SIMULATION SOFTWARE FOR AUTONOMOUS NAVIGATION OF UAVS IN GPS DENIED ENVIRONMENTS

Irene Kyriacou

This thesis

Submitted for Partial Fulfillment of

Requirements for the Acquisition

Master's degree

in Artificial Intelligence

in the

University of Cyprus

Recommended for Acceptance

from the Department of Computer Science

June 2024

Abstract

In recent years, the rapid evolution of drone technology has revolutionized various industries, ranging from search and rescue operations to disaster management. One of the key advantages of drones is their ability to access hard-to-reach areas quickly and efficiently, providing valuable data and support in critical situations. As technology continues to advance, the potential applications of drones in various fields are only expected to grow. Now, there is a strong dependency on Unmanned Aerial Vehicles (UAV) localization and mapping with the Global Positioning System (GPS), but it faces many challenges like signal loss, etc., so the navigation of the drone becomes limited. To overcome these challenges, researchers are exploring alternative methods, such as computer vision and artificial intelligence, for precise drone navigation. These advancements could potentially enhance the capabilities of drones in various applications, leading to more efficient and effective operations.

In the context of this thesis, our aim is to create a model that, based on computer vision techniques, allows the UAV to navigate autonomously in a city environment. By harnessing computer vision techniques, this model aims to enable drones to identify and navigate towards recognizable landmarks or features in their surroundings, offering a robust alternative to GPS-dependent navigation. Through the initial use of the AirSim simulator, it gives us the opportunity to fine-tune the algorithm and models to ensure optimal performance and efficiency. This helps to minimize potential risks and errors that could arise during the actual deployment of the drone in real-world drilling operations. Furthermore, the simulator allows for testing in various scenarios, ensuring the algorithm's adaptability and reliability. This comprehensive approach for testing will ultimately result in a more accurate and dependable autonomous navigation system for UAVs in urban settings.

APPROVAL PAGE

Master's Thesis

SIMULATION SOFTWARE FOR AUTONOMOUS NAVIGATION OF UAVS IN GPS DENIED ENVIRONMENTS

Presented by

Irene Kyriacou

Research Consultant

Dr. Chrysis Georgiou

Member of the Committee

Dr. Andreas Aristidou

Member of the Committee

Dr. Margarita Chli

University of Cyprus

June 2024

ACKNOWLEDGEMENTS

As I stand on the threshold of completing my master's journey, I am overwhelmed with gratitude for the support and guidance I have received along the way. This thesis represents not only the culmination of years of academic pursuit but also a testament to the unwavering support of many incredible individuals.

First and foremost, I would like to express my deepest appreciation to Dr. Harris Zacharatos and Dr. Chrysis Georgiou, for their invaluable mentorship throughout this entire process. Your wisdom, encouragement, and constructive feedback have been instrumental in shaping both the content and the quality of this thesis.

To my family and friends, whose unwavering support has been my anchor through both the highs and lows of this journey, I am deeply indebted. Your encouragement, patience, and understanding have been a constant source of strength, motivating me to persevere even in the face of challenges. It's now much clearer to me who are the people who provide their unconditional support no matter what, and I am profoundly grateful for your presence in my life.

Lastly, I want to acknowledge and thank all the individuals who may not be named here but have contributed in ways big and small to my academic and personal growth. Whether through a word of encouragement, or a helping hand, your presence has made a difference, and I am grateful for each and every one of you. Thank you for believing in me when I found it difficult to believe in myself.

As I embark on the next chapter of my life, I carry with me not only the knowledge and skills acquired during my studies but also the profound sense of gratitude for the incredible support network that has surrounded me. Thank you, from the bottom of my heart, for being a part of this journey.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1: Introduction

In this chapter, we will discuss the core idea of our thesis and what inspired us in the first place. Then we'll go over an overview of unmanned aerial vehicles, as well as some of the benefits and drawbacks associated with their use. Then we shall discuss our contributions to this particular field. Finally, the document's structure is offered, along with a brief explanation for each chapter.

1.1 Motivation

1.1.1 UAVs in Rescue Operations

In times of crisis, whether natural catastrophes like earthquakes and hurricanes or man-made emergencies like industrial accidents, one of the most difficult issues for rescue operations is the prompt and effective deployment of resources to impacted regions. These occurrences frequently occur in isolated or hazardous areas where typical modes of access are unfeasible or too slow. Furthermore, the safety of rescue professionals is critical, and placing them in dangerous situations might heighten the risk. Delays in reaching survivors might drastically reduce their chances of survival, making timely intervention critical. As a result, there is an urgent need for creative solutions to address these problems, enabling timely and successful rescue attempts while limiting the risk to human responders.

The use of autonomous drones into rescue operations represents a paradigm leap in disaster response capabilities. Unlike traditional approaches that rely on humans and ground-based vehicles, autonomous drones provide unrivaled agility, speed, and flexibility. These drones, which are equipped with powerful sensors, cameras, and communication systems, can quickly navigate over challenging terrain, including fallen structures, rocky landscapes, and dangerous settings like chemical spills or wildfires. Their ability to fly over barriers and into difficult-to-reach regions makes them helpful in search and rescue efforts. Furthermore, drones with autonomous navigation skills may operate without continual human supervision, allowing rescue personnel to focus on coordinating operations and conducting important duties on the ground [1][2].

Autonomous drones can evaluate large volumes of data in real time using artificial intelligence and machine learning algorithms, allowing them to detect signs of life, assess structure integrity, and identify risks. This real-time situational awareness provides rescue teams with actionable knowledge, allowing them to make educated decisions and use resources more efficiently. Furthermore, drones outfitted with payload delivery systems can carry critical supplies such as medical kits, food, water, or communication devices straight to survivors, bridging the gap until help comes. The importance of autonomous drones in rescue missions goes beyond their immediate effect on response efficiency. They also help in post-disaster evaluation and recovery by providing high-resolution aerial images and mapping information. This data helps with damage assessment, infrastructure appraisal, and recovery effort prioritization, resulting in faster restoration of vital services and community resilience.

1.1.2 Overview of Unmanned Aerial Vehicles

Unmanned Aerial Vehicles (UAVs), commonly known as drones, are aircrafts that can fly without a human pilot on board. They are controlled remotely by a ground-based operator or can fly autonomously using pre-programmed flight plans and onboard sensors and systems. UAVs come in various sizes and shapes, ranging from small quadcopters to large fixed-wing aircraft. They are powered by batteries or fuel and can carry different payloads, such as cameras, sensors, or even packages for delivery. There are so many diverse applications across various sectors due to their versatility and ability to access hard-to-reach areas. Some of the major uses of UAVs include, Remote sensing and monitoring, surveillance and security, delivery and transportation, aerial photography and videography as well as for search and rescue operations [3]. The last use case of the UAVs is also one of the most important, and this was the main source of inspiration for the deployment of my model.

Autonomous navigation of drones signifies the capability of unmanned aerial vehicles (UAVs) to independently navigate and accomplish tasks without human intervention. This advanced capability encompasses various critical elements. Some of these elements include obstacle avoidance, path planning, and decision-making based on real-time data. Autonomous navigation allows drones to operate efficiently in complex environments and execute missions with precision and accuracy.

Firstly, drones utilize onboard sensors such as cameras, LiDAR, and radar to perceive their surroundings and construct a detailed map or model of the environment [4][5]. Secondly, they determine their precise position and orientation through methods like GPS, visual odometry, or simultaneous localization and mapping (SLAM) [5][6][7]. Subsequently, algorithms compute an optimal collision-free path from the drone's current location to the designated goal, factoring in obstacles and constraints [4][5]. Once the path is planned, the drone autonomously executes it using its actuators while adjusting for any disturbances encountered along the way. However, achieving seamless autonomous navigation poses several challenges, including ensuring robust perception, attaining accurate localization, dynamically re-planning paths in real-time amidst changing environments, optimizing energy consumption, and effectively dealing with environmental uncertainties like wind. Despite these hurdles, autonomous navigation empowers drones to undertake intricate missions such as delivery, inspection, and search and rescue operations without the need for remote piloting, showcasing their potential for diverse applications in various fields. It also opens up new possibilities for the future of autonomous technology.[6], [8] With advancements in artificial intelligence and sensor technology, the capabilities of autonomous drones continue to expand, paving the way for more sophisticated and efficient operations. As these technologies evolve, we can expect even greater autonomy and versatility in drone navigation, revolutionizing industries and services across the board.

1.1.3 Advantages of UAVs Usage

Drones have become increasingly integral to rescue operations, offering a range of benefits that enhance the efficiency and safety of these critical missions. The use of drones in search and rescue (SAR) operations has revolutionized the approach to emergency response by providing rapid, reliable, and versatile solutions in various challenging environments. UAVs are deployed in rescue operations to perform tasks such as searching for missing persons, delivering emergency supplies, and assessing disaster areas [9]. These UAVs are equipped with high-resolution cameras, thermal imaging, and other sensors that allow them to capture detailed images and data from above. This capability is crucial in extensive and inaccessible terrains where traditional methods would be less effective and more time-consuming.

There are several advantages of UAVs over traditional rescue methods. One of the most significant advantages of drones is their ability to be quickly deployed. Drones can be airborne within minutes and cover large areas much faster than ground teams or manned aircraft. This rapid response is crucial in rescue operations where time is often a critical factor in saving lives [10]. Secondly, drones are more cost-effective compared to traditional manned aircraft. They require less fuel and maintenance and can be operated with fewer personnel [11]. This cost efficiency allows rescue organizations to manage their resources better and conduct more frequent operations without the financial burden associated with manned aircraft. Also, using drones minimizes the risk to human life as they can enter hazardous environments that are unsafe for human teams. This is particularly important in situations like chemical spills, fires, or natural disasters where the conditions may be too dangerous for direct human intervention [12].

Another advantage in the use of UAVs in rescue operations is the high-quality acquisition data. Drones can capture high-resolution images and other data types, which are invaluable for assessing damage and planning rescue operations. This data can be used to create detailed maps and models, helping decision-makers to better understand the situation and plan appropriate responses [10][13]. Additionally, drones are highly versatile and can be equipped with various sensors to suit different mission needs. They can operate in diverse weather conditions and at different times of the day or night, thanks to their robust design and advanced technology. This flexibility ensures that drones can be adapted to a wide range of emergency scenarios. Equally important advantage of the usage of the UAVs is the accessibility they provide. Drones can reach areas that are otherwise inaccessible or difficult for human teams to navigate, such as mountainous terrains, dense forests, or disaster-stricken urban areas. This capability ensures that no area is beyond the reach of rescue operations, thus increasing the chances of successful rescues [13].

1.1.4 Disadvantages/Weaknesses of UAVs Usage

Despite their advantages, several disadvantages impact their effectiveness in such critical missions. Key issues include GPS signal loss, privacy concerns, limited payload capacity, and the impact of weather conditions. One significant disadvantage of using UAVs in rescue operations is the potential loss of GPS signals, which is crucial for navigation and operational efficiency. UAVs rely heavily on GPS for location tracking and navigation to the target sites. Loss of GPS signal can occur due to several reasons, including signal jamming, which can be intentional (as a security threat) or due to environmental factors that disrupt signal transmission. This loss can lead to UAVs deviating from their intended paths or failing to complete their missions, which is particularly critical in time-sensitive rescue operations [11]. Privacy issues are another critical concern associated with the use of UAVs in rescue operations. Drones equipped with cameras and other sensors can collect vast amounts of data, some of which may be sensitive or personal. This raises concerns about the unauthorized use or potential misuse of this data, leading to privacy violations. Ensuring that data collected during UAV operations is handled securely and in compliance with privacy laws and regulations is a significant challenge [14].

Another disadvantage is that UAVs are also limited by their payload capacity, which restricts the amount of equipment, medical supplies, or other essentials they can carry to disaster-stricken areas or individuals in need. This limitation affects the effectiveness of UAVs in operations where large amounts of supplies or specialized equipment are necessary for rescue or relief efforts [15]. Finaly, weather conditions significantly impact UAV operations. Adverse weather, such as high winds, rain, or fog, can hinder the UAV's ability to fly, navigate accurately, or maintain communication with the control station. Such conditions can lead to reduced operational efficiency or complete mission failures, which are detrimental in emergency and rescue scenarios where time is of the essence [15].

In conclusion, while UAVs offer substantial benefits for search and rescue operations, their effectiveness can be compromised by several factors, including GPS signal loss, privacy issues, limited payload capacity, and susceptibility to adverse weather conditions. Addressing these challenges is crucial for enhancing the reliability and efficiency of UAVs in critical missions.

1.2 Objective and Contribution

Focusing on the first problem described above (GPS signal loss), we aim to deploy a model for autonomous navigation that does not base its operation on the signal of the GPS. Building an Artificial Intelligence model that is robust to the signal loss helps to safely return the drone to its starting point without it becoming disoriented and crashing in an unknown location.

To this respect, the main objective for this master's thesis was to develop an algorithm that would enable a drone to independently return to its initial location. This is necessary when there is no GPS signal available, making it impossible for the drone to determine its position and make a safe return.

This constitutes the main contribution of this thesis, the development of such a navigation algorithm. The algorithm optimizes the route for the UAV's return autonomously and without the need for human intervention by combining the IMU data, which is already present in the UAV, with computer vision technologies such as *feature matching*: it compares the current image obtained from the UAV’s camera with the features from previous images that were obtained from the UAV’s path. To provide subpar landmark-based navigation, we also include additional methods, such as object identification. Furthermore, we incorporate the trilateration approach into our algorithm, which may compute an unknown position based on other known locations and their respective distances.

1.3 Document Organization

The rest of the document is organized as follows.

In Chapter 2 we examine the most recent approaches being used by academic institutions to enable drone (UAV) autonomy in navigation. Increasing our expertise and being inspired by other experts allows us to explore new areas and enhance our studies. The various simulators are also included in this chapter, along with a brief explanation of each, so that we may compare them and make the best decision to meet our needs.

In Chapter 3 we present the methodology we followed in order to fulfil the objectives of the thesis. Here we also present the navigation algorithm along with two main investigation scenarios.

In Chapter 4 we present the results of the algorithm's evaluation. Here, we list not only our model's successes, but also its shortcomings, leading to the discussion on Future Research directions in Chapter 5.

Chapter 2 : Background

This chapter consists of a detailed literature review that focuses on cutting-edge algorithms that are critical for Unmanned Aerial Vehicle (UAV) operations. We thoroughly investigate cutting-edge approaches for autonomous navigation and investigate suitable simulators for modeling and evaluating UAVs' performance.

2.1 Literature Review

Current research in autonomous navigation is focused on improving existing methods and developing new models. The innovative ideas and approaches proposed in recent literature offer researchers opportunities to expand their knowledge and inspire the creation of new models. These articles provide a comprehensive overview of the latest advancements in autonomous navigation, highlighting the potential for interdisciplinary collaboration in this field. Building on these studies, we have developed models for autonomous navigation that integrate data from UAV inertial sensors and operator vision through drone cameras.

Two methods that the scientific community is particularly interested in using and researching further are *SLAM* and *Deep Reinforcement Learning*. Robots and autonomous vehicles employ a technology called Simultaneous Localization and Mapping, or SLAM, to map an unfamiliar area while also tracking their own location inside it. The two main procedures involved are localization and mapping. The process of mapping involves the robot making a map of its environment by utilizing sensors to identify objects and characteristics, such as cameras, lasers, or sonar. The process of localization is how the robot establishes its location on the map it is making. As the robot moves, it continuously updates its position based on the map and sensor readings. This is essentially what SLAM does using advanced technology, making it crucial for robots and self-driving cars to navigate new and changing environments without relying on pre-existing maps. SLAM technology is essential for autonomous vehicles to navigate in real-time, as it allows them to build maps on the go and avoid obstacles. This capability enables UAVs to operate efficiently and safely in dynamic environments where pre-existing maps may not be accurate or available.

Deep Reinforcement Learning (DRL) is an advanced technique combining deep learning and reinforcement learning, and it is particularly effective for the autonomous navigation of unmanned aerial vehicles (UAVs). Through interaction with its surroundings and feedback in the form of rewards or penalties, a UAV in DRL learns to navigate. The UAV seeks to maximize its cumulative reward over time by approximating the optimal course of action in different scenarios using a neural network. By means of trial and error, the unmanned aerial vehicle (UAV) refines its navigation approach and gains the capacity to perform intricate tasks like obstacle avoidance, path planning, and stability maintenance in dynamic settings.

For UAVs, DRL involves training the neural network on vast amounts of simulated flight data before deploying it in real-world scenarios. The UAV can acquire strong navigational skills even in erratic and variable situations thanks to this training. The neural network makes judgments regarding speed, direction, and altitude modifications in real-time by analyzing inputs from the UAV's sensors, including cameras, lidar, and GPS. DRL is a useful tool for applications like delivery services, search and rescue, and surveillance because it allows unmanned aerial vehicles (UAVs) to execute tasks autonomously with great precision and efficiency through continuous learning and adaptation.

Upon introducing the fundamental theoretical framework, we delve deeper and examine increasingly sophisticated models utilized in the academic and research domains.

In the article “Deep reinforcement learning-aided autonomous navigation with landmark generators” [16], Xuanzhi Wang and her team introduce the deep reinforcement learning-aided autonomous navigation that integrates neural network-based landmark generators.

The methodology for deep reinforcement learning-aided autonomous navigation involves the integration of landmark generators to facilitate the learning process. These generators, implemented as neural networks, extract pertinent features from the agent's observations, acting as an abstract representation of the environment to enhance the agent's learning efficiency.

These extracted landmark features are then utilized alongside the agent's observations as input for a Deep Q-Network (DQN), which generates Q-values for each potential action.

In addition to the landmark generators, reward shaping is employed to provide the agent with more informative feedback during training. This shaping of the reward function guides the agent towards states more conducive to achieving the goal, thereby expediting the learning process. Furthermore, curriculum learning is implemented, gradually increasing the complexity of training environments to enable the agent to initially learn in simpler settings before transferring this knowledge to more intricate scenarios.

The methodology is validated through experiments conducted in both simulated and real-world environments. Initially trained in a simulated setting, the model is subsequently fine-tuned on a physical robot in the real world. This simulation-to-reality transfer strategy reduces the training required on the physical robot and demonstrates the efficacy of the methodology in enabling successful navigation in real-world scenarios.

An innovative deep reinforcement learning approach for UAV autonomous navigation and collision avoidance is presented by Fei Wang [17]. It makes use of important elements including reward shaping, a Deep Q-Network (DQN), and landmark generators. By using neural networks as landmark generators, important details are taken from the UAV's views and transformed into an abstract representation of the surroundings, which improves learning effectiveness. These ground-breaking characteristics greatly enhance the UAV's navigational capabilities in novel and dynamic surroundings.

A DQN is used as the main reinforcement learning algorithm in the technique. The DQN creates Q-values for possible actions based on inputs from the UAV's observations and the extracted landmark characteristics, allowing the UAV to navigate and avoid collisions with more knowledge. In addition, reward shaping is used, which makes use of the landmark features to give precise training feedback. By directing the UAV in the direction of states that facilitate effective navigation and collision avoidance, this procedure speeds up learning and enhances performance in unfamiliar situations. The after steps of this research is to deploy the model in real-life drones, and it will be the extension of this research that will include also the speed of the wind. Also, it will be needed to extend this research for collision avoidance in air.

Without depending on LiDAR, Sebastian Barbas Laina, Simon Boche, and Sotiris Papatheodorou offer an approach for scalable autonomous drone flight in forest environments utilizing dense submaps and Visual-Inertial Simultaneous Localization and Mapping (VI-SLAM) [18]. The core of this method is an autonomous Micro Aerial Vehicle (MAV) that navigates beneath forest canopy using lightweight, inexpensive passive optical and inertial sensors. To provide a scalable mapping framework, the MAV uses VI-SLAM to offer precise state estimations that are combined with a volumetric occupancy submapping system.

These submaps, in contrast to monolithic maps, automatically control drift and corrections from the VI-SLAM by adjusting based on updated posture estimations. The authors present a unique reference trajectory anchoring mechanism to ensure the MAV's safety. Based on state updates from the VI-SLAM system, this technique modifies and deforms the reference trajectory that the MAV follows, even considering notable alterations brought about by loop closures.

Extensive trials in actual and simulated thick forest environments, with tree densities reaching 400 trees per hectare and speeds up to 3 m/s, were conducted to validate the technique and achieve navigation free from collisions and system failures. The authors claim that this is the first system to use entirely onboard computer, including VI-SLAM, along with inexpensive passive visual sensors to achieve such performance in unstructured situations.

Similarly, Ganesh Sapkota and Sanjay Madria in their paper “Landmark-based Localization using Stereo Vision and Deep Learning in GPS-Denied Battlefield Environment” use deep learning and stereo vision methods to solve the problem of localization in combat situations where GPS is unavailable [19]. Their approach makes use of the YOLOv8s model, which was trained using a real-world dataset, for accurate landmark detection and specially calibrated stereo vision cameras for accurate distance calculation. This dual technique enables robust and consistent localization in dynamic and unexpected warfare conditions by utilizing deep learning for landmark identification and stereo vision for depth perception. Through the combination of these cutting-edge technologies, Sapkota and Madria have shown how to overcome the constraints of GPS-denied situations, potentially improving localization accuracy and dependability in crucial military applications.

To sum up, the techniques that have been discussed demonstrate the noteworthy progress made in autonomous navigation and localization by combining many state-of-the-art technologies. The application of deep reinforcement learning and visual sensing technologies in various demanding scenarios is demonstrated by these methods, which range from using landmark generators and Deep Q-Networks in UAV navigation to using VI-SLAM for drone flights in forests and integrating stereo vision and deep learning in combat zones.

Lastly, it is worth mentioning that we have looked into the bibliography in order to find research papers that follow the same, or at least a similar methodology in order to extract their results. When it comes to the methodology that will follow in the first scenario (will be described in a following chapter) the navigation based entirely on IMU data is not possible in real life demonstration due to the noise in the measurements and accelerometers and gyroscope drifts [20]. That’s why further techniques were implemented as described in [20] by Abdullah Mohammed Bahasan.

2.2 Potential Simulators

To create our model that bases its operation in computer vision, and not in GPS signal the use of a drone simulator is crucial. Drone simulators offer a controlled environment where developers can fine-tune algorithms and test various scenarios without the constraints and risks associated with real-world deployment [21]. One significant advantage is the ability to iterate rapidly. Since simulations can be run quickly and repeatedly, developers can efficiently tweak parameters and algorithms to improve performance [22]. This iterative process accelerates the development cycle, allowing for faster innovation and refinement.

Moreover, simulators provide a safe space to explore edge cases and failure modes that may be impractical or dangerous to replicate in real life. This capability is particularly crucial when developing computer vision-based models, as it enables thorough testing across a wide range of conditions, lighting environments, and object appearances [23]. By exposing the algorithm to diverse scenarios in the simulated world, developers can enhance its robustness and reliability, ultimately leading to better performance when deployed in the real world. Another advantage of drone simulators is their scalability. Developers can simulate multiple drones operating simultaneously, allowing for complex swarm behavior testing and optimization. This scalability facilitates the development of algorithms that are not only effective for individual drones but also capable of coordinating and collaborating with other drones seamlessly. Additionally, drone simulators can significantly reduce costs associated with hardware experimentation and maintenance [24]. Instead of relying on expensive physical drones and risking potential damage during testing, developers can utilize virtual drones within the simulator environment at minimal cost. This cost-effective approach enables experimentation at scale, opening up opportunities for smaller teams and startups to innovate in the drone technology space [25].

Overall, the use of a drone simulator provides a comprehensive and efficient platform for developing, testing, and refining computer vision-based algorithms for drone operation. By leveraging the advantages of simulation, developers can accelerate progress, mitigate risks, and ultimately deliver more capable and reliable drone systems to the market.

We proceed to overview the main simulators we have considered. Selecting the most appropriate simulator that would satisfy our needs was a very important task. To this respect, we set specific evaluation criteria, leading to an informed decision. Here is the list of simulators we have considered:

- Air-Sim
- Ros + Gazeebo
- MAVProxy
- SwarmSim
- RotorS

Some factors to consider when evaluating simulators include ease of use, scalability, and compatibility with existing systems. The criteria that we will be using to evaluate each simulator include available documentation, be open source, have a commercial license, the potential to fly the drone programmatically, and the potential of UAV swarms. By carefully weighing these criteria, we were able to choose the simulator that best meets the needs of the project.

2.2.1 AirSim Simulator

The AirSim by Microsoft (see Figure 1) was the best choice for our project because it offered comprehensive documentation, a commercial license for use (MIT License), the ability to program drone flights, and support for UAV swarms. Another important factor to consider when evaluating simulators is the level of realism they offer. AirSim had a more realistic physics and environmental factors because its build on Unreal Engine (discussed below), that can enhance training and testing scenarios for drones. We now provide some more details on AirSim.

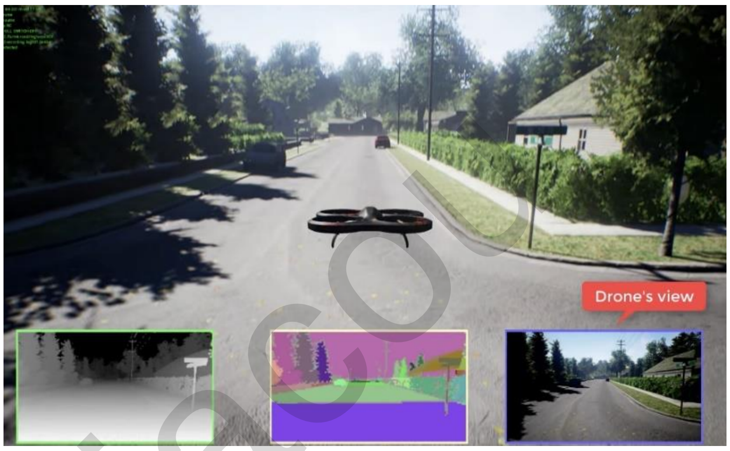


Figure 1:The AirSim simulator

AirSim, is an open-source simulator tailored for drones, cars, and other vehicles, serving as an instrumental tool for training autonomous systems and conducting research in robotics, artificial intelligence, and computer vision. AirSim provides a realistic environment for testing and developing algorithms, with features such as high-fidelity physics simulation, customizable environments, and integration with popular machine learning frameworks. Researchers and developers can leverage AirSim to accelerate their projects and innovations in the field of autonomous systems. At its core lies a realistic simulation environment, meticulously crafted with detailed 3D urban landscapes and physics-based simulations, all powered by the renowned Unreal Engine. This foundation ensures visually captivating and physically accurate virtual worlds where developers can explore and refine their algorithms. AirSim boasts support for a diverse array of vehicles, including multirotor drones, fixed-wing aircraft, ground vehicles like cars and trucks, and even boats, enabling researchers to test autonomous algorithms across various platforms. Another standout feature is its comprehensive sensor simulation, encompassing cameras, LiDAR, radar, and GPS, all replicating realistic noise, distortion, and environmental effects crucial for training perception algorithms under diverse conditions. Users can tailor environments to their needs, either by crafting custom scenarios or importing real-world maps, further enhancing the simulation's applicability and realism. With a robust API supporting languages like C++, Python, and C#, AirSim facilitates seamless integration and control, enabling developers to experiment with autonomy algorithms effortlessly. Moreover, its integration with machine learning frameworks like TensorFlow and PyTorch empowers researchers to train and deploy models directly within the simulation, facilitating the development of AI-driven control systems. As an open-source project, AirSim fosters collaboration and community support, with a dedicated community actively contributing to its development and providing assistance through forums and online channels. Its cross-platform compatibility ensures accessibility across Windows, Linux, and macOS, making it an indispensable tool for researchers striving to advance the field of autonomous systems within a safe and realistic virtual environment.

2.2.2 The ROS Software Libraries and Tools

ROS (Robot Operating System) is a comprehensive set of software libraries and tools widely used in robotics development (see Figure 2). It is available under the Apache 2.0 License, making it suitable for commercial applications. ROS integrates with various software frameworks like PX4 for drone flight control and OpenCV for computer vision and machine learning tasks. Additionally, there's ample support available through YouTube tutorials and a vibrant community, making it accessible for developers of all levels.

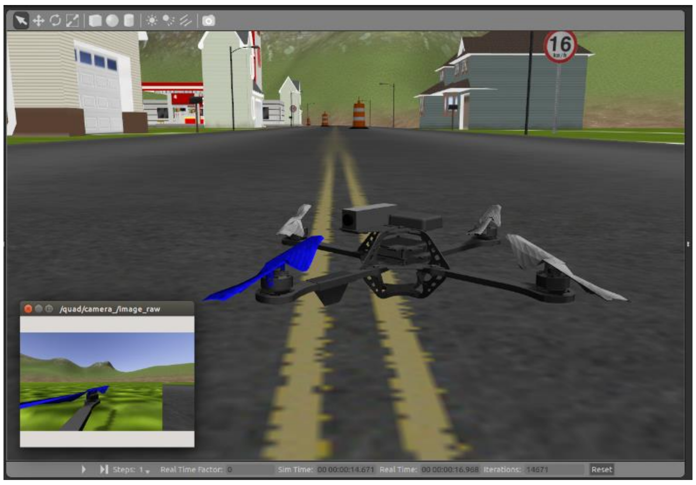
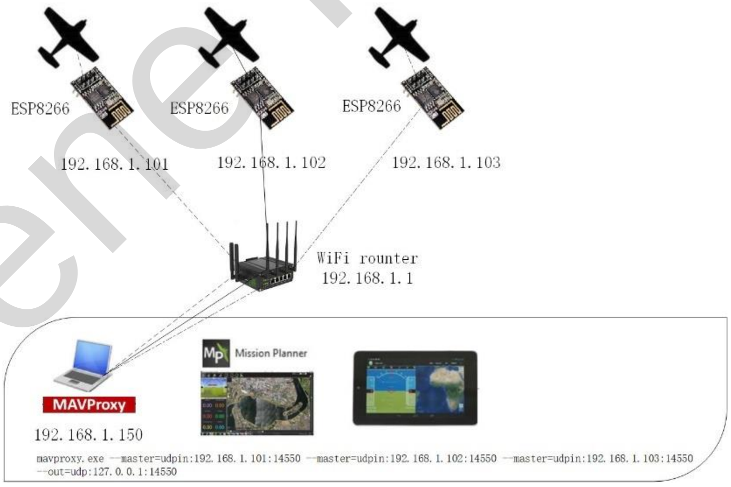


Figure 2:The ROS + Gazeebo environment

Despite the above-mentioned advantages of ROS it was not the one selected because at the time AirSim seemed to be the right and more promising choice.

2.2.3 The MAVProxy Simulator

MAVProxy serves as ground station software for UAVs, facilitating communication and control between ground stations and unmanned aerial vehicles. It is licensed under the GNU General Public License v3, allowing for commercial use. MAVProxy's functionality can be extended through add-on modules, including intriguing features like the "Follow Me Test". It supports swarms of UAVs, typically ranging from 2 to 10 vehicles, and offers post-flight analysis capabilities (see Figure 3). However, despite its features, there are concerns that the



```
mavproxy.exe --master=udpin:192.168.1.101:14550 --master=udpin:192.168.1.102:14550 --master=udpin:192.168.1.103:14550
--out=udp:127.0.0.1:14550
```

Figure 3:MAVProxy, a ground station software

available information may not adequately address all requirements or demands of specific problem

2.2.4 The SwarmSim Simulator Platform

SwarmSim is a simulation platform utilizing the Unreal Engine, although its licensing information couldn't be determined. It offers a Python API with PyTorch integration, empowering developers to implement algorithms effectively. The platform supports multi-agent scenarios, facilitating the simulation of interactions between numerous agents. Additionally, users have the capability to create custom environments within the platform, enabling tailored experimentation. However, it's worth noting that no demo was available to test whether SwarmSim meets specific needs.

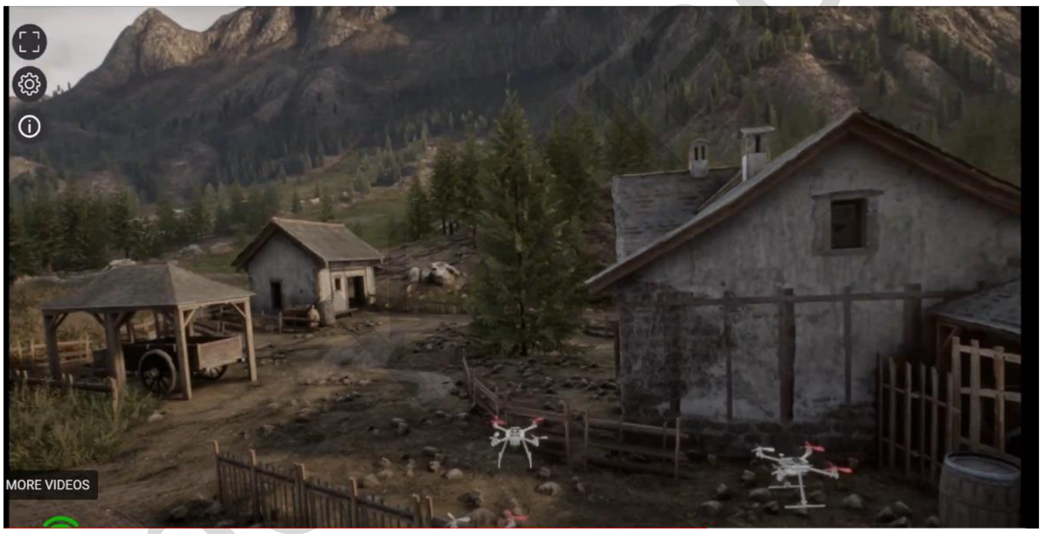


Figure 4: The swarm of UAVs inside the SwarmSim environment.

2.2.5 The RotorS Drone Simulator

RotorS (see Figure 5), developed by the Autonomous Robots Lab at ETH Zurich, is a drone simulation framework primarily tailored for rotorcraft research and development. It is built upon the Gazebo simulator and integrates with ROS (Robot Operating System) for its functionalities. Released under the Apache 2.0 License, RotorS is suitable for commercial use. Even though it seemed promising taking into consideration the team that created it, there's limited detailed information available.

Figure 5:RotorS Simulator

The Unreal Game Engine.

The Unreal Engine [26] is a powerful and widely used game engine developed by Epic Games. It is renowned for its cutting-edge graphics, advanced physics simulation, and robust development tools, making it a popular choice for game developers, architects, filmmakers, and increasingly, researchers and engineers in fields such as robotics and autonomous systems. The Unreal Engine's versatility and user-friendly interface have contributed to its widespread adoption across various industries. Its ability to create immersive virtual environments and realistic simulations has revolutionized the way professionals in different fields approach their work.

At its core, the Unreal Engine provides a highly customizable and visually stunning environment for creating interactive experiences. Its real-time rendering capabilities allow developers to achieve lifelike visuals, dynamic lighting, and immersive environments, essential for creating realistic simulations in AirSim. Furthermore, the Unreal Engine's robust toolset enables developers to easily iterate and refine their projects, saving time and resources in the development process. Its seamless integration with various hardware devices and platforms makes it a versatile choice for professionals looking to create cutting-edge virtual experiences. Furthermore, the Unreal Engine's extensive suite of tools enables developers to design complex virtual worlds with ease, from intricate cityscapes to vast outdoor landscapes, providing a rich and diverse backdrop for testing autonomous algorithms.

In the context of AirSim, the Unreal Engine serves as the underlying framework for rendering the simulation environment and simulating physics-based interactions. It leverages Unreal's advanced rendering pipeline to create visually compelling landscapes, complete with realistic lighting, materials, and effects. Additionally, Unreal's physics simulation engine enables accurate modeling of vehicle dynamics, environmental interactions, and sensor behaviors, crucial for creating a realistic and immersive simulation experience [26].

Moreover, the Unreal Engine's flexibility and extensibility make it an ideal platform for integrating custom features and extensions, such as AirSim's support for various vehicles and sensors. AirSim leverages Unreal's scripting capabilities to implement vehicle dynamics, sensor models, and other simulation components, allowing developers to customize and extend the simulation to suit their specific research needs.

Overall, the Unreal Engine's rich feature set, powerful rendering capabilities, and flexibility make it a perfect match for AirSim, providing a solid foundation for creating realistic and immersive simulation environments for training autonomous systems and conducting research in robotics and artificial intelligence.

Chapter 3: Methodology

This chapter describes the methods used in our research, including the approach and experimental framework used to achieve our objectives. This chapter demonstrates how to create a realistic simulation environment with the Unreal Engine, providing a dynamic and immersive testing platform. It also describes the step-by-step process for the two separate scenarios we explored, guaranteeing a clear path from initial setup to data collection and analysis.

3.1 Environment Creation

After the decision about the most compatible simulation software, we begun the implementation process. We needed to download and install the Unreal Engine, which, as already mentioned, is popular for the creation of video games. Then we proceed with the build and customization of the simulation environment.

First, all the necessary configuration needed to be made to connect the AirSim plug-in to the Unreal Engine and be able to simulate the drone in a customizable environment. After that, we tested the different functionalities and interactions to ensure a smooth and realistic simulation. To accurately represent the physical environment where the drone will be operating, the implementation of the 3D word was necessary. This was achieved with the "Photorealistic 3D Tiles from Google Maps Platform in Cesium for Unreal" (see Figure 6). This allowed for a more realistic and accurate representation of the drone's operating environment, enhancing the overall simulation experience.

For the needs of this project the Unreal Engine version 5.2 was used, that was compatible with the AirSim as well as with the Cesium Photorealistic 3D Tiles. Additionally, fine-tuning of the environmental variables (like the sun position - lighting of the terrain) was crucial to

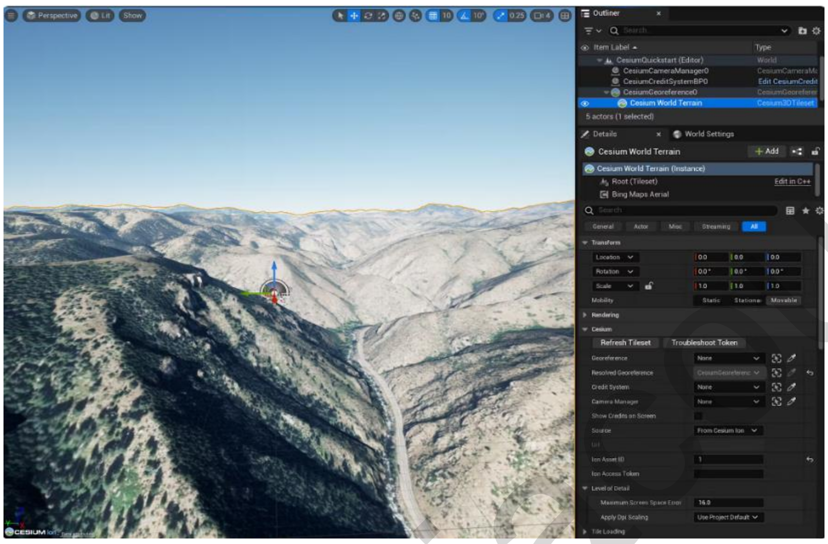


Figure 6-Cesium Photorealistic 3D Tiles as it appears in the Unreal Engine

ensure that the simulation accurately reflected real-world conditions.

The addition of the photorealisitc 3D tiles gave us the opportunity to place the drone in the scenery of our choice, including big cities like California and mountains. The main idea was to deploy the drone in Nicosia city. With that we were going to be able to also test the final project in real-life conditions to validate the accuracy of the model. Unfortunately, that wasn't possible due to the fact that for this specific geographical area, the buildings weren't 3D as they were not been mapped yet. This gives us the opportunity for future work and research to import new environments that are the 3D maps of Cyprus like the iNicosia. The iNicosia project [27] is a digital twin of the Nicosia city that includes a 3D model of the city representing real buildings and monuments (such as the walls and the old city), the road network, parking areas, etc.

Therefore, we placed the starting point of the UAV in Redmond, Washington, USA (47°38'37.2"N 122°08'24.5"W). This location provided a suitable alternative for testing the drone's capabilities and ensuring that the model was functioning correctly. The scenery includes a variety of objects like houses, rivers, and lakes, and this variety allowed us to simulate different scenarios and challenges for the drone to navigate through, ultimately improving its performance and reliability. Despite the change in scenery, we were still able to gather valuable data and insights for the project.

After the deployment of the drone inside the Unreal environment, the UAV had to be controlled through our Python Script. So, we proceed with the creation of a function inside the main Python script, in which the drone could be controlled, for its main movements through the keyboard. After the creation of the manual control of the drone, an important step for the development of the project was the familiarization with all the available AirSim functions.

First, it was very important to learn how to navigate and control the drone smoothly before we proceeded with the development of the main algorithm that aims the autonomous navigation of the drone. For that reason, we proceed in the creation of a testing environment, that can be built through the basic configuration of the AirSIm. This testing environment include several geometrical shapes, that were predefined in the AirSim pluggin. A relevant picture (Figure 7) can be found below.

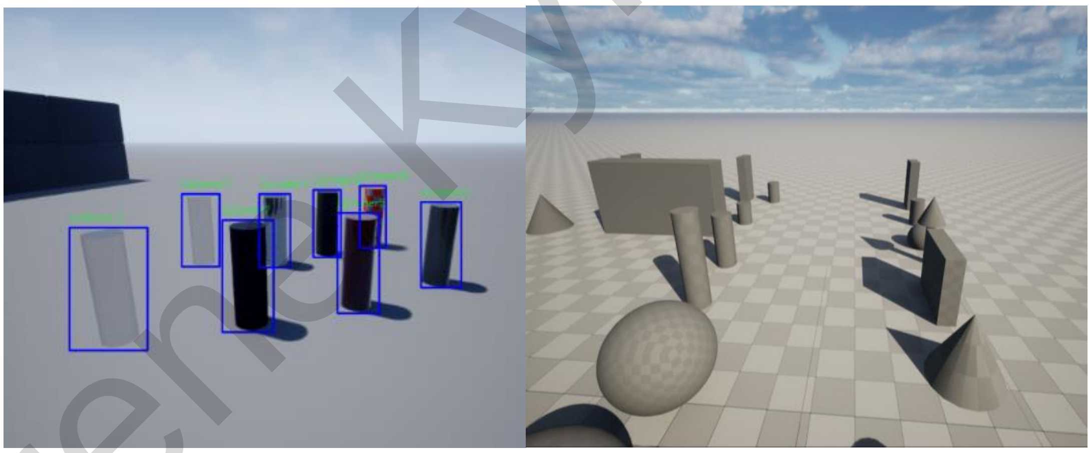


Figure 7: The testing environment with the Cylinders

At this point, the main idea of the detection algorithm was beginning to build. The first step of the algorithm was the detection of the objects that the drone came to be able to capture within its field of view, and while been manual control. An extensive description of the algorithm is given next.

3.2 Navigation Algorithm

The primary body of this thesis is made up of two separate scenarios for autonomous drone navigation. Both scenarios rely on computer vision and detected places of interest, which are employed in different ways to calculate the return trip. Both scenarios begin the same way: the drone is released and, under the user's direction, navigates above Washington DC.

While navigating controlled by the user, the drone detects and identifies several points of interest and records all relevant information for this specific detection. To be more precise, the following variables have been stored in a txt file. These data include GPS coordinates, altitude, and the type of site of interest found. This information will be required for the drone to independently return to its starting place using the path planning algorithm developed. To be more explicit, the following variables have been saved in a text file.

- class_name
- Score
- Normalized Height
- Normalized Width
- IMU_Data
- Altitude
- Latitude
- Longitude
- Distance Covered
- Keypoints1
- Descriptors1

All the above variables have been used in some stage of the Python code, and this is the main reason that they have been extracted and saved. This helps in maintaining a clean and organized code structure and also accesses them efficiently when needed. We proceed to explain further the variables.

Class name: name of the object detected. This will filter the objects that are going to be saved from all the detected objects. This class name will help to organize and categorize the detected objects, making it easier to manage and analyze the data. It ensures that only relevant objects are saved for further processing or storage. From all the available classes ('car', 'van', 'truck', 'building', 'human', 'gastank', 'digger', 'container', 'bus', 'u_pole', 'boat', 'bike','smoke','solarpanels', 'arm', 'plane'), only the following were selected: 'building', 'gastank', 'container', 'solarpanels'. The reasoning behind my decision to filter the detected objects to only this category is because in real-life scenarios, the drone should not pinpoint and "remember" the location of moving objects that most likely will not be there in the return of the drone to locate them. That's why only stationary objects were saved.

Score: The score is required to assess the accuracy and the quality of detection. Objects having a score < 0.5 will not be examined or taken into account. The primary objective is to acquire accurate and trustworthy data for mapping and surveillance applications.

Normalized Height and Width: Normalized Height and Width: The height and width values represent the size of the boundary detection box that seems to contain the object of interest inside the picture. The term normalized appears to refer to the fact that values are adjusted to a standard scale for comparison and analysis. One easy explanation for this is the drone's altitude. Because of the drone's altitude, items in the image may look smaller or bigger than they are. As a result, the bounding box of the same item seems smaller as the drone's distance from it increases.

IMU_ Data_orientation: From the available data in the IMU category (angular acceleration, angular velocity, linear acceleration, linear velocity, orientation, position), the orientation was the one that was archived and further preprocessed for analysis. The orientation data was converted from the quaternion number system to the cardesian coordinate system. This conversion allowed for easier interpretation and visualization of the IMU data, providing valuable insights into the movement and positioning of the object being tracked. Additionally, by focusing on orientation data, more accurate assessments could be made regarding the object's spatial orientation and changes in direction. A mathematical approach for the conversion from quaternion to an array in Cartesian coordinate system. Firstly, we divide each element of the quaternion by its magnitude to normalize it.

Altitude: As the name implies, it refers to the height above the ground at which the UAV is flying during that specific detection. The variable is also used for height and width normalization, as previously explained.

GPS coordinates (longitude and latitude): GPS coordinates will be used to assess how accurate the drone's estimated positions are. These coordinates will also help in determining the exact location of the UAV during the detection process. Additionally, they provide crucial information for mapping and analyzing the data collected.

Distance Covered: This is the calculated distance that was calculated from two different GPS coordinates using the Haversine formula. This distance will be used in the trilateration function for the coordinate's estimation.

$$Harvesine\ formula\ (d/r) = harvesive\ (\varphi_2 - \varphi_1) + \cos\varphi_1 * \cos 2 * harvesive\ (\lambda_2 - \lambda_1)$$

where:

d is the distance between the two points along a great circle of the sphere

r: is the radius of the sphere (earth in our case)

φ1, φ2 are the latitude of point 1 and latitude of point 2,

λ1, λ2 are the longitude of point 1 and longitude of point 2.

$\theta = \frac{d}{r}$: central angle

$$harvesive\ (\theta) = \sin^2\frac{\theta}{2}$$

Once the central angle (θ) was found, it was used to compute the distance between the two points along the surface of the sphere. The Earth's radius (approximately 6371 kilometers) was multiplied by the central angle to get the distance between the points in kilometers.

Keypoints and Descriptors: Keypoints and descriptos are calculated using the ORB (Oriented FAST and Rotated BRIEF) feature detector and descriptor extractor from the OpenCV library in Python.

Keypoints: A list of keypoint objects, where each keypoint represents a distinctive feature in the image, containing its coordinates, size, angle, and other properties.

Descriptors: A numpy array containing the descriptors (feature vectors) for each keypoint. These descriptors are used for matching and tracking keypoints across different images.

The ORB feature detector and descriptor extractor is known for its efficiency and robustness in computer vision applications. The Scale-invariant feature transform (Sift) was also evaluated; however, the two images' similarity score fell outside of an acceptable range.

Building on the overview from before, the two situations under examination both started in the same manner. Within the Unreal environment, which is a representation of the 3D world, the user is using programming to operate the drone. The drone with the bottom camera scans the city while in midair, picking up various items that are present there.

It is crucial to note at this point that the WALDO (Whereabouts Ascertainment for Low-lying Detectable Objects) detection model was employed up to this point (see Figure 8). The detection AI model WALDO is built on a substantial YOLO-v7 backbone. Stephan Sturges developed the WALDO model, which was trained using sets of synthetic and "augmented" / semi-synthetic data. Promising outcomes have been observed in the WALDO model's ability to correctly recognize a variety of items in the urban environment. It opens up new possibilities for mapping and aerial surveillance applications since it can identify things while the drone is

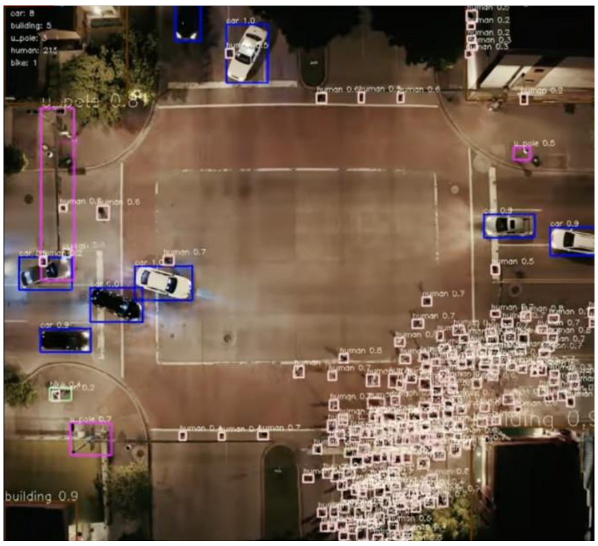


Figure 8: WALDOS detection model

in flight from the perspective of the bottom camera. While detecting the afore mentioned classes from real-world photos is WALDO's primary goal, the detection from the AirSim environment was also quite accurate, particularly for buildings, which were the most often recognized class along with the cars [28]. Even while Waldo assists us in achieving our objective and detecting items in the area we've created in the Unreal Engine, there's always space for development. The ideal case is to re-train WALDO based on labeled training data exported by AirSIm Environment, so that we can fully utilize the potential we have. By doing that, we may reduce the possibility of mistakes arising from the model's training on disparate sets of data. Although we won't be going in this way right now, it will undoubtedly be a future area of study for us.

After the drone has been surrounded over the environment, when the user decides to do so, he can activate the no signal function by pressing the "t" key. At this point, the two scenarios begin to differ.

3.2.1 Scenario 1 – IMU Navigation

At this point, the GPS signal has been jammed and the UAV cannot locate itself in the surrounding environment. So it relies its functionality and the return to its starting position solely on the data that have been collected from the Inertial Measurement Unit (IMU). The IMU device includes an **Accelerometer** that measures acceleration in X, Y, and Z. The drone's orientation and variations in rotational movement may be ascertained with the aid of the Gyroscope (see Figure 9). The magnetometer is used to measure the magnetic field's intensity and direction. It aids in figuring out how the drone is oriented in relation to the magnetic field of Earth.

Based on the above data, the drone proceeds to rotate 180 degrees in the Z axis and thus head to the direction that it came from. In the previously acquired data, with the various reference points where the drone has made the relevant detections, the GPS coordinates are also

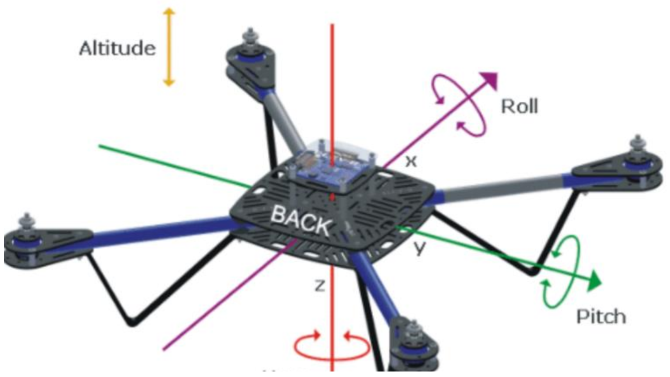


Figure 9: The available movements of the drone.

stored, thus forming a path that leads to its starting point. So by using the Haverstine function, we can calculate the distance the drone had covered to reach this specific point from the previous drone.

Now that both the direction of the movement as well as the distance covered is known , the drone can moved to the exact opposite direction. The velocity of the drone is considered known by the integrated sensors it contains and from the following equation of the average velocity :

$$U = {}^{\Delta x}/_{\Delta t}$$

Where *U*: is the average velocity

Δx: displacement

Δt : the time

the only unknown parameter that can be easily calculated is the time.

Now that the parameters that describe the movement of the drone are known, the drone can move with well-defined velocity towards the opposite direction from the one acquired by the gyroscope. In that way they, and according to the calculations, the UAV will land in the penultimate point and then by repeating the same procedure for all acquired points of interest the drone is arriving to its starting position. This method allows for navigation and control of the drone, ensuring that it reaches its intended destination accurately. By utilizing the data from the sensors and motion equation, the drone can autonomously navigate through various points with ease and efficiency. Figure 10 summarizes the methodology followed for the first scenario.

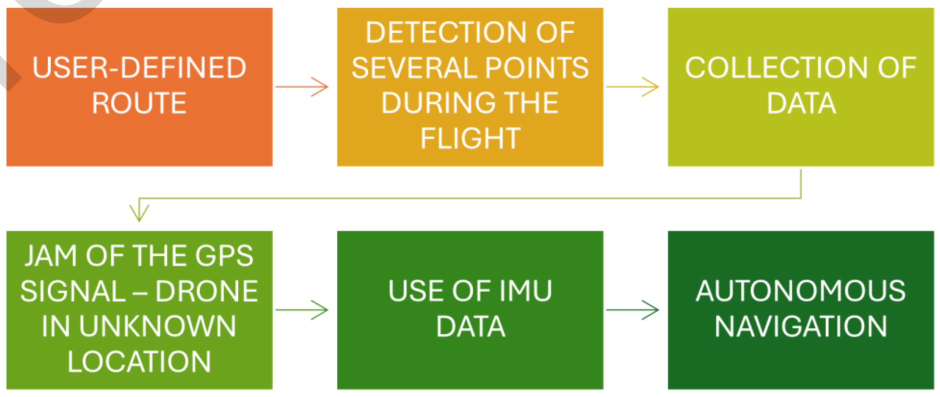

USER-DEFINED ROUTE
DETECTION OF SEVERAL POINTS DURING THE FLIGHT
COLLECTION OF DATA
JAM OF THE GPS SIGNAL – DRONE IN UNKNOWN LOCATION
USE OF IMU DATA
AUTONOMOUS NAVIGATION

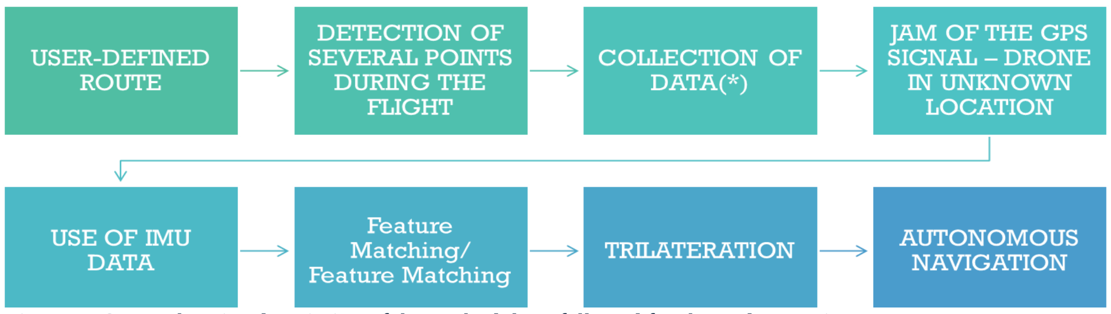
Figure 10 Comprehensive description of the methodology followed for the 1st scenario

3.2.2 Scenario 2 – Feature Matching

The second scenario that our algorithm was based on is more advanced than the first one and is a faster, but more complicated, approach to returning the drone to its home position. This scenario considers more variables and factors to return to the starting position of the drone. It requires a higher level of precision and processing power compared to the first scenario.

Again, in this case, as soon as the UAV loses its signal, it proceeds to move in the opposite direction to locate the last known point. This time, the position of the drone is verified not only by the IMU data but also by feature matching. While trying to return, the drone continues to acquire images from the bottom camera and performs feature matching by using the descriptors that were saved previously. The confidence parameter is an indicator of the correct location of the UAV, because if the number of detections in the current image match with the number of detections is the same previous position, it means that the drone is in a known position. It has been introduced to the model in order to implement α substandard landmark-based navigation. To be more specific, in order to be sure that you are in the correct place, you need to recognize/detect more than one item. That's why if we combine the now detected objects with the previously detected, we will have another indication for our correct position.

Combining the results from the UMU data, the similarity score that rose from the feature matching needs to be higher than 60%, but also the confidence parameter, we can assume that the estimated location of the drone is much more accurate this time. Then the UAV proceeds with the same rationale as before, to locate the next points based on the IMU data. With the same way, firstly the drone rotates 180º in order to head to the direction that it previously came


USER-DEFINED ROUTE
DETECTION OF SEVERAL POINTS DURING THE FLIGHT
COLLECTION OF DATA(*)
JAM OF THE GPS SIGNAL – DRONE IN UNKNOWN LOCATION
USE OF IMU DATA
Feature Matching/ Feature Matching
TRILATERATION
AUTONOMOUS NAVIGATION
Figure 11Comprehensive description of the methodology followed for the 2nd scenario

from, then based on calculations of the velocity, etc (as described in scenario 1) it reached the previous position where it also performs feature matching as described above. Then, the drone locates successfully another 3 points, then through the trilateration procedure it can calculate its current position. Then the results of the current position are evaluated, and the UAVs position is known. This allows the drone to return directly to its starting position, something that makes the return much faster. Figure 11 summarizes the methodology followed for the second scenario.

3.3 Assumptions

During our research, we faced many uncertainties and had to deal with several parameters/variables. To complete our work and obtain results, we had to make several assumptions. These assumptions were essential for guiding our investigation and helping us interpret the data despite the complexities involved. By making these assumptions, we were able to streamline our research process and focus on key factors that would ultimately lead us to meaningful conclusions. While we understood that these assumptions introduced some level of risk, we believed they were necessary in order to move forward with our study.

The absence of wind or any other environmental variables (such as rain) was our initial assumption. The wind plays a big part in the drone's flying circumstances. It has the potential to change the drone's behavior and outcomes by causing the UAV to stray from its presumptive position, changing its path entirely. We wouldn't be able to continue in the context of this thesis since the problem becomes much more difficult when the wind is taken into account. Additionally, the wind modifies the drone's location and has an impact on the model's detecting effectiveness. The trees will always take on a distinctive form when there is wind in the surrounding area. The quality of the outputs and the model's overall performance will suffer as a result of less effective feature matching in the acquired images. Generally speaking, the effects of several natural elements and how they can affect the environment and the visibility were not taken into account. For example, the model's performance will be significantly impacted by the lack of natural light due to limited visibility. Additionally, the presence of dust or fog in the surroundings can provide poor results when attempting to discover places of interest.

We also assume that there is no noise in the IMU model we considered. We do not have any noise in our measurements since our atmosphere is considered perfect, which results in flawless data that is ready for processing. This isn't the case in real life, since noise in GPS locations and IMU data can skew the findings and cause errors.

This work served as a proof-of-concept to begin our research from an initial position and, subsequently, to break through existing barriers and presumptions through this particular field of study. These difficulties serve as a catalyst for us to further our investigation and conquer these difficulties. With the conclusion of this work, we plan to implement a far more realistic model that will take into consideration environmental parameters and data noise.

3.4 Comparison with Literature Review

Now that the methodology we followed has been introduced, we proceed to compare it with the current state-of-the-art methods that were described in the literature review, such as reinforcement learning using deep neural networks.

During our research, we did not use any neural network algorithms, let alone deep reinforcement learning algorithms, for several reasons. First, in order to build a model using advanced techniques, you first need to expose yourself to a more basic concept in order to elevate it in the future. Our research focused on exploring the effectiveness of computer vision techniques for solving the specific problem at hand, rather than delving into the complexities of deep reinforcement learning. This approach allowed us to gain a deeper understanding of the fundamental principles before potentially incorporating more advanced techniques in future studies. As previously indicated, our study serves more as a proof-of-concept for when we move forward with more advanced methods. Second, it takes a lot of time and resources to train a deep neural network. Both were not readily available to us at this point in the investigation. There is no need at this point in comparing our algorithm with a DRL algorithm because they base their functionality on different theoretical backgrounds, and we have not yet reached the level of building such advanced models. In conclusion, it is important to acknowledge the limitations of our current research and recognize the need for further development before attempting comparisons with more advanced algorithms. Additionally, focusing on mastering fundamental principles will lay a solid foundation for future studies involving more sophisticated techniques.

When it comes to the SLAM algorithms, there are several issues to discuss and compare. Both methods based their functionality on the creation of a map, in a way. Our method maps the area where, during the drone flight, it was able to recognize several points of interest. The SLAM method makes a much more detailed version of the current environment, and the drone is able to locate itself inside the map it creates. In contrast, our algorithm is not so elevated. When the environment is GPS denied, the drone aims to locate itself by recognizing known points. That makes our model very sensitive to the scenery, because any changes may lead to the drone not being able to recognize known points and be able to proceed.

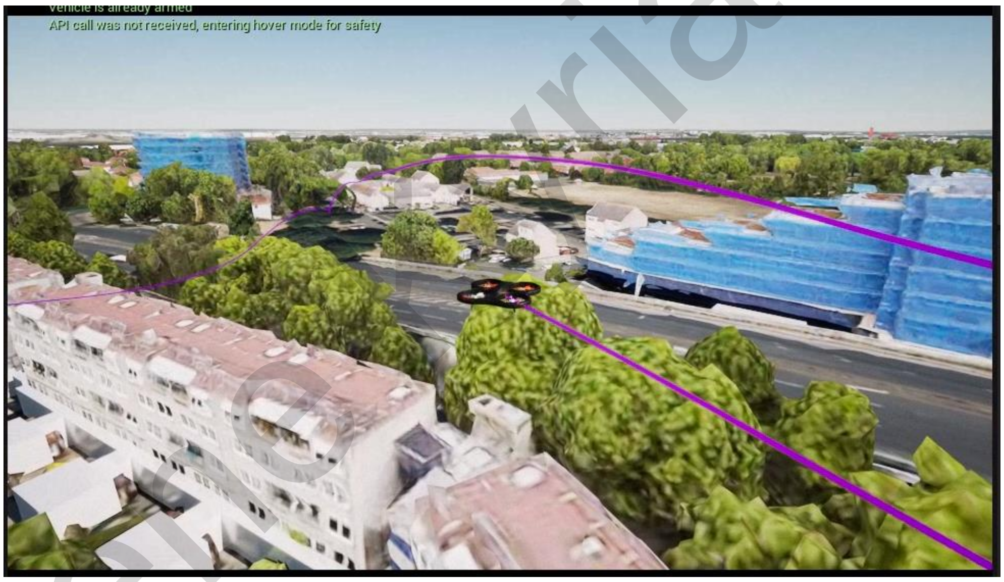
Another difference between the two methods is that our method uses inertial sensors as the main source of the data. On the other hand, for the SLAM method, sensors like lidar can also be used. While the two approaches share a similar basic idea, we are unable to say that our method is functionally equivalent to SLAM.

Chapter 4: Results

This chapter presents the results of our algorithm's assessment based on the two previously described scenarios. This chapter presents a thorough study of the performance measures and outcomes, demonstrating the algorithms' efficacy. By methodically comparing the outcomes across multiple cases, we provide insights into our approach's strengths and potential limitations, paving the path for future advances and applications in UAV technology.

4.1 Scenario 1

For the first scenario (see Figure 12), many different routes of the drone were tested, so to evaluate the model under both simpler and complex routes.



Vehicle is already armed
API call was not received, entering hover mode for safety

Figure 12: The drone returning to the starting point with exactly the opposite direction.

The drone's route is user-defined, making it nearly impossible to follow the same path repeatedly. For this reason, the 2D display of the locations below is suggestive.

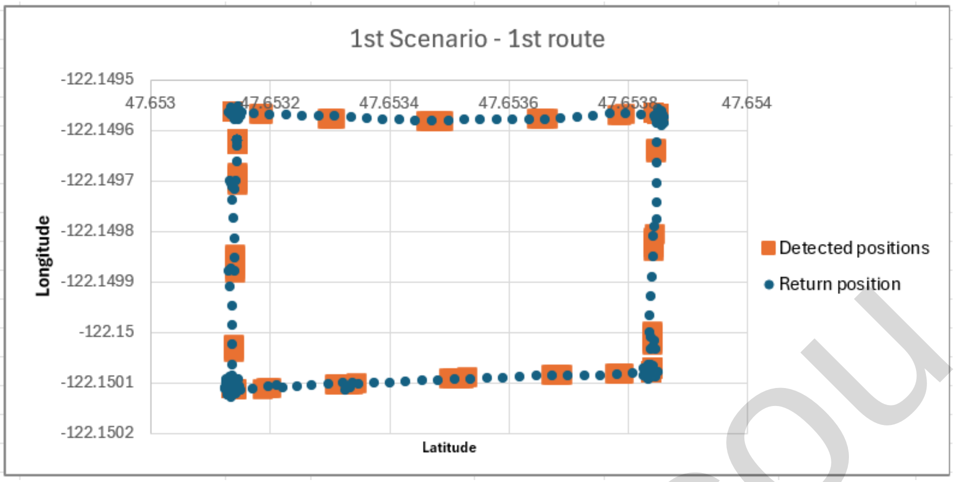


Figure 13: The different points were the drone passed by in 1st scenario /1st route

However, the key point is that the route's complexity falls within the same range. Therefore, the visualization serves as a general guide for understanding the distribution of points and the overall pattern of movement. It may not accurately represent the exact path taken by the drone during each flight due to variations in user-defined routes.

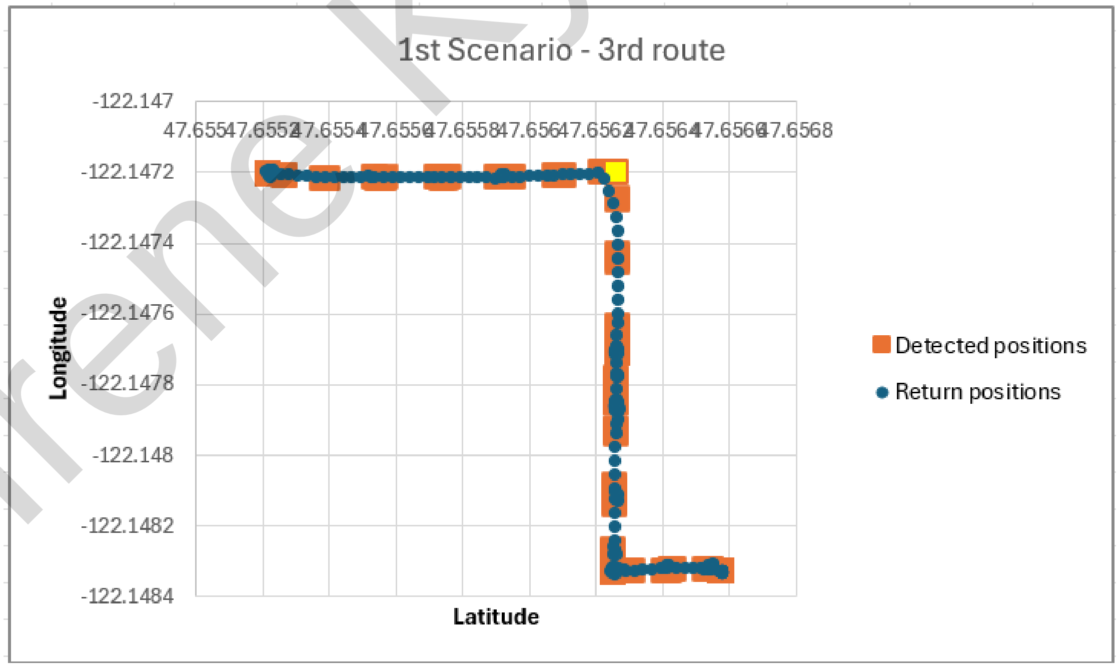


Figure 14: The different points were the drone passed by in 1st scenario /3rd route

Figures 13 and 14 depict the GPS coordinates when the drone was navigating controlled by the user (orange points), and when it was autonomously navigating back to the home position. What is worth mentioning at this point is the difference in the density of the points. The orange points are a bit sparser than the blue points. This is because during the time the UAV is controlled by the user, due to the low quality of image during flight that affected the detection (will be further discuss in the end of this chapter), we needed to make some stops, in order for the drone to acquire some detections. That is the reason the orange points are a bit sparser than the blue. We considered two subscenarios.

First Subscenario

Table 1 showcases the numerical results for all the times we repeat the measurements for the first subscenario.

Table 1: Table of the results of the 1st scenario/ 1st subscenario

	1st time	2nd time	3rd time	4th time	Average value of the per cent difference
% difference starting point altitude	6.04951	28.07803	4.78264	5.787446	11.1548
% difference finishing point altitude	0.01411	0	0.023223	0	≈0
% difference starting point Latitude	1.18405E-05	1.21E-05	1.15E-05	1.63E-06	≈0
% difference starting point Latitude	1.07059E-11	2.31E-12	8.6E-12	1.45E-11	≈0
% difference starting point Longitude	-6.93884E-06	-2.8E-06	-2.5E-06	-7.1E-07	≈0
% difference starting point Longitude	-3.26915E-12	-8.1E-13	-2.5E-12	-4.1E-12	≈0

As can be seen from the above table, the drone navigates back to the starting point with great precision. This is because the calculations for the implementation of the model, were performed very carefully, but also because the experiment was performed in the isolated environment of the simulator. What that means is that they weren't any external factors that may affect the route of the drone. To be more specific, the air can drift the drone away from a point and then it would be much more difficult with this model to obtain the starting position. And in our case, there was absence of air.

Also, another reason why the percent difference is really small, is because the compared values were the latitude and the longitude, that in the scale that the drone was moving the fluctuation of the numbers was negligible.

When compared to the other variables, the altitude is the only one that exhibits the most variance. As things stand right now, this is not an issue because operating the drone will not be harmed if it makes it to the starting position, which is somewhat higher than it should be. But if the altitude variation keeps rising, it may have an impact on the drone's accuracy and performance in future research.

Second Subscenario

In the second subscenario, the same conclusions apply as in the first scenario (see Table 2). In that case we tried to increase and decrease the altitude for bigger distances to see how this will affect the results. The results nevertheless remain in the same scale as the ones in the previous scenario. By that it is safe to say that our model is not that prude to the fluctuation of the elevation. Of course, the huge deviation of the starting point for the first time cannot remain unnoticed. The reason why this number can be ignored is due to the fact, in that specific case the drone was almost in the ground. So, it couldn’t make any detections and therefore have access to their data. For that specific reason this value can be ignored.

Table 2: Table of the results of the 1st scenario/ 2nd subscenario.

	1st time	2nd time	3rd time	4th time	Average value of the per cent difference
% difference starting point altitude	109.6345	5.12196	7.948166	0.115031	30.7049
% difference finishing point altitude	0.324339	0.00318	0.46129	0.002383	0.1978
% difference starting point Latitude	1.08161E-05	2.67363E-06	2.62E-06	4.4755E-06	≈0
% difference starting point Latitude	2.12827E-10	4.20552E-12	2.33E-06	4.1757E-13	≈0
% difference starting point Longitude	-2.40109E-06	-1.55309E-06	-1.2E-06	-3.57524E-06	≈0
% difference starting point Longitude	-2.29205E-11	-8.26059E-13	-6.9E-08	-2.16142E-09	≈0

4.2 Scenario 2

In the second scenario, we combined the first scenario, which mainly used IMU data, with computer vision and the advantages it can bring. In that case, through computer vision and feature matching, we tried to identify known points of interest that were previously detected.

That's why the confidence parameter was introduced. The more objects with a score of ≥ 0.5 that were detected, the higher the confidence was.

Once we were able to access (through the IMU data) and recognize through feature matching (see similarity score), we were able to calculate the position of the drone through trilateration. Now that the drone's coordinates are known, i.e., the drone can determine its position along with the other known points, it can easily return to its starting position without having to pass by every single point, like the previous scenario.

For the second scenario, we obtained the following results:

Table 3: Table of the results of the 2nd scenario/ 1st attempt

1st attempt	Objects detected	Confidence	Calculated position from Trilateration	
Coordinates: (47.64217824457132, -122.14286415160201)	class building score 0.74 score 0.73 score 0.86	0.66	Real	Calculated
Coordinates (47.642187997514846, -122.144062617662)	class building score 0.75 score 0.66 score 0.56 score 0.94 score 0.92	0.60	(47.641528963877654, -122.14274065332683)	(47.18436518585982.- 122.1430475221796)
Coordinates (47.64369971007048, - 122.14445671837244)	class building score 0.88 score 0.85 score 0.87 score 0.58 score 0.85 score 0.61	0.5		

The similarity score from the feature matching can be found in the table below.

Table 4: Similarity score of the 1st scenario/ 1st attempt

1st Image	Similarity score =0.63
2nd Image	Similarity score =0.57
3rd Image	Similarity score =0.48

Looking at the results of our first attempt we can extract the following. First of all the percentage difference of the calculated value compared to the real value is ≈0.1% which is negligible. Also, we can see that most (>50%) of the detected objects were also detected the second time that the drone autonomously navigated to that position. This can be shown through the confidence factor that is >= 0.5. On the other hand, when it comes to the similarity score of the images obtained from the 3 different positions, we can see that the 3rd image did not pass the threshold of 0.5 that we set, and also the similarity score for the other 2 images is not that high either, but in acceptable range, nevertheless. This helps us understand that the image quality is not the ultimate and it will be a challenge to overcome.

Table 5: Table of the results of the 2nd scenario/ 2nd attempt

2nd Attempt	Objects detecte d	Confidenc e	Calculated position from Trilateration	
Coordinates: (47.644122842278830, -122.14451556576326)	class building score 0.89 score 0.30	0.80	Real	Calculated
Cordinates (47.64411529980673, -122.1440092296648)	lass building score 0.8 score 0.66 score 0.61 score 0.53 score 0.81 score 0.76 score 0.76	0.72	(47.644431318129705 ,-122.14136595508255)	(47.6421879975146476, -122.144062618762)
Cordinates (47.64409755091687 - 122.14134777730263)	class building score 0.89	1		

The similarity score from the feature matching can be found in the table below.

Table 6: Table1: Similarity score of the 1st scenario/ 2nd attempt

1st Image	Similarity score =0.56
2nd Image	Similarity score =0.57
3rd Image	Similarity score =0.66

For our second attempt we can see that our extracted results are better than our first attempt. To start with, the confidence has higher value which means that objects that were detected in the first time we obtain the information (while the drone was controlled by the user) were also detected the second time while the drone was autonomously navigated. Also the similarity score for all three images is above the threshold we set (>=0.5), which can be considered success. Last but not least, the difference between the calculated value compared to the real value is ≈0%.

Table 7: Table of the results of the 2nd scenario/ 3rd attempt

3rd Attempt	Objects detected	Confidence	Calculated position from Trilateration	
Coordinates: (47.647314481628676 ,- 122.14155474784825)	class building score 0.78 score 0.7 score 0.65 score 0.56	0.80	Real	Calculated
Cordinates (47.648660004882935 ,- 122.14335125896753)	class building score 0.8 score 0.56 score 0.63	0.2	(47.64484136404663 , - 122.14136674641986)	(47.6421879975146476, -122.144062618762)
Cordinates (47.650219223291316, - 122.14403105078732)	class building score 0.79 score 0.71 score 0.65	0.58		

The similarity score from the feature matching can be found in the table below.

Table 8: Similarity score of the 2nd scenario/ 3rd attempt

1st Image	Similarity score =0.46
2nd Image	Similarity score =0.53
3rd Image	Similarity score =0.66

Finally, for our last attempt, even though the difference of the real and calculated GPS Coordinates is really small (0.002%), the individual results of similarity score and confidence, are poor. When it comes to the similarity score 2 out of 3 images are within the acceptable range we set, but one image has score similarity score <0 .5. Also, the confidence for the second point is very low. This means that the not all objects were detected the second time. This can be caused due to the fact that our model is prone to the altitude that can alter the quality of the image, that will affect the confidence as well as the similarity score.

4.3 Challenges

Of course, there were many challenges to be able to achieve the above-presented results as well as assumptions that were mentioned earlies. In the list below the main challenges, we faced to extract our results can be found.

- Poor image quality for detection- from AirSim
- Results prone to altitude
- Huge dependency in detection

First, we had to lower the threshold of the accepted detections to 0.5. Otherwise, there were going to be very few detections, and we wouldn't be able to proceed with the rest of the procedure. The same applies to the similarity score. In normal conditions, with real images and not from AirSim, the threshold was going to be at least 0.7 to ensure the quality of the data. These values for the similarity score and for the accuracy of the detection were set to this low level, in order to overcome the challenge of the poor quality of the image. Returning to the topic mentioned earlier, the quality of the image that was extracted from the drone was getting worse as the altitude was rising.

In the pictures depicted in Figure 15, it is obvious how they are altered due to the high elevation of the drone. That is why, for the sake of better results, it was crucial to fly the drone at a normal altitude of approximately 30m.

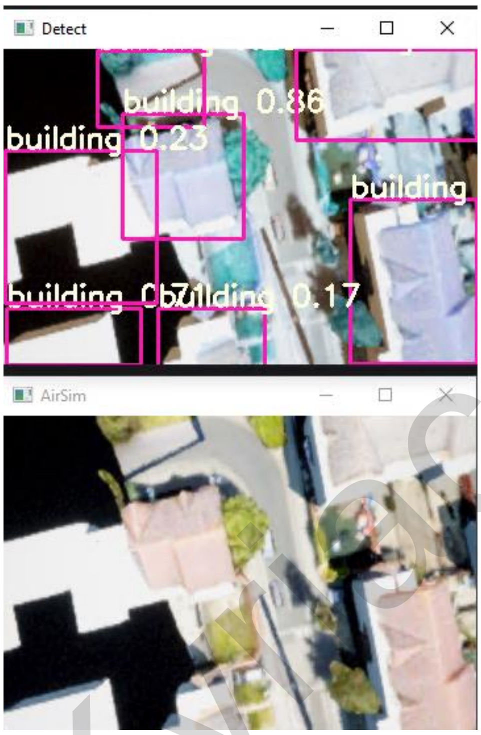


Figure 15: Images acquired from AirSIm during detection.

The last of the challenges we faced was the huge dependency on the detections for the autonomous navigation. Currently, in the environment we were working, there were plenty of objects to detect. In the case our environment was a desert, then it would be much more challenging to detect points of interests and then proceed with the rest of the algorithm.

Chapter 5: Future Work

The research that has been done so far is only the tip of the iceberg regarding what can be optimized in the future as well as what different paths it can take. Further exploration into different methodologies and technologies could potentially lead to even more efficient and innovative solutions in the field. The future of this research for the advancements in autonomous navigation has many capabilities.

The development of autonomous navigation includes advancements in artificial intelligence and machine learning algorithms to improve decision-making capabilities. Future models will be able to handle much more advanced and complicated scenarios. To be more specific, a very common scenario during rescue operations is one where the drone is not able to follow the initial path or there are restricted areas (with fire, for instance) where the drone cannot pass by and it will have to follow a different path in order to reach its final position. This requires drones to have the ability to adapt and reroute their navigation in real-time, ensuring they can still reach their destination efficiently. These will be achieved by introducing deep reinforcement learning algorithms that will be trained in many different use cases. As technology continues to evolve, we can expect autonomous drones to become even more adept at navigating complex and dynamic environments.

Combining the two scenarios, as described above, our model will be further improved to achieve better accuracy in the results. The main idea is that while the drone returns from the same path following the previously acquired points of interest, it will perform along the way the trilation process with many different points. This will lead to an improvement in the accuracy of the calculated positions, with much better results along the way.

Additionally, another improvement for our model is its implementation in a closer environment to our location. To be more specific, as described in previous chapter, it would be a better option if the environment that the UAV was placed to be in Cyprus, and ultimately, here in Nicosia, were we will be able to test and verify the results of our implementation.

Naturally, there is no need to restrict the use of drones to just one. Leveraging the combined capabilities of several unmanned aerial vehicles working in unison, UAV swarms have shown to provide substantial advantages. Swarms can provide increased coverage and persistent surveillance over vast areas, with the ability to loiter for extended durations and observe targets from various vantage points. The distributed nature of swarms enhances robustness and redundancy, ensuring that the overall system continues to function even if individual UAVs are lost or compromised. Additionally, UAV swarms can also perform complex tasks in search and rescue mission more efficiently than a single drone. By dividing tasks among multiple drones, swarms can accomplish objectives faster and with greater precision. Furthermore, the use of swarms can enable more sophisticated communication and coordination strategies, allowing for seamless collaboration between the drones to achieve common goals. Also decentralized coordination algorithms are essential for the autonomous navigation of UAV swarms, enabling individual UAVs to make decisions based on local data while contributing to collective objectives. Unlike centralized systems, decentralized approaches offer advantages in robustness, scalability, and resilience. In future research, we will explore key aspects and potential directions for decentralized coordination algorithms. This includes developing adaptive algorithms capable of real-time responses to dynamic environments and unexpected events, such as obstacle avoidance, mission reconfiguration, and handling UAV malfunctions. Additionally, enhancing communication within UAV swarms is crucial for their success. Research efforts should focus on improving the reliability, speed, and range of inter-UAV communication. This involves designing scalable communication protocols that support large numbers of UAVs without significant performance degradation, as well as investigating methods to mitigate the impact of communication interference caused by environmental factors or intentional jamming, especially in hostile or cluttered environments. Overall, the potential benefits of utilizing UAV swarms in various applications are vast and continue to expand as technology advances.

Also, a new detection model will be introduced that is more suitable than the current one. This new model will include the Ultralytics YOLOv8 which is a state-of-the-art model.

YOLOv8 is designed to be fast, accurate, and easy to use, making it an excellent choice. It will be re-trained from data extracted through the AirSim environment. In this way, the detection will be much more advanced and accurate and will reflect the current environment. Retraining YOLOv8 with footage captured from a drone here in Nicosia, in addition to data retrieved from the AirSim environment, is another pertinent contribution. The detection will then approach the necessary level, allowing us to more accurately assess the algorithm's accuracy.

The model's actual drone deployment is, of course, the ultimate objective. To do that, the algorithm developed using the simulator must be evaluated in a variety of environmental settings, and variables that might have an impact on the drone's performance must be considered. One aspect of nature that might cause the drone to drift and increase computation errors is the wind. By predicting and testing the algorithm in different settings (night/ dark environments, snow, windy day) through the simulator, we can make a more robust model. This will ultimately lead to more successful real-world drone deployments and advancements in drone technology. Additionally, by fine-tuning the algorithm in various environmental conditions, we can ensure that the drones are equipped to handle any challenges they may face during deployment. So, by ensuring that all the assumptions that were made in this thesis were resolved then we can be much more confident for the robustness of our model in real-life circumstances. As we continue to delve deeper into this field, we aim to build upon the initial findings of this proof-of-concept study and ultimately develop innovative solutions that challenge the status quo. By questioning existing assumptions and pushing the boundaries of current knowledge, we hope to pave the way for new advancements and breakthroughs in this area of research. Our goal is to not only overcome the current barriers but also to revolutionize the way we approach this field and drive progress forward.

Looking ahead, the ongoing advancements in autonomous navigation of drones hold tremendous potential. Future research should focus on enhancing the robustness and adaptability of navigation algorithms to ensure reliable performance in diverse and unpredictable environments. Integrating cutting-edge technologies such as artificial intelligence, machine learning, and edge computing will be crucial for enabling real-time decision-making and seamless coordination among multiple drones.

References

[1] J. Kedys, I. Tchappi, and A. Najjar, “UAVs for Disaster Management - An Exploratory Review,” *Procedia Comput. Sci.*, vol. 231, pp. 129–136, 2024, doi: https://doi.org/10.1016/j.procs.2023.12.184.

[2] R. D. Arnold, H. Yamaguchi, and T. Tanaka, “Search and rescue with autonomous flying robots through behavior-based cooperative intelligence,” *J. Int. Humanit. Action*, vol. 3, no. 1, p. 18, 2018, doi: 10.1186/s41018-018-0045-4.

[3] B. He, X. Ji, G. Li, and B. Cheng, “Key Technologies and Applications of UAVs in Underground Space: A Review,” *IEEE Trans. Cogn. Commun. Netw.*, p. 1, 2024, doi: 10.1109/TCCN.2024.3358545.

[4] G. Gugan and A. Haque, “Path Planning for Autonomous Drones: Challenges and Future Directions,” *Drones*, vol. 7, no. 3, 2023, doi: 10.3390/drones7030169.

[5] M. Y. Arafat, M. M. Alam, and S. Moh, “Vision-Based Navigation Techniques for Unmanned Aerial Vehicles: Review and Challenges,” *Drones*, vol. 7, no. 2, 2023, doi: 10.3390/drones7020089.

[6] V. R. F. Miranda, A. M. C. Rezende, T. L. Rocha, H. Azpúrua, L. C. A. Pimenta, and G. M. Freitas, “Autonomous Navigation System for a Delivery Drone,” *J. Control. Autom. Electr. Syst.*, vol. 33, no. 1, pp. 141–155, 2022, doi: 10.1007/s40313-021-00828-4.

[7] A. Patrik *et al.*, “GNSS-based navigation systems of autonomous drone for delivering items,” *J. Big Data*, vol. 6, no. 1, p. 53, 2019, doi: 10.1186/s40537-019-0214-3.

[8] T. Elmokadem and A. V Savkin, “Towards Fully Autonomous UAVs: A Survey.,” *Sensors (Basel).*, vol. 21, no. 18, Sep. 2021, doi: 10.3390/s21186223.

[9] V. S. Ajith and K. G. Jolly, “Unmanned aerial systems in search and rescue applications with their path planning: a review,” *J. Phys. Conf. Ser.*, vol. 2115, no. 1, p. 12020, Nov. 2021, doi: 10.1088/1742-6596/2115/1/012020.

[10] N. Tuśnio and W. Wróblewski, “The Efficiency of Drones Usage for Safety and Rescue Operations in an Open Area: A Case from Poland,” *Sustainability*, vol. 14, no. 1, 2022, doi: 10.3390/su14010327.

[11] M. Lyu, Y. Zhao, C. Huang, and H. Huang, “Unmanned Aerial Vehicles for Search and Rescue: A Survey,” *Remote Sens.*, vol. 15, no. 13, 2023, doi: 10.3390/rs15133266.

[12] P. Holzmann, C. Wankmüller, D. Globocnik, and E. J. Schwarz, “Drones to the rescue? Exploring rescue workers’ behavioral intention to adopt drones in mountain rescue missions,” *Int. J. Phys. Distrib. Logist. Manag.*, vol. 51, no. 4, pp. 381–402, Jan. 2021, doi: 10.1108/IJPDLM-01-2020-0025.

[13] C. Wankmüller, M. Kunovjanek, and S. Mayrgündter, “Drones in emergency response – evidence from cross-border, multi-disciplinary usability tests,” *Int. J. Disaster Risk Reduct.*, vol. 65, p. 102567, 2021, doi: https://doi.org/10.1016/j.ijdrr.2021.102567.

[14] Y. Mekdad *et al.*, “A survey on security and privacy issues of UAVs,” *Comput. Networks*, vol. 224, p. 109626, 2023, doi: https://doi.org/10.1016/j.comnet.2023.109626.

[15] S. A. H. Mohsan, N. Q. H. Othman, Y. Li, M. H. Alsharif, and M. A. Khan, “Unmanned aerial vehicles (UAVs): practical aspects, applications, open challenges, security issues, and future trends,” *Intell. Serv. Robot.*, vol. 16, no. 1, pp. 109–137, 2023, doi: 10.1007/s11370-022-00452-4.

[16] X. Wang, Y. Sun, Y. Xie, J. Bin, and J. Xiao, “Deep reinforcement learning-aided autonomous navigation with landmark generators,” *Front. Neurorobot.*, vol. 17, 2023, doi: 10.3389/fnbot.2023.1200214.

[17] F. WANG, X. ZHU, Z. ZHOU, and Y. TANG, “Deep-reinforcement-learning-based UAV autonomous navigation and collision avoidance in unknown environments,” *Chinese J. Aeronaut.*, vol. 37, no. 3, pp. 237–257, 2024, doi: https://doi.org/10.1016/j.cja.2023.09.033.

[18] S. B. Laina *et al.*, “Scalable Autonomous Drone Flight in the Forest with Visual-Inertial SLAM and Dense Submaps Built without LiDAR,” 2024, [Online]. Available: https://arxiv.org/abs/2403.09596v1

[19] G. Sapkota and S. Madria, “Landmark-based Localization using Stereo Vision and Deep Learning in GPS-Denied Battlefield Environment,” 2024, [Online]. Available: http://arxiv.org/abs/2402.12551

[20] A. M. Bahasan, “An Accurate Indoor Navigation System Based on IMU/LP-MM Integrated Method Using Kalman Filter Algorithm,” *Hadhramout Univ. J. Nat. Appl. Sci.*, vol. 19, no. 2, 2022.

[21] L. Rabiu, A. Ahmad, and A. Gohari, “Advancements of Unmanned Aerial Vehicle Technology in the Realm of Applied Sciences and Engineering: A Review,” *J. Adv. Res. Appl. Sci. Eng. Technol.*, vol. 40, pp. 74–95, 2024, doi: 10.37934/araset.40.2.7495.

[22] T. Oliveira, P. Trindade, D. Cabecinhas, P. Batista, and R. Cunha, “Rapid Development and Prototyping Environment for Testing of Unmanned Aerial Vehicles,” in *2021 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, 2021, pp. 191–196. doi: 10.1109/ICARSC52212.2021.9429816.

[23] Q. Bu, F. Wan, Z. Xie, Q. Ren, J. Zhang, and S. Liu, “General simulation platform for vision based UAV testing,” in *2015 IEEE International Conference on Information and Automation*, 2015, pp. 2512–2516. doi: 10.1109/ICInfA.2015.7279708.

[24] Z. Chen, J. Yan, B. Ma, K. Shi, Q. Yu, and W. Yuan, “A Survey on Open-Source Simulation Platforms for Multi-Copter UAV Swarms,” *Robotics*, vol. 12, no. 2, pp. 1–24, 2023, doi: 10.3390/robotics12020053.

[25] “Drone Simulators: Enhancing Skills and Reducing Risk for Pilots”, [Online]. Available: https://www.linkedin.com/pulse/drone-simulators-enhancing-skills-reducing-risk-pilots-clearspot-ai

[26] “Unreal Editor Documentation 5.2”, [Online]. Available: https://dev.epicgames.com/documentation/en-us/unreal-engine/tools-and-editors-in-unreal-engine?application_version=5.2

[27] “inicosia - Digital Twin”, [Online]. Available: https://inicosia.cyens.org.cy/#about

[28] “WALDO v2.5 (Whereabouts Ascertainment for Low-lying Detectable Objects)”, [Online]. Available: https://github.com/stephansturges/WALDO