PRE-FLIGHT BATTERY CONSUMPTION MODEL FOR UAV MISSIONS

Eric R. Altenburg
Computer Science
Stevens Institute of Technology
Hoboken, NJ
ealtenbu@stevens.edu

John H. Graves
Computer Science
Brown University
Providence, RI
john_graves@brown.edu

Qijun Gu
Computer Science
Texas State University
San Marcos, TX
qg11@txstate.edu

August 8, 2019

ABSTRACT

Providing a user with the knowledge of how much battery an unmanned aerial vehicle (UAV) will consume for a given flight before it takes off gives them a tremendous advantage. With this, they will be able to act accordingly in the case where their drone battery is not fully charged, or if they have multiple flights planned. Using simulation data, a machine learning model is capable of creating a timeline of maneuvers which then allows the user to predict the average power of a flight and ultimately the maximum amount of time the drone can fly before reaching critical battery levels. By grouping similar drone actions into broader categories and accounting for variation in total energy of LiPo batteries, this model was able to improve on previous research to create a functioning pre-flight battery consumption model. The model preformed well on test data sets, with the machine learning achieving a 95% success rate in predicting maneuvers, and the flight time regression had an R-square of 0.91 on this new data.

Keywords Unmanned Aerial Vehicle · UAV · Drone · Battery · Battery Consumption · Battery Usage · Predict

1 Introduction

In recent years, unmanned aerial vehicles (UAVs) have become increasingly popular in non-military contexts. As their technology improves, these devices become more and more integrated into society as they allow for tasks to be easily completed by a user in a location along a route. For example, Amazon and other distribution companies are developing methods that use these devices to deliver packages to their users in a fast and simple manner. As drones become more effective at delivering packages, they can replace trucks on short, relatively light deliveries, which will reduce the total amount of carbon dioxide emissions and have a positive impact on the environment [1]. This technology is not limited to the aforementioned companies, instead, it is also used by archaeological sites and other sports companies to conduct surveillance missions for certain dig sites where a camera might not be the best choice, or at a football game. Police forces also have begun to use these UAVs in situations where it is not entirely safe to send in an officer. By allowing that officer to remotely control a drone, it allows them and many others to be safe.

One of the many things holding UAVs back from doing such things are the simple fact that the Lithium Polymer (LiPo) battery usage is not easily predictable. On a basic level, the reason this battery consumption is difficult to predict is that LiPo batteries do not consume voltage linearly. Its voltage curve starts off fairy steep, but quickly levels out for the majority of its capacity before dropping off quickly when it reaches low energy levels (See Figure 1). If these non-linear nature of a battery could be properly quantified though, users could have the ability to accurately predict how much battery a given drone flight will consume which would allow them to determine whether or not certain flight missions were feasible. Going back to the previous example with Amazon, their delivery drones will not always be charged to maximum capacity, therefore, if they can see how much battery a delivery will take up, then they can determine whether or not they will need to send a different drone or replace the battery. Up until now, most UAV software can only communicate to the user how much battery is remaining in real time, and while that may be enough, in some

situations—like the one mentioned previously—it is important to know how much will be consumed before the UAV leaves the ground.

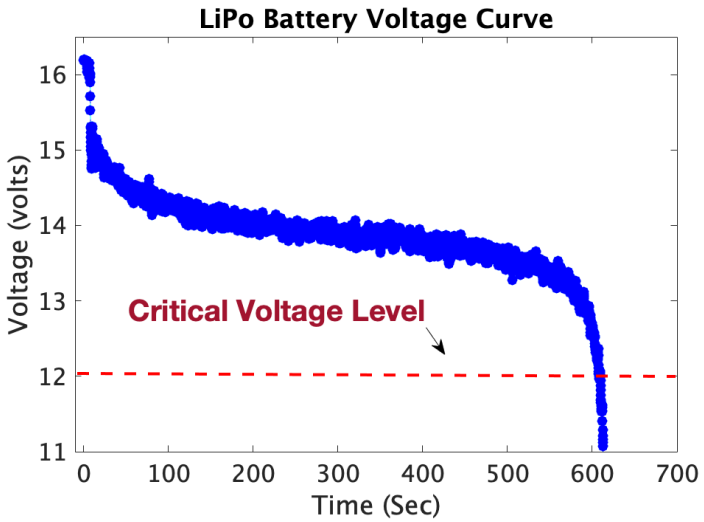

Figure 1: The voltage curve of a standard LiPo battery. This curve fits especially well with a four cell LiPo battery.

While there have been a few attempts at creating such a model, there has not been a successful method that results in an accurate or efficient prediction. This is partially due to drones being relatively new, having only been introduced in the past few years, and as a result, not much research has been conducted. It is from this lack of literature that many have had difficulty creating a functional model. Specifically, one of the assumptions made about these drones is that the total battery energy consumption is constant from flight-to-flight [2], however, through various data samples, this is not the case; it is variable. This and many other assumptions make up some of the shortcomings in existing literature, that if not proven correctly, can result in a considerable setback in the UAV field.

2 Related Work

With very few pieces of literature in the field of UAV battery consumption, there were a select few that stood out. Previous attempts at creating a similar model have attempted to separate the maneuvers a UAV performs during its flight into separate categories such as vertical, hover, and horizontal movements [2]. Others took a more detailed approach and had back, down, forward, roll left, roll right, up, yaw left, and yaw right [3]. Although these two studies were not entirely successful, they were still able to provide motivation to isolate the maneuvers into ascending, descending, hovering, and any horizontal movement.

Other research in the field focused primarily on quantifying the sub-optimal nature of battery discharge with a model the considered the rate voltage dropped in comparison to the batteries current to form a battery-aware consumption model [4]. This model improved up to 16% on traditional battery models, which in some cases could be the difference between a drone crashing or returning safely.

2.1 Flaws and Improvements

As previously stated, these studies come with slight caveats. For the case of the detailed maneuver model, the data they collected on their quad-copter showed surprising results as shown in Figure 2. Here, they were able to map the current in Amps being drawn by each maneuver and deduced that down and roll left consumed little to no battery; up consumed the most; back, forward, roll right, yaw left, and yaw right consumed an amount in between the others. There are some aspects of this finding that raised concern, for example, roll left consumed a minuscule amount of current whereas roll right consumed much more. This puts roll left on the same level as moving down which seemed questionable.

After collecting data from several flights, there was no similar pattern to that of their original findings, therefore, leading to the conclusion that roll left did not consume less power than roll right. Additionally, another finding in the data set showed that the differences in power consumption between all maneuvers in the horizontal plane were minute, thus allowing the classification of maneuvers to only include ascend, descend, hover, and horizontal movement. However,

it is important to note that the UAV they used had four total rotors (quad-copter) while the one used primarily in this project had six rotors (hexa-copter) which could lead to a difference in results.

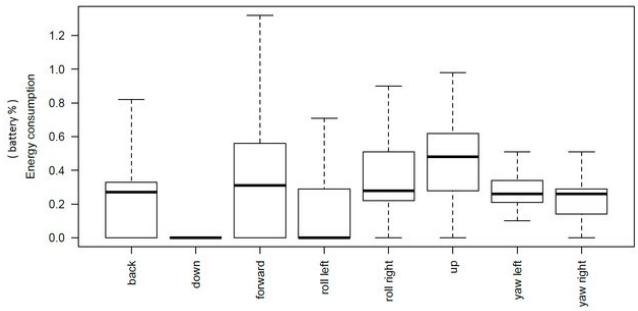

Figure 2: Graph showing the difference in energy consumption between maneuvers of a quad-copter [3]

.

Another potential flaw in much of the research we found was that they did not consider variation in total energy of the battery from flight to flight, instead assuming it was always constant [2] [4]. Depending on how much the total energy can change, this could have enormous impacts on the success rates of the models. Our data found that total energy did in fact change from one flight to the next which will be discussed in detail later.

3 Methods and Procedure

One a very basic level, the model used two steps to predict pre-flight battery consumption. The first step was to determine which maneuvers a drone would fly. To do this, simulations [1] were completed in the drone software QGroundControl [2] and manually labeled with different maneuvers. This data was then used to train a machine learning model which could then take in a simulated projected flight path of an actual drone flight, and output a timeline of what maneuvers the drone would be doing and for how long.

The second modeling step was to predict the power associated with each maneuver, and ultimately the maximum flying battery life of a drone for a particular flight. First, five actual drone flights were manually categorized by maneuver, and analyzed to find the average power associated with each drone action. Next a separate series of tests were done with a grounded drone, draining the battery at different rates to test how much total energy the drone used before it dropped below the critical voltage level indicating the battery was empty. A rational model was then fit to this data to predict maximum flight time of a drone given an average power during a flight.

Putting these parts together, the maximum flight time of a drone could be predicted by estimating how long the drone would be doing different maneuvers, using these durations and average power of maneuvers to determine an average power of the entire flight, and then using that number in the regression equation to predict the total flight time.

3.1 Data Collection

Three types of data were used for this model, each collected in their own separate way. First the machine learning model used simulation data from the QGroundControl software which simulates drone flights. The way this works is that a flight path with actions along the way is planned out, just as you would normally do pre-flight, but instead of going straight to doing the real test, this software can run a simulation before hand, included fairly detailed data about the position, rotation, acceleration, and speed of the drone. To train the machine learning model, ten tests were run in this simulation software, including a wide variety of paths and hovering at different altitudes. These paths included, squares, triangles, figure eights, simple straight paths, or just hovering in one spot the entire time.

The next type of data was the actual drone flights. The analysis relied on five different flights, taken at different times. Three of these flights had been recorded for a previous research project, but comprehensive data was available to analyze them. Two more of these flights were completed specifically for this research. One featured hovering for a minute at 20 meters and the other a clockwise circle at 10 meters.

[1] http://ardupilot.org/dev/docs/sitl-simulator-software-in-the-loop.html

[2] http://qgroundcontrol.com/

The third type of data collected was data from a grounded drone with its propellers flipped upside down to push it into the ground. With this setup, the drone would be put into manual mode and the throttle input was adjusted to test the results of draining the battery at different rates. The drone would be allowed to run until its voltage dropped into critical levels, the same level at which it would normally go into emergency landing procedure. These tests were repeated with a different battery to to compare the results, and ensure the the trends were not specific to one battery. On some of this data, the sensors stopped recording so power data could not be obtained, but in those cases total current draw was manual reported.

3.2 Machine Learning

In order to reduce the total amount of human error while evaluating maneuvers and streamline the overall process of estimating the battery consumption, machine learning proved to be a helpful tool. With the use of a classifier decision tree, the user would be capable of loading up simulation data from a desired flight plan, and the model would output a precise timeline of maneuvers the drone will perform every tenth of a second. As previously mentioned, this was a useful alternative to simply evaluating a flight pattern by eye as having the maneuvers of a drone off by a couple seconds could result in immensely different results when in practice.

The process of choosing this model was done so by evaluating what was needed for this project. Seeming as though there was no need for regression, it was clear that a model that worked with classification had to be used. While this narrowed the overall amount of options to choose from, there were still quite a bit left. Originally, random forests were set to be implemented, however, they tend to be costly in terms of time, and with the data smoothing algorithm that will be evaluated later on, speed was a concern. Finally, a decision tree was settled on due to its swiftness, easiness to understand, and favorable accuracy results. The entire tree was based on the Gini index–the probability of whether a decision is wrong or not, therefore, generally as one traverses farther down the tree, the Gini index should become smaller. An example of the tree can be seen in Figure 3 in which it shows a tree with a depth of 3. Generally speaking, finding the right depth of a tree depends entirely on the data as one does not want to over or under fit the data which will yield sub par results.

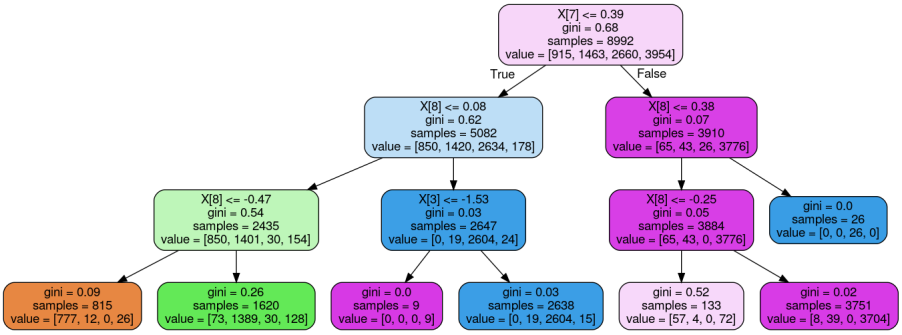

Figure 3: Classifier decision tree visualized. Each color represents a maneuver with orange being ascend, green hover, blue descend, and pink with horizontal movement.

After the model is finished with all of its predictions, it outputs them all into a standard array, and once a simple compression algorithm is completed (see Section A.1) the timeline resembles one similar to Figure 4. In the first column, each maneuver is assigned to a number: 1 ascend, 2 hover, 3 descend, and 4 horizontal movement, while the second column contains the total amount of time each maneuver took every tenth of a second. However, this is still not the final result as there are still instances of sporadic predictions with a time value less than 10 (1 second); this is most likely an incorrect guess. If these were to remain in the final timeline output, then calculations can become more complicated and inaccurate. To correct this, a data smoothing algorithm is then applied (see page A.2) where it will evaluate each maneuver to see whether it is below the 10 time unit threshold. If it is, it will then look above and below to determine which value to change it to; the maneuver will default to the one with the highest time value. The final result is then parsed to a comma-separate values (CSV) file where it will be used in average power prediction. An example of a final timeline can be seen in Figure 5.

2	172
4	167
2	7
4	212
3	86
2	90

Figure 4: This is a portion of an initial timeline. In this scenario, maneuver 2 (hover) has an instance with a low time unit of 7.

2	172
4	386
3	86
2	90

Figure 5: This is a portion of the final timeline that gets used in the power calculation. All values in this are above the time unit threshold of 10 to reduce complication and to retain accuracy.

3.3 Power Statistics

To examine the average power of different maneuvers, the five five drone flights were manually divided up into sections representing the four maneuvers that were used in the machine learning model, by looking at altitude and ground speed. The data was reduced to one second intervals using the pchip method in Matlab and then further smoothed by using a moving mean algorithm covering nine seconds (four on each side of the data point).

As Figure 6 shows, comparing the data from a single maneuver, in this case hovering, shows that there are large differences between these tests. This was explainable by different configurations in the drone, including parameters, settings, and even different copies of the same type of drone and battery. Of the five flights, flight one, flights two and three, and flights four and five were all under the same conditions, so the flights were separated into those three groups. A weighted average of power for each group of flights was preformed, which controlled for the frequency of different maneuvers. It did this by weighting each maneuver with a weight proportional to the smaller duration of that maneuver between the two flight groups being compared. Figure 7 shows this process with real data. By finding the ratio of these weighted averages, a coefficient was found to scale the power data from the first two groups of flights, which put them on par with the average power of group three. The normalized data was then analyzed with a box plot split by maneuver and by finding the average power of each maneuver to use in the final model.

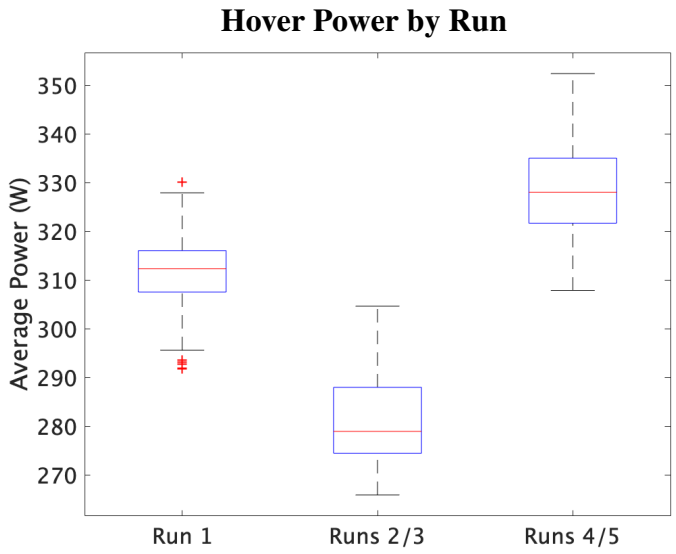

Figure 6: Comparing the distribution of hovering between three groups of flights.

Normalizing Power Data

Normalizing Average Power for Run 1			
Maneuver	Overlapping Count	Average Power (A)	Run 4/5 Average Power (A)
Ascend	33	315	376
Descend	NA		
Hover	68	312	331
Horizontal	NA		
Weighted Average Power:		313	343
Ratio		343/313 =	**1.10**

Normalizing Average Power for Runs 2 and 3			
Maneuver	Overlapping Count	Average Power (A)	Run 4/5 Average Power (A)
Ascend	24	303	376
Descend	46	267	330
Hover	68	281	331
Horizontal	34	271	360
Weighted Average Power:		288	343
Ratio		343/278 =	**1.23**

Figure 7: A demonstration of the weighted average computation to normalize the power data for run 1 (left) and runs 2 and 3 (right).

To examine the maximum flight time the data from the grounded tests which drained the battery were parsed to find the total energy of each flight and the average power of each flight. For flights where the sensors stopped recording mid-test, the total current draw was recorded and the relationship between total current and total energy was analyzed with linear regression to attempt to find a power approximation from these flights, even though the actual metric was not recorded. While processing this data it was noticed that the total energy of these flights was not constant as previous battery analysis had assumed [2], so further analysis was done to compare the total energy consumption of a test with its average power.

4 Results and Discussion

4.1 Machine Learning

The way the accuracy of the model is measured is through simulated flights in which the maneuvers are already known. Once the initial predictions are made, it is then compared to the correct predictions and this tends to yield a 93% accuracy rating. This is then improved upon with the data smoothing algorithm, and the new accuracy rating is 95%. The confusion matrix for a test can be seen in Figure 8; the shaded boxes are the correct guesses made by the model.

		Predicted			
		Ascend	Hover	Descend	Horiz.
Actual	Ascend	81	8	0	41
	Hover	0	389	7	54
	Descend	0	3	277	1
	Horiz.	0	0	0	1180

Figure 8: From a prediction on simulation data, this confusion matrix shows what the model predicted versus the actual value. The accuracy is 95%.

While the accuracy of the model may be sufficient, other models should be explored. A classifier decision tree was chosen based purely on its reputation for speed. However, if a different model were to be chosen, say a random forest, then the accuracy could potentially be improved upon.

In terms of efficiency, the model itself is optimized very well, however, other functions being used can certainly be improved. The time complexities in the worst case can be quite time consuming if the data set is large enough. If the algorithms were more optimized, then the overall efficiency of the model would surpass that of its past rendition.

4.2 Power Statistics

Once the maneuver data had been normalized, and smoothed, the final results could be analyzed and average results calculated. As Table 1 shows, ascending used the most power for the drone, while descending used the least, and

hovering and horizontal movement were in the middle with effectively the same average power. These results generally support common assumptions, since ascending and descending are going against and with gravity respectively, but it was a bit surprising that there wasn't more of a difference between hovering and horizontal movement. Figure 9 shows the distribution of the data median average power. The hover data has a higher median and less variation than horizontal movement, which has a lot of outliers with high powers. In general, the large amount of variance signifies that more data would be beneficial. The interquartile range of the results overlap for every pair of groups besides ascending and descending, which is another indication of how the significance level of these results is low. Understanding the variance, is extremely important on a flight by flight basis, and less so on a second by second basis, since differences between flights is relevant when accounting for variance in the total amount of flight time. Five flights is not nearly enough to test flight to flight variance, so this is one area where future work would be beneficial. Regardless, the facts that the graph of the residuals shows no trends within particular maneuvers, and that the results are explainable by scientific principles, are good indicators that the results are close to reality.

Average Maneuver Power (W)

Ascend	Descend	Hover	Horizontal
365.2	326.8	341.4	341.5

Table 1: Average power for each of the four maneuvers.

Adjusted Maneuver Power Distribution

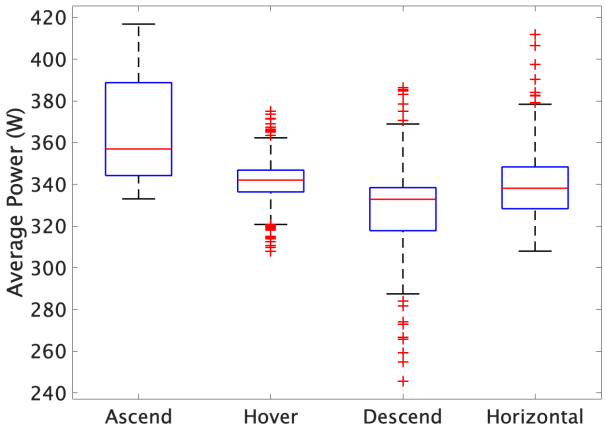

Figure 9: Sample figure caption.

The average power and flight time data demonstrated a clear non-linear trend when graphed as a scatter plot which could be modeled with a rational regression equation (Equation 1). As Figure 10 shows, the regression fit the data well, with an R-square value of 0.95. As the average power increased, flight did not last as long, which was expected, but at high powers, the fight was actually using more total energy than at lower power levels. This rational equation can work in tandem with the machine learning model and average maneuver power predictions to estimate the maximum flight time of a drone flight. Given a timeline of how long a drone will be completing each maneuver, the average power predictions can estimate the average power of an entire flight. Plugging this value into the regression equation will output an expected maximum flight time. Users can then compare that flight time to the simulation flight time to determine whether or not the flight is viable and if it is, roughly what percent of battery consumption is left (by dividing simulated flight time by maximum flight time).

$$T = \frac{321.9p + 98770}{p + 29.3} \tag{1}$$

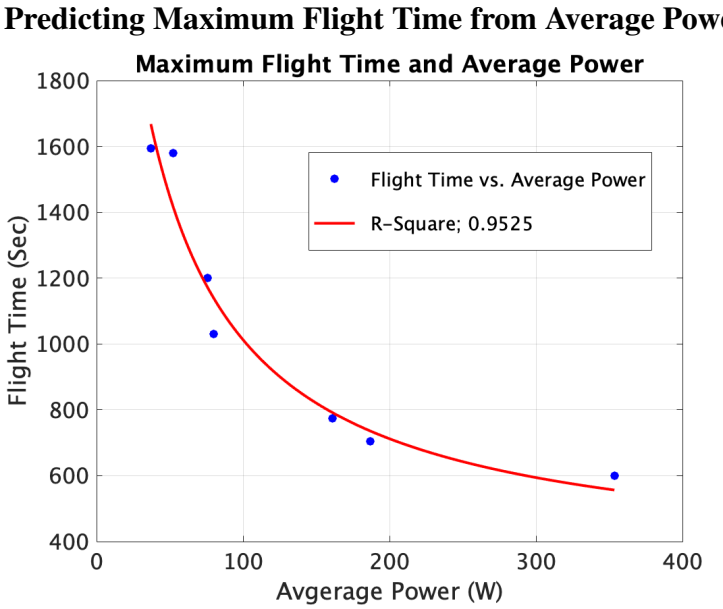

Figure 10: A rational regression equation used to predict the maximum flight time from average power.

Figure 11 demonstrates the trends in battery energy with two different batteries by comparing average power and total energy in a scatter plot which shows that total energy is far from constant. This was a major finding that is relevant to both drones and other fields, since LiPo batteries are common across a wide range of products including smartphones and electric cars, and it is unclear how universal is the belief that battery energy is not dependent on discharge rate. In the field of drone research, these trends in battery discharge were not accounted for in some of the most prominent analysis on drone battery consumption. Most models assumes this constant battery energy, which could lead to harmful predictions. Our model finds that battery energy, at least within the range we are considering, increases linearly with a model that fits the data very well. Battery 1's linear regression has an R-square of 0.99, and battery 2, even with just three data points, has an R-square of 0.88.

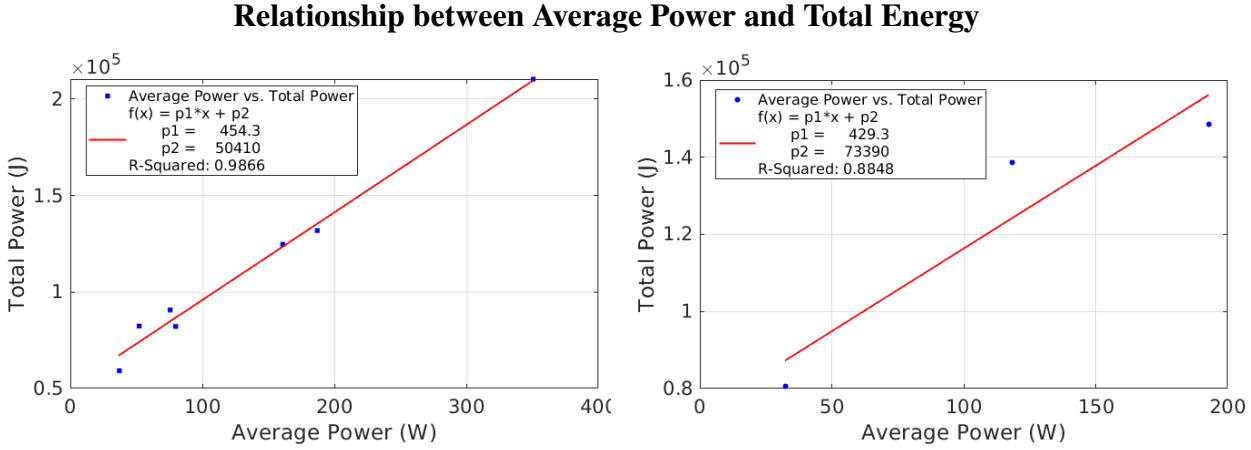

Figure 11: Finding the linear regression relationship between average power and total energy.

To test the regression equations, tests which had not recorded power were used but did have average and total current information. To do this, first the relationship between average current and average power was analyzed, which needed to be very strong for the analysis to work. This was expected to be the case since power is calculated by multiplying current and voltage and the voltage curve should be pretty consistent from test to test. This would mean that average current should move proportionally to average power, which is what Figure 12 found. A linear regression fit the data almost perfectly with an R-square value of 0.999.

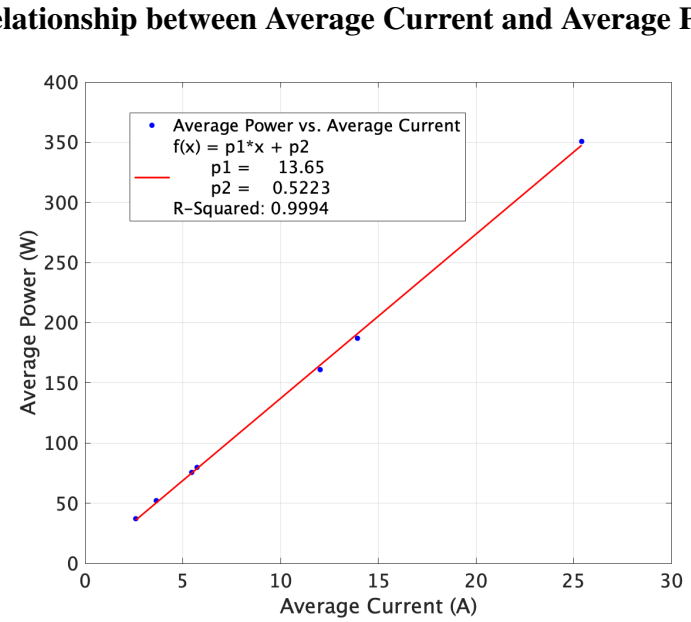

Figure 12: The linear relationship between average current and average power is strong with an R-Squared value of 0.99.

Given this strong relationship, average power was able to be predicted from the average current data available in the extra tests that were not used in the originally flight time regression equation. The predicted average power data could then be graphed against flight time, to test the original regression. Given these five new data points, Figure 13 shows that the model continued to preform well with an R-square of 0.91. The sensor on these specific flights seemed to stop working pretty randomly, making them a perfect test group, which is positive evidence for the validity.

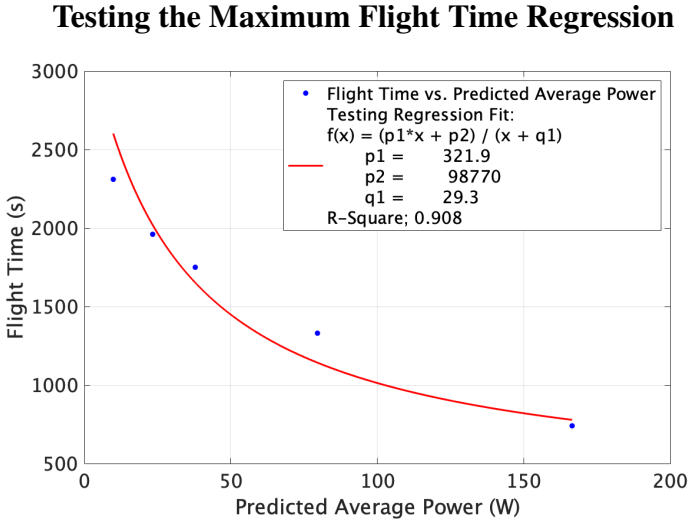

Figure 13: Using average power data predicted from average current to test the rational regression equation estimating maximum flight time. It preforms well with a R-Squared value of 0.91.

5 Conclusion

A Appendix

A.1 Timeline Creation

```
# Iteraves through the prediction data to create a time line for each of the maneuvers
def maneuverTimelineHelper(prediction_data, maneuver, man_counter, timeline):
   maneuver = prediction_data[0]

   for i in range(prediction_data.size):
      if ((prediction_data[i] != maneuver) or (i == prediction_data.size-1)):
         if (i == prediction_data.size-1):
            timeline = np.append(timeline, [[maneuver, man_counter+1]], axis = 0)
         else:
            timeline = np.append(timeline, [[maneuver, man_counter]], axis = 0)
            maneuver = prediction_data[i]
            man_counter = 1
      elif (prediction_data[i] == maneuver):
         man_counter += 1

   return timeline

# Calls helper method
def maneuverTimeline(prediction_data):
   timeline = np.empty(shape=[0, 2])

   return maneuverTimelineHelper(prediction_data, 0, 0, timeline)
```

A.2 Data Smoothing

```
# Smoothes the data to get rid of any extraneous values. If found, will default to the biggest
    next to it. If surrounding are equal, loop to the next until it finds one that isn't equal to
    each other
def smoothData(timeline):
   if (timeline.shape[0] == 1):
      return timeline

   timeline = removeRepeats(timeline)
   re_smooth = True

   while (re_smooth == True):
      for i in range(timeline.shape[0]):
         if (timeline[i][1] < 10.0):
            if (i == 0 and (timeline[i+1][1] > timeline[i][1])):
               timeline[i][0] = timeline[i+1][0]
               continue
            elif (i == timeline.shape[0]-1 and (timeline[i][1] < timeline[i-1][1])):
               timeline[i][0] = timeline[i-1][0]
               continue

            above = timeline[i-1][1]
            below = timeline[i+1][1]

            if (above > below):
               timeline[i][0] = timeline[i-1][0]
            elif (below > above):
               timeline[i][0] = timeline[i+1][0]
            elif (above == below):
               count_up = i+1
               count_down = i-1
               searching = True

               while (searching == True):
                  if (timeline[count_up][1] == timeline[count_down][1]):
                     if (count_up+1 in range(0, timeline.shape[0])):
                        count_up += 1
```

```
                if (count_down-1 in range(0, timeline.shape[0])):
                    count_down -= 1
                else:
                    searching = False

            if (timeline[count_down][1] > timeline[count_up][1]):
                timeline[i][0] = timeline[count_down][0]
            else:
                timeline[i][0] = timeline[count_up][0]

    timeline = removeRepeats(timeline)

    for i in range(timeline.shape[0]):
        if (timeline[i][1] < 10.0):
            re_smooth = True
            break
        else:
            re_smooth = False

return timeline
```

References

[1] A. Goodchild and J. Toy, "Delivery by drone: An evaluation of unmanned aerial vehicle technology in reducing co2 emissions in the delivery service industry," *Transportation Research Part D: Transport and Environment*, vol. 61, pp. 58 – 67, 2018. Innovative Approaches to Improve the Environmental Performance of Supply Chains and Freight Transportation Systems.

[2] A. S. Prasetia, R. Wai, Y. Wen, and Y. Wang, "Mission-based energy consumption prediction of multirotor uav," *IEEE Access*, vol. 7, pp. 33055–33063, 2019.

[3] L. Corral, I. Fronza, N. El Ioini, and A. Ibershimi, *A Measurement Tool to Track Drones Battery Consumption During Flights*. 01 2016.

[4] Y. Chen, D. Baek, A. Bocca, A. Macii, E. Macii, and M. Poncino, "A case for a battery-aware model of drone energy consumption," pp. 1–8, 10 2018.