NEW YORK UNIVERSITY ABU DHABI
SENIOR DESIGN CAPSTONE: FINAL REPORT

Collaborative Control Design for a Drone Swarm

Submitted by:

Prince Steven Annor
Computer Engineering Program

Alison Waterman
Electrical Engineering Program

Sampanna Bhattarai
Mechanical Engineering Program

Advisors:

Prof. Anthony Tzes
Professor of Electrical and Computer Engineering

Prof. Jeremy Teo
Assistant Professor of Mechanical and Biomedical Engineering

May 10, 2020

Abstract

Key words: Visual Area Coverage, Simultaneous Localization and Mapping, Relative Localization, PixHawk Autopilot System, Fiducial Markers, Ad-hoc Communication between Mobile Agents

Simultaneous localization and mapping (SLAM) allows a drone to navigate and perceive its surrounding environment. In the past, drone systems have employed many technologies to perform relative localization, including various motion detection systems [Babinec et al. (2014)]. With the reduced cost of high-resolution cameras and increased computational capabilities of on-board computers, computer vision techniques are now being investigated as additive sensors to inertial measurement unit systems [Romero-Ramirez et al. (2018)]. Several computer vision techniques employ fiducial markers with a recognizable pattern, in order to perform pose estimation between the camera and the attached markers [Garrido-Jurado et al. (2014)] in the collaborating drones. This concept is used for relative drone localization while each drone performs SLAM using cameras, LiDAR and an IMU [Bosse et al. (2012); Pierzchała et al. (2018)]. In this project, a multi-sensor integration of a LiDAR sensor and a spherical-lens camera is used along with a wireless ad-hoc communication channel to assist in a collaborative control design for a drone swarm for visual aerial coverage purposes. Using simulations, a real-time Voronoi tessellation of the area in NYUAD's Robotics lab for two of the designed drones flying at the same height is also derived and the area coverage is achieved by commanding each drone to move towards the centroid of its assigned Region of Responsibility.

CONTENTS

1 Project Management **5**
- 1.1 Work Breakdown Structure . . . 5
- 1.2 Design Structure Matrix . . . 6
- 1.3 Critical Path . . . 7
- 1.4 Gantt Chart . . . 8
- 1.5 Changes to Project Management . . . 9

2 Problem Definition **9**
- 2.1 Problem Analysis . . . 9
- 2.2 Problem Clarification . . . 9
- 2.3 Problem Statement . . . 10
- 2.4 Design Constraints . . . 11
  - 2.4.1 Technical Constraints . . . 11
  - 2.4.2 Non-Technical Constraints . . . 11

3 Conceptualization **12**
- 3.1 Background Research & Key Terminology . . . 12
- 3.2 Concept Generation with Morphological Chart . . . 18
- 3.3 Concept Selection with Pugh Chart . . . 18

4 Modeling, Simulation, and Optimization Plan **19**
- 4.1 Trajectory Planning . . . 19
- 4.2 Swarm-Drone Wireless Network . . . 20
- 4.3 Virtual Simulations . . . 23
- 4.4 Optimization Performed . . . 24

5 Final Design **25**
- 5.1 Initial Design Proposed . . . 25
- 5.2 Changes Made to Initial Design . . . 25

6 Budget **26**

7 Implementation Details **27**
- 7.1 Description of Implementation . . . 27
- 7.2 Voronoi-tessellation of rectangular area . . . 28
- 7.3 Control law development . . . 30
- 7.4 Challenges Faced During Implementation . . . 31
- 7.5 Changes Made During Implementation . . . 32
- 7.6 Initial Results . . . 32
- 7.7 Possible Future Extensions of the Project . . . 39

8 Impact of COVID-19 on the Capstone Project **40**

9 Ethics **40**

10 Design Expectations vs. Achievements **41**
10.1 Design Testing using the Motion Capture System 41
10.2 Explanation of Testing and Evaluation Criteria 42
10.3 Conclusion 43

11 Appendix **43**

1 Project Management

1.1 Work Breakdown Structure

The work breakdown structure provides a tentative timeline for project milestones and sub-tasks.

Work Breakdown Structure: Collaborative Control of a Drone Swarm		Dates and Duration		
Primary Tasks	**Sub-Tasks**	**Duration (Days)**	**Start**	**End**
1.0 Achieve basic familiarity with lab equipment and software	1.1 Complete ROS installation and introductory tutorials	8	9/3/2019	9/11/2019
	1.2 Complete ROS tutorials on nodes, topics, messages, publisher, subscriber	5	9/11/2019	9/16/2019
	1.3 Basic startup of Turtlebot, driving turtlebot with computer keyboard	9	9/16/2019	9/25/2019
	1.4 Create ad-hoc connection between robot and the 360 camera	7	9/17/2019	9/23/2019
	1.5 Stream images from the 360 camera to the Intel Nuc (robot computer)	7	9/25/2019	10/1/2019
2.0 Implement 2D control using Turtlebots and conduct testing to determine the best sensing mechanism	2.1 Drive turtlebot in a circular path 5x and quantify extent of failure	9	9/25/2019	10/2/2019
	2.2 Become familiar with Vicon system to locate turtlebot in fly cage, attach spherical markers	1	9/18/2019	9/19/2019
	2.3 Read data from Turtlebot lidar and accelerometer	10	10/2/2019	10/12/2019
	2.4 Use Vicon to track turtlebot position and determine deviation from ideal path	10	10/2/2019	10/12/2019
	2.6 Map spherical images to 2D planar rectangles that can be used for computer post-estimation	14	10/23/2019	11/6/2019
	2.7 Install the Velodyne LiDAR on top of the turtlebot using 3D-printed custom mount and screws	2	11/6/2019	11/8/2019
	2.8 Test obstacle detection using the Velodyne LiDAR, creating a program that stops the Turtlebot when it is a certain distance away from the obstacle.	5	11/8/2019	11/13/2019
	2.9 Implement SLAM navigation code and conduct a preliminary test of its effectiveness by mapping the A5 atrium space	14	11/13/2019	11/27/2019
	2.10 Create a test setup using obstacles in a small room and create a video documenting the SLAM navigation	6	11/27/2019	12/1/2019
	2.11 Finalize capstone proposal and presentation, refining the design conceptualization and creating a more detailed plan for future implementation	8	12/1/2019	12/8/2019
3.0 Integrate multi-sensor design into ground vehicles, test and refine the control system	3.1 Integrate the 360 camera onboard the Turtlebot ground vehicle	7	1/6/2020	1/13/2020
	3.2 Test the performance of the localization and mapping using both LiDAR and 360 camera techniques	14	1/13/2020	1/27/2020
	3.3 Design a damping system for use onboard the drones, and finalize sensor mounting strategies	7	1/27/2020	2/3/2020
	3.4 Develop an emergency strategy plan and review safety procedures for working with drones	4	2/3/2020	2/7/2020
	3.5 Learn how to pilot the drones using the Ardupilot Mission Planner	26	1/15/2020	2/10/2020
4.0 Integrate multi-sensor design into drone swarm, test and refine the control system	4.1 Test the spherical camera and LiDAR pose estimation systems	7	2/10/2020	2/17/2020
	4.2 Conduct models and simulations of the system performance, and update the design as suggested by simulations	27	2/17/2020	3/15/2020
	4.3 Develop the trajectory planning and develop a test flight procedure according to desired outcomes	7	2/24/2020	3/2/2020
	4.4 Conduct trajectory planning test flights	13	3/2/2020	3/15/2020
5.0 Finalize the design, compile results of testing, and prepare final report	5.1 Finalize the design and troubleshoot any points of failure	14	3/9/2020	3/23/2020
	5.2 Conduct second round of trajectory planning experiments (solving delay issue)	14	3/23/2020	4/6/2020
	5.3 Prepare and troubleshoot area coverage experiment	14	4/6/2020	4/20/2020
	5.4 Compile test results into final report	25	4/6/2020	5/1/2020
	5.5 Code the area coverage simulation and add it to the report	4	4/29/2020	5/3/2020
	5.6 Prepare the poster and remaining deliverables	9	4/24/2020	5/3/2020
	5.7 Edit and submit the final report under the direction of capstone advisor and capstone mentor	9	5/1/2020	5/10/2020

Figure 1.1: Work Breakdown Structure

1.2 Design Structure Matrix

The design structure matrix indicates how the tasks are interrelated, by showing which tasks depend on the completion of previous tasks.

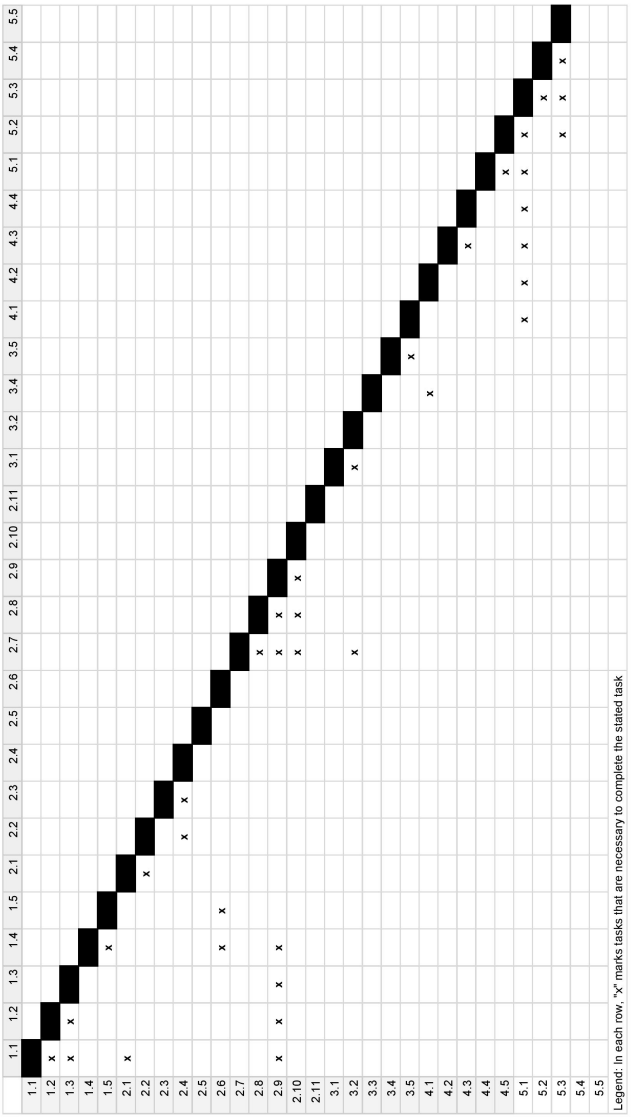
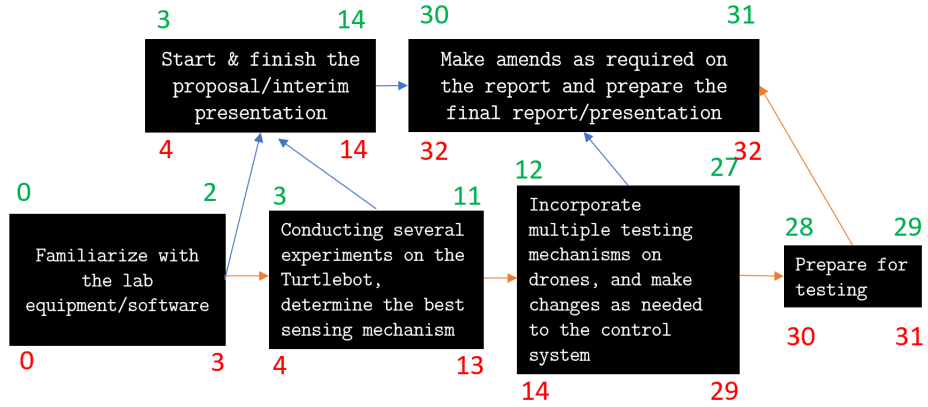


Figure 1.2: Design Structure Matrix

1.3 Critical Path

The Critical Path diagram reveals the optimal order of task completion in the minimum amount of time. It also reveals how delays in one task will affect the timing of the other tasks. This information is used to optimize the workflow during the project and to avoid unnecessary delays.



Green: Earliest start and finish
Red: Latest start and finish
→: Critical path
Critical path weeks: Fall 2019+ Winter 2019+ J-term 2020+Spring 2020

Figure 1.3: Critical Path

1.4 Gantt Chart

The Gantt chart is a useful tool to visualize the work flow timeline, as it creates a clear picture of the start and end dates of sub-tasks. For more detailed descriptions of each sub-task, please see the corresponding entries in the Work Breakdown Structure.

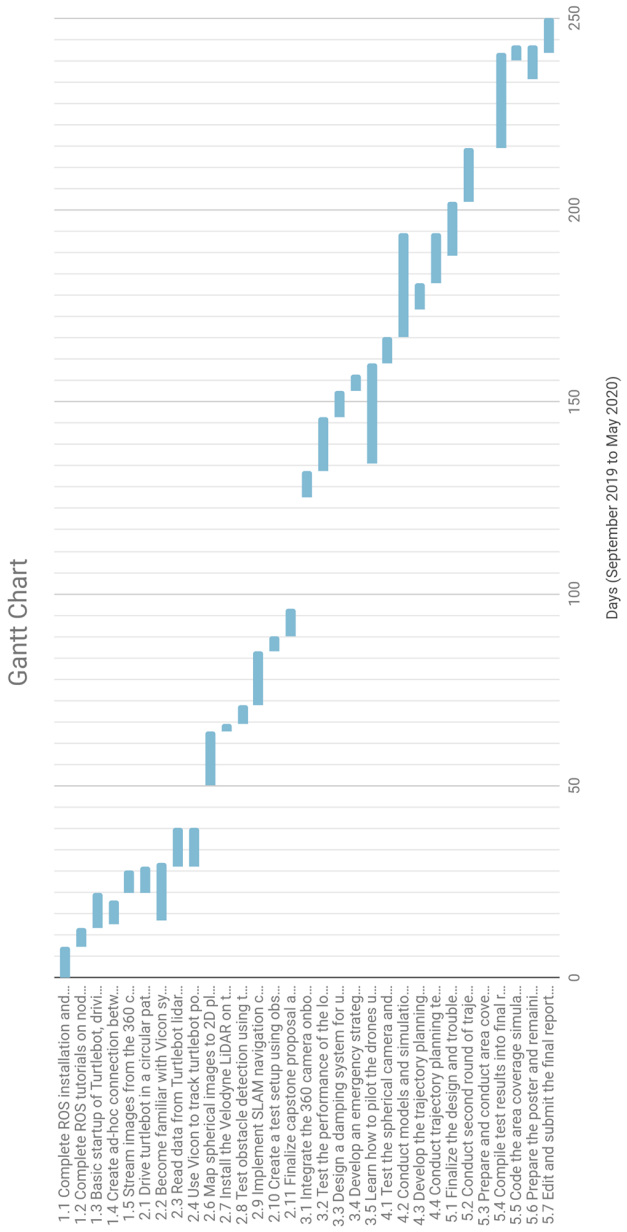


Figure 1.4: Gantt Chart

1.5 Changes to Project Management

Due to the rapid spread of the COVID-19 pandemic, all NYUAD facilities have closed for the past six weeks, with most activities only able to take place virtually. This initially posed a problem for our project, as it relies heavily on access to the NYUAD's CTP-Kinesis Lab. Several key tests were delayed as a result of the intermittent lab access described in *Impact of Covid-19 on the Capstone Project.*

2 Problem Definition

2.1 Problem Analysis

In agriculture, search and rescue, environmental management, and defense domains, autonomous systems are employed to navigate and map unfamiliar environments [Albani et al. (2017), Packer and Josh (2013)]. Historically, drone-swarms, comprised of at least two collaborating Unmanned Aerial Vehicles (UAVs), have been used to perform tasks in these sectors. These drones need to map the target area, avoid obstacles, and estimate the relative pose and distance of other drones in the swarm. For some tasks, drones must collectively patrol a target area, with each drone covering a portion of the total space. Previous approaches such as Voronoi partitioning (see *Key Terminology*) have been implemented to optimize the area coverage problem [Tzes et al. (2018)]. However, current navigation and localization systems are often inaccurate, prone to sensor noise, and cannot perform robustly across various environmental conditions, which poses a challenge to aerial area coverage systems. One cause of this sensor noise is the induced vibration of the drone as it flies. In addition, GPS-localizationcannot be used indoors, which further complicates the issue of localization. This project seeks to address these challenges and implement a control design for localization, mapping, and area coverage using drone swarms.

2.2 Problem Clarification

The following structure is used to represent this project.

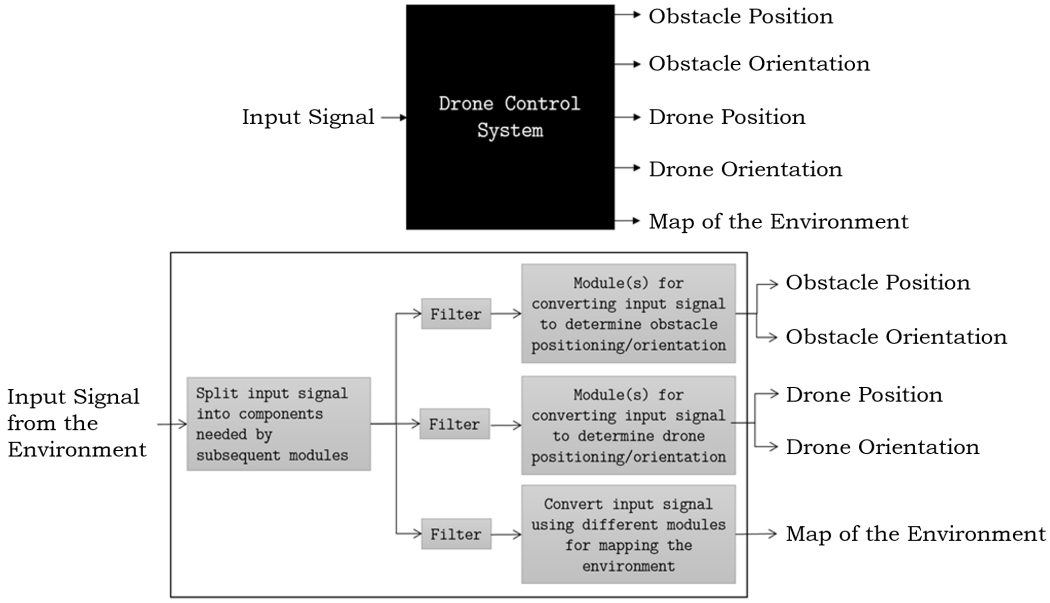


Figure 2.1: Drone-Swarm Problem Structure

In this case, the input signal from the environment represents data captured by the sensor system on-board the drone. This signal is then filtered to remove noise and used to obtain the desired output parameters: position and orientation of obstacles, position and orientation of other drones in the swarm, and a comprehensive map of the environment. The key sub-tasks in this problem are therefore to identify and estimate the pose and location of other drones in the swarm, to generate the environment map, and to localize the drone itself relative to the objects in its environment. This can be done with the output data shown in the aforementioned structure.

2.3 Problem Statement

The initial aim of this project is to design a solution for the relative localization of drones in a team and the mapping of a target region with maximum dimensions of 15m x 5m x 8m (length x width x height). To avoid collisions, each drone should be able to detect the location of neighboring drones at a distance of up to 200 cm and perform pose estimation with linear distance and angular orientation accuracy of 0.5cm and 2° respectively. Based on the relative localization, each on-board computer unit should perform the Voronoi tessellation of the target region under the condition of a fleet of two drones flying at the same height. The optimal visual area coverage will be achieved by moving each drone towards the centroid of its assigned Region of Responsibility (RoR). The project should deliver a robust control design that can be used across a variety of relevant industrial applications and environmental conditions.

2.4 Design Constraints

2.4.1 Technical Constraints

A number of technical constraints were taken into account when designing a SLAM-system for visual coverage. One is the range of the sensors used for mapping. Most methods of acquiring environmental data only collect information within a certain proximity to the sensor. Some sensors are directional and only collect data from a limited field of view, while others collect data from all directions simultaneously. For the purposes of our problem, drones in the swarm must be able to localize other drones within a maximum radius of 200cm in all directions. Thus, sensing techniques with an adequate range must be used.

Another technical constraint is the accuracy of the sensors used for pose estimation. The sensors should be able to pinpoint a neighboring drone with a distance accuracy within 0.5cm and angular orientation accuracy within 2°. Related to the sensor accuracy, the noise created by the drone's unsteady movements as it flies is another factor to take into account.

An additional technical constraint is power consumption. The design must use adequate batteries and limit power consumption from sensors to enable the drone to fly for at least 20 minutes, depending on the needed applications. The design must also consider the memory capacity for data storage. While the specific quantitative memory requirement depends on the implementation, the controller system will need to store a current map of its environment. It may also need to store past location data, the positions of known obstacles, and data from various on-board sensors.

With regard to pose exchange between drones, the medium needs to be wireless and omnidirectional. The latency needs to be no more than 200ms to ensure that the drones do not collide in case the other sensors fail. The medium also needs to be robust and have little interference during data transmission to prevent significant packet loss.

The design should perform adequately across various environmental conditions. For instance, preliminary testing revealed that the on-board magnetometers do not work well inside the lab space, due to interference from electrical equipment. A localization system that relied heavily on the magnetometers would not meet the constraint to perform in indoor environments. An ideal design should perform in poor visibility, high altitude, poor weather, narrow spaces, in the presence of moving obstacles, and other challenging environments.

The average network latency experienced at the NYUAD Kinesis laboratory from the motion capture system using the Robot Operating System (ROS) is around 40ms. An ideal design will mitigate this latency by using the loopback address for the ROS master and only publish requisite data like pose estimates. Images and other large media would increase the latency of the network and thus will not be transmitted between two external nodes.

2.4.2 Non-Technical Constraints

One non-technical constraint is the cost. The total budget allocated for this team is $7,500 ($3 \times \$2500$). The equipment typically used for swarm robotics can be costly. For instance,

Velodyne LiDAR costs approximately $4,000, and Intel's NUC costs roughly $1,000 (please refer to *Background Research* and *Bill of Materials* for more details about each device).

Another non-technical constraint is the legal restrictions surrounding drone operation and imaging of public areas. While this does not directly affect our project when it is being tested in an isolated lab setting and within the NYUAD premises where it is permissible to fly drones, the long-term goal is to use this localization and mapping system in other environments. As such, our design and testing process must comply with local regulations related to the types of drones that can fly, which areas they can fly in, what images can be taken in public areas, and who can access the data collected by the drones. Information access is also related to a similar non-technical constraint, which is the ethical concerns surrounding privacy. Images obtained by the drone's sensors, as well any maps generated for the localization process, should be stored securely, and access to this data should comply with local regulations.

Safety concerns are another non-technical constraint for the drone control system. While personal protective equipment can partially mitigate this risk, the design must account for potentially hazardous scenarios like battery depletion, unexpected collisions with obstacles, and hardware failure. All mounting devices for the sensors should be designed such that the sensors are attached securely and are not at risk of falling off. The safety of our design should be optimized and verified during the testing and evaluation phase.

3 Conceptualization

3.1 Background Research & Key Terminology

The drone-swarm structure shown in Figure 3.1 shows the given input and the desired output of our drone control system. This design aims to utilize input signal from each of the sensors mounted on the drones: Velodyne lidar's raw point cloud data, measurements from the native IMU (linear acceleration/angular velocities in three axes), and Ricoh-V's 360° camera's field of view to detect fiducial markers on neighboring drones and estimate the relative distance between neighboring drones. The filter system is designed to remove noise from the raw data. Next, we develop modules that convert these input signals and determine from them the position and orientation of both the drones and obstacles along the way, and other modules that map the environment of the drone, all of which help in the simultaneous localization and mapping of the drone swarm within the given environment.

A collaborative control approach that is designed to be generalized across different environmental scenarios, like the one we are proposing, is yet to be implemented effectively. Traditionally, although collaborative control problems have used sensing methods such as GPS technology, this does not work at the Kinesis lab, which is indoors. Some other constraints, such as the inability to use the drone's magnetometer in the Kinesis lab (see *Technical Constraints*), make our project innovative, because it provides an alternative to traditional approaches. Furthermore, specific constraints such as latency over wireless networks, uncertainty of data measurements, and the range of camera measurements, make it different from other swarm robotics control designs (see *Design Constraints*) [Alexis et al. (2014), Arvanitakis

et al. (2018), Tzes et al. (2018)].

The following paragraphs describe the key concepts and terms used in the design project, briefly described for better cross-referencing of the concepts used throughout the report:

- Kinesis Lab

The Kinesis Lab is a CTP-facility at New York University Abu Dhabi that provides resources for experimentation and motion capture of drones and robotic equipment. This project uses the flying arena as a testing environment.

Figure 3.1: NYUAD's Kinesis Lab Flying Arena [NYUAD (2018)]

- Voronoi Partitioning

Voronoi partitioning is a technique to compute optimal coverage of a region by dividing it into a given number of cells [Gusrialdi et al. (2008)]. A sensor's Voronoi region is the set of points that are closer to it than to any other sensor. One of the goals of this project is to develop a control law for the drone swarm based on the Voronoi partitioning of the space.

- Robot Operating System (ROS)

The Robot Operating System (ROS) is a set of tools, libraries, and conventions that provide a framework for writing robot software. It is a distributed, modular system that works for a variety of robotics applications. ROS has a communication system for data transmission that involves publishing and subscribing to nodes in a system.

- TurtleBot3

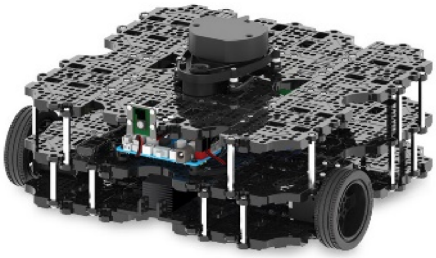


Figure 3.2: TurtleBot3, Waffle Pi Model

The TurtleBot3 is a standard robot that uses the ROS platform. It is notable for its applications in education and research because of its small size, and functional simplicity. It is also easily customizable and presents in three different configuration models: Waffle, Waffle Pi, and Burger. These model names correspond to the shape of the robot. For purposes of testing the robot swarm control algorithms in two dimensions, the TurtleBot Version 3 Waffle Pi model is used in this project. The core technologies implemented in the TurtleBot3 are SLAM, navigation, and manipulation. The TurtleBot3 uses a LiDAR sensor, mounted on the top, for distance sensing. It also features internal accelerometer and gyroscope sensors to track its position and orientation as it moves. The image in Fig. 3.2 illustrates the TurtleBot3 model used in this project.

- Intel® Nuc

The Intel® Nuc is the mini computer used in this project for both image processing and machine learning . It is a 4x4-inch board that supports up to 32GB of RAM and and 8th Gen i7 processor (8M Cache, up to 4.20 GHz). It has an integrated graphics card, SSD, integrated LAN, four USB 3.0 ports and a PCI Express Revision Gen 3/M.2 Card Slot for expansion.

Figure 3.3: Intel® Nuc Board NUC7I7DNBE

- 3D-LiDAR

LiDAR (Light Detection and Ranging) is a sensor that uses laser to measure distances. The LiDAR sensor on the TurtleBot3 Waffle Pi, for example, is a mechanical sensor placed at the

top, which rotates rapidly and gives proximity data of objects in the form of a point cloud. This feature is useful for us because it enables us to obtain precise location data in real time. This project will eventually use the Velodyne Puck Lite, shown in Fig. 3.4, which supports 16 channels and generates approximately 300,000 points/second from a 360° horizontal field of view and a 30° vertical field of view (±15° from the horizon) at distances up to 100m.

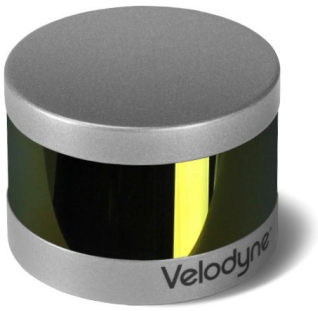


Figure 3.4: Velodyne LiDAR Puck Lite

- Gapter Drone

The Gapter Drone [EDU], manufactured by Gaitech EDU, supports the Robot Operating System (ROS), and is based on the Pixhawk autopilot software [Pixhawk] (an open-source hardware autopilot project), with sensors such as GPS and optical flow. It can also be connected to WiFi. We will be using this drone for testing purposes andour algorithms will be executed on an onboard Odroid XU-4 minicomputer manufactured by Hardkernel.

Figure 3.5: Gapter drone

- Fiducial Markers

Aruco's fiducial markers are composed of an external black border and an internal region that encodes a binary pattern [Aruco; Romero-Ramirez et al. (2018)]. These markers will be placed at the faces of a rhombicuboctahedron structure which is to be attached at each drone. Using the four corners of the marker, camera pose estimation can be performed for relative localization of the drone [Romero-Ramirez et al. (2018)]. In this project, we use the Aruco

library that contains various dictionaries of fiducial markers that are detected effectively and very quickly [Aruco].

Figure 3.6: Aruco markers placed in a Rhombicuboctahedron solid

- Vicon Motion Capture System

This is an array of cameras, built by Vicon, which have been positioned in the closed space at NYUAD's Kinesis Lab, can detect objects labelled with spheres that the Vicon cameras. It can collect position data for objects within the flying arena at a frequency of 100 frames per second with a sub-millimeter accuracy. It is used to verify the implemented trajectory of the TurtleBot/Gapter drones.

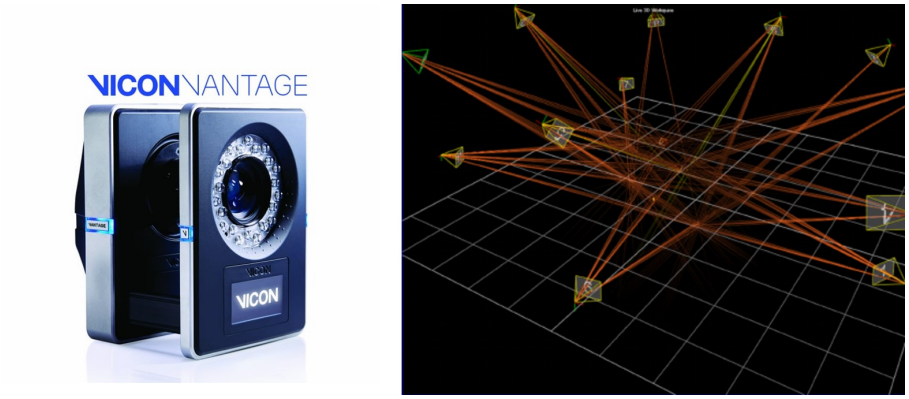


Figure 3.7: Vicon Motion Capture System

- Spherical Camera

The Ricoh Theta V uses a 1/2.3 CMOS Image sensor to capture internally stitched image frames from its two 12 megapixel cameras [Theta]. In this project, the frames are streamed at 30fps HD via USB with TCP/IP. An ad-hoc connection is made between the Intel® Nuc and the Ricoh Theta. Image processing is done in Robot Operating System (https://www.ros.org/) using OpenCV (https://opencv.org/) and OpenGL (https://opengl.org/) (see Appendix for code developed).

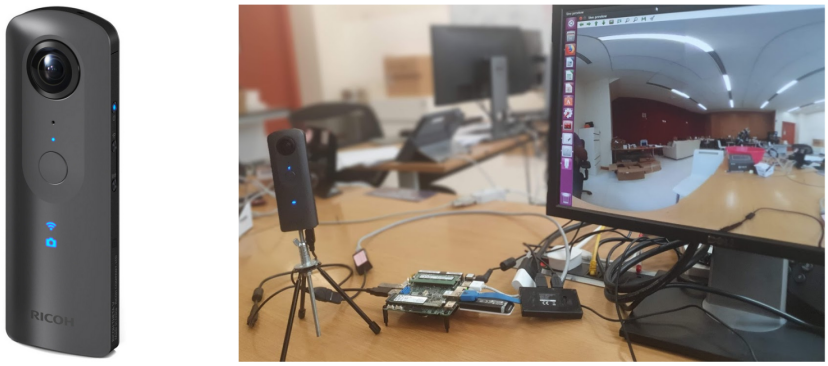


Figure 3.8: Ricoh Theta V and ad-hoc connection with Intel® Nuc in ROS

- Wireless Router

TP-Link's TL-WR802N Wireless N Nano Router has a bandwidth of 300Mbps, and can be configured as a router, repeater, client or as an access point [TP-Link Technologies Co.]. In this project, two of these routers are used. One was used on both the drones: in one as an access point and in one as a client. These routers produce an omnidirectional wireless signal so they were chosen to facilitate good connection irrespective of the drone's spatial location. The router that was configured as a client was connected to the access point and a static IP address was assigned to the client. The Service Set Identifier was then hidden for the access point router so the two cards were semi-permanently paired on a wifi ad-hoc network.

Figure 3.9: TP-Link's TL-WR802N Wireless N Nano Router

- LiDAR

The RPLIDAR A1 has a range of 12 meters at 5.5Hz [Shanghai Slamtec Co.]. It is a 2D 360° omnidirectional LiDAR operating at 8000 samples per second. It uses the laser triangulation ranging principle, high-speed vision acquisition, and processing hardware to make accurate scans.

Figure 3.10: RPLIDAR A1

The first half of this project, Fall 2019, focused on using the ground robot TurtleBot3 Waffle Pi, using fiducial markers while planning its trajectory confined to a 2D plane. Once the results were obtained on this 2D plane, in Spring 2020, we moved to GaiTech's Gapter drones, which added an extra z-dimension to our system. The TurtleBot3, which is originally equipped with a Raspberry Pi, was replaced by an Intel® NUC, which gave us higher processing power and a compact, portable design that could be easily transported around on the back of a TurtleBot.

3.2 Concept Generation with Morphological Chart

A morphological chart is a useful tool for refining the project concept. In this case, we identify different solutions for the sub-tasks in the project. This helps us explore the problem further. The morphological chart is pictured below.

Solutions	Sub-Problem 1.0	Sub-Problem 2.0	Sub-Problem 3.0
1	Use spherical cameras	Use Computer Vision	Understand drone piloting and how to use the fly cage
2	Pre-process the images	Use Optical-Flow-Based Variable Object Tracking	Use MATLAB or other software to understand Voronoi
3	Try different techniques to clean the training data	Use Vicon System for verification	Evaluate and test the control system
Sub-Problem 1.0: Capture 360 images and pre-process into rectangular sections to optimize for feature detection			
Sub-Problem 2.0: Use computer vision and optical-flow-based variable object tracking techniques to estimate drone position and distance			
Sub-Problem 3.0: Understand Voronoi partitioning and control systems to optimize swarm behavior.			

Table 3.1: Project's Morphological Chart

3.3 Concept Selection with Pugh Chart

A Pugh chart helps us to better understand our project by weighing the relative impact of different considerations, such as the cost, difficult of implementation, speed, and effectiveness of the method. The following chart is an example of how these factors were weighed during the concept selection process.

Localization Concept	Effectiveness	Speed	Cost	Difficulty	Sum
Computer Vision	0	0	0	0	0
Optical-Flow-Based Variable Object Tracking	1	1	0	0	2
GPS	-1	0	-1	1	-1
Legend: -1 indicates "worse", 0 indicates "default", 1 indicates "better"					

Table 3.2: Pugh Chart

4 Modeling, Simulation, and Optimization Plan

4.1 Trajectory Planning

First, path planning was modeled using simulated Turtlebots in RViz, a built-in visualization tool in ROS. A Python program was written to allow the user to specify several waypoints in the form of (x, y) coordinates. The Turtlebot then traveled to those points one by one. This simulation helped us to understand how to use ROS to direct a robot to navigate to certain points within a given map, which is an important part of the collaborative control system.

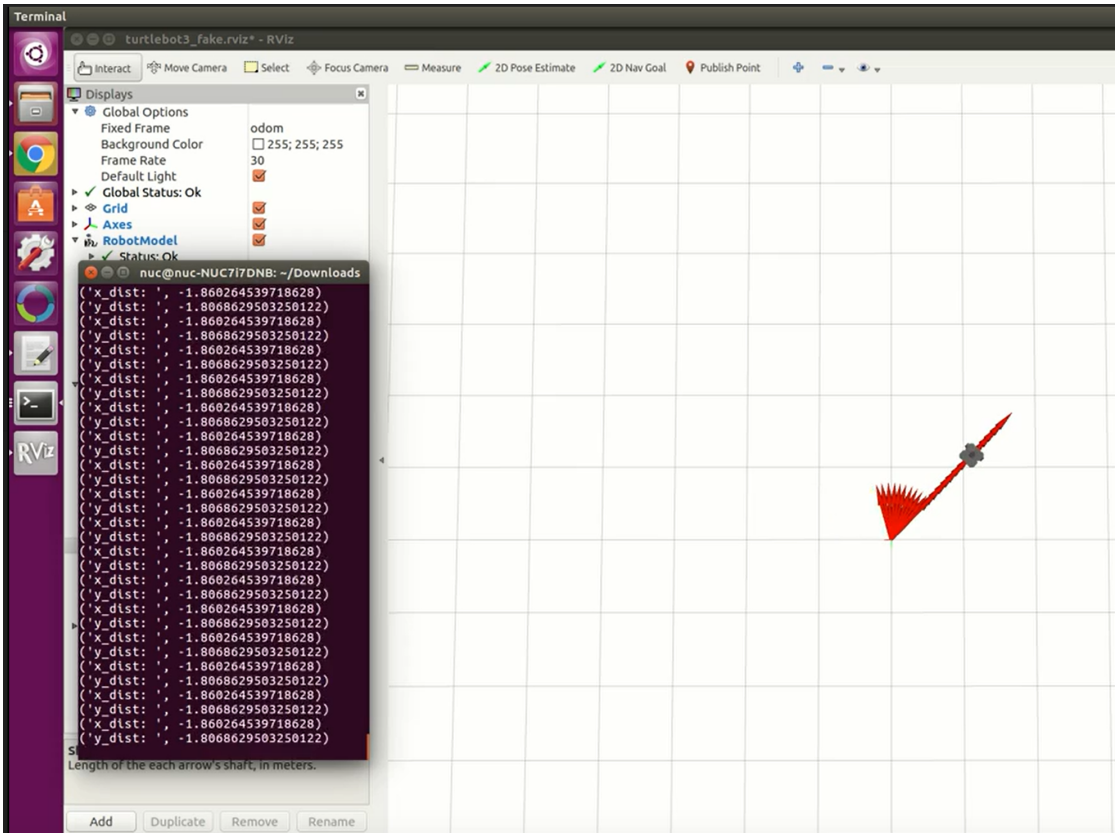
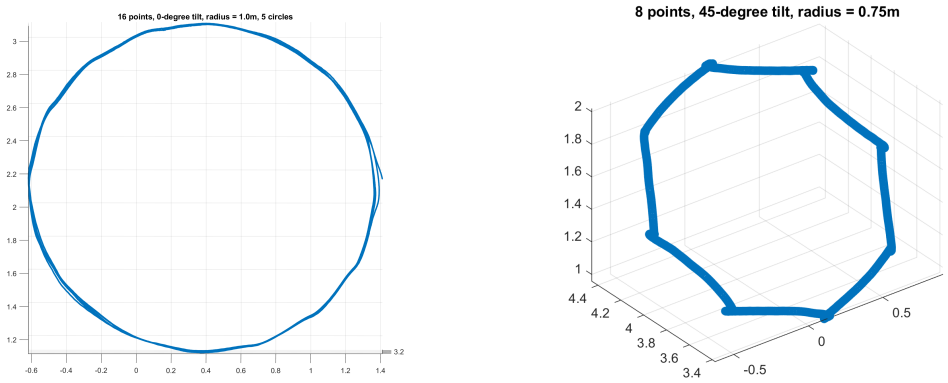


Figure 4.1: Path Planning Simulation Using Turtlebot in RViz

The next step was to implement trajectory planning in three dimensions. ROS-enabled Gapter EDU drones enable us to program the desired trajectory autonomously with the help of telemetry devices, which allows for the remote transmission of data to the drone. This telemetry radio is configured using the *Ardupilot Mission Planner*, a ground control station that provides setup and flying support. After this connection is established, the drone is placed inside the flying arena surrounded by the *Vicon Camera System*, and the Odroid – a single board computer inside the drone – is accessed through a remote command line via the Secure Shell protocol. The *Vicon Camera System* with its multi-precision cameras further tracks the different motion capture markers placed on the drone, which increases the accuracy of the drones when flying to the specified way-points in the trajectory. The magnetometer is turned off due to inaccuracies in its use due to interfering waves at the lab, while the accelerometers and gyroscopes are turned on.

After calibrating the drone and setting up the necessary equipment, the drone is specified to travel in a circle with varying radius, number of way-points, and tilt angles. The following figures demonstrate the trajectory of sample circles with varying number of waypoints, radius, and tilt angle. Note that this trajectory data is recorded through a separate script using the *Vicon camera system*. The drone is connected to an RC controller, which overrides any ROS commands and can be used for manual control in case the drone goes off trajectory for several reasons (improper camera calibrations, faulty code or unexpected execution times).

The following figure demonstrates two sample circular trajectories from this experiment.



(a) Circular Trajectory, 0° Tilt

(b) Circular Trajectory, 45° Tilt

Figure 4.2: Drone's Trajectory Planning Trials

Please refer to *Implementation Details* for further explanation of these results.

4.2 Swarm-Drone Wireless Network

An experiment was run to validate the ad-hoc system by flying the two drones and running the pose exchange nodes designed. The results after flying for about two minutes showed

that there was no packet loss as can be seen by the multiple perfectly aligned ad-hoc and Vicon measurements (Fig. 4.3 and Fig. 4.4). However, there were about 120 duplicate packets detected in the data out of 14522 packets. This packet duplication is still being investigated although it does not violate our technical constraints since the average latency is about 0.01ms and no more than 3 duplicates were detected for a unique pose. Therefore for three duplicate packets, the delay is about 0.04ms which is still within the technical constraints of 200ms. The packet duplication was due to the fact that we were sending coordinates at a set frequency which was higher than the frequency of the Vicon's transmission. To mitigate this, we modified the code to only send coordinates via the ad-hoc channel when it received coordinates from the Vicon.

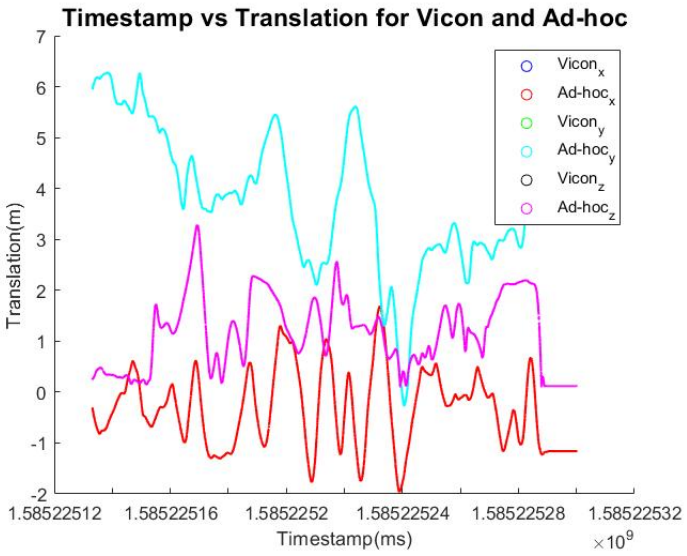


Figure 4.3: Timestamp vs Translation for Vicon and Ad-hoc

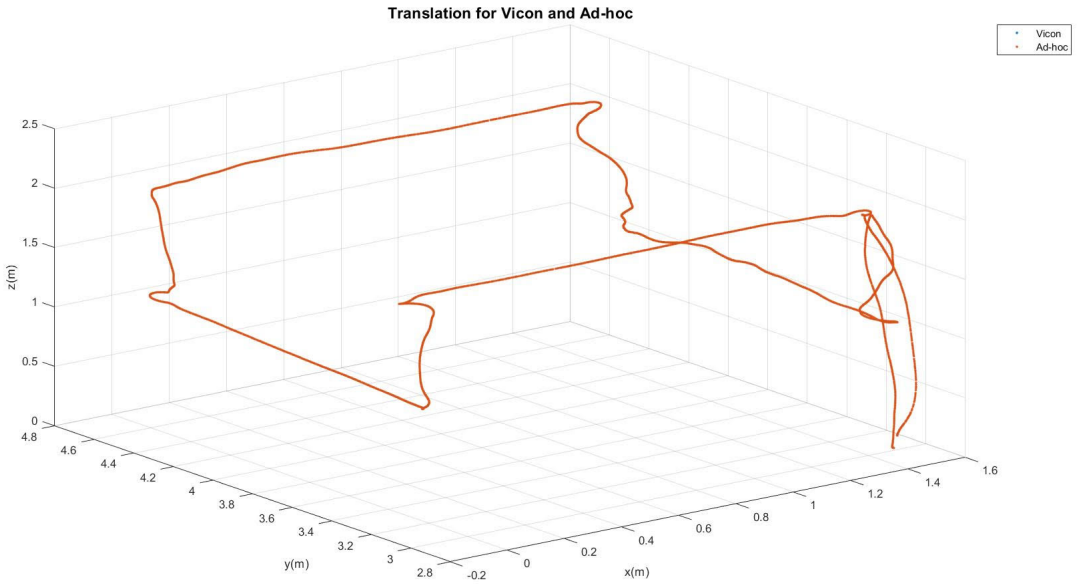


Figure 4.4: Translation for Vicon and Ad-hoc

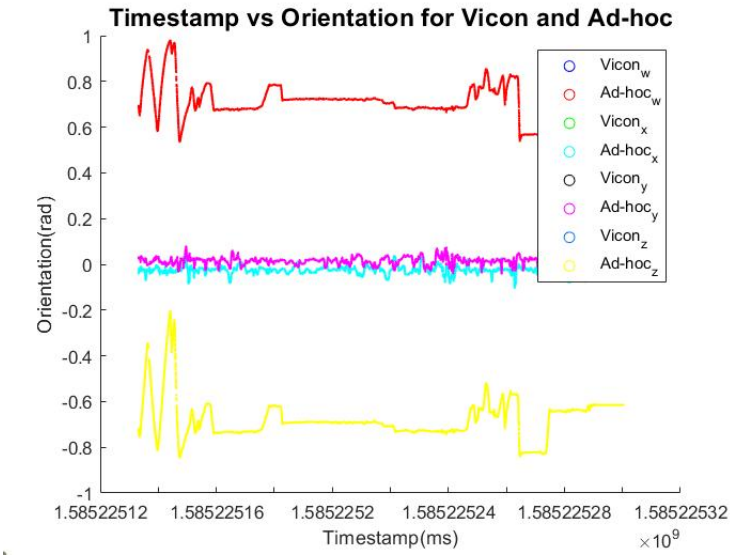
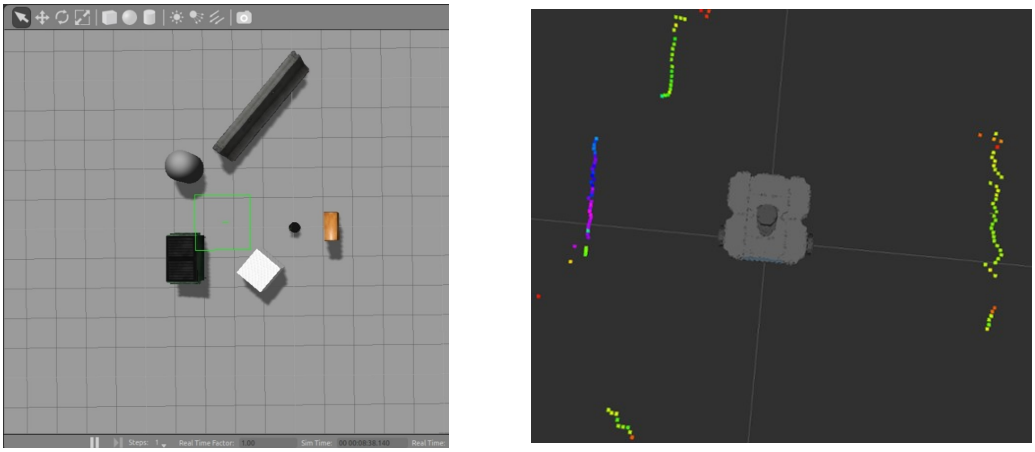


Figure 4.5: Timestamp vs Orientation for Vicon and Ad-hoc

4.3 Virtual Simulations

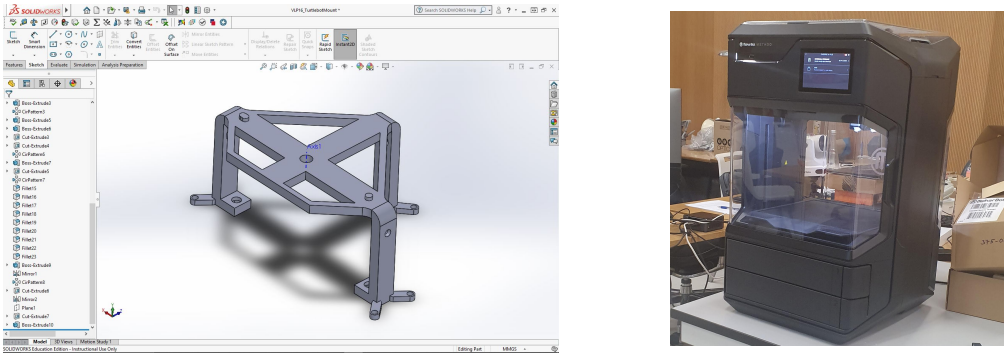
Robot simulation is essential to our project as there is a high risk of accidents when working with drones. For this purpose, we employ the robot simulator, Gazebo. Gazebo is a physics engine that makes it possible to design robots, and accurately simulate populations of robots in complex indoor and outdoor environments. We have used Gazebo so far with the Turtlebot but are yet to use it with the Gapter drone. The plan is to design the environments of the drone with Gazebo before conducting actual experiments in the Kinesis lab to discover likely issues. We also employ ROS Visualization that helps us visualize the data Gazebo is generating with its physics engine. With regard to modeling, we use SolidWorks 2019 to design all our 3D models and print them with the Makerbot 3D printer in the Robotics laboratory.



(a) Gazebo simulator

(b) ROS Visualization tool

Figure 4.6: Simulation software



(a) LiDAR base in Solidworks 2019

(b) Makerbot 3D printer

Figure 4.7: Modeling tools & 3D-printed components & printers

4.4 Optimization Performed

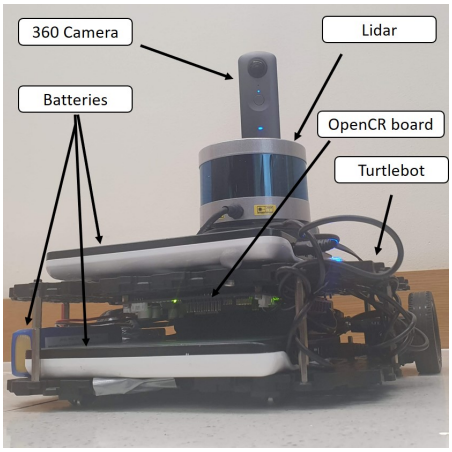
Conducting these simulations allowed us to optimize the trajectory planning and area coverage code for the later tests using the Turtlebot and Gapter drones. They allowed us to explore the best algorithms and tune necessary parameters before implementing them in hardware. The simulations reinforced many of the design choices, such as the selection of the Hector SLAM algorithm for use with the LiDAR. Debugging the code using simulations was helpful before testing the code on the ground vehicles and UAVs, because the simulated robots were not vulnerable to calibration issues and equipment failure. A number of CAD models also allowed for customized 3D structures to be printed and attached to the drones, such as the wire covers and Vicon detection marker mounts. Visualizing and testing the project in simulations allowed us to refine the design.

5 Final Design

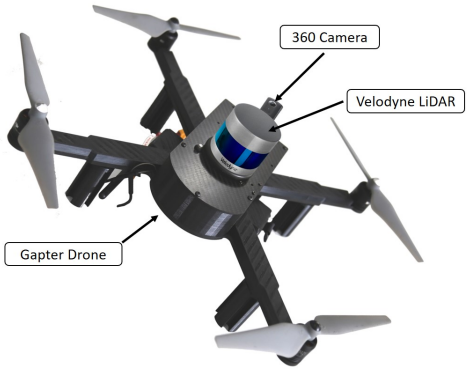
5.1 Initial Design Proposed

The preliminary ground vehicle design, shown in Figure 5.1.a, is composed of a Velodyne LiDAR and a Ricoh Theta V mounted unto a TurtleBot3. The Turtlebot3 is composed of two Dynamixel motors, an OpenCR board and three batteries. The blue 12 V LiPo battery powers the OpenCR board and the Dynamixel motors. The white 19 V battery in the lower deck powers the Intel Nuc that runs the ROS nodes and the white 12 V battery in the upper deck powers the Velodyne LiDAR. The Turtlebot is controlled remotely via Secure Shell and Virtual Networking Computing.

The preliminary drone design, shown in Figure 5.1.b, is composed of a Velodyne LiDAR and a Ricoh Theta V mounted unto a Gapter drone. The Gapter drone is composed of the PixHawk flight controller, Gyroscope, Barometer, 3D accelerometer, GPS, Optical Flow and the Odroid XU4 board. The Velodyne LiDAR, Ricoh Theta V and Odroid board are powered from a Lipo Battery (3S, 4S) with an adjustable power-supply module.



(a) Current design with Turtlebot

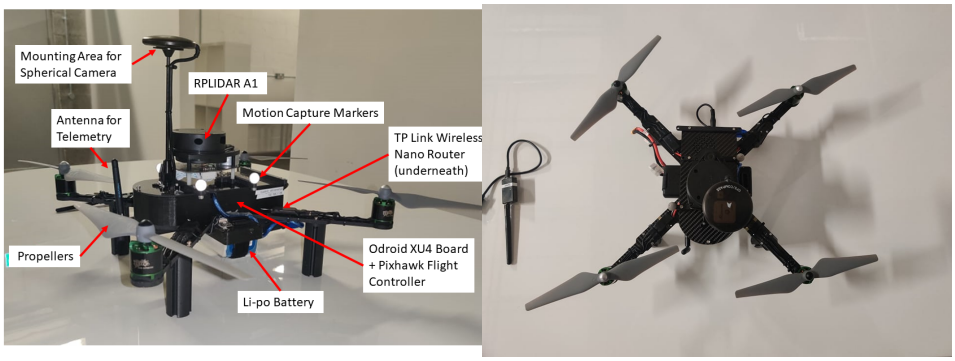


(b) Rendering of Drone Design

Figure 5.1: Initial Design Expected

5.2 Changes Made to Initial Design

One key change made to the initial design is the LiDAR sensor used. The original Velodyne LiDAR exceeded the load bearing limit of the Gapter drone, and so a smaller 2D RPLIDAR A1 device was used instead. The images below display the current implementation with the LiDAR mounted on top of the Gapter drone. It also has a compass mounted in place of the 360 camera, although this compass is currently disabled and is not used in the final design.



(a) Side View with Labeled Components

(b) Top View

Figure 5.2: Current Design with Gapter Drone

6 BUDGET

The following bill demonstrates the cost of all equipment that we require for this design project.

S.N.	Part Name	Function	Specification/Model	Qty	Cost (USD)	Webpage
1	Turtlebot3® Waffle Pi	Conceptualizes the problem in 2D space. Drones are an extension of the concept in 3D space.	Standard Model	2	1,399	http://www.robotis.us/turtlebot-3-waffle-pi/
2	Gapter EDU Copter	Extension of the 2D design to 3D space. Final design implementation includes collaboration between these two drones.	Standard Model	2	1,000	https://discourse.ros.org/t/gapter-edu-pre-order-offer/2108
3	Intel® NUC	CPU that powers the Turtlebot. Raspberry Pi, which comes out of the box, is replaced with this CPU for better computation.	Intel Core i7-8559U, 32GB RAM, 1TB ROM, Intel Coffeelake HD Graphics	2	701.4	https://www.intel.com/content/www/us/en/products/boards-kits/nuc.html
4	Ricoh Theta V	Spherical 360 camera for identifying fiducial markers in neighboring drones	Standard Model (30 fps @ 3840 x 1920)	2	379.95	https://theta360.com/en/about/theta/v.html
5	Velodyne lidar Puck	For Simultaneous Localization and Mapping of the drones	VLP-16	2	4,000	https://velodynelidar.com/vlp-16.html
				Total:	14,961	*Prices may vary based on region and time.*

Table 6.1: Bill of Materials

Note that testing/manufacturing systems such as Vicon camera system (€500,000/$552,937), 3D printers (varies greatly by the type of model used, from $200 to several thousand dollars), and easily available items such as the fiducial markers, are not included.

7 Implementation Details

7.1 Description of Implementation

This project implements a sensor system and control mechanism that allows a drone swarm to map the environment, detect obstacles, and detect the relative translation and orientation of other drones within the swarm. The drones can also communicate directly using wireless ad hoc networks. As a preliminary step, SLAM and trajectory planning were first implemented on a Turtlebot3 Waffle Pi (a ground-based vehicle) to visualize the problem in 2D. An overview of the 3D implementation using Gapter drones, currently in progress, is given in *Final Design*. A labeled image of the current design is included again below for reference:

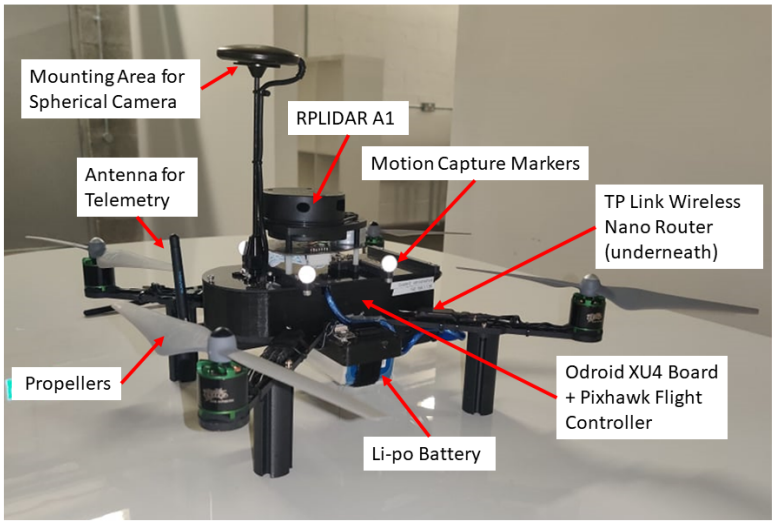


Figure 7.1: Gapter Design with Labeled Components

Ardupilot Mission Planner is the command center for drone calibration and reception of telemetry data. Specifically, the Ardupilot interface allows us to tune PID controller parameters, calibrate the accelerometer, compass, and radio controller. It also allows us to monitor the drone mode and status during flight and whether position data is being transmitted properly.

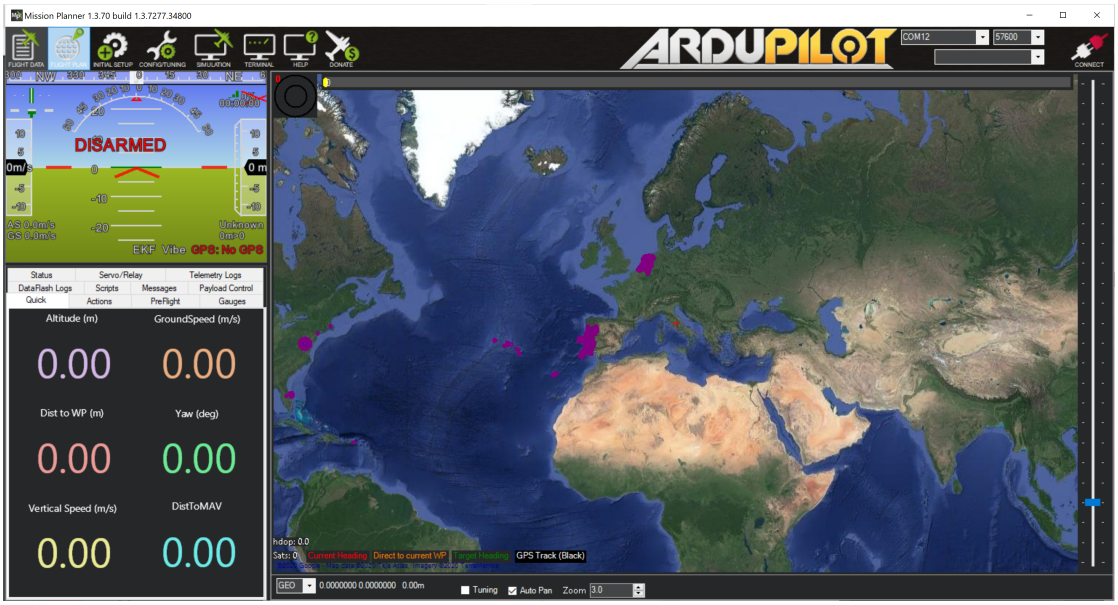


Figure 7.2: Ardupilot Mission Planner Interface

Four 3D-printed legs were chosen and attached to the drone frame using bolts. The legs stabilize the drone for takeoff and allow it to land without damaging the Li-po battery strapped to the underside of the drone. 3D-printed wire covers also cover the motor controllers and components on the arms of the drone. DJI Original 9-inch Carbon Fiber Reinforced self-tightening propellers were chosen for their size, lightweight design, and reliable performance. The TP link device and battery are mounted underneath the drone. The Odroid onboard computer and Pixhawk flight controller are secured inside the body of the drone. The telemetry antenna protrudes from the nose (front) of the drone.

The areas for mounting the spherical camera and LiDAR are located on top of the drone in order to give adequate viewing range for the sensors. In particular, the spherical camera is mounted on a stick to minimize the drone's obstruction of the spherical image. More details regarding the ad-hoc network and rectification of spherical images are given in later in this section, in *Initial Results*.

7.2 Voronoi-tessellation of rectangular area

Consider the rectangular area Ω of the Robotics/CTP-Kinesis lab, shown in Figure 7.4. The four vertices defining this area are $u_i \in U = \{(0,0),(l,0),(l,w),(0,w)\}$. Two UASs are located at $p_i \in \left[(x_i, y_i)\right]$, $i = 1,2$. These UASs are assumed to have the same altitude z, as shown in Figure 7.3.

The rectangular area is tessellated into two disjoint Voronoi sets $V_i : p_i \in V_i$, $V_1 \cap V_2 = \emptyset$ and $V_1 \cup V_2 = \Omega$. The attributes of each set is $\forall p \in V_i, \parallel p - p_i \parallel \geq \parallel p - p_j \parallel$, $i, j \in 1,2$. Each border of the set is defined by four vertices, of which two correspond to members of U. As

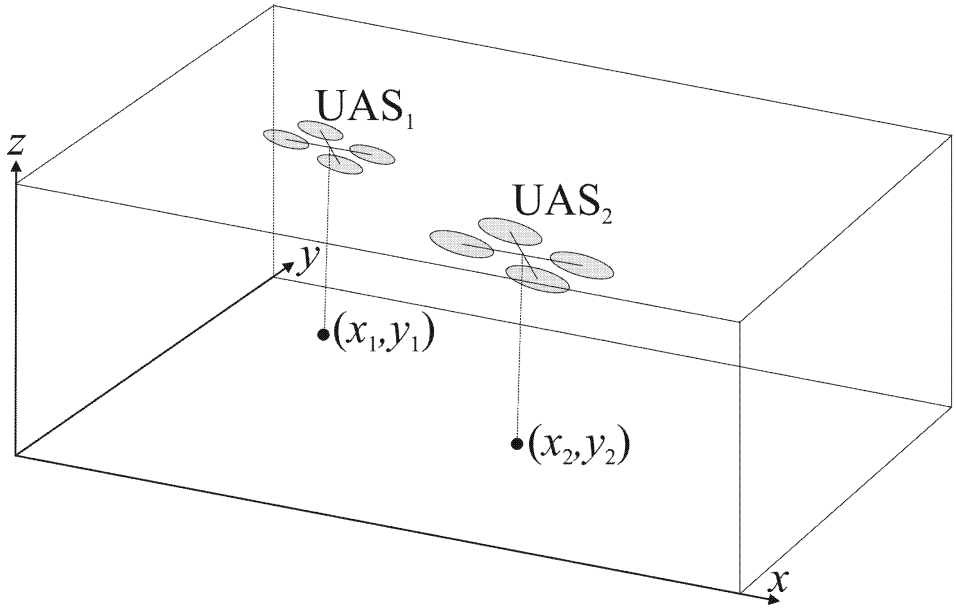


Figure 7.3: Visual Coverage using two UASs

an example, V_1 is defined as $V_1 = \text{co}\left(u_4, \left(x_1, y_1\right), \left(x_2, y_2\right), u_1\right)$, where 'co' corresponds to the convex hull operator.

To compute these two vertices for the ith RoR, the distances $\parallel x_i - u_j \parallel,\ j = 1,\ldots,4$ are computed and ranked in an ascending manner $\parallel x_i - \tilde{u}_i^j \parallel,\ j = 1,\ldots,4$. In our case, for $i = 1$, $\tilde{u}_1^1 = u_4, \tilde{u}_1^2 = u_1, \tilde{u}_1^3 = u_3, \tilde{u}_1^4 = u_2$, since $\parallel x_1 - u_4 \parallel \leq \parallel x_1 - u_1 \parallel \leq \parallel x_1 - u_3 \parallel \leq \parallel x_1 - u_2 \parallel$.

Then for the ith Voronoi cell two out of its four vertices correspond to the first two vertices (say for $V_1 : \tilde{u}_1^1 = u_4$ and $\tilde{u}_1^2 = u_1$). The remaining two vertices correspond to the points where the perpendicular bisector of the vector $\left[x_2 - x_1, y_2 - y_1\right]$ intersects the edges defined by $\tilde{u}_1$ and $\tilde{u}_2$ and their Delaunay border neighbors $\tilde{u}_3$ or $\tilde{u}_4$.

The Delaunay neighbor to $\tilde{u}_1$ is defined as

$$\tilde{u}_{1d} = \tilde{u}_{j^*} : j^* = arg \min_{j=3,4} \parallel \tilde{u}_1 - \tilde{u}_j \parallel , \tag{7.1}$$

where in our example $\tilde{u}_{1d} = \tilde{u}_3$. The remaining vertex is the Delaunay neighbor of $\tilde{u}_{2d}$; (in our case $\tilde{u}_{2d} = \tilde{u}_4$).

The perpendicular bisector line is defined as

$$y \quad = \quad -\frac{x_2 - x_1}{y_2 - y_1}\left(x - \frac{x_1 + x_2}{2}\right) + \frac{y_1 + y_2}{2} . \tag{7.2}$$

The remaining two vertices are defined as the intersection of this line with the edges defined by their vertices, namely $e_1 = [\tilde{u}_1 - \tilde{u}_{1d}]$ and $e_2 = [\tilde{u}_2 - \tilde{u}_{2d}]$. In our example, for the given

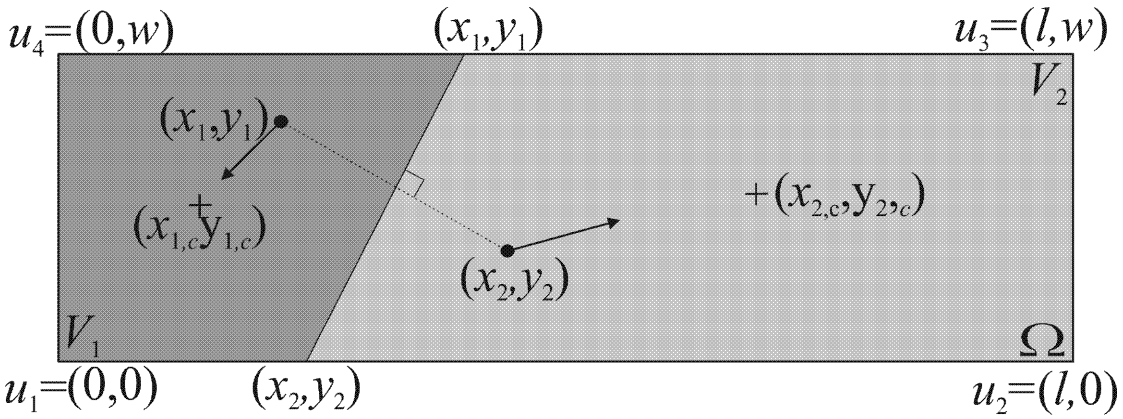


Figure 7.4: Tessellated Area (top view) for Visual Coverage

spatial UAS-configuration

$$\begin{aligned}(x_1,y_1) &= \left(\left(w-\frac{y_1+y_2}{2}\right)\frac{y_2-y_1}{x_1-x_2}+\frac{x_1+x_2}{2},w\right) \text{ and}\\ (x_2,y_2) &= \left(\left(0-\frac{y_1+y_2}{2}\right)\frac{y_2-y_1}{x_1-x_2}+\frac{x_1+x_2}{2},0\right).\end{aligned}$$

Similarly, this should be repeated for the second Voronoi cell V_2.

7.3 Control law development

The centroid of Voronoi cell V_1 defined by $V_1 = \text{co}\left(\tilde{u}_1,\tilde{u}_1^d,\tilde{u}_2^d,\tilde{u}_2\right) = \text{co}\left(x_0,x_1,x_2,x_3\right)$ can be computed as

$$\begin{aligned}x_{1,c} &= \frac{1}{6A}\sum_{i=0}^{n-1}(x_i+x_{i+1})\left(x_iy_{i+1}-x_{i+1}y_i\right),\\ y_{1,c} &= \frac{1}{6A}\sum_{i=0}^{n-1}\left(y_i+y_{i+1}\right)\left(x_iy_{i+1}-x_{i+1}y_i\right),\\ A &= \frac{\|\,\tilde{u}_1-\tilde{u}_1^d\,\|+\|\,\tilde{u}_2-\tilde{u}_2^d\,\|}{2}\|\,\tilde{u}_1-\tilde{u}_2\,\|.\end{aligned}$$

This process should be repeated for $V_2 = \text{co}\left(\tilde{u}_3,\tilde{u}_3^d,\tilde{u}_4^d,\tilde{u}_4\right)$.

Each UAS should move towards the centroid of its RoR, hence the control law for the ith UAS should be

$$v_i = K\frac{\left[\left(x_{i,c}-x_i\right),\left(y_{i,c}-y_i\right)\right]}{\sqrt{\left(x_{i,c}-x_i\right)^2+\left(y_{i,c}-y_i\right)^2}}, \tag{7.3}$$

where the commanded v_i is a normalized vector pointing towards the centroid of the ith RoR, and $K>0$ is a positive gain.

The real-time Voronoi tessellation of the area in NYUAD's Robotics lab using two drones flying at the same height was derived and the area coverage achieved by commanding each drone to move towards the centroid of its assigned Region of Responsibility. This program was written and ready for experimental evaluation, however, due to the pandemic, experimentation was not possible. It was then ported to Java and a simulation was produced in Processing to validate the theory for two drones as shown in Figure 7.5. This sample simulation demonstrates the Voronoi partitioning of two drones, with Drone 1 placed at coordinates (5,2), and Drone 2 placed at (1,4). We observe each drones moving towards the centroids of their region of responsibilities. The arena dimensions are 15m × 5m.

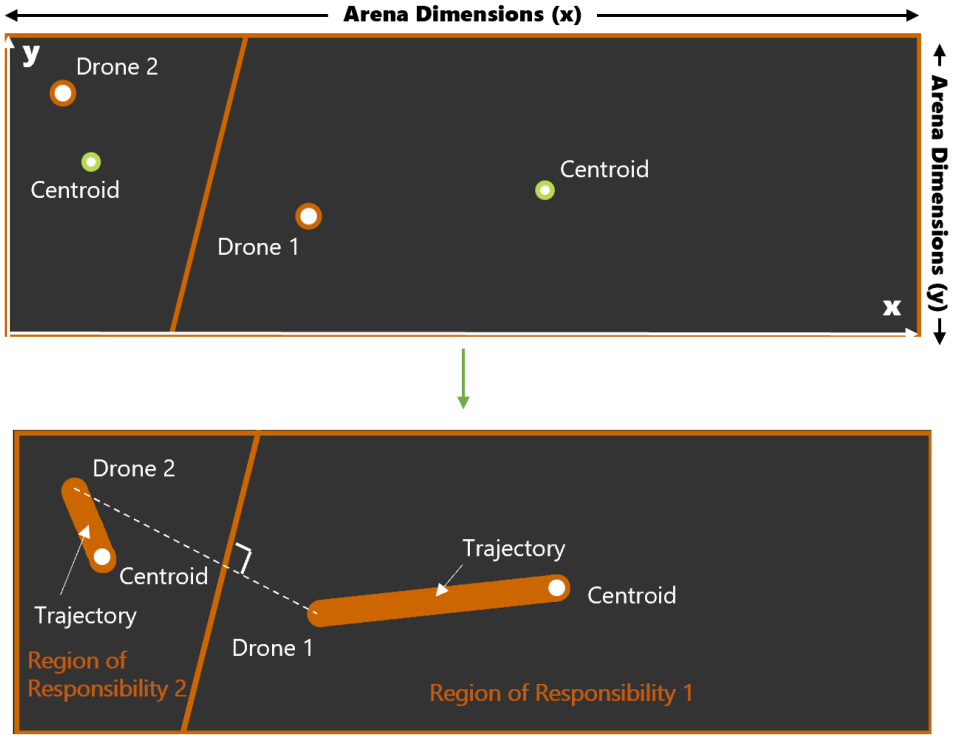


Figure 7.5: Java Simulation of two UAVs moving towards calculated centroid of assigned RoR

7.4 Challenges Faced During Implementation

One of the challenges faced during implementation was the difficulty of calibrating the drones. If the radio controller, compass, accelerometer, and other systems were not properly calibrated, the drone would drift or yaw during flight, which made it difficult to test our control system and pinpoint the source of errors. Unexpected crashes also damaged the drone's legs, propellers, and frame, which caused us to spend time repairing the drone instead of making progress with our experiments. As we gained more experience flying the drone both manually and through ROS, these crashes became less frequent. We also created a document listing the steps to start up, calibrate, and fly the drone, which helped us to overcome this challenge.

In our initial testing with a circular trajectory *see Modeling, Simulation, and Optimization Plan*, the drone follows a trajectory that closely resembles a circle, but there are many factors that prevent it from following an ideal path; firstly, although an increase in the number of way-points does make the trajectory more circular, this also increases the amount of time taken to complete the circle as the drone makes a stop at each way-point. Although best efforts were put to calibrate the *Vicon camera system*, it may not be properly calibrated, which also might have resulted in the improper shape of the circles. At each way-point, the drone hovers around the target location. In our initial testing of the Gapter drone with a calibrated radio controller, we also realized that this drone was more susceptible to position fluctuations in stabilize mode compared to other commercially available drones that we tested, such as the DJI Mavic Pro. This lack of stability makes the task of working with the Gapter drones even more challenging.

Due to the pandemic, the final experiment could not be conducted in the laboratory. Therefore, simulation models were developed in Java as shown in Figure 7.5.

7.5 Changes Made During Implementation

The TP-Link wifi router was mounted underneath the drone to facilitate direct communication between the drones in case the sensor fusion failed and the drones were in close proximity. The details and results are discussed under *Modeling, Simulation and Optimization Plan, Ad-Hoc Network*.

After the legs of the drone broke during crashes, they were temporarily secured using electrical tape and zip ties until the bolts could be repaired, although this is not a permanent change to the design. This did not affect the drone performance in our experiments.

In addition, to reduce payload weight, the RP Lidar was also mounted on top of the drone for 2D SLAM and the details and results are discussed under *Modeling, Simulation and Optimization Plan, SLAM with LiDAR*.

7.6 Initial Results

Preliminary tests of trajectory planning were conducted using one Gapter drone and the Vicon motion capture system. The results of the trajectory planning experiments can be found in *Modeling, Simulation and Optimization Plan, Trajectory Planning*.

Spherical cameras were used to localize rhombicuboctahedron fiducial markers in a 360-degree field of view. Each drone's marker was tracked passively by neighboring drones in the intermediate surrounding to infer relative pose. The image from the 360-degree camera was sent via TCP/IP to the onboard computer. The system then optimally partitioned the surrounding spherical image into 12 partitions and rectified the partitioned images (Figure 7.6). The rectified images are optimally distortionless rectilinear images. With the rectified images that have markers, the pose was estimated using computer vision algorithms and neighboring drones were detected.

Figure 7.6: Partitioned and rectified spherical images

To facilitate direct communication between the drones, an ad-hoc TCP/UDP communication was developed (Figure 7.7). For this setup, two Gapter drones were used. One TP-Link TL-WR802N Wireless N Nano Router was used on both the drones: in one it was used as an access point and in one it was used as a client (Figure 7.8). These routers produced an omnidirectional WiFi signal so they were chosen to facilitate good connection irrespective of the drone's relative location in space. Tests were run to determine which wifi channel had the least interference, and it was found to be channel 6. The access point was therefore using only channel 6. The router that was configured as a client was connected to the router that was configured as an access point and a static IP address was assigned to the client. The *Service Set Identifier* was then hidden for the access point router so the two cards were semi-permanently paired on a wifi ad-hoc network. Two UDP applications were then developed in C++ and integrated into ROS as communication nodes.

Since UDP has no handshake, the two drones conveniently exchanged their coordinates (3 translations and 4 quaternion rotations) at a set frequency of 100Hz. Currently, the two drones are localized using spherical markers and the Vicon system. Therefore, the two drones received their coordinates by connecting to the Vicon system via TCP/IP on one network card and sent the coordinates to each other via the dedicated ad-hoc network. An experiment (Figure 7.10) was conducted to validate the ad-hoc system and it proved to be successful with a latency of 0.01ms as discussed in *Modeling, Simulation, and Optimization Plan, Ad-Hoc Network*.

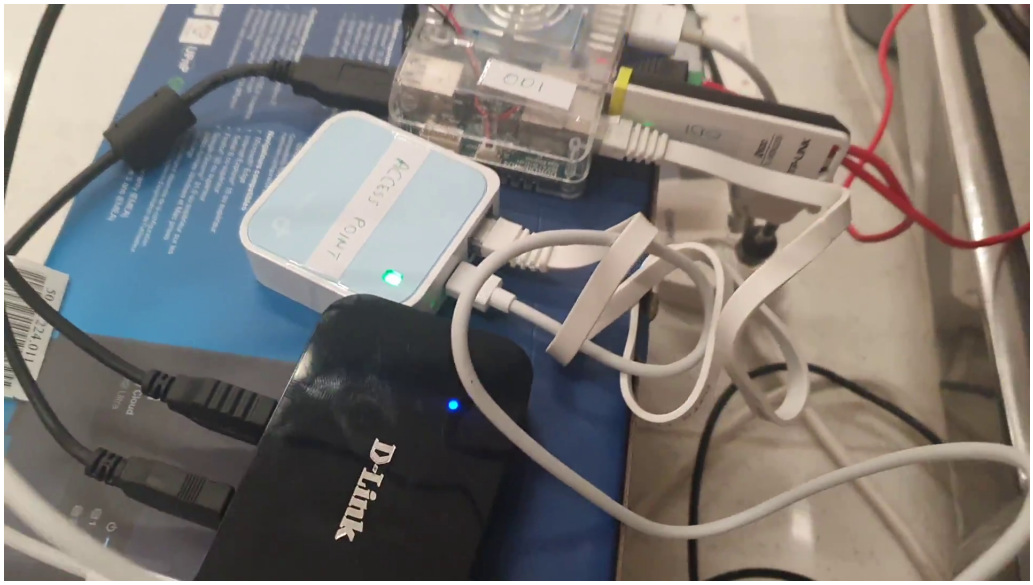


Figure 7.7: TP Link initial setup

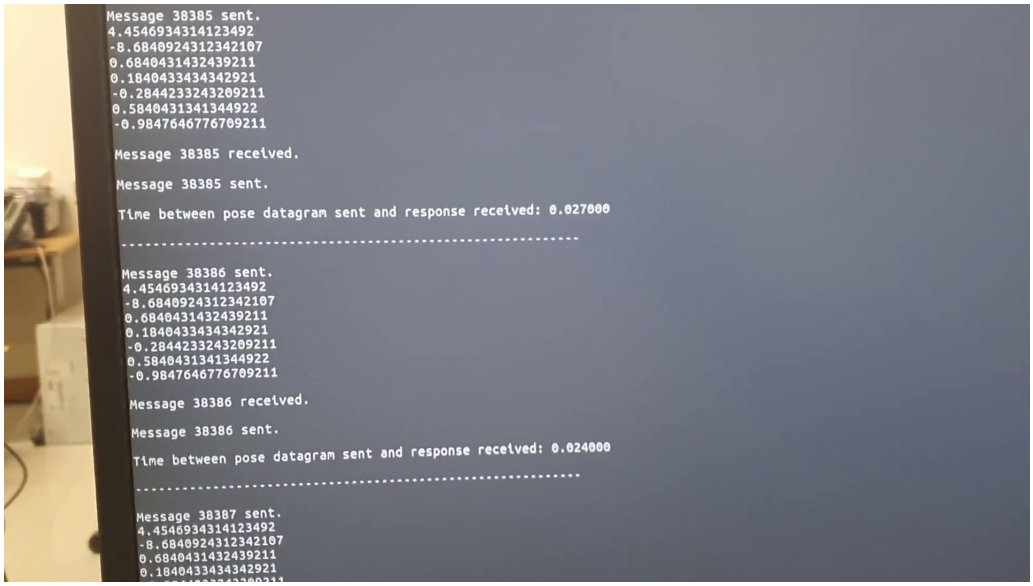


Figure 7.8: UDP/IP application developed

Figure 7.9: TP Link setup on drones

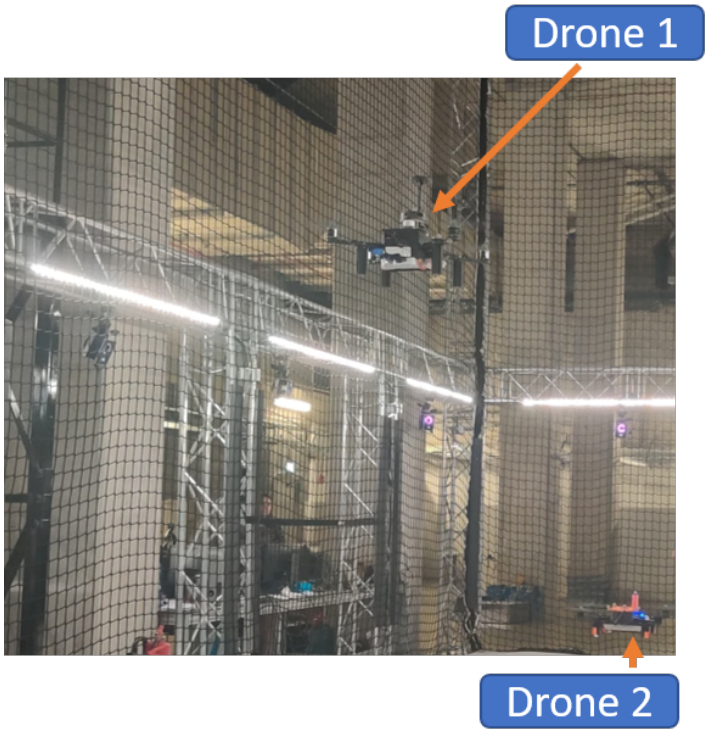


Figure 7.10: Swarm inner-communication experiment

Initially, the Velodyne was mounted on the Turtlebot3 for experiments but to save time during algorithm testing, a simulator was employed. The Velodyne simulator was used in Gazebo and Rviz to simulate the operation of the VLP-16 LiDAR (Figure 7.11). The Velodyne was used for experiments with the trolley setup (Figure 7.13) to try and achieve loop closure and improve the overall accuracy of the map. The SLAM algorithm employed is called Laser Odometry and Mapping. It computes the motion of the LiDAR and incrementally builds the map. As is the case with Hector SLAM (Figure 7.15), it also does not require GPS nor IMU data for the SLAM. We faced a couple of issues with the 3D map like inconsistent map duplications laterally and longitudinally. After many experiments, we found that when we made very abrupt movements or entered tight corners, the algorithm assumed we were in a different location so failed to do loop closure and duplicated the maps instead (Figure 7.14). We also found that the LOAM algorithm relied on the floor in the map construction, so we made the base of the trolley as clear as possible and reduced the height of the entire setup.

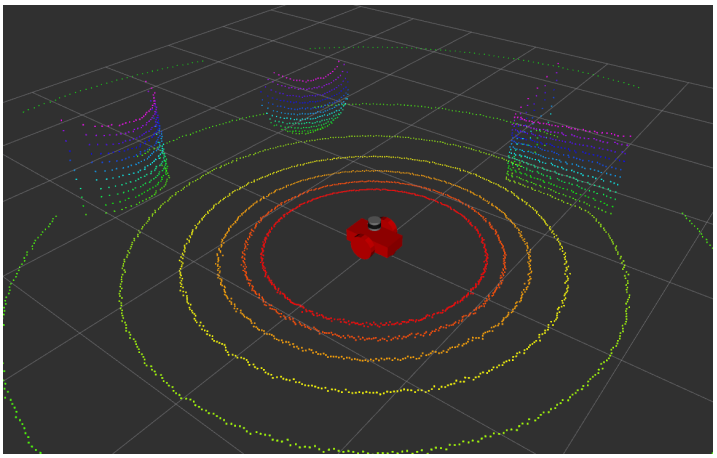


Figure 7.11: Velodyne simulator

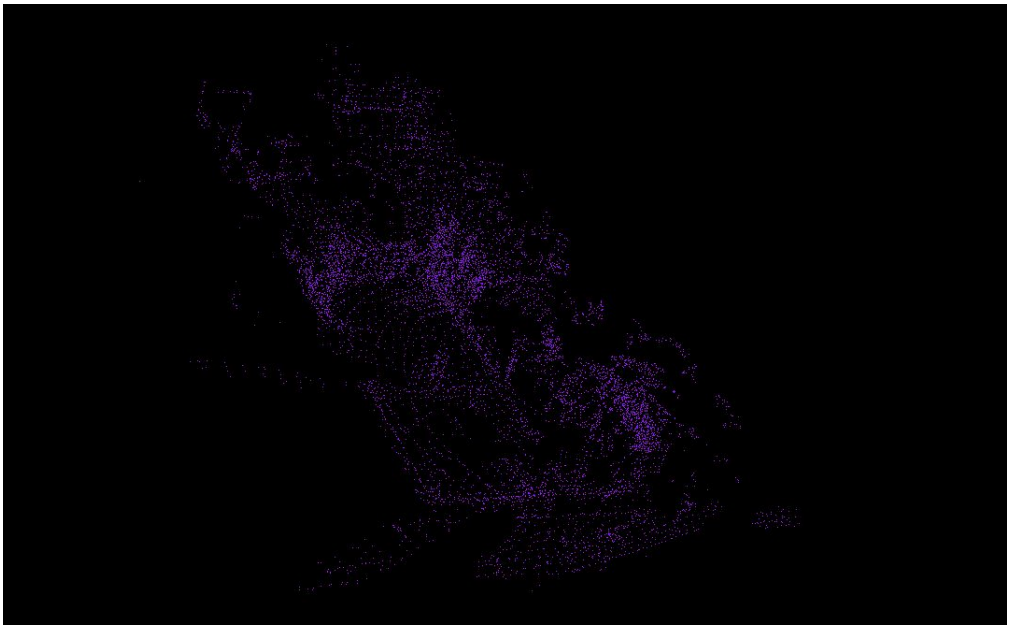


Figure 7.12: Point cloud with loop closure of NYUAD's A1 corridor

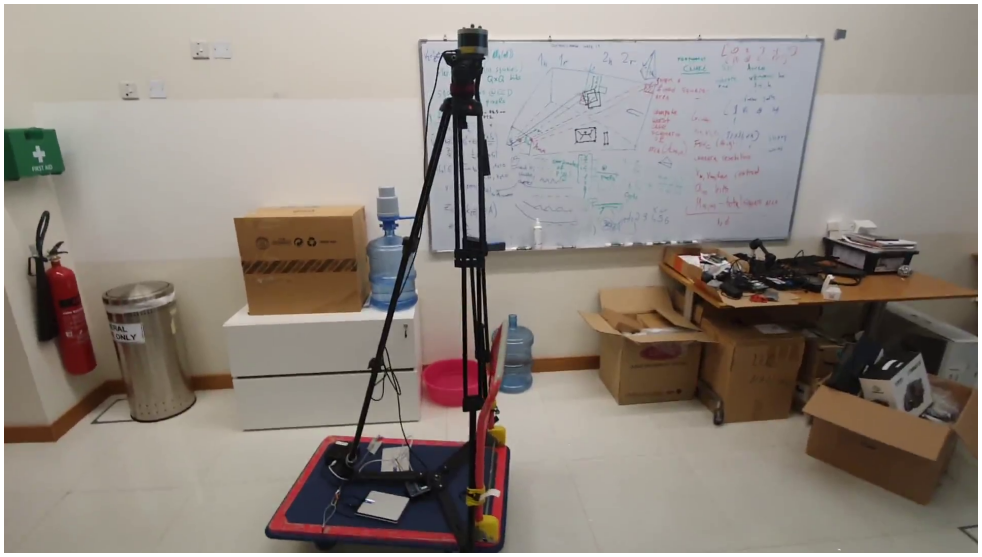


Figure 7.13: Trolley setup for 3D SLAM

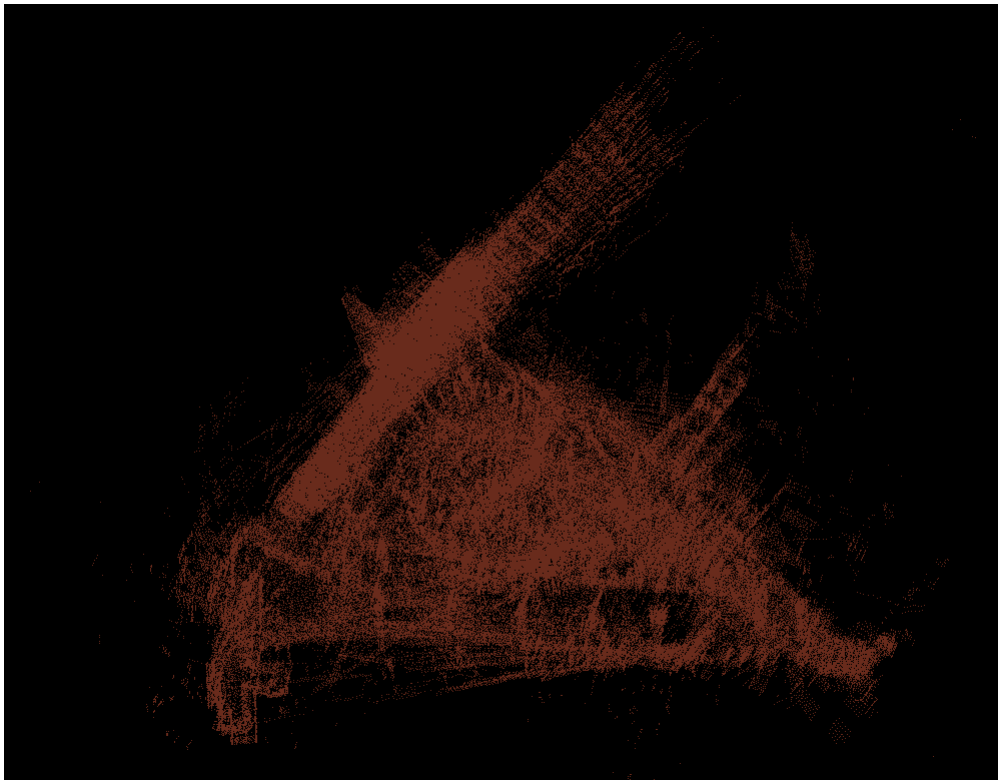


Figure 7.14: Erroneous map due to map point cloud duplication

To reduce the payload weight on the drone and streamline the point cloud data, we decided to incorporate a 2D 360-degree laser range scanner at this stage in the design. With the point cloud data generated from the LiDAR, we employed a popular SLAM algorithm, called Hector SLAM, that provides a local position estimate without GPS and IMU odometry. It does this internally by using scan matches from one scan to the currently generated map at high sampling rates. We could not use GPS because of most of the experiments were conducted indoors. Moreover, due to the strong magnetic interference in our laboratories, we could not rely on our IMU;s magnetometers, so Hector SLAM was a great alternative. The loop closure with Hector-SLAM was not explicit. However, from our experiment, we found out that most conditions, the map was usable for simple path planning, so the drone could use this instead of the Velodyne 3D LiDAR.

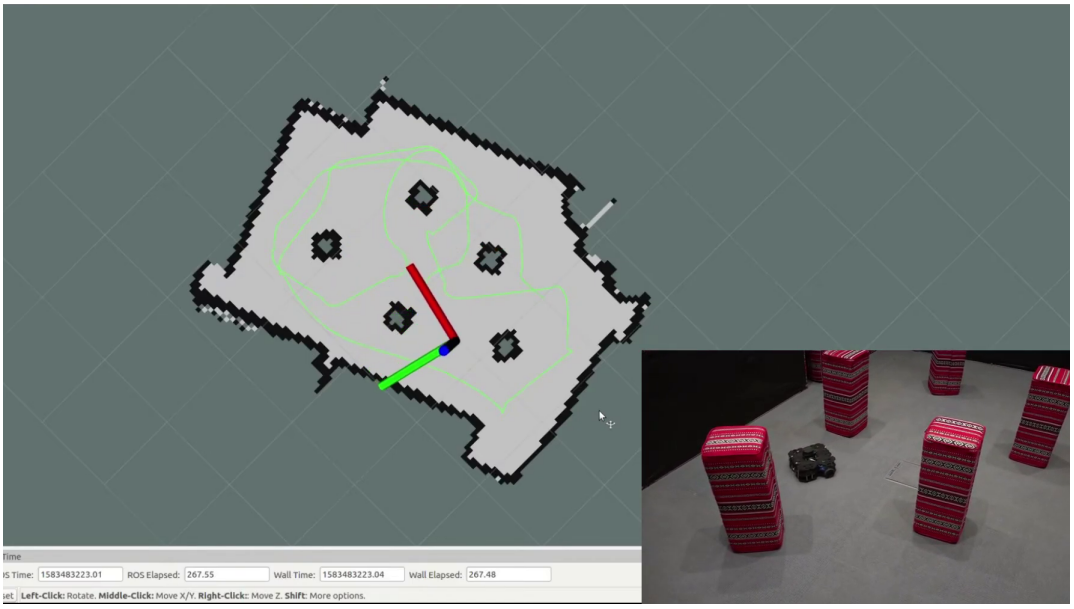


Figure 7.15: Hector SLAM experiment

7.7 Possible Future Extensions of the Project

One of the limitations of the computer vision component of this design is range. A system with a higher resolution camera with larger markers could improve the range but this would require further investigation. In addition, three dimensional drone path planning for area coverage or energy efficiency could also be investigated in future work. Area coverage experiments were limited due to the impact of Covid-19 on the project, so further testing in this area would be beneficial. Additionally, one major limitation of this system is short flight time due to power limitations, as the batteries need to be recharged often. The current system might not perform well in environments with poor visibility and in very confined environments. The experiments conducted have laid the groundwork for these future possibilities.

8 Impact of COVID-19 on the Capstone Project

Following the U.A.E Ministry of Education's and NYU Abu Dhabi's decision on March 4, 2020 to implement a distance learning system in response to the rapid spread of the novel Coronavirus (COVID-19), our capstone team obtained a special approval from the Dean of Engineering, Prof. Samer Madanat, on March 19, 2020, for uninterrupted access to our capstone space, the NYUAD's CTP-Kinesis Lab. We were instructed to maintain proper social distancing protocols and work under the direct supervision of our lab expert, Dr. Nikolaos Evangeliou. Rapidly changing regulations, however, have periodically interrupted lab access. This included the university's decision on April 14, 2020, to limit campus access to on-campus residents only, which restricted Dr. Evangeliou from supervising us at the lab. Since his supervision is required for us to use the drones, we once again updated our request to accommodate for this change. After a week, we again received approval from the department and resumed our work with increased social distancing in place.

We are grateful to the Engineering Department for granting us access to the lab space when requested and ensuring our safety through extensive social distancing and equipment sanitization protocols – however, the uncertainties brought forward by COVID-19 have at times delayed the completion of experiments due to the intermittent lab access described previously. Although our final goals for the capstone remain the same, we abbreviated the testing process using fewer experimental trials. For instance, after preparing our final experiment, it had to be conducted via a simulation due to lab access restrictions. While we were ultimately able to complete the minimum tests necessary to fulfil our project goal and implement a collaborative control design, we have chosen to forego some further research and testing that we had hoped to complete.

9 Ethics

When implementing engineering design projects such as this one, it is critical to consider the ethical implications of the technology used.

One particular ethical concern with our design is the potential violation of privacy that would come with using spherical cameras on drones in public areas. The image data could potentially be used for surveillance, which is ethically questionable without the consent of people in the image. We plan to mitigate this issue by testing the design in the private lab space, instead of venturing into public areas.

When the image data is no longer needed, it will be deleted after collection. We will not share the image data with anyone outside the research group. In the event that the drones need to be tested outdoors, we will choose an area without many people around and obtain the consent of those who pass by. We will not fly the drones near any private property and will be careful to comply with local regulations regarding drone flight.

Another ethical concern with our design is safety. While it is unlikely, injuries are possible from drone propellers colliding with people, or from components falling off of the drone and

hitting people below. We will mitigate this by securing all components, following emergency procedures in case of equipment failure, wearing personal protective equipment (goggles and close-toed shoes) when flying the drones, and making sure that everyone near the drones is aware of their presence and location at all times.

10 Design Expectations vs. Achievements

10.1 Design Testing using the Motion Capture System

To test the design and evaluate whether our design met the requirements, we employed the use of 24 Vicon Vintage cameras operating at 120Hz. These coordinated groups of cameras comprised the Vicon motion capture system that we used as ground-truth values for the precise localization values for objects on the experimental grounds. This system was able to achieve sub-millimeter accuracy using reflective markers that were placed on the object to be tracked, allowing us to evaluate the performance of our design.

To illustrate how the Vicon system works, we will include data from a preliminary test that was conducted using ground vehicles. The vehicle was programmed to drive in a circular trajectory three times, and the Vicon was able to capture how accurately the vehicle followed this trajectory, thus evaluating the system performance. A plot of the vehicle's position is displayed in the sequel:

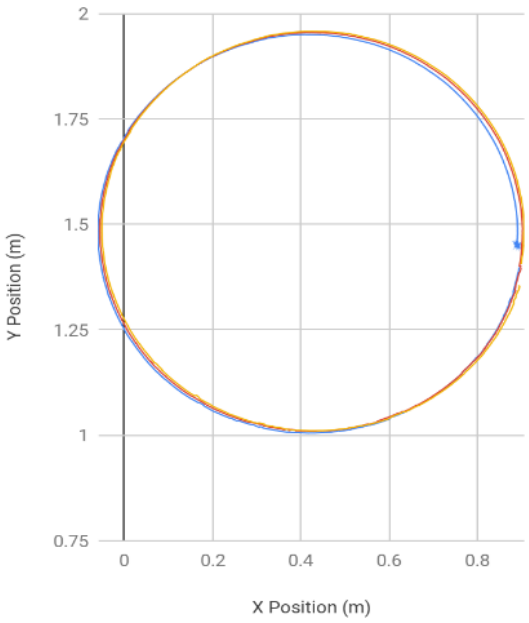


Figure 10.1: Sample Turtlebot3 Data Collection from Vicon System

The tracking was measured in terms of the RMS error between the expected and the real

trajectory; this criterion will also be used for the drone-swarm system.

10.2 Explanation of Testing and Evaluation Criteria

The first design evaluation criterion is whether the pose estimation system performs with the desired accuracy of at least 4cm. This was evaluated by comparing results from the relative localization experiments that employed the Ricoh Theta with the Vicon system measurements. The localization error in any dimension was at an average between 2.2cm and 3.8cm. This fits within our design evaluation criterion, however, there were some instances where the drone's fiducial marker was not detected even though it was within range. This was due to loss of sight of the marker when the drone flew below the camera and the marker was therefore blocked by the body of the drone.

The second design evaluation criterion involves measuring the vibration of the sensor or camera mounted on top of the drone. As the drone flies, it makes small irregular movements, and this shaking causes noise in the sensor data. If the damping system design works properly, it will reduce the shaking by at least 0.05g. This will be assessed using an accelerometer to determine how much the sensor is shaking. Additionally, if this evaluation criterion is unmet, the data will have too much noise and will not meet the previous pose estimation criterion. Although this criterion was not evaluated due to the experimental limitations imposed by the COVID-19 outbreak, the frames captured from the camera were good for the localization algorithm so the vibrations did not pose a significant roadblock in this design.

The third design evaluation criterion is whether the accuracy of the map of the target area is accurate to within 3cm. The map should include all obstacles in the environment and clearly indicate the boundaries of the environment, if it is enclosed. This criterion was evaluated using visual analysis of the map in comparison to the experimental grounds and using a surveyor's tape to estimate the relative distances from the objects from a fixed map origin. From the visual analysis, the 2D hector SLAM map passed this criterion, the RMSE was at an average of 2.8cm. The 3D SLAM using LOAM, however, could not be evaluated due to the constraints posed by the COVID-19 pandemic on the experiments.

The fourth design evaluation criterion is whether the latency meets the specified requirements. What is the pose estimate response time for the drones, and how fast would *drone 1* know the simultaneous location of *drone 2* via the ad-hoc channel? Rough calculations show us that this *pose estimate response* is roughly 303ms (*Pose Estimate Response* = 33 ms (for one frame) + 40 ms (latency from the camera to ROS) + 220 ms (post estimation latency) + 10 ms (latency from sending double numbers for positions (x,y,z) and (roll, pitch, yaw))= 303 ms), which tells us that anything slower than this might result in a collision. The latency of the ad-hoc channel for direct pose exchange was 0.01ms which also fulfilled this evaluation criterion.

The fifth design criterion is whether the design fulfills the safety concerns described in the Non-Technical Constraints section. The drones will be flown several times in various trajectories, so it is pertinent to ensure that all added components are mounted securely and do not fall off and that the drone swarm flight does not pose a safety hazard to people nearby.

The sensor fusion system was securely attached with bolts and nuts and none of the components fell out during flight time. The drones were also set to land when the battery was critical so they never fell off due to power loss. The ad-hoc system was also implemented so that in the case the sensor fusion system failed, the drones could still have direct communication to avoid collisions.

10.3 Conclusion

This project successfully fulfilled the objectives discussed in the problem definition and generally met the design criteria that we were able to test. This project deepened our understanding of the engineering design process and key developments in the field of swarm robotics. Despite the many challenges we faced during the implementation and testing phases of the project, we were able to complete the project through dedicated teamwork and problem-solving. We anticipate that the range of hardware, software, and general design skills obtained through the completion of this project will be valuable to our work on future engineering design projects.

11 Appendix

Detailed documentation and code files can be viewed at:
`https://github.com/RISC-NYUAD/capstone1920/blob/master/README.md`
(private repository)

REFERENCES

Albani, D., IJsselmuiden, J., Haken, R., and Trianni, V. (2017). Monitoring and mapping with robot swarms for agricultural applications. *14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 1.

Alexis, K., Nikolakopoulos, G., and Tzes, A. (2014). On trajectory tracking model predictive control of an unmanned quadrotor helicopter subject to aerodynamic disturbances. *Asian Journal of Control*, 16:209–224.

Aruco. Aruco: An efficient library for detection of planar markers and camera pose estimation. Available at `https://docs.google.com/document/d/1QU9KoBtjSM2kF6ITOjQ76xqL7H0TEtXriJX5kwi9Kgc/edit`.

Arvanitakis, I., Tzes, A., and Giannousakis, K. (2018). Synergistic exploration and navigation of mobile robots under pose uncertainty in unknown environments. *International Journal of Advanced Robotic Systems*, pages 1–10.

Babinec, A., Jurišica, L., Hubinský, P., and Duchoň, F. (2014). Visual localization of mobile robot using artificial markers. *Procedia Engineering*, 96:1 – 9. Modelling of Mechanical and Mechatronic Systems.

Bosse, M., Zlot, R., and Flick, P. (2012). Zebedee: Design of a spring-mounted 3-d range sensor with application to mobile mapping. *IEEE Transactions on Robotics*, 28:1104–1119.

EDU, G. Gapter drone documentation. Available at `http://edu.gaitech.hk/gapter/`.

Garrido-Jurado, S., Muñoz-Salinas, R., Madrid-Cuevas, F., and Marín-Jiménez, M. (2014). Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition*, 47:2280–2292.

Gusrialdi, A., Hirche, S., Takeshi Hatanaka, and Masayuki Fujita (2008). Voronoi based coverage control with anisotropic sensors. In *2008 American Control Conference*, pages 736–741.

NYUAD (2018). Research report. Available at `https://nyuad.nyu.edu/en/research/impact/our-research.html`.

Packer, J. and Josh, R. (2013). Romancing the drone: Military desire and anthropophobia from sage to swarm. *Canadian Journal of Communication*.

Pierzchała, M., Giguère, P., and Astrup, R. (2018). Mapping forests using an unmanned ground vehicle with 3d lidar and graph-slam. *Computers and Electronics in Agriculture*, 145:217 – 225.

Pixhawk. Pixhawk autopilot. Available at `https://pixhawk.org/`.

Romero-Ramirez, F., Munoz-Salinas, R., and Medina-Carnicer, R. (2018). Speeded up detection of squared fiducial markers. *Image and Vision Computing*, 76.

Shanghai Slamtec Co., L. Rplidar a1. Available at https://www.slamtec.com/en/Lidar/A1.

Theta, R. Ricoh Theta V User Guide. Available at https://support.theta360.com/en/manual/v/content/add_info/add_info_01.html.

TP-Link Technologies Co., L. TL-WR802N|300Mbps Wireless N Nano Router|TP-Link. Available at https://www.tp-link.com/us/home-networking/wifi-router/tl-wr802n/#overview.

Tzes, M., Papatheodorou, S., and Tzes, A. (2018). Visual area coverage by heterogeneous aerial agents under imprecise localization. *IEEE Control Systems Letters*, vol. 2, no, 4, pp. 623-628.