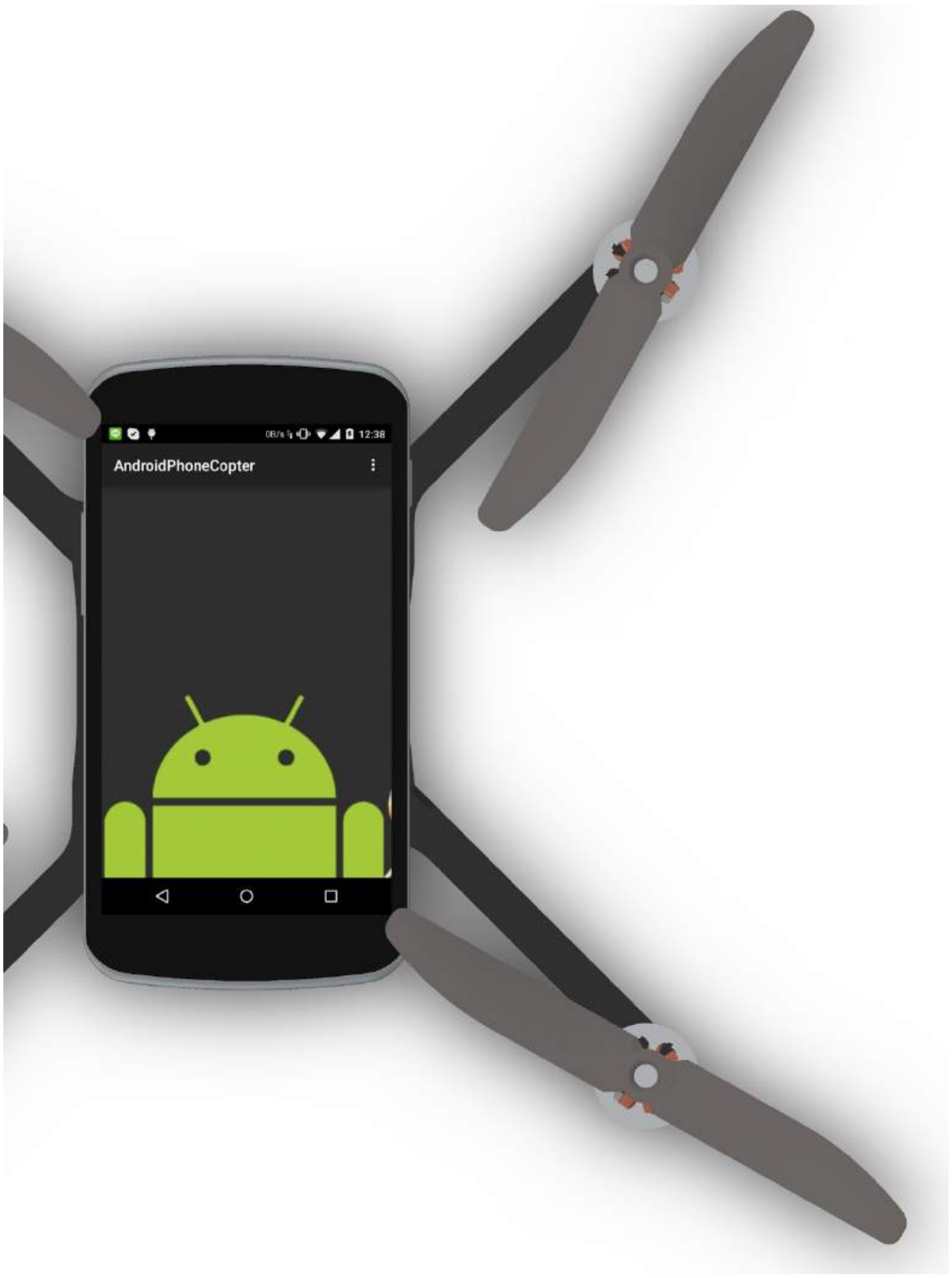


LYU1401 AndroidCopter

Supervised by Prof. LYU Rung Tsong Michael

Prepared by Lam Ka Ho (1155018465)

Abstract

In this project, we will implement a quadcopter fully based on Android System. We are trying to make use of the sensors (accelerometer, gyroscope and barometer, etc) on Android phone plus four motors, no other existing flight control board is needed. For this reason, our project is called “AndroidCopter”.

The report contains four parts. Firstly, we will talk about how we started the project and the study on the quadcopter, IOIO board and optical flow sensor. Secondly, we will show how the AndroidCopter is being implemented step by step. The basic software and hardware design of the entire system will be introduced. Furthermore, we will introduce the AndroidCopter API which is an extension for other developers. Finally, the testing, limitations and difficulties of AndroidCopter are also discussed.

Table of Contents

Abstract 2

Table of Contents 3

1 Introduction 7

1.1 Overview 7

1.2 Motivation 8

1.3 Objectives 10

1.4 How We can Achieve the Goal 11

1.5 Summary of Term one 13

2 Study of Traditional Multicopters 14

2.1 Hardware 14

2.2 How it can fly? 17

3 Study on System Components 19

3.1 Study of IOIO Board 19

3.2 Hovering in a Point and Path tracking using Optical Flow Sensor 21

3.3 Study of Visual Programming Language 26

4 AndroidCopter 28

4.1 Overall Architecture 29

5 AndroidCopter - Core 31

5.1 What Hardware is Needed? 31

5.1.1 Android Phone 31

5.1.2 Quadcopter Frame 32

5.1.3 IOIO Board 33
5.1.4 Sonic Sensor 33
5.2 What Software is Needed? 34
5.2.1 Android Version 34
5.2.2 Android Permissions and Features 34
5.3 Design 35
5.3.1 AndroidCopter Layout 35
5.3.2 Wiring Diagram 36
5.3.3 Class Diagram 38
5.3.4 Stabilization Algorithm 39
5.3.5 Data Flow 43
5.4 Interface of Android Application 46
6 AndroidCopter - Remote Control Panel 47
6.1 Implementation 47
6.2 Design UI Design 48
7 AndroidCopter API 55
7.1 Overview of the API 55
7.2 Strength of the API 56
7.3 Example Usage 56
7.4 Classes and Methods 57
8 Design Diagram 63

8.1 Use Case Diagram 63
8.2 Sequence Diagram 64
8.3 JSON Data Format 72
9 Safety Measurement 80
9.1 Arming and Disarming Features 80
9.2 Ranging Limitation using iBeacon 80
10 Testing and Experiment Data 82
10.1 Testing in Term One 82
10.2 Testing in Term Two 83
10.2.1 Stable Flying Height 83
10.2.2 Best Ground Surface 83
10.2.3 Brightness Testing 85
10.2.4 Path Tracking Testing 86
11 Current Limitations of AndroidCopter 96
12 Difficulty of the project 97
13 Usage 98
14 Visions 100
15 Conclusions 103
16 Contributions 104
17 References 105
Appendices 107

Appendix 1: Casualty List 107

Appendix 2: Product List 107

Appendix 3: Google Play 108

1 Introduction

1.1 Overview

Nowadays, Android is a popular operating system for mobile devices. From recent statistics, Android takes up nearly half of the market share of the mobile operating system in 2014.

Due to the popularity and open-sourced nature of Android, Android can be the operating system of many different electronics. We have Android phone, Android TV, Android Gear (smart watches) and Android Auto (automobiles mobile devices).

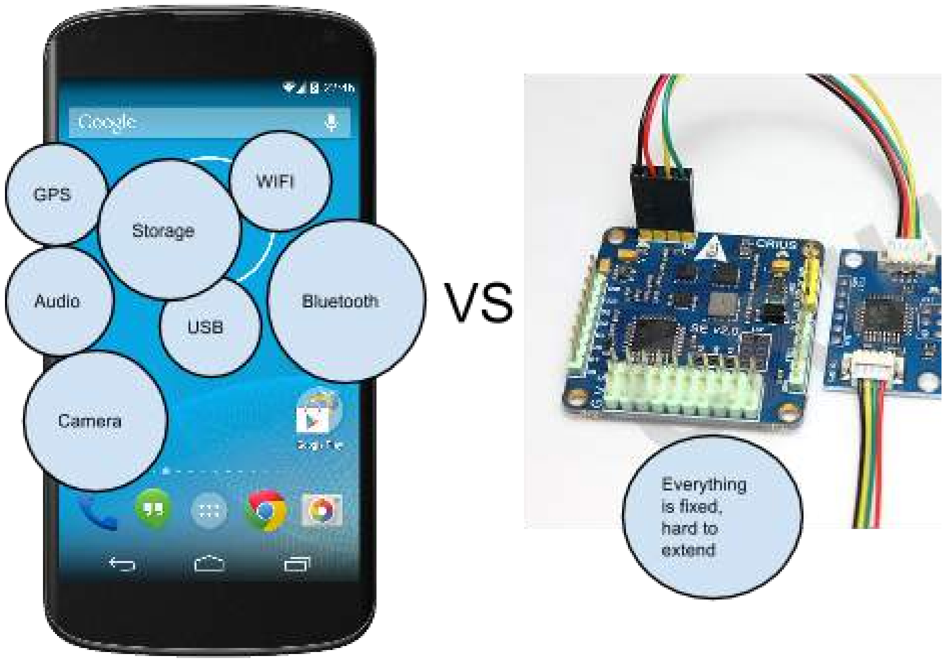
With infinite possibility of Android extension, it is expected more and more amazing Android application will be developed in the coming years. In this project, we will try to explore the possibility of Android. We will try to equip a quadcopter with Android and develop some meaningful features on it. More, we will build a quadcopter API which others developers can make use of the features and extend more possibility with it.

1.2 Motivation

Cool Idea

It is an interesting idea that only one Android phone and four rotors can build a quadcopter. The Android phone can fly in the sky. Next, we are curious on the applications can be developed with Android system on a quadcopter.

Absent of Quadcopter API



Quadcopter is useful nowadays in many applications such as rescue, inspection and aerial mapping. Although there is already quadcopter applications which are dedicated for specific functions, these applications usually cannot be extended easily by other developers. Also, most of them are written in Assembly or C/C++ language in which developers will need the knowledge on machine languages in order to extend it. The idea of making an Android API for quadcopter developer becomes our topic. We would like to make an easy and efficient quadcopter API for quadcopter developers which shorten the development time and simplify the development environment.

Existence Software and Hardware Support on Android

On the hardware side, Android phone is a powerful electronic as many optional components can be equipped. The hardware components include accelerometers, gyroscopes and barometers. The hardware support equips features for Android phone, Android phone is no longer just a mobile phone, it can be pedometer, smart card reader and location tracker etc. There is hardware support for Android phone which makes them possible to become part of the flight controller of a quadcopter.

On the Software side, Android has a huge community of Applications, Application Frameworks and Libraries. Many support libraries can be found, developers can focus on the programming on program flow and logic. There are libraries for iBeacon, Bluetooth and Websocket which provides functionality to the developers. Due to the object-oriented nature of JAVA, it is easier for extension. It provides better environment for extension.

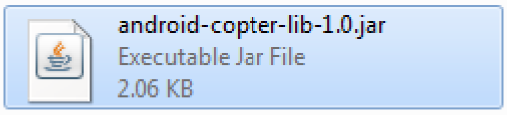
1.3 Objectives

Replace Flight Control Board with Android Phone

Our main objective is replacing the flight control board. We would like to bring the Quadcopter to the Android family and implement it on Android. So an Android phone can control the motors of a quadcopter directly.

Build a Super Easy API for extensions

When a developer need to develop a custom quadcopter application, he need to know many background information of quadcopter. He will need to know what values of input are needed to make it move and the flight control theory. However, not much developers know these knowledge before the development. The study will takes up a lot of time. Also, a lot of programming is needed to make the quadcopter fly normally. We believe many quadcopter developers will have a hard time in the initial stage of their project. To reduce the difficulty faced by quadcopter developers, we would like to take up the challenge to develop an easy API for other developers. With just few method calls, it will be able to perform some tasks.



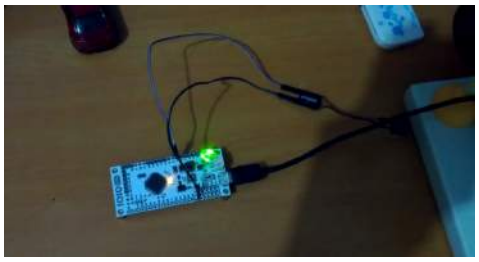
android-copter-lib-1.0.jar
Executable Jar File
2.06 KB

1.4 How We can Achieve the Goal

It is hard for us to make a AndroidCopter. It is because we did not owned or played any quadcopters before we started the project. Also, we did not know much about electronic. So, our strategy was removing the components of the quadcopter step by step to achieve the goal.

1. We have tried to control a normal quadcopter in early summer time. We have got some experiences on controlling a normal quadcopter.

2. We have tried to study the IOIO controller board. We have written some simple programs that can turn on or turn off the LED and generate signals to speaker of the Android phone.



3. At the end of the summer, the android phone is able to connect to the quadcopter with the IOIO Board. Also, we have studied how to program an IOIO board to make the communication between the Android phone and the flight control board becomes possible. Additionally, we have added the joystick control so the RC 2.4G transmitter is no longer needed. However, it was still using the flight control board.

4. In September, we planned to fully replace the legacy flight control board with an Android phone. So we started to get some sensors data and implement the flight control algorithm with the sensors data on Android.

5. At the same time, we have tried to build the Remote Control Panel, including WebSocket server and client using PHP, Javascript and HTML.

6. In the middle of September, the initial version of Android App was finished. The flight control board was replaced by the Android Phone finally.

7. We have tested multiple parts of the application. We have written some test cases and test them out. Troubleshooting and bug fixing after testing are needed sometimes. Moreover, photo taking and video recording function were implemented.

8. In November, we kept tuning and testing the AndroidCopter in order to make it more stable.

1.5 Summary of Term one

In term one, we have finished a lot of goals. Firstly, the Android Copter can stabilize for a period of time. Secondly, we have developed the Android Application, Remote Control Panel and web socket server to support the operation of the Android Copter. Users can send commands to the Android Copter via Remote Control Panel. Upon receiving commands, the Android Copter will perform required operations. Thirdly, The Android Copter can take video and photo when it receive commands. The photo taken will be sent back to the Remote Control Panel for a preliminary preview. Both video and photo will be stored in the Android Phone.

Next, we have planned some goals for term two. Firstly, the optical flow algorithm can be used to improve the stabilization. Using the optical camera facing the ground, we will be able to get back real distances moved based on the sensor’s values. It can improve the accuracy of attitude and provide better and accurate flight.

Secondly, autopilot can be equipped to support more meaningful operations. For example, the Android Copter can be scheduled by route. The route gives commands like upward and backward. The Android Copter will follow the route.

2 Study of Traditional Multicopters

2.1 Hardware

Frame

Frame holds all components of copters together. It should be rigid, lightweight and able to absorb as much vibration from motors as possible.

Rotors

Rotors (brushless motors) provide thrust to control the copters. By using the speed controller, different rotors are controlled separately.

Propellers

Each rotor mounted with a propeller. Propellers provide lifting thrust to copter. For example, there are four propellers for Quadcopter. Front and back propellers are turning to the right. Left and right propellers are turning to the left.

Electronic Speed Controller

The rotors are multi-phased, usually with three phases. The Electronic Speed Controller can generate three high frequency signals to keep the rotors rotate in different phases.

Battery

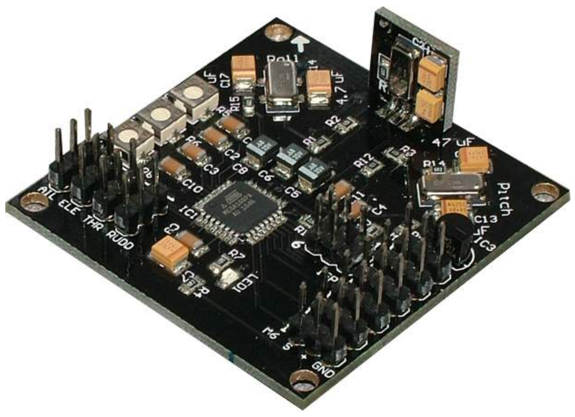
The Battery is the power supply of the copter.

Inertial Measurement Unit

The Inertial Measurement Unit is the sensor measurement unit of the copter. The Inertial Measurement Unit composes of accelerometer, gyroscope and magnetometer. It measures the orientation, velocity and gravitational forces of the copter.

Flight Controller

Flight Controller is a controller board which control the components of the copter.

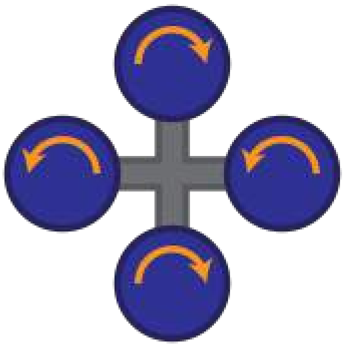


RC Transmitter

RC Transmitter is the joysticks for user to control the motion of the copter.

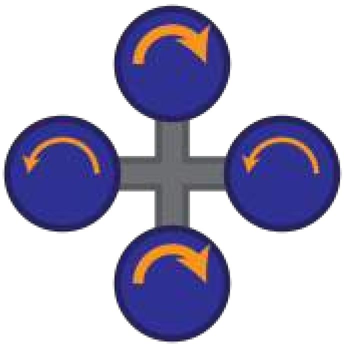
2.2 How it can fly?

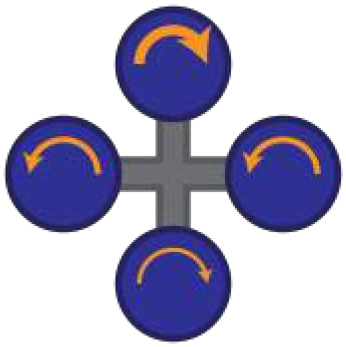
Multicopters utilize fixed-pitch blades. By changing the relative speed of different rotors, the thrust and torque produced change, thus control the motion. There are four degree of control for multicopter, they are yaw, roll, pitch and altitude.



To adjust the altitude of the copter which makes the copter go up or down, it is controlled by turning up or turning down the speed of all rotors.

To adjust the yaw of the copter which makes the copter turn left or right, it is controlled by turning up the speed of the rotating rotors in the same rotational direction and turning down the speed of rotors in counter rotation.





To adjust the roll of the copter which makes the copter go to left or right, it is controlled by turning up the speed of one rotating rotor and turning down the speed of rotor on the opposite side.

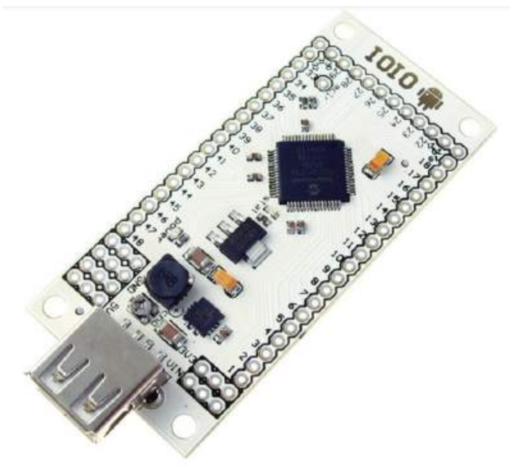
To adjust the pitch of the copter which makes the copter move forward and backward, it is controlled by similar method as roll.

3 Study on System Components

3.1 Study of IOIO Board

3.1.1 What is IOIO board?

IOIO board provides the capability for the Android devices in communicating with external hardware. It was originally designed for Android devices, later version can be worked on Android device and computers. Another similar example is Arduino.

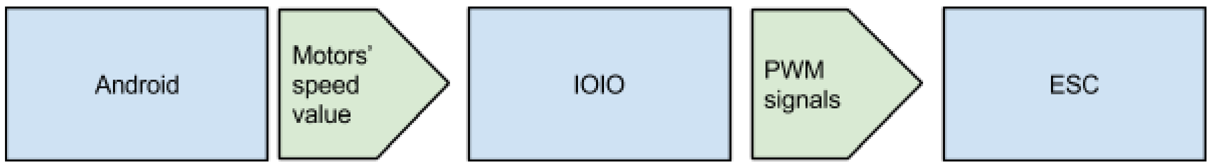


3.1.2 How the IOIO board work?

Firstly, the host (Android device) connects the IOIO board with USB or bluetooth. Secondly, the developer can start to program using the Java API in the host in order to utilize the I/O functions. The board has 46 pins for input/output, it supports pulse input/output, analog input/output and digital input/output.

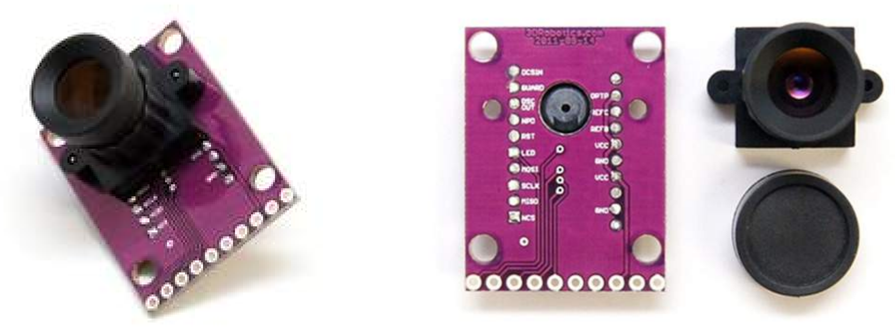
3.1.3 Usage in this projects

In this project, the IOIO board is the 'bridge' between Quadcopter and Android. Firstly, the Android application calculates the motors' speed and then send the values to IOIO. After that, the IOIO will convert the values to PWM signals. Finally, the signals will be passed to the speed controllers (ESC). This illustrates how an Android phone can control the motors directly using Android SDK.



3.2 Hovering in a Point and Path tracking using Optical Flow Sensor

3.2.1 What is Optical Flow Sensor?



From the pattern of objects, the movement between the camera and the scene can be estimated. The optical flow sensor contains of a mouse sensor and a camera to make film capture.

3.2.2 Why we must need this?

According to Newton's first law, a moving object will continue moving. If the quadcopter is going forward, it will not stop unless being interrupted by an external force. This is known as "Drifting". And normal hardware is unable to detect these kinds of movement accurately. For example, although the accelerometer is able to detect acceleration, the accelerometer is useless if the copter go with uniform speed. Besides, it seems that GPS should be another possible solution, we may calculate the delta of two GPS positions to detect such movement. However, the accuracy of mobile phone's GPS is limited at around two to three meters even in an outdoor environment and its accuracy is even worst in indoor environment.

To detect these kinds of movement, optical flow sensor is the best choice in this project.

3.2.3 How the Optical Flow Sensor work?

The optical flow sensor makes use of ADNS3080 mouse sensor. It supports high definition with 30 X 30 frames. It has a high speed with up to 6400 frames update rate per second. It also supports SPI interface which can communicate with different micro controllers and work with other sensors.

Although it can captures up to 6400 frames per second, the update rate is around 80 times per second as bounded by hardware limitations. The delta X and Y come back 80 times every second.

3.2.4 Why not use Camera instead of Optical Flow Sensor?

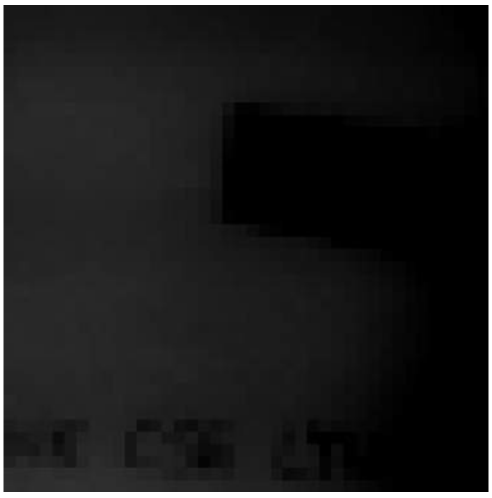
Firstly, the camera supports a lower frame per second (FPS) than the optical flow sensor. Secondly, the camera requires higher CPU power. Thirdly, the camera needs to record video.

3.2.5 Usage in this project

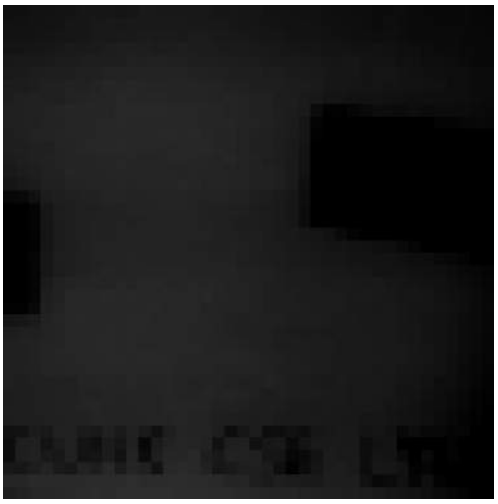
The optical flow sensor is mainly used for increasing the stability of the Android Copter. With better stability, functions of Android Copter including auto pilot and routing can be carried out with better performance. The Android Copter will be able to follow a predefined path and follow commands with better performance.

Frame dumps from the optical flow sensor:

This is the initial frame.



When the copter move to the left, the frame will become this.



Then the sensor will try to calculate the movement of objects between two frame and outputs the delta x and delta y values back to the Android Phone.

Hovering in a fixed Point

We can illustrate the idea using example of our copter. For example, we want the copter hover in a fixed point but the copter is drifting to the left currently without the sensor. With the use of the optical flow sensor, it will output data like { x: -4, y: 0 }. Upon receiving the data, the copter will try to go right in order to neutralize the force. And then the copter keeps doing this, the copter will hover in a fixed point.

Path Tracking

The second important usage is path tracking. This optical sensor is more or less the same as a optical mouse. When we are moving the mouse, the mouse sensor will try to track the surface on the mouse pad, outputs the value to the computer and calculate the movement of the mouse pointer on the screen.

The sensors of an optical mouse and an optical flow sensor are almost the same, but the usage is a bit different. The main purpose of optical flow sensor is like "trying to ask the mouse pointer to follow the red line from left to right". If you are browsing this report using a mouse, you may try. Normally, you should be able to following the red line.

And imagine that your mouse together with your hand is a quadcopter with a optical flow sensor. And the red line is a "go-right(1.0 meter)" command. Normally, the copter is able to follow the command.

Last but not least, the sensor's output delta x and delta y has no specific units. To convert output to meter, we have to find out the constant in a certain height.

Delta X * K = Delta X Distance in Meter

Delta Y * K = Delta Y Distance in Meter

3.2.6 Optical Flow Sensor Application

Odometry

With the use of motion sensors, calculate the change in position from a period of time.

Obstacle avoidance

Avoid interaction and collision with other objects.

3.2.7 Limitation

Firstly, it works in environment with sufficient light. Secondly, it fails to work in environment with Fluorescent lights. Thirdly, it works in surface with variation in order to show the moment obviously.

This inexpensive optical flow sensor is unable to focus automatically. The copter cannot fly too high, it is because it may captures unclear objects if it is out of focus.

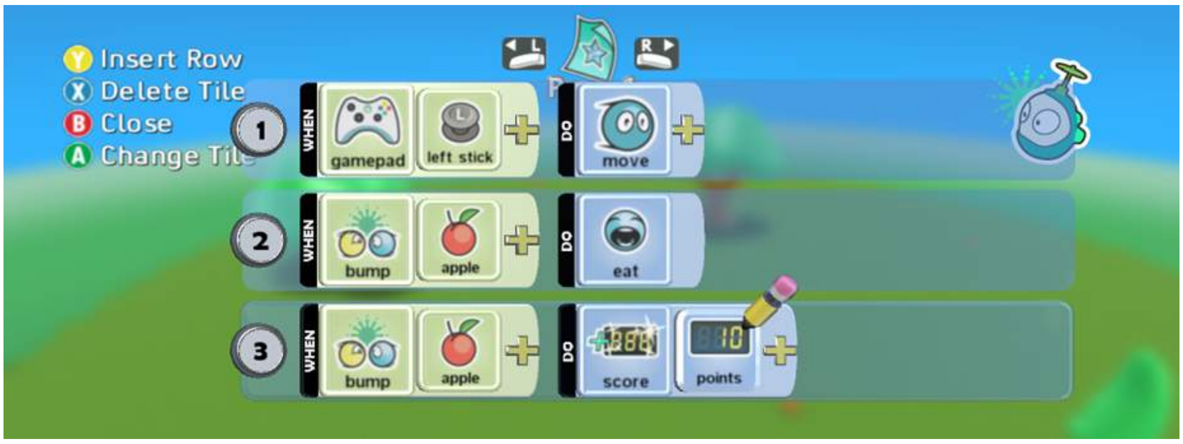
3.3 Study of Visual Programming Language

3.3.1 What is Visual Programming Language?

Visual Programming Language (VPL) provides an environment for users to develop their program using graphical representation. The entities and relation are represented using graphical view. For example, entities are blocks while relation are edges. VPL has many types of visual style, like representation using flow chart and UML are common.

3.3.2 How the Visual Programming Language work?

The VPL provides a simple user interface and easy to learn environment for any user to program. It can shorten the time of development. More, it is widely used in education, multimedia and game industry.

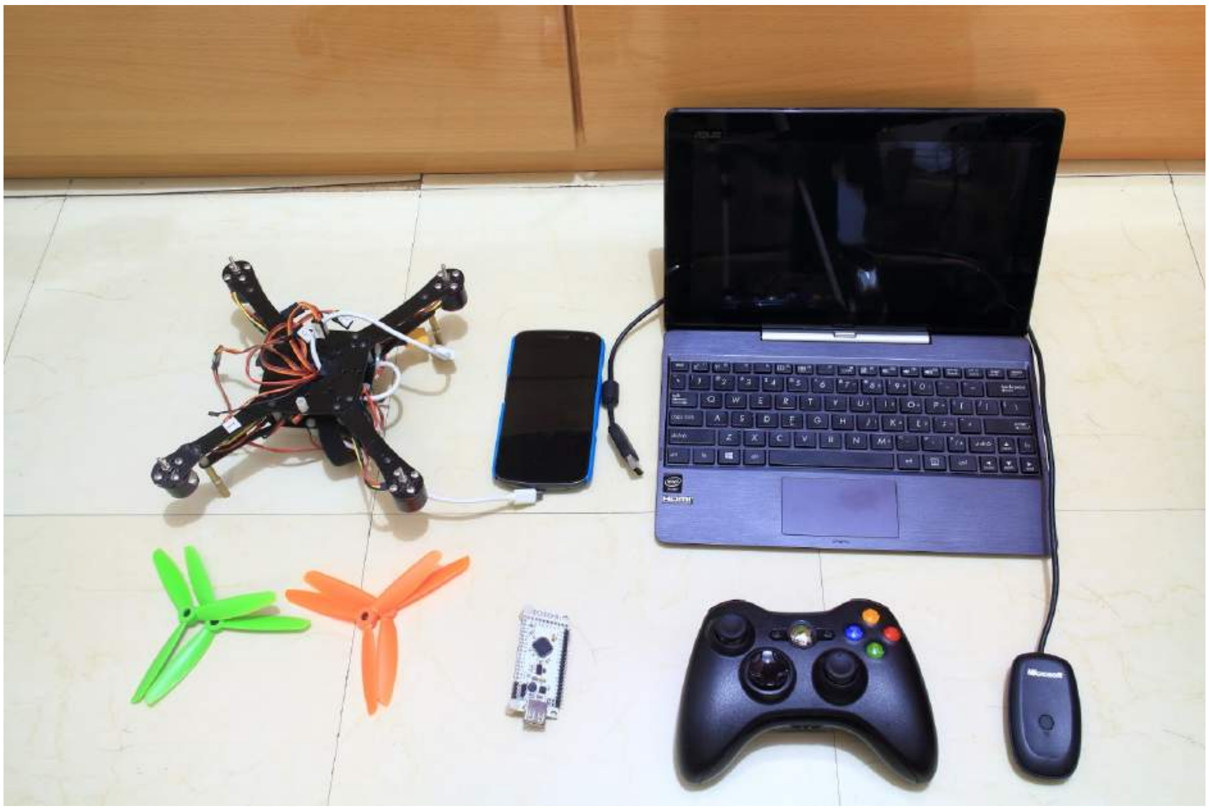


3.3.3 Usages in this project

The VPL will be used as a routing tools in our project. User can drag and drop commands using VPL in the Remote Control Panel. They can assemble commands into small program. The program will be sent to the Android Copter for the purpose of auto pilot. The operation is very simple, even a primary student will be able to control it.

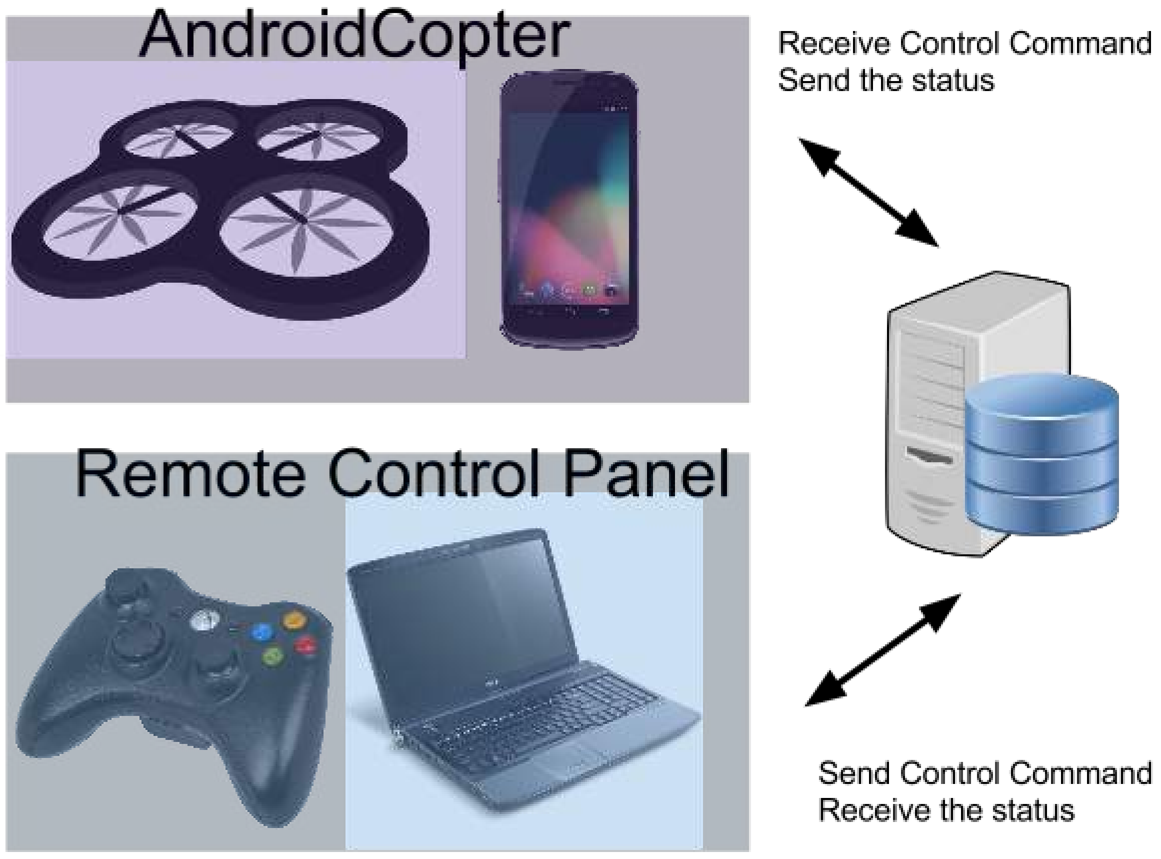
4 AndroidCopter

To start our project, we need an Android phone which is able to establish a connection to the quadcopter.



4.1 Overall Architecture

The system can be divided into **three** parts, including the AndroidCopter Core Part, the Remote Control Panel and the WebSocket server.



AndroidCopter Core Part

AndroidCopter Core Part consists of the Quadcopter Frame and the Android phone.

Remote Control Panel

Remote Control Panel consists of the PC and the Joystick. The joystick is optional for manually control. we highly recommend the user to use the joystick instead of the PC keyboard for better control.

WebSocket server

Although the WebSocket server can be installed on the same PC with the Remote Control Panel, the operation of PC usually restricted by the firewall. For this reason, the WebSocket server should be installed on a dedicated server on the Internet. Also, WebSocket has higher accessibility than normal socket. WebSocket provides a communication channel across different browsers. In other words, the Remote Control Panel can be used on different platforms such as Android, iOS, Linux or Windows.

5 AndroidCopter - Core

The core part of AndroidCopter is formed by a quadcopter frame set and an Android mobile phone.

For the programming part of the flight controller, Android SDK is used, it is written in **JAVA** language. In order to achieve the goal of project, Android SDK is not enough, some external libraries are used in the project such as **IOIO Library** (For IOIO board communication), **Google Play Services** (For Fused Location), **AndroidAsync** (For WebSocket communication) and **AltBeacon** (For scanning iBeacon).

5.1 What Hardware is Needed?

To make a working AndroidCopter, the following requirements are needed.

5.1.1 Android Phone

Sensor Requirement

The Android phones are required to have an **accelerometer** and a **gyroscope** sensors. Without any of them, the copter will be extremely unstable in the air. Besides, the Android phones must support IOIO Board [1]. More, the phone is able to connect to Wi-Fi network at least. These are minimum requirements for Android phones as a flight controller. That means it can be controlled by human only, but it cannot fly automatically, as the copter do not know the current direction and the current flying height.

For advanced features such as flying from one point to another point, a **magnetic sensor**, a **barometer** (pressure sensor) and a **GPS sensor** are required. The barometer helps the copter to keep pose in a certain height, for example, keeping the copter in 50 meters in the air. In the program, atmospheric pressure can be converted to the height. And the magnetic sensor helps to determine current direction, or else it cannot know which direction it should go. The copter retrieves current position with the GPS sensor.

Network Requirement

Additionally, the phone is recommended to have a working SIM card which is able to connect to 2G, 3G or LTE network reliably, so that users can keep tracks and control their quadcopter through remote control panel when the quadcopter is far away from the user.

Weight Requirement

To have a better flight, the Android phone should be lightweight. If the phone is too heavy, the motors' output will be higher. In other words, it consumes more energy in the air, so the flight time will be shortened.

Currently, there are not many phones which can fulfill all the requirements stated, neither because of missing sensors or not supported by IOIO Board. One of the supported devices is Google Galaxy Nexus.

5.1.2 Quadcopter Frame

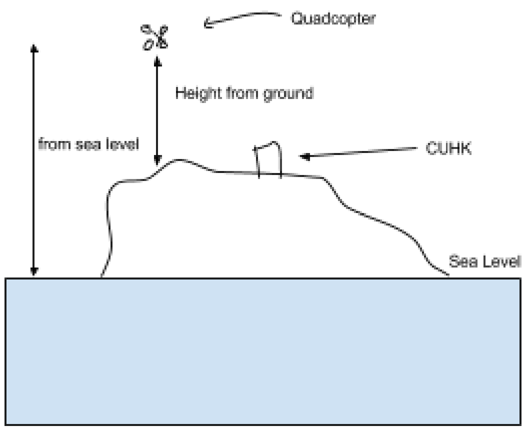
Most frame sets with **brushless electronic speed controls (ESCs)** and **brushless motors** are supported.

Currently our flight controller Android application is focusing on the quadcopter with 'X' layout. So other frames such as helicopter, tricopter or hexacopter are not supported by our program currently. The quadcopter frame size is not limited. However, the brushless motors should be powerful enough to carry the weight of the frame and the mobile phone.

5.1.3 IOIO Board

As previously described, it is the bridge between the quadcopter frame and the Android mobile phone.

5.1.4 Sonic Sensor



This is optional. It is used for auto landing function. The sonic sensor is installed at the bottom of quadcopter and facing the ground. The flight controller can use it to get the height from the ground by some calculation. The reason why we cannot use the barometer to calculate the height from the ground is because the height measured by barometer is the distance from the sea level.

5.2 What Software is Needed?

The requirements of software are much fewer than hardware.

5.2.1 Android Version

Currently, Android 4.0 or newer are supported. Theoretically, Android 2.3 should be supported, but most Android phones with 2.3 have a slower computational speed. To have better performance, Android version 4.0 or newer is recommended.

5.2.2 Android Permissions and Features

android.permission.WAKE_LOCK

The CPU must keep running in order to keep the application running.

android.permission.INTERNET

If user want to control the AndroidCopter through the Remote Control Panel, the Internet permission is needed.

android.permission.CAMERA and android.hardware.camera.autofocus

The application can take photos or record a videos in the air.

android.permission.WRITE_EXTERNAL_STORAGE

After photos were taken or videos were recorded, the related media files will store in the external storage.

com.android.future.usb.accessory

IOIO board is a USB accessory in Android environment, so the application needs the permission to connect USB devices.

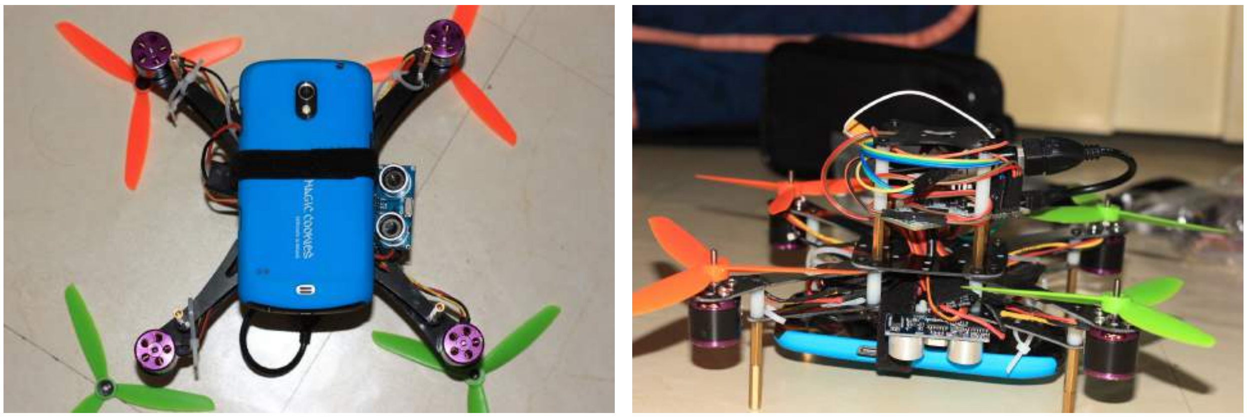
android.permission.BLUETOOTH

iBeacon is needed. Besides, the alternative way to connect IOIO board to the Android is using Bluetooth.

android.permission.ACCESS_FINE_LOCATION

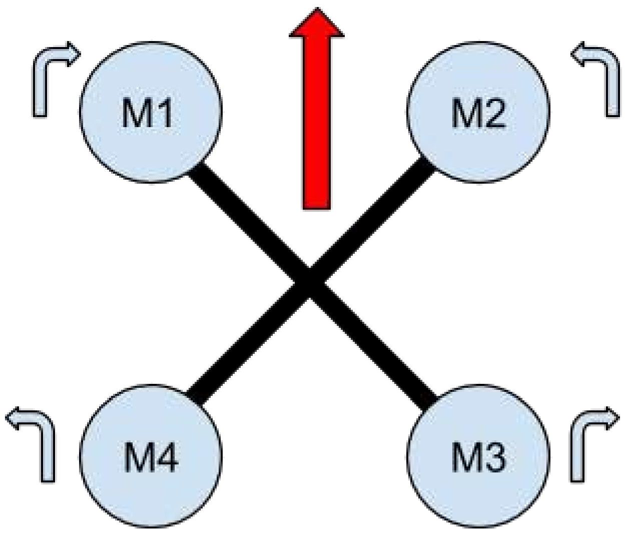
The application can get the GPS information with the permission.

5.3 Design



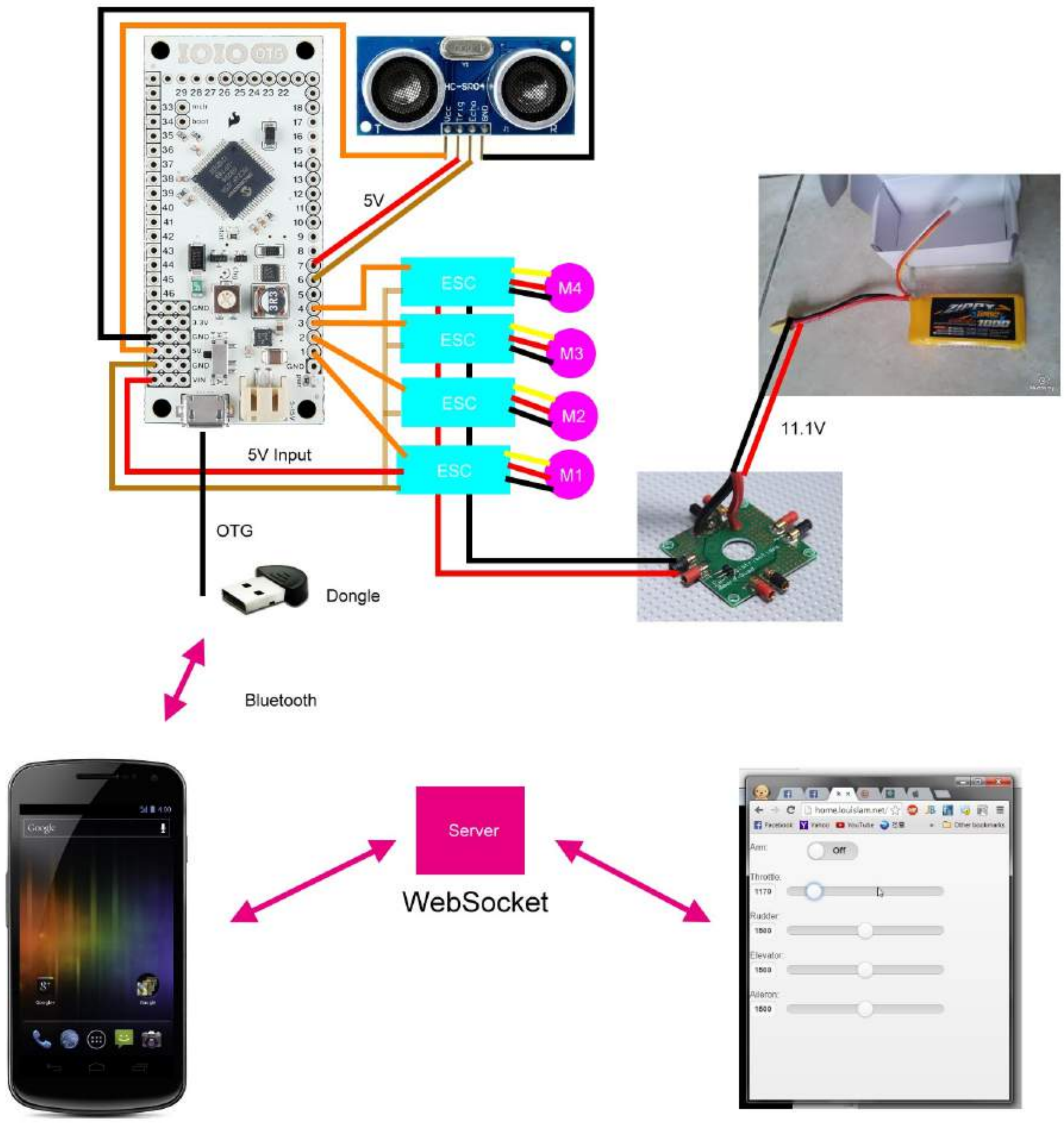
230mm quadcopter frame is used for the project. The mobile phone is installed below the frame, this installation can capture clearer photo. The IOIO board is installed on the frame.

5.3.1 AndroidCopter Layout



In our design, X layout of quadcopter is used. Assuming the head of the quadcopter is north. Motors M1 and M2 are the front motors, and motors M3 and M4 are the rear motors. M1 and M3 are spinning clockwisely, M2 and M4 are spinning anti-clockwisely.

5.3.2 Wiring Diagram



Arm: Off
Throttle: 1170
Rudder: 1500
Elevator: 1500
Aileron: 1500

Wiring Diagram description

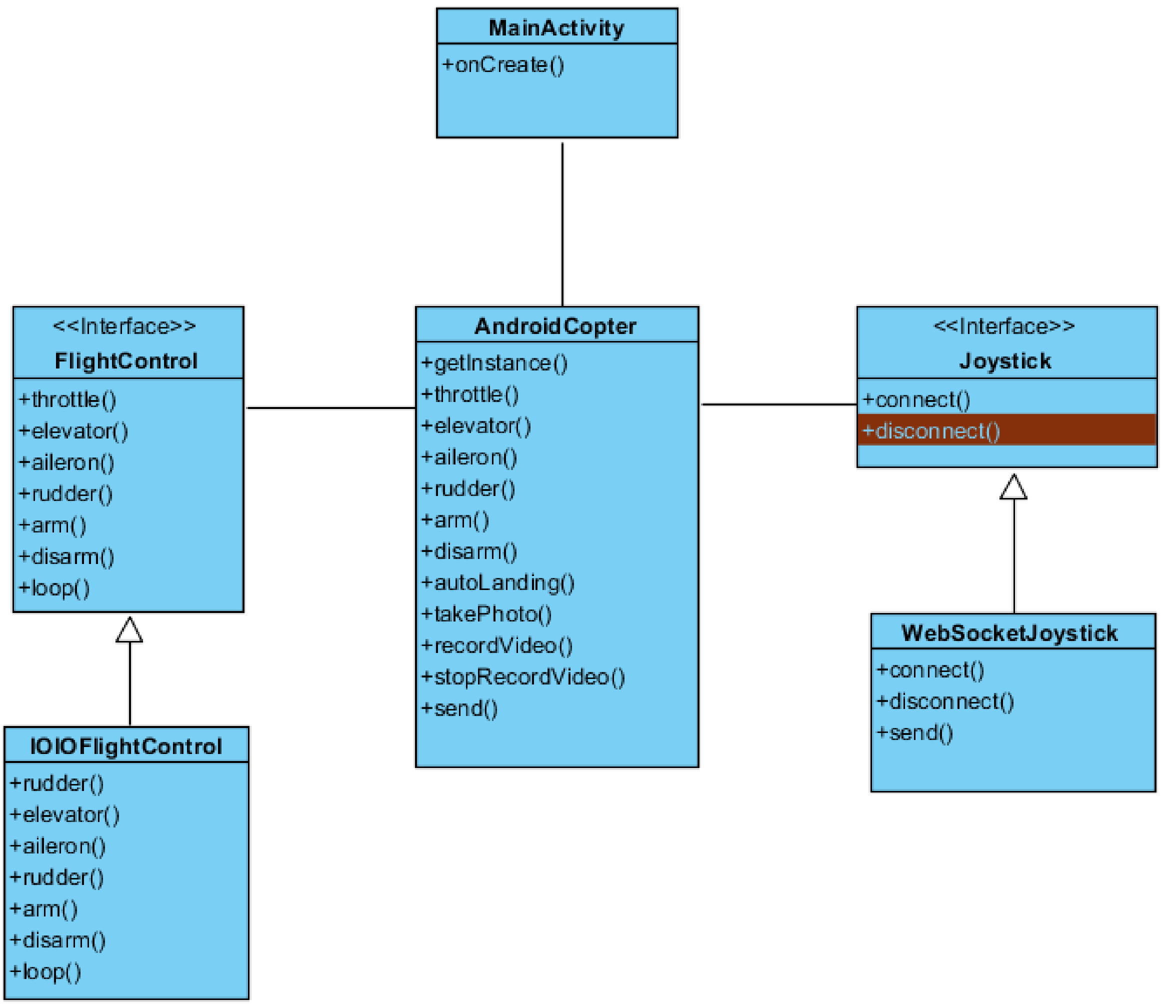
The basic setup is very similar to the traditional quadcopter.

1. The battery is connected to the **power distribution board**.
2. All **speed controllers (ESCs)** is connected to the power distribution board.
3. Each motor is connected to its ESC.

4. Each ESC's PWM cable (orange) is connected to the IOIO board, the pin number range is 1 to 4. If an ESC is connected to the M1 motor, then the ESC should be connected to IOIO board's pin 1, so on and so forth.
5. To supply power for the IOIO board, one of ESC's power is used. The ESC's red cable is connected to the IOIO board's VIN pin, and the ground cable (black) is connected to the IOIO board's GND pin.
6. If **sonic sensor** is needed, the ECHO pin is connected to IOIO board's pin 7, the TRIG pin is connected to IOIO board's pin 6, GND pin is connected to IOIO board's GND pin and VCC pin is connected to the IOIO board's V5 pin.
7. To connect the IOIO board and the Android phone, there are TWO possible ways. The first way is connecting them with an **OTG cable** and a **USB cable**. The second way is the IOIO side is connected to an OTG cable and a **Bluetooth dongle**. After that, the phone can pair up with the IOIO device via Bluetooth. Using Bluetooth may affect the performance since there may be latency. According to our experiment, it is still fast enough to update the motors' signal to IOIO board.
8. If **Remote Control Panel** is needed, the Android phone should be able to establish a connection via Wifi network or 3G network. And then the App is connected to WebSocket server.

Step 1 to step 6 are one-time-setup, after these steps are done, the user only need to connect the USB Cable to their Android phone and everything is going to be work.

5.3.3 Class Diagram



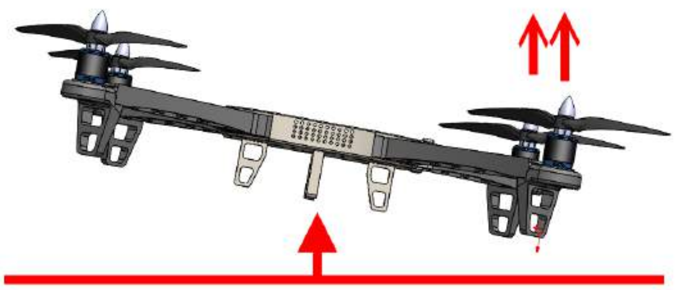
MainActivity is an Android Activity object. An AndroidCopter object can be created by calling getInstance() method. An AndroidCopter object has a FlightControl object and a Joystick object and both of them are interfaces. Currently, we have implemented IOIOFlightControl and WebSocketJoysitck, so that the android application can send control signal to quadcopter and we can control it via WebSocket.

Flight control algorithm or logic both are implemented in the IOIOFlightControl class.

5.3.4 Stabilization Algorithm

Stabilization (Auto Leveling) is one of the important topic in Quadcopter, or even the helicopter or the aeroplane. Without stabilization algorithm, a quadcopter is unable to keep stabilize in the air. In order to implement the stabilization algorithm, two sensors is needed: **Accelerometer** and **Gyroscope**.

Example Scenario:



Imagine that we want the quadcopter to keep in balance state, but the quadcopter is in an unstable state currently, like the picture in the right hand side. In common sense, the right set of motors just need to speed up in order to keep the quadcopter in balance state. However, as the quadcopter, there are TWO questions:

(1) How does the quadcopter know the current state?

(2) How does the quadcopter know that it should generate how many power to motors in order to keep the balance state?

The quadcopter is able to obtain the angular speeds of 3-axis from the Gyroscope, so the quadcopter is able to keep a pose in the air in short term. Regarding the X-axis only, we can make use of the following simple linear equation. (Assume that M1,M2,M3,M4 are motors output, 1500 is the user input, gyroX is the angular speed of X axis and the K_1 is a constant value)

```
m1 = 1500 - (gyroX * K1);
m2 = 1500 - (gyroX * K1);
m3 = 1500 + (gyroX * K1);
```

```
m4 = 1500 + (gyroX * K_1);
```

However, in long term, it is not quite working, because the quadcopter only know the angular speed, but it does not know whether it is balanced currently. So the quadcopter can get the actual angle from the **Accelerometer**. (Assume that accX is the accelaration of X-axis and K_2 is also a constant)

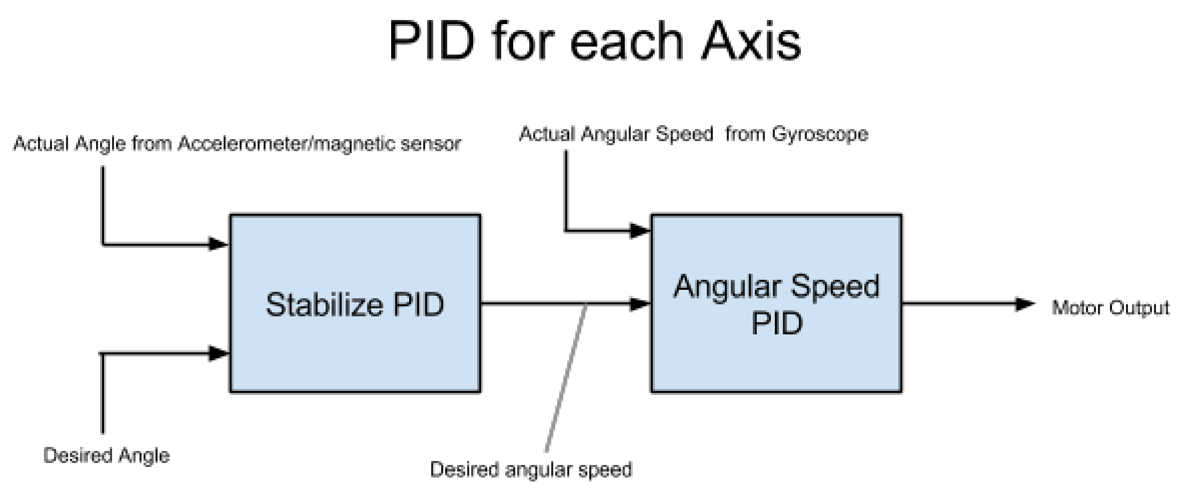
```
m1 = m1 - (accX * K_2);
m2 = m2 - (accX * K_2);
m3 = m3 + (accX * K_2);
m4 = m4 + (accX * K_2);
```

The constants(K_1, K_2) are given by the user due to the different frame size, different weight and different power of motors.

With the linear equation, it is able to balance in the air in the **ideal environment** or in **the simulation** only. In real world, there are many error and many outside force such as wind force. So PID controllers are needed for each axis.

The PID controller definition from Wikipedia: "A proportional-integral-derivative controller (PID controller) is a control loop feedback mechanism (controller) widely used in industrial control systems. A PID controller calculates an error value as the difference between a measured process variable and a desired setpoint. The controller attempts to minimize the error by adjusting the process through use of a manipulated variable."[2]

So PID controller is much better that a constant in the real situation. And the final design is like this:



The actual angle of X-axis and Y-axis can be obtained from the accelerometer and Z-axis is obtained from the magnetic sensor. Furthermore, the quadcopter can go forward/backward/left/right by changing the desired angle input, Combining everything, the code in the Android Applications like this:

PID Methods:

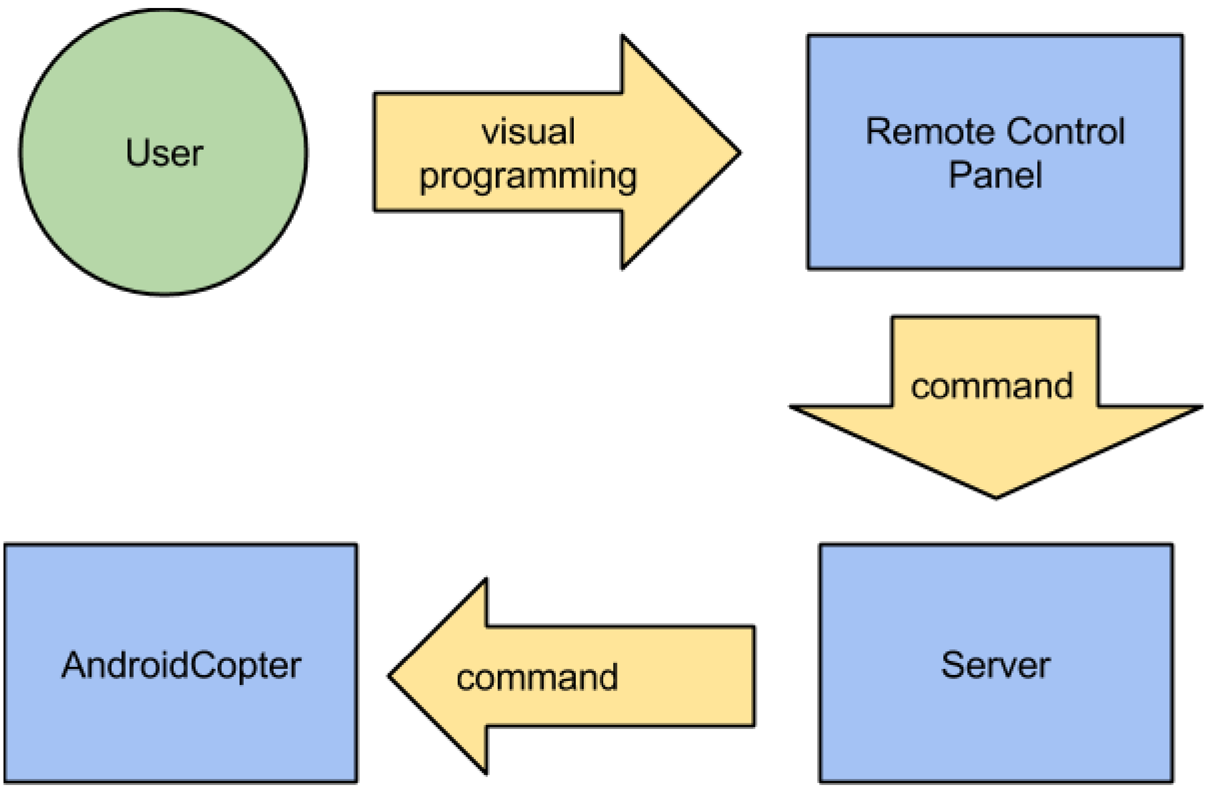
```
// Update PID
pid.update(actualValue, desiredValue);

// Get Output
pid.getOutput();
```

Algorithm in JAVA:

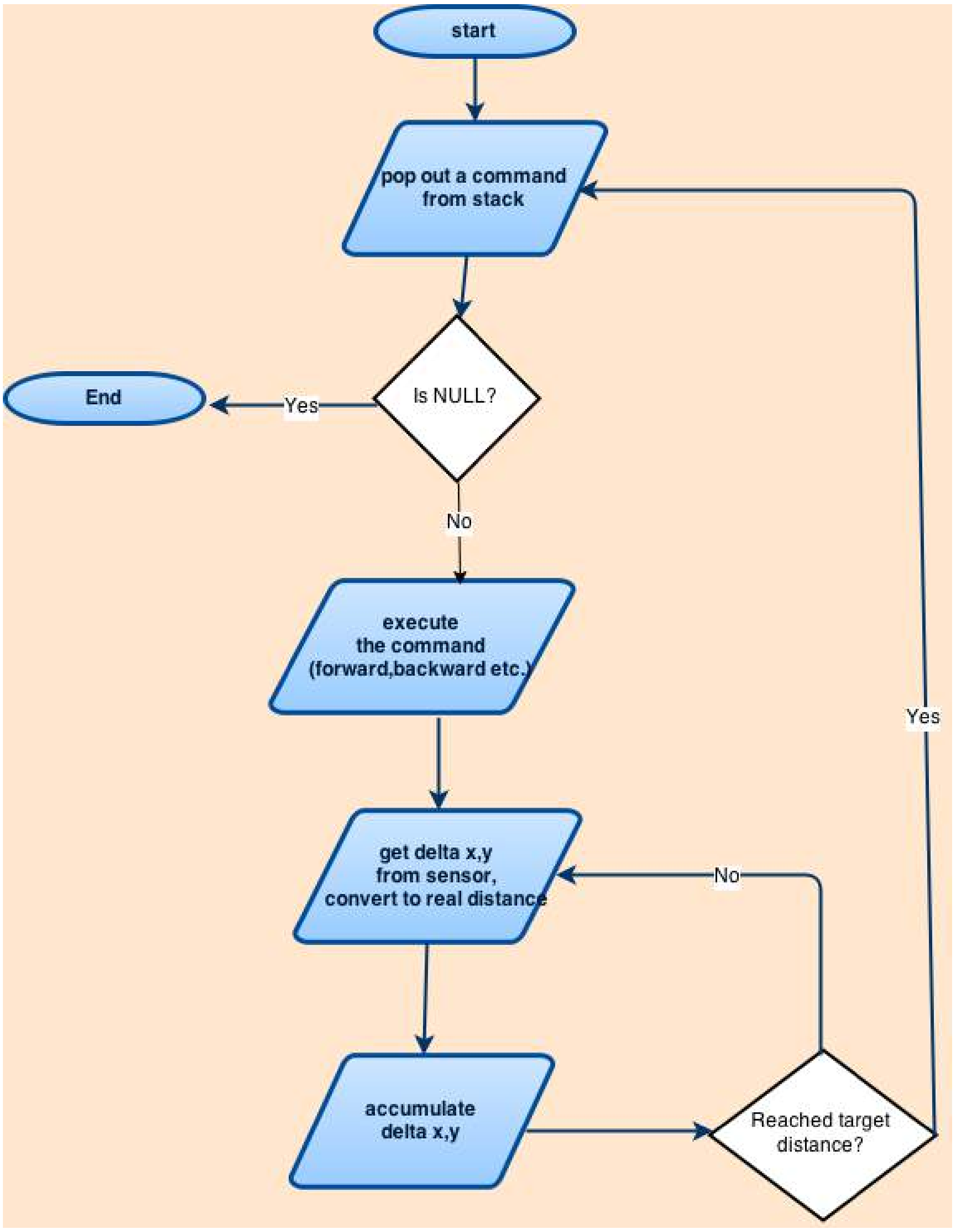
```
// Desired Throttle
m1 = m2 = m3 = m4 = throttle();

// PID for Stabilization
pitchStablePID.update(currentOrientation.getPitch(),
targetOrientation.getPitch());
rollStablePID.update(currentOrientation.getRoll(),
targetOrientation.getRoll());
```

```
yawStablePID.update(currentOrientation.getAzimut(),
targetOrientation.getAzimut());

// PID for Angular Speed
pitchPID.update(inputDegreePitch,
Util.radToDegree(currentGyro.getX()));
rollPID.update(inputDegreeRoll,
Util.radToDegree(currentGyro.getY()));
yawPID.update(inputDegreeYaw,
Util.radToDegree(currentGyro.getZ()));

// Pitch Axis (X)
m1 = m1 - (int) pitchPID.getOutput();
m2 = m2 - (int) pitchPID.getOutput();
m3 = m3 + (int) pitchPID.getOutput();
m4 = m4 + (int) pitchPID.getOutput();

// Roll Axis (Y)
m1 = m1 - (int) rollPID.getOutput();
m4 = m4 - (int) rollPID.getOutput();
m2 = m2 + (int) rollPID.getOutput();
m3 = m3 + (int) rollPID.getOutput();

// Yaw Axis (Z)
m1 = m1 - (int) yawPID.getOutput();
m3 = m3 - (int) yawPID.getOutput();
m4 = m4 + (int) yawPID.getOutput();
m2 = m2 + (int) yawPID.getOutput();
```

5.3.5 Data Flow



The user will produce their program using visual programming tools. The program will be created easily with drag and drop controls. Upon the Remote Control Panel receive the visual program, it will evaluate the program and send out related command to the server. Upon the server receive the command, it will pass the command to Android Copter. The Android Copter will perform appropriate operations according to the command.

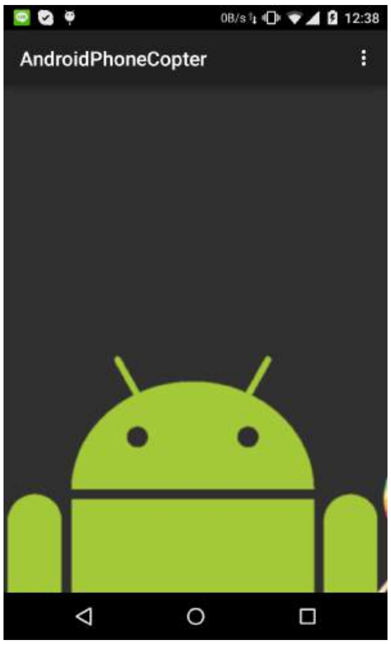

start
pop out a command from stack
Is NULL?
Yes
End
No
execute the command (forward,backward etc.)
get delta x,y from sensor, convert to real distance
accumulate delta x,y
Reached target distance?
No
Yes

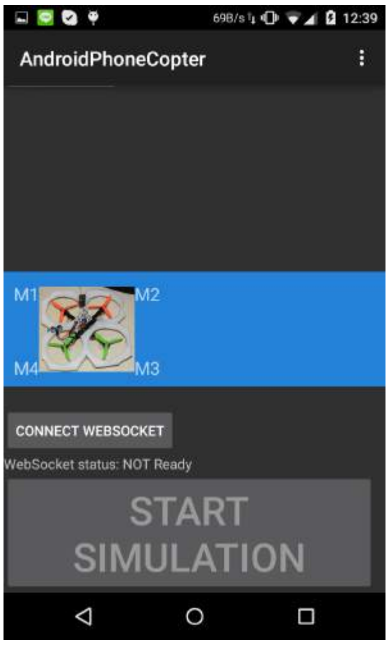
In the data flow aspect, Android Copter first pops out a command from the stack. If there are no items in the stack, it will end the program. Otherwise, it will execute the command being popped out.

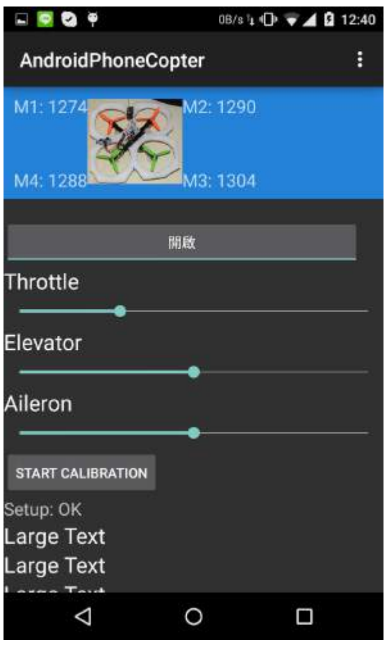
After that, it will retrieve the delta x, y from the sensor and convert them to real distances. It accumulates delta x and delta y. It keeps retrieving and accumulating delta x and delta y until the target distance has been reached. When the target distance has been reached, it will start another iteration.

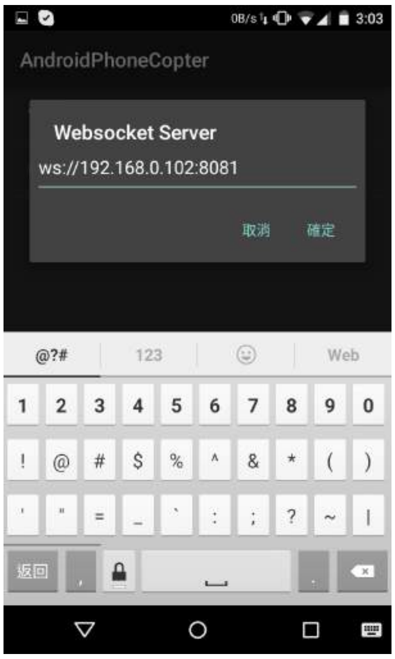
5.4 Interface of Android Application

The Android application provides many basic function of control. While users connected the IOIO board and start this application, the AndroidCopter will work immediately. For testing purpose, users can start the simulation, so that we can illustrate the output of motors.

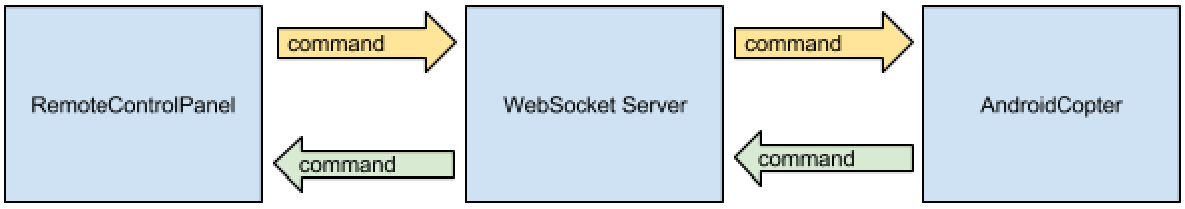








6 AndroidCopter - Remote Control Panel



The RemoteControlPanel establishes a connection with the WebSocket Server. The WebSocket Server will recognize the connecting party is RemoteControlPanel or AndroidCopter. RemoteControlPanel will send command which control the AndroidCopter via WebSocket Server. When AndroidCopter receive a valid command, it will process the command. When the command is processed, there may be some return values. If it is a updating command, there will not be any return value. If the command is a photo taking command, there will be an compressed image returned to the RemoteControlPanel.

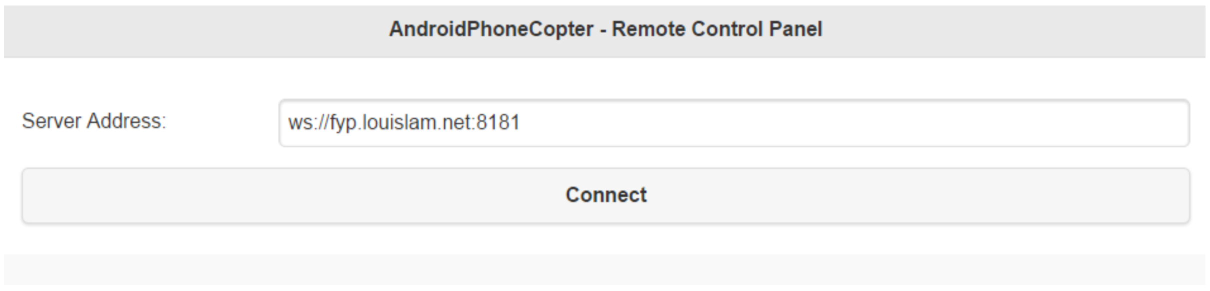
6.1 Implementation

The Remote Control Panel is a web-based control panel for users. No installation is needed to access it. The Remote Control Panel is implemented using php and javascript. The control panel utilizes jQuery Mobile library. The jQuery Mobile library supports responsive layout, the size and appearance of website component change against screen size to best fit with the user's screen. Also, websites have high accessibility as it can be viewed on different platforms and different operating systems.

6.2 Design UI Design

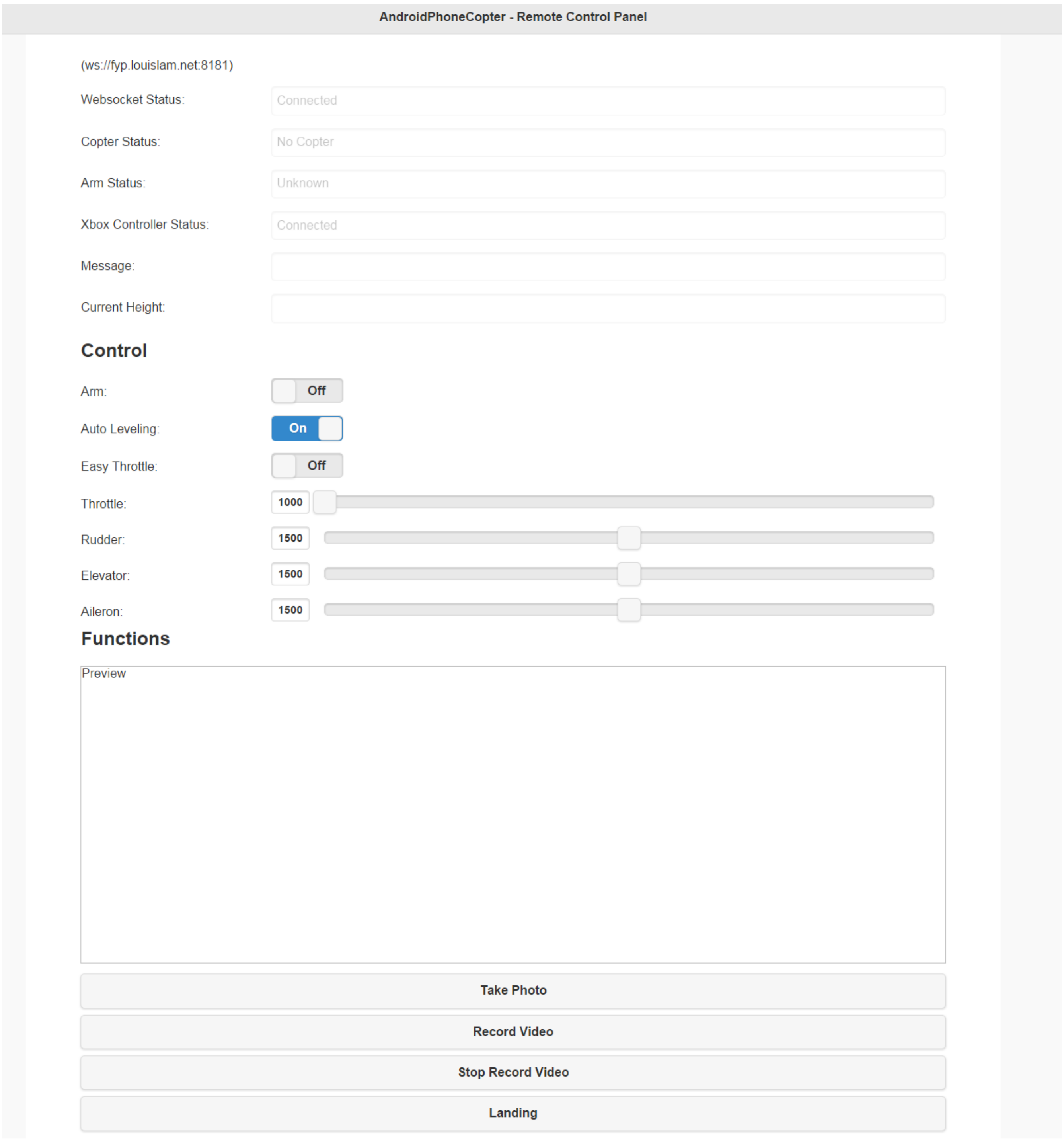
6.2.1 Connect Page

First, the user is required to enter the address of the Websocket Server. The address format is ws://hostname:port. We can further develop the application to enable secure connection is established using wss://.


AndroidPhoneCopter - Remote Control Panel
Server Address:
ws://fyp.louislam.net:8181
Connect

6.2.2 Control Page

The control pages provides important information to the users, for example, the quadcopter is armed or not. Also, it allows the users to changes the values of quadcopter. By changing the values, user is able to control the copter's motion. More, it can be connected to a joystick so that the user can control the copter using the joystick. Furthermore, photo taking and video recording can be triggered using the RemoteControlPanel.



AndroidPhoneCopter - Remote Control Panel

(ws://fyp.louislam.net:8181)

Websocket Status:	Connected
Copter Status:	No Copter
Arm Status:	Unknown
Xbox Controller Status:	Connected
Message:	
Current Height:	

Control

Arm:	Off
Auto Leveling:	On
Easy Throttle:	Off
Throttle:	1000
Rudder:	1500
Elevator:	1500
Aileron:	1500

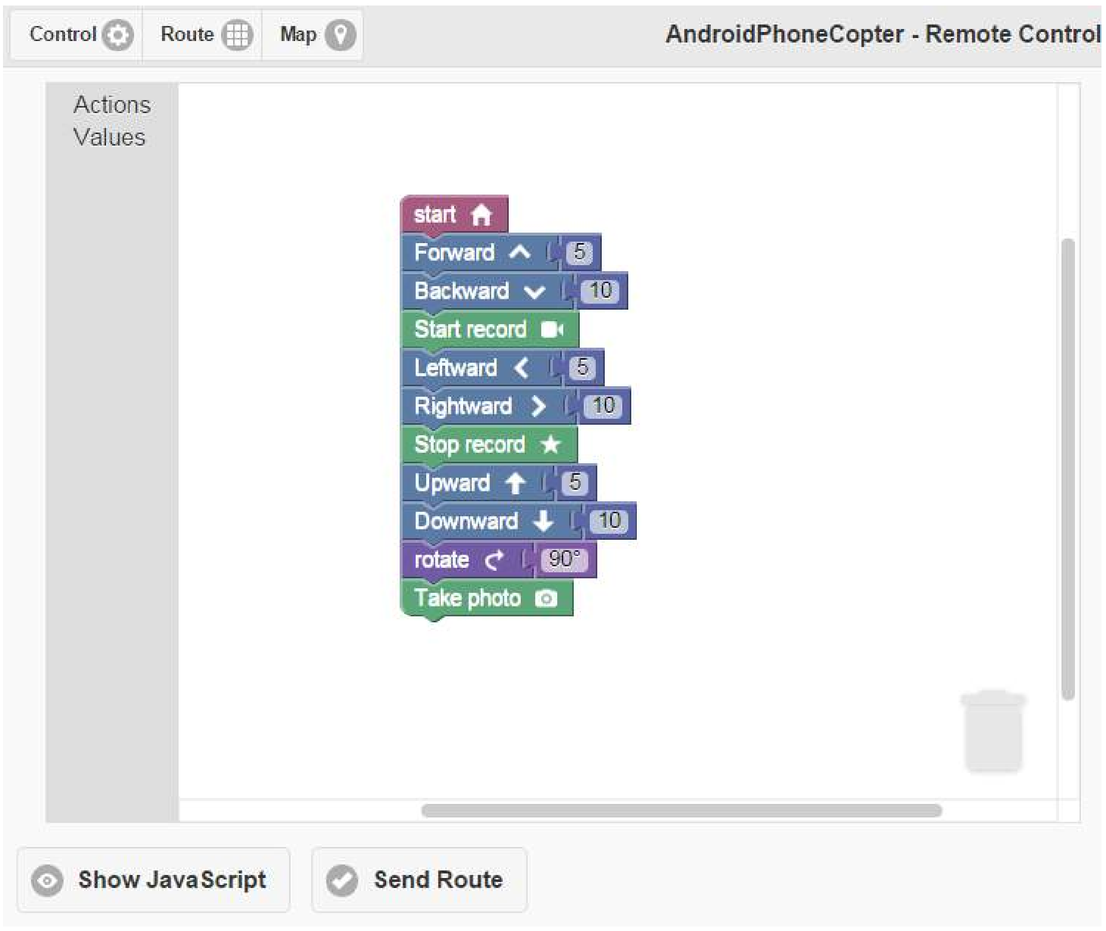
Functions

6.2.3 Route Page

The route page of the remote control panel provides different operation, includes send route and show javascript.

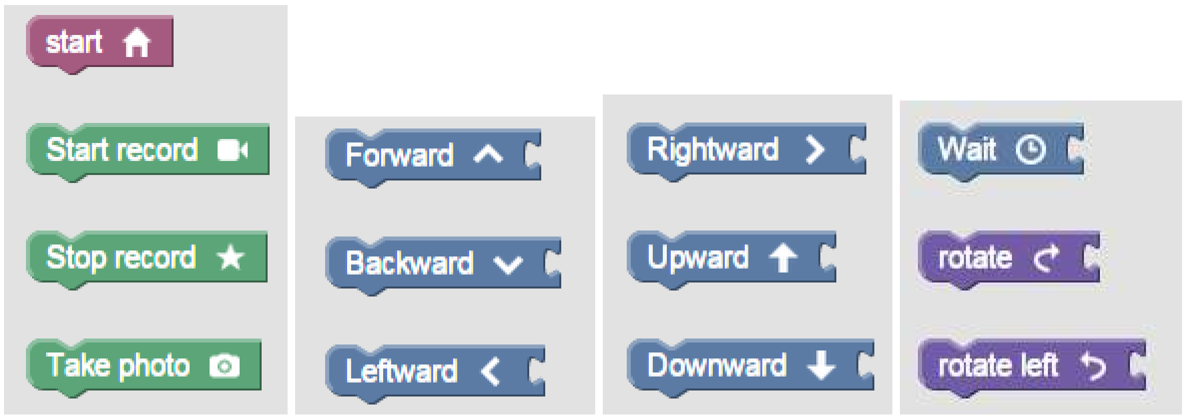
The route page of the remote control panel provides basic operation of the AndroidCopter. This page allows user to plan their route using puzzle-like control. The basic operation includes forward, backward, leftward, rightward, upward, downward and rotate. There are some advanced operation includes start recording video, stop recording video and take photo.

For example, the following page showing a route:
(1) Forward 5 units, (2) Backward 10 units, (3) Start record,
(4) Leftward 5 units, (5) Rightward 10 units, (6) Stop record,
(7) Upward 5 units, (8) Downward 10 units, (9) Rotate 90 degree, (10) Take photo

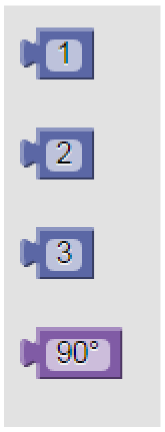


These operations are represented by puzzles in the Actions toolbox which can be dragged easily to the panel. We can append a value to the basic operation, otherwise, 0 will be appended.

Action puzzle:



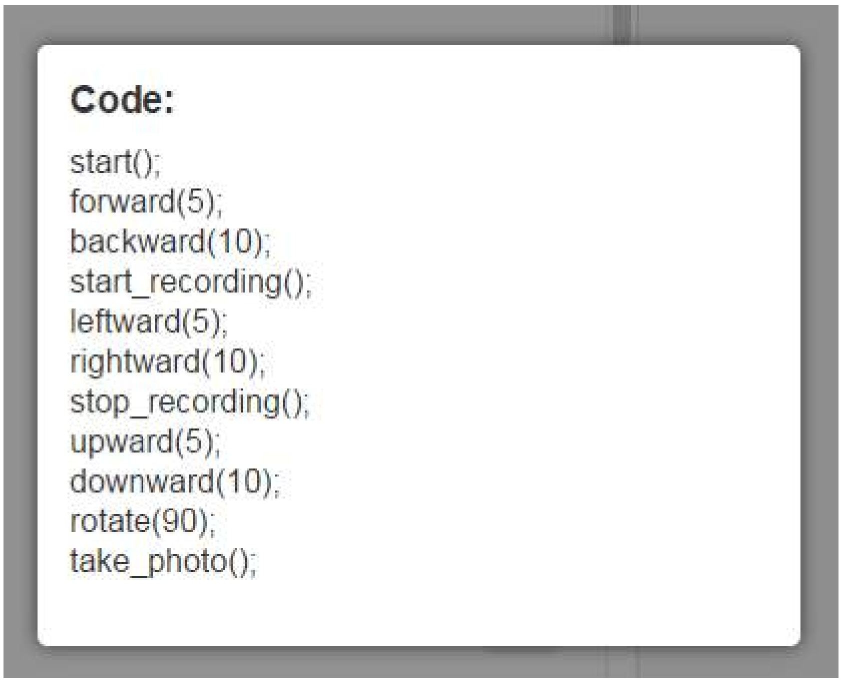
Value puzzle:



Apart from the default value of the value puzzle, the value can be changed easily by user as well.

Once we have confirmed our route, we can send route to the server. More, you can see the pseudo code generated from the route using “show javascript”, it will show a javascript pseudo code of your route.

This is an example of code generation.



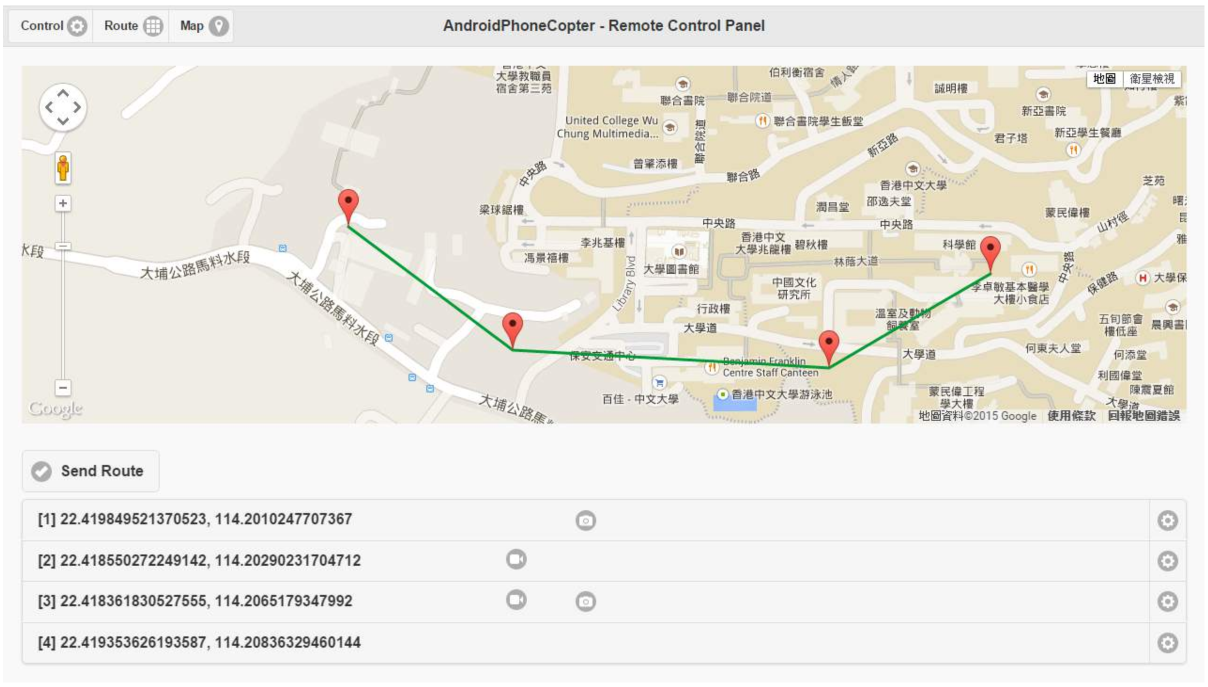
Code:

```
start();
forward(5);
backward(10);
start_recording();
leftward(5);
rightward(10);
stop_recording();
upward(5);
downward(10);
rotate(90);
take_photo();
```

6.2.4 Map Page

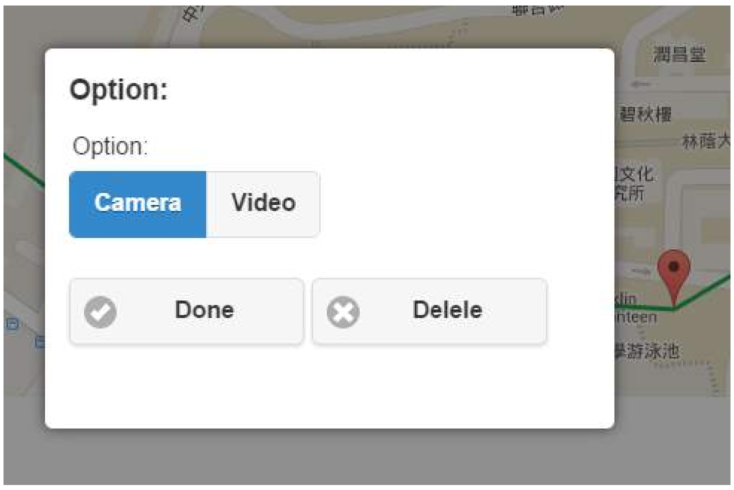
The map page of the remote control panel allows user to plan their route using google map. The page provides functionalities includes add marker, delete marker, reorder markers, send route and marker preferences.

By clicking on the map, it will add a marker on the mouse-clicked location. The map marker list shows a list of markers added.



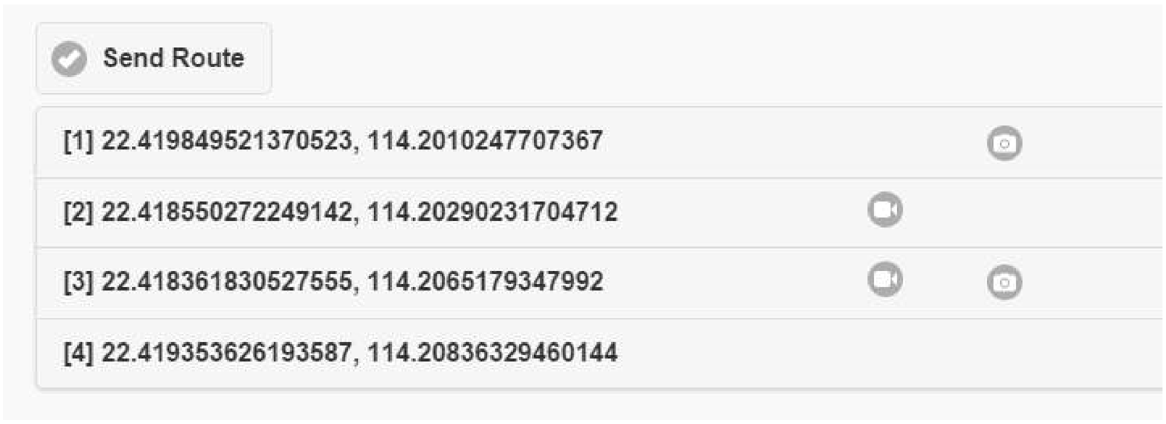
[1] 22.419849521370523, 114.2010247707367
[2] 22.418550272249142, 114.20290231704712
[3] 22.418361830527555, 114.2065179347992
[4] 22.419353626193587, 114.20836329460144

To remove a marker, user need to clicked on the marker preferences. He can thus deleted the marker by clicking “Delete”.



The marker preferences also provide options like take photo and record video at a particular point.

To reorder the markers, user can simply drag and drop the items in map marker list. The list also indicates the options selected in a particular point. For example, Marker 1 is enabled for photo taking, Marker 2 is enabled for video recording. Marker 3 is enabled for photo taking and video recording.



[1] 22.419849521370523, 114.2010247707367

[2] 22.418550272249142, 114.20290231704712

[3] 22.418361830527555, 114.2065179347992

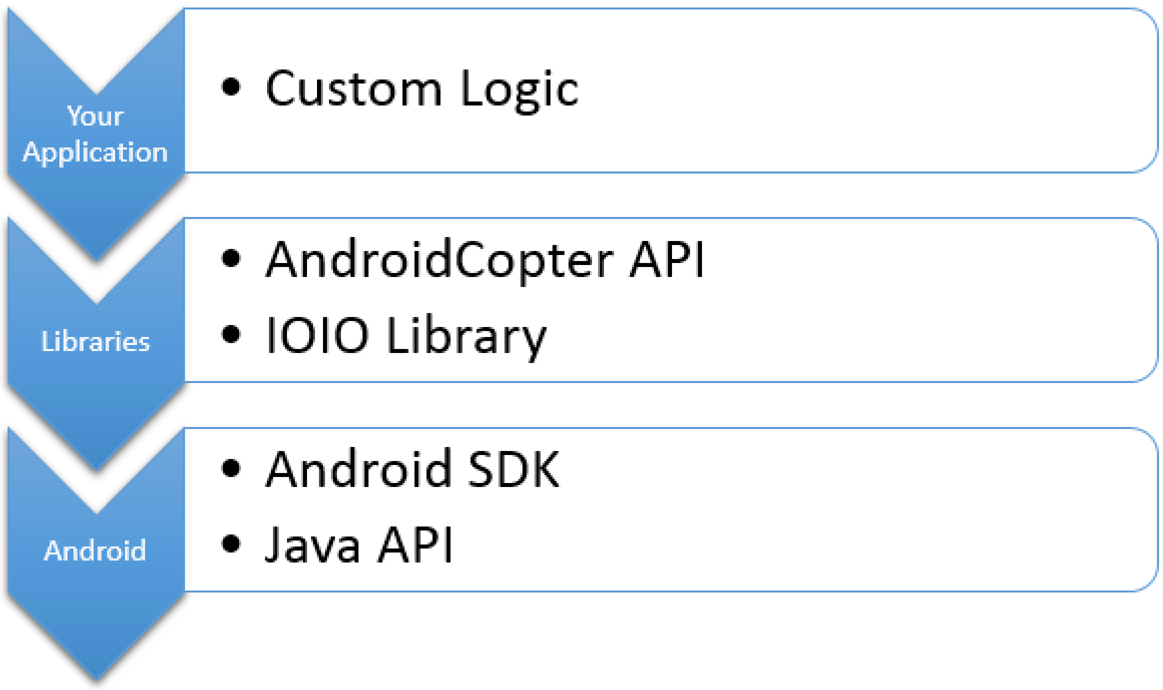
[4] 22.419353626193587, 114.20836329460144

Once the route is confirmed, user can send the route to the server.

7 AndroidCopter API

7.1 Overview of the API

AndroidCopter API is a Android library written in JAVA language. All basic flight control logic is encapsulated into TWO objects mainly: **IOIOFlightControl** object and **AndroidCopter** object. For next term, we will implement and extend more brilliant functions based on the library. Ultimately, other developers can use our AndroidCopter API to implement their quadcopter applications. It is much easier for them.



7.2 Strength of the API

Nowadays, Android system have not been widely used on Quadcopter. Most existing open-source flight controllers system are running on slow CPU. Most of them are written in C/C++ or Assembly, for example, KK Flight Controller is written in Assembly language and MultiWiiController is written in C language. It is very difficult for developers to extend the functions with these languages.

Easy to use

Undoubtedly, JAVA language is much easier than other lower level languages. The advantages of JAVA including object-oriented programming and automatic garbage collection which make hardware programming becomes easier. With few lines of code, the developer is able to make some specific control to the quadcopter.

For example, if a developer wants the quadcopter to go forward, the developer just need to call ***copter.forward()***.

7.3 Example Usage

The AndroidCopter API is very handy to use, the following example shows how to control the AndroidCopter to fly up to 100 cm, take a photo and go back to the ground.

The codes below can be put into the onCreate() of a main activity of an Android app.

```
flightControl = new IOIOFlightControl(this);
copter = AndroidCopter.getInstance(this, flightControl);

copter.setLooper(new AndroidCopterLooper () {
        public void loop() {
                if (copter.getHeight() >= 100) {
                        copter.takePhoto();
                        copter.autoLanding();
                        copter.endLooper();
                } else {
                        copter.setHeight(100);
                }
        }

});
```

7.4 Classes and Methods

The full description of the class and the methods of the API.

Class AndroidCopter

Constructor
private AndroidCopter(Context context, IOIOFlightControl flightControl) Construct a AndroidCoper object with a IOIOFlightControl object. The constructor is a private constructor. The instance should be created from the static method of the class AndroidCopter.getInstance().

Modifier and Type	**Method and Description**
static AndroidCopter	getInstance(Context context, IOIOFlightControl control)

	Get an instance of AndroidCopter. It is a singleton object. The static method ensures there is only one AndroidCopter object in the context.
void	throttle(int val) The setter method of throttle. The value ranges from 1000 to 2000.
void	rudder(int val) The setter method of rudder. The value ranges from 1000 to 2000.
void	aileron(int val) The setter method of aileron. The value ranges from 1000 to 2000.
void	elevator(int val) The setter method of elevator. The value ranges from 1000 to 2000.
int	throttle() The getter method of throttle. It returns current throttle value.
int	rudder() The getter method of rudder. It returns current rudder value.

int	aileron() The getter method of aileron. It returns current aileron value.
int	elevator() The getter method of elevator. It returns current elevator value.
void	arm() ‘Arm’ means the motors are able to be controlled by input value.
void	disarm() Lock all motors immediately.
boolean	isArmed() Check whether if the AndroidCopter is armed.
void	setWebSocketJoystick(WebSocketJoystick joystick) The AndroidCopter can be controlled through network only if it has related to an instance of WebSocketJoystick. The alternative name is Remote Control Panel.
WebSocketJoystick	getJoystick() Return the WebSocketJoystick object.
void	send(String key, String val)

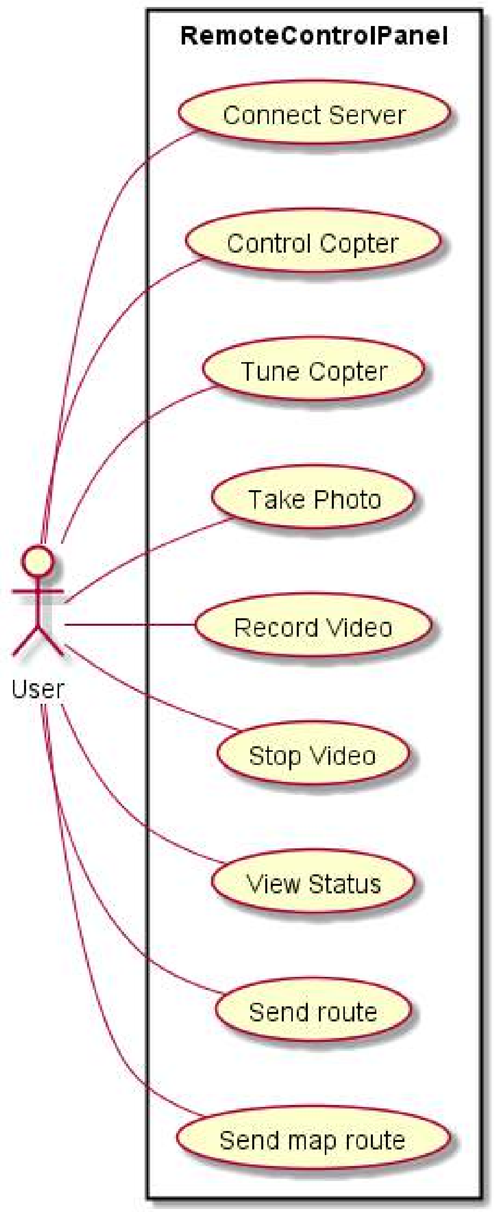
	Send a command or message with specific value to the Remote Control Panel.
void	send(JSONObject json) Send commands or messages in JSON format.
void	reloadSettings() If some values are changed, this method should be invoked to apply changes.
void	enableAutoLeveling(boolean val) This method can be invoked to enable or disable Auto Leveling function of the AndroidCopter. An accelerometer is required in order to use this feature.
void	autoLanding() The AndroidCopter will go back to the ground after this method is called. Once this method is invoked, other commands cannot interrupt it until it lands on the ground. A sonic sensor and a barometer are required.
void	setHeight(float height) Go to desired height. A sonic sensor and a barometer are required. The input value unit is centimeter.
float	getHeight() Get the current height.

void	up() An alternative simpler method of throttle(). A sonic sensor and a barometer are required.
void	down() An alternative simpler method of throttle(). A sonic sensor and a barometer are required.
void	left() An alternative simpler method of aileron().
void	right() An alternative simpler method of aileron().
void	forward() An alternative simpler method of elevator().
void	backward() An alternative simpler method of elevator().
void	rotate(float degree) An alternative simpler method of rudder(). The value ranges from 0 to 359. Magnetic sensor is required.
void	takePhoto() Take a photo.

void	recordVideo() Start to record video.
void	stopVideo() Stop to record video.
void	setLooper(AndroidCopterLooper looper) Custom actions should be setted in the looper.
void	endLooper() Once this method called, the looper is no longer being executed.

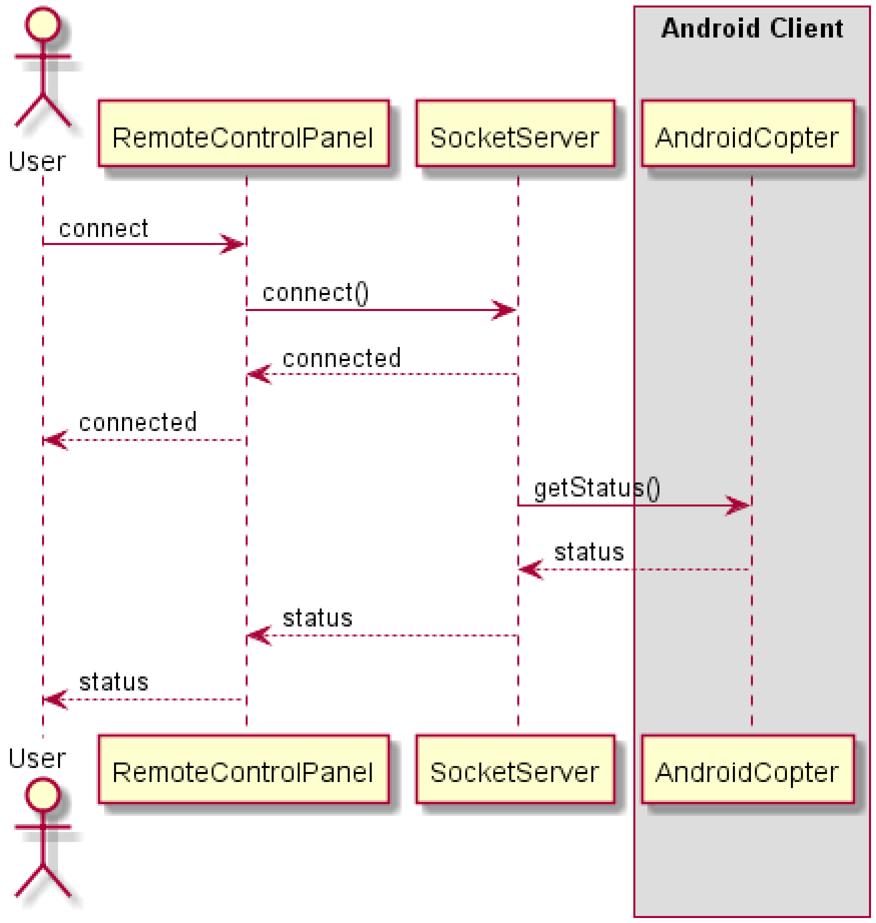
8 Design Diagram

8.1 Use Case Diagram



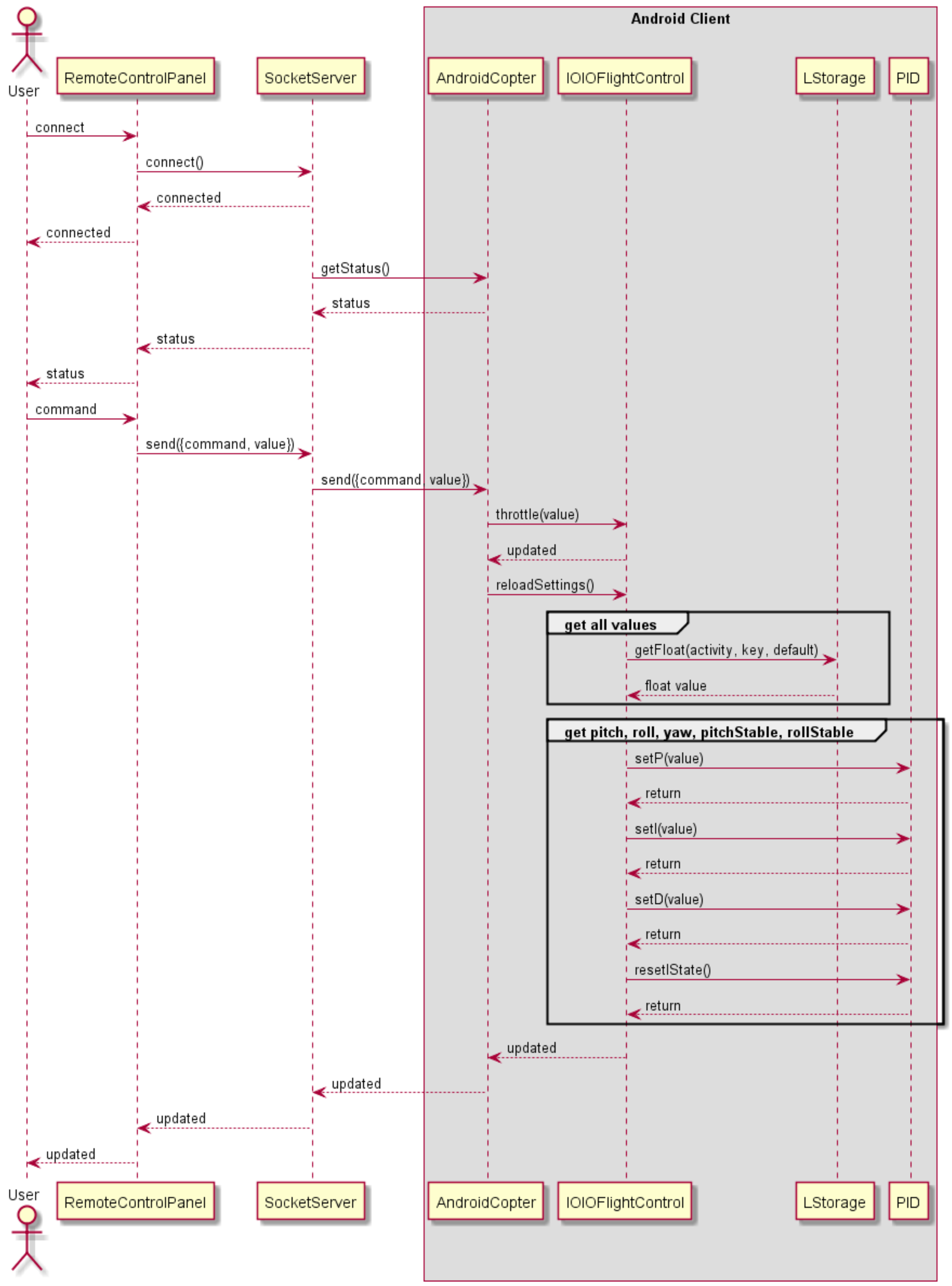
8.2 Sequence Diagram

8.2.1 Connect Server



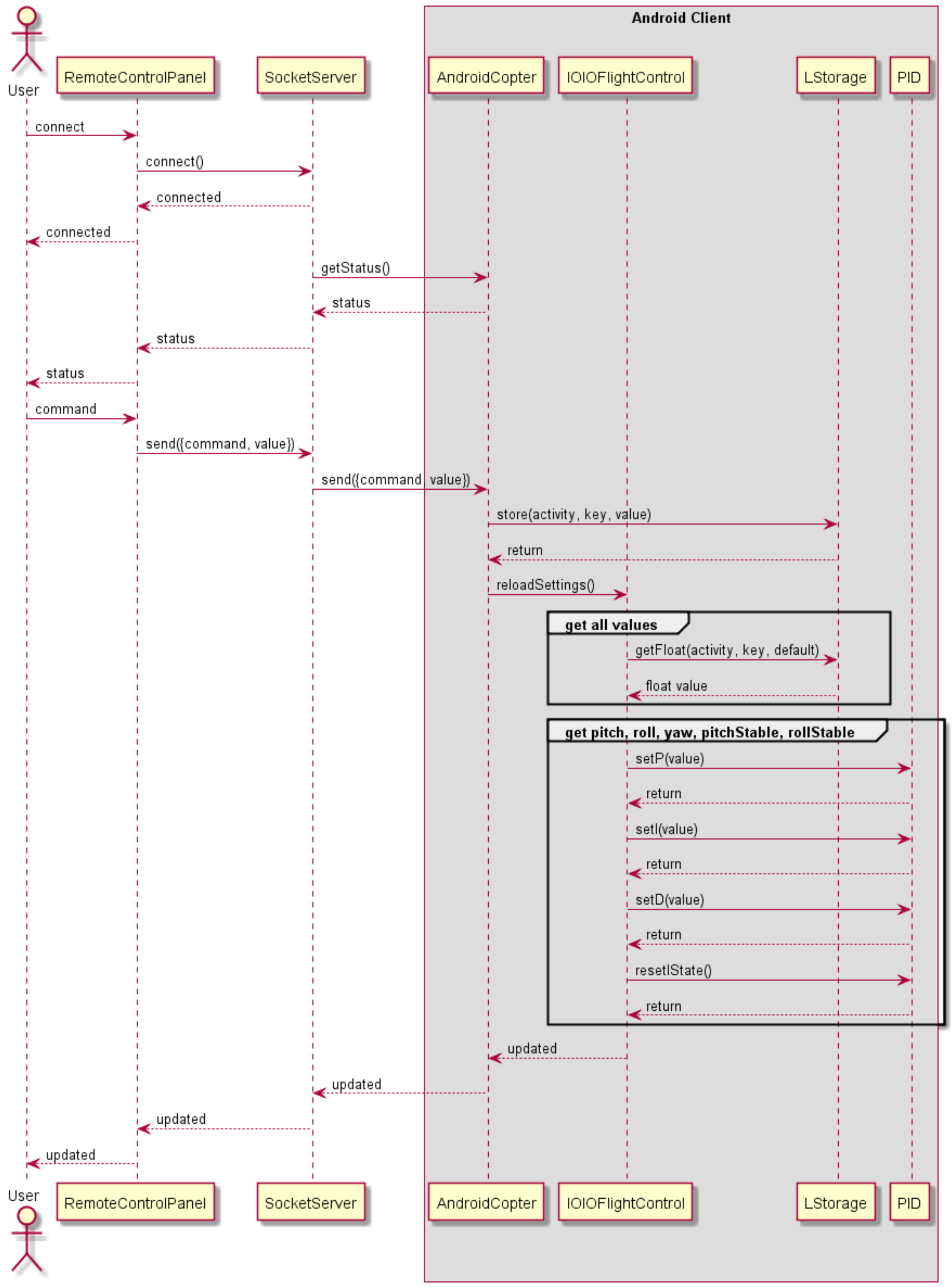
8.2.2 Control Copter

For variables including throttle, rudder, aileron and elevator, they are controlled via the Control Copter function.

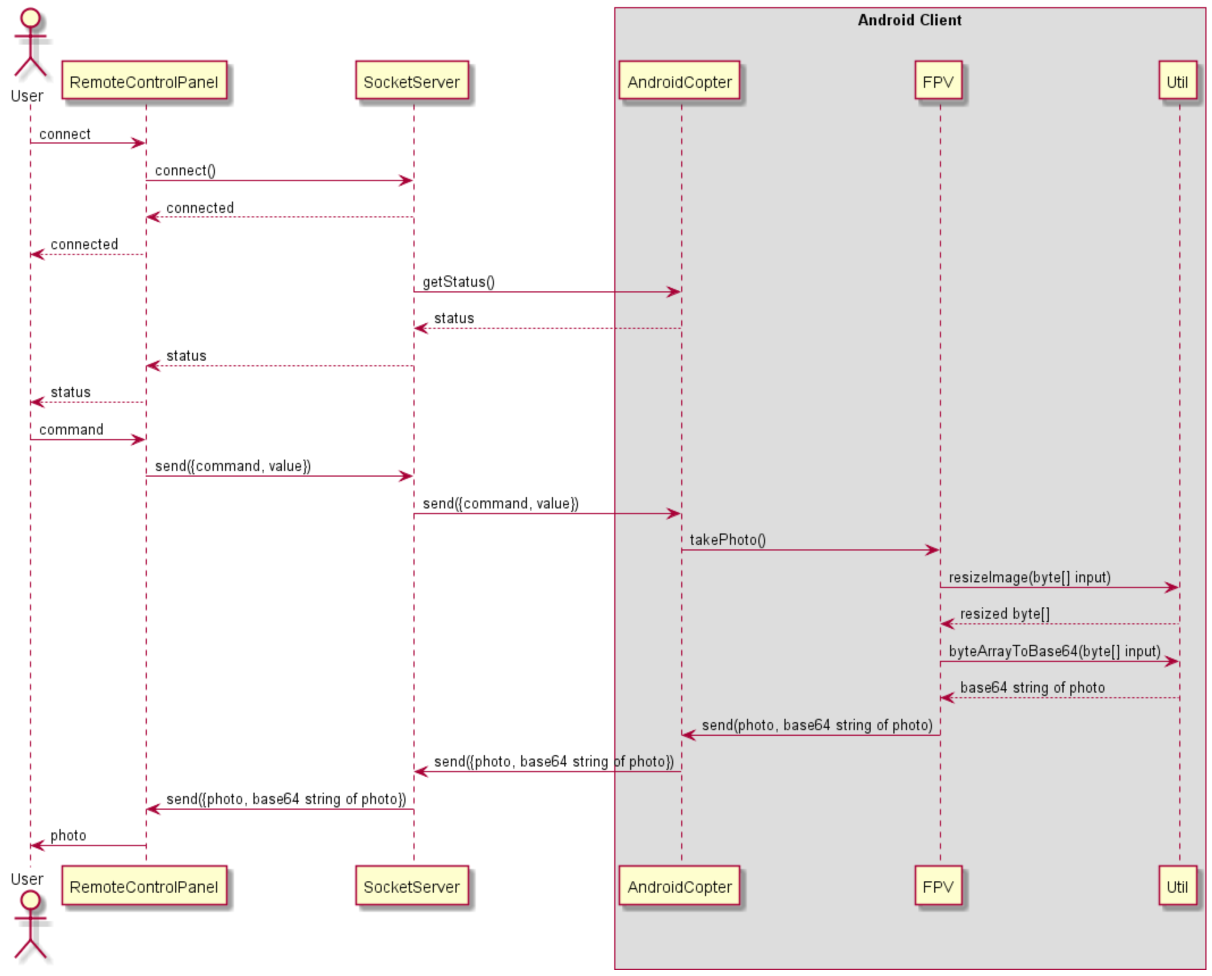


8.2.3 Tune Copter

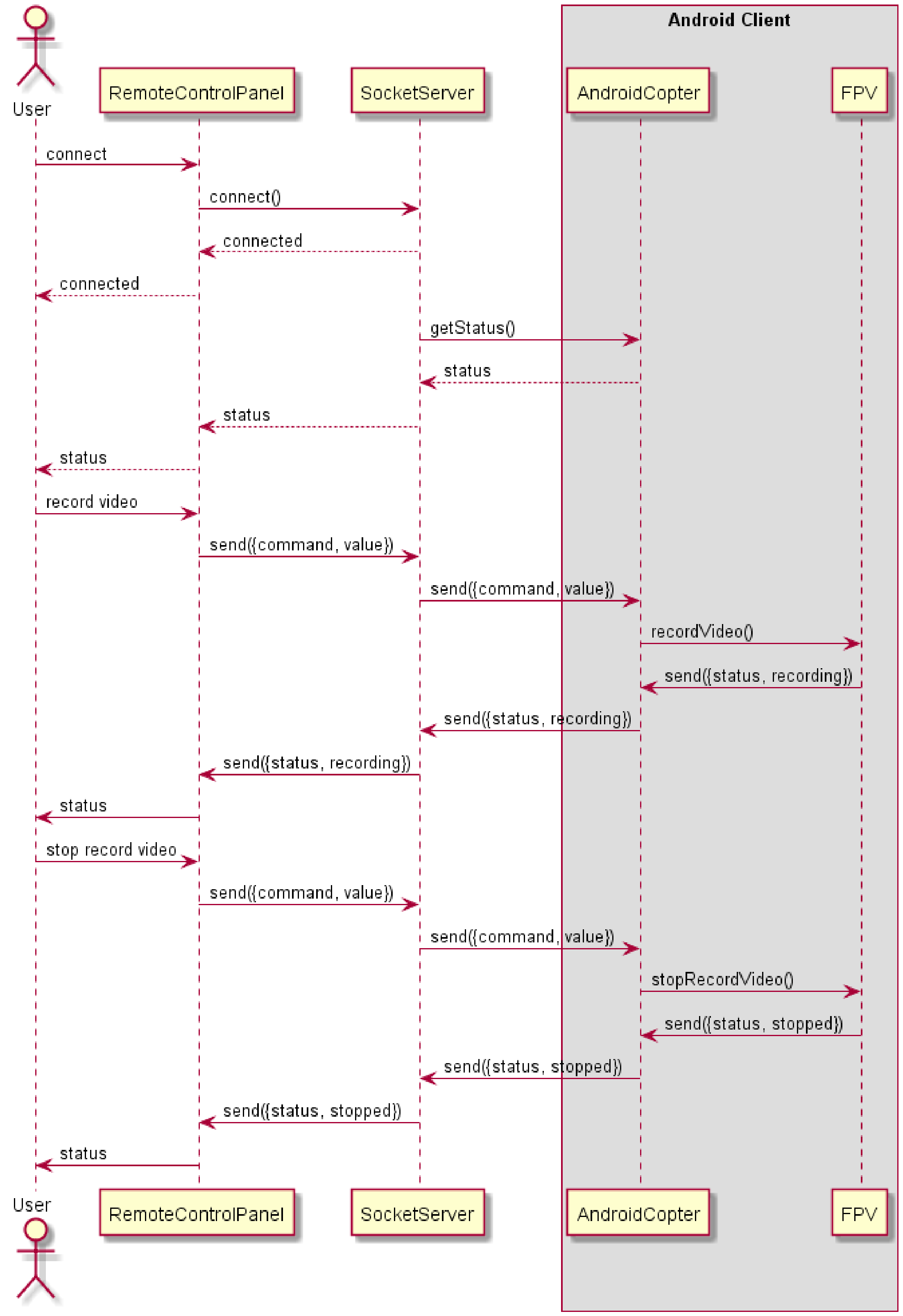
For variables including PitchP, PitchI, PitchD, rollP, rollI, rollD, yawP, yawI and yawD, they are tuned via the Tune Copter function.



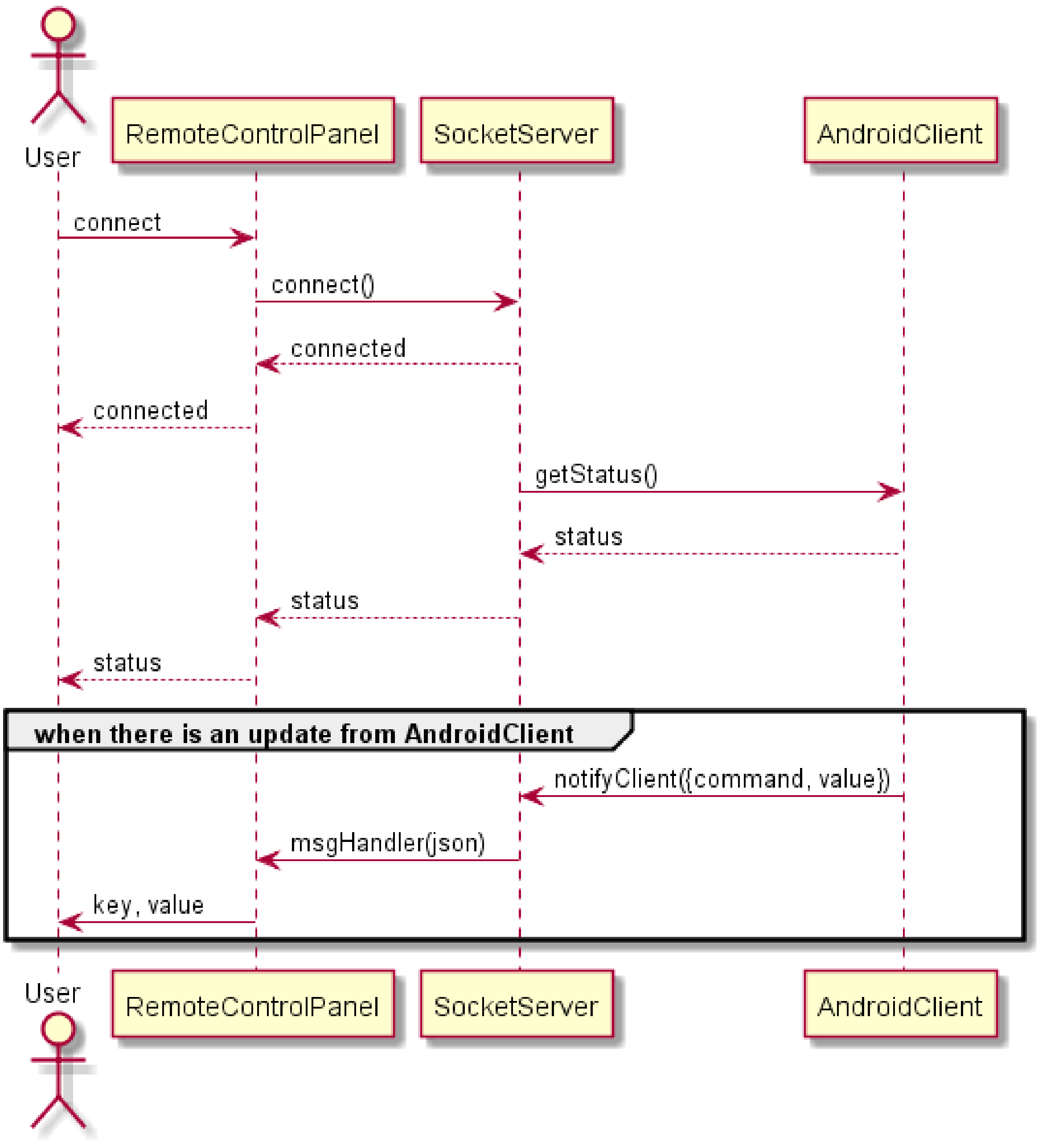
8.2.4 Take Photo



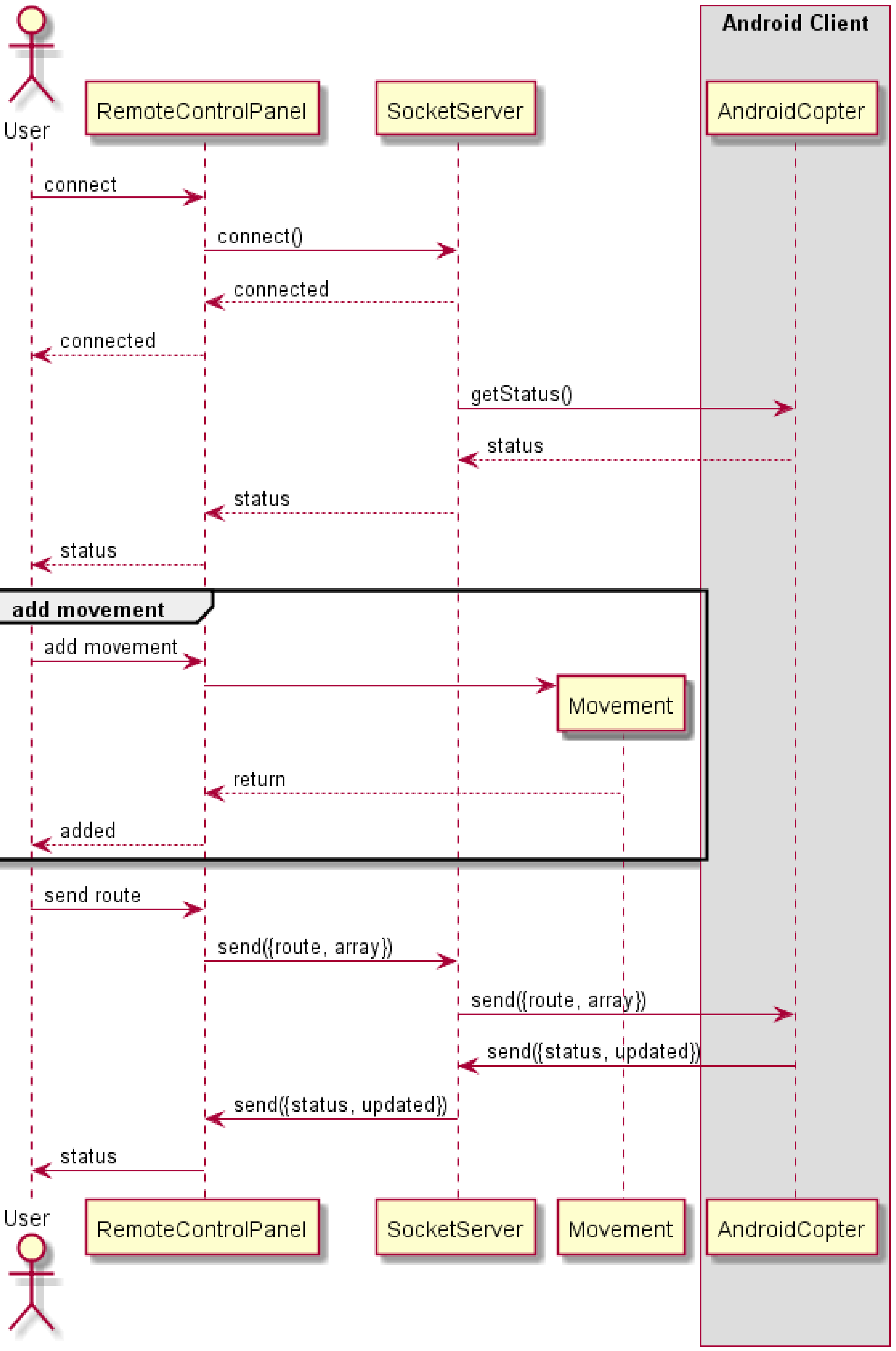
8.2.5 Record Video & Stop Video



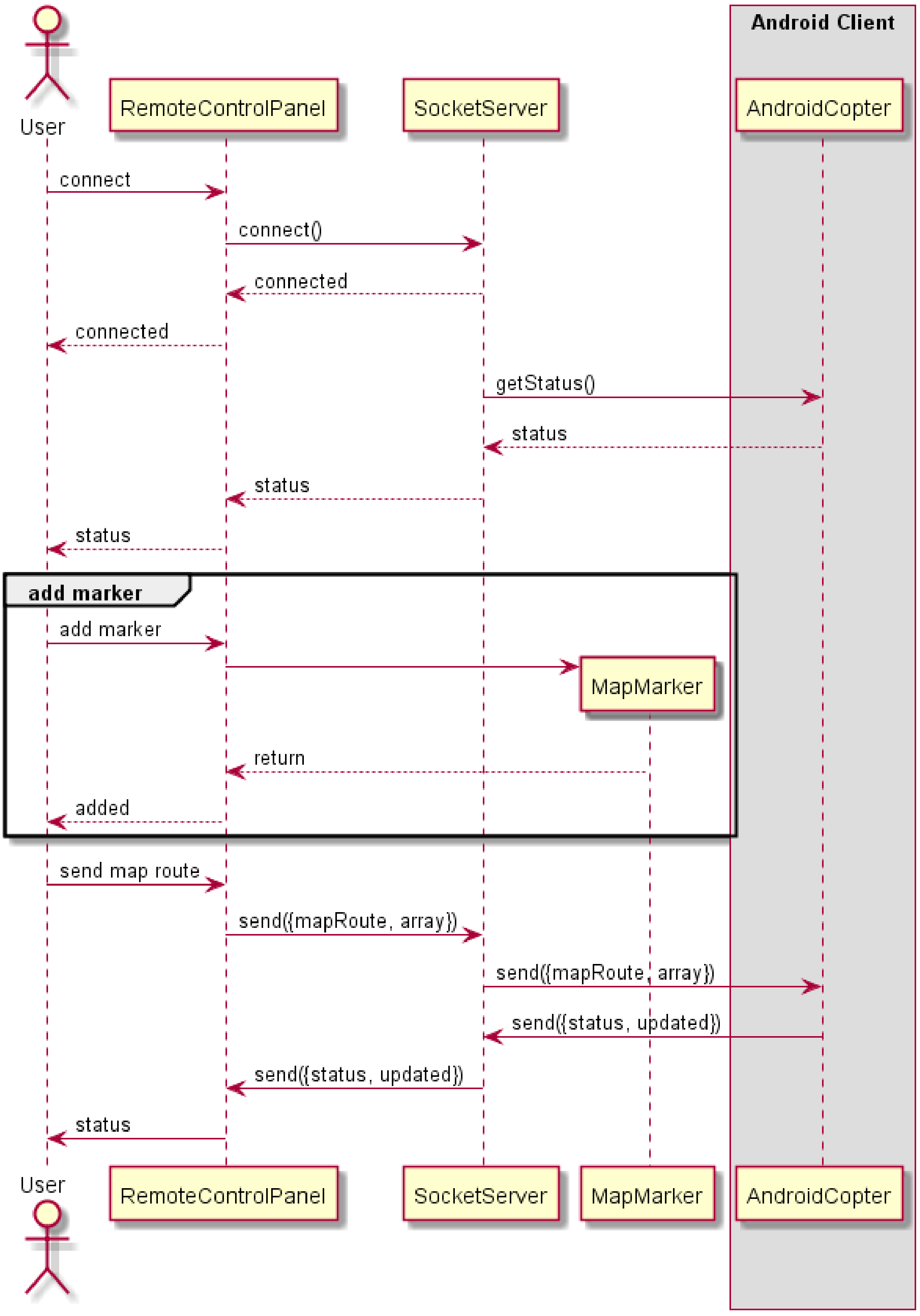
8.2.6 View Status



8.2.7 Send Route



8.2.8 Send Map Route



8.3 JSON Data Format

In order to make the AndroidCopter and the RemoteControlPanel to communicate, a standard command key is needed to make them understand with each other. In the example below, with the key “arm” and value “true”.

Example:

```
{
        “key” : “arm”,
        “value” : “true”
}
```

JSON Data Format List

Key	arm
Description	set the arm level of copter
Type	Boolean
Default Value	False

Key	autoLeveling
Description	set auto leveling or not
Type	Boolean
Default Value	True

Key	easyThrottle
Description	set easy throttle or not
Type	Boolean
Default Value	False

Key	throttle
Description	set the throttle level of copter
Type	Integer
Default Value	1000
Value	1000 to 2000

Key	rudder
Description	set the rudder level of copter
Type	Integer
Default Value	1500
Value	1000 to 2000

Key	elevator
Description	set the elevator level of copter
Type	Integer
Default Value	1500
Value	1000 to 2000

Key	aileron
Description	set the aileron level of copter
Type	Integer
Default Value	1500
Value	1000 to 2000

Key	takePhoto
Description	take a photo
Type	String
Value	takePhoto

Key	recordVideo
Description	start video recording
Type	String
Value	recordVideo

Key	stopRecordVideo
Description	stop video recording
Type	String
Value	stopRecordVideo

Key	landing
Description	land the copter
Type	String
Value	landing

Key	route
Description	Copter moves according to the route
Type	Array
Value	Array of Movement

Key	forward
Description	Copter moves forward
Type	Movement
Default Value	0
Value	0 to 1000

Key	backward
Description	Copter moves backward
Type	Movement
Default Value	0
Value	0 to 1000

Key	leftward
Description	Copter moves leftward
Type	Movement
Default Value	0
Value	0 to 1000

Key	rightward
Description	Copter moves rightward
Type	Movement
Default Value	0
Value	0 to 1000

Key	upward
Description	Copter moves upward
Type	Movement
Default Value	0
Value	0 to 1000

Key	downward
Description	Copter moves downward
Type	Movement
Default Value	0
Value	0 to 1000

Key	rotate
Description	Copter rotates
Type	Movement
Default Value	0
Value	0 to 360

Key	rotate_left
Description	Copter rotates left
Type	Movement
Default Value	0
Value	0 to 360

Key	wait
Description	Copter wait for some second
Type	Movement
Default Value	0
Value	0 to 10000

Key	mapRoute
Description	Copter moves according to map route
Type	Array
Value	Array of mapMarker

Key	mapMarker
Description	Map marker with latitude and longitude
Type	Object
Value	latitude, longitude, photo, video

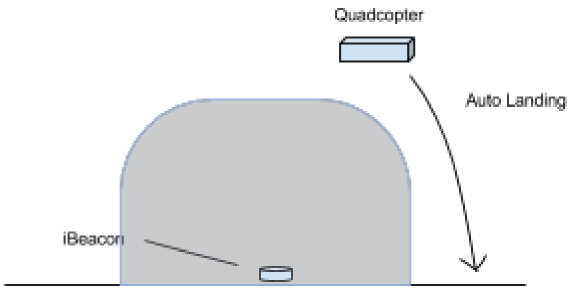
9 Safety Measurement

9.1 Arming and Disarming Features

If the quadcopter can start a flight by simply power it up, it may post safety risks on the users or nearby people, especially when the controlling person is not familiar with the control of quadcopter. One possible way to solve this is that using the arming and disarming function. When the controlling person have confirmed that no people is nearby the quadcopter, he can thus arming the motors. After the flight have ended, he can thus disarming the motors to ensure it is safe to get close to the quadcopter.

9.2 Ranging Limitation using iBeacon

Even in real product, there were some cases that the copters suddenly disconnected. These copters flew away and never came back. One of popular case was “Flutterbye Flying Fairy Doll”. After the girl have turned on the power, the toy never came back.

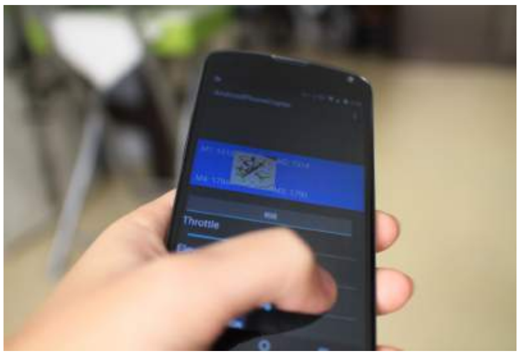


Using iBeacon can avoid this situation. iBeacon is mainly used for an indoor positioning system which is develop by Apple Company. In this project, iBeacon is used for ranging and only one piece of iBeacon is needed. The quadcopter can get the approximate distance from the iBeacon. So if the quadcopter is out of certain distance or the iBeacon is turned off, the quadcopter will trigger the auto landing function automatically. As a result, the quadcopter will not fly away easily in the open area.

10 Testing and Experiment Data

Testing is very important in any project, especially in this project. Any small mistakes can crash the quadcopter. So the Android application must be tested intensively and completely.

10.1 Testing in Term One



After the stabilization algorithm was finished, to ensure the correctness of output values of motors, these values were printed on screen first instead of putting the phone on the quadcopter. Then we could shake or move the phone to see whether the output values are correct or not.

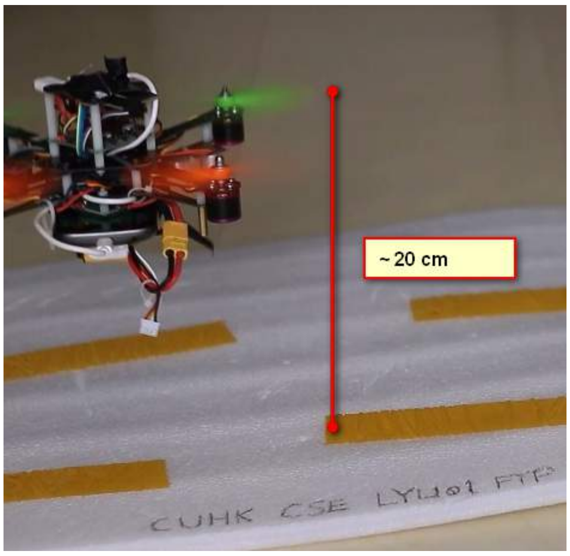
If the values are reasonable, then the Android phone can be put on the quadcopter. However, the propellers should not be installed on the quadcopter in this step. After that, we could arm the quadcopter and increase the throttle, so that we could see the motors were working properly.

Then the propellers can be installed on each motor. The quadcopter should be attached to a rope, it will not crash even if there is any problem. And the AndroidCopter must be connected to the Remote Control Panel, as we would not be able to touch the screen of the phone anymore.

For more testing videos, please go to the Youtube playlist:
https://www.youtube.com/playlist?list=PLjfSRxTTcLFm4EFfrrjjIjNMc18WZLQRd

10.2 Testing in Term Two

10.2.1 Stable Flying Height



Due to the optical flow sensor cannot focus automatically. The most stable flying height is around 20 cm. Too high or too low may result in instability or even out of control.

10.2.2 Best Ground Surface

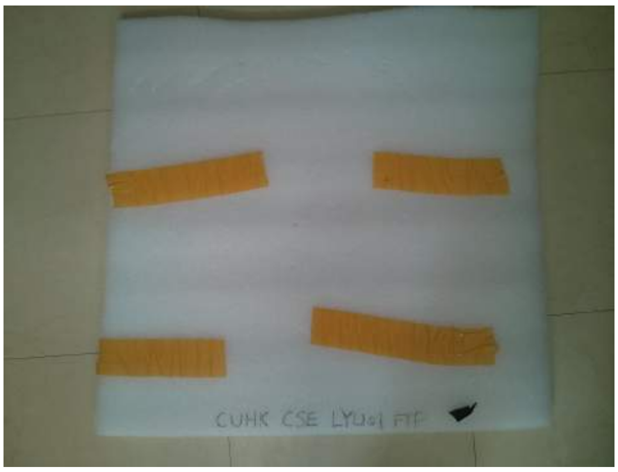
We have conducted testing in several ground surfaces.

Ground 1: normal ground

For example, this ground is a normal ground with some thin lines on it. Unfortunately, The optical flow sensor reports nothing even though the copter is moving. So our copter cannot fly on the ground like this.

Ground 2: normal outdoor ground

This a normal outdoor ground. Similar to the previous test, the result is very bad.



Ground 3: detailed surfaces with big objects

This is our custom-made platform with some markers and text on it. And we discovered that only detailed surfaces with big objects like this are suitable for our AndroidCopter to fly stably.

10.2.3 Brightness Testing

<u>Light source: daylight</u>

This frame was captured with daylight in an indoor environment at around 2 pm. The optical flow sensor was able to report the movement data.

<u>Light source: indoor light</u>

This frame was captured with indoor light in the evening at around 9 pm. And the optical flow sensor reported zero delta X and Y even if the copter is moving around.

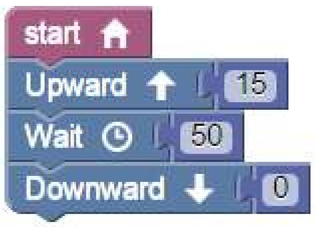
To conclude, if there is not enough light source, the copter could not fly stably even though the copter is on the best surface.

10.2.4 Path Tracking Testing

All movement actions are specified in centimeter.

Hover 50 seconds in the air

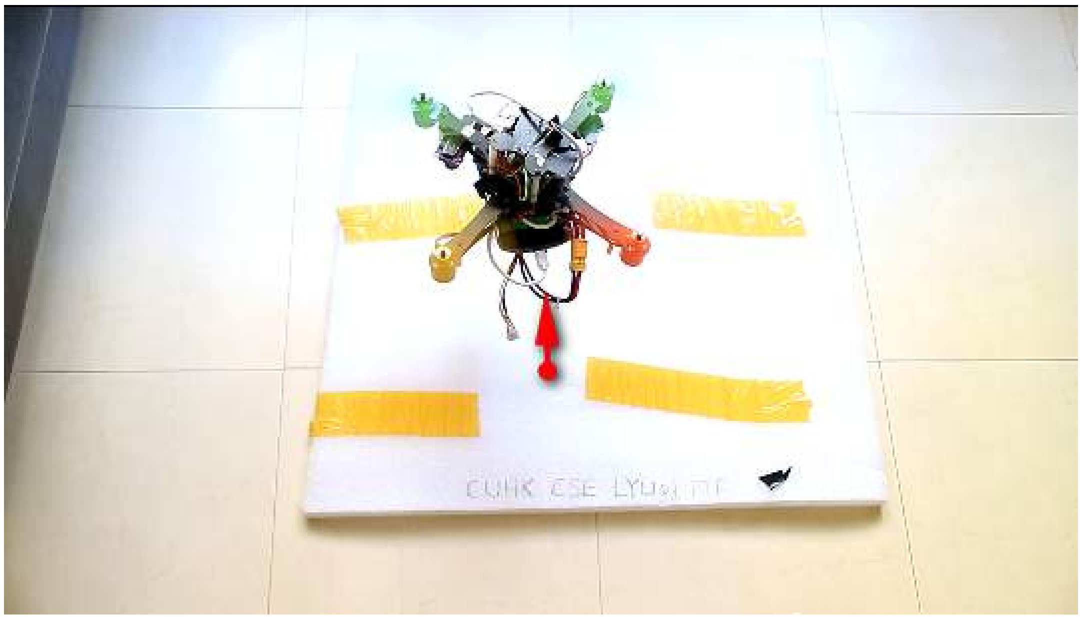
Command:



```
start
Upward 15
Wait 50
Downward 0
```

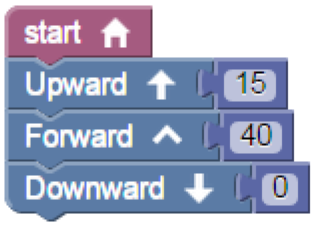
Result:

The AndroidCopter moves upward by 15 cm. The AndroidCopter is able to hover for 50 seconds in the air. It starts landing.



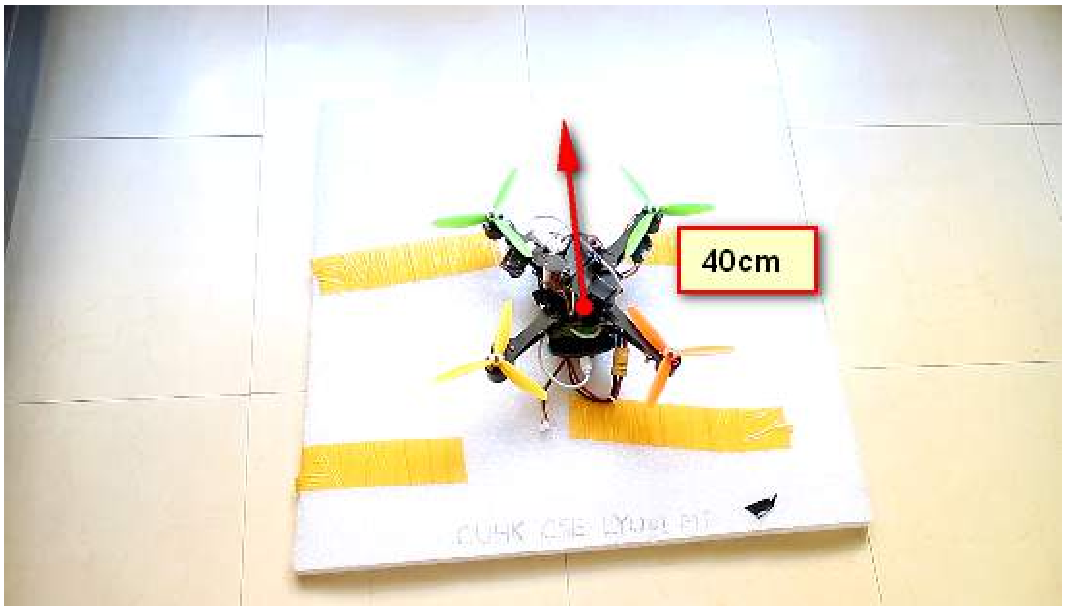
Forward

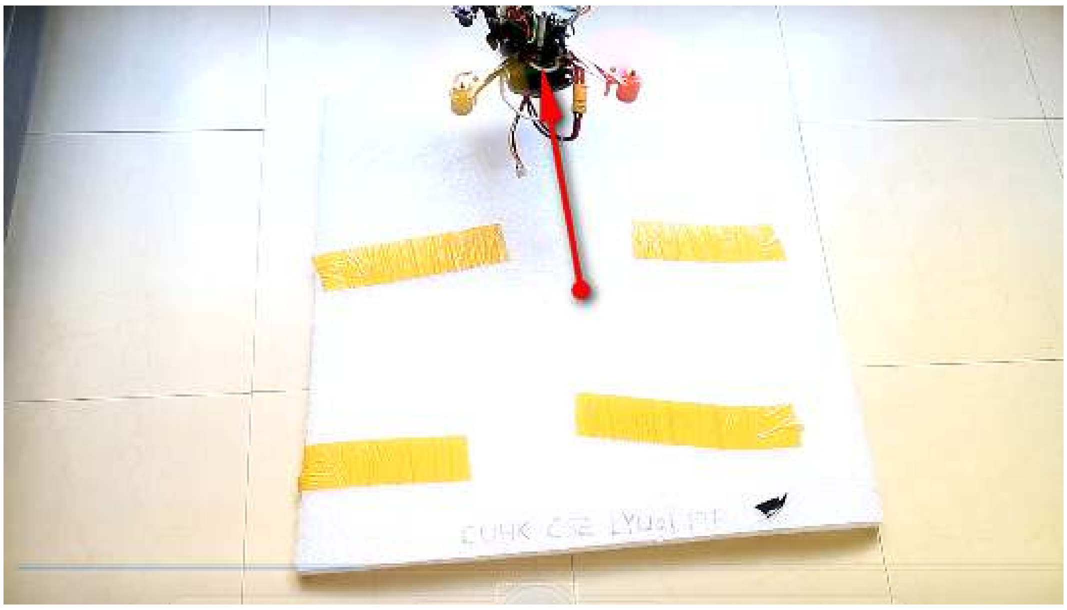
Command:



Result:

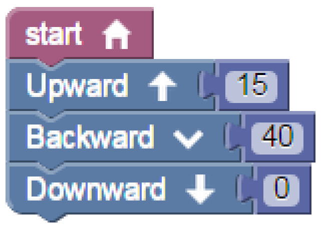
The AndroidCopter moves upward for 15 cm. Then, it moves forward by 40 cm. After that, it starts landing.





Backward

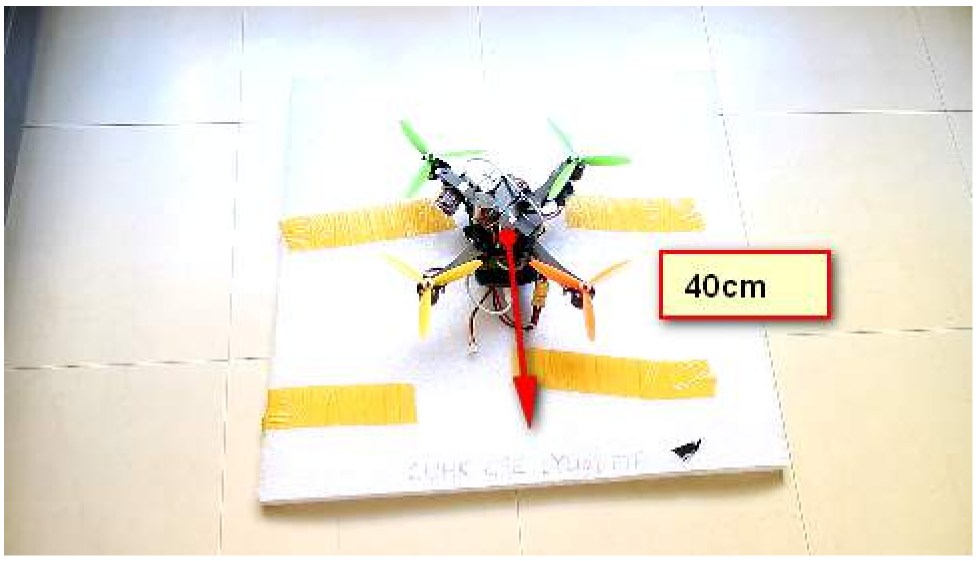
Command:



```
start
Upward 15
Backward 40
Downward 0
```

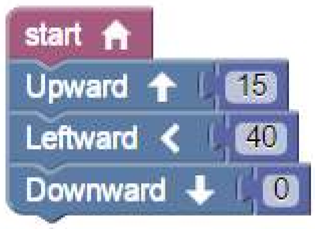
Result:

The AndroidCopter moves upward for 15 cm. Then, it moves backward by 40 cm. After that, it starts landing.



Left

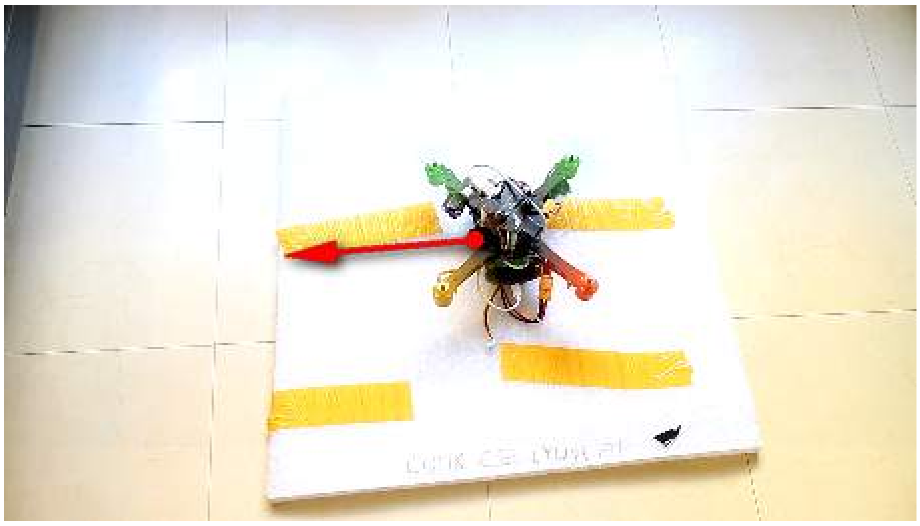
Command:

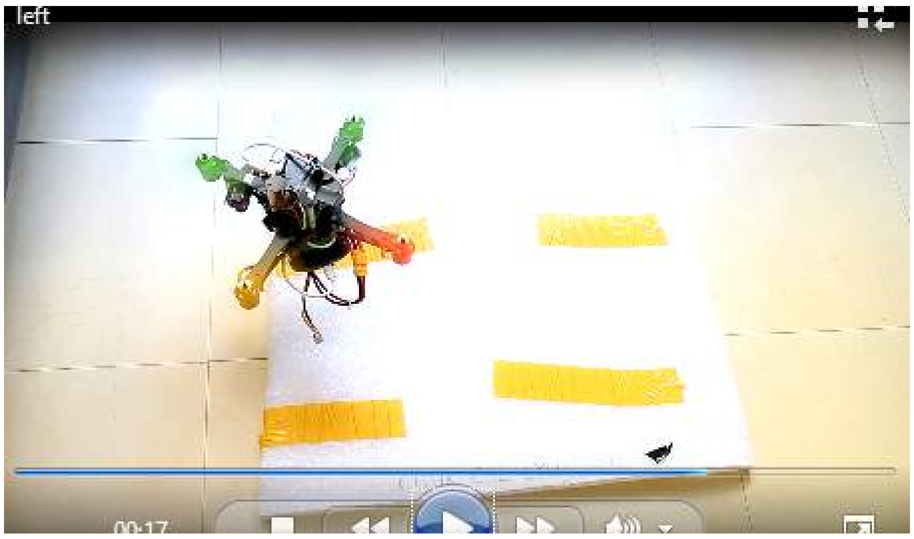


```
start
Upward 15
Leftward 40
Downward 0
```

Result:

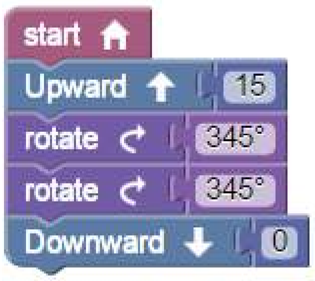
The AndroidCopter moves upward for 15 cm. Then, it moves leftward by 40 cm. After that, it starts landing.





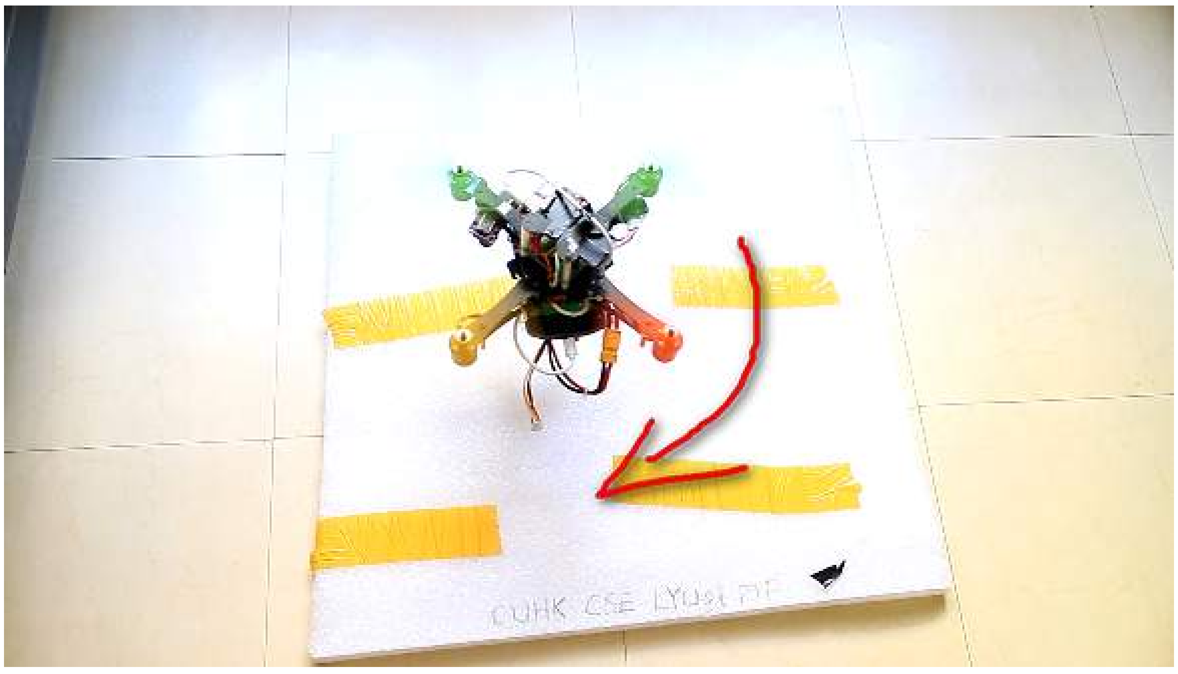
<u>Rotation</u>

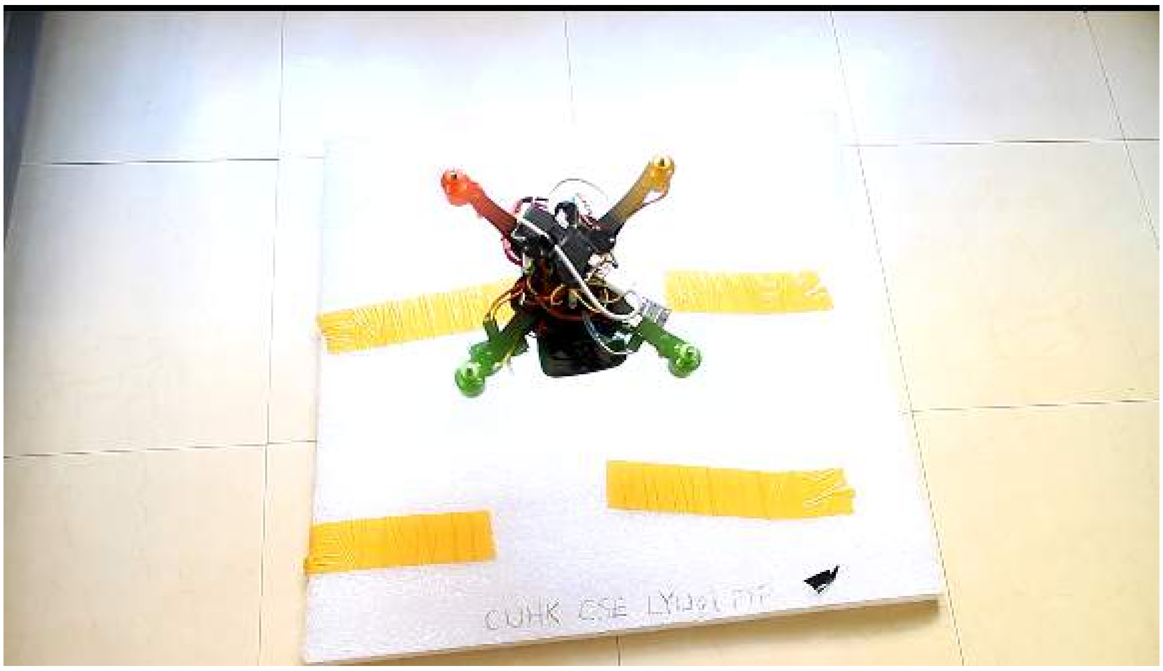
Command:



Result:

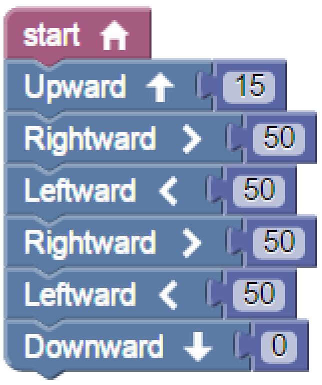
The AndroidCopter moves upward by 15 cm. Then, it rotates by 345 degree. After that, it rotates by 345 degree again. Next, it starts landing.




CUHK CSE

Right, left, right, left

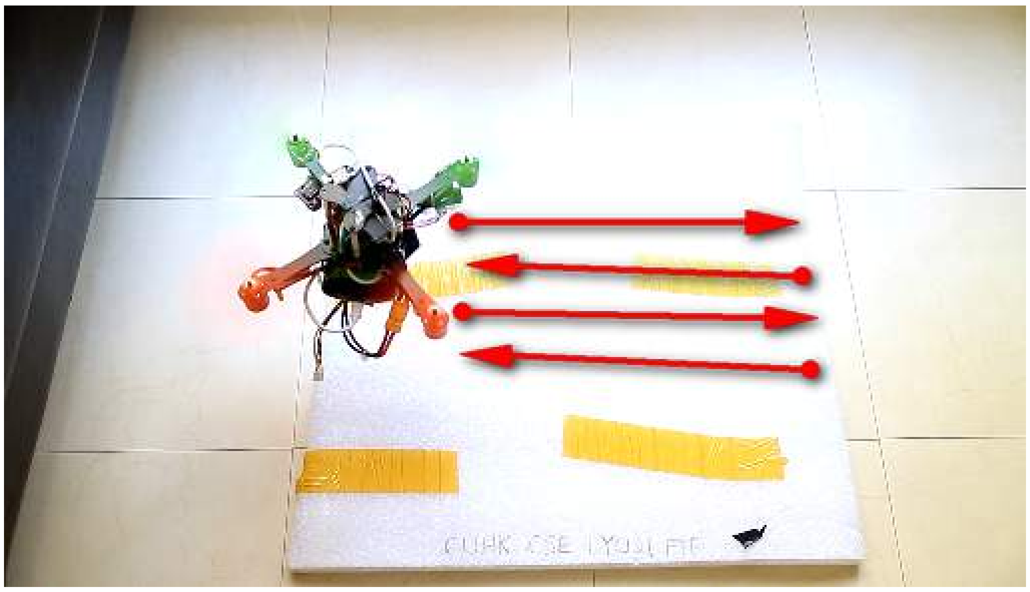
Command:



```
start
Upward 15
Rightward 50
Leftward 50
Rightward 50
Leftward 50
Downward 0
```

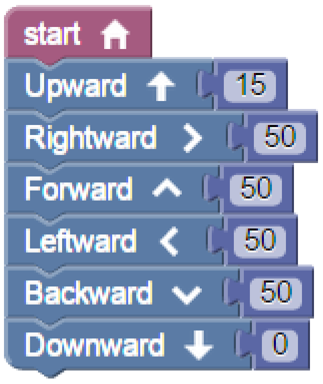
Result:

The AndroidCopter moves upward by 15 cm. Then, it moves rightward by 50cm. Next, it moves leftward by 50 cm. After that, it moves rightward by 50cm and moves leftward by 50 cm. Next, it starts landing.



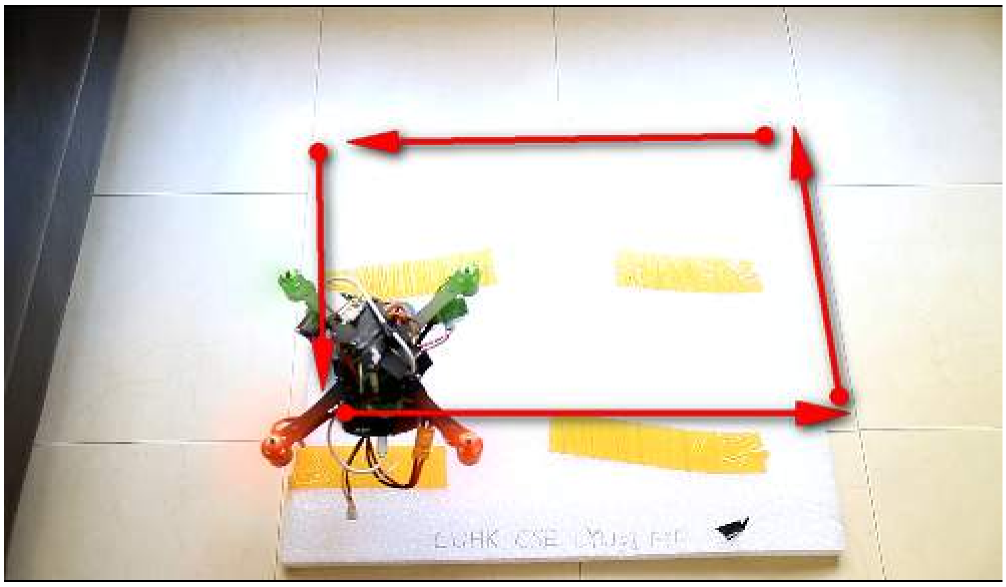
Square Pathing

Command:

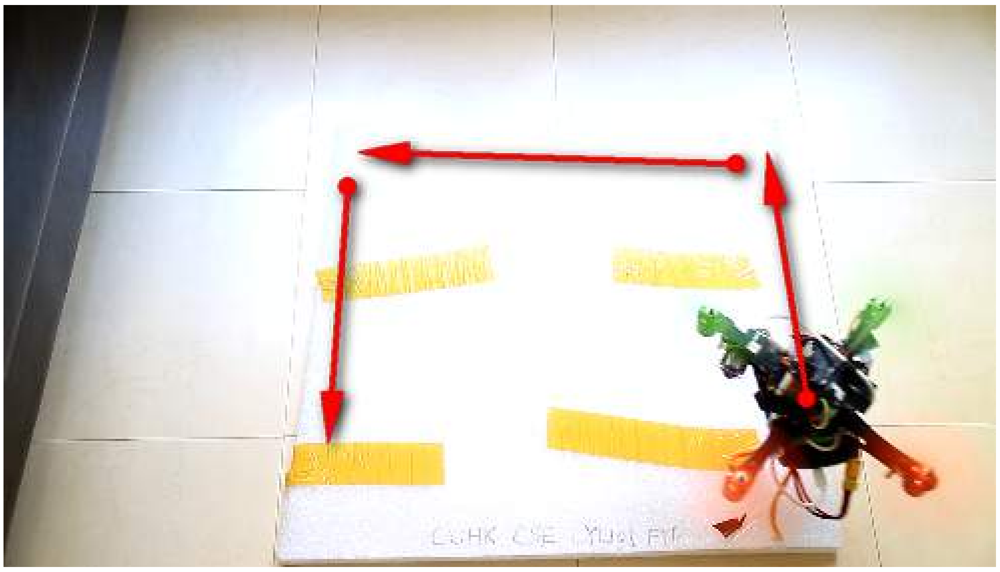

start
Upward 15
Rightward 50
Forward 50
Leftward 50
Backward 50
Downward 0

Result:

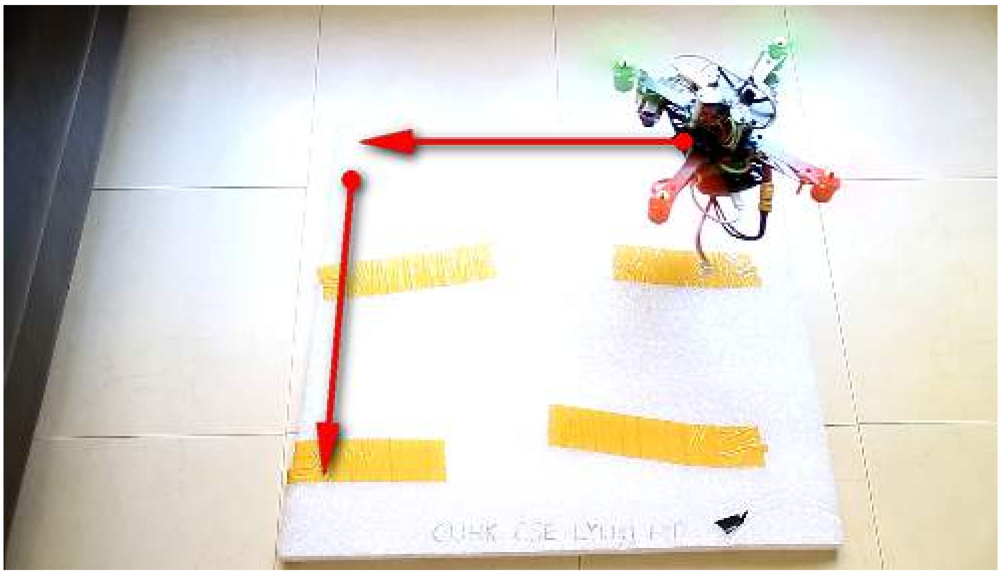
Firstly, the AndroidCopter moves upward by 15 cm.



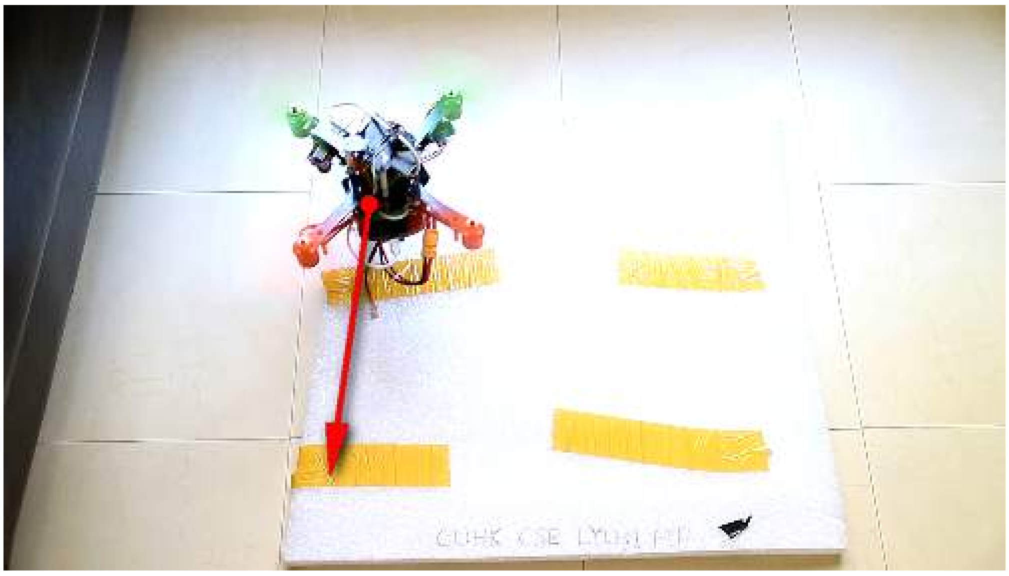
Secondly, the AndriodCopter moves rightward by 50 cm.



Thirdly, the AndriodCopter moves forward by 50 cm.



Next, the AndriodCopter moves leftward by 50 cm.



More, the AndriodCopter moves backward by 50 cm. After that, it starts landing.

11 Current Limitations of AndroidCopter

Hardware Dependencies

Currently, there are not many phones fulfill all hardware requirements. Only Google Nexus Serious have the barometer sensor. As many phone manufacturers have a high degree of customization on the Android system of their Android product, the IOIO board do not support many Android models currently.

Short Flying Time

Short flying time is sourced from the limited capacity of power source. Besides, because of the weight of Android phone, the flying time is shorten as well.

12 Difficulty of the project

Testing Environment

When the quadcopter is flying, it produces some huge noise continuously, some people may think it is noisy and reject such action. Next, for safety reason, some people may reject as well. More, flying of a quadcopter requires a large and open area for better navigation. It is not easy to find a place to test with these requirements.

Dangerous

Quadcopter may pose threats to the controlling person and nearby people. When the quadcopter is flying, the propellers rotate at a high speed which is able to injure someone's body. We need to ensure there is no human activities in its route.

Hardware Compatibility

As described previously, there are sensor, network and weight requirements for an Android phone that can be used in the project. We have spent some time to find a suitable phone for the project.

13 Usage

Emergency Management

Multicopter can be used to accessing damage and risk, help decision-making on rescue and control the hazard. When there is fire, we can send the quadcopter to have a better view of incident. When there is natural disaster such as volcanic eruption, people can use quadcopter as a tool for assessing the damage after disaster or understanding the situation.

Search and Rescue

Multicopter can be used to find lost people and rescue them.

University and research

Multicopter is helpful in multiple studies, such as archaeology, geography and agriculture. It is useful in many fields, such as flight control, navigation and robotics.

Aerial Mapping

Multicopter is useful in constructing larger map for GIS system or mapping software.

Inspections of danger place

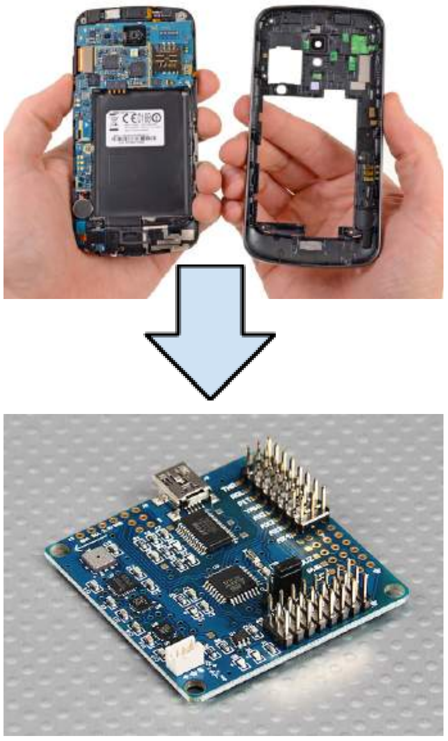
Compared with manned aircraft, multicopter is cheap and safe to inspect dangerous places such as powerline. Also, people needed to go to some potential dangerous place when constructing a building or bridge. If we can use multicopter to help to do this, it can eliminate dangerous action carry out by human.

Aerial Photography and Aerial Videography

The aerial photos and videos taken by multicopters can be used as marketing and promotion material of activities, properties and any other businesses.

14 Visions

Reducing the weight



The main disadvantage of using an Android phone as a flight controller is that the mobile phone is usually heavier than traditional flight controller board. Because of this, the flight time is heavily affected. One of possible way to reduce the weight is removing the case and the screen of a phone. Unfortunately, the task is not as easy as we imagine, because we cannot control the phone without the screen.

Ultimately, manufacturers actually can develop and produce a small flight controller which is based on Android. As we can see, Android mobile phone's motherboard is not very big, it is possible to re-design the motherboard to smaller PCB. The firmware can pre-installed our AndroidCopter application. After that, we can control it through the Remote Control Panel, so no screen is needed.

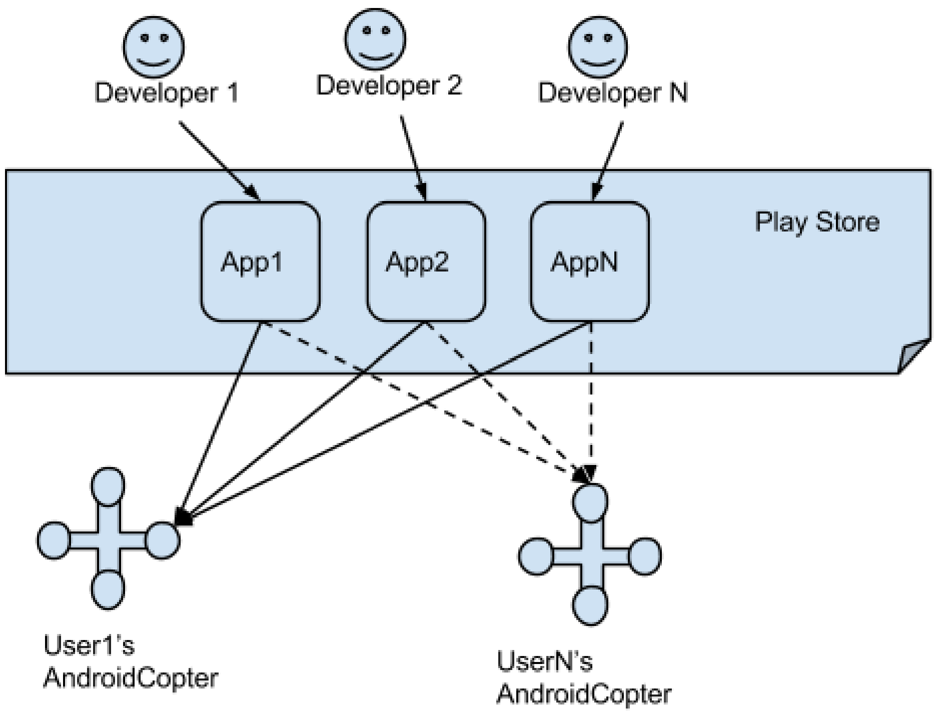
Productize

Although the application development of AndroidCopter is not difficult, it is still very hard to make a flyable quadcopter for those developers who have not touched these hardware before. And some toy-level quadcopters usually cannot be modified to connect to the IOIO board and the Android phone.

If we can have a product which has installed all essential components, what a normal user needed to do is turning on the power of the quadcopter, then control the AndroidCopter through the panel. For the developers, they only need to connect the AndroidCopter to their PC via USB port, they can thus develop their application using any Android IDE.

Rich Applications on Play Store

Any developers can make their own AndroidCopter applications and publish it on the Play Store. On the other hand, normal users can install these applications on their AndroidCopter to upgrade the functions of their AndroidCopter easily without any difficult tasks such as flashing rom into the quadcopter.



HazMat

People may not notice of placard sometimes, we can send a quadcopter flying around to give the message to public.

Police

Police can make use of quadcopter to find the wanted.

15 Conclusions

To conclude, the AndroidCopter is developed in this year. The AndroidCopter is fully controlled by the Android system. The Android Copter is able to perform autopilot features and fly stably.

We start this project in an early stage. We start the background searching in the summer. This project is an challenging project to computer science students which involves quite a lot of hardware knowledge. We have studies hardware like IOIO board, optical flow sensor on our own. Luckily, our supervisor and his team have gave great support to our team. We were able to solve difficulties and achieve satisfactory result.

Furthermore, there are many benefits of AndroidCopter, it is a low-cost and flexible design. we hope these features will be able to make quadcopter application becomes popularize.

16 Contributions

Summer

1. Bought hardware and a quadcopter
2. Learned basic concepts of quadcopter
3. Try to control a quadcopter with a remote control
4. Learned IOIO board
5. Designed AndroidCopter
6. Soldered components

SEM1

1. Android App
    a. Implemented Flight control logic (Stabilization using accelerometer and gyroscope)
    b. Implemented sonic sensor logic (get current height)
    c. manual control through Remote Control Panel with Xbox360 controller
2. Socket Server
    a. Setup a server on VPS
    b. Relay commands between remote control panel and Android App
3. Flight testing

SEM2

Find out a possible solution to improve the stabilization

1. Android App
    a. Re-designed AndroidCopter so it can fits the optical flow sensor
    b. Improved Flight control logic
    c. Hover in a fixed point using Optical flow sensor
    d. Path tracking using Optical flow sensor
    e. Handled new commands which are generated by visual programming.

17 References

[1] "IOIO Supported Devices"
<https://github.com/ytai/ioio/wiki/Supported-DevicesIOIO>

[2] “PID Controller - Wikipedia”
<http://en.wikipedia.org/wiki/PID_controller>

[3] “multicopter”
<http://en.wikipedia.org/wiki/Multirotor>

[4] “Build A Quadcopter From Scratch – Hardware Overview”
<http://blog.oscarliang.net/build-a-quadcopter-beginners-tutorial-1/>

[5] “Application of Quadcopters within Public Safety”
<http://droneflyers.com/talk/threads/application-of-quadcopters-within-public-safety.523/>

[6] “Application of Quadcopters within Public Safety”
<http://diydrones.com/forum/topics/application-of-quadcopters-within-public-safety?commentId=705844%3AComment%3A1717972&xg_source=activity>

[7] “Multirotor Applications”
<http://www.versadrones.com/#!hexacopter-applications/cee5>

[8] “Quadcopters”
<http://en.wikipedia.org/wiki/Quadcopter>

[9] “Quadcopters”
<http://www.slideshare.net/aakashgoyal3532/quadcopter-33355358>

[10] “What makes the quadcopter design so great for small drones?”
<http://www.quora.com/What-makes-the-quadcopter-design-so-great-for-small-drones>

[11] “IOIO”
<https://github.com/ytai/ioio/wiki>

[12] “Optical Flow Sensor”
<http://copter.ardupilot.com/wiki/optical-flow-sensor/#How_it_works>

[13] “Odometry”
<http://en.wikipedia.org/wiki/Odometry>

[14] “Obstacle avoidance”
<http://en.wikipedia.org/wiki/Obstacle_avoidance>

[15] “Optical flow”
<http://en.wikipedia.org/wiki/Optical_flow>

[16] “ADNS3080 Optical flow sensor now available in the DIYDrones store!”
<http://diydrones.com/profiles/blogs/adns3080-optical-flow-sensor-now-available-in-the-diydrones-store>

[17] ”Visual programming language”
<http://en.wikipedia.org/wiki/Visual_programming_language>

[18] “What are the advantages and disadvantages of visual programming languages?”
<https://answers.yahoo.com/question/index?qid=20070823112111AAd0Onw>

Appendices

Appendix 1: Casualty List

From the start to current stage, the project have some damages on the components. We hope we can reduce as much damages as possible to achieve cost effectiveness of the project. Here is the casualty list in our record.

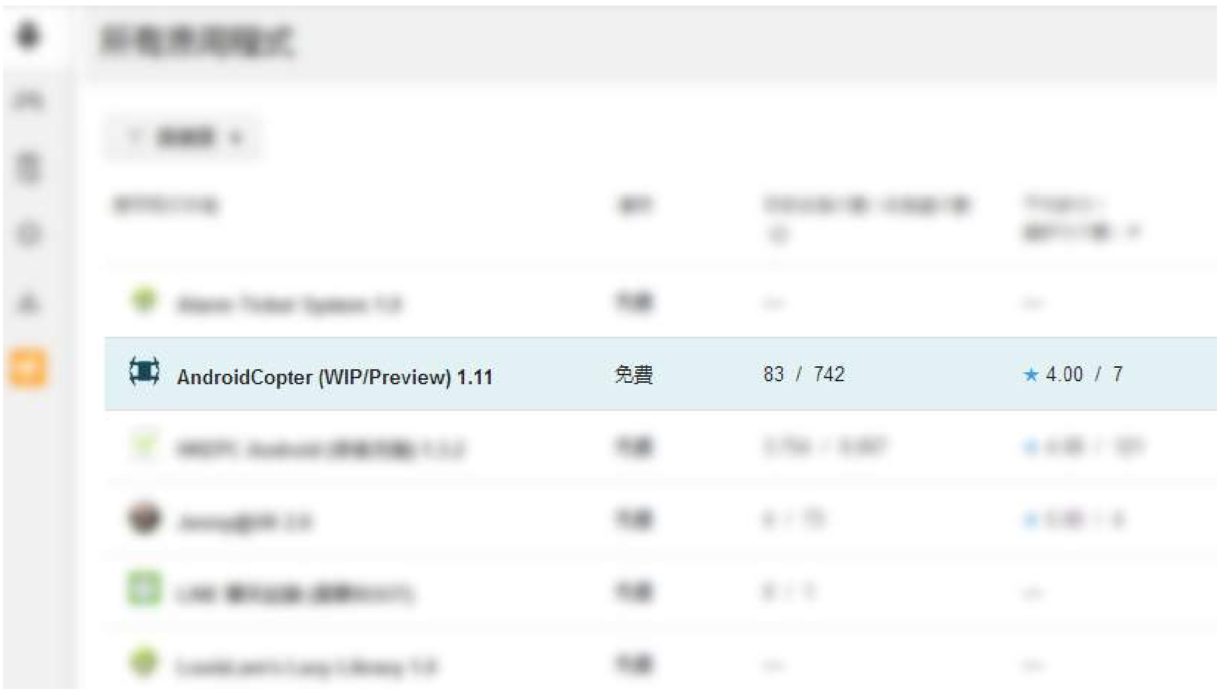
Item Name	Unit
Propeller	22
IOIO board	1
Motor	1

Appendix 2: Product List

Item Name
AndroidCopter (Hardware)
AndroidCopter (Android Program)
Websocket Server (PHP)
Remote Control Panel (JS + HTML)

Appendix 3: Google Play

We have put the AndroidCopter app on Google Play for our convenience purposes only. Surprisingly, there are more than 700 users downloaded the app and the app got 4-star averagely.
(https://play.google.com/store/apps/details?id=cuhk.fyp.androidcopter)



[illegible]	[illegible]	[illegible]	[illegible]
AndroidCopter (WIP/Preview) 1.11	免費	83 / 742	★ 4.00 / 7
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]