Calibration of an Accelerometer Using GPS Measurements

Alonzo Lopez

December 9, 2020

1 Abstract

An accelerometer may be used in conjunction with GPS measurements to accurately measure a position and velocity state over time. In order to accurately measure the state, a filter is constructed to fuse the estimates accounting for noise. Sensor noise is modeled as Gaussian noise. Additionally, the accelerometer has a bias that must be calibrated, or estimated. This paper describes the process of calibrating the accelerometer bias using GPS measurements and a Kalman Filter.

2 Introduction

A linear Kalman Filter estimates the state of a dynamic system using noisy sensor measurements in a way that minimizes the variance. Given a system with a dynamic model of the state transitions, Gaussian process noise, and sensor models with Gaussian noise, we can construct a Kalman Filter to accurately estimate the state of the system. Here we construct a filter that accurately estimates the bias of an accelerometer as well as position and velocity of a vehicle under acceleration with the help of a noisy GPS sensor. The Kalman Filter is constructed so its model is independent of the actual acceleration.

3 Modeling and Kalman Filter Design

The experiment considers a vehicle that accelerates in one dimension in an inertial frame by the function

$$a(t) = 10sin(\omega t) \quad m/s^2$$

3.1 Accelerometer Model

The accelerometer measures at sample times t_j with a frequency of 200 Hz. The accelerometer is modeled with additive white Gaussian noise with zero mean and variance $V = 0.0004\ (m/s^2)^2$. The accelerometer has a bias b_a with a priori statistics $b_a \sim N(0, 0.01\ (m/s^2)^2)$. Together with bias and Gaussian noise, the accelerometer is modeled as

$$a_c(t_j) = a(t_j) + b_a + w(t_j)$$

From this acceleration, the position and velocity can be computed by an Euler formula as

$$v_c(t_{j+1}) = v_c(t_j) + a_c(t_j)\Delta t$$

$$p_c(t_{j+1}) = p_c(t_j) + v_c(t_j)\Delta t + a_c(t_j)\frac{\Delta t^2}{2}$$

with initial conditions $v_c(0) = \bar{v}_0, p_c(0) = \bar{p}_0 = 0$

3.2 GPS Model

The GPS receiver gives position and velocity estimates in an inertial space at sample times t_i with a frequency of 5 Hz (synchronized with the accelerometer). The GPS measurements come in the following form:

$$z_{1i} = x_i + \eta_{1i}$$

$$z_{2i} = v_i + \eta_{2i}$$

The noise η is modeled with a zero mean Gaussian distribution as

$$\begin{bmatrix} \eta_{1i} \\ \eta_{2i} \end{bmatrix} \sim N\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1\ m^2 & 0 \\ 0 & 16\ (cm/s)^2 \end{bmatrix} \right)$$

$$V = \begin{bmatrix} 1\ m^2 & 0 \\ 0 & 16\ (cm/s)^2 \end{bmatrix}$$

3.3 Derivation of the Dynamic Model

The assumption is made that the actual or true acceleration is integrated via the same Euler integration that is used to compute the position and velocity estimates from the accelerometer data.

$$v_E(t_{j+1}) = v_E(t_j) + a(t_j)\Delta t$$

$$p_E(t_{j+1}) = p_E(t_j) + v_E(t_j)\Delta t + a(t_j)\frac{\Delta t^2}{2}$$

Where here, $a(t_j)$ is the true acceleration. The initial conditions are drawn from

$$v(0) = v_E(0) \sim N(\bar{v}_0, M_0^v)$$

$$p(0) = p_E(0) \sim N(\bar{p}_0, M_0^p)$$

Using the two sets of Euler integration formulas, we can construct a stochastic discrete time system that is approximately independent of the acceleration profile as

$$\delta x(t_j + 1) = \Phi \delta x(t_j) + \Gamma w(t_j)$$

$$\begin{bmatrix} \delta p_E(t_{j+1}) \\ \delta v_E(t_{j+1}) \\ b(t_{j+1}) \end{bmatrix} = \begin{bmatrix} 1 & \Delta t & \frac{\Delta t^2}{2} \\ 0 & 1 & -\Delta t \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \delta p_E(t_j) \\ \delta v_E(t_j) \\ b(t_j) \end{bmatrix} + \begin{bmatrix} -\frac{\Delta t^2}{2} \\ -\Delta t \\ 0 \end{bmatrix} w(t_j)$$

This stochastic discrete time system is used to consruct the Kalman Filter where the state is

$$\hat{\delta} x(t_j) = \begin{bmatrix} \hat{\delta} p(t_j) \\ \hat{\delta} v(t_j) \\ \hat{b}(t_j) \end{bmatrix} = \begin{bmatrix} \hat{p}(t_j) - p_c(j) \\ \hat{v}(t_j) - v_c(j) \\ \hat{b}(t_j) \end{bmatrix}$$

3.4 The Kalman Filter Algorithm

The Kalman filter starts with an initial state estimate and associated covariance matrix as below.

$$\bar{\delta} x(0) = \begin{bmatrix} \bar{\delta} p(t_0) \\ \bar{\delta} v(t_0) \\ \bar{b}(t_0) \end{bmatrix} = \begin{bmatrix} \hat{p}(t_0) - p_c(0) \\ \hat{v}(t_0) - v_c(0) \\ \hat{b}(t_0) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$M_0 = \begin{bmatrix} 100m^2 & 0 & 0 \\ 0 & 1(m/s)^2 & 0 \\ 0 & 0 & 0.01(m/s^2)^2 \end{bmatrix}$$

The initial covariance matrix is derived from the a priori statistics used to run the Monte Carlo simulation by initializing x_0, v_0, and b_a from Gaussian distributions defined by

$$x_0 \sim N(0m, 100m^2)$$

$$v_0 \sim N(100m/s, 1(m/s)^2)$$

$$b_a \sim N(0, 0.01(m/s^2)^2)$$

At time $t = 0$, the GPS outputs its first sensor measurement of the form

$$z_i = Hx_i + \eta$$

$$\begin{bmatrix} z_{1i} \\ z_{2i} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_i \\ v_i \end{bmatrix} + \begin{bmatrix} \eta_{1i} \\ \eta_{2i} \end{bmatrix}$$

where the noise η is described by its previously mentioned mean and covariance. This measurement will likely conflict with our current state estimate, so it must be resolved using the a posteriori update

$$K_0 = M_0 H^T (H M_0 H^T + V)^{-1}$$

$$\hat{\delta}x(0) = \bar{\delta}x(0) + M_0 H^T V^{-1}(z_0 - H\bar{x}(0))$$
$$P_0 = (I - K_0 H) M_0 (I - K_0 H)^T + K_0 V K_0^T$$

At the next time step, we first construct an a priori estimate, then receive a new GPS sensor measurement, and again resolve the difference between the a priori estimate and the sensor measurement using the a posteriori update equations above. The a priori update is constructed by propagating the mean and covariance of the state estimate as

$$\bar{x}_1 = \Phi \hat{x}_0$$

$$M_1 = \Phi P_0 \Phi^T + \Gamma W \Gamma^T$$

When the measurement z_1 comes in, the a posteriori update is run as

$$K_1 = M_1 H^T (H M_1 H^T + V)^{-1}$$

$$\hat{\delta}x(1) = \bar{\delta}x(1) + M_1 H^T V^{-1}(z_1 - H\bar{x}(1))$$
$$P_1 = (I - K_1 H) M_1 (I - K_1 H)^T + K_1 V K_1^T$$

This process of a priori and a posteriori update is repeated for the duration of each simulation. The a priori update equations are

$$\bar{x}_k = \Phi \hat{x}_{k-1}$$

$$M_k = \Phi P_{k-1} \Phi^T + \Gamma W \Gamma^T$$

and the a posteriori update equations are

$$K_k = M_k H^T (H M_k H^T + V)^{-1}$$

$$\hat{\delta}x(k) = \bar{\delta}x(k) + M_k H^T V^{-1}(z_k - H\bar{x}(k))$$
$$P_k = (I - K_k H) M_k (I - K_k H)^T + K_k V K_k^T$$

An important quantity in this update is the residual, or the difference between our sensor estimate and propagated conditional mean

$$z_k - H\bar{x}_k$$

This quantity is used to adjust our a priori propagated conditional mean, $\bar{\delta}x_k$, and produce the new conditional mean, $\hat{\delta}x$

From the state estimate, we can reconstruct the estimated position and velocity at any time as

$$\hat{p}(t_j) = \hat{\delta}p(t_j) + p_c(t_j)$$

$$\hat{v}(t_j) = \hat{\delta}v(t_j) + v_c(t_j)$$

Throughout the duration of the Kalman Filter, p_c and v_c are computed via the Euler integration formulas in the accelerometer model.

4 Simulation Results and Discussion

4.1 Simulation Initialization

The Monte Carlo simulation consists of 10,000 realizations, each over a 30 second interval. Each realization's initial conditions are drawn from

$$x_0 \sim N(0m, 100m^2)$$

$$v_0 \sim N(100m/s, 1(m/s)^2)$$

$$b_a \sim N(0, 0.01(m/s^2)^2)$$

The true acceleration, velocity, and position are provided to the simulation as

$$a(t) = asin(\omega t) \quad m/s^2$$

$$v(t) = v(0) + \frac{a}{\omega} - \frac{a}{\omega}cos(\omega t)$$

$$p(t) = p(0) + (v(0) + \frac{a}{\omega})t - \frac{a}{\omega^2}sin(\omega t)$$

Sensor noise is added to the above true position, velocity, and acceleration by the Gaussian distributions previously defined for each sensor.

4.2 Analysis of One Realization

The results for one realization are plotted to analyze the effect of the Kalman Filter.

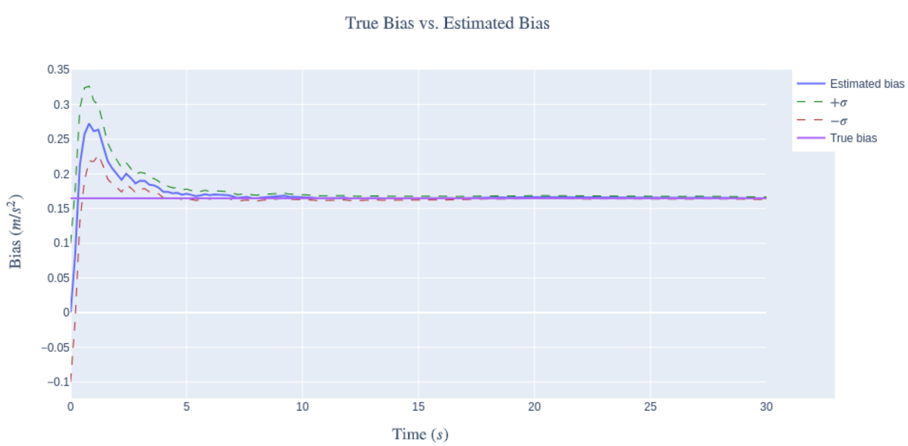

Figure 1: The estimated bias converges to the true bias for this realization.

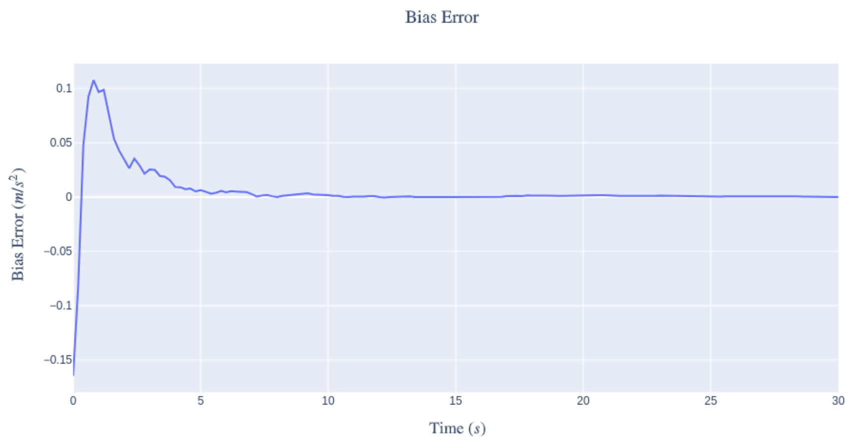

Figure 2: The error in the bias estimate when compared to the true bias converges to zero as time progresses.

Figure 1 illustrates the estimated bias as it converges to the true bias, and Figure 2 plots the error in the bias estimate over time. The one σ bounds are plotted to show that the covariance element P_{33} associated with the bias shrinks as time goes on. This is interpreted as a shrinking uncertainty in our bias estimate thanks to the GPS measurement used to calibrate the estimate. Indeed, the convergence to the true bias supports the claim that uncertainty shrinks as time progresses.

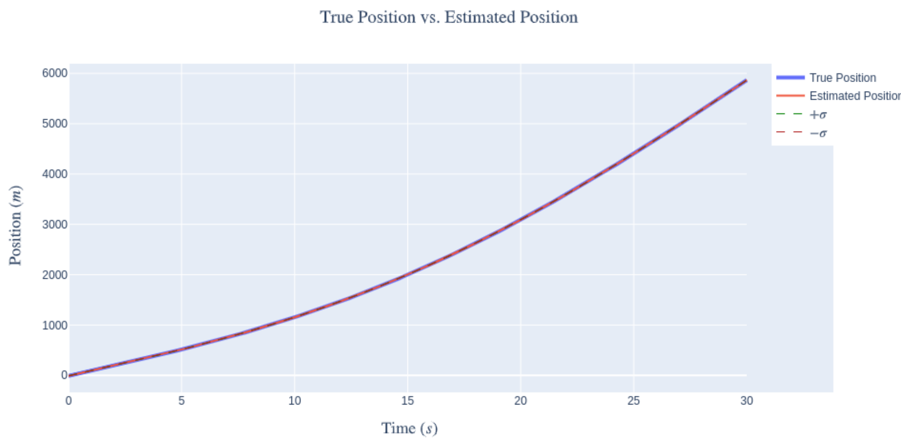

Figure 3: The true position and estimated position over time are nearly identical at this resolution.

Figure 3 shows that the estimated position tracks the true position very well, as the lines seem to overlap at this resolution. To get a better understanding of how well the estimated position tracks the real position over time, the estimate error is plotted in Figure 4. The error is largest over the initial ”warm-up” period of the filter, then reduces in magnitude as time progresses. Similarly, Figure 5 shows that the velocity estimate tracks the true velocity very closely, and Figure 6 gives a closer look at the performance. Starting after approximately 2 seconds and then continuing throughout the duration of this realization, the velocity estimate error remains close to zero.

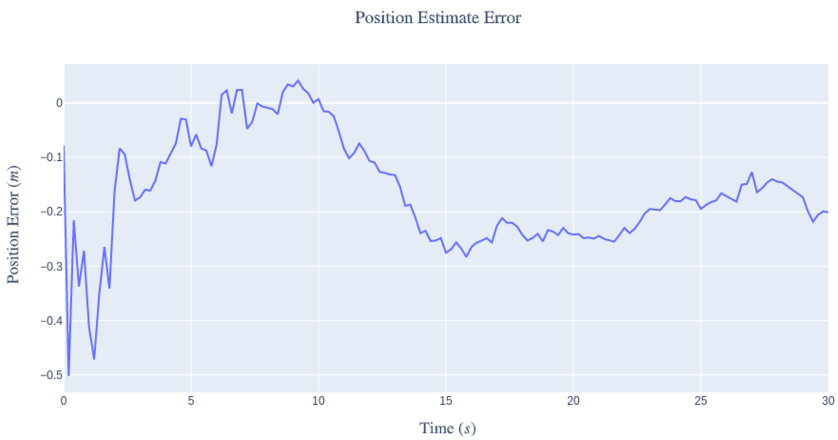

Figure 4: The error between the estimated position and true position over time reveals a closer look at the accuracy of our position estimate. The error decreases after the initial "warm-up" period of the filter, as expected.

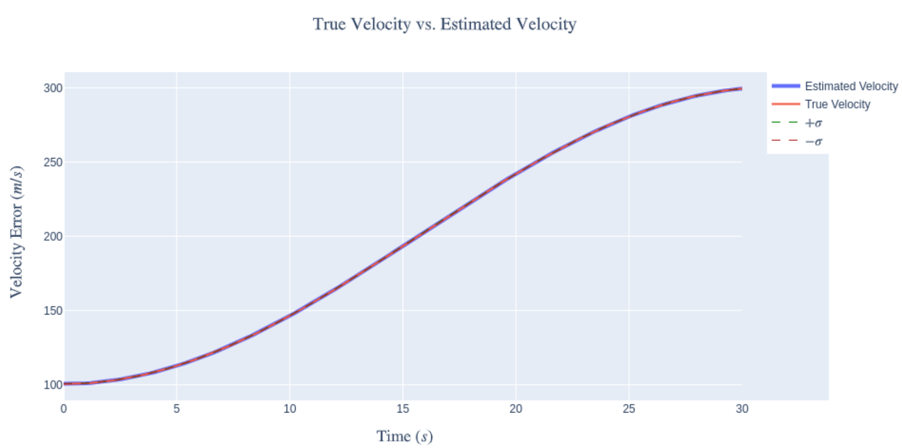

Figure 5: The true velocity and estimated velocity are nearly identical at this resolution.

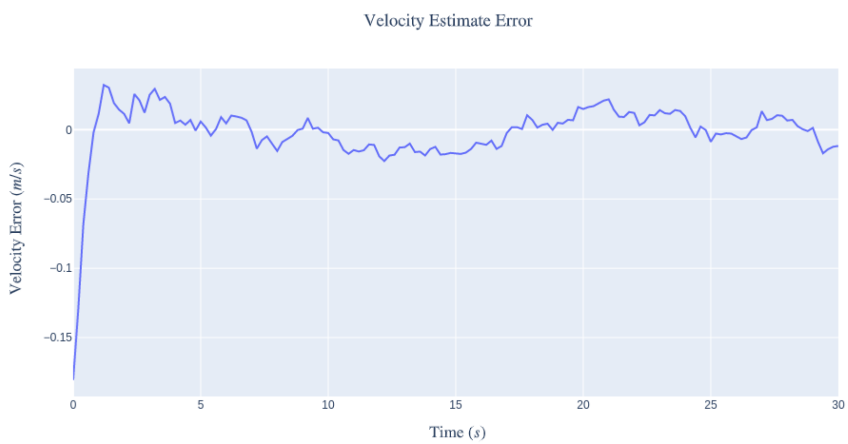

Figure 6: The velocity estimate error is plotted for a closer look at the performance of the filter in estimating the true velocity. The estimate remains close to zero for the duration of the filter after the "warm-up" period.

4.3 Analysis of the Monte Carlo Simulation

In order to properly assess the performance of the Kalman Filter, the difference between the filter's covariance matrix and the covariance matrix resulting from the 10,000 Monte Carlo realizations is compared and shown to converge. The covariance of the data is calculated by first computing the average error over time over all ensembles

$$e^{ave}(t_i) = \frac{1}{N_{ave}} \sum_{l=1}^{N_{ave}} e^l(t_i)$$

Next, the covariance of the Monte Carlo simulation is computed by

$$P^{ave}(t_i) = \frac{1}{N_{ave} - 1} \sum_{l=1}^{N_{ave}} [e^l(t_i) - e^{ave}(t_i)][e^l(t_i) - e^{ave}(t_i)]^T$$

Figure 7 illustrates the norm of the difference between the simulation data's covariance and the filter's covariance at each time index i, calculated in the Kalman Filter as P_i. The convergence of the two covariance matrices validates the performance of the Kalman Filter. In other words, the covariance calculated by the Kalman Filter reflects the actual covariance in the data.

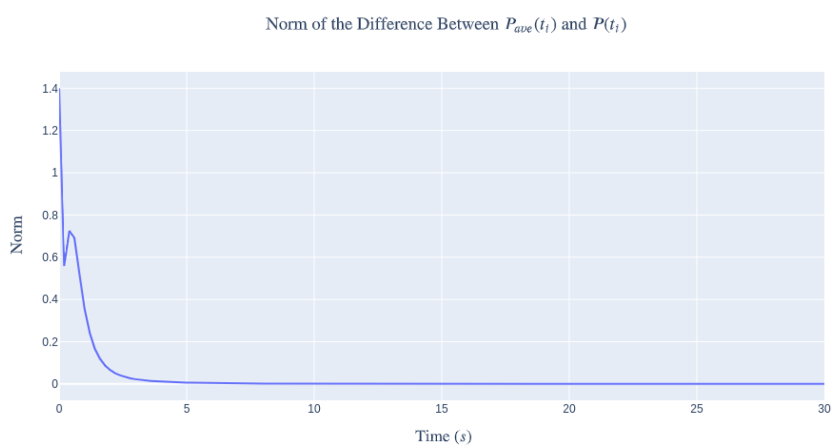

Figure 7: The magnitude of the difference between the covariance calculated by the Kalman Filter and the covariance of the Monte Carlo simulation data remains close to zero for the majority of the simulation duration.

As a further check on the performance of the filter, the orthogonality of the error in estimates is checked. The orthogonality of the error in estimates is checked by the average

$$\frac{1}{N_{ave}} \sum_{l=1}^{N_{ave}} [e^l(t_i) - e^{ave}(t_i)]\hat{x}(t_i)^T \approx 0 \; \forall \, t_i$$

Figure 8 illustrates that the norm of this orthogonality matrix goes to zero as time progresses.

One final check on the performance of the filter is the correlation of the residuals. The correlation of residuals should be approximately zero for all time steps, calculated as

$$\frac{1}{N_{ave}} \sum_{l=1}^{N_{ave}} r^l(t_i) r^l(t_m)^T \approx 0 \; \forall \, t_m < t_i$$

The correlation is only calculated for $t_m = 20.0s$ and $t_i = 20.2s$, with a value of 0.01143. This value satisfies the check on the correlation of the residuals.

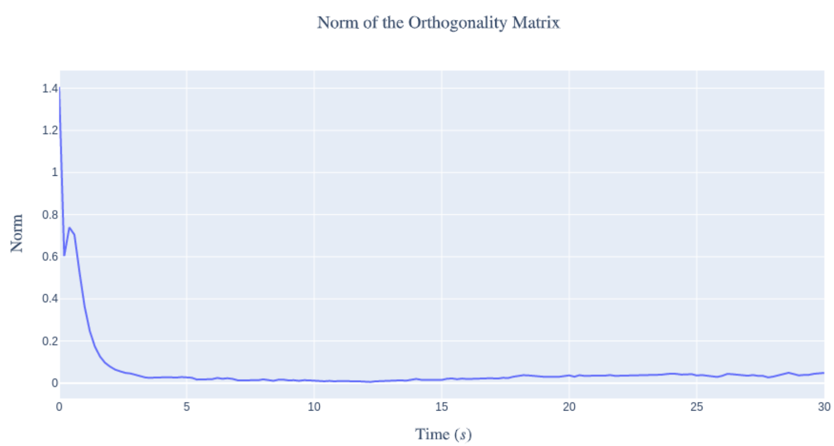

Figure 8: The magnitude of the orthogonality of the error in the estimates remains close to zero for the majority of the simulation duration.

5 Conclusion

The Kalman Filter is shown to accurately estimate the bias of the accelerometer as well as the position and velocity of the vehicle by using both the noisy accelerometer and noisy GPS measurements. The Monte Carlo simulation with 10,000 realizations verifies the performance of the Kalman Filter, and the filter is shown to enable accurate calibration of the accelerometer bias.