

OPERATING DRONES UNDER ROBOT  
OPERATING SYSTEM (ROS)

**DAREN KHO YING CHENG**

ELECTRICAL AND ELECTRONIC ENGINEERING  
UNIVERSITI TEKNOLOGI PETRONAS  
JANUARY 2017



# **Operating Drones under Robot Operating System (ROS)**

by

Daren Kho Ying Cheng  
18246

Dissertation submitted in partial fulfilment of  
the requirements for the  
Bachelor of Engineering (Hons)  
Electrical and Electronic Engineering

JANUARY 2017

Universiti Teknologi PETRONAS  
Bandar Seri Iskandar  
31750 Tronoh  
Perak Darul Ridzuan

CERTIFICATION OF APPROVAL

**Operating Drones under Robot Operating System (ROS)**

by

Daren Kho Ying Cheng

18246

A Project dissertation submitted to the  
Electrical and Electronics Engineering Programme

Universiti Teknologi PETRONAS

In partial fulfilment of the requirement for the

BACHELOR OF ENGINEERING (Hons)

(ELECTRICAL & ELECTRONIC)

Approved by,

---

Dr Mohamad Naufal Bin Mohamad Saad

UNIVERSITI TEKNOLOGI PETRONAS

TRONOH, PERAK

## CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.

---

Daren Kho Ying Cheng

## **ABSTRACT**

An Unmanned Aerial Vehicle (UAV) is a type of aircraft which can be driven without any pilot, crew or passenger on board. Hence, the size of UAV can be as small as a toy or even as big as a normal aircraft depending on its usage. The term UAV does include both Remotely Piloted Vehicles (RPVs) as well as autonomous drones [1]. Thus, an UAV is widely known as a drone. A drone is normally equipped with lots of sensors such as ultrasonic sensors, gyroscope, accelerometer, proximity sensor, camera and more which allows the drone or the flying robot to be more intelligent while collecting more data for the controller. Basically, the drone will be operated and controlled under Robot Operating System (ROS). Whereas the final aim of this project is to operate multiple drones simultaneously using only one controller.

# Table of Contents

<b>ABSTRACT.....</b>	<b>IV</b>
<b>TABLE OF CONTENTS.....</b>	<b>1</b>
<b>LIST OF FIGURES.....</b>	<b>3</b>
<b>LIST OF TABLES.....</b>	<b>4</b>
<b>CHAPTER 1.....</b>	<b>5</b>
1 BACKGROUND .....	5
2 PROBLEM STATEMENTS .....	8
3 OBJECTIVES .....	8
4 SCOPE OF STUDIES.....	9
<b>CHAPTER 2.....</b>	<b>10</b>
<b>CHAPTER 3.....</b>	<b>14</b>
1 RESEARCH METHODOLOGY .....	14
2 KEY MILESTONES.....	17
3 GANTT-CHART .....	18
<b>CHAPTER 4.....</b>	<b>19</b>
1 CONNECTING DRONE.....	20
2 CONTROLLING DRONE.....	23
3 RETRIEVING DATA FROM DRONE .....	24
4 CHANGING FIRMWARE OF DRONE 2014 .....	26
5 SETTING UP OTHER COMPUTERS .....	28
6 BUILDING MULTI-OUTPUT-CONTROLLER .....	30

6.1 Single-multiplexer-dual-output controller .....	30
6.2 Dual-multiplexer-dual-output controller .....	33
6.3 Trio-multiplexer-trio-output controller .....	35
7 DUAL DRONES .....	36
8 TRIO DRONES .....	39
8.1 Vertical Movement .....	39
8.2 Triangular Movement .....	40
<b>CHAPTER 5.....</b>	<b>42</b>
<b>REFERENCES .....</b>	<b>43</b>



# List of Figures

Figure 1 Spy Drone [3]	5
Figure 2 Agricultural Drone [4]	6
Figure 3 Flight Dynamic of UAV	12
Figure 4 Project Key Milestones	17
Figure 5 First Terminal	20
Figure 6 Second Terminal	21
Figure 7 Third Terminal	23
Figure 8 Forth Terminal	24
Figure 9 Terminal on MAC OS	26
Figure 10 Interface of Cyberduck	26
Figure 11 Uploading plf file to Drone	27
Figure 12 Interface of Virtual Box Manager	28
Figure 13 Export Setting	28
Figure 14 Output of Export	29
Figure 15 Importing Appliance	29
Figure 16 Layout of USB Cable Wiring	30
Figure 17 Connection of Single-multiplexer-dual-output	31
Figure 18 Single-multiplexer-dual-output Controller	32
Figure 19 2 Multiplexers in used	33
Figure 20 Connection of Dual-multiplexer-dual-output Controller	34
Figure 21 Connection of Trio-multiplexer-trio-output Controller	35
Figure 22 Flight Movement of Dual Drones	36
Figure 23 Search and Rescue using dual drones	37
Figure 24 Testing on Dual Drones	37
Figure 25 Dual Drones Flying in an Opposite Direction	38

Figure 26 Vertical movement of Trio Drones	39
Figure 27 Triangular Movement of Drones	40
Figure 28 Search and Rescue using Trio Drones	40
Figure 29 Testing on Trio Drones	41
Figure 30 Trio Drones Spreading Outwards in a Triangular Shape	41

## List of Tables

Table 1 Gantt Chart of FYP 1	18
Table 2 Gantt Chart of FYP 2	18
Table 3 State of Drone	25

# CHAPTER 1

## INTRODUCTION

### 1 Background

The drones as the first pilotless aircraft was created during the World War 1 [2]. With the exist of drones, armies are able to attack as well as spying enemies while safely remaining thousands of miles away from the battlefield. The size of a drones can be small as well for spying purposes which allowed it to remain stealth and detected while carrying mission [3].



*Figure 1 Spy Drone [3]*

Even though the drone was firstly created for military purposes, several huge company such as DJI, Parrot did some modification and redefine the industries which allowed the drones to be user friendly and harmless. Nowadays, drones are

able to benefit the society by bringing new perspective to professionals such as conservation, filmmaking, agriculture, search and rescue which hence allow them to work with higher efficiency [4]. Drones available in market are mostly quadcopter drone which focus more in stability and hovering unlike the fixed-wing aircraft that aim to be stealth with high speed for military use.



*Figure 2 Agricultural Drone [4]*

As for this project, the drone chosen must be equipped with a HD camera at its front part which allow it to run the tag recognition feature through its camera. Sensors such as gyroscope, accelerometer and ultrasonic sensor should be included as well in order to allow the drone to be more intelligent while flying autonomously. Moreover, the design of the drone must come with propeller guard as the drone will be tested with program that is not mature for both indoor and outdoor. Parrot AR Drone 2.0 is the final choice of hardware after all sort of consideration including price and availability.

As for software, this project utilizes Robot Operating System (ROS), a free and commonly used platform for robot developing. However, ROS is only available in

the Ubuntu Platform rather than the commonly used Windows. Hence, a virtual box booting Ubuntu 12.04 is running on top of Windows for this case. The concept of communicating between laptop and drone is that the laptop is linked to the drone via Wi-Fi. Hence, we are able to control as well as extracting data retrieved by the drone through ROS and leads to further development on top of these basis.

## **2 Problem Statements**

ROS is a free and commonly used platform for robot developing such as Parrot AR 2.0. Thus, one of the challenge of this project is to figure out the way to connect and control multiple drones simultaneously using ROS.

Besides, a drone can be known as the flying robot which is equipped with lots of sensors with no doubt. However, the problem lies on how to implement features on drone by utilizing its sensors & flight movement.

## **3 Objectives**

The objectives of this project are on programing the Parrot AR Drone 2.0 which can be further elaborated as below:

- To communicate with & control drones via ROS
- To implement features on drone by utilizing its sensors and flight movement while flying multiple of them at once (drone swarm)

#### **4 Scope of Studies**

This project is mainly about developing the drone with new features by using the equipped sensors and camera. In other words, this project is trying to upgrade the drone to make it smarter without doing changes on its hardware. Hence, this project is carried out by developing the drone via ROS running on Ubuntu 12.04 Precise. Multiple testing will be done as well in order to ensure the reliability of the newly developed feature.

Although this project is taking Parrot AR Drone 2.0 as testing, the features developed are aimed to benefit all kinds of programmable drones equipped with basic sensors such as cameras and ultrasonic sensors. The output of this project is also suitable for drones that are used in all kinds of professionals including entertainment. It is mainly because the upgraded feature such as obstacle avoidance via tag recognition is able to increase the survivability of drones as well as increasing the flying hours of drones if it is used to replace the usage of multiple ultrasonic sensors.

## **CHAPTER 2**

### **LITERATURE REVIEW**

Basically, the UAVs can be categorized into 3 categories based on their technologies which are the mini and micro UAVs, tactical UAVs and strategic UAVs [1].

As the name meant it, the mini and micro UAV is the tiniest UAV technology available. These small-designed platforms are only able to fly beneath 300m altitude. It is because these types of drones are mainly designed to operate indoor or “urban canyons” while recording and streaming via the camera equipped as well as feedbacking the data collected by the sensors armed. To be more specific, mini UAVs have the weight of less than 30kg and fly at the height ranging from 150m to 300m [5]. As for the micro UAVs, they are even tinier than the mini UAV with the weight of as little as 100g. Both mini and micro UAVs are designed for commercial and civil used.

Nevertheless, the second category, the tactical UAVs are designed for military purposes with the weight in between of 150kg and 1500kg. They are able to travel at higher altitude with the maximum of 8km. One of the best example of tactical UAV is the MQ-1 Predator that can be controlled at a maximum distance of 3000km with a for 40 hours [6]. Moreover, it is designed to be armed with precision guided missiles.

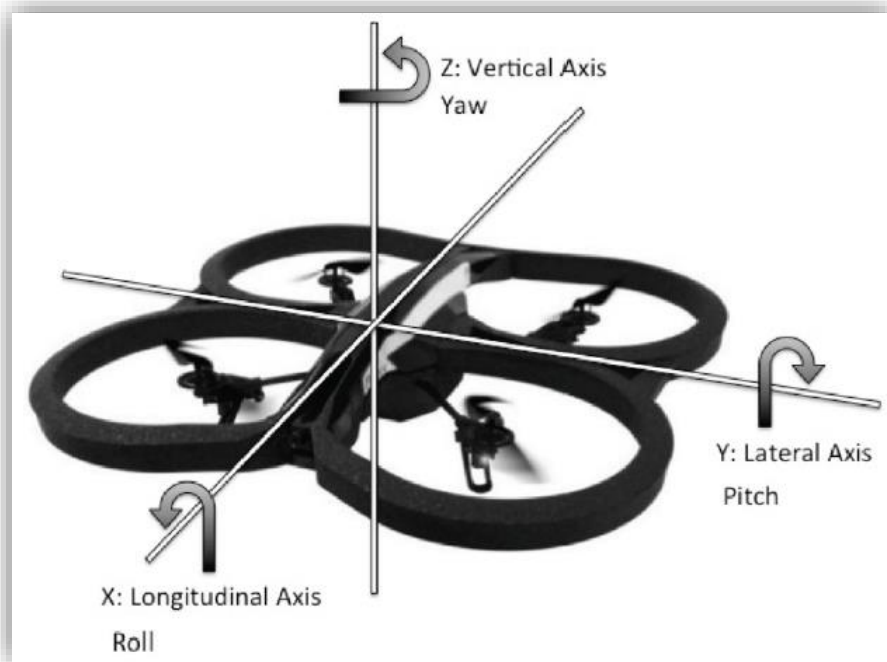


Next is the 3<sup>rd</sup> category, strategic UAVs which are bigger and heavier in design are able to travel at a higher altitude and longer flight ranges. The High Altitude Long Endurance (HALE) UAVs are the heaviest UAVs. With its heavy and large design, HALE is able to fly up to 20km altitude with maximum takeoff weight of 12000kg [7]. Helios is the HALE designed and operated by National Aeronautics and Space Administration (NASA) to do mapping, Earth monitoring as well as atmospheric observing.

Considering all sorts of limitation such as cost, availability and suitability, the type of UAV chosen for this project is micro UAV. Micro UAV have been applied in lots of fields and professionals since 20<sup>th</sup> century as it is able to accomplish tasks that are risky and hard for human. For example, UAV able to provide surveillance from bird-eye-view which is very useful for monitoring like search and rescue operations, monitoring area with high danger as well as filming from angle of helicopter without using a helicopter [8]. Moreover, UAV can be set or programmed to fly autonomously via all of the sensors equipped or manually under our control.

The most attractive part of a quadcopter UAV is its ability to hover, takeoff and landing vertically and flying either longitudinally or laterally [9]. Unlike a plane, it requires a large place to take off and landing while not able to hover at one spot which unable causes lots of trouble if use it for surveillance purposes. Thus, more and more researches are carried out in order upgrade the quadcopters to be more intelligent by making advances in surrounding exploration, ability to maneuver and also multi-craft communications.

Movement of a quadcopter UAV can be best described using the X, Y and Z axis based on the 3-dimensional Cartesian Coordinate System [10]. The z-axis is commonly known as the vertical axis whereas the x-axis is also known as the longitudinal axis while the y axis can be referred as the lateral-axis. Yaw is the term referring to the turning in z-axis, pitch as the noun for turning in y-axis and roll used to describe turning in x-axis [11]. All of the description above are visualized in the diagram below for a better understanding.



*Figure 3 Flight Dynamic of UAV*

The kinematics of drone also plays an important role in researching the physics and motion of a quadcopter drone [12]. First, the velocity and position of quadcopter in the inertial frame can be defined as:

$$\mathbf{x} = (x, y, z)^T$$

Next, the angle of roll, pitch and yaw in the body frame can be referred as:

$$\theta = (\phi, \theta, \psi)^T.$$

with the corresponding angular velocities are equal to

$$\dot{\theta} = (\dot{\phi}, \dot{\theta}, \dot{\psi})^T.$$

Relation as below can be applied to convert angular velocities above into angular velocity vector.

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

$\omega$  above is known as the angular velocity vector in the body frame. Rotation matrix  $R$  can be used to relate both the inertial frame and the body. By applying the ZYZ Euler angle convention while “undoing” the yaw, pitch and roll, formula as below is obtained [13].

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

Moreover, the torque generated by the electric motors can be calculated by using the formula:

$$\tau = K_t(I - I_0)$$

$T$  is the torque of the motor,  $K_t$  is the torque proportionality constant,  $I$  is the input current whereas  $I_0$  is the current with no load on motor. Besides, the voltage across the motor can be calculated as it is the total of the back-EMF and some resistive loss:

$$V = IR_m + K_v\omega$$

$V$  above is the voltage drop across the motor,  $R_m$  is the resistance of the motor,  $K_v$  is the proportionality constant while  $\omega$  is the angular velocity of the motor.

## **CHAPTER 3**

### **METHODOLOGY**

#### **1 Research Methodology**

This project is estimated and expected to be completed within 8 months or 2 semesters. In order to ensure the project to be done effectively and efficiently, it is separated into few sections as below:

##### **1. Background Understanding**

The first and the foremost is the preparatory stage that focus in collecting, familiarizing and hence understanding information regarding the project. As this is a continuation project by seniors, advises from supervisor and the graduated senior which is also the pioneer of this project are important.

Furthermore, research via online and published paper related to the technology of UAV do help. Basically, information gathered are related to the existing UAVs, pros and cons of existing UAVs as well as the consequences faced by UAVs. All of these are required in writing the background and literature review.

##### **2. Hardware and Software Setting Up**

As mentioned earlier, UAVs that will be used in this project are quadcopter micro UAV by Parrot with the model name of AR Drone 2.0 which are available in Centre for Intelligent Signal and Imaging Research (CISIR).

Next for the software, this project required to be ran on Robot Operating System (ROS) Fuerte based in Ubuntu 12.04 Precise on a laptop. Virtual Box plays an important role in running Ubuntu on my Windows based laptop virtually. Then, ROS Fuerte was installed in the Ubuntu according to the tutorial available in [wiki.ros.org](http://wiki.ros.org).

### 3. Connecting and Controlling Drone

This is one of the most difficult steps as it requires multiple trials in order to success in this step. In shorts, the steps are connecting the drone and laptop via the WiFi emitted by the drone. Then, run several packages such as `ardrone_autonomy`, `ar_recog` on the ROS terminal of Ubuntu in order to control the drone as well as running the tag recognition feature. Lots of example packages can be reviewed, downloaded and try via [github.com](http://github.com). Although this sound simple, there are lots of errors to be solved and this process takes a lot of time especially for a newbie like me who just start using ROS for the first time.

### 4. Changing Firmware of Drone Power Edition 2014

During the FYP 2, three AR Drones 2.0 Power Edition 2014 are provided by CISIR which allowed me to achieve the final objective of this project which is the drone swarm. However, the features done on the Drone 2010 are not compatible with the Drone 2014 as both of them are having different firmware. Downgrading the firmware of Drone 2014 helps in solving such problem.

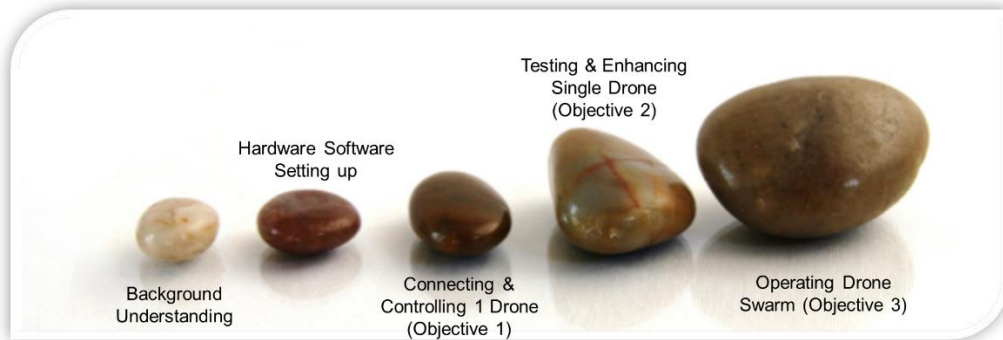
### 5. Drone Swarm

Drone swarm can be defined as controlling more than 1 drone. Hence, this objective will be tested by connecting and controlling 3 drones using 3

computers but with only 1 controller. All of the drones will perform some pattern while the input from the controller is received.

As 3 ROS are required, the other computers are required to be setup with the ROS Fuerte based in Ubuntu 12.04 Precise as well. An external keyboard is modified into a controller which is able to produce multiple similar outputs simultaneously.

## 2 Key Milestones



*Figure 4 Project Key Milestones*

Based on the project key milestones as shown above, all of the key milestones excluding the last were done in FYP1. Furthermore, this does mean that both objective 1 and objective 2 were successfully achieved as well.

Next, the last key milestone is meant to be approached during FYP2. Achieving the 3<sup>rd</sup> objective does signify completing this project. With all hard works and effort, the drone swarm can finally be operated in week 10 of FYP2 and further improvement were done in the following weeks until the end of FYP2.

### 3 Gantt-Chart

Table 1 Gantt Chart of FYP 1

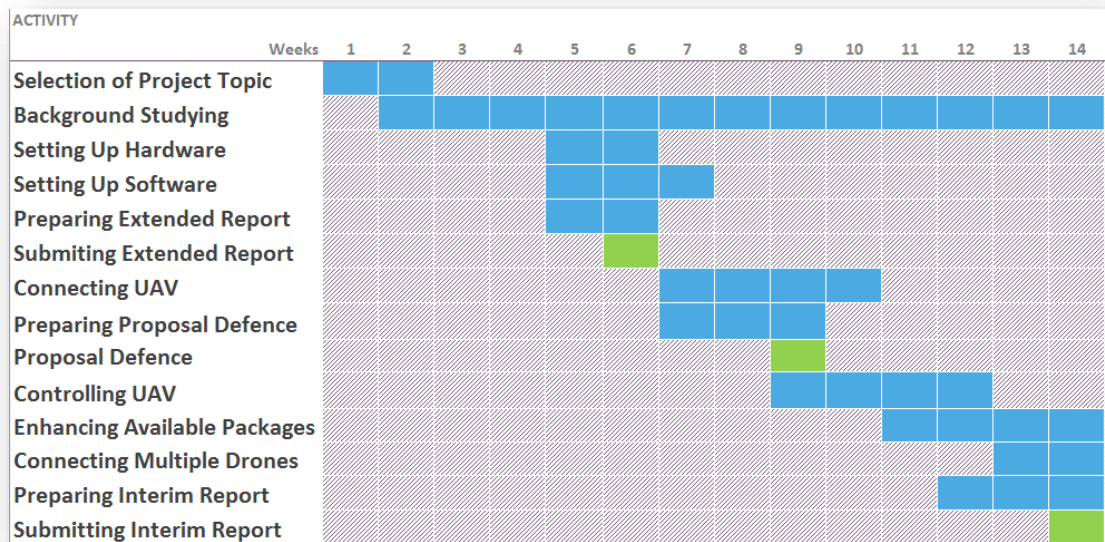
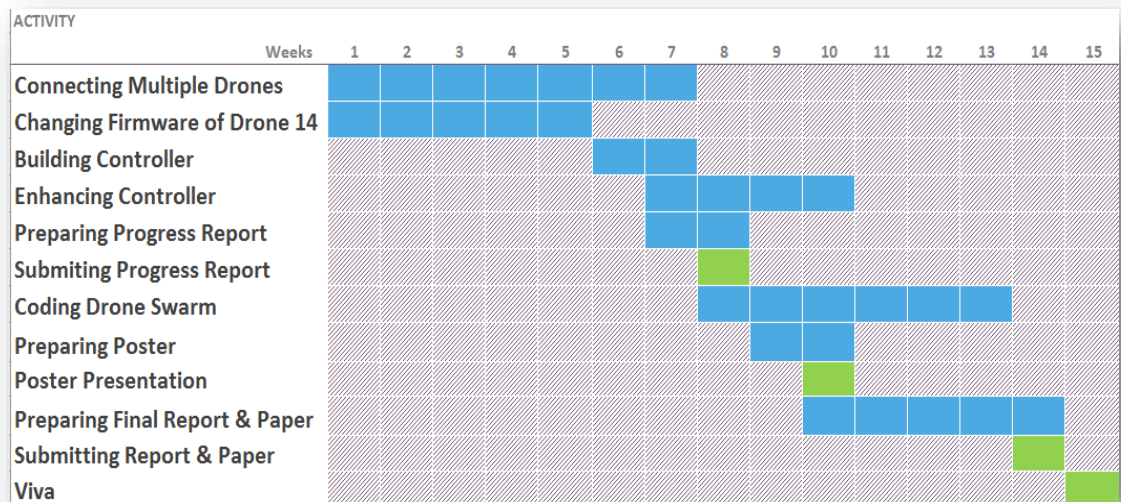


Table 2 Gantt Chart of FYP 2





## **CHAPTER 4**

### **RESULTS AND DISCUSSION**

Results of this project are gathered through several stages as follow:

1. Connecting Drone
2. Controlling Drone
3. Retrieving Data from Drone
4. Changing Firmware of Drone 2014
5. Setting up other Computers
6. Building Multi-Output-Controller
7. Dual Drones
8. Trio Drones

## 1 Connecting Drone

As ROS is a platform that runs on terminal of Ubuntu, the results below are gathered from the terminal of Ubuntu 12.04 LTS running on ROS Fuerte.

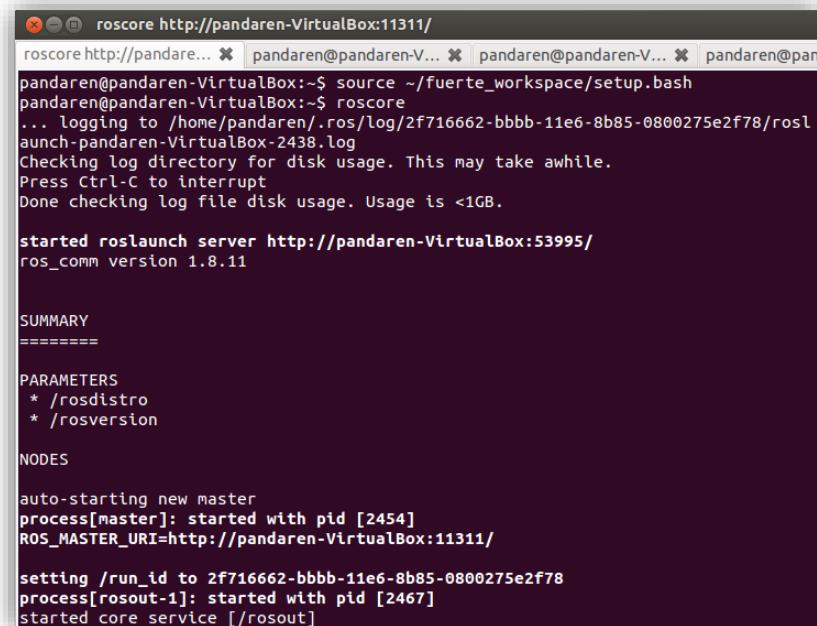
### i. First Terminal: Launching ROS Service

```
$ source ~/fuerte_workspace/setup.bash  
$ roscore
```

Command above is inserted into a new terminal of Ubuntu. Function of commands are explained as below:

1<sup>st</sup> line: Setup working environment.

2<sup>nd</sup> line: Launch ROS service in the terminal.



```
roscore http://pandaren-VirtualBox:11311/  
roscore http://pandare... ✖ pandaren@pandaren-V... ✖ pandaren@pandaren-V... ✖ pandaren@pan  
pandaren@pandaren-VirtualBox:~$ source ~/fuerte_workspace/setup.bash  
pandaren@pandaren-VirtualBox:~$ roscore  
... logging to /home/pandaren/.ros/log/2f716662-bbbb-11e6-8b85-0800275e2f78/rosl  
aunch-pandaren-VirtualBox-2438.log  
Checking log directory for disk usage. This may take awhile.  
Press Ctrl-C to interrupt  
Done checking log file disk usage. Usage is <1GB.  
  
started roslaunch server http://pandaren-VirtualBox:53995/  
ros_comm version 1.8.11  
  
SUMMARY  
=====  
  
PARAMETERS  
* /rostdistro  
* /rosversion  
  
NODES  
  
auto-starting new master  
process[master]: started with pid [2454]  
ROS_MASTER_URI=http://pandaren-VirtualBox:11311/  
  
setting /run_id to 2f716662-bbbb-11e6-8b85-0800275e2f78  
process[rosout-1]: started with pid [2467]  
started core service [/rosout]
```

*Figure 5 First Terminal*

### ii. Second Terminal: Establishing Connection with Drone

```
$ source ~/fuerte_workspace/setup.bash

$ roscd

$ cd sandbox

$ rosrn ardrone_autonomy ardrone_driver
```

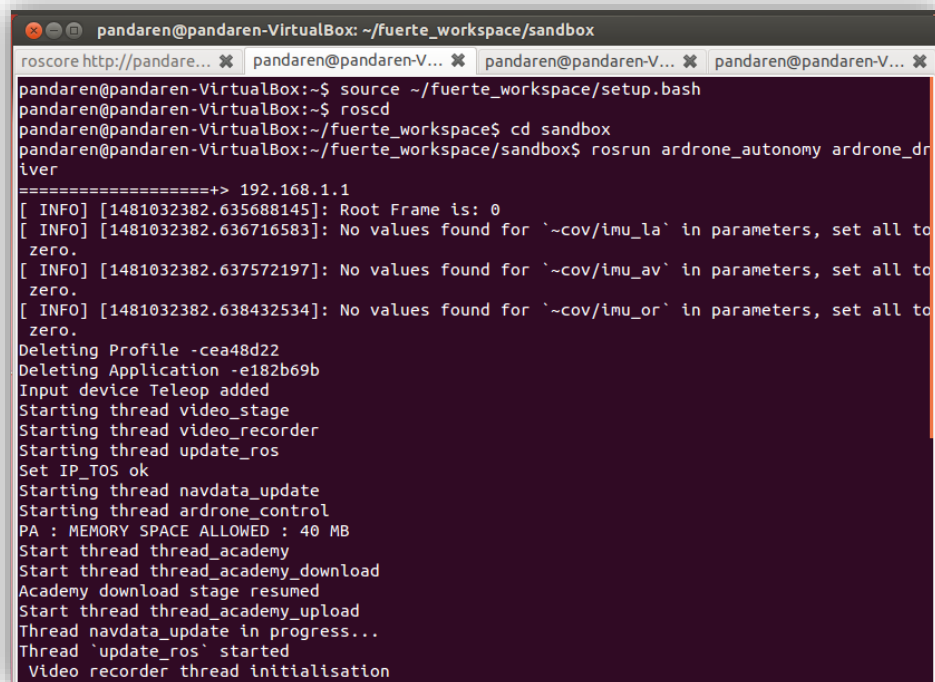
A new tab is open by right clicking the terminal. Commands above are inserted into the new tab. Function of commands are explained as below:

1<sup>st</sup> line: Setup working environment.

2<sup>nd</sup> line: Locate the package.

3<sup>rd</sup> line: Locate the package.

4<sup>th</sup> line: Run the ardrone\_driver inside the ardrone\_autonomy package



```
pandaren@pandaren-VirtualBox: ~/fuerte_workspace/sandbox
roscore http://pandare... pandaren@pandaren-V... pandaren@pandaren-V... pandaren@pandaren-V...
pandaren@pandaren-VirtualBox:~$ source ~/fuerte_workspace/setup.bash
pandaren@pandaren-VirtualBox:~$ roscd
pandaren@pandaren-VirtualBox:~/fuerte_workspace$ cd sandbox
pandaren@pandaren-VirtualBox:~/fuerte_workspace/sandbox$ rosrn ardrone_autonomy ardrone_driver
=====+> 192.168.1.1
[ INFO] [1481032382.635688145]: Root Frame is: 0
[ INFO] [1481032382.636716583]: No values found for '~cov/imu_la' in parameters, set all to
zero.
[ INFO] [1481032382.637572197]: No values found for '~cov/imu_av' in parameters, set all to
zero.
[ INFO] [1481032382.638432534]: No values found for '~cov/imu_or' in parameters, set all to
zero.
Deleting Profile -cea48d22
Deleting Application -e182b69b
Input device Teleop added
Starting thread video_stage
Starting thread video_recorder
Starting thread update_ros
Set IP_TOS ok
Starting thread navdata_update
Starting thread ardrone_control
PA : MEMORY SPACE ALLOWED : 40 MB
Start thread thread_academy
Start thread thread_academy_download
Academy download stage resumed
Start thread thread_academy_upload
Thread navdata_update in progress...
Thread 'update_ros' started
Video recorder thread initialisation
```

Figure 6 Second Terminal

Roscore functions as a bunch of programs and nodes are pre-requisites of a ROS-based system. Hence, the command ‘roscore’ is required to startup the ROS.

The command “\$ rosmake ardrone\_autonomy” is required for the first time running this package in order to extract the driver’s executable node, ardrone\_driver from the package itself.

Ardrone\_autonomy is the package which allowed the computer to communicate with the AR Drone 2.0 via the terminal. All of the motors, sensors and camera are accessible with the help of this package.

The package ardrone\_autonomy is available at the github website with the link [https://github.com/AutonomyLab/ardrone\\_autonomy/tree/fuerte-devel](https://github.com/AutonomyLab/ardrone_autonomy/tree/fuerte-devel)

## 2 Controlling Drone

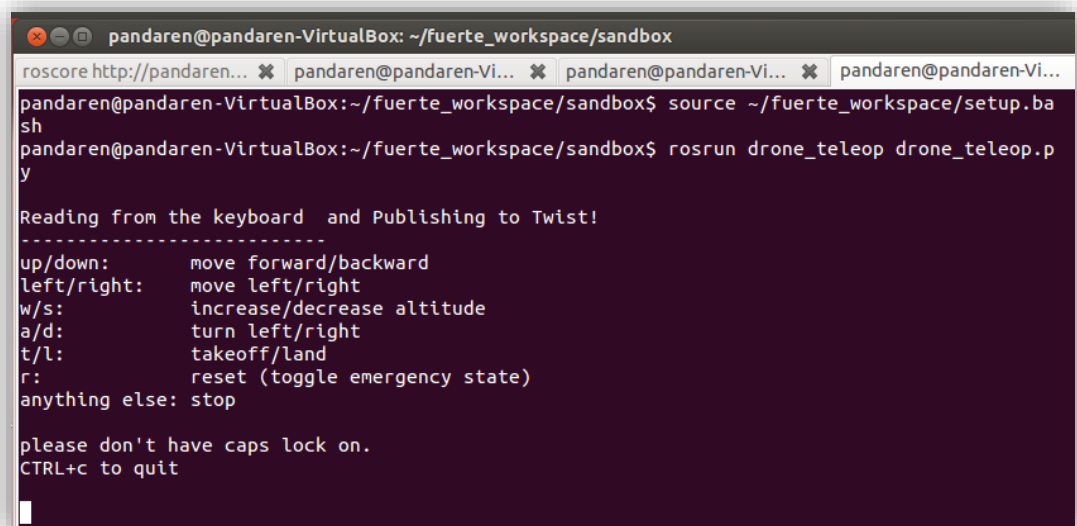
### i. Third Terminal: To Fly the Drone

```
$ source ~/fuerte_workspace/setup.bash  
$ rosrn drone_teleop drone_teleop.py
```

Another terminal is open in new tab. Commands above are inserted into the new tab. Function of commands are explained as below:

1<sup>st</sup> line: Setup working environment.

2<sup>nd</sup> line: Run the drone\_teleop.py inside the drone\_teleop package



```
pandaren@pandaren-VirtualBox: ~/fuerte_workspace/sandbox  
roscore http://pandaren... ✖ pandaren@pandaren-Vi... ✖ pandaren@pandaren-Vi... ✖ pandaren@pandaren-Vi...  
pandaren@pandaren-VirtualBox:~/fuerte_workspace/sandbox$ source ~/fuerte_workspace/setup.ba  
sh  
pandaren@pandaren-VirtualBox:~/fuerte_workspace/sandbox$ rosrn drone_teleop drone_teleop.p  
y  
Reading from the keyboard and Publishing to Twist!  
-----  
up/down:      move forward/backward  
left/right:    move left/right  
w/s:           increase/decrease altitude  
a/d:           turn left/right  
t/l:           takeoff/land  
r:             reset (toggle emergency state)  
anything else: stop  
  
please don't have caps lock on.  
CTRL+c to quit
```

*Figure 7 Third Terminal*

The code drone\_teleop.py in python language contain the algorithm to control the drone using input from keyboard as shown in figure 7. However, the drone can be only be controlled by using keyboard with Capslock off. It is important to keep a finger ready at 'l' for emergency control as it will set the drone to land before it collides with any obstacles or before it goes wild.

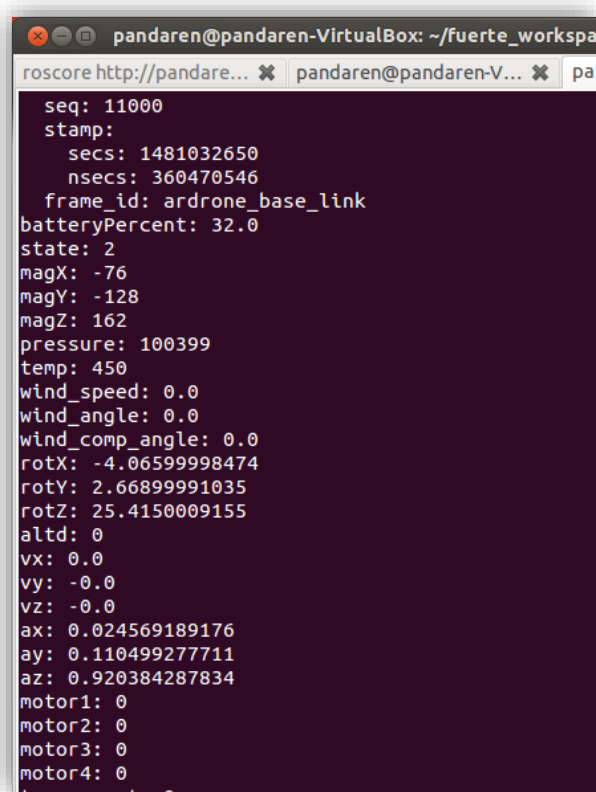
### 3 Retrieving Data from Drone

- i. Forth Terminal: Echo Data from Drone

```
$ source ~/fuerte_workspace/setup.bash  
$ rostopic echo ardrone/navdata
```

1<sup>st</sup> line: Setup working environment.

2<sup>nd</sup> line: Echo the data collected by the drone.



```
seq: 11000  
stamp:  
  secs: 1481032650  
  nsecs: 360470546  
frame_id: ardrone_base_link  
batteryPercent: 32.0  
state: 2  
magX: -76  
magY: -128  
magZ: 162  
pressure: 100399  
temp: 450  
wind_speed: 0.0  
wind_angle: 0.0  
wind_comp_angle: 0.0  
rotX: -4.06599998474  
rotY: 2.66899991035  
rotZ: 25.4150009155  
altd: 0  
vx: 0.0  
vy: -0.0  
vz: -0.0  
ax: 0.024569189176  
ay: 0.110499277711  
az: 0.920384287834  
motor1: 0  
motor2: 0  
motor3: 0  
motor4: 0
```

*Figure 8 Forth Terminal*

The node rostopic plays an important role in displaying all of the data collected by drone through its sensor. Lots of useful information can be obtained here.

‘BatteryPercent’ shows the battery remaining. Please note that the drone is programmed with a safety measure which restrict it from taking off if the battery remaining is less than 15%.

‘State’ is referring to the current state of drone where:

*Table 3 State of Drone*

0	1	2	3	4
Unknown	Initiated	Landed	Flying	Hovering

‘MagX, MagY, MagZ’ refers to the reading of magnetometer which helps in measuring magnetism.

‘Pressure, Temp, Altd’ gives the reading of barometer, thermometer and altimeter installed inside the drone.

‘Wind\_speed, Wind\_angle, Wind\_comp’ estimate the wind speed, the wind angle and wind compensation faced by the drone.

‘RotX, RotY, RotZ’ provides the data on the rotation of the drone about the X-axis, Y-axis and Z-axis respectively.

‘Vx, Vy, Vz, Ax, Ay, Az’ gives details on the velocity and acceleration of the drone about the X-axis, Y-axis and Z-axis accordingly.

‘Motor1, Motor2, Motor3, Motor4’ states the Pulse Width Modulation (PWM) values of the motor.

## 4 Changing Firmware of Drone 2014

- i. Open a telnet session to 192.168.1.1 by using terminal of MAC OS

```
$ echo "1.1.1" > /firmware/version.txt
```

```
$ echo "1.1.1" > /update/version.txt
```

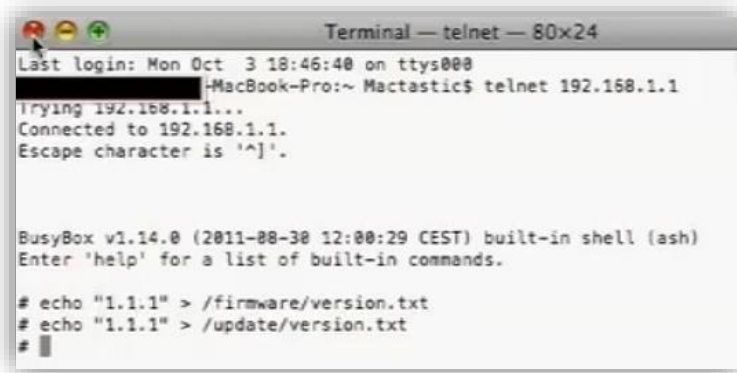


Figure 9 Terminal on MAC OS

- ii. Connect an FTP Client to 192.168.1.1 Port 5551 using Cyberduck

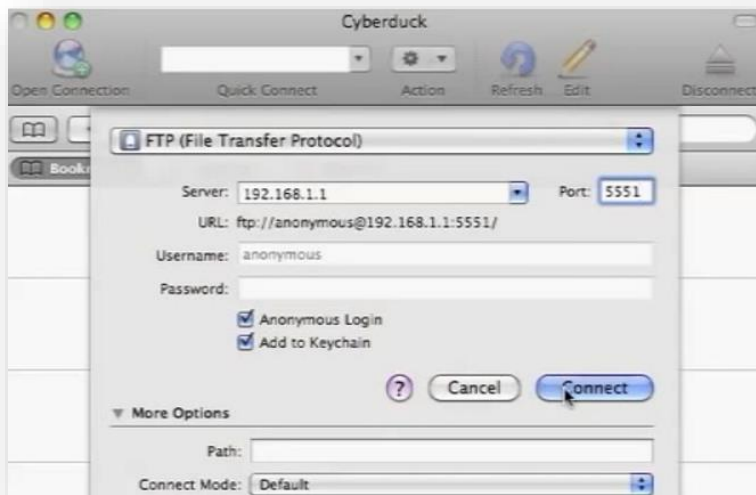
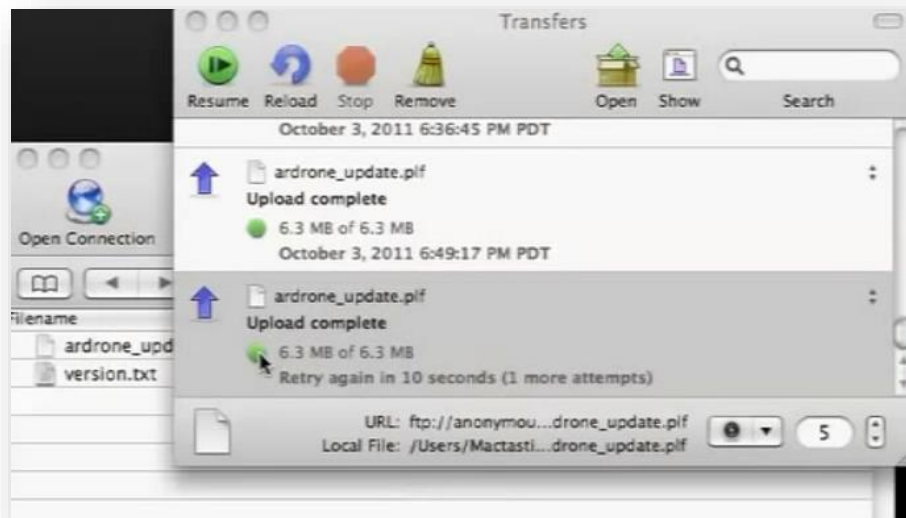


Figure 10 Interface of Cyberduck



- iii. Upload the downloaded plf file to FTP server.



*Figure 11 Uploading plf file to Drone*

Downgrading Drone 2014 edition (firmware 2.4) is required for it to support the coding and connection of Drone 2010 edition (firmware 2.0).

Any File Transfer Protocol (FTP) client will do the job. As for the 'Cyberduck' used as in figure 10 is available for download at <https://cyberduck.io/>

Whereas the plf file used for the firmware change as shown in figure 11 is available at [http://drone-apps.com/firmware/176/ardrone\\_update.plf](http://drone-apps.com/firmware/176/ardrone_update.plf)

MAC OS is used in this scenario as it is more user friendly compared to Ubuntu.

A reboot on drone is required after the upload of the plf file.

## 5 Setting up other Computers

Other than reinstall everything including the Ubuntu and ROS, there is a faster way to do the setup on the other computer which is by exporting appliance.

- i. Choose the **Export Appliance** on Virtual Box or press the shortcut key “Ctrl+E”.

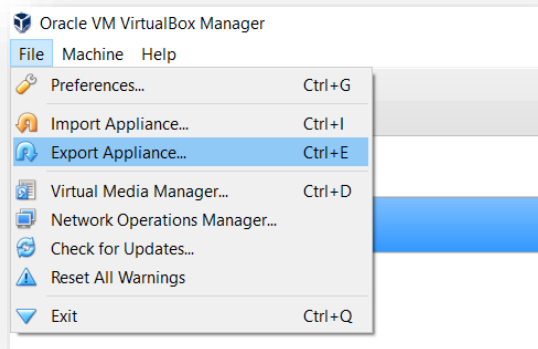


Figure 12 Interface of Virtual Box Manager

- ii. Choose the desired virtual machine and location to export.

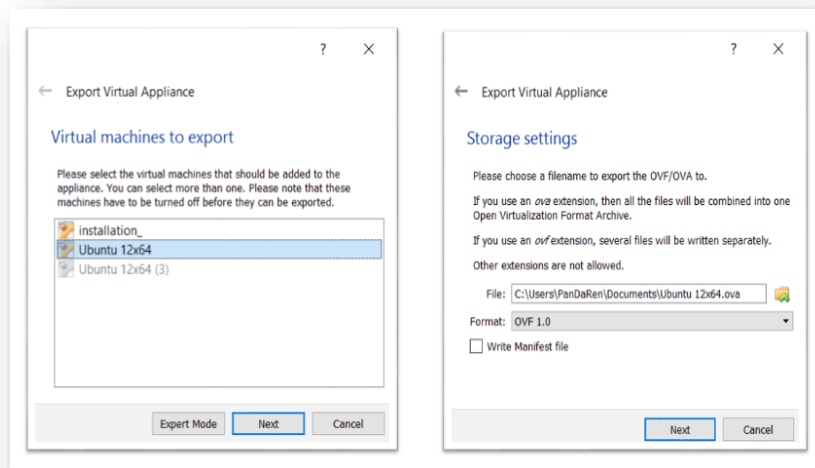


Figure 13 Export Setting

- iii. Output file in .ova format will be available at the chosen location after roughly 10 minutes of waiting.

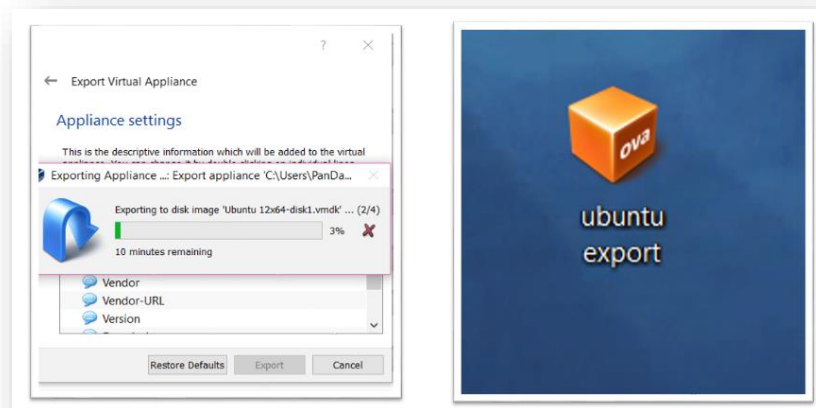


Figure 14 Output of Export

- iv. Choose the **Import Appliance** on Virtual Box of the new computer or press the shortcut key “Ctrl+I” and locate the .ova file exported earlier.

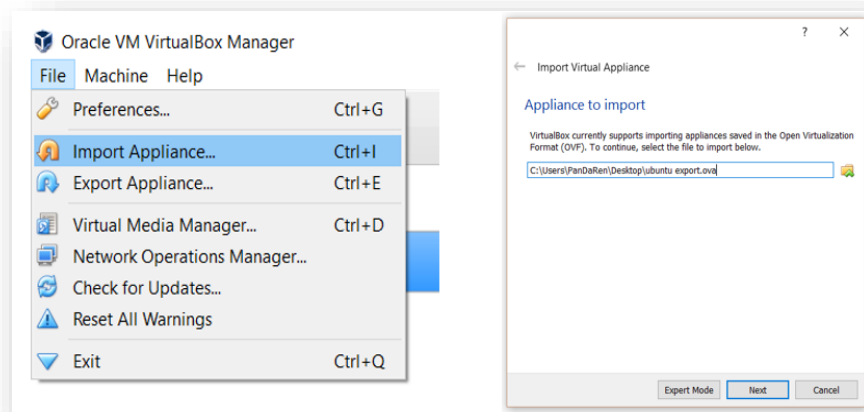


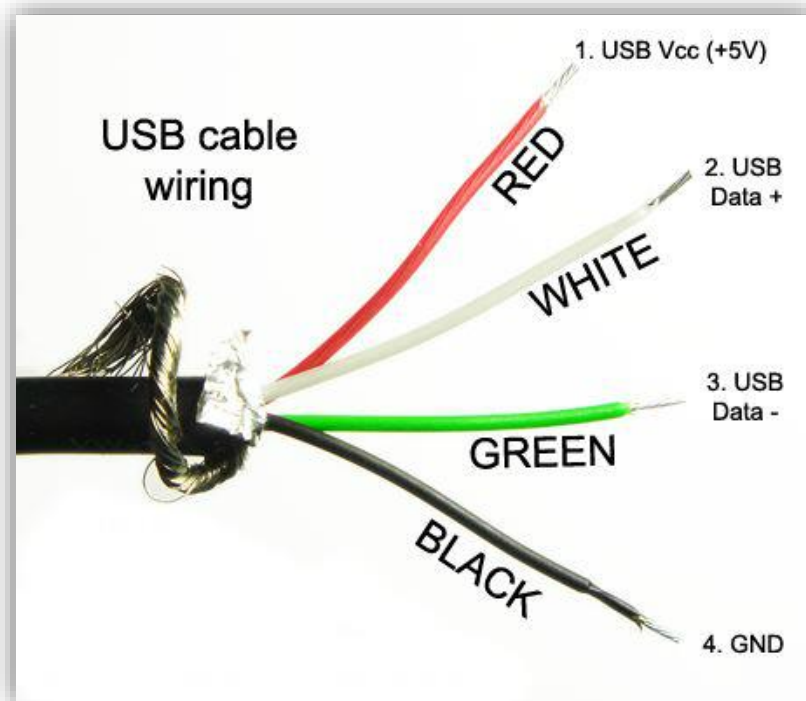
Figure 15 Importing Appliance

## 6 Building Multi-Output-Controller

A multi-output-controller is required in order to provide 2 computers with a similar output simultaneously. This type of controller is not available in the market hence it is my obligation to build one. There are 2 ways in building the controller: single-multiplexer-dual-output controller and dual-multiplexer-dual-output controller.

### 6.1 Single-multiplexer-dual-output controller

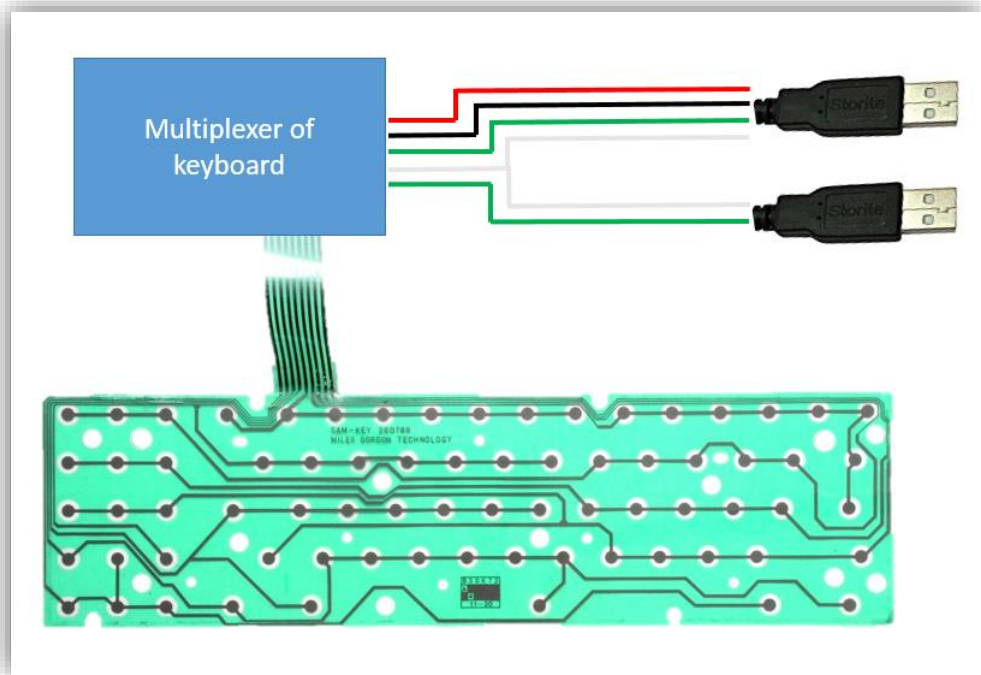
It is easy to build this type of controller as it only requires a keyboard and Universal Serial Bus (USB) male cable.



*Figure 16 Layout of USB Cable Wiring*

By referring to the figure 16 above, the red and black wires are the 5V live and ground respectively. The white wire is the positive data transfer whereas the

green wire is the negative data transfer. The single-multiplexer-dual-output controller can be built by connecting the positive and negative data transfer wire from the multiplexer parallelly to 2 male USB cable. Next, connect 5V and ground wire to one of the USB cable. Then, the multiplexer is connected to the keyboard membrane which allow it to receive input from user. The connection is illustrated in the figure 17 below.



*Figure 17 Connection of Single-multiplexer-dual-output*

The theory behind this connection is to allow the multiplexer to gain power supply from one of the USB cable while transferring data to 2 computers simultaneously.



*Figure 18 Single-multiplexer-dual-output Controller*

The figure 18 above is the controller built based on the theory and logic discussed in previous. The 2 male USB cables act as the input of 2 computers that wished to be controlled.

Although this connection does work, it is not so reliable as the keyboard sometimes will get confused and switching between 2 computers instead of being input of 2 computers simultaneously. Hence, another connection, dual-multiplexer-dual-controller which is having a reliable performance is proposed in the next subtopic.

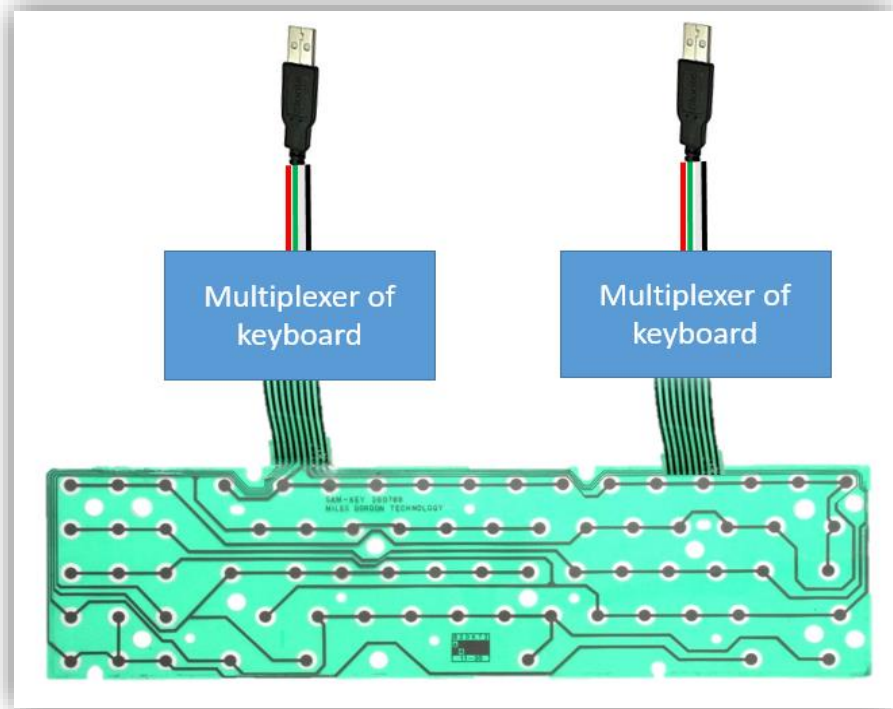
## 6.2 Dual-multiplexer-dual-output controller

Through several tests and troubleshoots on the controller built in figure 18, it is found that a single controller is not able to perform well in serving 2 computers simultaneously for some cases. Hence, 2 multiplexers as shown in the figure 19 below are used instead of 1.



*Figure 19 2 Multiplexers in used*

The dual-multiplexer-dual output controller can be done by simply putting 2 keyboard membrane together while each of the membrane is individually linked to a multiplexer which then connected to a USB male cable. The connection of is illustrated as the figure 20 below.



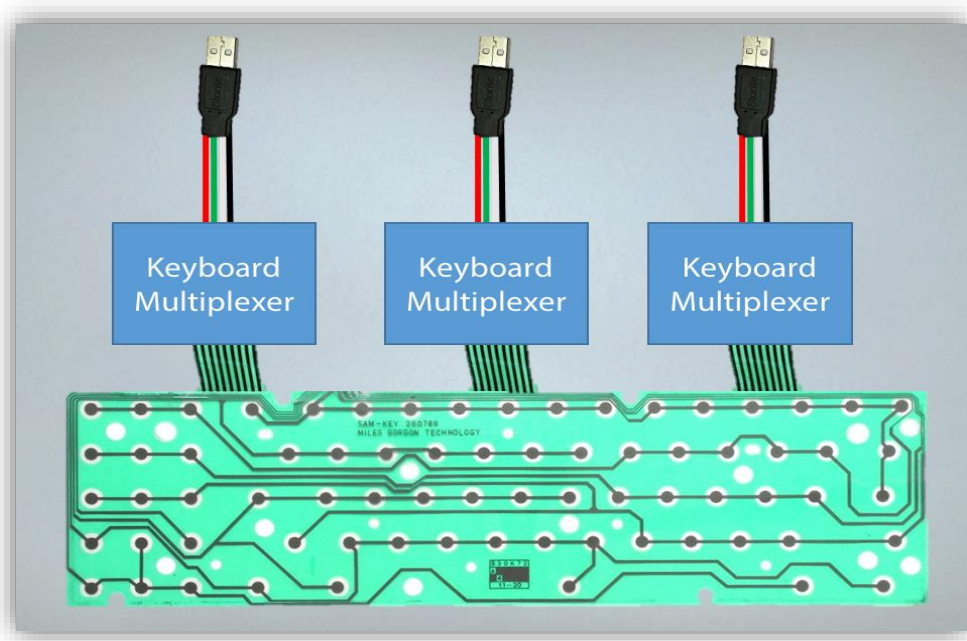
*Figure 20 Connection of Dual-multiplexer-dual-output Controller*

As both of the keyboard membranes are stacked on each other, 2 multiplexers will receive the same input when one key is pressed and hence transferring it to 2 different computers simultaneously. With the situation of each computer is relied on each multiplexer separately, the input to the computer will not go wild at all time. Thus, the performance is more reliable compare to the single-multiplexer-dual-output controller.



### 6.3 Trio-multiplexer-trio-output controller

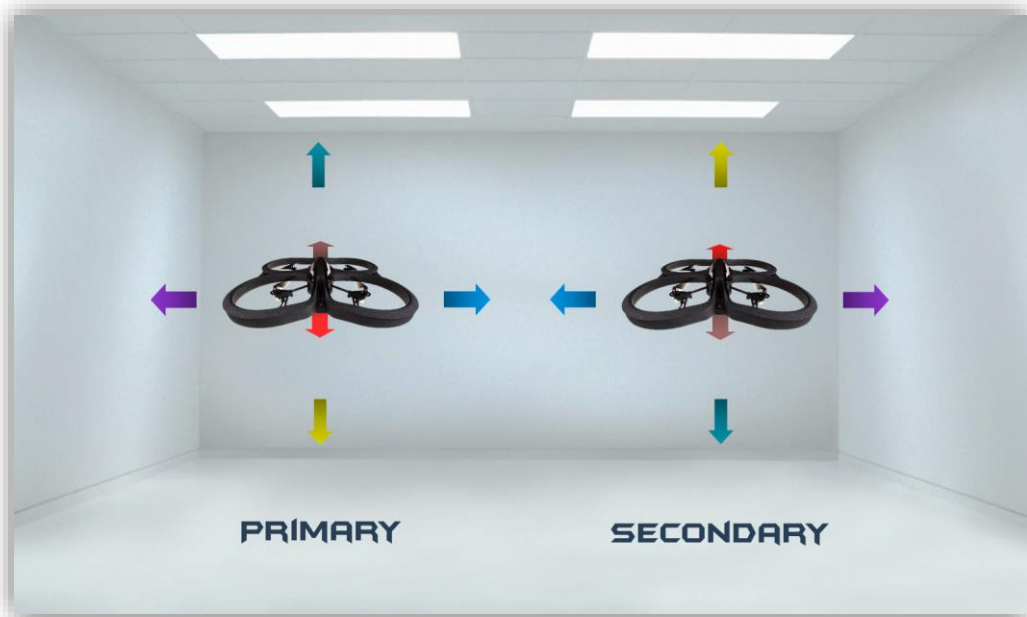
In order to control 3 ROS simultaneously, a trio output controller is required. By using the theory of building a dual-multiplexer-dual-output controller, a trio-multiplexer-trio-output controller can be built by stacking the third keyboard membrane on top which linked to the third multiplexer. The layout of the connection is illustrated in figure 21 below.



*Figure 21 Connection of Trio-multiplexer-trio-output Controller*

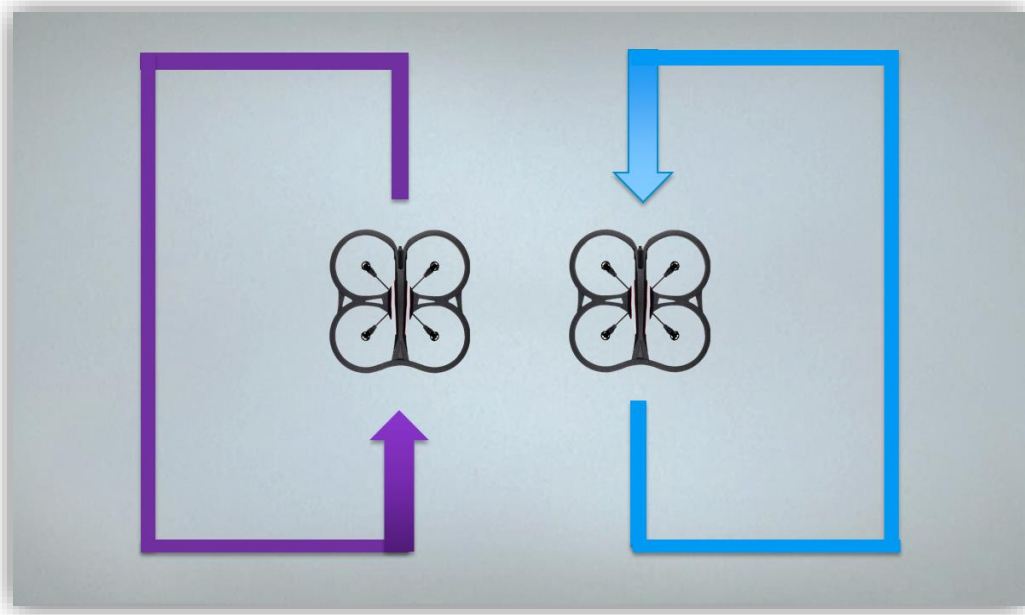
## 7 Dual Drones

Drone Swarm can be defined as controlling multiple drones at once whereas multiple can be any number more than one. In this section, drone swarm will be tested by using 2 drones performing patterned-actions while input is being received. The patterned-action mentioned above can be defined as the secondary drone doing the opposite action as the primary drone which is illustrated as the picture below.



*Figure 22 Flight Movement of Dual Drones*

For example, the secondary will fly to the right (purple arrow) while the primary drone receives the command to fly to the left (purple arrow). Such pattern can be efficiently applied in operations such as search and rescue or surveillance and monitoring which require to cover a large area that start from its center point as shown in figure 23 below.



*Figure 23 Search and Rescue using dual drones*



*Figure 24 Testing on Dual Drones*

Figure 24 above is showing the operation of flying dual drones simultaneously in an indoor condition to eliminate the distraction of wind. Next, the figure 25 below is showing the testing of dual drones flying in an opposite direction of one and another with first drone flying towards the north whereas the other flying towards the south.

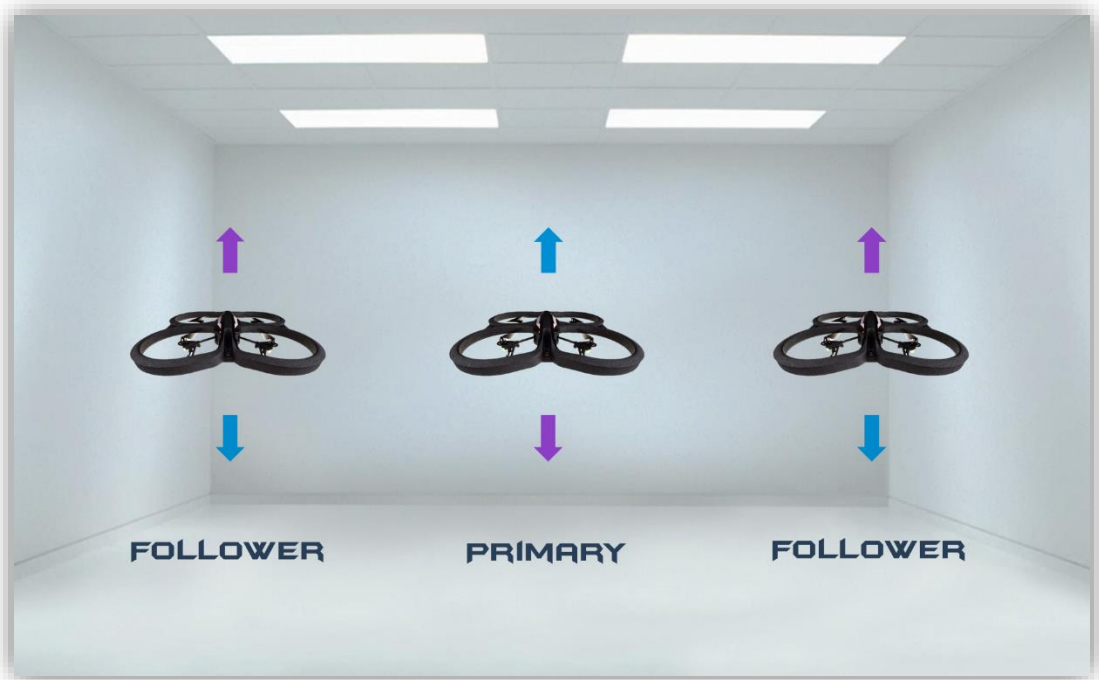


*Figure 25 Dual Drones Flying in an Opposite Direction*

## 8 Trio Drones

Three drones are connected and tested simultaneously for this section. The 2 follower drones will perform certain pre-defined pattern while the primary drone receives command from the user. In order to fly 3 drones simultaneously, the trio-multiplexer-trio-output controller is required to control 3 ROS simultaneously.

### 8.1 Vertical Movement



*Figure 26 Vertical movement of Trio Drones*

The follower drones are set to fly as oppose to the primary drone. As the primary drone is controlled to raise its altitude, the follower drones will lower their altitude. On the other hand, the follower drones will be flying upwards while the primary drone is flying upwards.

## 8.2 Triangular Movement

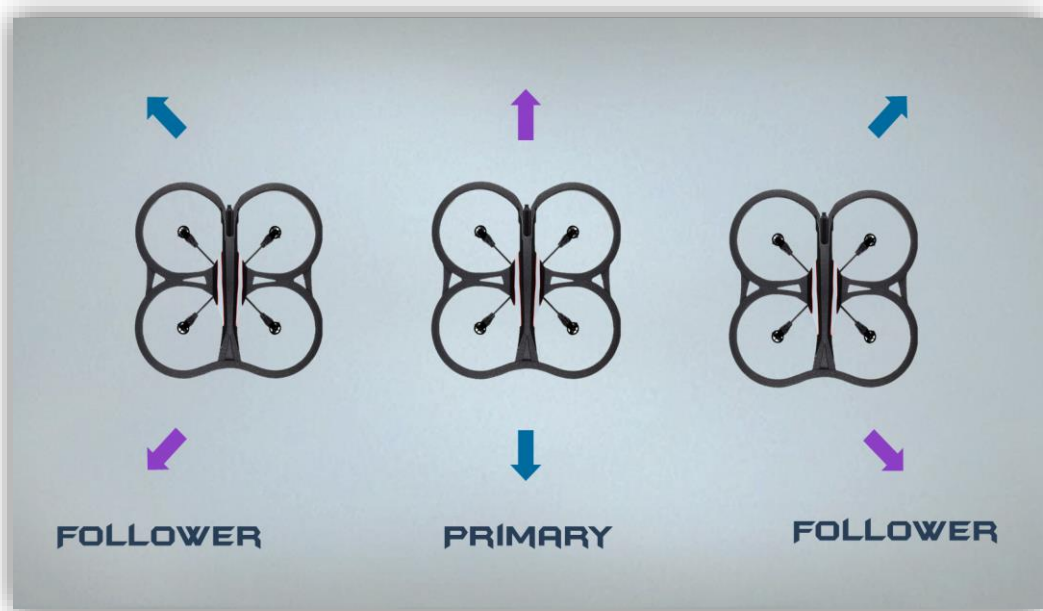


Figure 27 Triangular Movement of Drones

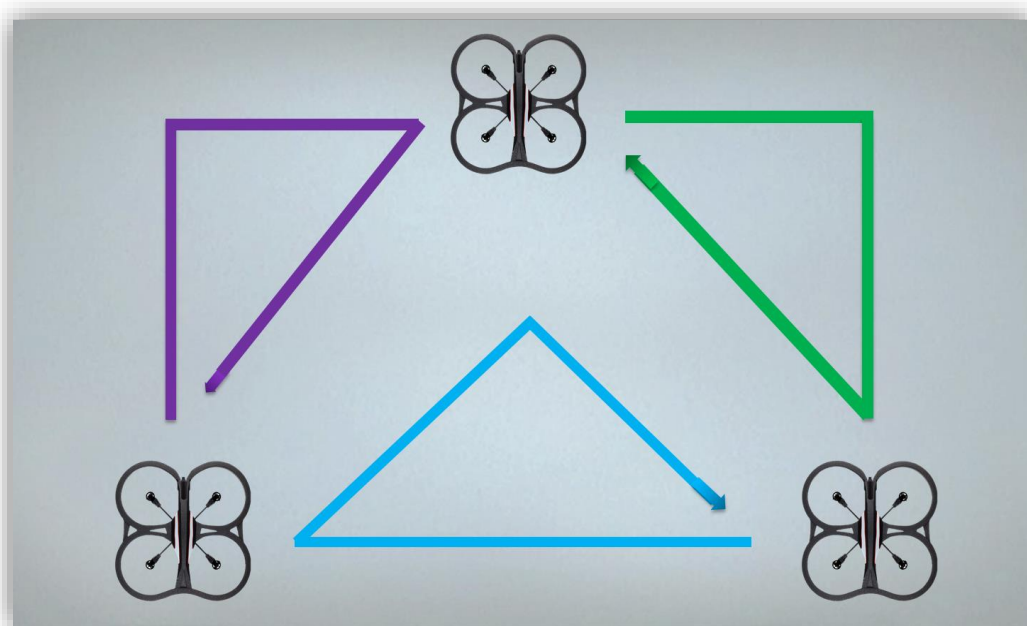
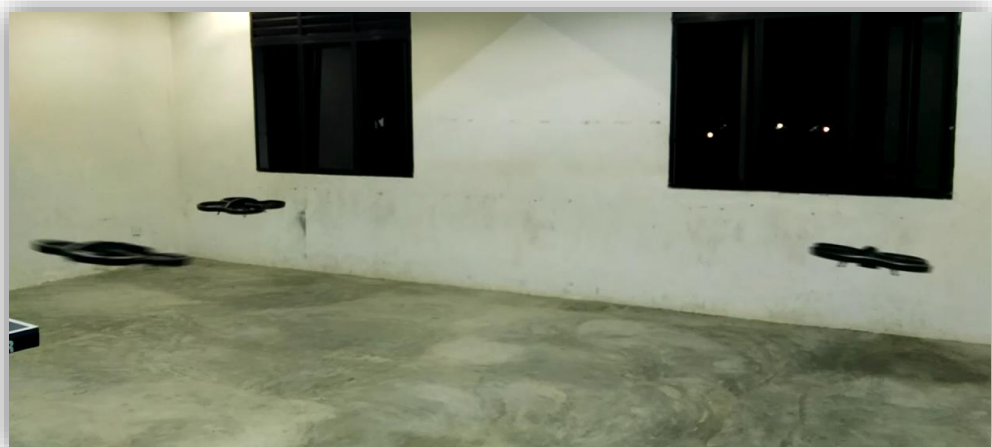


Figure 28 Search and Rescue using Trio Drones

All of the drones are spreading outward in a triangular shape as in figure 27. Then, each of the drone will flies on their own path as shown in figure 28 above which hence allowing them to cover a large area starting from its center point efficiently. Then, the figure 29 below is showing the testing done on trio drones in an indoor condition. Whereas the figure 30 below is showing 3 drones spreading outwards simultaneously in a triangular shape.



*Figure 29 Testing on Trio Drones*



*Figure 30 Trio Drones Spreading Outwards in a Triangular Shape*



## **CHAPTER 5**

### **CONCLUSION AND RECOMENDATION**

In a nutshell, behaviors and responses of multiple AR Drones 2.0 can be customized by running the self-written python code in ROS on the terminal of Ubuntu. Unlike the commonly seen drones swarm that flies synchronously, this project customizes the drones to fly in certain pre-defined pattern such as triangular spreading out. Such patterns will be helpful in operation like search and rescue or surveillance and monitoring which need to cover a large area that start from its center point.

As for recommendation, it will be better if the flying patterns of multiple drones are able to be optimized with the obstacle avoidance features of drones via ultrasonic sensors and cameras. Moreover, controlling multiple drones via single ROS instead of multiple ROS will be more convenient. However, the 8-months timeframe does limit the development of this project. Hence, this project will be passed to someone who are interested in it for further development as recommended.



## REFERENCES

- [1] A. Cavoukian, *Privacy and drones: Unmanned aerial vehicles*: Information and Privacy Commissioner of Ontario, Canada Ontario, Canada, 2012.
- [2] M. R. Smith and L. Marx, *Does technology drive history?: The dilemma of technological determinism*: Mit Press, 1994.
- [3] G. Faure and T. M. Mensing, "The urge to explore," in *Introduction to Planetary Science*, ed: Springer, 2007, pp. 1-12.
- [4] B. Theys, G. Dimitriadis, P. Hendrick, and J. De Schutter, "Experimental and Numerical Study of Mini-UAV Propeller Performance in Oblique Flow," *Journal of Aircraft*.
- [5] H. Eugster and S. Nebiker, "UAV-based augmented monitoring-real-time georeferencing and integration of video imagery with virtual globes," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 37, 2008.
- [6] P. C. Ulrich, "Why Obama needs drones," 2013.
- [7] A. Colozza and J. L. Dolce, "High-altitude, long-endurance airships for coastal surveillance," *NASA Technical Report, NASA/TM-2005-213427*, 2005.
- [8] J. Irizarry and E. N. Johnson, "Feasibility study to determine the economic and operational benefits of utilizing unmanned aerial vehicles (UAVs)," 2014.
- [9] A. Colozza and J. L. Dolce, "High-altitude, long-endurance airships for coastal surveillance," *NASA Technical Report, NASA/TM-2005-213427*, 2005.
- [10] E. S. Grood and W. J. Suntay, "A joint coordinate system for the clinical description of three-dimensional motions: application to the knee," *Journal of biomechanical engineering*, vol. 105, pp. 136-144, 1983.

- [11] T. Krajník, V. Vonásek, D. Fišer, and J. Faigl, "AR-drone as a platform for robotic research and education," in *International Conference on Research and Education in Robotics*, 2011, pp. 172-186.
- [12] M. Mohd Iqbal, "Control of a quadcopter," 2016.
- [13] K. Boudjit and C. Larbes, "Control and stabilization applied to micro quadrotor AR. Drone," in *Proceedings of the 3rd International Conference on Application and Theory of Automation in Command and Control Systems*, 2013, pp. 122-127.