Proceedings of the Sustainable Research and Innovation Conference
JKUAT Main Campus, Kenya
6 - 7 October, 2021

Detection of Apogee with Kalman Filter for Flight Computer of Model Rockets

Michael G. Kimani, David R. Osodo and Shohei Aoki

***Abstract*—**Raw sensor values are often noisy and unreliable in high sampling frequencies. Accuracy is important when it comes to guidance, navigation, and control of vehicles more so spacecraft or ships. We made use of the Kalman filter to get stable values throughout our flight. The purpose of this study is to come up with a proper filtering method that accurately estimates the distance, velocity, and acceleration from the sensor readings, BMP180 altitude readings, and MPU6050 Z-axis acceleration readings. We come up with a physical model that describes our system dynamics to get the most out of our filter. With this data, we made three algorithms to accurately predict our apogee for parachute ejection. The first one checks if the altitude readings start decreasing after launch by 0.1 meters. The second one checks if the velocity value is within an envelope of -1 m/s and +1 m/s after launch while the third one checks if the acceleration value is within an envelope of -1 m/s^2 and +1 m/s^2 also after a successful launch was detected. With this system, we were able to accurately predict apogee since they worked in tandem.

***Keywords*—**Kalman filter, sensor noise, state estimation.

I. Introduction

Apogee detection is important in model rockets because it is the best time to deploy a parachute for a recovery mechanism. Timely deployment ensures that the parachute will not tear mid-flight due to high acceleration hence high downward force. This way we will be able to keep track of the rocket during descent. Late or early parachute deployment will result in forces acting against the parachute in flight. While it is possible to create a parachute that can withstand these forces, it is ideal to minimize the forces first. We are also constrained by the budget and resource requirements to make a bigger and stronger parachute. Without optimum parachute deployment, the recovery stage will fail. Exploring rocket apogee detection methods, and in particular, the use of the Kalman filter in rocket apogee detection helps optimize performance while minimizing costs and disadvantages.

The current methods used to eject parachutes after detecting apogee are mostly mechanical and aren't electronic. The most common is to make a secondary ignition to push the components of the rocket out towards the nose cone called a pyrotechnic mechanism. An alternative method is to use a timer that triggers to open the nose cone or where the parachute is housed after the specified duration has elapsed. To calculate time to apogee then set the timer to open up the nose cone or where the parachute is housed after the specified duration has elapsed. This method relies on precomputed estimates of apogee from OpenRocket and OpenMotor software. There is a large source of error introduced by the lack of precision of apogee prediction. Without relying on real-time data this method can be unreliable. The accurate method is, therefore, employing real-time data, gyro, accelerometer, and altimeter readings to deploy the parachute [1].

For our case, we planned on using a barometer, BMP180 module, and an accelerometer with a gyroscope, MPU6050 module as our primary sensors to collect data during flight. The microcontroller will constantly check on the real-time data to detect the launch and also apogee. The acceleration of the rocket steadily increases due to the thrust, reaches a peak, and drops down to a constant -9.8 m/s^2 after burning out (since the only force working on the rocket at that point is the downward force of gravity) [2]. Therefore, in a perfect world where the acceleration and the accelerometer match, the accelerometer would integrate in real-time to and the velocity, and as soon as the velocity reached zero, it would deploy the parachute. This poses a challenge since the data from the barometer and accelerometer is very noisy.

Another approach to electronic apogee detection comes from using a barometer. The barometer measures air pressure and detects changes in air pressure to determine the height of the rocket. Apogee is detected when the air pressure stops decreasing and begins to increase. Of course, without digital filtering, apogee is impossible to detect; there is too much noise and quantization error [3]. In addition, because barometers measure air pressure, the pressure wave produced when the rocket nears the speed of sound may be registered as a drop-in altitude and therefore apogee. Barometers, like accelerometers, are easily fooled. The noise caused in the data alone could cause the parachute to deploy anytime the data dipped below a past value, even if the model rocket is in the thrust phase.

To overcome the noisy data problem, we implemented a Kalman filter. Using sensor fusion allows getting the best out of both the accelerometer sensor and the barometer [4]. By predicting a joint probability distribution over the variables in each time, Kalman filtering, also known as linear quadratic prediction, uses a series of measurements taken over time that contains statistical noise and other inaccuracies to produce

Michael G. Kimani, Department of Mechatronic Engineering, JKUAT
David R. Osodo, Department of Mechatronic Engineering, JKUAT
Shohei Aoki, Department of Mechatronic Engineering, JKUAT
(+2540742101078; e-mail: aoki@jkuat.ac.ke).

guesstimates of unknown variables that are more precise than those based on a single measurement alone [5]. Generally, our purpose is to find the estimate of the state x at the given time k, based on the current and previous sensor readings, and we wish to find it for each consequent time. The Kalman filter finds the most optimum averaging factor for each consequent state [6].

II. KALMAN FILTER

The following section describes how the Kalman filter was implemented to acquire useful sensor values from our accelerometer and barometer sensors.

We use extra input to estimate the state at time k called the a priori state. The state variables represent attributes of the system that we wish to know something about.

$$\hat{x}_k^- = A\,\hat{x}_{k-1} \quad (1)$$

$$P_k^- = AP_{k-1}A^T + Q \quad (2)$$

In (2), the error covariance matrix P_k^- represents how much we trust our current values of the estimated state. The smaller the value the more we trust it. Error covariance will increase since we last updated the estimate of the state, therefore we multiplied the error covariance matrix by the system model A and its transpose A^T and current process noise Q.

Matrix Q represents process noise. It decides accuracy and time lag in the estimated value. Higher Q results in a higher gain and hence more weight to the noisy measurements. The estimation accuracy is thereby compromised. Lower Q leads to a better estimation accuracy but a time lag may be introduced to the estimated value. Q was therefore used as a tuning factor.

Matrix R represents measurement noise. It represents sensor noise characteristics. R is calculated from the sensor accuracy which is represented using a standard deviation of measured values from true values.

We try to predict how much we trust the measurement based on a priori error covariance and measurement covariance matrix. H maps the a priori error covariance matrix into an observation matrix. Bigger measurement noise, a bigger value of S_k hence not trusting the measurement.

$$S_k = H\,P_k^-H^T + R \quad (3)$$

$$K_k = P_k^-H^TS_k^{-1} \quad (4)$$

$$\hat{x}_k = \hat{x}_k^- + K_k\,(z_k - H\hat{x}_k^-) \quad (5)$$

$$P_k = (I - K_k\,H)\,P_k^- \quad (6)$$

We use H to extract data from state error covariance and compare it with the current estimation. When the state x_k is multiplied by H, it is converted to measurement z_k. Since our measurements map directly to the state, we just multiply them all by 1 to convert our states to measurements. We initialized P as a unit matrix. R is our estimate of the measurement error covariance.

Checking deviation in the rocket's altitude was used to detect lift-off. Lift-off was detected once altitude readings increased by 0.08 m or higher for more than five consecutive readings. The method to detect the apogee is described in the following section.

III. APOGEE DETECTION

Three methods were used to detect apogee:

1. Altitude
2. Velocity
3. Acceleration

1) Altitude

This is an easier method to implement but has a large chance of failure. The same logic was applied to lift-off as mentioned previously. Checking deviation in the rocket's altitude was used to detect apogee. Apogee was detected once altitude readings increased by 0.03 m or higher for more than three consecutive readings.

2) Velocity

This is the most promising method since from our simulations, it does work. First, we check if we have detected a launch. Once a launch was detected, the velocity values were checked for when they were within an envelope of 1 and -1 which meant that the rocket was at apogee. This works well if the Kalman filter works well thus producing a velocity graph identical to the actual velocity graph.

Acceleration

This is our second most promising method since it worked from our simulations. First, we check if we have detected a launch. Two seconds after a launch was detected, acceleration values were checked until they were within an envelope of 1 and -1 which meant that the rocket was at apogee. From the physical simulation, we see that at apogee the acceleration comes back from negative values and starts oscillating around 9 m/s. We check this by a range value. The next time the acceleration comes to the range from 1 to -1 then we will be at apogee.

IV. RESULTS

Altitude

The application of the Kalman filter was successful in acquiring altitude values.

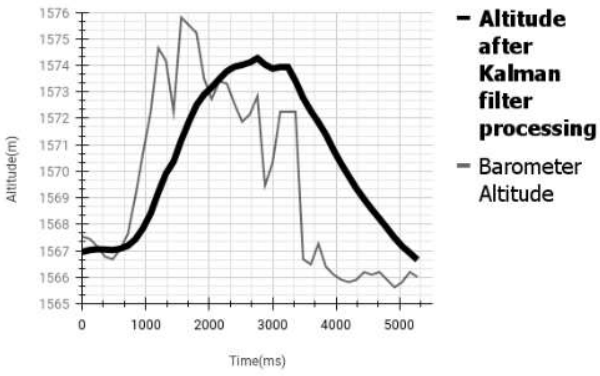

Fig. 1 Raw data and data processed by Kalman filter

In Fig. 1, the raw and filtered altitude values show the level of success of our Kalman filter. The raw data is acquired by reading the pressure value from the barometer sensor and these

pressure values are converted to altitude values.

TABLE I
ALTITUDE AND VELOCITY

Altitude (m)	Velocity (m/s)
1573.94	2.67
1573.94	2.58
1573.41	2.18
1572.77	1.68
1572.28	1.35
1571.85	1.06
1571.38	0.76
1570.8	0.34
1570.28	0.02
1569.81	-0.26

As shown in Table I, the rocket altitude readings stagnated at 1573.94 m which was our apogee. The apogee was detected within four readings from this point which was about 160 ms from the actual apogee.

Velocity

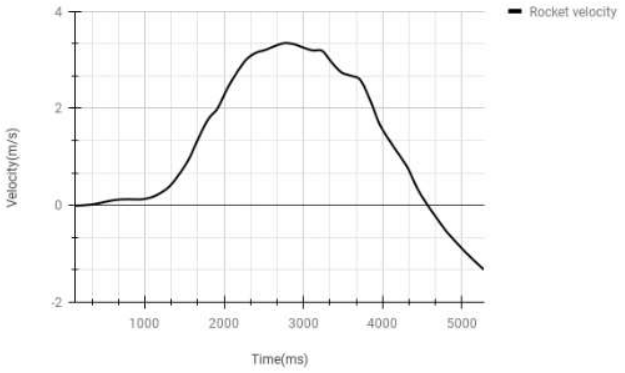

Fig. 2 Rocket velocity with time

In Fig. 2, the rocket velocity increases from zero and then begins to fall.

TABLE II
VELOCITY AND TIME

Velocity (m/s)	Time (ms)
3.18	920
2.94	960
2.74	1000
2.67	1040
2.58	1080
2.18	1120
1.68	1160
1.35	1200
1.06	1240
0.76	1280
0.34	1320
0.02	1360
-0.26	1400

From the rocket's highest point which occurs at 1080 ms in Table II, its velocity reduced and eventually fell to a negative value. Using this method apogee was detected within 7 readings which were about 280 ms from the rocket's highest point.

Acceleration

The rocket's acceleration values never went past 1 which meant that this method was unsuccessful.

V. Discussion

We generally assume our system's behavior is according to the trajectory model, where the velocity is indeed the position derivative, and the acceleration is indeed the velocity derivative. However, in real-life scenarios, we deal with non-Gaussian noise, discretization error, numerical considerations, measurement noise, nonlinearity, and many more [7] [8]. Hence, the trajectory model for using the Kalman filter must be established by considering the physical relations. Nevertheless, adding some white noise enables us to handle real-life scenarios. This noise is injected into the model, more specifically, into the Kalman filter to compensate for the uncertainty of the model. For example, we may assume the constant velocity model where in real life the velocity changes rapidly.

An appropriate Q matrix should be defined to handle this scenario. It is important to mention that choosing high values for the Q matrix might result in filter divergence, as there is huge uncertainty for his modeling. So, there is an optimal matrix Q for each scenario- which is not trivial to find. One way of finding Q is trial and error. Starting by large value and reducing it until convergence might work, however the overfitting must be avoided. Another approach is the adaptive method. The most common adaptive approach to tune Q in real-time is to consider the innovation quantity of the Kalman filter.

We optimized the parameters offline with a Grid Search algorithm to find the minimum difference between the real data and the noisy data. We also trained our Kalman filter on displacement data and acceleration so that we can be able to obtain the velocity values which should be similar to the initial velocity values. This is a supervised-based approach as we thought we would learn the data well and reduce computation on the actual flight computer if we were to run the optimization in flight.

VI. Conclusion

The apogee detection was successful when using altitude and velocity methods, whereas the method using rocket acceleration proved to be unsuccessful. The Kalman filter was instrumental in reducing noise in the data which we achieved. In future work, we will consider using sophisticated filter techniques such as the extended Kalman filter.

Acknowledgment

The authors thank Jomo Kenyatta University of Agriculture and Technology for the use of the Integrated Prototyping and Innovation Center. The authors also acknowledge the financial support of the AFRICA-*ai*-JAPAN Project (JICA) and the technical support of Mr. Ben Maniafu.

References

[1] C. Sneider, K. Kurlich, S. Pulos, and A. Friedman, "Learning to control variables with model rockets: A neo-Piagetian study of learning in field settings," Science Education, vol. 68, no. 4, pp. 465–486, 1984.

[2] R. N. Hernandez, H. Singh, S. L. Messimer, and A. E. Patterson, "Design and performance of modular 3-D printed solid-propellant rocket airframes," Aerospace, vol. 4, no. 2, p. 17, 2017.
[3] R. G. Brown and P. Y. Hwang, "Introduction to random signals and applied Kalman filtering: with MATLAB exercises and solutions," Introduction to random signals and applied Kalman filtering: with MATLAB exercises and solutions, 1997.
[4] D. W. Schultz, "Application of the Kalman filter to rocket apogee detection." NARAM46 R&D July, 2004.
[5] O. L. R. Jacobs and O. L. R. Jacobs, Introduction to control theory. Oxford University Press, USA, 1974.
[6] S. Julier and J. K. Uhlmann, "A general method for approximating nonlinear transformations of probability distributions," 1996.
[7] S. J. Julier, J. K. Uhlmann, and H. F. Durrant-Whyte, "A new approach for filtering nonlinear systems," in Proceedings of 1995 American Control Conference-ACC'95, 1995, vol. 3, pp. 1628–1632.
[8] R. E. Kalman, "A new approach to linear filtering and prediction problems," Journal of basic Engineering, vol. 82, no. 1, pp. 35–45, 1960.