A Framework for the Analysis of UAV Strategies using Co-simulation

***Abstract*—Systems using Unmanned Aerial Vehicles (UAV) are typical examples of cyber-physical systems. Designing such systems is not a trivial task because it brings the challenge of dealing with the uncertainty that is inherent to this type of system. Therefore, it is necessary the usage of appropriate tools for design that can ensure implementation of these systems with a certain level of confiability. Thus, the purpose of this work is to integrate two simulators via HLA in order to simulate and evaluate different flights strategies. For this, it is presented a simulation environment that can execute flight plans in order to evaluate different strategies in uncertain scenarios. The simulator was developed in Ptolemy and integrated with SITL/ArduPilot via HLA. With the use of the approach presented in this paper it is possible to obtain results closer to reality, thus more efficient flight strategies can be developed and evaluate.**

***Index Terms*—Co-simulation, UAV, Testing**

I. Introduction

Unmanned Aerial Vehicles (UAVs) are aircrafts capable of performing flight missions without the presence of a crew on board controlling the vehicle. Most of these vehicles are radio-controlled by a pilot on the ground. In the current context of aviation technologies, the use of this type of vehicles is in ascendancy, arousing interest from companies, institutions and individuals for many different applications [1] [2] [3]. UAVs are present in several areas, especially the military, which is applied to the monitoring of borders, attacks on enemy troops, positioning and monitoring troop movements, among other applications. In the civil context, these vehicles are being increasingly used in various applications as for photographic recording, monitoring agricultural areas, oceanography, monitoring and a myriad of possibilities of use for various purposes [4]. More recently, some UAVs are controlled by an embedded computer running applications in order to accomplish pre-programmed missions.

In recent years the computing capacity of embedded devices has increased considerably as the cost decreased, as has been predicted by Moore's Law [5]. This advance allowed the development of systems with a complexity that was not possible before, and has raised the demand for design and simulation tools following this trend. Embedded systems controlling UAVs are normally critical and complex, often part of a distributed system.

Complex embedded systems are generally heterogeneous, it means that the same model may be composed of separate modules in relation to programming languages, abstraction levels and combination of software and hardware. Therefore, the design of this kind of system requires a number of specific tools for modeling and simulation. The Ptolemy project proposes a tool that allows design and simulation of heterogeneous embedded systems in a single environment [6].

Specifically about the UAVs, there is a lack of a unique tool which allows the designer not only plan the flight, but integrate it with the specific mission and evaluate the adopted strategy in hazardous enrironments. Even though, there are many consolidated tools specialized in simulate and evaluate specific features. Thus, a solution might be the integration of different simulators in an unique framework.

Thus, the focus of this work is the integration between heterogeneous simulators to analyze different flight strategies and enable the simulation of flight Aerial Vehicles Unmanned (UAV) in an environment with the representation of uncertainty. For the integration of Ptolemy with other simulators, a standard called High Level Architecture was used, or HLA [7]. The HLA specifies an infrastructure for time management in the various simulators, called Runtime Infrastructure, or RTI. It enables the integration of heterogeneous systems synchronously and transparent manner, taking advantage of all legacy systems.

To represent the flight environment in the most real way as possible, SITL/ArduPilot was used [8], which simulates a real controller used in many UAVs, called Ardupilot. It simulates the performance of UAS flights using maps, GPS and other features that approximate the actual flight environment. In the proposed environment in this paper, on one side are the flight strategies developed using Ptolemy, and across the flight environment available in SITL/ArduPilot. The strategy commands are sent from Ptolemy to the SITL/ArduPilot by HLA which is responsible for integrating the two simulators.

The main contribution of this paper are: i) the development of a simulation framework (SITL/Ardupilot-Ptolemy) to evaluate different flight strategies; ii) use of HLA to future integration with other tools of even a physical UAV; iii) experiments as proof of concept that demonstrates the possibility to evaluate different strategies for UAVs.

We work on the assumption that the design of cyber-physical systems is a complex task and, therefore, there is no single tool that can fulfill all demands required for the design of these systems. Thus, it is necessary cooperation between different tools in a simple and synchronized manner and to make the tools and prior knowledge. As the main contribution and scientific relevance of this work should be highlighted the application simulator for UAVs deployed in Ptolemy and integration by HLA, the Ptolemy with SITL/ArduPilot for simulation of uncertainties.

The paper is organized as follows. Section II presents some

related works. In Section III the basic fundamentals of HLA are shortly described. The proposed framework is described in Section IV, followed by the experiments and results presented, in Sections V and VI, respectively. Some final considerations are described in Section VII.

II. Related Work

There are several tools able to assist the design and validation of embedded systems [6], [9], [10] and, therefore the use of these tools should not be overlooked. Nevertheless, simulation using only one tool can not always meet all the characteristics of embedded systems because these systems are often complex and heterogeneous. This issue is gaining attention from researchers.

In [11] is proposed a new approach to co-simulate hardware and software with the concept of a bridge between two simulators. In this approach the Giano and ModelSim simulators specifically integrated. Unlike our work that proposes the integration between any simulators via a consolidated standard.

It was also proposed a method based on Matlab/Simulink, which consists of modeling, simulation, verification and code generation [12]. The software codes and embedded systems prototyping can be checked step-by-step using co-simulation between Matlab and Simulink. The tool proposed in this work is based on only open-source tools and standards.

Co-simulation is also used in [13], which presents a software platform that can be used design system-level embedded system composed by multiple processors. That solution is based on the interaction of a software running in a processor model and the hardware device simulated by SystemC. This platform can perform virtual prototyping of new hardware devices, unlike our proposal, where different simulators are integrated to allow the simulation of UAV flights strategies.

In [14] is presented a collaborative approach that allows engineers from different areas the construction of individual models in the most appropriate ways, and also allows the co-simulation of these models in a common platform. The approach was performed using Crescendo[1] technology, which allows the definition and simulation of composite Discrete Event models expressed in VDM notation (Vienna Development Method) and Continuous Time models expressed using the 20-sim Framework[2]. Crescendo allows models running in different simulators, transferring data and managing simulation time. Differently, our approach relies on HLA to transfer data and manage synchronization among all simulators.

A modeling plaform for the design of cyber-physical systems is presented in [15]. A case study with Unmanned Aerial Vehicles is modeled and simulated using Ptolemy. The authors afirm that adding more detail to the physical processes would bring credibility to the project. In our work, although the flight strategies have been implemented in Ptolemy, the flight plan simulator (SITL/Ardupilot) adds details of the physical environment.

[1]http://www.crescendotool.org

[2]http://www.20sim.com

The design of systems based on UAVs need to rely on tools that are capable of delivering details of the vehicle itself and also from its interaction with the environment. In addition, there are uncertainties in the environment where the system will act to be taken into account during the design, because they somehow influence the system operation.

In this scenario, the goal of this work is to build a simulation environment where it is possible to analyze UAV flights strategies using co-simulation. The idea is to take advantage of each tool and use it in order to analyze strategies before the flight itself. For this, is performed the integration of Ptolemy [6] with SITL/ArduPilot[8] using HLA [7] as middleware.

III. High Level Architecture

The HLA is a standard of the Institute of Electrical and Electronic Engineers (IEEE), developed by Simulation Interoperability Standards Organization (SISO). Initially it was not an open standard, but it was later recognized and adopted by the Object Management Group (OMG) and IEEE.

There are several standards based on distributed computing, such as SIMNET, Distributed Interactive Simulation (DIS) , ServiceOriented Architecture (SOA) , Data Distribution Service (DDS) , HLA, among others. HLA was chosen as middleware to integrate distributed heterogeneous devices because it manages both, data and synchronization, and allows the interoperability and composition of the widest possible range of platforms.

HLA is defined by three documents: the first deals with the general framework and main rules [16], the second concerns the specification of the interface between the simulator and the HLA [17] and the third is the model for data specification (OMT) transferred between the simulators [18].

The main HLA characteristics are defined under the leadership of Defence Modelling and Simulation Office (DMSO) to support reuse and interoperability. Interoperability is a term that covers more than just send and receive data, it also allows multiple systems to work together. However, the systems must operate in such a way that they can achieve a goal together through collaboration.

In HLA architecture (see Figure 1), the set of various interoperating systems within a domain is called Federation. Each member of a federation is called Federate [16] . The Federates are registered and managed through a Runtime Infrastructure (RTI) , as can be shown in Figure 1.

Each Federate is locally associated with a RTI Ambassador (RTIA) process via TCP socket. Messages among RTIA and RTI Gateway (RTIG) are exchanged through a TCP/IP network protocol in order to perform the RTI services in a distributed manner. The RTIG is the central point in the architecture. It manages the data exchanging and synchronization among all Federates in a Federation.

IV. A Framework for Analysis of Strategies for UAVs

The proposed simulation framework consists of two parts (as presented in Figure 2), one part is responsible for representing the flight environment (using SITL/Ardupilot), and

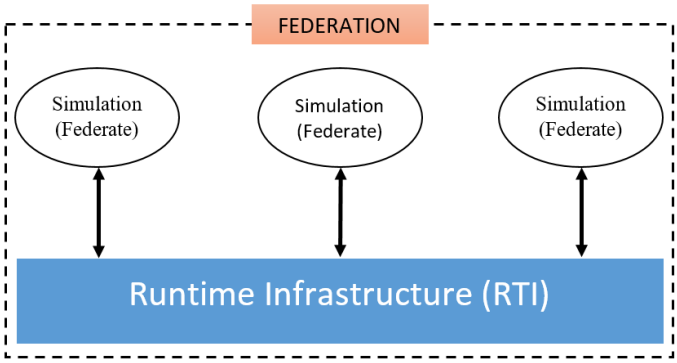


Fig. 1: Architecture of a Federation.

the other part is responsible for the definition of the flight strategy that will be executed by the UAV (using Ptolemy). Synchronization and communication between the parts is made using High-Level Architecture - HLA.

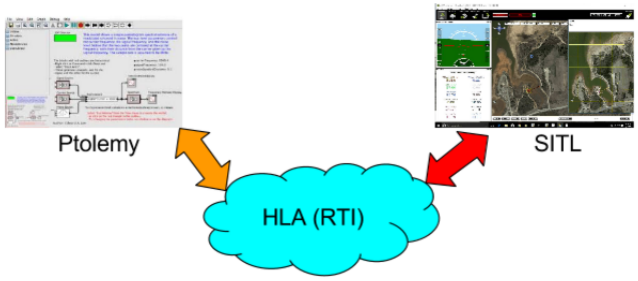


Fig. 2: General architecture of the framework.

This integration between the simulators follows the idea that each simulator is a Federate in the simulation, which is responsible for sending and receiving data. The Federate running on SITL/Ardupilot was developed using the PyHLA library [19] and Federate on Ptolemy was made using the JCerti [20].

In Ptolemy, the communication with the HLA is performed by actors shown in Figure 4: *SlaveHLADirector* in green and *SlaveFederateActor* in black. The latter implements the Federated this module, more details are presented in [21]. *SlaveHLADirector* is responsible to manage all simulation, mainly the time management in order to keep Ptolemy synchronized with all other Federates, while *SlaveHLADirector* is the actor that receives data from HLA and transfers to other actors and the opposite, receives data from actors and transmits to HLA.

Figure 3 shows the module responsible for representing the flight environment in SITL/Ardupilot. It is possible to see that it uses satellite image and maps (from Google Maps) and produces accurate values of UAV compared to real flight environments, like speed, distances and power consumption.

The module responsible for setting a strategy can be seen in Figure 4, where are presented the actors *StrategyA* and *StrategyB*. The model is configured to use the connected strategy during the flight simulation. In this case, the *StrategyA* is in

Fig. 3: Flight environment running on SITL/Ardupilot

use but could be easily replaced by *StrategyB*, as presented in Section V.

Although Figure 4 presents only two actors strategies, others may be added through the creation of new actors that implement them. The implementation of a new strategy is done by creating a new actor in Ptolemy using Java language. To create a new actor it must be created a new class in the Java language that inherits a *TypedAtomicActor* (or other existing Ptolemy actor). After that, input and output must be defined. Finally the *fire()* method of the inherited super class *Actor* should be overwritten with the implementation of the new strategy. This method is responsible for reading the information from the input ports, the execute the strategy and send the action to be executed through output ports. The *fire()* method of each actor is invoked by Director based on relations among actors, resulted from a schedule algorithm (in our case, Discret Event).

For the experiments, a surveillance scenarios was chosen. In this scenario, a list of location points are passed to the UAV, which should visit all of them continously. The UAV must take a picture of the location every time it flight over it. None location can stay more than a established time without be visited. Also, the UAV must never flight out of battery under risk to fall down. Thus, the strategy must define battery level as high priority requisite.

In *StrategyA* the UAV visits the target points in the order they are registered, regardless of the distance between the current position of the UAV and the target location. This means that even if the first registered location is far from the current UAV position, and there is another point nearest to be visited, the UAV will not take this under consideration and will visit initial programmed location.

Differently, in *StrategyB* the UAV visits the target points not considering the order they were registered, but the distance between the current position of the UAV and the target point. This means that the closest point to UAV will be visited first, followed by the other points ordered by the distance to UAV at each instant.

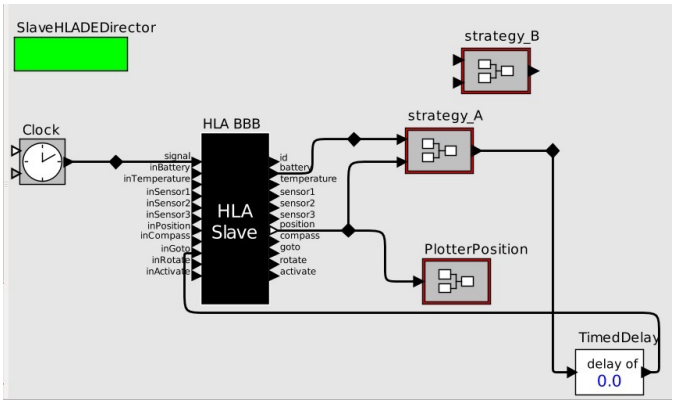


Fig. 4: Ptolemy model configured with StrategyA

Still in Ptolemy, the model of UAV (presented in Figure 4) is encapsulated inside another actor presented in Figure 5. In this figure the UAV is represented by three circles. At each corner, four developed actors were added to cause interference in the UAV movement. They emulate the influence of wind. When the UAV approaches one of them, it sends a message (a number) to UAV, which add this number to intensity of its movement. At the end, the UAV may move unexpectedly always when flying besides one of these actors. The Wind Actors are configured to generate interference in 15% of the cases. Thus, it is possible to simulate how efficient a flight strategy can be even when there are uncertainties.

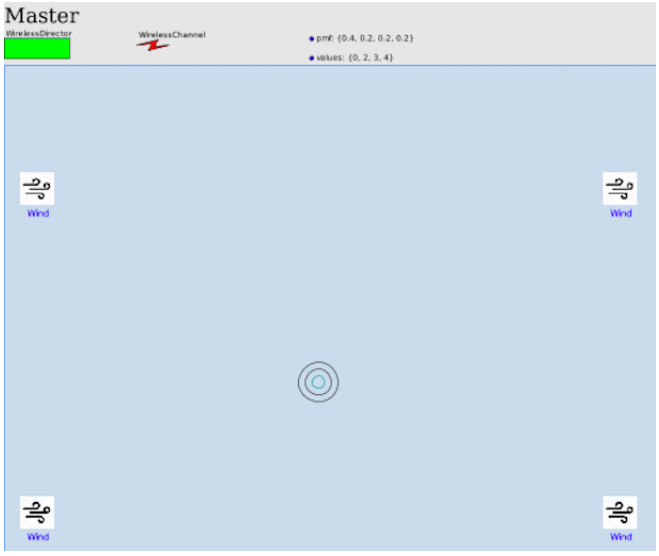


Fig. 5: UAV in Ptolemy with four actors emulating wind effect.

A. Data modeling

When using HLA, the Federates exchange data in the form of objects defined following the Object Model Template (OMT) from HLA [18], which is specified in an specific file common to all Federation and present at each machine.

As presented in Figure 4, the actor used was one presented in [21]. The HLASlave actor has a port for each possible data to be exchanged via HLA. In our approach the ports dedicated to transfer the ID of the UAV (for future usage of multiple UAVs), battery level and position were used for data exchanging, plus one port for sending commands (called ”goto”). Through this last port the strategy actor sends to the UAV which movement it should make at each instant. The specification of the data model can be seen in Code 1.

Code 1: Data model used by HLA

```
1  (FED
2    (Federation TestFed)
3    (FEDversion v1.3)
4    (spaces)
5    (objects
6     (class ObjectRoot
7       (attribute privilegeToDelete reliable
            timestamp)
8       (class RTIprivate)
9       (class robot
10       (attribute id reliable timestamp)
11       (attribute battery reliable timestamp)
12       (attribute position reliable timestamp)
13       (attribute goto reliable timestamp)))
14   )
15 )
```

V. Experiments

In order to evaluate which strategy is more efficient, the two strategies were executed in two different scenarios, one without interference, and the other under influence of wind. The same four points to be visited were set for all scenarios, as well as an initial point for takeoff and landing of UAV.

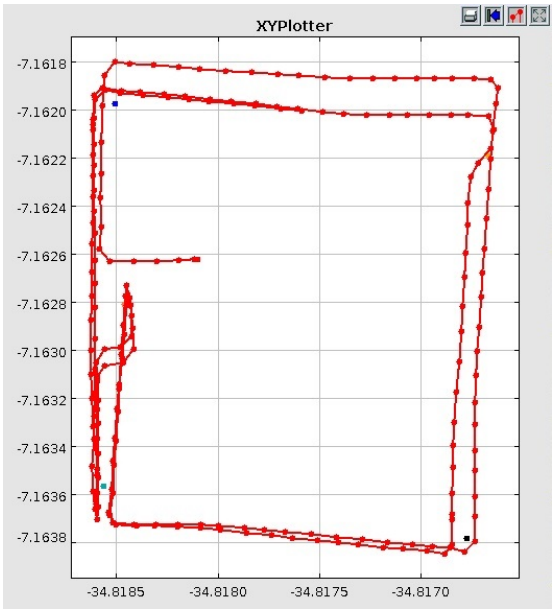
Figure 6 shows the trajectory traveled by the UAV running StrategyA in SITL/Ardupilot and received by Ptolemy. In Figure 6a can be seen the trajectory of the UAV without the occurrence of environmental interference and in Figure 6b with presence of wind. Similarly, Figure 7 shows the trajectory of the UAV running StrategyB in both situations, without environmental interference (Figure 7a) and with presence of wind (Figure 7b). Apparently, StrategyA results a more homogeneous trajectory than StrategyB. In next section the quantitative data from each strategy are analyzed.

VI. Results

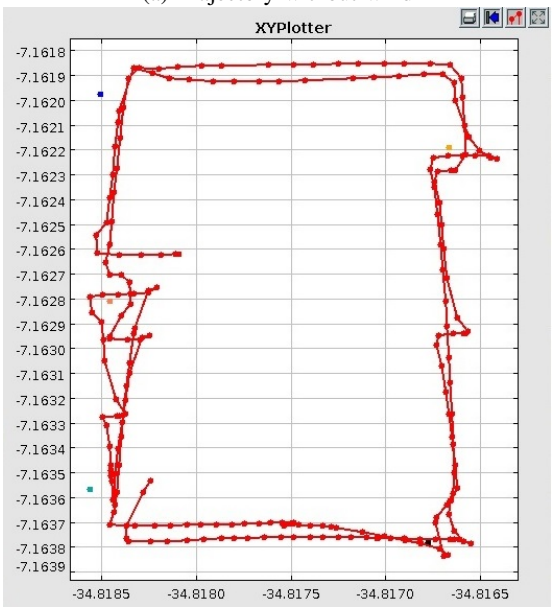
The main data related to the UAV flights were collected during each simulation in order to compare how the strategies behaved in each situation. In Table I it is possible see the distances in meters travelled by the UAV. It is possible to see that in the simulations without wind the distance was greater in both strategies, which means the UAV was more efficient when does not suffered external influence. Also, without wind, the UAV with *StrategyA* traveled greater distances than *StrategyB*. However, in presence of wind it was the opposite, *StrategyB* was more efficient.

TABLE I: Traveled distances in meters

Strategy	Without wind	With wind
A	2332.02	1995.03
B	2190.58	2047.99

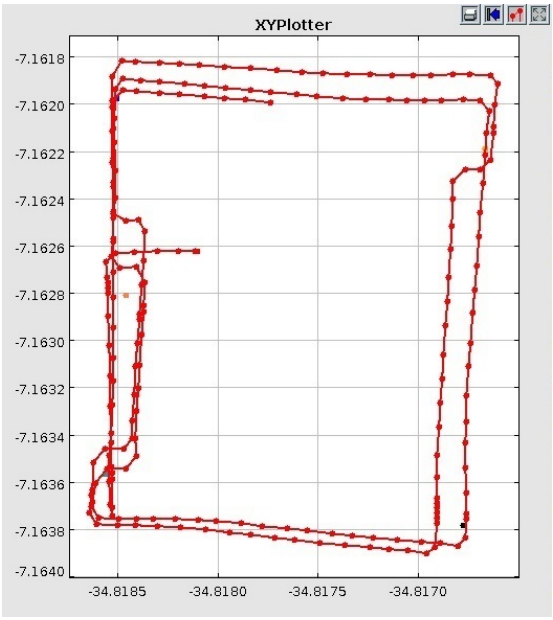


(a) Trajectory without wind

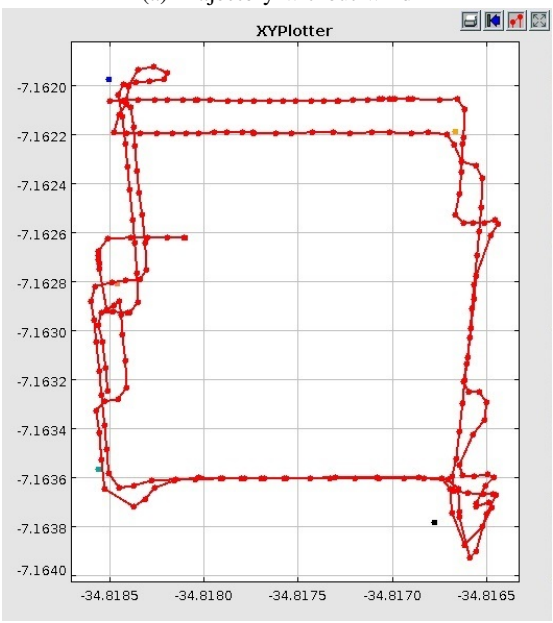


(b) Trajectory with wind

Fig. 6: Trajectory of UAV running StrategyA



(a) Trajectory without wind



(b) Trajectory with wind

Fig. 7: Trajectory of UAV running StrategyB

In addition to the distance traveled criteria, it was also analyzed which strategy resulted in a greater amount of obtained pictures. The more captured photos, the better is the coverage of the surveillance system.

The data in Table II show that the number of pictures captured in *StrategyB* was higher compared to *StrategyA*, and this occurred even without the occurrence of wind. Despite *StrategyA* be more efficient with respect to distance traveled, *StrategyB* was more efficient regarding the amount of captured photos.

The last (but not least important) criteria evaluated is the remaining battery level after simulation. In both strategies, the UAV always evaluate if its actual battery level is enough to visit the next point and flight back to landing point or not. In negative case, the UAV will stop the mission and land on the initial point. Thus, a riskier strategy is that one that finishes the mission with very low battery level. Table III shows these data in percentage. Both strategies may be considerated

TABLE II: Captured photos

Strategy	Withou wind	With wind
A	11	10
B	14	11

conservative, finishing the simulation with a reasonable charge level.

In general, it is possible to see that *StrategyB* was more efficient than *StrategyA* because it allows the UAV take more pictures, even when travelling less than with *StrategyA* and, at the end, higher or equivalent battery level remains with UAV.

TABLE III: Battery level after simulation (%)

Strategy	Withou wind	With wind
A	37	32
B	42	31

After the experiments, it can be seen that the proposed environment enables simulation of UAVs using different strategies on each flight. This allows the designer to make the analysis of various strategies and select the most suitable for each context.

VII. Final considerations

This paper presented an environment to assist design of critical Cyber-Physical Systems, specifically to validate flight strategies for UAV systems. In the developed simulator it was possible to perform simulations of a model with the representation external interfereces.

Using co-simulation, it was possible to obtain results closer to reality, thus more efficient and safe strategies can be developed and tested. This approach follows the idea that complex systems can be better modeled and tested when integrating different simulators, joining the best of the worlds in an unique environment.

The environment is formed by the integration of two simulators. One is responsible for the configuration of the flight strategies (Ptolemy), the other is responsible for representing the flight environment and the telemetry of the UAV (SITIL/ArduPilot). Communication between the simulators was made using High-level Architecture (HLA).

As further work, new strategies with other flight algorithms should be implemented to compose a library of strategies ready for use in future simulations. Furthermore, it is expected that some of these new strategies take into account the scenarios where multiple UAVs work together in specific missions. Also, Monte Carlo simulations will be executed in order to improve the confidence on the results.

References

[1] C. Luo, S. McClean, G. Parr, L. Teacy, and R. De Nardi, "UAV Position Estimation and Collision Avoidance Using the Extended Kalman Filter," *IEEE Transactions on Vehicular Technology*, vol. 62, no. 6, pp. 2749–2762, July 2013.

[2] T. Lam, H. Boschloo, M. Mulder, and M. van Paassen, "Artificial Force Field for Haptic Feedback in UAV Teleoperation," *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 39, no. 6, pp. 1316–1330, Nov 2009.

[3] W. Teacy, J. Nie, S. McClean, and G. Parr, "Maintaining connectivity in UAV swarm sensing," in *IEEE GLOBECOM Workshops (GC Wkshps)*, Dec 2010, pp. 1771–1776.

[4] H. Chen, X. m. Wang, and Y. Li, "A Survey of Autonomous Control for UAV," in *International Conference on Artificial Intelligence and Computational Intelligence*, vol. 2, Nov 2009, pp. 267–271.

[5] G. Moore, "Cramming more components onto integrated circuits," *Proceedings of the IEEE*, vol. 86, no. 1, pp. 82–85, Jan 1998.

[6] C. Ptolemaeus, Ed., *System Design, Modeling, and Simulation using Ptolemy II*. Ptolemy.org, 2014. [Online]. Available: http://ptolemy.org/books/Systems

[7] "IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA)– Federate Interface Specification," *IEEE Std 1516.1-2010 (Revision of IEEE Std 1516.1-2000)*, pp. 1–378, Aug 2010.

[8] (2016) SITL/Ardupilot Simulator (Software in the Loop) . [Online]. Available: http://ardupilot.org/dev/docs/sitl-simulator-software-in-the-loop.html

[9] A. Forin, B. Neekzad, and N. L. Lynch, "Giano: The two-headed system simulator," Microsoft Research, Tech. Rep. MSR-TR-2006-130, September 2006. [Online]. Available: http://research.microsoft.com/apps/pubs/default.aspx?id=70343

[10] Accellera, "The language for system-level modeling, design and verification," Accellera, Tech. Rep., October 2015. [Online]. Available: http://accellera.org/community/systemc/about-systemc

[11] P. H. Cheung, K. Hao, and F. Xie, "Component-based hardware/software co-simulation," in *Digital System Design Architectures, Methods and Tools, 2007. DSD 2007. 10th Euromicro Conference on*, Aug 2007, pp. 265–270.

[12] C. Ren, Y. Huang, H. Chen, and G. Tian, "Control software development of drive motor for electric vehicles," in *Transportation Electrification Asia-Pacific (ITEC Asia-Pacific), 2014 IEEE Conference and Expo*, Aug 2014, pp. 1–6.

[13] Y.-T. Hsu, Y.-J. Wen, and S.-D. Wang, "Embedded hardware/software design and cosimulation using user mode linux and systemc," in *Parallel Processing Workshops, 2007. ICPPW 2007. International Conference on*, Sept 2007, pp. 17–17.

[14] J. Fitzgerald, K. Pierce, and P. Larsen, "Co-modelling and co-simulation in the engineering of systems of cyber-physical systems," in *System of Systems Engineering (SOSE), 2014 9th International Conference on*, June 2014, pp. 67–72.

[15] A. Kanduri, A. M. Rahmani, P. Liljeberg, K. Wan, K. L. Man, and J. Plosila, "A multicore approach to model-based analysis and design of cyber-physical systems," in *SoC Design Conference (ISOCC), 2013 International*, Nov 2013, pp. 278–281.

[16] "IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA)– Framework and Rules," *IEEE Std 1516-2010 (Revision of IEEE Std 1516-2000)*, pp. 1–38, Aug 2010.

[17] "IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA)– Federate Interface Specification," *IEEE Std 1516.1-2010 (Revision of IEEE Std 1516.1-2000)*, pp. 1–378, Aug 2010.

[18] "IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA)– Object Model Template (OMT) Specification," *IEEE Std 1516.2-2010 (Revision of IEEE Std 1516.2-2000)*, pp. 1–110, Aug 2010.

[19] (2016) Pyhla — python bindings for m&s hla. [Online]. Available: http://www.nongnu.org/certi/PyHLA

[20] (2016) Certi - summary. [Online]. Available: http://savannah.nongnu.org/projects/certi

[21] A. L. V. d. Negreiros and A. V. Brito, "The development of a methodology with a tool support to the distributed simulation of heterogeneous and complexes embedded systems," in *Brazilian Symposium on Computing System Engineering (SBESC)*, Nov 2012, pp. 37–42.