Critical Design Review

Introduction

There is currently high demand for a service which can deliver online orders quickly, cheaply, and effortlessly to consumers. Existing solutions which attempt to address this need incur too high of a cost to consumers and distributors, have a detrimental impact on the environment, require an excessive amount of human labor, and are still slow by most impatient consumer's standards. The Quick Unmanned Air Courier project intends to create a delivery service which can outclass all others by delivering packages swiftly and at an affordable price.

The primary aspiration of this project is to design and implement a system consisting of an unmanned aerial vehicle and a user interface capable of transporting small packages between any two geographic locations.

The objectives that were considered when designing the drone are as follows:

- The drone must have a cost of less than $600
- The drone must be able to handle payloads which weigh up to 3 lbs and up to the size of a large novel.
- The drone must be able to fly at a height which allows it to go over most obstacles while still complying with FAA regulations regarding UAV operation.
- The drone must receive instructions from a central server which dictate its starting and ending locations.
- The drone must be capable of changing its destination mid-flight via user input from an online web interface.
- The drone must send its location to an online web interface, allowing any user to check the status of his or her packages.
- The drone must land safely if low on power or afflicted by some other mechanical malfunction.

The design constraints considered for our delivery drone can be classified into different aspects. One of these constraints is a financial constraint. Our delivery drone must be able to satisfy our goals and objectives while still maintaining a low cost to allow manufacturers to sell this product at a reasonable price. The constraint we have chosen for this project is for our drone prototype to cost less than $600. However, this price is not taking into account the maintenance that the drone will require, which may add a substantial amount of money over a long period of time.

Our drone is also affected by a physical constraint, especially when having to carry a package around. Having more weight put into the drone means that it will require more energy to lift up and fly, which in turn means it requires a bigger and stronger battery. For this reason, we have decided to constraint the weight of the package our drone will carry to 3 pounds and the operating time of our drone to at most 20 minutes before needing to recharge the battery.

Another physical constraint is that of weather conditions. The GPS module could cease to function during cloudy weather, and rain might damage the electronic components of the drone. For this reason, we have chosen to constrain the operating environment of our system prototype to fair weather conditions.

The last constraint of our project is a legal constraint. It must abide by all the FAA laws and regulations. This means our drone needs to fly at a safe height and always in line-of-sight of the operator.

The validation and testing procedures of our project consists of a bottoms-up approach, where we will test each component individually, making sure it works, before putting the different elements together and testing how they interact with each other. Once the individual pieces have been tested, control of the drone will be tested by sending signals to it through the telemetry. Flight will also be tested, first by being able to rise from the ground and stabilize in the air, and then by doing some slight maneuverability to ensure the flight controller works as intended. Once flight and communications have been tested, we will send coordinates to the flight controller to see if the drone can safely travel to its destination. After all of this has been successfully tested, we will add weight to the drone and test how it performs under more physical strain. If we are able to successfully deliver a package, then the project is a success.

Proposed Design

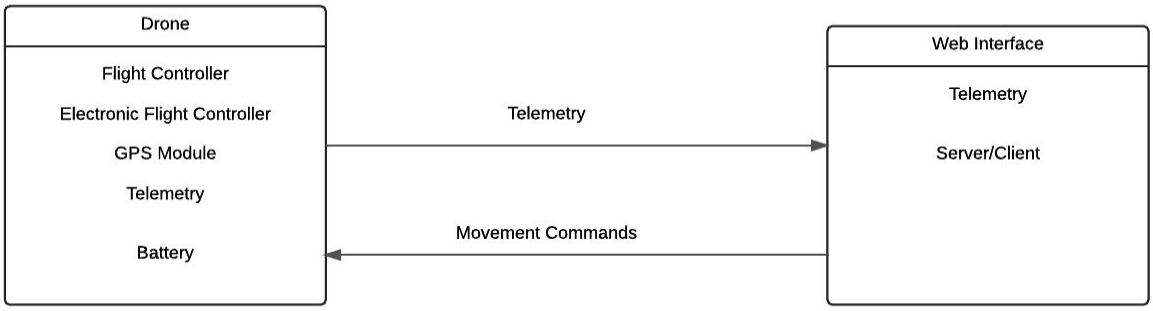
Updates to proposal design

Due to an unexpected illness, our former team leader Andrew Kirfman has been hospitalized and will not be coming back this semester. While the initial project design has not been drastically changed, we have decided to scale down the complexity of the front-end aspect of our online web application. A more simplistic design has been chosen, where the user will input only the delivery coordinates in a map and see the current location of the drone on the same map. The server will communicate the coordinates to the drone, initializing a new route every time a new coordinate is given.

The delivery process has also been scaled down. Previously, we wanted to have the user input both starting and final coordinates. The drone would travel to the starting location and await for a user to load a package before departing to the final destination. However, this adds unnecessary complexity that is not relevant to our main goal. This and the fact that we have one less person in our group has led to the decision of dropping this feature. Now the user will only input the destination coordinates. The package is expected to be loaded before setting the coordinates. The drone will be traveling to the destination point, and then after a short delay it will return to the initial coordinates. The initial coordinates will be a default value in our server.

System Description

High-level block diagram:



Functional description of the different parts and interfaces:

Drone

The drone is the physical aspect of our project. It contains all the necessary elements for it to be able to communicate with an online server and fly to a provided coordinate while carrying a package. A description of these elements is as follows:

- The drone will contain a sturdy frame with an area to load a package. This frame includes a cover to protect the electric components of the drone from natural elements.
- The drone will have four motors, each with rotary blades to give enough lift to allow for flight.
- The motors will be controlled by an Electric Speed Controller (ESC) which controls how much power goes to each of the motors to allow for stable flight.
- The drone will contain a GPS module to allow for real-time location.
- The drone will contain a telemetry module to allow for communication with our main server.
- The drone will contain a flight controller which will be used to connect and control the other elements. This is the brains of our drone. It is what controls the motors to allow for flight and movement. It uses the telemetry to communicate with the server and a GPS compass module to receive coordinates and travel to them.
- To power the motors and the flight controller, a 3S 5200mAh battery will be used. This should give enough capacity to allow for around 20 minutes of flight.

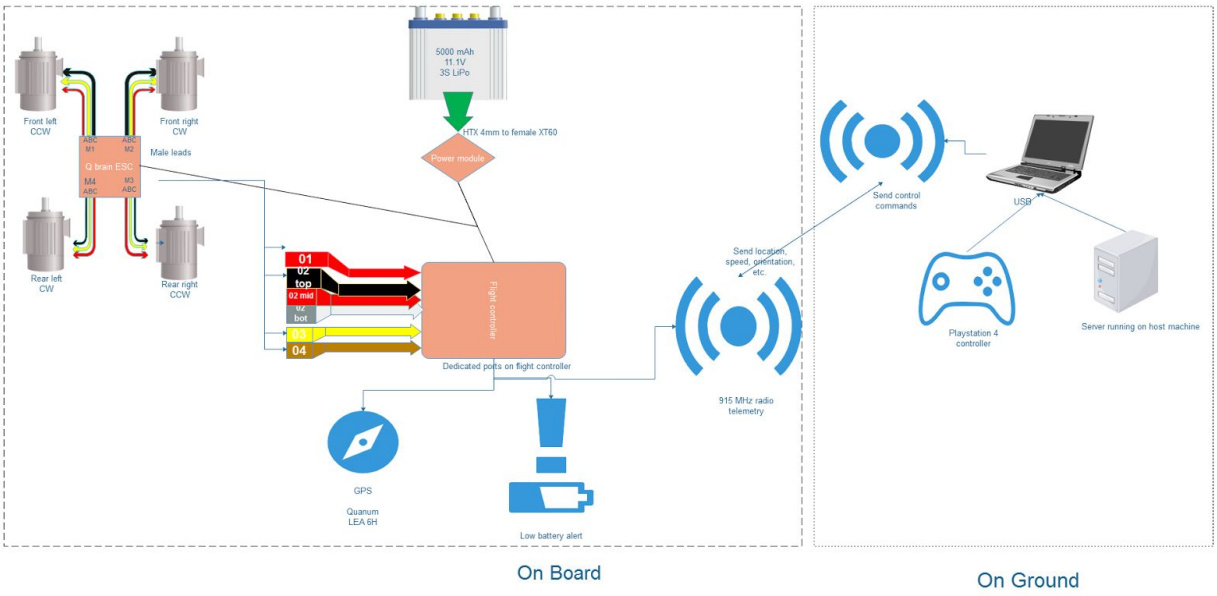
Online web interface

Our online web interface will consist of a map provided by the Google Maps API. The user will be able to input coordinates on this map, which our server will then send to the drone's mission planner. The user may also provide an address, in which case the Google Geocoding API will decode the address into geological coordinates before sending them to the drone's mission planner. The map will also show in real time the current location of the drone.

Online web server

Our server will be controlled by a Mission Planner which will be communicating in real time with the drone's flight controller via a telemetry. This mission planner will also communicate with the online web interface to provide updates and to receive the drone's destination coordinates. The mission planner is responsible for making sure the drone reaches the destination provided by the customer through the online web interface.

Complete module-wise specifications



Online web interface

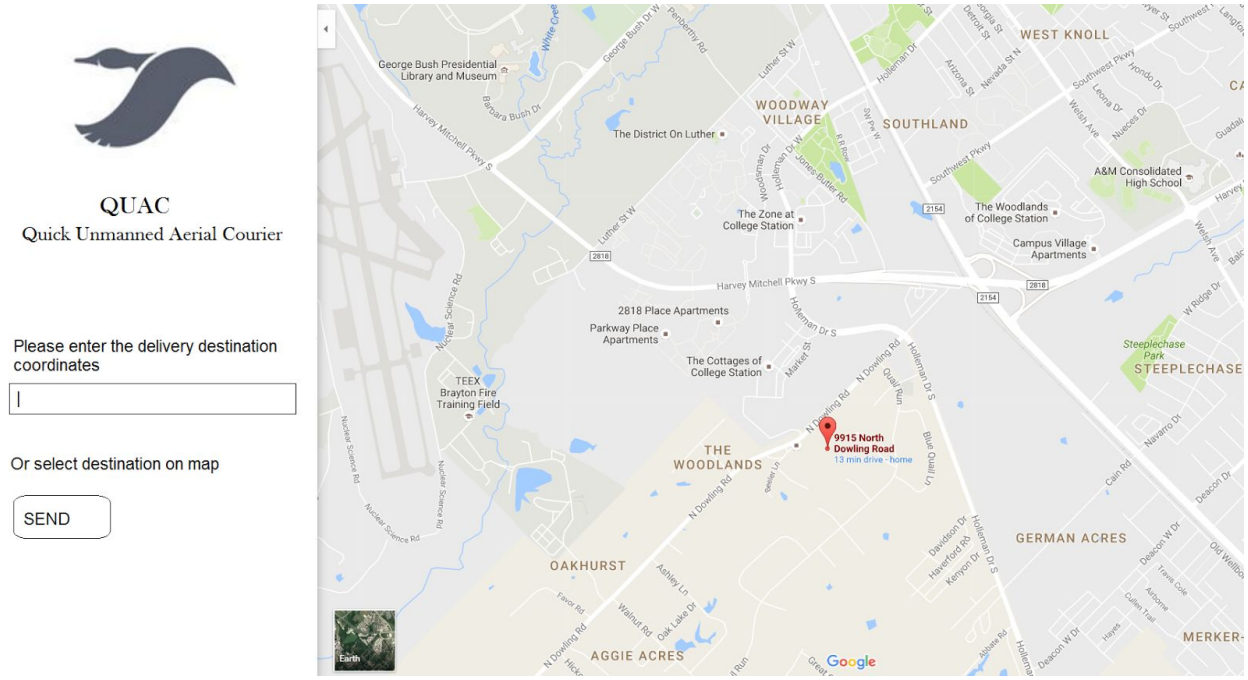
Our project's interface has been designed with a simplistic goal. What is required of the online web interface is that the user can intuitively select a location where the user wishes to deliver a package, and then track the progress of the delivery by being able to see in real time the current location of the drone. This will be achieved by using Google Maps API, which is free for up to 2,500 requests per day. For the purpose of this project, this will be more than sufficient.

To use the Google Maps API, we are required to obtain an API key from Google, which is free for standard API users. Using this key allows us to monitor our application's API usage in the Google API console, as well as a free daily request quota. After obtaining a key, it can be specified in the online web interface when loading the Google Maps API.

Google's Geocoding API will also be used. Geocoding lets us convert addresses into geographic coordinates. This will let the users input a street address which will be converted to a geographic coordinate which the mission planner can then send to the drone for it to follow. This is achieved by sending a JSON formatted request with the address attached to it through an HTTP interface.

There are plenty of guides provided in the Google Maps Developers webpage which will let us succeed in what we are trying to accomplish. We have decided to use JavaScript for our website, so we will be using the Google Maps JavaScript API, as well as the Geocoding API.

A rough image of how we plan to display our online web interface is as follows:



As we can see, the user will have an option to introduce the delivery destination address as an input, or he may opt to select the destination on the map itself. When the SEND button is clicked, the server will send the location coordinates to the mission controller, which will then send the instructions to the drone to travel to the desired destination. If the delivery destination is given as an address, the Google's Geocoding API will be used to translate this address into coordinates before sending them to the mission controller. This map will also display the real-time location of the drone in flight.

Mission Planner

The Mission Planner that will be used for this project was developed by ArduPilot. It provides all the necessary tools to effectively communicate with the drone and to send it instructions, including traveling to a geographical location. It will also allow us to monitor the vehicle's location in real time to display to the user. The Mission Planner allows the drone to take off, land, control the drone's height, attach different waypoints, perform actions depending on conditional variables, and so on.

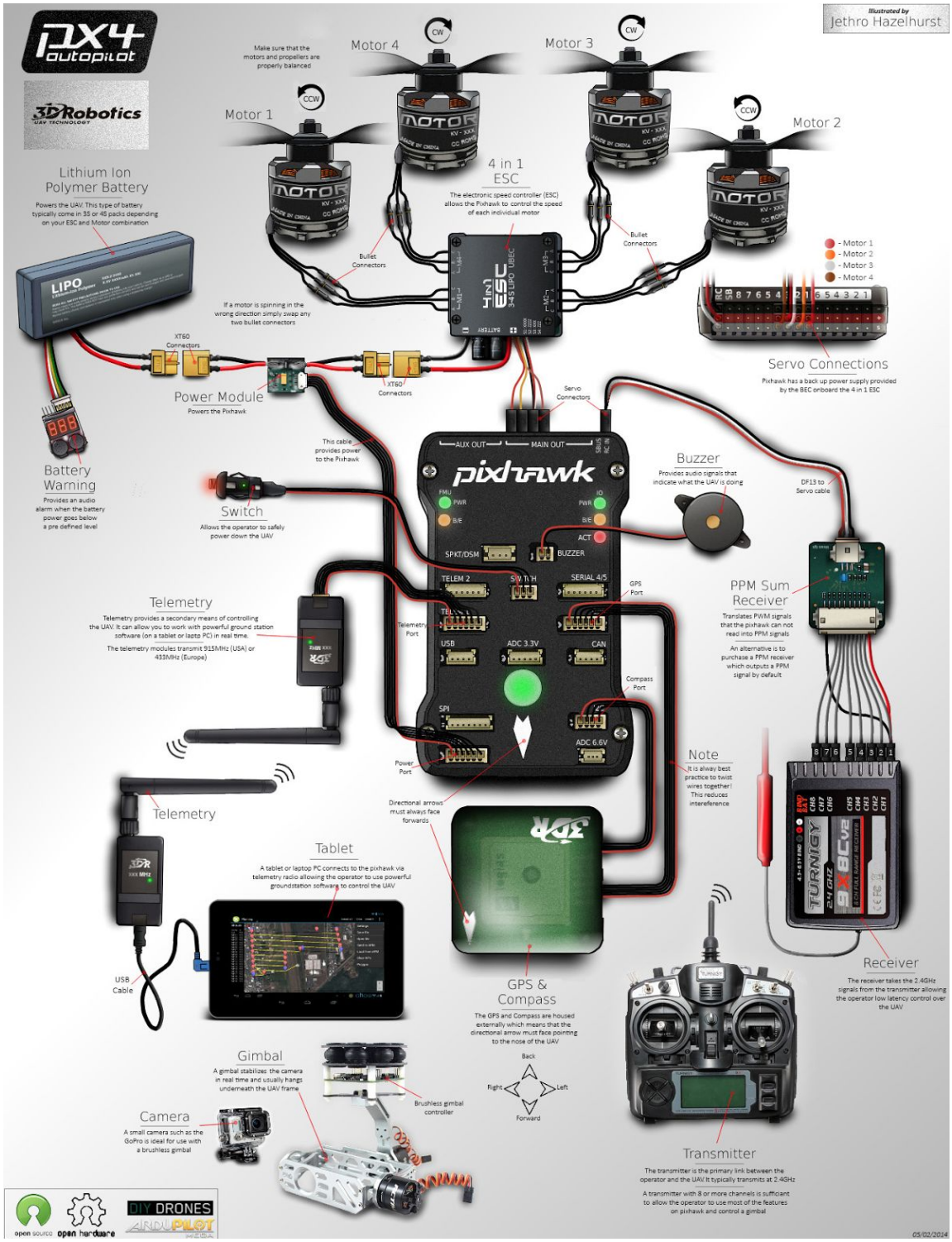
The Mission Planner is what we will use as our server. We will have the Mission Planner running in a computer. Each time a new request is sent to the online web interface, it will send it to the Mission Planner, which will then prepare the drone to travel to the specified coordinates.

The Mission Planner is also responsible for getting the drone's GPS location and sending that data to our online web interface to display to the user. The Mission Planner can save previous flight missions' data, which might be added as an option for the user to download if time allows it.

While the Mission Planner will not be visible to the user, it has a very nice interface which will help us debug any problems that we might have with our drone. It displays the drone's battery information, the GPS coordinates, the direction and tilt of the drone, and the signal strength. A sample image of the Mission Planner display is as follows:

This image was obtained from the ArduPilot website. Here we can see all the information about the drone displayed to the left, and the GPS location to the right.

Drone Flight Hardware



The flight hardware consists of the flight controller, GPS compass, ESC, motors, and propellers. The above diagram is mostly accurate except we will will not be using a camera mount and will be using a video game controller instead of a standard RC controller.

There have been no changes in regards to the type of each item listed above, but the final layout on the drone will need to be carefully laid out. Sam will 3D print parts that will secure the flight hardware as well as the payload. He will use Solid Works to design the pieces and use the CS department's 3D printer (must go through the TA) to print them. As of now, the flight controller is temporarily attached with zip ties and the battery is velcroed underneath.

Power

The drone is powered by the turnigy nano-tech 5000mAh 11.1V battery. It is a 3S LiPo battery with a 35C constant discharge rate and 70C burst discharge. This means the battery could provide a constant 35A for 5 hours. This will be sufficient to give us a flight time of at least 20 min. The battery weighs 409g. The current plan is to attach the battery underneath by creating a slot in the 3D printed carriage. The battery provides power to both the ESC (and thus the motors) and the flight controller.

ESC and Motors

The electric speed controller chosen is the Q Brain 4x25A brushless quadcopter ESC. The choice to use an 4-in-1 ESC will allow us to mount the device in the middle of the frame and keep our wiring tidy and give us better balance. The ESC was programmed using the Turnigy BESC programming card. This is necessary to set parameters such as the number of cells of the battery, type of battery, sensitivity of throttle, brake, and threshold for low cutoff voltage. The Q Brain can provide continuous current of 25A to each motor which is more than a motor requires. The ESC weighs 112g.

The ESC has 3 - 3.5mm bullet leads for each motor. These will be run through the arms of the drone and connect to the Turnigy Multistar 3535- 850Kv 14 pole motors. Each motor has a working current of 19 amps and weighs 58g. The choice of 850Kv means the motors will spin at 850 RPM for every volt supplied. The relatively low choice of 850Kv has a lower top speed than higher rated motors but higher torque. The motors will be laid out as following: M1 - front left, spin CCW, M2 - front right CW, M3 - rear right, CCW, M4 - rear left CW.

Telemetry/ Communication

Communication to and from the drone happens over the bundled telemetry module. The telemetry module consists of a transmitter and receiver on the drone and another connected to a laptop computer base-station. The telemetry communicates in a serial-like fashion over 933 MHz Radio Frequency (RF), this allows full-duplex communication with the drone with mission information streaming from drone to server and desired movements from server to drone.

Undercarriage

The undercarriage consists of a 3D printed design that carries both the package payload and the battery. This design enables each delivery to have a fresh battery when the package is changed out for a new destination. The container will be designed such that it hangs from a platform connected under the main stage where the flight controller is held and rests on the landing skids under the quadcopter.

Project Management

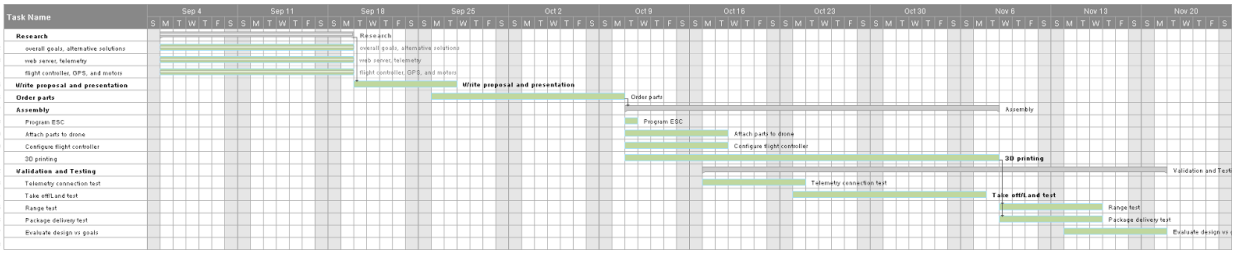
Significant changes have happened in regards to project management. Our team leader Andrew is still out sick and it is very unlikely he will be coming back. The project will have to be completed by the remaining three team members. The work has been redivided among Regan, Sam, and Diego, the updated Gantt and Pert charts can be seen below. Fortunately, the issue with Andrew happened sooner rather than later and it should not affect our overall design goals. We will continue to make progress as scheduled.

In the absence of our team leader, we have decided not to elect a new team leader and instead will carry on as mutually responsible individuals. We have been meeting twice a week and will continue that trend (at a very minimum) until the project is complete. Every week we discuss our objectives for the week, divide up tasks to accomplish those objectives, and report on results and accomplishments from the previous week. The general breakdown of tasks is as follows:

- Regan - keep minutes, order parts, program ESC, mount parts on drone
- Sam - mount and wire parts on drone, 3D printing, testing and validation
- Diego - design back end and front end software, flight controller configuration

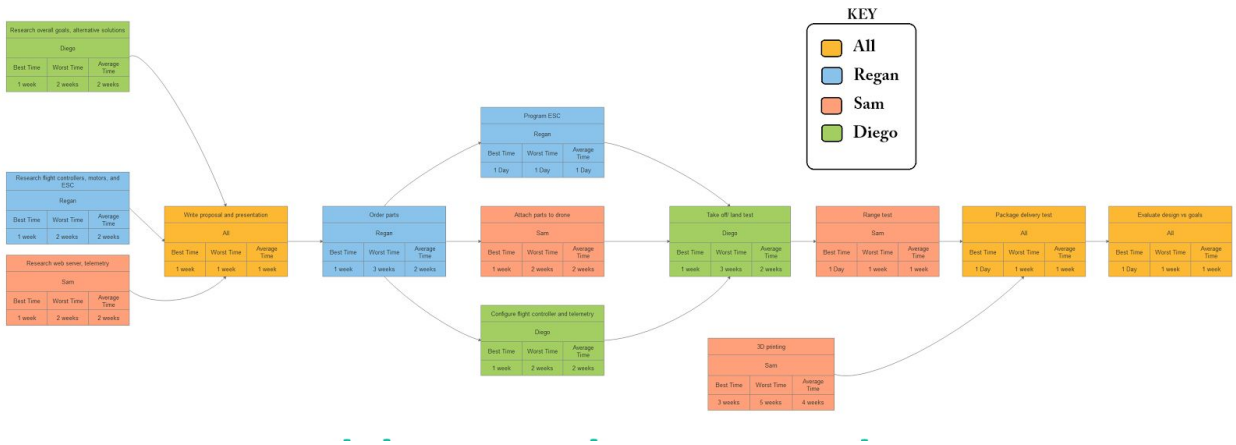
Updated implementation schedule

The Gantt chart has been updated to reflect Andrew's absence as well as the actual time to order parts. The current projected completion date is November 20.



The pert chart has also been reorganized and the tasks are color coded according to the individual responsible for the work. The worst case scenario results in a 14 week critical path. This scenario accounts for delays in shipping and a three week period to get the drone to take

off and land. The more likely scenario is a 12 week schedule completed at the time specified above.



Updated validation and testing procedures

Since our design is complex and consists of a multitude of interconnected and interdependent parts, testing will be a critical part of our design and implementation process. Even though it may seem self-contained, our UAV delivery system consists of several key components whose individual function must be assured. From these key components, more complex systems will be built on top of them which also will need to be tested. This hierarchy of components and testing will continue until the highest level, the whole delivery system itself. Therefore, ensuring the functionality of each system as it is built is necessary in order to guarantee the functionality of our design as a whole.

Since a delivery UAV is not one monolithic device, it can be broken down into modules that can then be tested one by one as the device is built. This allows the team to change designs when problems arise which is cheaper in terms of money and time than realizing at the end of the project that the full design does not work because of choices made early on. The testing procedure can be best described top-down but will be performed bottom-up, from a full demo where the delivery UAV does a full automated delivery using information it receives over a GSM radio and GPS sensors, all the way down to testing if individual motors power on upon arrival. A full breakdown of testing procedures will be explained below.

The delivery UAV can be broken down into two constituent parts: flight and control. Flight can be tested using a pilot and a sending commands via telemetry. The drone should be able to take off, land and travel the distances laid out in our technical specifications. Being able to to do this comprises success for flight. Control is made up of the web server and its communication with the drone. Success for control means being able to take a GPS waypoint and send it to the drone. The onboard radios on the drone should then be able to communicate with the flight controller on the board, moving the drone along desired path.

Flight can be broken down into two parts: flight control and freight. Freight can be tested by showing that the drone can hover, take off, and land while holding a package of the desired size and weight. Flight control can be tested by being able to control the motors from a remote. At this point it should be determined how long the batteries can last under full use.

Control can also be broken down into parts. There are two parts of the control apparatus: the server software and its radio and the onboard radios on the drone itself. This can be tested by sending test GPS waypoints from the server to the onboard radios in lab. If the onboard radios can receive the data then acting upon it can be done in the flight control stage detailed previously. The server software can be tested in smaller modules with a test driven development methodology (TDD). Using TDD the developers can build the server and client software with tests for each function, which can then be used to test logical modules.

After testing the server software, the online web interface can be tested to communicate with the server. If an address is input on the online web interface, the Google Geocoding API should be able to transform the address into geographical coordinates, which we will send to the Mission Planner. The online web interface will be successfully tested if the Mission Planner can successfully receive the coordinates.

Lastly, it is important to test each device that we order to see if it is in working condition. The frame should be assembled and a mock package should be suspended from it to assess whether the skids can support the desired load. Next the flight controller and telemetry should be powered on together. If they power on they can then be connected to our workstation and it can be affirmed whether or not the system recognizes the flight controller. The motors should each be tested to see if they power on. Once propellers arrive thrust can be measured by attaching the motors with propellers to a scale. Comparing this info with our estimates will determine if adjustments need to be made to our specifications or if different parts are required.

When all of these tests are completed the UAV system should be able to satisfy our need statement within design constraints.

Preliminary Results

Electric Speed Controller (ESC) and Motors

At the time of writing, we have yet to get the motors operational. This is because the ESC must be configured in two different stages. While Regan successfully programmed the ESC using the programming card, Diego was unable to calibrate it with the flight controller because the battery connection to the ESC has not yet been soldered. The ESC was programmed using the following device and settings:

1. Brake: OFF
2. Battery Type: Li-xx

3. Cut Off Type: Soft Cut
4. Cut Off Voltage: Low
5. Start Mode: Normal
6. Timing Mode: High
7. Music/LiPo Cells: “B”
8. Governor Mode: OFF

When wiring the motors, the initial design required that we run the wires through the arms of the drone. This is still the plan however, a small screw will have to be removed and later replaced in order to do so. Also, the black bullet lead on the front left motor slipped off while disconnecting the motors. For now, the wire was simply reinserted and the connection taped with electrical tape.

Telemetry and Mission Planner

Currently, we have tested the communication between a local server and the drone while being close to each other. The telemetry has been successfully tested and works with very good connection and provides almost real-time updates on its orientation. However, the GPS module and the compass has not been successfully calibrated since we have only tested the devices inside the lab. These elements will have to be tested on an open environment to ensure a proper signal can be obtained. We also have not yet tested the telemetry over long-distances, and we assume the signal will be slower the farther away the drone is. However, since most of the stabilization and travel is done through the drone itself, we do not believe this will be a problem.

The Mission Planner used for testing was developed by ArduPilot. We successfully connected our drone to the Mission Planner, and calibration was started. Some of our components have not yet been calibrated, which is why we have not been able to test the stabilization tools of the Mission Planner, but once all our components have been calibrated, we will be able to use a remote controller to control the drone and test the flight capabilities of the drone.