Czech Technical University in Prague

Faculty of Electrical Engineering
Department of Cybernetics
Multi-robot Systems

# Development of a Human-Robot Control System Based on Virtual Reality for Swarms of Unmanned Aerial Vehicles (UAVs)

**Bachelor's Thesis**

**Andrej Pajtaš**

Prague, May 2023

Study programme: Open Informatics
Branch of study: Artificial Intelligence and Computer Science

**Supervisor: Daniel Bonilla Licea, Ph.D.**
**Cosupervisor: Akash Chaudhary, M.Sc.**

## Author statement for undergraduate thesis

I declare that the presented work was developed independently and that I have listed all sources of information used within it in accordance with the methodical instructions for observing the ethical principles in the preparation of university theses.

Prague, 26.05.2023

Andrej Pajtaš

## Acknowledgments

I would like to express my gratitude to my supervisors, Daniel Bonilla Licea, Ph.D. and Akash Chaudhary, M.Sc., who introduced me to the field of multi-robot systems and guided me throughout this project. I would also like to thank my friends and family, who supported and encouraged me during my studies.

# BACHELOR‘S THESIS ASSIGNMENT

## I. Personal and study details

| | | | |
|---|---|---|---|
| Student's name: | **Pajtaš Andrej** | Personal ID number: | **499136** |
| Faculty / Institute: | **Faculty of Electrical Engineering** | | |
| Department / Institute: | **Department of Cybernetics** | | |
| Study program: | **Open Informatics** | | |
| Specialisation: | **Artificial Intelligence and Computer Science** | | |

## II. Bachelor’s thesis details

Bachelor’s thesis title in English:

**Development of a Human-Robot Control System Based on Virtual Reality for Swarms of Unmanned Aerial Vehicles (UAVs)**

Bachelor’s thesis title in Czech:

**Vývoj systému k ovládání rojů bezpilotních helikoptér člověkem využívající virtuální realitu**

Guidelines:

The goal of this thesis is to design and implement a system for controlling swarms of Unmanned aerial vehicles (UAVs) using Robot Operating System (ROS), Unity, and a virtual reality headset (Meta Quest 2). The control system will be experimentally tested on a set of hybrid swarms consisting of both real and virtual UAVs. The following tasks will be involved:

1. Review of at least 20 scientific papers related to human-robot interaction for swarms, at least 5 of which must deal with the use of virtual reality. The review must include references [1]-[12].
2. Development of an environment in the Unity game engine capable of two-way communication with UAVs, providing the robot’s state to the environment and, in return, sending control commands to the swarm.
3. Integration of the environment/simulator with a virtual reality headset, to give a more intuitive control to the user of the system similar to the system in [2]. The control will be executed using the headset’s controllers, and different methods like artificial force fields and static meshes will be used to influence the swarm’s behaviour.
4. (Optional) Creation of a method to translate the hand movement of the user to control commands that can be used to alter the goal of the members of the swarm dynamically.
5. Experimental verification of the system and comparison with existing methods for Human-swarm interaction [1] to validate the system’s advantages.
6. (Optional) Utilization of the PrimeX Haptic gloves as an input method to the simulator to allow for a more comprehensive set of possible actions and enable the user to execute multiple commands simultaneously.

Bibliography / sources:


[1] A. Chaudhary, T. Nascimento and M. Saska, "Controlling a Swarm of Unmanned Aerial Vehicles Using Full-Body k-Nearest Neighbor Based Action Classifier," 2022 International Conference on Unmanned Aircraft Systems (ICUAS), Dubrovnik, Croatia, 2022, pp. 544-551, doi: 10.1109/ICUAS54217.2022.9836097.
[2] I. Jang, J. Hu, F. Arvin, J. Carrasco and B. Lennox, "Omnipotent Virtual Giant for Remote Human–Swarm Interaction," 2021 30th IEEE International Conference on Robot & Human Interactive Communication (RO-MAN), Vancouver, BC, Canada, 2021, pp. 488-494, doi: 10.1109/RO-MAN50785.2021.9515542.
[3] D. Biediger, A. Mahadev and A. T. Becker, "Investigating the survivability of drone swarms with flocking and swarming flight patterns using Virtual Reality," 2019 IEEE 15th International Conference on Automation Science and Engineering (CASE), Vancouver, BC, Canada, 2019, pp. 1718-1723, doi: 10.1109/COASE.2019.8843173.
[4] N. Gio, R. Brisco and T. Vuletic, "Control of a drone with body gestures", Proceedings of the Design Society, vol. 1, pp. 761-770, 8 2021.
[5] A. Naghsh, J. Gancet, A. Tanoto and C. Roast, "Analysis and design of human-robot swarm interaction in firefighting", Proc. 17th IEEE Int. Symp. Robot Human Interactive Commun., pp. 255-260, 2008.
[6] A. Kolling, K. Sycara, S. Nunnally and M. Lewis, "Human swarm interaction: An experimental study of two types of interaction with foraging swarms", J. Human-Robot Int., vol. 2, no. 2, pp. 103-128, 2013.
[7] A. Kolling, S. Nunnally and M. Lewis, "Towards human control of robot swarms", Proc. 17th Ann. ACM/IEEE Int. Conf. Human-Robot Interaction, pp. 89-96, 2012.
[8] M. Goodrich, B. Pendleton, P. Sujit and J. Pinto, "Toward human interaction with bio-inspired robot teams", Proc. IEEE Int. Conf. Syst. Man Cybern., pp. 2859-2864, 2011.

[9] S. Nunnally, P. Walker, M. Lewis, N. Chakraborty and K. Sycara, "Using haptic feedback in human robotic swarms interaction", Proc. Human Factors Ergonomics Soc. Ann. Meeting, pp. 1047-1051, 2013.
[10] P. Walker, S. A. Amraii, M. Lewis, N. Chakraborty and K. Sycara, "Human control of leader-based swarms", Proc. IEEE Int. Conf. Syst. Man Cybern., pp. 2712-2717, 2013.
[11] D. Lee, A. Franchi, P. Giordano, H. Son and H. Bulthoff, "Haptic teleoperation of multiple unmanned aerial vehicles over the internet", Proc. IEEE Int. Conf. Robot. Autom., pp. 1341-1347, 2011.
[12] A. Steinfeld, T. Fong, D. Kaber, M. Lewis, J. Scholtz, A. Schultz, et al., "Common metrics for human-robot interaction", Proc. 1st ACM SIGCHI/SIGART Conf. Human-Robot Interaction, pp. 33-40, 2006.
[13] A. Kolling, P. Walker, N. Chakraborty, K. Sycara and M. Lewis, "Human Interaction With Robot Swarms: A Survey," in IEEE Transactions on Human-Machine Systems, vol. 46, no. 1, pp. 9-26, Feb. 2016, doi: 10.1109/THMS.2015.2480801.

Name and workplace of bachelor's thesis supervisor:

**Daniel Bonilla Licea, Ph.D. Multi-robot Systems FEE**

Name and workplace of second bachelor's thesis supervisor or consultant:

**Akash Chaudhary Multi-robot Systems FEE**

Date of bachelor's thesis assignment: **31.01.2023** Deadline for bachelor thesis submission: **26.05.2023**

Assignment valid until: **22.09.2024**

| Daniel Bonilla Licea, Ph.D. | prof. Ing. Tomáš Svoboda, Ph.D. | prof. Mgr. Petr Páta, Ph.D. |
|---|---|---|
| Supervisor's signature | Head of department's signature | Dean's signature |

## III. Assignment receipt

The student acknowledges that the bachelor's thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the bachelor's thesis, the author must state the names of consultants and include a list of references.

.

Date of assignment receipt Student's signature

## Abstract

This thesis presents a system for controlling real, simulated or hybrid swarms of unmanned aerial vehicles using a virtual reality headset. It deals with designing the human-swarm interaction system for virtual reality, implementation of the distributed software, challenges of using artificial force fields and static meshes to influence drones in the real-world environment, experimental verification of the system in a simulation and the real world, and discussing and comparing the results to existing solutions.

**Keywords** Unmanned Aerial Vehicles, Human-Robot Interaction, Robot Swarm, Virtual Reality

## Abstrakt

Tato práce představuje systém pro řízení skutečných, simulovaných nebo hybridních rojů bezpilotních letadel pomocí náhlavní soupravy pro virtuální realitu. Zabývá se návrhem systému interakce člověka a roje pro virtuální realitu, implementací distribuovaného softwaru, problémy s využitím umělých silových polí a statických sítí k ovlivňování bezpilotních letounů v reálném prostředí, experimentálním ověřením systému v simulaci a reálném světě a diskusí a porovnáním výsledků s existujícími řešeními.



**Klíčová slova** Bezpilotní Prostředky, Interakce Člověka s Robotem, Roj Robotů, Virtuální Realita

## Abbreviations

**API** Application Programming Interface

**LiDAR** Light Detection and Ranging

**ROS** Robot Operating System

**UAV** Unmanned Aerial Vehicle

**VR** Virtual Reality

**WLAN** Wireless Local Area Network

# Contents

# Chapter 1

# Introduction

An Unmanned Aerial Vehicle (UAV) is an aircraft, helicopter, drone, or any flying vehicle that humans do not control from within the machine. It comes in a variety of sizes and shapes (e.g., fixed-wing UAVs, quadcopters), from small quadcopters to large aircraft, and can be equipped with various sensors, such as camera, ultrasonic sensor or Light Detection and Ranging (LiDAR). It can be independent and take only the most high-level commands, fully dependent on human control with no ability to identify the steps needed to achieve its goal, or anything in between [1]. UAVs are often used in the military [1], but they are also used for real-time monitoring, search and rescue, delivery of goods, security and surveillance, agriculture [2], firefighting [3], and in the film industry [4]. With advances in technology, UAVs are becoming increasingly integral to various industries.

A UAV swarm is a group of UAVs that work together to achieve a specific goal. Although they try to achieve the same goal, the individual members of the group often have different tasks. An example of a UAV swarm is shown in Figure 1.1. Swarm robotics, as a field, is focused on designing collective behaviours that are robust, scalable, and flexible, to effectively coordinate large numbers of robots [5]. The main characteristics of swarm robotics are the robots' autonomy, ability to influence the environment and lack of central control and global knowledge [5]. They are further characterized by cooperation on a shared task and local sensing and communication [5]. Since the first mentions of swarm robots in the late 1980s, this research field has progressed significantly [6].

Figure 1.1: Flying swarm of three UAVs used in Multi-robot Systems group at CTU in Prague.

Human-swarm interaction is an emerging area of research in the field of swarm robotics [7]. It has the potential to enable new forms of collaboration between humans and intelligent swarm systems. A human operator, the person who controls the robot swarm, can interact with the swarm in different ways depending on the design of the human-swarm interaction system. Common methods of human-swarm interaction include human body control [8], haptic feedback utilisation [9], Virtual Reality (VR) utilisation [10], and remote control [11]. There are many possibilities for adapting human-swarm interaction.

Human-swarm interaction systems using virtual reality often benefit from the user's better overview of the environment. The VR headset can let the user move around the area, zoom in or out, quickly move with robots or create virtual obstacles to influence robots' behaviour [12]. Another approach is to provide users with an intuitive interface and extend their possibilities with virtual tools, which would be challenging to replicate in the real world. An example of that can be the research on the survivability of UAVs, where scientists let users shoot down as many virtual UAVs as possible [10]. Virtual reality offers many applications in human-swarm interaction research.

This thesis aims to develop a system for controlling real, simulated or hybrid swarms of UAVs using a VR headset Meta Quest 2 and its controllers. Existing human-swarm interaction research is required to be reviewed to provide a theoretical foundation. The software must be based on Robot Operating System (ROS) and use Unity Game Engine for the application interface. The application must be capable of two-way communication with the swarm, allowing the user to influence the swarm with artificial force fields and static meshes. Experimental verification and comparison with existing methods for human-swarm interaction are required to validate the system's advantages.

This thesis is organised as follows. Chapter 2 reviews existing research in the field of human-swarm interaction. Chapter 3 introduces the control system developed as a part of this work and discusses its architecture and implementation in detail. Chapter 4 presents the results of experimental verification together with the software and hardware tools used during its execution. The comparison of the control system to existing human-swarm interaction methods is presented in Chapter 5. Finally, Chapter 6 summarises the presented software and its results.

## 1.1 Mathematical notation

The mathematical notation used in the text of this thesis is described in Table 1.1.

| Symbol | Description |
|---|---|
| Lowercase italic letter ($a$, $b$) | scalar |
| Uppercase italic letter ($A$, $B$) | set |
| Bold lowercase letter ($\mathbf{x}$, $\mathbf{y}$) | vector |
| $\mathbf{0}$ | zero vector |
| $\|\cdot\|$ | euclidean distance |

Table 1.1: Mathematical notation.

# Chapter 2

# Human-swarm interaction

Human-swarm interaction studies the involvement of human operators in robot swarm decision-making processes, as well as human interaction with robotic swarms in general. Examples such as firefighters cooperating with autonomous robots [3] or using body gestures for swarm control [8] can be found in human-swarm interaction research. While the field of robot swarms has gained much attention in recent years, both in academia and among the general public, the first use of a robot swarm concept dates back to 1988 [6]. This chapter reviews some of the human-swarm interaction research relevant to the topic of this thesis, in particular, field surveys, body control, haptic feedback, virtual reality, behaviour models and special use cases.

## 2.1 Field surveys

A field survey, or field review, is an overview of books, publications, scientific articles, and other sources relevant to a particular issue, theory or area of research [13]. It includes a description, summary, and critical evaluation of cited works [13]. Surveys are an excellent introduction as they give the reader an overview of the terms used, the progress made, the directions explored and possible future developments in the study area. They capture and summarise what has been achieved in the research field without the need to go through many publications. This section mentions part of the surveys carried out in the human-swarm interaction research.

In 2015, Kolling et al. published a comprehensive review of existing human-swarm interaction research [7]. In addition to comparing more than one hundred publications, they categorised swarm models and swarm tasks and behaviours into various groups. The swarm models were divided into "Bioinspired", "Control Theory", "Amorphous Computing", and "Physics-Inspired", and the swarm tasks and behaviours were divided into "Aggregation and Rendezvous", "Deployment and Area Coverage", "Flocking and Formation Control" and "Foraging and Transport". In one of the sections, the influence of cognitive complexity, communication, state estimation and visualisation, and control methods on human-swarm interaction were discussed in detail. The paper further discusses the importance of solution scalability and focuses on the level of autonomy, leader influence and neglect benevolence of swarms. An example approach to mitigate selected human-swarm interaction scalability issues is the utilisation of more advanced distributed algorithms for swarm control. The authors emphasised the difference between a multi-robot system and a swarm and pointed out the lack of unified terminology in the human-swarm interaction field. Overall, the survey provides a robust introduction to the human-swarm interaction research, describes many of its properties in detail and suggests possible future research trends.

In [14], the authors identified standard metrics used in task-oriented human-swarm interaction. The primary motivation for this publication was to unify the metrics used across the research field, as different research results could not be compared. Other goals were to identify standard metrics that could be used to evaluate a wide range of robot applications and to provide measurement tools for future studies. The authors analysed human-robot interaction from three perspectives: human, robot, and system. Five robot tasks were considered: Navigation, Perception, Management, Manipulation, and Social. The purpose of the navigation task was to move from one position to another. The perception task required understanding the remote environment for use, such as surveillance or target identification. The management task covered the coordination of robots and humans. The manipulation task focused on robot interaction with the environment. The social task required the robot to perform social interaction with people. For each listed task, the authors of [14] identified several metrics, like the obstacle encounter, identification errors or distance judgement. Finally, these metrics were combined into key metrics for human, robot and system performance.

Survey [15] focuses on human interaction with a swarm of robots in an open and cluttered environment and looks at the subject from the perspective of migrants moving to a distant place. It starts with a discussion on how recent population migrations have resulted in accidents and deaths and highlights the lack of research in helping migrants on their journey. The survey presents a literature review on the latest research trends in human-swarm interaction to identify techniques that can be used to find, locate, protect, and generally help migrants in hazardous environments. A taxonomy of swarm interaction is presented, and algorithms related to navigation with and without interaction in different environments are compared. The presented taxonomy divides the swarm interaction model into visual-based and human-centred. Visual-based interaction is categorised into fixed interaction and utilisation of touch screen, smartwatch or augmented reality. Hman-centred includes interaction types with and without gestures. In conclusion, it is pointed out that the choice of interaction medium depends primarily on the intended purpose use of the swarm to facilitate the operator's control of the swarm.

## 2.2 Body control

One of the existing approaches to swarm control is the use of human body gestures and movements. In this thesis, the human body as a means to control a swarm is not directly used, but the selected technique is closely related. The body control methods are often implemented using a camera that captures the human body, identifies the body position or specific movement, and translates it into a predefined command. However, this is not the only approach that leverages body gestures. Another possibility is the haptic control described in more detail in Section 2.3. The following paragraphs mention several human-swarm interaction publications in which people used their bodies to control robot swarms, and cameras were used to capture the bodies.

An example of swarm control using the human body was presented in [16]. In their work, the authors employed the MediaPipe framework [17] to estimate human body pose and implemented a custom k-Nearest Neighbors classifier to translate the pose into an action. For the swarm control, the leader-follower model was used. The proposed method was tested in a Gazebo simulator [18] using a virtual human operator and an actual human operator separately. The method was experimentally verified in different conditions in a real-world environment with an onboard camera attached to a UAV and a stationary camera in front of the human operator. In real-world experiments, the MRS Platform [19] was used to control the UAVs. The presented results showed notable accuracy prediction and feasibility of the method proposed.

In [8], a human-drone interaction system using human body gestures was designed and implemented. The system's interface offers the human operator to fully control the movement and speed of the connected drone with eight different body gestures. The human operator could control the system by raising both arms outwards, crossing arms in front of the body, leaning left or right, rotating the torso to the sides, and bending forward or backwards. Simultaneously, the person could visually observe the real-time state of the drone by following its onboard camera broadcast in the custom graphical interface shown on a computer monitor. Microsoft Kinect [20] motion sensor was used to detect the person's body pose, generating a data structure representing a human skeleton composed of sixteen joints. Using goniometric functions, the authors calculated angles between the different parts of the operator's body and assigned the gesture corresponding command, which was then propagated to the drone. Six participants were selected to experimentally verify the human-drone interaction system's ability of gesture recognition and flight control. Based on the experimental verification, the authors claim that their user interface is intuitive and highly accurate in selected tasks and suitable for various use cases.

Various gesture-based human-swarm interaction approaches are described in [21]. The paper proposes a taxonomy for gesture-based interaction between a swarm of robots and a human. It divides the gestures into three categories: deictic, representational, and manipulation. The deictic interaction includes pointing gestures. The representational interaction consists of gestures representing an object or idea. The manipulation interaction utilises gestures associating shapes. Through the three categories, the authors of the research propose two human-swarm interaction methods: free-form and shape-constrained. The free-form interaction allows unconstrained movement and positioning of robots without following predefined rules or constraints. The shape-constrained interaction allows modification of the swarm's shape by controlling a subset of meaningful degrees of freedom instead of controlling each robot directly. A software capable of recognising the described gestures with a depth sensor was implemented, and experimental verification was conducted to validate the feasibility of the developed system. The results showed that the proposed system is intuitive and easy to learn.

Swarm control using the human body has shown good accuracy and suitability for various tasks in the selected studies. However, it has also shown that it is important to think about the intuitiveness of the control when developing the system. It can be assumed that further research will be devoted to this topic in the future.

## 2.3 Haptic feedback

While implementations of body control methods described in Section 2.2 use a camera to capture the human operator's gestures, implementations that leverage haptic feedback rely on many sensors in direct contact with the human skin and can provide the operator with immediate haptic feedback. This offers broader possibilities of use, especially in combination with other types of user interfaces that can provide, for example, visual or auditory feedback. Selected publications related to haptic feedback are listed below.

In [9], haptic feedback is utilised in a simple human-swarm interaction system. The haptic feedback is utilised by a haptic robot controller, which is connected to the system and the graphical user interface using ROS. The authors decided to take advantage of attractive and repulsive forces algorithms and built the swarm control mechanism on top of them. The attractive and repulsive forces form an artificial force field that influences each robot in the swarm depending on its relative position to the field. Four distinct test environments were built to measure thirty-two participants' performance in various tasks. The first environment resembled a corridor maze where the user was regularly interrupted by unrelated tasks. The second was a setting where the user had to explore as much of the area as possible while the swarm velocity was changing. The third environment did not have any special features, and the users were told to explore it at their own pace. The last environment offered invisible obstacles for the user to go around. After the tests, the authors indicated that the participants performed slightly better with the haptic controller than without it. However, this was only true for selected measurements. This study shows that haptic feedback, in general, is beneficial to human-swarm interaction but might not be beneficial in every scenario.

A framework for haptic teleoperation control for a swarm of UAVs is presented in [22]. The framework leverages the Internet for communication. It is divided into three separate control layers: the UAV control layer, the Virtual Point control layer, and the Teleoperation layer. The UAV control layer represents each UAV by its virtual kinematic point. The Virtual Point control layer modulates the motion of virtual points to follow the teleoperation commands and local artificial potentials. The Teleoperation layer allows the human operator to command selected virtual points' velocity remotely and provides the operator current state of UAVs through haptic feedback at the same time. The control mechanism and the haptic feedback are provided by a haptic device connected to the system. Experimental verification with eight UAVs and all-to-all communication enabled was conducted during the research. In the simulated environment, the human operator flew the UAVs twice from one side to another. In the experimental verification, the system proved stable with acceptable teleoperation and tracking performance.

A haptic display system using a swarm of small wheeled robots to generate tactile feedback is presented in [23]. The robots move across a surface and create a sensation of touch. They are controlled by a central computer that generates complex haptic patterns to create the desired tactile feedback. The hardware and software architecture of the system is described, and several applications are demonstrated, including displaying shapes and moving physical objects. A user study was conducted to evaluate the system's effectiveness in conveying information, showing that the users could successfully recognise various haptic patterns displayed by the system. The paper presents an innovative approach to haptic feedback that could be applied in virtual reality, teleoperation, and robotics.

Haptic feedback is a promising area in the field of human-swarm interaction research. However, at the same time, using haptic feedback might not be beneficial in all applications, as indicated in [9]. This highlights the importance of system design in software development.

## 2.4 Virtual reality

Virtual reality may seem a product of the twenty-first century, but it has been studied since the 1960s, and some references are even older [24]. Despite this research's age, VR has received more attention in the last decade than ever. This section describes a few selected papers related to the use of virtual reality in human-swarm interaction.

The software described in [12] is in many ways similar to the control system proposed in this thesis. The paper proposes a human-swarm interaction framework inspired by childhood memories of playing with ants in a way that the user represents a super-powered giant that can arbitrarily move the UAV from one location to another or add obstacles to the environment. In addition, the user can resize the environment or fly around like Superman. To achieve intuitive control, the authors have implemented the framework with the use of virtual reality. The user interacts with the environment using a VR headset and hand gestures that the headset track. Experimental analysis and a usability study carried out as part of the work showed favourable outcomes for trained human operators.

The survivability of UAV swarms in a potentially hostile environment is investigated in [10]. The authors focused on two nature-inspired flight formations: flocking and swarming. While flocking originated in a flock of birds, insect swarms inspired the swarming pattern. The authors proposed a random motion as a means of increasing survivability. They designed a VR simulation where the user was tasked with shooting down as many UAVs as possible. The implementation incorporates a scoring system to measure and further analyse the data. The work evaluates the effectiveness of the random motion system and proposes enhancements to UAV swarm survivability.

In [25], a multichannel human-swarm interaction system in augmented reality based on Microsoft HoloLens 2 [26] headset is proposed. The authors of the paper took advantage of augmented reality and voice recognition and designed a swarm control system which allows the human operator to command the swarm using hand gestures and voice commands. Voice recognition is achieved by using the speech recognition engine provided by the headset. Language recognition is developed using a text classifier based on the maximum entropy model. Hand gesture recognition is supplied by a body motion controller Leap Motion. The experiments conducted during the research showed that the proposed multichannel human-swarm interaction system with augmented reality is capable of interactions such as robot selection, trajectory motion, and state control.

The authors of [27] introduced a desktop test platform for quantifying human perception, actions, and cognitive load in human-swarm interaction using VR. The test platform was represented by an underwater environment with a fish-inspired robot swarm. The paper starts with describing ideas used within the research, such as cognitive load, visual acuity and model of collective behaviour. It continues defining the graphical model of fish, their analysis-based behavioural, and foraging principle. The paper discusses the ability of the human eye to recognise individual fish in a swarm to quantify swarm perception and estimate acuity accurately. While testing the proposed platform, the human operator wore an eye-tracking headset. The authors compared how many robots displayed in the environment were perceived by the human and how the swarm's behaviour reacted to the human's cognitive load. The publication shows that during human-swarm interaction, the human operator's cognitive load correlates with the swarm's average distance and that the number of recognised robots does not correlate with the number of robots displayed in the environment.

A virtual reality simulation for developing a robotic fish model was created in [28]. The paper's authors created a robotic fish whose purpose was to follow and monitor a school of real fish in the water. The robotic fish was composed of a waterproof core containing electronics and of parts printed on a 3D printer. Because software development and testing on robots in real-world environments are complicated, a virtual reality simulation was created to allow the authors to make significantly faster progress in implementing the control system for the robotic fish. A faithful virtual copy of the robot, including all sensors, was implemented in the simulation. Furthermore, test fish with predefined behaviours were added to the simulation to mimic the natural underwater environment faithfully. A person wearing a VR headset could move through this environment and interact with the whole system, including the environment itself. When the virtual robot was sufficiently well trained in the simulation, experimental tests were conducted in the pool to verify the quality of the control system of the physical robot fish. The experimental tests were successful and confirmed the quality of the implementation. This work has shown that virtual reality can extensively aid in developing and testing software for swarm robots.

The use of virtual reality in developing human-swarm interaction systems has become more common in recent years. The main advantages of VR include augmenting reality with elements that would be very difficult to create in real life, the ability to create advanced simulations that reduce the cost of developing and testing real robots, and a large selection of development tools that speed up the programming of human-swarm interaction systems. One significant disadvantage is that some VR headsets do not allow the user to use them outdoors, as some sensors could be damaged by exposure to sunlight. Another disadvantage is motion sickness and dry eyes, which can plague the user if the headset is worn for long periods. Overall, virtual reality integration with systems for human-swarm interaction has the potential to be increasingly used.

## 2.5 Behaviour models

An essential part of human-swarm interaction is the swarm behaviour model. This behaviour is governed by algorithms that are often inspired by the behaviour and interaction of groups of animals in nature, birds, insects, or fish. This section describes several papers dealing with models of robot swarm behaviour.

An experimental research study about foraging robot swarms [29] focuses on two types of human-swarm interaction: intermittent and environmental. The authors built a test platform on which they studied the ability of human operators to solve the information foraging problem [30] in a robot swarm. Intermittent interaction allowed human operators to select a subgroup of robots from the swarm and instruct them to take the chosen action, and the environmental interaction allowed operators to manipulate the surrounding environment rather than the robots themselves. Human-swarm interactions were implemented using a selection control method based on intermittent interaction and a beacon control method based on environmental interaction. The study results showed that the beacon control method does not scale as well as the selection control method, which generally had better results. The authors pointed out the importance of understanding and selecting the correct type of human-swarm interaction in swarm systems.

Paper [11] from 2012 can be referred to as a predecessor of article [29] published by the same authors a year later. A significant part of the research was already published in the previous work. While they share the same research direction and many of the findings, the paper [11] is not so robust and elaborated. Overall, there are only slight differences between these two works. The most notable difference is the extent of the content.

Bio-inspired robot swarms and human interaction with them are discussed in [31]. First, the paper formalises the principles of human interaction with bio-inspired swarms of robots. Then, it presents metrics for responsiveness and cohesiveness of human input and presents nearest-neighbour and metric-based topologies, which were used in the final experiments. Further, the paper empirically studies the responsiveness and cohesiveness of predator-based and leader-based human-swarm interaction in both presented topologies. Two types of experiments were carried out to show and describe differences between leader and predator models and between nearest-neighbour and metric-based topologies. The first experiment type used dynamics of artificial physics, and the second experiment type used dynamics of fish behaviour. The cohesiveness was measured by an agent's influence on other agents, and the responsiveness was measured by the operator's influence on agents. The conducted experiments demonstrated the relevance of the used metrics.

Human interaction with leader-based swarms was studied in [32]. The main objective of this study was to examine two straightforward techniques for information propagation, namely flooding and consensus. The flooding method instructs each UAV to move at the same speed and in the same direction as the swarm leader. The consensus method instructs each UAV to move at the average speed and direction of the surrounding UAVs. The paper compares the manoeuvrability of the swarm using the two methods with and without sensing error. Testing the approaches revealed that the consensus method falls far short of the flooding method. However, the consensus was advantageous in isolated scenarios, especially at lower speeds, during the testing.

The papers mentioned above show the importance of choosing a suitable behavioural model. The chosen model significantly affected the results in most of the tasks studied.

## 2.6 Special use cases

The human-swarm interaction research is extensive and includes many sub-disciplines. This section contains a few selected publications that do not entirely fall into any previous categories but are nevertheless important in the context of human-swarm interaction research.

The feasibility of using human-swarm interaction in firefighting is analysed in [3]. The paper was published as a part of the European project GUARDIANS (Group of Unmanned Assistant Robots Deployed In Aggregative Navigation supported by Scent detection) [33], which developed and applied autonomous robots' assistance in rescue and firefighting operations. The analysis describes the key characteristics of fire firefighting and the different roles of people participating in the operation. It also defines the main challenges of human-swarm interaction. The authors proposed various types of human-swarm interaction interfaces, which they divided into "Direct Human-Swarm Interaction", "Direct Swarm-Human Interactions", and "Remote Interaction Via Base station". Implementation of the user interface and experimental verification were not part of the work, but the authors covered that in their following publications [34] and [35].

Study [36] focuses on streamlining human-swarm interaction in searching for and localising the radiation source. It identifies ten desirable characteristics a human-swarm interaction system should meet and proposes a general human-swarm interaction architecture that satisfies all of these properties. A simulation environment modelling a swarm of robots was created, and a human-swarm interaction interface based on the proposed architecture was developed. This system was tested on a radiation source search and localisation task, comparing the performance of an autonomous swarm and a swarm controlled by humans. The performance comparison showed that human-controlled swarms were more effective in achieving their goals and that the proposed user interface was easy to learn.

Despite efforts to categorise an area of research into specific groups, there will always be studies that could be classified into multiple groups at once or belong to neither. The papers described in this section highlight the diversity of robotic swarms' applications and their interactions with humans.

## 2.7 Summary

This section summarises all reviewed scientific papers for a better overview. A list of these papers with a short description is provided in Table 2.1.

| Ref. | Title | Description |
|---|---|---|
| [16] | "Controlling a Swarm of Unmanned Aerial Vehicles Using Full-Body k-Nearest Neighbor Based Action Classifier" | Robot swarm control software using human body gestures. |
| [12] | "Omnipotent virtual giant for remote human-swarm interaction" | Robot swarm control software utilising virtual reality. |
| [10] | "Investigating the survivability of drone swarms with flocking and swarming flight patterns using virtual reality" | Robot swarm survivability in a hostile environment using flocking and swarming flight formation. |
| [8] | "Control of a drone with body gestures" | Robot swarm control software using human body gestures. |
| [3] | "Analysis and design of human-robot swarm interaction in firefighting" | A utilisation of robot swarms in firefighting. |
| [29] | "Human swarm interaction: An experimental study of two types of interaction with foraging swarms" | Intermittent and environmental interaction in foraging robot swarms. |
| [11] | "Towards human control of robot swarms" | Intermittent and environmental interaction in foraging robot swarms. |
| [31] | "Toward human interaction with bio-inspired robot teams" | Human interaction with bio-inspired robot swarms. |
| [9] | "Using haptic feedback in human robotic swarms interaction" | A utilisation of haptic controller in robot swarm control. |
| [32] | "Human control of leader-based swarms" | Human interaction with leader-based robot swarms. |
| [22] | "Haptic teleoperation of multiple unmanned aerial vehicles over the internet" | Haptic teleoperation framework for robot swarm control. |
| [14] | "Common metrics for human-robot interaction" | Identification and unification of standard metrics across the research field. |
| [7] | "Human interaction with robot swarms: A survey" | A comprehensive review of existing human-swarm interaction research. |
| [25] | "A multichannel human-swarm robot interaction system in augmented reality" | Robot swarm control software utilising augmented reality. |
| [27] | "Designing a Virtual Reality Testbed for Direct Human-Swarm Interaction in Aquatic Species Monitoring" | Underwater test environment for quantifying human perception, actions, and cognitive load in human-swarm interaction using VR. |
| [36] | "Human swarm interaction for radiation source search and localization" | Human-swarm in radiation source search task. |
| [28] | "On the development of robot fish swarms in virtual reality with digital twins" | Underwater virtual reality simulation for robot swarms. |
| [15] | "Swarm robotic interactions in an open and cluttered environment: a survey" | Overview of human-swarm interaction in an open and cluttered environment. |
| [23] | "Swarmhaptics: Haptic display with swarm robots" | Haptic-based small wheeled robot framework. |
| [21] | "Gesture based human-multi-robot swarm interaction and its application to an interactive display" | A utilisation of deictic, representational, and manipulation gestures in robot swarm control. |

Table 2.1: List of reviewed scientific papers related to human-swarm interaction.

## 2.8 Conclusion

For the period of its existence, human-swarm interaction research has demonstrated the benefits of combining the robustness and adaptability of robot swarms with human decision-making and creativity. The research has covered many areas, including utilising virtual reality, body control methods, and haptic feedback. However, the research has also highlighted several challenges, such as choosing the suitable swarm behaviour model or designing the human-swarm interaction interface concerning subjective human needs. Despite these challenges, the findings of human-swarm interaction research suggest that this field has significant potential to keep growing.

# Chapter 3

# Control system

The control system is built on the Robot Operating System[1], an open-source set of software tools and libraries for robot software development. In addition to sensor drivers and implementation of commonly used algorithms, ROS offers messaging framework for communication between processes. Each process is called a Node and offers communication with other Nodes using Topics and Services. Topics take advantage of asynchronous communication based on a publisher-subscriber pattern. Services provide synchronous calls with a response. Both Topics and Services must be named and have a message type. This work mainly uses ROS as a communication platform between different parts of the software.

The software architecture of the control system is shown in Figure 3.1. As can be seen in the diagram, the system is divided into three main parts: VR headset, computer and drones (UAVs). Each part is physically separated and communicates with others over Wireless Local Area Network (WLAN). This design assumes that the WLAN is configured and available to all devices. It is not discussed any further.

Figure 3.1 shows what components the VR headset, computer and drones consist of. On the computer and drones, ROS instances are running. On each ROS instance, the MRS Platform is running. MRS Platform consists of many ROS packages, from which only the most relevant to this work are shown in the diagram. Both ROS instances and MRS Platform instances communicate across the devices. The computer has three extra ROS packages: Controller package, Octomap Merger package and Rosbridge suite. In addition, there is a Gazebo robotic simulator for simulating virtual UAVs. On the VR headset, there is only the user interaction application running.

Figure 3.1 also shows communication between different processes. All ROS instances and MRS Platform instances share data across the devices. VR headset communicates with Rosbridge Server running on the computer using JSON messages. The Rosbridge Server parses messages from the application and sends them further to the Octomap Merger and Controller packages. The Rosbridge Server also receives messages from instances of the MRS Platform and sends them to the application. Octomap Merger package subscribes to data from Rosbridge Server and MRS Octomap Server packages and publishes data to MRS Octomap Planner packages. Controller package publishes data to MRS Platforms. Communication within the instances of the MRS Platform and ROS is not shown.

This chapter is divided into three sections. Section 3.1 deals with the VR headset and the control application installed on the headset. It discusses the user interface and how it is implemented. Section 3.2 describes the software running on the computer. It includes all packages implemented for the control system. Section 3.3 presents drones and their onboard systems.

[1] ROS website: `https://www.ros.org/`

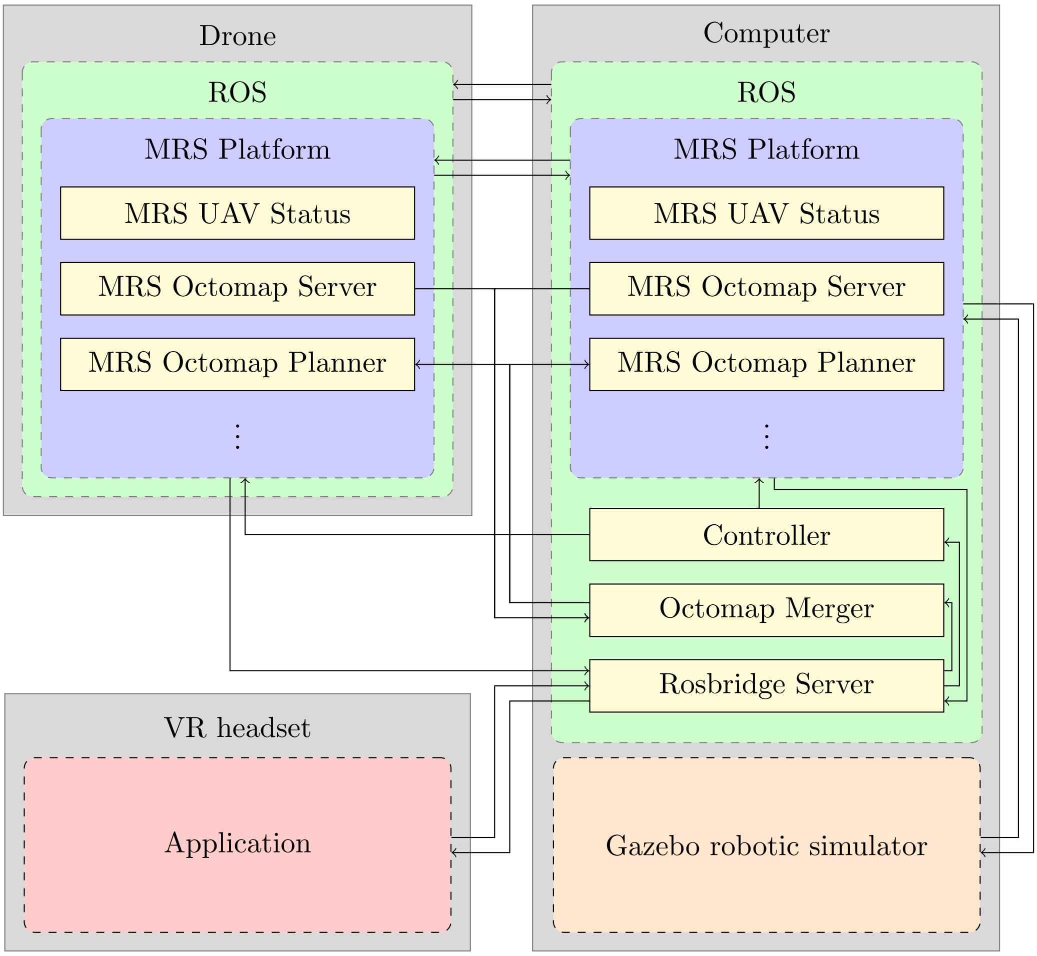



Figure 3.1: High-level architecture diagram of the control system.

## 3.1 VR headset

Designing a usable user interface during application development is extremely important. Even the best application with a poorly designed interface is difficult for the user to use. On the other hand, even if there are flaws in the software, a simple and intuitive user interface can make the application more pleasant for the user and easier to use.

The only interface a user of the control system is exposed to is Meta Quest 2 headset and its controllers. Meta Quest 2 is a virtual reality headset released initially as Oculus Quest 2 in October 2020. It was created by Reality Labs, a subsidiary company of Meta Platforms[2]. The product consists of a headset and two wireless controllers. Although the headset can be controlled with hand gestures, the controllers offer the user more interaction options. Meta Quest 2 and its two controllers are shown in Figure 3.2 and Figure 3.3, respectively.

This section describes the human-swarm interaction interface of the presented application and its implementation. Subsection 3.1.1 describes the features of the VR headset and how it can be operated. Subsection 3.1.2 describes the application environment, its implementation, and how the user can interact with it.

[2]Meta Platforms website: `https://www.meta.com/`

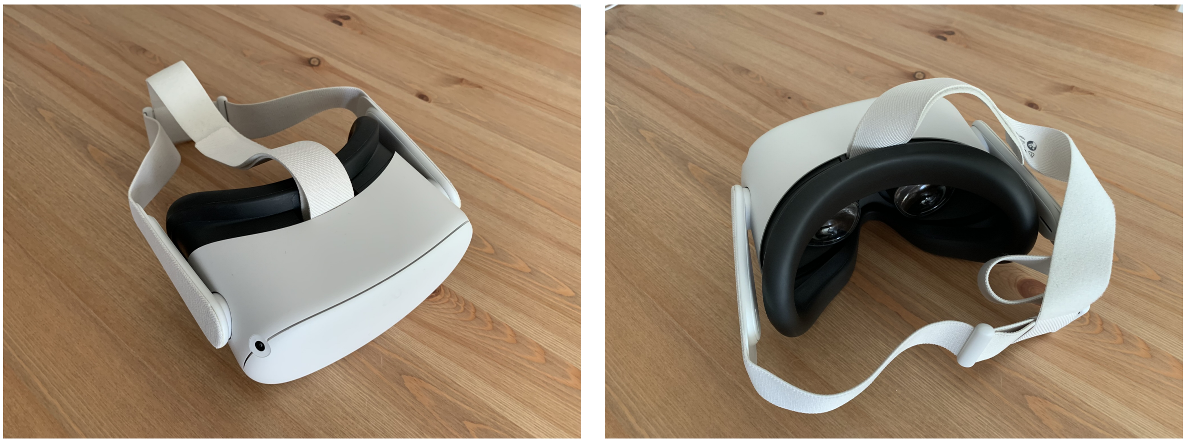


Figure 3.2: Meta Quest 2 headset.

### 3.1.1 Meta Quest 2 headset

Meta Quest 2 headset has two fast-switch LCDs, one for each eye, two built-in speakers and four infrared cameras, and weighs 503 grams. Each screen has a resolution of $1832 \times 1920$ px and a refresh rate of 90 Hz. The infrared cameras track the position and rotation of controllers and capture the surrounding environment allowing users to see what is happening around them, which is essential for augmented reality. The headset has 6 GB of RAM and 256 GB of internal memory and runs the Quest operating system, which is based on Android. It has its battery and charges via a USB-C connector that can also be used to transfer data between the headset and other devices.

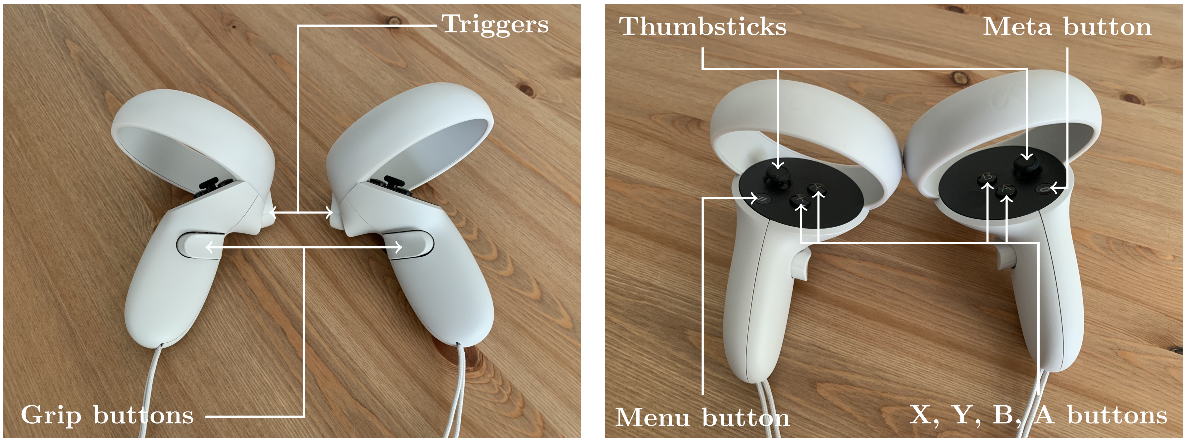



Figure 3.3: Meta Quest 2 controllers with annotated buttons.

The two controllers provided with the headset measure $9 \times 12$ cm, weigh 126 g each, can be connected to the headset wirelessly and require AA batteries as they cannot be recharged. Each controller has two classic buttons (A and B, or X and Y), one reserved button (Menu button or Meta button), a thumb rest button and a thumbstick. In addition, there is one trigger button in the front and a grip button on the side of each controller. A picture of annotated controllers is shown in Figure 3.3. The controllers also share their position and rotation with the headset.

To use the presented application, the user must use the headset together with its controllers. The application itself runs directly on the headset and communicates with ROS over WLAN. As far as the headset is concerned, there is no need to set up anything else.

### 3.1.2 Application

The application introduced as part of this project was implemented in Unity. Unity is a cross-platform development tool from Unity Technologies and offers an intuitive development environment supported by a strong creator community. It provides a flexible and customisable game engine, allowing developers to create unique gameplay mechanics and visual effects in both 2D and 3D. Another advantage is the presence of Unity Asset Store, which offers many ready-to-go game assets for download or purchase. Overall, Unity offers many benefits for game development and is widely used by developers of all levels of experience.

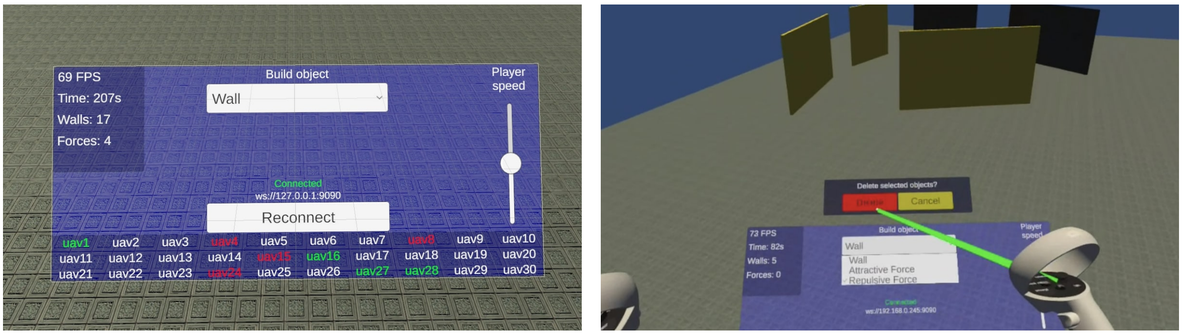
The application allows the user to control and influence real and simulated UAVs using artificial force fields and static meshes in a way that also considers the real-world environment and its limitations. In other words, when the user commands the swarm to fly to a new position, the swarm will avoid not only real-world obstacles but also virtual obstacles that exist only within the application. The following paragraphs discuss the various features of the virtual environment and how the user can leverage it in human-swarm interaction.

#### Navigation

The control panel provides the user with a simple graphical interface for intuitive application control. It rotates with the user so that whichever way the user is looking, the control panel is always available at the bottom of their waist. The control panel is shown in Figure 3.4a. The top left corner of the panel displays general information: the current FPS count, the time since the application was launched, the number of walls placed, and the number of artificial force fields in the environment. In the top middle of the panel is a dropdown list for selecting the object the user wants to place in the environment. The choices are *Wall*, *Attractive Force* and *Repulsive Force*. The placement itself can be done using the controller. On the right side of the panel is a slider for setting the speed of user movement in the virtual environment. Down the middle, the connection's current state to the rest of the system is displayed, as well as a *Reconnect* button to re-establish the connection in case of disconnection. Below is a table showing the status of a total of thirty UAVs. The green fields represent the UAVs that are connected, and the red fields represent the UAVs that have lost connection or have been disconnected. UAVs that were not previously connected have white fields.

The user can interact with the control panel by using the right controller, pointing the controller at the selected element and pressing the A button. How the interaction with the control panel looks from the user's perspective can be seen in Figure 3.4b.

The thumbstick on the left controller allows horizontal movement in the environment, and the thumbstick on the right controller rotates the camera view to the left or right. Camera view rotation is enabled so the user does not have to rotate the body to change the view. If the user wants to change their height, they can use the X button to go up or the Y button to go down.




(a) Control panel in the application.

(b) User interacting with the control panel.

Figure 3.4: Control panel in the application (a) and the user interacting with it (b).

### Swarm control

The application represents each UAV in the swarm with an animated drone model. The visualisations of the models in the application are shown in Figure 3.5. When the application is connected to ROS, it automatically checks which UAVs are available. When a new UAV is connected, its model is displayed to the user in the application within moments. When a UAV is disconnected, the model is destroyed after a failed attempt to maintain the connection. Thanks to this mechanism, the user needs not to be involved in maintaining the connection with existing UAVs and does not have to worry about it.

Each virtual UAV represents one real or simulated UAV. The virtual UAV continuously updates its position in the virtual environment according to its position in the real world. Simultaneously, the virtual UAV sends back information about the surrounding environment, virtual obstacles and artificial force fields existing within the application.

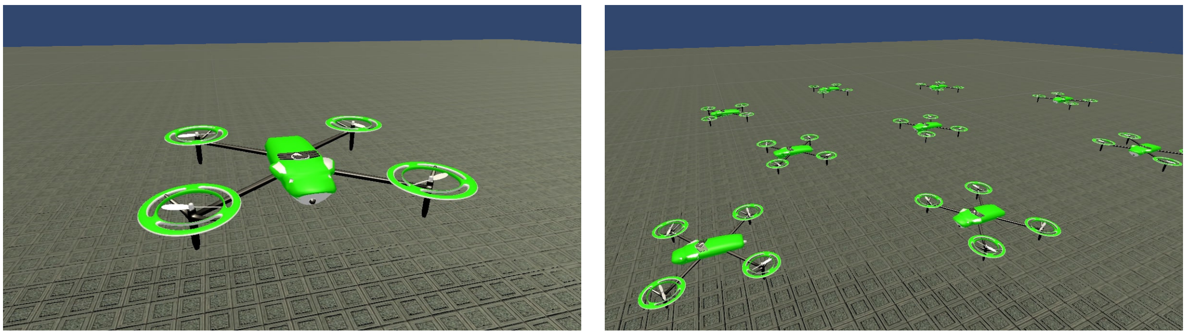
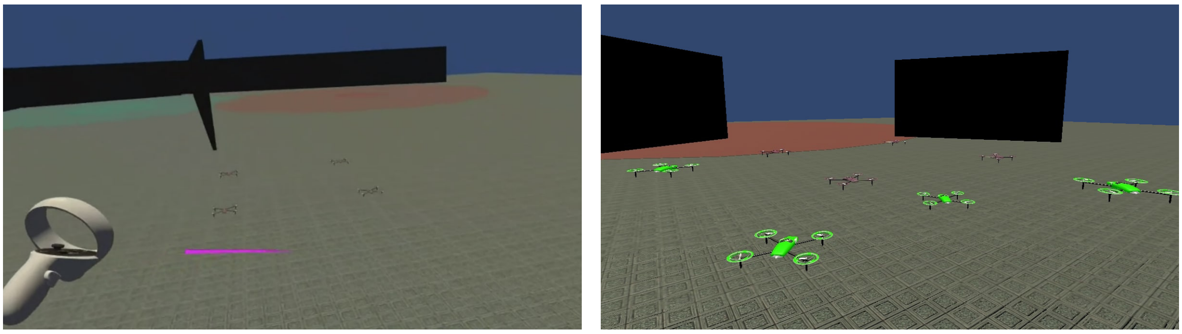


Figure 3.5: Graphical representation of UAVs in the application. The model comes from the Unity Asset Store, where it is freely available.

In order to command the swarm, the user has to press the trigger button on the left controller. While the trigger button is pressed, the left controller produces a laser pointing in the same direction. When the laser is pointed at the ground, a transparent marker representing the future position of the UAV swarm is shown above the point where the laser hits it. This process is shown in Figure 3.6a. Once the user releases the trigger button on the left controller, the laser disappears, and the swarm is commanded to move to the selected position. While UAVs move to the new position, they avoid both virtual and real-world obstacles. A swarm of UAVs moving to their goal is shown in Figure 3.6b.

As multiple control mechanisms are implemented, their priorities must be defined to resolve conflicting orders. The priorities of control elements are shown in Table 3.1. UAV avoidance has the highest priority of all control elements. In case two UAVs are about to collide, the system takes over the control and prevents the collision. Obstacle avoidance has the second highest priority. It works similarly to the UAV avoidance system and prevents collisions with physical obstacles to protect the drone against damage. When there is no imminent risk of collision with any real-world object, the path planning control mechanism determines the swarm's movement. If the swarm is commanded to move to a new position, the UAVs ignore all artificial force fields until they reach their goal. Artificial forces are taken into account only when UAVs are idle.



(a) User selecting a new position for the swarm.

(b) Swarm of UAVs moving to their goal.

Figure 3.6: User selecting a new position for the swarm (a) and a swarm of UAVs moving to their goal (b).

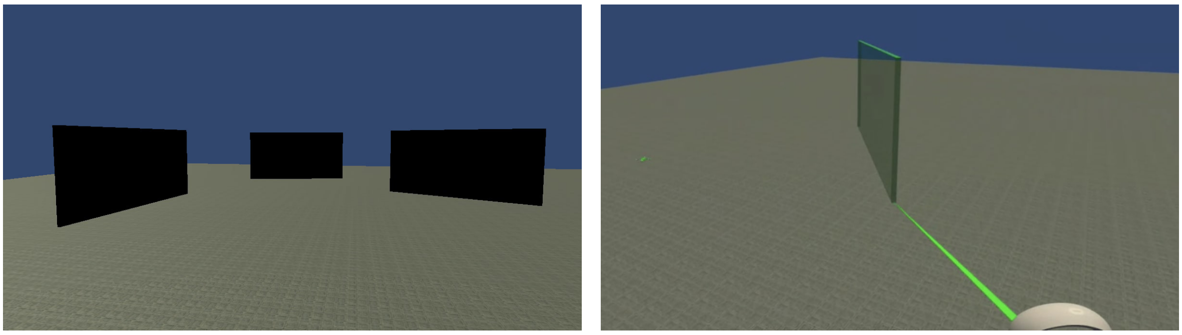
| Control element | Abstraction level | Priority |
|---|---|---|
| Collision avoidance with other UAVs | Real-world | 1 |
| Obstacle avoidance | Real-world | 2 |
| Path planning | Virtual environment | 3 |
| Artificial forces | Virtual environment | 4 |

Table 3.1: Hierarchy of individual swarm control elements.

### Static meshes

Another option for the user to interact with the swarm is to place static meshes in the environment. Static meshes are non-animated stationary 3D polygons used in many game engines or simulations. In the application presented in this work, they can be placed or removed from the virtual environment during runtime. Once the object is placed, all UAVs notice it through their LiDARs and perceive it as a real obstacle. For simplicity, only one type of static mesh is implemented in the application: a wall. Walls in the virtual environment of the application are shown in Figure 3.7a.

Before placing a new wall, the user has to select the *Wall* option on the control panel. Once the object is selected, it can be placed by pressing the trigger button on the right controller and pointing it at an arbitrary place on the ground. While the trigger button is pressed, the user can adjust the wall position, as shown in Figure 3.7b, or abort the build process. When satisfied, the user releases the trigger button and the wall is built.



(a) Walls in the virtual environment.

(b) User placing a new wall.

Figure 3.7: Walls in the virtual environment of the application (a) and user placing a new wall (b).

### LiDAR

In the virtual environment, each UAV has a virtual LiDAR implemented. The virtual LiDAR work similarly to a physical one. It casts 1500 lasers evenly distributed horizontally at 360 degrees and vertically at 90 degrees, and whenever the laser hits an object, the hit point is considered an obstacle. All hit points together form a point cloud. A point cloud is a set of point coordinates in 3D space. It can form objects of any shape and is often used for computer animations or modelling. In ROS, it is represented by the PointCloud message, the structure of which is shown in Definition 3.1. Each UAV publishes its LiDAR point cloud outside the application via ROS Topic. More details on how the point clouds are used to build a surrounding environment map are in Section 3.2.

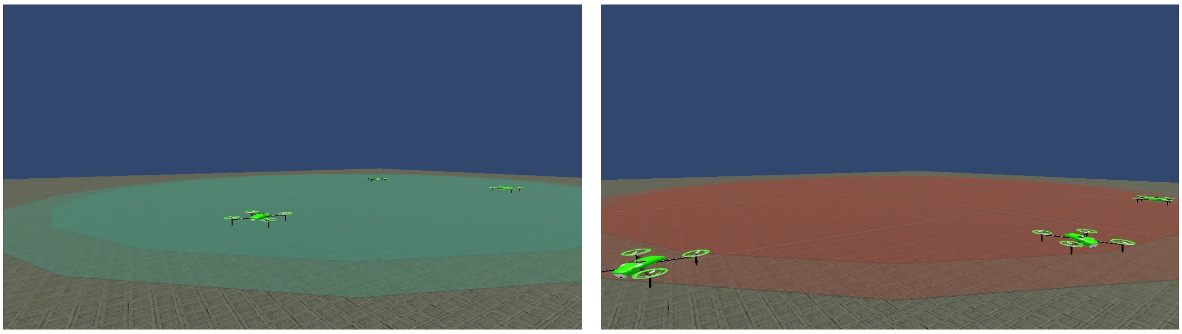
```
# This message holds a collection of 3d points, plus optional additional
# information about each point.

# Time of sensor data acquisition, coordinate frame ID.
Header header

# Array of 3d points. Each Point32 should be interpreted as a 3d point
# in the frame given in the header.
geometry_msgs/Point32[] points

# Each channel should have the same number of elements as points array,
# and the data in each channel should correspond 1:1 with each point.
# Channel names in common practice are listed in ChannelFloat32.msg.
ChannelFloat32[] channels
```

Definition 3.1: sensor_msgs/PointCloud.msg [37].

Publishing point clouds from the virtual LiDAR was chosen to achieve similarity to the real swarm operations. An alternative would have been to publish complete information about the virtual environment, including the position and size of all objects. However, this would have led to slightly different behaviour where the UAV would have known about all virtual obstacles much earlier than the real ones.

### Artificial force fields

Artificial force fields are virtual circular forces influencing surrounding UAVs. The closer the UAV is to the centre of the artificial force field, the stronger the influence is. The application implements two types of force fields: an attractive force field and a repulsive force field. While the attractive force field draws the UAVs inside the field centre, the repulsive force field draws the UAVs outside the field centre. An attractive artificial force field and a repulsive artificial force field in the virtual environment are shown in Figure 3.8a and Figure 3.8b, respectively.



(a) Attractive force field attracting surrounding UAVs. (b) Repulsive force field repelling surrounding UAVs.

Figure 3.8: Attractive force field (a) and repulsive force field (b) influencing surrounding UAVs.

Artificial force fields can be built similarly to walls. Before placing it, the user has to select either the *Attractive Force* option or the *Repulsive Force* option on the control panel. Once the proper option is selected, it can be placed by pressing the trigger button on the right controller and pointing the controller at an arbitrary place on the ground. While the trigger button is pressed, the user can adjust the position of the force field or abort the build process. When the trigger button is released, the force field is successfully generated.

Effects of the artificial force fields on the surroundings are expressed by mathematical functions. Function $f(d)$ determines the speed effect of the attractive force field for all points at a distance of $d$ metres from the force field centre. Function $g(d)$ determines the speed effect of the repulsive force field for all points at a distance of $d$ metres from the force field centre. Equation (3.1) and Equation (3.2) show both functions, respectively.

$$f(d) = \begin{cases} e^{-\frac{d}{32}} \cdot (\ln 25d - \ln 100) & \text{if } d > 0 \land d \leq 32 \\ 0 & \text{if } d > 32 \end{cases} \tag{3.1}$$

$$g(d) = \begin{cases} -2e^{-\frac{d}{16}} & \text{if } d > 0 \land d \leq 32 \\ 0 & \text{if } d > 32 \end{cases} \tag{3.2}$$

Graphs of $f$ and $g$ are shown in Figure 3.9. The graphs show how the forces change depending on the distance from the centre of the artificial force field. For the attractive force, it can be seen that its attractive speed effect (red curve) is roughly the same from 32 to 10 metres but decreases from 10 metres closer to the centre of the force field. At a distance of 4 metres, the attractive speed effect is zero, and for objects even closer to the centre, the attractive speed effect is negative. This means that attractive force fields repulse objects located within this distance. In the graph, this point is visualised by a red dotted line. The reason for repulsion in the attractive force field centre is to prevent the UAVs from colliding. On the other hand, the repulsive force field speed effect (blue curve) is as one would expect. The further from the force field centre, the smaller the speed applied to objects within the range is. Artificial force fields do not influence objects beyond 32 meters.

The vector determining the direction and speed effect of the immediate force applied at a given point to a given force field can then be calculated by multiplying the normalized vector pointing to the force field by the speed effect of the force at that distance. To obtain the vector pointing to the force field, one subtracts the vector representing the position of the force field from the vector representing the position of the point. Function $\alpha(\mathbf{a}, \mathbf{u})$, expressing the immediate force vector applied at a given point $\mathbf{u}$ towards an attractive artificial force field centred in position $\mathbf{a}$, is shown in Equation (3.3). Function $\beta(\mathbf{r}, \mathbf{u})$, expressing the immediate force vector applied at a given point $\mathbf{u}$ towards an attractive artificial force field centred in position $\mathbf{r}$, is shown in Equation (3.4).

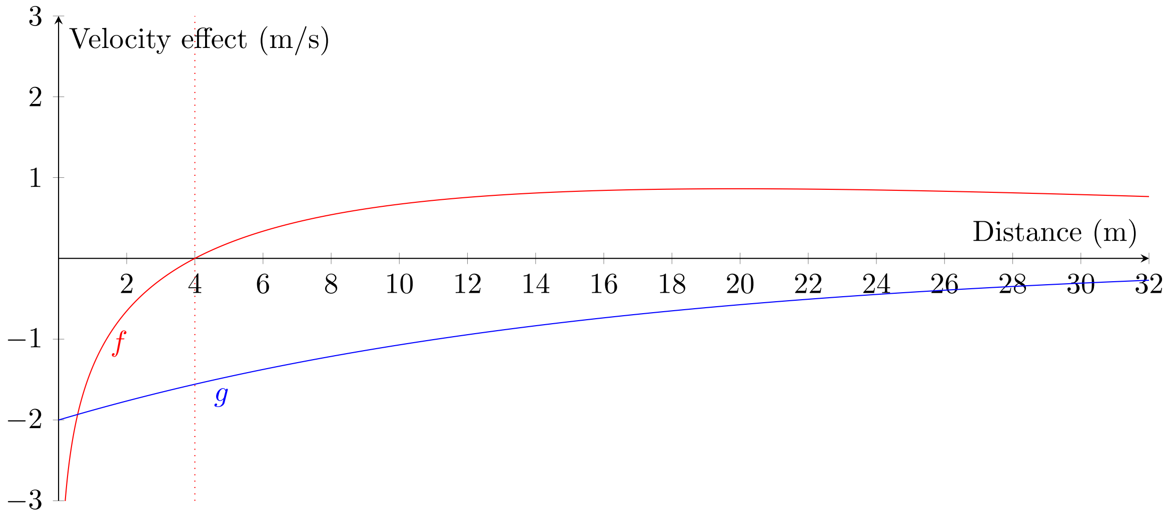



Figure 3.9: Speed effect of the attractive force field (red) and repulsive force field (blue) as functions of distance.

It is important to note that all position coordinates used for force calculations are in $\mathbb{R}^2$. The height of points is not considered as artificial force fields implemented in the proposed application do not affect it.

$$\alpha(\mathbf{a}, \mathbf{u}) = \begin{cases} \frac{\mathbf{a}-\mathbf{u}}{\|\mathbf{a}-\mathbf{u}\|} \cdot f(\|\mathbf{a}-\mathbf{u}\|) & \text{if } \mathbf{a} \neq \mathbf{u} \\ \mathbf{0} & \text{otherwise} \end{cases} \tag{3.3}$$

$$\beta(\mathbf{r}, \mathbf{u}) = \begin{cases} \frac{\mathbf{r}-\mathbf{u}}{\|\mathbf{r}-\mathbf{u}\|} \cdot g(\|\mathbf{r}-\mathbf{u}\|) & \text{if } \mathbf{r} \neq \mathbf{u} \\ \mathbf{0} & \text{otherwise} \end{cases} \tag{3.4}$$

The overall immediate force and direction in a given point determined by all existing artificial force fields can be calculated simply by summing the individual forces belonging to the force fields. Let $A = \{\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n\}$ be a set of $n$ attractive force fields' position coordinates and $R = \{\mathbf{r}_1, \mathbf{r}_2, \dots, \mathbf{r}_m\}$ be a set of $m$ repulsive force fields' position coordinates. Then, the overall force in point $\mathbf{u}$ corresponds to $\gamma(A, R, \mathbf{u})$ formulated in Equation (3.5).

$$\gamma(A, R, \mathbf{u}) = \sum_{\mathbf{a} \in A} \alpha(\mathbf{a}, \mathbf{u}) + \sum_{\mathbf{r} \in R} \beta(\mathbf{r}, \mathbf{u}) \tag{3.5}$$

All UAVs deployed in the virtual environment are affected by the overall force given by function $\gamma$.

## 3.2 Computer

The computer is an integral part of the proposed control system's architecture. In particular, it serves as middleware between applications running on the VR headset and virtual or real UAVs. It can be divided into two essential parts: ROS and Gazebo. ROS runs most of the packages introduced in this work, including virtual UAVs. Gazebo is a robotic simulator used to simulate a real-world environment. More details on Gazebo are given in Section 4.1.

The packages running on ROS include Rosbridge suite, Controller package and Octomap Merger package. They mainly provide essential functionalities such as communication between the virtual environment and the UAVs or creating a joint map composed of real and virtual obstacles.

Another software component running on ROS is the MRS-UAV system from the Multi-robot Systems (MRS) research group at CTU in Prague [19]. It is an open-source platform allowing the deployment of multiple multi-rotor helicopters outdoors and indoors, taking care of all the essential functions from takeoff and trajectory planning to the collision avoidance system. The MRS Platform is ready for advanced simulations allowing proper testing of the software before deploying it on real hardware. This work uses the platform to verify the proposed human-swarm interaction control system's function. More details on the MRS Platform and the software running on it are given in Section 3.3.

This section describes individual programs running on the computer, except for the MRS Platform, which is described in detail later in the thesis.

### 3.2.1 Rosbridge suite

Rosbridge suite is a set of packages providing an interface between ROS and web interface. It includes a Rosbridge Server, which, with its JSON Application Programming Interface (API), allows any program that is able to communicate using HTTP to connect to ROS. Therefore, instead of integrating ROS into an application, starting the Rosbridge server and communicating with it using its API is sufficient. This possibility makes it significantly easier for third-party applications to use ROS features. Rosbridge's design is shown in Figure 3.10

In the application created as part of this work, Rosbridge is used to communicate between the application running on the VR headset and the ROS-based MRS Platform. The open-source ROSBridgeLib[3] library for Unity is used to integrate the Rosbridge interface into the code. This library provides programmed classes for subscribing and publishing ROS Topics

[3] ROSBridgeLib codebase: `https://github.com/MathiasCiarlo/ROSBridgeLib`

but does not support calling ROS Services directly. As Services are part of the communication between the application and the MRS Platform, a Controller package that converts Topics to Services was introduced. Its only function is to subscribe to predefined ROS Topics, parse received messages and call corresponding ROS Services.

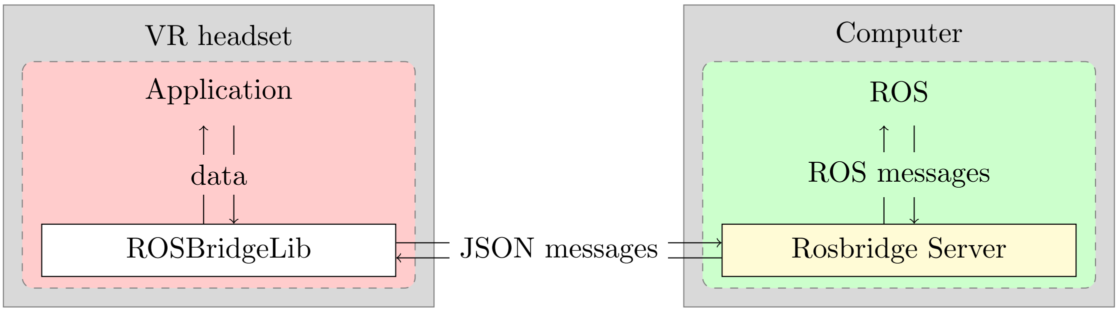



Figure 3.10: Rosbridge communication diagram.

### 3.2.2 Controller package

The Controller package is a package running as a ROS Node that takes care of converting messages from ROS Topic to ROS Service. As can be seen in Figure 3.11, it is situated between the Rosbridge Server and the MRS Platform.

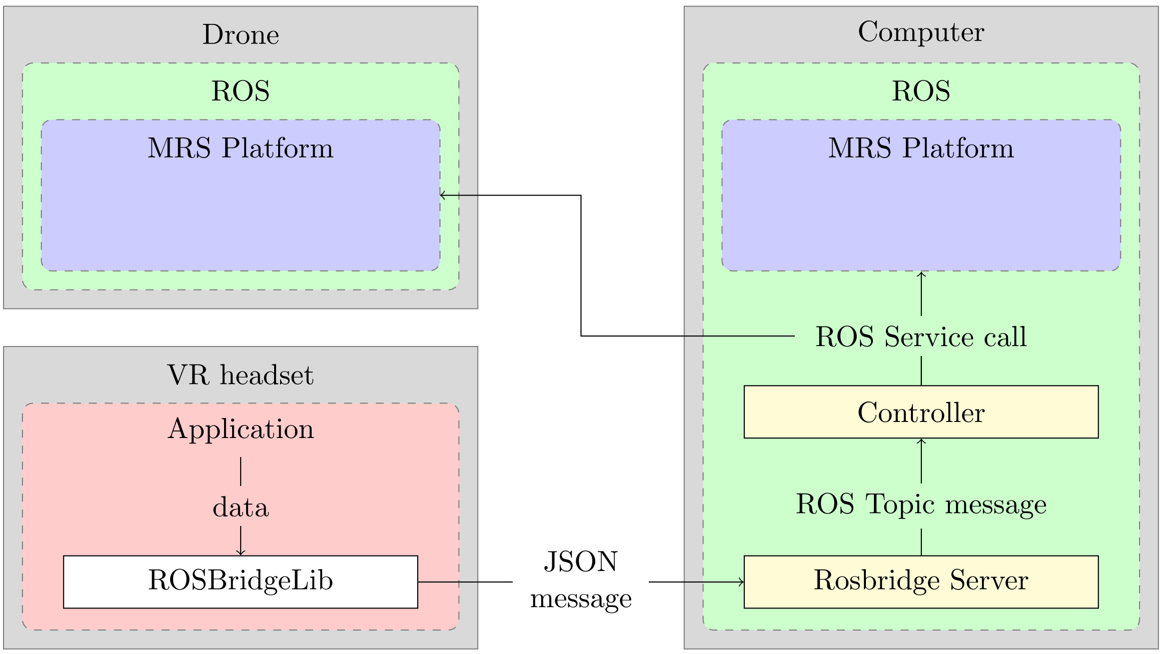



Figure 3.11: Controller package communication diagram.

The Controller package is used in cases where drones are commanded to move to a new position. The application running on the VR headset publishes, via Rosbridge, the target position coordinates for each UAV in the form of `geometry_msgs/Point` messages, whose structure is shown in Definition 3.2. The messages are published in two separate ROS Topics.

The Controller Node subscribes to the messages and calls the corresponding Services on the MRS Platform.

```
# This contains the position of a point in free space
float64 x
float64 y
float64 z
```

Definition 3.2: geometry_msgs/Point.msg [38].

### 3.2.3 Octomap Merger package

A robot avoidance detection system can be reactive, proactive, or both reactive and proactive. In the reactive approach, the robot only takes action at the moment when a collision with a barrier is detected. This behaviour is favourable in many scenarios, e.g., in real-time collision avoidance in a robotic arm [39]. However, the reactive method is not very useful in cases when the robot's movement can not be changed immediately, which is typical for UAVs [40]. In a proactive collision avoidance scenario, the robot actively checks for possible future collisions and tries to avoid them. A benefit of being proactive is that the robot can save time by reacting in advance. A drawback is that the proactive method requires accurate knowledge of the environment. If an obstacle in the environment is not detected, the navigation system can lead the robot confidently to an unwanted collision.

One of the requirements for the collision avoidance system is to reflect both real and virtual barriers. Detecting both types of objects can not be physically achieved by the same sensor. Thus, a UAV must simultaneously collect data about real-world and virtual environments. The final environment model used for the navigation systems is created by merging the data together.

Whether the UAV is flying in the real world or in the Gazebo robotic simulator, it creates a map of the surrounding environment using sensors and the MRS Octomap Server package. The MRS Octomap Server package is described in Subsection 3.3.1. The Octomap Merger package proposed in this subsection is designed to add virtual obstacles to the map to meet the necessary properties of the collision avoidance system described above. As can be seen in Figure 3.12, the Octomap Merger package receives Octomap from the MRS Octomap Server package and point cloud from the application's virtual environment. Octomap Merger produces an enhanced Octomap containing both real-world and virtual obstacles to the MRS Octomap Planner package. More details on the MRS Octomap Planner package are given in Subsection 3.3.1. Octomap is further described in the following paragraphs.

#### Octomap

Octomap, as described in [41], is a mapping framework based on the Octree structure initially proposed by D. Meagher [42]. Octomaps are widely used in multi-robot systems, as they can store full environment models, including occupied space, free space and unknown areas. They use probabilistic estimation to eliminate noise from sensors and have low memory requirements at the same time. Another benefit of Octomap is that Octree is a multi-resolution structure. In cases where speed is crucial, it can retrieve a small-resolution map much faster than a high-resolution map would usually be retrieved. In Octree, search and insert operations take logarithmic time, which makes it easy to merge maps from different sensors or robots in real time.

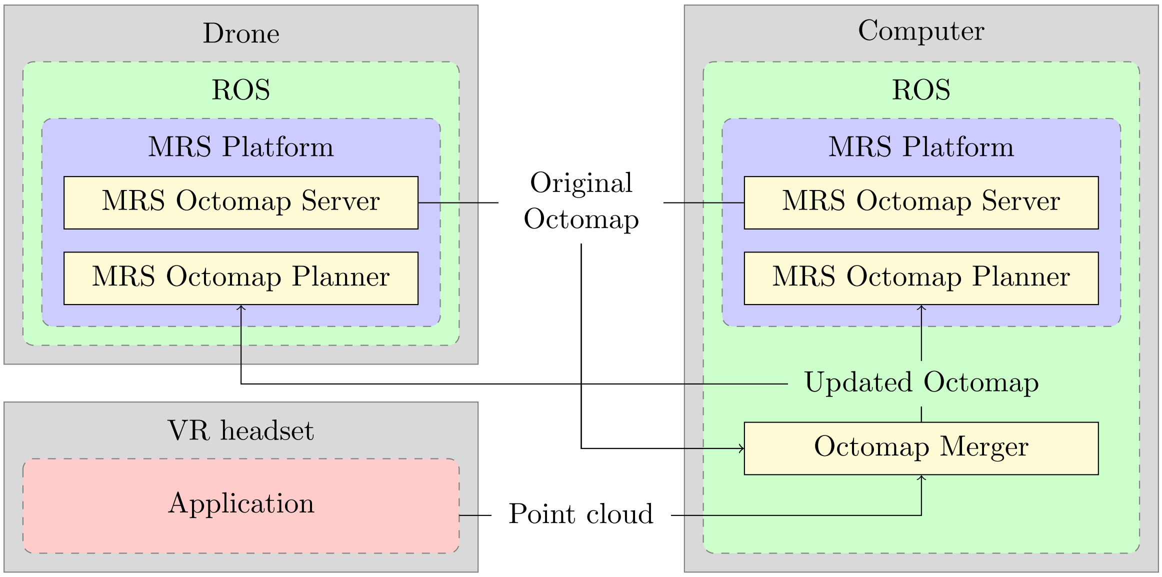


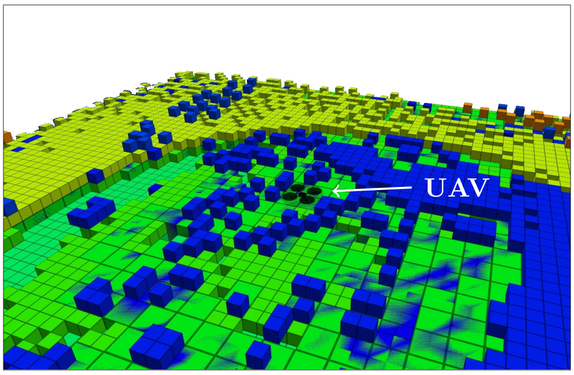

Figure 3.12: Octomap Merger package communication diagram.

In ROS, Octomap is represented by `octomap_msgs/Octomap` message. The structure of the message is shown in Definition 3.3.

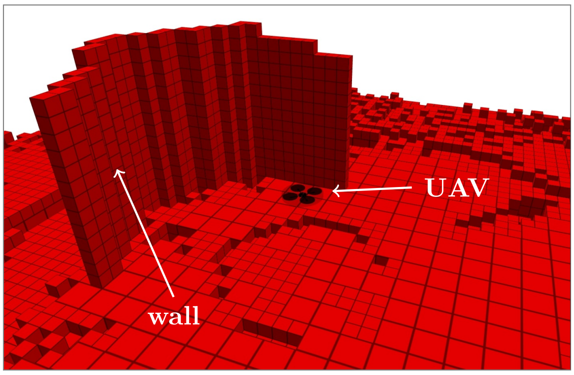
```
# A 3D map in binary format, as Octree
Header header

# Flag to denote a binary (only free/occupied) or full occupancy octree (.bt/.ot file)
bool binary

# Class id of the contained octree
string id

# Resolution (in m) of the smallest octree nodes
float64 resolution

# binary serialization of octree, use conversions.h to read and write octrees
int8[] data
```

Definition 3.3: octomap_msgs/Octomap.msg [43].

Figure 3.13 shows visualisations of the Octomap in RViz. Figure 3.13a shows the Octomap captured by the UAV sensors and constructed by the MRS Octomap Server package. Each coloured block indicates an occupied position in the surrounding environment. The space without blocks is either free of obstacles or yet to be explored. The blocks are distinguished by colour according to their relative height to the UAV. A different Octomap can be seen in the Figure 3.13b. This Octomap was created based on the original map with virtual obstacles added to it from the application's virtual environment. It is shown in red for easy differentiation from the original one. From both figures, it can be seen that there is a wall in the virtual environment, while the real UAV is surrounded only by free space in the real-world environment. The planning algorithms then use the final (red) Octomap. If the original Octomap had been used, the UAV would have ignored the virtual obstacles.




(a) Octomap captured by UAV sensors.




(b) Octomap merged with virtual obstacles.

Figure 3.13: Visualisation of Octomap captured by UAV sensors (a) and visualisation of the same Octomap merged with virtual obstacles (b).

## 3.3 Drones

In this thesis, drones are programmable physical UAVs on which the MRS Platform can be run. Each drone should contain sensors that allow it to locate itself, orient itself in the surrounding environment and communicate with the computer and other drones within the swarm. This section describes the parts of the MRS Platform relevant to the control system. More details on the hardware of the drones can be found in Section 4.1.

### 3.3.1 MRS Platform

MRS Platform is a ROS-based open-source software developed by Multi-robot Systems (MRS) research group at CTU in Prague [19], allowing the deployment of multiple multi-rotor helicopters outdoors and indoors. It bundles all essential UAV functions from takeoff and trajectory planning to the collision avoidance system. The MRS Platform can run on a computer and simulate virtual UAVs or run on board a physical drone and control it. It is capable of advanced simulations allowing proper testing of the software before deploying it on real hardware. This subsection presents selected parts of the MRS Platform. Due to the comprehensiveness of the system, it focuses mainly on the packages utilised in this thesis.

#### UAV Core

UAV Core[4] bundles together all crucial packages needed for UAV operation to make the packages comfortable to use. It consists of the following packages: MRS Bumper, MRS Lib, MRS Mavros Interface, MRS ROS Messages, MRS Rviz Plugins, MRS UAV Controllers, MRS UAV General, MRS UAV Managers, MRS UAV Odometry, MRS UAV Status, MRS UAV Testing, MRS UAV Trackers and MRS UAV Trajectory Generation. MRS Bumper is used for 1D LiDAR, 2D LiDAR, and depth-camera data aggregation. MRS Lib consists of different data structures and algorithms used across the MRS Platform. MRS Mavros Interface integrates MAVLink (Micro Air Vehicle Link protocol) extendable communication. MRS ROS Messages package consists of ROS Messages used throughout the MRS Platform.

[4]UAV Core codebase: `https://github.com/ctu-mrs/uav_core`

MRS Rviz Plugins package contains packages for the Rviz tool. MRS UAV Controllers can control UAVs by setting a desired angular rate and thrust. MRS UAV General contains launch files, configurations, and utilities for other MRS packages. MRS UAV Managers package includes high-level flight managers. MRS UAV Odometry is responsible for state estimation of the UAV dynamics. MRS UAV Status visualise information about the UAV state and its sensors. MRS UAV Testing package consists of testing utilities. MRS UAV Trackers package is used for UAV navigation. MRS UAV Trajectory Generation is used for generating trajectory to a goal position.

#### MRS UAV Status package

MRS UAV Status package[5] is a terminal user interface for real-time monitoring and control. It displays essential state and sensor information and allows the user to take control of the UAV and remotely control it via SSH.

#### MRS Octomap Server package

MRS Octomap Server package[6] running on each UAV collects data from its LiDAR using ROS Topic. It uses this data to create a map of the surrounding environment in Octomap format. The Octomap is described in Subsection 3.2.3. Probabilistic estimation is used to generate the map to eliminate noise and dynamic objects. If the LiDAR beam passes through a point, that point is considered as empty. If the beam hits an obstacle at a certain point, that point is saved in a buffer. The point is considered an obstacle only after several more confirmations about its existence. All other points are considered unknown. ROS Topic publishes the real-time environment map to the Octomap Merger package running on the computer. The Octomap Merger package is described in Subsection 3.2.3.

#### MRS Octomap Planner package

The primary role of the MRS Planner package[7] is to plan the trajectory of the UAV to ensure that it does not hit any obstacles. To work correctly, it needs an Octomap, which it gets from the MRS Server package or from the Octomap Merger package if it is intended to take virtual obstacles into account. The A* Search algorithm with custom heuristics is used to plan the shortest trajectory.

[5] MRS UAV Status package codebase: `https://github.com/ctu-mrs/mrs_uav_status`
[6] MSR Octomap Server package codebase: `https://github.com/ctu-mrs/mrs_octomap_server`
[7] MRS Octomap Planner package codebase: `https://github.com/ctu-mrs/mrs_octomap_planner`

# Chapter 4

# Experimental verification

Experimental verification is a crucial part of every research in the field of robotics. Two experiments were conducted to validate the human-swarm interaction system proposed in this thesis. The first experiment was executed in a simulated environment. For the simulation, Gazebo robotic simulator and RViz 3D visualiser were used. The second experiment was performed on real hardware in an open environment. Hardware from the Multi-robot Systems (MRS) research group at CTU in Prague [44] was used to perform the experiment. Both experiments, including the software and hardware tools used, are described in this chapter.

This chapter is organised as follows. Section 4.1 describes the software and hardware used for experimental verification. Section 4.2 describes the simulated tests in Gazebo robotic simulator. Section 4.3 describes the experimental verification conducted with a physical UAV.

## 4.1 Tools for verification

Several software and hardware platforms were used for the experimental verification of the proposed system. This section describes the details of the tools used.

### 4.1.1 Software platform

The software platforms for simulation and visualisation were essential in the development of the proposed human-swarm interaction system. It allowed straightforward testing of implemented features, simple debugging and quick feedback to the developer. Gazebo robotic simulator was used to emulate a real-world environment. RViz 3D visualiser was used to visualise sensor data and validate all modules' correctness. Both systems are further described below.

#### Gazebo robotic simulator

Gazebo is an open-source 3D robotic simulator with a physics engine, sensor simulations, and a wide range of customisable robot models and environments. In robotics, it is often used for testing, developing, and simulating robots in a virtual environment. It allows users to validate their software before implementing it on real-world hardware, saving time and reducing costs. Gazebo offers easy integration with ROS software.

In this work, the Gazebo robotic simulator was used to simulate virtual UAVs. It can run any number of virtual UAVs without the need for complex hardware preparation. Gazebo runs on a computer, and its only limitation is the computer processing capacity. Each additional UAV added to the simulation is a separate unit requiring all sensors and processes to be simulated. Two simulated UAVs in the Gazebo robotic simulator are shown in Figure 4.1.

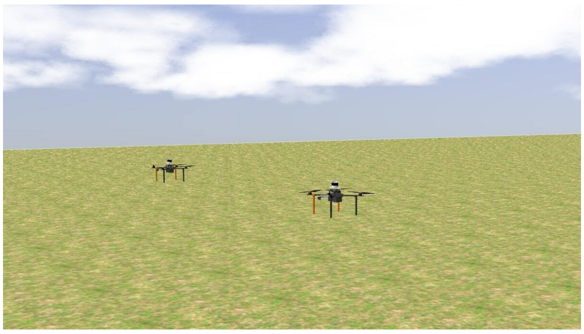


Figure 4.1: Two UAVs in Gazebo robotic simulator.

#### RViz 3D visualiser

RViz is a 3D visualisation tool developed for ROS that allows users to visualise and interact with data from various sensors and robots. Its graphical user interface provides a convenient way to display and manipulate various data types, including point clouds, laser scans, maps, and robot models. RViz is a powerful tool for debugging and testing ROS applications. In this work, RViz was used to verify that the data sent between processes were correct and in the appropriate format.

Figure 4.2 shows a UAV's surrounding environment in the RViz visualiser. It was created using captured data from the UAV's sensors. Each coloured block indicates an occupied position, and the relative height of each block to the UAV is distinguished by colour.

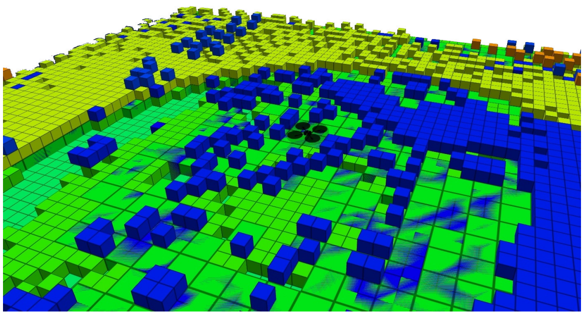


Figure 4.2: RViz visualisation tool showing a map of a UAV's surrounding environment.

### 4.1.2 Hardware platform

Physical hardware was necessary to validate the control system's endurance in real-world conditions. The hardware used for the experimental verification with physical UAVs is described in this section.

#### MRS Modular UAV Hardware Platform

For the validation of the control system's functionality and endurance in a real-world environment, the MRS Modular UAV Hardware Platform was used [44]. The platform allows the combination of custom UAV frames with various devices such as LiDAR, cameras, GPS, gyroscope, accelerometer, and many more. Furthermore, its compatibility with the software MRS Platform used in this work makes it ideal for experimental verification without extra preparation overhead. Figure 4.3 shows UAVs built from the MRS Modular UAV Hardware Platform.


Figure 4.3: Different types of UAVs built from the MRS Modular UAV Hardware Platform.

Experiments conducted as a part of this work used a UAV equipped with GPS, gyroscope, accelerometer and 2D LiDAR. GPS was used to determine the current position of the UAV. The gyroscope and accelerometer helped the UAV to maintain stability during flight. The 2D LiDAR allowed the software to create a map of the surroundings.

## 4.2 Simulation experiment

A small city model from MRS Resources[1] was chosen to verify the system's functionality in a simulation experiment. This type of environment was chosen for its many obstacles, such as buildings, poles, flagpoles, cars, or playgrounds. These obstacles are suitable for testing a smooth operation of the integration of this simulated environment with virtual barriers in a user application running on a VR headset. A small city environment is also suitable for testing the robustness of the control system in cluttered conditions.

In the experiment, the human operator was tasked to safely move a swarm of three UAVs from a start to a target position. An essential condition for completing the task was to limit the UAV swarm flight in potentially dangerous areas with a high probability of people being present (e.g., houses and playgrounds). During the experiments, the time in which the human operator was able to complete the task was measured.

Figure 4.4 shows an overhead view of the city model used in the experimental verification. The figure shows the start and destination areas of the UAV swarm and the dangerous area the UAVs were to avoid. In general, all the obstacles visible in the figure were to be avoided during the swarm navigation.

For more reliable results, three measurements were conducted. It should be noted that in all cases, the human operator of the control system was a person familiar with the system. The measurement results and discussion are described in the following subsection.

[1] MRS Resources for Gazebo: `https://github.com/ctu-mrs/mrs_gazebo_common_resources`

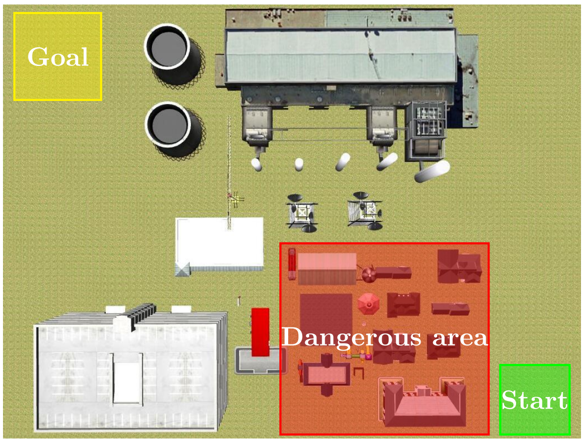



Figure 4.4: Overhead view of the city model used in the simulation experiment.

### 4.2.1 Results and discussion

A total of three measurements were conducted. The human operator successfully navigated the UAV swarm to the target area in all three measurements. At the same time, in all measurements, the UAVs successfully avoided the dangerous area. The total times of individual experiments are shown in Table 4.1. The average time to complete the task was 197.44 seconds.

| Experiment | Time |
|---|---|
| 1 | 209.47 s |
| 2 | 186.60 s |
| 3 | 196.25 s |

Table 4.1: Total times of individual tests.

Figure 4.5 shows the measured trajectories of each UAV in the swarm during the third experiment, which lasted 196.25 seconds. The figure shows the swarm avoiding the dangerous area on whose border the human operator inserted virtual walls. Once the UAVs had bypassed the dangerous area, their trajectories to the target were more straightforward as the individual UAVs only avoided obstacles in the simulated world.

The control of the swarm was well managed, and the control system proved its functionality. However, during the experiment, it was observed that it is arduous for the human operator to delineate the areas where the swarm should not fly properly. This issue is related to the operator having no data from the UAV sensors other than their position. Therefore, it is difficult for a human operator, who cannot see real obstacles with the headset, to estimate the correct position to place obstacles in virtual reality.

## 4.3 Real-world experiment

Experiments in real-world environments are more challenging to perform compared to simulated experiments. Experiments on real hardware cannot be performed as often, are more expensive, and are more prone to unpredictable failures. Nevertheless, they are a crucial part of the testing of robotic systems as they can reveal flaws that have remained hidden in the simulation.

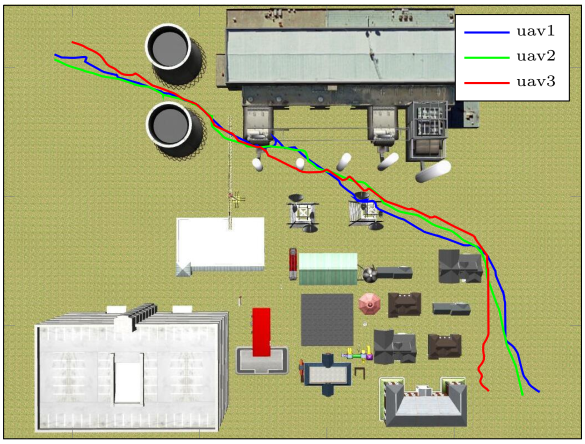


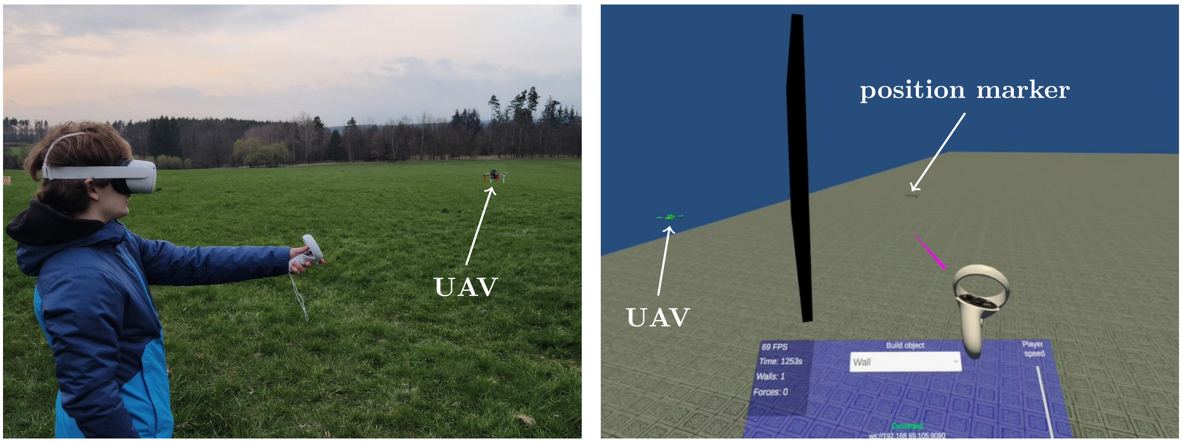

Figure 4.5: Trajectories of individual UAVs during the third experiment.

In this work, one real-world experiment was conducted to verify the control system's robustness and functionality. The experiment was executed using the MRS Modular UAV Hardware Platform [44], and its main objective was to confirm that the control system could manage physical UAVs. In the experiment, the human operator was tasked to build a virtual wall and command a UAV to fly from one side to the other. The ability to avoid virtual obstacles in a real-world environment was monitored. The test results and related discussion can be found in the following subsection.

### 4.3.1 Results and discussion

A photo of the execution of the experiment and the operator's view in the application are shown in Figure 4.6a and Figure 4.6b, respectively. When flying, the UAV controlled by the control system's operator was able to account for virtual obstacles in the environment and avoid them. However, due to the unwanted shift in the coordinate system, the UAV avoided the virtual walls in a different location than they were actually positioned. Unfortunately, this deficiency could not be eliminated during the duration of the test, and thus the experiment can be considered only partially successful. There was no further opportunity to repeat the experiment before the thesis deadline.

Another observation from the experiment was the degraded motion tracking quality of the headset controllers. As mentioned in this thesis, Meta Quest 2 is unsuitable for use outdoors. Furthermore, direct exposure to sunlight can damage the headset's sensors. In our experiment, the cloudy weather allowed us to use the headset outdoors and shield it from direct sunlight. However, the adverse effects on its sensors were still visible.




(a) Human operator commanding a UAV.

(b) Human operator's view in virtual reality.

Figure 4.6: Human operator commanding a UAV to change its position (a) and the operator's view in virtual reality (b).

# Chapter 5

# Comparison with existing methods

This chapter begins with a brief discussion of various aspects of the proposed system and compares the system with other existing solutions in human-swarm interaction research. Section 5.1 presents a case study that compares the system proposed in this thesis with a system utilising a k-nearest neighbour based action classifier [16].

The proposed control system for human-swarm interaction using virtual reality has various benefits and drawbacks compared to other existing solutions. Some solutions provide the human operator with a complete overview of the swarm surrounding environment, which the proposed system does not allow [25]. Other solutions utilise haptic feedback control but do not benefit from the use of virtual reality [9]. There are also systems designed for a specific application and specifically adapted to it [3], [36]. Some systems are very similar to the system proposed in this work, allowing a human operator to place obstacles into the virtual environment to influence real UAVs [12].

Overall, many different methods are used within the research field, and each approaches the question of human-swarm interaction slightly differently.

## 5.1 Case study

This section describes a case study that compares the proposed human-swarm interaction control system with a system utilising a k-nearest neighbour based action classifier [16]. Subsection 5.1.1 compares eight specific aspects of the two systems and presents a table summarising these differences.

An experiment with the k-NN classifier-based system was conducted to validate the system's features. A swarm of three UAVs was flown in a simulated open environment with a human operator controlling the warm with body gestures. Figure 5.1a shows the human operator controlling a group of simulated UAVs in the Gazebo robotic simulator. The controlled UAVs are shown in Figure 5.1b.

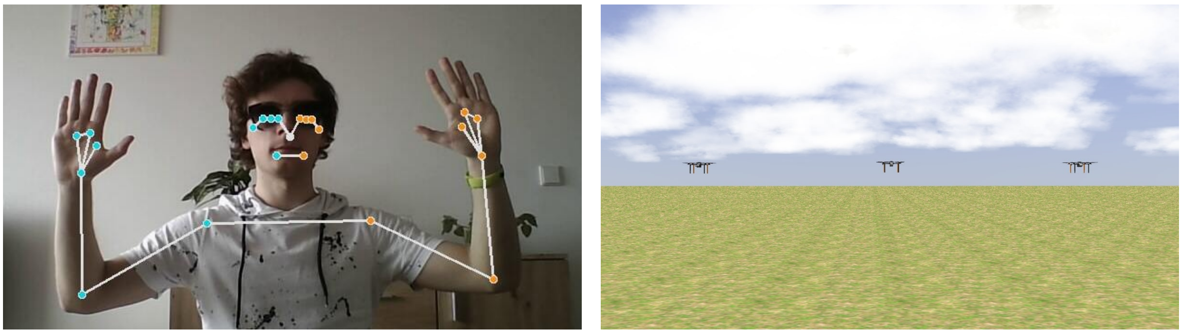
### 5.1.1 Feature comparison

Both compared systems are based on a human-swarm interaction interface built on top of the MRS Platform. They offer remote control of a UAV swarm intending to enhance human-swarm interaction. They are straightforward and require only one human operator to control the systems. Eight specific aspects of the two systems are compared:

- Action detection time — represents the time between the issuing of the command and its processing by the swarm.
- Operator position — describes the human operator's position relative to a swarm of UAVs.

- Special equipment — deals with whether any special equipment is needed to control the swarm.
- Physical obstacle detection — indicates whether UAVs can respond to physical obstacles in the environment.
- Virtual obstacle detection — indicates whether UAVs can respond to virtual obstacles in the environment.
- Flocking — indicates whether the compared system is capable of managing UAV formations.
- Complete situation overview — indicates whether the human operator can have an overview of the entire UAV swarm when using the compared system.
- Localised commands — indicates whether the human operator can command the swarm to move to a specific position or to avoid a chosen area altogether.

Table 5.1 provides a summary comparison of these features of the two systems.



(a) Classification of the human operator gestures. (b) UAVs in the Gazebo robotic simulator.

Figure 5.1: Human operator (a) controlling a group of simulated UAVs (b) using the k-nearest neighbour based action classifier control system.

| Comapred feature | Proposed system | Classifier-based system |
|---|---|---|
| Action detection time | 0 ms | 20 ms |
| Operator position | indoors, WLAN connection | anywhere visible |
| Special equipment | Meta Quest 2 | none |
| Physical obstacle detection | yes | yes |
| Virtual obstacle detection | yes | no |
| Flocking | yes | yes |
| Complete situation overview | yes | no |
| Localised commands | yes | no |

Table 5.1: Summary of the comparison of the features of the system proposed in this work and the k-nearest neighbour based action classifier system presented in [16].

### Action detection time

While the system based on the k-NN classifier requires, on average, 20 ms to detect the gesture of a human operator, the application proposed in this thesis has no such detection time because commands are sent directly after the button on the headset's controller is pressed. As a result, the proposed application has less delay in transmitting commands. This comparison neglects the communication time between the VR headset and the swarm. It can be assumed that this time is the same for both systems.

#### Operator position

An essential role in the usability of a human-swarm interaction system is the operator's position relative to the UAV swarm. In the proposed control system, the human operator must be connected to the same WLAN as the UAVs they want to control. This does not allow the swarm and the operator to move too far apart. Another limitation when using the system is the Meta Quest 2 virtual reality headset, which does not allow a human to wear it outdoors. A system using a k-nearest neighbour based action classifier allows a human operator to be anywhere in sight of the swarm of UAVs, as only visual communication takes place between the swarm and the operator.

#### Special equipment

An essential factor in software usability, in general, are special hardware requirements needed by the software to operate correctly. While the system based on the k-nearest neighbour based classifier requires no special hardware except for drones, the application proposed in this work also requires a virtual reality headset. Specifically, Meta Quest 2 must be employed to use the software. These needs must be considered.

#### Physical obstacle detection

Both compared systems are capable of detecting and avoiding physical obstacles. They use the MRS Platform, on which both solutions are based. In this respect, there is no difference between the two compared systems.

#### Virtual obstacle detection

Of the two systems compared, only the one proposed in this work allows a human operator to insert virtual obstacles into the surroundings of the UAV swarm. The system allows the operator to place static meshes in specific areas where UAVs are undesirable to fly. Using artificial force fields, the operator can influence the swarm's movement through the selected locations.

#### Flocking

The two compared systems allow the UAV swarm to flock. However, the implementation of this behaviour is different. The system based on the k-NN classifier uses a leader-follower approach where only one UAV receives the command and the rest of the swarm follows it. The system presented in this thesis achieves the flocking behaviour by navigating each UAV within the swarm independently.

### Complete situation overview

The system proposed in this work gives a human operator a complete overview of the swarm. The operator wearing a VR headset can see the positions and movements of all UAVs and can reposition themselves freely in the virtual environment. A system built on a k-NN action classifier does not allow a similar overview. The human operator has only as much overview of the current situation as their visual perspective allows.

### Localised commands

Localised commands allow the operator to command the swarm to move to a specific position or to avoid a chosen area altogether. While the system utilising the k-nearest neighbour based action classifier allows the human operator to only move the swarm relatively to its current position (e.g., forward, left, right), the application proposed in this work allows the operator to specify the exact location where the swarm should move to. This gives the operator more options in interacting with the UAV swarm.

# Chapter 6

# Conclusion

The virtual-reality-based human-swarm interaction control system presented in this thesis demonstrated the ability to control a swarm of real or virtual UAVs and influence it using virtual objects. The solution proved suitable for tasks involving dangerous areas the swarm was required to avoid. However, it was observed to be arduous for the human operator to identify its position in the virtual environment relative to the real world. Another factor reducing the effectiveness of swarm management was the lack of information about the surrounding environment. It can be stated that while the proposed solution is beneficial in human-swarm interaction applications, its adoption in practice will require the incorporation of UAVs' sensor data into the user interface or the utilisation of augmented reality. Future research in this area should be devoted to mitigating these deficiencies.

# Chapter 7

# References


[1] Zhu Xiaoning. “Analysis of military application of UAV swarm technology”. In: *2020 3rd International Conference on Unmanned Systems (ICUS)*. IEEE. 2020, pp. 1200–1204.

[2] Hazim Shakhatreh et al. “Unmanned aerial vehicles (UAVs): A survey on civil applications and key research challenges”. In: *Ieee Access* 7 (2019), pp. 48572–48634.

[3] Amir M Naghsh et al. “Analysis and design of human-robot swarm interaction in firefighting”. In: *RO-MAN 2008-The 17th IEEE International Symposium on Robot and Human Interactive Communication*. IEEE. 2008, pp. 255–260.

[4] Ioannis Mademlis et al. “High-level multiple-UAV cinematography tools for covering outdoor events”. In: *IEEE Transactions on Broadcasting* 65.3 (2019), pp. 627–635.

[5] Manuele Brambilla et al. “Swarm robotics: a review from the swarm engineering perspective”. In: *Swarm Intelligence* 7 (2013), pp. 1–41.

[6] Ahmad Reza Cheraghi, Sahdia Shahzad, and Kalman Graffi. “Past, present, and future of swarm robotics”. In: *Intelligent Systems and Applications: Proceedings of the 2021 Intelligent Systems Conference (IntelliSys) Volume 3*. Springer. 2022, pp. 190–233.

[7] Andreas Kolling et al. “Human interaction with robot swarms: A survey”. In: *IEEE Transactions on Human-Machine Systems* 46.1 (2015), pp. 9–26.

[8] Nicolas Gio, Ross Brisco, and Tijana Vuletic. “Control of a drone with body gestures”. In: *Proceedings of the Design Society* 1 (2021), pp. 761–770.

[9] Steven Nunnally et al. “Using haptic feedback in human robotic swarms interaction”. In: *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*. Vol. 57. 1. SAGE Publications Sage CA: Los Angeles, CA. 2013, pp. 1047–1051.

[10] Daniel Biediger, Arun Mahadev, and Aaron T Becker. “Investigating the survivability of drone swarms with flocking and swarming flight patterns using virtual reality”. In: *2019 IEEE 15th International Conference on Automation Science and Engineering (CASE)*. IEEE. 2019, pp. 1718–1723.

[11] Andreas Kolling, Steven Nunnally, and Michael Lewis. “Towards human control of robot swarms”. In: *Proceedings of the seventh annual ACM/IEEE international conference on human-robot interaction*. 2012, pp. 89–96.

[12] Inmo Jang et al. “Omnipotent virtual giant for remote human-swarm interaction”. In: *arXiv preprint arXiv:1903.10064* (2019).

[13] Abdullah Ramdhani, Muhammad Ali Ramdhani, and Abdusy Syakur Amin. “Writing a Literature Review Research Paper: A step-by-step approach”. In: *International Journal of Basic and Applied Science* 3.1 (2014), pp. 47–56.

[14] Aaron Steinfeld et al. “Common metrics for human-robot interaction”. In: *Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*. 2006, pp. 33–40.

[15] Maxime Vaidis and Martin J-D Otis. “Swarm robotic interactions in an open and cluttered environment: a survey”. In: *Designs* 5.2 (2021), p. 37.

[16] Akash Chaudhary, Tiago Nascimento, and Martin Saska. “Controlling a Swarm of Unmanned Aerial Vehicles Using Full-Body k-Nearest Neighbor Based Action Classifier”. In: *2022 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE. 2022, pp. 544–551.

[17] Camillo Lugaresi et al. "Mediapipe: A framework for building perception pipelines". In: *arXiv preprint arXiv:1906.08172* (2019).

[18] Nathan Koenig and Andrew Howard. "Design and use paradigms for gazebo, an open-source multi-robot simulator". In: *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*. Vol. 3. IEEE. 2004, pp. 2149–2154.

[19] Tomas Baca et al. "The MRS UAV system: Pushing the frontiers of reproducible research, real-world deployment, and education with autonomous unmanned aerial vehicles". In: *Journal of Intelligent & Robotic Systems* 102.1 (2021), p. 26.

[20] Zhengyou Zhang. "Microsoft kinect sensor and its effect". In: *IEEE multimedia* 19.2 (2012), pp. 4–10.

[21] Javier Alonso-Mora et al. "Gesture based human-multi-robot swarm interaction and its application to an interactive display". In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2015, pp. 5948–5953.

[22] Dongjun Lee et al. "Haptic teleoperation of multiple unmanned aerial vehicles over the internet". In: *2011 IEEE International Conference on Robotics and Automation*. IEEE. 2011, pp. 1341–1347.

[23] Lawrence H Kim and Sean Follmer. "Swarmhaptics: Haptic display with swarm robots". In: *Proceedings of the 2019 CHI conference on human factors in computing systems*. 2019, pp. 1–13.

[24] Heather C Lum et al. "Virtual Reality: History, Applications, and Challenges for Human Factors Research". In: *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*. Vol. 64. 1. SAGE Publications Sage CA: Los Angeles, CA. 2020, pp. 1263–1268.

[25] Mingxuan Chen et al. "A multichannel human-swarm robot interaction system in augmented reality". In: *Virtual Reality & Intelligent Hardware* 2.6 (2020), pp. 518–533.

[26] Dorin Ungureanu et al. "Hololens 2 research mode as a tool for computer vision research". In: *arXiv preprint arXiv:2008.11239* (2020).

[27] Arunim Bhattacharya and Sachit Butail. "Designing a Virtual Reality Testbed for Direct Human-Swarm Interaction in Aquatic Species Monitoring". In: *IFAC-PapersOnLine* 55.37 (2022), pp. 295–301.

[28] Matthew Joordens and Mo Jamshidi. "On the development of robot fish swarms in virtual reality with digital twins". In: *2018 13th Annual Conference on System of Systems Engineering (SoSE)*. IEEE. 2018, pp. 411–416.

[29] Andreas Kolling et al. "Human swarm interaction: An experimental study of two types of interaction with foraging swarms". In: *Journal of Human-Robot Interaction* 2.2 (2013).

[30] Peter Pirolli and Stuart Card. "Information foraging." In: *Psychological review* 106.4 (1999), p. 643.

[31] Michael A Goodrich et al. "Toward human interaction with bio-inspired robot teams". In: *2011 IEEE International Conference on Systems, Man, and Cybernetics*. IEEE. 2011, pp. 2859–2864.

[32] Phillip Walker et al. "Human control of leader-based swarms". In: *2013 IEEE International Conference on Systems, Man, and Cybernetics*. IEEE. 2013, pp. 2712–2717.

[33] Jacques Penders et al. "Guardians: a swarm of autonomous robots for emergencies". In: *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI'07) Workshop on Multirobotic Systems for Societal Applications*. 2007, pp. 8–16.

[34] Jeremi Gancet et al. "User interfaces for human robot interactions with a swarm of robots in support to firefighters". In: *2010 IEEE International Conference on Robotics and Automation*. IEEE. 2010, pp. 2846–2851.

[35] Jacques Penders et al. "A robot swarm assisting a human fire-fighter". In: *Advanced Robotics* 25.1-2 (2011), pp. 93–117.

[36] Shishir Bashyal and Ganesh Kumar Venayagamoorthy. "Human swarm interaction for radiation source search and localization". In: *2008 IEEE Swarm Intelligence Symposium*. IEEE. 2008, pp. 1–8.

[37] Robot Operating System. *sensor_msgs/PointCloud Documentation*. 2023. URL: `http://docs.ros.org/en/noetic/api/sensor_msgs/html/msg/PointCloud.html` (visited on 02/20/2023).

[38] Robot Operating System. *geometry_msgs/Point Documentation*. 2023. URL: `http://docs.ros.org/en/noetic/api/geometry_msgs/html/msg/Point.html` (visited on 03/12/2023).

[39] Vladimir J Lumelsky and Edward Cheung. "Real-time collision avoidance in teleoperated whole-sensitive robot arm manipulators". In: *IEEE Transactions on Systems, Man, and Cybernetics* 23.1 (1993), pp. 194–203.

[40] Dávid Rau, Jozef Rodina, and Filip Štec. "Generating instant trajectory of an indoor UAV with respect to its dynamics". In: *2020 23rd International Symposium on Measurement and Control in Robotics (ISMCR)*. IEEE. 2020, pp. 1–5.

[41] Kai M Wurm et al. "OctoMap: A probabilistic, flexible, and compact 3D map representation for robotic systems". In: *Proc. of the ICRA 2010 workshop on best practice in 3D perception and modeling for mobile manipulation*. Vol. 2. 2010, p. 3.

[42] Donald Meagher. "Geometric modeling using octree encoding". In: *Computer Graphics and Image Processing* 19.2 (1982), pp. 129–147. ISSN: 0146-664X. DOI: `https://doi.org/10.1016/0146-664X(82)90104-6`.

[43] Robot Operating System. *octomap_msgs/Octomap Documentation*. 2023. URL: `http://docs.ros.org/en/noetic/api/octomap_msgs/html/msg/Octomap.html` (visited on 02/20/2023).

[44] Daniel Hert et al. "MRS modular UAV hardware platforms for supporting research in real-world outdoor and indoor environments". In: *2022 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE. 2022, pp. 1264–1273.