Integrated State Estimation Development for Actively Controlled Rockets

Albert Zheng*, Patrick T. Barry†, Hridai Ambati‡, Kandarp Vadia§
Georgia Institute of Technology, Atlanta, GA, USA

This paper details the development process of an inertial navigation system to support a collegiate team's actively controlled jet vanes rocket development efforts. Precise and accurate estimation of the rocket's position, velocity, and attitude are required for successful active feedback controls, validation of the team's in-house 6-DOF simulations, and ensuring reliable real-time state transition detection. First, sensor models are used to convert a 6-DOF reference trajectory to sensor readings. Next, these tools are fed into a ground-based static Kalman Filter for sensor bias and initial attitude determination (IAD), along with an in-flight Extended Kalman Filter for translational and attitude estimation. The validation of these algorithms in simulations was performed with Monte Carlo analysis in MATLAB on multiple trajectories and sensor parameters. Lastly, the algorithms were implemented on a custom flight computer in the STM32 embedded C environment and details the solutions to any practical implementation issues encountered. The paper establishes a step-by-step process of designing a navigation architecture from whiteboard to real-time flight, with performance analysis included.

I. Introduction

The Guidance, Navigation, and Control (GNC) project in the Ramblin' Rocket Club is a collegiate effort to design and build an actively-controlled jet vanes rocket. A fundamental component of active feedback controls is the need for a robust real-time state estimation system that can consistently provide precise and accurate feedback about a rocket's state. High-fidelity simulations and sensor suites are provided first to act as truth data for characterizing navigation performance. Then, four algorithms for state estimation will be described: static bias estimation, initial attitude estimation, attitude propagation, and in-flight translational estimation. Outside of attitude propagation, the other three algorithms fall under the well-defined Extended Kalman Filter (EKF) architecture. Finally, this paper demonstrates the end-to-end design and successful implementation of a state estimation system for a guided rocket under constraints on cost, computational resources, and sensor accuracy.

II. Sensor Modeling

Building the infrastructure of a navigation testbed requires a 6-DOF reference trajectory to act as truth data and a set of sensor models to help execute navigation algorithms in simulations before real flight tests. Due to the limited launch cadence of collegiate rocketry, it becomes especially necessary to have a proper testing suite on the ground. As a result, the GNC team has developed a 6-DOF closed-loop dynamical simulation that generates a trajectory, including position, velocity, and attitude of the rocket body at discrete time steps. The states are expressed in the body frame $b = \{x^b, y^b, z^b\}$ with respect to a custom defined Flat Earth frame $f = \{x^f, y^f, z^f\}$. The Flat Earth frame enables the rocket to ignore Earth's curvature and rotation, contains its origin at the launch pad, and is treated as an inertial frame whose x-axis points upward, y-axis points downrange, and z-axis completes the right-hand rule, which is equivalent to a 90 degree rotation from the popular North-East-Down (NED) frame. The simulation is plugged into a differential equation solver and uses standard 6-DOF equations of motion, integrated controls, CFD-derived aerodynamics, and thrust allocation curves from solid propellant motor static fires. The implementation of this simulation is not detailed in this paper; rather, only the usage and interfacing of this simulation with navigation is explained.

The first key usage of a reference trajectory is the generation of sensor models. Sensor models serve to emulate the output of raw sensor data that one would expect to encounter in real-time flight by adding fake noises and errors on top

*Controls Team Member, Undergraduate, Daniel Guggenheim School of Aerospace Engineering, AIAA Student Member 1603522
†Avionics & Controls Lead, Undergraduate, Daniel Guggenheim School of Aerospace Engineering, AIAA Student Member 1418912
‡Controls Team Member, Undergraduate, Daniel Guggenheim School of Aerospace Engineering, AIAA Student Member 1537560
§Controls Team Member, Undergraduate, Daniel Guggenheim School of Aerospace Engineering, AIAA Student Member 1745264

of the truth state. These models can replace the measurement vectors in a navigation algorithm. The team developed custom accelerometer, gyroscope, magnetometer, and GPS models to support navigation testing[1]. Fixed turn-on biases are denoted as $\boldsymbol{\beta}$, Gaussian noises are denoted as $\boldsymbol{\eta}$, and rotation matrices describing body attitude with respect to the Flat Earth frame are denoted as $\mathbf{R}^{bf}$. The high-level equations of the sensor models and explanations of how each term is obtained is provided below,

$$\begin{aligned}
\mathbf{f}^b &= \mathbf{a}^b_{true} - \mathbf{R}^{bf}\mathbf{g}^{\mathbf{f}} + \boldsymbol{\beta}_a + \boldsymbol{\eta}_a \\
\omega^b &= \omega^b_{true} + \boldsymbol{\beta}_\omega + \boldsymbol{\eta}_\omega \\
\mathbf{p}^f &= \mathbf{p}^f_{true} + \boldsymbol{\eta}_{gps} \\
\mathbf{m}^b &= \mathbf{R}^{bf}\mathbf{m}^f_{true} + \boldsymbol{\beta}_m + \boldsymbol{\eta}_m
\end{aligned}$$

where:

$$\begin{aligned}
\mathbf{f}^b &= \text{body-coordinated specific force} \\
\omega^b &= \text{body angular velocity} \\
\mathbf{p}^f &= \text{position in Flat Earth frame} \\
\mathbf{m}^b &= \text{magnetic field in body frame}
\end{aligned}$$

Since the accelerometer is not guaranteed to be placed at the center of gravity (CG), an inertial measurement unit (IMU) level arm $\mathbf{r}_{12}$ induces rotations and oscillations of the rocket; these provide additional centrifugal accelerations that must be accounted for. Equation 1 shows the conversion of an IMU reading at a lever arm distance away from the CG location at a given time. The CG is assumed to change at a fixed rate along the rocket's axis. Apart from the additional acceleration caused by the Coriolis effect, it assumed that the IMU is fixed on the body and that angular acceleration with respect to the CG is negligible.

$$\mathbf{a}^f_{true} = \mathbf{a}^f_{cg} - \mathbf{R}^{fb}(\omega^b_{bf} \times \mathbf{v}^b_{12} + \omega^b_{bf} \times \omega^b_{bf} \times \mathbf{r}^b_{12}) \tag{1}$$

With the accelerometer model considered, one can examine the GPS sensor. The GPS model includes preprocessing steps that transform raw latitude, longitude, and altitude (LLA) coordinates into an Earth-centered, Earth-fixed (ECEF) coordinate system, which ensures consistent global positioning. The next transformation converts ECEF to an NED frame. Finally, the coordinates are expressed in the local Flat Earth frame. The LLA to ECEF conversion follows standard WGS84 ellipsoidal equations [2] shown below:

$$N = \frac{a}{\sqrt{1 - e^2 \sin^2(\phi)}}$$

$$\begin{aligned}
x_{ECEF} &= (N + h)\cos(\phi)\cos(\lambda) \\
y_{ECEF} &= (N + h)\cos(\phi)\sin(\lambda) \\
z_{ECEF} &= \Big(N(1 - e^2) + h\Big)\sin(\phi)
\end{aligned}$$

where:

a = 6378137 m, the Earth's semi-major axis
e = 0.081819, the eccentricity of the WGS84 ellipsoid
ϕ = latitude (radians)
λ = longitude (radians)
h = altitude above the WGS84 ellipsoid (m)

The transformation of ECEF to NED is performed using a local rotation matrix centered at the reference point. The NED frame is transformed to a Flat Earth frame for visualization and analysis. It ensures that the position data aligns with the sensor models. The gravity vector $\mathbf{g^f}$ is defined assuming a standard gravitational acceleration (g) of 9.807 m/s^2 where $\mathbf{g}^f = \begin{bmatrix} g & 0 & 0 \end{bmatrix}^T$. The World Magnetic Model 2020 is used to obtain the true magnetic field $\mathbf{m}^f_{true}$ by specifying the LLA coordinates of the Flat Earth frame.

Finally, implementation into flight software needs to convert raw sensor data into the desired measurement vectors in the navigation algorithm. The equations used to calculate this will be the inverse of what is used in sensor modeling. For example, Equation 2 solves for the CG acceleration instead of the true acceleration in Equation 1.

$$\mathbf{a}^f_{cg} = \mathbf{a}^f_{true} + \mathbf{R}^{fb}(\boldsymbol{\omega}^b_{bf} \times \mathbf{v}^b_{12} + \boldsymbol{\omega}^b_{bf} \times \boldsymbol{\omega}^b_{bf} \times \mathbf{r}^b_{12}) \tag{2}$$

III. Ground Estimation

Before flight, the ground estimator solves two problems: calibrating the substantial bias of commercial-off-the-shelf (COTS) sensors and obtaining a precise estimate of the initial attitude of the rocket. Properly estimating the accelerometer and gyroscope bias improves the performance of the in-flight translational EKF, detailed in Section IV.B. Properly estimating the initial attitude from an onboard accelerometer and magnetometer is necessary because in-flight quaternions are estimated from integration of angular velocity, which requires a proper initial condition to initialize the algorithm. The ground estimator consists of a pair of static Kalman Filters, which take advantage of static state transition dynamics, or an unchanging state vector. The first filter is a static linear Kalman Filter that converges to an optimal estimate of the accelerometer and gyroscope biases, then autonomously calibrates the accelerometer and gyroscope. The second filter is a static Extended Kalman Filter that converges to an optimal estimate of the initial quaternion using fused, calibrated accelerometer and magnetometer measurements.

A. Bias Estimation

Accelerometers and gyroscopes both need to be calibrated upon powering up due to electrical imperfections. If not properly accounted for, these errors can propagate themselves in subsequent state estimation algorithms, leading to inaccurate navigation solutions. Commonly known calibration issues are turn-on bias, bias drift, and axis misalignment. For high-powered rockets at low altitudes, which is the use case of these state estimation algorithms, flight time is low, so it can be assumed that bias drift is negligible and all bias terms except for turn-on bias have a negligible effect in integration drift. Hence, the bias estimator only needs to estimate six states: the built-in biases on each of the three accelerometer measurements as well as the three gyroscope measurements.

$$\mathbf{x}_{bias} = \begin{bmatrix} \beta_{a,x} & \beta_{a,y} & \beta_{a,z} & \beta_{\omega,x} & \beta_{\omega,y} & \beta_{\omega,z} \end{bmatrix}^T$$

Since the dynamics are static, the state transition function $\mathbf{f}(\hat{\mathbf{x}}_{n,n})$ predicts future bias states as identical to present bias states, and the state transition Jacobian $\frac{d\mathbf{f}}{d\mathbf{x}}\big|_{\hat{\mathbf{x}}_{n,n}}$ is identity $\mathbf{I}_6$. Assuming the rocket is oriented perfectly upwards during calibration, the observation function $\mathbf{h}(\hat{\mathbf{x}}_{n,n})$ directly maps accelerometer and gyroscope measurements to states, and the observation Jacobian $\frac{d\mathbf{h}}{d\mathbf{x}}\big|_{\hat{\mathbf{x}}_{n,n}}$ is identity $\mathbf{I}_6$. Off-diagonal terms in the process noise covariance matrix $\mathbf{Q}$ and the measurement noise covariance matrix $\mathbf{R}$ can typically be zero in this case, because there is no coupling of measurements on different axes with each other. The measurement noise covariance $\mathbf{R}$ can be expressed in terms of sensor uncertainty specified by the sensor data sheet.

$$\mathbf{f}(\hat{\mathbf{x}}_{n,n}) = \hat{\mathbf{x}}_{n,n} \qquad \mathbf{h}(\hat{\mathbf{x}}_{n,n}) = \hat{\mathbf{x}}_{n,n} + \begin{bmatrix} 0 & 0 & g & 0 & 0 & 0 \end{bmatrix}^T \qquad \mathbf{R} = \begin{bmatrix} \sigma_a^2 \mathbf{I}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \sigma_\omega^2 \mathbf{I}_3 \end{bmatrix}$$

Using custom sensor models for the accelerometer and gyroscope as specified in Section II, a set of noisy but static sensor measurements is fed into a Kalman Filter architecture to validate bias estimation. Convergence is incredibly rapid due to a low process noise covariance, which reflects confidence in the static dynamics of the system. Once the bias estimates and diagonal terms of the error covariance are all below a specified threshold, the system deems a bias solution successful and stores the final estimates as fixed biases for all operational stages moving forward. A comparison of raw sensor measurements with calibrated measurements after the bias solution has been solved is shown in Figure 1.

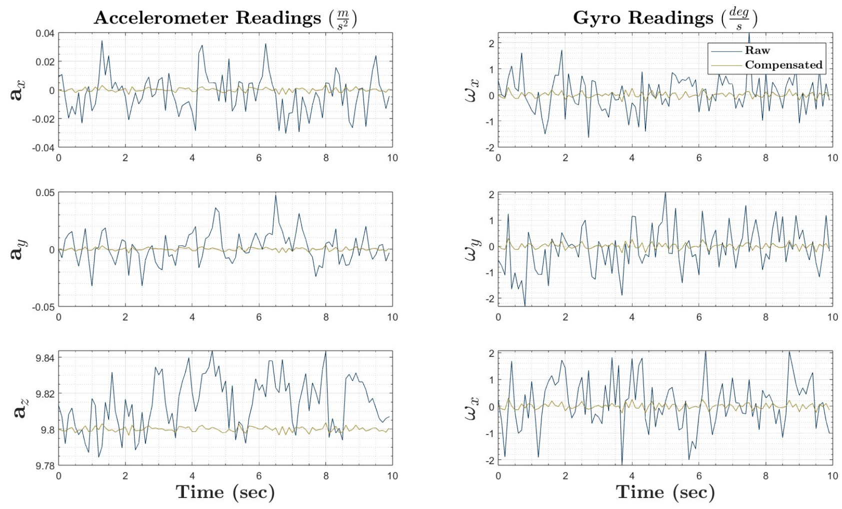

Figure 1. Comparison of raw and calibrated accelerometer and gyroscope data.

B. Initial Attitude Determination

Initial attitude determination is the next step in the ground estimation process using a fusion of calibrated accelerometer and magnetometer measurements. Gyroscopes are not used because typical COTS gyroscopes do not have sufficiently high fidelity to detect Earth's angular rate; otherwise, gyroscopes can also be used for attitude estimation using gyrocompassing. Additionally, low-altitude rockets do not require enough degree of precision for gyrocompassing to make a significant difference. Finally, rather than develop custom magnetometer calibration methods, one can use open source magnetometer calibration software such as Magneto, which allows for manual calibration using provided soft and hard iron calibration data.

According to the infamous TRIAD method [3], a minimum of two unit vectors is required to obtain an orientation estimate, so this section relies on just the gravity and magnetic north unit vectors. A snapshot estimate, an estimate at a single timestep, of the attitude can be obtained by using body-coordinated accelerometer measurements $\mathbf{a}^b$ and magnetometer measurements $\mathbf{m}^b$ to construct a rotation matrix that rotates vectors expressed in the NED frame $v = \{x^v, y^v, d^v\}$ to the body frame. The snapshot rotation matrix then converts to a corresponding snapshot quaternion solution $\mathbf{q}_{bv}$ that defines the attitude of the body frame with respect to the local vehicle-fixed NED. While ambiguity appears to arise in the scalar term of the quaternion, the decision to choose a positive or negative scalar is purely conventional. Both choices lead to an identical rotation due to quaternion properties; therefore, consistency of choice is more important. The widespread standard, which is used here, chooses a positive scalar.

$$\mathbf{e}_z^b = -\frac{\mathbf{a}^b}{|\mathbf{a}^b|}$$

$$\mathbf{e}_y^b = \frac{\mathbf{e}_z^b \times \mathbf{m}^b}{|\mathbf{e}_z^b \times \mathbf{m}^b|}$$

$$\mathbf{e}_x^b = \frac{\mathbf{e}_y^b \times \mathbf{e}_z^b}{|\mathbf{e}_y^b \times \mathbf{e}_z^b|}$$

$$\mathbf{R}^{bv} = \begin{bmatrix} \mathbf{e}_x^b & \mathbf{e}_y^b & \mathbf{e}_z^b \end{bmatrix}^T$$

$$q_0 = \pm\frac{1}{2}\sqrt{\mathrm{Tr}(\mathbf{R}^{bv}) + 1}$$

$$\mathbf{q_{1:3}} = \frac{1}{4q_0}\begin{bmatrix} R_{23} - R_{32} \\ R_{31} - R_{13} \\ R_{12} - R_{21} \end{bmatrix}$$

$$\mathbf{q}_{bv} = \begin{bmatrix} q_0 \\ \mathbf{q_{1:3}} \end{bmatrix}$$

While this process indeed provides an adequate initial quaternion solution, snapshot estimates can be refined using a static Extended Kalman Filter to give a more confident estimate of the initial quaternion to combat initial accelerometer and magnetometer noise and unexpected spikes. The main significance of the snapshot method in this paper is to provide a high accuracy initial guess in the Extended Kalman Filter.

In this filter, the state vector of interest is the quaternion, $\mathbf{x} = \mathbf{q}_{bv}$. Since dynamics are static, the state transition function and state transition Jacobian are identical to the ones specified in the bias estimation section of this paper. Since directions are the key component of information, the measurement vector $\mathbf{z} = \begin{bmatrix} \hat{\mathbf{a}}^b & \hat{\mathbf{m}}^b \end{bmatrix}^T$ is defined as normalized accelerometer and magnetometer readings.

The observation function maps the quaternion state to the measurement vector. Since the measurement function is nonlinear, an observation Jacobian must be computed as well. In this case, the quaternion state can be directly used as a rotation operation.

$$\mathbf{h}(\hat{\mathbf{x}}_{n,n}) = \begin{bmatrix} \mathbf{R}^{bv}\hat{\mathbf{a}}^v \\ \mathbf{R}^{bv}\hat{\mathbf{m}}^v \end{bmatrix} = 2 \begin{bmatrix} g_x(\frac{1}{2} - q_2^2 - q_3^2) + g_y(q_0q_3 + q_1q_2) + g_z(q_1q_3 - q_0q_2) \\ g_x(q_1q_2 - q_0q_3) + g_y(\frac{1}{2} - q_1^2 - q_3^2) + g_z(q_0q_1 - q_2q_3) \\ g_x(q_0q_2 + q_1q_3) + g_y(q_2q_3 + q_0q_1) + g_z(\frac{1}{2} - q_1^2 - q_2^2) \\ m_x(\frac{1}{2} - q_2^2 - q_3^2) + m_y(q_0q_3 + q_1q_2) + m_z(q_1q_3 - q_0q_2) \\ m_x(q_1q_2 - q_0q_3) + m_y(\frac{1}{2} - q_1^2 - q_3^2) + m_z(q_0q_1 + q_2q_3) \\ m_x(q_0q_2 + q_1q_3) + m_y(q_2q_3 - q_0q_1) + m_z(\frac{1}{2} - q_1^2 - q_2^2) \end{bmatrix}$$

$$\frac{d\mathbf{h}}{d\mathbf{x}} = 2 \begin{bmatrix} g_xq_0 + g_yq_3 - g_zq_2 & g_xq_1 + g_yq_2 + g_zq_3 & -g_xq_2 + g_yq_1 - g_zq_0 & -g_xq_3 + g_yq_0 + g_zq_1 \\ -g_xq_3 + g_yq_0 + g_zq_1 & g_xq_2 - g_yq_1 + g_zq_0 & g_xq_1 + g_yq_2 + g_zq_3 & -g_xq_0 - g_yq_3 + g_zq_2 \\ g_xq_2 - g_yq_1 + g_zq_0 & g_xq_3 - g_yq_0 - g_zq_1 & g_xq_0 + g_yq_3 - g_zq_2 & g_xq_1 + g_yq_2 + g_zq_3 \\ m_xq_0 + m_yq_3 - m_zq_2 & m_xq_1 + m_yq_2 + m_zq_3 & -m_xq_2 + m_yq_1 - m_zq_3 & -m_xq_3 + m_yq_0 + m_zq_1 \\ -m_xq_3 + m_yq_0 + m_zq_1 & m_xq_2 - m_yq_1 + m_zq_0 & m_xq_1 + m_yq_2 + m_zq_3 & -m_xq_0 - m_yq_3 + m_zq_2 \\ m_xq_2 - m_yq_1 + m_zq_0 & m_xq_3 - m_yq_0 - m_zq_1 & m_xq_0 + m_yq_3 - m_zq_1 & m_xq_1 + m_yq_2 + r_zq_3 \end{bmatrix}$$

The process noise covariance $\mathbf{Q}$ can be very low since it is a static system. The measurement noise covariance $\mathbf{R}$ can be expressed in terms of sensor variances:

$$\mathbf{R} = \begin{bmatrix} \sigma_a^2\mathbf{I}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \sigma_m^2\mathbf{I}_3 \end{bmatrix}$$

Custom sensor models for the accelerometer and magnetometer are used to emulate noisy measurements and are fed into this Kalman Filter. Results are shown in Figure 2. The magnetic field in the NED frame at a particular latitude and longitude can be obtained from the World Magnetic Model 2020. Additionally, because the EKF is not an optimal estimator, the norm constraint of quaternions must be applied at each iteration.

$$\mathbf{q}_{bv,\ n+1} = \frac{\mathbf{q}_{bv,\ n}}{|\mathbf{q}_{bv,\ n}|}$$

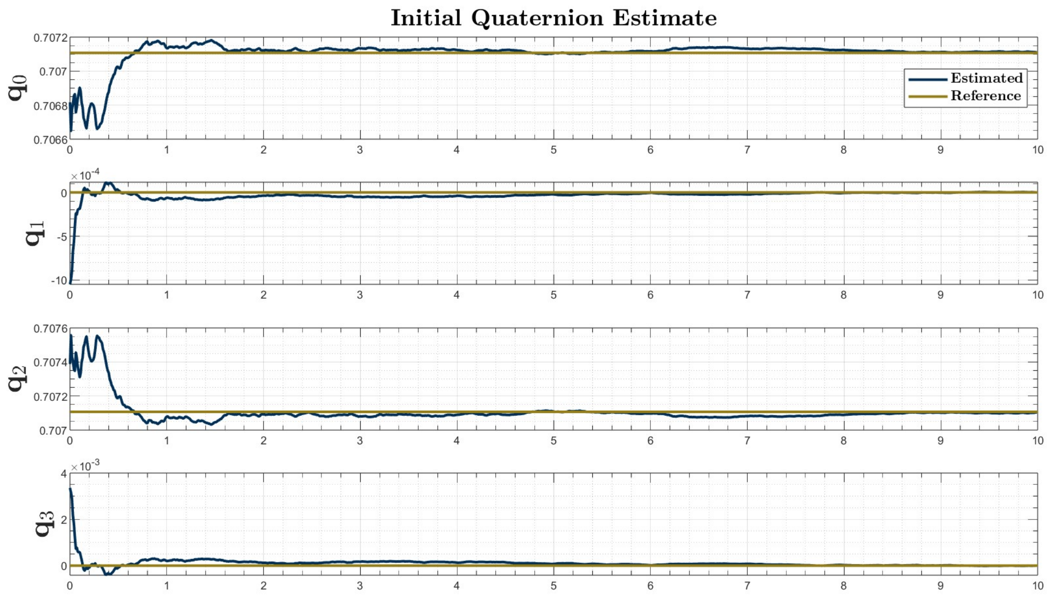

Figure 2. Initial static quaternion estimate over time.

IV. In-Flight Estimation

Once the rocket ignites its motor and leaves the launchpad, it is appropriate to apply a in-flight navigation mode. This mode consists of two coupled algorithms: 1) a translational Extended Kalman Filter that estimates position in the Flat Earth frame and velocity in the body frame, and 2) a gyro-propagated quaternion algorithm. These algorithms provide real-time knowledge for the onboard controller, drive state machine transition detection (motor burnout, apogee, chute deployment, etc.), and provide controller troubleshooting and post-flight analysis.

A. In-Flight Attitude Estimation

Quaternions are used to represent the attitude of the vehicle relative to the Flat Earth frame. The attitude propagator is initialized with a precise attitude estimate from the ground estimator. The only inputs to the in-flight attitude estimation algorithm are gyroscope measurements about each of the three body axes: ω_x, ω_y, and ω_z. No reference sensors are used to filter the attitude estimate. While this is prone to building error over a long period of time due to integration drift, flight times of low-altitude rockets are short enough that the drift does not substantially degrade attitude estimation performance.

At every time step, the gyroscope measurements are taken and converted into an instantaneous rotation quaternion, $\mathbf{q}_\Delta$, representing the rotation that occurred in the previous time step. Letting $\boldsymbol{\omega} = [\omega_x, \omega_y, \omega_z]^T$, the following holds:

$$\theta = ||\boldsymbol{\omega}|| \cdot \Delta t \tag{3}$$

$$\mathbf{q}_\Delta = \begin{bmatrix} sin(\frac{\theta}{2}) \\ \boldsymbol{\omega} \cdot cos(\frac{\theta}{2}) \end{bmatrix} \tag{4}$$

To update the quaternion representing the rocket's attitude, the current rocket quaternion and the instantaneous rotation quaternion are multiplied together.

$$\mathbf{q}_{n+1} = \mathbf{q}_n \otimes \mathbf{q}_\Delta \tag{5}$$

It should be noted that quaternion multiplication is a special type of operation, well-defined in the literature. [4] Additionally, quaternions must always retain a norm of 1, so the quaternion $\mathbf{q}_{n+1}$ undergoes normalization before the algorithm's next iteration.

This attitude estimation algorithm runs over the rocket's entire flight. The quaternion representing the rocket's attitude may then be fed into the rocket's state vector for the controller to act upon. The algorithm is summarized in Algorithm 1, and a Monte Carlo simulation of its error performance during the controlled flight time is shown in Figure 3, where the algorithm is run under randomly sampled gyroscope noise densities with a standard deviation of 0.005 radians. As proven by the Monte Carlo simulation, since the flight time is so short, the gyroscope drift from dynamic and sensor noise encountered during the flight is minimal.

Algorithm 1 In-Flight Attitude Estimation

```
Given: $\mathbf{q}(0) \leftarrow$ (Initial Attitude Determination)
while $t_{curr} < t_{final}$ do
    $\mathbf{q}_\Delta \leftarrow \omega_x, \omega_y, \omega_z, \Delta t$
    $\mathbf{q}_{n+1} \leftarrow \mathbf{q}_n \otimes \mathbf{q}_\Delta$
    $\mathbf{q}_{n+1} \leftarrow \frac{\mathbf{q}_{n+1}}{||\mathbf{q}_{n+1}||}$
    $n \leftarrow n + 1$
end while
```

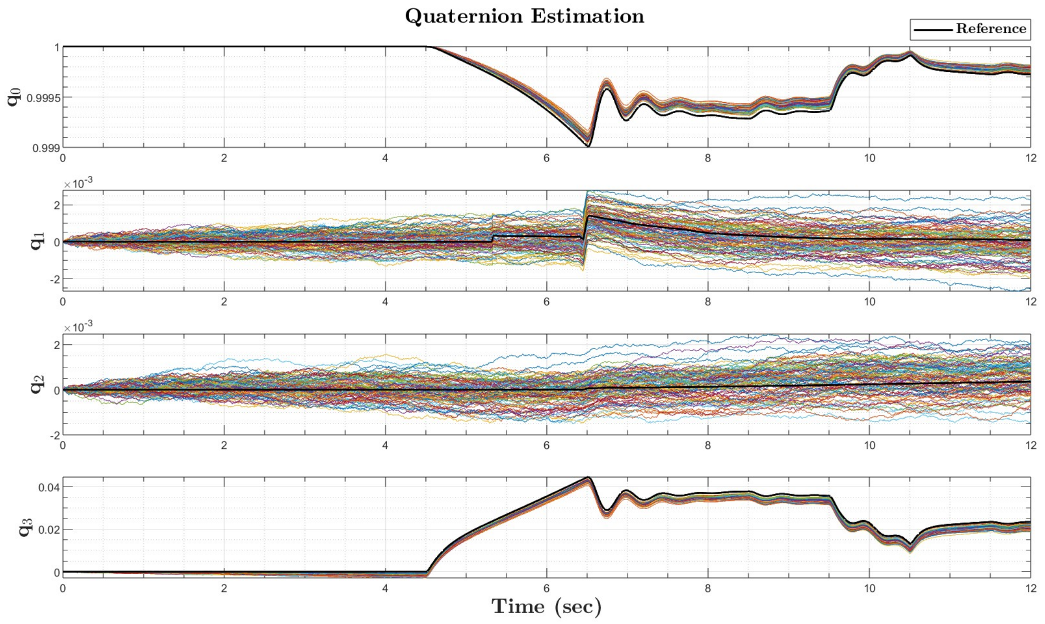

Figure 3. 100-Run Monte Carlo simulation of quaternion propagation algorithm with varying gyroscope noise parameters.

B. In-Flight Extended Kalman Filter

The translational state Extended Kalman Filter estimates a six-element state containing position in the Flat Earth frame and body-coordinated velocity. The state is chosen as such because the controller gain scheduler requires knowledge of the body frame velocity, so estimating velocity in other frames would introduce additional compute overhead. The state vector's dynamics are propagated using accelerometer measurements and its observations correct any dynamic or accelerometer hardware noise using a GPS in the update step. Since GPS is not always reliable and has proven to struggle with consistent GPS signal lock throughout the flight regime, simple fault detection measures are taken to only rely on the accelerometer, or only execute the predict step and skip the update step, when GPS is detected to be outputting faulty or incorrect measurements.

Instead of predicting the rocket's motion using forces, kinematics can be used with more success. This is because characterizations of the rocket's aerodynamics and thrust models are not nearly as accurate as measuring the force acting on the rocket through the accelerometer. Hence, the accelerometer reading acts as a dynamic parameter rather than a raw measurement in the navigation algorithm. Also, effects of acceleration on position propagation are negligible, as acceleration updates position on the order of the square of the time step, which is on the order of hundredths of a second. The following sequence of calculations can be merged to form the state transition function to predict the future state $\mathbf{x}_{n,\ n+1}$, in which the state transition Jacobian $\frac{d\mathbf{f}}{d\mathbf{x}}$ can be subsequently derived.

$$\begin{aligned}
\mathbf{v}_n^f &= \mathbf{q}_{bf} \otimes \mathbf{v}_n^b \otimes \mathbf{q}_{bf}^* \\
\mathbf{p}_{n+1}^f &= \mathbf{p}_n^f + \Delta t \mathbf{v}_n^f \\
\mathbf{v}_{n+1}^f &= \mathbf{v}_n^f + \Delta t \mathbf{a}_n^f
\end{aligned}$$

$$\mathbf{f}(\mathbf{x}_{n+1,\ n}) = \begin{bmatrix} p^f_{x,\ n+1} & v^f_{x,\ n+1} & p^f_{y,\ n+1} & v^f_{y,\ n+1} & p^f_{z,\ n+1} & v^f_{z,\ n+1} \end{bmatrix}^T$$

$$\frac{d\mathbf{f}}{d\mathbf{x}} = \begin{bmatrix}
1 & \Delta t(2q_0^2 + 2q_1^2 - 1) & 0 & \Delta t(2q_0q_3 + 2q_1q_2) & 0 & -\Delta t(2q_0q_2 - 2q_1q_3) \\
0 & 1 & 0 & 0 & 0 & 0 \\
0 & -\Delta t(2q_0q_3 - 2q_1q_2) & 1 & \Delta t(2q_0^2 + 2q_2^2 - 1) & 0 & \Delta t(2q_0q_1 + 2q_2q_3) \\
0 & 0 & 0 & 1 & 0 & 0 \\
0 & \Delta t(2q_0q_2 + 2q_1q_3) & 0 & -\Delta t(2q_0q_1 - 2q_2q_3) & 1 & \Delta t(2q_0^2 + 2q_3^2 - 1) \\
0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}$$

The observation step is performed after taking a GPS position measurement. Note that measurement inputs in this EKF are treated as one-to-one relationships, meaning that measurements are equivalent to the positions the GPS is sensing but in the Flat Earth frame. This is because preprocessing of the raw LLA reading from GPS is done beforehand to create the appearance of a sensed position. Additionally, the observation Jacobian $\frac{d\mathbf{h}}{d\mathbf{x}}$ is easily derived as follows.

$$\mathbf{z} = \begin{bmatrix} p^f_{x,\ GPS} & p^f_{y,\ GPS} & p^f_{z,\ GPS} \end{bmatrix}^T$$

$$\mathbf{h}(\mathbf{x}_{n,\ n}) = \begin{bmatrix} p^f_{x,\ n,n} & p^f_{y,\ n,n} & p^f_{z,\ n,n} \end{bmatrix}$$

$$\frac{d\mathbf{h}}{d\mathbf{x}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

The tuning procedure of the Extended Kalman Filter involves varying the ratio of the process noise covariance Q and measurement covariance R. The GPS has an internal navigation module that outputs measurement covariances in real-time, acting as a baseline that doesn't require manual tuning. The process noise covariance is derived from basic kinematics occurring in the state transition function, with minor tweaking and scaling of σ_a, accelerometer noise variance, to adjust for accelerometer noise and dynamic model inaccuracies.

$$\mathbf{Q}_n = \sigma_a^2 \begin{bmatrix}
\frac{\Delta t^4}{4} & \frac{\Delta t^3}{2} & 0 & 0 & 0 & 0 \\
\frac{\Delta t}{2} & \Delta t^2 & 0 & 0 & 0 & 0 \\
0 & 0 & \frac{\Delta t^4}{4} & \frac{\Delta t}{2} & 0 & 0 \\
0 & 0 & \frac{\Delta t^3}{2} & \Delta t^2 & 0 & 0 \\
0 & 0 & 0 & 0 & \frac{\Delta t^4}{4} & \frac{\Delta t}{2} \\
0 & 0 & 0 & 0 & \frac{\Delta t^3}{2} & \Delta t^2
\end{bmatrix} \qquad \mathbf{R}_n = \begin{bmatrix} \sigma_{GPS,\ x} & 0 & 0 \\ 0 & \sigma_{GPS,\ y} & 0 \\ 0 & 0 & \sigma_{GPS,\ z} \end{bmatrix}$$

The integrated in-flight navigation steps are executed in Algorithm 2, showcasing the interdependence of the in-flight attitude propagator and in-flight translational EKF algorithms. A Monte Carlo simulation is also conducted on the translational EKF by randomly sampling accelerometer and GPS noise densities (in conjunction with the Monte Carlo attitude propagation) of 0.01 m/s and 1 m respectively, along with initial state guesses and initial covariance guesses.

Analysis of the translational EKF performance is conducted in terms of deviation from a reference trajectory generated by the team's in-house closed-loop 6-DOF dynamic model. Flat Earth position and body velocity error trajectories are provided in Figures 4 and 5, respectively. Convergence of the translational EKF in terms of covariances are provided in Figure 6.

Algorithm 2 In-Flight 6-DOF Navigation

```
while $t_{curr} < t_{final}$ do
    $p_{sensed} \leftarrow LLAtoWorld(lat, lon, alt)$
    $a^b_{sensed} \leftarrow accelerometerModel()$
    $\omega_{bi,sensed} \leftarrow gyroModel()$
    if GPS Lock then
        $z \leftarrow p_{sensed}$
        $K_n \leftarrow P_{n,n-1}(\frac{dh}{dx})^T \times (\frac{dh}{dx}P_{n,n-1}(\frac{dh}{dx})^T + R_n)^{-1}$
        $x_{n,n} \leftarrow x_{n,n-1} + K_n(z_n - h(x_{n,n-1}))$
        $P_{n,n} \leftarrow (I - K_n\frac{dh}{dx})P_{n,n-1} \times (I - K_n\frac{dh}{dx})^T + K_n R_n K_n^T$
    else  Skip Update Step
        $x_{n,n} \leftarrow x_{n,n-1}$
        $P_{n,n} \leftarrow P_{n,n-1}$
    end if
    $x_{n+1,n} \leftarrow f(x_{n,n}, q_{curr}, a^b_{sensed})$
    $P_{n+1,n} \leftarrow \frac{df}{dx}P_{n,n}(\frac{df}{dx})^T + Q$
    $q_{new} \leftarrow f_q(q_{curr}, \omega_{bi,sensed})$
end while
```

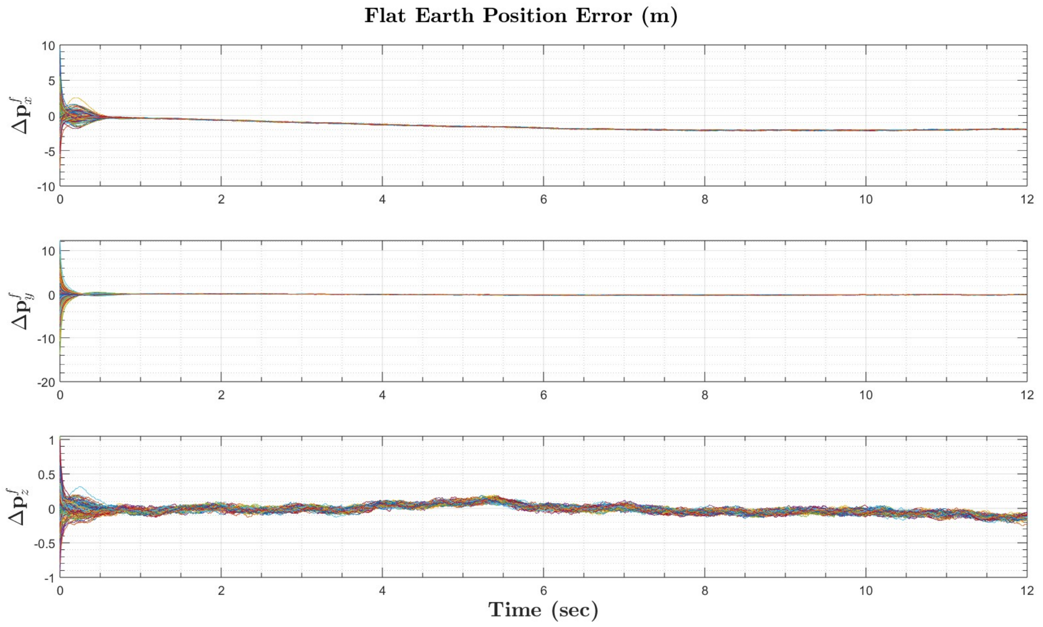

Figure 4. 100-run Monte Carlo simulation of error of Flat Earth position estimate over time.

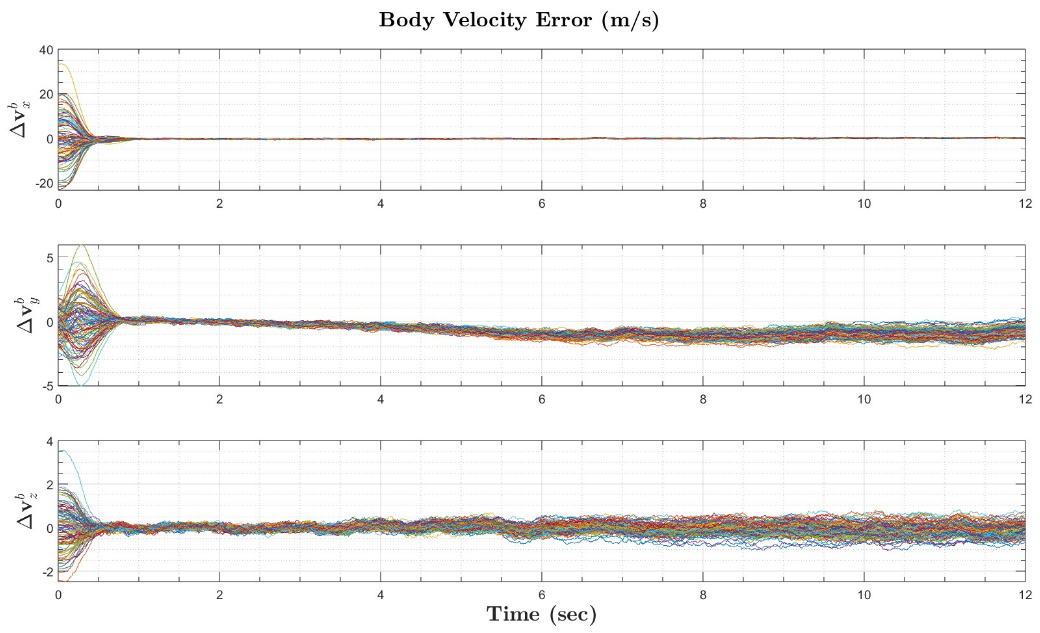

Figure 5. 100-run Monte Carlo simulation of error of body velocity estimate over time.

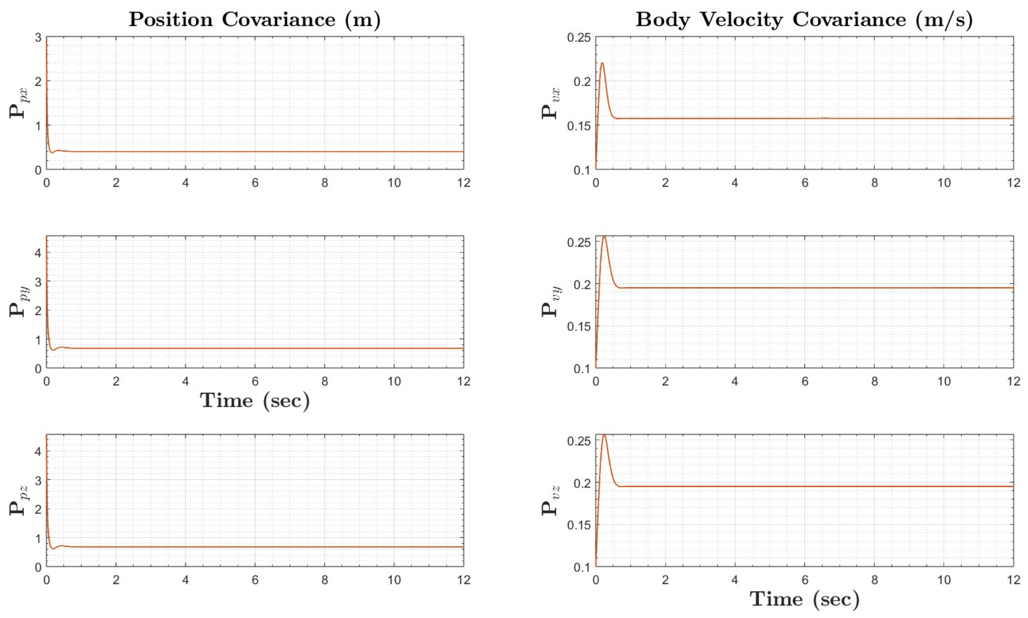

Figure 6. Convergence of position and velocity covariances.

V. Flight Software Implementation

A. Hardware

The sensors used in this project included an accelerometer, a gyroscope, a magnetometer, and a GPS. The accelerometer, gyroscope, and magnetometer used were contained in Analog Devices' ADIS16500 IMU, and the GPS used was a u-blox ZED-F9P-04B. The GPS antenna used was mounted away from the rest of the electronic hardware to prevent electromagnetic interference from harming the GPS' ability to obtain lock. GPS with real-time kinematics (RTK) was used, with GPS measurements being reported in LLA. The LLA position measurement was subsequently converted to the ECEF frame before being transformed into the custom Flat Earth frame used by the jet vanes rocket. As specified in section II, the Coriolis effect experienced by the accelerometer due to the IMU not sitting at the vehicle's center of mass were removed before measurement insertion into the navigation algorithms.

All the state estimation code was run on an STM32H7 processor. This processor was embedded in a custom flight computer PCB that also contained the sensors, an SD card slot, a flash chip, an Xbee antenna, and debug tools. A separate power PCB powered the flight computer board with a single lithium polymer (LiPo) battery as the ultimate power source for all avionics.

B. Software

The code running on the STM32 microcontroller unit (MCU) was written entirely in C, and all algorithms were translated from MATLAB into C. The task of reading from the sensors and running the state estimation algorithms is completed by a "State Estimation MCU", and the state estimate is transmitted to a second microcontroller, the "Main MCU," for conducting Linear Quadratic Regulator attitude control as well as data logging, telemetry, and servo actuation commands.

The in-flight attitude estimation and Extended Kalman Filter both run at 30 Hz according to a timer in the State Estimation MCU, with sensor readings being collected at higher frequencies. Because it is possible for the GPS to lose its navigation solution under high acceleration, the in-flight EKF is written to run solely its "predict" step if the GPS has lost lock, only utilizing the accelerometer.

VI. Conclusion

In this paper, a comprehensive development approach and algorithmic basis to state estimation of an actively-controlled collegiate rocket was provided. By showcasing the implementation details of these algorithms, a robust navigation architecture framework is demonstrated that can easily be extended beyond the scope of this particular rocket. The successful implementation of these algorithms on embedded hardware showcases the feasibility of advanced state estimation techniques in collegiate-level rocketry projects, setting a precedent for future teams striving to push the boundaries of guided and non-guided rockets alike.

References

[1] Markley, F. L., and Crassidis, J. L., *Fundamentals of Spacecraft Attitude Determination and Control*, Springer, 2014.

[2] Groves, P., *Principles of GNSS, Inertial, and Multisensor Integrated Navigation Systems*, 2^{nd} ed., Artech, 2008.

[3] Markley, F. L., "Attitude Determination Using Two Vector Measurements," *NASA Goddard Space Flight Center*, 1998.

[4] Kuipers, J. B., *Quaternions and Rotation Sequences: A Primer With Applications to Orbits, Aerospace, and Virtual Reality*, Princeton University Press, 1999.