Flexible Open Architecture for UASs Integration into the Airspace: Paparazzi Autopilot System

Elgiz Baskaya*, Guido Manfredi*, Murat Bronz†‡ and Daniel Delahaye‡
*ENGIE Ineo - Safran RPAS Chair, ENAC, Toulouse, France
Email: elgiz.baskaya@enac.fr
†UAS Lab, ENAC, Toulouse, France
‡Laboratoire MAIAA, ENAC, Toulouse, France

***Abstract*—Air safety authorities are forced to develop regulations for UAS due to incidents disturbing public safety and demands from UAS operators. Despite numerous studies from the FAA and EASA, none of them decided on a regulation for UASs. The reliability of the flight is considered to be one of the main obstacles for UAVs integration. This is not an easy topic considering the unknowns of the systems, environment and possible failures. We believe the flexibility required for such solutions calls for open architectures. More specifically, this paper shows how the use of the *Paparazzi* open source auto-pilot system can ease the integration of low altitude UAS. To ensure safety, this integration needs to be achieved through airspace management and UAS reliability.**

Preliminary airspace designs, e.g. Amazon's, identify different zones depending on the UAS capabilities, population density and altitude. Plus, national rules evolution push to cope with a variety of requirements. Open source and modular architectures are key to adapt to these requirements. From a UTM point of view, *Paparazzi* provide features to ease congestion management, such as dynamic geofencing, trajectory communication and collision avoidance. Concerning reliability, current regulations focus on flight constraints but might be expected to involve regulations on software and hardware components as well. In such case, the increased cost will be inevitable for the demands of certification. In the *Paparazzi* software case, parts of the code have been formally proved and stable versions have thousands of flight hours. Such heritage might ease the certification process for smaller companies.

On top of its flexibility and reliability, *Paparazzi* offers a unique set of features, as an open source software, to achieve safe integration of low altitude UAS in the G airspace. To conclude this work, desirable new features and future work are discussed.

I. Introduction

The cost effectiveness and reachability of commercial off-the-shelf elements, and shrinking size of electronics serve as a perfect environment for small flying vehicles to emerge. Although inherited as military purposes in its infancy, nowadays Unmanned Aircraft System (UAS) are becoming efficient platforms for scientific/commercial domains. They offer benefits in terms of cost, flexibility, endurance as well as realizing missions that would be impossible with a human onboard. Increasing usage of these vehicles for a variety of missions, such as defense, civilian tasks including transportation, communication, agriculture, disaster mitigation applications pushes demand on the airspace. Furthermore, this congestion is predicted to accelerate with the growing diversity of these systems.

Commercial advantages, offered by these efficient systems, are already targeted by big companies worldwide, specifically in the US. The airspace regulatory authorities seem to be squeezed in between the companies, demanding a fast as possible access to airspace, and the concerns of the public about potential privacy breaches, safety and liability issues [1], [2]. Even with today's strictly regulated airspace, reported occurrences show that there are hurdles to solve before a further integration of UAS to airspace.

Advent of the new era of UAS seems to be hold by an unseen barrier of lack of regulatory framework for now. Different institutions all over the world, specifically National Aeronautics and Space Administration (NASA) and Federal Aviation Administration (FAA) in US, easa in Europe and international bases such as International Civil Aviation Organization (ICAO) are addressing safe integration of UAS in airspace. Although the approaches of regulatory bodies may vary, the aim remains the same: safe integration as soon as possible.

The tackles of safety during integration of UAS to airspace refer to different technical and organizational aspects including but not limited to control of traffic in segregated and non-segregated airspace, reliable communication, robust control of the Unmanned Air Vehicle (UAV), trajectory planning, detect&avoid.

In this study, we present the *Paparazzi* open source auto-pilot system, and its features, as a tool to ease the safe integration of low altitude UAS into National Air Space (NAS). In our argument, *Paparazzi* is favored thanks to its modular design, its support for congestion management and evolving reliability.

II. Open-source Autopilots for UAS

With the new era of First Person View (FPV) flights, especially for multi-rotor UAVs, there has been an exponential increment on the hardware and software of the open-source autopilots. A brief comparison of the popular current open-source and commercial autopilots is available in Table I. Usually the trend is to make the hardware as cheap as possible for recreational consumers. This reality is damaging the reputation of open-source autopilots as being not reliable or robust, just because they are being used without attention. There exists a difference on the quality of sensors used on the open-source autopilot systems compared to commercial autopilots for sure, and this is extremely represented on the price of the units. This makes a difference on the flight quality of the vehicles, however with the new on-board processing power, more complex estimation algorithms and filters are being used in order to overcome this problem.

III. The PaparazziUAV project

The *Paparazzi* Autopilot System aims to provide a software and hardware solution for low-cost mini and micro unmanned air vehicles. It is started as a personal project in 2003 by Pascal Brisset and Antoine Drouin, and afterwards supported by ENAC in 2005. Being one of the first (if not the first) open-source autopilot system in the world, *Paparazzi* has attracted attention and led the others to start new branches and systems. The software is originally packaged for Debian/Ubuntu but can be manually installed on any GNU/Linux operating system even including MacOS-X. However, it is not compatible with Windows which automatically eliminates 90 % of the possible user community and therefore it is not as popular as other existing autopilot systems on the user market.

A. System Architecture

The system architecture of *Paparazzi* consists of three segments: Ground, Airborne, and the communication link between (Figure 1). The ground software of *Paparazzi* is mainly written in Ocaml, with some additional parts in Python and C. Thanks to its middle-ware communication bridge called Ivy-Bus, external softwares can be directly connected with publish and subscribe method to the ground segment without needing to modify and code at all. Airborne software is completely written in C, however there is an on-going work which implements the possibility of adding some C++ code parts.

B. Distinguishing Features

One of the distinguishing property of the *Paparazzi* is to support multi-UAV flight (Figure 2). Several projects have been using *Paparazzi* Autopilot System because of this additional feature. The communication of each vehicle is being transmitted to the ground control station directly on the current version, however air to air communication between flying vehicles and internal relay of information to ground control station is being implemented for a new coming version.

Modules are the easiest and most flexible way of adding new code into *Paparazzi*. There are over 130 modules written for *Paparazzi* by several developers and researchers including subjects on meteorology, imagery, surveillance, advance navigation, formation flight and collision avoidance, etc...

Paparazzi has its own complete flight plan language, where the user can define any possible trajectory by existing commands such as circle, line, hippodrome, figure-eight, survey, etc... Additionally any function, written in C language, can be called from the flight plan and executed. This opens up a

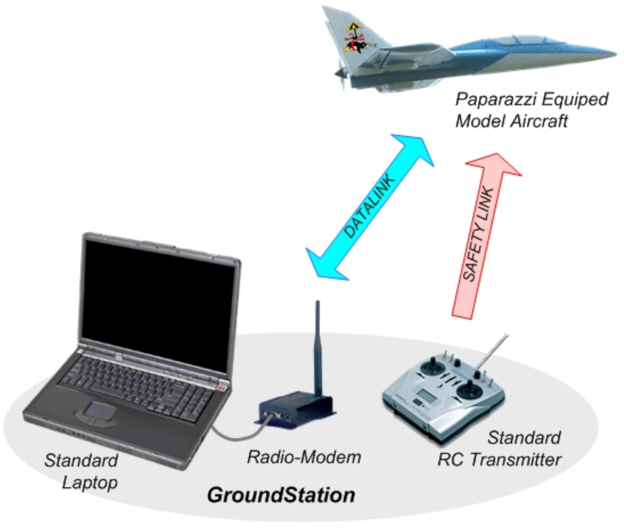

Fig. 1. Paparazzi autopilot system overview.

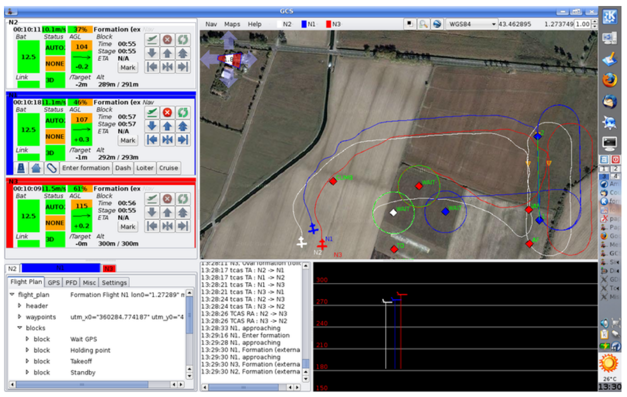
```
13:28:11 N3, Oval formation (follo
13:28:17 tcas TA : N2 -> N1
13:28:17 tcas TA : N1 -> N2
13:28:21 tcas TA : N1 -> N3
13:28:21 tcas TA : N3 -> N1
13:28:24 tcas TA : N2 -> N3
13:28:24 tcas TA : N3 -> N2
13:28:26 TCAS RA : N2 -> N3
13:28:26 TCAS RA : N3 -> N2
13:28:33 N1, approaching
13:29:16 N1, Enter formation
13:29:28 N1, approaching
13:29:30 N1, Formation (externa
13:29:30 N1, approaching
13:29:30 N3, Formation (externa
13:29:30 N2, Formation (externa
```

Fig. 2. Ground Control Station showing trajectories of 2 UAVs using the TCAS system

lot of application possibilities such as triggering a navigation procedure via a sensor output.

A real-time operating system based on ChibiOS is started to be used since 2015. The first implementation was mainly for adding a separate thread in order to make on-board logging at a high frequency. On going work is to divide all autopilot tasks into individual threads and manage everything according to priorities which will increase the safety and reliability.

C. Hardware

There exists over 20 different autopilot boards capable of running *Paparazzi*. Additionally, several mission based custom sensor boards have been designed under the *Paparazzi* project, such as *Meteo-Stick*[1].
The rest of this work highlights *Paparazzi*'s capacity to tackle the issues encountered during integration of UASs in the airspace. These issues have been grouped in three categories: modularity, congestion management and reliability. For each

[1]http://wiki.paparazziuav.org/wiki/MeteoStick_v1.10

TABLE I
COMPARISON OF POPULAR OPEN-SOURCE AND COMMERCIAL AUTOPILOTS.

Open-Source	Vehicle type	Hardware	Multi UAV	Flight Plan	Path Definition	Geofencing	Collision Avoidance	Latest stable release
Paparazzi	Fully configurable	varied	yes	scriptable	Formula Based	3-D	UAV TCAS	21-06-16
PixHawk	Fully configurable	specific	yes	scriptable	Circles Lines Patterns	3-D	none	06-08-16
Ardupilot	Copter Plane Rover	varied	alpha	scriptable	Circles Lines Patterns	2-D	none	22-06-16
OpenPilot	multicopter	specific	none	NA	NA	none	none	15-05-15
AeroQuad	multicopter	varied	none	NA	NA	none	none	31-01-13
Commercial								
Picollo	Copter Plane	specific	none	dynamic	Way Point	NC	NC	NC
MicroPilot	Copter Plane Rover Blimp	specific	yes	dynamic	Way Point	NC	NC	NC

category, *Paparazzi* offers a unique set of features to deal with the issues at hand.

D. Security

Latest micro processors used in the *Paparazzi* hardware has an additional feature that can be use for security augmentation. The cryptographic processor can be used to both encipher and decipher data using the DES, Triple-DES or AES (128, 192, or 256) algorithms. It is a fully compliant implementation of the following standards:

- The data encryption standard (DES) and Triple-DES (TDES) as defined by Federal Information Processing Standards Publication (FIPS PUB 46-3, 1999 October 25). It follows the American National Standards Institute (ANSI) $X9.52$ standard.
- The advanced encryption standard (AES) as defined by Federal Information Processing Standards Publication (FIPS PUB 197, 2001 November 26)

Since, with *Paparazzi*, the uav needs to be flashed after each modification, it is easy to rely on the random generator of the operating system (/dev/random under linux) to generate a key that will be flashed alongside with firmware, so this key will be :

- shared between UAV and groundstation
- regenerated for each flight, this make hostile decypher difficult
- in a multi UAV configuration, each UAV will have private cypher key

For the moment, the crypto driver for chibiOS is being written in order to be implemented into *Paparazzi* soon. This is generally a complicated task, however as the datalink bandwith is low, it becomes easier compared to a device driver.

E. Regular Use

Highly flexible architecture of the *Paparazzi* systems comes with the disadvantage of being too complex for a regular user such as a recreational (model aircraft) pilot. From the developer's point of view this is a great property, however beginners and regular users do not need to use all of the new features right away so they do not need to maintain the software at the latest version. Therefore the regular user will be using a recent stable release of the system, and usually will keep on using it without modification till he needs the new additional features coming from the developer community.

On the other hand, the developers will be able to use the software versioning system with GIT, maintained at Git-Hub, and can check out and contribute to the latest software version in the beta test status. The tools that the developers need in order to do that are not too different than the ones that are required for other software projects.

IV. MODULARITY

The current evolving nature of regulations and the variety of organizations in charge of the airspace rule making calls for flexible solutions to cope with these fruitful environments. *Paparazzi*, as an open source autopilot system, is largely modifiable through its modules to offer such flexible solutions.

A. Airspace Categorization

The UAS in the NAS project points to a performance-based routine access to all segments of the national airspace for all unmanned aircraft system classes, after the safety and technical issues are addressed thoroughly. As a start, NASA and faa seem to have a short term goal to integrate UAS in low-altitude airspace as early as possible. They further aim to accommodate increased demand safely, efficiently in the long term. NASA and faa seem to handle the airspace above 500 feet and the one below separately. European Aviation Safety Agency (EASA), tasked by the European Union, is planning a risk based approach, accommodating the UAS in the airspace under three different categories, low risk, specific and high risk [3]. Both regulators seem to categorize the airspace and scale regulatory needs according to some criteria. To answer different needs of different categories, flexibility given by the high level of modularity of open source autopilot systems will be a handy tool. Customizability of the software and hardware depending on the airspace gives a chance for larger airspace community to utilize *Paparazzi* scaled for their specific needs. A larger community using the same system would lead to a natural evolution of the system toward a better design.

B. *National Regulations*

Circulation of UAS internationally is somewhat prevented by the Chicago convention unless an agreement holds between Contracting States [3]. ICAO is aiming to develop international standards and recommended practices to which the member states could refer to when developing their national civil aviation regulations. Even though a similar base is aimed, national aviation legislations will not be the same because of the different expectations of nations about UAS aviation. Thanks to the modular nature of the *Paparazzi* software and hardware suite, the functionality of the system could be enhanced according to the specific regulations held in the area of utilization.

C. *Accommodating evolution of regulations*

Prescriptive rules seem to cause some difficulties since the technical area on UAS systems develop rapidly [3]. Innovations both on the aircraft and the operation type of UAS will accelerate especially after the regulations are set. Thus, regulatory bodies call for refinable operational requirements and system architectures to evolve into a safer and efficient integration of UAS into airspace. The systems to cope with the regulations should also be modular and flexible in order not to be superseded by the innovations in the area. Thus, the aviation regulatory bodies aim to achieve designs with flexibility where possible, structure where needed. Having flexible hardware and software points to modularity, which is pretty much best supported via open source systems.

V. Congestion Management

According to *UAV Factory*, one of the large European UAS companies, "The future of the UAV industry is likely to be shaped by airspace congestion" [4]. Indeed, high level airspaces are getting crowded and large scale solutions, such as NextGen (US) or SESAR-JU (EU), are necessary to increase airspace capacity while maintaining the current safety levels. However, there is no such management solution existing for Very Low Level (VLL). Yet, large projects like Amazon's Prime Air and Google's Project Wing are already waiting to populate the VLL airspace.

Part of the congestion management problem is to avoid conflicts, and more importantly collisions, between UAS through strategic deconfliction and safety nets. Another mission of the congestion management system is to make sure that UAS do not go where they are not supposed to go, thus requiring geofencing. In order to implement the previously mentioned systems, the UAS autopilot needs to be able to perform complex operations, e.g. static waypoints following is likely to be insufficient. In the following, we divide these issues into four topics of interest: 4-D trajectory management, geofencing, safety nets, complex operations, and show how the *Paparazzi* system addresses them.

A. *4-D Trajectory Management*

As noted in [5], 4-D trajectories will be central in future airspace management methods. The principle of 4-D trajectory management is to have every UAS broadcast its trajectory up to some time horizon and receive Unmanned Aircraft System Traffic Management (UTM)'s clearances under the form of trajectories. The trajectory information contains a path, the series of points through which the UAS will pass, and times associated to each of these points. Thanks to this information, the idea is to perform pro-active deconfliction, as explained by Thomas et al. in [6]. In clear, it implies that UTM detects future conflicts along the trajectories of all UAS and deconflicting them as safely and early as possible. This kind of approach requires the autopilot to represent UAS motions with trajectories and to be able to transmit them, which is the case in *Paparazzi*.

Paparazzi originally supports a basic description of trajectories based on circles and straight lines, but recent updates allow it to process advanced trajectories as well. Indeed, *Paparazzi* can represent trajectories as functions in 2-D (x, y), 3-D (x, y, z) or 4-D (x, y, z, t). The only requirement is for the function to be differentiable at least two times on every point. The function and its derivatives analytical formulas are computed offline and can then be quickly evaluated online. Moreover, gains can be tuned to adjust the convergence speed from the UAS starting point to the given trajectory. These gains influence the convergence path to a given trajectory and the resulting trajectory (convergence + commanded) can be computed offline given initial conditions and the commanded trajectory. This is of particular interest for UTM as it allows the UAS to provide not only its trajectory over time but also how it will reach this trajectory from its starting point.

It is important to note that UTM will have to solve conflicts between manned and unmanned aircraft. So an air traffic controller is needed in the loop with an appropriate display for 4-D trajectories and a communication link with both manned and unmanned aircraft.

B. *Safety Nets*

Trajectory deconfliction is the first step to manage congestion, however safety nets are also part of the congestion management. Indeed, safety nets such as self-separation and collision avoidance allow UAS to fly close to each other while preserving an acceptable safety level. Though *Paparazzi* does not include self-separation algorithms, it do contains a light version of the TCAS II collision avoidance system. It considers intruders one at a time and is capable of coordinated avoidance maneuvers. There is no strengthen resolution as the resolutions are not speed rates but an objective altitude to reach as fast as possible.

TABLE II
TYPICAL VALUES FOR MAIN VARIABLES OF THE *Paparazzi* TCAS MODULE.

TAU_TA	TAU_RA	DMOD	ALIM
10s	6s	15m	10m

Values such as distance thresholds have been adapted to suite the performances of small UAS. Table II shows example values used for fixed-wing UASs. Keep in mind that the *Paparazzi* philosophy is to be configurable, so these parameters can be easily changed from a configuration file.

On top of the TA and RA provided by the TCAS, a module has been recently developed to input data from traffic services and display them into *Paparazzi*'s ground control station, thus providing situational awareness to the remote pilot. Preliminary test have been done using opensky-network to display traffic around a given area, based on these information the remote pilot can effectively perform conflict resolution.

C. Geofencing

Keeping UAS away from each other is an important point. But keeping them out of forbidden areas is also crucial. Geofencing allows determining no-fly zones where the UAS should not enter. To accommodate land owners while managing traffic and limiting congestion, Foina et al. [7] proposed a participative dynamical airspace management method: the air-parcel model. It allows land owners to authorize/forbid flights over their lands through a web interface. However, this type of approach asks from the UASs to be able to handle dynamical geofencing. Plus, though initially this model considers only cuboid parcels, the need for more precise airspace shapes may emerge making 3-D geofencing a need.

This type of model can be handled by *Paparazzi* by defining geofenced zones manually or programmatically. The zone is defined as a simple geometrical shape (e.g. a circle, a polygon, etc.) plus altitude limits thus effectively creating a 3-D geofencing. These zones can be static, activated dynamically from the ground station or from the flight plan with associated conditions.

D. Complex Operations

Having 4-D trajectory management, safety nets and geofencing is useless if the UASs cannot follow the instruction from UTM regarding these tools. Indeed, new UTM paradigms imply being able to change flight plans dynamically to answer to UTM demands. In [8] two types of complex operations examples are mentioned: space transition corridors and temporary flight restriction. Both these airspace management methods require from the UAS to be able to modify its flight plan according to new UTM instructions.

Modifying the flight plan dynamically is not possible, and not desirable, in *Paparazzi*. However, appropriate response to these complex operations can be provided through conditional flight plans. These enable the UAS to follow different courses of action depending on given parameters. These parameters are defined offline by the user and can then be modified online by values coming from sensors or from UTM. This allows intelligent reaction to exterior instruction, in particular it can be used by the user to give UTM control on portions of the flight.

VI. RELIABILITY

Improvement of the reliability of the flight is considered to be one of the main goals for integrating military UAVs into civil airspace according to Unmanned systems roadmap by US Office of the Secretary of Defense, DoD [9]. Compared to manned counterparts, UAVs experienced failures with a frequency of two orders of magnitude more in the military domain. Although this changed last years with the technological improvements, making the UAVs as reliable as early manned military aircraft, it seems not enough from the DoD perspective. This can be realized by checking the biggest chunk of control technologies budget for research and development, which is health management and adaptive control.

To achieve a safe flight is not an easy task considering the unknowns of the systems hardware, environment and possible system faults and failures to emerge. Also, increasing demand on cost effective systems, resulting in smaller sensors and actuators with less accuracy, impose the software to achieve even more. The expectation that UAVs should be less expensive than their manned counterparts might have a hit on reliability of the systems. Cost saving measures other than the need to support a pilot/crew onboard or decrement in size would probably lead to decrease in system reliability.

A. Small and Medium Enterprise (SME)s and Certification Costs

Utilizing drones for quicker and cheaper deliveries could be rewarding for SMEs since cost per mile of a drone is less then 1 / 30 of the average diesel truck. Being an early bird might put the SMEs in an advantageous position, considering the increase in the capabilities of the drones with inevitable acceleration thrusted by research activities and their widened application areas. Nevertheless, the fairly cheap access to drones and their relatively cheap utilization cost does not seem to be enough to put them to air right now due to the heavy cost of certification and regulatory hurdles [10]. In this concern, capable open source solutions could be a good way to loosen the tie. Otherwise, SMEs, an important factor in drone business might not survive. Even worse, they might operate them without relevant permission, scarifying a substantial fine as reported by Civil Aviation Authority (CAA). This will compromise security in the system contradicting the hopes for reliable integration of drones into airspace. As an open source autopilot system, *Paparazzi* is a known platform for computer scientist who want to test the viability of complex software systems. Thus parts of *Paparazzi* code (Ground Control Station (GCS)) have been formally proven [11] and stable version have thousands of flight heritage. Furthermore, although Paparazzi is not certified, to be able to have access to the certified airspace, the users can built up their configuration on readily and freely available Paparazzi system, which might reduce the cost to have a certified system to reach specific airspaces.

B. Individuals and Education

Individuals, as well as SMEs, suffer from the same budget constraints. Personal UAV usage counts for a substantial amount of the drone ecosystem. Both US and European authorities mention the importance of individuals in utilizations of drones. There is a community with a passion of aviation and potential, but most probably not very experienced. *Paparazzi* offers a whole community to help/educate these beginners through its forums enriched by rising number of its users. A rich documentation is available through the Wikipedia page, encouraging users to self-teach.

C. Flight Heritage for Risk Assessment

Drone industry being extremely innovative, technical developments could supersede the prescriptive rules as regulations. Thus a solution might be to follow a risk based approach rather that to have strict rules to cope with. Predicted regulations in Europe seems to evolve under different categories dedicated to specific operation risks. Flight heritage and occurrence reporting is expected to be an inevitable part of safety risk assessment to achieve reliable flight. Wide utilization of *Paparazzi* and real time connection to ethernet could offer reliable and practical solutions to report occurrences.

D. Support for real time planning and onboard vehicle automation

To access low-altitude airspace with the use of small unmanned aircraft safely, an important ability could be to implement real-time planning and on-board vehicle automation. Amazon offers that this approach will allow some flexibility to adapt to variable situations such as weather changes, severe winds or any other emergency needs. *Paparazzi*, has a real-time planning ability already implemented, letting the user change the trajectory in real-time through its ground control station via different strategies. The user could switch between trajectories already available or even drag the waypoints to his/her preference. The automation of the vehicles is handled in different stages. For now, the most used modes of autonomy is no autonomy, assisted mode and the fully autonomous mode. No autonomy mode, or manual mode, gives whole responsibility of the aircraft dynamics to the pilot through the RC link. The assisted mode closes some attitude control loops, giving some stability to aircraft. This will ease the challenges of piloting which could be a great advantage especially for the unexperienced pilots. The fully autonomous mode handles both heading and navigation through selected trajectories. In case of an emergency, the pilot take the control of the aircraft anytime, through the RC link. Features already implemented on *Paparazzi*, such as geofencing, go home, and its ease in adding/modifying thresholds to various variables, support safety procedures.

VII. Future Evolutions

As many open source projects *Paparazzi* is in constant evolution through regular contributions. With its large community, it is hard to keep track of all the ongoing projects. In this section we present three features being currently explored and developed at ENAC's labs.

Though there is no life a stake, avoiding UAS collisions is desirable for safety and operational reasons. *Paparazzi*'s current TCAS-like collision avoidance allows operating numerous UAVs with little risk of collision between them. However, future missions will include more and more UAVs in bounded size airspaces. In this context, an efficient collision avoidance algorithm is desirable to allow closer operations with an acceptable amount of nuisance alerts. Because it relies on advanced logics and can be adapted to different types of performances, we have chosen to implement the ACAS Xu algorithm. Though this standard's definition just started, the baseline is already determined and may allow developing a simplified version for micro UAVs.

Regarding geofencing, most current methods use boundaries in the horizontal plane in which the UAV cannot enter. However, the utilization of the VLL airspace is likely to demand more sophisticated methods to handle 3-D geofencing where a 3-D no-fly shape can be described. Though *Paparazzi* can emulate 3-D geofencing, it cannot use full 3-D models yet. Work is underway to extend the current geofencing to allow loading terrain models and padding them with arbitrary shaped no-fly zones.

Finally, conventional procedures to handle failures onboard could be one of three approaches in order to achieve safe flight. First one is the fail operational systems which are made insensitive to any single point component failure. The second approach is the failsafe systems where a controlled shut down to a safe state is practiced whenever a critical fault is pointed out by a sensor. The third approach is fault tolerant control systems in which components of the system are monitored and action taken in whenever needed. The strategy is most probably to try to keep the system availability and accept reduced performance. For *Paparazzi* autopilot system, there is ongoing research to implement indigenous fault detection and diagnosis module to enable reconfigurable controller loops.

VIII. Conclusion

The introduction of UASs in the VLL airspace comes with numerous challenges. Though various ways to address them have been proposed, there is still no certainty about how it will be done in the end. One sure thing is that it will involve two main actors: UTM and UASs. This work focused on the later.

When it comes to autopilot systems, there exist a wide range of solutions. However, in the context of a quickly evolving regulation and technology, we showed that it may be beneficial to rely on an open source solution. Indeed, the community around it and the ability to adapt it to one's needs are precious assets. Among open source solutions, the *Paparazzi* autopilot system distinguishes itself with a unique set of features that, we have showed, allow tackling many of the existing challenges in the UASs integration. The most interesting feature being its linux-like philosophy which confers it an unparalleled level of configurability. With a complete control, the user is not limited by the software and nothing stops him from doing the smartest designs.

Paparazzi is a living project and keeps evolving to satisfy it community, both from the research and industry. New features are added on a regular basis and the shape it will take in the future only depends on the new members that join the project.

Acknowledgment

This work was supported by the ENGIE Ineo - Sagem RPAS Chair. The authors would like to thank Gautier Hattenberger and Hector Garcia de Marina for sharing their expertise in *Paparazzi*.

References

[1] C. Forrest. (2015) 12 drone disasters that show why the faa hates drones. [Online]. Available: http://www.techrepublic.com/article/12-drone-disasters-that-show-why-the-faa-hates-drones

[2] U. E. Franke. (2015) Civilian drones: Fixing an image problem? [Online]. Available: http://isnblog.ethz.ch/security/civilian-drones-fixing-an-image-problem

[3] "Introduction of a regulatory framework for the operation of drones," Advance Notice of Proposed Amendment 2015-10, Office of the Secretary, 2005.

[4] L. Probst, L. Frideres, B. Pedersen, and C. Martinez-Diaz. (2015) Uav systems for civilian applications. [Online]. Available: http://ec.europa.eu/DocsRoom/documents/13430

[5] H. Erzberger and R. A. Paielli, "Concept for next generation air traffic control system." *Air Traffic Control Quarterly*, vol. 10, no. 4, pp. 355–378, 2002.

[6] E. Thomas and O. Bleeker, "Options for insertion of rpas into the air traffic system," in *Digital Avionics Systems Conference (DASC), 2015 IEEE/AIAA 34th*. IEEE, 2015, pp. 5B4–1.

[7] A. G. Foina, C. Krainer, and R. Sengupta, "An unmanned aerial traffic management solution for cities using an air parcel model," in *Unmanned Aircraft Systems (ICUAS), 2015 International Conference on*. IEEE, 2015, pp. 1295–1300.

[8] C. A. Wargo, G. Hunter, K. Leiden, J. Glaneuski, B. Van Acker, and K. Hatton, "New entrants (rpa/space vehicles) operational impacts upon nas atm and atc," in *Digital Avionics Systems Conference (DASC), 2015 IEEE/AIAA 34th*. IEEE, 2015, pp. 5B2–1.

[9] "Unmanned systems roadmap 2005 - 2030," Report, Office of the Secretary of Defence, DoD, 2005.

[10] "Unmanned aerial vehicle reliability study," Report, Office of the Secretary of Defence, DoD, 2005.

[11] M. M. van Paassen, M. L. Bolton, and N. Jimnez, "Checking formal verification models for human-automation interaction," in *2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Oct 2014, pp. 3709–3714.