EVOLUTION IN ELECTRICAL AND ELECTRONIC ENGINEERING
e-ISSN: 2756-8458

EEEE

Vol. 6 No. 2 (2025) 155-164
https://publisher.uthm.edu.my/periodicals/index.php/eeee

Obstacle Avoidance System for Px4-Based Quadrotor in The Hardware in the Loop (HITL) Simulation of Simulink

Brian Jordan Wilfred[1], Mohamad Fauzi Zakaria[1*]

[1] *Faculty of Electrical and Electronic Engineering*
Universiti Tun Hussein Onn Malaysia, Batu Pahat, Johor,86400, MALAYSIA

*Corresponding Author: mfauzi@uthm.edu.my
DOI: https://doi.org/10.30880/eeee.2025.06.02.019

Article Info

Received: 9 January 2025
Accepted: 22 September 2025
Available online: 30 October 2025

Keywords

UAV Obstacle Avoidance, MATLAB-Simulink UAV Simulation, PX4 Autopilot, Nvidia Jetson Nano, 3D-VFH+ Algorithm, Hardware-in-the-loop (HITL), Software-in-the-loop (SITL)

Abstract

Unmanned Aerial Vehicles (UAVs) have become essential for various applications, with obstacle avoidance systems critical in ensuring safety and collision prevention in dynamic environments. Current testing methods often rely on real-world flight testing, which can be time-consuming, expensive, and risky for hardware. Thus, Hardware-in-the-loop (HITL) simulation provides a significant advantage by offering a safe, repeatable, and cost-effective virtual environment for testing and validating obstacle avoidance algorithms. This study aims to integrate a PX4-based autopilot system with a companion computer in MATLAB-Simulink and evaluate the effectiveness of an obstacle avoidance algorithm within a HITL simulation for a quadcopter. The proposed system integrates a 3D-VFH+ algorithm deployed on an Nvidia Jetson Nano for real-time obstacle avoidance, leveraging simulated sensor data and Unreal Engine for a realistic 3D environment. The results demonstrate that the system effectively navigates and avoids obstacles in dynamic scenarios with a parameter setting for a minimum distance to the obstacle at 0.3m.

1. Introduction

Unmanned Aerial Vehicles (UAVs), also known as drones, are aircraft without any human pilot, crew, or passengers on board. This aircraft was created for military missions in the twentieth century; as technology improved, it has become increasingly relevant in recent years. Its application expanded to a non-military application, which ranges from search and rescue operations, package deliveries, aerial video and photography, and environmental monitoring [1], [2], [3]. However, there are critical challenges to ensure safe, reliable, and efficient drone flight control. The current drone systems development and verification often rely on real-world flight testing. This approach can be time-consuming and expensive, and can risk damaging drone hardware. Thus, Hardware-in-the-loop (HITL) simulation offers a safe and controlled alternative for testing and validating algorithms such as obstacle avoidance systems in a virtual environment [4], [5]. Moreover, recent research on UAV simulation for drone obstacle avoidance has focused on software-in-the-loop (SITL) configurations that lack the real-time hardware interaction needed for accurate performance evaluation.

This research aims to integrate a PX4-based autopilot system with an Nvidia Jetson Nano companion computer to create an effective Hardware-in-the-Loop (HITL) simulation for a quadcopter using MATLAB-Simulink. This setup enables real-time testing and validation of control algorithms to enhance flight performance. Furthermore, the study will implement and evaluate the Three-Dimensional Vector Field Histogram Plus (3D-VFH+) obstacle avoidance algorithm on the Nvidia Jetson Nano within the HITL framework

2. Methodology

The methodology focuses on key aspects, including the system design and modeling overview, hardware setup and preparation, and the setup and execution of HITL and SITL simulations in Simulink. The project has three main components for the development of a quadcopter with an obstacle avoidance system on the HITL simulation environment by using components such as Pixhawk 4 Flight Controller, Simulink Integration, and Companion Computer implementation, illustrated in a block diagram in Fig. 1.

The block diagram from Figure 1 shows a HITL simulation setup that integrates Pixhawk 4, Nvidia Jetson Nano, Simulink, and QGroundControl (QGC). In this setup, the Host PC flight controller that runs the MATLAB-Simulink is deployed to the Pixhawk 4 autopilot hardware via Universal Serial Bus (USB). Next is the onboard autonomy of obstacle avoidance, which is modeled within Simulink and is built and uploaded onto the Nvidia Jetson Nano from the Host PC via Ethernet, that uses 3D-VFH+ for its obstacle avoidance. Scenario simulation and flight visualization blocks are also created in Simulink, implementing Unreal Engine, which allows the creation and visualization of various flight scenarios. QGC is the ground control station, interfacing with the Pixhawk 4 through MAVLink protocol via the same USB connection. It provides a graphical user interface for real-time flight data monitoring, planning missions, and adjusting parameters.

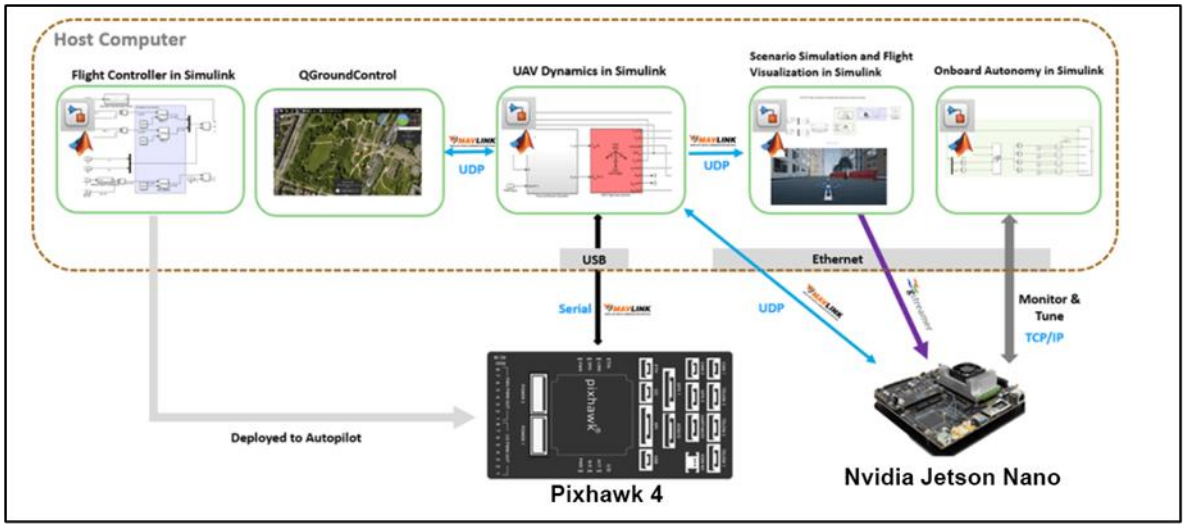


Fig. 1 *Block diagram of HITL simulation with hardware connection*

The flowchart in Fig. 2 illustrates the systematic workflow for integrating and testing a UAV system with obstacle avoidance capabilities. The process begins with setting up the Pixhawk 4 in HITL mode using QGC to establish a reliable connection. The firmware is then configured in MATLAB-Simulink, enabling the deployment of the autopilot controller, which is executed in the first MATLAB session. Following this, flight visualization is implemented through Unreal Engine, allowing realistic simulation environments for evaluation in the second MATLAB session. The workflow integrates and executes the obstacle avoidance model in the third MATLAB session, ensuring the UAV can identify and respond to environmental obstacles effectively. Next, UAV dynamics are modeled and executed in the fourth MATLAB session, ensuring the UAV's responses align with its physical parameters. The mission is then initiated via QGC by evaluating the stability of the simulated sensor performance post-flight. If instability is detected, troubleshooting and adjustments are made. Otherwise, flight data is collected and analyzed to evaluate the system's performance comprehensively. This iterative process ensures that all components, from control algorithms to real-time autonomy, are robustly validated in simulated and real-world conditions, culminating in reliable UAVs.

Evolution in Electrical and Electronic Engineering Vol. 6 No. 2 (2025) p. 155-164

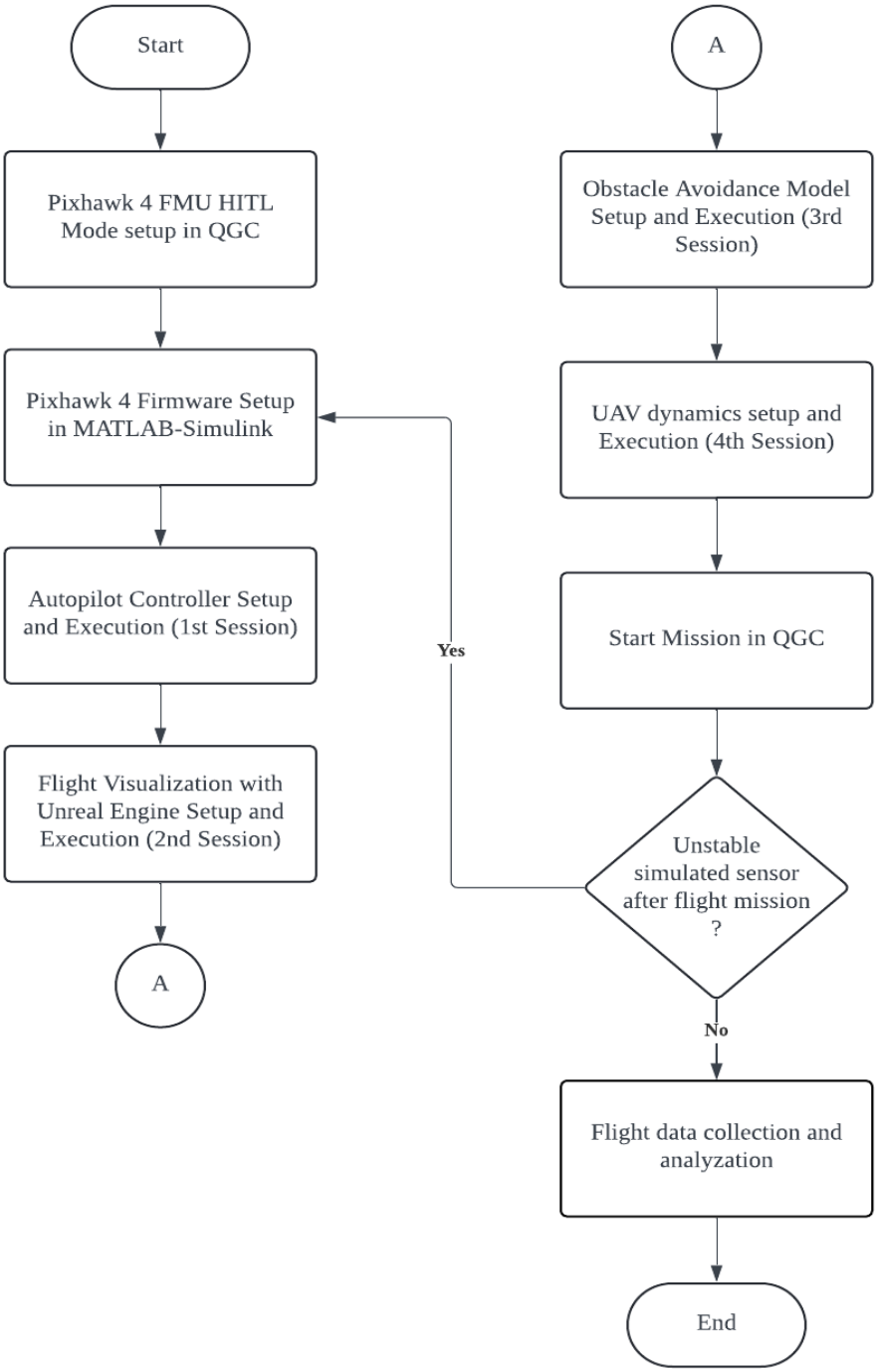


Fig. 2 *Flowchart of the HITL simulation development procedure*

2.1 Hardware-in-the-loop Simulink Setup and Implementation

The autopilot controller in a HITL simulation autonomously manages a quadcopter's flight dynamics using input commands and sensor data. It ensures stable flight by adjusting the drone's motors to follow a set path or achieve tasks like hovering, ascending, descending, or waypoint navigation. Processing real-time sensor and environmental data from the simulation generates control commands for throttle, pitch, roll, and yaw, mimicking real-world drone behavior. This functionality enables the drone to perform as expected under different conditions, ensuring precise and efficient operation in simulated and real-world scenarios. The Autopilot Controller is set up to deploy HITL Simulation for generating and deploying code to the Pixhawk 4 board. The "Hardware Board > Select Another Hardware Board" menu is selected to open the parameters dialog box. In the parameters "dialog box > Hardware Implementation", the PX4 Pixhawk 4 board is selected. Then, HITL mode is selected in the target hardware resources with Simulink as the simulator. Next, in the MAVLink tab, MAVLink on "/dev/ttyACM0" is also selected. Deployment of the generated code is done where the tab

"Hardware > Build, Deploy & Start" is selected. Since a companion computer is implemented for this HITL simulation, a block diagram for path correction from the onboard computer is chosen where "guidanceType" is configured to number 1. This is for the deployment of the first session, namely for "Autopilot Controller". Fig. 3 shows the block diagram of autopilot controller withn the Simulink model.

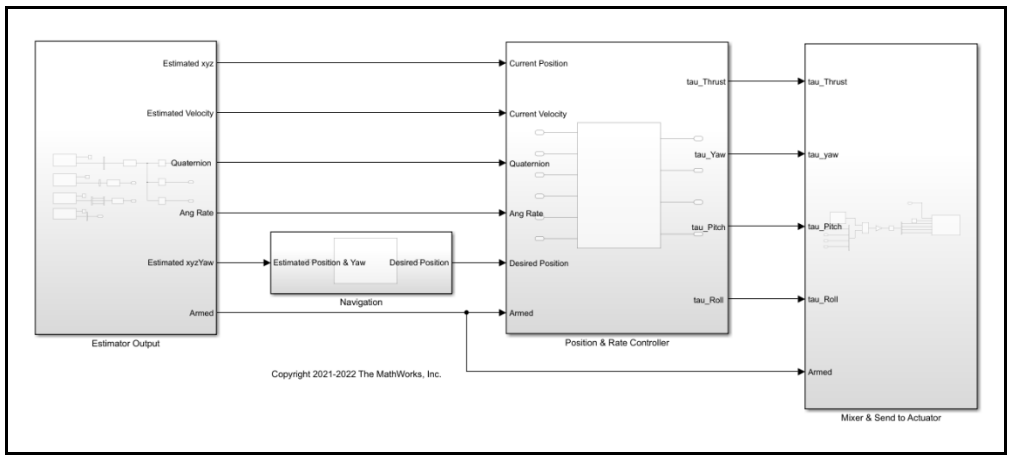


Fig. 3 *Autopilot controller block diagram within the Simulink model*

The flight visualization blocks, as illustrated in Fig. 4, are used to showcase the HITL simulation results in a comprehensible visual format, making it easier to understand the UAV system's performance and behavior. In the context of obstacle avoidance for quadcopter drones, flight visualization plays a role by providing a real-time interaction with potential obstacles. This visual feedback allows for observation and analysis of the drone navigation around obstacles based on the implemented obstacle avoidance algorithm. The setup of the simulation for scenario and flight visualization using an Unreal environment is as follows, where a second instance of the same MATLAB version was opened within the host PC. The px4demo_HITLWithOnboardComputer project was opened, and within this project, the "Open 3D Visualization with Unreal Engine" shortcut was accessed via the Project Shortcuts tab in MATLAB. This action launched the onboard model named Unreal_3DVisualization. Streaming simulated depth sensor data to the NVIDIA Jetson is enabled with the variable "enableOnboardStreaming" set to 1. The Simulation 3D Camera block provided camera images from the Unreal environment. Depth images from the camera block were streamed to the NVIDIA Jetson Nano using the Video Send block. The block parameters dialog box was accessed to add the IP address of NVIDIA Jetson Nano's companion computer, '192.168.56.1'.

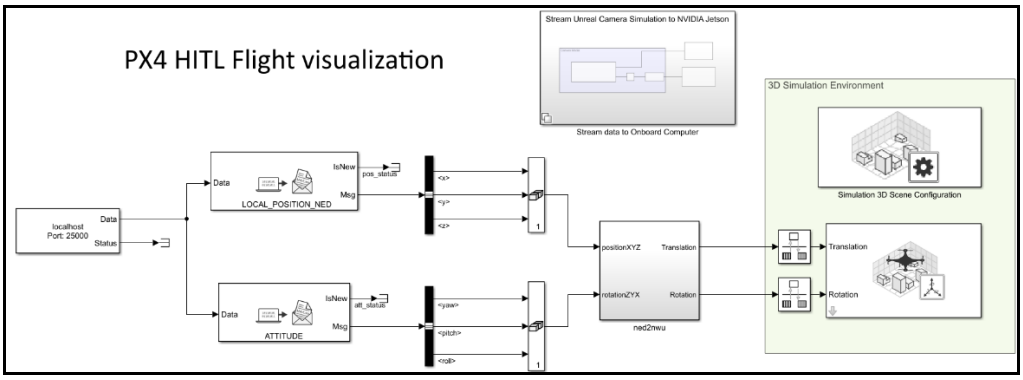


Fig. 4 *Simulink flight visualisation blocks with Unreal Engine*

This Simulink project utilizes the Obstacle Avoidance block available in the UAV Toolbox to be built onto the Nvidia Jetson Nano, allowing real-time obstacle detection and avoidance. This model implements a waypoint following a set point, which is planned from QGC with an obstacle avoidance algorithm that uses the 3D VFH+ algorithm to provide an obstacle-free path, as such algorithms are embedded in the block diagram in Fig. 4. Implementing the onboard algorithm for HITL simulation in Simulink involved several steps. First, the onboard model named "Onboard_ObstacleAvoidance" was launched by clicking "Open Obstacle Avoidance Model" in the Project Shortcuts tab. The next step involved entering the IP address and login credentials for the Nvidia Jetson Nano. This was done by navigating to Target hardware resource > Board Parameters. The IP address used was '192.168.56.1'. For the login credentials, the username is "px4" and the password is Jetson Nano's assigned password. Following this, the block parameters dialog box for the UDP Send block was accessed by double-

clicking the block. The IP address of the host PC running “UAV_Dynamics_Autopilot_Communication” was entered as the Remote IP address, which is ‘192.168.56.200’ in this case, with the port set to 14580. It was ensured that both the host PC and the companion computer were connected to the same network. Finally, the "Monitor & Tune" option was selected from the Run on Hardware section of the Hardware tab in the Simulink Toolstrip to complete the setup. This is the third running session for the HITL simulation. Fig. 5 shows the obstacle avoidance model build onto Nvidia Jetson within Simulink.

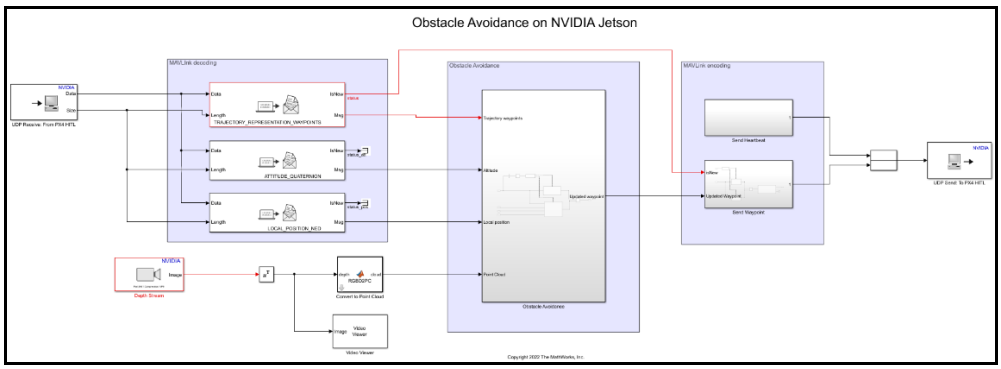


Fig. 5 *Obstacle avoidance model buildable onto Nvidia Jetson within Simulink*

In HITL simulations for quadcopter drones using Simulink, the UAV dynamics block diagram represents the mathematical model and physical behavior of the UAV. This block diagram encompasses the equations of motion, aerodynamics, propulsion system, and environmental interactions. Each of these simulation blocks requires several other tools for it to be functional, such as an “Aerospace toolbox” for the IMU sensor simulation block and the “Navigation toolbox” for the GPS sensor simulation blocks. Performing the set-up for UAV Dynamics in the Simulink plant model for HITL simulation involves first opening the “UAV_Dynamics_Autopilot_Communication” model in Simulink and configuring the serial port to match the Pixhawk connected to the host computer, ensuring that the serial port settings correspond to the Pixhawk's connection. Then, for setting up the parameters dialog box, UDP connections need to be established with the Jetson Nano using the MAVLink Bridge blocks. In the MAVLink Bridge Source and MAVLink Bridge Sink blocks, the IP address of the NVIDIA Jetson Nano is added, and ensures successful communication by verifying that the host PC can ping the Jetson by entering the port number 14540 for these connections. Additionally, flight visualization is enabled by adding a local host connection in the MAVLink Bridge Source block with port number 25000. The model simulation was initiated by clicking "Run" on the Simulation tab for the fourth session of Simulink. Once the plant model started running, a connection was established in QGC. Fig. 6 shows the Simulink block diagram of UAV dynamics data serialization.

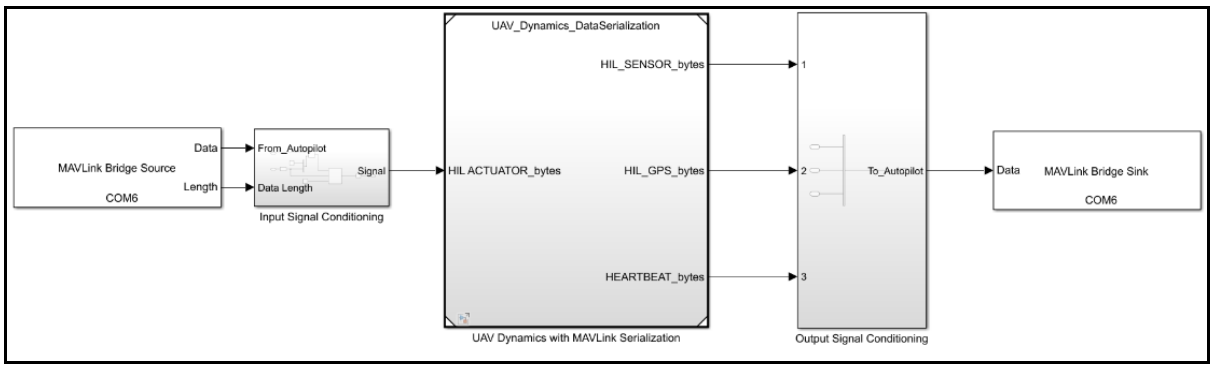


Fig. 6 *UAV dynamics data serialization Simulink block diagram*

2.2 Software-in-the-loop (SITL) Simulink Implementation and Setup

Software-in-the-Loop (SITL) is a simulation environment used to test and validate algorithms or control systems for UAVs without the need for physical hardware, which allows for an evaluation of obstacle avoidance system performance. Moreover, it is essential to detail the process of validating the obstacle avoidance algorithm using SITL simulation before transitioning into HITL testing. The block diagram of the SITL simulation environment is illustrated in Fig. 7.

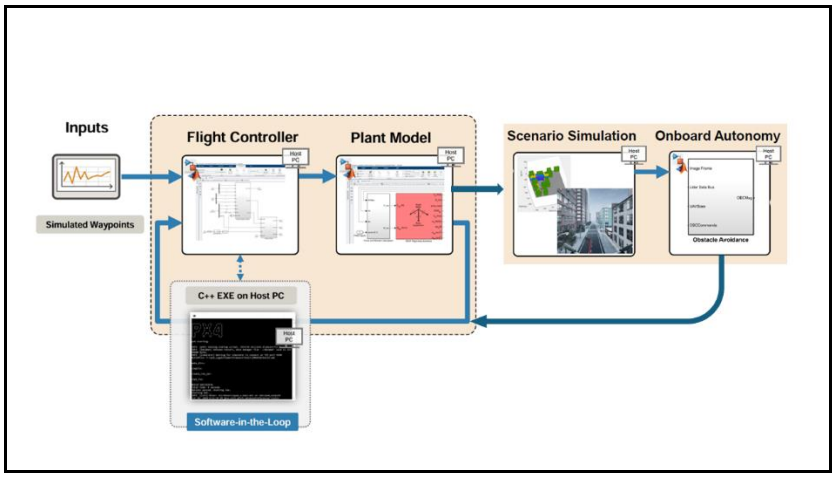


Fig. 7 *SITL simulation block diagram*

The code can be written in script files within Simulink, which defines a simulation environment for UAV obstacle avoidance testing using MATLAB. First, create a 3D scenario with a UAV platform that is initialized at a starting position with a specific orientation. Next, key waypoints for the UAV's trajectory are defined, along with several obstacles placed in the environment modeled as a polygon with specified positions, widths, and heights. Then, a 3D visualization of the environment is generated to display the UAV, waypoints, and obstacles. The UAV is then equipped with a simulated LiDAR sensor that generates point cloud data based on its surroundings. The UAV's motion is governed by a proportional-derivative-integral (PID) control system, with gains and parameters specified for smooth flight. Finally, the UAV's trajectory is simulated using a Simulink model, and the resulting path is plotted in the 3D visualization, showing both the intended waypoints and the actual trajectory. Next, the Obstacle Avoidance block parameters were fine-tuned in SITL to optimize UAV detection and avoidance. Sensor Range Limits were set to [0.2, 7] meters, and the Field of View to horizontal: [-179, 179] and vertical: [-15, 15] degrees for efficient obstacle detection. Sensor Location and Orientation were adjusted to [0, 0, -0.4] meters and [180, 0, 0] degrees, simulating realistic mounting. Vehicle parameters, including a 0.1-meter radius and 1-meter minimum obstacle distance, ensured safety and maneuverability. Fig. 8 shows UAV obstacle avoidance method of 3D-VFH+ in Simulink.

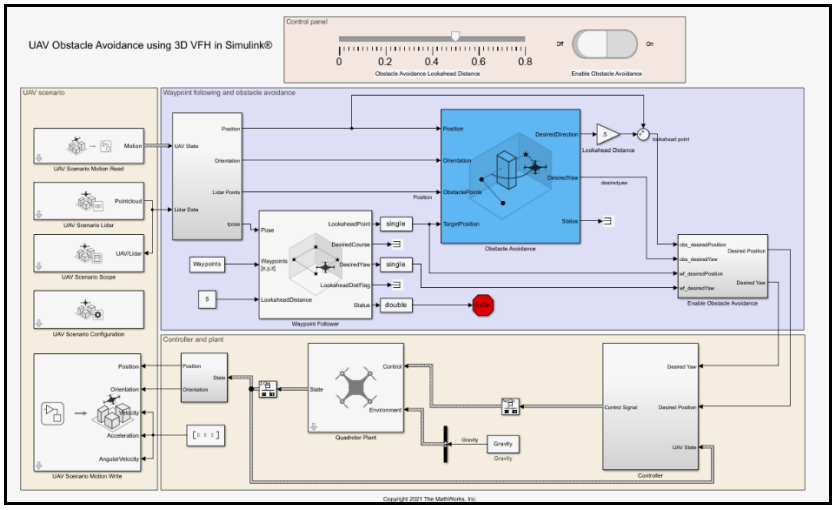


Fig. 8 *UAV obstacle avoidance method of 3D-VFH+ in Simulink*

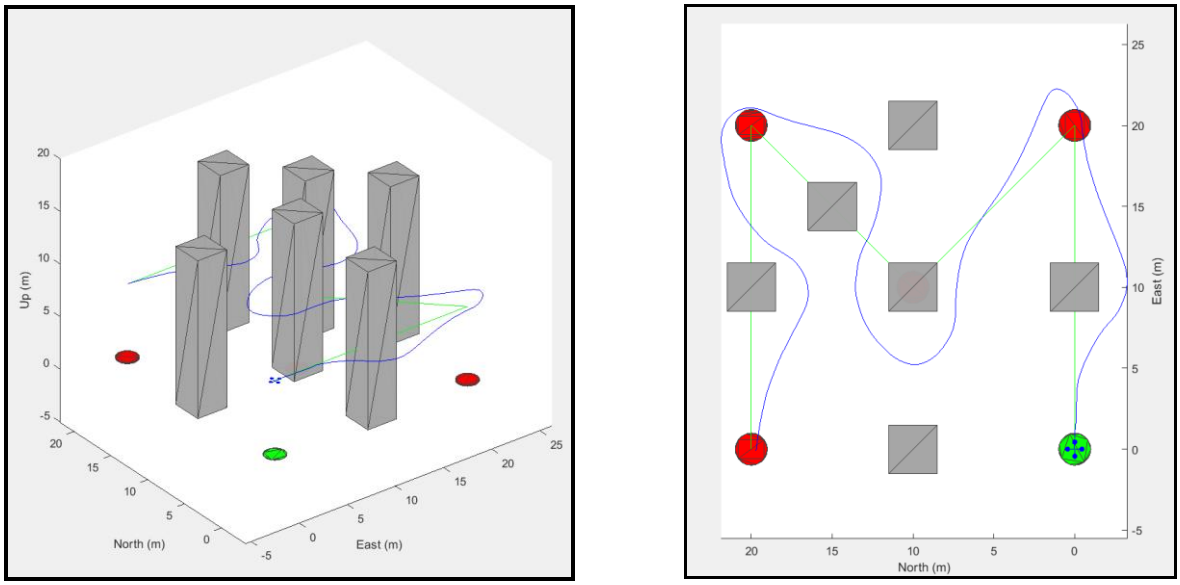
3. Result and Discussion

3.1 Validation of obstacle avoidance in Software-in-the-loop (SITL) simulation of Simulink

Based on Fig. 9, it can be observed that the drone in the SITL environment successfully navigated a complex obstacle-filled traversal path using the parameters detailed in section 2.2 for the 3D-VFH+ algorithm. The trajectory demonstrates successful, precise obstacle avoidance as the drone effectively adapts its path to maneuver around obstacles while maintaining its intended direction toward the final target. Thus, the drone's

Evolution in Electrical and Electronic Engineering Vol. 6 No. 2 (2025) p. 155-164

behavior highlights the robustness of the set parameter, where it maintains safe distances from obstacles while optimizing its flight efficiency. Moreover, the combination of adjusted azimuth, elevation, sensor range, and minimum distance to obstacles has contributed to a highly adaptive and reliable navigation strategy.



(a) **(b)**

Fig. 9 *Drone path tragectory for complex 3D obstacle avoidance (a) Side view; (b) Top view*

3.2 Hardware connection setup for HITL simulation

The hardware configuration for the HITL simulation, illustrated in Fig. 10, consists of three primary components which is a Pixhawk 4 autopilot that is used for the UAV's flight controller, Nvidia Jetson Nano which serves as the onboard processing unit for the obstacle avoidance algorithm and a host PC that is used for uploading buildable to both Pixhawk 4 and Jetson Nano as well as running the simulation environment via Unreal Engine and data logging. Communication between the Jetson Nano and the host PC is established via an Ethernet connection with an additional Ethernet adapter, which is used to connect the Jetson Nano to the network. The Pixhawk 4 also connects to the host PC via a USB 2.0 Micro-B cable. USB extensions are connected to the host PC to provide additional ports.

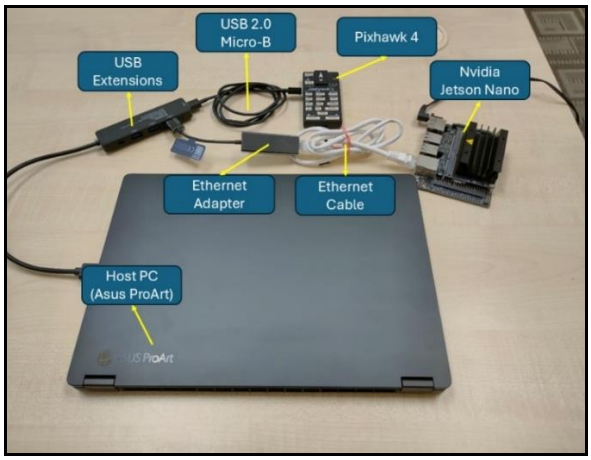


Fig. 10 *Hardware setup and connection*

3.3 Validation of Obstacle Avoidance in Hardware-in-the-loop (HITL) through Parameter Tuning

Within the results of the Hardware-in-the-Loop (HITL) testing, four (4) tests were conducted to evaluate the drone's performance under various obstacle avoidance scenarios using the 3D-VFH+ algorithm. The evaluation focuses on several key aspects, such as minimum distance to an obstacle (m), drone flight path behavior, drone speed, and obstacle avoidance status. The obstacle avoidance status refers to the feedback generated by the 3D-

VFH+ algorithm during the simulation; this status is visualized using the Simulink scope block. It is categorized into four distinct statuses, which are:

- 0 - An obstacle-free direction is found,
- 1 - No obstacle-free direction is found,
- 2 -An obstacle-free direction is found but is close to the obstacle,
- 3 - No obstacle-free direction is found and is close to the obstacle

3.3.1 First Flight Scenario Results With 0.5m of Minimum Distance to Obstacles

The 0.5m minimum distance to the obstacle parameter is the default value for the obstacle avoidance block. In Fig. 11, for the first flight scenario, the drone successfully detected and avoided obstacles, with status fluctuating between 3 and 2, indicating proximity to the obstacle while finding a clear path. This fluctuation was due to instability, causing temporary reversion to previous paths. Despite this, the system performed well, actively avoiding collisions. The flight path also closely aligns with GPS data, confirming proper flight controller function. The drone's average 4.0 km/h speed reflects cautious navigation due to nearby obstacles.

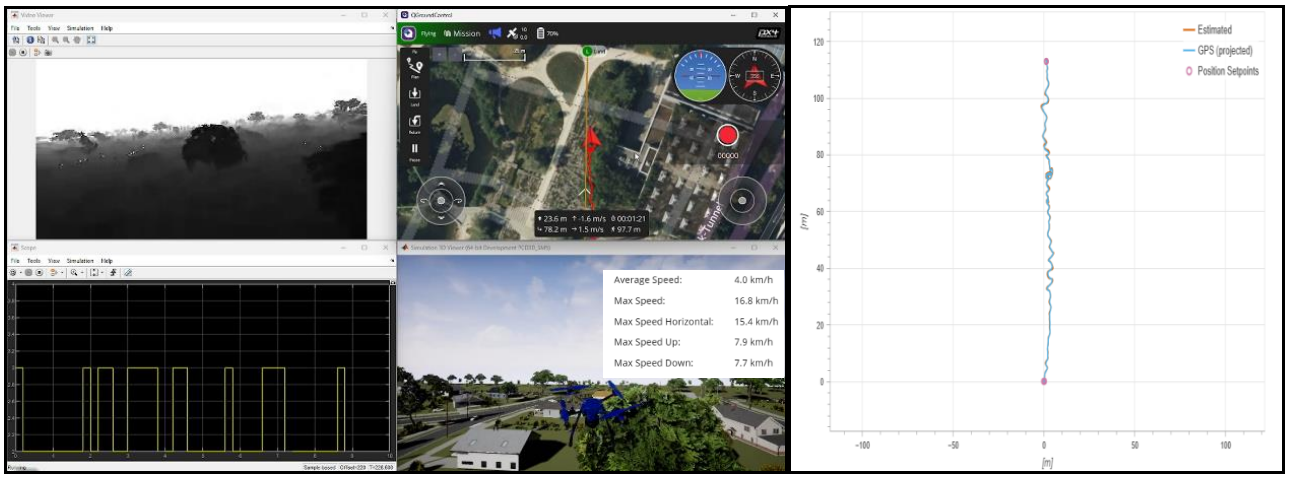


Fig. 11 *First flight scenario simulation with flight path and speed data*

3.3.2 Second Flight Scenario Results With 1m of Minimum Distance to Obstacles

Fig. 12 shows the second flight scenario with a 1-meter minimum distance to the obstacle parameter. The drone encountered a tree, became trapped in a local minima, and displayed status 3, indicating no obstacle-free direction. A safety measure switched the drone to "hold flight mode," halting progress. This highlights the lack of a strategy to bypass obstacles, such as altitude adjustments. The increased detection range caused a jagged trajectory due to constant path corrections and consistent obstacle detection. The drone also stalled at the 100-meter mark along the y-axis. Despite this, it averaged 4.2 km/h, reflecting frequent directional changes.

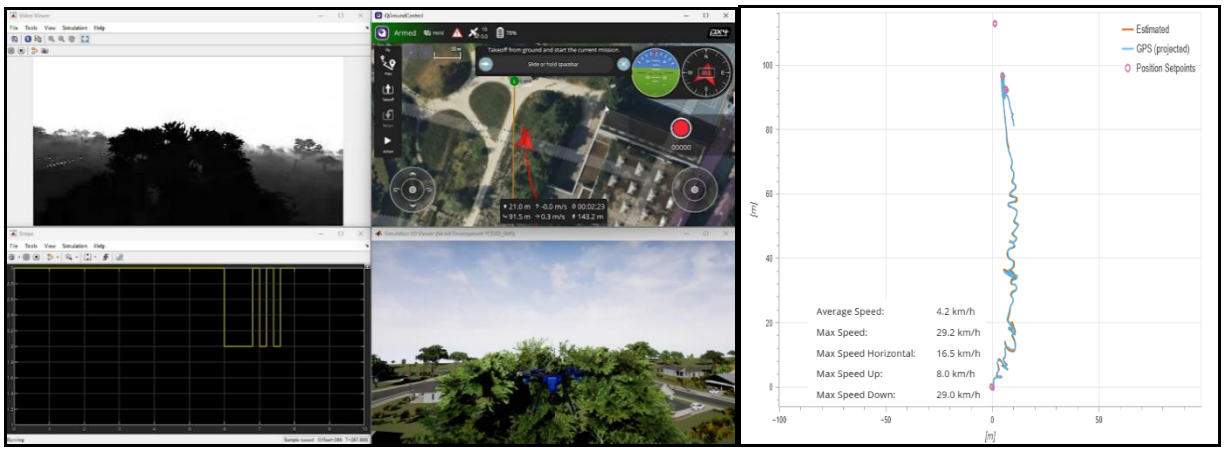


Fig. 12 *Second flight scenario simulation with flight path and speed data*

3.3.3 Third Flight Scenario Results With 0.3m of Minimum Distance to Obstacles

Evolution in Electrical and Electronic Engineering Vol. 6 No. 2 (2025) p. 155-164

Reducing the minimum distance to obstacle parameters allowed the drone to identify a clear path, even near obstacles consistently. Fig. 13 shows only two status three fluctuations, indicating effective obstacle navigation. For the flight path, the drone followed the set path with slight deviations but exhibited jagged segments due to instability. The average 5.9 km/h speed reflects slower, deliberate adjustments, as the reduced detection distance made the drone less cautious. This behavior suggests that the drone prioritized maintaining forward progress while navigating its surroundings with minimal impact on overall path adherence and obstacle avoidance performance.

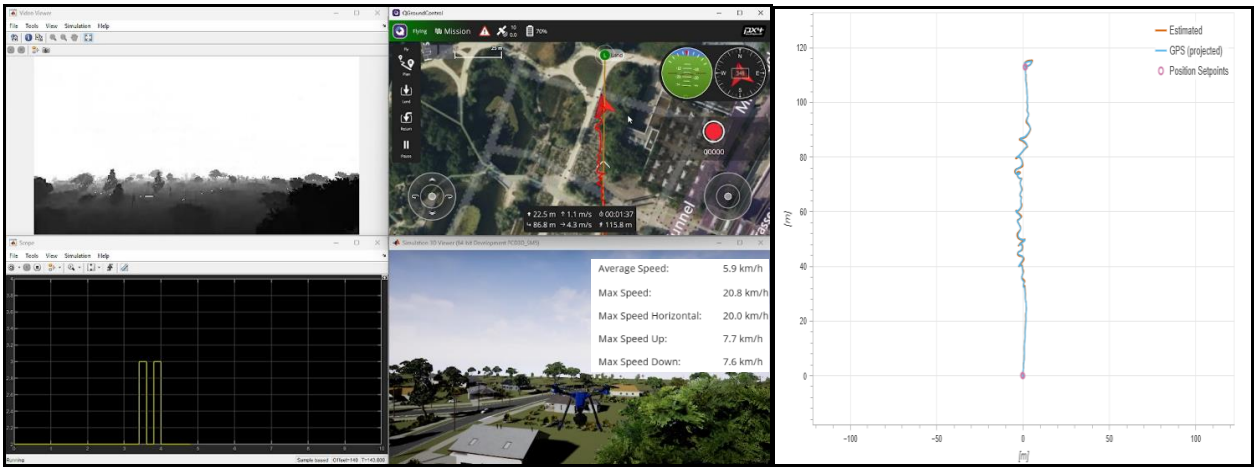


Fig. 13 *Third flight scenario simulation with flight path and speed data*

3.3.4 Fourth Flight Scenario Results With 0.15m of Minimum Distance to Obstacles

In the fourth flight scenario, as shown in Fig. 14, reducing the object detection distance to 0.15 meters allowed the drone to consistently find obstacle-free directions, even near objects that can be seen with the status 2 in the scope. The flight path is smoother and closely follows the set trajectory compared to earlier scenarios. However, the drone's speed decreased to 3.7 km/h, reflecting the flight controller's cautious approach near obstacles. This speed reduction minimizes collision risks due to the minimal distance between the drone and the obstacles, ensuring safer navigation while maintaining practical obstacle avoidance.

Fig. 14 *Fourth flight scenario simulation with flight path and speed data*

3.4 Discussion

Testing four flight scenarios with varying minimum obstacle distances reveals distinct differences in drone performance and navigation strategies. In the first scenario, with a 0.5-meter distance, the drone managed competent obstacle detection at a moderate speed of 4.0 km/h, despite some instability. Comparatively, the second scenario, set at 1 meter, resulted in the drone becoming trapped and highlighted the need for better navigation strategies. The third scenario, with a reduced minimum distance of 0.3 meters, showed significant improvement, achieving 5.9 km/h with fewer status fluctuations and more efficient navigation. Lastly, the fourth scenario, at 0.15 meters, allowed the drone to navigate safely near obstacles but at a decreased speed of 3.7

km/h due to increased caution. Overall, the 0.3-meter distance provided the best balance between speed and obstacle avoidance performance, emphasizing the importance of advanced navigation algorithms for improved safety and efficiency in future flights.

4. Conclusion

In conclusion, this project successfully integrated the Pixhawk 4 (PX4) autopilot with the Nvidia Jetson Nano in a MATLAB-Simulink Hardware-in-the-Loop (HITL) simulation environment. The 3D-VFH algorithm enabled real-time obstacle avoidance, achieving a good response at a faster response and navigation speed of 5.9 km/h with the setting parameter for minimum distance to the obstacle at 0.3m. These results validate the effectiveness of HITL simulations for UAV obstacle avoidance and highlight the potential of the 3D-VFH+ algorithm in enhancing autonomous navigation. The next step is implementing these findings in a physical quadcopter for real-world testing. Exploring advanced sensors and machine learning techniques may improve adaptability in complex dynamic environments.

Acknowledgement

The authors gratefully acknowledge Universiti Tun Hussein Onn Malaysia (UTHM) for laboratory facilities and funding via Tier-1 grant (Vote U857) and Final Year Project support.

Conflict of Interest

Authors declare that there is no conflict of interests regarding the publication of the paper.

Author Contribution

The authors confirm contribution to the paper as follows: ***study conception, design, data analysis and manuscript preparation:*** *Brian Jordan Wilfred; Mohamad Fauzi Zakaria. All authors reviewed the results and approved the final version of the manuscript.*

References

[1] R. Sharma and R. Arya, "UAV based long range environment monitoring system with Industry 5.0 perspectives for smart city infrastructure," Comput. Ind. Eng., vol. 168, p. 108066, Jun. 2022, doi: 10.1016/j.cie.2022.108066.

[2] J. I. Damanik, I. M. D. Sitanggang, F. S. Hutabarat, G. P. B. Knight, and A. Sagala, "Quadcopter unmanned aerial vehicle (UAV) design for search and rescue (SAR)," in Proc. 2022 IEEE Int. Conf. Comput. Sci. Inf. Technol. (ICOSNIKOM), Laguboti, North Sumatra, Indonesia, Oct. 2022, pp. 1–6, doi: 10.1109/ICOSNIKOM56551.2022.10034866.

[3] M. Khosravi and H. Pishro-Nik, "Unmanned aerial vehicles for package delivery and network coverage," in Proc. IEEE 91st Veh. Technol. Conf. (VTC2020-Spring), Antwerp, Belgium, May 2020, pp. 1–5, doi: 10.1109/VTC2020-Spring48590.2020.9129495.

[4] N. Aoki and G. Ishigami, "Hardware-in-the-loop simulation for real-time autonomous tracking and landing of an unmanned aerial vehicle," in Proc. IEEE/SICE Int. Symp. Syst. Integr. (SII), Atlanta, GA, USA, Jan. 2023, pp. 1–6, doi: 10.1109/SII55687.2023.10039438.

[5] Z. Zuo, C. Liu, Q.-L. Han, and J. Song, "Unmanned aerial vehicles: Control methods and future challenges," IEEE/CAA J. Autom. Sin., vol. 9, no. 4, pp. 601–614, Apr. 2022, doi: 10.1109/JAS.2022.105410.