

**AUTONOMOUS PATH PLANNING OF UNMANNED  
AIR SYSTEMS USING COMPUTATIONAL  
METHODS AND OPTIMIZATION ALGORITHM**

**TAUSIFUL ISLAM, 170011017  
JAHIDUZZAMAN TANVIN, 170011032  
MUHAMMAD SAMIN HASAN, 170011043**

Submitted in Partial Fulfillment  
of the Requirements  
for the Degree of

Bachelor of Science in Mechanical Engineering

**DEPARTMENT OF MECHANICAL AND PRODUCTION  
ENGINEERING**

**MAY, 2022**

## **CERTIFICATE OF RESEARCH**

*This thesis titled “AUTONOMOUS PATH PLANNING OF UNMANNED AIR SYSTEMS USING COMPUTATIONAL METHODS AND OPTIMIZATION ALGORITHM” submitted by TAUSIFUL ISLAM (170011017), JAHIDUZZAMAN TANVIN (170011032) and MUHAMMAD SAMIN HASAN (170011043) has been accepted as satisfactory in partial fulfillment of the requirement for the Degree of Bachelor of Science in Mechanical Engineering.*

### ***Supervisor***

---

**Dr. Mohammad Ahsan Habib**

**Professor**

Department of Mechanical & Production Engineering (MPE)  
Islamic University of Technology (IUT), OIC  
Board Bazar, Gazipur  
Dhaka, Bangladesh.

### ***Head of the Department***

---

**Dr. Md. Anayet Ullah Patwari**

**Head of the department**

Department of Mechanical and Production Engineering (MPE)  
Islamic University of Technology (IUT)  
Board Bazar, Gazipur  
Dhaka, Bangladesh.

## **DECLARATION**

*I hereby declare that this thesis entitled “AUTONOMOUS PATH PLANNING OF UNMANNED AIR SYSTEMS USING COMPUTATIONAL METHODS AND OPTIMIZATION ALGORITHM” is an authentic report of study carried out as requirement for the award of degree B.Sc. (Mechanical Engineering) at Islamic University of Technology, Gazipur, Dhaka, under the supervision of Dr. Mohammad Ahsan Habib, Professor, MPE, IUT in the year 2022.*

*The matter embodied in this thesis has not been submitted in part or full to any other institute for award of any degree.*

---

*TAUSIFUL ISLAM*

*170011017*

---

*JAHIDUZZAMAN TANVIN*

*170011032*

---

*MUHAMMAD SAMIN HASAN*

*170011043*

## **ACKNOWLEDGEMENT**

*We would like to begin by saying Alhamdulillah and thanking Almighty Allah for making it possible for me to complete this job on time. We also deserve to thank Dr. Mohammad Ahsan Habib, Professor, Department of Mechanical and production Engineering, IUT, for his strong and patient assistance through unanticipated obstacles throughout the project and his valuable advice when confronted with difficulties. Because of his generosity, friendliness, and firm guidance, we were able to better manage our personal life and career when faced with unforeseen challenges.*

*We would want to take this opportunity to thank our parents for their continuing support and devotion to our pursuit of higher education.*

## **ABSTRACT**

In recent years, the use of Unmanned Air Systems (UAS) has become popular in many industries including agriculture, logistics, security agencies and humanitarian missions. An Unmanned Air System consists of a drone or Unmanned Aircraft (UA) connected to the Ground Control Station (GCS) via various means of communication. During the early stages of development, most of these UAs were remotely operated by a pilot. However, various modes of autonomy are added nowadays to make way for autonomous flights. This allows the systems to be implemented in long haul flights, specifically in the realm of parcel delivery and humanitarian aid missions. To perform these missions, path planning is required since there is a multitude of paths that the UA can follow in order to reach its destination. The paths may vary in terms of distance, environmental parameters including wind speed and temperature, potential danger zones and so on. In order to minimize the costs of the user and to ensure the success of the mission, the operator has to select the best possible path for the flight. Since the problem comprises a lot of variables, a multiobjective function may be devised to help with the selection process. This paper explores the use of a custom-made cost function which is used to check the usefulness of a set path. The cost function takes into account the distance, time of completion and the energy consumption of the path to come up with a score for that specific path. The start and end points of the journey are fed to the system and an optimization algorithm is used with the custom-made cost function to derive the optimum path for the UA to complete the mission. The process is run for an array of environments, each with a different start and end point, and the optimized path is fed to the UA. The UA then actually flies through this route and the results of the actual flight are compared to the results obtained from the theoretical process to ensure that there is harmony between them.

# TABLE OF CONTENTS

CERTIFICATE OF RESEARCH .....	2
DECLARATION .....	3
ACKNOWLEDGEMENT .....	4
ABSTRACT.....	5
TABLE OF CONTENTS.....	6
LIST OF FIGURES .....	10
LIST OF TABLES.....	11
NOMENCLATURE .....	12
CHAPTER 1: INTRODUCTION.....	13
1.1 Background of the study .....	13
1.2 Applications .....	13
1.2.1. Agriculture .....	13
1.2.2. Industry Inspection.....	13
1.2.3. Package Delivery .....	14
1.3 Problem Statement .....	14
1.4 Goal and Objective.....	15
1.5 Scope and Limitations .....	16
CHAPTER 2: LITERATURE REVIEW .....	16
2.1 Introduction .....	16
2.2 Multi Objective Optimization for UAV Path Planning.....	16
2.3 Energy Consumption Model .....	19
2.4 Nelder Mead Algorithm .....	21
2.5 Summary .....	22

CHAPTER 3: STUDY OF SWARM INTELLIGENT ALGORITHM.....	23
3.1 Particle Swarm Optimization (PSO):.....	23
3.1.1. Identification of PSO .....	23
3.1.2. Objectives of PSO.....	24
3.1.3. Features of PSO .....	24
3.1.4. Methodology of PSO .....	24
3.1.5. Exploitation and exploration in PSO .....	25
3.1.6. Mathematical model of PSO.....	26
3.1.7. Flowchart of PSO.....	27
3.2 Artificial Bee Colony (ABC) .....	27
3.2.1. Identification of ABC .....	28
3.2.2. Stratification of bee based on their functions: .....	28
3.2.3. Methodology of ABC .....	29
3.2.4. Exploitation and exploration in ABC .....	30
3.2.5. Mathematical Model of ABC.....	30
3.2.6. Flowchart of ABC:.....	31
3.3 Ant Colony Optimization for Continuous Domain (ACOR) .....	32
3.3.1. Identifications of ACOR:.....	32
3.3.2. Exploitation and Exploration in ACOR:.....	33
3.3.3. Mathematical Model of ACOR.....	33
3.3.4. Flowchart of ACOR.....	34
CHAPTER 4: AUTONOMOUS SYSTEM .....	35
4.1 UAS autonomy:.....	35
4.2 Quadcopter Flying Theory .....	36
4.3 UAV Manufacturing .....	38

CHAPTER 5: SOFTWARE STACK.....	41
5.1 Flight Controller Firmware .....	41
5.2 Installing and Configuring APM Firmware .....	41
5.3 DroneKit-Python .....	43
CHAPTER 6: PROBLEM FORMULATION .....	44
6.1 Environment and path representation:.....	44
6.2 Cost Function: .....	44
6.2.1. Objective functions: .....	45
6.2.2. Path length: .....	45
6.2.3. Traverse Time: .....	45
6.2.4. Path Smoothness: .....	46
6.2.5. Collision Cost: .....	47
6.2.6. Energy Consumption: .....	48
6.2.7. Equations of Motion: .....	49
6.3 Proposed Path Planning Algorithm:.....	50
CHAPTER 7: SYSTEM DESIGN.....	51
7.1 Airframe .....	51
7.1.1. Functional Description:.....	51
7.1.2. Rationale for Selection:.....	51
7.2 Navigation & Mission Control.....	51
7.2.1. Functional Description:.....	51
7.2.2. Rationale for Selection:.....	52
7.3 Sensors .....	52
7.3.1. Functional Descriptions: .....	52
7.3.2. Rationale for Selection:.....	53



7.4	Autonomy.....	53
7.4.1.	Functional Description:.....	53
7.4.2.	Rationale for Selection:.....	54
CHAPTER 8:	EXPERIMENT DESIGN .....	56
8.1	Introduction .....	56
8.2	Methodology .....	56
8.2.1.	Environment Creation:.....	56
8.2.2.	Path Generation:.....	56
8.2.3.	Reference Frame Transformation: .....	57
8.2.4.	Mission Flight: .....	57
8.3	Results Evaluation.....	58
CHAPTER 9:	RESULTS AND FINDINGS.....	60
9.1	Result Discussion: .....	60
9.2	Conclusion and Further Study.....	66
References	.....	67
APPENDIX-A:	Pseudo Code .....	72

## LIST OF FIGURES

Figure 3-1: Flow Chart of PSO Algorithm .....	27
Figure 3-2: Flowchart of Artificial Bee Colony. ....	31
Figure 3-3: Flowchart of ACOR.....	34
Figure 4-1: Autonomy from start to finish[31].....	35
Figure 4-2: quadcopter motor spin direction[33].....	36
Figure 4-3: Movements on Quadcopter's Axes[33].....	38
Figure 4-4: Typical Quadcopter Wiring Layout[35] .....	39
Figure 6-1: Collision cost.....	47
Figure 7-1: PIXHAWK [39] .....	51
Figure 7-2: HERE 2 GPS[39] .....	52
Figure 7-3: LiDAR Lite V3[39].....	53
Figure 8-1: 3D environment.....	56
Figure 8-2: Optimized path considering length, time and energy consumption.....	57
Figure 8-3: Generated path in mission planner.....	57
Figure 8-4: Test flight .....	58
Figure 8-5: Power consumption graph from data log .....	59
Figure 8-6: Power consumption and flight path .....	59
Figure 9-1: Flight path for test no. 01 .....	61
Figure 9-2: Flight Path for test no. 02.....	62
Figure 9-3: Flight Path for test no. 03.....	63
Figure 9-4: Flight Path for test no. 04.....	64
Figure 9-5: Flight Path for test no 05.....	65

## LIST OF TABLES

Table 9.1: Theoretical and experimental data for test no. 01.....	61
Table 9.2: Theoretical and experimental data for test no. 02.....	62
Table 9.3: Theoretical and experimental data for test no. 03.....	63
Table 9.4: Theoretical and experimental data for test no. 04.....	64
Table 9.5: Theoretical and experimental data for test no. 05.....	65

## **NOMENCLATURE**

RPAS	Remotely Piloted Aircraft System
LoS	line-of-sight
UAV	Unmanned aerial vehicles
UA	Unmanned Aircraft
GCS	Ground control station
GPS	Global Positioning System
MOO	Multi-objective optimization
MOMVO	Multi-Objective Multi-Verse Optimization
MOGWO	Multi-Objective Grey Wolf Optimizer
MSSA	Multi-Objective Salp Swarm Algorithm
MOPSO	Multi-Objective Particle Swarm Optimization
CSP	constraint satisfaction problem
MOEA	multi-objective evolutionary algorithm
MPP	mission planning problem
GBNM	Globalized Bounded Nelder Mead
GA	genetic algorithms
ESC	Electronic Speed Controllers
EKF	Extended Kalman Filter

# CHAPTER 1: INTRODUCTION

## 1.1 Background of the study

Unmanned Air System or UAS in short is an assembly of a drone and the Ground Control Station or GCS in short with various means of communication between them[1][2]. The system can be reportedly piloted by a trained pilot in which case the aircraft is known as a Remotely Piloted Aircraft System (RPAS)[3]. However, various modes of autonomy can be added to the system to facilitate autonomous missions. Semi-autonomous features like auto pilot, steering assist, auto land is available, whereas more complex missions like autonomous touch and go, mass land survey, surveillance missions and so on.

## 1.2 Applications

### 1.2.1. Agriculture

- a) *Agricultural Mapping:* For agricultural mapping, survey drones are used which incorporate a specialist camera which includes multispectral/LiDAR/RGB sensors.
- b) *Crop Health Assessment:* From the images captured by the drones, Produce NDVI Image of the corn fields in the site, generate NDVI for the fields using multispectral data, Define regions for healthy crops, stressed crops & unhealthy crops[4].
- c) *Spraying:* A tank, pump, and nozzles are all on board the spraying drone. Batteries may be used to power the drone. By using an automated device called the "Ground Station" to fly above their field, an operator may send a downward force of propellers to push a spray onto their crop. The turbulence from the drone's propellers helps to guarantee that the spray coverage is constant and comprehensive since the drone is usually just 9-12 feet (3-4 meters) above the crop[5].

### 1.2.2. Industry Inspection

*Line of Sight Survey (LOS):* It is the level of obstruction present between two places. This level of obstruction governs the visibility and quality of signal between the two points. Unmanned aerial vehicles (UAVs) are a promising component of future wireless communication networks because of their potential to link devices without access to existing infrastructure at a reasonable cost. On-demand wireless systems with low-altitude UAVs are in general quicker to deploy, more flexible to reconfigure, and likely to have

stronger communication channels owing to the availability of short-range line-of-sight (LoS) connections[6].

### ***1.2.3. Package Delivery***

Drones can be used to supply and transport packages containing food, medical equipment, ration supplies etc over large distances in an exceeding short amount of time. The reduction in time is crucial while transporting goods and supplies in time-sensitive scenarios. This has been implemented in many countries all over the world. Most notably, Zipline, a company based in the USA, is pioneering drone delivery all over the world. Besides this, there have been constant efforts from Amazon, Airware etc to repeat the same[7].

## **1.3 Problem Statement**

While commercial airlines rely on radar technology coupled with an array of Air Traffic Control or ATC for coordination and movement in an airspace, drone technology is still not supervised in real time by a systemic set of coordinators. This is because most drone operations are carried out within a very small airspace which is in the range of 1-2 km from the take-off and landing position. Furthermore, a Remote Pilot is usually present during the operation of the drone within the 2 km operational circle or geofence. However, for long haul missions like, surveillance across the coast of Bangladesh or entire district survey of agricultural land, the path or route taken by the Unmanned Aircraft (UA) has to be clearly defined. This is to ensure that there is compliance between the autonomous mission and the laws of commercial aviation. Therefore, the GCS has to identify and plan the path to be taken by the UA by maintaining complete liquidity of the local airspace laws.

When planning for the path taken by the UAV, the GCS Operators have to consider an abundance of factors before feeding the path to the autonomous system. Some of the factors are outlined below:

- I. *Permitted Routes by Local Aviation Authority:* Aerial robotics is subject to a lot of regulatory constraints from local government agencies. This is to ensure the safety of the people and the cause of the national safety regulations. As a result, drone operators have the liberty of flying in only certain green zones which are away from areas of commercial aviation to ensure the set standard of safety.

- II. *Energy Consumption of UA:* The energy usage of the UA is a driving factor behind the route to be selected for a particular mission. If the energy consumption for a route is too high, it will lead to substantial losses for the commercial user. This can lead to a variety of other multivariable economic problems. Therefore, in order to make a profit and to cut down on costs, the drone operator at the GCS has to select the route that has the lowest energy consumption for the UA.
- III. *Velocity of UA:* The speed at which the UA flies is not usually constant. During turns, gaining height, the velocity is constantly changing. However, if 2 journeys are considered with 2 different average speeds, the total time to complete the journey will vary. This leads to the paradigm that the average speed of the UA affects the outcome of the mission in the sense that it controls how fast or how slow the mission is being carried out.
- IV. *Total Distance Covered:* The length of the path taken by the UA is also a factor to be considered when selecting the best possible path for flight during a mission. This is because the electro mechanical components of the UA operate within a certain threshold of operational limit. Crossing the limit would alter the overall safety of the mission. Hence, the total distance to be covered has to be controlled and the lowest distance to be covered should be selected.
- V. *Miscellaneous Variables:* Other variables including the sound or noise from drone operation in a neighborhood has to be controlled. Environmental factors such as wind, temperature etc. are also deciding factors while selecting the perfect path for the operation.

## **1.4 Goal and Objective**

The goal of this thesis project is to find and derive the computational methods necessary to generate paths to be taken by a UA in an autonomous mission. The methods will then be used to run a custom-made cost function which will be used to evaluate the usefulness of the individual paths by considering a range of different factors. The best path will be selected by running an optimization algorithm which will find the best possible path for a given set of parameters. The entire process will be validated in real time using a custom-made drone and using the drone to perform the real time mission using the results obtained

from the optimization algorithm[8]. The results of the experiment will then be evaluated against the theoretical data to prove the validity of the experiment.

## **1.5 Scope and Limitations**

It should be noted that the path planning paradigm can be done in two ways. The scope of this paper will explore the path generation, optimization and selection of the best path from an array of different paths using various computational methods and optimization algorithm. However, this still leaves the drone susceptible to real time dangers and obstacles. In fact, real time obstacle avoidance using various image processing techniques may be used to avoid any obstacles that the drone encounters in real time. Furthermore, onboard computers coupled with various probes can allow the drone to consider various parameters including wind speed, outdoor temperature etc. to accurately carryout the path planning on air.

The Global Positioning System (GPS) used in this project has an operational resolution of 5m. This means that the positional data obtained may all vary by a certain degree. Using a Real Time Kinematics GPS would mitigate this problem but that is beyond the scope of this paper.

## **CHAPTER 2: LITERATURE REVIEW**

### **2.1 Introduction**

Conventional Unmanned Aerial Vehicle (UAV) mission and route optimization has been developed with a single aim or set of control parameters in mind, as has been the case for many years. With each new application, whether military or civil, the technology and its use cases get more and more sophisticated. As a result, the goals are no longer linear or solitary in character, but instead reliant on a variety of factors. Several scholars have attempted to address the topic from multiple perspectives.

### **2.2 Multi Objective Optimization for UAV Path Planning**

The application of the Multi-Objective Multi-Verse Optimization (MOMVO) method to handle the route planning issue of quadcopters with moving obstacles has been attempted



by [9]. This method is concerned with the quadcopter taking the shortest route possible while avoiding collisions with moving objects.

The deployment of a set of homologous metaheuristics, such as the Multi-Objective Grey Wolf Optimizer (MOGWO), the Multi-Objective Salp Swarm Algorithm (MSSA), the Non-Dominated Genetic Algorithm II and the Multi-Objective Particle Swarm Optimization (MOPSO), has been compared to the discovery of different possible alternative paths (NSGAI).

[10] addresses the issues that arise during the design of automated missions in an unmanned aerial vehicle. These missions comprise a variety of duties, unmanned aerial vehicles, and ground control stations (GCS). The following goals are taken into account for optimization: make span, fuel consumption and cost, and reliability. An evolutionary algorithm with several objectives, paired with a constraint fulfillment issue model, is used by the researchers to resolve these challenges.

It is treated as a constraint satisfaction problem (CSP) and addressed using a multi-objective evolutionary algorithm (MOEA) to solve the mission planning problem (MPP). This is the method that optimizes amongst a number of different variables in the situation at hand.

[11] is concerned with route planning for search and rescue operations. The main goal is to shorten the time required to complete the mission while locating a target and sending the coordinates of that target back to the ground control station.

Using a genetic algorithm technique, the researchers intend to solve this multi-objective issue by assessing tactics via the use of a data mule, a relay chain, and a unique hybrid way to connect with the ground crew members.

Depending on the mission requirements, the algorithm may be tweaked to favor coverage or connection above others. As a result, connection is seen as a mission objective rather than a limitation in these cases.

The goal of [12]'s study is to develop a solution to the problem of route planning in three-dimensional space.

Meta-pathways are used to construct a large number of different solution paths. When generating several solution pathways, the path planner uses Particle Swarm Optimization (PSO) to ensure that they all meet the established requirements. It is these criteria that are regarded as the goals for which the solutions are optimized.

It is possible to keep a safe distance from any barrier by including topographical information in the UAV's navigation system.

[13] describes a three-dimensional offline route planner for unmanned aerial vehicles (UAVs) that is based on a multi-objective optimization approach.

The optimization of two competing goals: one is the minimization of the total route length; and two, the maximization of the margin of safety from ground impediments, is accomplished at the same time. In this way, it is ensured that both goals are given top priority, and no extra issue information is required for transforming this two-objective problem into a single-objective problem. According to their findings, a variety of optimal pathways with variable trade-offs between the goal functions are produced by the evolutionary optimization approach used by the researchers.

In the past, evolutionary algorithms (EA) have been employed effectively to calculate near-optimal pathways in blocked and constantly changing environments. When the uncertainty of the barriers is explicitly taken into consideration, it is possible for "optimal" pathways to survive that are different from those that would be preferred in a fully deterministic environment.

[14] have studied and investigated the application of evolution-based route planning to the travel of an unmanned aerial vehicle (UAV) over a field of obstacles at a variety of unpredictable locations in an uncertain environment. Beginning with the static version of the EA method, which is used to generate a route at a single moment in time, it can be seen that it is quite efficient. We here discussed the algorithm and demonstrated its behavior, especially how it reacts in various ways depending on the known accuracy of the predictions of the surrounding environment. After that, they also demonstrate how the static structure may be expanded to take into account uncertainties that vary over time.

Application to route planning across a field of moving obstacles whose future motion is unpredictable allows us to show our theory.

[15] study examines numerous unmanned aerial vehicle (UAV) route planning algorithms that have been developed over a long period of time. In route planning approaches, the goal is not just to discover the shortest and most efficient path, but also to offer a collision-free environment for unmanned aerial vehicles (UAVs). It is critical to have route planning strategies in place in order to calculate a safe path to the end destination in the shortest amount of time feasible. Different route planning strategies for unmanned aerial vehicles (UAVs) are discussed in this work and grouped into three major categories: representational techniques, non-cooperative techniques, and cooperative techniques. The connection and coverage of the UAVs' network communication are explored and studied in the context of these strategies. The present ideas for UAV path planning have also been subjected to a critical study based on each category of UAV path planning. Different comparison tables based on factors such as route optimality, length, cost-efficiency, completeness, energy-efficiency, time-efficiency, collision, and robustness avoidance are also given in the text to aid in comprehending the subject matter better. Other unsolved problems in UAV route planning and network communication are also addressed so that the paper's audience may have a more comprehensive understanding of the subject matter at hand.

### **2.3 Energy Consumption Model**

Drone operations have a significant energy limitation if they are to realize their full promise in terms of delivering speedy delivery, lowering costs, and decreasing emissions. A large number of optimization models used to build drone or drone delivery systems only partially account for energy consumption as a fixed limitation on drone endurance (flight time limit) or range (discounted in certain circumstances) (flight distance limit). The fundamental physical forces that are involved during flights, or data taken from the ground, are used in other drone delivery studies to directly incorporate energy. [16] makes a crucial addition to the modeling of quad model energy consumption by converting the all the fundamental flying principles of manned aircraft into a considerably smaller sized model for the of UAVs, which is a first in the field. Lift-to-drag ratio, a critical factor in drone design, is

examined in this paper utilizing an integrated approach that takes into account both aerodynamic and drone design considerations. In addition, the energy model contains a fixed component for the power consumption of avionic systems. In addition, the RAND Corporation is using this integrated model in a series of studies looking at the amount of energy consumed by city-scale drone delivery systems.

When estimating drone energy consumption, an alternative approach might be taken from the perspective of helicopter operations, assuming how much energy has been put to use takeoff, during level flight, and landing is about equal to the energy spent when hovering. According to [17], a multirotor helicopter's hover power consumption is a function of payload mass and battery capacity. They also describe field trials and create regression parameters for modest payloads, which they include in this paper as well. This hovering model is modified for the MikroKopter MK8-3500 drone, based on payload mass, a regression model is described in detail in [18]. They write, "The suggested energy consumption model gives realistic results that are equivalent to the experiment results," although they don't specify which parameters should be used to create the model.

According to several previously mentioned studies, one method of predicting drone energy usage is to use regression based on field tests. [19]. A nine-term nonlinear regression models for drone energy usage that consider payload mass, as well as horizontal and vertical speed and acceleration and wind velocity are provided by the authors.

[20] gives a uniform framework to aid with the understanding of diverse drone energy consumption models in addition to the interrelationships between important elements and performance measurements, in order to aid in the decision-making process for delivery drone. Drone energy consumption models are reviewed, evaluated, and classified. There follows a detailed discussion of how extremely large discrepancies exist in the estimated energy consumption rates due to varying factors such as (a) the specifics of their expected activities and uses; (b) the precise designs of the drones; and (c) the breadth and characteristics of the models. This study demonstrates that extreme caution must be used when selecting a specific drone energy consumption model and that more research,

particularly empirical research, is required to guarantee that the chosen model appropriately represents delivery drone designs and usage.

Another prominent method of modeling is the white-box approach, which is based on unique vehicle dynamics to understand the energy consumption of any electric vehicle [21]. The black-box approach is yet another technique for simulating electricity use. In this technique, the power consumption of automobiles is modeled using a broad statistical approach based on a regression model rather than a vehicle dynamics model [22]. This technique is more straightforward and offers adequate information for power consumption estimations for unmanned aerial vehicles (UAVs) mission planning. So, in the power consumption model, they use a "black box" approach that takes into account different flying situations but not the way the vehicle moves.

## **2.4 Nelder Mead Algorithm**

One of the most challenging aspects of engineering system design is dealing with the large number of possible local solutions. This has prompted significant efforts to be made in the development of global search algorithms. Globality, on the other hand, has a prohibitively large numerical cost when applied to real-world issues. It is planned to establish a fixed-cost local search that will eventually become worldwide. Globalization is accomplished via the use of probabilistic restart. The local optimizer is created using an upgraded Nelder–Mead algorithm. It takes into consideration varying boundaries. By reinitializing degraded simplexes, they are also made more resilient in general.

The Nelder-Mead method, also known as the simplex search algorithm, was first described in 1965 (Nelder and Mead, 1965)[23]. It is one of the most well-known algorithms for multivariate unstructured optimization involving no derivatives and has been around since then. Due to the fact that it does not need any derivative information, it is ideal for situations involving functions that are not smooth. A common use of this technique is to address parameter estimation and comparable statistical issues when the values of the function are unclear or sensitive to noise. Also, it can be used to solve problems in statistics and experimental mathematics that involve discontinuous functions.

A Globalized Bounded Nelder–Mead (GBNM) algorithm was developed by [24]. As a consequence of this research to deal with multimodal, discontinuous optimization situations in which it is unknown whether or not a global optimization can be performed. Various ways of re-starting the local search are explored in this article. It is possible to do numerical experiments on analytical test functions and composite laminate design challenges. The GBNM method is better than an evolutionary algorithm in terms of numerical cost and accuracy, and it is also faster.

[25] presented an algorithm that is comprised of two processes, each of which is dedicated to a single job. Global metaheuristics, such as simulated annealing, tabu search, and genetic algorithms (GAs), are effective in identifying the "optimal" locations to focus attention. Local search techniques, on the other hand, are well-established and include, for example, hill climbing (e.g., the quasi-Newton method) and the Nelder–Mead simplex search (SS). Consequently, we developed a hybrid approach, known as the continuous hybrid algorithm (CHA), that does the exploration with a GA and the extraction with a Nelder–Mead SS while maintaining the consistency of the exploration and extraction. The effectiveness of CHA was measured by putting in place a set of "benchmark" functions and comparing the results to those of other competing methods.

Approximation functions and traditional approaches for recovering external orientation are examined by [26]. That technique doesn't take into consideration the unique features of photographs captured by unmanned aerial vehicles, as shown in this study by Nelder-Simplex. Mead's approach is examined as an alternative. The independent variables' external orientation angles were measured in order to derive the components of external orientation. Angles and reference points were used to compute the spatial location.

## **2.5 Summary**

In order to formulate the objective function for this study, path length, travel time, energy consumption, path smoothness and collision with obstacles were considered.

Nelder-Mead algorithm was chosen as the optimization algorithm. It works with multidimensional search space and suitable for solving nonlinear objective function with unknown derivatives.

## **CHAPTER 3: STUDY OF SWARM INTELLIGENT ALGORITHM**

swarm intelligence is a computational intelligence field devoted to studying collective behavior in self-organizing communities of agents. swarm intelligence Animals and other organisms in their local habitat use these methods to accomplish tasks such as obtaining food and avoiding danger, as well as mating, in order to be more efficient in doing these tasks. Intuitive computing techniques based on data-driven problem optimization are used in these methodologies. SI and similar ecosystems do not have a single set of rules that control the movement of individual agents; rather, their aggregate behavior is developed by the interactions of several individuals. Small time interval adjustments are used to define each agent's mobility according to the ambient parameters that it is aware of. The SI algorithm is used to improve algorithms that search for the optimum answer to difficult circumstances in terms of flexibility, robustness, and resilience.

### **3.1 Particle Swarm Optimization (PSO):**

In 1995, Eberhart and Kennedy came up with the first concept for PSO. First thoughts on particle swarms were mostly centered on the production of intelligent calculation via the use of important social interaction models, rather than just individual computing. This algorithm is a SI algorithm because it mimics the way animals interact socially in nature. A multi-agent population, known as swarm intelligence, is more precisely impacted by PSO than a single-agent population[27].

#### ***3.1.1. Identification of PSO***

In order to address a variety of optimization problems, PSO employs stochastic, hybrid, and artificial intelligence techniques. It is feasible to create random locations of candidate solutions in PSO using a stochastic probability distribution, although this is not guaranteed. Metaheuristic search techniques may be identified even when the data collection is insufficient or the processing capability is limited since PSO is metaheuristic. Particle swarm optimization, according to the researchers, is based on a simple notion that can be implemented in a minimal amount of code. It just necessitates basic mathematical operators and makes effective use of available storage and processing power.

### ***3.1.2. Objectives of PSO***

As a non-expert user, you may take use of the PSO method's multiple aims to achieve your objective. The following are the goals:

- I. As a long-term strategy, it is necessary to collect a comprehensive evaluation of the most often occurring PSO variables.
- II. An overview of the theoretical elements of the algorithm and their relationship to PSO parameters.
- III. In order to improve the algorithm's efficiency.
- IV. Analysis of a range of algorithm applications and results will help determine the algorithm's present functioning and regulations.

### ***3.1.3. Features of PSO***

Particle Swarm Optimization (PSO) is a popular approach for solving non-convex, integer variable type, discrete, nonlinear problems. In this system, competition and cooperation among colonies or swarms influence swarm intelligence. Swarm animals have various unique qualities that have been included into the PSO algorithm to obtain best outcomes when they work together to accomplish shared community goals. When it comes to optimizing a process, there are a lot of useful traits that include communication, information sharing, and collaborative decision-making. Our work attempts to minimize an objective function as a means of evaluating the optimization problem. The method's solutions are assessed in terms of the objective functions in order to establish their acceptability and efficacy.

### ***3.1.4. Methodology of PSO***

A number of particles are used to begin the search process, and each of these particles has been programmed to look for appropriate results in the search region. If there is a community of particles, the computation time is reduced since everyone is searching for exactly the same item at once. Particles, on the other hand, are able to explore every nook and cranny of the search area in order to gather relevant data.

As the cost function in the algorithm developed to minimize costs, our thesis replicates the requirement, which is also known as an optimization issue. After establishing the goal



function, the particles are now ready to search for the best possible answer. Optimal positions for each particle indicate the best solution.

Information is transmitted among the population when each particle's motion is finished so that a global ideal position may be agreed upon. They are repeated and their values adjusted in every loop. Particle velocity, the particle's personal best location, and the population's best solution are all important components in the algorithm that governs and decides the movement of individual particles. The statistical model uses the three vectors indicated above to determine the particle's displacement and velocity in the next iteration of the model. Because of this, the calculations included both individual and collective perspectives.

Each additional iteration of the PSO algorithm improves the method's performance. Using extra parameters in the equations for calculating particle movement improves the algorithm's efficiency. Random functions are employed in the equations to prevent the parameters from being stuck in the same place. Acceleration coefficients are included in the equation, enabling the agents to move faster toward their own and the world's best values. There must be some kind of balance when deciding on the constant values, since a greater number forces the particles to move swiftly toward or beyond the intended areas, while a lower number allows particles to traverse through the search region without being prompted to return. The inertia weight parameter is added to the particle velocity, which ranges from 0.9 to 0.4, to ensure that both individual and global optimal outcomes are satisfied. For a global poll, the greater the inertial weight number, the more likely the respondent is to like it. Finally, after the necessary number of iterations, the method is complete.

### ***3.1.5. Exploitation and exploration in PSO***

Exploitation has to do with the algorithm's ferocity, while exploration has to do with its breadth. This algorithm has both of these properties. When solving an optimization issue, pertinent knowledge is often exploited. To find the greatest deals, we apply a local search strategy. When it comes to exploring the search area, random functions may help.

The PSO algorithm uses the global best solution for decision, however the personal best solution's consequences are not evident in the PSO method. The PSO algorithm is more

mobile and energetic in its search because of the absence of crossover. Exploration and exploitation must be balanced, however, in order to maximize the pace of convergence while avoiding an early convergent with non-profitable options.

### **3.1.6. Mathematical model of PSO**

Using PSO's mathematical model, one may track the movement of particles.' A particle's mobility may be assessed by its location and velocity in the search area.

The particle velocity equation for the traditional Particle Swarm Optimization technique is provided below.

$$v_n^{t+1} = w * v_n^t + c_1 * F_1 * (p_{nb} - d_n^t) + c_2 * F_2 (g_b - d_n^t) \quad 3.1$$

The displacement of the particles is computed using the following equation

$$d_n^{t+1} = d_n^t + v_n^{t+1}$$

The terms carry out the following representation:

w= Inertia weight

c<sub>1</sub>,c<sub>2</sub>= Acceleration Constants

F<sub>1</sub>,F<sub>2</sub>= Random Function

d<sub>n</sub><sup>t</sup>= Movement of the n<sup>th</sup> particle in the t<sup>th</sup> iteration

d<sub>n</sub><sup>t+1</sup>= Movement of the n<sup>th</sup> particle in the (t+1)<sup>th</sup> iteration

v<sub>n</sub><sup>t</sup>= Speed of the n<sup>th</sup> particle in the t<sup>th</sup> iteration

v<sub>n</sub><sup>t+1</sup>= Speed of the n<sup>th</sup> particle in the (t+1)<sup>th</sup> iteration.

p<sub>nb</sub>= Personal best value of the n<sup>th</sup> member

g<sub>b</sub>= Global best value among the entire community.

### 3.1.7. Flowchart of PSO

The functions and the algorithm's computation are outlined in the following steps:

Step 1: Begin the procedure by creating a community of particles.

Step 2: The optimization problem may be solved by defining the fitness function.

Step 3: Calculate each particle's new velocity using the previous location data.

Step 4: find the new location.

Step 5: Analyze the optimum reaction of each individual particle.

Step 6: Compare the best individual solutions of all particles to arrive at the global best value.

Step 7: Count how many times you have to repeat a process to reach the threshold.

Step 8: Go back to step 3 and do it all over again if you haven't completed all of the repetitions.

Step 9: When the best option is discovered, stop the procedure.

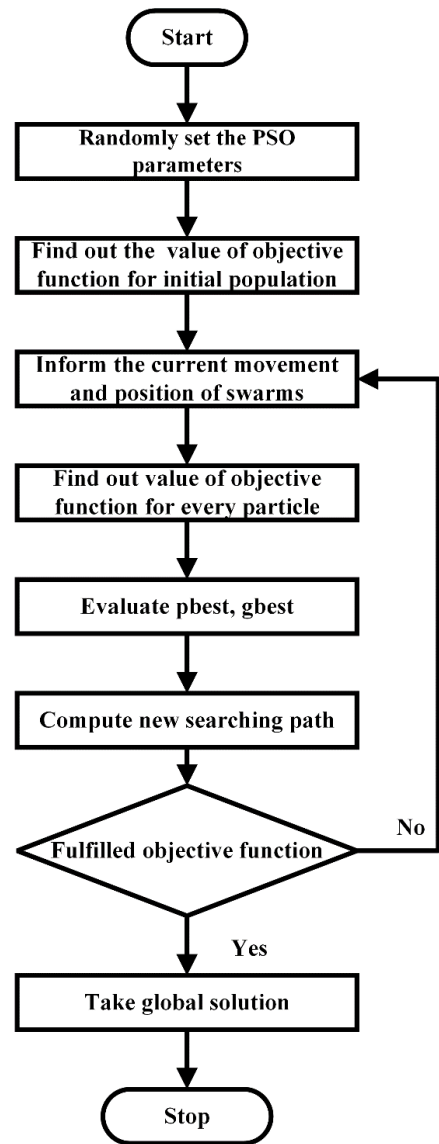


Figure 3-1: Flow Chart of PSO Algorithm

The *Figure 3-1* shows the flowchart of the PSO.

## 3.2 Artificial Bee Colony (ABC)

In 2005, Karaboga originally proposed the concept of a computer program based on a swarm intelligence principle, based on a model of a synthetic bee colony. This algorithm is based on honey bees' social and cooperative behavior in their daily foraging operations. Honey bees are very intelligent and resourceful creatures that work diligently to ensure that the hive has enough food to last them through the winter. According to the algorithm,

artificial bees must choose food sources depending on the quality of the food. In order to achieve optimum efficiency, the bees' purposes and activities are categorized, and each category fulfills its role while simultaneously providing information to the other categories. The ABC algorithm employs a similar approach to find the best solution to a comparable problem. By fine-tuning the algorithm's control parameters, the output may be easily governed and controlled[28].

### ***3.2.1. Identification of ABC***

Accordingly, Artificial Bee Colony is an algorithm that uses a combination of metaheuristics, stochasticity and artificial intelligence. Iterative discovery of effective solutions to problems with minimal prior information is possible because of this method. Food sources with a high nectar content are recognized by the bee colony as the algorithm's optimal options. One of the algorithm's advantages is that it evaluates the probability of the new food sites in terms of fitness, resulting in faster convergence of the algorithm. While prior and current solutions are evaluated, the greedy selection strategy is utilized to preserve the best of both. Because of the algorithm's extensive testing, it is more likely that optimal results will be obtained. Bee colony actions such as searching for solutions, comparing the findings, selecting the best-quality options in terms of fitness, removing low-quality food sources and finally producing superior solutions are all included in the ABC algorithm. When a bee performs a ceremonial dance while searching for food, the data it transmits becomes accessible to other bees. The algorithm's applicability is improved by adjusting several parameters.

### ***3.2.2. Stratification of bee based on their functions:***

The honey bees are divided into 3 groups according to their jobs.

I. *Employee:* A team of bees is on the prowl, continuously searching for new food sources. As soon as a food source has been found, each recruited worker bee goes to work gathering information about it. Following that, the adequacy of the food sources is reviewed in order to give a reasonable chance of a solution while also engaging with the community at large.

II. *Onlooker:* Searching for the best answer, the bees look at all of the options that have been generated and choose the one that is the most fit. Updates to food sources will

take use of these potential solutions next. A scout bee is classed as a worker bee, and food sources that do not provide feasible solutions are eliminated.

III. *Scout*: It is for this reason that scout bees have been developed to make up for the shortcomings of the hired bees of the past. Searching for food sources that the hired bees have missed; the scout bees survey the region. For this reason, Scout beehives are unlikely to come upon highly optimized food supplies. Food options for scout bees are being reviewed to see whether they can be used commercially.

The ABC algorithm uses employed and spectator bees for local investigations and scout bees for global ones to do research both locally and globally. That way, the search space may be thoroughly examined while also having a high rate of convergence.

### ***3.2.3. Methodology of ABC***

Honey bees of three distinct species work in three stages to complete the ABC algorithm. In the optimization problem, the nectar content of the food sources is proportional to an objective function, suggesting that the solutions are better matched. The target function of our job is designed to be as minimal as feasible.

To begin, the algorithm creates a colony of honey bees with each bee providing a possible food source. Track the position, direction, and quality of food establishments using their GPS systems. When nectar in food sources has been examined, a solution's fitness and probability may be computed. Comparing previously found answers to newly discovered ones using the 'greedy selection strategy' helps to preserve the better solutions. Afterward, the bees in the search arena exchange probability data and solutions in a dance.

The number of observers equals the number of hired bees. The observation bees then choose the food sources they believe are most likely to help them achieve their fitness goals. The new food sources are updated using the best solutions that have been found.

This means that certain food sources are excluded from the search since they don't provide feasible possibilities. The hired bees transform into scout bees and scour the search area to make up for the lack of answers. They're on the hunt for any kind of high-quality food they can get their hands on. Consequently, they provide low-cost search services and meals of bad quality. Scouts are only sometimes able to locate important food sources, though. The

transition from employed bees to scout bees is controlled by a limiting parameter. if the answers don't become better after a particular number of tries, they're thrown out of the equation A predetermined number of iterations, often known as the maximum number of cycles, is used to execute the algorithm (MCN). The algorithm's methodology offers a search engine that is dependable, easy, and speedy.

### 3.2.4. *Exploitation and exploration in ABC*

Using exploration and exploitation simultaneously allows the algorithm to strike a compromise between quicker convergence and finding the best possible answers in the search space.

It's a double-edged sword for both the bees who are employed and those that are spectators. Unlike honey bees, scout bees conduct their research by scanning a large area in search of new possibilities. Scout bee exploration is often effective, although convergence may be slow since crossover is not present in this phase, and hence exploitation capability is limited. The parameters for the bees employed and scouts may be adjusted to achieve a more even distribution.

### 3.2.5. *Mathematical Model of ABC*

The equation to calculate the optimal solution

$$X_{nm} = x_{nm} + \varphi_{nm}(x_{nm} - x_{km}) \quad 3.2$$

Where  $m \in 1,2,3,4,\dots\dots\dots N$

And,  $k \in 1,2,3 \dots \dots \dots D$

The probability of individual obtained solution with respect to fitness function using the following equation:

$$Probability = \frac{fitness_n}{\sum_{n=1}^N fitness_n}$$

The updated solution by the scout bees are found by the equation below

$$x_n^{m(new)} = x_{min}^m + rand()(x_{max}^m - x_{min}^m) \quad 3.3$$

Here,  $m \in 1,2,3, \dots \dots \dots D$

The description of notations used here are given below.

$N$  = Population of Employed bees = Population of onlooker bees.

$X_{nm}$  = New optimal solution

$x_{nm}$  = Initial optimum solution.

$\varphi_{nm}$  = Random value ranging from -1 to 1.

Probability<sub>n</sub> = Probability of individual solution according to fitness function.

fitness<sub>n</sub> = Fitness of each solution n.

$x_n^{m(new)}$  = Updated solution by scout bees.

$x_{max}^m$  = Maximum limit of parameter  $j \in 1,2,3 \dots \dots D$

$x_{min}^m$  = Lower limit of parameter  $j \in 1,2,3, \dots \dots D$

rand() = Array of random numbers which are uniformly distributed between 0 to 1

$D$  = Optimum parameter number of each solution.

### 3.2.6. Flowchart of ABC:

The following stages outline the ABC algorithm's strategy.

Step 1: Start with a honey bee colony.

Step 2: Declare that the goal function has to do with the food source's nectar content.

Step 3: Find out where the bees get their food.

Step 4: The fitness function should be calculated.

Step 5: Find the likelihood of a fitness function solution.

Step 6: Using a selection process based on greed to find the best answer possible.

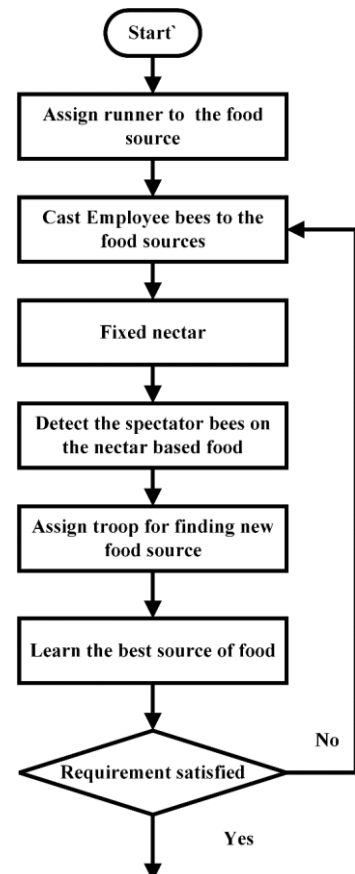


Figure 3-2: Flowchart of Artificial Bee Colony.

Step 7: Update new food sources with the help of probable fixes.

Step 8: Keep an eye on the threshold. Consider dumping honey if it's not plenty.

Step 9: Count the number of times a scout bee finds a new food source and the number of iterations that occur.

Step 10: If the halting requirements are met, the iteration may be stopped or the process can be restarted from step 3.

Now the steps are minutely depicted in the *Figure 3-2* flowchart.

### **3.3 Ant Colony Optimization for Continuous Domain (ACOR)**

Since its conception in 1992 by Dorigo, Ant Colony Optimization (ACO) has been used to solve a wide range of combinatorial optimization problems[29]. In 2008, Socha and Dorigo proposed ACO as a solution to the problem of continuous optimization. Foraging habits of ants have an effect on it[30]. Ants are highly sociable creatures. It's a team effort for a variety of reasons. Pheromones are a kind of hormone that people use to communicate while working. An animal's pheromone is a substance that influences the behavior of another animal of the same species. The term "behavioral-altering agents" is used to describe them. Many people don't know that pheromones may influence animals of the same species to engage in a variety of different behaviors.

#### ***3.3.1. Identifications of ACOR:***

An ACOR algorithm is just like another algorithm. In other words, it is a metaheuristic process that is stochastic and random. ACO was first developed for discrete optimization issues before being adopted for use in the more general continuous setting. It is essential that each choice variable in this class of optimization problems be genuine. Applied to the task of training neural networks for pattern recognition, it performed well on a variety of low-dimensional benchmark functions. However, ACOR and other ACO-based continuous algorithms have not been properly examined on widely available higher-dimensional benchmarks, such as those in the forthcoming Smart Computation magazine special issue. People respond favorably to our combined demeanor. Converge and develop an ideal solution using this tool. ACOR relies on pheromone-biased probabilistic selection of



solution components for its progressive evolution. Each construction step is assigned a different Probability Density Function by the ant.

### 3.3.2. *Exploitation and Exploration in ACOR:*

As a pseudo-random proportional approach, ACO is often referred to as such. On the basis of the beginning value and the modified value, ants migrate from node to node. The balance between exploration and exploitation may be found using the state transition rule.

Several tests are conducted to this end. An analogy can be that of pheromone intensity reinforcement on trips taken by ants to discover the optimal solutions. An ant's ability to deposit additional pheromone may be construed in this way. However, this rule of transition usually converges quite quickly. It's because pheromone strength has a significant impact on an ant's choice of the next node to visit, but heuristic information has very little effect on this decision.

### 3.3.3. *Mathematical Model of ACOR*

Gaussian kernel PDF was used to represent the search area's multi-promising area. The sampling shape of a Gaussian kernel pdf is more versatile than that of a simple Gaussian function. ACOR's solution archive file is where all of your solutions are stored. The solution archive is begun by producing n entries at random at the beginning of the program. Assuming that all of the solutions have been collocated, k will be maintained and the rest will be discarded. As a result, the specified standard deviation must be computed as follows:

$$\sigma_l^i = \xi \sum_{e=1}^n \frac{|s_e^i - s_l^i|}{n-1}; \quad i = 1,2,3 \dots n \quad 3.4$$

Here,  $\xi$  remains unchanged in all dimension and has an impact on evaporation rate of ACO.

### 3.3.4. Flowchart of ACOR

Step 1: Classify the storage capacity according to duration.

Step 2: Create a trail of pheromones and other cues.

Step 3: Initiate each ant from the beginning.

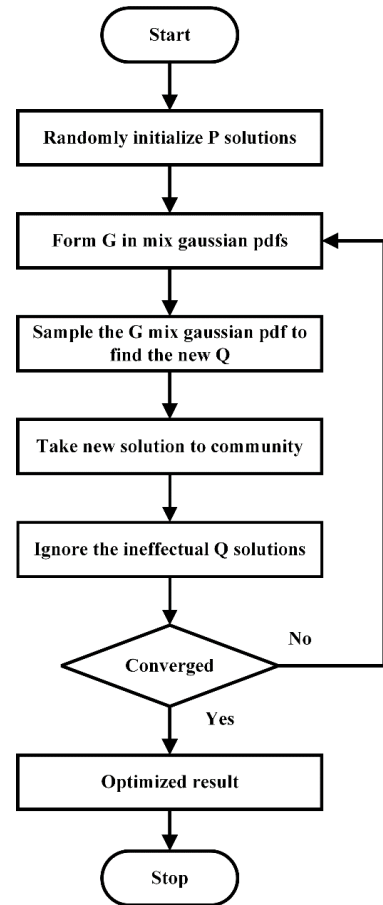
Step 4: The transition rule should be applied to each ant.

Step 5: On the basis of pheromones, choose the appropriate course of action.

Step 6: Calculate the local and global values for each cycle separately.

Step 7: Terminate or repeat step 3 if the end condition is met.

*Figure 3-3* shows the flowchart of ACOR.







*Figure 3-3: Flowchart of ACOR.*

# CHAPTER 4: AUTONOMOUS SYSTEM

## 4.1 UAS autonomy:

Self-flying drones are electronic robots powered by artificial intelligence and computer vision that use integrated circuits and programming to perform airborne tasks. *Figure 4-1* shows the concept of autonomy from start to finish[31]

AUTONOMY FROM START TO FINISH			
LEVEL	CONCEPT	DEFINITION	WHO'S IN CONTROL
0	Human Operation	The operator <b>controls the machine at all times.</b>	
1	Automation (Function-specific)	The operator <b>has overall control of the machine</b> and is responsible for its safe operation, but can transfer limited control over a specific function (like moving a bucket or blade) to the machine.	
2	Semi-autonomous	The machine accomplishes a <b>subset of its defined tasks</b> without operator interaction. The operator performs the remaining tasks.	
3	Autonomous	The machine accomplishes <b>all its defined tasks</b> without operator interaction and is responsible for all safety-critical earthmoving functions.	

*Figure 4-1: Autonomy from start to finish [31]*

It used to be the case that unmanned aerial vehicles (UAVs) and autonomous drones (drones) were solely utilized by the military to carry out espionage or military-grade operations.

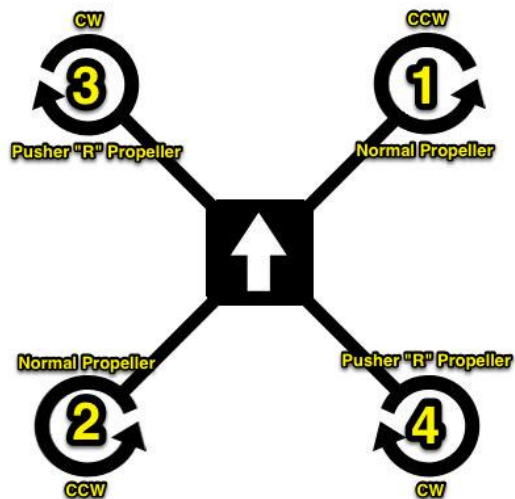
Commercial drones for commercial use have also grown very popular in recent years and are being developed by corporations throughout the globe. Drones that are able to fly on their own have no limitations. Whether they're flying inside, underground, in the skies, or somewhere else, they're smart and competent enough to serve one's needs. The military has been using autonomous drones for a long time now. Due to their high price tags (tens of thousands, if not hundreds of thousands of dollars), these self-piloting drones were originally designed only for military usage.

Computer vision is used by autonomous drones to make judgments about movement and fly without the need for a pilot. Using high-quality image sensors and computer vision, autonomous drones gather images and videos from their surroundings, convert them to digital signals, and produce maps using GPS. To comprehend their environment, computers employ computer vision, which is the process of converting images and movies into digital signals. As a starting point, computer vision is all that's required. Here are some things to consider to answer why are current autonomous drones so capable of navigating through a wide variety of obstacles at fast speeds, and how did they get to this point. One is the AI or artificial intelligence. By applying artificial intelligence approaches and algorithms, today's consumer and industrial grade autonomous drones are able to distinguish between bigger and smaller obstructions. They devise new routes of travel and carry out the duties for which they were built. This is the fundamental operation of a self-flying drone, which does not need the assistance of a pilot or remote control. It's awe-inspiring, and drone autonomy will continue to improve over time.[32]

## 4.2 Quadcopter Flying Theory

In a quadcopter, four motors are attached to the frame's four limbs. Each motor's rotational orientation is designed to offset the torque created by the motor on the other side, as seen in *Figure 4-2*[33].

As a result of the torque effect, this is how the quadcopter maintains its position in the sky. There are two sorts of propellers because they are spun in opposite directions by the engines. Pusher propellers are used to provide thrust for motors that rotate counterclockwise. To create thrust, puller propellers are used on both clockwise and counterclockwise rotating engines.



*Figure 4-2: quadcopter motor spin direction [33]*

### *Hover:*

To hover in place, quadcopter requires that:

- I. There are no varying speeds among the motors.
- II. The rotation speed must be sufficient to provide lift that counteracts the quadcopter's weight.
- III. In order to offset the quadcopter's weight, the rotation speed must be high enough to provide lift. A quadcopter's body should be able to handle the torque exerted by all of its motors.

### *Gaining and Losing Altitude:*

All four of the quadcopter's motors must spin faster at the same time in order to acquire altitude. Similarly, to lower altitude, the rotational speed of all four motors must be reduced concurrently.

### *Pitch:*

The quadcopter's pitch control instructs it to yaw forward or yaw back. The speed of the quadcopter's rear motors must be increased in relation to the speed of the front motors in order for it to pitch forward. This causes the quadcopter's nose to be lowered, causing it to go forward. The speed of the front motors must rise in relation to the speed of the rear motors in order to pitch backwards.

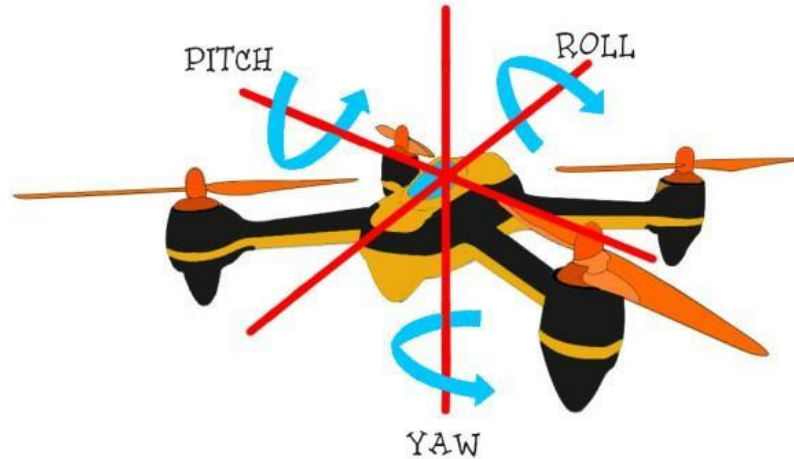
### *Roll:*

The quadcopter's roll control instructs it to travel from left to right. The quadcopter can only roll to the right if the left motor speed is increased in relation to the right motor speed. Sideways movement will occur as a consequence of the quadcopter rolling left. A similar ratio must be used for the quadcopter's right motors to the quadcopter's left motors in order to roll the quadcopter left. When you roll the quadcopter left, you'll see a leftward movement.

### *Yaw:*

The yaw is the quadcopter's z-axis rotating movement. In order to do this, two similar directional spinning motors must be increased or decreased in order to accomplish this.

Increased torque causes the quadcopter to rotate in the direction of that torque. There is an image in



*Figure 4-3* exhibiting pitch, roll and yaw.

*Figure 4-3: Movements on Quadcopter's Axes [33]*

### **4.3 UAV Manufacturing**

*Gathering parts:*

To build a high-quality autonomous drone, one needs the following components and equipment.

The drone's structure with quad-propellers, serving as its foundation.

- It is also crucial to consider the motors. They'll power the rotors and allow the drone to take off and land.
- The motors will be powered by the battery.
- The drone's speed is controlled by an electrical speed controller.
- In order to make the drone self-driving, that needs a circuit board.
- Using an RC controller to configure the drone's flight direction and calibrate it is essential.

Assembly:

Figure 4-4 depicts the usual wiring of a quadcopter in visual form. If anyone is unfamiliar with quadcopter wiring, this graphic provides an excellent image of how it should be done. Using the illustration, it's clear that each ESC should be attached to one of the two motors. A third cable connects the flight controller to the ESCs, which in turn get power from the Lipo battery. Signals are sent by this third wire, which links to the ESCs from the flight controller. The ESC (Electronic Speed Controller) receives a PWM (Pulse Width Modulation) signal from the flight controller that tells it what RPM to run the motor. It is shown in Figure 4-2, the numbers on each propeller match to the signal output number on the FC. The motors should be attached to the flight controller in a certain manner as it is pre-determined by the software developer.

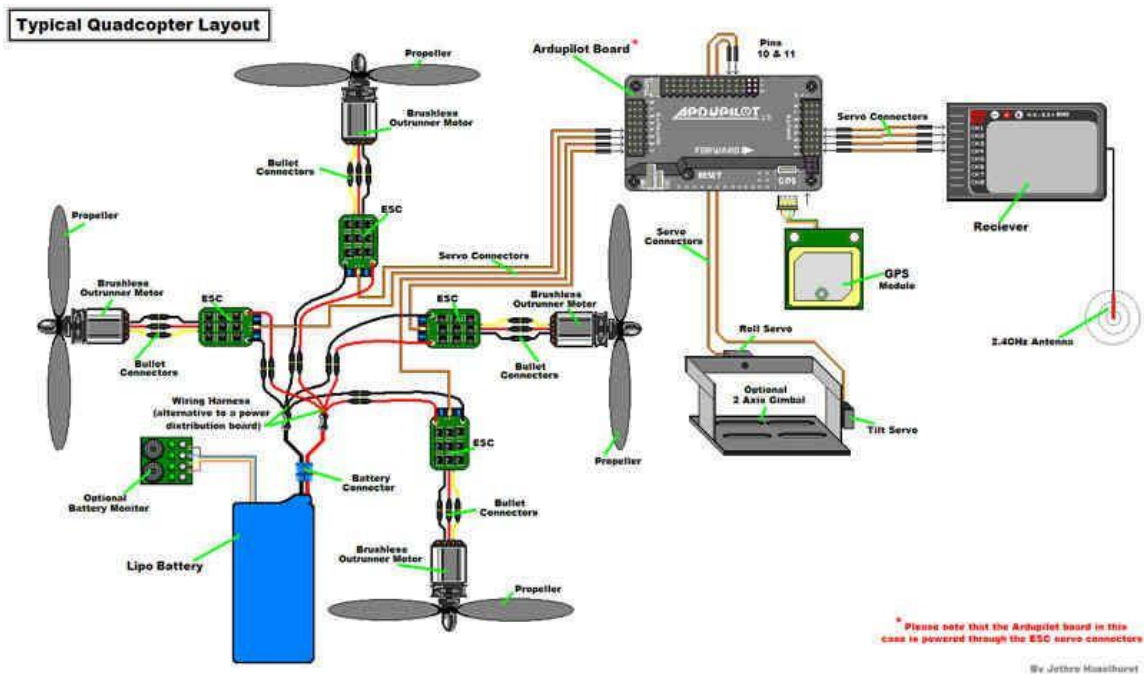


Figure 4-4: Typical Quadcopter Wiring Layout [35]

When it comes to flight controller, this makes sense since it must be able to know the position of each motor it is driving, along with its rotational axis.

The illustration doesn't illustrate how the flight controller gets its power. The Pixhawk's power distribution board was included in our arrangement. Connecting power and ground

wires to the PDB included into the frame connects this board to the battery, which then delivers 5V to the Pixhawk. Connecting the GPS and RC receivers to the flight controller is as simple as inserting the appropriate connections into the flight controller's relevant sockets.

#### *Coding:*

There are several steps in this process, but coding is crucial. Any open-source coding platform, such as PX4 or MultiWii, will be used for the project, and it has to ensure that the setting is correct.

Once the circuit specifics and essential autonomy functions are provided, all the coding and installation will be completed into the frame's corresponding circuit.

The RC controller will then be programmed in line with the drone circuit using the program or a template found on the internet.

#### *Flying:*

In order to begin flying your autonomous flight drone, both the hardware and software development and integration must be completed.

Using an RC controller and the right drone settings, pilot need to fine-tune the drone's settings. The calibration process is now complete, and the drone is ready to go to the skies.[34]



## **CHAPTER 5: SOFTWARE STACK**

### **5.1 Flight Controller Firmware**

APM and PX4 are two of the most frequently utilized controller firmware in the industry. The Ardupilot Mega (APM) is an IMU autopilot based on the Arduino Mega platform of professional grade. This means multirotor helicopters, fixed-wing aircraft, and standard helicopters may all be controlled by this autopilot. Fully autonomous stabilization, waypoint-based navigation, and two-way telemetry are all possible with this autopilot. There are a number of other light controllers that can run APM, including the Pixhawk, but it was initially developed for the Arduino Mega platform. Like APM, PX4 is part of the Drone code Effort, which is an open-source project that aims to provide a complete platform for unmanned aerial vehicles (UAVs) from start to finish.

As the DroneKit-Python interface with Pixhawk, we chose to utilize it instead of the APM flight controllers designed for mission planner. There was a problem at this point. It works fine with the old APM hardware, however the recent stable version 3.4 of APM, is not compatible with the Pixhawk Mini because of the newer hardware platform's design. For now, we were able to get past this problem by switching to the Pixhawk 2.4.8 or Pixhawk Cube. For our experiments, APM 3.6 on the Pixhawk Cube seems to operate well enough and some of the features, like as monitoring the power consumption level and RTH the drone when it becomes low, are working as expected[35].

### **5.2 Installing and Configuring APM Firmware**

In order to install and configure the APM firmware, we had to have a GCS or Ground Control Station installed on our desktop or laptop. APM provides social support for two GCS initiatives. If someone has a Windows system, Mission Planner is advised, whereas if anyone with a Linux distro or MacOS, APM Planner 2 is recommended. Mission Planner is the original GCS program for APM installation. APM Planner 2 is a cross-platform successor to Mission Planner. We've noticed that there is minimal difference between the two programs after playing with both. anyone may install a variety of APM firmware from the GCS software. Even after a fresh firmware installation, it provides a wizard that will guide someone through the process of configuring.

Before the UA is ready for test flight after a new firmware installation, a number of tasks must be completed. Frame type, accelerometer calibration, compass calibration, radio calibration, failsafe configuration, and flight mode setting are all included in these required configurations. With the GCS program, it's easy to set up the frame type, calibrate the compass, and calibrate the accelerometer.

It's a bit more difficult to calibrate the radios. Binding the RC transmitter and setting the controller which channels will be utilized for certain instructions and what the thresholds are for each of those channels during this stage. It's possible to adjust the current lighting mode using a three-position switch on the controller's front right, which corresponds to Channel 5. As of right now, the three flight modes (STABILIZE, ALT HOLD, and UP) are all set to their default settings (LAND).

A manual flight mode, STABILIZE, is similar to this mode. In stabilize mode, pilot is in complete control of the drone, and the flight controller's only purpose is to maintain the drone's stability. Stabilize flight mode is the most difficult mode to manage the copter, and we have had a number of rough landings when flying in this mode. ALT HOLD mode is much easier mode for flying the drone. To keep the copter at the present height, ALT HOLD lets pilot manage the rest of the plane's functions. ALT HOLD keeps trying to hold the current altitude, but in windy conditions there is still some drift, particularly. If this happens, pilot may still use the throttle joystick to command the flight controller to raise or decrease altitude. However, the rise or drop rate will be extremely slow which can be changes from default values assigned by the software for safety. The LAND flight mode should be utilized as the final flight mode configuration. Landing mode disables the joystick's throttle control, allowing the copter to level out and gently drop to the earth. In case the operator loses control of the copter, the LAND button may be used to land it without fear of destroying the aircraft. The failsafe configuration is the last essential configuration to be put up. The failsafe setting tells the flight controller what to do if the battery voltage goes too low or the RC transmitter loses signal. Faulty landings have been pre-set in our system to land in the present location. Pilot may fly the quadcopter using the RC transmitter after the APM software has been loaded and verified.

### 5.3 DroneKit-Python

A program on a companion computer was required in order to transmit orders to the flight controller so that we could independently fly our drones. We used DroneKit-Python for this.

As a result, developers may write programs that run on the companion computer and connect with the APM flight controller over a low-latency link using DroneKit-Python (DK-Python).

MAVLink, or Micro Air Vehicle Link, is the method through which the API connects with drone. It allows programmatic access to a connected condition, vehicle's telemetry, and parameter information, as well as direct control over vehicle movement and operation and mission management. With the classes and methods provided by the DroneKit-Python API, a script

- I. can connect to a car (or a group of cars).
- II. Inquire about the vehicle's current state/telemetry/parameters.
- III. Receive notification of state changes in an asynchronous manner.
- IV. To direct a UAV to a certain location, use a gyroscope (GUIDED mode).
- V. control the movement of the UAV and associated hardware (GUIDED mode) by sending arbitrary custom messages.
- VI. Set up and maintain missions based on waypoints (AUTO mode).
- VII. The RC channel settings will be overridden.

DroneKit-Python may be learned by anybody with a working knowledge of Python. The API documentation is very excellent, providing step-by-step instructions and examples for all of the built-in features. [36] contains all of the relevant documents.

## CHAPTER 6: PROBLEM FORMULATION

### 6.1 Environment and path representation:

Creating a route that avoids obstacles and meets the constraints in a complicated 3D environment is the purpose of the algorithm. A route in the 3D mission space is defined as  $(x, y, z)$  as the coordinates of a point in the mission space.

So, the task space may be described as

$$\{(x, y, z) \mid X_{min} \leq x \leq X_{max}, y_{min} \leq y \leq y_{max}, Z_{mij}^{\beta} \leq z \leq Z_{max}\} \quad 6.1$$

The cost of path is evaluated using the cost function which takes a collection of 3D points as input and outputs the cost value of the input path.

Mathematically obstacles are defined as geometric cylinder with infinite height, such as

Obstacle  $i = (x_i, y_i, r_i)$  where  $x_i, y_i$  denotes the position of the obstacle and  $r_i$  denotes the radius and the value of the function defines the associated cost. For a no-fly zone scenario, cost is regarded as infinite.

### 6.2 Cost Function:

A cost function is a function that transforms an event or the values of one or more variables into a real number that intuitively displays some "cost" connected with the occurrence.

A cost function is used to determine how far off the mark the model is in terms of establishing a relationship between the input and output. It informs you of how poorly your model is doing and forecasting[37].

When fitting a linear regression model, a straight line is used to fit the model. This is accomplished by the use of the following equation for a straight line:

$$\text{Output} = a * \text{Input} + b \quad 6.2$$

Notice that the variables  $a$  and  $b$ , which represent the point at which a line intersects the x-axis and the slope of a line, respectively, may have different values (variables).  $a$  is a changeable value (variable), and  $b$  is a changeable value (variable). If the variables are not adequately optimized, you will initially obtain a line that may or may not be a good match for the model. As you refine the parameters of the model, you will get the best possible fit

for certain variables. Optimization If the data is perfectly fitted, a straight line will be drawn across most of the data points, disregarding noise and outliers in the process. The minimal feasible Root Mean Squared Error of the model, which can then be obtained by deducting the estimated values from the actual values, will serve as the cost function for the Linear Regression model. Taking the smallest of these error values, the cost function will be defined as Furthermore, the cost function will be

$$\text{Cost Function } (J) = \frac{1}{n} \sum_{i=0}^n (h_{\theta}(x^i) - y^i)^2 \quad 6.3$$

### 6.2.1. Objective functions:

For this work objective function consists of five objectives, they are – path length, time required to traverse the planned path, collision cost, the smoothness of the path and the energy consumed. The final evaluation function can be represented as follows:

$$F = F_{\text{Length}} + F_{\text{Time}} + F_{\text{Smoothness}} + F_{\text{collision cost}} + F_{\text{energy consumption}} \quad 6.4$$

### 6.2.2. Path length:

Total Path length s nothing but the cumulative sum of the length of all the path segments.

For a given path, Path Length is –

$$\sum_{i=0}^{i=n-1} \left( \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2 + (z_{i+1} - z_i)^2} \right) \quad 6.5$$

### 6.2.3. Traverse Time:

The time required for the quadrotor to traverse through the optimized path is determined by flight controller firmware. The flights are conducted in AUTO. The behavior of the flight controller in “AUTO” mode is determined by parameters that have the “WP\_NAV” prefix. The flight parameters affect flight time –

**WPNAV\_SPEED:** Waypoint Horizontal Speed Target -When flying on a WP mission, the aircraft will attempt to maintain this horizontal speed in centimeters per second.

*WPNAV\_SPEED\_UP*: Waypoint Climb Speed Target - During a WP mission, the speed in centimeters per second (cm/s) that the aircraft will strive to maintain while ascending is defined.

*WPNAV\_SPEED\_DN*: Waypoint Descent Speed Target - During a WP mission, the speed in centimeters per second (cm/s) at which the aircraft will try to hold its position while falling is defined.

*WPNAV\_ACCEL*: Waypoint Acceleration - Specifies the horizontal acceleration in centimeters per second per centimeters per second utilized during missions.

*WPNAV\_JERK*: Waypoint Jerk - The horizontal jerk, measured in milliseconds per second, is employed throughout the mission.

These flight parameters are set before flight. Using these parameters as constraints, minimum possible flight time required for traverses is calculated.

#### **6.2.4. Path Smoothness:**

In order to properly describe a twice differentiable plane curve, we may use  $Y(t) = \{x(t), y(t), z(t)\}$ . In this context, appropriate implies that the derivative  $d/dt$  is defined, differentiable, and not equal to the zero vector on the domain of definition of the parametrization. A parametrization like this one yields a signed curvature where the prime numbers relate to derivatives of  $t$ . The curvature  $\kappa$  is thus-

$$\kappa = \frac{|x'y'' - y'x''|}{(x'^2 + y'^2)^{\frac{3}{2}}} \quad 6.6$$

The curvature value is found at every point on the path and the summation of curvature values is taken as the path smoothness cost. Without smoothness in the cost function, the path generated by the optimizer may not be suitable for traverse.

6.2.5. **Collision Cost:**

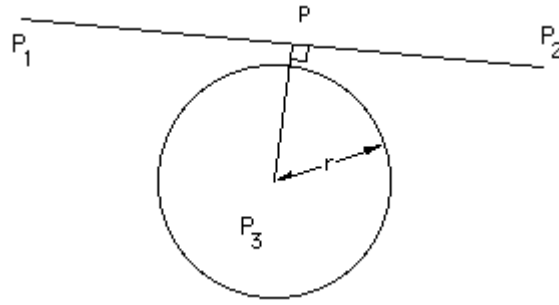


Figure 6-1: Collision cost

A set of points  $P(x, y)$  along a line represented by two points  $P_1(x_1, y_1, z_1)$  and  $P_2(x_2, y_2, z_2)$  as shown in Figure 6-1 is described by,

$$P = P_1 + u(P_2 - P_1) \quad 6.7$$

Alternatively, for each coordinate

$$x = x_1 + u(x_2 - x_1); y = y_1 + u(y_2 - y_1); z = z_1 + u(z_2 - z_1) \quad 6.8$$

A cylinder positioned at  $P_3(x_3, y_3)$  with a radius of  $r$  is an infinite height can be represented by

$$(x - x_3)^2 + (y - y_3)^2 = r_2 \quad 6.9$$

Putting the linear equation into the equation of the cylinder's equation, it gives a quadratic equation which can be described as

$$au_2 + bu + c = 0 \quad 6.10$$

where:

$$a = (x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2$$

$$b = 2[(x_2 - x_1)(x_1 - x_3) + (y_2 - y_1)(y_1 - y_3) + (z_2 - z_1)]$$

$$c = x_3^2 + y_3^2 + z_3^2 - 2[x_3x_1 + y_3y_1 + z_3z_1] - r_2$$

This quadratic's solutions may be found by the following equation

$$\frac{-b \pm \sqrt{(b^2 - 4ac)}}{2a} \quad 6.11$$

Expression inside the square root determines the exact behavior.

$$(b^2 - 4ac)$$

6.12

If this value becomes less than zero then the path will never intersect the cylinder.

If the value is equal to zero then the path is a tangent to the cylinder crossing it at a single point, namely at  $u = -b/2a$ .

If the value is greater than zero the line intersects the cylinder at two points.

For every path segment, distance of the edges of the segments, and distance of the line from the center of the cylinder is calculated. If the line segment intersects the cylinder, inverse of the distance is calculated as a collision cost. The same process is done for all the line segments and summation of all collision cost is regarded as the final collision cost for a particular path

#### 6.2.6. Energy Consumption:

In order to calculate energy consumption from a given path, a mathematical model of the quad copter is generated.

In order to generate the quadcopter, two reference frames are defined – the body frame and the inertial frame. the roll, pitch, and yaw angles for quad model frame can be described as  $\theta = (\phi, \theta, \psi)^T$ , with comparable angular velocities equal to  $\dot{\theta} = (\dot{\phi}, \dot{\theta}, \dot{\psi})^T$ . Similarly, the velocity and position of the quadcopter in the inertial frame can be described as  $\dot{x} = (\dot{x}, \dot{y}, \dot{z})^T$  and  $x = (x, y, z)^T$ , respectively. Here, the angular velocity vector  $\omega \neq \dot{\theta}$ . There are two types of rotational motion: rotational velocity and rotational acceleration. The latter is defined as the time derivative of rotational pitch, yaw, and roll. To get an angular velocity vector from these angular velocities, we may apply the following relationship:

$$\omega = \begin{bmatrix} 1 & 0 & -s_\theta \\ 0 & c_\phi & c_\theta s_\phi \\ 0 & -s_\phi & c_\theta c_\phi \end{bmatrix} \dot{\theta} \quad 6.13$$

Using a rotation matrix R, the body and inertial frame may be linked together by way of the body frame. Utilizing the ZYZ Euler angle conventions and redoing roll, pitch, and yaw in a stepwise fashion, this matrix is generated[38].



$$R = \begin{bmatrix} c_\phi c_\psi - c_\theta s_\phi s_\psi & -c_\psi s_\phi - c_\phi c_\theta s_\psi & s_\theta s_\psi \\ c_\theta c_\psi s_\phi + c_\phi s_\psi & c_\phi c_\theta c_\psi - s_\phi s_\psi & -c_\psi s_\theta \\ s_\phi s_\theta & c_\phi s_\theta & c_\theta \end{bmatrix} \quad 6.14$$

for A specific vector  $\vec{v}$  in a frame structure corresponds to a vector  $R\vec{v}$  in an inertial reference frame.

### 6.2.7. Equations of Motion:

For a quadcopter, acceleration is caused by propulsion, linear friction, and gravity when considered within an inertial frame of reference. Rotation matrix  $R$  may be used to transfer the inertial thrust vector from the quad model body frame to its inertial frame equivalent. Thus, the linear motion for the model can be described as –

$$m\ddot{\vec{x}} = \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} + RT_B \quad 6.15$$

where  $T_B$  is the thrust vector,  $\vec{x}$  is the position of the quadcopter in the body frame and  $g$  is the acceleration due to gravity. In contrast to having the linear equations of motion in an arbitrary frame, having the rotational equations of motion in a body frame is useful, since it allows rotations to be represented around the quadcopter's center of gravity rather than the inertial center. Euler's equations, which reflect rigid body dynamics and are used to generate the rotational equations of motion, are used to represent the dynamics of rigid bodies. When Euler's equations are stated in vector form, the following is how they are written:

$$I\dot{\omega} + \omega \times (I\omega) = \tau \quad 6.16$$

where  $\tau$  is a vector of external torques,  $I$  is the inertia matrix, and  $\omega$  is the angular velocity vector. It can be rewritten as-

$$\dot{\omega} = \begin{bmatrix} \dot{\omega}_x \\ \dot{\omega}_y \\ \dot{\omega}_z \end{bmatrix} = I^{-1}\{\tau - \omega \times (I\omega)\} \quad 6.17$$

In a simplified representation, the quadrotor may be represented by two thin uniform rods that are crossed at the origin and have a point mass (motor) at the end of each. The diagonal inertia matrix becomes-

$$I = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \quad 6.18$$

As a consequence, we derive our final conclusion for the rotational equations of motion of the body frame.

$$\dot{\omega} = \begin{bmatrix} \tau_{\phi} I_{xx}^{-1} \\ \tau_{\theta} I_{yy}^{-1} \\ \tau_{\psi} I_{zz}^{-1} \end{bmatrix} - \begin{bmatrix} \frac{I_{yy} - I_{zz}}{I_{xx}} \omega_y \omega_z \\ \frac{I_{zz} - I_{xx}}{I_{yy}} \omega_x \omega_z \\ \frac{I_{xx} - I_{yy}}{I_{zz}} \omega_x \omega_y \end{bmatrix} \quad 6.19$$

After deriving the equations of motion, a quadcopter object was made in python which followed the above mention equations. a quintic polynomial trajectory generator was used to generate a trajectory from the given path. A PD controller was used to calculate control commands in terms of thrust and torque required follow the trajectory. Using the output of PD controller power required was logged at every timestep. From the calculated power data, energy required for the path was estimated.

### 6.3 Proposed Path Planning Algorithm:

The following is proposed path planning algorithm:

Initial guess paths are created by linking the start and target locations, as well as the sample point along that line, according to a preset resolution value. It was decided that random noise should be added to the sample points to give the straight-line route a small change, and this was thought to be the first answer.

The original route was used to figure out the limits of the optimized route.

The original solution was utilized as an input to the optimization process, which produced the final solution. The Needler Mead optimizer was used to find the best way through the forest. As an objective function, it was necessary to use the cost functions listed above to figure out how much the route would cost.

Finally, the improved route was stored as a csv file together with the related cost values so that it could be evaluated further.

# CHAPTER 7: SYSTEM DESIGN

## 7.1 Airframe

**7.1.1. Functional Description:** The Unmanned Aircraft used for this experiment is a quad rotor drone which is powered by a 3200 mAh battery. The drone utilizes 4 high torque motors which are connected to 4 Electronic Speed Controllers (ESC). The entire system is then connected to the Pixhawk 2 cube which acts as the heart of the entire system.

**7.1.2. Rationale for Selection:** The simple design of the quadcopter is useful for a lot of reasons. This makes the system easy to design and repair in case of any damage. The system is also very stable which makes it easier to perform the missions since there are already an exceeding number of variables which might alter the results of the experiment. Fixed wing aircrafts would not be able to take the sharp turns which are possible via a rotorcraft. The quadrotor drone also has the ability to hover in air which is a huge advantage compared to fixed wing configuration.

## 7.2 Navigation & Mission Control

**7.2.1. Functional Description:** A PixHawk 2.1 as shown in *Figure 7-1* is used for navigation of the UA, the flight controller (PixHawk cube) runs Arduplane firmware onboard[39]. Data collected from the on-board sensors: GPS, IMU, Airspeed sensors, Lidar data, barometer reading, is fed into the flight controller in order to determine the state of the UA[40]. The ground control station is connected to the flight controller using a RFD 868 UX Telemetry module. “Mission Planner” will be used as ground control software. It will be used to give commands to the UA like - ‘Takeoff’, ‘Navigate to Waypoint’, ‘Land’ etc. from the ground control station. Mission control can be used to pre-plan flights as well as giving commands in real time. Each of the commands will take necessary parameters as arguments and implement them, i.e., take-off will take height as an argument.



*Figure 7-1: PIXHAWK [39]*

The sensor outputs can also be observed from the Ground Control Station, to determine any discrepancy. “Radio link AT-9S” is used as the “master controller” and is used to take manual control of the UA in case something goes wrong. There is a manual FTS trigger on the master controller and an option to switch flight modes.

**7.2.2. Rationale for Selection:** PixHawk 2.1 cube is an all-in-one FMU. It has Triple redundant vibration damped IMU with dual barometers and support for up to 3 GPS modules. It also has a heating system to maintain consistent performance. It also offers an adequate number of IO ports for servos, ESCs and sensors with various communication standards (UART, i2c, SPI etc.). An Extended Kalman Filter (EKF) algorithm is used to estimate vehicle position, velocity and angular orientation based on gyroscopes, accelerometer, compass, GPS, airspeed and barometric pressure measurements. Based on these data, the UA will navigate accordingly. It offers separate power lines for FMU and servos. RFD 868 UX Telemetry module is chosen as it offers wireless connectivity with the flight controller, transmits data at compliant frequencies, offers adequate range (>40 km) and data rate. It is also highly configurable to meet any legal and mission requirements. Such as power consumption can be reduced to meet regional law.

Radiolink at 9 offers a mode-2 configuration for manual control. It uses FHSS technology and operates at 2.4GHz and its transmitter consumes less than 100 mW. It has an operating range of 1 km. It also offers multiple configurable switches and knobs.

### 7.3 Sensors

**7.3.1. Functional Descriptions:** The used flight controller PixHawk has built-in -- 9 axis IMU (Inertial Motion Sensor), which consists of an accelerometer, gyroscope, and magnetometer. It'll provide the acceleration, rate of rotation and orientation of the UA. PixHawk also has a barometer, which is used to measure the height of the UA with respect to sea level. The IMU sensor is isolated, minimizing the interference due to vibration. The system utilizes the following external sensors:

a) *GPS and Compass:* The GPS sensor is used to determine the position of the UA. Magnetometer is used to provide the heading of the UA with respect to magnetic



Figure 7-2: HERE 2 GPS [39]

North. It will send the longitude and latitude, heading and the ground velocity of the UA. *Figure 7-2* shows the Here 2 GPS used for our mission.

b) *Airspeed Sensors*: Consists of a pitot tube, which utilizes Bernoulli's law to measure the airspeed. The sensor also measures temperature to calculate the true airspeed from induced air speed.

c) *Lidar*: the sensor uses laser to measure the height of the UA, with respect to the terrain below. Shown in *Figure 7-3*

d) *Power module*: It monitors the onboard battery voltage, the current draw and power consumption of the UA.



*Figure 7-3: LiDAR Lite V3[39]*

**7.3.2. Rationale for Selection:** Each of the sensors described above helps the flight controller to measure the value of necessary state variables which is essential for making the state estimation of the UA more reliable in order to ensure robust automatic operation.

GPS with integrated compass makes it easier to place both of these sensors away from the source of noise (mainly from power cables, esc cables). Airspeed sensor offers true airspeed of the UA and aids the TECS controller which is used for height and speed. The downward looking lidar measures height from the terrain below and aids the flight controller to follow terrain and obstacles below. It also aids in landing and take-off.

The power module logs and monitors power consumption, battery voltage and current draw. These are necessary for safe operation of the UA.

## **7.4 Autonomy**

**7.4.1. Functional Description:** For automatic operation, Mission Planner will be used. It is a free, open-source, UI application for the arduplane firmware. Mission planner communicates with the autopilot (the flight controller board- Pixhawk cube) using telemetry modules. The pair telemetry modules (one connected to the GCS computer and another one with UA's flight controller) acts as a virtual USB cable between the GCS and the UA. Mission planner offers various interfaces such as - data monitoring, automatic mission planning, hardware and software setup of the GCS and UA, SITL simulation etc. A fully automatic mission can be uploaded using the planning interface of Mission Planner's UI.

A custom algorithm will be used to plan the optimal path to maximize the desired property of the path (example- fastest time, least energy usage, shortest path, max area coverage etc.). The algorithm will take the flight dynamics and the payload into consideration.

The mission planning procedure is described below –

For the core mission task, the UA will have to navigate through the given waypoints and supply the given cargo and then proceed to complete the climb and glide task. Since all the mandatory waypoints will be given, a python script will be used to plan the most optimal path which will consider the flight dynamics of the UA, minimizing time and energy consumed, maximizing the accuracy of cargo drop and landing.

After dropping the cargo, the UA will proceed to launch through a direct path and do the climb and glide mission. For this portion of the mission, preconfigured waypoints will be set in such a way that the glide time is maximized and proper approach for landing is maintained.

For the Speed Trial and the marker identification task path will be planned to minimize flight time and maximize average speed. For the Area Search task, a path will be planned in such a way so that the maximum area is covered in the shortest time.

After uploading the output of the algorithm, which are desired waypoints, a mission start command will be issued from the GCS and the UA will start executing the uploaded mission. While the UA completes the uploaded mission, flight critical parameters of the UA will be monitored using the GCS, which will display data received over radio telemetry.

#### **7.4.2. *Rationale for Selection:***

Mission planner along with Arduplane was chosen for the following reasons:

- a) Interactive UI with Google map support
- b) Easy to understand commands from drop-down menus
- c) Mission log files visualizer
- d) Real time configuration of autopilot settings for UA
- e) SITL support

- f) Geo-fencing
- g) Configurable Failsafe options
- h) Open-source community support

The Controller board running the free plane firmware (Arduplane) provides full autonomous capabilities to any tilt rotor fixed wing craft. The firmware is also capable of enabling hovering and cruising of tilt-rotor fixed wing aircrafts in different configurations.

The firmware works with a variety of GCS software. Mission Planner was chosen as it offers an all-around UAV solution for programming and mission operations incorporating support for hundreds of 3D waypoints, automatic takeoff and landing as well as sophisticated mission planning. The easily configurable package can be modified for custom education and research applications as well.

## CHAPTER 8: EXPERIMENT DESIGN

### 8.1 Introduction

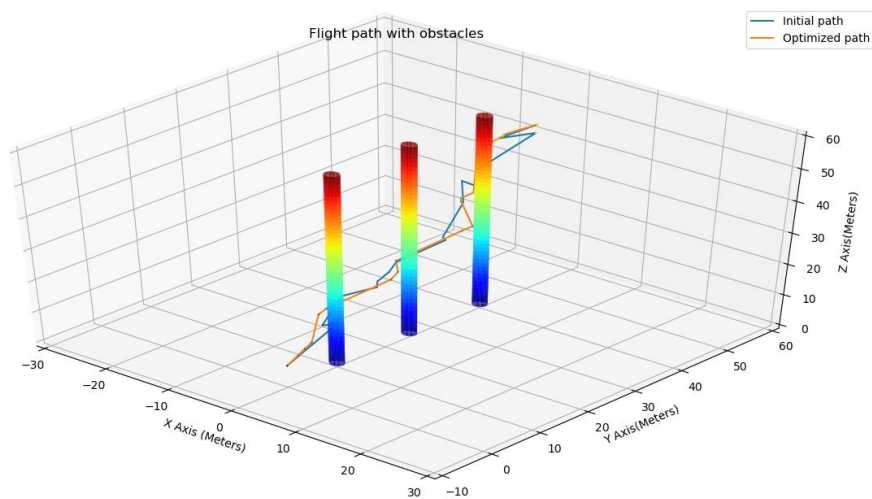
For each given environment, the optimization algorithm was run 3 times. This gave way to 3 different optimized paths being delivered by the system with 3 different costs. The UA was then fed with each of the optimized path and the test flight was carried out. Hence, the experimental data for each of these 3 flights was obtained. The procedure was repeated for 5 different environments, with 3 flights for each environment, that is a total of 15 flights.

### 8.2 Methodology

**8.2.1. Environment Creation:** A hypothetical environment was created to perform the experiment. The environment consisted of 3 obstacles of cylindrical shape with infinite height in XYZ space which represented the potential danger/red zones for the drone to fly in.

**8.2.2. Path Generation:** The start and end points of the mission were first defined in the system. A random path was generated afterwards.

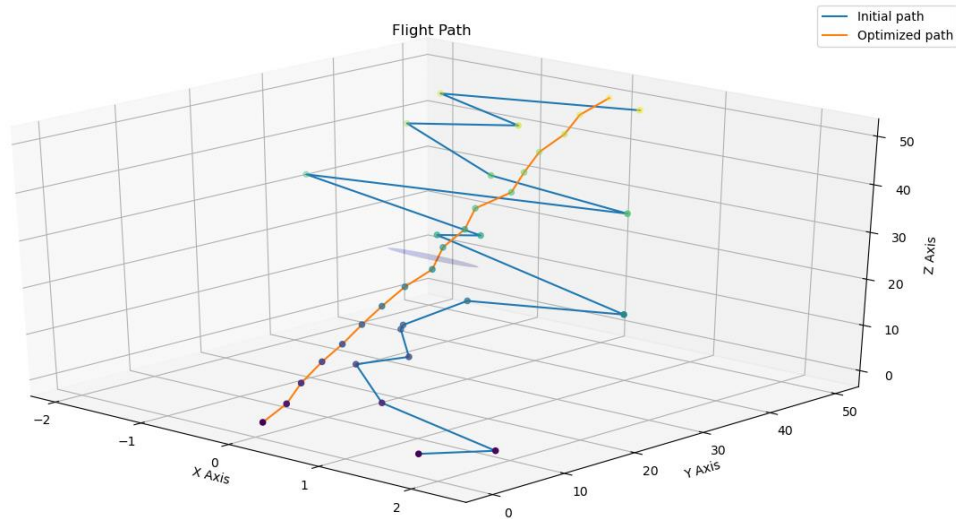
The path shown in *Figure 8-1* was then optimized using our homegrown cost function and optimization algorithm[41].



*Figure 8-1: 3D environment*



**8.2.3. Reference Frame Transformation:** As we can see in *Figure 8-2*, the optimized path was generated in the XYZ coordinate system. This was then transformed to real-life latitude longitude altitude (LLA) frame. This path was then converted into a mission file with dot way point extension using a python script.



*Figure 8-2: Optimized path considering length, time and energy consumption*

**8.2.4. Mission Flight:** By running the Arducopter firmware and the Mission Planner UI, the path can be directly uploaded to the UA. The UA takes off and performs the mission, taking the path generated by the computational methods and optimization algorithm. At the end of the mission, a log file is generated which contains all the critical information about



*Figure 8-3: Generated path in mission planner*

the flight. *Figure 8-3* shows the optimized path plotted in mission planner and *Figure 8-4* shows a picture of a demo taken in our university playground.



*Figure 8-4: Test flight*

### **8.3 Results Evaluation**

The results were evaluated from the log files received from the UA after the completion of the mission. The experimental data was extracted from the log files in the form of total distance flown, flight time and battery energy consumption as shown in *Figure 8-5* and in *Figure 8-6*. The theoretical data was obtained from the output of the optimization script. This also included the total distance to be flown, flight time and theoretical battery consumption. The theoretical data also included the cost of each flight as per the optimization algorithm. The results were then compared to see whether the theoretically low-cost path actually gave the best possible path as defined by distance, time and energy consumption experimentally.

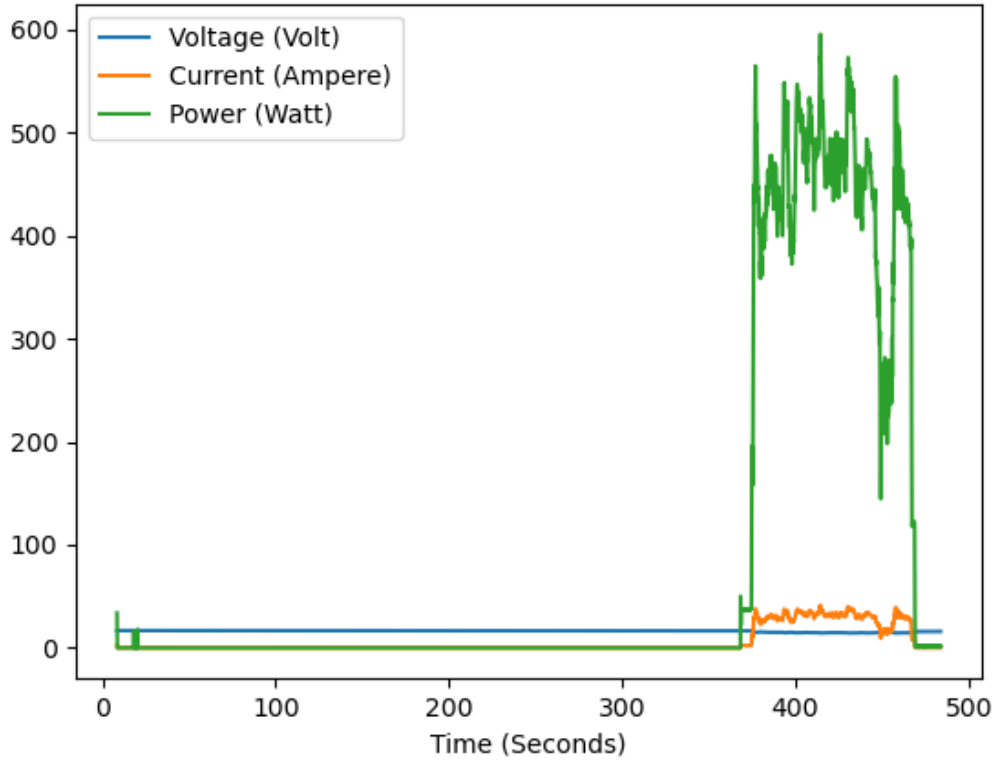


Figure 8-5: Power consumption graph from data log

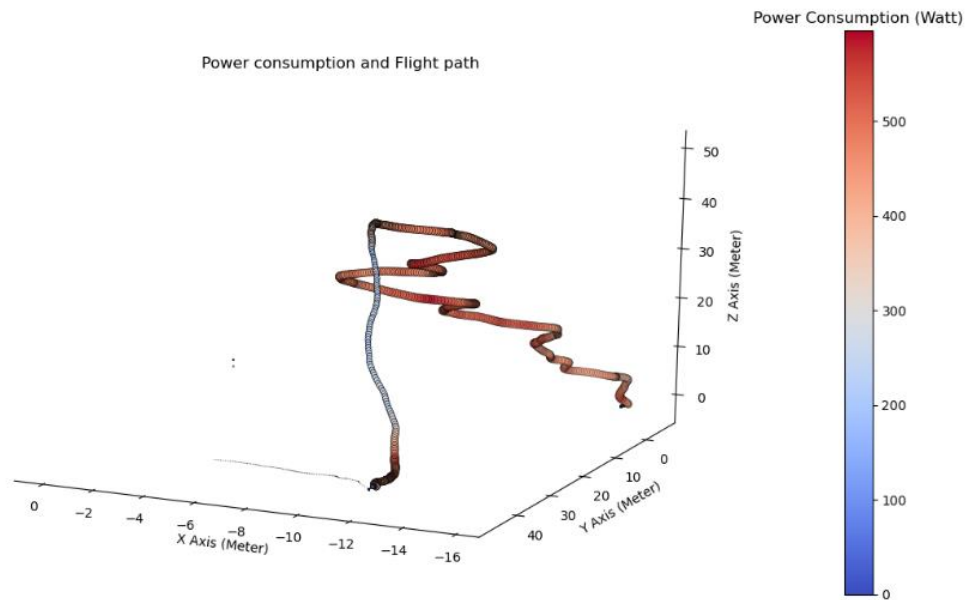


Figure 8-6: Power consumption and flight path

## CHAPTER 9: RESULTS AND FINDINGS

**9.1 Result Discussion:** The simulation and experimental results for five distinct environments are shown in the following tables. Each area has a unique collection of obstacles, as well as a unique starting route that must be navigated.

The proposed method finds the best route four times for each environment, once for each environment.

The ideal route was determined by evaluating it against the objective function that was constructed. After that, the quadrotor was flown along the optimal route using the experimental methods described in chapter 5.

If we look into the first test data in [Table 9.1](#), here for total 4 flights, flight 3 gives the minimum cost which is 95.03. [Figure 9-1](#) shows the optimized path given for flight 3, the best in this test scenario. Here theoretical and experimental path length, time and energy consumptions were the minimum amongst the other 3 flights. Which validate the theoretical and experimental similarities.

Similarly [Table 9.2](#), [Table 9.3](#), [Table 9.4](#), [Table 9.5](#) shows the data from the test no 02, test no 03, test no 04 and test no 05 accordingly. [Figure 9-2](#) shows the best flight path, which is flight 4 in test no 02. [Figure 9-3](#) shows the path of test no 03, which is flight 2. [Figure 9-4](#) shows flight 4 of test no 04 and [Figure 9-5](#) is the flight 4 of test no 05. Here only the path that got minimum cost after optimization is shown.

It is clear from the following tables that there is a strong correlation between the least cost path and the experimental outcomes, in both the simulation and the experimental scenarios.

From the following figures optimized paths are visible. Here it is clear that optimized path gives best route considering length, time and energy consumption avoiding collision with the obstacles. Blue line in the figures shows initially randomly generated path, orange line is the optimized path.

Test No.1	Theoretical			Experimental			Total Cost
	Path Length (m)	Minimum Time (s)	Energy Consumption (KJ)	Path Length (m)	Time (s)	Energy Consumption (KJ)	
Flight - 1	81.14	85.29	59.11	89.91	93.34	60.56	99.3
Flight - 2	82.48	86.7	60.09	89.39	92.8	60.21	98.72
Flight - 3	82.09	86.29	59.8	86.04	89.32	57.95	95.03
Flight - 4	79.4	83.46	57.84	90.36	93.81	60.87	99.8

Table 9.1: Theoretical and experimental data for test no. 01

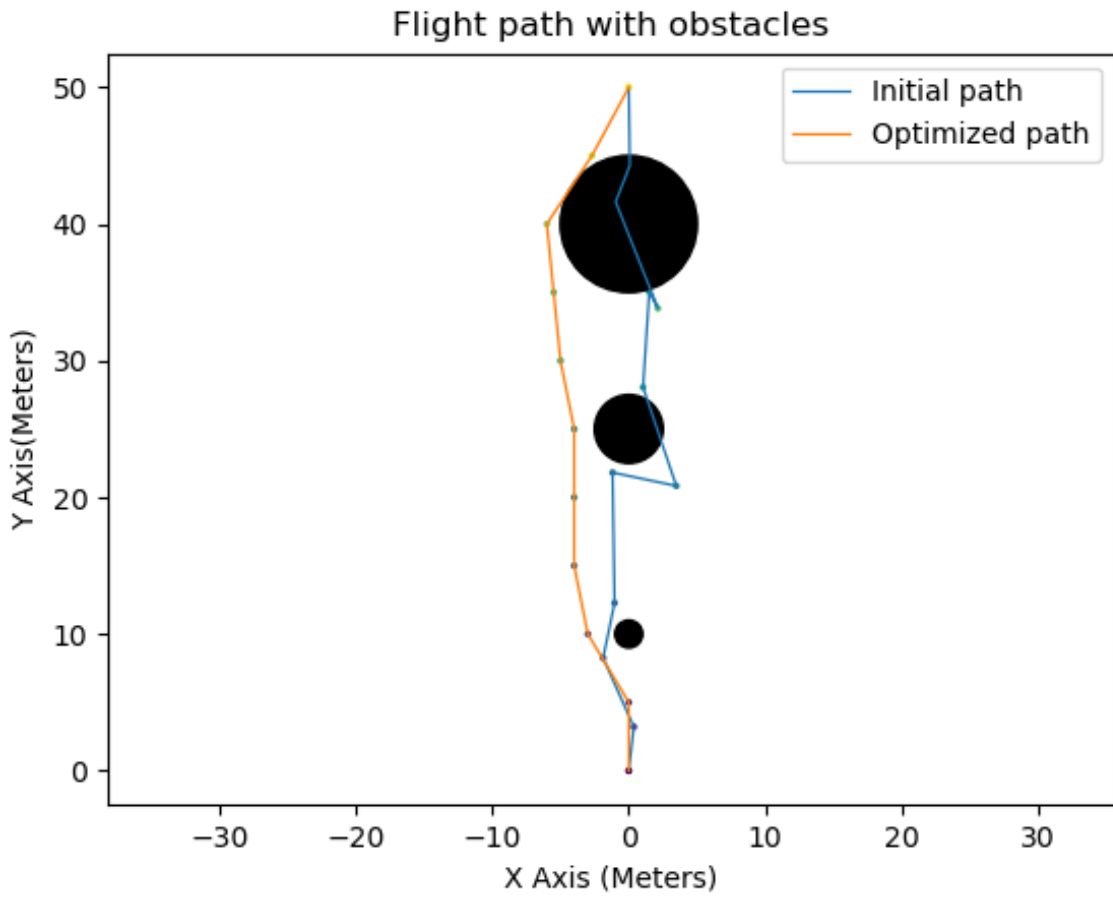


Figure 9-1: Flight path for test no. 01

Test No.2	Theoretical			Experimental			Total Cost
	Path Length (m)	Minimum Time (s)	Energy Consumption (KJ)	Path Length (m)	Time (s)	Energy Consumption (KJ)	
Flight -1	81.53	85.31	57.72	92.83	91.31	58.06	95.52
Flight -2	83.25	87.11	58.93	95.07	93.52	59.46	97.83
Flight -3	80.19	83.91	56.77	91.53	90.03	57.24	94.18
Flight -4	82.44	86.26	58.36	86.12	84.71	53.87	88.62

Table 9.2: Theoretical and experimental data for test no. 02

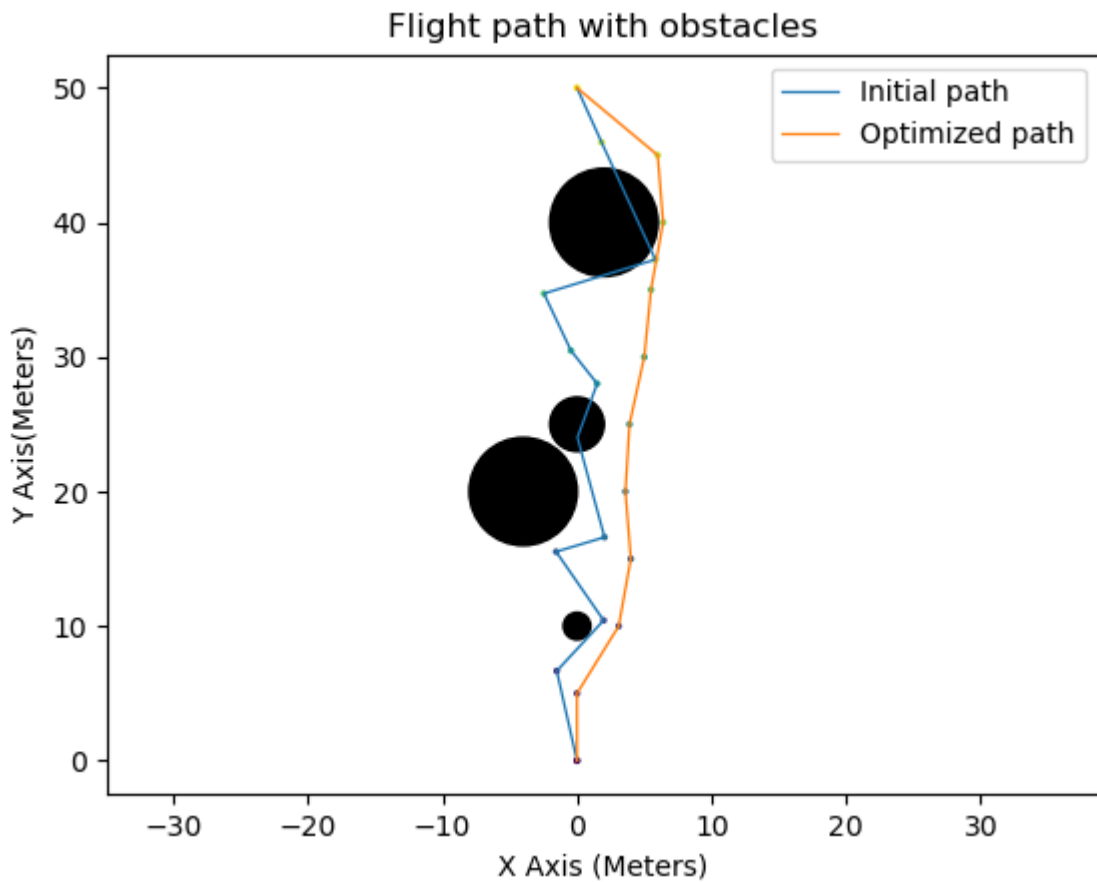


Figure 9-2: Flight Path for test no. 02

Test No.3	Theoretical				Experimental		Total Cost
	Path Length (m)	Minimum Time (s)	Energy Consumption (KJ)	Path Length (m)	Time (s)	Energy Consumption (KJ)	
Flight -1	82.54	86.78	56.88	89.09	98.05	61.58	92.1
Flight -2	82.97	87.24	57.18	88.75	97.68	61.35	91.75
Flight -3	79.71	83.81	54.93	92.54	101.85	63.96	95.67
Flight -4	81.6	85.79	56.23	87.87	90.84	60.73	96.7

Table 9.3: Theoretical and experimental data for test no. 03

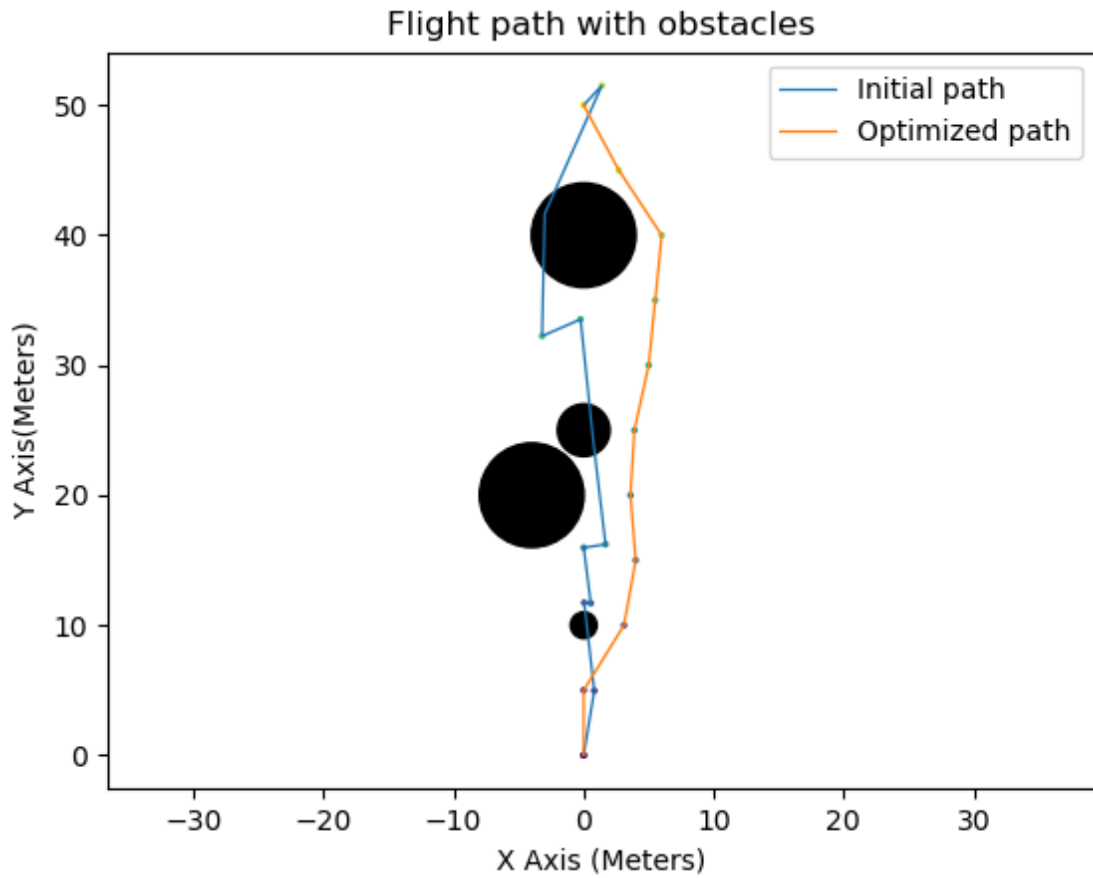


Figure 9-3: Flight Path for test no. 03

Test No.4	Theoretical			Experimental			Total Cost
	Path Length (m)	Minimum Time (s)	Energy Consumption (KJ)	Path Length (m)	Time (s)	Energy Consumption (KJ)	
Flight -1	81.56	85.31	58.98	84.74	93.53	64.05	98.52
Flight -2	82.21	85.99	59.45	83.68	92.36	63.25	97.28
Flight -3	84.34	88.22	60.99	82.75	91.33	62.55	96.21
Flight -4	79.64	83.3	57.59	81.66	90.13	61.72	94.94

Table 9.4: Theoretical and experimental data for test no. 04

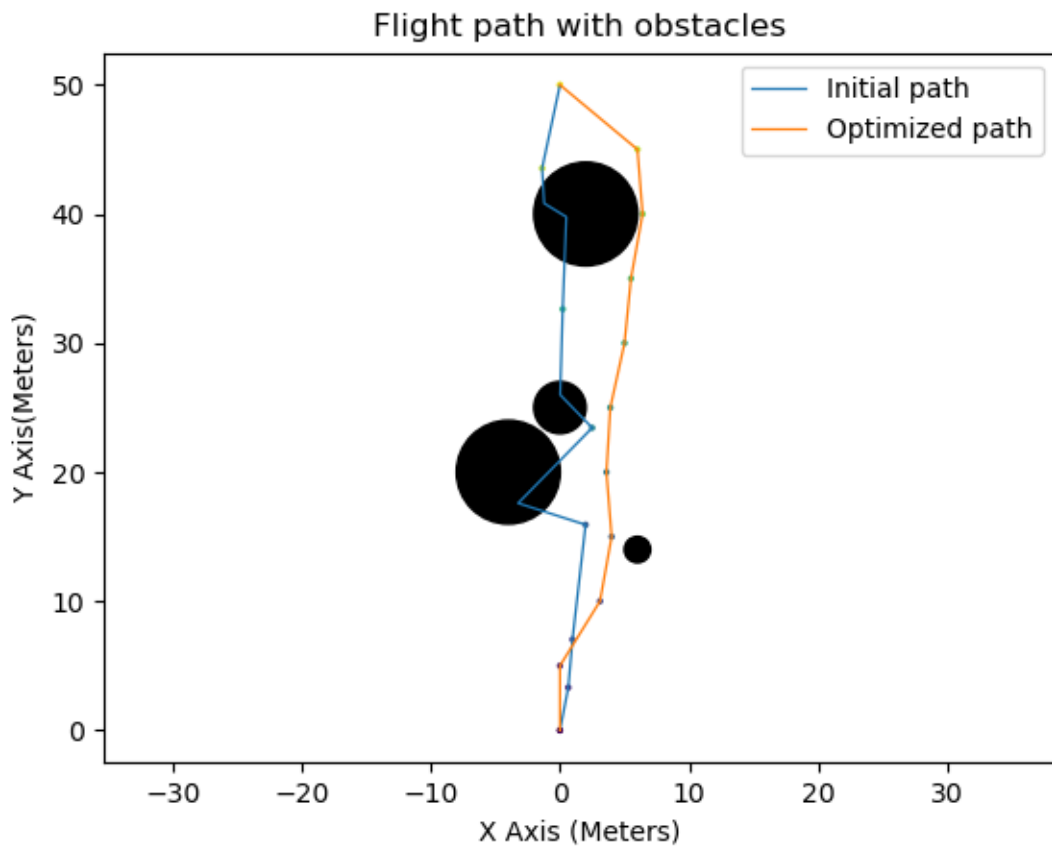


Figure 9-4: Flight Path for test no. 04



Test No.5	Theoretical			Experimental			Total Cost
	Path Length (m)	Minimum Time (s)	Energy Consumption (KJ)	Path Length (m)	Time (s)	Energy Consumption (KJ)	
Flight -1	86.5	92.06	60.09	100.65	103.71	65.69	103.94
Flight -2	84.67	90.11	58.82	99.01	102.02	64.62	102.25
Flight -3	85.39	90.88	59.32	104.34	107.51	68.1	107.75
Flight -4	84.76	90.21	58.88	99.	102.01	64.61	102.24

Table 9.5: Theoretical and experimental data for test no. 05

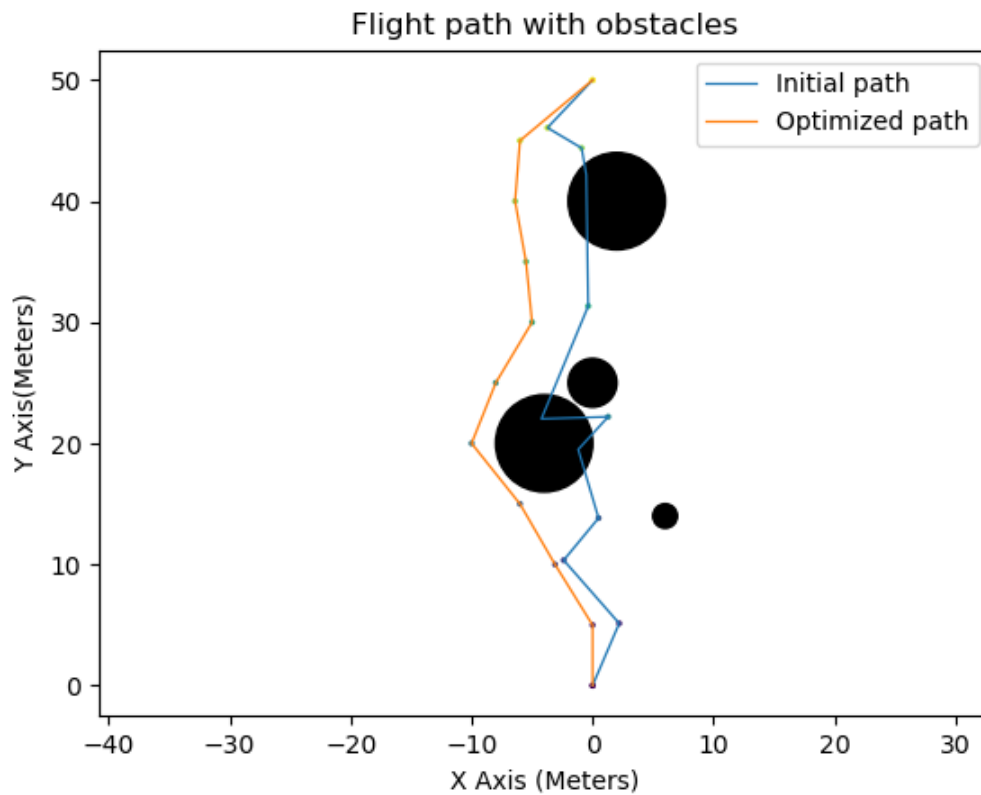


Figure 9-5: Flight Path for test no 05

## **9.2 Conclusion and Further Study**

In this paper, a multi-objective optimizer-based method has been proposed and successfully applied to solve the path planning problem of quadrotor UAVs in a 3D static environment. The path planning problem was formulated as a multi-objective optimization problem under operational constraints. The proposed planning approach aims to lead the drone to traverse a short and fast path in a static environment without collision while consuming the minimum amount of energy. The demonstrative results and experiments show that the proposed MMO-based method is a viable process to plan paths in complex 3D environments. The future scope of this work includes – Using RTK GPS to get better experimental results with less noise. Using an RF beacon to perform the experiment with a reduced effect of random variables increasing the fidelity of the cost functions to better approximate the dynamics of a quadrotor. Optimize the code to run faster and enable it to run in real time on a remote/companion computer in conjunction with the flight controller.

## References

- [1] *What is the difference between a drone and an RC plane or helicopter?* Drones Etc.
- [2] P. Drones, “The Differences Between UAV, UAS, and Autonomous Drones,” *Percepto*, Jan. 2019, Accessed: May 10, 2022. [Online]. Available: <https://percepto.co/what-are-the-differences-between-uav-uas-and-autonomous-drones/>.
- [3] *State government gears up for autonomous RPAS mapping*. 2017.
- [4] “Automated Drone (UAV) Applications | Airobotics.” [https://www.airoboticsdrones.com/applications/?fbclid=IwAR3js5BgRyVHUp\\_YM\\_9Cc-OVdPAu1JEvkCazDrtjHTxAjRS4TlceLrfbEgA](https://www.airoboticsdrones.com/applications/?fbclid=IwAR3js5BgRyVHUp_YM_9Cc-OVdPAu1JEvkCazDrtjHTxAjRS4TlceLrfbEgA) (accessed May 10, 2022).
- [5] B. Vergouw, H. Nagel, G. Bondt, and B. Custers, “Drone Technology: Types, Payloads, Applications, Frequency Spectrum Issues and Future Developments,” *Futur. Drone Use Oppor. Threat. from Ethical Leg. Perspect.*, pp. 21–45, 2016, doi: 10.1007/978-94-6265-132-6\_2.
- [6] V. K. Saxena, “The Amazing Growth and Journey of UAV’s and Ballistic Missile Defence Capabilities: Where the Technology is Leading to?,” p. 6, 2013, Accessed: May 10, 2022. [Online]. Available: <https://books.google.com/books?id=hwWqCQAAQBAJ&pg=PA6>.
- [7] “Applications and Uses for Multirotor Drones – Rise Above.” <https://riseabove.com.au/pages/uav-applications-and-uses?fbclid=IwAR3nnGS4YuIJZDNwC3Czbkuw5W52gKXVIelDgBgo4cenMOSdBeNKxi1Dj-M> (accessed May 10, 2022).
- [8] “11. Optimization Algorithms — Dive into Deep Learning 0.17.5 documentation.” [https://d2l.ai/chapter\\_optimization/?fbclid=IwAR2jM2LqOMf8z-WpD7Xdbqr6Dmjqr1XOkBaz1aE-yY-\\_Vi92YkCb-MaPZA](https://d2l.ai/chapter_optimization/?fbclid=IwAR2jM2LqOMf8z-WpD7Xdbqr6Dmjqr1XOkBaz1aE-yY-_Vi92YkCb-MaPZA) (accessed May 10, 2022).

- [9] R. Jarray, M. Al-Dhaifallah, H. Rezk, and S. Bouallegue, "Path Planning of Quadrotors in a Dynamic Environment Using a Multicriteria Multi-Verse Optimizer," *C. Mater. Contin.*, vol. 69, no. 2, pp. 2159–2180, 2021.
- [10] C. Ramirez-Atencia and D. Camacho, "Constrained multi-objective optimization for multi-UAV planning," *J. Ambient Intell. Humaniz. Comput.*, vol. 10, no. 6, pp. 2467–2484, 2019.
- [11] S. Hayat, E. Yanmaz, T. X. Brown, and C. Bettstetter, "Multi-objective UAV path planning for search and rescue," in *2017 IEEE international conference on robotics and automation (ICRA)*, 2017, pp. 5569–5574.
- [12] L. Swartzentruber, J. L. Foo, and E. Winer, "Three-dimensional multi-objective UAV path planner using terrain information," in *50th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference 17th AIAA/ASME/AHS Adaptive Structures Conference 11th AIAA No.*, 2009, p. 2222.
- [13] S. Mittal and K. Deb, "Three-dimensional offline path planning for UAVs using multiobjective evolutionary algorithms," in *2007 IEEE Congress on Evolutionary Computation*, 2007, pp. 3195–3202.
- [14] D. Rathbun, S. Kragelund, A. Pongpunwattana, and B. Capozzi, "An evolution based path planning algorithm for autonomous motion of a UAV through uncertain environments," in *Proceedings. The 21st Digital Avionics Systems Conference*, 2002, vol. 2, pp. 8D2-8D2.
- [15] S. Aggarwal and N. Kumar, "Path planning techniques for unmanned aerial vehicles: A review, solutions, and challenges," *Comput. Commun.*, vol. 149, pp. 270–299, 2020.
- [16] R. D'Andrea, "Guest editorial can drones deliver?," *IEEE Trans. Autom. Sci. Eng.*, vol. 11, no. 3, pp. 647–648, 2014.
- [17] K. Dorling, J. Heinrichs, G. G. Messier, and S. Magierowski, "Vehicle routing problems for drone delivery," *IEEE Trans. Syst. Man, Cybern. Syst.*, vol. 47, no. 1,

pp. 70–85, 2016.

- [18] H. Y. Jeong, B. D. Song, and S. Lee, “Truck-drone hybrid delivery routing: Payload-energy dependency and No-Fly zones,” *Int. J. Prod. Econ.*, vol. 214, pp. 220–233, 2019.
- [19] C.-M. Tseng, C.-K. Chau, K. M. Elbassioni, and M. Khonji, “Flight tour planning with recharging optimization for battery-operated autonomous drones,” *CoRR*, *abs/1703.10049*, 2017.
- [20] J. Zhang, J. F. Campbell, D. C. Sweeney II, and A. C. Hupman, “Energy consumption models for delivery drones: A comparison and assessment,” *Transp. Res. Part D Transp. Environ.*, vol. 90, p. 102668, 2021.
- [21] E. Kim, J. Lee, and K. G. Shin, “Real-time prediction of battery power requirements for electric vehicles,” in *2013 ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS)*, 2013, pp. 11–20.
- [22] A. Capiello, I. Chabini, E. K. Nam, A. Lue, and M. Abou Zeid, “A statistical model of vehicle emissions and fuel consumption,” in *Proceedings. The IEEE 5th international conference on intelligent transportation systems*, 2002, pp. 801–809.
- [23] K. I. M. McKinnon, “Convergence of the Nelder–Mead simplex method to a non-stationary point,” *SIAM J. Optim.*, vol. 9, no. 1, pp. 148–158, 1999, doi: 10.1137/S1052623496303482.
- [24] M. A. Luersen and R. Le Riche, “Globalized Nelder–Mead method for engineering optimization,” *Comput. Struct.*, vol. 82, no. 23–26, pp. 2251–2260, 2004.
- [25] R. Chelouah and P. Siarry, “Genetic and Nelder–Mead algorithms hybridized for a more accurate global optimization of continuous multim minima functions,” *Eur. J. Oper. Res.*, vol. 148, no. 2, pp. 335–348, 2003.
- [26] S. Berezina, O. Solonets, and M. Bortsova, “Referencing of UAV Images Using the Nelder-Mead Method,” in *2018 IEEE 17th International Conference on Mathematical Methods in Electromagnetic Theory (MMET)*, 2018, pp. 103–106.

- [27] R. C. Eberhart, Y. Shi, and J. Kennedy, *Swarm intelligence*. Elsevier, 2001.
- [28] D. Karaboga and B. Akay, “A survey: algorithms simulating bee swarm intelligence,” *Artif. Intell. Rev.*, vol. 31, no. 1, pp. 61–85, 2009.
- [29] A. Akhtar, “Evolution of Ant Colony Optimization Algorithm--A Brief Literature Review,” *arXiv Prepr. arXiv1908.08007*, 2019.
- [30] K. Socha and M. Dorigo, “Ant colony optimization for continuous domains,” *Eur. J. Oper. Res.*, vol. 185, no. 3, pp. 1155–1173, 2008.
- [31] “What Are Autonomous Drones? How Do They Work? - Remoteflyer.”  
<https://www.remoteflyer.com/what-are-autonomous-drones-how-do-they-work/>  
(accessed May 17, 2022).
- [32] “Autonomous Drone Using RPi : 9 Steps - Instructables.”  
<https://www.instructables.com/Autonomous-Drone-Using-RPi/> (accessed May 17, 2022).
- [33] “Copter Home — Copter documentation.” <https://ardupilot.org/copter/> (accessed May 17, 2022).
- [34] “First Flight with Copter — Copter documentation.”  
<https://ardupilot.org/copter/docs/flying-arducopter.html> (accessed May 17, 2022).
- [35] H. Gossett, “Building An Autonomous Indoor Drone System,” 2018.
- [36] “DroneKit.” <https://dronekit.io/> (accessed May 17, 2022).
- [37] “What is Cost Function in Machine Learning [Updated] | Simplilearn.”  
<https://www.simplilearn.com/tutorials/machine-learning-tutorial/cost-function-in-machine-learning> (accessed May 10, 2022).
- [38] D. Brescianini, M. Hehn, and R. D’Andrea, “Nonlinear quadrocopter attitude control: Technical report,” ETH Zurich, 2013.
- [39] “Pixhawk Overview — Copter documentation.”  
<https://ardupilot.org/copter/docs/common-pixhawk-overview.html> (accessed May 17, 2022).

- [40] J. Lesko, M. Schreiner, D. Megyesi, and L. Kovacs, “Pixhawk PX-4 Autopilot in Control of a Small Unmanned Airplane,” *2019 Mod. Saf. Technol. Transp.*, pp. 90–93, Nov. 2019, doi: 10.1109/MOSATT48908.2019.8944101.
- [41] “How to Choose an Optimization Algorithm.”  
[https://machinelearningmastery.com/tour-of-optimization-algorithms/?fbclid=IwAR0EA6\\_TcpbDuJrNXYwNbXiSBiZ\\_n4UAveVvJ5Q2LV M0OcC-sJf7Y7pZGUY](https://machinelearningmastery.com/tour-of-optimization-algorithms/?fbclid=IwAR0EA6_TcpbDuJrNXYwNbXiSBiZ_n4UAveVvJ5Q2LV M0OcC-sJf7Y7pZGUY) (accessed May 10, 2022).

## **APPENDIX-A: Pseudo Code**

Step 1: SET obstacles array

Step 2: SET step size

Step 3: SET start location

Step 4: SET destination location

Step 5: Generate Initial path in NED frame

Step 6: Optimize path

Step 7: Optimize path using objective function and Nelder Mead algorithm

Step 8: Convert path from NED frame to earth frame

Step 9: Generate mission file from path in global frame

Step 10: Save mission file

Step 11: Conduct test flight

Step 12: Download flight log

Step 13: Extract length, time, energy consumption from flight log

Step 14: Compare with saved path using objective function