MAE 199

E-Drive Simulator

Shashwat Sparsh

Introduction

Developing multi-rotors for specific tasks may seem trivial given the sheer number of “drones” and drone kits available on the market. This could not be further from the truth. Specific tasks determine what characteristics of the drone are more important than others and this hierarchy of importance then drives the specification of components. It is however daunting to anyone without deep knowledge of multi-rotor or RC propulsion to even know where to begin. At first glance, the Flight Time of a multirotor is directly governed by the multi-rotors power usage. The thrust of the multirotor is directly related to the movements and motions of the drone itself. However it is impossible to know prior to construction what kind of movements and velocities the drone will travel at; there are more variables than equations.

Online calculators such as ecalc.ch provide an entry point for teams such as UAV forge to size their components. However, these calculators effectively black box the math they use to size components and recommended configurations. Later on the results prove to be inaccurate at best and extremely misleading at worst despite their usefulness as a starting point. The aim of this project is to create a proof of concept electronic drive simulator that *does not* black box the computation and its methodology to create estimations for Flight Time and Thrust which can be invaluable for determining which components to buy. It is important to note that the values computed still require an effective thrust stand and test-bed to validate their efficacy. That being said, it is an important first step for all future project teams to employ.

Process

The project process can be broken down into two main phases: determining the variables and the necessary calculations and estimations and then implementing that process in python. Naturally, the majority of the time spent was focused on determining what variables are required in computing Flight Time and Thrust. From there the necessary components are selected. In order to reduce complexity, only the motors and their specifications were varied while all other variables were constant. The components kept constant are outlined in Table 1 while the varied motors are outlined in Table 2.

Component	AirFrame	Battery	ESC	Prop
Specifications	S500 Quadcopter Frame	POVWAY	Fr Sky Neuron 40 Bl-32	EOLO 15*5.5
	454 grams	410 grams	58 grams	21 grams
	4 arms	5200 mAh	40A	15in diameter
		11.1 V	60A Peak	5.5in pitch
		50C, 3S	3-6S	8450 max RPM

Table 1, Constant Variables

Motors	kV	mass
V2806	650	47
V3508	700	97
V4004	300	51
V4006	320	66
V4008	600	105

Table 2, Motors Tested

Because the initial goal of this project was to compute varying TWR and Flight Times for varying component configurations, the two main components to vary would be the Motors and Props. The motors were varied while the Prop was kept because the prop size is typically affected by the size of the airframe; the diameters are thus limited spatially. Motors spatially tend to avoid this problem as the main spatial considerations are to ensure the housing can fit snugly on the mounts. Because the bolts and their spacings for motor mount connections are relatively standard across brands, sizing is not a big concern.

Variables and Definitions

Table 3 outlines the variables, definitions, and corresponding units, used in the calculations. It is important to note that the units for All Up Weight are in grams making it a unit of mass, not weight. This terminology is due to typical usage.

Variable	Definition	Units
T	Thrust	N
RPM	Rotations per Min	
d	Prop diameter	in
pitch	Prop pitch	in
V_o	Forward Air Speed	m/s
AUW	All Up Weight	grams
AAD	Avg Amp Draw	Amps
Pdot	Power/kg required for Lift	150 W/kg
V	Voltage	Volts
B.C.	Battery Capacity	mAh

Table 3, Variables, Definitions, and Units

Equations and Respective Limitations

$$RPM_{Unloaded} = kV_{Motor} * V_{Battery}$$

Equation 1, Unloaded RPM

$$RPM_{Loaded} = RPM_{Unloaded} * k$$

Equation 2, Loaded RPM

$$T = 4.392399E^{-8} * RPM_L * \frac{d^{3.5}}{\sqrt{pitch}} * ((4.23333E^{-4} * RPM_L * pitch) - V_o))$$

Equation 3, Single Motor Thrust

$$T_{Total} = \sum_{0}^{n}(T),\ n \equiv number\ of\ arms$$

Equation 4, Total Thrust

$$AUW = (n * (m_{Prop} + m_{ESC} + m_{Motor})) + m_{Air\ Frame} + m_{Battery} + m_{Flight\ Controller}$$

Equation 5, All Up Weight

$$Total\ Weight = (n * (W_{Prop} + W_{ESC} + W_{Motor})) + W_{Air\ Frame} + W_{Battery} + W_{Flight\ Controller}$$

Equation 6, Total Weight

$$TWR = \frac{Total\ Thrust}{Total\ Weight}$$

Equation 7, Thrust to Weight Ratio

$$AAD = \frac{(AUW * \dot{P})}{Voltage_{Battery}},\ \dot{P} = 120\ to\ 170\ W/kg$$

Equation 8, Average Amp Draw

$$Estimated\ Flight\ Time = \left[\frac{Battery\ Capacity * 0.8}{AAD}\right] * 60$$

Equation 9, Estimated Flight Time

All of these equations have their limitations in use. Their respective caveats are listed below.

For Equation 2, Loaded RPM, the calculator uses an efficiency factor of 0.8 (which is arbitrary). Typically, the Loaded RPM value changes with the directional airspeed of the multirotor and can vary and therefore the thrust will vary as well. An 80% efficiency factor is selected as at hover, the aircraft should be relatively stationary with little impact due to cross winds (Flight, 2022).

For Equation 3, Thrust, the calculator uses a value of 0 for forward air-speed. This is because it is impossible to determine what airspeeds the multirotor would be traveling at in any given environment *prior* to its construction and test. Therefore this Thrust calculation is a *static* thrust calculation. In addition, some of the changing thrust values are accounted for by using the Loaded RPM values in the calculation. In addition, the Thrust calculation is based on the propeller specifications and is only valid for RPMs ranging from 5000 to ~25000. In addition, the props can not have more than 2 blades as the empirical constants were calculated via a dataset with only 2 bladed props. The equation is also hard-coded for atmospheric density which means that the results are only applicable for low altitude flight where the value does not differ much. Furthermore, because all props have cambered airfoils, thrust can still be produced at a 0 angle of attack. The thrust calculation however calculates a value of zero thrust when the pitch speed of the prop is equal to the free-stream velocity. Therefore the thrust calculations are an under-estimate of actual thrust (Staple, 2013).

For Equation 8, Average Amp Draw, a conservative estimate for $\dot{P}$, that is Power required to lift 1kg of weight. This value can range from 120 to 170 and is dependent on each individual drone. For this calculator, the value used is 150 W/kg but stricter estimates are possible by passing alternative values. This value is a rule of thumb and is useful for the purposes of estimation (Bogna, 2018).

For Equation 9, a value of 0.8 is used to ensure that the Battery Capacity is not exhausted beyond that point as it results in degradation of the battery on a chemical level. Again, stricter values can be used to obtain more stringent calculations.

Calculation Process

The computations can be broken into two stages, stage 1 and 2. Stage 1 computes the All Up Weight and then the Average Amp Draw. These are then used to compute the Estimated Flight Time. Stage 2 computes the RPM which is used to compute Thrust. The TWR is then calculated using the Thrust and the AUW. The respective equations have been outlined above.

Implementation

The python implementation of the equations is Object Oriented. Each component type is created with its characteristics and constructed, reducing the overall complexity of the program when new components need to be added. In addition, all functions are compiled in a separate python file and they are passed the component objects as a whole leading to simpler modularity. Finally, the third and last python script actually runs the test-case along with a sample calculation for 5 SunnySky High Efficiency Motors with a single prop.

Because of the design, the program can be run to compute Flight Times and TWRs varying any component or set of components. Currently the main component being varied is the motor. Multiple Props can also be varied in addition to create more charts to effectively narrow down optimal component configurations.

The implementation is saved on GitHub and can be downloaded using the link in the references section.

Results

The main results of this program are three chart types, TWR vs Motor, Flight Time vs Motor, and Flight Time vs TWR vs Motor. These charts provide a proof of concept for the program. They are included below.

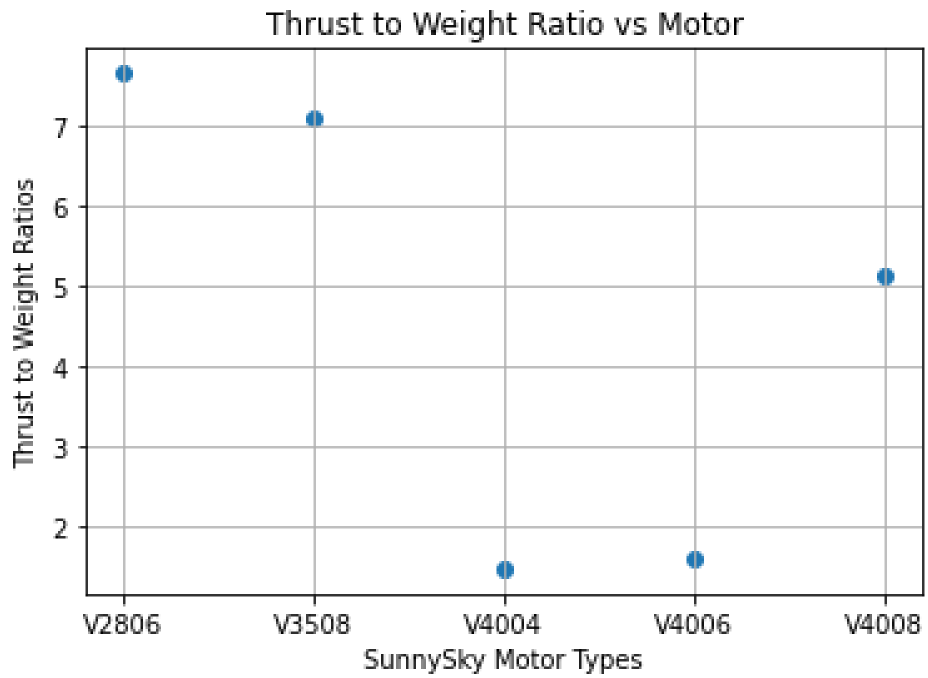

Chart 1, TWR vs Motor Type

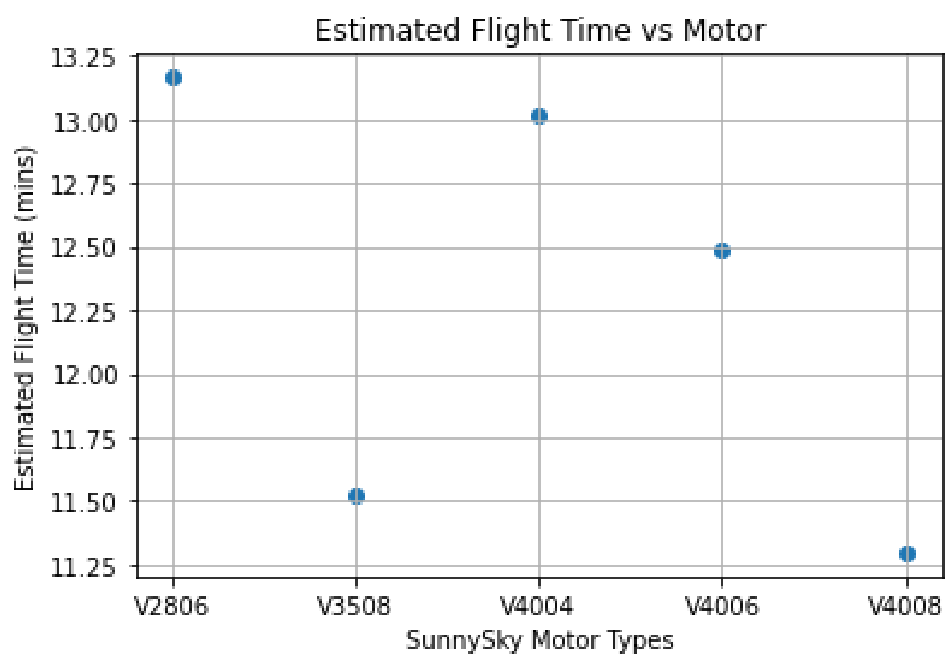

Chart 2, Estimated Flight Time vs Motor

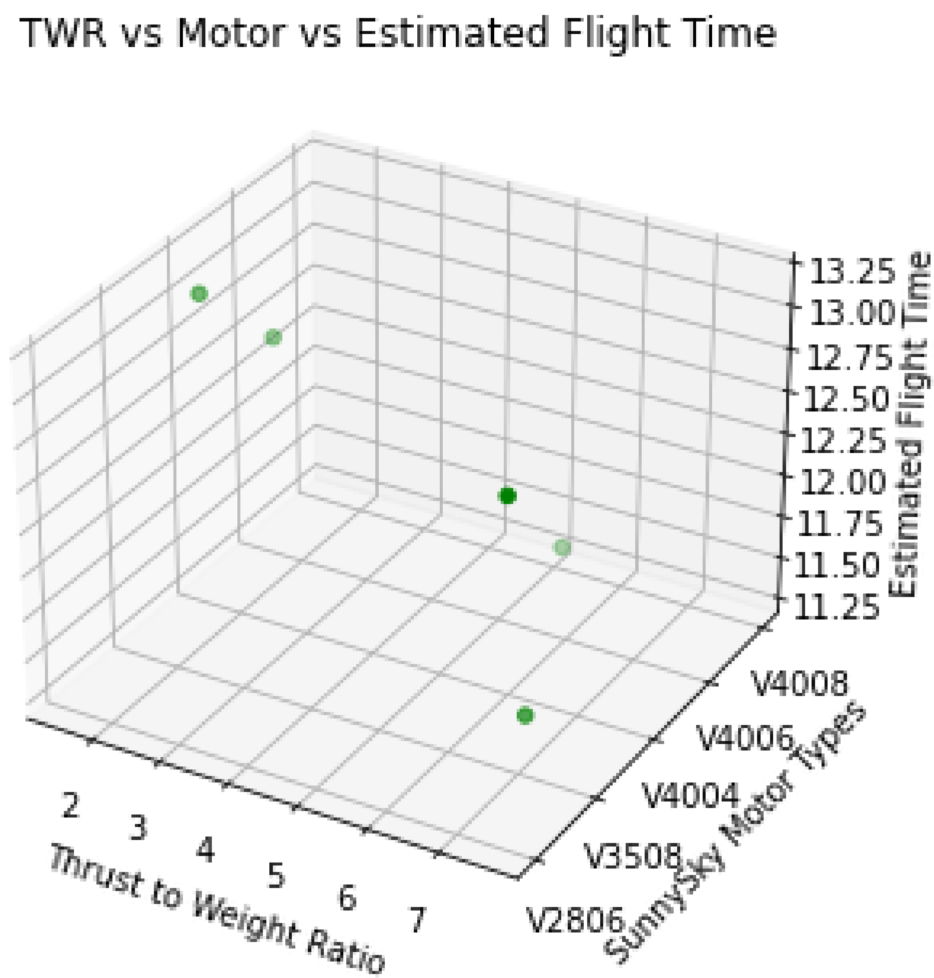

Chart 3, TWR vs Motor vs Estimated Flight Time

Next Steps

The next steps include an implementation that varys both propellers and motors in order to generate 3 new charts: TWR vs Motor vs Prop, Estimated Flight Time vs Motor vs Prop, and TWR vs Estimated Flight Time vs Motor + Prop combination. These three charts can be relatively easily generated as all they require is some simple python loop knowledge.

Another major step would be to construct a test-bed and thrust stand in order to test these varying component configurations and verify their TWR and associated Flight Times.

It is also possible to use an external thrust calculator like QPROP which has a proven history of computing accurate thrust values. These results can be saved as a CSV and then imported into this script for comparison in order to be validated. In addition, these values can also be plotted in the above charts *instead* of the local results if the accuracy of local results falls.

References

Astro, Flight. “Motor Terminology.” *MOTOR TERMINOLOGY*, 2022, https://www.astroflight.com/explanation-of-motor-terminology.html.

Drela, Mark. “QPROP Release.” *MIT*, 2007, https://web.mit.edu/drela/Public/web/qprop/.

“ECalc - Drive Calculator.” Edited by ECalc, *ECalc*, 2004, https://ecalc.ch/.

Scout, UAV. “Scout UAV.” *Scout UAV RSS*, Scout UAV, 2022, https://www.scoutuav.com/category/guide/power-system/calculate-flight-time/.

Sparsh, Shashwat. *Electronic Drive Simulator, MAE 199*, Shashwat Sparsh, 2022, https://github.com/shashwatSparsh/Electronic-Drive-Simulator.

Staples, Gabriel. “Propeller Static & Dynamic Thrust Calculation - Part 1 of 2.” *ElectricRCAircraftGuy.com--RC, Arduino, Programming, & Electronics*, 1 Jan. 2013, https://www.electricrcaircraftguy.com/2013/09/propeller-static-dynamic-thrust-equation.html.

Staples, Gabriel. “Propeller Static & Dynamic Thrust Calculation - Part 2 of 2 - How Did I Come up with This Equation?” *ElectricRCAircraftGuy.com--RC, Arduino, Programming, & Electronics*, 1 Jan. 2014, https://www.electricrcaircraftguy.com/2014/04/propeller-static-dynamic-thrust-equation-background.html.

Szyk, Bogna. “Drone Flight Time Calculator.” *Omni Calculator*, Omni Calculator, 7 June 2018, https://www.omnicalculator.com/other/drone-flight-time.