Documentation of the WiMA-Extension

Valentin Platzgummer

September 22, 2019

1 Introduction

WiMA is a abbreviation for **Wi**reless **M**easurement **A**pplication. The WiMA extension is a customization who's purpose is to simplify mobile coverage measurements. The current goal is to provide software which is able to generate flight paths form minimal user input. Future goals could be to develop interfaces such that KPI measurements could be triggered from within QGroundControl. An other goal could be to automatically retrieve, merge, analyze and visualize the measured data.

This document was created to explain the functionality of the WiMA-Extension at one hand, and to encourage the reader to find bugs inside the program. As the extension is still being developed the contents demonstrated inside this document may differ from those ones in the program.

The folder "deploy" in the QGroundControl root directory (can be cloned from Gitlab) contains a AppImage of the program. QGroundControl can be launched by double-clicking the AppImage. Currently only a Linux version is available.

2 Documentation

2.1 Structure of QGroundControl with WiMA-Extension

Figure 1 shows a detail view of the QGC main window. Relevant for this documentation are the flight view, the plan view and the WiMA main window. Their function will now be briefly summarized.

The **plan view window** is used to create flight plans. They can be stored as .plan files or directly be uploaded to a vehicle (either a real one or a simulated one). Flight plans consist of way-points. Each way-point stores, among others, a coordinate (latitude, longitude, altitude) and a command (take off, land, wait for x seconds, etc.). To define a sensible flight plan a sequence of way-points should begin with a take off command and end with a land command. Besides way-points more complex patterns can be defined, which is done by inserting a Survey, a Circular Survey, a Structure Scan or a Corridor Scan. Just try them out! Using the simulator can be very helpful.

The **Flight View Window** comes in necessary as soon as you want a vehicle to get in action. Once the vehicle established a connection (serial, TCP, UDP, etc.), QGroundControl starts to communicate automatically with it, in most cases. If not, check the settings menu of QGroundControl. A red arrow will appear on the map, showing the vehicles position and orientation. On the top indicator strip, telemetry data will be published. At the left edge a tool strip will be activated, which can be used to command the vehicle.

The **WiMA Main W4indow** is used to automatically generate flight paths from minimal user input. The user can define a Measurement Area, a Service Area (for take off, land, supply tasks, etc.) and a Corridor, which connects the previous two areas. Below the WiMA main window will be described in more detail.

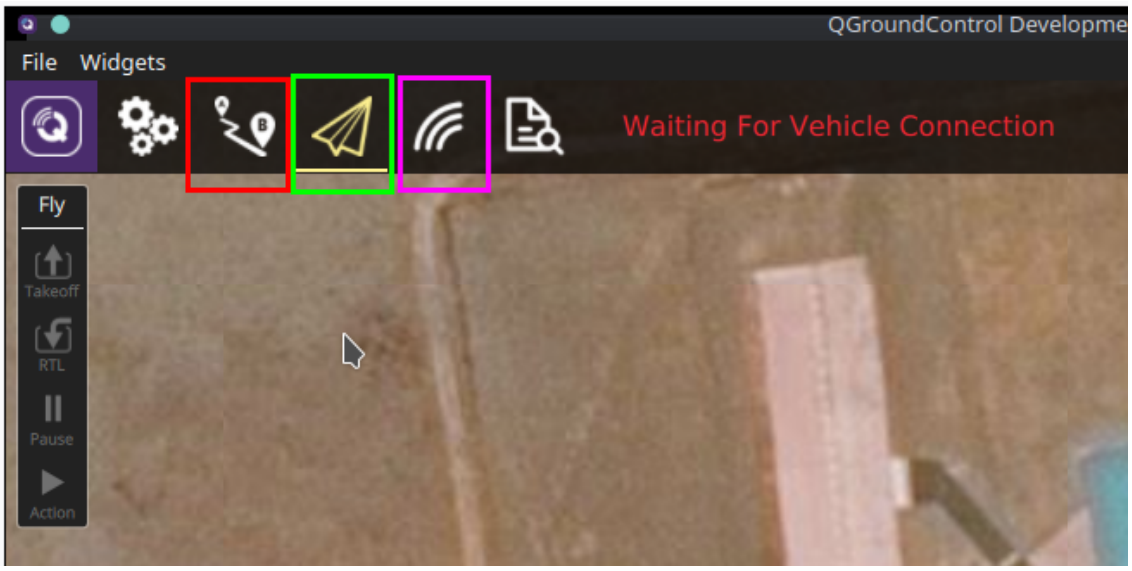

Figure 1: Detail view of the QGC window, which appears after start-up. Marked in red is the button for switching to the Plan View Window, green indicates the Flight View button (current window) and marked in magenta is the button for switching to the WiMA Main Window.

2.2 WiMA Main Window

2.2.1 WiMA Tool Strip

By clicking the wave symbol (see fig. 1; magenta square) the WiMA main window appears. After entering, at the left edge, the WiMA tool strip will appear (depicted in fig. 2).

Pressing the **File** button opens a menu which offers saving and loading operations. At one hand all WiMA areas and mission items can be saved using the `.wima` file extension, at the other hand the mission items only can be stored using the `.plan` file extension. The Open button can be used to load previously stored files. Pressing the New button deletes all contents within the WiMA Main Window. Additionally with the Upload, Download and Clear Vehicle Mission button the mission items (if present) can be uploaded, downloaded or be deleted from the vehicle respectively.

The **Measure**, **Service** and **Corridor** buttons are used to insert a measurement, service and corridor area, respectively. At the time only one of each areas can be inserted. To automatically generate a flight plan at least a measurement and a service area must be defined. Both must be overlapping.

To generate a flight plan the **Calculate** button must be pressed. Further information about how to display the generated flight plan will follow below. In the future this button might be removed and be replaced by a routine which automatically triggers recalculation after any modification. However as the flight plan generation can be time consuming on older devices, this button will remain in the near future, up to the point, underlying routines get optimized.

As flight generation is manually triggered the flight plans might look wrong (for e.g. after user modification), pressing the **Calculate** button often will remove errors. If not, please report any

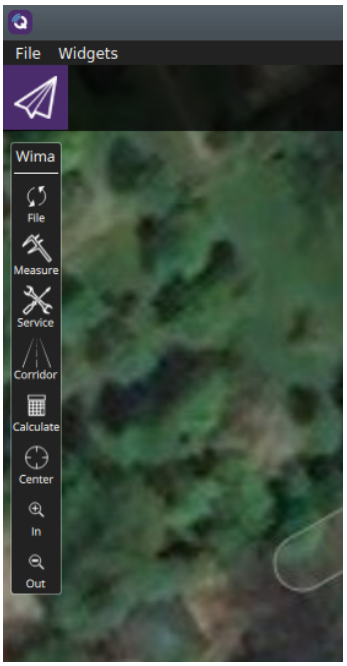

Figure 2: WiMA Toolstrip inside WiMA Main Window.

bugs.

The last three buttons, **Center**, **In** and **Out** are used to center the view and for zooming respectively.

2.2.2 WiMA Areas

As all ready mentioned the three areas, namely the Measurement Area, the Service Area and the Corridor can be defined using the corresponding buttons of the WiMA Tool Strip (see 2.2.1).

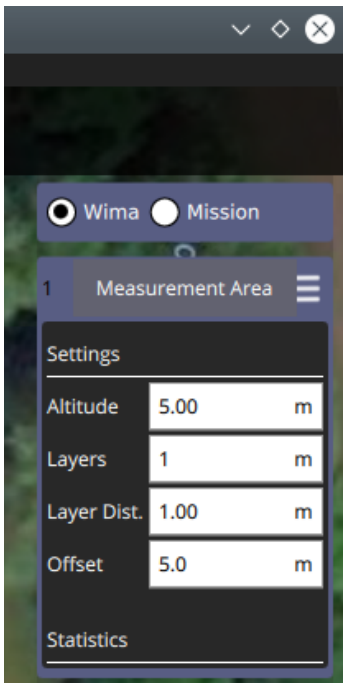

Figure 3: WiMA Item Editor at the right edge of the WiMA Main Window.

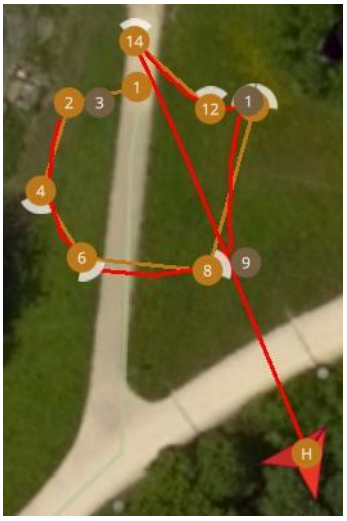

Figure 4: Flight path of a real vehicle (red line).

The **WiMA Measurement Area** defines, as the name indicates, the area of interest within which any measurements should be performed. The area will be displayed as a green shaded rectangle surrounded by a white line, after pressing the Measure button. The area will be listed at the right side within the WiMA Item Editor, after creation (see fig. 3). The WiMA Item Editor can be used to modify area parameters.

The Offset parameter (see fig. 3) changes the distance between the measurement area and its

Figure 5: Adjust the vertex by dragging it's handle.

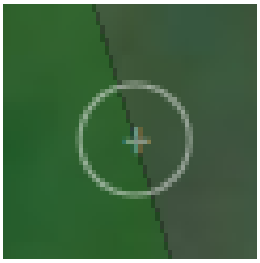

Figure 6: Add vertices by hitting the plus sign.

Figure 7: Move any area by dragging its drag handle.

Figure 8: A valid configuration of Measurement, Service Area and Corridor.

surrounding polygon. The surrounding polygon is a helper to provide sufficient clearance between the measurement area and surrounding obstacles like trees or buildings.

All other parameters shown in fig. 3 don't yet have any impact (intended for future use).

The **WiMA Service Area** is meant to be the area were takeoff and landing happens as well as battery exchange or (hopefully no) repair work is done, hence the name.

The **WiMA Corridor** connects the two previous areas and defines a corridor which the vehicles uses to travel between Service and Measurement Area.

The flight plan will be generated such that all way-points are within or at least at the edge of the above mentioned areas. However it should be taken in account, that the vehicle could still leave the save area, even under fully functional operation. Depending on software implementation of the vehicles flight stack (firmware) the flight controller could decide to perform any kind of path optimization. Additionally drifts caused by wind gusts could occur. An example can be seen in fig. 4. The vehicles flight controller has no information about any WiMA Areas, they exist exclusively within the QGroundControl application.

All WiMA areas can be shaped by dragging the vertex handles (see fig. 5). New vertices can be created by hitting the plus signs at the edges (see fig. 6). The whole area can be moved by dragging its drag handle (see fig. 7).

A valid configuration including Measurement, Service Area and Corridor could look like in figure 8.

2.2.3 The generated Flight Path

The flight plan can be displayed by checking the mission radio button, as depicted in fig 9. An example of a automatically generated flight path can be seen in figure 10. Within the green

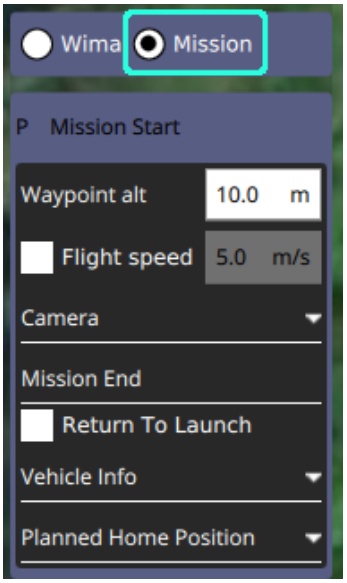

Figure 9: Checking the Mission radio button displays the mission items.

Figure 10: The flight path generated from the areas depicted in fig. 8.

Figure 11: An example of a Circular Survey pattern.

measurement area the program has generated a Circular Survey (see 2.2.4 for further information). The survey has an entry and an exit point. The program has routed paths from the take off point to the surveys entry point and from the surveys exit point to the landing point. The paths have been chosen to be within the areas and as short as possible.

2.2.4 Circular Survey

The Circular Survey (see fig. 11) is a flight pattern provided by the WiMA extension. It is similar to the Survey pattern of the standard QGroundControl application. The Circular Survey was adapted for mobile coverage measurements. It consists of circle segments refereed to as transects. Ideally, the transects are connected with each other such that the path between entry and exit point is as short as possible. Currently this goal is approximated by a heuristics. The solution might not always be the global optimum.

The **Reference** point (Circle with R symbol and Reference tag, see fig. 11) defines the center of the circle segments. Normally it well be placed on the base stations position.

The Circular Survey editor (on the left side of the window) offers some options. The **Altitude** input box adjusts the altitude of the survey. The distance between adjacent circle segments can be modified by changing the Delta R value. The circle segments are approximated by polygonal structures, consisting of way-points. Two arbitrary adjacent way-points of a circle segment have a constant angle between the. This angle can be adjusted by editing the **Delta Alpha** input box. The Delta Alpha value can assume everything between 0.3° and 90°. The higher the value the rougher the circle appears. The survey will contain the less way-points the higher the Delta R and the Delta Alpha values are. The number of way-points impacts how fast the computer can do recalculations and how long the upload to the UAV takes. Sometimes transects are short (length of less than e.g. 3 m). Such ones are often undesired, because the real vehicle will not follow them correctly. To remove them in advance, the minimal transect length can be adjusted by editing the **Min. Length** value. Transects with a length smaller than Min. Length will removed. The **Rotate Entry Point** button and the **Relative altitude** check box don't yet have a function.

2.3 ArduPilot Simulator

For tasks like debugging, program verification or flight plan testing a simulated vehicle can be very useful. It can save time, money and prevent you from any excessive sunburns, if you forgot that you are actually outside, starring on your screen, exposed to the hot summer sun.

For this task the ArduPilot simulator can be used. It simulates a vehicle runnig the ArduPilot flight stack (firmware) on your local machine. Data is beeing published by the simulator via UDP and should ideally connect to QGroundControl without any further tweaks.

The simulator is part of the ArduPilot project, which can be downloaded from Github: `https://github.com/ArduPilot/ardupilot`. It is recommanded to fork the repository. After cloning the repository, the submodules must be initialized and updated. Execute the following code to do this.

```
git clone https://github.com/ArduPilot/ardupilot
cd ardupilot
git submodule init
git submodule update
```

The simulator is launched by a Python script, thus Python must be installed on your machine. The simulator is now ready to run, it can be launched from your ArduPilot root directory with the following command.

```
./Tools/autotest/sim_vehicle.py -v ArduCopter
```

Thereby the `-v` option specifies the vehicle type. The `-l` option can be used to define a custom start location. See the `sim_vehicle.py --help` option for further information. After launching, the vehicle should appear inside QGC.