Connor Phillips

Professor Trihinas

Data Analysis for Drones

July 8th 2021

Data Analysis, Drones, and Machine Learning

Abstract

In this paper, I showcase how machine learning can be used to fly drones, with focus being placed upon how data from other drone flights can be analyzed. This data can then be used to draw conclusions about the behavior of the drones, or be used to train a later generation of algorithms. To do this properly, Exploratory Data Analysis is utilized to ensure that clean data is being examined. The ability of drones to be operated by AI controllers is demonstrated using the WeBots simulation program. The controller in question functions on facial recognition software developed in Python for use in images, but is adapted to act as a guiding point for the drones. The ability of machine learning to control drones gives a great indication of how drones may be used in the future: with little to no human interaction.

Introduction

It's not controversial to say that one of the most fascinating inventions of the modern era is the drone. For eons, the idea of a man made flying machine was simply a day dream. But the drone stands as a reminder of how technology has advanced to the point where artificial flying machines can be controlled remotely over massive distances and be of varying shapes and sizes. As technology has continued to improve, humans are no longer required to pilot these drones. Indeed, having an individual controlling every individual drone is infeasible, especially as the number of drones continues to rise. This is where machine learning comes in. Machine learning is a branch of AI which focuses on computer algorithms which are able to learn and act based upon input data. With the power of machine learning, drones are able to perform tasks far more complex and intricate than those capable by manual controls. Drones piloted by machine learning algorithms are capable of responding to all kinds of stimuli, like changes in humidity and adjust their flights accordingly. Even basic machine learning algorithms, like FlockAI, allow drones to utilize facial recognition and interact with their surroundings beyond the human limit. This versatility comes at a cost however, as machine learning algorithms require training on the data that they will be accepting, and even then success is far from guaranteed. The nature of machine learning means that as input data drifts from the data used to train the AI the less likely the AI is to properly handle the data. This is why AI is often trained using a set of data chosen to be as diverse as possible, but since creating a data set of all possible inputs is infeasible, the possibility of error is omnipresent. Thus it is of great importance that data used to train be as accurate as possible to minimize this risk.

Background

Beginning my work with FlockAI, I first downloaded and familiarized myself with some of Python's many libraries, including numpy, pandas, and seaborn. These libraries consist of pre-written collations of code, to help reduce the effort required in running frequently used processes. Within the provided examples, numpy handles mathematical processes for matrices and arrays, pandas allows for fast data analysis and importation, while seaborn helps produce visualizations of statistical models. All of these libraries are used in conjunction to perform Exploratory Data Analysis, or EDA. This is a process where a large collection of data is examined and sorted to allow new data to be derived.

Besides EDA, FlockAI required an understanding of facial recognition algorithms, specifically the facial_recognition Python module. This module utilizes a C++ library of facial recognition algorithms to identify faces in an input image. These algorithms were in turn built upon deep learning, which is a kind of machine learning which attempts to simulate the human brain's thought processes. This allows algorithms to perform tasks which are trivial for humans, like facial recognition, but are difficult to explicitly code.

Finally, we utilized the program WeBots to actually utilize the code we developed. Because of the long distance between our locations and Cyprus, where the drones are actually located, we required an alternative way of simulating a drone flight. WeBots is a program designed to simulate drones and other autonomous robots for the purpose of design and controller testing. FlockAI came with a simulation already created, which included both recreated drones and a world environment to test them in. WeBots nature as a computer simulation means that data can be collected from the drones under controlled conditions, without the need for physical sensors. This freedom is important in creating a wide array of data to

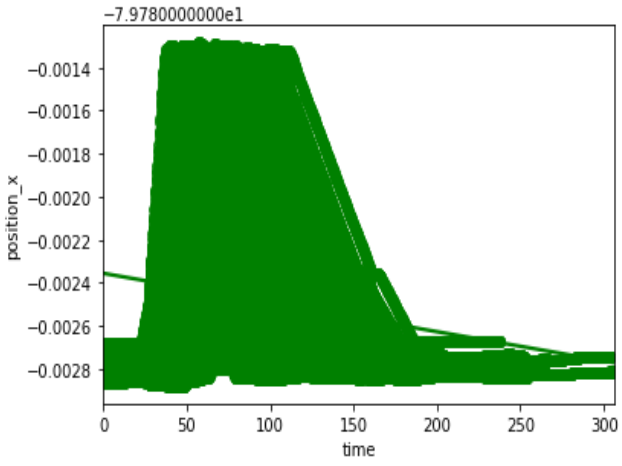
analyse for EDA, and to use in machine learning for later generations of the drones and their controllers.

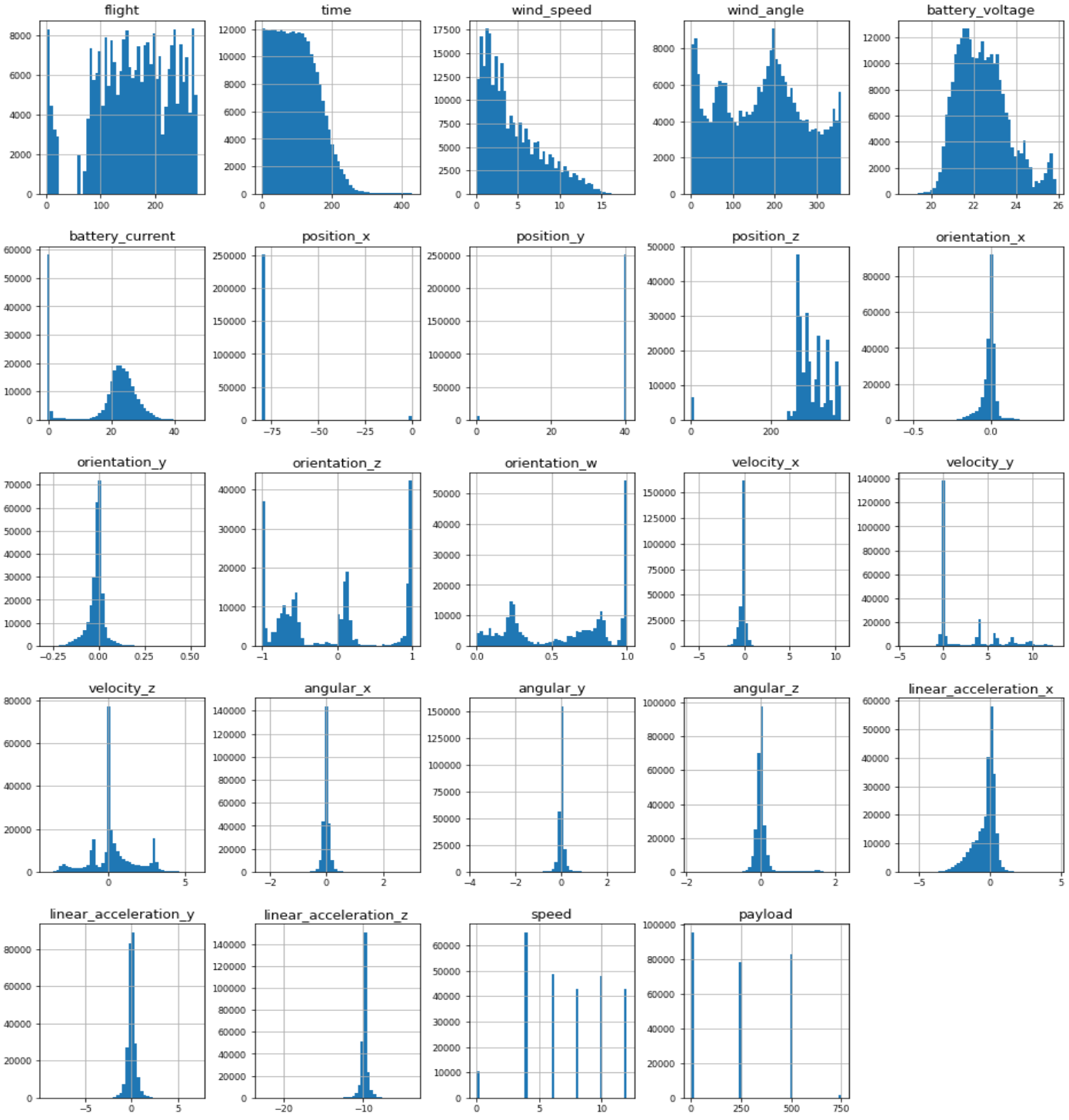
Design/Implementation

I performed my exploratory data analysis on a dataset of recorded flights by Unmanned Aerial Vehicles. There were 279 different flights performed at a variety of times and under a variety of conditions. In total, 28 variables were recorded. These were: flight number, time, wind speed (in m/s), wind direction (degrees with respect north), battery voltage, battery current, position in xyz components (x and y recorded in latitude and longitude and z recorded in meters above sea level), orientation in quaternions, linear velocity in xyz components, angular velocity in xyz components, linear acceleration in xyz components, speed, payload (of which there were 4 unique values), altitude (which the drone was set to fly to), date, local time (in 24 hour format), and route type (of which there were 11 unique values). Each variable had 257,896 recorded values, resulting in 7,221,088 data points.

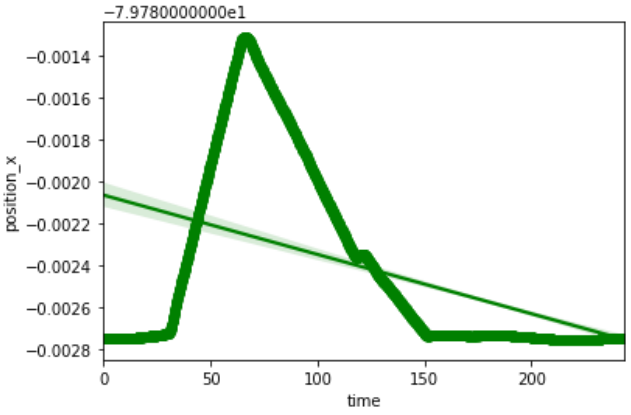
I began my EDA by reading the drone data directly from the .csv file using Pandas. After confirming that everything was imported properly, I displayed histograms of all numerical values to better understand the distribution of the data. These histograms are included on the next page.

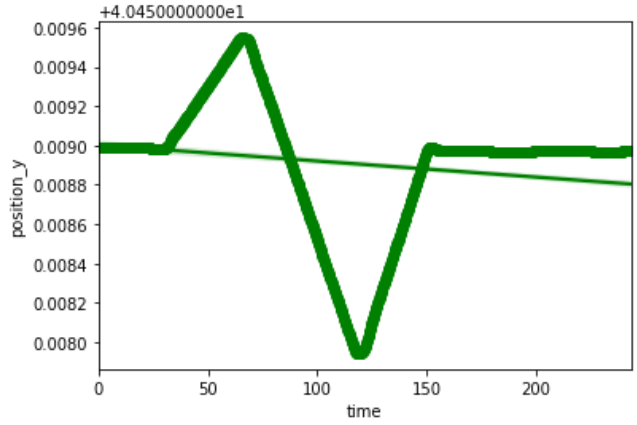
Following this, I wanted to get a better understanding of the routes the drones took. To do this, I selected a random route to examine. I selected the rows of data containing flight details for route type ‘R1’. Attempting to plot the x value of the drones along the time axis gave the graph you see here. As you can see, it is nearly

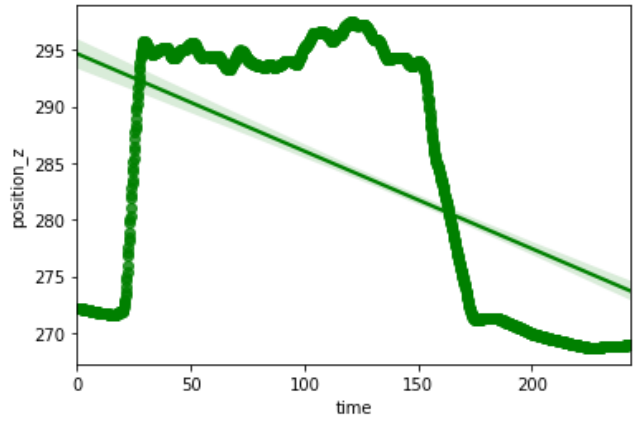


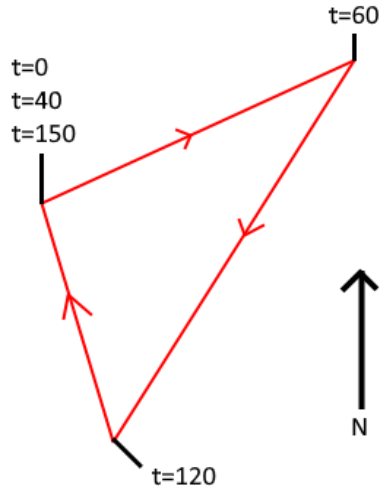


impossible to determine the true path of the drone given this data. It's likely, then, that one of the other variables which is adjusted between flights was affecting the ability of the drone to fly along the path. I decided that it was most likely the payload which was causing the drift.



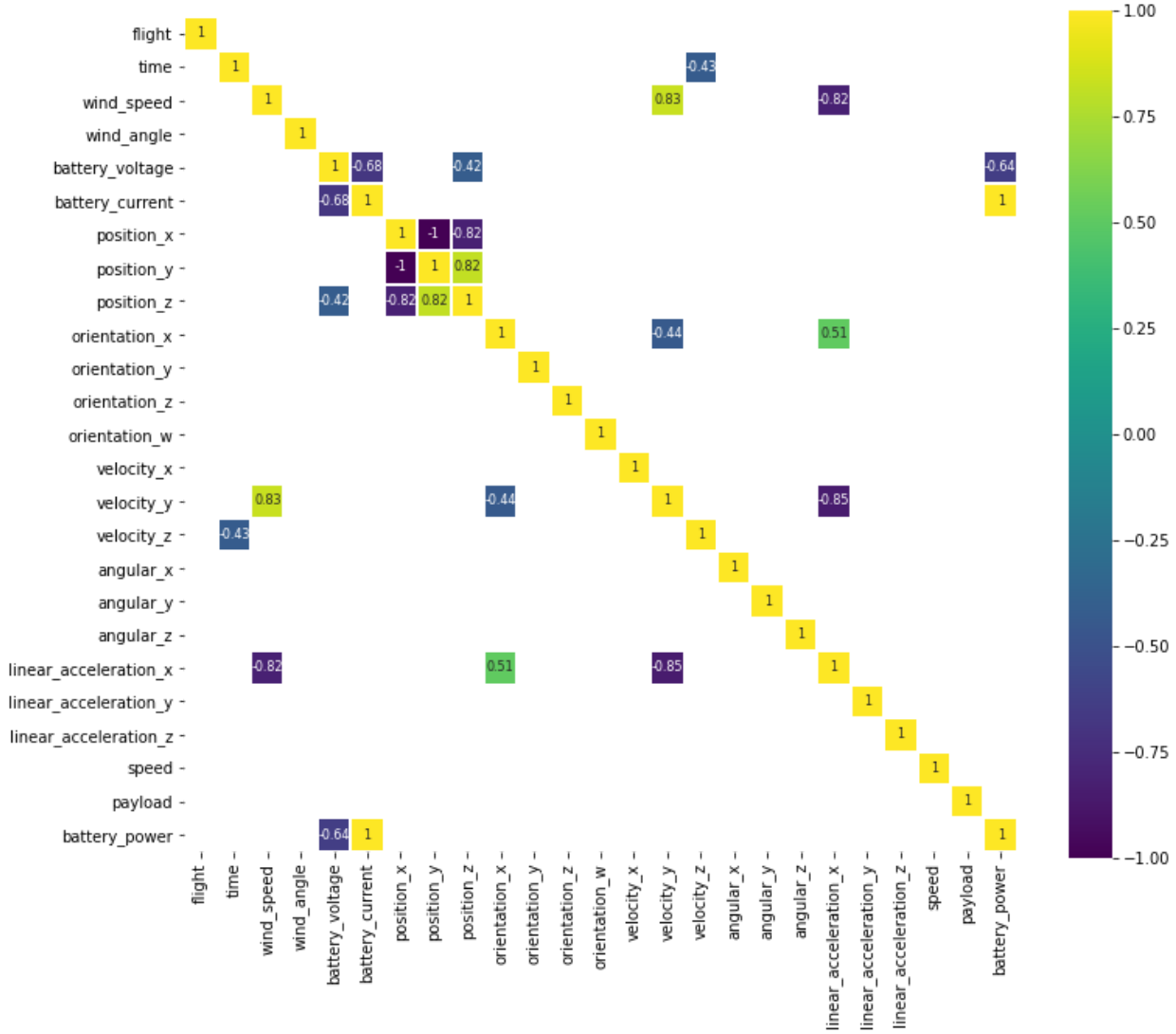




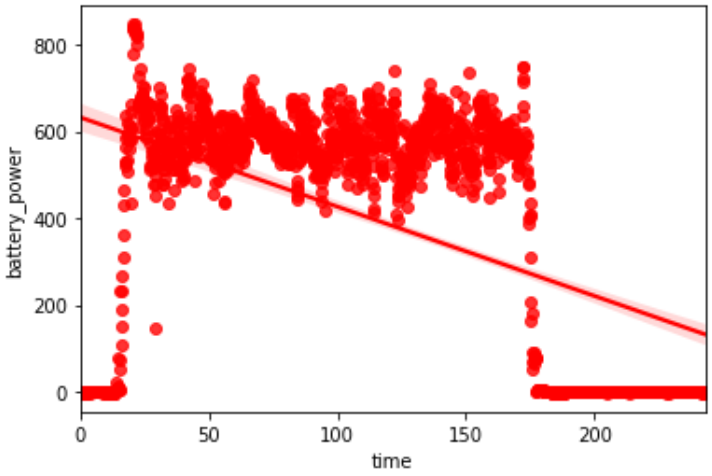


Limiting the data to only flights on route “R1” with a payload of 750g, produced this graph. Here you can clearly see the path that the drone takes. In addition, when comparing this graph to the previous one, we can see that this same general pattern was taken by all the drones flying on route type ‘R1’, with the differences in time causing the paths to all overlap. Performing a similar process with the y and z values of the drone flights allows us to have a complete understanding of the path the drone takes. Initially it takes time until its rotors are at a fast enough speed to achieve lift, it then flies to its maximum altitude, which occurs at t=40, and attempts to preserve that height throughout its route. It begins heading East-North-East until roughly t=60 at which point it begins to travel South-South-West until t=120 when it finally heads North-West until t=150, at which point it descends until it has returned to its original position. This same method of analysis can be applied to all of the routes, although route “R1” is the most difficult to perform due to its variations in flight times.

With a solid understanding of the flight paths that the drones took in each route, we can begin examining some of the further details of the drone's performance. Key among these is the drone's battery performance. We have two variables related to the drone's battery: current and voltage. These two variables are correlated, as seen in the diagram below, which shows all correlations between every variable. This makes sense, as Ohm's law states that voltage and current are directly proportional to each other. Using these values, we can actually find a new piece of data, that being the power consumed by the drone at that moment. The power law states that P=IV, thus by multiplying the current and voltage we can produce a new column of power.



This new variable opens up a brand new door of possibilities for comparison, so let's begin by plotting it against time. Doing so when constrained to route “R1” and of payload weight 750g, produces the following graph.

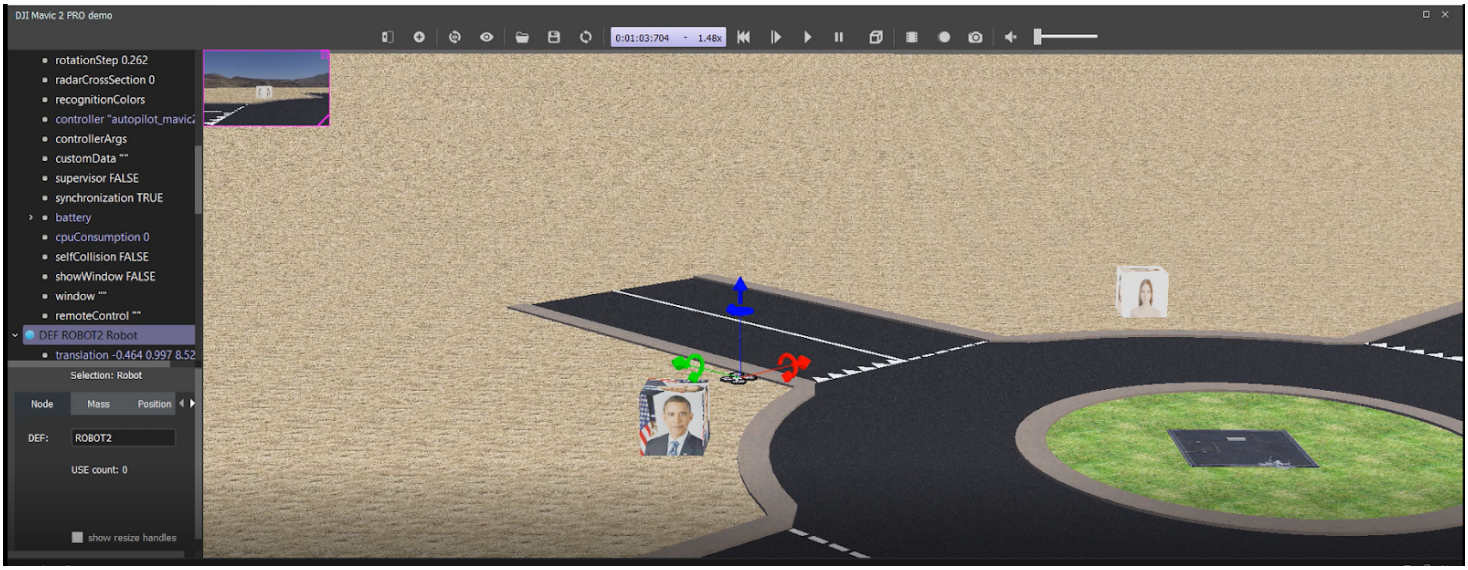


Immediately, there appears to be a connection between battery power and the z-position of the drone, as their graphs appear to be relatively the same shape. To confirm this, we can display all variables with a strong correlation to power. We can see voltage and current have the strongest correlation to power, which makes sense as they’re used to make power. Besides these, the z-position is far and away the most strongly correlated value. This strongly implies that the majority of the drone’s power is used in maintaining its height rather than in gaining horizontal velocity, which may seem counterintuitive at first glance. However, when you consider that the drone must be constantly fighting against the force of gravity while in the air, this actually does make sense.

```
There is 9 strongly correlated values with Battery Power:
battery_current    0.999720
position_z         0.844612
wind_speed         0.671045
velocity_y         0.650868
position_x         0.532068
orientation_z      0.511086
time              -0.513095
velocity_x        -0.573153
battery_voltage   -0.945748
```

With the EDA complete, we can finally move on to the drones themselves. Using WeBots, we can begin the FlockAI simulation. We chose to have the drone controller be “autopilot_mavic2dji”, our automatic facial-recognition based program. When run, the drone takes off to a set height and begins moving towards the first detected face. When it reaches the face, it turns and attempts to find another face. By placing faces as textures on cubes, we can

make the drone fly in a triangular shape similar to the one we determined from our Exploratory Data Analysis.



Evaluation

So, what exactly can be gathered from this data? From the EDA, we have gained a better understanding of how the power used by the drone is spent. This information may prove invaluable when actually deploying drones, as it provides evidence that drones are capable of complex flight paths as long as they remain at a constant altitude. This is great news for the Flock AI drones, which remain at a constant height to maintain the best viewing angle for facial recognition.

Besides evaluating our EDA, I also want to evaluate my own successes with the programs I used in creating this paper. I ran into the largest issues with WeBots, which frequently crashed due to its high CPU usage on my laptop. After launching WeBots in safe mode, I ran into my next roadblock when the drones refused to work with the controller scripts. Through roughly a week of troubleshooting, I was able to fix the issue and use WeBots normally. Throughout my internship, I've greatly expanded my knowledge of both machine learning and Python. Despite

taking courses in both statistics and computer science, the combination of both seemed completely foreign to me. This is now no longer the case. Using Exploratory Data Analysis, I have successfully discovered new information by analyzing graphs and data. By working with FlockAI and WeBots, I now have an understanding of how the principles of machine learning can be applied to complex processes such as flight. In addition, my work with the facial recognition script in Python, helped give me an idea for how similar programs would operate in real life, whether used by drones or not.

Conclusion

Drones are fascinating with the way that they effortlessly seem to move through the air. This hides, however, the sheer amount of work to be performed by the controller to handle whatever conditions may arise. AI and machine learning, though, allow humans to offload this work, and make the process of flying as effortless as it looks. With only simple analysis of data and a few generations of algorithms and training, we can create drones which are able to recognize people with their own sensors and move accordingly. The future is bright for both the fields of machine learning and drones if this rate of development continues, and there certainly doesn’t seem to be any signs of slowing down.