IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 18 August 2023, accepted 8 September 2023, date of publication 20 September 2023, date of current version 2 October 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3317434

RESEARCH ARTICLE

Mixed-Reality Based Multi-Agent Robotics Framework for Artificial Swarm Intelligence Experiments

DILSHANI KARUNARATHNA[1], (Member, IEEE), NUWAN JALIYAGODA[1], GANINDU JAYALATH[1], JANAKA ALAWATUGODA[2,3], ROSHAN RAGEL[1], (Senior Member, IEEE), AND ISURU NAWINNE[1]

[1]Department of Computer Engineering, Faculty of Engineering, University of Peradeniya, Peradeniya 20400, Sri Lanka
[2]Research and Innovation Centers Division, Faculty of Resilience, Rabdan Academy, Abu Dhabi, United Arab Emirates
[3]Institute for Integrated and Intelligent Systems, Griffith University, Nathan, QLD 4111, Australia

Corresponding authors: Dilshani Karunarathna (dilshanik@eng.pdn.ac.lk), Nuwan Jaliyagoda (nuwanjaliyagoda@eng.pdn.ac.lk), and Janaka Alawatugoda (jalawatugoda@ra.ac.ae)

This work was supported in part by the University of Peradeniya, Sri Lanka; and in part by the Rabdan Academy, United Arab Emirates, under research funding RF042.

ABSTRACT The term ''Swarm intelligence'' outlines a broad scope, which is generally defined as the collective behaviour of many individuals towards a certain task, each operating autonomously in a decentralised manner. Swarms are inspired by the type of biological behaviours of animals and this technology involves decentralised local control and communication, which ensures the problem-solving process is more efficient through modification and infusion into swarm robotic technology. Swarm robotics related experiments and applications are relatively expensive compared to other types of robotics experiments. The most common solution is to use computer based virtual simulations and accordingly, this study introduces a reliable and cost-effective hybrid solution using Mixed Reality (MR) technologies to perform swarm experiments on real robots and virtually deployed robots simultaneously. MR bridges the gap between the virtual and physical realms through Augmented Reality (AR) and Augmented Virtuality (AV) and addressing the weaknesses of simulators and providing a more accurate representation of real-world performance. This approach provides a platform for researchers to explore swarm robotics with enhanced realism and interactivity and conduct experiments that mimic real-world conditions. To validate this method we conducted experiments with both physical and virtual robots in near real-time in a mixed reality environment. In addition, an open-source, distributed, and modular mixed reality simulation framework was implemented with support libraries and applications. Our work not only demonstrates the potential of MR in swarm robotics but also emphasizes its value in advancing the understanding and application of swarm intelligence principles in practical contexts.

INDEX TERMS Mixed-reality, multi-agent robotics, swarm intelligence, artificial intelligence, virtual reality.

I. INTRODUCTION

Swarm Intelligence is a rapidly growing field of research, which focuses on the coordination and collaboration of multiple robots to achieve a common goal in a decentralised manner [1], [2], [3], [4]. Swarms are inspired by the biological behaviours of animals, such as ants and bees [5], [6].

Inspired by the collective behaviour of biological creatures and their advantages (robustness, adaptability, and scalability), researchers are developing new approaches to enhance the process of solving problems more efficiently through

The associate editor coordinating the review of this manuscript and approving it for publication was Lei Wei.

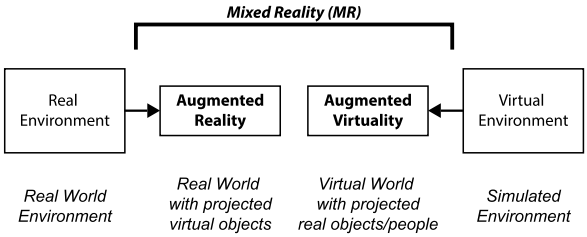

FIGURE 1. Reality virtuality continuum introduced by milgram and kishino.

cooperation and division of labour in various domains, such as search and rescue operations, environmental monitoring, and industrial automation.

A. CHALLENGES IN SWARM ROBOTICS

Designing, controlling, and experimenting with a swarm of robots is a challenging task, which requires addressing various technical and practical difficulties. One such challenge is the high cost and time required for setting up a physical experiment with a considerable number of physical robots with diverse features [7].

As a solution to this, computer-based simulations with virtual robots have been predominantly used in the literature [7], [8] (e.g. Player [9], Stage [10], Gazebo [11], ARGoS [12]). Simulations are easier to set up, less expensive, typically faster, and more convenient than physical swarms. The two main advantages of virtual swarm robot simulators are the flexibility to add new features and high efficiency. The persistent problem with the pure virtual approach is that the simulators do not ensure how robots act in an actual environment, how they react to complex physics, noisy sensor data, control loop delays, etc [10].

B. MIXED REALITY IN SWARM ROBOTS

Virtual Reality describes the simulation of a synthetic environment similar to the actual environment [13]. It has been used in flight simulation training for pilots, procedural training for surgeons, treatments for phobias and disorders, gaming consoles, etc.

Mixed Reality is a term introduced by Milgram and Kishino [14], which links both real and virtual environments through Augmented Reality (AR) and Augmented Virtuality (AV) as shown in Figure 1.

Enabling a robot to sense both physical and virtual environments via augmented means allows the ability to interact with both environments. Also, it is a promising solution for reducing the experimental and development costs of robots whereas maintaining the scale and scope of swarm intelligence experiments [15] mainly due to the remote accessibility feature. Also, it creates a safe and low-risk environment for extensive testing of swarm robot behaviour, as well as allowing an unlimited means of achieving aggregations for robots that would otherwise be physically impossible or expensive in reality.

C. OUR CONTRIBUTION

The novelty of this paper lies in its exploration of mixed reality as a framework for conducting swarm behavioural experiments using general purpose robots. Swarm robotics is an emerging field with a range of applications including surveillance and disaster response. Conducting experiments to understand swarm behaviour is essential for optimizing the performance of swarm robotic systems. Our primary goal is to address the following central research question: How can mixed reality technologies enhance swarm robotics experiments by providing a bridge between simulated and physical environments in ways that virtual simulations cannot? To achieve this goal, we explored the potential of mixed reality technologies to facilitate coordination, communication, and sensing with a swarm of robots and designed several basic interaction models to execute experiments with both physical and virtual robots in near real-time in a mixed reality environment. Through this research, we aim to contribute to the growing field of swarm robotics by developing a deeper understanding of the potential benefits and challenges associated with mixed reality.

In this approach, a general purpose robot was introduced with basic functionalities and designed virtual instances to simulate similar functionalities in a virtual manner. Then a distributed framework was implemented to build the bridge between the two realities, and an application was built to visualize the movements and behaviours of physical and virtual robots. The location of physical robots was tracked using an overhead camera, and that data was fed to the simulator to control and pass to the visualizer application.

Our mixed reality based framework for swarm behavioural experiments is presented as a notable contribution to the field. We have put effort into designing and implementing this framework with the intention of enabling researchers to utilize mixed reality in their studies and to use this open-source framework for swarm behavioural experiments. This paper aims to guide researchers in harnessing the power of mixed reality to gain deeper insights into swarm behaviour.

In the subsequent sections of this paper, existing work related to this approach will be discussed to understand the current state of mixed reality swarm robotics, followed by the architectural design and system implementation of our proposed system. Next, the experimental setup will be discussed with some performance tests to analyse the effects of mixed reality on the swarm's collective behaviour and performance. Then the discussion section will dispute the limitations, and considerations and conclude with suggestions for possible future works.

II. RELATED WORKS

In this section, we will discuss the recent related works in this field.

A. MULTIROBOT SIMULATION

Imitating the real world in a virtual environment has become a new trend in the world. On the other hand, robotic simulation

TABLE 1. Comparison of some existing simulators by considering the reality of robots, sensors, the environment, and the performance.

Parameter	Simulators							
	Argos [12]	**Stage** [10]	**Gazebo** [11]	**Webots** [17]	**USARSim** [16]	**SwarmSimX** [19]	**Morse** [18]	**Microsoft Robotics Studio** [20]
Support robots								
Special model	Variety of robots	Player controllers	Variety of robots	Any mobile robot	Variety of robots	Variety of mobile robots	Large variety of heterogeneous	Variety of robots
Physical robots	Yes	No	No	No	No	No	No	No
Virtual robots	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Support swarm robotics simulation	Directly	Directly	Via an extension/ programming	Via an extension/ programming	Directly	Directly	Via an extension/ programming	Directly
Scalability								
Number of maximum robots can be simulated	10000 simple robots	100000 simple robots, 100 complex robots	Depends on the robot	Depends on the robot	Depends on the robot	Depends on the robot	simple 15 robots	30 robots
Mixed reality integration								
Virt and Phy robots together	No	No	No	No	No	No	No	No
Virtual Sensors	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Virtual Sensors for physical robots	No	No	No	No	No	No	No	No
Virtual Environment	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Virt and Phy Environments in same time	No	No	No	No	No	No	No	No
Customizable environment	Yes	2 ready-made	Yes	Yes	Yes	Yes	Yes	Yes
Visualization								
Visualization techniques	Inbuilt	Inbuilt	Inbuilt	Inbuilt	Inbuilt	Inbuilt	Inbuilt	Inbuilt
Virtual arena through display	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
AR view through camera	No	No	No	No	No	No	No	No
3D	3D	2.5D	3D	3D	3D	3D	3D	3D
Performance requirement								
	Can run on PC	Can run on PC	Can run on PC	Can run on PC	Can run on PC	Need to run on a server	Can run on PC	Can run on PC
Multithreading in simulation	Yes	No	Yes	Yes	No	Yes	No	Yes
Open-source	Yes	Yes	Yes	Yes	Yes	No	Yes	No
Extensible	No	No	Yes	Yes	Yes	No	No	No
Simulation mechanism	Discrete Time	Discrete Time	Discrete Time	Discrete Time	Discrete Time	Discrete Time	Discrete Time	Discrete Event
Architecture	Monolithic	Monolithic	Distributed	Distributed	Hybrid	Distributed	Monolithic	Monolithic

reduces the cost, and the system delivery time, and also provides proof of concept and proof of the design to ensure that there are no flows in robotic systems [16].

The Stage is one such simulator that simulates multiple mobile robots in a 2.5D environment [10]. The Gazebo simulator is a 3D simulator that was developed based on the Stage simulator [11]. Similar to the features of the Gazebo simulator, Webots [17], and Morse [18] simulators also provide a 3D simulation for a customizable virtual environment and support swarm robotics via an extension.

Two other simulators used for swarm robotic simulation were SwarmSimx [19]. However, they were not open source and were designed for commercial purposes. USARSim [16] and Microsoft robotics studio [20] also provide a virtual simulation environment for robotic experiments with virtual sensors and other features.

The ARGoS simulator, which is one of the widespread simulators, provides the ability to simulate swarm robot agents. It provides either virtual robot simulation or remote control of the physical robots via a set of APIs. However, it doesn't support both physical and virtual robots being simulated together [12].

Table 1 shows a comparison of existing robotics simulation platforms. Some of the simulators listed here follow a monolithic architecture in which all the functionally distinguishable aspects are interconnected without separating the components (e.g. Stage, Morse, and Microsoft robotics studio). On the other hand, some followed a distributed architecture in which the components were presented on different platforms and several components could collaborate with each other through a communication network to achieve a specific goal or objective (e.g. Gazebo, Webots, SwarmSimX). Monolithic architecture is a design architecture that is simple and easy to handle, whereas distributed architecture is more scalable, loosely coupled, and can be tested individually. Therefore, in terms of scalability and extensibility, a distributed architecture is more appropriate

than a monolithic architecture for the framework to simulate the robots.

Robot simulators were usually discrete-time simulators with fixed time steps (e.g. Stage, ARGoS, Gazebo), whereas network simulators were discrete-event simulators (e.g. NS-3 [21], RoboNetsim [22]) which were specially designed to simulate communication. A discrete-time simulation models the operation of a system at uniform distributed time points and updates the state of the system periodically, whereas discrete event simulation models the operation of a system as a sequence of events in time and changes its state in response to events. There were pros and cons of both these types. However, when it is considered a distributed system, it involves internal communication to a great extent. In such a case, the best method to limit the communication overhead is discrete-event simulation.

TABLE 2. Comparison of some research works by considering the reality of robots, sensors, and the environment.

Research	Robots		Sensors[1]		Setup		Supported Robots
	Phy[2]	Vir[3]	Phy	Vir	Phy	Vir	
Pickem et al. [23]	Yes	Yes	No	Yes	Yes	No	GRITSBot
Reina et al. [24].	Yes	No	Yes	Yes	No	No	e-pucks
de Almeida et al. [25]	Yes	No	Yes	Yes	Yes	Yes	Any
Payton et al. [26]	Yes	No	Yes	Yes	Yes	No	Pherobots
Honig et al. [15]	Yes	Yes	Yes	Yes	Yes	Yes	Quadcopters
Chen et al. [27]	Yes	Yes	Yes	Yes	Yes	Yes	Mobile robots
Reina et al. [28]	Yes	Yes	Yes	Yes	Yes	Yes	Kilobots

[1] The physical and virtual setup here means the possibility of changing the environmental conditions for experiments
[2] Physical
[3] Virtual

B. MIXED REALITY IN SWARM ROBOTICS

Mixed reality involves both physical and virtual robots, where Pickem et al. implemented a remotely accessible testbed for swarm robotics research by building general-purpose miniature differential drive robots with basic functionalities. The physical robot's coordinates were tracked using an overhead camera with AR markers. Furthermore, they were able to add virtual robots to the testbed through the server to interact with physical robots. The server controls user code, routing commands, data transmission, positioning, and virtual robots to increase the overall robustness of the system. However, physical robots were only capable of moving, and there were not any hardware-level sensors. Also, there was a lack of support for the environment setup for different types of experiments [23].

Reina et al. implemented a system that includes virtual sensors for swarm robots. The movements of the physical robots were tracked using markers, which were composed of a matrix of 6 cells (2×3), similar to a very simple QR code. They used the ARGoS simulator since it allows them to simulate complex experiments with different types of swarm robots [24].

Similarly, de Almeida et al. also worked on a system to develop virtual sensing and communication using mixed reality concepts for very simple robots with limitations. They used AR Tags to identify objects in the physical environment [25].

None of the above listed works involves a virtual environment setup. Nevertheless, Payton et al. used virtual pheromones with swarms of robots for communication and processing. The virtual pheromones were implemented via optically transmitted signals from each robot and guided the physical robots to a certain destination. The signals were decoded into arrows and were visible via the Augmented Reality (AR) head-mounted display system with the camera [26].

Another such realisation was implemented by Honig et al. which followed a target with virtual and physical quadcopters. They used Crazyflie 2.0, an open-source nano quadcopter, and the Unity 3D game engine (4.6.2 Free Edition) combined with the Virtual Human Toolkit to simulate the environment and humans [15].

Chen et al. simulated virtual components such as robots and sensors using the Gazebo simulator while observing the results of real robot behaviours. They tracked four coplanar points and used them to derive the camera's translation and rotation parameters to track the location of robots. Meanwhile, they used a planar object detection algorithm to recover when the physical robot tracking failed [27].

Due to the scale and time requirements associated with calibration, sensing, tracking, and control challenges, Reina et al. introduced an AR system for Kilobots. This system is controlled by a base station, which allows users to calibrate robots, configure the virtual environment, and plug in unique experiments. However, it limits the capabilities of robots as they only consist of a single sensor [28].

Table 2 shows a summary of a few research works considering physical and virtual robots, sensors, and environmental setups.

The majority of the work focuses only on virtual sensing for robots. However, Honig et al. considered mixed reality sensing and environment for virtual and physical quadcopters, whereas Chen et al. considered those features for mobile robots.

In a nutshell, it is always difficult to choose between developing a control system to match the hardware and matching the hardware to the required controller. A solution to this problem is rapid prototyping of the hardware in parallel to software implementation. As a result, the benefits of both pans could be achieved with minimal time and effort while also reducing their weaknesses. Prototyping is very common in the software domain, but it is rare in hardware since it is slow and expensive [11].

To the best of our knowledge, we were unable to locate a solution that satisfies all of the requirements. Therefore, taking all this into account, a more generalised

simulation framework for swarm behavioural experiments was introduced in the mixed reality domain.

III. METHODOLOGY

In this section, we will provide a detailed description of the methodology.

A. SYSTEM OVERVIEW

Simulation in a mixed reality environment requires mechanisms to interact with the different realities. This includes how the virtual simulated objects interact with the real objects and vice versa. This chapter is focused on identifying protocols and mechanisms to conduct swarm robotics-related experiments in a mixed reality environment in near real-time. As the first step, the following subsystems were identified as required in a mixed reality multi-agent simulation system.

1) Physical Robots
2) Virtual Robot instances
3) Visualizer
4) Reality simulator/integrator

1) PHYSICAL ROBOTS

Swarm robotic studies usually involve wheel-driven robots with sensors to observe the surroundings while indicating internal changes with lights. Also, they need to communicate and interact with other robots to facilitate swarm behaviour.

2) VIRTUAL ROBOTS

Virtual robot agents will be the same as the physical robots in functionality but only exist in the virtual domain. The virtual robot implementation makes it possible to initiate as many virtual robot instances as required for the experiments whereas there are a constrained amount of physical robots. Those implemented virtual instances should be able to emulate[4] the behaviours of real robots.

3) VISUALIZER

Since the simulation involves both physical and virtual realities, it was necessary to have a method to visually observe the behaviours and interactions of both realities in a single environment. To fulfil that requirement, an application was needed that can render and visualise the movements of robots in both realities.

There were two possible methods for visualising the behaviour of robots. The first method was to implement a virtual space defined at the software level and render the robots of both realities in that virtual space, enabling users to view the virtually rendered environment through the application window.

The second method was using AR technologies to project the virtual robots into the physical space on top of the video feed taken by a camera [29]. This is still a developing technology, but it is a more realistic, helpful, and interactive method of visualising.

4) SIMULATOR

In order to carry out an experiment in a mixed reality domain, each robot in either reality should be able to interact with the robots and environmental elements in both realities. A novel framework that facilitates real-time inter-reality integration was proposed to meet this requirement. That framework will include a set of communication protocols, data structures, and workflows to support seamless integration. When considering the swarm behaviour of robots, four main types of interactions among realities that the simulator should support were identified.

1) Motion and Localization
2) Environment Building
3) Sensing
4) Communication

B. MOTION AND LOCALIZATION

From the perspective of the simulation, the locations and movements of the physical robots need to be tracked. There are adequate robot localization technologies such as MEDUSA [30], RSSI [31], GPS [32], and AR Markers [33]. Considering the indoor use, simplicity in implementation, and extendibility, it was decided to use an overhead camera with AR markers to identify the absolute coordinates of each physical robot.

Since the virtual robots are in the virtual domain, a mathematical model was required to calculate and update the motion of the robots based on the control actions such as moveForward, turnLeft, turnRight, rotate, etc. In most robotics research, robots are based on two-wheel differential drives. Therefore, the Dead Reckoning method [34] can be used to estimate the current position of the robot based on its previous position and motion parameters.

The raw ''Dead Reckoning'' method is subjective to considerable cumulative errors in the physical applications. This can be minimised for virtual robots by using high precision floating-point arithmetic.

The designed workflow of the robot localization is shown in Figure 2. The movements of the physical robots can be tracked by an overhead camera, while virtual robots can calculate their own movements based on their control actions. Both the camera-based localization system and the virtual robots send coordinate updates to the simulator application.

In this study, only 2-dimensional movements were considered in the x-y plane of robots. Due to the need for real-time adaptation and reduced computational complexity and communication overhead, other movements caused by actuators like grippers and servos were not implemented. However, a similar technique might be employed to implement them.

[4]The term ''emulator'' refers to a functional service that is responsible for providing virtual actuator/sensor support. For example, virtual robots do not have actual distance sensing capabilities. Hence, a distance emulator will mimic the functionality of distance sensors for the connected virtual robot instances.

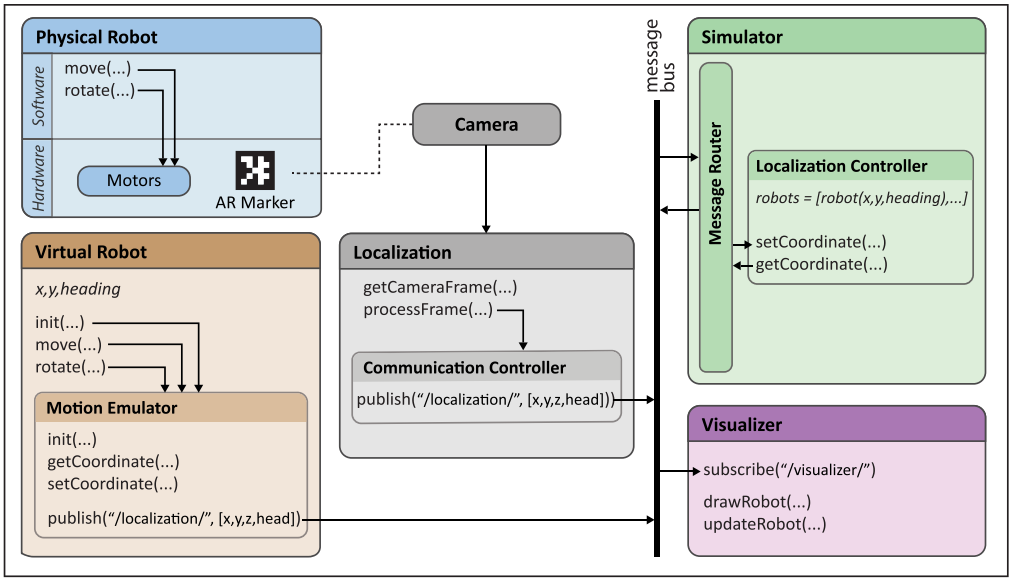

FIGURE 2. The architecture of handling the localization in the mixed reality domain. The overhead camera monitors and tracks the positions of the physical robots, and the Localization Sub-system sends those coordinates to the Localization Controller in the Simulator. The virtual robots should be specified with their initial coordinates. During the experiment, the virtual robots calculate coordinates by themselves through the Motion Emulator and send occasional updates to the Localization Controller in the simulator. The simulator sends the update to the Visualization subsystem when there is a change in robot coordinates.

C. ENVIRONMENT SETUP

When conducting experiments with swarm behaviours, a specific environmental setup might often be required. However, it can be difficult or time-consuming to create or modify the environment physically.

Nevertheless, it is easier to create a virtual environment with mixed reality technologies. Object-oriented concepts can be followed to define primitive shapes such as boxes, cylinders, and spheres by incorporating geometric details (dimensions such as width, height, positioning coordinates, orientation, etc.) and properties like 'colour' as attributes.

Likewise, to represent the physical environment details in the mixed reality domain, the same methodology can be used, and geometric parameters can be provided similar to the virtual obstacles but with a property flag to indicate the reality of the obstacle as a physical obstacle.

For instance, a primitive box shape could be used to model a wall obstacle. In addition to that, it was required to provide implementations for a couple of methods to support reality integration.

With respect to the Figure 3 if the robot R_1 needs to determine whether or not the wall obstacle O_1 is in front of it, this can be achieved mathematically using the geometrical properties and the coordinates of the obstacles. To begin, calculate the angles between the obstacle's two endpoints P_1 and P_2 relative to the heading angle (θ) of the robot using equation (1) and equation (2).

$$\gamma_1 = \theta - \alpha_1 \tag{1}$$

$$\gamma_2 = \theta - \alpha_2 \tag{2}$$

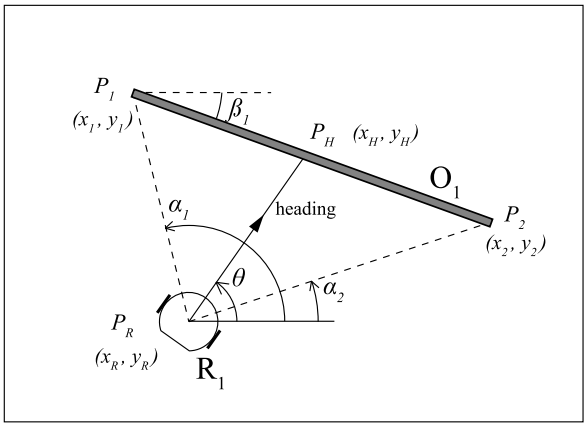

FIGURE 3. A graphical representation of calculating the distance from a robot to a wall by the simulator application using the placement parameters.

If one of them is positive and the other angle is negative, and the absolute value of an angle is less than 90 degrees, O_1 will be identified as in front of the robot. The following logical statement (equation (3)) governs the exact condition of the action. If this condition is satisfied, O_1 is identified as being in front of the robot; otherwise, it is not in the vicinity of the robot.

$$(\,|\gamma_1| \leq 90\; or\; |\gamma_2| \leq 90\,)\; and\; (\gamma_1\gamma_2 < 0) \tag{3}$$

If the simulator needs to calculate the distance from a point to the obstacle through a specific direction, first, it needs to obtain the line equations of the wall obstacle and the line

equation of the robot,R_1 through the specific direction by using this general equation (4) with the corresponding angle and the coordinates.

$$ax + by + c = 0 \tag{4}$$

Where,

a = sin(angle)
b = − cos(angle)
c = − x_0 sin(angle) + y_0 cos(angle)

Equation (5) is the line equation through the heading direction obtained by applying the angle and the coordinate of the heading point to the above equation (4) (angle=θ, x_0=x_R, y_0=y_R),

$$sin(\theta)x - cos(\theta)y - x_R sin(\theta) + y_R cos(\theta) = 0 \tag{5}$$

Equation (6) is the line equation of the wall obstacle obtained by applying the angle and the coordinate of the P_1 endpoint to the above equation (4) (angle=β_1, x_0=x_1, y_0=y_1),

$$sin(\beta_1)x - cos(\beta_1)y - x_1 sin(\beta_1) + y_1 cos(\beta_1) = 0 \tag{6}$$

Then the point of intersection of two lines can be found by considering the coefficients of the two-line equations (equation (7) and equation (8)). Here, a_1, b_1 and c_1 are the coefficients of the line equation through the heading direction, and a_2, b_2 and c_2 are the coefficients of the line equation of the wall obstacle.

$$x_H = \frac{b_1c_2 - b_2c_1}{a_1b_2 - a_2b_1} \tag{7}$$

$$y_H = \frac{a_2c_1 - a_1c_2}{a_1b_2 - a_2b_1} \tag{8}$$

The equation (9) was then used to calculate the distance from the coordinates of the robot to the intersection point by applying the coordinates of the robot (x_R, y_R) and the intersection point (x_H, y_H).

$$distance = \sqrt{(x_R - x_H)^2 + (y_R - y_H)^2} \tag{9}$$

Using this approach, such mathematical models can be implemented for different types of obstacles in a modular manner. The simulator application can invoke those mathematical models, to determine the possible interactions with the robot agents.

D. SENSING IN MIXED REALITY

As previously mentioned, the mixed reality environment contains both physical and virtual robots and environmental objects. For most swarm behavioural experiments, robots must have sensors to explore the environment.

Physical robots will have physical sensors that can obtain physical parameters such as distance, colour, proximity, etc. However, these sensors can not sense virtual objects in the experiment environment.

Therefore, an external application should support sensing for swarm robot agents in both realities. For example, when a physical robot takes a distance sensor reading, the distance sensor will return the actual distance reading of the nearest physical object in front of the sensor. At the same time, the physical robot can request virtual distance sensor readings from the simulator, and the simulator should be able to consider the virtual environment setup and reply with the distance sensor response related to the virtual objects.

The same procedure can be followed for virtual robots. They can request the distance sensor readings from the simulator, which will consider both physical and virtual obstacles defined within the simulator to calculate the distance sensor response (The simulator can use the methods described in the section III-C to calculate the distances).

Figure 4 shows the suggested workflow of the sensor reading by a physical robot and a virtual robot.

E. COMMUNICATION IN MIXED REALITY

Inter-agent communication is an important area of swarm behavioural research. Due to the mixed reality approach, the communication between physical and virtual robots can be modelled virtually with the support of the swarm simulator application.

Similar to sensing in mixed reality, physical robots can have hardware-based communication interfaces with the support of the simulation application while implementing communication emulators for virtual robots. However, it is possible to have entire virtual communication modules for both physical and virtual robots. Accordingly, experiments with various software-defined inter-robot communication techniques can be implemented without using costly hardware modules.

The simplest form of communication is to broadcast messages to nearby robots within a given radius. This is referred to as ''simple communication'' for the rest of this article.

In the ''Simple Communication'' workflow shown in Figure 5, robots can broadcast messages to nearby robots within a defined radius (The radius of transmission can be defined by the robot itself or in the Communication Simulator module). When a robot transmits a message via ''Simple Communication'', it transmits that message to the 'communication-out' channel of the simulator (from the robot's perspective). Once the simulator receives the message, it considers the robot's coordinates and determines which robots are eligible to receive the message. This can be easily modelled using Euclidean distance geometric equations.

Let us assume that (x1, y1) and (x2, y2) are two points in a two-dimensional plane. The Euclidean distance can be obtained using the below equation.

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{10}$$

Then the simulator sends this message into the 'communication-in' channels of the selected robots, and the robots will receive it as a communication interrupt.

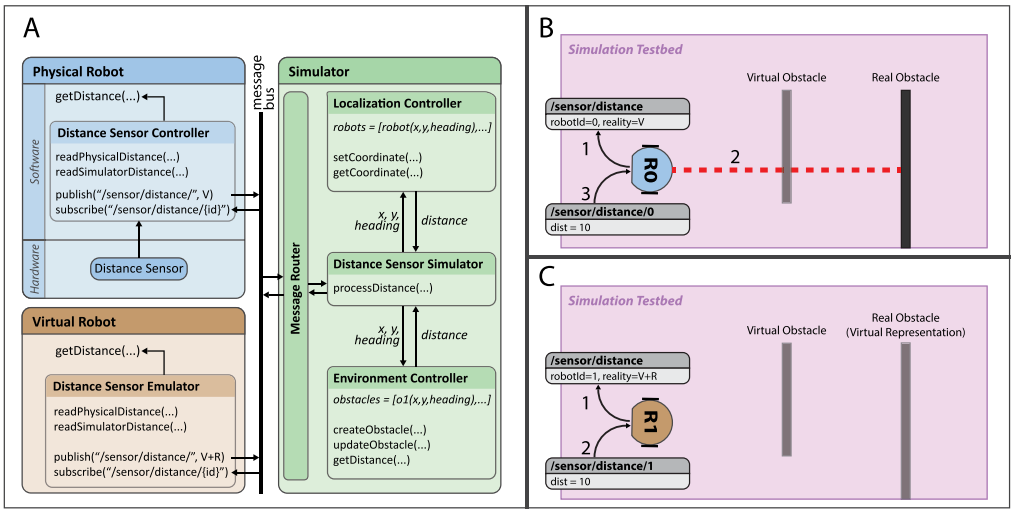

FIGURE 4. A) Component diagram with the workflow for sensation in mixed reality. B) Demonstrates how a physical robot senses virtual and physical obstacles. The physical robot measures the distance to the physical obstacle using the distance sensor hardware and requests the reading of the virtual domain from the simulator. C) Demonstrates how a virtual robot senses virtual and physical obstacles. The physical objects must be provided into the simulator as obstacles, with a flag indicating that reality is "Real". Then the virtual robot requests the distance sensor readings, considering both realities from the simulator.

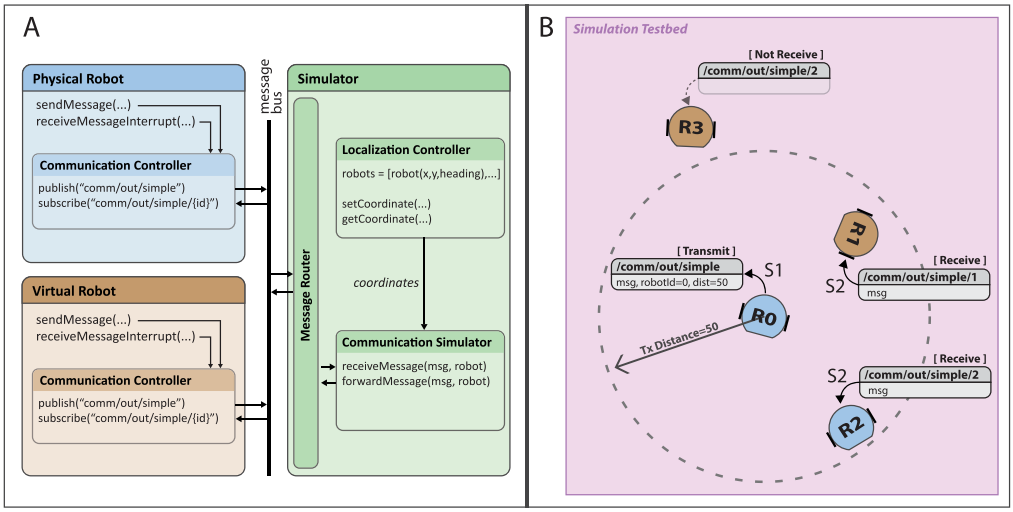

FIGURE 5. A) Component diagram for communication in mixed reality. B) Demonstrate a scenario where the robot R0 broadcasts a message to robots within 50 units of distance. The robots R1 and R2 will receive the message and the robot R3 will not receive the message.

Similarly, different types of communication protocols can be implemented in the simulator, with support messages passing between the robots and the simulator.

IV. IMPLEMENTATION AND EXPERIMENTS

This section describes the "Pera Swarm" simulation framework, our solution that makes it possible to simulate and visualise multi-robot systems in a mixed reality environment, emphasising swarm behaviour. The next chapter will discuss some of the experiments conducted to prove the functionality of the system with its outcome.

A. SIMULATION FRAMEWORK

Figure 6 depicts the overall design of the simulation framework and how the main components interact with each other. This system was implemented as a distributed system by considering the scalability, flexibility of implementation, and execution. When implementing a simulation application with distributed components, it is necessary to consider an effective communication method between nodes. Considering the nature of the subsystems, a repository architecture was used to handle the communication, which means there is a central data repository, and each subsystem publishes or subscribes to information from that repository.

1) COMMUNICATION

A network communication protocol named MQTT was used to communicate between these components. It is a lightweight, publisher-subscriber based protocol that allows

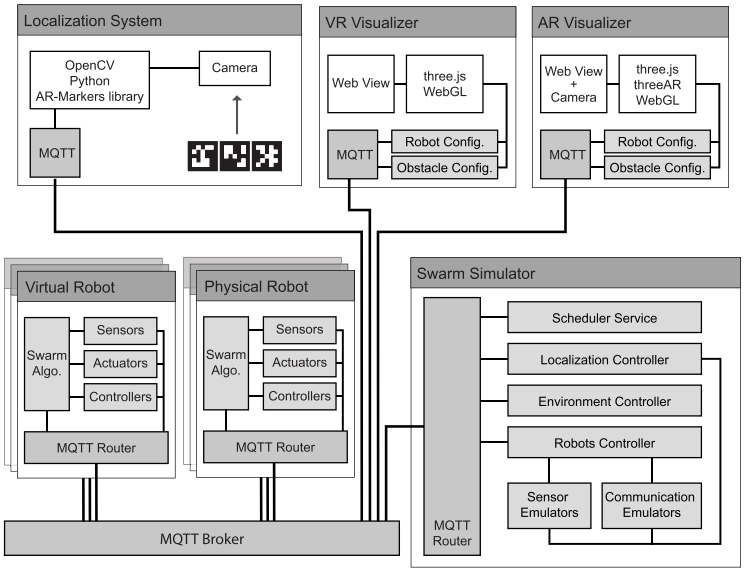

FIGURE 6. Design of the simulation framework of the simulation system, including all the sub-components and their interactions.

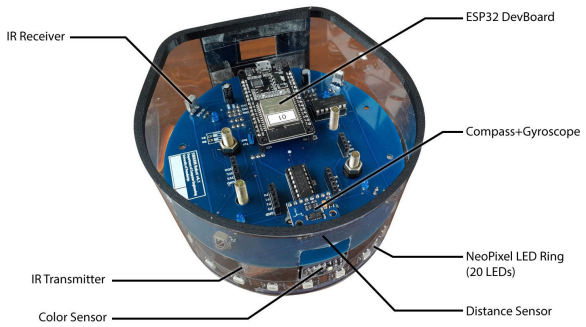

FIGURE 7. Preliminary design of a robotic agent, with sensors and actuators.

communication between remote devices using a small code footprint and less network bandwidth.

2) PHYSICAL ROBOTS

For the requirements of the physical robots, a low-cost, general-purpose robot was designed, especially targeting the swarm behavioural experiments. The driving mechanism of the robot was based on a differential drive with two DC geared motors. The robot contains various types of sensors, such as distance, colour, acceleration, compass, hall effect, etc. Furthermore, the robot has 4 IR transmitters and receivers for IR-based communication. In addition to that, it has BLE (Bluetooth Low Energy), WiFi communication functionalities, and 20 RGB LED indicators (NeoPixel), which can be individually controlled to represent the robot and program status. The firmware was written in C++ based on a modular design. The design of the robot that was created is depicted in Figure 7.

An overhead camera with AR markers on top of the robots was used to identify the coordinates of the physical robots. It uses image processing with OpenCV2 and ARMarker support libraries.

Virtual robots were designed and implemented to be similar in function to physical robots. All the sensors and emulators were implemented virtually using Java classes. It can be provided with a unique ID, starting coordinates, and an initial heading direction for each robot at the beginning of the robot instances. Each virtual robot instance was executed in parallel threads. Those instances can be executed on a cluster of computers. As an example, a cluster of 5 computers can handle 100 virtual robot instances in such a way that each computer handles 20 robot instances.

The simulator application follows a modular architecture with different interfaces and modules for the main behaviours. Each module contains a blueprint and programming interfaces that can be easily reused to implement new sensors/communication methods, environmental objects, etc. Detailed information can be found in the appendix B.

The visualizer (Figure 8) represents the mixed reality information according to the simulator configuration and robot instances. The information exchange ensues through an MQTT connection that follows the standardised communication protocols that have been implemented.

B. EXPERIMENT SETUP

This section briefly reviews the procedure of conducting swarm intelligence-based experiments using the mixed reality simulator and its components.

Before starting an experiment, it is necessary to set up the environment. Obstacles can be added to the simulation testbed by defining them in a configuration file. The physical

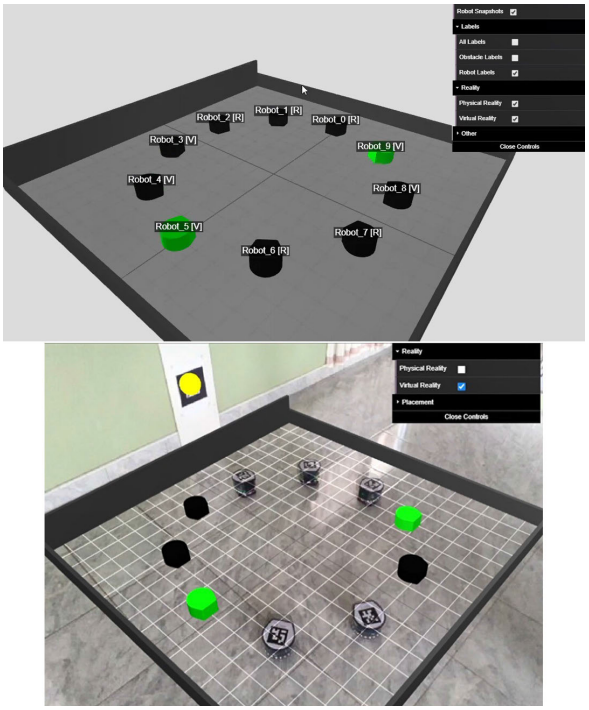

FIGURE 8. Preview of implemented Virtual Reality Visualizer (left) and the AR Visualizer (right) during an experiment.

objects in the testbed, such as walls and other environmental elements, should be defined along with the virtually added obstacles.

The next step is to implement the algorithms in the virtual and physical robots by following the provided libraries and code samples. Furthermore, the framework allows the addition of new functionalities to the system with minimum effort if necessary.

Physical robots can be added manually to the physical robot testbed. The localization system will identify and register these robots in the simulator. It was required to initiate the virtual robot instances on a computer by providing the robot IDs, starting coordinates, and heading directions.

Once the experiment setup was completed, a start command was sent to the robots, and the behaviour of both physical and virtual robots can be observed through the visualisation application.

C. PERFORMANCE TESTS

4 experiments were conducted to measure the reliability and functionality of the system. An Ubuntu 18.04 (LTS) x64 Cloud Server was used in the SGP region, with 1 CPU and 1 GB of memory to run the simulator and a laptop computer with an Intel i7-7500U 2.7GHz CPU with 2 cores to run the virtual robot simulator during the first 3 experiments. Besides, the 4th experiment was run on a server with Intel(R) Xeon(R) CPU E5-2670 0 @ 2.60GHz (20MB Cache, 32 Threads) since that included hundreds of virtual robot instances.

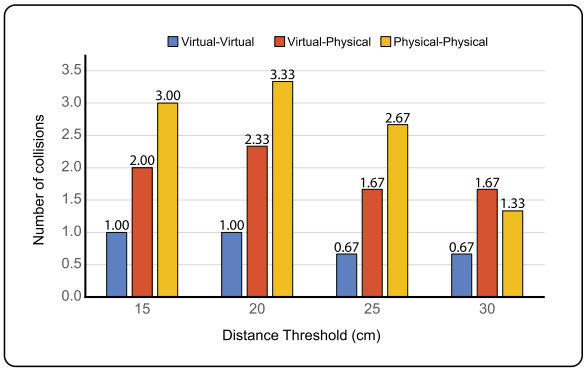

FIGURE 9. The average number of collisions between virtual-virtual, virtual-physical, and physical-physical robots within 3 trials.

The physical robot testbed used for the experiment could accommodate only a limited number of physical robots during the test period. As a result, 10 robots were incorporated (5 physical and 5 virtual) into the first 3 experiments to justify the experiment in both realities. The 4th experiment was specially designed to test the scalability of the simulator with only virtual robots in a large virtual testbed.

1) EXPERIMENT 1 - THE PERFORMANCE OF ROBOT INTERACTIONS WITH THE ENVIRONMENT

Interactions with the environment and other agents are crucial to the success of swarm and multi-agent-based robotics experiments. As a first experiment, the performance of robots interacting with the environment was tested.

Here, a simulation test bed without any obstacles, 5 physical robots, and 5 virtual robots were set up, and the robots were allowed to move randomly. Once a robot detects an obstacle within a given ''distance threshold'', it will turn in a random direction and repeat the movement. The ''distance threshold'' value is the minimum distance needed to maintain between the robot and an obstacle or another robot in front while moving to avoid collisions.

However, collisions between robots may occur depending on the value of the ''distance threshold'' as well as the time taken for mixed reality interaction processing through the simulator application. Collisions between robots were considered in different realities by varying the distance threshold value. Collisions were calculated for a 2 minute period and averaged over 3 trials to test this accurately.

A higher collision rate was observed between physical robots as they encountered laws of physics such as inertia, acceleration, and response delay of motors that are not applicable to virtual robots. (Figure 9).

Moreover, the simulator needs to have the exact locations of the robots to interact with each other. However, the vision-based localization system needs more time to track the motion of physical robots and identify the coordinates compared to virtual robots. Therefore, this delay can lead to a higher collision rate between virtual and physical robots compared to collisions between virtual robots.

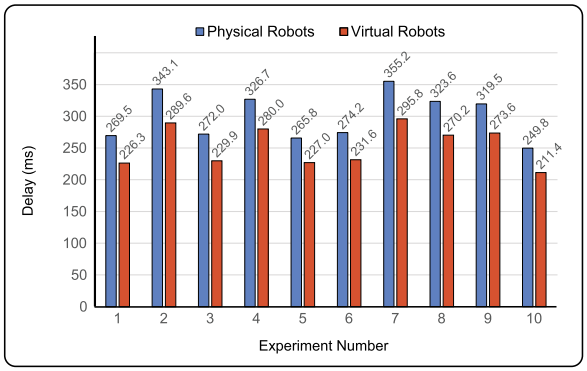

FIGURE 10. Average sensing delay for robots obtained after 10 trials with 5 physical robots and 5 virtual robots moving randomly for 2 minutes.

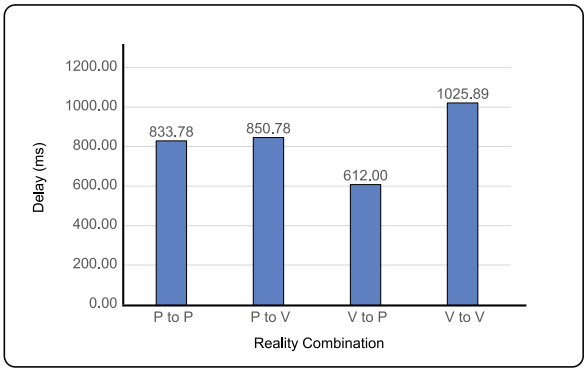

FIGURE 11. Average round trip delay of the messages sent by robots, categorised by source and destination realities.

As the "distance threshold" increases, the probability of collisions decreases because it helps robots to decide and respond to the obstacles beforehand.

Although there have been many smart collision avoiding algorithms for swarm robots, here only the raw behaviours of robots in different realities were considered to observe how they interact in a mixed reality environment. However, depending on the requirements, users can model the virtual robots to behave more similarly to physical robots or fine-tune the physical robots.

2) EXPERIMENT 2 - FUNCTIONALITY AND PERFORMANCE OF MIXED REALITY SENSING OF ROBOTS

Sensor measurements from the environment are a dominant feature that is required in swarm behavioural experiments, and several sensors have been implemented to accomplish this goal. In this experiment, our objective was to test the functionality and performance of mixed reality sensing of the robots. Therefore, the delay in obtaining a distance sensor reading was measured while running an algorithm with the support of the mixed reality simulator. Physical robots request the virtual distance measurements from the simulator, while virtual robots request distance measurements of both realities from the simulator, through their augmented distance sensor.

This sensing had a delay, including the time taken to prepare the request, transfer the request to the simulator over a network, process the request by the simulator, fetch the response by the robot, and decode the response. However, physical robots need to obtain physical sensor measurements in addition to the virtual readings from the simulator. That leads to a higher delay in measurements compared to virtual robots (Figure 10).

The delay between virtual and physical robots differs by 46 ms on average, with a standard deviation of 6 ms. Network delays can cause minor fluctuations in various trials and result in physical robots averaging 300 ms and virtual robots averaging 254 ms.

An additional programmed delay can be introduced as a correction in the virtual robot instances when taking sensor measurements to make them work similarly to the physical robots.

3) EXPERIMENT 3 - INTERCOMMUNICATION OF ROBOTS (ROUND TRIP DELAY)

In the framework designed, there were two types of communication. The first type was communication between the submodules, such as physical robots, virtual robots, the simulator, the visualizer, etc. Inter-robot communication, which actually belongs to the swarm robotics concepts, was the other type of communication.

The previous experiment remarks on communicating with the simulator, and this will perceive the intercommunication of robots. In order to test the efficiency of communication between different realities, the round trip delay of the messages was used.

During the experiment, robots will make message requests to other robots, and the receivers will make acknowledgment messages back. Then the transmitter robot will measure the time duration between the initial message and the reply.

The round trip delay between each robot was calculated for three rounds using three physical robots and three virtual robots. The average values were then computed using the reality of origins and destinations.

As shown in the Figure 11 the round trip delay of virtual to virtual robots is comparatively higher than others. This may be because of the multi-threading of the virtual nodes. All virtual robots were run as separate threads on the same device, whereas physical robots were run on dedicated micro-controller units. The latency of communication between the virtual robots can be minimised by running the virtual robot emulator application on a multi-threaded CPU or a computer cluster. Moreover, the difference in hardware capabilities and processing resources between physical robots and virtual simulations makes the physical-to-virtual round-trip delay higher than that of virtual-to-physical.

4) EXPERIMENT 4 - SCALABILITY OF THE SIMULATOR

Scalability is another main factor expected from a simulation platform. In all three previous experiments, the number

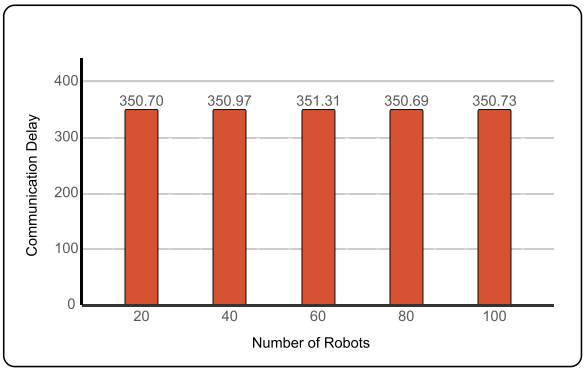

FIGURE 12. Average sensing delay of virtual robots that are moving randomly compared to the number of robots.

of physical robots was limited because of the hardware availability. However, those experiments proved that the simulator could respond to both physical and virtual robots fairly equally. As a result, a virtual robotic approach was used in this experiment to assess the scalability of the simulator, which was one of our research objectives.

The experiment started with 10 robots and gradually increased the number of robots by monitoring the time taken by the simulator for each request to calculate the virtual distance sensor reading and provide that value to the robots. This experiment was performed on a web server with 32 CPU threads since it involves a lot of virtual robot instances.

As shown in Figure 12, the time taken on each request is approximately the same because NodeJS is an asynchronous event-driven JavaScript runtime, having a non-blocking event loop that is specially designed to build scalable network applications.

V. DISCUSSION

After a thorough literature review, we realised that there are similarities and also notable differences in this work. While previous studies have primarily focused on virtual sensing for robots, our study explored the integration of both virtual and physical environments through mixed reality for general purpose robots, so it can be easily extended as rapid prototyping. This study contributed to the literature by providing insights into the design and implementation of a framework to be used for mixed reality swarm behavioural experiments while achieving a seamless transition between simulated and real world scenarios. Also, this will lead to the development of more effective swarm algorithms, foster the creation of advanced simulation platforms, and aid in developing mixed reality technologies tailored specifically for swarms of robots.

By conducting comprehensive simulations and real-world experiments combining both virtual and physical robots, the functionality of the system has been validated despite its reality. Also, we uncovered the advantages and challenges associated with mixed reality environments, declaring future research and development efforts. Introducing virtual robot units or instances to the swarm robotic environment for the conducted experiment was carried out using our local computer hardware configurations, despite the fact that more scalability and performance could be achieved in a cloud computing environment to address the performance overhead. In order to perceive the mixed reality interactions effectively, the simulator and the visualizer need to be updated in real time. However, there was a considerable variation in the delay during the control message transmission that could not be avoided.

As MQTT was used to design protocols for inter-component communication, it ensured confidentiality, integrity, and availability, which are the main components of security, up to some degree. However, apart from the "Quality of Service" (QoS) supported by the MQTT protocol, successful delivery of MQTT packets was not considered. This was because it added unnecessary complexity and generated blocking I/O calls, which affects real-time simulation.

Furthermore, researchers can use this open-source platform to swarm behavioural experiments, and the insights gained from these experiments can be used to improve the framework as well as to design and implement mixed reality swarm systems.

VI. CONCLUSION AND FUTURE WORKS

Swarm robots provide innovative solutions for complex challenges by harnessing collective intelligence. However, the cost and logistical constraints associated with real-world swarm experiments and applications have caused significant barriers to research and deployment. To overcome these challenges, many researchers use virtual simulators, which provide controlled environments for swarm behavioural experiments. Nevertheless, simulators often fail to bridge the gap between virtual experiments and real-world performance due to simplifications and abstractions. In response to these challenges, this study introduced a hybrid solution that can execute swarm behavioural experiments on physical robots and virtually deployed robots at the same time by bridging the gap between virtual simulation and physical implementation. This approach offered several advantages as it combined the benefits of both realities, including flexibility, convenience, low cost, and reliability. The proposed concept of virtual environments can be used to test the functionalities of swarm robots with a vast array of different environmental arrangements swiftly and efficiently rather than physically creating them.

Another possibility is to introduce virtual sensors for physical robots using the concept of mixed reality, which will be useful when conducting experiments with robots that have limited resources and space for some mandatory sensors required for certain experiments that enable more comprehensive and subtle experiments. For example, it may be difficult for a single robot to have eight proximity sensors due to cost and the limitations of the robot controller.

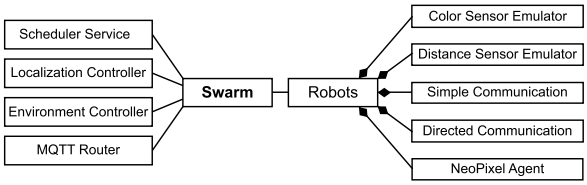

FIGURE 13. The design of the swarm simulator application, with key components and their relationships.

However, it can have a few physical sensors and emulate the rest virtually. Moreover, it has the potential to enhance the models used to depict the real world, bringing them closer to reality.

Furthermore, an open-source library was implemented for general swarm robotic usage, which was also potentially capable of more improvements and addresses specific swarm behavioural requirements. Moreover, this work could be expanded by integrating with custom AR libraries and frameworks for complex real-world applications and facilitating real-time monitoring and control.

The practical implications of our proposed framework were significant across various domains. This includes improved performance and efficiency, flexibility to dynamic situations, cost-effective development and testing, risk mitigation and safety, training and skill development, as well as collaborating and swarm coordination. Mixed reality in swarm robotics can facilitate collaborative exploration in hazardous environments like search and rescue missions, bomb disposal, planetary explorations, etc. Similarly, in environmental monitoring, industrial automation, surveillance and security, and disaster response, mixed reality systems provide a scalable and cost-effective solution, allowing for the planning, optimization, and coordination of swarm activities in a virtual environment, whereas physical robots serve the practical implementation of swarm algorithms and strategies developed in virtual simulations. Apart from the mixed reality simulation, the framework designed can be used by researchers to access the swarm robotics testbed, connect the physically existing robot testbed from a remote computer and conduct the swarm and multi-agent robotics experiments.

Although our research has provided valuable insights, it is important to acknowledge its limitations and consider further implications. One restriction was the real time communication during control message transmission caused by various reasons such as network congestion, limited bandwidth, interference and signal loss, propagation time, synchronisation and protocol overhead etc. In addition, the scalability of mixed reality swarm systems and the complexity of coordinating large swarms in real-time in different domains remain for further exploration.

In conclusion, this work proposes a new approach to overcome cost and realism challenges in swarm robot studies. By adopting a mixed reality based environment, researchers can conduct experiments that bridge the gap between virtual simulations and real-world performance. This novel approach distinguishes our work from existing methods, as it addresses the inherent limitations of virtual simulations and provides a more reliable, insightful, and realistic platform for developing swarm robotics in a wide range of application domains. While challenges and limitations persist, the potential for mixed reality in swarm robotics research and development is promising, raising new opportunities for exploration and innovation in this dynamic field.

APPENDIX A SUPPLEMENTARY DOCUMENTS

The Supplementary Materials for this article and associated work can be found online at https://pera-swarm.ce.pdn.ac.lk/r/mr-sim/. This link includes the open-source implementations of the simulator and support libraries described in this study.

The following sections include additional technical details related to the framework introduced throughout the paper.

APPENDIX B SWARM SIMULATOR

The swarm simulator application was implemented using NodeJS and TypeScript and is available as open-source software via GitHub from the link, https://github.com/Pera-Swarm/swarm-simulator.

It consists of two major components. The first one is the ''pera-swarm'' TypeScript library, which contains all the basic implementations, interfaces, and blueprints for the modules in the library. The second component, the Simulation Application, was implemented based on the TypeScript library, and its functionality can be extended based on the experiment requirements. The module arrangement used for the experiments in this study is shown in Figure 13.

APPENDIX C SWARM VISUALIZERS

The web-based visualisation application was used to visualise experiments in the mixed reality domain, which was developed using JavaScript and TypeScript. The Three.js library was used to generate 3D renderings, which were implemented based on WebGL. Any authenticated user can observe the experiment remotely using a web browser. The applications were also open-source and can be found at the below URLs.

- VR Visualizer: https://github.com/Pera-Swarm/visualizer
- AR Visualizer: https://github.com/Pera-Swarm/visualizer-ar

APPENDIX D ROBOT SIMULATION TESTBED

The simulation testbed is the arena that can be used to accommodate the physical robots during experiments. For the experiments described in this study, a testbed that was 180 cm x 195 cm in size (WxL) was used. An overhead camera was placed above the arena to identify the coordinates of the physical robots. Figure 14 shows the design of the robot simulation testbed.

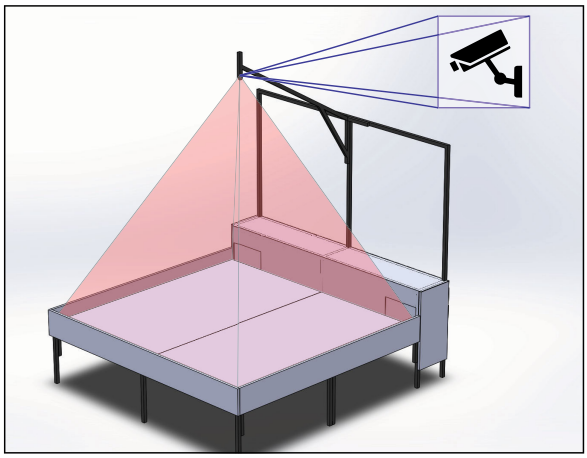

FIGURE 14. The design of the testbed with the overhead monitoring system.

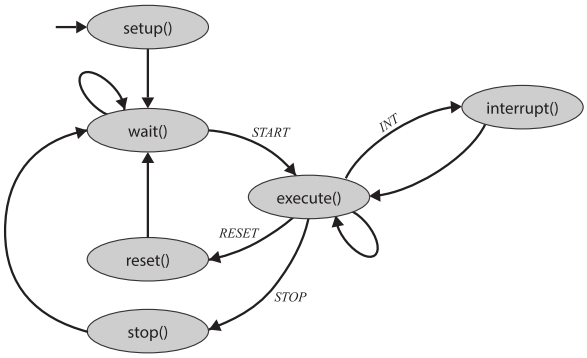

FIGURE 15. The state machine diagram of the robot design. START, RESET, STOP, and INT state transitions were invoked remotely when a state change was requested by the user or the simulator app.

APPENDIX E ROBOT IMPLEMENTATIONS

The physical robots used for the experiments related to this study were developed with an ESP32 SoC as the main control unit. It is an SoC especially designed for IoT applications and comes with built-in Bluetooth and WiFi. The firmware for the robots was written in C++, on top of the hardware abstract libraries provided by the ESP-IDF firmware development kit.

The complete firmware implementation used for this study is available as an open-source repository from the link, https://github.com/Pera-Swarm/firmware and the software documentation is available from the link, https://pera-swarm.ce.pdn.ac.lk/docs/robots/v4/firmware/.

The virtual robot instances were modelled using Java, with the Maven build environment. By using the Maven library we implemented, it is possible to create robot instances by defining the initial coordinates and the heading directions as separate software threads. Each thread operates as an individual robot, following implementations that were similar to physical robots. The implementations can be found from the link, https://github.com/Pera-Swarm/robot-library-java and the software documentation from https://pera-swarm.ce.pdn.ac.lk/docs/robots/virtual/v1/java/.

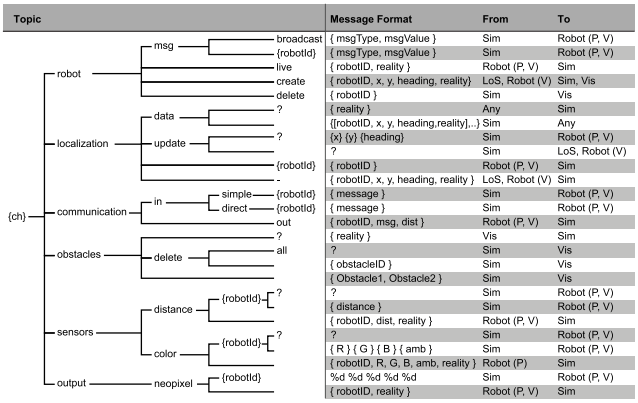

FIGURE 16. The hierarchical design of the topics. {ch}: Channel number, {robotId}: The ID of the robot, Sim: Simulator, Vis: Visualizer, P: Physical, V: Virtual, LoS: Localization System.

Additionally, to execute the experiments on both physical and virtual robots synchronously, the state machine model shown in Figure 15 was used.

The state transition commands such as START, RESET, STOP, and INT will be given through a web-based application, which is capable of coordinating and managing the experiments. The application used for this study is available through the link, https://pera-swarm.ce.pdn.ac.lk/sandbox/ and the source code can be found through this link, https://github.com/Pera-Swarm/sandbox.

APPENDIX F COMMUNICATION PROTOCOLS

The communication between subsystems was implemented using the MQTT protocol. It follows topic-based publication and subscription architecture.

To efficiently manage the topics, a hierarchical naming convention was followed as shown in Figure 16. The topics were generated by going through the tree and combining the texts in each node with slashes in between them. (e.g.: channel1/robot/msg/broadcast, channel1/robot/live).

With this approach, messages can be easily decoded and routed into the relevant module or object without comparing the entire topic string. This is pivotal in physical robots, where the firmware executes on a microcontroller with limited memory and computation power.

REFERENCES

[1] M. Brambilla, E. Ferrante, M. Birattari, and M. Dorigo, "Swarm robotics: A review from the swarm engineering perspective," *Swarm Intell.*, vol. 7, no. 1, pp. 1–41, Jan. 2013, doi: 10.1007/s11721-012-0075-2.

[2] R. Gro, M. Bonani, F. Mondada, and M. Dorigo, "Autonomous self-assembly in swarm-bots," *IEEE Trans. Robot.*, vol. 22, no. 6, pp. 1115–1130, Dec. 2006, doi: 10.1109/tro.2006.882919.

[3] Y. Mohan and S. G. Ponnambalam, "An extensive review of research in swarm robotics," in *Proc. World Congr. on Nature Biol. Inspired Comput. (NaBIC)*, Coimbatore, India, Dec. 2009, pp. 140–145, doi: 10.1109/NABIC.2009.5393617.

[4] A. E. Turgut, H. Çelikkanat, F. Gökçe, and E. Şahin, "Self-organized flocking in mobile robot swarms," *Swarm Intell.*, vol. 2, nos. 2–4, pp. 97–120, Aug. 2008, doi: 10.1007/s11721-008-0016-2.

[5] I. Navarro and F. Matía, "An introduction to swarm robotics," *ISRN Robot.*, vol. 2013, Sep. 2013, Art. no. 608164, doi: 10.5402/2013/608164.
[6] E. Sahin, "Swarm robotics: From sources of inspiration to domains of application," in *Swarm Robotics* (Lecture Notes in Computer Science). Berlin, Germany: Springer, 2005, pp. 10–20, doi: 10.1007/978-3-540-30552-1_2.
[7] K. Szwaykowska, I. B. Schwartz, L. Mier-y-Teran Romero, C. R. Heckman, D. Mox, and M. A. Hsieh, "Collective motion patterns of swarms with delay coupling: Theory and experiment," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 93, no. 3, Mar. 2016, doi: 10.1103/physreve.93.032307.
[8] V. Edwards, P. deZonia, M. A. Hsieh, J. Hindes, I. Triandaf, and I. B. Schwartz, "Delay induced swarm pattern bifurcations in mixed reality experiments," *Chaos*, vol. 30, no. 7, Jul. 2020, Art. no. 073126, doi: 10.1063/1.5142849.
[9] T. H. J. Collett, B. A. MacDonald, and B. Gerkey, "Player 2.0: Toward a practical robot programming framework," in *Proc. Australas. Conf. Robot. Automat. (ACRA)*, 2008, pp. 1–8.
[10] R. Vaughan, "Massively multi-robot simulation in stage," *Swarm Intell.*, vol. 2, nos. 2–4, pp. 189–208, Aug. 2008, doi: 10.1007/s11721-008-0014-4.
[11] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2004, pp. 2149–2154, doi: 10.1109/IROS.2004.1389727.
[12] C. Pinciroli, V. Trianni, R. O'Grady, G. Pini, A. Brutschy, M. Brambilla, N. Mathews, E. Ferrante, G. Di Caro, F. Ducatelle, T. Stirling, Á. Gutiérrez, L. M. Gambardella, and M. Dorigo, "ARGoS: A modular, multi-engine simulator for heterogeneous swarm robotics," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2011, pp. 5027–5034, doi: 10.1109/IROS.2011.6048500.
[13] Z. Pan, A. D. Cheok, H. Yang, J. Zhu, and J. Shi, "Virtual reality and mixed reality for virtual learning environments," *Comput. Graph.*, vol. 30, no. 1, pp. 20–28, Feb. 2006, doi: 10.1016/j.cag.2005.10.004.
[14] P. Milgram and F. Kishino, "A taxonomy of mixed reality visual displays," *IEICE Trans. Inf. Syst.*, vol. E77-D, no. 12, pp. 1321–1329, Dec. 1994. [Online]. Available: https://cs.gmu.edu/~zduric/cs499/Readings/r76JBo-Milgram_IEICE_1994.pdf
[15] W. Honig, C. Milanes, L. Scaria, T. Phan, M. Bolas, and N. Ayanian, "Mixed reality for robotics," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Hamburg, Germany, Sep. 2015, pp. 5382–5387, doi: 10.1109/IROS.2015.7354138.
[16] S. Carpin, M. Lewis, J. Wang, S. Balakirsky, and C. Scrapper, "USARSim: A robot simulator for research and education," in *Proc. IEEE Int. Conf. Robot. Autom.*, Rome, Italy, Apr. 2007, pp. 1400–1405, doi: 10.1109/ROBOT.2007.363180.
[17] O. Michel, "Cyberbotics Ltd. Webot^{TM}: Professional mobile robot simulation," *Int. J. Adv. Robotic Syst.*, vol. 1, no. 1, p. 5, Mar. 2004, doi: 10.5772/5618.
[18] G. Echeverria, N. Lassabe, A. Degroote, and S. Lemaignan, "Modular open robots simulation engine: Morse," in *Proc. IEEE Int. Conf. Robot. Autom.*, Shanghai, China, May 2011, pp. 46–51, doi: 10.1109/ICRA.2011.5980252.
[19] J. Lächele, A. Franchi, H. Bülthoff, and P. R. Giordano, "SwarmSimX: Real-time simulation environment for multi-robot systems," in *Simulation, Modeling, and Programming for Autonomous Robots* (Lecture Notes in Computer Science). Berlin, Germany: Springer, 2012, pp. 375–387, doi: 10.1007/978-3-642-34327-8_34.
[20] J. Jackson, "Microsoft robotics studio: A technical introduction," *IEEE Robot. Autom. Mag.*, vol. 14, no. 4, pp. 82–87, Dec. 2007, doi: 10.1109/m-ra.2007.905745.
[21] G. F. Riley and T. R. Henderson, "The *ns*-3 network simulator," in *Modeling and Tools for Network Simulation*. Berlin, Germany: Springer, 2010, pp. 15–34, doi: 10.1007/978-3-642-12331-3_2.
[22] M. Kudelski, L. M. Gambardella, and G. A. Di Caro, "RoboNetSim: An integrated framework for multi-robot and network simulation," *Robot. Auto. Syst.*, vol. 61, no. 5, pp. 483–496, May 2013, doi: 10.1016/j.robot.2013.01.003.
[23] D. Pickem, P. Glotfelter, L. Wang, M. Mote, A. Ames, E. Feron, and M. Egerstedt, "The Robotarium: A remotely accessible swarm robotics research testbed," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2017, pp. 1699–1706, doi: 10.1109/ICRA.2017.7989200.
[24] A. Reina, M. Salvaro, G. Francesca, L. Garattoni, C. Pinciroli, M. Dorigo, and M. Birattari, "Augmented reality for robots: Virtual sensing technology applied to a swarm of e-pucks," in *Proc. NASA/ESA Conf. Adapt. Hardw. Syst. (AHS)*, Montreal, QC, Canada, Jun. 2015, pp. 1–6, doi: 10.1109/AHS.2015.7231154.
[25] J. P. L. S. de Almeida, R. T. Nakashima, F. Neves-Jr, A. S. de Oliveira, and L. V. R. de Arruda, "Autonomous navigation of multiple robots with sensing and communication constraints based on mixed reality," *J. Control, Autom. Electr. Syst.*, vol. 31, no. 5, pp. 1165–1176, Jul. 2020, doi: 10.1007/s40313-020-00629-1.
[26] D. W. Payton, R. Estkowski, and M. D. Howard, "Pheromone robotics and the logic of virtual pheromones," in *Swarm Robotics* (Lecture Notes in Computer Science). Berlin, Germany: Springer, 2005, pp. 45–57, doi: 10.1007/978-3-540-30552-1_5.
[27] I. Y.-H. Chen, B. MacDonald, and B. Wunsche, "Mixed reality simulation for mobile robots," in *Proc. IEEE Int. Conf. Robot. Autom.*, Kobe, Japan, May 2009, pp. 232–237, doi: 10.1109/ROBOT.2009.5152325.
[28] A. Reina, A. J. Cope, E. Nikolaidis, J. A. R. Marshall, and C. Sabo, "ARK: Augmented reality for Kilobots," *IEEE Robot. Autom. Lett.*, vol. 2, no. 3, pp. 1755–1761, Jul. 2017, doi: 10.1109/lra.2017.2700059.
[29] Z. Makhataeva and H. Varol, "Augmented reality for robotics: A review," *Robotics*, vol. 9, no. 2, p. 21, Apr. 2020, doi: 10.3390/robotics9020021.
[30] S. Zug, C. Steup, A. Dietrich, and K. Brezhnyev, "Design and implementation of a small size robot localization system," in *Proc. IEEE Int. Symp. Robotic Sensors Environ. (ROSE)*, Montreal, QC, Canada, Sep. 2011, pp. 25–30, doi: 10.1109/ROSE.2011.6058536.
[31] H. Wu, S. Qu, D. Xu, and C. Chen, "Precise localization and formation control of swarm robots via wireless sensor networks," *Math. Problems Eng.*, vol. 2014, Nov. 2014, Art. no. 942306, doi: 10.1155/2014/942306.
[32] G. Reina, A. Vargas, K. Nagatani, and K. Yoshida, "Adaptive Kalman filtering for GPS-based mobile robot localization," in *Proc. IEEE Int. Workshop Saf., Secur. Rescue Robot.*, Rome, Italy, Sep. 2007, pp. 1–6, doi: 10.1109/SSRR.2007.4381270.
[33] T. Krajník, M. Nitsche, J. Faigl, P. Vaněk, M. Saska, L. Přeučil, T. Duckett, and M. Mejail, "A practical multirobot localization system," *J. Intell. Robotic Syst.*, vol. 76, nos. 3–4, pp. 539–562, Apr. 2014, doi: 10.1007/s10846-014-0041-x.
[34] G. Antonelli and S. Chiaverini, "Linear estimation of the physical odometric parameters for differential-drive mobile robots," *Auto. Robots*, vol. 23, no. 1, pp. 59–68, May 2007, doi: 10.1007/s10514-007-9030-2.
[35] M. Friedmann, K. Petersen, and O. von Stryk, "Simulation of multi-robot teams with flexible level of detail," in *Simulation, Modeling, and Programming for Autonomous Robots*. Berlin, Germany: Springer, 2008, pp. 29–40, doi: 10.1007/978-3-540-89076-8_7.

DILSHANI KARUNARATHNA (Member, IEEE) received the B.Sc. degree (Hons.) in computer engineering from the University of Peradeniya, Kandy, Sri Lanka, in 2021. From 2021 to 2022, she was a temporary Instructor with the University of Peradeniya, where she engaged in research. She published a paper under the name "Segmentation and Significance of Herniation Measurement Using Lumbar Intervertebral Discs from the Axial View" at the Moratuwa Engineering Research Conference (MERCon) (8th International Multidisciplinary Engineering Research Conference). She received the Best Research Paper Award under big data, machine learning, and cloud computing for this paper. She was also awarded the Best Research Presentation Award at the Engineering Students Conference by Peradeniya (ESCaPe 2021).

NUWAN JALIYAGODA received the B.Sc. degree (Hons.) in computer engineering from the University of Peradeniya, Kandy, Sri Lanka, in 2021. From 2021 to 2022, he worked as a temporary instructor at the University of Peradeniya and engaged in various research activities carried out at the University. Apart from robotics, he is involved in research activities on smart agriculture, and recently published a paper titled the "Internet of Things (IoT) for smart agriculture: assembling and assessment of a low-cost IoT system for polytunnels" (PlosOne, 2023). His research interests include multi-agent systems, robotics, the IoT, and computer aided designing and manufacturing. He was a member of the winning team of the IEEE R10 Region Humanitarian Technology Products Competition (2018) for "Non-intrusive Infant Sleep Apnea Detection System." He was a recipient of the Best Research Presentation Award for the Engineering Students Conference by Peradeniya (ESCaPe 2021).

GANINDU JAYALATH received the B.Sc. degree (Hons.) in computer engineering from the University of Peradeniya, Kandy, Sri Lanka, in 2021. He was the President of the Hackers' Club, University of Peradeniya, from 2020 to 2021. He is currently a Senior Software Engineer with Sysco LABS Technologies Pvt., Ltd. In addition to his involvement in robotics, his research interests include cloud computing, web development, and JAMStack. He was a recipient of the Best Research Presentation Award for the Engineering Students Conference by Peradeniya (ESCaPe 2021).

JANAKA ALAWATUGODA received the B.Sc. degree (Hons.) in computer science from the University of Peradeniya, Sri Lanka, and the Ph.D. degree from the Queensland University of Technology, Australia. He is currently an Associate Researcher–Assistant Professor with the Research and Innovation Centers Division, Rabdan Academy, United Arab Emirates. He is an Adjunct Research Fellow with the Institute for Integrated and Intelligent Systems, Griffith University, Australia; a Visiting Lecturer with the Postgraduate Institute of Science, University of Peradeniya; and a Reviewer of zbMATH Open Database, FIZ Karlsruhe, Germany. He is a fellow of the British Computer Society (BCS), a Professional Member of the Association for Computing Machinery (ACM), and a member of the International Association for Cryptologic Research (IACR). He was awarded the Commendations for Faculty Research Excellence, in 2018 (for the years 2016–2017), 2020 (for the years 2018–2019), and 2022 (for the years 2020–2021) from the Faculty of Engineering, University of Peradeniya.

ROSHAN RAGEL (Senior Member, IEEE) received the Ph.D. degree in computer science and engineering from the University of New South Wales, Sydney, Australia. He is currently a Professor and the Head of the Department of Computer Engineering, University of Peradeniya, Sri Lanka. Moreover, he is the CEO of the Lanka Education and Research Network (LEARN). His current research interests include systems on chip, the Internet of Things, accelerated and high-performance computing, computational biology, and wearable computing. He is a member of the IEEE Computer Society. He was awarded a number of awards for his excellent services and achievements.

ISURU NAWINNE received the B.Sc. degree in engineering from the University of Peradeniya, Sri Lanka, in 2011, and the Ph.D. degree in computer science and engineering from the University of New South Wales, Sydney, Australia, in 2016. He is currently a Senior Lecturer of computer engineering with the University of Peradeniya. He has a keen interest in improving the quality of engineering education and research. He is also involved in curriculum development and devising learner-centred delivery methods, in addition to the research focus on various directions in computer engineering.