EECE 5554 Lab 5: IMU and GPS Sensor Fusion (Dead Reckoning)for Vehicle Navigation

Kiran Sairam Bethi Balagangadaran

November 30, 2025

1 Introduction

Performing dead reckoning using accurate position and heading estimates from sensors mounted in a moving car turns out to be surprisingly difficult. This lab looks at that problem using data collected from a VectorNav VN-100 IMU and GPS receiver during an actual drive around Boston. The difficult part of the challenge is that different sensors have opposite strengths and weaknesses. Magnetometers and GPS give us absolute measurements that do not drift over time, but they are noisy and update slowly. Gyroscopes and accelerometers are smooth and fast, but small errors accumulate when we integrate them, becoming large at the end.

The data, graciously shared by Professor Kris Dorsey, was recorded during two driving sessions: first, driving circles around the Ruggles circle for magnetometer calibration, and a longer drive through Boston streets. The goal was to fuse the data from these sensors, taking the best parts of each while canceling out their weaknesses. This report goes through magnetometer calibration, heading estimation via complementary filtering, velocity estimation from acceleration, and finally trajectory reconstruction through dead reckoning.

2 Magnetometer Calibration

2.1 Calibration Matters

In theory, a magnetometer should measure the Earth's magnetic field and directly tell us which way is north. In practice, mounting one in a car creates problems. The Earth's field in Boston is only about 200-250 milligauss in the horizontal plane. That is a relatively weak signal, and our car, its engine, and the car's relatively large magnets (in the form of speaker magnets, motors, etc) add their own magnetic fields on top of it.

The car's engine, battery, alternator motor, and speaker magnets all generate magnetic fields. These create hard iron distortions, an offset that shifts all our measurements away from where they should be. On top of that, the car's steel body distorts Earth's field lines. This soft iron distortion turns a circular pattern into an ellipse.

Looking at uncalibrated data, we measured hard-iron offsets of around 200 milligauss in both axes.

2.2 Calibration Procedure

The important part of calibration is getting measurements from all possible orientations. This was done by driving atleast 4-5 loops around the Ruggles Circle. This sweeps the magnetometer through every heading, creating a complete picture of the distortion pattern.

When we plot mag_x versus mag_y from this circular driving, we should see a circle if everything is perfect. What is actually observed is an ellipse that is shifted away from the origin. The center of this ellipse shows the hard iron offset:

$$\text{offset}_x = \frac{\max(\text{mag}_x) + \min(\text{mag}_x)}{2}, \quad \text{offset}_y = \frac{\max(\text{mag}_y) + \min(\text{mag}_y)}{2} \tag{1}$$

After subtracting these offsets, we have removed the hard iron effects. Now we are left with an ellipse centered at the origin, which reveals the soft iron distortion. To fix this, we fit an ellipse to the data using eigenvalue decomposition. This gives us the orientation of the ellipse and how stretched it is stretched in each direction. We then build a transformation matrix that rotates and scales the ellipse back into a circle.

We calculate the covariance matrix of the centered data, find its eigenvalues and eigenvectors, then construct a transformation that makes both axes equal in length. The result is a 2×2 matrix that corrects both rotation and stretching.

$$\mathbf{mag}_{\text{calibrated}} = \mathbf{S} \cdot (\mathbf{mag}_{\text{raw}} - \mathbf{offset}) \tag{2}$$

2.3 Calibration Results

Our calibration produced these values:

- Hard iron offset: [197.85, 128.90] mG
- Soft iron matrix: $\begin{bmatrix} 1.00017 & -0.00837 \\ -0.00837 & 0.99997 \end{bmatrix}$
- Ellipse eccentricity before correction: 0.1814
- Field magnitude after correction: 220.28 mG

Figure 1 shows the before and after. The raw data forms an ellipse offset from the origin, while the calibrated data makes a round circle centered at zero.

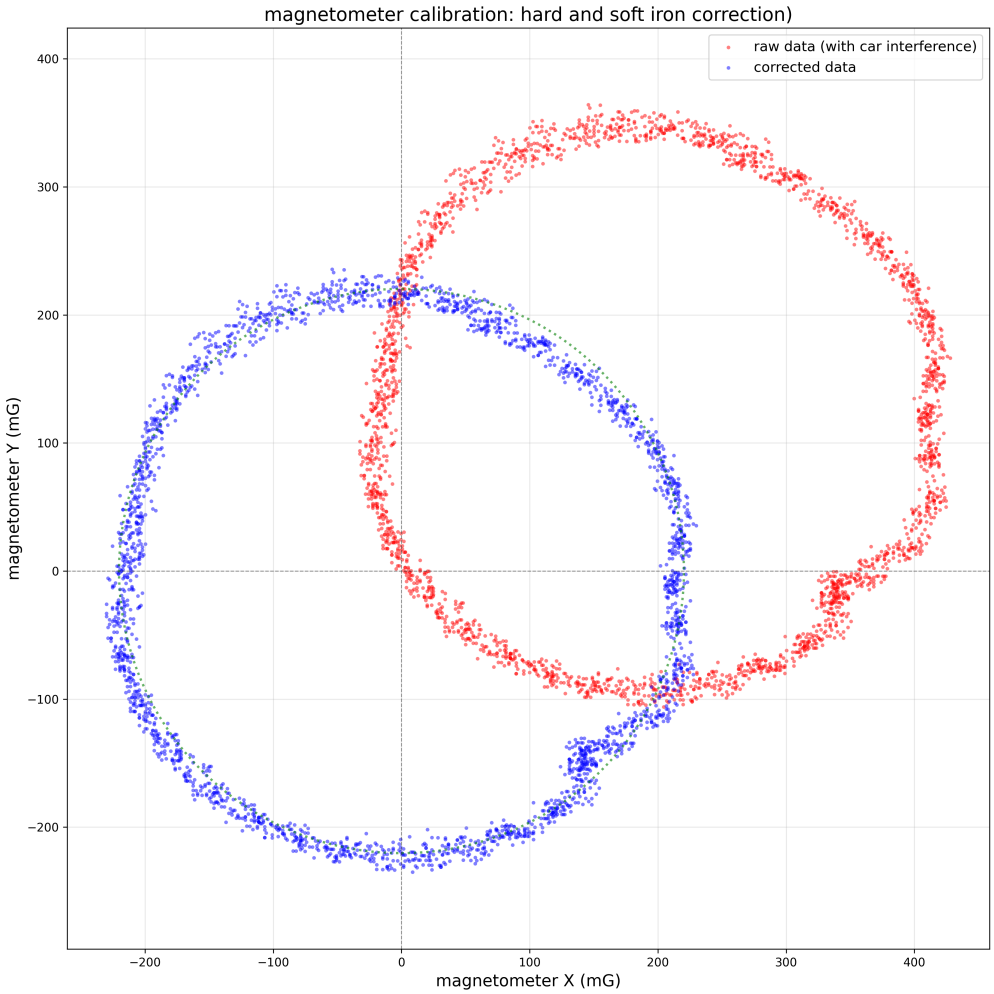

Figure 1: Magnetometer calibration results. Raw measurements (red) show the offset ellipse caused by magnetic interference. After correction, the calibrated data (blue) forms a round circle centered at the origin.

3 Heading Estimation

3.1 Magnetometer vs Gyroscope

Once the magnetometer is calibrated, we can find the heading directly using arctangent:

$$\psi_{\text{mag}} = \text{atan2}(\text{mag}_{y,\text{cal}}, \text{mag}_{x,\text{cal}}) \tag{3}$$

This gives absolute heading. It does not drift, and we can trust it to point toward the magnetic north. The problem is noise. Even after calibration, the magnetometer jumps around by 0.1-0.2 radians. This makes it difficult to use for trajectory estimation.

The gyroscope measures how fast we are rotating (angular velocity), which we integrate to get heading:

$$\psi_{\text{gyro}}(t) = \psi_0 + \int_0^t \omega_z(\tau)\, d\tau \tag{4}$$

Gyroscope data is smooth. It shows turns and heading changes without noise, but the problem is that integration blows the proportions to very high values. Even a tiny bias in the gyroscope reading accumulates over time. I attempted to remove the bias by subtracting the mean value over the entire drive, but that only helps so much.

So we have two sensors that are almost perfectly complementary. The magnetometer does not drift but it is noisy. The gyroscope is smooth but it drifts. The solution is to combine them using a complementary filter.

3.2 Complementary Filter Implementation

The complementary filter splits the frequency between the two sensors. We feed the magnetometer reading through a low pass filter, keeping only the slow, stable heading changes. Similarly, we run the integrated gyroscope through a high pass filter, keeping only the fast movements while discarding the slow drift. Then we just add them together:

$$\psi_{\text{fused}}(t) = \text{LPF}[\psi_{\text{mag}}(t)] + \text{HPF}[\psi_{\text{gyro}}(t)] \tag{5}$$

The important part is choosing a cutoff frequency. This value determines when we switch from trusting the gyroscope to trusting the magnetometer.

To find the optimal cutoff, I tried values from 0.05 Hz up to 0.25 Hz and looked at how well each performed. Too low (0.05 Hz) and the result follows the magnetometer too closely, keeping excessive noise. Too high (0.2 Hz) and it follows the gyroscope too much, not correcting drift enough. I decided from each option by looking at smoothness, how much it drifted from the magnetometer, and average deviation.

The analysis pointed to 0.1 Hz as an ideal value. This corresponds to a time constant of about 10 seconds. We can trust the gyroscope for heading changes happening over a few seconds, but beyond that, we need the magnetometer to keep consistent readings.

3.3 Heading Results

Figure 2 shows the calibration of magnetometer yaw. The raw magnetometer yaw (blue) shows a large offset. After applying our calibration (red), the yaw measurements are centered in the proper range and track the correct heading changes during the drive.

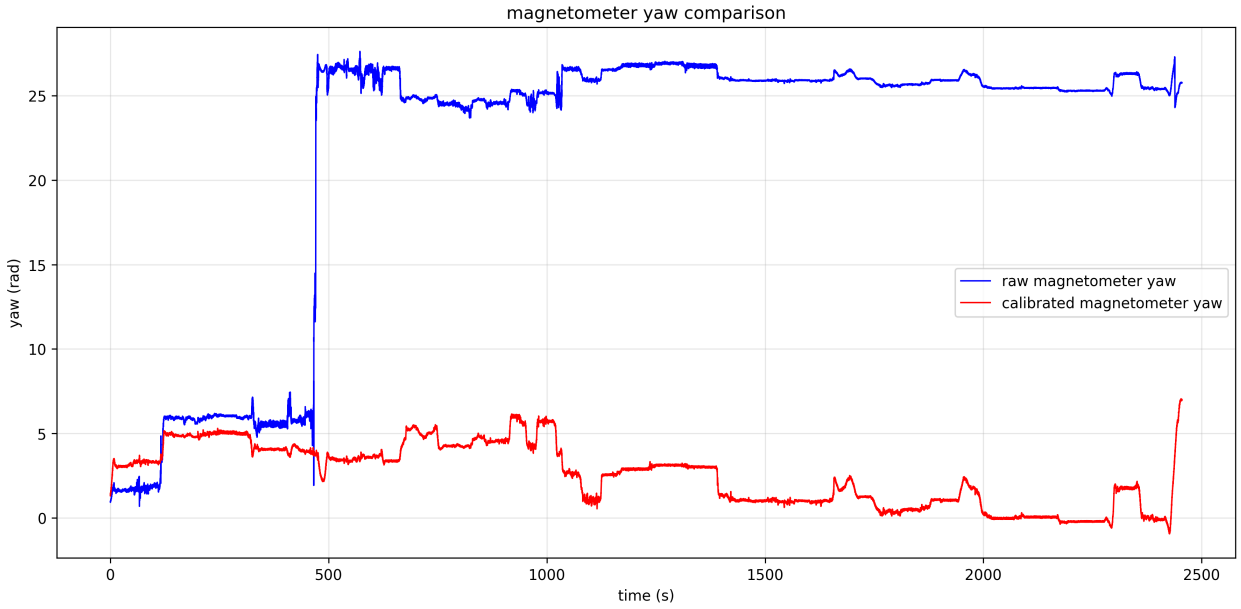

Figure 2: Comparison of raw and calibrated magnetometer yaw, showing the large offset correction done by calibration

Figure 3 compares the yaw of the calibrated magnetometer with the yaw of the integrated gyroscope. The magnetometer (red) provides the absolute heading value, but has noise, while the integrated gyroscope (blue) is smooth in the beginning but drifts over time, accumulating large errors by the end.

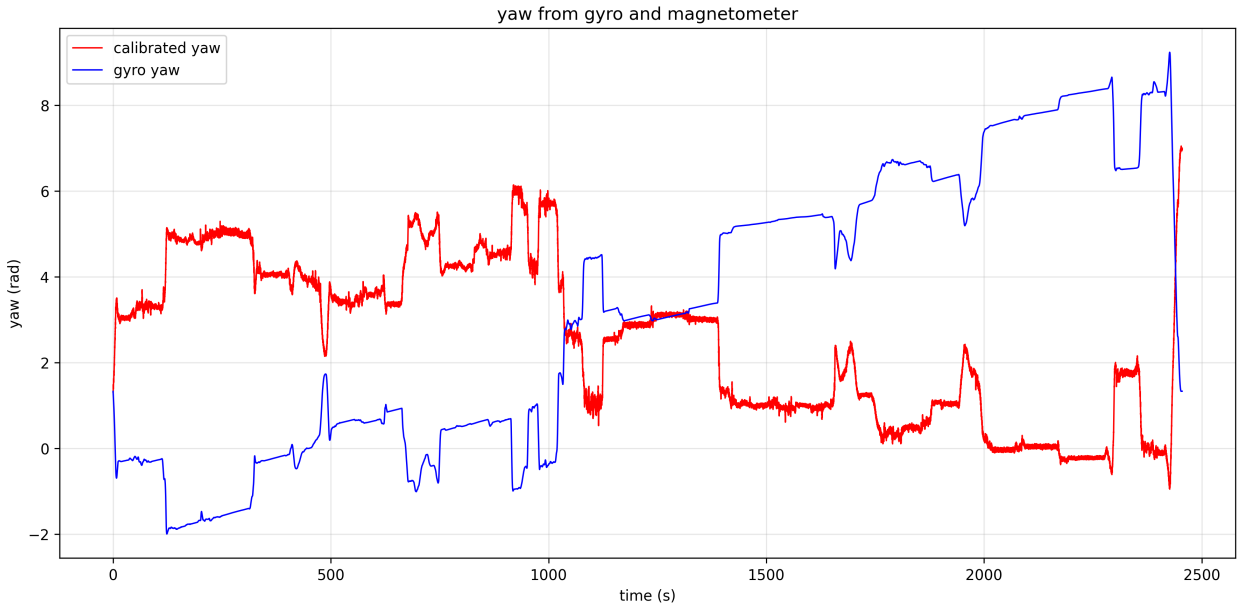

Figure 3: Calibrated magnetometer yaw versus integrated gyroscope yaw, showing the complementary error characteristics that motivate sensor fusion

Figure 4 shows the output of the complementary filter. The top plot shows the two

filtered components separately. The low pass filtered magnetometer (blue) gives smooth heading. The high pass filtered gyroscope (red) captures just the short term movement, around zero.

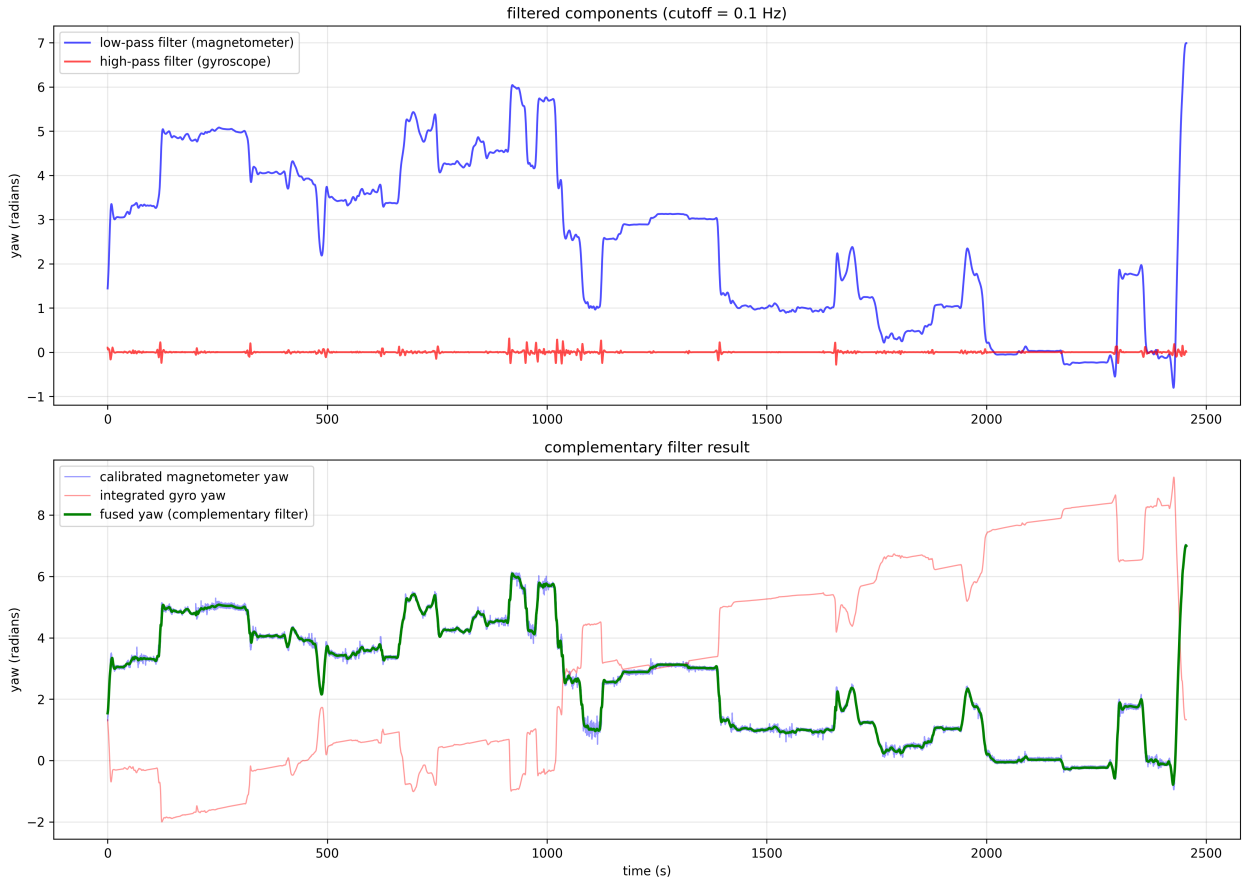

Figure 4: Complementary filter output for heading. Top plot shows filtered components (LPF magnetometer and HPF gyroscope). Bottom plot compares fused output with raw sensor measurements.

The bottom plot compares everything together. The raw magnetometer (light blue) is jagged and noisy. The raw integrated gyroscope (light red) drifts upward continuously. The fused result (green) manages to be smooth like the gyroscope while staying anchored to the magnetometer's correct heading.

We also compared our complementary filter against the IMU's internal heading estimate (Figure 5). For the purposes of this lab, we trust our complementary filter result more because we have full control over the calibration and sensor fusion.

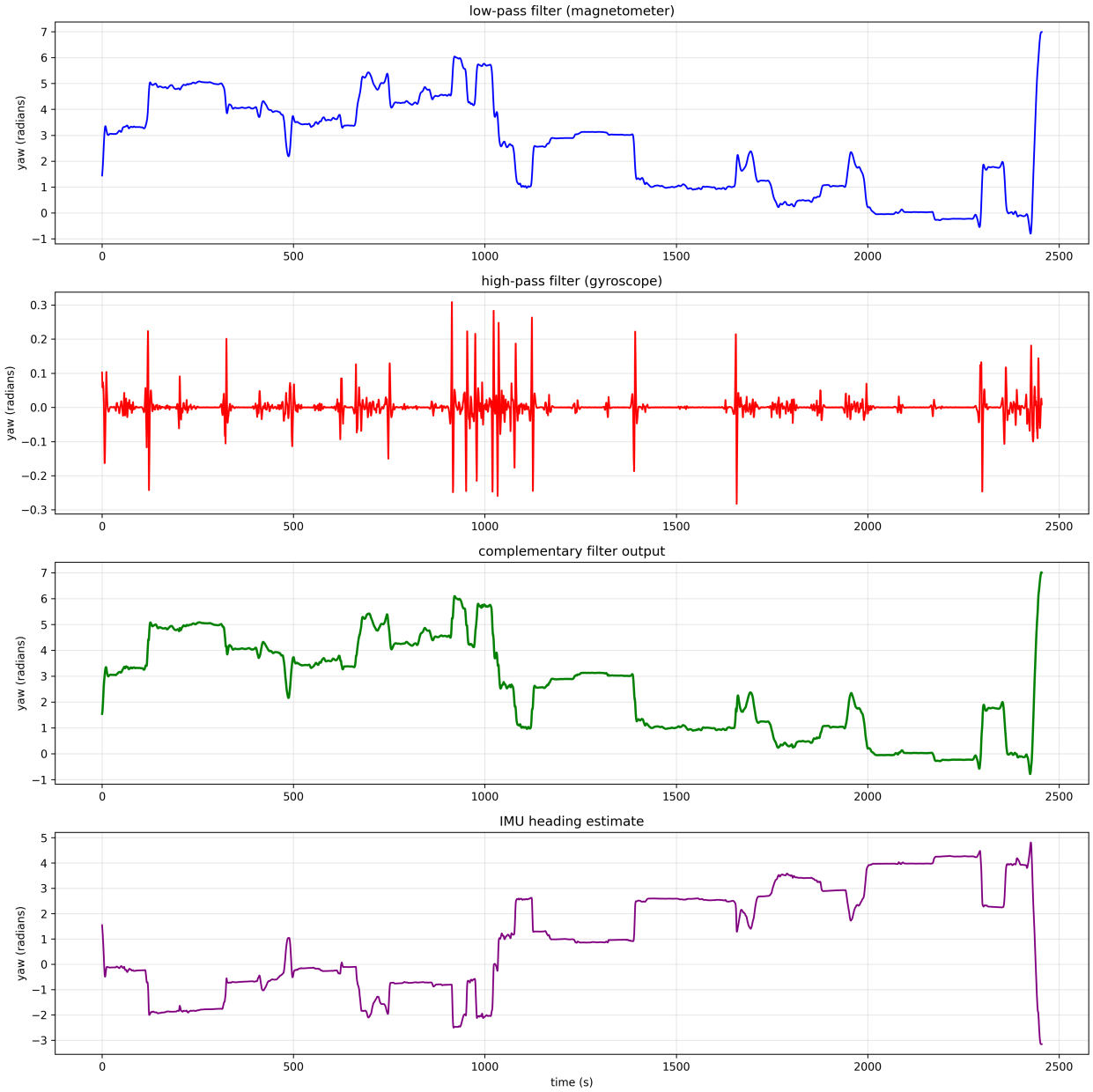

Figure 5: Comparison showing all heading estimation plots. From top to bottom: low pass filtered magnetometer, high pass filtered gyroscope, complementary filter output, and IMU internal heading estimate.

4 Velocity Estimation

4.1 Integration Problems

Estimating velocity from an accelerometer sounds straightforward in theory. Simply integrate acceleration to get velocity. In practice, it is a lot more complicated. I tried this first with just bias correction (subtracting the mean acceleration during the first 10 seconds when the

car was parked). The result was velocity drifting down to -250 m/s, which would be about 900 KMPH backwards. Oops.

The problem is that even microscopic biases in the accelerometer add up to a lot. If there is a 0.1 m/s^2 error (which is tiny and barely noticeable), after 2500 seconds we have accumulated 250 m/s of error.

There is also an interesting detail about our IMU mounting. The velocity goes negative not because of a bug, but because the IMU's x-axis physically points backward in the car. When the car accelerates forward, the IMU measures negative acceleration. When we integrate this, we get negative velocity. It is not wrong, just a coordinate frame issue that would need a sign inverted to match the vehicle frame.

4.2 GPS as a Ground Truth

GPS gives us an independent way to check velocity. We calculated it using the Haversine formula, which accounts for Earth's curvature when calculating the distance between latitude and longitude points:

$$a = \sin^2\left(\frac{\Delta\phi}{2}\right) + \cos(\phi_1)\cos(\phi_2)\sin^2\left(\frac{\Delta\lambda}{2}\right) \tag{6}$$

$$c = 2 \cdot \text{atan2}(\sqrt{a}, \sqrt{1-a}) \tag{7}$$

$$d = R \cdot c \tag{8}$$

The velocity is simply distance divided by time. This GPS velocity ranges from 0 to about 14 m/s (roughly 30 mph), which matches what we would expect for city driving with some faster road segments mixed in.

Figure 6 shows the issues faced. The top panel has IMU velocity at -250 m/s while GPS stays sensibly close to zero. We cannot use the raw IMU velocity as is, it needs data processing.

4.3 Fixing Velocity with Complementary Filtering

I applied the same complementary filter idea that worked for heading. Low pass filter the GPS velocity to smooth it out, high pass filter the IMU velocity to remove drift, then add them together:

$$v_{\text{fused}} = \text{LPF}[v_{\text{GPS}}] + \text{HPF}[v_{\text{IMU}}] \tag{9}$$

The cutoff frequency needed optimization again. I tested values from 0.01 Hz up to 0.25 Hz, comparing root mean square errors against the GPS velocity for each. This was the results:

- 0.01 Hz: RMSE = 3.59 m/s (too much drift)
- 0.05 Hz: RMSE = 1.00 m/s
- 0.10 Hz: RMSE = 0.41 m/s (ideal)

Higher cutoffs worked better for velocity than heading. This makes sense since velocity changes occur faster than heading changes during normal driving. We decided on 0.10 Hz, which gave the lowest error.

Figure 6 shows the results. The fused velocity (green in bottom panel) tracks the GPS magnitude closely while being smoother than raw GPS. We now have a velocity estimate that is both accurate and good enough for our lab.

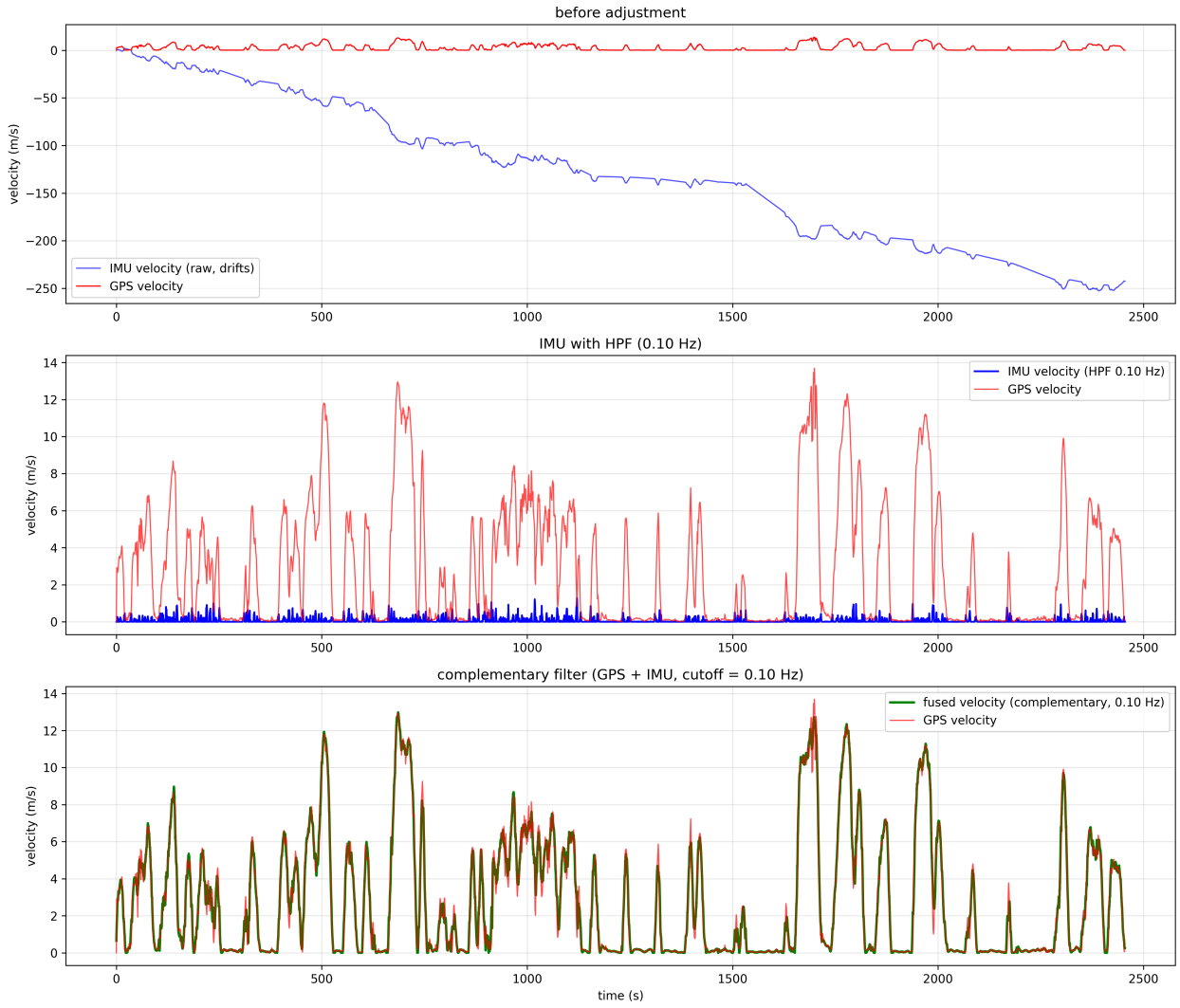

Figure 6: Velocity estimation showing the improvement from unusable raw IMU velocity (top) through filtering (middle) to final GPS-IMU fusion (bottom). The complementary filter removes drift.

5 Vehicle Model

5.1 Physics

Before we trust our velocity and heading estimates for trajectory calculation, we should check if they make physical sense. There is a simple relationship for a turning car: the lateral acceleration should equal the yaw rate times forward velocity.

$$\ddot{y}_{\text{obs}} = \omega \dot{x} \tag{10}$$

This comes from kinematics. When you are driving in a circle at speed $\dot{x}$ with angular velocity ω, centripetal acceleration pulls you inwards.

The model assumes we are driving on a flat road with no skidding (It is not yet snowing in Boston, and ice is yet to appear on the roads), and that the IMU sits at the car's center of mass. Real life is more complicated. Roads may be bad, tires may slip a bit, and our IMU isn't necessarily at the exact center of mass. But for the purpose of this lab, this will be good enough.

5.2 Our Findings

Figure 7 plots both signals. The blue line is $\omega \dot{x}$ computed from our IMU. The red line is the y-axis accelerometer reading after light filtering (we used a 1 Hz low pass filter to reduce noise).

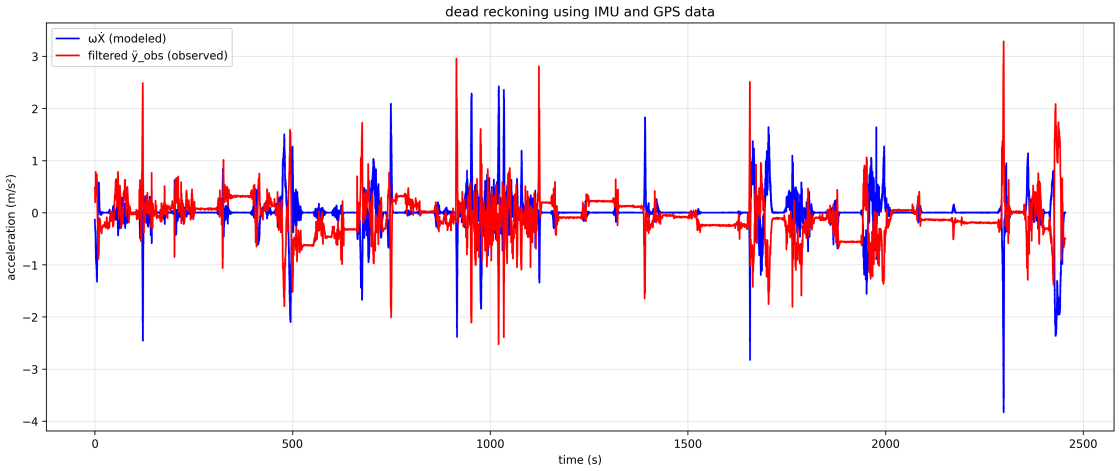

Figure 7: Comparing predicted lateral acceleration (blue) with measured acceleration (red). The graphs track together to a reasonable extent.

They track reasonably well. Both signals swing positive and negative together, reaching similar peak values around ± 3 to ± 4 m/s^2. During turns, both spike at the same time in the same direction.

There are some discrepancies when sometimes one signal spikes more than the other. The accelerometer is noisier. These are due to a variety of reasons. One of this is sensor noise. Another is our simplifying assumptions. (We ignore the IMU offset from the center

of mass, we assume perfectly flat roads, and we assume the tires never slip.) In reality, all those effects contribute a bit.

The turning radius is not perfectly constant either. It varies based on how fast we are going and how hard we are turning. Using $r_t = \dot{x}/\omega$, we can see it ranges from maybe 5-10 meters during tight turns at intersections up to over 100 meters. The model handles this variable radius fine, since we are calculating it instantaneously at each time step.

Overall, the agreement between predicted and measured lateral acceleration gives us confidence that our velocity and heading estimates are reasonable, which means we can move on to trajectory estimation.

6 Dead Reckoning Trajectory

6.1 Building the Trajectory

Now, we move on to the real purpose of the lab. Reconstructing how the car went using just IMU and heading. Dead reckoning works by breaking down the forward velocity into north and east components based on the heading, then integrating to get position:

$$v_e(t) = v_{\text{fused}}(t) \cdot \cos[\psi_{\text{fused}}(t)] \tag{11}$$

$$v_n(t) = v_{\text{fused}}(t) \cdot \sin[\psi_{\text{fused}}(t)] \tag{12}$$

Integrating these gives us easting and northing displacements:

$$x_e = \int v_e \, dt, \quad x_n = \int v_n \, dt \tag{13}$$

We used cumulative trapezoidal integration as per the instructions in the lab notebook.

6.2 Aligning with GPS

To compare our IMU trajectory with GPS trajectory, we need to get them in the same coordinate frame. GPS gives us UTM coordinates (easting and northing in meters), which we zeroed to start at the origin. The IMU trajectory starts at (0,0) too.

But starting at the same point is not enough because we also need to point the same direction. We calculated the GPS heading from its first two position measurements:

$$\psi_{\text{GPS,init}} = \text{atan2}(\Delta N, \Delta E) \tag{14}$$

Then we rotated the entire IMU trajectory by the angle difference between GPS and IMU initial headings. This alignment ensures both paths start from the same spot pointing the same direction.

We did not apply any scaling. Both trajectories use meters, and we wanted to see how accurately dead reckoning estimates distance without scaling to match.

6.3 Trajectory Comparison

Figure 8 shows the result. The GPS path (red) traces the actual route through Boston. The route is a loop covering around 2 kilometers. The IMU path (blue) follows the same general pattern as the GPS size.

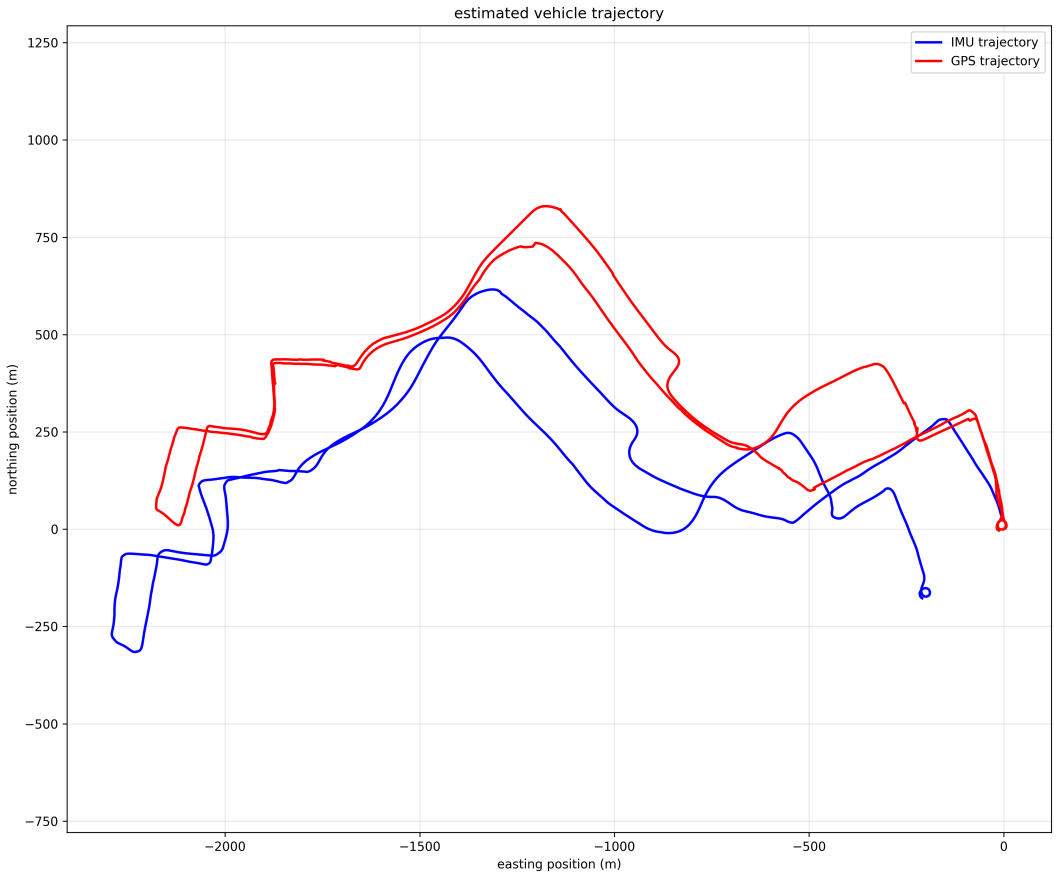

Figure 8: Trajectory comparison between GPS ground truth (red) and IMU dead reckoning (blue). Both start at the origin with aligned headings.

Despite some mismatching, the IMU trajectory gets the important points right. The sequence of turns matches GPS. When GPS shows a left turn, IMU shows a left turn. Straight segments appear as straight segments. The overall loop structure is recognizable.

6.4 How Long Can We Navigate Without GPS?

Looking at how the errors accumulate over time, the trajectories match well for the first 500-1000 seconds. During this period, the paths run nearly parallel with only small separation. By 1500 seconds, we start seeing noticeable divergence. By the end at 2500 seconds, the endpoints are separated by about 150 meters.

Based on this, I would guess that dead reckoning with our complementary filter approach could maintain accuracy within about 2 meters for roughly 5-10 minutes of driving. After that, GPS corrections would be necessary to prevent errors from growing. This is relatively

good for pure inertial navigation. Good enough to navigate through a tunnel or during temporary GNSS signal loss in urban scenarios.

7 Discussion and Observations

This lab shows how important and difficult sensor fusion is for navigation. No single sensor is good enough on its own. The magnetometer cannot give us smooth heading because it is too noisy. The gyroscope cannot give us long term heading because it drifts. GPS cannot give us high rate velocity updates. The accelerometer cannot give us velocity over time because integration errors expand to a large number.

But when we combine them properly, each sensor's weaknesses are compensated by another sensor's strengths. The complementary filter is very effective at this.

Some practical knowledge I gained was: First, calibration matters quite a bit. Without magnetometer calibration, heading errors would make dead reckoning impossible. Second, cutoff frequency selection needs to be based on actual data, and there is no "standard" values for these. Third, coordinate frame is important. The IMU's x-axis pointing backward led to negative velocity, which is not necessarily wrong (easy to correct) but needs to be understood.

The biggest issue was velocity estimation. Even with GPS fusion, our velocity was low, causing trajectory issues. Using wheel speed sensors instead of pure acceleration integration would probably help a lot. (I assume modern self driving cars such as Teslas use them) Modern cars have ABS wheel speed sensors and probably do not suffer from integration drift.

8 Conclusion

We successfully did a sensor fusion system (post-data collection, not in real time, mind you) for vehicle navigation, starting from raw sensor data and ending with trajectory estimation. The magnetometer calibration removed significant distortions (197 mG and 129 mG offsets), transforming an offset ellipse into a centered circle with only 4% radius variation.

Complementary filtering at 0.1 Hz cutoff effectively fused heading estimates from magnetometer and gyroscope. The filter eliminated 2 radians of gyroscope drift while suppressing magnetometer noise, producing smooth, drift free heading suitable for navigation. Similarly, GPS-IMU velocity fusion at 0.1 Hz cutoff reduced velocity error to 0.41 m/s RMSE.

Dead reckoning trajectory estimation captured the nuance (pun intended) of actual day to day driving, correctly reproducing the sequence and direction of turns over the 40-minute drive. Accuracy degraded due to velocity estimation issues. Position accuracy remained within 2 meters for roughly 5-10 minutes before accumulated errors became large, showing us the upper limit of inertial navigation.

The complementary filtering was effective for fusing sensors with opposing error characteristics. This approach allows navigation that degrades during GPS outages while benefiting from GPS when available.

9 Statement on Use of AI Tools

This report was prepared using the following workflow:

9.1 Original Work

All analysis, computations, interpretations, and written content in this report are my original work. The sensor fusion algorithms, complementary filter implementations, calibration procedures, and trajectory reconstructions were developed and executed by me. All figures were generated from my Python analysis scripts processing the VectorNav sensor data.

9.2 Use of Claude AI

Claude Sonnet 4.5 by Anthropic was used solely for formatting assistance in converting my original draft (draft.txt) from plain text to LaTeX format. The AI tool was not used to generate content, analysis, or scientific conclusions.

Specific usage:

- **Input provided:** The draft.txt files along with plot images (.png files)
- **Task requested:** Please convert this plain text maintaining the exact content, analysis, and tone into LaTeX format. Convert my annotated mathematical notation, add section headers, and add the figure/plot references with my included captions as needed.
- **Output received:** LaTeX formatted version of my original content in the form of code with proper document structure
- **Review process:** I have carefully cross checked the LaTeX output and confirm:
  - All technical content remained unchanged and accurate
  - Mathematical notation match my original equations
  - Figure/plot references and captions match my intended look
  - No content was added, removed, or altered beyond formatting
  - The document structure was enhanced rather than changing my narrative

Alignment with intent: The LaTeX output reflects my original work and intent. The formatting improvements (mathematical notation, figure placement, section structure) make the report more readable while preserving the integrity of the analysis and conclusion.

The original plain text draft (draft.txt) has been uploaded to the EECE5554 course repository as documentation of the pre-formatting content.

References

[1] RSN EECE5554 Lab 5 Repository. *Instructions on this lab was provided by this link.* https://github.com/ECE-RSN/lab5/tree/main

[2] Overleaf. (2024). *Online LaTeX Editor.* https://www.overleaf.com

[3] Claude Sonnet 4.5 by Anthropic. *Claude was used for help in formatting this report from a text file to LaTeX. The actual contents of this report are original work, and not a product of generative AI tools. Claude was used to visually enhance the look of this report, which would otherwise be a MS WORD/.txt file. The original draft in the form of a .txt file I made is uploaded to the EECE5554 submission repository, the prompt to the tool, the output from the tool and the statement of how I have reviewed the output for correctness and alignment with my intent is provided in the section above.* https://claude.ai/new

[4] Python 3. (2024). *Used for all analysis scripts, visualizations, and computation.* https://www.python.org/