2014 International Conference on Unmanned Aircraft Systems (ICUAS)
May 27-30, 2014. Orlando, FL, USA

Cooperative Sense and Avoid: Implementation in Simulation and Real World for Small Unmanned Aerial Vehicles

Armin Strobel, Marc Schwarzbach

***Abstract*— In this paper we present a framework and a method for sense and avoid to be used with unmanned areial sytems (UAS). The framework is an essential part to develop and validate the sense and avoidance algorithms. It is implemented and used for simulation and on a real system by using a common interface. The avoidance method in this paper is based on the fact that the intruder vehicle has certain limitations in turn rate and acceleration or deceleration. This is used to define the geometry of the threat zone for every intruder. In comparison to other geometry based algorithms the presented method uses the abilities (type) of the intruding aircraft and its current state. Diverse scenarios have been simulated successfully in that framework and code for the real system was generated and implemented.**

I. Introduction

This work is based on a cooperation in the field of simulation and real word test of cooperative Sense and Avoid with Automatic Dependent Surveillance - Broadcast (ADS-B). In the US ADS-B-Out will be mandatory from 2020 for all aircrafts in Class A, B, and C airspace [1]. This provides the chance to use ADS-B for Sense and Avoid, since all information for detecting a collision threat and avoiding it is available.

The advantage of geometric based approaches compared to other approaches like probabilistic [2] and optimisation [3] based algorithms is that they require less processing power. So they are usually more suitable for small UAVs with limited computational resources.

Other geometric based algorithms like [4][5][6] are often based on a fixed definition, which causes a less efficient way to avoid possible threats. Predictive algorithms [7] solve this problem by predicting the intruders and its own path and check if there is a risk of collision without or minimal use of a collision zone. The prediction is usually very consuming on processing power or restricted. The method presented here is a combination of predictive and geometry based algorithm.

A general overview of sense and avoid can be found in [8] [9] [10]. The main focus of this survey is the avoidance part and possible solutions of it.

The avoidance algorithm is first implemented into a simulation environment. The focus was set on the interfaces, which reflect those being present in a real system. This gives the ability of an easy and reliable implementation in the real system via automatic code generation.

A. Strobel is with the Research Training Group Cooperative, Adaptive and Responsive Monitoring in Mixed Mode Environments (GRK 1362) funded by DFG, Germany, armin.strobel@gmail.com
M. Schwarzbach is with the Robotics and Mechatronics Center, German Aerospace Center (DLR), 82230 Wessling, Germany, marc.schwarzbach@dlr.de

II. Structure of the Simulation

The Simulation was divided into several modules, which communicate via UDP commands. This allows an easy interchange of modules or the ability to relocate them. The Structure of the whole simulation is shown in Fig. 1.

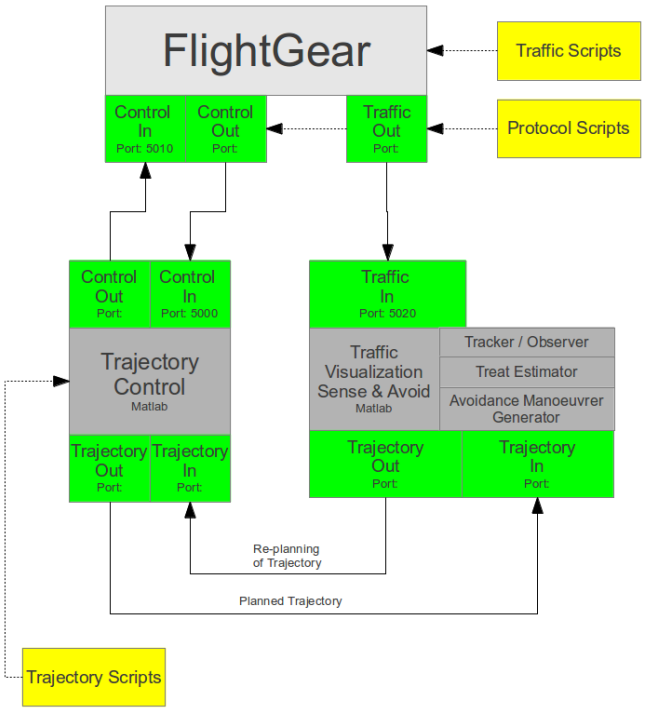

Fig. 1. Structure of the simulation

The Simulation is divided in three separate parts. FlightGear was used to simulate the the flight mechanics of the UAV and Intruders as well as for Visualization. The UAV is controlled by the trajectory control module which consists of a basic control with an interface for replanning a trajectory. This is realized in Matlab and runs under soft real-time. Another Matlab instance tracks the surrounding traffic, conducts a threat estimation and gives, if necessary an avoidance advisory to the trajectory control module.

In a real system the trajectory control module is part of the autopilot. Since the autopilot needs to provide the appropriate interfaces to conduct the avoidance advisory in the expected

way this part was implemented by ourself. The other part can be implemented outside of the main loop of the autopilot or even on a separate controller / computer.

III. Basic Control of the UAV

A. Lateral Control

For short distances, the following approximation of the latitude φ, longitude λ and Earth radius R are sufficient.

$$\Delta x = \cos(\varphi_k) R(\lambda_{k+1} - \lambda_k) \tag{1}$$

$$\Delta y = R(\varphi_{k+1} - \varphi_k) \tag{2}$$

Whereas Δx and Δy are the distances between the waypoint WP_k and the following waypoint WP_{k+1}. The desired heading χ_c to the next waypoint is calculated by the following formula.

$$\chi_c = 90^\circ - \tan(\frac{\Delta y}{\Delta x}) \tag{3}$$

The difference χ_Δ between the actual heading χ and the desired heading χ_c is used to define the desired roll angle ϕ_c

$$\chi_\Delta = \chi - \chi_c \tag{4}$$

The difference between the heading of the UAV and the heading of the desired track is limited $\pm 45deg$. This ensures that the UAV is still flying into the desired direction towards the next waypoint.

$$-45deg < \chi_\Delta < 45deg \tag{5}$$

Whereas desired heading χ_{Δ_c} is determined by the inverse of the actual velocity over ground V, the shortest (perpendicular) distance to the line between the two waypoints d and a parameter k_d.

$$\chi_{\Delta_c} = k_d \frac{1}{V} d \tag{6}$$

The desired roll angle is calculated as followed. Whereas k_ψ is a parameter which has to chosen suitable.

$$\phi_c = k_\psi \frac{V}{g} (\chi_{\Delta_c} - \chi_\Delta) \tag{7}$$

The desired roll angle is limited $\pm 45deg$ to avoid too extreme reactions.

$$-45deg < \phi < 45deg \tag{8}$$

The difference between the desired roll angle ϕ_c and the actual ϕ is used to calculate the necessary deflection for the aileron ζ.

$$\zeta = k_\phi (\phi_c - \phi) \tag{9}$$

B. Vertical Control

The vertical control needs to control two settings, the deflection of the elevator ξ and the power setting η for the engine. Both are coupled through their limitations.

1) Elevator: The desired flight path angle θ_c is dependent on the actual hight h, the desired hight h_c and a parameter k_h.

$$\theta_c = k_h (h - h_c) \tag{10}$$

θ is limited in this case to $5deg$ to avoid overwhelming the aircraft.

$$-5\deg < \theta_c < 5\deg \tag{11}$$

The difference between the desired flight path angle θ_c and actual flight path angle θ together with the parameter k_θ is used to get an elevator command ξ. To avoid a continuous offset a integrating part is added.

$$\xi = k_\theta (\theta - \theta_c) + k_{\int \theta} \int (\theta - \theta_c) dt \tag{12}$$

2) Motor Power Requirement: In comparison to most other flight control algorithms the speed is not directly controlled. The speed is a result of the flight path angle θ and the power setting of the engine η. In this work is the power setting is depending on the flight path angle θ and desired speed $\dot{V}_{TAS}$. This works very well since small derivations on the power P cause even smaller changes on the speed V_{TAS}.

$$P \sim V_{TAS}^2 \tag{13}$$

The power required by the engine is dependent on the change of the potential energy $mg\dot{h}$, the horizontal acceleration maV and the drag $\frac{1}{2}\rho C_w V^3$.

$$P = mg\dot{h} + maV + \frac{1}{2}\rho C_w V^3 \tag{14}$$

For a desired constant speed the power requirement η can be calculated as follows.

$$\eta = c_1 \theta_c + c_2 V_{TAS} \dot{V}_{TAS} + c_3 \tag{15}$$

Whereas c_1, c_2 and c_3 are factors which have to be chosen depending on the UAV.

IV. Avoidance

The Avoidance is achieved by determining the possibility of a collision a short time period into the future (e.g. $30s$). If there is a possibility of collision then a heading command is given to the basic control. The intention of the heading command is to increase the side distance as much as possible. For this, the heading command is calculated as the gradient at the possible intersection point in the future of the threat function 2.

A. Threat Region

The threat region is calculated for a certain time Δt into the future. Figure 2 shows a curve with a constant azimuth rate $\dot{\psi}_i$ which is equivalent to a constant roll angle ϕ_i (16). Whereas V_i is the speed of the intruder and g is the gravitational acceleration.

$$\dot{\psi}_i = \frac{g}{V_i} \tan(\phi_i) \tag{16}$$

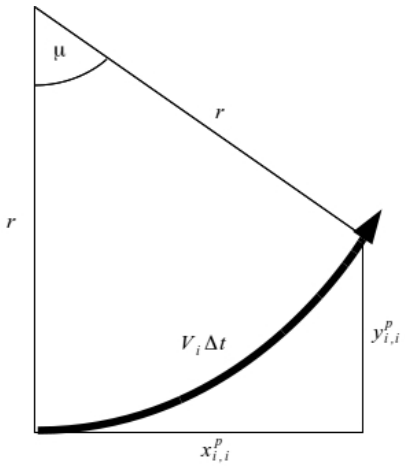

Fig. 2. Determination of the threat region

With formula (17) to (20) the position of the intruder ($x^p_{i,i}$, $y^p_{i,i}$) can be calculated as if it would fly for the time period Δt with a constant roll angle ϕ_i.

$$\mu = \dot{\psi}_i \Delta t \tag{17}$$

$$r = \frac{V_i \Delta t}{\mu} \tag{18}$$

$$x^p_{i,i} = r sin(\mu) \tag{19}$$

$$y^p_{i,i} = r(1 - cos(\mu)) \tag{20}$$

This can be done for the whole range of possible roll angles so that the shape of the threat region is calculated by (21)-(23).

$$\begin{bmatrix} x \\ y \end{bmatrix}^p_{i,i} = V_i \Delta t \begin{bmatrix} \frac{sin(\mu)}{\mu} \\ \frac{1-cos(\mu)}{\mu} \end{bmatrix} \tag{21}$$

$$-\dot{\psi}_{i,max} \Delta t < \mu < \dot{\psi}_{i,max} \Delta t \tag{22}$$

$$\dot{\psi}_{i,max} = \frac{g}{V_i} \tan(\phi_{i,max}) \tag{23}$$

The result is the threat region shown in Fig. 3.

$\dot{\psi}_{i,max}$ is the expected maximum turn rate for the intruder aircraft. A standard full circle takes $2min$ which is equal to a turn rate of $3\frac{\text{deg}}{s}$.

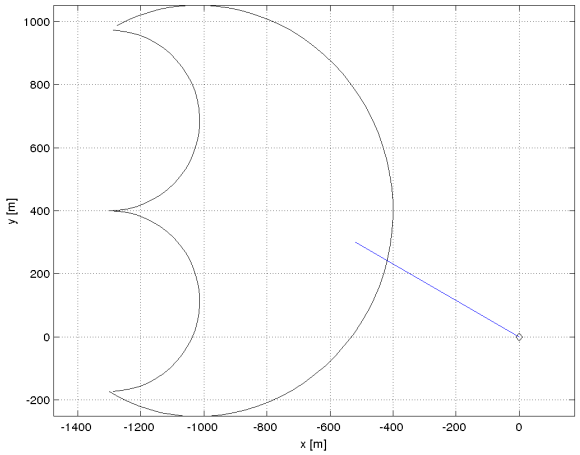

Fig. 3. Region where threat of a collision can exist

B. Threat Detection

For the threat detection the position of the UAV in global coordinates $P_{o,g}$ has to be transformed into intruder coordinates $P_{o,i}$ for easier calculations (24)-(26). $\psi_{i,g}$ is the azimuth angle of the intruder in the global coordinate system and $\psi_{o,g}$ is the azimuth angle of the UAV in the global coordinate system.

$$\Omega_{g,i} = \begin{bmatrix} \cos(-\psi_{i,g}) & \sin(-\psi_{i,g}) \\ -\sin(-\psi_{i,g}) & \cos(-\psi_{i,g}) \end{bmatrix} \tag{24}$$

$P_{o,i}$ is the position and $\psi_{o,i}$ is the azimuth angle of the UAV in the intruder coordinate system.

$$P_{o,i} = \Omega_{g,i} \begin{bmatrix} x_{o,g} - x_{i,g} \\ y_{o,g} - y_{i,g} \end{bmatrix} \tag{25}$$

$$\psi_{o,i} = \psi_{o,g} - \psi_{i,g} \tag{26}$$

With the heading of the UAV the position $P^{\Delta t}_{o,i}$ is determined at the time Δt.

$$P^{\Delta t}_{o,i} = P_{o,i} + V_i \Delta t \begin{bmatrix} \cos(\psi_{o,i}) \\ \sin(\psi_{o,i}) \end{bmatrix} \tag{27}$$

If this position is touching or inside the threat region, a possible threat of a collision is detected. To determine if $P^{\Delta t}_{o,i}$ is inside the threat region in an efficient way, the threat region is approximated by (28)-(29).

$$\mu^* = 2 \cos^{-1}(\frac{x^{\Delta t}_{o,i}}{V_i \Delta t}) \tag{28}$$

$$y^* = V_i \Delta t \frac{1 - \cos(\mu^*)}{\mu^*} \tag{29}$$

If $|y^*| > |y^{\Delta t}_{o,i}|$, a possible collision is detected. An avoidance resolution will be generated in this case.

C. Avoidance Resolution

Based on the current location of the UAV $P_{o,i}$ relative to the intruder, a new desired heading is generated. If $y_{o,i} > 0$, the new desired heading is set to $\frac{\pi}{2} + \psi_{i,g}$. In any other case it is set to $-\frac{\pi}{2} + \psi_{i,g}$. The basic controller is supposed to hold that heading for the time period of dt.

D. Simulation Results

The simulation shows the view from the UAV. The right side of Fig. 4 shows the map with the threat region of the intruder with the aircraft (UAV) in the middle.

Fig. 4. Screen shot of simulation short before threat detection

Once a possible threat is detected (predictor intersects threat region) the map turns red and the avoidance manoeuvre is performed (Fig. 5).

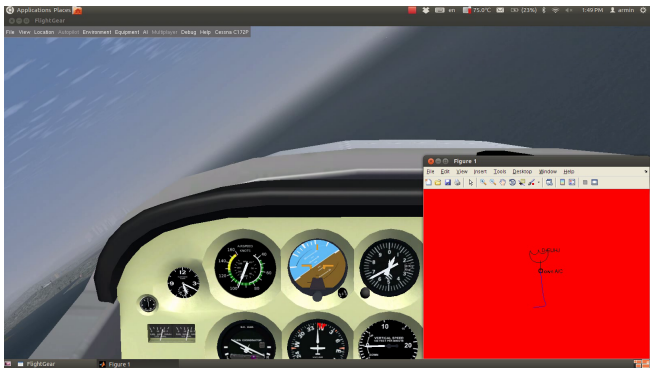

Fig. 5. Screen shot of simulation short after threat detection

After the threat no longer exists the UAV goes back to its original flight path (Fig 6).

In figure 7 trajectories of the UAV and the position of the intruder with threat region for different times are shown.

A video of this example scenario can be found in [11]

V. Implemetation

The algorithm developed should now be tested in an unmanned flying system. Threat detection and resolution will therefore be implemented in an existing autopilot system.

Since most civil applications for UAS are developed using small unmanned aircraft systems (sUAS), we will focus on this kind of implementation first. As a consequence, ground

Fig. 6. Screen shot of simulation while intersecting planned trajectory

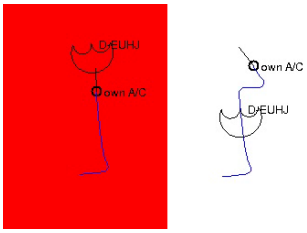

Fig. 7. Trajectories for different times

based systems for traffic detection will be used. It also allows for operation and testing in visual line of sight (VLOS), which does not require restricted airspace in most countries.

A. Platform

The UAV autopilot system *Paparazzi* [12] was started as an open source project in 2003 [13]. Since then, it has been widely used in research and applications [14][15]. Several hardware platforms are available, allowing the control of small to large flying systems, both fixed and rotary wing. Through its main base is in academia, a focus is modularity of the software system, which is a big difference to many other systems, including commercial ones. This allows for the addition of flight algorithms and interfaces, being very important in the context of the problem considered here.

On-board software is generated by a build-system, including all configured modules and subsystems. The ground station is modular by connecting data-link, user interface and data server by a publisher-subscriber bus (Fig. 8), called *IVY* [16]. There are also simulator and interface modules to be connected to the ground station bus to allow for simulation and interfacing with other applications.

Multi-UAV functionality is already included in the system, providing information of all planes connected to the ground station to all other flying systems. This information has since been used for following of aircraft or formation flying. We will use the data as input to the avoidance algorithm.

A Paparazzi *Lisa/M2.0* Autopilot is used for the implementation presented here. The complete system including GPS and telemetry weighs less then 40 grams. The setup is shown in Fig. 9. Small and lightweight aircraft are used for testing to reduce cost and risks.

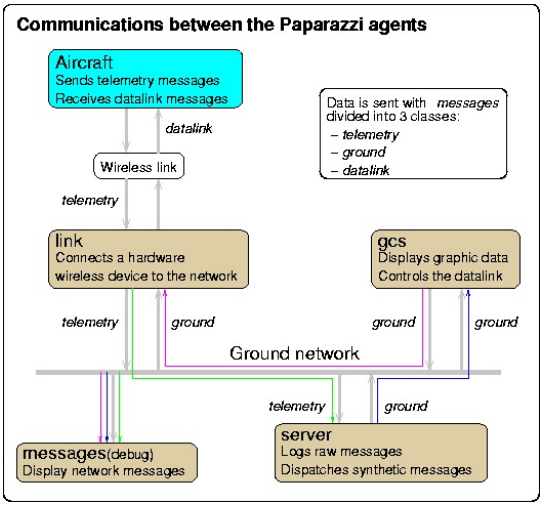

Fig. 8. Paparazzi ground station bus system (source: [12])

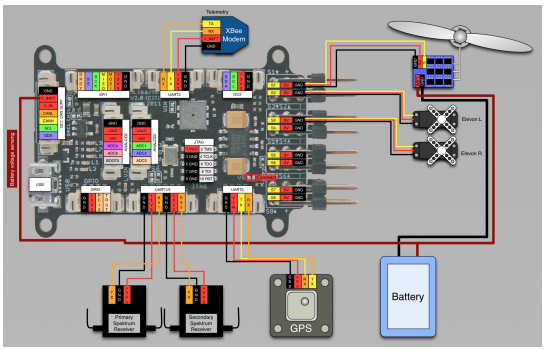

Fig. 9. Paparazzi Lisa/M hardware setup (source: [12])

B. Integration

A functional interface was defined to use the same algorithms in Matlab simulation as in the autopilot. For compilation in the Paparazzi environment, a wrapper was created making it a navigation module. Every other aircraft in the autopilot system will now be recognized as a threat, when criteria of the algorithms are met.

As described earlier, a software in the loop (SIL) simulator is included in the paparazzi system. It is used to test functionality of the implementation by starting two of these airplane simulation processes, one using the avoidance algorithm. After functionality has been verified, the software is loaded onto the autopilot and tested in flight, while the intruder is still a simulated aircraft. In the last step of testing, two UAV are used, one acting as the intruder.

In the final setup, all intruders to be recognized will be put as signals to the ground station bus. The next chapter will describe the technology.

By using this step by step approach, we can safely test the limits and the performance of the avoidance algorithm.

C. Detection of manned aircraft

Intruder detection is one of the most challenging tasks in the field of UAS. Especially small UAS do not allow the installation of complicated and heavy optical or radar systems like proposed in [17], [18]. Therefore external systems involving cooperating traffic are considered to be a solution. As stated before, ADS-B is considered a candidate for such a system [19]. ADS-B only covers powered aircraft at the moment, we also include the FLARM system [20]. The FLARM system is installed in the majority of glider airplanes in Europe.

We have implemented two interfaces to our autopilot system. Using a *PowerFlarm* (Fig. 10) system as a receiver, airplanes equipped with ADS-B or FLARM are detected. Via a serial interface and a special software module, this data is read from the device and injected to the Paparazzi data bus as intruder aircraft. Drawback of the solution is the price, which is 1800 €.

Fig. 10. Powerflarm device (source: [20])

As a cheap solution an open source software defined radio receiver for ADS-B was adapted to the autopilot system. Using a commercial USB digital terrestrial television receiver (DVB-T), the signal can be decoded [21], [22]. The interface program calculates the positions of all received aircraft relative to the UAV and sends the closest one to the autopilot as the intruder. All data is also visualized in the ground station, as seen in Fig. 11.

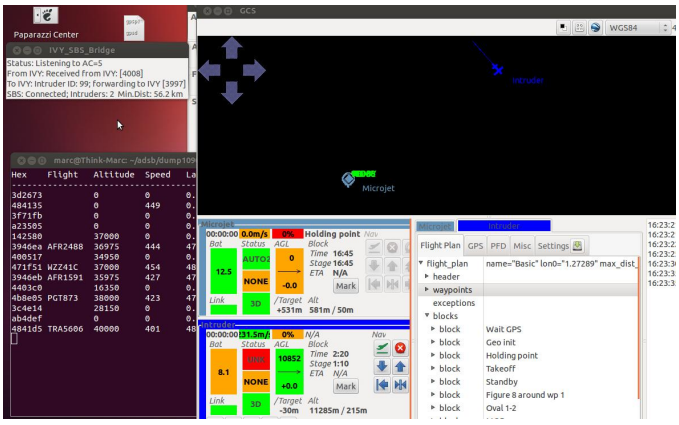

Fig. 11. ADS-B receiver showing intruder in ground station

As an additional test, a simulated aircraft can be used to test the reaction of the UAV to a real (ADS-B) intruder without having to fly it close to a manned aircraft.

When using larger UAV, the receiving and processing of intruder data can be performed on-board, increasing range significantly.

D. Status

Integration of all parts of the sense and avoid system to the autopilot has been performed. Intensive SIL testing confirmed proper fuctionality. Planes with integrated autopilots are in use for several tasks since many years. Flight tests included multi-UAV flights. Step by step testing of the sense and avoid system in flight will commence as soon as conditions allow.

VI. Conclusion

The method presented works well for single intruders. Through the prediction, the UAV can detect a possible threat and avoid it well in advance. Compared to other algorithms using prediction this method is very efficient with computational requirements and can, as shown, be implemented in a small autopilot like Paparazi. Compared to algorithms with a fixed threat zone the introduced algorithm incorporates a more realistic threat prediction. This leads to avoidance manoeuvres when the risk actually exists, and less unnecessary avoidance manoeuvres.

All simulations have been performed with a single intruder at a time. Problems could arise if there is more than one intruder at one time or if the intruder is changing its speed quickly. For general aviation this is usually not the case, since the performance is very limited. Commercial airliners have more performance and can accelerate very fast, but for passenger comfort and through limitations they are not using these capabilities.

An implementation using the open source autopilot Paparazi has been performed. Simulation results are as expected allowing for start of real flight experiments. By including the two most common cooperative sense and avoid systems, ADS-B and FLARM, the system has the potential to be used in real applications.

References

[1] F. A. A. Washington DC, US Department of Transportation, "Nextgen implementation plan," 2013.

[2] K.-Y. Kim, J.-W. Park, and M.-J. Tahk, "Uav collision avoidance using probabilistic method in 3-d," in *Control, Automation and Systems, 2007. ICCAS'07. International Conference on*. IEEE, 2007, pp. 826–829.

[3] A. Richards and J. How, "Aircraft trajectory planning with collision avoidance using mixed integer linear programming," in *American Control Conference, 2002. Proceedings of the 2002*, vol. 3, 2002, pp. 1936–1941 vol.3.

[4] S.-C. Han and H. Bang, "Proportional navigation-based optimal collision avoidance for uavs," in *2nd International Conference on Autonomous Robots and Agents*, 2004, pp. 13–15.

[5] J. Van Tooren, M. Heni, A. Knoll, and J. Beck, "Development of an autonomous avoidance algorithm for uavs in general airspace," in *Proceedings of First CEAS European Air and Space Conference*, 2007.

[6] F. Lindsten, P.-J. Nordlund, and F. Gustafsson, "Conflict detection metrics for aircraft sense and avoid systems," in *Proceedings of the 7th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes*. IFAC, 2009, pp. 65–70.

[7] E. Boivin, A. Desbiens, and E. Gagnon, "Uav collision avoidance using cooperative predictive control," in *Control and Automation, 2008 16th Mediterranean Conference on*, June 2008, pp. 682–688.

[8] B. Albaker and N. Rahim, "A survey of collision avoidance approaches for unmanned aerial vehicles," in *Technical Postgraduates (TECHPOS), 2009 International Conference for*. IEEE, 2009, pp. 1–7.

[9] P. Angelov, *Sense and Avoid in UAS: Research and Applications*, ser. Aerospace Series. Wiley, 2012. [Online]. Available: http://books.google.de/books?id=jvRIiEYc048C

[10] K. Valavanis and G. Vachtsevanos, *Handbook of Unmanned Aerial Vehicles*. Springer Netherlands, 2014. [Online]. Available: http://books.google.de/books?id=CZSVngEACAAJ

[11] "Video of avoidance scenario," http://youtu.be/E1_loTL5CX4.

[12] "Homepage of the Paparazzi-project," http://paparazzi.enac.fr.

[13] P. Brisset, "The Paparazzi solution," in *Micro Air Vehicle Conference 2006*, 2006.

[14] J. Reuder and et. al., "SUMO: A small unmanned meteorological observer for atmospheric boundary layer research," *2008 IOP Conf. Ser.: Earth Environ. Sci.*, vol. Vol.1, 2008.

[15] M. Schawarzbach, U. Putze, U. Kirchgner, and M. von Schoenermark, "Acquisition of high quality remote sensing data using a UAV controlled by an open source autopilot," in *ASME 2009 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference (IDETC/CIE2009) August 30September 2, 2009 , San Diego, California, USA*, vol. 3, 2009, pp. 595–601.

[16] "Homepage of ivy bus system," http://www.eei.cena.fr/products/ivy/.

[17] A. Zarandy, T. Zsedrovits, Z. Nagy, A. Kiss, and T. Roska, "Visual sense-and-avoid system for uavs," in *Cellular Nanoscale Networks and Their Applications (CNNA), 2012 13th International Workshop on*, Aug 2012, pp. 1–5.

[18] S. Kemkemian, M. Nouvel-Fiani, P. Cornic, P. Le Bihan, and P. Garrec, "Radar systems for sense and avoid on uav," in *Radar Conference - Surveillance for a Safer World, 2009. RADAR. International*, Oct 2009, pp. 1–6.

[19] B. Stark, B. Stevenson, and Y. Chen, "Ads-b for small unmanned aerial systems: Case study and regulatory practices," in *Unmanned Aircraft Systems (ICUAS), 2013 International Conference on*, May 2013, pp. 152–159.

[20] "Homepage of FLARM," http://www.flarm.com.

[21] "Homepage of rtl-sdr," http://sdr.osmocom.org/trac/wiki/rtl-sdr.

[22] "Homepage of Dump 1090," https://github.com/antirez/dump1090.