IMU/GPS Sensor Integration for Pedestrians in Urban Environments

Nicolas San Miguel, Alana Sanchez, and Jacob Beardslee, *Stanford University*

BIOGRAPHY

Nicolas San Miguel is a graduate student in the Aeronautics and Astronautics Department at Stanford University. He received his B.S. degree in Aerospace Engineering from the Georgia Institute of Technology in 2021.

Alana Sanchez is a graduate student in the Aeronautics and Astronautics Department at Stanford University. She received her B.S. degree in Physics from the Massachusetts Institute of Technology in 2021.

Jacob Beardslee is a graduate student in the Aeronautics and Astronautics Department at Stanford University. He received his B.S. degree in Aerospace Engineering from The Ohio State University in 2020.

ABSTRACT

Dense urban environments pose a challenge for the Global Positioning System (GPS) due to man-made structures, such as tall buildings, subways, and tunnels, that induce multipath and receiver errors making it hard to pinpoint a user's location. Integrated inertial measurement unit (IMU) and GPS systems offer a method for improved state estimation accuracy, while accounting for individual errors associated with each sensor. In this work, a methodology for IMU/GPS integration is proposed for pedestrians in urban environments, due to the lack of prior research focused on pedestrians. We implement an extended Kalman filter with IMU accelerometer and gyroscope data to improve state estimation accuracy from raw GPS information for users in San Francisco's financial district. We find the integration of IMU and GPS measurements improves the accuracy of a user's position from raw GPS data when compared to other filtering techniques and can easily be implemented with the average pedestrian in a city environment.

I. INTRODUCTION

Urban environments contain a multitude of obstacles that decrease the accuracy of the Global Positioning System's (GPS) positioning estimation for a user. Man-made and natural features, such as tall buildings, subways, and foliage, can reflect or block signals completely, decreasing the accuracy of GPS signals that reach the receiver [1]. These errors are amplified depending on the amount and size of obstacles with the potential to interfere with GPS information and are maximized in "urban canyons," where tall buildings prevent a direct path to the receiver and degrade GPS data to a sometimes unusable degree [2].

In order to mitigate some of the positioning errors induced from urban environments, sensor integration with GPS is a widely applied practice across many types of navigation [3, 4]. A commonly used sensor to help increase accuracy is an inertial measurement unit (IMU). IMUs utilize a combination of gyroscopes and accelerometers to calculate attitude and acceleration, which can be integrated to provide velocity and position information for a user. IMUs can be acquired for a low-cost and are more accessible than in previous decades, to the extent they are now generally included in the average smartphone. These IMUs benefit from a high update rate when compared to the GPS update rate, but they suffer when used as a single sensor due to errors in the sensor bias accumulating over time, leading to large drifts in the IMU positions [3].

Pedestrians in urban environments require precise positioning information to accurately navigate to their desired location. In dense urban environments, obscured GPS signals can cause errors in determining directionality, positioning, and a precise location. Due to the high prevalence of smartphones, with IMUs located within, IMU/GPS sensor fusion could a viable option to improve state estimation accuracy for a pedestrian. Past work has researched the combination of IMUs with pedestrians and shows a promising step forward for both general and urban navigation [5, 6].

II. RELATED WORK

The integration of IMU sensor information and GPS has a large base of past research for state estimation, especially in the case of land-based navigation [4, 7–9]. More recently, lower cost IMUs have been studied in relation to autonomous vehicles in urban environments to improve position estimation in areas riddled with multipath and receiver errors [10]. In general, these works seek to employ an extended Kalman filter that combines IMU sensor information to outperform normal methods of filtering. Additional methodologies have also been proposed to further refine the accuracy of the Kalman filter, including

introducing weighting and thresholding sensor data to remove residual errors from the sensors [3, 8]. In general, land-based vehicle navigation has been the focus of research for users in urban environments and points to the viability of sensor fusion techniques in improving state accuracy.

One study looked at how accurate a low cost micro-electrical mechanical system (MEMS) IMU can be when paired with a global navigation satellite system (GNSS) for ground vehicle navigation in an urban environment [11]. An additional MEMS IMU of superior quality was analyzed concurrently to serve as a comparison of performance in position and attitude. For static analysis, the IMU's were logged for 24 hours and an Allan variance technique was applied to the data to identify and quantify sources of noise for each sensor. A normality test was then run on the static data to verify if the inertial measurements follow a Gaussian Distribution. For kinematic analysis, the performance of three possible integrated navigation systems were analyzed: a low cost IMU with a lower quality GNSS, a low cost IMU with a higher quality GNSS, and a high cost IMU with a higher quality GNSS. Three different trajectories were analyzed: one with heavy leaf foliage and urban canyons, and two trajectories where loss of the GNSS signal was forced. They concluded that the low cost IMU can be effectively used in mass market ground navigation systems when coupled to a mid-range GNSS sensor for applications that can tolerate errors of about 2m for 3D positions and 3 degrees for heading angle.

Past studies of pedestrian IMU integration have been conducted in areas with and without obstructive objects [5, 6]. One study looked at a foot-mounted IMU to measure the acceleration and angular rate of a pedestrian walking in an open sky environment without multipath effects [5]. This study implemented an extended Kalman filter and error reduction by utilizing a zero velocity update condition based on the foot operating at zero velocity when it is placed on the ground while walking. Another study looked at general IMU/GPS integration with an extended Kalman filter but also included a correction based on the stride length of the pedestrian, which affects the bounce of the hips while walking and induces additional accelerations for each step [6]. Both studies found IMU/GPS integration with an extended Kalman filter to increase the accuracy of position estimations for pedestrians in both open sky and urban environments.

This work seeks to combine an extended Kalman filter with raw GPS and IMU sensor data for pedestrians in the financial district around San Francisco, an area with an abundance of urban canyons. We simplify the pedestrian model by taking experimental data walking with a constant velocity, with no purposeful accelerations or decelerations. This simplification is in contrast past work that models the pedestrian based on the motion of the foot [5] and accounts for acceleration contributions from hip variations while walking [6]. We seek to understand the accuracy of this simplified pedestrian model and it's state estimation in an urban environment when compared to similar urban studies with land-vehicle navigation.

III. PROBLEM STATEMENT

After collecting concurrent IMU and GPS data from a pedestrian, we will apply different filtering methods to generate a more accurate path than achieved with the GPS data alone, especially in urban environments.

1. Mathematical Formulation

a) Complementary Filter

The design of the complementary filter is fairly straightforward and consists of a weighted average of estimated future states. Beginning at a state μ_t, we seek to estimate the next state, $\hat{\mu}_{t+1}$, given the true next state is μ_{t+1} as closely possible. To do this, we estimate the next state independently using each measurement, so the next state according to only the IMU data, $\tilde{\mu}_{IMU}$, and the next state according to only the GPS measurements, $\tilde{\mu}_{GPS}$. Then, a weighted average between these two states is taken as the predicted next state. The update is then calculated as $\hat{\mu}_{t+1} := \mu_t + \Delta t(w_{IMU}\tilde{\mu}_{IMU} + (1 - w_{IMU})\tilde{\mu}_{GPS})$ where w_{IMU} is the weight assigned to the IMU measurements. A visual description of this algorithm is provided in the figure below.

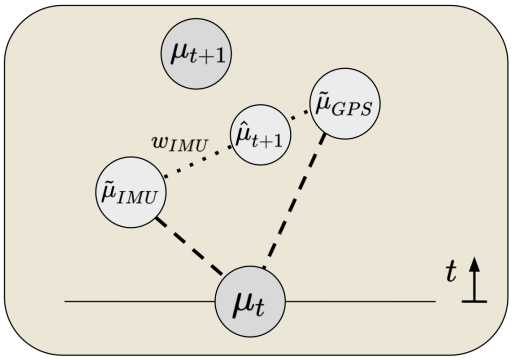


Figure 1: Complementary filter

An interesting extension of this would be to change weights over time depending on feedback from the system; however, this is beyond the scope of this investigation.

b) Kalman Filter

Our original filter had intended to describe the motion of a pedestrian using linear dynamics and linear measurements that allow for a basic Kalman Filter to work over the system. This doesn't consider biases and noise that are considered in the extended Kalman filter later on.

In this case, the state vector uses the position and velocity information as well as the gyroscope data and ϕ, θ, and β are the angles about the x-, y-, and z-axes of the IMU, respectively.

$$\mu_{9\times1} = [x\ y\ z\ \dot{x}\ \dot{y}\ \dot{z}\ \phi\ \theta\ \beta]^\intercal$$

The measurement matrix, which we take as linear, consists of the measurements recorded by the GPS. This generates a linear measurement matrix produced below.

$$z_{3\times1} = \begin{bmatrix} x_{GPS} \\ y_{GPS} \\ z_{GPS} \end{bmatrix} \qquad H_{3\times9} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

The IMU is taken as the process, to generate the process noise matrix, we begin with the measured accelerometer variances as the first three diagonal elements, the measured gyro variances as the last three diagonal elements. The three diagonal elements in the center correspond with the uncertainties of the velocity measurements, which are integrated from the accelerometer data. We still presume the velocity variances to be Gaussian because they came from the variances of the accelerometer measurements, but a coefficient was applied to these three terms to tune the filter by hand after it was working.

$$Q_{9\times9} = diag(\sigma^2_{x,IMU}, \sigma^2_{y,IMU}, \sigma^2_{z,IMU}, \sigma^2_{x,v}, \sigma^2_{y,v}, \sigma^2_{z,v}, \sigma^2_{\phi,IMU}, \sigma^2_{\theta,IMU}, \sigma^2_{\beta,IMU})$$

The measurement noise is much more straightforward. Since we aren't solving for the GPS positions, the variances are given by the computed values in the stationary test. There were smaller variance values that computed in GNSSLogger, but we use our own measured variance to be more affording.

$$R_{3\times3} = \begin{bmatrix} \sigma^2_{x,GPS} & 0 & 0 \\ 0 & \sigma^2_{y,GPS} & 0 \\ 0 & 0 & \sigma^2_{z,GPS} \end{bmatrix}$$

Our state transition model consists of a simple 3D force model of a pedestrian, assuming the accelerometer measurements have already been converted to the Earth-Centered, Earth-Fixed (ECEF) reference frame. The equations describing this transition are presented below and the state transition matrix is also presented. The assumption of a priori body-to-inertial frame rotation is dropped for the extended Kalman filter (EKF).

$$
\begin{aligned}
x_{t+1} &= x_t + \dot{x}_t \Delta t \\
\dot{x}_{t+1} &= \dot{x}_t \\
y_{t+1} &= y_t + \dot{y}_t \Delta t \\
\dot{y}_{t+1} &= \dot{y}_t \\
z_{t+1} &= z_t + \dot{z}_t \Delta t \\
\dot{z}_{t+1} &= \dot{z}_t
\end{aligned}
\qquad
F = \begin{bmatrix}
1 & \Delta t & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & \Delta t & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & \Delta t & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix} x \\ y \\ z \\ \dot{x} \\ \dot{y} \\ \dot{z} \\ \phi \\ \theta \\ \beta \end{bmatrix}
$$

c) Extended Kalman Filter

The extended Kalman Filter is developed to consider the biases and noise that were omitted from our previous implementation of a linear Kalman Filter. First, we describe the state as a 15-component vector to include the 3D position, velocity, rotation, and the biases for the accelerometer and gyroscope in each axis. The state matrix is provided below.

$$\mu_{15\times 1} = [x\ y\ z\ \dot{x}\ \dot{y}\ \dot{z}\ \phi\ \theta\ \beta\ b_x\ b_y\ b_z\ b_\phi\ b_\theta\ b_\psi]^\intercal$$

Here, we describe the GPS as the measurements and the IMU as the process, so the measurement vector is simply the GPS measurements, output in the ECEF frame by the smartphone, and the measurement matrix is a linear 3×15 matrix since we are not solving for the positions using raw satellite data. Both matrices are provided below.

$$z_{3\times 1} = \begin{bmatrix} x_{GPS} \\ y_{GPS} \\ z_{GPS} \end{bmatrix} \qquad H_{3\times 15} = \begin{bmatrix} 1\,0\,0\,0\ldots\,0 \\ 0\,1\,0\,0\ldots\,0 \\ 0\,0\,1\,0\ldots\,0 \end{bmatrix}$$

The measurement noise is characterized by the stationary analysis conducted in the previous section and the resulting variances are used in the R matrix below.

$$R_{3\times 3} = \begin{bmatrix} \sigma^2_{x,GPS} & 0 & 0 \\ 0 & \sigma^2_{y,GPS} & 0 \\ 0 & 0 & \sigma^2_{z,GPS} \end{bmatrix}$$

Next, we take our state transition model, which is nonlinear in this case, and produce our state-transition matrix, F. What makes this model different from the basic Kalman filter is that this one consists of the rotation matrix R applied to the measured acceleration values, and this R is inherently nonlinear as it converts the body frame of the IMU to the ECEF inertial frame. Additionally, the measured acceleration consists of not only the true acceleration of the IMU, but also the bias and noise terms enveloped within that measurement. The equations specifying the state transition are provided below and the F matrix as well. Note in the F matrix, the values X are placeholders for the time derivative of the rotation matrix, R. The Jacobian is taken in the code and is not included here for space. Note that the derivative of the gyroscope bias is assumed to be constant here because not much drift in the gyroscope angle was observed over the relatively short duration of these experiments, which lasted only a few minutes.

$$
\begin{aligned}
\overrightarrow{P}_{t+1} &= \overrightarrow{P_t} + \overrightarrow{v_t}\Delta t \\
\overrightarrow{v}_{t+1} &= \overrightarrow{v_t} + \mathbf{R}\overrightarrow{a_t}\Delta t \\
&= \overrightarrow{v_t} + \mathbf{R}(\overrightarrow{a_t} - \overrightarrow{b_t} - \overrightarrow{w_t})\Delta t \\
\overrightarrow{\alpha}_{t+1} &= \overrightarrow{\alpha_t} \\
\overrightarrow{b}_{t+1} &= \overrightarrow{b_t}
\end{aligned}
\qquad
F = \begin{bmatrix}
1 & 0 & 0 & \Delta t & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & \Delta t & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & \Delta t & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & X & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & X & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & X & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}
$$

Perhaps one of the most important metrics is the Q matrix which describes the process noise. The process is defined as the IMU in this setup, and the variance in IMU measurements set in this matrix are taken from our stationary analysis, which is discussed later in the paper. Note while the biases are analyzed and found to be consistent with the technical requirements for the IMU we purchased, the technical specification values for the accelerometer and gyroscope are used, and these are assumed to be isotropic.

Source	x-direction	y-direction	z-direction
IMU variance (accel.) $[\text{m/s}^2]^2$	0.5841^2	0.4789^2	0.6951^2
IMU variance (gyro.) $[^\circ/hr]^2$	0.0322^2	0.0304^2	0.0379^2
IMU bias (accel.) $[\text{m/s}^2]^2$	0.001	0.001	0.001
IMU bias (gyro.) $[^\circ/hr]^2$	0.00001	0.00001	0.00001

IV. APPROACH

The code used to process/clean data, run the stationary analyses, and process the Complementary and Kalman filters are included in the GitHub repository here [12].

1. Proposed Algorithm

The proposed experiments and analyses consist of first conducting stationary tests of the IMU and GPS measurements. These data are analyzed to develop estimated on biases and noise to be used in later filtering techniques. The experiments mentioned here are elaborated upon in the next section. In short, we then collect walking data from an open-sky area on Stanford's campus and use this as a baseline where GPS measurements are relatively reliable. Finally, we collect data from walks taken in San Francisco's Financial District, where there are numerous skyscrapers obstructing the view of the sky and much less reliable GPS measurements. In terms of data processing, we develop three filtering techniques: a Complementary filter, a basic Kalman filter, and an Extended Kalman Filter.

2. IMU Setup

Obtaining IMU data for the trajectories was a bit more involved compared to the GPS data. First, an IMU was purchased online. The MPU-6050 was chosen based on positive reviews as well as it's low cost. The IMU was connected to an Arduino Uno using a breadboard and jumper wires with instructions found online [13]. The IMU/Arduino setup can be seen below.

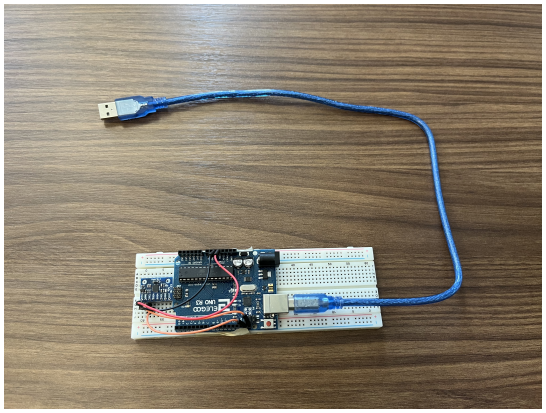


Figure 2: IMU/Arduino Setup

3. Stationary Analysis

To later implement filtering techniques, we need to develop an understanding of sensor errors for both the accelerometer and the gyroscope within the IMU. The expected factors that would contribute to the accelerometer error would be gravitational acceleration and the Coriolis acceleration while the gyroscope error would ideally consist of only the rotation rate of the Earth. In addition to these biases, random noise affects these measurements, especially for the low-cost IMU used in this experiment.

Stationary data was recorded using the IMU for three minutes and was normalized according to the product specifications to convert from the measured voltages to acceleration values in m/s^2. The recorded accelerometer values for each coordinate axis were plotted on a histogram and are included in the figure below. Next, a normal distribution was fit to characterize the bias and variation in each direction. Though no formal hypothesis testing was conducted to quantify how close to Gaussian the data actually is, through visual inspection we observe that the normal distributions appear to encapsulate the histograms well.

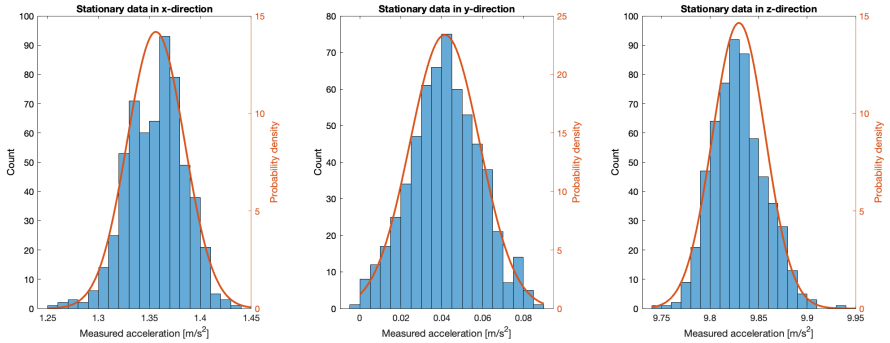
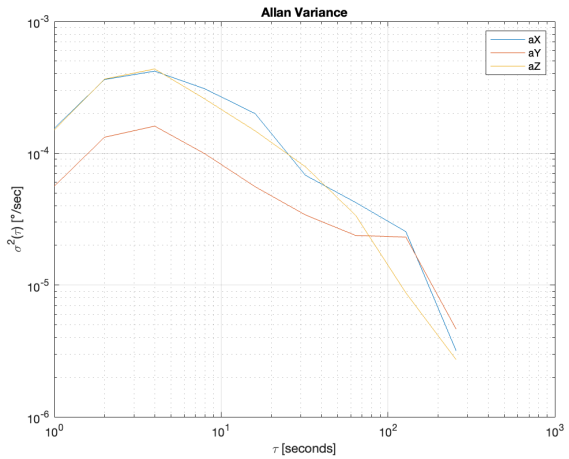
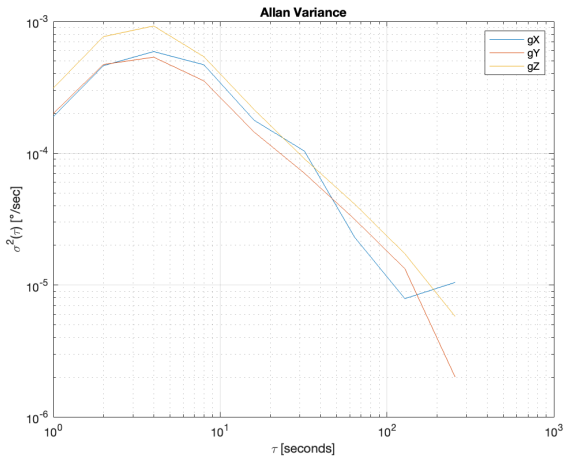


Figure 3: Histogram of accelerometer measurements along each axis

A common way to analyze the stability of a sensor is using Allan variance [14]. This metric describes noise and oscillator drift for a sensor in stationary conditions and is frequently used to characterize IMUs. The metric itself is computed as the standard deviation over a cluster of samples, but in this report is computed using built-in MATLAB functions. The resulting values are plotted below for the accelerometer and the gyroscope.



(a) Allan variance for accelerometer measurements



(b) Allan variance for gyroscope measurements

Figure 4: Allan Variance for IMU

The interpretation of these graphs is developed extensively in literature, but using the information pertinent to our results, an initial downward slope of -1/2 on the log-log plot indicates only white noise is present. While this slope is observed in our plots, a slight upward slope is seen at the beginning of the data, indicating that there is slight bias in the sensors. Angle drift in the gyroscope would be identified by an increase in the variance as time progresses. This too is not observed significantly as the variance is observed to decrease but not increase again. Part of the reason no large angle drift is seen is because the stationary tests were only taken over a few minutes. If the results were extended over longer time series, there would likely be more drift. This informs our assumption that the derivative of the gyro bias is constant.

4. Design Decisions

a) Experimental Design

The IMU setup was very simple and includes a small test bed the size of a phone. Rather than construct a power unit and set up scripts to record memory, the Arduino and IMU were directly connected to a laptop by powered USB. This reduced the experimental complexity and allowed for more flexibility when troubleshooting the setup. The IMU was held level and directly in front of the user and oriented in the direction of motion for each test. This allows for consistency across each run and straightforward expectations for the resulting data.

More details regarding the selection of variances for the covariance matrix are discussed in the previous section. While some referenced projects consider the noise from an IMU sensor to be isotropic, based on our measurements, we decided not to make such an assumption.

b) Complementary Filter

We noticed that performance varies significantly when the relative weight of the IMU measurements is changed. When $w_{IMU} = 0.0$, then the complementary filter exactly tracks the GPS measurements and when $w_{IMU} = 1.0$, the complementary filter tracks the "random walk" of the IMU measurements. In the urban navigation examples presented, we varied w_{IMU} over 0 to 1 to identify a value that had the best performance relative to the other weights.

c) Kalman Filter

Our original implementation of a linear Kalman Filter was informed by our assumption that the state transition model is linear. Since covariance matrix design is important as well, we took the variances of the stationary data, which through our procedure included the bias, noise, and slight drift of the accelerometer and gyro measurements. We decided not to solve for the position solution using raw GNSS data to reduce the complexity of the problem both in the linear and extended Kalman filters. Instead, we use the final output position of the GNSSLogger app for our GPS measurements, which provides us with a linear measurement matrix and straightforward measurement noise values.

d) Extended Kalman Filter

Our final implementation of an extended Kalman filter considers the nonlinear effects of the state transition model and implements the biases for the accelerometer and gyroscope across each axis. A diagram for this extended Kalman filter is included below and highlights the two sensor inputs, both GPS and IMU, into the system to output updated position estimation information. We follow the same extended Kalman filter approach as described in the previous section, including the updates to the F matrix.

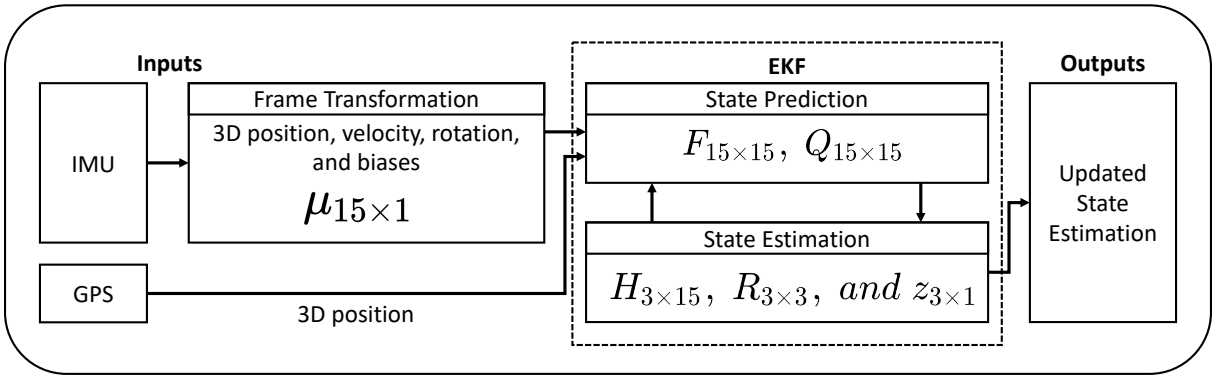


Figure 5: Extended Kalman Filter Diagram

V. RESULTS

1. Experimental Procedure

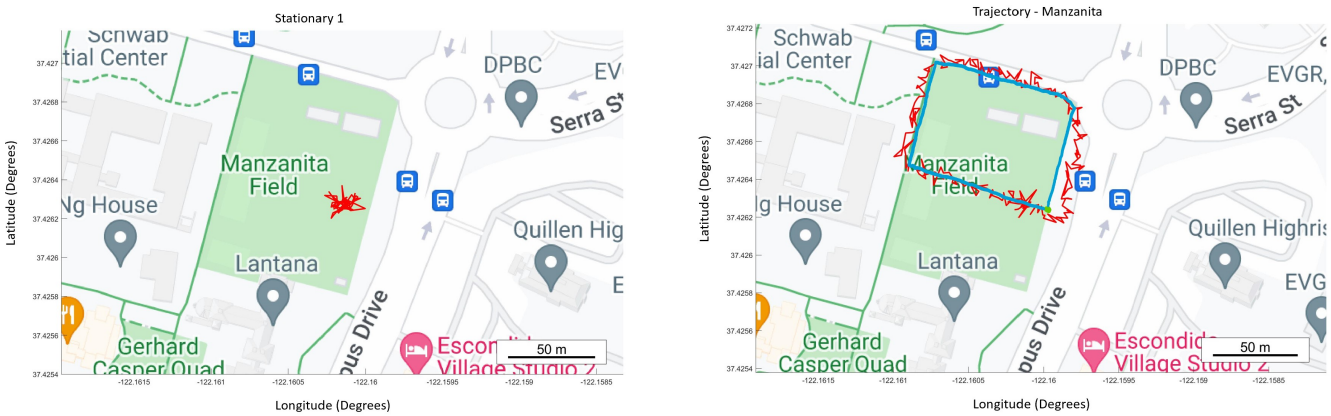
For each trajectory, the GPS and IMU devices were held adjacent at the same height and orientation at all times. This was to ensure as similar of a path as possible to aid in the fusion of the sensors from each device. Extra care was taken to maintain a constant orientation for the IMU in particular to improve the accuracy of attitude measurements at each instance.

GPS trajectories were logged using the GNSS Logger App on the android phone provided. The trajectory data was then read into MATLAB using the recommended GPS Measurement Tools code available on GitHub. The data required for implementing the Kalman Filter was extracted and output to a separate CSV file. This include the UTC time, time increment between measurements, and user's position in latitude, longitude, and altitude as well as in ECEF coordinates.

For each run, the IMU sensor data was written into a CSV file using the Arduino application [15]. The X/Y/Z accelerations were extracted from the accelerometer data and the roll/pitch/yaw axis angles were extracted from the gyroscope data. The accelerations were then integrated to obtain velocity data. An offset velocity of zero m/s was assumed for the integration.

2. Baseline

Stationary data, as well as a simple trajectory with optimal conditions, were obtained in an open sky environment on Stanford campus. For the stationary data, data was logged for at least one minute while not moving the IMU or android phone. The variation in velocity and position provided the standard deviations used within the process noise matrix discussed above. A simple rectangular path walking around Manzanita Field was used to ascertain how well the GPS performed under ideal conditions with no aid from the IMU. The results for the static test and simple trajectory can be seen below.



(a) Stationary

(b) Walking

Figure 6: Raw Results for Optimal Conditions

The red line corresponds to the raw data measurements while the blue line represents the true trajectory in the walking figure. The GPS does well to follow the true trajectory when no obstacles are blocking the signals to the user.

Now that results were obtained under optimal conditions, trajectory data within a dense urban environment was collected. All urban data was obtained in San Francisco's financial distract. Simple walking paths with normal patterns of movement (straight, constant speed) were maintained for each trajectory. For each run, a Strava run was also logged to serve as a comparison. The results for one of the logged trajectories can be seen below.

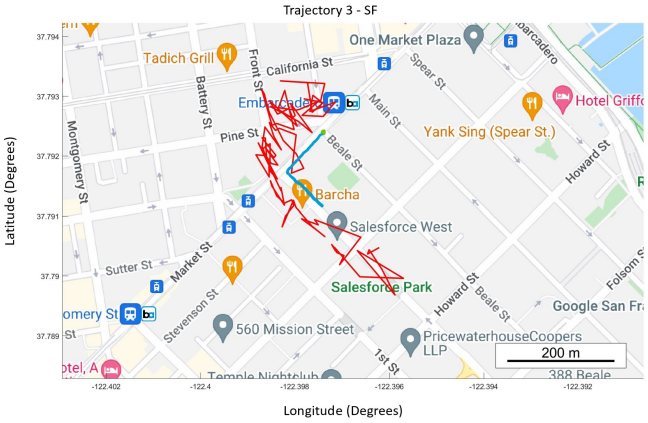


Figure 7: Raw Results for Urban Environment

The GPS performs much worse without optimal conditions and records the receiver's position to be several blocks away from their true position at some instances during the trajectory. This reinforces the importance of an additional localization method to help provide a more accurate position of the receiver.

3. Quantitative Analysis

To develop a quantitative metric to represent the error of a path, a true path is first identified. The true path is plotted by hand using Google Maps after each experiment. Waypoints along each path were identified based on where we know we walked, and the coordinates of these points are imported into MATLAB. Hundreds of intermediate points are linearly interpolated between each set of waypoints to create a fine vector of true position points with a resolution approximately an order of magnitude greater than the measured values. In other words, for each GPS point, there were about 10 true position points along the true path over that 1 second of travel. A function is written to compute the distance between a measured position point and the nearest true position point, and this error is calculated over the course of the Manzanita Walk Test. The errors of the filtering techniques included too.

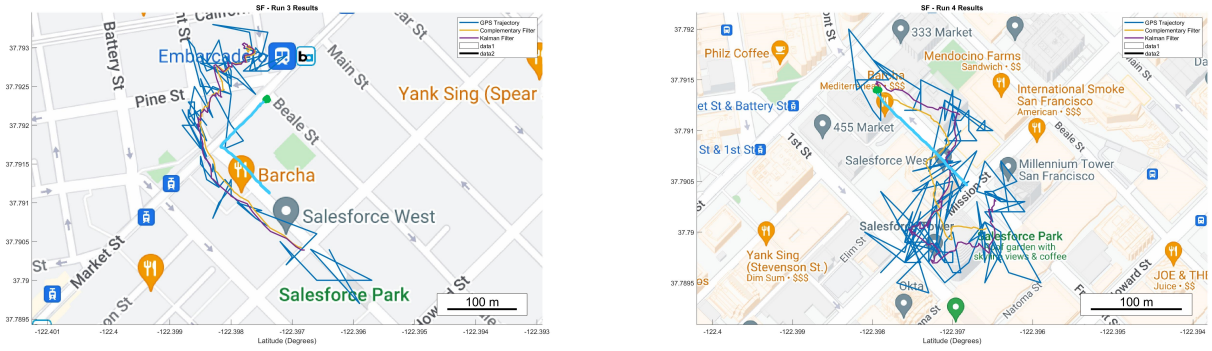
Figure 8: Error over test for the Manzanita Walk Test

The root mean squared error is calculated for each series, and the results are presented in the following table. The filtered methods improve upon the raw data error, and in this case the complementary filter outperforms the Kalman filter.

	Raw GPS measurements	Complementary Filter	Kalman Filter
RMSE	1.519	1.4742	1.495

4. Qualitative Analysis

Performance results for the initial Kalman filter and complementary filter in an urban environment can be seen in Figure 7 below.



(a) **(b)**

Figure 9: Filtered Results for Urban Environment

For the trajectory shown in Figure 9(a), the filters still struggle to accurately track the receiver and are drifting to the right of the receiver's true path due to the GPS data. The filter's may have had better estimated paths if the GPS location for the starting position of the receiver was more accurate. The drifted starting position of the GPS is where the filters assume the true starting position is, thus creating an offset that carries throughout the entire path for the filter's estimated position. Both filters perform relatively equal for this trajectory.

For the trajectory shown in Figure 9(b), the GPS provides an accurate starting position for the filters to work with. As seen, both filters perform relatively well for the first half of the path with the complementary filter performing better than the Kalman filter. For the latter portion of the trajectory though, the user is walking under a ledge which causes significant errors to the GPS location. Both filters are unable to ignore all of the error skewed GPS data, which in turn causes large errors in their predicted receiver position.

VI. CONCLUSIONS

We designed an extended Kalman filter with IMU sensor integration to improve the state estimation accuracy for a pedestrian in an urban environment with incomplete information from GPS. We implemented the linear and extended Kalman filter with raw GPS data and IMU sensor information for a series of on-street scenarios in a dense urban environment in the San Francisco area. We demonstrated that sensor integration with both a complementary filter and Kalman filter improves the accuracy of state estimation with a quantitative root mean square error analysis. However, despite the implementation of both a linear and extended Kalman filter with IMU state information and the improvement of our state estimation, we were unable to accurately track the receiver location across all city runs. With additional information to define the initial state position of the receiver, IMU/GPS sensor fusion with an extended Kalman filter offers a feasible solution to improve state estimation accuracy for pedestrians in dense urban environments.

VII. FUTURE DIRECTIONS

Future work for this project focuses on correcting the drift of the starting position and subsequent GPS information and further filtering data to remove extreme outliers. Kalman filtering techniques are unable to correct for these positional errors without additional information so defining a precise starting position is essential. As the extended Kalman filter did not produce noticeable improvements upon the initial linear Kalman filter, a different Kalman filtering process could be implemented to see if it improves state estimation accuracy. Past work shows the benefits of a sigma Kalman filter for improved accuracy over an extended Kalman filter for a similar procedure that utilizes IMU sensor fusion [10]. Our work also didn't account for variations in acceleration calculated with the IMU due to the user walking, introducing hip bounce and variations in stride length. This acceleration variation has been shown to affect precision of acceleration values [6], and should be accounted for in future work.

ACKNOWLEDGEMENTS

We would like to thank Tara Mina for her insight and Professor Grace Gao for her feedback over the length of this project.

VIII. CONTRIBUTIONS OF TEAM MEMBERS

1. Nicolas: IMU build, static/baseline data collection, EKF design and implementation
2. Alana: Literature Review, city data collection, EKF diagram, and future directions
3. Jacob: IMU build, literature review, city data collection, figure generation

REFERENCES

[1] E. D. Martí, D. Martín, J. García, A. de la Escalera, J. M. Molina, and J. M. Armingol, "Context-aided sensor fusion for enhanced urban navigation," *Sensors (Switzerland)*, vol. 12, pp. 16 802–16 837, 12 2012.

[2] R. Sun, Y. Yang, K.-W. Chiang, T.-T. Duong, K.-Y. Lin, and G.-J. Tsai, "Robust imu/gps/vo integration for vehicle navigation in gnss degraded urban areas," *IEEE Sensors Journal*, vol. 20, pp. 10 110–10 122, 9 2020. [Online]. Available: https://ieeexplore.ieee.org/document/9075286/

[3] S. Sukkarieh, E. M. Nebot, and H. F. Durrant-Whyte, "A high integrity imu/gps navigation loop for autonomous land vehicle applications," *IEEE Transactions on Robotics and Automation*, vol. 15, pp. 572–578, 1999.

[4] M. G. Petovello, "Real-time integration of a tactical-grade imu and gps for high-accuracy positioning and navigation," 2003.

[5] L. Chen and H. Hu, "Imu/gps based pedestrian localization." IEEE, 9 2012, pp. 23–28. [Online]. Available: http://ieeexplore.ieee.org/document/6375373/

[6] K. Bikonis and J. Demkowicz, "Data integration from gps and inertial navigation systems for pedestrians in urban area," *TransNav, the International Journal on Marine Navigation and Safety of Sea Transportation*, vol. 7, pp. 401–406, 9 2013.

[7] L. Qingqing, J. P. Queralta, T. N. Gia, Z. Zou, and T. Westerlund, "Multi sensor fusion for navigation and mapping in autonomous vehicles: Accurate localization in urban environments," 3 2021. [Online]. Available: http://arxiv.org/abs/2103.13719http://dx.doi.org/10.1142/S2301385020500168

[8] F. Caron, E. Duflos, D. Pomorski, and P. Vanheeghe, "Gps/imu data fusion using multisensor kalman filtering: Introduction of contextual aspects," *Information Fusion*, vol. 7, pp. 221–230, 6 2006.

[9] Y. Zhao, *GPS/IMU integrated system for land vehicle navigation based on MEMS*. Architecture and the Built Environment, Royal Institute of Technology (KTH), 2011.

[10] P. Zhang, J. Gu, E. Milios, and P. Huynh, "Navigation with imu/gps/digital compass with unscented kalman filter," in *IEEE International Conference Mechatronics and Automation, 2005*, vol. 3, 2005, pp. 1497–1502 Vol. 3.

[11] R. Gonzalez and P. Dabove, "Performance assessment of an ultra low-cost inertial measurement unit for ground vehicle navigation," *Sensors (Switzerland)*, vol. 19, 9 2019.

[12] N. San Miguel, "Gps/imu sensor integration," https://github.com/NicolasSanMiguel/gps-imu-sensor-integration, 2022.

[13] M. Schoeffler. (2017) Tutorial: How to use the gy-521 module (mpu-6050 breakout board) with the arduino uno. [Online]. Available: https://mschoeffler.com/2017/10/05/tutorial-how-to-use-the-gy-521-module-mpu-6050-breakout-board-with-the-arduino-uno

[14] N. El-Sheimy, H. Hou, and X. Niu, "Analysis and modeling of inertial sensors using allan variance," *Instrumentation and Measurement, IEEE Transactions on*, vol. 57, pp. 140 – 149, 02 2008.

[15] G. Lambert. (2021) How to write arduino sensor data to a csv file on a computer. [Online]. Available: https://www.circuitbasics.com/logging-arduino-data-to-files-on-a-computer