

*Excerpt from*  
Intelligent Surveillance with MAVs  
R&D 1042-AM-01  
**Paparazzi User's Manual**

École Nationale de l'Aviation Civile, Toulouse, France

October 2007  
*Updated: February 3, 2008*



## Abstract

This document is a quick guide to use the Paparazzi system. It is divided into two parts. In the first part an extensive procedure, which can also be used as a check-list, is given to prepare a flight, to operate the UAV and to analyze the results. The procedure is described for a given flight plan and a pre-programmed and tuned aircraft. The second part gives a description of the different components and tools of the system.

The Paparazzi system is in perpetual evolution; the up to date documentation can be found in the project wiki: [paparazzi.enac.fr](http://paparazzi.enac.fr).

*Important note: This manual has originally be written for a specific airframe (Multiplex Microjet), a specific flight plan (`basic.xml`), specific settings, etc. Some details may differ with another configuration.*

# Contents

<b>1</b>	<b>Quick Guide</b>	<b>2</b>
1.1	Flight Preparation . . . . .	2
1.2	On the field . . . . .	6
1.3	Post Flight Analysis . . . . .	10
<b>2</b>	<b>Reference Manual</b>	<b>12</b>
2.1	Paparazzi Center . . . . .	12
2.2	Ground Control Station . . . . .	14
2.3	Flight Plan . . . . .	17
2.4	Airframe File . . . . .	23
2.5	Settings . . . . .	24
2.6	Tuning . . . . .	25

# Chapter 1

## Quick Guide

Operation of the UAV is divided into three parts: preparation, flight and debriefing.

### 1.1 Flight Preparation

The flight preparation has to be done several hours before the flight, especially because of the batteries charging time.

#### 1.1.1 Hardware

##### Batteries

Charge the batteries for:

1. the aircraft;
2. the RC transmitters;
3. the laptop.

##### Mechanics

The following checks have to be done prior to every flight:

1. Check all the fixed and mobile surfaces. Check that all the removable parts correctly fit in place.
2. Check the gears and throws of the servos.
3. Balance with every parts in place: The aircraft must be horizontally stable when it is supported by the CG marks. Adjust with the position of the battery if needed.
4. Check the fixation of the propeller

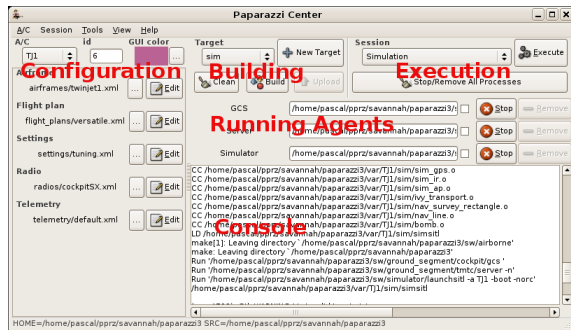



Figure 1.1: Paparazzi Center

#### 1.1.2 Simulation

To ease the flight operations on the field, it is a good idea to simulate the mission. With an Internet connection, it also permits to download the appropriate map tiles.



On the GCS laptop, run the the Paparazzi Center (figure 1.1, c.f. section 2.1) application (from the Paparazzi icon .

1. From the A/C combo box (top left), select the desired aircraft.
2. From the Target combo box (top middle), select the desired sim target and click Build (by the way, this action is useless since the configuration of the aircraft has not been changed).
3. From the Session combo box (top right), select Simulation and click Execute. A small window for the aircraft and a large window for the GCS appear. In the

Paparazzi Center, three running agents (GCS, Server and Simulator) are listed

4. If you want to change the initial default location of the mission, you need to restart the simulator agent:
  - (a) Stop the Simulator agent (with the **Stop** button on the right side).
  - (b) Edit the Simulator agent line to remove the `-boot` option.
  - (c) Restart the Simulator agent (with the **Redo** button).
  - (d) In the aircraft window (which just appeared), click **Set Pos** and enter the desired location (in WGS84 decimal degrees) and ground altitude (in  $m$ ).
5. The GCS window (figure 1.2) displays the status of the aircraft. Center the aircraft in the map with the **Center A/C** entry from the aircraft menu. Adjust the window to the displayed elements (aircraft and waypoints) with **Fit to window** from the **Nav** menu (or with the `f` key). Zoom with the mouse wheel, **PageUp**, **PageDown** keys or from the spin button (top right). Pan with the arrow keys or with purple arrows (top left). A detailed description of the GCS is given in section 2.2.
6. If you want to download map tiles for the location of the mission:
  - (a) Connect the laptop to the Internet
  - (b) Downloads can be done in several ways:
    - Mouse right-click provides a menu to download one tile at the mouse location
    - Google Maps fill from the **maps** menu fills the whole window with tiles
    - Google Maps auto will fill the window with tiles while the window is panned or zoomed.

The downloaded tiles are stored in a cache; Next time the GCS is run, they will be fetched from this cache.

7. The operation of the mission from the GCS is then the same in simulation than for a real flight. It is described in section 1.2. After waiting for the GPS initialization time (10s), navigation should be in the **Holding Point** block (strip figure 1.3). Switch to the **Takeoff** block (  ) and launch the aircraft, from the aircraft window (**Launch**) or from the GCS strip (  ).
8. If you want to take into account the wind for your simulation:
  - (a) From the **Tools** menu in the Paparazzi Center, run an **Environment Simulator** agent. The agent is listed in the running agents and a new window (named **Gaia**) pops (figure 1.4).
  - (b) In this window, set the direction (in degrees, clockwise from the north) and the speed (in  $m/s$ ) of the wind.

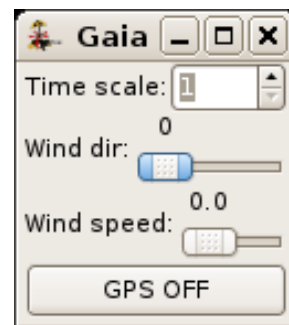


Figure 1.4: Environment Simulator

9. To accelerate the simulation, up to a factor of 10, the time scale can be changed from the **Environment Simulator** agent.
10. To start a new simulation, it is possible to restart (from the Paparazzi Center)

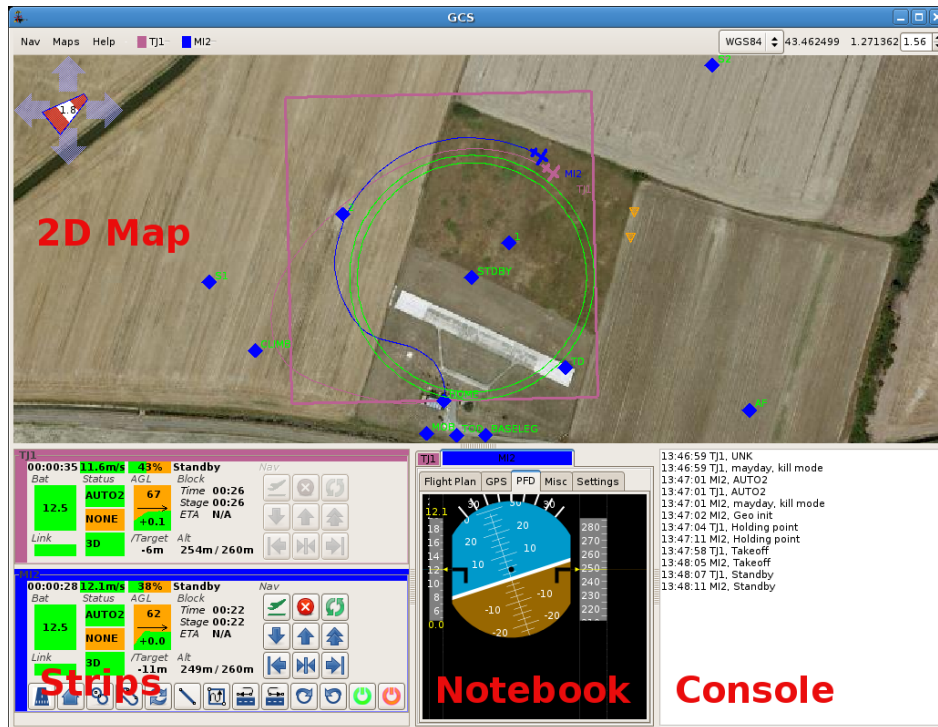


Figure 1.2: GCS

- Only the Simulator agent (Stop and Redo).
- All the agents, one by one or with Stop/Remove all processes and Execute.
- 5. For repeated flights: battery charger and power supply.

11. The set of running agents can be saved as a custom Session from the Sessions menu for later use.

To quit, click Stop/Remove processes and close the Paparazzi Center.

### 1.1.3 List of items to carry to the field

1. Complete aircraft with all removable parts
2. RC transmitter
3. Ground datalink modem
4. GCS Laptop, flashing USB cable

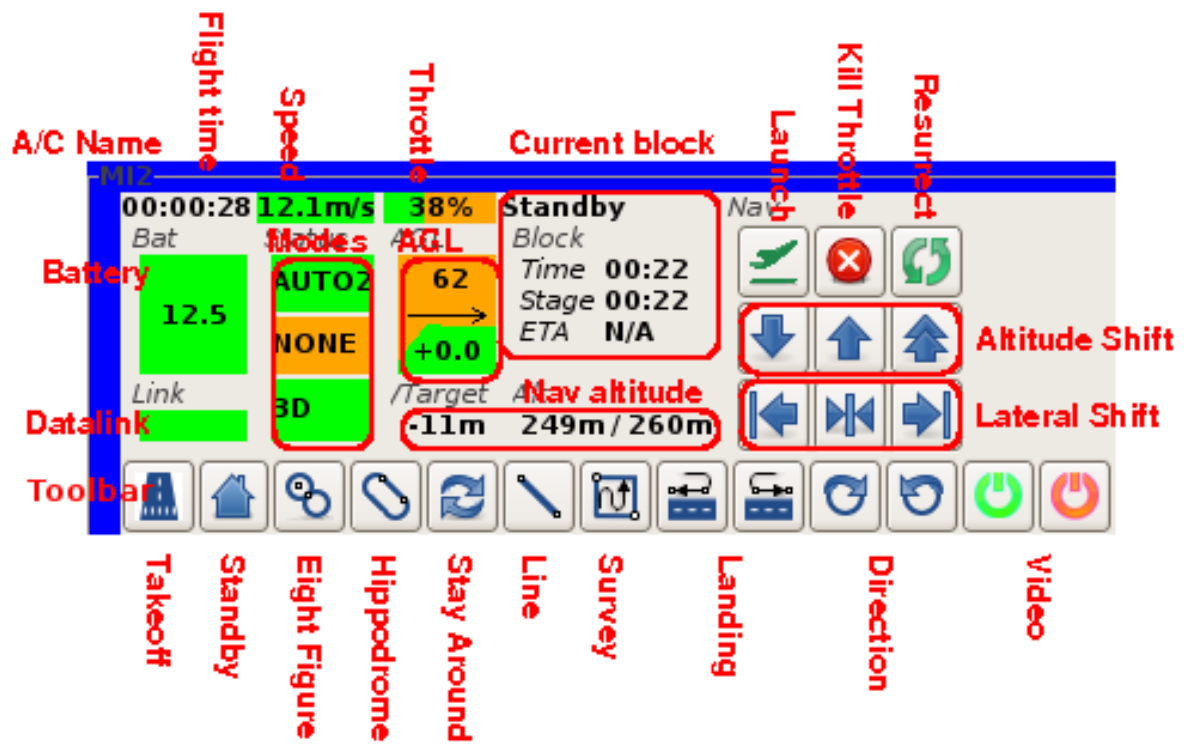


Figure 1.3: The strip regroups the main flight parameters. Note that the bottom toolbar is a set of buttons related to the flight plan and to the settings configuration.

## 1.2 On the field

### 1.2.1 Installation

1. Boot the GCS laptop (Linux) and launch Paparazzi Center
2. Plug the USB datalink modem. Adjust the antenna to get a vertical orientation.

### 1.2.2 Pre-Flight

Run the system:

1. Select and execute the Flight session from the Paparazzi Center.
2. Switch on the RC transmitter, throttle off, MANUAL mode.
3. Plug the aircraft to its battery. Close its cockpit.
4. Check the downlink and uplink on the GCS.
5. Wait for GPS acquisition (c.f. figure 1.5)
6. Load background maps (fetched from the cache if they have been downloaded during a previous session).

Aircraft:

1. Check with the RC transmitter the elevons' travel and direction in MANUAL and AUTO1.
2. Check the motor (full throttle) in MANUAL and AUTO1.
3. Check the stabilization in AUTO1:

**Infrared** Handle the aircraft by the nose and tilt it to its right side by about 30°. Check that the right elevon raises up and that the left one goes down. Handle the aircraft by a wing and tilt it nose down. Check that both elevons raise up. Check the other directions in the same way.

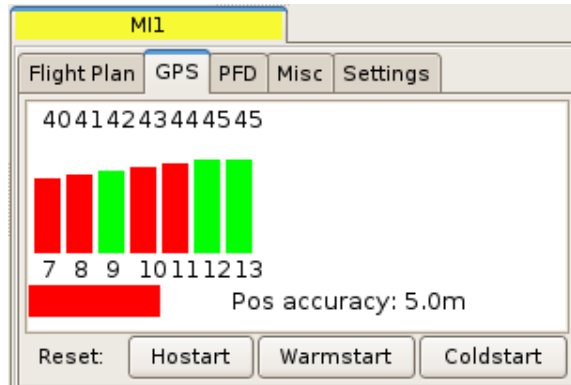




Figure 1.5: The GPS page of the A/C notebook displays the levels for the tracked satellites. During the pre-flight, the operator can ask for a reset of the receiver.

**Gyro (optional)** Handle the aircraft by the nose and tilt it quickly. Check that the elevons move in the opposite direction.



### 1.2.3 Takeoff

1. Check the flight time (strip, figure 1.3). If it is not null, reset it by clicking on it (confirmation is asked).
2. Move the CLIMB waypoint in the desired takeoff direction. Switch to the Takeoff block ()
3. On the RC, switch to AUTO1 or AUTO2:
  - Takeoff in AUTO1:
    - (a) Pull half-elevator.
    - (b) Set full throttle. Wait for the motor to actually run full speed (slew effect) and throw.
    - (c) Check left and right control.
    - (d) Switch to AUTO2.
  - Takeoff in AUTO2:
    - (a) Start the throttle from the GCS (launch, )
    - (b) Wait for the motor to actually run full speed (slew effect) and throw.
4. Check the battery level during the initial full throttle climb.

After reaching a safe altitude the navigation switches to the Standby navigation block.

### 1.2.4 Cruise

During the mission, the role of the GCS operator is both to ensure that the aircraft safely stay in flight and to supervise the navigation.

#### Check list

The following parameters must be continuously monitored during flight:

- Datalink status: The indicator on the strip (figure 1.3) becomes red and count the number of seconds since the last received battery level message. If the link is lost, the operator can try to restart the Data Link agent from the Paparazzi Center or to reboot the modem by unplugging, replugging it and restarting the Data Link agent.
- Flight time and battery level: The aircraft must be landed as soon as possible when the battery level continuously stay low.
- Altitude, AGL and tendency. Except in flat non-wooded countryside, it is not recommended to fly under 50m AGL. Note that 0m is used for the ground altitude if the terrain model for the current location is not available (in the `data/srtm/` folder).
- Speed and throttle. An estimation is available as a tooltip of the speed indicator.
- Active navigation block: Check that the current block is the desired one. A message is issued in the console on each block change.
- Target altitude: Check that it is compatible with the ground altitude and with the allocated airspace.
- Console: Read the messages ...

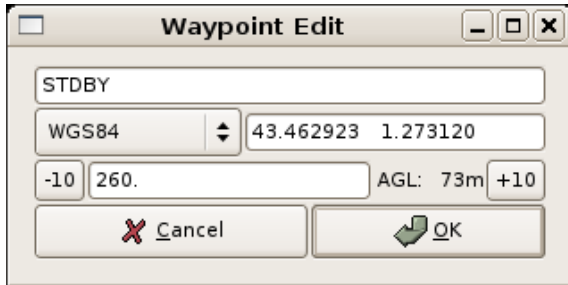



















Figure 1.6: Waypoint edition. Position may be set in geographic coordinates or relative to another waypoint (azimuth and distance). Altitude may be directly set or shifted by 10m. AGL is indicated if the terrain model is available.

## Mission

During the flight, the operators can change the mission by

- Moving waypoints: Click and drag the waypoint. A dialog box is opened for confirmation (figure 1.6).
- Changing altitude: Shift the altitude by 5m up or down, or 30m up with the arrow icons: (icondown  .
- Shifting to left or right ( ). A single shift is 5m. It is applicable on any straight route (the carrot is no longer on the route) and circle (the radius of the circle is modified). It is reset in every stage initialization.
- Going to another block pattern:
  -  Standby: Turn around STDBY. Direction can be changed with the circle arrows ( ). Radius also can be set from the Settings/nav notebook page (nav\_radius variable).
  -  Eight figure: Fly over waypoint 1 and turn around waypoint 2. When the block is active, waypoint 2 is automatically moved

close to waypoint 1 to avoid too large figures. Radius is controlled by nav\_radius.

-  Hippodrome: Fly an oval along waypoints 1 and 2. Change of direction ( ) sets the side of the oval relative to the 1-2 segment.
-  Fly around (Man Over Board): The MOB waypoint is set at the current A/C location and the UAV flies a circle around.
-  Line: Fly between waypoints 1 and 2 and back using smooth U-turns along circle arcs. This block is protected against datalink loss: If the aircraft does not receive a message from the GCS for more than 22s, the navigation switches to the Standby block. Note that the ground station periodically send at least one message (the wind estimation) to the aircraft every 10s.
-  Survey: Fly a north-south sweep in the rectangle defined by the waypoints S1 and S2, which can be anytime moved.
-  Landing: Fly a track to glide the final from waypoint AF to TD. The waypoint BASELEG is automatically placed to fly a circle around and reach AF.
-   Change the direction along circle, eight figure and hippodrome.

## HOME mode

The HOME mode is a failsafe mode where the standard navigation is suspended and the aircraft flies a circle around the HOME waypoint at a safe altitude. This mode is triggered on different events:

- Loss of the RC uplink in MANUAL or AUTO1 modes.

- Distance to the HOME waypoint is greater than a threshold set in the flight plan (displayed as a circle on the GCS map).

To get out of this mode and go back to AUTO2, click on the **HOME** indicator in the strip.

### Kill Throttle

The kill throttle status prevents the autopilot to start the motor. It is the initial status in the flight plan and is reset in the takeoff block (A). It is indicated in red color in the throttle gauge. This status is triggered

- If the battery level goes under the catastrophic low level (defined in the airframe file);
- If the aircraft goes farther than 1.5 times the HOME mode distance;
- By the operator with the kill (X) button (with confirmation).

This status is reset by the resurrect button (G).

### Plotting Data

It is possible to plot data in real time during the flight.

1. From the Paparazzi Center Tools menu, launch a Messages agent. This agent displays all the messages sent by the aircraft (figure 1.7). Select the BAT message.
2. From the Paparazzi Center Tools menu, launch a Real-time plotter agent (figure 1.8). Drag from the Messages window the voltage field of the BAT message on the Real-time plotter agent. A curve is plotted with the data and periodically scrolled to the left.

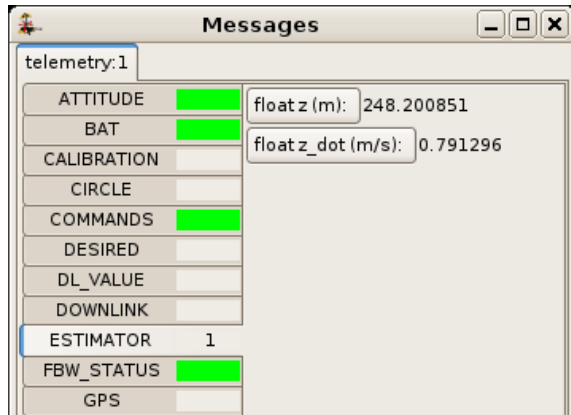






Figure 1.7: Messages agent displays the telemetry from the aircraft.



Figure 1.8: Real-time plotter.

### 1.2.5 Landing

Because of the low vertical precision and the low angle slope descent, a fully autonomous landing requires a large flat field. Even if the final touch is done manually, the procedure computed and followed by the autopilot can greatly help a RC pilot beginner.

1. Set the TD waypoint (Touch Down) to the ground level at the desired place for the landing.
2. Set the AF waypoint downwind (the wind direction estimation is indicated in the upper left corner of the 2D map), about 250m away from TD, 30m higher than TD.
3. Go to the right-hand (i.e. last turn is a right turn, ) or left-hand () landing block.
4. Final decision:
  - Touch in AUTO2: Kill throttle from the GCS () after landing.
  - Switch to AUTO1 before the touch: Control the touch and cut throttle with the RC.
  - Go around: If the landing is impossible, go back to the Standby block ()

### 1.2.6 After landing

1. Unplug the aircraft battery
2. Stop the GCS session (Stop/Remove Processes from the Paparazzi Center) and quit the Paparazzi Center.

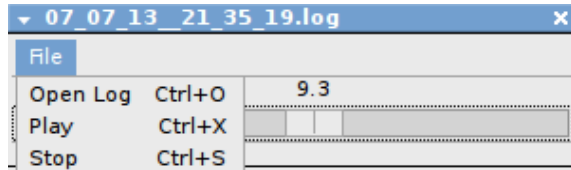


Figure 1.9: Replay agent.

## 1.3 Post Flight Analysis

1. Check that the aircraft has not been damaged by the landing.
2. Save a copy of the log files (.log and .data from var/logs/) if needed.

### 1.3.1 Data analysis

All the telemetry data received during a flight are stored and can be analyzed.

#### Replay the Flight

From the Paparazzi Center Tools menu, start a Replay session. From the Replay agent, open the log file and play it (actions from the menu, figure 1.9). The play can be suspended and restarted anywhere along the timeline.

All the recorded telemetry messages are synchronously regenerated and can be used by all the agents (GCS, Messages, ...) like during simulation and flight. Of course, the GCS operator can just observe and not modify the mission during a replay.

#### Plot Data

The Log Plotter agent (from the Paparazzi Tools menu) provides the same features than the real time plotter, using recorded data from a log file.

After opening a log file, a field of a message can be selected from the log menu to be plotted (figure 1.10).

The track can also be exported as a KML file which can be opened with the Google Earth navigator.

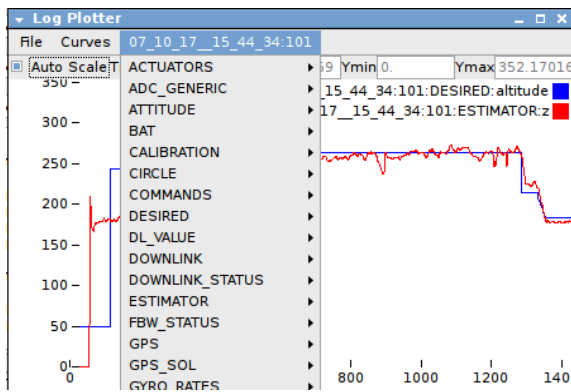


Figure 1.10: Log Plotter.

# Chapter 2

## Reference Manual

Some features of the system **are not described in the present document** and can be found on the documentation page of the project:

`ppaparazzi.enac.fr`

### 2.1 Paparazzi Center

The Paparazzi Center is the initial graphical interface of the Paparazzi System. It is used to

- Configure an aircraft as the union of its airframe, its flight plan, its settings file ...
- Build the code and the simulator.
- Run a session to simulate, fly, replay ...

#### 2.1.1 Aircraft

An *aircraft* is specified as the set of its name, its numerical identifier, its color in the GCS and a list of XML files:

- The `Airframe` file (c.f. 2.4) which contains all the characteristics of the hardware (servos, sensors, ...) and the tuning of the autopilot (control loop gains, neutrals, ...)
- The `Flight plan` which contains the set of waypoints and the set of navigation blocks which define the mission (c.f. 2.3). Examples: `basic.xml`, `versatile.xml`.

- The `Settings` files (c.f. 2.5) which contains the list of parameters which can be tuned during the flight. Several XML files can be simultaneously selected. Examples: `basic.xml`, `tuning.xml` (for extensive tuning), `light.xml` (for light duration setting), `switch.xml` (for video power switch).
- The `Radio` file which describes the RC transmitter (number and name of channels, PPM signal range, ...).
- The `Telemetry` which lists the messages which are periodically sent by the aircraft. Only expert users and developers will need something different from the `default.xml` file.

An editor can be launched from the interface for any of these files. A graphical interface is proposed to edit the flight file (c.f. 2.3).

#### 2.1.2 Compiling and Flashing

The code of the embedded autopilot (`ap` target) and of the simulator (`sim` target) must be built from the configuration files. The appropriate target is chosen from the combo box (only experts will need a *new* target). During the building, the details are reported in the console. Compilation errors usually are identified and highlighted in color.

Flashing of the code in the autopilot is executed with the `Upload` button. The aircraft has to be booted with the USB cable plugged to the computer to be flashed.

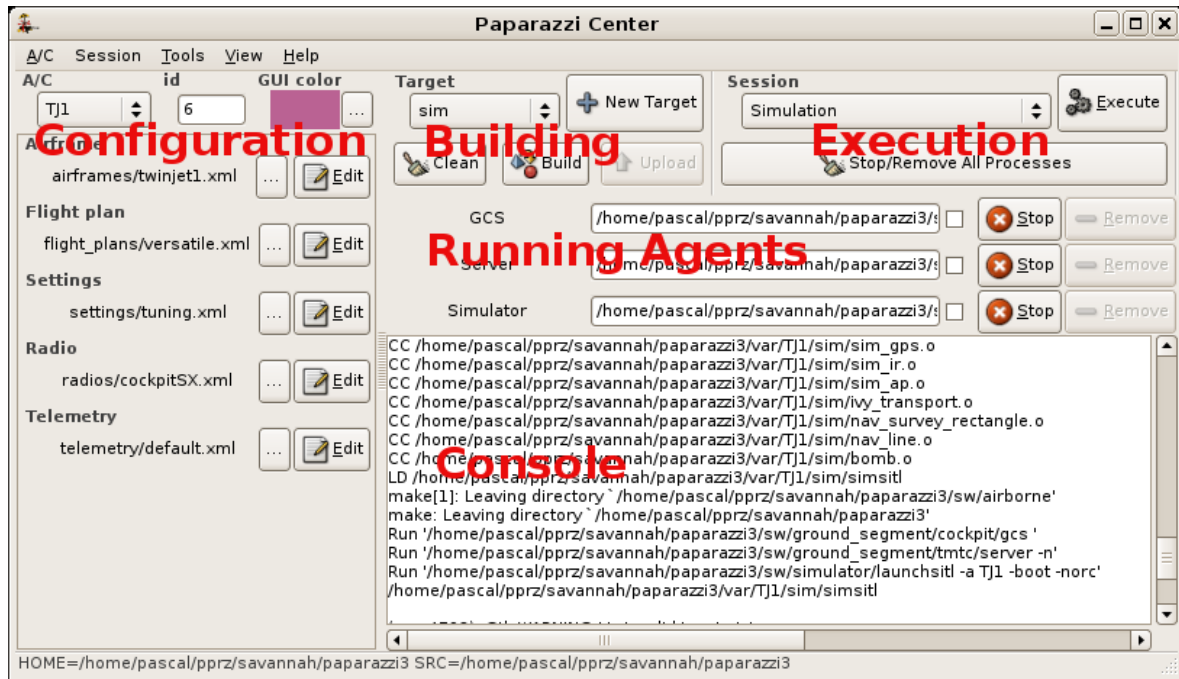


Figure 2.1: Paparazzi Center.

### 2.1.3 Simulation, Flight and Replay

Execution is done with the help of a *session* which is a set of communicating agents. Pre-defined sessions are available and it is possible to create a new custom one (from the Sessions menu). These sessions are listed in the combo box.

The running agents are listed in the Paparazzi Center and can be stopped and restarted. The automatic respawn flag restarts the agent as soon as it has been stopped.

The Simulation session runs a server, a GCS interface and a simulator of the selected aircraft. It requires that the `sim` target has been compiled for this aircraft.

A flight session at least includes a modem Link agent, a server and a GCS.

### 2.1.4 Other Agents

The Paparazzi agents can be launched from the Tools menu. Some of them need addi-

tional options to work.

**Flight Plan Editor** It runs a GCS configured with only the 2D map and a flight plan tree (c.f. 2.3.8). Edition features are available from the Edit menu.

**Simulator** It requires the `-a` option followed by the name of the aircraft to be simulated. Additional options:

- `-boot`: The aircraft is automatically started.
- `-launch`: The aircraft is automatically launched.
- `-norc`: The RC transmitter graphical interface is not displayed.

**Data Link** It drives a modem. More important options:

- `-d <port>` The serial device where the modem is plugged (usually `/dev/ttyUSB0`)
- `-s <baudrate>` (usually 9600 or 57600)

`-transport <mode>` Must be set to `xbee` when a Maxstream modem is used with several aircraft.

**Messages** Telemetry debugger (figure 1.7).

**Environment Simulator** It permits to change the wind, the time scale and the GPS status during a simulation (this agent historically is named **Gaia**)

**A Real Time Plotter** To graphically plot numerical data from the telemetry messages during a flight or a simulation.

**Log Plotter** To graphically plot numerical data from the telemetry messages recorded in a log file.

## 2.2 Ground Control Station

The GCS (figure 1.2) is the main graphical interface of the Paparazzi system. It displays the critical parameters of one or several aircraft, during flight, simulation or replay. It also allows the operator to interact with the UAV by moving waypoints, changing the navigation pattern or setting a parameter of the autopilot.

The information are displayed inside four main areas: a 2D map, the strips, a notebook and a console.

### 2.2.1 Strips

Each A/C has an associated strip that displays information about the A/C and provides buttons for common commands. When several aircraft are monitored, only one is *active*. For the others, the navigation buttons are grayed and the toolbar is hidden. Click on the strip (not on the buttons !) to *activate* the aircraft. The strip is commented on figure 1.3.

### Flight Parameters

The main flight parameters are located on the left area of the strip. The different indicators are colored: the greener, the better. Throttle becomes red if the kill throttle status is on. The datalink indicator becomes red after 5s without any battery level message received. The tooltip of the speed indicator gives the airspeed estimation. The battery and the AGL indicators display a short history of the values. AGL (Above Ground Level) indicator also displays the vertical speed and the tendency with an arrow. Flight time and navigation mode are sensitive: A click on the flight time will reset it; A click on the mode will reset to AUTO2 mode (it is useful to get out of the HOME mode).



## Navigation Parameters

The middle area regroups information about the navigation status. The name of the current block is displayed on top. Time since the beginning of the block, since the beginning of the stage and ETA (Estimated Time of Arrival) to the next waypoint (in a go stage) are given. The bottom of the area gives the current altitude, the target altitude and the difference of these two values (on the left, under the AGL).

## Navigation Control

The right panel is a group of 9 buttons. The first row controls the throttle. The launch (🚀) is used during takeoff. The kill (🛑) sets the kill throttle status (after confirmation) which prevents the autopilot to run the motor. The resurrect (🔄) resets this kill status.

The second row controls the navigation target altitude by shifting by  $+5m$  (⬆️),  $-5m$  (⬇️), or  $+30m$  (⬆️).

The third row of buttons controls the navigation in the horizontal plan. It allows to shift  $5m$  to the left (⬅️), to the right (➡️) or to recenter on the initial track (🏠).

## Toolbar

The toolbar is a set of custom buttons defined in the flight plan (c.f. 2.3.4) and in the setting file. So they change according to these two files. The buttons defined by the `basic.xml` flight plan file and by the `basic.xml` setting file are detailed in the user guide (c.f. 1.2.4).

### 2.2.2 Map

The 2D map display contains the following information:

- The A/C track: it can be erased via the Clear track option from the A/C menu.
- The A/C label contains the name of the A/C, its altitude and its ground speed.

This option default is off. It can be activated with the A/C label option from the A/C menu.

- The carrot (the orange triangle). This is the point the A/C is following during autonomous navigation.
- The waypoints defined in the flight plan (diamonds).
- The intended trajectory is shown in green (circle or segment).
- The default background is black. Google tiles or user defined maps can be loaded to provide navigation reference (maps menu).
- The camera footprint (a gray polygon) is representative of the swath of land currently seen by the on board camera. This option default is off. It can be activated with the Cam footprint option from the A/C menu.
- The WGS84 coordinates of the mouse cursor are displayed at the top right hand corner.
- A UTM kilometric grid can be added to the background via the UTM grid option from the Nav menu.
- The SRTM option from the Nav menu displays the ground altitude of the mouse location near the geographic position in the top right corner. The SRTM data files (`.hgt.zip` or `.hgt.bz2`) must be copied to the `data/srtm` directory. They can be downloaded from <ftp://e0srp01u.ecs.nasa.gov/srtm/version2/SRTM3>.

## Navigation

You can pan/zoom the map using the following:

- Pan with the blue arrows on the map or use the arrow keys on the keyboard;

- Zoom in/out with the mouse wheel, the page up/page down keys or the small up/down buttons at the top right hand corner where the zoom factor is displayed
- Fit the map to the window, in order to see all the waypoints and A/C, with the f key or the Fit option from the Nav menu;
- Center the map on an A/C with the Center A/C option from the corresponding A/C menu.

### Google Tiles

The default black background can be automatically filled with calibrated satellite images from Google servers.

### Waypoint Editing

The properties of any waypoint in the currently loaded flight plan can be modified by two methods:

- Drag and drop the waypoint to a new location (a confirmation dialog will appear);
- A single left click on a waypoint opens a dialog box where you can edit the waypoint coordinates and altitude.

Waypoint modifications are sent to the aircraft immediately upon confirmation in the dialog box. The GCS will re-send the data and the waypoint will blink until the aircraft confirms reception of the move request. New waypoints cannot be added during flight.

### 2.2.3 Notebook

The notebook frame contains one page for each running aircraft. Each aircraft page is itself divided into sub-pages displaying telemetry data and giving access to the autopilot tuning parameters.

The selected aircraft in this notebook is the *active* aircraft (c.f. 2.2.1).

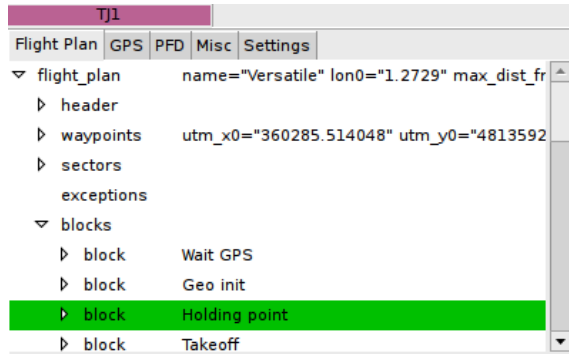


Figure 2.2: The Flight plan page of the A/C notebook displays the mission program.

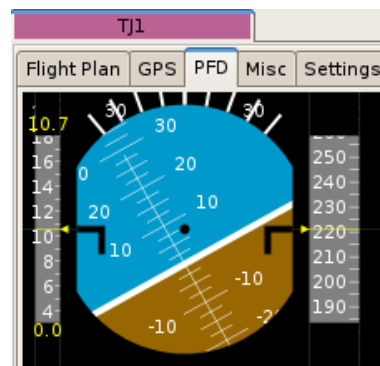


Figure 2.3: The PFD page of the A/C notebook displays attitude of the aircraft, ground speed and altitude.

### Flight Plan

This page (figure 2.2) gives a detailed view of the flight plan. Active block and stage are highlighted. Double-click on a block header switches the navigation to this block.

### GPS

The GPS page displays the level of the tracked satellites and the estimated position accuracy (figure 1.5) and gives the opportunity to reset the receiver.

### PFD

The PFD page displays the estimated attitude of the aircraft, its ground speed and its

altitude. Minimum (resp. maximum) speed is displayed below (resp. above) the speed indicator (a click on the indicator resets these values).

## Settings

The setting page allows the operator to change variable values during flight. The layout of the page is generated from the `dl_settings` section of the `settings.xml` file, one tab is associated to every section and sub-section. On each line is displayed (from left to right), the name of the variable, its current value (periodically sent by the A/C), a slider or radio buttons for user input, and commit/undo buttons.

Buttons included in the strip are associated to the major binary settings (figure 1.3).

## 2.2.4 Console

Navigation block changes and alarms are displayed in the text console.

## 2.3 Flight Plan

The formal description of the flight plan file is given in the DTD. This DTD must be referenced in the header of the flight plan XML file using the following line:

```
<!DOCTYPE flight_plan SYSTEM "flight_plan.dtd">
```

The flight plans are stored in the `conf/flight_plans` directory. The flight plan editor can be used to create basic flight plans in the GUI.

### 2.3.1 Structure of a flight plan

Extract from the DTD:

```
<!ELEMENT flight_plan
  (header?,waypoints,sectors?,
  include*,exceptions?,blocks)>
```

A flight plan is composed of two compulsory elements: `waypoints` and `blocks` and typically contains optional `include`'s and global `exceptions`.

The root `flight_plan` element is specified with several attributes:

```
<flight_plan
  name lat0 lon0 ground alt security
  height alt max_dist_from_home>
```

**name** the name of the mission (a text string).

**lat0, lon0** define the latitude and longitude of the point 0,0 in WGS84 degree coordinates.

**ground\_alt** the ground altitude (in meters).

**security\_height** the altitude used by the circle-home failsafe procedure.

**alt** the default altitude of waypoints.

**max\_dist\_from\_home** the maximum allowed distance (in meters) from the HOME waypoint.

Here is an example of the first line of a flight plan:

```
<flight_plan name="Example Muret"
  lat0="43.46223" lon0="1.27289"
  max_dist_from_home="300" alt="250"
  ground_alt="185" security_height="25">
```

### 2.3.2 Waypoints

The waypoints are the geographic locations used to specify the trajectories. A waypoint is specified with its name and its relative coordinates:

```
<waypoint name x y [alt]/>
```

where *x* and *y* are the coordinates in meters from point 0,0. The *alt* attribute is optional and can be used to assign an altitude to a particular waypoint that is different from the globally defined *alt*. Note that a waypoint named HOME is required as it is used by the failsafe HOME mode procedure.

Example:

```
<waypoints>
  <waypoint name="HOME" x="0" y="30"/>
  <waypoint name="1" x="-100" y="60" alt="270"/>
  <waypoint name="2" x="-130" y="217" alt="300"/>
</waypoints>
```

Waypoints are easily edited with the flight plan editor (c.f. 2.3.8). Waypoints which name starts with an underscore are not displayed in the GCS, except in edition mode.

### 2.3.3 Sectors

Flat Sectors can be described as a list of waypoint corners. Such an area will be displayed in the GCS. A function is generated to check if a point (usually the aircraft itself) is inside the polygon. Currently, this feature requires that the polygon is convex and described in a clockwise order. For a sector named Sector, the generated function is

```
bool_t InsideSector(float x, float y);
```

where *x* and *y* are east and north coordinated, in meters, relative to the geographic reference of the flight plan. Note: If the flight plan is dynamically relocated, such a sector will be relocated but the display is currently not updated on the GCS.

For example, with the following element in a flight plan:

```
<sectors>
  <sector name="Muret">
    <corner name="_1"/> <corner name="_2"/>
    <corner name="_3"/> <corner name="_4"/>
  </sector>
</sectors>
```

It is then possible to write an exception to stay inside the sector:

```
<exception
  cond="! InsideMuret(estimator_x, estimator_y)"
  deroute="Standby"/>
```

### 2.3.4 Blocks

Block elements are the main part of a flight plan: they describe each unit of the mission. They are made of various primitives, called stages and exceptions, you can put one after the other. When a stage (or a block) is finished, the autopilot goes to the next one. The behavior after the last stage of the last block is undefined.

As described in the DTD, the `blocks` element is composed of `block` elements which are sequences of stages:

```
<!ELEMENT blocks (block+)>
<!ELEMENT block
  (exception|while|heading|attitude|go|
  xyz|set|circle|deroute|stay|follow|
  survey_rectangle|call)*>
```

Example:

```
<block name="circlehome">
  <circle radius="75" wp="HOME"/>
</block>
```

If your system is equipped with a datalink, you can add a button in the strip of the aircraft with the attribute `strip_button`:

```
<block name="descent" strip_button="Descent">
...
</block>
```

This button will activate the block.

An icon can be specified to display the button. The `strip_button` label then is a tooltip for the icon. The icon must be an image file available in the directory `data/pictures/gcs_icons`:

```
<block name="Takeoff"
  strip_icon="takeoff.png"
  strip_button="Takeoff">
```

### 2.3.5 Expressions

Most of the numeric attributes in stages are analyzed as C expressions. The syntax of this C expression is restricted to

- Numeric constants;
- Internal autopilot variables (not fully documented, see examples);
- Binary operators: `>`, `<=`, `>=`, `<>`, `==`, `+`, `-`, `/`, `*`;
- Some utility functions.

Some examples of usable expressions are given in the next sections.

### 2.3.6 Exceptions

The flight manager can handle exceptions. They consist in conditions periodically checked (at the same pace as the navigation control), allowing the control to jump to a given block. Here is the syntax of exceptions:

```
<exception cond="..." deroute="...">
```

where `cond` is an expression and `deroute` is the name of the block we want to switch to as soon as the condition is true.

Here are some example of exceptions:

```
<exception
  cond="10 > PowerVoltage()"
  deroute="go_down"/>
<exception
  cond="(ground_alt+10 > estimator_z)"
  deroute="go_up"/>
<exception
  cond="(estimator_flight_time > 840)"
  deroute="quick_land"/>
```

Exceptions can be local to a block or global to the flight plan, in the `<exceptions>` element. In the following example, time since last reception of a message from the ground station is monitored and the navigation is switched to the Standby block if no message have been received for 22s. This exception is valid for all the blocks.

```
<flight_plan ...>
  <waypoints> ... </waypoints>
  <exceptions>
    <exception
      cond="datalink_time > 22"
      deroute="Standby"/>
  </exceptions>
  <blocks> ...
```

### 2.3.7 Navigation Primitives

Navigation modes give the description of the desired trajectory in 3D. While the horizontal mode is specified through stages, the vertical control is specified with various attributes of these stages. The current available navigation stages are

**attitude** just keep a fixed attitude;

**heading** keep a given course;

**go** go to a given waypoint;

**circle** circle around a waypoint;

**follow** follow another aircraft.

The vertical control is achieved using the `vmode` attribute of these stages. The possible values are

**alt** (default): The autopilot keeps the desired altitude which is the altitude of the waypoint (if any) or the altitude specified with the **alt** attribute;

**climb** The autopilot keeps the desired vertical speed specified with the **climb** attribute (in *m/s*);

**throttle** The autopilot sets the desired throttle specified with the **throttle** attribute (between 0 and 1);

**glide** The autopilot keeps the desired slope between two waypoints.

The default control is done with the throttle. However, setting the **pitch** attribute to **auto** and the **throttle** attribute to a constant allows a vertical control only by controlling the attitude of the A/C. The **pitch** attribute also can be set to any value (in degrees) while the throttle control is in use: it usually affects the airspeed of the aircraft.

The different navigation modes are detailed in the next sections.

## Circle

The circle primitive is one of the main navigation modes: the trajectory is defined as a circle around a given waypoint with a given radius:

```
<circle wp="HOME" radius="75"/>
```

A positive radius makes the UAV fly clockwise, a negative counter-clockwise.

The **until** attribute may be used to control the end of the stage. The following example defines an ascending trajectory at constant throttle, nose up, over growing circles, until the battery level is low:

```
<circle wp="wp1"
  radius="50+(estimator_z-ground_alt)/2"
  vmode="throttle" throttle="0.75"
  pitch="0.3"
  until="10>PowerVoltage()"/>
```

## Go

The go primitive is probably the most useful one. Basically, the autopilot will try to join a given waypoint (**wp**, the only required attribute). So the simplest thing you can ask for is

```
<go wp="HOME"/>
```

which will set the **HOME** waypoint as the desired target position. Note that since **vmode="alt"** is the default, the altitude of the target waypoint is also taken into account. The navigation will switch to the next stage as soon as the target is reached.

It is usually not a good idea to try to join a waypoint without asking for a precise trajectory, i.e. a given line. Setting the **hmode** attribute to **route**, the navigation will go over a segment joining two waypoints:

```
<go from="wp1" wp="wp2" hmode="route"/>
```

The target altitude is the altitude of the target waypoint; It can also be set with the **alt** attribute. The following example keeps an altitude with fixed throttle:

```
<go from="wp2" wp="wp3"
  hmode="route"
  pitch="auto" throttle="0.75"
  alt="ground_alt+100"/>
```

The attributes related to the vertical control can also be set to replace the default altitude mode:

```
<go from="wp1" wp="wp2" hmode="route"
  vmode="climb" climb="1.5"/>
```

Finally, the **approaching\_time** (in seconds) attribute helps to decide when the target is reached. It can be set to 0 to go over the target waypoint (default value is the **CARROT** time, set in the airframe configuration file).

```
<go from="wp1" wp="wp2" hmode="route"
  approaching_time="1"/>
```

## Attitude

Element `attitude` is the navigation mode which corresponds to the current lowest control loop for horizontal mode. The autopilot then keeps a constant attitude. The roll attribute is required (in degrees, positive to put right wing low).

To fly away, at constant airspeed:

```
<attitude roll="0"
  vmode="throttle", throttle="0.5"/>
```

To fly around, holding a given altitude:

```
<attitude roll="30" alt="ground_alt+50"/>
```

Note that it is not a safe navigation mode since the geographic position of the plane is not controlled. However, this mode is useful to tune the roll attitude control loop.

## Heading

Heading primitive is relative to the second level loop for horizontal mode in the autopilot which will keep the given course, a required attribute (in degrees, clockwise, north=0, east=90).

One example to takeoff, following the QFU, 80% throttle, nose up until height of 30m is reached:

```
<heading course="QFU"
  vmode="throttle" throttle="0.8"
  pitch="0.3"
  until="(estimator_z>ground_alt+30)"/>
```

## Deroute

The `deroute` is the `goto` directive of the flight plan; It switches the navigation to the given block: `<deroute block="landing"/>`

Note that this primitive should not be used to execute loops which are provided by the elements of the next sections.

## Loops

Unbounded loops are written with the `while` element whose `cond` attribute is a boolean expression. Children of `while` are stages:

```
<while cond="TRUE">
  <go wp="A"/>
  <go wp="B"/>
  <go wp="C"/>
  <while cond="5 > stage_time"/>
</while>
```

In this example, we run an infinite loop, going to waypoints A, B and C and waiting for 5 seconds before repeating.

Bounded loops are written with the `for` element:

```
<for var="i" from="0" to="3">
  ...
</for>
```

where the body of the loop will be run four times.

The variable of a `for` loop can be used inside expressions appearing as attributes of the stages:

```
<for var="i" from="1" to="5">
  <circle
    wp="HOME" radius="75"
    alt="ground_alt+50*$i"
    until="stage_time>10" />
</for>
```

In this example, we circle around HOME for 10s at height 50m, 10s at height 100m, ... until height 250m. Two bounded loops using the same control variable are not allowed in the same block.

## Follow

The `follow` is a special primitive which makes the UAV follow another UAV (real or simulated, named with its `ac_id`) at a given distance (in meters) behind and and given height (in meters) above.

In this example, the autopilot will try to follow A/C number 4, staying 50m behind and 20m above.

```
<follow ac_id="4"
  distance="50" height="20"/>
```

### Set

The `set` element is a dangerous one which should be used only by expert users: It is used to directly set an internal variable of the autopilot. For example, you can change the value of the default ground altitude, a variable used by the home mode failsafe procedure (and maybe by your own flight plan):

```
<set var="ground_alt"
  value="ground_alt+50"/>
```

This directive is extremely powerful and has great potential for error. Use with caution.

### Call

The `call` allows the user to define its own navigation procedures in C. The value must be a call to a boolean function which must return `TRUE` as long as the stage is not completed (a function which should be called only once would then return immediately `FALSE`). This feature is illustrated with the line pattern:

```
<call fun="nav_line_init()"/>
<call fun="nav_line(WP_1, WP_2, nav_radius)"/>
```

where `nav_line_init()` returns `FALSE` and `nav_line()` always returns `TRUE` (this stage never ends). Such functions usually are defined in a supplementary C file which must be specified in the airframe file (in the makefile section, c.f. 2.4)

```
ap.srscs += nav_line.c
sim.srscs += nav_line.c
```

These functions also must be declared in a header file which must be mentioned in the header element of the flight plan:

```
<header>
#include "nav_line.h"
</header>
```

These C source file and H header file must be located in the `sw/airborne` directory.

### 2.3.8 Edition

The easiest way to edit a flight plan is to use the graphical interface, the Flight Plan Editor (figure 2.4) available from the Tools menu of the Paparazzi Center (c.f. 2.1). A flight plan file can be opened and closed from the Edit menu. The upper part displays the standard 2D map where the waypoints can be moved and edited. A new waypoint is created with the left mouse button while the CTRL key is hold. The bottom area displays the flight plan as an XML tree. The elements of this tree can be edited:

- The elements can be deleted, copied, inserted from the right click menu.
- The elements can be dragged and dropped in the tree (left click).
- The attributes of the selected element are shown in the right column. The values of the attributes are editable entries. Some attributes can be added from the right click menu.

Note that this is a semantic editor: The attribute and element menus try to propose entries which are valid with respect to the DTD.

A flight plan file can also be edited with any text editor. The common editors (e.g. `gedit`) recognize the syntactic features of the XML files and provide fancy highlightings.





Figure 2.4: Flight Plan Editor

## 2.4 Airframe File

The airframe configuration file is located in `conf/airframes` and contains all the hardware and software settings for an aircraft. This is an XML file containing some Makefile code at the bottom. All gains, trims, and behavior settings are defined with standard XML elements. The hardware definitions such as processor type, modem protocol, servo driver, etc. are contained in the `makefile` raw section. This file can be edited with any standard text editor.

The complete description of this file can be found in the project wiki documentation. We give here only the more commonly changed attributes in the file.

### Servos Neutrals

The servos neutral values are specified in the `servos` element (usually at the top of the file).

```
<servos>
  <servo name="AILEVON_LEFT" no="1"
    min="2000"
```

```
    neutral="1490"
    max="1000"/>
```

```
...
</servos>
```

This `neutral` attribute must be the neutral trim in `MANUAL` mode. It is available in the `ACTUATORS` telemetry message (in the Messages agent, c.f. figure 1.7) during flight.

### Infrared Neutrals

The `INFRARED` section contains the infrared neutrals for the roll and pitch angles. These neutrals are usually trimmed in `AUTO1` mode.

```
<section name="INFRARED" prefix="IR_">
  <define name="ROLL_NEUTRAL_DEFAULT"
    value="5" unit="deg"/>
  <define name="PITCH_NEUTRAL_DEFAULT"
    value="5" unit="deg"/>
```

```
...
```

Note: These values are specified in degrees in the file but they are displayed in radians

in the GCS setting tool. So the values resulting of a tuning with the GCS have to be multiplied by  $\frac{180}{\pi} = 57$  to be set in the file.

### Complementary C Module

The makefile section may need to be modified to add other C modules. For example, the `anemotaxis` search strategy is not included in a standard airframe file. To be used in a flight plan, the `anemotaxis` module has to be added, for the airborne target (`ap`) and for the simulator target (`sim`):

```
ap.srcs += anemotaxis.c
sim.srcs += anemotaxis.c
```

## 2.5 Settings

The grammar of the setting file is described in `conf/settings/settings.dtd`. It is a tree of named variables. Each variable is associated with a `min`, `max` and `step` attributes. These attributes are used to build the graphical interface of the `Settings` page in the GCS. A simple entry looks like

```
<dl_setting max="2" min="0" step="1" var="pprz_mode">
```

More attributes may be added:

- `shortname="s"`: `s` will replace the variable name for the label in the GCS.
- `module="m"`: It specifies the file where the variable is coming from. A corresponding `#include "m.h"` will be generated in the corresponding C code.
- `handler="h"`: Specifies a macro to be called to do the setting. Associated with a module `m` the macro actually must be named `m_h()`.

Buttons, packed in the strip in the GCS, may be associated to variables. They are described as `strip_button` child elements of a `dl_setting` element:

```
<dl_setting max="1" min="0.0" step="0.05"
  var="v_ctl_auto_throttle_cruise_throttle"
  shortname="cruise throttle"
  module="fw_v_ctl"
  handler="SetCruiseThrottle">
  <strip_button name="Dash" value="1"/>
  <strip_button name="Loiter" value="0.1"/>
</dl_setting>
```

For a prettier strip, an icon can be used for a strip button:

```
<dl_setting max="1" min="0" step="1"
  var="power_switch"
  module="autopilot" handler="SetPowerSwitch">
  <strip_button name="Switch on" icon="on.png" value="1"/>
  <strip_button name="Switch off" icon="off.png" value="0"/>
</dl_setting>
```

The image file must be located in the `data/pictures/gcs_icons` directory.

The files `basic.xml` and `tuning.xml`, located in `conf/settings` are versatile examples of settings files.

## 2.6 Tuning

Paparazzi being an extremely customizable system, it is beyond the scope of this document to describe a complete tuning procedure that could be followed during the adaptation of the autopilot to a new type of airframe. Nevertheless, we will here underline a simple tuning procedure that could be used on a previously tuned aircraft, for example following a heavy maintenance, such as the replacement of an actuator.

### 2.6.1 Actuators layer

For optimal performances, the control surfaces of the vehicle need to be trimmed such as to allow for a neutral flight with zero command applied. In order to perform the vehicle trimming, the following procedure can be followed. It must be noted that the procedure involves fully manual flight, which requires the assistance of an experienced R/C pilot.

- Make sure no trim is applied on the radio control transmitter.
- Let the R/C pilot takeoff in manual mode.
- Let the R/C pilot adjust the trims as would be done with a conventional R/C aircraft. Level flight should be trimmed for an airspeed similar to the nominal airspeed expected in automatic mode.
- Once satisfactory behavior is achieved, land the aircraft, keeping trims offsets.
- Launch the Messages program and look for the ACTUATORS message.
- Change the neutral fields in the servos section of the airframe configuration file ( see 2.4) to match the values read in the ACTUATORS message. The association is to be done using the no field of the servos section elements.

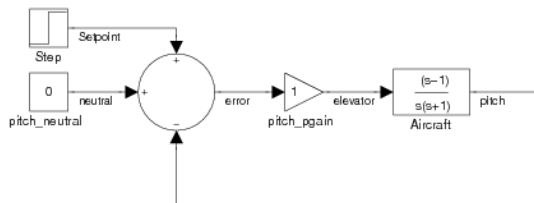


Figure 2.5: Pitch loop.

- Build and flash the airborne program. Re-center trims on the R/C transmitter.

Any adjustment to the neutral position of the control surfaces is likely to have an impact on the automated modes behavior which may in turn require adjustments.

### 2.6.2 Stabilization Layer

The attitude stabilization layer consists in two loops.

- The roll loop involves two gains and one neutral.
- The pitch loop only involves one gain and one neutral.

Although the pitch loop could seem easier to tune, this is in practice not the case due to the difficulty to maintain high pitched attitudes for a long period of time.

#### Pitch loop

The pitch control is a simple proportional loop as represented on figure (figure 2.5).

- Adjust `pitch_neutral` to obtain the desired airspeed while leaving the set-point at 0.
- Adjust `pitch_pgain` to obtain a satisfactory response to a pitch step command. A too low value won't allow for reaching the set-point value. A too high value will induce oscillations. Once obtained a correct value for the gain, make sure

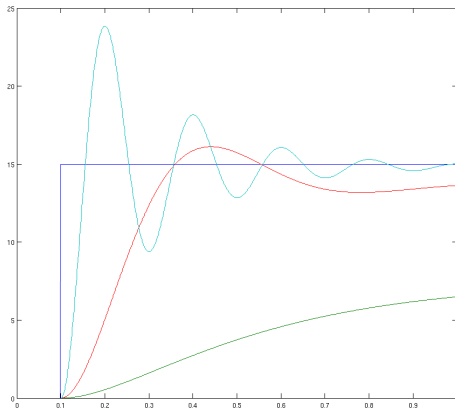


Figure 2.6: Insufficient, correct and too high value of pitch\_pgain.

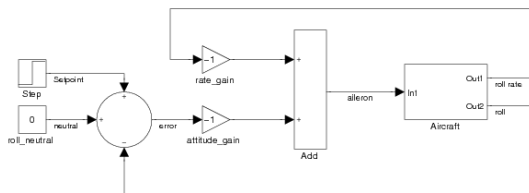


Figure 2.7: Roll loop.

to test it for oscillations at higher speed. Place the aircraft in dive and stall positions and make sure the autopilot is able to recover.

Note : In order to be able to successfully adjust the neutral value, the gain must have a sufficient value. Be careful not to force a high neutral value with a low gain, and then increase the gain. This could lead the aircraft to a commanded extreme pitch attitude and cause stall or over-speed.

### Roll loop

The roll control relies on a slightly more sophisticated P-D loop. Just as with pitch, `roll_neutral` should be adjusted so that the aircraft flies straight with zero command applied. And just as with pitch, the neutral value can only be meaningfully adjusted

when the gain parameters have a high enough value to allow for an efficient control.

The roll loop tries to achieve a pair of antagonist goals which consist in reaching a given roll angle and maintaining a zero roll rate. The roll attitude `pgain` and roll rate `gain` respectively account for the importance given to each one of these goals. Increasing the value of roll attitude `pgain` will allow the vehicle to reach its commanded bank angle faster, but will induce overshoot and oscillations. Increasing the value of roll rate `gain` will dampen the oscillations, at the cost of a decreased attitude change response time.

- You can apply the steps using the rc transmitter stick in `auto1` or a specially written flight plan in `auto2`
- It's easier to look at the plane for scarce adjustments, then to use the real time plotter for fine tuning
- You will want to choose the lowest value of gain that allows for a good control. This will provide an advantage in term of robustness and power efficiency.
- This adjustment is obviously linked with the accuracy of the attitude estimation. If your attitude is off by a factor of two, this could be compensated by a twice larger value of roll attitude `pgain`. This would nevertheless be a bad idea to do so as it would confuse the upper navigation layer.

### 2.6.3 Navigation Layer

The navigation layer is mostly based on a proportional loop just like the pitch loop. An error in course ( you can think heading if neglecting the wind ) is transformed into a bank set-point by a coefficient called course `pgain`

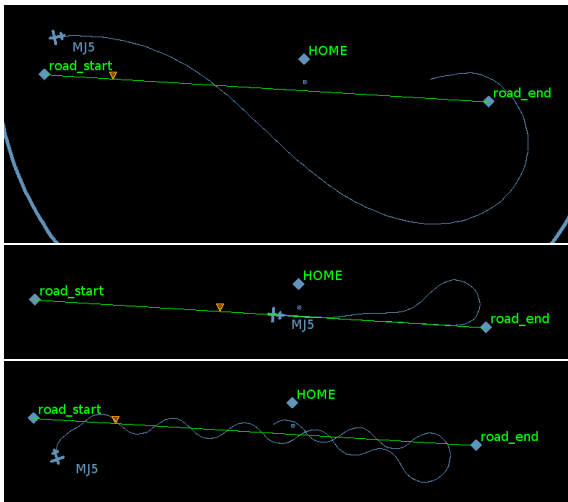


Figure 2.8: Insufficient, correct and too high value of course\_pgain.