DEC1705
Chen/Geiger
Dustin Reed: Communication, Justin Howe: Team Leader, Jeffery Schons, Chidike Ubani: Webmaster, Avanish Kuntla
dec1705@iastate.edu
dec1705.sd.ece.iastate.edu/index.html

Revised: 12/5/2017

Contents

1 Introduction

1.1 Project statement

We have created a proof of concept design for delivering energy using a drone based charging system. Our system replaces the potentially difficult or ungainly wiring required to transfer energy with a more flexible way of transporting energy.

1.2 Purpose

Drone Energy Delivery revolves around the idea of transferring energy from where it is cheap and easily available, like a wall outlet, to places where energy is expensive or difficult to obtain. An extreme case of this would be a drone delivering a charge to a node that is in a hard to reach or in a very remote location. The use of a drone to transfer energy allows adaptability and can be scaled up to be used in such extreme examples.

1.3 Goals

Our goals for the project were:

- Be able to fly the drone
- Communicate effectively with the drone
- Remotely command the drone to fly and land at a node
- Autonomously and precisely land at the node
- Be able to deliver a charge to the node

2 Deliverables

We have delivered both software and physical products.For software, we have created an Android app that will control the drone. The app runs on an Android device, such as a smartphone. The app is written in Java, using the SDK provided by the drone manufacturer, DJI. The SDK provides methods for controlling the drone's flight, pointing and retrieving video from the camera, retrieving GPS coordinates, and much more. The app controls the drone using Waypoint Missions, which are provided in the SDK. When the drone arrives at the landing pad, it will capture a video stream with it's downward facing camera. The app sends this stream to an OpenCV implementation, which analyzes the video and identifies the location of our landing pad.This generates an "error" which is a representation of how far away the drone is from the landing pad in the X and Y coordinate system. The error is passed to a PID controller, which uses the drones flight controller to position the drone directly above the landing pad. Reference Appendix II for more information on how we reached this design.

We created a system of contacts on the drone and node. These contacts are broken out USB connectors which are connected to battery chargers. When the drone lands on the pad, a connection is made between the onboard battery and the USB battery charger. The battery charges the node through this connection. Reference Appendix II for previous iterations of the charging apparatus.

Finally, we have created landing pads. These pads have a unique pattern on them that is distinguishable from anything else that might be in the field. These patterns are used by our OpenCV implementation to align the drone over the landing pads. Our pads are made out of plywood, but realistically they should be heavy and durable enough to be resistant against weather conditions. For more information on how we chose the design, reference appendix II

3 Design

3.1 Previous work/literature

Previous work in the problem field of “deliver a payload via autonomous drones” foremost includes Amazon’s Prime Air project. Prime air drones include the standard 4 rotor design for controlled vertical takeoff and 2 horizontal rotors for speedy flight over a distance. The Prime Air drones use a downward facing camera to identify a vision target to aid in precision landing. These vision targets are supplied to users of the service and placed outdoors in a well lit environment.

There has been work done into placing homing beacons at nodes, as detailed in the patent in the Appendix III, link 1. The technology uses GPS to go towards the node until the drone picks up the homing beacon signal. Then, it uses the beacon to precisely land. The beacon is accurate, and a lot easier to implement then our solution. However, it is fairly expensive to buy beacons like this, so it was not an option for the scope of our project.

We also looked at alternate ways to do the charging, such as wireless charging. Link 2 in appendix III showed a project that used wireless charging. The coils they used to make it possible were rather large, and required direct access to the drone battery. Not only would they be hard to carry, the commercial drone we have restricts our access to the internal battery, so this is not feasible. Their method of guidance and landing is very similar to the one we ended up using.

One other thing we looked at was the IBM Internet of Things management suite. Using this suite or something similar, we could have a backbone of a system that talks to nodes, gets their power, and sends drones out to recharge them. We thought about making a similar system, but

it would be much easier build in on the back of an established network if we were going to actually make a drone energy delivery system beyond a prototype.

We have the ability to develop an application which can be used to control our DJI drone using the DJI Android SDK in order to achieve autonomous flight.

For precise landing we used OpenCV. This allowed us to find our landing location using image processing. The node has a distinct pattern which we will be able to recognize using color and then shape recognition.

Charging will be done with a lithium ion battery pack using 5v USB charge controllers and corresponding batteries.

3.2 Validation

Our project was validated through a series of mission goals.

1. Correctly identify a node that needs charging
    a. Receive a request via the server including the nodes current power level and coordinates
2. Navigate to the node
    a. Fly from the base station to within 1 meter of the node detectable via GPS
    b. Navigate the remaining meter via a downward facing camera and vision targets on the node. Land with an error of +/- 5cm
3. Charge the node
    a. Make electrical contact with the node
        i. Detect current battery level of node
        ii. Deliver energy
4. Proceed to the next node or return to the base station if low on energy

4 Project Requirements/Specifications

4.1 Functional

The functional requirements of the project are:

- The drone must be able to fly to a location, charge the node, and return to the base node
- The drone must be able to acquire the node's coordinates using GPS
- The drone must then fly to node using the GPS coordinates
- The drone should detect the node's landing area after reaching its destination
- The drone must be able to land on the node
- Upon landing on the node, the drone must be able to charge the node
- The drone must be able to return to its home base after charging the node
- The drone's flight patterns must be managed by an android application
- The drone must use image processing to detect its landing site

4.2 Non-functional

The non-functional requirements of the project are:

- The drone needs to be able to be tracked using a radio tracker
- The network of nodes needs to be connected to database in order to arrange flights
- The drone must be capable of autonomous flight using GPS data
- The application for controlling the drone should be able to show the surroundings of the drone using its camera.
- A predetermined symbol for image processing must be applied to the node

4.3 Standards

- The protocol for writing code will be using Git to share and collaborate on code
- Git is approved by IEEE whose ethical standards we will be following.
- These standards are applicable to our projects as we will ensure that our drone operates safely and does not cause harm to the environment it travels in, property it comes in contact with, and people it encounters.

5 Challenges

- Cost of drones is high
- Many commercial drones don't carry payloads well
- Legal issues of flight near the airport and on campus
- Limited knowledge in electronics or power transfer
- Limited ability to access the drones sensors directly
- Documentation of drone SDK not very good
- SDK code is constantly upgraded, we have to upgrade app each time
- Changing weather conditions make it hard to test
- The code libraries we chose didn't integrate together easily

6 Timeline

Our overall goal is the creation of a system to transfer energy from a home to a node by the end of our senior design courses.

6.1 First Semester

Our main focus for the first semester was to get the drone in the air, to land, and to do basic navigation. We then increased the precision of landing.

- 3/2/17 Become comfortable flying the drone
- 3/16/17 Have meaningful communications ability with the drone automated
- 3/30/17 Land the drone using an automated system, evaluate sensors/changed required
- 4/20/17 Have a basic sensor package under development for precise landing

6.2 Second Semester

The second semester had us focusing on the precision landing and charging elements of the project, along with potential add-on components such as web interfaces.

- Week 2 Precise landing is consistent and reliable
- Week 4 Be able to make electrical contact with the node and base reliably
- Week 6 Transfer a charge from the home to the drone reliably after automatically landing
- Week 8 Transfer a charge from the drone to the node reliably after automatically landing

7 Conclusions

We have created a system to transfer energy to hard to access locations or otherwise places that it is not feasible to run a cable. We have shown how a drone can be used to achieve this. Since the drone relies on GPS and visual analysts to find the node, it can be implemented anywhere. As long as the drone has a big enough energy supply to get to the node and back, this system could be deployed over any distance. We have shown that there is a way to deliver power to anything, anywhere it is, as long as it has a charging pad.

8 References

https://developer.dji.com/mobile-sdk/
http://www.physics-and-radio-electronics.com/electromagnetics/electrostatics/methods-of-charging.html

https://www.amazon.com/Amazon-Prime-Air/b?ie=UTF8&node=8037720011

Implementation Details

Our final implementation of the Drone Energy Delivery system consisted of the following components: a landing pad, an electrical contact system for the landing pad, an electrical contact system for the drone, a drone, and an Android application that provides flight instructions to the drone and applies visual analysis to precisely land on the landing pad. In our implementations model we assume that we have complete control over the landing pad or are able to provide specifications for the design of the landing pad. Based on this assumption, we designed a base landing pad that would be present at the site of every node. The landing pads have a predetermined pattern consisting of blue circles of varying shades in order to prove the landing process. These landing pads also contain an electrical contact system that is wired to receiver of the energy we are delivering. In practice this will often take the form of a sensor, however for testing purposes we have used lithium ion batteries instead. The drone itself has electrical contacts attached to its skids. These contacts are then wired to an external battery that the drone carries. Due to our assumed control of the landing pad we have arranged the electrical contacts of both the landing pad and the drone to be oriented north to simply our landing procedures. The drone that we utilized for this implementation is the DJI Phantom 3 Standard. This particular drone was chosen because it provided us access to the DJI Android SDK, relatively compact, affordable, and good battery life. Our main motivation for getting this drone was the SDK as it had many features such as flight stabilization, basic camera control

and output, location tracking, and readily usable flight control built into it. The final component of our system was our Android application. The app's main purpose is to provide the drone with instructions on how to reach a desired node. To do so we utilized the waypoint missions in the DJI SDK to utilize the drone's GPS and set its behavior as it traveled to the node. The other purpose of the Android app was to improve the precision of the drone as it prepared to land on the node. To do so OpenCV, a pair of PID controllers, and flight commands were used to orient the drone above the landing pad. After aligning itself correctly, the drone lands and makes electrical contact with the node's landing pad.

Testing Process

The initial testing process for the drone energy delivery system centered around around making sure the drone was able to accurately find waypoints using GPS signals and testing its behaviour as it travelled to the waypoint and landed on them. The tests used in this phase typically followed the pattern of: declaring a set of coordinates as the home base, physically moving the drone to another location and then storing the new location as node. We would repeat the node storage when testing variations of the waypoint missions. Then we would launch the mission to observe the behavior of the drone and make corrections as needed. Successfully implementing the waypoint missions, we moved on to testing programmatically driven flight control. In these tests we would fly the drone manually and set it to hover at a given altitude. Then we would attempt to use our Android application to give input to the drone. Similar to the waypoint missions, we would observe the behavior of the drone and then troubleshoot the behavior using the DJI documentation.

For the visual analysis portion of our implementation, we began creating a standalone Android application that would recognize red circles as that was our landing pads initial target. Once this was accomplished we applied varying filters and optimized the framerate of the app. After each modification the red circle test was applied to the application. When we began combing the visual analysis with the flight control, we used a test similar to the flight control test. In these tests however, we tried to have the drone land on the node based on recognizing the red circle instead of using the autoland feature. When the PID was being integrated we used the same test, except the landing pad had a different blue pattern instead. We found that the use of blue made the target easier to locate in various lighting than the red that could be emulated by evening sunlight. The PID was tuned by first hand-held testing where the quadcopter is held instead of flown to check the proper reactions of the PID controllers to the motion of the quadcopter. After making sure we would send propper commands to the quadcopter (at least in the right directions) we implemented the control of the quadcopter by the PID controller. Initial testing proved that the PID was very aggressive, and caused overshoots and oscillations in flight. Dampening the motion and tuning the constant values resulted in the correct reactions and tracking of the quadcopter in flight.

APPENDIX I: OPERATING INSTRUCTIONS

In order to use the drone energy system prototype, you must first charge the battery attached to the drone. Then, take the landing pad out into the field and secure it. Make sure it is secure in the ground, or the rotors of the drone will blow it away. Make sure the contacts are properly connected to the landing pad. Then, turn on the drone and the controller, and open the app. Make sure you have data turned on or are connected to a wifi source. After the app registers with the DJI database, connect to the ad-hoc wifi network generated by the drone. Once that is done, you can proceed past the opening splash screen. Take the drone, and place it on the pad. Using the drone's internal GPS, press the mark waypoint button on the app to save the GPS coordinates of the landing pad. Then, place the drone at least 5 feet away from the pad, and press launch. The drone will fly up, over to the pad, then lock on and land on the pad. If the contacts are properly attached to the drone and the landing pad, the battery on the drone will charge the battery on the landing pad.

APPENDIX II: PAST ITERATIONS

Many of our designs were different from when they started. Our navigation was originally done using Google Maps. We would take a point on the Google Map and get the GPS coordinates from that, then the drone would fly to it. We found that using the map was expensive for data and processing use. The drone has a built in GPS, so we instead utilized it for pinning down the location of the landing pad. We sat the drone on the pad and captured the coordinates that way. In a broader implementation, we would have the coordinates of landing pads saved, and would reference them from a database instead of trying to pull them from Google Maps before every flight.

The design of the landing platform for the quadcopter changed significantly in order to allow different methods of tracking the target node. We initially had a design similar to a dot with a marker on one corner in order to mark a direction of orientation for landing.This design was eventually left for the simplification of having a single dot and locking the quadcopter to face north using the integrated compass already onboard. Once tracking a single dot was possible, we experimented with using a secondary dot inside the primary dot, but it turned out to be a complication that came too late in our design process. We stuck with the single dot that occluded the camera view completely on landing while locking the quadcopter north.

Our flight controls went through a few iterations as we tried to figure out the DJI SDK. We originally were using a class called Virtual Stick to fly the drone, but some of its functionality was getting in the way of what we wanted to do. It tried to do too much, so we ended up using some of the functions that the Virtual Stick class called itself, so we would have more direct control of the drone's movements.

Another thing that went through an iterative cycle was the charging system. At first, we tried making a charger from scratch. We looked at several micro-controllers that would control the charge flowing into the battery and stop it when it was charged. We bought one or two to test, but we were not skilled enough with electrical engineering practice to build our own circuit. We finally decided to look at options that were already built, and saw a standard battery charger via USB that we all had for our phones. We pulled up the USB breakout schematic and saw it was very straight forward, with only 2 pins, so we ended up breaking out a USB connection and,

through the contacts on the drone and landing pad, we connect the drone directly to a pre-manufactured USB battery charger. This not only simplified our design, but allowed us to be versatile with our charger, since USB is a widely used source for charging. Many batteries on various nodes we could be charging may already be on USB, or could be quickly converted to using it as a charging source.

APPENDIX III: LINKS AND SOURCES

[1] :https://www.google.com/patents/US20160033966
[2] http://www.mdpi.com/1996-1073/10/6/803/htm