Utah NASA Space Grant Consortium Fellowship Symposium
Brigham Young University

A Low-Cost Trajectory Estimation System for Drones and Rockets

Stephen A. Whitmore* and Tyler J. Gardner†
Utah State University, Logan, Utah, 84322-4130

Drone aircraft have become very popular during the last few years. These remote-controlled aircraft often rely on GPS receivers to track their position and velocity. However, it is often important for aircraft to measure other information that GPS cannot provide. Recent developments in single-board computers have facilitated the creation of cheap and effective embedded systems. This paper details the development and testing of a Raspberry Pi-based low-cost trajectory estimation system. Data acquired from a pitot probe, GPS receiver, altimeter, and accelerometer are fed into a Kalman filter to calculate a best-fit trajectory. The performance of the filter algorithm was evaluated using test case data. Further testing of the hardware and estimator performance will be performed during various future rocket test flights.

I. Nomenclature

x_{bf}	=	Body-fixed x-axis
y_{bf}	=	Body-fixed y-axis
z_{bf}	=	Body-fixed z-axis
x_n	=	NED x-axis (North)
y_e	=	NED y-axis (East)
z_d	=	NED z-axis (Down)
ϕ	=	roll (measured from local horizontal)
θ	=	pitch (measured from local horizontal
ψ	=	yaw (measured clockwise from true north)
p	=	roll rate
q	=	pitch rate
r	=	yaw rate
M	=	North-east-down to body-fixed rotation matrix
F	=	force
m	=	mass
g	=	gravitational acceleration (magnitude)
g_0	=	standard scale factor (9.81 m/s^2)
u, v, w	=	velocity components along body-fixed axes
A_x	=	x component of acceleration
A_y	=	y component of acceleration
A_z	=	z component of acceleration
μ	=	= gravitational parameter
$\Omega_\oplus$	=	angular velocity of the earth
$R_\oplus$	=	radius of the earth corrected for bulge
R_{eq}	=	radius of the earth at the equator
e	=	spherical eccentricity of the earth

II. Introduction

OVER the past few years, there has been tremendous growth in the sales and production of Unmanned Aerial Vehicles (UAVs) more commonly known as drones. In the past, UAVs and drones have been used largely by military and government organizations because of the cost and size of these systems. However, many low-cost and smaller UAV systems are now available to consumers. These remote-controlled drone aircraft have become very popular among photographers, researchers, and general hobbyists. Research groups and universities have also developed their own UAV systems, many of which are autonomous, to perform low-cost aerial surveys [1] [2].

Drones often rely on GPS receivers to track their position and velocity. However, it is often important for aircraft to measure other flight information that GPS cannot provide (such as airspeed). This makes it difficult for these aircraft to capture a complete picture of their flight behavior. Work has been conducted to investigate the use of cameras and visual tracking to control drones. Some of these systems have been designed to operate on-board an aircraft to track obstacles [3]. Other systems use external tracking devices to keep track of the vehicle's trajectory and process tracking data [4]. These methods have been targeted towards small, relatively slow, quad-rotor drones and only provide obstacle-avoidance benefits.

Larger and faster UAVs will need more information on airspeed and trajectory, especially if they have a fixed-wing design. Many organizations and companies are currently working to develop autonomous drones for use in delivery systems. While companies such as Amazon and UPS may get most of the focus for their autonomous drone development efforts, some organizations are actually using drone-based delivery systems right now [5]. As efforts such as these to develop autonomous drone aircraft continue, it will be necessary to develop a more complete trajectory and airspeed estimation system. One simple way to do this is to process available GPS and accelerometer data in a filter and then add a pitot tube to provide an airspeed measurement.

This paper details the development of a Raspberry-Pi-based trajectory estimation system for drones and rockets. This system will use the Raspberry Pi Zero W shown in Fig. 1. The Raspberry Pi architecture was chosen for this system because of its low cost, wide popularity, and extensive documentation. The Zero W model was selected to save weight and space.

Two main questions will be investigated throughout this project. First, how well can the selected Raspberry Pi run a trajectory estimation system? A Kalman filter has been developed to run on the Raspberry Pi and estimate the trajectory of a rocket as detailed in section IV. The filter will keep track of the rocket's position and velocity and a pitot probe will provide an airspeed measurement to keep track of the rocket's Mach number. The Kalman filter algorithm was tested in MATLAB using a test case data set. This test case, and the results of the algorithm test, are presented in section V.

Second, how well does the Raspberry Pi operate under high acceleration loads? Testing this system on a rocket will allow the system to experience a wide-range of dynamics and conditions. Flight tests of the trajectory estimation system will be performed this summer. The predicted trajectory of these tests flights is shown in section VI.

III. Instrumentation

This section details the instrumentation used is this trajectory estimation system. Two altimeters, the RAVEN 4 and the TeleMetrum v2.0, will be used to provide altimeter data and deployment charge ignition. IMU data will be provided by the LORD MicroStrain 3DM-CV5-25 IMU. Various pressure measurements will be taken by the PX142-030A5V. These devices will be connected to the Raspberry Pi Zero W to provide data to the Kalman filter. Table 1 provides a summary of key specifications for the selected instruments.

As mentioned, the Raspberry Pi Zero W[a] is the primary device that this project is focused on. Like other Raspberry Pi devices, the Zero W is capable of running various distributions of Linux. However, its 1 GHz single-core and 512 MB of RAM can be easily clogged up by GUI

*Professor, Mechanical and Aerospace Engineering Department, 4130 Old Main Hill, AIAA Associate Fellow.

†Graduate Research Assistant, Mechanical and Aerospace Engineering Department, AIAA Student Member.

[a]https://www.adafruit.com/product/3400

operating systems (OS). Therefore, the Zero W will run in a console-only mode for this project to limit OS overhead. Besides the Zero W's small size (66.0mm by 30.5mm by 5.0mm or 2.6" by 1.2" by 0.2") and weight (9.3g or 0.3oz), this computer also consumes relatively little power (0.6-1.2W).

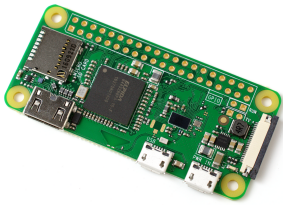

Figure 1. Picture of the Raspberry Pi Zero W which will run the Kalman filter.

The RAVEN 4[b] Altimeter shown in Figure 2 includes a 105G accelerometer (400Hz axial and 200Hz lateral) and a dual-deploy charge ignition controller. Primary charge ignition will be controlled by the RAVEN 4. The RAVEN's on-board high-rate data storage will be used to log flight data independently of the Zero W and flight telemetry.

Figure 2. Picture of the RAVEN 4 Altimeter which will provide primary charge ignition and flight logging.

Flight telemetry will be broadcast by the TeleMetrum v2.0[c] on the 70cm (440 MHz) amateur radio band during flight. The TeleMetrum v2.0, shown in Figure 3 includes an integrated accelerometer, altimeter, and charge ignition controller. The TeleMetrum will provide altimeter data to the Raspberry Pi and secondary charge ignition for the parachute deployment charges.

Figure 3. Picture of the TeleMetrum v2.0 Altimeter which will broadcast flight telemetry and provide altimeter readings.

The LORD MicroStrain 3DM-CV5-25 IMU[d] shown in Figure 2 is a compact and light-weight Inertial Measurement Unit (IMU) that provides sample rates upwards of 1000Hz. The LORD MicroStrain includes an adaptive Extended Kalman Filter with output rates up 500Hz. These features allow the Kalman filter on the Raspberry Pi to process IMU measurements without the need for additional filtering.

Figure 4. Picture of the LORD MicroStrain 3DM-CV5-25 which will provide accelerometer and gyroscope data.

[b]https://www.featherweightaltimeters.com/raven-altimeter.html
[c]https://altusmetrum.org/TeleMetrum/
[d]https://www.microstrain.com/inertial/3dm-cv5-25

Pressure measurements at the rocket's nose and inside the avionics bay will be taken during flight by the PX142-030A5V pressure transducer[e] shown in Figure 5. Because this transducer produces a high-level output of 1-6Vdc, an analog-to-digital converter will be used to connect the transducers to the GPIO on the Raspberry Pi. The PX142-030A5V includes temperature compensation and offers a response time of 1ms.

Figure 5. Picture of the PX142-030A5V pressure transducer.

Table 1. Summary of instrumentation specifications.

RAVEN 4 Altimeter	
Pressure Error	< 1mbar (0.1% FS)
TeleMetrum v2.0 Altimeter	
GPS horiz position	±2.5m
LORD MicroStrain IMU	
Accelerometer	$100\mu g/\sqrt{Hz}$
Gyroscope bias	8 dph in-run
Gyroscope ARW	$0.3°/\sqrt{hr}$
Pitch-roll dynamic accuracy	±0.4°
PX142-030A5V Pressure Transducer	
Output	1 to 6 Vdc (±2.5 Vdc PX143)
Linearity	±0.75% FS BFSL
Hysteresis	0.15% FS (0.30% FS ≤ 5 psi)
Zero Balance	1.0 Vdc ±0.05

IV. Development of the Kalman Filter

The method of formulating a Kalman filter is fairly well known, so only the key equations for this system and filter will be presented here. Reference [6] provides a good introduction to the Kalman filter. Further background into the derivation of the Kalman filter and its various forms can be found in [7].

A. Coordinate Definitions

This system will make use of two main coordinate systems and a set of Euler angles as defined in this section. The body-fixed coordinate system is defined as show in Figure 6. This coordinate system will align with the axial directions of the IMU.

[e]https://www.omega.com/pptst/PX140.html

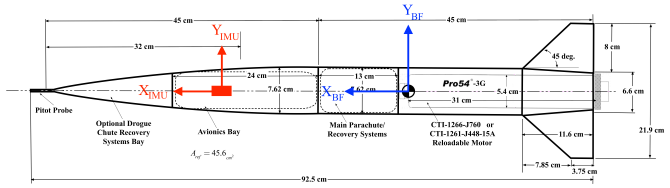

Figure 6. Illustration showing the orientation of the body-fixed coordinate system (axis on the right) and the IMU (axis on the left).

The second coordinate system that will be used is Local North-East-Down (NED). This coordinate system is centered in the same location as the body-fixed frame, but is always aligned with true-north, east, and down (towards the center of the earth). The relationship of the body-fixed frame to the NED frame can be seen in Figure 7. Also show in this figure are the definitions of the roll, pitch , and yaw Euler angles. All rotations and angular rates are defined as positive in the "clockwise" or "right-handed" direction.

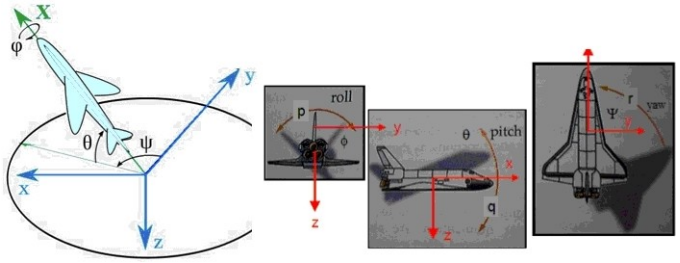

Figure 7. Illustration of the north-east-down (x-y-z) frame and the roll, pitch, yaw Euler angles. Notice that while the NED frame is not centered on the craft in this depiction, the NED frame used for this system is centered in the same location as the body-fixed frame.

The NED and body-fixed coordinate frames can be related to each other through use of the Euler angles defined previously. These angles can be used to form the following rotation matrix.

$$M = \begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \\ M_{31} & M_{32} & M_{33} \end{bmatrix} \tag{1}$$

Where

$$\begin{aligned} M_{11} &= \cos\phi\cos\theta \\ M_{12} &= \sin\psi\cos\theta \\ M_{13} &= -\sin\theta \\ M_{21} &= \cos\psi\sin theta\sin\phi - \sin\phi\cos phi \\ M_{22} &= \sin\psi\sin\theta\sin\phi + \cos\psi\cos\phi \\ M_{23} &= \cos\theta\sin\phi \\ M_{31} &= \cos\psi\sin\theta\cos phi + \sin\psi\cos\phi \\ M_{32} &= \sin\psi\sin\theta\cos\phi - \cos\psi\sin\phi \\ M_{33} &= \cos\theta\cos\phi \end{aligned}$$

This rotation matrix would be used as follows

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \\ M_{31} & M_{32} & M_{33} \end{bmatrix} \begin{bmatrix} v_n \\ v_e \\ v_d \end{bmatrix} \tag{2}$$

where u, v, and w are the velocity components of the rocket in the body-fixed frame. To go from body-fixed to NED coordinates, one would simply use the transpose of the rotation matrix M. To simplify notation, the rotation matrix M and its components will be referred to from here on out using matrix element notation (M_{ij}).

B. Equations of Motion

This section will present the system's equations of motion which will be used in the Kalman filter. First, the equations for the velocity and acceleration of the rocket will be derived and presented. These equations will be numerically integrated to keep track of the vehicle's state. Two possible models for gravity will then be shown and the differences between these models will be discussed. The collected state equations for the system will then be presented.

1. *Acceleration and Velocity*

A vector equation for the acceleration of the rocket can be found as follows using Newton's second law.

$$\frac{\sum \vec{F}}{m} = \frac{d\vec{v}}{dt} = \frac{\partial\vec{v}}{\partial t} + \vec{\omega}\times\vec{v} \tag{3}$$

The left side of equation 3 can be found using the linear acceleration values output by the IMU plus the effects of gravity.

$$\frac{\sum \vec{F}}{m} = g_0\vec{A}_{IMU} + gM\begin{bmatrix}0\\0\\1\end{bmatrix} = g_0\begin{bmatrix}A_x\\A_y\\A_z\end{bmatrix} + g\begin{bmatrix}M_{13}\\M_{23}\\M_{33}\end{bmatrix} \tag{4}$$

The rotational acceleration term on the right side of equation 3 comes out to be

$$\vec{\omega}\times\vec{v} = \begin{bmatrix}p\\q\\r\end{bmatrix}\times\begin{bmatrix}u\\v\\w\end{bmatrix} = \begin{bmatrix}0 & -r & q\\ r & 0 & -p\\ -q & p & 0\end{bmatrix}\begin{bmatrix}u\\v\\w\end{bmatrix} \tag{5}$$

Substituting equations 4 and 5 into equation 3 and solving for $\frac{\partial\vec{v}}{\partial t}$ results in the first differential equation of motion for this system.

$$\frac{\partial\vec{v}}{\partial t} = \begin{bmatrix}\dot{u}\\\dot{v}\\\dot{w}\end{bmatrix} = \begin{bmatrix}0 & r & -q\\ -r & 0 & p\\ q & -p & 0\end{bmatrix}\begin{bmatrix}u\\v\\w\end{bmatrix} + g_0\begin{bmatrix}A_x\\A_y\\A_z\end{bmatrix}_{cg} + g\begin{bmatrix}M_{13}\\M_{23}\\M_{33}\end{bmatrix} \tag{6}$$

The second equation of motion for this system simply relates the body-fixed velocity of the rocket to the NED velocity of the rocket. This equation comes almost directly from equation 2.

$$\frac{\partial\vec{x}}{\partial t} = \begin{bmatrix}\dot{x}_n\\\dot{y}_e\\\dot{z}_d\end{bmatrix} = \begin{bmatrix}M_{11} & M_{21} & M_{31}\\ M_{12} & M_{22} & M_{32}\\ M_{13} & M_{23} & M_{33}\end{bmatrix}\begin{bmatrix}u\\v\\w\end{bmatrix} \tag{7}$$

2. *Gravity Models*

The first gravity model presented includes the first harmonic of the earth's gravity field and provides a correction for the centrifugal force due to the earth's rotation.

$$g = \frac{\mu}{(R_\oplus + h_{MSL})^2} - \Omega_\oplus^2(R_\oplus + h_{MSL}) \tag{8}$$

Where

$$\begin{aligned} \mu &= 3.9860044\times10^5 km^3/s^2 \\ \Omega_\oplus &= 7.2929115\times10^-5 rad/s \\ R_\oplus &= \frac{R_{eq}}{\sqrt{1+\frac{e^2}{1-e^2}\sin^2\lambda}} \\ R_{eq} &= 6378.13649km \\ e &= 0.08181980 \end{aligned}$$

Another reasonably accurate and less complex model suitable for real-time calculations is

$$\begin{aligned} g_{alt} =& (9.780318m/s^2)[1 + 0.0053024\sin^2(\lambda) \\ &+ 0.0000058\sin^2(2\lambda)] - 3.086\times10^-6 s^{-2}\cdot h_{MSL} \end{aligned} \tag{9}$$

In both of these models, λ is the geodetic latitude of the rocket. The North Branch, Minnesota launch site[f] is located at 45.547 deg latitude and 270 m. At this location, these two models give the following results:

$$g = 9.7972$$
$$g_{alt} = 9.8060$$

These results differ by approximately 0.0898%. Considering the altitude range and duration of our rocket's flight, both of these models are acceptable for this application.

3. *Accelerometer Corrections for Center of Gravity Offset*

Because the rocket's avionics bay will not allow the IMU to be placed precisely at the vehicle's center of gravity, it is necessary to correct the sensed accelerations for centrifugal forces resulting from angular rotations. These corrections are given by the following transformation.

$$\begin{bmatrix} A_x \\ A_y \\ A_z \end{bmatrix}_{cg} = \begin{bmatrix} A_x \\ A_y \\ A_z \end{bmatrix}_{IMU} + \frac{1}{g_0} \begin{bmatrix} (r^2+q^2)X_{IMU} - (pq-\dot{r})Y_{IMU} - (rp+\dot{q})Z_{IMU} \\ -(pq+\dot{r})X_{IMU} + (p^2-r^2)Y_{IMU} - (rq+\dot{p})Z_{IMU} \\ -(rp-\dot{q})X_{IMU} - (rq-\dot{p})Y_{IMU} + (q^2+p^2)Z_{IMU} \end{bmatrix} \quad (10)$$

Since the angular acceleration rates $\dot{p}$, $\dot{q}$, and $\dot{r}$ are not available in real time, only the corrections due to the angular rates will be performed. The cg-corrected acceleration values will be used in the Kalman filter calculations. For a rigid airframe, the angular rates at the IMU will be the same as those at the center of gravity. The X, Y, Z position of the IMU is measured along the body axes of the vehicle as defined in Fig. 6 from the vehicle's center of gravity.

4. *Collected State Equations*

Referring back to the equations of motion presented earlier, there are six state variables that will be directly estimated by the Kalman filter. The collected velocity and position equations are

$$\frac{\partial \vec{x}}{\partial t} = \begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \\ \dot{x}_n \\ \dot{y}_e \\ \dot{z}_d \end{bmatrix} = \begin{bmatrix} 0 & r & -q & 0 & 0 & 0 \\ -r & 0 & p & 0 & 0 & 0 \\ q & -p & 0 & 0 & 0 & 0 \\ M_{11} & M_{21} & M_{31} & -1/\tau_x & 0 & 0 \\ M_{12} & M_{22} & M_{32} & 0 & -1/\tau_y & 0 \\ M_{13} & M_{23} & M_{33} & 0 & 0 & -1/\tau_z \end{bmatrix} \begin{bmatrix} u \\ v \\ w \\ x_n \\ y_e \\ z_d \end{bmatrix} + g_0 \begin{bmatrix} A_x \\ A_y \\ A_z \\ 0 \\ 0 \\ 0 \end{bmatrix}_{cg} + g \begin{bmatrix} M_{13} \\ M_{23} \\ M_{33} \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad (11)$$

which match the general state-space form

$$\dot{\vec{x}} = A\vec{x} + B\vec{u} \quad (12)$$

The inverse time-constant ($-1/\tau$) terms which can be seen in equation 11 are added to the collected state equations to provide low-pass filtering of the GPS-based position data. This will help compensate for noisy data from the GPS receiver. The acceleration and rotation-rate data are filtered by the IMU and therefore do not need to be filtered by the real-time state estimation algorithm.

[f]The main flight tests will be performed during the Midwest High Power Rocket Competition in North Branch, Minnesota. Therefore, the comparison of the gravity models is conducted assuming this location.

5. *Measurement Equation*

In addition to the collected state equations, a model for the measurements of the system is needed to form the Kalman filter. As mentioned previously, this system will make use of altitude, position, and airspeed measurements. The measurement model for this system is given by the matrix equation

$$\begin{bmatrix} h_{altimeter} \\ V_{\text{inf}}^2 \\ x_{n,GPS} \\ y_{e,GPS} \\ z_{d,GPS} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & -1 \\ u & v & w & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ w \\ x_n \\ y_e \\ z_d \end{bmatrix} \quad (13)$$

which is equivalent to

$$\vec{y} = H\vec{x} + D\vec{u} \quad (14)$$

For this system, the D matrix is a zero matrix because none of the control values appear in the measurement equations.

C. Kalman Filter Equations

An Extended-Discrete Kalman filter will be used as the core of the state estimation algorithm. The filter will propagate the state estimate by integrating equation 11. The state transition matrix will be used to propagate the state covariance. The state estimate and covariance will then be updated using the discrete Kalman filter update equations.

1. *State Propagation Equations*

Because accelerometer and gyroscope information is needed to propagate the state equations, the state estimate (estimates are denoted by the hat notation) will be propagated using the trapezoidal rule.

$$\hat{\vec{x}}_{k+1}^- = \hat{\vec{x}} + (\dot{\hat{\vec{x}}}_{k+1} + \dot{\hat{\vec{x}}}_k)\Delta t/2 \quad (15)$$

The state covariance will be propagated using the standard formulation for a discretized Kalman filter.

$$\hat{P}_{k+1}^- = \phi_k P_k \phi_k^T + Q_k \quad (16)$$

The approximate state transition matrix for the system can be found using the matrix exponential. Only the first two terms of the expansion will be used. To simplify the work, the A matrix can first be subdivided into the following matrices.

$$A = \begin{bmatrix} [\vec{\omega}\times] & 0_{3\times3} \\ M^T & [\tau] \end{bmatrix} \quad (17)$$

The state transition matrix can then be approximated by

$$\phi = e^{A\Delta t} \approx I + A\Delta t + A^2\Delta t^2/2 = \begin{bmatrix} I_{3\times3} + [\vec{\omega}\times]\Delta t + [\vec{\omega}\times][\vec{\omega}\times]\Delta t^2/2 & 0_{3\times3} \\ M^T\Delta t + (M^T[\vec{\omega}\times] + [\tau]M^T)\Delta t^2/2 & I_{3\times3} + [\tau] + [\tau][\tau] \end{bmatrix} \quad (18)$$

2. *State Update Equations*

The state estimate and state covariance are both updated when measurement information is received from a sensor. Every update requires the Kalman gain to be calculated using the following equation.

$$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + R_k)^{-1} \quad (19)$$

Because each measurement will be processed individually, the Kalman gain will be calculated separately for each measurement. Once the Kalman gain is calculated for a particular measurement, the state estimate is updated as follows.

$$\hat{\vec{x}}_k = \hat{\vec{x}}_k^- + K_k(z_k - H_k\hat{\vec{x}}_k^-) \quad (20)$$

The state covariance matrix is updated for each measurement using the standard covariance update equation. The Joseph form is not used here to save on computation time and resources.

$$P_k = (I - K_k H_k) P_k^- \tag{21}$$

To streamline the computation performance of the real-time algorithm, each of these matrix equations will be expanded and calculated as a series of equations for each vector or matrix element. Each measurement will also be processed separately by the Kalman filter as measurement information is received from the sensors.

V. Test Case Results

The Kalman filter algorithm was tested in MATLAB using a two-dimensional test case. White noise was added to the accelerometer, gyroscope, altitude, and GPS measurements to match the instrument specifications given in Table 1. The algorithm was first run without any measurements to test the performance of the state propagation. Two more tests were then run simulating altitude and then GPS position data.

A. No Measurement Data

Running the algorithm without any altitude or GPS measurements gave the following results as shown in Figures 8 and 9. Both the estimate of the rocket's altitude stayed within ±2m of the actual value, and the estimate of the rocket's velocity stayed within ±0.25m/s of the actual velocity. However, notice how the position estimate appears to be drifting away from the expected value as time increases. Some drift is expected because this estimate is only based on integration of the accelerometer and gyroscope data.

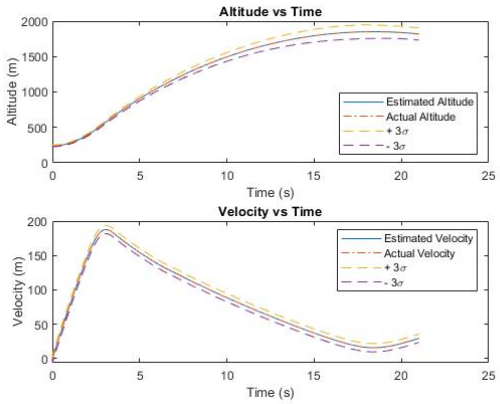

Figure 8. Plot of the rocket's estimated altitude and velocity without any altitude or GPS measurements.

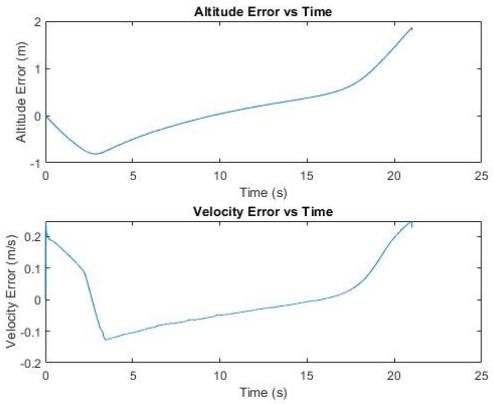

Figure 9. Plot of the error in the rocket's estimated altitude and velocity without any altitude or GPS measurements.

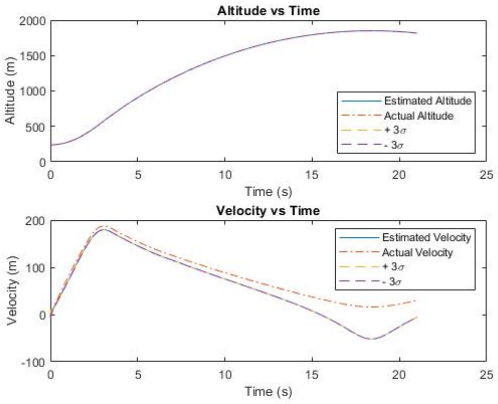

Figure 10. Plot of the rocket's estimated altitude and velocity using an altitude measurement.

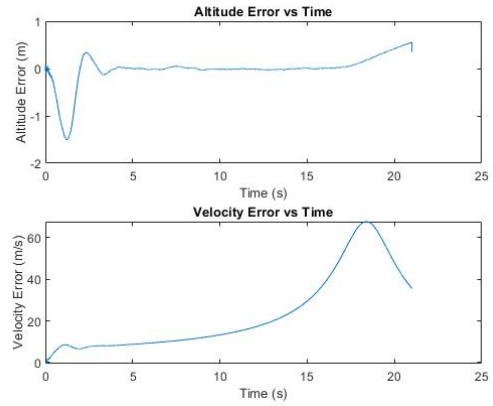

Figure 11. Plot of the error in the rocket's estimated altitude and velocity using an altitude measurement.

B. Altitude Measurement Only

Incorporating an altitude measurement to update the filter's state estimate gave the results shown in Figures 10 through 11. As expected, adding a measurement of the rocket's altitude helped to decrease the error in the estimate's altitude. The largest error in the estimated altitude occurs around motor burnout when the rocket is traveling fastest. However, notice how the error in the rocket's estimated velocity increased after including this measurement.

C. GPS Measurement Only

Testing a simulated GPS height measurement to update the filter's state estimate gave results similar to the last case. this is not surprising, since this gives almost identical information to the altitude measurement. Figures 10 through 11 show the estimates and errors for this test. The largest error in the estimated altitude once again occurs around

motor burnout. After that, the filter successfully decreases the error in the altitude measurement. Including this GPS measurement did not improve the estimate of the velocity. No x-position data was available for this test case, so future tests of the filter will include more dimensions for testing.

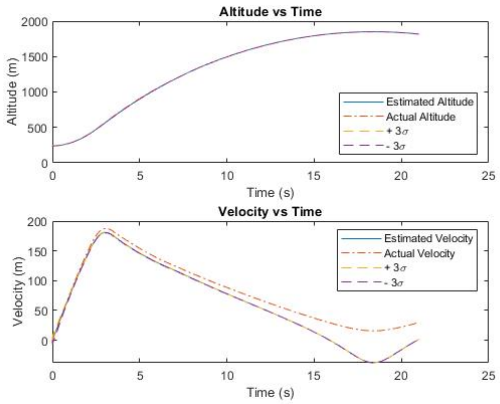

Figure 12. Plot of the rocket's estimated altitude and velocity using a GPS position measurement.

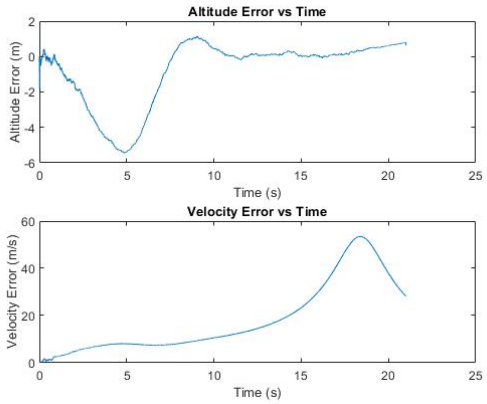

Figure 13. Plot of the error in the rocket's estimated altitude and velocity using a GPS position measurement.

VI. Flight Testing

The flight trajectory of the planned rocket flight tests was simulated using custom code. This simulation uses a fourth-order Runge-Kutta integrator to numerically integrate the 2D ballistic equations of motion. The launch altitude and other location-dependent conditions were set to match those of North Branch, Minnesota. Atmospheric drag effects were modeled using the 1976 US Standard Atmosphere and models for skin friction, pressure, wake, fin dither, and fin leading edge drag. The rocket drag coefficient calculated from these equations was adjusted for compressibility effects using the modified Karman rule. Table 2 summarizes the magnitudes of key parameters associated with the launch and recovery elements of the trajectory.

A. Flight Analysis

1. *Launch to Apogee*

Figure 14 shows the predicted altitude of the rocket over time from launch to apogee. The simulation predicts that the rocket will reach apogee 20.73 seconds after launch. The peak altitude of the test flights is estimated to be 2538.4 m Above Ground Level (AGL) (8328.1 ft).

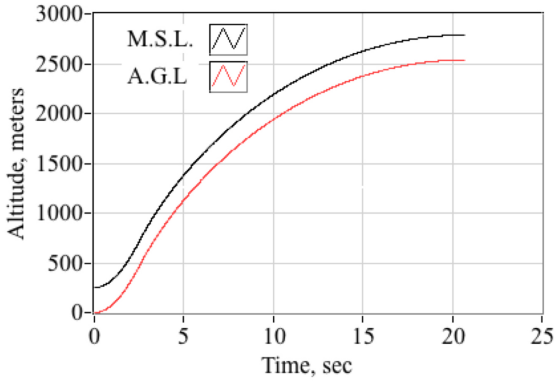

Figure 14. Plot of the rocket's expected altitude from launch to apogee.

Figure 15 shows the estimated velocity of our rocket the time, with the velocity peaking at about 2.5 seconds after launch. The simulation estimates the peak velocity to be 340.757 m/s (1117.970 ft/s - Mach 1.01).

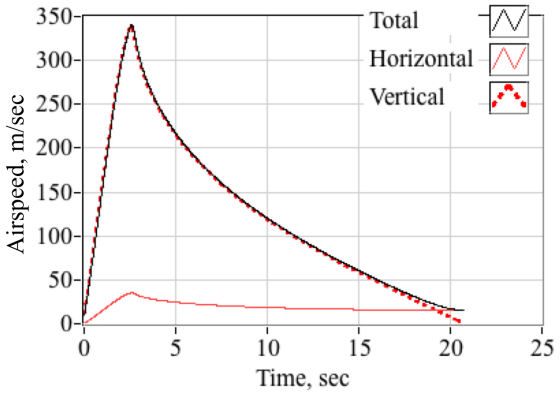

Figure 15. Plot of the rocket's expected airspeed from launch to apogee.

As expected, the simulation predicts that the rocket's peak acceleration will occur shortly after launch. Figure 16 shows a plot of the predicted acceleration loads the rocket will experience. Both simulation cases predict that the rocket's peak acceleration will be 17.45 g. After reaching a maximum, the acceleration loads will then drop off and reverse. This load reversal corresponds with motor burnout. Thus, it can be seen that the rocket will experience the majority of its acceleration loading during the first five seconds of flight.

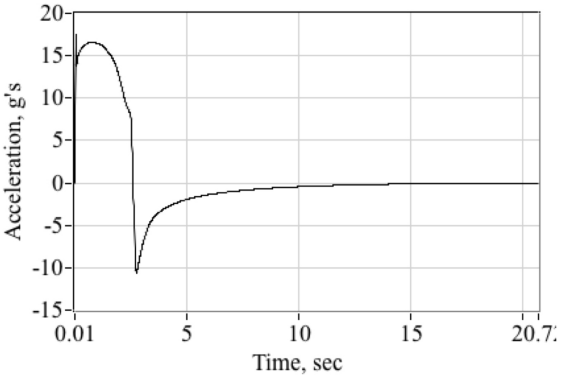

Figure 16. Plot of the rocket's expected acceleration from launch to apogee.

The thrust and drag forces acting on the rocket are predicted to result in the loads shown in Figure 17. The peak thrust load predicted by both simulations is just above 505 N (113.5 lbf). Both simulations then

predict a peak drag load of around 300 N (67.4 lbf). This peak drag load once again corresponds with the transonic drag increase. Figure 18 illustrates this correlation, showing a comparison of the rocket's predicted Mach number and drag coefficient over time. The drag coefficient clearly peaks as the rocket reaches Mach 1.

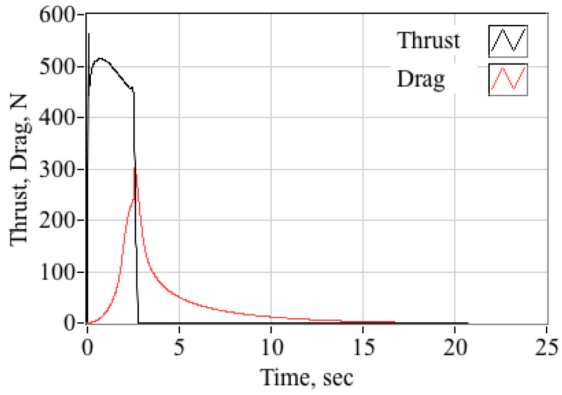

Figure 17. Plot of the expected thrust and drag loads that the rocket will experience from launch to apogee.

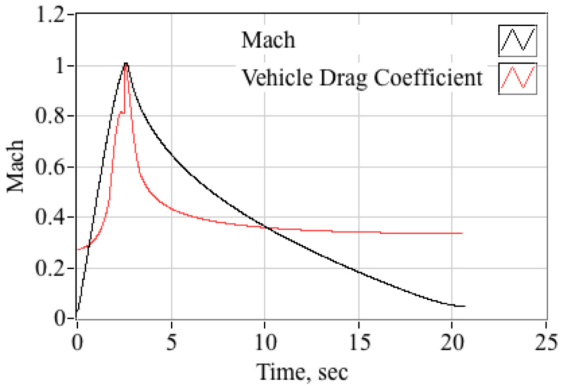

Figure 18. Plot of mach number and the rocket's expected drag coefficient from launch to apogee.

2. Apogee to Landing

For the second part of flight from apogee to landing, the simulation included a drogue and a main parachute. Both chutes were simulated with an inflation exponent of 0.86 and a fill time shape exponent of 2. Both chutes also were simulated using an opening shock load factor of 1.8. The drogue deploys at an altitude of 2500 m (8202.1 ft) AGL. After drogue deployment, the rocket is predicted to have a drogue descent speed of about 15 m/s (49.2 ft/s).

The main chute deploys afterwards at an altitude of 300 m (984.3 ft) AGL. With both the main and drogue chutes deployed, the rocket is predicted to have a final vertical descent and landing speed of about 4.6 m/sec (15.1 ft/sec). Figure 19 shows the predicted horizontal, vertical, and total airspeed of our rocket from apogee to landing.

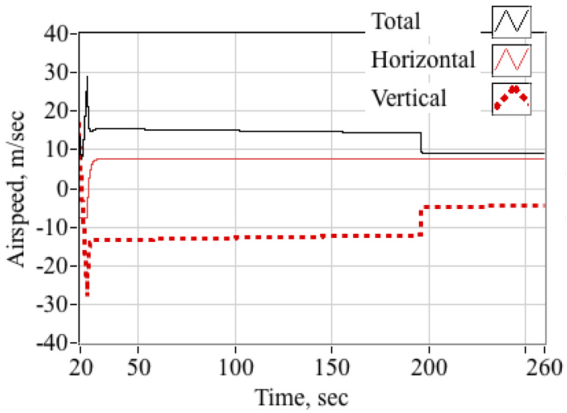

Figure 19. Plot of the rocket's expected airspeed from apogee to landing.

The acceleration load produced by the drogue and main chutes at deployment, shown in Figure 20, are predicted to be 7.32 g's and 2.02 g's respectively. These acceleration loads correspond to deployment loads for each parachute between 210 N (47.2 lbf) and 225 N (50.6 lbf) shown in Figure 21.

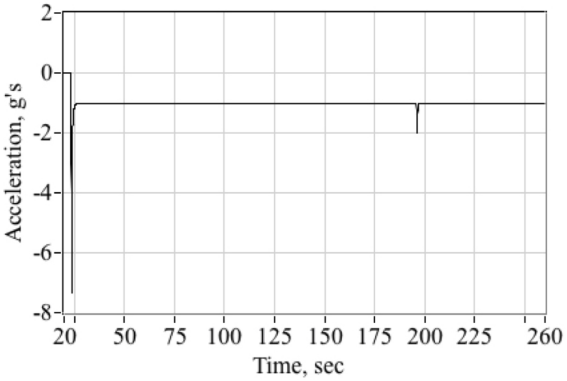

Figure 20. Plot of the rocket's expected acceleration from apogee to landing. The peaks correspond to the deployment of the drogue and main chutes.

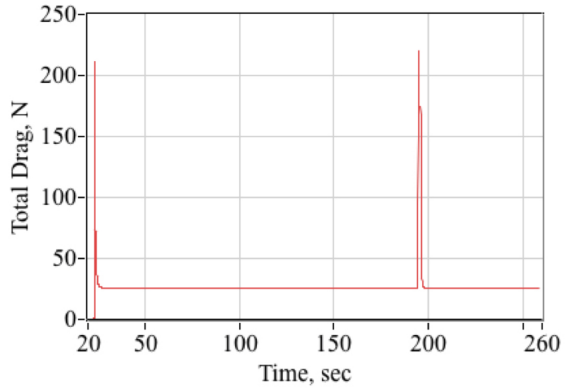

Figure 21. Plot of the rocket's expected drag loading (right) from apogee to landing. The peaks correspond to the deployment of the drogue and main chutes.

With a 15 knot wind, the simulation predicts that the rocket will drift approximately 2464 m (8084 ft) down range of the launch site. Figure 22 shows the down range position of the rocket with respect to time. The rocket is expected to land 260 seconds after launch.

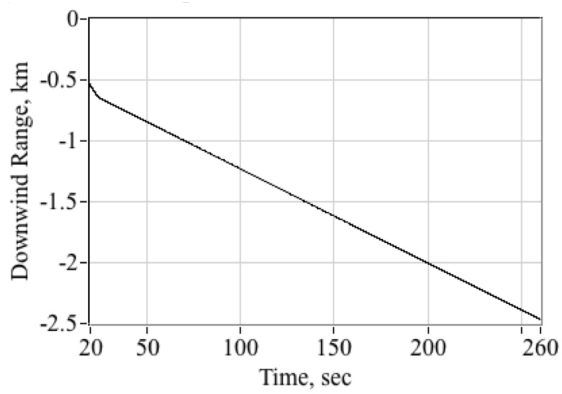

Figure 22. Plot of the rocket's expected down range ground position from apogee to landing with a wind of 15 knots.

Table 2. Summary of key trajectory events.

Launch to Apogee	
Peak Altitude	2539.34 m AGL
Peak Velocity	340.869 m/s
Peak Acceleration	17.45 g's
Peak Mach Number	1.01
Time to Apogee	20.73 s

Landing Conditions	
Wind Speed	15 Kts (7.72 m/s)
Horizontal Velocity	7.72 m/s
Vertical Velocity	4.592 m/sec (15.1 ft/s)
Drogue Opening Load	-7.32 g's
Main Opening Load	-2.02 g's
Time to Landing (from Launch)	260 s
Downwind Drift Range	2464 m

VII. Conclusion

The presented Kalman filter algorithm works well with test case data and is performing as expected. However, additional fine-tuning is needed so that the filter will provide a more accurate estimate of the rocket's velocity using altitude and GPS measurements. Adding additional GPS data in the x and y directions should help to improve this estimate.

However, despite how well the filter does or does not perform using test data, the true test will be to see how well it performs in flight. Test flights this summer will help to determine how the Zero W performs under high acceleration loads. The most challenging portion of flight for the filter will be the first five seconds as the rocket accelerates to Mach 1 and then decelerates.

Acknowledgments

I would like to acknowledge Dr. Stephen Whitmore for providing support and mentorship throughout this project as my major professor. I would also like to acknowledge Dr. David Geller for his help with portions of the Kalman filter algorithm. Finally, I would like to thank the Utah NASA Space Grant Consortium for funding this project.

References

[1] Jensen, A. M., Hardy, T., McKee, M., and Chen, Y., "Using a multispectral autonomous unmanned aerial remote sensing platform (AggieAir) for riparian and wetlands applications," *2011 IEEE International Geoscience and Remote Sensing Symposium*, 2011, pp. 3413–3416. .

[2] Zaman, B., Jensen, A. M., and McKee, M., "Use of high-resolution multispectral imagery acquired with an autonomous unmanned aerial vehicle to quantify the spread of an invasive wetlands species," *2011 IEEE International Geoscience and Remote Sensing Symposium*, 2011, pp. 803–806. .

[3] Kendall, A. G., Salvapantula, N. N., and Stol, K. A., "On-board object tracking control of a quadcopter with monocular vision," *2014 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2014, pp. 404–411. .

[4] Santana, L. V., Brandão, A. S., Sarcinelli-Filho, M., and Carelli, R., "A trajectory tracking and 3D positioning controller for the AR.Drone quadrotor," *2014 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2014, pp. 756–767. .

[5] Ackerman, E., and Strickland, E., "Medical delivery drones take flight in east africa," *IEEE Spectrum*, Vol. 55, No. 1, 2018, pp. 34–35. .

[6] Welch, G., and Bishop, G., "An Introduction to the Kalman Filter," *Aviation Week & Space Technology*, Vol. 145, No. 24, 1997, pp. 44–46.

[7] Crassidis, J. L., and Junkins, J. L., *Optimal Estimation of Dynamic Systems*, Chapman & Hall/CRC Applied Mathematics and Nonlinear Science Series, CRC Press, New York, 2011, pp. 144–148.