Abstract Command to Control

Parrot AR. Drone 2.0

BY

FU QUONG TING

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF INFORMATION TECHNOLOGY (HONS) COMMUNICATIONS AND NETWORKING

Faculty of Information and Communication Technology

(Perak Campus)

MAY 2015

DECLARATION OF ORIGINALITY

I declare that this report entitled "**Abstract Command to Control Parrot AR Drone 2.0**" is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature : ____________________________

Name : ____________________________

Date : ____________________________

ACKNOWLEDGEMENT

I would like to express my very great sincere and appreciation to my project supervisor, Dr. Goh Hock Guan for his valuable advice and constructive suggestion throughout this final year project. He is willing to provide guidance as well as give me motivation from time to time. His willingness to offer assistance is generously appreciated.

I would also like to express my gratefulness to all of my course mates who are actively shares their idea and valuable feedback in this project.

Lastly, my thankfulness goes to my beloved family and friends. They supported me both emotionally and financially. It was really heartwarming to attain family love and continuous encouragement throughout accomplishment of this project. This project would not have been possible without them.

ABSTRACT

This paper introduces and discovers the possibilities of “Navigation and Control” technology that are embedded in a well commercialized mini-sized Unmanned Aerial Vehicle (UAV) – the Parrot AR. Drone 2.0 by abstract control module.

In general, this unmanned aerial flying device relays heavily on the following list of state-of-the-art sensors to fly:

(i) A 3-axis accelerometer with +/-50 mg precision

(ii) A 3-axis Gyroscope 2000 °per second precision capabilities

(iii) A 3-axis Magnetometer with precision up to 6 °

(iv) A Pressure Sensor with +/- 10 Pa precision

(v) A 60 FPS vertical armed QGVA Sensor type camera for measure ground speed

(vi) Ultrasound Sensors aboard for measure ground altitude

Parrot AR. Drone 2.0 is an interesting flying machine. It is an ideal and affordable toy for helicopter or quadcopter enthusiast to fly with. This inexpensive flying drone is often cost around $300. Parrot AR. Drone 2.0 is a well designed indoor navigation systems that is combines several low-cost inertial sensors, sonar, computer vision techniques, and aerodynamics models.

There are three existing systems have been reviewed in this project. In order to carry out this project, V-model as system development life cycle has been selected as a project guideline. As mentioned above, one of the technologies that are involved in this project is the Parrot AR. Drone 2.0. Besides that, personal computer (laptop) was used as well in this project to develop indoor flight autonomous system. The proposed system is written in Java programming language.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbreviation	**Meaning**
API	Application Programming Interface
BLP	Bi-Level Programming
DVD	Digital Video Disc
FPS	Frame per Second
GPS	Global Positioning System
GUI	Graphical User Interface
HUD	Head-Up Display
IMU	Inertial Measurement Unit
MG	Micro-G
MAV	Micro Unmanned Aerial Vehicle
PA	Pascal
PC	Personal Computer
PTAM	Parallel Tracking And Mapping
QGVA	Quality Graphics Video Accelerator
RF	Radio-frequency
SDLC	Software Development Life Cycle
SLAM	Simultaneous Localization And Mapping
UAV	Unmanned Armed Vehicle

Chapter 1 – Introduction

1.1 Problem Statement and Motivation

There are two major problem statements that have been successfully identified. The problem statements are presented in this section of the report. These issues are the main driving force and motivation to develop a new approach for flying AR. Drone 2.0 in indoor environment.

One of the most critical problem issues is GPS module cannot be well performed in indoor environment. Since GPS service is limited to outdoor activities and environment, hence the drone cannot fly autonomously within indoor environment. This is because GPS module cannot identify the GPS location and coordinate without the GPS service in indoor environment. Therefore, flying autonomously with GPS module onboard Parrot AR. Drone seems to be pointless and not workable in indoor environment.

Conventional flying device technology requires constant human supervision which is a very exhausting and time-consuming process for flyers. In order to solve this issue, the drone should be able to fly on itself by inputting some important abstract commands before flight. Simple abstract commands such as take off, landing, flying up and flying down can be send to drone to performing autonomous flight within indoor environment.

1.2 Project Objectives

This final year project improves the existing Parrot AR. Drone 2.0 to have a better flying experience. There are some objectives that need to be pre-defined prior to develop the application. These are:

- To study and compare existing flying technology

Commonly, remote-control flying electronic devices such as airplane and helicopter pilots controls their device wirelessly from a short distance using a 2.4 GHz radio controller. A 2.4 GHz radio controller is a convenience device for pilots so that they can operate their flying drones easily.

As flying technology evolve from time to time, now the advancement of remote control enable pilots to fly their flying drone via different methods. Most of the flying drones can be controlled and fly via Wi-Fi connection nowadays. This breakthrough allows drone developers to develop flight systems where the flying drone can be piloted with ease by using any Wi-Fi features enabled devices such as smart phone and PC.

In this report, we study the technique for controlling and flying Parrot AR. Drone 2.0 autonomously. However, Parrot AR. Drone 2.0 greatly depends on GPS module at this moment. The drone deploys and tests fly in indoor environment. The GPS module is not accurate in terms of identifying the drone’s location. Notably, the GPS module is not functional due to GPS limitation. Therefore, a series of studies needed to be done in identifying and selecting a better approach to fly AR. Drone in indoor environment. In addition, the drone should be able to fly and avoid any frontal obstacles along its flying path to a selected destination.

- <u>To introduce Parrot AR. Drone 2.0 by flying in Autonomous Navigation mode (Without GPS module and GPS services)</u>

Currently, AR. Drone 2.0 is capable of flying autonomously with the GPS module onboard. However, a single piece of GPS module might be a burden for flyer since it is expensive (approximately RM462.47 as of 18 Jan 2015 per module). This is definitely not a good news for flyers since they might only perform some simple flying tasks. On-board sensors – frontal camera and the vertically mounted camera can be fully utilized for address this problem if flyer flies the drone in indoor environment. The drone should be able to react independently to avoid hitting any environmental obstacles.

The drone learns from its environment since there is no assumptions can be made of the surrounding environment. Therefore, the flying drone has to have the ability of knowing its position and movement in the environment to operate automatically explore unknown environment as well as to avoid any frontal obstacles. Simultaneous Localization and Mapping (SLAM) technique as the process of incremental mapping construction and localization is adopted to get information about the surrounding environment.

- <u>To enable the drone flying in Autonomous Navigation Mode to reach any selected destination</u>

In order to communicate and control Parrot AR. Drone 2.0, there are four main communication services need to implement in this project. The communication with AR. Drone is handle and done via abstract AT commands. The abstract AT commands are issues and sent to the AR. Drone in UDP or TCP packets format.

We control and configure the drone by sending abstract AT commands on regular basis. The navigation data (navdata) which consists of information about the drone is sent by the AR. Drone to client at appropriate time-lapse interval (in times per second).

Client receives video streams from AR. Drone. These images from the video stream are decoded for further collision avoidance detection purpose at client PC. Feedback from the Collision Avoidance System is used to select suitable commands for the flying drone.

Now, the drone should be able to fly to any selected destination which is set up by the flyer. This is an important step for setting up any necessary parameters for drone in order to reach the destination. Upon completion of setup the necessary parameters, the drone know its destination and fly toward the selected destination without hitting any frontal obstacles.

1.3 Project Scope

There are five main steps required to implement this project. These are:

1. Enable Wi-Fi access connection from PC to AR. Drone 2.0
2. Enhance existing flight control of AR. Drone
3. Develop abstract command to AR. Drone
4. Obstacle Avoidance System
5. Carry out evaluation to test the purpose solution

At first, this project starts with enabling the Wi-Fi access connection from PC to AR. Drone 2.0. By default, when the drone switches on, the flying drone automatically creates an ad-hoc Wi-Fi hotspot. Electronic controlling device such as Smartphone or PC can connect to the drone's Wi-Fi and communicates with the drone. This implies a major security issue of the AR. Drone. Hacker can easily gain unauthorized access to AR. Drone as they use an open network infrastructure. This security flaw make possible of taking over someone else's flying drone illegally and perform some malicious activities. In order to fix this vulnerability, Wi-Fi protected access (WPA) and pairing function can be implement so that these functions protects the drone.
Upon completion of establish Wi-Fi connection to the drone, the PC should be able to communicate with the AR. Drone 2.0 wirelessly. Some simple abstract commands should be able sent from PC Wi-Fi to the drone for further instruction processing.

After successfully establishing connection to AR. Drone, the next critical problem is to fly the drone in a more stable way. By default, the drone might be unstable when it starts to take-off for the first time. It flies left to right and versa. This glitch should be taking into consideration since it might affect the overall flight performance. Flight control enhancement in the aspect of controlling drone wirelessly needed to be carried out in order to ensure that the drone would fly more stable.

Next, all of the abstract commands that are required to send over via PC Wi-Fi module and communicate with the flying drone are develop and implement here. Simple commands such as take off, landing, recording the flight, and screen capture should be able to send over to drone. The drone executes the abstract commands.

Next, problem needed to be solved is how to acquire and fetch real-time video streaming from the Parrot AR Drone 2.0 to application. This module is very important in this project. This is because only when the real-time live streaming module is working, then the flying drone can process the real-time streaming video.

After getting “live data”, that is, live streaming video from on-board cameras, the application at client side (PC) should be able to perform a series of algorithms to compute an absolute estimation of the drone’s position whether it might be on the course of hitting any obstacle. Some people called this module as “AR. Drone obstacle avoidance system”. The drone should now have the ability to detect and avoid any frontal obstacles. Pilots are capable to control and navigate the Parrot AR. Drone in an unknown environment using only onboard sensors.

Finally, a series of evaluations on this proposed solution needs to be carried out from time to time. This is a very important stage as it verifies the propose solution is working as it intended to be and flying correctly according to the command that issues by drone flyer via PC Wi-Fi module.

1.4 Main Contributions from the Project

Upon accomplishment of this final year project, flyers no longer flying their drone in conventional method where they just need to switch on PC, establish Wi-Fi connection to AR. Drone 2.0 and send abstract commands to control the drone. This is a new and efficient approach for flyers since they can access into their AR.Drone wirelessly at any time and the video output is directly streams into PC.

Unlike other flying control application such as ARFreeFlight 2.4 which is developed by Parrot SA, this final year project allows live camera viewing on their PC which is much more convenience since they can watch video streaming lively from the drone into their computer screen that displays more content on the screen.

Flyer can issues some simple abstract commands such as take off, landing and hover flying to their drone via PC. The drone receives commands and executes these flying operations accordingly.

1.5 Organization of the report

This section is intended for report readers and lecturers in order to provide them useful information as well as serves as reading guidance about this report.

The information in this final year project report is organized as follows:

Chapter	Description
Chapter 1, "Introduction"	Provides an overall introduction for readers who are relatively new to the AR. Drone product. This chapter includes driving force and motivation to develop a new approach on controlling AR. Drone 2.0 wirelessly via PC. At the same time, project objectives and project scope is well-defined and listed out in this Chapter. Lastly, expected contributions from the project also highlighted.
Chapter 2, "Literature Review"	Describes reviews on the Technologies which includes Hardware Platform, Programming Language and Algorithm. Reviews and comparisons on existing systems/applications also discussed in this section.
Chapter 3, "System Methodology"	Lists out 4 tropical types of System Development Models. Discuss the reasons why I select that System Development Model. Describe the Hardware and Software that

	is implements in this project. Defines project milestones and estimated cost for this project.
Chapter 4, “System Design”	Describe how the project works in the system. Readers will know the system architecture, functional modules in the system and how the system flows. Simple Graphic User Interface (GUI) shown in this section too.
Chapter 5, “System Implementation”	Describe on how to setup hardware and software of this project. Setting and configuration is discussed as well. Step by step on how to use the system (System Operation) with screenshots.
Chapter 6, “System Evaluation and Discussion”	Project system testing and performance metrics are listed out. Testing Setup and Result are examined. Project challenges and objectives evaluation are discussed as well.
Chapter 7, “Conclusion and Recommendation”	Discuss the outcome of project. Further possible improvements are proposed here.

Table 1-5-1: Organization of report

2.1 Review of the Technologies

Unmanned Aerial Vehicles (UAV) can perform many valuable tasks. They are being used for incredible activities. Some of the most popular emerging applications include conservations and wildlife measurement, agriculture and aerial photography.

Drone with cameras are used to track wildlife poachers in Africa. Some Japanese farmers fly their drones to manage rice paddies from UAVs that able to monitor crops and spray pesticides.

In addition, some of the government departments such as police forces and fire services are looking into the possibilities of deploying the drone in their daily services. Drones have become an appealing flying machine for them due to its small size and portability.

A flying drone can be used for aerial photography at the scene of serious or deadly accidents. This is also another good approach to manage traffic at events of traffic congestion.

In Chapter 2, we introduce overall literature review on the flying quadcopter, Parrot AR. Drone 2.0, which is used in this project. In addition, we also further looking into the details of Parrot AR. Drone 2.0 capabilities and its available sensors.

The following sub-chapters explain the detail techniques and existing systems that have been studies in our approach: In Chapter 2.1.1, discuss the hardware platform used. In Chapter 2.1.2, present the choice of programming language that will implement in the system. The communication interfaces as well as the SDK are briefly summarized here. In Chapter 2.1.3, all of the algorithms, in particular from the aspect of Anti-collision System are presented. In Chapter 2.1.4, summarized all of the technologies that have been reviewed in this project. From Chapter 2.2 to Chapter 2.2.4, evaluate the existing Collision Avoidance Systems, which are available for flying electronic drone.

2.1.1 Hardware Platform

Many people are quite familiar with these terms, for examples, "Phantom, Dronie, Drone, Quadcopter, UAV" in this modern era. These terms are widely used these days along with the increasing enjoyment in recent months. There are a number of electronic flying drones that are available in the market today.

In despite of the variation models, we discuss some of the most suitable flying drones in this project.

The first flying drone that captures a great attention from people is **Phenox**.

Figure 2-1-1: Phenox micro-drone

Phenox flying drone is designed and developed by Phenox Lab. This is a unique new Kickstart project from Tokyo. Phenox is an interactive, intelligent and programmable drone for flyer.

Phenox Lab claims that the Phenox drone is an intelligent flying drone because it does not requires external controller to fly it. In addition, Phenox also has build-in one microphone and two cameras to recognize the pilot. These sub-electronic devices enable interactive between pilot and the drone. Phenox flying drone is programmable since it runs on Linux-based system. Pilot of a drone can write their own program to fly their drone in more interesting flight.

This intelligent micro-drone knows on how to fly and responds to specific spoken commands. It also recognizes whistle sounds and hand gestures. This autonomous mini-sized flying drone is extremely light with the payload of 25 gram that allows taking off and landing on a person's hand. The on-board intelligence electronic device, SoC's ARM Cortex-A9 MP Core processor, is supplied by Xilinx Zyng SoC Company.

A microphone, two on-broad video cameras, an inertial measurement unit (IMU) as well as a range sensor are mounted on Phenox drone to monitor its surrounding environment. The drone is able to respond accordingly based on the input from its sub-devices.

Phenox Lab develops computer-vision apps by using OpenCV libraries. Open source Julius speech-recognition engine for developing voice-recognition and whistle- or sound-control apps is used as well.

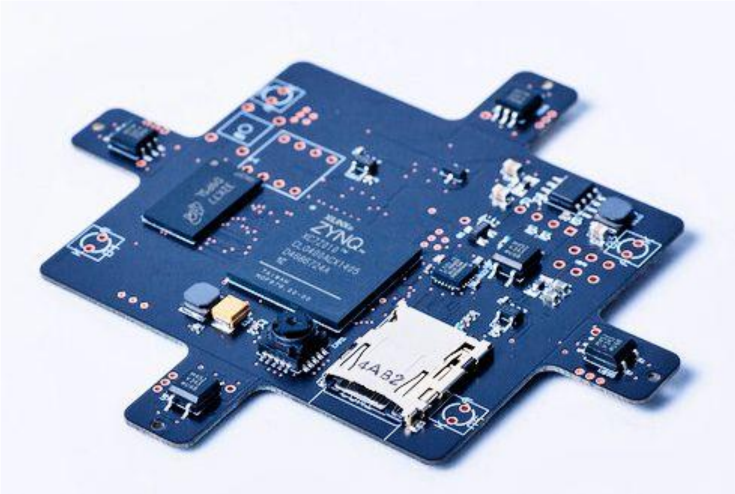

Figure 2-1-2: Control board of Phenox micro-drone

A control board of Phenox micro-drone equipped with SoC's ARM Cortex-A9 MP Core processor, an SD card socket, 256Mbytes of DDR3 SDRAM, vertical downward-facing camera, IMU, and power devices.

Pilot can fly and control Phenox micro-drone by using bluetooth connection.

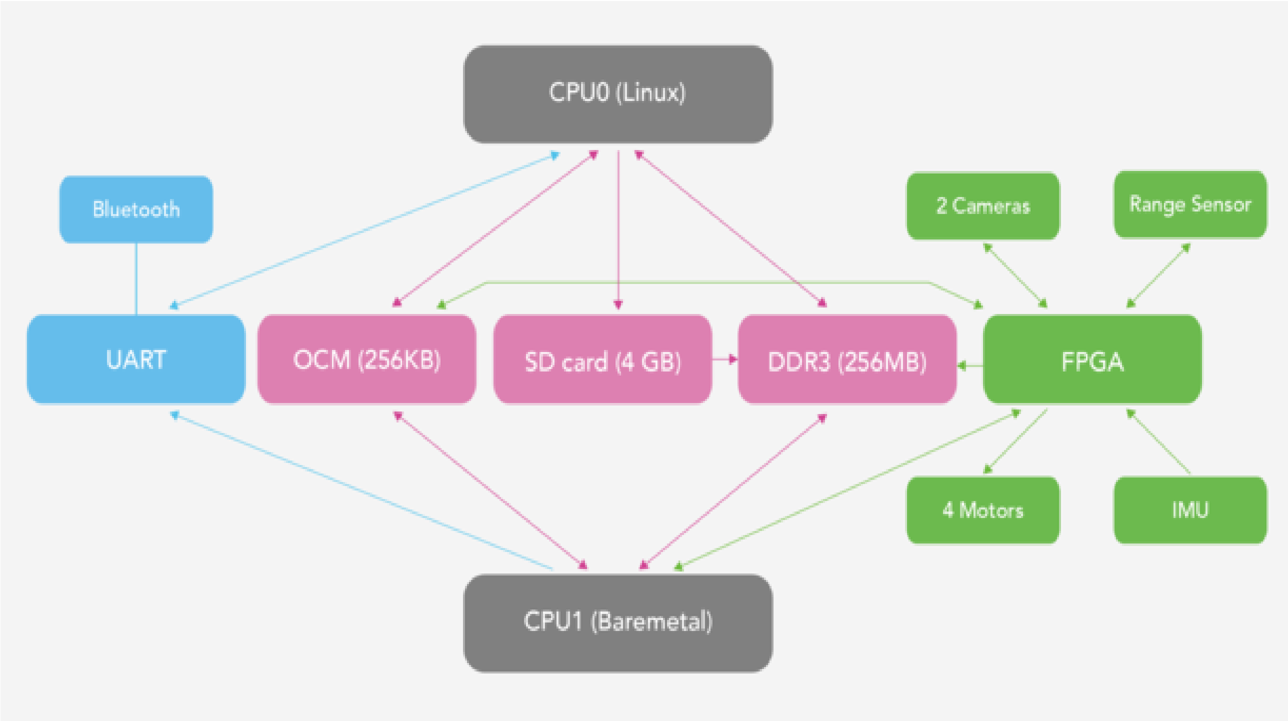

Figure 2-1-3: System Structure of Phenox micro-drone

Next, another flying drone that offer good capabilities and functions to pilots is **3DR IRIS^{+}**.

Figure 2-1-4: 3DR IRIS^{+} quadcopter

The 3DR IRIS^{+} quadcopter is redefined by developer as a good shooting photos and video flying device from incredible new perspectives. IRIS^{+} is an autonomous flying drone that is equipped with full GPS-guided autonomous capabilities hence it could fly itself to any specific location while keeping its camera steadily with two-axis camera gimbals.

Pilot can fly the 3DR IRIS^{+} quadcopter by using a nine-channel radio control transmitter or a Smartphone that runs on Android platform. Flight planning of 3DR IRIS^{+} quadcopter is made easier for pilot since they can draw and fly it with a single touch on well-designed 3DR IRIS + quadcopter tablet-based mobile application.

It is a perfect flying drone for personal aerial photographer since it is armed with several of state-of-the-art electronic systems such as advanced Pixhawk autopilot system, battery charger, high-capacity flight battery, ground station radio, and a remote controller.

The 3DR IRIS$^+$ quadcopter uses the latest ST Microelectronics sensors and runs on a real-time operating system. Its high-performance 32-bit autopilot system is fully supported by an ARM Cortex-M4 processor.

These are the most eye-catching features that are offers by IRIS$^+$:

1. Real-time mission monitoring, data-logging and control is made possible by built-in data radio
2. Simple point-and-click programming and configuration is available on their powerful cross-platform ground station and analysis software. They run on different operating systems which include Windows, OS X and Linux.
3. Intuitive control method, "draw a path to fly", make mission planning easily
4. Camera options : A live video link with programmable on-screen-display
5. Flight plans for mapping, scientific research and other applications can be programmed before flight into GPS waypoints which allows for professional missions.
6. Press a button for auto take off and landing along with Return-To-Launch point command that could be runs under failsafe conditions
7. Geo-Fencing feature offer a virtual box to keep the drone within a suitable space
8. Simple firmware updates are available periodically since it develops with open source code. Ground station software is free distributed under standard open source licenses. IRIS capabilities are always expanding and enhancing.

Lastly, we discuss about **Spiri**.

Spiri is yet another programmable and autonomous flying robot. It is a flexible and versatile airborne Linux-based device. Spiri is developed as a well-balanced and sensitive physical machine. The ultimate goal of Spiri is finding a way to bring people together as one of the autonomous and social creature.

Figure 2-1-5: Spiri quadrotor drone

With Spiri, pilot can interacts with the flying drone by talking to it via a smart phone. Play music to it, blows whistle and other list of pre-defined commands. It could make autonomous flight even more interesting than before.

There are pairs of high resolution of video cameras onboard this quadrotor drone. Spiri can record videos in 3D and use stereoscopic effect for obstacles detection. It can alter and fly its own flight path autonomously.

The Spiri main electronic chip is runs on Ubuntu Linux with Robot Operating System (ROS) which allow running variety of programs using 1GHz dual-core Freescale iMX6 DualLite processor. It has 4GB flash memory attached along with three-axis accelerometer.

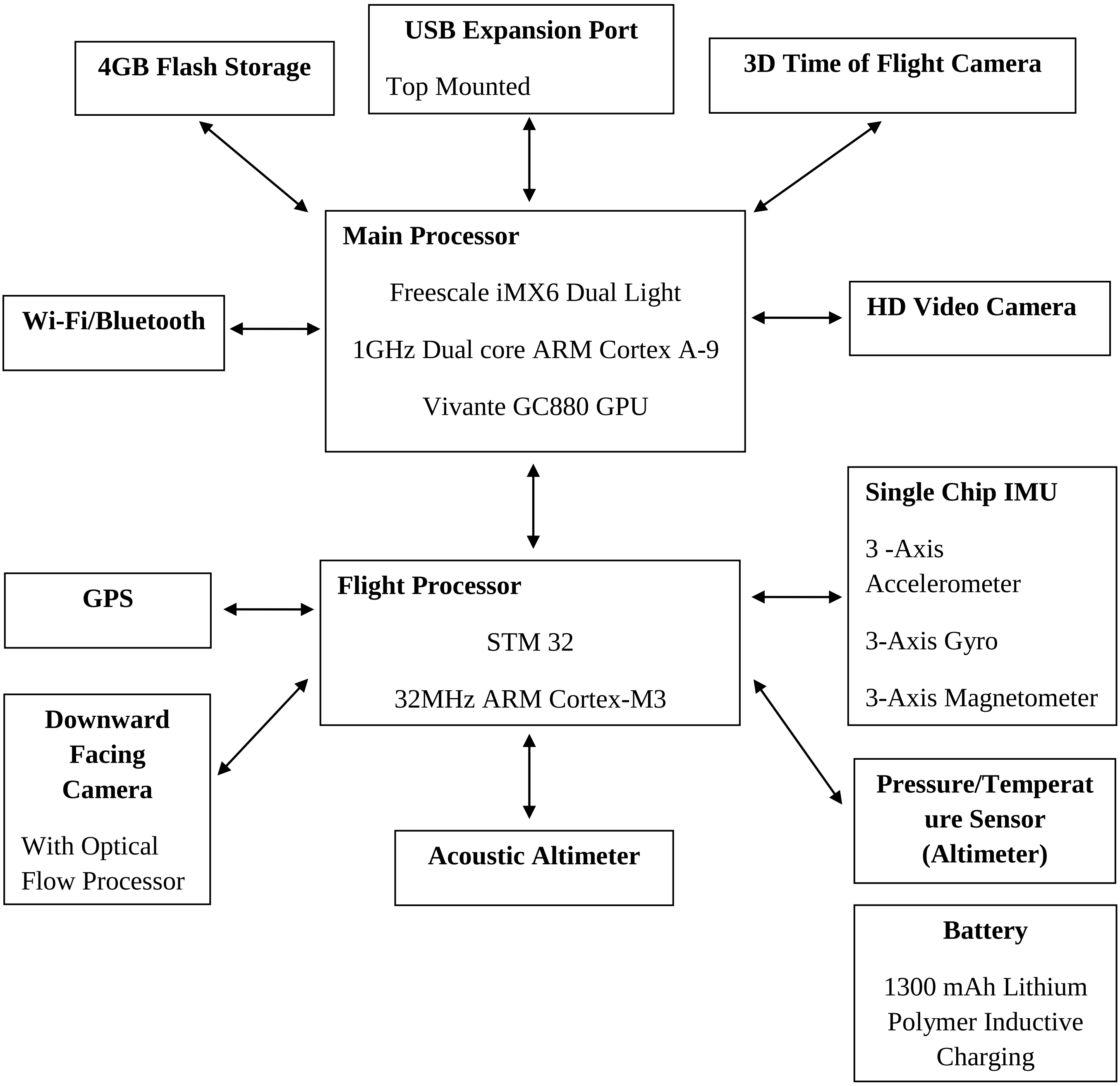

Figure 2-1-6: System Structure of Spiri drone

Wi-Fi or Bluetooth connection is used to connect Spiri drone. Spiri allows remote connect via different Wi-Fi network. When Spiri fly out of reachable signal area, it will execute scripts from its on-board electronic chip. At the same time, when Spiri is running low in battery, it will land on its inductive perch to recharge battery. In other words, Spiri is able to take care of itself during those critical situations. Since Spiri is free from needing constant human-manning controls, pilot can launch several Spiri drones at the same time.

Proper documentation about Spiri, API and libraries of flight commands for programmers are available online throught GitHub. The libraries to access Spiri's sensors and other electronic hardware are written in C++. Other programming languages such as Java, Node.JS and Python can be used as well.

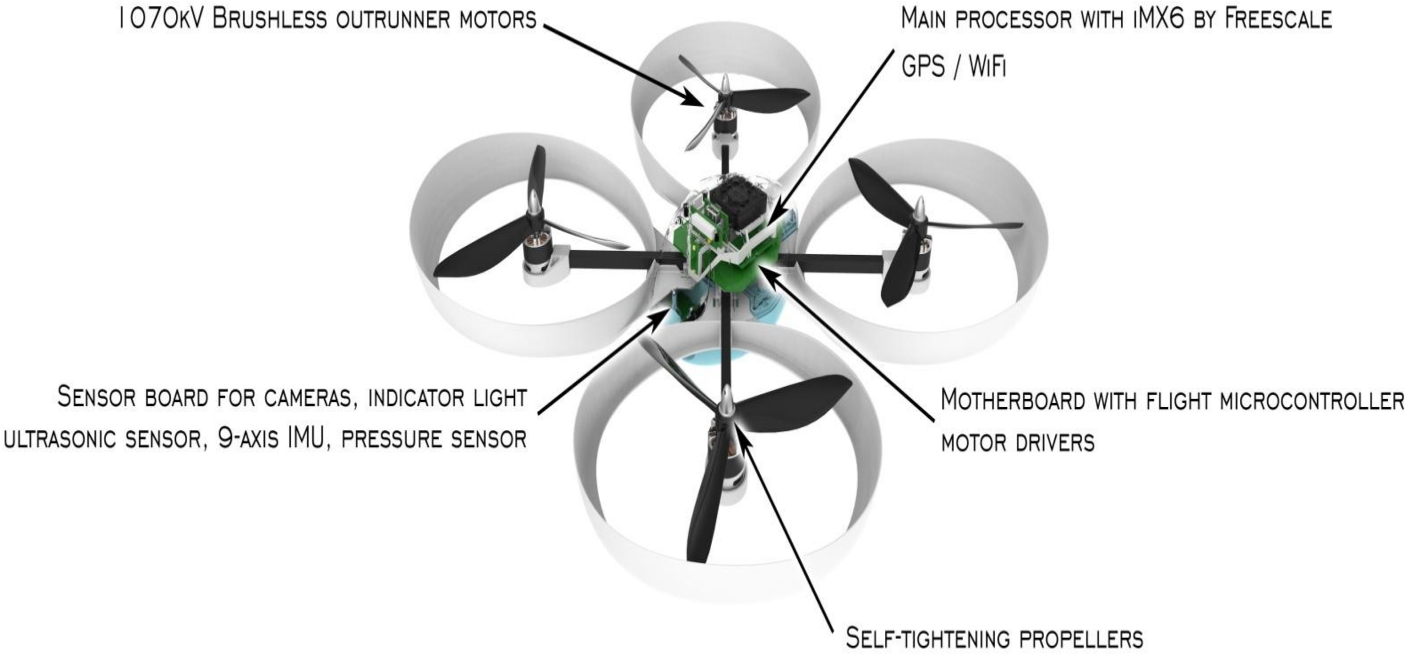

Figure 2-1-7: Structure of Spiri quadrotor drone

2.1.2 Programming Language

There are three common programming languages that are used in developing computer codes for flying drones. These are:

1. C programming language
2. C++ programming language
3. Java programming language

1. <u>C programming language</u>

Parrot Company released a SDK for developers to create innovation applications using the AR.Drone. The AR. Drone API (Application Programming Interface) is a good reference project to develop applications for the AR. Drone.

Software developers implement the SDK (Software Development Kit) in development of Augmented Reality games based on the AR. Drone. The SDK source code is written in C programming language and proper documentation is included. The SDK package consists of API source code and control applications for iPhone and Android. Parrot Company also suggest to use the AR.Drone Tool in their drone software development since it is a framework that has designed for create applications easily.

2. <u>C++ programming language</u>

In addition, some computer programmers develop their AR Drone 2.0 control application by using C++ programming language. For example, Appmethod is a well-defined computer application for user as it can connect both desktop and mobile devices with the latest Internet of Things (IoT) gadgets such as computer devices, AR. Drone, cloud services and sensors. Besides that, Appmethod solves a lot of difficulties that faces by application developer when they build and extend applications for IoT in different operating systems architecture.

3. Java programming language

In this project, Java programming language is used to develop the flying system. Java is a generic programming language and used as computing platform that released by Sun Microsystems in year 1995.

Java is an object-oriented (OO) programming language. It usually consists of packages, prewritten classes, and interfaces. It is known as Java API (Application Programming Interface). With Java API, computer programmer is able to write their applications by creating many of the objects that are needed.

Java programming language has been widely used as general-purpose language since it is easier to understand and implement among programmers. Java programming is well accepted nowadays as Java programming language is English-like hence it is a simple, familiar and object-oriented language structure.

Java programming language provides outstanding performance as compare to other available computer programming languages such as C and C++. It offers better portability and simplified programming syntax.

Furthermore, the term “Write Once Run Anywhere”, a.k.a WORA that coined by Sun Microsystems is realized as it allows compiled Java code to run on all common supported platforms without recompile program. Java can be coded on any electronic computing device and compile into standard byte code.

The reason why programmer prefers to write computer program in Java programming language is because there are a lot of free comprehensive sources including third-party libraries and codes which is well written.

2.1.3 Algorithm

In this section, the method that is implemented to estimate the state vector of the drone is presented. At the same time, feedback to the drone is required for performing indoor autonomous flight. The method that adopted is the Kalman Filter (KF) algorithm.

The algorithm is named after his name, Rudolf E. Kálmán, a successful Hungarian-born American. Rudolf E. Kálmán is a famous electrical engineer, inventor, and mathematician.

The Kalman filter algorithm is presented over 50 years. It is one of the most important and yet common data fusion algorithms that are widely used nowadays.

The development of Kalman filter algorithm bring a lot of benefits in most of the applications in technology field such as control systems, signal processing, and Guidance, navigation and control. It is recognized and effective since it only require small computational requirement. It has an elegant recursive properties and optimal estimator for one-dimensional linear systems. This powerful filter algorithm able to estimate of past, present, or even future states when the precise nature of the modeled system is remain unknown.

A most commonly adopted approach in UAV state estimation is the Extended Kalman Filter (EKF). It is a generalization of the Kalmon filter algorithm for non-linear systems. Kalman filter has been adopted and widely used in the extensive research and application, especially in the field of autonomous navigation.

2.1.4 Summary of the Technologies Review

Different hardware platform of flying drones that are available in the market are reviewed. In the previous chapter, the following three flying drones are discussed in details:

1. Phonex micro drone
2. 3D IRIS+ quadcopter
3. Spiri

Name of Flying Drone	Advantages	Disadvantages	Comment
1. Phonex micro drone	- Intelligent, interactive and programmable drone - Respond to specific instructions : Spoken commands, recognize whistle sounds and hand gestures - Extremely light in weight	- Low payload weight : Unable to mount extra add-on electronic sensors or electronic boards	- Although it is consider as smart and intelligent flying drone, it does not have the capability to fly in autonomous navigation mode.
2. 3D IRIS+ quadcopter	- Fly stable : Good choice for photo shooting and video recording - Equipped with full GPS-guided autonomous capabilities - Quick re-	- Customer-based drone application : Software customization and programming not available (Non-programmable drone)	- The drone is rather expensive compare to other flying drones that are available in market. - The drone primarily being used for

	programmable GPS waypoints for missions : Scientific research and Mapping		commercial purposes such as aerial shots.
3. Spiri	- Programmable and autonomous flying drone - Interesting interactive methods : give verbal instruction to Spiri by using a smartphone, play music to it and blows whistle to command it - Built-in Obstacle Detection Functions	- Proper documentation of Spiri : API and flight control libraries is still in progress - Low payload weight : approximately 100g [3]	- The Spiri drone project relatively new as the project is started on year 2013. 4. However, Spiri drone has the ability to avoid obstacles during flight which is better than Phonex micro drone and 3D IRIS+ quadcopter.

Table 2-1-8: Summary of the technologies review

Respect to the programming language selection, Java programming language will be used in this project as discussed earlier on. And lastly, Kalman Filter algorithm (KF) also implements to estimate the state vector of the drone.

2.2 Review of the Existing Systems

It is important to understand the existing Obstacles Avoidance Systems clearly before starting the progress on developing mobility enhanced system of a flying drone.

Existing system evaluation is carried out to identify the areas where improvement can be done on existing systems that are under-performing. Comparisons between existing systems are presented in section 2.2.4. Table 2.2 clearly shows the different localization and mapping methods adopted in order to perform autonomous flight of a flying drone. Besides that, online journals and articles that related to drone mapping strategies are shown as below:

2.2.1 Framework for Autonomous Onboard Navigation with the AR. Drone

In the past recent years, a lot of researches have been done on micro unmanned aerial vehicles. Quadrotors are become a famous and welcomed choice among MAV platforms because of their simplicity in mechanical, robustness, small in size, and low in weight.

Quadrotor has the ability of perform aggressive maneuvering, ball juggling, and triple flips. These abilities are fully utilize in following areas: search and rescue, surveillance, exploration, and artistic performance.

Commercialize drones from Microdrones and Ascending Technologies companies are the most frequently used in research due to their robustness. However, they are expensive and the firmware is not customizable.

Therefore, some researches started to develop their own platforms to solve the problem. Their platforms are designed according to application's needs. It will be costly in terms of time and knowledge required to build any customize platform. Low cost solutions are most favored among researches. Nowadays, low cost sensors are widely implement and used for autonomous navigation for quadrotor.

For example, in 2010, Parrot Company designed and launched a brand new AR. Drone. The drone was developed for fulfill the needs of video games. The development of AR. Drone captures attention of research communities with its special features such as stable hovering and on-board cameras. Soon, the AR. Drone becomes a very welcomed platform for research and academic needs. In the past few years, researches on object following, autonomous navigation and position stabilization have been conducted and tested in AR. Drone.

Frontal and vertical cameras that are mounted in AR. Drone are used for learning feature map of a pre-defined area. To navigate through the area, drone has to learn the feature map and compute on AR. Drone's odometry. Bills et al. stated that indoor environments are mostly based on images from the AR.Drone.

The drone depends on visual cues from the environment to fly through it. Engel et al. carried out a series of flight tests that are using images from the drone as an input for mapping algorithm, also known as "PTAM" and the parallel tracking.

The result of PTAM was integrated with the drone's inertial measurement unit (IMU) data for estimate it's position more accurately. Normally, a base station communicates with the drone via a wireless network. It limits the flying area and experiences delay in sensor data exchange and control commands.

2.2.2 Identification and Path Following Control of an AR. Drone Quadrotor

A large number of researches and papers on this topic have been established in the past years. In order to implement the idea of Identification and Path Following Control of an AR. Drone Quadrotor, Simultaneous Localization and Mapping, also known as "SLAM" was implemented for create 3D environmental maps. SLAM also used for establish quadrotor position in space. Most of the automatic navigation and object recognition projects are implement closely with various visual methods. They filter the data that obtain from on-board cameras and sensors.

The Simultaneous Localization and Mapping (SLAM) enable the possibilities for flying drone to be placed at an unknown location in an unknown environment. This process incrementally builds a consistent map of the environment. At the same time, it also simultaneously determines its location within the map. In SLAM, both of the trajectory of the platform and location of every landmark are estimated online without knowing the starting position given.

Theoretically, if a quadrotor set-up to be fly in "Path following" method, it follow a desired geometric path. Its constraint is limited to space but not in time that it takes to reach the destination. The Path following is implemented by controlling of the position controllers in 2D vertical plane. Combination of on-board cameras and sensors are used for follow as a reference geometric path in its visual frame.

2.2.3 Adaptive path planning for unmanned aerial vehicles based on bi-level programming and variable planning time interval

On the basis of a novel adaptability-involved problem statement, bi-level programming (BLP) and variable planning step techniques are introduced to model the necessary path planning components and then an adaptive path planner is developed for the purpose of adaptation and optimization.

They extend the considerations of diverse environments and a UAV's single ability change to an integrated demand on multiple performance changes or fluctuations for better adaptability. Meanwhile, to overcome the obstacles of the fixed step mode, a new idea of variable planning step is adopted to adaptively update flight paths. These two novel ideas are their improvements and advantages over the existing path planning formulations in a sense that they deal with the adaptability problem under uncertain conditions. This study focuses on the development of an adaptive path planner that is able to quickly search optimal or near-optimal flight paths for general UAVs with different performances, including sensory capability, maneuverability, and flight velocity limit. However, the UAV with limited maneuverability would fail to proceed in practice.

2.2.4 Summary of the Existing Systems

Existing System	Advantages	Disadvantages	Critical Comments
1. Parallel Tracking and Mapping (PTAM)	1. Robustness 2. Learn the feature map based on images from drone	1. Expensive and non-customizable firmware 2. Limit the flying area 3. Experience delay in sensor data exchange and control commands	This approach is not effective since its flying area is limited up to Wi-Fi range. Besides that, it also suffers delay in sensor data exchange. This can be a serious issue when drone is carrying out real-time live critical task.
2. Simultaneous Localization and Mapping (SLAM)	1. Slightly faster in filter data from on-board cameras and sensors 2. Follow a desired geometric path	1. Limitation of spaces 2. The estimated path and constructed map are easily affected by surrounding noise. As it travelling further, both of them become increasingly inaccurate.	This approach is slightly better than previous approach. However, this approach is limited to space. This approach could be a good choice if drone is set to be flying by following a pre-defined geometric path.
3. Adaptive path planning based on Bi-level Programming	1. Better adaptability when changes or	1. Not all drones can implement in this practice	This approach is suitable and useful in rapid changes or fluctuations

and variable planning time interval (BLP)	fluctuations happen 2. Adopted to update flight paths		environment during flight. However, this approach is not working in those UAV or drone that has limited maneuverability.

Table 2-2-1: Summary of the existing systems

2.3 Concluding Remark

Few of the existing collision avoidance systems for electronic flying devices are discussed and studied in this chapter. Effort in investigation of each collision avoidance systems should effectively play important role for the best flying performance. However, it may be worth to consider any other anti-collision algorithms as well as collision avoidance systems that are even more elaborated configurations and roles.

Chapter 3 – System Methodology

3.1 System Development Models

System Development Model refers as SDLC. It is an abbreviation for "Software Development Life Cycle". Professional entities often also refer it as System Development Life Cycle or Software Development Process. Software Development Life Cycle (SDLC) is a method, process or a set of methodologies that are applied in software project for create or alter the project. Each type of these methodologies is unique and useful in creating a new computer program or software module.

In summary, there are some topical standardized stages in any Software Development Life Cycle (SDLC) process that are consists of:

- Stage 1 : Planning and Requirement Analysis
- Stage 2 : Defining Requirements
- Stage 3 : Designing the product architecture
- Stage 4 : Building or Developing the Product
- Stage 5 : Testing the Product
- Stage 6 : Deployment in the Market and Maintenance

Fundamentally, there are five common system development approaches which can be serves as pillar of the SDLC process. These techniques are known as SDLC methodology.

They can be summarized as following:

- Waterfall Model
- Incremental Model
- Spiral Model
- Evolutionary Prototyping Model
- V- Model

3.1.1 System Development Model 1 – Waterfall Model

This SDLC model works in linear sequential flow. Waterfall model can be described as which its progress flows steadily downwards (from top to bottom) throughout each phase of software development. In other words, only if the previous phase is completed, then its following development only can be begins. However, this SDLC model does not specify the process of going back to the previous phase to handle changes in requirement. Waterfall model is the earliest and widely used approach in software development.

This model is often implemented in projects that do not focus on changing requirements, for example, responses for request for proposals.

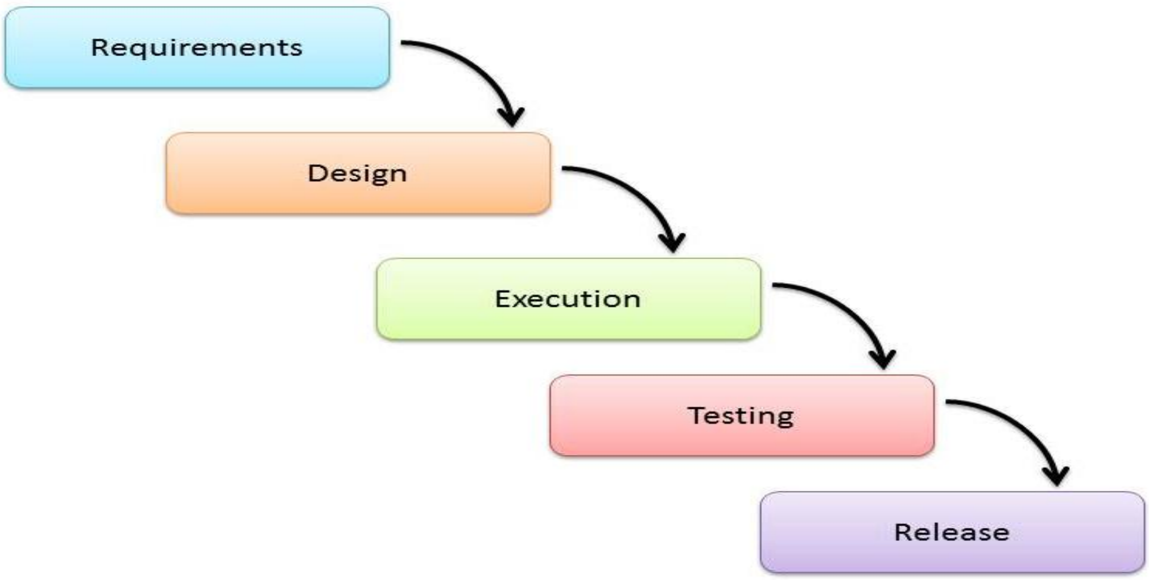

Figure 3-1-1: Waterfall Model

3.1.2 System Development Model 2 – Incremental Model

This SDLC model is introduced and developed to solve the weaknesses in the waterfall model. Incremental model starts with an initial planning and it is ends with the cyclic interactions in between. This model is used to develop a system through a series of repeated cycles, which also known as "iterative". In smaller portions at a time, we named it as "incremental". This approach allows software developers to learn earlier parts of the system during development stages.

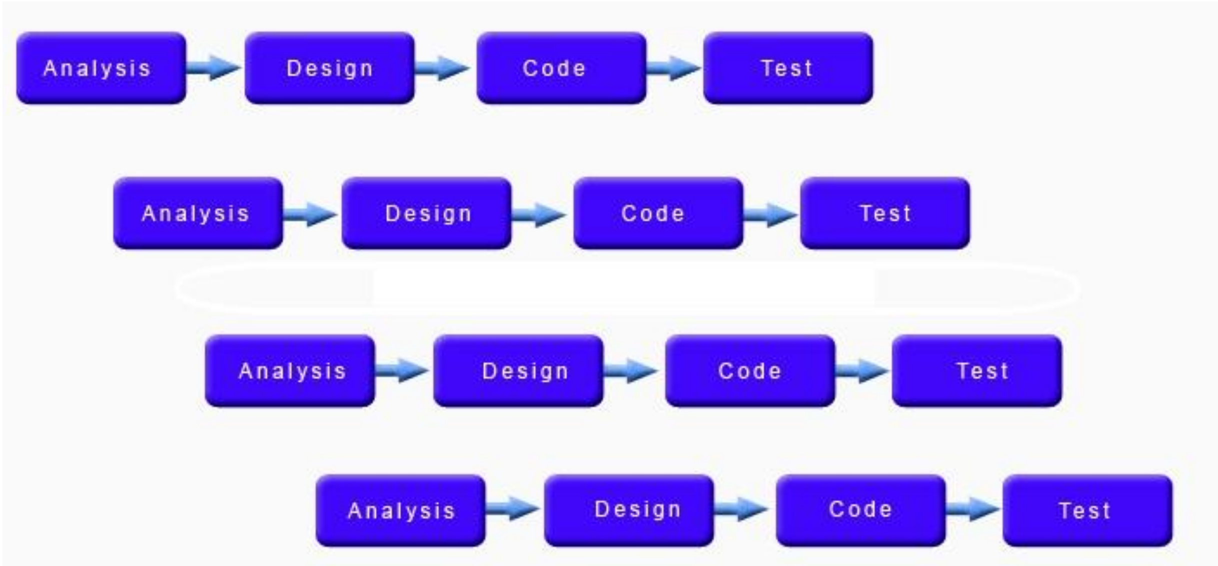

Figure 3-1-2: Incremental Model

3.1.3 System Development Model 3 – Spiral Model

This SDLC model is combine both elements of designing and prototyping-in-stages. This is a good approach in combining advantages of top-down and bottom-up concepts. It combines features of waterfall model and prototyping model. Project that is large in scale and complicated is mostly preferred to adopt this SDLC approach.

Spiral model re-uses many of the same phases as in the waterfall model. Theoretically, it follows the same order and separated by planning stages. It has risk assessment, prototyping, and simulations.

Spiral model is normally implements in large-scale system and shrink-wrap application which it can break into small phases in project.

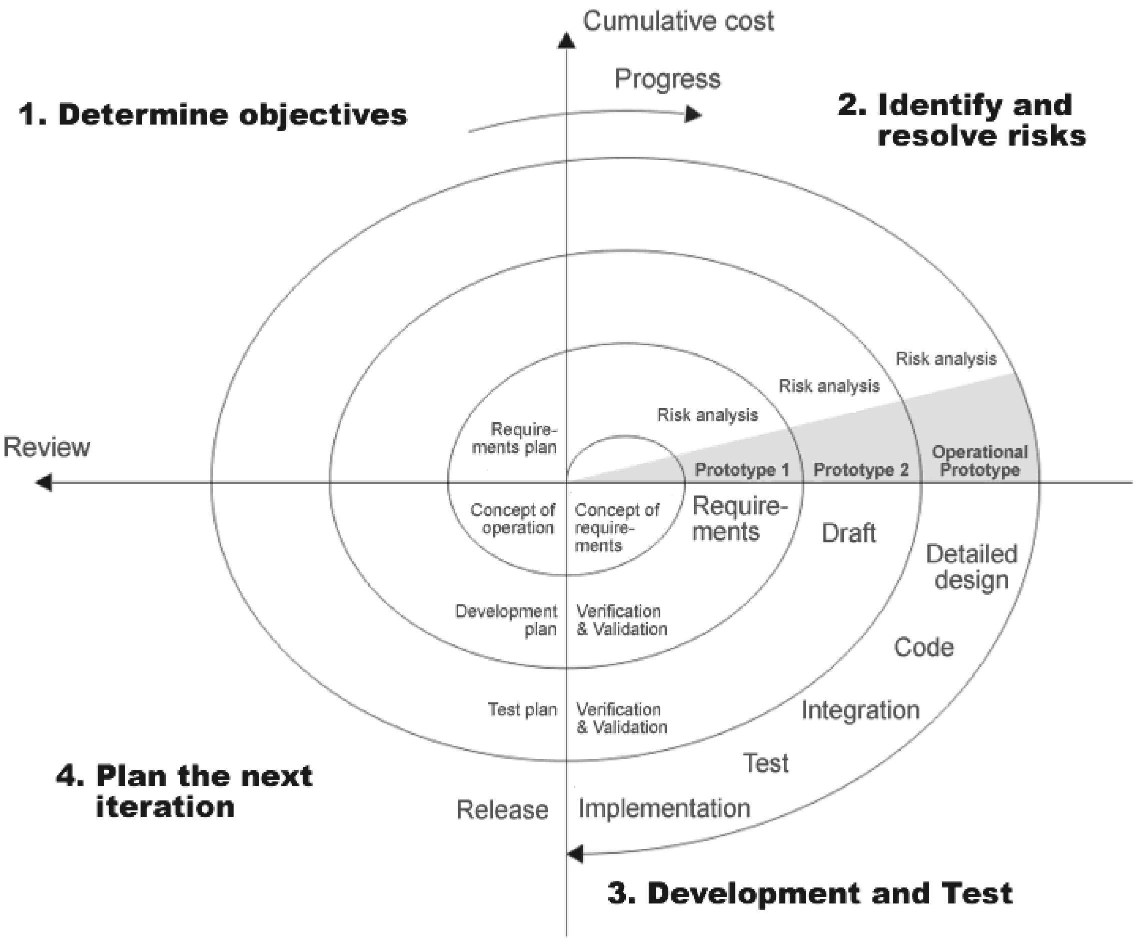

Figure 3-1-3: Spiral Model

3.1.4 System Development Model 4 – V Model

This SDLC model is an extension for waterfall model. This is a unique approach that each process steps are bent upwards after the coding phase to form the V shape. The most significant difference between V- model and waterfall model is the early test planning in V- model.

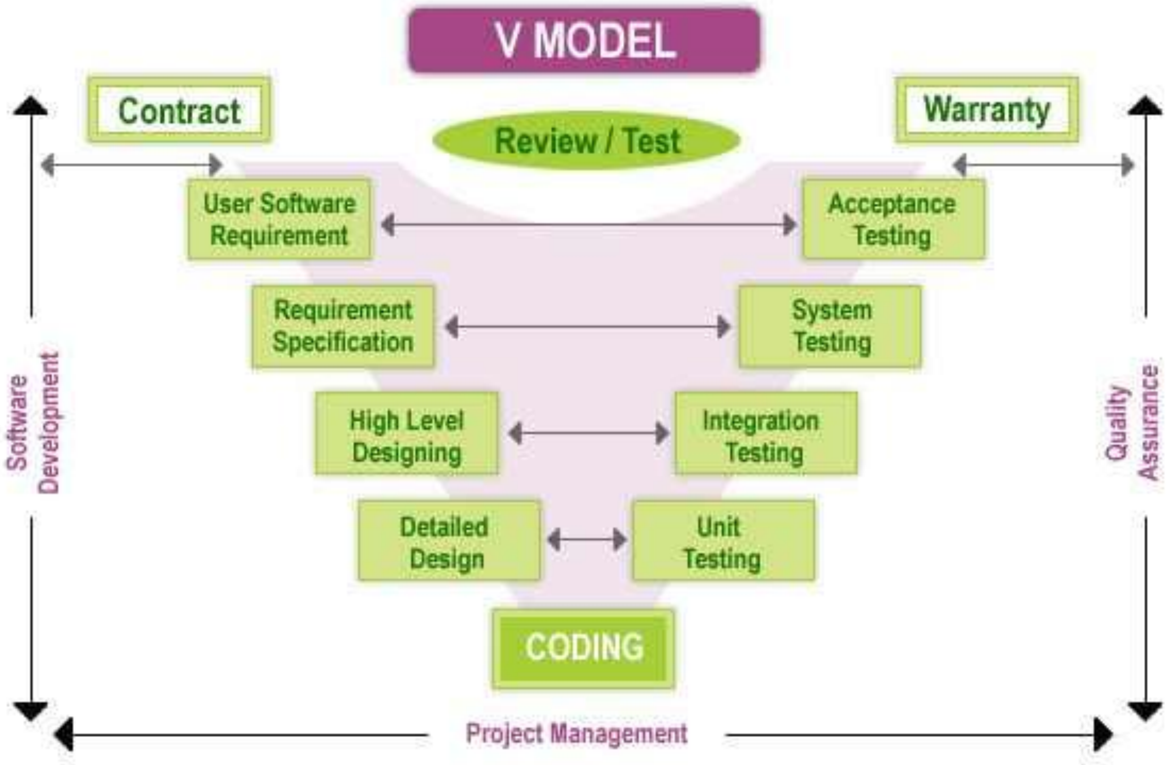

Figure 3-1-4: V- Model

V- model can be implemented when the software requirements are clearly specified and defined. In addition, this SDLC model also can be adopted when we know the software development technologies and tools very well.

The table below lists out the advantages and disadvantages of V- model:

Advantages	**Disadvantages**
Simple to use and understand	Not suitable for long and ongoing projects
Works well in small project whereby its requirements are very well defined	Bad choice for complex and object-oriented projects
Highly disciplined model and each phase is complete at a time	Uncertainty and high risk
Easier to manage since each phase has its deliverables and review on each process	Hard to "roll back" to change its functionality once an application is in the testing stage
	Not suitable for moderate to high risk of changing projects

Table 3-1-5: Advantages & disadvantages of V- model

3.1.5 Selected Model

V-model as SDLC methodology model is selected for this project. The below figure illustrates a clear view on different activities in each phase of the V-model:

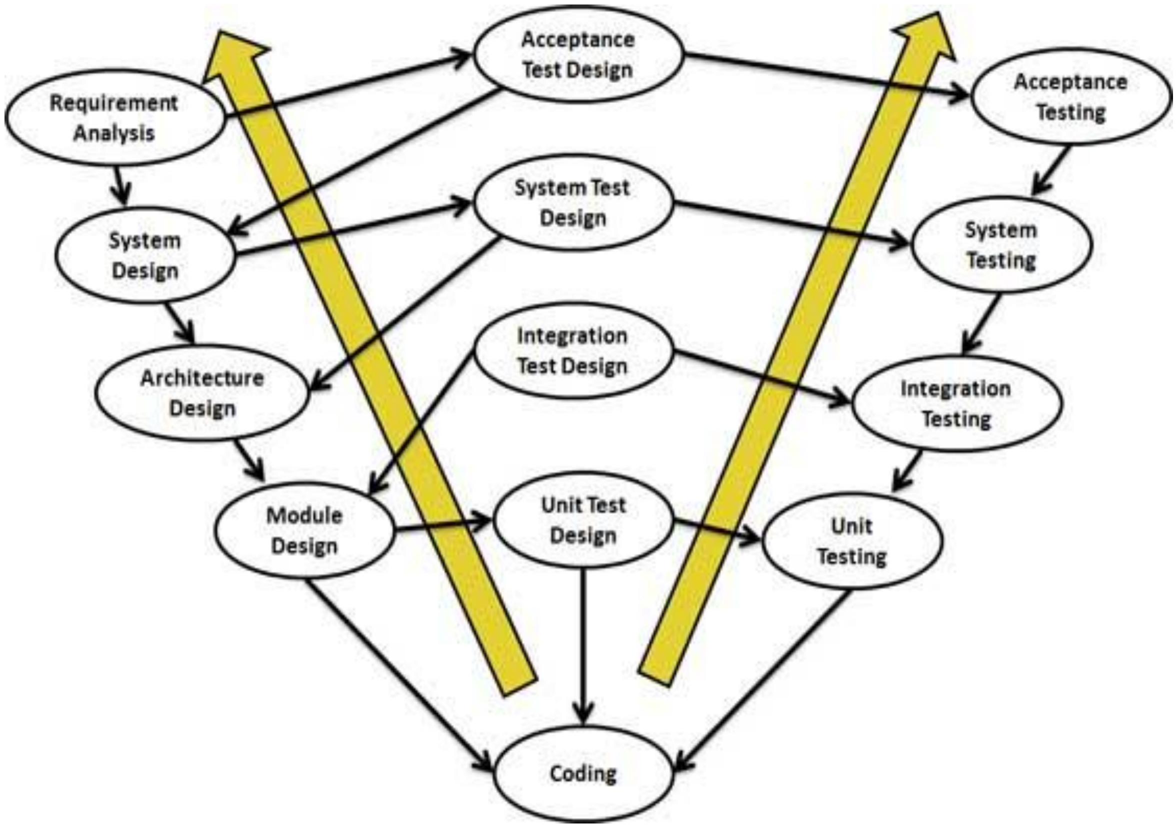

Figure 3-1-6: V- model Structure

In V- model, execution of processes are happens in V-shape sequential manner. This model is also widely known as "Verification and Validation" model.

Each single phase in the development cycle is directly associated with its testing phase. This is a very highly ordered model which means next phase only will be executed after the completion of the previous phase.

The primary factor to consider selecting this operative approach is each testing phase need to carry out to ensure every stage is free from error. This model emphasize on the completion of the previous phase. In other words, this project only can be continuing if the previous phase is verified and validated well.

For example, if camera detection function is not working correctly, then the project cannot be proceed to develop since this project relies heavily on camera detection for flying in Autonomous Navigation mode (Without GPS module and GPS service present).

As a conclusion, only if problem solved, then proceed to the next stage is allowed and this can guarantee the correctness of this project.

3.2 System Requirement

A system requirement is a well structured document that describes and list out all the description of a system's function, services and operational constraints. In other words, it is detailed specifications of user requirements. It serves as a foundation for designing of a system. System requirement also specify what should implement in a system.

In this section, the system requirements of this project are highlighted and listed in sequence as follow:

1. The system shall be able to communicate with drone by establish Wi-Fi connection into drone network
2. The system shall be able to send abstract command to drone
3. The system shall be able to receive data from drone and process data packets
4. The system shall be able to issue appropriate abstract to drone for avoid any frontal obstacle along its flight path

3.2.1 Hardware

1. Parrot AR. Drone 2.0

Figure 3-2-1: Parrot AR. Drone 2.0 with protection hull attached

The AR.Drone 2.0 is a remote-controlled quadcopter that is developed by Parrot SA Company. AR.Drone 2.0 drone body structures are made up of high impact resistance plastic as well as carbon fiber tube. The protection hull is made of Expanded Polypropylene (EPP) foam. It is durable and light in weight. The protection hull also provides extra protection for indoor flights. There are four DC brushless motors which operate at 35,000 rpm and consume 15W each. The drone equipped with four high-efficiency propellers. A Lithium polymer battery with a capacity of 1000 mAh is needed to power up the drone. A 1000 mAh battery allows approximately 10 minutes flight time.

The AR. Drone is operated with an internal-built processor: a 468 MHz ARM9-processor. It also consists of 128MB RAM. AR. Drone is running on a custom Linux operating system. Besides that, there is a mini-USN connector attached for the purpose of software flashing and to attach other add-ons, for example, GPS sensor. An integrated 802.11g wireless card is mounted inside AR. Drone to provide network connectivity.

This remote-control quadricopter is chosen as flying device for testing out this final year project. The on-board electronic devices such as frontal and vertical mounted cameras are used for obstacle detection purpose. The flying drone should able to perform simple flying abstract commands such as take off and land at selected destination by the flyer.

This unmanned aerial flying device relays heavily on different types of state-of-the-art sensors to fly:

(i) A 3-axis Accelerometer with +/-50 mg precision

(ii) A 3-axis Gyroscope 2000 °per second precision capabilities

(iii) A 3-axis Magnetometer with precision up to 6 °

(iv) A Pressure Sensor with +/- 10 Pa precision

(v) A 60 FPS vertical armed QGVA Sensor type camera for measure ground speed

(vi) Ultrasound Sensors aboard for measure ground altitude

These sensors are installed and used during flights for flight stabilization and on-board cameras provide visual feedback from the flying drone.

In fact, Parrot AR. Drone 2.0 is an interesting flying machine. It is an ideal and affordable toy for helicopter or quadcopter enthusiast to fly with. This inexpensive flying drone is often cost around $300.

2. Personal Computer (Laptop)

Figure 3-2-2: Personnel Computer (Laptop)

This electronic equipment is used in developing and implements programming codes as well as object collision avoidance algorithms for the final year project. A series of flight tests carried out by establish PC Wi-Fi connection to the Parrot AR. Drone 2.0 using laptop.

3.2.2 Software

1. JavaDrone API

JavaDrone open API platform is selected and implement in this project.

The JavaDrone API is a main reference platform since PC client needs to communicate with AR.Drone cameras and other electronic sensors such as Accelerometer and Gyrometer to perform Autonomous flying. It does provide flexibility in supporting both video formats which are H.264 and MPEG video.

It also includes some simple API source code and control application examples.

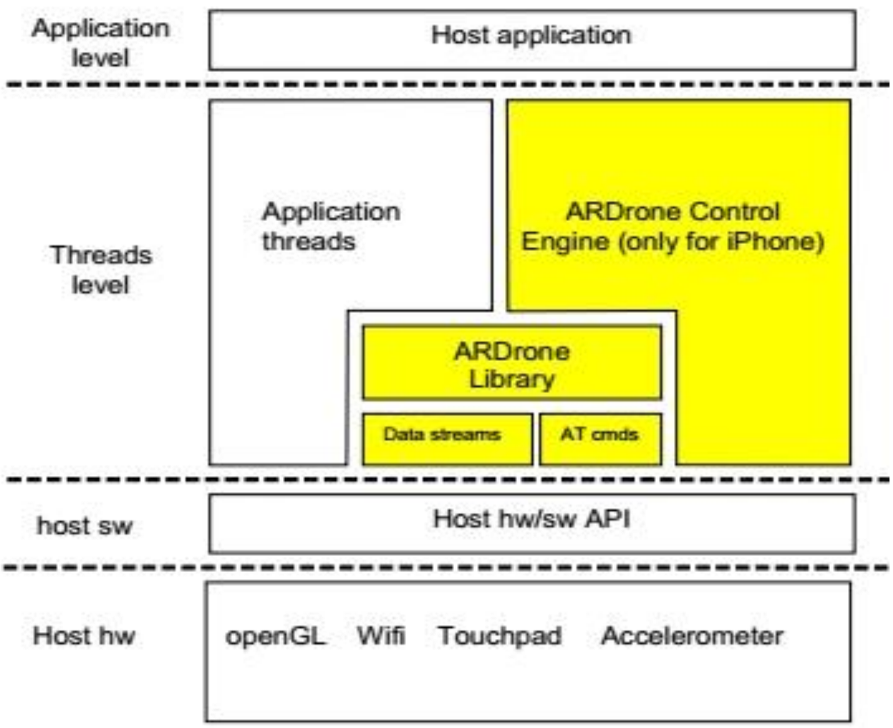

Figure 3-2-3: Overview of layered architecture of a host application

2. JavaCV

Figure 3-2-4: JavaCV logo

OpenCV stands for “Open Source Computer Vision”. OpenCV is widely use in academic and commercial field. It has support a variety of programming languages, for example, C++, C, Python and Java. It supports Linux, Windows, iOS, Android, and Mac OS.

OpenCV was developed in year 1999 at Intel research labs. It was written in C programming language. OpenCV can be coded and implement in any real-time application. These are the reasons of why choosing OpenCV in this project. OpenCV library provide all of the necessary libraries to capture and analyze real-time images that capture by the AR.Drone. OpenCV’s implementation allows computation of vision algorithms and linear algebra operations easier and faster.

JavaCV is one of the OpenCV’s wrappers. In other words, when running a JavaCV based program, the program is actually using OpenCV as well by calling another interface.

JavaCV offers many wrappers around OpenCV. It comes along with various well known image processing libraries such as OpenKinect, FFmpeg and others. Therefore, whether choose to use OpenCV or JavaCV, it is not a big issue.

However, JavaCV is chosen as it suits and fulfill for the needs of image processing in this project.

3.3 Functional Requirement

This section describes what the basic function requirements in this project are. A function requirement specifies the behavior and functionality of the system. In order words, it should describe what a software system is supposed to perform. Functional requirement usually involves data manipulation, scientific calculations, technical details, processing and other specific functionality.

Hardware Interfaces: Define the physical and logical characteristics of each interface. These include the software product and hardware components of the system.

In this project, several on-board sensors such as frontal and vertical cameras for obtaining visual feedback from the flying drone are used. Besides that, in order to keep the drone flies steadily, sensors such as pressure sensors will be relied heavily. On-board pressure sensors are able to perform any necessary flight adjustment automatically and maintain a fixed position in the air regardless of altitude and wind up to 15 m/h.

Software Interfaces: Define the relationship between this project and other specific software components such as tools and API libraries. Nature of communications and services needed are listed here as well. Data that send and receive across software components are defined in this section as well.

In this project, the JavaDrone API libraries to control the AR. Drone 2.0 are studied and implemented. API clearly states out how a program will interact with the rest of the software. It is a set of requirements that control and govern how an application communicates to another.

Communications Interfaces: Evaluate the requirements that associated with any communications functions which are required by this project including “client-and-drone” communications protocols. Communication standards such as TCP or UDP protocol that will be used in this project are also identified. In this project, we are fully utilized on AR. Drone 2.0 as our hardware platform to test the flight performance as well as frontal obstacle avoidance functionalities. The AR. Drone 2.0 generates a Wi-Fi hotspot (open network) once it switches on. Communications between PC (laptop) and the flying drone is handled by Wi-Fi modules on both devices.

AT commands are used as an important medium to communicate with the AR. Drone 2.0. Commands are sent to the AR. Drone in TCP or UDP packets format. A simple UDP packet should contain at least one complete command or more. A slightly delay latency of 30 MS will occur to pass between the commands.

There are four main important ports that are available for communications:

1. Port 5554 – Reply from AR. Drone of UDP data packets.
   It contains information about the drone status, speed, engine rotation speed, etc.
2. Port 5555 – Reply from AR. Drone of video stream packets.
   All of the images or video that captured by frontal and vertical mounted cameras will be send by the flying drone and pc receives these data via this port.
3. Port 5556 – Send UDP packets with regular commands.
   This port is mainly used for configuring and controlling the flying drone. The commands are sent on a regular basis about 30 times per second.
4. Port 5559 – This port also called as control port. It can be established on TCP port 5559 for transfer critical data that cannot be lost. It is used to fetch important data such as configuration data and acknowledge critical information.

3.4 Project Milestone

ID	Task Name	Duration	Start	Finish
1	Review Project Proposal	4 days	January 12, 2015	January 15, 2015
2	Literature Review on Technology	4 days	January 16, 2015	January 21, 2015
3	Study on Existing Systems	5 days	January 22, 2015	January 28, 2015
4	Outline System Design	6 days	January 29, 2015	February 5, 2015
5	Design Graphic User Interface	2 days	February 6, 2015	February 7, 2015
6	System Prototyping	30 days	February 9, 2015	March 20, 2015
7	System Testing 1	7 days	March 23, 2015	March 31, 2015
8	System Debugging	4 days	April 1, 2015	April 6, 2015
9	System Testing 2	7 days	April 7, 2015	April 15, 2015
10	Project Documentation	3 days	April 16, 2015	April 18, 2015
11	Presentation	1 day	April 19, 2015	April 19, 2015

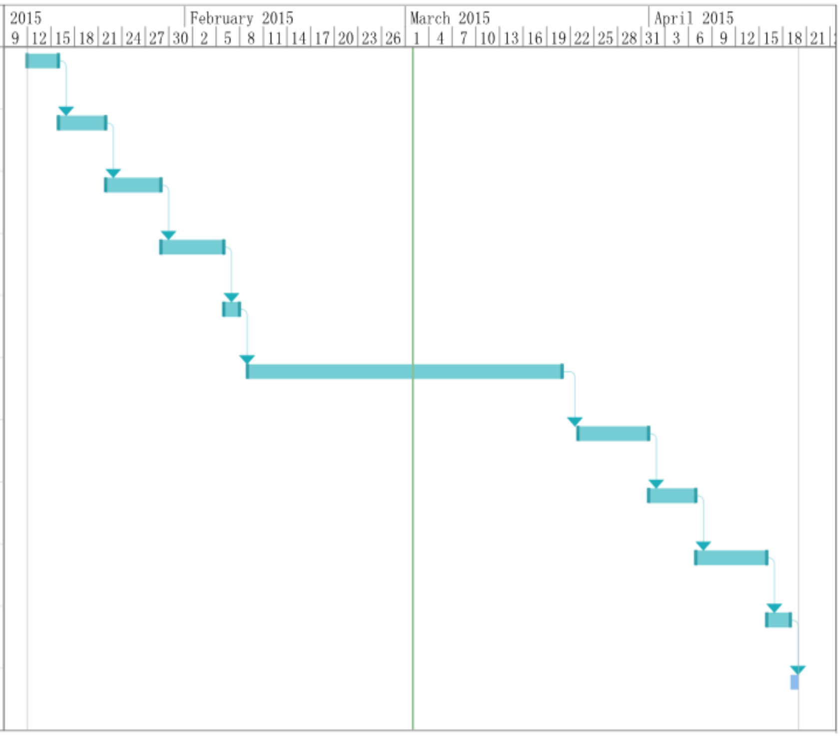

Figure 3-4-1: Project Milestone &Gantt chart for Final Year Project 1

There are total of eight study weeks to complete the report on Final Year Project 1. Figure 3-4-1 above clearly shows the duration and timeline used in the project. The project started with "Review on Project Proposal" and end with "Project Presentation". The longest duration in this Gantt chart falls on the activity of system prototyping. Out of 56 days, there are 30 days, that is equivalent to 53.57% of days, has been concentrate on develop basic system functions and integrate initial system.

There are two project milestones to achieve in the Project 1. The first milestone falls on activity of System Prototyping. The second milestones are the project proposal submission and oral presentation which conduct on Week 14.

ID	Task Name	Duration	Start	Finish	Predecessors
1	Review Project I Proposal	3 days	June 8, 2015	June 10, 2015	
2	System Prototype Development	14 days	June 11, 2015	June 30, 2015	1
3	System Testing and Debugging	3 days	July 1, 2015	July 3, 2015	2
4	Review System Prototype	6 days	July 6, 2015	July 13, 2015	3
5	Full System Development	30 days	July 14, 2015	August 24, 2015	4
6	System Testing and Debugging	7 days	August 25, 2015	September 2, 2015	5
7	Project Documentation	7 days	September 3, 2015	September 11, 2015	6
8	Project Submission and Presentation	2 days	September 14, 2015	September 15, 2015	7

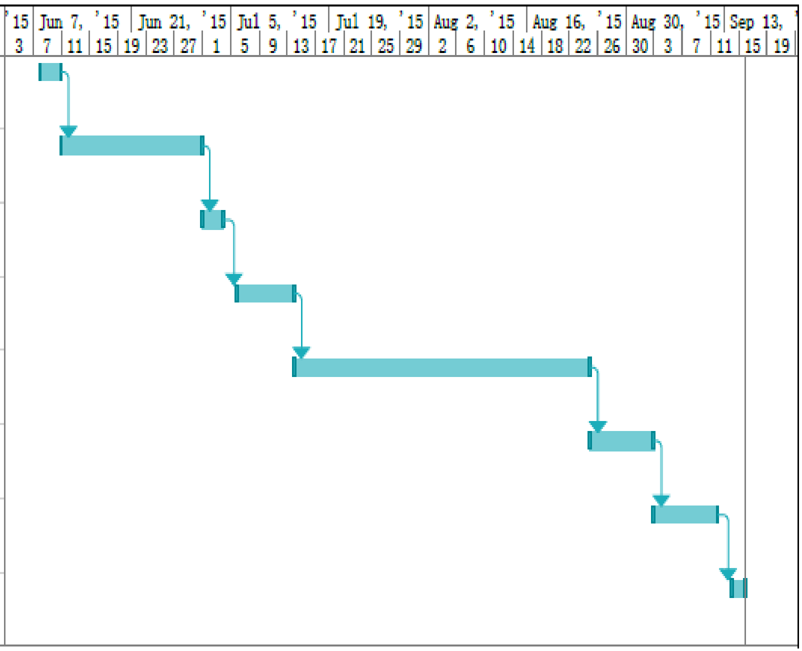

Figure 3-4-2: Project Milestone &Gantt chart for Final Year Project 2

Figure 3-4-2 above shows the project milestone for Final Year Project 2. There is a lot of time dedicated on focusing to develop the proposed system and perform system testing on it. The important project milestones for Final Year Project 2 are submission of report and presentation of the final system.

3.5 Estimated Cost

Items	For Final Year Project Development	For Commercialization
1. Parrot AR. Drone 2.0	RM 0	RM 1199.00
2. 1,500 mAh LiPo Battery	RM 0	RM 99.00
3. Computer (Laptop)	RM 0	--
4. JavaDrone API	RM 0	--

Table 3-5-1: Estimated Costs Allocation

Based on the Table 3-5-1 above, there is no cost incurred since all of the required items can be obtained from either from university lab or it is a personal belonging of mine. The Parrot AR. Drone 2.0 and a 1,500 mAh LiPo battery pack are prepared by University. Hence, there is no necessity to purchase another brand new AR. Drone for this project. The existing AR. Drone 2.0 capable of serve and fulfills the needs of this final year project. A 1,500 mAh LiPo battery was purchased by University recently. The 1,500 mAh LiPo battery could be used to extend the flight time of the drone. At the same time, the computer that is using in this project is adequate for writing system coding and performs compilation for this project. In addition, the JavaDrone API that used in this project is free of charge. It is an open source toolkit for developer and can download online for free. The URL of website that provides the JavaDrone API is https://code.google.com/p/javadrone/downloads/list.

As for commercialization, the Parrot AR. Drone 2.0 and a 1,500 mAh LiPo battery are sells at prices of RM 1,199.00 and RM99.00 respectively. The prices are standardized in the market.

As a conclusion, the total budget that is needed to allocate in this project sums up around RM 1,200 to RM 1,500.

3.6 Concluding Remark

In this chapter, studies on system methodology are carried out. Analyses different system development models and briefly explains on how the system development models works. A detailed explanation on selected system development model also included for reader to understand the importance of choosing the correct system development model. Functional requirement section gives reader overview on what are the system behavior and functionality.

Project milestones are also illustrated as in Grant Chart diagram and further explanation is provided as well. Lastly, there is a table of estimated cost for this project is outlined. This provides a glimpse on the costs for final year project development and commercialization purpose.

Chapter 4 – System Design

4.1 System Architecture

System architecture is a fundamental and conceptual model that defines a system elements, system structure, system behavior, interfaces, constraints and processes. It refers to how a system is designed and what technologies are used with. Therefore, system architecture usually consists of wide range of underlying structure of system components. Properties and relationship among components also described as well in system architecture. It shows components work together in order to create behavioral and structural capabilities. Well-designed system architecture is greatly important as it often leads to success of a project.

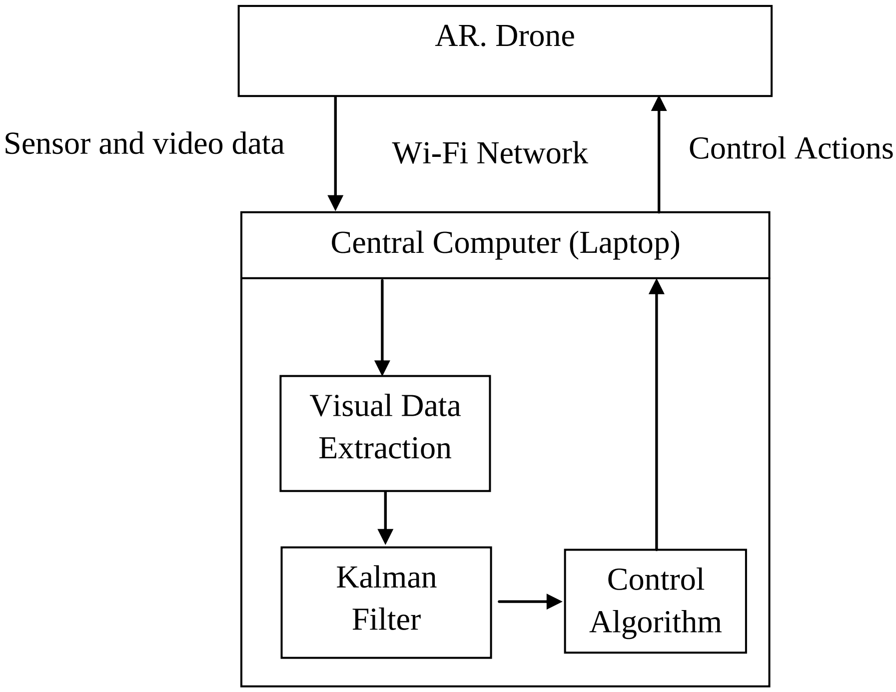

Figure 4-1-1: Autonomous indoor flight systems

For better comprehension of the overall system implementation, figure 4-1-1 illustrates on how different system components works together as one independent system. There is a centralized computer station – laptop, runs the code in an infinity loop. The sensor and video data are required to send over a communication channel that has established with the AR. Drone over a wireless LAN network on every cycle.

With the presence of data available in the centralized computer, the KF algorithm is executed. The results of the KF state estimation will be used for calculate the necessary control actions. Finally, the appropriate control action command is send to the AR. Drone over the Wi-Fi network.

4.2 Functional Modules in the System

There are three main functional modules that are implementing in this project.

1. **Wi-Fi access control module**: This module is responsible for controlling the accessibility to the AR. Drone 2.0. In order to communicate with the drone, there should be a module which allow user to turn on PC Wi-Fi module and establish Wi-Fi connection to the drone. The Wi-Fi module handles all of the communication activity between laptop and the drone. It receives and transmits data packets from to AR. Drone 2.0. It serves as gateway to communicate with the drone. Basic functions such as turn on Wi-Fi module, establish Wi-Fi connection to the drone and maintain connectivity with the drone shall be execute here.
2. **Abstract command module**: In this module, all of the abstract commands a.k.a AT commands will be coded and implemented. This module is responsible for issues abstract command to the drone. Necessary abstract commands are required to pilot the AR. Drone 2.0.
3. **Obstacle detection module**: This module is accountable for detecting drone's frontal obstacle. After receiving visual feedback from flying drone, pc client will process all of these flight data packets and the video stream packets. An obstacles detection algorithm called KF algorithm will be carried out at the same time. The KF state estimation calculates necessary control actions to prevent the drone collide with any frontal objects.

4.3 System Flow

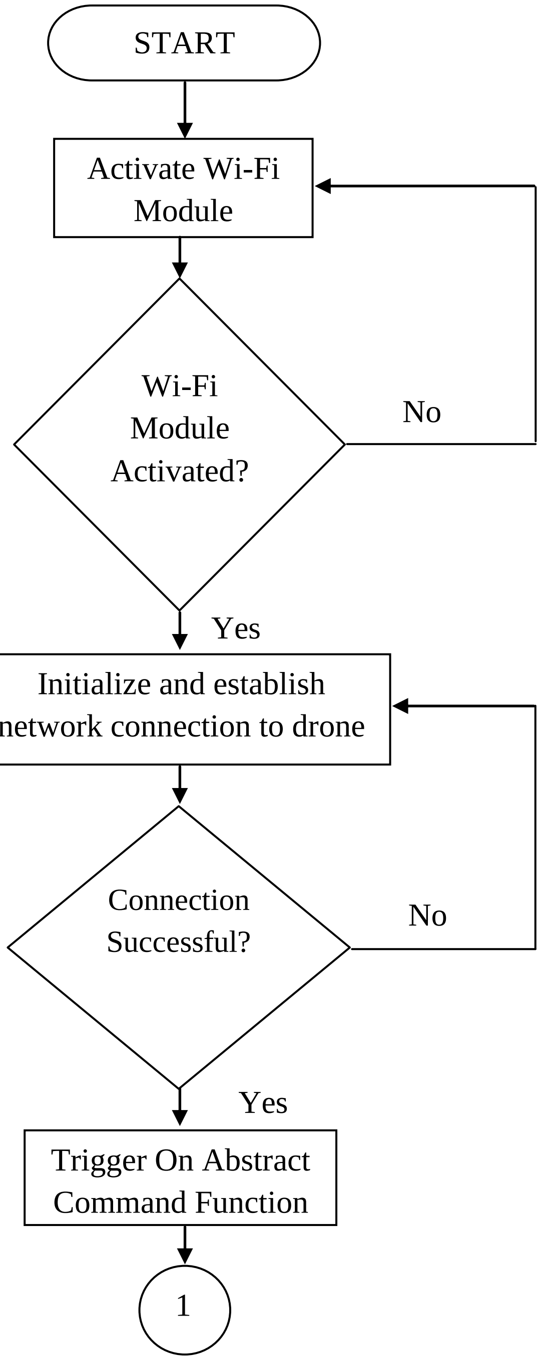

Figure 4-3-1 Flowchart of the Wi-Fi module

Abstract Command to Control Parrot AR Drone 2.0

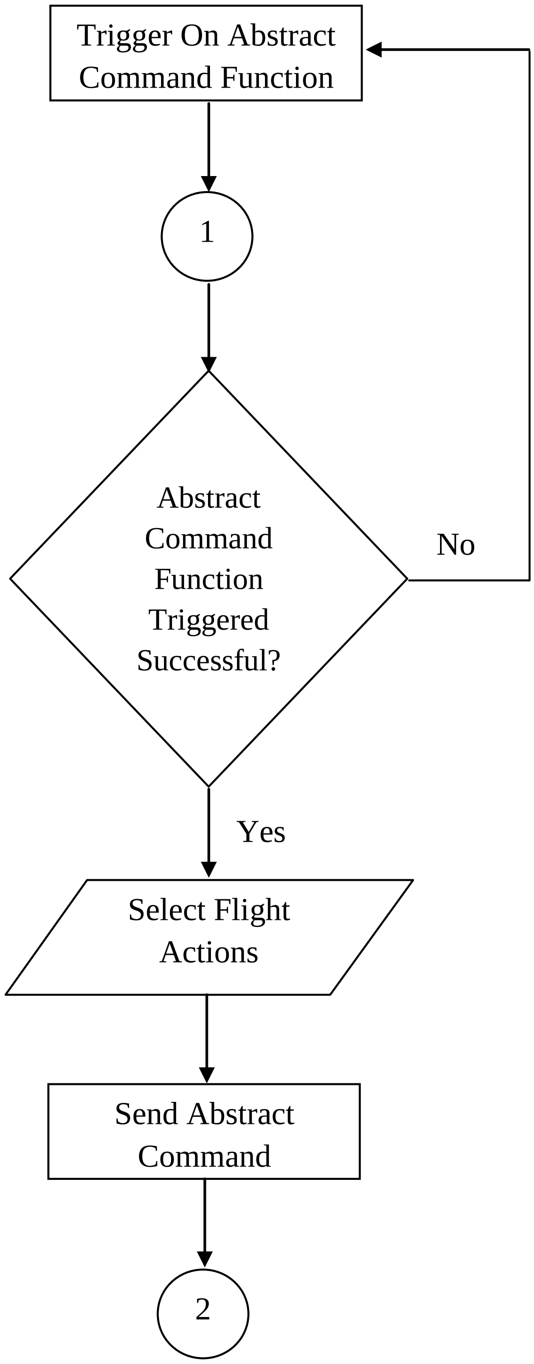

Figure 4-3-2 Flowchart of the Abstract Command module

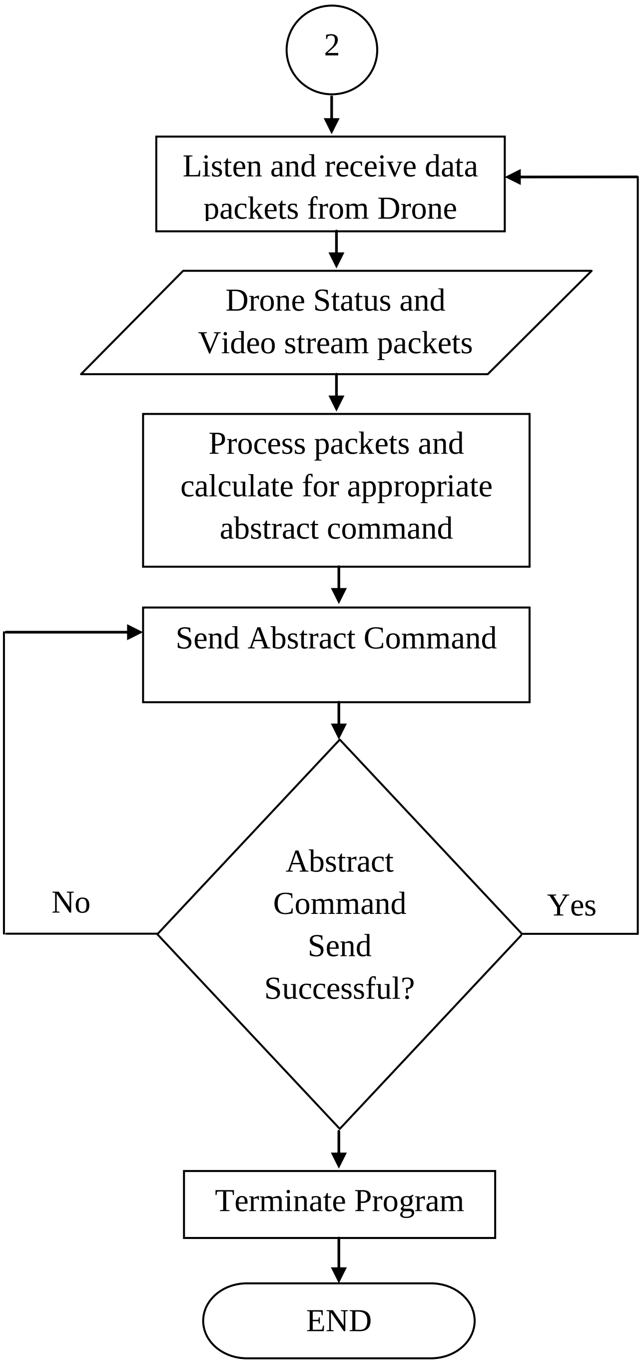

Figure 4-3-3 Flowchart of the Process Packets and Generate Abstract Command module

The figure 4-3-1, figure 4-3-2 and figure 4-3-3 above illustrates the sequence on how the proposed system executes to achieve indoor autonomous flight.

Explanation on System Flow:

1. System turns on laptop Wi-Fi module by activating Wi-Fi function.
   i. If system fails to activate laptop Wi-Fi, return to step 1.
   ii. If system successfully activates laptop Wi-Fi, proceed to step 2.
2. System initializes and establishes network connection to drone.
   i. If system fails to initializes and establishes network connection to drone, return to step 2.
   ii. If system successfully initializes and establishes network connection to drone, proceed to step 3.
3. System triggers abstract command function.
   i. If system fails to trigger abstract command function, return to step 3.
   ii. If system successfully triggers abstract command function, proceed to step 4.
4. System waits for user to select flight actions.
5. System sends abstract command to drone according to the flight actions that selected by user.
   i. If system fails to send abstract command to drone, return to step 5.
   ii. If system successfully sends abstract command, proceed to step 6.
6. System listens and receives data packets from drone.
7. System receives and accepts data packets (Drone navigation and Video stream packets) into system.
8. System continues to process data packets and issue appropriate abstract command.
9. System sends the appropriate abstract command to drone.
   i. If system fails to send abstract command to drone, return to step 9.
   ii. If system successfully sends abstract command, proceed to step 6 and step 10.
10. System prompts for user to terminate the program.

4.4 Algorithm Design

Algorithm design is a conceptual process. It consists of set of steps that take into consideration to achieve specific goal. Therefore, most of the computing operations are inter-corporate with algorithm design. Algorithms are some instructions or patterns to complete a given task in a more efficient way.

In this project, there are two important algorithms needed to identify and studies in order to implement them as one dedicated system.

Those two algorithms are: Kalman Filter Algorithm for obstacle avoidance detection purpose and Control Algorithm for controlling the AR. Drone 2.0.

Kalman Filter Algorithm:

Estimate the state vector of the drone in order to obtain feedback from the drone is very essential. Without proper feedback from the drone, state vector estimation for a drone is not possible. Hence, Kalman Filter algorithm is studied and has been chosen to implement in this project.

Kalman Filter algorithm consists of a comprehensive set of mathematical equations. It has a predictor-corrector type estimator to optimal the overall performance and reduces the estimated error covariance.

The algorithm is extensively discussed and used, especially in the area of assisted and autonomous navigation application. It use state estimation technique to control a vehicle.

A simple mathematical equation as shown below demonstrates how the Kalmon Filter works.

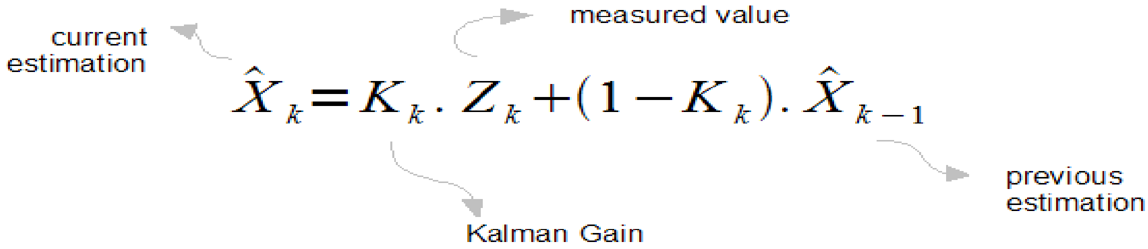

Figure 4-4-1: Generic Equation for Kalman Filter Algorithm

Refer to figure 4-4-1 above, the subscript of alphabet **k**’s is states. It can be expressed in discrete time intervals. For example, k=5 means 5 milliseconds and k=10 means 10 milliseconds.

This mathematical equation focuses on find the value of $\hat{x}_k$. $\hat{x}_k$ is the estimate of respective signal x. Each consequent of **k**'s also might need to figure out their values.

z_k is the measurement value for Kalman Filter algorithm. K_k is the key point of this algorithm and called as “Kalman Gain” in general. Lastly, $\hat{x}_{k-1}$ is the signal estimation on the previous state.

Assume that the unknown element in this equation is Kalman Gain K_k . The measurement values of K_k and previous estimated signal of $\hat{x}_{k-1}$ elements are known in this case.

Therefore, calculation on Kalman Gain K_k is needed for every consequent state.

At the same time, there are two mathematical equations of Kalman Filter that are frequently used as well. Those are:

Equation 1: $x_k = Ax_{k-1} + Bu_k + w_{k-1}$

Equation 2: $z_k = Hx_k + v_k$

The first equation calculates and evaluates the signal values of each x by using linear stochastic equation. The x_k value can be obtain by combining of the previous value, a control signal u_k, and a process noise.

The second equation combines signal value and measurement noise which can be done on a linear combination. The signal value and measurement noise are considered as Gaussian. The measurement noise and process noise are both distinct.

Generally, the entities of A, B and H are in matrices forms. These entities expressed in numeric values. Most of the time, assumption is made that these values are remain as constant while in fact, these values might change between states.

After that, estimation of mean and standard deviation for the noise functions $\mathbf{W_{k-1}}$ and **v** need to carry out as well. The Kalman Filter algorithm will try to converge all of the calculations into proper and accurate estimation.

After successfully fit the model into Kalman Filter, all of the necessary initial values and parameters are required to identify.

Another two set of mathematic equations: (1) Time Update (for prediction) and (2) Measurement Update (for correction) are applied for every k^{th} state**.**

Time Update *(prediction)*	**Measurement Update** *(correction)*
$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_k$ $P_k^- = AP_{k-1}A^T + Q$	$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1}$ $\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-)$ $P_k = (I - K_k H)P_k^-$

At this stage, we know the matrices of A, B and H will be in numerical constants whereby their values shall equal to 1.

The most difficult in this stage is to figure out the values of R and Q. However, the value of R is rather easier to identify as the noise level in the environment are known. The value of Q is not so straightforward to be determined.

The process is starts with the estimation of $\mathbf{x}_0$, and $\mathbf{P}_0$.

After collecting all of the necessary information that are needed, process can be started and iterate through the estimates. The previous estimates will affect the next input for the current state.

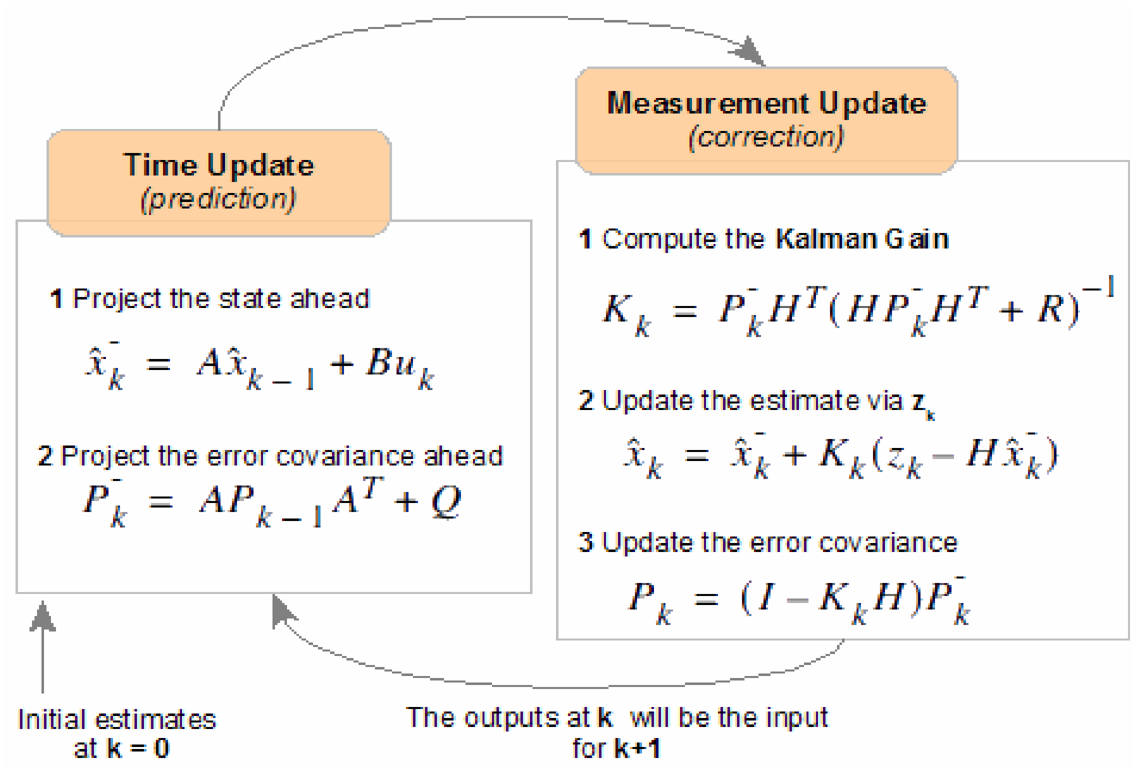

Figure 4-4-2: Time Update and Measurement Update processes of Kalman Filter algorithm

As figure shown above, the $\hat{x}_k^-$ element serve as the “prior estimate” before the measurement updates process done. Meanwhile, P_k^- serve as the “prior error covariance”.

The word “prior” refer as “rough” values in this case. These “prior” values then will be

used in the Measurement Update equations.

In the Measurement Update equations, the value of $\hat{x}_k$ which is the estimate of **x** at time **k** need to compute out. Computation on the value of P_k for the **k** **1** *(future)* estimate and $\hat{x}_k$ is carried out as well.

“Posterior values” are also refers to the values that have successfully identify at Measurement Update stage.

Control Algorithm:

Before sending proper AT commands to the AR. Drone 2.0, a Wi-Fi connection must be set up and establish before flight. After successfully connected to the drone, UDP port 5556 is required to open and listen for receiving all of the incoming AT command that will be sent over from PC. To avoid Wi-Fi connection being dropped, it is advised to send AT command at the rate of every 2 seconds.

In order to control the AR. Drone 2.0, AR. Drone 2.0 AT command protocol is implemented and used in this project. All AT commands are sent over PC Wi-Fi connection. These AT commands are in text strings forms and encoded as 8-bit ASCII characters. Every AT commands is ended with a Carriage Return character.

AT commands are starts with three characters "AT*" and followed by command name, an equal sign, a sequence number (usually starts with 1) and lastly, a list of comma-separated command arguments.

One or more AT commands can be fit into one single UDP packet and separated by newline byte (\r 0x0A). However, it is impossible to split one AT command in two or more UDP packets.

Most frequently used AT commands are listed as below:

1. **AT*REF** – Receives an input as an argument. This command is called for take-off, landing, emergency stop and rest.
2. **AT*PCMD** – Receives 4 inputs as arguments. This command is called for move the drone by control the roll, yaw, pitch, and gaz values.
3. **AT*PCDM_MAG** – Receives 7 inputs (flag, roll, pitch, gaz, yaw, psi, and psi accuracy) as arguments. This command is called for move the drone by control the flag, roll, pitch, gaz, yaw, psi, and psi accuracy values. This command support for Absolute control mode.

4. **AT*FTRIM** – Does not receive any argument. This command is called for sets the reference for the horizontal plane. To execute this command correctly, the AR. Drone 2.0 must be sits on a flat ground surface.
5. **AT*CONFIG** – Receives 2 inputs (key and value) as arguments. This command is called for configures the AR. Drone 2.0.
6. **AT*CONFIG_IDS** – Receives 3 inputs (session, user, and application ID) as arguments. This command is called for as unique identifiers for AT*CONFIG commands.
7. **AT*COMWDG** – Does not receive any argument. This command is called for reset the communication watchdog.
8. **AT*LED** – Receives 3 inputs (animation, frequency, and duration) as arguments. This command is called for set LED animation on the drone.
9. **AT*ANIM** – Receives 2 inputs (animation and duration) as arguments. This command is called for set a flight animation on the drone.

A detailed version of all available AT commands can be found and download from Internet (link: http://www.msh-tools.com/ardrone/ARDrone_Developer_Guide.pdf). In AR. Drone open API documentation, all of the AT commands are well explained of the functions.

4.5 Graphic User Interface Design

4.5.1 Free Flight Mode

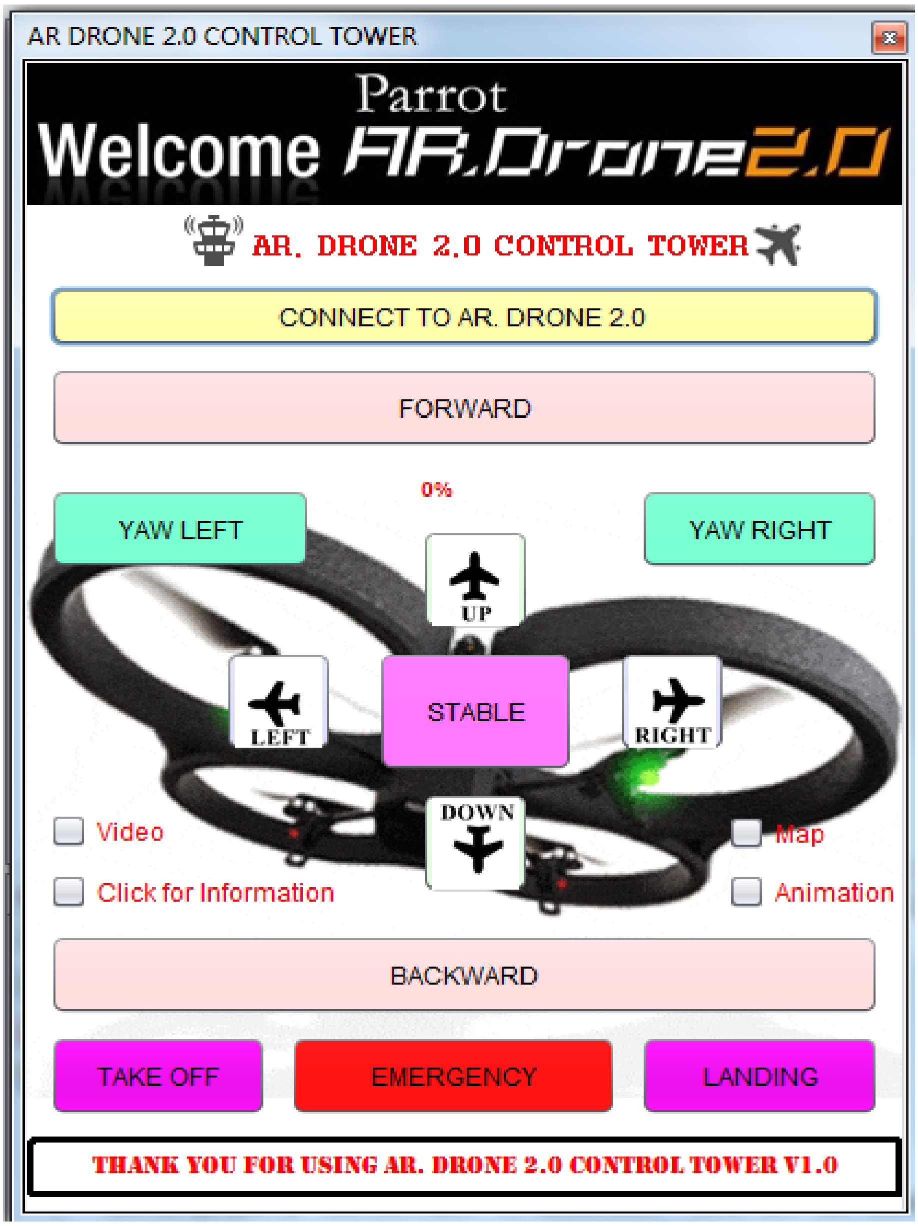

Figure 4-5-1: Free Flight Mode

Function of GUI buttons:

1. **Connect to AR. Drone 2.0 button**: Click to initialize and start connection into AR. Drone Wi-Fi network.
2. **Forward button**: Click to fly the drone toward front direction.
3. **Yaw Left button**: Click to turn left from side to side around the vertical axis.
4. **Yaw Right button**: Click to turn right from side to side around the vertical axis.
5. **Left button**: Click to fly the drone to left position.
6. **Right button**: Click to fly the drone to right position.
7. **Up button**: Click to increase the altitude of AR Drone.
8. **Down button**: Click to decrease the altitude of AR Drone.
9. **Stable button**: Click to stabilize the AR Drone.
10. **Video button**: Click to start and view live video streaming from Parrot AR. Drone 2.0
11. **Click for more information button**: Click to get more detailed information about Parrot AR. Drone 2.0
12. **Map button**: Click to open "Map for Parrot AR. Drone" interface for directional flight.
13. **Animation button**: Click to open "Drone Animation" interface for more advanced drone flight movements.
14. **Backward button**: Click to fly the drone toward back direction.
15. **Take off button**: Click to start the drone's motors and the drone position itself automatically at an altitude approximately of 0.77 meter.
16. **Emergency button**: Click to stop motors rotates instantly and the AR Drone will drop down regardless of its altitude is.
17. **Landing button**: Click to start descends altitude and land on a flat surface.

4.5.2 Live Video Streaming

Figure 4-5-2: Video Streaming from Parrot AR.Drone 2.0

4.5.3 Map for Parrot AR. Drone 2.0

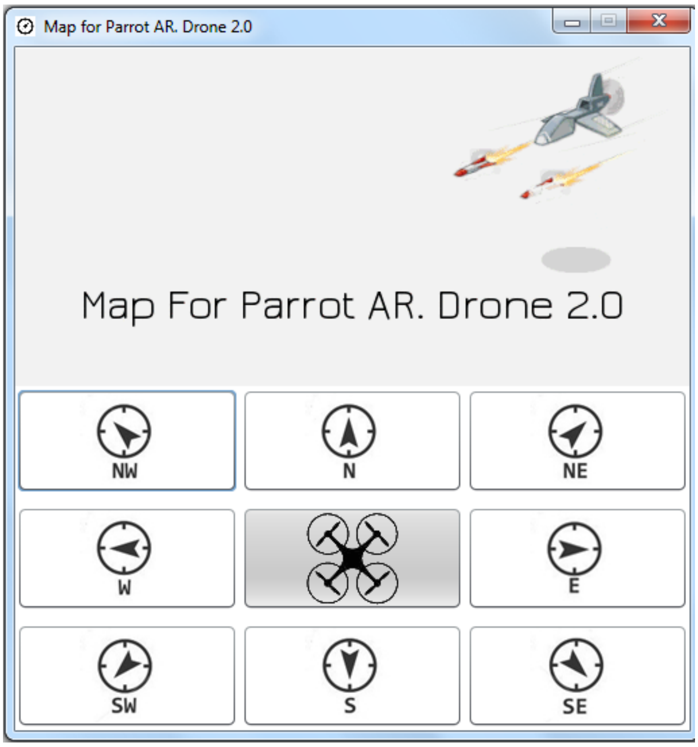

Figure 4-5-3: Map for Parrot AR. Drone 2.0

4.5.4 Parrot AR. Drone Information

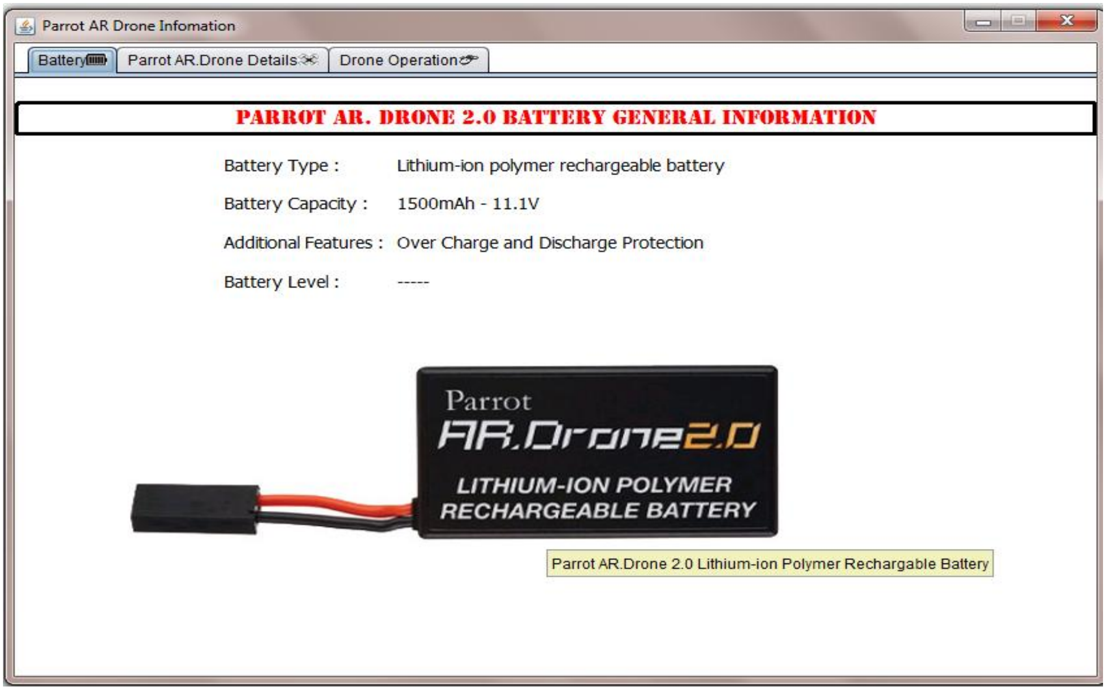

Figure 4-5-4: Parrot AR. Drone Information

4.5.5 Drone Animation

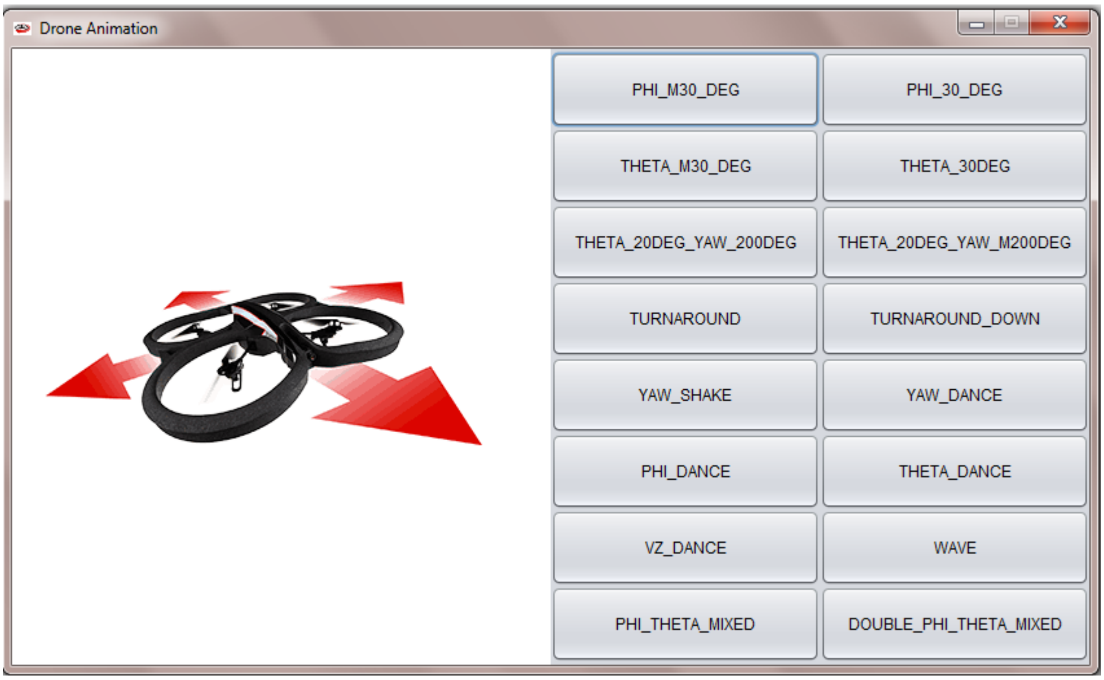

Figure 4-5-5: Drone Animation

4.5.6 Abstract Command Mode

Figure 4-5-6: Abstract Command Mode

4.6 Concluding Remark

In a nutshell, system architecture is explained and illustrated with a diagram to make reader more understand and clear of how the system works. In addition, function modules in the system also listed out and describe the respective functions. At the same, there is a complete system flow chart is attached in Section 4.3. The system flow chart clearly gives reader a clearer picture on the flow of the proposed system. As for algorithm design section, further explanation on the system flows is made as well. Lastly, some of the graphical user interface (GUI) photos are attached in section 4.5. Functions of GUI buttons in Free Flight Mode are discussed under section 4.5.1.

Chapter 5 – System Implementation

5.1 Hardware Setup

As discussed in the Chapter 3 under Hardware section, Parrot AR. Drone 2.0 and a personal laptop are being applied in this project. Parrot AR. Drone 2.0 acts as flying remote control devices which constantly receive abstract command from user whilst laptop will be used as controller to issue abstract command to Parrot AR. Drone.

Before get started, let’s check the contents of the Parrot AR. Drone 2.0 packaging. There are total of 8 items inside the Parrot AR. Drone 2.0 box as shown in figure below. Those are 1 unit of Parrot AR. Drone 2.0 main body, 1 unit of indoor hull protector (for indoor flight), 1 unit of outdoor hull protector (for outdoor flight), 2 units of Lithium-Polymer batteries (1,000 mAh and 1,500 mAh), 1 unit of battery AC charger with International Adaptors, 1 unit of multilingual instruction manual and decal stickers for augmented reality gaming.

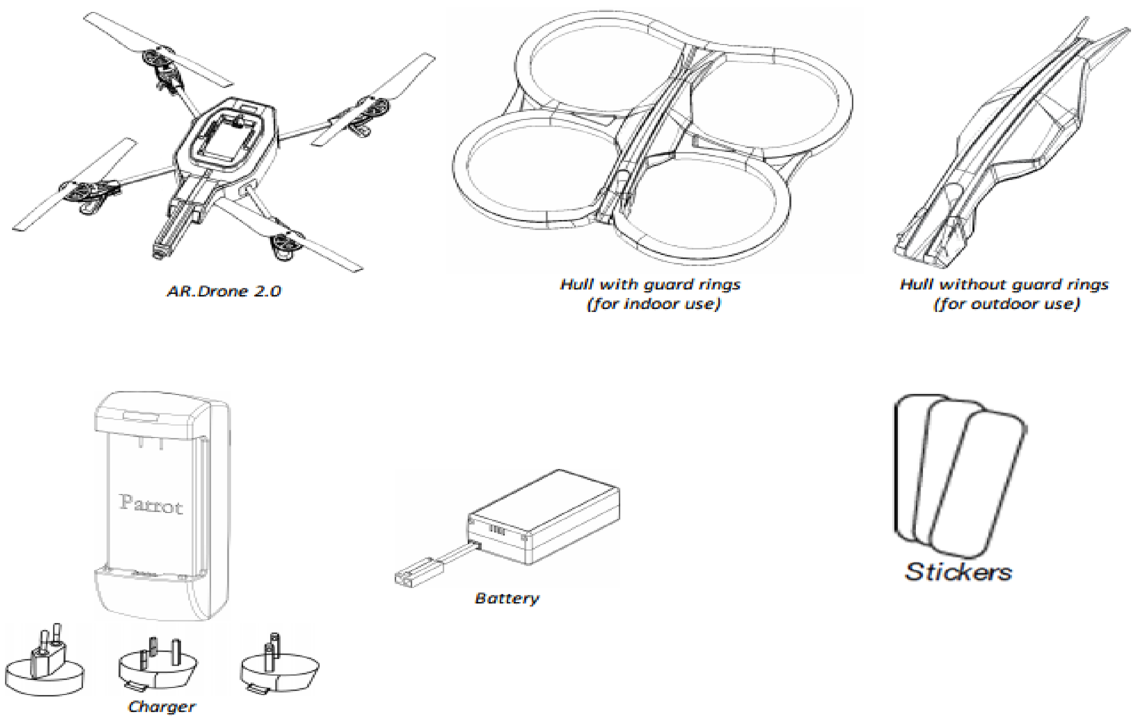

Figure 5-1-1: Contents of Parrot AR. Drone 2.0 Packaging

Slowly take out the parrot AR. Drone 2.0 main body from the box and place on a flat surface.

Kindly remind that to charge the battery as soon as unbox the Parrot AR. Drone 2.0 package. This is a very important step to follow to ensure the battery level always get ready for safe flight. The charging process should takes approximately 1.5 hours. The duration of recharging time depends on how discharged the battery was.

Figure 5-1-2 is an example showing the battery is performing recharging process and the battery recharger should illuminates steady red lights.

Figure 5-1-2: Recharging battery process in progress – red color LED

Figure 5-1-3: Recharging battery done – light green color LED

The battery recharger illuminates in light green color indicate that the battery is fully recharged as shown in figure 5-1-3. Unplug the battery from battery recharger and insert the battery slowly into the appropriate compartment. Plug in the battery and verify that it is correctly secured by using the attachment mechanism as shown in figure 5-1-4.

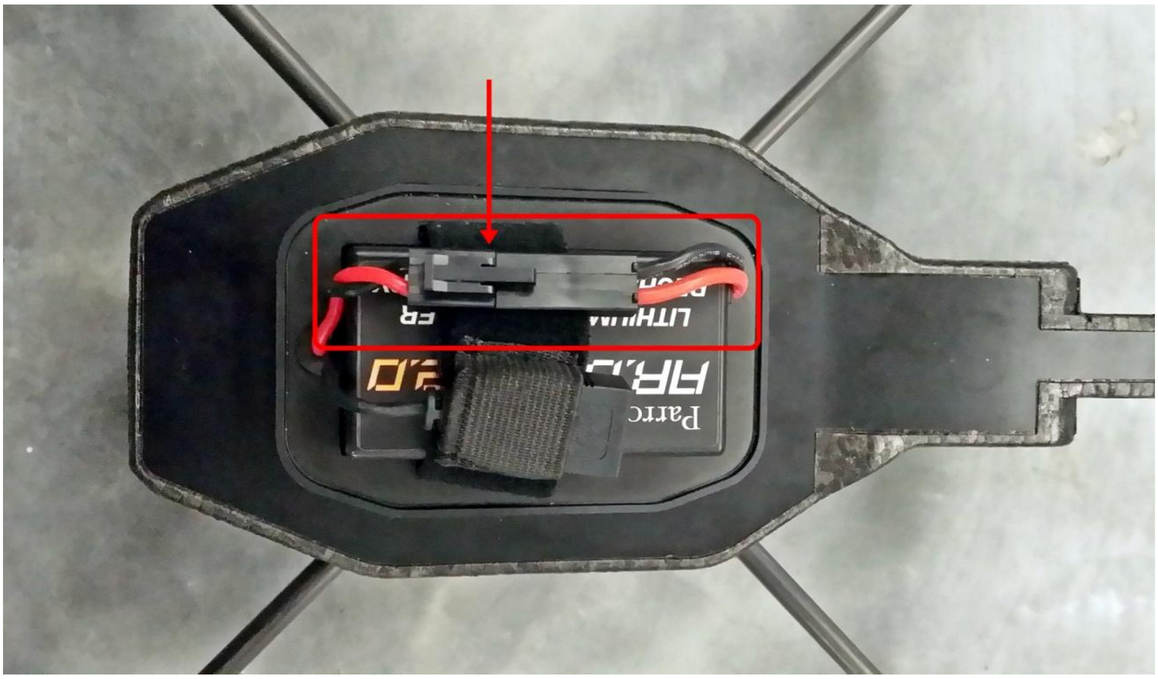

Figure 5-1-4: Battery attachment mechanism

Wait for few seconds. The drone should start to operate by rotating 4 motor blades respectively follow one by one (called as rotor test). After that, all of the 4 LED lights (light emitting diodes) which are located near to motor and propellers are should illuminating in light green colors now as shown below.

Figure 5-1-5: 4 motor LED green lights indicates startup is okay

There are 2 different types of LED on the Parrot AR. Drone.

(i) 4 motor LEDs located close to propellers
(ii) a system LED located on the belly of the AR. Drone

If the motor LED lights are showing different colors – flashing red or orange colors as shown in figure 5-1-6, it indicates something went wrong with the device.

User need to disconnect and reconnect the battery. Wait until the system and motor LED turns green color again.

If the problem persists after few attempts, please contact lab assistance immediately.

Figure 5-1-6: 4 motor LEDs shows red lights

According to official Parrot AR. Drone 2.0 user guide UK version (link: http://ardrone2.parrot.com/media/uploads/support_ardrone_1/ar.drone_user-guide_uk.pdf), different LED lighting patterns advise the user what is happening. This can be summarized in the following table:

Abstract Command to Control Parrot AR Drone 2.0

Motor LED Behavior	Meaning
1. 4 LEDs are showing red	Power is connected (Start-up phase) or drone entered into Emergency mode (E.G. Too much angle emergency & battery low emergency)
2. Each LED flashes red, one after the other	Motors are starting up
3. 4 LEDs flashing green	AR. Drone is taking off or landing at this moment
4. 2 frontal LEDs are green while 2 rear LEDs are red	AR. Drone is flying at this moment. These colors pattern make it easy for pilot to distinguish the front and rear of the AR. Drone
5. 4 LEDs flashing red and green	Memory of the AR. Drone has been deleted by pressing on the Unpair button

Table 5-1-7: Motor LED behavior table

User should now checks for system LED. If the system LED is showing in green color as shown in figure 5-1-8, user could continue to perform next action. However, if after 20 seconds start-up process, the system LED is either flash red or orange color, user should disconnect and reconnect the battery again.

Figure 5-1-8: System LED shows green color

Once the AR Drone powers up without any problem, it will start to broadcast a Wi-Fi hotspot that user could connect to as shown in figure 5-1-9. The Wi-Fi hotspot usually starts with ardrone2_ and followed by a random with 6 digits number. Connect to the ardrone Wi-Fi network. Wait for few seconds until it successfully connect into the drone Wi-Fi network.

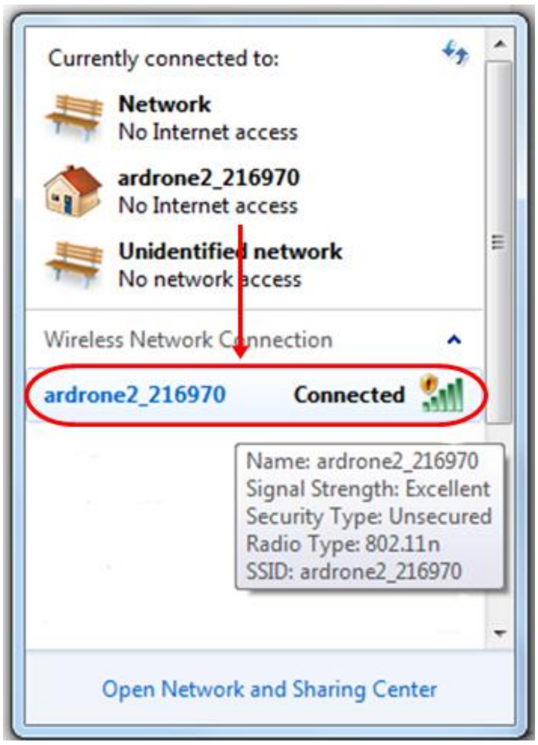

Figure 5-1-9: Parrot AR. Drone 2.0 Wi-Fi hotspot

Launch the Abstract Command controller application in NetBeans IDE software.

5.2 Software Setup

The Parrot AR. Drone 2.0 is now connected with the personal laptop. All the communications between AR. Drone and personal laptop are done through Wi-Fi network. In this project, Java IDE and compiler is needed for doing all the necessary Java coding as well as serves as intermediary software that sitting between user and the AR. Drone.

5.3 Setting and Configuration

1. NetBeans IDE

Free and open-source Java IDE - NetBeans is downloaded and use in this study. The NetBeans IDE offers and supports a wide variety of programming languages (for example, Java, JavaScript, JavaFX, PHP, C/C++, etc.).

NetBeans enable programmer to develop solutions more efficiently and quickly by taking the benefits of strengths of the Java platform.

Log on to the official NetBeans website as shown in figure 5-3-1 below and search for Download Section: https://netbeans.org/downloads/index.html.

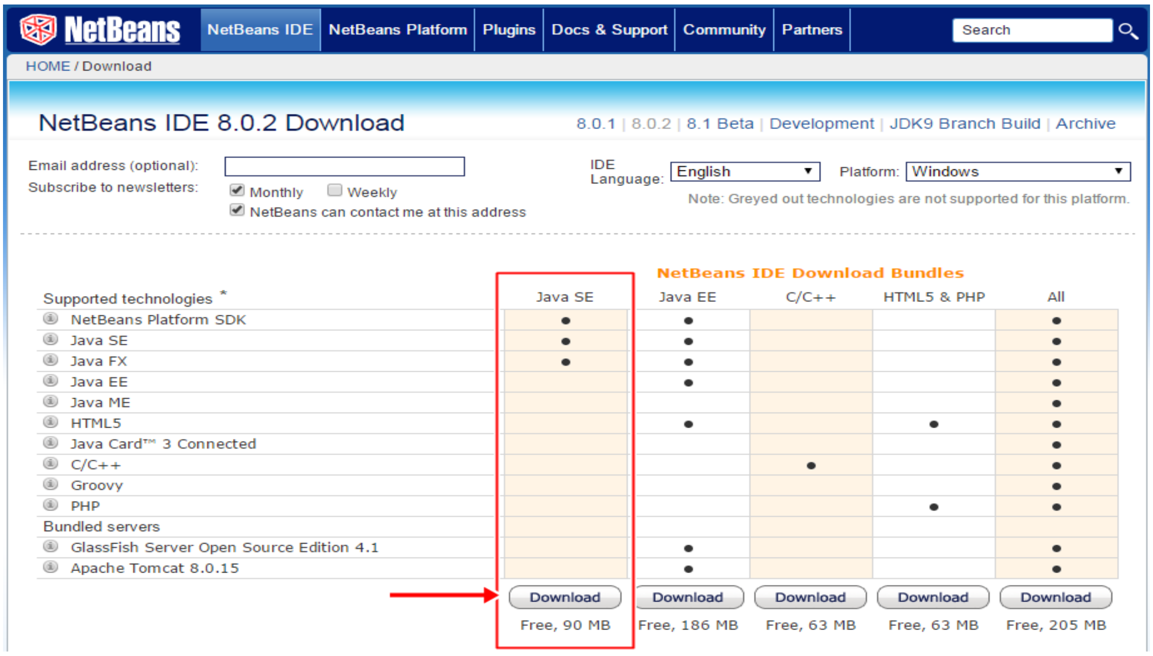

Figure 5-3-1: NetBeans IDE Download Website

1. Download the Java SE IDE version. Upon the completion of downloading NetBeans IDE, install the NetBean Java SE into appreciate computer drive (in this case, follow the default installation path - C:\Program Files\NetBeans 8.0.2).
2. Wait for NetBeans installation progress to complete.
3. Java GUI and fully functional abstract command modules can be started to develop using the NetBeans IDE now.

2. Javadrone API

1. Log on to https://code.google.com/p/javadrone/downloads/list as shown in figure below.

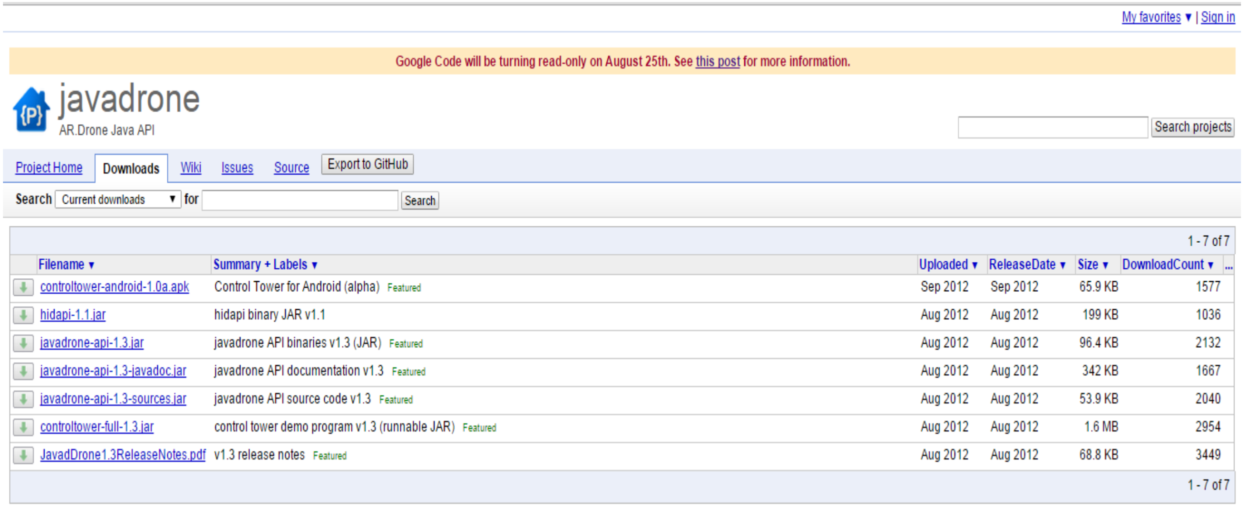

Figure 5-3-2: Javadrone API download website

2. Under the Search section, select “All downloads” and press Search button as shown in figure below.

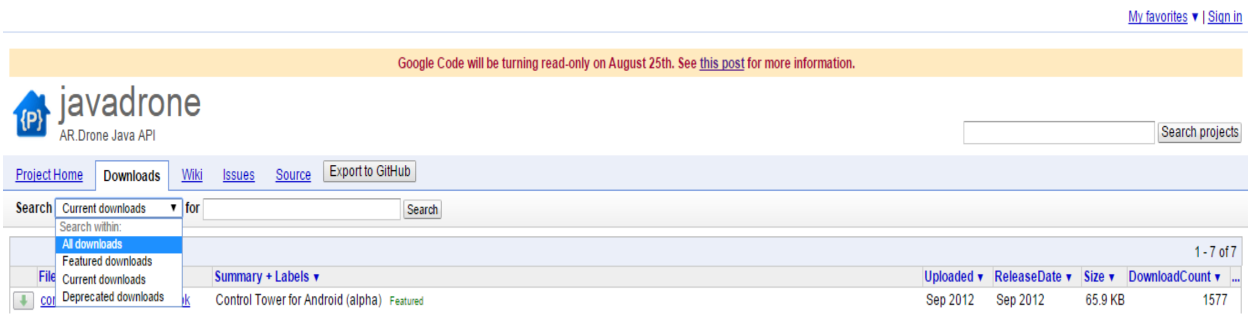

Figure 5-3-3: Select “All downloads”

3. Click “**javadrone-api-1.3-sources.jar**” and download the file. The file size for the jar file is 53.9KB. Verify and double check before proceed to download.

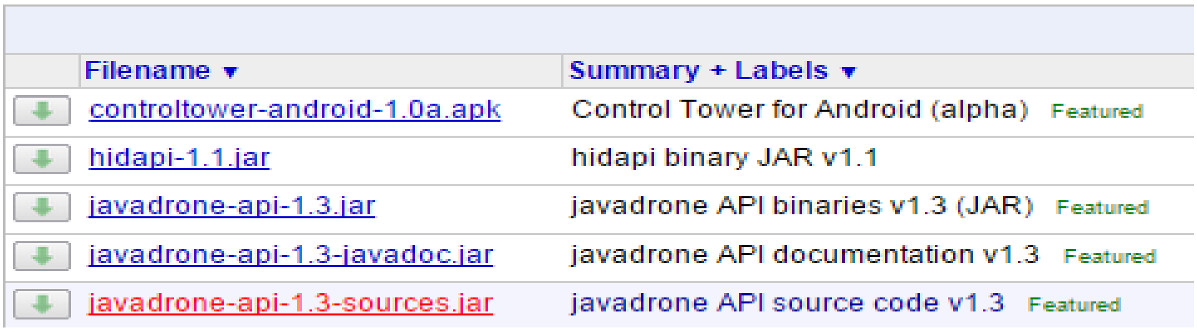

Figure 5-3-4: Click "javadrone-api-1.3-sources.jar"

4. Wait for few minutes until the download process complete.
5. Navigate to the download folder.
6. Right click the "**javadrone-api-1.3-sources.jar**" file and extract to Desktop.
7. Open NetBeans IDE.
8. In the NetBeans IDE, go to File → Open Project… → Select "**javadrone-9f63e04e8a13**" → Select "Open Project" button.

The javadrone API and all the necessary library files are now installed into NetBeans IDE successfully.

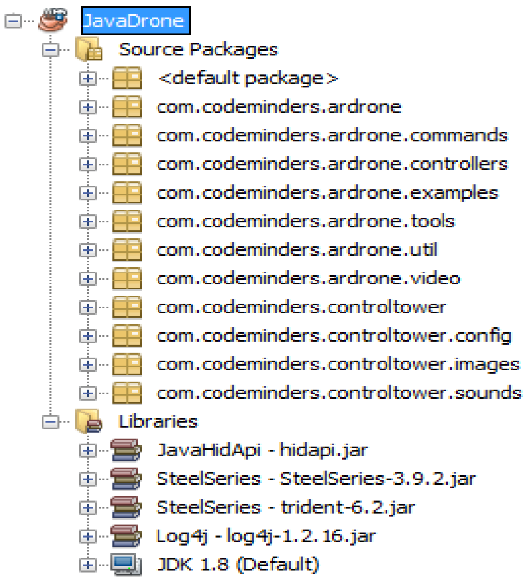

Figure 5-3-5: Javadrone API and library files installation complete

3. JFreeChart

JFreeChart was started to develop since 15 years ago by David Gilbert. It is a widely used chart library for Java. JFreeChart is a free and open source Java chart library. It allows software developers to display various types of charts in more professional approach. Flexibility in design of charts provides software developers to easily extend the chart designing formats with extensive feature sets that consists of a well-documented API.

In this project, JFreeChart application is adopted and used in the function of "Landing". After user pressing "Landing" button, the drone will land and prompt user to select whether they wishes to display "Battery Graphic" or "Altitude Graphic". A simple graphic will be shown to user after pressing either one of the option.

1. Log on to http://sourceforge.net/projects/jfreechart/files/ as shown below.

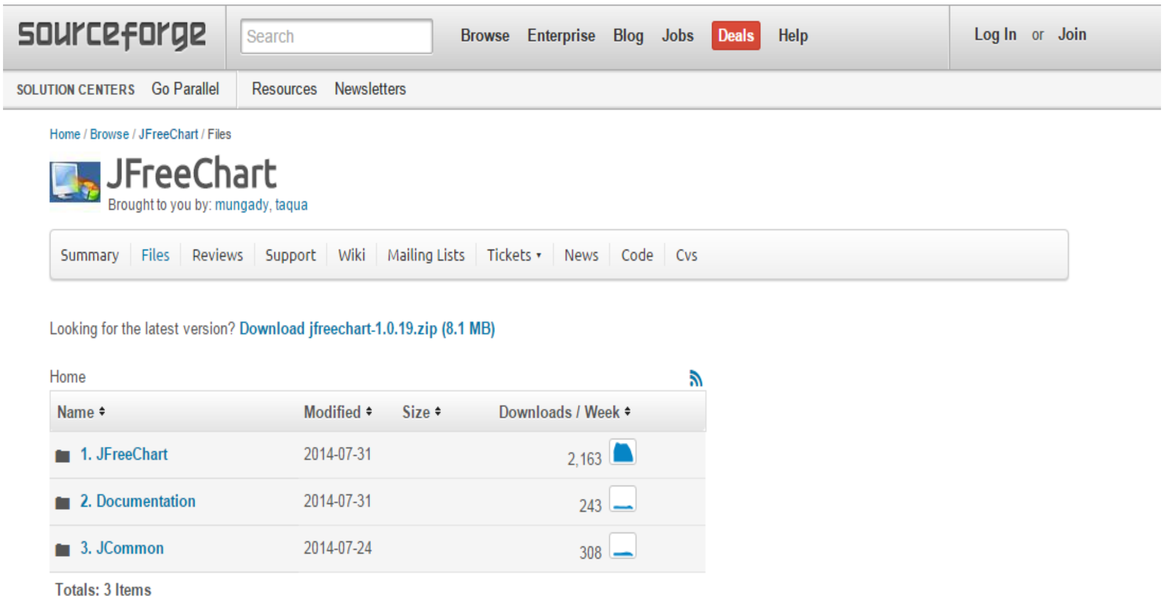

Figure 5-3-6: JFreeChart

1. Click on "**Download jfreechart-1.0.19.zip (8.1 MB)**" to start downloading process.
2. Wait for few minutes until the download process complete.

3. Navigate to the download folder.
4. Right click the “**jfreechart-1.0.19.zip**” file and extract to Desktop.
5. Open NetBeans IDE.
6. Right click “Java Project” → Select “Properties” → Select “Libraries” → Select “Add JAR/Folder” → Access into “jfreechart-1.0.19\lib” → Select all JAR files.
7. Press “Open”.

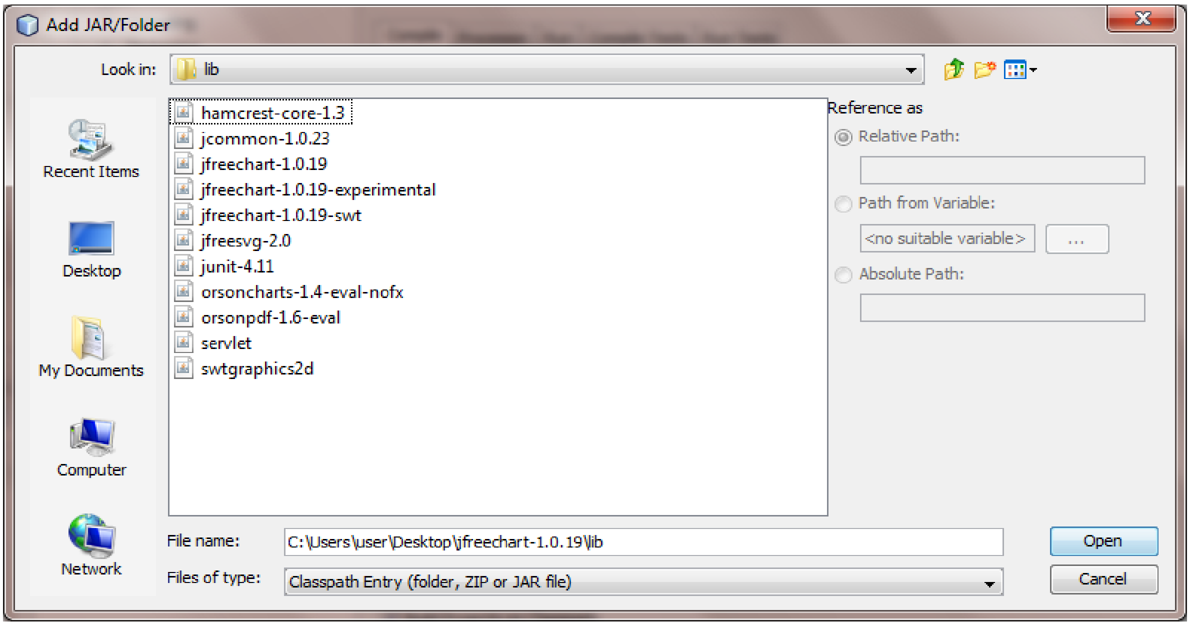

Figure 5-3-7: Add JAR/Folder in NetBeans (Import JFreeChart JAR files)

If successfully inserted into Project, all the selected JFreeChart JAR files should listed out under “Libraries” as shown below.

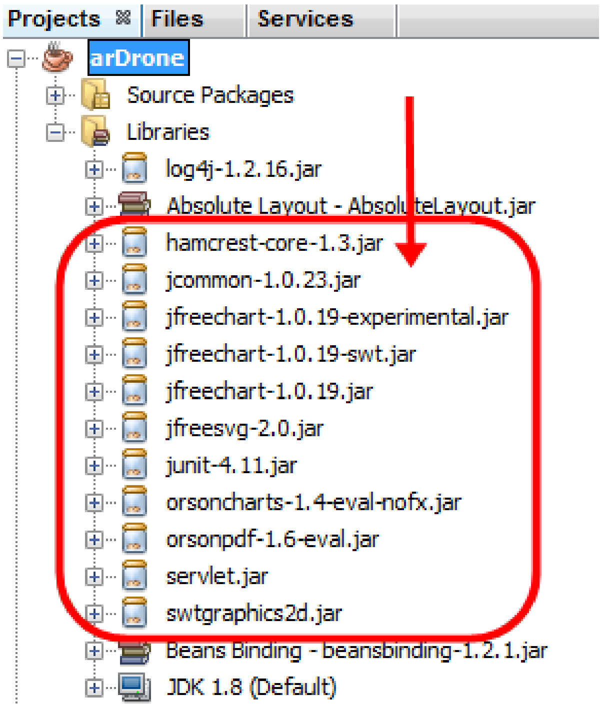

Figure 5-3-8: JFreeChart JAR files are successfully added into Project

5.4 System Operation

1. Double click on “NetBeans IDE” icon. The NetBeans IDE takes few seconds to launch. Wait patiently as the NetBeans IDE busying with loading modules and perform background scanning of projects.
2. Now the NetBeans IDE is ready to use.
3. In order to run the project, press F6 shortcut key on keyboard or click the “Play” icon in green color on top of the menu bar as shown in figure 5-4-1 below. The NetBeans IDE will start to compile and run the project automatically.

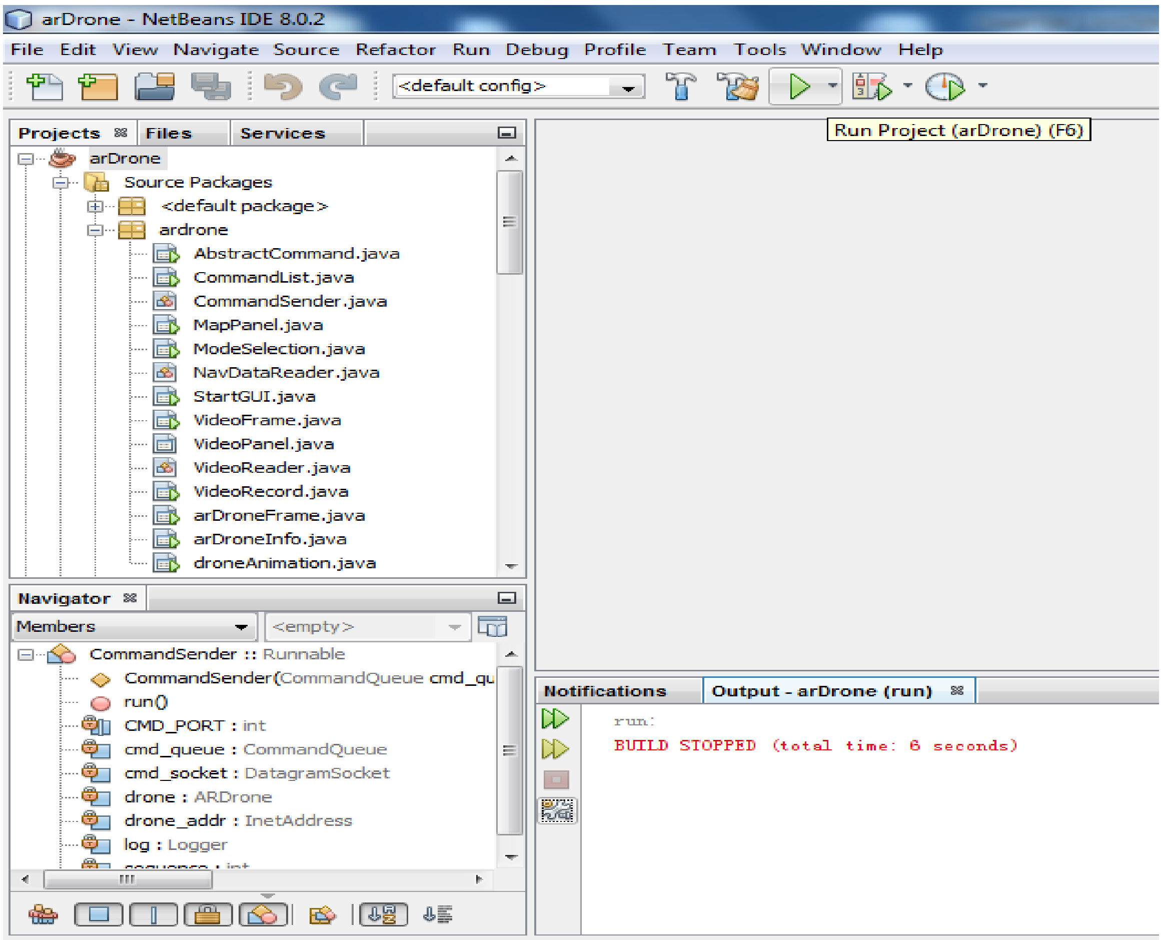

Figure 5-4-1: NetBeans IDE

Figure 5-4-2: Parrot AR. Drone 2.0 splash screen

Parrot AR. Drone 2.0 splash screen will appear for few seconds.

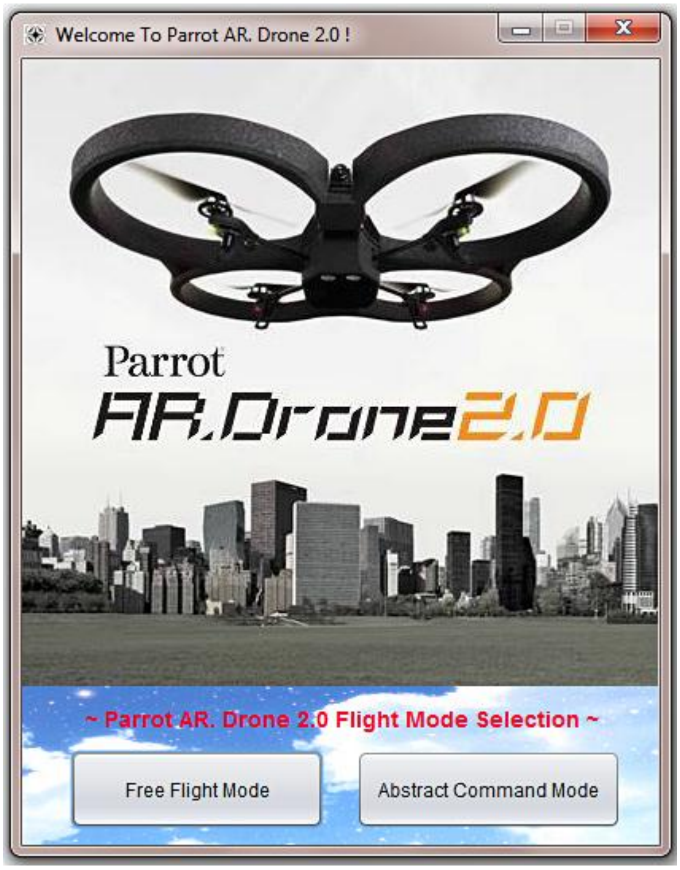

Figure 5-4-3: Parrot AR. Drone 2.0 mode selection

There are 2 flight modes to select:

I. Free Flight Mode
II. Abstract Command Mode

Free Flight Mode:

In the Free Flight Mode, user is allowed to click on each individual different buttons to perform the corresponding action. User can perform basic simple flight maneuvers (e.g. fly up, fly down, fly forward, fly backward, etc.) in this mode.

Only one abstract command is being sent at a particular time in the Free Flight Mode. For example, when user click "TAKE OFF" button, the program will send out "takeoff" command to the Parrot AR. Drone. The AR. Drone receives the "takeoff" command from laptop and it will start to lift off from ground. After taking off successfully, the drone hovers at the approximate height of 0.77m.

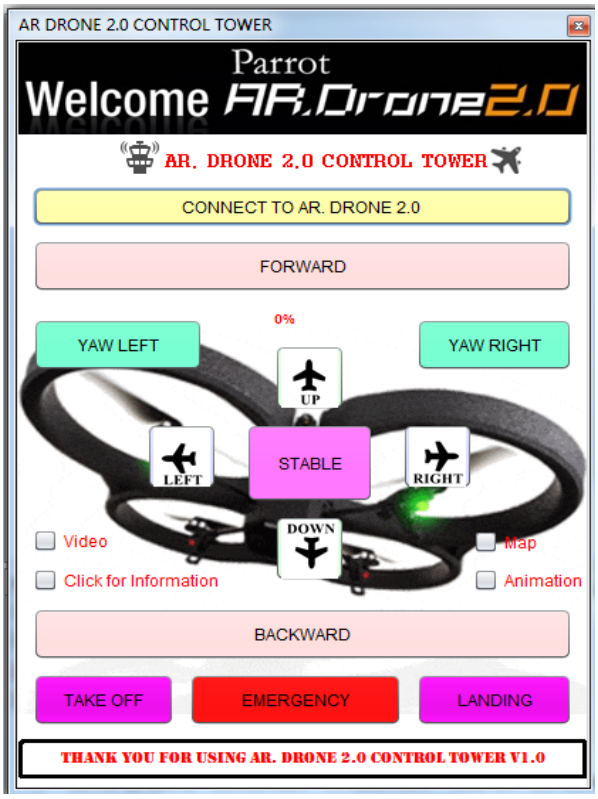

Figure 5-4-4: Free Flight Mode

Connect to the Parrot AR. Drone

Before flight, user is required to connect into drone by clicking the "CONNECT TO AR.DRONE 2.0" button as shown in figure 5-4-5 below. This is a must "To-Do" action for user before flight so that all of the abstract commands could send out to AR. Drone as well as receive live feedback from the drone.

The "0%" value at the bottom of "FORWARD" button indicates the initial value of drone battery. This is the default value for drone battery before connect into drone network.

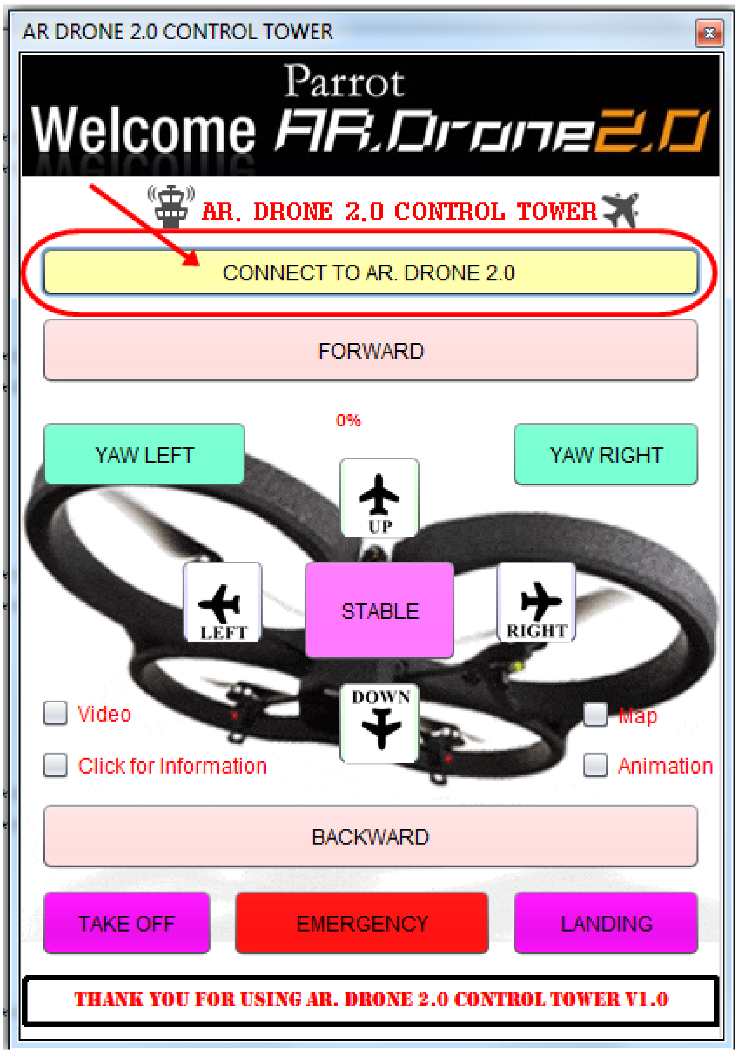

Figure 5-4-5: Press CONNECT TO AR. DRONE 2.0 button

After successfully connected into AR. Drone, the programs will automatically pop up a window called “Previous Command List & Drone Data”.

The “0%” value will changes from “0%” to the remaining battery level that obtain from the drone upon successfully connect into the drone. In the example below, the client side (laptop) is connected into drone network and receives the updated remaining battery level – 83%.

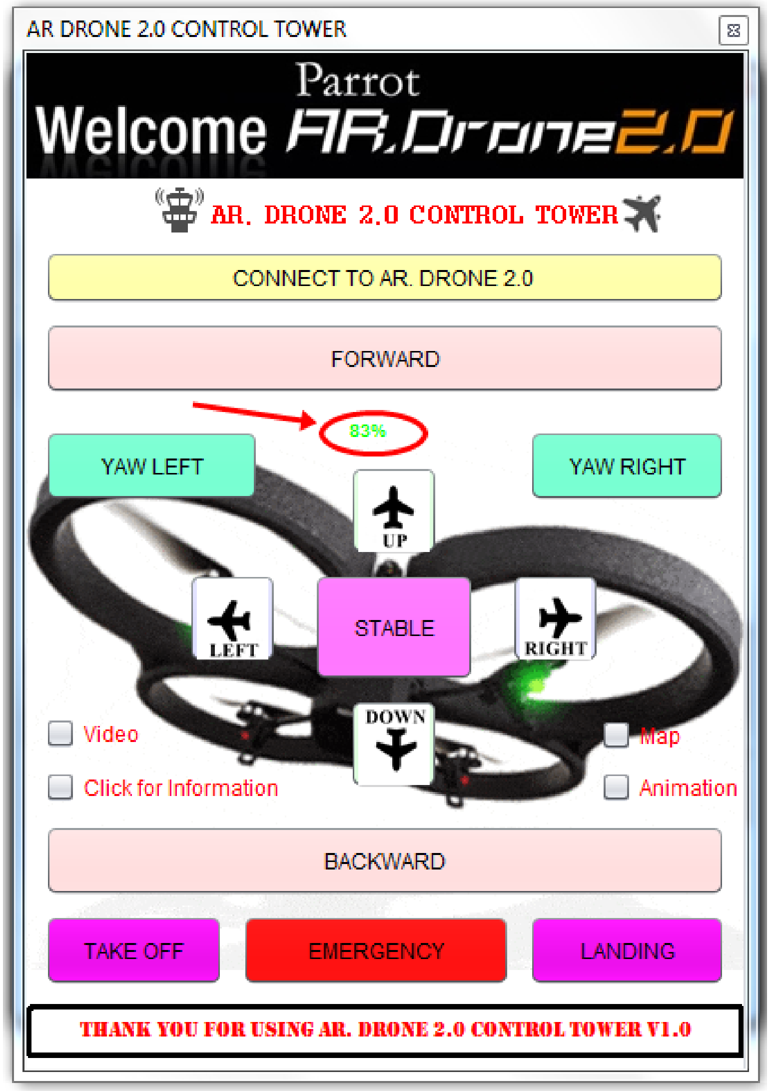

Figure 5-4-6: Default battery value “0%” changes after receiving drone navigation value

Under "Previous Command List & Drone Data" window as shown in figure 5-4-7 below, user can clearly check what the abstract commands that they were sent before. Important live updated drone navigation data are also being shown here. User can view all the essential flight data values such as remaining battery level, altitude level, pitch value, roll value, yaw value, and etc.

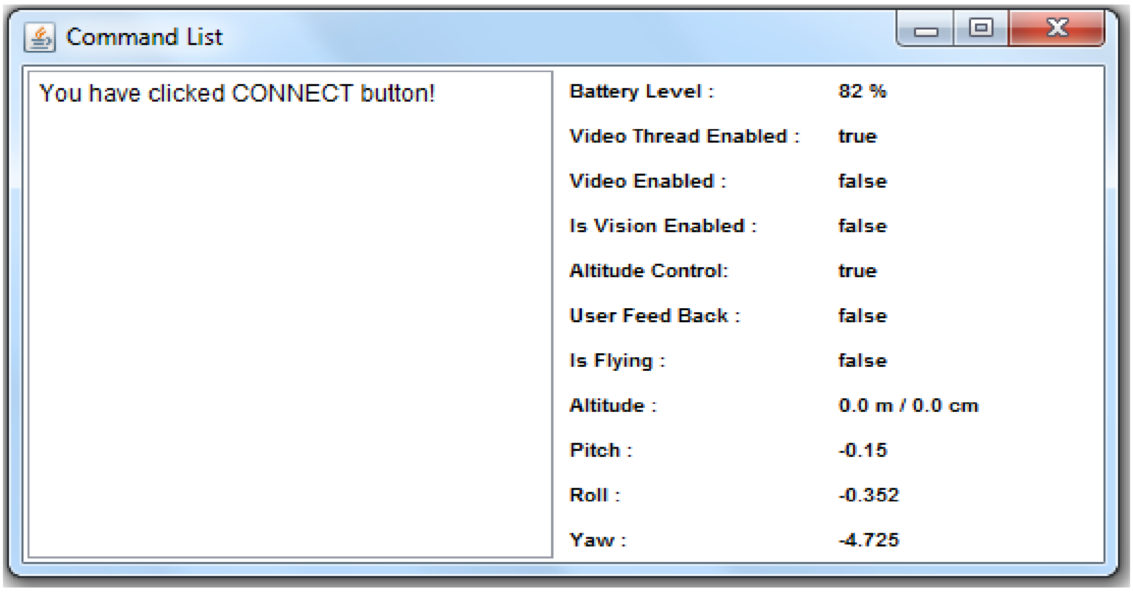

Figure 5-4-7: Previous command list & drone data interface

When the drone battery level lower than 20%, alarm sounds will be sounded continuously and a warning message will be displayed to notify user. The drone land automatically and user needs to recharge the drone battery or insert a new battery for the next flight.

Figure 5-4-8: Low battery warning message

Live camera viewing function also included in this program. By clicking "Video" box as shown in figure 5-4-8, user can stream live frontal camera viewing from drone onto laptop while on ground or during flight. The real-time video feed from Parrot AR. Drone is directly stream to laptop so that user can see images as if they were in the pilot's seat.

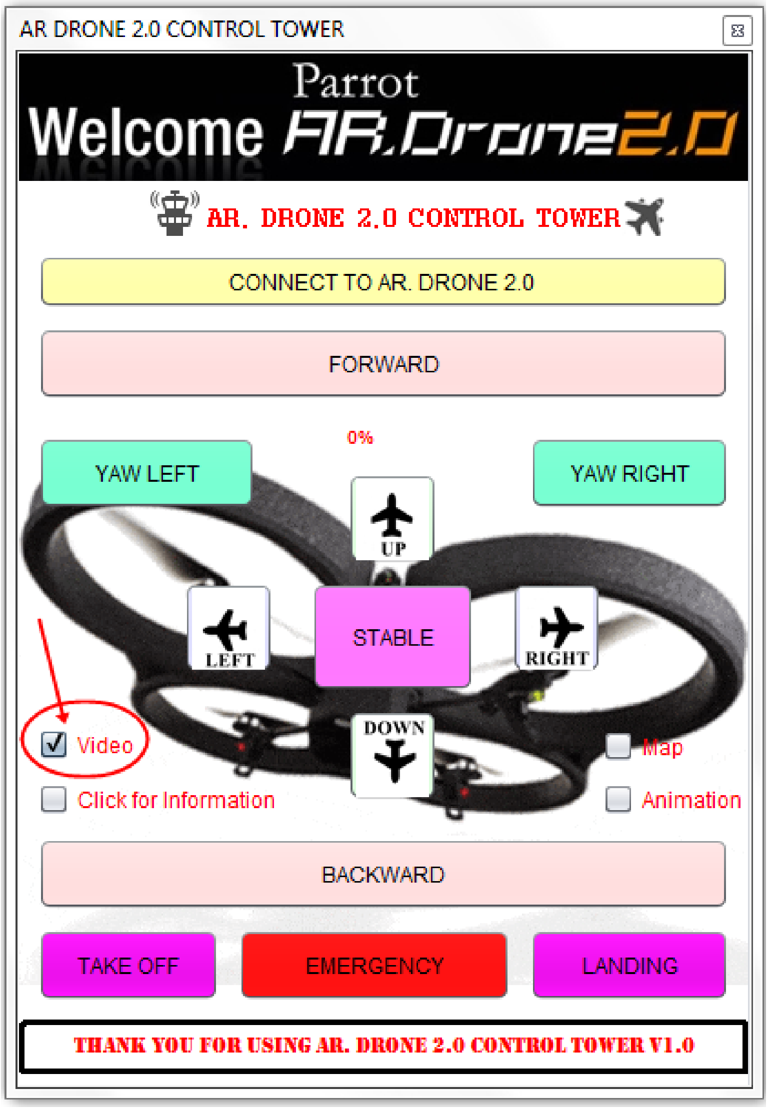

Figure 5-4-9: Click "Video" box for streaming live video

Wait for few seconds. The drone starts to transmit live video streaming to laptop. Figure 5-4-10 below display the live video for user to view.

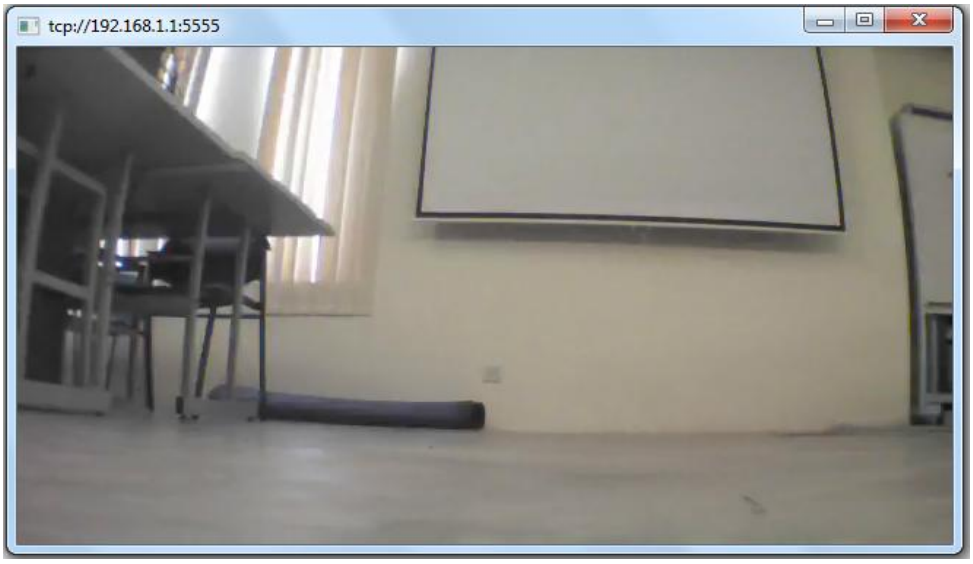

Figure 5-4-10: Live video viewing from drone to laptop

There is another window called "Screen Recorder & Capture" as shown in figure 5-4-11 below. This allows user to:

(i) Record live video streaming from the drone
(ii) Play recorded video
(iii) Convert recorded video from default video file format (.cap) to QuickTime Movie (.mov) format
(iv) Screen capture of live streaming video

Record live video streaming from the drone

Figure 5-4-11: Screen Recorder & Capture

If user wishes to record down the live video streaming from the drone, press on the "Video Recorder" button. The program will pop up a new window as shown in figure

5-4-12 and it is get ready to record. User can continue to press “Start Recording” button for recording video from the drone.

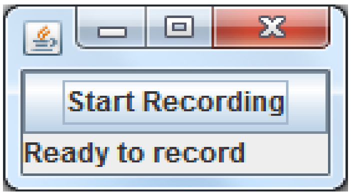

Figure 5-4-12: Ready to record

After pressing the “Start Recording” button, the program will starts to record down the live video streaming.

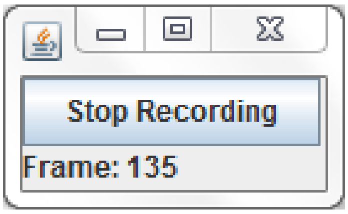

Figure 5-4-13: Start to record

When user wants to stop recording, user can press on the “Stop Recording” button. After terminated the recording process, the program will prompt user where to store the recorded video as shown below.

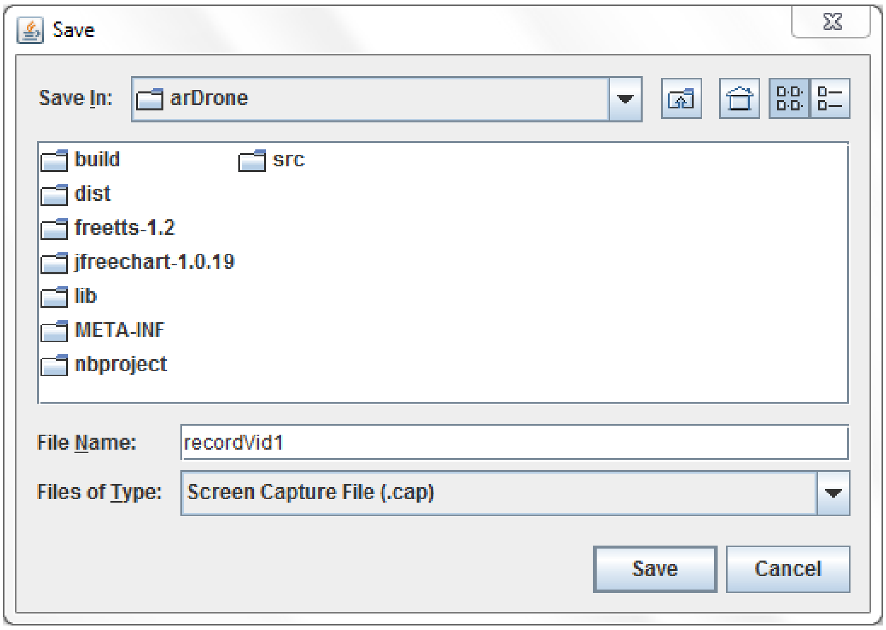

Figure 5-4-14: Prompt user where to store the recorded video

In this example, the recorded video saves inside "arDrone" folder and the file name of the recorded video set as "recordVid1".

The recorded video saved in .cap format.

Play recorded video

User can play the recorded video after flight. To view the recorded video, press "Video Player" button as shown below.

Figure 5-4-15: Video Player

After that, new windows named "Screen Player" will be display. Click on "Open Recording" button.

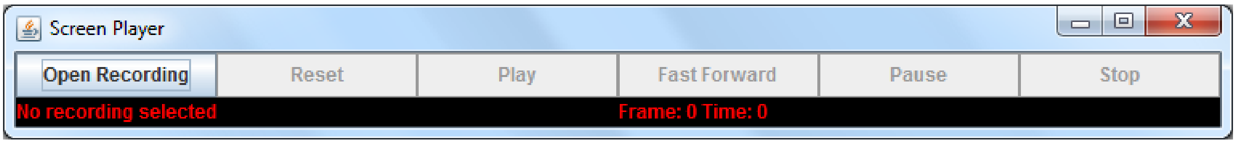

Figure 5-4-16: Screen Player windows

Navigate to the folder where the recorded video stored. Click on “recordVid1.cap” and press “Open” button to continue as shown below.

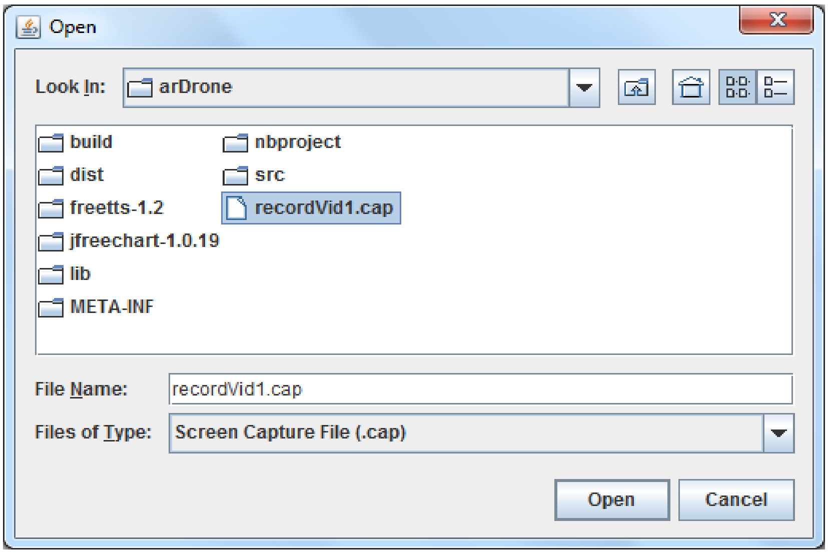

Figure 5-4-17: Open recorded video

Now the video is loaded and waiting to play. Click on “Play” button for viewing.

Figure 5-4-18: Ready to play the recorded video

This is an example of the recorded video. User can pause, stop or reset video as required.

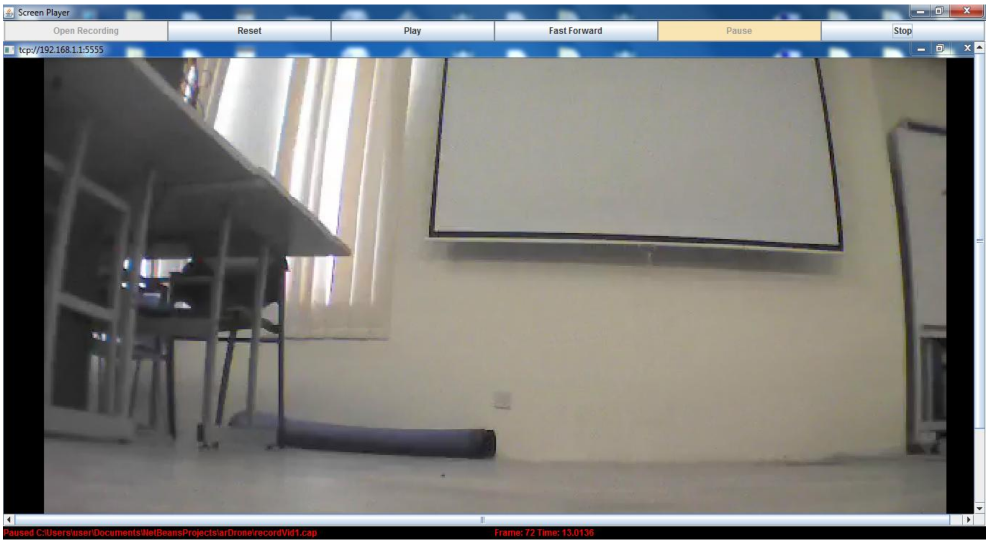

Figure 5-4-19: Playing recorded video

<u>Convert recorded video from default video file format (.cap) to QuickTime Movie (.mov) format</u>

Press on the "Video Converter" button. Wait patiently as video starts to convert. The converted video will display on same folder where user save the recorded video.

Figure 5-4-20: Video Converter

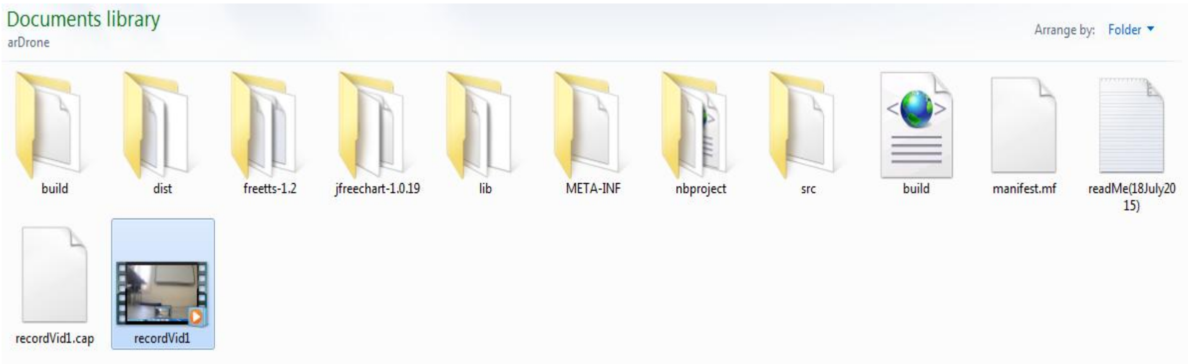

Figure 5-4-21: Converted video

Screen capture of live streaming video

If user wishes to capture the screenshot of the live streaming video, press on "Screen Capture" button as shown below. The screen captured photo will be saved and shown on Desktop. The file name should be "screenshot.jpg" and it is a JPEG image.

Figure 5-4-22: Screen Capture

User could obtain more detailed Parrot AR. Drone general information by clicking "Click for Information" box as shown in figure 5-4-23.

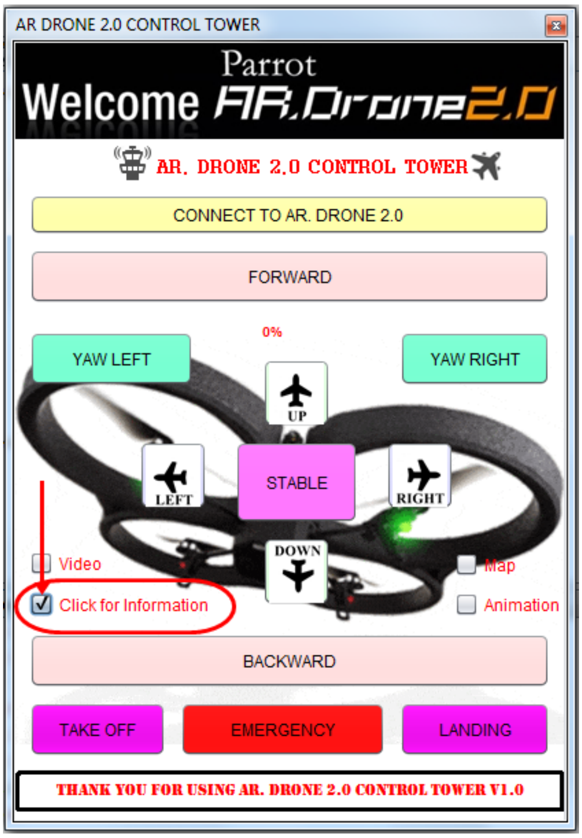

Figure 5-4-23: Click "Click for Information" box

The program will pop up a new window called "Parrot AR. Drone Information".

There are total of 3 sub-windows. Those are: Battery, Parrot AR. Drone Details and Drone Operation.

By default, general information of drone battery is shown after user click the "Click for Information" box.

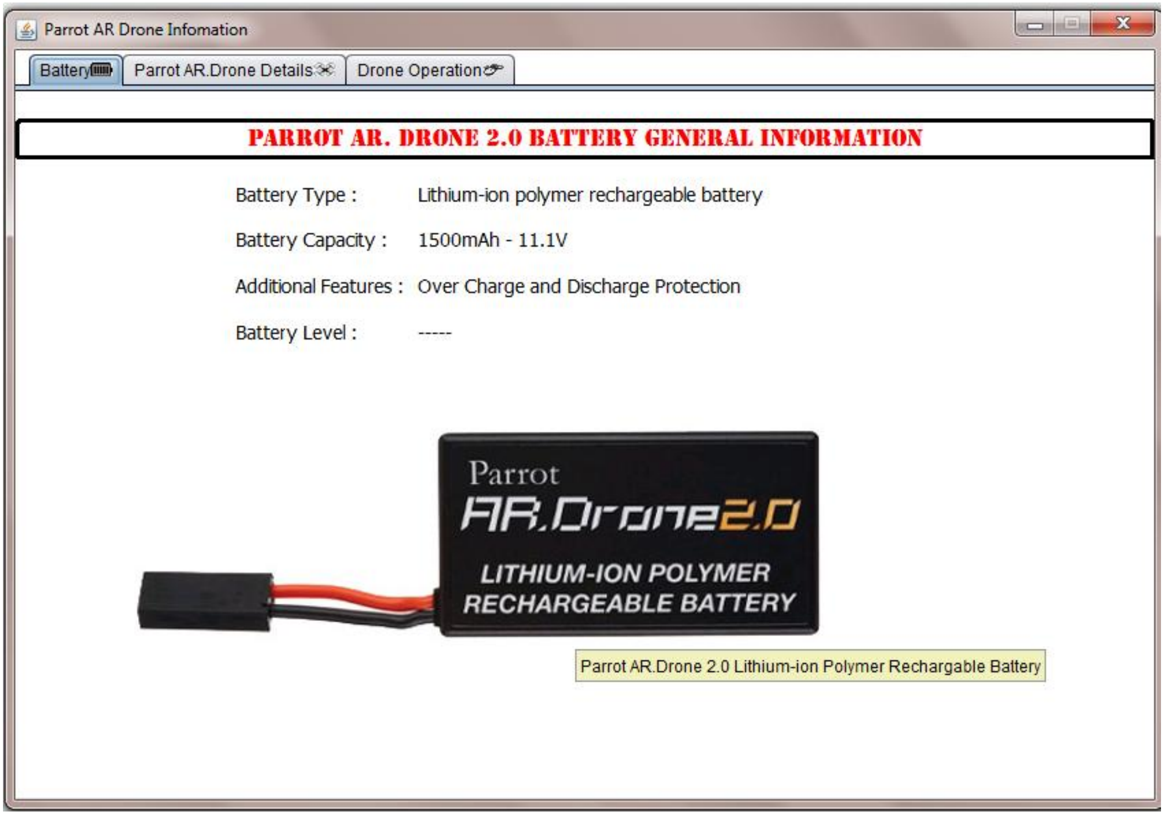

Figure 5-4-24: Parrot AR. Drone 2.0 Battery General Information

If user wishes to order new drone battery from official online store, user can click on the battery image as shown in figure 5-4-24 above and the program will pop up a confirmation message to user. If user want to continue with purchase of new drone battery, press "Yes" button to continue else press "No" to cancel.

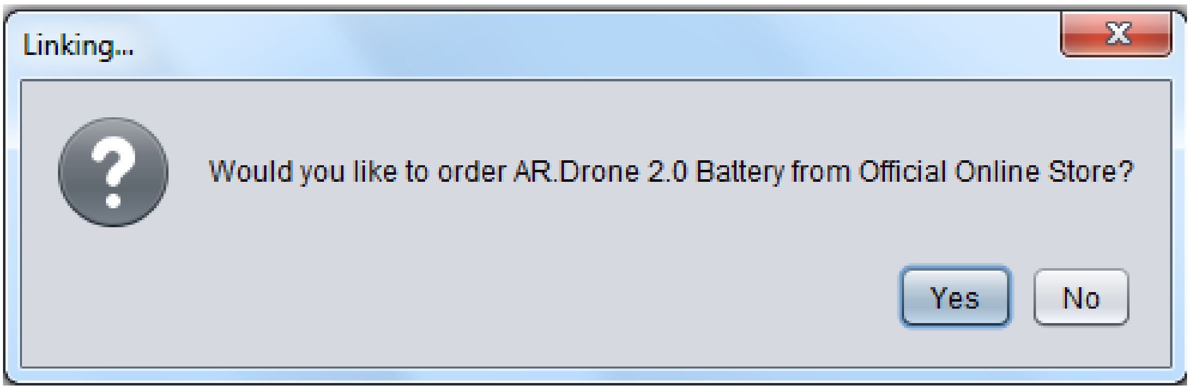

Figure 5-4-25: Confirmation message to user before proceed to official web store

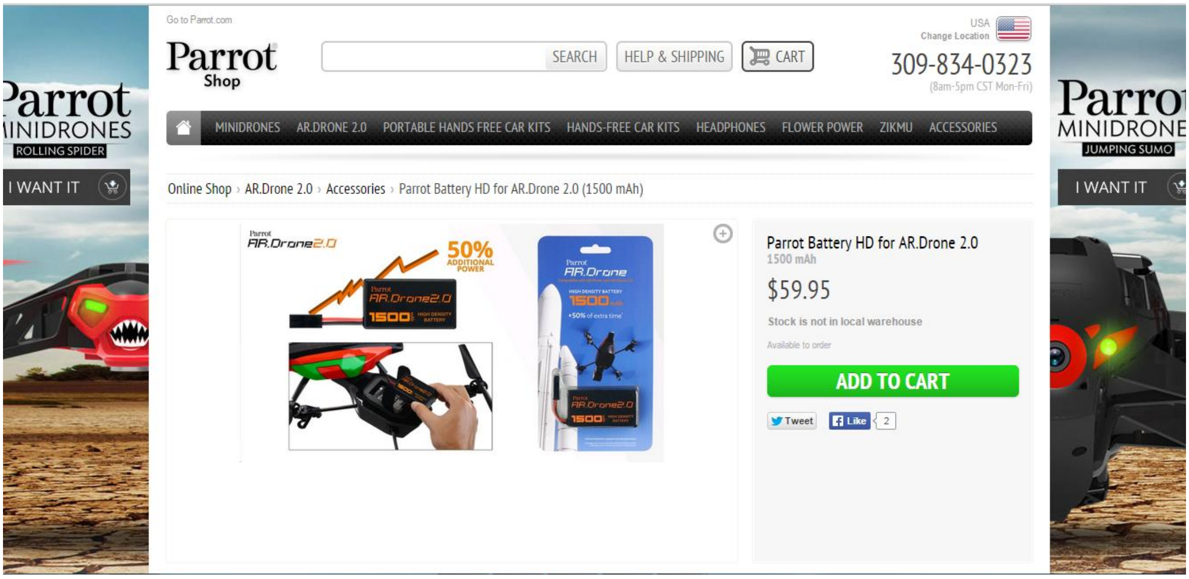

Figure 5-4-26: Official Parrot Online Shop

Abstract Command to Control Parrot AR Drone 2.0

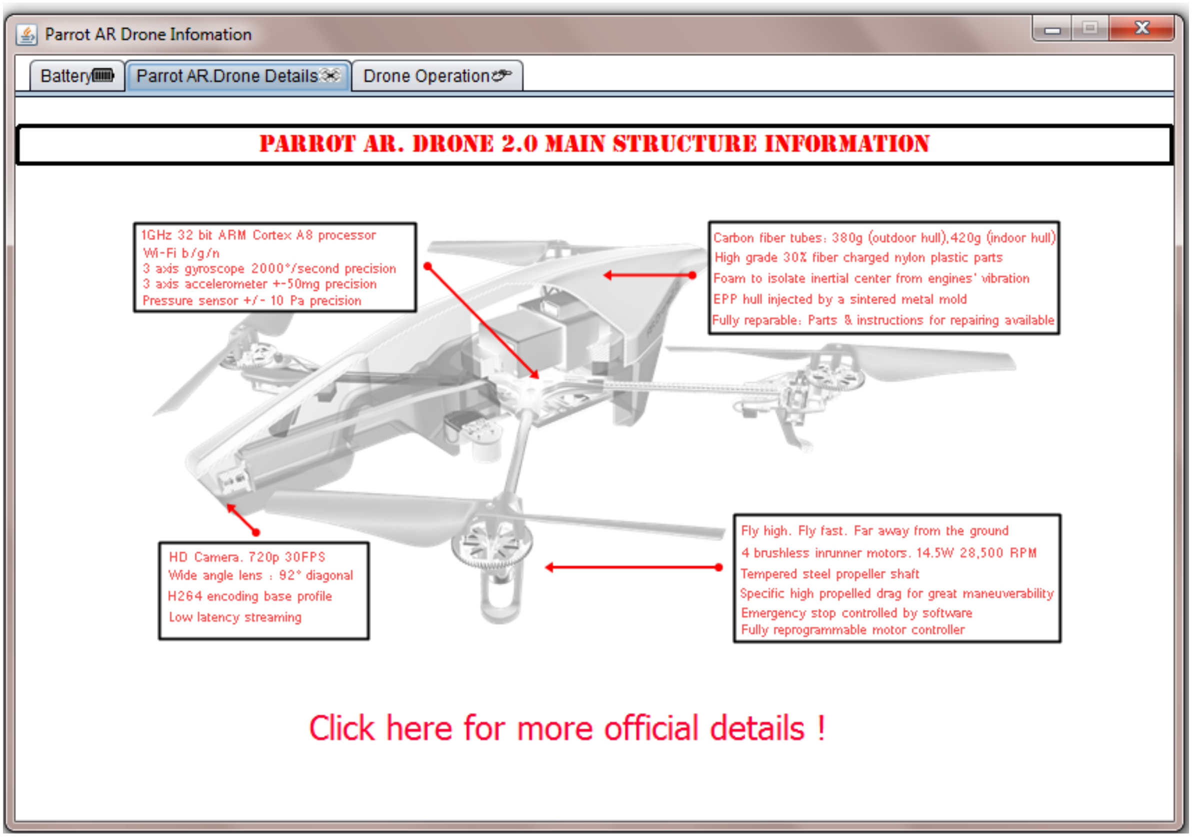

Figure 5-4-27: Parrot AR. Drone 2.0 main body structure

Simplified Parrot AR. Drone 2.0 main body structure information is shown to user when user clicks the “Parrot AR. Drone Details” tab. User will able to know most of the important features and technology that the AR. Drone have.

At the same time, if user wants to know latest information about Parrot AR. Drone, user can click on the “Click here for more official details !” label. User is directed to the Parrot company product website as shown in figure 5-4-28. All of the new Parrot products are shown in the website for user to read more information.

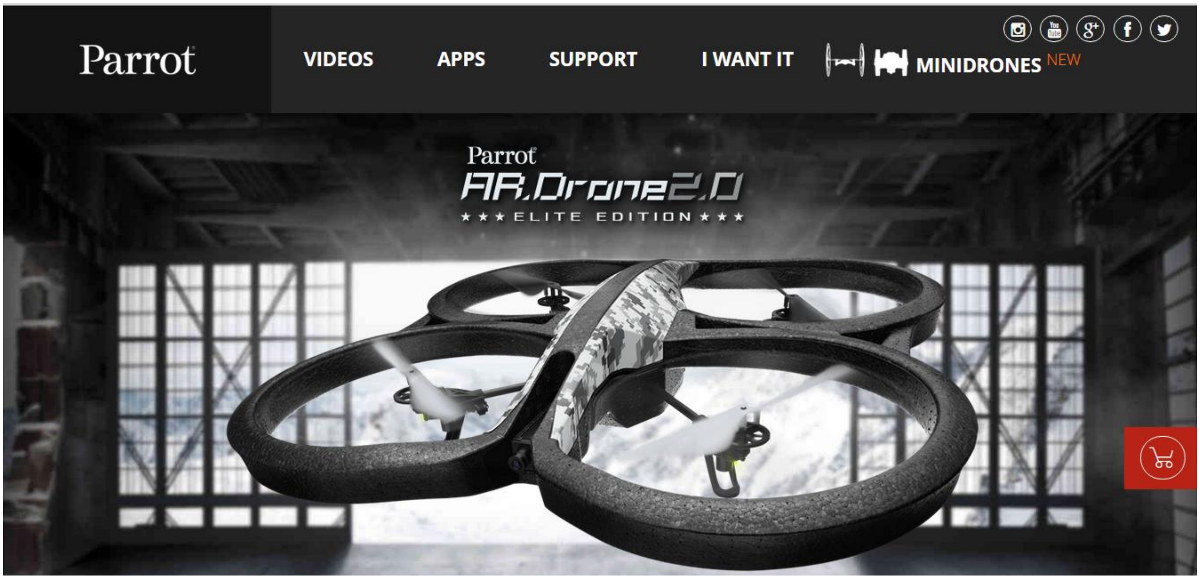

Figure 5-4-28: Parrot company product website

Under the "Drone Operation" tab, there are two crucial messages for the user as shown in figure 5-4-29 and figure 5-4-30. By default, the "DON'Ts" message is shown first and press the "Click here for next message!" for the "DOs" message.

Those "DON'Ts" and "DOs" drone regulations should be aware of and follow the rules by user whenever applicable for every flight.

Figure 5-4-29: DON'Ts rules message to user

Note – The above image was taken from: http://www.totallyunmanned.com/2015/04/20/singapore-releases-new-regulation-flying-uavs-airspace/

DON'Ts Rules:

1. Don't fly the aircraft over any crowd.
2. Don't fly an aircraft weighing more than 7kg (in total).
3. Don't suspend, carry or attach any item to the aircraft, unless it is manufactured to hold the item.
4. Don't carry hazardous substances using the aircraft.
5. Don't drop or discharge any item or substance from the aircraft.
6. Don't fly when you may interfere with emergency service providers; or over moving vehicles where you can endanger or distract drivers.

7. Don't fly the aircraft over or within restricted, prohibited or danger areas, including security-sensitive locations.
8. Don't fly within 5km of any airport or military airbase or higher than 200 feet.

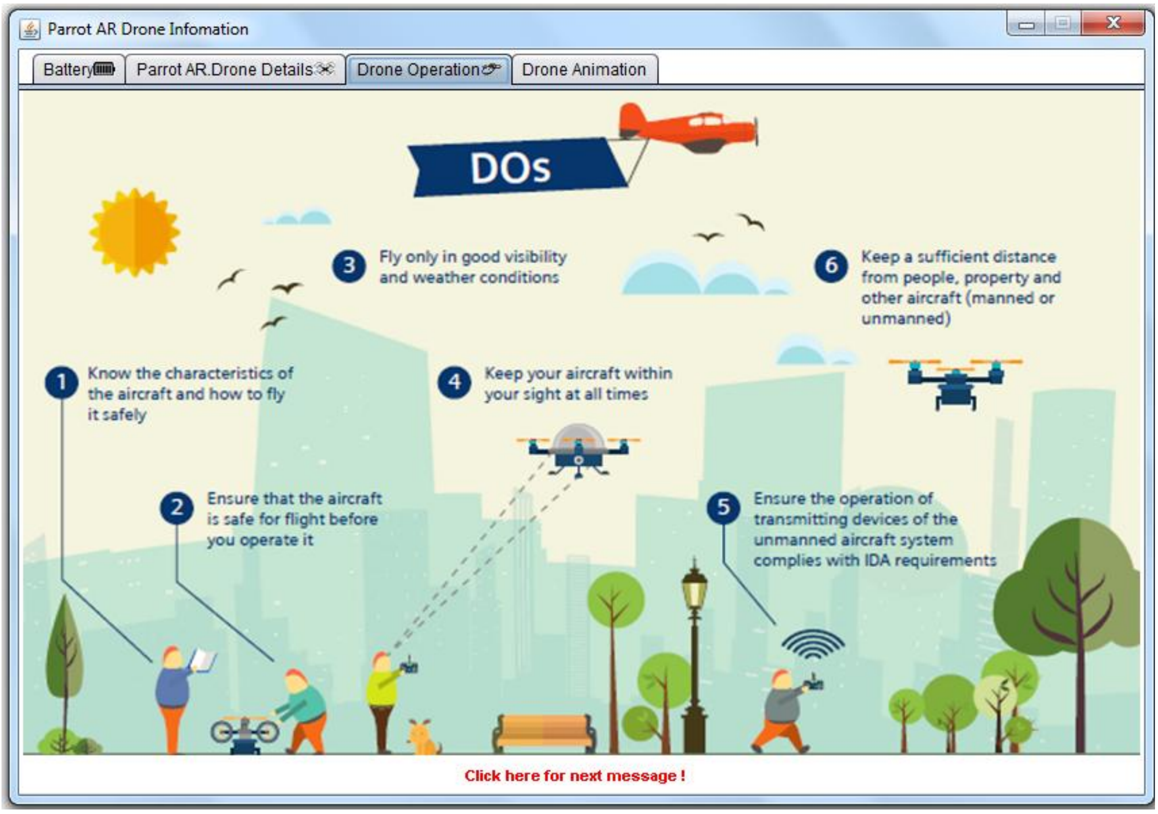

Figure 5-4-30: DOs message to user

Note – The above image was taken from: http://www.totallyunmanned.com/2015/04/20/singapore-releases-new-regulation-flying-uavs-airspace/

DOs:

1. Know the characteristics of the aircraft and how to fly it safely.
2. Ensure that the aircraft is safe for flight before you operate it.
3. Fly only in good visibility and weather conditions.

4. Keep your aircraft within your sight at all times.
5. Ensure the operation o transmitting devices of the unmanned aircraft system complies with IDA requirements.
6. Keep a sufficient distance from people, property and other aircraft (manned or unmanned).

In order to perform directional flight, user can click on “Map” button. New windows called “Map for Parrot AR. Drone 2.0” will be shown as in figure 5-4-32.

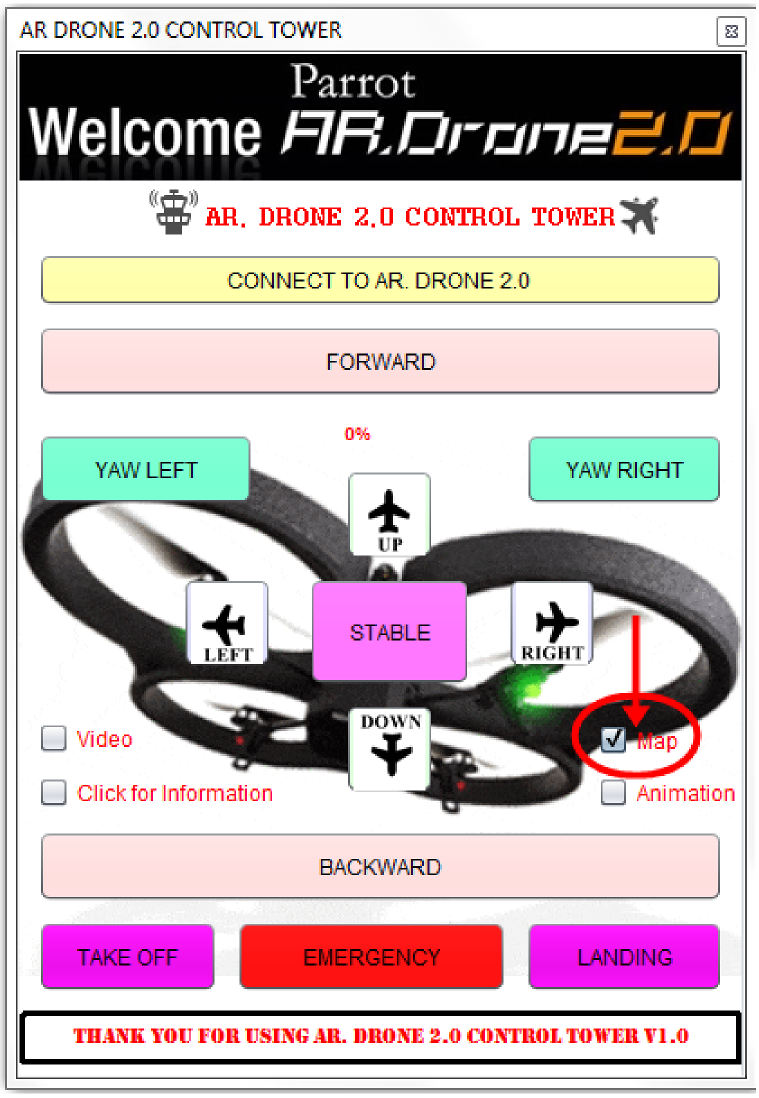

Figure 5-4-31: Click “Map” box

Abstract Command to Control Parrot AR Drone 2.0

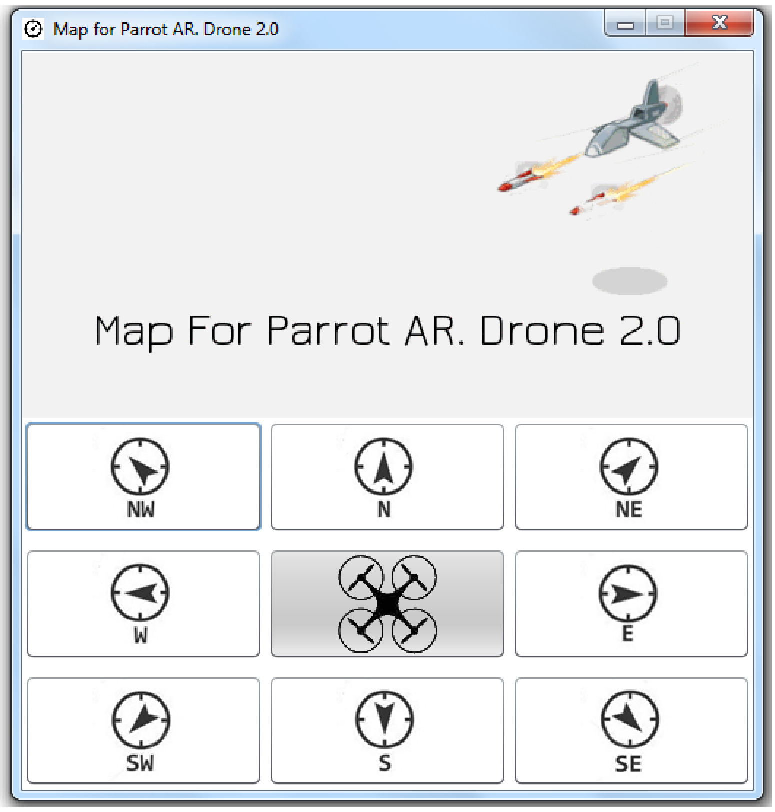

Figure 5-4-32: Map for Parrot AR. Drone 2.0

To take off, user double clicks on the grey color button where the Parrot AR. Drone icon is.

Wait for few seconds. The drone should take off now.

User clicks on the button for the desired direction. For example, if the user wants to fly to North, press on "N" button. The drone will fly heading toward to North direction and fly back to Center.

User can also perform special flight patterns by clicking "Animation" box as shown below.

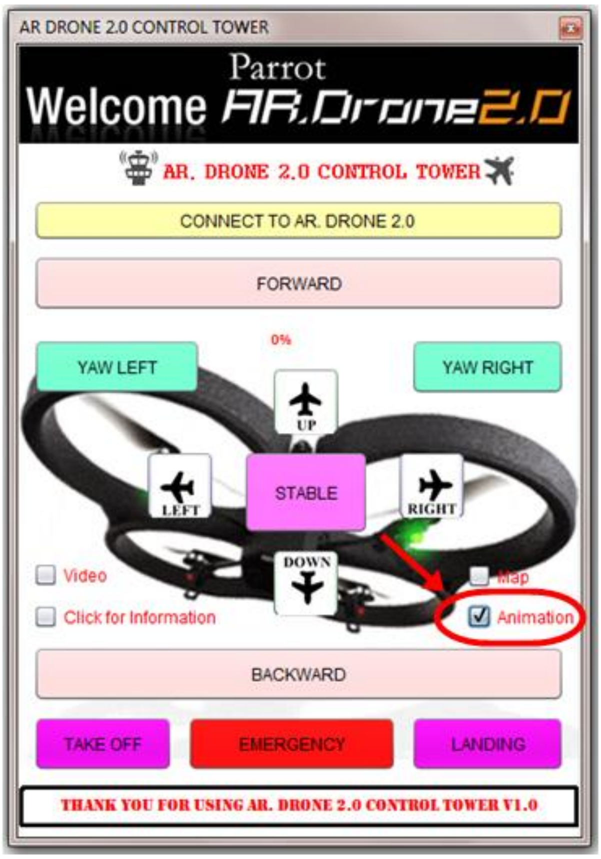

Figure 5-4-33: Drone Animation

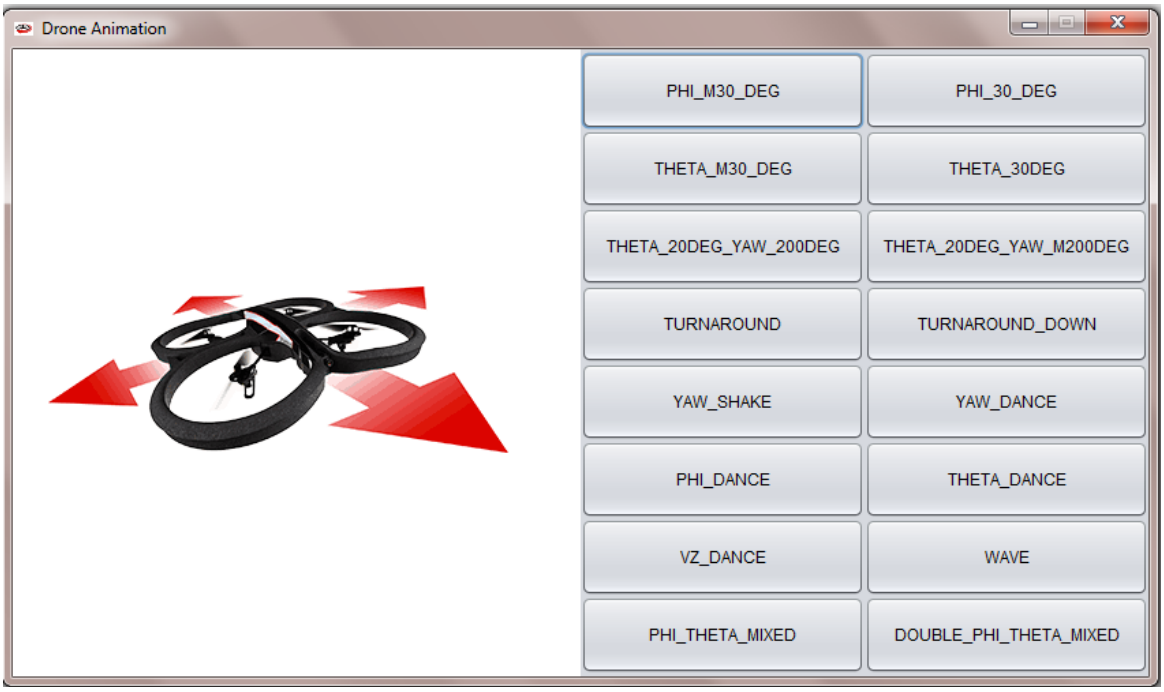

Figure 5-4-34: Drone Animation Windows

The drone animation windows display all of the possible drone animations for user. By clicking on button, it allows user to perform different unique flight actions that cannot be done by simple flight actions such as fly up, fly down, fly left, fly right, etc. Drone animations are combination of different parameters which makes the drone behaves in such special flight action.

In addition, there are 3 shortcuts functions in Free Flight Mode for user to perform free flight. These are:

(i) By left-click mouse within application window, the program pop up a simple confirmation message to perform take off or cancel take off whenever necessary as shown in figure 5-4-35.

(ii) By right-click mouse within application window, the program pop up a simple confirmation message to perform landing or cancel landing as shown in figure 5-4-36.

(iii) By middle-click mouse with within application window, the program sends emergency signal command to drone. The drone stops flying and enter into emergency mode.

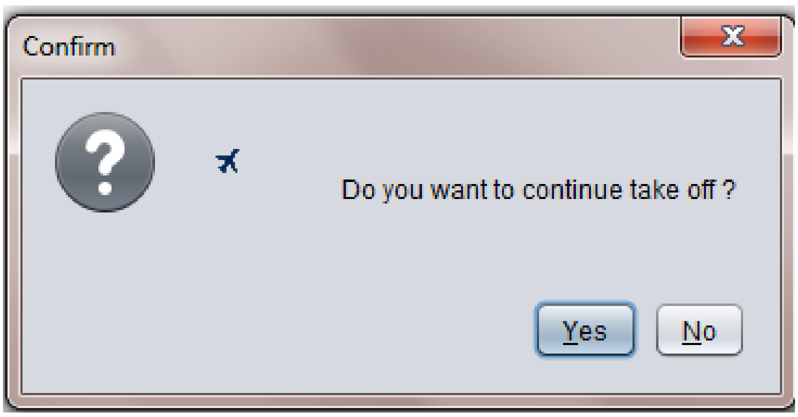

Figure 5-4-35: Confirmation message before take off

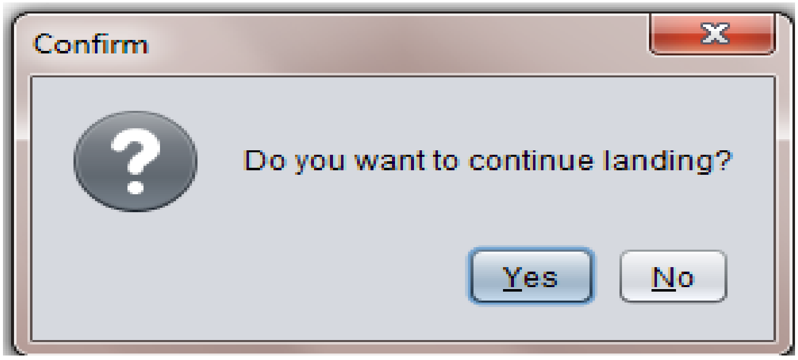

Figure 5-4-36: Confirmation message before landing

After pressing "Landing" button, the program send out "landing" command to the drone. The drone receives the landing command and perform landing. Then, the program prompts a message to select whether to display "Battery Graphic", "Altitude Graphic" or cancel the request. A simple data graph is shown to user corresponding to user selection.

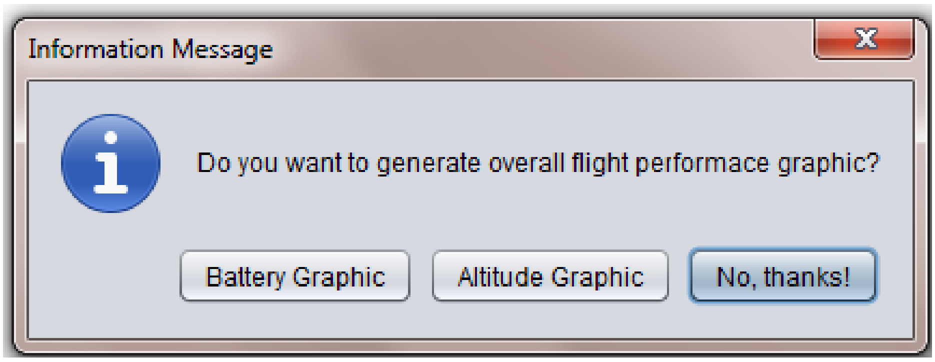

Figure 5-4-37: Confirmation message to make selection for generate data graph

This is an example of battery graph after a short flight. The battery values are remains at 95% level as shown below.

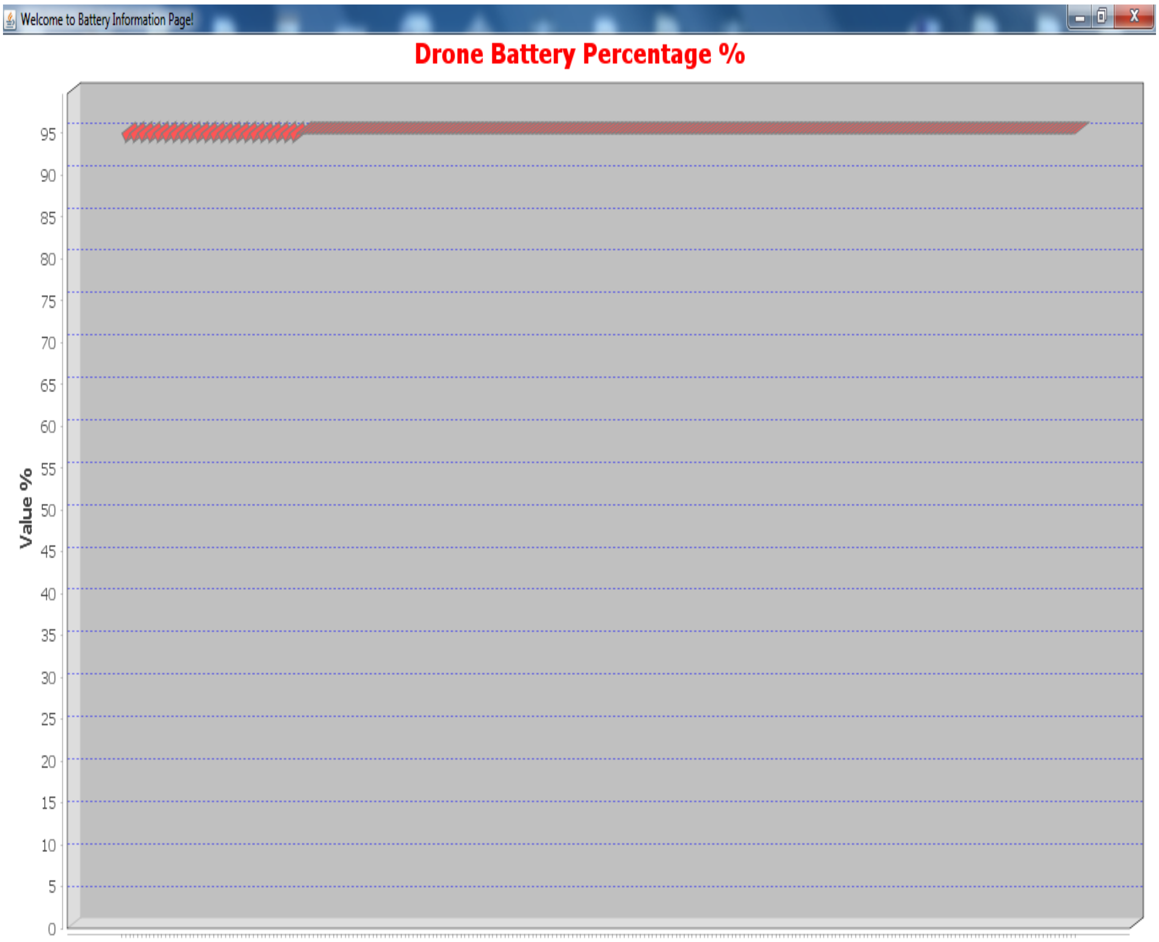

Figure 5-4-38: Battery graph

This is an example of a drone flight attitude graph after flight.

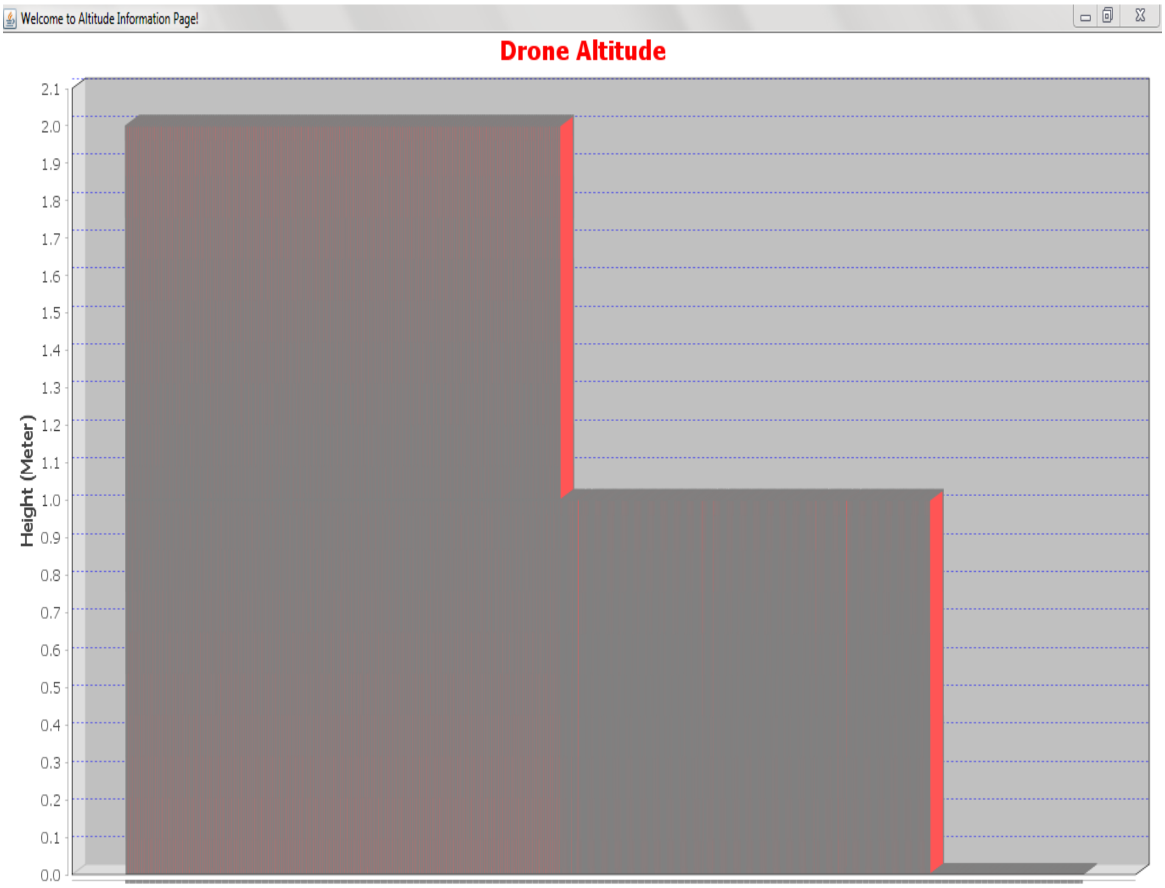

Figure 5-4-39: Drone altitude graph

Abstract Command Mode

In abstract command mode, user is allowed to select a series of abstract command and sends to the drone for perform such flight pattern. The figure below is the actual program interface for user to select abstract command.

Figure 5-4-40: Abstract Command Mode interface

User requires to press "Connect" button as shown below before start to send abstract command to the drone.

This is a must "To-Do" action before flight to ensure abstract commands could send out successfully to the AR. Drone as well as receive live feedback from the drone.

After successfully connected into the drone network, the drone should able to listen all of the abstract commands that are issue by user.

At the same time, new windows called "Command List" will shown to user. User can check what the abstract commands that they were sent before. Critical live updated drone navigation data are also being shown here. All of the essential flight data values such as remaining battery level, altitude level, pitch value, roll value, yaw value, and etc are being displayed as shown in figure 5-4-42 below.

Figure 5-4-41: Press “Connect” button before flight

Abstract Command to Control Parrot AR Drone 2.0

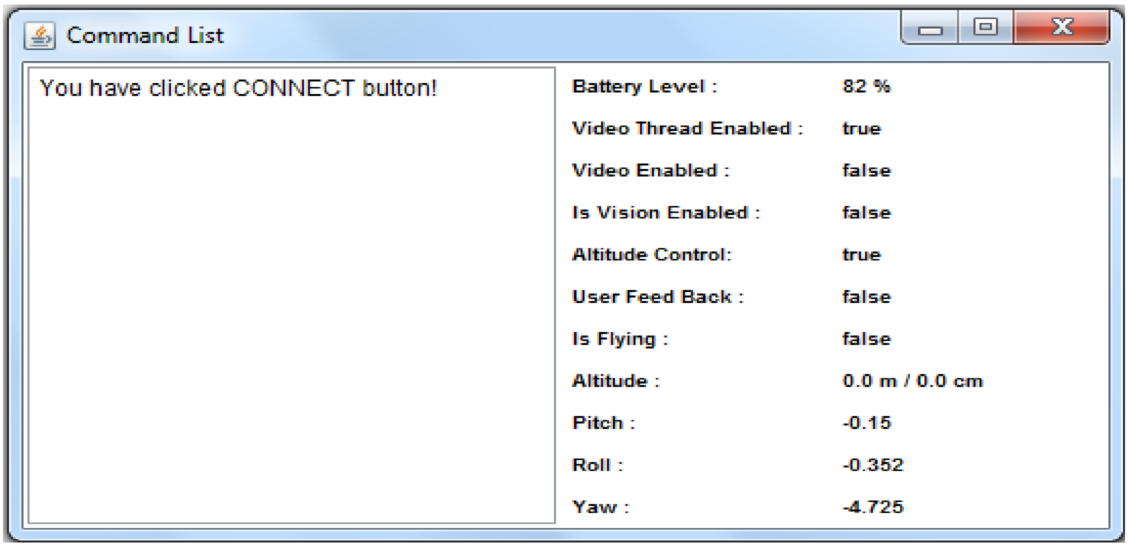

Figure 5-4-42: Command List windows

The figure 5-4-43 below shows that user has selected "Take Off", "Fly Forward" and "6 meter" commands.

Press "Execute!" button to continue sending the abstract commands to AR. Drone.

Figure 5-4-43: Abstract Command Selection

The drone lifts off from ground upon receipt of "Take off" command. After take-off, it hovers at approximately altitude of 0.77 meter for few seconds and then flies forward for 6 meter. After completed one series of abstract command, the drone continue to hovers and waiting for next series of abstract command.

If user wants to uncheck the abstract commands that are selected, press "Clear" button for one time and all of the checkbox will be unchecked. The previous abstract command will be deleted at the same time and waiting for new abstract command from user.

Figure 5-4-44: Press "Clear" button to uncheck all abstract command checkboxes and delete the previous abstract command

If user does not select any abstract command and continue to press "Execute!" button, a reminder message – "Sorry! No abstract command clicked yet…" will be shown below to alert them.

Abstract Command to Control Parrot AR Drone 2.0

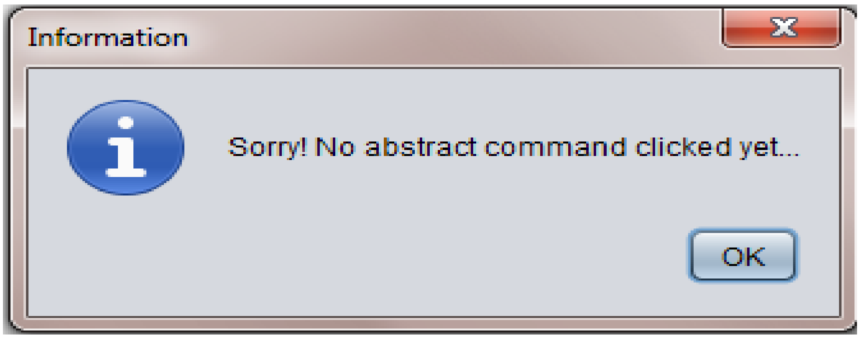

Figure 5-4-45: Reminder message to user

5.5 Concluding Remark

In this chapter, all of the necessary hardware and software setup have been described. By following all the setting up and configuring steps, the system will be able to run properly without any problem.

The system is able to perform sends abstract command to the drone and receives drone data continuously upon connect into the drone network.

Chapter 6 – System Testing and Discussion

6.1 System Testing and Performance

In this chapter, a series of system testing carried out in order to ensure that the project is working as intended. With a complete and integrated system testing, the proposed system is being evaluated to compliance with its specified requirements.

Total of 15 test flights with the AR. Drone in different cases have been tested out to the performance as well as reliability of the system.

Method: Performance and Reliability Testing

Applicable Requirement: Able to perform the abstract commands

Purpose: To calculate the number of successful times the AR. Drone able to perform the abstract commands which is considered as one trial

Item Under Test: Number of successful trail

Precautions: Drone battery should be fully charged and the application must be connected to the drone

Limitation: The testing is conducted inside indoor environment. No external environmental factor (such as wind condition) that affect overall flight performance

Equipment: Parrot AR. Drone 2.0 and personal laptop

Acceptance Criteria:
(i) Number of successful trails should more than the number of failure trails.
(ii) Acceptable difference of distance range should be within –0.5 meter and +1.0 meter based on testing case.
(iii) Drone heading should be precise or at least not vary too much.

During the testing phase of the proposed system, all of the activities that are involved on how the system expected to behave are evaluated. This is called software performance testing. Basically, it is tested from a user's perspective.

This is a good opportunity to assess the level of system performance. It is essential to ensure the system always available and runs as intended. There are several goals needed to achieve when execute performance testing. These include:

1. **Reliable command**: It is an utmost stage to check and verify that the Parrot AR. Drone accepts the correct commands from client (a.k.a. laptop). Before sending command to the drone, make sure the laptop is connected to the access point of AR. Drone. In order to pilot the AR. Drone, essential AT commands are needed to send over via UDP port 5556. Command validity is required to ensure prior to flight execution. Reliable command make sure the drone is controllable and able to fly according to the proper AT command that issued by the flyer. For example, correct AT command of take off should be able to send from laptop to AR. Drone successfully. The drone should detect and listen to every AT command that issued by client. The drone is expected to fly according to these AT command respectively.

2. **Accuracy of flight**: In the system testing, flight performance tests such as control and fly in "straight and level" and fly to specific direction (e.g. North, East, South and West) are being tested. As the name implies, a straight and level flight fly at a constant altitude and heading. Altitude and heading of the drone are maintained during the flight. Any unintentional turns or climbs should not be occurring and it affects overall flight performance. Accuracy of flight relies heavily on the correctness of appropriate AT commands and how the drone execute the AT commands.

3. **Accuracy of landing**: The drone is programmed attempt to land at precise landing location as closely as possible. After successfully reach the selected location, the drone should able to perform its landing automatically.

6.2 Testing Setup and Result

The table below indicates the number of successful trails on performing the abstract commands on different testing cases.

The setup of the hardware has been explained in Chapter 5 under section 5.1.

Testing Case 1: (Abstract Command Mode)

Take Off, Fly Forward, 2 Meter

Number of Trails	Success/Fail	Distance (meter)	Actual Behavior of AR. Drone
1	Success	1.8 m	Take off, Hover, Fly forward and Land
2	Success	1.7 m	Take off, Hover, Fly forward and Land
3	Success	2.0 m	Take off, Hover, Fly forward and Land
4	Success	1.5 m	Take off, Hover, Fly forward and Land
5	Success	2.8 m	Take off, Hover, Fly forward and Land
6	Fail	1.0 m	Take off, Hover, Fly forward, Stop, Continue to fly forward and Land
7	Success	2.2 m	Take off, Hover, Fly forward and Land
8	Fail	4 m	Take off, Hover, Fly forward and Land
9	Success	3 m	Take off, Hover, Fly forward and Land
10	Success	3 m	Take off, Hover, Fly forward and Land
11	Success	1.75 m	Take off, Hover, Fly forward and Land
12	Success	1.6 m	Take off, Hover, Fly forward and Land
13	Success	2.4 m	Take off, Hover, Fly forward and Land
14	Fail	3.7 m	Take off, Hover, Fly forward and Land
15	Success	2.8 m	Take off, Hover, Fly forward and Land

Table 6-2-1: Result of Testing Case 1 (Take Off, Fly Forward, 2 Meter)

Number of trails: 15
Number of successful trails: 12
Number of failed trails: 3
Success rate: **12/15 * 100% = 80%**
Failure rate: **3/15 * 100% = 20%**

Based on the above mathematics calculation, the outcome of the testing case 1 is considered as acceptable. It was failed 3 times out of total of 15 trails. In other words, the failure rate is 20% out of 100%. Meanwhile, with the success rate of 80% which is higher than the failure rate of 20%, the abstract command is considered as relatively reliable.

Testing Case 2: (Abstract Command Mode)

Take Off, Fly Forward, 4 Meter

Number of Trails	Success/Fail	Distance (meter)	Actual Behavior of AR. Drone
1	Success	4.3 m	Take off, Hover, Fly forward and Land
2	Success	4.1 m	Take off, Hover, Fly forward and Land
3	Fail	5.1 m	Take off, Hover, Fly forward and Land
4	Success	3.9 m	Take off, Hover, Fly forward and Land
5	Success	3.9 m	Take off, Hover, Fly forward and Land
6	Success	3.9 m	Take off, Hover, Fly forward and Land
7	Success	4.0 m	Take off, Hover, Fly forward and Land
8	Fail	2.3 m	Take off, Hover, Fly forward, Stop, Continue to fly forward and Land
9	Success	3.8 m	Take off, Hover, Fly forward and Land
10	Fail	2.3 m	Take off, Hover, Fly forward, Stop, Continue to fly forward and Land
11	Success	4.5 m	Take off, Hover, Fly forward and Land
12	Success	4.1 m	Take off, Hover, Fly forward and Land
13	Success	4.0 m	Take off, Hover, Fly forward and Land
14	Fail	3.7 m	Take off, Hover, Fly forward, Stop, Continue to fly forward and Land
15	Success	4.1 m	Take off, Hover, Fly forward and Land

Table 6-2-2: Result of Testing Case 2 (Take Off, Fly Forward, 4 Meter)

Number of trails: 15

Number of successful trails: 11

Number of failed trails: 4

Success rate: **11/15 * 100% = 73%**

Failure rate: **4/15 * 100% = 27%**

Based on the above mathematics calculation, the outcome of the testing case 2 is considered as acceptable. It was failed 4 times out of total of 15 trails. In other words, the failure rate is 27% out of 100%. Meanwhile, with the success rate of 73% which is higher than the failure rate of 27%, the abstract command is considered as reliable.

Testing Case 3: (Abstract Command Mode)

Take Off, Fly Forward, 6 Meter

Number of Trails	Success/Fail	Distance (meter)	Actual Behavior of AR. Drone
1	Success	6.3 m	Take off, Hover, Fly forward and Land
2	Success	6.0 m	Take off, Hover, Fly forward and Land
3	Fail	3.8 m	Take off, Hover, Fly forward, Stop, Continue to fly forward and Land
4	Fail	4.2 m	Take off, Hover, Fly forward, Stop, Continue to fly forward and Land
5	Success	5.8 m	Take off, Hover, Fly forward and Land
6	Success	5.9 m	Take off, Hover, Fly forward and Land
7	Success	6.2 m	Take off, Hover, Fly forward and Land
8	Success	6.3 m	Take off, Hover, Fly forward, Stop, Continue to fly forward and land
9	Success	5.7 m	Take off, Hover, Fly forward and Land
10	Success	5.8 m	Take off, Hover, Fly forward and Land
11	Success	6.0 m	Take off, Hover, Fly forward and Land
12	Success	6.3 m	Take off, Hover, Fly forward and Land
13	Success	6.3 m	Take off, Hover, Fly forward and Land
14	Fail	3.7 m	Take off, Hover, Fly forward, Stop, Continue to fly forward and Land
15	Success	5.7 m	Take off, Hover, Fly forward and Land

Table 6-2-3: Result of Testing Case 3 (Take Off, Fly Forward, 6 Meter)

Number of trails: 15

Number of successful trails: 12

Number of failed trails: 3

Success rate: **12/15 * 100% = 80%**

Failure rate: **3/15 * 100% = 20%**

Based on the above mathematics calculation, the outcome of the testing case 3 is considered as acceptable. It was failed 3 times out of total of 15 trails. In other words, the failure rate is 20% out of 100%. Meanwhile, with the success rate of 80% which is higher than the failure rate of 20%, the abstract command is considered as relatively reliable.

Testing Case 4: (Abstract Command Mode)

Take Off, Fly Backward, 2 Meter

Number of Trails	Success/Fail	Distance (meter)	Actual Behavior of AR. Drone
1	Success	2.0 m	Take off, Hover, Fly backward and Land
2	Success	2.0 m	Take off, Hover, Fly backward and Land
3	Success	2.0 m	Take off, Hover, Fly backward and Land
4	Success	1.8 m	Take off, Hover, Fly backward and Land
5	Success	2.5 m	Take off, Hover, Fly backward and Land
6	Success	2.0 m	Take off, Hover, Fly backward and Land
7	Success	2.2 m	Take off, Hover, Fly backward and Land
8	Success	2.2 m	Take off, Hover, Fly backward and Land
9	Fail	1.1 m	Take off, Hover, Fly backward, Stop, Continue to fly backward and Land
10	Success	1.8 m	Take off, Hover, Fly backward and Land
11	Success	2.3 m	Take off, Hover, Fly backward and Land
12	Success	2.0 m	Take off, Hover, Fly backward and Land
13	Success	1.7 m	Take off, Hover, Fly backward and Land
14	Success	1.8 m	Take off, Hover, Fly backward and Land
15	Success	2.0 m	Take off, Hover, Fly backward and Land

Table 6-2-4: Result of Testing Case 4 (Take Off, Fly Backward, 2 Meter)

Number of trails: 15

Number of successful trails: 14

Number of failed trails: 1

Success rate: **14/15 * 100% = 93%**

Failure rate: **1/15 * 100% = 7%**

Based on the above mathematics calculation, the outcome of the testing case 4 is considered as successful. It was failed only 1 time out of total of 15 trails. In other words, the failure rate is 7% out of 100%. Meanwhile, with the success rate of 93% which is higher than the failure rate of 7%, the abstract command is considered as significantly reliable.

Testing Case 5: (Abstract Command Mode)

Take Off, Fly Backward, 4 Meter

Number of Trails	**Success/Fail**	**Distance (meter)**	**Actual Behavior of AR. Drone**
1	Fail	3.0 m	Take off, Hover, Fly backward and Land
2	Success	5.0 m	Take off, Hover, Fly backward and Land
3	Success	3.8 m	Take off, Hover, Fly backward and Land
4	Success	4.0 m	Take off, Hover, Fly backward and Land
5	Success	4.1 m	Take off, Hover, Fly backward and Land
6	Success	4.0 m	Take off, Hover, Fly backward and Land
7	Fail	2.1 m	Take off, Hover, Fly backward, Stop, Continue to fly backward and Land
8	Success	3.9 m	Take off, Hover, Fly backward and Land
9	Fail	6.0 m	Take off, Hover, Fly backward and Land
10	Success	4.1 m	Take off, Hover, Fly backward and Land
11	Success	3.7 m	Take off, Hover, Fly backward and Land
12	Success	4.0 m	Take off, Hover, Fly backward and Land
13	Success	4.5 m	Take off, Hover, Fly backward and Land
14	Success	3.7 m	Take off, Hover, Fly backward and Land
15	Success	4.0 m	Take off, Hover, Fly backward and Land

Table 6-2-5: Result of Testing Case 5 (Take Off, Fly Backward, 4 Meter)

Number of trails: 15 Number of successful trails: 12 Number of failed trails: 3 Success rate: **12/15 * 100% = 80%** Failure rate: **3/15 * 100% = 20%**

Based on the above mathematics calculation, the outcome of the testing case 5 is considered as successful. It was failed 3 times out of total of 15 trails. In other words, the failure rate is 20% out of 100%. Meanwhile, with the success rate of 80% which is higher than the failure rate of 20%, the abstract command is considered as reliable.

Testing Case 6: (Abstract Command Mode)

Take Off, Fly Backward, 6 Meter

Number of Trails	Success/Fail	Distance (meter)	Actual Behavior of AR. Drone
1	Fail	4.0 m	Take off, Hover, Fly backward, Stop, Continue to fly backward and Land
2	Success	5.8 m	Take off, Hover, Fly backward and Land
3	Fail	4.3 m	Take off, Hover, Fly backward, Stop, Continue to fly backward and Land
4	Success	6.5 m	Take off, Hover, Fly backward and Land
5	Success	6.0 m	Take off, Hover, Fly backward and Land
6	Success	5.8 m	Take off, Hover, Fly backward and Land
7	Success	6.1 m	Take off, Hover, Fly backward, Stop, Continue to fly backward and Land
8	Success	6.5 m	Take off, Hover, Fly backward and Land
9	Success	6.5 m	Take off, Hover, Fly backward and Land
10	Fail	4.3 m	Take off, Hover, Fly backward, Stop, Continue to fly backward and Land
11	Fail	4.3 m	Take off, Hover, Fly backward, Stop, Continue to fly backward and Land
12	Success	6.0 m	Take off, Hover, Fly backward and Land
13	Success	6.5 m	Take off, Hover, Fly backward and Land
14	Success	5.8 m	Take off, Hover, Fly backward and Land
15	Success	5.7 m	Take off, Hover, Fly backward and Land

Table 6-2-6: Result of Testing Case 6 (Take Off, Fly Backward, 6 Meter)

Number of trails: 15 Number of successful trails: 11 Number of failed trails: 4 Success rate: **11/15 * 100% = 73%** Failure rate: **4/15 * 100% = 27%**

Based on the above mathematics calculation, the outcome of the testing case 6 is considered as acceptable. It was failed 4 times out of total of 15 trails. In other words, the failure rate is 27% out of 100%. Meanwhile, with the success rate of 73% which is higher than the failure rate of 27%, the abstract command is considered as reliable.

Testing Case 7: (Abstract Command Mode)

Take Off, Fly Left, 2 Meter

Number of Trails	Success/Fail	Distance (meter)	Actual Behavior of AR. Drone
1	Success	2.0 m	Take off, Hover, Fly left and Land
2	Success	2.0 m	Take off, Hover, Fly left and Land
3	Fail	1.1 m	Take off, Hover, Fly left, Stop, Continue to fly left and Land
4	Success	1.9 m	Take off, Hover, Fly left and Land
5	Success	2.1 m	Take off, Hover, Fly left and Land
6	Success	1.7 m	Take off, Hover, Fly left and Land
7	Fail	1.2 m	Take off, Hover, Fly left, Stop, Continue to fly left and Land
8	Success	2.0 m	Take off, Hover, Fly left and Land
9	Success	1.8 m	Take off, Hover, Fly left and Land
10	Success	1.7 m	Take off, Hover, Fly left and Land
11	Fail	1.2 m	Take off, Hover, Fly Left, Stop, Continue to fly left and Land
12	Success	1.9 m	Take off, Hover, Fly left and Land
13	Success	2.1 m	Take off, Hover, Fly left and Land
14	Success	1.9 m	Take off, Hover, Fly left and Land
15	Success	2.0 m	Take off, Hover, Fly left and Land

Table 6-2-7: Result of Testing Case 7 (Take Off, Fly Left, 2 Meter)

Number of trails: 15
Number of successful trails: 12
Number of failed trails: 3
Success rate: **12/15 * 100% = 80%**
Failure rate: **3/15 * 100% = 20%**

Based on the above mathematics calculation, the outcome of the testing case 7 is considered as successful. It was failed 3 times out of total of 15 trails. In other words, the failure rate is 20% out of 100%. Meanwhile, with the success rate of 80% which is higher than the failure rate of 20%, the abstract command is considered as reliable.

Testing Case 8: (For Directional Flight – Map for Parrot AR. Drone)

Directional Flight Testing (North) - Take Off, Fly Heading North, Right Turn for 180 Degree, Fly back to Center

Number of Trails	**Success /Fail**	**Actual Behavior of AR. Drone**
1	Success	Take Off from Center, Fly toward North direction, Right Turn for 180 Degree, Fly back to Center and Land
2	Success	Take Off from Center, Fly toward North direction, Right Turn for 180 Degree, Fly back to Center and Land
3	Fail	Take Off from Center, Fly toward North direction, Right Turn for more than 180 Degree, Fly back to Center and Land
4	Success	Take Off from Center, Fly toward North direction, Right Turn for 180 Degree, Fly back to Center and Land
5	Success	Take Off from Center, Fly toward North direction, Right Turn for 180 Degree, Fly back to Center and Land
6	Success	Take Off from Center, Fly toward North direction, Right Turn for 180 Degree, Fly back to Center and Land
7	Fail	Take Off from Center, Fly toward North direction, Right Turn for more than 180 Degree, Fly back to Center and Land
8	Success	Take Off from Center, Fly toward North direction, Right Turn for 180 Degree, Fly back to Center and Land
9	Success	Take Off from Center, Fly toward North direction, Right Turn for 180 Degree, Fly back to Center and Land
10	Success	Take Off from Center, Fly toward North direction, Right Turn for 180 Degree, Fly back to Center and Land
11	Success	Take Off from Center, Fly toward North direction, Right Turn for 180 Degree, Fly back to Center and Land
12	Fail	Take Off from Center, Fly toward North direction, Right Turn for more than 180 Degree, Fly back to Center and Land
13	Success	Take Off from Center, Fly toward North direction, Right Turn for 180 Degree, Fly back to Center and Land
14	Success	Take Off from Center, Fly toward North direction, Right Turn for 180 Degree, Fly back to Center and Land
15	Success	Take Off from Center, Fly toward North direction, Right Turn for 180 Degree, Fly back to Center and Land

Table 6-2-8: Result of Testing Case 8 (Take Off, Fly Heading North, Right Turn for 180 Degree, Fly back to Center)

***Note:-**

Success: Fly toward North direction, able to achieve 180 degree right turns and flies back to Center.

Fail: Due to drone drifting or wind condition issue, the drone did more than 180 degree right turns.

Number of trails: 15
Number of successful trails: 12
Number of failed trails: 3
Success rate: **12/15 * 100% = 80%**
Failure rate: **3/15 * 100% = 20%**

Based on the above mathematics calculation, the outcome of the testing case 8 is considered as successful. It was failed 3 times out of total of 15 trails. In other words, the failure rate is 20% out of 100%. Meanwhile, with the success rate of 80% which is higher than the failure rate of 20%, the abstract command is considered as reliable.

Testing Case 9: (For Directional Flight – Map for Parrot AR. Drone)

Directional Flight Testing (South) - Take Off, Right Turn for 180 Degree, Fly Heading South, Right Turn for 180 Degree, Fly back to Center

Number of Trails	Success /Fail	Actual Behavior of AR. Drone
1	Success	Take Off from Center, Right Turn for 180 Degree, Fly toward South direction, Right Turn for 180 Degree, Fly back to Center and Land
2	Fail	Take Off from Center, Right Turn for 180 Degree, Fly toward South direction, Right Turn for less than 180 Degree, Fly back to Center and Land
3	Fail	Take Off from Center, Right Turn for 180 Degree, Fly toward South direction, Right Turn for less than 180 Degree, Fly back to Center and Land
4	Success	Take Off from Center, Right Turn for 180 Degree, Fly toward South direction, Right Turn for 180 Degree, Fly back to Center and Land
5	Success	Take Off from Center, Right Turn for 180 Degree, Fly toward South direction, Right Turn for 180 Degree, Fly back to Center and Land
6	Success	Take Off from Center, Right Turn for 180 Degree, Fly toward South direction, Right Turn for 180 Degree, Fly back to Center and Land
7	Success	Take Off from Center, Right Turn for 180 Degree, Fly toward South direction, Right Turn for 180 Degree, Fly back to Center and Land
8	Success	Take Off from Center, Right Turn for 180 Degree, Fly toward South direction, Right Turn for 180 Degree, Fly back to Center and Land
9	Success	Take Off from Center, Right Turn for 180 Degree, Fly toward South direction, Right Turn for 180 Degree, Fly back to Center and Land
10	Success	Take Off from Center, Right Turn for 180 Degree, Fly toward South direction, Right Turn for 180 Degree, Fly back to Center and Land
11	Success	Take Off from Center, Right Turn for 180 Degree, Fly toward South direction, Right Turn for 180 Degree, Fly back to Center and Land
12	Fail	Take Off from Center, Right Turn for 180 Degree, Fly toward South direction, Right Turn for less than 180 Degree, Fly back to Center and Land
13	Success	Take Off from Center, Right Turn for 180 Degree, Fly toward South direction, Right Turn for 180 Degree, Fly back to Center and Land
14	Fail	Take Off from Center, Right Turn for 180 Degree, Fly toward South direction, Right Turn for more than 180 Degree, Fly back to Center and Land
15	Success	Take Off from Center, Right Turn for 180 Degree, Fly toward South direction, Right Turn for 180 Degree, Fly back to Center and Land

Table 6-2-9: Result of Testing Case 9 (Take Off, Right Turn for 180 Degree, Fly Heading South, Right Turn for 180 Degree, Fly back to Center)

***Note:-**

Success: 180 degree right turns, Fly toward South direction, 180 degree right turns and flies back to Center.

Fail: Due to drone drifting or wind condition issue, the drone did more or lesser than 180 degree right turns

Number of trails: 15

Number of successful trails: 11

Number of failed trails: 4

Success rate: **11/15 * 100% = 73%**

Failure rate: **4/15 * 100% = 27%**

Based on the above mathematics calculation, the outcome of the testing case 9 is considered as successful. It was failed 4 times out of total of 15 trails. In other words, the failure rate is 27% out of 100%. Meanwhile, with the success rate of 73% which is higher than the failure rate of 27%, the abstract command is considered as reliable.

6.3 Project Challenges

There are a lot of difficulties were faced during the development of the proposed system. It is never an easy assignment to work on. Positive learning attitude needed to possess and maintain at all time. This is an important yet useful interpersonal skill in studies so that problem can be treated as an opportunity to learn and develop problem-solving skill.

One of the biggest challenges that encountered in this project was the fast draining battery issue. This possesses a very big challenging issue in this project because every system testing session could only last for 10 to 12 minutes only. The Parrot AR. Drone 2.0 Wi-Fi quadricopter has short battery life time like others remote-controlled flying devices, a fully charged battery can only supply 1,000 mAh allows about 12 minutes of flight time. During system testing session, the battery level drops quickly. A low powered battery no longer can be used in system testing session. It might affect performance of the drone as well as accuracy of the system testing. Once the drone battery level lower than 20%, the drone will automatically land and the motor LED lights illuminates in steady red color. A 1,000 mAh battery takes approximately 1 hour and 30 minutes for a fully recharge. Most of the time was actually spent on battery charging process. In fact, a lot of system testing cases needed to be evaluated. Such limitation greatly affects overall system testing progress since longer testing duration is needed to carry out various system testing.

Besides that, sometimes the drone might drift when it starts to hover shortly after take-off. The drifting problem causes inaccuracy when comes to system testing phase. Due to the drifting issue, it moves slightly either forward or backward. A series of abstract command - “Take off, Fly Forward, 4 Meter” which suppose can be realized with more or less 4 meter forward distance result is hard to be achieved.

In addition, due to limited and minimal JavaCV reference code that available online, vision obstacle detection function requires longer development time to explore and make it works. A simple vision detection module was created and tested but it is only limited to detect colors of an item.

Weather events such as high winds and gusty weather also affect the overall flight performance of the flying drone. The flying drone might not able to fly properly under such condition. Therefore, all of the flight systems testing cases carried out within indoor environment whereby external factors that affect the performance of drone can be reduced to minimum and actual behavior of the flying drone can be exhibited.

6.4 Objectives Evaluation

Two project objectives are achieved successfully in this project. These are:

1. <u>To study and compare existing flying technology</u>

This is one of the first project objectives that is required to achieve in order to develop the fundamental of the project. Without proper and sufficient studies on existing flying technology, it will be difficult to understand what are the latest technologies and features of existing flying drones. Better understanding of drones might need some time to review on. Once finish reviewing and comparing on existing flying technology, a clear picture on how to develop the proposed system can be visualized.

Review starts from old remote-control technology - 2.4 GHz radio controller to most state-of-the-art remote-control technologies such as Bluetooth and Wi-Fi. Studies on the technique to control and flying Parrot AR. Drone 2.0 autonomously are focused in this project as well.

2. <u>To enable the drone flying in Autonomous Navigation Mode to reach any selected destination</u>

The second project objective is achieved when the “Map for Parrot AR. Drone 2.0” interface developed in this project. In the “Map for Parrot AR. Drone 2.0” interface, there are several buttons for user to click.

User double clicks on the grey color button and the drone take off from ground. After that, user can clicks on the button for any desired direction. For example, if the user wants to fly to North, press on “N” button. The drone will fly heading toward to North direction and fly back to Center.

However, as mentioned earlier on, the third project objective –

3. <u>To introduce Parrot AR. Drone 2.0 by flying in Autonomous Navigation mode (Without GPS module and GPS services)</u>

In the proposed system, the drone should be able to react independently to avoid hitting any environmental obstacles. That means the drone should able to detect and avoid any frontal obstacles along its flight path. However, due to limited and minimal JavaCV reference code that available online, vision obstacle detection function requires longer development time to explore and make it works. A simple vision detection module was created and tested but it is only limited to detect colors of an item.

Even through the third project objective as stated above still trying to achieve, core of the project objective which enable the drone flying in Autonomous Navigation Mode to reach any selected destination is achieved.

6.5 Concluding Remark

After the system undergoes a set of extensive reliability and performance testing, the system considered relatively reliable. All of the testing cases passed with satisfactory outcome which yields average success rate more than 75%.

Throughout the development of the system, evaluation on project objectives carried out from time to time. Besides that, during the reliability and performance system testing, actual behavior of how the drone react is observed and record down the outcome of the experiments.

After the completion of assessing testing cases, two out of three project objectives that stated in chapter 1.2 have been fulfilled.

These two project objectives are:

(i) To study and compare existing flying technology

(ii) To enable the drone flying in Autonomous Navigation Mode to reach any selected destination

However, another project objective that is to introduce Parrot AR. Drone 2.0 by flying in Autonomous Navigation mode (Without GPS module and GPS services) is still trying to achieve it. Research on how to detect obstacles along flight path is much needed.

Nevertheless, the main project objective which is to enable the drone flying in Autonomous Navigation Mode to reach any selected destination is achieved.

Chapter 7 – Conclusion and Recommendation

7.1 Conclusion

In this section, all of the project objectives and scope are being review once again in order to assess what have been done are align with the propose solution as in earlier chapter and flying correctly according to the abstract command that issues by user via PC Wi-Fi module. The main objective of this project is to improve the existing Parrot AR. Drone 2.0 to have a better flying experience.

Therefore, a series of existing flying drone technology have been studied and compared before start to develop this project. Besides that, different techniques of controlling and flying Parrot AR. Drone 2.0 autonomously also have been evaluated. Wi-Fi wireless connection technology is being used in this project.

The drone automatically creates an ad-hoc Wi-Fi hotspot after it is powered on. Electronic controlling device such as Smartphone or PC can connect to the drone’s Wi-Fi network and communicates with the drone. After successfully connect into the AR. Drone’s Wi-Fi network, simple abstract command such as take off, landing, flying up, flying down, recording video, screen capture and etc should be able to send from PC Wi-Fi to the drone. The drone should able to execute the commands accordingly.

Besides that, the ability to fly the drone in autonomous navigation mode to reach any selected destination also included in this project. This allows user to fly the drone in more convenience method whereby user are no longer flying the drone in conventional method. By clicking button, the drone will take off from ground and fly forward to the selected destination. In conventional method, flying drone requires constant human supervision. It might be a tiring and time-consuming process if user flies for a long time.

Live streaming video that captured by frontal mounted camera are sent to PC after it is activated by user. Subsequently user is able to get the latest camera live viewing at

PC. This certainly gives user a better viewing experience as well as overall flying experience. Obstacles such as tables, chairs, cupboards and etc that are exists along flying path can be avoided easily by observing the live streaming video from the flying drone.

There are a lot of technical and programming issues had encountered throughout the development of this project. Despite of that, those headache problems were solved following one by one which also provide a good opportunity for understanding how the drone works. Upon the completion of this project, most of the general knowledge in abstract command programming and controlling flying drone devices have been gained and widen as well.

7.2 Recommendation

Further improvement and enhancement can be done on this project. One of the modules that can be enriched on is the obstacle detection system. Due to time constraint and limited prior knowledge of JavaCV, the obstacle detection module is still under development and explores the possibilities of developing a good obstacle detection module for the flying drone. The obstacle detection module is proposed to enable the drone fly in autonomous flying mode and it can avoid any obstacle along the flying path by itself.

In addition, precision on direction heading also can be enhanced as well. In this project, the drone moves by calling move function. The move function receives 4 different float data type parameter values. The fourth parameter value of move function is used for turning the drone heading. Positive value of the fourth parameter makes the drone turn to right direction around its vertical axis. For example, "drone.move(0.0f, 0.0f, 0.0f, 1.0f);" of an abstract command that makes the drone turn right to 45 degree. In order to make a 180 degree right turn, the "drone.move(0.0f, 0.0f, 0.0f, 1.0f);" of abstract command needs to be executed continuously for 4 times.

During the right turning process, the drone drifts slightly which cause it to turn more than 180 degree. Therefore, precision on the direction turning should be more focus on making it work to ensure that accurate direction heading can be achieved.

References:

Long Cheng, Cheng-dong Wu, Yun-zhou Zhang & Yan Wang, 2012, An Indoor Localization Strategy for a mini-UAV in the Presence of Obstacles. Available from: <http://cdn.intechopen.com/pdfs/40602.pdf>. [29 January 2015].

Tobias Nisser & Carl Westin, 2006, Human Factors Challenges in Unmanned Aerial Vehicles (UAVs): A Literature Review. Available from: <http://www4.lu.se/upload/Trafikflyghogskolan/HumanFactorsChallengesUnmanned AerialVehicles.pdf>. [29 January 2015].

Johann-Sebastian Pleban, Ricardo Band & Reiner Creutzburg, 2013, Hacking and securing the AR.Drone 2.0 quadcopter - Investigations for improving the security of a toy. Available from:<http://www.researchgate.net/profile/Reiner_Creutzburg/publication/260420467_Hacking_and_securing_the_AR.Drone_2.0_quadcopter_investigations_for_improving_the_security_of_a_toy/links/00463531f196516535000000.pdf>. [29 January 2015].

KickStarter.com, n.d., Phenox: intelligent, interactive and programmable drone. Available from:< https://www.kickstarter.com/projects/253110733/phenox-intelligent-interactive-and-programmable-dr>. [1 March 2015].

3drobotics.com, n.d., IRIS+. Available from:<http://s3.amazonaws.com/3dr.production/1257/zoom/Iris_5.jpg?1410123478>. [1 March 2015].

Greg Welch & Gary Bishop, 2006, An Introduction to the Kalman Filter. Available from:< http://www.cs.unc.edu/~welch/media/pdf/kalman_intro.pdf>. [1 February 2015].

Lucas Vago Santana, Alexandre Santos Brandao, Mario Sarcinelli-Filh & Ricardo Carelli, 2014, Trajectory Tracking and 3D Positioning Controller for the AR.Drone Quadrotor.

Available from: <http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6842321&pageNumber%3D42213%26rowsPerPage%3D75>. [1 February 2015].

Evan Ackerman, 2013, Is Spiri the Programmable Quadrotor You've Always Wanted?. Available from:< http://spectrum.ieee.org/automaton/robotics/aerial-robots/spiri-programmable-quadrotor-drone-kickstarter>. [1 March 2015].

Appmethod.com, n.d., Developing the Internet of Things: Parrot AR Drone. Available from:<http://blog.appmethod.com/developing-the-internet-of-things-parrot-ar-drone>. [1 March 2015].

Java.com, 2014, What is Java technology and why do I need it?. Available from:<https://www.java.com/en/download/faq/whatis_java.xml>. [1 March 2015].

Nick Dijkshoorn, 2012, Simultaneous localization and mapping with the AR.Drone. Available from:< http://www.nickd.nl/dl/thesis_Nick_Dijkshoorn.pdf>. [February 2015].

Parrot.com, n.d., New AR. Drone 2.0 now with ALTITUDE Pressure Sensor. Available from:<http://ardrone2.parrot.com/ardrone-2/altitude/>. [5 February 2015].

Robotappstore.com, n.d., How to program AR. Drone remotely over WIFI. Available from:<http://www.robotappstore.com/Knowledge-Base/How-to-Program-ARDrone-Remotely-Over-WIFI/96.html>. [5 February 2015].

Bilgin.esme.org, n.d., Kalman Filter For Dummies. Available from: <http://bilgin.esme.org/BitsBytes/KalmanFilterforDummies.aspx>. [23 February 2015].

msh-tools.com, n.d., ARDrone Developer Guide. Available from: <http://www.msh-tools.com/ardrone/ARDrone_Developer_Guide.pdf>. [23 February 2015].