ECE 4901 Fall 2016 Final Report

Autonomous Battery Charging of Quadcopter

Team 1722:
Thomas Baietto – Electrical Engineering
Gabriel Bautista – Computer Engineering
Ryan Oldham – Electrical Engineering
Yifei Song – Electrical Engineering

Sponsor: Professor Ashwin Dani, Ph.D. Assistant Professor
Electrical and Computer Engineering
University of Connecticut

347 Fairfield Way, U-4157 ITE-Building
Storrs, CT 06269
E-mail: ashwin.dani@engr.uconn.edu
Contents

1. Introduction .................................................. 2
   a. Summary
   b. Background

2. Platform: Parrot AR.Drone 2.0 ................................ 4
   a. Choosing a Drone
   b. Hardware
   c. Application Programming Interface

3. Solution .......................................................... 7
   a. Design Limitations
   b. Software Technical Design
   c. Hardware Technical Design

4. Project Plan ..................................................... 18
   a. Budget
   b. Timeline

5. Conclusion ...................................................... 20
   a. Summary
   b. Relevant Future Work

6. References ..................................................... 22
1. Introduction

Summary

Quadcopters have a limited flight time due to current battery supply limitations. Commercially available quadcopters typically last anywhere from 20 to 30 minutes on a single battery charge. Adding more batteries is not a desirable solution because increasing payload weight will consequently consume more power, ultimately reducing battery life and flight time. Our Design Team will develop a UAV system, specifically a quadcopter, which will autonomously recharge itself. This means that the quadcopter will be able to recharge its battery when needed without any human interaction.

For this design project, we plan to develop an autonomous system that can navigate our quadcopter to the charging station and then recharge the battery. Thus, this project can be split into two separate parts. First is the software design, which will allow for autonomous flight and docking of our quadcopter on the charging station. Second, is the hardware design of the charging system that will succeed in recharging the drones battery in a timely manner. By the end of this project, our goal is to successfully design a fully autonomous battery charging quadcopter system.

Background

Unmanned Aerial Vehicles (UAV) provide aerial surveillance at an affordable cost with easy to learn controls. UAVs have grown increasingly popular over the last decade causing a high demand to continuously improve their functionality. Current UAV technology is severely restricted by battery storage capacity and requires frequent human interaction between sessions to manually recharge a battery.

A quadcopter is a class of UAV similar in design to that of a helicopter. Quadcopters are composed of four vertically oriented rotors attached to a frame, each controlled by their own individual motor. Flight is made possible by altering the thrust generated from each rotor according to feedback collected through a collection of sensors and commands. Software can be implemented to automatically send commands to the quadcopter motors based off of the sensory information collected onboard. This automated system can be developed to navigate our quadcopter to a recharging station within a close proximity. The quadcopter will then connect with the recharging station, supplying power to the battery. Finally, after being fully charged, our quadcopter will resume automated flight.
An important aspect of our project to consider is the environment that our system will be performing within. Our first test will be completely indoors in an empty room. The quadcopter will be stationed at one end of a room and the charging station will be placed at the other side. The goal is to have the quadcopter successfully locate and dock itself upon the charging station on the opposite side of the room. The drone will then recharge its battery and take off again. All of this will be done without any human interaction.

After achieving this first task, we will then have to successfully develop a more refined procedure. That is, we will then have to consider a design that works well with obstacles in the room such as furniture and more importantly a system that cooperates with the weather conditions.

For this project to be brought to real world applications, the weather must be considered or the autonomous charging design is useless outdoors. In order to move forward with this design, we will first achieve a fully autonomous battery charging quadcopter system in an empty room indoors. After solving this initial problem, we will then have a platform for future development of a system that works in all environmental conditions.

Below is a visual representation of our project.

Figure 1. Project Visual. The drone will be placed in an empty room on the opposite side of the charging station.
2. Platform: Parrot AR.Drone 2.0

Choosing a Drone

To begin the design, our team first needed to select a quadcopter to work with. The Systems Lab at UConn had two quadcopters readily available for us to use, the Parrot AR.Drone 2.0 and the Iris 3DR. We decided to choose between these two because they were free of cost and easily accessible. A comparison was made between the two drones which can be observed below.

![Comparison Chart of AR.Drone 2.0 and Iris 3DR](image)

From this chart, the most important factors are highlighted. As you can see both of the drones have an Open Source Application Programming Interface (API) providing third party developers a platform for application design. We decided to choose the AR.Drone 2.0 for two main reasons. The AR.Drone 2.0 is equipped with two onboard cameras, one looking forward in the horizontal direction and one facing downward in the vertical direction. The Iris 3DR requires an external camera such as a GoPro to attach to the payload of the drone and only has a horizontal field of view. The cameras on the AR.Drone 2.0 will be very useful for object tracking used to dock itself on the charging station. The second biggest factor was the onboard GPS that the AR.Drone 2.0 is equipped with. The AR.Drone 2.0 Flight Recorder adds 4GB of flash storage to record GPS and flight data. The Iris 3DR does not have GPS. The GPS on the AR.Drone 2.0 will be very useful for navigation of the charging station.
AR Drone 2.0

The AR.Drone 2.0 is a remote-controlled consumer quadcopter helicopter developed by Parrot. The body is made of carbon fiber tube structure and high resistance plastic. A protection hull is made of Expanded Polypropylene foam, which is both durable and light in weight. The hull provides protection during indoor flights. The propellers are powered by four brushless motors (28,500 RPM, power: 15W). Energy is provided by a Lithium polymer battery with a capacity of 1000mAh, which allows a flight time of approximately 10 minutes.

The AR.Drone 2.0 carries an internal computer, specifically, a 1 GHz 32 bit ARM Cortex A8 processor that runs a custom Linux operating system. The onboard computer and sensors allow for automatic take-off, hovering with constant altitude and landing. This is accomplished by sending feedback from the sensors to the control system. This feedback is required because the quadcopter is an unstable system without it. As mentioned earlier, the AR.Drone Flight Recorder is a mini-USB connector that adds a GPS sensor to the drone. An integrated 802.11g wireless card provides network connectivity with an external device that controls the drone. It is possible to control the AR.Drone from a Linux PC with the software designed for application developers.

The AR.Drone is equipped with different types of sensors that are used for automatic stabilization and the ability to send navigation data such as (status, altitude, attitude, speed). The AR.Drone features a 6 degrees of freedom inertial measurement unit. It provides onboard software with pitch, roll and yaw measurements used for automatic stabilization. The AR.Drone is also equipped with a 3 axis gyroscope 2000 degrees/second precision, 3 axis accelerometer +/- 50mg precision, 3 axis magnetometer 6 degrees precision, pressure sensor +/- 10 Pa precision and ultrasound sensors for ground altitude measurement. Lastly, the drone has two cameras. The
60 fps vertical QVGA camera can be used to measure the ground speed. The HD 720p, 30fps front camera can be used for video storage in real time with Wi-Fi directly on an off-board device. Due to the tolerance of all the sensors, they are used to determine the state estimation of the drone. The sensor measurements are sent through algorithms that obtain a relatively accurate state estimation. All of these onboard electronics will be useful for obtaining autonomous flight and docking of the quadcopter.

Application Programming Interface

Due to many third party developers, Parrot launched the AR.Drone open API that can be accessed for research and application purposes. It includes SDK (Software Development Kit) source code. The API does not include software that is embedded on the AR.Drone. Communication with the drone is done through four main communication services which are implemented in the SDK and listed below.

1. Controlling and configuring the drone is done by sending AT commands on a regular basis. AT commands are text strings sent to the drone to control its actions.
2. Information about the drone (status, altitude, attitude, speed) is called navdata. The navdata also includes estimated sensor measurements. This information is sent by the drone to its off-board computer at approximately 200 times per second.
3. A video stream is sent by the AR.Drone to the off-board device. The SDK includes an algorithm to decode this video for processing.
4. Critical data is communicated over a channel called the control port. It is used to retrieve configuration data such as the state of the drone. It also acknowledges important information such as the sending of configuration information.

![Network Communications](image)

Figure 4. Visual of Network Communications between the AR.Drone and the off-board computer
3. Solution

Design Limitations

The following limitations need to be realized during the design process.

1. **Network Connection.** The response time between quadcopter and PC terminal over a network can lead to crippling delays in our system. A wireless network will also be limited in range.

2. **Computational Speed.** Both the quadcopter and PC terminal will have delays in processing information. Image processing can require large amounts of processing power.

3. **Sensor Accuracy.** Sensors are designed to operate within certain limits. The sensory information is estimated and may not be 100 percent accurate.

4. **Charging Time.** Current Lithium Polymer battery technology limits the flow of current in and out of the cells. These limits mean battery charging time will be over 2 hours.

5. **Weight Capacity.** As we increase weight, battery life will decrease as well.

6. **Budget.** We are limited to roughly $1000.00 for any additional parts.

Software Technical Design

Objective

Our main goal for the software design is to develop a tracking system that will allow the drone to autonomously navigate and dock itself on the charging station. We decided to approach this problem by breaking it up into two separate parts. Our first objective is to design a system that will allow the drone to detect and fly in the general area of the charging station. Once the drone is hovering above the charging station the second objective is to design a control system that will steadily and accurately dock the drone on the charging station. In order to achieve autonomous flight, we must write a set of functions for the drone that commands its movement and tracking. These functions will be written in Python using Robot Operating System (ROS) which is compatible with Ubuntu (Linux based operating system).
Robot Operating System

ROS is a collection of tools, packages and libraries that act as a flexible framework for writing robot software. All of the libraries and packages are open source so it is free and open to the public. ROS consists of nodes and topics. A node is a block of code that performs a desired task. There are publisher nodes that will continuously broadcast a message and there are subscriber nodes that will continuously receive messages. It simplifies the programming of the drone because each node is written to compute one specific task. Communication between nodes allows each of these separate tasks to work together in accomplishing the overall goal of the system. This communication is obtained through topics. Topics are named buses over which nodes exchange messages. ROS contains many open source implementations of common robotics algorithms which will act as a useful structure for our software design.

Navigate the Charging Station

As mentioned before, the first objective is to autonomously fly the drone in the general area of the charging station. To accomplish this we will make use of the AR.Drone Flight Recorder to implement a GPS system. The AR.Drone Flight recorder is an external device that connects to the drone through a USB port, pictured below.

![Figure 5. AR.Drone Flight Recorder](image)

The charging station will be placed on the opposite side of the room to the drone. The coordinates of the charging station will be pre-determined and act as a waypoint. Once this waypoint is set, the drone will be programmed to map out the room using GPS and detect this waypoint. Then motor commands will be sent to the drone to navigate and hover above the charging station. The GPS on the AR.Drone has an accuracy of +/- 2 meters. Thus the drone will be hovering within 2 meters of the charging station.
Docking the AR.Drone

Once the drone is hovering above the charging station, our objective is to design a control system that will allow the drone to accurately descend and land on the charging station. In order to accomplish this we will need to design an image processing based tracking system utilizing the vertical camera on the drone. There will be a colored tag placed on the center of the charging station. The drone will be programmed to identify this tag with a tag detection algorithm. Once the drone detects the tag on the charging station, we will need to implement a PID control algorithm that will send motor commands to the drone. The motor commands will steadily descend the drone and the PID controller will work to keep the tag in the center of the camera’s view. Once the drone is docked on the charging station, the drone will be programmed to wait and take off again once the battery is charged to the desired amount.

Tag Detection Algorithm

Image processing can be achieved by developing four ROS nodes and allowing for communication between these nodes with ROS topics. Below is a brief visual of how the tag detection algorithm will work.

Figure 6. Visual of tag detection algorithm. The blue boxes represent ROS nodes and the white boxes represent ROS topics.
As you can see from Figure 6, we need to write four separate blocks of code that each have a specific task. When these four nodes work together they will achieve a successful tag detection and tracking system. The four nodes are:

1. **Camera Node**: The camera node obtains a video stream from the camera and converts this video to a ROS readable format.

2. **Visual Perception Node**: The visual perception node extracts a set of pixels defining the object we want to track and publishes the coordinates to the Region of Interest topic. This is done using a blob detection algorithm where we simply set the parameters to define the object we want to track.

3. **Head Tracking Node**: The head tracking node uses a PID controller to compute movement commands that keep the target in the center of the camera’s view. These commands are published to the Motor Commands Topic.

4. **Motor Control Node**: The motor control node subscribes to the Motor Commands Topic and maps movement commands to be performed by the quadcopter.

**PID Controller**

A Proportional Integral Derivative (PID) controller will be implemented in the design. A PID controller is a closed loop controller that continuously calculates an error value as the difference between a desired setpoint and a measured process variable. The objective of the controller is to apply corrections to minimize the error using proportional (present), integral (past) and derivative (future) terms while continuously sending feedback. A visual of the PID controller is shown below.

![Figure 7. Visual of PID controller](image)
The tag detection algorithm will place a virtual box around the tag on the charging station. The center and radius of the tag will be computed and returned. The PID controller will take input from the navigation data and the vertical camera. Based off of the quadcopters current state and its desired target the PID controller will provide feedback and allow the quadcopter to adjust its positioning to keep the desired tag in the center of the camera’s view. The PID controller will send motor commands to the drone, allowing for a smooth and accurate landing on the charging pad.

Summary

The goal of the software design is to implement a system that will allow the drone to navigate and dock itself on the charging station when the battery is low. First we need to get the drone to identify and fly over to the charging station when the battery drops below 25%. This will be done using GPS, allowing the drone to map out the room, detect the charging station waypoint and send motor commands to fly within 2 meters of this waypoint. Once the drone is hovering above the charging station, the second objective is to get the drone to land on it. This will be done using image processing, allowing the drone to detect the tag on the charging station. Once the tag is detected by the drone, a PID controller will be used to send motor commands that will descend the drone until it is docked on the charging station. All of this will be processed with an off-board computer that is connected to the drone through Wi-Fi. However, after we accomplish this, we will then run tests with the drones on-board computer and determine if this software system can be processed completely on-board. This procedure is completely autonomous so there will be no human interaction throughout the process.

Hardware Technical Design

Charging Station Requirements

Objective

There are two primary goals to our hardware development.

1. **Design of a Docking Station**
   - Create a docking station to transfer power to the quadcopter.

2. **Design of Battery Charging Circuit**
   - Recharge onboard lithium battery pack in a time efficient manner.
**Station Design Requirements**

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Consistent and Dependable</td>
<td>- High tolerance for landing error.</td>
</tr>
<tr>
<td></td>
<td>- Repeatable docking.</td>
</tr>
<tr>
<td>Easily Accessible and Detectable</td>
<td>- Drone must recognize station.</td>
</tr>
<tr>
<td></td>
<td>- Station must be easily docked with.</td>
</tr>
<tr>
<td>Safe and Optimized Power Transfer</td>
<td>- Battery charging must be controlled.</td>
</tr>
<tr>
<td></td>
<td>- Charging rate must provide optimal operation time.</td>
</tr>
<tr>
<td>Fully Autonomous</td>
<td>- System must work without any human input.</td>
</tr>
</tbody>
</table>

We must develop a charging station that allows for acceptable tolerance to allow the AR Drone 2.0 to dock on the charging station autonomously with a high amount of success. Fundamentally our project is to design an autonomous system. We want to make sure there is no human interaction. This is divided into two parts. First, create a docking station to transfer power to quadcopter. Second, Recharge onboard lithium battery pack both safely and efficiently.

**Choosing a Charging Method**

To choose a charging method we first researched available options to us.

1. **Inductive (wireless) power transfer**

   **Problem:** After initial research we found that inductive charging was highly sensitive to displacement. The displacement could lead to an insufficient energy transfer and cause the drone to not receive any charge. Additionally Inefficient/Low power transfer rate.
2. **Battery Swapping System**

**Problem:** A battery swapping system would allow for instant battery recharging. This instant recharge design though also requires the use of many different parts. The number of parts necessary would cause a great increase in unreliability.

Finally looking at a conductive power transfer method, we decided this would be best.

**Conductive Power Transfer:**
1. Provides consistent power flow compared to inductive charging.
2. Provides reliable power charging compared to both battery swapping and inductive.
3. The station is easily reproduced for multiple charging station units.
4. Additional design features can be easily implemented while using this design when compared to other methods such as battery swapping.

<table>
<thead>
<tr>
<th></th>
<th>Reliability</th>
<th>Power Transfer Rate</th>
<th>Reproducible</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inductive</td>
<td>RED</td>
<td>RED</td>
<td>GREEN</td>
</tr>
<tr>
<td>Battery Swapping</td>
<td>RED</td>
<td>GREEN</td>
<td>RED</td>
</tr>
<tr>
<td>Conductive</td>
<td>GREEN</td>
<td>GREEN</td>
<td>GREEN</td>
</tr>
</tbody>
</table>

Fig 8. Comparison of charging stations at a glance.

**GREEN** = Fits within design needs.  **RED** = Problematic.

Conductive charging is clearly the optimal choice in this design. Conductive charging fits all critical design criteria to creating a successful charging station.

**Design of the Charging Station**

To begin the design of the charging station we first created a basic flow diagram that we would be following.

![Flow Chart](image)

Fig 9. Station Design Flow Chart
The charging station first requires an input power source. To create ease of access and reproducibility we will be implementing the station for use with any AC power outlet at the national standard of 120V. This AC voltage will then need to be converted to DC before the battery can be charged. This step requires a typical AC to DC rectifier.

The AC to DC power rectifier will consist primarily of two key parts.
1. Transformer
2. Diode Bridge

The rectifier is to be built after all other hardware development stages as power supplies can be supplemented for the current testing of the system. The resulting output of the AC to DC conversion, as seen in the Station Design Flow Chart above, will lead to a battery control circuit control. This circuit will control the flow of current that is being supplied to the battery through four separate contact points.

For the description of the charging station, refer to the following diagram.

1. Battery control connects to 4 copper coils aboard a wooden frame.
2. The spacing of the copper coils will match the cross-section spacing of the quadcopter.
3. Copper coils will have a PVC funnel enclosed around it to increase landing accuracy.
4. Copper contacts will be attached to each leg on drone, located under each rotor.

5. Copper contacts on AR Drone are wired to battery. Contacts cannot be interchanged with copper coils on the station.
For initial development stages we plan to have the charging station only operate in ideal conditions. As we complete more of the design we will add weather retardant features to our charging station.

Also to note, the charging station will include easily recognizable properties for the quadcopter to be able to detect. The funnels around each copper coil located on the wooden frame of the station can be color coded. This color separation will allow the vertical camera on the AR drone to line up the appropriate contacts and guide itself into place.

**Lithium Battery Charging**

**Cell Balancing**

The AR Drone 2.0 uses a 3 cell lithium polymer battery. The battery is located centrally within the drone and is accessed by removing a protective outer casing. To charge the battery through our charging station we must connect copper wiring to each charging lead. Each leg on the quadcopter is attached with a copper contact that is then wired to the battery. Each connection will have a specific cell that is being balanced charged. For this reason the AR Drone must connect at the same orientation every time it lands.

The figure below is a visual representation of lithium battery balancing.

![Battery Cells Diagram](image)

**Fig 12. Battery Cells**
To balance the cells, a circuit was constructed to control the flow of current in and out of each cell. The following figure depicts the schematic design that was created to do this procedure. Pspice ORCAD simulation is located on the left and EAGLE PCB schematic is located to the right. These schematics have been tested to work within a simulation.

![Battery Balance Schematic](image)

The schematic located above will charge the lithium polymer battery at a constant current. This mode of charging is the fastest way to charge the battery and will as a result maximize operation time.

**Maximizing Operation Time**

In order to maximize the operation time which indicates we want to short the charging time and increase quadcopters flying time. The figure shown below[5] is the characteristic graph of typical LiPo Battery charging. In order to charge our battery in about an hour, we want to keep charging our battery in the red region shown below.

Reasons are as followed.

- If we charge the quadcopter below 15%, it will reduce the life of the battery which againsts our purpose to maximizing the operation time.
- When the battery has been charged about 80%, it requires another 2 hours to fully charge the battery.
- During the 80% to 100% charging period, the battery will also increase the temperature dramatically. But we want temperature be remained under 65 Celsius.
- We decided to charge the battery from 15% to 80% which will maximize our operation time.
Parrot AR drone Battery selections

In order to maximize the flying time that we compared flying time and Price for different battery capacity. To keep in mind that we will only charge the battery up to 80% the actual flying time will be shorter than the chart shown below. Our original plan was using 1000mAh battery capacity because it is come with the quadcopter. But we realized that the actual flying time will be not enough for our quadcopter to do its mission dock locate the charging station and dock on it. We decided to purchase a 2000 mAh capacity battery which will actually have an about 18 minutes flying time when we charge up to 80%.

<table>
<thead>
<tr>
<th>Flying time</th>
<th>Battery capacity</th>
<th>Estimate Price (dollars)</th>
</tr>
</thead>
<tbody>
<tr>
<td>8-10 minutes</td>
<td>1000 mAh</td>
<td>$79.99</td>
</tr>
<tr>
<td>12-18 minutes</td>
<td>1500 mAh</td>
<td>$42.34</td>
</tr>
<tr>
<td>20-25 minutes</td>
<td>2000 mAh</td>
<td>$23.98 (Variable)</td>
</tr>
</tbody>
</table>

Figure 15. Comparison chart of different battery capacity
4. Project Plan

Budget

For this project, our sponsor is willing to spend 1,000 dollars. The software design only requires the expense of the AR.Drone Flight Recorder. We will be using open source software that is free for public use. The AR.Drone is provided by the Department of Electrical and Computer Engineering, thus the only development cost will come from the hardware design of the battery charging station. Below is a table that outlines our spendings. We plan to spend around 300 dollars which will leave 700 dollars left over in case we need to purchase any materials in the future.

<table>
<thead>
<tr>
<th>Item</th>
<th>Cost</th>
<th>Quantity</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Copper wiring</td>
<td>$20</td>
<td>1</td>
<td>$20</td>
</tr>
<tr>
<td>Copper sheet for contacts</td>
<td>$10</td>
<td>1</td>
<td>$10</td>
</tr>
<tr>
<td>Neodymium magnets</td>
<td>$10</td>
<td>1</td>
<td>$10</td>
</tr>
<tr>
<td>Battery (2000mA)</td>
<td>$40</td>
<td>1</td>
<td>$40</td>
</tr>
<tr>
<td>PVC</td>
<td>$20</td>
<td>1</td>
<td>$20</td>
</tr>
<tr>
<td>Flight Recorder (GPS)</td>
<td>$90</td>
<td>1</td>
<td>$90</td>
</tr>
<tr>
<td>Circuit components</td>
<td>$50</td>
<td>2</td>
<td>$100</td>
</tr>
<tr>
<td>Estimated Total Budget</td>
<td></td>
<td></td>
<td>~$300</td>
</tr>
</tbody>
</table>

Budget: ~$1000
Estimated Remaining Budget ~$700.

Figure 16. Budget Layout

The current budget plans to be well within our given constraints. Our budget was calculated in terms of only completing our goal for this project with one charging station. Further development such as multiple charging stations will lead to an increase in our expected spending. Spending is expected to change as progression is made through the design.
Solving our problem includes numerous amounts of planning, research, design, development, and testing. This is a dense project and there are many parts that must be handled separately. To begin, we start by researching the two sides of our project and possible approaches that we can take. After deciding the path that we want to follow for designing a charging station (conductive charging) and autonomously flying and tracking the station with the drone (image processing and GPS), we went on to figure out what algorithms we needed to control the drone and came up with a part list for the charging station.

The algorithms to autonomously fly the drone using scripts in ROS and Python should work theoretically. We must now implement this code to the drone and finish developing and testing it on the drone. An outline for image processing using ROS nodes and online packages must also be implemented, developed and tested.

The outline for the hardware design was also created throughout the semester. Once we came up with an approach to designing a station and a part list, we made schematics for our the station will be designed. Our team will then order parts and put together the station to get ready for testing in the future.
5. Conclusion

Summary

Our team was assigned to customize a commercially available quad-rotor UAV to demonstrate autonomous docking of quadcopter to a charging station. The drone must take off and fly itself autonomously. Once the battery life reaches a critical level, it will then fly to a charging station. The drone will dock itself to the station and charging the battery until it is at an acceptable level. Once done charging, it will take off and fly autonomously again. There are multiple ways to solve this problem, with the AR Drone supporting an open source API and the charging station having multiple design implementations to choose from.

To achieve autonomous flight, we will be using ROS to make the drone control itself. ROS nodes are created to implement scripts for the drone using Python. A Flight Recorder Module for GPS will be used to get the drone to general area of the station. The previously mentioned ROS nodes are also crucial in docking to the station. We will be using image processing that sends data to the drone ensure precise docking to the station. There are many available open source packages that we can use to implement these designs, which is why the open source API is useful.

There were many different approaches that we could have went with the charging station. We decided to go with conductive battery charging as our method or recharging the battery. It provides very little disadvantage in comparison to other methods such as inductive and battery swapping. Using copper wire coils connected to metal plates, the station will provide power to the drone once it is docked and connected to the contact points. When designing the charging station, we must keep into consideration balancing the battery cells so that they are neither overcharged or undercharged. Charging the battery safely is expected to take about an hour, so long as temperatures remain under 65 degrees Celsius. The schematic for the station has been put together.

This project is outlined for us, and it is time to write the necessary scripts for the drone to function autonomously as needed. Testing for GPS accuracy, reading sensor data from the drone for image processing, flight control must be developed. It is also time to start putting together the station, designing the circuit making sure that it functions properly once designed.
Future Work

As mentioned earlier, a significant aspect of our project to consider is the environment that our system will be performing in. Our first design will be targeted for testing indoors. The quadcopter will be stationed at one end of an empty room and the charging station will be placed at the other side. After designing a system that successfully works in this setting, we will have developed a platform for further research with an autonomously charging quadcopter in different environments. That is, we will then have to consider a design that works well with obstacles in the room such as furniture and more importantly a system that cooperates with outside weather conditions. Throughout the research process our team discussed the possibility of multiple charging stations, allowing the drone to detect the closest one. However, due to our current budget and time constraints for this project, we are only considering a single charging station that will be used to recharge the drones battery. After conducting research, we found that a SLAM algorithm can be used to solve these problems.

Simultaneous localization and mapping (SLAM) is an algorithm that consists of two main parts. This algorithm constructs and updates a map of an unknown environment while simultaneously keeping track of the drones location within this map. The first part is localization, where the drone uses its vision processing and onboard sensors to compute its current location in space. The second part is mapping, which calculates the drones environment and the location of any obstacles. The global maps and paths inform the drone of a decision after observing obstacles. For example, the drone observes a wall with the SLAM algorithm and can then decide whether to go forward and hit the wall or turn around. The more obstacles in the room or outside environment increases the accuracy of the mapping because it has more data to read and make decisions based off of.

The SLAM algorithm can be directly applied to the AR.Drone 2.0 and this project when there are obstacles in the room. Since there are no distance-detection lasers onboard the AR.Drone 2.0 location is determined using tags for certain landmarks which are processed with the onboard cameras. A video of the drone flying past the tags is recorded from the front-facing camera. After the flight is recorded, the video is processed through software, which recognizes the distance and orientation of all the tags the drone encounters. SLAM algorithm allows for multiple target tracking which would allow for the recognition and tracking of multiple charging stations. Lastly, in extreme weather conditions where there is no GPS signal, SLAM can be used to create a map for itself and maneuver around obstacles to find the desired charging station. Once we have completed our desired task for this project, if there is still time, we will continue to do research on SLAM and conduct testing with these scenarios.
6. References

1. https://www.amazon.com/Parrot-AR-DRONE-2-0-1500mAh-Battery(dp/B00DAL5GD2


3. https://www.amazon.com/Kastar-2000mAh-Upgrade-Ar-drone-Helicopter(dp/B01N2NX6PC/ref=sr_1_1?sr=1-1&keywords=Parrot+AR.DRONE+2.0+-+2000mAh+LiPo+Battery
