A Project Report on

Design, simulation and demonstration of swarm flying formation of drones

Sponsored By

DRDO

Domain

Robotics & IOT

Submitted to

Savitribai Phule Pune University
In Partial Fulfillment of the requirement for the Award of

BACHELOR’S DEGREE IN
ELECTRONICS NAD TELECOMMUNICATION ENGINEERING

Submitted By
AMEY KORE (GR. No. U1610325)
KUNAL GANDHI (GR. No. U1610239)
RUSHIKESH GAMARE (GR. No. U1510132)
Under the guidance of
Prof. Vishal B. Ambhore

Department of Electronics and Telecommunication Engineering

VISHWAKARMA INSTITUTE OF INFORMATION TECHNOLOGY, PUNE

(An Autonomous Institute Affiliated to Savitribai Phule Pune University)

2019-20

Acknowledgement

This work could not have been completed without the guidance and encouragement of many people. We would like to particularly acknowledge those below.

We pay our humble regards and gratitude to **Jaywant Kolhe (Scientist D)** and **Dhananjay Rajmane (Scientist B)** from **DRDO** for guiding us and giving moral support and timely boost.

We wish to express our special thanks to **Prof. Vishal Ambhore (Project Guide)**, and project evaluators who helped us a lot in the preparation of our seminar topic.

I would like to express my sincere thanks to my beloved family who has been the pillars of strength in all my endeavors.

AMEY KORE (GR. No. U1610325)

KUNAL GANDHI (GR. No. U1610239)

RUSHIKESH GAMARE (GR. No. U1510132)

Bansilal Ramnath Agarwal Charitable Trust's

Vishwakarma Institute of Information Technology

(An Autonomous Institute Affiliated to Savitribai Phule Pune University)

An ISO 9001-2008 Certified Institute Accredited with 'A' Grade By NAAC

Certificate

This is to certify that project work is entitled "Swarming of Drones" carried out in the eighth semester by,

AMEY KORE (GR. No. U1610325)
KUNAL GANDHI (GR. No. U1610239)
RUSHIKESH GAMARE (GR. No. U1510132)

In the partial fulfillment for the award of Bachelor of Engineering degree in Electronics and Telecommunication Engineering from Vishwakarma Institute of Information and Technology, Savitribai Phule Pune University, Pune during the academic year 2019-20.

Prof. Vishal B. Ambhore
Guide
VIIT, Pune

Mrs. Pallavi Deshpande
Project Coordinator
VIIT, Pune

Prof. (Dr.) S. V. Kulkarni
Head, Department of E&Tc Engineering
VIIT, Pune

Prof. (Dr.) Vivek S. Deshpande
Director
VIIT, Pune

Date: MAY 2020

Place: Pune

Bansilal Ramnath Agarwal Charitable Trust's

Vishwakarma Institute of Information Technology

(An Autonomous Institute Affiliated to Savitribai Phule Pune University)

An ISO 9001-2008 Certified Institute Accredited with 'A' Grade By NAAC

Certificate to the Guide

Certified that the work incorporated in the report, **"Swarming of Drones"** submitted by,

AMEY KORE (GR. No. U1610325)

KUNAL GANDHI (GR. No. U1610239)

RUSHIKESH GAMARE (GR. No. U1510132)

Work carried out by the candidate for the Final Year project under my supervision and guidance.

Prof. Vishal B. Ambhore

Guide

VIIT, Pune

Date: MAY 2020

Place: Pune

Abstract

In recent years, several research groups are working on new procedures and technologies to operate and monitor complex scenarios. Two specific areas include search, rescue and environmental monitoring. Both these topics require solution to critical issues related to the mission requirements and the mission profile. The choice of a specific aerial platform for the monitoring of complex scenarios should be made by examining particular correspondence to the needs of the mission at the same time, and the multiplying effect of what is measurable by sensors positioned on the ground as fixed configuration. Advanced aerial platforms such as Unmanned Aerial Vehicles (UAVs), often called drones, are today the most frequent response to the needs of different missions. In particular, the specific category of mini-UAV is a perfect solution for the missions that are generally categorized as the "three Ds" (Dull, Dirty and Dangerous). Moreover, drones have recently received a strong technological acceleration, thanks to recent advances in miniaturization of battery, communication, processing and sensing technology. The remote/proximal sensing data obtained using mini-UAVs were validated in several environmental monitoring missions with complex scenarios as reported in previous research. These include fusion of optical data with synthetic aperture radar data to detect environmental hazards, use of thermal imagery to monitor landfills, surface waters contamination and to detect illegal dumping and to identify other illegal activities. In addition, remote sensing data can be strategically combined with other data layers in geographic information systems to monitor the vulnerability of cultural sites and anticipate environmental violations.

The use of a range of aerial platforms and advanced sensors to detect the illegal activities was validated in several real missions in Italy. These are the first known use of these methods in both the fields of environmental research and law enforcement/environmental forensics. They also represent an example of collaboration between law enforcement and university teams on developing enhanced environmental protection methods.

A swarm or fleet of Unmanned Aerial Vehicles (UAVs) is a set of aerial robots i.e., drones that work together to achieve a specific goal. Each drone in a swarm is propelled by a specific number of rotors and has the ability to vertically hover, take-off, and land (VTOL). The flight of the drones is controlled either manually, i.e. by remote control operations, or autonomously by using processors deployed on the drone. A common purpose for drones is in military, but their civilian applications are attracting increased attention in recent times. Indeed, low-cost drones and their swarms provide a promising platform for innovative research projects and future commercial applications that will help people in their work and everyday lives.

In the operational surveillance for successful identification and prosecution of environmental pollution culprits, it is required to integrate system-based data from several sources. The surveillance service must also include geospatially tagged forensic data analysis (information arising from navigation/positional systems). The detection, identification and localization of a target are key elements in all the above operations. Group of mini-UAVs equipped with self-localization and sensing capabilities offer new opportunities; indeed, group of mini-UAVs can explore cluttered outdoor environments, where access to conventional platforms is inefficient, limited, impossible, or dangerous. In brief, the main motivations for adopting the UAV technology in the survey process are the following: reduction of risk of human falling, reduction of safety costs for plant stoppage, improved data density and quality due to a better proximity, accessibility to locations where people or vehicles have no access, faster and cheaper

data acquisition due to the involvement of less workforce and equipment. The coordinated swarming drones could be also considered as a single array of sensors configured to the measure of a host of environmental parameters. In search and rescue tasks, for example, a more effective approach is to achieve a quick “survey” of the area to identify key locations as quickly as possible. This exclusion process enables organizers to rescan the key locations that provided some circumstantial evidence. In this context, the quality of the sensing has also a direct impact on the overall mission performance.

Therefore, an important aspect of the swarm coordination is the possibility to require a sufficient number of redundant samples of the target to reliably classify it as “detected” or “undetected”.

A cooperative approach that exploits drones sensing, minimizes the error in target recognition. In contrast, to use a unique drone implies costly structure and design, as well as vulnerability. Hence, a number of considerations support the use of coordinated swarming drones. An important requirement of the coordination strategy is to avoid centralized control approaches, leading to exponential increases in communication bandwidth and software complexity. Swarm intelligence methodologies can be investigated to solve problems cooperatively while maintaining scalability. The main inspiration for swarming drones comes from the observation of social animals, such as insects, winged animals, and fish, which exhibit a collective intelligence which appears to achieve complex goals through simple rules and local interactions. The main benefits of a swarm drones include: robustness (for the ability to cope with the loss of individuals); scalability (due to the ability to perform well with different group size); and flexibility (thanks to the capability to manage a broad spectrum of different environments and tasks). To this aim, each individual of the swarm: acts with a certain level of autonomy; performs only local sensing and communication; operates without centralized control or global knowledge, and cooperates to achieve a global task.

Introduction

Overview

Drone swarm technology- The ability of drones to autonomously make decisions based on shared information has the potential to revolutionize the dynamics of conflict. Drones can disperse over large areas to identify and eliminate hostile surface to air missiles and other air defenses. Swarms will have significant application in almost every area of national and homeland security. Drones within the Swarm may serve different roles based on their capability. Customizable drone swarm offers flexibility to commanders enabling them to add or remove drones as needed. This requires a common standard for inter-drone communication so that new drones can easily be added to this swarm. Similarly, the swarm must be able to adapt to the removal of drones either intentionally or through hostile action.

Motivation

Nowadays we have all types of robots performing task we can think of. Here we are going to deal with a group of robots. We are dealing with drones. The group of drones combined to achieve certain goal is called swarming of drones. The motivation comes from observing insects, animals and fish that exhibit a collective intelligence. Which appears to achieve complex goals and reduce the risk. It improves data density and quality. Accessibility to location where people or vehicles have no access, Faster data acquisition, search and rescue task can be more effectively approached using group of coordinating drones. The quality of the sensing has also a direct impact on the overall mission performance. This research presents the fusion of swarm behavior in multi robotic systems, specifically the unmanned aerial vehicle (UAV) operation. This study directed on using robot swarms because of its key feature of decentralized processing amongst its members. This characteristic leads to advantage of robot operations because an individual robot failure will not affect the group performance. The algorithm emulating the animal or insect swarm behaviors is presented in this paper and implemented into an artificial robotic agent (UAV) in computer simulations. The simulation results concluded that for an increasing number of UAV the aggregation accuracy increases with an accuracy of 90.62%. The experiment for foraging revealed that the number of UAV does not affect the accuracy of the swarm instead the iterations needed are greatly improved with an average of 160.53 iterations from 50 to 500 QUAV. For swarm tracking, the average accuracy is 89.23%. The accuracy of the swarm formation is 84.65%. These results clearly defined that the swarm system is accurate enough to perform the tasks and robust in any UAV number.

Problem definition and objectives

To make a system of drones which have ability to autonomously make decisions based on shared information instead of being individually directed by a human controller. The basic idea of drone swarm is that its machines are able to make decisions among themselves. Advanced aerial platforms such as unmanned aerial vehicles (UAV) are often called drones. Drones are today’s most frequent response to the needs of different missions. Drones have advanced communication, processing and sensing technology as well as the minimization of battery. A group of UAVs equipped with self-localization sensing capability is offering new opportunities. Indeed, groups of UAVs can explore outdoor environments where access to conventional platforms is inefficient, limited, impossible or dangerous.

Objectives

- Precision landing
- ROS Communication
- Scheduling of UAVs
- Video transferring from UAVs to the ground station.
- Object detection using YOLO/Avoidance
- Path planning

Literature survey

Considered to be one of the most resourceful and multipurpose innovation of the century, UAVs have managed to perforate numerous segments of the global economic sphere. Having uses in fields ranging from filmmaking to farming, UAVs have managed to capture a major share of the commercial, personal goods, and military service market.

From quick deliveries at rush hour to scanning an unreachable military base, UAVs are proving to be extremely beneficial in places where man cannot reach or is unable to perform in a timely and efficient manner. Increasing work efficiency and productivity, decreasing workload and production costs, improving accuracy, refining service and customer relations and resolving security issues on a vast scale are a few of the top uses UAVs offer industries around the world.

Military usage of UAVs or RPAS (Remotely Piloted Aerial Systems) has become the primary use in today's world. Used as target decoys, for combat missions, research and development, and for supervision, UAVs have been part and parcel of military forces worldwide.

UAVs are used in situations where manned flight is considered too risky or difficult. They provide troops with a 24-hour "eye in the sky", seven days a week. Each aircraft can stay aloft for up to 17 hours at a time, loitering over an area and sending back real-time imagery of activities on the ground. The published works in the field can be distinguished into three categories: using a physical substance as a pheromone, which is necessarily transmitted in an indirect way between robots, by means of the physical environment; using a digital pheromone, transmitted via direct communication between robots; using a digital pheromone, transmitted via an indirect communication between robots. The latter is the category of our approach.

Kuyucu et al. use a swarm of robots releasing physical substance as a repulsive pheromone, for environment exploration. In particular, robots act combining three basic behaviors, with decreasing priority: wall avoiding, pheromone coordination, and random walk. Actually, there are various approaches in the literature using physical pheromones, because they do not require a computational structure. Although real pheromones are not usable with aerial vehicles, they can be simulated. Thus, this type of research can be interesting to model new types of digital stigmergy.

An example of stigmergic coordination between drones using direct communication is presented by Dasgupta, where he focuses on automatic target recognition. Potential target is marked by drones, which also communicate the gossiped pheromone to nearby drones, with probability inversely proportional to the distance from the source. The proposed stigmergic schema employs also repulsive pheromone, as a negative feedback, when a predefined number of drones identify the same target. A disadvantage of such scheme is that the bandwidth required goes into an exponential explosion as the population grows. To avoid redundancy in target evaluation each UAV has to maintain in memory the state of each potential and confirmed target. In this way, the direct communication in the swarm should be strongly limited [28].

A swarm coordination schema with indirect coordination is proposed by Sauter et al.[18]. Here the coordination of a swarm of vehicles is based on digital pheromones maintained in an artificial space called pheromone map and composed by an arbitrary graph of place agents, that is, intermediate control nodes. There are two classes of agents which deposit, withdraw, and read pheromones, that is, walkers and avatars. A walker agent aims to make movements and action decisions, whereas avatars collect location information to make estimates when sensor information is not available. The schema has been applied to a range of scenarios, among which target acquisition. An important problem of this approach is that the exploration depends on the initial position of the swarm. This model does not consider complex targets but only simple targets without structure. To handle the unreliability in sensing, a certain number of drones must be attracted on a potential target. To achieve this goal a spatial organization of the available drones is required in order to sense the pheromone deposit released during a survey leaded by a peer of the same group. This result can be achieved keeping flocking formation. Flocking behavior is exhibited during the group flight. It is an emergent effect caused by the observance of three rules: preserving heading alignment with flock-mates, while maintaining separation with respect to the nearest one and cohesion with the entire group, as described by Reynolds [24]. This flocking behavior formalization have been extensively used in swarming robots and drones coordination. Bouraq Di et al. [29] accomplish an unknown environment survey via a group of robots which has to stay close enough to maintaining the ability to communicate with each other. This objective is reached using Reynolds rules to organize the robot's distribution and movements. Hauert et al. [30] apply flocking rules for the management of a drone's swarm in order to keep an ad-hoc network during their flight and to coordinate their task. However, this application is based on the assumption of well-known search field, and then it is not applicable to unstructured environments, which is one of our requirements.

Project scope

In military swarming drones can be used in three ways to attack, defense and provide support functions: switches intelligence, surveillance and reconnaissance. Swarming is advantage for offensive missions because it can overwhelm enemy defenses with large number of potential targets.

In a swarming attack the drones are dispersed, which makes it difficult and expensive for the adversary to defend itself. If 10 drones attack the target simultaneously and 7 are shot down 3 will still be able to complete their mission. Because individual drones in Swarm do not need to survive at high rates, they may be dramatically cheaper than stand-alone weapon systems which are often extremely sophisticated and expensive. It is possible that even a large Swarm of several dozen unmanned aircraft may be both more effective and less expensive than a single manned aircraft. The Navy is currently doing research on using defensive swarm to halt attackers. Swarm of drones can also be used for defensive purposes by creating large numbers of decoys that cause confusion or actively disrupt attacking force.

Limitations

Advisories may actively try to undermine swarms effectiveness by following the drones, hacking their sensor or jamming their communication links the precise level of control and of swarm autonomy will vary by mission however military Doctrine requires human controller to ensure that a swarm follows proper rules of engagement and principal of humanitarian law to address the risk of unauthorized actions due to jamming or hacking that may affect command and control.

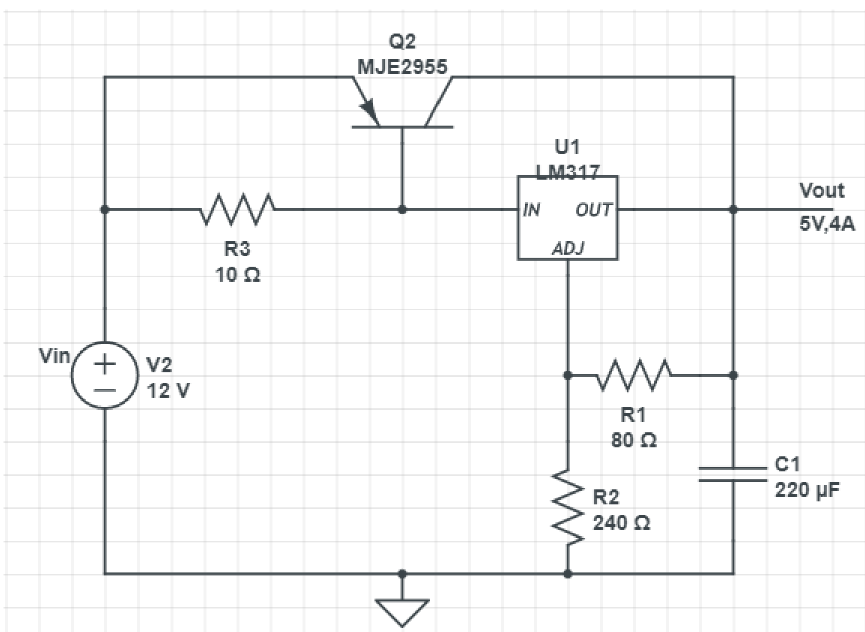
Power supply design

The drones have increasing demands due to good battery efficiencies. For any of the drone applications like surveillance, there are few peripheral devices need to be included like cameras, processors. These overhead peripherals need the power to work on the drone and also need to work in the least power without hampering the basic drone functioning.

The power supply required for the processor has the specification (5V,4A). To build the power supply of this specification either the linear or switching power supplies can be designed. Each of these types has its own advantages and disadvantages.

Here, we have used a linear power supply. LM317 is used as a voltage step down IC from 12V to 5V. The current provided by LM317 is limited to 2A. So, to get a total of 4A an external P-type MOSfet (MJE2955) is connected. Linear power supply is easy to design and implement compared to switch mode power supply. The processor powered by the power supply is used for continuous video transmission from drone camera to GCS.

Circuit diagram:



The prototype of the above circuit was designed on GPP board and used.

Calculations:

$$\boldsymbol{Vo = 1.25(1 + (R2/R1))}$$

R2 = 240 ohm
Vo = 5V

Therefore **R1** = 80 ohm

Selection of components:

Voltage step down IC:
1. 7805
2. LM317
LM317 was selected due to less current provided by 7805 (1.5A).

Current Booster IC:(P-type MOSfet)
1.IRF9540
2.MJE2955

Component list:

1.LM317
2.MJE2955
3.Capacitor (220 uF)
4.Resistors (10 Ω,80 Ω,240 Ω)
5.GPP board

System requirements

Software requirements

For this project we have used many different software's. To write the programs we have used Python language for that we select sublime text editors. We used the Ubuntu system so we can directly execute our scripts on the terminal instead of using any third-party compiler.

For simulation we have use Qgroundcontroller and mission planner software. These software's provide map and virtual drone simulation. We can connect our drones using elementary radio to these software's. Then it shows the virtual drones on map. These software's also provide Drone firmware. We compared many firmware and gave priority to stability of the drone. We select one ardupilot firmware which was very stable compared to other.

We also used one third party Android applications which is map coordinator. This app gives latitude and longitude of any position on the map. We use this application to take different coordinates on a map to create one mission.

The main job in our project is to share data of every drone to each other and use this data to take decisions. For this communication purpose we have used ROS. ROS is a robotic operating system which creates a platform for communication. Every drone is connected to a single network and ground stations are also connected over that network. ROS works on publishing and subscribing data. Every drone publishes its own data using the node created by ROS and the ground station subscribes to that node. Using the data, the ground station makes a decision and select drone to perform the task. Then the ground station publishes the coordinates over the network. And drones get the coordinates and perform tasks.

Hardware requirement

Drone is a balanced combination of hardware and software. The hardware used to design drones need to be robust. F450 is a Drone chassis we have used. This chassis gives flexibility. BLDC motor which has 1800KV rpm speed. Which can generate 1.2kg of thrust with 3S LiPo battery. This LiPo battery gives 12V and has a capacity of 3000 mAh. We use a power management board which takes input from the battery and gives required output to the motor and flight controller. Output on power management board is 5V and 12V. We used 20 ampere Hobbyking ESC. ESC converts dc voltage to 3 phase ac voltage which is required to drive BLDC motors. 433 MHz of telemetry radio to communicate with ground stations. Pixhawk flight controller which provides plug and play accessories. It contains a STM32F427 microcontroller. GPS to get geolocation. We used an Odroid microprocessor to install ROS on it. We use Taranis radio control remotely when it is in the manual mode. Landing gears are also attached to chassis for stable landing. We worked on a Pixymon camera which is used for precision landing. This piximon camera is connected to the drone and its leads are on the ground where the drone is going to land.

Ground station - at ground station we directly used laptop. Which has ROS installed in it. Every drone is connected to a ground station via telemetry radio. Every drone provides its own data as heartbit message. Ground station compare that data and publish the task information on network. The selected drone gets that information and perform task. Ground station must have following specifications

Windows 7 or above or ubuntu 14.04 and above, minimum 2GB of RAM, minimum 2 GB of available space, processor Intel core I3 or above, broadband internet connection. If these minimum requirements satisfy then only ROS can be installed on system. These minimum requirements can be satisfied by Odroid microprocessor. So, we have selected Odroid microprocessor to install ROS instead of selecting Raspberry Pi. As well as the processing speed of Odroid board is 10 times faster than Raspberry Pi. So, we are using Odroid board on the drone and laptop on ground station. Because ground station also has the USB ports to connect telemetry radio. There is limitation on USB ports. We have 4 USB ports then we can connect 4 drones to one ground station via telemetry radio.

Platform used for programming

There are various platforms used for development of programs. Some of the most popular platform developers use to program projects are Linux, Windows, macOS, Raspberry Pi, Arduino, etc. When it comes to actually deploying projects, developers and IT professionals are most likely to use Linux (81%) or Windows (72%), followed by Amazon Web Services (AWS) (37%), macOS (35%), and Docker containers (35%), the report found. [1]

While developing our project we have used multiple platforms, Windows, Ubuntu 18.04 LTS and Ubuntu 18.04 mate. Ubuntu 18.04 LTS is mostly used for program development and windows platform used for research. Hence basically any platform can be used instead of windows, but as per availability we used windows.

Why Ubuntu18.04 LTS?

1. Firstly, Linux based kernels provide a faster and cleaner experience which is good for development purposes. It will always receive feature and software updates as it is free source software so you can always remain up to date.
2. You don’t need to install drivers all the time on your PC (which is time consuming).
3. It is generally better protected from viruses than windows. A developer will find it good for him/her.
4. Secondly, the development atmosphere, tools and utilities that Linux provides can’t be matched with Windows. After all Windows is a consumer-oriented OS not development based whereas Linux is an all-rounder.
5. Linux Kernel can be readily modified as per ones need. For example, Google created Android by using the Linux kernel. They don’t need to create a new kernel. They borrowed the Linux kernel and started working on it. It saves their time and that’s why android’s new version releases very frequently. (less than 1 year)
6. Linux (some versions, not all) is bundled with many great applications and software that you can use and download free of cost that will make you pay if you use their counterparts on Windows.
7. Due to these facts, major agencies like NASA, ISRO, and even companies like Google, Yahoo are using Linux-based OS. That is why 99.998% of the world’s supercomputers are using Linux based OS.

We used Ubuntu 18.04 mate on the Odroid_XU4 board. Ubuntu Mate is one of the most famous Linux distributions that provides a graphical desktop environment. It is lightweight, fast, and customizable at your taste. [2]

Programing language used

We have used python language and BASH (Bourne Again Shell). BASH is the most widely used shell in Linux systems. It is used as default login shell in Linux systems and in macOS. It can also be installed on Windows OS. Bash is a command processor that typically runs in a text window where the user types commands that cause actions. Bash can also read and execute commands from a file, called a shell script. Like all Unix shells, it supports filename globing (wildcard matching), piping, here documents, command substitution, variables, and control structures for condition-testing and iteration. The keywords, syntax, dynamically scoped variables and other basic features of the language are all copied from sh. Other features, e.g., history, are copied from csh and ksh. Bash is a POSIX-compliant shell, but with a number of extensions.

Python has a clean and structured code base, making the updating and maintenance tasks of the software easier for developers. They don’t require developers to write additional code, and this in turn saves the effort and valuable time. Hence, they can utilize their time on something more productive for the organization. Large standard library Python has a robust and large standard library that makes it stand out from other programming languages. Its standard library contains a wide range of modules, operations and web service tools that you can select and use for your applications without writing additional code.

Formula Used:

Formula Used:1. Calculating Coordinate from distance [3]:

R: Radius of earth in Kilometers
D: Distance in meters
Lat1: Home latitude
Lon1: Home longitude
Lat2: Destination latitude
Lon2: Destination longitude
Theta: Bearing

Constant:
R = 6373.0 KM
Pi = 3.14

Calculation:

atan2(y, x) = 2arctan(y/((x^2 + y^2)^0.5 + x)
X = cos Lat1 * sin (Lon1)
Y = cos 0 * sin Lat1 – sin 0 * cos Lat1 * cos (Lon1)
Theta = atan2(X, Y) radians [4]
Lat1_radian = Lat1*(Pi/180)
Lon1_radian = Lon1*(Pi/180)
Theta_radian = Theta*(Pi/180)
north_displacement = distance*(sin(theta)/111111)
Lon2 = lon1_degree + north_displacement
east_displacement = distance*((cos(theta)/cos(lat1_radian))/111111)
Lat2 = lat1_degree + east_displacement

Endurance

In our drone system each drone has a 3S (3 Li-ion cells in series) battery with 3000 mAh,12 V. The maximum charging is 12.6 V above which it gets permanently damaged. We can't use the battery pack to its full capacity till it gets used up. The lower voltage till which battery can be used is around 10.8 V. So, each cell can charge maximum up to 4.2 V and discharge to 3.6V.

According to the calculation the battery voltage which is usable for drone operation is around 2V. In drone scheduling application which uses multiple drones to complete a certain task, each drone may have a different battery voltage at any given time. Some batteries may have high discharging rates, their internal constructions may vary, their count of charging and discharging cycles may vary. As the charging and discharging cycles for the battery increases its storage capacity decreases. To overcome the issues of battery difference, a term called BATTERY CONSTANT is used to find the maximum distance which can be covered by the drone in the available battery. The term battery constant is denoted by $\boldsymbol{k}$. Battery Constant for a drone is defined as the battery consumed by the drone to cover 1m distance. i.e. Battery Gradient.

Assumptions done for calculation of $\boldsymbol{k}$:

- Wind speed is not considered. If the region is windy the fly time of drones decreases by 2-3 mins. It needs to be considered when flying in windy area.
- Weight of Drone is kept constant. The change in weight increase the up thrust required to pick up its weight which in turn affects the battery capacity.
- Drone speed is kept constant. According to the application the speed is kept constant.

Now when the drones are used first, they are scheduled based on the shortest distance between their home and destination coordinates (Latitude, Longitude). The flying altitude for a particular mission is kept constant (5m). Here the region covered by each drone is fixed. This is a circular area around the home coordinates with a certain radius. If the destination falls under their area the drone is available for task completion. If the destination falls under two drone the both drones are available to complete the assigned task. Now when two drones can complete the task here comes the scheduling algorithm. The drone with highest efficiency is sent first.

Calculation of Battery Gradient ($\boldsymbol{k}$)

After completing the first mission the parameters like battery voltage(takeoff), battery voltage(land), total distance covered by drone, the flying altitude are required to calculate battery constant $\boldsymbol{k}$.

$$\boldsymbol{k} = \left(\frac{\boldsymbol{v_2 - v_1}}{\boldsymbol{D_t}}\right)$$

Where k is battery constant
V_2 = Battery after landing
V_1 = Battery During Takeoff
D_t = Total distance covered by the drone
D_t = (2h+2d)

Where h is the flying altitude
d = Distance between home and destination

This calculated battery constant value $\boldsymbol{k}$ is stored in a file. Now for the next flight the battery constant is used to calculate the maximum flying distance for the particular drone keeping the assumptions constant.

$$\boldsymbol{Max\ fly\ Distance} = \left(\frac{\boldsymbol{v_{takeoff} - v_{threshold}}}{\boldsymbol{k}}\right)$$

Where,

$V_{takeoff}$ = Battery voltage before takeoff from ground station.

$V_{threshold}$ = Minimum voltage under which battery gets discharged and require a charging.

Ideally $V_{threshold}$ = 10.8

The $\boldsymbol{k}$ value is read from the file saved after previous flight.

User Interface

Multiple drone scheduling requires the user to just enter the destination coordinates moreover it’s not always necessary for a user to sit and enter the coordinates it can be received from radar system. For this purpose, an independent desktop with any Windows or Ubuntu can be used.

- Hardware Interface

  An independent desktop with Windows or Ubuntu can be used.

- Software Interface

  Each system needs to be ROS compatible with a good network connection.

- Communication Interface

  All the GCS’s are connected to a single network. Each drone sends its own home coordinate, battery and many more parameters to the GCS. The MAVLink protocol is used for this purpose. Further for scheduling each GCS needs ROS over the system.

Nonfunctional Requirements

Performance Requirements

- Response Time

  The Response time of our system is wholly dependent on the Network available. It is also dependent on the system over which the scheduler is running. Further the response time increases if the ROS nodes don't function properly.

- Scalability

  This system can be easily scaled to at least 10 UAVs scheduling at a time, at the cost of increase in response time. Scalability also causes increase in computations and gives the final results a bit late.

- Platform

  The system runs on Windows or Ubuntu which have ROS installed over it. As the system is scaled the platform remains the same.

Safety Requirements

Security Requirements

One of the most important non-functional requirements is security.

Security requirements can come in many different forms:

- Privacy

Requirements can dictate protection for sensitive information. Some types of privacy requirements include: data encryption for database tables, policies regarding the transmission of data to 3rd parties (e.g., scrambling user account numbers), etc... Sources for privacy requirements could be legislative or corporate.

- Physical

physically the system needs to be protected from the errors like flying with excess weight, flying in windy regions and flying with altitude above 15m.

- Access

The access to the system is only to the organization which we are working for. This project wholly belongs to the same organization.

Software Quality Attributes

- Reliability

  The designed system is highly reliable and can sustain in any condition. This system also works in all areas having good network connections.

- Maintainability

  It's easy to add or remove any of the feature. It is easily upgradable. The system is easy to maintain and correct defects or make a change in the software.

- Usability

  The system is user friendly and can be learned to operate with just one trial. It is easy to operate ones fully understood.

- Correctness

  The system designed is as per the functional requirements. It is correct in terms of its functionality, calculations used internally and the navigation is correct.

- Efficiency

  The system is highly efficient if all the functional requirements are fulfilled. The system utilizes processor capacity, disk space and memory efficiently.

- Integrity or Security

  The system is secure from any unauthorized access, it prevents information loss and protect the privacy of data entered into the system.

- Testability

  The system is easy to test and find defects as it is divided in different modules.

- Flexibility

The designed system is flexible enough to work on any desktop or laptop with ROS and network connection.

- Reusability

The designed system can be reused by adding different system modules. The modules are also divided so that they can be reused across the application.

Project plan and implementation

To make this project first of all we need drones which are capable to generate sufficient thrust. First, we assembled drone considering thrust and weight. We assembled 3 quadcopter drones which contain Pixhawk4 microcontroller, BLDC motor, power management board, propellers, landing gear, GPS, telemetry radio. After assembly is done, we have three drones ready. Now we need to select suitable firmware. To select suitable firmware, we gave priority to stability of the drone. We checked stability of many firmware's. We got many firmware's which were stable but not suitable to our microcontroller. After comparing with many firmware, we selected ardupilot firmware. Which is more stable but it is not compatible with our software which is Qgroundcontroller so we converted that firmware in the form suitable form of Qgroundcontroller. Now we got stable firmware.

Now we can actually fly drones in mission mode with Qgroundcontroller. Mission mode in autonomous mode we need to only give coordinates to Drones. At this time, we face many problems. The biggest problem was GPS glitch. When drone is in air the GPS glitch may occur. When GPS glitch occur, the drone loses the map and cannot find any coordinates. And it can be dragged in any direction. The GPS glitch is depending upon sat-count (satellite count). Means drone is connected to number of satellites. The minimum sat-count required is 7. For more accuracy we need to connect with more satellite. Means we need more sat-count. To avoid this problem, we create a code. When GPS glitch is occurred, the drone should hover for some time. Waiting to recover the sat-count. If GPS glitch continues then it will land where it is. We try to find path with history of coordinates but when map is lost then it cannot recover coordinates.

Now we need to land our drone in accurate position. GPS has offset of 3 to 5m. For this we selected one IR lock sensor. it has two components camera and IR light emitter. The Pixymon camera can connect with pixhawk4 flight controller with i2c and IR light emitter is on the ground where drone is going to land. IR light always blink and Pixy camera detect that light when drone is in land mode. This IR-lock sensor has offset of 15 to 20 cm.

When hardware is ready. Now we need to establish communication between drones. for communication we select ROS which is robotic operating system this can be installed on Odroid microprocessor. ROS works on data publishing and subscription method. Every drone is connected to a single network / router and the ground station is also connected with same network. Post making suitable environment for ROS and installing all the packages of ROS. We can share data over network. Every drone publishes its own information called heartbit. It contains battery voltage, coordinates of home position, sat-count, etc. This data is published by every drone and ground station subscribe that data. Now new mission is taken by ground station where the scheduling code is running. The ground station compares every drone with new mission and select appropriate drone to complete that mission. After selecting a drone, the ground station publishes the drone Id to take a mission. The selected drone takes the mission and complete the mission.

MAVLink

There are two protocols that can be used to communicate with drone. Using Wi Fi and using MAVLink, we have used MAVLink over Wi Fi communication. MAVLink is ready to use protocol with less complexity and reliable. Wi Fi communication is slow and not directly usable. Hence, we have used MAVLink over Wi Fi.

Key Features of MAVLink:

1. Very efficient. MAVLink 1 has just 8 bytes overhead per packet, including start sign and packet drop detection. MAVLink 2 has just 14 bytes of overhead (but is a much more secure and extensible protocol). Because MAVLink doesn't require any additional framing it is very well suited for applications with very limited communication bandwidth.
2. Very reliable. MAVLink has been used since 2009 to communicate between many different vehicles, ground stations (and other nodes) over varied and challenging communication channels (high latency/noise). It provides methods for detecting packet drops, corruption, and for packet authentication.
3. Supports many programming languages, running on numerous microcontrollers/operating systems (Including ARM7, ATMega, dsPic, STM32 and Windows, Linux, MacOS, Android and iOS).
4. Allows up to 255 concurrent systems on the network (vehicles, ground stations, etc.)
5. Enables both offboard and onboard communications (e.g. between a GCS and drone, and between drone autopilot and MAVLink enabled drone camera).

The MAVLink (Micro Air Vehicle Link) protocol is a communication protocol widely used by small UAV’s developed under the LGPL (Lesser General Public License). MAVLink is a lightweight protocol, which is ideal for exchanging small amounts of a data between the control station and the UAV, avoiding high processing costs to handle these messages. The message structure consists of a set of mandatory fixed size fields (1 Byte each), i.e., the header, and an optional payload filed containing the message data to be transmitted. The message size varies from 8 to 263 Bytes.

Each MAVLink message is identified by the value in the identifier field in its packet header and has built-in content in the payload data field. The messages are logically defined by an XML document, which can represent the types of data being transmitted and the order in which the application should be interpret and process the received bytes. The protocol also allows the specification of a new message for communication between applications specific purpose. To define new message, it is necessary to represent it as a XML file, which is then automatically converted by code generation tool.

The Micro Air Vehicle Communication Protocol (MAVLink Protocol) is a point-to point communication protocol that allows two entities to exchange information. It is used for bidirectional communications between the drone and the GCS. MAVLink is a part of the DroneCode project, governed by the Linux Foundation. A MAVLink message is sent bytewise over the communication channel, followed by a checksum for error correction. If the checksum does not match, then it means that the message is corrupted and will be discarded. Figure.1 shows the structure of a MAVLink message. We will now give a brief description of the fields included in the message:

- STX: start-of-text indicates the beginning of a new messages.
- Length: indicates the length of the payload field.
- Sequence number: indicates the sequence number of the packet.
- System ID: ID of the sending system.
- Component ID: ID of the sending component.
- Message ID: ID of the message in the payload.
- Payload: payload of the packet, which contains the parameters of the message.
- CRC: checksum for validation. MAVLink messages are handled by the handleMessage (msg) function. This function has a switch statement, handling the different message IDs.

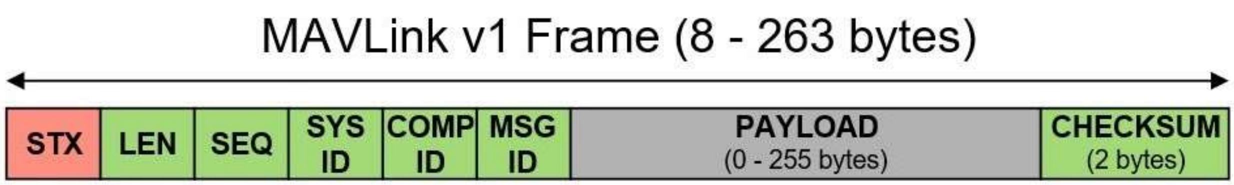


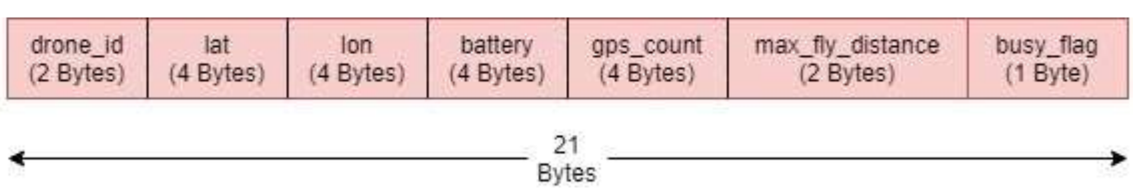
Figure.1

Message format for drone scheduling

For communication between drones and for scheduling we need some message format to establish. By defining message format, we can get essential information about drone. This is information is required for making decision which drone will able to complete the mission effectively.

Heartbit (Heartbeat):

Heartbit Message is continuously publishing over a network. This will tell us connectivity of drone.



Drone_ID:

Every drone on the field has its own unique characteristics. To identify drone with its unique characteristics we need to assign Drone_ID i.e. drone identification to each drone. This will help to identify which data belong to which drone. Also, not every mission belongs to all drones, sometimes only two drones of many times only one drone. So, to give command to specific drone using its Drone_ID is very easy work. So, each drone will transmit data along with its Id.

Home (lat, lon):

The home position of drone or known as the position where drone powered on, it is an important field in the message format. Drone always returns to the Home position after completing the mission and also if commanded as “Return to launch” (RTL). Drone is surveying in specific region drone has to have a station to return where it can charge itself or rest while there is mission to complete. By Transmitting Latitude and longitude we can determine distance to destination coordinate from position. This is important in various formulas such as Battery constant and backup distance.

Battery:

Battery status of drone is required to determine the battery level of drone. It is very important for drone to have sufficient battery to complete mission. If battery is not sufficient, script can decide which drone can complete the mission before assign task to battery inefficient drone. So it is very important to transmit the battery field. Battery can also use to decide ‘max back up distance’ and max fly distance. By getting information about battery we can decide whether battery is drained or not. If battery reading is below 10.5, battery is at minimal required charging immediately (LiPo battery of 3000 mAh and 3S).

gps_count:

As all mission given to drones are GPS based. Hence it is important for drone to transmit its GPS count so that if GPS count is not sufficient drone will not perform the mission. When the compass is calibrated, it then seeks the location of GPS satellites. When more than 6 are found, it allows the drone to fly in “Ready To Fly” Mode. Allows the UAV drone and remote-control system to know exactly its flight location. A home point can be set and this is the location the drone will return to, if the drone and the remote-control system stop connecting. This is also known as “fail-safe function”.

max_fly_distance:

Battery will only find current status of drone what if, while in between mission drone battery drains out and drone falls to the ground. Hence, we need field that can determine max distance drone can travel. If destination point is greater than max fly distance of drone then drone won’t go it will charge the battery until it will acquire sufficient battery to travel to that position.

busy flag:

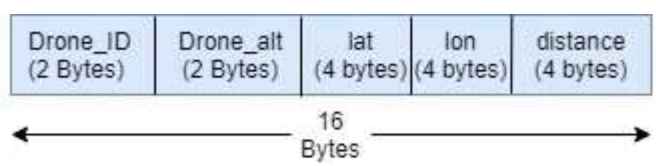
This field will notify whether the drone is performing any mission or not. While performing mission drone could not perform new mission. So instead of busy drone available drone in that region should perform that task. Hence multiple intrusion detection is possible here. We can say that multiple and independent mission are performed by drones.

Schedule message:

This message is transmitted by scheduler to drone to command them towards destination. Also, what altitude drone should acquire.

Drone_ID []:

This field is same as stated before in Hearbit but with slightest difference. Schedule Drone_ID field is tuple of Ids of drone. This is because Many drones can be available to perform task and if user want to command all drone to reach towards one destination point, it can be possible because of arrays of Drone_IDs.



Drone_alt[]:

This field is important if multiple drones are travelling to same point. As collision avoidance is not implemented yet and if more than two will collide to one another. So, if they could travel to different height the chance of collision is minimal. Hence this automatically assigns different height to different drone according to number of drones performing task.

Destination (lat, lon):

This is same field as home lat lon but instead of home, the destination coordinates are specified.

Distance:

Finally, nothing but the destination distance from UAV.

ROS (Robot Operating System)

What is ROS?

- ROS, the Robot Operating System, is an open source framework for getting robots to do things.
- The Robot Operating System (ROS) is a framework for writing robot software. It is a collection of tools, libraries, and conventions that aim to simplify the task of creating complex and robust robot behavior across a wide variety of robotic platforms.

Why should you learn/use ROS?

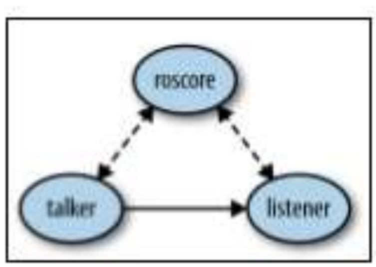
- The short answer is because it will save you time. ROS provides all the parts of a robot software system that you would otherwise have to write. It allows you to focus on the parts of the system that you care about, without worrying about the parts that you don't care about.

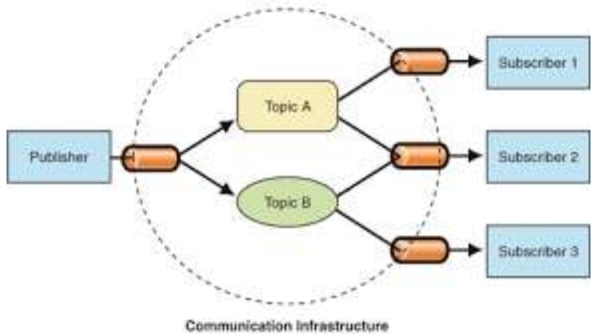
Roscore

- Roscore is a service that provides connection information to nodes so that they can transmit messages to one another. Every node connects to roscore at startup to register details of the message streams it publishes and the streams to which it wishes to subscribe.

Node in ROS

- "Node" is the ROS term for an executable that is connected to the ROS network. Here we'll create the publisher ("talker") node which will continually broadcast a message.



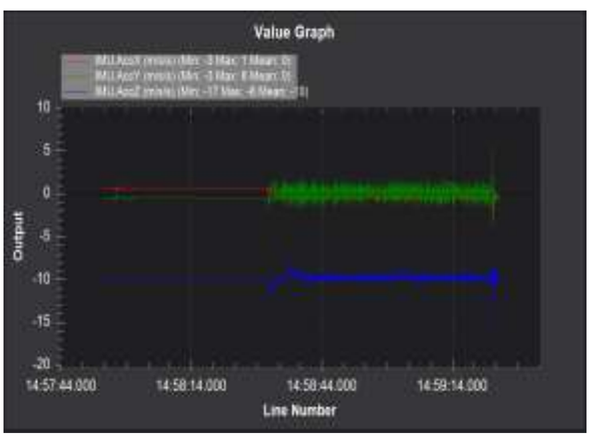


Messages
ROS data type used when subscribing or publishing messages.

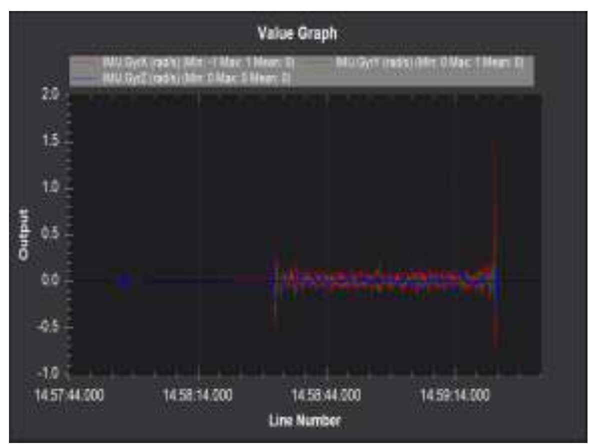
For using ROS over a machine, it is necessary to firstly setup the environment. All the installation process is in the reference link.

Further after successfully creating the environment we created a publisher node which publishes the message after every 0.1s. This rate of publishing message can be changed according to user requirements. Next the message contains the information which needs to be published with the data type of each parameter. At the subscriber node it subscribes to the publishing node and if all the data type and required parameters match the message is received over ROS network.

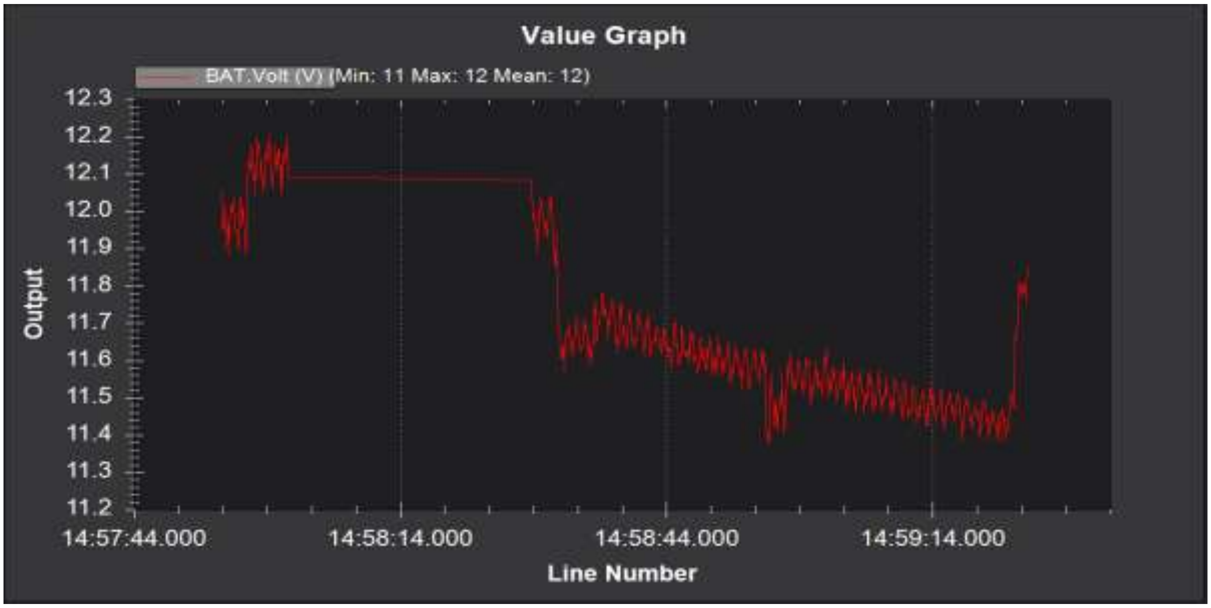
Results and analysis



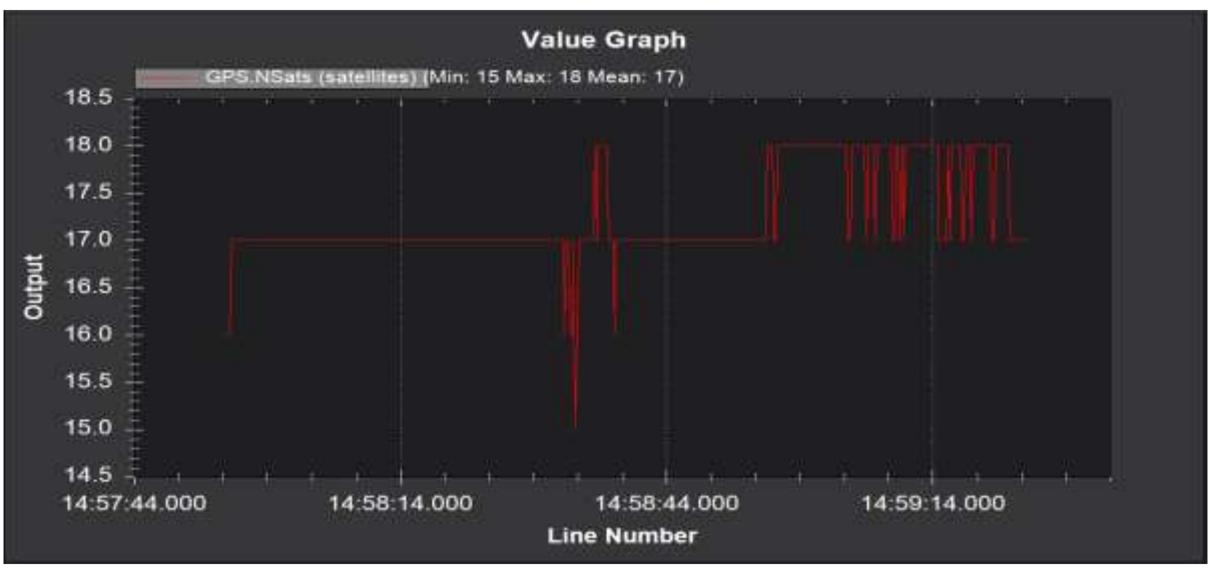
Accelerometer vs Time



Gyroscope vs Time



Voltage vs Time

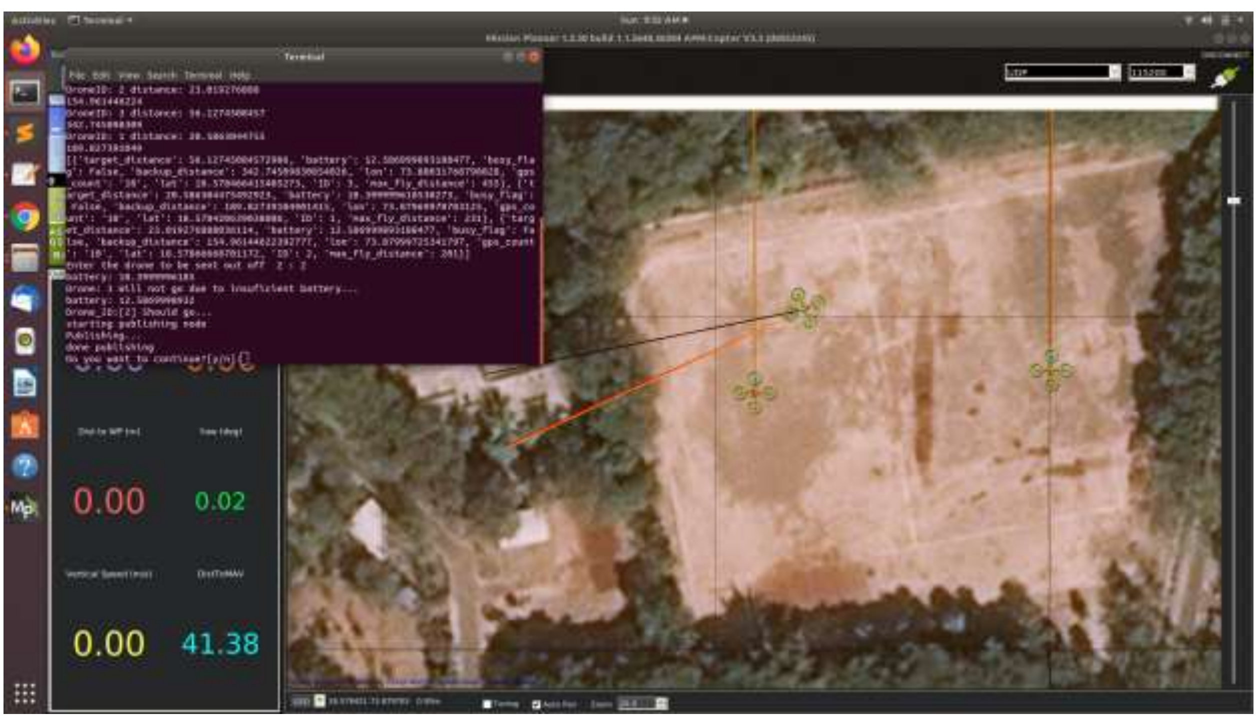


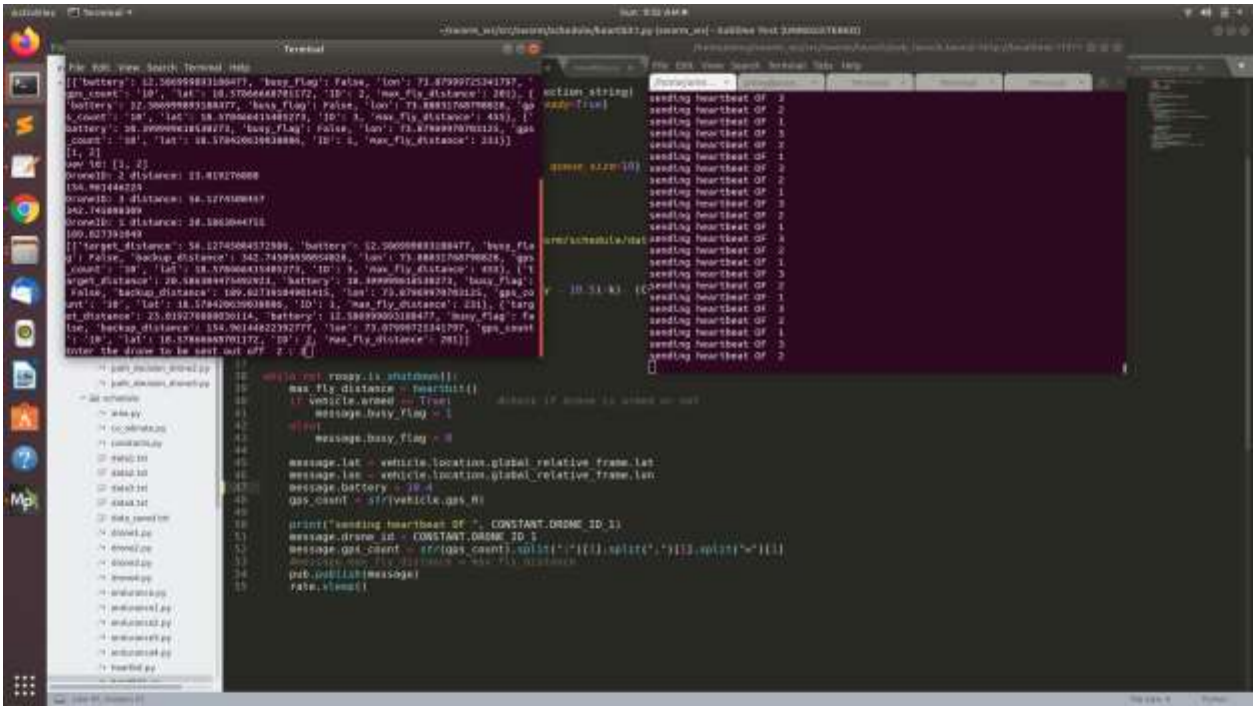
GPS vs Time

Test cases and results:

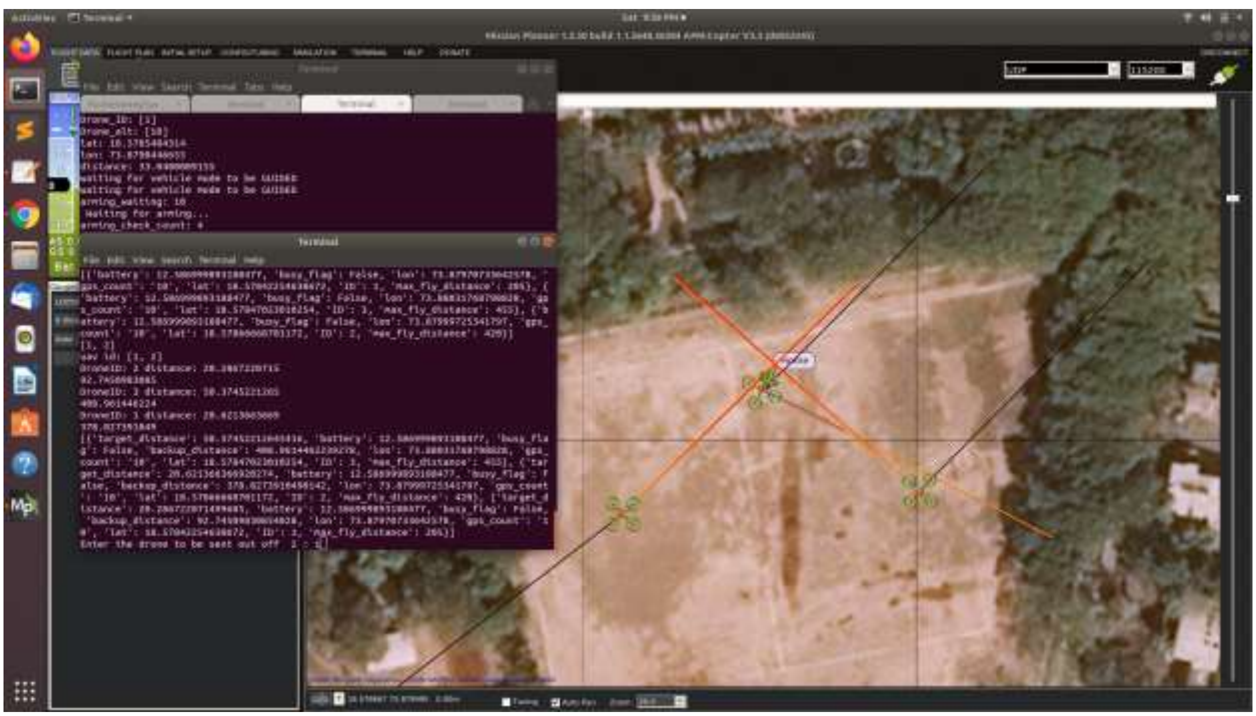
1. All the available drone has same Parameters

2. Destination in between Drone1 and Drone2 but Drone1 battery is low

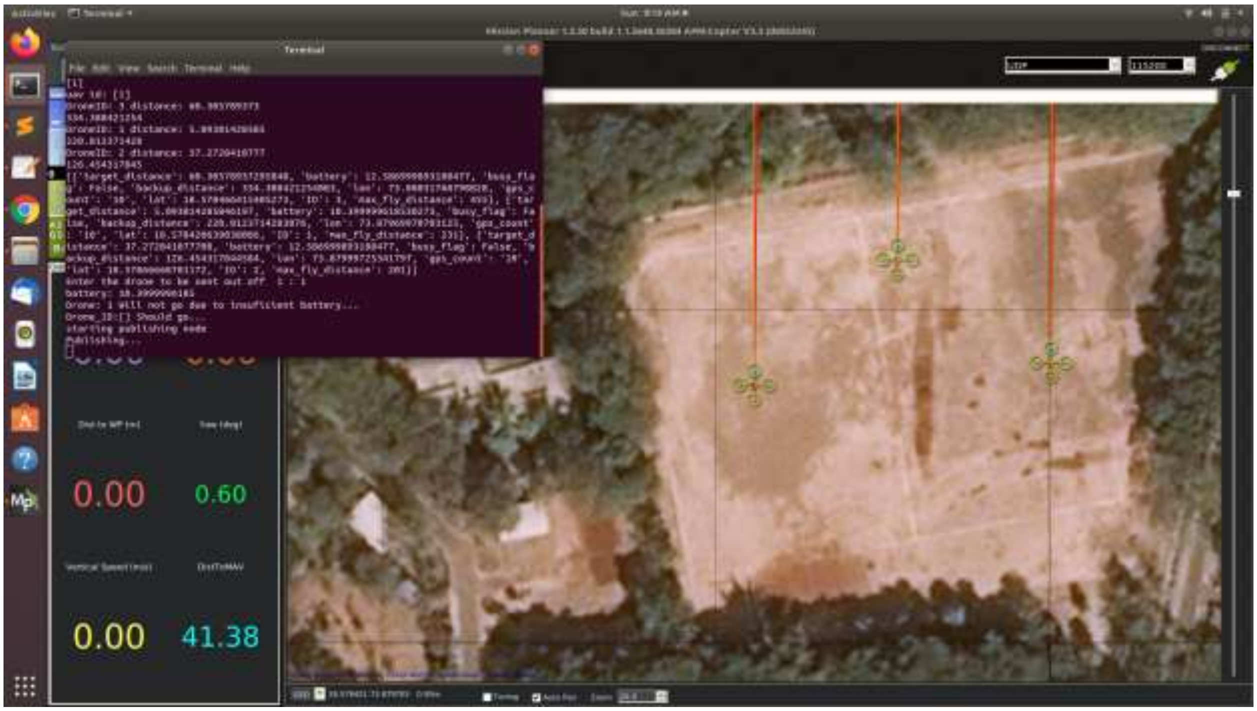




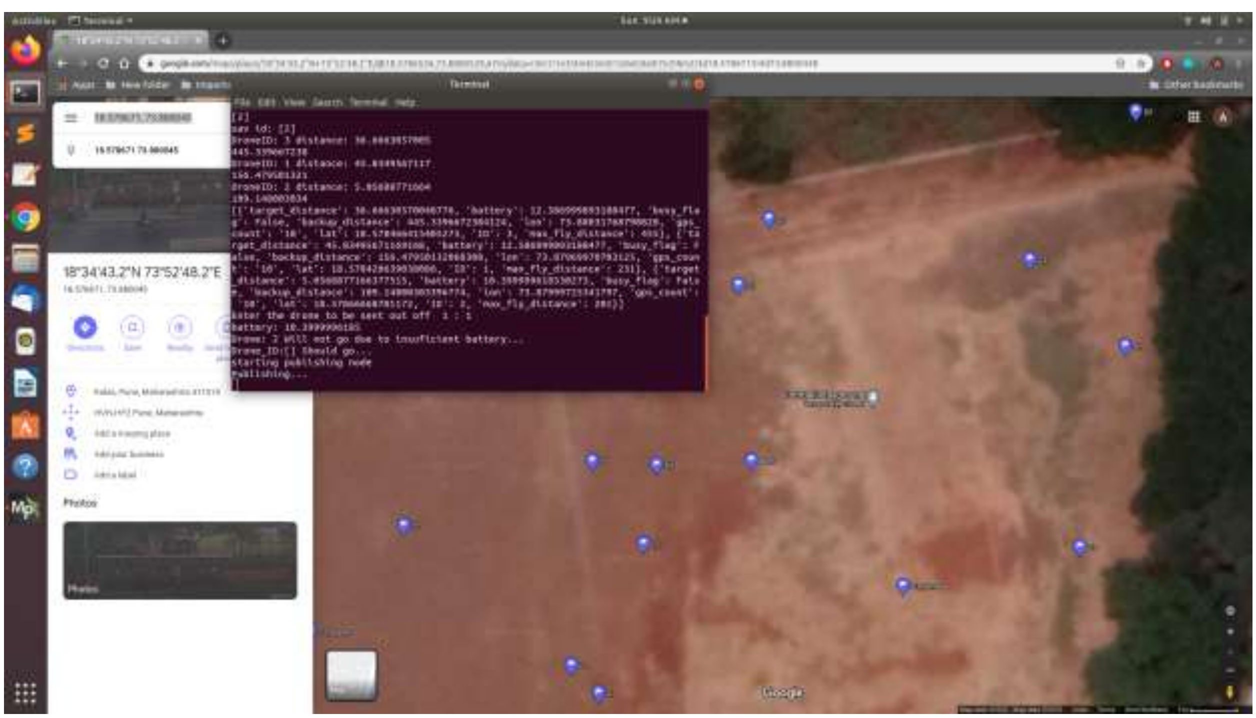
3. Destination point between 2 Drones and max_fly_distance same

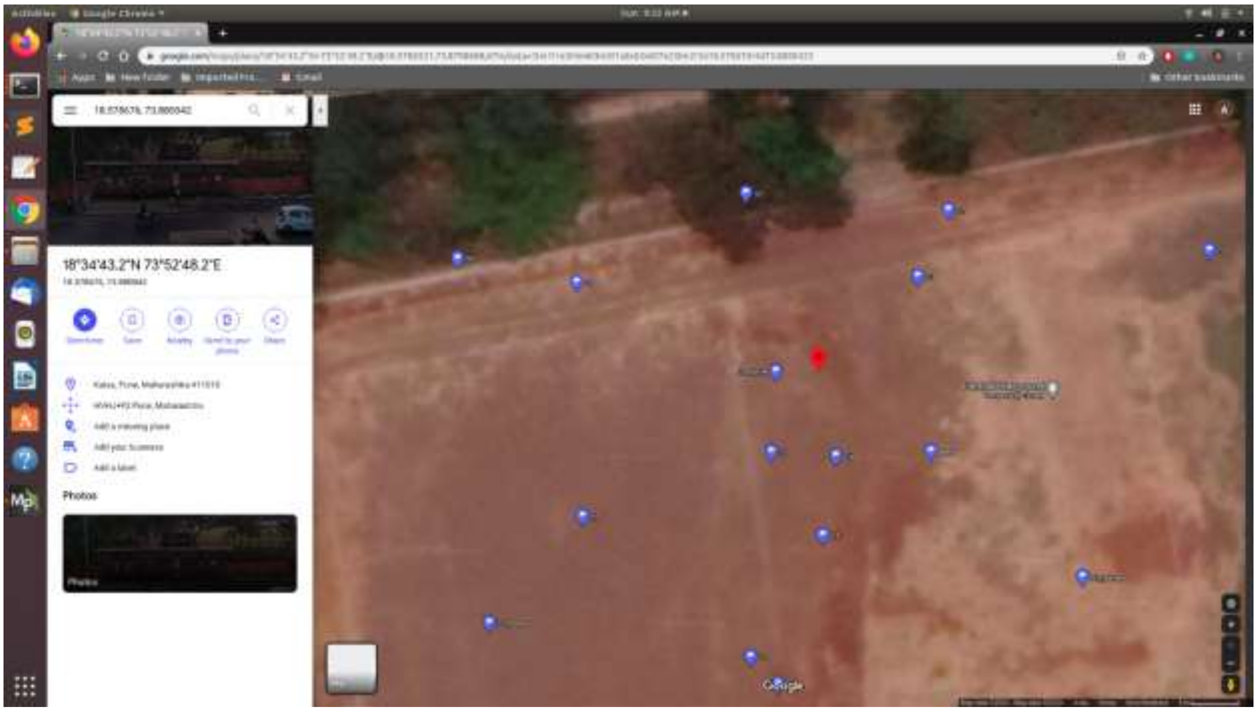


4. Destination near to Drone1 but battery is below critical

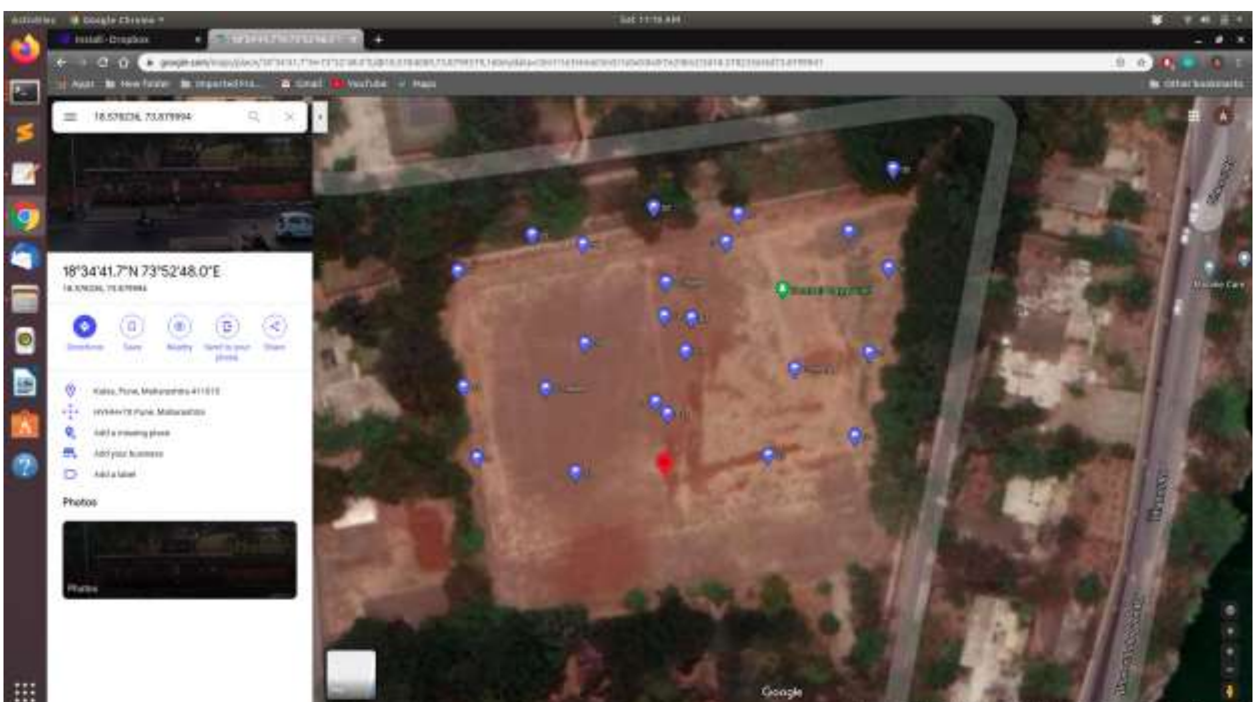


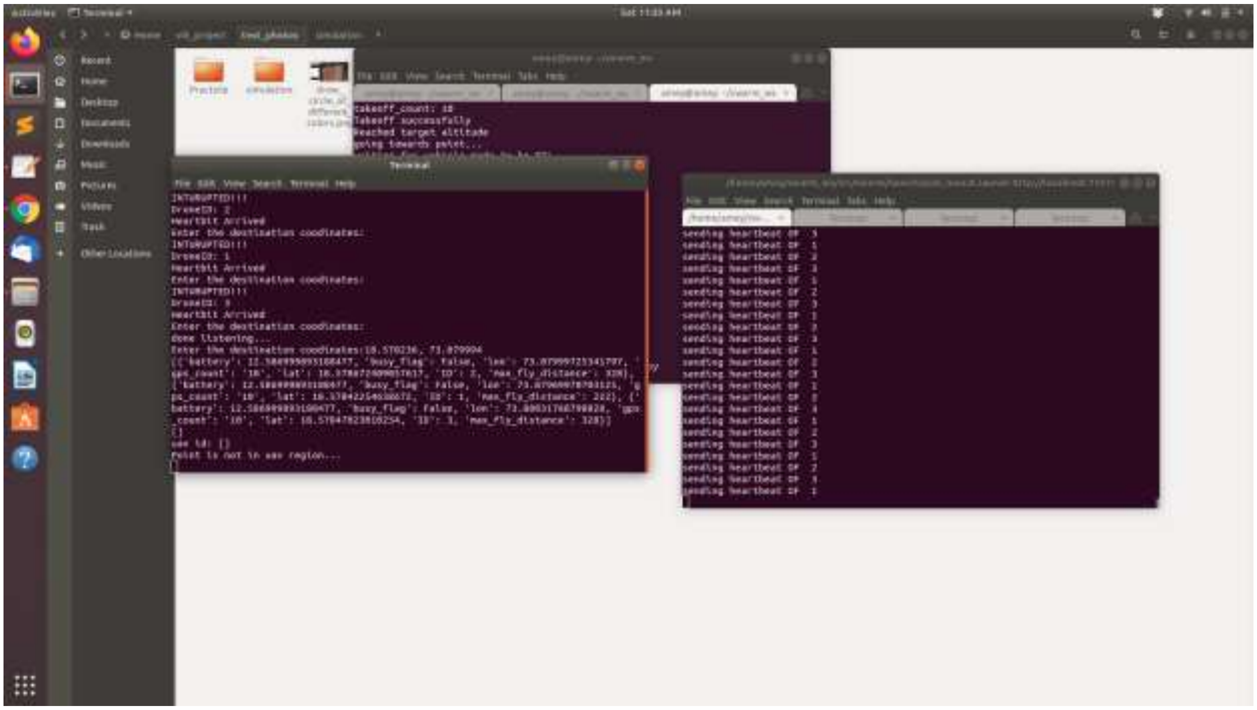
5. Near to Drone2 but battery is critically low





6. Point out of all Drone region





Applications

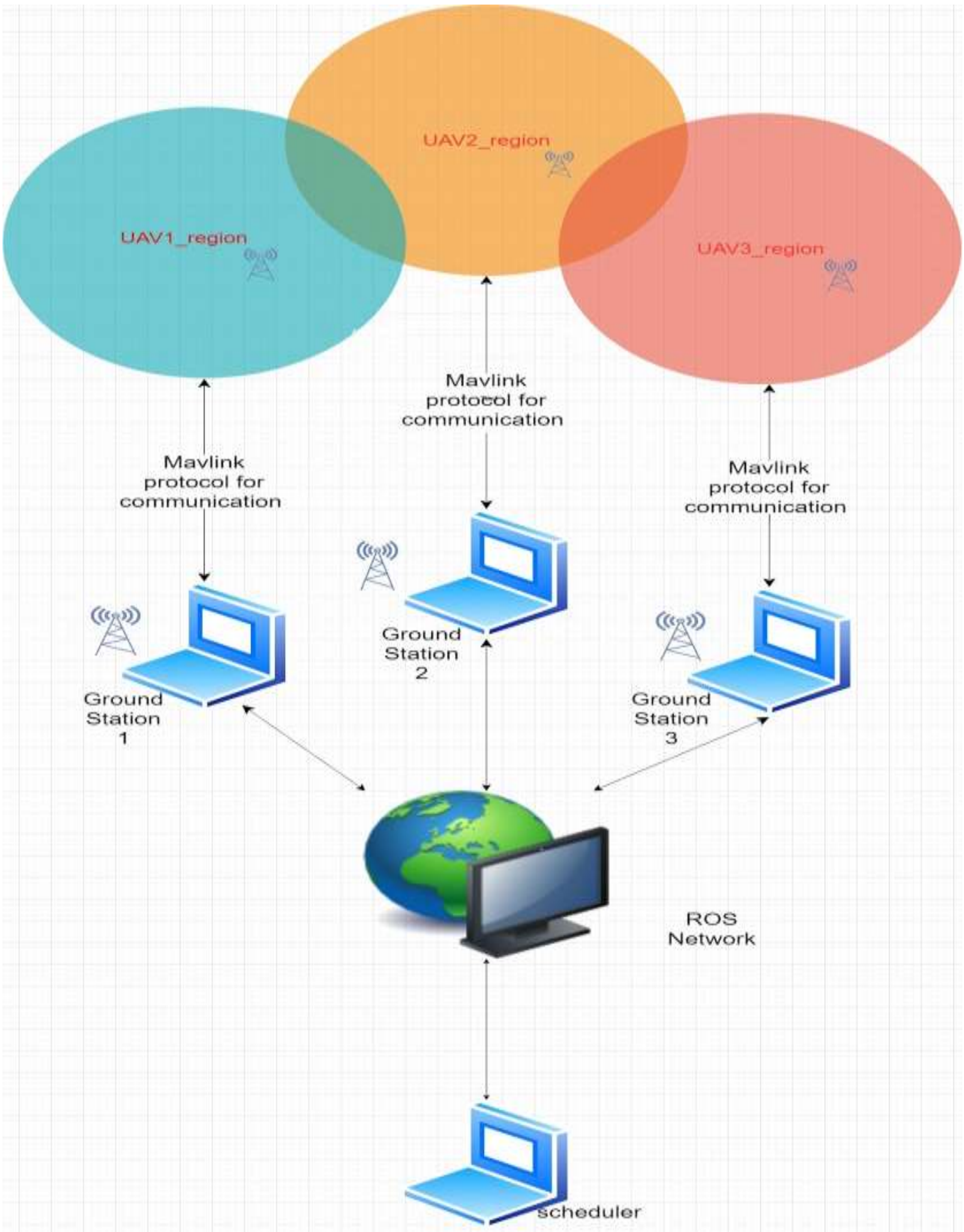
- These systems of UAVs can be widely used for surveillance of prohibited area. This system is totally autonomous and reliable.
- These UAVs with object detection can track the enemy and send its location to the ground station.
- Advantages to swarm include time-savings, reduction in man-hours, reduction in labor, and a reduction in other associated operational expenses. The use of a coordinated number of UAVs surveying an entire farmstead with little to no operator intervention would greatly increase efficiency and could revolutionize precision agriculture.
- The most notable application of UAV swarm is delivery services. Amazon and United Postal Service have indicated interest in using UAS for package delivery (Amazon 2017; MacFarland 2017). Using a typical remote pilot and a single sUAS, package delivery would be inefficient. Swarms of UAV's with coordinated control and communication capabilities would be efficient in this application.
- Search and rescue, forest fire monitoring and fighting, flood and earthquake response, etc.
- First Aid delivery
- Military observation of regions

- Theatrical Performance
- Building inspections - more UAV’s, more coverage
- Agriculture

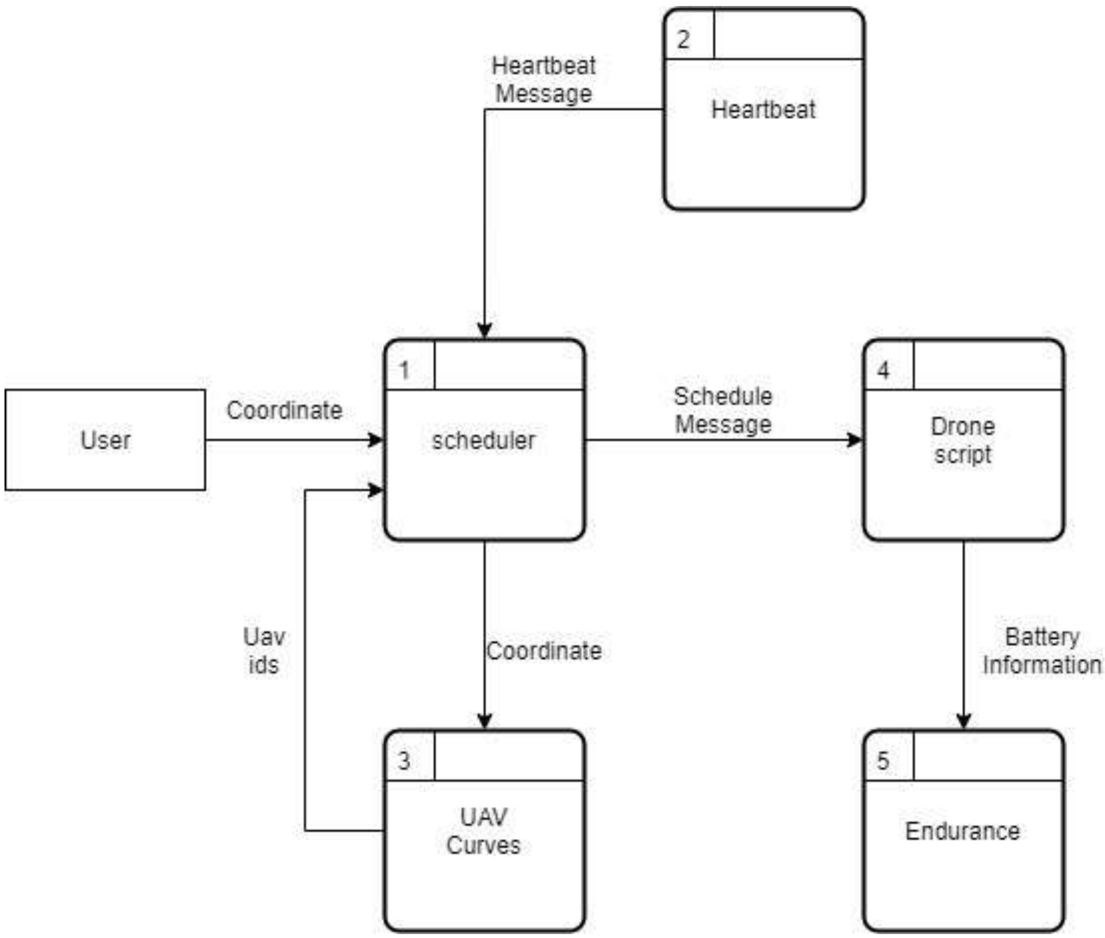
Scope and future work

- The data received by the robots need to be stored as a log and processed as per the assigned tasks.
- This data will be used for efficient implementation of swarming.
- For collision avoidance sensors need to be added on the four sides of the UAV.
- For object detection a camera and a processing unit is needed on each UAV. This requires a rigorous training on the huge data sets for efficient results.

System architecture

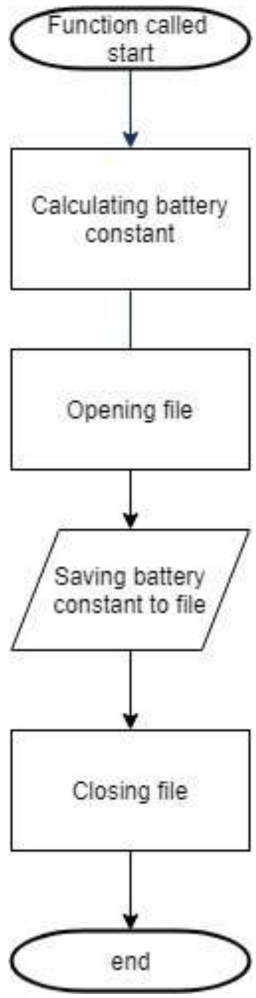


Data flow diagram

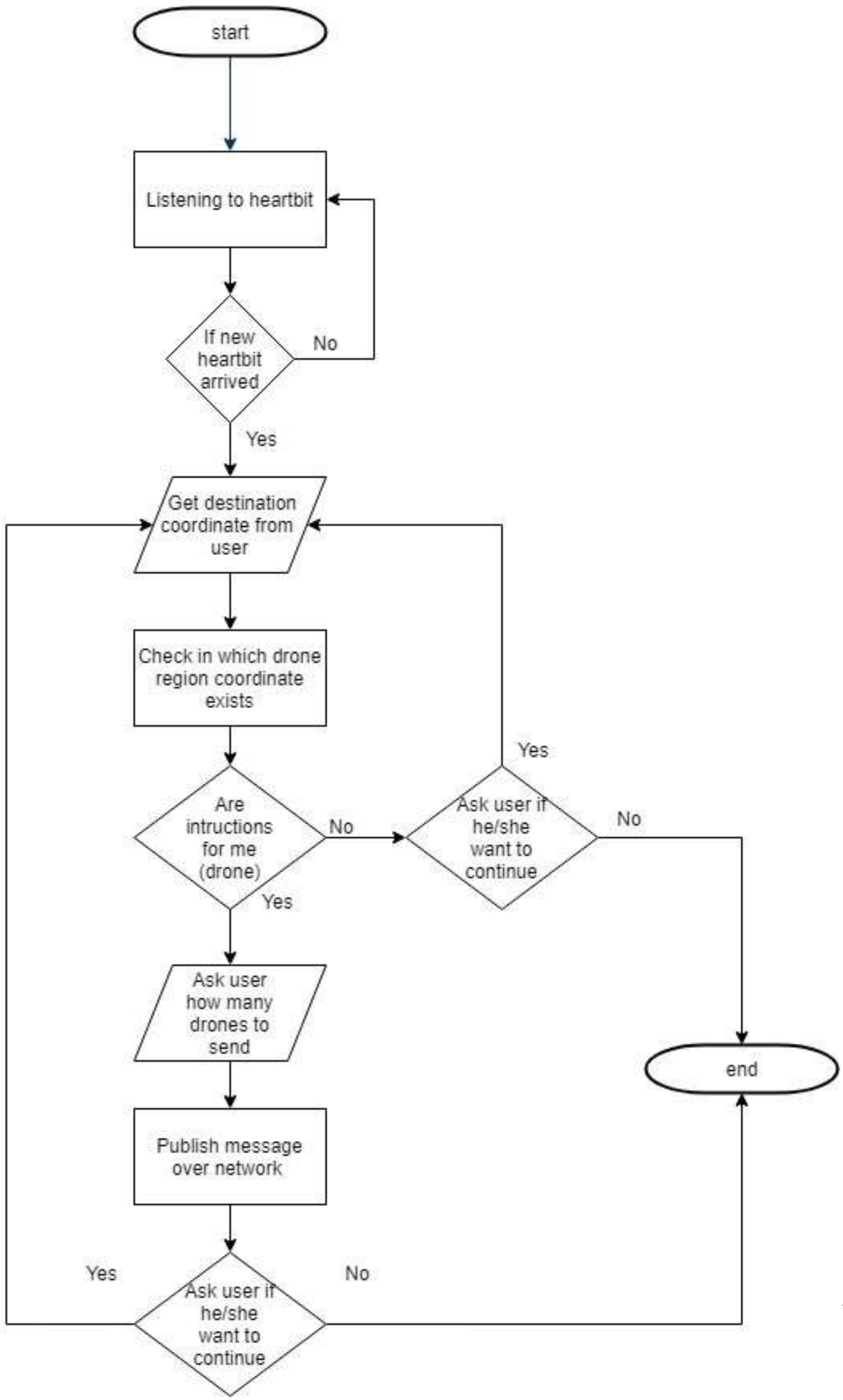


Flow chart

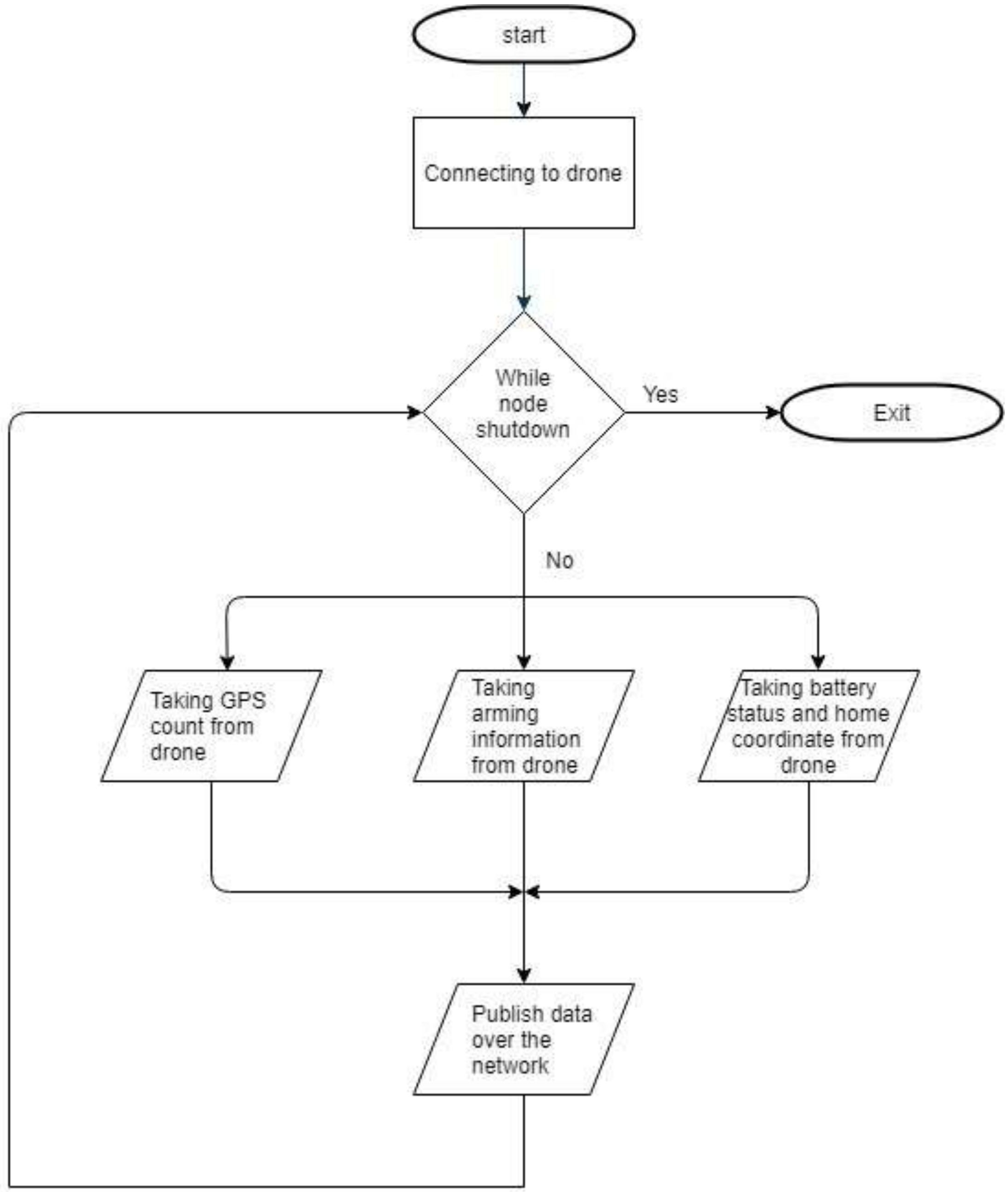
Endurance:



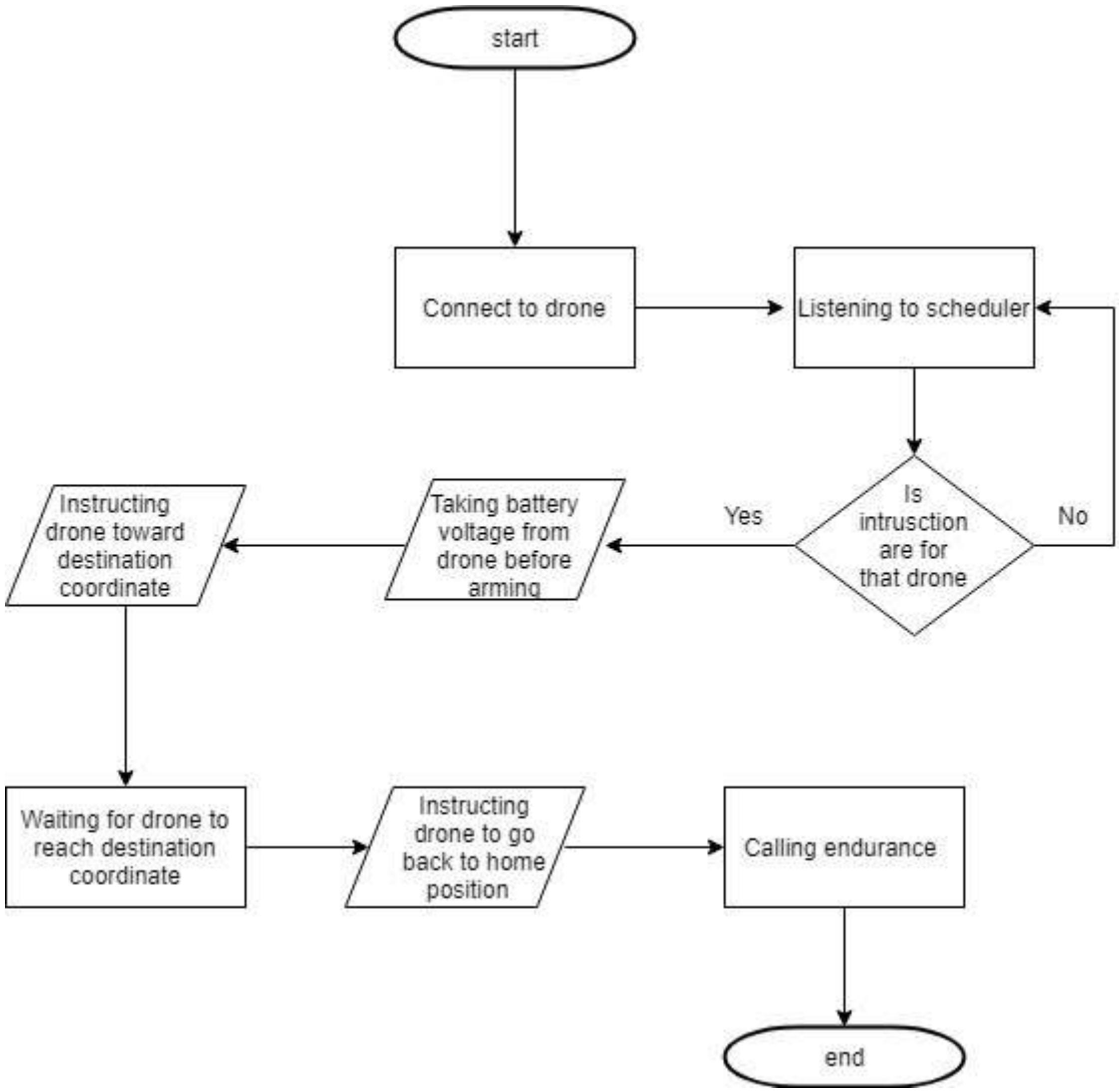
Scheduler


start
Listening to heartbit
If new heartbit arrived
No
Yes
Get destination coordinate from user
Check in which drone region coordinate exists
Are intructions for me (drone)
No
Yes
Ask user if he/she want to continue
Yes
No
Ask user how many drones to send
Publish message over network
Ask user if he/she want to continue
Yes
No
end

Heartbeat


start
Connecting to drone
While node shutdown
Yes
Exit
No
Taking GPS count from drone
Taking arming information from drone
Taking battery status and home coordinate from drone
Publish data over the network

Drone


start
Connect to drone
Listening to scheduler
Is intrusction are for that drone
Yes
No
Taking battery voltage from drone before arming
Instructing drone toward destination coordinate
Waiting for drone to reach destination coordinate
Instructing drone to go back to home position
Calling endurance
end

Reference:

1. 1.https://www.techrepublic.com/article/the-10-most-popular-platforms-developers-use-to-code-projects/
2. https://ubuntu-mate.org/
3. https://www.movablExternal Interface Requirements: e-type.co.uk/scripts/latlong.html
4. 4.https://www.igismap.com/formula-to-find-bearing-or-heading-angle-between-two-points-latitude-longitude/
5. 5.https://www.sciencedirect.com/science/article/pii/S2452414X18300086
6. https://www.google.com/search?q=Swarm+coordination+of+mini-UAVs+for+target+search+using+imperfect+...+-+arXiv&rlz=1C1CHBF_enIN894IN894&oq=Swarm+coordination+of+mini-UAVs+for+target+search+using+imperfect+...+-+arXiv&aqs=chrome..69i57.793j0j4&sourceid=chrome&ie=UTF-8
7. https://www.semanticscholar.org/paper/Nature-Inspired-Drone-Swarming-for-Real-Time-Aerial-Hildmann-Kovacs/6d4b05b83242b7a7a0ac17ba0f85396f8045c52
8. https://authors.library.caltech.edu/87925/1/tro-aerial-robotics_final.pdf
9. https://www.researchgate.net/publication/266740038_Swarming_Algorithm_for_Unmanned_Aerial_Vehicle_UAV_Quadrotors_Swarm_Behavior_for_Aggregation_Foraging_Formation_and_Tracking
10. https://wiki.ros.org/ROS/Tutorials