Master Thesis

Swarm Drones Control Platform

A modular and scalable platform for swarm drone control

Nicolas Leboucher

October 15, 2024

Supervisors
Dr Xiao Xiao and Dr Marc Teyssier Professor at IFT

Contents

1 Introduction

1.1 Background and Motivation

Swarm robotics is an emerging field in robotics inspired by biological systems, such as flocks of birds or colonies of ants, where simple units collaborate to achieve complex tasks. In recent years, advancements in drone technology have expanded the potential applications of swarm robotics, including environmental monitoring, search-and-rescue operations, and industrial automation [1]. In particular, drones operating as swarms offer a promising solution for tasks requiring adaptability and scalability.

One key area of interest is the integration of drone swarms into environments shared with humans. Applications such as drone-based assistance systems in healthcare, smart warehouses, and collaborative robotics emphasize the need for reliable, safe human-drone interaction [1]. The use of drones in indoor environments, where traditional positioning systems such as GPS are not viable, presents both opportunities and challenges, particularly when human safety and interaction are prioritized.

In this context, existing swarm control platforms like Crazyswarm and CFlib offer robust tools for drone coordination and control. However, these platforms often require specialized equipment, such as motion capture systems, and rely on Python-based libraries, limiting accessibility for non-expert users and those working in non-research settings [2, 3]. This raises a gap: a need for more accessible, intuitive control systems that allow seamless interaction between human operators and drone swarms in real-time indoor environments.

1.2 Problem Statement

While existing platforms provide powerful tools for swarm control, they do not address the challenge of creating a system that integrates human pose detection with real-time drone control in a user-friendly and accessible way. Most current systems are limited in their ability to allow developers to work directly with drones in environments where they need to safely interact with human operators.

This research addresses this gap by proposing a web-based platform for controlling swarms of Crazyflie drones in indoor spaces using real-time human pose estimation. The platform aims to enable intuitive and responsive interaction between humans and drones without requiring complex external systems.

We want to have our scene of drones including our operators and drones. Similarly in red is the drone system and in yellow the human system with the human pose detection. In red above there is the drone localization system. In grey above is the human localization system.

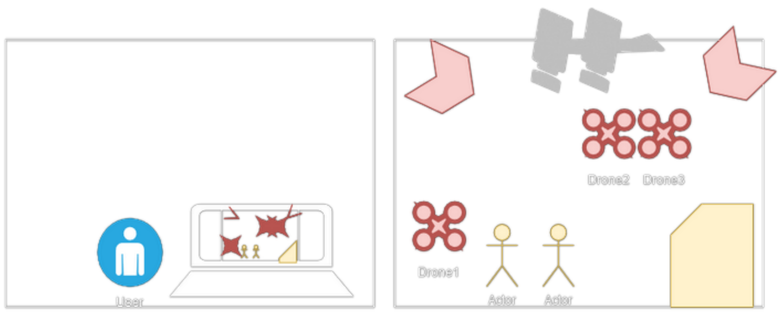


Figure 1: Our setup for controlling drones indoors around humans. Similarly in red is the drone system and in yellow the human system with the human pose detection. In red above there is the drone localization system. In grey above is the human localization system.

1.3 Research Objectives

This research aims to address the limitations of existing swarm control platforms by developing a user-friendly, web-based system for controlling swarms of drones in indoor environments where interaction with humans is required. The main objectives of this research are as follows:

- Develop a web-based platform for controlling swarms of Crazyflie drones, enabling easier access for developers, researchers, and non-experts.

- Integrate real-time human pose detection using affordable and accessible technologies, such as Intel RealSense and MediaPipe [4].
- Ensure safe and responsive drone interactions in indoor environments by integrating collision avoidance systems and real-time feedback.
- Evaluate the platform's performance in both controlled and dynamic indoor environments with human interaction.

This research will demonstrate how a web-based interface, coupled with advanced human pose detection, can make drone swarm control more accessible and enhance human-drone collaboration in shared spaces.

1.4 Significance of the Research

The proposed platform has the potential to significantly impact the field of swarm robotics by broadening access to swarm control technologies. Currently, systems like *CrazySwarm* and *CFLib* require Python-based development, which limits their use to researchers or developers with specific technical expertise [2, 3]. By shifting to a web-based architecture, this research will make drone control systems more accessible to a wider audience, enabling cross-platform development and more intuitive interfaces.

Furthermore, by integrating real-time human pose detection, this platform introduces a novel method for enabling drones to interact with humans safely in indoor environments. This has wide-ranging applications, from smart warehouses, where drones assist human workers [5, 6]. The platform also has the potential to be used in educational settings, allowing students to interact with drones without complex technical setups.

Ultimately, this research fills a critical gap in current swarm robotics systems, creating a foundation for future development in both research and practical applications, particularly in areas where drone-human interaction is necessary.

1.5 Scope and Limitations

The proposed platform has limitations that should be considered. First, the platform is designed for indoor environments, where GPS signals are unreliable, and human interaction is a primary concern. It can not work outdoors.

The platform will not adress internal positioning of the drones, only external positioning.

The platform will not address drone swarm control in outdoor or GPS-based environments.

The platform will use real-time human pose detection technologies like Intel RealSense and MediaPipe, which may have limitations in complex environments.

Despite these limitations, the proposed platform offers a significant advancement in swarm robotics by providing a user-friendly interface for controlling drones in indoor environments around humans. The platform's focus on accessibility and safety makes it a valuable tool for researchers, developers, and educators interested in swarm robotics and human-drone interaction.

1.6 Structure Overview

Chapter 1 will review the State of the Art, including existing swarm control platforms like CrazySwarm, ModQuad, and human pose detection systems.

Chapter 2 will describe the design and implementation of the proposed Drone Control Kernel.

Chapter 3 Will describe the integration of real-time human pose detection.

Chapter 4 will present the platform for human-swarm interaction will present the experimental setup and results, validating the platform's functionality including possible extensions of the platform for larger-scale deployment.

Chapter 5 will discuss the conclusions

2 State of the Art

In recent years, swarm drone technology has advanced significantly, leading to breakthroughs in human-drone interaction and multi-robot systems. While several research efforts have focused on the safety, control, and interaction between drones and humans, there remains a gap in the availability of robust platforms for developers to control drone swarms indoors around humans. This section will explore existing works that demonstrate safe operation of drones around humans, systems that enable real-time drone control based on human input, and the identified gaps in the current technological landscape.

2.1 Safe Drone Flight Around Humans

Safety is a primary concern when operating drones in close proximity to humans, especially in confined spaces such as indoor environments. Several research efforts have focused on ensuring that drones can operate around humans without posing significant risks.

One such project is Drone & Me, which investigates how drones can safely navigate around humans by recognizing human poses and responding accordingly [7]. In addition, Ju et al. (2017) proposed real-time trajectory adjustments for drones to ensure safe flight paths when operating around humans indoors. These systems emphasize the importance of autonomous collision avoidance and the use of protective hardware to mitigate risks [8].

2.2 Controlling Drones with Human Input

Beyond safety, there has been significant progress in enabling human control over drones through various input methods, such as body gestures, hand movements, or wearable technology. The FlyJacket platform, for instance, allows a user to intuitively control a drone by wearing an exoskeleton that translates upper body movements into flight commands [5]. This system demonstrates that drones can be controlled naturally and in real-time based on human gestures, but it is limited to controlling a single drone at a time.

In a more general approach, Cai et al. (2019) explored how drones can follow human postures and gestures using computer vision, allowing for more flexible interactions with drones [9]. These systems highlight the potential for intuitive drone control but lack the scalability needed to control swarms of drones effectively.

2.3 Technological Platforms for Swarm Drones

Several robust platforms have been developed to enable scalable swarm drone systems. Among these, Crazyflie is a widely used platform for research on nano-quadrotor swarms. It provides the infrastructure for advanced coordination and trajectory planning of multiple drones in a shared airspace. The platform's compatibility with tools like ROS (Robot Operating System) makes it a flexible tool for swarm applications [2].

Another well-known platform is Kilobot, developed by Harvard's Wyss Institute. Though primarily a ground-based swarm robotics platform, Kilobot's decentralized control algorithms provide valuable insights into swarm intelligence and distributed behaviors, which are applicable to aerial drones as well [10]. ModQuad further builds on swarm modularity by allowing multiple drones to self-assemble mid-air, forming complex structures that can adapt to various tasks [11].

The CrazyChoir platform demonstrates how multiple Crazyflie drones can perform synchronized movements in an artistic or musical context, emphasizing scalability in both control and safety [12]. These platforms provide strong technological foundations for swarm robotics, yet none fully integrate human interaction into swarm control in indoor environments.

CrazySwarm [13] represents the current state of the art for large-scale swarm control in research environments, particularly in fields like swarm robotics, distributed systems, and human-swarm interaction. It leverages ROS for modularity and scalability, and its integration with real-time motion capture systems provides a sophisticated platform for studying collective behaviors. Conversely, CFLib Swarm focuses on providing accessible, low-level control of Crazyflie drones for smaller-scale applications, mainly for educational purposes and basic experimentation.

CrazySwarm provides a more advanced, scalable solution for handling large swarms in complex environments. It overcomes the limitations of CFLib [3] Swarm, such as lack of real-time positioning and high-level control, making it the preferred platform for research in swarm robotics.

We want to note here that in the Institute for Future Technologies there was a project that started with ROS but had issues as ROS adds a layer of abstraction and infrastructure that provides many advantages but also introduces overhead. [14]. CFLib avoids this overhead by interacting directly with the drones in a more lightweight package.

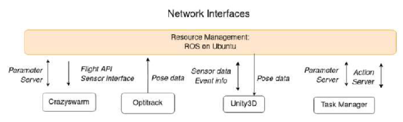


Figure 2: Drone Control Kernel developed by previous dronelab team at IFT.[14]

2.4 Human Pose Detection for Drone Control

Human-drone interaction has advanced with technologies that allow drones to react to human movements. One notable platform is FlyJacket, a wearable exoskeleton that translates upper body movements into drone control. FlyJacket emphasizes intuitive and immersive control, but it is focused on single-drone systems [5].

Beyond specialized wearables, recent advances in computer vision have made it possible to detect human poses and gestures without specialized hardware. Intel RealSense, combined with MediaPipe, allows real-time human pose estimation from RGB and depth cameras. This technology can detect skeletal movements and translate them into commands for drone control [4]. The system's accuracy and low latency make it highly applicable to environments where humans and drones must coexist.

In addition, Cai et al. (2019) proposed a monocular vision system for real-time posture recognition, where drones respond dynamically to human gestures in indoor environments. This system allows for direct interaction between humans and drones but does not yet support multi-drone systems [9].

Leap Motion offers another avenue for gesture-based control by detecting hand and finger movements. While Leap Motion has been successfully integrated with single drones, there is still potential to expand this to swarm systems for more complex indoor tasks [15].

The paper titled Ërone control using Kinect-based posture detection¨ [4] presents a compelling approach to enhancing human-drone interaction by leveraging RGB-D $Red, Green, Blue - Depth$ data from the Kinect sensor. The authors demonstrate how Kinect can effectively capture real-time human poses and positions, enabling drones to respond to user gestures with a high degree of accuracy. By utilizing the depth data, the system can determine the spatial relationship between the human operator and the drone, allowing for intuitive control commands based on body movements.

This approach significantly reduces the need for additional hardware or complex controllers, as it translates natural human actions directly into drone maneuvers. The paper emphasizes that such a system can enhance the usability and accessibility of drones, particularly in indoor environments where GPS signals may be unreliable. Furthermore, the authors highlight the potential for combining this technology with other methods, such as those explored in [4] and [5], to create a more robust human-drone interaction framework that addresses current limitations in the field.

2.5 Gaps in Human-Swarm Interaction Platforms

Despite advancements in both safety mechanisms and control systems, there remains a significant gap in providing a unified platform that enables developers to control swarms of drones indoors while safely interacting with humans. Current platforms like Crazyflie and ModQuad provide scalable solutions for drone swarms but lack the necessary integration with human pose recognition and safety protocols for operating indoors around people [2, 11]. Furthermore, while platforms like FlyJacket enable human control, they focus on single-drone control, leaving a gap in swarming capabilities.

Thus, a key challenge remains in the development of a platform that can:

Enable real-time control of multiple drones based on human input (e.g., gestures, body movements). That includes real-time posture and position as all platforms do not seem to include. [**cflib**, **crazyswarm**, 5] Ensure safe flight paths in indoor environments.

Be accessible to developers for customization and open-source integration. Python is accessible but we want to be able to use these drones in unity or a web service to not require client-drone connection.

3 Contribution 1: Swarm Control

3.1 Introduction

In this section, we present our first contribution, which is the drone control kernel. We present the software architecture that we have developed to control drones in a swarm. We also present the results of the evaluation of the drone control kernel.

We were provided a synchronous control driven API we have worked around it to handle practial asynchronous calls. We show here this work.

3.2 Drone Control Kernel

Introduction, Motivation and Objectives

Currently, Crazyflie drones are controlled using their Python library, cflib, which provides developers with a robust toolkit for sending commands, receiving sensor data, and managing flight tasks. While this library offers powerful capabilities, it limits the flexibility of using the Python ecosystem, as direct access to the drones requires interacting through this specific interface. As a result, broader accessibility for developers or users unfamiliar with Python is restricted, making it challenging for people from diverse technical backgrounds to engage with drone control.

To address this limitation and open drone control to a wider audience, we propose a web-based control platform. By developing a web interface, we aim to abstract the complexities of the underlying Python-based system and make the control of drone swarms more accessible to a larger range of developers, including those familiar with other technologies. This approach democratizes drone control, allowing users to interact with Crazyflie drones through web-based APIs and interfaces, facilitating easier integration into various applications, including human-drone interactions in indoor environments.

Instantaneous API Calls and Background Processing: Asynchronous APIs enhance the functionality of drone control applications by handling requests in the background. This approach maintains application functionality while keeping resources free to process new requests. Especially in drone operations where real-time responsiveness is crucial, this capability ensures efficient handling of multiple tasks simultaneously.

Adaptability to Connectivity and Execution Time: Asynchronous APIs are particularly beneficial in environments with variable connectivity or where requests have longer execution times. They allow for the processing of complex operations without causing delays in the application's responsiveness, crucial for real-time drone control where delay can have significant consequences.

Event-Driven Communication: Asynchronous APIs, enable more intelligent communication between internal and external services. This is especially important in drone operations where multiple events or commands may need to be handled in real time, ensuring smoother and more efficient workflows

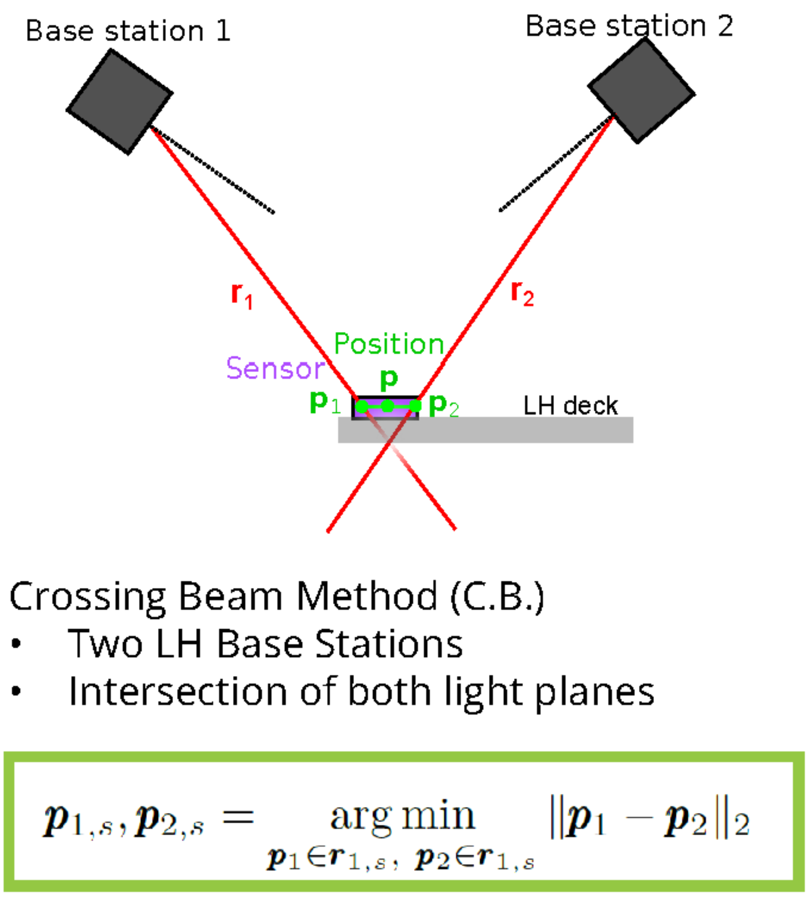
Support for Various Protocols: These APIs support a range of messaging protocols and transports such as WebSockets and UDP subscriptions, which are essential for the dynamic and varied requirements of drone control. This flexibility allows for fast and more accessible and versatile drone operation management.

Heritage and Components Used in the Project

The drone control kernel consists of several components that work together to provide a robust and flexible control system for managing drone swarms. Crazyflie drones being used in this project communicate using the Radio component Crazy Radio PA. We include also decks that can be added to the drones to add the location of the drones using light house technology. This requires a specific deck to be added to the drones and lighthouses to be placed around the room. The drones are then able to locate themselves in the room. [14]

Theory

According to the state of the art the drones can be controlled using different localization methods. The lab's legacy left us the Lighthouse positioning system. This system uses infrared light to locate the drones in space. The drones have a deck that can be added to them to allow them to locate themselves in space.



Crossing Beam Method (C.B.)

- Two LH Base Stations
- Intersection of both light planes

$$\boldsymbol{p}_{1,s}, \boldsymbol{p}_{2,s} = \underset{\boldsymbol{p}_1 \in \boldsymbol{r}_{1,s},\ \boldsymbol{p}_2 \in \boldsymbol{r}_{1,s}}{\arg\min} \|\boldsymbol{p}_1 - \boldsymbol{p}_2\|_2$$

Figure 3: Drone Control Kernel, showing the workings of the Lighthouse positioning system.The idea is to calculate the vectors from two base station to a sensor on the Lighthouse deck. This vector is defined by the intersection line between the two light planes of the base station and is sometimes referred to as a "beam", hence the name. In theory the beams should cross in the point where the sensor is located. In reality the point that is closest to both beams is used instead, and uses this as the estimated position. [16]

Components Evaluation

We have evaluated the components of the drone control kernel to ensure that they meet the requirements of the system. The results of the evaluation are presented in the following sections.

We expected the drones to follow a smooth path and found that the drones were able to follow the path with a high degree of accuracy. The drones were able to maintain their position and orientation while moving along the path, demonstrating the effectiveness of the control system.

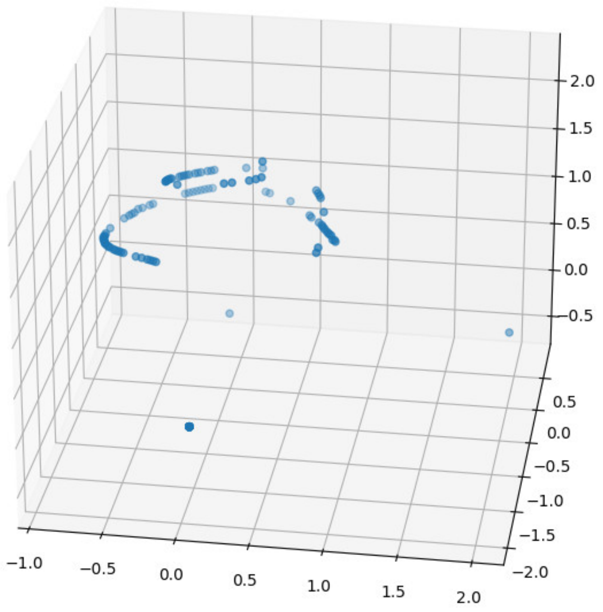


Figure 4: Drone Components evaluation.

Architecture

The drone control kernel is designed to provide a robust and flexible control system for managing drone swarms. The architecture of the drone control kernel is shown in Figure 1. The main components of the system are: Web API, which provides the web-based interface for controlling the drones; Socket, which allows for lower level client drone server interface with the drones; and the Drone Control Kernel, which manages the communication with the drones using Radio. The UDP server runs in parallel with the FastAPI application, thanks to $asyncio.create\_task()$. [17] This allows the UDP server to handle datagram messages asynchronously while the FastAPI server responds to HTTP requests.

Using FastAPI we are able to provide a basic web based interface to control the drones. The UDP Client is used to get positions of the drones without having to wait for the drones to respond or the client to send an Acknowledgment. It is used to lower the latency of the system and to stop the package drops errors given by the TCP server on load.

Each function in the UDP server is a coroutine, which allows for asynchronous execution. The UDP server listens for incoming messages from the drones and processes them accordingly. The FastAPI application

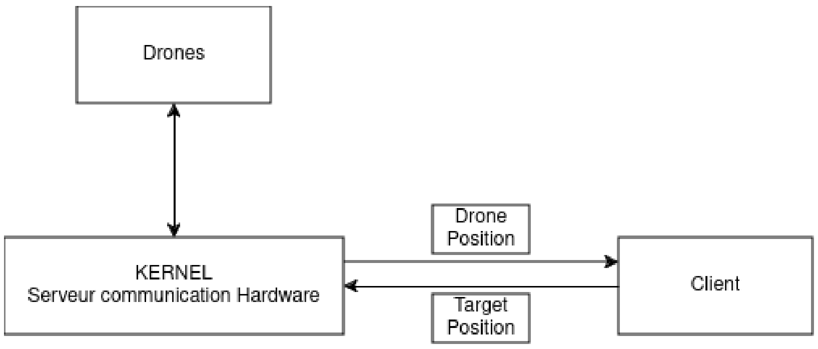


Figure 5: Drone Control Kernel, showing the main components of the system.

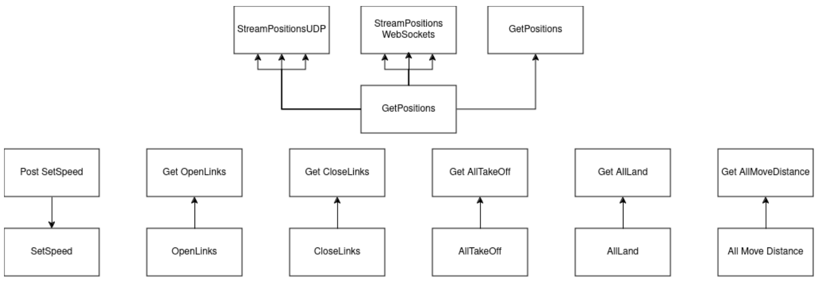


Figure 6: Drone Control Kernel, inside logics.

provides the web-based interface for controlling the drones. It exposes various endpoints for sending commands to the drones, such as takeoff, land, and move. The FastAPI application communicates with the UDP server to send commands to the drones and receive responses. The Socket component provides a lower-level interface for communicating with the drones using the Crazyflie Python library. It handles the connection to the drones and sends commands to them using the cflib API. The Drone Control Kernel manages the communication with the drones using the Radio. It sends commands to the drones and receives responses from them using the Radio component (Crazy Radio PA).

Also a ThreeJS interface is provided to visualize the drones in 3D space. This interface allows users to see the drones' positions and orientations in real-time, providing a visual representation of the swarm's behavior. The ThreeJS interface communicates with the FastAPI application to receive updates on the drones' positions and orientations.

3.3 Evaluation and Results

We have realized that the drone control kernel had a big flaw using TCP based technologies as we would observe drops in the communication. We have then switched to UDP based technologies and have seen a big improvement in the communication. We have also seen that the drones were able to follow a path with a high degree of accuracy. The drones were able to maintain their position and orientation while moving along the path, demonstrating the effectiveness of the control system. We werent able to break this server as we were able to send 1000 requests per second to the server and the server was able to handle them all as no function blocks the system. This was possible because the server was made as stateless as possible. So after setup no command could break it.

3.4 Conclusion

Using the drone control kernel, we have developed a robust and flexible control system for managing drone swarms. The system provides a web-based interface for controlling the drones and visualizing their positions and orientations in real-time. The system is designed to be scalable and extensible, allowing for the addition of new features and functionalities as needed. The system has been evaluated and shown to be effective in controlling drones in a swarm, with the drones able to follow a path with a high degree of accuracy using the Lighthouse positioning system. The system provides a valuable tool for developers and researchers working with drone swarms, enabling them to experiment with different control strategies and algorithms.

4 Contribution 2: Human pose estimation Kernel

4.1 Introduction

In this brief section we describe the workings of the human detection system. It was developed by Etienne Pommel for a similar use case. The system is based on the mediapipe library [18], which is a state-of-the-art object detection algorithm that is capable of detecting humans in real-time. [4]. The system is able to detect humans with high accuracy as they move around the environment. To the standard RGB image, the system adds depth.

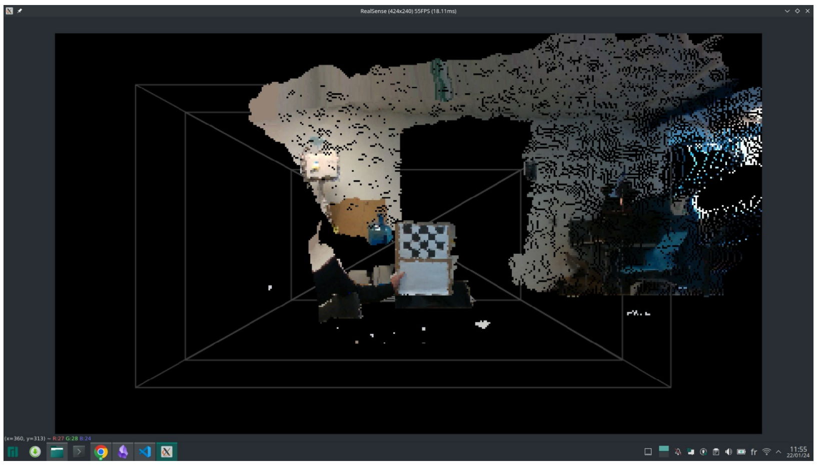


Figure 7: Intel Realsense D435, showing the workings of the depth camera and calibration capabilities, each pixel has been projected in the 3D space.

The difficulties of the system are to detect the objects in the image and to estimate the distance to the object. Finaly to project this distance in the 3D space.

Here we want to detect humans in the image this the use of mediapipe.

Architecture

The system was modified from the original algorithm as it would continuously calibrate using the chessboard. 1

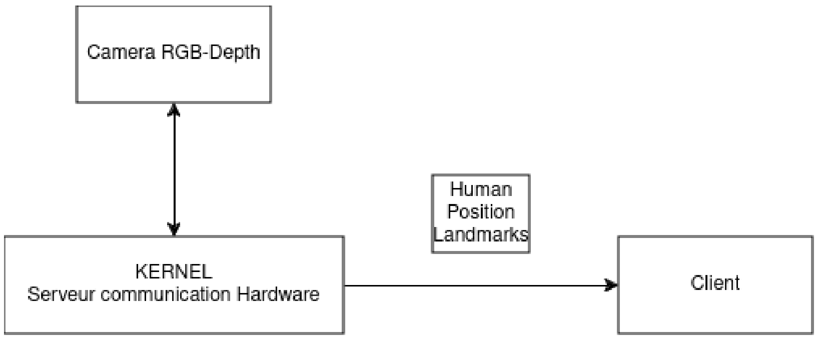


Figure 8: Object Control Kernel, showing the main components of the system.

4.2 Evaluation and Results

Also a ThreeJS interface is provided to visualize the drones in 3D space. This interface allows users to see the humans' positions and orientations

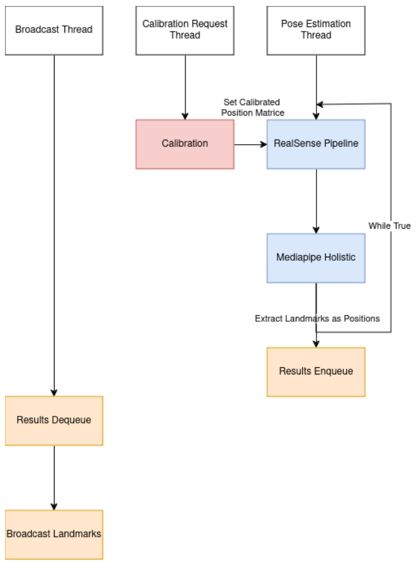


Figure 9: Object Control Kernel, inside logics.

in real-time, providing a visual representation of the system's behavior. The ThreeJS interface communicates with the FastAPI application to receive updates on the humans' positions and orientations.

We have evaluated the components of the human detection kernel to ensure that they meet the requirements of the system. The results of the evaluation are presented in the following sections.

We expected the measures to be accurate to at least 5% of the real distance but expect it to be less reliable the further the subject is to the camera. The system was not able to provide this much accuracy but 10% at 5 meters. The system though was able to provide a good estimate of the distance to the object, constant at 2% of the real distance at less then 7m. Indeed the results are less satisfying then expected but the system is still

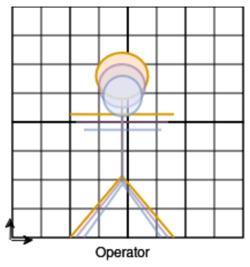


Figure 10: Object Control Kernel Issue, further the object is from the camera and the center of the image the smaller it will be estimated. The distance to the camera does not change only the size of the object in the image plane.

able to provide a good estimate of the distance to the object. The system is able to detect the object in the image and to estimate the distance to the object. The system is also able to project this distance in the 3D space, we will need to takee this into account in the next part of the project.

Similarly to the drones kernel, the system was made as stateless as possible. So after setup no command could break it. This system does not require a lot of resources and is able to send all the 30 landmarks per frame at the rate of 30 per second to the client so

900 landmarks per second.

4.3 Conclusion

The human pose estimation kernel provides a valuable system for detecting and estimating the distance of humans in real-time, using the Mediapipe library in combination with depth sensors. While the results showed a degree of inaccuracy beyond 5 meters, the system successfully achieved consistent performance within 2% accuracy for shorter distances (less than 7 meters). Despite some limitations—such as reduced reliability at greater distances and issues with object size estimation at image boundaries—the system still delivers an effective method for projecting human positions in 3D space. Future improvements should focus on enhancing the calibration mechanisms and addressing the accuracy drop at longer ranges, which will be vital for the next phase of the project.

5 Contribution 3: Platform : Human-Swarm Interaction

ADD IMAGE SHOWING Parallel access to services.

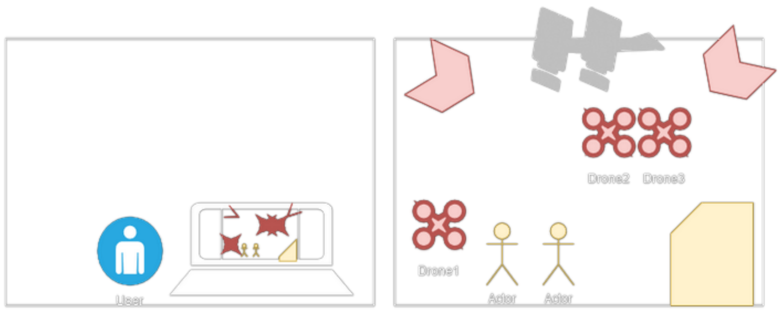


Figure 11: Our setup for controlling drones indoors around humans.

5.1 Introduction

In this section, we present our third contribution, which is the platform for human-swarm interaction. We present the setup and the software architecture that we have developed to control drones indoors around humans. We also present the results of the evaluation of the platform.

Our solution works on premise with a web-based interface to control the drones as it requires real-time computation. Both for drone control and human pose detection computation both have to be synchronized as both subjects are in the same environment. But the applications can be hosted on a cloud or other server allowing for remote control of the drones. This allows for a more flexible and scalable system.

5.2 Platform for Human-Swarm Interaction

Simulating drones in a virtual environment allows for the development of control algorithms without the need for physical drones. This is especially useful for testing new control strategies and algorithms in a safe and controlled environment. The platform provides a web-based interface for controlling the drones and visualizing their positions and orientations in real-time. The platform also provides a 3D visualization of the drones in the environment, allowing users to see the drones' positions and orientations in real-time. The platform is designed to be flexible and extensible, allowing for the addition of new features and functionalities as needed.

5.3 Opportunitites

As we have successfully built a usable platform with known people and subjects. We plan on testing it further and giving it to students to test further and develop on. Students can access the platform and develop their own algorithms to control the drones and have them interact with them in real-time. This will allow students to experiment with different control strategies and learn about the challenges of controlling drones in indoor environments. Especially without having to work on the hardware part of the drones.

One student Maximilien Menesguen [19] developed the famous boids [20] algorithm in Unity to make the drones follow a moving point in space using our UDP server, a work in progress. One of the main challenges in controlling drones indoors around humans is ensuring that the drones can avoid collisions with people and objects. The system continuously monitors the surroundings and adjusts the simulated drones' trajectories to avoid collisions. it is based on the original boids algorithm. [20]. The boids algorithm is a simple and elegant model for simulating the flocking behavior of birds. It is based on three simple rules: separation, alignment, and cohesion. The separation rule ensures that the drones maintain a safe distance from each other, the alignment rule ensures that the drones move in the same direction, and the cohesion rule ensures that the drones stay together as a group. By following these rules, the drones can navigate the environment safely and efficiently.

The alignment rule is modified to take into account the position of the human operator so the drones interact more with the human. Thus we plan on seting the red Ball of the UAVs to be the user. This modification allows the drones to follow the human operator while maintaining a safe distance from them. The cohesion rule is also modified to ensure that the drones stay within a predefined area around the human operators. This modification would allow the drones to move around the human operator without straying too far from them.

We plan on adding the repulsion rule directly to the human operator and the drones. This rule would ensure that the drones maintain a safe distance from the human operator at all times. Adding the security layer to the platform is also a priority.

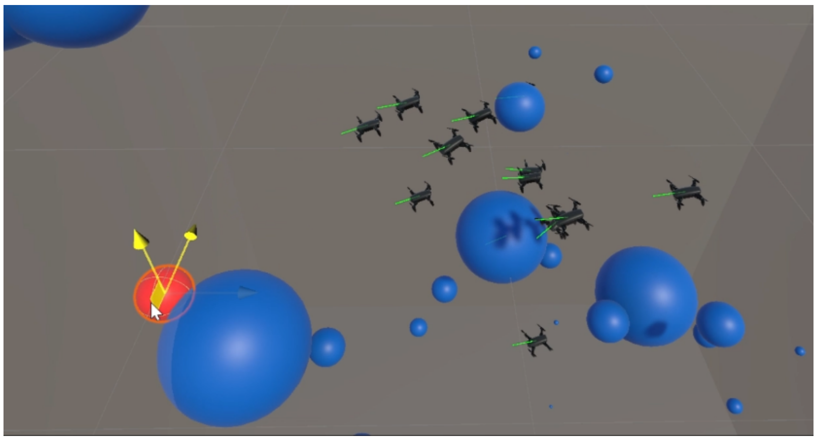


Figure 12: Drones following a moving point in space.

Conclusion

This technology gives possibilities in developpement to warehouses for adding drone carriers around sensible objects or animals.

Limits and Future Work

As discussed previously in the paper, the current platform has some limitations that need to be addressed in future work. One of the main limitations is the reliance on lighthouse technology [21] for positioning. While it uses sensors that are effective, they are light sensitive and may not be suitable for all indoor environments especially close to windows. Future work will focus on developing arount a more robust location system that can adapt to different environments and scenarios.

There is also a need to provide a cybersecurity layer to the platform to ensure that the drones are secure from external threats. This will involve developing a secure communication protocol to protect the drones from unauthorized access.

Finaly the platform needs to be tested in a real-world environment to evaluate its performance and reliability. This will involve conducting experiments in different indoor environments to assess the platform's intuitiveness against students. The results of these experiments will help to identify any issues with the platform and guide future development efforts.

1 Pose etimation pipeline.

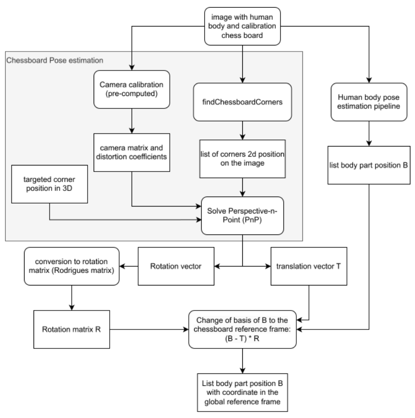


image with human body and calibration chess board
Chessboard Pose estimation
Camera calibration (pre-computed)
findChessboardCorners
Human body pose estimation pipeline
camera matrix and distortion coefficients
list of corners 2d position on the image
list body part position B
targeted corner position in 3D
Solve Perspective-n-Point (PnP)
conversion to rotation matrix (Rodrigues matrix)
Rotation vector
translation vector T
Rotation matrix R
Change of basis of B to the chessboard reference frame: (B - T) * R
List body part position B with coordinate in the global reference frame

Figure 13: Object Control Kernel, inside logics.

The system leverages pose estimation to detect human body landmarks using computer vision, specifically OpenCV's pose estimation pipeline. First, a camera captures RGB and depth data from the environment, with the depth data being crucial for calculating the distance of each detected point. The system uses a monocular camera setup, and it processes images through MediaPipe's holistic model, which detects human body landmarks, including key body joints like the head, shoulders, and limbs. These 2D landmarks are then projected into 3D space using depth data from the RealSense camera.

The system performs camera calibration to compute intrinsic parameters such as the camera matrix and distortion coefficients. Calibration ensures that the system accurately maps 2D pixel coordinates to real-world 3D points. OpenCV's solvePnP function computes the human's pose (rotation and translation vectors) relative to the camera using chessboard patterns, which serve as a known reference in the scene. After obtaining the pose, the Rodrigues rotation matrix is used to transform the body landmarks into the coordinate system of the chessboard, essentially re-aligning the 3D positions of the detected human body with the physical world.

This process ensures that the system can track human movement in 3D space accurately, which is crucial for human-drone interaction and control.

Acknowledgment

A Testbed Environment for Task Development was developed at the IFT. Marc Teyssier, Xiao Xiao and Clement Duhart showed much faith to enable such a project.

Etienne Pommel and Maximilien Menesguen of the Institute for future technologies, thank you for joining yourselves to me. Your enthusiasm and dedication to the project have been a great source of inspiration.

Thomas Carstens and Nicolas Stas of the Institute for future technologies have been a great help in the development of the platform. Your technical expertise and guidance have been invaluable.

References

Here are the references in citation order.

[1] Manuele Brambilla et al. ‘Swarm robotics: A review from the swarm engineering perspective’. In: Swarm Intelligence 7.1 (2013), pp. 1–41 (cited on pages 1, 2).

[2] J. A. Preiss et al. ‘CrazySwarm: A large nano-quadcopter swarm’. In: 2017 IEEE International Conference on Robotics and (2017) (cited on pages 2, 3, 5, 7).

[3] Bitcraze. CFLib - Python library for Crazyflie. https://www.bitcraze.io/documentation/repository/crazyflie-lib-python/master/. Accessed: 2024-10-10. 2020 (cited on pages 2, 3, 5).

[4] Mediapipe Dev Team. ‘Real-time pose estimation using Intel RealSense and MediaPipe for human-drone interaction’. In: (2020) (cited on pages 3, 6, 13).

[5] Chien Nguyen, Pascal Fua, et al. ‘FlyJacket: An upper body soft exoskeleton for immersive drone control’. In: IEEE International Conference on Robotics and Automation (ICRA). 2018 (cited on pages 3, 5–7).

[6] Maziar Shahbazi et al. ‘A review of human-drone interaction: Methods and applications’. In: Computers & Human Behavior 106 (2020), p. 106270 (cited on page 3).

[7] Jessica R. Cauchard and et al. ‘Drone & Me: An Exploration Into Natural Human-Drone Interaction’. In: Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (2015) (cited on page 4).

[8] X. Ju et al. ‘Safe human-drone interaction in indoor environments’. In: International Conference on Human-Robot Interac 2017 (cited on page 4).

[9] L. Cai and et al. ‘Human-drone interaction based on posture recognition with monocular vision’. In: Journal of Intelligent & Robotic Systems (2019) (cited on pages 5, 6).

[10] M. Rubenstein et al. ‘Kilobot: A low-cost scalable robot system for collective behaviors’. In: 2012 IEEE International Confer (2012) (cited on page 5).

[11] David Saldana et al. ‘ModQuad: The Flying Modular Structure that Self-Assembles in Midair’. en. In: 2018 IEEE International Conference on Robotics and Automation (ICRA). Brisbane, QLD: IEEE, May 2018, pp. 691–698. DOI: 10.1109/ICRA.2018.8461014. (Visited on 10/10/2024) (cited on pages 5, 7).

[12] J. L. Sanchez-Lopez et al. ‘CrazyChoir: Artistic performance with multiple Crazyflie drones’. In: IEEE International Conference on Robotics and Automation (ICRA). 2022 (cited on page 5).

[13] James A. Preiss et al. ‘Crazyswarm: A large nano-quadcopter swarm’. In: 2017 IEEE International Conference on Robotics IEEE. 2017, pp. 3299–3304 (cited on page 5).

[14] Thomas Carstens. ‘Intelligent Systems for Next Generation Drone Functionality’. en. In: () (cited on pages 6, 9).

[15] Amine Tayebi. ‘Gesture-based drone control with Leap Motion and Arduino’. In: International Journal of Advanced Rese (2020) (cited on page 6).

[16] Lighthouse positioning methods | Bitcraze. URL: https://www.bitcraze.io/documentation/repository/crazyflie-firmware/master/functional-areas/lighthouse/positioning_methods/ (visited on 10/15/2024) (cited on page 9).

[17] Python Software Foundation. ‘asyncio: Asynchronous I/O, event loop, coroutines and tasks’. In: (2022). Accessed: 2024-10-10 (cited on page 10).

[18] Google Research. MediaPipe: A Framework for Building Perception Pipelines. https://google.github.io/mediapipe/. Accessed: 2024-10-10. 2020 (cited on page 13).

[19] Maximilien Menesguen. Maximilien Menesguen. https://ift.devinci.fr/member/maximilien-menesguen. Accessed: 2024-10-10. 2023 (cited on page 17).

[20] Craig W. Reynolds. ‘Flocks, Herds, and Schools: A Distributed Behavioral Model’. In: Computer Graphics 21.4 (1987), pp. 25–34 (cited on page 17).

[21] DVIC - De Vinci Innovation Center. URL: https://dvic.devinci.fr/tutorial/lighthouse-positioning (visited on 09/07/2023) (cited on page 18).

Special Terms

Numbers

3D	Three Dimensional. 11, 13–16

A

API	Application Programming Interface (a set of tools for building software). 8, 10, 11

R

RGB	Red, Green, Blue (standard color model for digital images). 6, 13
RGB-D	Red, Green, Blue with Depth (imaging that captures depth along with standard color). 6
ROS	Robot Operating System (a flexible framework for writing robot software). 5, 6