HARRIS RECON DRONE

Sean F Flemming, Senior in Mechanical Engineering, University of Michigan

Abstract

This project was sponsored by Harris Corporation as part of the Multidisciplinary Design Program (MDP). The project was to take an off the shelf quadcopter with only manual flight and retrofit it with a video and control system to add functionality. My team was tasked with adding two new flight modes to the quadcopter: autonomous flight via GPS way-points and autonomous flight following road lines. We were also tasked with adding an image recognition system where the drone could identify user defined shapes or images that it passed over. In addition the team also created a user interface to control the drone during operation. The context of this project is a proof concept for a military scouting drone. The main draw of our design over current options was to be its low cost design. As the mechanical engineering student on the team, I was responsible for designing and fabricating all of the component mounts, test fixtures, and safety guards. I was also the head of the electrical system so I designed all of the wiring between the components and completed all of the electrical hardware assembly. I also took the role as the systems engineer and oversaw all of the hardware and high level software integration. This report reviews my work in detail and summarizes the project as a whole.

1.0 DRONE DESIGN

This section will detail the component selection, my modifications to the base frame to support the new components, and the work I performed assembling the drone.

1.1 Component Selection

To start the project the team needed to choose what base quadcopter to start with and what additional hardware was necessary. As the mechanical student I was in charge of choosing the frame and everything to do with the propulsion system. I chose the Aeroquad Cyclone frame because it is a tiered design which I knew would allow the easy addition of new custom tiers for component mounting. Aeroquad recommended motors, electronic speed controllers (ESCs), and propellers, which would work with their frame, but I decided to look for higher quality parts to increase the drone’s thrust. Originally when I went about selecting new motors and propellers I attempted to optimize the motor and propeller combination myself using theoretical calculations. I quickly learned that this was almost impossible without empirical data due to the limited information provided by the motor and propeller manufacturers. Consequently, I found motor distributors that had published experimental thrust information for different propeller combinations. Once I had selected the motors I found ESCs which would integrate smoothly with the flight controller. The final combination of motors, ESCs, and propellers provided just over 4 kg of thrust, which is an improvement of 25% over the stock components. The cost of the new components was only $16 more than the originals, and the motors have a negligible increase in power consumption over the originals at comparable thrust levels.
As the head of the power system I was also in charge of selecting our battery and any other components necessary for the power system; such as the voltage regulator to power the Raspberry Pi. During the battery selection process I completed a weight/cost/flight time analysis to optimize the battery. My final battery selection had an estimated flight time of 12.4 minutes. I also closely monitored the components that my teammates were selecting to ensure that I would be able to properly power them and that they would integrate smoothly with the rest of the drone.

1.2 Drone Modifications

After the component selection was complete, I created block diagrams of the hardware components to ensure that they would integrate properly. Once I laid out all of the physical connections and figured out what adapters were necessary between components, I worked on the design and fabrication of the modifications to the base drone. To accommodate the Raspberry Pi, GPS, and Xbee, I added another mounting layer to the frame, cut out of 1/16” aluminum using the water jet. This piece has the same connection points as the other frame layers so that it integrated smoothly, but it also has specific mounting holes for the Raspberry Pi. The GPS and Xbee also mounted on this plate, but they were just mounted via Velcro due to their small size and in order to keep their locations easily adjustable.

I also designed and fabricated the propeller guards. These helped to prevent accidental contact with the propellers during tests, protecting both our surroundings and our propellers. As with the Raspberry Pi plate, they are also made out of 1/16” aluminum cut on the water jet.

The final modification that I made to the original drone design was how to mount the camera. Due to the desire to create a stable pendulum I decided to mount the battery (which is the heaviest component) on the very bottom of the drone to keep the center of mass below the point of rotation. The camera also had to be mounted on the bottom so that it has a clear view of the ground. Because of this, I decided to Velcro it to the underside of the battery. Using Velcro instead of a rigid mount allowed me to quickly switch out the battery during testing.

1.3 Drone Assembly

The actual assembly of the quadcopter took a lot of time. Since some of the sensors and boards lacked detailed measurements I originally created dimensional representations in the CAD model. Once the physical components were in hand I was able to refine these measurements and figure out the best placement for them. As the electrical lead it was also my job to design and create all of the wiring for the drone. While designing the wiring, I worked to ensure that all of the components were modular and could be swapped out independently of one another in case there was an issue. An exploded view of the CAD model with the components labeled can be seen on the following page in Fig. 1.
Figure 1. Exploded view of drone with major components labeled
2.0 FLIGHT MODES

This section will explain the high level operation of the GPS and computer vision flight modes. While I did not write any of the software myself, I helped with the high level design.

2.1 GPS Flight Mode

For operating the GPS flight mode the user would enter GPS points via the graphical user interface (GUI) that the team developed.

![Figure 2. Graphical User Interface (GUI)](image)

It can be seen that the GUI displays a map with the drone’s current location. The user can enter GPS points manually or by clicking on the map. The GUI also displays a live video feed from the drone in the top left, images with detected objects in the top right, and allows the user to change the flight mode. This GUI was created by my team and I assisted with its layout and design.

After the drone receives a new GPS point that point is placed in a queue on the drone. Once the user is ready to switch into GPS mode they must activate altitude hold (which is a feature on the flight controller) by using the manual controller, then select GPS mode on the GUI. When in GPS mode the drone will look at the next point in the GPS queue and compare that with its current GPS location. Comparing the two points and using the reading from the magnetometer on the flight control board, the program determines a heading adjustment which will result in the drone facing the target GPS point. The drone then adjusts to this target heading using a control loop that I helped write. Once the drone is facing the proper heading it will pitch forward a small static value until it reaches the target GPS point within 3 m². The flight control loops send imitated receiver values into the flight control board via a serial connection with the Raspberry
Pi. Since the drone is in altitude hold the program can adjust the heading and pitch while all of the other flight control values will be adjusted by the flight controller. Once the drone reaches a GPS point that point is cleared from the queue. Ultimately, the control loops for adjusting the heading and pitch should be tuned PID controllers, but there was not enough time to make this adjustment before the end of the project.

2.2 Computer Vision Flight Mode

For the computer vision flight mode, the drone takes a picture of the ground below it and that picture is then fed into a line detection algorithm. The line detection algorithm used finds all edges in a picture and then determines the average of all the edges found. This algorithm can work well, but it needs to be properly tuned depending on the surface around the line and the lighting. Once the program finds a line it uses the orientation of the line in the frame, along with the magnetometer data from when the picture was taken to calculate a GPS point which is five meters in the direction of that line. It then feeds this GPS point into the GPS navigation program. Once the target point has been reached the drone takes another picture and looks for another line to follow.

In addition to assisting with the high level design of this program, I also helped debug the low level math for calculating the orientation of the reported line in the image frame. This program could be improved by using a more robust line detection algorithm that was not as sensitive to background and lighting. It would also be useful to have it be independent of GPS navigation since GPS has an inherent inaccuracy of 3 m².

3.0 TESTING

Due to FAA regulations on commercial drones, the team was not able to conduct actual test flights. This necessitated the creation of a test fixtures that would be able to simulate flight. I design and fabricated two test fixtures from the ground up for this project.

3.1 Test Fixture Cart

The test fixture cart consists of a frame placed on caster wheels with a u-joint in the middle. The drone mounts to this u-joint, which is restricted to a range of motion of 15° in all directions. The shafts between the frame and the u-joint and the drone and the u-joint are restricted in rotation so that in order to rotate, the drone must rotate the entire test fixture. The camera is mounted on the underside of the test fixture so the drone is able to “fly” over shapes, as well as follow road lines while on the test fixture. The frame itself is 1in x 1in 1/8in walled aluminum tubing that I welded together.
3.2 Test Handle

During testing it became clear that I needed a more mobile test piece. In order to test the altitude hold feature of the flight controller (which is used in the GPS flight mode) I needed to be able to adjust the height of the drone while it was running. This caused me to create a test handle, which consisted of a six foot long steel pole with a ball and socket footing joint welded to one end. This footing joint allows 360° rotation, as well as 15° pitch in all directions.


4.0 RESULTS

This section details the results of the tests performed by myself and my team.

4.1 Flight Tests

For the GPS flight mode the team was able to successfully run a flight tests where the drone was placed into GPS flight mode via the GUI and provided with 4 GPS points that made a 20m x 20m box. The drone was able to successfully navigate to all of these points. This test was performed using the test handle, so that when the drone rotated and pitched the person holding the handle would move in the direction that the drone was attempting to move. While we had a successful test it was seen that different factors could cause the system to malfunction. The main issue that we had during testing was weather affecting the GPS and magnetometer sensors.

For the computer vision mode test, the drone was successfully able to follow a large “U” shape laid out with white tape on asphalt. The drone was positioned over the first white tape line and placed into computer vision mode via the GUI. It was able to locate the line and move five meters in the correct direction. There was another tape line placed on the ground five meters from the first, so that when the drone stopped to take another picture it found the next tape line. From there it followed all of the lines as part of the “U” shape.

4.2 Object Detection and Drone Cost

The team did not meet the object detection requirement. The software team became overburdened with figuring out the communication system and the flight modes, so they did not have time to create a reliable object detection algorithm. The implementation for the object detection was all completed, but the algorithm used didn’t provide consistent successful results.

The total cost for the drone and ground system was $765. Compared to similar sized drones used by the military which cost over $100,000, our drone was very inexpensive.

5.0 CONCLUSION AND RECOMMENDATIONS

In conclusion, the team was successful in meeting all of the given requirements except for object detection. The hardware designs and modifications that I created provided the team with a functioning drone. The test fixtures that I designed and fabricated from scratch also provided the team with necessary ways of simulating flight.

The main recommendations that I have for continuing with the design as follows: 1) Replace Xbee communication system with higher quality communication hardware. During the testing the Xbee units had unreliable and low bandwidth. This forced the team to have to send lower resolution pictures down to the ground computer. Higher quality communications hardware would allow high quality pictures to be sent more often. 2) Use a higher quality flight control board with better onboard sensors and stability loops. During testing it was clear that the flight control board was not doing a good job at keeping the drone level. A higher quality board with tighter control loops would help this. Having better onboard sensors would also increase the
accuracy of the GPS navigation since it relies on magnetometer data. 3) Create properly tuned PID loops for the GPS navigation system. Properly tuned loops are essential in order to make the system as efficient as possible. 4) Make the computer vision mode separate from the GPS navigation. Since all GPS navigation has an inherent error of 3 m\(^2\) the road line detection could drift off of position by up to 3 m if the GPS signal is lost and then recovered. At lower altitudes this could easily cause the drone to miss the line. Also, the current method of having the drone move forward in 5 m steps is not optimal for a curved line. Having a computer vision mode independent of GPS would fix both of these issues.