Abu Dhabi University

Capstone Project

Progress Report

Hand Gesture Controlled Emergency Aerial Assistance Using Smartphone Based 4G Quadcopter

---

Team Members:
Muhammad Obaidullah 1030313
Sifat Sultan 1003289
B.Sc. Electrical Engineering
Abu Dhabi University, U.A.E.
Communication Major

Supervisor:
Dr. Mohammed Assad Ghazal
Acting Chair, Department of Electrical and Computer Engineering
Lab Instructors:
Eng. Ahmed Sweleh
Eng. Ibrahim Ibrahim

May 25, 2014

In The Name of Allah, The Most Beneficent, The Most Merciful.

Contents

Abstract

This project aims to solve the problem of providing emergency medical or any other type of assistance in a un-accessible area by using a 4G quadcopter which is controlled by hand gestures. The hand gestures provided by the user will be used to maneuver the quadcopter to reach the emergency location and provide assistance. The uniqueness of this project lies in the fact that it uses a 4G smartphone to connect the quadcopter to the base station for getting the user commands, providing the GPS location, and live video stream of the quadcopter's vision.

1 Project Background

The number of researches and developments on the unmanned aerial vehicles have increased over the past years. Many hobbyists and robotics enthusiasts are particularly interested in quadcopters because of their ability to hover in place, take-off and land vertically. [4] People can easily now buy and assemble a quadcopter with low cost. Within seconds a quadcopter can lift-off reach a point and land back. This ultra fast response and ease of control make the quadcopter ideal for reaching areas dangerous for humans and performing tasks.

Parallel to this, the field of Human-Computer Interaction has been a growing field in the past few years due to the introduction of innovative interaction devices such as Nintendo Wii Remote and Microsoft Kinect. The idea of using gestures as a mean of interaction has become the recent trend as it makes the experience more natural and hence the possibility of application.

Quadcopter: The term quadcopter or quadrotor comes from the four narrow chord propellers used by the air vehicle in order to maneuver in air. The quadcopter uses these four propellers to push the air downwards in order to generate lift force. It maneuvers by varying the speeds of these four motors.

Leap Motion: In 2012, the Leap Motion controller was introduced. Leap motion comes with a 150 degree field of view and makes use of a depth sensor to follow and scan the hand features up to 1/100th of millimeter. This extremely precise control gives us the opportunity to use it as our interface over Kinect which tracks body instead of hand since our gestures will be mainly finger movements.

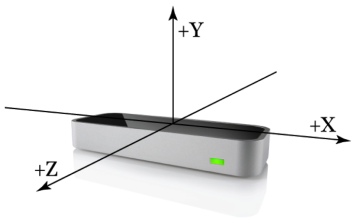


Figure 1: A leap motion device with 3D coordinates.

The past few years there have been extensive research on autonomous micro-aerial vehicles. The application ranges from hobby to serious circumstances.

Quad Copter Application XPROHELI developed quad copter that is used to take high quality video and photography. `http://www.youtube.com/watch?v=HoQQlRzybmA#t=14` This video was taken using the XPROHELI quad copter.

FAE developed tri copter that is used to take long range aerial surveillance over oil field. `http://www.youtube.com/watch?v=a5H5paygWQw&feature=youtu.be` Quadcopter has the potential to be used in a variety of application and we decided to work on a case where quad copter deserves to be used; to save human lives where time is of paramount.

2 Problem Discussion

In this stone hearted world, everyday accident happen and people die. These lives can be saved by timely medical response. But the response faces countless obstacles sometimes like traffic jams, remote places, radiation etc. This is a growing problem causing engineers and specialists to think and come up with different solutions.

UAE itself has a mortality rate of about 37 per 100,000 which is one of the highest in the world. [1] This mortality rate can be further skimmed down by providing timely first aid kit to seriously injured people.

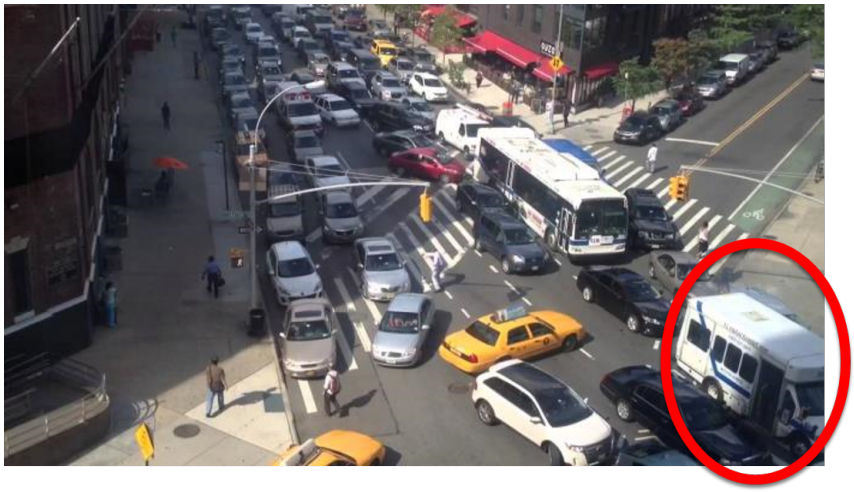


Figure 2: Lives are lost as fast-aids do not reach on time.

So all in all, our problem statement is how to provide quick and timely emergency assistance without the restriction of the range by using low-cost quadcopter.

3 Project Description

We proposed a system which is complete package and will solve all the problems of providing quick emergency assistance. The proposed system was widely accepted by the faculty members and approved.

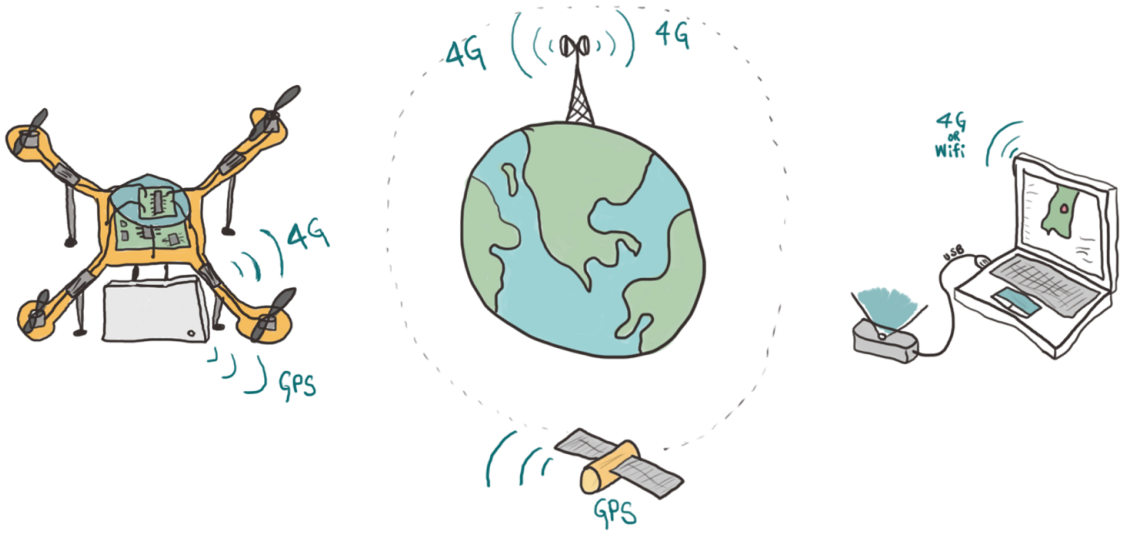


Figure 3: The system diagram of the project.

Our system has a Quadcopter with a gyro control chip which controls the basic maneuvers for the UAV and we propose to build a custom designed chip to decode the signals coming from the on-board android smartphone and send commands to the controller chip. We used a android smartphone to provide the quadcopter with GPS, Camera, and 4G internet connection. So this quadcopter is now capable of performing completely unmanned flight and be controlled using http requests coming to it through 4G connection. The quadcopter also provides a http feedback to the computer about its GPS position. The GPS can also be used to make a flight plan for the quadcopter. The on-board mobile also provides a UDP live stream of the camera vision and can be controlled using.

Lets take a typical emergency assistance scenario and apply our proposed solution to it. A call comes to the police about a car accident that has happened on a highway in Abu Dhabi. This is emergency case it might take several hours for the police to arrive and analyze the situation. The police places a pin on the map in the quadcopter software in computer. The computer sends the http request to the quadcopter to move to that specific location. The quadcopter immediately starts up and goes to the GPS location. This is done with a feedback loop which calculates the current GPS location and the required GPS location. While all of this is done, the policeman can see the current location of the quadcopter on the map. When the quadcopter reaches the end of its destination, the policeman turns on the gesture control mode and starts controlling the movement of the quadcopter precisely using hand. Now the policeman can access the situation and can also possibly drop the first aid kit if the situation is too serious.

3.1 Using a UAV for agility and accessibility

Unmanned Air Vehicles are light, fast and can be designed to carry the essential aid equipment. Along with that unmanned vehicles can be used to perform critical operations in areas where human life cannot be risked. Furthermore, quick response to a distress call can differentiate between life and death of a person. Therefore, air emergency medical assistance is needed very quickly in some cases and the delay cased by set-up and start of manned vehicles cannot be afforded at this point.

3.2 Use of a quadcopter for ease of control and high payload delivery

With the use of four propellers pushing the air downwards with immense speed, a typical quadcopter is capable of lifting high payloads. UAVs for search and rescue operations are required to be small and at the same time carry high payloads. Quadcopter is perfect for such kinds of ease of control applications because of its usage of four high rpm brushless motor run propellers. These brushless motor speeds can be intelligently varied to provide 6 degrees of freedom by sensing the gyroscopic values of Pitch, Yaw, and roll which.

The quadcopter can be controlled using two types of configurations, one is the X configuration and the other is the + configuration. For a very stable flight and camera view being not blocked, we propose to use a X configuration quadcopter.

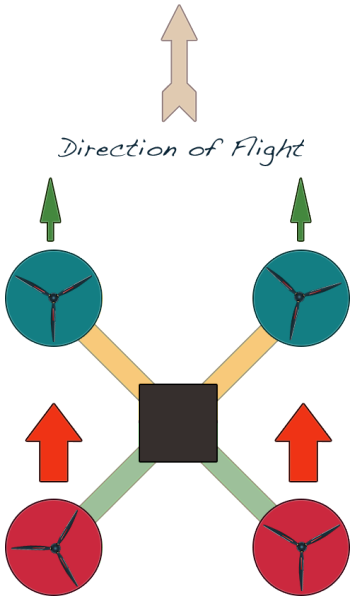


Figure 4: X configuration forward motion of quadcopter.

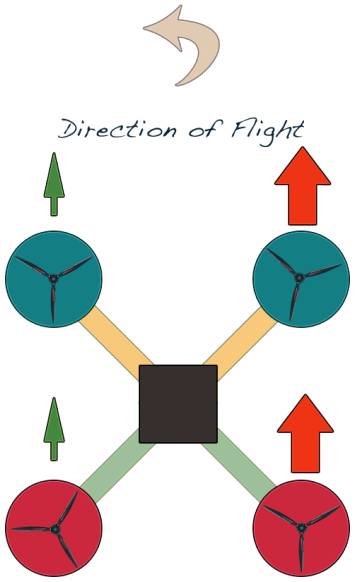


Figure 5: Left motion of quadcopter in X configuration

Forward Direction: By speeding up the back motors, a forward direction motion can be accomplished. This makes the thrust of the motors at an angle to the z-axis. The thrust has force components in the weight direction to balance or cancel the weight and a unbalanced forward component to make the quadcopter go forward.

Left Direction: By speeding up the right motors, a left direction motion can be accomplished. The thrust has force components in the weight direction to balance or cancel the weight and a unbalanced forward component to make the quadcopter go left.

Right Direction: By speeding up the left motors, a right direction motion can be accomplished. The thrust has force components in the weight direction to balance or cancel the weight and a unbalanced forward component to make the quadcopter go right.

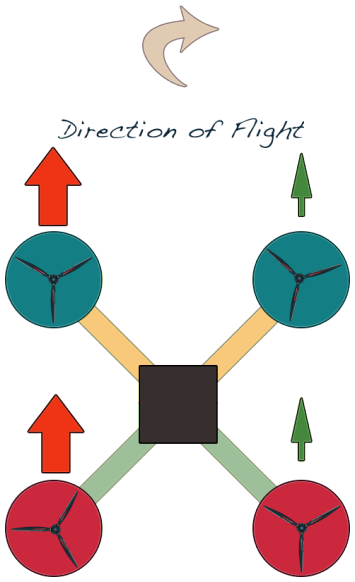


Figure 6: X configuration forward motion of quadcopter.

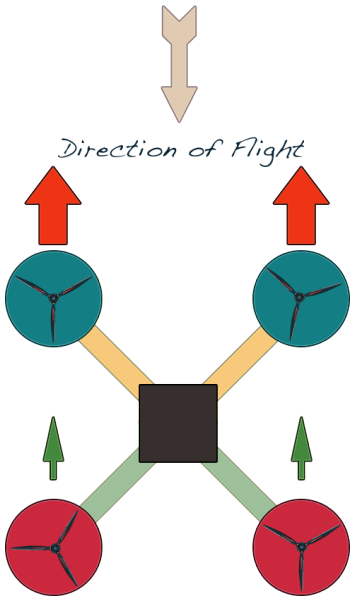


Figure 7: Left motion of quadcopter in X configuration

Backward Direction: By speeding up the front motors, a backward direction motion can be accomplished. This makes the thrust of the motors at an angle to the z-axis. The thrust has force components in the weight direction to balance or cancel the weight and a unbalanced backward component to make the quadcopter go forward.

3.3 Global control by using on-board 4G smartphone

Usually quadcopters have a limited range radio link but emergency assistance should not have limited range of control. It was decided to use 4G network for control and communication with the quadcopter as 4G network is accessible from within 35Km range of a cellular tower. Furthermore, a typical city is divided into cells and at almost each node a telecommunication tower is located. This is a huge advantage for our quadcopter since it can have a connection to base control all around the city where the 3G radiation pattern is in range. A smartphone will be mounted on the quadcopter for providing a 4G internet connection. The internet connection on the quadcopter can be used for several things such as providing the GPS location, video stream, and controlling the quadcopter maneuvers.

3.4 Live streaming video vision by using UDP

Video streaming of quadcopter's vision is possible by using the on-board android smartphone. An app it to be designed in such a way that it streams video back to the base station by use of UDP. This is possible through 4G connectivity since use of WiFi for such long range is not advisable.

A video stream is acceptable if there is any fault of packages lost in the way. It is because if during the video streaming the packages get lost due to the router being overloaded for instance then the video streaming can still be resumed with a reduced quality.

The TCP/IP protocol would force the video stream to wait until the dropped packages are found to resume processing newer packages. The point that is usually misunderstood is that when few packages are lost it doesnt mean a whole frame is lost but rather few pixels from a certain frame are lost and hence waiting for these lost pixels to come back is a waste since the video stream has already moved on to new frames. However TCP could have been a good option if the case was about recorded video streaming since when a package is lost the system pauses the frame until the all the data for the next frame has been collected in the buffer giving the term buffering.

Therefore, we decided to use UDP for our live video streaming.

3.5 Smartphone to Controller Board USB Connection

The receiver in our schematics is going to be a smartphone which will be using 4G internet to receive the data. The smartphone onboard the quadcopter will act as client while the PC with the user will act as the server to send the controls. The data will then be passed to an Atmega328 chip using USB-Serial FTDI chip that will in turn send the flight data to controller board. The application will send byte array through serial to Atmega328 Chip that will contain the flight values such as aileron value to control the roll, elevation value to control the elevation angle with horizontal, rudder value to control the yaw movement, and thrust value to control lift force. [**?**]

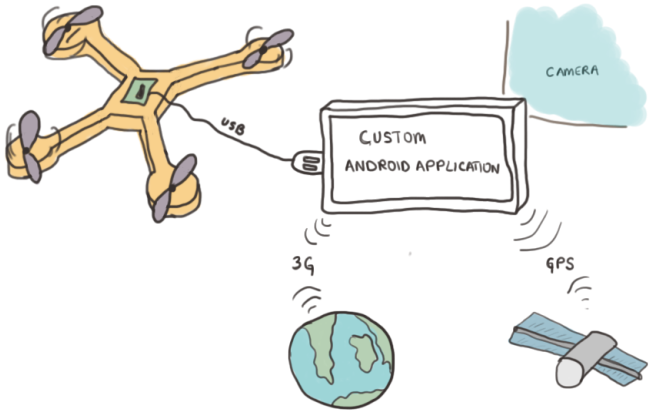


Figure 8: A custom designed and programmed AtMega 328 chip will act as a intermediate connector to decode the signals coming from the android smartphone through USB and send appropriate commands to the control board.

4 Software Development Progress

4.1 Smartphone to AtMega 328 communication protocol development

Since our approach of controlling the quadcopter using the smartphone is new and little or no documentation is present on how to send the data through Android to quadcopter, we had to develop our own network communications protocol. The figure below gives and overview of the OBFlightPacket network protocol we developed.

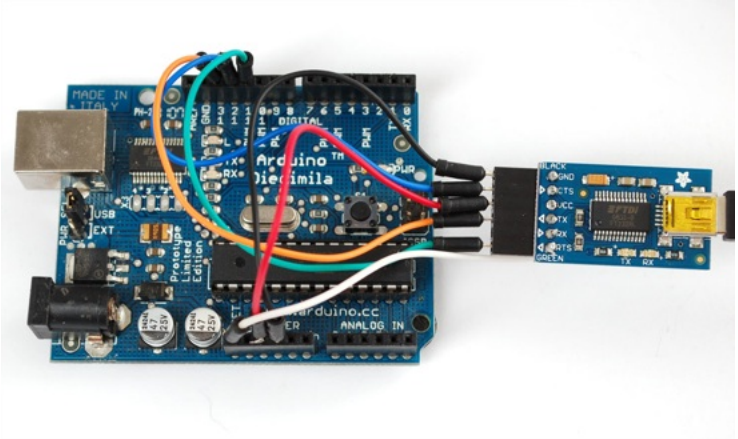


Figure 9: The figure shows how an AtMega 328 is connected to the FTDI USB-Serial converter chip. [2]

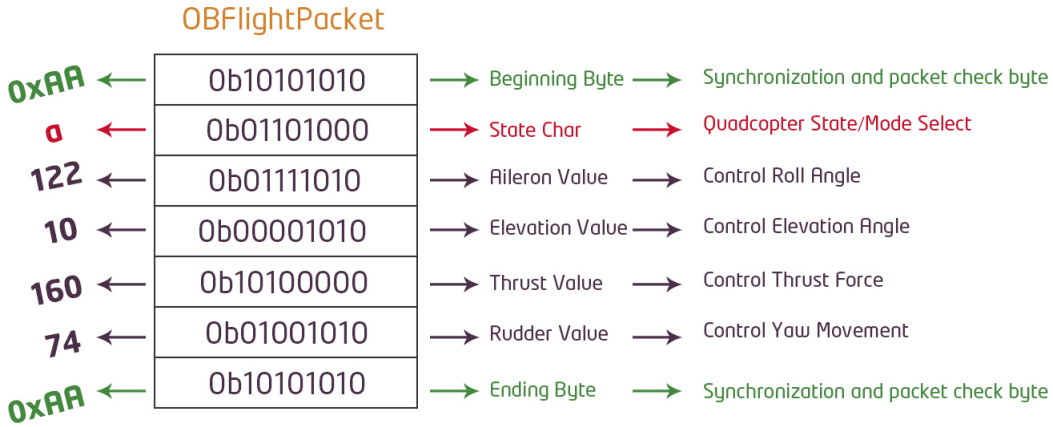


Figure 10: The figure explains the network protocol which was designed by us for sending flight data from Android USB to micro-controller.

OBFlightPacket Network Protocol Once the communication link is established, the android starts sending the OBFlightPacket. The OBFlightPacket is nothing but a byte array of size 7. As the byte is composed of 8 bits, there are $2^8 = 256$ possibilities. So if we convert from byte to unsigned integer, the value can vary from 0 - 255. If we convert the byte to ASCII character (American Standard Code for Information Interchange), each of these possibilities (symbols) can represent all charecters on keyboard and some special characters too.

Now the micro-controller AtMega 328 can communicate with the smartphone using the USB connection. The control data can be transferred back and forth between the smartphone and AtMega. There is a library in Android to communicate with the smartphone's USB port called USBManager.

We used this library in order to send the byte array to the USB port. A USB-serial converter chip called FTDI is connected to the smartphone's USB port and the FTDI chip is then connected

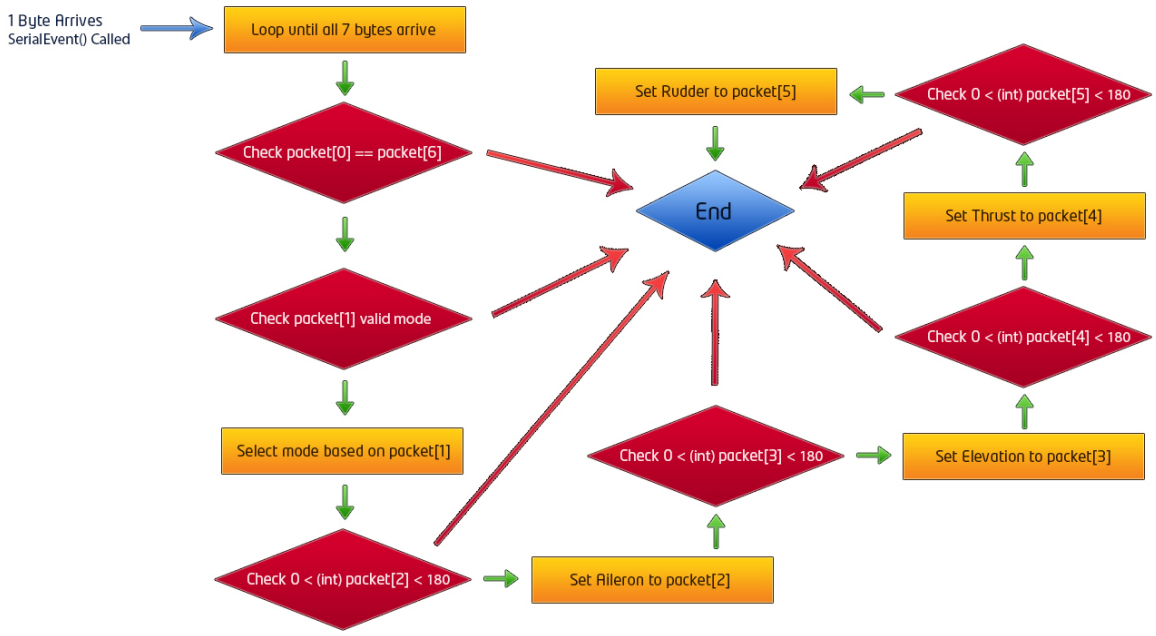


Figure 11: The figure which was developed in Photoshop explains the algorithm to unpack the OBFlightPacket. There are several checking stages and even if one checking results in false, the packet is rejected.

to XBee.

Figure 12: The figure shows one of the ways the smartphone can communicate with the AtMega328. The XBee module is attached using a USB cable to the smartphone and the XBee explorer which has FTDI chip on it, takes care of the conversion of data signals from USB to Serial.

4.2 Leap Motion frame decoding and GUI binding

A leap motion device was bought and the API for grabbing frame data was used to develop a user-friendly GUI for visualizing the Pitch, Yaw, Roll angles. In the Application Programming Interface (API), the leap motion provides vectors for each finger positions and access to several other objects such as hands, fingers, palm normal, palm direction, and palm Position.

```
1 private void LeapFrameUpdate_Tick(object sender, EventArgs e)
{
3           // Get the current frame object from the Leap Motion Class
            currentHandFrame = controller.Frame();
```

```
            // Extracting valuable information from fram and displaying on TextBoxes
            frameIdBox.Text = currentHandFrame.Id.ToString();
            timeStampBox.Text = currentHandFrame.Timestamp.ToString();
            handsBox.Text = currentHandFrame.Hands.Count.ToString();
            fingersBox.Text = currentHandFrame.Fingers.Count.ToString();
            guestureBox.Text = currentHandFrame.Gestures().Count.ToString();

            // Do this is the current frame contains hands
            if (!currentHandFrame.Hands.IsEmpty)
            {
                // Get the first hand
                Hand hand = currentHandFrame.Hands[0];

                // Check if the hand has any fingers
                FingerList fingers = hand.Fingers;
                if (!fingers.IsEmpty)
                {
                    // Calculate the hand's average finger tip position
                    Leap.Vector avgPos = Leap.Vector.Zero;
                    foreach (Finger finger in fingers)
                    {
                        avgPos += finger.TipPosition;
                    }
                    avgPos /= fingers.Count;
                }

                // Show the 3D coordinate position of hand in text box
                PositionBox.Text = hand.PalmPosition.ToString();

                // Get the hand height from sensor and check its valid range
                if ((hand.PalmPosition.y < 500) && (hand.PalmPosition.y > 0))
                {
                    Thrust = (int) (((hand.PalmPosition.y / 500.0) * 71.0) + 57.0);
                }

                // Get the hand height from sensor and show in progress bar
                if ((hand.PalmPosition.y < 700) && (hand.PalmPosition.y > 0))
                {
                    positionProgress.Value = (int)hand.PalmPosition.y;
                }

                // Get the hand's normal vector and direction
                Leap.Vector normal = hand.PalmNormal;
                Leap.Vector direction = hand.Direction;

                // Defining a center point for the arrow picture
                System.Drawing.Point p = new System.Drawing.Point();
                p.X = 128;
                p.Y = 128;
                System.Reflection.Assembly thisExe;
                thisExe = System.Reflection.Assembly.GetExecutingAssembly();
                // Loading the arrow png image for nice angle representation
                System.IO.Stream file = thisExe.GetManifestResourceStream("
                    OregoController.Properties.Resources.arr.png");

                float pitchVal = (direction.Pitch * 180.0f / (float)Math.PI);
                // Rotate the arrow according to the pitch value
                pictureBox2.BackgroundImage = RotateImage(global::OregoController.
                    Properties.Resources.arr, p, pitchVal);
```

```
                // Display pitch in textbox
63              pitchBox.Text = pitchVal.ToString();
                // Check the pitch value range and set it
65              if ((pitchVal < 90) && (pitchVal > -90))
                {
67                  Elevation = (int)pitchVal+90;
                }
69
                float rollVal = (normal.Roll * 180.0f / (float)Math.PI);
71              // Rotate the arrow according to the roll value
                pictureBox3.BackgroundImage = RotateImage(global::OregoController.
                    Properties.Resources.arr, p, rollVal);
73              // Display roll in textbox
                rollBox.Text = rollVal.ToString();
75              // Check the roll value range and set it
                if ((rollVal < 90) && (rollVal > -90))
77              {
                    Aileron = (int)rollVal+90;
79              }

81              float yawVal = (direction.Yaw * 180.0f / (float)Math.PI);
                // Rotate the arrow according to the yaw value
83              pictureBox4.BackgroundImage = RotateImage(global::OregoController.
                    Properties.Resources.arr, p, yawVal);
                // Display yaw in textbox
85              yawBox.Text = yawVal.ToString();
                // Check the yaw value range and set it
87              if ((yawVal < 90) && (yawVal > -90))
                {
89                  Rudder = (int)yawVal;
                }
91          }
}
```

In the ground base station, following things can be done. In other words, the features of the ground station are as following:

- Frame is being captured from the leap motion device.
- After extensive image processing by leap motion library on the current frame, the library initializes hands, fingers, and palms objects in the frame parent object and returns frame for programmer to access and use.
- This frame object is accessed by the $C\#$ application and some of the data enclosed is displayed on the screen in forms of text-boxes, labels and pictures.
- The arrow picture is loaded and rotated to the same angle according to the "Pitch" of one the hands detected by the leap motion.
- The arrow picture is loaded and rotated to the same angle according to the "Roll" of one the hands detected by the leap motion.
- The arrow picture is loaded and rotated to the same angle according to the "Yaw" of one the hands detected by the leap motion.

- The hand y position in 3D is extracted from the current frame and displayed in a textbox and a progress bar indicates how high above is the hand approximately from the sensor. The maximum sensitivity we detected using our Leap Motion device was about $70cm$.
- The current frame time-stamp, frame ID., number of hands detected, number of fingers detected, and gesture detected state are extracted from the frame object and displayed using the text-boxes.
- Pitch, Yaw, Rudder, and Thrust values are then converted to byte array according to the Flight protocol which we discussed earlier.
- The GPS way points for a quadcopter can be set and stored in the database.

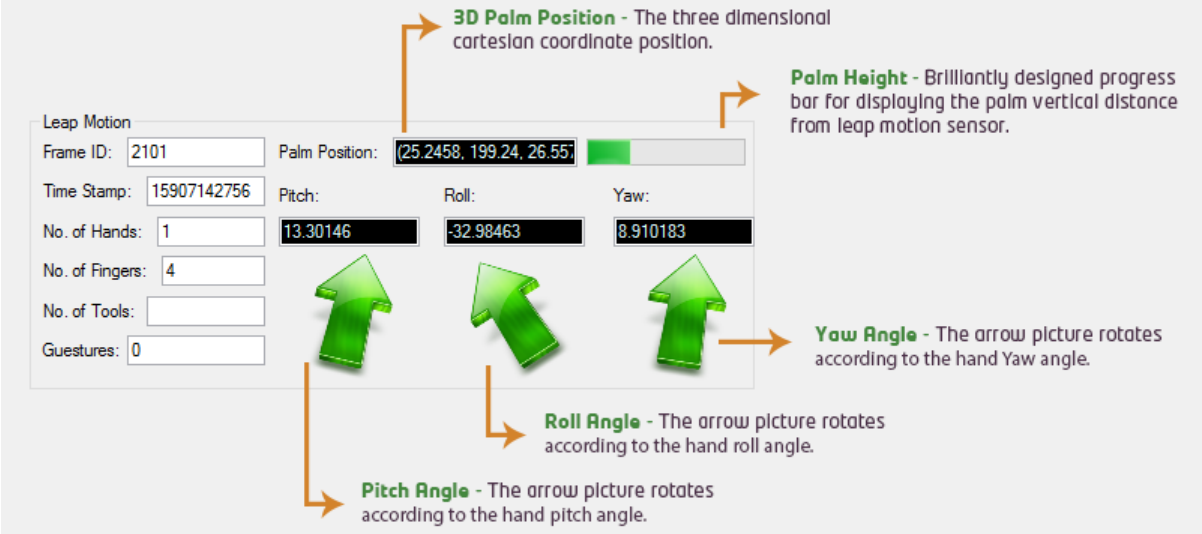


Figure 13: The figure explains a groupbox which is embedded inside the main application and contains the GUI for viewing the Leap Motion data and variables.

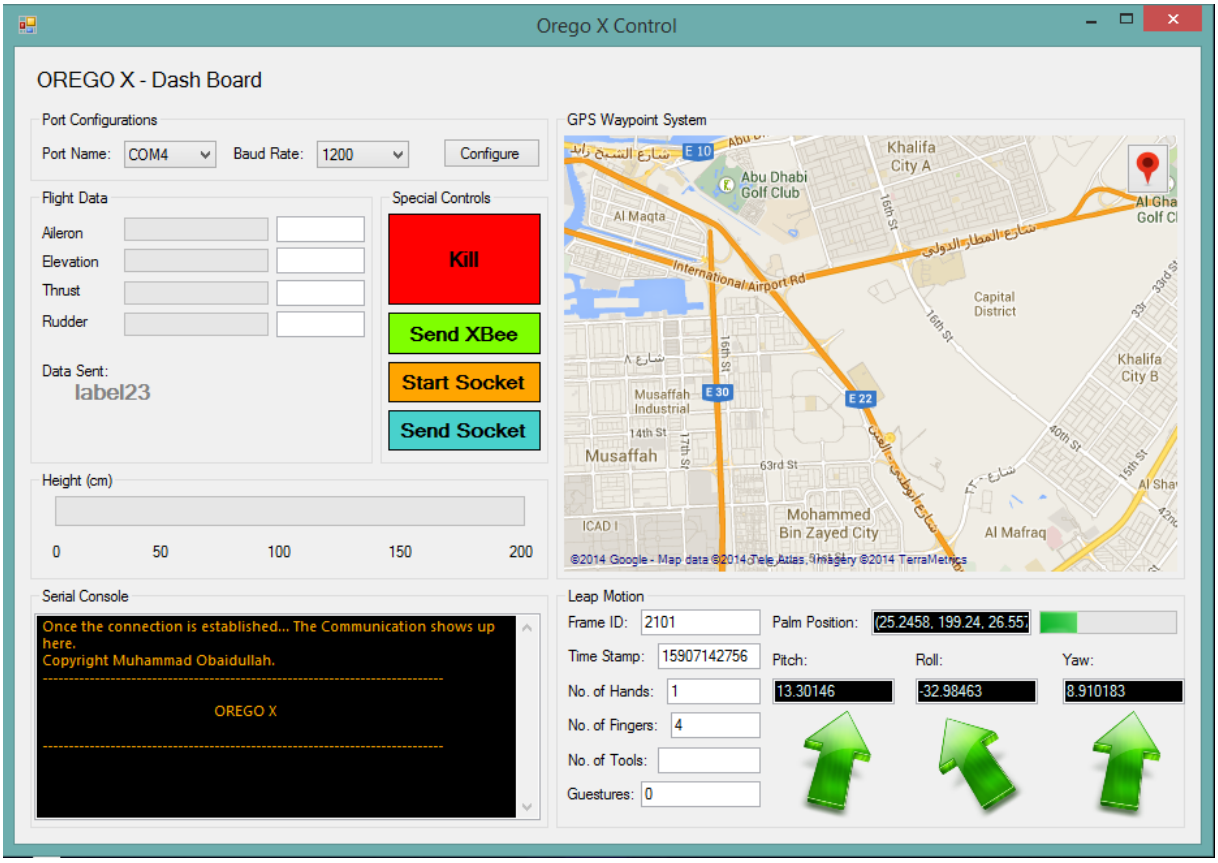


Figure 14: The overall *C#* application main window with all the settings and options. The application allows communication through internet and XBee as well. The commands can be sent manually by entering it into textboxes or directly decoded from hand gestures.

4.3 Google Map for Quadcopter Navigation

4.3.1 Getting the SHA1 Fingerprint For Google Map

To work with Google map we need to use the API; however; since the Google map API is not intended to be open source and only belong to Google in order to be changed the API comes as a part of Google Play Services. It simply means we need permission from the Google Play Service to allow us use their Google Map API.

The process of obtaining the permission is by first letting my app to contact with the Goolge API site. This is made possible by the following steps:

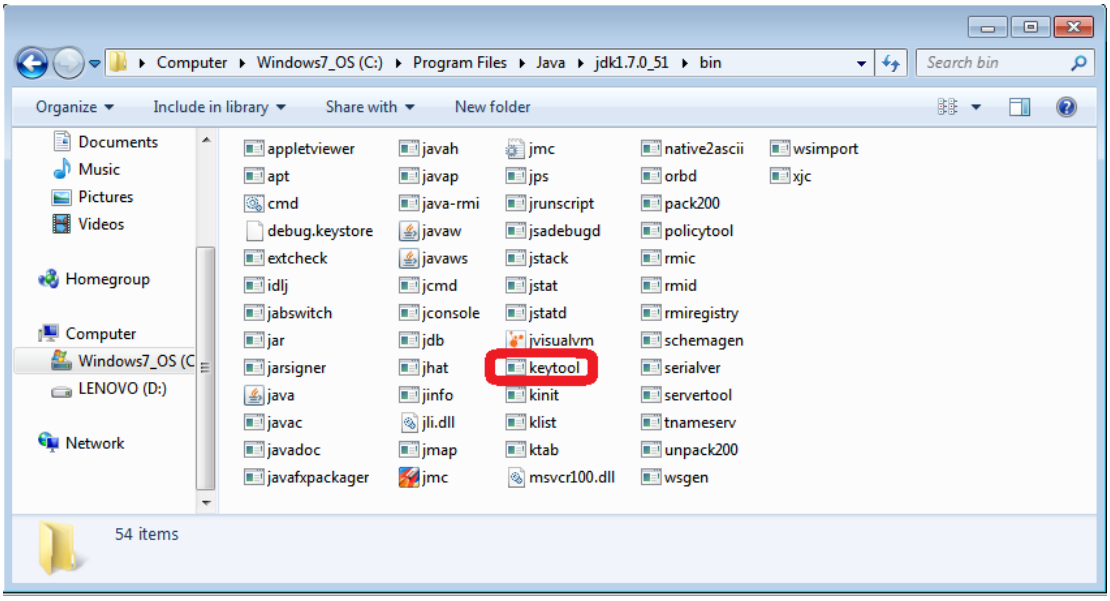
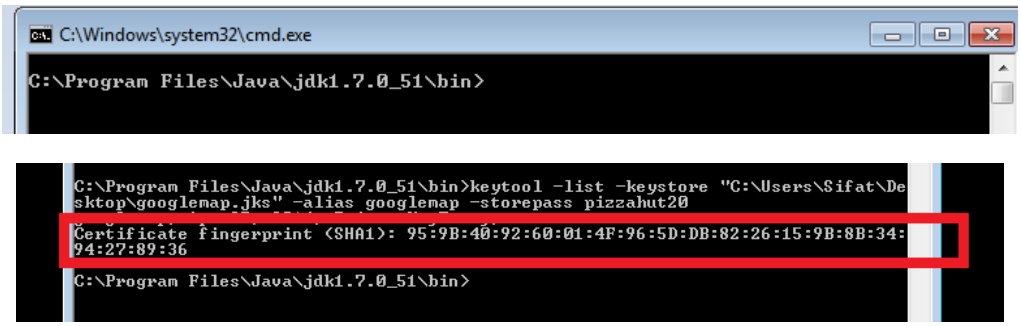


Figure 15: We generate an APK that will have a SHA1 finger-print. This finger-print will be stored in a .jks file format which cannot be opened except with the "keytool.exe' application that comes along when one downloads the Java SDK.

In order to tell keytool to open that .jks file we use cmd as the interface. Below are a series of simple commands that we need to type in order to open keytool application and then tell it to open my .jks file to retrieve the SHA1 finger print

Figure 16: Signed APK was generated and stored the key in the following file Now it was time to retrieve the key that is stored inside the googlemap.jks

Step 1: We entered the path of keytool by typing the following commands
cd $C:\backslash Program Files\backslash Java\backslash jdk1.7.0\_51\backslash bin$
Step 2: Then we retrieved the SHA1 Fingerprint in the following way:



5 Hardware Development Progress

5.1 3D Model Design

A 3D Model of the quadcopter was custom designed to fit our needs. The 3D Model was supposed to have arms of specific lengths to allow the propellers to rotate freely. We designed the whole quadcopter in Google Sketch Up taking care of these parameters. The design also was supposed to have a holder for the mobile phone. We designed a intelligent mechanism for controlling the tilt angle of the mobile. We used 10 inch diameter propellers with 3.8 inch of pitch. The pitch controls the amount of air scoped with each rotation. The diameter controls two things, amount of air the propeller pushes down, and the weight of the propeller. Careful calculations and testings were done to calculate the amount of weight of each motor can lift.

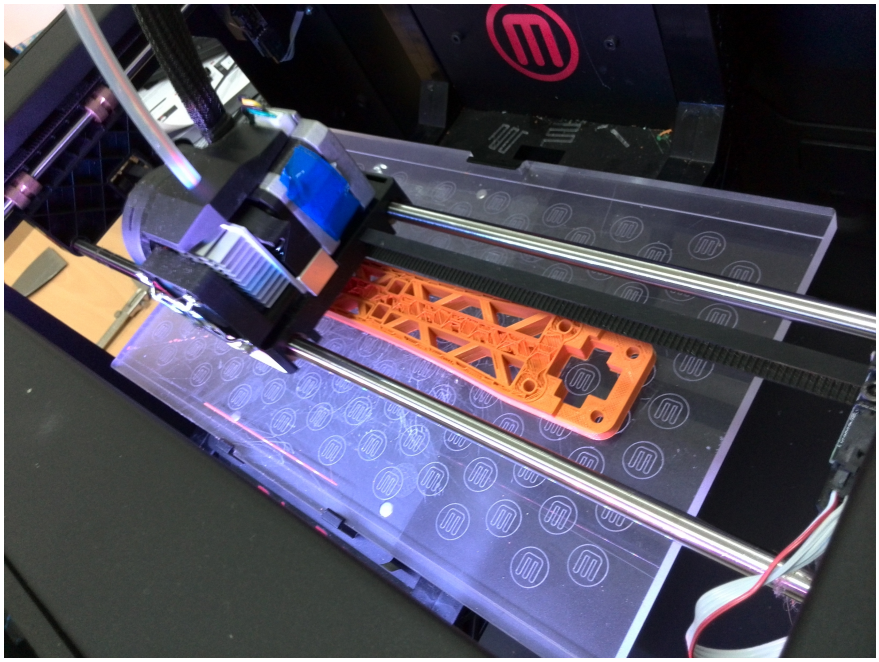


Figure 17: A arm of the quadcopter is being printed by the MakerBot.

Features of the Model:

- Four less infill light 10” motor arms.
- Three center platforms with screw holes to mount the control chip.

- A raft to hold and rotate the mounted android smart-phone.
- Four stands for landing gear.

5.2 Gyroscope Sensor

A quadcopter is highly prone to instability due to irregular distribution of mass or wind.Basic moves of a quadcopter like lifting straight off the ground to complex moves like pitch(right) and roll or yaw cannot be achieved without stabilizing the quadcopter.

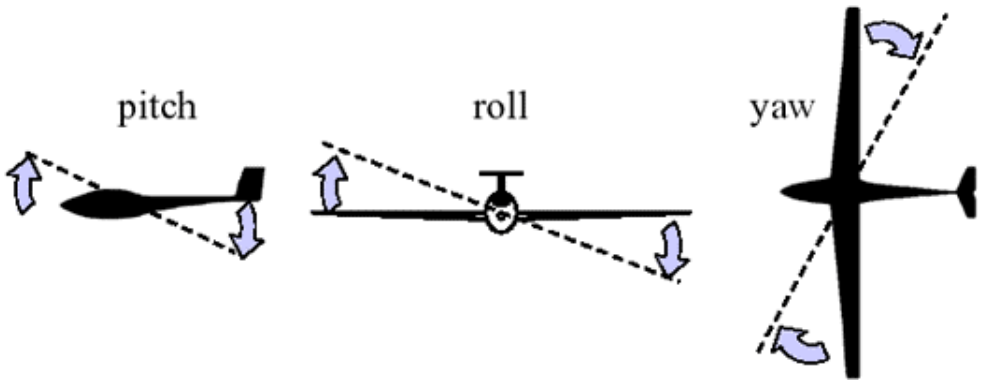


Figure 18: Explanation of the traditional 3 Dimensional rotation movements. These movements were defined for airplanes but as time passed by, these same terminologies were adopter by quadcopters although quadcopters don't bend their wings for roll(aileron).

It has become the industry standard to implement sensors to know the precise measurment of imbalance instantly and exert the proper combination of force to bring it back to balance. These sensors are known as Gyroscope and they use the law of physics to measure this phenomenon.

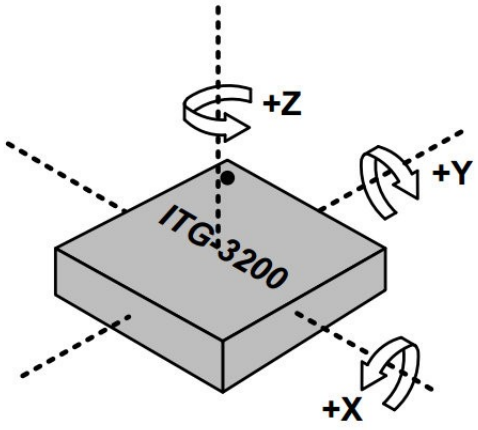


Figure 19: The gyroscope sensor is designed to measure the rotation per second and the angle of rotation of the object its mounted to when it experiences an imbalance.Below is a typical 3-axi gyroscope ITG3200. [6]

Hence, when the object tilts a little to the right(roll), the gyroscope on the y-axis will record a reading while the other three will not record any reading, or when the device tilts up(pitch) the sensor in x-axis will record a value while the rest wont record any value and finally if the device rotate in the horizontal plane the sensor in z plane will measure a current due to angular rotation around the z-axis

The working of the gyroscope is as fascinating as it is clever. There is a small mass inside each gyroscope on a particular axis which is held by very small springs. When there is a rotation, this will cause a centripetal force to form whose direction will depend on the direction of rotation. For instance; if the gyroscope is rotating right there will be a force that is exerted toward the center while it the gyroscope is rotating outward there will be a force that opposite to the center.

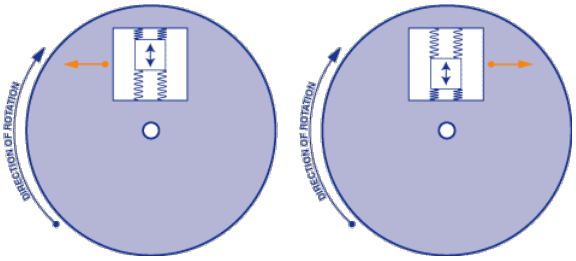


Figure 20: These forces are exerts the spring and this movement generates very low-current electrical signal that is amplified and later read by the micro-controller. [6]

5.3 An unexpected tragedy

After designing the whole body using the lab's 3D printer, the quadcopter's body melted and twisted away one day due to sun heat. This gave us a strong lesson that the plastic used to manufacture the quadcopter body is very sensitive to the high temperatures. Since UAE's temperature is usually very high in major part of the year, the 3D plastic cannot be used for building the quadcopter.

5.3.1 Temperature Sensitivity of PLA plastic used in 3D printer

The plastic which is used in the 3D printer is Polylactic Acid thermoplastic aliphatic polyester. It is usually derived from renewable resources such as corn starch. It has a melting point of 150-160 degrees centigrade. The temperature in UAE reaches 45 degree centigrade in summers and even with a small amount of insulation, this temperature can reach high enough to just twist or deform the plastic. The choice of PLA plastic for quadcopter body is really bad when it comes to temperature sensitivity.

5.3.2 Strength of PLA plastic

In the first two-three attempts, the quadcopter crashed several times and broke the arms within milliseconds after coming in contact with the ground. This was also a big issue as we want the final product to have strength enough to withstand huge jerks and shocks in case the quadcopter falls down from huge height.

Mechanical	Nominal Value Unit	Test Method
Tensile Modulus		
73°F	293000 to 514000 psi	ASTM D638
73°F	96500 to 516000 psi	ISO 527-2
Tensile Strength		
Yield, 73°F	8840 to 9500 psi	ASTM D638
Yield, 73°F	2860 to 10400 psi	ISO 527-2
Break, 73°F	7080 to 8150 psi	ASTM D638
Break, 73°F	2320 to 10200 psi	ISO 527-2
73°F	6930 to 10000 psi	ASTM D638
Tensile Elongation		
Yield, 73°F	9,8 to 10 %	ASTM D638
Yield, 73°F	1,0 to 8,5 %	ISO 527-2
Break, 73°F	1,5 to 10 %	ASTM D638
Break, 73°F	1,0 to 7,2 %	ISO 527-2
Flexural Modulus		
73°F	347000 to 684000 psi	ASTM D790
73°F	319000 to 1,38E+6 psi	ISO 178
Flexural Strength		
73°F	6950 to 16000 psi	ASTM D790
73°F	5000 to 16100 psi	ISO 178

Figure 21: The table shows strength of the plastic used for 3D printing in our lab according to different standards. [3]

5.4 New Body Design

After getting the lesson, we changed the body from PLA 3D printed plastic to Carbon Fiber body. The body was bought as a kit from HobbyKing named " Turnigy Talon Carbon Fiber Quadcopter Frame". Link: `http://hobbyking.com/hobbyking/store/__22397__Turnigy_Talon_Carbon_Fiber_Quadcopter_Frame.html`

5.5 Circuit Design

5.5.1 Circuit Design Version 1

Initially when we started our research on Quadcopter controller board, we found that we needed a PCB to find out what signals exactly are required by the quadcopter in order for it work perfectly. Additionally, connecting and disconnecting all the wires on the breadboard was very hectic and time consuming. For this purpose we created the version 1 of the PCB. It was the pre-final version of the PCB. This PCB's job is to transmit the appropriate Aileron, Rudder, Thrust and Elevation PWM signals to the controller board so that it can further perform PID control on brush-less motors. Essentially, this PCB is a bridge between the controller board and the smartphone. The following were the features of the first circuit.

- Servo connectors for four ESCs.
- Servo connector for mini servo which will be used for changing the mobile angle.
- Reset push button.
- Tilt Compensated Compass output pins.
- SPI programming header.
- 16 MHz crystal with 22pF capacitors for providing clock signal to AtMega328.

- Ultrasonic height sensor connection.
- FTDI connection pin.
- 3.3V Regulator for stepping down he voltage from 5V to 3.3V for the XBee.
- 3 indicator LEDs of different colors.
- XBee connection support.
- Bluetooth module connection support using the FTDI pins.

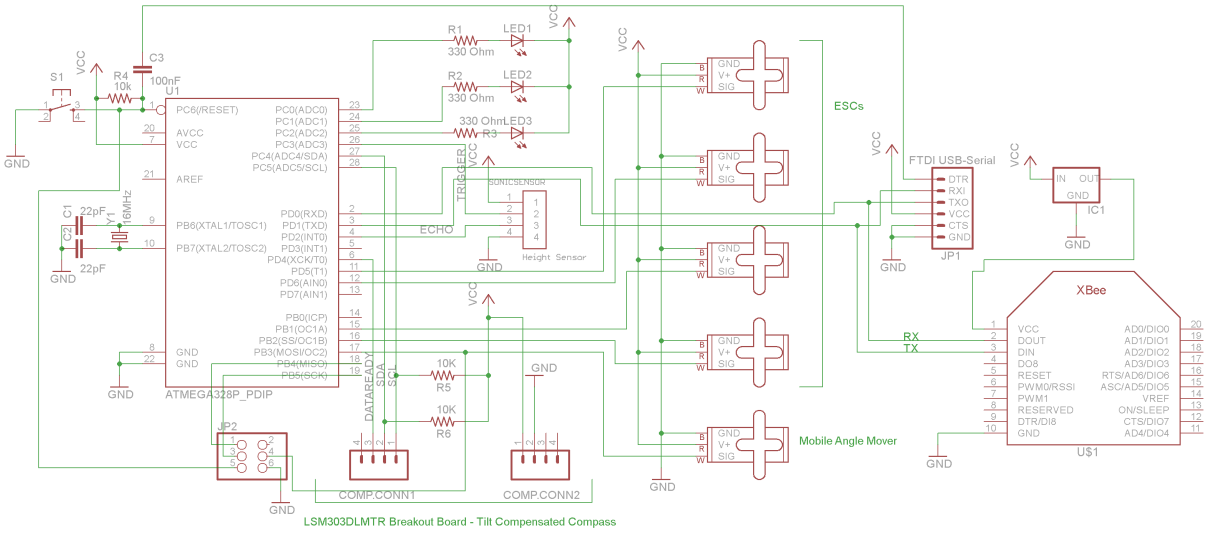


Figure 22: A circuit diagram of our prototype board design version 1.

5.5.2 Circuit Design Version 2

We quickly realized that several improvements can be done to the initial circuit to make the quadcopter have more and more features. The following are the additional features of the new PCB.

- IR Receiver for receiving Infra red signals. Basically this includes TV, CD player and other IR transmitters. The IR receiver already has a electronic circuit to amplify the IR signal and provide a signal when a change in IR level is detected. Also the Sensor already has the background noise removal circuit. SO we can just connect the infra-red sensor directly to the interrupt pin to read the bits transmitted and change the states of the quadcopter. This IR Receiver will be used
- 5V Barrel Jack connector for providing 5V to the circuit.
- USB A jack for supplying power.
- USB B jack for supplying power.

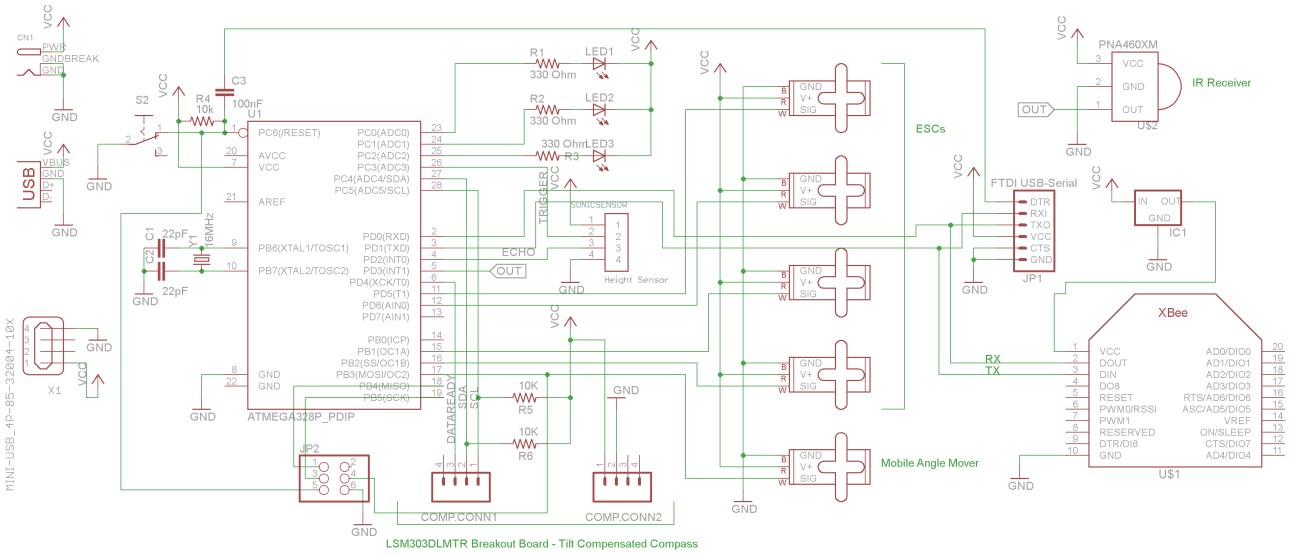


Figure 23: A circuit diagram of our prototype board design version 2.

5.6 PCB Design

The PCB was designed to be compact and at the same time to contain all the essential components for communication. There are two ways a communication link can be made with the on-board ATmega 328 on our PCB. The first method is by using XBee and the second method is by using the USB Port.

Features of the Prototype PCB

- 4 servo connectors for connecting to the Electronic Speed Controllers.
- 8 Pin connector for connecting a tilt compensated compass.
- 6 pin connector for FTDI chip to convert USB data signals to serial data.
- 4 pin connector for connecting a height measurement sensor.
- A 3 pin connector to connect the mobile tilting sensor.
- A SPI JTAG programming header.
- XBee connector for wireless communication.
- 3 LEDs to indicate the status of the AtMega microcontroller.

In the above figure, the prototype board is shown where we have used XBee to communicate with the computer and receive the serial data to control the motor speed. This is just for testing purposes and in the final design we will use the data coming from the android to determine the speed of the motors. Furthermore, the final design will use the XBee as a kill switch to completely turned off the quadcopter in case it reaches a forbidden area or any similar situation.

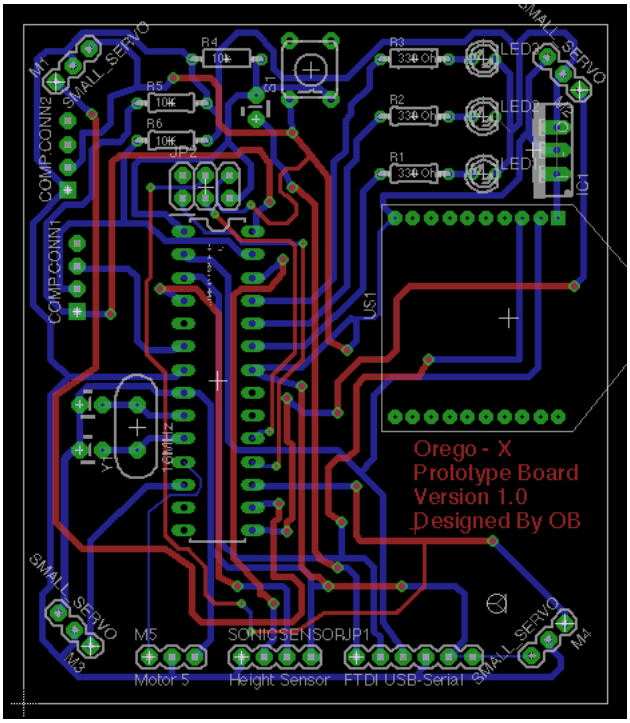


Figure 24: A x-ray vision of our prototype board design version 1.

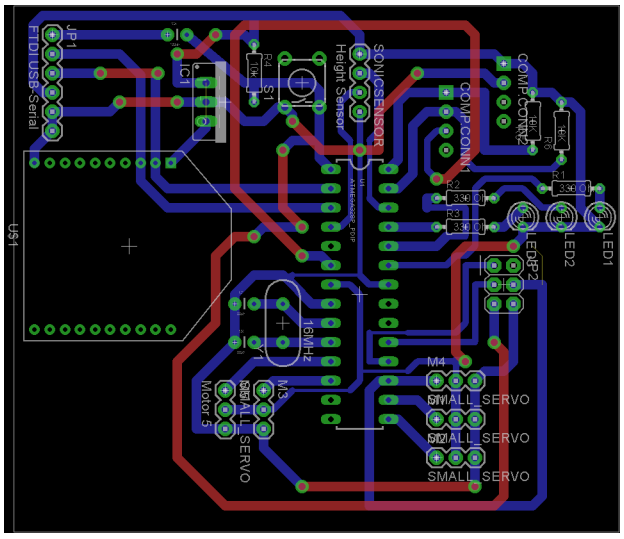


Figure 25: A x-ray vision of our prototype board design version 2.

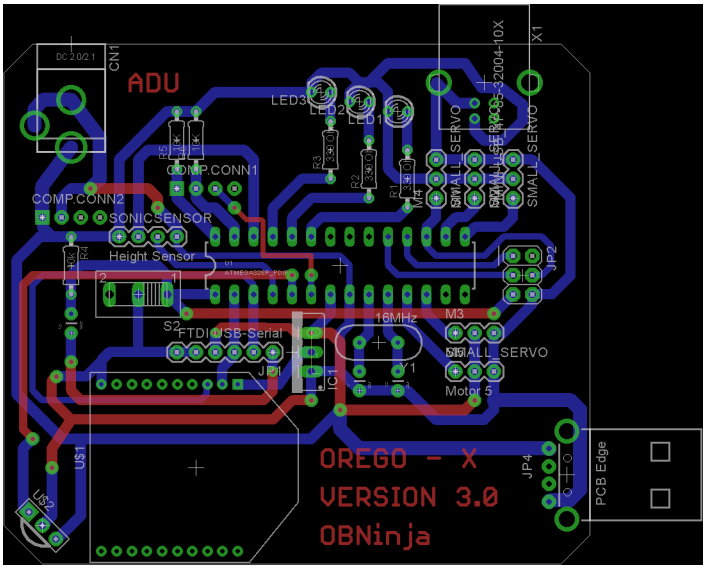


Figure 26: A x-ray vision of our prototype board design version 3.

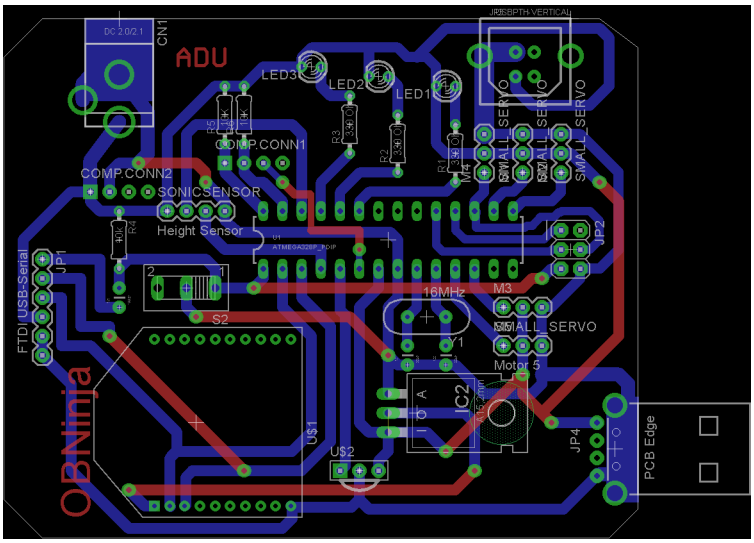


Figure 27: A x-ray vision of our final board design version 4.

5.7 Hand Gesture Models

In order to control the quadcopter using a leap motion device, a model of the hand gestures should be made in order to approximate the finger positions in three dimensional space.

Leap Motion provides a library to use for creating hand objects and approximates the finger positions in the 3 dimensional space. According to these positions of the fingers, we can code the Leap Motion to generate http requests to the android on the quadcopter through internet.

We will program a java application to detect the hand gestures and then according to the gesture input, a particular http request will be made to the android. Eclipse IDE will be used to program the desktop application which will provide the internet connection to the quadcopter.

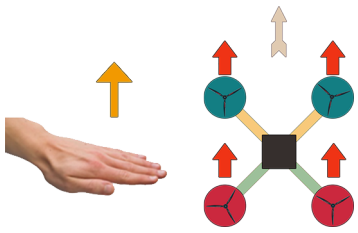


Figure 28: Moving the hand upwards in the z-axis in front of the Leap motion should provide us with a +ve change in z value.

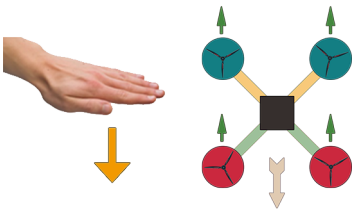


Figure 29: Moving the hand downwards in the z-axis in front of the Leap motion should provide us with a -ve change in z value.

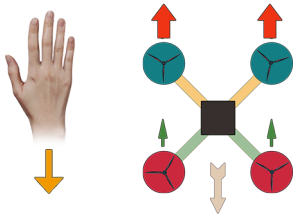


Figure 30: Moving the hand towards the body can give us -ve value of movement in y-axis.

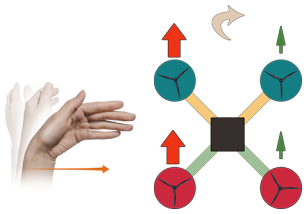


Figure 31: A +ve value in x-axis can be obtained when the user moves the hand to the right.

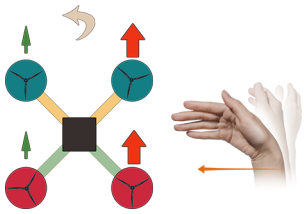


Figure 32: A -ve value in x-axis can be obtained when the user moves the hand to the left.

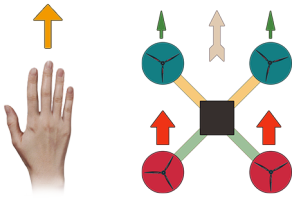


Figure 33: Moving the hand away from the body can give us +ve value of movement in y-axis.

5.8 Kalman Filter

There is a module in the controller board that processes the huge amount of samples the microprocessors take from the sensors. To put that into perspective lets say one have a micro-controller that has a clock speed of 15 MHz, that means the micro-controller will be taking 15 million samples from the sensor every second. All this amount of data is meaningless if the right data is not processed from this, Kalman Filter is a sequence of steps that will initially remove the sampled data that are purely noise and then average out the important data to pick the most appropriate one for input.

The leap motion captures about 200 frames per second or more depending upon the computer processing power. This is huge data speed and sudden errors can cause the system to In our project, we will implement a Kalman Filter on the hand gesture data.

6 Testing Results

6.1 Custom 3D Designed Body

Successful printing of the body was achieved but arms had to be manually drilled for the holes so that the motors fit precisely. The body was assembled using the screws with washers to prevent the plastic to breaking down. The hols for the circuit board to mount were made and the place for the mobile angle mover servo was also cut out.

In the website from Makerbot named Thingiverse (`www.thingiverse.com`), there are several 3D models designed for several sized quadcopters. Although these models are similar to what our final product required, these models need to be modified in several places to fit out needs and requirements.

6.1.1 Version 1 - PLA Plastic, Unchanged print

We printed the 3D quadcopter model as is from the thingiverse website without modifying any part of the quadcopter. This was done in order to make sure that the existing printer in the lab can print the model, all the parts fit together, and allow additional electronic components to be mounted onto the central body.

We found that all the parts indeed fit together but a major increase in the height of the arms from ground was needed. Also the motors we had did not fit on top of the arms perfectly. An increase in the diameter of motor mounts was also needed.

6.1.2 Version 2 - PLA Plastic, Modified print

We increased the arm motor mount diameters and increased the height of the landing pads and printed the parts using the 3D printer.

We found that all the parts indeed fit together but a tragedy occured when we left the model in the car for two days. Under the heat of the sun and no air flow, the quadcopter body melted and got twisted.

6.1.3 Version 3 - Carbon Fiber Body

We bought the carbon fiber body from Hobby King and made sure that the dimensions matched rest of the parts. The carbon fiber is composite material which is many times stronger than steel and quite light in weight.

6.2 Leap Motion Library for Gesture Control

There is a huge documentation online for programming the leap motion controller in Java. The gesture detection can be achieved using a Controller Object which can return the hand tracking data by using frame object within this class. [5] For example in the following code:

```
SampleListener listener = new SampleListener();
Controller controller = new Controller();
// Have the sample listener receive events from the controller
controller.addListener(listener);
Frame frame = controller.frame();
System.out.println("Frame id: " + frame.id()
                   + ", timestamp: " + frame.timestamp()
                   + ", hands: " + frame.hands().count()
                   + ", fingers: " + frame.fingers().count()
                   + ", tools: " + frame.tools().count());
```

6.3 Payload Capability calculation using MATLAB

A MATLAB code was written according to the propeller efficiency, power of the motors and the propeller size to calculate the payload the quadcopter is capable of carrying.

```
% Propeller hover efficiency
eta = 0.75;
% Power of the motor Max. for our motor is 125
Power = 110;
% Propeller Radius in meters diameter = 10 inches = 0.2794
R = 0.2540;
% Usual Air Desnity kg/m^3
rho = 1.22;
Thrust = ((eta+Power)^2 * 2 * pi * R^2 * rho)^(1/3);
disp('Thrust in Newtons:');
Thrust
disp('Weight Liftable by one motor in Kg:');
Weight = Thrust/9.80665002864;
Weight
disp('Weight Liftable by all four motors in Kg:');
Weight = (Thrust/9.80665002864)*4;
Weight
```

The following was the result of the simulation:

```
>> MotorCalculation
Thrust in Newtons:

Thrust =

   18.2375

Weight Liftable by one motor in Kg:

Weight =

    1.8597

Weight Liftable by all four motors in Kg:

Weight =

    7.4388
```

Figure 34: Moving the hand away from the body can give us +ve value of movement in y-axis.

6.4 Weight Calculation of the Quadcopter.

A weight load estimation excluding the extra payload was tabulated:

Component	**Type**	**Qty.**	**Unit**	**Total**
ESC	30Amps	4	34.92g	139.7g
Motor & propellers	Super Tigre	4	59.45g	237.8g
Plastic Frame	3D Custom	1	248g	248g
Screws	Metallic	14	9.0g	125.0g
Servo	Tiny	1	11.3g	11.3g
Control Brd. & wires	Hobby King	1	21.0g	21.0g
AtMega Brd.	Custom	1	29.9g	29.9g
Android Smartphone	Huawei P6	1	120g	120g
11.V Battery	ZOP	1	369.7g	369.7g
Ultrasonic Sensor	N/A	1	10.2g	10.2g
Wiring	N/A	1	20g	20g
Total				1332.6g

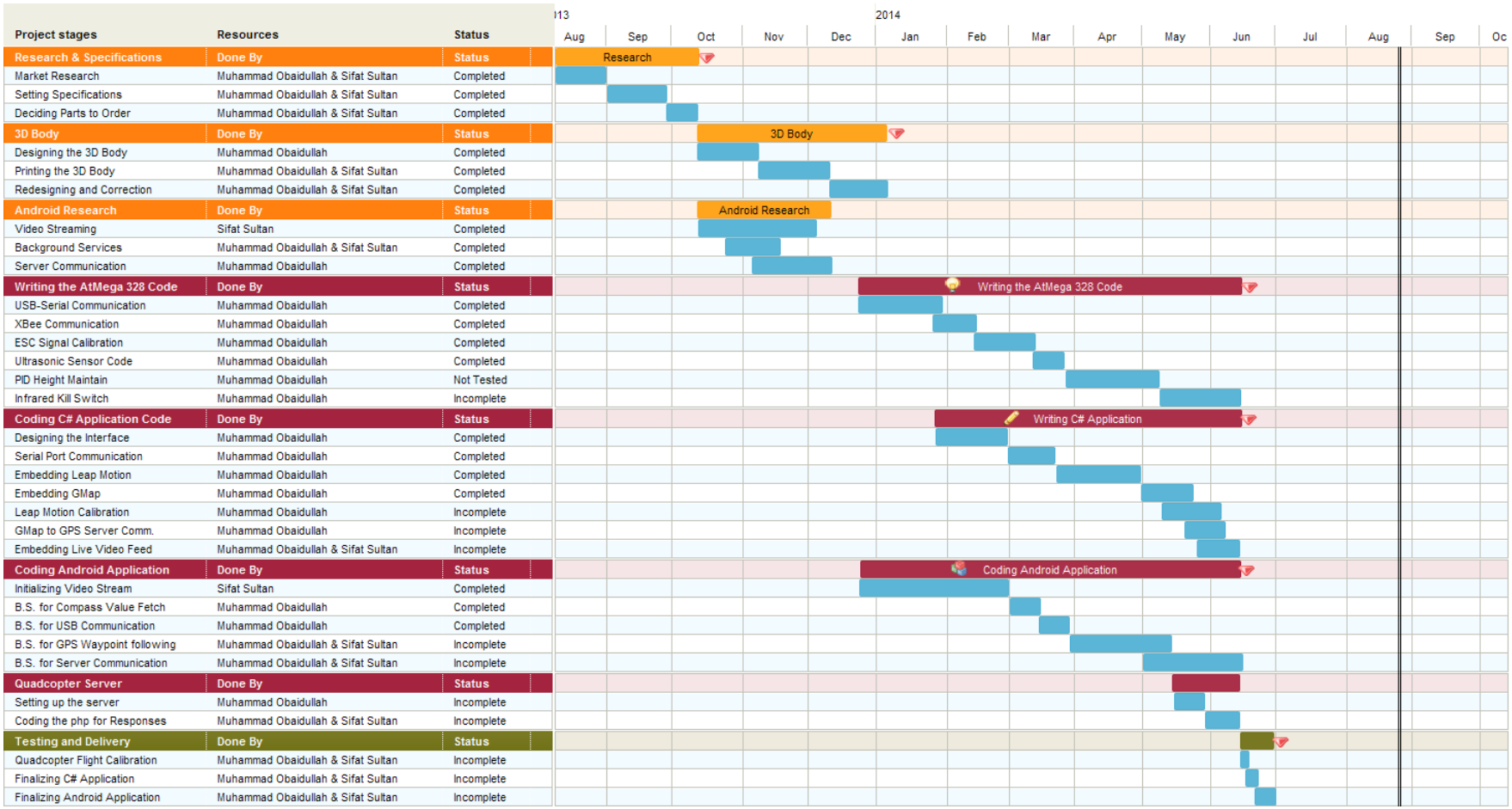
The weight calculation showed that the quadcopter parts were no heavier than 1332.6 grams. in order for a quadcopter to have nice thrust, typically the thrust produced by the motors should be twice the weight. Our first test runs of the motors show that the quadcopter is easily capable of lifting these weights.

7 Updated Cost Table

S.No.	Part Name	Quantity	Total Cost (AED)
1.	ESC – 30 Amps	4	240.00
2.	30C LiPo Battery – 5000mAh	1	230.00
3.	Propeller Pusher Type – 10 x 4.7	2	43.20
4.	Propeller Slo Flyer Type – 10 x 4.7	2	41.70
5.	Propeller Adapter	4	70.60
6.	Screws	30	25.00
7.	ESC – 60 Amps	1	220.16
8.	Hobby King Control Board V3.0	1	66.02
9.	Brushless Motor SUPG8030	4	396.21
10.	Prop Drive Series 2830-1100KV Motor	4	237.72
11.	Prop Drive 28 Series Accessory Pack	4	27.76
12.	Tilt Compensated Compass Breakout	1	89.02
13.	XBee Explorer Dongle	1	91.64
14.	PCB	1	25.00
15.	Electrical Components	1	52.00
16	Talon Carbon Fiber Quadcopter Frame	1	110.15
	Total		**1966.18**

Figure 35: The table shows all the parts which were bought from various sources and countries including parts which were bought and brought personally from canada.

8 Gantt Chart


13
2014
Project stages
Resources
Status
Aug
Sep
Oct
Nov
Dec
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oc
Research & Specifications
Done By
Status
Research
Market Research
Muhammad Obaidullah & Sifat Sultan
Completed
Setting Specifications
Muhammad Obaidullah & Sifat Sultan
Completed
Deciding Parts to Order
Muhammad Obaidullah & Sifat Sultan
Completed
3D Body
Done By
Status
3D Body
Designing the 3D Body
Muhammad Obaidullah
Completed
Printing the 3D Body
Muhammad Obaidullah & Sifat Sultan
Completed
Redesigning and Correction
Muhammad Obaidullah & Sifat Sultan
Completed
Android Research
Done By
Status
Android Research
Video Streaming
Sifat Sultan
Completed
Background Services
Muhammad Obaidullah & Sifat Sultan
Completed
Server Communication
Muhammad Obaidullah
Completed
Writing the AtMega 328 Code
Done By
Status
Writing the AtMega 328 Code
USB-Serial Communication
Muhammad Obaidullah
Completed
XBee Communication
Muhammad Obaidullah
Completed
ESC Signal Calibration
Muhammad Obaidullah
Completed
Ultrasonic Sensor Code
Muhammad Obaidullah
Completed
PID Height Maintain
Muhammad Obaidullah
Not Tested
Infrared Kill Switch
Muhammad Obaidullah
Incomplete
Coding C# Application Code
Done By
Status
Writing C# Application
Designing the Interface
Muhammad Obaidullah
Completed
Serial Port Communication
Muhammad Obaidullah
Completed
Embedding Leap Motion
Muhammad Obaidullah
Completed
Embedding GMap
Muhammad Obaidullah
Completed
Leap Motion Calibration
Muhammad Obaidullah
Incomplete
GMap to GPS Server Comm.
Muhammad Obaidullah
Incomplete
Embedding Live Video Feed
Muhammad Obaidullah & Sifat Sultan
Incomplete
Coding Android Application
Done By
Status
Coding Android Application
Initializing Video Stream
Sifat Sultan
Completed
B.S. for Compass Value Fetch
Muhammad Obaidullah
Completed
B.S. for USB Communication
Muhammad Obaidullah
Completed
B.S. for GPS Waypoint following
Muhammad Obaidullah & Sifat Sultan
Incomplete
B.S. for Server Communication
Muhammad Obaidullah & Sifat Sultan
Incomplete
Quadcopter Server
Done By
Status
Setting up the server
Muhammad Obaidullah
Incomplete
Coding the php for Responses
Muhammad Obaidullah & Sifat Sultan
Incomplete
Testing and Delivery
Done By
Status
Quadcopter Flight Calibration
Muhammad Obaidullah & Sifat Sultan
Incomplete
Finalizing C# Application
Muhammad Obaidullah & Sifat Sultan
Incomplete
Finalizing Android Application
Muhammad Obaidullah & Sifat Sultan
Incomplete

9 Tasks

1

Task Name: Modifying 3D body design
Accomplished By: Muhammad Obaidullah
Starting Date: 4 December 2013
Completion Date: 26 December 2013
Obstacles Faced:

- The Sketch Up did not allow modification of the motor mount unless the landing pad was removed. Used an older version of the Sketch Up file provided online and printed the leg separately.

Percentage Completion: 100%
Output: The quadcopter arms and body is now completely designed in Google Sketch Up and can be exported as .stl format for printing using the Makerbot.

2

Task Name: Printing the 3D model and painting it
Accomplished By: Muhammad Obaidullah
Starting Date: 30 December 2013
Completion Date: 26 March 2014
Obstacles Faced:

- Several times, the electricity of the lab went and the model had to be printed again.
- After repeated printing and experimenting, the size of the holes and arms was modified for perfect fitting.
- Several test flights resulted in broken parts of the quadcopter, these parts were printed again.

Percentage Completion: 100%
Output: PLA plastic body is completely printed using the lab's 3D printer and painted using spray paint.

3

Task Name: Reverse engineering the HobbyKing flight control board
Accomplished By: Muhammad Obaidullah
Starting Date: 28 January 2014
Completion Date: 12 March 2014
Obstacles Faced:

- Surface mount resistor values were very hard to find out because of small size.
- The signals required by the flight control board were found and confirmed using the oscilloscope.

Percentage Completion: 100%
Output: We know exactly what signals are coming out from the HobbyKing flight control board and what input signals (Elevation, Aileron, Thrust, Rudder) signals are required as inputs. It was also of utmost importance to find out the V_{p-p} and duty cycle of the output PWM which is fed into the ESCs (Electronic Speed Controllers).

4

Task Name: AtMega 328 to Android Protocol Development
Accomplished By: Muhammad Obaidullah
Starting Date: 12 March 2014
Completion Date: 26 March 2014
Obstacles Faced:

- Serial communication gives low error rate under slow bit rate and XBee performs well close to 1200 bps serial data rate. Therefore the baud rate was reduced to 1200 bps. And appropriate delay was added in the Serial data read loop to wait for the next byte to arrive.

Percentage Completion: 100%
Output: Using this protocol, the quadcopter can be controlled completely. The android smart-phone can send flight data and control the global positioning of the quadcopter.

5

Task Name: Configuring XBee for serial communication
Accomplished By: Muhammad Obaidullah
Starting Date: 26 March 2014
Completion Date: 27 March 2014
Obstacles Faced:

- New version of XCTU does not recognize the XBees bought. Downloaded older version of XCTU and configured.
- The baud rate was reduced to 1200 bps to ensure low bit error rate. This rate was chosen after many tests using the oscilloscope channel 1 connected to the RX pin of XBee to detect whether it received every byte sent at different data speeds.

Percentage Completion: 100%
Output: The quadcopter can now be controlled by using either XCTU, C# windows application, or Android application. The following link was used: `http://www.science.smith.edu/~jcardell/Courses/EGR328/Readings/XbeeGettingStarted.pdf`

6

Task Name: Capture video using Camera object
Accomplished By: : Sifat Sultan
Starting Date: 1 January 2014
Completion Date: 5 June 2014
Obstacles Faced:

- Our quad copter is required to take a video stream of the surrounding and using any USB camera that are found in the market to carry out this task is bad choice as with every addition of weight the battery life of the quad copter is bound to reduce.

Percentage Completion: 80%
Output: Since the quad copter comes with smartphone which is intended to do the task of providing with 4g communication service, it only makes sense to use the other features that comes along with a standard android smartphone. It comes with a powerful built in camera that can capture HD video. Therefore we programmed the app to open the camera and capture the video through the following code.

7

Task Name: Play Live Stream using HttpURLConnection
Accomplished By: : Sifat Sultan
Starting Date: 15 January 2014
Completion Date: 5 June 2014
Obstacles Faced:

- The smartphone will stream the video to a certain IP and Port address. This stream is very efficiently encoded such that it needs powerful library to be decoded.
- I planned to use a WebView or a Media Player to play the video streaming. However; neither WebView nor Media Player in android is not powerful enough to decode the stream and play video.

Percentage Completion: 70%
Output: I will use HttpURLConnection object to establish socket connection between the user smartphone and the server smartphone that is mounted on the quad copter. Once the connection is established the input stream of the socket is opened. Then a Bitmap object will be used to decode the jpeg image that is stored in the input stream using the function BitampFactory.decodeStream(InputStream inputStream).

8

Task Name: Android background service for live video streaming
Accomplished By: Sifat Sultan
Starting Date: 30 March 2014
Completion Date: 15 April 2014
Obstacles Faced:

- The activity in the foreground should be able to bind to the background service and access all data.
- Almost no documentation existed for streaming video from background.

Percentage Completion: 100%
Output: The andorid device which is mounted on top of the quadcopter can stream live video using the 4G network, 3G network or WiFI internet.

9

Task Name: Android background service for getting direction from compass
Accomplished By: Muhammad Obaidullah
Starting Date: 5 April 2014
Completion Date: 15 April 2014
Obstacles Faced:

- Tried to implement the tilt compensated compass using the libraries provided by researchers in the field but found out that the tilt compensation reduces the accuracy of the smart-phone compass.
- A smartphone compass does not provide the tilt compensated direction. It is the raw compass direction where the value can be extremely wrong if the orientation of the android phone itself is not correct.

Percentage Completion: 100%
Output: The andorid device which is mounted on top of the quadcopter can know the direction of the quadcopter and is ready for performing flight calculations.

10

Task Name: Ground Station Windows *C#* application interface design
Accomplished By: Muhammad Obaidullah
Starting Date: 2 February 2014
Completion Date: 20 February 2014
Obstacles Faced:

- Completely new to *C#* programming and **no part of the *C#* application is copied from internet. All lines of code written by Obaidullah.**

Percentage Completion: 100%
Output: Base station was designed to incorporate XBee communication control.

11

Task Name: Embedding GPS functionality in the *C#* application.
Accomplished By: Muhammad Obaidullah
Starting Date: 28 April 2014
Completion Date: 23 May 2014
Obstacles Faced:

- GMap library was used to embed the map into the *C#* application.
- GMap provides many pre-built functions like drawing a polygon and defining GPS point using Latitude and longitude.

Percentage Completion: 70%
Output: The ground station can be used to set way-points for the quadcopter to move and also get the updated longitude and latitude position of the quadcopter back from the quadcopter.

Task Name: Embedding the XBee serial port in the *C#* application.
Accomplished By: Muhammad Obaidullah
Starting Date: 19 February 2014
Completion Date: 20 February 2014
Obstacles Faced:

12

- Programming the Serial port and changing the settings for the port by using the combo box.

Percentage Completion: 100%
Output: The user can select the port to start communicating with the XBee explorer and select the baud rate by which the user can communicate. The data can be sent automatically using the hand gestures or it can be manually entered in the text-boxes and sent.

Task Name: GPS follower background service for way-point determined flight control.
Accomplished By: Muhammad Obaidullah & Sifat Sultan
Starting Date: 25 May 2014
Completion Date: 7 June 2014
Obstacles Faced:

13

- No previous knowledge of php and databases.
- Figuring out the perfect closed-loop control for reaching the way-point.

Percentage Completion: 0%
Output: The user can select the waypoints on the *C#* application and those waypoints will be uploaded to the database. From the database, the waypoints are grabbed by this background service and accordingly the OBFlightPackets are generated to make the quadcopter go to the desired way-point.

Task Name: Embedding live update of the altitude in the *C#* application.
Accomplished By: Muhammad Obaidullah
Starting Date: 24 February 2014
Completion Date: 25 February 2014
Obstacles Faced:

14

- Getting the serial string from the XBee and performing string operations to extract the height data.

Percentage Completion: 100%
Output: The sonar sensor which is mounted on the quadcopter can give live height feedback to the micro-controller and XBee. The *C#* application shows the current height in the nice progress bar.

Task Name: Writing the Arduino C code for the AtMega328.
Accomplished By: Muhammad Obaidullah
Starting Date: 11 February 2014
Completion Date: 27 May 2014
Obstacles Faced:

- Sonar sensor requires very slow triggers of about 20 Hertz. The micro-processor cannot wait for the sonar sensor to send the sound signal, receive the signal back and calculate the distance.
- Drawing a clear line between the inputs and the outputs of the quadcopter system and figuring out the finite state machine diagram.
- Implementing the OBFLightPacket protocol in the C language by casting the incoming data into byte array and send to the ESCs.

Percentage Completion: 90%
Output: The AtMega 328 chip can be mounted on the controller board and the quadcopter is fully functional and ready with features such as AtMega 328 to android communication, height stabilization, XBee control, and mode select.

15

Task Name: Designing & updating the circuit diagram for the controller board.
Accomplished By: Muhammad Obaidullah
Starting Date: 4 February 2014
Completion Date: 26 March 2014
Obstacles Faced:

- The tilt compensated compass is connected via I^2C Master-slave configuration and the JTAG SPI programmer header is also connected to AtMega328.
- Providing the 5V logic power to the circuit board by using the ESC 3 pin headers.
- Kill switch is an essential and necessary requirement especially when in initial testing stages. Implemented the kill switch using the infrared receiver.

Percentage Completion: 100%
Output: The quadcopter can communicate with the tilt compensated compass and be programmed using the JTAG SPI header directly using the *.hex* format binary file. Additionally, the power is given to the circuit board using the ESC's logic output. This improves the efficiency of the system because the ESCs have a buck converter inside to step the voltage down from 11.1V to 5V using the PWM switching regulation technique. Any TV remote with infrared LEDs can be used to shut the quadcopter OFF completely as any change in the infrared signal causes and interrupt to be raised whereby the quadcopter switches immediately to the emergency mode/state.

16

17

Task Name: Soldering & mounting the equipment on the PCB
Accomplished By: Muhammad Obaidullah
Starting Date: 24 December 2013
Completion Date: 28 May 2014
Obstacles Faced:

- There was a vias under the AtMega so it had to be soldered before all bracket was to be soldered.
- The PCB machine failed several times before a good PCB was printed. A lot of time and effort was wasted in this. This also caused the project to be delayed.
- The first versions had some faults in them so had to be printed again and again.

Percentage Completion: 100%
Output: The quadcopter PCB is completely soldered with all the features and ready to be mounted on top of the quadcopter for autonomous flight.

10 Conclusion

The project comes with a variety of features that needs to be accomplished in the mentioned schedule. This is made possible due to the Gantt chart that shows a detailed breakdown of how the tasks will be approached. In our chart we tried to give a very reasonable duration to each of the task that needs to be carried. The duration was based on various factors and experiences. For instance, one may notice that; Printing Body; one of our sections in Gantt chart took a duration that is about double and triple that of other parts, it was decided after a serious discussion and a unanimous agreement from all the member that it; in reality; required a considerable amount of time as it often demanded that the task be redone a couple of times until a desired result is reached. Similarly there are some tasks that were given a very brief amount of time as the team believed that we cannot afford spending a significant portion of the given time in a task that doesnt have that amount of an impact in the later and the main portion which deserved our focus time. The task of choosing the part resulted to be a challenging endeavor since we had to balance the quality with the cost of the product, it must be also mentioned that a great deal of our part was ordered from international countries. This was among other a primary obstacle that this project will face.

The team members would like to ask everyone for honest prayer so that the project can be successful.

References

[1] Jennifer Bell. (2013) *Road accidents account for almost 70% of head injuries at one UAE hospital* [Online]. Available: `http://www.thenational.ae/uae/health/road-accidents-account-for-almost-70-of-head-injuries-at-one-uae-hospital`

[2] ADAFRUIT (2010) *FTDI Friend Breakout Board+ (tutorial)* [Online]. Available: `http://www.adafruit.com/blog/2010/09/16/ftdi-friend-breakout-board-tutorial/`

[3] Prospector. *Polylactic Acid (PLA) Typical Properties* [Online]. Available: http://plastics.ides.com/generics/34/c/t/polylactic-acid-pla-properties-processing

[4] Mark Johnson Cutler. (2010) *Design and Control of an Autonomous Variable-Pitch Quadrotor Helicopter* [Online]. Available: http://acl.mit.edu/papers/Cutler_Masters12.pdf

[5] Leap Motion Documentation. (2013) *Understanding the Java Sample Application* [Online]. Available: https://developer.leapmotion.com/documentation/Languages/Java/Guides/Sample_Java_Tutorial.html

[6] Mando. *How a Gyro Works* [Online]. Available: https://learn.sparkfun.com/tutorials/gyroscope/how-a-gyro-works