The Paparazzi Solution

Pascal Brisset, Antoine Drouin, Michel Gorraz, Pierre-Selim Huard, Jeremy Tyler

The Paparazzi Solution[*]

Pascal Brisset, Antoine Drouin, Michel Gorraz
Pierre-Selim Huard and Jeremy Tyler
ENAC, Toulouse, France
`mailto:drones@enac.fr`
`http://www.recherche.enac.fr/paparazzi`

October 2006

Abstract

When attending Micro Air Vehicle (MAV) competitions, one notices a majority of teams concentrating on the vehicle itself, be it from an aerodynamics, structures, propulsion or control point of view. A total system approach is not often seen, though MAVs pose a very unique challenge to the system designer who must propose a platform and architecture to support the autopilot, the communications, user interface, simulation, analysis and tuning tools, flight plan management and many other details. Since its inception, the Paparazzi team considered the MAV as a system and used a bottom to top approach focusing on safety, robustness and usability of the system as a whole. This paper describes the Paparazzi project approach: The complete and integrated design of hardware, software, and airframe into a robust and reliable system, proven by formal analytical methods and thousands of flight hours.

[*]Reference to *The Paparazzi Problem*[3], the problem of providing sensor coverage of a target robot.

Introduction

Paparazzi is an open-source autopilot system oriented toward inexpensive autonomous aircraft of all types. The project began in 2003 and has enjoyed constant growth and evolution ever since. The system has been used on dozens of airframes and implemented by several teams around the world. Thousands of hours of autonomous flight have been successfully achieved with the Paparazzi system.

The Paparazzi system includes the airborne processor with its required sensors, the airborne autopilot software, a ground control station, the communication protocols linking the different components and a simulation environment.

Safety and reliability have been the primary goals throughout all phases of the Paparazzi system development. The critical airborne code has been carefully designed to be as simple and short as possible for reliability and robustness. All critical airborne code has been formally proven [1], regenerated in Lustre, and isolated in hardware with a separate microcontroller.

The system has been designed to be eas-

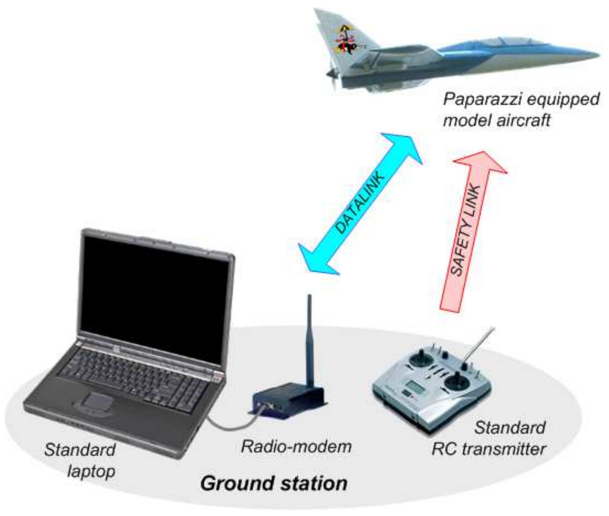


Figure 1: Paparazzi Aircraft Components

ily adapted to any type of airframe and is currently used in both fixed and rotary wing systems. The airborne software is also supported by several microprocessor architectures and many hardware configurations.

The powerful flight plan language allows the operator to define complex autonomous missions and create a logical tree of autonomous decisions for the system to make while in flight to perform any task or adapt to any scenario. The ground station operator can also manually navigate the aircraft at any time using video, real-time GPS data, and/or visual contact while relying on the autopilot to perform only the flight stabilization.

The ground control station utilizes a powerful and flexible client/server architecture which allows the operator to control one or more aircraft from a single location or from multiple locations.

Source code, hardware schematics and thorough documentation are released under the GNU Public License and are freely available for anyone to download from the project website. This open source license was chosen in the hope that our work will be used by and improved by the great number of people interested in this technology.

In this paper, we give an extensive overview of the Paparazzi system. Two other sections describe the vehicle which will be used for the MAV06 competition and a solution for the payload deployment problem. We conclude with some thoughts about the Paparazzi success story.

1 The UAV system

We give in this section a short history of the project, a description of the hardware and the different components of the system. We then conclude with some words about the safety and the usability of the system.

1.1 History

Paparazzi started in early 2003 as a hobby project. The first objective was to autonomously fly a 1.4m fixed wing model with the simplest possible airborne system. The first iteration employed a standard device for stabilization (based on infrared sensors), a GPS receiver and a basic data downlink system. A trivial interface was implemented in the ground station, receiving position information through the downlink and comparing it to a predefined flight plan. The ground computer then computed simple navigation commands which were sent back to the plane through a standard R/C transmitter. Using this primitive system, the aircraft was successfully able to achieve basic waypoint navigation and reliable artificial stability.

The next step for the project was to develop a complete airborne stability and navigation system. A simple circuit board in-

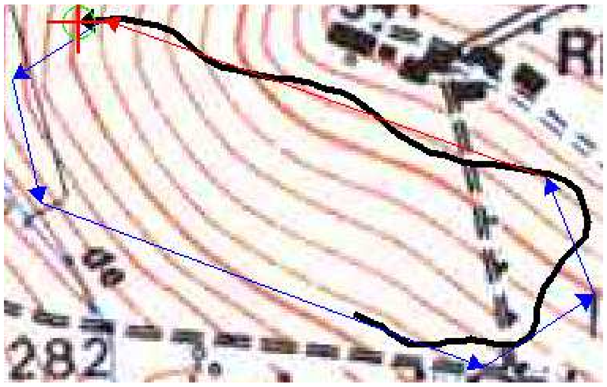


Figure 2: The first (2003) automatic navigation realized with the Paparazzi autopilot.

cluding two AVR Atmel micro-controllers was designed and constructed. The system used infrared sensors for stabilization, a GPS receiver for position and heading, and an R/C receiver for manual control. The autopilot operated all actuators and included a modem for real-time telemetry and status. Telemetry was obtained through an analog downlink channel and allowed the operator to monitor the path of the flight on the ground station. A Multiplex Twinstar aircraft (1.4m wingspan) equipped with this system won the JMD03 (Micro Air Vehicle workshop, Toulouse, France) competition.

In 2004, the code was redesigned to be configurable and customizable for any aircraft. Then the more challenging Multiplex Microjet (65cm) was equipped and tuned to fly autonomously. It successfully fulfilled the missions of EMAV05 (Braunschweig, Germany) and JMD04 (Toulouse, France) competitions.

In 2005, the Paparazzi system became packaged and popular enough to be used by other teams. The system was also once more redesigned to be able to handle (configure, simulate and control) multiple aircraft at the same time. Three new kinds of aircraft were

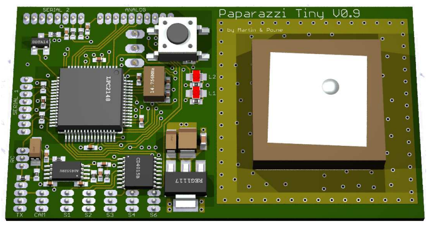


Figure 3: The *tiny* controller board includes the GPS receiver

equipped. Four teams using the system took part to the MAV05 (Garmisch, Germany) competition, and out of 11 teams, The four Paparazzi teams were the only ones to fulfill the challenging mission autonomously. The ENAC team even demonstrated two autonomous aircraft at the same time.

2006 has been the most productive of all. The airborne software has been adapted to a vastly more capable ARM7 microprocessor, while retaining full compatibility with the original AVR processor. Full 2-way communication abilities have been implemented in the ground and airborne code, allowing the operator to modify any aspect of the mission while in-flight as well as perform all tuning of a new design in a single flight. The airborne hardware has been redesigned into a smaller and more reliable package with USB connectivity and a complete GPS receiver and antenna all on a single board. Finally, the ground station has been updated with provisions for in-flight tuning, touchscreen controls, and automatic integration of Google map tiles for mission planning.

1.2 Hardware

Several versions of the autopilot hardware have been created using Atmel AVR and Philips ARM7 LPC micro-controllers. The current models include single (*tiny*, figure

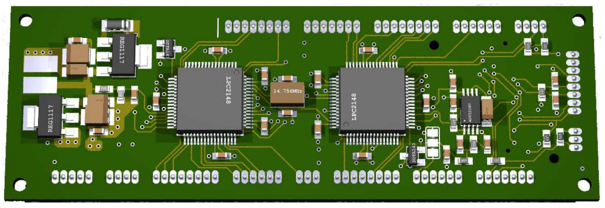


Figure 4: The dual processor *classix* controller board

3) or dual micro-controllers (*classix*, figure 4) and the required connectors to handle the servos, motor controller(s), a variety of sensors, R/C receiver, a radio modem, and other payloads. The *classix* board is designed to be connected to a payload processor, a *gumstix* based on an ARM9 microprocessor running the GNU/Linux operating system. This arrangement will allow for a tremendous amount of onboard processing for video compression, object recognition, and virtually unlimited autonomous decision making and sensor management.

1.2.1 Minimum set of sensors

In effort to keep costs low and reliability high, the current design of the Paparazzi stabilization and guidance system relies on a minimum set of sensors. Attitude estimation is done with a set of infrared sensors while position and altitude are obtained from a standard GPS receiver. Accuracy of the roll rate measurement may be improved with a gyroscope and the hardware supports the addition of a full set of inertial and magnetic sensors.

Infrared Sensors The Paparazzi autopilot uses infrared thermopiles for attitude sensing. The theory[6] is that at zero bank or pitch angle, the difference of temperature between the two sensors should be zero, and

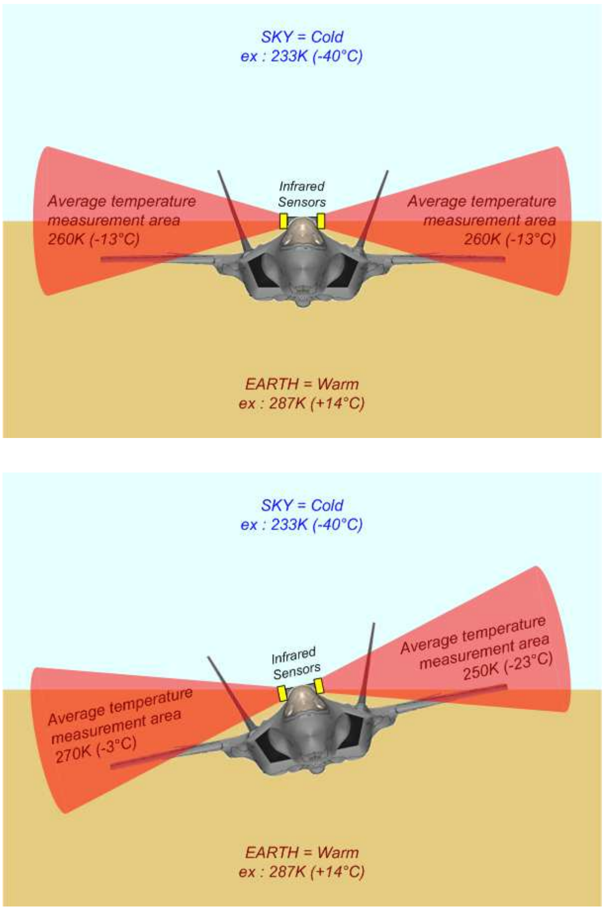
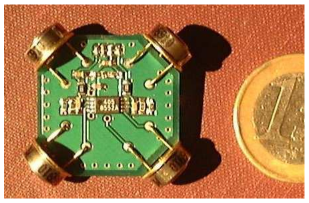

SKY = Cold
ex : 233K (-40°C)
Average temperature measurement area 260K (-13°C)
Infrared Sensors
Average temperature measurement area 260K (-13°C)
EARTH = Warm
ex : 287K (+14°C)
SKY = Cold
ex : 233K (-40°C)
Infrared Sensors
Average temperature measurement area 250K (-23°C)
Average temperature measurement area 270K (-3°C)
EARTH = Warm
ex : 287K (+14°C)

Figure 5: Infrared sensors for attitude estimation. Roll angle is deduced from the difference of temperature measured on right and left sides.

at 90 degrees it should be maximum (figure 5). From this relationship, a linear regression is made and angles are calculated during flight.



In early models two sets of sensors were used for pitch and roll, recently a third set of sensors has been integrated in order to update the maximum contrast (temperature difference between sky and earth) during the flight. For large aircraft this third set of sensors may not be necessary but for small UAVs it is a must.

It is important to note that the infrared sensor stability system is sensitive only to a very specific wavelength (7.5um 13.5um) of IR radiation that emanates from objects of common earthly temperatures and as such, the system is not affected by direct sun, water or ice. Extensive testing of IR sensor performance and limitations has been conducted by members of the Paparazzi team and the system has proved 100% reliable at night, in heavy rain, at high altitudes (1500m AGL), over open ocean, near large mountains or structures, in temperatures ranging from -10C to 48C, and across vastly different terrain transitions such as asphalt-to-water. The only known limitation of this sensor system is its expected poor behavior in dense fog or clouds.

GPS Receiver A standard u-blox GPS receiver module (based on the Antaris technology) is used for position, heading, velocity, and altitude measurement. This GPS unit (receiver and antenna) may be on the board itself (figure 3) or independent. The devices we are using support the classic GPS augmentations (WAAS, EGNOS, DGPS, ...).

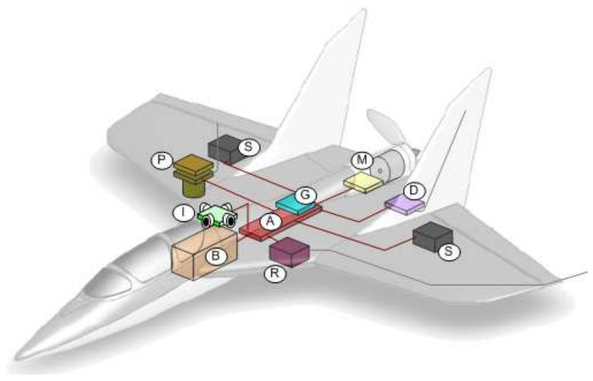


Figure 6: The Paparazzi airborne system components: (A)utopilot Control Board, (B)attery, (D)atalink Radio-Modem & Antenna, (G)PS Receiver, (I)R Sensors Board, (M)otor & Controller, (R)C Receiver & Antenna, (S)ervos, (P)ayload = Camera & Video Transmitter

Optional Gyroscope The infrared stabilization system may be slightly improved with a gyroscope on the roll axis. This gyroscope has proven especially advantageous for small aircraft with a high roll rate.

1.3 Airborne Autopilot

1.3.1 Architecture

The autopilot uses a dual processor architecture: the first processor (Fly-By-Wire) handles all critical code such as servo control and r/c override while the second (AutoPilot) manages navigation, sensors, payloads, communication, and any autonomous processing . In the case of catastrophic code error in the AP processor, the aircraft may be recovered via manual control thru the FBW processor. Two electronic boards have been designed for this architecture, one with

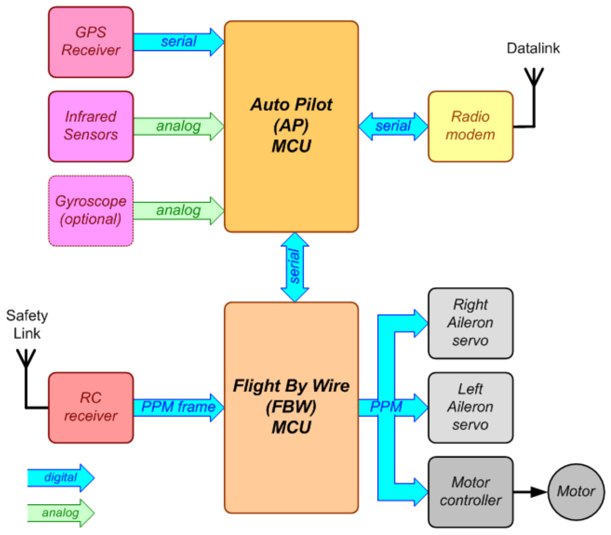


Figure 7: Links between the sensors, the communication links and the actuators around the dual processor architecture autopilot board.

Control Board	19g	$110 (parts)
Infrared Sensors	5g	$60
Gyro (optional)	(2g)	($50)
Modem	3g	$30
Total	27g (29)	$200 (250)

Figure 8: Weight and cost of the airborne autopilot system.

Atmel AVR micro-controllers and one with Philips LPCs.

Figure 7 shows the links between the sensors, the communication links and the actuators around the autopilot board.

1.3.2 Control Loops

The core of the autopilot is composed of three parts: sensor acquisition, state estimation and a stack of control loops (figure 9). Raw data provided by the sensors are filtered with a state estimator which then gives smooth inputs to the controllers. These controllers are run periodically and compute commands, either for actual actuators or for other controllers.

On the top of the stack, we run the flight plan execution at 4Hz which basically processes the flight plan in order while monitoring any special conditions associated with each stage of the flight plane (i.e. battery voltage, distance from home, distance to waypoint, sensor state, etc).

Below are the 3D position controllers (altitude and navigation). The navigation controller is responsible for following a basic section of the trajectory, a segment or a circle for example. Both controllers can be linked to follow a 3D climb or descent.

The lateral axis is then controlled with a navigation controller giving input to the stabilization controller; the latter takes a roll angle as input. The vertical axis control (climb controller) gives an input command either to the pitch controller or to the throttle slew-rate limiter. The last pitch controller is responsible for the longitudinal axis.

This control architecture allows the operator to maintain the vehicle inside of the desired flight domain both in stability-augmented and fully autonomous modes.

All these controllers are currently standard PID (Proportional-Integral-Derivative) controllers.

1.4 Ground Control Station

The role of the ground control station is to give to the user the ability to monitor the status of one or more UAVs and control them with high level orders. We describe in this section the different components and the graphical user interface of the control station.

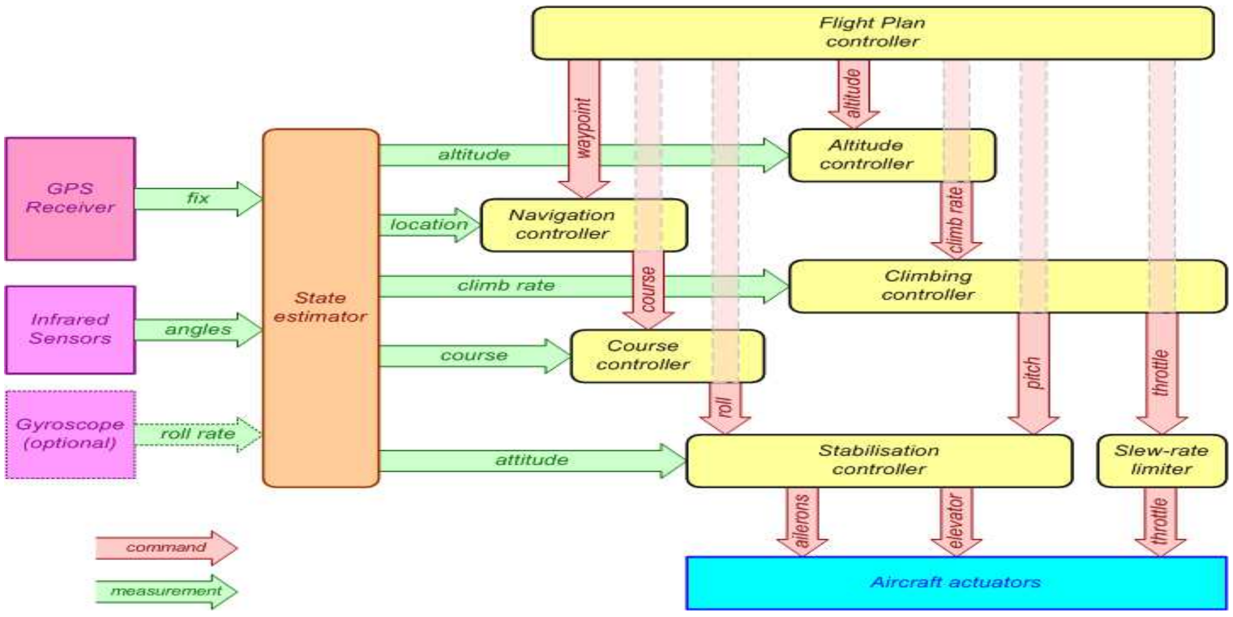


Figure 9: The stack of controllers in the autopilot.

1.4.1 Architecture

The Paparazzi ground control station is a distributed architecture (figure 10) with three main agents connected to a *bus*:

- The *link* agent is responsible for the hardware peripherals. It handles the message translation between the bus and the peripheral (usually a serial link). Several *link* agents can be used in the same time and one single agent can handle several vehicle communications on the same peripheral.

- The *server* agent records the messages coming from the *link* agents in a log and dispatches synthetic information to other listening agents (e.g. the graphical user interfaces). It also handles the configuration (airframe description, flight plan, ...) of the flying aircraft and makes it available on the bus. It also computes some environmental in-

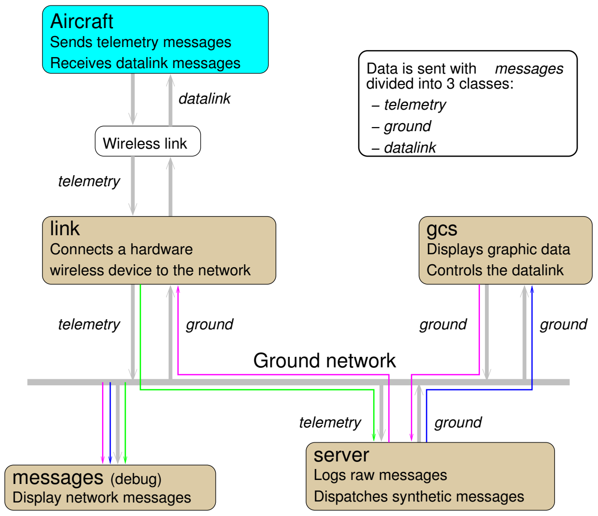


Figure 10: The agents of the Paparazzi system. Aircraft, link and gcs agents may appear several times.

formation (wind estimation from aircraft mean speed, ground altitude from an elevation model, ...) and broadcasts it on the bus.

- The *gcs* is the graphical user interface. It displays information coming from the server and sends back orders from the operator. A detailed description is given in the next (1.4.2) section.

Other agents working on the bus are also provided: a message tracer for debugging purposes, a 3D viewer, ... The simulator and the log player are also agents connected to the bus.

Paparazzi uses the Ivy software bus[1], a simple middleware on top of an IP network. Ivy is available for most programming languages and operating systems. It provides a low cost interface to connect the components of a distributed application.

Thanks to this architecture, the ground control station can run for free on a local network or even on the Internet on top of a VPN (virtual private network), with one single *server*, several *link* agents and several graphical user interface agents.

1.4.2 Graphical User Interface

The human machine interaction of UAV ground control station must look more like an air traffic controller screen than a pilot cockpit: the operator does not want to pilot but to monitor and control the UAV.

A screenshot of the Paparazzi ground station is displayed in figure 11. In this example, two aircraft are being controlled at the same time. The main window (top right) is a 2D top view of the environment showing both aircraft tracks, the waypoints, the safety radius, ... Any calibrated map can be set in the background. If an Internet connection is available, tiles of the Google database can be downloaded and displayed on the fly. The interface can also utilize SRTM (Shuttle Radar Topography Mission) data to show the altitude of the ground at any location.

The top left window shows synthetic and essential information for each aircraft (strips): battery level, flight time, datalink status, speed, altitude, etc. These strips also include some buttons for common commands to be sent to the aircraft.

The bottom right panel displays various tabs for each aircraft, particularly the flight plan progress and the tuning controls. The user can directly interact with the flight plan and adjust any parameter of the autopilot using the *settings* tab.

The lower left side shows real-time video transmitted by the aircraft and an alarm list (alarms are also audible). Snapshots of the video can be extracted from the video window and associated with the current location of the aircraft.

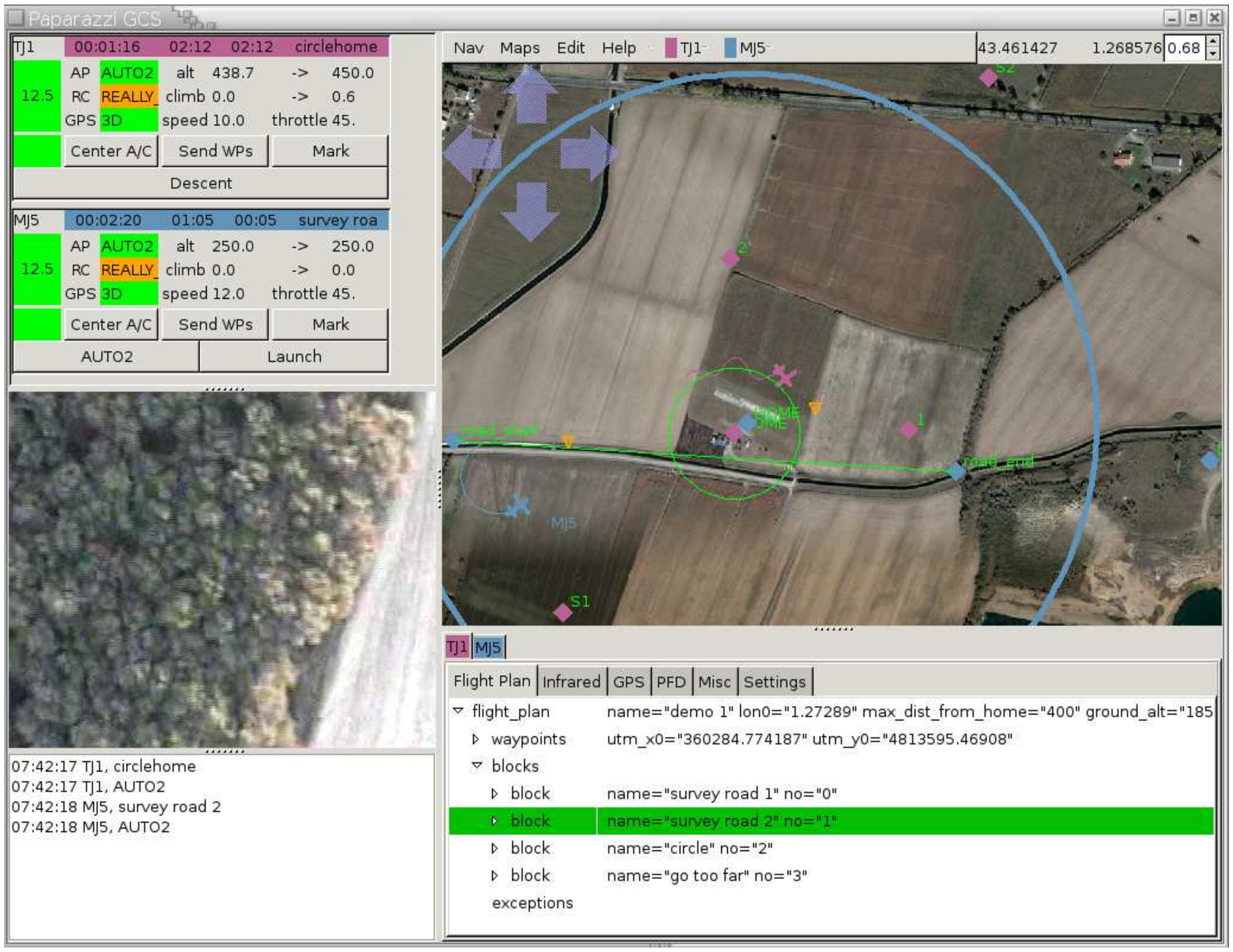
The layout of the GUI is fully configurable and the functionality of the interface has been validated by many users. Interaction is also compatible with the constraints of a touchscreen (single click, no keyboard).

1.4.3 Simulation

The flight plan language offers the ability to write complex navigation behaviors. However, the operator should of course have good confidence of this behavior before actually launching the aircraft.

The Paparazzi system includes a simulation facility able to run the airborne code on the host computer (SITL, Software In The

[1]The Ivy software, distributed under LGPL, is available from `http://www.tls.cena.fr/products/ivy/`.



```
TJ1   00:01:16   02:12   02:12   circlehome
12.5  AP  AUTO2    alt   438.7   ->   450.0
      RC  REALLY_  climb 0.0     ->   0.6
      GPS 3D       speed 10.0    throttle 45.

MJ5   00:02:20   01:05   00:05   survey roa
12.5  AP  AUTO2    alt   250.0   ->   250.0
      RC  REALLY_  climb 0.0     ->   0.0
      GPS 3D       speed 12.0    throttle 45.
```

```
07:42:17 TJ1, circlehome
07:42:17 TJ1, AUTO2
07:42:18 MJ5, survey road 2
07:42:18 MJ5, AUTO2
```

```
▽ flight_plan      name="demo 1" lon0="1.27289" max_dist_from_home="400" ground_alt="185
  ▷ waypoints      utm_x0="360284.774187" utm_y0="4813595.46908"
  ▽ blocks
    ▷ block        name="survey road 1" no="0"
    ▷ block        name="survey road 2" no="1"
    ▷ block        name="circle" no="2"
    ▷ block        name="go too far" no="3"
    exceptions
```

Figure 11: Ground Control Station (GCS) graphical user interface. Top-left: aircraft labels (strips); Top-right: 2D maps; Bottom-right: flight plan trace; Bottom-left: Alarm window; Left-center: real-time video.

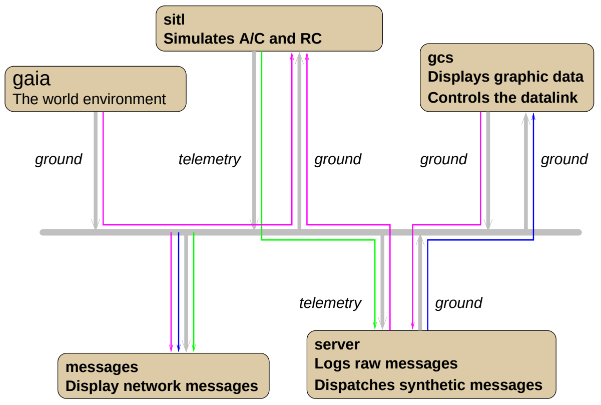


Figure 12: The agents of a Paparazzi simulation.

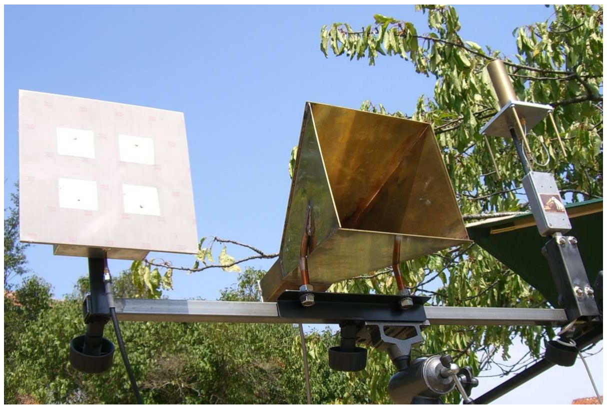


Figure 13: Using strong gain antenna for long range missions

Loop simulation) or on the autopilot hardware (HITL, hardware In The Loop simulation), while the environment, i.e. the input of the sensors, is simulated (cf figure 12). The simulator is based on the *minimum complexity* model of [4].

1.5 Safety

Safety has been the primary concern throughout all phases of the Paparazzi system development. The airborne code has

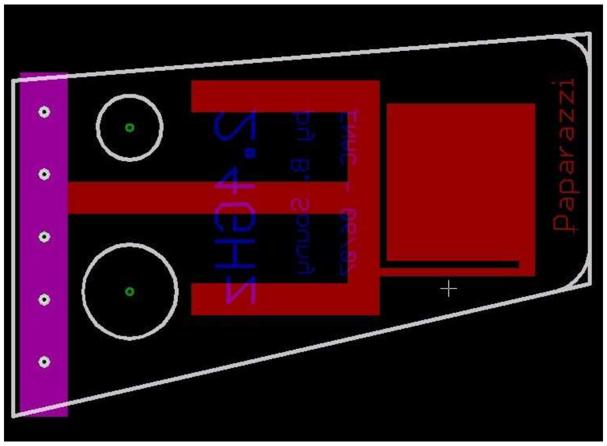


Figure 14: Custom 8cm airborne antenna (2.4GHz) designed to be included in the fin of an aircraft.

been created with an emphasis on simplicity and robustness, and all critical code has been segregated in both software and hardware for error tolerance and recovery. The critical code has been thoroughly analyzed with the help of formal methods[1] and regenerated from a high level specification in Lustre, a declarative and synchronous programming language[2], taking into account the real-time constraints.

Several failsafe modes have been implemented in order to specify a behavior in the presence of faults:

Too far from origin : Autonomous navigation back to a specified waypoint.

Not able to come back : Motor is stopped and plane begins a spiral descent.

GPS failure : User-defined behavior (e.g. circle with constant throttle for a fixed-wing aircraft).

Autopilot failure : Predefined values for all commands and safety RC link.

Figure 15: A typical GCS for Paparazzi

The ground control station also monitors the state of the system and can trigger alarms on a variety of event detections: low battery voltage, low altitude, inconsistent telemetry, telemetry failure, etc.

Thanks to its open availability, the code has also been heavily reviewed by many users.

1.6 Usability

The graphical user interface of the Paparazzi ground control station supports interaction through a single touchscreen (like the popular *Panasonic Toughbook*, figure 15). Full control of the system is possible without keyboard and only with mouse clicks.

User Feedback The Paparazzi system has evolved thanks to the contributions of the users, contributions which are possible thanks to the distribution of the whole source code and hardware schematics under the GNU license. Among the features added on demand or by users themselves, we can list:

- Greater integration of components on the board such as the GPS receiver, the power supply, etc.
- Improved software controllers in the autopilot code.
- New elements allowed in flight plans.
- Enhanced customization, video plugin, simple interaction with the graphical user interface.

Another kind of user has also appeared recently: The Paparazzi autopilot source code has been proposed as a benchmark to the WCET (Worst-Case Execution Time) community[5]. The structure of the full application has been described in AADL (a modeling language, `www.aadl.info`) providing also the map between AADL threads and the C functions. This precise analysis and modelization of the code will help to improve the behavior and the performance of the airborne autopilot.

Live CD Paparazzix, a remastered KNOPPIX (`www.knoppix.org`), is a bootable Live system on CD or DVD, consisting of all the tools required to configure, simulate and fly an aircraft with the Paparazzi system. Most laptops are supported thanks to automatic hardware detection and support for many graphics cards, sound cards, SCSI, USB devices and other peripherals.

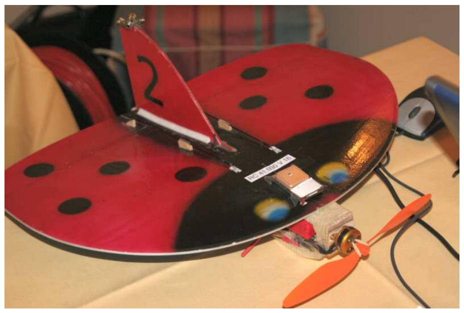
2 Vehicle

The Paparazzi team will take part in the competition with an airframe designed by

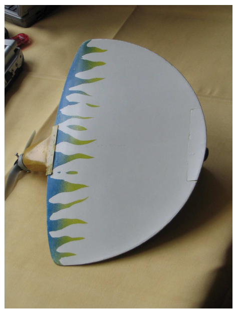
Twinstar (1.4m, 1.4kg), Twinjet (0.9cm, 1kg), both equipped with lights

Microjet (65cm, 400g)

Glotzer (50cm, 300g)



Plaster (50cm, 300g)



DragonFly (30cm, 225g)

Figure 16: Different aircraft equipped with the Paparazzi autopilot

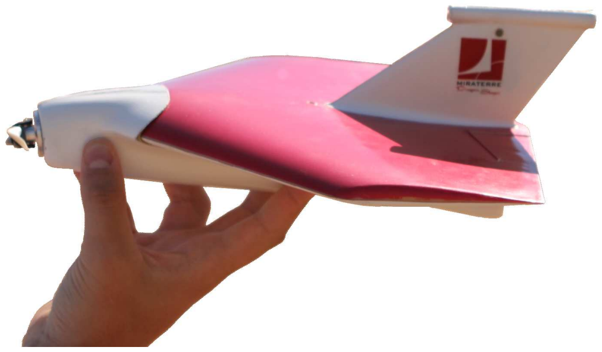


Figure 17: The Dragon Slayer of Miraterre Flight Systems

Miraterre Flight Systems from Tucson, Arizona (miraterre.com). The Dragon Slayer (figure 17) has a wingspan of 33cm and a weight of approximately 300g (including payload). With an endurance of 30mn, it can fly from 18m/s up to 40m/s. Over 500 autonomous flights have been achieved with this airframe designed specifically around the Paparazzi autopilot. A survey trajectory realized by this aircraft is shown in figure 18.

The Dragon Slayer was designed as the first commercial example of a Paparazzi-based surveillance system. All aerodynamic analysis and structural design was done with modern computer tools. Construction consists of a thin Kevlar skin bonded to a core of expanded polypropylene foam (EPP).

The payload includes two 380 line NTSC Color CMOS cameras and an internal paintball deployment mechanism.

3 Payload Deployment

The strategy for a payload deployment at a given location is to fly stabilized on a straight route towards the target. Of course, the precision of the drop will be improved if the aircraft flies upwind. This strategy is easy to implement in Paparazzi, requiring

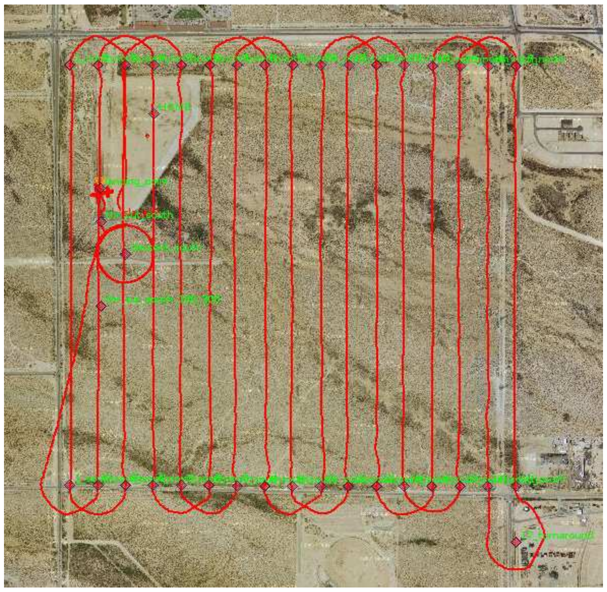


Figure 18: An actual 30mn autonomous survey trajectory done by a Dragon Slayer.

only some basic extra functions to plug into the autopilot code.

3.1 Integration in Paparazzi

A possible Paparazzi flight plan for this mission is the following:

```
<block name="stack">
   <circle radius="75" wp="WAIT"/>
</block>

<block name="approach">
   <call fun="BombComputeApproach()"/>
   <go from="START" to="RELEASE"/>
</block>

<block name="shoot">
  <call fun="BombShoot()"/>
  <deroute block="stack"/>
</block>
```

While the UAV is waiting, circling around waypoint WAIT (stack block), the operator may set the waypoints TARGET and START on

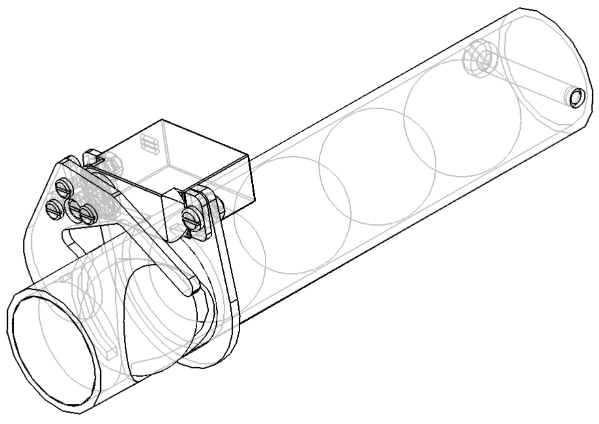


Figure 19: Paintball machine gun.

the GCS. When switching the control to the `approach` block, the location of a `RELEASE` waypoint is computed from the wind and aircraft speeds (`BombComputeApproach()`). Then, the aircraft follows the segment from `START` to `RELEASE`. When this waypoint is reached, the next block (`shoot`) is activated, the hatch is open (`BombShoot()`) and the aircraft is rerouted to the original circle. This flight plan allows the operator at any time to shoot or restart the approach.

To practice the flight plan, we have built a device able to launch several balls per flight, one at a time(figure 19).

3.2 Ballistics

To compute the release waypoint location, we have to study the trajectory of the ball taking into account the drag and the wind.

We first have to look at the Reynolds number of the ball flight to model the air drag.

$$Re = \frac{V\, d\, \rho}{\mu} \tag{1}$$

where V is the fluid velocity, d is the characteristic length of the object, ρ is the fluid density and μ the dynamic fluid viscosity. For a short fall (20 to 30m, around 2s), the expected speed is about $20m/s$. For air, at 25^oC, we have $\rho = 1.2$ and $\mu = 1.8\,10^{-5}$. So we get

$$Re = 23000$$

so the flow around the ball will be turbulent. Then air friction is proportional to the square of the speed, $F = \alpha\, V^2$ with

$$\alpha = \frac{1}{2}\rho\, S\, C_d \tag{2}$$

where S is a surface of the object. For a sphere, for this Reynolds number, the drag coefficient about $C_d = 0.45$.

We are now ready to write the differential equations:

$$\begin{cases} m\ddot{x} = -\alpha\,\dot{x}\,v \\ m\ddot{y} = -\alpha\,\dot{y}\,v \\ m\ddot{z} = -m\,g - \alpha\,\dot{z}\,v \end{cases}$$

where $v = \sqrt{\dot{x} + \dot{y} + \dot{z}}$ and with the initial conditions $x(0) = y(0) = z(0) = 0$, $\dot{x}(0) = x_0$, $\dot{y}(0) = y_0$ and $\dot{z} = 0$.

For a ball of diameter $17.3mm$ and weight $3.31g$, the speed limit of the object, defined by

$$V_0^2 = \frac{g\,m}{\alpha} \tag{3}$$

is about $22.6m/s$.

Some trajectories are plotted in figure 20. We observe that for an airspeed of $20m/s$ and an upwind of $10m/s$, the optimal launch altitude is $30m$: in this case, the precision of the drop will be less sensitive to the altitude error.

Conclusion

During MAV05 (Garmisch, Germany, July 2005), despite some static presentations of

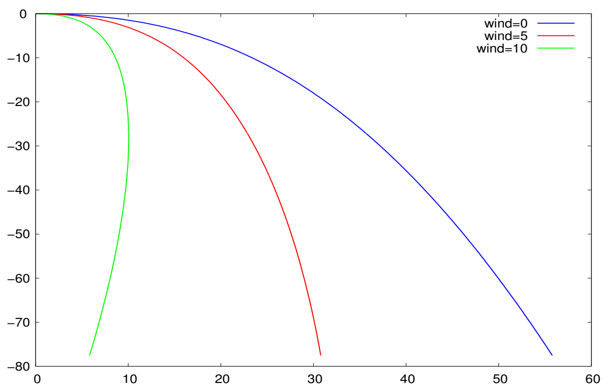


Figure 20: Paintball trajectory for several upwind values. The dilution of precision is smaller with stronger wind.

several autopilots, the only aircraft to fly autonomously were equipped with Paparazzi. During last European MAV competition (Braunschweig, Germany, July 2006), all four participating aircraft were equipped with the Paparazzi autopilot and they all fulfilled the required mission with full autonomy.

Several factors can help to explain these results:

- Most of the efforts of MAV development are done around the vehicle itself, not the complete system which includes the autopilot, the wireless link, the ground control station, etc.
- Developers often try to set up their system directly on the smallest possible airframe. On the contrary, we think than most of the tests of an MAV system can be done with a larger, easy to manage aircraft. Once the architecture is validated, fine tuning on smaller airframes is possible.
- A MAV system is too complex to be a closed black box. We think that the open source approach has been the key to the Paparazzi success.

We are convinced that there is much to be done in the micro air vehicle field from a system point of view, and we hope that Paparazzi will lead the way to creative development. We also hope that our system will serve as starting point for similar projects and that it continues to be improved by the technical community.

References

[1] Nicolas Albert. Certification du code embarqu d'un micro-drone. Master's thesis, University of Toulouse, 2005.

[2] N. Halbwachs, P. Caspi, P. Raymond, and D. Pilaud. The synchronous dataflow programming language LUSTRE. *Proceedings of the IEEE*, 79(9):1305–1320, September 1991.

[3] M. Jenkin and G. Dudek. The paparazzi problem. In *Proc. of the Int. Conf. on Intelligent Robots and Systems (IROS'00)*, 2000.

[4] Eric N. Johnson, Sebastian Fontaine, and Aaron D. Kahn. Minimum complexity uninhabited air vehicle guidance and flight control system. In *AIAA Digital Avionics Conference*, 2001.

[5] F. Nemer, H. Cassé, P. Sainrat, and J.P. Bahsoun. Papabench: a free real-time benchmark. In *Workshop on Worst-Case Execution Time Analysis 2006 (WCET 06)*, Dresden, July 2006.

[6] B. Taylor, C. Bil, and S. Watkins. Horizon sensing attitude stabilisation: A VMC autopilot. In *18th International UAV Systems Conference*, Bristol, 2003.