Multi-Drone Simulation Report

By Suchit Gupta

Problem Statement:

Most Unmanned Aerial Vehicle (UAV) Systems which are currently in use are either completely manually operated or at best semi-autonomous. In this project, we aim to provide a fully autonomous solution for UAV applications.

The TELEDRONES Project at RBCCPS, IISc primarily aims to develop a system of drone swarms to aid in Surveillance at various army bases. This project has three crucial components -

1. Autonomous Surveillance – The drones must have the capability to survey along a given path and filter out anomalies (such as human intruders) without any human intervention.

2. Charging Pads – These are landing stations for the drones, where a drone may land and charge its battery in order to perform another sortie.

3. Precision Landing – In order for the drone to land on the charging pads, we require a precision landing algorithm using which a drone may land with minimal error in position.

In this report, we will focus on the Autonomous Surveillance aspect of the drone.

Prior Work:

Efforts have been made to modify Quad copter Drones to make them suitable for Surveillance tasks. However these solutions do not provide an end to end solution to enable a swarm of drones to work together and effectively provide surveillance. Developing a working system of drone swarms is crucial because to provide surveillance coverage over a significant area, drones will need to cooperate and also be aware of their surroundings as other objects might be flying in close vicinity. Further, a centralised Ground Station is required for smooth centralized operation of the drone swarm.

This project is not only a research foundation for Surveillance Systems but also aims at deployment of such a system at various army camps across India. For this purpose, certain practical and real world constraints and considerations are also taken into account.

Simulation Software Specifications:

* Physics Environment: Gazebo 9.0
* Application Software: ROS Melodic
* Communication Protocol: mavlink
* Program Interface: Python (rospy)

Hardware Specifications:

* Vehicle: Q450 Quad copter
* Autopilot: Pixhawk Cube
* Firmware: Px4
* Onboard System: Odroid

BASIC SETUP:

Working with Drones requires a good understanding of both, the hardware and software components involved. My learning of the two components was done in parallel. So, I will divide the further text into two parts: Software and Hardware.

Please Note: Here, I have used the heading “Software” to include parts which are involved in simulations (in the lab), and the heading “Hardware” includes parts which are involved in testing of the drone (in the airfield).

Software

To start off with the software aspects of the project, one must first access the RBCCPS-TELEDRONES page on Github. A link to the same is given below (If the link does not open, it is likely that you either do not have permission or have not logged into GitHub with your account) –

<https://github.com/rbccps-iisc/AutonomousDrones>

In the page (referred by the link above), open README.md and install all the softwares and libraries as mentioned under the heading, Simulation Setup.  
Please note: Recommended operating system version is UBUNTU 18.04 and the recommended Gazebo Software version is gazebo9 (as of 2021).

Installing UBUNTU Operating System: If you do not already have UBUNTU installed on your laptop, I recommend following the link below, which describes a procedure to bootstrap UBUNTU from a CD or pen drive

<https://ubuntu.com/tutorials/tutorial-create-a-usb-stick-on-windows#1-overview>

The following link describes the procedure to dual boot your system with UBUNTU  
<https://itsfoss.com/install-ubuntu-1404-dual-boot-mode-windows-8-81-uefi/>

Hardware

Learning of the hardware aspects of the drone is best done by observing the Drone itself.

First, I shall mention details of the drone’s hardware components.  
Please Note: All details mentioned here are as per the drone used in 2021.

**Chassis:** The mechanical body of this drone was assembled at RBCCPS. It is an F450 Quad copter.

**Battery:** The drone uses Lithium Polymer Batteries as its power source. It is a 3-cell battery.

**Processor:** A processor is the core functional unit of the drone, which takes in data from the sensors, and computes algorithms, enabling the drone to perform tasks. The processor on this drone is a 64-bit [NVIDIA TX2 Processor](https://developer.nvidia.com/embedded/jetson-tx2-4gb).

**Sensor:** As per requirement, a variety of sensors are used in drones which aid in flying, as well as performing tasks which make the drone intelligent. This drone primarily uses a [LIDAR sensor](https://en.wikipedia.org/wiki/Lidar#:~:text=Lidar%20(%2F%CB%88la%C9%AAd,D%20representations%20of%20the%20target.) (to measure height of the drone above ground), Cameras (to help identify human intruders), and GPS (to locate the drone in world coordinates, and aid in navigation). Apart from these, there are sensors inside the auto pilot itself. These include an [Inertial Measurement Unit](https://en.wikipedia.org/wiki/Inertial_measurement_unit#:~:text=An%20inertial%20measurement%20unit%20(IMU,%2C%20gyroscopes%2C%20and%20sometimes%20magnetometers.) and a Barometer.

There are many modes in which the drone can be flown. The full list of modes available can be seen in the [Px4 Documentation](https://docs.px4.io/master/en/getting_started/flight_modes.html). However, in our project, the drone is flown in mainly the four following modes-

* Stabilize Mode: the drone is completely dependent on the control sticks of the RC
* Position Mode: similar to Stabilize mode but also uses the GPS and periphery sensors to stabilize itself under adverse conditions
* Mission Mode: drone performs the autonomous navigation without any input from the companion computer
* Offboard Mode: the companion computer takes over the control of the drone and commands the drone to do certain actions based on the code being executed in the companion computer
* Land Mode: drone stops and lands at the same position

In addition to these, there also exists a fail-safe mode, which is activated in case the communication between the drone and controller is lost. In this case, the drone automatically goes to “Return to Land” and descends to ground immediately.

In addition to the drone, we also require a Ground Control Station (GCS). The GCS communicates with the drone through the autopilot. The GCS runs using a firmware named MAVProxy, which uses [MAVLink protocol](https://mavlink.io/en/messages/common.html) for establishing communication. Here, we use [QGroundControl](http://qgroundcontrol.com/) as GCS software.

Implementation:

Implementation of the multi-drone system has mainly been carried out through Gazebo Simulations. The software stack was developed in the following phases -

Phase 1 – One Drone, One Landing Pad

Initially base software was developed to enable a single drone to carry out autonomous surveillance mission. In this development phase, a single drone was programmed to continue surveillance along the perimeter of a boundary wall until its battery level dropped below a certain threshold. The drone would then autonomously guide its path to a landing pad, and on being charged to a sufficient battery level, it would take off and resume its surveillance mission.

**Note:** The program is uploaded to the drone through its onboard computer.

The following are the functionalities provided in the drone’s onboard computer source code-

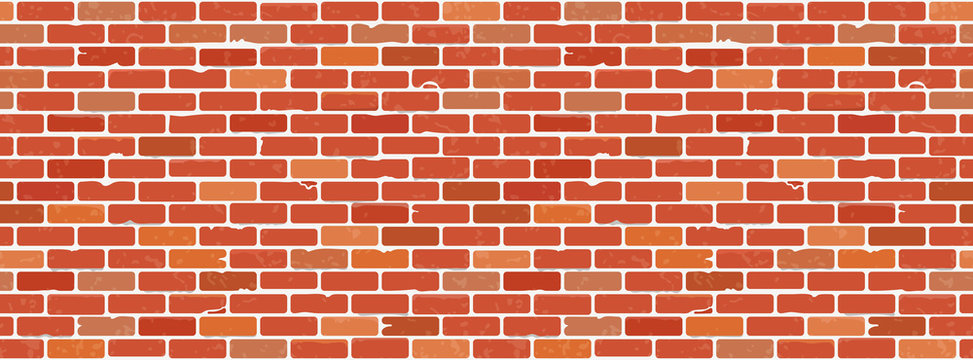
**go\_to\_location:** After charging, drone takes off and goes to its previous location to continue surveillance. (In case of the first sortie, drone goes to initial point)

**normal\_mission:** This is the mission mode in which the drone surveys along the perimeter boundary wall.

**go\_to\_home:** When the drone’s battery level falls below a threshold level, the drone lands on a charging pad.



Surveillance Drone



OpenCV: Detection of ArUco Markers

Landing Pad

Fig 1: Surveillance using One Drone and One Landing Pad

The following link points to simulation of autonomous surveillance with a single drone and a single landing pad –

<https://youtu.be/AkpkbVNCzhg>

**Problem:** Since there is only one drone, while it charges there is no surveillance taking place. This defeats the purpose as practically surveillance needs to be done 24/7.

Phase 2 – Two Drones, One Landing Pad

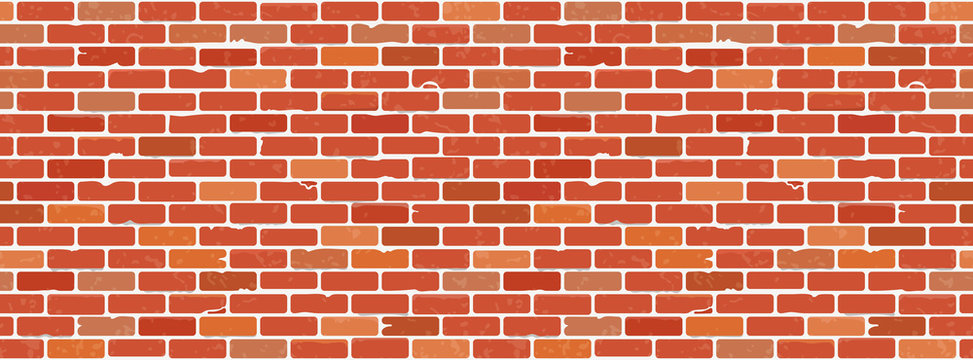
To solve the above mentioned problem, another drone was introduced into the system. In this setup the drones work in tandem. So while one drone charges, the other drone is doing surveillance.

In order to accommodate these enhancements, two major changes were made –

1. In each of the drones’ onboard computer program, a function for **hover\_and\_wait** was added. This function is called when a drone’s battery is critically low and requires charging. In this scenario, the drone rises to a higher altitude and waits for the replacement drone to arrive.
2. A **Ground Control Station (GCS)** program was developed. This program runs on a remote server and is responsible for coordination between the drones. The GCS takes in sensory information from the drone and relays back orders, which the particular drone must then execute.



Drone 1





Drone 2

OpenCV: Detection of ArUco Markers

WiFi Links

Landing Pad

**GCS**

Fig 2: Surveillance using Two Drones and One Landing Pad

The following link points to simulation of autonomous surveillance with two drones and a single landing pad –

<https://youtu.be/ukTMm_-_qzw>

**Problem:** This setup assumes 1:1 charging to discharging ratio of battery. However this is not practically possible with LiPo batteries which are used to power drones. Further, the single landing pad works well in the normal case, however if in a situation both drones require charging, we would need more such landing areas.

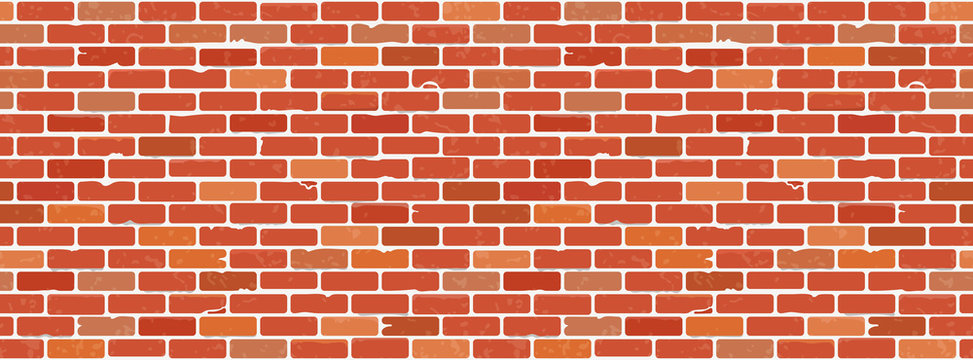
Phase 3 – ‘n’ Drones, ‘n’ Landing Pads

On account of the above mentioned problem, the GCS program was made more robust and can now handle systems with ‘n’ number of drones and ‘n’ number of charging pads.

In this setup, each drone is located on top of a charging pad in the beginning. The initial GPS location of the drone is stored and assigned as the coordinates for the landing pads’ locations.



Drone 1



Drone 3

Drone 2

OpenCV: Detection of ArUco Markers OpenCV: Detection of ArUco Markers

WiFi Links

Landing Pad 2

Landing Pad 1

**GCS**

Fig 3: Surveillance using ‘n’ Drones and ‘n’ Landing Pads

The following link points to simulation of autonomous surveillance with three drones and three landing pads –

<https://youtu.be/13igzEYSsic>

**Problem:** As seen in the video, this system can handle multiple drones, but can only manage one drone in flight at any given point in time. This will not work in practice, due to lack of coverage. For example, if there is an army base with a perimeter of 2km and one drone can provide full coverage for only up to 200m, then we would need 10 drones flying simultaneously for surveillance of the complete boundary.

Phase 4 – ‘m’ Groups

The above mentioned problem was solved by grouping the drones. In this system, we form ‘m’ groups of drones, and each group is in charge of only a particular sector along the wall. Therefore, each group will have one drone flying at any given point in time and that would result in a total of ‘m’ drones flying at any given point in time.

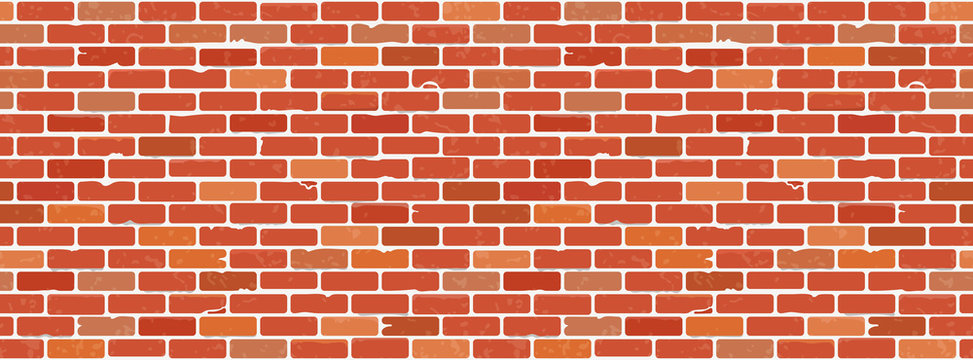
Group 1

Group 2

Drone 3

Drone 1



WiFi Links

Drone 2

Drone 4

OpenCV: Detection of ArUco Markers OpenCV: Detection of ArUco Markers

Landing Pad 1

Landing Pad 2

**GCS**

Fig 4: Surveillance using ‘m’ Groups of Drones

The following points to simulation of autonomous surveillance with two groups of drones. The 1st group does surveillance along the west side, while the 2nd group does surveillance along the east side –

<https://youtu.be/r56CYxFSMZo>

Demo:

The following link points to a simulation demo of the system with 2 drones and 1 landing pad in a real world scenario. Here the landing pad is located on top of a tall building which serves as a good vantage point for the drones. The end of the video also includes a short portion with a drone landing on a prototype of the landing pad.

<https://youtu.be/z5cgxDlHG88>

Future Work:

* **Field Test:** We are currently preparing drones in order to test the multi-drone autonomous surveillance system in the field.
* **Anomaly Detection:** The program for anomaly detection is yet to be included in the drone’s onboard computer.
* **Telecommunication:** In case of an anomaly being detected, we plan to give control of the drone to a guard who may use simple keyboard commands to manoeuvre the drone in order to follow the anomaly’s movements, use other components onboard the drone to deal with the situation, take a closer look at the anomaly to get a clearer picture, or capture the scene through different camera angles.