Multi-UAV control with the Paparazzi system

Pascal Brisset, Gautier Hattenberger

Multi-UAV Control with the Paparazzi System

Pascal Brisset and Gautier Hattenberger

ENAC, Toulouse, France
firstname.lastname@enac.fr
http://paparazzi.enac.fr

Abstract. We present in this paper two experiments involving multi-UAV control. In the first one, three UAVs flew in formation flight, controlled by a single ground control station. In the second one, two aircraft flying in different places in the world have been controlled from the same GCS somewhere else. The two experiments have been conducted with Paparazzi, The Free Autopilot.
Human interactions with the system were complex during these experiments. Controlling three aircraft in the same time in a safe way requires the help of a carefully studied interface. Controlling a UAV from a distant remote place requires to share the control authority between the local and the distant operators.
We believe that the human work for UAVs control and command in this context is more similar to the work of an air-traffic controller than to the work of a pilot. The graphical interface of the Paparazzi system has been developed with this purpose in mind.

Introduction

A digest of the Paparazzi history can be given in a few sentences :

2003 A system is proposed to fly an autonomous MAV (Micro Air Vehicle).
2004 The system is improved to handle several aircraft (i.e. airframe configuration).
2005 The system is augmented to simultaneously control several aircraft.
2006 - 2007 Constant evolutions aim to ease the control of multi-UAVs by a single operator.

Thanks to these effort to provide an ergonomic way of controlling UAVs, several results have been achieved:

- In November 2007, operators from the US army have been trained for the Paparazzi system in two days. Then, they were able to play the role of instructors for other operators who were able to control a fully autonomous flight, from takeoff to landing, after a couple of hours of training.
- In February 2008, after a one week training by Martin Mueller[1], a team from the Norwegian university of Bergen were able to fly an autonomous

[1] Martin Mueller Engineering is a small UAV company which builds, operates and delivers a complete Paparazzi system.

unmanned air vehicle equipped with meteorological sensors over the Spitsbergen island, in the Arctic circle, under very harsh conditions.
- Up to five aircraft have been flown in the same time, controlled by a unique operator on a single standard laptop display.

We present in this paper two other experiments with the Paparazzi system where the operators had to interact in a complex way with a multi-UAVs. In the first one, the operator has to ensure safety while competing with a completely automatic flight planner which control a three aircraft formation flight. In the second one, several operators control several aircraft 1000km from each other.

These two experiments are described after a brief overview of the Paparazzi system.

1 The Paparazzi System

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. Hundreds of hours of autonomous flight have been successfully achieved with the Paparazzi system.

The Paparazzi system is extensively described in [3, 4] and cooperatively documented in a wiki[2].

This section focuses on the distributed architecture, the airborne subsystem, the navigation scripting language and the current available interactions with the navigation kernel.

1.1 Distributed Architecture

In order to offer maximum flexibility and openness, the system was designed from start as a distributed one (figure 1). Ground agents use the `http` protocol for data transfers and the Ivy software bus [5] for middleware. Ivy provides a basic message subscription mechanism and is available for most languages and architectures.

The system was designed to allow simultaneous operation of several vehicles and multiple ground stations.

The airborne vehicles may share a RF network or use a point to point connection to the ground station.

The "link" agent acts as a gateway between the RF side of the world and Ivy. It can speak different protocols and provides retry and acknowledgment services for hardware lacking those functionalities.

The "server" is a central agent that holds the configuration of the aircrafts (flight plan, settings etc) and other data (maps, topographic). It process the raw aircraft telemetry messages, log them for replay or analysis. It can redispatch synthetic asynchronous message to agents like ground control stations.

[2] `www.recherche.enac.fr/paparazzi`

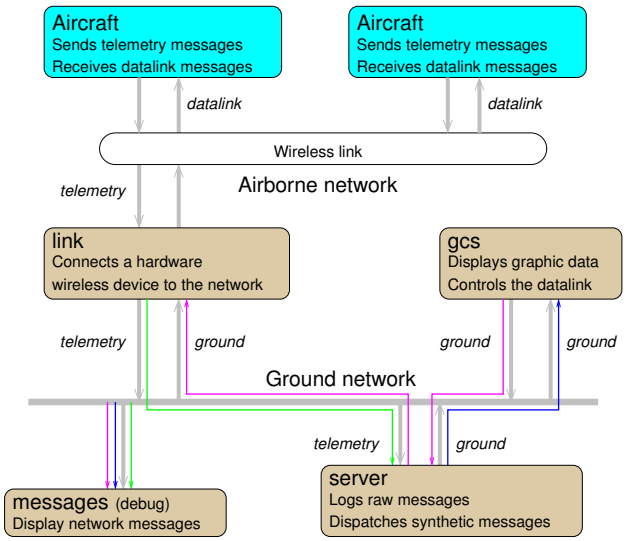

Fig. 1. The agents of the Paparazzi system. Aircraft, link and gcs agents may appear several times.

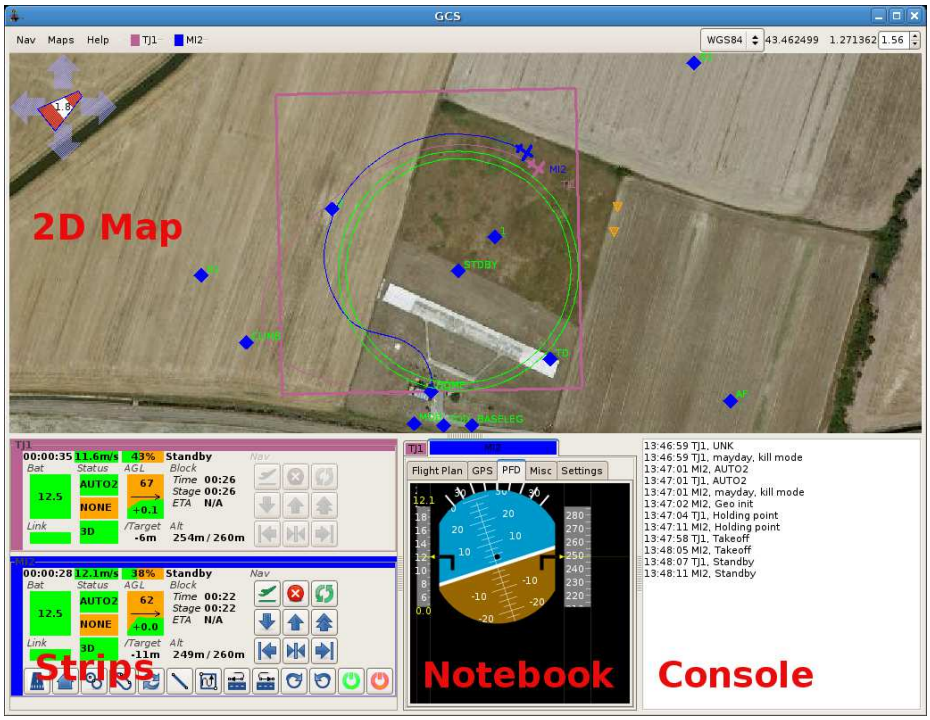

Fig. 2. The current ground control station

The ground control station (GCS) allows for configuration, flight plan edition, simulation, flight control and log analysis. It typically runs in a laptop (figure 2).

"Simulators" are just regular ground agents and can be mixed with real vehicles or replay data.

The "Gaia" agent provides simulated data for environmental factors such as wind conditions and GPS coverage.

1.2 Airborne subsystem

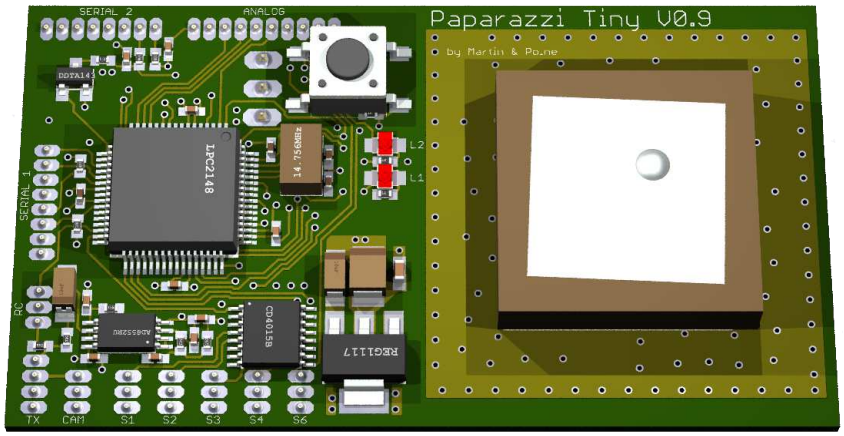

Fig. 3. The *tiny* controller board includes a GPS receiver and an antenna

The figure 3 shows a "Tiny" controller board, which at 20g and 70mm*40mm, is the most integrated hardware supported by Paparazzi.

One of the unique features of the system is the use of infrared thermopiles for attitude sensing (figure 4 shows a typical airborne setup). For the airborne code, focus has been placed on security and reliability with the use of formal methods, code generation and critical code segregation[2].

1.3 Navigation Scripting

The behavior of the vehicle while in autonomous mode is described using a scripting language. This language defines stages and defines blocks as groups of stages. It provides a complete flow control mechanism (jump, conditional, loops, etc.) as well as expressions evaluation and exceptions.

Parameters used in a flight plan can be computed expressions. In the next example, UAV is asked to perform 5 circles at progressively increasing altitudes for exactly one minute at each altitude:

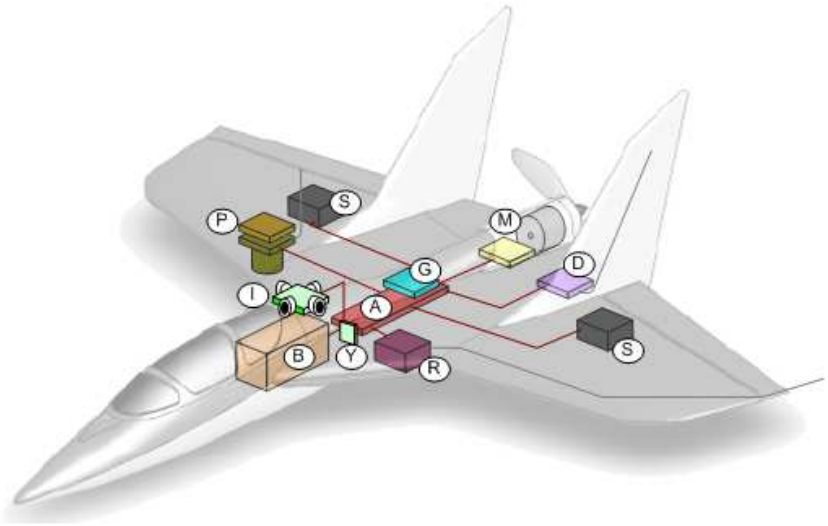

Fig. 4. The Paparazzi airborne system components: (A)utopilot Control Board, (B)attery, (D)atalink Radio-Modem & Antenna, (G)PS Receiver, (I)R Sensors Board, (M)otor & Controller, (R)C Receiver & Antenna, (S)ervos, (P)ayload = Camera & Video Transmitter

```
<for var = "i" from = "1" to = "5">
  <circle wp = "HOME" radius="75"
          alt = "ground_alt+50*$i"
          until = "stage_time>60" />
</for>
```

We provide an API to access the hardware, so that the control of peripherals such as a pan/tilt camera or landing gears/lights is possible from the flight plan.

Complex functions such as landing procedures can be factorized in libraries and called from the flight plan.

Being a Turing-complete language, it guarantees that any problem can be addressed and offers a very high flexibility. The tradeoff is that its complexity makes it poorly suited to real time interactions as we'll see in the next part.

2 Formation Flight Experiments

This section focuses on the management of a set of UAVs flying in formation. Formation flight can improve the safety and the efficiency of the UAVs as it allows mutual support in hostile environments and synchronized arrivals on targets. It is therefore particularly suited for military applications.

The goal of these experiments, conduct at the LAAS-CNRS, was to evaluate an approach that manages the formation to achieve mutual support along a

pre-planned itinerary [6]. The configuration of the formation is autonomously selected as threats are encountered, and the UAVs trajectories that achieve the transition between configurations are dynamically planned and adapted. The algorithms constitute an intermediate layer between the mission planning system and the autopilot of each UAV: the planning layer is therefore released from "internal" formation problems.

After introducing the problem of our experiment, the experimental setup will be presented, and then some improvements will be proposed in the HMI and in the autopilot in order to reduce the operator workload.

2.1 Problem Statement

A planned mission is defined by a list of *waypoints* and a set of *tactical constraints* to satisfy, mainly consisting in inter-UAV distances and protection with respect to threats. Two kinds of threats are considered: early warning radars (EW), whose location is known at the time of mission planning, and missile tracking systems (TF), that are discovered during mission execution. EW can only be jammed by a specific device which is not available on all aircraft, whereas TF threats can be jammed with a device available on all aircraft, that blinds the tracking system in a given direction.

The configuration of the formation is defined by a set of *slots* corresponding to the relative positions of the aircraft. The formation layer is in charge of *(i)* the definition of a configuration according to the constraints and the environment, and *(ii)* the planning of reconfiguration trajectories for safe transitions between two configurations.

The position of the slots hosting the EW jammers is defined in a deterministic way, on the basis of geometrical parameters (the planned formation trajectory and the position of the threats to be jammed), and of an allocation that associate a jammer to a set of threats. To jam TF threats, a potential field approach is used to place the slots and provide mutual support in a reactive way.

Once a configuration is specified, a *Branch&Bound* algorithm is used to allocate the UAVs to the slots according to the constraints. As safe flight is a prime issue, reconfiguration maneuvers are planned to switch between two configurations, considering the inter-UAV security distance. An algorithm relying on A* is proposed for that purpose.

2.2 The Experimental Setup

For those experiments, we had 2 or 3 hobby aircraft (figure. 5) equipped with a Paparazzi board, a Paparazzi GCS, an Ivy bus and an external control architecture where the formation flight controller is implemented. This external system is designed to be embedded and so is fully distributed with local network communication capabilities. As the payload and CPU is very limited on our UAVs, the system is run on a ground laptop with an emulated network.

The control architecture of the UAVs has been developed based on the LAAS architecture [1]. Each aircraft is a functional layer composed of three modules. A

Fig. 5. The three autonomous aircraft used at the LAAS-CNRS.

first module is connected to the Paparazzi communication bus as shown figure 6. It gathers information about an aircraft's state and sends back the commands produced by the other modules (altitude, throttle and roll setpoints). The next module is dedicated to the navigation with a trajectory tracking algorithm and waypoint management. The last module is for the formation control (distributed) and configuration management (centralized). The formation module of each aircraft communicates with the others by an independent bus based on YARP [7]. A Mission Control Station (MCS) is used to send the mission data to the appropriate modules.

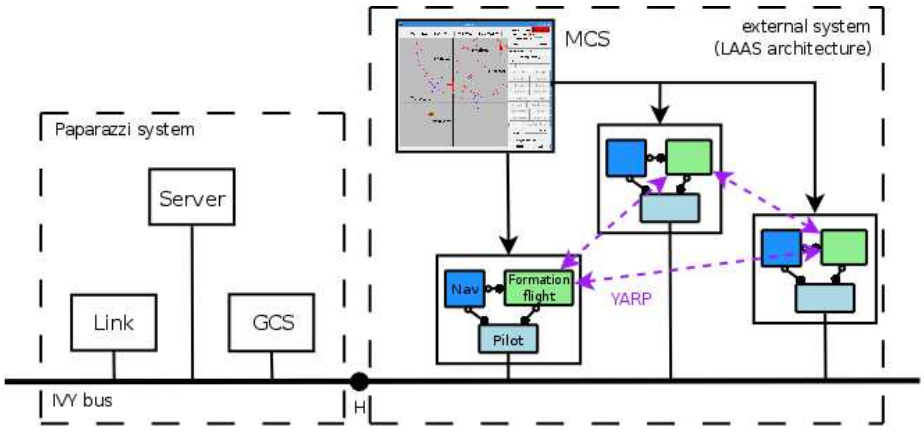

Fig. 6. The ground segment architecture with an external system connected to the Ivy communication bus.

2.3 Achievement of the Experiments

The first steps of our experiments was to evaluate the control of the formation in a given configuration. A typical mission is achieved this way: the UAVs controlled by the Paparazzi GCS are placed on separated waiting circles. When they are ready, the operator sends them on parallel segments as synchronized as possible. In the meantime, the external control is started from the MCS and at the end of the segments the autopilot switch from the Paparazzi flight plan to the external system control.

At the first try, we had two UAVs with safety pilots and one operator with the GCS and MCS on the same laptop. Even with a wide screen, the two GUI cannot be shown on the same desktop, so the operator had to flip between virtual desktops. In those conditions, the workload is quite high and the operator missed to start the gateway that sends commands from the external system to the Ivy bus. As a result, the UAVs go stray to the ground with altitude setpoint to 0. One was taken back by the safety pilot, the other one was too low and land with no harm.

What clearly arise in this situation is that the operation of the aircraft from the GCS should be separated from the conduct of the experiment from the MCS. When the operator realizes on the MCS display that the altitude was wrong, he had to switch to the right desktop and then place the aircraft on their waiting circles. This last actions need at least one click to select the aircraft and a double click to change the current flight plan block. As a result, it took several seconds to place the aircraft in a safe state. The next experiment was conducted by two operators on two separated laptop connected over a network (point H on figure 6). With one operator on GCS, it was safer and easier to manage the UAVs during the initial and final phases. This way, we have been able to perform formation flights with two and three aircraft.

2.4 Reducing Operator Workload

This situation is not fully satisfying. It is pretty difficult for the operator of the GCS to perform the synchronized rendezvous during the initial phase for three aircraft. So, we can't have a better ratio than two operators for three UAVs. In order to improve this ratio, the operator's workload must be reduced. This has been done by improving the following points:

- the current navigation block can be changed faster;
- the UAVs can be automatically synchronized or they can start the mission already in formation flight;
- the UAVs can perform basic collision avoidance.

The first point has been easily solved as the flight plan language of the Paparazzi system allows to set shortcuts buttons to go directly to a given block with one click. Moreover, the layout of the GCS can be configured to the needs of the experiments with multiple-UAVs. Figure 7 shows the strips of each aircraft clearly visible. A vertical view has also been added for this mission. Monitoring

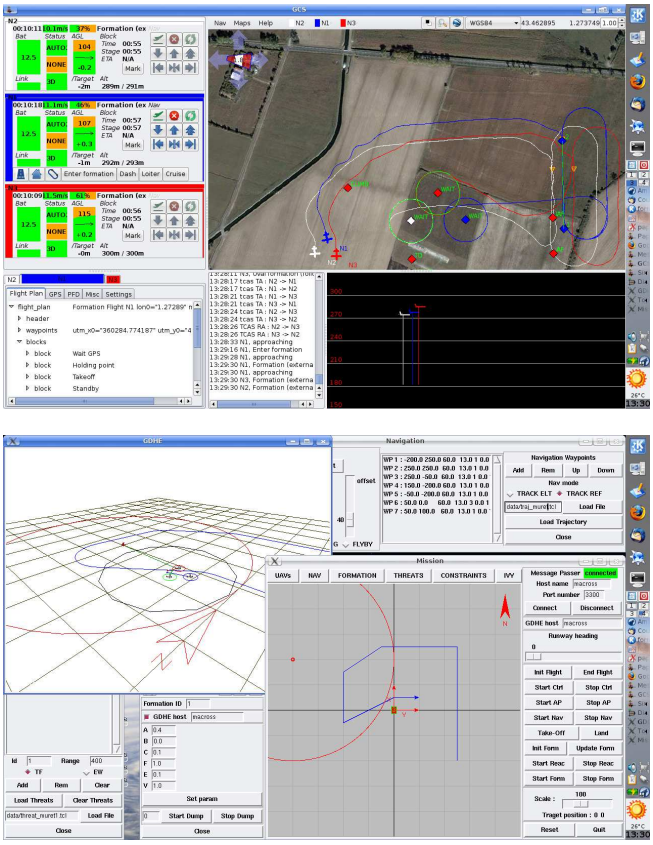

Fig. 7. On the top, the Paparazzi GCS. On the bottom, the Mission Control Center that allows to set and display virtual threats in the environment. Thanks to the Paparazzi HMI flexibility, safety has been improved by providing an efficient display to the safety operator of the mission.

the altitude of aircraft for air traffic controllers is always a problem. This solution is a good help as it shows all the planes in the same plot. However, from what we have seen, the operator can experience some difficulties to use it as the 2D map mainly catches his attention. Visual warning may be a good solution to make it really helpful.

A strong limit of the external system is that he cannot deal with the initial phase, *i.e.* placing the UAVs in formation from any initial situation. To overcome this problem, we have add basic formation flight capabilities directly in the Paparazzi system. One aircraft, declared as the *leader*, follows a standard trajectory (an oval in figure 7, on the right of the GCS). The other two try to fly keeping a given pattern. The main advantage is that the planes are automatically synchronized and in the right configuration. This way, the switch between

the MCS and GCS is fairly easy. Once all the services are started on the MCS, the operator just need to click on a button on the *leader* strip and all UAVs switch their control at the same time. At the end of the mission, they can be placed again in formation or on waiting circles before landing.

The last point is a common problem to all multi-UAVs situation. How can we guarantee that an aircraft will not collide with an other? It is to much workload for an operator to ensure the lateral and vertical separation between UAVs if they fly in a reduced airspace and at the same level. Our solution is to implement a basic collision avoidance system based on the TCAS system used on civil aircraft. This system ensures a minimal vertical separation between two UAVs if they are in autonomous navigation. This way, safety pilots do not need to interrupt the experiment.

All these improvements allows a single operator to deal safely with two or three UAVs, even from two Control Stations. If the number of aircraft were to increase, two operators would be necessary, but the overall ratio between planes and operators would be better. This work is a start to prospect other situations of control shared between several GCS for a large number of UAVs, or for an air traffic control on top of the GCS.

3 Remote Control over the Internet

An experiment has been conducted in December 2007 with aircraft flying in Toulouse (France), Hildesheim (Germany), both of them controlled from the conference room of the 24C3 conference[3] in Berlin (Germany)[8]. Live video from the two aircraft were also pushed to a server providing real time streaming to any client around the world. While a standard DSL link was used in Toulouse, datalink and live video used a GSM connection in Hildesheim.

3.1 Architecture

The architecture of the experiment is displayed in figure 8. The different elements connected to the Internet network were :

- Two standard Paparazzi systems were set up in Hildesheim and in Toulouse, both involving
  - A fully autonomous UAV connected to the ground station through two wireless connection, one for control and command and another one for video;
  - A ground station organized around an Ivy Ethernet network;
  - A graphical user interface and its operator.
- A graphical user interface located in a conference room in Berlin.
- A server, located somewhere in Germany, responsible for relaying messages between the flight locations and the control interface in Berlin. This relay was convenient

[3] http://events.ccc.de

1. to stream video images which were uploaded from Hildesheim and Toulouse;
2. to handle the datalink with the Hildesheim setup because the GSM connection was not symmetric and was not able to receive packets from any host, but only to receive data through an established connection.

When the UAV operators were ready on the flight fields, the operator in Berlin just clicked on the graphical interface to start the motor; the local operators just launched the UAVs. The operator in Berlin had a full control on the two missions, to change the flight altitude, change the navigation pattern ... or switch on and off the airborne video transmitter.

3.2 Human Interactions

In this experiment, the authority on UAV control was shared between the local operator and the remote operator. No priority had been defined between the two except that the local operator always had the option to stop (with one mouse click) the Ivy2TCP agent responsible of the connection between the local Paparazzi network and the Internet.

The possibility of conflict between the two operators never occurred during the experiment or during the preparation.

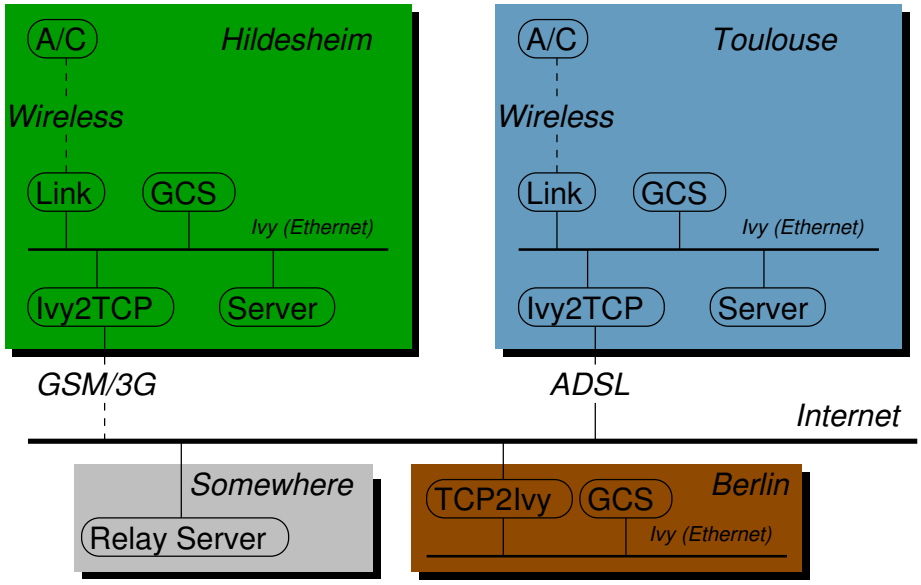

Fig. 8. Multi-UAVs control over a worldwide distributed network. Authority is shared between the local operators (in Toulouse and Hildesheim) and the remote operator (in Berlin). Note that the Relay Server allows several remote agents to interact with the UAVs.

Conclusion

We have presented two experiments realized with the Paparazzi system. These experiments involve multi-UAVs control by several operators in the same time. These results demonstrate that the design of Paparazzi provides the right architecture for distributed agents, vehicles or/and operators.

The current Paparazzi graphical user interface, after years of experience and improvement, has been proved to be flexible, configurable enough, to be customized thanks to the operator needs. Hours of flight show that the system must be autonomous enough to give time to the operator to react to unexpected events; the UAV or MAV must autonomously stay in the air as long as possible while audible or visible alarms are shown to the operator.

Next step is to precisely handle authority sharing between different operators.

References

1. Rachid Alami, Raja Chatila, Sara Fleury, Malik Ghallab, and Felix Ingrand. An architecture for autonomy. *International Journal of Robotics Research*, 17(4):315–337, April 1998.
2. Nicolas Albert. Certification du code embarqué d'un micro-drone. Master's thesis, University of Toulouse, 2005.
3. P. Brisset and A. Drouin. PaparaDzIY: do-it-yourself UAV. In *Journées Micro Drones*, Toulouse, France, September 2004.
4. P. Brisset, A. Drouin, M. Gorraz, P.-S. Huard, and J. Tyler. The Paparazzi solution. In *MAV2006*, Sandestin, Florida, November 2006.
5. M. Buisson, A. Bustico, S. Chatty, F-R. Colin, Y. Jestin, S. Maury, Ch. Mertz, and Philippe Truillet. Ivy : Un bus logiciel au service du développement de prototypes de systèmes interactifs . In *IHM 2002, Poitiers*, pages 223–226, http://www.acm.org/, novembre 2002. ACM Press.
6. Gautier Hattenberger, Simon Lacroix, and Rachid Alami. Formation flight: Evaluation of autonomous configuration control algorithms. In *Proceedings of the International Conference on Intelligent Robots and Systems*, 2007.
7. G. Metta, P. Fitzpatrick, and L. Natale. YARP: yet another robot platform. *International Journal on Advanced Robotics Systems, Special Issue on Software Development and Integration in Robotics*, 3(1):043–048, 2006.
8. Martin Mueller and Antoine Drouin. Paparazzi - the free autopilot. build your own uav. In *24th Chaos Communication Congress*, December 2007.