MCEN90032 Sensor Systems Workshop 2 Project Report

Jarod Fozhun Tang

1175923

Content

Introduction:

A Global Positioning System (GPS) is a global navigation satellite system which orbits the earth and provides relatively precise information on the GPS receiver's location, velocity and time. Initially intended for military use, GPS has become a staple in modern day operations pertaining to navigation, tracking and positioning systems, especially in industries such as aviation and surveying. However, GPS does not perform well in indoor settings and therefore, an internal navigation system (INS) is employed in place of a GPS. The purpose of this report is to develop an indoor positioning system using inbuilt motion and rotation sensors within the user's mobile phone to track the user's walked path. The aim of this project is to develop an understanding of sensor calibration and the evaluating appropriate algorithms and design choices in the implementation of an indoor position system.

Methodology:

Part 1: Designing a Distance Estimator

Using the previously developed walking based pedometer from Workshop 1, three algorithms were considered in the implementation of a distance estimator and their subsequent performance was compared. These algorithms are listed below:

- Basic Walking Distance Estimation (Algorithm 1)
- Adaptive Walking Distance Estimation (Algorithm 2)
- Direct Integration from Acceleration (Algorithm 3)

Algorithm 1

Algorithm 1 is the most simplistic implementation of a distance estimator. It involves multiplying the recorded number of steps by a fixed length to estimate the walked distance. Due to its simplicity, algorithm 1 was used as a baseline to measure the performance of more sophisticated algorithms.

Algorithm 2

Algorithm 2 expands upon the premise of algorithm 1 further. For this algorithm, it is assumed that the average step length per stride is 0.46m, where a stride in the context of this report is the movement caused by pushing off from your toes of one foot and landing on your heel of the opposite foot. The walker in this report has a shoe length of 0.3m. This length is added onto the step length value after manipulating the step length per stride. The distance covered per stride is estimated based on two factors; the time in between each heel and toe strike for each step (the relative step duration) and the acceleration recorded for that step (relative acceleration). Figure 1 depicts all the heel and toe strikes being marked by the algorithm, generated from "fieldwalk3.mat".

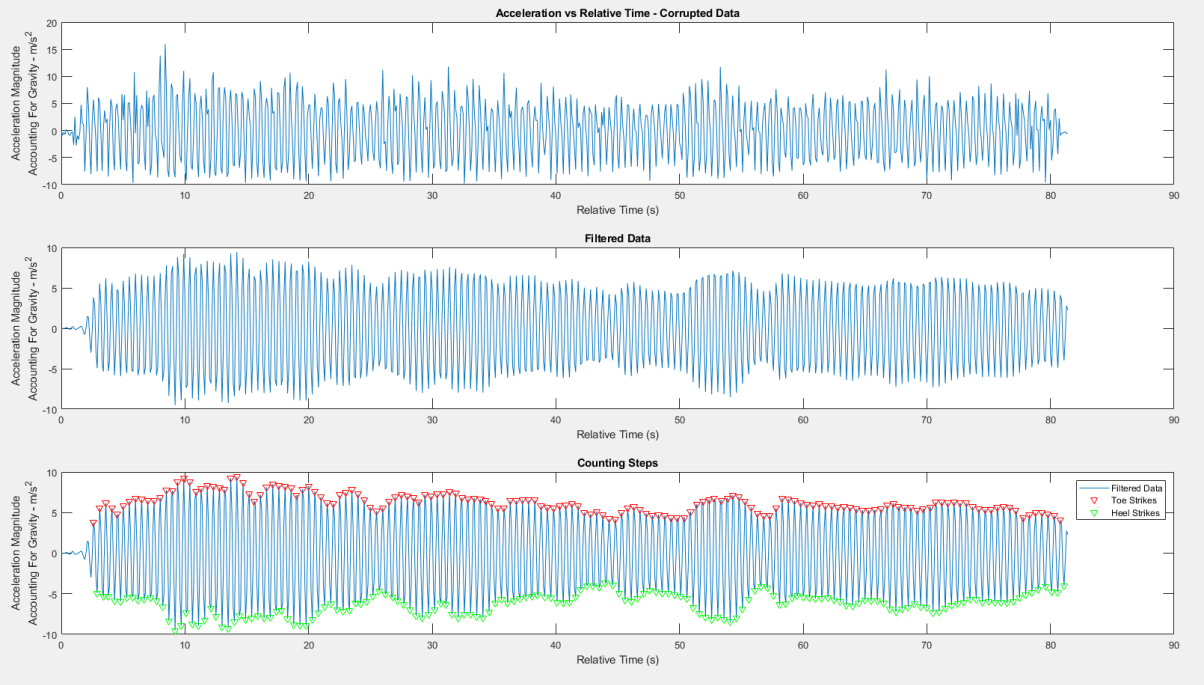

Figure 1: Heel and Toe Strikes Detected

The calculation for relative step duration is as follows. The average step duration across the parsed dataset ($Step_{AD}$) is calculated and is scaled by a constant α. A suitable value for α is determined experimentally, where the scalar associated with attaining an estimated walking distance closest to the true value is used. The variable $Step_{weight}$ describes the contribution of the step duration at a particular instance ($Step_i$) on the estimated distance covered.

$$Step_{weight} = \alpha \times \frac{Step_i}{Step_{avg}}$$

Equation 1: Relative Step Duration

The calculation for the relative acceleration per step follows a similar logic as explained above. The average acceleration for each step within the dataset (a_{avg}) is scaled by a constant β. β is determined experimentally, where again, the scalar associated with attaining an estimated walking distance closest to the true value is used. The a_{rel}, is a constant, describing the contribution of the measure acceleration at a particular instance of time (a_i) on the estimated distance covered for that step.

$$a_{weight} = \beta \times \frac{a_i}{a_{avg}}$$

Equation 2: Relative Acceleration Per Step

Once both relative step duration and acceleration have been determined, the displacement pertaining to that step is calculated. Both factors are averaged out and used to scale the average step length ($Step_{avg}$). This value is then summed with the fixed shoe length ($Shoe_{len}$) to find the distance covered from that step (s_i).

$$s_i = \frac{a_{weight} + Step_{weight}}{2} \times Step_{avg} + Shoe_{len}$$

Equation 3: Displacement Per Step

This process is repeated for every step recorded and the displacement per step is summed together to find the total distance covered.

Algorithm 3

Using the MATLAB function *cumtrapz()*, the approximate cumulative integral of the acceleration and subsequent velocity is calculated to derive an estimation for the walked distance.

Part 2: Designing a Heading Estimator

Estimate Heading using Magnetometer

Utilising the inbuilt magnetometer in the MATLAB mobile application, the corresponding magnetic field strength detected along the X, Y and Z axes of the sensor are recorded and parsed into MATLAB as .mat files. The recording of such measurements was performed in environments absence of other metals and electronics, to prevent any magnetic interference and the distortion of the dataset. Since magnetometer readings are not always reliable due to both hard and soft iron effects, the inbuilt function *magcal()* is used to calibrate the parsed dataset. To get the orientation for each recorded instance of time (magnetometer heading), the following equation is used, where $x_{corrected}$ and $y_{corrected}$ is the calibrated versions of the magnetic field strength along the X and Y axes respectively.

$$mag_{heading} = atan2(\frac{x_{corrected}}{y_{corrected}})$$

Equation 4: Magnetometer Heading

After applying the function *atan2()*, the heading gets bounded within the range $-180° \leq mag_{heading} \leq 180°$ and the heading is flipped about the X axis (in this context, the X axis represents time). To address this, the function *unwrap()* is used to unbound the function and the derived results for $mag_{heading}$ are multiplied by a factor of -1. The data is then shifted such that the initial position is at 0° by subtracting the entire dataset by the starting position. The magnetometer heading is then plotted against time. Figure 2 visualises the magnetometer heading before and after the calibration process. The figure below is generated using the dataset "rotate.mat'.

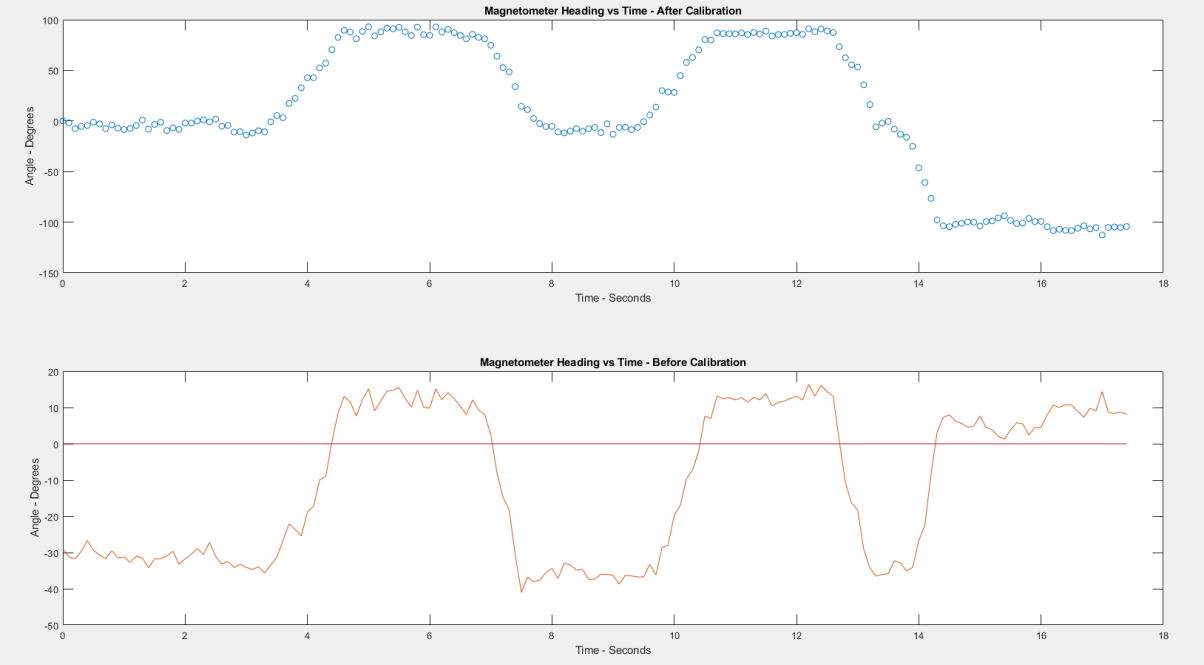

Figure 2: Magnetometer Heading Before and After Calibration

Estimating Heading using Gyroscope

Using the inbuilt angular velocity sensor in MATLAB mobile, the angular velocity along the sensor's X, Y and Z axes is recorded. Angular position is then found by numerically integrating the recorded dataset, using the function *cumtrapz()*. Since when recording the data, rotation occurs about the Z axis of the phone, the angular position derived from the angular velocity about the Z axis is plotted against time.

Part 3: Designing an Indoor Positioning System

Designing a Kalman Filter

In the context of this report, the magnetometer is good at producing absolute measurements however can be easily corrupted from electromagnetic disturbances due to both soft and hard iron effects. This was outlined above when estimating heading using magnetometer but even with these precautions in place, the elements and modules within the sensor will still cause disruptions. The gyroscope however will produce relative measurements however it requires an initial orientation and is subject to drift. Therefore, to mitigate the weakness of each sensor and emphasise their respective strengths, sensor fusion utilised to produce an overall more accurate heading. To perform sensor fusion, the Kalman filter will be used to fuse the magnetometer and gyroscope measurements to improve the heading estimation.

Firstly, an appropriate continuous-time state space model of heading estimator is derived, where $\dot{\theta}$ represents the angular velocity of the model, $u(t)$ is the measured value of angular velocity, $v(t)$ is the modelling uncertainty and $\omega(t)$ is the measurement noise. Modelling uncertainty represents random fluctuations that arise whilst taking measurements, whilst measurement noise represents errors within the sensor used.

$$\dot{\theta} = u(t) + v(t), \theta(0) \in R$$

$$y = \theta + \omega(t)$$

Equation 5: Continuous time state space model

Next, a discrete-time state space model is obtained below, depicted by equation 6, where T_s is the sampling period and k represents some recorded instance in time.

$$\dot{\theta} = \frac{\theta[k+1] - \theta[k]}{T_s}$$

Equation 6: Discrete Time State Space Model

The above equation is then rearranged for clarity. The final equation is depicted below as equation 7.

$$\frac{\theta[k+1] - \theta[k]}{T_s} = u[k] + v[k]$$

$$y[k] = \theta[k] + \omega[k]$$

Equation 7: Rearranged Discrete Time State Space Model

The above process is performed twice, to derive the measurement and prediction models. Afterwards, the sensor fusion process can be segmented into two stages, a prediction and update stage. The prediction part consists of equations 5 and 6 whilst the update part consists of equations 7, 8 and 9. The matrices A, B and C represent the state transition, input control and measurement matrix respectively. The variables R and Q represent the measurement model noise and prediction model noise respectively and dictates the weight of both the measurement and prediction model on the updated posterior estimate. These values are chosen based on the variance of the sensor used. This is calculated by recording both magnetometer and gyroscope data whilst the sensor is stationary, then using the MATLAB inbuilt function *var()* to calculate the variance.

$$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_k$$

Equation 8: Prior State Estimate

$$P_k^- = AP_{k-1}A^T + Q$$

Equation 9: Error Covariance

$$K_k = \frac{P_k^- C^T}{CP_k^- C^T + R}$$

Equation 10: Kalman Gain

$$\hat{x}_k = \hat{x}_k^- + K_k(y_k - C\hat{x}_k^-)$$

Equation 11: Updated Posterior Estimate of State

$$P_k = (I - K_k C)P_k^-$$

Equation 12: Updated Error Covariance

Using the above equations, the optimal heading estimation at each recorded instance of time is calculated recursively. This data is then plotted against both the magnetometer and gyroscope headings, as well as a baseline dataset to compare the performance of the Kalman filter. The baseline dataset comes from MATLAB's inbuilt orientation sensor, which directly measures the sensor's orientation along the X, Y and Z axis.

Designing a Path Plotter

The subsequent position relative to the starting position over time is calculated using both the optimal heading estimation and a chosen distance estimation algorithm from the ones described earlier. This is calculated using the two equations below, where x_k and y_k is the position of the sensor at instance k, x_{k-1} and y_{k-1} represent the previous position of the sensor, Δx represents the distance covered at instance k and θ is the orientation at instance k. Since the below formula depends on the previously calculated position, the initial position is set as $(0, 0)$ and the calculations below occur for $k \geq 2$.

$$x_k = x_{k-1} + |\Delta x| \times \cos(\theta[k])$$

Equation 10: X Position of Sensor at the kth Recorded Instance of Time

$$y_k = y_{k-1} + |\Delta y| \times \sin(\theta[k])$$

Equation 11: Y Position of Sensor at the kth Recorded Instance of Time

Once the position along the X and Y axis was calculated, the results were plotted on a X-Y graph. Similar plots using magnetometer, gyroscope and GPS results were also plotted on the same graph to act as baseline performance indicators and help assist in performance analysis.

Results and Discussion

Evaluating Performance of Different Distance Estimators

Table 1: Algorithm 1 Performance

Dataset	Filename	Estimated Distance (m)	True Distance (m)	Percentage Error
1	capstonewalk_12steps_8.2m.mat	8.36	8.20	1.95%
2	capstonewalk2_22steps.mat	15.20	15.33	0.85%
3	uni47steps_run2.mat	35.72	37.48	4.70%

Table 2: Algorithm 2 Performance

Dataset	Filename	Estimated Distance (m)	True Distance (m)	Percentage Error
1	capstonewalk_12steps_8.2m.mat	8.61	8.20	5.00%
2	capstonewalk2_22steps.mat	15.66	15.33	2.15%
3	uni47steps_run2.mat	36.80	37.48	1.81%

Table 3: Algorithm 3 Performance

Dataset	Filename	Estimated Distance (m)	True Distance (m)	Percentage Error
1	capstonewalk_12steps_8.2m.mat	42.46	8.20	417.80%
2	capstonewalk2_22steps.mat	160.24	15.33	945.27%
3	uni47steps_run2.mat	35035	37.48	9.34×10^4%

In this performance evaluation, algorithm 1 is used a baseline to gauge the practicality of implementing more complex distance estimating algorithms due to algorithm 1's simplicity. Comparing the tabulated data above, algorithm 3 is clearly unsuitable to be used as a distance estimator due to the incredibly high percentage error relative to algorithm 1. Furthermore, it is also noted that the percentage error of algorithm 3 appears to increase exponentially as the walked distance increases. This can be attributed to a phenomenon known as drift, where errors accumulate over time, resulting in an ever increasing error, causing the calculated value to drift away from the true value.

Both algorithm 1 and algorithm 2 performed well against the parsed datasets, with algorithm 1 outperforming algorithm 2 for datasets 1 and 2 whilst algorithm 2 performed better against dataset 3. It is observed that the average performance across the three datasets of algorithm 1 in comparison to algorithm 2 is marginally better, with algorithm 1 having an average percentage error of 2.50% whilst algorithm 2 has an average percentage error of 2.99%. However, given the limited number of datasets available and the potential to further tune the values of α and β to increase

performance, in the implementation of an indoor position system, algorithm 2 was chosen to be used.

Magnetometer and Gyroscope Performance Evaluation

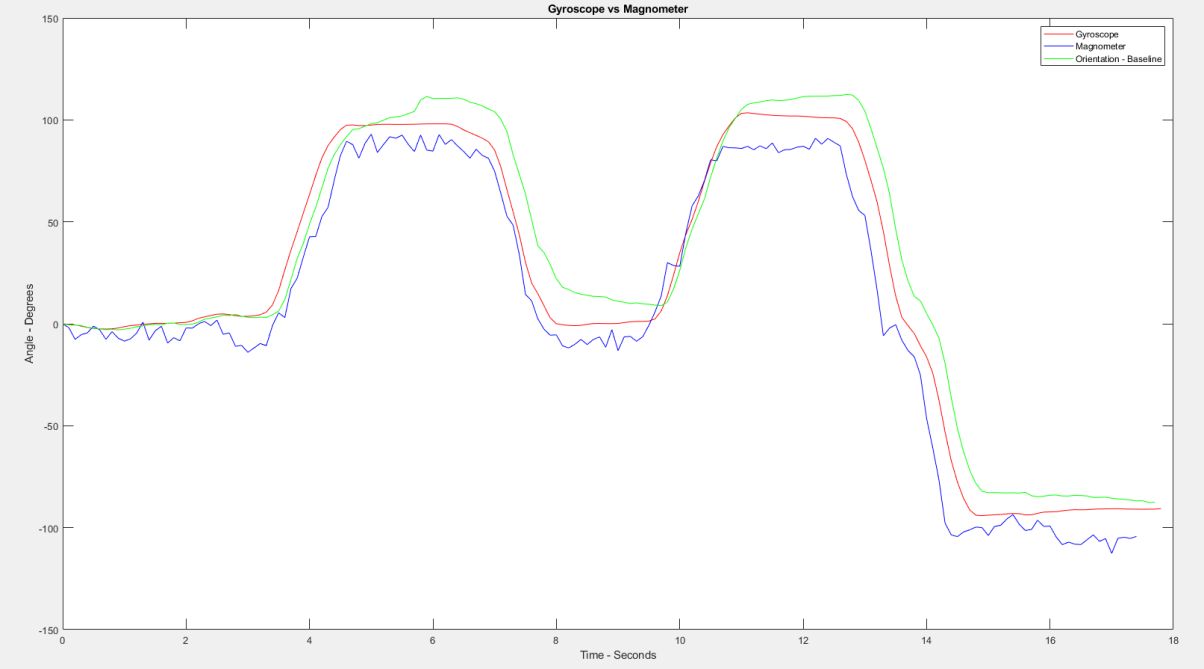

Figure 3: Heading Performance

The above figure from “rotate.mat” depicts the sensor performing the following rotations: one 90° anticlockwise turn, one 90° clockwise turn, one 90° anticlockwise turn followed by one 180° clockwise turn. From initial observation, both the gyroscope and magnetometer plots roughly match the shape plotted by the orientation plot.

Comparing the magnetometer and orientation headings, the magnetometer plot exhibits many irregular and sharp changes when the sensor is stationary, however behaves similarly to the orientation heading during the time the sensor is rotated either clockwise or anticlockwise. As mentioned previously in the methodology, the erratic behaviour of the magnetometer heading when the sensor is stationary is most likely due to soft and hard iron effects corrupting the magnetometer measurements.

Comparing the gyroscope and orientation headings, the gyroscope headings are quite smooth relative to the magnetometer, mimicking the behaviour and position of the orientation heading both when the sensor is stationary and when the sensor is rotating. Additionally, the gyroscope heading does not appear to deviate from the orientation heading as much as the magnetometer heading. A potential reason for this could be that the process of integration itself acts as low pass filter, smoothing out high frequency noise to some extent. However due to phenomenon known as random walk as well as the integration of any bias contained within the measurements, the gyroscope heading overtime will slowly drift away from the true orientation. Based on the above figure however, this disadvantage does not affect the reliability of the gyroscope heading over the recorded time, as the plot still matches the orientation heading quite well, however

it is an issue worth considering if sensor orientation was to be recorded over a longer period of time.

Kalman Filtered Heading Estimation

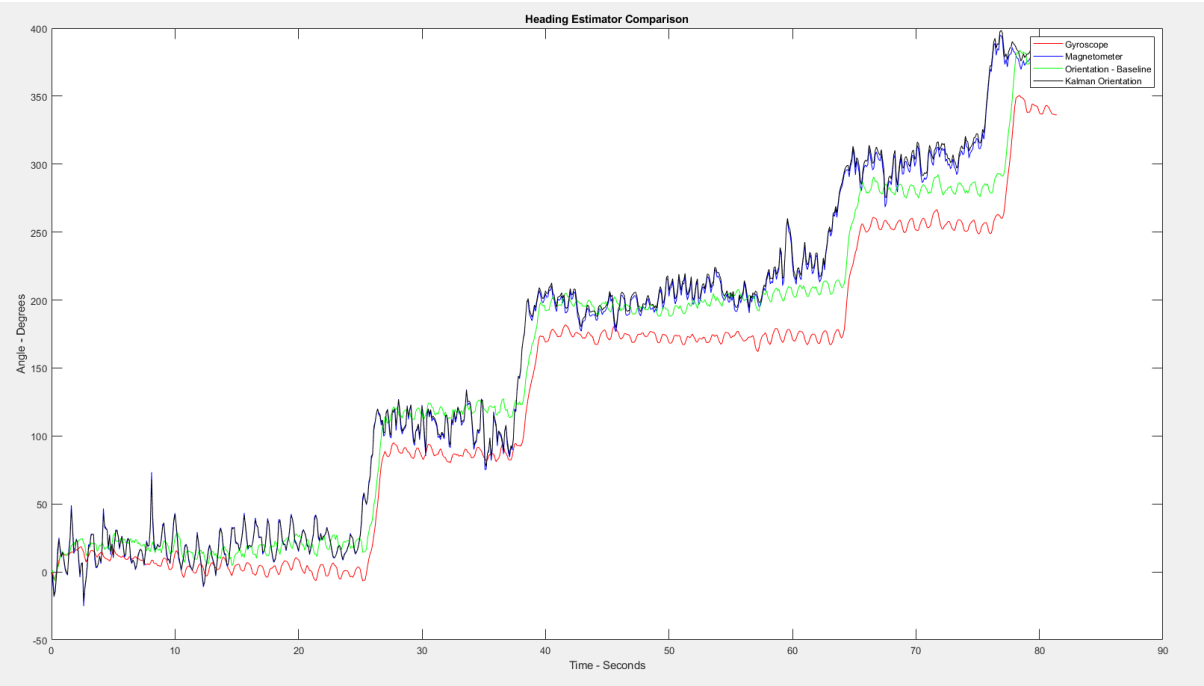

Figure 4: Kalman Filter Heading Comparison

The above figure is taken from the file "uni47steps_run2.mat". As depicted in figure 4, the Kalman filtered heading estimation maps quite close to the magnetometer heading as opposed to the gyroscope heading. This can be attributed to the values set for both the magnetometer model noise (R) and the prediction model noise (Q). Using the method outlined above to measure sensor variance, R was found to be 0.1697 and Q was found to be 6.819. Since the measurement model noise is small, the measurement is trusted more and will contribute more to the calculation $\hat{x}_k$ more than the prior state estimate does, explaining the observation made regarding the Kalman filtered heading behaviour. This means that there is a large amount of variance in the gyroscope measurements as opposed to the magnetometer measurements, meaning that the gyroscope measurements are not as reliable as the magnetometer measurements.

The values of A and C were both set to 1 whilst B was set to the sampling period (T_s). The values of A and C were set as such values since there was no need to transform the magnetometer and gyroscope heading values. The value of B was based on the derivations performed in the methodology.

The error covariance P quantifies how much the state estimate deviates from the true state. This value was set to the sum of variance of the sensors used ($P = Q + R$), however for the dataset used, it was observed that P converges to value of 7.8947 relatively quickly.

Overall, the Kalman filtered heading estimation follows the trend depicted by the orientation heading.

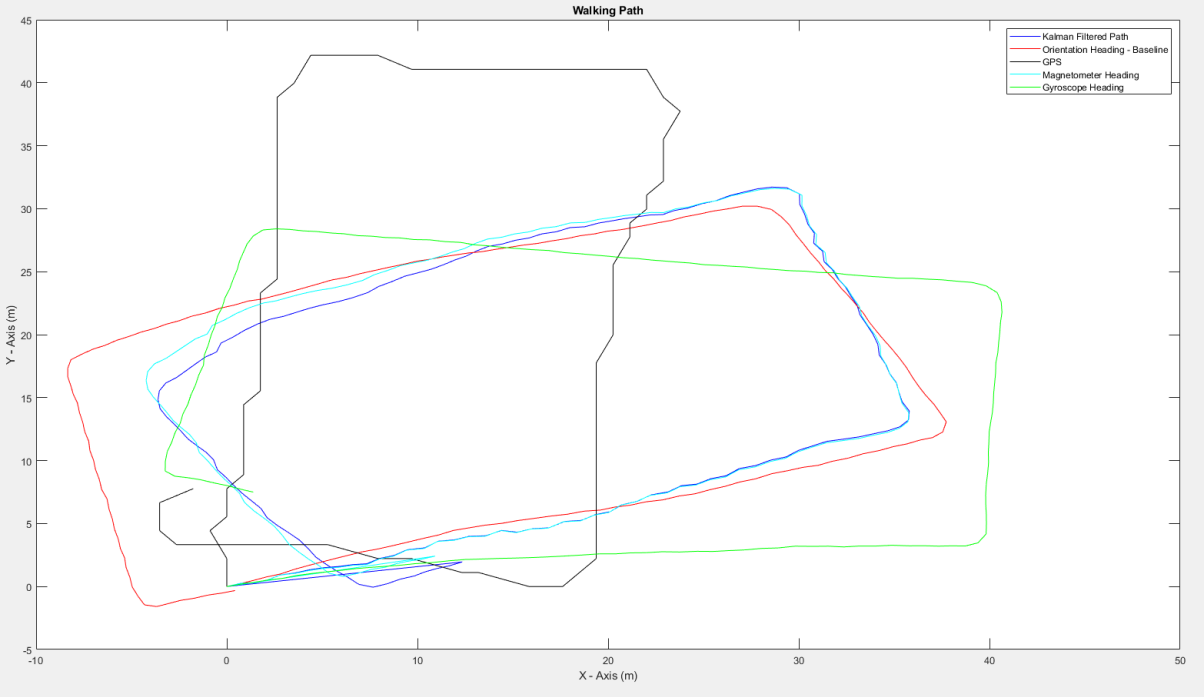

Figure 5: Plot of Walked Path

Figure 5 is taken from the file "uni47steps_run2.mat". Comparing the Kalman filtered heading estimation to the GPS measurements, it is obvious that the Kalman filtered heading estimation is superior in its precision, as the Kalman filtered heading estimation maps the true path taken relatively well. Additionally, the plotted GPS data is jagged in comparison to the rest of the plots. This is because the GPS data is sampled at 1Hz, in comparison to the sampling frequency of the IPS (10Hz). Both the GPS and IPS plot a rectangular path, which matches the true path taken during the walk, however, as stated prior, the IPS maps substantially closer to the true path.

It is worth noting that due to the error discrepancy between the magnetometer and gyroscope discussed earlier, that the indoor positioning system (IPS) maps quite closely to the magnetometer plotted path.

Limitations and Considerations

The developed IPS generally performed better than the GPS and reliably illustrated the true path walked. However, the designed IPS is incredibly sensitive to noise and could potentially be unreliable if only short number of steps is detected or taken. This is because the IPS depicts a change in position only if a step is detected. If the data pertaining to the orientation at that instance of time is unreliable, and if the dataset is not large, then the error will have a greater impact on the depicted path and corrupt the IPS results as there will not be enough data present to reduce the visual impact of that data. In this report, this potential liability was accounted for, by generating datasets with over 100 steps.

Another potential limitation is the IPS' tendency to map closely to the magnetometer results. This limitation stems from the high variance present in the gyroscope and limits the usefulness of the Kalman filter, as the sensitivity of the magnetometer could have been further mitigated if a more precise gyroscope is used.

Conclusion

An indoor positioning system was developed using inbuilt motion and rotation sensors within the user's mobile device to track the walked path. The developed indoor positioning system through developing appropriate estimation algorithms, sensor calibration and sensor fusion, was observed to be considerably more precise than the GPS measurements and therefore, at least based on the parsed datasets used to measure performance, is more reliable than a GPS in an indoor setting. Furthermore, this report demonstrates However, given the limited number of datasets used, further testing is recommended to confidently gauge the precision of the indoor positing system.

References

1. au.mathworks.com. (n.d.). *Cumulative trapezoidal numerical integration - MATLAB cumtrapz - MathWorks Australia*. [online] Available at: https://au.mathworks.com/help/matlab/ref/cumtrapz.html [Accessed 6 May 2023].

2. au.mathworks.com. (n.d.). *Magnetometer calibration coefficients - MATLAB magcal - MathWorks Australia*. [online] Available at: https://au.mathworks.com/help/nav/ref/magcal.html [Accessed 6 May 2023].

3. au.mathworks.com. (n.d.). *Magnetometer Calibration - MATLAB & Simulink - MathWorks Australia*. [online] Available at: https://au.mathworks.com/help/fusion/ug/magnetometer-calibration.html.

4. Kyes, J. (2020). *What Is GPS?* [online] Geotab. Available at: https://www.geotab.com/au/blog/what-is-gps/.

5. www.youtube.com. (n.d.). *Optimal State Estimator Algorithm | Understanding Kalman Filters, Part 4*. [online] Available at: https://www.youtube.com/watch?v=VFXf1lIZ3p8&list=PLn8PRpmsu08pzi6EMiYnR-076Mh-q3tWr&ab_channel=MATLAB [Accessed 6 May 2023].

6. Zhao, Q., Zhang, B., Wang, J., Feng, W., Jia, W. and Sun, M. (2017). Improved method of step length estimation based on inverted pendulum model. *International Journal of Distributed Sensor Networks*, 13(4), p.155014771770291. doi:https://doi.org/10.1177/1550147717702914.