HARDWARE-IN-THE-LOOP SIMULATION FRAMEWORK FOR THE EVALUATION OF AUTONOMOUS FLIGHT FUNCTIONS

David Pla Guerrero[1], Anoosh Hegde[1], Jia Wan[2], Elias Allegaert[1] & Yves Lemmens[1]

[1] Siemens Digital Industries Software, Interleuvenlaan 68, 3001 Leuven, Belgium
[2] Flanders Make, Gaston Geenslaan 8, 3001 Leuven, Belgium

Abstract

This paper presents a hardware-in-the-loop (HiL) simulation framework for unmanned aerial vehicles (UAV) to validate the simulation models and the performance of autonomous algorithms under realistic flight conditions. A real-time simulation of both manual and autonomous flights is completed by coupling the modelled aircraft subsystems and sensors in Simcenter Amesim and Simcenter Prescan with a Pixhawk flight controller. The recordings of the camera sensor modelled along the flight path were be forwarded to a visual simultaneous localization and mapping (vSLAM) algorithm.

Keywords: UAV; HiL; SLAM; Siemens Simcenter; flight controller

1. Introduction

The increased capability of unmanned aerial vehicles (UAV) has enabled their introduction in diverse applications such as construction, aerial imaging, search and rescue operations, and parcel delivery [1]. This diverse integration of UAVs underscores the critical need for robust simulation frameworks that ensure their operational safety and reliability. Consequently, there is an increasing dependency on sophisticated simulation environments that bridge the gap between virtual testing and real-world operations.

Hardware-in-the-loop (HiL) simulations represent a significant advancement over traditional software-in-the-loop (SiL) methods by incorporating actual hardware components into the simulation process, thereby providing a more accurate assessment of an UAV's performance under controlled yet realistic conditions. This paper introduces a comprehensive HiL simulation framework leveraging Siemens Simcenter software, designed to evaluate the integration of hardware and software components critical for autonomous navigation. This study demonstrates the application of Siemens Simcenter Prescan and Simcenter Amesim tools in evaluating the Simultaneous Localization and Mapping (SLAM) capabilities during the development phase of autonomous UAM vehicles, thereby ensuring their reliability and effectiveness in real-world scenarios [2].

One of the paramount challenges in deploying these vehicles is achieving precise navigation under varying environmental conditions where traditional navigation systems, such as GPS, may fail or be unavailable [3]. The motivation behind this research stems from the critical need for autonomous vehicles to possess robust position estimation capabilities to maintain accurate localization and navigation without relying solely on GPS [4]. Specifically, we focus on a Pixhawk flight controller and the implementation of a visual Simultaneous Localization and Mapping (vSLAM) algorithm. vSLAM is instrumental in enabling vehicles to autonomously determine their location and navigate effectively using camera imagery, thereby constructing a real-time 3D map of their surroundings. It is particularly well-suited for vehicles operating in urban environments where GPS signals are often obstructed [5][6].

2. HiL framework

2.1 Overview

The simulation framework outlined in this paper, see Figure 1, is an integral component of our study, designed to accurately mimic the operational environment and functionalities of an UAV through HiL simulations. This framework comprises several key hardware and software elements configured to work in synergy for a comprehensive evaluation of UAV systems.

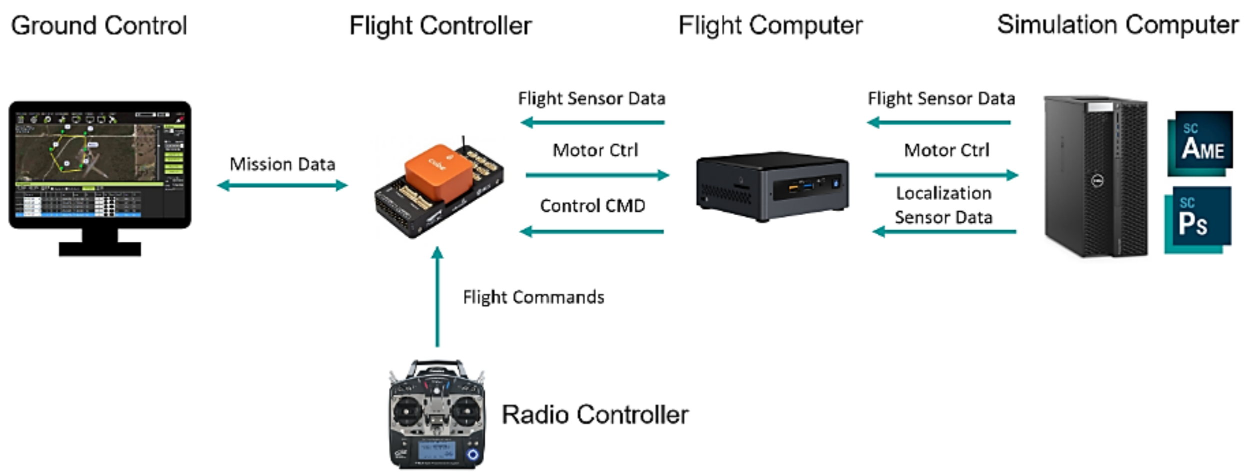

Figure 1: HiL simulation framework overview.

Hardware Components:

- Pixhawk Flight Controller: This controls the motors of the UAV to follow the required trajectory. It runs the PX4 autopilot software. In this HiL setup the motor controls are sent to the flight dynamics model in Simcenter Amesim. The data from built-in inertial measurement unit, barometer, magnetometer, and the GPS signal are replaced by simulated data.
- Onboard Flight Computer: The main purpose is the process the camera data with the SLAM algorithm to provide alternative attitude information. In the HiL setup, it also serves as an interface between the simulation environment and the Pixhawk flight controller.
- Simulation Computer: This runs the modeled aircraft subsystems and sensors using Siemens Simcenter software. This includes the simulation of flight dynamics, power systems, and propeller aerodynamics, which are modeled in Simcenter Amesim.
- Radio Controller: This controls the Pixhawk and is required to initiate the flight and is used to control the UAV manually during flight.
- Ground Control Laptop: This run the QGroundcontrol software that can be used to provide waypoints for autonomous flights and can be used to monitoring the aircraft data during flight.

At the onset of each simulation run, the flight controller is configured in HiL mode, effectively disabling its onboard sensors since their outputs are simulated within the framework. The ground control laptop dispatches the flight plan to the flight controller, initiating the simulation process. Communication between the flight computer and the simulation computer is established through ROS [7]. Widely used in the field of robotics, ROS is an open-source software communication framework with a collection of tools, libraries, and conventions that aim to simplify the task of creating complex and robust robotic software across a wide variety of robotic platforms [8]. A physics model of the UAV is running on the simulation computer. Sensor information (position, acceleration and GPS location) coming from the Amesim model, are fed to the flight controller which computes the necessary motor throttle to navigate the UAV according to the desired trajectory. These commands are communicated back to Amesim which provides the new states of the UAV. This creates a continuous loop that simulates real-time flight dynamics. To evaluate the vSLAM algorithm, a virtual environment was created in Simcenter Prescan which simulated the camera images that the vSLAM algorithm would rely on to estimate the position and attitude of the UAV.

2.2 Flight physics simulation in Simcenter Amesim

Within the comprehensive simulation framework employed in this study, Simcenter Amesim is utilized to simulate the physical behavior of the UAV that was built by our collaborators at the Flanders Make research institute. This model, see Figure 2, is meticulously designed to simulate the actual flight dynamics of the UAV based on empirical data gathered through rigorous testing. From a CAD model of the UAV, see Figure 3, the inertia and exact location of the propellers and landing pads were determined. The 6 DOF inertial model will keep its parameters constant during the whole simulation.

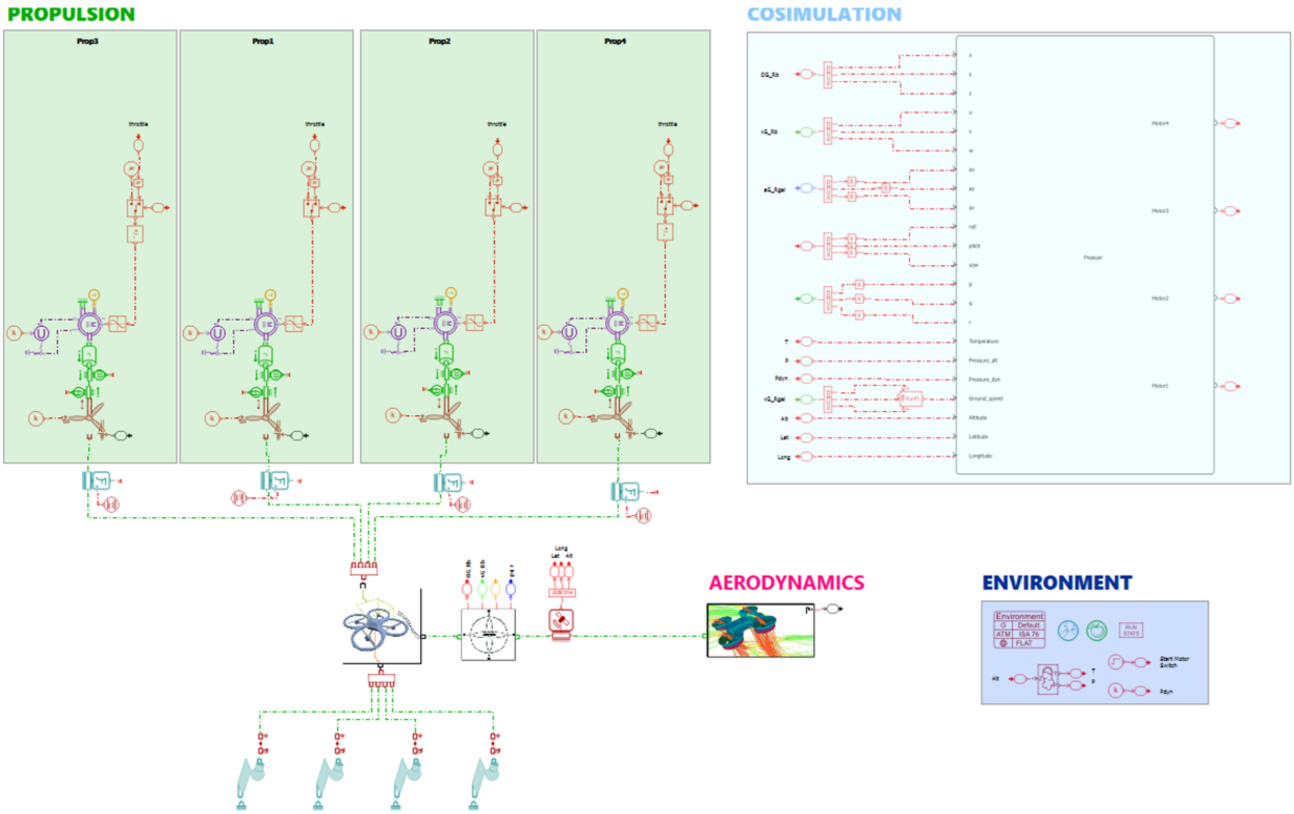

Figure 2: Simcenter Amesim UAV model.

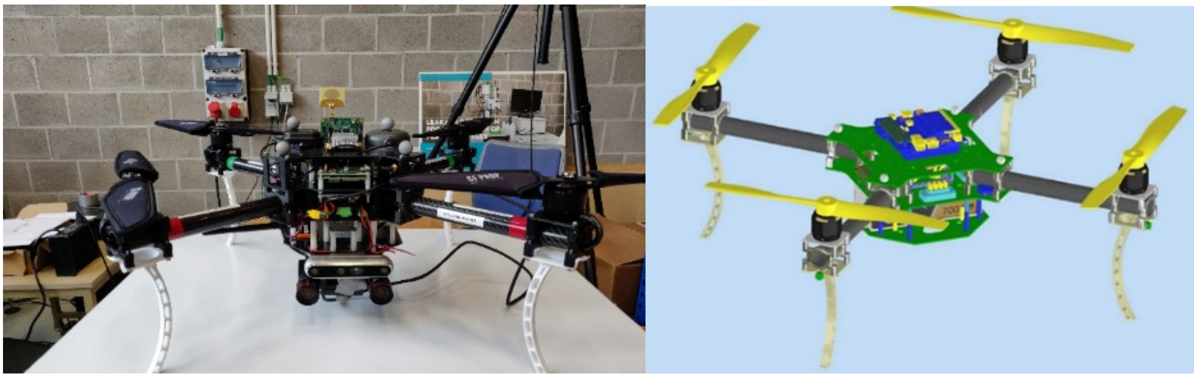

Figure 3: (left) picture of the Flanders Make UAV and (right) a CAD Model of it.

2.2.1 Propulsion system

The propulsion system of the UAV is characterized by detailed testing of its motors and propellers. These tests determine the thrust and power outputs as functions of the propellers' revolutions per minute [9]. The propellers receive throttle inputs directly from the flight controller, though their operation is constrained by the saturation limits of the motors to prevent over-exertion, which could lead to mechanical failures or inefficiencies.

The battery model used in the simulation maintains a constant voltage input of 22 volts, ensuring a stable power supply to the UAV's systems during flight operations. The motor model represents an averaged version of an electric drive system, characterized by its efficiency, maximum torque, and power output. These characteristics are derived from the results of the experiments conducted, allowing for a realistic simulation of the UAV's performance under various operational conditions.

2.2.2 Body aerodynamics

The aerodynamic behavior of the UAV is captured using a 6 Degrees of Freedom (6DOF) linearized model, which provides a simplified yet effective representation of the UAV's response to control inputs and external forces. A 6DOF linearized system for aerodynamics refers to the mathematical modeling of an aircraft's motion, considering its six fundamental modes of movement: three translational and three rotational. This linearized system simplifies the analysis of an aircraft's dynamics by approximating the equations of motion around a steady flight condition, such as straight and level flight. The linearization allows for easier control system design and stability analysis.
This model is sensitive to changes in wind magnitude and the incidence angle of wind, essential for assessing the UAV's stability and handling under different weather conditions. Furthermore, while models for wind gusts and turbulence were initially incorporated to enhance the simulation's realism, they have been deactivated to maintain the simulation's performance at real-time speed, ensuring that the computational load remains manageable without sacrificing the accuracy needed for effective analysis and decision-making. The wind velocity relative to the body calculated here is transmitted to the propellor models.

2.2.3 Co-simulation block

The co-simulation block will specify the data exchange with the other components in the framework. In this manner, on the right side of the block Simcenter Amesim receives from the Pixhawk flight controller the four throttle inputs to the motor models. On its left side (in the co-simulation block in Figure 2) are determined the states computed in Simcenter Amesim, based on the commanded throttle, to be sent through the framework. The co-simulation block communicates with the Simcenter Prescan model via the C++ interface as was demonstrated in previous work [10]. The communication frequency of the Amesim model with the flight computer is 10 Hz.

2.3 Sensor simulation in Simcenter Prescan

The urban environment was simulated in Simcenter Prescan and the models utilized in the simulations are high-fidelity representations of urban areas in Toulouse and London. Figure 4 shows the London model in Simcenter Prescan GUI and Simcenter Prescan Viewer. These models were developed in Blender, leveraging satellite imagery from OpenStreetMaps to accurately capture the urban infrastructure and ground imagery [11]. Once created, the models were imported into Simcenter Prescan utilizing its Model Preparation Tool. This process ensures that the simulations incorporate detailed models of actual environments, thereby facilitating the generation of realistic camera imagery and enabling autonomous navigation within the same settings as those defined in the QGroundcontrol software [12].

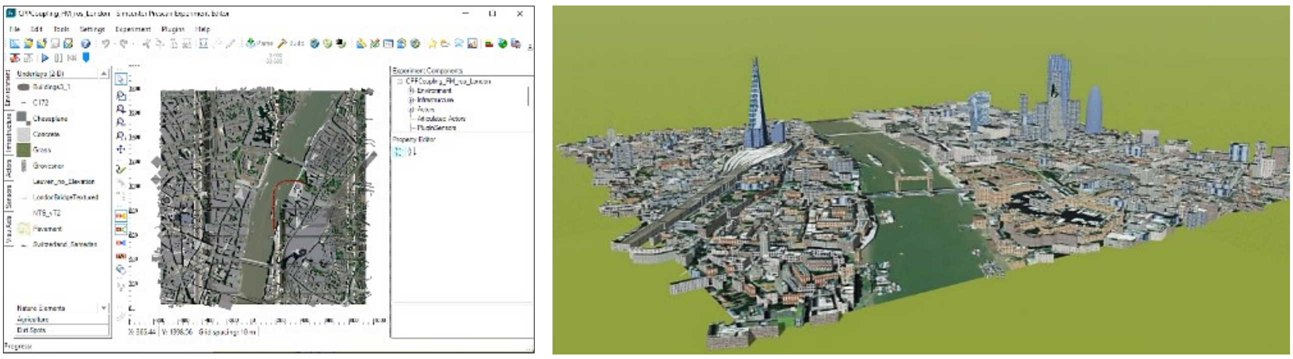

Figure 4: (left) a screenshot in Simcenter Prescan GUI and (right) the rendered 3D London City model in Simcenter Prescan Viewer.

Regarding the camera sensor modelling, two cameras were attached at the front of the craft with a distance between them of 0.1m to represent the operation of a stereo camera. The camera models operate at a frequency of 10Hz and produce a grayscale output with a resolution of 1280x960 pixels. The far clipping distance i.e., the distance until which the objects are rendered, is 500 meters.

3. The vSLAM algorithm

SLAM is a technology initially developed for robot autonomy, now extended to applications such as 3D modeling, augmented reality (AR) visualization, and autonomous driving and flying. Initially utilizing a mix of sensors including lasers, GPS, and cameras, recent advancements have focused on vSLAM, which relies solely on camera inputs, simplifying sensor configurations while increasing technical challenges. vSLAM is effective in situations demanding real-time processing to merge real and virtual objects. These algorithms are not only pivotal for UAVs but also in augmented reality (AR) [13].

There are many vSLAM methods and they can be categorized into different ways. One categorization is feature-based vs. direct methods. Feature-based methods involves detecting distinct features (like edges) in the environment from images captured by a camera. These features are extracted using algorithms. Since the methods relies on specific features, they can be robust against changes in lighting or minor blurring, as these features can often still be recognized under different conditions. Feature detection and matching can be computationally intensive and may struggle in environments where features are sparse or repetitive. Also, the extraction of features may fail in low-texture areas. Since only features need to be stored, data management is simple. The second category, direct methods, use the intensity values of all pixels in the image directly, rather than extracting features. This method aligns images based on the minimization of photometric error, i.e., the difference in pixel intensity between successive images. By using all the pixel data, direct methods can be beneficial in texture-rich indoor environments. These methods can be faster in terms of execution since they skip the feature detection and extraction phase. However, they might require more computational resources because. Direct methods can also be more sensitive to changes in lighting and exposure, as these affect the pixel intensities significantly. They might also struggle in high dynamic range scenarios or where there is significant camera motion causing motion blur [14].

The SLAM algorithm used for the presented HiL setup is RTAB-map (Real-Time Appearance-Based Mapping) [15]. It is a graph-based vSLAM technique, optimized for large-scale and long-term environments, utilizing RGB-D, stereo cameras, or LiDAR inputs. It features an incremental bag-of-words approach for efficient visual loop closure detection, enhancing accuracy by comparing new images against key frames and a feature map to identify and refine the robot's location. It can therefore be counted to the feature-based algorithms. RTAB-Map incorporates a memory management system to handle real-time constraints and maintain performance in extensive environments, limiting the number of features stored to ensure efficient processing. The software is compatible with ROS, benefiting from a supportive scientific community and continual updates.

4. Simulation results

Two different kinds of HiL simulation were executed to test the capabilities of the proposed framework. First, the steering of the aircraft was performed manually, having a pilot flying a trajectory using the radio controller. Second, a more complex trajectory was specified on the ground control laptop and sent to the flight controller, which must fly the trajectory autonomously, without the need of any pilot input.

In the simulations, the aircraft's flight path was recorded and subsequently replayed offline in Simcenter Prescan to simulate various environmental conditions to examine the impact of weather on the pose estimation capabilities of the vSLAM algorithm. This procedure ensured that the trajectory itself did not interfere with the performance of the vSLAM algorithm.

4.1 Manual flight

The manual flight takes place in the Toulouse environment where a flight path was flown and recorded. To obtain real-time performance, the generation of camera images was done off-line by replaying the trajectory in Simcenter Prescan. Figure 5 shows the Toulouse environment near the international airport of Toulouse-Blagnac.

Figure 5: Simcenter Prescan Viewer Toulouse City model with 3D buildings.

The camera sensor generated images that were directly forwarded to the vSLAM algorithm, which calculated the pose estimation based on the features detected in the environment. This was done for various weather conditions. Figure 6 shows two of such frames.

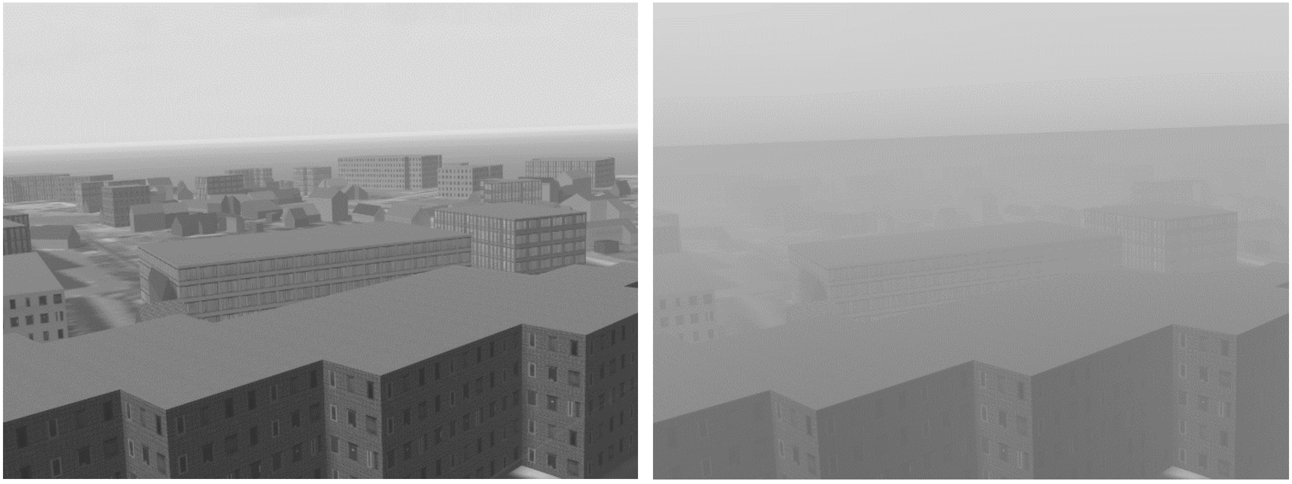

Figure 6: Camera output during light rain (left) and fog (right) in Toulouse environment simulation

Figure 7 presents the results of the vSLAM pose estimation algorithm for various simulation scenarios—clear weather, light rain, fog, and the presence of another UAV—compared against the ground truth (depicted as a black line). The plot illustrates that the pose estimation under clear weather conditions (represented in blue) yields the most accurate outcomes, as these conditions are optimal for vSLAM's environmental feature detection. Following in terms of performance is the scenario involving a moving obstacle. The observed additional drift to the right in this case is attributed to the movement of the obstacle, which is not accounted for by the vSLAM algorithm, given its assumption that all objects remain static.

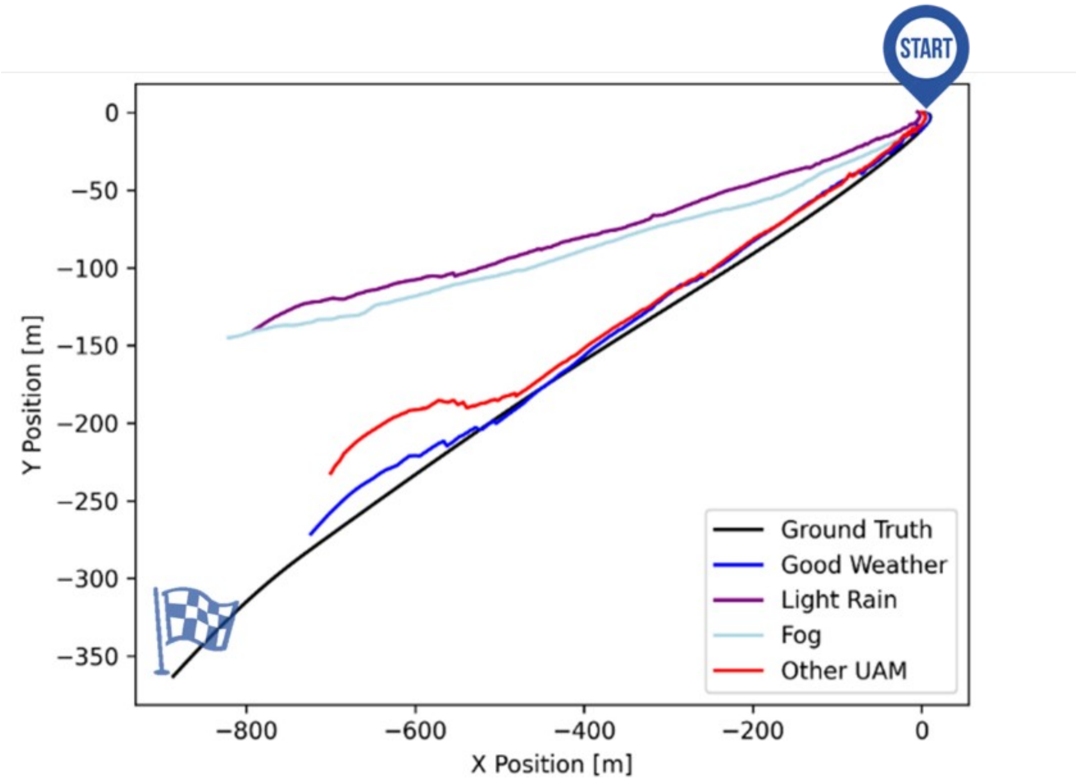

Figure 7: vSLAM position estimation for multiple simulation conditions versus ground truth in the Toulouse environment.

The rain and fog weather simulations, present the most deviation. After takeoff, these two cases deviate from the actual trajectory. This fact could be explained due to its reduced perception of the environment with higher angular rates suffered during this flight phase. However, it is seen that in both cases the estimations keep a straight trajectory and in the rain simulation it slightly captures the small curvature performed at the end of the ground truth trajectory.

4.2 Autonomous flight

The autonomous flight simulation was performed in a section of the City of London. The flight path is created via the ground control laptop which runs the QGroundControl software. Figure 8 shows a screenshot of the ground control laptop. The trajectory (in orange) is specified for the aircraft to fly around the Tower of London. A series of waypoints are selected and straight connections with a specific velocity are created between them. This flight plan was sent to the flight controller at the start of the simulation and it will try to steer the vehicle by controlling the motor inputs following this specific path.

The takeoff position and the landing point are coincident and marked as an enlarged green waypoint with the number 7. The resulting trajectory of the simulation is shown in white. One can see how the aircraft quite accurately followed the desired trajectory except for some small deviations at waypoints 4 and 6.

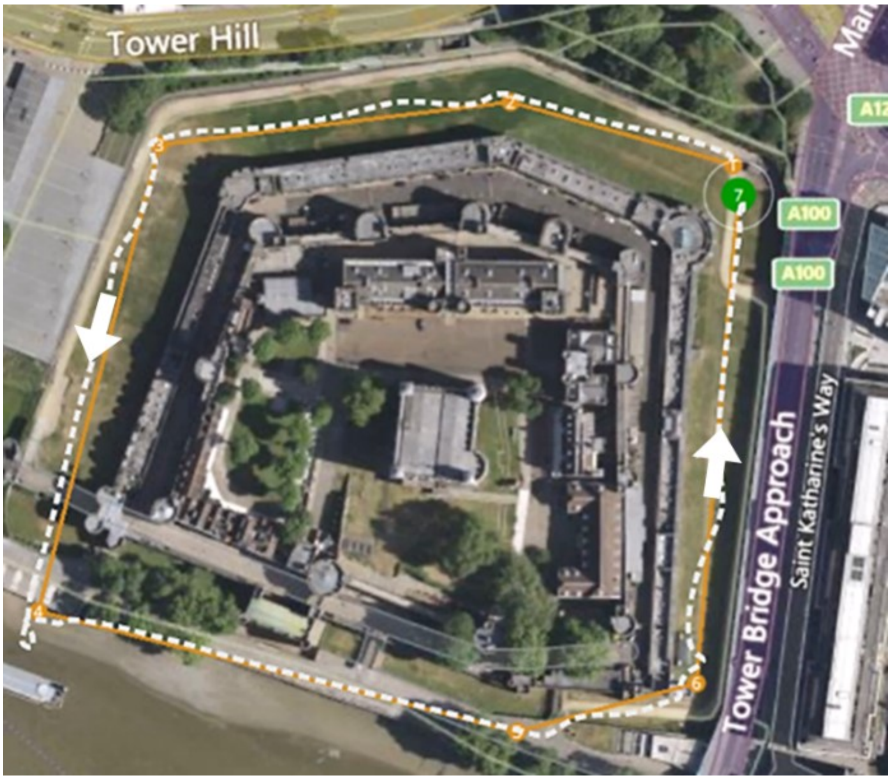

Figure 8: Trajectory waypoints specified in QGroundControl (orange) and overlayed the path followed in simulation (white).

As previously noted, the ground truth trajectory is utilized to generate the camera feeds offline for different weather conditions. Figure 9 illustrates the results of the vSLAM pose estimation conducted in good weather conditions within the City of London environment. The complex trajectory, depicted in black, poses a greater challenge for the pose estimation compared to that of the Toulouse environment. Notably, during takeoff, when the aircraft experiences high angular rates, there is already an error in the position estimate. However, once the initial drift is discounted, the vSLAM algorithm demonstrates considerable accuracy in capturing the aircraft's movement. It correctly captures a straight movement with a slight curve around the middle of that last path but ends in a position deviated 20 meters from the ground truth.

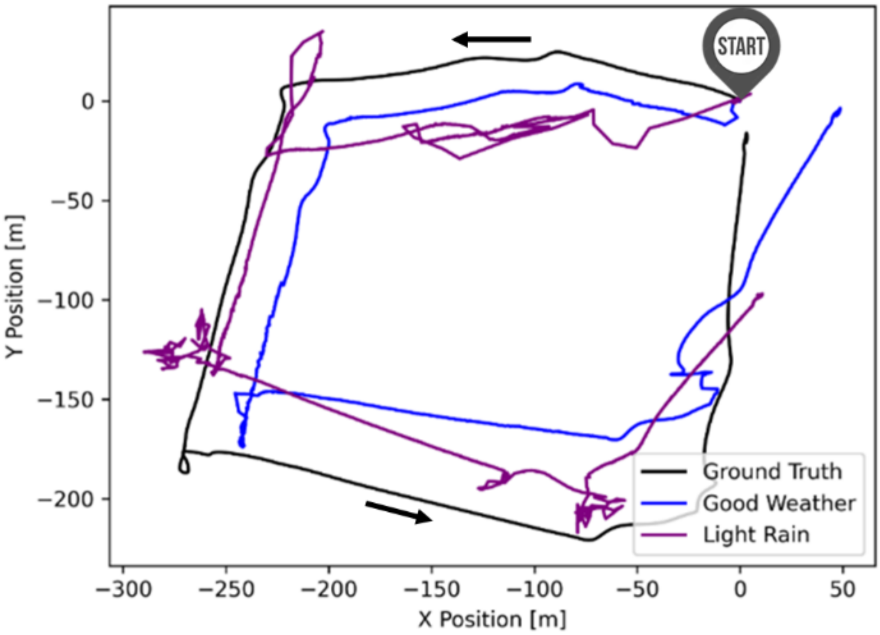

Figure 9: vSLAM pose estimation results versus ground truth in the London environment.

The simulation with light rain corresponds quite accurately with the ground truth (excluding the corners and the back-and-forth movement in the first stretch). These errors accumulate and let the final position deviate quite a lot from the ground truth. Simulations with heavy rain and fog resulted in much worse pose estimations which is most likely because of the lack of features in the images for the algorithm to detect.

The pose estimation error during the simulations comes from two factors: the high rotation rates of the vehicle during turning and unintended simulation artifacts more prominent during straight sections. The angular velocities appear to be overestimated which results in erratic behavior in turns. The fact that the estimated pose of the vehicle coming out of the turn corresponds well with the ground truth, suggests that the error is constant. It can be stated that this error can be attributed to the used RTAB map algorithm. On the other hand, the errors on the straight sections cannot. The source of them is the sporadic flickering of the ground surface overlay and the abrupt appearing of buildings when coming into the field of view. Both problems can be eliminated by choosing different settings and using different modelling techniques in the Simcenter Prescan software.

5. Conclusion

In this paper, we have introduced a hardware-in-the-loop (HiL) simulation framework that facilitates the testing of flight control hardware in conjunction with autonomous algorithms, such as the visual Simultaneous Localization and Mapping (vSLAM) algorithm. The utility of this framework has been demonstrated through its application in two operational modes: manual control of the aircraft via a radio controller, and autonomous operation through trajectory specification at the ground control station. The HiL setup allowed for evaluating the flight controller and UAV behavior in a virtual environment. Of course, this required some fine-tuning between the components of the setup.

The accuracy of vSLAM is highly contingent upon the environmental elements captured in the camera's field of view and the type of algorithm that is used. In this paper, RTAB-map was used because of its real-time performance. Factors such as degraded visibility and high angular rates of motion significantly impair the precision of position estimation, underscoring the necessity for realistic simulations within the HiL setup to obtain accurate outcomes. The RTAB-map relies on the detection of features. Therefore, it is important to provide realistic detailed 3D environment data for the accurate evaluation of autonomous flight performance. It would also be interesting to analyze the effect of altitude on vSLAM accuracy.

The setup shows great promise as an initial step towards safely testing localization algorithms and flight controllers in various scenarios and conditions. Integrating all components at the required rates was challenging due to the large volume of information exchanged. However, successfully operating a real flight controller in a simulated environment is a significant achievement. Additionally, the 3D environment-generated images produced fairly accurate pose estimations in some cases. Improving this HIL setup's performance could be highly beneficial for the future development and testing of SLAM and flight controller algorithms.

Looking ahead, future research will incorporate wind turbulence into the flight dynamics and develop a more detailed model of the electrical system. These enhancements will enable comprehensive studies on power consumption across various flight phases and conditions, further augmenting the fidelity and applicability of the simulation framework.

6. Contact Author Email Address

The authors can be contacted at yves.lemmens@siemens.com.

7. Acknowledgment

The research presented in this paper has been supported by the MARLOC research project and has received funding from the Flemish government through the VLAIO ICON Programme (ref. HBC.2020.2600)

8. Copyright Statement

References

[1] Shakhatreh H., Sawalmeh A.H., Al-Fuqaha A., Dou Z., Almaita E., Khalil I., Othman N.S., Khreishah A. and Guizani M. Unmanned aerial vehicles (UAVs): A survey on civil applications and key research challenges. IEEE Access, Vol. 7, pp 48572-48634, 2019

[2] Gupta A., Fernando X. Simultaneous Localization and Mapping (SLAM) and Data Fusion in Unmanned Aerial Vehicles: Recent Advances and Challenges. Drones, Vol. 6, no. 85, pp 1-35, 2022.

[3] Wang C., Wang T., Liang J., Chen Y., Zhang Y. and Wang C. Monocular visual SLAM for small UAVs in GPS-denied environments. IEEE International Conference on Robotics and Biomimetics, Guangzhou, China, pp. 896-901, 2012.

[4] Taketomi, T., Uchiyama, H. and Ikeda, S. Visual SLAM algorithms: a survey from 2010 to 2016. IPSJ Transactions on Computer Vision and Applications, Vol. 9, no. 16, pp. 1-11, 2017.

[5] Schleicher D., Bergasa L. M., Ocana M., Barea R. and Lopez E., Real-time hierarchical GPS aided visual SLAM on urban environments. IEEE International Conference on Robotics and Automation, Kobe, Japan, pp. 4381-4386, 2009.

[6] Vetrella A.R., Opromolla R., Fasano G., Accardo D. and Grassi M. Autonomous Flight in GPS-Challenging Environments Exploiting Multi-UAV Cooperation and Vision-aided Navigation. AIAA SciTech Forum Information Systems, Grapevine, Texas, pp. 1-14, 2017.

[7] ROS: Why ROS? Retrieved from: https://www.ros.org/blog/why-ros/ Accessed: April 2024.

[8] Quigley M., Gerkey B., Conley K., Faust J., Foote T., Leibs J., Berger E., Wheeler R. and Ng A. Ros: an open-source robot operating system. IEEE Intl. Conf. on Robotics and Automation Workshop on Open Source Robotics, Kobe, Japan, pp. 1-6, 2009.

[9] Theys B. and De Schutter J. Virtual Motor Torque Sensing for Multirotor Propulsion Systems. IEEE Robotics and Automation Letters, vol. 6, no. 2, pp. 4149-4155, 2021.

[10] Lemmens, Y., Teirlink, C., Hegde, A., Guerrero, D.P., Olivares, G. and Shah, H. Investigation of simulation frameworks for the evaluation of automated flight functions for eVTOL aircraft. 33rd Congress of the International Council of the Aeronautical Sciences, ICAS, Stockholm, Sweden, pp. 1-9, 2022.

[11] Prochitecture. Blosm for Blender. Github repository: https://github.com/vvoovv/blosm Accessed April 2023.

[12] QGC – QgroundControl Retrieved from: http://qgroundcontrol.com/ Accessed: April 2024.

[13] Taketomi, T., Uchiyama, H., & Ikeda, S. Visual SLAM algorithms: A survey from 2010 to 2016. IPSJ transactions on computer vision and applications, vol. 9, no. 16, pp. 1-11, 2017.

[14] Servieres M., Renaudin V., Dupuis A., Antigny and N. Visual Visual-Inertial SLAM: State of the Art, Classification, and Experimental Benchmarking. Journal of Sensors, vol. 2021, pp. 1-27, 2021.

[15] Labbé, M. and F. Michaud, RTAB-Map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation. Journal of Field Robotics, vol. 36, pp. 416–446, 2019.