Team 3 First Deliverable Report

Steve Kosovich, Brian Dignam, Paul Rodriguez, William Daniels

# Goals For the first deliverable

For our first deliverable, we had set forth a rather broad definition of what we wanted to have accomplished, since many aspects of the project were still unknown. The goal that we said we would have done by now was: make sure all the sensors work properly. Now, this is a bit vague, so here is a more substantiated breakdown of what we were actually trying to accomplish:

1. Make sure that we could obtain information from the various air quality sensors, via’ the shield and Seeduino device, and that the serial communication seemed to be working fine over USB.
2. Get android (ice cream sandwich) installed on the PCDuino, and build the SDK test-app provided by DJI on it
3. Start work on getting the GPS mapping routines build up, so that we don’t get caught up on that later
4. Start building the physical mounting devices for the DJI (aka: plexiglass ‘support’ for all the sensors/hardware) as well as getting the test app (on my phone) connected to the Phantom 2 to at least remote-control some aspect with the app.

Accordingly, here is the breakdown of how we have been distributing the work (numbered to match the work above)

1. Brian Dignam
2. Paul Rodriguez
3. Steve Kosovich
4. William Daniels

# Results from first deliverable

We had some good success leading up to this first deliverable. The main ‘wall’ that we’ve encountered, was the fact that it seems the DJI Phantom 2 does not seem to have any built-in wifi capabilities. Although these may be exposed through the use of the “Wifi Range Extender” it is unclear whether or not this will even work at all with the regular phantom 2, and not the phantom 2 vision, or vision + (P2, P2V, P2V+). While this investigation continues, we will continue work on aspects of the project which are not directly related to the drone itself, in the hopes that if we need to, we can actually migrate to a different drone with similar capabilities. We were able to verify that we could properly connect to the SEED air quality sensors using serial, however, and

## Results from air quality sensors

The air quality sensors gave us the largest success, as we were able to interact with them, and pull data from the sensors quite easily, and without any real fuss. An example of what the code looks like for pulling data from a few of the sensors, looks as simple as the following sketch:

/\*

Author: Brian Dignam

Date Created: 3/1/2015

\*/

#include "DHT.h"

#define DHTPIN A0

#define DHTTYPE DHT11 // DHT 11

DHT dht(DHTPIN, DHTTYPE);

#define MQ2PIN A3

int TIMESTAMP;

void setupDHT()

{

dht.begin();

}

void setup()

{

Serial.begin(9600);

Serial.println("Serial Connection Established");

TIMESTAMP = 0;

setupDHT();

}

float getHumidity()

{

// Reading temperature or humidity takes about 250 milliseconds!

// Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)

float h = dht.readHumidity();

if (!isnan(h))

return h;

else

return 0.0;

}

float getTemp()

{

// Reading temperature or humidity takes about 250 milliseconds!

// Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)

float t = dht.readTemperature();

if (!isnan(t))

return t;

else

return 0.0;

}

float getMQ2Gas()

{

float vol;

int sensorValue = analogRead(MQ2PIN);

vol=(float)sensorValue/1024\*5.0;

//Serial.println(vol,1);

return vol;

}

void loop() {

Serial.print("Timestamp: ");

Serial.println(TIMESTAMP);

Serial.println();

Serial.print("Temp: ");

Serial.println(getTemp());

Serial.print("Humidity: ");

Serial.println(getHumidity() );

Serial.print("MQ2 Gas: ");

Serial.println(getMQ2Gas(),1);

Serial.println();

TIMESTAMP++;

delay(2000); //2s

}

As you can see, it’s (thankfully) almost trivial to obtain information from these sensors, which makes that portion of the project all the simpler.

## Results from the PCDUINO

The PCDuino portion of this deliverable, was admittedly small, however it’s still an important step. We first wanted to make sure that we could get android loaded onto the PCDuino itself (since other teams had struggled with this). And then, we wanted to make sure that we were able to deploy the bundled test SDK Demo app onto the pcduino, without any trouble. After some initial difficulties, we were able to get the app installed on the PCDuino, and would be happy to provide some screenshots of that as necessary. It is currently running Android SDK 14 (Ice Cream Sandwich), which is the minimum required SDK for the DJI libraries. The next deliverable will have a much larger influence on the PCDuino development, so stay tuned for exciting updates from that end of the spectrum.

## Results from the GPS/KML

{Insert results from Steve}

## Results from the Drone Interaction/Physical setup

For the actual interaction with the Phantom 2 drone itself, we had mixed results. The majority of the work was attempting to find a way to interface with the drone via’ the android application (just used from my phone at this time). The attempt to reach the drone over wifi was utterly unsuccessful. From various research, looking online, and asking in the forums, it doesn’t appear that the drone itself broadcasts any wifi, unless you are using the Phantom 2 Vision + , which then includes a wifi module built-in. I even found pictures online of the built-in wifi module, and our phantom is definitely devoid of it. (more discussion can be found on the dji forums here: <http://forum.dji.com/forum.php?mod=viewthread&tid=11719&page=1#pid91908> ). I have some remaining hope that we can still access the drone via’ WIFI using something called the ‘wifi range extender’ that can be purchased, but I am losing hope on that front. As far as the power of the drone to lift heavy objects, it seems to be fine in that regard. See the following youtube video for a demonstration of it lifting object attached to it: {insert youtube link here}

# Summary

In summary, I feel that we met this deadline, and are currently on track to success for our project. Once the main blocker (the ability of the drone to be controlled via’ android) is resolved, we can really start providing some quantitative results that can be easily seen, such as having the drone respond to sensor input. In the event that the drone CANNOT be controlled via’ wifi in any way, we will hopefully switch over to the parrot Bepop (the version 3.0) drone, which can then also hopefully lift enough weight. If it cannot, the project will be more ‘proof of concept’ and we will create a simulated environment with inputs synthesized to reflect what a real world situation would look like.