Multi-Platform Hardware In The Loop (HIL) Simulation for Decentralized Swarm Communication Using ROS and GAZEBO

Saran Khaliq*, Shahzeb Ahsan*, and M.Danish Nisar†

*Swarm Robotics Lab, NCRA, University of Engineering & Technology (UET), Taxila.
†Sir Syed CASE Institute of Technology (SSCIT), Islamabad, Pakistan.
Email: *{saran.khaliq,shahzeb.ahsan}@uettaxila.edu.pk, †{m.danish.nisar@case.edu.pk}

***Abstract*—Swarm of robots is a group of multiple autonomous agents, collaborating with each other to achieve collective missions such as surveillance and tracking. In Unmanned Aerial Vehicles (UAVs) swarms, robust and decentralized communication between the agents is required. To this end, Software-In-The-Loop (SIL) is performed before the actual flight to avoid the risk of failure. However, single-platform-based simulations can neither fully encounter nor resolve the communication challenges. In this work, we target to incorporate the real-time communication challenges such as network failure, faced by multi-agent systems via Hardware-In-The-Loop (HIL) simulation using Robot-Operating-System (ROS), Gazebo and off the shelf communication modems. A Ground-Control-Station (GCS) is developed to monitor and supervise autonomous UAVs missions. The multi-platform HIL completely emulates the multi-agent system and the communication between the agents, thereby reducing the risk of swarm failure.**

***Index Terms*—Decentralized Communication, Swarm Communication, Multi-agent, Software-In-The-Loop (SIL), Hardware-In-The-Loop (HIL), ROS, Gazebo.**

I. Introduction

Lately, the swarm of UAVs, consisting of multiple UAVs collaborating to perform different missions, are being used in military, industrial, agriculture, emergency rescue, and numerous other applications. Formation control and path planning are the most important tasks for the swarm to achieve its goal. These critical missions motivate researchers to develop robust architecture to maintain specific formations and to avoid obstacles and collisions. Hence, we need robust and faultless communication among the swarm nodes.

Mainly, two methods are used for the swarms to communicate with each other, which are either centralized and decentralized. The centralized communication method needs each robot to communicate through one central agent, which could be a ground control station (GCS), as pursued in [1]–[6]. A consequence of centralized control is that if the GCS link breaks, the whole swarm fails. This single point of failure is highly undesirable for swarm missions. On the other hand, in a decentralized approach, the robots communicate with each other using mesh connectivity where each robot is connected to one or multiple robots within its range. This makes the swarm independent of a single point of failure. Hence, if one agent fails, it does not affect any other agent in the swarm, making the system more robust.

Considerable work has been done on UAV swarms with both centralized and decentralized communication. Elkilany et al. proposed a neural network (NN) model of decentralized formation control algorithm [7]. The experiment is done using three turtlebot3 swarm running ROS, whereas the algorithm was implemented in MATLAB. The proposed NN algorithm performs fine in searching and tracking various trajectories. However, decentralization of swarm communication through a single ROS-master remains unresolved.

Several algorithms can be found on leader-follower formation. Souza et. al. proposed two strategies based on leader-follower formation [8]. This work is implemented on ROS framework and accessed on Gazebo simulator, thus highlighting the behavior of drones in the terms of response time and accommodation time of the formation. Although they have done experiments on gazebo using ROS, the implementation of decentralized swarm communication is not fully investigated.

Petracek experimented with vision-based self-localization in aerial swarms [9]. They have proposed a simple method to navigate, control and stabilize aerial robots while having information about only the local agent, thus making it decentralized. As they have used ROS for implementing the swarm, but their communication is still dependent on a single ROS-master. Indriyanto et al. developed a basic simulation-based swarming structure for search, monitoring, and mapping applications [10]. They have developed a centralized but modular UAVs solution which updates the mission if any of the UAV's link gets broken. Therefore, the rest of the UAVs complete the mission with the trade-off of additional time. However, this setup is yet to be tested on hardware.

Lamping et al. worked on a multi-agent UAV system based on ROS [11]. They have experimented with control and supervision algorithms on multiple UAVs and created a software package as flyMASTER. They have done both SIL and HIL and created a GCS to control and monitor the swarm. Their system can work on any flight controller that can be integrated with Micro Aerial Vehicle Link (MAVLink) communication

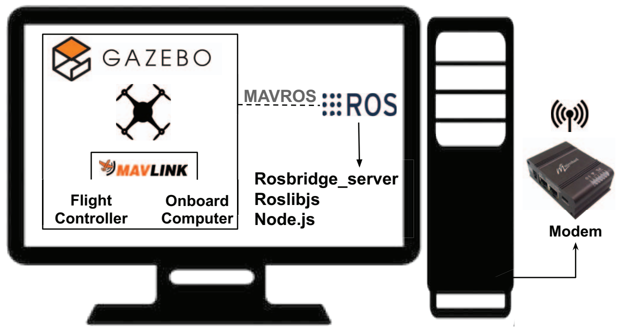


Fig. 1: System Description

protocol. Thus, experimenting on multiple UAVs having PX4 and Pixhawk flight controllers. However, their SIL consists of a single platform simulating multiple drones ultimately depending upon a single ROS-master.

In our work, we propose a decentralized approach on multi-platform HIL simulation of UAV swarm. We handpicked the linking libraries/ tools and developed a ROS network of UAVs that does not depend upon a single ROS-master, making the communication link more robust. We have also tested our proposed HIL simulation and monitored respective swarm behaviors on GCS. The main features of our work are as follows:

A. Emulating Individual UAV

- Each UAV is simulated in a physically isolated system equipped with a ROS-Gazebo simulation.
- Each system is attached to a wireless modem.

B. Multi-Platform communication

- Communication between physically isolated systems connected via modems.
- Immune to team-member failure i.e. out-of-network, malfunctioning.

C. Ground Control Station (GCS)

- Monitoring the sensory data of each UAV.
- Visualizing the video stream of UAVs.
- Geo-locating all UAVs on a single 2D map.

II. SYSTEM ARCHITECTURE

Our multi-platform HIL relies on existing software and tools such as ROS, Gazebo, and MAVROS. We employ Microhard modems for wireless communication between UAVs. The complete architecture of a testing platform can be seen in Figure 1. Individual components are briefly explained below.

A. ROS

When it comes to programming robots, researchers face lots of complexities to develop software. ROS, however, is a set of libraries that provides the researchers with the opportunity to fast-track product development by reducing software complexity [12]. ROS provides a distributive architecture, which allows sharing of information without relying on a single link. ROS supports different programming languages i.e. C++, Python, JAVA, Lisp, and other famous tools such as Lab-VIEW. It provides domain-specific tools and libraries to smooth the process of development. ROS architecture can be seen in Figure 2.

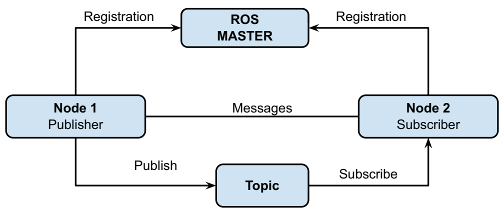


Fig. 2: ROS architecture

B. Gazebo

Gazebo is a simulator that provides a three-dimensional (3D) environment implanted with both indoor as well as outdoor environments, containing 3D models of robots and accompanying sensors to set up. It comes with real-world models with their GPS information [13]. Many simulators can be used for the modeling and simulating UAV hardware-in-the-loop system. The jMavSim [14] is a simple open-source used for simulating a lightweight multi-rotor UAV system. However, there are some limitations regarding simulating more complex models. Other simulators that can be used for HIS such as FlightGear [15], Vrep [16], GEFS [17], Xplane [18], and MAV3DSim [19] offers a good simulation platform but a significant amount of effort is required to test high-level control algorithms.

To avoid these limitations, we chose the most popular open-source simulation software in the robotics community, Gazebo [20]. The main advantage of this simulator is the easiness to set up a 3D virtual environment, modeling robots, and their sensors, and defining the communication protocol. Furthermore, its open dynamics engine (ODE) is useful to perform real-time experiments accurately [21]. Thus, providing real-time sensor readings in a dynamic simulation environment for the UAVs.

C. MAVROS

A UAV is equipped with a Flight Control System (FCS) to control different components of a UAV. FCS uses a specific protocol named MAVLink. It is a library explicitly designed for limited bandwidth applications i.e. UAVs. Micro Aerial Vehicle Robotic Operating System (MAVROS) is a dedicated ROS package that works as a bridge between MAVLink and ROS in order to provide high-level control over FCS using ROS [22].

D. Ground Control Station (GCS)

Rosbridge is a web-socket server, which lays an extra layer to the ROS. This bridge allows the communication of non-ROS applications with ROS applications. $Roslibjs$ is a javascript-based ROS library that is used in applications particularly

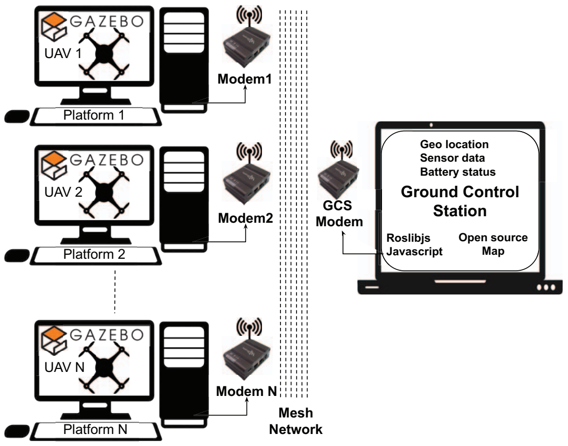


Fig. 3: Communication of multi-UAVs on different systems

using IP sockets, consenting them to get interfaced with ROS applications [23] [24].

We have developed a GCS, a webpage using $Roslibjs$ which monitors the sensor's data, GPS location, and respective swarm formations and flocking on a 2D world map provided by leaflet OpenStreetMap contributors [25]. The customized GCS web page can be seen in Figure 4.

III. UAV Mathematical Model

We used quad-rotor UAV having a 6-DOF model in the gazebo software, in which we have to control the angular velocity of each motor. Two diagonal motors of the UAV rotate in the clockwise direction and the other two rotate in the anticlockwise direction. Motor force and the moment are produced around its axis by providing thrust and torque such as

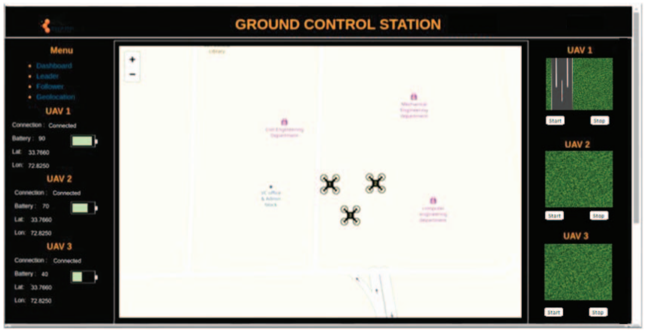
$$F_i = C_f - \omega_i^2 \tag{1}$$

$$M_i = C_m - \omega_i^2 \tag{2}$$

where C_f is the motor thrust constant, C_m is the motor torque constant and ω_i is the angular velocity of i^{th} motor. Therefore, the control inputs i.e. I_1, I_2, I_3, I_4 can be defined as

$$\begin{bmatrix} I_1 \\ I_2 \\ I_3 \\ I_4 \end{bmatrix} = \begin{bmatrix} C_f & C_f & C_f & C_f \\ -l_1C_f & -l_1C_f & l_1C_f & l_1C_f \\ l_2C_f & -l_2C_f & -l_2C_f & l_2C_f \\ -C_m & C_m & -C_m & C_m \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix} \tag{3}$$

where l_1 and l_2 are the distances along the x and y-axis to UAV's center of gravity. This motor control function is described in the motor plugin in the gazebo. These motors are derived using two methods, first is to take the signal from the controller and use the inverse function of Equation 3 to get the angular velocity of all motors, which is further used for the rotation of the propellers. The second is to use force and moments for each motor (I_1, I_2, I_3, and I_4) from Eq. 3



UAV 1
Connection : Connected
Battery : 90
Lat: 33.7660
Lon: 72.8250

UAV 2
Connection : Connected
Battery : 70
Lat: 33.7660
Lon: 72.8250

UAV 3
Connection : Connected
Battery : 40
Lat: 33.7660
Lon: 72.8250

Fig. 4: Ground Control Station

and feed these control inputs to the 6DOF model of the quad-rotor UAV. The dynamic model contains the translational and rotational motion for the quad-rotor UAV which is defined as roll ϕ, pitch θ, yaw ψ, and x, y, z respectively. The model is implemented and explained in [26], which is described by the following equations

$$\begin{cases} \ddot{\phi} = \frac{1}{M_{xx}}\big(I_2 - J_r\theta\omega_r + \big(M_{yy} - M_{zz}\big)\dot{\psi}\dot{\theta}\big) \\ \ddot{\theta} = \frac{1}{M_{yy}}\big(I_3 - J_r\phi\omega_r + \big(M_{zz} - M_{xx}\big)\dot{\phi}\dot{\psi}\big) \\ \ddot{\psi} = \frac{1}{M_{zz}}\big(I_4 + \big(M_{xx} - M_{yy}\big)\dot{\theta}\dot{\phi}\big) \\ \ddot{x} = \cos\theta\cos\phi\sin\theta + \sin\theta\sin\phi\frac{I_1}{m} \\ \ddot{y} = \cos\theta\sin\phi\sin\theta - \sin\theta\cos\phi\frac{I_1}{m} \\ \ddot{z} = \cos\phi\cos\theta\frac{I_1}{m} - g \end{cases} \tag{4}$$

where M_{xx}, Myy, M_{zz} are moments of inertia around the frame axis; ω_r is the sum of all angular velocities; J_r is the inertia of the motor; m is the total mass and g is gravitational constant. Global positioning system (GPS) and the Inertial measurement unit (IMU) sensor modules are used in each of the quad-rotor models in Gazebo. The GPS data contains the longitude (δ), latitude (λ), and altitude (z), whereas IMU data contains the information of the angular velocity (v_a), linear acceleration (a), and orientation (ξ). Gazebo uses physics library to provide us the current position (x, y, z), IMU data (v_a, a, ξ) and the initial location (δ_0, λ_0, z_0). Based on the above data the current location is calculated as [27],

$$\begin{cases} \delta_1 = \arcsin\big(\cos d\sin\lambda_0 + \frac{x}{\gamma}\sin d\cos\delta_0\big) \\ \lambda_1 = \lambda_0 + \arctan\left(\frac{y\sin d}{\gamma\cos d\cos\delta_0 - x\sin d\sin\delta_0}\right) \\ z_1 = z_0 + z \end{cases} \tag{5}$$

where $\gamma = \sqrt{x^2 + y^2}$, $d = \frac{\gamma}{R}$ and R is the radius of Earth.

IV. Proposed Methodology

A. Emulating individual UAVs

A Gazebo simulator is required to simulate a UAV in a virtual world. In Gazebo, MAVLink is responsible for establishing a connection between the FCS and its onboard

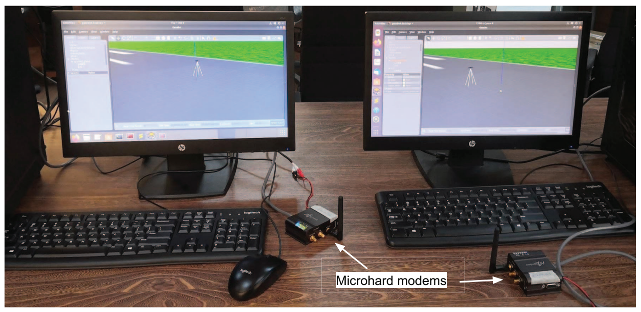


Fig. 5: Communication of two UAVs simulating on Gazebo at two separate platforms

computer. To give high-level control, MAVROS creates a bridge between FCS and ROS.

The ROS messages of UAV simulated in Gazebo are published on a certain topic through MAVROS, which are forwarded to a web socket using ros_bridge. The modems attached to each platform will create a mesh network so that the UAVs emulated on multiple platforms can share information.

B. Establishing communication between UAVs and with GCS via modems

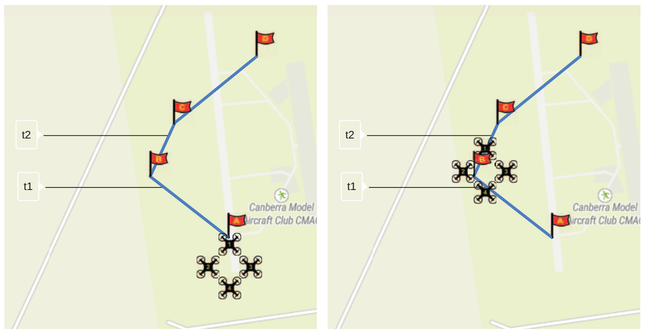
UAVs are simulated on Gazebo-simulators on three different platforms physically isolated communicating over a mesh network of modems. Each UAV is assigned a unique $IP\_address$ so that the UAVs can communicate with each other using these addresses in a network as shown in Figure 3.

A GCS using $Roslibjs$ can fetch information through an IP socket. GCS being the part of the network can subscribe to the published topic provided the $IP\_address$ of the particular system with Gazebo-simulator. The communication of GCS with other UAVs simulated on physically isolated systems can be seen in Figure 3.

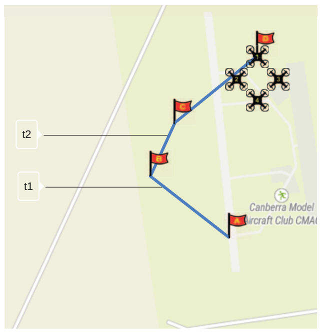
The GCS will receive the Global Position System (GPS) coordinates from UAVs in Gazebo which can be visualized on a 2D Map, thanks to $Roslibjs$. This functionality is important to see the collaboration as the UAVs are in the same world but on different platforms. Similarly, data from sensors are monitored which are published by ROS on topics and are subscribed on GCS.

C. The architecture of Decentralized Communication

Microhard modem pMDDL2450 provides a built-in feature of mesh connectivity, which helps in creating a decentralized communication network for UAVs swarm operations [28]. Each platform is connected to every other platform via a mesh network through this modem. If any of the nodes leaves or joins the network, the rest of the nodes will communicate independently in a network. If any node is not directly connected to the other node, the modem uses its proprietary routing algorithms to send the data using relay nodes. The complete architecture of decentralized communication is shown in Figure 3.


t2
t1
Canberra Model Aircraft Club CMA
t2
t1
Canberra Model Aircraft Club CMA

(a) Initial point (b) Mid flight



(c) Destination point

Fig. 6: GCS showing the formation and trajectory of UAVs, mission points A, B, C and D.

V. Experimentation and Results

Each platform in the network is installed with Ubuntu 18.04, ROS melodic, Gazebo9 simulator, and an open-source ROS package provided by Intelligent Quads (IQsim) community [29]. IQsim provides high-level control of UAVs to develop intelligent drone applications. The gazebo is loaded with a model of a runway at Canberra Model Aircraft Club Flying Field, located in Symonstion, Australia. Each platform is further connected to wireless modems through Ethernet for their communication. The wireless modems are sending their data between the platforms and the GCS with a transmission power of 7 dBm. UAVs simulated in a Gazebo on two physically isolated platforms can be seen communicating with each other in Figure 5. Similarly, more platforms can be added to increase the number of UAVs in the network and their collective information can be seen on GCS.

To investigate the communication between UAVs over a mesh network, a leader-follower formation is created. The four platforms, one leader-UAV, and three follower-UAVs are emulated on the same runway in Gazebo. The mission points are only dictated to the leader to fly from point A to point D as shown in Figure 6a. Initially UAVs maintain a specific distance, resulting in a diamond formation. The three followers go along with the leader while maintaining a specific distance based on IMU readings of their leader until the leader reaches its destination point. These platforms are communicating GPS

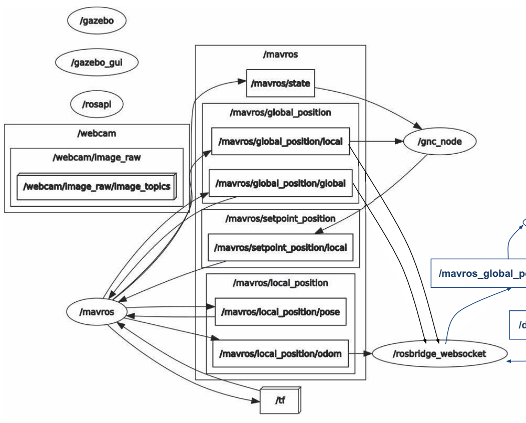

/gazebo
/gazebo_gui
/rosapi
/webcam
/webcam/image_raw
/webcam/image_raw/image_topics
/mavros
/mavros/state
/mavros/global_position
/mavros/global_position/local
/mavros/global_position/global
/mavros/setpoint_position
/mavros/setpoint_position/local
/mavros/local_position
/mavros/local_position/pose
/mavros/local_position/odom
/gnc_node
/rosbridge_websocket
/tf
Node.js
/mavros_global_position_local
/drone1_gps_local

Fig. 7: ROS-node graph of Leader

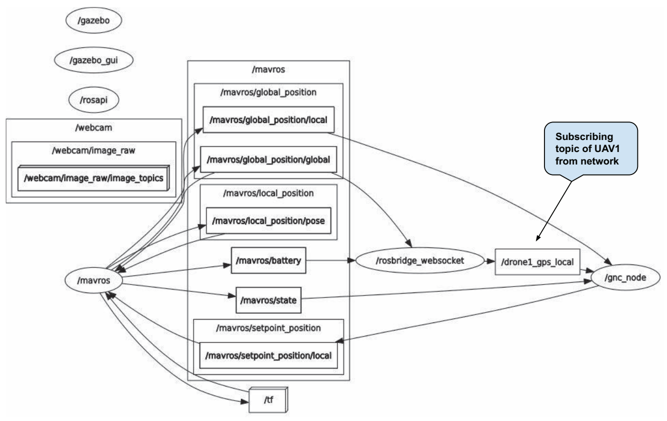
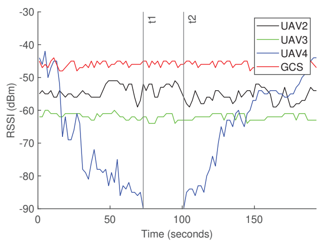

/gazebo
/gazebo_gui
/rosapi
/webcam
/webcam/image_raw
/webcam/image_raw/image_topics
/mavros
/mavros/global_position
/mavros/global_position/local
/mavros/global_position/global
/mavros/local_position
/mavros/local_position/pose
/mavros/battery
/mavros/state
/mavros/setpoint_position
/mavros/setpoint_position/local
/rosbridge_websocket
/drone1_gps_local
/gnc_node
/tf
Subscribing topic of UAV1 from network

Fig. 8: ROS-node graph of Follower

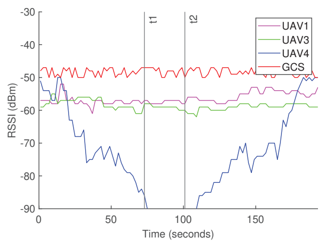
coordinates and IMU readings with each other and GCS over a mesh network through modems. The GPS coordinates assigned by the gazebo to all the UAVs are monitored on GCS throughout the mission. The formation of UAVs at initial point A, between B and C, and at a final point D can be seen in Figure 6. The formation remains the same throughout the mission, as the UAVs are maintaining equal distance on the basis of shared information in the network.

Multiple ROS nodes are running in a particular UAV. UAV1 is acting as a leader and knows the mission. UAV1 needs to share its ROS nodes with other UAVs in the mesh network through a web socket server. We are using ROS node to send information of leader to followers (UAV2, UAV3 & UAV4) after subscribing the topic from rosbridge and then publish the same message on respective IP_sockets of UAVs in the network. The ROS node graph can be seen in Figure 7. Similarly, a ROS-node graph of follower UAVs can be seen in Figure 8. The UAVs acting as the follower will continuously receive information from the UAV acting as a leader and will keep on following the UAV1.

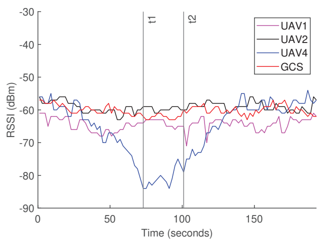
Received Signal Strength Indicator (RSSI) is recorded from the modems connected with the platforms. We see that the RSSI of the UAV4 platform is degraded with respect to the UAV1 platform over time due to channel impairments by including a metallic plate between them. During the time interval of 73 to 101 seconds, the RSSI goes below -90 dBm and the link between UAV1 and UAV4 gets broken.This is shown in Figure 9 where t_1 and t_2 donate the time instances where link gets broken and restored respectively. Figure 9a shows that while the link of UAV1 with UAV2,3 remain intact throughout, link with UAV-4 gets broken during t_1 and t_2. Similar observations can be made from Figure 9b, 9c. In Figure 9d, it can be seen that the link of UAV4 breaks down with UAV1 and UAV2 while it remains connected with UAV3. Moreover, we see that even in the interval when the follower UAV4 platform is disconnected from the leader UAV1, UAV4 still follows the path of UAV1 by maintaining a specified distance as shown in Figure 6b. This is because of the UAV3



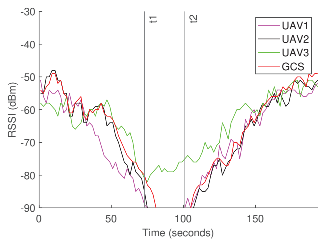
(a) RSSI of UAVs with respect to UAV1



(b) RSSI of UAVs with respect to UAV2



(c) RSSI of UAVs with respect to UAV3



(d) RSSI of UAVs with respect to UAV4

Fig. 9: RSSI of each UAV with respect to other UAVs. Time $t1$ and $t2$ shows the interval in which UAV1 is disconnected with UAV4

platform, which is acting as a relay and providing a path to UAV4 to remain in the network with other UAV's. Thus, all the platforms stay in the network throughout the simulation period. Therefore, to be part of the network, each platform needs at least one direct link with any other platform in the network.

The UAV platform is considered to be out of the network when it is not in direct connection with all of the other UAV platforms and the GCS. As the mission information is only known to the leader UAV, its disconnection will result in all follower UAVs stopping at their positions. The follower UAVs will remain in the network even in the absence of the leader UAV. To continue the mission, the remaining follower UAVs can select a new leader which is not in the scope of this paper and maybe discussed in future work.

VI. Conclusions

In this work, we have proposed and demonstrated a novel multi-platform hardware-in-the-Loop (HIL) emulation mechanism for realistic modeling of swarm communication. The proposed mechanism consists of ROS and Gazebo running on physically isolated platforms connected via wireless modems allowing us to explore real-time communication and its latency. Moreover, the HIL is implemented using the principles of decentralized communication, thereby avoiding a single point of failure, which is crucial to the successful operation of critical missions that the swarm is intended to pursue. Results and discussion of a simple leader-follower swarm formation have been provided as an example.

Acknowledgment

This research work is carried out at Swarm Robotics Lab under National Centre for Robotics and Automation (NCRA). The NCRA is funded by Higher Education Commission (HEC), Pakistan. The authors acknowledge the NCRA and HEC for the technical and financial support.

References

[1] L. Chaimowicz and V. Kumar, *Aerial Shepherds: Coordination among UAVs and Swarms of Robots*, 01 2007, pp. 243–252.

[2] M. Egerstedt and X. Hu, "Formation constrained multi-agent control," *Robotics and Automation, IEEE Transactions on*, vol. 17, pp. 947 – 951, 01 2002.

[3] W. Kowalczyk, "Target assignment strategy for scattered robots building formation," in *Proceedings of the Third International Workshop on Robot Motion and Control, 2002. RoMoCo '02.*, 2002, pp. 181–185.

[4] R. O. Saber, W. B. Dunbar, and R. M. Murray, "Cooperative control of multi-vehicle systems using cost graphs and optimization," in *Proceedings of the 2003 American Control Conference, 2003.*, vol. 3, 2003, pp. 2217–2222 vol.3.

[5] S. Zelinski, T. Koo, and S. Sastry, "Optimization-based formation reconfiguration planning for autonomous vehicles," 10 2003, pp. 3758 – 3763 vol.3.

[6] Z. Cao, M. Tan, S. Wang, Y. Fan, and B. Zhang, "The optimization research of formation control for multiple mobile robots," 02 2002, pp. 1270 – 1274 vol.2.

[7] B. Elkilany, A. Ali, A. Fath El Bab, and H. Ishii, "A proposed decentralized formation control algorithm for robot swarm based on an optimized potential field method," *Neural Computing and Applications*, 05 2020.

[8] A. M. de Souza Neto and R. A. F. Romero, "A decentralized approach to drone formation based on leader-follower technique," in *2019 Latin American Robotics Symposium (LARS), 2019 Brazilian Symposium on Robotics (SBR) and 2019 Workshop on Robotics in Education (WRE)*, 2019, pp. 358–362.

[9] P. Petráček and M. Saska, "Decentralized aerial swarms using vision-based mutual localization," 11 2018.

[10] T. Indriyanto, A. Rizki, M. Hariyadin, M. Akbar, and A. Syafi, "Centralized swarming uav using ros for collaborative missions," vol. 2226, 04 2020, p. 030012.

[11] A. Lamping, J. Ouwerkerk, N. Stockton, K. Cohen, M. Kumar, and D. Casbeer, "Flymaster: Multi-uav control and supervision with ros," 06 2018.

[12] W. Garage, "Robot operating system," 2007, date accessed 05-11-2020. [Online]. Available: https://www.ros.org/about-ros/

[13] "Gazebo," 2018, date accessed 03-10-2020. [Online]. Available: http://gazebosim.org/

[14] R. Wandarosanza, B. R. Trilaksono, and E. Hidayat, "Hardware-in-the-loop simulation of uav hexacopter for chemical hazard monitoring mission," in *2016 6th International Conference on System Engineering and Technology (ICSET)*, 2016, pp. 189–193.

[15] Y. A. Prabowo, B. R. Trilaksono, and F. R. Triputra, "Hardware in-the-loop simulation for visual servoing of fixed wing uav," in *2015 International Conference on Electrical Engineering and Informatics (ICEEI)*, 2015, pp. 247–252.

[16] E. Rohmer, S. P. N. Singh, and M. Freese, "V-rep: A versatile and scalable robot simulation framework," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 1321–1326.

[17] Geo-Fs, "Geo-fs free online flight simulator," date accessed 01-03-2021. [Online]. Available: https://www.geo-fs.com/

[18] H. KORKMAZ, O. B. ERTİN, C. KASNAKOĞLU, and ünver KAYNAK, "Design of a flight stabilizer system for a small fixed wing unmanned aerial vehicle using system identification," *IFAC Proceedings Volumes*, vol. 46, no. 25, pp. 145–149, 2013, 1st IFAC Workshop on Advances in Control and Automation Theory for Transportation Applications. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S147466701535223X

[19] I. Lugo-Cárdenas, S. Salazar, and R. Lozano, "The mav3dsim hardware in the loop simulation platform for research and validation of uav controllers," in *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2016, pp. 1335–1341.

[20] Gazebo, "Player/stage/gazebo," 2017, date accessed 01-03-2021. [Online]. Available: https://sourceforge.net/projects/playerstage/

[21] C. E. Agüero, N. Koenig, I. Chen, H. Boyer, S. Peters, J. Hsu, B. Gerkey, S. Paepcke, J. L. Rivero, J. Manzo, E. Krotkov, and G. Pratt, "Inside the virtual robotics challenge: Simulating real-time robotic disaster response," *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 2, pp. 494–506, 2015.

[22] "mavros - ros wiki," 2018, date accessed 21-10-2020. [Online]. Available: http://wiki.ros.org/mavros

[23] "roslibjs - ros wiki," 2018, date accessed 05-11-2020. [Online]. Available: http://wiki.ros.org/roslibjs

[24] K. Alisher, K. Alexander, and B. Alexandr, "Control of the mobile robots with ros in robotics courses," vol. 100, p. 1475–1484, 2015.

[25] O. contributors, "leaflet," 2020, date accessed 05-11-2020. [Online]. Available: https://leafletjs.com/

[26] J.-J. Xiong and G.-B. Zhang, "Global fast dynamic terminal sliding mode control for a quadrotor uav," *ISA Transactions*, vol. 66, pp. 233–240, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0019057816303627

[27] J. P. Snyder, "Map projections: A working manual," Washington, D.C., Tech. Rep., 1987, report. [Online]. Available: http://pubs.er.usgs.gov/publication/pp1395

[28] Microhard, "Microhard pmddl2450," date accessed 06-11-2020. [Online]. Available: http://microhardcorp.com/pMDDL2450.php

[29] E. Johnson, "Intelligent quads community," 2020, date accessed 05-11-2020. [Online]. Available: https://github.com/Intelligent-Quads/iq_tutorials