

GPS-Aided 2D Inertial Navigation System for Formula SAE Car

Elio Saade

School of Electrical, Computer and Energy Engineering

Arizona State University

Tempe, USA

esaade@asu.edu

Abstract—Accurate position and attitude estimation is essential for many automotive, aerospace, and maritime applications. This project presents a GPS-aided Inertial Navigation System for the Formula SAE car at Arizona State University. A discrete-time kinematics model is formulated under a planar motion assumption, and the vehicle sensor suite, comprising a 9-DOF IMU and a GPS module, is described. The estimation algorithm uses a bank of four Kalman filters: three Linear Kalman Filters for high-rate IMU-driven dead-reckoning of the heading, North, and East states, and one Extended Kalman Filter for low-rate GPS corrections. The architecture is designed to handle asynchronous IMU and GPS measurements natively. The system is evaluated on a dataset from a simulated ground vehicle, achieving a heading RMSE of 0.94° , position RMSEs of 0.78 m and 1.2 m in the North and East directions, and a ground speed RMSE of 0.09 m/s. An ablation study confirms that GPS fusion is essential for bounding long-term position drift. The proposed framework provides a foundation for robust state estimation in Formula SAE applications, with future work focused on real-world validation and extension to full 3D navigation.

Index Terms—Position, Attitude, Inertial Navigation System (INS), Global Positioning System (GPS), Extended Kalman Filter (EKF), Formula SAE

I. INTRODUCTION

The navigation system is a crucial component in many automotive, aerospace, and maritime applications. It aims to provide position and attitude estimation of the vehicle from a suite of sensors. Guidance and control systems rely heavily on these estimates, hence the need for robust navigation systems that tolerate sensor noise, bias, and potential failures. A common approach is to combine Inertial Navigation System (INS) information and Global Positioning System (GPS) information in order to improve the vehicle position and attitude estimate. This is known as a GNSS/INS system [1]. The INS sensor data, including accelerometer, gyroscope and magnetometer, are used to calculate states through dead reckoning, which are then updated with GPS measurements using sensor fusion methods like an Extended Kalman Filter (EKF) [1].

This project focuses on developing a GNSS/INS system for the Formula SAE (FSAE) car at Arizona State University. State estimation is important and is needed in FSAE to accurately track the car on the racetrack and better understand its characteristics, which aids vehicle and driver development. The car is equipped with a 9-DOF Inertial Measurement

Unit (IMU) and a GPS module. The data-driven estimation from the available sensors presents some challenges. First, the IMU sensors are highly susceptible to noise and suffer from non-constant bias, which leads to noisy estimates and integration drift. Second, the GPS measurements are typically accurate to a few meters and they are obtained at a lower rate compared to the IMU. Consequently, fast movement can introduce lag and position errors.

The rest of the document is organized as follows: section II introduces the kinematics model, section III describes the sensor suite and data, section IV presents the methodology, section V discusses the results, section VI presents an ablation study, section VII concludes the work, and section VIII discusses the limitations and future work.

II. SYSTEM DESCRIPTION

A. Reference Frames and Conventions

Reference frames and sign conventions must be specified because the IMU sensor measurements are relative to the car, while the GPS measurements are taken in a fixed frame. As such, two reference frames are defined:

- **Body frame:** this is the frame that is attached to the car. It follows the Forward-Left-Up convention, meaning that the x-axis always points to the forward direction of the car, the y-axis points to the left side, and the z-axis points up.
- **Navigation frame:** this is the global fixed reference frame. It follows the North-East-Up (NEU) convention, meaning that the x-axis represents North, the y-axis represents East, and the z-axis represents Up.

The following is to be noted:

- Even though the Earth is curved, the NEU frame can be approximated and plotted as a rectangular surface because the typical racetrack distance (few hundred meters) is significantly small relative to the size of the Earth
- The ENU convention would make the plotting more intuitive (East on the x-axis) as compared to NEU (North on the x-axis). However, the NEU convention is chosen instead of ENU, because, in NEU, the heading angle of the vehicle (angle between x-axis and y-axis) is equal to 0 when the vehicle points North, which is consistent with the compass convention of True North.

B. Discrete-Time Kinematics Model

A kinematics model is used for this project. Kinematics describe the motion of bodies without considering the forces or mass that cause the motion; instead, they focus on measuring and calculating position, velocity and acceleration as a function of time [2]. A dynamics model of the car could be used for position and attitude estimation; however, it requires a significantly more complex model that includes throttle, steering, suspension, drivetrain and tires. Therefore, a kinematics model offers a good tradeoff between complexity and accuracy, and matches the available sensors well.

The main assumption for the project is that the vehicle only operates in a flat 2-D plane, meaning that it does not experience any roll or pitch movements. Based on that, the discrete-time kinematics model of the vehicle in the navigation frame is as follows:

$$x_N[k+1] = x_N[k] + v_N[k] \cdot dt + \frac{1}{2} a_N[k] \cdot dt^2 \quad (1)$$

$$v_N[k+1] = v_N[k] + a_N[k] \cdot dt \quad (2)$$

$$a_N[k+1] = a_N[k] \quad (3)$$

$$x_E[k+1] = x_E[k] + v_E[k] \cdot dt + \frac{1}{2} a_E[k] \cdot dt^2 \quad (4)$$

$$v_E[k+1] = v_E[k] + a_E[k] \cdot dt \quad (5)$$

$$a_E[k+1] = a_E[k] \quad (6)$$

$$\psi[k+1] = \psi[k] + (r[k] - b_r[k]) \cdot dt \quad (7)$$

$$b_r[k+1] = b_r[k] \quad (8)$$

where x_N, v_N, a_N denote the position, velocity and acceleration in the North direction in the navigation frame, x_E, v_E, a_E denote the position, velocity and acceleration in the East direction in the navigation frame, ψ denotes the orientation angle in the navigation frame, r denotes the yaw rate, and b_r denotes the yaw rate bias. The yaw rate is identical in the body and navigation frames since it is the rate of rotation around the z or Up axis. The orientation angle ψ is measured counterclockwise from North to East. SI units are used, that is, m for position, m/s for velocity, m/s^2 for acceleration, rad for angle, and rad/s for angular velocity.

C. Conversion Between Frames

Since the accelerometer measurements are taken in the body frame, they need to be converted to the navigation frame in order to be used in the kinematics model and estimation [3]. The Direction Cosine Matrix (DCM) is used:

$$a_n = C_b^n a_b$$

$$\begin{bmatrix} a_N \\ a_E \end{bmatrix} = \begin{bmatrix} \cos \psi & -\sin \psi \\ \sin \psi & \cos \psi \end{bmatrix} \begin{bmatrix} a_x \\ a_y \end{bmatrix} \quad (9)$$

The inverse transformation can be used to convert acceleration from the navigation frame to the body frame:

$$a_b = C_n^b a_n$$

$$\begin{bmatrix} a_x \\ a_y \end{bmatrix} = \begin{bmatrix} \cos \psi & \sin \psi \\ -\sin \psi & \cos \psi \end{bmatrix} \begin{bmatrix} a_N \\ a_E \end{bmatrix} \quad (10)$$

D. Sensors Modeling

The vehicle is equipped with a 9-DOF IMU and a GPS module. The IMU includes a 3-axis accelerometer, a 3-axis gyroscope and a 3-axis magnetometer. Based on the assumption of movement in 2-D, five measurements out of nine are needed from the IMU:

$$a_x^m = a_x + b_{ax} + n_{ax} \quad (11)$$

$$a_y^m = a_y + b_{ay} + n_{ay} \quad (12)$$

$$r^m = r + b_r + n_r \quad (13)$$

$$m_x^m = m_x + n_{mx} \quad (14)$$

$$m_y^m = m_y + n_{my} \quad (15)$$

where a_x^m and a_y^m are the measured longitudinal and lateral accelerations in the body frame, r^m is the measured yaw rate, m_x^m and m_y^m are the measured components of the magnetic field, b_{ax} , b_{ay} and b_r are the biases in the measurements, and n_{ax} , n_{ay} , n_r , n_{mx} and n_{my} correspond to the noise components. The accelerometer biases b_{ax} and b_{ay} are assumed to be zero, while the yaw rate bias b_r is unknown and is to be estimated online by the INS.

The GPS module measures longitude, latitude, altitude, and ground velocity. Longitude and latitude can be converted to x_N and x_E components using the *pymap3d* Python library and by considering the first measurement as the origin. Equation (16) links the ground velocity to the navigation frame components:

$$v_{gps} = \sqrt{v_N^2 + v_E^2} \quad (16)$$

III. DATA DESCRIPTION

There are issues preventing from collecting consistent sensor measurements from the real car on track. As a result, there is no real-world data to perform state estimation on. Therefore, a dataset by MathWorks is used instead, which includes all the required measurements [4]. The dataset comes from the MathWorks example **Estimate Position and Orientation of a Ground Vehicle**, which is an introductory example to the Navigation Toolbox and does not provide any details or code on how to design or implement a GNSS/INS. There are, however, some discrepancies between the dataset and the real operation of the car:

- 1) The dataset includes a vehicle traveling in a constant circular motion, while FSAE trajectories are usually much more dynamic.
- 2) In the dataset, the IMU is sampled at 100 Hz and the GPS is sampled at 5 Hz, while in the FSAE car, the IMU is sampled at 100 Hz, and the GPS at 40 Hz.

Therefore, the dataset is used as a starting point for the algorithm development and testing until real-world data is available. Appendix A presents the MathWorks dataset.

IV. METHODOLOGY

A. Asynchronous Updates

Typically, in sensor fusion applications, the measurements are sampled at the same frequency and are all used in one filtering step. However, in this project, the measurements are available at different rates; therefore, the filtering algorithm must be able to handle asynchronous updates. The frequency of the main inertial navigation system is then chosen to be the least common multiple of the different measurement frequencies: $f_{INS} = LCM(f_{IMU}, f_{GPS}) = LCM(100, 5) = 100 \text{ Hz}$. In other words, the system uses IMU measurements for state prediction and correction at every iteration, and performs additional correction whenever GPS measurements are available (every 20 iterations).

The main drivers of sensor fusion are the Linear and Extended Kalman filters. Appendix B covers the theory of the Linear Kalman filter, and Appendix C goes over the Extended Kalman Filter. Because the dynamics of the North components, East components, and yaw motion are decoupled and do not share inputs, the state estimations can be separated. Therefore, three Linear Kalman filters are derived to perform correction based on the IMU measurements, and one Extended Kalman Filter is derived for GPS updates. Table I presents the estimated states and required measurements for each filter.

TABLE I
KALMAN FILTERS SUMMARY

Index	Type	States	Measurements	Rate	Section
1	LKF	ψ, b_r	r^m, m_x^m, m_y^m	100Hz	IV-B
2	LKF	x_N, v_N, a_N	a_x^m, a_y^m	100Hz	IV-C
3	LKF	x_E, v_E, a_E	a_x^m, a_y^m	100Hz	IV-D
4	EKF	x_N, v_N, a_N x_E, v_E, a_E	lon, lat, v_{gps}	5Hz	IV-E

B. Filter 1: Orientation and Yaw Bias

Filter 1 is a Linear Kalman Filter that performs fusion of the yaw rate and magnetometer measurements to estimate the heading angle and yaw rate bias. The states are $[\psi, b_r]^T$. The yaw rate measurement is used as an input to the state equation ($u = r^m$), and the measurement vector includes the heading angle which is calculated from the measured magnetometer components following Eq. (17).

$$\psi^m = \arctan2\left(\frac{-m_y^m}{m_x^m}\right) - d \quad (17)$$

where d is the magnetic declination angle, that is, the angle between True North (geographic North Pole) and magnetic North (where a compass/magnetometer points). The correction of d is needed because the NEU reference frame uses True North, while the magnetometer points to magnetic North. The value of d depends on the location on Earth, and is obtained using [5] and GPS measurements.

The matrices of the LKF are derived from Eq. (7),(8).

$$A = \begin{bmatrix} 1 & -dt \\ 0 & 1 \end{bmatrix} \quad (18)$$

$$B = \begin{bmatrix} dt \\ 0 \end{bmatrix} \quad (19)$$

$$C = \begin{bmatrix} 1 & 0 \end{bmatrix} \quad (20)$$

The LKF parameters of Filter 1 are presented below:

$$P_1 = \begin{bmatrix} 0.1 & 0 \\ 0 & 10 \end{bmatrix} \quad (21)$$

$$Q_1 = \begin{bmatrix} 1 & 0 \\ 0 & 10^{-2} \end{bmatrix} \quad (22)$$

$$R_1 = [0.1] \quad (23)$$

C. Filter 2: North Components

Filter 2 is a Linear Kalman Filter that uses the IMU's accelerometer measurements to estimate the position, velocity and acceleration components in the North direction. Its states are $[x_N, v_N, a_N]^T$ and the measurement vector includes the calculated a_N^m from a_x^m and a_y^m following Eq. (9). The matrices are derived from Eq. (1),(2),(3).

$$A = \begin{bmatrix} 1 & dt & \frac{1}{2}dt^2 \\ 0 & 1 & dt \\ 0 & 0 & 1 \end{bmatrix} \quad (24)$$

$$C = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \quad (25)$$

Filter 2, in a sense, acts as a low-pass filter for the accelerometer measurements and performs dead-reckoning of the North components.

The LKF parameters of Filter 2 are presented below:

$$P_2 = 0.1 \mathbf{I}_3 \quad (26)$$

$$Q_2 = 0.01 \mathbf{I}_3 \quad (27)$$

$$R_2 = [0.1] \quad (28)$$

D. Filter 3: East Components

Filter 3 is similar to Filter 2, but deals with the dead-reckoning of the East position, velocity and acceleration. The measurement vector includes the calculated a_E^m from a_x^m and a_y^m following Eq. (9). The matrices are the same as the ones in Eq. (24), (25).

E. Filter 4: GPS Fusion

Filter 4 is an EKF that uses the GPS measurements to correct the North and East position, velocity, and acceleration obtained through dead-reckoning of Filter 2 and Filter 3. It is linear in the dynamics, but nonlinear in the measurements (Eq.(16)).

The dynamics transition matrix is obtained from Eq. (1)-(6) and presented in Eq. (29).

$$A = \begin{bmatrix} 1 & dt & \frac{1}{2}dt^2 & 0 & 0 & 0 \\ 0 & 1 & dt & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & dt & \frac{1}{2}dt^2 \\ 0 & 0 & 0 & 0 & 1 & dt \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad (29)$$

The nonlinear measurements function and their Jacobians are presented in Eq. (30) and (31) respectively.

$$\mathbf{y} = \begin{bmatrix} x_{N,gps} \\ x_{E,gps} \\ v_{gps} \end{bmatrix} = \begin{bmatrix} x_{N,gps} \\ x_{E,gps} \\ \sqrt{v_N^2 + v_E^2} \end{bmatrix} \quad (30)$$

where $x_{N,gps}$ and $x_{E,gps}$ are obtained from the latitude and longitude using the Python *pymap3d* library.

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & \frac{v_N}{\sqrt{v_N^2 + v_E^2 + \epsilon}} & 0 & 0 & \frac{v_E}{\sqrt{v_N^2 + v_E^2 + \epsilon}} & 0 \end{bmatrix} \quad (31)$$

where $\epsilon = 10^{-6}$ is a small regularization constant to prevent division by zero.

The EKF parameters of Filter 4 are presented below:

$$P_4 = 0.1 \mathbf{I}_6 \quad (32)$$

$$Q_4 = 0.01 \mathbf{I}_6 \quad (33)$$

$$R_4 = \begin{bmatrix} 100 & 0 & 0 \\ 0 & 100 & 0 \\ 0 & 0 & 0.5 \end{bmatrix} \quad (34)$$

F. Pseudocode and Implementation

Algorithm 1 summarizes the proposed asynchronous IMU/GPS inertial navigation system. At every iteration, the dynamics are stepped forward once for all filters, since the prediction step is common regardless of which measurements are available. The IMU-based corrections (Filters 1–3) are then applied at every iteration at 100 Hz, while the GPS correction (Filter 4) is only triggered when a new GPS measurement is available, i.e., every 20 iterations at 5 Hz.

Algorithm 1: Asynchronous IMU/GPS Inertial Navigation System

Input: IMU measurements at 100 Hz, GPS measurements at 5 Hz

Output: $\hat{x}_N, \hat{v}_N, \hat{a}_N, \hat{x}_E, \hat{v}_E, \hat{a}_E, \hat{\psi}, \hat{b}_r$

```

1 for  $k = 0, 1, 2, \dots$  do
2   Step the dynamics forward (Eq. (1)-(8));
3   Perform correction of Filter 1 (Section IV-B);
4   Perform correction of Filter 2 (Section IV-C);
5   Perform correction of Filter 3 (Section IV-D);
6   if new GPS measurement available then
7     | Perform correction of Filter 4 (Section IV-E);
8   end
9 end

```

The code is implemented in Python, and the LKF and EKF are coded from scratch. The code is available in the following GitHub repository.

V. RESULTS

The proposed asynchronous IMU/GPS inertial navigation system is evaluated on the MathWorks dataset described in Section III. The estimated states are compared against the ground truth provided in the dataset. The results are organized by filter: heading and yaw bias (Filter 1), North components (Filter 2), East components (Filter 3), and GPS fusion (Filter 4).

A. Filter 1: Heading Angle and Yaw Rate Bias

Figure 1 shows the estimated heading angle $\hat{\psi}$ and gyroscope bias $\hat{b}_r$ compared to their true values. The heading estimate tracks the ground truth closely with a root-mean-square error (RMSE) of 0.94° . The green line represents the heading angle obtained by integration of the yaw rate without correction. It shows the significant error caused by the gyroscope bias and the positive effect of the magnetometer measurements. The magnetometer provides an absolute reference that prevents unbounded drift in the heading angle, while the Kalman filter smooths out the magnetometer noise. The estimated yaw rate bias $\hat{b}_r$ converges to the true value after around 20 s.

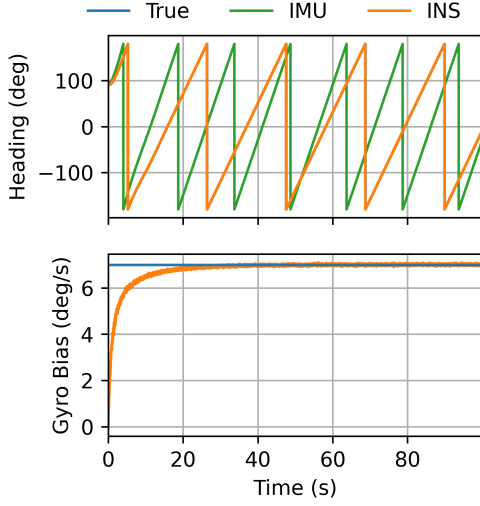


Fig. 1. Filter 1 results. Top: estimated vs. true heading angle ψ . The green line shows yaw rate integration without correction. Bottom: estimated yaw rate bias $\hat{b}_r$. The estimated quantities match the true values closely.

B. Filters 2, 3 and 4: Positioning Estimation

Filters 2, 3, and 4 work together to produce the final North and East state estimates. Filters 2 and 3 perform IMU-driven dead-reckoning of the position, velocity, and acceleration in the North and East directions respectively, while Filter 4 periodically corrects the accumulated drift using GPS measurements. Because the three filters are tightly coupled (Filter 4 takes the outputs of Filters 2 and 3 as its prior), their performance is evaluated jointly on the final corrected estimates.

1) *Position*: Figure 2 shows the estimated North and East positions compared to the ground truth as a function of time, along with the GPS measurements. The GPS measurements are off by around 2-4 meters, but the estimated values remain close to the true values. The position RMSEs are **0.78m** in the North direction and **1.2m** in the East direction.

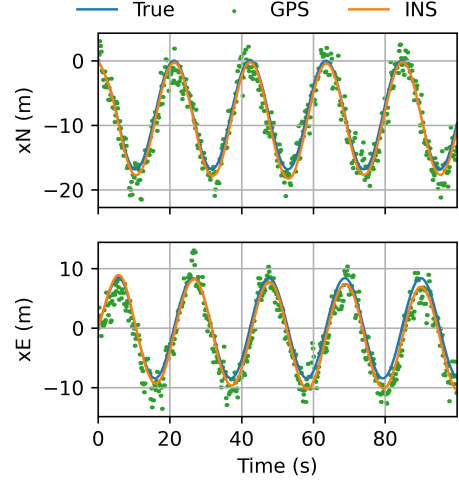


Fig. 2. Estimated vs. true position. Top: North position x_N , RMSE=0.78m. Bottom: East position x_E , RMSE=1.2m.

Figure 3 overlays the full estimated 2D trajectory on the ground truth in the NEU navigation frame, along with the GPS measurements. The diamonds indicate the final position of the vehicle. The plot does not visualize time information, but shows the extent to which the estimates follow the perfect circular pattern of the true trajectory given the variance in GPS position.

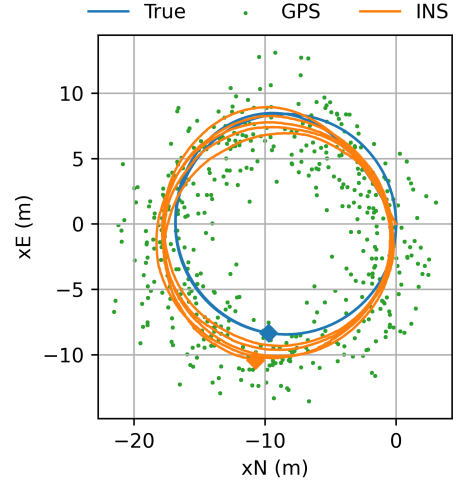


Fig. 3. Estimated vs. true trajectory in the North-East plane. The diamonds represent the final position of the vehicle

2) *Velocity*: Figure 4 shows the estimated North and East velocities compared to the truth. The velocity estimates benefit from both the IMU integration (Filters 2 and 3) and the indirect correction through the GPS ground speed measurement in Filter 4. Because the GPS only provides ground speed (the magnitude of velocity) and not the individual North and East components, the decomposition of the correction into the two directions is handled through the Jacobian of Filter 4

(Eq. 31), which weights the correction proportionally to each component's contribution to the total speed. The velocity RMSEs are 0.1 m/s in the North direction and 0.1 m/s in the East direction.

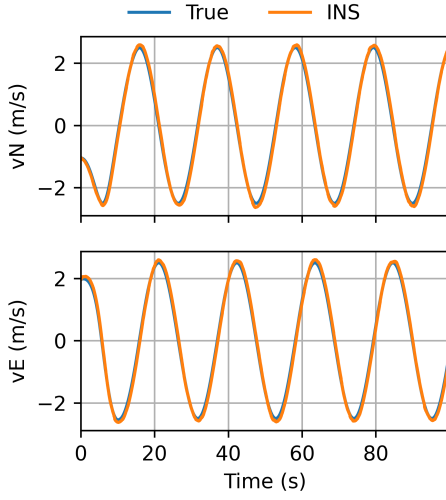


Fig. 4. Estimated vs. true velocity components in the navigation frame. Top: North velocity v_N . Bottom: East velocity v_E . The estimated signals are produced by the combined IMU dead-reckoning and GPS fusion.

Figure 5 plots the estimated vs. true ground speed, along with the GPS ground speed measurements used for correction. The estimate tracks the true ground speed closely with an RMSE of 0.09 m/s (0.324 km/h), which is well within the accuracy requirements of a typical vehicle navigation system. The GPS measurements themselves are noisier than the final estimate, which demonstrates the benefit of the Kalman filter: by blending the GPS measurements with the smoother IMU-driven prediction, the filter produces a cleaner speed estimate than either sensor alone would provide.

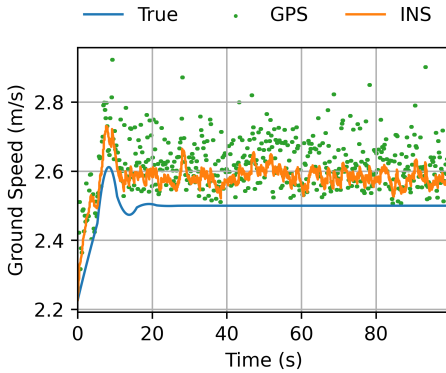


Fig. 5. Estimated vs. true ground speed, alongside the raw GPS speed measurements used for Filter 4 correction. The Kalman filter estimate is visibly smoother than the raw GPS signal while remaining close to the ground truth.

3) *Acceleration*: Figure 6 shows the estimated North and East accelerations compared to the ground truth. Even though

acceleration is not directly observed by the GPS, it is still part of the state vector of Filter 4, and therefore receives indirect corrections from it whenever a GPS measurement is available. Additionally, Filters 2-3 act as low-pass smoothers on the noisy accelerometer signal at every IMU update, reducing high-frequency noise while preserving the underlying trend. The acceleration RMSEs are 0.01 m/s^2 in the North direction and 0.01 m/s^2 in the East direction. These errors are driven primarily by the accelerometer noise and the heading error in the DCM (Eq. (9)), since any error in $\hat{\psi}$ introduces a cross-coupling between the North and East acceleration components during the body-to-navigation frame transformation.

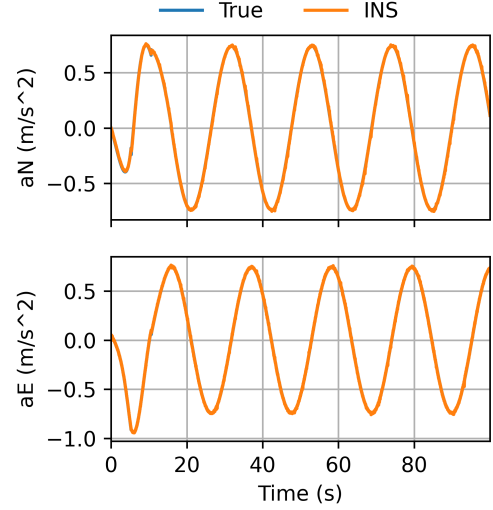


Fig. 6. Estimated vs. true acceleration components in the navigation frame. Top: North acceleration a_N . Bottom: East acceleration a_E . Although acceleration is not directly measured by the GPS, it receives indirect corrections from Filter 4 through the Kalman gain.

VI. ABLATION: DEAD-RECKONING ONLY

To assess the contribution of the GPS fusion step, an ablation study is conducted in which Filter 4 is disabled and the system relies solely on the IMU-driven dead-reckoning of Filters 1–3. This isolates the effect of the GPS correction and quantifies how much the position and velocity estimates degrade without it.

A. Heading

The heading estimate is unaffected by this ablation, since Filter 1 relies exclusively on the gyroscope and magnetometer and does not interact with the GPS. The heading RMSE remains 0.94° , confirming that the orientation estimation is self-contained and robust independently of the GPS correction.

B. Position

Figure 7 shows the North and East position estimates under dead-reckoning only, compared to the full system and the ground truth. Without GPS corrections, the position

error, especially North, grows unboundedly over time. Small errors in acceleration accumulate first into velocity drift and then into a quadratically growing position error through double integration, which is the fundamental limitation of dead-reckoning. In contrast, the full system keeps the position estimate bounded by periodically anchoring it to the GPS measurements. The position RMSEs under dead-reckoning only are **2.07 m** in the North direction and **1.2 m** in the East direction.

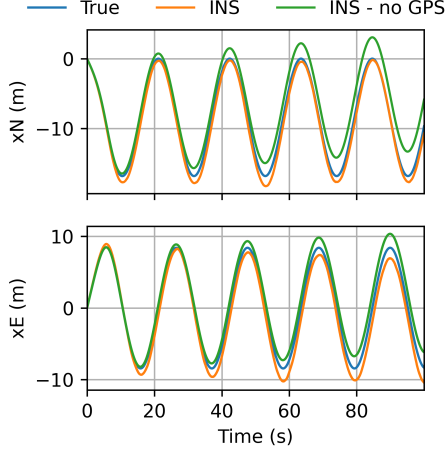


Fig. 7. Position estimates under dead-reckoning only vs. the full INS, compared against ground truth. Top: North position x_N . Bottom: East position x_E . The drift is especially clear in the North component.

Figure 8 overlays the dead-reckoning trajectory on the ground truth and the full system estimate in the NEU frame. The drift in the North and East positions accumulates into a visible divergence from the true circular path, with the dead-reckoning trajectory spiraling away from the ground truth over time. This provides the clearest illustration of why GPS fusion is indispensable: without it, the vehicle's estimated position becomes increasingly unreliable, regardless of the quality of the IMU measurements.

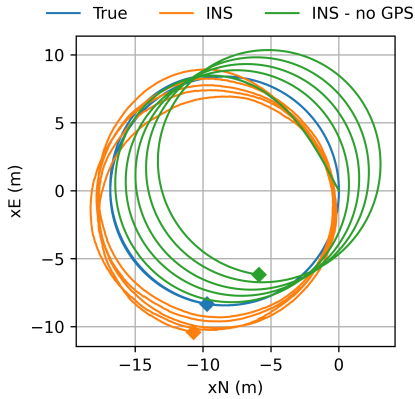


Fig. 8. Trajectory comparison in the NEU navigation frame. The dead-reckoning only path diverges progressively from the ground truth. The full IMU/GPS estimate follows the true trajectory better

C. Velocity

Interestingly, the velocity and ground speed estimates under dead-reckoning remain closer to the ground truth than the full system estimate, as shown in Figure 9. This can be attributed to two factors. First, the GPS ground speed measurements used in Filter 4 carry a small offset relative to the true speed, meaning that the GPS correction slightly pulls the velocity estimate away from the truth in this particular dataset. Second, and more importantly, the accelerometer bias is assumed to be zero in this project, which prevents velocity drift from occurring. In a more realistic scenario where the accelerometer carries a non-zero bias, the single integration of that bias would cause the velocity estimate to drift linearly over time, just as the position drifts without GPS. Therefore, while the dead-reckoning velocity appears more accurate in this dataset, this should not be interpreted as evidence that GPS correction is unnecessary for velocity estimation in general.

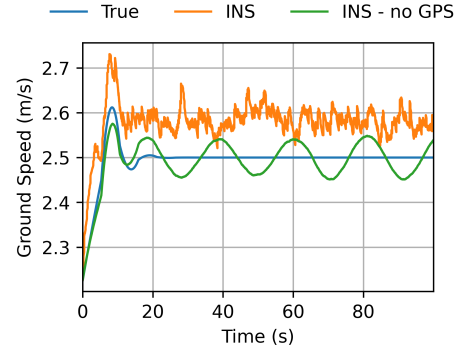


Fig. 9. Ground speed comparison between the full INS, the dead-reckoning only system, and the ground truth. The dead-reckoning estimate tracks the true ground speed more closely than the full system due to the offset in the GPS speed measurements and the zero accelerometer bias assumption.

D. Summary

Table II summarizes the RMSE of the position and velocity states for both configurations. The GPS fusion step provides substantial improvements in position accuracy. The velocity results are comparable between the two configurations for the reasons discussed above, but would be expected to diverge significantly in the presence of accelerometer bias.

TABLE II
RMSE COMPARISON: FULL SYSTEM VS. DEAD-RECKONING ONLY

State	Units	Full INS	INS - No GPS
x_N	m	0.78	2.07
x_E	m	1.2	1.2
v_N	m/s	0.1	0.05
v_E	m/s	0.1	0.03
a_N	m/s ²	0.01	0.004
a_E	m/s ²	0.01	0.01
ψ	deg	0.94	0.94

VII. CONCLUSION

This project presented the design and implementation of an asynchronous GNSS/INS system for 2D vehicle state estimation, developed as a foundation for the Formula SAE car at Arizona State University. The system fuses IMU and GPS measurements through a bank of four Kalman filters: three Linear Kalman Filters for high-rate IMU-driven dead-reckoning of the heading, North, and East states, and one Extended Kalman Filter for low-rate GPS corrections. The asynchronous architecture allows the two sensor modalities to operate at their native rates, 100 Hz for the IMU and 5 Hz for the GPS, without requiring synchronization.

The system was evaluated on a MathWorks ground vehicle dataset, and the results demonstrate accurate state estimation across all quantities. The heading angle is estimated with an RMSE of 0.94° , with the magnetometer providing an absolute reference that prevents gyroscope drift. The position estimates achieve RMSEs of 0.78 m and 1.2 m in the North and East directions respectively, despite the GPS measurements carrying errors of 2–4 m, demonstrating the effectiveness of the Kalman filter in producing estimates that are more accurate than the raw measurements. The velocity and acceleration estimates are similarly accurate, with ground speed estimated to within 0.09 m/s (0.324 km/h). The ablation study further confirms the complementary roles of the two sensor modalities: the IMU-driven filters provide smooth, continuous, high-rate estimates, while the GPS fusion step is essential for bounding the long-term position drift that is otherwise unavoidable with dead-reckoning alone.

Accurate and reliable state estimation is necessary for motorsports, and the results presented here establish a solid algorithmic foundation upon which a full, real-world GNSS/INS system for the ASU Formula SAE car can be built.

VIII. LIMITATIONS AND FUTURE WORK

A. Limitations

1) *Dataset*: The primary limitation of this project is the absence of real-world data from the Formula SAE car. Hardware and integration issues with the onboard sensors prevented the collection of consistent measurements during track operation, and the system was therefore developed and validated exclusively on the MathWorks dataset. This dataset captures a vehicle in constant circular motion, which is significantly less dynamic than a real FSAE track session involving hard acceleration, braking, and cornering. Furthermore, the GPS in the dataset operates at 5 Hz, whereas the GPS module on the FSAE car samples at 40 Hz, meaning that the asynchronous update logic has not been validated at the higher GPS rate. As a result, the performance figures reported in this project should be interpreted as an indicative starting point rather than representative of real-world operation.

2) *2D Motion Assumption*: The system assumes that the vehicle operates exclusively on a flat 2D plane, with no roll or pitch motion. This simplifies the kinematics model and

reduces the number of states and required IMU measurements. While this assumption is reasonable for most Formula SAE racetracks, it would break down on any surface with elevation changes or in the presence of significant vehicle body motion over kerbs and bumps.

3) *Accelerometer Bias*: The accelerometer bias is assumed to be zero in this project. In practice, accelerometer biases are non-zero and time-varying, and their single integration into velocity would cause a linearly growing drift that the GPS fusion step would need to correct at every update. The ablation study illustrated the consequence of this assumption: the velocity estimates under dead-reckoning appeared artificially accurate because there was no bias to drive a drift. In a real deployment, neglecting accelerometer bias would degrade both the velocity and position estimates between GPS corrections.

B. Future Work

1) *Validation on the FSAE Car*: Testing on real data would expose the system to the full dynamics of a race car, including aggressive maneuvers and sensor imperfections not captured by the MathWorks dataset, and would provide a more meaningful assessment of the system's accuracy and robustness.

2) *Full 3D INS with Quaternions and Bias Estimation*: The longer-term direction is to extend the system to a full 3D INS that removes the flat-plane assumption and handles the complete six-degrees-of-freedom motion of the vehicle. This would require replacing the DCM-based heading representation with a quaternion attitude parameterization. The state vector would be expanded to include roll, pitch and yaw, as well as explicit bias estimates for all three accelerometer axes. These extensions would bring the system significantly closer to the production-grade GNSS/INS implementations used in professional motorsports.

REFERENCES

- [1] K. M. Smalling and K. W. Eure, "A Short Tutorial on Inertial Navigation System and Global Positioning System Integration." National Aeronautics and Space Administration, Sep. 2015.
- [2] M. Bersani, M. Vignati, S. Mentasti, S. Arrigoni, and F. Cheli, "Vehicle State Estimation Based on Kalman Filters," 2019.
- [3] C. K. Song, M. Uchanski, and J. K. Hedrick, "Vehicle Speed Estimation Using Accelerometer and Wheel Speed Measurements," 2002.
- [4] MathWorks, "Estimate Position and Orientation of a Ground Vehicle," [Online]. Available: <https://www.mathworks.com/help/nav/ug/estimateposition-and-orientation-of-a-ground-vehicle.html>
- [5] National Centers for Environmental Information, "Magnetic Field Calculators" [Online]. Available: <https://www.ngdc.noaa.gov/geomag/calculators/magcalc.shtml>
- [6] G. Im, "Notes on Kalman Filter (KF, EKF, ESKF, IEKF, IESKF)," arXiv preprint arXiv:2406.06427, 2024. [Online]. Available: <https://arxiv.org/abs/2406.06427>

APPENDIX A MATHWORKS DATASET

The dataset is taken from the MathWorks example **Estimate Position and Orientation of a Ground Vehicle**, which captures the circular motion of a vehicle on a flat 2D plane [4]. The example does not provide any details or code on how to design or implement a GNSS/INS; instead it is an example on how to use the Navigation Toolbox, which is private. The dataset contains all the needed values to perform state estimation and to check its accuracy.

In terms of sensor measurements, the dataset includes: 3-axis accelerometer, 3-axis gyroscope, 3-axis magnetometer, 3-axis GPS velocity, and GPS latitude, longitude and altitude.

Figure 10 plots the required IMU measurements for 2D estimation, while Figure 11 plots the needed GPS measurements. It is to be noted that the GPS ground speed is used instead of North and East components in order to match the GPS module that is onboard the FSAE car:

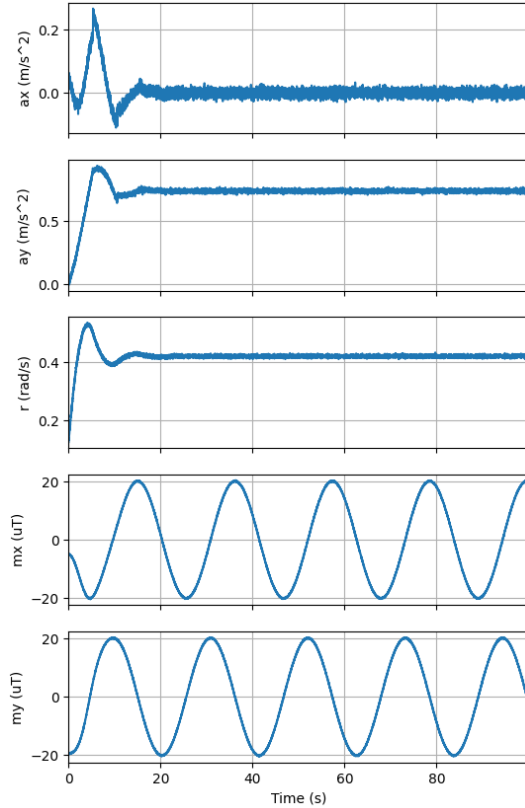
$$v_{gps} = \sqrt{v_N^2 + v_E^2}.$$


Fig. 10. IMU measurements from the MathWorks dataset. From top to bottom: longitudinal acceleration, lateral acceleration, yaw rate, magnetometer longitudinal intensity and magnetometer lateral intensity. All the IMU measurements are taken in the body reference frame.

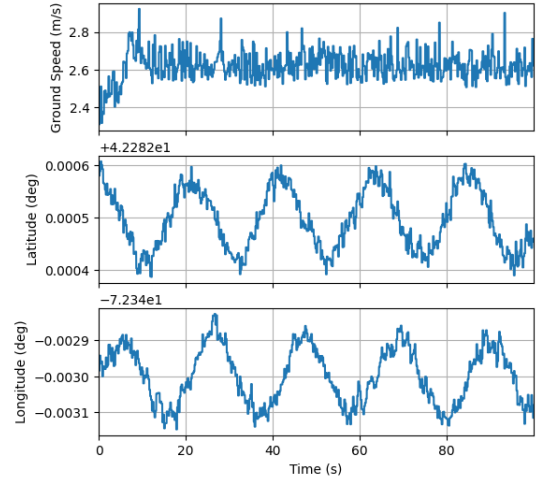


Fig. 11. GPS measurements from the MathWorks dataset. From top to bottom: ground speed, latitude and longitude. All the GPS measurements are taken in the navigation reference frame. The GPS North and East velocities are combined into a single ground speed to be consistent with the GPS module in the car.

As for the true states, the dataset includes an extensive list of variables that help in assessing the accuracy of the estimation algorithm. All the true values are in the navigation frame (NEU): 3 positions, 3 velocities, 3 accelerations, 3 angular velocities, and 3 orientation angles. Only a subset of these are plotted here (Fig. 12 and Fig. 13) to give an overview of the trajectory and maintain brevity. More states are shown in the results when discussing the performance of the filter.

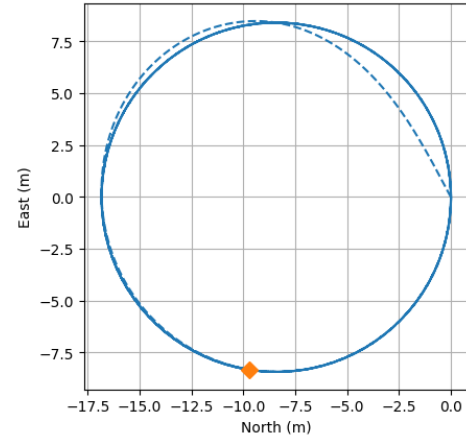


Fig. 12. True vehicle trajectory in the dataset. The vehicle starts at (0,0) in the NEU frame and performs counterclockwise circles. The orange diamond shows the final position of the vehicle

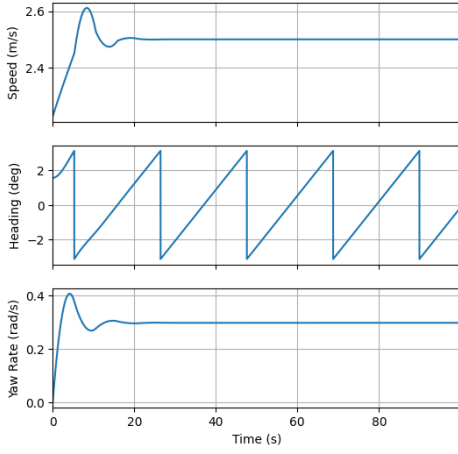


Fig. 13. A subset of true vehicle states from the dataset. From top to bottom: ground speed, heading angle (from North to East) and yaw rate

APPENDIX B LINEAR KALMAN FILTER

The Linear Kalman Filter is an optimal recursive estimator that computes the minimum mean square error estimate of a system's state from noisy measurements, assuming linear dynamics and Gaussian noise [6]. The system is described by the discrete-time state-space model:

$$\mathbf{x}_k = \mathbf{A}_k \mathbf{x}_{k-1} + \mathbf{B}_k \mathbf{u}_k + \mathbf{w}_k, \quad \mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_k) \quad (35)$$

$$\mathbf{y}_k = \mathbf{C}_k \mathbf{x}_k + \mathbf{v}_k, \quad \mathbf{v}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_k) \quad (36)$$

where $\mathbf{x}_k$ is the state vector, $\mathbf{A}_k$ is the state transition matrix, $\mathbf{u}_k$ is a known control input, $\mathbf{y}_k$ is the measurement vector, $\mathbf{C}_k$ is the observation matrix, and $\mathbf{Q}_k$, $\mathbf{R}_k$ are the process and measurement noise covariance matrices, respectively. The filter alternates between a *prediction* step and an *update* step:

Prediction:

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{A}_k \hat{\mathbf{x}}_{k-1|k-1} + \mathbf{B}_k \mathbf{u}_k \quad (37)$$

$$\mathbf{P}_{k|k-1} = \mathbf{A}_k \mathbf{P}_{k-1|k-1} \mathbf{A}_k^\top + \mathbf{Q}_k \quad (38)$$

Update:

$$\mathbf{K}_k = \mathbf{P}_{k|k-1} \mathbf{C}_k^\top (\mathbf{C}_k \mathbf{P}_{k|k-1} \mathbf{C}_k^\top + \mathbf{R}_k)^{-1} \quad (39)$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k (\mathbf{y}_k - \mathbf{C}_k \hat{\mathbf{x}}_{k|k-1}) \quad (40)$$

$$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k \mathbf{C}_k) \mathbf{P}_{k|k-1} \quad (41)$$

where $\mathbf{K}_k$ is the Kalman gain, $\hat{\mathbf{x}}_{k|k-1}$ is the *a priori* state estimate, $\hat{\mathbf{x}}_{k|k}$ is the *a posteriori* state estimate, and $\mathbf{P}_k$ is the corresponding error covariance matrix.

APPENDIX C EXTENDED KALMAN FILTER

The Extended Kalman Filter (EKF) generalises the LKF to nonlinear systems by linearising the dynamics and observation

models at each time step via first-order Taylor expansion [6]. The system is now described by:

$$\mathbf{x}_k = f(\mathbf{x}_{k-1}, \mathbf{u}_k) + \mathbf{w}_k \quad (42)$$

$$\mathbf{y}_k = h(\mathbf{x}_k) + \mathbf{v}_k \quad (43)$$

where $f(\cdot)$ and $h(\cdot)$ are nonlinear functions. The key difference from the LKF is that these functions are linearised using their Jacobians:

$$\mathbf{F}_k = \left. \frac{\partial f}{\partial \mathbf{x}} \right|_{\hat{\mathbf{x}}_{k-1|k-1}}, \quad \mathbf{H}_k = \left. \frac{\partial h}{\partial \mathbf{x}} \right|_{\hat{\mathbf{x}}_{k|k-1}} \quad (44)$$

The prediction and update equations follow the same structure as the LKF, but with the nonlinear functions applied to propagate the state estimate and the Jacobians used for the covariance propagation:

Prediction:

$$\hat{\mathbf{x}}_{k|k-1} = f(\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}_k) \quad (45)$$

$$\mathbf{P}_{k|k-1} = \mathbf{F}_k \mathbf{P}_{k-1|k-1} \mathbf{F}_k^\top + \mathbf{Q}_k \quad (46)$$

Update:

$$\mathbf{K}_k = \mathbf{P}_{k|k-1} \mathbf{H}_k^\top (\mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^\top + \mathbf{R}_k)^{-1} \quad (47)$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k (\mathbf{y}_k - h(\hat{\mathbf{x}}_{k|k-1})) \quad (48)$$

$$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_{k|k-1} \quad (49)$$

Because linearisation is only an approximation, the EKF is no longer guaranteed to be optimal. It can diverge when the nonlinearities are strong or the initial estimate is poor.