UAV Assisted Energy Delivery

FINAL REPORT

Team: 43
Client/Advisers: Randall Geiger & Degang Chen
Alexandra Lowry
Brendan Rohlik
Connor Wehr
Garth Flaming
Kaitlin Maass
Kevin Angeliu

Team Email: sdmay19-43@iastate.edu
Team Website: http://sdmay19-43.sd.ece.iastate.edu

Revised: April 29, 2019

Table of Contents

List of Figures

List of Definitions

UAV: Acronym for Unmanned Aerial Vehicle, such as a drone

FAA: Acronym for Federal Aviation Administration

LiPo: Acronym for Lithium Polymer, a type of battery

DroneKit: API (see definition of API) used to code drone

API: Acronym for Application Programming Interface

0 Executive Summary

Our project is UAV Energy Delivery. In today's world there are countless devices that need power but there are not many ways to power them and most ways are very inflexible. We would like to address that issue with drones. Drones are able to take power to several locations easily. Drones are also incredibly flexible, so if the locations that need power are moved often or if the landscape is rough none of that maters to the drone. Our drone is able to fly to a location and land on the landing station, autonomously. Once it has landed it will make a physical connection with the landing station in order to transfer electricity. In the scope of our project we just attempted to have one drone go to one landing station to deliver power. The vision of the project goes beyond that with multiple drones and landing stations. We would also have the drone(s) have a home base where they would be able to dock autonomously and recharge their batteries. We have a few different flight modes for the drone. The first is controlled primarily by GPS location in order for the drone to fly to the landing station. Once the drone gets to the landing station's coordinates it will switch to its landing modes. This is comprised of color detection and QR code detection. Using those two pieces to detected the landing pad's relative location to the drone the drone will drone and move precisely in order to land in the direct center. There is a battery tied to the underside of the drone that has a cable attached to it that is in the center of the drone. Once the drone lands in the center of the landing station it will make a physical connection with the landing station. The cord is magnetic so that it does not have to be exactly in line in order to make the connection. Once it has made the connection the drone just has to wait for the power to be transferred and then take off once it is finished. Our project scope does not include a way for the drone to monitor the amount of electricity being transferred or if it has finished. However, in the vision of the project this would also be a component.

1 Requirements Specification

1.1 Functional Requirements

1. The drone will be able to take off and fly to a specified location autonomously
2. Drone will dock with landing pad for power transfer autonomously
3. Drone will be able to deliver power from the payload battery to the landing pad

This project is made out of three different pieces that work together in order to accomplish the objective of having UAV power delivery. The first piece is to fly the drone to a remote location. In order to do this the drone will have to have the ability to fly autonomously to a given GPS coordinate. The drone will fly up to an altitude where it is expected to not have to run into any obstacles. We do not have object avoidance in the scope of this project and will not implement anything to avoid obstacles besides where we are testing and the previously mentioned flying to a high altitude. This is the first step for the drone to complete its requirements and is needed for anything else to function. The drone will also receive the coordinates of the node before it takes off and will map a route to it. The drone will be able to receive commands as necessary.

The second part is to attach the drone to the remote node. In order to do this, there needs to be a secondary flying mode that will allow the drone to precisely land. To complement this there is also image processing to be able to identify where the landing pad is and how close the drone is to it. With the image processing the drone will be able to calculate how centered it is on the landing pad or how far off it is. Once it has calculated how far it is off it will adjust and recalculate to see how close it is. It will be able to repeat this process until it has landed on the node and as it lands makes a connection to the node. Here it will attach to make a valid connection to the node.

The third part is to deliver energy to the node. During this process the power in the drone will transfer over to the node. The drone will keep track of how much charge it has so that it knows when it is empty so that I can leave and return to base. During this process the drone will efficiently and quickly transfer power to the node. It is important that this is a quick process because if the drone's battery gets to low it will go back to base before it has transferred all of the payload's energy. The drone will also remember the place it left from in the first step so it can return there in this step.

1.2 Non-Functional Requirements

1. Scalability of system from meters to miles
2. Security of system
3. Performance with regards to wind resistance and flight efficiency

The scope of our project calls for our system to be functional within the length of a standard football field. However, the system itself should have the capability to be scaled up to distances of one mile or more, with the proper hardware. This will allow the application to have greater use in future iterations.

Due to the testing location, our drone will have to deal with wind, possibly significant gusts of wind. As such, our system should be able to handle this wind and still arrive at the landing station. This wind will also create a strain on the drone's performance, possibly affecting its flight time and efficiency. The drone will have to be able to land safely if its battery life will not allow it to reach its destination.

2 System Design and Development

2.1 Design Plan

Our project consists of the programming of an Intel Aero Ready-to-fly drone to travel to a landing station designed by our group, initiate its multi-process landing procedure, and magnetically connect to the energy receptor located in the center of the landing station. This process was split between two main groups, energy transfer and drone programming.

Our first semester was focused on our decision of which drone to use for the project, followed by mostly research and finding any relevant information or prefabricated code that our group could reuse to limit the reinvention of the wheel. We were able to create and run a few simulations for the drone, but had some technical issues it limited what we were able to accomplish in the fall. Our energy transfer group and other members created the frame for the landing station as well as designed and fabricated a carriage for the payload battery.

The programming group was split further into a flight & troubleshooting team and precision movement & landing code team. We were to separate the coding efforts and concurrently develop and integrate our respective pieces of code together. The troubleshooting team finished the flight code in early February and was working on resolving package corruption. The precision movement & landing code team finished the code in late March; However, we incorporated another design feature into the landing to create more consistency when landing.

The portions of our code are GPS directed takeoff & flight, image processing for color-mapping, precision movement, QR assisted landing code. The landing station consists of platform with a magnetic cord connection located in the center, attached to an off-loading battery. The station is colored bright orange to create a distinct contrast between most environmental objects and the station. The landing station, as well, has a relatively large QR pattern placed on the center (with a 10-inch gap between the QR pattern's edge and the station's edges) to be used by the QR assisted landing code.

The overall design of the process is as follows; The drone after receiving GPS coordinates, initiates takeoff and proceeds to the designated location. After the drone arrives it will activate a separate onboard color camera (one for color another for black and white), and the image processing code will lock onto the specified color of the landing station and proceed to begin landing. Simultaneously running the precision movement code to prevent mid-landing drift from either the wind or other environmental factors. Once the drone is within a specified distance threshold from the landing station it will swap to the QR assisted landing code. The code will observe key points on the pattern and orient itself correctly to ensure that the cord connection will be within 4 centimeters of the landing stations connection and finish landing.

2.2 Design Objectives

The following objectives listed are portions we required or used to simplify design difficulties within the project. The objectives were set either in the project's scope or by our own decisions early within the first semester of work.

- Our drone system will need to be able to takeoff, land and deliver energy autonomously (no human interaction beyond initiating the request for energy). The language of choice was Python.
- A reliable and consistent connection method for energy transfer between the drone's payload battery and the landing station battery. We used a magnetic connecting cord.
- An easily distinguishable landing station as to limit mistakes with drone landing prematurely. Its color has a high color contrast to a majority of environmental objects and a QR pattern in the center.
- To use a lithium-ion battery for our payload as they are light-weight and high capacity battery.
- We are using a multi-staged landing procedure, comprised of image processing, precision movement and QR pattern detection.
- A future practice to incorporate is system scaling to a hub of multiple drones that take different requests. As well as the drones being capable of making multiple visits before returning to the drone hub.

2.3 System Constraints

2.3.1 Flight Locations

Due to FAA (Federal Aviation Administration) regulations drones aren't legally allowed to fly within 5 miles of an airport, limiting our testing locations as well as energy delivery locations.

2.3.2 Weather Conditions

Iowa and many other states experience inclement weather conditions at many times of the year, such hard rains, snows, or strong winds and other such conditions. Due to these conditions it severely hampers times of the year in which we were able to test our drone's capabilities and code.

2.3.3 Low On-Board Battery Life

A majority drones in today's market have relatively short battery lives, due to their high energy consumption from weight and system. Our drone has approximately a 20~30-minute battery life with a disproportionate recharge time, causing our group an even shorter window of time to test anything related to our drone.

2.3.4 Testing Location Safety Considerations

We also have to worry about safety. When we have a flying drone with four spinning blades moving at high speeds, it is possible that someone could get hurt. We have to keep in mind where and when we are flying the drone; and to always have a backup plan to turn the drone off or retrain it in some capacity.

2.3.5 Weight Limitations

The drone itself has limitations. The drone is only able to carry so much weight at a time. This demands that we keep our payload under certain weights. The drone only has so much power in its rotors so it won't be able to handle certain wind speeds. The drone also has a certain amount of precision in its movements. These are all things that we kept in mind when working to complete this project.

2.4 Design Trade-Offs

There are many different design decisions that we made throughout the duration of the project. Each decision we made had its pros and cons.

The landing station that we designed for our drone had many different components that we had to decide on. The first of which was how large to make it. The bigger it was the easier it would be for our drone to see it up in the air. However, if it got to big then it would be too hard to transport. We decided that 41.5 x 41.5 inches would be a good middle ground between being large enough for the drone to see easily as well as being small enough to be transported in an average sedan. We also chose for the color of the landing pad to be a bright orange. This was so that it contrasted well with its background. We expected that we would be testing in a park throughout the year. Therefore, the main colors we would have to contrast with are green grass, white snow, and light brown dying grass. Orange contrasts very well with green and does well with white and is passible with light brown. We expected the period of time we had to contend with dying grass was fairly low and we would be testing with green grass a lot. This assumption was correct as we did most of our testing when the grass was green.

We decided to use a magnetic cord in order to connect the drone to the landing station. This would allow the drone to not have to be exactly on top of the port and would let the cord complete the connection instead of the drone having to be exact. The drawback of this choice is that the takeoff of the drone has to be more aggressive in order to break that magnetic connection.

We chose to use DroneKit to control the drone's systems with code. The main alternative to this is Mavros. We chose to use DroneKit as that is what intel suggested to use on their drone. DroneKit had also been updated in the past year when we began so we knew it was up to date. DroneKit and Mavros both use python but some things in Mavros use C++ so we were able to avoid interfacing different languages and systems together by just using DroneKit that was all Python.

We had several different drones to choose from at the beginning of the project. There were 2 different DJI drones as well as the single Intel drone. We chose to go with the Intel Ready-to-fly drone. We had heard that it was easier to code with the intel drone. Also, DJI could update their software and break any software that we were to write and we would have to go back over everything that we had written and fix it so that it met the new software update requirements. While this is unlikely to happen, we thought that since we had the chance to change that chance to zero by choosing a non DJI drone. Also, the previous group before us had chosen the Intel drone and that was some positive reinforcement that the positives outweighed the negatives. The previous group did have issues with having all the pieces to do what they want on the drone. However, DroneKit had been updated since they had worked on it in order to allow those functions they were lacking.

We did run into issues with the Intel drone that we chose, however. The intel drone had issues with being able to connect to it from an external computer multiple times that required us to alter the config files as well and uninstall and reinstall some packages. We also had an issue on the intel drone where there were corrupted packets on the drone. We were unable to uninstall reinstall those packets because the uninstall/re-installer was also corrupted. We were unable to fix the uninstall/re-installer because the OS repair was also corrupted. We were unable to fix the OS repair because the OS updater was also corrupted. We had many issues with the drone. It also had issues with getting the code on the drone to fun the actual facilities on the drone like the motors. We had to reimage the drone in order to allow the code to control the drone. The drone also had an issue where it shut off its proprietary port that went to USB. This stopped us from doing anything because we were unable to interact with the drone directly and we weren't able to get it connected to the WIFI in order to SSH into the drone. We did find out that we could boot to a different OS mode in order to keep the USB connection going so we could interact with the drone again. The trade-offs for the drone we picked outweighed the positives. However, we quickly had a sunk cost too high to switch drones.

We chose to use a QR code in order to get the distance of the drone to the landing station easily as well as being able to get orientation of the drone to the landing station. This was good because there are already libraries out there with the work done to get the information on the coordinates of the drone. The negatives of using a QR code is that from a long distance the drone would be unable to distinguish the code from the random noise of the background.

2.5 Architectural Diagram

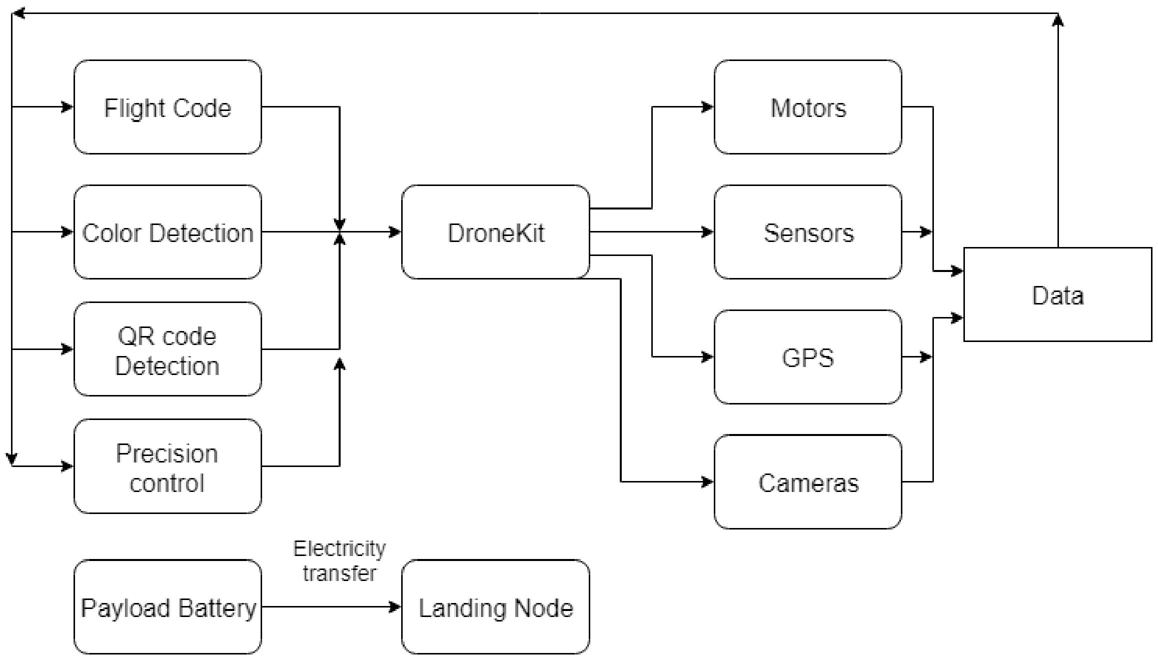


Figure 1

This is the Architectural Diagram of our project. There are many parts feeding into each other. The Flight code, Color Detection, QR code Detection, and Precision control all feed into DroneKit that interprets the code into commands to send to the drone's systems. The drone systems ie the motors, sensors, GPS, and Cameras all act on DroneKit's directions as well as send Data back to the code systems so the code can interpret that data in order to take more actions. The Payload Battery on the underside of the drone connects physically to the Landing Node (landing station) and transfers electricity through a magnetic cable.

2.6 Design Block Diagram

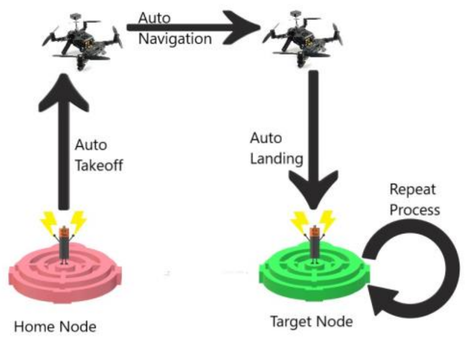


Figure 2

There are a few different pieces/modules to our high-level design. We have a home node that the drone will begin at. The drone will charge its battery here and the payload battery. It will then upon receiving a signal will takeoff and navigate autonomously to above the target node (landing station) and then the drone switches flying modes to its autonomous landing procedure. Once it has landed on the target Node (landing station) it will transfer electricity to the node until it has reached an acceptable level. Then this process will repeat with the drone either going to another landing station or back to the Home Node to refill.

2.7 INTERFACES

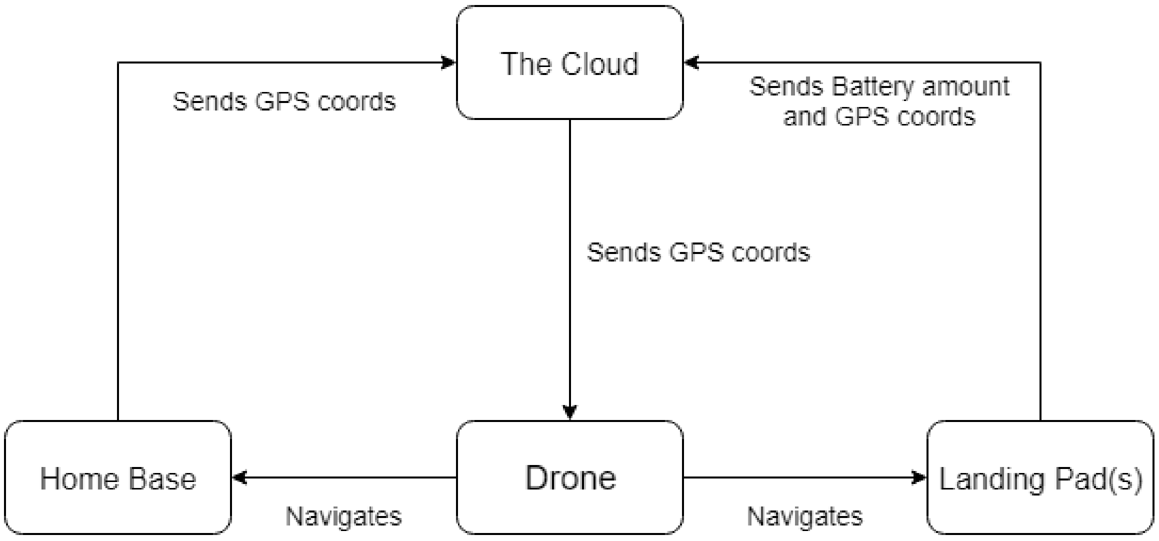


Figure 3

There are several pieces interacting in the high-level plan that need to interface with each other. The drone is at the center of everything. The drone will navigate to the Home base and the landing pad(s). The Drone only directly speaks to the cloud / server. The Home base as well as the Landing Pads(s) (landing stations) send their GPS coordinates and Battery amount to the cloud so the cloud can process the information and tell the drone what to do when it is logical to do so. The drone just takes coordinates and flies from location to location transferring energy once it arrives. We did not implement the cloud in our project as our focus was on the drone and the landing pad. However, the project is set so that it could be upscaled to more drones with more landing pads all controlled through the cloud server.

3 Implementation

3.1 Technology

3.1.1 Software Used

1. DroneKit – Open-source UAV communication tool. This is used to allow us to connect to and provide flight and docking instructions for the drone.

2. MAVLink – Communication tool that allows us to use PX4 to connect to the drone.

3. Image Processing – A program written in python to find our landing station on the ground by looking for a specific color and QR code that is placed on the landing station.

4. Python – Most of the instructional code is written in Python.

5. Ubuntu – Linux based OS, Open source software package allows us to flash our drone and construct the required software to achieve a particular task.

3.1.2 Hardware Used

1. Intel Aero Ready-to-Fly Drone – The drone which handles DroneKit information and can receive additional instructions for flight and precision control.

2. LiPo Battery – Relatively compact and powerful, can be extremely dangerous if not handled properly and should be used responsibly.

3. Battery Carriage – 3D printed box that holds the exterior battery onto the drone, and feeds the magnetic connection from the battery to the landing station.

4. Landing Station – Square space elevated off the ground for the drone to land on. It has the other end of the magnetic connection, an orange border so the image processing detects the square, and a QR code for the drone to land precisely on the magnetic connection.

3.2 Applicable Standards and Best Practices

1028-2008 Software Audits – Standard to review and audit software and hardware used in the project. It describes how to carry out a review of used software or hardware.

1625-2008 Multicell Batteries – This standard corresponds to the lithium ion polymer rechargeable battery used in the project. It ensures the safe use of said batteries.

16085-2006 Risk Management – The standard for software life cycles described in the below standard.

14764-2004 Software Lifecycle Process and Maintenance – This standard describes the maintenance required to upkeep certain software used, so it has the proper life cycle changes it needs.

4 Testing, Validation, and Evaluation

4.1 Test Plan

For our full testing plan, we will be dividing up the testing into two parts: unit testing and integration testing. Each part of these tests will need to be done manually, as we need to see the real time feedback from the drone to see if there are any problems with the different parts of the system. We wanted to know all the parts worked separately before moving on to putting it all together to ensure that we would not run into problems within each unit when integrated. We have no interface for our system or user level GUI for the program, so we will not be needing any tests for these functions. The testing plan for each functional requirement can be found below.

4.2 Unit Testing

4.2.1 Requirement 1

Requirement 1: The drone will be able to take off and fly to a specified location autonomously.

Test Case: For this requirement we want to test and see if the code we load onto the drone will allow it to take off and fly to a predetermined location without any manual control.

Test Steps:

1. Load code onto drone and activate it from ground control resource.
2. Observe how the drone flight for stability and any difficulties flying, such as swaying and wrong direction of flight.
3. Measure how far away the drone lands from the predetermined destination set in the code, measured in centimeters.

4.2.2 Requirement 2

Requirement 2: Drone will dock with landing station for power transfer autonomously.

Test Case: We want to have the drone as it is landing align itself correctly over the landing station and plug to ensure there will be a connection to deliver energy.

Test Steps:

1. Have drone already in the air over the target destination.
2. Observe how the drone aligns itself over the target as it is landing, observe any big changes over a meter or sudden drops to land.
3. Measure how far off the target the drone is in centimeters, and see if we can get footage of camera to see what used to align itself to make tweaks.

4.2.3 Requirement 3

Requirement 3: Drone will be able to deliver power from the payload battery to the landing station.

Test Case: Have the cord used to connect the payload battery connected to the payload, and connect the plug together to get a voltage reading.

Test Steps:

1. Connect one end of the cord to the payload battery, the other end will attach to the landing station plug.
2. Measure the voltage reading on the landing station side to see if there is power transfer.

Testing using the drone will be done with the payload battery and carriage attached to see how the extra weight can impact flying. We will also be testing each "leg" of the flying procedure separately to ensure all directions are functional separately. After all of these tests are completed, verified and accepted, we would then move on to integration testing. Integration testing would be much simpler, and would be as follows:

4.3 Integration Testing

Requirement: Drone will fly to, dock, and deliver power to set destination.

Test Case: We want to have the drone carry out all of the unit steps to complete one continuous system and deliver energy to a remote node.

Test Steps:

1. Send command to drone to start the operation.
2. Oversee the drone flying to the general area of landing and ensure flying efficiency.
3. Observe change from general flying to precision, watch for any sudden drops and alignment over the target
4. When drone is landed, observe the voltmeter attached to the cord that there is a connection to the payload battery.

4.4 Validation

Each part of the plan above will be extensively tested not only to ensure that we have repeated and solid results, but also to gauge what kind of the conditions the drone will work best in with our code. For each part of the unit testing, we will use the following validation criteria before we move on to integration testing:

4.4.1 Requirement 1

The drone will take off and fly to the specified area approximately. It does not have to be exact destination, within 1 meter of the target. The drone should also have a stable flight, no swaying or sudden drops when flying. There should be some telemetry data for the flight such as battery level, altitude, speed, etc.

4.4.2 Requirement 2

The drone orients itself to where it expects to land as well as the actual landing points to compare to where we set the landing point to be. Landing should take place within a couple of centimeters of the target location. The drone will have a smooth landing procedure, no sudden drops or movements.

4.4.3 Requirement 3

Voltage level reading on the landing station side of the plug is within 1V of the payload battery voltage when the payload battery is attached to the landing station plug.

After all the above validation criteria have been met after repeated tests under multiple conditions, we will then move on to integration testing where we will put the system together and observe the result. The expected result is that all of the above requirements are met in one continuous flight as opposed to the separate testing in unit testing. The specific criteria we used is outlined below.

4.4.4 Integration Validation

The drone will be able to fly to the general location of the landing area (within one meter), and then change over to precision landing without any problems with flight. The drone should then be able to recognize the color or QR code on the landing station and begin the precision landing. The drone should land on the landing station within the tolerance for the magnetic cable to connect to the receiver on the landing station and we will have a voltage reading within 1 V of pre-take off payload battery level at the landing station, ensuring a secure connection.

If all the criteria are met for the integration validation case, then we are able to call our project fully successful.

4.5 Testing Evaluation

Unfortunately, due to many drone hardware and software issues that occurred throughout the semester, we were not able to get all of our testing done. We were able to have tests for requirement 1 and requirement 3, but we were not able to test requirement 2 due to camera issues. For testing requirement 1, we gave the drone a direction to fly 10m north to the landing station from the starting point and then we observed the flight for efficiency and stability, and then measured how far away the drone actually landed from the landing station, which was the target. We then repeated the same tests for having the landing station be 10m East of the drone take off location. We found that on the tests where the drone went North, the distance to the landing station was more that our validation target, so they would not work. However, when we conducted the tests where the drone went East, we found that the distance the drone landed from the landing station decreased significantly, but we were only able to get it down to about 2 meters, so it would still not fit in our validation criteria. However, we think that with more time to test we would be able to get it to land on the landing station or within the required one meter. While we were not able to get within the tolerance of the landing for these tests, we were able to see that the drone would have a stable flight (no drops or wobbling in flight) with the attached payload. It should also be noted that due to weather, our days to test were limited so we chose a day that had high wind speeds, and that may have impacted how the drone would fly and land.

For requirement 3, testing would be much simpler. We would move the plug at a variety of speeds above the connection and from different directions and see if a connection was made using a voltmeter. We tried moving the cord at a fast speed and a slow speed in an "up-down" direction and a "left-right" direction and see if there was a connection. We found that with all tests, a secure connection was made. The payload battery discharges about 5V when connected, and we found that with all connections the voltmeter read about 4.8-4.9V, well within the tolerance outlined in the validation section. Our testing with the landing station and battery connected to the drone can be found below in figure 4.

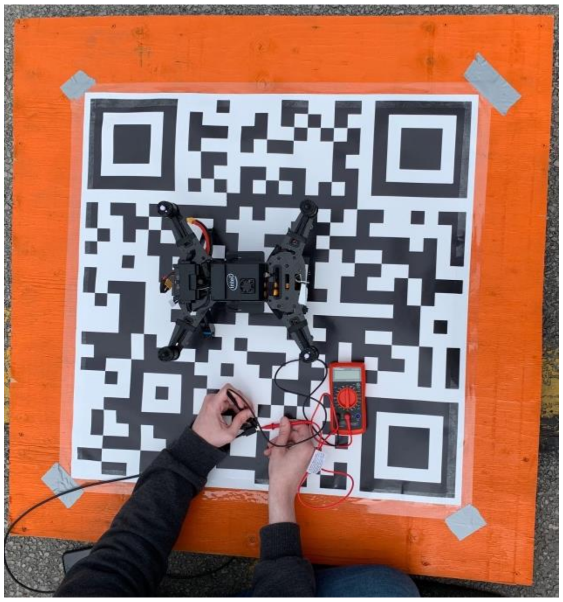


Figure 4

While testing, we noticed that if one end of the magnetic plug came in at a height just equal to or below the receiving end, the two magnets would repel each other, making connection very had to do autonomously. To solve this issue, we seated the receiving plug a little bit inside of the landing station so that the magnet on the drone side will not repel the magnet on the landing station.

5 Project and Risk Management

5.1 Task Decomposition, Roles and Responsibilities

Our project will consist of programming a drone to autonomously fly to a node, land, and then transfer power. This will be done by having our group divide and conquer most of the time. One group will mainly be focused on coding, and the other group on design and power transfer.

The design will be broken into two big parts as explained above, with the autonomous flying/programming being one section and then the power transfer being the other section. The flying will also be further broken down into two sections, one being for flying to the node and then landing on the node. This can be shown below in figure 1. The flying to the node code will be needed to be done first, as we need to make a part of the code "hand off" control of the drone once it reaches a certain distance from the landing node. The precision landing code will need to then interface with the same API the flying will use so that the transition will be seamless for when the drone goes back from the docking to the general flying.

The energy transfer group will need to develop a way that the drone can carry a payload battery that will not impede the flying of the drone, and that is light enough for the drone to carry it long distances. They will also need to develop a system that will attach and detach from a target node without the assistance of humans, so that the power transfer can also be autonomous. The group will also be in charge of making a QR code and adding color on the target node that will be recognized by the precise landing code developed by the software team. They will need to quickly learn an AutoCAD software to design the carrier, as this is something, they have limited experience in. They also need to communicate frequently with the coding group to ensure that any changes the power transfer parts do interfere with the coding. Within the design group one person will work on building a carriage for the battery, another on creating a design for the target node. After those tasks were accomplished, the group reassigned their members to working with the software team's precision landing member to figure out the QR and color recognition within the code and drone. And the other will help with drone testing and mounting everything to the landing station.

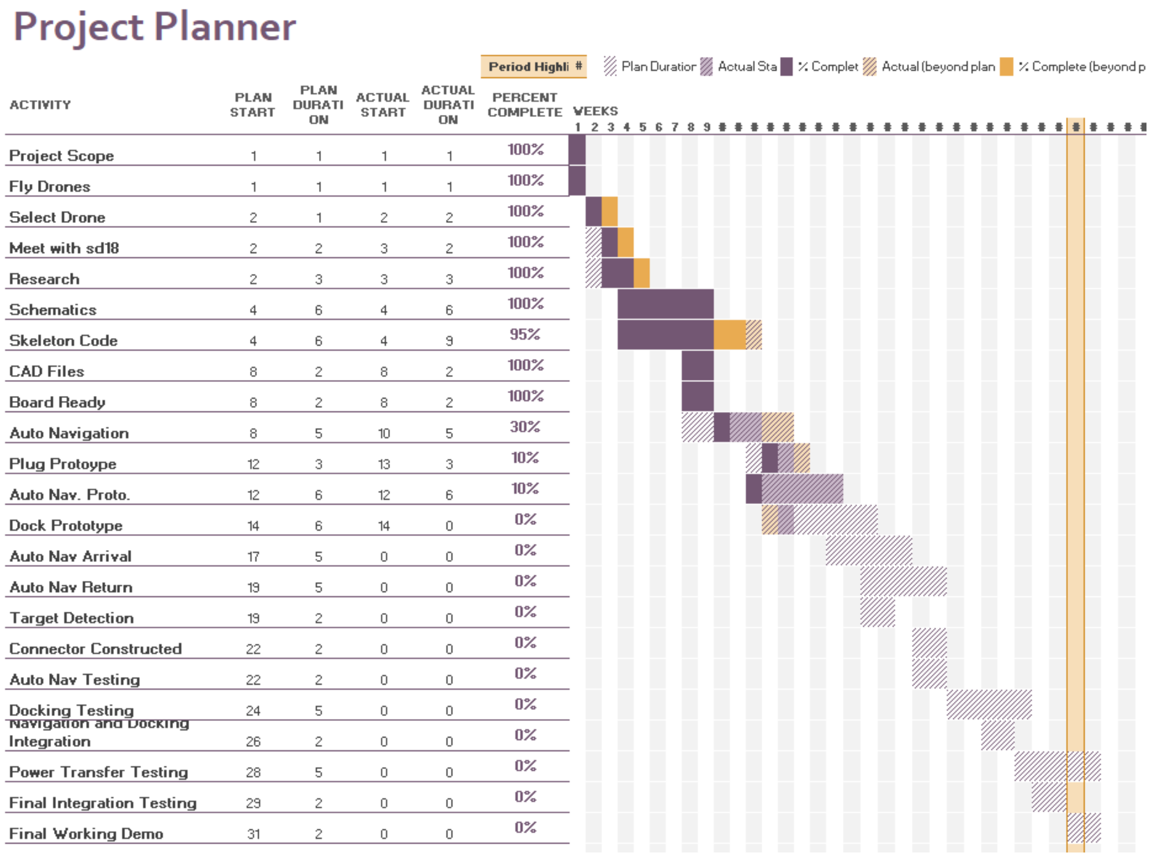
The progress we have made in the first semester of this project is mainly research, but we do have a few physical things we can present. Starting with the coding side, we have done research and tried connecting to the drone. We have all created a GitHub and connected to it so we can share our work. Our current status is we can run simulations, but not connect to the drone with an SSH port. We have since looked into resetting the drone to defaults so we could set it up the way we would like it. They have also had accomplishments like being able to test code on a drone simulation program. This semester has mainly been research for general movement paths, precision movements and alignment, dropping the drone through the air smoothly, and hovering. These topics will all be implemented next semester. We will also get the drone reset and usable next semester.

The progress made the second semester would include for the software team all the code working, but unable to integrate and test some aspects on the actual drone. The first step was to finish the OS reset that was started the previous semester. That lead to other issues: USB ports not working, SSH port not allowing connection, and camera feed not available. All of these issues were fixed by the group except the camera error. While half of the group members took on those challenges, other members of the group were performing other tasks that needed to be done in the code. The fly to location and auto navigation was able to be tested on the drone and ran via GPS coordinates that we supplied to the drone. Those pieces of the drone code are functioning. The next step would be landing, this consists of recognizing a QR code on the landing station and/or a color recognition software. The other part to landing is aligning itself with the QR code so the drone is able to line up and land properly for the power transfer. The QR code functionality was not able to be tested on the drone because of the camera issue the software team encountered after a much needed reset of the drone's OS. The software team spent many hours trying to correct the problem with no avail. However, the QR software was tested on another Linux machine with a camera and on mobile phones, and it did provide the functionality we were looking for. As for the other part of the code, aligning with the landing station that was also unable to be tested on the actual drone as it also needed to access the camera which would not provide a live feed to the drone. However that code was tested on the drone's simulation software and did work and provide the functionality that was needed to properly align with the landing station. The next part of the software team's job was to integrate the code and do further testing. Unfortunately without all the working parts on the drone, the code was not able to be integrated, because without a camera 1/3 of the code would crash and cause system failure. However the group did do more extensive testing on the individual parts of the code.

The hardware side was able to accomplish building the frame of the dock this semester. They have also began researching the best colors and patterns for image processing. This is important so the drone can use the camera to see the landing pad and align itself properly. If it cannot recognize the landing pad it will lead to landing failure. Another item that has been accomplished by this team is creating a carriage for the battery to attach. This was done in Autodesk as a file that can be 3D printed or manufactured. Overall our group is making progress toward our goal and will be ready to test more in the spring semester and finishing our project.

At the start of the second semester the hardware team was to help redesign and work with the landing station and code. As we were now adding a QR image to the landing station for the drone to detect. The landing station finished with a 5-inch border of orange color for the image detection with a 36-inch by 36-inch QR code for the drone land precisely on top of the landing station. They also tested their power equipment after getting it installed on the landing station and drone. The power transfer was successfully tested separately from the system itself by using the rechargeable battery, connecting it to the magnetic connection and successfully charging a device.

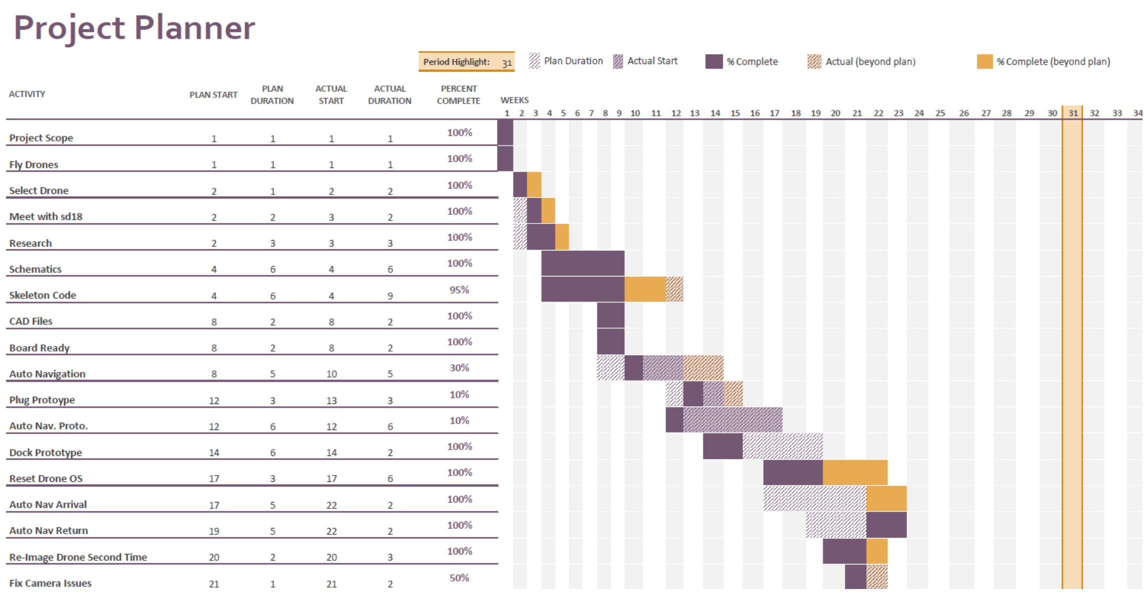
5.2 Project Schedule – Gantt Chart (Proposed vs. Actual)



ACTIVITY	PLAN START	PLAN DURATION	ACTUAL START	ACTUAL DURATION	PERCENT COMPLETE
Project Scope	1	1	1	1	100%
Fly Drones	1	1	1	1	100%
Select Drone	2	1	2	2	100%
Meet with sd18	2	2	3	2	100%
Research	2	3	3	3	100%
Schematics	4	6	4	6	100%
Skeleton Code	4	6	4	9	95%
CAD Files	8	2	8	2	100%
Board Ready	8	2	8	2	100%
Auto Navigation	8	5	10	5	30%
Plug Protoype	12	3	13	3	10%
Auto Nav. Proto.	12	6	12	6	10%
Dock Prototype	14	6	14	0	0%
Auto Nav Arrival	17	5	0	0	0%
Auto Nav Return	19	5	0	0	0%
Target Detection	19	2	0	0	0%
Connector Constructed	22	2	0	0	0%
Auto Nav Testing	22	2	0	0	0%
Docking Testing	24	5	0	0	0%
Navigation and Docking Integration	26	2	0	0	0%
Power Transfer Testing	28	5	0	0	0%
Final Integration Testing	29	2	0	0	0%
Final Working Demo	31	2	0	0	0%

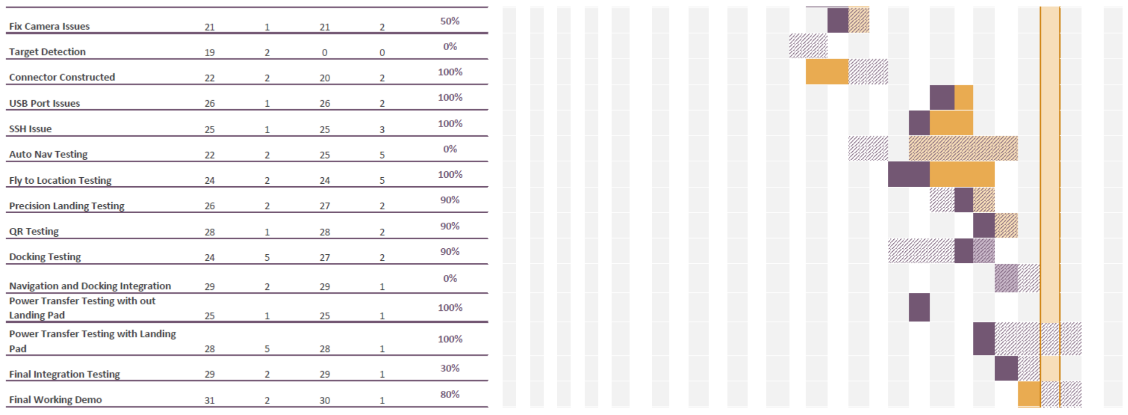
Figure 5

As you can see above that is our Gantt chart that we worked with first semester. We did get about halfway down our requirements list.



ACTIVITY	PLAN START	PLAN DURATION	ACTUAL START	ACTUAL DURATION	PERCENT COMPLETE
Project Scope	1	1	1	1	100%
Fly Drones	1	1	1	1	100%
Select Drone	2	1	2	2	100%
Meet with sd18	2	2	3	2	100%
Research	2	3	3	3	100%
Schematics	4	6	4	6	100%
Skeleton Code	4	6	4	9	95%
CAD Files	8	2	8	2	100%
Board Ready	8	2	8	2	100%
Auto Navigation	8	5	10	5	30%
Plug Protoype	12	3	13	3	10%
Auto Nav. Proto.	12	6	12	6	10%
Dock Prototype	14	6	14	2	100%
Reset Drone OS	17	3	17	6	100%
Auto Nav Arrival	17	5	22	2	100%
Auto Nav Return	19	5	22	2	100%
Re-Image Drone Second Time	20	2	20	3	100%
Fix Camera Issues	21	1	21	2	50%

Figure 6



Fix Camera Issues	21	1	21	2	50%
Target Detection	19	2	0	0	0%
Connector Constructed	22	2	20	2	100%
USB Port Issues	26	1	26	2	100%
SSH Issue	25	1	25	3	100%
Auto Nav Testing	22	2	25	5	0%
Fly to Location Testing	24	2	24	5	100%
Precision Landing Testing	26	2	27	2	90%
QR Testing	28	1	28	2	90%
Docking Testing	24	5	27	2	90%
Navigation and Docking Integration	29	2	29	1	0%
Power Transfer Testing with out Landing Pad	25	1	25	1	100%
Power Transfer Testing with Landing Pad	28	5	28	1	100%
Final Integration Testing	29	2	29	1	30%
Final Working Demo	31	2	30	1	80%

Figure 7

The two images above show how our Gantt chart adapted to finish our project in the second semester. We did run into some issues that did affect our schedule. However, since we did build in some extra time we were able to come back from that and follow through. With all the tasks that needed to be done for the project listed above you can see we did finish or attempt to finish all of them. But, as discussed earlier, we did encounter a couple of unfixable errors that meant we could never fully finish our project.

5.3 Risks and Mitigation

When our group started we knew flying the drone could have negative consequences if precautions were not taken. So to take extra precaution we always ran the code on a simulation first. After the simulation testing then we would try it on the drone but not without tethering the drone so it couldn't go anywhere we didn't want it to. We also knew that we needed to reset the drone's OS. We knew it would take time, but the reset didn't take until the second full reset. So we needed to adjust our schedule and work with that problem. As well as the reset not working, after the reset we encountered problems like the SSH and USB port not working so we also worked until those were fixed to ensure proper function of the drone. The last error was the camera not giving feed to the drone, and since we were never able to fix that, we never ran code that needed a camera on the drone.

5.4 Lessons Learned

Overall, throughout both semesters we have learned invaluable lessons. Starting with everyone being able to fly and work with different types of drones. Then specializing in the Intel Ready To Fly drone. Everyone learned Python coding and having experience in many different languages is always a plus. Along with coding there was experience with many libraries including PyBoof, OpenCV, and others. We have learned more than just about new software and the uses of drones. We have also learned how to work with a large group in a yearlong project in stressful situations. As we have found out over this project, not everything will go as planned. This second semester has really tested us as a group and team to overcome a lot of obstacles, from communication within the team to attempting to fix a drone with more issues than we could count.

6 Conclusions

Over the course of the year, we have made significant progress in our project. As a team, we have encountered many unforeseen issues in most aspects of our project, and still diligently worked on fixing these major problems. Unfortunately, we were unable to fully meet all requirements and goals set by ourselves and our clients due to these long-standing problems. Each separate component of our project (QR reading, image processing, flying to predetermined location, transferring energy from drone to landing station) was completed and everything worked in simulations, but due to these significant problems, we were unable to completely integrate these into a single system. Despite these significant issues, we were still able to create a mostly-functioning system. In the future, this project could be expanded upon by working with a different drone, since most of our issues were related to the drone itself. Additionally, if our basic system were to work on a different drone, the system could be expanded to travel to and deliver energy to multiple nodes across a given area.

7 Appendices

7.1 Notes for Future Iterations

This project has great potential for future iterations, whether in another Senior Design project, or in a professional industry situation. With a different drone, the code we have written could be very easily debugged and implemented into a completely working system. Additionally, with a greater budget, a company could develop a fleet of drones and remote landing stations, which could be used in a variety of situations. If another Senior Design group were to work on this, it would be highly recommended that they use a different drone than the Intel RTF drone that we used. We encountered significant problems with the drone throughout the entirety of the project, and despite finding and implementing solutions, these problems ended up creating too many roadblocks to create a completely implemented, working project. As such, we strongly recommend that any future group uses a different drone, possibly one of the two DJI drones models that the Senior Design program currently has. There may be some issues with transfering the code we have written to a different model of a drone, but our recommendation would be to debug the code issues, rather than the drone issues that we dealt with. Additionally, we would recommend that a Senior Design team be mostly Computer and Software Engineering, rather than Electrical Engineering, because the majority of the project is very heavily software based, which CprE and SE students focus on, not hardware that EE students would focus on.

7.2 Other Considerations

Our group encountered many more issues than we anticipated that caused our project to not be fully implemented, despite the different facets working separately. When developing timelines multiple times throughout the semester, we included additional time for testing and debugging code. Each time we revised our timeline, we were either correct or overestimated the time it would take for researching, writing code, and testing code; we completed each of these different phases in the time we allotted or less. However, where we were unable to accurately determine the time work would take was with the issues involved with the drone, explained further below. Despite all of the problems that put us at a standstill many times throughout the year, we always continued working to solve these problems, as well as writing and testing code despite the many issues we encountered that kept us from being able to completely test the various parts of our code.

7.3 Timeline of Unforeseen Problems

Below is a timeline of unforeseen problems we encountered relating exclusively to the drone, along with the steps we took to troubleshoot and solve these problems. These were entirely outside of the scope of our project, and despite meeting with professors who were recommended to us, we still had to debug the various problems without any prior experience to work off of.

November 8, 2018:
Obtained cord to connect to drone, work on drone begins
+ Put code on drone
- Code not able to run on drone hardware, only simulation

November 15, 2018:
- SSH issues (cannot SSH into drone from separate laptop)
- Code not able to run on drone, only simulation
- Troubleshoot: only run on simulation issue
- Attempt reset OS / reset

Winter Break, new semester begins

January 2019:
- SSH issues
+ SSH issues resolved
+ Email help from Zambreno
+ Code ran on drone once
  (This was a fluke that we were unable to replicate after no changes.)

February 21, 2019:
- SSH not resolved
- Code only runs on simulation, fails when run without simulation
- Corrupted packets, corrupted installer, corrupted uninstaller, corrupted OS, corrupted OS updater, corrupted OS repair
- SSH security block issues
+ SSH config admin port fixed
+ Fixed ssh
+ Fixed deep corruption

February 25, 2019:
- Port issue, drone not accepting code. Runs on simulation or fails on drone
  - Large amount of troubleshooting with port issue
+ Meet with Zambreno (Diagnosis: wipe the drone, unfixable)

March 7, 2019:
+ Re-image drone
- USB port issue (USB powers off on startup, stops you from doing absolutely anything)
- Unable to set SSH up (due to above issue)

March 11, 2019
+ Solved USB issue
+ Installation of software
- Camera issues (not able to get image from camera)

April 8, 2019:
+ Code finished
+ Drone flying
+ Tests
- Camera issues (not able to get image from camera)

April 18, 2019:
+ Code touch ups
+ Drone tests flying out in park

- Camera issues (not able to get image from camera)

8 Team Information

Kevin Angeliu – Electrical Engineer focusing in control systems or power distribution. Kevin is the chief engineer of the project.

Garth Flaming – Computer Engineer with a wide variety of experience working with large projects and being a leader. Garth is one of the facilitators in the project.

Alexandra Lowry – Software Engineer, with a minor in music technology, focusing in the entertainment business. Alexandra is the report manager in the project.

Kaitlyn Maass – Computer Engineer, with a minor in engineering sales, focusing on helping clients find technical solutions for their problems. Kaitlyn is the meeting scribe for the project.

Brendan Rohlik – Electrical Engineer focusing in power distribution or working in the power grid. Brendan is the head of the timeline for the project.

Connor Wehr – Computer Engineer focusing in network security, seeking a position in IT or something related. Connor is the other facilitator in the group.

9 References

1. "gf dvx by 3D Robotics," DroneKit. [Online]. Available: http://dronekit.io/. [Accessed: 12-Oct-2018].

2. "Intel® Aero Ready to Fly Drone," Intel. [Online]. Available: https://www.intel.com/content/www/us/en/products/drones/aero-ready-to-fly.html. [Accessed: 12-Oct-2018].