Development of an Energy Model for Quadcopter Package Delivery Drones

Thiago A. Rodrigues[1], Jay Patrikar[2], Bastian Wagner[3], Sebastian Scherer[2], Constantine Samaras[1]

***Abstract*— The paper presents analytical models to estimate the energy usage of drone-based system focused towards last-mile delivery operations in urban areas. The detailed model also explicitly takes into account factors like wind that make it more suitable for real world deployment. Experimental test are conducted to collect data to regress the coefficients of the analytical results and accuracy is reported.**

I. ENERGY MODEL

The energy model developed takes into account a first-principle analysis based on helicopter aerodynamics. All the components were considered regarding the drone's body frame as described next.

A. Coordinate Frames

There are many coordinate frames that can be used to understand the dynamic behavior of a UAV ([**?**]. For this study we will work with the inertial frame, vehicle frame, body frame, stability frame and wind frame

The *inertial frame* has a fixed latitude-longitude origin and uses the North-East coordinates as **i** and **j** directions and the **z** axis towards the center of the Earth (Figure 1). Similarly, the *vehicle frame* uses the same the north-east-center-of-the-earth directions, but with its origin fixed on the UAV's center of mass (G) (Figure 2).

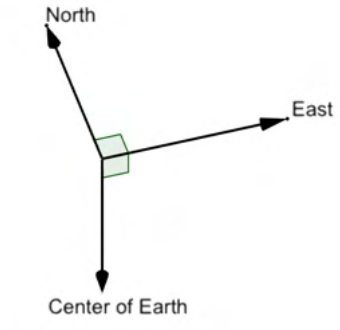

Fig. 1. Inertial Frame

The *body frame*, on the other hand, considers the **i**, **j** and **z** directions regarding the the body of the aircraft. For instance, the x axis was defined as the direction along the power button and the mass center of the drone used (Matrice 100). The angle between the i^v (North) and i^b on the i^vGj^v plane is defined as ψ (Figure 3a). The angle between i^b and i^v on the i^vGk^v plane is defined as θ (Figure 3b). Finally, the angle between j^v and j^b on the j^vGk^v plane is defined as ϕ (Figure 3c).

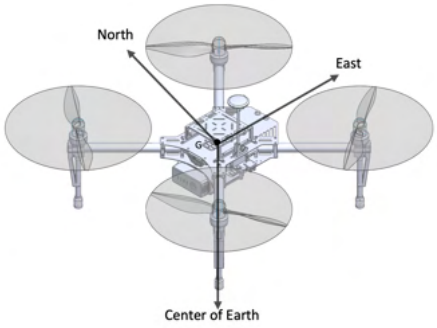

Fig. 2. Vehicle Frame

The *stability frame* correspond to the frame of air surrounding the aircraft. The airspeed vector (V_a), defined as the difference between the ground velocity (V_g) and the wind velocity (V_w) flows through the aircraft at a i^sGj^s plane, that is rotated from the body frame reference by α, the angle of attack. Finally, the wind frame assumes the **i** axis flowing throughout the airspeed vector. The angle between i^s and V_a is defined as β (Figure 4).

With the main working frames described, it is possible to transcribe any vector written as $\mathbf{A} = a_x\mathbf{i} + a_y\mathbf{j} + a_z\mathbf{k}$ from frame F^0 to frame F^1 using a rotation matrix (R_0^1).

$$\mathbf{A^1} = R_0^1\mathbf{A^0} \tag{1}$$

For this work, we have established the body frame as the standard working frame. Therefore, all vectors were transcribed to the body frame using the following rotation matrices.

From the vehicle frame to the body frame (R_v^b) the rotation matrix is given as:

$$R_v^b = \begin{bmatrix} c_\theta c_\psi & c_\theta s_\psi & -s_\theta \\ s_\phi s_\theta c_\psi - c_\phi s_\psi & s_\phi s_\theta s_\psi + c_\phi c_\psi & s_\phi c_\theta \\ c_\phi s_\theta c_\psi + s_\phi s_\psi & c_\phi s_\theta s_\psi - s_\phi s_\psi & c_\phi c_\theta \end{bmatrix}$$

Where c_θ, c_ψ and c_ϕ are the cosines of the angles θ, ψ and ϕ, respectively, and s_θ, s_ψ and s_ϕ are the sines of the angles θ, ψ and ϕ, respectively.

This work was supported by LORD MicroStrain Sensing Systems and the Department of Energy Grant DE-EE0008463

[1] Department of Civil and Environmental Engineering. Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, USA

[2] Robotics Institute, School of Computer Science, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, USA

[3]Airbus Defence and Space GmbH, Rechliner Str., 85077 Manching, Germany

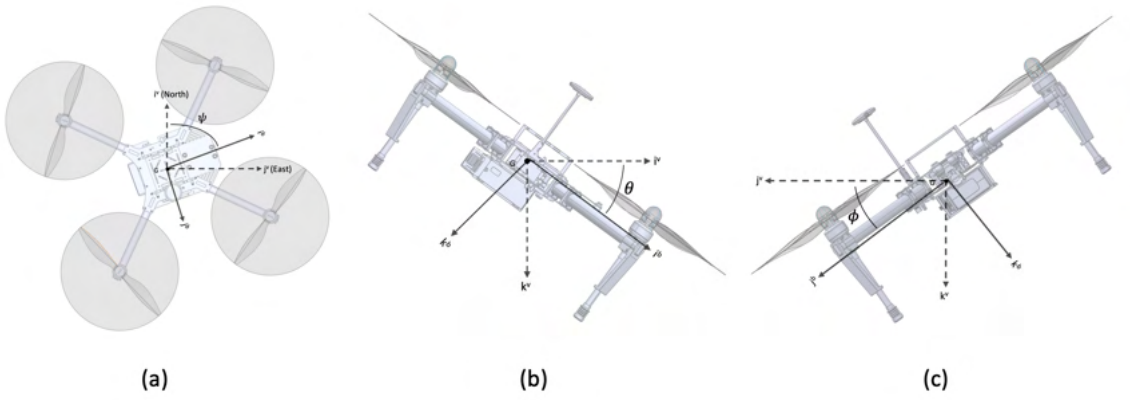

Fig. 3. Body Frame

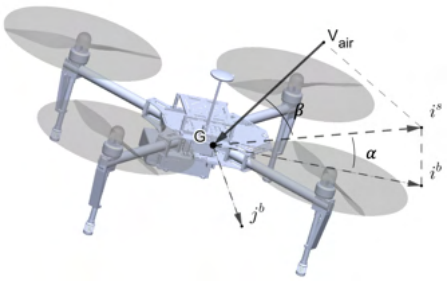

Fig. 4. Stability and Wind Frame

The transformation from the stability frame to the body frame is given by the matrix R_s^b:

$$R_s^b = \begin{bmatrix} cos(\alpha) & 0 & -sin(\alpha) \\ 0 & 1 & 0 \\ sin(\alpha) & 0 & cos(\alpha) \end{bmatrix}$$

And the transformation from wind frame to the body frame is given by the matrix R_w^b:

$$R_w^b = \begin{bmatrix} cos(\beta)cos(\alpha) & -sin(\beta)cos(\alpha) & -sin(\alpha) \\ sin(\beta) & cos(\beta) & 0 \\ cos(\beta)sin(\alpha) & -sin(\beta)sin(\alpha) & cos(\alpha) \end{bmatrix}$$

B. First Principle Analysis

The power required (P) by a quadcopter drone can be divided into induced power (P_i), profile power (P_p), parasitic power (P_d) and ancillary power (P_a).

$$P = P_i + P_p + P_d + P_a \tag{2}$$

1) Induced Power: The induced power(P_i) represents the power required to overcome the force of gravity to keep the aircraft in the air. The P_i can vary according to the flight maneuver (Rotaru and Todorov 2017). In order to calculate P_i it is necessary to understand how Thrust (T) is estimated for a drone propeller. Most of the current models provide T for a 2D approach, where β = 0 (Figure 5), so that T is given as:

$$T = 2\rho A v_i \sqrt{(V_{air} cos(\alpha)^2 + (V_{air} sin(\alpha) + v_i)^2} \tag{3}$$

where, ρ = air density; v_i = induced velocity; V_{air} = airspeed vector; and A = total area covered by the propellers.

$$A = M\pi R^2 \tag{4}$$

where, M = number of propellers; R = radius of the propeller.

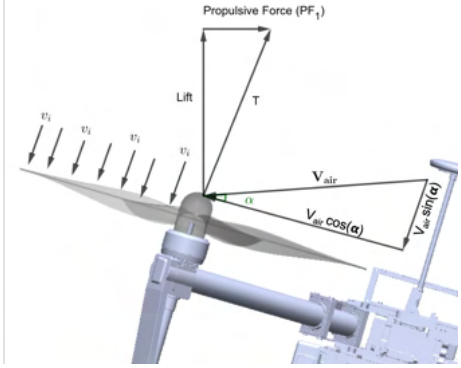

Fig. 5. Thrust, β = 0

However, once $\beta > 0$, T must be recalculate. Using the same approached used by Rotaru and Todorov (2017), and adding β to the relative position of the air flowing through the aircraft, T becomes a resultant of a three-dimension force system (Figure 6).

Therefore, T might be calculated as:

$$T = 2\rho A v_i \sqrt{V_{air_{j^b k^b}}^2 + V_{air_{i^b k^b}}^2 + (V_{air_{i^b j^b}} + v_i)^2} \tag{5}$$

where, $V_{air_{i^b j^b}}$ represents the airspeed vector flowing perpendicularly to the propeller, and $V_{air_{j^b k^b}}$ and $V_{air_{i^b k^b}}$ the components acting on the propellers plane. They can be written as a function of V_{air}, α and β:

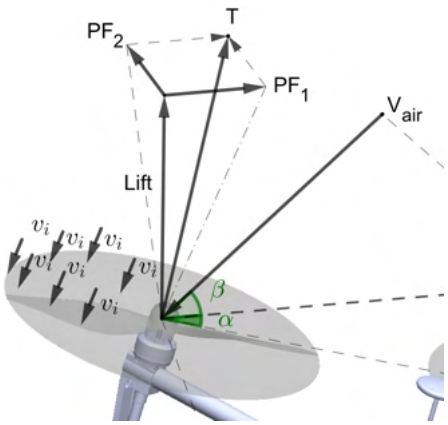

Fig. 6. Thrust, $\beta > 0$

$$V_{air_{j^b k^b}} = V_{air}\sqrt{\frac{\sin^2(\alpha)\sin^2(\beta) - \sin^2(\beta)}{\sin^2(\alpha)\sin^2(\beta) - 1}} \tag{6}$$

$$V_{air_{i^b k^b}} = V_{air}\frac{\cos(\alpha)\cos(\beta)}{\sqrt{\sin^2(\alpha)\cos^2(\beta) + \cos^2(\alpha)}} \tag{7}$$

$$V_{air_{i^b j^b}} = V_{air}\sqrt{\frac{\sin^2(\alpha)\sin^2(\beta) - \sin^2(\alpha)}{\sin^2(\alpha)\sin^2(\beta) - 1}} \tag{8}$$

Therefore, P_i for a moving forward flight regime is:

$$P_i = T\left(V_{air}\sqrt{\frac{\sin^2(\alpha)\sin^2(\beta) - \sin^2(\alpha)}{\sin^2(\alpha)\sin^2(\beta) - 1}} + v_i\right) \tag{9}$$

T can be calculated as a function of θ, ϕ and the Rotor Lift (L), which in turn can be found as a function of the drone's weight (W), the Wind Force and the angles θ and ϕ.

$$T = L\sqrt{\frac{\sin^2(\phi)\cos^2(\theta) + \cos^2(\phi)}{\cos^2(\phi)\cos^2(\theta)}} \tag{10}$$

Assuming a constant V_{air}, L is:

$$L = W + F_{drag,Wind}\sqrt{\frac{\cos^2(\phi)\cos^2(\theta)}{\sin^2(\phi)\cos^2(\theta) + \cos^2(\phi)}} \tag{11}$$

where, W is the drone's weight (W = mg) and $F_{drag,Wind}$ is the drag caused by the wind blowing perpendicularly to the propellers.

$$F_{drag,Wind} = \frac{C_{d_{sup}}\rho A_{sup} V^2_{air_{i^b j^b}}}{2} \tag{12}$$

where, $C_{d_{sup}}$ is the drag coefficient of the air blowing through the drone's upper surface (Figure 3a), ρ is the air density, A_{sup} is the reference area and $V_{air_{i^b j^b}}$ is the airspeed vector blowing perpendicularly the propeller.

Therefore, v_i can be calculated through the implicit equation:

$$v_i = \frac{\frac{T}{2\rho A}}{\sqrt{V^2_{air_{j^b k^b}} + V^2_{air_{i^b k^b}} + (V_{air_{i^b j^b}} + v_i)^2}} \tag{13}$$

During a hovering condition α = 90°, β = 0° and V_{air} = 0. Therefore, P_i will depend only on the induced velocity (v_i) and thrust (T).

$$P_i = T \cdot v_i \tag{14}$$

v_i for hovering can be simplified as:

$$v_i = \sqrt{\frac{T}{2\rho A}} \tag{15}$$

For a steady take off regime, α = 90° and β = 0°. Thus, P_i can be defined as:

$$P_i = T(v_i + V_{air}) \tag{16}$$

v_i for a take off regime can be defined as:

$$vi = -\frac{V_{air}}{2} + \sqrt{\frac{V^2_{air}}{4} - \frac{T}{2A\rho}} \tag{17}$$

For a rapid steady landing regime, with α = 90°, β = 0° and $|V_{air}| \geq 2\sqrt{\frac{T}{2\rho A}}$, P_i can be defined as:

$$P_i = T(v_i + V_{air}) \tag{18}$$

and,

$$vi = -\frac{V_{air}}{2} + \sqrt{\frac{{V_{air}}^2}{2} + \frac{T}{2\rho A}} \tag{19}$$

Table 1 shows a summary of the P_i forms,

2) Profile Power: The Profile Power (P_P) represents the power necessary to overcome the drag from the rotating propeller blades (Liu, Sengupta, and Kurzhanskiy 2017). P_p for a hover regime is calculate as:

$$P_{p,hover} = \frac{Ncc_{d_{blade}}\rho R^4}{8}\Omega^3 \tag{20}$$

where, N = number of blades in a single propeller; c = blade chord width; $c_{d_{blade}}$ = drag coefficient of the blade; Ω = angular speed

For a horizontal flight, P_p becomes:

$$P_p = P_{p,hover}(1 + \mu^2) \tag{21}$$

where, μ = advance ratio for the propeller;

$$\mu = \frac{\sqrt{V^2_{air_{j^b k^b}} + V^2_{air_{i^b k^b}}}}{R\Omega} \tag{22}$$

Assuming that thrust is linearly proportional to the angular speed squared ($T = k\Omega^2$), where k is a scaling factor

TABLE I
INDUCED POWER

Regime	Induced Power (P_i)	Induced Velocity (v_i)
Hover	$P_i = T \cdot v_i$	$v_i = \sqrt{\frac{T}{2\rho A}}$
Steady Take Off	$P_i = T(v_i + V_{air})$	$vi = -\frac{V_{air}}{2} + \sqrt{\frac{V_{air}^2}{4} - \frac{T}{2A\rho}}$
Rapid Landing	$P_i = T(v_i + V_d)$	$vi = -\frac{V_{air}}{2} + \sqrt{\frac{{V_d}^2}{2} + \frac{T}{2\rho A}}$
Cruise	$P_i = T\left(V_{air}\sqrt{\frac{\sin^2(\alpha)\sin^2(\beta) - \sin^2(\alpha)}{\sin^2(\alpha)\sin^2(\beta) - 1}} + v_i\right)$	$v_i = \frac{\frac{T}{2\rho A}}{\sqrt{V^2_{air_{j^b k^b}} + V^2_{air_{i^b k^b}} + (V_{air_{i^b j^b}} + v_i)^2}}$

converting rotor angular speed to thrust, the total profile power is

$$P_P = \sum_{i=1}^{M}\left(\frac{Ncc_{d_{blade}}\rho R^4}{8}\Omega^3(1+\mu^2)\right) \tag{23}$$

$$P_P = c_2 T^{3/2} + c_3\left(V^2_{air_{j^b k^b}} + V^2_{air_{i^b k^b}}\right)T^{1/2} \tag{24}$$

where, $c_2 = \frac{Ncc_{d_{blade}}\rho R^4}{8k^{3/2}}$ and $c_3 = \frac{Ncc_{d_{blade}}\rho R^2}{8k^{1/2}}$

3) Parasitic Power: The parasitic power (P_d) is the power required to overcome the drag force resulting from the movement of the multicopter body through the air.

$$P_d = \sum F^b_{drag} V^b_{air} \tag{25}$$

Where, $F_{drag}{}^b$= drag force on the multicopter body, and V^b_{air} = airspeed vector acting on the body frame.

$$F^b_{drag} = \frac{1}{2} C^b_d \rho A^b_{ref} (V^b_{air})^2 \tag{26}$$

Where, C^b_d = drag coefficient of the drone's body; A^b_{ref} = reference area

Therefore,

$$P_d = \sum \frac{1}{2} C^b_d \rho A^b_{ref} (V^b_{air})^3 \tag{27}$$

P_d will be, therefor, a sum of three parasitic powers, $P_{d_{i^b k^b}}$, $P_{dj^b k^b d}$ and $P_{di^b j^b}$.

4) Ancillary Power: The ancillary power (P_a) is the power required to run all the electronic devices and sensors used for the flight navigation systems and data collection.

Although P_a may vary under extreme operation conditions, such as extreme high or low temperatures, it has been assumed that the Pa will be relatively constant throughout the flight as long as the number and specifications of the electronic devices and sensors remain the same.

5) Energy Efficiency: The energy efficiency of travel (e) is calculated dividing the total power (P) by the ground speed of the drone (V_g) (Stolaroff et al. 2018):

$$e = P/V_g \tag{28}$$

And the total energy consumed for a delivery trip:

$$E = (e_{loaded} + e_{unloaded})d^v \tag{29}$$

Where, d^v = one-way distance (Inertial Frame)

II. EXPERIMENT

A series of flights was performed to empirically measure the energy consumption of a quadcopter UAV in order to evaluate the accuracy of the energy model developed. An experimental protocol (Appendix A) was created and followed to ensure a reliable approach for data acquisition.

A quadcopter DJI Matrice 100 (M100) was equipped with a sensor suite to measure the energy usage. We use a FT Technologies FT205 UAV-mountable, pre-calibrated ultrasonic wind sensor. The sensor is accurate up to $\pm 0.1 m/s$ and has a refresh rate of 10Hz. The on-board current and voltage are measured using a Mauch Electronics PL-200 sensor. This sensor is based on the Allegra ACS758-200U hall current sensor, and can record currents up to 200A and voltages up to 33V. We record the state of the system (position/velocity/orientation) using the $3DM - GX5 - 45$ GNSS/INS sensor pack. These sensors use a built-in Kalman filtering system to fuse the GPS and IMU data. The sensor was capped at to an output rate of 10Hz. The flights were performed in a pre-established route (Figure 9) varying altitude (25 m, 50 m, 75 m and 100 m), speed (4 m/s, 6 m/s, 8 m/s, 10 m/s and 12 m/s) and payload weight (no payload, 250g and 500 g). Each combination was repeated three times, totaling 180 flights. Figures 7 and 8 show the experimental platform and the sensor suite respectively.

III. DATA ANALYSIS

The data provided by each sensor were synchronized to a frequency of approximately 5Hz using the ApproximateTime message filter policy of ROS.

A. Defining Flight Regime Intervals

The combined data had then to be divided into the different flight regimes for a better analysis of the energy consumption for each stage. Given the high number of flights performed an algorithm was created to automatically identify the time interval were the flight was under Take-Off, Cruise or Landing regime.

An example of the drone's altitude versus time (Figure 10) shows very clearly when all the three flight regimes occur.

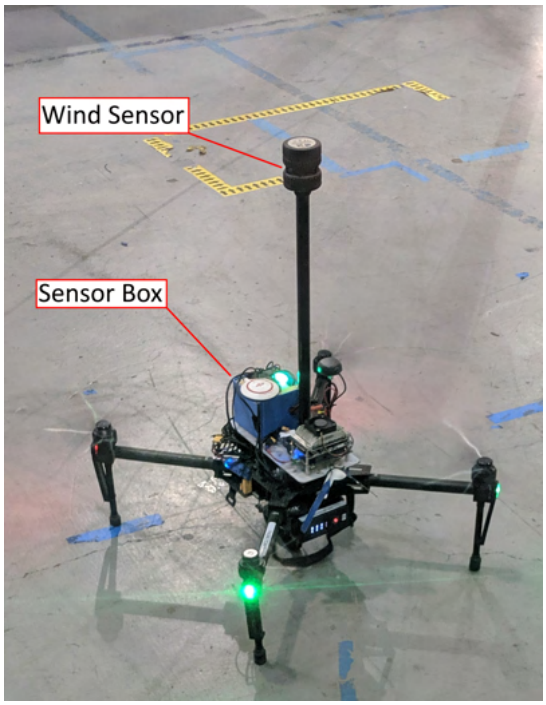

Fig. 7. The test platform

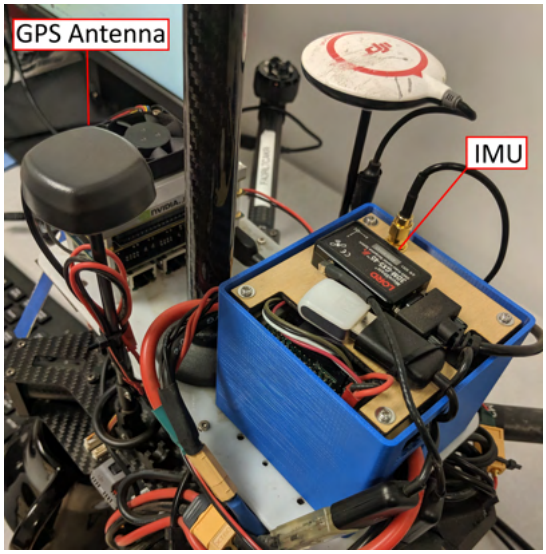

Fig. 8. The sensor suite

The steady climbing movement in the first part of the flight represents the Take-Off regime. The relative constant altitude in the middle of the graphic correspond to the Cruise regime. Lastly, the steady decent movement at the end of the flight represents the Landing regime.

Although defining the interval where the three regimes occur seems to be quite intuitive for human eyes, the inherent variability of the data provided by the GPS make the process less efficient for an algorithm. Therefore, a Gaussian Filter was applied to the data in order to reduce eventual noises. A sigma of 5 was used, as it reflects approximately the accuracy of the GPS measurements. The result (Figure 11), is much

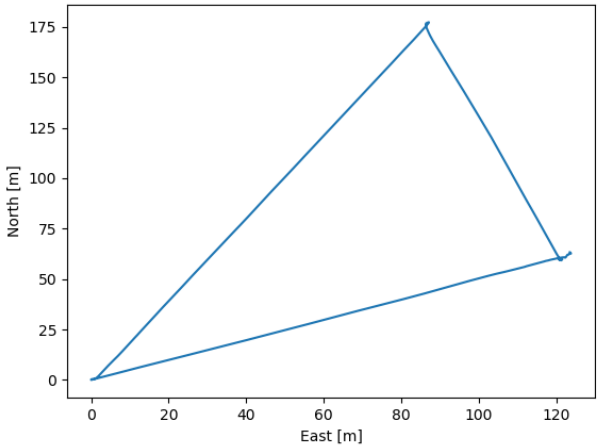

Fig. 9.

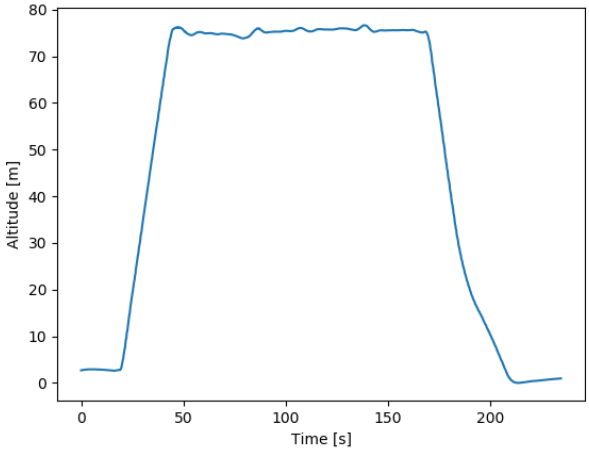

Fig. 10. Altitude over time

smoother curve.

Once the Gaussian filter is applied to the altitude curve, the first derivative provides the points of maximum and minimum altitude variations (Figure 12) that will be used to define the beginning and ending of each regime.

Then, it was established that the Take-Off regime would start when the slope reached 0.05 for the first time at the beginning of the flight. Moreover, the Take-Off regime would finish once the slope reached 0 (zero), parting from the maximum slope point. Similarly, the beginning of the Landing regime was defined to be the first time the slope reached 0 moving backwards from the minimum slope point, and the finishing point of the Landing regime when the slope reached 0 moving forward from the minimum slope point. The Cruise regime is defined between the Take-Off and Landing regimes.

The result, shown in Figure 13, allows the automatic identification of the time interval where the three flight regimes throughout the flights.

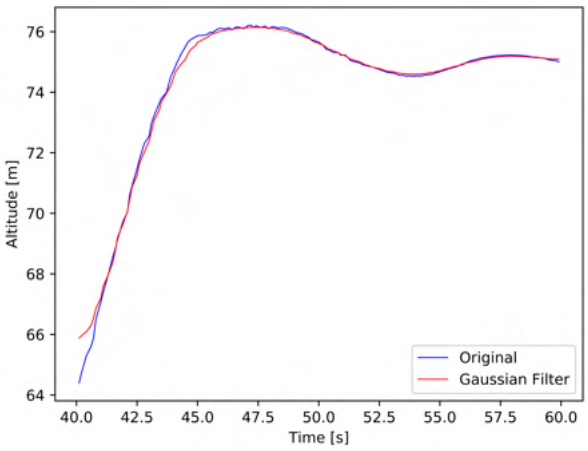

Fig. 11. Gaussian filter applied to a portion of the altitude data

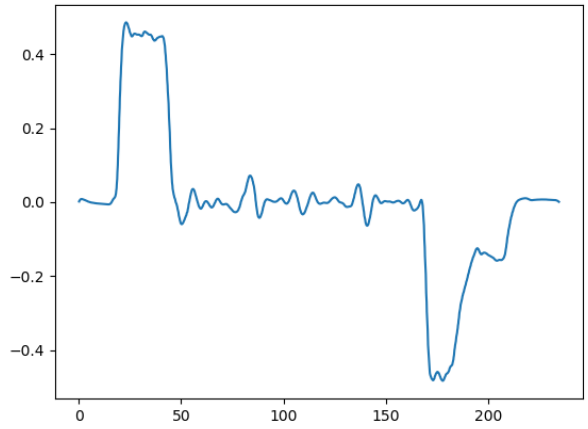

Fig. 12. First derivative of the altitude over time

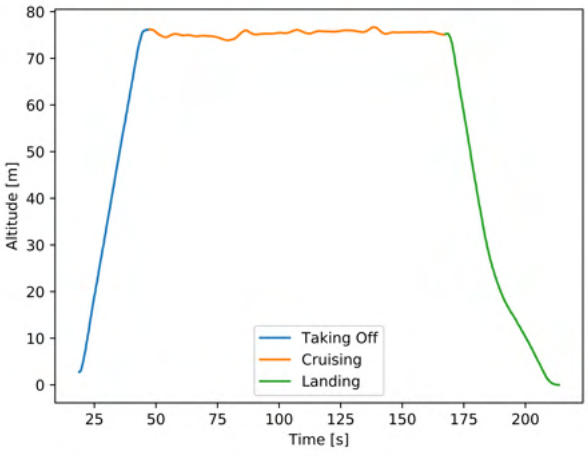

Fig. 13. Flight Regimes

B. Energy Model Validation

Multiple parameters are needed for the model to depict the real world. Because these parameters are unknown, one has to estimate these parameters based on real-world test data. This section describes the method used to regress these parameters.

1) Least Squares Method: The goal of this method is to variate the parameters of a model in such a way that the sum of the squared distances of the model generated data and the measured data is minimized. This can be expressed with the following formula:

$$D = \sum_{i=1}^{n} [y_i - f(\vec{x}_i, \vec{\beta})]^2 \tag{30}$$

with y_i being the measured value at i, $f(\vec{x}_i, \vec{\beta})$ being the function of the model and $\vec{\beta}$ being a vector containing the parameters of the model and $\vec{x}_i$ being a vector of input variables for the model at i.

To actually estimate the parameters using this method, partial derivatives of D with respect to each parameter in $\vec{\beta}$ has to be taken and solved for zero. The derivatives can be represented using an Jacobian matrix.

This matrix can be used in the Levenberg-Marquadt algorithm to estimate the parameters. Implementations of the algorithm also estimate the Jacobian matrix due to the complexity this matrix can assume. When finding the minimum sum of a squared function with one or multiple parameters, such as the least squares function described above, multiple algorithms can be used to find the optimal parameters minimizing this sum. One of these algorithms is the Levenberg-Marquardt algorithm. It combines the Gauss-Newton and Gradient-Descent algorithms to compute the parameters faster and with greater accuracy compared to only using one of the algorithms. As the other algorithms, the Levenberg-Marquardt algorithm only finds a local minimum which might change depending on the initial parameter guesses. This is due to the algorithm only checking in the vicinity of the initial guess.

$$[J^T W J + \lambda I] h_{lm} = J^T W (y - \hat{y}) \tag{31}$$

The equation 31 describes a step of the Levenberg-Marquardt algorithm. The leftmost part in the bracket is the Gaussian part of the algorithm, accompanied by the Gradient-Descend part of the algorithm. The factor λ is either in- or decreased in each step based on the whether the solution gets more optimal or not. In normal conditions, the factor is decreased in each iteration but will be increased when the solution gets worse. The higher λ is, the higher the influence of the gradient descend part will be.

C. Discussions

The gathered data shows that the average power consumption for each flight, depending on the altitude, speed and payload, lies between 700W and 800W. During the analysis of the data it became clear that the data provided by the sensor has a high fluctuation and therefore had to

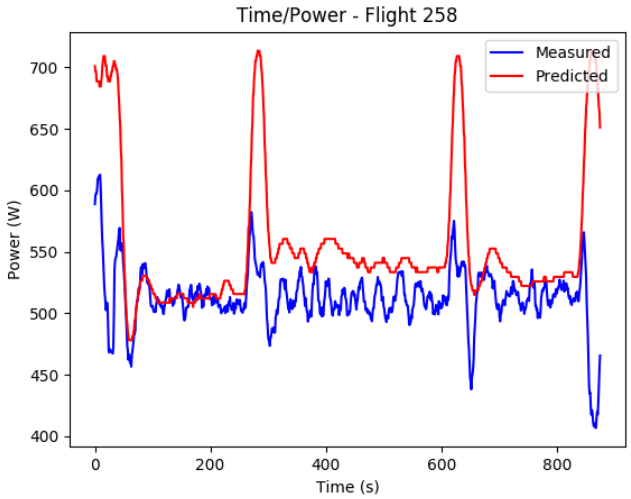

Fig. 14. Lowest average Power Consumption deviation in the physical model. The average deviation is 8.66%

be smoothed with a moving average filter. By analyzing all datapoints gathered during the flights it becomes clear that the power consumption is proportional to the speed and altitude of the flight. One can also see that the climb and descend phases of the flight only account for a small portion of the energy consumption due to the energy consumption on the different altitude only differs slightly. The energy consumption on the slower flights is significantly higher than on the faster flights, but including the findings of the motor characterization, flying at even higher speeds than 12 m/s will result in a higher energy consumption due to the efficiency curve of the motors. One drawback on flying with higher speeds is that the drone only reaches those speeds for a small amount of time on the given flight path in the outdoor testing facility. Figures 14 shows the results of the physical energy model for a representative flight. A model accuracy of 80% was achieved. The average deviation using the physical model was 13.87%.

IV. Conclusions

The goal of this paper was to design a model to predict the energy consumption of an aerial drone. Flight tests were carried out to collect data to regress the coefficients of the proposed model and a accuracy of 80% was recorded. Future work involves collecting more random test datasets and comparing against blackbox approaches that use deep recurrent neural networks (RNNs). The project will also extend to included other types of aerial systems like VTOLs and fixed wing aircrafts to provide a unified framework for aerial vehicle energy estimations.