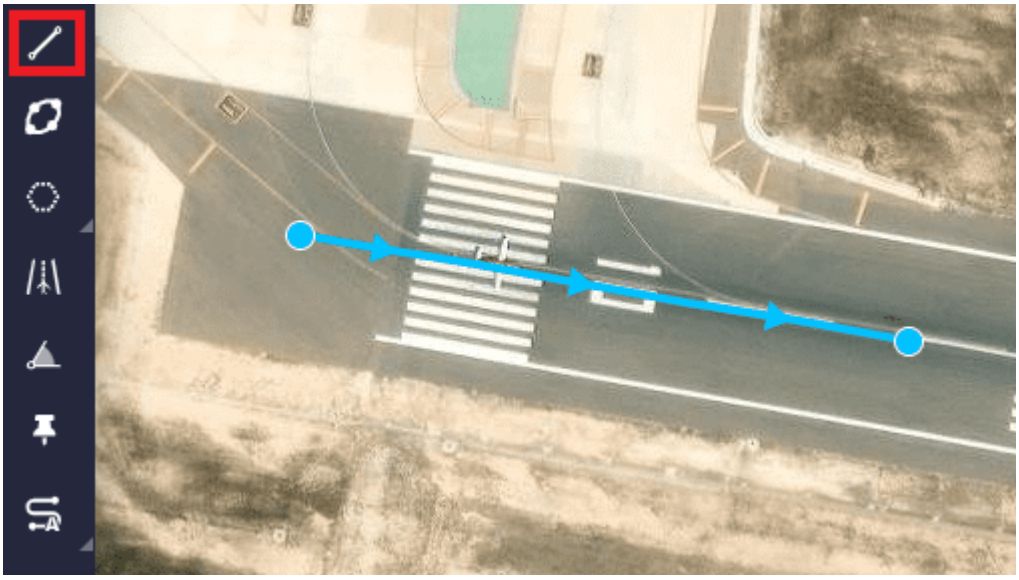
- **Set speed configuration**: Set Speed function defines the velocity parameters that the UAV must adopt when reaching or passing a specific waypoint.

Waypoint information - Set speed configuration

  - **Speed Type**: Defines the physical reference for the velocity.
  - **Waypoint to speed**: Numerical value of the target speed.
  - **Mode**: Determines how the command is applied to the flight profile.
- **Remove Waypoint**: Deletes the waypoint.

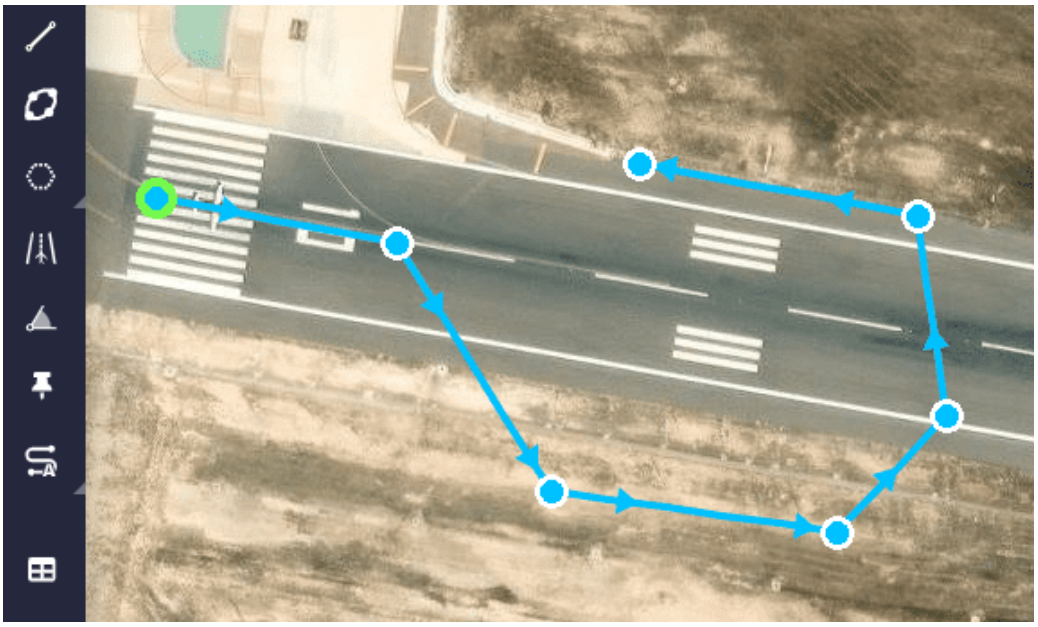
Segment

To create a new segment, click on the segment icon and then select on the map the point where the segment will start. To end the track, users have to **double click** directly with the **left mouse button** to create the last waypoint of the segment. Then, after creating the last waypoint, **right-click on the map** to close/finish the 'creation event'.



Segment

In addition, users can **concatenate two or more segments** by clicking on various points on the map with the **left button**, as shown in the figure below:

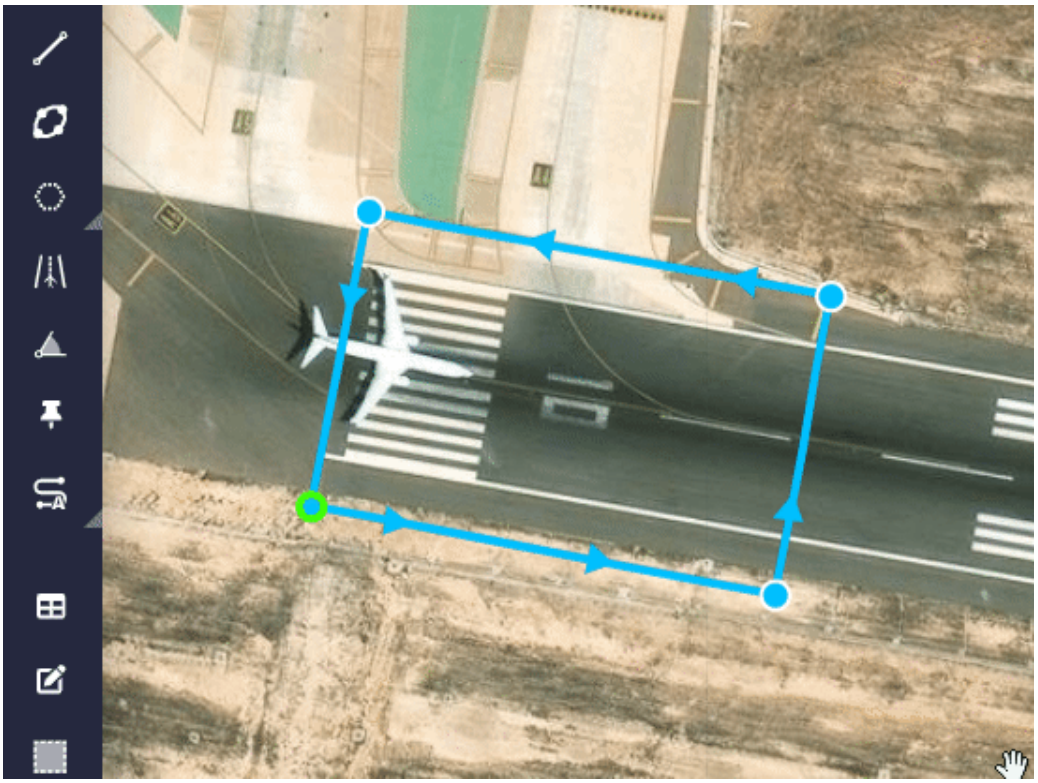


Route

Besides, to create a closed route, users must match the last waypoint of the segment with the first one:

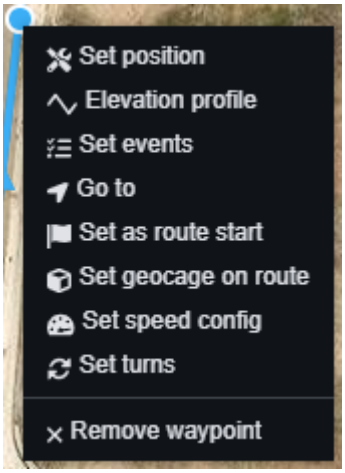
Closed route

It is also possibe to curve a segment. To do so, users must create the segment, edit the mission by clicking on the icon, and then move the **translucent orange waypoint** ( ) in the center of the segment. The distance moved will correspond to the radius of the curve:



Curved segment

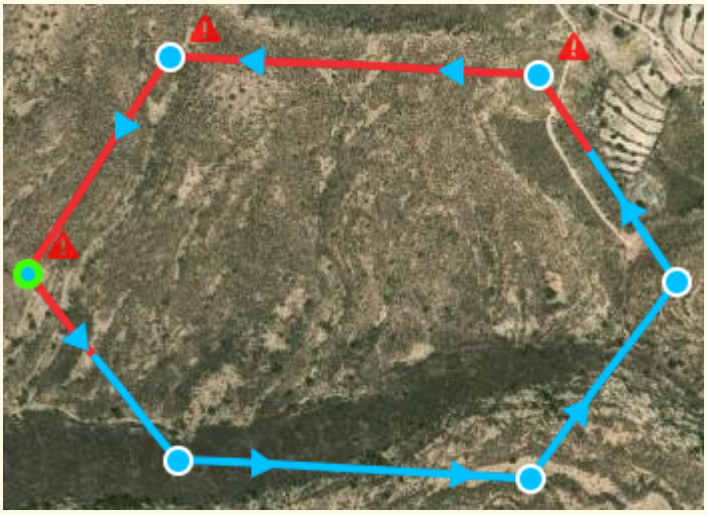
There is an extra option in the options for a waypoint that corresponds to the start of a curved segment:



Curved segment options

- **Set Turns**: This option allows the user to set a desired number of turns. After these turns the platform will continue with the defined mission. An example is shown below:

Curved segment - Turns

> ⚠ **Warning**
>
> When a segment is created, be careful with the height of the waypoints.
>
> Users can **check for collisions** between route waypoints and terrain **directly on the map** or **using the 'Open elevations' functionality** described above.
>
> To check directly on the map, if some of the waypoints are below the terrain altitude indicated by the meshes, a warning symbol will appear next to them and the affected segment will change its color to red. In addition, it will also be indicated in the waypoint information as "Threshold AGL alert".
>
> To solve this, change the altitude of the waypoints. An example is shown below:
>
> 
>
> **Segment under the ground altitude**

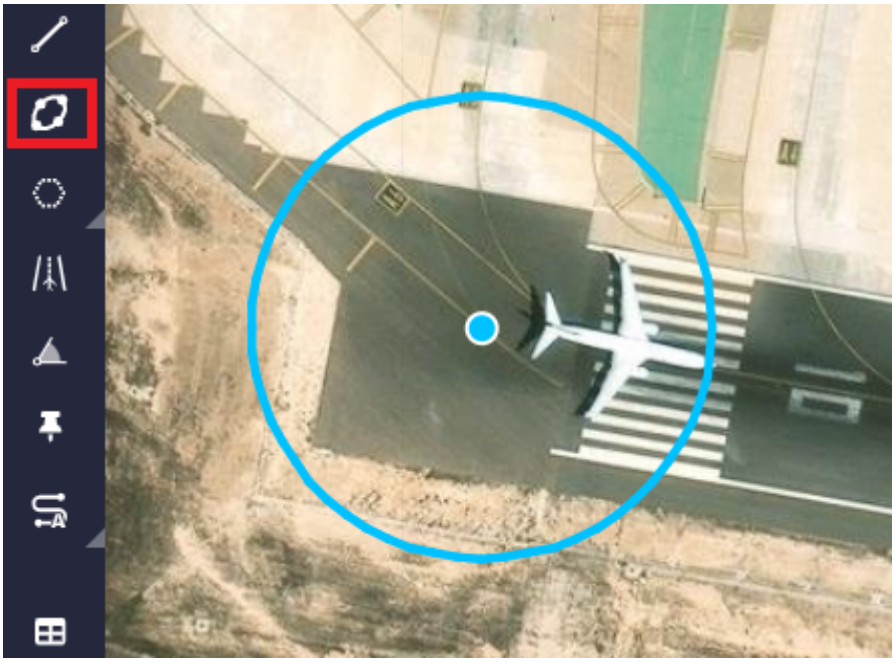
- **Set geocage on route**

Allows configuring a geocaging zone around the segment. For more information, consult Geocaging area.

Orbit

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. Then, when users have created all the desired orbits, **right-click on the map** to close/finish the 'creation event'.

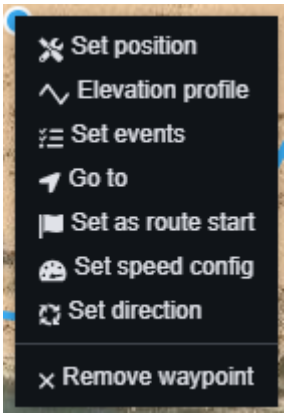


Orbit

In addition, the initial circular orbit can be converted into an ellipse by modifying the length of the axes and can also be rotated. To do this, edit the mission by clicking on the icon, then move the **4 translucent orange waypoints** ( ) as desired to turn them into an ellipse and to rotate it, click and rotate the small blue circle ( ):

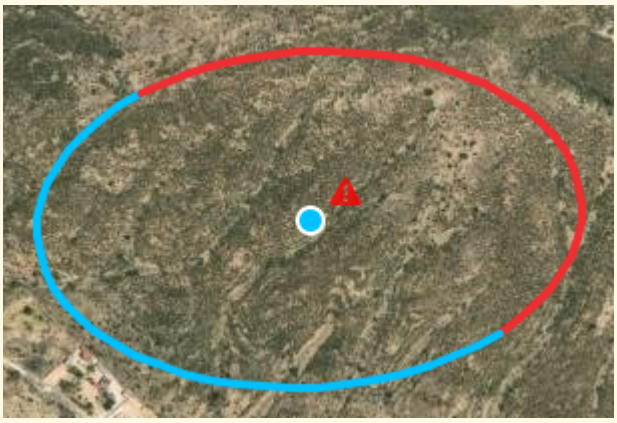
Ellipse orbit

Besides, there is an extra option in orbits regarding waypoints and segments:



Orbit options

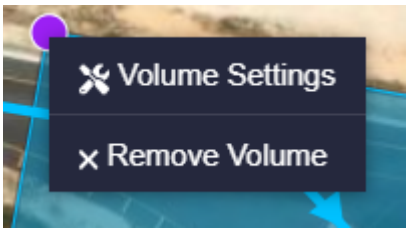
- **Set Direction**: It is possible to select the direction of the loiter: **Auto**, **Clockwise** and **Anticlockwise**.

> ⚠ **Warning**
>
> When creating an orbit, be careful with its altitude and the ground level (all the points of the orbit will be at the same altitude).
>
> Users can **check for collisions** between orbit and terrain **directly on the map** or **using the 'Open elevations' functionality** described above.
>
> To check this directly on the map, if any part of the orbit is below the terrain altitude indicated by the meshes, a warning symbol will appear next to the waypoint and the affected segment will change its color to red. In addition, it will also be indicated in the waypoint information as "Threshold AGL alert".
>
> To solve this, change the altitude of the waypoint. An example is shown below:
>
> 
> **Orbit partially below ground level**

Shape

Within this option areas/volumes with different shapes can be created: prisms, spheres, cylinders and geocaging areas.

The vertices (waypoints) that compose these volumes are all configured in the same way. **Right-click** on them to access their options:



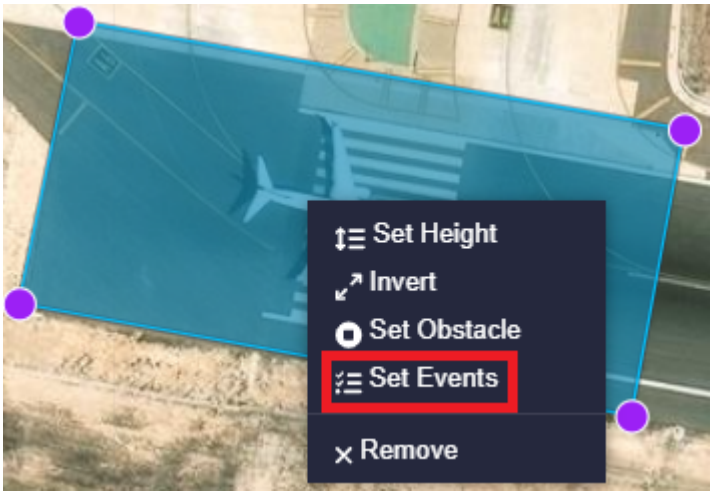
Volume vertex options

- **Volume Settings**: The user can change the configuration of the volume vertex (coordinates and elevation) in the menu shown here. Its configuration is the same as for a waypoint (a detailed explanation of this has been described in the Waypoint section above).

  > ⓘ **Note**
  > The **geocaging area** does not have this option.

- **Remove Volume**: Deletes this volume.

Furthermore, these volumes have a shared configuration option, "Set Events". This option allows to associate this volume to a specific event.

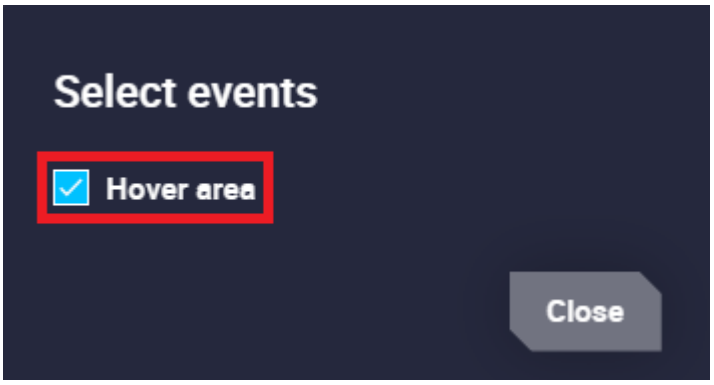


Volume options

This works as follows, in this panel the user links this volume to an "Event Volume". This Event Volume is the one that has to be linked to the desired event in the Automations menu of the **1x PDI Builder software**.

> **Important**
>
> To do this, it is first necessary to have defined (renamed) an Event Volume in the **UI menu of the 1x PDI Builder software**, as described and explained in the corresponding section of the **1x PDI Builder** manual, click here to access it.

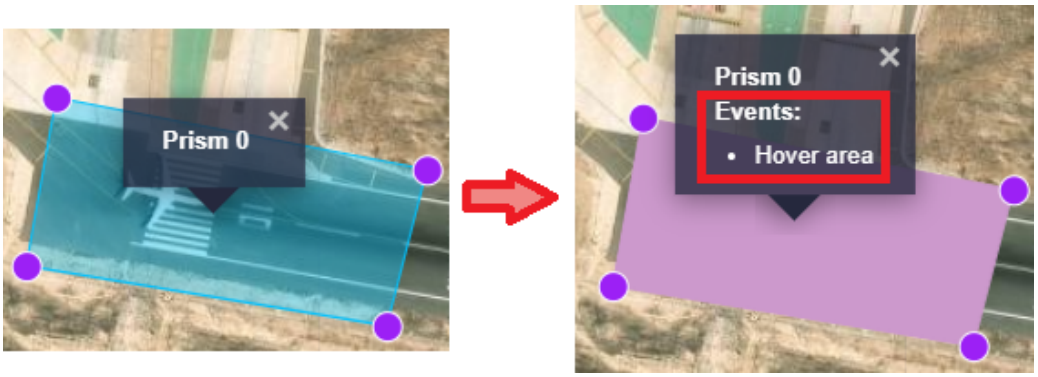
Simply check the desired Event Volume to be associated:



Volume options - Set Events

> **Note**
>
> - Users can associate a volume to as many Event Volumes as there are created (a maximum of 60 Event Volumes can be defined in the system).
> - Different volumes can be linked to the same Event Volume.

Moreover, if the Event Volume that is linked to the volume has a custom color, the volume will turn that color. In addition, when clicking on it, the name of the Event Volume defined by the user will appear in its description. An example is shown below:



Volume information - Set Events

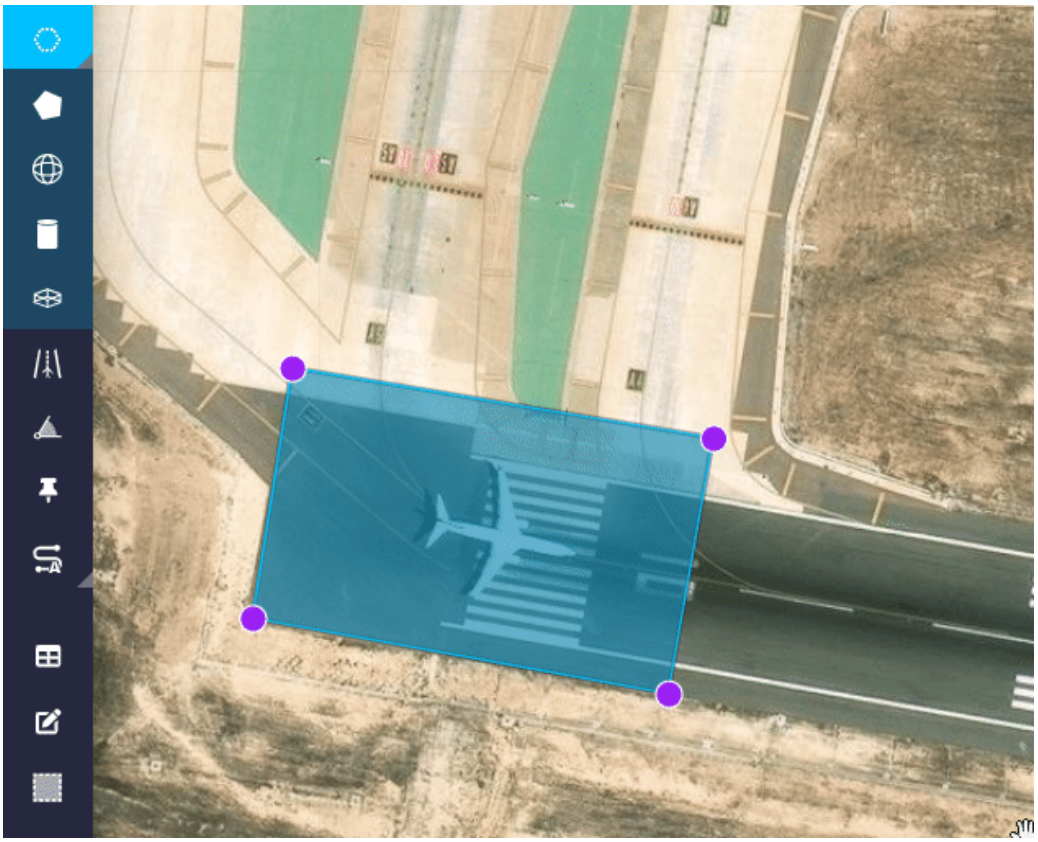
Prism

This tool is used to determinate an area in which an action is wanted to be performed. When the aircraft enters or leaves the prism an event may be triggered and it can be used to start an automation.

To add a prism, click on its icon, click on the desired points on the map and match the last waypoint of the prism with the first one. Then, when users have created all the desired prisms, **right-click on the map** to close/finish the 'creation event'.

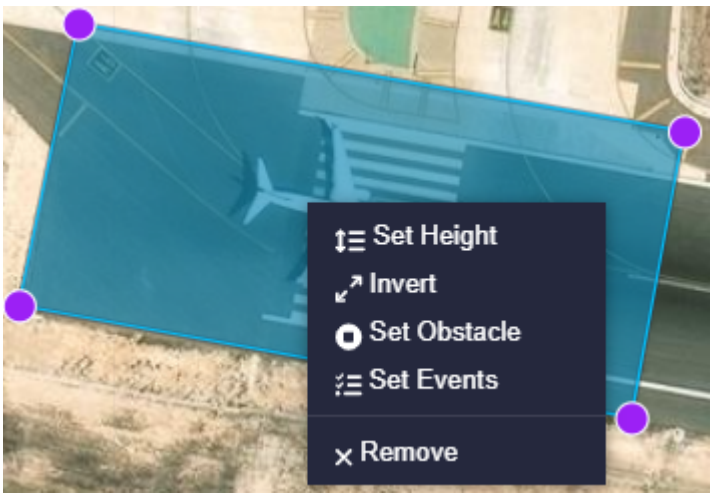
Prism

Like the other elements, it can be edited to change its shape by moving its vertices. Click on the icon:



Prism shape

Right-clicking on the prism, users will access to its **options**:



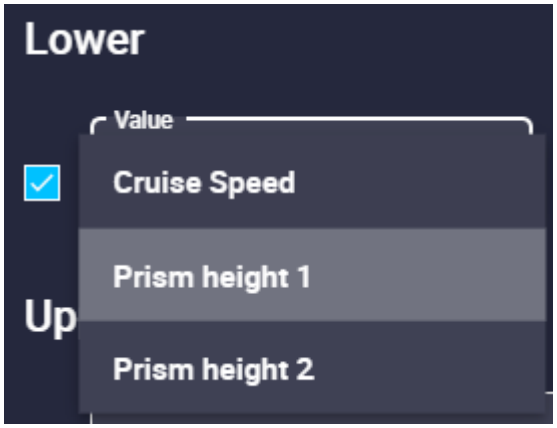
Prism options

- **Set Height**: Users can set the upper and lower heights of the prism. By default, prisms have infinite lower and upper boundaries.

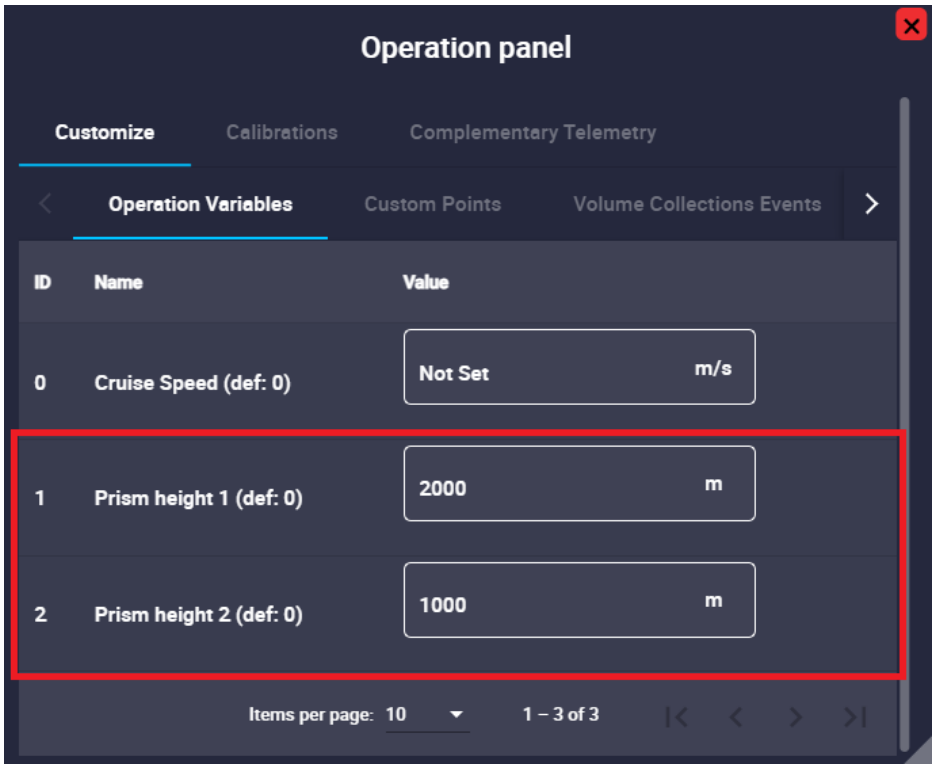
Prism options - Set Height

As can be seen in the figure above, the lower and upper limits can be set as absolute or relative:

- **Relative** icon: The altitude is relative to an **Operation Variable**, previously defined in **1x PDI Builder**. But, it is required to set the value of the variable in the Operation Variables of the **Operation Panel**. In addition, it is necessary to establish whether it is relative to AGL, MSL, WGS84 or to a Fld (feature) that has to be selected.



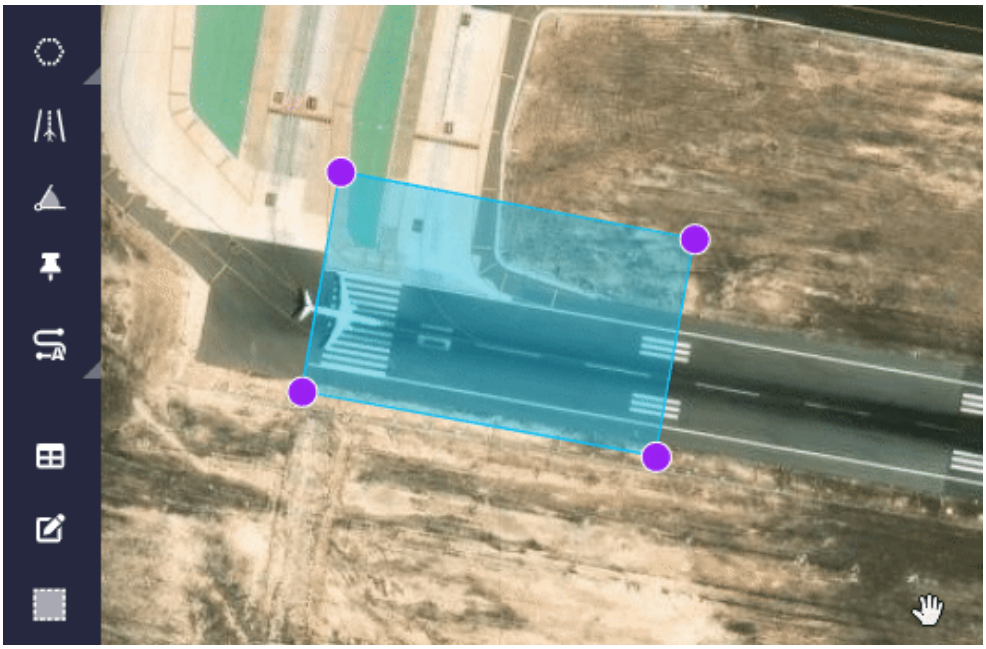
Prism options - Height relative



Prism options - Operation Panel

- **Absolute** icon: The value must be entered manually and it is also necessary to establish whether it is relative to AGL, MSL, WGS84 or to a Fld (feature) that has to be selected.

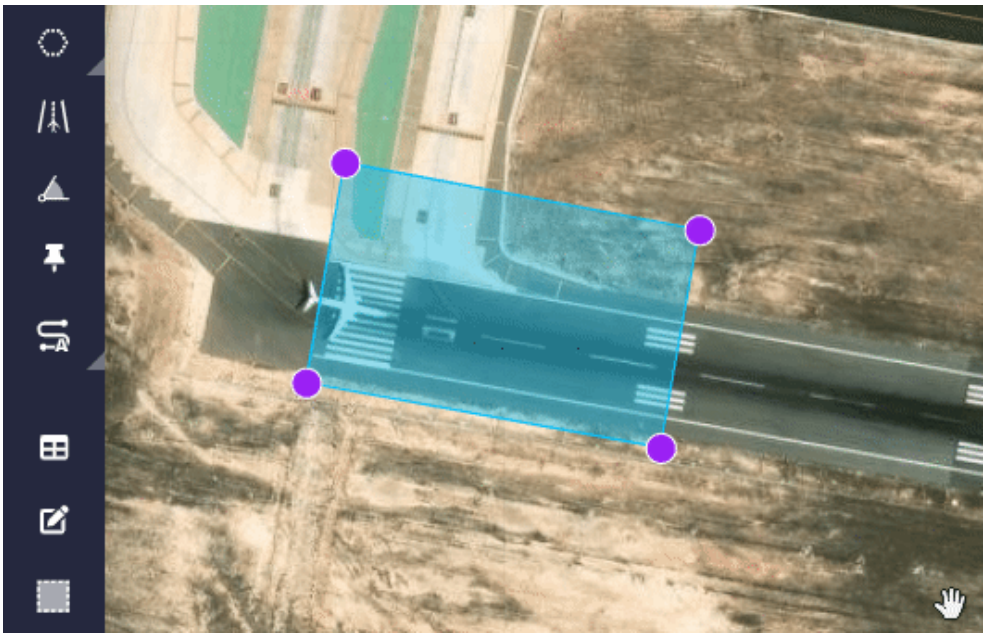
- **Invert**: This inverts the defined area.



Prism options - Invert

- **Set Obstacle**: A prism can become an obstacle, an area that is desired to avoid. This option has been implemented to be used as the **geofencing** funcionality.

> ⓘ **Note**
> When setting the obstacle, the option "**Set Obstacle**" changes to "**Set Prism**". This option turns the obstacle back into a prism.



Prism options - Set Obstacle

This tool allows to set an exclusion area on the map that can not be crossed by the RPAS. Main functions of the tool:

- Avoid collisions with obstacles as for example buildings, trees or antenna towers.
- Avoid flying restricted access areas.

What can happen with high speed aircrafts (Airplanes)?

It is possible that due to its speed, the RPAS will enter the obstacle area, but will immediately apply the corrections to exit the obstacle area and return to the path. The solutions to this problem is to 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 an obstacle area (staying very close to its center), at that moment the Ground Speed Vector and the Field Repulsion Vector have the same direction but opposite sign. This phenomenon causes a conflict and until the directions are different enough to allow the multirotor to move, it will remain in an indecision situation. To solve this problem the user can configure the obstacle area by leaving the center of the obstacle away from the path line.

- **Set Events**: Explained previously in the Shape section.
- **Remove**: Deletes the prism.

Sphere

This tool is used to determinate a spherical volume (3D) in which an action is wanted to be performed. When the aircraft enters or leaves the sphere an event may be triggered and it can be used to start an automation.

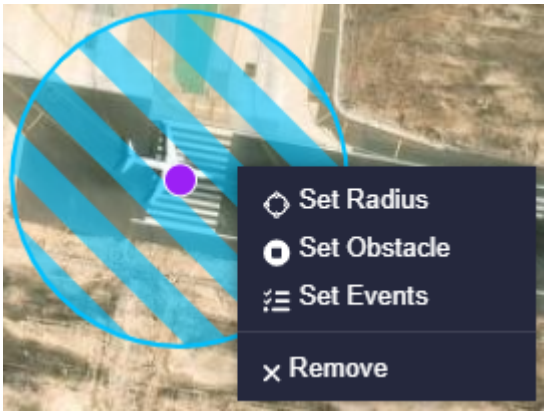
To add a sphere, click on its icon, click on the desired point on the map and drag it to set the radius. Then, when users have created all the desired spheres, **right-click on the map** to close/finish the 'creation event'.

Sphere

Like all other elements, it can be edited to move it or change its radius by clicking on the icon:

Sphere radius

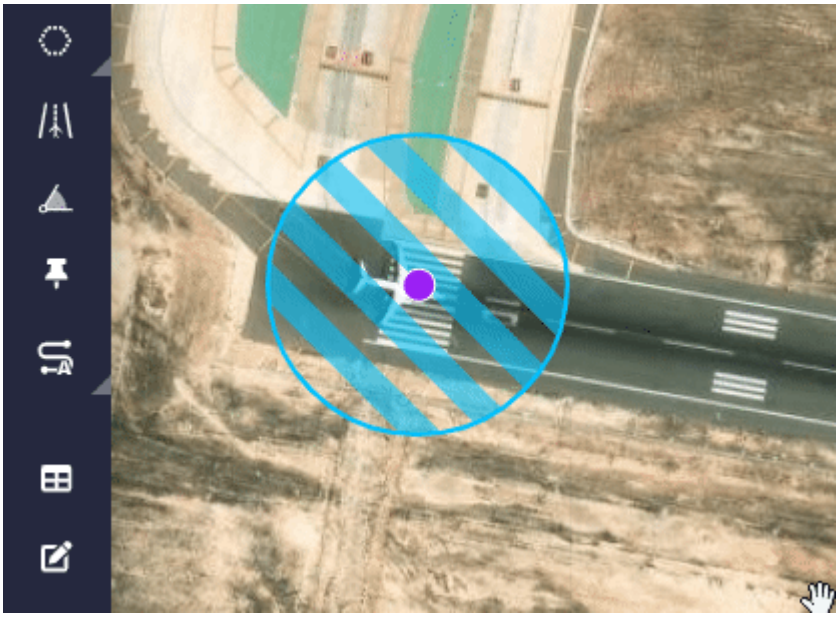
Right-clicking on the sphere, users will access to its **options**:



Sphere options

- **Set Radius**: To accurately set the radius of the sphere, users can enter the value manually.
- **Set Obstacle**: A sphere can become an obstacle, an space that is desired to avoid. As it is a 3D space, it allows the platform to avoid it by passing over it, taking into account the height of the obstacle. This option has been implemented to be used as the **geofencing** funcionality.

> ⓘ **Note**
> When setting the obstacle, the option "**Set Obstacle**" changes to "**Set Sphere**". This option turns the obstacle back into a sphere.



Sphere options - Set Obstacle

This functionality acts in the same way as when defining a prism as an obstacle, but taking into account **3 dimensions**. Therefore, the platform will be able to prevent the obstacle from passing over it.

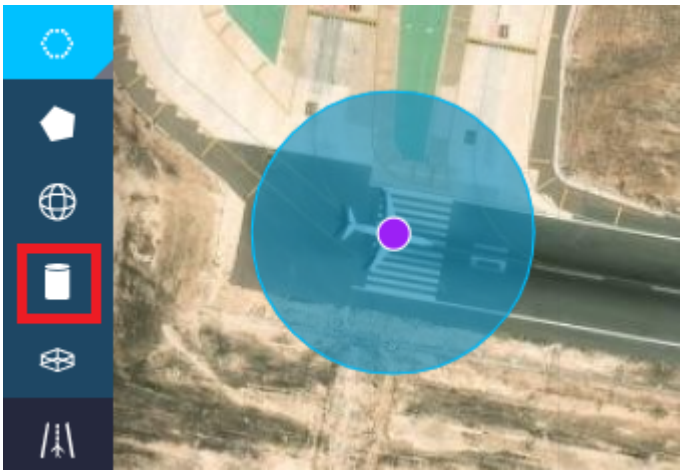
- **Set Events**: Explained previously in the Shape section.
- **Remove**: Deletes the sphere.

Cylinder

The way it works is very similar to **Prism**, described above.

This tool is used to determinate a cylindrical area in which an action is wanted to be performed. When the aircraft enters or leaves the cylinder an event may be triggered and it can be used to start an automation.

To add a cylinder, click on its icon, click on the desired point on the map and drag it to set the radius. Then, when users have created all the desired cylinders, **right-click on the map** to close/finish the 'creation event'.

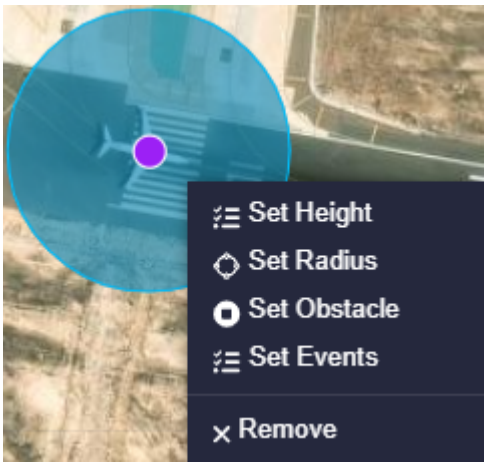


Cylinder

Like all other elements, it can be edited to move it or change its radius by clicking on the icon:

Cylinder radius

Right-clicking on the cylinder, users will access to its **options**:



Cylinder options

- **Set Height**: Users can set the upper and lower heights of the cylindrical area. By default, cylindrical areas have infinite lower and upper boundaries. It works in the same way as in the prism areas, described above.
- **Set Radius**: To accurately set the radius of the cylinder, users can enter the value manually.
- **Set Obstacle**: A cylinder can become an obstacle, an area that is desired to avoid. This option has been implemented to be used as the **geofencing** funcionality.

> ⓘ **Note**
>
> When setting the obstacle, the option "**Set Obstacle**" changes to "**Set Cylinder**". This option turns the obstacle back into a cylindrical area.

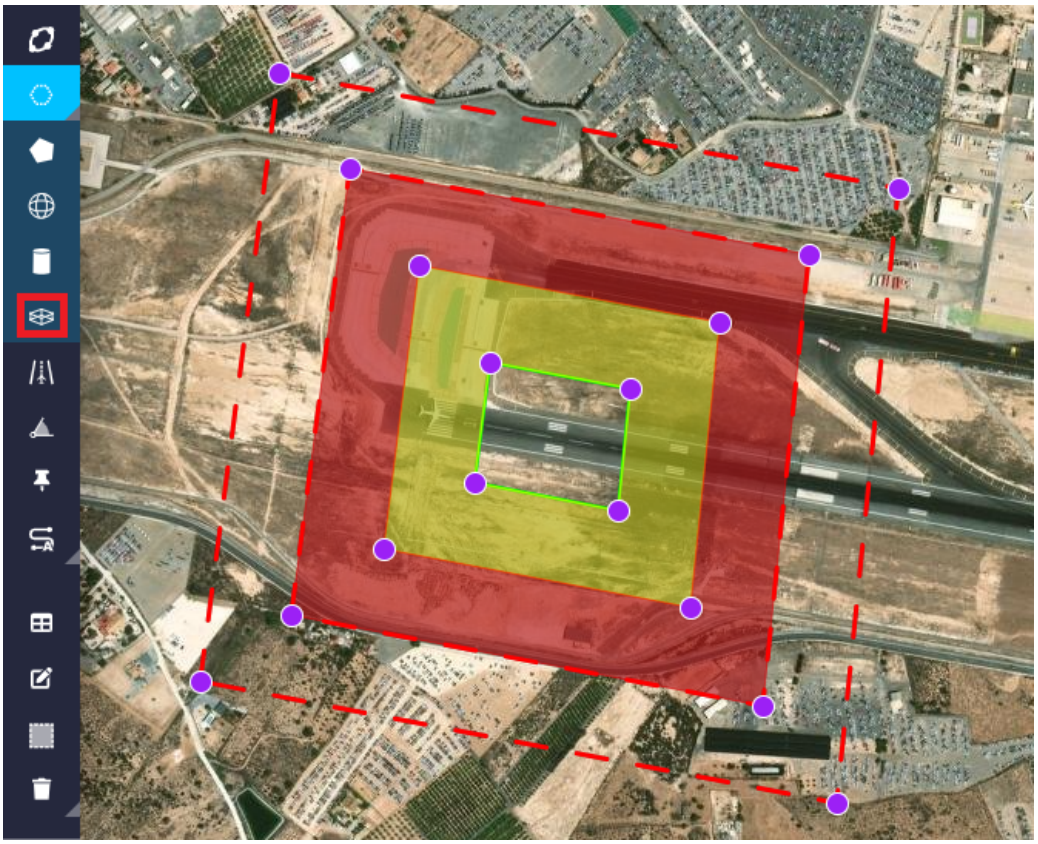
Cylinder options - Set Obstacle

This functionality acts in the same way as when defining a prism as an obstacle.

- **Set Events**: Explained previously in the Shape section.
- **Remove**: Deletes the cylinder.

Geocaging area

The geocaging function is an UAS function that helps the remote pilot to prevent the platform from exiting out of a volume defined (geocaging area) on purpose for this operation. This function operates in both horizontal plane and vertical axis.



Geocaging area

There are three grades defined to accommodate different levels of risk in case of exit. This function will provide, depending on the grade, one or several barriers to the exit. These barriers can be alerts to the remote pilot that will perform the adequate actions or, optionally, can be automatic actions.

In addition to the 3 areas to comply with the different risk levels, an additional zone has been defined:

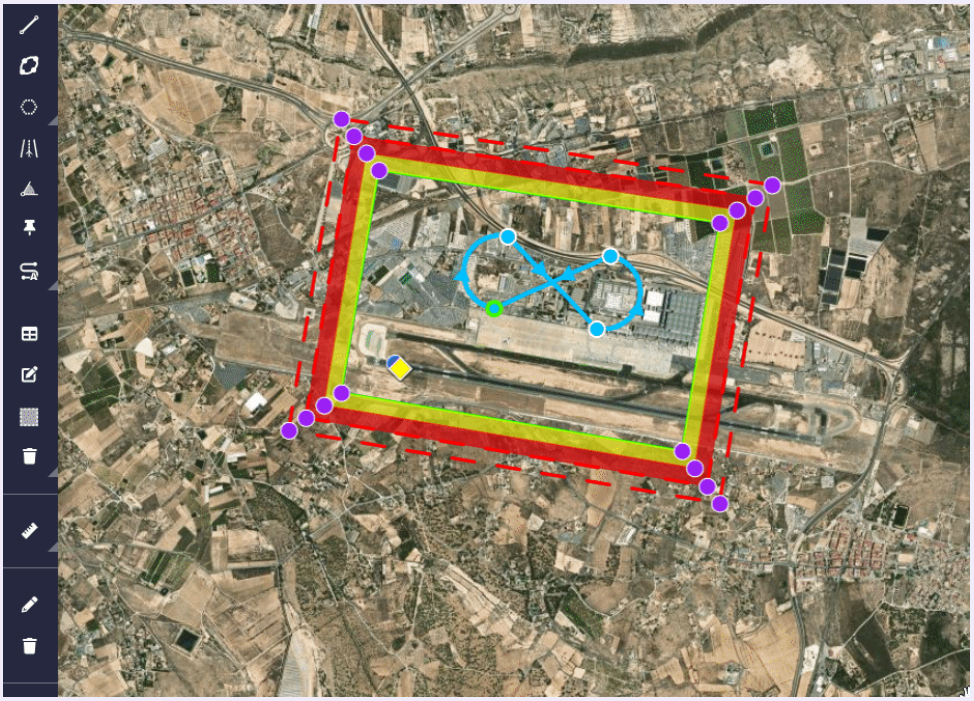
- Flight Geography Area: This area corresponds to the volume intended normally for the flight. It is delimited by green lines.
- Contingency Area: This area is the volume needed to accommodate **contingency alerts or actions**. It is yellow colored area delimited by red lines.
- Emergency Area: Volumen beyond the Contingency Area which is computed to accommodate **emergency warning alerts or actions**. This area is delimited by dashed red lines.
- Additional safety zone: Volume added after the computed Contingency and Emergency Areas, and before the "not to enter" volume. It is delimited by dashed red lines.

These contingency and emergency actions, as well as actions that can be added related to the Additional safety zone, are customizable. They have to be configured as any action associated to a volume from the Automations menu of the **1x PDI Builder** software.

An example of geocaging corrective actions that can be configured following the standard can be found in the Geocaging - Integration examples section of the **1x PDI Builder** user manual.

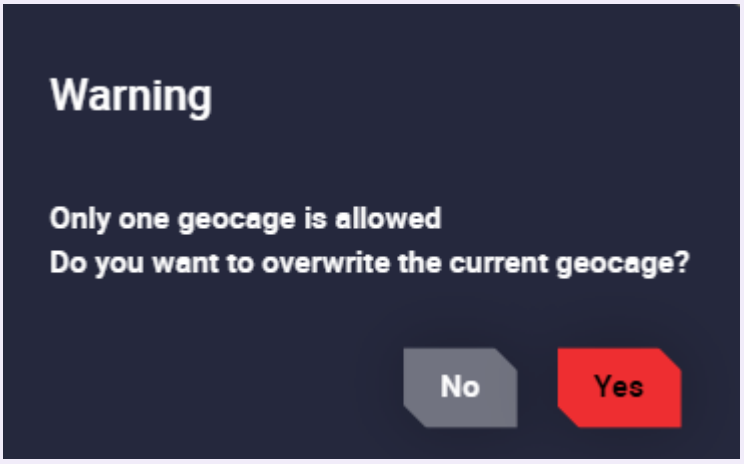
Important

Only 1 geocaging area can be created. However, if a second geocaging area is created that intersects with the current one, a "merge" of both geocaging areas will be performed generating a single area. For example:



Geocaging area - Intersection of geocaging areas

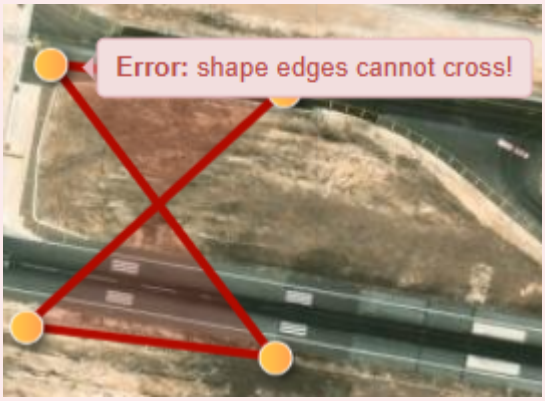
If users try to create a second geogacing area that does not "touch" the first one, i.e. there is no intersection between them, the first one will be deleted if the following warning confirmation message is accepted:



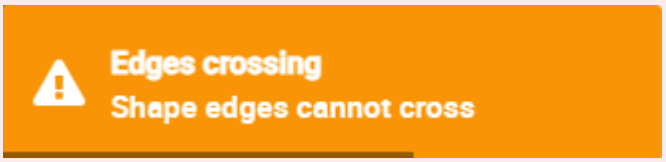
Geocaging area - Warning confirmation message

Error

If users try to generate a geocaging area that "crosses" the edges, Veronte Ops will not allow it, neither when creating it from scratch nor when modifying an existing area:



Geocaging area - Shape error



Geocaging area - Shape error message

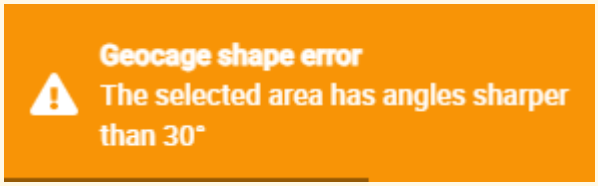
⚠ **Warning**

Geocaging warnings

- The generated area must be between 200 m² and 100 000 km². The following messages will appear if the area does not respect these limits, neither when creating it from scratch nor when modifying an existing area:

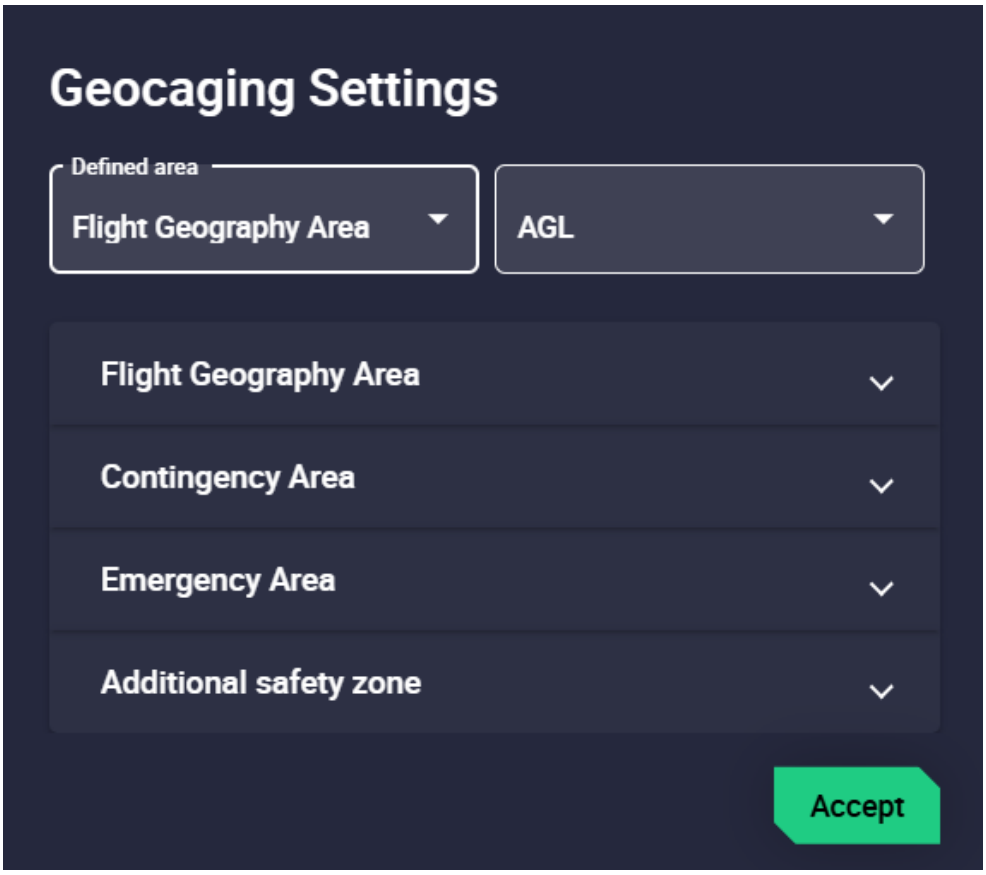
Geocaging area - Surface error

- The created area cannot have angles greater than 30º. The following message will appear when the area has angles that exceed this limit, either when it is created from scratch or when an existing area is modified:



Geocaging area - Edges sharper than 30º error

- **Geocaging Settings**:



Geocaging options - Settings

- **Defined area**: Users have to choose to which zone they want the drawn area to correspond, whether Flight Geography Area or Emergency Area, and the rest of the zones will be assigned automatically based on the configured margins.
- **Elevation**: Select between AGL, MSL or WGS84 reference system to define the altitude of the different zones within the geocaging area.
- **Flight Geography Area**:
  - Set height limits: Users must set the desired **upper limit** for this area. In addition, the lower limit can also be defined (optional).
  - Set Events: Users must associate this area to the specific event that must be previously defined in the **1x PDI Builder** software to trigger an alert or a corrective action. This option is explained in detail in the Shape section above.
- **Contingency Area / Emergency Area / Additional safety zone**:
  - Set Events: Users must associate this area to the specific event that must be previously defined in the **1x PDI Builder** software to

trigger an alert or a corrective action. This option is explained in detail in the Shape section above.

- Contingency / Emergency / Additional safety zone width: Depending on the Defined area selected when creating the geocaging area, this margin width will act differently in the calculation of the area size.
  - Flight Geography Area selected: The size of these zones is calculated by adding the margin entered here to the width and altitude of the previous area (3D zones).
  - Emergency Area selected: The calculation for the size will be:
    - For the Additional safety zone, adding the margin entered here to the width and altitude of the previous area (3D zone).
    - For the Contingency and Flight Geography Areas, subtracting the margin entered here to the width and altitude of the previous area (3D zones).

If the defined geocaging area complies with the standard, the "Geocaging not OK" bit (ID 504) will be **TRUE**, otherwise it will be FALSE.

> ⓘ **Note**
>
> This bit checks the following:
>
> - Regarding **area**, the geocaging area created must be between 200 m² and 100 000 km².
> - In terms of elevation:
> - Must be defined **absolute**, not relative.
> - All of the heights of the vertical limits shall be in the **same reference system** (MSL, AGL, WGS84).
> - **Upper limit** must **always** be set.
> - **Lower limit** must not be set in any area, except in the Flight Geography area where it is optional.

Example

The following actions are an example of user-configurable corrective actions that comply with the standard:

- An alarm bit is activated when entering the Contingency Area.
- A command is sent to switch to the **"Return to home" phase** when entering the Emergency Area.
- The **FTS** is activated when entering the Additional safety zone.

In the case where the corrective actions are as described above, the principle of operation will be as follows:

If the Flight Geography Area limit is trespassed, the platform enters in the Contingency Area and the defined "Contingency area" bit will turn **FALSE**, causing an alert to appear next to the platform icon (this alert must be configured in the settings of the platform icon, the user can find an explanation of this in the Platform icon - Veronte Ops configuration section of the present manual). Next, if the plaform exits this area and enters the Emergency Area, the current flight phase will switch to **Return to home**. Finally, if the platform also exits this area and enters the Additional safety zone, the **FTS** will be activated.

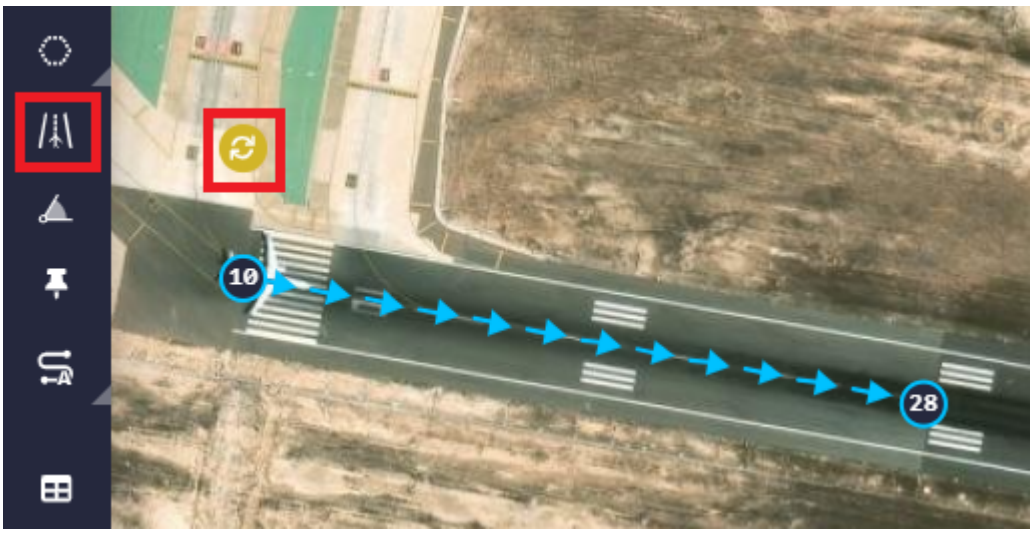
> **Geocaging Vs Geofencing**
>
> **Geocaging** defines an extensive 3D operational flight area by operating in both the horizontal and vertical axes and restricts movement outside of it. In this system, the behavior is that alerts and responses configured in the **1x PDI Builder** software are triggered, depending on the zone in which the platform is located.
>
> **Geofencing** restricts or prevents entry into specific areas by creating virtual boundaries around physical locations. In this system, a repulsion is implemented around the user-defined obstacle to achieve geofencing behavior.

Runway

This option allows the user to configure a **Runway** which is used during flight phases.

> ⓘ **Note**
>
> The linking of a runway to an automation is done in the **Operation Panel**, for more information about this, see Operation Panel - Operation section of this manual.

To add a runway, click on its icon, click on a point on the map to define the start and click on a second point to define the end of the runway.



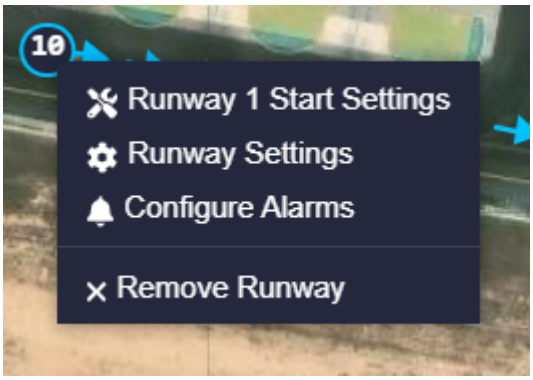
Runway

> ⓘ **Note**
>
> The loiter position is automatically defined and it is identified by this icon: , as can be seen in the image above.

By accessing the mission edition (icon) it is possible to modify the position of the start and end points of the runway, as well as the position of the loiter point:

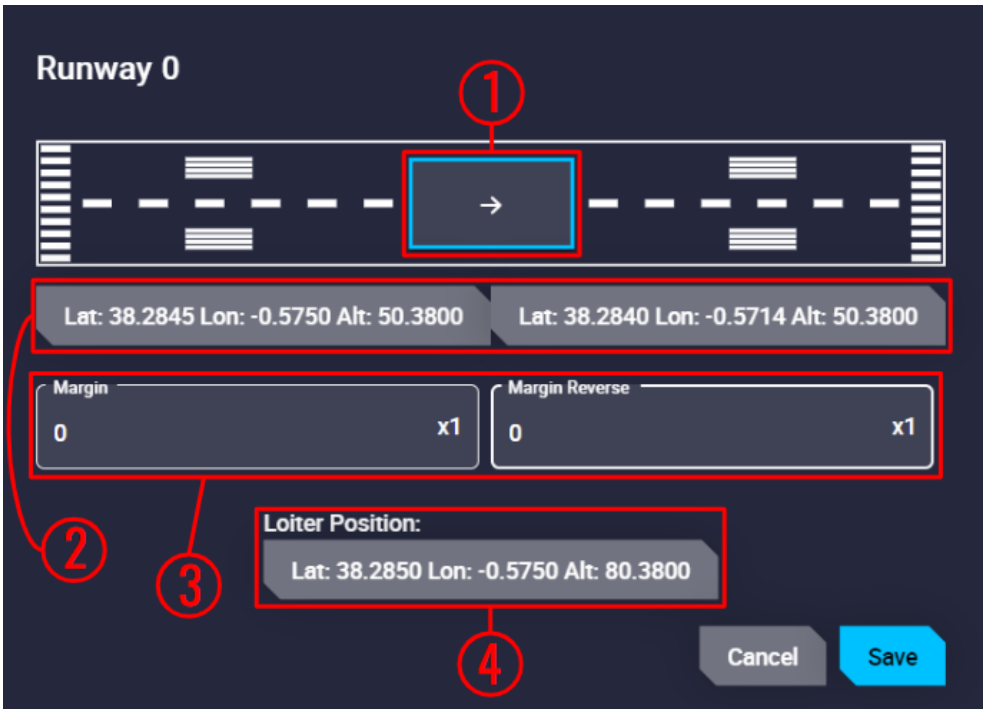
Runway position

Right-clicking on the start, end or loiter point of the runway, users will access to its **options**:



Runway options

- **Runway Start/End/Loiter Settings**: The coordinates of this runway point can be edited manually. Its configuration is the same as for a waypoint (a detailed explanation of this has been described in the Waypoint section above).
- **Runway Settings**: The following parameters can be configured:



Runway options - Settings

1. **Runway direction**: The direction of the runway is defined here with an arrow.

   By default it is → , but user can define it as desired clicking on it. The available options are → , ← and **AUTO**.

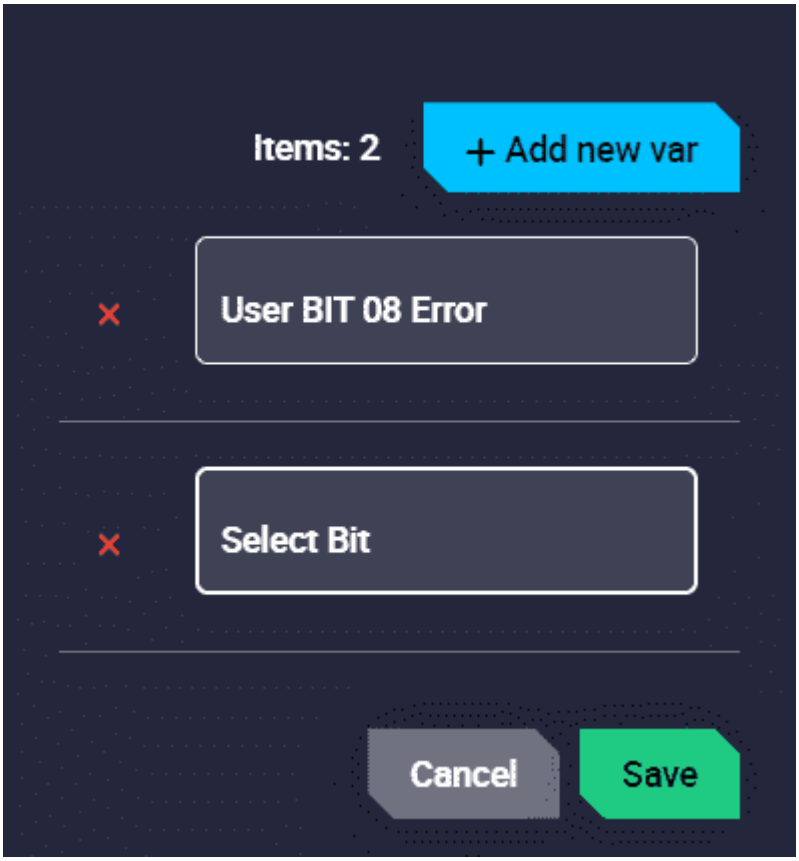
   > ⓘ **Note**
   >
   > When the **AUTO** option is selected, the runway direction will be defined by the **wind direction**.

2. **Runway coordinates**: The user can manually introduce the coordinates of the start and end point of the runway. Its configuration is the same as for a waypoint (a detailed explanation of this has been described in the Waypoint section above).
3. **Margin/Margin Reverse**: Percentage of the runway distance at which the airplane will try to touch the ground.
4. **Loiter Position**: To accurately define the loiter point of reference (**Runway Loiter**), its coordinates and the altitude that the aircraft will reach during climb, can be defined in this option. Its configuration is the same as for a waypoint (a detailed explanation of this has been described in the Waypoint section above).

- **Configure Alarms**: When an alarm is selected, the aircraft shall perform the actions associated to that alarm on the selected runway.

Runway options - Configure Alarms

Alarms are configured with bit variables, users can select any bit they want.



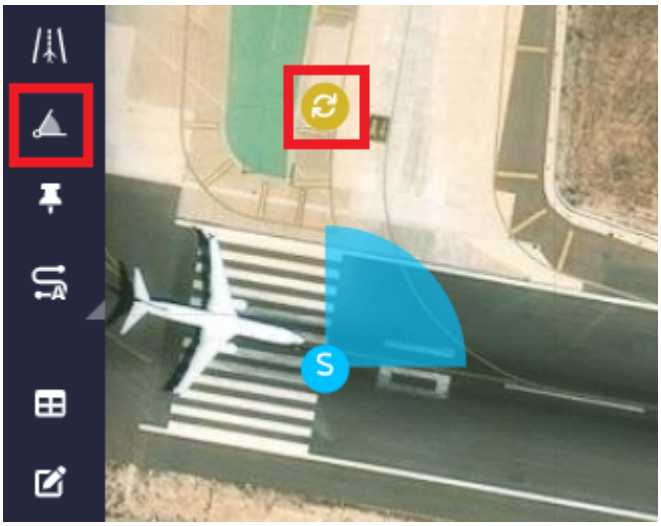
Runway options - Alarms

- **Remove Runway**: Deletes the runway.

Spot

This option allows the user to configure a **Spot** which is used during flight phases. A spot refers to a kind of runway where a initial point and its azimuth is defined. Besides, it is necessary to define a delta angle.

> **Note**
>
> The linking of a spot to an automation is done in the **Operation Panel**, for more information about this, see Operation Panel - Operation section of this manual.



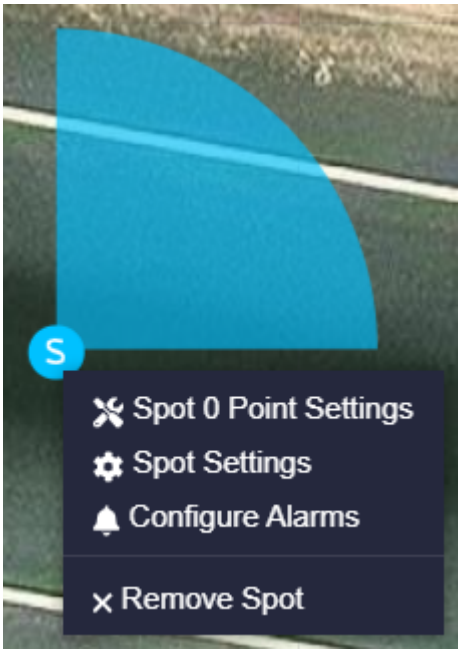
Spot

> **Note**
>
> The loiter position is automatically defined and it is identified by this icon: , as can be seen in the image above.

By accessing the mission edition (icon) it is possible to modify the **position**, **azimuth** and **delta** of the 'spot' and also the position of the loiter point:

Spot position

Right-clicking on the 'spot' or loiter point, users will access to its **options**:



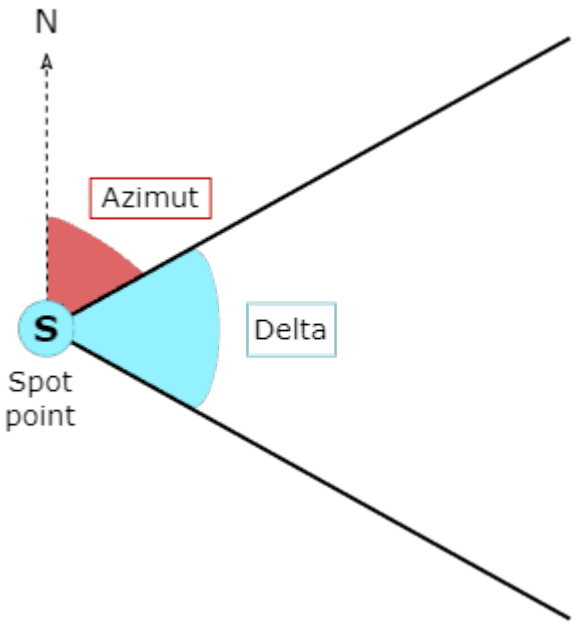
Spot options

- **Spot Point/Loiter Settings**: The coordinates of this point can be edited manually. Its configuration is the same as for a waypoint (a detailed explanation of this has been described in the Waypoint section above).
- **Spot Settings**: The aircraft will land or take-off using the best orientation computed within the area bounded by the parameters entered here:

Spot options - Settings

- **Position**: The user can manually introduce the coordinates of the 'spot' point. Its configuration is the same as for a waypoint (a detailed explanation of this has been described in the Waypoint section above).
- **Azimuth and Delta**: Users must configure the desired **azimuth** and **delta** for the spot. The units available for these angular values are $^{\circ}[0,\ 360]$, **º ' "** , rad, $rad[0,\ 2\pi]$, $^{\circ}$, $rad[-\pi,\ \pi]$ and $^{\circ}[-180,\ 180]$.

> ⓘ **Note**
> Azimut parameter can also be defined as relative to another point instead of absolute. To do this, click on the icon.



Spot parameters

  - **Loiter Position**: To accurately define the loiter point of reference (**Spot Loiter**), its coordinates and the altitude can be defined in this option.

    Its configuration is the same as for a waypoint (a detailed explanation of this has been described in the Waypoint section above).
- **Configure Alarms**: When an alarm is selected, the aircraft shall perform the actions associated to that alarm on the selected spot. It is configured in the same way as runways, described above.
- **Remove Spot**: Deletes the spot.

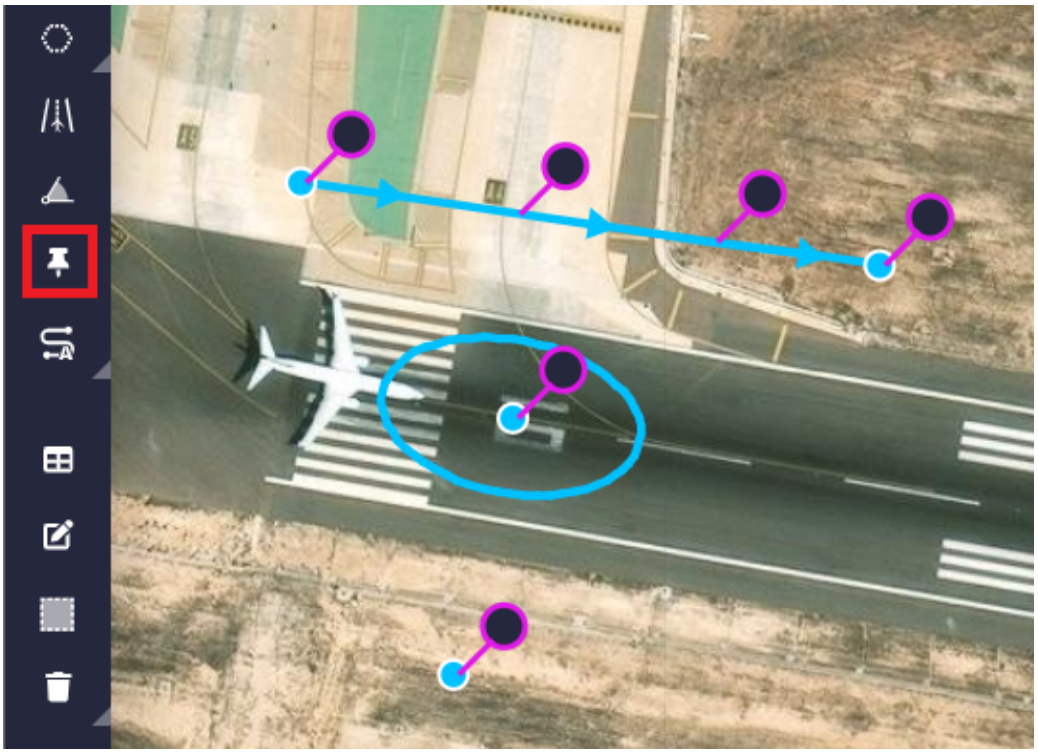
Marks

This tool allows to set an event mark on a patch. Marks are useful to initiating automations. When the aircraft flies over it, an event is triggered and 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, etc.

> ⓘ **Note**
>
> The linking of a mark to an automation is done in the **Operation Panel**, for more information about this, see Operation Panel - Operation section of this manual.

To place a mark, it is **first required to create a patch**. The elements described above that are patches are **waypoints**, **segments** and **orbits**. To add it, select the tool and click on the desired patch. Then, when users have created all the desired marks, **right-click on the map** to close/finish the 'creation event'.

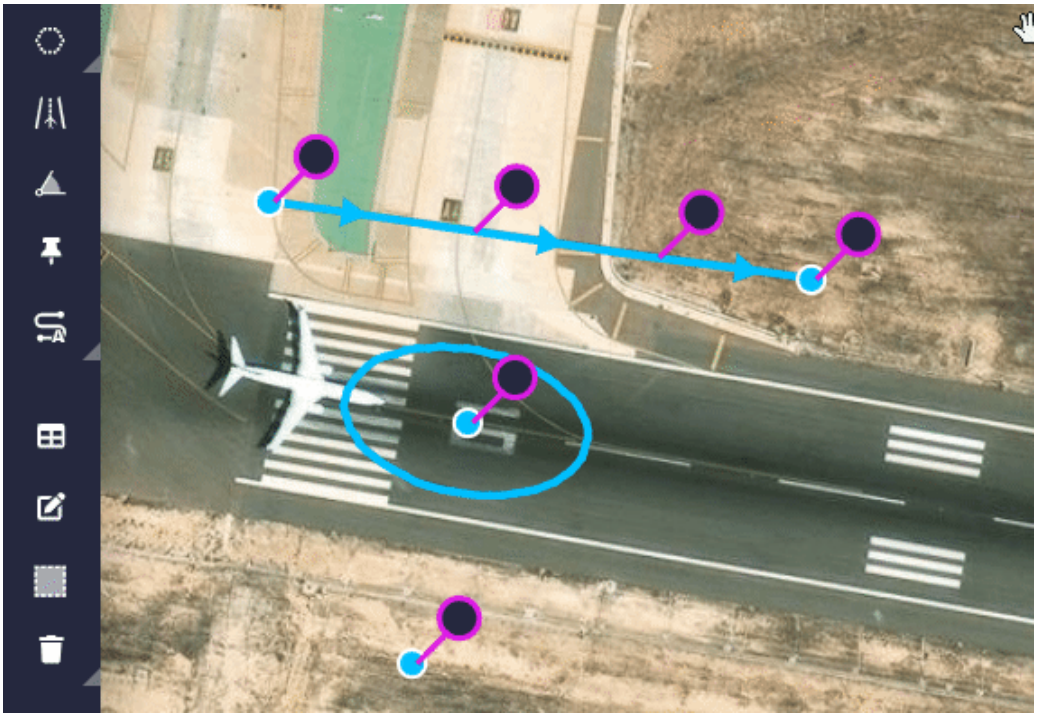
Initially, new marks appear as a pin ( ). If the user links an event to one of them, the icon selected for this event will be displayed.



Mark

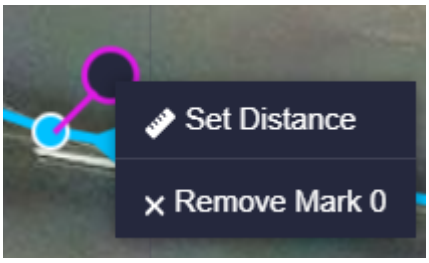
Marks can be moved along the patch by clicking on the icon:

> **Note**
>
> A mark can only be displaced if it is in a segment.



Marks moved

Right-clicking on the mark, users will access to its **options**:



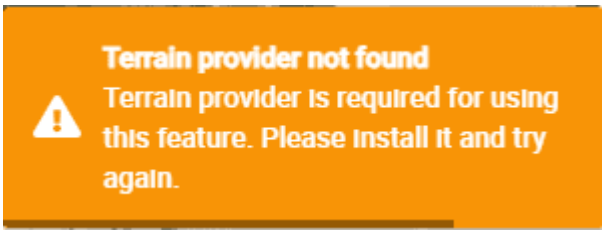
Mark options

- **Set Distance**: This distance is the horizontal distance along the patch to the starting waypoint. If the mark is attached to a waypoint, distance will be zero.
- **Remove Mark**: Deletes the mark.

Generate route

To be able to create these elements, it is first necessary to have the application Veronte Terrain Provider running.

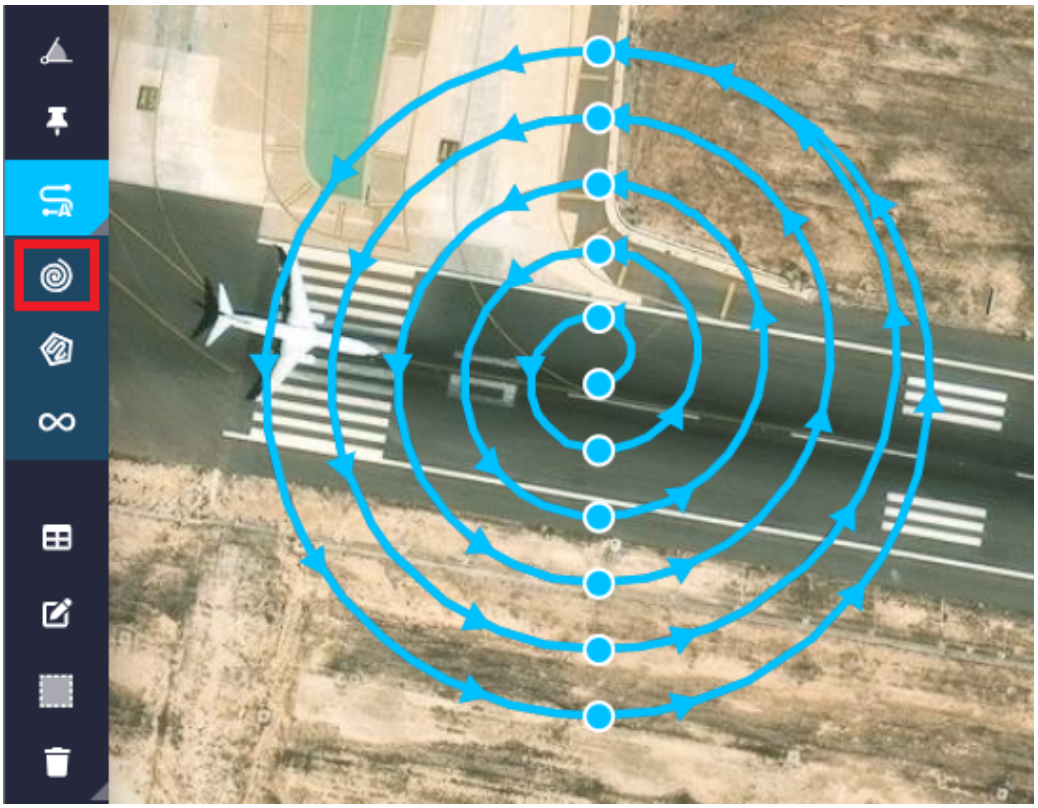
If this tool is not initialized, the following warning message will appear when trying to use any 'Generate Route' element:



Generate route warning message

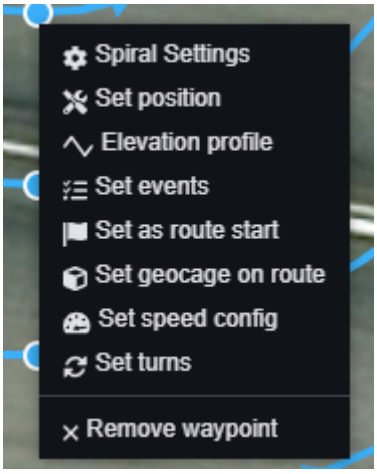
Spiral

Spiral tool draws a spiral that can cover a target area.



Generate route - Spiral

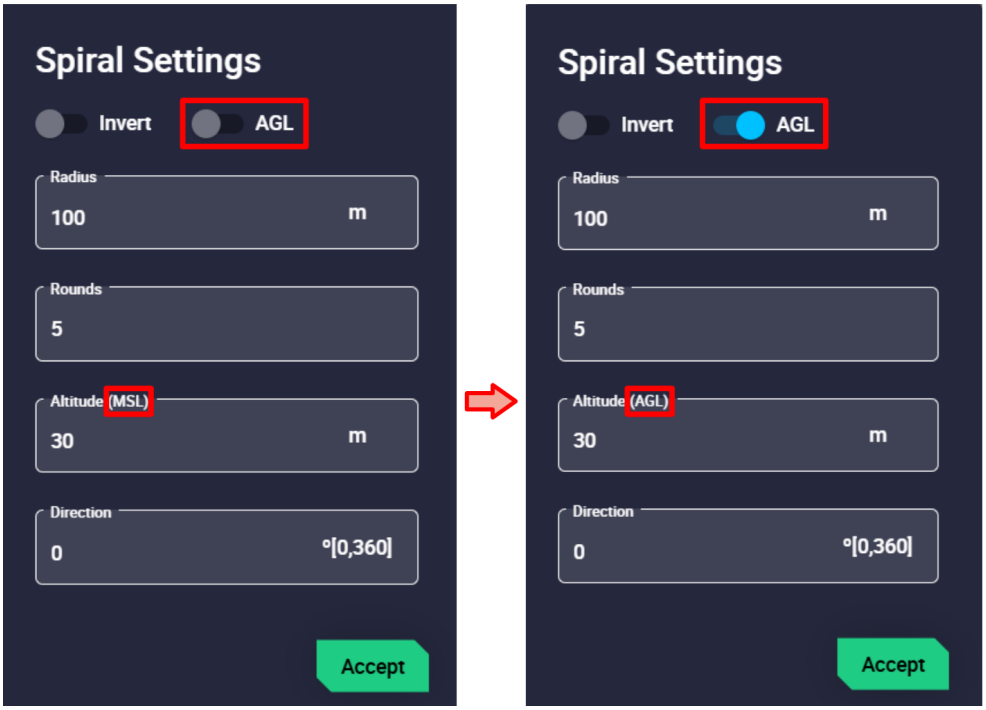
The spiral options are accessed in the same way as for the other elements described above, and are the same as the Waypoint options but with the addition of 3 more:



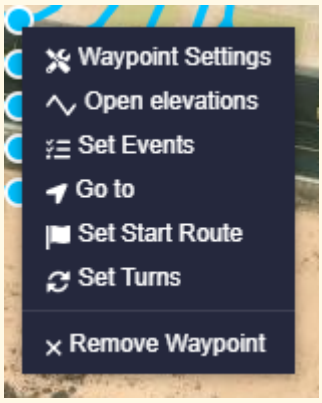
Generate route - Spiral options

- **Spiral Settings**: The following parameters can be configured:
  - **Invert**: When enabled, the direction of the route will be the opposite to the direction configured below.
  - **AGL**: If enabled, the waypoint altitude is set in AGL, otherwise it is expressed in MSL.

- **Radius**: Spiral maximum radius.
- **Rounds**: The number of spiral rounds.
- **Altitude (MSL/AGL)**: Altitude of the waypoints expressed in **MSL/AGL** (depending on whether the AGL parameter is enabled or disabled).
- **Direction**: Direction of the spiral axes on the map.



Generate route - Spiral Settings

> ⚠ **Warning**
>
> If the user 'refreshes' **Veronte Ops**, it will no longer be possible to access this configuration menu.
>
> 
>
>
> **Generate route - Spiral options after refresh**

- **Set geocage on route**: Allows configuring a geocaging zone around the segment. For more information, consult Geocaging area.

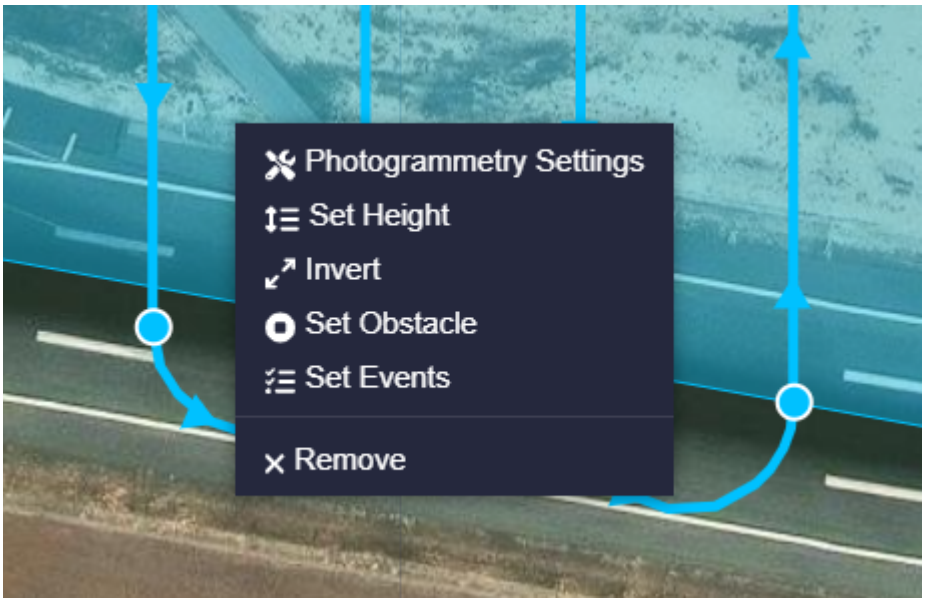
- **Set Turns**: The spiral also acts as a curved segment so it has the same option as this; a detailed explanation has been described in the curved segment section above.

Photogrammetry Mission

Photogrammetry Mission tool allows users to draw a prism on the map to **automatically generate a mapping mission**.

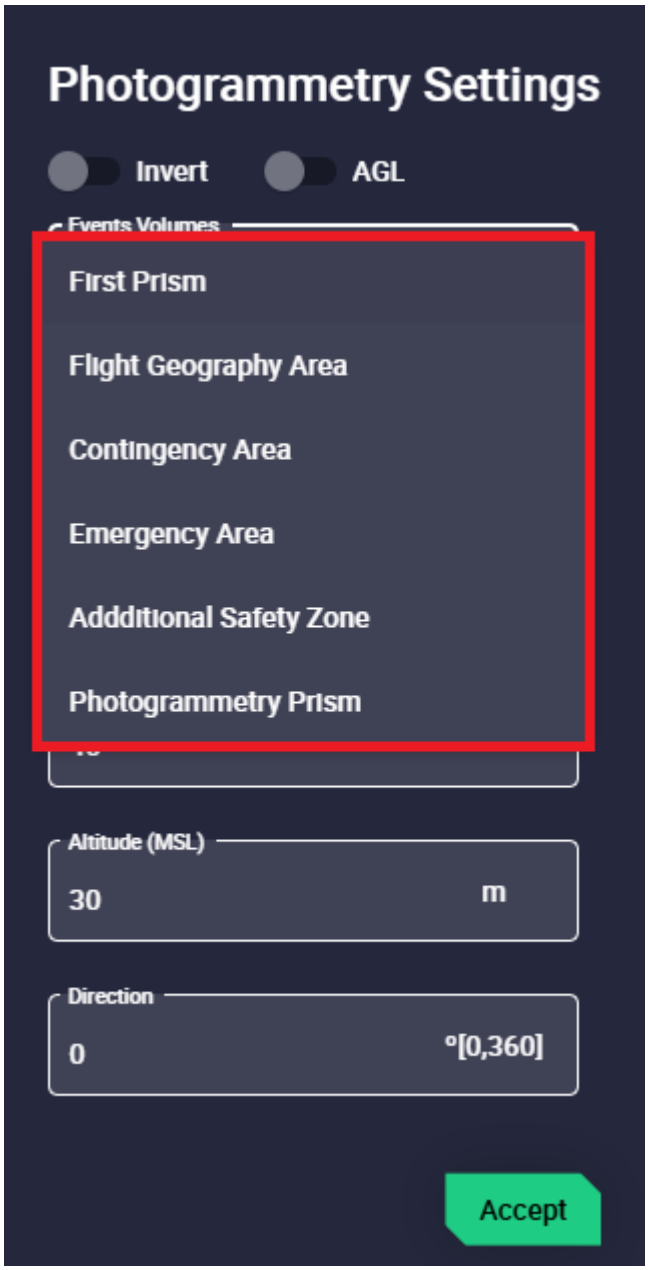
Generate route - Photogrammetry Mission

The photogrammetry mission options are accessed in the same way as for a prism.



Generate route - Photogrammetry Mission options

- **Photogrammetry Settings**: The following parameters can be configured:
  - **Invert**: When enabled, the direction of the route will be the opposite to the direction configured below.
  - **AGL**: If enabled, the waypoint altitude is set in AGL, otherwise it is expressed in MSL.
  - **Events Volumes**: As with all volumes, users can link a predefined Event Volume to this photogrammetry mission. This configuration option is the same as the **Set Events** option.

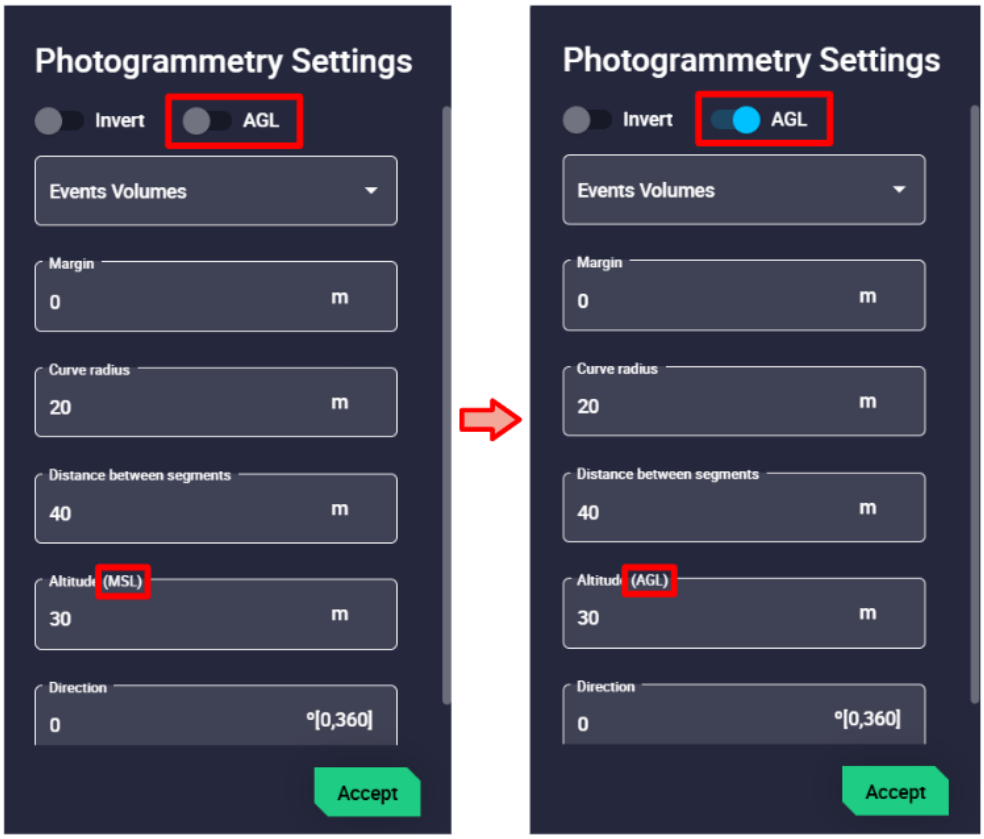


Generate route - Photogrammetry Settings Events Volumes

- **Margin**: This option allows the user to add boundary margins for the route to each side of the photogrammetry mission.
- **Curve radius**: Radius of the route turns.
- **Distance between segments**: Distance between segments can be modified.

> **Note**
>
> Depending on the **curve radius** value and the **distance between segments**, there are three cases for this part of the route (the radius must be set according to the minimum achievable by the platform):
>
> - Radius **0** $\Rightarrow$ There are no 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 segments ($R < \frac{d}{2}$) $\Rightarrow$ **Veronte Ops** generates a semicircle with the diameter equal to the distance between parallel segments.
> - Radius bigger than half the distance between segments ($R > \frac{d}{2}$) $\Rightarrow$ The path between straight lines is formed by two curves and a straight line.

- **Altitude (MSL/AGL)**: Altitude of the waypoints expressed in **MSL/AGL** (depending on whether the AGL parameter is enabled or disabled).
- **Direction**: Direction of the route generated on the map.

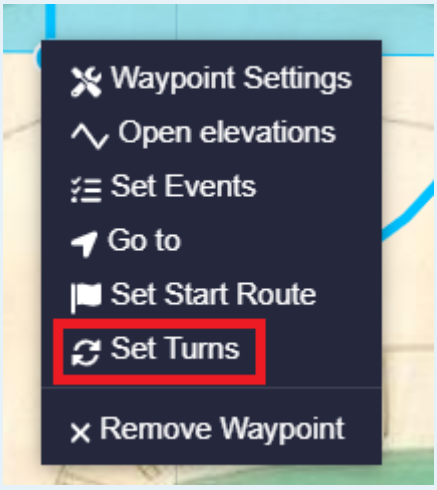


Generate route - Photogrammetry Settings

- **Set Height**: This option is configured in the same was as for a prism, refer to Prism section for a detailed explanation.
- **Invert**: This option is configured in the same was as for a prism, refer to Prism section for a detailed explanation.
- **Set Obstacle**: This option is configured in the same was as for a prism, refer to Prism section for a detailed explanation.
- **Set Events**: Explained previously in the Shape section.

> ⚠ **Warning**
>
> If the user 'refreshes' **Veronte Ops**, the photogrammetry mission will become a mapping route with a "normal" prism, therefore its options will be the same as those of a prism. Consequently, the “Photogrammetry Settings” option will disappear, and it will no longer be possible to access the "Photogrammetry Settings" configuration menu.

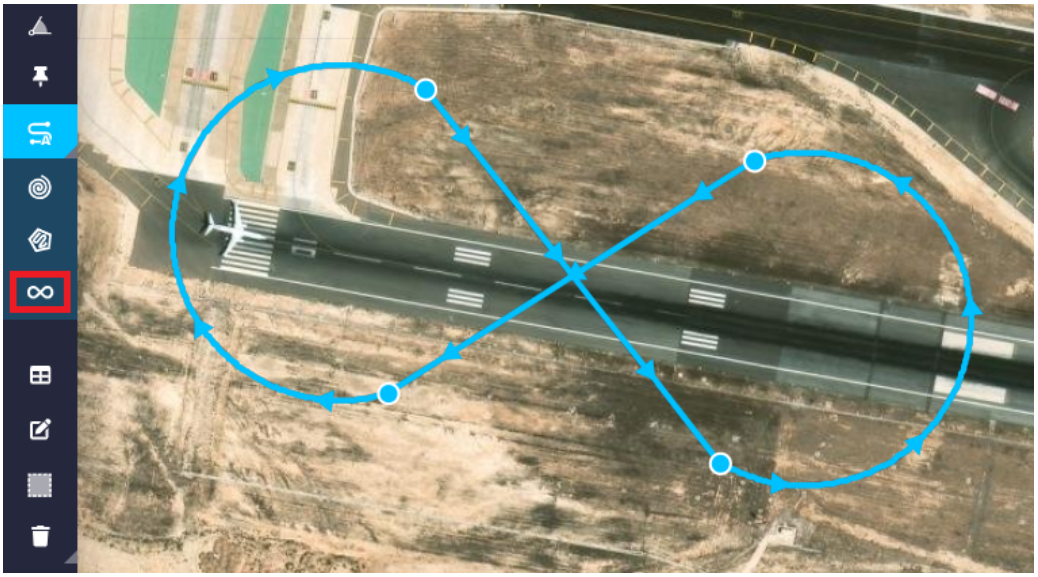
> ⓘ **Note**
>
> The options for the waypoints forming the route generated for this photogrammetry mission are the same as for a "normal" waypoint. For more information on waypoint configuration, see Waypoint section described above.



Generate route - Waypoint route options

Figure

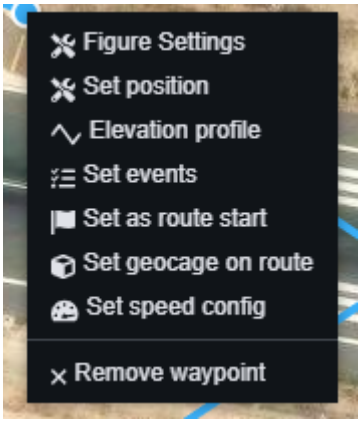
Figure. This tool allows users to create their own mission from a pre-designed route. To create it, click and drag to make the figure as big as desired.



Generate route - Figure

The figure options are accessed in the same way as for the other elements described above, and are the same as the Waypoint or curved segment options, depending on the waypoint in the route. This is because it is basically made up of straight and curved segments.

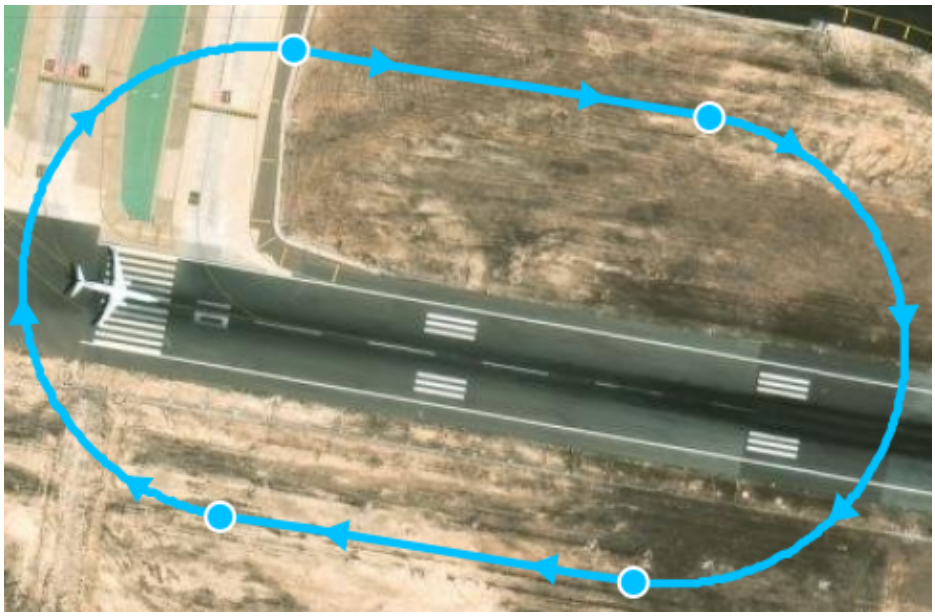
However, an extra option appears to access the '**Figure Settings**':



Generate route - Figure options

The following parameters can be configured:

- **Invert**: When enabled, the direction of the route will be the opposite to the direction configured below.
- **AGL**: If enabled, the waypoint altitude is set in AGL, otherwise it is expressed in MSL.
- **Type**: The available options are:
  - **Figure of 8**: This is a path in the shape of 8/infinity.
  - **Racetrack**: This route is like a racetrack.



Generate route - Figure racetrack

- **Altitude (MSL/AGL)**: Altitude of the waypoints expressed in **MSL/AGL** (depending on whether the AGL parameter is enabled or disabled).

Generate route - Figure Settings

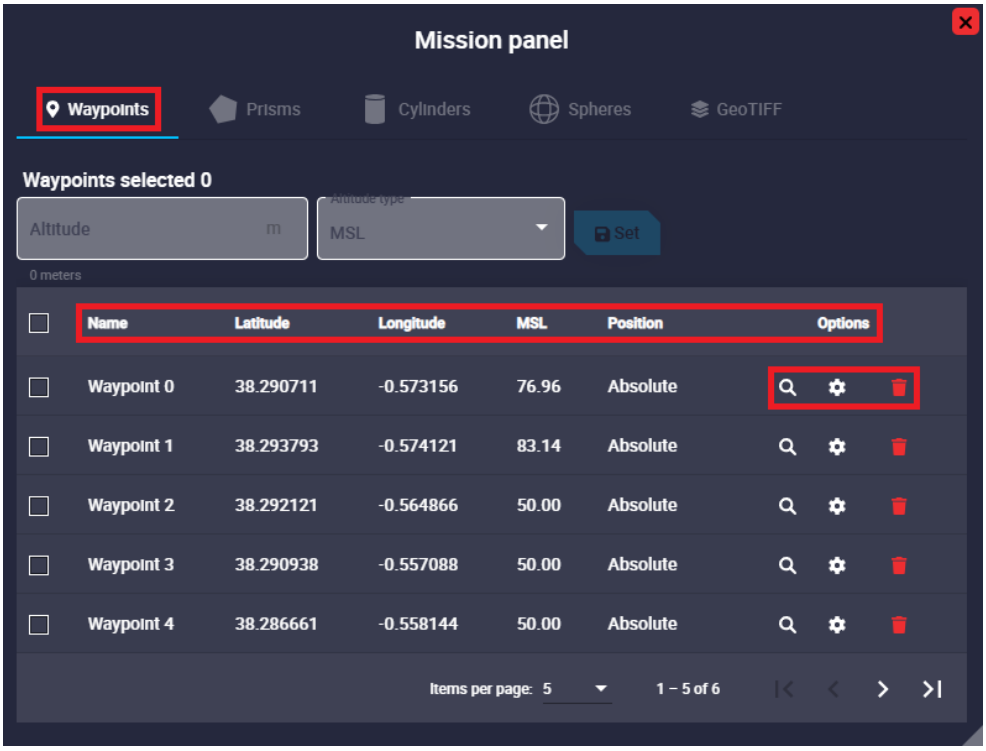
> ⚠ **Warning**
>
> If the user 'refreshes' **Veronte Ops**, it will no longer be possible to access this configuration menu. Therefore, the figure will act as a "normal segment".

Mission panel

This mission panel allows the user to monitor and manipulate the **waypoints**, **prisms**, **cylinders** and **spheres** created.

- **Waypoints**

  The following fields can be found in this menu:



Mission panel - Waypoints

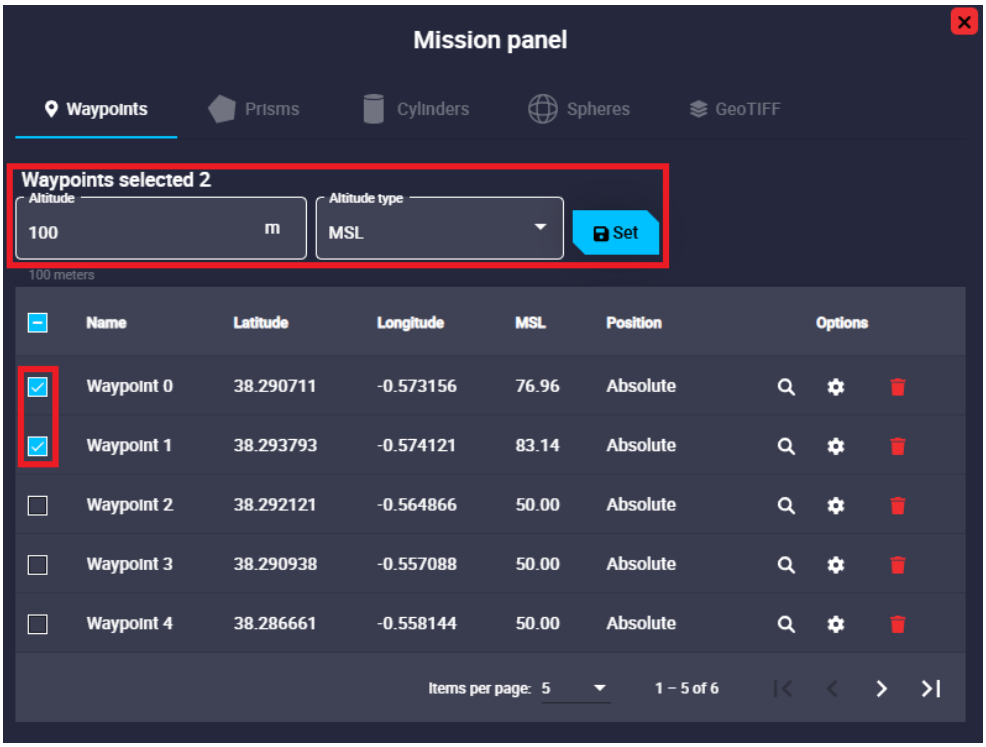
  - **Name**: Name of this waypoint.
  - **Latitude/Longitude/MSL**: Coordinates of the waypoint.
  - **Position**: Absolute or Relative.
    - **Absolute** means that the position of the element is fixed.
    - **Relative** indicates that its position is relative to the position of another element and may change during the operation.

- **Options**:
  - **Search**: Clicking here will center and zoom the waypoint on the map.
  - **Edit**: Opens the Waypoint Settings menu. For more information on this configuration menu, click here.
  - **Delete**: Deletes this waypoint.

It is also possible to **modify the altitude** of one, several or all the selected waypoints by using the **Altitude edit** function **on the top of the list**.

> **Note**
> Selecting all will select the waypoints on **all pages** of the panel.

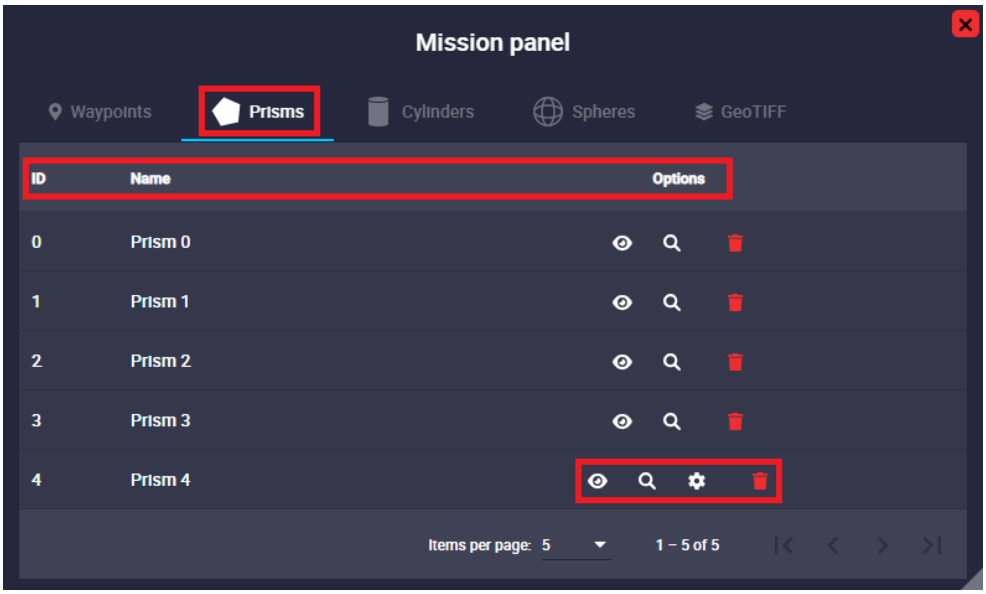
Users must enter the desired **altitude value**, select the **altitude type** (WGS84, MSL or AGL) and finally click '**Set**'.



Middion panel - Waypoints altitude

- **Prisms**

  The following fields can be found in this menu:



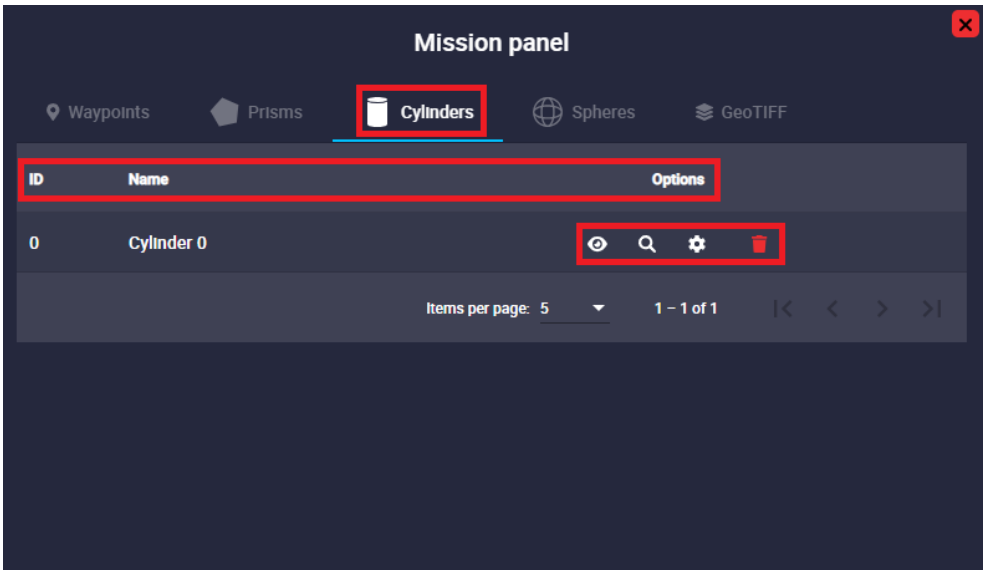
Mission panel - Prisms

- **ID**: Feature Id of this prism.
- **Name**: Name of this prism.
- **Options**:
  - **Show/hide**: It is possible to show/hide the prism on the map by clicking here.
  - **Search**: Clicking here will center and zoom the prism on the map.
  - **Edit**: Opens the prism **height** editing menu. For more information on these settings, click here.
  - **Delete**: Deletes this prism.

> **Important**
>
> When the prism is part of a **geocaging area**, the Edit option will not appear is this panel. This is why, in the image above Prisms 0 to 3 do not have this option available.

- **Cylinders**

  The following fields can be found in this menu:

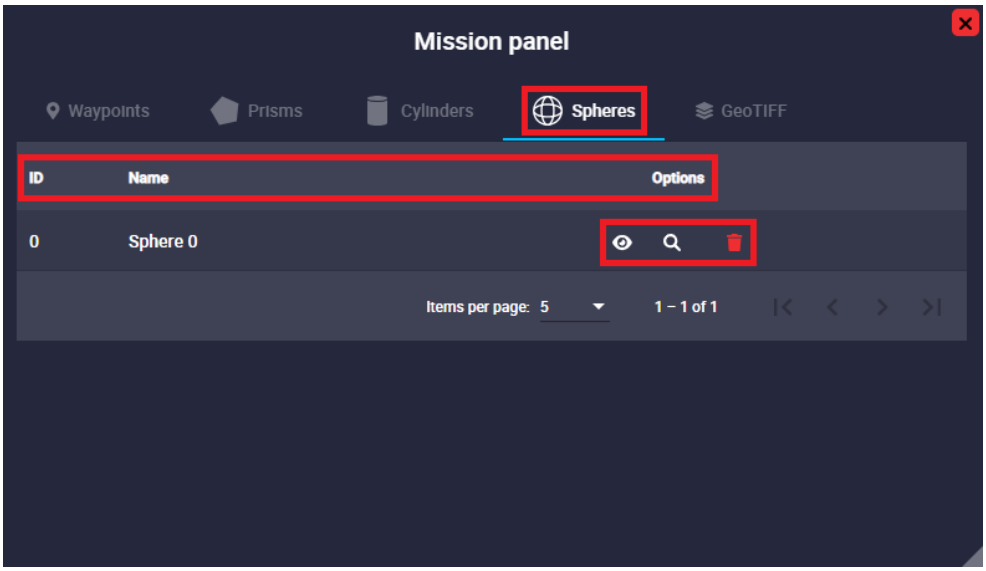


Mission panel - Cylinders

- **ID**: Feature Id of this cylinder.
- **Name**: Name of this cylinder.
- **Options**:
  - **Show/hide**: It is possible to show/hide the cylinder on the map by clicking here.
  - **Search**: Clicking here will center and zoom the cylinder on the map.
  - **Edit**: Open the cylinder **height** editing menu. For more information on these settings, click here.
  - **Delete**: Deletes this cylinder.

- **Spheres**

  The following fields can be found in this menu:



Mission panel - Spheres

- **ID**: Feature Id of this sphere.
- **Name**: Name of this sphere.
- **Options**:
  - **Show/hide**: It is possible to show/hide the sphere on the map by clicking here.
  - **Search**: Clicking here will center and zoom the sphere on the map.
  - **Delete**: Deletes this sphere.

- **GeoTIFF**

  From this panel users can insert **GeoTIFF** images into the map.

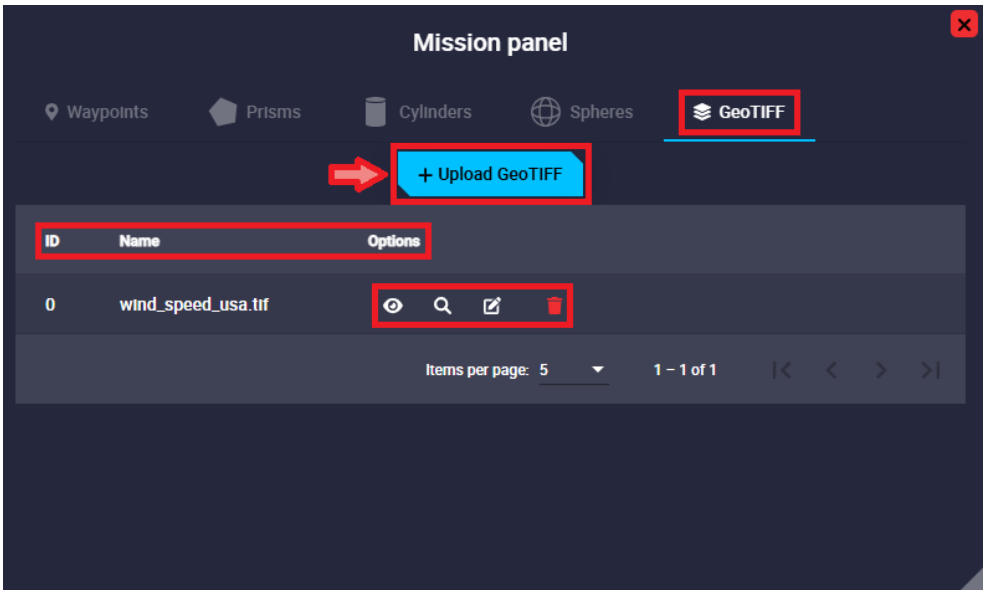
  A **GeoTIFF** image is a standard TIFF image file that contains geographic information embedded directly in the file itself. This "georeferencing" allows the software to place the image in its correct position on the Earth's surface. This type of image differs from a normal image because, in addition to storing pixel data (colors), it also includes additional metadata that describes its spatial context.

  In summary, these images are characterized by integrated georeferencing, detailed metadata, and a raster format.

To insert a **GeoTIFF** image, click the **Upload GeoTIFF** button and select the desired raster image from the user's local storage.

The allowed extensions for these images are `.tif`, `.tiff`, and `.geotiff`.

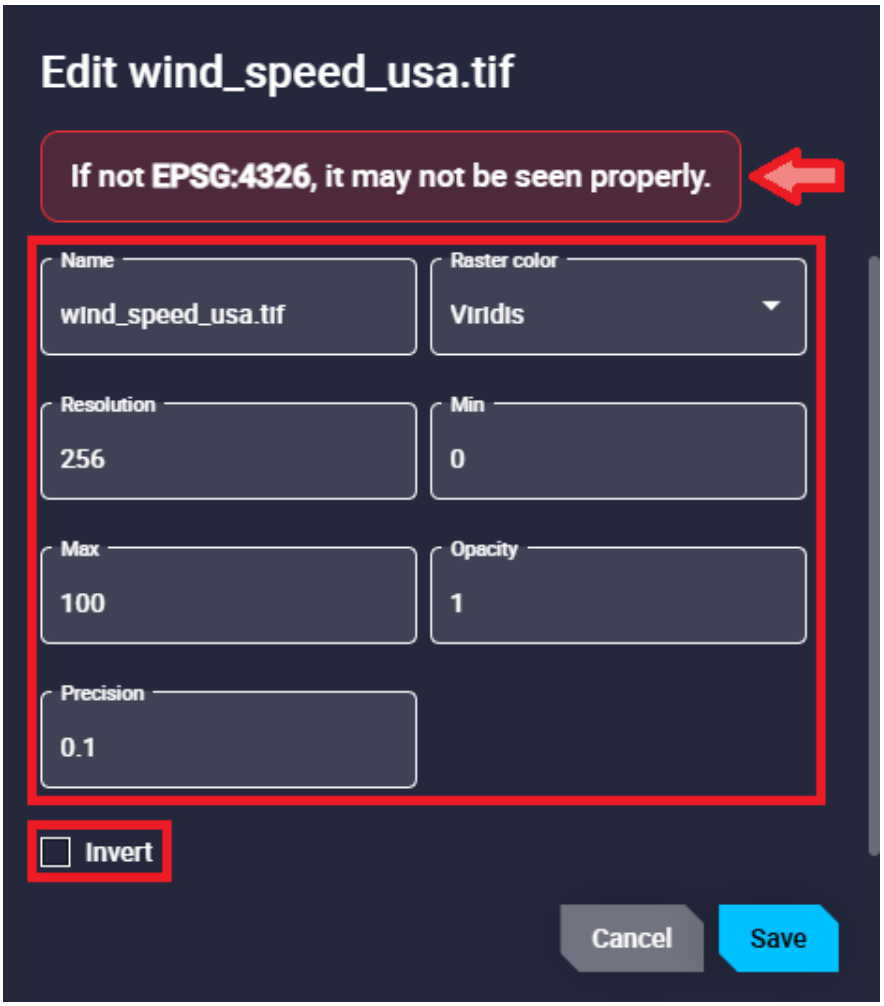
> ⚠ **Caution**
>
> If the GeoTIFF image contains extensive information (high file size), this can cause the Veronte Ops application to crash.



Mission panel - GeoTIFF

The following fields can be found in this menu:

- **ID**: Feature Id of this geoTIFF.
- **Name**: Name of this geoTIFF.
- **Options**:
  - **Show/hide**: It is possible to show/hide the geoTIFF on the map by clicking here.
  - **Search**: Clicking here will center and zoom the geoTIFF on the map.
  - **Edit**: Open the geoTIFF editing menu. The following parameters can be configured to control how the data contained in the image is displayed:



Mission panel - GeoTIFF edit menu

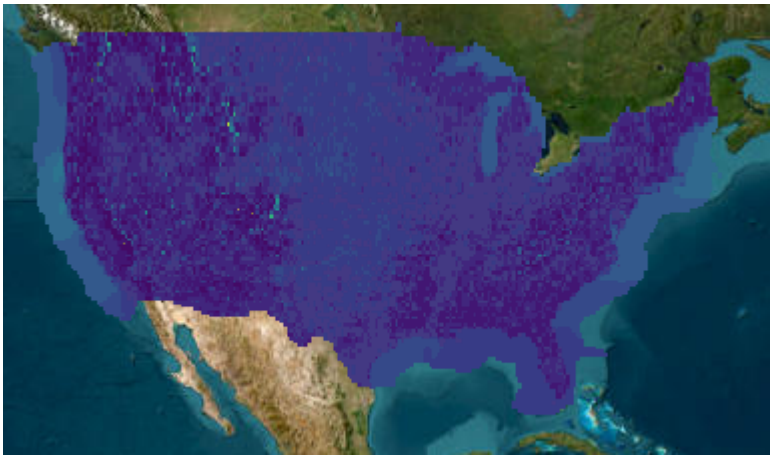
- **Name**: Users can rename the image as desired. By default, the file name is set.
- **Raster color**: Color palette used to represent pixel values. Users can choose between None, Rainbow, Viridis or Grayscale.
- **Resolution**: Number of colors or discrete steps in the color palette. By default 256.
- **Min / Max**: Range of data represented by the color palette. By default 0/100.
- **Opacity**: This controls the transparency of the image layer. The range of values is from 0 to 1, where 1 is completely opaque and 0 is completely transparent. Default 1.
- **Precision**: Specifies the amount of decimal numbers to be used to display or interpret data values.

In addition, the colors of the raster image can be inverted by enabling the Invert checkbox.

> ⚠ **Warning**
> If the image is not in the map projection standard, **EPSG: 4326**, it may not be seen properly.

- **Delete**: Deletes this geoTIFF.

For example:



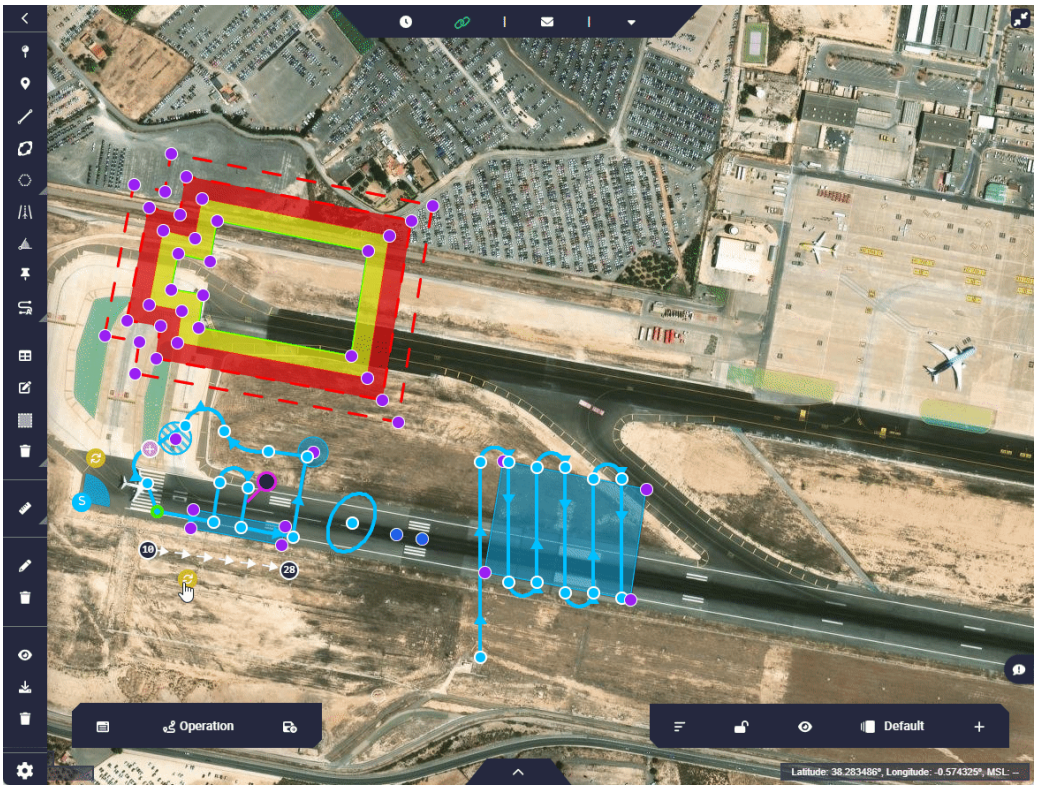
Mission panel - GeoTIFF example

Edit mission

This tool allows the user to move waypoints, marks, areas, etc., as well as to change their shape.

A detailed explanation on how to use this tool can be found in each of the elements described above.

In addition, **when modifying a mission with a Veronte Autopilot 1x connected**, the "original operation" loaded in the autopilot will be greyed out and the modified one will be colored. This will be the case until the new changes are saved and uploaded to the autopilot, so the previous operation will be completely removed from the map. An example is shown below:



Edit mission

Turn on multiselection

Multiselection allows to **move and rotate mission's path** from one place on the map to another.

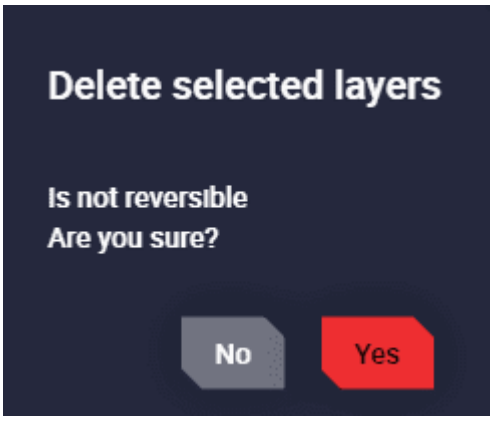
- Select the tool and create a rectangle by clicking and dragging until the elements you want to move fall into it.
- With the elements selected, click and drag the rectangle to move it to the desired location. And to rotate it, click on the 'black pin' ( ) and drag it to the desired rotation angle.
- After that, click again in the tool's icon to deselect.

> ⓘ **Note**
>
> Runways and spots cannot be moved with this tool.

Multiselection tool

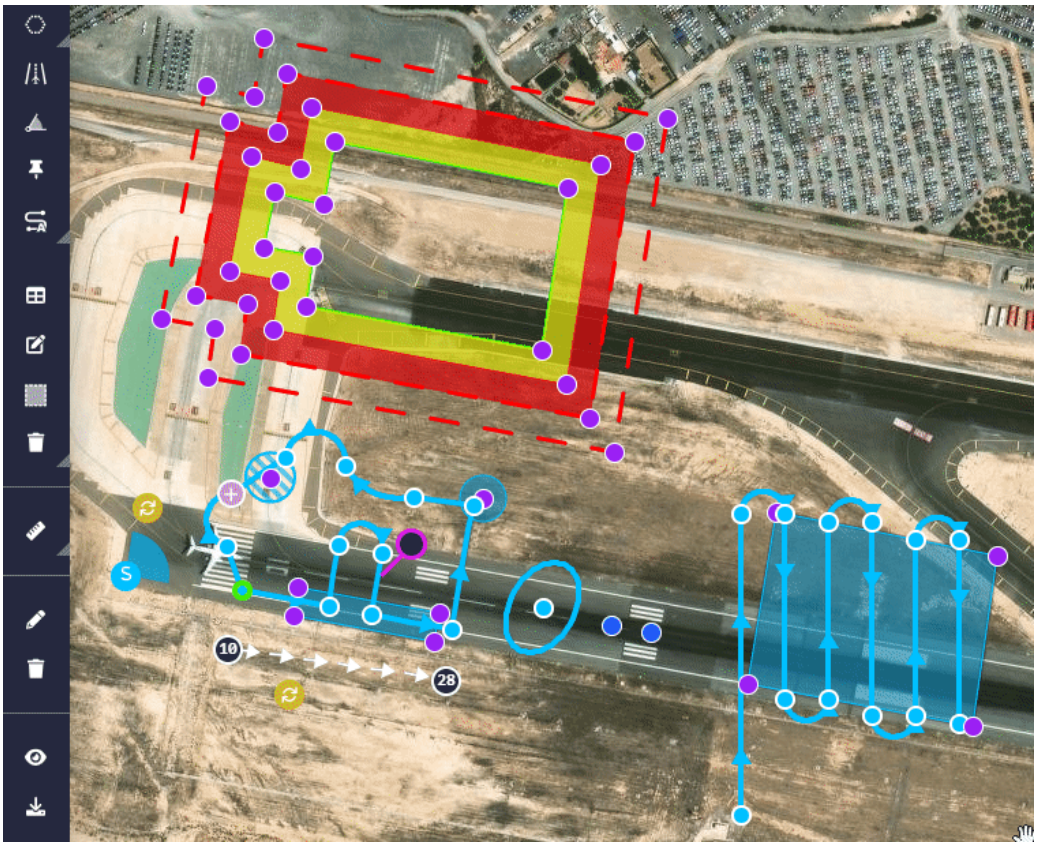
In addition, **this tool also allows to delete all items selected** in two different ways, **right-clicking** on the selected area or **pressing the 'delete' key** on the user's keyboard.

Before deleting the selected items, the following confirmation message appears:



Multiselection tool - Remove message

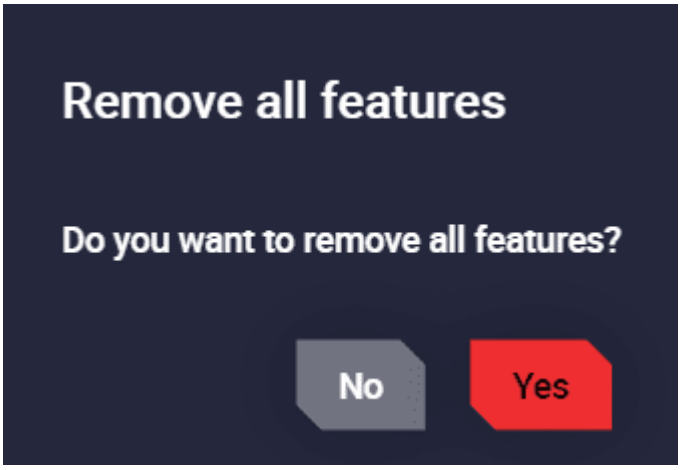
> ⓘ **Note**
>
> Runways and spots cannot be deleted in this way.



Multiselection tool - Remove

Remove mission

By clicking on this icon , the user can remove all the elements created by simply clicking on each of them. However, if the user wishes to remove all the missions created, simply click on the second icon that appears: . A confirmation pop-up window will then appear to make sure that all missions wish to be deleted.



Remove all missions

Import features

Select this option to load mission elements from a `.KML` or `.GeoJSON` file located in the user local storage. The system will interpret the file configuration to match **Veronte Ops** elements (e.g., mapping **KML** shapes to Waypoints, Segments, or Obstacles).

> ⓘ **Note**
>
> When importing polygonal shapes (such as Prisms or Cylinders):
>
> - The **Upper Limit** of the volume is automatically set to the maximum height of the vertices found in the file.
> - The **Lower Limit** is restored from the file's extra properties if available.

> ⚠ **Warning**
>
> During Import, Veronte Ops analyzes the data to reconstruct the original geometries. If specific **parameters are missing**, the element will be imported as a **standard geometry** or **ignored** if incompatible.

Export features

Select this option to save the current mission elements to a `.KML` or `.GeoJSON` file.

- **Supported Elements**: Waypoints, Segments, Orbits, Prisms, Cylinders, Spheres, Spirals, Photogrammetry Mission, and Figure.
- **Excluded Elements**: Mission Points, Marks, Runways, Spots, and Geocages cannot be exported.

> **Note**
>
> When exporting polygonal shapes, the system applies specific logic to handle altitude:
>
> - The height of the exported vertices corresponds to the configured **Upper Limit** of the volume.
> - If no Upper Limit is defined, the vertices are exported at ground level.
> - The **Lower Limit** is saved within the file's extra properties to ensure data integrity upon re-importing.

Export features

> **Warning**
>
> All application elements can be exported in `.KML` format, with the exception of:
>
> - Mission Points
> - Marks
> - Runways
> - Spots
> - Geocages

Ruler

Turn on ruler

Using the ruler, the user can measure distances and azimuths.

Ruler

The following information can be found in the each point of the figure above:

- In (º): **Azimuth** in **degrees** of the 'incoming' measure. The **azimuth** is the angle of the segment with respect to North.
- Out (º): **Azimuth** in **degrees** of the 'outcoming' measure. The **azimuth** is the angle of the segment with respect to North.
- Blue measurement: Distance from the previous point.
- Green measurement: Distance from the first point.

> ⓘ **Note**
>
> There 3 different types of points on the ruler segments:
>
> - Green point: Indicates the start of the route.
> - Blue point: Midway point of the route.
> - Red point: End of the route.

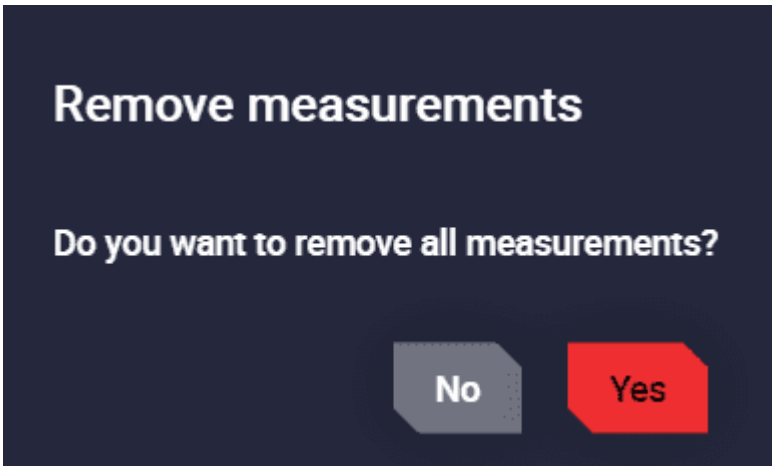
In addition, by clicking on the ruler icon, two more options will appear:

-

Change units: The user can change the units of the measurements by clicking on this icon. The available units are: *km* (kilometeres), *mi* (miles) and *nm* (nautical miles).

  > ⓘ **Note**
  > The ruler indicates the **measures with 2 decimals** in order to have more precision, as can be seen in the figure above.

-
Clear measurements: By clicking on it, it is possible to delete all the measurements in the mission. A confirmation pop-up window will then appear to make sure that all measurements wish to be deleted.

Remove all measurements

Paint

Enable paint

This tool allows the user to make quick notes during the operation.

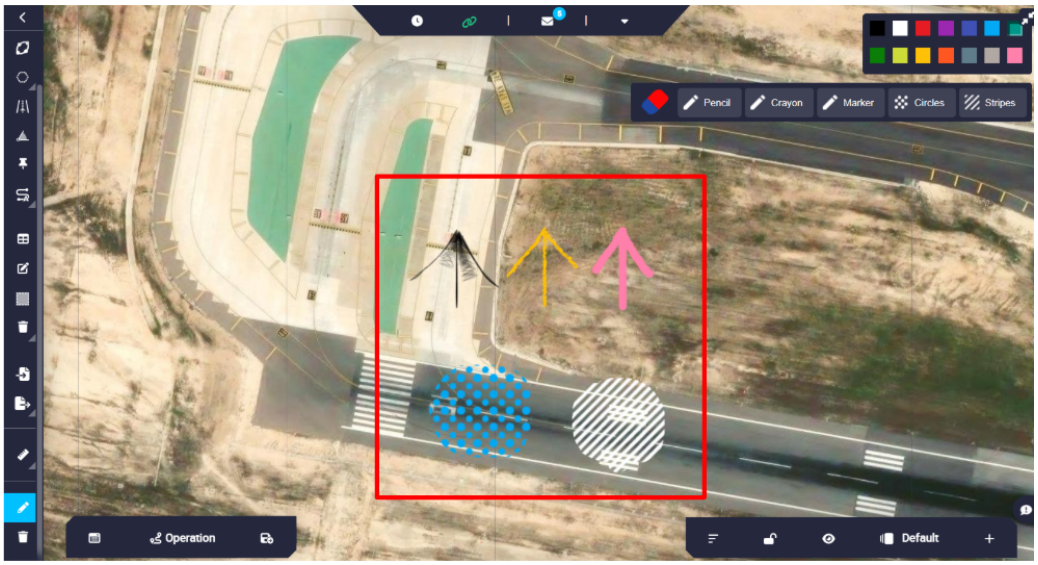
To access the paint option, click on this icon, a 'Paint toolbar' will then appear:

Paint toolbar

Users can choose from a variety of 'writing elements' and colors to paint:

- Pencil
- Crayon
- Marker
- Circles
- Striples

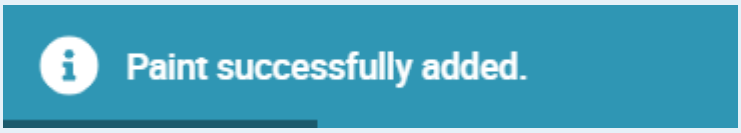
An example of each is shown in the figure below:



Paint examples

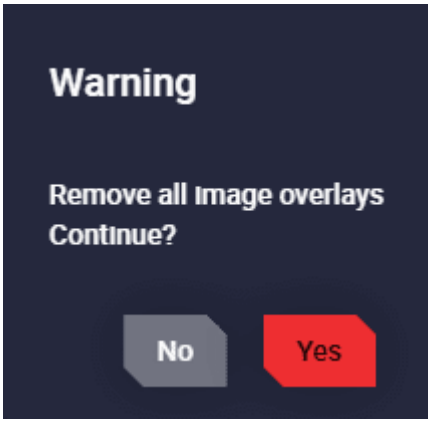
In addition, all notes made can be erased with an eraser, if they are not already saved, by clicking on the 'eraser option' in the toolbar.

Paint erased

> **Note**
>
> Changes will be saved automatically upon exiting the '*Paint' option. When exiting, the following message appear:
>
> 
>
>
> **Message paint added**

Delete paint

By clicking the following icon , **all paints will be removed**. The following confirmation pop-up window will appear:



Remove paint

After clicking 'Yes', the following confirmation message will be displayed

Message paint deleted

Offline

This option is designed for when users operate offline, i.e. they do not have internet access to download map information. For this reason, the map tiles of the area to work on can be pre-downloaded.

The following options are available in this section:

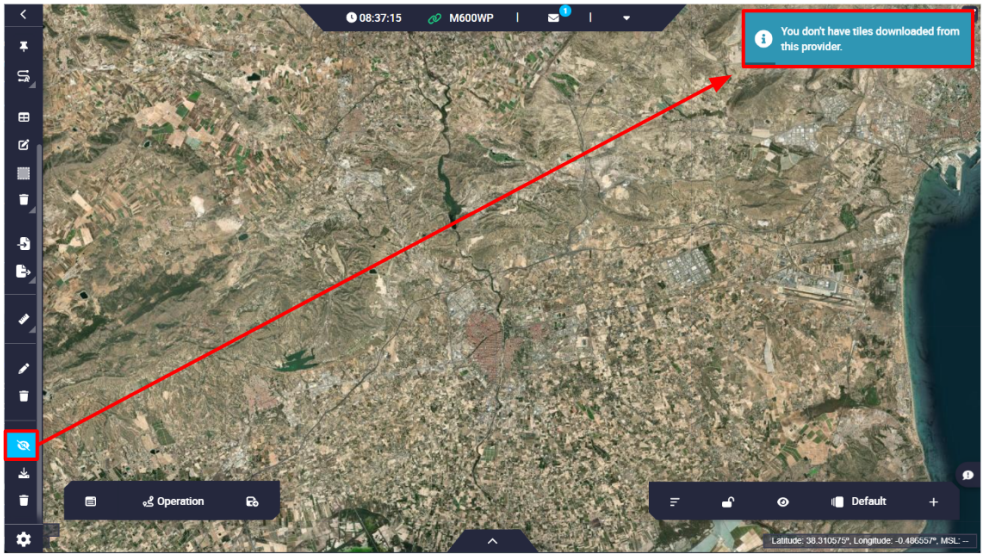
Show downloaded tiles

This option works differently when working online and offline:

- **Online**: If the user has already downloaded tiles, when clicking on this icon, the area with the downloaded tiles will be marked with a blue rectangle:

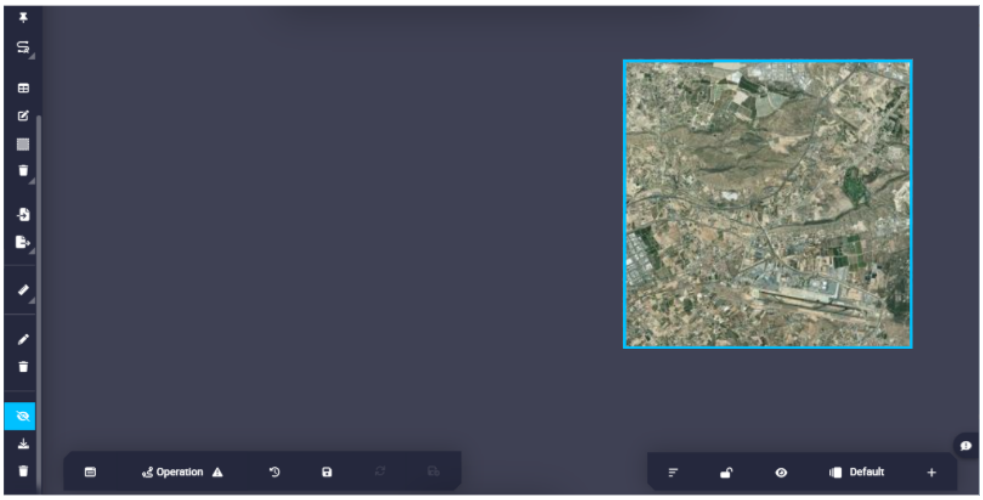
Show tiles

If the user has not downloaded any tiles, the following message will appear when clicking here:



Show tiles - don't have downloaded

- **Offline**: If the user has already downloaded tiles, when clicking on this icon, these tiles will always appear:



Show tiles

If the user has not downloaded any tiles, they may appear because they are stored in the **Veronte Ops** cache, however, it is not certain that this will always happen. Therefore, it is recommended to have the map tiles downloaded when working offline.

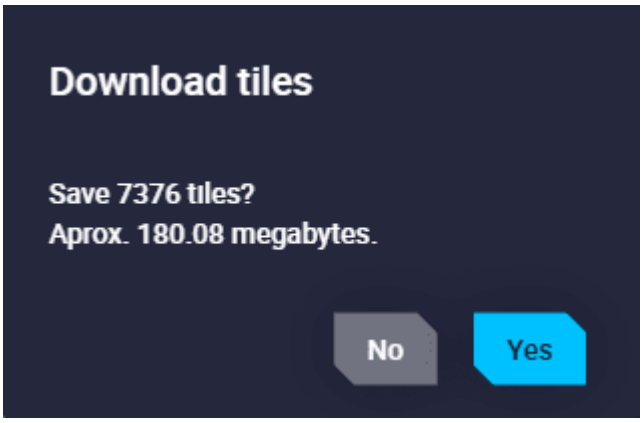
Save tiles

Save tiles

The tiles that the user is viewing will be downloaded. An example is shown below:

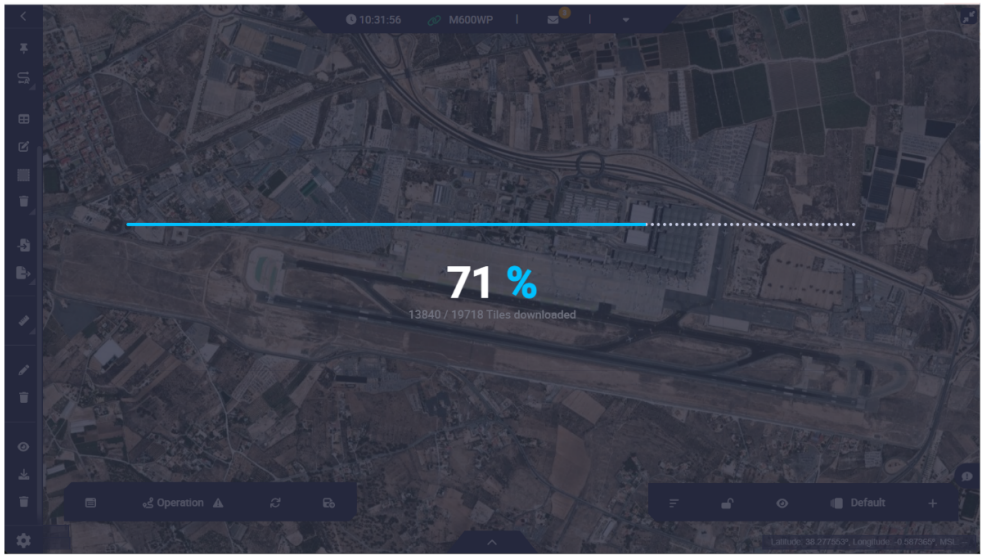
1. Click on this icon when the user is zooming in on the tiles in the desired area to download:

   > ⚠ **Warning**
   > As each zoom level in **Veronte Ops** counts as an independent map tile. When downloading map information, make sure that all relevant zoom levels are downloaded.

   Download tiles

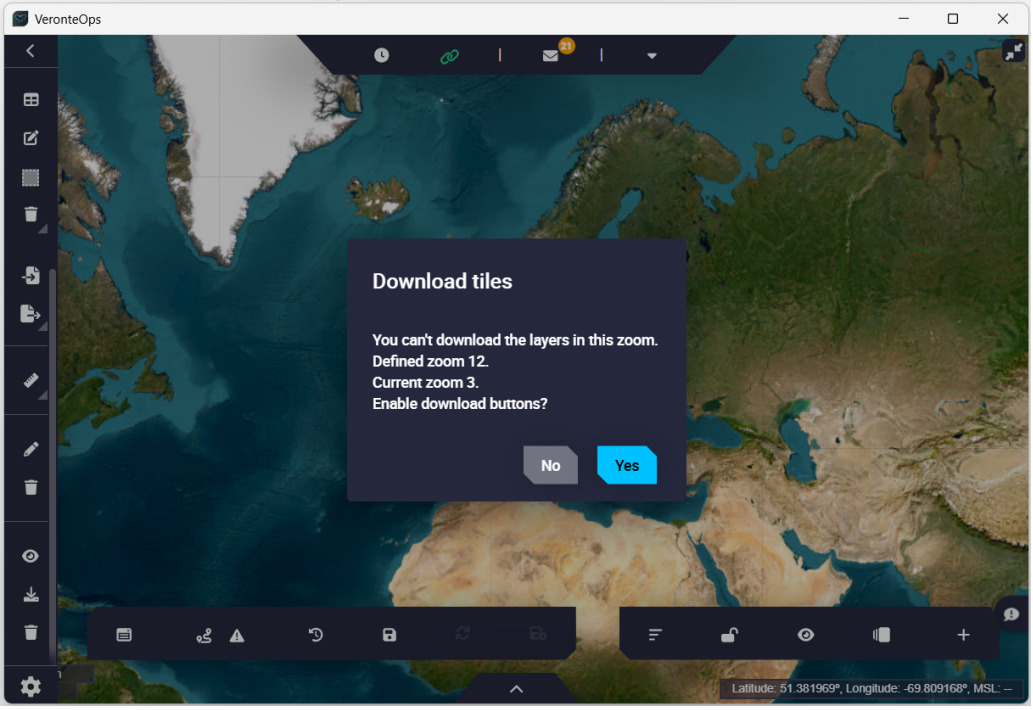
2. The tile download process will then begin:



Downloading tiles

3. Finally, by clicking on the above option, users can view the downloaded map tiles of the desired area.

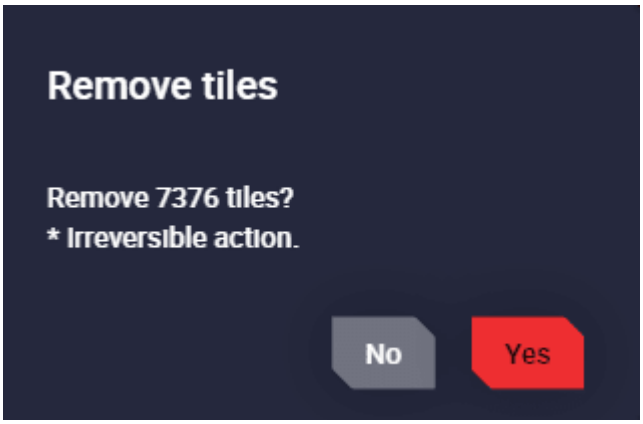
However, if there is no significant zoom, **Veronte Ops** will not download these files because there are too many of them. The following message will appear:



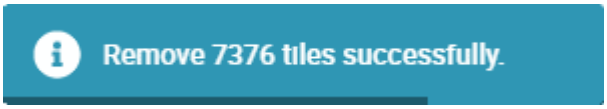
Not possible to download tiles

Remove tiles

Clicking on this icon will delete all tiles downloaded tiles. The following messages will appear:

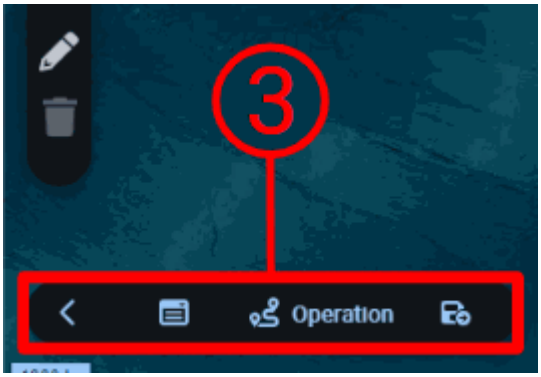


Remove tiles



Remove tiles successfully

3. Operation



Operation toolbar

The Operation toolbar is composed by 3 'parts': **Operation Panel**, **Operations** and **Operation actions**. However, in the figure above there is no Operation actions part because there was no **Veronte Autopilot 1x** connected.

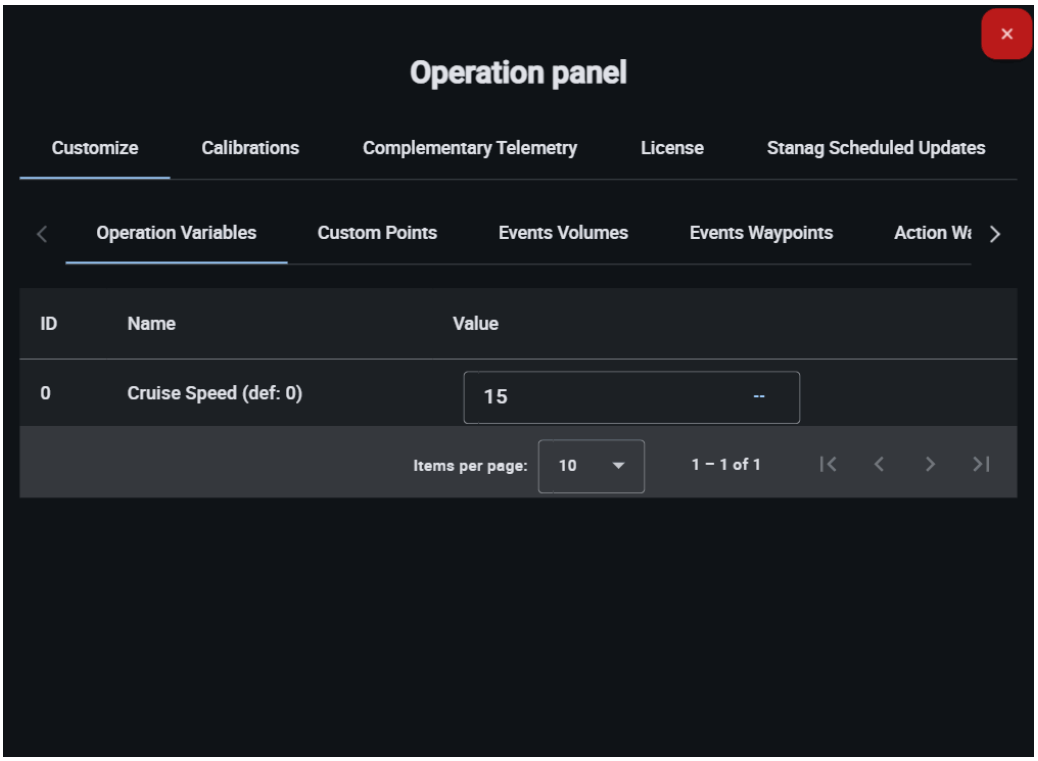
In the figure below, users can see these 3 'parts' when an **Autopilot 1x** is connected:

Operation toolbar - Autopilot connected

1. Allows closing Operation bar.
2. Operation Panel: Here the user can customize operation elements related to the operation, carry out some simple calibrations such as calibration of atmosphere, wind, DEM, etc., and manage complementary telemetry.
3. Operations: Users shall be able to export the operation loaded into the autopilot or import a different one and load it into the autopilot.
4. Operation actions: These are actions related to when changes are applied to the operation/mission, such as Revert, Save, Reload and Upload changes. This 'part' is only available when a **Veronte Autopilot 1x** is connected. In addition, when a change is made, the other actions (Revert, Save and Reload) appear.

Operation Panel

This menu is divided into 5 different functionalities: manage **operation elements** in the **Customize** tab, **simple calibrations** in the **Calibrations** tab, manage **complementary parameters** in the **Complementary Telemetry** tab, manage **software licensing** in the **License tab**, and handle interoperability updates in the **Stanag Scheduled Updates** tab.

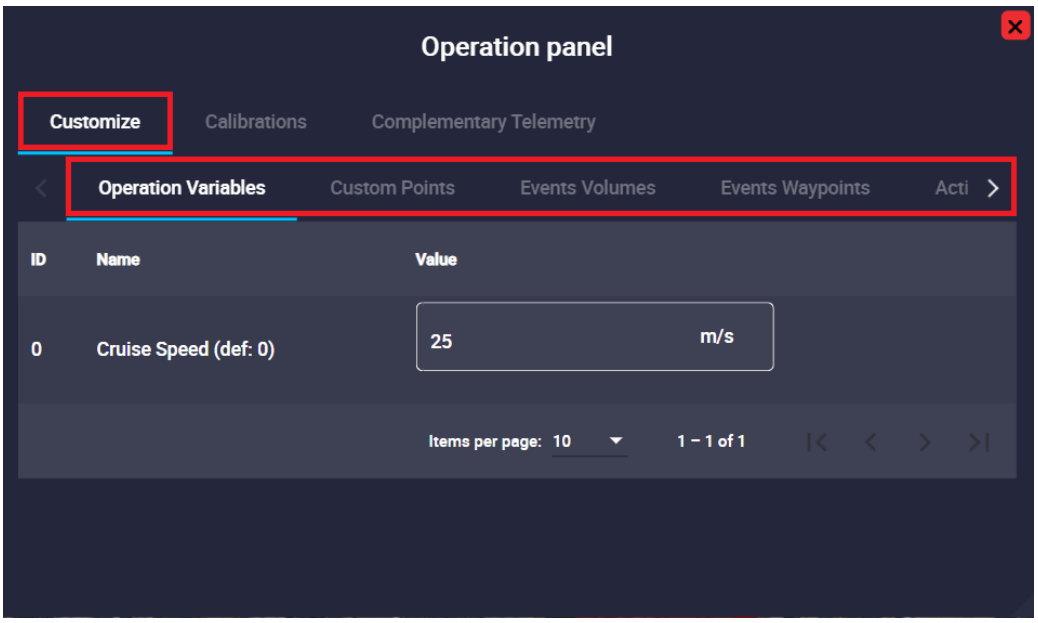


Operation Panel

All parameters included in this panel are explained in detail in the following sections.

Customize

In this part of the menu users can find all the operation elements used during the operation and mission: Operation Variables, Mission Points, Events Volumes, Events Waypoint, Action Waypoints, Events Marks, Runways, Spots, Initial position UAV and Operator position.



Operation Panel - Customize tab

If users wish to link a mark, Mission Point, prism, etc., to one of these variables, it is first necessary to define (rename) them in the **UI menu of the 1x PDI Builder software**, as described and explained in the corresponding section of the **1x PDI Builder** manual, click here to access it.

Operation Variables

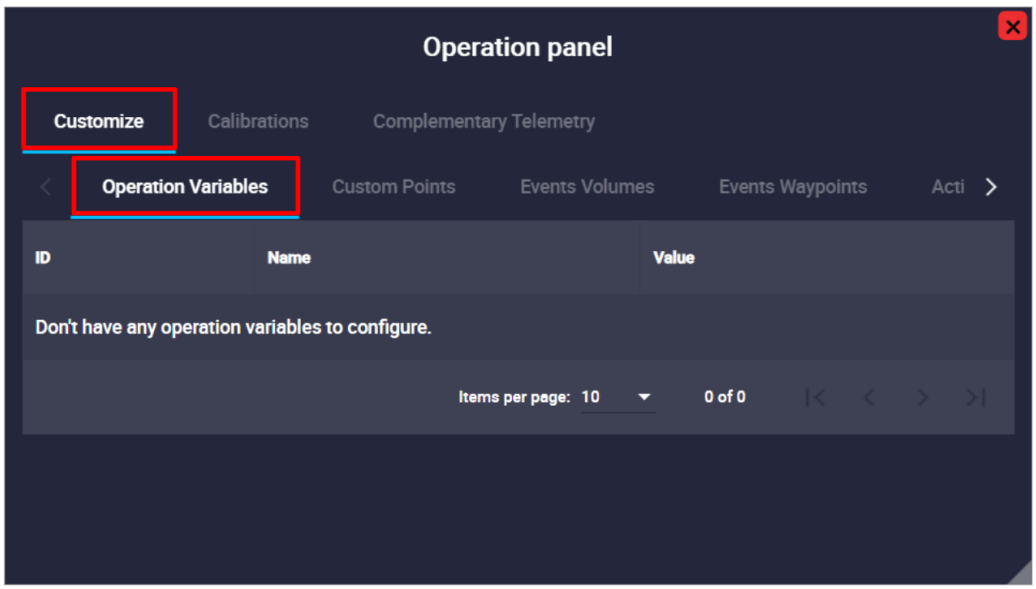
Operation Variables are configurable values, postitions and directions that can vary depending on the mission.

> **Important**
>
> Not setting the value of an operation variable may raise errors during the operation.

Examples of Operation Variables can be:

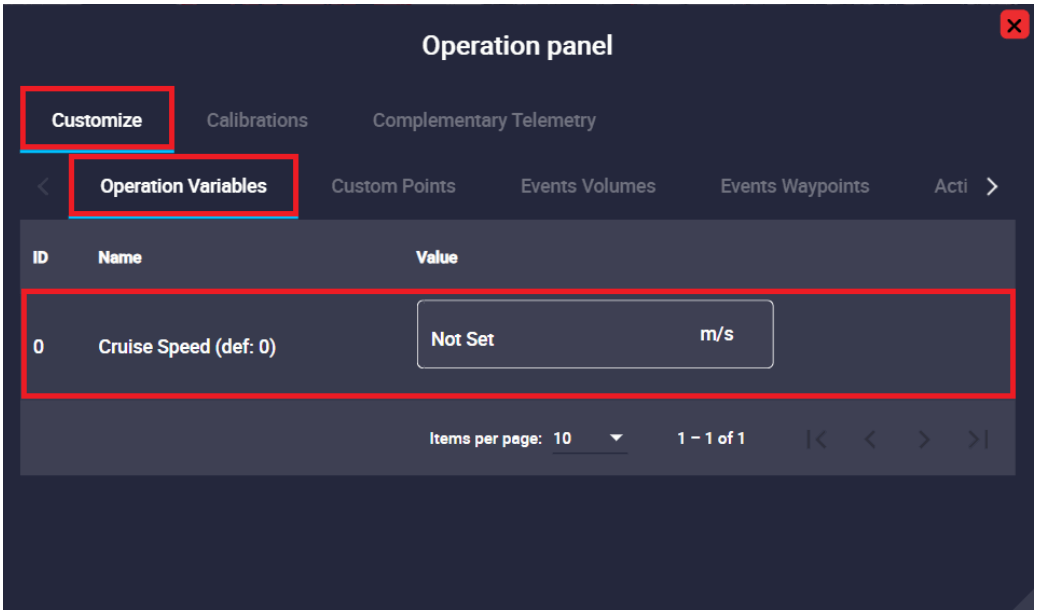
- Mission duration
- Cruise speed
- Flight level
- Takeoff and landing direction
- Home point
- Start of route

Their main advantage is that **it is not necessary to access Veronte Autopilot configuration to modify them**. In this way, the operator can modify certain parameters without the need of having access to the entire configuration.



Operation Variables

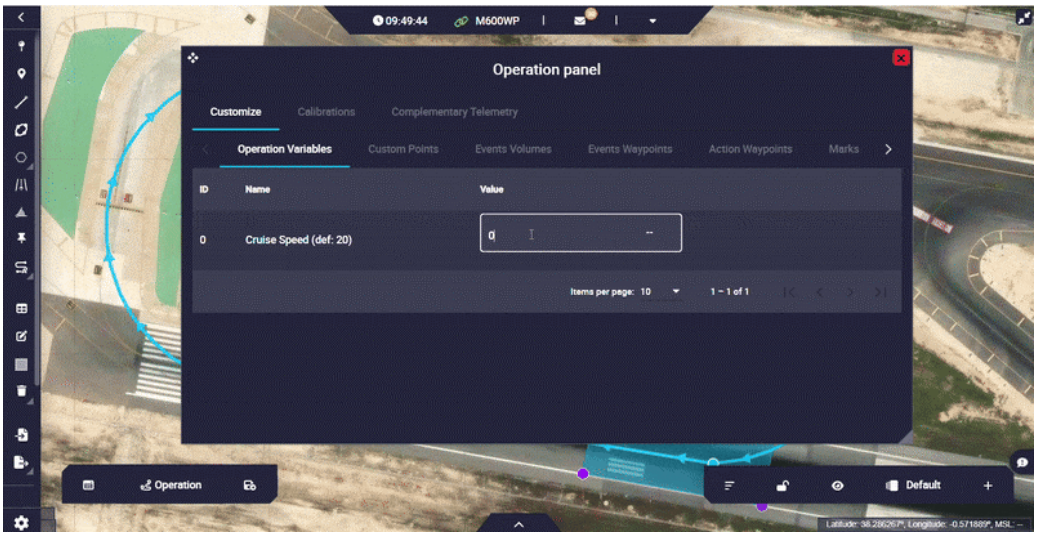
As explained above, operation elements must be created while building a Veronte Autopilot configuration. So, when a new Operation Variable is created, a new field will appear in the operation variables tab:



New Operation Variable

After this, the **value** of this variable can be defined using this menu. The user simply clicks on the **value cell**, enters the desired new value and saves the change (button, this is explained in the Save operation section below).

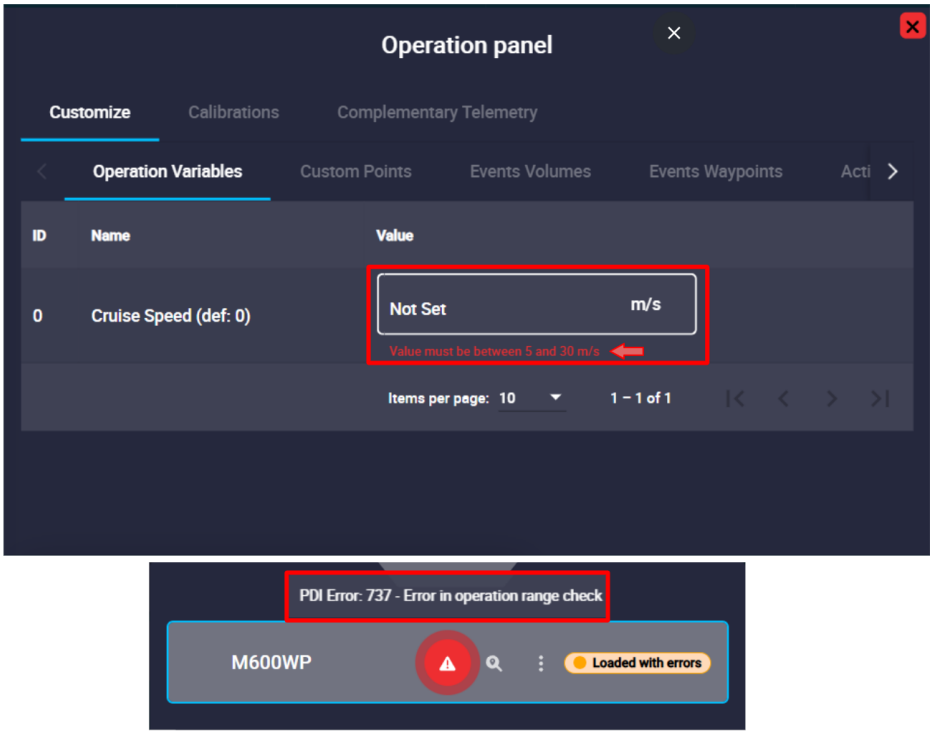
Below is an example of how to save changes locally and then upload them to the device:



Operations Variables configuration

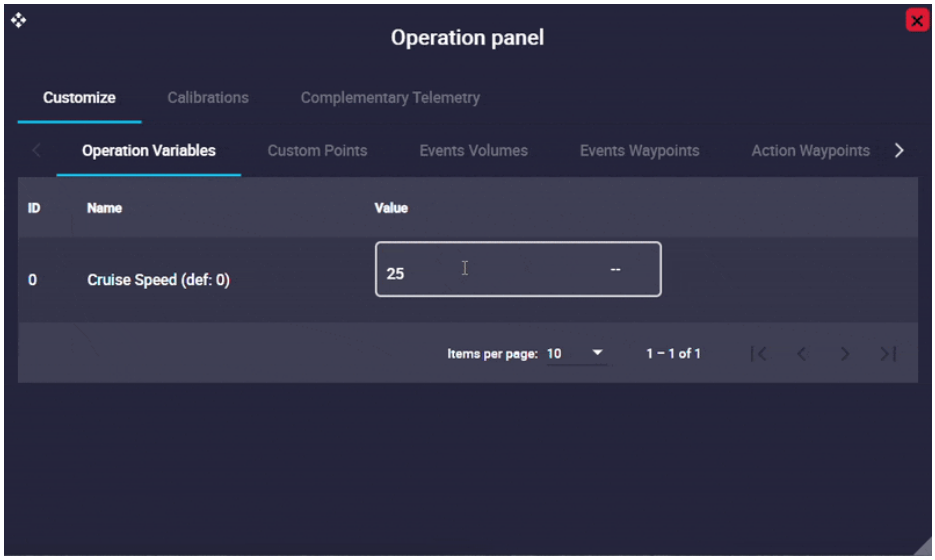
Users should note that these **Operation Variables can have minimum and maximum ranges** defined in their configuration in the **1x PDI Builder** software.

- When the Operation Variable **has a range** already configured:
  - It is mandatory to set a value in the **Operation Panel**, i.e. the value cell cannot be empty. Otherwise, the PDI error:737 will be shown in the Platform panel.



Operation Variable error: Value Not Set

- The **value** that users enter here must be **within the defined range**, otherwise, they will not be able to save it and the software will **automatically convert it to the closest value of the defined range** (minimum or maximum value of the range). In addition, an **error message** will be displayed in the cell indicating the range.



Operation Variable error: Value not in range

- When the Operation Variable **does not have** a range configured in **1x PDI Builder** software, setting a value is an option, not a requirement. If any value is entered, the value cell will display "Not Set".

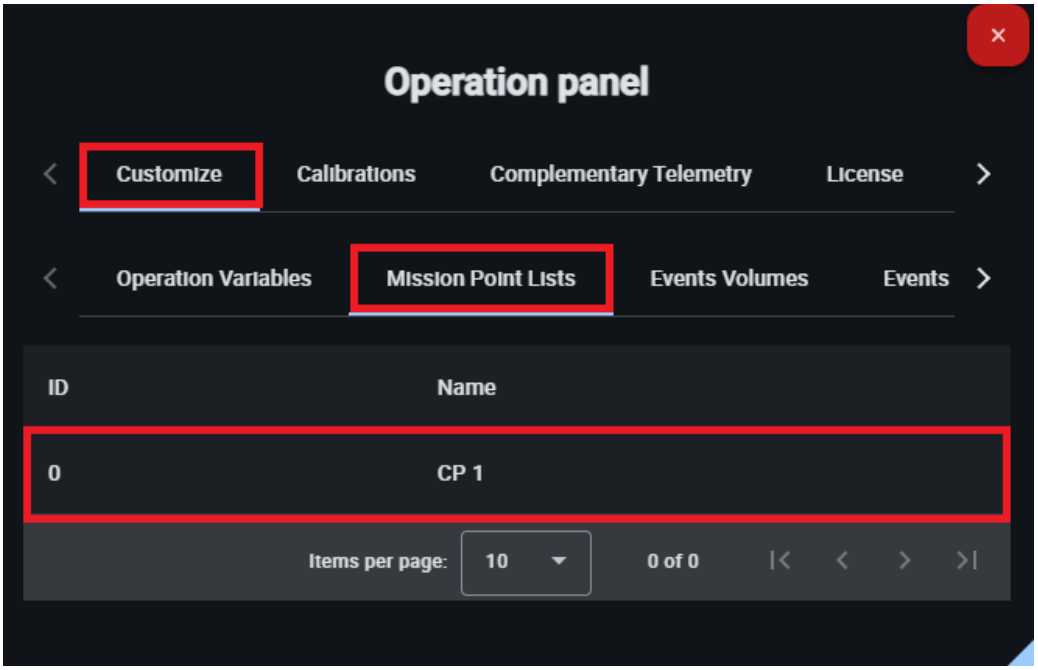
Furthermore, **units** can be changed in **Veronte Ops**, if they have been defined **1x PDI Builder** software.

> ⚠ **Warning**
>
> Although it is possible to modify operation elements 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.

Mission Points

A **Mission Point** is an operation element that can be linked to a waypoint of the configured mission.

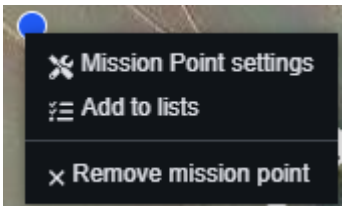
As explained above, an operation element only appears in this panel when it has been previously defined in the **1x PDI Builder** software.



Mission Point

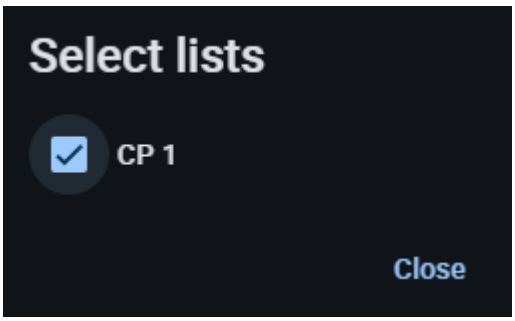
- **ID**: Mission Point identification number.
- **Name**: Name of this Mission Point **defined in the 1x PDI Builder** software.

Click on Add to lists To add a waypoint to a list.



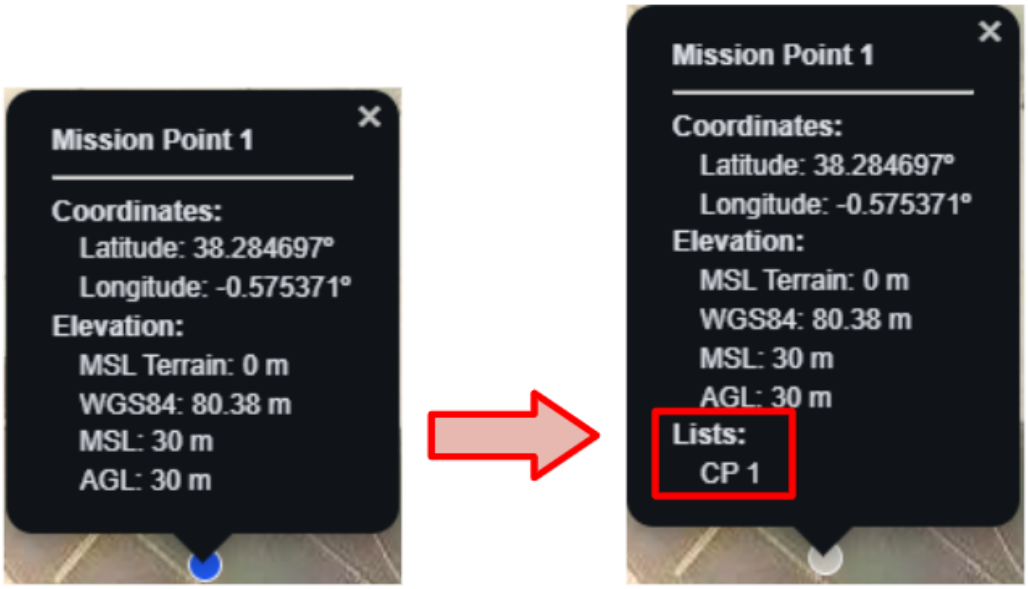
Mission Point configuration - Select point

Select the desired list and click Close.



Mission Point configuration - Select list

Moreover, by clicking on the linked waypoint, it will appear with the name of the lists:

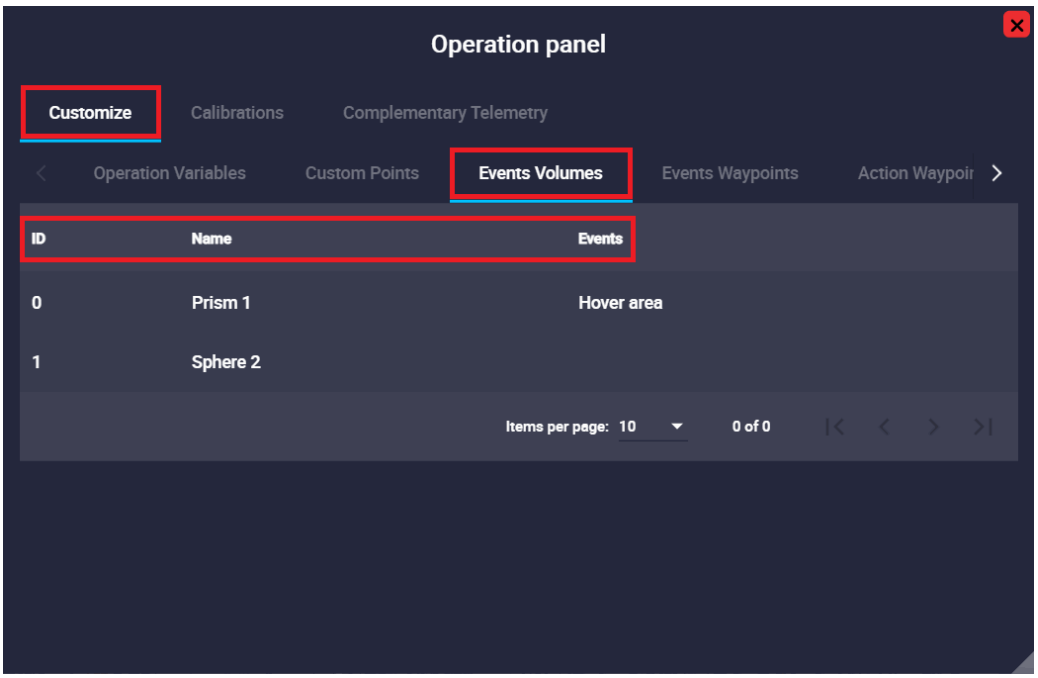


Mission Point description

Events Volumes

Set of volumes that assign events to **Prisms**, **Cylinders**, and **Spheres**.

These operation elements, Prisms, Cylinders and Spheres are detection volumes that can be linked to prisms, cylinders or spheres of the configured mission.



Volumes

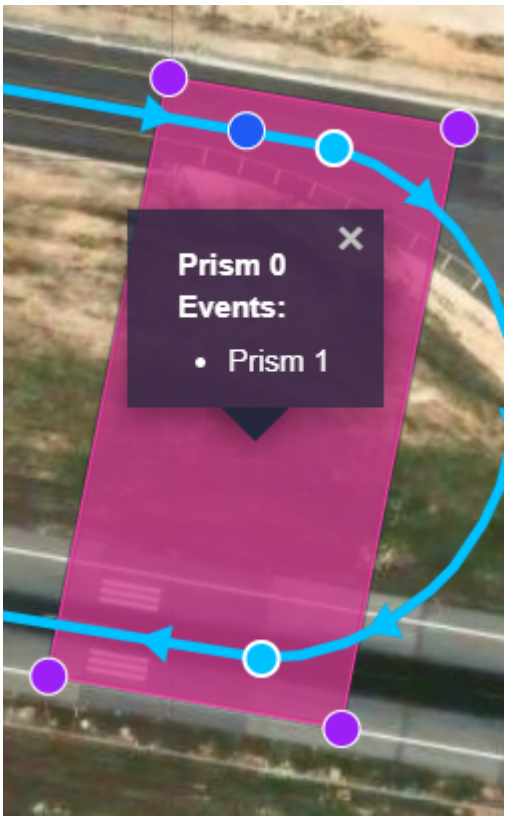
In the image above, the user can identify that there is a prism already defined and associated with an event, a sphere defined but without an associated event, and that there is no cylinder defined or associated to an event.

If the user wants to **define** cylindrical or spherical volumes, this must be done in the **1x PDI Builder** software, as described at the beginning of the Customize section. Then, to associated an event with a volume, the process to follow is explained in the Shape - Mission section.

The following information is displayed in the operation panel:

- **ID**: Event volume identification number.
- **Name**: Element identifier, this has been **defined in the 1x PDI Builder** software.
- **Events**: Here is displayed the list of any events that are linked to this element, such as triggers for automatic actions.

In addition, when clicking on the linked volume, its description will appear with the name defined by the user and the events linked to it will also appear:

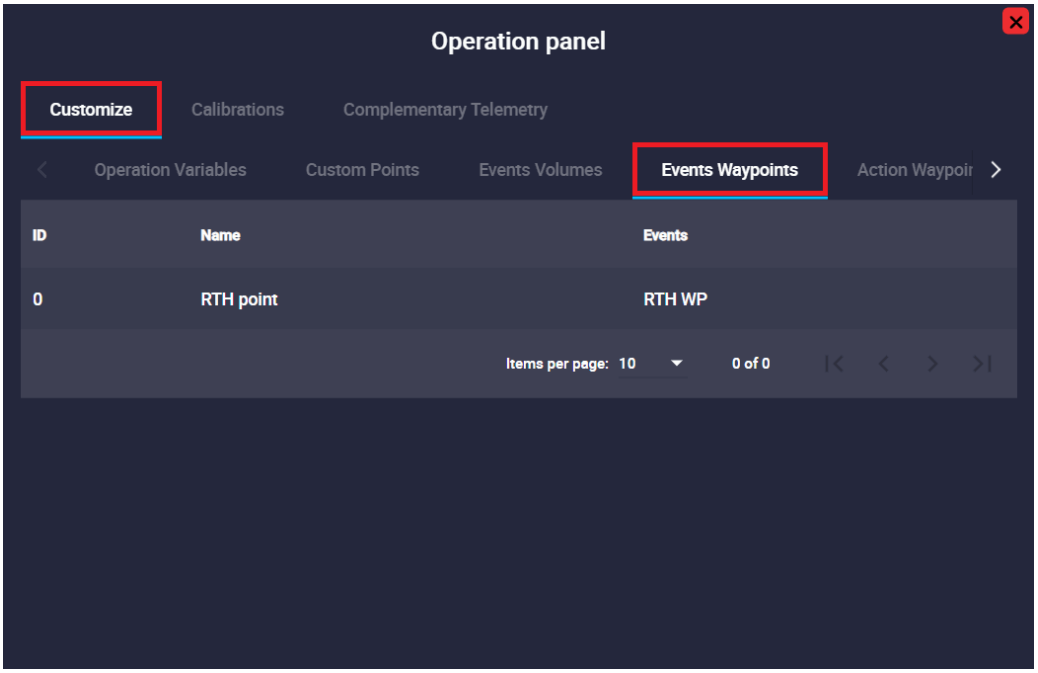


Volume description

Events Waypoints

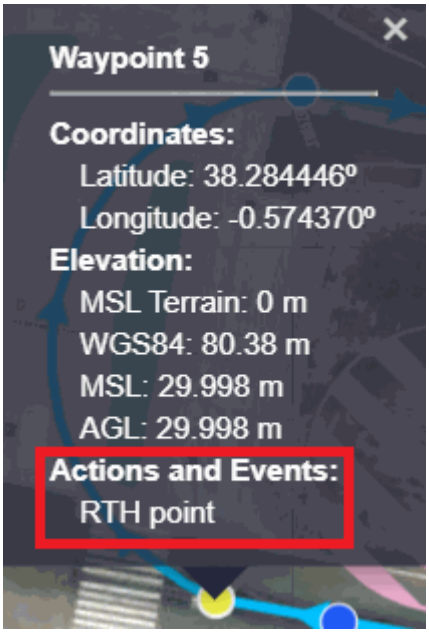
Events Waypoints allows for assigning properties and events to all waypoints belonging to this group.

This tab has the same options and works exactly the same way as the previous section (Events Volumes section).



Events Waypoints

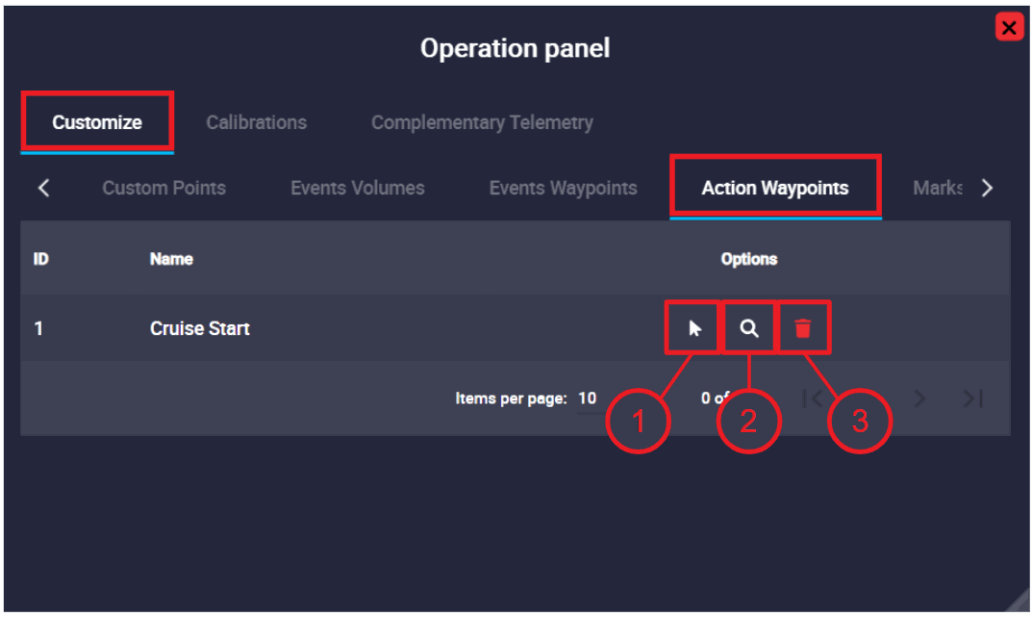
In addition, when clicking on it, the Waypoint name defined by the user will appear in its description, as well as the events linked to it. An example is shown below:



Events Waypoints - description

Action Waypoints

An **Action Waypoints** is a navigation points that are not just passing points, but are intrinsically linked to the execution of a complex action by the system (e.g., initiating a search pattern, releasing a payload).



Action Waypoints - Settings

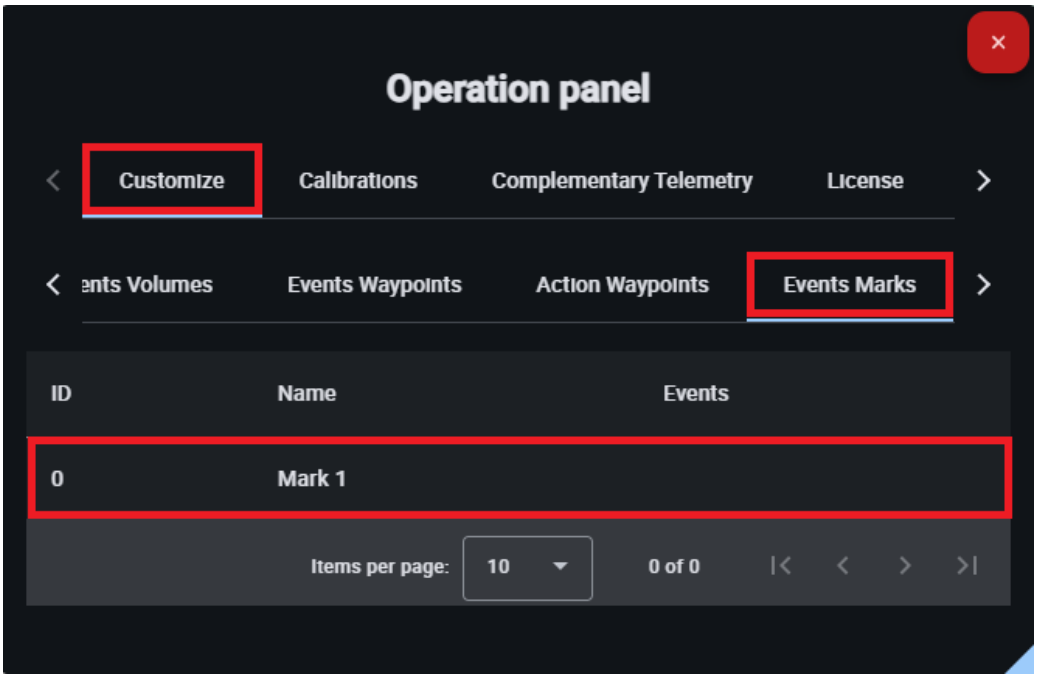
1. **Click Waypoint to set action**: To link the Action Waypooint point to a different waypoint, click here and select it on the map.
2. **Serach**: By clicking here, Veronte Ops will center and zoom in on the area of the map where this Waypoint is defined.
3. **Remove reference**: The user can remove the Waypoint configuration by clicking here.

Events Marks

A **Mark** is an operation element that can be linked to a Mark (reference placed in a patch) of the configured mission.

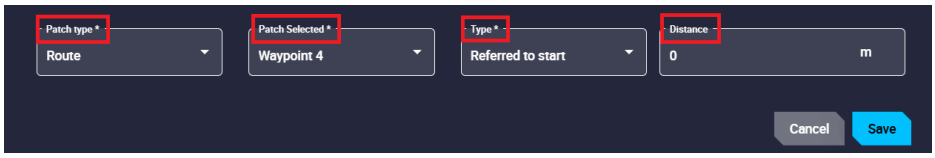
The procedure for defining and configuring a Mark is the same as described above in the Events Waypoints section.

Once a Mark is configured, the following options will appear in the operation panel:



Marks

- **ID**: Mark identification number.
- **Name**: Element identifier, this has been **defined in the 1x PDI Builder** software.
- **Events**: Here is displayed the list of any events that are linked to this Mark, such as triggers for automatic actions.
- **Options**: These are configuration options of the Mark:
  - **Edit/Create**:



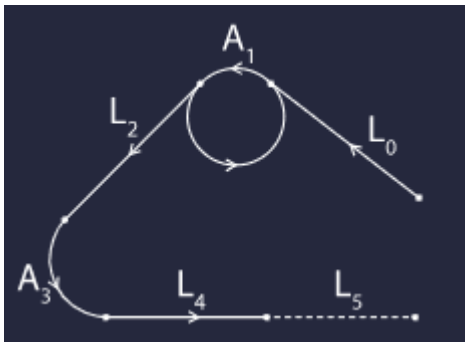
Marks - Edit

    - **Patch type**: The user can select different flight guidance phases where the Marks will be placed: **Approach**, **Climbing**, **Route**, **Taxi**, **VTol** and **Rendezvous**. Except for Route patches, for the rest of the guidance phase, **patches are generated when the user**, or an automation, **activates them**. As the user cannot select these patches, as **they cannot be generated initially**, this option will automatically create the Marking on the selected patch.
    - **Patch Selected**: Most flight guidance phases have predefined patches with specific names. The user can select where the Mark

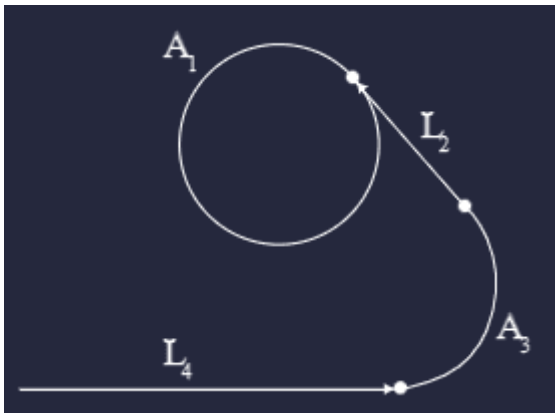
will be placed on those patches. A table summarizing the available options is shown below.

- **Type**: Right now the only possible option is "Referred to start" of the selected patch.
- **Distance**: Distance from the Mark to the start of the patch.

Climbing and Approach guidances are defined as follows:



Marks - Approach patches



Marks - Climbing patches

Approach	L0/A1/L2/A3/L4/L5
Climbing	A1/L2/A3/L4
Route	512 patches
Taxi	Taxi 0/Taxi 1
VTol	VTol 0/VTol 1/VTol 3
Rendezvous	

	Rendezvous 0/ Rendezvous 1/ Rendezvous 2

1. **Create Mark in map and select**: If the mark is not yet created in the mission, by clicking on this option, **Veronte Ops** will create it at the point on the map where the user clicks. In addition, **Veronte Ops** will automatically link this operation element to this newly created Marker.

   > ⓘ **Note**
   > This option also appears although the Mark is not yet configured (linked to a created Mark).

2. **Select in map**: To link this operation element to a different Mark, click here and select it on the map.
3. **Remove reference**: The user can remove the Mark configuration by clicking here.

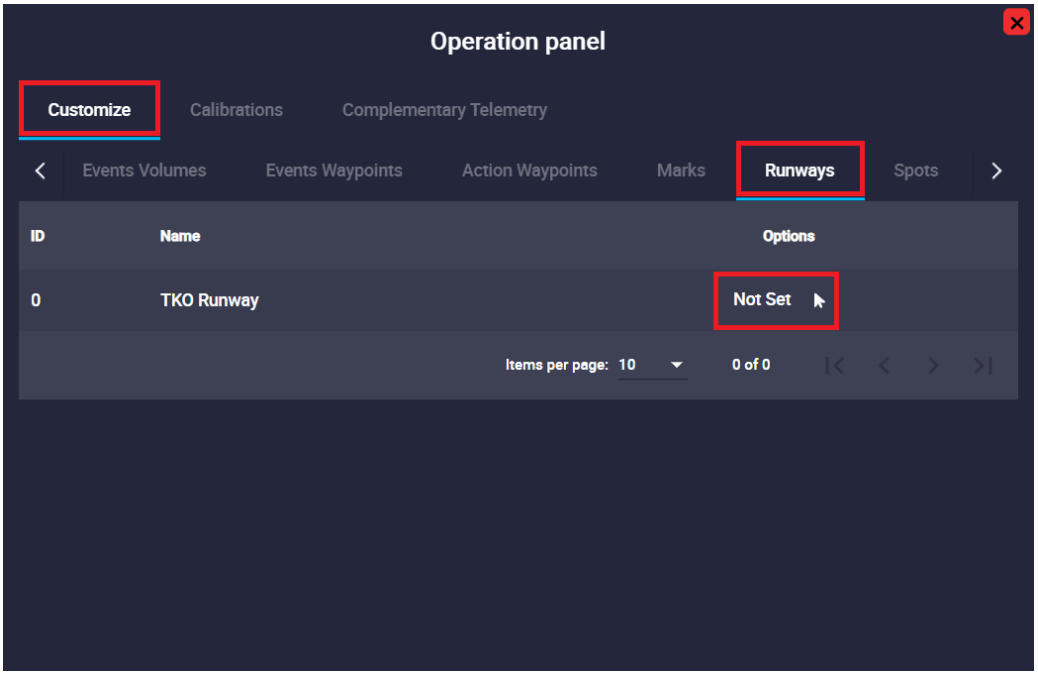
Moreover, if the event that is linked to the Mark has an associated icon, the icon will appear on that Mark. In addition, when clicking on it, its description will appear with the name defined by the user and the events linked to it will also appear. An example is shown below:

Marks - Icon and description

Runways

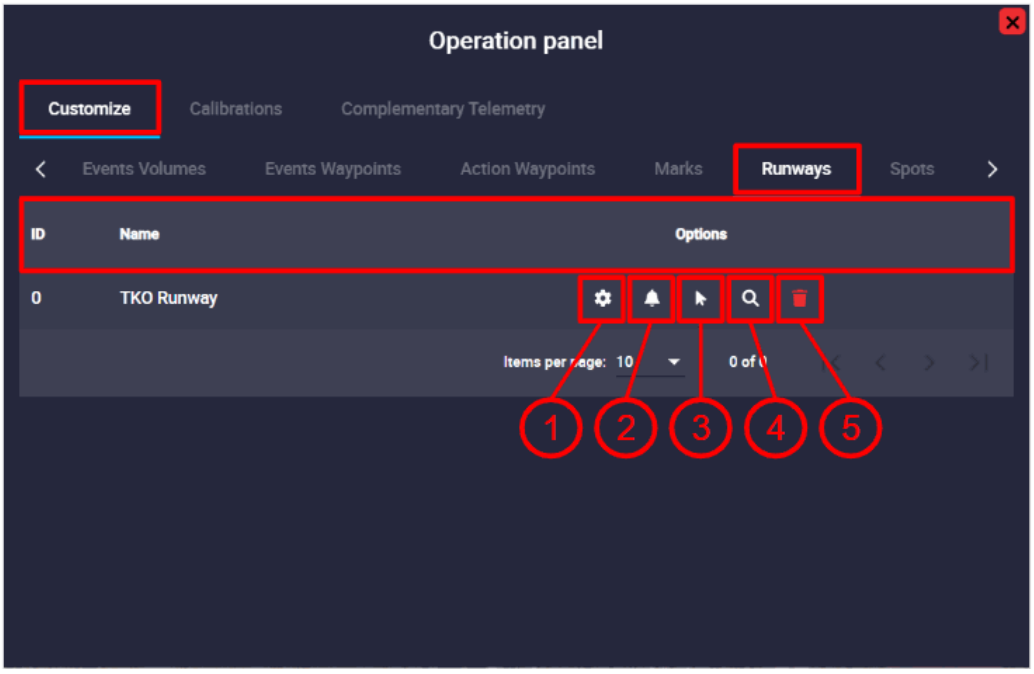
Runways are operation elements that can be linked to the runways of the configured mission.

Like all other operation elements, Runways have to be **defined in 1x PDI Builder** and then **configured by selecting a runway from the map**.



Runways

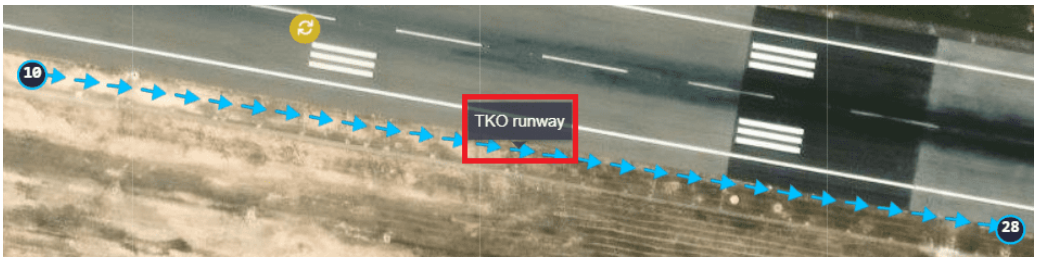
The following configurable options will then appear in the operation panel:



Runways configuration

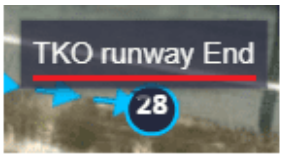
- **ID**: Runway identification number.
- **Name**: Element identifier, this has been **defined in the 1x PDI Builder** software.
- **Options**: These are configuration options of the Runway:
  1. **Edit**: By clicking here the user will be able to modify the runway settings. A detailed explanation of how to configure it can be found in the Runway - Mission section of this manual.
  2. **Configure Alarms**: Alarms can be associated to Runways. A detailed explanation of how to configure them can be found in Runway - Mission section of this manual.
  3. **Click to map**: To link this operation element to a different runway, click here and select it on the map.
  4. **Search**: By clicking here, **Veronte Ops** will center and zoom in on the area of the map where this Runway is defined.
  5. **Remove reference**: The user can remove the Runway configuration by clicking here.

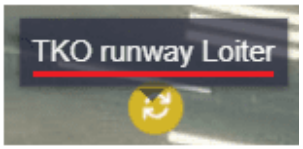
Moreover, by clicking on the linked runway, it will appear with the name of the configured Runway:



Runways description

If the user clicks on the Start, End or Loiter points of the Runway, they are also renamed:



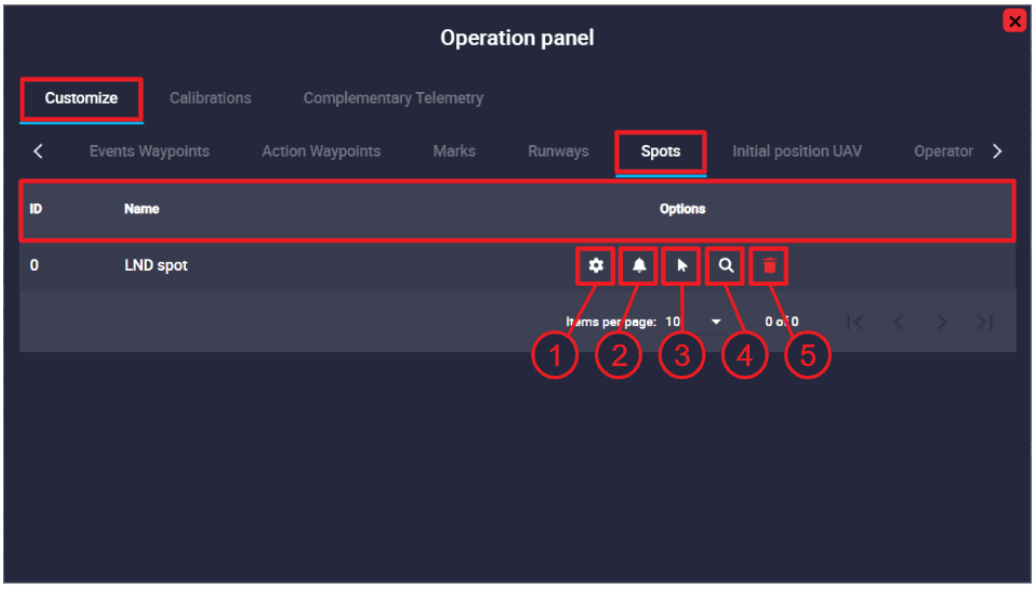


Runways - Start, End and Loiter points

Spots

Spots are operation elements that can be linked to the spots of the configured mission.

They function in the same way as the Runways operation elements:



Spots configuration

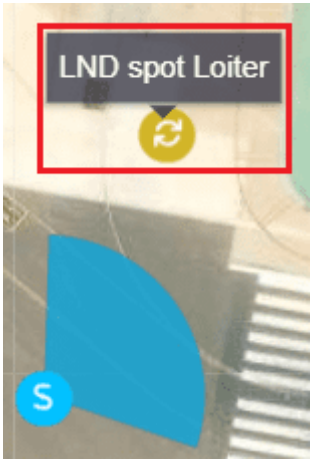
- **ID**: Spot identification number.
- **Name**: Element identifier, this has been **defined in the 1x PDI Builder** software.

- **Options**: These are configuration options of the Spot:
  1. **Edit**: By clicking here the user will be able to modify the spot settings. A detailed explanation of how to configure it can be found in Spot - Mission section of this manual.
  2. **Configure Alarms**: Alarms can be associated to Spots. A detailed explanation of how to configure them can be found in Spot - Mission section of this manual.
  3. **Click to map**: To link this operation element to a different spot, click here and select it on the map.
  4. **Search**: By clicking here, **Veronte Ops** will center and zoom in on the area of the map where this Spot is defined.
  5. **Remove reference**: The user can remove the Spot configuration by clicking here.

Moreover, by clicking on the linked spot, it will appear with the name of the configured Spot:

Spots description

If the user clicks on the spot loiter point, it is also renamed:

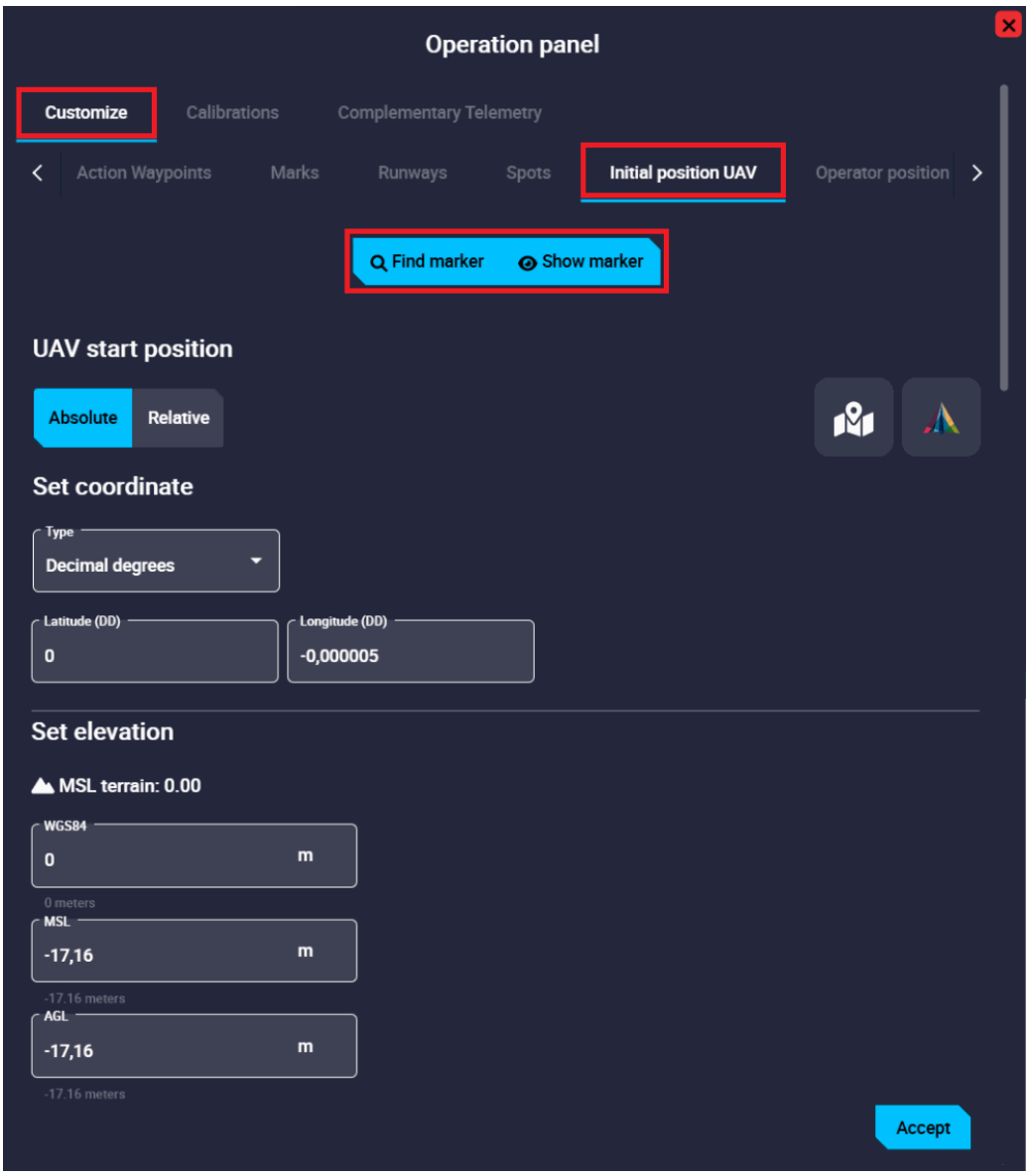


Spots - Loiter point

Initial position UAV

It is a marker indicating the initial position of the UAV.

In addition, users can find and center this marker on the map, as well as show/hide it.

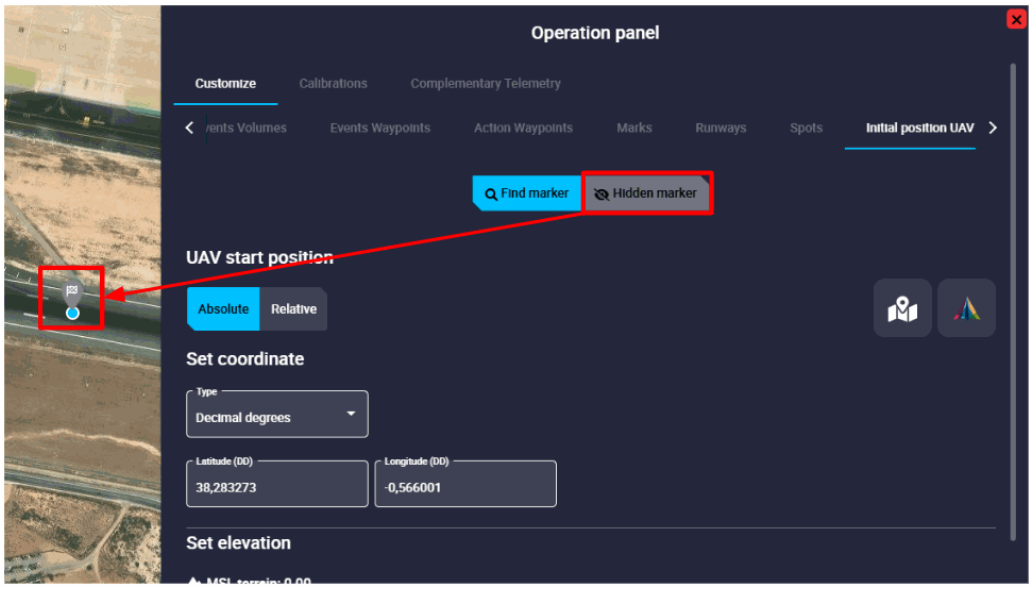


Initial position UAV

It is configured in the same way as **waypoints**. For a detailed explanatin of this, please refer to the Waypoint - Mission section of this manual.

After setting the position, it is necessary to click **Accept** and then **Save** and **Upload** the operation.

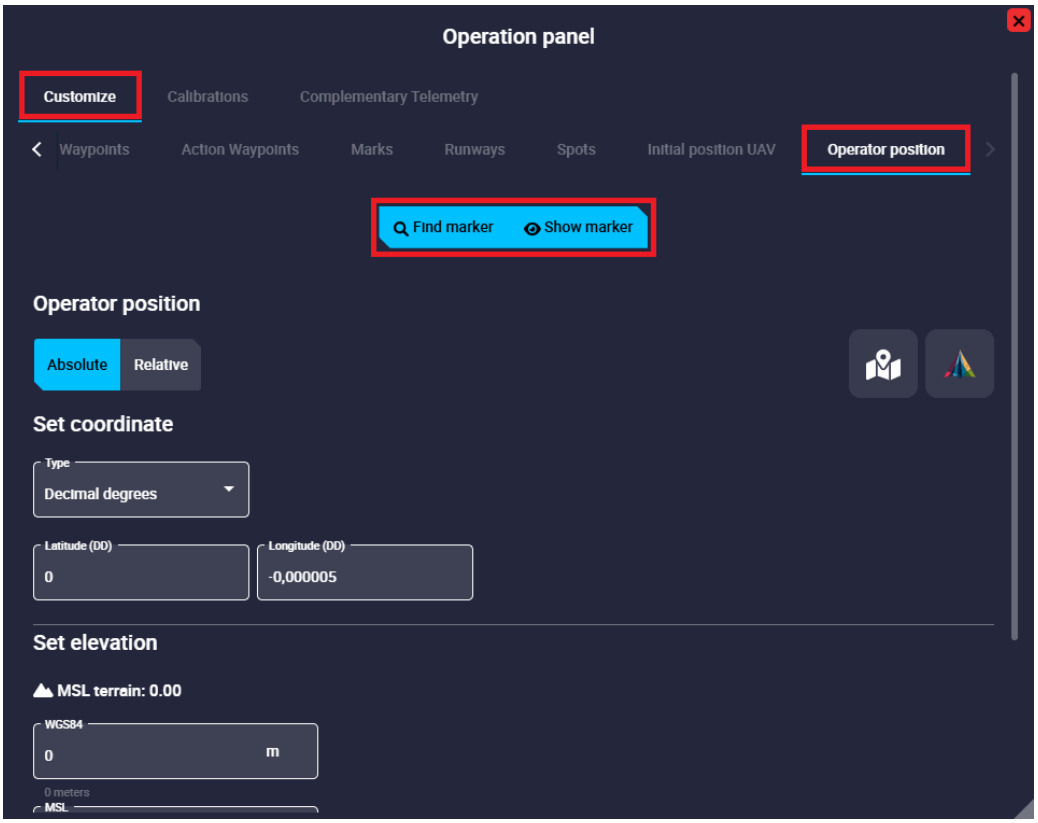
In the figure below, the marker is "shown" (as the hidden button appears):



Initial position UAV - Marker in map

Operator position

This tab enables the configuration of the Operator position.

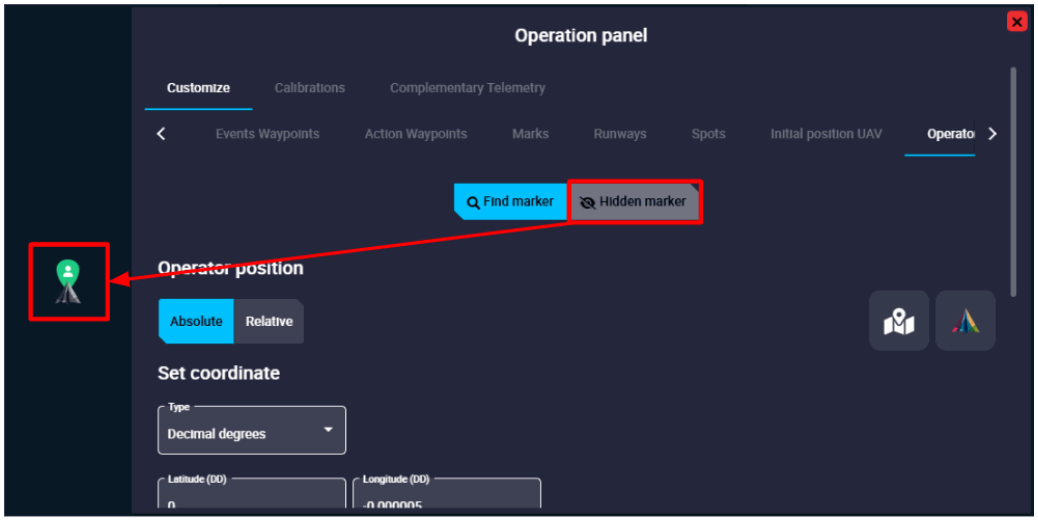


Operator position

It is configured in the same way as waypoints. For a detailed explanatin of this, please refer to the Waypoint - Mission section of this manual.

After setting the position, it is necessary to click **Accept** and then **Save** and **Upload** the operation.

In the figure below, the marker is "shown" (as the hidden button appears):



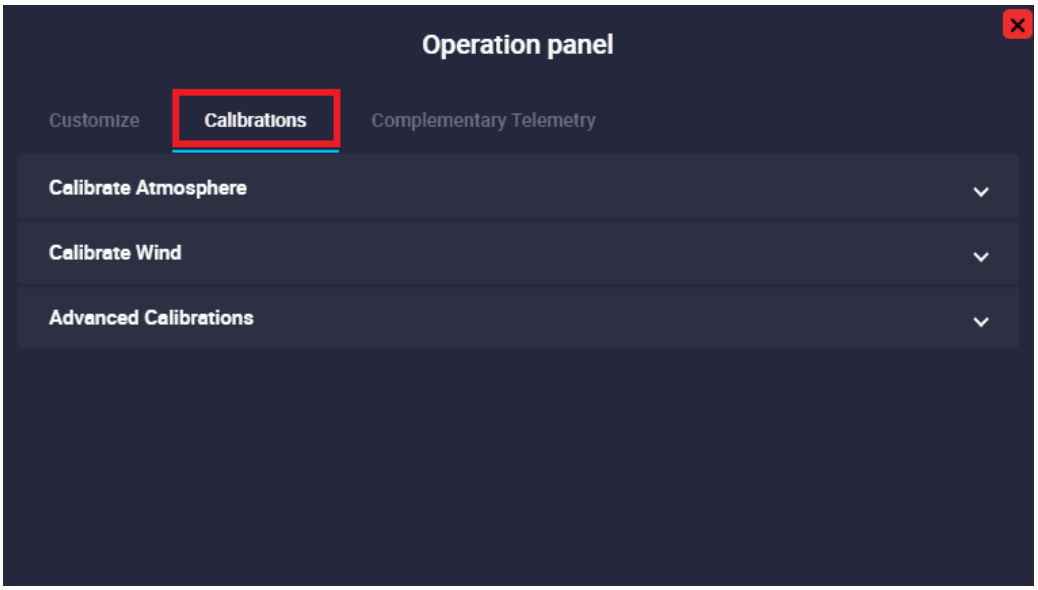
Operator position - Marker in map

> **Note**
>
> From the operator position defined in this panel, the distance allowed by the license to operate is calculated. Remember that Veronte Autopilot 1x has limited-operation depending on the license status. For more information on this limitation, please refer to the 1x Hardware Manual.

Calibrations

In this tab, the user can carry out simple calibrations during a standard operation. This way, there is no need of modifying the configuration or building a specific automation.

> **Note**
>
> Some of the following calibrations can also be triggered automatically using Automations.



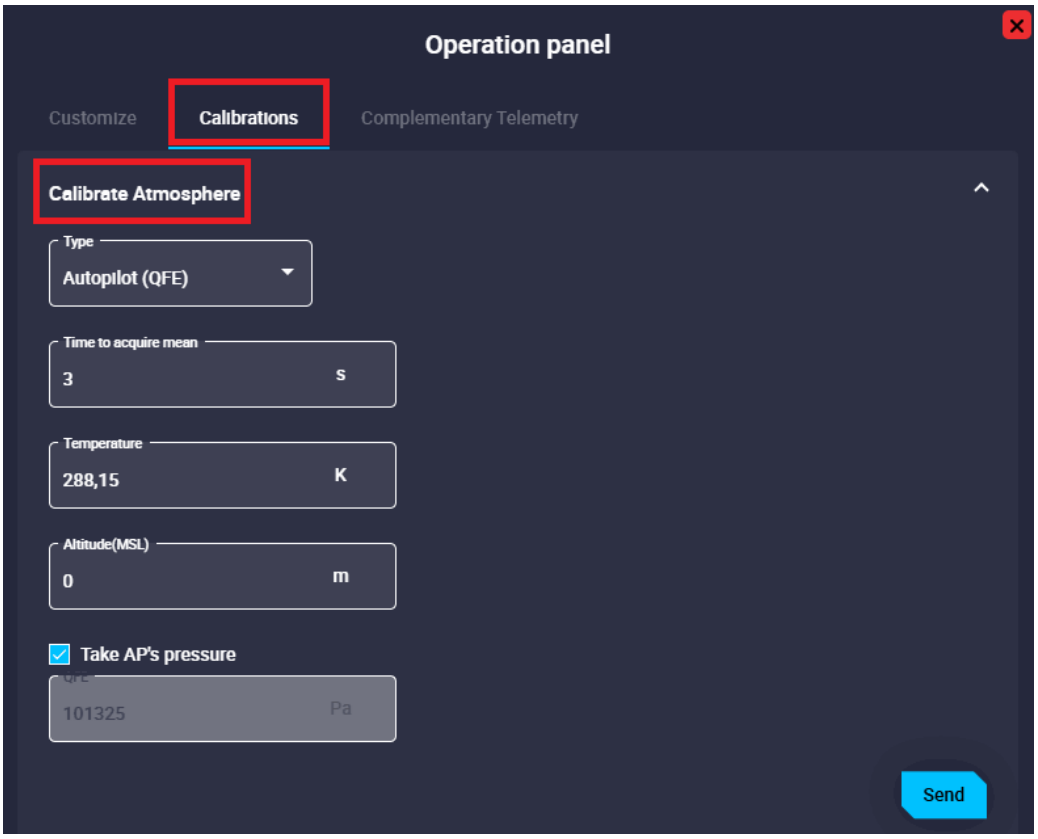
Operation Panel - Calibrations tab

> **Note**
>
> These calibrations will never modify the autopilot's current configuration: the changes are volatile, and will disappear once the system is rebooted.

> **Warning**
>
> To send any of these calibrations, it is necessary to click on the 'Send' button.

Calibrate Atmosphere

Calibration for MSL calculation with barometric pressure.

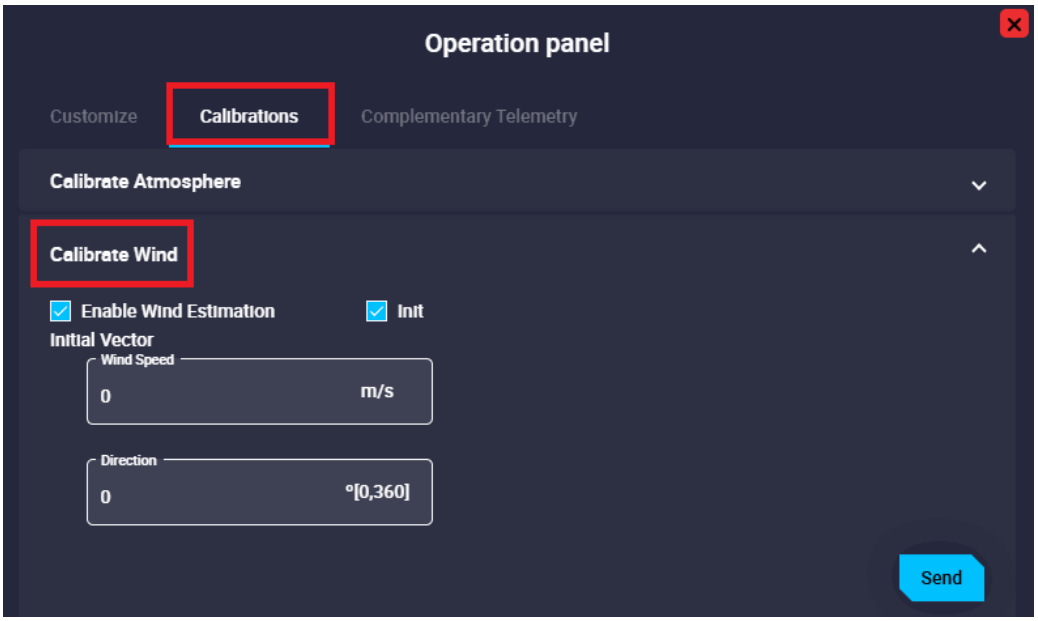


Calibrate Atmopshere

- **Type**: QFE and QNH options are available.
- **Time to acquire mean**: Specified time during which the static pressure is read from the static pressure sensor.
- **Temperature**: Outside air temperature.
- **Altitude (MSL)**: Actual MSL altitude.
- **Take AP's pressure**: If enabled, static pressure will be selected from the autopilot measurement, otherwise users will have to enter it manually.

Calibrate Wind

This command allows to enter initial values for wind state and start wind estimation algorithm.

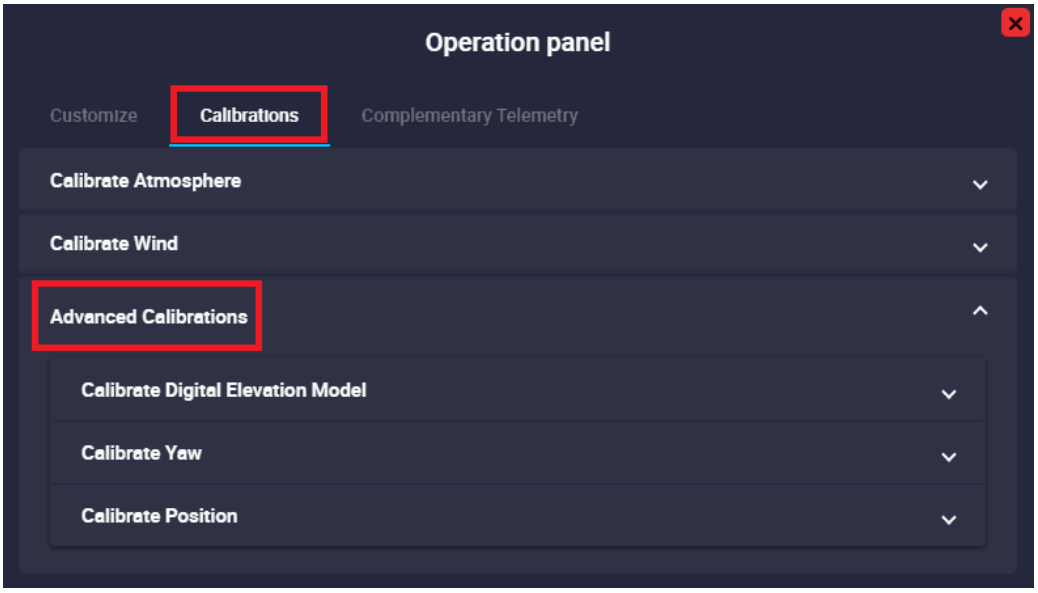


Calibrate Wind

- **Enable Wind Estimation**.
- **Init**: By enabling it, an initial wind vector can be set to a faster convergence of the estimation.
- **Wind Speed**: Module of the initial wind speed.
- **Direction**: Argument/Direction of the initial wind speed.

Advanced Calibrations

The following are the advanced calibrations that can be performed by the user during operation:

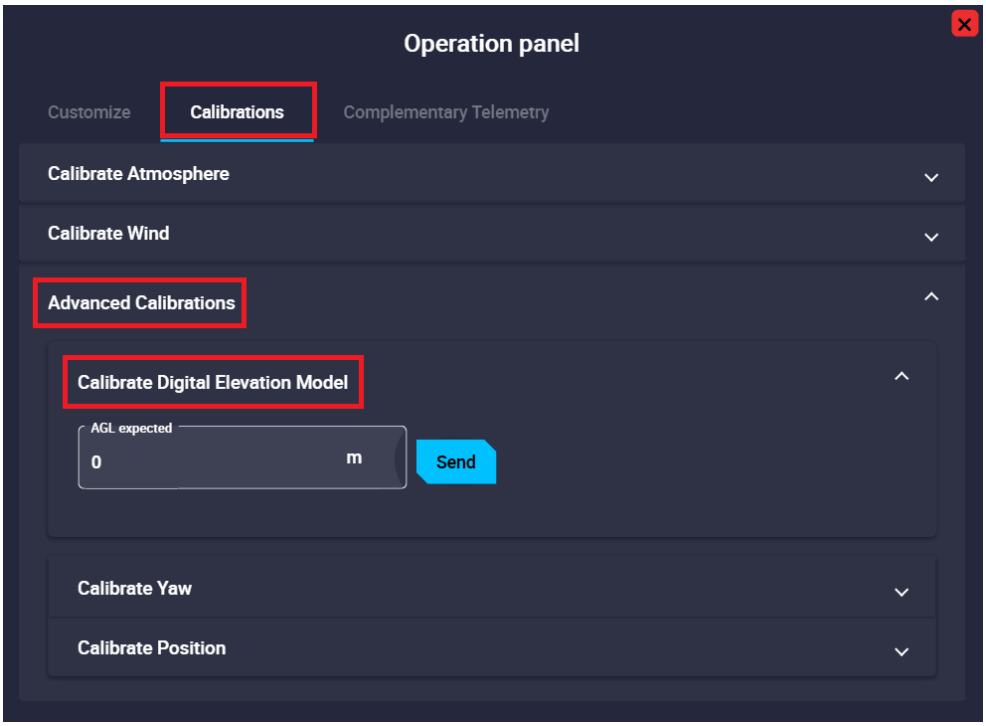


Advanced Calibrations

- **Calibrate Digital Elevation Model (DEM)**

  This calibration calculates the offset that the SRTM of the current point should have so that the estimated AGL results in the desired AGL (the one indicated in the calibration).

  > ⚠ **Warning**
  > - This offset is only valid for the point where the DEM has been calibrated.
  > - Always perform this action on the ground, unless an accurate estimation of current AGL is available.

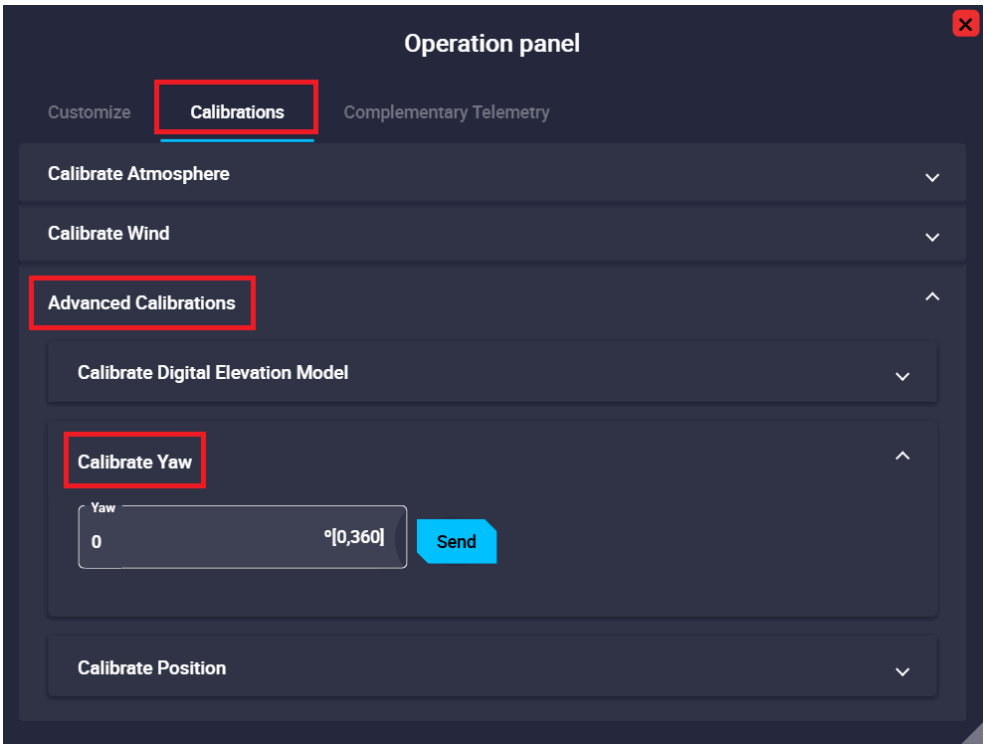


Advanced Calibrations - Calibrate DEM

- **Calibrate 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 **overrided**.



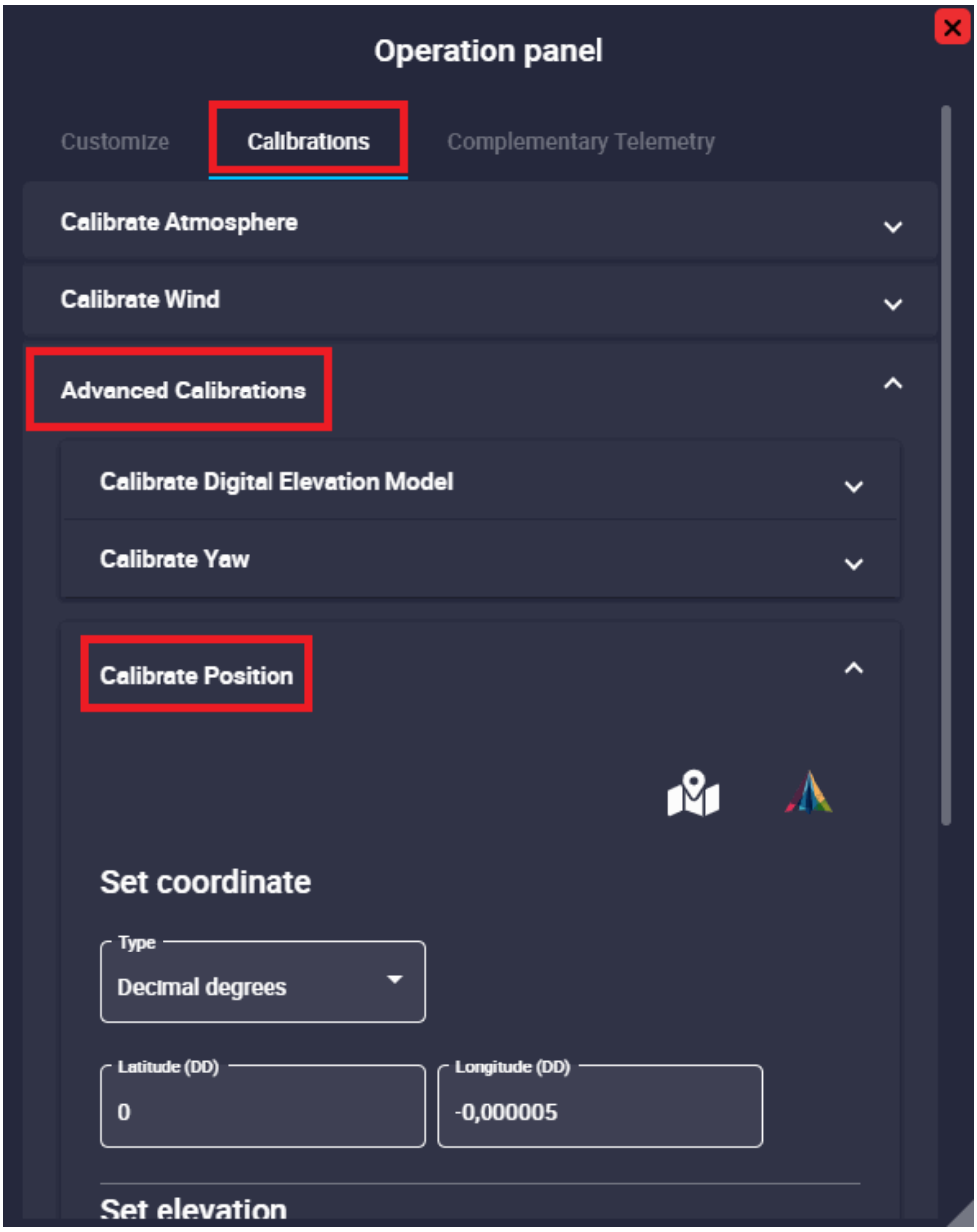
Advanced Calibrations - Calibrate Yaw

- **Calibrate Position**

  Allows to manually modify the Position Navigation state.

  The configurable parameters of the position calibration are the same as for configuring the position of a **waypoint**. So they have already been described in detail earlier in the Waypoint - Mission section of this manual.

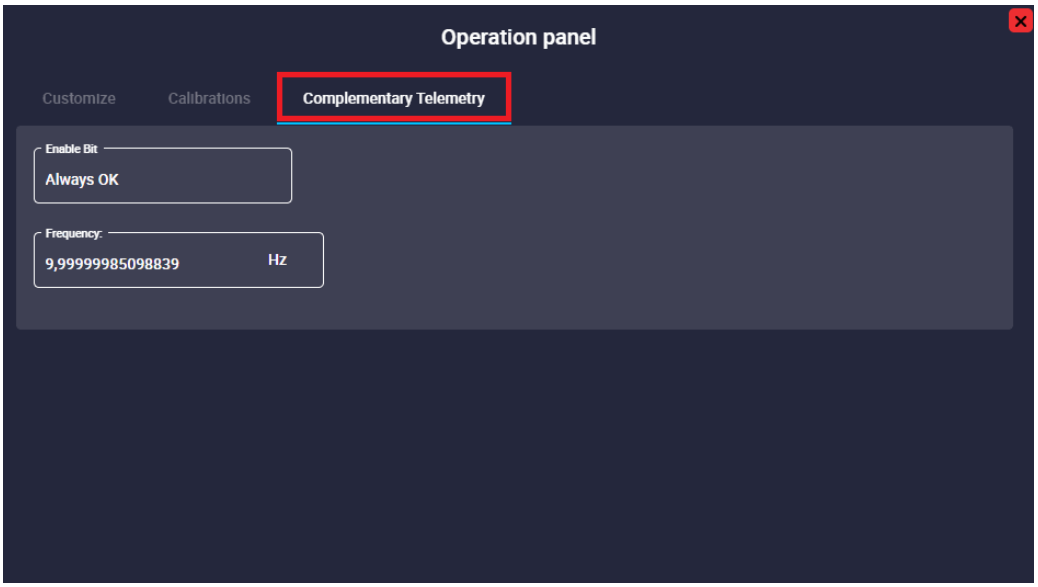
> ⚠ **Warning**
>
> If there is any **absolute positioning sensor active** (i.e. GNSS), this command will **not work** since it will be automatically **overrided**.



Advanced Calibrations - Calibrate Position

Complementary Telemetry

The Complementary Telemetry panel enables users to include extra flight data. While telemetry is primarily configured in 1x PDI Builder, further parameters can be added directly within Veronte Ops.



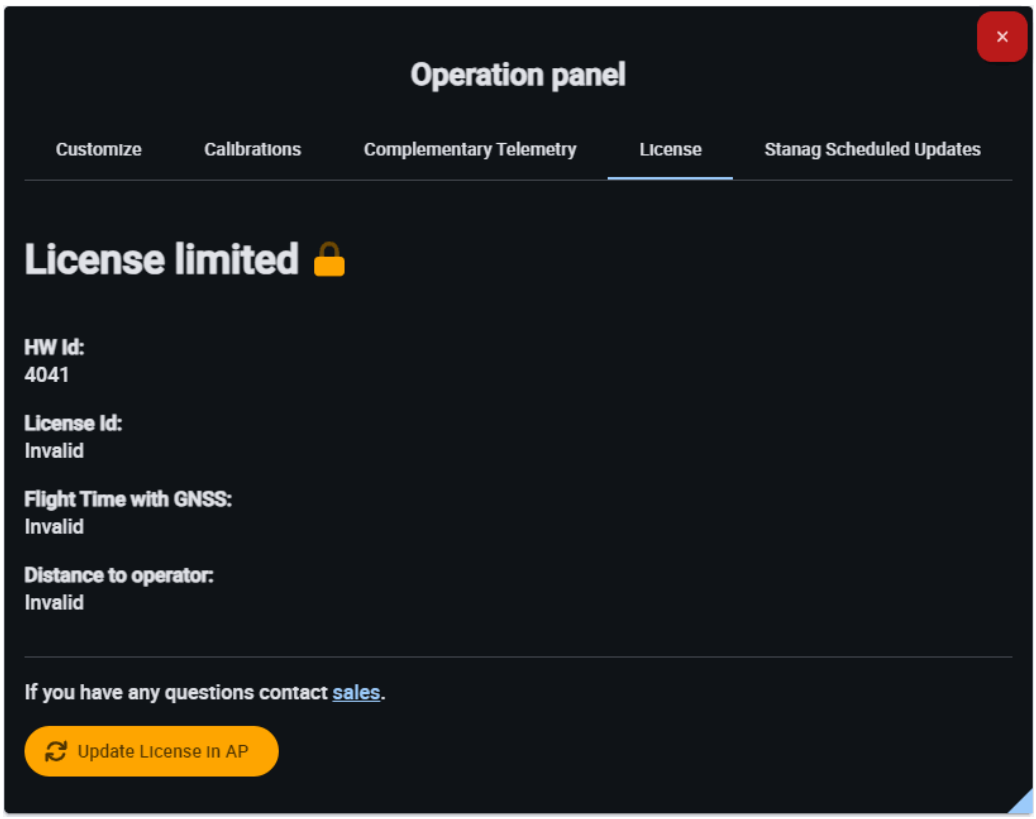
Complementary Telemetry - Panel

- **Enable Bit**:
  - Always OK: Enable Complementary Telemetry.
  - Always Fail: Disable Complementary Telemetry.
- **Period**: Determines the time interval between the transmission of one data packet and the next. It is possible to choose between s, min, h, µs and ms.

> **Important**
>
> The complementary vector telemetry always runs at a fixed frequency of 10 Hz (not configurable), and variables are sent uncompressed. Therefore, it is recommended to add the variables to the 'Data to Vapp' telemetry vector to configure the desired frequency, variable compression/uncompression, etc.

License

The License section of the Operation panel allows managing activation and operational restrictions for the autopilot.

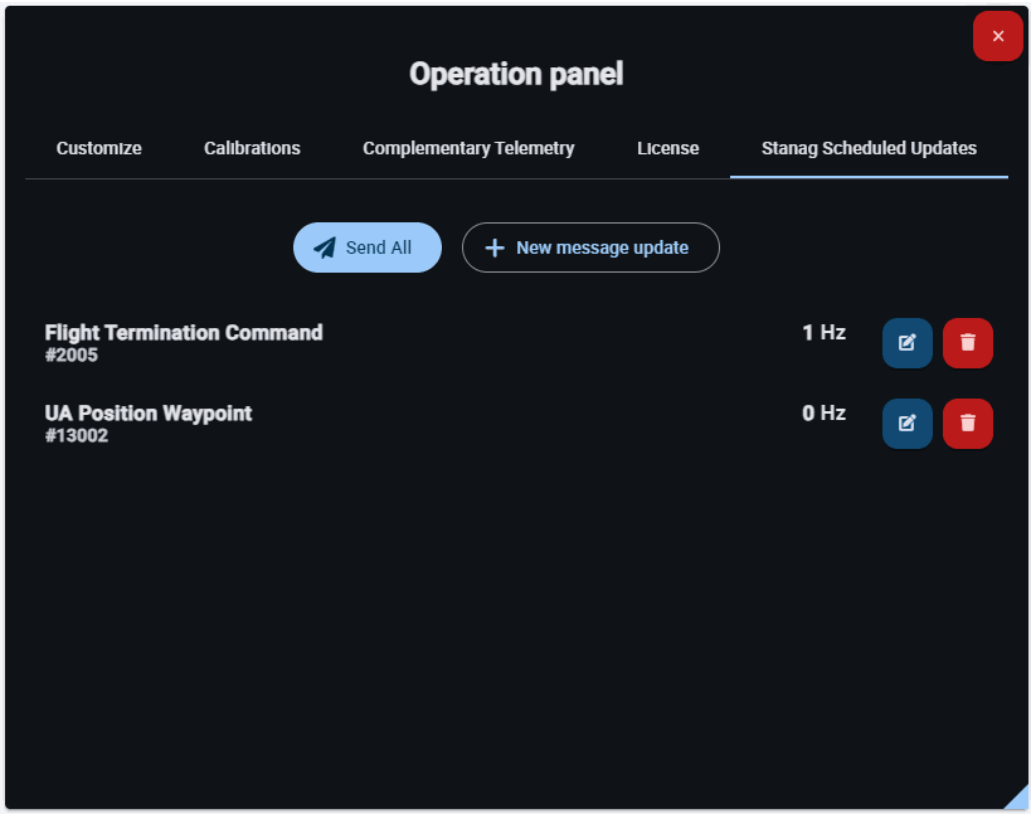


License

- **Status**: Displays license status (e.g., "License limited"), hardware identifier (HW Id), and current License ID.
- **Operational Limits**: Reports restrictions on critical parameters such as Flight Time with GNSS and maximum Distance to operator.
- **Update**: The Update License in AP button allows loading new permissions or renewing the license directly on the unit.

Stanag Scheduled Updates

The Stanag Scheduled Updates section allows configuring the transmission frequency of messages compliant with the STANAG standard.

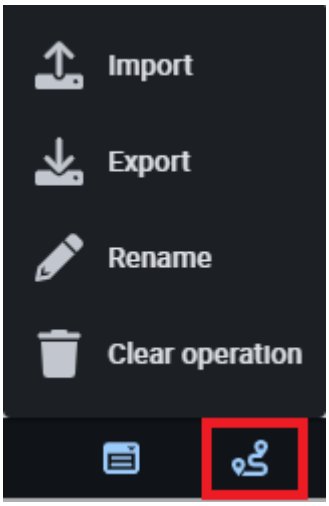


Stanag Scheduled Updates

- **Message Scheduling**: Displays the list of active messages.
- **Frequency (Hz)**: Allows defining the number of updates per second for each individual data packet.
- **New message update**: Permits adding new message identifiers to the transmission list.
- **Send All**: Immediately sends all messages configured in the list.
- **Edit and Delete**: Enable changing parameters or removing scheduled messages.

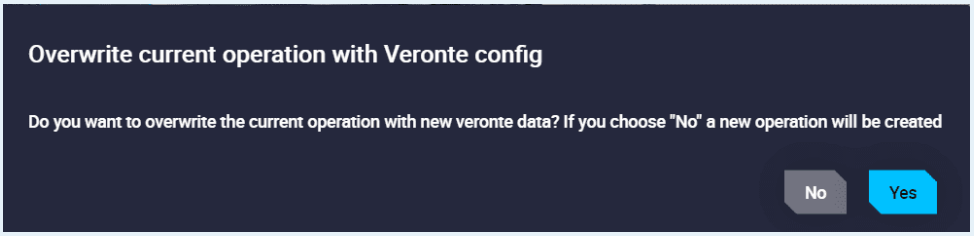
Operations

By clicking here, it is possible to modify the operation.
The following options are available:



Optionns

- **Import**: The user can import an operation, the available formats are: `ZIP`, `KML` and `GEOJSON`.
- **Export**: The current operation can be exported, the available formats are: `ZIP`, `KML` and `GEOJSON`.
- **Rename**: The user can rename the operation as desired.
- **Clear operation**: Clicking here will delete the selected operation.

> ⓘ **Note**
>
> When **a configuration is saved in the 1x PDI Builder software**, the following message will appear to inform the user that there is a new operation loaded in the **Autopilot 1x**:
>
> 
>
>
> **Operation overwrite/new**
>
> - If the user does not select anything, **after 10 seconds** this message will disappear and a **new operation will be created**.
> - If the user selects **NO**, a **new operation** will also be created, as described in the message.
> - If the user selects **YES**, no new operation will be created and the changes saved in the **1x PDI Builder** software will be applied to the current operation, i.e. the **current operation will be overwritten**.

Operation actions

Revert Operation changes

- When a change is made, it is possible to revert it by pressing this button. This is only possible **if the changes have not yet been saved**.

Save Operation

- **Local saving** of the mission.

Reload

- The Reload command reverts the operation to the **last saved version**.

> ⚠ **Warning**
>
> This operation is only available if the mission has **not been uploaded** to the device.

To apply any change to the device, it is necessary to save it by pressing this button.

Upload to '(Platform name)'

- To update the operation loaded on the autopilot with the **new saved changes**, click on this button.
- If there is no change to upload to the operation configuration, the following message will appear:

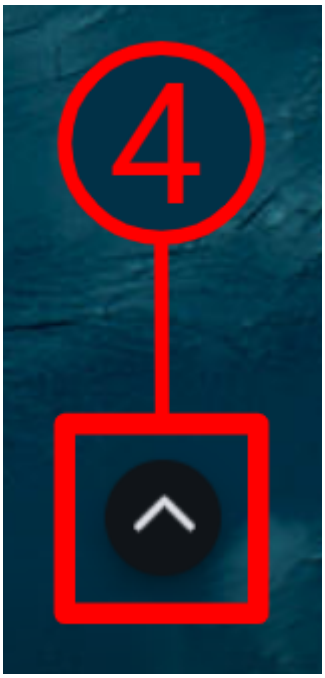
Nothing to Save in '...'

4. Platform

This panel lists the linked **Veronte Autopilots**, either connected by radio link or directly to the laptop, summarizing important information about these platforms such as:

- Platform name
- Platform connection status
- Platform license

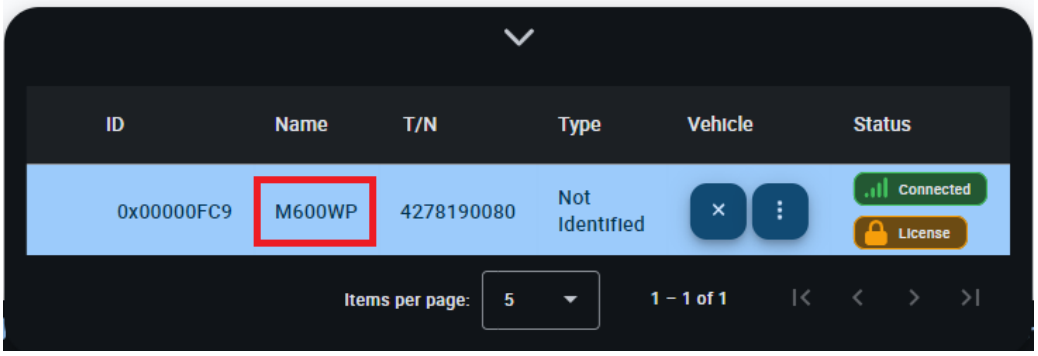
Morever, this menu allows to interact with the platform in Veronte Ops workspace through Platform actions.



Platform list

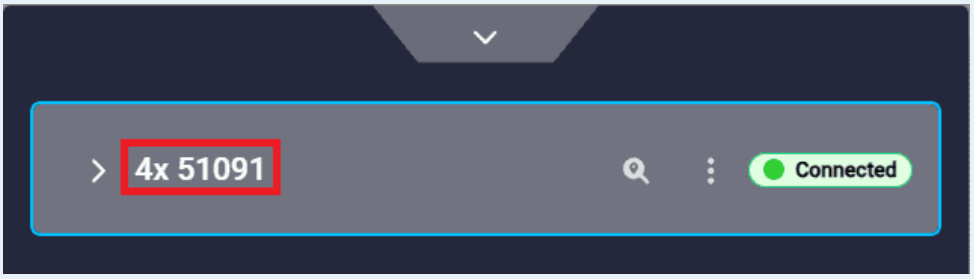
Platform name

The name of the unit configured in **1x PDI Builder** is displayed:



Platform name

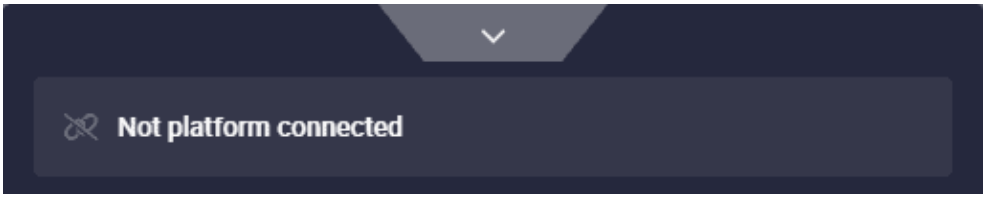
For more information on customizing the platform name, please refer to Unit name -Veronte section of **1x PDI Builder** manual.

> **Note**
>
> In the case of **Veronte Autopilot 4x**, the displayed name corresponds to the address configured in the **Control** menu of the **1x PDI Builder**. For further details, refer to 4x Veronte - Control section of **1x PDI Builder** manual.
>
> 
>
>
> **Platform name - Veronte Autopilot 4x**

Platform connection status

This menu allows the user to check the connection status of linked **autopilots**:

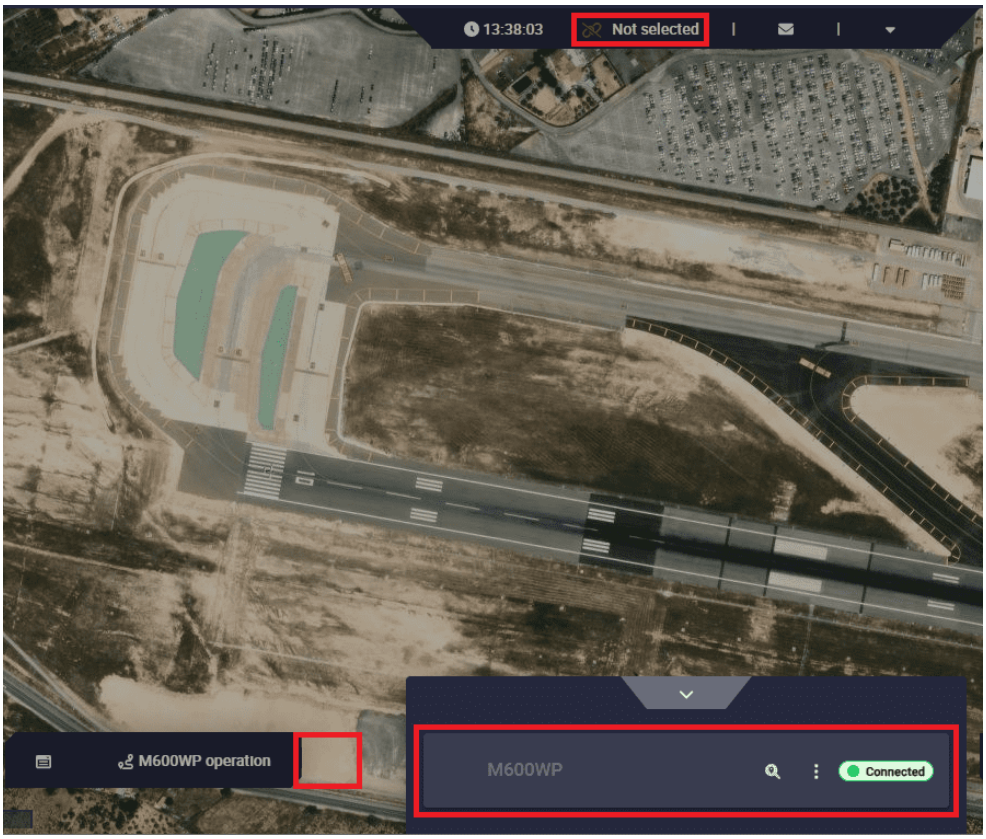
- **Not connected**

  Connection status - Not connected

- **Not selected**:

  Even if an **Autopilot 1x** is connected, until it is selected by the user, **Veronte Ops** will operate as if no autopilot is connected:



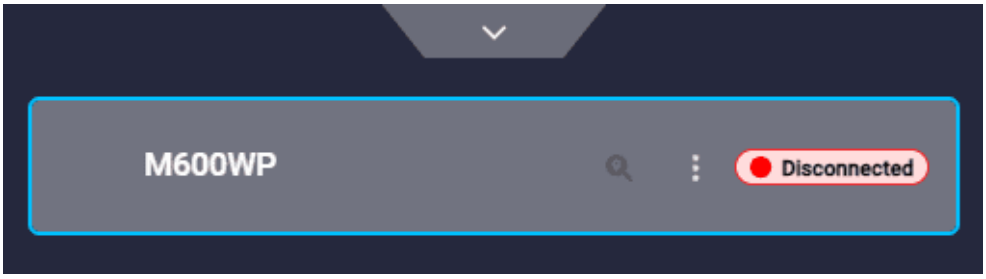
Connection status - Not selected

- **Connected**:

  Once Veronte Autopilot is **connected and selected**, **Veronte Ops** appears as shown in the figure below:

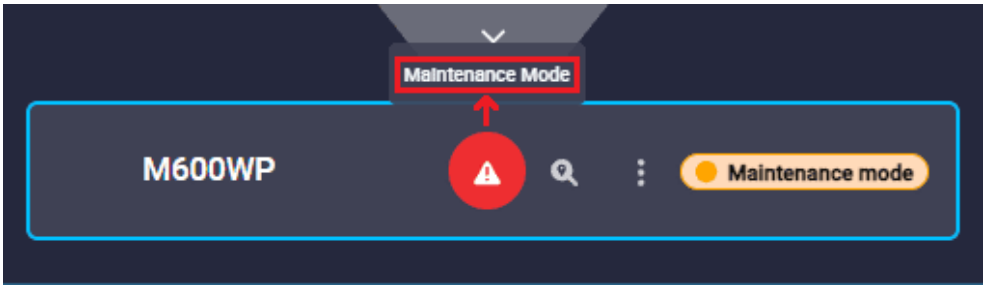
Connection status - Connected

- **Disconnected**:



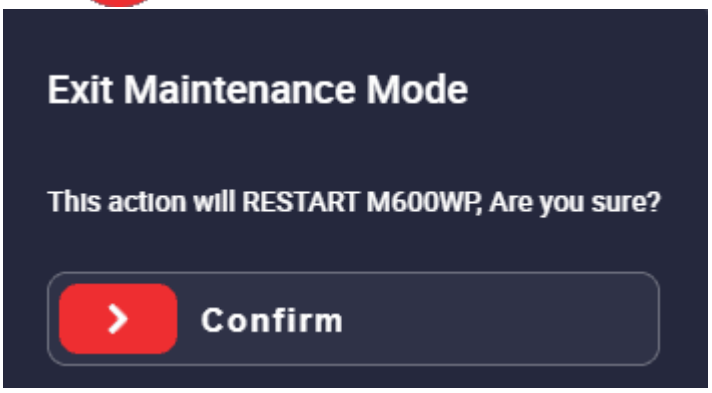
Connection status - Disconnected

- **Maintenance mode**: By **placing** the mouse cursor over the icon, a message appears indicating that the autopilot is in maintence mode:



Connection status - Maintenance mode

It is possible to exit of maintenance mode from this menu by **clicking** on the icon. Then, the following confimation message will appear:



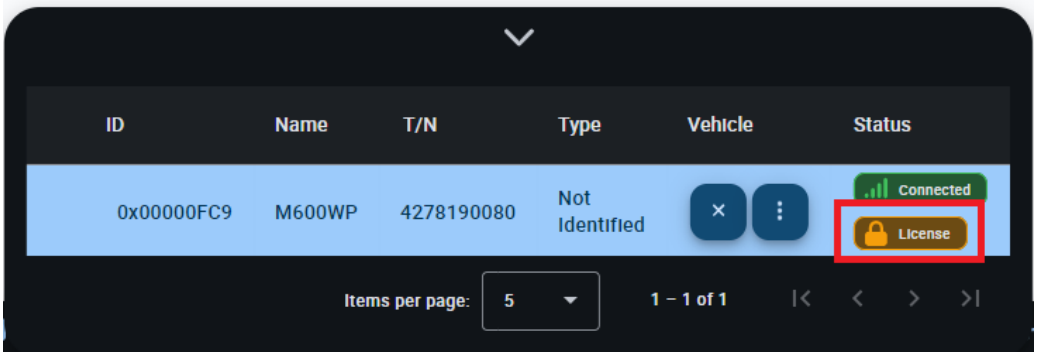
Connection status - Exit of maintenance mode

- **Maintenance mode (loaded with errors)**: In this case, by placing the mouse cursor over the icon, **Veronte Ops** will report the **ID of the PDI error and a brief description of the error**.

Connection status - Maintenance mode (loaded with errors)

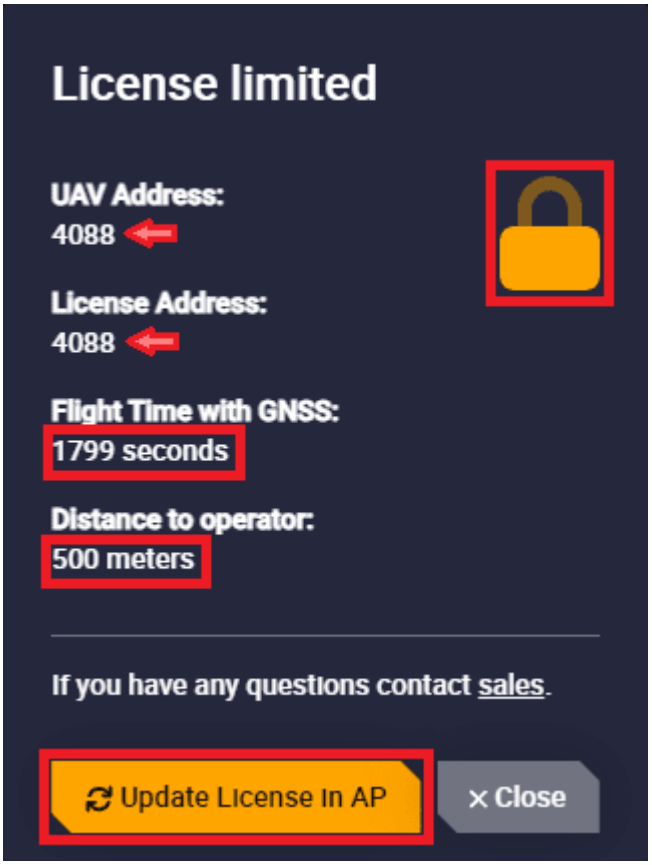
Platform license

The following indicator determines that the platform has a **limited operation license**. This means that the aircraft is submitted to several restrictions, such as limits on the area of operation.



Platform indicator - Limited operation license

By **clicking** on License, the following information panel will pop up summarizing the details of the license:

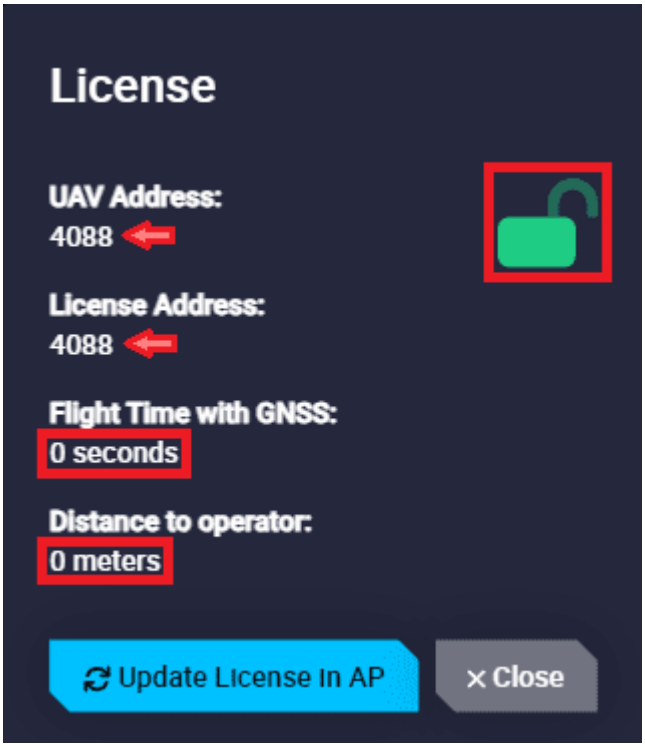


Platform list - Operation License limited message

Clicking on the button "**Update License in AP**", if the users have the **license activated**, **Veronte Ops** will be updated and they will be able to **operate without limitation**.

> ⓘ **Note**
>
> The License Address must match the UAV Address.

In this case, the information regarding the license changes and the padlock opens and turns green:



Platform list - Operation License updated message

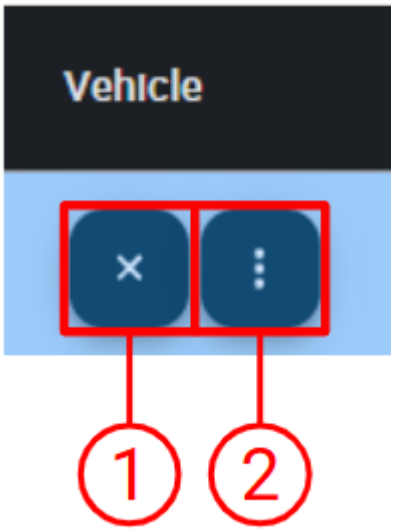
However, if users do **not have the license activated**, they will continue to have **limited operation**, in which case they should contact sales@embention.com.

For more information on this operation license, see Limited Operation section of the **1x Hardware Manual**.

> **Important**
>
> Users should take into account that if the platform is a PCS, it will not have an operation license. This is because PCS units do not need to fly.

Platform actions

Moreover, when in **Veronte Ops** the **Autopilot 1x is displayed in any connection state other than 'not connected'**, the following actions appear in this menu:



Platform list actions

1. **Select type of LOI**

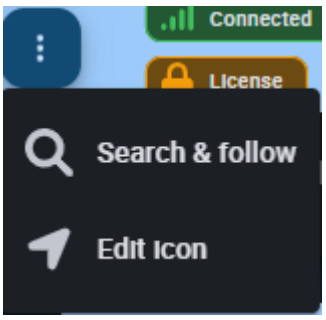
   The LOI menu defines the level of authority and control the operator holds over the UAV system during the mission.

   - **LOI2**: Allows telemetry monitoring and sensor data reception.
   - **LOI3**: Enables control of the payload and onboard sensors.
   - **LOI4**: Provides control over aircraft navigation and routing.
   - **LOI5**: Guarantees full control, including takeoff and landing phases.

     > ⚠ **Warning**
     >
     > **LOI 5** must be selected to enable Fly to loiter or Fly to hover.

   - **Release**: Releases any level of control previously acquired over the system.

2. **Options**
   - **Search & follow**: Centers the view on the aircraft and follows it during movement.
   - **Edit Icon**: Allows modifying the icon representing the aircraft on the map.

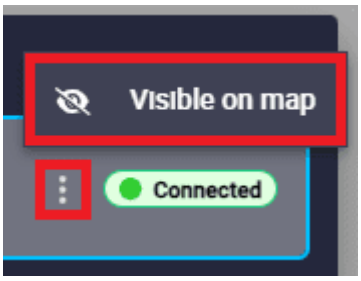


Platform Options

3. **Search & follow**: This action searchs the platform's position and centers it on the map.

> ⓘ **Note**
>
> This action is only **available** when the platform is **visible** on the map. In addition, it is **disabled** when **Autopilot 1x** is in '**disconnected**' state.

4. **Options**



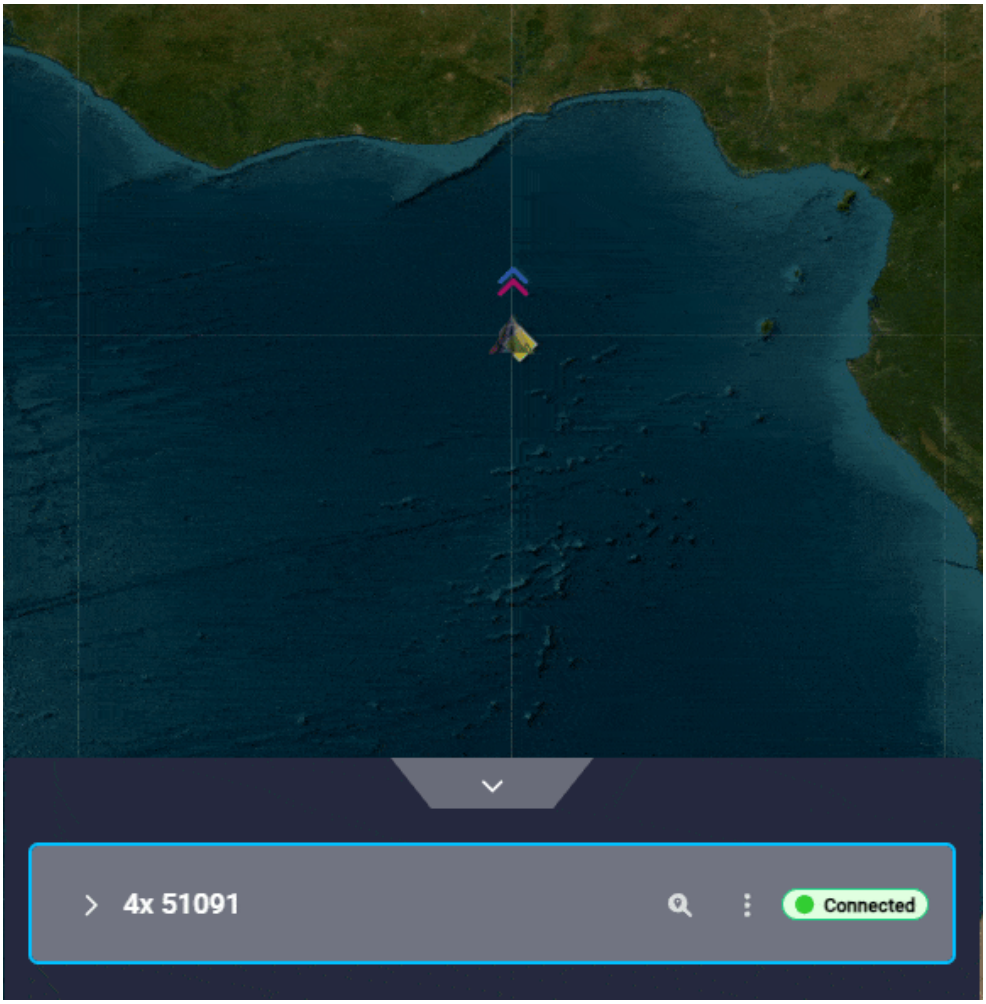
Platform list actions - Options

   - **Visible/Invisible on map**: When the icon is , the platform icon is visible on the map, when the icon is , the platform does not appear on the map.
5. **Minimize/maximize** this pill: By clicking here, this 'pill' will be minimized/maximized.

Autopilot 4x features

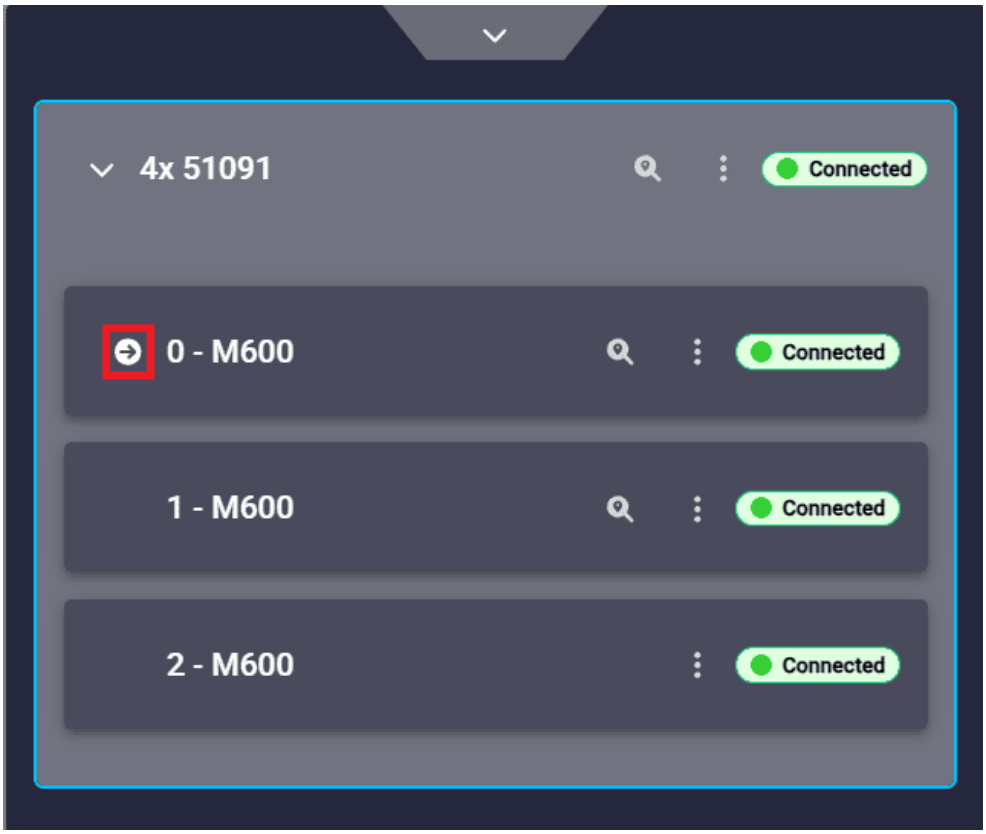
When an **Autopilot 4x** is linked to **Veronte Ops**, the following extra features are available:

- **Show Platforms** button: By expanding the 4x tab, each **Autopilot 1x** within the 4x group is displayed.



Autopilot 4x features - Show Platforms

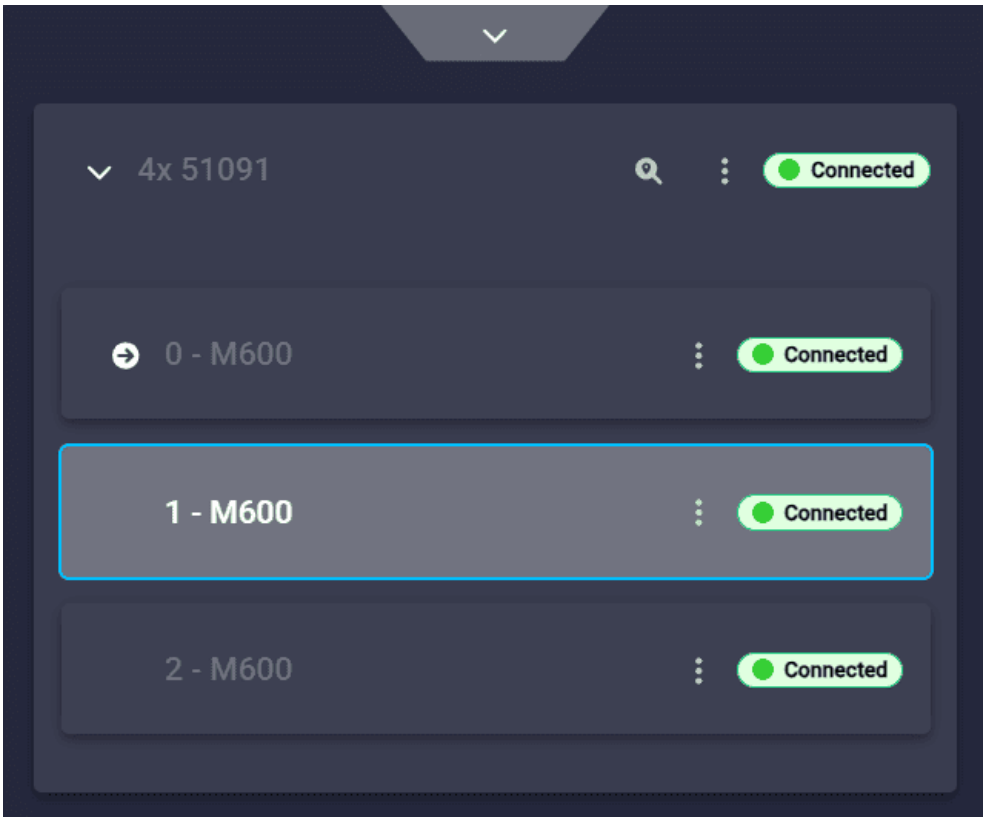
- **Autopilot 1x in command**: The **Autopilot 1x** selected **by the arbiter** is marked with the icon.



Autopilot 4x features - 1x in command

> **Important**
> '1x Selected by the arbiter' is not equivalent of '1x Selected in Veronte Ops'. For further information about **Autopilot 4x** general functioning, refer to Control diagram - Introduction section of the **4x Hardware Manual**.

Example:

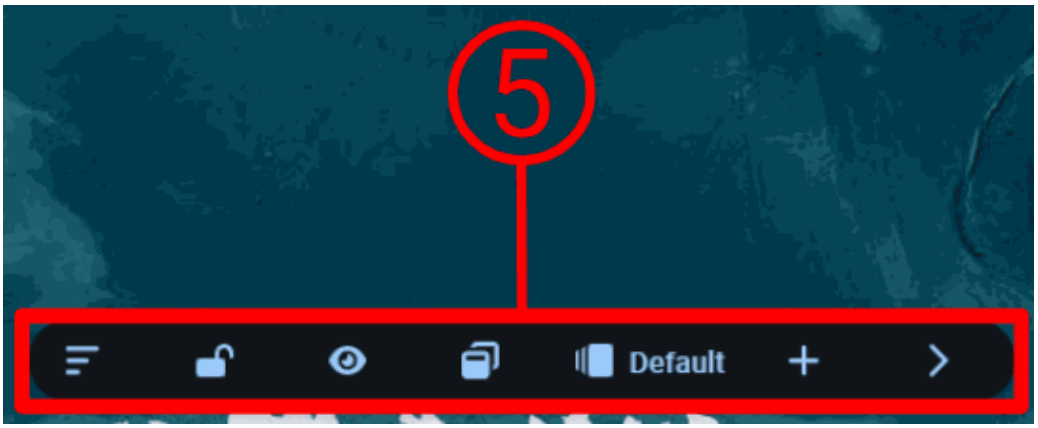


Autopilot 4x features - 1x Selected vs 1x in command

In the image above:

- `Autopilot 1x 0`: Autopilot **selected by the arbiter**, i.e. in command.
- `Autopilot 1x 1`: Autopilot **selected in Veronte Ops**.

5. Workspace



Workspace toolbar

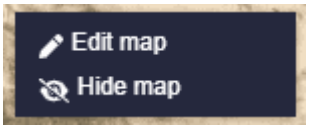
Workspace settings allow the user to customize any information to be displayed on the screen for monitoring the operation.

> **Attention**
>
> Although **Veronte Ops cannot be extended to multiple screens**, it is possible to **open more than one Veronte Ops** in order to display all the widgets necessary for the operation.

Map options

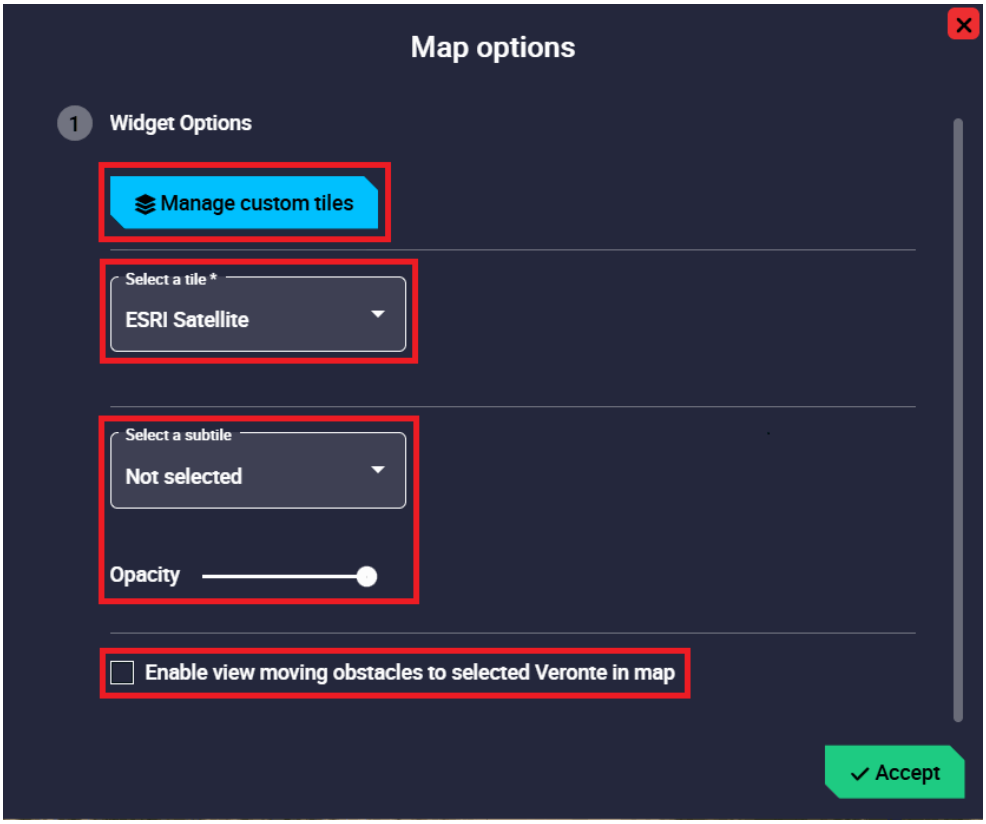
The map widget configures the background map that appears in **Veronte Ops** screen.

By **right clicking** on the map the following map options are available:



Map options

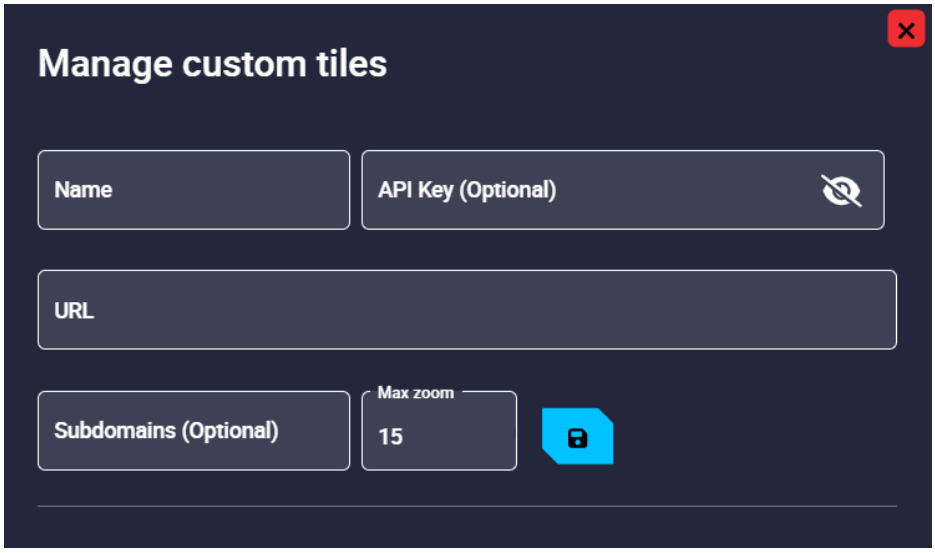
- **Edit map**: In this menu the following options can be configured:



Edit map options

  - **Manage custom tiles**: This menu allows the user to add and manage their own map tiles.

When a custom tile is added, it will appear as a new option under the "Custom tiles" category to select it either as a "tile" or as a "subtile".



Manage custom tiles

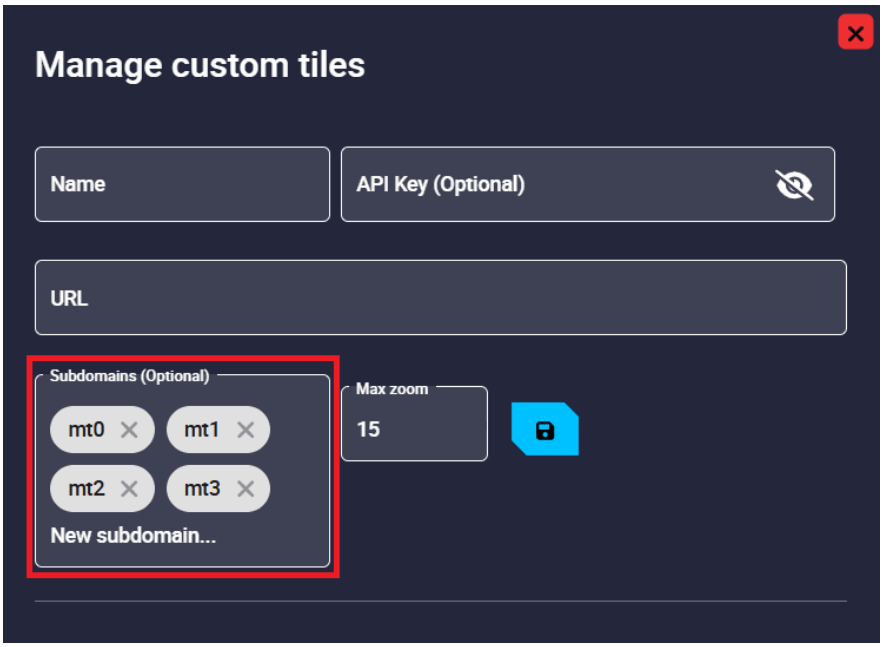
- **Name**: Users can customize the name of this tile.
- **API Key (Optional)**: Enter the API Key if needed (this depends on the map server).
- **URL**: Enter the URL of the map to be added. This URL must be of type "**Slippy map tilename**".

  The necessary "parts" of the URL are listed below with an example: `https://tile.waymarkedtrails.org/hiking/{z}/{x}/{y}.png`

  - The first part of the URL specifies the tile server ⇒ `tile.waymarkedtrails.org`
  - Tile coordinates are usually specified using the /zoom/x/y.png or /z/x/y.png tail ⇒ `/{z}/{x}/{y}.png`
  - Some tileservers will use a directory to specify a particular stylesheet ⇒ `/hiking/`

> ⚠ **Caution**
>
> If the added map requires an **API Key**, users must add as part of the URL the following: `{apiKey}` (it is important to respect the **lowercase and uppercase** of this). This is so that **Veronte Ops** correctly recognizes that the API Key entered above must be substituted in the URL.
>
> An example is shown below:
>
> `https://{s}.api.tiles.openaip.net/api/data/openaip/{z}/{x}/{y}.png?apiKey={apiKey}`

- **Subdomains (Optional)**: This is not needed, but can be entered. Multiple subdomains can be entered (as many as the server provides), they are added as pills as shown below.



Manage custom tiles - Subdomains

- **Select a tile**: It is possible to select the map provider from several options, including "Custom Tiles" if they have been created in the Manage custom tiles option.

  The URLs used by each of the available tile providers are listed in the following table:

> ⓘ **Note**
>
> These URLs are used internally by Veronte Ops.

Tile provider	URL
ESRI Satellite	`https://server.arcgisonline.com/ArcGIS/rest/services/World_Imagery/MapServer/tile/{z}/{y}/{x}`
CartoDB Voyager	`https://{s}.basemaps.cartocdn.com/rastertiles/voyager/{z}/{x}/{y}{r}.png`
CartoDB Dark Matter	`https://{s}.basemaps.cartocdn.com/dark_all/{z}/{x}/{y}{r}.png`
CartoDB Positron	`https://{s}.basemaps.cartocdn.com/light_all/{z}/{x}/{y}{r}.png`
Open Topo Map	`https://{s}.tile.opentopomap.org/{z}/{x}/{y}.png`
Open Street Map	`https://{s}.tile.osm.org/{z}/{x}/{y}.png`
Google Street	`https://{s}.google.com/vt/lyrs=m&x={x}&y={y}&z={z}`
Google Hybrid	`https://{s}.google.com/vt/lyrs=s,h&x={x}&y={y}&z={z}`
Google Satellite	`https://{s}.google.com/vt/lyrs=s&x={x}&y={y}&z={z}`
Google Terrain	`https://{s}.google.com/vt/lyrs=p&x={x}&y={y}&z={z}`
TianDiTu Normal (CN)	`https://t{s}.tianditu.gov.cn/DataServer?T=vec_w&X={x}&Y={y}&L={z}&tk={apiKey}`

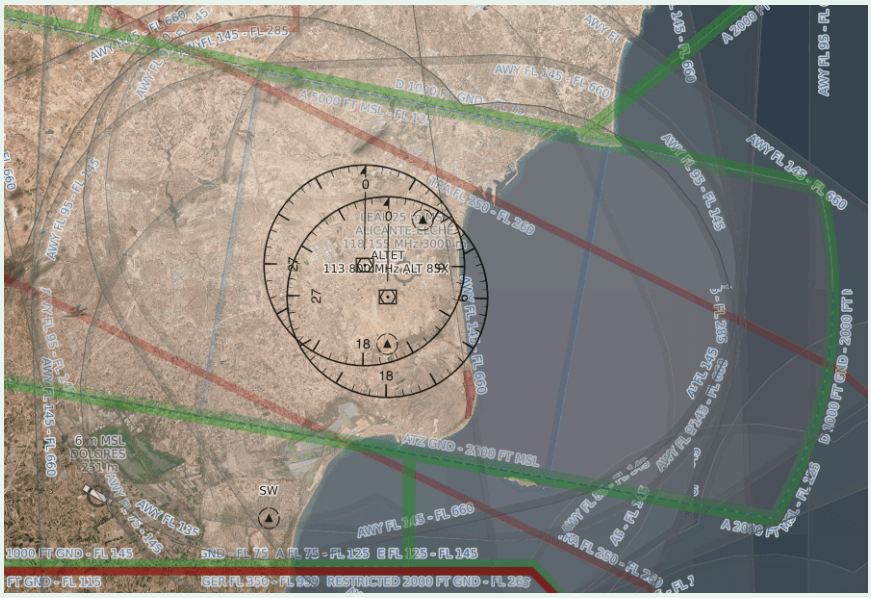
Tile provider	URL
TianDiTu Satellite (CN)	`https://t{s}.tianditu.gov.cn/DataServer?T=img_w&X={x}&Y={y}&L={z}&tk={apiKey}`
TianDiTu Terrain (CN)	`https://t{s}.tianditu.gov.cn/DataServer?T=ter_w&X={x}&Y={y}&L={z}&tk={apiKey}`
Geoq Normal (CN)	`https://map.geoq.cn/ArcGIS/rest/services/ChinaOnlineCommunity/MapServer/tile/{z}/{y}/{x}`
Geoq PurplishBlue (CN)	`https://map.geoq.cn/ArcGIS/rest/services/ChinaOnlineStreetPurplishBlue/MapServer/tile/{z}/{y}/{x}`
Geoq Gray (CN)	`https://map.geoq.cn/ArcGIS/rest/services/ChinaOnlineStreetGray/MapServer/tile/{z}/{y}/{x}`
Geoq Warm (CN)	`https://map.geoq.cn/ArcGIS/rest/services/ChinaOnlineStreetWarm/MapServer/tile/{z}/{y}/{x}`
Geoq Hydro (CN)	`https://thematic.geoq.cn/arcgis/rest/services/ThematicMaps/WorldHydroMap/MapServer/tile/{z}/{y}/{x}`
GaoDe Normal (CN)	`https://webrd0{s}.is.autonavi.com/appmaptile?lang=zh_cn&size=1&scale=1&style=8&x={x}&y={y}&z={z}`
GaoDe Satellite (CN)	`https://webrd0{s}.is.autonavi.com/appmaptile?style=6&x={x}&y={y}&z={z}`

> ⚠ **Warning**
> If a **firewall** is being used, the URL of the tile provider to be used must be added to the **whitelist**. Otherwise, the map tiles cannot be downloaded and the map will not be displayed.

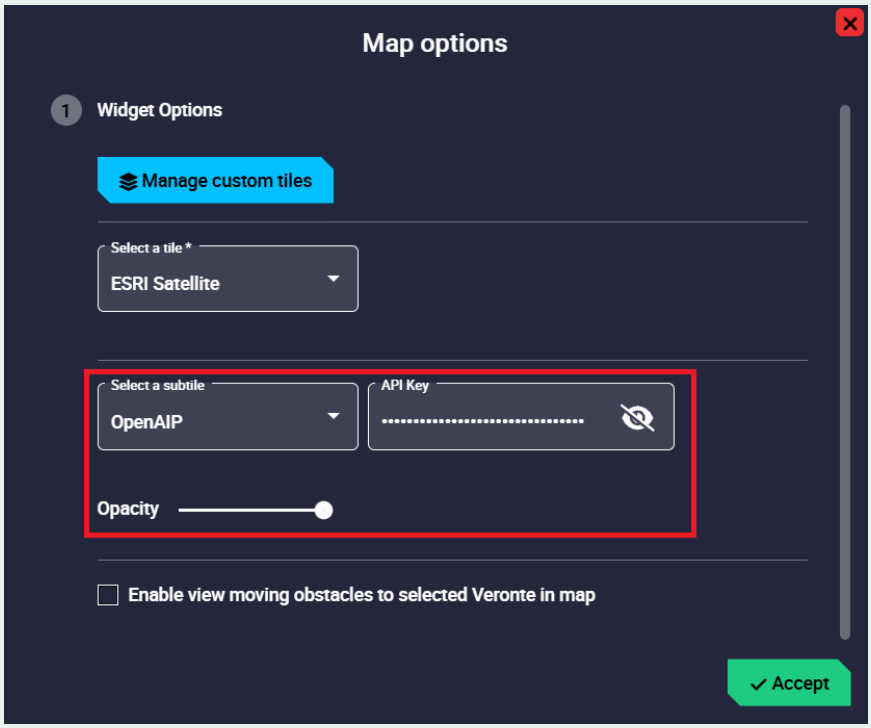
- **Select a subtile**: Users can add a second map. The available options are:
  - Not selected: This is the default selected option.
  - OpenAIP.
  - GaoDe Annotion.
  - **Custom tiles**: Only appears when a custom tile has been created in the Manage custom tiles option.
- **Opacity**: The user can set the opacity of this subtile. It is expressed as a percentage of 1 (i.e. the minimum is 0 and the maximum is 1).

Tip

This allows the user to have one map overlayed on top of another. For example, it can be used to have as **tile** the main map and as a **subtile** a map with airspace information (OpenAIP map):



Example of map with subtile



Example of map with subtile configuration

- **Enable view moving obstacles to selected Veronte in map**: If enabled, moving objects configured in the **1x PDI Builder** software will be displayed on the map. In addition, if the autopilot possesses an

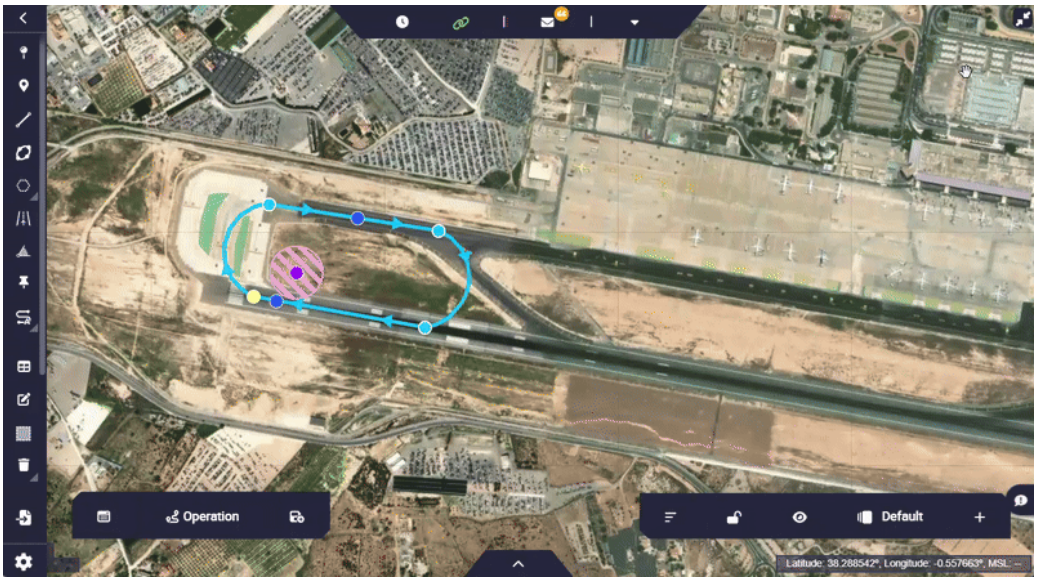
ADS-B system and it is enabled, Veronte Ops will show aircraft detected by it on the map as moving objects.

Moving objects - ADS-B activated

- **Hide map**: The map will be hidden. To show it again, go to '**Main Widgets' menu** and select 'Map'.

In addition, **double-clicking** on the map will **zoom in** on the area the user is clicking on.

Furthermore, the map can be minimized/maximized by clicking on the icon in the top right corner of the application. An example is shown below:



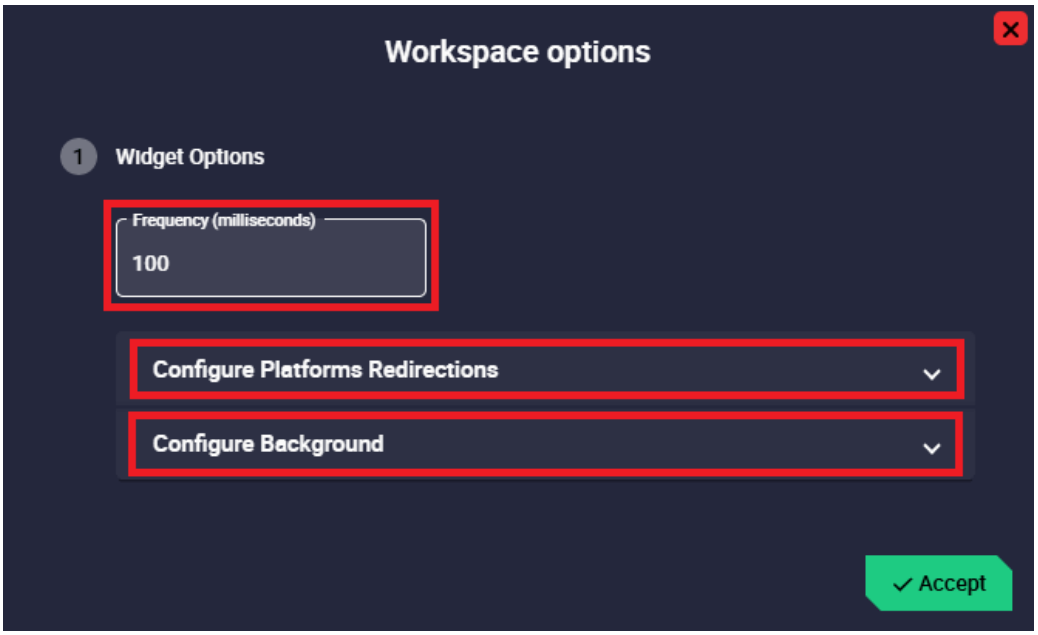
Minimize/Maximize map

Workspace options

It is possible to edit the background behind the map. To access this editing menu it is necessary to first **minimize the map** (or hide it) and then **right click on the background**:

Access background edition

In this menu, users will find:

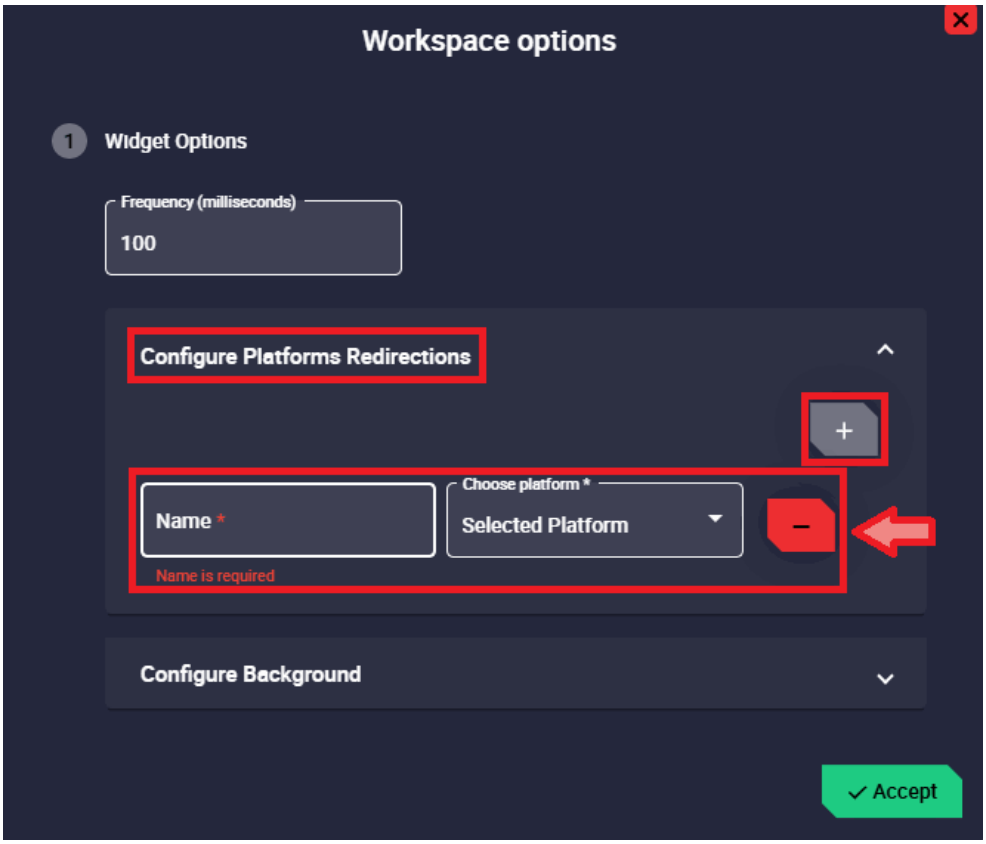


Background edition

> **Tip**
>
> By right-clicking on the button, it is possible to open the Workspace options.

- **Widget options**: The user can change the refresh frequency of the workspace, of the widgets. By default this frequency is **100 miliseconds**.
- **Configure Platforms Redirections**: Users can create a redirection to link a platform to a custom "role".

> **Hint**
>
> A common approach is to create two roles: "Air" and "Ground", and assign the desired widgets to the corresponding role instead of linking them directly to a specific platform. This allows users to easily switch the platform assigned to the Air/Ground role from this menu if a different Autopilot 1x unit is used, without needing to update each widget individually.

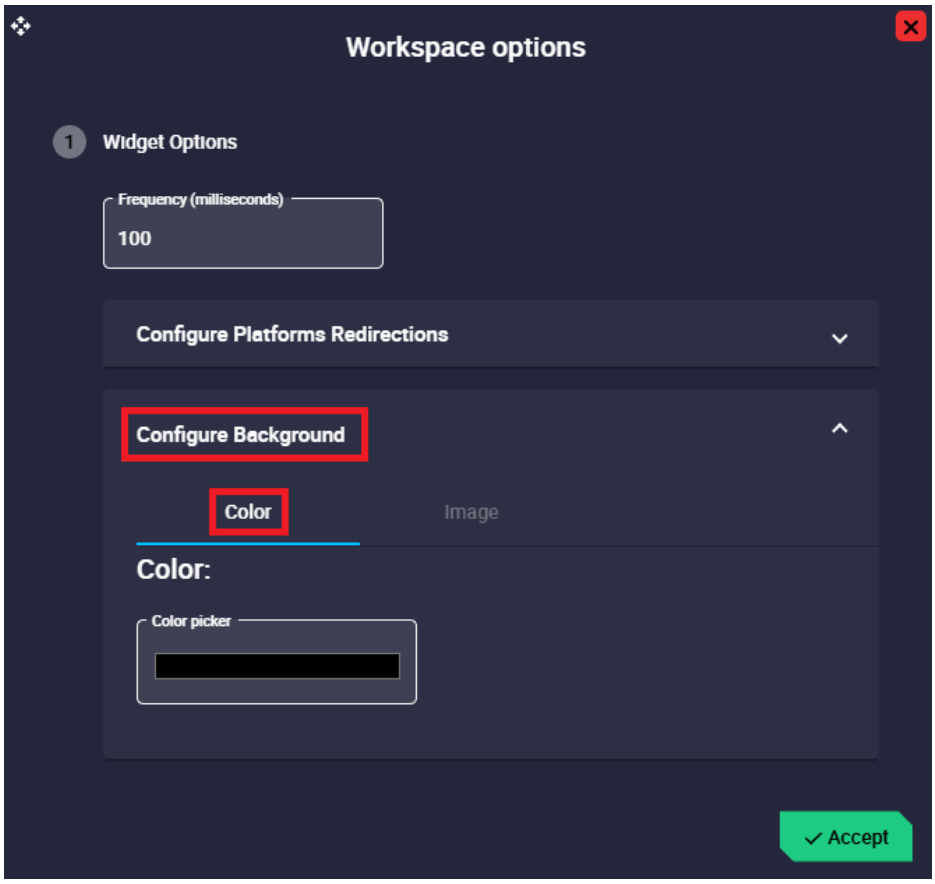


Background - Platforms Redirections

Click on "+" to add a new redirection:

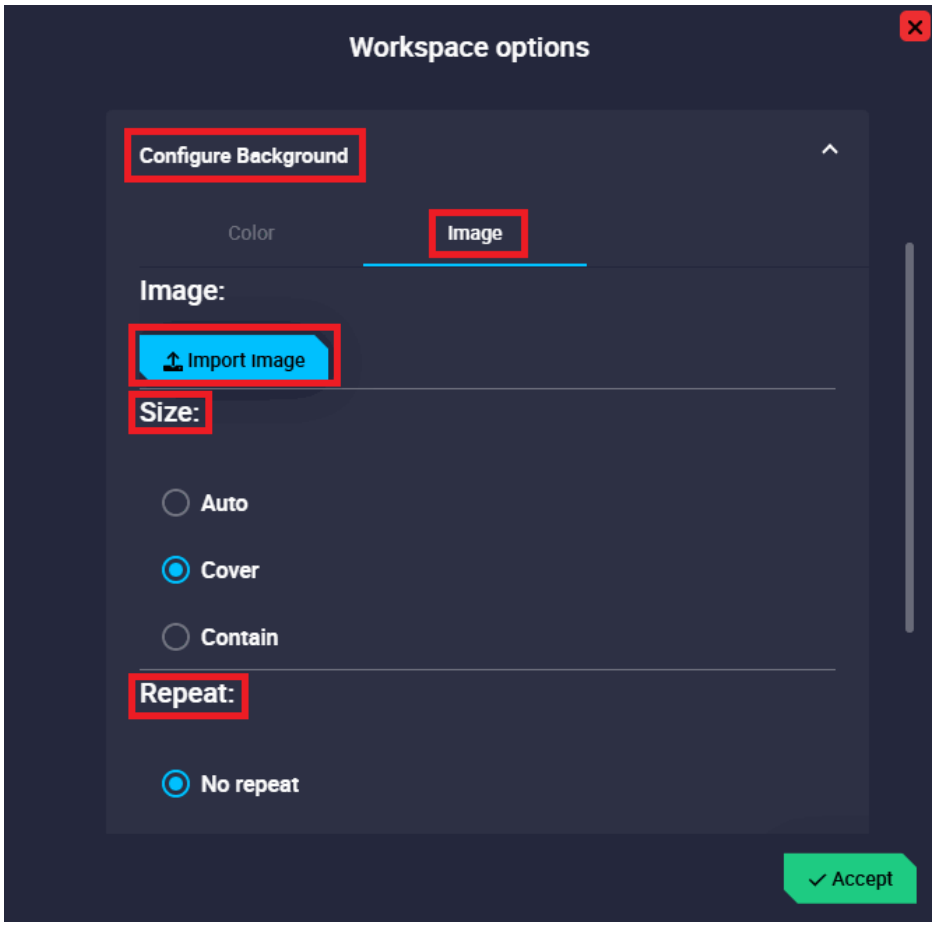
- **Name**: Enter the desired name for the rule.
- **Choose platform**: Users must select the platform for which the redirection is configured. The available options will always be the IDs of the connected Autopilots 1x and '**Selected platform**', i.e. the platform that is selected.
- Delete button

- **Configure Background**: Choose the type of background style between a color or an image:
  - **Color**: Users can select the desired color for the background.



Background - Configure color

- **Image**: When choosing 'Image', users must configure some parameters:

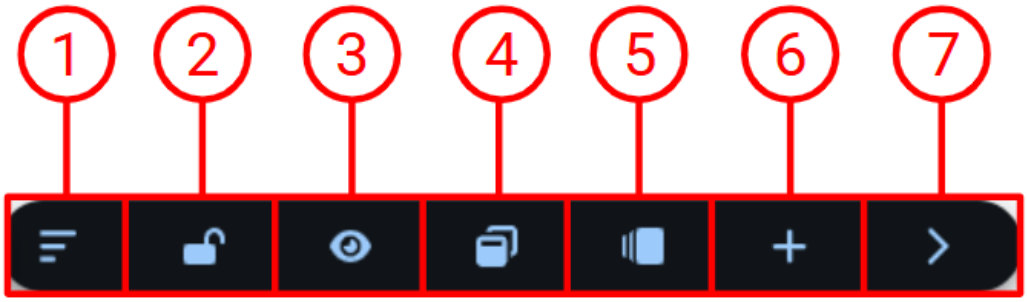


Background - Configure image

- **Image**: A **Veronte Ops** image is set by default, but it is possible to import another image from the local PC.
- **Size**: The size of the background can be modified, the available options are: Auto, Cover and Contain.
- **Repeat**: The repetition of the image can also be customized, choose from: No repeat, Repeat, Repeat X, Repeat Y, Space and Round.

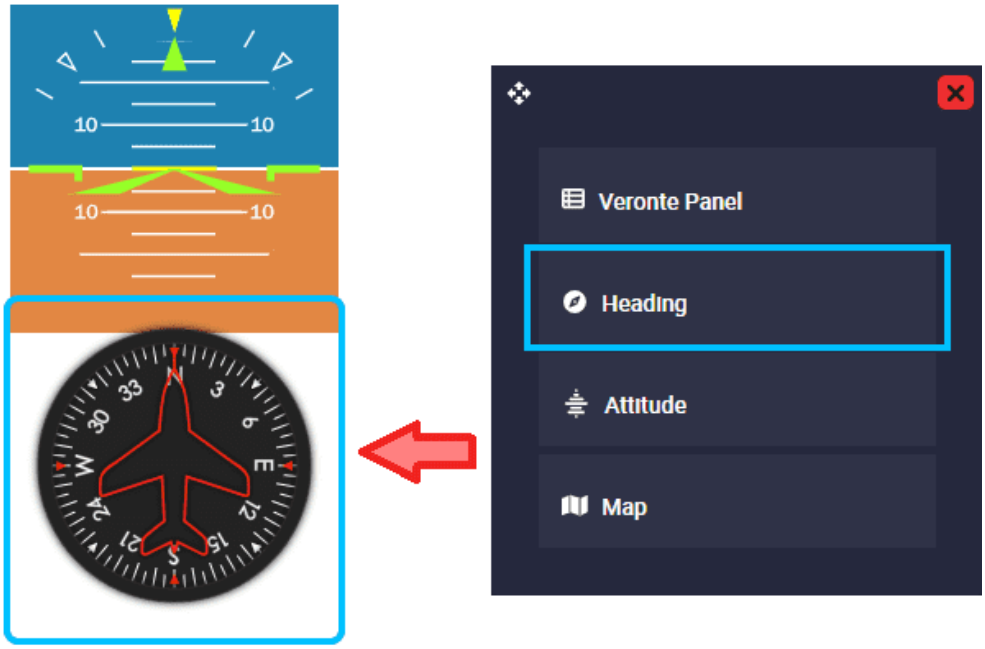
Workspace toolbar

The workspace toolbar is divided into 5 different 'parts':



Workspace toolbar parts

1. **Sort widgets**: With this button it is possible to sort the widgets, i.e. the user can place one widget on top of another.
   To do this, click on this button and a 'sorting panel' will appear for the user to sort widgets, so that the first in the list will be on top of the second, the second on top of the third and so on. For ease of use, the widget selected for sorting will be outlined in blue, as shown in the example below:

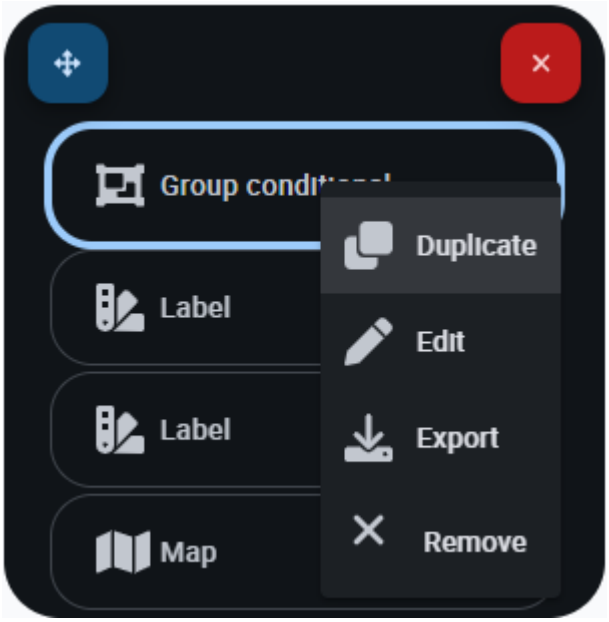


Sort widgets

In this example, 'Heading widget' is selected and is ordered above the 'Attitude widget'.

> ⚠ **Warning**
> The **map** will always be at the **back** (the last widget in the list), except when minimized, which can be sorted as desired.

In addition, by **right-clicking** on each widget from this sorting panel it is possible **to access its options** (note that groups in the sorting panel also behave like widgets):



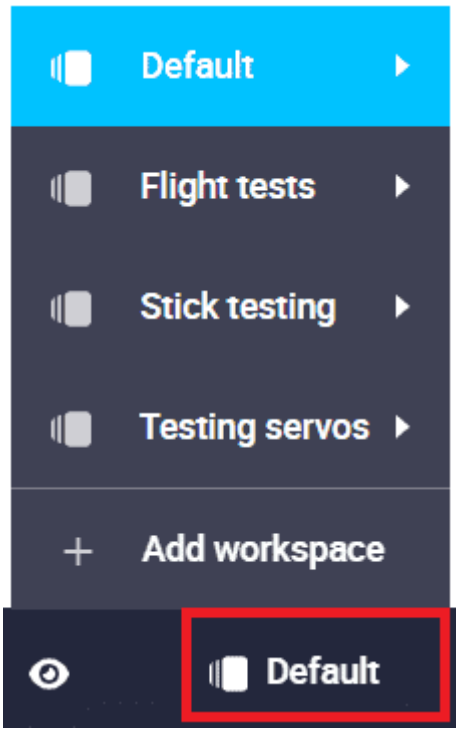
Sort widgets options

- **Duplicate**: Duplicates this widget.
- **Edit**: Accesses the edit configuration menu of each widget. For more information on the configuration of each widget, please refer to the corresponding widget section of this manual.
- **Export**: Saves the element.
- **Remove**: Removes this widget.

2. **Lock/Unlock widgets**: When widgets are locked, users will not be able to move them.
3. **Show/Hide widgets**: Users can choose to show/hide **all widgets** by pressing this button.
4. **New window**: User can create another window to view the operation.
5. Workspaces: **Veronte Ops** allows users to create different workspaces. This is useful to change the displayed information/widgets depending on the purpose of the operation, i.e., the widgets desired to be displayed during a simulation or flight test may not be the same as the widgets desired to be displayed for the final mission.
6. Widgets: **Veronte Ops** offers several applications to display parameters and variables in real time of the flight mission.
7. Close **Worspace** bar

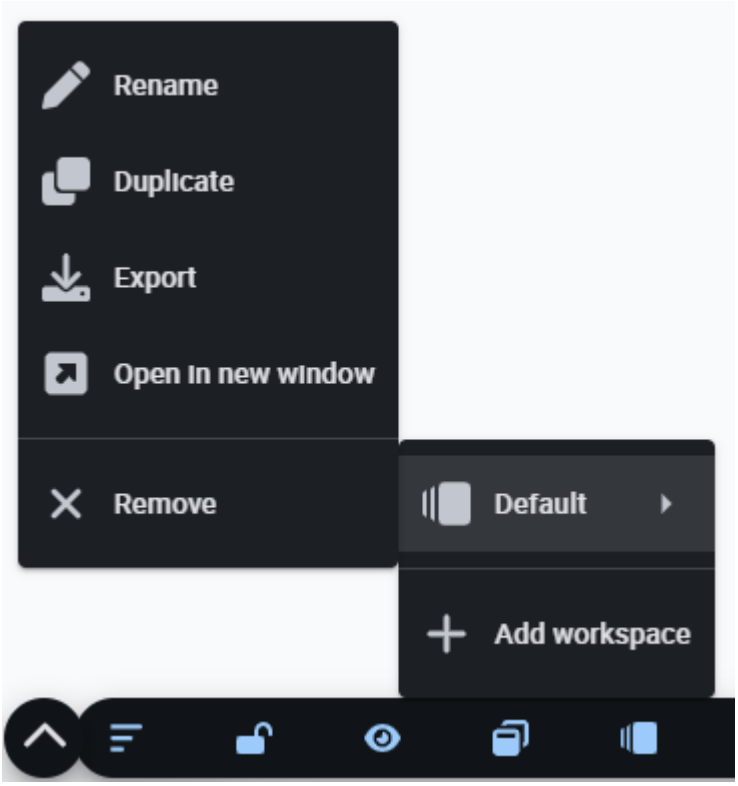
Workspaces

By clicking here, a list of all workspaces loaded in **Veronte Ops** will appear. The current workspace is the one that is selected, the one shown in blue.



Workspaces

The following options are available for each workspace:



Workspaces options

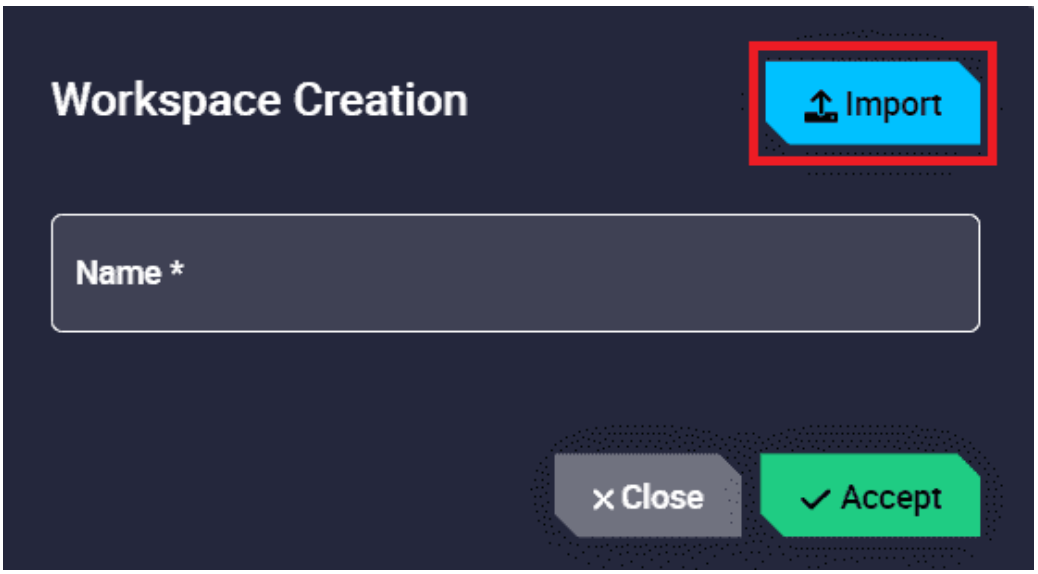
- **Rename**: The user can rename the workspace as desire.
- **Duplicate**: Duplicates this workspace.
- **Export**: The current workspace can be exported. This file is exported in `.vws` format.
- **Open in new window**: User can open the desired workspace in a different window.
- **Remove**: Clicking here will delete the selected workspace.

> ⓘ **Note**
> It is possible to delete several or all workspaces at once from the **Workspace manager menu** of the **Status bar**, click here for more information.

Adding a workspace

It is posible to create a new workspace by simply clicking on '**Add workspace**'.

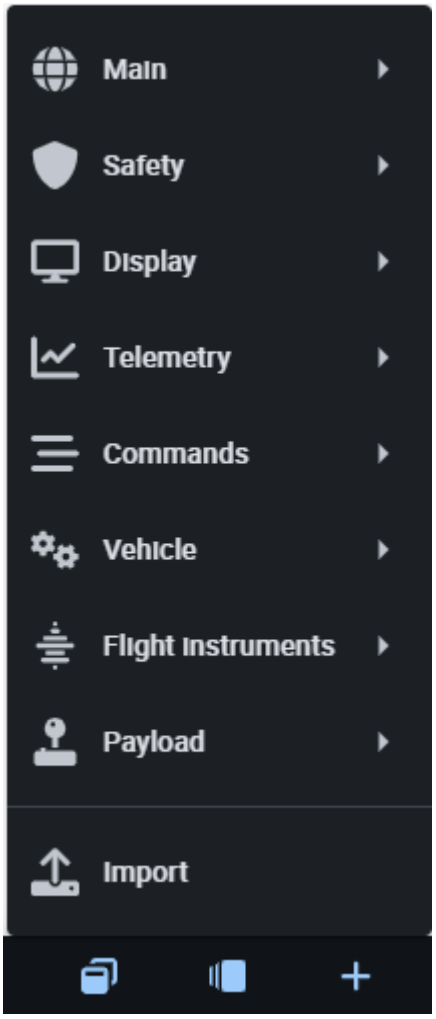
The user can choose between adding an empty workspace or importing one from the laptop, which has been previously exported. To **import** a workspace, the format file must be a `.vws`.



Add workspace

Widgets

By clicking here, the user will see different widget menus sorted by categories.



Widgets

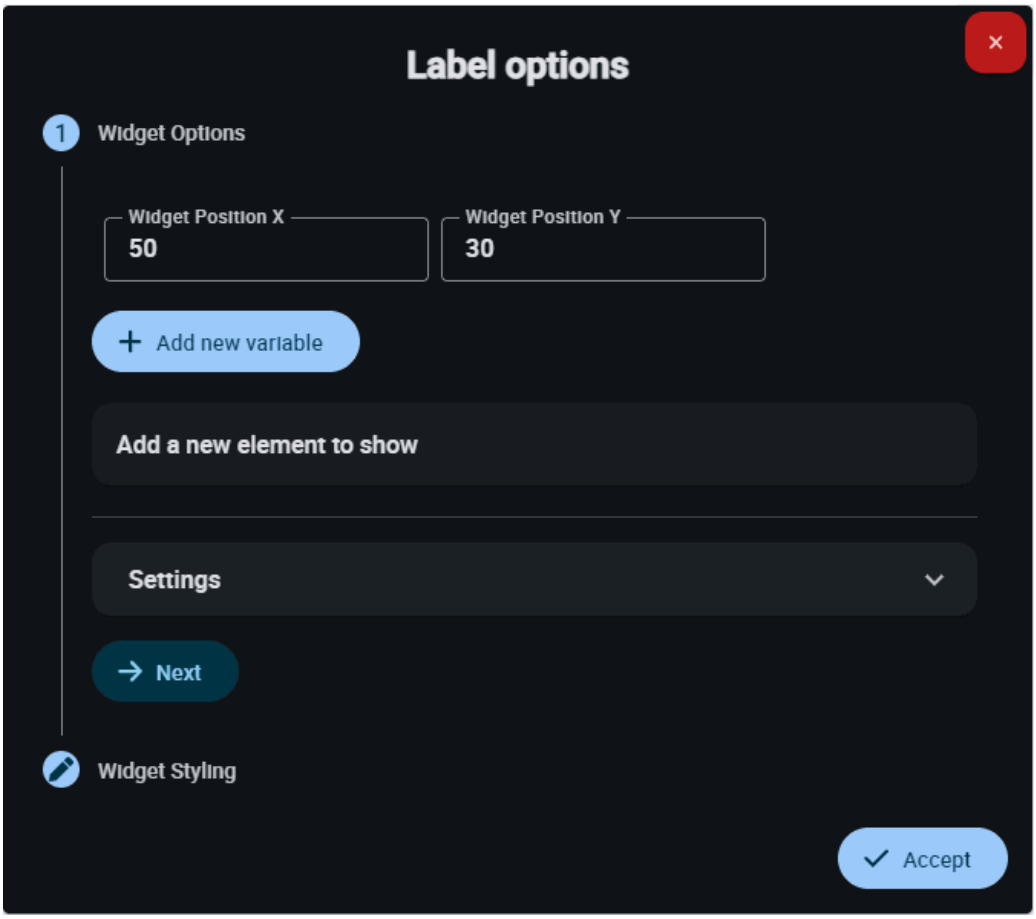
To display any widget on the map, just search for the desired widget and configure it.

> ⚠ **Warning**
>
> If the variable configured for a widget is not present in the telemetry configuration of the PDI, the widget will appear with a red triangle (warning icon), indicating that the selected variable is not being received in **Veronte Ops**.
>
>
>
>
> **Not receiving telemetry in this widget**
>
> Therefore, the variable must be added to the Complemetary Telemetry.

Widgets common configuration

> ⚠ **Warning**
>
> This does not apply to widgets in the '**Statics**' category.

All widget editing menus except those in the 'Statics' category have **at least** the following configurable parameters:



Edit menu - Widget Options

- **Widget Position X / Y**: Allows entering exact numerical coordinates to determine where the label appears on the screen.
- **Add new variable**: Button to link telemetric data or specific system variables for monitoring.
- **Add a new element to show**: Area dedicated to adding new graphic or textual components within the same widget.

Settings: Drop-down menu to access advanced widget settings.

Edit menu - Widget Stylying

Here the user can choose the style of the widget:

1. **Select style**: Default and custom styles can be selected. The available default styles are Default_box, Default_box_transparent, Default_box_blur, Default_box_black, Default_pill and Default_pill_blur.

   > ⓘ **Note**
   > Not all these default styles are available in all widgets.

2. **Add new style**:A new style can be created from the currently selected style by clicking here.
3. **Import style**: Users can import a custom style from local storage. It must be a `.css` file.
4. **Export style**: By clicking here, the **selected custom style** will be exported in a zip folder. More information about the contents of this folder is described in the Style manager - Veronte Ops configuration section of this manual.

   > **Important**
   > This option is only **available** when a **custom style is selected**.

5. **Edit style**: Users can directly edit the configuration (code in `.css` format) of the selected style by clicking here.

> **Important**
> This option is only **available** when a **custom style is selected**.

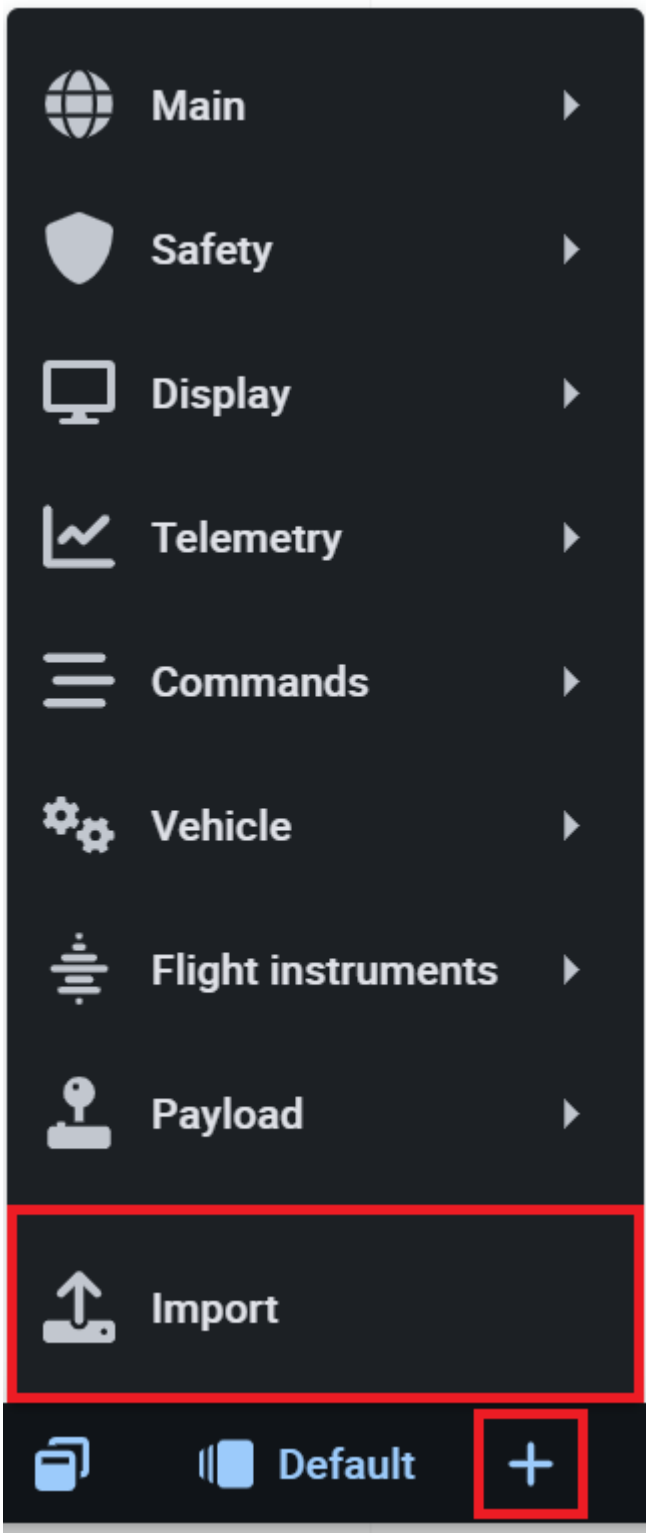
6. **Remove style**: The selected style will be deleted.

> **Important**
> This option is only **available** when a **custom style is selected**.

All widgets are explained in detail in the following sections:

- Main
- Safety
- Display
- Telemetry
- Commands
- Vehicle
- Flight instruments
- Payload

The Import feature enables the user to import a single widget or a widget group.



Widget - Import

Main

Map

This widget corresponds to the map and **only appears** in this menu when the map has been **hidden** by the user.

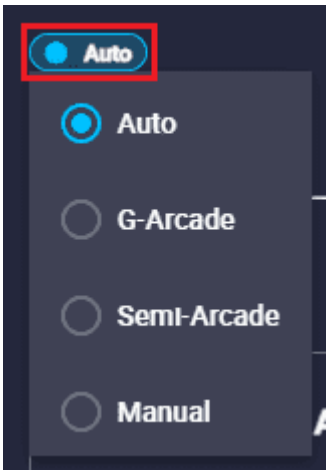
Veronte Panel

This pannel is the **basic operator tool**. It includes all basic commands (phase and action buttons) and information needed during a standard mission. These commands can be triggered with a single click, by sliding and/or automatically.

Veronte Panel

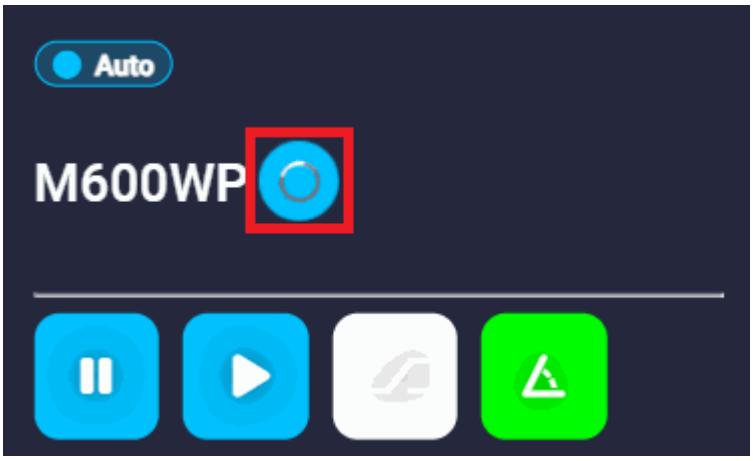
1. **Flight mode**: Displays the currently selected flight mode in which the user is operating.

   By clicking on it, it is possible to **manually** change the selected mode. The flight modes available here must be previously defined in the Modes - Control section of the **1x PDI Builder** software.



Veronte Panel - Flight modes

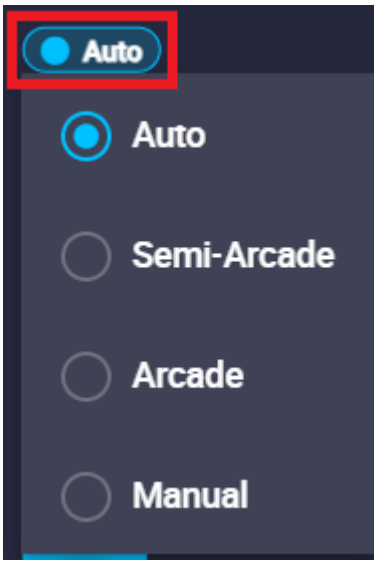
When in manual mode, the lettering of the icon will be **red**, to warn the operator to be careful in this mode.



Veronte Panel - Manual flight mode

Besides, the mode can also be changed automatically with an automation or manually with a configured stick.

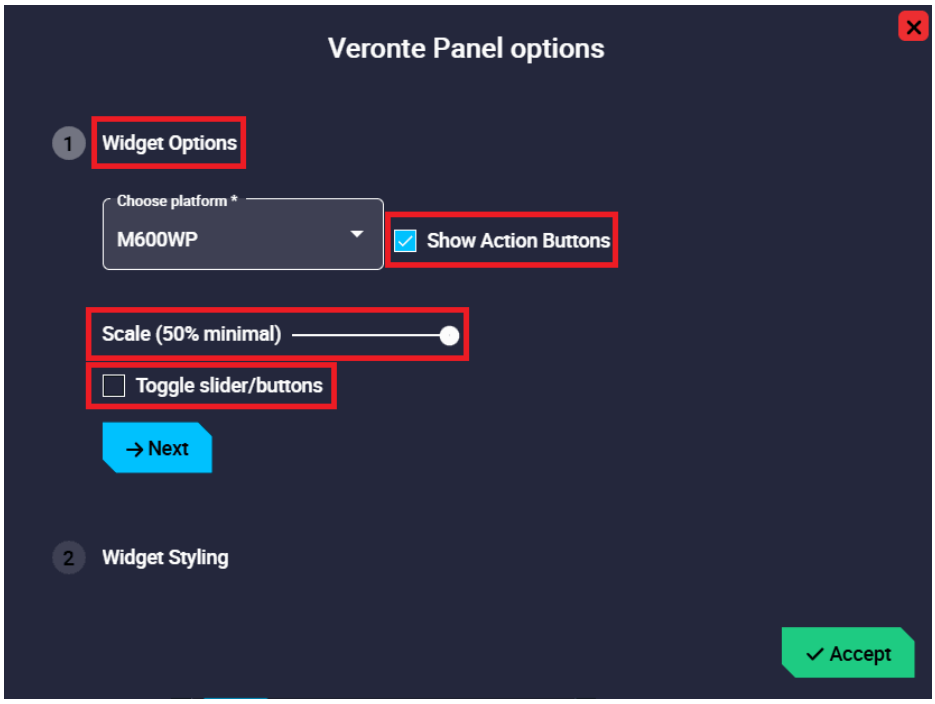
2. **Platform name**: The name of the selected platform is shown here.
3. **Options**: By clicking here, the following options are available:



Veronte Panel options

- **Edit**: This allows the user to access the Veronte panel configuration menu.

  This widget has **two extra edit parameter** compared to the ones described in Widgets common configuration:

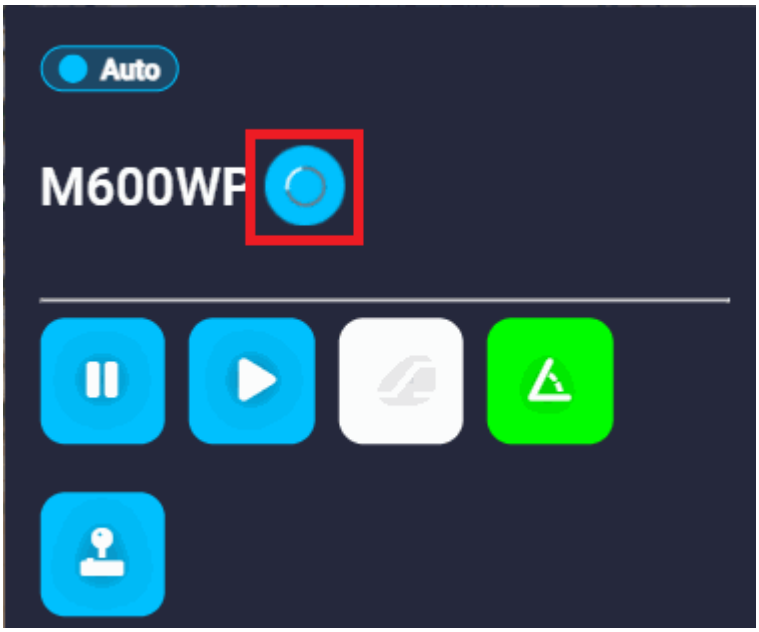


Veronte Panel configuration

- **Widget Options**
  - **Show Action Buttons**: When enabled, all the Action buttons are displayed in the Veronte panel.
  - **Scale (50% minimal)**: Veronte panel widget can be scaled from 50% to 100% of its size. By default it is set to 100%.

        - **Toggle slider/buttons**: Allows to choose whether to display the widget controls via a sliding bar or through pressable buttons.
    - **Remove**: Deletes this widget.

4. **FastLog**: By **pressing this button**, the **Veronte Autopilot 1x** will **start recording the fastlog** and this option will be as shown in the figure below:



Veronte Panel - FastLog recording

To **stop fastlog recording**, simply **click this button again**.

For more information on Fast Log, see the Fast Log - Telemetry section of the **1x PDI Builder** user manual.

> **Important**
> This button only appears when **Fast Log** has been configured in the **1x PDI Builder** software.

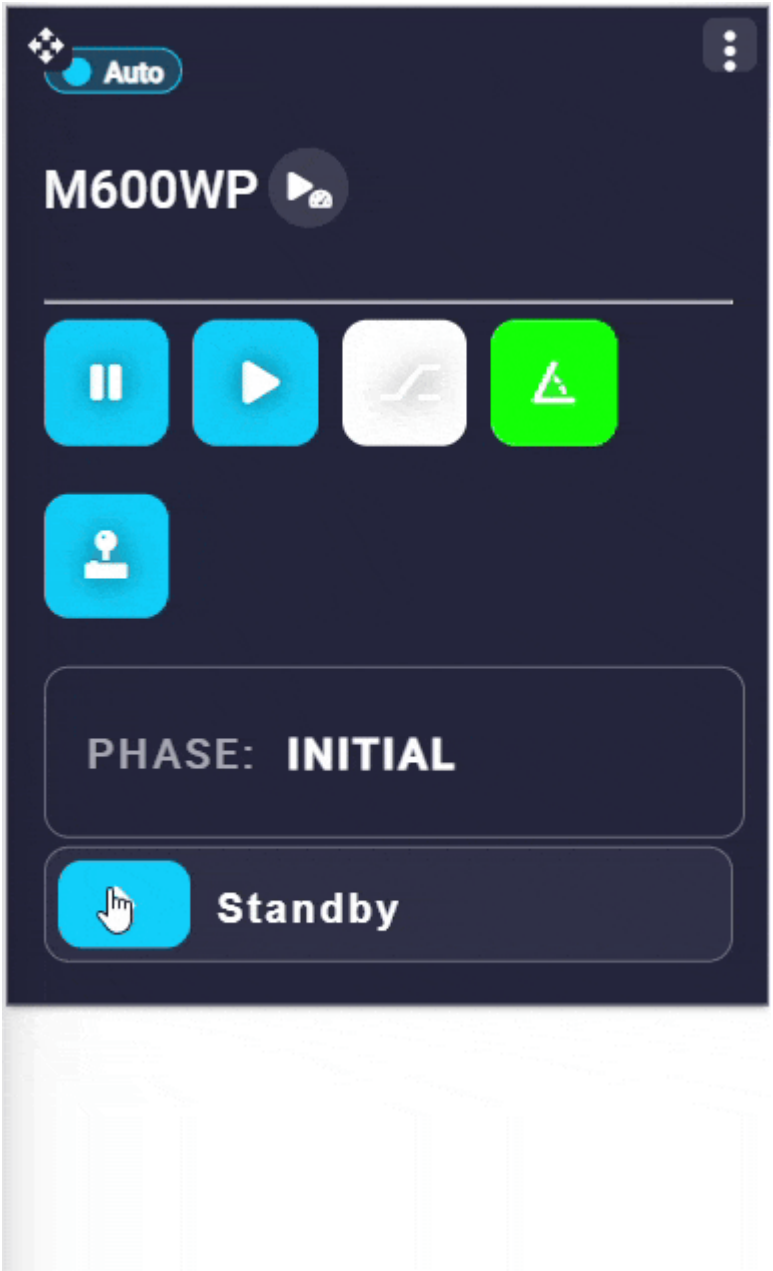
5. **PDI Mode**: This **only** appers if **PDI Mode** is **configured** in the **1x PDI Builder** software.
   For more information on PDI Mode, check the PDI Mode description of the **1x PDI Builder** user manual.
6. **Action buttons**: Action buttons can be used to manually trigger certain actions. They must have been previously defined in the Automations menu of the **1x PDI Builder** software.

> **Note**
> For further details about **Action buttons**, refer to Action Button - Inputs section of the present manual.

7. **Phase buttons**: The currently selected phase is always displayed at the top in the current phase identifier.

> **Important**
> Not all existing flight phases are initially displayed. Only flight phases that can be entered from the currently selected phase are displayed.

To change phase, simply slide the icon to the right. Then, if everything is correct, the phase will be displayed in the current phase indentifier and will be colored green while switching to that phase. An example is shown below:



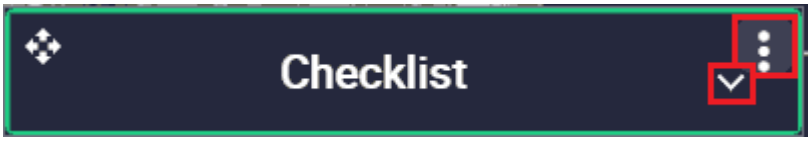
Veronte Panel - Changing phase

> ⚠ **Warning**
> An **error** may occur when changing phase, so that the phase change does not take place. To solve this problem, please check the Error when changing phase - Troubleshooting section of this manual.

Checklist

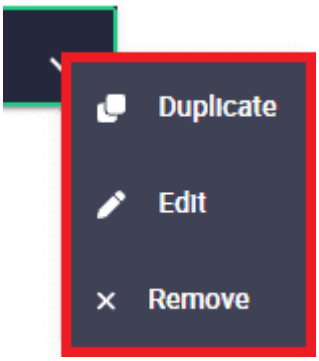
This panel is used to make sure that some requirements have been accomplished, for example, prior to a phase change or to avoid a possible malfunction.

It must be previously defined in the Checklist - Safety section of the **1x PDI Builder** software.



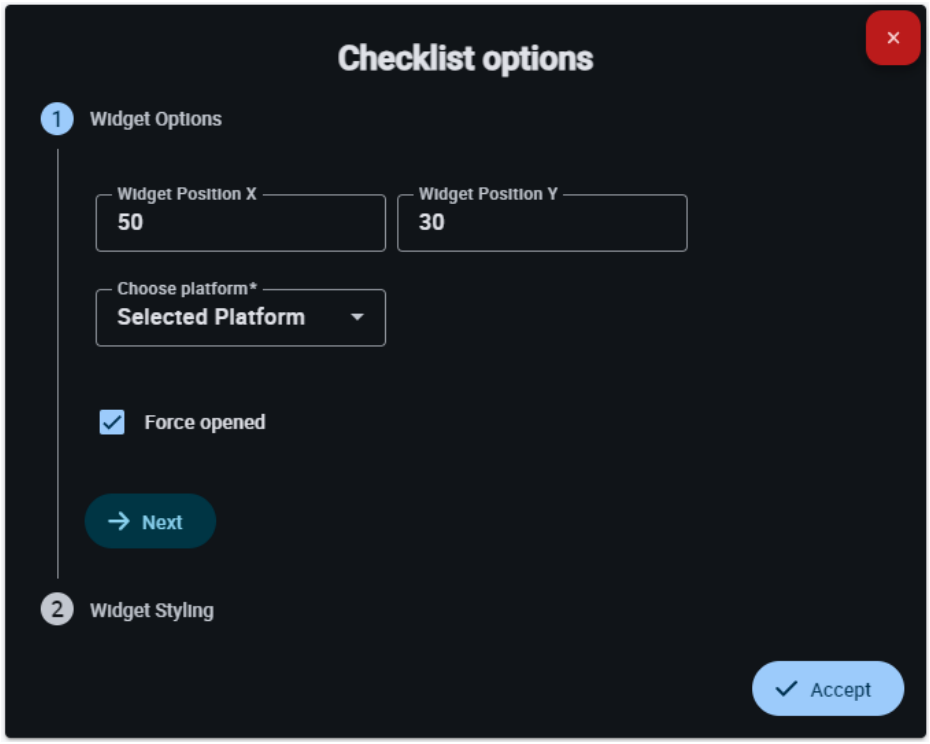
Checklist

- **Options**:



Checklist options

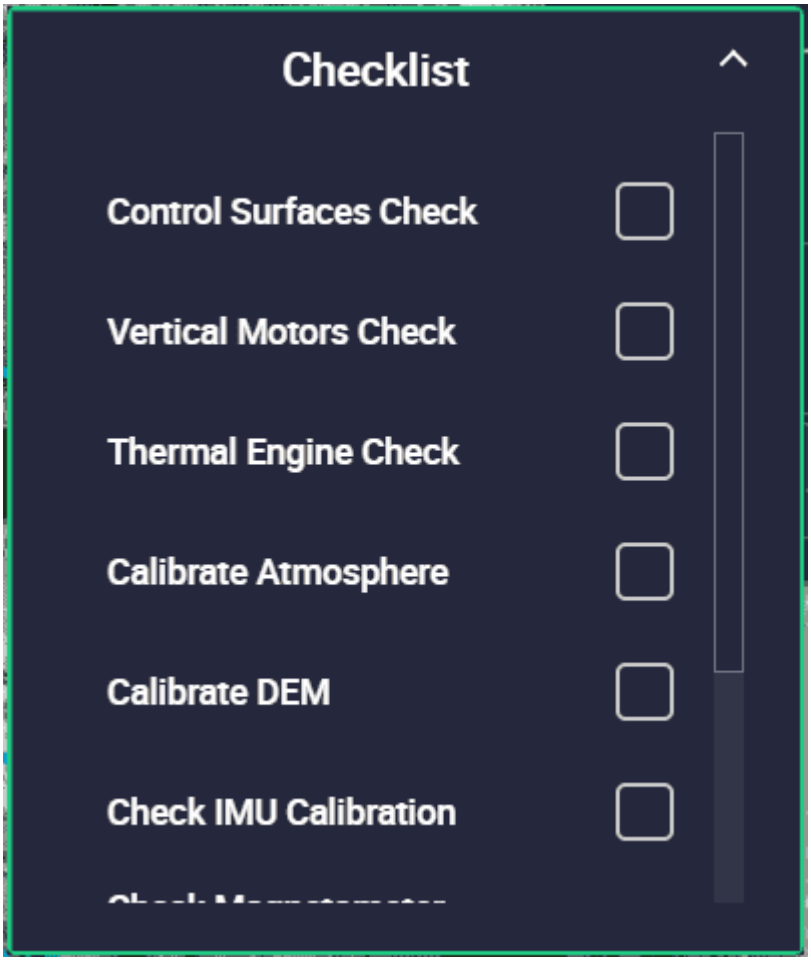
  - **Duplicate**: Duplicates this widget.
  - **Edit**: This allows the user to access the Checklist configuration menu. This widget has only the 'basic' configuration described in the Widgets common configuration.



Checklist Edit

    - **Forced opened**: When selected, the checklist remains open with no possibility of closing the window or minimizing it to an icon.
  - **Remove**: Deletes this widget.

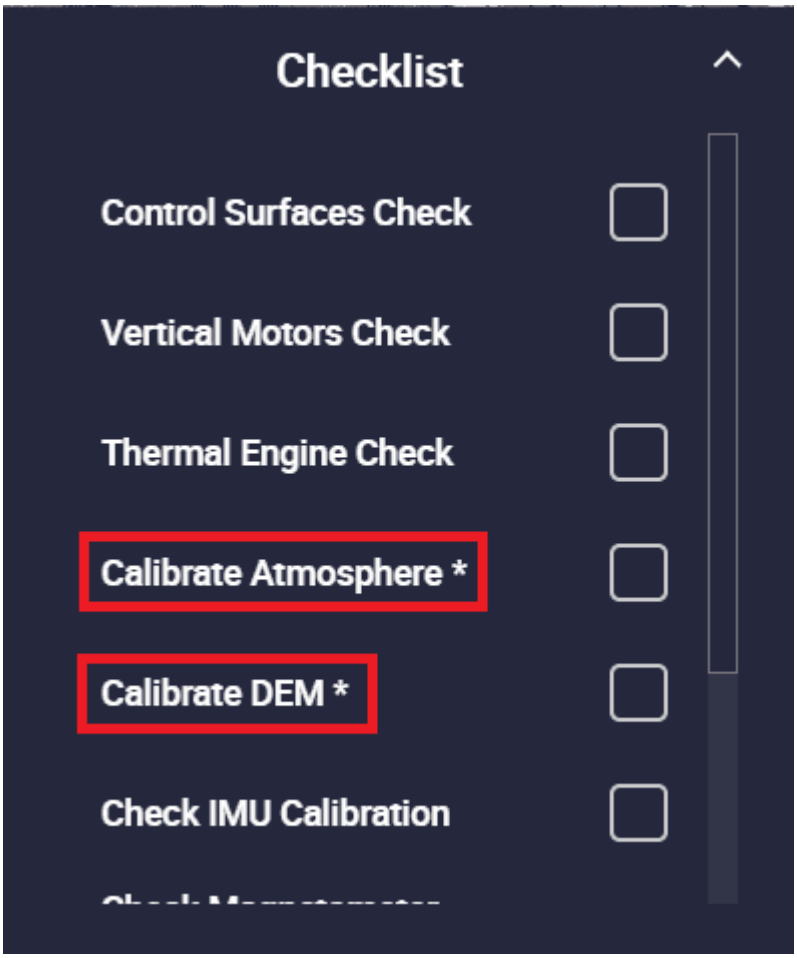
As this widget behaves like a drop-down menu, clicking on the icon will display all checklists:



Checklist deployed

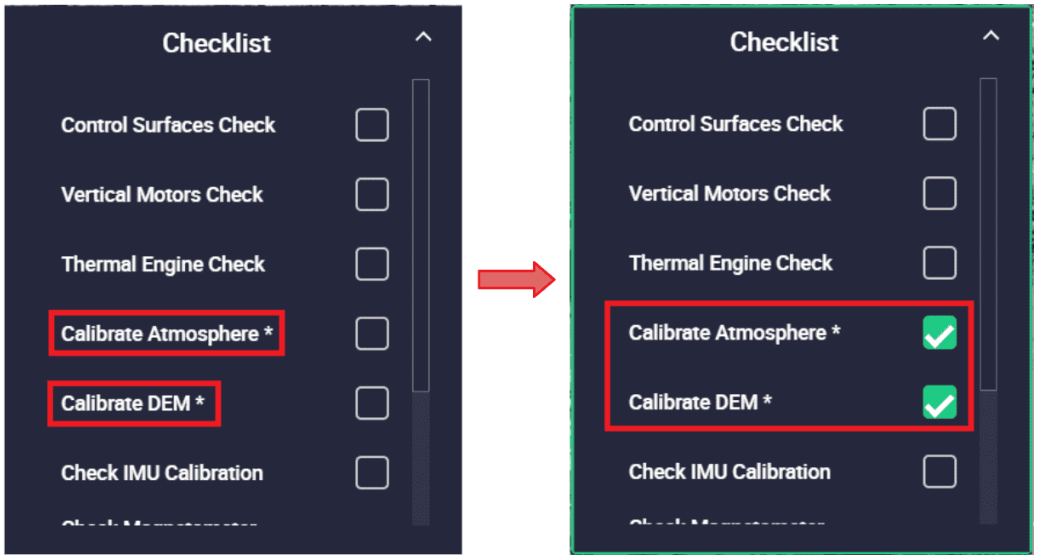
Besides, as can be seen in the figures above, the **checklist widget has a small green border**. This indicates that **everything is OK to start the operation**, even though there is no 'check' in the list.

This is because **none of these checks are mandatory** to switch phases or start the operation. **The mandatory checks are marked with an asterisk (*)** on the right, as shown in the image below.



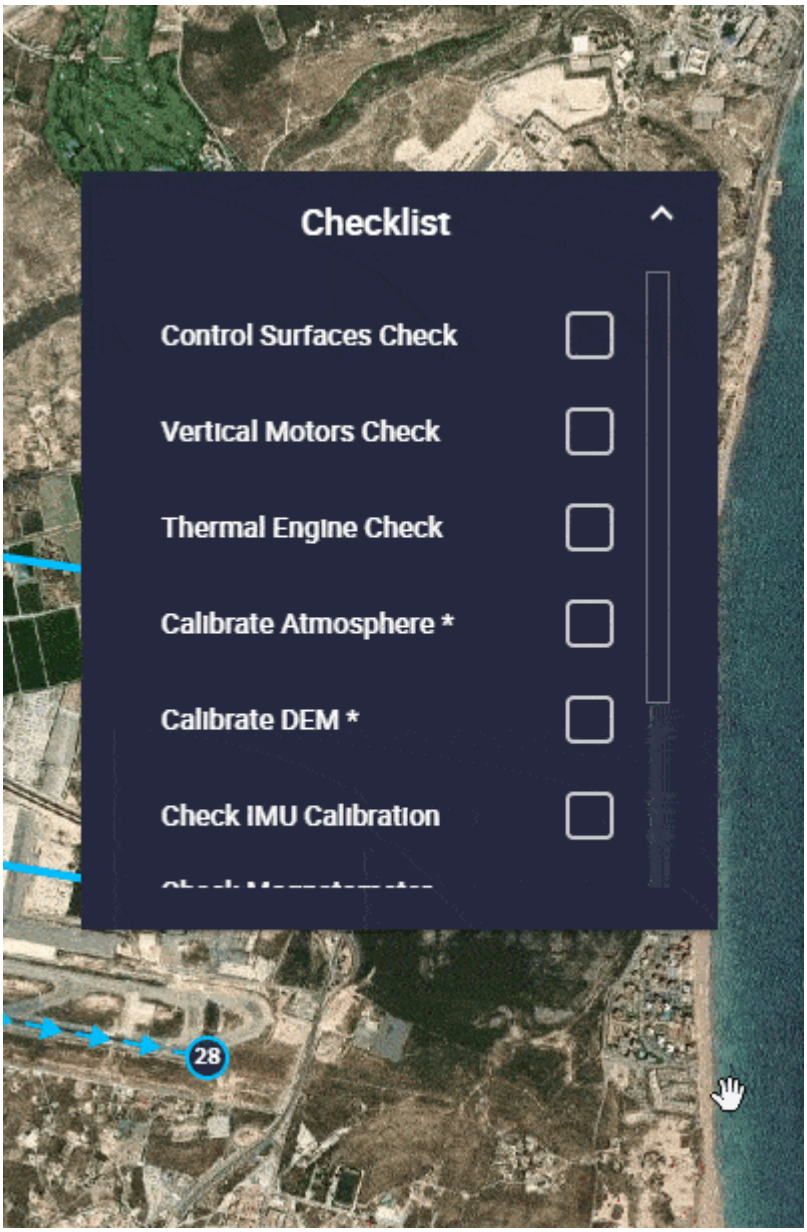
Checklist deployed - Not checked

In this case, the widget has **no green border until these mandatory actions are performed**:



Checklist deployed - Checked

To do so, checking them opens a menu that allows the user to perform these actions, in this case, some calibrations. An example is shown below:

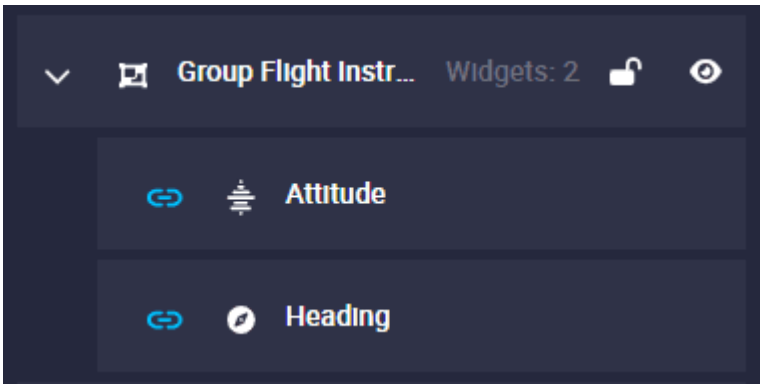


Checklist - Actions

Group Conditional

Veronte Ops allows users to create groups of widgets, where they can group as many widgets as they wish. This is useful for linking widgets and moving them together. In addition, it is possible to create a condition to choose when the group is shown.

Moreover, it is also common to create groups containing a single widget when using a condition for the group.



Group

> **Important**
>
> This widget is **managed** from the sorting panel.

Once the group is created, go to the **sorting panel** and **right click** on the group to **access its options**:

- **Remove**: Removes the group and also the widgets in the group.
- **Edit**: This allows the user to access the Group configuration menu.

Group configuration

  - **Widget Options**

    This widget has extra edit parameters compared to the ones described in Widgets common configuration.

    - **Name**: For easy identification of each group, users can enter a custom name for each one.

- **Check**: Here users can set the conditions they want for a group to be displayed or not.

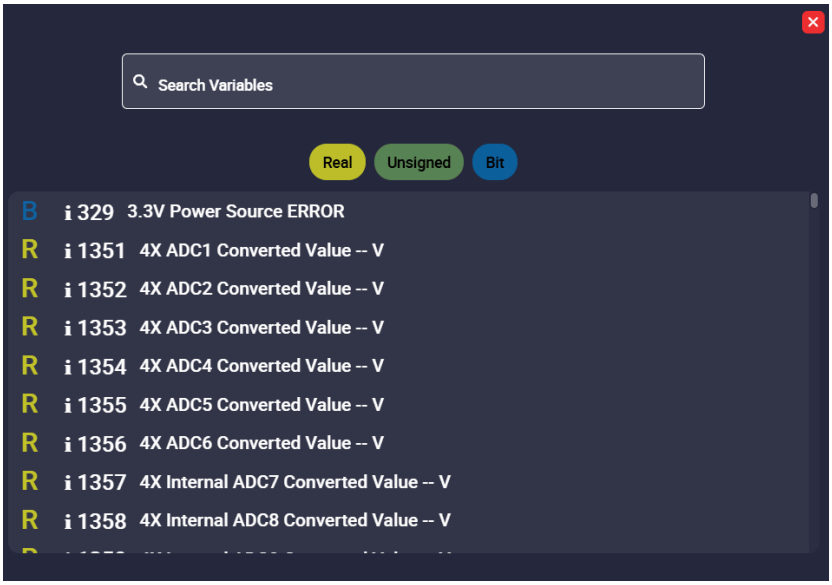
  These conditions must be set according to the values of the system variables.

  > ⓘ **Note**
  >
  > It is not mandatory to have a condition for a group.

  > ⚠ **Warning**
  >
  > For these conditions to apply correctly, widgets must be **locked** from the Workspace toolbar. Otherwise, these conditions will have no effect.

  Follow the next steps to correctly define a condition:

  1. Click on the + icon and select the desired variable from the list to established the condition:

     Group configuration - Variables to create a condition

  2. Build the condition with **logical operators**. The following is a list of logical operators that can be used by the user:

Condition	Operator	Description
Equality	==	

Condition	Operator	Description
		For more information click here
Inequality	!=	For more information click here
Greater than	>	For more information click here
Greater than or equal	>=	For more information click here
Less than	<	For more information click here
Less than or equal	<=	For more information click here
Logical AND	&&	For more information click here
Logical NOT	!	For more information click here
Logical OR	||	For more information click here

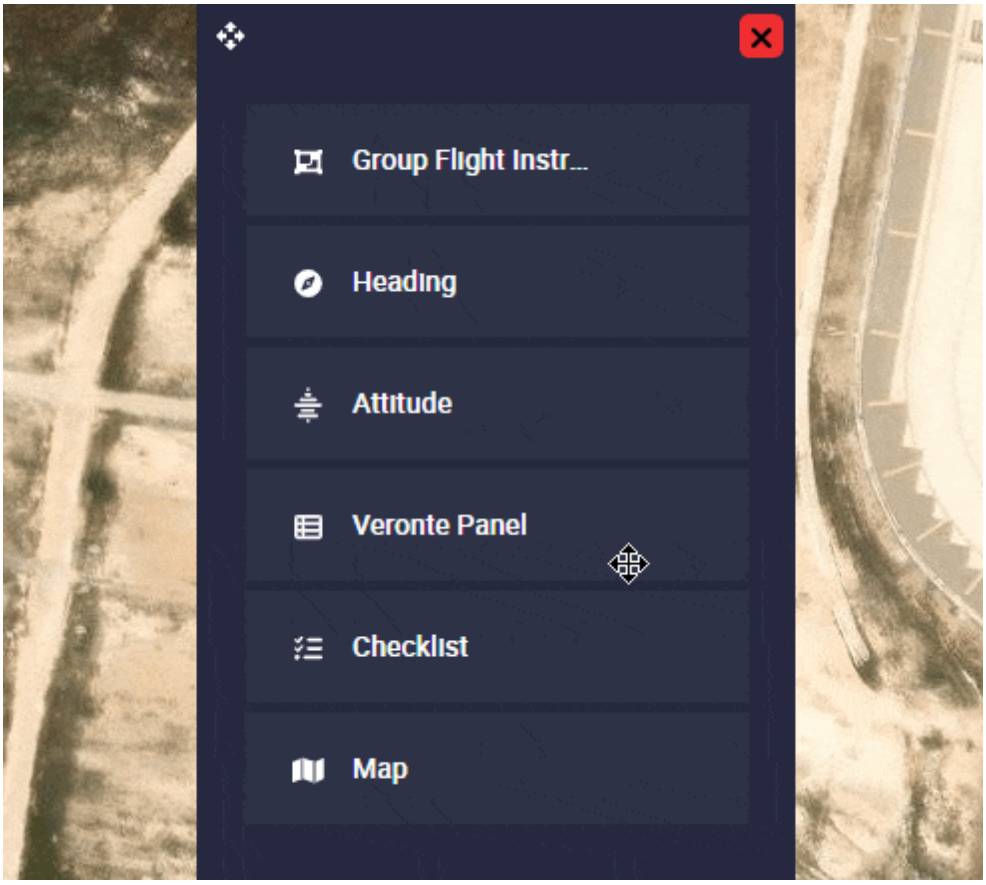
Condition	Operator	Description
Grouping operator	()	For more information click here

3. Press the **Accept** button to create the group with this condition.

After configuring it, return to the sorting panel where users can **add widgets to the created group**, **link/unlink, show/hide, lock/unlock, move and remove the added widgets**, as well as access the Group widget configuration menu (as described above).

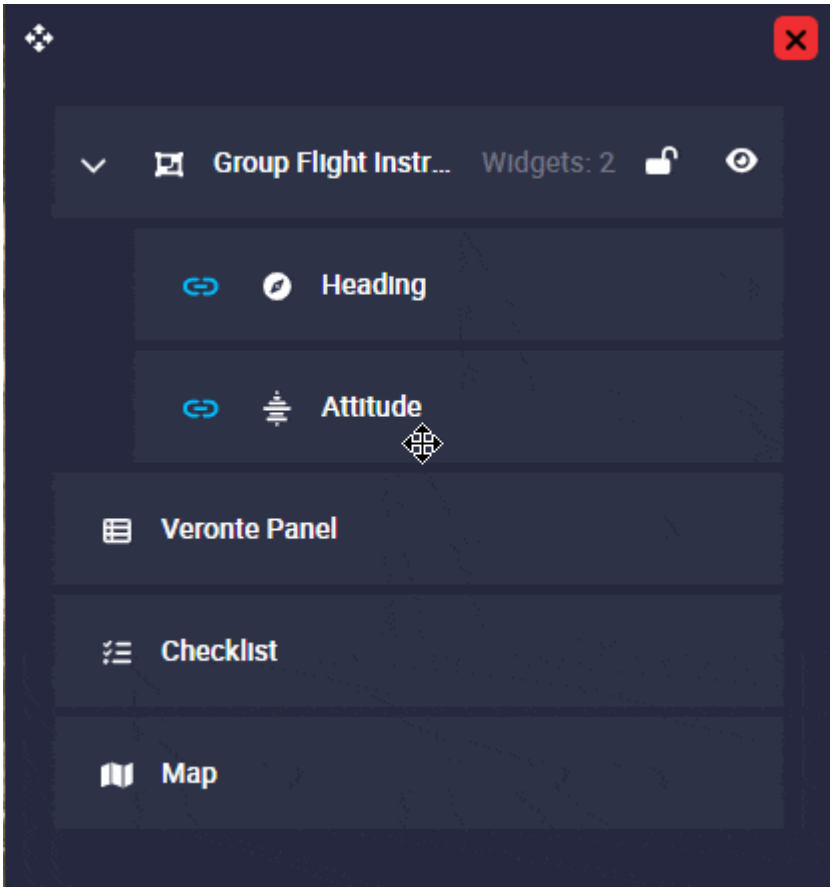
The following are the actions to be performed in the **sorting panel** in order to manage the widgets within the group:

- **To add a widget to a group**, simply select it and drag it into the Group widget:



Adding widgets to a group

- **To remove a widget from a group**, select it and drag it out of the Group widget:



Removing widgets from a group

- **Linking widgets**. By default, widgets are linked and appear with a icon next to them in the group. Therefore, if users move one of the widgets in the group, the rest of the linked ones will move together. To unlink them, click on the icon on the widgets to be unlinked, then they will appear as unlinked ( ). Thus, if one is not marked, it may move with respect to the other elements in the group.
- **Show/Hide groups**. By default, all groups are shown in the workspace (they appear as). Nonetheless, by clicking on they can be hidden. To show them again, simply click on .

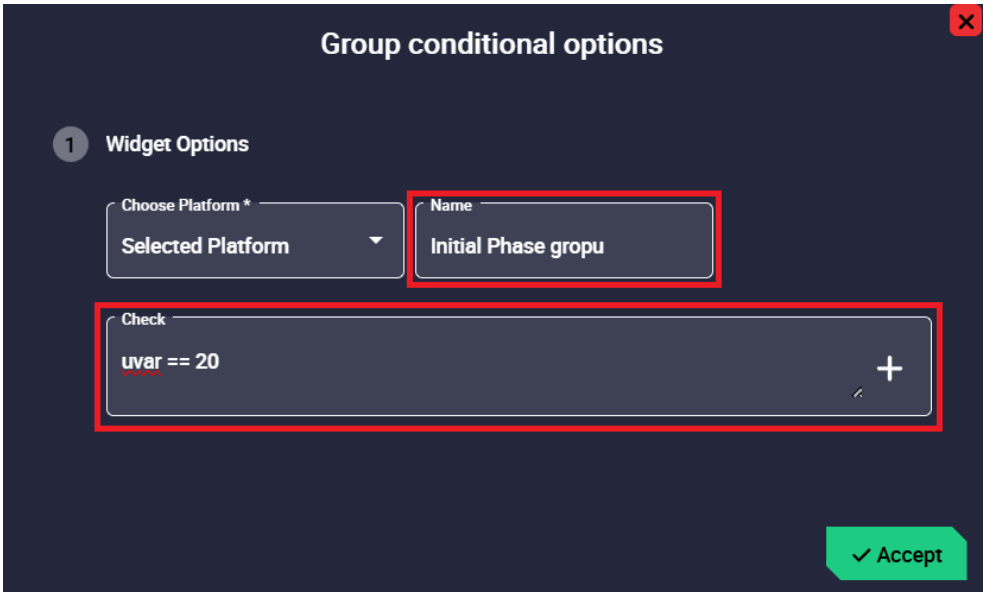
> **Hint**
>
> If users want to show different groups at different "moments" of the flight, it may be useful to hide them during creation. In this way, groups can be placed in the same place and thus "save" space in the workspace.

- **Lock/Unlock groups**. When groups are locked, users will not be able to move them in the workspace. By default, all groups are unlocked ( ). To lock them, simply click on and they will appear as . In addition, when a **group is locked**, users **cannot add, remove, link or unlink the widgets** that belong to it.

Example of a Group widget with condition

If users only want to have a pre-flight Checklist widget before starting the flight (i.e. only in the **initial phase**):

- For only displaying this group in the initial phase, build the following condition in the Group widget configuration:

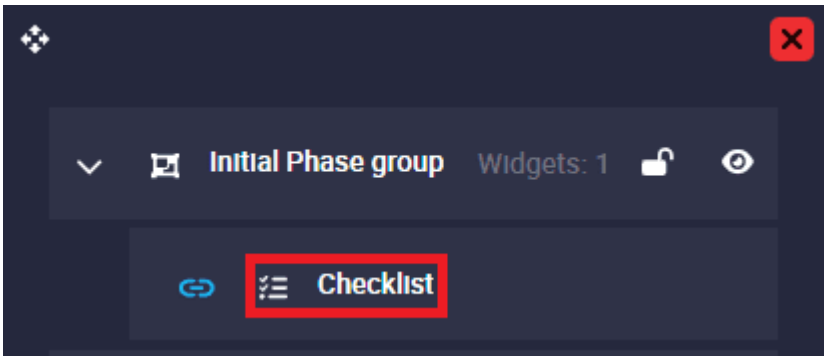


Configuration of group with condition

  Where:

  - **uvar1** is the **Phase identifier** variable
  - = = is the **equality** operator
  - **20** is the ID of the initial phase

- Add the desired Checklist widget to the group just created

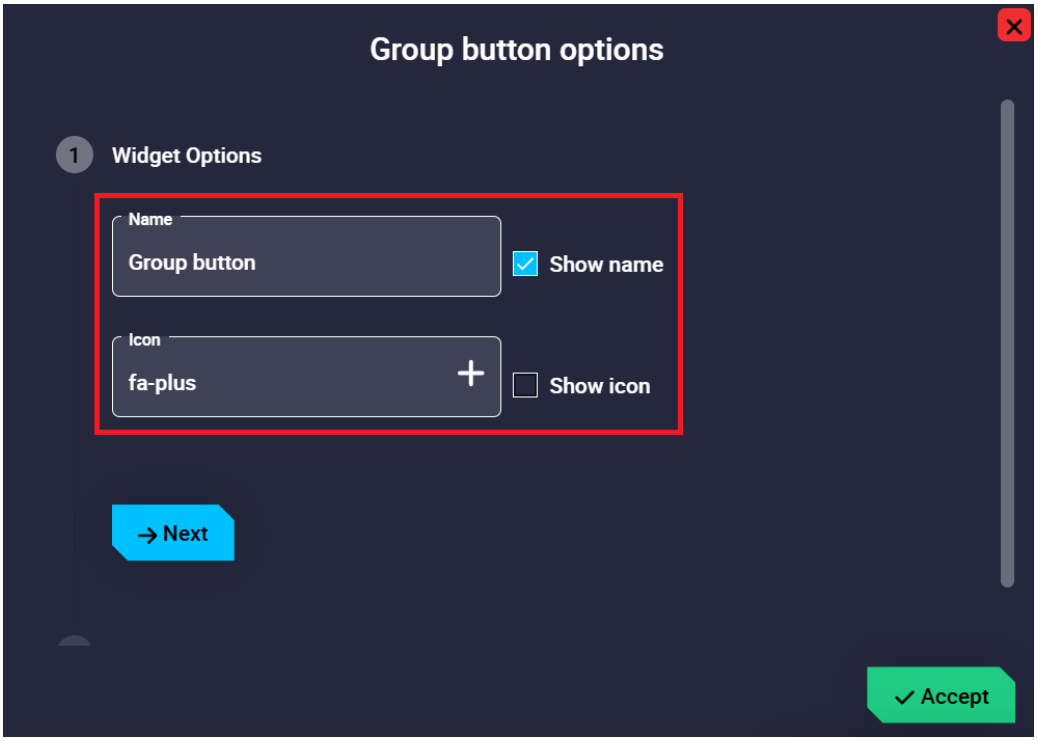


Group with condition

Therefore, this implies that: Only when the flight phase is the one with ID 20 (initial phase), widgets added to this group, in this case the Checklist widget, will be displayed.

Group button

This panel is used to group more tha one variable in a single graphical element, optimizing space on the control dashboard and allowing multiple commands to be activated from a single interface.



Group Button Options

- **Name**: Defines the name of the button group. By checking the Show name box, the user decides whether to display the text on the widget.

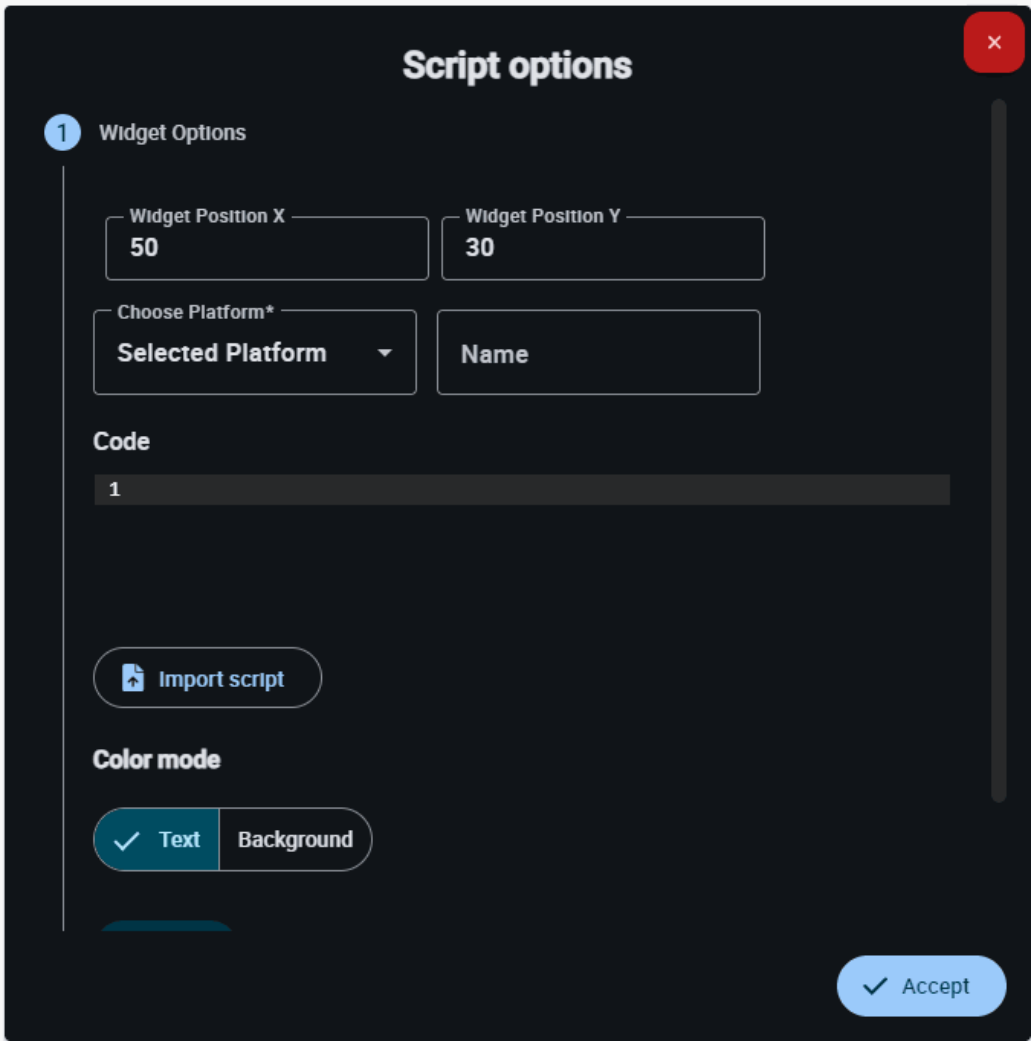
- **Icon**: Allows choosing an identification icon for the group. By checking the Show icon box, the user can decide whether to show the icon on the widget.
- **Show name**: If this checkbox is selected, the text entered in the Name field will be displayed directly on the widget within the dashboard.
- **Show icon**: If this checkbox is selected, the icon chosen in the Icon field will be visible on the widget. This allows for visual identification of the button group's function without needing to read the text.

As this widget behaves like a pop-up menu, clicking on the main button will display all the buttons configured within it, ready to be activated by the operator during the mission.

Group Button

Script

The Script options panel allows integrating and configuring custom scripts, defining their behavior and display settings.



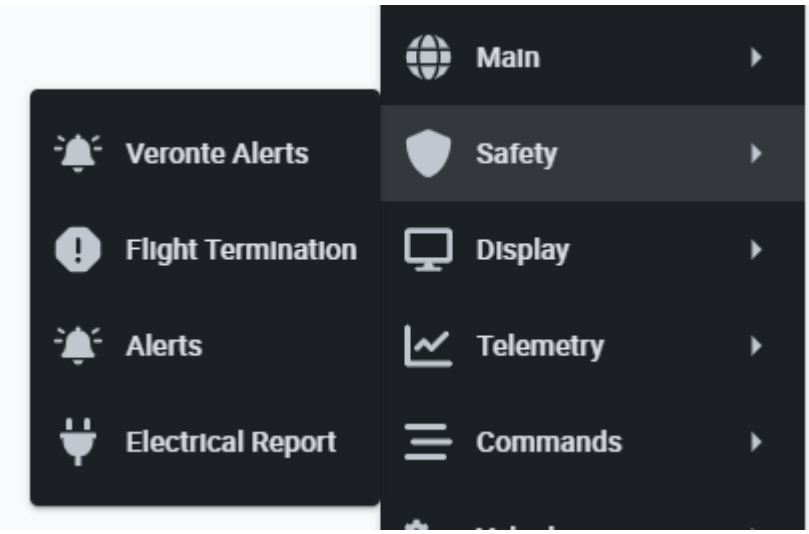
Script

- **Name**: Field for assigning an identifying name to the script.
- **Code**: Area dedicated to entering or viewing the numerical or logical code of the script.
- **Import script**: Button to upload script files from outside the system.
- **Color mode**: Allows customizing the visual appearance by choosing between Text or Background color modification.

For more information about this widget, check the Script Widget - Appendix section of this manual.

Safety

The Safety menu groups features dedicated to system health monitoring and the execution of emergency procedures according to proprietary and international standards.



Safety

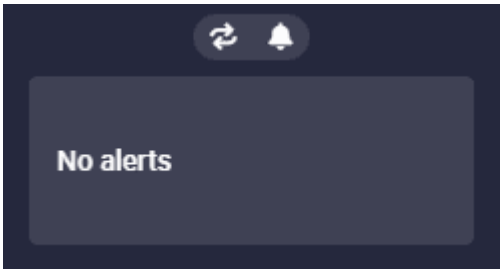
Veronte alerts

These alerts are **bits** variables that **appear depending on the state of the bit, failure or success**. The user can define in the widget configuration which of them will be the triggering state.

By default, alerts are "collapsed" into the widget, as the widget acts as a drop-down menu. However, users can quickly find out how many alerts there are and what type they are without having to drop it down thanks to the colored indicators in the form of pills:

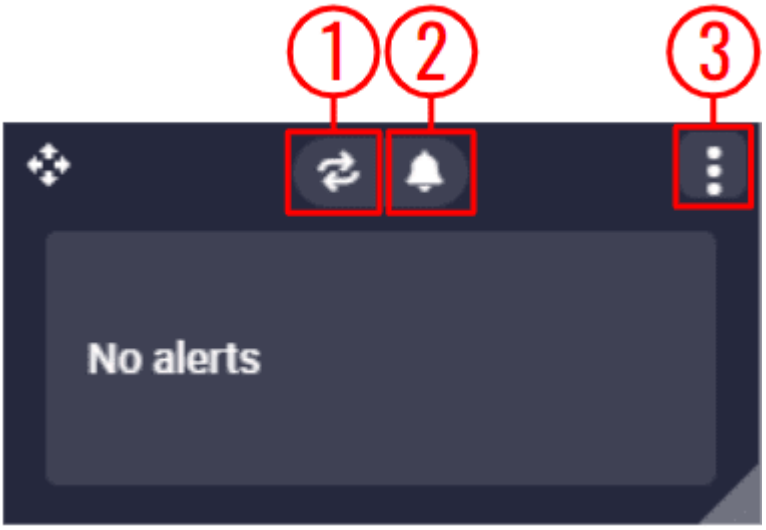
- Red: Alerts on failure state
- Orange: Alerts in failure state and configured as warnings
- Green: Alerts in success state
- Grey: Alerts for which the necessary telemetry is not being received.

> ⓘ **Note**
>
> - This color code is also extended to the displayed alerts.
> - The alert type can be configured by accessing its edit menu.



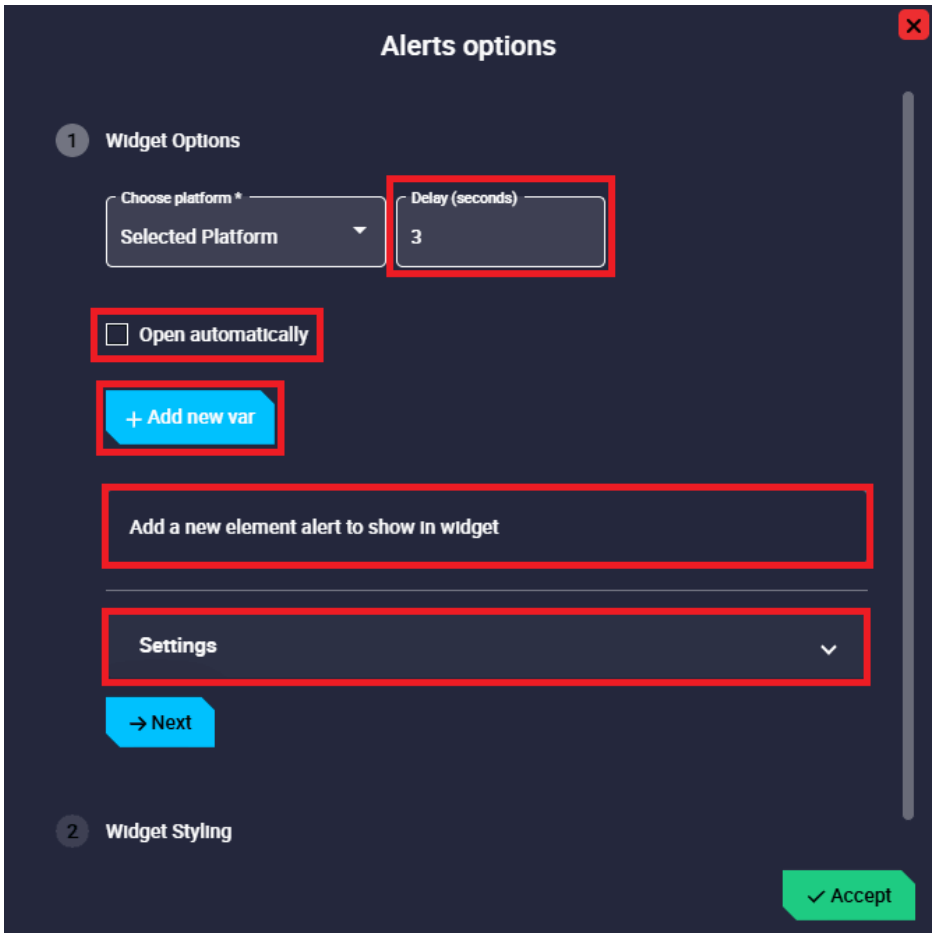
Alerts

By hovering the mouse cursor over the widget, the following actions appear:



Alerts

1. **Enable/Disable Loop**: Users can enable/disable the sound of the alerts to be in loop, i.e. sounding continously.
2. **Mute/Unmute all alerts**: The sound of all alerts can be mute/unmute.
3. **Options**:
    - **Duplicate**: Duplicates this widget.
    - **Edit**: This allows the user to access the Alerts configuration menu.



Alerts configuration

- **Widget Options**

  This widget has **extra edit parameters** compared to the ones described in the Widgets common configuration.

  - **Delay**: This is the time (in seconds) between the sounding of alerts **when the loop option is enabled**.
  - **Open automatically**: If enabled, Alerts widget will automatically "expand" when a configured alert is triggered.

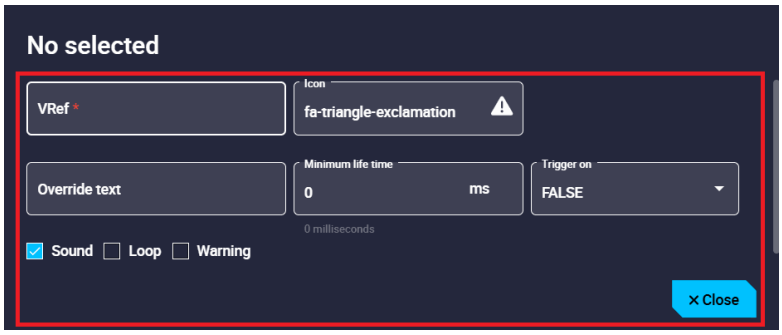
    > ⓘ **Note**
    > This check only takes effect when the widget is not deployed.

  - **Add new var**: Press this button to add a new alert variable. They are configured below.
  - **Add a new element alert to show in widget**: When adding a variable (by clicking on the **'Add new var' button**), the following field is added to the configuration menu:

Alerts configuration - Variables

Next, 2 buttons appear to configure the new variable added:

- **Settings var**: Pressing this button takes the user to the configuration menu of the added variable:



Alerts configuration - Variables settings

  - **VRef**: Users can select the desired bit variable to be displayed as an alert.
  - **Icon**: Choose the icon to be displayed next to the variable. By default, it is an exclamation icon, as can be seen in the figure above.
  - **Override text**: The alert name can be changed by typing a new one different from the variable name.

> ⚠ **Warning**
> This does not rename the selected variable in the configuration, it is only the name shown in the alert.

  - **Minimun life time**: Minimum delay before the alert activates.
  - **Trigger on**: Users can choose when they want the alert to be displayed, when **FALSE** or **TRUE**. By default, **FALSE** is set.

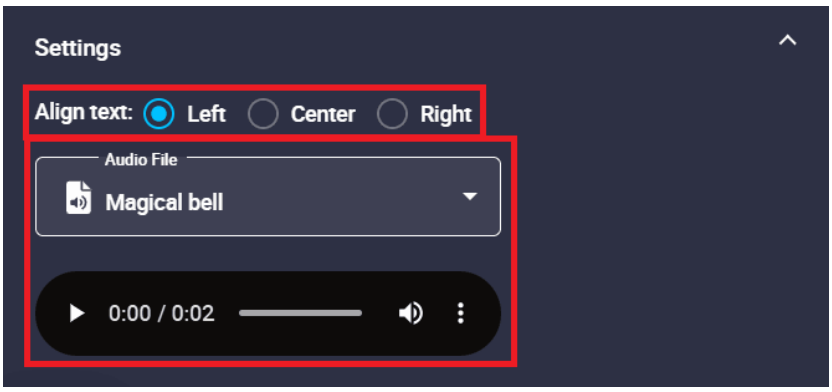
- **Sound**: Here users can choose whether each alert sounds or not by simply enabling/disabling it. By default, it is activated.
- **Loop**: If **enabled**, the sound of the selected alert will be **looped with the delay time specified** above. **Otherwise**, the **alert sound will only play when the alert is triggered**.
  For example, if the delay is set to 3 seconds, the loop option is enabled and the selected variable is in fail mode, the alert sound will sound every 3 seconds.

  > **Important**
  > This is only available when **Sound** is activated.

- **Warning**: If enabled, the selected alert will be displayed in orange instead of red.
- **Close**: Save and close button.

- **Delete var**.

- **Settings**: Here the user can modify the settings common to all alerts:

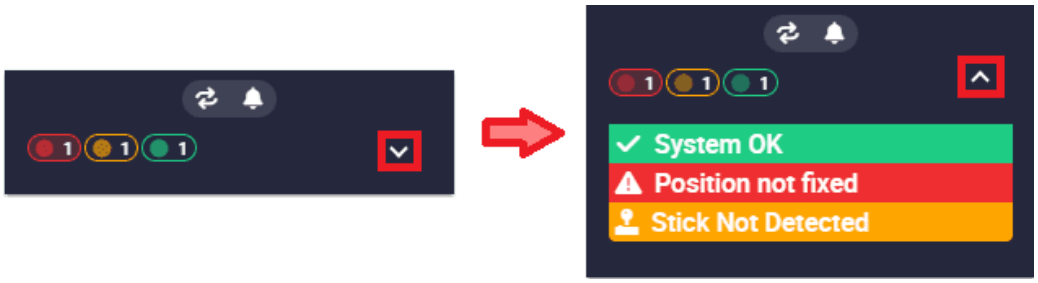


Alerts configuration - Settings

- **Align text**: The variable text can be left, center or right aligned.
- **Audio File**: Users can select the audio of the alert from the audio list.

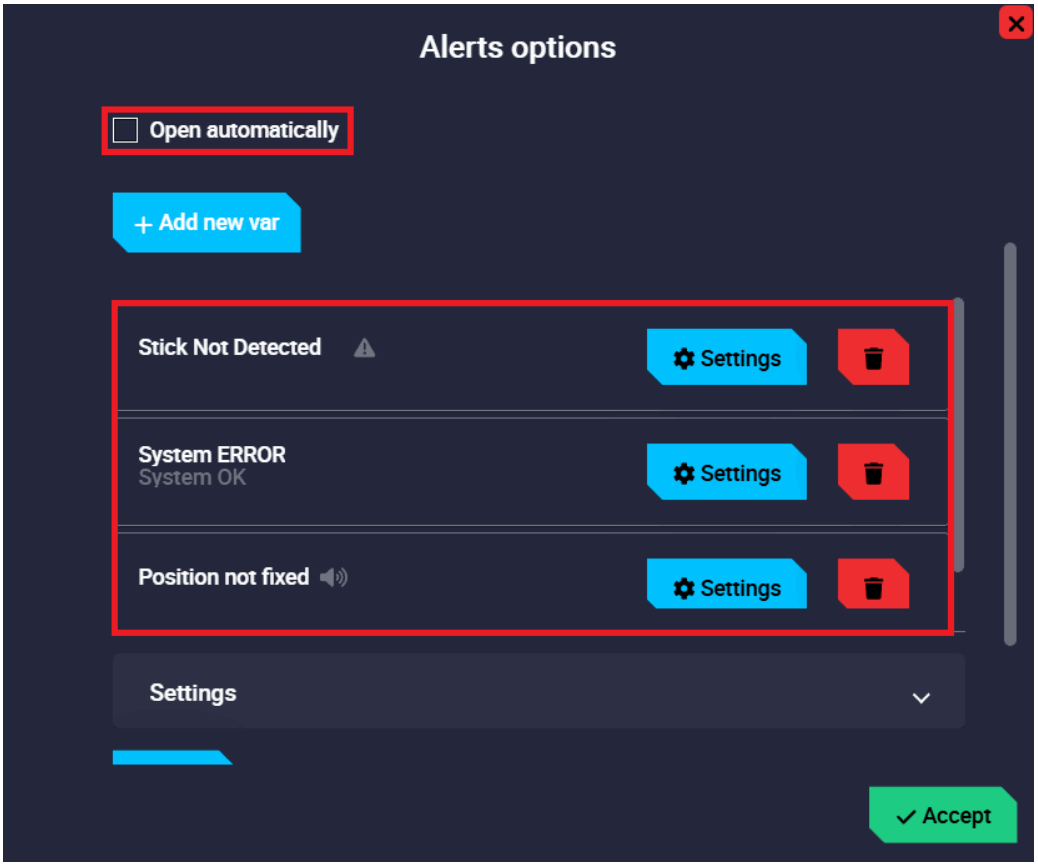
- **Remove**: Deletes this widget.

Some examples of alerts are shown below:



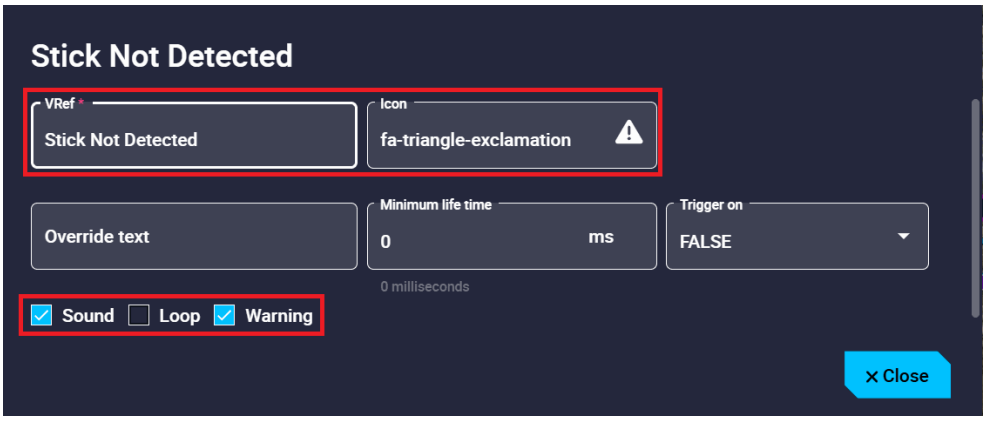
Alerts example

The following figures show how these alerts have been configured:



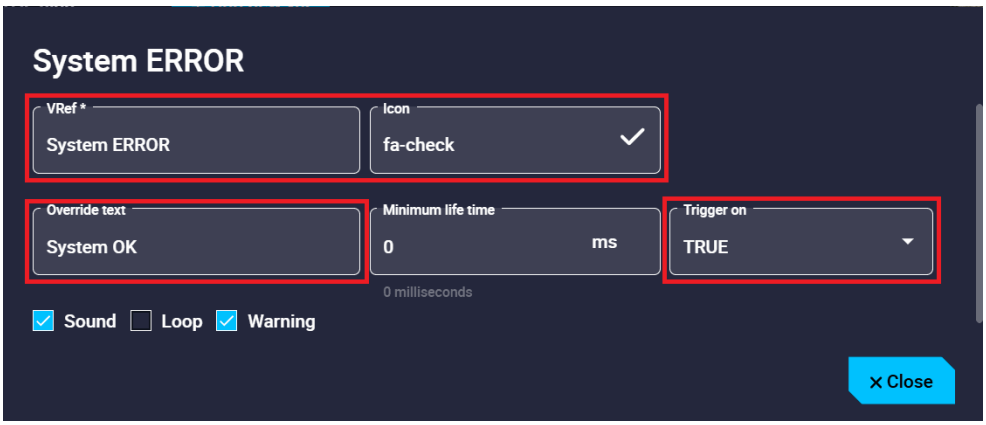
Alerts configuration example

- Stick Not Detected: This alert has been defined as a **warning**, its icon has been replace by another icon more related to the variable (a **joystick icon**) and has been configured without sound.



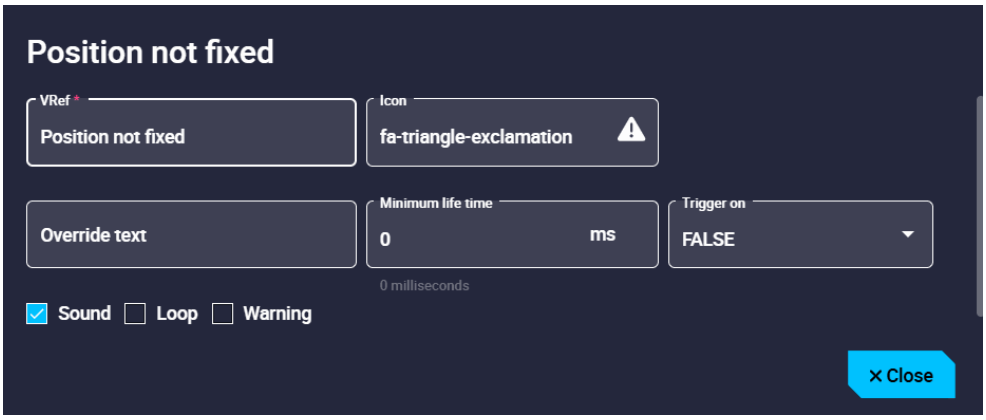
Alerts configuration example - Stick Not Detected

- System Error: This variable has been configured as an alert but it is triggered in **successful state**, so it is colored green.
  In addition, it has been customized with a different text from the variable name, with a 'check' icon and no sound.



Alerts configuration example - System Error

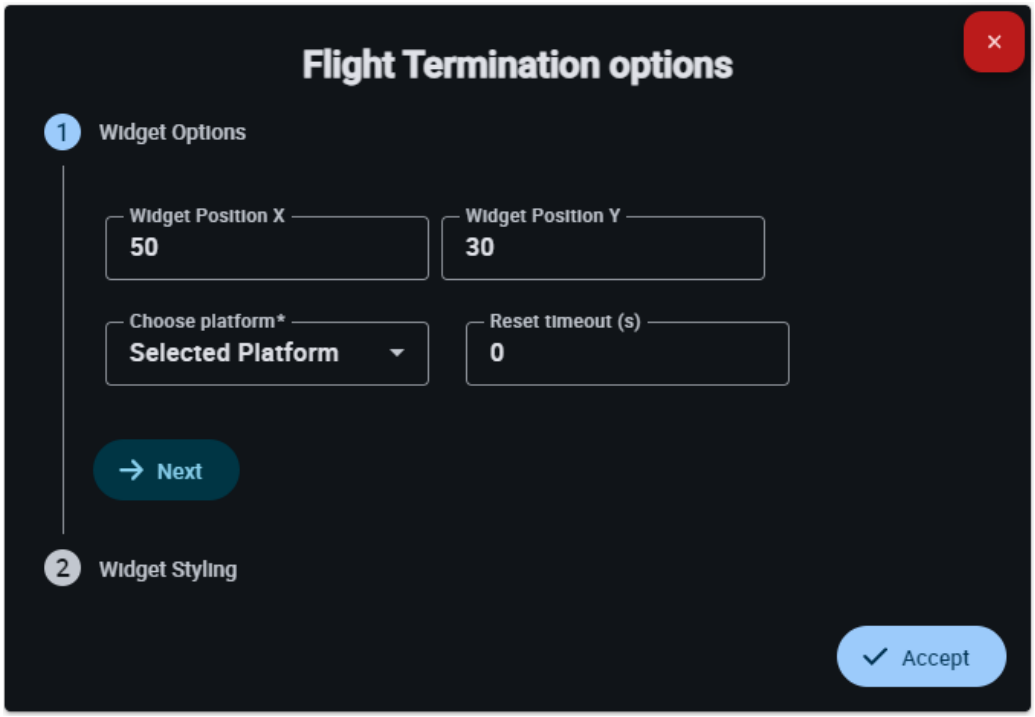
- Position not fixed: This alert has not been customized, it appears with the default configuration.



Alerts configuration example - Position not fixed

Flight Termination

Interface dedicated to executing the Flight Termination System command via the specific message defined by the STANAG standard.



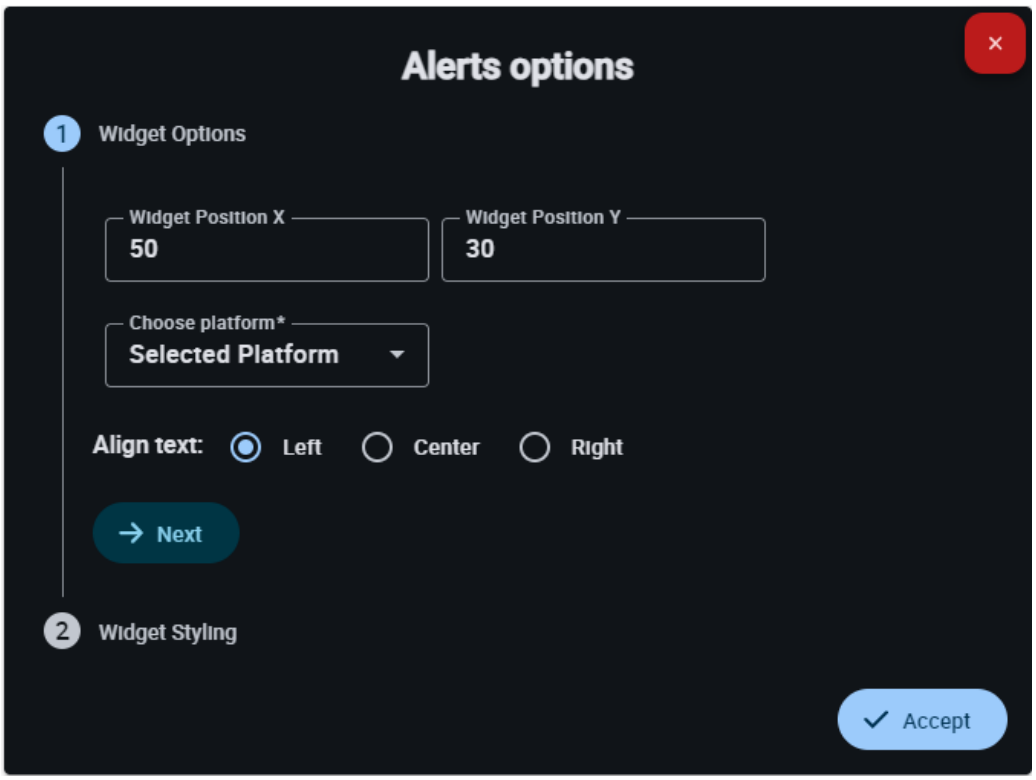
Flight Termination options

This widget has **extra edit parameters** compared to the ones described in the Widgets common configuration.

- **Reset timeout (s)**: Specifies the time after which the command or widget status is reset.

Alerts

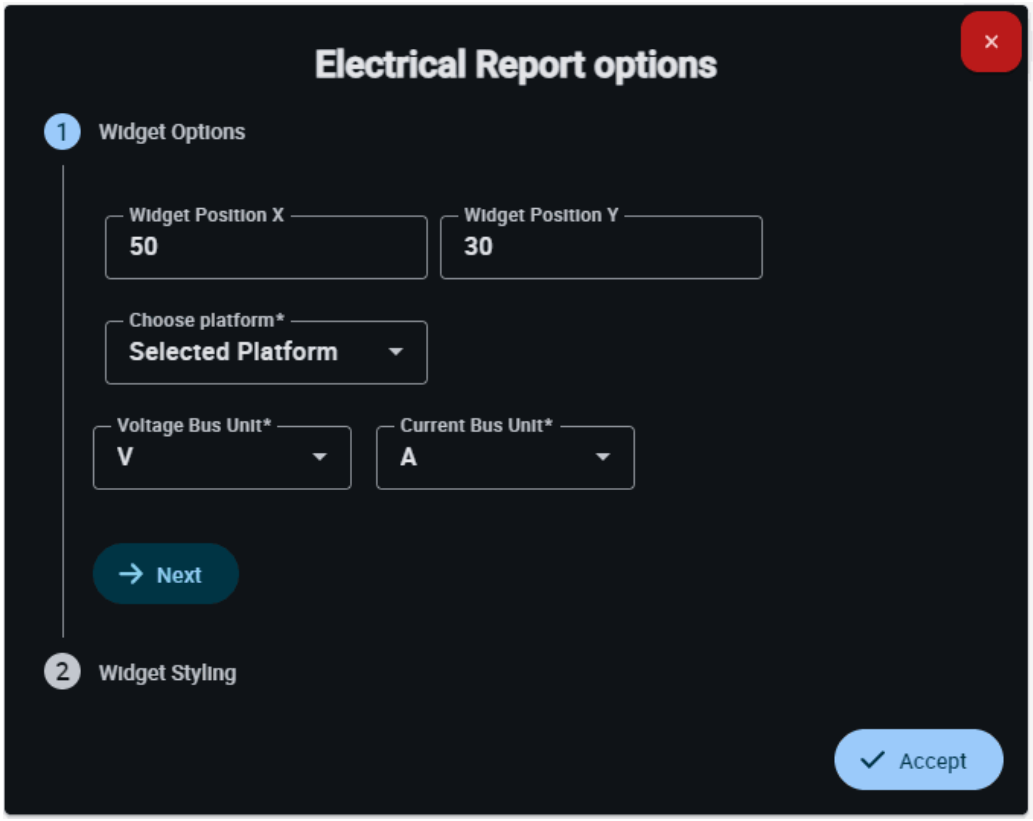
Alert panel compliant with the STANAG standard; it displays exclusively the messages defined by the international protocol.



Alerts options

Electrical Report

Tool based on STANAG messaging for real-time monitoring of the aircraft's electrical system parameters.



Eletrical Report options

This widget has **extra edit parameters** compared to the ones described in the Widgets common configuration.

- **Voltage Bus Unit**: Allows selecting the unit of measurement for the bus voltage between V and mV.
- **Current Bus Unit**: Allows selecting the unit of measurement for the bus current between A and mA.

Display

Statics widgets **do not depend on the selected platform**, therefore the 'Choose platform' parameter will not appear in any of the widgets described below.

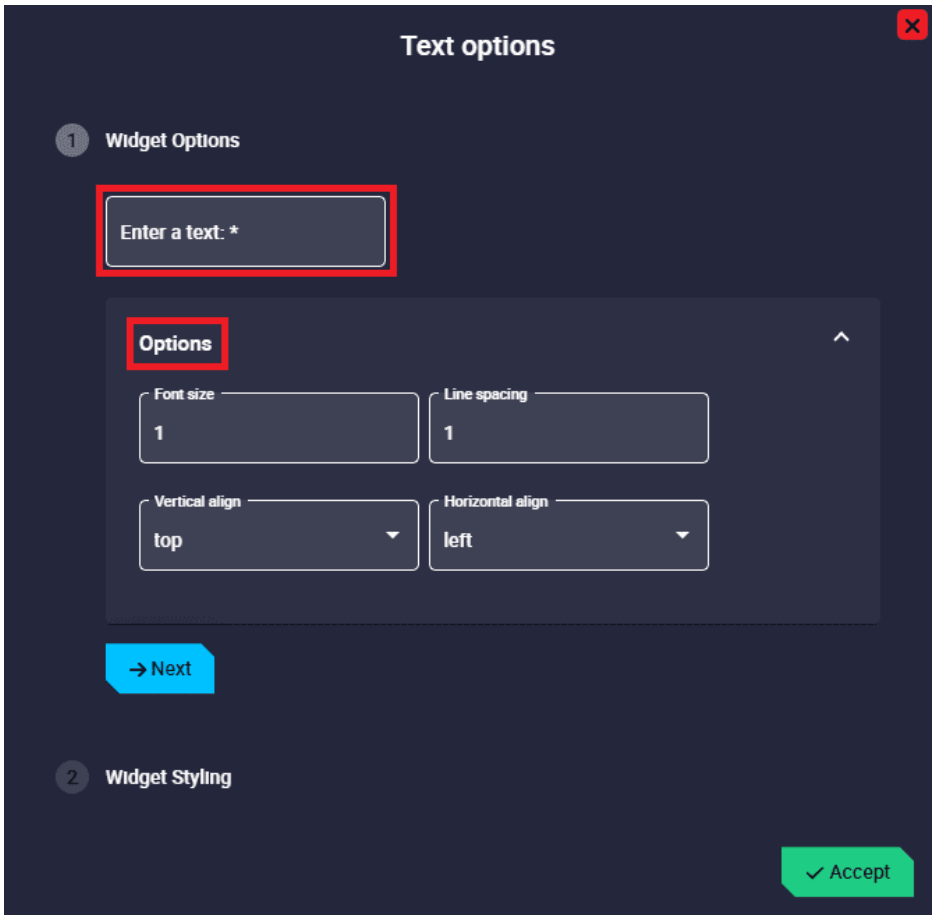
Text

Text widget allows the user to insert text labels in the workspace.

Text

Right click on the **text** to access its options (**Duplicate**, **Edit** and **Remove**).

- **Options → Edit**: This allows the user to access the Text configuration menu.
  - **Widget Options**:



Text configuration

- **Widget Position X / Y**: Defines the spatial coordinates for the script widget's placement on the screen.
- **Enter a text**: Enter the desired text to be displayed as a label.
- **Options**: The following options allow the user to customize the Text widget as desired:
  - **Font size**: The text size can be adjusted by the user. The value must be ≥ 1 .
  - **Line spacing**: When the text in the widget becomes a paragraph, this is the space between each line of that paragraph. The value must be ≥ 1 .
  - **Vertical align**: Text can be vertically aligned. The available options are center, bottom and top.
  - **Horizontal align**: Text can be aligned horizontally. The available options are center, justify, left and right.
    The justify option only makes sense when the text is a paragraph.

- **Widget Styling**: This widget has the 'basic' Widget Styling configuration described in the Widgets common configuration.

Image

Image widget allows the user to insert an image in the workspace.

> ⓘ **Note**
>
> A gif can also be added with this widget.

Image

Right click on the **image** to access its options (**Duplicate**, **Edit** and **Remove**).

- **Options → Edit**: This allows the user to access the Image configuration menu.
  - **Widget Options**:

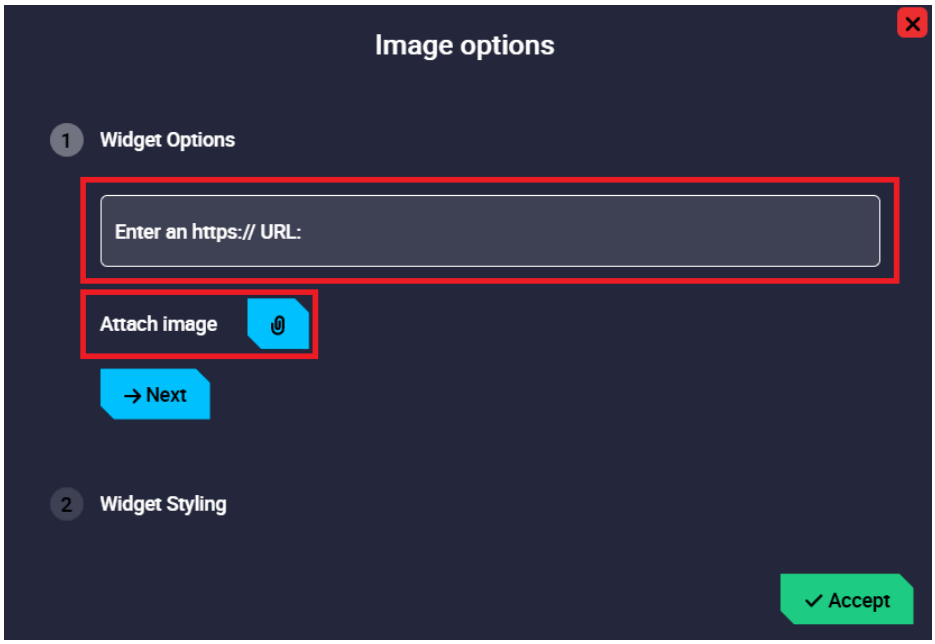


Image configuration

- **Enter an https:// URL:** Users can enter the URL of the image to be displayed.

- **Attach Image**: It is possible to import an image from PC.
- **Widget Styling**: This widget has the 'basic' Widget Styling configuration described in the Widgets common configuration.

An example is given below:

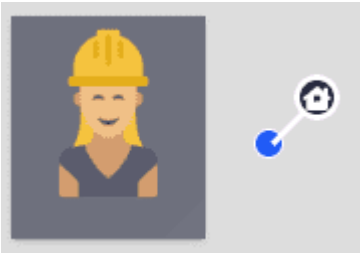


Image example

As can be seen in the figure below, this image has been imported from the browse.

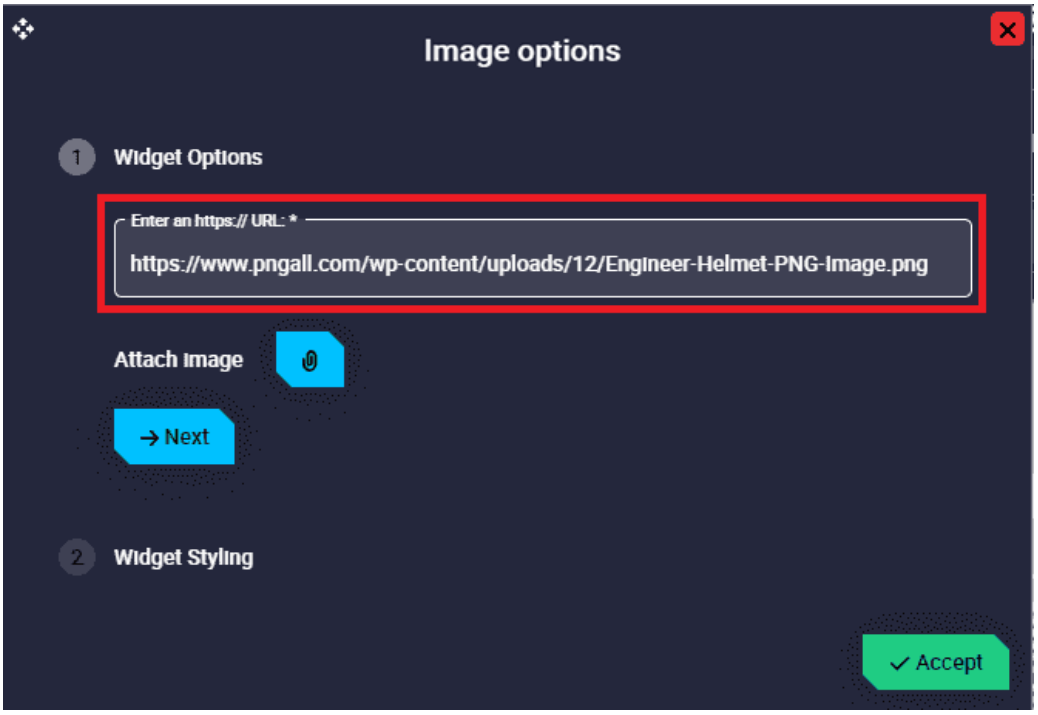


Image configuration - Widget Options example

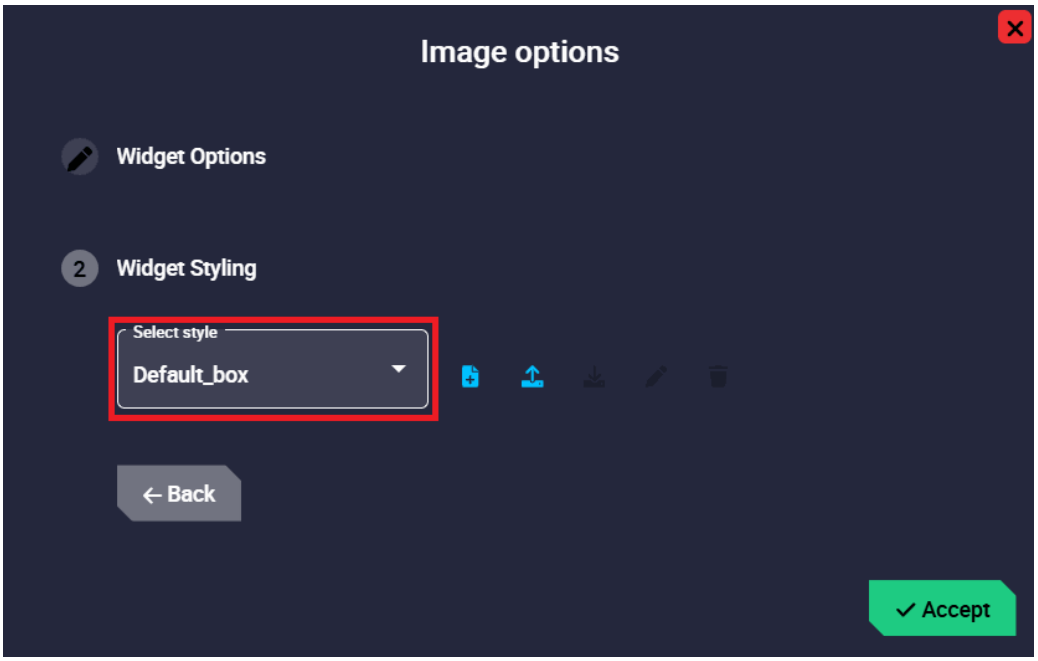
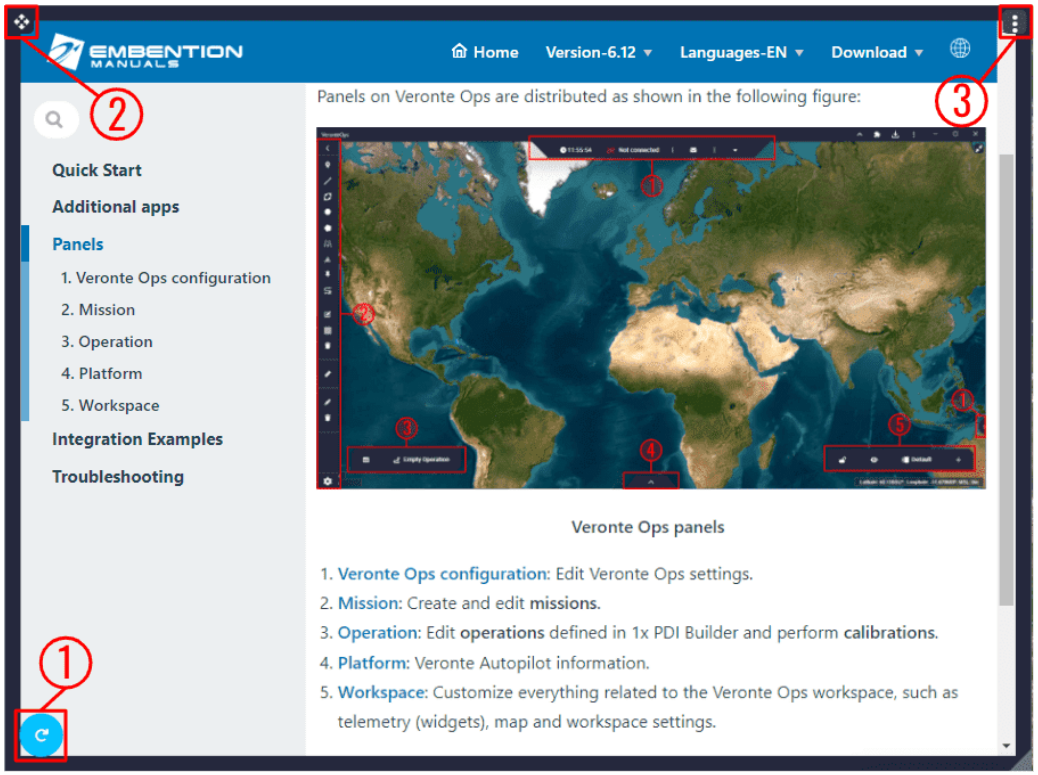


Image configuration - Widget Styling example

Iframe

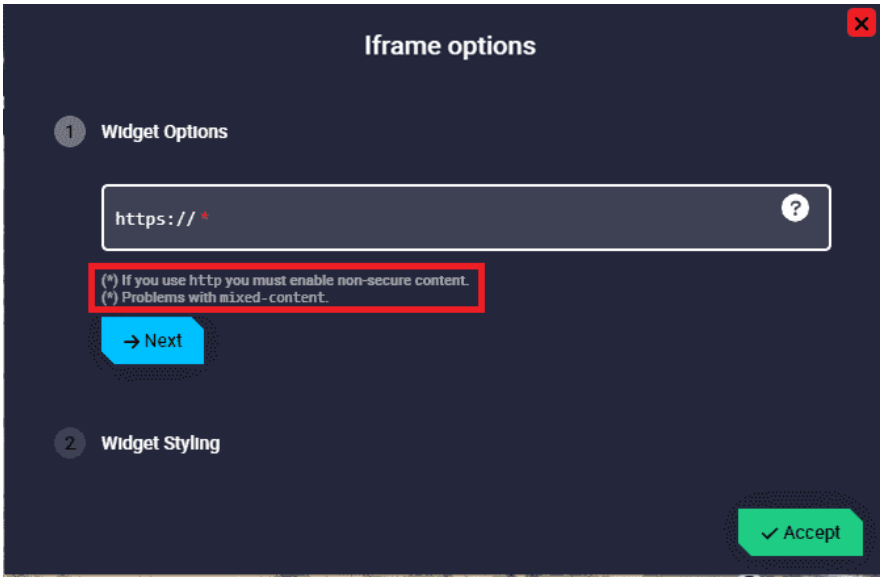
Iframe widget allows the user to embed a **web page** as a widget in the workspace, as well as to visualize video from a gimbal camera.

For example, useful websites could be: the windy website, the veronte operations manuals, the web application of the external radio if it is being used for operation, etc.



Iframe

1. **Reload page**: Refreshes the web page embedded as a widget.
2. **Move**: Allows the user to move the widget to the desired position in the workspace.
3. **Options**:
   - **Duplicate**: Duplicates this widget.
   - **Edit**: This allows the user to access the Iframe configuration menu.
     - **Widget Options**:



Iframe configuration

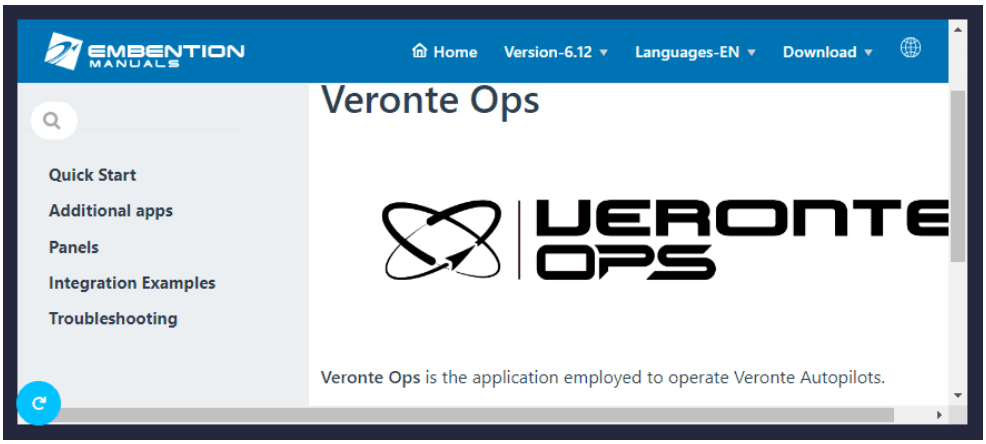
       - **https:// :** Users can enter the URL of the web page to be displayed.
         In order to be able to view the video recording from a gimbal and also tracking directly with the image, users must copy here the URL generated by the Web Converter, which converts from **RTSP** to **WebRTC**.

> ⚠ **Warning**
> Please take into account the notes in the widget.

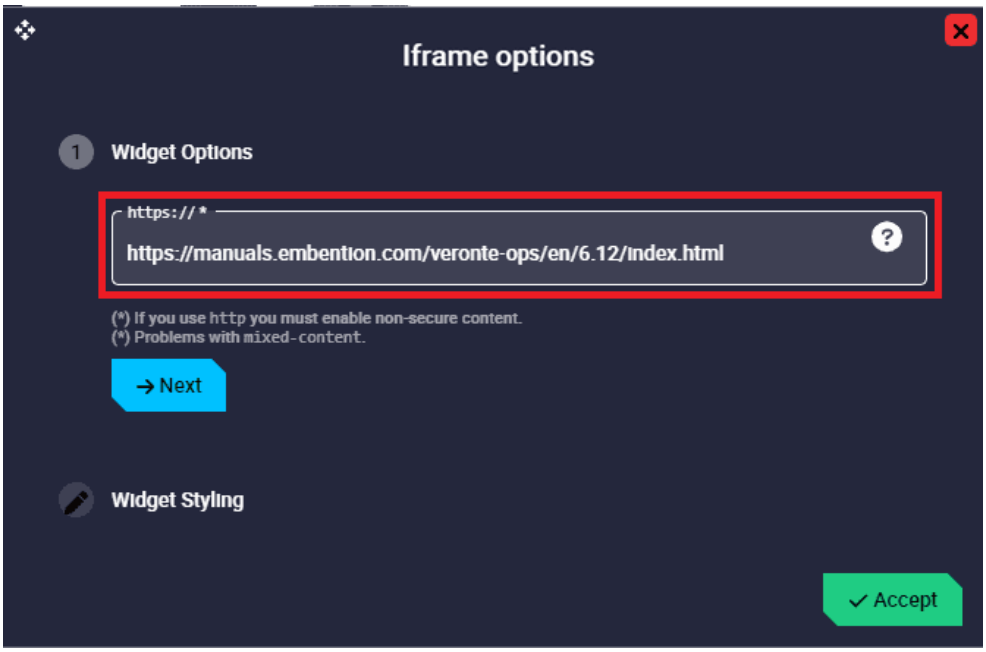
     - **Widget Styling**: This widget has the 'basic' Widget Styling configuration described in the Widgets common configuration.
   - **Remove**: Deletes this widget.

Some examples example are given below:

- **Example 1**:

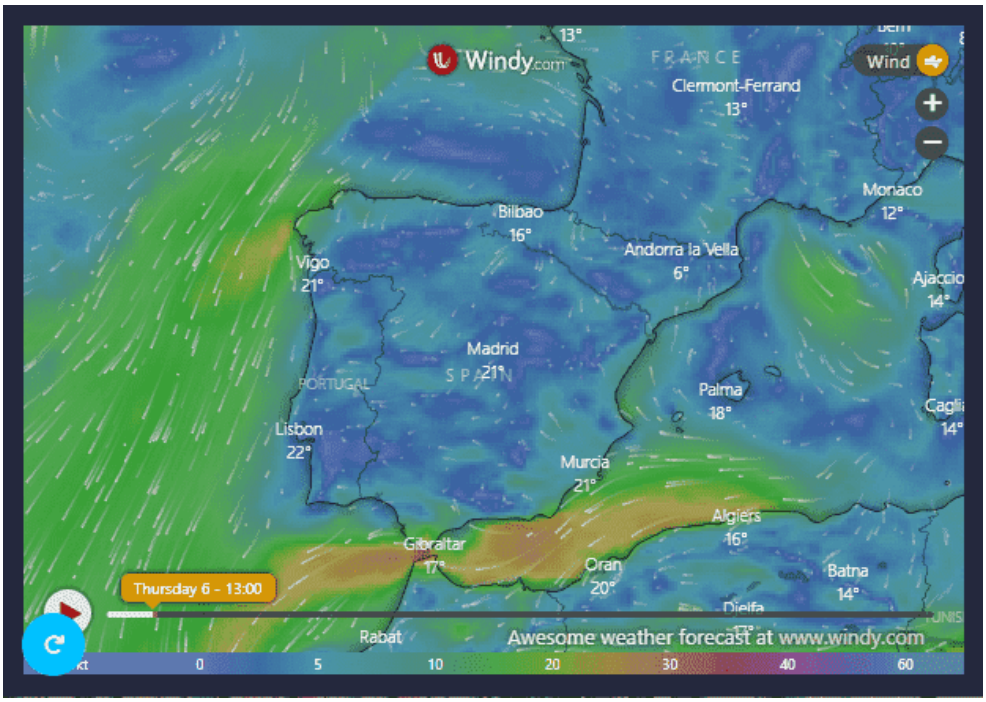


Iframe example 1

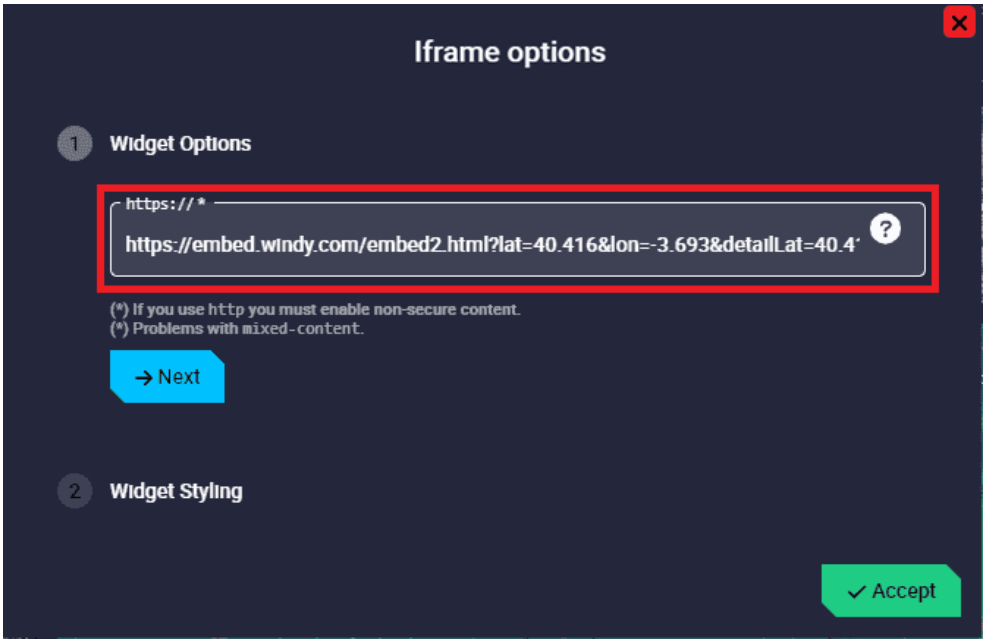


Iframe configuration example 1

- **Example 2**:



Iframe example 2



Iframe configuration example 2

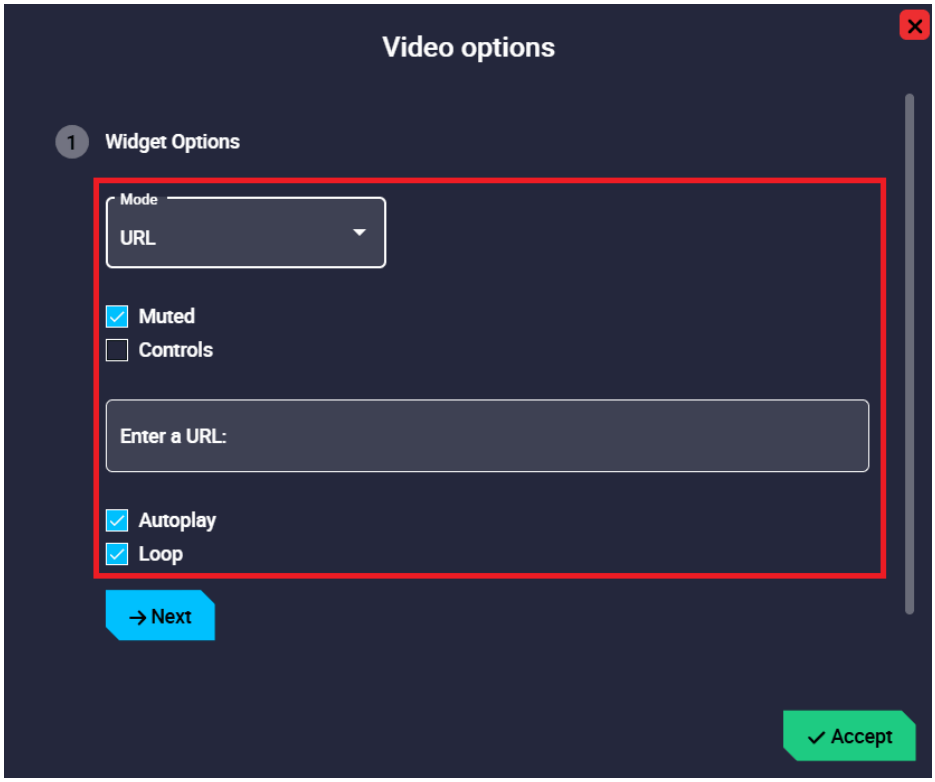
Video

Video widget allows the user to insert a video in the workspace.

Video

Click the button to access its options (**Duplicate**, **Edit** and **Remove**).

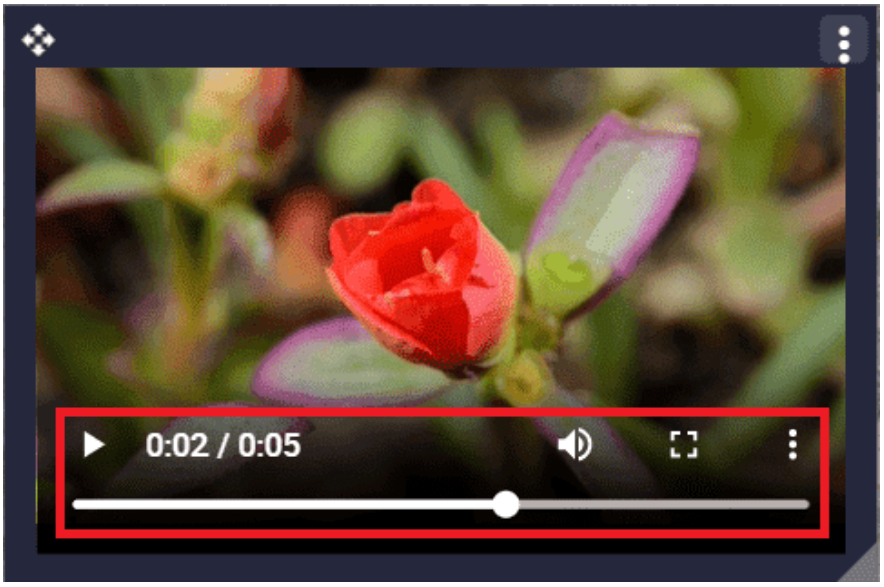
- **Options → Edit**: This allows the user to access the Video configuration menu.
  - **Widget Options**:



Video configuration

- **Mode**: Choos between URL, SCREEN, WEBCAM.

- **Muted**: The video can be muted/unmuted. By default it is enabled.
- **Controls**: Video controls can be added to the widget. Controls such as play/pause, mute/unmute, full screen, etc. By default it is disabled.



Video configuration - Controls

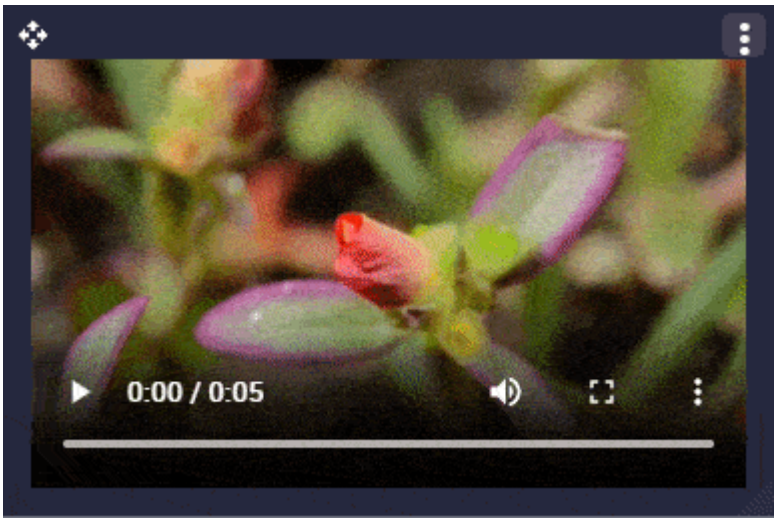
- **Enter a URL:** Users can enter the URL of the video to be displayed.

> ⚠ **Caution**
>
> If **URL mode** has been selected, the video must be in `.mp4` , `.ogg`, `.webm` or `.avi` format.

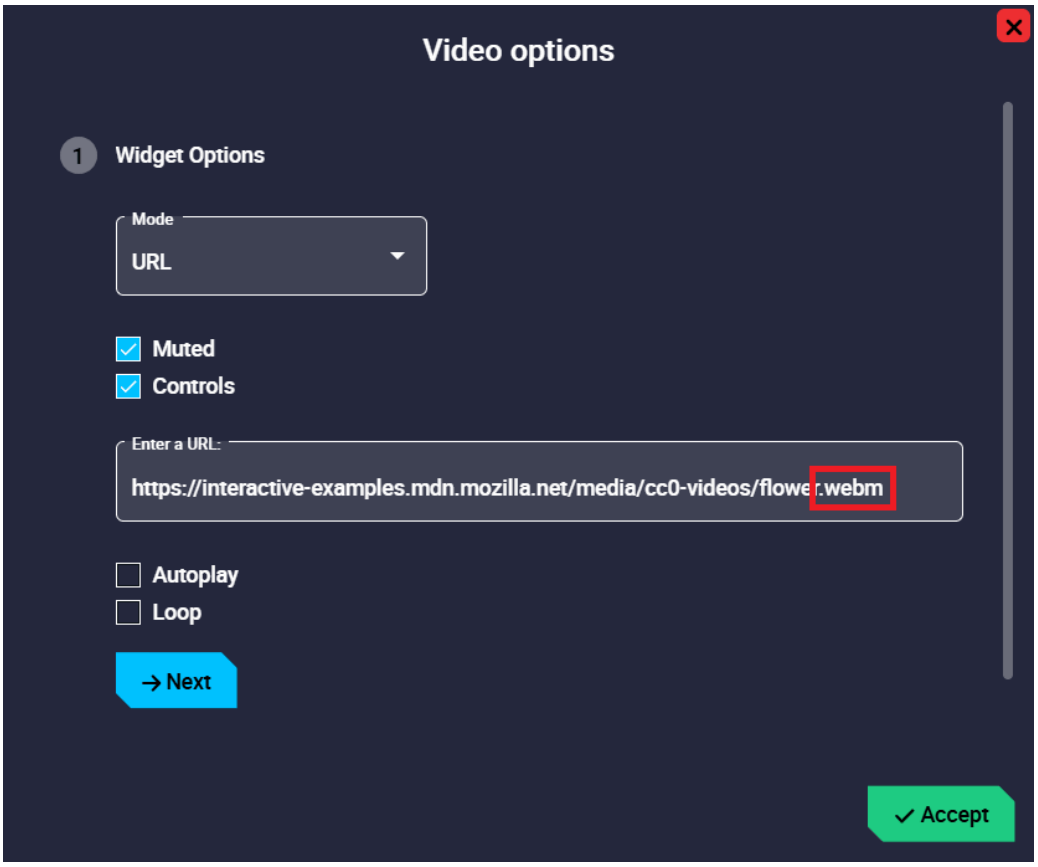
- **Autoplay**: The video plays automatically when the user enters its URL. By default it is enabled.
- **Loop**: Video plays continuously, like a gif. By default it is enabled.

◦ **Widget Styling**: This widget has the 'basic' Widget Styling configuration described in the Widgets common configuration.

An example is given below:



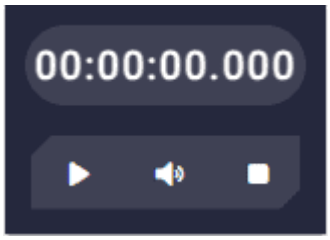
Video example



Video configuration example

Timer

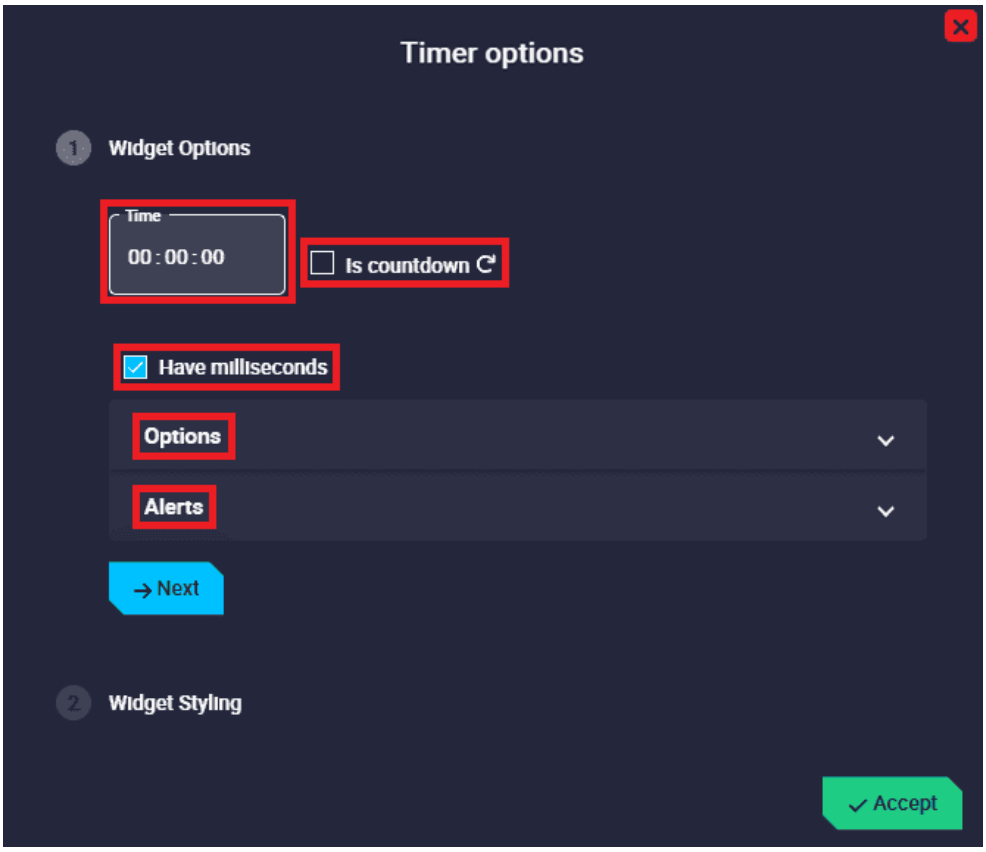
This widget is a counter, which can also function as a countdown counter if configured as such.



Timer

Right click on the **widget** to access its options (**Duplicate**, **Edit** and **Remove**).

- **Options → Edit**: This allows the user to access the Timer configuration menu.



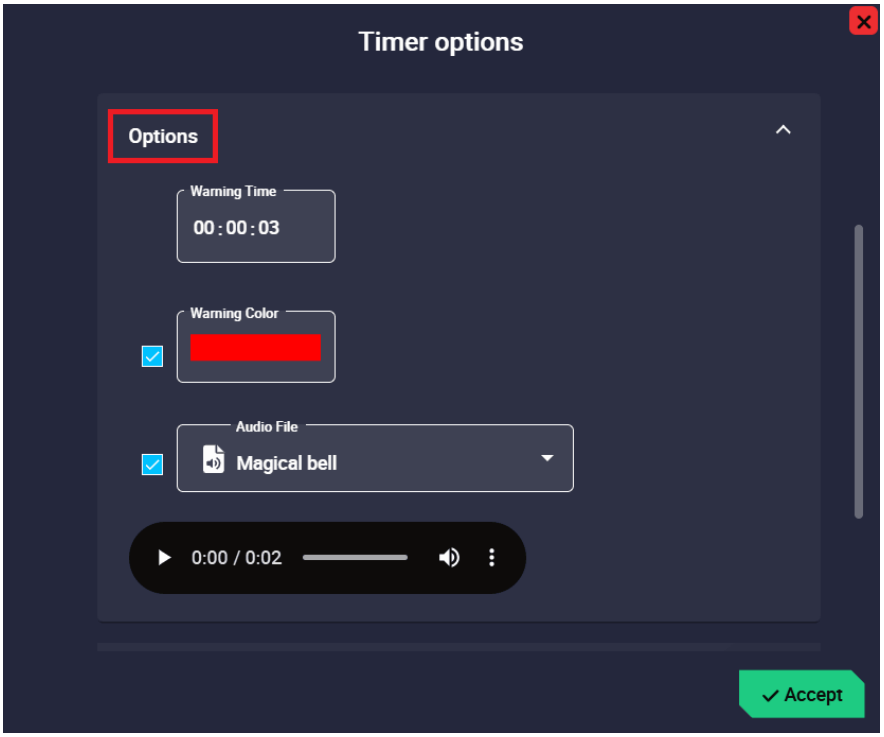
Timer configuration

  - **Widget Options**

    This widget has **different edit parameters** compared to the ones described in the Widgets common configuration.

    - **Time**: Users must enter the desired counter time. Note that if the **Is countdown** checkbox is **enabled**, this becomes the **countdown time**.
    - **Is countdown**: Enabling this option turns the widget into a **countdown timer**. Default is disabled.

- **Have milliseconds**: If enabled, the time will be displayed with milliseconds. By default it is enabled.
- **Options**: Users can set a warning time, color and sound for when the timer reaches the end of the set counter/countdown time.



Timer configuration - Options

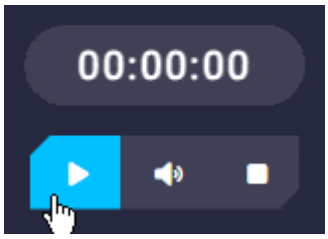
- **Warning Time**: Time for the color and/or sound to be displayed until the end of the counter/countdown time.
- **Warning Color**: If enabled, this color shall be displayed flashing continuously from the time the warning time is reached until the end of the counter/countdown.
- **Audio File**: If enabled, the audio that users have selected from the audio list will play when the warning time is reached, sounding continuously until the end of the counter/countdown.

  For example, if the audio file has a duration of 2 seconds and the warning time is set to 3 seconds, the audio will start playing for the first time from 3 seconds left until 1 second left and then play for the second time from 1 second left until the end.

  In other words, the audio will sound 1.5 times.

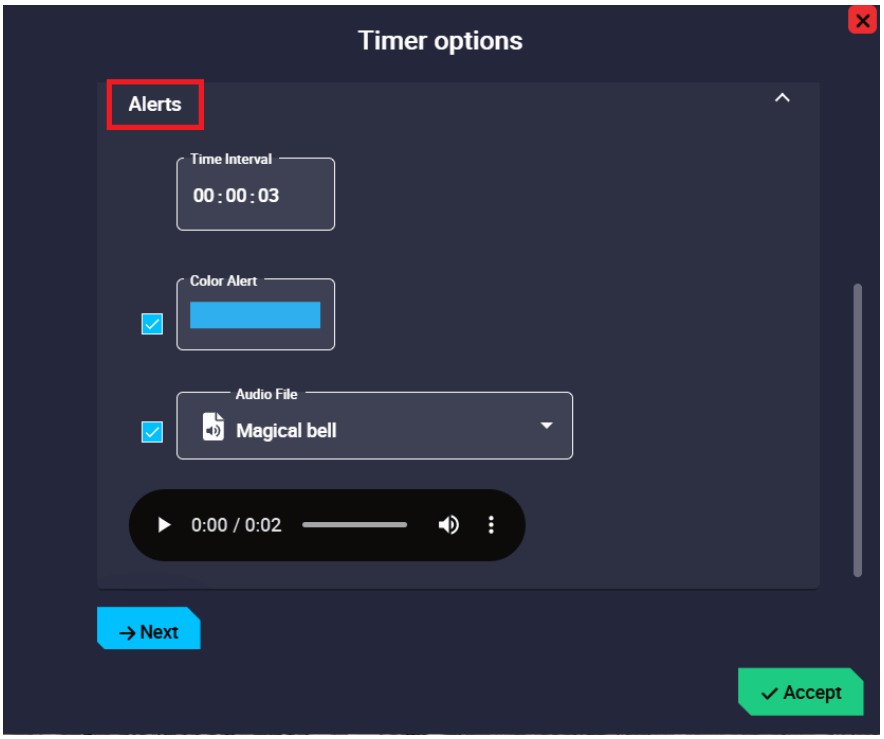
> **Important**
> If **Warning Color** and **Audio File** parameters are disabled, the warnings will be disabled.

For example, if the timer is configured as a **countdown** with a time of 8 seconds and the warning time is set to 3 seconds, the widget will be displayed with the color and/or sound defined here each of those final 3 seconds, i.e. from 3 to 0 seconds:



Timer - Countdown example

- **Alerts**: Light and/or sound alerts can be activated every certain specified time.



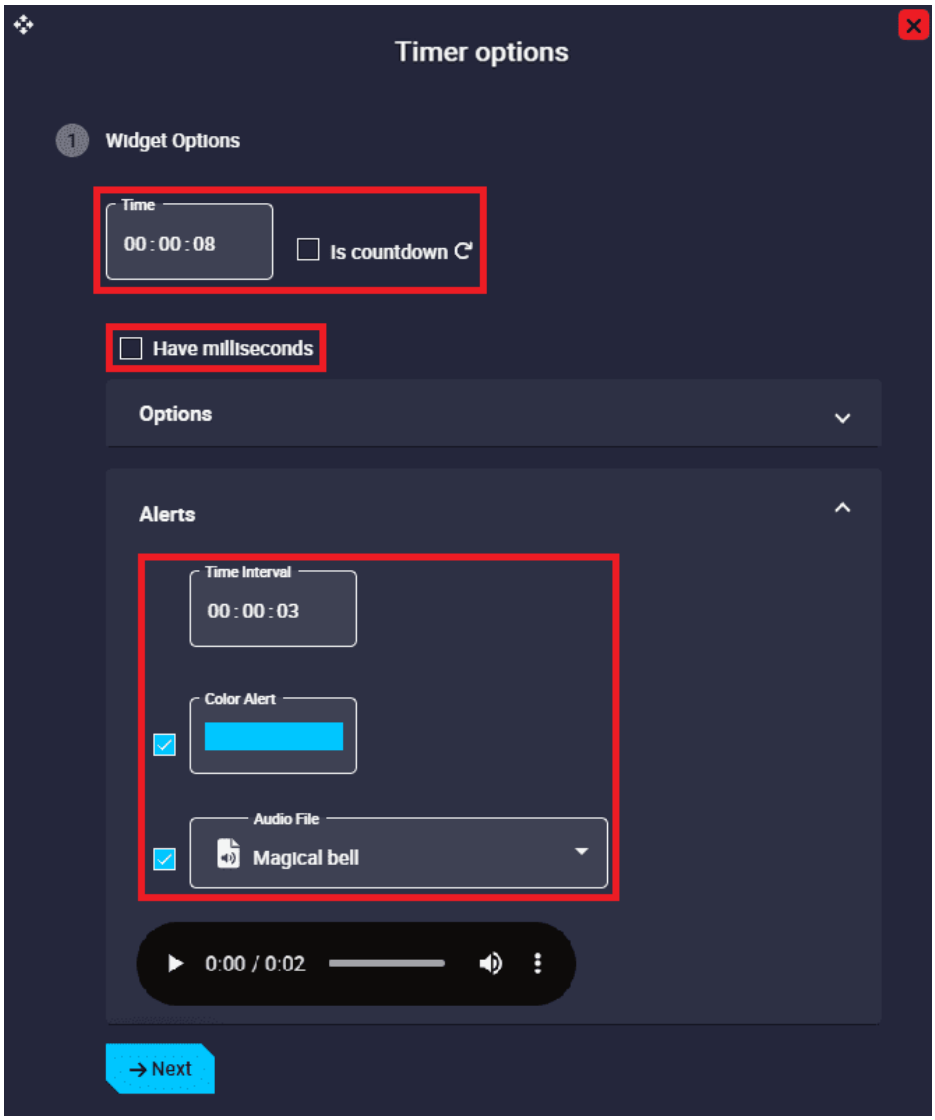
Timer configuration - Alerts

  - **Time Interval**: The time interval for the light and sound alerts is set here.
  - **Color Alert**: If enabled, the timer will have a light alert with the color specified here when the time interval elapses.

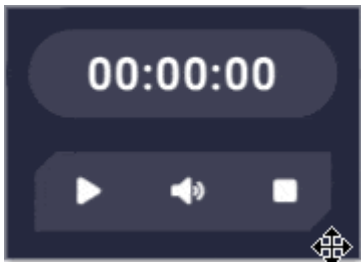
- **Audio file**: If enabled, the timer will have an audible alert with the audio specified here when the time interval elapses. Users can select the audio of the alert from the audio list.

> **Important**
> If **Color Alert** and **Audio File** parameters are disabled, the alerts will be disabled.

An example is given below:



Timer - Alert example



Timer - Alert example

Telemetry

Label

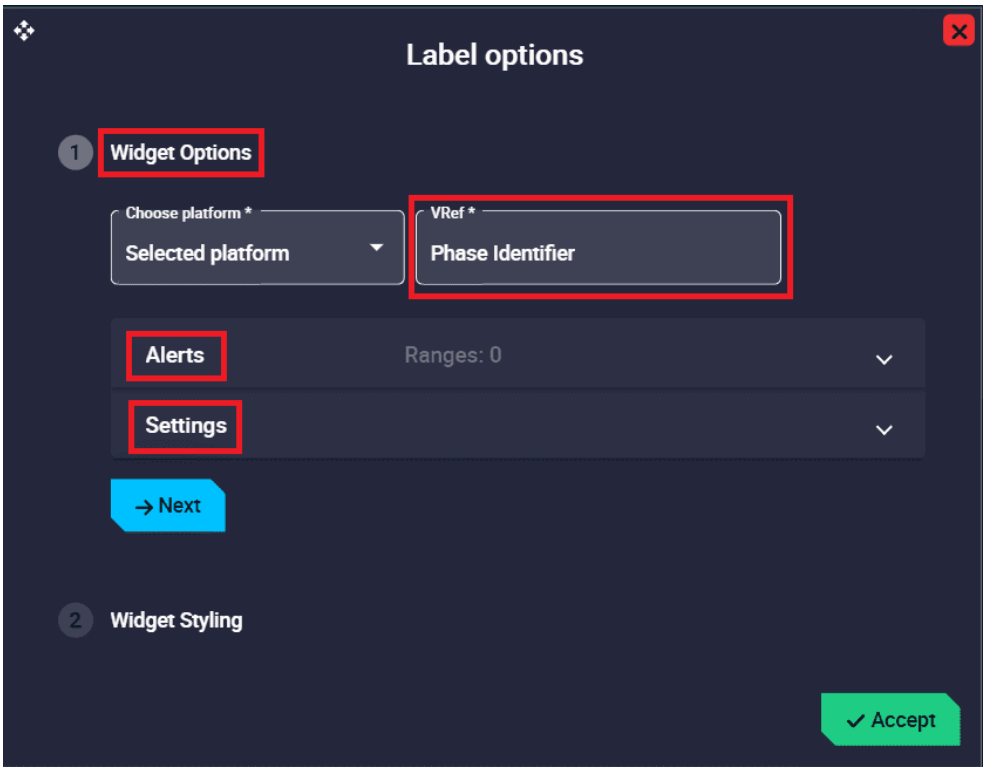
Depending on the type of variable selected, the appearance of the label (by default) is different.

Label

1. **Integer variables**: Only the variable with its value is displayed.
2. **Real variables**: The units of the variable are displayed.
3. **Bit variables**: These labels have failure/success color.

Right click on the **labels** to access their options (**Duplicate**, **Edit** and **Remove**):

- **Options** → **Edit**: This allows the user to access the Label configuration menu.



Label configuration

- **Widget Options**

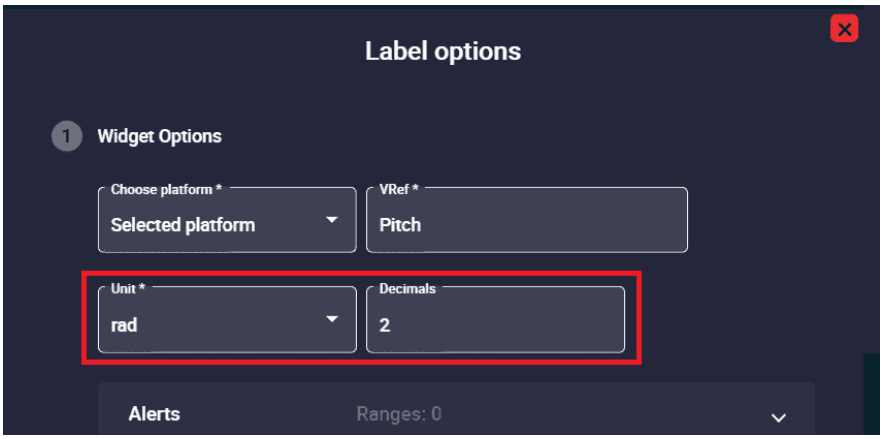
  This widget has **extra edit parameters** compared to the ones described in the Widgets common configuration.

  - **VRef**: Users can choose any real, integer or bit variable to be displayed on the label.
  - **Unit**: Unit of measurement of the displayed variable.

    > ⓘ **Note**
    > This parameter is only available with Real variables.

  - **Decimals**: Number of decimals displayed for the selected variable.

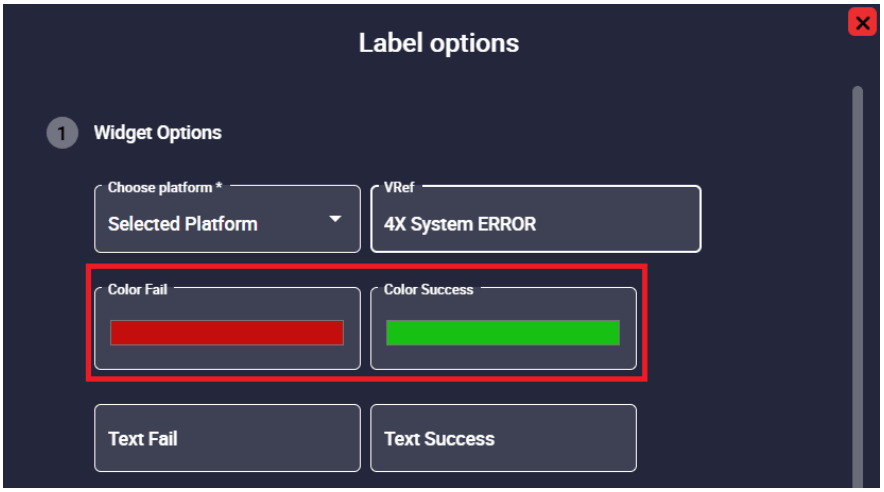
    > ⓘ **Note**
    > This parameter is only available with Real variables.



Real label configuration

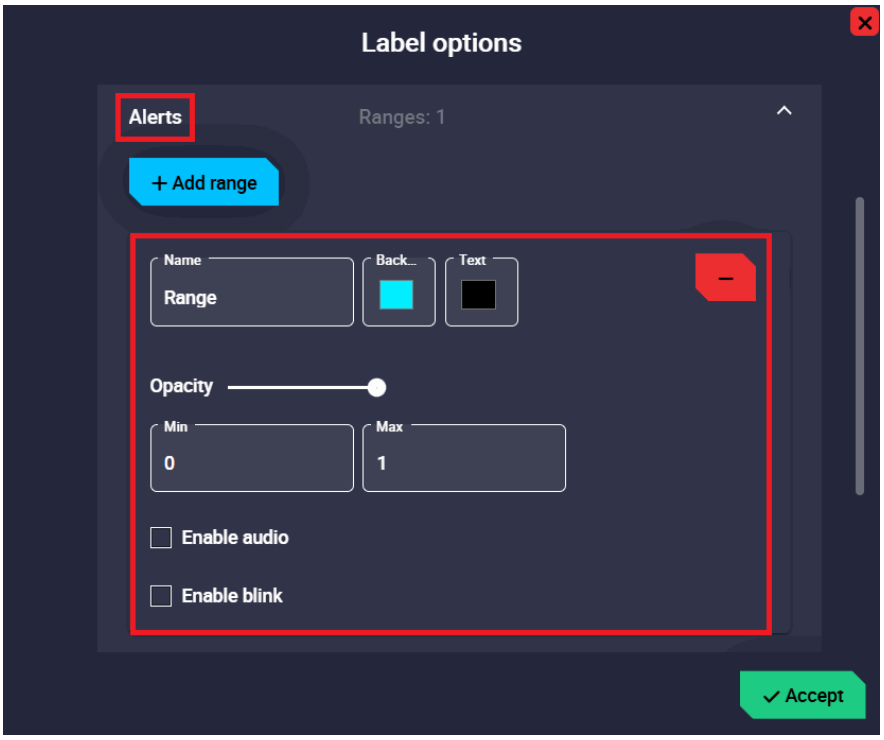
- **Color Fail/Success**: By default, when the bit variable is in 'fail mode', the label is red and when it is in 'success mode', it is green.

> ⓘ **Note**
>
> This is only available with Bit variables.



Bit label configuration

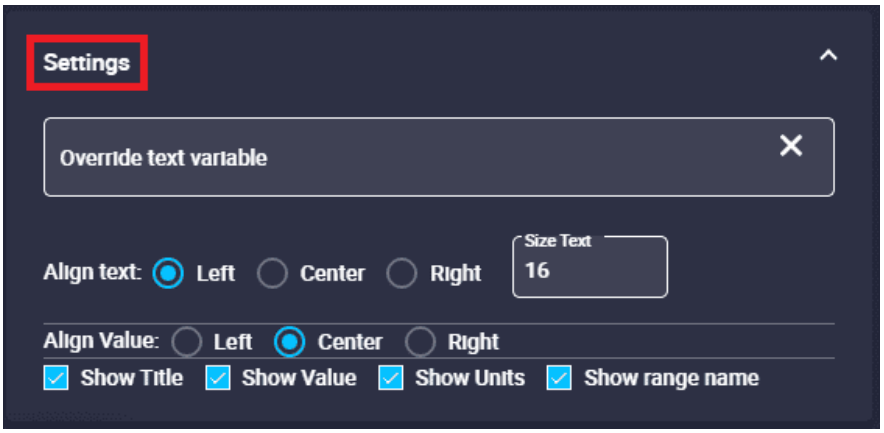
- **Alerts**: Allows the user to set different ranges. For this range several customizations can be made:



Label configuration - Alerts

- **Name**: An indentifier name of the range can be set.
- **Background color**: Changes the color of the label when the variable is in this range.
- **Text color**: Changes the color of the label text when the variable is in this range.
- **Opacity**: The opacity of the label can be customized in this interval.
- **Min/Max**: Defines the minimum and maximum values of the range.
- **Remove**: Click to delete this interval.
- **Enable Audio**: Activates acoustic alerts and voice notifications to monitor the aircraft's status.
- **Enable Blink**: Enables the visual flashing of the label on the map to identify the aircraft during specific events or alarms.

- **Settings**: The 'shape' of the displayed text can be edited:



Label configuration - Settings

- **Override text variable**: The label name can be changed by typing a new one different from the variable name.

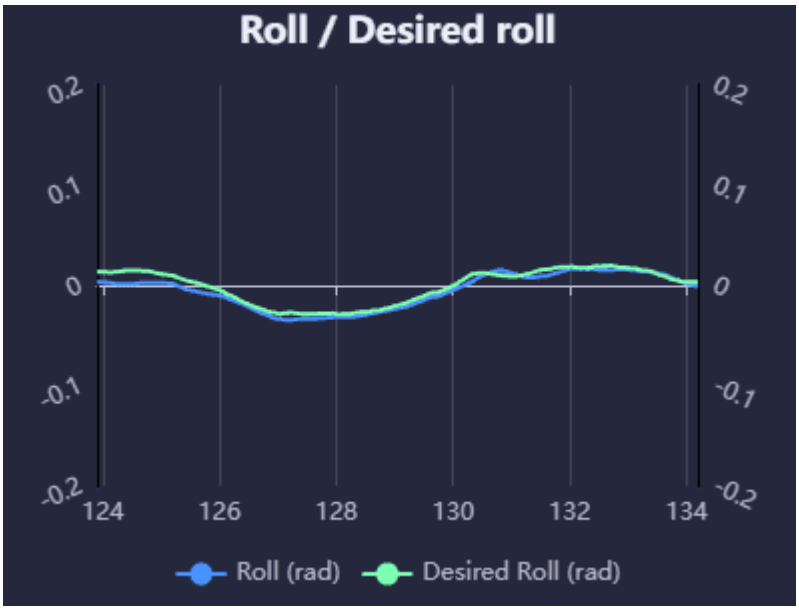
  > ⚠ **Warning**
  > This does not rename the selected variable in the configuration, it is only the name shown in the label.

- **Align text**: The text of the variable can be left, center or right aligned.
- **Size text**.
- **Align Value**: The value of the variable can be left, center or right aligned.
- **Show Title/Value/Units**: The title, value and units of the variable can be shown/hidden.
- **Show range name**: If enabled, the name assigned to the range will appear next to the variable.

Chart

Charts widgets allow the user to represent any variable of the system with respect to time. Thus, the user can visualize the evolution of that variable over time.

It is very useful during flight tests (gain tuning) and operations to monitor the behavior of the platform.



Chart

Right click on the **chart** to access its options (**Duplicate**, **Edit** and **Remove**).

- **Options → Edit**: This allows the user to access the Chart configuration menu.

Chart configuration

  - **Widget Options**

This widget has **extra edit parameters** compared to the ones described in the Widgets common configuration.

- **Title**: Title of the chart.
- **Maximum Sample Size**: Maximum number of samples of the selected variable drawn in the chart. By default it is set to 100.
- **Number of series**: Number of variables drawn in the chart. Users can add as many series as they wish to represent in the same chart.

  The appearance of this series in the chart can be configured below:

Chart configuration - Series

  - **VRef**: Users can choose any real, integer or bit variable to be displayed on the chart.
  - **Unit Y**: Unit of measurement of the displayed variable.

    > ⓘ **Note**
    > This parameter is only available with Real variables.

  - **Decimals**: Number of decimals displayed for the selected variable.

    > ⓘ **Note**
    > This parameter is only available with Real variables.

- **Color**: This is the color in which the line with the values of the series is drawn.
  By default, the first series is drawn in blue, the second in green, the third in yello, the fourth in red, etc.
- **Auto Min/Max Y**: If enabled, minimum and maximum of the Y axis of the chart are automatically adjusted.
- **Min/Max Y**: Minimum and maximum Y axis of the chart can be adjusted manually.

- **Interface settings**:

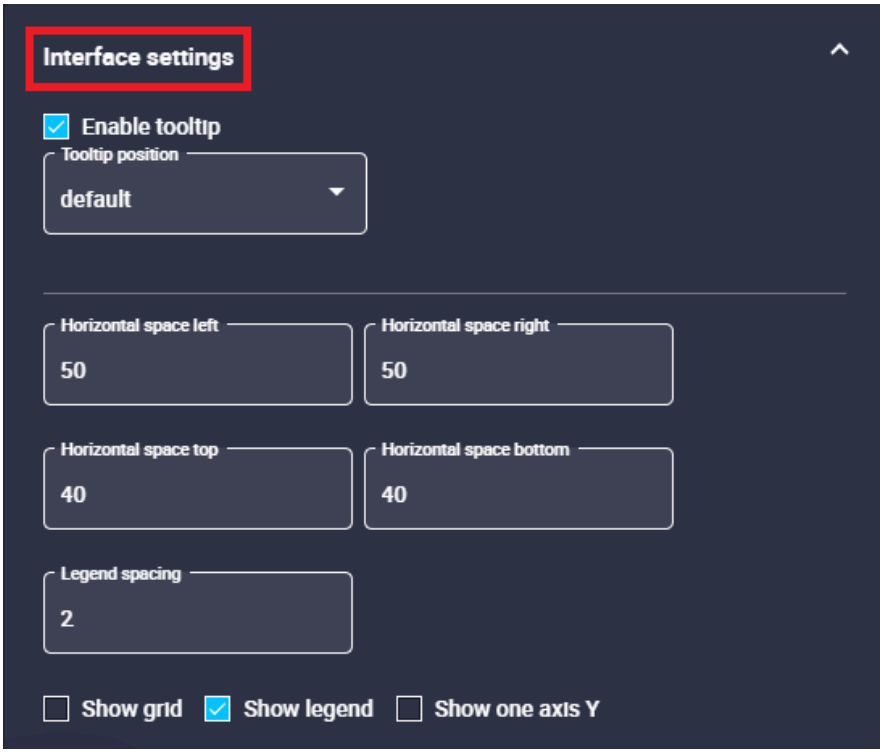


Chart configuration - Interface settings

- **Tooltip position**: When users hover the mouse over the series, a tooltip appears with the exact value of the selected variable at that instant of time. It is then possible to define where this tooltip is displayed in the chart, the available positions are:
  - default: The tooltip position is set by default by the chart.
  - top: The tooltip is displayed on the top center of the chart.
  - left: The tooltip is displayed in the center left of the chart.
  - right: The tooltip is displayed in the center right part of the chart.

- bottom: The tooltip is displayed in the bottom center of the chart.

Below is an example when the right position is selected:

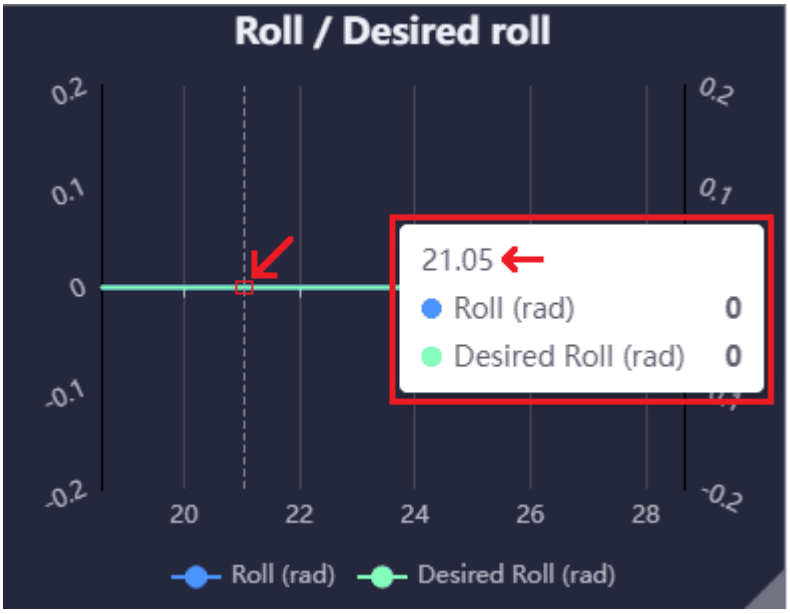


Chart configuration - Tooltip position

- **Horizontal space left/right**: The horizontal space left/right of the chart background can be customized. By default they are set to 50.
- **Show grid**: The grid of the chart can be shown/hidden.
- **Show legend**: The legend of the series can be shown/hidden on the chart.
- **Show one axis Y**: When more than one series is added, it is possible to show only one Y axis by enabling this option.

The following is an example of a typical chart used during operation:

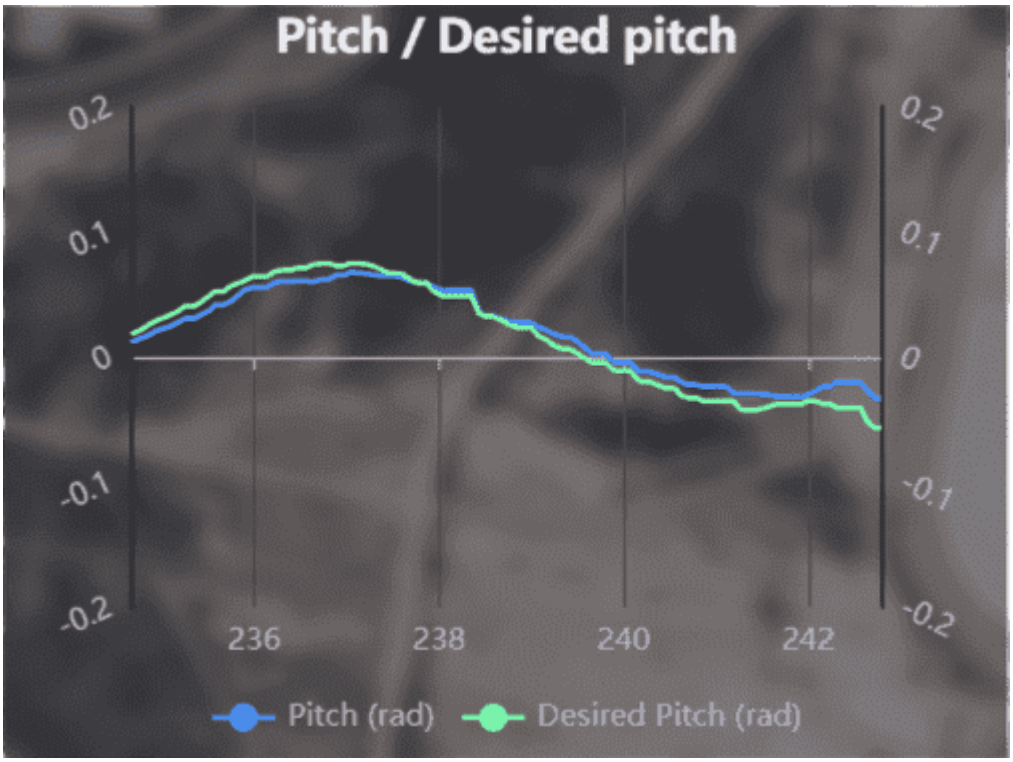


Chart example

The custom configuration made for this chart is as follows:

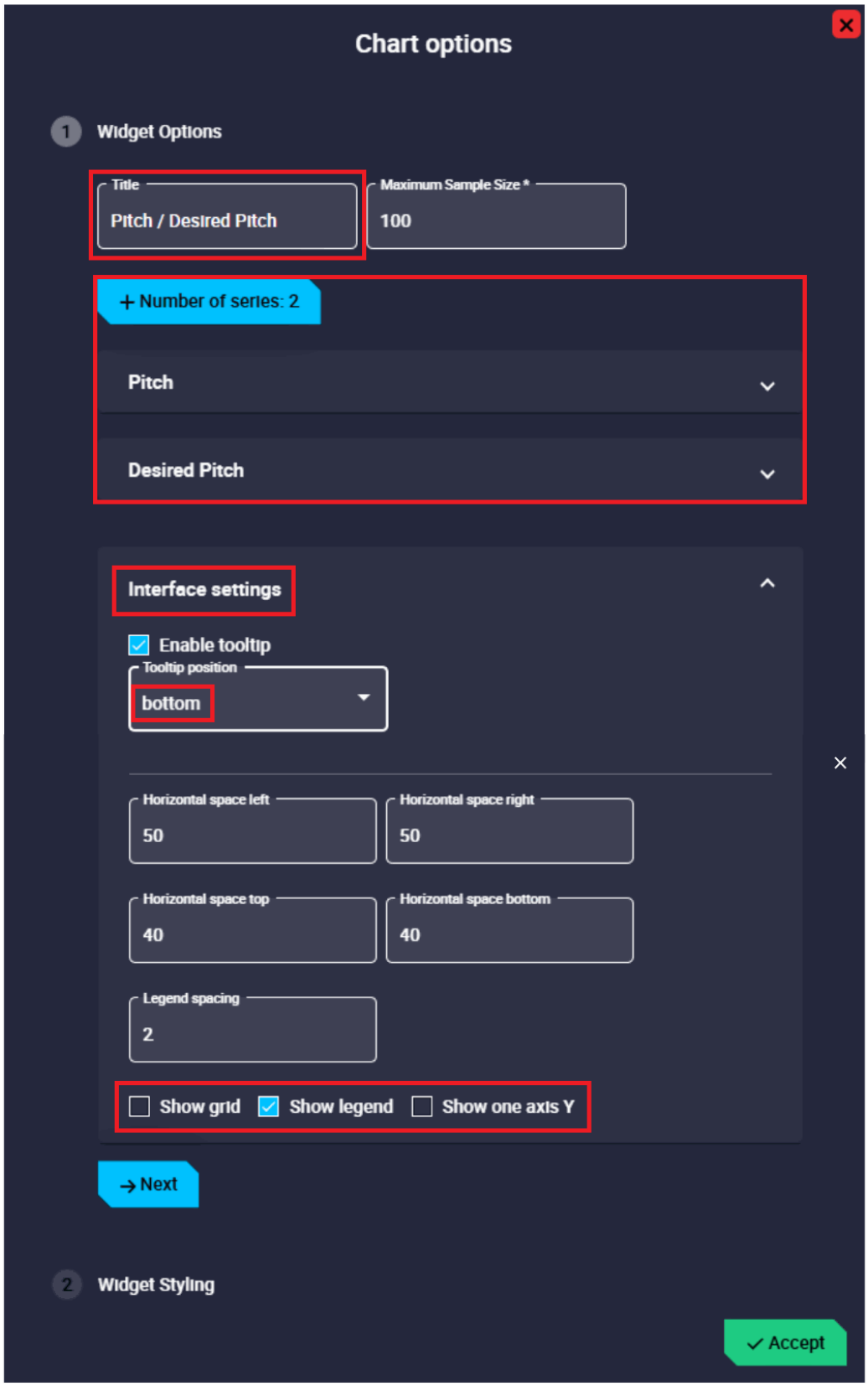


Chart configuration - Widget Options example

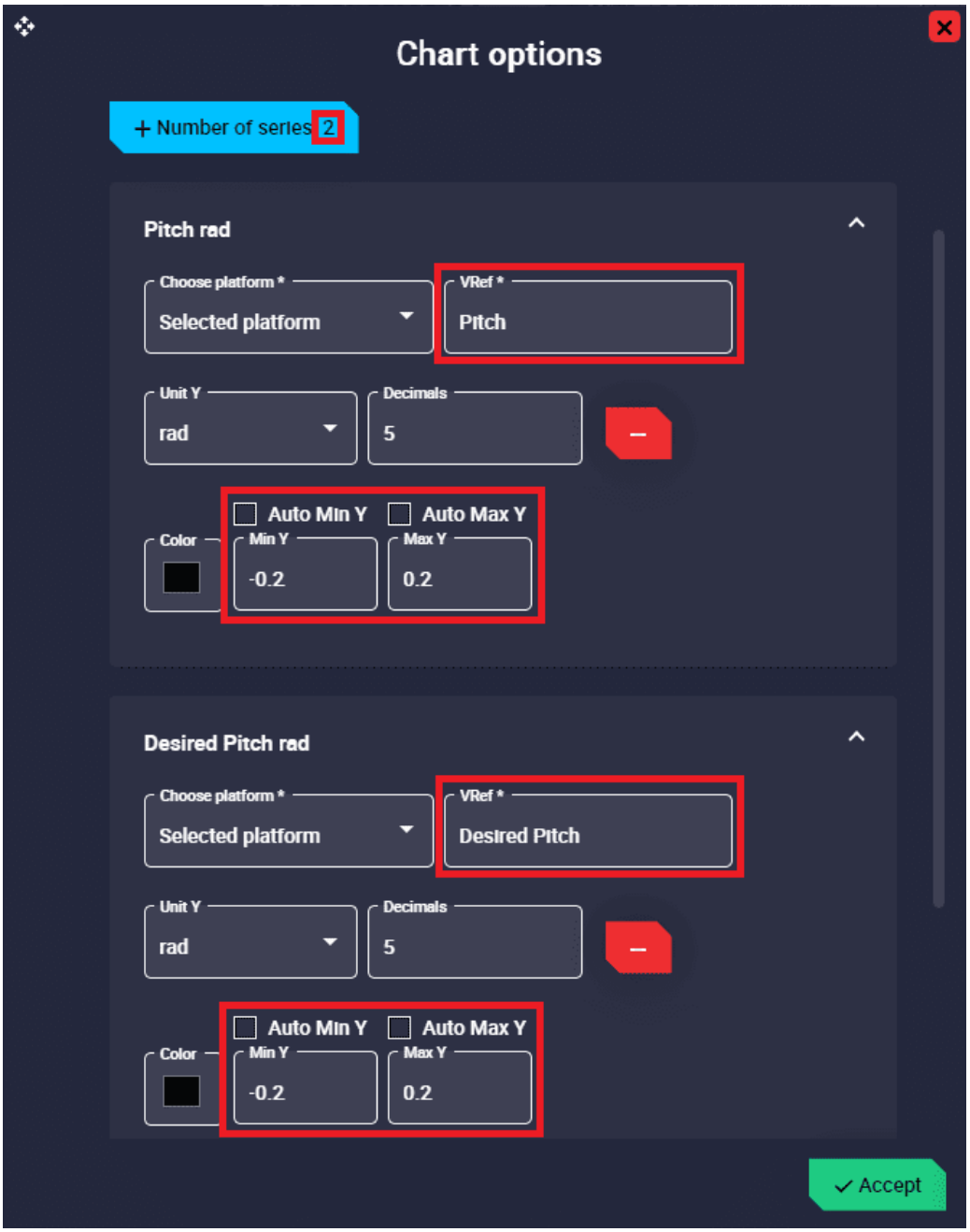


Chart configuration - Series example

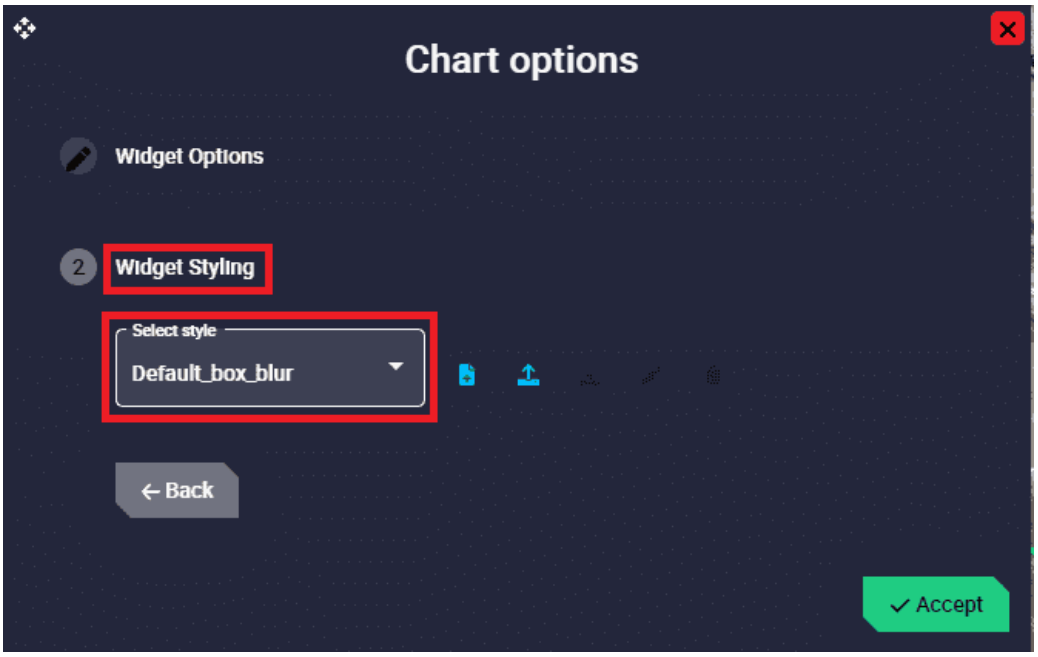
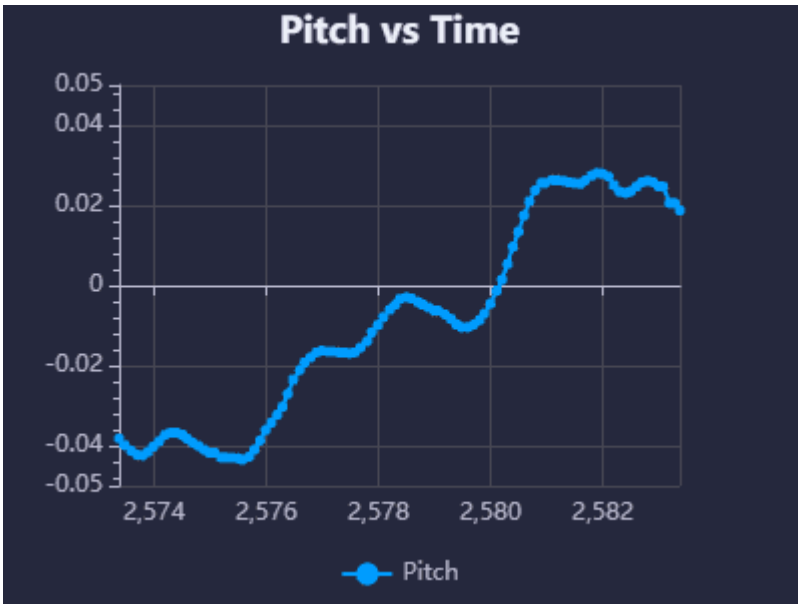


Chart configuration - Widget Styling example

Scatter

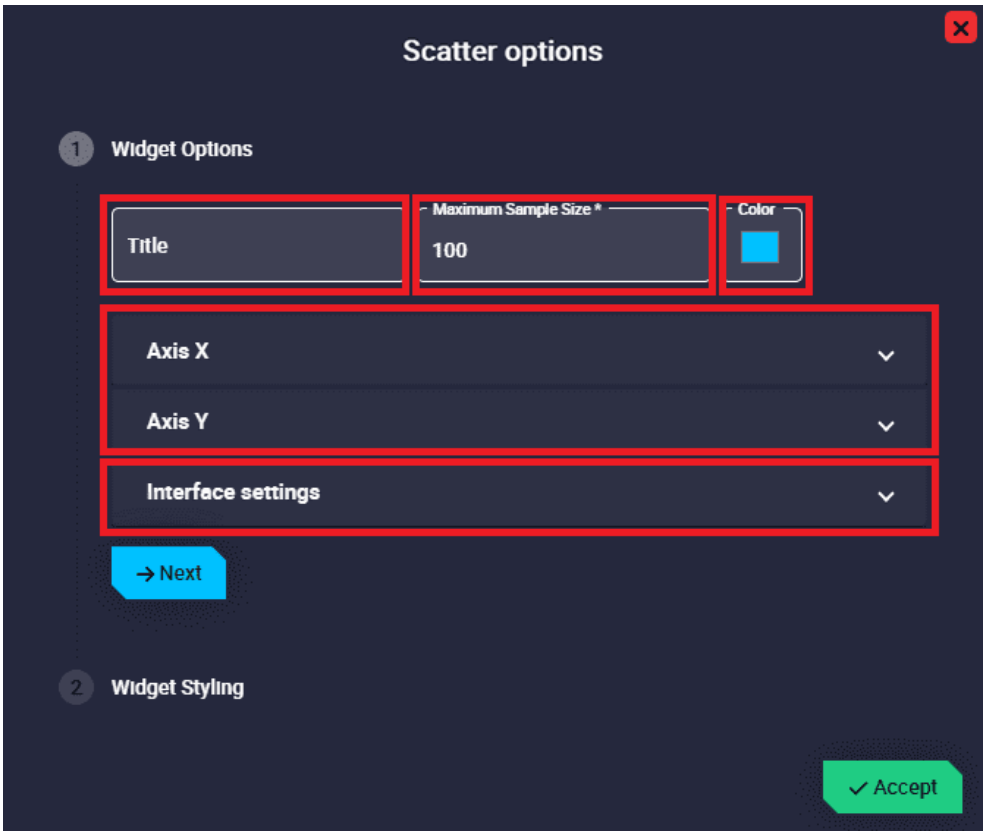
Scatter widget allows the user to represent any variable in the system with respect to another variable.



Scatter

Right click on the **scatter** to access its options (**Duplicate**, **Edit** and **Remove**).

- **Options → Edit**: This allows the user to access the Scatter configuration menu.



Scatter configuration

- **Widget Options**

  This widget has **extra edit parameters** compared to the ones described in the Widgets common configuration.

  - **Title**: Title of the scatter.
  - **Maximum Sample Size**: Maximum number of samples of the selected variables that are drawn in the scatter. By default it is set to 100.
  - **Color**: This is the color in which the values are drawn.
  - **Axis X/Y**: The user must select the variables to be plotted on both axes.

Scatter configuration - Axis X/Y

The appearance of the variables in the scatter can then be modified:

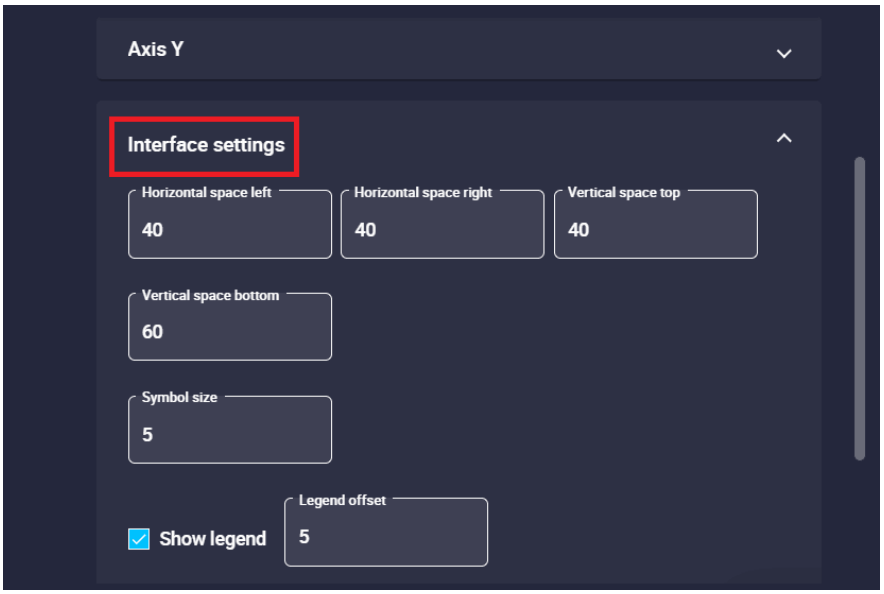
- **Auto Min/Max X/Y**: If enabled, the minimum and maximum of the X/Y axes of the scatter are automatically adjusted.
- **Min/Max X/Y**: The minimum and maximum X/Y axes of the scatter can be adjusted manually.
- **VRef**: Users can choose any real, integer or bit variables to be displayed in the scatter.
- **Unit**: Unit of measurement of the displayed variable.

  > ⓘ **Note**
  > This parameter is only available with Real variables.

- **Decimals**: Number of decimals displayed for the selected variable.

  > ⓘ **Note**
  > This parameter is only available with Real variables.

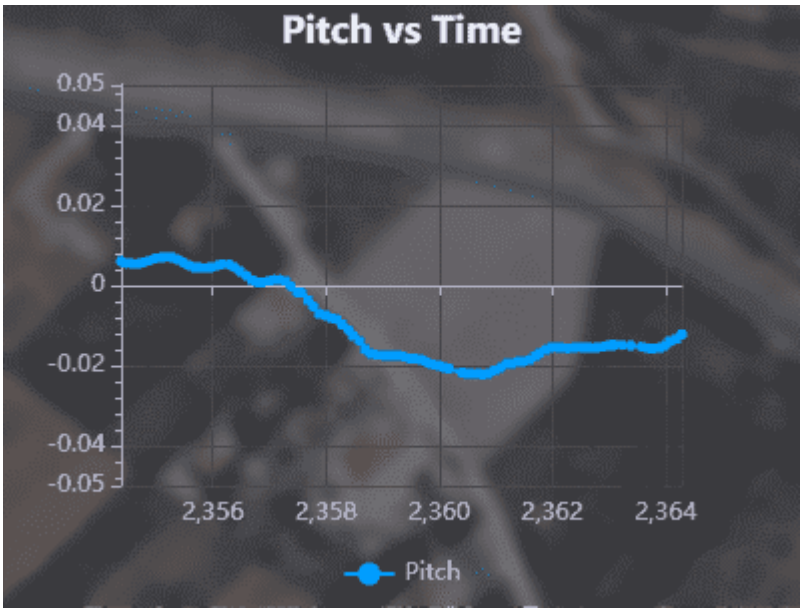
- **Interface settings**:



Scatter configuration - Interface settings

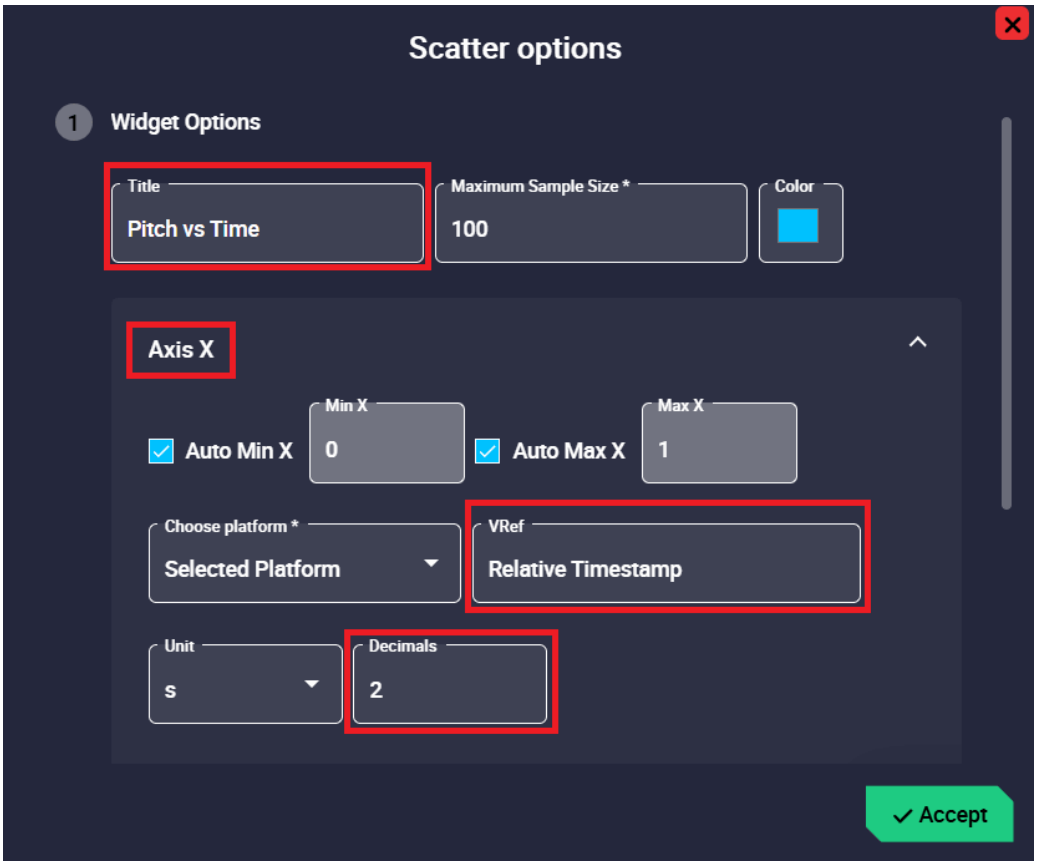
- **Horizontal space left/right**: The left/right horizontal spacing of the scatter background can be customized. By default they are set to 40.
- **Vertical space top/bottom**: The top/bottom vertical spacing of the scatter background can be customized. By default they are set to 40.
- **Symbol size**: The size of the symbol in the scatter (points) can be modified by the user. Default is set to 5.
- **Show legend**: The legend of the variable on the Y axis can be shown/hidden in the scatter.
- **Legend offset**: Vertical position of the legend relative to the bottom edge of the scatter, where 0 matches the bottom edge of the scatter.
  By default it is set to 5.

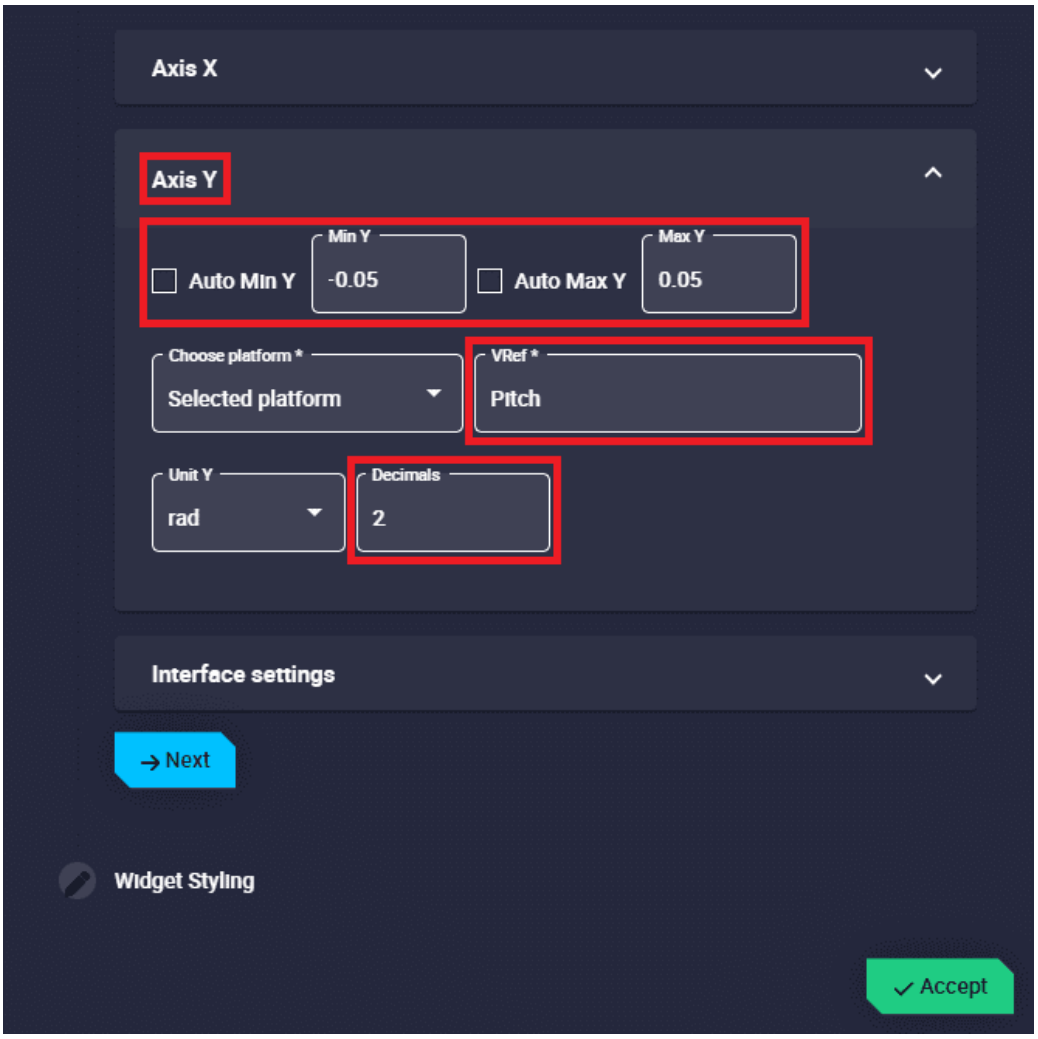
An example is given below:

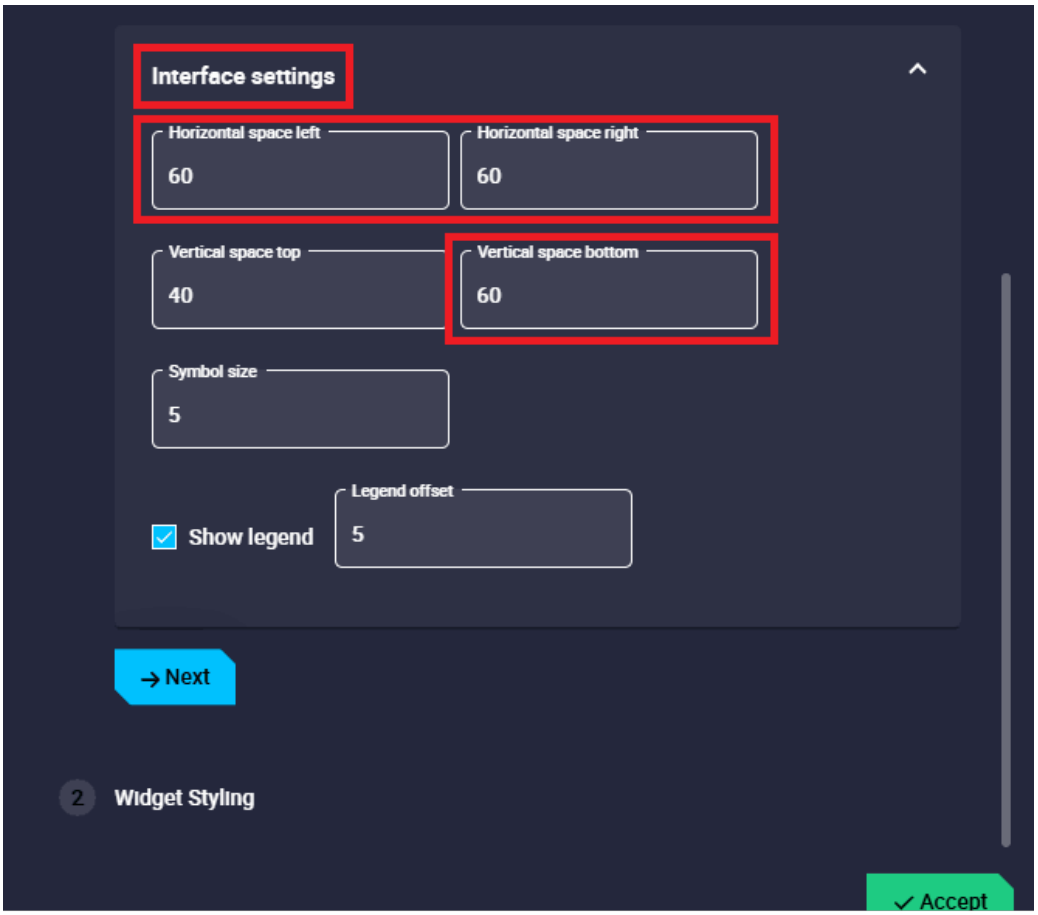


Scatter example

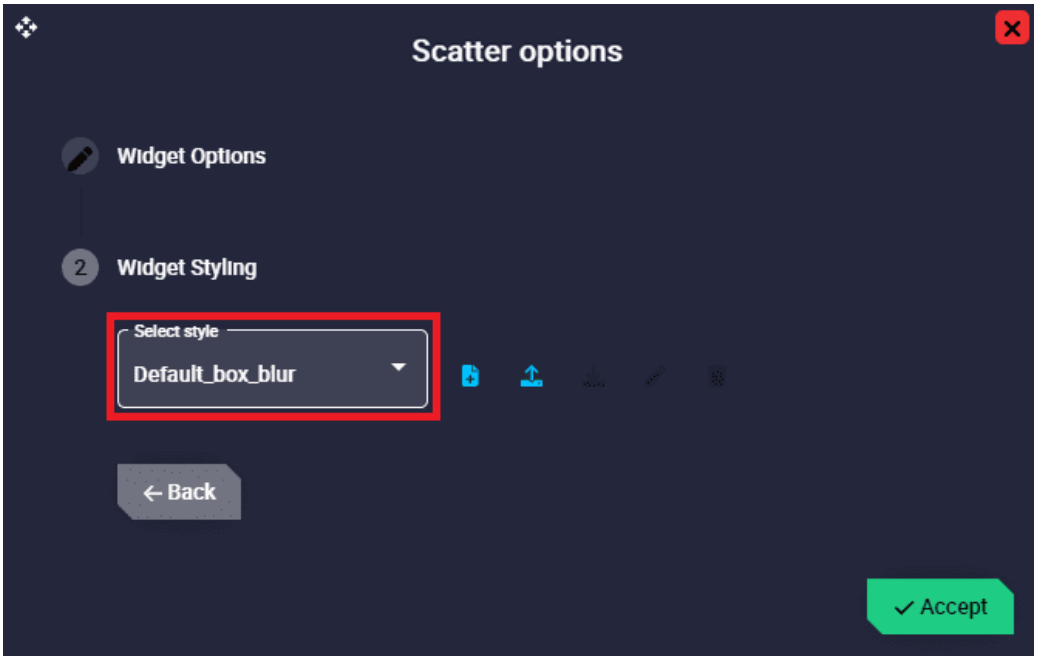
The custom configuration made for this scatter is as follows:




Axis X
Axis Y
Auto Min Y
Min Y
-0.05
Auto Max Y
Max Y
0.05
Choose platform *
Selected platform
VRef *
Pitch
Unit Y
rad
Decimals
2
Interface settings
Next
Widget Styling
Accept



Scatter configuration - Widget Options example



Scatter configuration - Widget Styling example

Gauge

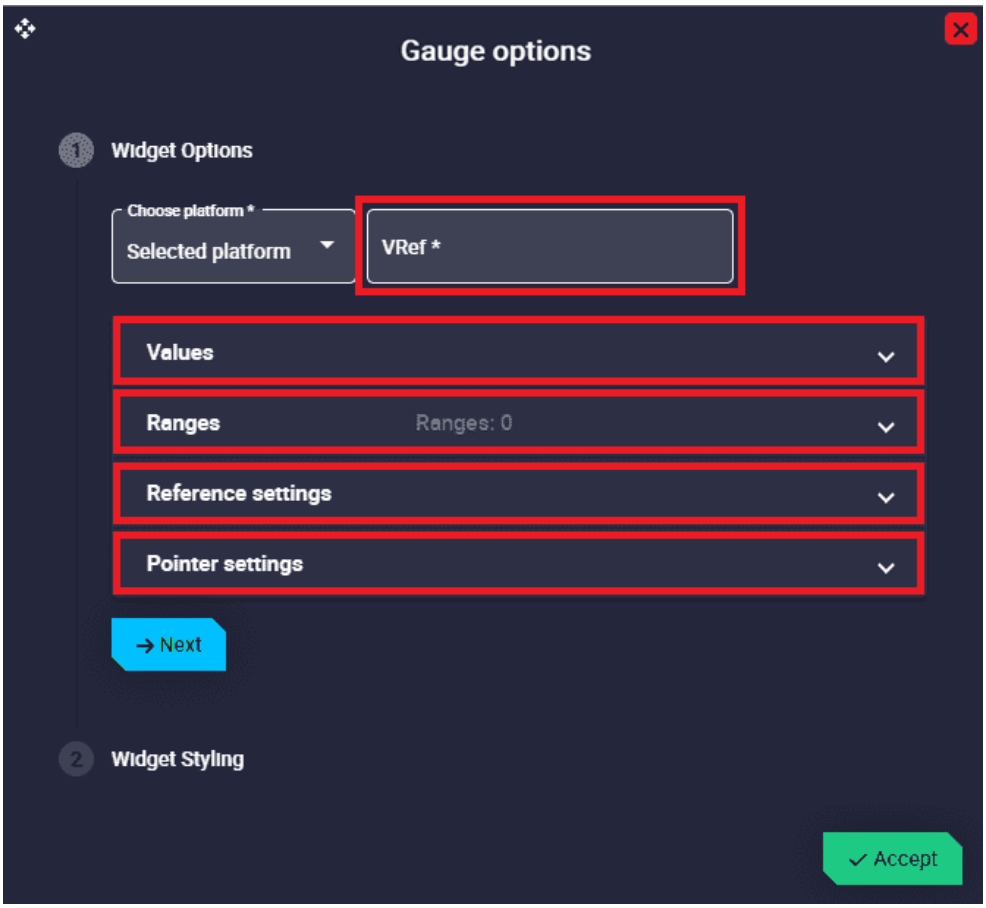
Gauge widget is a data visualisation tool that can be used to show the progress of data or display data in ranges in a precise and compact area.

This allows to display the progress of measuremetns in a circular arc or to see these numerical values in a range of colors that can have a certain meaning for the user.

Gauge

Right click on the **gauge** to access its options (**Duplicate**, **Edit** and **Remove**).

- **Options → Edit**: This allows the user to access the Gauge configuration menu.



Gauge configuration

- **Widget Options**

  This widget has **extra edit parameters** compared to the ones described in the Widgets common configuration.

  - **VRef**: Users can choose any real, integer or bit variable to be displayed on the gauge.
  - **Unit**: Unit of measurement of the displayed variable.

    > ⓘ **Note**
    > This parameter is only available with Real variables.

  - **Decimals**: Number of decimals displayed for the selected variable.

    > ⓘ **Note**
    > This parameter is only available with Real variables.

  - **Values**:

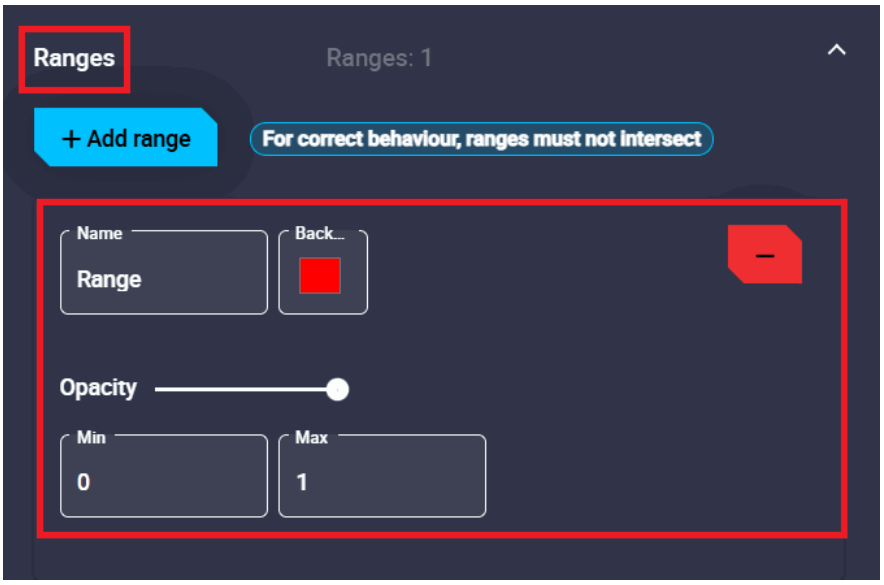
Gauge configuration - Values

- **Min/Max**: Minimum and maximum values displayed on the gauge.
- **Start/End angle**: Start and end angle of the position of the minimum and maximum values respectively.

- **Ranges**: Allows the user to set different ranges.

> ⚠ **Warning**
> The panel warns the user that for correct behavior, ranges must not intersect.

Several customizations can be made in the different ranges:

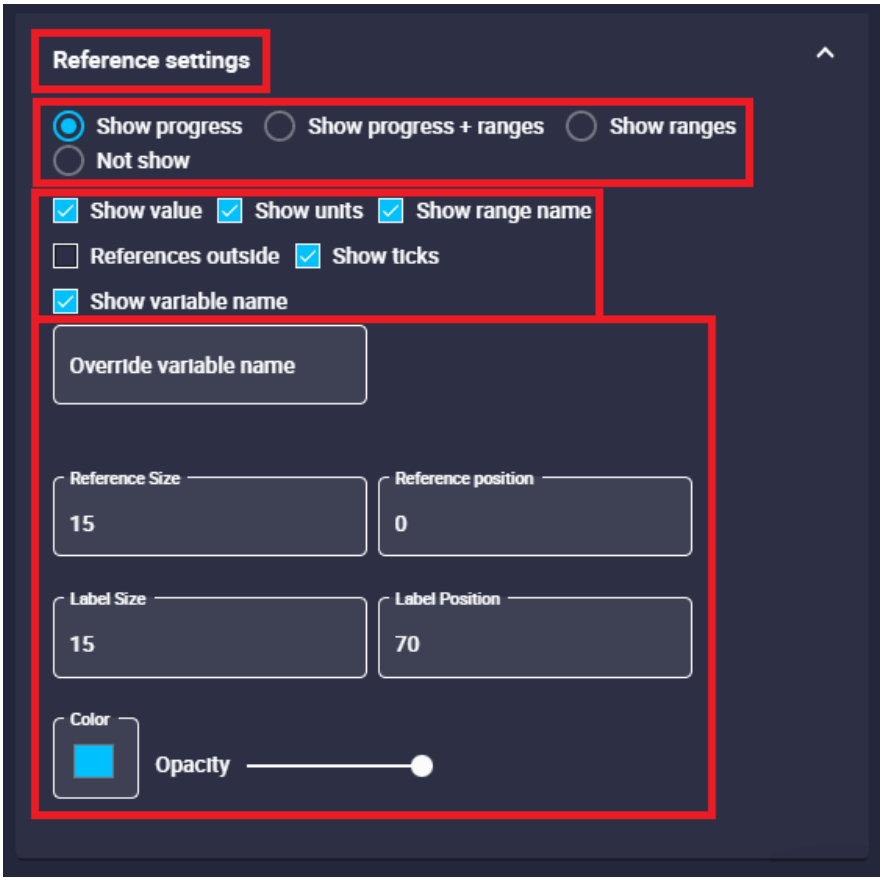


Gauge configuration - Ranges

- **Name**: An identifying name for the range can be set.
- **Background color**: Sets the color of the gauge arc in this range.
- **Opacity**: The opacity of the circular arc can be customized in this range.

- **Min/Max**: Defines the minimum and maximum values of the range.
- **Remove**: Click to delete this range.

- **Reference settings**:



Gauge configuration - Reference settings

The gauge can be configured to show different cases:

- **Show progress**: Shows the progress arc of the selected variable in the gauge.
- **Show progress + ranges**: Displays the progress arc of the selected variable in the gauge together with the previously defined ranges.
- **Show ranges**: Displays the gauge only with the previously defined ranges.
- **Not show**: The needle indicates the value but neither the progress nor the ranges are shown in the gauge.

Reference settings - 'Show progress', 'Show progress + ranges', 'Show ranges' and 'Not show' respectively

> **Important**
> - **Only one** of the four options described below can be selected.
> - If the progress bar falls within a defined range, it will always be colored in the color of that range even when users choose not to display them.

In addition, this widget is highly customizable so that users can configure it to their wishes:

- **Show value**: If enabled, displays on the gauge the value of the selected variable that the needle is pointing to.
- **Show units**: If enabled, displays the units of the value of the selected variable that the needle is pointing to.
- **Show range name**: If enabled, it displays, next to the value, the name of the range (previously defined) in which the value is in.

  In addition, the color of this text will be the same as that of its corresponding range.
- **References outside**: References refers to the displayed axis and its numbering.

  By default, the arc axis numbering is "inside" the arc, by enabling it, it will be in the "outside" part.
- **Show ticks**: Axis ticks can be shown/hidden. By default it is enabled, i.e. they are shown.
- **Override variable name**: Allows for the replacement of the default variable name with a custom label.
- **Reference Size**: The size of the axis numbering can be modified.
- **Reference position**: The position of the axis numbering can be adjusted by the user. The default position is 0.
- **Label Size**: The size of the text value can be modified.

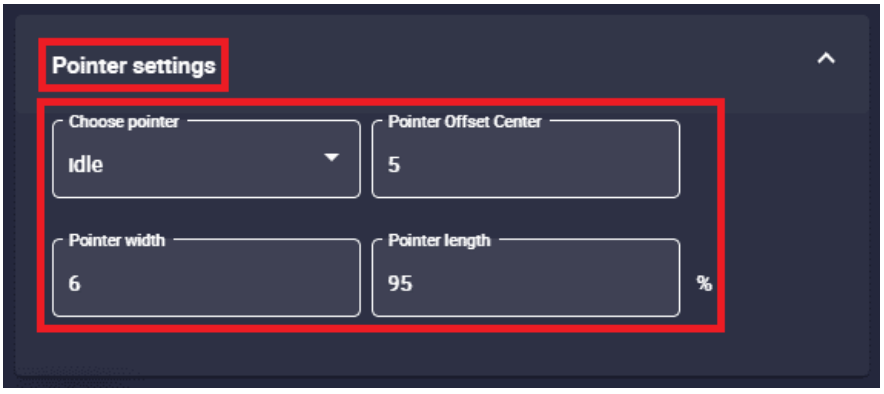
- **Label Position**: The position of the value can be adjusted. The default position is at 70, at the bottom of the gauge.
  If set to 0, the label will be in the center of the gauge.
- **Color**: This is the color in which the value is drawn.

> **Note**
> If **ranges** are defined and the value falls within a range, the color of the value will not be this selected color, but will be colored according to the color of the range the value is in.

- **Opacity**: The opacity of the progress arc can be customized in this range.

> **Tip**
> This is useful when the option Show progress + ranges * has been selected. Since if a range is overpainted by the progress bar, users can still see where the range is.

- **Pointer settings**: Here the user can configure the pointer of the gauge.



Gauge configuration - Pointer settings

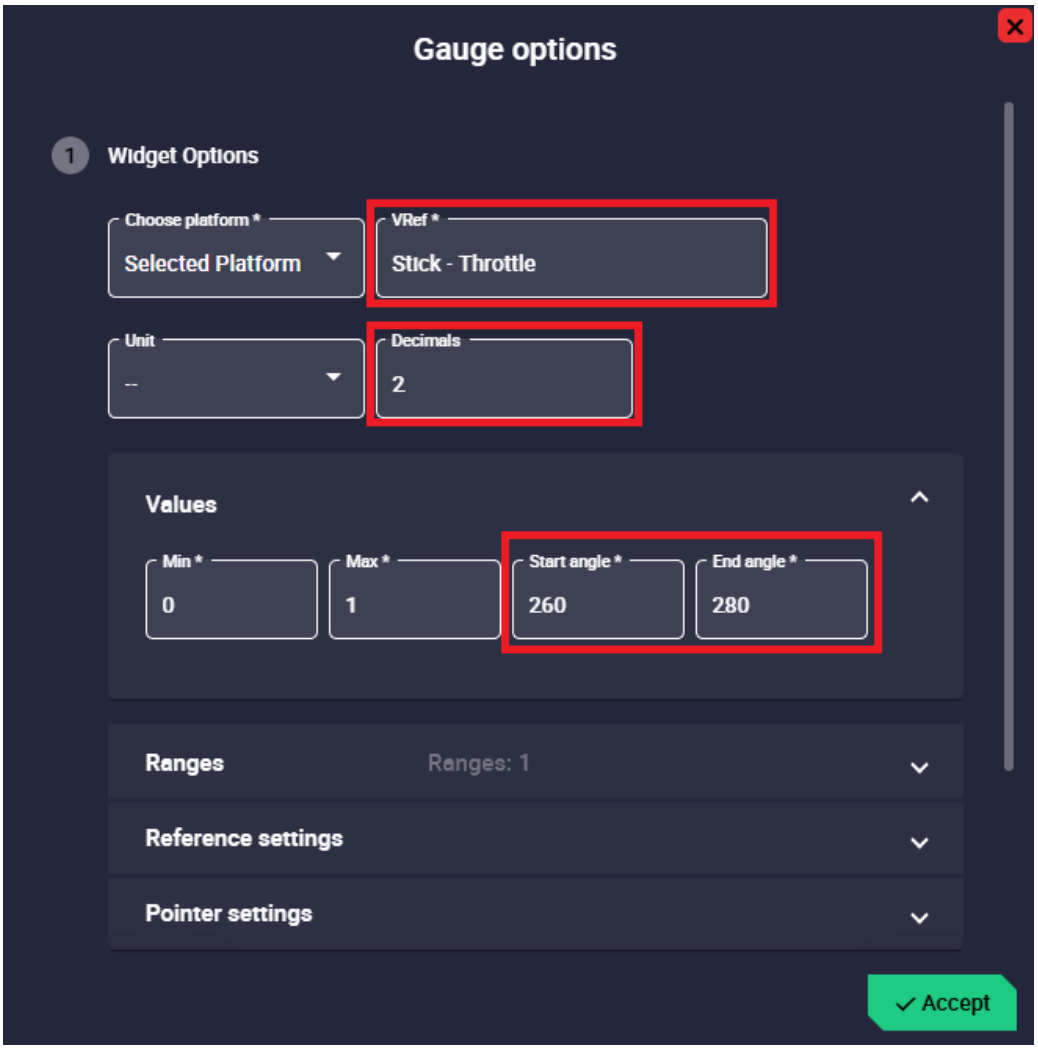
- **Choose pointer**: Users can choose a desired pointer from the list. If **none** is selected, no pointer will be drawn.
- **Pointer Offset Center**: Position of the "start" of the needle with respect to the center of the gauge. Default is set to 5.
- **Poiter width**: Width of the pointer. The default width is 6.

- **Pointer length**: Pointer length is set as a percentage of the distance between the center and the edge of the gauge.

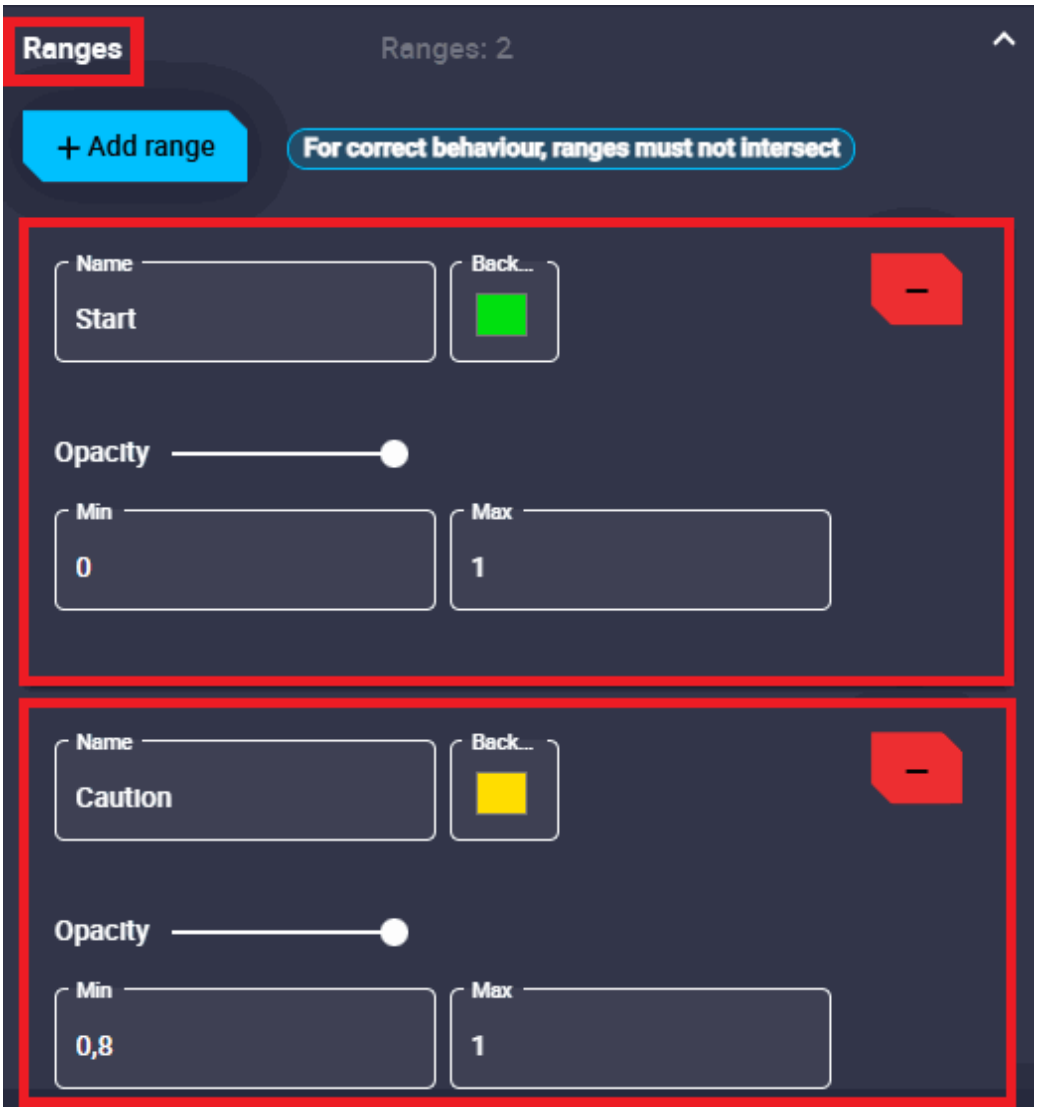
An example is given below:

Gauge example

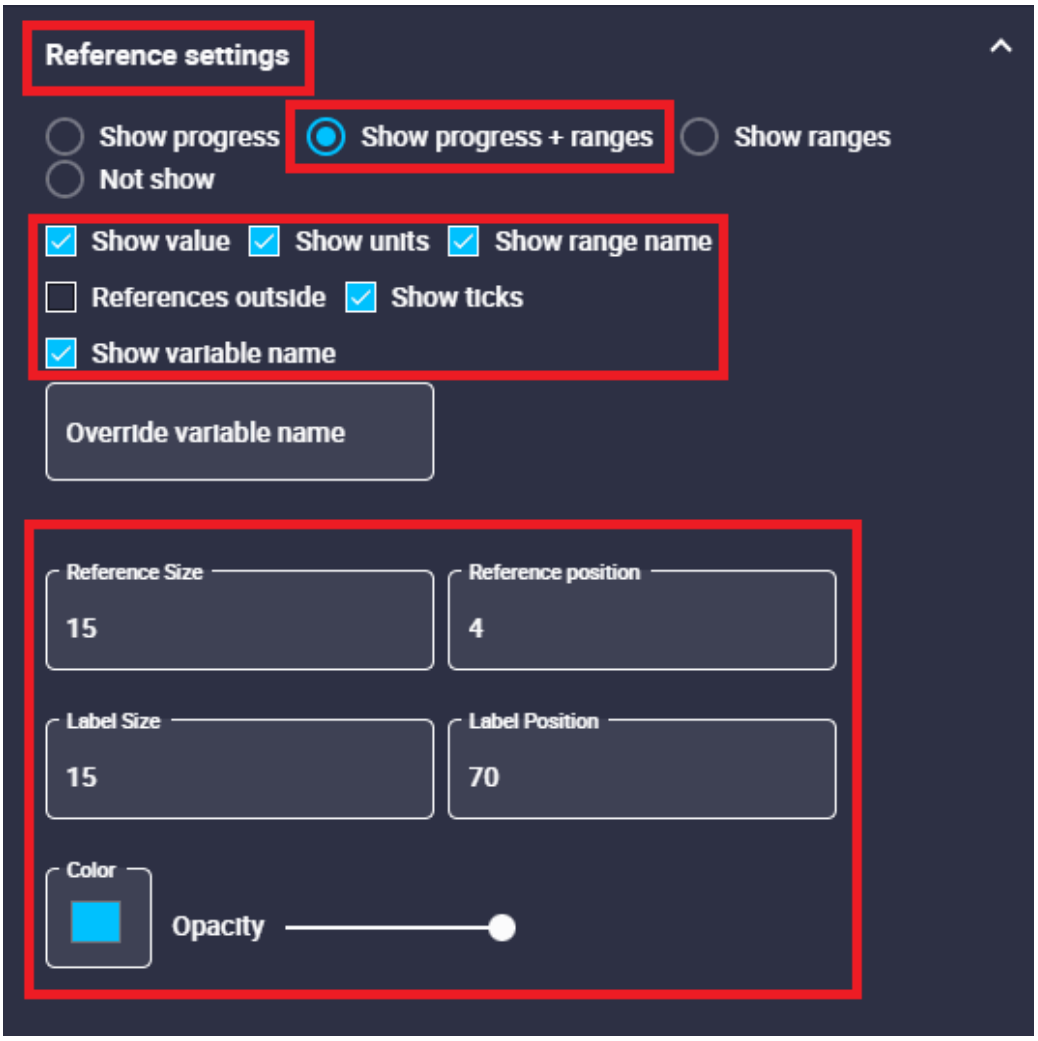
The custom configuration made for this gauge is as follows:



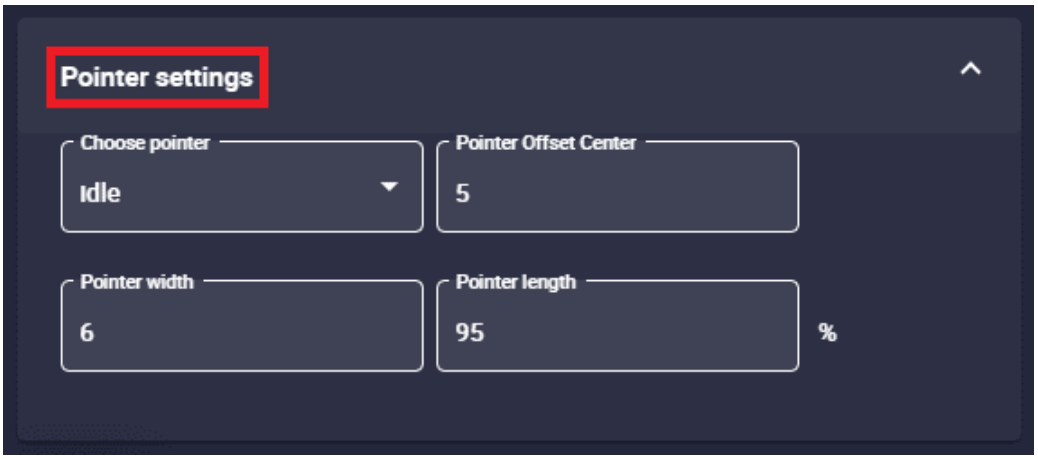
Gauge configuration - Values example



Gauge configuration - Ranges example



Gauge configuration - Reference settings example

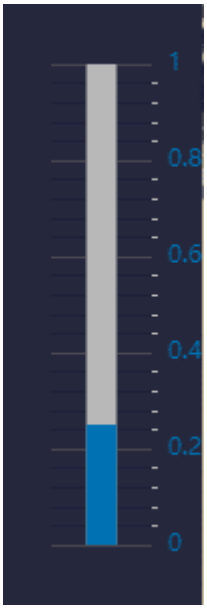


Gauge configuration - Pointer settings example

Bar

Bar widget graphically displays the value of a selected variable by means of a bar. In addition, color ranges can be specified according to specific values.

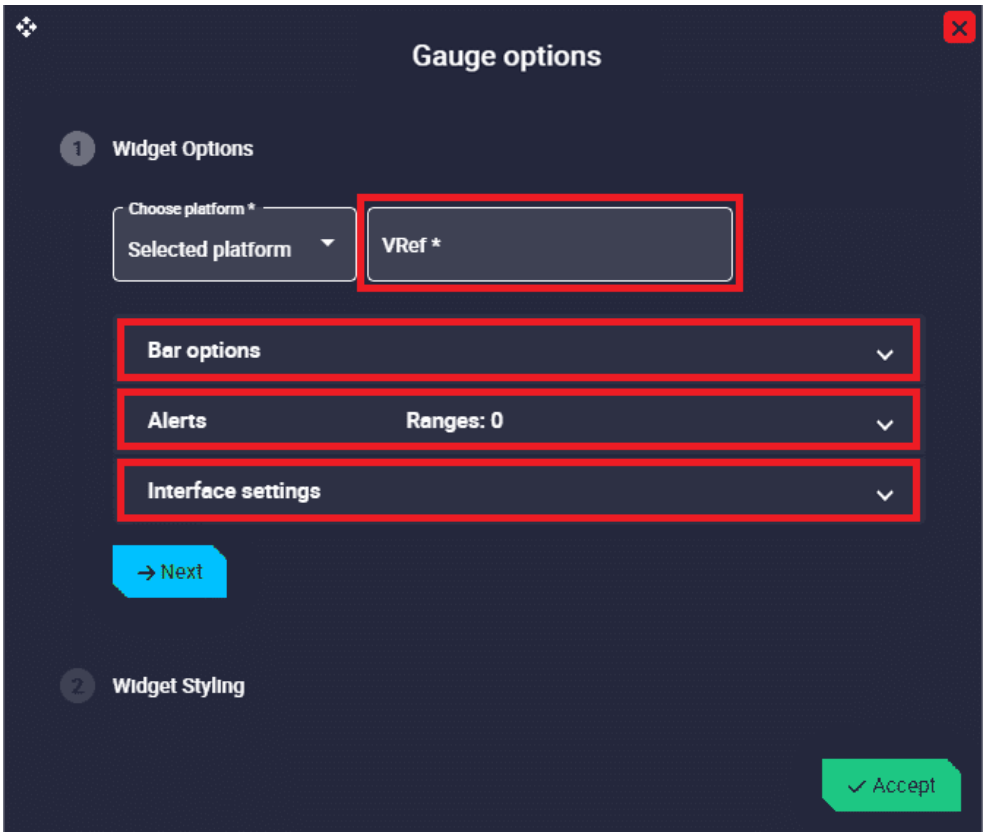
The appearance of this widget is highly customizable, allowing the user to adapt it to the desired use.



Bar

Right click on the **bar** to access its options (**Duplicate**, **Edit** and **Remove**).

- **Options → Edit**: This allows the user to access the Bar configuration menu.



Bar configuration

- **Widget Options**

  This widget has **extra edit parameters** compared to the ones described in the Widgets common configuration.

  - **VRef**: Users can choose any real, integer or bit variable to be displayed on the bar.
  - **Unit**: Unit of measurement of the displayed variable.

    > ⓘ **Note**
    > This parameter is only available with Real variables.

  - **Decimals**: Number of decimals displayed for the selected variable.

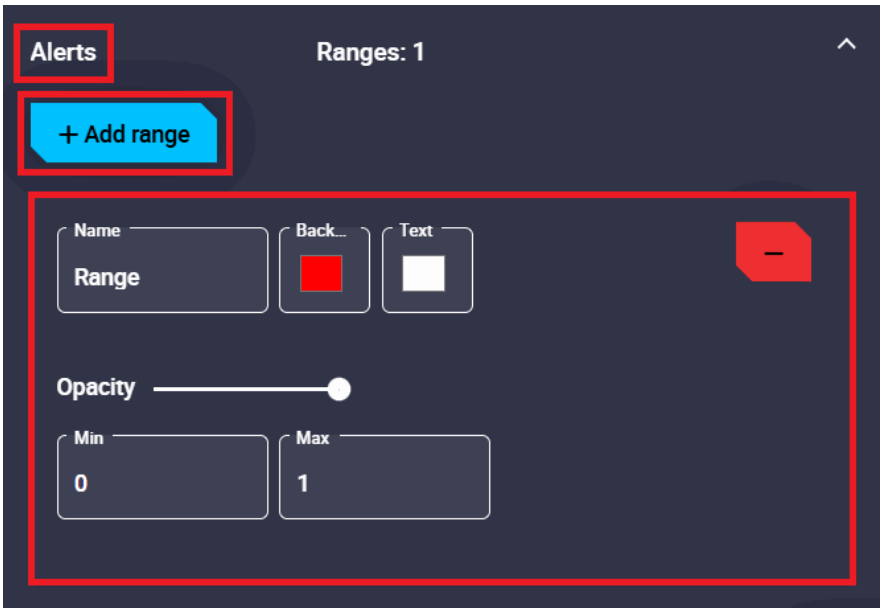
    > ⓘ **Note**
    > This parameter is only available with Real variables.

  - **Bar options**:

    Bar configuration - Bar options

    - **Bar color**: The color of the bar with the value can be changed here.
    - **Color text**: The color of the selected variable value can be modified.
    - **Bar Width**: The user can modify the width of the bar as desired. By default it is set to 30.
    - **Min/Max**: Minimum and maximum values displayed in the bar.
    - **Background color**: The color of the background bar can be set here.

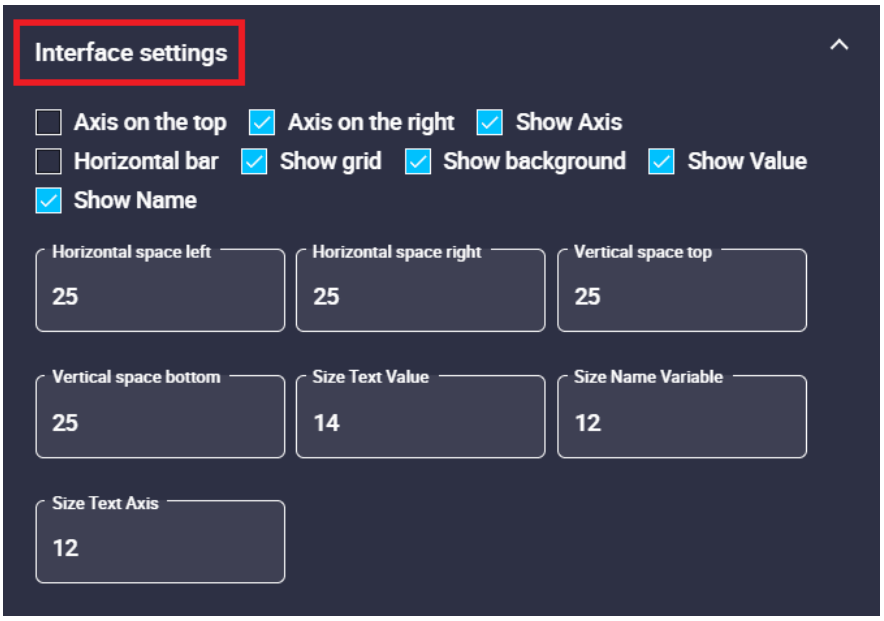
  - **Alerts**: Allows the user to set different ranges. For these ranges several customizations can be made:



Bar configuration - Alerts

- **Name**: An indentifier name of the range can be set.
- **Background color**: Sets the color of the bar for this range.
- **Text color**: Changes the color of the selected variable value in this range.
- **Opacity**: The opacity of the bar can be customized in this range.
- **Min/Max**: Defines the minimum and maximum values of the range.
- **Remove**: Click to delete this range.

- **Interface settings**:



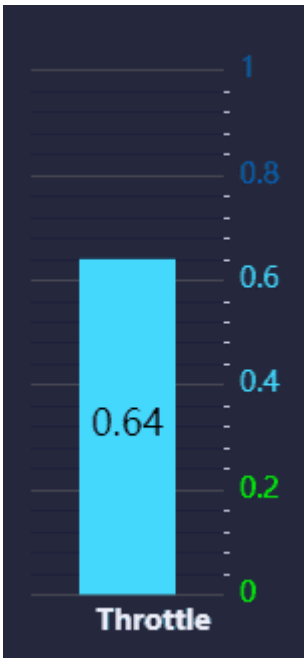
Bar configuration - Interface settings

The appearance of the bar widget can be highly customized using the following parameters:

- **Axis on the top**: If enabled, permits placing the axis above the bar instead of below it.
- **Axis on the right**: If enabled, the axis is placed to the right of the bar, otherwise to the left. By default it is enabled.
- **Show Axis**: If enabled, the axis of the bar is shown, otherwise it is hidden. By default it is enabled.
- **Horizontal bar**: If enabled, the bar and the axis are displayed in horizontal format, otherwise it is in vertical position.
  By default disabled (in vertical position).
- **Show grid**: If enabled, a grid with the subdivisions of the axes is shown, otherwise it is hidden. By default it is enabled.
- **Show background**: If enabled, the rest of the bar that is not colored by the value is shown with the color previously defined in **Bar options**.
  By default it is enabled.
- **Show Value**: If enabled, shows the value of the selected variable in the bar. By default it is enabled.
- **Show Name**: If enabled, the name of the selected variable is displayed at the bottom of the bar. By default it is enabled.
- **Horizontal space left/right**: The left/right horizontal spacing of the bar widget background can be customized. By default they are set to 25.
- **Vertical space top/bottom**: The top/bottom vertical spacing of the bar widget background can be customized. By default they are set to 25.
- **Size Text Value**: The size of the text value can be modified. Default is 14.
- **Size Name Variable**: The size of the selected variable name can be changed. Default is 12.
- **Size Text Axis**: The size of the axis text can be modified. Default is 12.

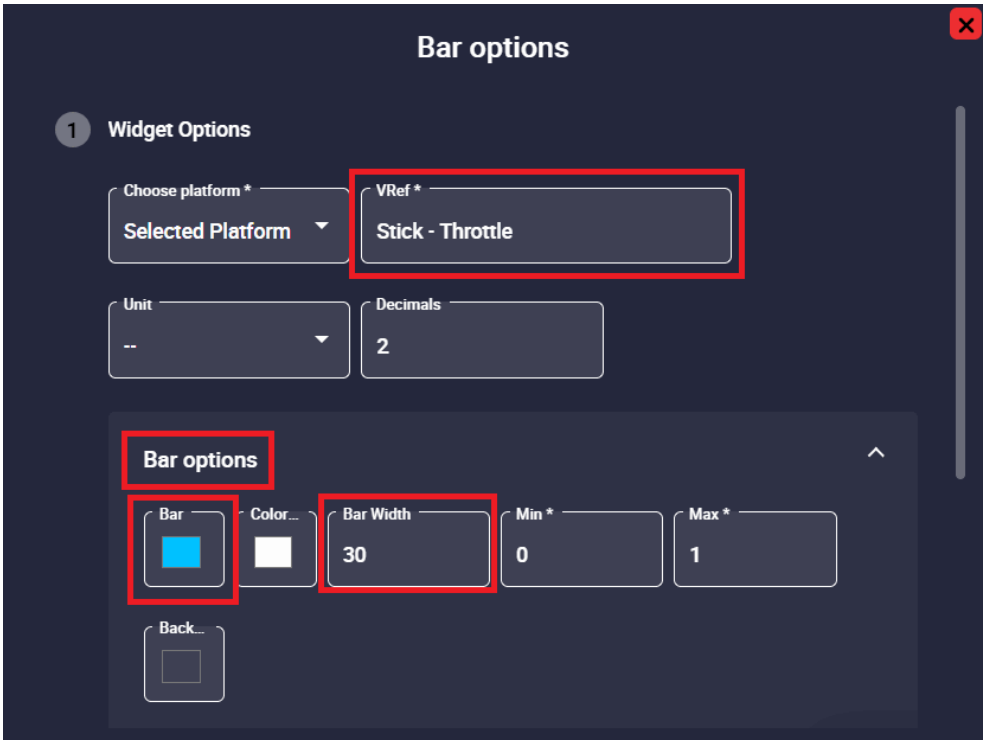
Some examples are given below:

- **Example 1**:



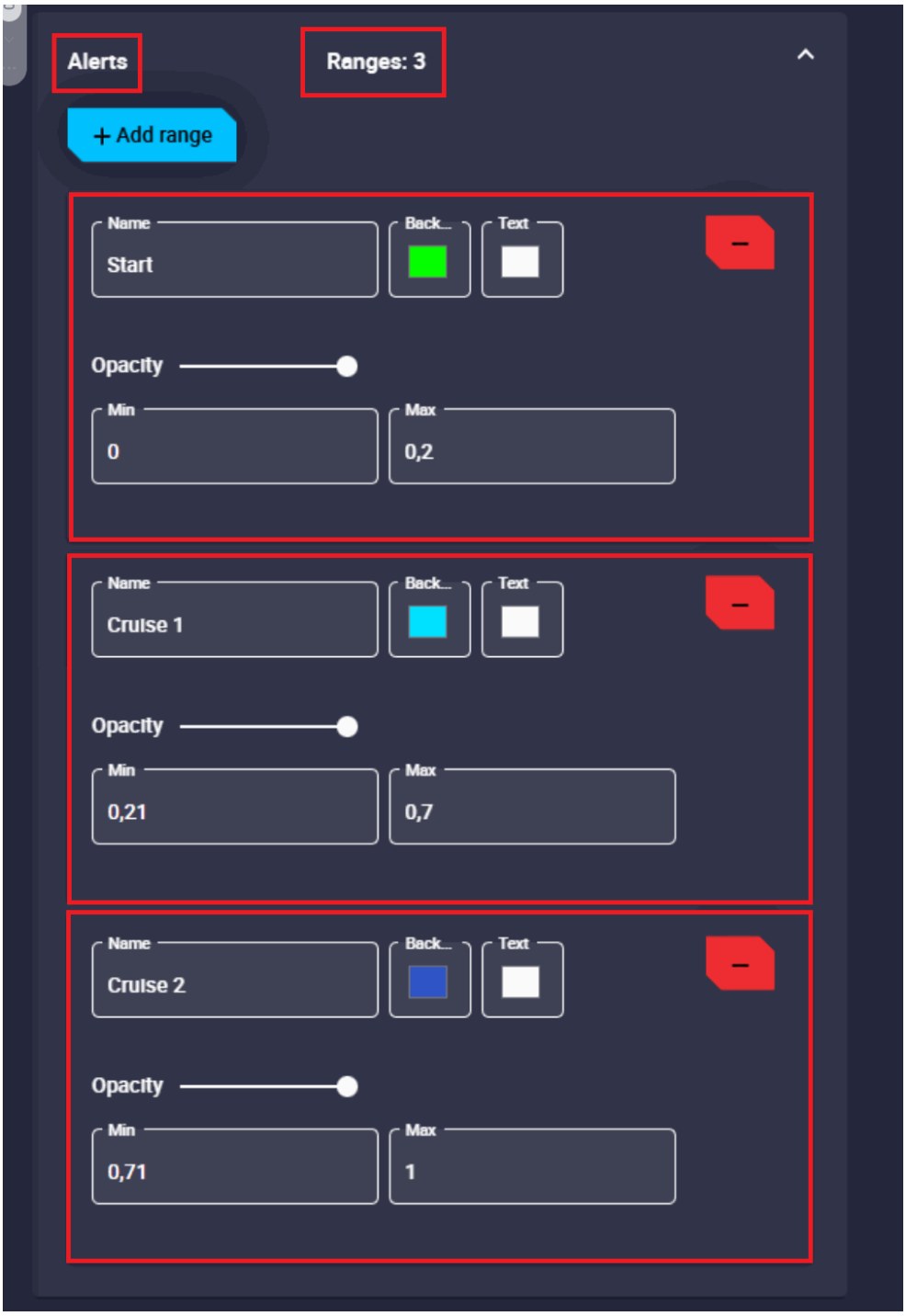
Bar example 1

The custom configuration made for this bar is as follows:

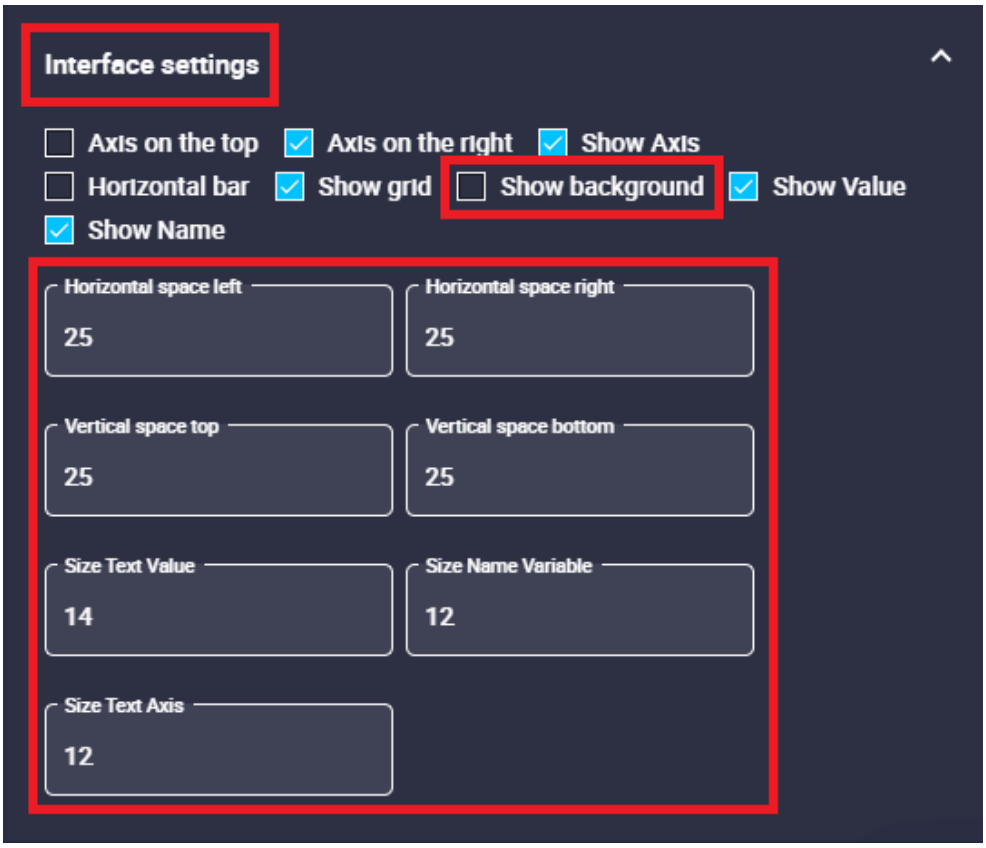


Bar configuration - Bar options example 1

As shown in the figure above, when the background bar is hidden ('Show background' option is disabled), the '**Background color**' parameter does not appear.

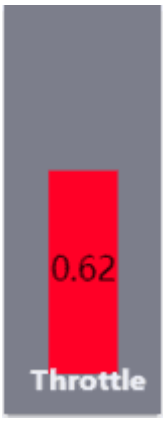


Bar configuration - Alerts example 1



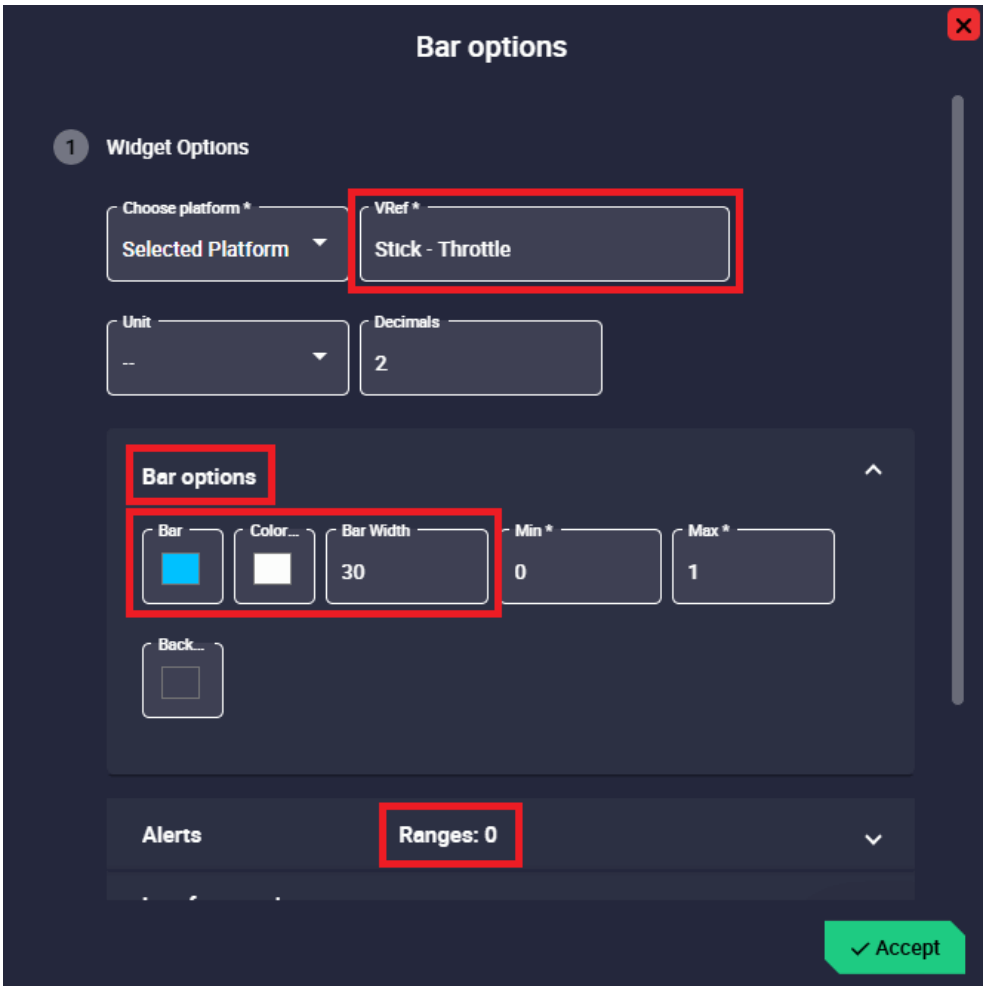
Bar configuration - Reference settings example 1

- **Example 2**:



Bar example 2

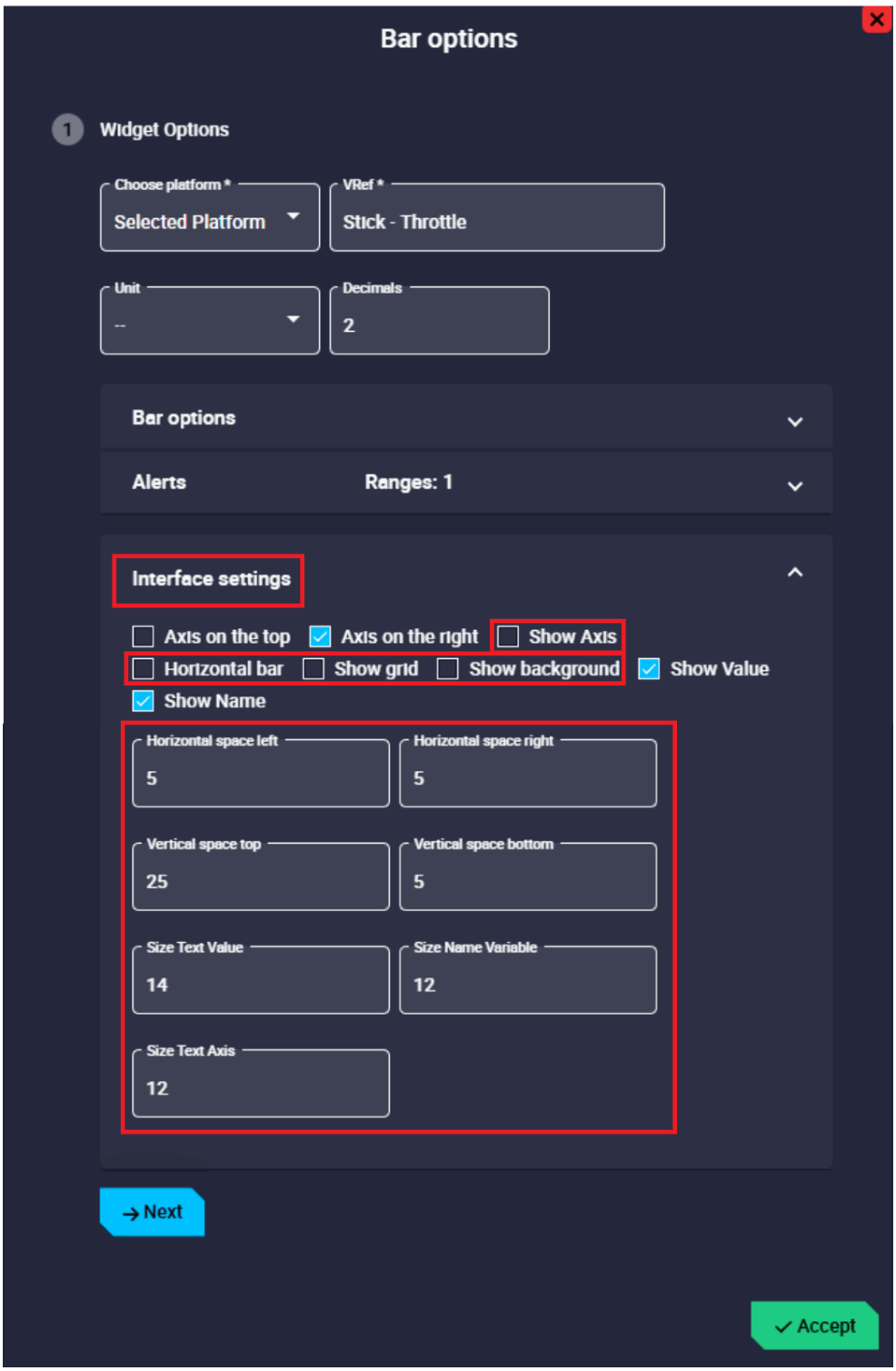
The custom configuration made for this bar is as follows:



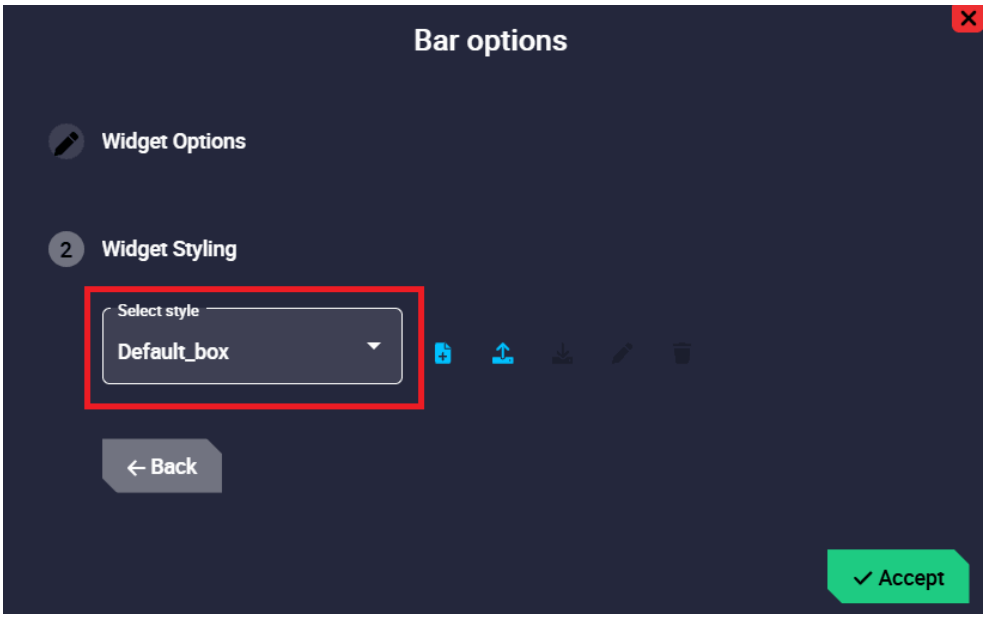
Bar configuration - Bar options example 2

As shown in the figure above, when the background bar is hidden ('Show background' option is disabled), the '**Background color**' parameter does not appear.

In addition, as there is no range specified, the color of the bar will always be the one specified in the '**Bar color**' parameter.



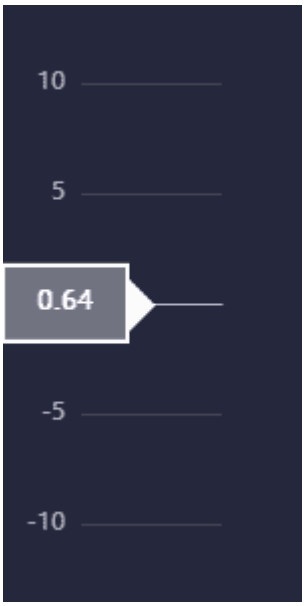
Bar configuration - Reference settings example 2



Bar configuration - Widget Styling example 2

Roll Tape

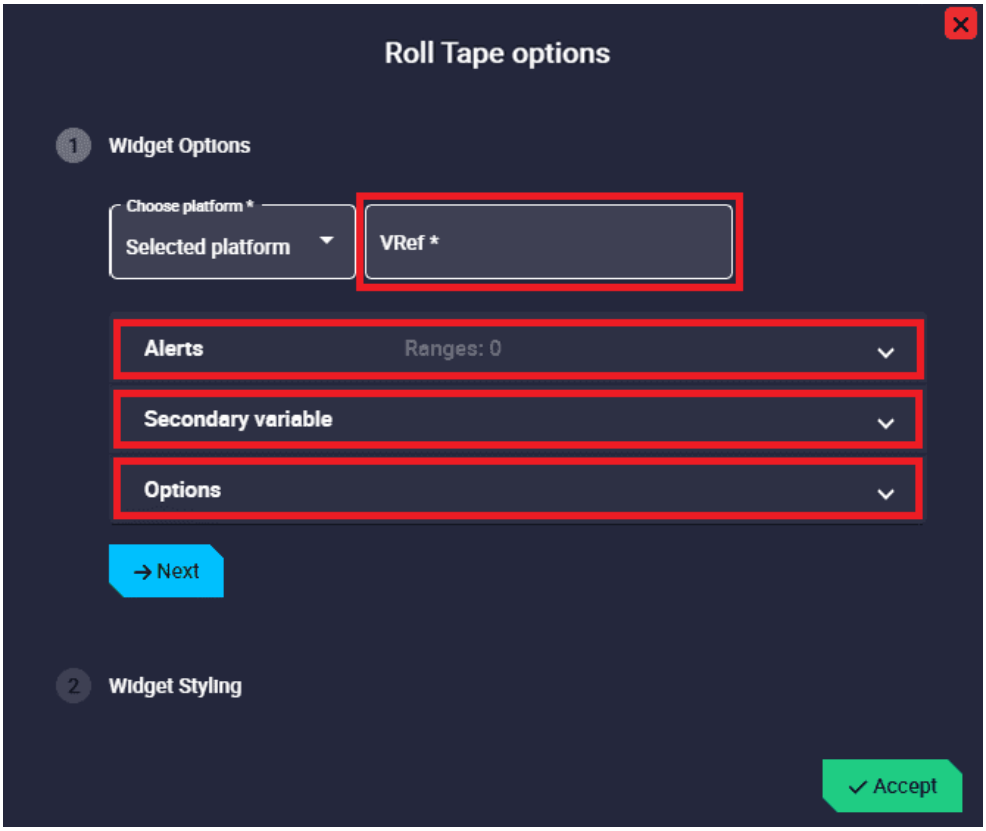
Roll tape widget shows the value of the selected variable with a marker always in the center position of the widget. It is also possible to add color ranges according to the values of the variable. In addition, a second variable can be used as a reference marker.



Roll Tape

Right click on the **roll tape** to access its options (**Duplicate**, **Edit** and **Remove**).

- **Options → Edit**: This allows the user to access the Roll Tape configuration menu.



Roll Tape configuration

  - **Widget Options**

    This widget has **extra edit parameters** compared to the ones described in the Widgets common configuration.

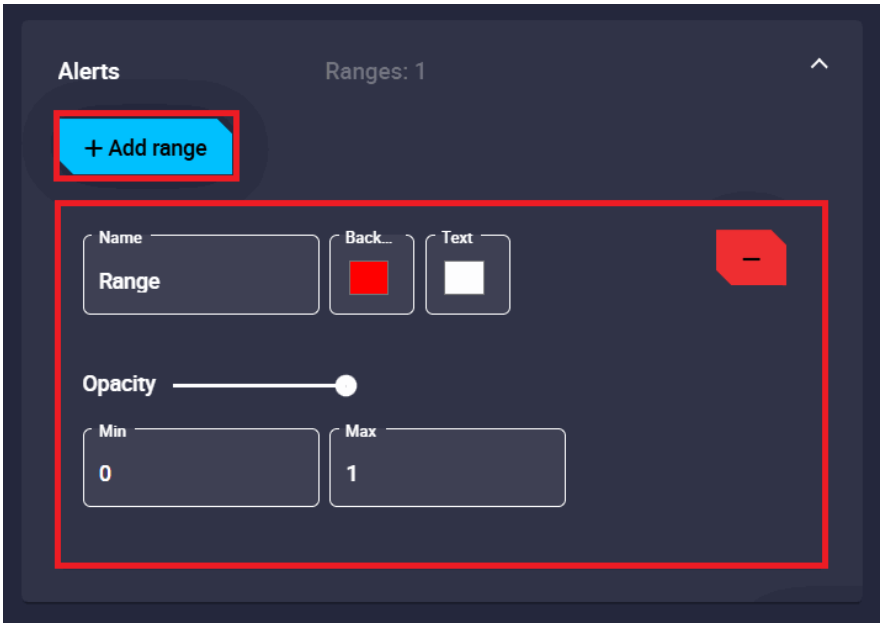
    - **VRef**: Users can choose any real, integer or bit variable to be displayed on the roll tape.
    - **Unit**: Unit of measurement of the displayed variable.

      > ⓘ **Note**
      > This parameter is only available with Real variables.

    - **Decimals**: Number of decimals displayed for the selected variable.

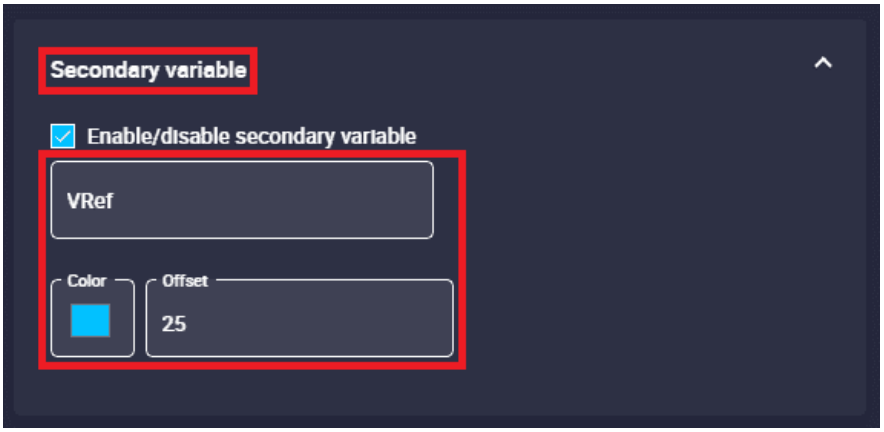
      > ⓘ **Note**
      > This parameter is only available with Real variables.

- **Alerts**: Allows the user to set different ranges. For these ranges several customizations can be made:



Roll Tape configuration - Alerts

  - **Name**: An identifying name can be set for the range.
  - **Background color**: Sets the color of the roll tape for this range.
  - **Text color**: Sets the color of the range name for this range.
  - **Opacity**: The opacity of the roll tape can be customized in this range.
  - **Min/Max**: Defines the minimum and maximum values of the range.
  - **Remove**: Click to delete this range.
- **Secondary variable**: A second variable can be added to be used, for example, as a reference.



Roll Tape configuration - Secondary variable

- **Enable/disable secondary variable**: If enabled, it is displayed as a marker.
- **VRef**: Users can choose any real, integer or bit variable to be displayed on the roll tape as a **secondary variable**.
- **Unit**: Unit of measurement of the displayed variable.

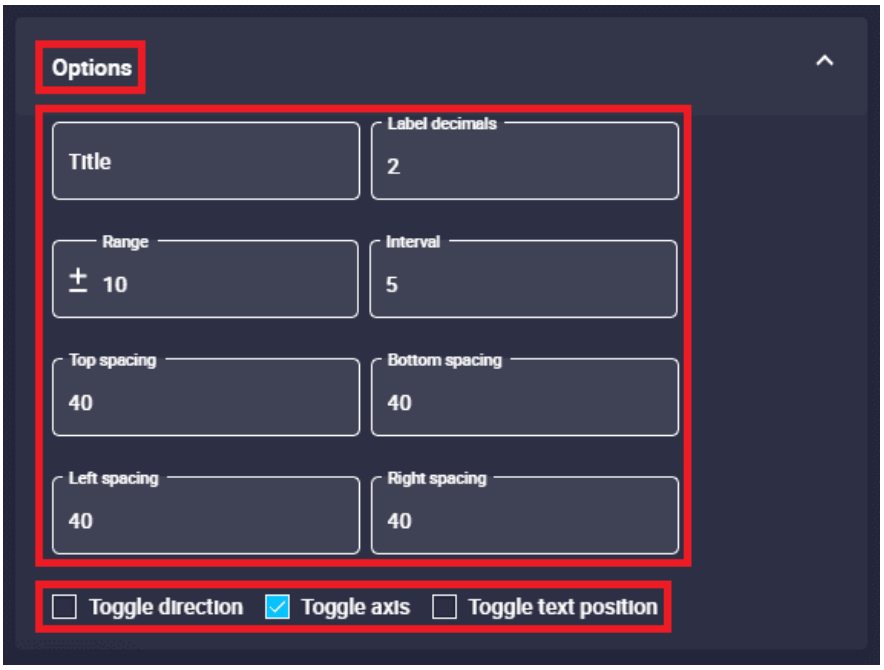
> ⓘ **Note**
> This parameter is only available with Real variables.

- **Decimals**: Number of decimals displayed for the selected variable.

> ⓘ **Note**
> This parameter is only available with Real variables.

- **Color**: Sets the color of the secondary variable marker.
- **Offset**: Horizontal position of the marker. By default it is set to 25.

- **Options**:



Roll Tape configuration - Options

The appearance of the roll tape widget can be customized using the following parameters:

- **Title**: Title of the roll tape widget

- **Label decimals**: Decimals of the value to be displayed on the roll tape. By default it is set to 2.
- **Range (+/-)**: Space between the center and the top and bottom boundaries. The default value is 10.
- **Interval**: Axis interval. Default is 5.
- **Top/Bottom spacing**: The top/bottom spacing of the roll tape widget background can be customized. The default value is 40.
- **Left/Right spacing**: The left/right horizontal spacing of the roll tape widget background can be customized. The default value is 40.
- **Toggle direction**: If enabled, the roll tape and the axis are displayed in horizontal format, otherwise it is in vertical position.

  By default disabled (in vertical position).
- **Toggle axis**: If enabled, the axis is positioned to the left of the roll tape, otherwise to the right. By default it is enabled.
- **Toggle text position**: If enabled, the previously defined title is positioned at the top of the widget, otherwise it is positioned at the bottom.

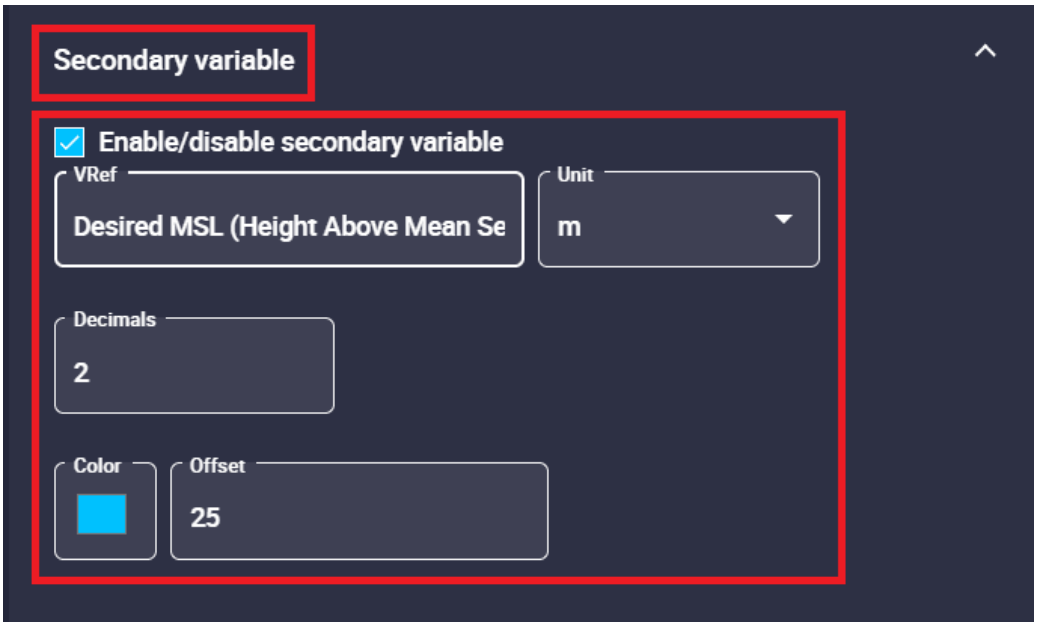
  By default disabled.

An example is given below:

Roll Tape example

The custom configuration made for this roll tape is as follows:

Roll Tape configuration - Alerts example



Roll Tape configuration - Secondary variable example

Roll Tape configuration - Options example

Label Script

The Label Script options (or Script options) panel allows creating highly customized widgets within the Veronte Ops interface using JavaScript code.

Right click on the **Label Script** to access its options (**Duplicate**, **Edit** and **Remove**).

- **Options → Edit**: This allows the user to access the Roll Tape configuration menu.

Label Script options

- **Widget Options**

  This widget has **extra edit parameters** compared to the ones described in the Widgets common configuration.

  - **Code**: The area where the logic is written. The script must return an object or an array of objects containing parameters such as label, color, backgroundColor, and value.
  - **Import script**: Allows uploading pre-configured external script files.

Commands

Veronte Ops allows the user to modify some variables during a flight to test some parameters or to simulate a stick control.

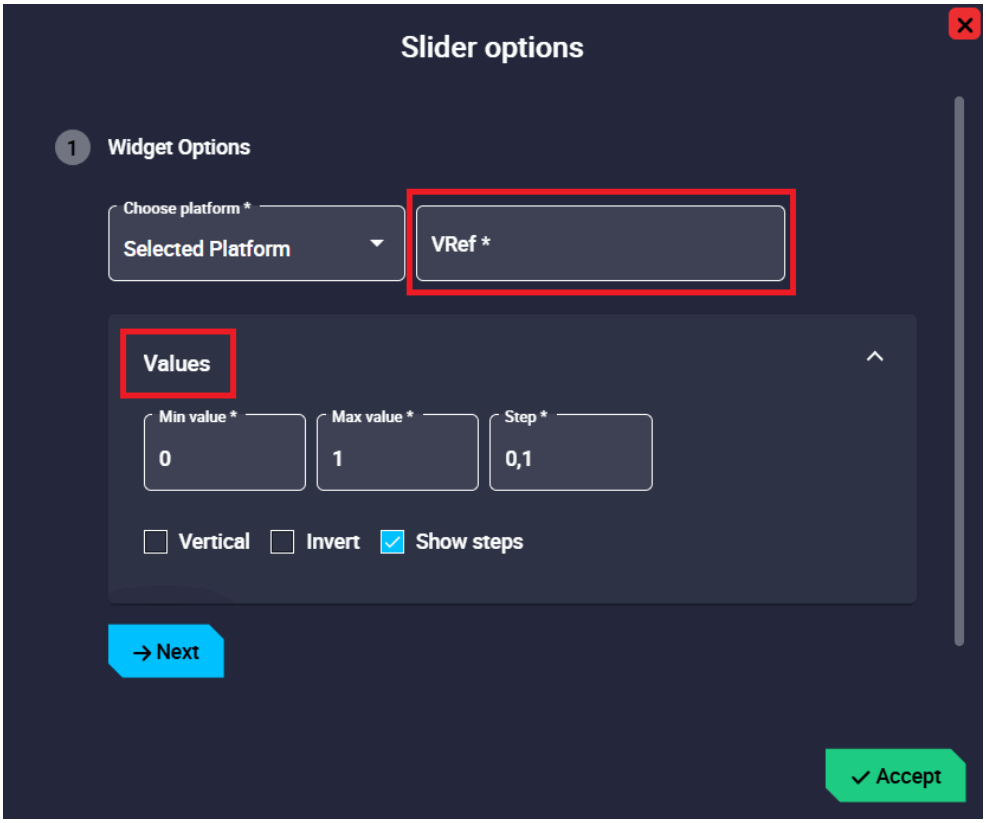
Slider

Slider widgets allow the user to choose a certain variable and change its value by simply moving the slider to the desired value from the workspace during a flight.

Slider

Click the button to access its options (**Duplicate**, **Edit** and **Remove**).

- **Options → Edit**: This allows the user to access the Slider configuration menu.



Slider configuration

  - **Widget Options**

This widget has **extra edit parameters** compared to the ones described in the Widgets common configuration.

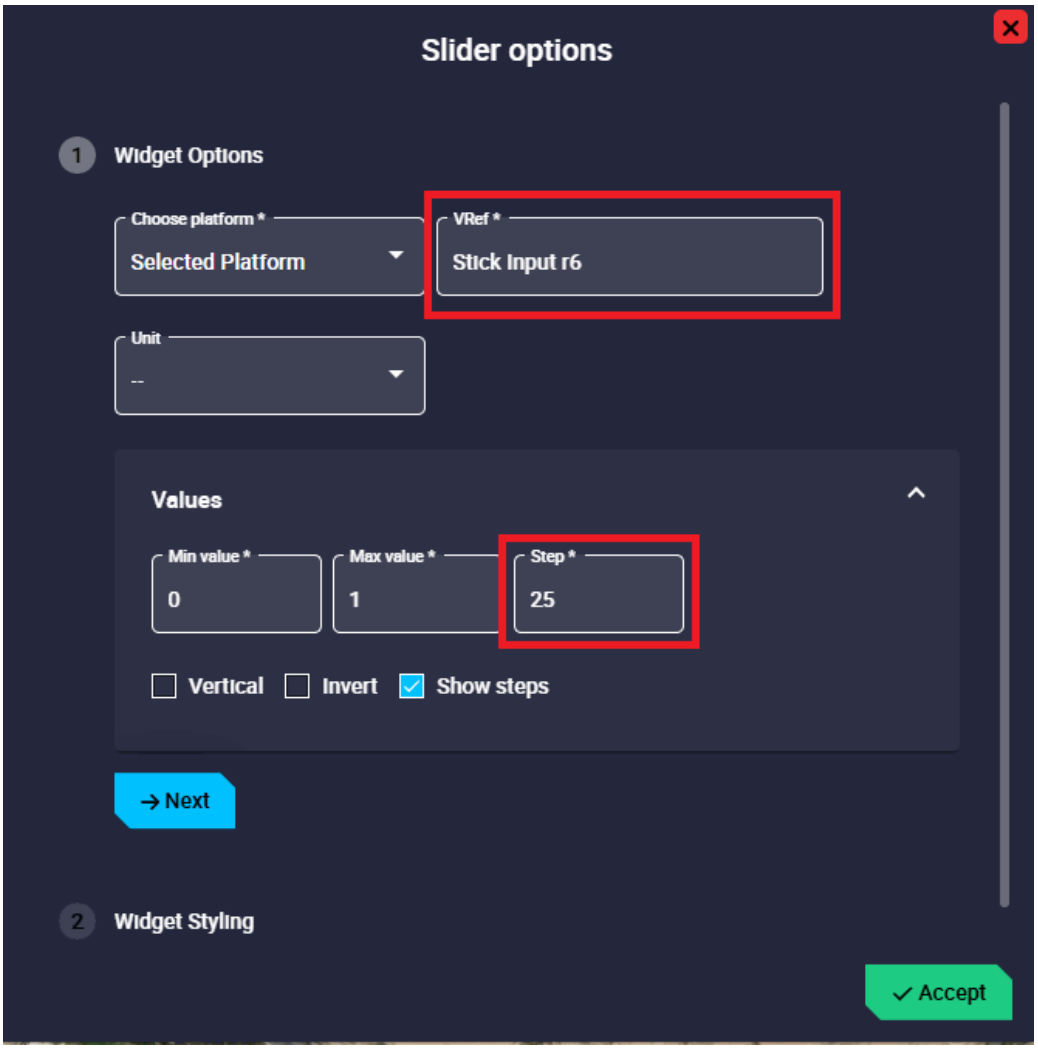
- **VRef**: Users can choose any real, integer or bit variable to be displayed on the slider.
- **Unit**: Unit of measurement of the displayed variable.

  > ⓘ **Note**
  > This parameter is only available with Real variables.

- **Values**:
  - **Min/Max value**: Minimum and maximum values displayed in the slider. Defaults are 0 and 1 respectively.
  - **Step**: Here users must enter the step they want the slider to have. By default it is set to 0.1.
  - **Vertical**: If enabled, the slider widget is displayed in vertically, otherwise it is displayed horizontally. By default disabled (in horizontal position).
  - **Invert**: If enabled, the minimum and maximum values are swapped, otherwise the minimum value is to the left of the widget and the maximum value to the right. By default disabled.
  - **Show steps**: If enabled, the step divisions of the slider are shown when interacting with the widget, otherwise they are hidden.
    By default it is enabled.

An example is given below:

Slider example



Slider configuration example

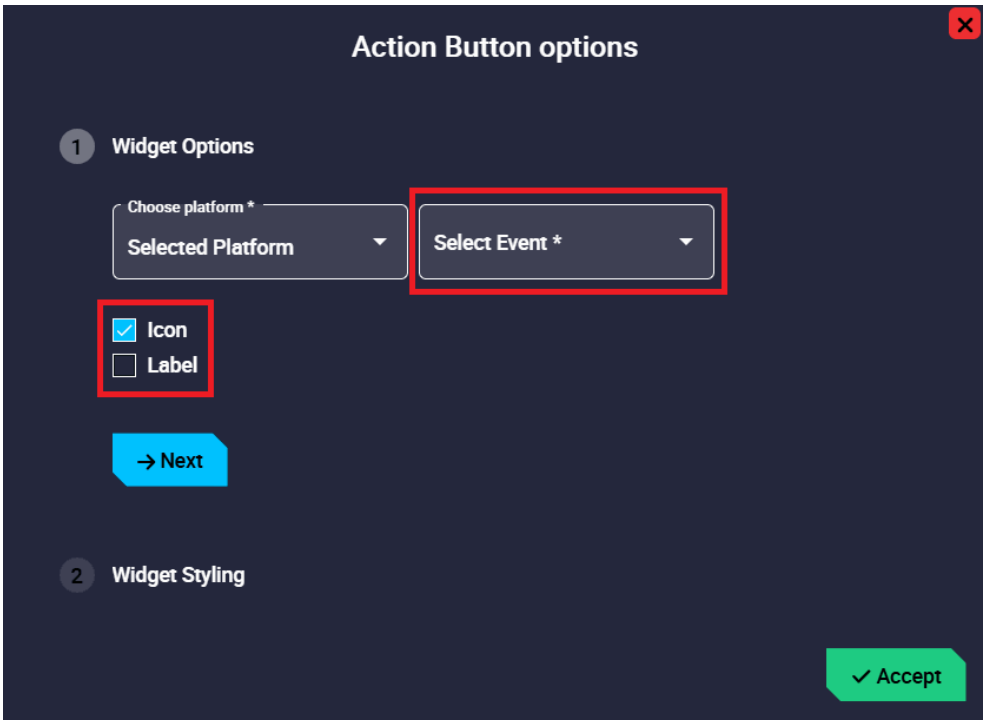
Action Button

Action Button widgets can be added to the workspace **independently** and also be displayed all together embedded in the Veronte Panel.

Action Button

Click the button to access its options (**Duplicate**, **Edit** and **Remove**).

- **Options → Edit**: This allows the user to access the Action Button configuration menu.



Action Button configuration

- **Widget Options**

  This widget has **extra edit parameters** compared to the ones described in the Widgets common configuration.

  - **Select Event**: Select the desired button event to be displayed as an action button.

    This event must be previously configured as a button automation in the **1x PDI Builder** software.
  - **Icon**: If enabled, the icon already configured for this event in the corresponding automation button is drawn. By default it is enabled.
  - **Label**: If enabled, users can add a desired text to be displayed on the action button next to the icon. By default, it is disabled.
    - **Override label**: Enter the desired text to be displayed in the widget. If no text is entered, the name of the selected event will be shown.
    - **Label size**: Size of the text displayed on the action button. Default is 16.

The triggering of linked actions is subjected to the button configuration carried out in **1x PDI Builder**:

- No time control: The action is triggered when the button is pressed.

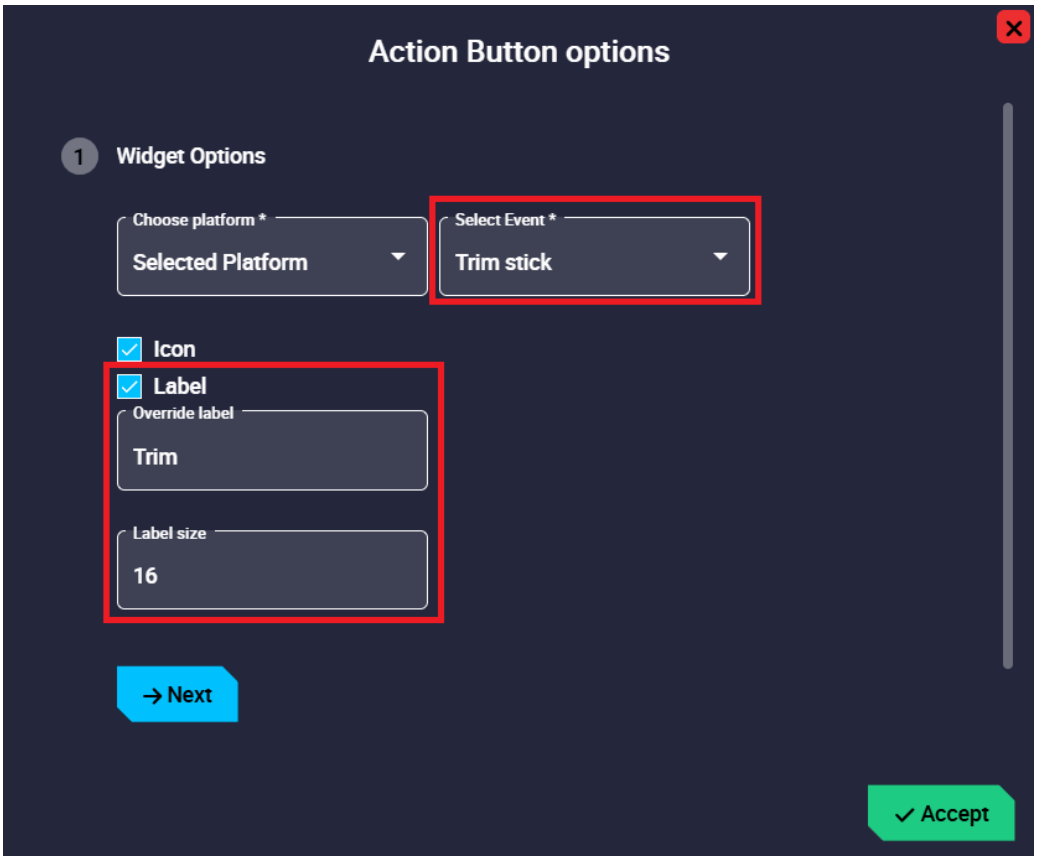
Action button - No time control

- Time control: The action is triggered when the button is pressed during the configured time.

Action button - Time control of 3s

An example of the **configuration of an Action button** widget is given below:

Action Button example



Action Button configuration example

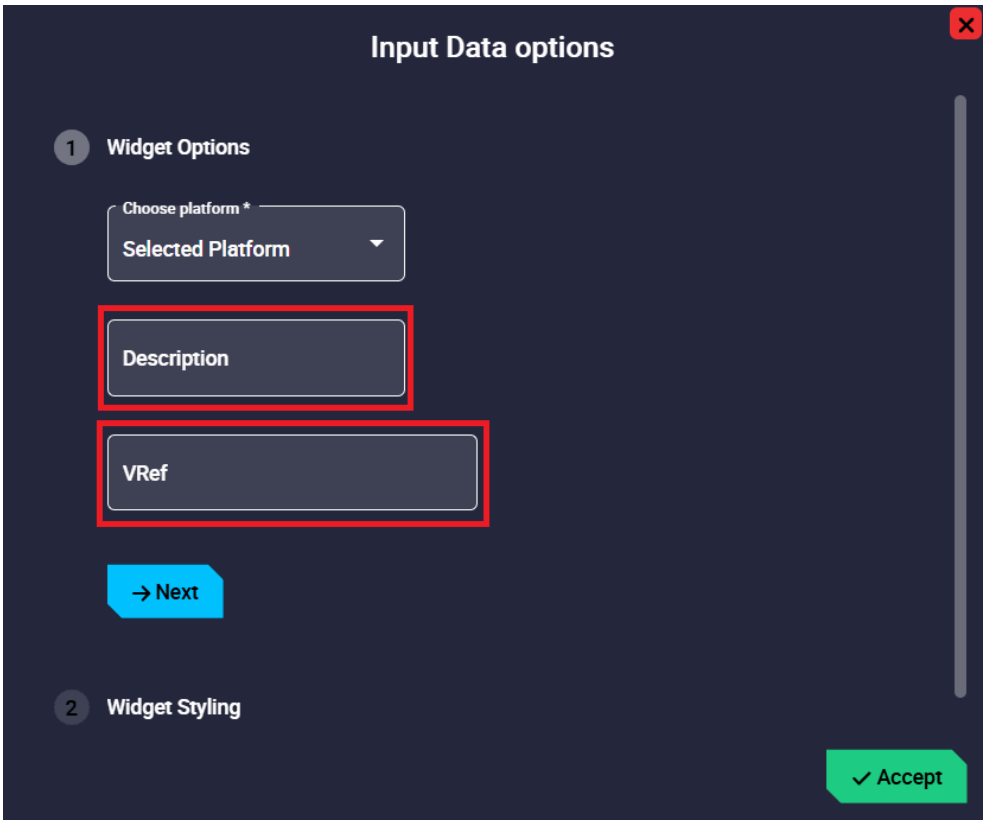
Input Data

Input Data widgets allow the user to choose a certain variable and change its value by manually entering the value and sending it to the platform from the workspace during a flight.

Input Data

Click the button to access its options (**Duplicate**, **Edit** and **Remove**).

- **Options → Edit**: This allows the user to access the Input Data configuration menu.



Input Data configuration

- **Widget Options**

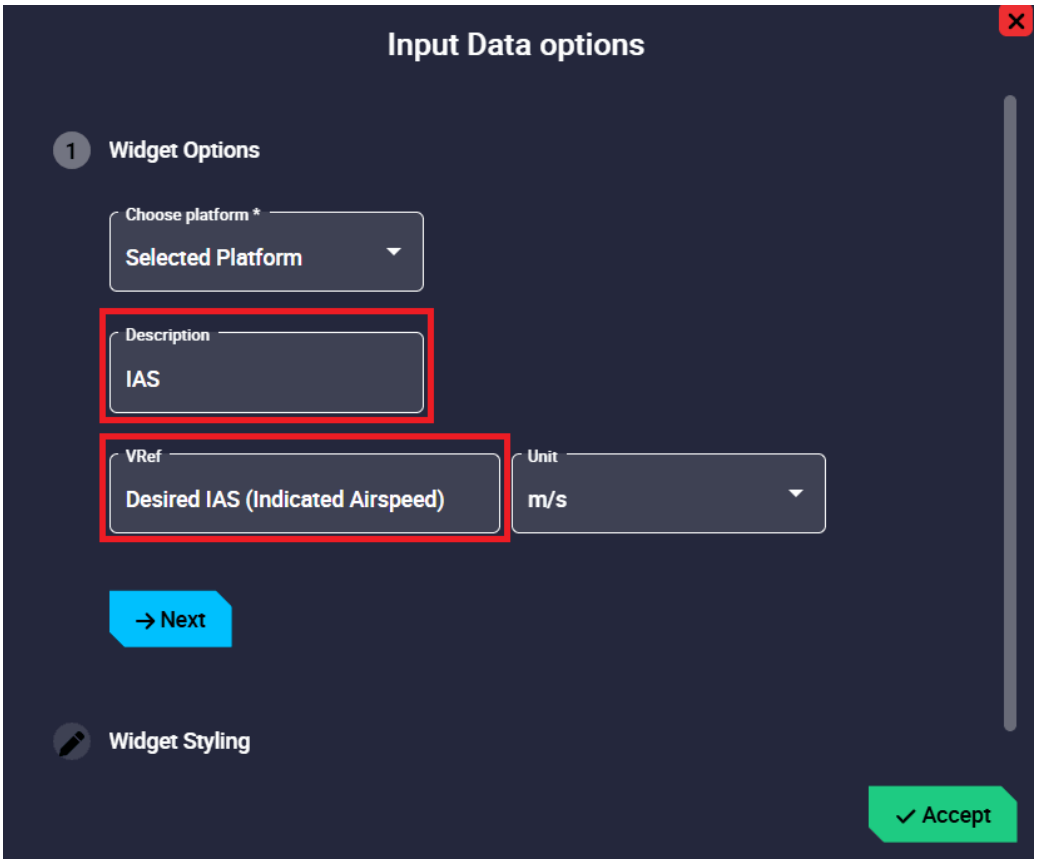
  This widget has **extra edit parameters** compared to the ones described in the Widgets common configuration.

  - **Description**: A description can be added to the widget by substituting the selected variable name.
  - **VRef**: Users can choose any real, integer or bit variable to be displayed on the widget.
  - **Unit**: Unit of measurement of the displayed variable.

    > ⓘ **Note**
    > This parameter is only available with Real variables.

An example is given below:

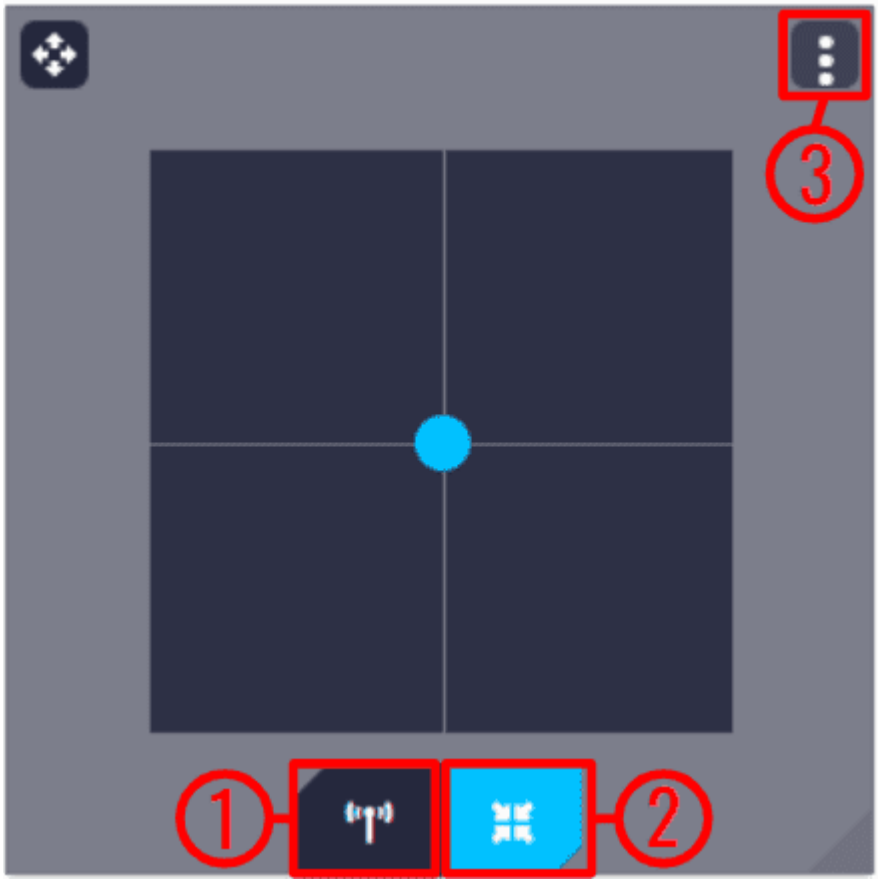
Input Data example



Input Data configuration example

Stick

Stick (Virtual stick) widgets are created to simulate a radio controller that controls the platform channels directly from the computer.



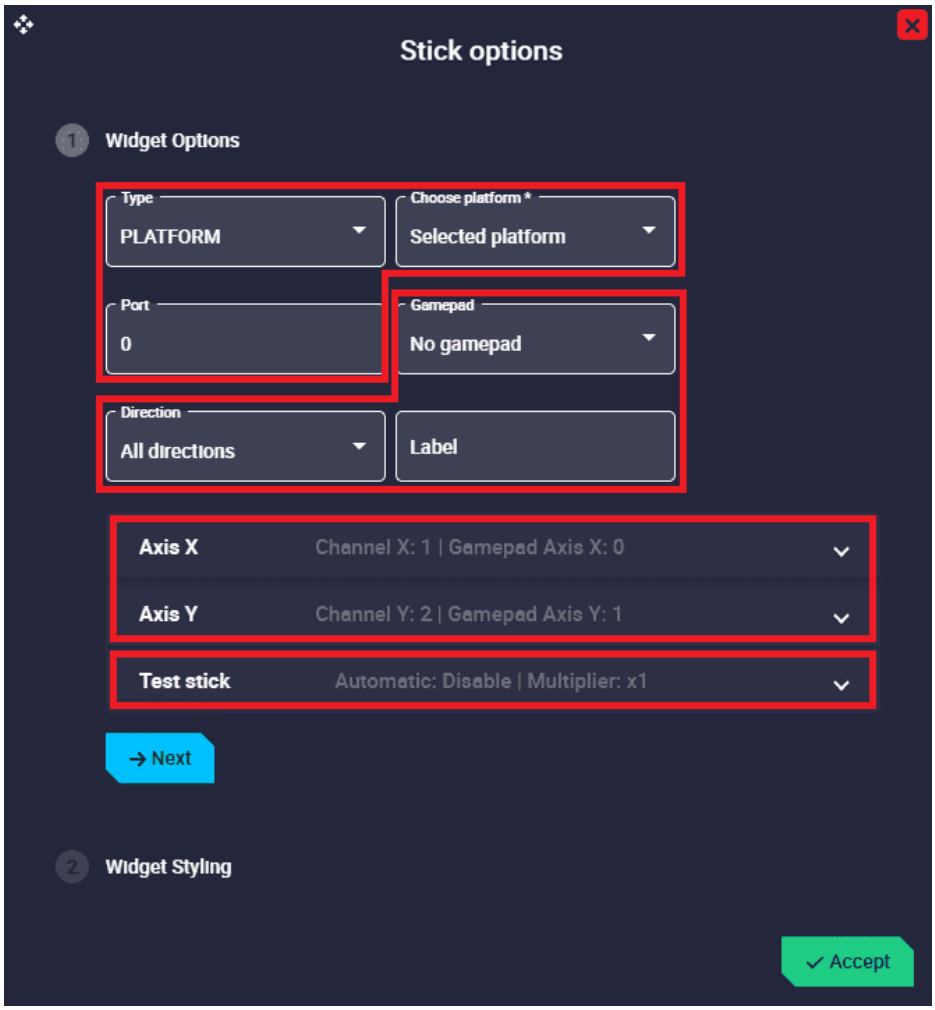
Stick

On the stick display, there are 3 icons/buttons and their functions are as follows:

1. **Send command enable/disable**: Enables/Disables the virtual stick commands.

   > ⚠ **Warning**
   > If the Stick is not actived, it will have no effect on the system.

2. **Go to center**: Returns the stick to the center.
3. **Options**
   - **Duplicate**: Duplicates this widget.
   - **Edit**: This allows the user to access the Stick configuration menu.

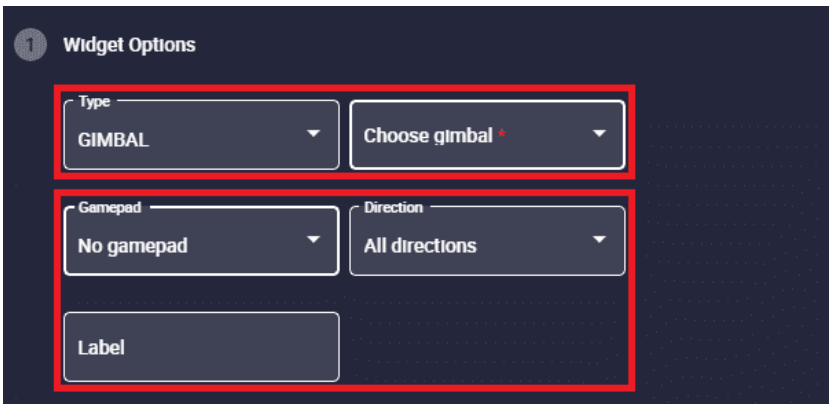


Stick configuration

- **Widget Options**

  This widget has **extra edit parameters** compared to the ones described in the Widgets common configuration.

  - **Type**: Users can configure the widget stick to be used in "normal" mode, i.e. to control the **platform**, or to control a **gimbal** instead.

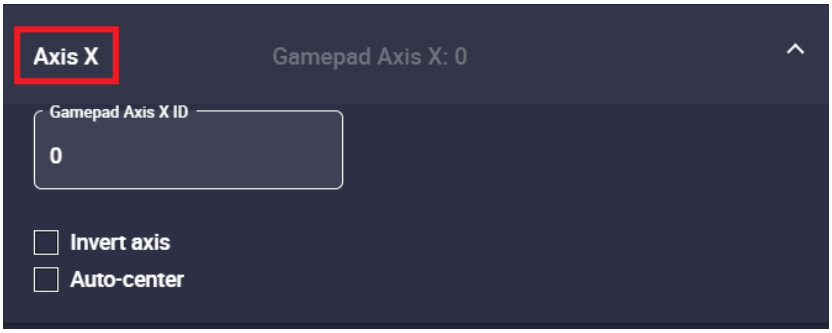


Stick configuration - Gimbal type

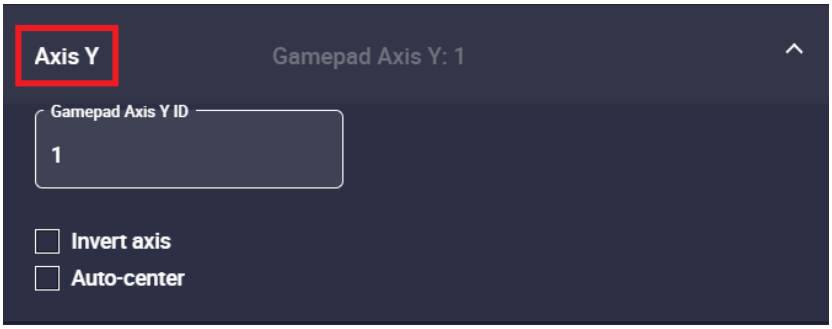
- **Choose platform/gimbal**: Depending on the type selected, the user will have to choose the **platform** or the **gimbal** for which the widget is configured.
- **Gamepad**: Users must select a desired gamepad from the list for which the widget is configured.

> **Important**
> If no physical gamepad (via USB) is connected to the PC, No gamepad option will be selected.

- **Direction**: The user has to configure the directions in which the stick can be moved, the available options are All directions, Only vertical and Only horizontal. By default, 'All directions' option is selected.
- **Label**: If there is more than one stick widget, users can easily differentiate between them by configuring an identifying label for each one.
- **Axis X/Y**: X and Y axes must be configured.



Stick configuration - Axis X



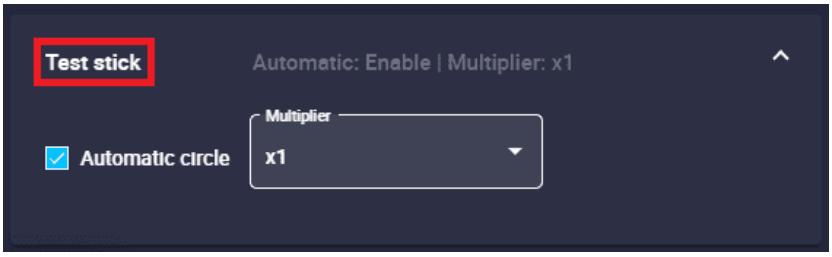
Stick configuration - Axis Y

  - **Gamepad Axis X/Y ID**: Correlation of the physical gamepad axis IDs with those of the virtual stick.

> **Important**
> If no physical gamepad (via USB) is connected, this parameter has no effect on the widget. It can be left at the default value.

- **Invert axis**: If enabled, the minimum and maximum of the axis in the variable associated with this channel will be inverted.
  For example, if in the Y axis, the **bottom of the axis** corresponds to a value of **0** in the stick input r2 variable. If the **Inverted axis** is **enabled**, now the **bottom of the axis** will correspond to a value of **1** in the variable.
- **Auto-center**: When activated, the stick automatically returns to the center position when released.

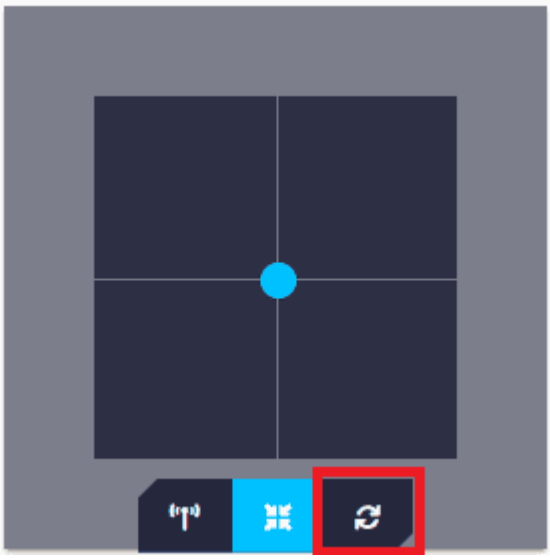
- **Test stick**: This option is used to generate stick inputs that are introduced in the system. This is a way to check how the system behaves when a stick command enters the autopilot.



Stick configuration - Test stick

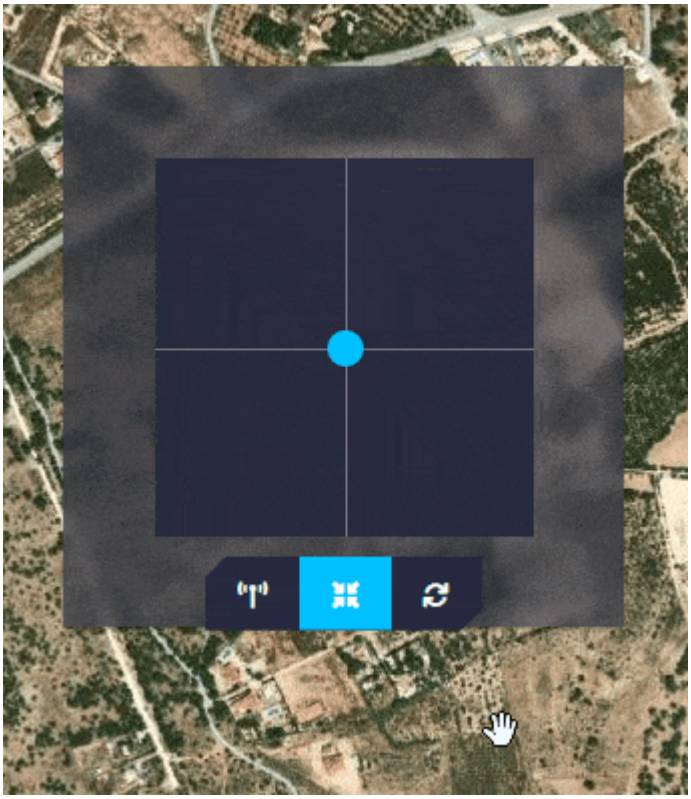
- **Automatic circle**: An automatic circle test is activated.
- **Multiplier**: The speed of the test can be adjusted. Available options are: x1, x2, x4, x8 and x16.

An additional button appears in the widget to start/stop the test.



Stick configuration - Test stick button

An example is presented below:



Stick configuration - Test stick example

- **Remove**: Deletes this widget.

Dial Button

Dial Button widget groups different action buttons into a single 'drop-down' button.

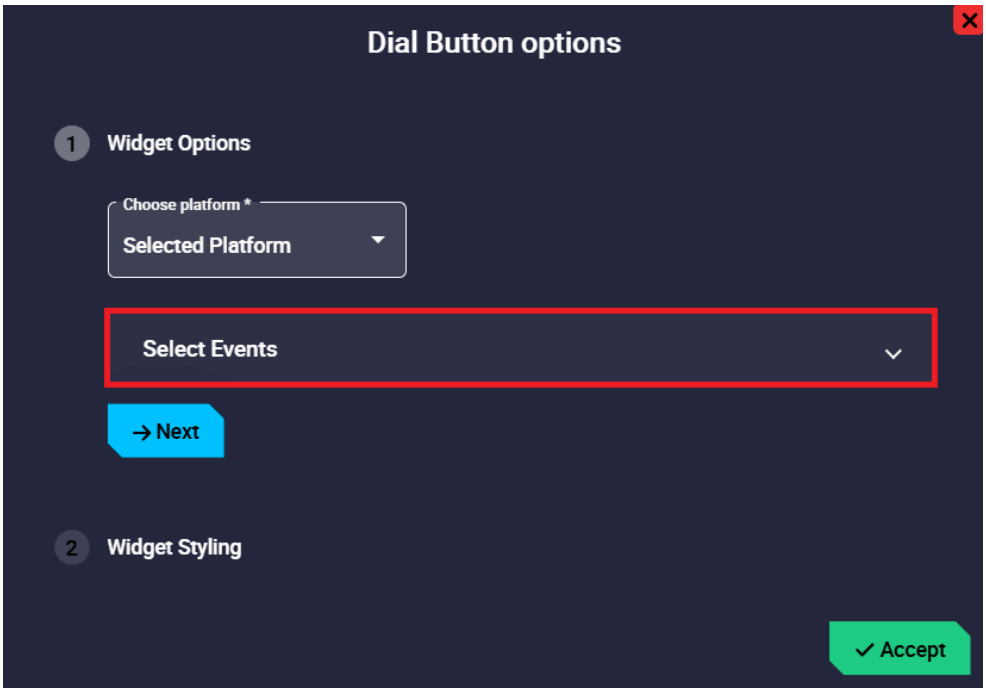
Dial Button

When clicked, the action buttons will appear/disappear.

Dial Button - Action buttons

Right click on the **widget** to access its options (**Duplicate**, **Edit** and **Remove**).

- **Options → Edit**: This allows the user to access the Dial Button configuration menu.
  This widget has **extra edit parameters** compared to the ones described in the Widgets common configuration.



Dial Button configuration

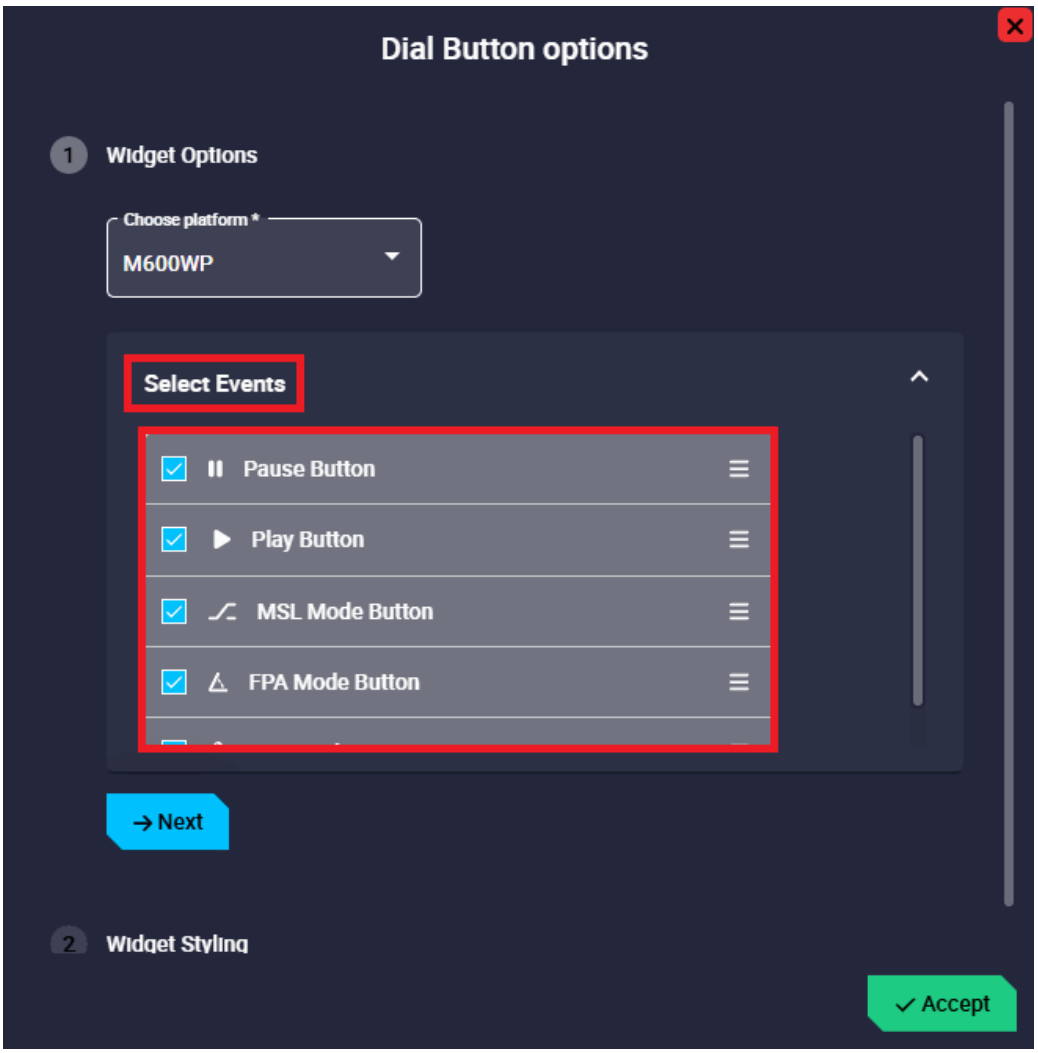
  - **Widget Options → Select Events**: Select the desired button events that will be displayed as action buttons on the Dial Button.

These events must be previously configured as automations in the **1x PDI Builder** software.

An example is given below:

Dial Button example

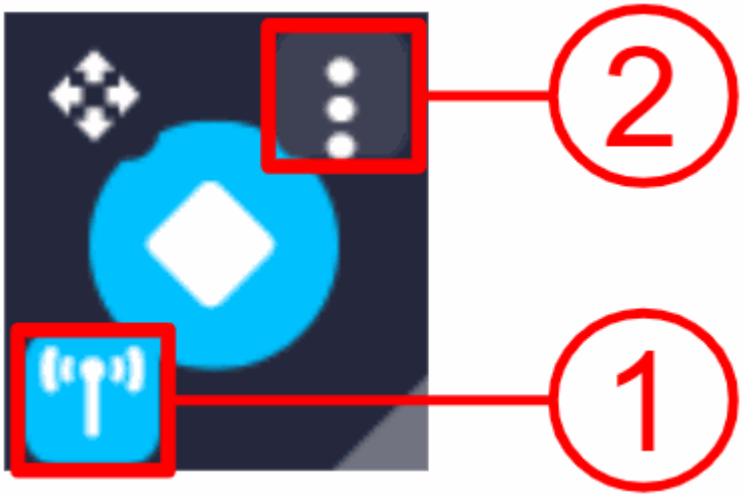
These action buttons can be ordered in the configuration:



Dial Button configuration example

Stick Buttons

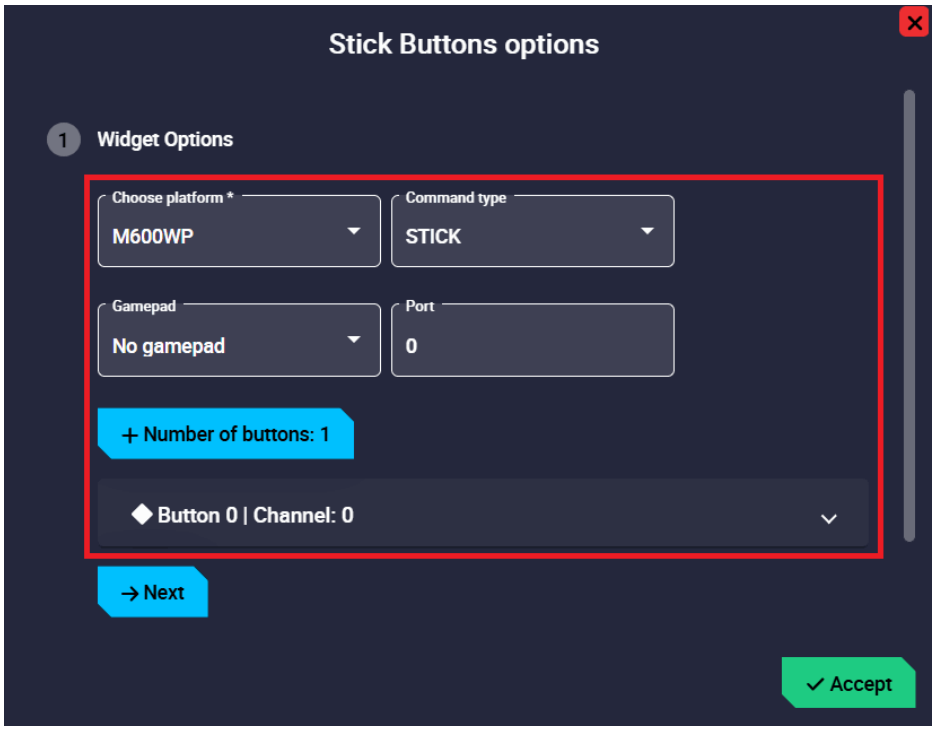
Stick Buttons widgets are created to map and manage the physical buttons of a connected controller to trigger specific actions or variables within the system.



Stick Buttons

On the stick buttons display, the icons allow for the following functions:

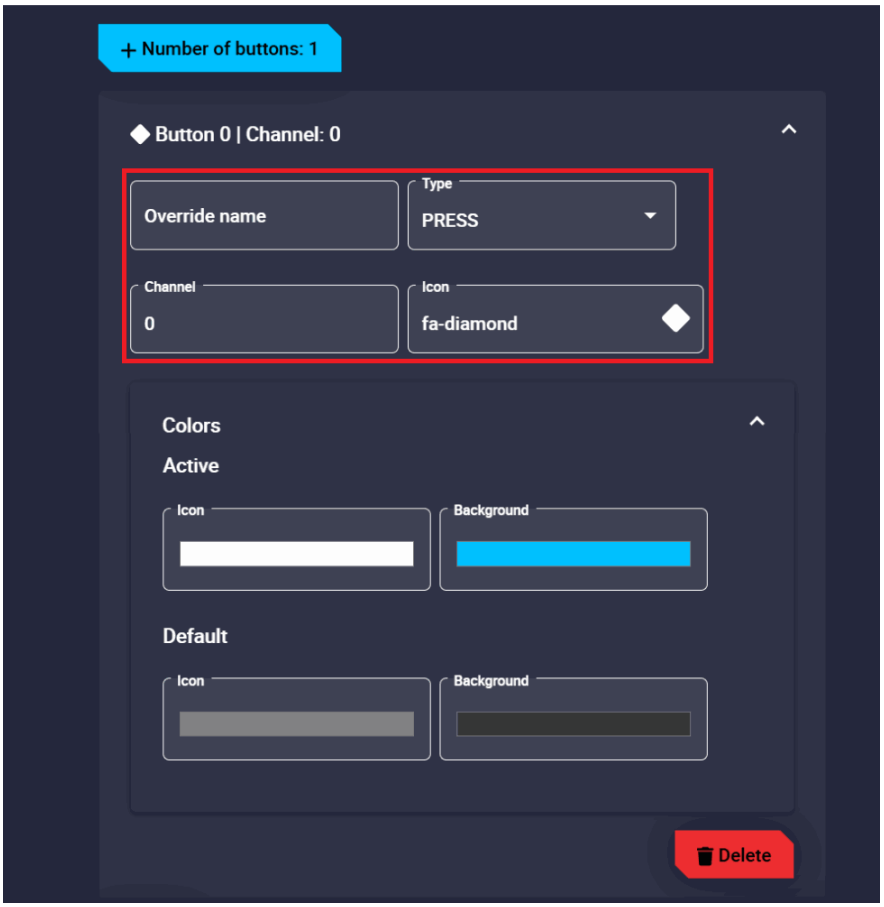
1. **Send command enable/disable**: Enables or disables the transmission of button commands to the platform.
2. **Options**:
   - **Duplicate**: Duplicates this widget.
   - **Edit**: Permits access to the Stick Buttons configuration menu.



Stick Buttons - Options

- **Widget Options**: This widget has extra edit parameters compared to the ones described in the Widgets common configuration.
  - **Choose platform**: Permits the selection of the specific platform for which the widget is configured.
  - **Command type**: Allows for the selection of the command category.
  - **Gamepad**: Permits the selection of the desired physical gamepad from the list of connected devices.
  - **Port**: As it is possible to have more than one stick configured, each transmitter must be configured on a different port.

    This port must match the port configured in the **1x PDI Builder** software. For more information, see the Output - Stick section of the **1x PDI Builder** user manual.
  - **Number of buttons**: Defines the total number of physical buttons to be configured for this widget.



Stick Buttons

  - **Override Name**: Assigns a custom label to identify the button.

        - **Type**: Determines the activation logic of the command, defining whether the action should be momentary (Press) or toggled (Switch).
        - **Channel**: Defines the controller channel associated with the command.
        - **Icon**: Allows selecting a graphic symbol to visually represent the function.
        - **Colors**: Defines the button's visual appearance to provide immediate feedback on its current operational status.
    - **Remove**: Deletes this widget.

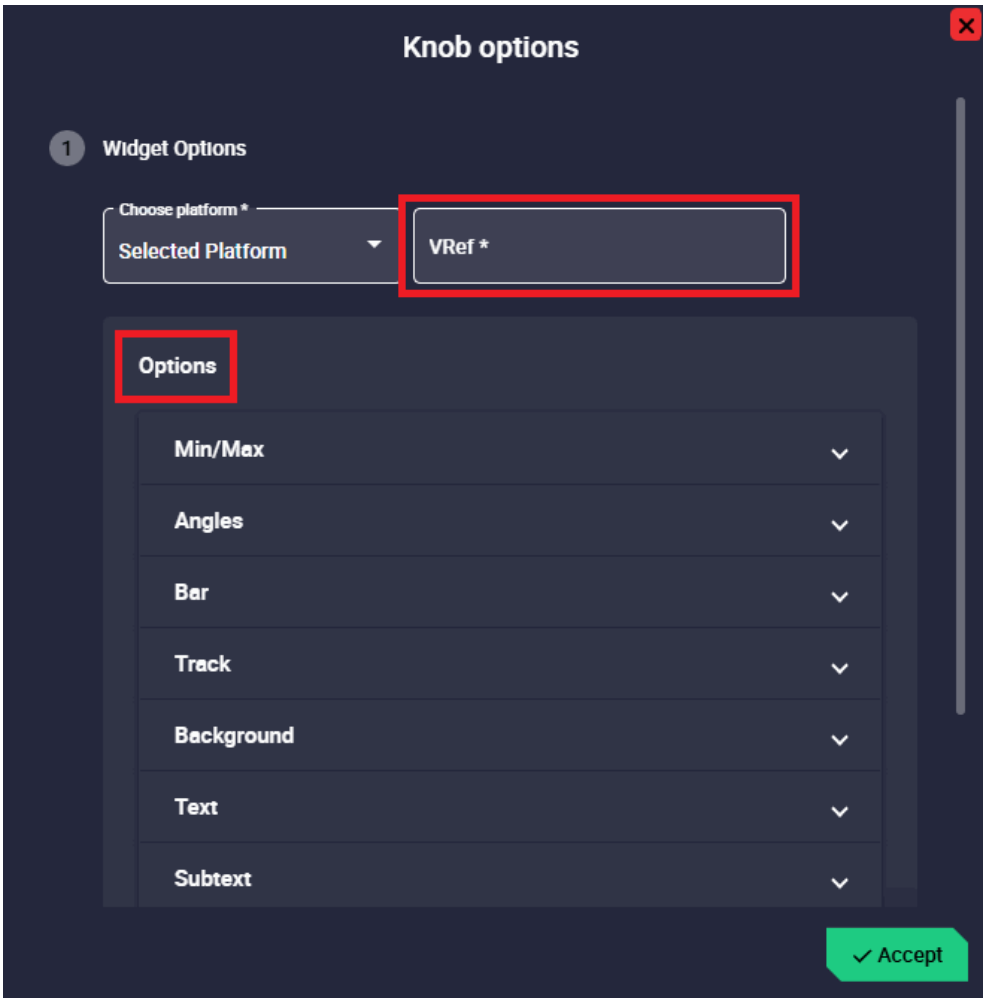
Knob

Knob widget is similar to the slider widget but in circular format. It also allows the user to choose a certain variable and change its value by simply moving the circular bar to the desired value from the workspace during a flight.

Knob

Right click on the **widget** to access its options (**Duplicate**, **Edit** and **Remove**).

- **Options** → **Edit**: This allows the user to access the Knob configuration menu.



Knob configuration

- **Widget Options**

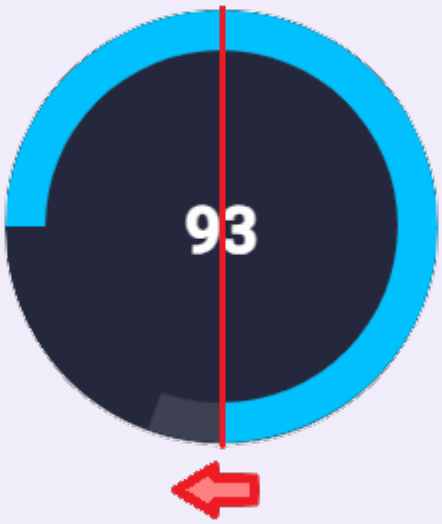
  This widget has **extra edit parameters** compared to the ones described in the Widgets common configuration.

  - **VRef**: Users can choose any real, integer or bit variable to be displayed on the knob.
  - **Unit**: Unit of measurement of the displayed variable.

    > ⓘ **Note**
    > This parameter is only available with Real variables.

  - **Options**: This widget is highly customizable by simply modifying the following parameters:
    - **Min/Max**: Minimum and maximum values displayed in the knob. Defaults are 0 and 100 respectively.

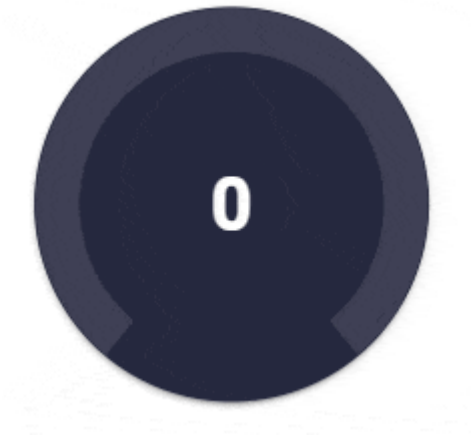
- **Angles**: Start and end angle of the position of the minimum and maximum values respectively. Defaults are -140 and 140 respectively.

> **Important**
>
> If the defined Start angle is a negative angle, the widget will **not allow** to pass from the right to the left side, at the most southern point. That is:
>
> 
>
>
> **Knob - Negative start angle**

- **Bar**:
  - **Cap**: This parameter allows users to choose whether they want the **ends of the bar** to be **right angles** or **rounded**.

    ⇒ **100% rounded = 50% of the width** parameter.
  - **Width**: The user can modify the width of the bar as desired. By default it is set to 20.
  - **Step**: Here users must enter the step by which they want the bar to move. By default it is set to 1.
  - **Color**: This is the color in which the bar is drawn.
  - **Display previous bar**: If enabled, the bar stays colored at the position of the previously set value while the bar is moving to set the new value. Default is enabled.
    - **Color**: Color of the previous bar.

    An example is given below:



Knob configuration - Display previous bar option enabled

- **Track**: This parameter refers to the bar between the minimum and maximum values defined above, which is always below the bar that indicates the desired value.
  - **Track cap**: This parameter allows users to choose whether they want the **ends of the track bar** to be **right angles** or **rounded**.

    ⇒ **100% rounded = 50% of the track width** parameter.
  - **Track width**: The user can modify the width of the track bar as desired. By default it is set to 20.
  - **Color**: This is the color in which the track bar is drawn.
- **Background**:
  - **Color**: Color of the knob widget background.
  - **Full**: If enabled, the entire knob widget is colored with the previously defined color.

    Otherwise, the knob will only be colored between the previously defined start and end angles. By default it is enabled.

    In the following example, the start and end angles are set to -90 and 90 respectively.

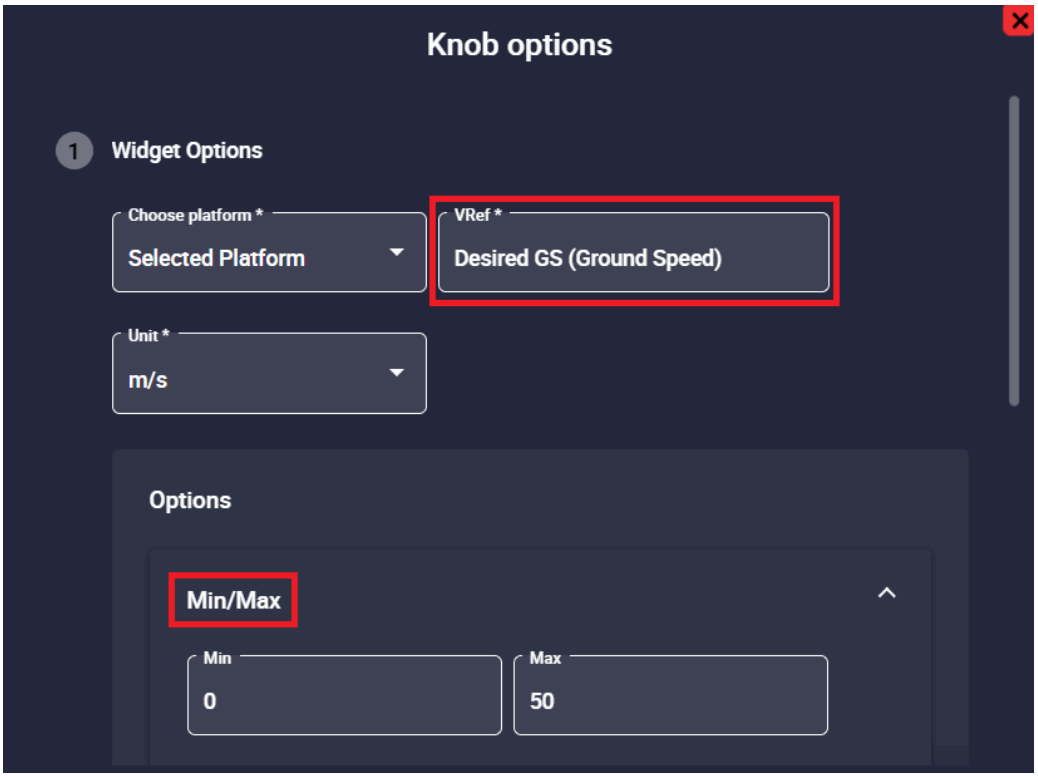
Knob configuration - Full option enabled/disabled respectively

- **Text**: This text refers to the value of the selected variable displayed in the knob.
  - **Display**: If enabled, displays on the knob the value of the selected variable. By default it is enabled.
  - **Size**: The size of the text value can be modified. Default is 32.
  - **Weight**: The weight of the text value can be modified. The available options go from 100 to 900, default is 700.
  - **Color**: Color in which the value is displayed.
- **Subtext**: This subtext refers to the name of the selected variable displayed in the knob.
  - **Display**: If enabled, displays on the knob the name of the selected variable. By default it is enabled.
  - **Offset**: Vertical position of the selected variable name can be adjusted as desired. Default is 7.
  - **Size**: The size of the selected variable name can be changed. By default it is set to 14.
  - **Weight**: The weight of the selected variable name can be changed. The available options go from 100 to 900, default is 700.
  - **Color**: Color in which the selected variable name is displayed.

An example is given below:

Knob example

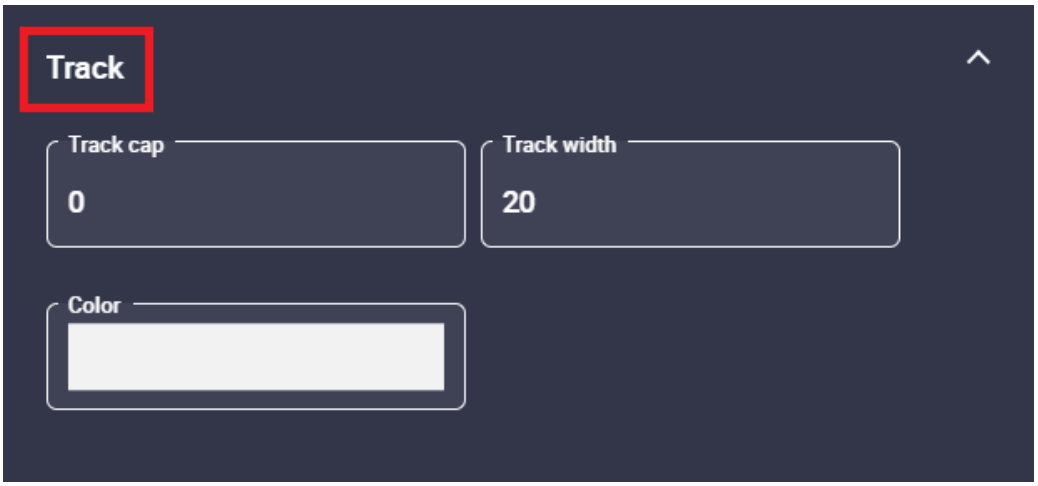
The configuration for this example is as follows:



Knob configuration example - Variable and Min/Max option

Knob configuration example - Angles options

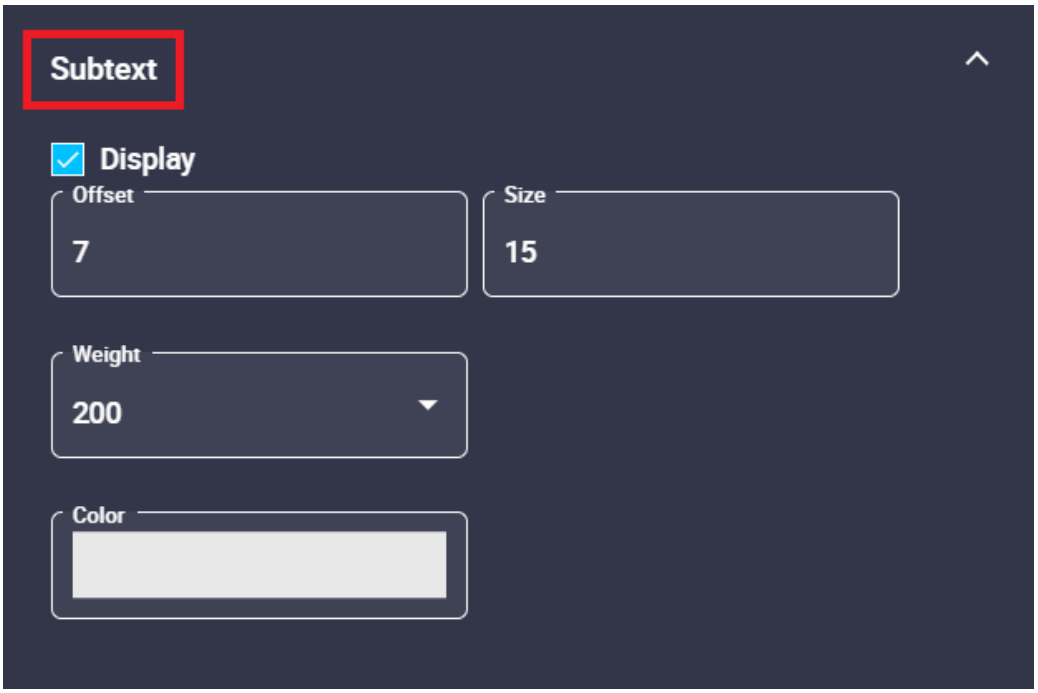
Knob configuration example - Bar options



Knob configuration example - Track options

Knob configuration example - Background options

Knob configuration example - Text options

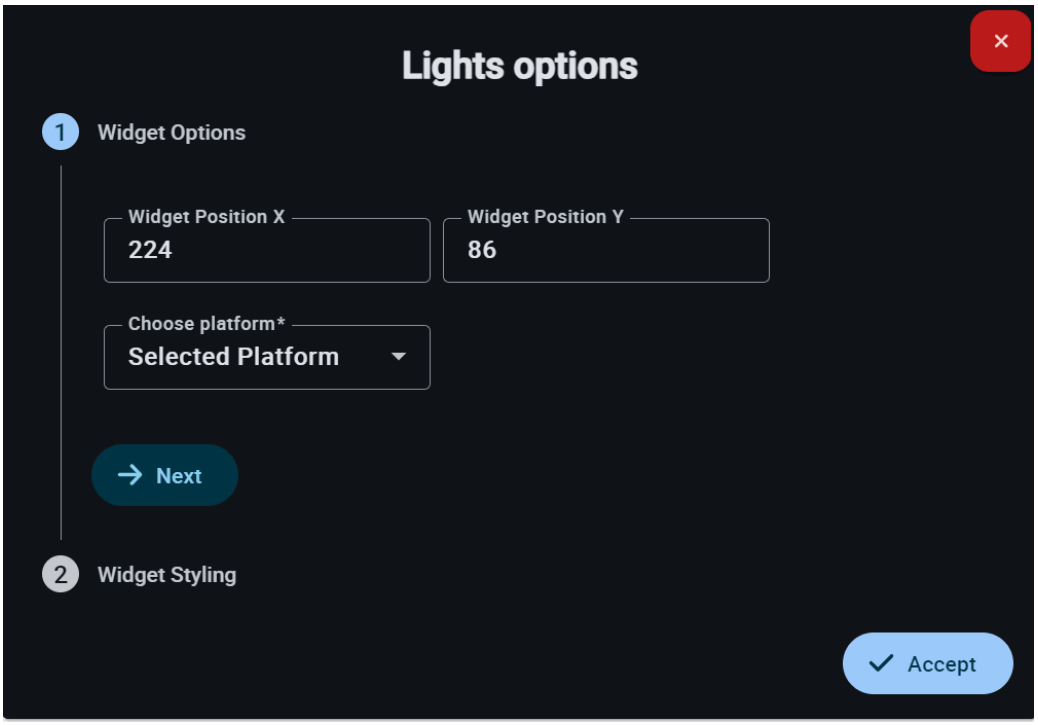


Knob configuration example - Subtext options

Vehicle

Lights

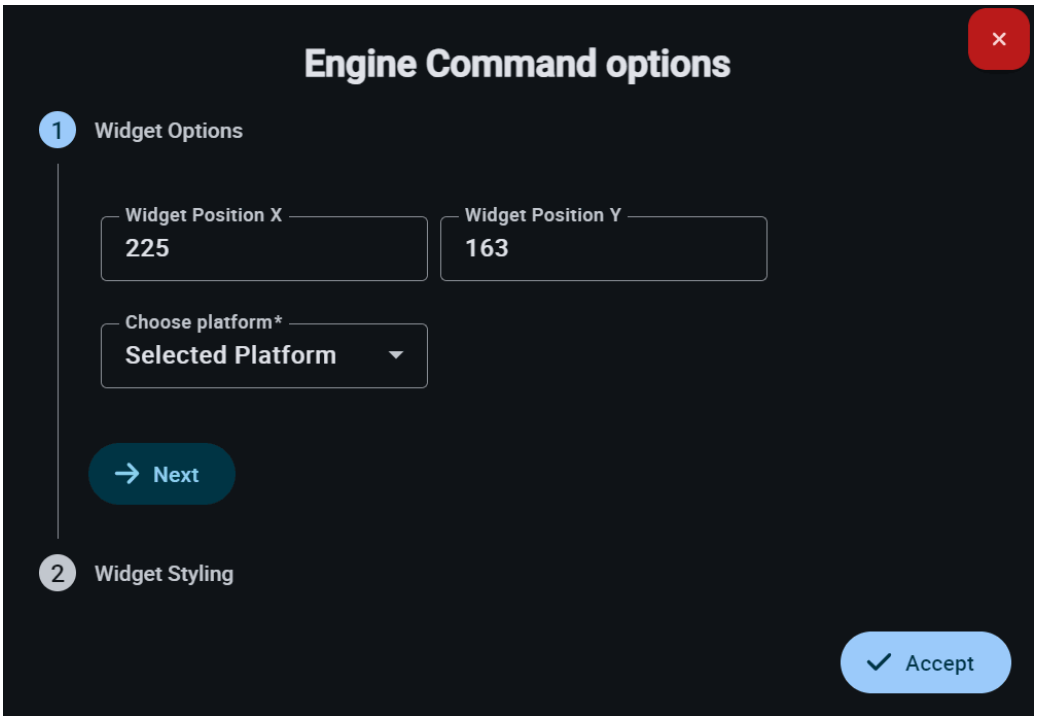
The Lights Options menu allows the user to configure the position and data source for the lighting system widget.



Lights options

Engine Command

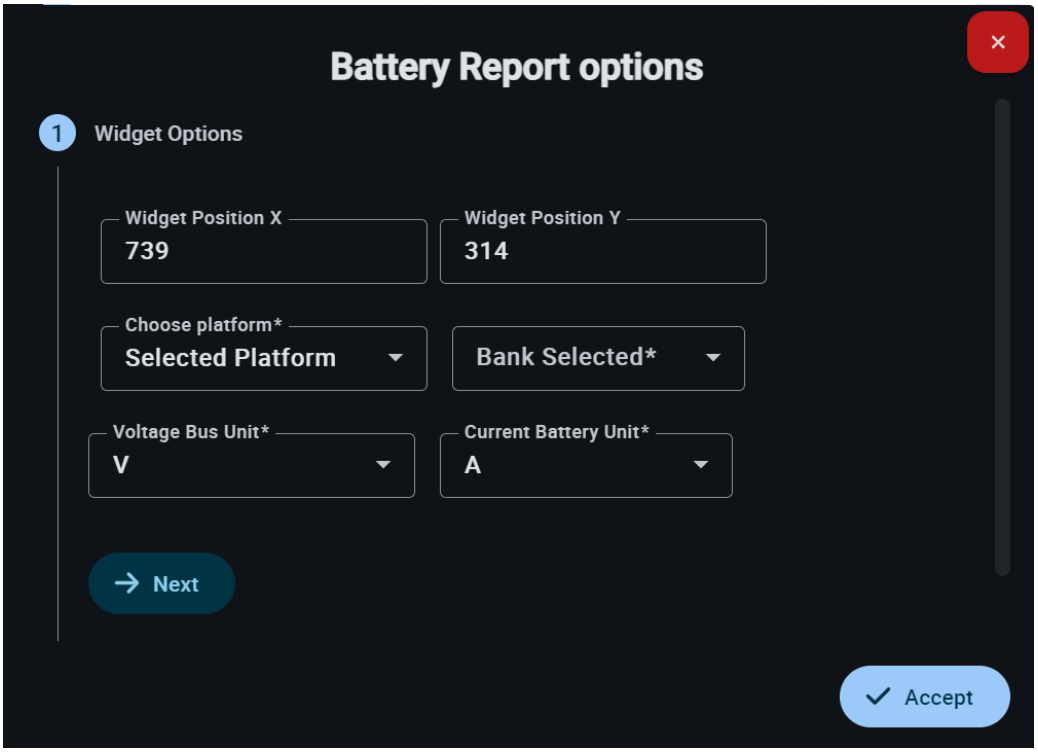
Engine Command allows the operator to configure the location and data source for the engine control widget on the display.



Engine Command options

Battery Report

The Battery Report Options menu enables the operator to configure the monitoring of the system's power parameters on the display.



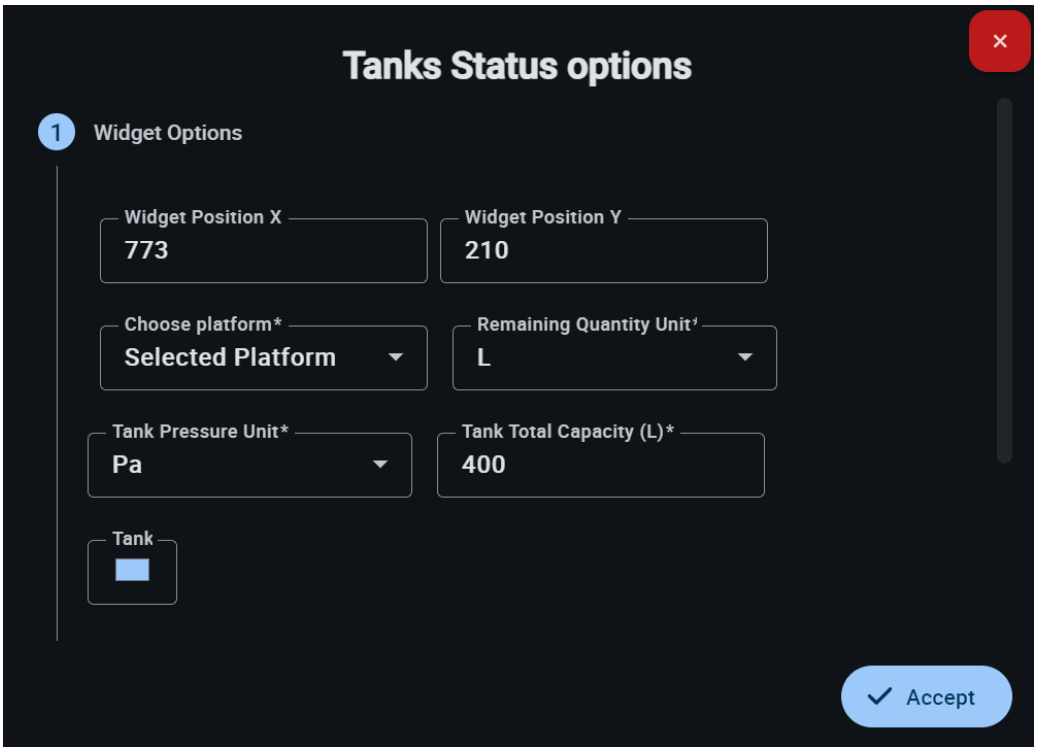
Battery Report options

This widget has **extra edit parameters** compared to the ones described in the Widgets common configuration.

- **Bank Selected**: Allows the user to select which bank or battery pack to monitor, in case the system features multiple independent units.
- **Voltage Bus Unit**: Defines the measurement unit for the system's bus voltage in V or mV.
- **Current Battery Unit**: Defines the measurement unit for the current delivered by the battery in A or mA.

Tanks Status

Tanks Status options allows the operator to configure the monitoring of onboard tanks (fuel or other liquids) on the Display.



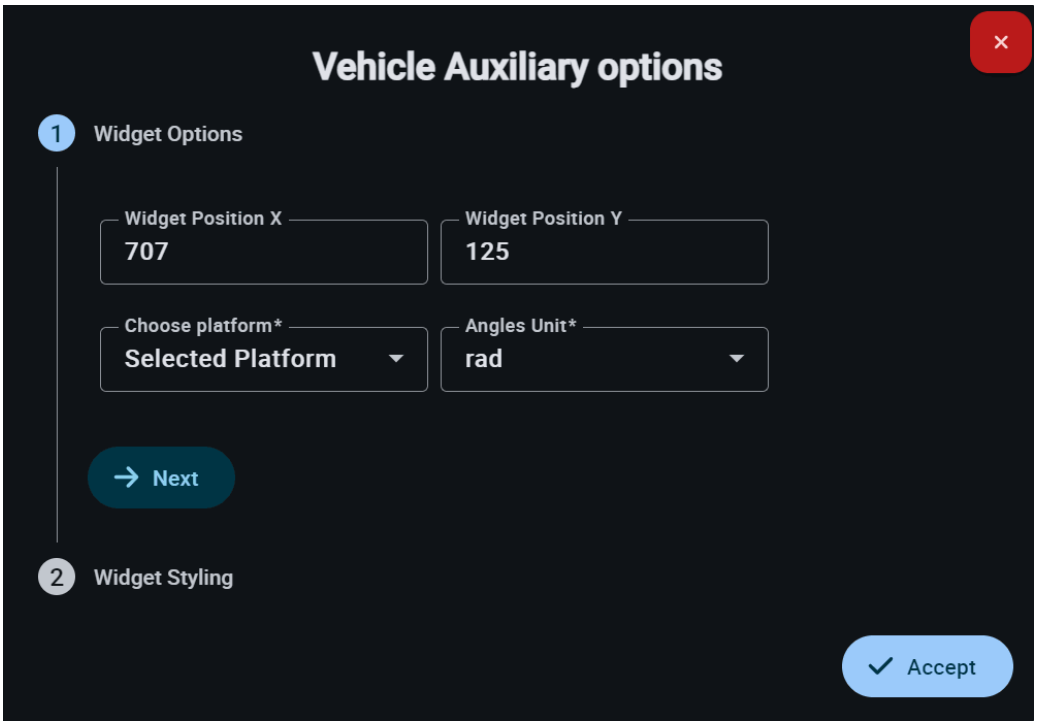
Tanks Status options

This widget has **extra edit parameters** compared to the ones described in the Widgets common configuration.

- **Remaining Quantity Unit**: Defines the measurement unit for the remaining liquid quantity.
- **Tank Pressure Unit**: Specifies the measurement unit for the tank's internal pressure.
- **Tank Total Capacity (L)**: Indicates the total maximum capacity of the tank expressed in liters.
- **Tank**: Allows the user to select the representative color for the tank in the graphic interface.

Vehicle Auxiliary

Vehicle Auxiliary options menu enables the user to configure the display of the vehicle's auxiliary data.



Vehicle Auxiliary - Options

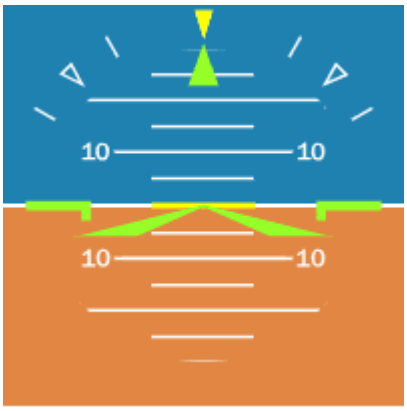
This widget has **extra edit parameters** compared to the ones described in the Widgets common configuration.

- **Angles Unit**: Defines the measurement unit for auxiliary angles on the display.

Flight instruments

Attitude

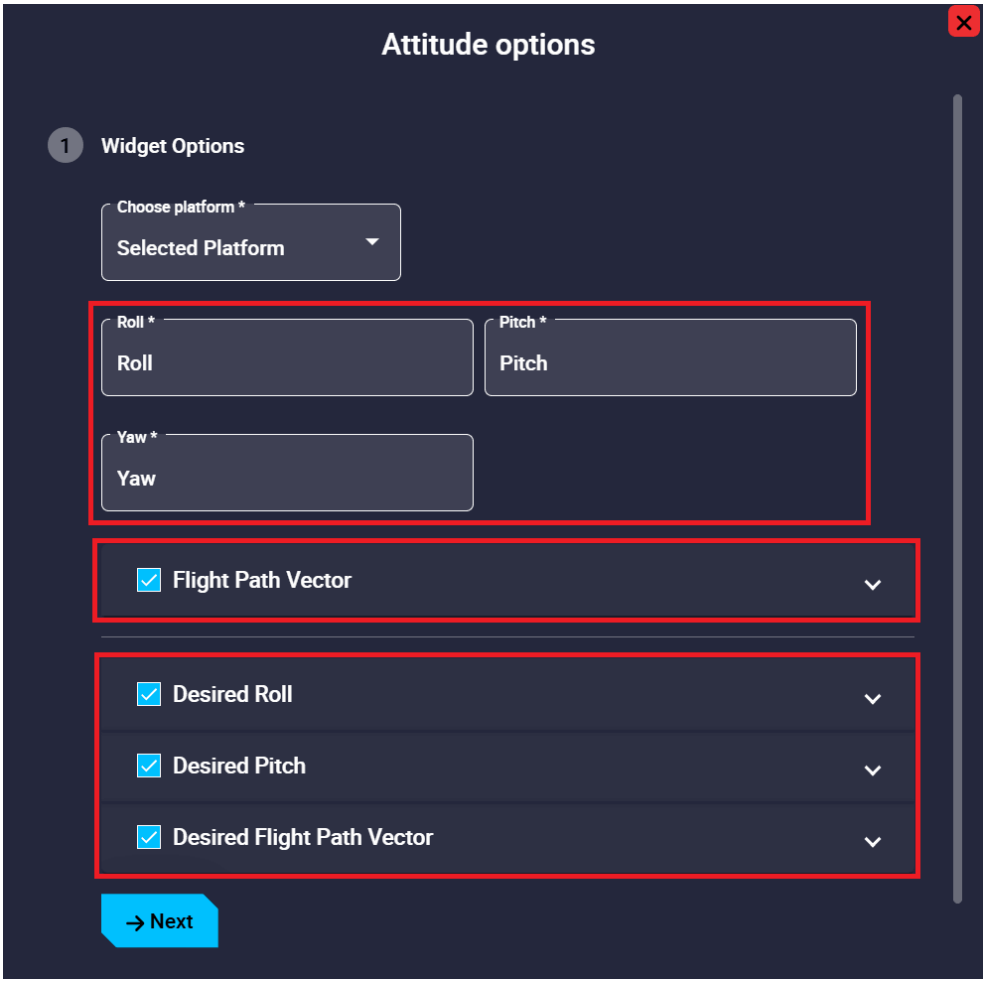
The attitude widget, commonly known as Primary Flight Display (**PFD**) or 'artificial horizon', represents graphically the attitude of the aircraft (**roll** and **pitch**).



Attitude

Right click on the **widget** to access its options (**Duplicate**, **Edit** and **Remove**).

- **Duplicate**: Duplicates this widget.
- **Edit**: This allows the user to access the Attitude configuration menu.



Attitude configuration

- **Widget Options**:

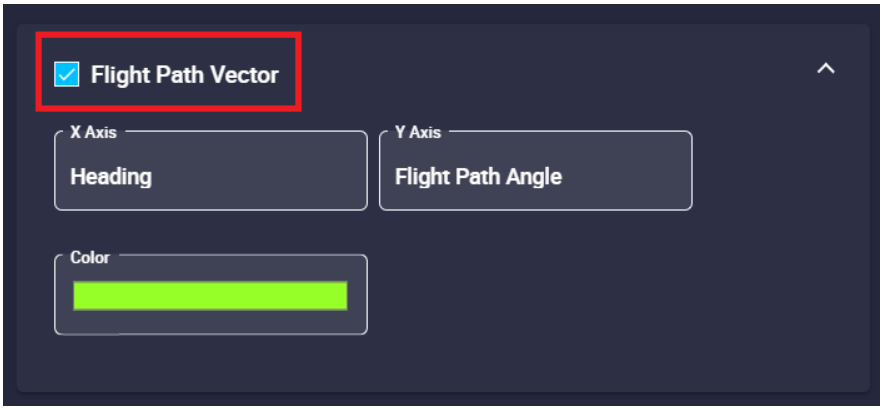
  This widget has **extra edit parameters** compared to the ones described in the Widgets common configuration.

  - **Roll**: Users can choose a variable from the **real angular variables** of the system to be displayed as **the roll of the widget**.

    By default, the variable Roll is displayed.
  - **Pitch**: Users can choose a variable from the **real angular variables** of the system to be displayed as **the pitch of the widget**.

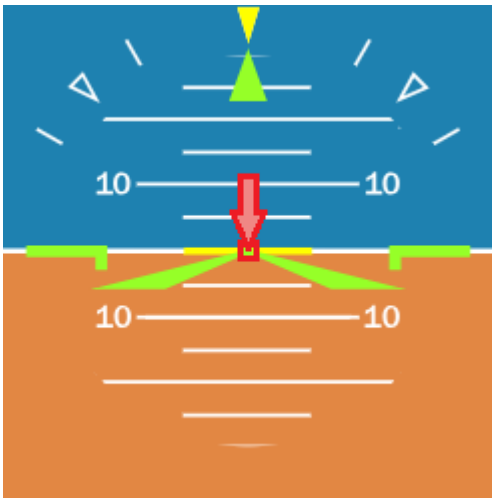
    By default, the variable Pitch is displayed.

- **Yaw**: Users can choose a variable from the **real angular variables** of the system to be displayed as **the yaw of the widget**.
  By default, the variable Yaw is displayed.
- **Flight Path Vector**: By activating/deactivating this parameter, it is possible to show/hide the flight path vector in the attitude widget.
  By default it is activated.



Attitude configuration - Flight Path Vector

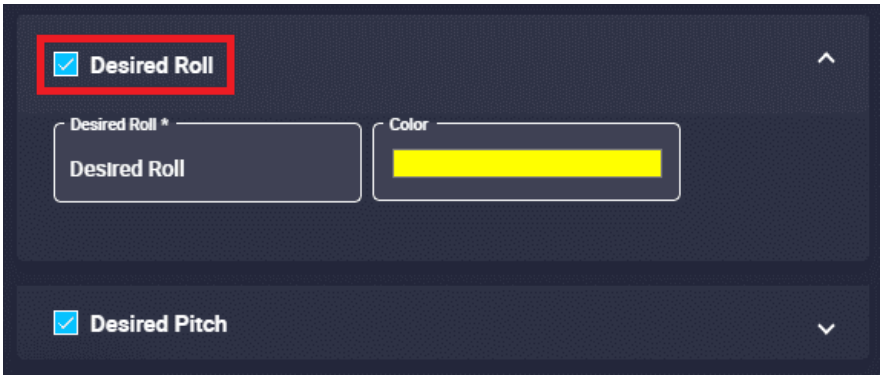
- **Flight Path Vector**: A variable from the real angular variables of the system can be selected to be displayed as the flight path vector of the widget.
  By default, the variable Flight Path Vector is displayed.
- **Color**: The color in which the Flight Path Vector is displayed can be customized. By default, green is selected.



Attitude configuration - Flight Path Vector example

- **Desired Roll**: By activating/deactivating this parameter, it is possible to show/hide the desired roll together with the roll in the attitude widget.

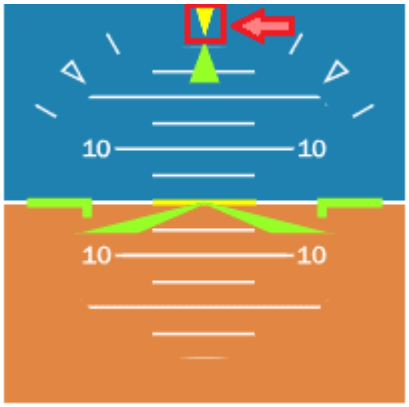
  By default it is activated.



Attitude configuration - Desired Roll

  - **Desired Roll**: Users can choose a variable from the **real angular variables** of the system to be displayed as **the desired roll of the widget**.

    By default, the variable Desired Roll is displayed.
  - **Color**: The color in which the desired roll is colored can be customized by the user. By default, yellow is selected.



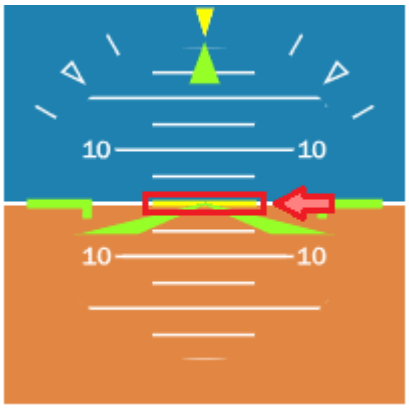
Attitude configuration - Desired Roll example

- **Desired Pitch**: By activating/deactivating this parameter, it is possible to show/hide the desired pitch together with the pitch in the attitude widget.

  By default it is activated.

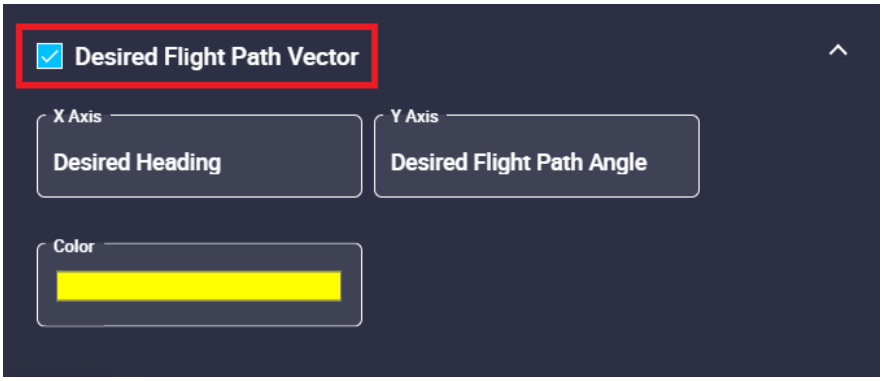
Attitude configuration - Desired Pitch

- **Desired Pitch**: Users can choose a variable from the **real angular variables** of the system to be displayed as **the desired pitch of the widget**.
  By default, the variable Desired Pitch is displayed.
- **Color**: The color in which the desired pitch is colored can be customized by the user. By default, yellow is selected.



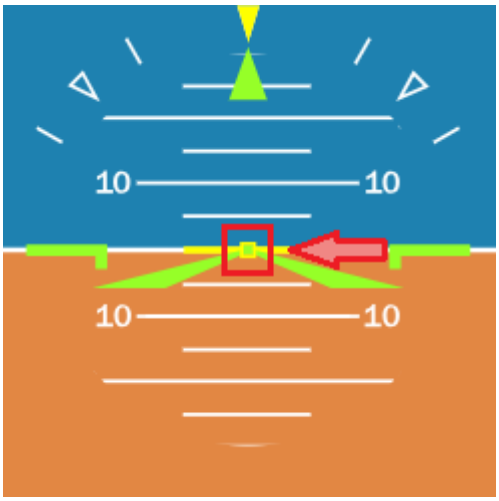
Attitude configuration - Desired Pitch example

- **Desired Flight Path Vector**: By activating/deactivating this parameter, it is possible to show/hide the desired flight path vector together with the flight path vector in the attitude widget.
  By default it is activated.



Attitude configuration - Desired Flight Path Vector

- **Desired Flight Path Vector**: Users can choose a variable from the **real angular variables** of the system to be displayed as the desired flight path vector of the widget.
  By default, the variable Desired Flight Path Vector is displayed.
- **Color**: The color in which the Desired Flight Path Vector is colored can be customized by the user. By default, yellow is selected.



Attitude configuration - Desired Flight Path Vector example

- **Remove**: Deletes this widget.

Heading

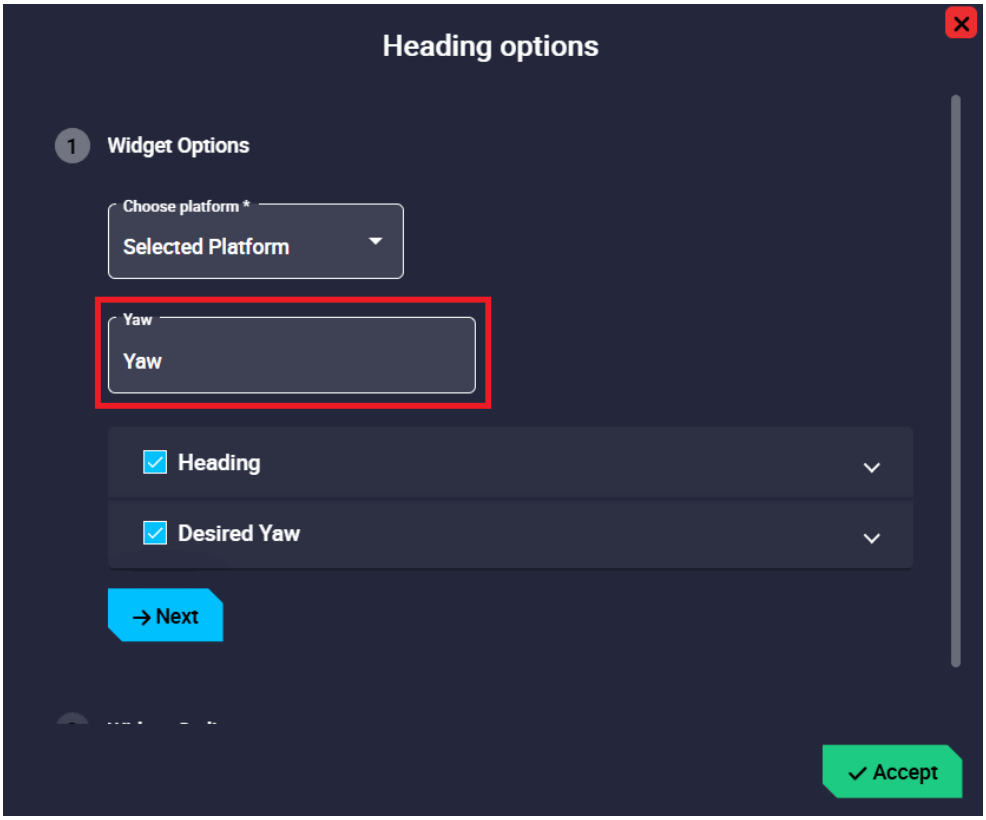
The heading widget, commonly known as **compass**, usually shows the platform's yaw relative to the magnetic north.

Heading

Right click on the **widget** to access its options:

- **Duplicate**: Duplicates this widget.
- **Edit**: This allows the user to access the Heading configuration menu.

  This widget has **extra edit parameters** compared to the ones described in the Widgets common configuration.



Heading configuration

  - **Heading**: By activating/deactivating this parameter, it is possible to show/hide the desired heading together with the heading in the attitude widget.

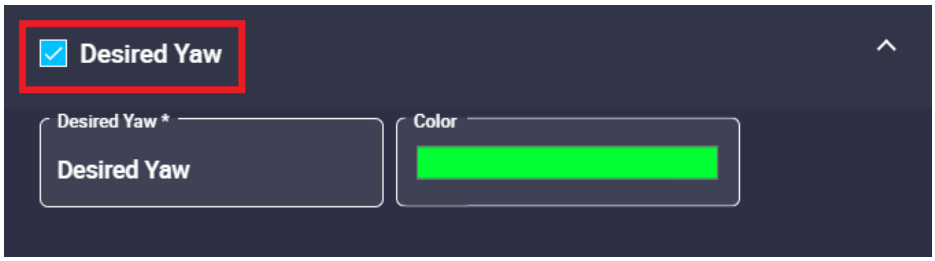
    By default it is activated.

Heading configuration - Heading

    - **Color**: The color in which the heading is colored can be customized by the user. By default, yellow is selected.

Heading configuration - Heading example

- **Desired Yaw**: By activating/deactivating this parameter, it is possible to show/hide the desired yaw together with the yaw in the attitude widget.
  By default it is activated.



Heading configuration - Desired Yaw

  - **Color**: The color in which the desired yaw is colored can be customized by the user. By default, green is selected.

Heading configuration - Desired Yaw example

- **Widget Options → VRef**: Users can choose a variable from the the **real angular variables** in the system to be displayed.
  By default, the variable Yaw is displayed.

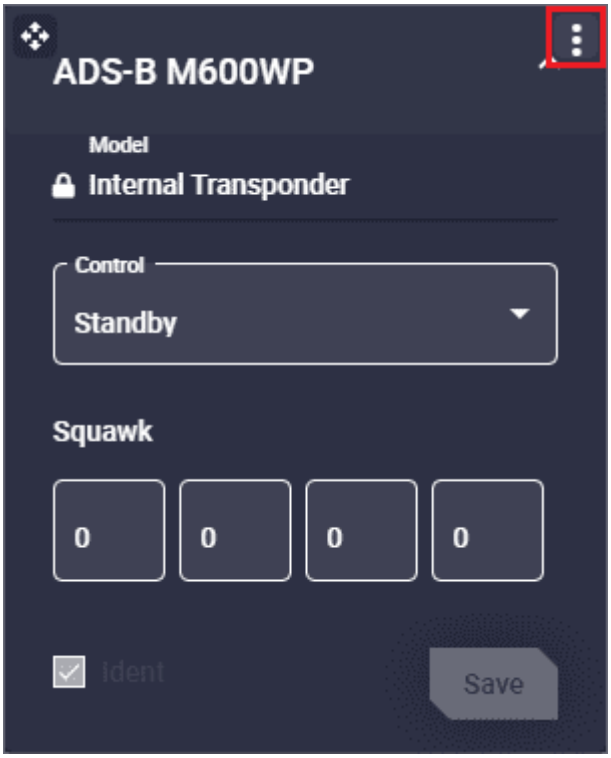
- **Remove**: Deletes this widget.

ADS-B

ADS-B widget inputs the required information into the ADS-B system.

ADS-B

Note that, as this widget behaves like a drop-down menu, clicking on the icon will display all ADS-B parameters:



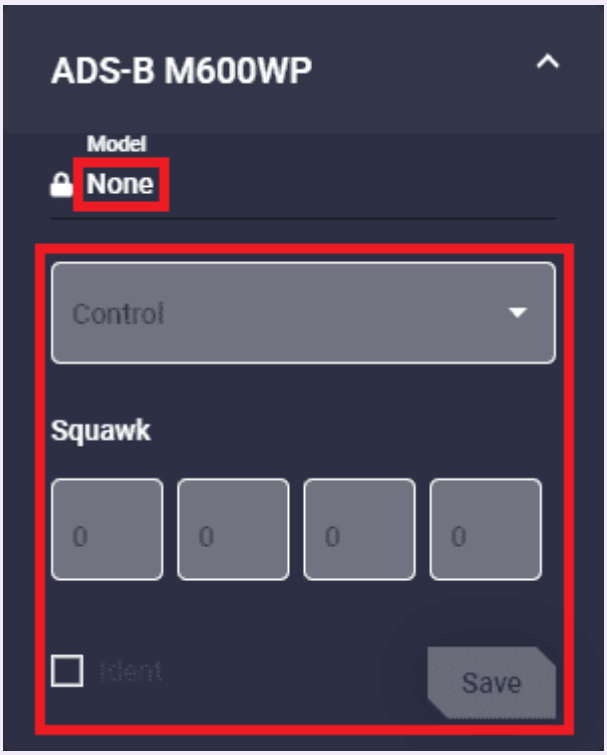
ADS-B deployed

Click the button to access its options (**Duplicate**, **Edit** and **Remove**).

- **Options**:
  - **Duplicate**: Duplicates this widget.
  - **Edit**: This allows the user to access the ADS-B configuration menu. This widget has only the 'basic' configuration described in the Widgets common configuration.
  - **Remove**: Deletes this widget.

Important

If the ADS-B has not been configured in the **1x PDI Builder** software, this widget will appear as unavailable as shown in the figure below.



ADS-B - No ADS-B configured

This widget has different parameters to modify the ADS-B configuration. For more information on the ADS-B configuration, please refer to the Transponder/ADS-B - Devices section of the **1x PDI Builder** user manual.

ADS-B configuration

- **Model**: This only indicates the ADS-B model that has been configured in the **1x PDI Builder** software.

  > **Important**
  > It **cannot** be modified from this widget, only in the **1x PDI Builder** configuration.

- **Control**: Users can change the transponder control type. It can be Standby, ADS-B In, ADS-B Out and ADS-B In/Out.

  > **Important**
  > The control modes displayed for setting this parameter depend on the selected ADS-B Model. Therefore, not all options described will always appear.

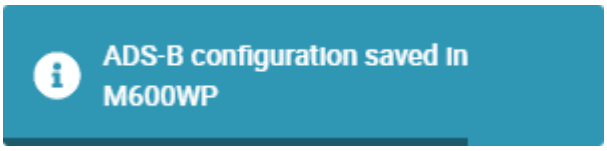
- **Squawk**: Users must introduce the Squawk Code provided. This is the transponder code to identify the flight.
  For certain type of flights and/or situations, specific transponder codes are used. These codes are four/digital octal numbers.

> **Important**
> This parameter may be configured depending on the ADS-B Model.

- **Ident**: This is an identification of the UAV at the request of ATC, in order to help them to locate the aircraft.

> **Important**
> The configuration of this option depends on the Model type selected.

- **Save**: After modifying any parameter, press here to save the changes into the configuration. The following message will appear:



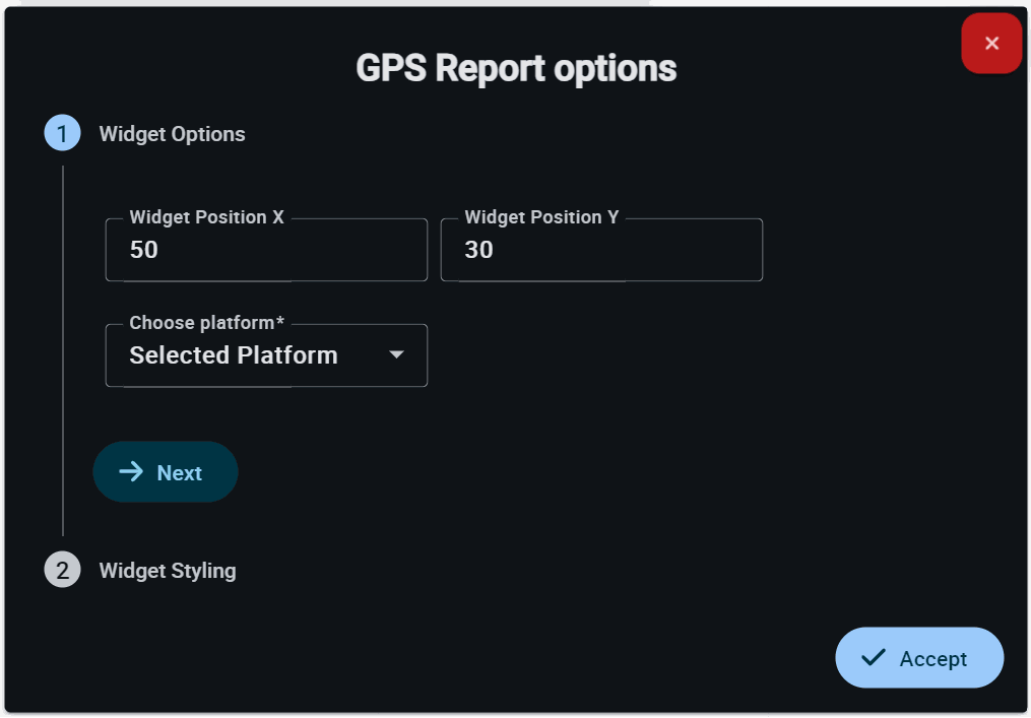
ADS-B configuration - Save message

GPS Report

The GPS Report Options menu enables the user to configure the location, data source, and styling of the global positioning widget on the Display

Right click on the **widget** to access its options:

- **Duplicate**: Duplicates this widget.
- **Edit**: This allows the user to access the Heading configuration menu.



GPS Report configuration

Wind

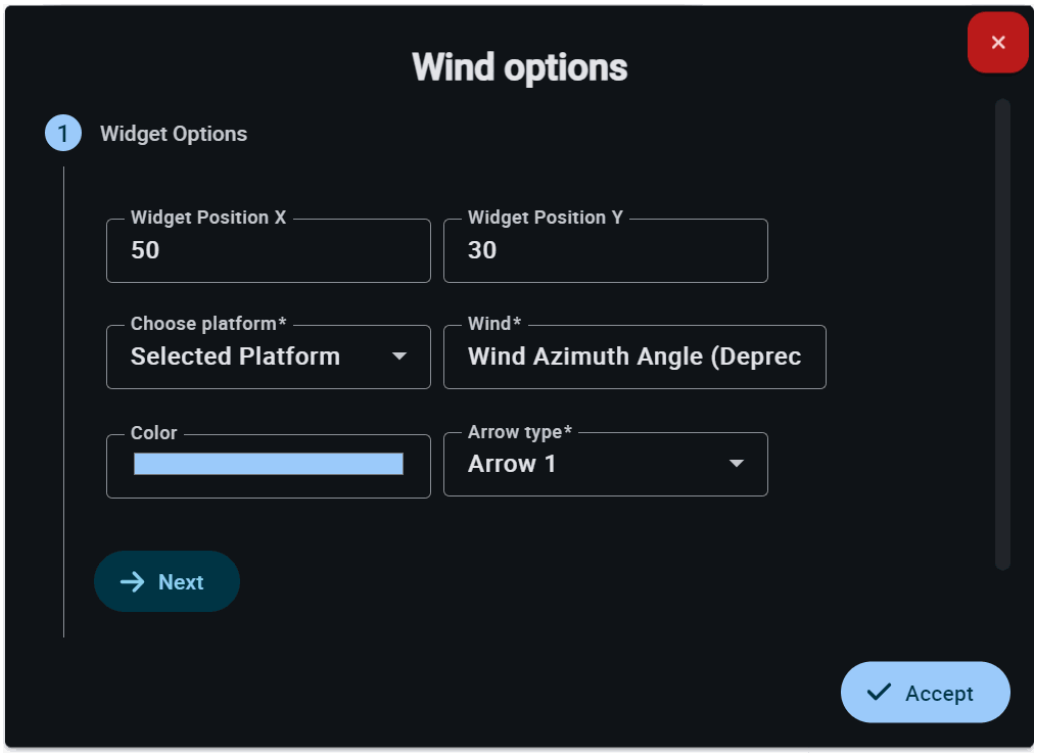
The Wind Options menu enables the operator to configure the location, data source, and graphic style of the wind telemetry widget on the display.

Wind Widgets

Right click on the **widget** to access its options:

- **Duplicate**: Duplicates this widget.
- **Edit**: This allows the user to access the Heading configuration menu.

This widget has **extra edit parameters** compared to the ones described in the Widgets common configuration.



Wind configuration

- **Wind**: Defines which telemetry variable is used to calculate wind direction.
- **Color: Allows**: the user to select the color of the graphic indicator to ensure optimal visibility against the map or background.
- **Arrow type**: Defines the graphic style of the wind arrow.

Payload

Gimbal Buttons

Gimbal Buttons widget groups different control buttons of the selected gimbal into a single 'drop-down' button.

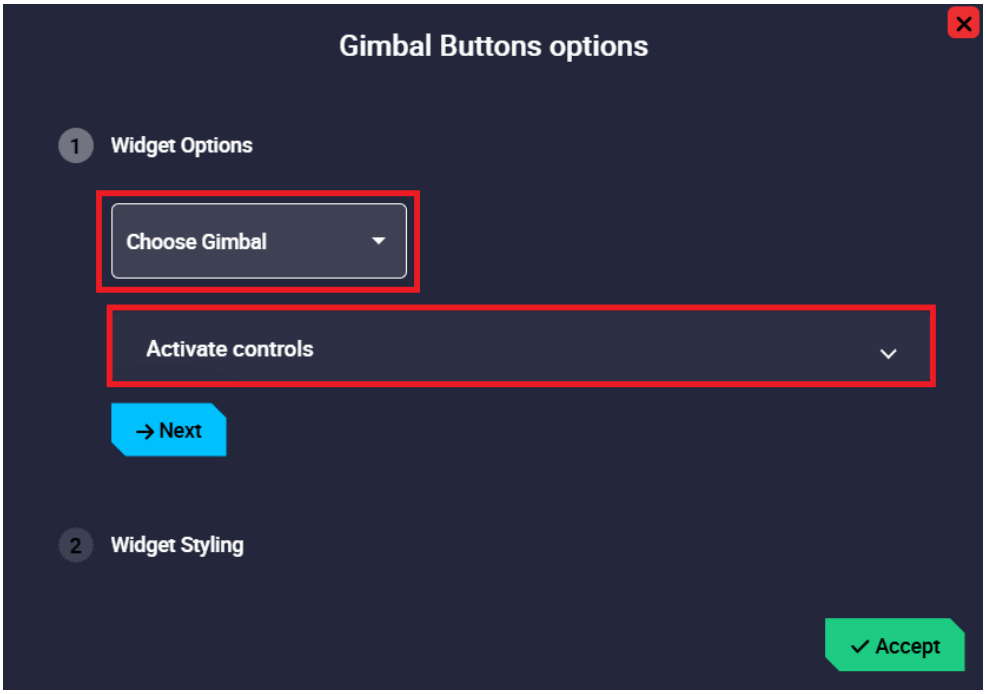
Gimbal Buttons

When clicked, the control buttons will appear/disappear.

Right click on the **widget** to access its options (**Duplicate**, **Edit** and **Remove**).

This widget has extra edit parameters compared to the ones described in Widgets common configuration.

- **Options → Edit**: This allows the user to access the Gimbal Buttons configuration menu.



Gimbal Buttons configuration

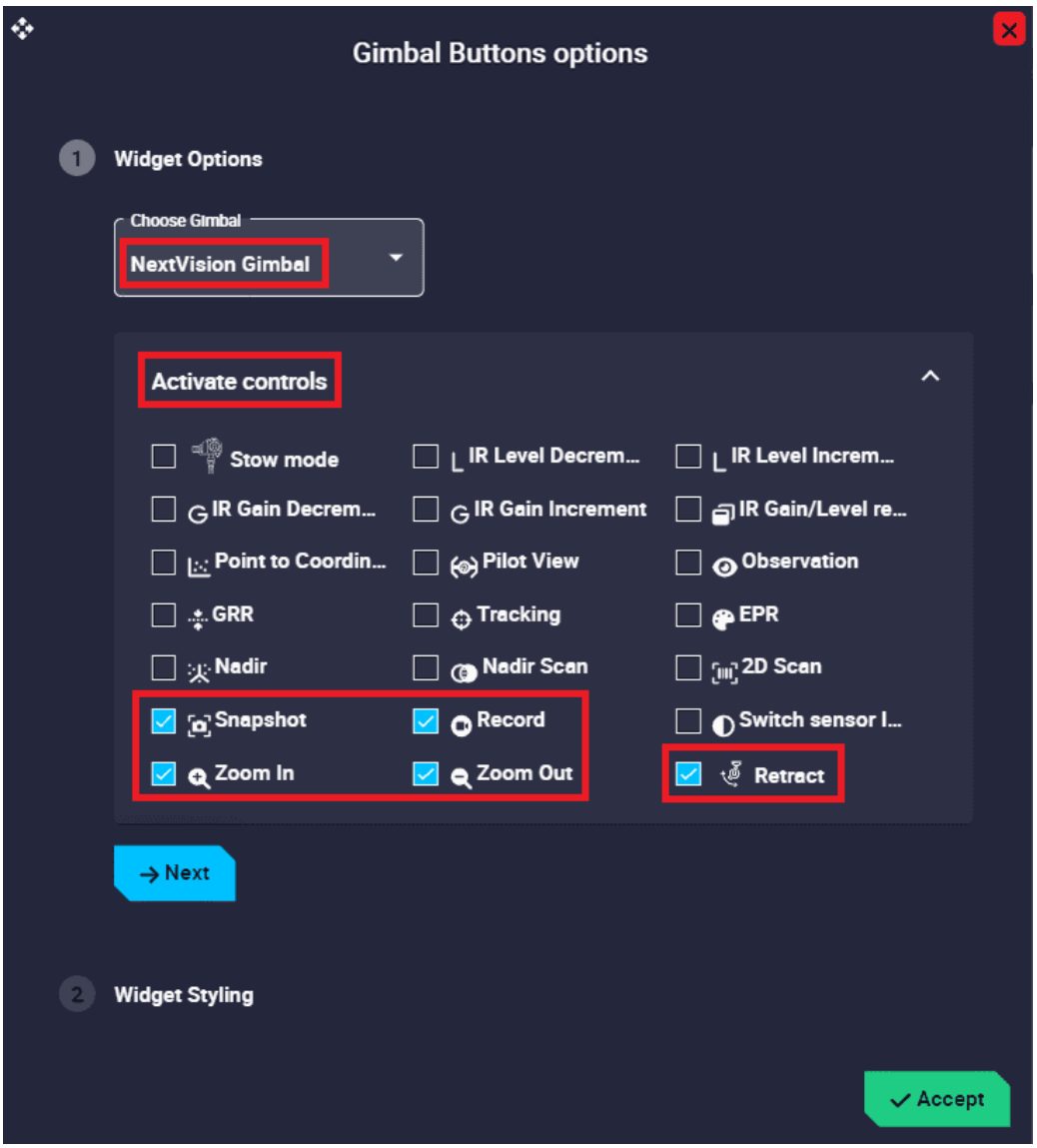
  - **Widget Options**:
    - **Choose Gimbal**: The user has to choose from the gimbal predefined list the gimbal for which the widget is configured.
    - **Activate controls**: Select the desired controls of the selected gimbal that will be displayed on the Gimbal Buttons.

> **Important**
> The controls shown here depend on the selected gimbal, each gimbal has its own control buttons.

Below is an example with the **NextVision gimbal**:

Gimbal Buttons example

These control buttons can be enabled/disabled in the configuration:



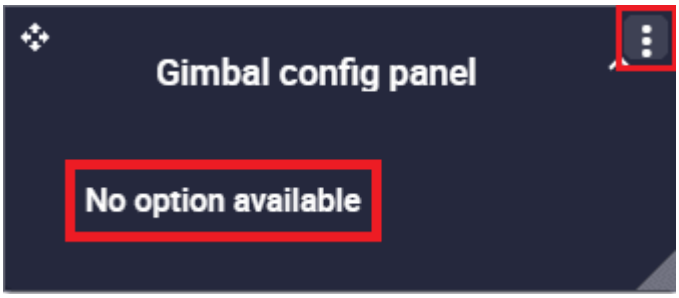
Gimbal Buttons configuration example

Gimbal Setup

Gimbal Setup widget is a gimbal configuration panel, where the user can choose which gimbal controls can be managed from **Veronte Ops**.

Gimbal Setup

Note that, as this widget behaves like a drop-down menu, clicking on the icon will display all gimbal controls:



Gimbal Setup deployed

> **Note**
>
> As no controls have been configured yet, No option available is displayed when deployed.

Click the button to access its options (**Duplicate**, **Edit** and **Remove**).

- **Options → Edit**: This allows the user to access the Gimbal Setup configuration menu.

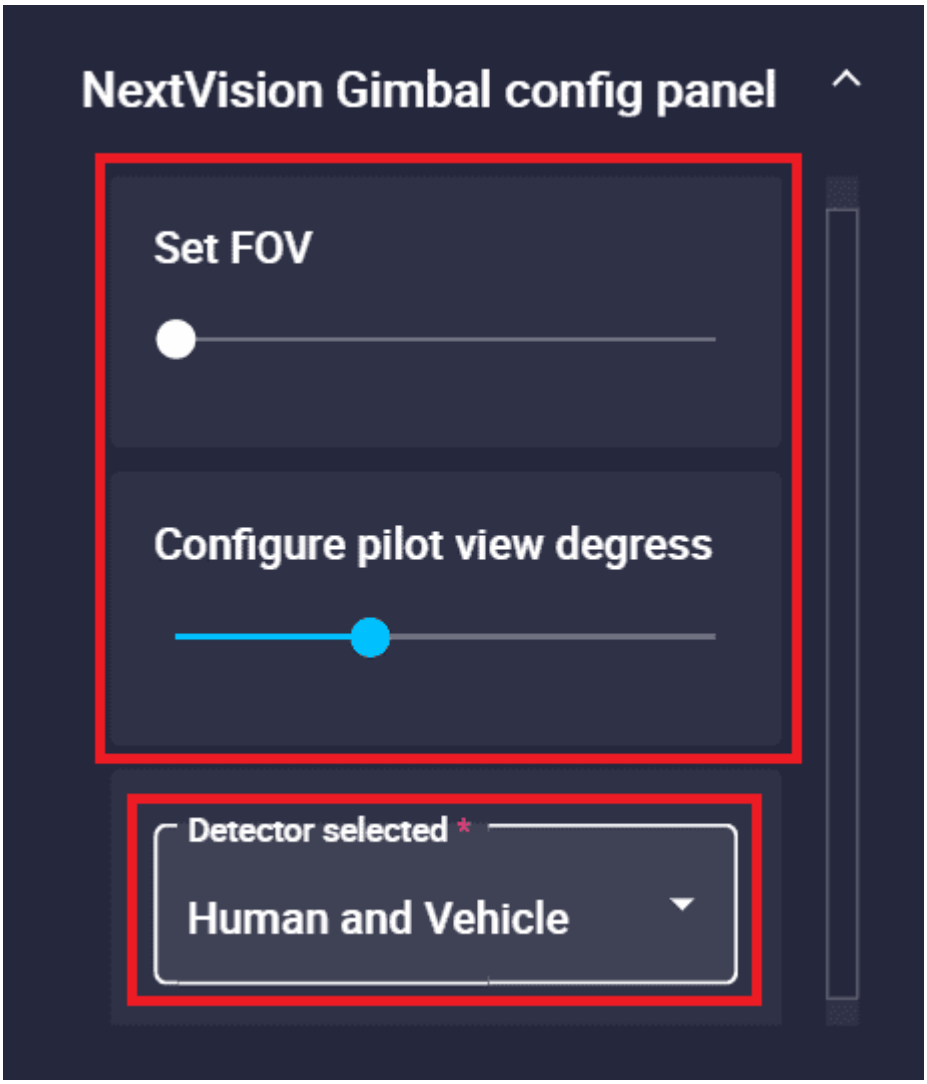
Gimbal Setup

This widget has extra edit parameters compared to the ones described in Widgets common configuration.

- **Widget Options**:
  - **Choose Gimbal**: The user has to choose from the gimbal predefined list the gimbal for which the widget is configured.
  - **Activate controls**: Select the desired controls of the selected gimbal that will be displayed on the gimbal configuration panel.

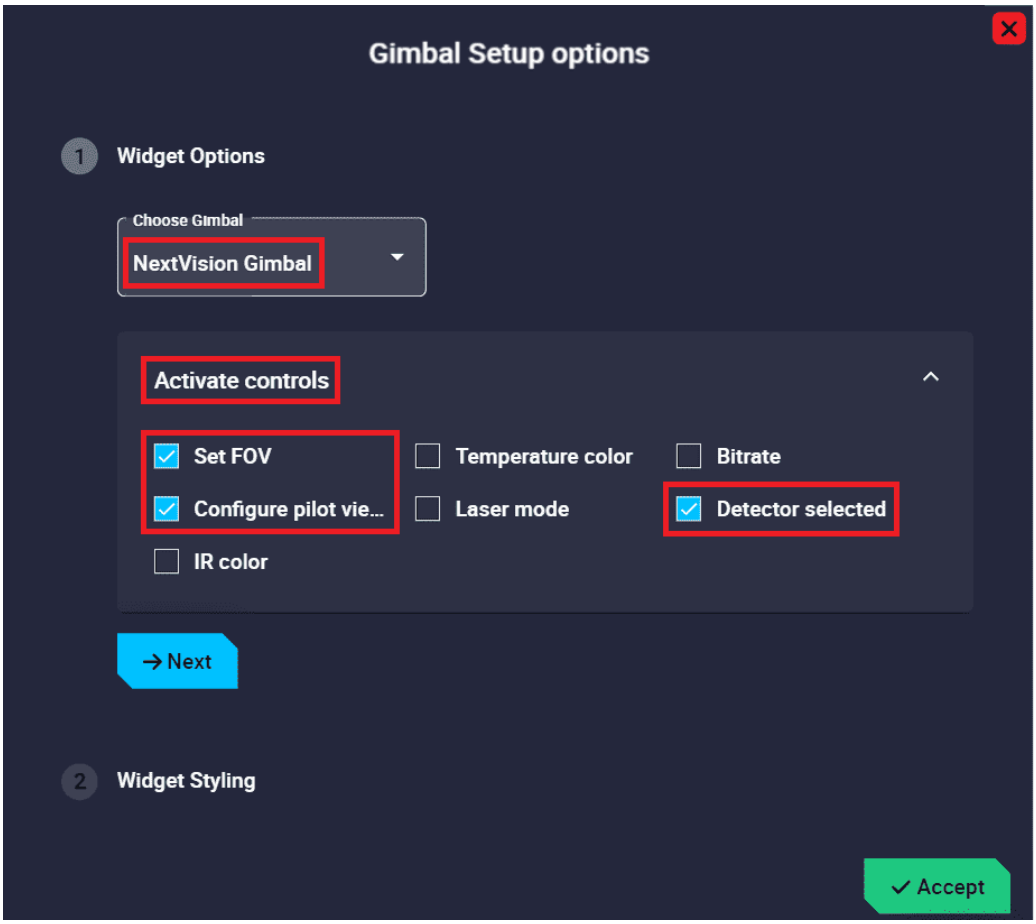
    > **Important**
    > The controls shown here depend on the selected gimbal, each gimbal has its own controls.

Below is an example with the **NextVision gimbal**:



Gimbal Setup example

These controls can be activated/deactivated in the configuration:



Gimbal Setup configuration example

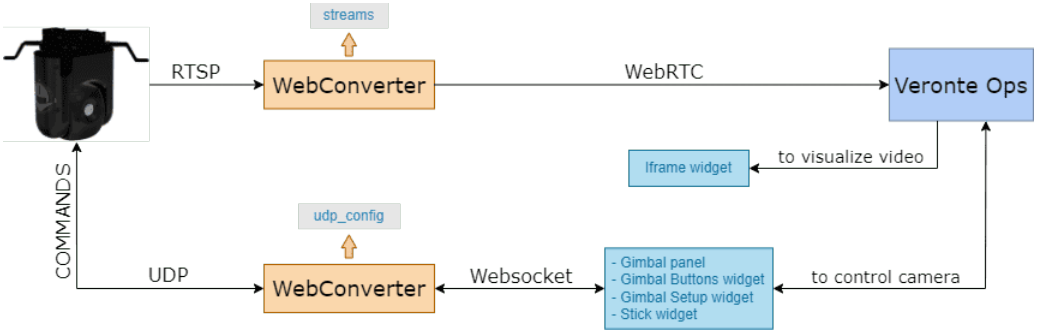
As can be seen in the example above, these controls can be in form of sliders, drop-down menus, etc.

> **Important**
>
> This will also depend on the pre-configured controls for each gimbal model.

Integration examples

Gimbal

The following diagram summarizes the "connection" and "elements" required to view video recording and/or command a **generic gimbal** in **Veronte Ops**:



Gimbal connection diagram

> **Note**
>
> The configuration described in this section covers how to set up and control a **generic gimbal** with **Veronte Ops**.

To configure and control a gimbal from **Veronte Ops**, the following must be considered:

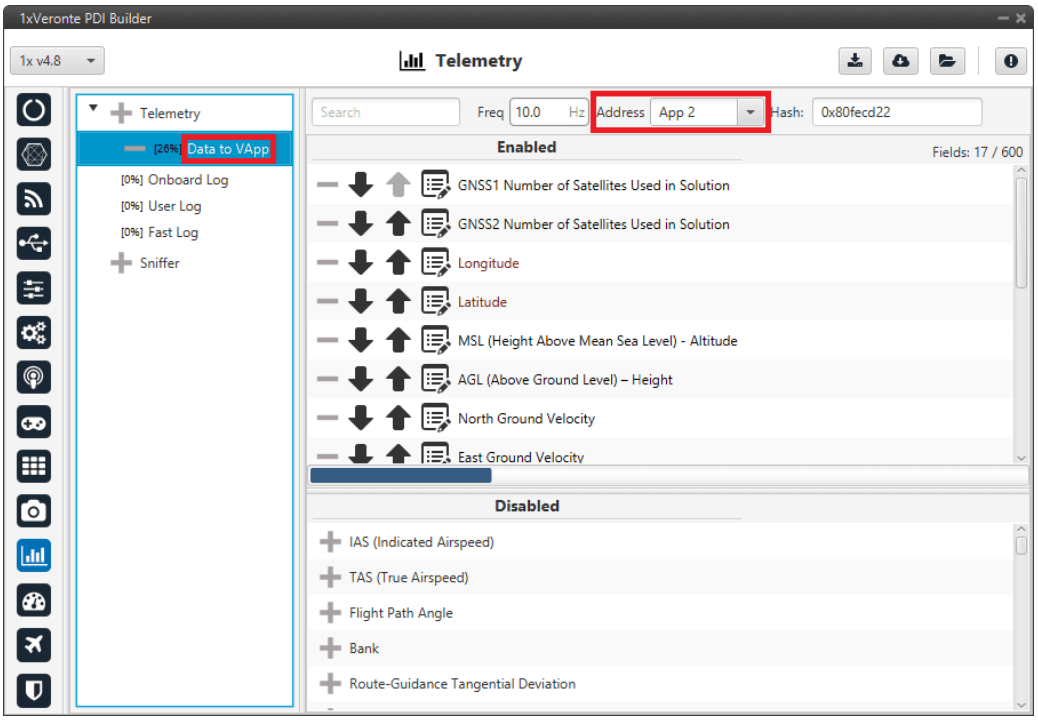
- The gimbal must be configured in the Gimbal panel.
- Use the Gimbal Buttons widget to command actions to the gimbal camera.
- Use the Gimbal Setup widget to configure some options of the gimbal camera video.
- To view the video recording and/or track directly with the displayed image, the Iframe widget is needed.
- Finally, if the user wants to manage the gimbal with a **Veronte Ops** virtual stick or a USB joystick, the Stick widget must also be configured.

Furthermore, it is necessary to configure the following variables in the telemetry vector of the **Autopilot 1x** in order to establish a correct

communication for commanding between **Veronte Autopilot 1x** ⇒ **Veronte Ops** ⇒ **Gimbal**:

- GNSS2 Number of Satellites Used in Solution
- GNSS1 Number of Satellites Used in Solution
- Longitude
- Latitude
- MSL (Height Above Mean Sea Level) - Altitude
- AGL (Above Ground Level) - Height
- North Ground Velocity
- East Ground Velocity
- Down Ground Velocity
- Heading
- GS (Ground Speed)
- p (Angular Velocity - X Body Axis)
- q (Angular Velocity - Y Body Axis)
- r (Angular Velocity - Z Body Axis)
- Yaw
- Pitch
- Roll

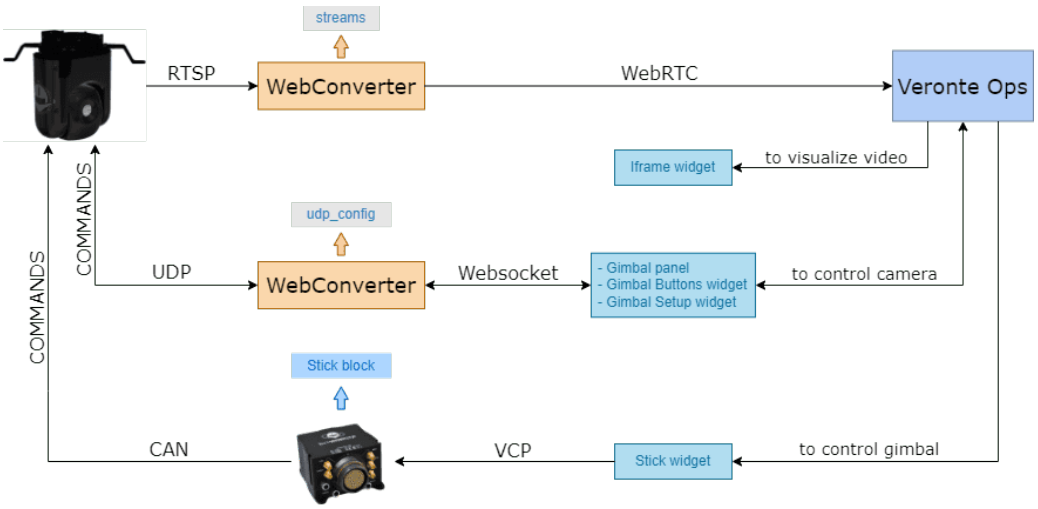
The **Data to Vapp** telemetry vector in the **Autopilot 1x** configuration (using **1x PDI Builder** software) should look like this:



Gimbal - Telemetry configuration

Veronte Gimbal

The following diagram summarizes the "connection" and "elements" required to view video recording and/or command a **Veronte Gimbal** in **Veronte Ops**:

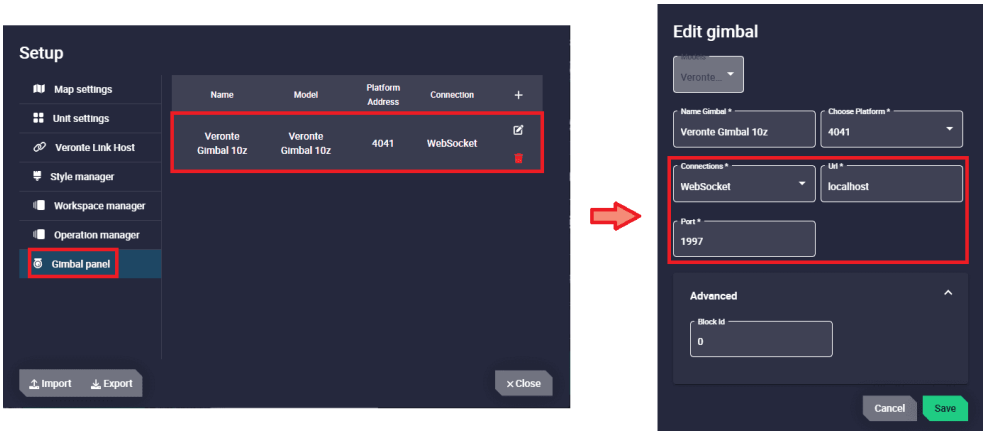


Veronte Gimbal connection diagram

> **Note**
>
> The configuration described in this section covers how to set up and control a **Veronte Gimbal** with **Veronte Ops**.

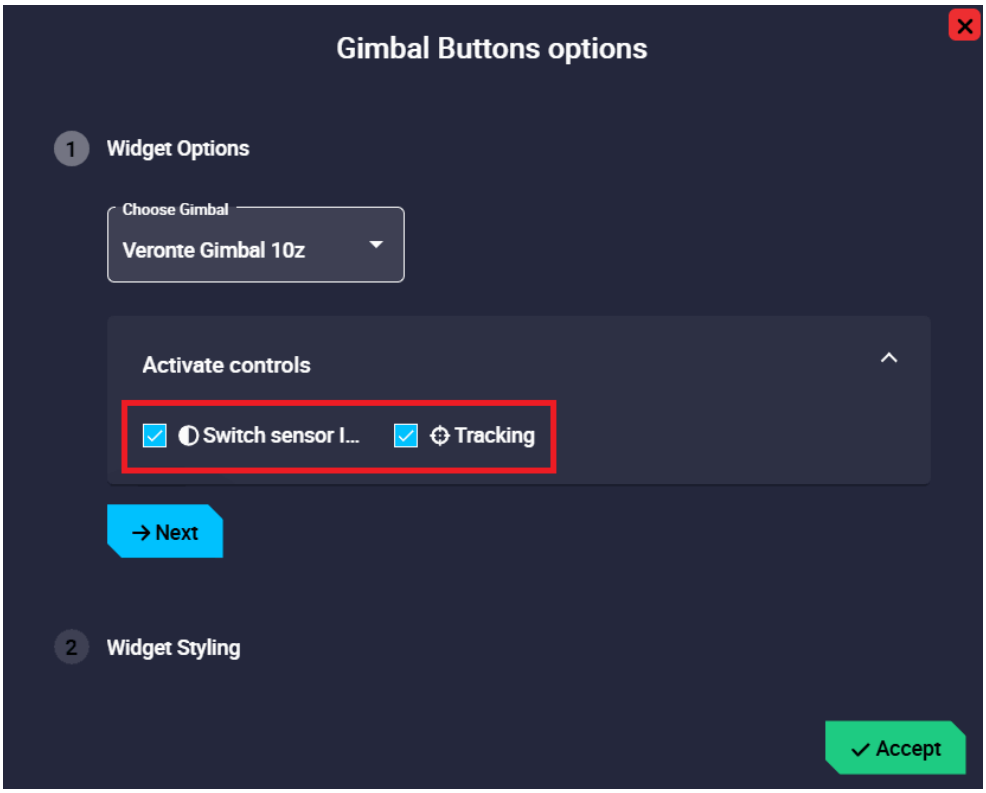
To configure and control a **Veronte Gimbal** from **Veronte Ops**, the following must be considered:

- The gimbal must be configured in the Gimbal panel. If the user has a Veronte Gimbal 10z, the configuration should look like this:



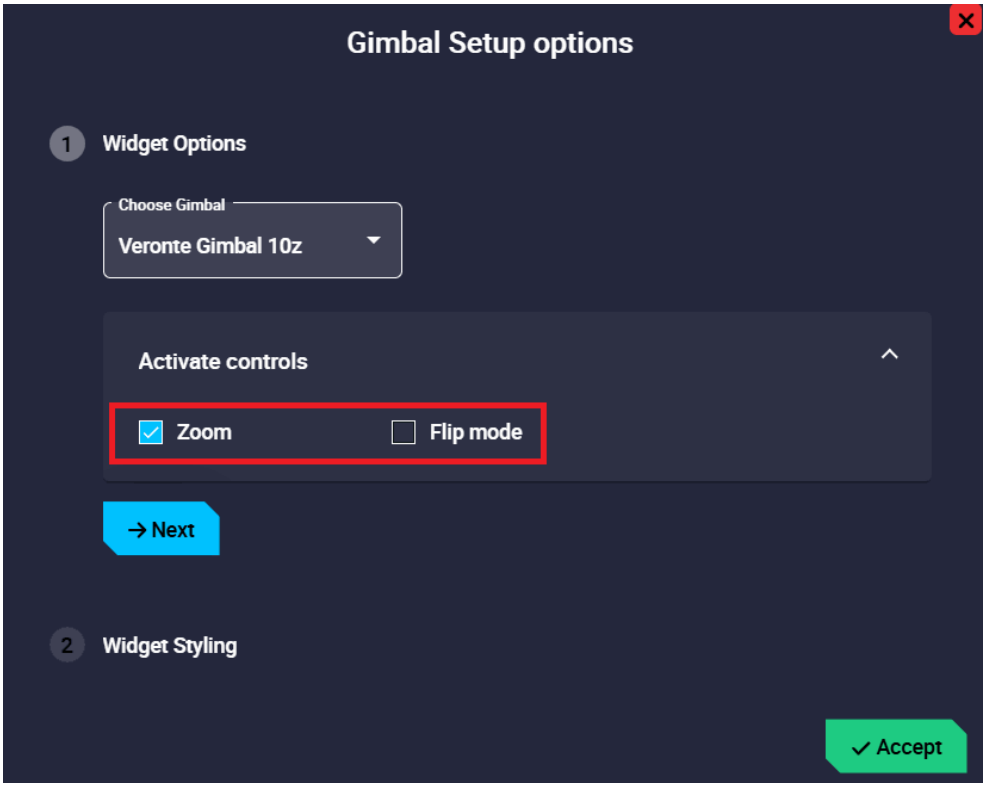
Veronte Gimbal configuration - Gimbal panel

  - **Model**: Veronte Gimbal 10z
  - **Connections**: Websocket
  - **Url**: `websocket_url` configured in the Web Converter app.
  - **Port**: `websocket_port` configured in the Web Converter app.
- Use the Gimbal Buttons widget to command actions to the gimbal camera. For the Veronte Gimbal 10z, the following actions are available:



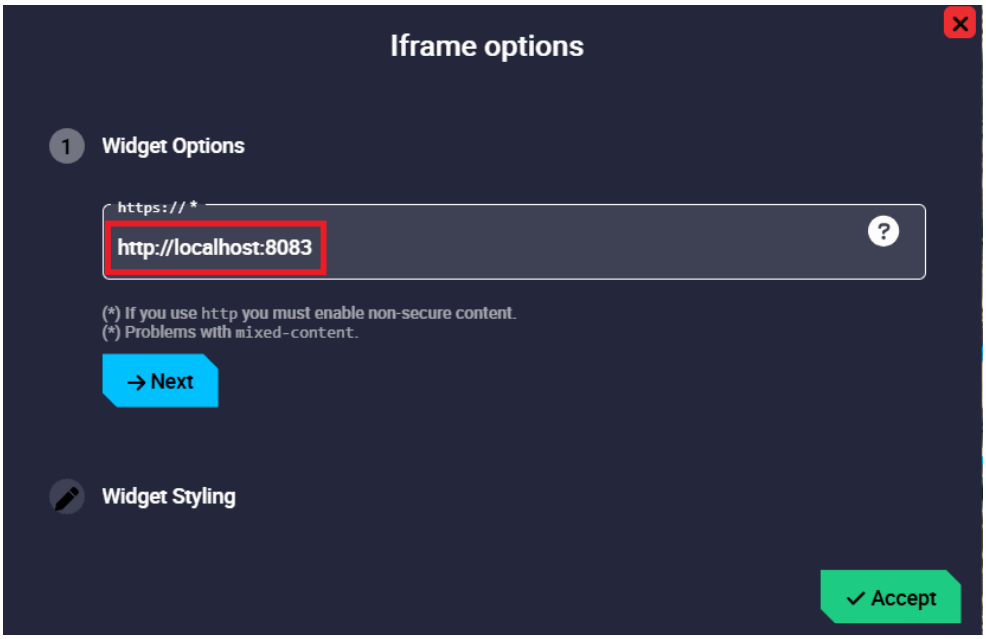
Veronte Gimbal configuration - Gimbal Buttons widget

  - **Switch sensor IR/Day**: Button to switch between IR and EO camera.
  - **Tracking**: Button to stop tracking.
- Use the Gimbal Setup widget to configure some options of the gimbal camera video. For the Veronte Gimbal 10z, the following controls are available:



Veronte Gimbal configuration - Gimbal Setup widget

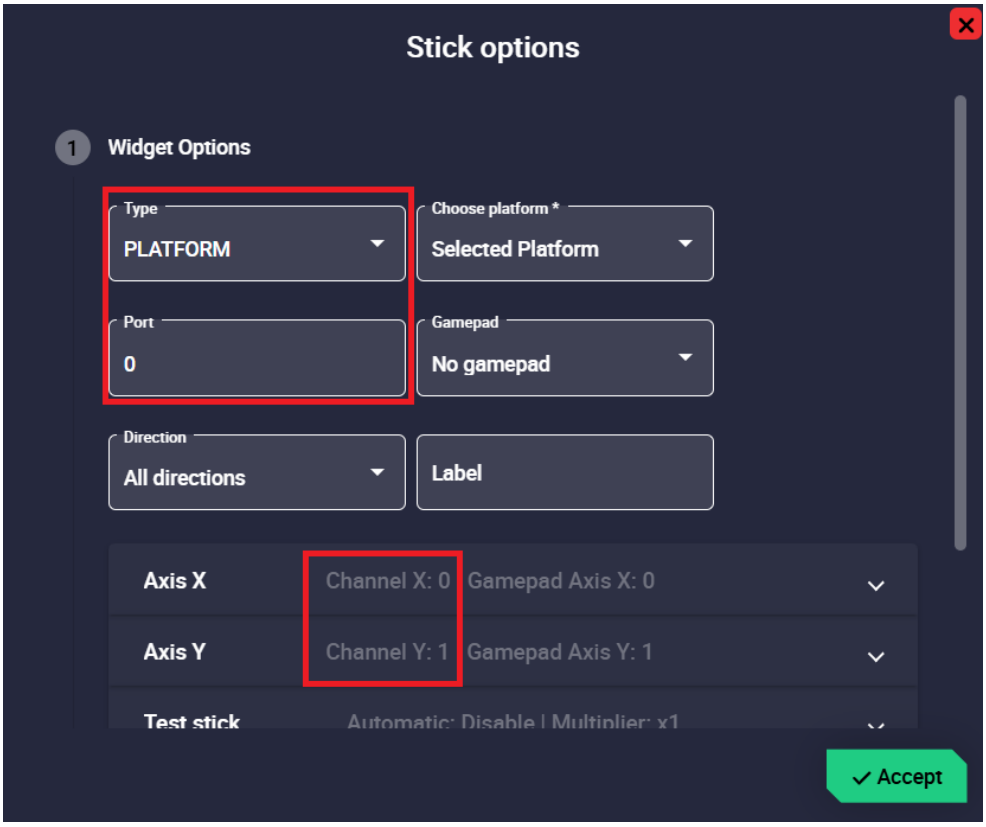
  - **Zoom**: Users can zoom from 0% to 100% with a slider.
  - **Flip mode**: Users can flip the camera as desired. The available options are Normal, Mirror, Flip and Mirror and flip.
- To visualize the video recording and/or track directly with the displayed image, the Iframe widget is needed.



Veronte Gimbal configuration - Iframe widget

- **https://**: Enter the URL generated by the Web Converter app.

- Finally, to control the gimbal with a **Veronte Ops** virtual stick or a USB joystick, the Stick widget must also be configured as shown in the figure below:



Veronte Gimbal configuration - Stick widget

> **Note**
> In this case, it is controlled directly with the virtual stick (no USB joystick is being used).

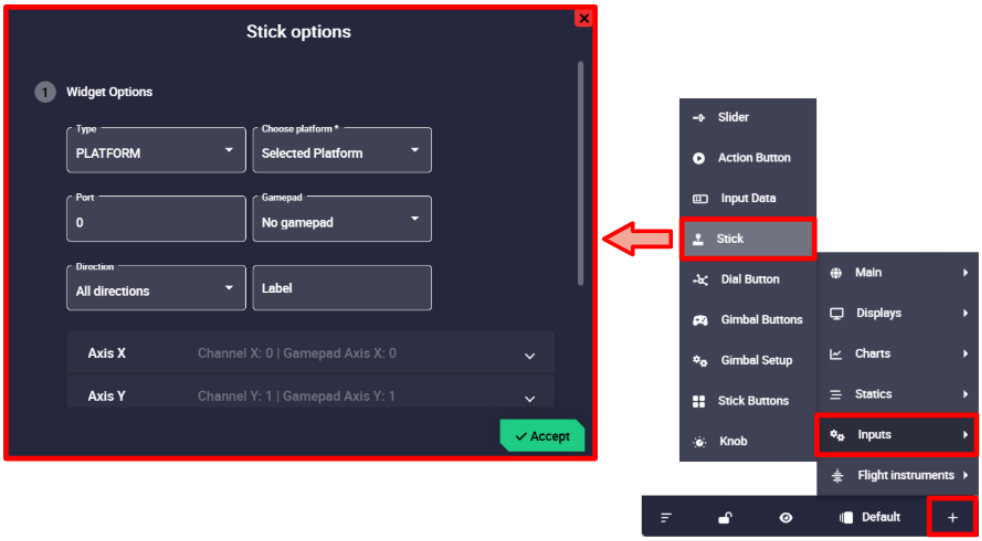
- **Type**: PLATFORM must be selected.
- **Port**: Select the port configured in the **Stick block** in the **1x PDI Builder** software.
- **Axis X/Y**: Configure both channels X and Y.

Stick widget

Once the steps described in the Stick widget - Integration examples section of the **1x PDI Builder** manual have been performed, configure **Veronte Ops** to enable the commands of the Stick widget. To do so:

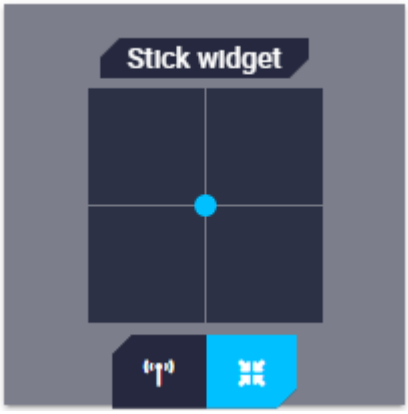
1. Add a **Stick widget**.

> **Note**
> The connection should work with the default settings. For more information on the configuration of the **Stick widget**, please refer to Stick - Workspace section of this manual.



Stick widget - Stick configuration

2. Enable the send of commands to Veronte Autopilot by clicking on the **Send command** button of the widget.



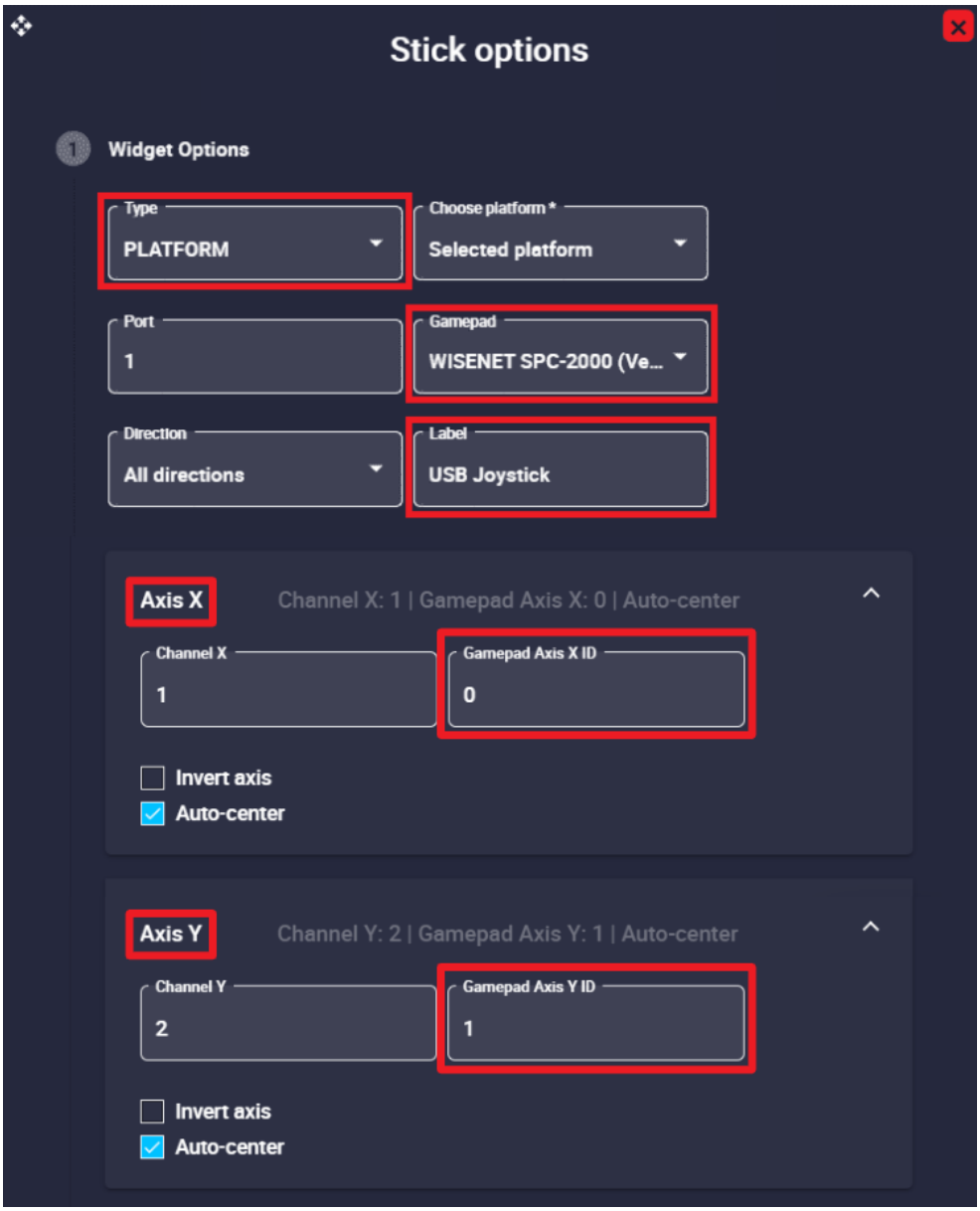
Stick widget - Enable send commands

USB joystick

Veronte Ops is able **to detect USB devices such as joysticks**. The buttons and axes of these devices can be read and configured to send stick information to **Veronte Autopilot 1x**.

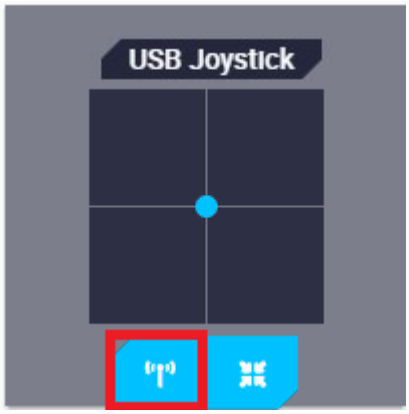
Once the steps described in the USB joystick - Integration examples section of **1x PDI Builder** manual have been performed, configure a **Stick widget** to be connected to a USB joystick. To do so, follow the steps below:

1. Select the **Gamepad** that corresponds with the USB joystick connected to the PC.
2. And configure the **Gamepad Axis X/Y ID** so that the axes IDs of the stick widget gamepad correspond to those of the physical joystick.



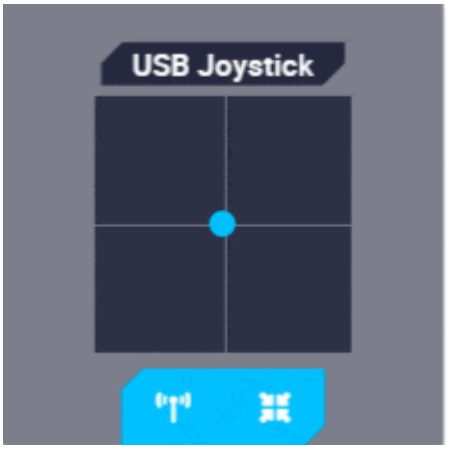
Stick widget - USB joystick

3. Enable the send of commands to Veronte Autopilot by clicking on the **Send command** button of the widget.



Stick widget - Enable send commands

Below is an example of how the stick widget works with the previous configuration:

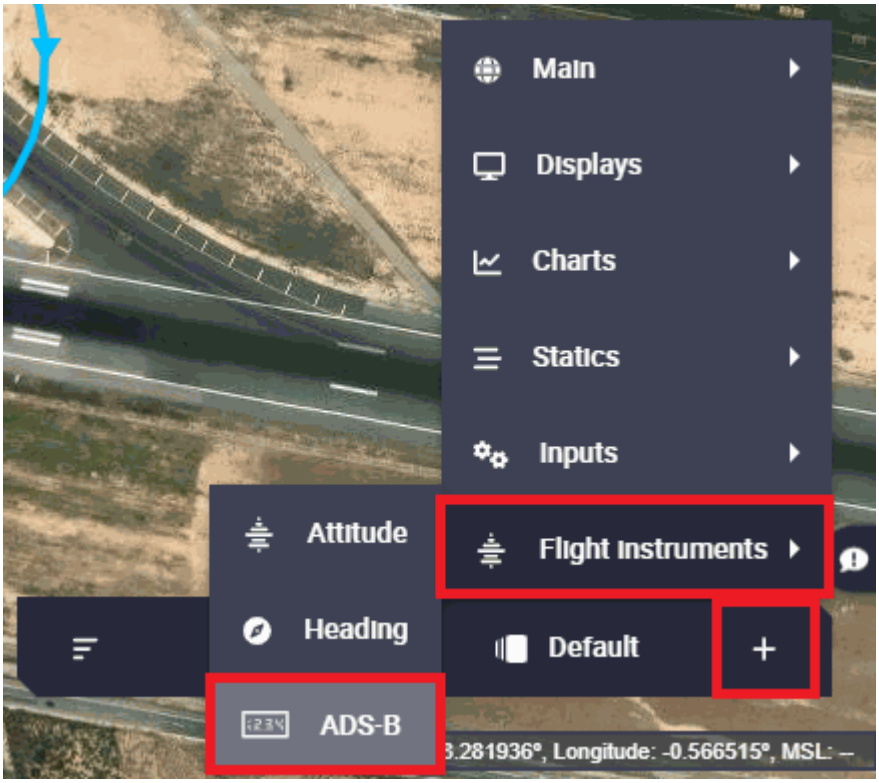


Stick widget - USB joystick movement

Remote ID

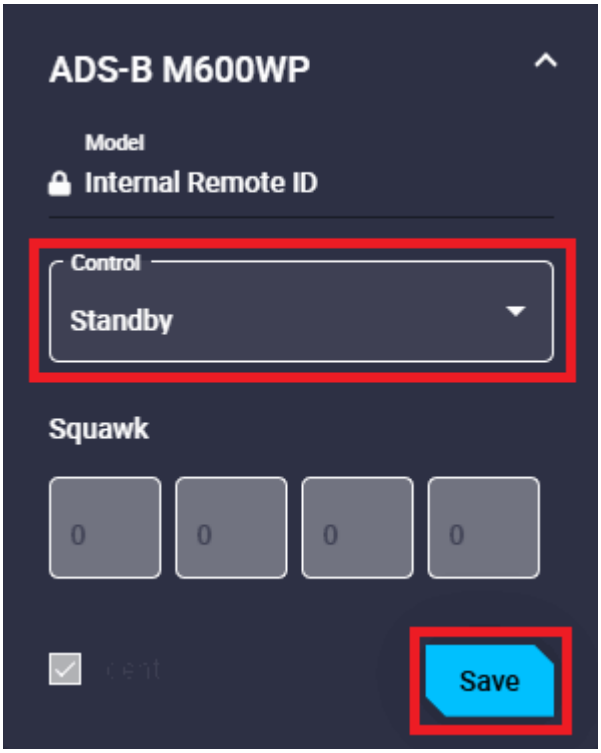
To configure the Remote ID, it is necessary to verify that it has been configured in 1x PDI Builder.

The control mode can be changed within Veronte Ops, so it is not necessary to reconfigure the PDI files.
To do this, open **Veronte Ops** → in the **Workspace toolbar**, click + → **Flight instruments** → **ADS-B**. An ADS-B widget will be added to the workspace:



Remote ID - Configuration

This widget displays the **Internal Remote ID** as the **Model** (the model previously selected in 1x PDI Builder).
The Control mode can be modified; press **Save** to upload the changes.

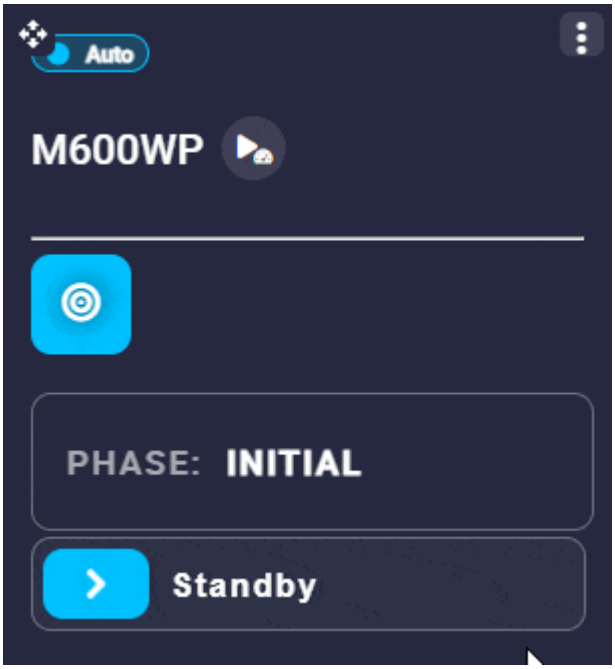


Remote ID - Widgets

Troubleshooting

Error when changing phase

When changing phase, an **error** may occur so that the phase change does not take place and also the current phase identifier is **colored red** (the phase that appears is the one the platform was in before the phase change was attempted):



Veronte Panel - Changing phase error

The reason for this error depends on the user's goal:

- **During simulation or simple test**
  - If this error occurs while the user is **running simulations or simple tests without all sensors connected**, it is most often because **Veronte Autopilot 1x** has not been configured in **PDI Mode**. If it has been configured in **PDI Mode**, a pill indicating this appears directly on the **Veronte Panel**.
    For more information on PDI Mode, visit the PDI Mode description of the **1x PDI Builder** user manual.
  - It may also be due to **navigation not initialized**. For more information, see Navigation does not start section of this page.
- **During flight test or operation**

If this happens when the user is **not in PDI mode** (e.g. because the user is operating with the configuration finished and all sensors connected), it may be due to one of the following reasons:

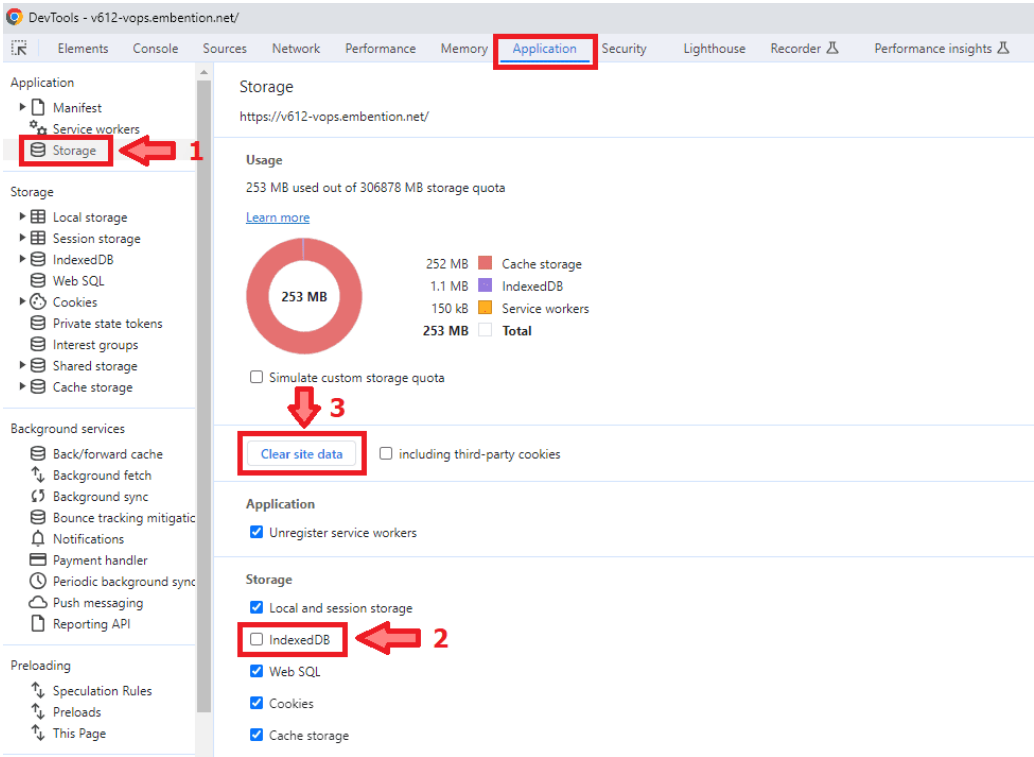
- **System bit** is in **error mode** (not OK).

  A list of all errors that can cause this bit to be set can be found in the Activation System Error bits - Lists of interest section of the **1x Software manual**.
- **Navigation attitude could not be initialized**. For more information, see Navigation does not start section of this page.

Installed Veronte Ops app does not load

When users open the installed Veronte Ops application and it does not load or takes a long time to load, follow the steps below to fix it in:

1. Press the `F12` key on the keyboard to open the **DevTools** window.
2. Go to **Application** tab ⇒ **Storage**.
3. To avoid losing **workspaces** and **operations** that are already loaded in the application, the **IndexedDB** checkbox must be **unchecked**.
4. Finally click on `Clear site data`



DevTools window

Navigation does not start

To start the navigation, the following requirements must be met:

- Correct **IMU measurements**.
- The **yaw** must be initialized with magnetometer measurements or with an **yaw initialization command**.

  To do the latter, see Calibrate Yaw - Operation section of this manual.

Sounds Notification Language

The app will only support the languages found in the user's system settings.

Appendix

Script Widget

This widget enables the definition and execution of **Flight Operational Procedures (FOPs)** through customizable JavaScript-based scripts. It provides a visual terminal, parameter inputs, confirmation dialogs, and interaction with onboard telemetry and telecommands.

The script widget allows:

- Upload and execute scripts.
- Send telecommands (TC) to the platform.
- Read telemetry from the platform.
- Show interactive dialogs for operator input or confirmation.
- Add logic (IF, WHILE, FOR) to control the script flow.
- Wait between actions.
- Check ACKs after sending commands.
- Evaluate telemetry conditions to continue or abort.
- Show real-time progress via the terminal.
- Apply transformations (scaling, offset, polynomial) to command data.
- Abort execution.
- Read a `.json` file and transform it in an object available in the script.
- Upload custom files, called **Script Libraries**, with functions or constants to be available globally in the scripts.
- Syntax errors will be shown when the editing area is unfocused, or in the terminal when executing an incorrect script.

Scripts are written in **JavaScript**, using an asynchronous context, and executed in the widget sandbox. Therefore, for instance, write comments with `//`, and all the well-known syntax of this language.

Getting Started

1. Open the **widget script configuration** panel.
2. Use the **file picker** to upload a file or directly write in the code text area.
3. Edit or review the script in the editor.

4. Click **Save** button.
5. Click **Run** to start execution.
6. The script will execute from top to bottom asynchronously.
7. A **spinner** indicates that the script is running.
8. There is the option to **stop** the script execution.

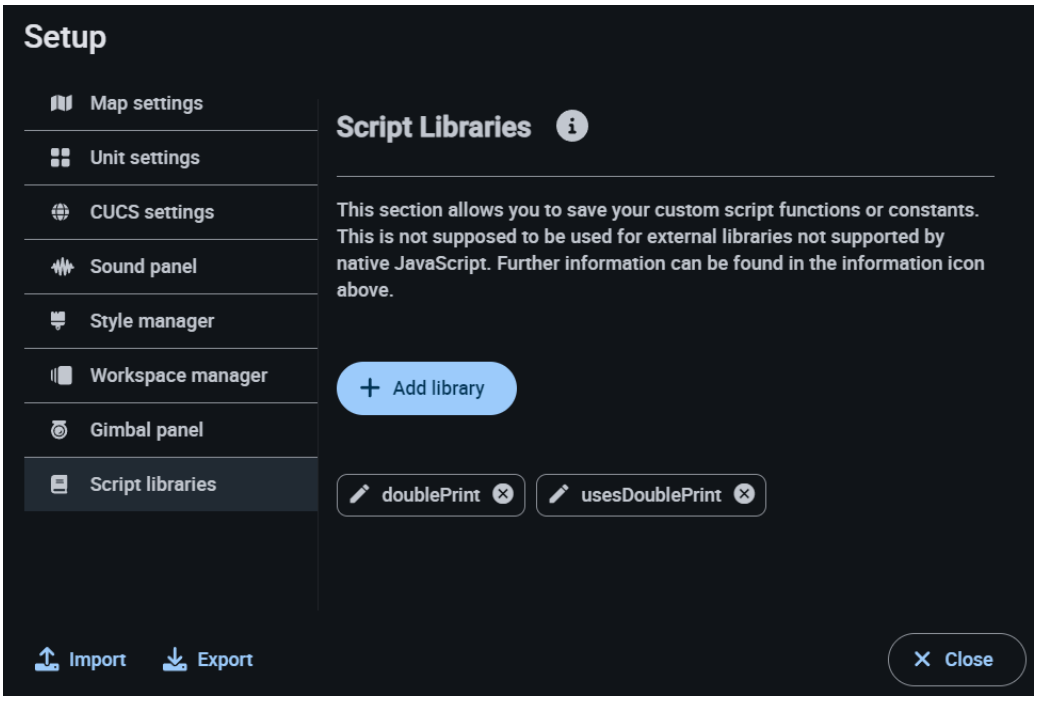
Importing custom libraries

It is possible to import custom libraries. To achieve this, navigate to **Setup → Script Libraries**. The export / import must not be used. Just declare the functions with a regular declaration (keywork function) and the variables with const/let. They will be available in all the scripts. This is not supposed to be used for external libraries not supported by native JavaScript.

The data from the libraries would be available in the scripts at the start of the execution. This means that, if there is currently a script running, the libraries will not be available until the next execution.

Libraries are evaluated in chronological order, meaning that is required to use a function or variable from another library, it must be imported before the one that uses it.

This is an example of the dialog of configuration (it is possible to rename the modules):



Script - Configuration

Stopping a Script

The execution can be stopped:

- Automatically from within the script (adding a `return;` clause). This is a graceful end.
- Force the script to abort at any point using `throwAbort()`.
- With the **stop** button. This is also a forced termination.

Available Scripting Functions

The script runs in a safe sandbox environment and offers the following helper functions available:

Function	Description
`getVar(typeId: number, id: number): Promise<number>`	Reads a telemetry variable (with Type ID*, and ID** indicated) from the UAV. Returns the value, or **-1** if not available
`commandVars(varList: SystemVarData[], values: number[]): Promise<void>`	Sends telecommands (TCs) with the given values. If the promise is successfully resolved, it means that the command has been properly received. If an error occurs, an Error will be thrown or the promise will be rejected.
`openPopUpInput(title: string, subtitle: string[], inputData: Array<InputData>): Promise<Array<number | null>>`	Opens a dialog to request user input (numbers or select) with validations.
	Opens a confirmation dialog for user confirmation.

Function	Description
`confirmationDialog(title: string, messages: string[]): Promise<boolean>`	
`openJsonInput(title: string): Promise<object>`	Opens a dialog to select a `.json` file, and returns the object contained in that file. If the format is invalid, the returned object would be null and the error would be shown.
`println(message: string, type?: TerminalMsgType): void`	Logs the message in the terminal panel.
`log(message: string, type?: LogType): void`	Logs the message in the log system file.
`sleep(milliseconds: number): Promise<void>`	Pauses the script for the given time.
`clear(): void`	Clears the terminal output.
`throwAbort(): void`	Forces the finish of the script.

SystemVarData interface:

```
interface SystemVarData {
    id: number;
    typeId: number;
}
```

InputData interface:

```typescript
interface SelectOption {
  label: string;
  value: number;
}

type InputType = 'number' | 'select';

interface InputData {
  title: string;
  type?: InputType; // defaults to 'number'
  defaultValue?: number;
  min: number | null; // only needed if type === 'number'
  max: number | null;
  options?: SelectOption[]; // compulsory if type === 'select'
}
```

TerminalMsgType enum:

Type	Value expected (strings)
Default	No value needed
Error	`error`
Warning	`warning`
Success	`success`

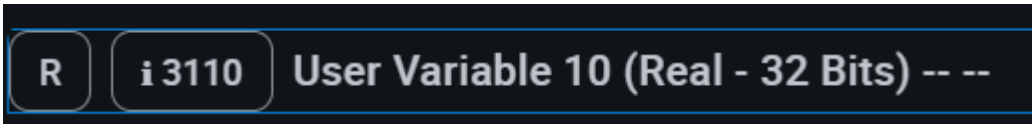
LogType enum:

Type	Value expected (strings)
Default	No value needed.
Error	`error`
Warning	`warning`

Variable types (typeId):

Type	ID
`Real`	0
`Integer`	2
`Bit`	3

The `ID` of the variable can be found in the dialog of variable selection in the application. Example:



Variable ID

The `ID` in this case would be **3110** (and type `Real`, then `typeId` is **0**).

Example Use Cases

1. 🛰 **Get the Current Phase**

To read the current phase:

```
const phase = await getVar(2, 1); // Type ID 2 = Integer, ID 1 = Phase
println("Current phase: " + phase);
```

1. 📥 **Get Input from Operator (Popup Form)**

To request user-entered values (two first are input number type, and the third one is a select list):

```
const result = await openPopUpInput("Enter Mission Parameters", [
    "Mode options:",
    "Passive > 0",
    "Position > 1"
  ],[
  {
    title: "Speed (m/s)",
    defaultValue: 10,
    min: 0,
    max: 100
  },
  {
    title: "Altitude (m)",
    defaultValue: 30,
    min: 10,
    max: 300
  },
  {
    title: 'Mode',
    type: 'select',
    defaultValue: 1,
    options: [
      { label: "Passive", value: 0 },
      { label: "Position", value: 1 }
    ],
  }
]);

if (result) {
  println("Speed: " + result[0] + ", Altitude: " + result[1] + ", Mode: " +
result[2]);
} else {
  println("No input provided.");
}
```

> ⓘ **Note**
>
> All fields are validated. The user cannot continue if values are out of range. If the limits are not provided, then no validation would take effect. If the default value is not changed, then that value would be sent.

1. 📋 **Print to the Terminal**

To show status/progress updates:

```
println("Step 1: Initialization ✅");
println("Waiting for confirmation...");
```

1. 🧹 **Clear the Terminal**

```
clear();
```

Removes all previous logs.

1. ⏳ **Add a Pause**

To wait two seconds before continuing:

```
await sleep(2_000);
```

The number is in milliseconds.

1. ✅ **Confirmation Dialog**

To confirm with the operator before continuing:

```
const confirmed = await confirmationDialog("Continue?", ["This will start the
mission."]);
if (!confirmed) {
  println("Aborted by operator ❌");
  return;
}
```

1. 📡 **Send Telecommands (TC)**

To command variables to the UAV:

```
await commandVars([
  { typeId: 0, id: 3110 },
  { typeId: 0, id: 3111 }
], [15, 50]); // Values to send
```

Multiple variables can be sent in one call.

1. ✅ **Verify ACK**

To confirm a telecommand was acknowledged correctly:

```
try {
  await commandVars([{ typeId: 0, id: 3110 }], [15]);
  println("ACK OK ✅");
} catch (e) {
  await println(e);
}
```

Retries may also be performed inside a loop.

```
for (let i = 0; i < 3; i++) {
  try {
    await commandVars([{ typeId: 0, id: 3110 }], [15]);
    await println("Sent successfully!");
    break;
  } catch (err) {
    await println("Retry " + i + ' ' + err);
    await sleep(500);
  }
}
```

1. 📊 Check Specific Telemetry

Telemetry conditions can be verified:

```
const battery = await getVar(0, 2002);
if (battery < 11) {
  println("Battery too low (" + battery + "V) ❌");
  return;
}
```

Useful for safety conditions.

1. 🔁 Conditional Execution (IF / WHILE / FOR)

Standard JavaScript control structures are supported:

```
if (await getVar(0, 1000) > 50) {
  println("Condition met, sending TC");
  await commandVars([...], [...]);
}

while (await getVar(0, 3110) !== 15) {
  println("Waiting for ACK...");
  await sleep(500);
}
```

1. ➕ Transform Values Before Sending (Transfer Functions)

Values can be modified before sending:

```
const rawSpeed = 10;
const scaledSpeed = rawSpeed * 1.5 + 2;
await commandVars([{ typeId: 0, id: 3110 }], [scaledSpeed]);
```

Apply any function, including polynomials:

```
function polynomialTransform(x) {
  return 0.01 * x ** 2 + 3 * x + 7;
}
await commandVars([{ typeId: 0, id: 3110 }], [polynomialTransform(20)]);
```

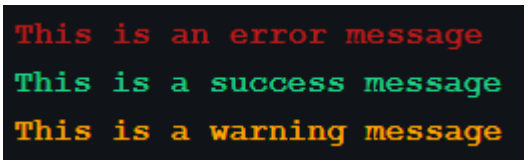
1. 📌 Checklist and Step Progress

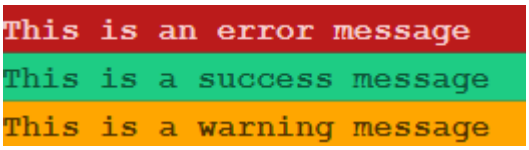
Use `println()` to simulate checklist status:

```
println("✅ Step 1: Pre-flight check passed");
println("✅ Step 2: Command sent");
println("❌ Step 3: ACK not received");
```

1. 🚩 **Colored outputs**

Use `println('This is an error message', 'error')` and the other types to see a different color in the output console:





Colored outputs

> ⓘ **Note**
>
> The type of colored output depends on the widget configuration (color mode).

1. 🎨 **Create custom functions**

A comfortable way of working with this widget is to create custom functions. An example of this:

```
async function setAltitude(value) {
  return await commandVars([{ typeId: 0, id: 3110 }], [value]);
}
```

The only thing that needs to be done is:

```
  await setAltitude(55);
```

And this will set the user variable `10` to `55`. If it is necessary to go a bit further, this values can be stated as constants at the start of the script:

```
const set_altitude_cmd = {
  id: 3110,
  typeId: 0
}

async function setAltitude(value) {
  await commandVars([set_altitude_cmd], [value]);
}
```

1. 🔨 **Upload a JSON and get an object**

Use `openJsonInput('Select the JSON with the altitude variables')` to read a JSON file and get the corresponding object. This file:

```
{
  "msl": 50,
  "wgs84": 70
}
```

JSON Upload & Validation

Can be read with this code:

```
const json = await openJsonInput('Select the JSON with the altitude variables');
const msl = json?.msl;
if (typeof msl !== 'number') {
  println('Missing position.msl in JSON ❌', 'warning');
  return;
}
println('Altitude (MSL): ' + msl);
```

We strongly recommend validating the structure of the JSON object (optional chaining and/or typeof).

1. **Using libraries**

After having imported the needed custom libraries, all the functions should be available in the script. No import should be used (as it should not be used any export).

Assuming a library defines the following code:

```
function doublePrint(value) {
  println(value);
  println(value);
}
```

The script could look like this:

```
const x = 2;
const y = 5;
const result = x * y;
doublePrint('The result is ' + result);
```

The terminal should show the following output:

```
The result is 10
The result is 10
```

1. **Creating enumerations**

The way of creating enumerations we strongly recommend is the following (a native feature of JavaScript):

```
const Position = Object.freeze({
  TOP: 0,
  RIGHT: 1,
  BOTTOM: 2,
  LEFT: 3
})
```

Using it is as easy as entering Position.RIGHT, which returns the value 3. The same can be done with strings:

```
const State = Object.freeze({
  ON: 'on',
  OFF: 'off'
})
```

In this case, the return value would be a string. Although, the use of Object.freeze is optional, its use is encouraged as it guarantees that the value of the enumeration does not change and that no new values are added, which is the expected behavior of an Enumeration.

Real use example:

```
const Vars = Object.freeze({
  PHASE: 1
})

const phase = await getVar(2, Vars.PHASE);
println('Current phase is: ' + phase);
```

Send a telecommand with custom enumeration:

```
// This code could be in a library called Constants.js
const Position = Object.freeze({
  PASSIVE: Object.freeze({
    typeId: 0,
    id: 3110
  }),
  POSITION: Object.freeze({
    typeId: 0,
    id: 3111
  })
})

// This function could be in a library called RadFns.js
async function setRadPos(pos, value) {
  return await commandVars([pos], [value])
}

// Main script
await setRadPos(Position.PASSIVE, 100)
```

1. **Using system logs**

The log function enables direct writing into the system log, allowing for the specification of different severity levels.

One example of how it could be used is as follows:

```
log('Example of log')
log('This case is a waning', 'warning')
log('And an error!!!', 'error')
```

With a result in the file log like this:

```
[2025-09-01 09:55:55.931] [info]  Example of log
[2025-09-01 09:55:55.932] [warn]  This case is a waning
[2025-09-01 09:55:55.933] [error] And an error!!!
```

The format is clear: `[date time] [type] Text message`. Using this as a starting point, a good idea would be to create a custom sub-format. For instance, something like `[script] [operation] status/result/info`. The final result could be something like this:

```
[2025-09-01 10:01:01.101] [info] [script] [setRadPos] Position.ABSOLUTE 100
```

This makes it easier to filter by operation or script logs. This can be achieved with the following code:

```
const Position = Object.freeze({
  ABSOLUTE: 0,
  RELATIVE: 1
})

log(`[script] [setRadPos] Position.ABSOLUTE 100`)

// Or parameterizing it
function logOperation(op, variable, value) {
  return `[script] [${op}] ${variable} ${value}`
}

log(logOperation('setRadPos', 'Position.ABSOLUTE', 100))
```

Error list

The most common errors that could appear while executing the script are:

- `No platform found`. Currently it is not possible to play a script without a platform selected.
- The message `functionName` argument must be `string` / `number` / `array`. Defined types must be respected.
- An error while reading a JSON file because of the format of the uploaded file. Please, do respect the standard. It would say something like `Error reading JSON file: ...`.

- `commandVars` function will be rejected if it could not be completed. For instance, it could throw any of this errors:
  - A timeout because the connection with the platform is lost.
  - Too many fields sent.
  - Trying to write a read-only variable.
  - Any other reason of this kind.
- `Unknown operation` when executing a non supported operation.
- In the event of a syntax error, it will be displayed as `Error`: followed by the reason provided by the language.

Notes and Tips

- The **use of** `**await**` may be required for asynchronous functions: `getVar`, `sleep`, `commandVars`, `openPopUpInput`, `confirmationDialog`. Remember JavaScript is an asynchronous and single-threaded language.
- `println()` shows logs in the terminal. It is the primary tool for guiding the operator.
- To have the possibility of comfortably using the `stop` button, `sleep` functions and dialogs are recommended.

Safety Notes

- Scripts are sandboxed and run asynchronously.
- The functions provided are limited due to safety reasons.
- Be sure to validate user inputs and telemetry ranges.
- Bear in mind a forced stop could left the system on a non-desired state.

📎 Summary Table

Feature	Supported
Load and edit script from file	☑
Run execution	☑
Send telecommands	☑

Feature	Supported
Wait between commands	☑
Operator input/select popup	☑
Confirmation dialogs	☑
ACK verification	☑
Telemetry check	☑ (via `getVar`)
Conditional execution	☑
Checklist progress (via logs)	☑
Stop execution	☑
Read a JSON and get object	☑
Import custom libraries	☑

For any further help writing scripts, refer to the examples above or contact support.

Software Changelog

This section presents the changes between versions of **Veronte Ops** application.

7.8.24

This section presents the changes between the previous software version **v. 7.6.52** and the current **v.7.8.24**.

Added:

- Network interface selector within the Setup CUCS panel
- Show operation variables in operation panel
- Group of marks events

Changed:

- Rename Operation Custom point to Mission Points

Fixed:

- Correct saving of widget input data options