
AndroidPhoneCopter
LYU1401 AndroidCopter
Lam Ka Ho
Wong Chor Man

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 and the IOIO board. 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 limitations of AndroidCopter are also discussed.

Table of Contents

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).

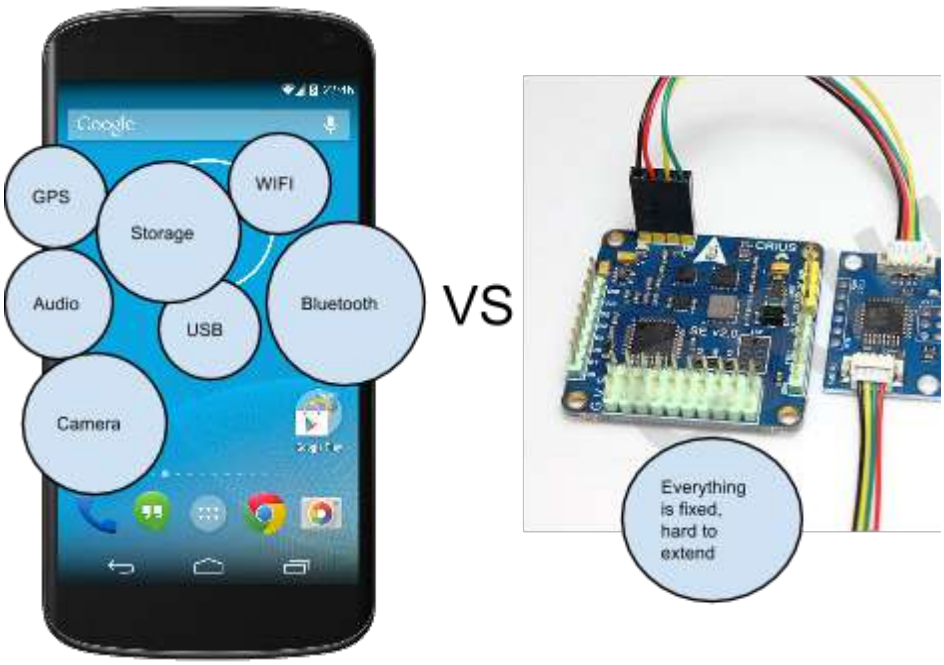
With infinite possibility of Android extension, it is expected more and more amazing Android application will be developed in the coming years. In this project, we will try to explore the possibility of Android. We will try to equip a quadcopter with Android and develop some meaningful features on it. 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.

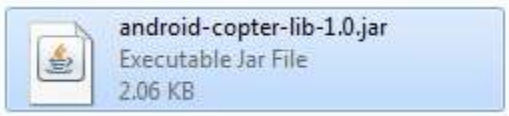
On the Software side, Android has a huge community of Applications, Application Frameworks and Libraries. Many support libraries can be found, developers can focus on the programming on program flow and logic. There are libraries for iBeacon, Bluetooth and Websocket which provides functionality to the 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



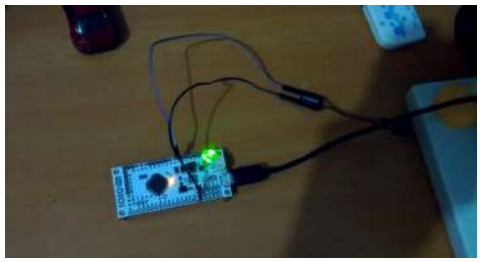
When a developer need to develop a custom quadcopter application, he need to know many background information of quadcopter. He will need to know what values of input are needed to make it move and the flight control theory. However, not much developers know these knowledge before the development. The 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.

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.

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.

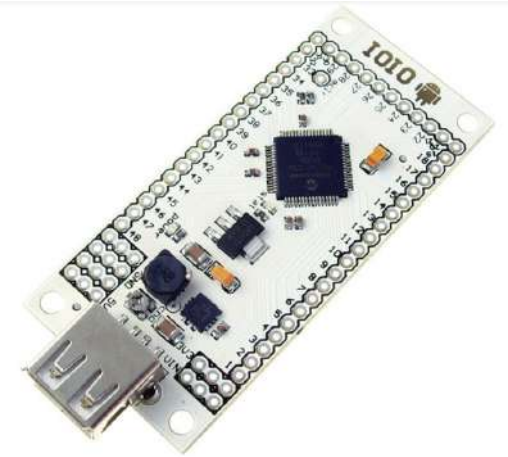
To adjust the roll of the copter which makes the copter go to left or right, it is controlled by turning up the speed of one rotating rotor and turning down the speed of rotor on the opposite side.

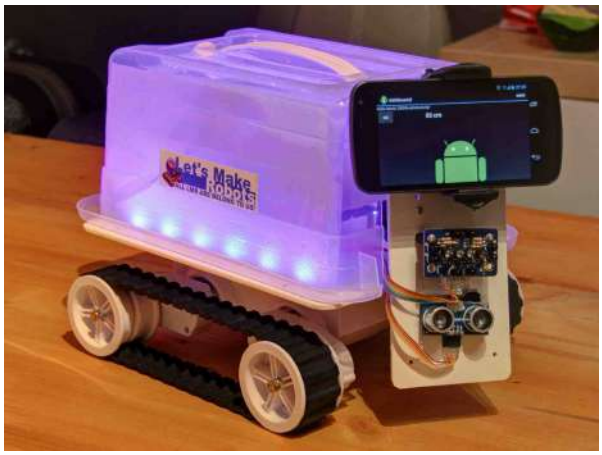
To adjust the pitch of the copter which makes the copter move forward and backward, it is controlled by similar method as roll.

3 Study of IOIO Board

3.1 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.





3.2 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.

3.3 Usage in this projects

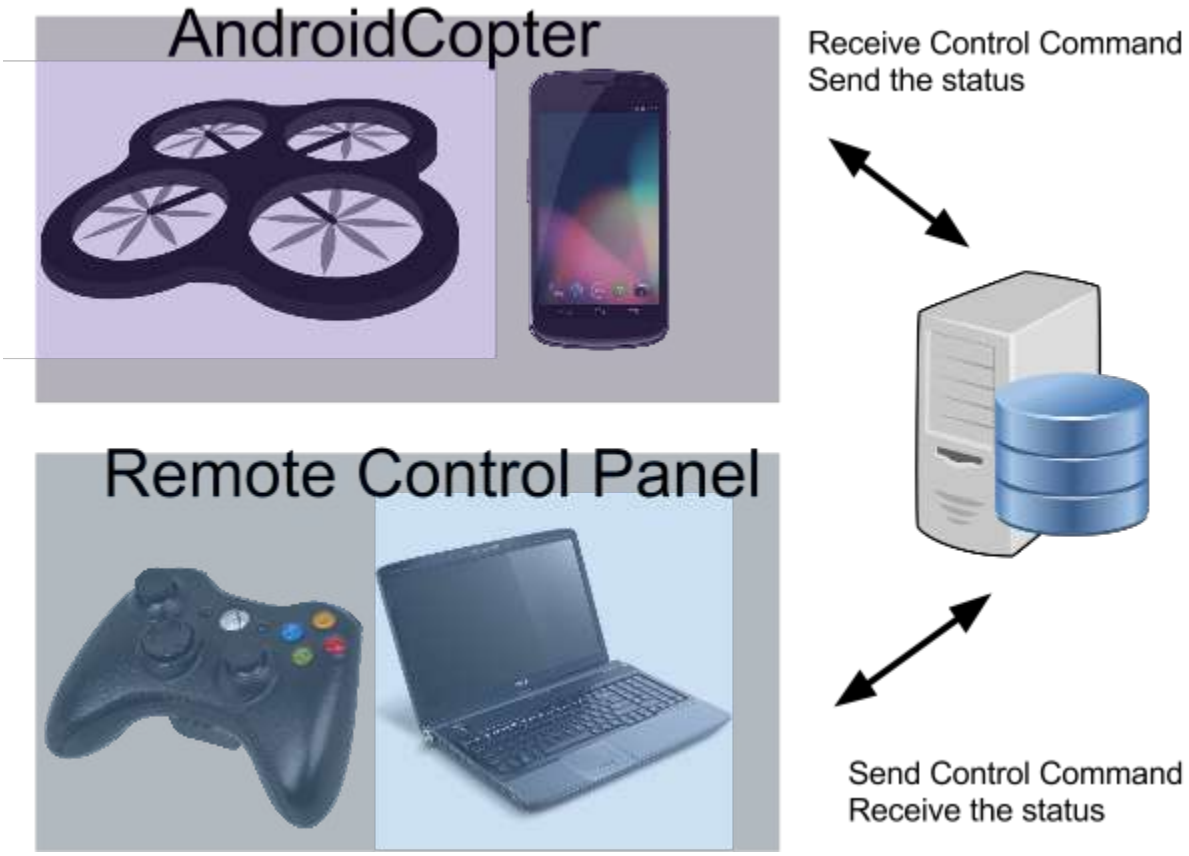
In this project, the IOIO board is the 'bridge' between Quadcopter and Android. Firstly, the Android application calculates the motors' speed and then send the values to IOIO. After that, the IOIO will convert the values to PWM signals. Finally, the signals will be passed to the speed controllers (ESC).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

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

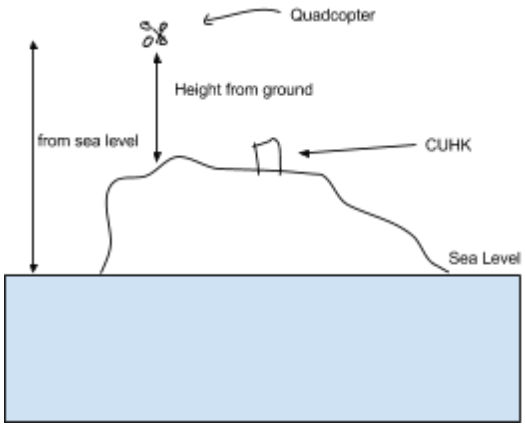
Most frame sets with **brushless electronic speed controls (ESCs)** and **brushless motors** are supported.

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.

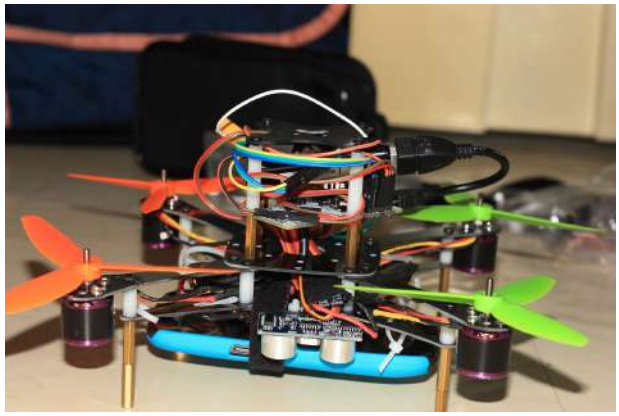
android.permission.BLUETOOTH

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

android.permission.ACCESS_FINE_LOCATION

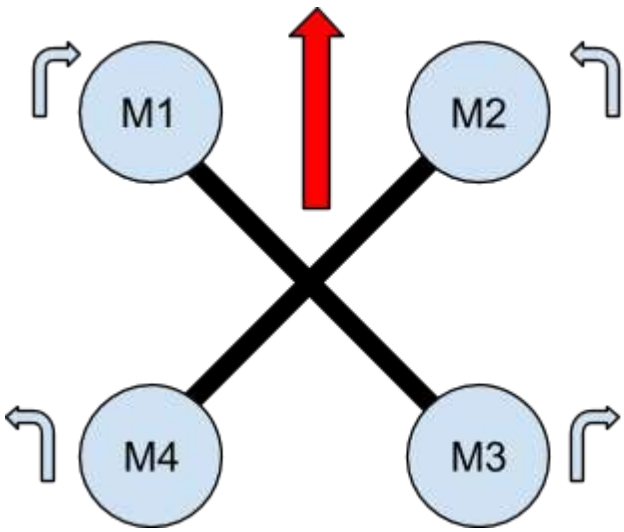
The application can get the GPS information with the permission.

5.3 Design



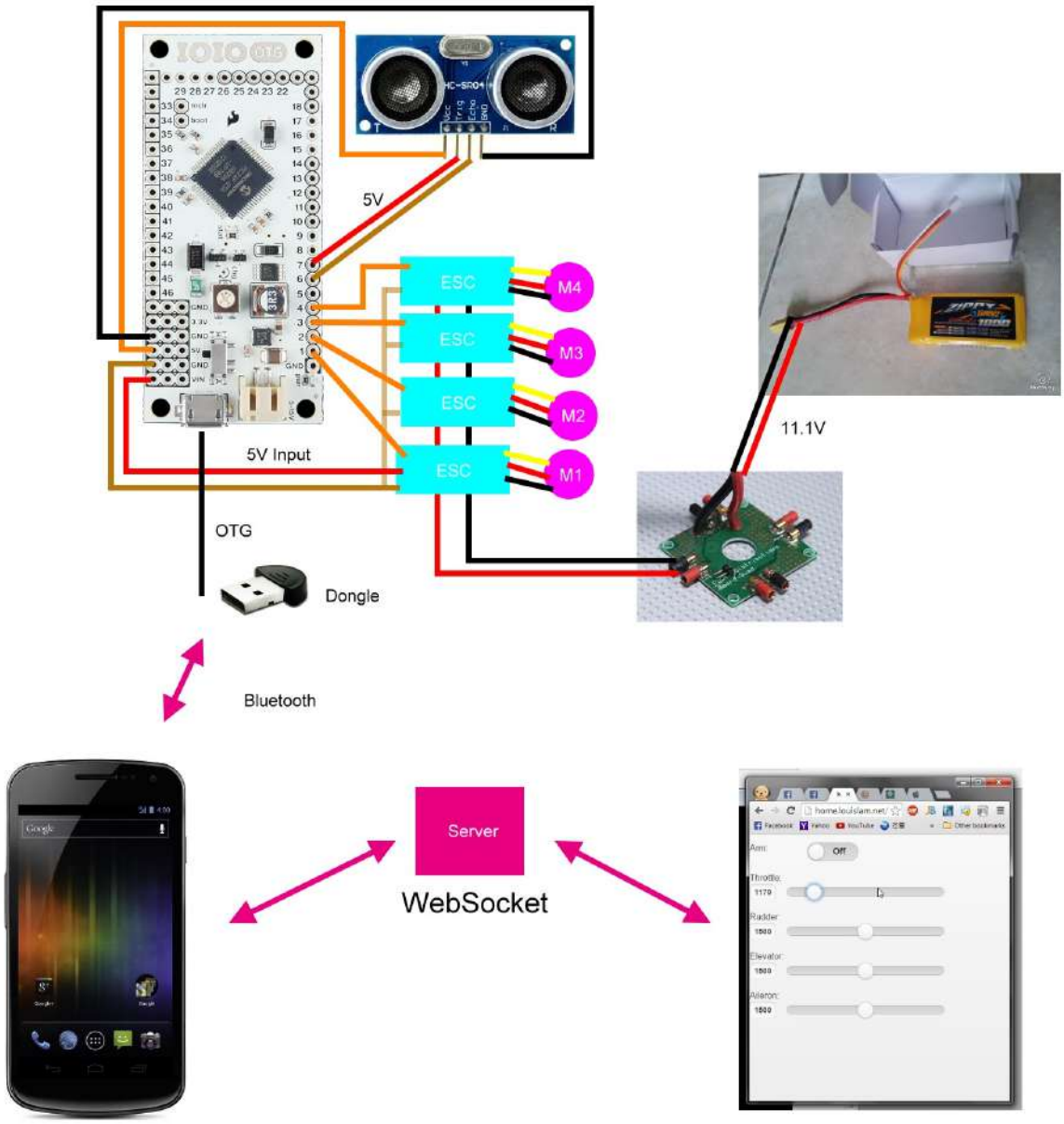
230mm quadcopter frame is used for the project. The mobile phone is installed below the frame, this installation can capture clearer photo. The IOIO board is installed on the frame.

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



Wiring Diagram description

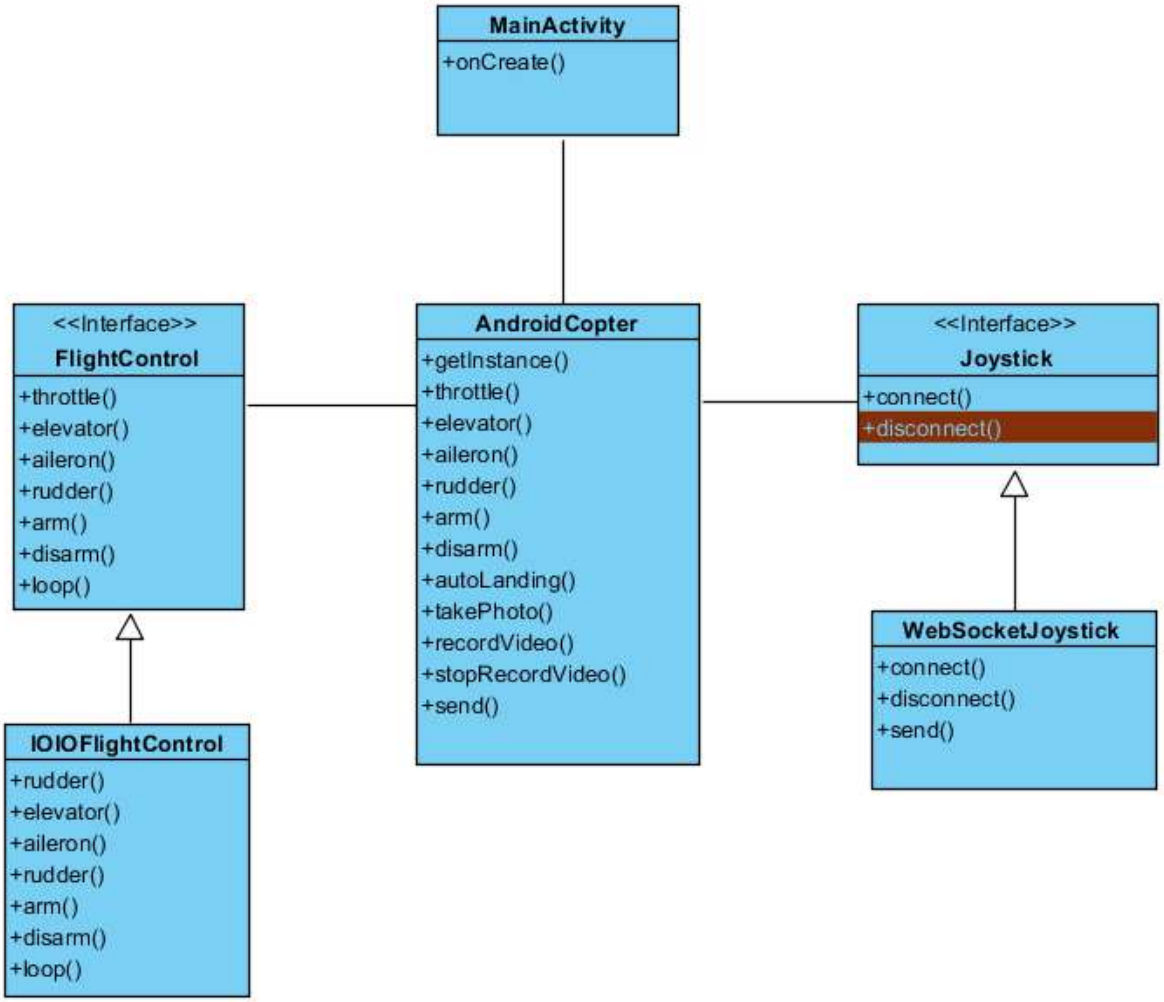
The basic setup is very similar to the traditional quadcopter.

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



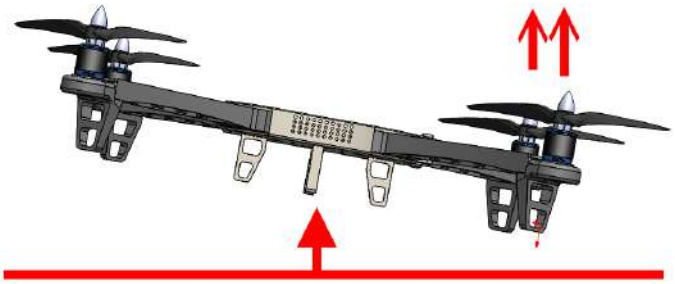
MainActivity is an Android Activity object. An AndroidCopter object can be created by calling getInstance() method. An AndroidCopter object has a FlightControl object and a Joystick object and both of them are interfaces. Currently, we have implemented IOIOFlightControl and WebSocketJoysitck, so that 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 * K1);
```

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)

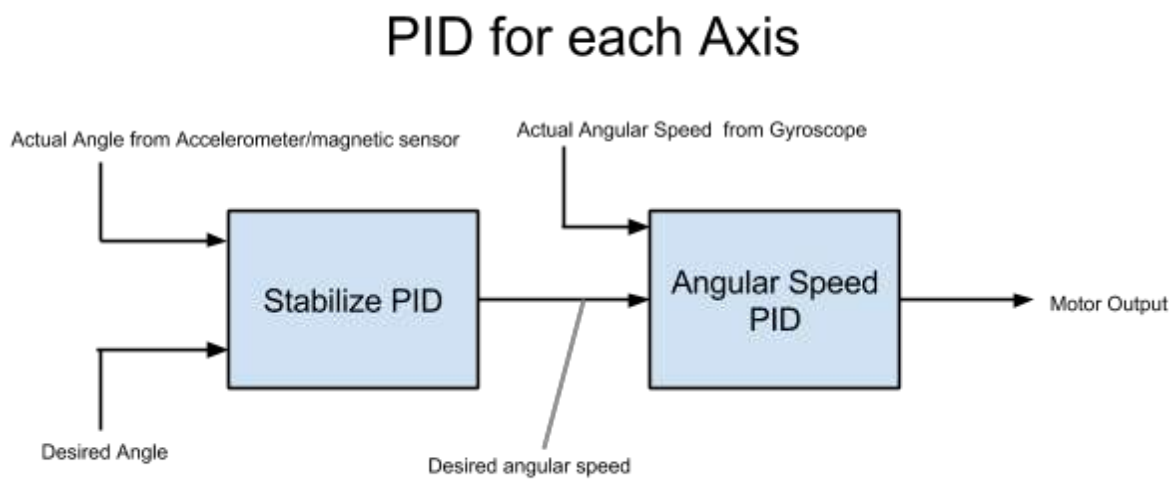
```
m1 = m1 - (accX * K2);
m2 = m2 - (accX * K2);
m3 = m3 + (accX * K2);
m4 = m4 + (accX * K2);
```

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();
```

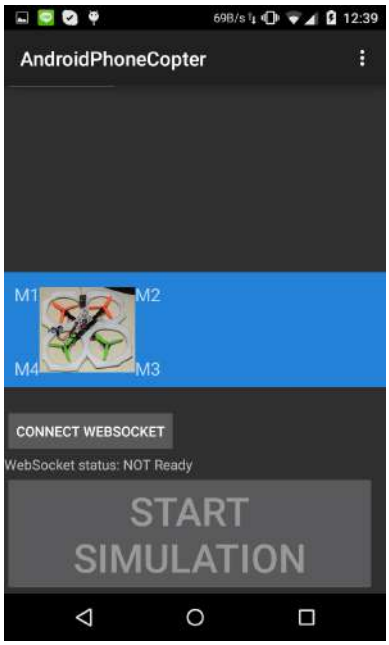
Algorithm in JAVA:

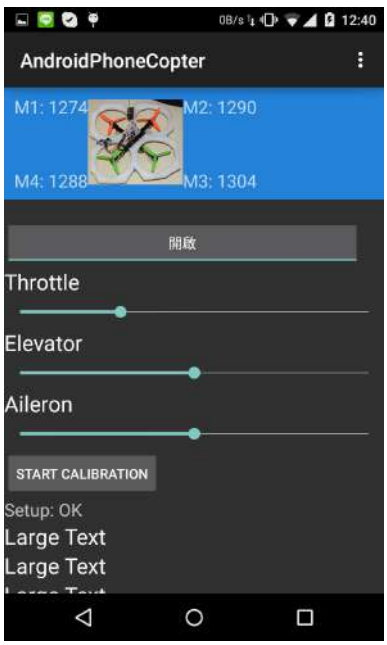
```
// Desired Throttle
m1 = m2 = m3 = m4 = throttle();

// 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();
```

5.4 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.

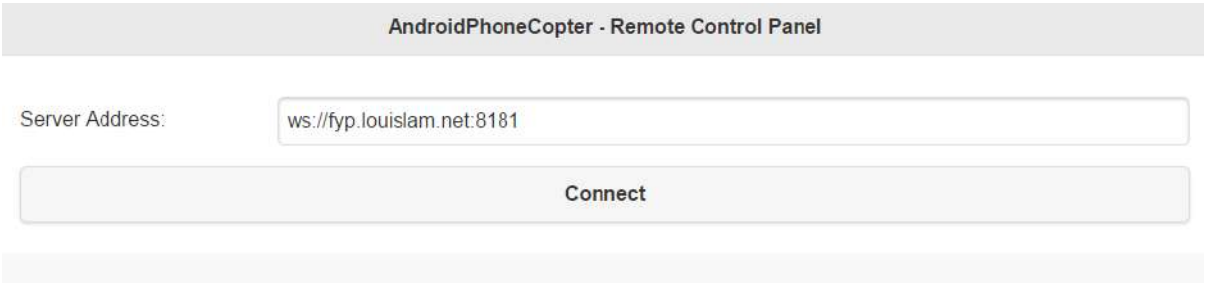
6.1 Implementation

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

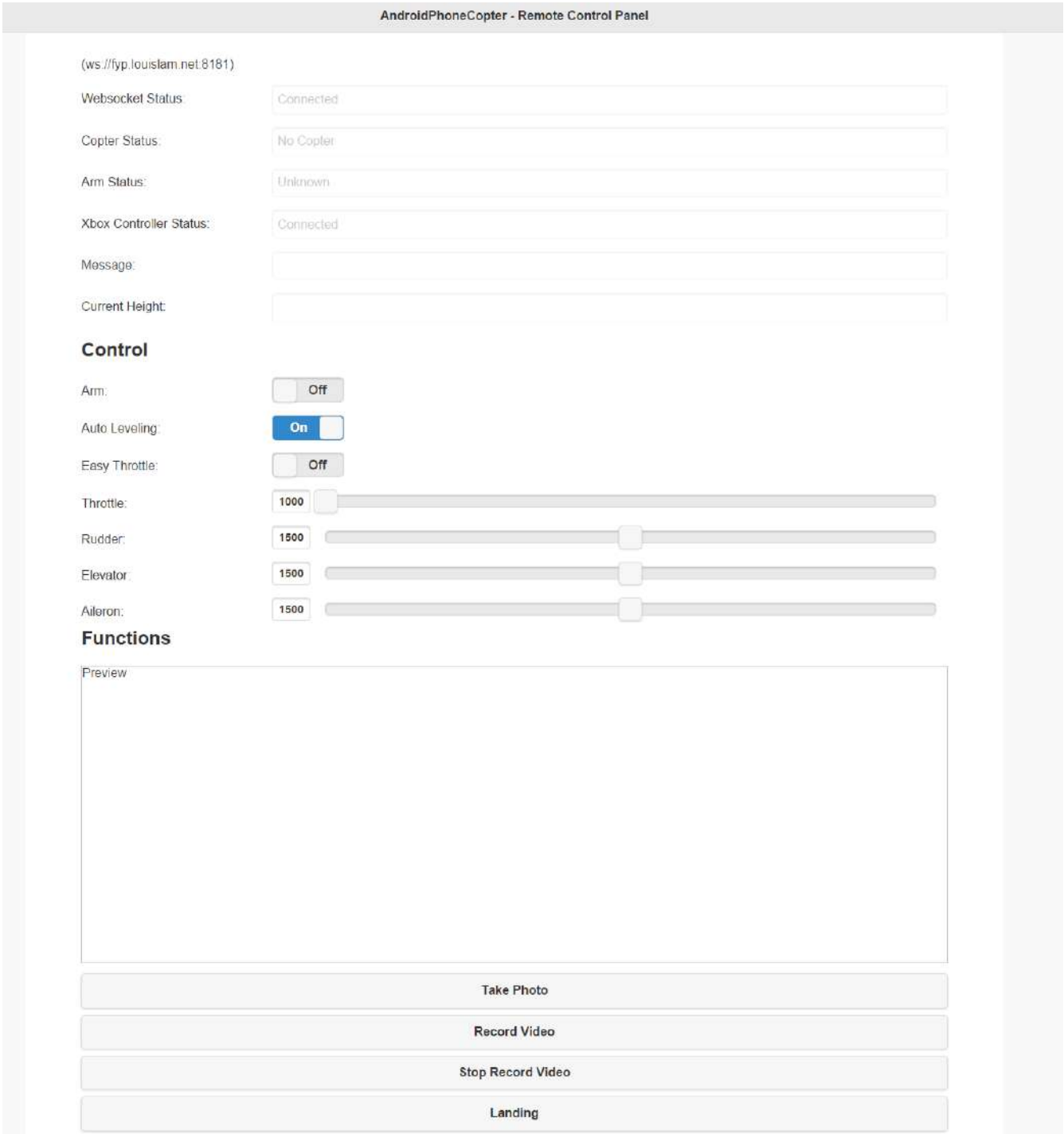
6.2.1 Connect Page

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://.



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
Preview
Take Photo
Record Video
Stop Record Video
Landing

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)

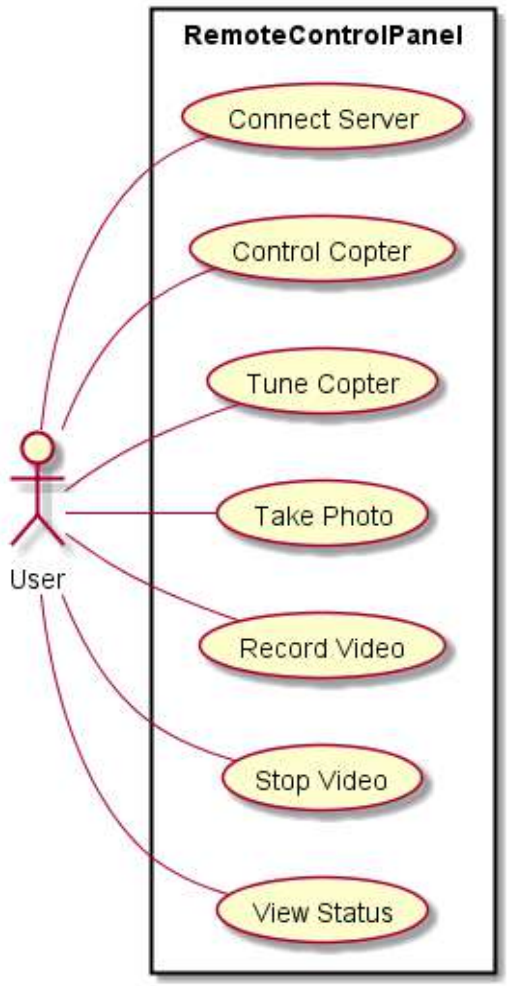
	Send a command or message with specific value to the Remote Control Panel.
void	send(JSONObject json) Send commands or messages in JSON format.
void	reloadSettings() If some values are changed, this method should be invoked to apply changes.
void	enableAutoLeveling(boolean val) This method can be invoked 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.

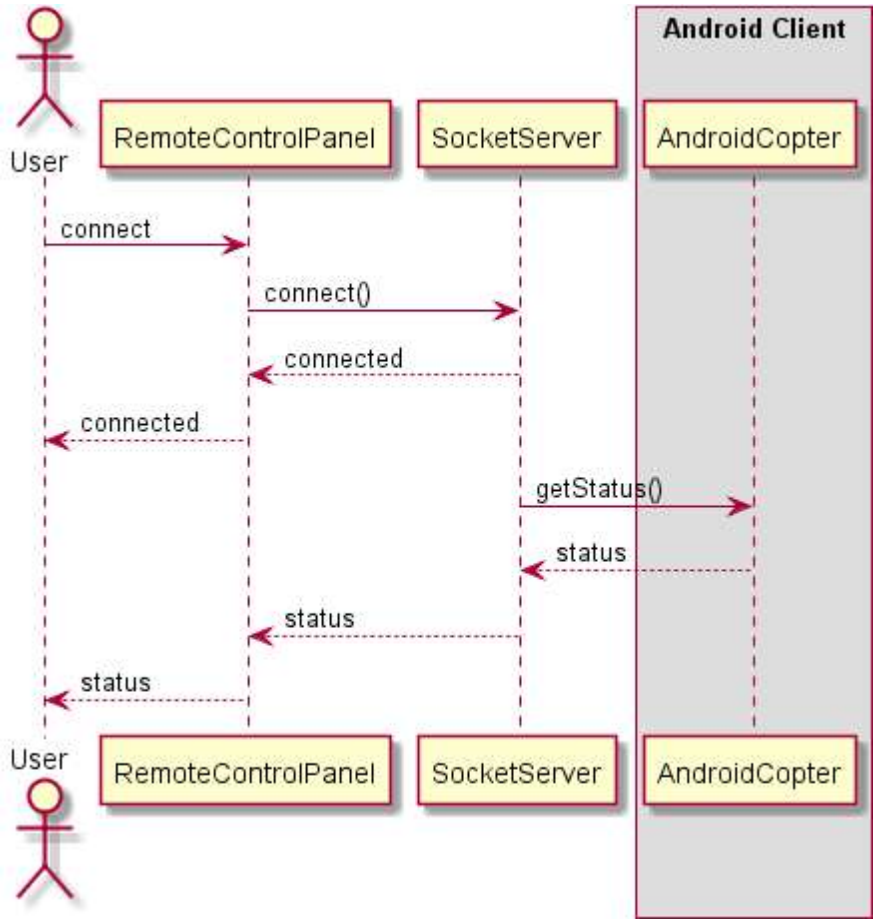
8 Design Diagram

8.1 Use Case Diagram



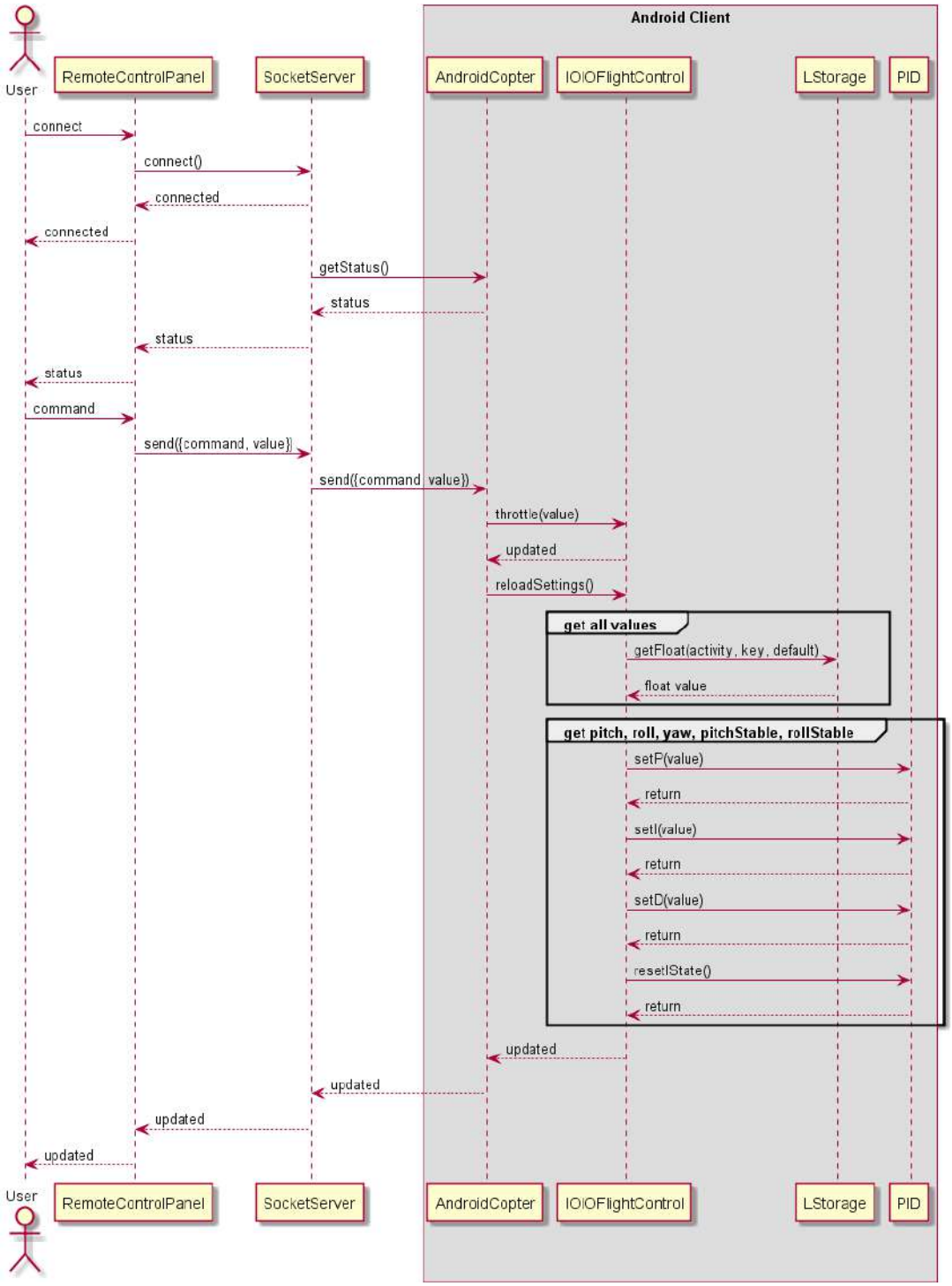
8.2 Sequence Diagram

8.2.1 Connect Server



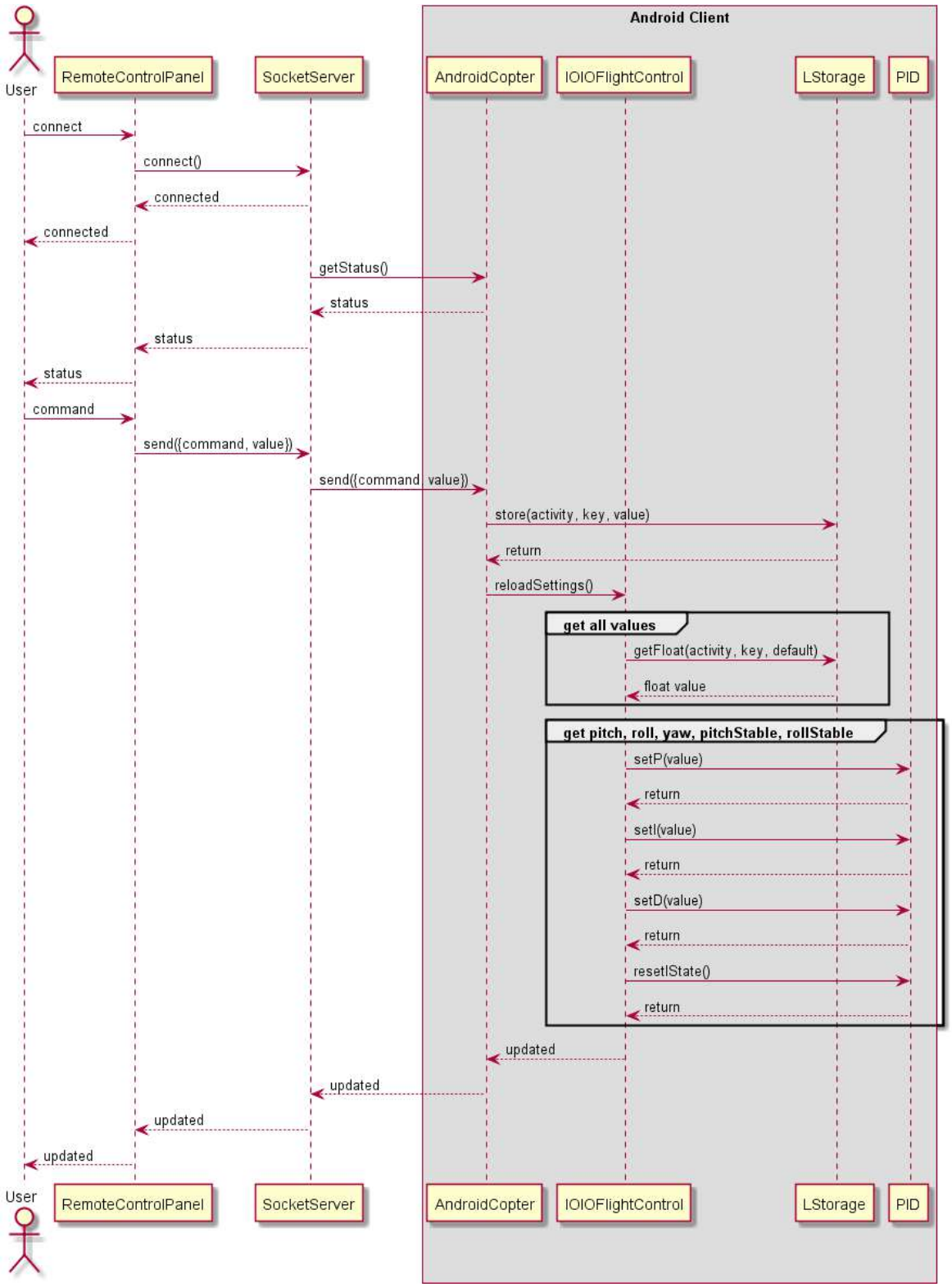
8.2.2 Control Copter

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

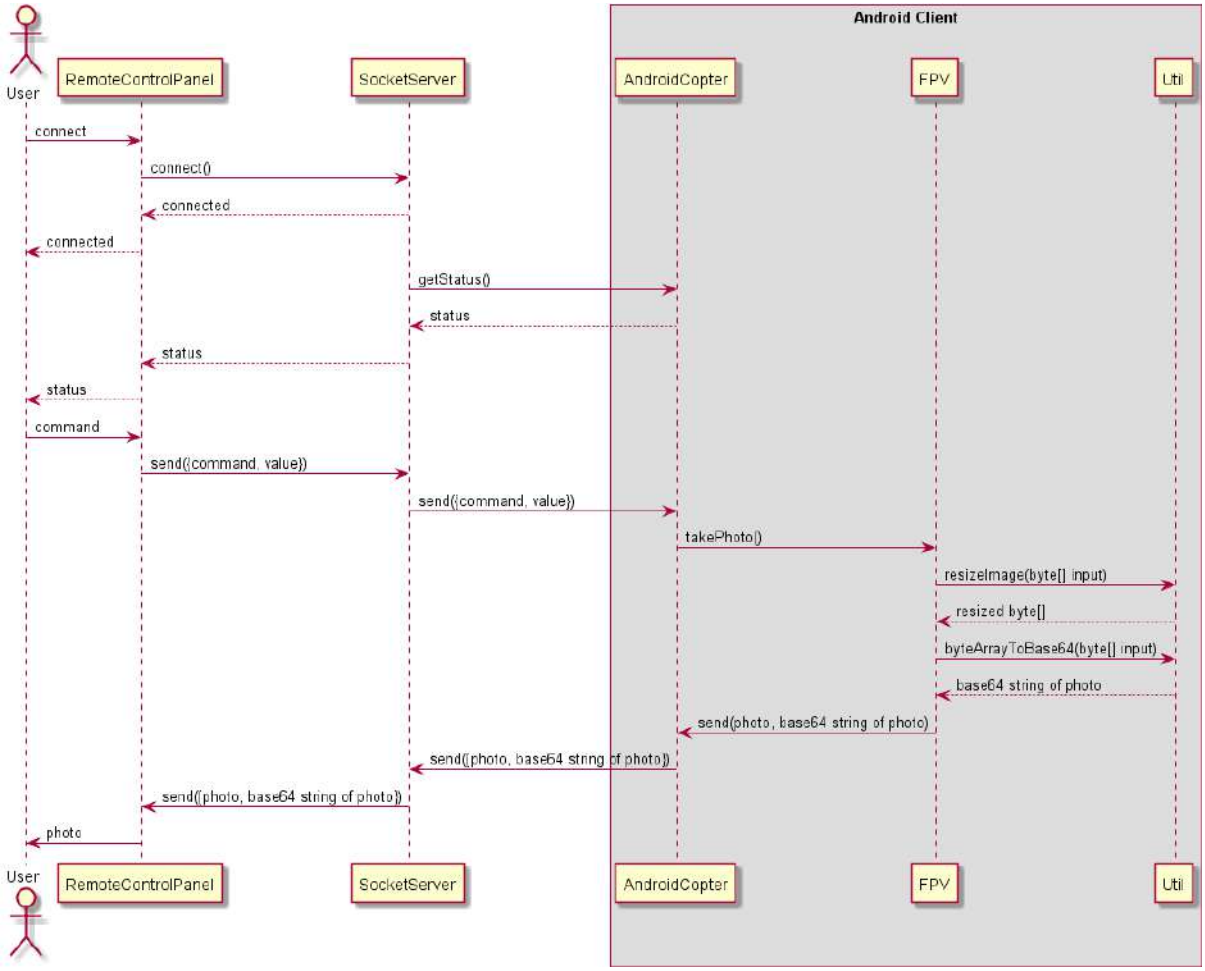


8.2.3 Tune Copter

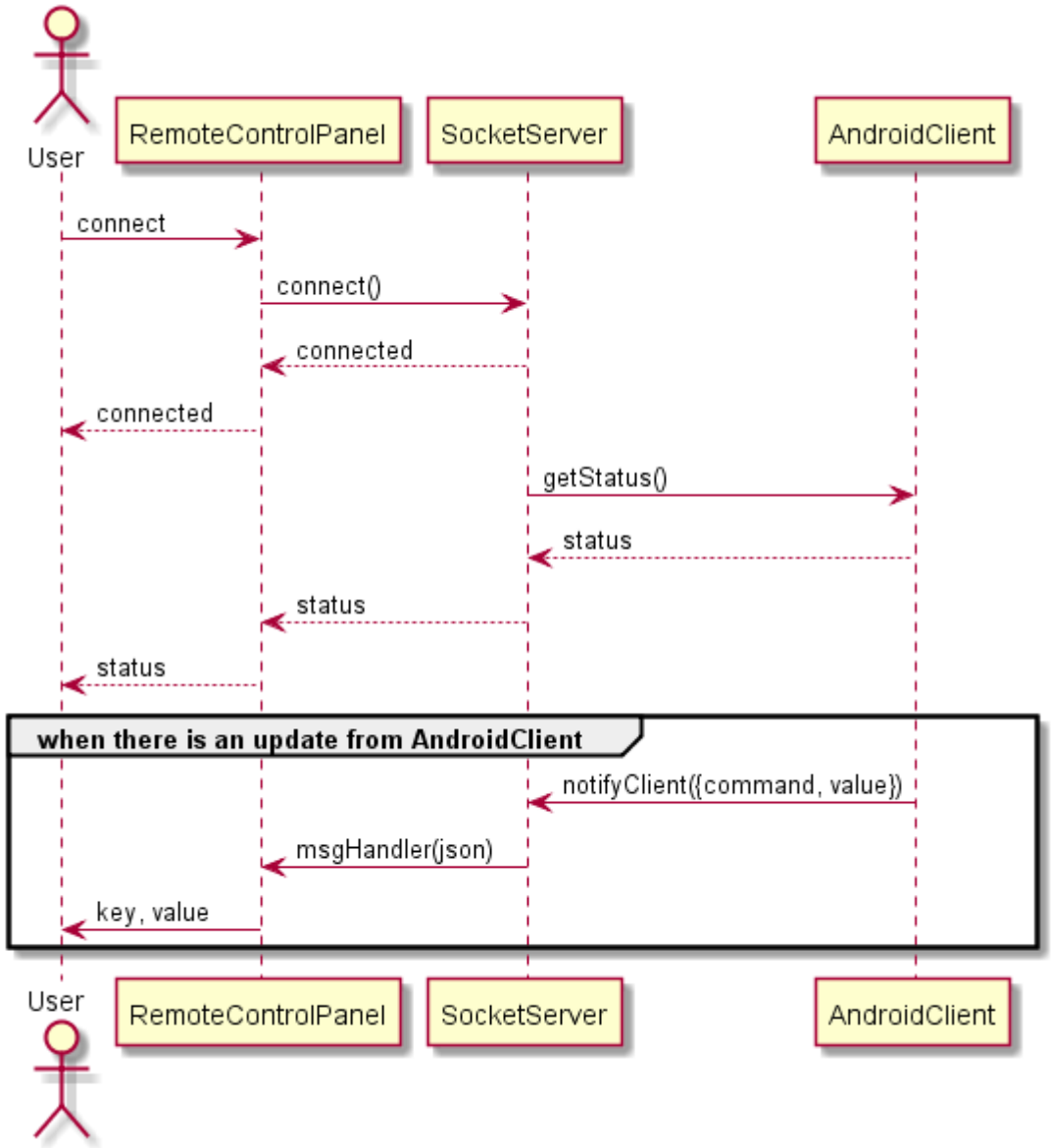
For variables including PitchP, PitchI, PitchD, rollP, rollI, rollD, yawP, yawI and yawD, they are tuned via the Tune Copter function.



8.2.4 Take Photo



8.2.5 Receive Message



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

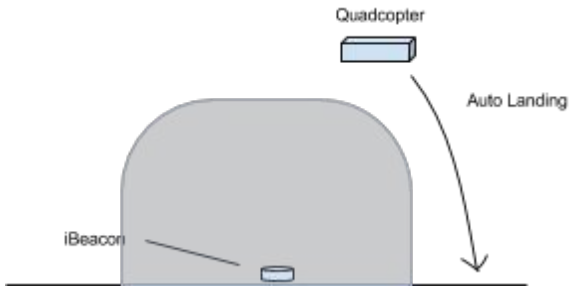
9 Safety Measurement

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.

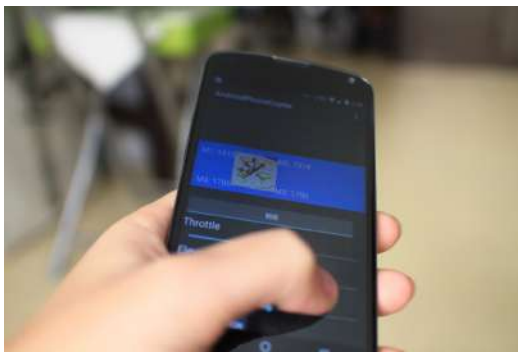


Using iBeacon can avoid this situation. iBeacon is mainly used for an indoor positioning system which is develop by Apple Company. In this project, iBeacon is used for ranging and only one piece of iBeacon is needed. The quadcopter can get the approximate distance from the iBeacon. So if the quadcopter 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

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.



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

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.

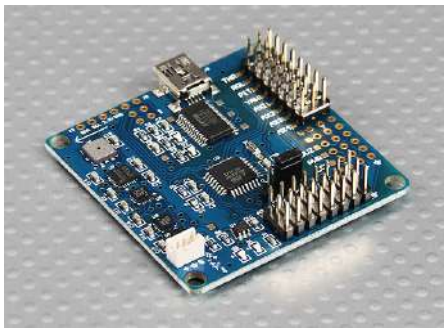
Aerial Photography and Aerial Videography

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.

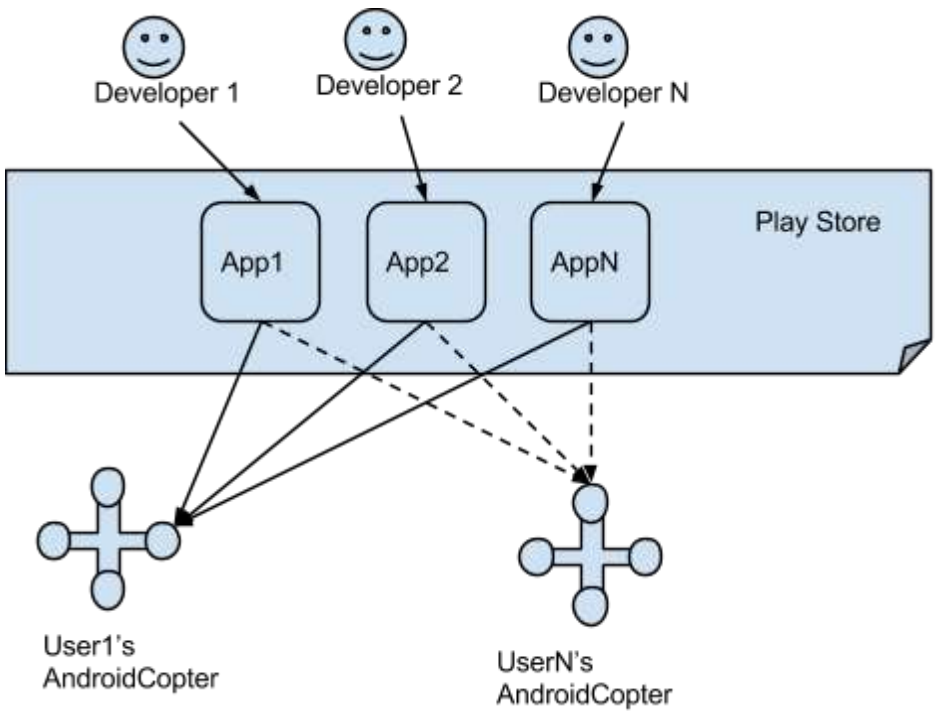
Productize

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 Further development

There are many goals that we would like to achieve in next term.

15.1 Auto Pilot

There will be a google map on the RemoteControlPanel. Users are allowed to plot the points of route on the map. The quadcopter will have an auto flight based on the route given. User can also specify the kind of action to be performed in certain points. For example, photo taking and sensor measuring are needed in point A, video taking is needed in point B.

15.2 Pre-scheduled Auto Pilot

The functions of a quadcopter can be triggered in specific time. When the specific time is reached, the quadcopter will start an auto pilot and perform the pre-scheduled tasks. By having this function, quadcopter will be able to perform repeated tasks several times a day and no human control will be needed.

15.3 Optical Flow Alogrithm

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.

16 Conclusions

As a result, the initial version of AndroidCopter is developed in this term. The AndroidCopter is fully controlled by the Android system. No 3rd party flight control board is needed.

However, the final product is a prototype only. Especially for hardware part, because most quadcopter frames are not expected us to put a mobile phone on it, we just stick the mobile phone under the quadcopter using a velcro only. In the future, we may consider putting our project on Kickstarter.com, so that we can have money to make it into a real product.

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.

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>

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	21
IOIO board	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 200 users downloaded the app and the app got 2 of 5-star.
(https://play.google.com/store/apps/details?id=cuhk.fyp.androidcopter)