A Multi-UAS Simulator for High Density Air Traffic Scenarios

David Martín-Lammerding
Department of Stats. Comput. Sci. Math
Public University of Navarre (UPNA)
Pamplona, Spain
email:david.martin@unavarra.es

José Javier Astrain
Department of Stats. Comput. Sci. Math
Public University of Navarre (UPNA)
Pamplona, Spain
email:josej.astrain@unavarra.es

Alberto Córdoba
Department of Stats. Comput. Sci. Math
Public University of Navarre (UPNA)
Pamplona, Spain
email:alberto.cordoba@unavarra.es

***Abstract*—Unmanned Aerial Systems (UASs) have become enormously popular as they improve many applications like structural inspections, homeland security, delivering packages, etc. UASs share the same airspace as manned aircrafts and thus autonomous UAS operations and current airspace management have to be integrated. It is recognized that the use of small UASs and taxi-drones at lower altitudes is now a driving force of economic development, but a safety risk when their numbers increase. UASs will fly and make decisions autonomously using only on-board sensors and processors. As the number of simultaneous UAS flights increase, keeping a safety airspace is a challenge that implies that UAS navigation autonomy must be verified for conflict avoidance in high density air traffic areas. The required scenarios to verify air traffic management algorithms and autonomous capabilities of UASs are complex to deploy and UASs may collide, therefore simulations are of great importance. This paper presents a conflict simulator of multiple UASs with different equipment. The main purpose of the simulator is to verify UAS subsystems, like collision avoidance, in different scenarios with a combination of different equipped UASs and flight plans. To evaluate its effectiveness, we performed an integrated simulation process for a Collision Avoidance Systems (CAS) implementation where Hardware In the Loop (HIL) and software simulations are combined to improve UAS subsystems safety and to reduce the time-to-market.**

***Keywords*—*Simulator*; *HIL*; *UAS*; *autopilot*; *autonomous*; *CAS*; *conflicts*; *anti-collision*; *air traffic*.**

I. Introduction

The use of Unmanned Aerial Systems (UASs) in various fields, such as military, policing, firefighting, etc. has evolved in multiple UAS types with different characteristics, such as size, speed and flight range. Each UAS must be equipped with new safety systems, like a Collision Avoidance System (CAS), to fly the shared airspace with other aircraft. CASs detect airplanes in airspace, discover potential collision hazards and perform maneuvers to avoid collisions. Rigorous simulations of UAS traffic benefits application development of any UAS subsystem, like CAS, and the integration of UASs in the common air-space safely. Therefore, there is a real need of a simulator that can combine multiple UASs with different equipment and capabilities.

Accelerating, enhancing the development and testing of CAS and other software elements that provide autonomy to UAS are a key challenge. The logistical effort to deploy multiple conflicting UASs imposes a high cost and complexity. Evaluating the effectiveness of UAS systems requires an automated process to simulate every new release.

In this paper, we present an UAS simulator, called *SIMUdrone*, that is suitable for simulating high density air traffic operations with UASs. The simulator supports multiple UASs in user-predefined scenarios and in randomly automatic generated scenarios. Each simulation consists of multiple UASs with a Flight Control Unit (FCU) and a Global Navigation Satellite System (GNSS). Simulated UASs may have an autopilot, a CAS, sensors for conflict detection (video-camera, LIDAR, etc.) or a radio equipment to control it remotely by a pilot. The main motivation for developing this simulator lies in the necessity to generate multiple conflict scenarios with combinations of multiple types of UASs and to integrate in the simulations specific UAS subsystems executed on specific hardware.

The rest of the paper is structured as follows. Section II presents the state of the art of UAS simulators, Section III defines the problem statement and Section IV describes our contribution. The simulator is presented in Section V. Section VI is devoted to presenting an experimental simulation. Section VII presents the conclusions and references end the paper.

II. Related Work

Next, we review the main Commercial Off-The-Shelf (COTS) UAS simulators available.

The *SITL simulator* is used for testing an autonomous quadcopter [1]. It is intended for executing a FCU based on ArduPilot in a PC. However, the *SITL simulator* does not allow to simulate multiple UASs simultaneously. Other UAS simulators focus on simulating the physical model of the vehicle. Some vehicle simulators are XPlane [2], Flightgear [3], Gazebo [4], JMavSim [5][6]. For our purposes, any of the previously reviewed simulators requires a costly adaptation to model UAS traffic.

AirSim [7] is a visual simulator of different types of vehicles, including UAS, based on the video game engine Unreal Engine. It is focused on the development of AI algorithms based on deep learning. Simulation time can be reduced but at the cost of losing accuracy. However, response time and multiple UASs simulations are limited as the simulation runs in

 ISBN: 978-1-61208-974-4

a desktop computer and it does not provide predefined conflict scenarios.

UTSim [8] is an UAS conflict simulator based on Unity. Anti-collision algorithms can be integrated in the simulation executed by UTSim, but UAS sub-systems can not be run in HIL mode. There are air traffic management simulators that include UAS traffic, such as BlueSky [9]. This type of simulator is focused on air traffic planning but not on simulating conflicts and collisions. In [10], two simulators are proposed: the TIMed State space Performance Analysis Tool (TIM-SPAT) [11] and the CPN-tools [12]. The former requires a complex configuration. The latter is a limited tool with short simulation configuration capabilities.

In [10], an anti-collision system is simulated using TIMed State space Performance Analysis Tool (TIM-SPAT) [11], developed at the Logistics and Aeronautics Unit of the Autonomous University of Barcelona. But it has a complex configuration and development. Another tool used in [10] is CPN-tools [12], a basic simulator with limitation in the UAS equipment configuration.

A Real-time Multi-UAV Simulator (RMUS) was presented in [13]. It supports multiple UAVs, data collection and control but lacks a flexible configuration of environments.

Multiple UASs simulators exist [14][15] but they are not focused on testing multiple conflicts and their avoidance.

Table I classifies the main UAS simulators using selected capabilities: HIL simulation, multiple UASs simulation, and external implementation/code integration in the simulator.

TABLE I
MAIN UAS SIMULATORS AND TOOLS FEATURES.

	Ref	HIL simulation	Multiple UASs	Data link	External code integration
Airsim	[7]		✓		
Gazebo	[4]		✓		
UTSim	[8]		✓	✓	✓
RMUS	[13]	✓		✓	
SIMUdrone	This	✓	✓	✓	✓

III. PROBLEM STATEMENT

UAS applications deployment require a security verification for the onboard systems to fly in high density air traffic scenarios. Very Low Level airspace (VLL) is the space below 500 ft. above ground level. It is the part of the airspace intended for new UAS applications and it will concentrate most UAS conflicts. A conflict between two UASs occurs when minimum separation, defined as the protection radius, r_p, is lost. Figure 1 shows a conflict between *local UAS* and *remote UAS*. A loss of separation does not always predict a future collision, but it is a key safety indicator.

An UAS flies over some particular locations, called waypoints. A waypoint is a location determined by GNSS. A flight plan consists of an ordered set of waypoints that the onboard autopilot follows. A CAS deployed in an UAS is aimed at maintaining a minimum safe separation between UASs. Once a conflict is detected, a CAS diverts the UAS to a new safe path, changing the flight plan autonomously. The number of simultaneous conflicts are denoted as n_C.

UAS traffic in VLL airspace consists of UASs with an autopilot, remotely piloted UASs and even autonomous UASs. Therefore, CAS must deal with conflicts with different UAS types, capabilities and in random relative locations.

Most CASs for UASs are distributed so they run in an onboard computer, commonly denoted as an embedded board. However, the payload of the UASs limits the weight and the energy consumption of the embedded board, as well as its computing power.

One of the derived requirements from the above is that the response time of the CAS running in an embedded board should be considered in any simulation as it can reduce the time to react.

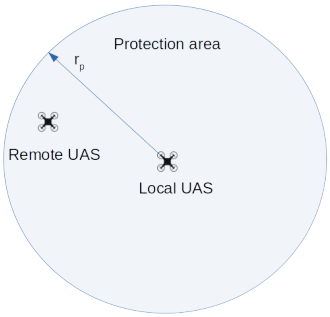

Figure 1. Conflict between a local UAS and a remote UAS.

UAS software is necessarily integrated with hardware and embedded systems for autonomous behavior. An intensive testing with live-fly field experiments are costly and highly time consuming. Therefore, simulations should combine HIL simulations and other simulations based on software models.

IV. CONTRIBUTION

SIMUdrone is a novel multi-UAS simulator for high density air traffic scenarios suitable for urban environments. It provides a unique conflict generation framework with extensive configuration capabilities in order to simulate the variability of situations in urban environments. Integration of external software components and hardware in the simulations are two key features of *SIMUdrone*.

SIMUdrone improves UAS safety and reduces the time to market for UAS subsystems development as it implements an automated integrated continuous simulation for each new UAS subsystem release. The automated integrated continuous simulation combines HIL simulations with simulation based on software model.

SIMUdrone simplifies the verification of UAS subsystems, like CAS, as it can generate extensive traffic combinations to simulate different scenarios. A large dataset of data from every simulation is available to verify its behavior.

 ISBN: 978-1-61208-974-4

V. SIMUdrone simulator

SIMUdrone simulates UAS traffic, the effect of the environment surrounding the UAS during the flight, the onboard equipment of the UAS and it can integrate external UAS hardware using the *HIL mode*.

SIMUdrone is implemented in JAVA 8, HTML, CSS and JS. The simulation output consists of multiple configurable log files and a HTML5 conflict animation. A conflict animation simplifies the analysis of a simulation scenario and the results obtained. It can be easily shared with others as it can be viewed in any PC with a browser.

To generate conflicting UAS scenarios, *SIMUdrone* provides custom scenarios, randomly generated with configurable UASs types. UAS equipped with an autopilot can fly following a set of waypoints that can be configured, as well as the flying speed. UASs with a CAS integrated with the autopilot can avoid conflicts by changing the predefined flight plan. UASs remotely piloted can be also included in a simulation. To simulate a remotely piloted UAS, *SIMUdrone* allows to configure an area centered on the pilot where the UAS flies randomly for a time interval to limit its duration.

A. Architecture

SIMUdrone consists of multiple software components that are combined to simulate a complete UAS system and a configurable traffic generator. A complete UAS system can be simulated combining software models with a HIL simulation to test a software component running on a specific piece of hardware.

To simulate a flight environment and provide sensor inputs during simulation, we develop a virtual collaborative transponder and a sensor system that generates conflict data. The CAS log, the autopilot status and the scenario animation are generated and stored in a proper format for each type.

SIMUdrone implements an autopilot and a reference CAS implementation to facilitate the comparison with other implementations. The CAS implemented in *SIMUdrone* is based on the Potential Field (PF) technique. PF is a collision avoidance path calculation method that simulates a force field where the UAS is attracted to its final destination and repulsed from obstacles or conflicts [16], [17].

SIMUdrone implements two modes of operation, the *HIL-mode* and the *integrated-conflict-mode*, as depicted in Figure 2.

The *HIL-mode* is devoted mainly to run a CAS implementation in the planned hardware that will be used as a companion-pc on the UAS. The *HIL-mode* is a testbed that provides a safe environment for testing any external systems in real time, like CAS. *SIMUdrone* implements two adaptation layers, a *SIMUdrone* Input Layer (SIL) and a *SIMUdrone* Output Layer (SOL), to integrate external implementations running on a specific piece of hardware. The communication between *SIL*, *SOL* and *SIMUdrone* could be configured by a REST interface or a serial communication using the MAVLink protocol.

A *SIMUdrone* HIL simulation can be deployed in two ways:

The first consists of a single embedded board running the CAS in order to profile its response time and use it in a later simulation. Therefore, a trusted CAS response time is obtained and it is available for subsequent simulations. This is the initial step before a complete software simulation is performed. The two layers are adaptable and modular to simplify the integration with external hardware. The HIL response time profiling of *SIMUdrone* works as follows: *SIMUdrone* sends conflict data to the input layer and receives the maneuver at the output layer. *SIMUdrone* coordinates the two layers to obtain the response time for each input-output.

The second consists of n_C boards connected to *SIMUdrone*. Each board runs the CAS of a simulated UAS whose FCU and the rest of its equipment are software models instantiated by *SIMUdrone*. Each board runs a CAS instance that receives conflict data and sends avoidance maneuvers through the SIL and SOL layers, respectively.

The *integrated-conflict-mode* is devoted to simulate multiple UASs in a cluster using software models for the UAS equipment, the environment and the communications. The *integrated-conflict-mode* simulates n_C conflicting UASs with different equipment flying in a conflict area. It simulates the scenario in virtual time considering the real response time of the UAS equipment, the latency of communications and the delay of any onboard processing.

External implementations of UAS subsystems can be integrated in the simulator as components or libraries linked to the simulator code. However, UAS subsystems response time may not be the same when running in a cluster. For this case, an initial HIL simulation is required to provide data to configure and tune a response time model.

B. Simulated scenarios

Simulated scenarios can be defined programmatically. However, there are two pre-defined scenarios for convenience, a *conflicting-area* and a *conflicting-point* scenario.

A *conflicting-area* scenario consists of a rectangular airspace area where n_C configurable UASs fly. A default configuration defines a two waypoint flight plan for any autopilot-equipped UAS. Initial UAS location, heading, speed and fly distance are randomly distributed using configured intervals. This scenario is useful to test different densities of conflicts and how a conflict influences others, as depicted in Figure 3.

In a *conflicting-point*, *SIMUdrone* simulates n_C UASs that converge in a collision point p_C. To achieve this, *SIMUdrone* defines a conflict-circle, whose center is the collision point p_C. Each UAS is randomly positioned on a conflict-circle circumference. The minimum separation considered among UASs on the conflict-circle circumference is a configuration parameter. The point p_C is the center of the conflict-circle circumference. Therefore, the fligh plan is the diameter composed of three points, the initial point on the circumference, w_1, the center, p_C and w_2. This scenario is useful when it is required to test a CAS with a fixed n_C conflicting UASs in a limited area, as depicted in Figure 4.

A *conflicting-point* simulation mission starts at waypoint w_1 and it is completed if the UAS arrives at w_2 without being involved in a collision. In addition to the conflict area around

 ISBN: 978-1-61208-974-4

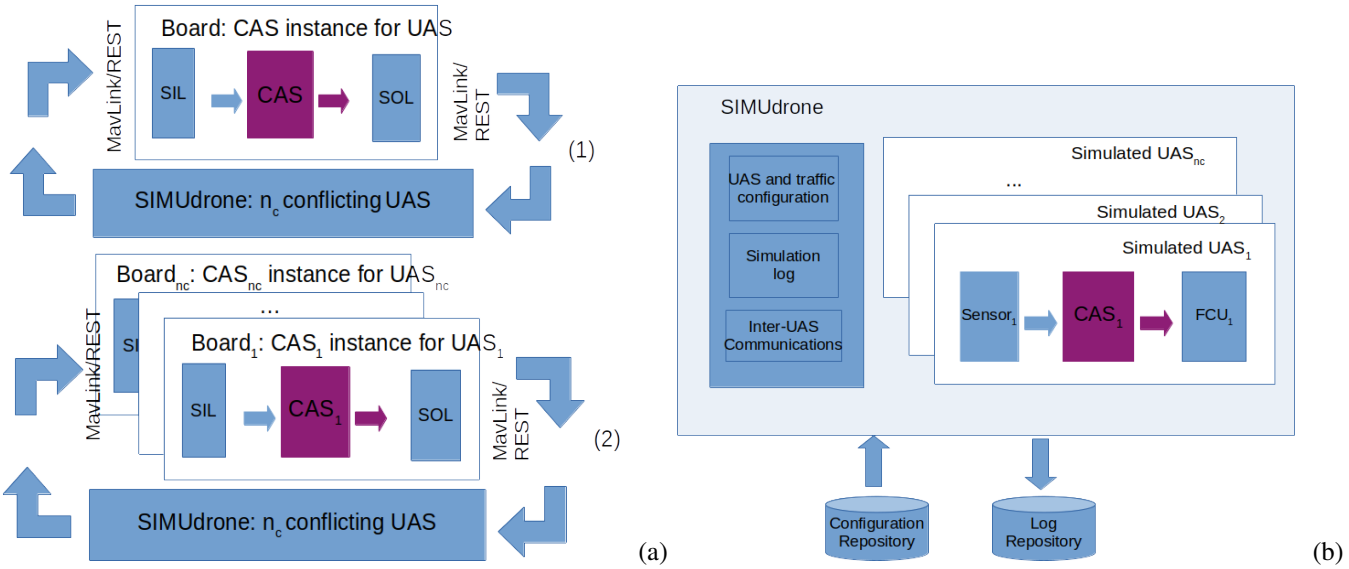

Figure 2. *SIMUdrone* modes: (a) HIL mode (b) Conflict mode

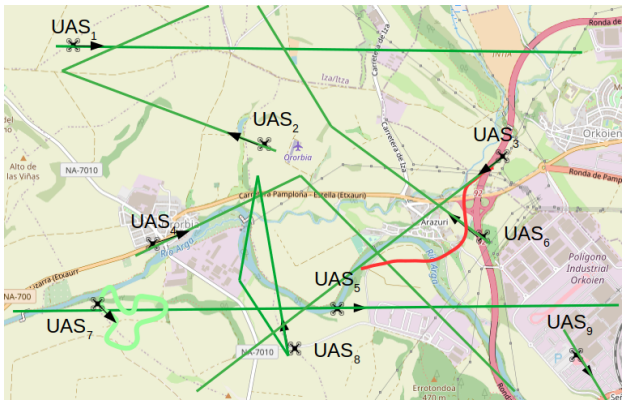

Figure 3. *Conflicting-area* simulation of 9 UAS in a 5x5 Km. area. UAS_3 is an autonomous UAS equipped with a CAS that avoids a collision changing its planned path (in green) to the one depicted in red. UAS_7 is a remotely piloted whose path revolves around a point as the pilot performs a visual line of sight flight. The rest of UAS are equipped with an autopilot.

point p_C, there may be other conflict areas located at the beginning of the flight if the separation between UASs is lower than the conflict distance configured. Animation examples are available at [19] and [20].

VI. SIMUDRONE SIMULATION

Next, we present a *SIMUdrone* simulation configured to show an automated integrated continuous simulation that combines an HIL simulation and a software model simulation.

A. Configuration

First, we review *SIMUdrone* general configuration capabilities. After that, we present the configuration used for the simulation.

SIMUdrone is a customizable simulator with a set of configuration parameters to simulate multiple conflict scenarios in any airspace class, multiple UAS typologies, multiple payloads and different flight plans. Simulated UASs types available in *SIMUdrone* are Vertical Take-Off and Landing (VTOL) or copter. Their class are *micro*, *mini light* or *small* [18] so their Maximun Take-Off Mass (mtom) is less than 150 Kg. This are the most common types in the VLL airspace. Multiple UAS flight altitudes can be simulated. Conflicts can be avoided by changing altitude maneuvers but the limited altitude in VLL airspace and the limited precision for latitude measurement, restrict the crosses at different altitudes. The PF implementation available in *SIMUdrone* does not change altitude. Other known CAS implementations, that can be simulated in any of the two *SIMUdrone* modes, may change altitudes.

The simulated UAS can be configured in three types: autopilot, autonomous with CAS and remotely piloted. The autopilot and the autonomous UAS require a flight plan defined with a set of waypoints, speeds between waypoints and flight altitudes. The remotely piloted UAS requires a pilot location and a maximum distance from it to generate a random path that is contained in a cylinder whose base is defined by the pilot location and the radius is the maximum distance for the visual flight allowed. *SIMUdrone* can be configured to simulate the effect of wind and GNSS's inaccuracies with random variations of the bearing required to fly to the next waypoint.

SIMUdrone simulates conflict detection sensors, like an ADS-B transponder or a vision camera. A simulated vision-based detection system consists of a vision camera mounted in the forward direction of the UAS on the symmetry axis. The CAS implemented in *SIMUdrone* generates an escape maneuver when the distance to a conflict is less than the protection radius (r_p). If the distance is less than the safety radius (r_s) the escape maneuver is sharper. A collision occurs between two or more UASs if the distance between them is

 ISBN: 978-1-61208-974-4

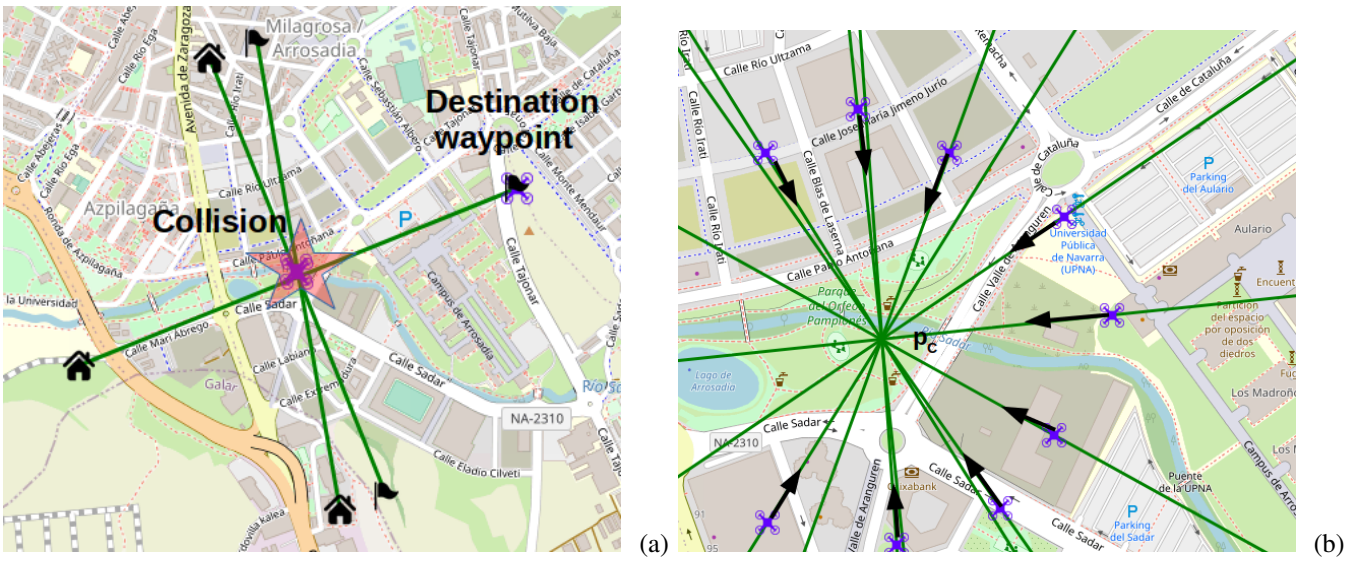

Figure 4. *Conflicting-point* simulations.(a) Two UASs collide and a third one arrives its destination waypoint (b) Nine UASs approaching a collision point p_C

less than the collision radius (r_{cc}).

For the simulation performed, a specific set of simulation parameters are selected and summarized in Table II.

TABLE II
MAIN *SIMUdrone* PARAMETERS CONFIGURED.

Parameter	Value
Typology of simulated UAS	Copter
UAS class	Light
Simulated UAS	All autonomous UAS with CAS
CAS integrated	*SIMUdrone* PF implementation
Onboard sensors	ADS-B and vision camera
n_C	[2,9]
Protection radius (r_p)	100 m
Safety radius (r_s)	50 m
Collision radius (r_{cc})	5 m
Speed range (sp)	[20, 25] m/s
UAS distance to p_C	[350, 650] m
Altitude (h)	50 m
GNSS inaccuracies distance	[-2, 20] m random
GNSS inaccuracies bearing	[0, 360] °
Wind effect bearing variation	[-4,4]°

Tables III and IV list the features of the equipment configured in the simulation.

TABLE III
SIMULATED PARAMETERS OF THE VISION-BASED CAMERA.

Parameter	Value
Detection distance (d_{DT})	[300,350] m
Coverage angle (θ)	[140,160] degrees
Frequency (f_z)	20 Hz

B. Simulation results

The *SIMUdrone* conflict simulation performed combines a HIL simulation and two *integrated-conflict-mode* simulations configured as *conflicting-point* scenarios.

TABLE IV
SIMULATED PARAMETERS OF A LOW POWER ADS-B TRANSPONDER.

Parameter	Value
Coverage distance (d_{CB})	[1500,2500] m
Frequency (f_{ec})	1 Hz

The HIL simulation obtains a Response Time, RT, of the *SIMUdrone* CAS implementation running in a PI3 embedded board [21]. The *SIMUdrone* CAS runs in a PI3 board connected to the main *SIMUdrone* instance via MAVLink. *SIMUdrone* generates conflicts that are sent to the board and keeps the conflict scenario state. Next, *SIMUdrone* waits for the CAS response, updates the internal simulation state and measures the response time.

The *integrated-conflict-mode* simulation is performed in two different *conflicting-point* scenarios: One of the scenarios simulates n_C UASs without CAS and the other simulates n_C UASs with the CAS implemented in *SIMUdrone* using the response time obtained from the previous HIL simulation.

Results of the simulations are shown in Table V. A simulated mission with n_C conflicting UASs is successful if every UAS arrives to its destination waypoint. The success rate measures how many iterations are successful. The number of conflicts simulated are between 2 and 9. The upper value is defined using the results of a worst-case high traffic volume (maximum of 100000 flights per day) presented in [22]. Every simulation performed repeats 100 iterations for each scenario.

The performed *SIMUdrone* simulation shows that it allows to combine multiple simulation modes in order to automate the test of an UAS subsystem implementation. When a new version of the UAS subsystem is available, *SIMUdrone* allows to update the simulations results performing a new simulation with the new implementation version. The integrated simulation process configured in *SIMUdrone* runs the updated imple-

 ISBN: 978-1-61208-974-4

TABLE V
SUMMARY OF SIMULATIONS PERFORMED.

	mean	success	Collided UASs (no CAS)		success	Collided UASs with CAS (equip CAS)	
n_C	RT (ms)	%	mean	sdev	%	mean	sdev
2	6.45	59	2.00	0.00	99	2.00	0.00
3	8.44	13	2.01	0.11	78	2.00	0.00
4	9.45	0	2.57	0.89	69	2.06	0.36
5	8.75	0	3.68	0.72	64	2.17	0.56
6	7.68	0	4.80	0.97	38	2.24	0.99
7	8.86	0	5.70	0.76	32	3.00	1.22
8	8.56	0	6.81	1.13	25	2.89	1.21
9	8.45	0	7.75	0.82	20	3.13	1.30

mentation in a embedded onboard, obtains updated response times, updates the software model and simulates multiple scenarios with UASs equipped with the new implementation.

VII. CONCLUSION

UAS evolution requires the rapid implementation and evolution of UAS subsystems for its integration in air traffic. The verification of such UAS subsystems must be performed in simulations because it is expensive if it is physically performed. Therefore, simulators are vital to evaluate proposed algorithms and their performance in specific hardware. In this paper, we present *SIMUdrone*, a simulator that combines scenarios with multiple UASs, HIL simulations and the integration of external algorithms in any subsystem. It allows to configure UASs with different equipment, like an ADS-B transponder or a CAS, and with custom or random generated flight plans. Furthermore, *SIMUdrone* has been tested using a combined HIL simulation with a software simulation of n_C conflicts to verify its capability to integrate and automate a complete stack of continuous simulations. A continuous simulation process reduces the time-to-market and improves air traffic safety.

Additional enhancements will be focused on the development of more simulated components in order to test other collaborative technologies, sensors or new conflict avoidance algorithms. Another line of work is to improve the *SIMUdrone* interoperability with algorithms implemented as external binaries that can not be integrated in the code or executed in *HIL mode*. Further developments of this work will include integration in the *HIL mode* of hardware that uses V2X (Vehicle-to-everything) communication protocols or Controller Area Network (CAN) BUS. Future work will be focused on the creation of a *dataset* with conflicts and resolution maneuvers to be used for UAS air traffic research and for development of CAS.

REFERENCES

[1] H. M. Qays, B. A. Jumaa, and A. D. Salman, "Design and implementation of autonomous quadcopter using sitl simulator," *Iraqui Journal of Computers, Communications, Control and System Engineering*, vol. 20, no. 1, pp. 1–15, 2020.

[2] R. Garcia and L. Barnes, "Multi-uav simulator utilizing x-plane," in *Selected papers from the 2nd International Symposium on UAVs, Reno, Nevada, USA June 8–10, 2009*, Springer, 2009, pp. 393–406.

[3] A. R. Perry, "The flightgear flight simulator," in *Proceedings of the USENIX Annual Technical Conference*, vol. 686, 2004.

[4] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, IEEE, vol. 3, 2004, pp. 2149–2154.

[5] Y. Jiang, J. Liao, and Q.-f. Chen, "Hardware-in-loop simulation system for small rotor uav based on jmavsim," *Computer Simulation*, 2019.

[6] A. I. Hentati, L. Krichen, M. Fourati, and L. C. Fourati, "Simulation tools, environments and frameworks for uav systems performance analysis," in *2018 14th International Wireless Communications Mobile Computing Conference (IWCMC)*, 2018, pp. 1495–1500. DOI: 10.1109/IWCMC.2018.8450505.

[7] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: High-fidelity visual and physical simulation for autonomous vehicles," in *Field and service robotics*, Springer, 2018, pp. 621–635.

[8] A. Al-Mousa, B. H. Sababha, N. Al-Madi, A. Barghouthi, and R. Younisse, "Utsim: A framework and simulator for uav air traffic integration, control, and communication," *International Journal of Advanced Robotic Systems*, vol. 16, no. 5, p. 1 729 881 419 870 937, 2019.

[9] J. M. Hoekstra and J. Ellerbroek, "Bluesky atc simulator project: An open data and open source approach," in *Proceedings of the 7th International Conference on Research in Air Transportation*, FAA/Eurocontrol USA/Europe, vol. 131, 2016, p. 132.

[10] J. Tang, F. Zhu, and L. Fan, "Simulation modelling of traffic collision avoidance system with wind disturbance," *IEEE Aerospace and Electronic Systems Magazine*, vol. 33, no. 4, pp. 36–45, 2018.

[11] O. T. Baruwa, M. A. Piera, and A. Guasch, "Timspat–reachability graph search-based optimization tool for colored petri net-based scheduling," *Computers & Industrial Engineering*, vol. 101, pp. 372–390, 2016.

[12] K. Jensen, L. M. Kristensen, and L. Wells, "Coloured petri nets and cpn tools for modelling and validation of concurrent systems," *International Journal on Software Tools for Technology Transfer*, vol. 9, no. 3, pp. 213–254, 2007.

[13] A. H. Goktogan, E. Nettleton, M. Ridley, and S. Sukkarieh, "Real time multi-uav simulator," in *2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422)*, IEEE, vol. 2, 2003, pp. 2720–2726.

[14] V. Rodriguez-Fernandez, H. D. Menéndez, and D. Camacho, "Design and development of a lightweight multi-uav simulator," in *2015 IEEE 2nd International Conference on Cybernetics (CYBCONF)*, IEEE, 2015, pp. 255–260.

[15] M. A. Day, M. R. Clement, J. D. Russo, D. Davis, and T. H. Chung, "Multi-uav software systems and simulation architecture," in *2015 International Conference on Unmanned Aircraft Systems (ICUAS)*, IEEE, 2015, pp. 426–435.

[16] J.-H. Chuang and N. Ahuja, "An analytically tractable potential field model of free space and its application in obstacle avoidance," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 28, no. 5, pp. 729–736, 1998.

[17] Y. Zhao, L. Jiao, R. Zhou, and J. Zhang, "Uav formation control with obstacle avoidance using improved artificial potential fields," in *2017 36th Chinese Control Conference (CCC)*, IEEE, 2017, pp. 6219–6224.

[18] A. C. Watts, V. G. Ambrosia, and E. A. Hinkley, "Unmanned aircraft systems in remote sensing and scientific research: Classification and considerations of use," *Remote Sensing*, vol. 4, no. 6, pp. 1671–1692, 2012.

[19] D. Martín-Lammerding. (2021). Two uas collision, https://dronetology.net/vehicular/collision.html, [Online]. Available: https : / / dronetology . net / vehicular / collision . html (visited on 01/02/2022).

[20] ——, (2021). Dense air conflicts, https://dronetology.net/vehicular/dense.html, [Online]. Available: https://dronetology.net/vehicular/dense.html (visited on 01/02/2022).

[21] Raspberry Pi Foundation. (2021). Raspberry Pi 3, https://www.raspberrypi.org/, [Online]. Available: https : / / www . raspberrypi.org/ (visited on 02/02/2021).

[22] V. Bulusu, R. Sengupta, V. Polishchuk, and L. Sedov, "Cooperative and non-cooperative uas traffic volumes," in *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*, IEEE, 2017, pp. 1673–1681.

 ISBN: 978-1-61208-974-4