Part-A - Data Collection for Dead-Reckoning:

- The data collected and the drivers used for this Lab were taken from yash mehwada.as he was the one who collected the data from his own driver code
-
- *The analysis of the data was done in python (magnetometer soft iron and hard iron corrections) and the analysis scripts are added in the folder] scripts are added.

Part-B - Analysis of the data collected:

We have collected IMU and GPS data which had a lot of noise and errors. The main aim of the lab is to reduce those errors/noise and plot that data for comparing with other alternatively calculated data(eg. trajectory from imu data and gps data). Using data from multiple sensor help us strengthen the data accuracy and reduce the noise , so by comparing same data by various means we can reduce the noise and better estimate the quantity which is represengtrd by the data

Q1. How did you calibrate the magnetometer from the data you collected? What were the sources of distortion present, and how do you know?

The plot in figure-1 shows the comparison between Raw Magnetometer data and corrected Magnetometer data after removing the hard iron and soft iron errors.

1. *Magnetometers measure the strength and direction of the magnetic field at their location. However, the Earth's magnetic field is not perfectly uniform, and there can be sources of distortion that affect the measurements of the magnetometer. These sources of distortion include:*

Hard iron correction:

- Hard-iron distortion is caused by materials that display a constant, additive field to the earth's magnetic field, resulting in a constant additive value to the output of each magnetometer axis.

- As observable from Fig-1 the uncorrected magnetic field plot(red circle) has a center at (0.225, -0.025) so to correct that translation we calculate the offset of each axis and subtract it from them. Considering X and Y to be our raw data from magnetic_x and magnetic_y.

- X offset = [maximum(x) + minimum(x)]/2 Y offset = [maximum(y) + minimum(y)]/2

Soft iron correction:

- Soft-iron distortion is created by material that alters or distorts a magnetic field, but does not necessarily create a magnetic field, and hence is not additive. soft-iron distortion is affected by the material's orientation relative to the sensor and the magnetic field. This distortion is found when an ellipse with an angular rotation is recognized. The following is a procedure for eliminating moderate iron distortion.
- We must estimate the length of the primary axis by eyeballing it or using an elliptical fit.

- find the maximum and minimum of x and y This gives us the distance of the farthest point (x, y) from the center (0,0). The distance r may be calculated using

$$r = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Θ = arcsin(y1/r)

We use the rotation matrix to rotate the ellipse to bring it to position where major axis and minor axis are aligned to the x and y cartesian line.

$$R = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}$$

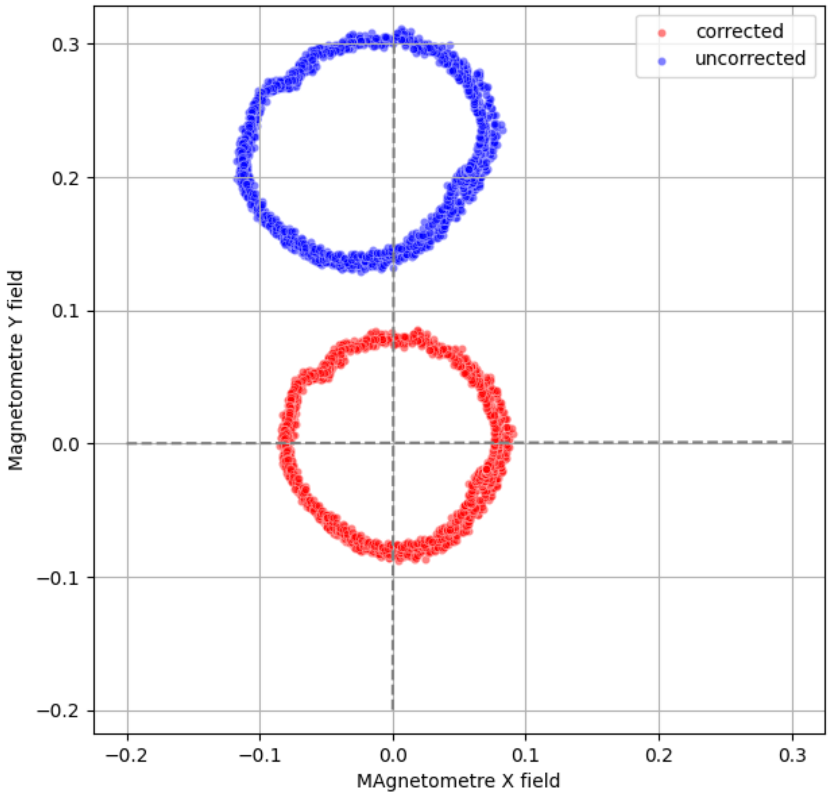
V1 is the product of rotation matrix R and vector components (v) of mag_x,mag_y.

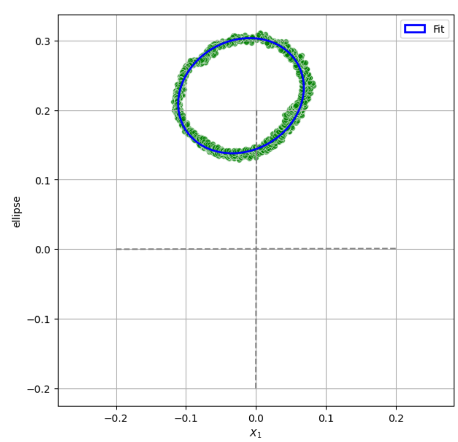
As a gist of how to remove the hard and soft iron errors we have to perform

Subtracting the following offsets from respective raw magnetic-x and magnetic-y values
X offset = [maximum(x) + minimum(x)]/2 Y
offset = [maximum(y) + minimum(y)]/2

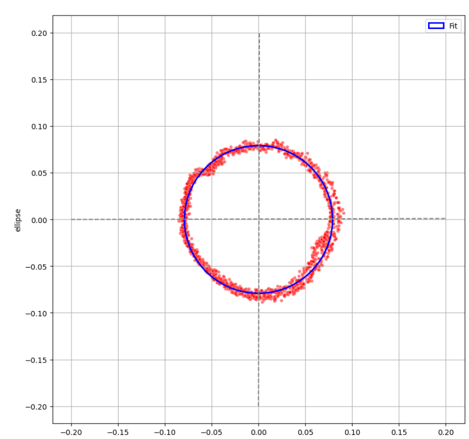
Followed by rotation and scaling with the scale factor which is the ratio of length of major axis to minor axis.

FIGURE 1 (comparing Hard iron vs soft iron plot of magnetic field)





Before fitting



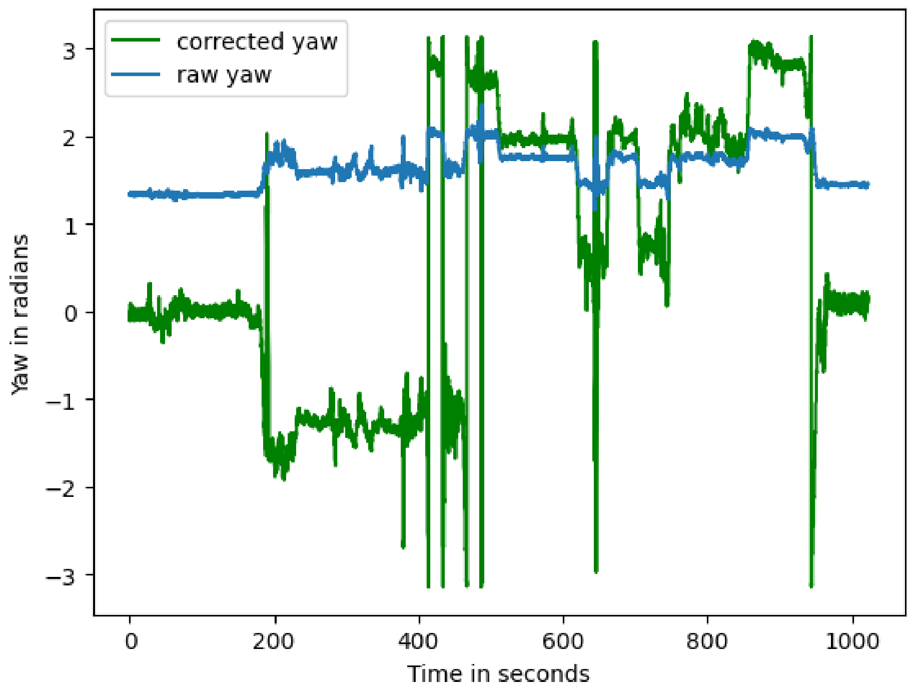
After fitting

The above plot shows the raw scattered plot(red circle) to be translated to a circle translated, rotated and scaled with origin as shifted which is the desired plot after correcting the hard and soft iron errors.

Q2 2. How did you use a complementary filter to develop a combined estimate of yaw? What components of the filter were present, and what cutoff frequency(ies) did you use?

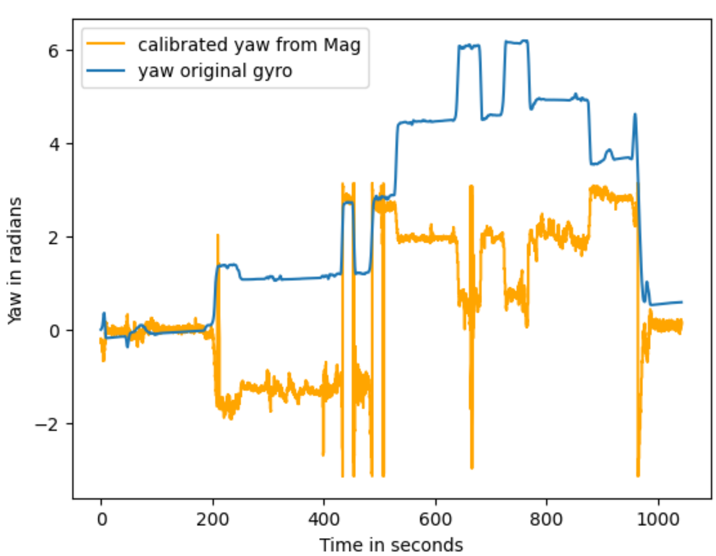
In order to use a complementary filter to find a combined estimate of yaw the following steps were followed:

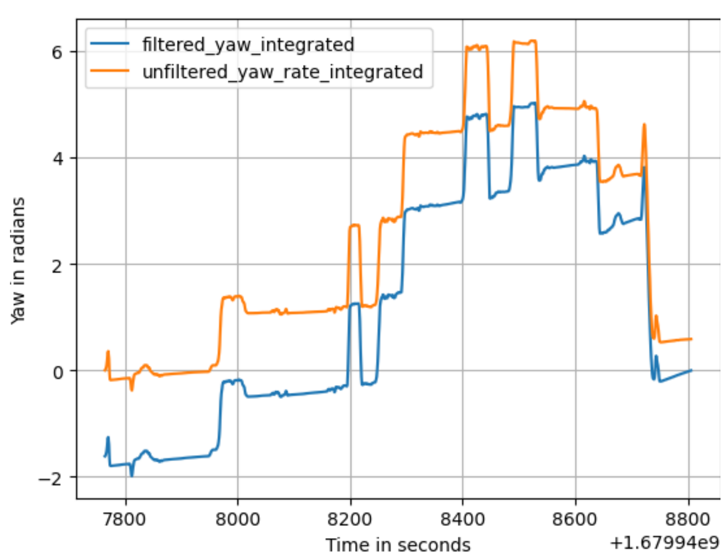
1. First the raw yaw is calculated from the magnetic x and y values using the arctan(-y/x). Where the y and x are the corrected Magnetic_Y and Magnetic_X.



2. Now this corrected magnetic yaw is unwrapped and passed through a low pass filter using, y = lowpass(x,fpass,fs) where x is the corrected magnetic yaw that has been sampled at a rate of 40 hertz. fpass is the passband frequency of the filter which was set to 0.05 hz

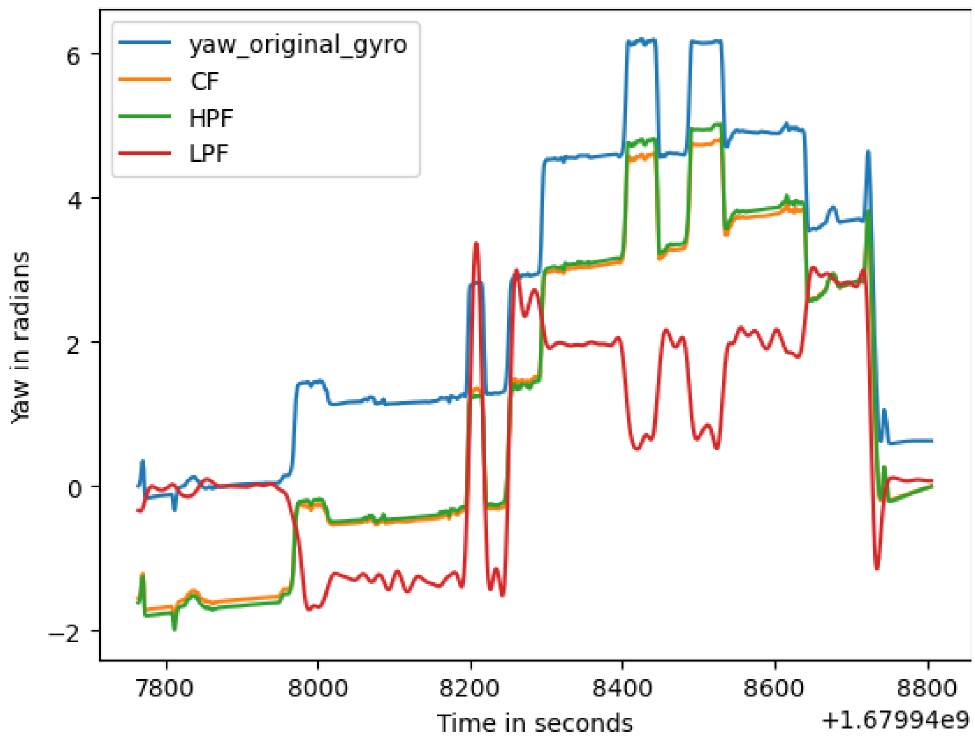
3. The gyro_z which is the angular velocity in z is integrated and the integrated value is passed through the high pass filter which has been sampled at a rate of 40hz and the pass band frequency of the filter is set to 0.000105.





1. Once the lpf and hpf are obtained, the sum of both lpf and hpf has given the complementary filter output.

All the graphs of LPF, HPF, CPF are plotted and plotted in the figure below



Q3. Which estimate or estimates for yaw would you trust for navigation? Why?
There were two reasons for considering the yaw from complementary output as better for navigation than the yaw computed by IMU.

- It is obtained as a sum of both the low pass and high pass filters,signal smoothening due to low-pass filter and signal sharpening due to high-pass filter are observed to the complementary filter output.

- When trajectory paths were estimated by considering both gyro integrated Yaw and complementary filter output yaw for velocity estimation and there by for path estimation separately and comparing with the gps trajectory, promising results were observed when complementary filter output is considered

Q4. What adjustments did you make to the forward velocity estimate, and why?
Before making any adjustments or corrections to the acceleration from IMU (Fig-7) by directly integrating it the obtained velocity shows high positive values and even the sections of the plot which were at rest/motion-less depict velocity in forward direction which isn't actually true hence bias was removed from the acceleration and then integrated. The resultant velocity estimate from accelerometer (Fig-8) appears to be in coherence with the gps velocity

plot calculated by v=s/t (v is the gps velocity, s is the displacement calculated considering utm easting and utm northing, t is the time interval). The main adjustment that were made to get correct estimate of the velocity were:

- Passing the raw acceleration from the low pass filter and then removing the bias subtracting the bias's from the data after passing it from low pass filter which helped to remove high spikes noise from the data,

- In this case we wanted to find regions of graph at rest (no motion) and calculating the mean and subtracting it from the remaining data and repeating this process for all stationary regions in the acceleration plot and then integrate that acceleration to obtain velocity from accelerometer after adjustment.

- We extracted the time window in which the velocity of the gps is zero and made the velocity of the imu also zero at that time frame also did a detrend of the signal so that it becomes flat and follows the velocity estimate of the gps as shown in (fig-9) and also removing the negative velocities from the estimation

Figure -9

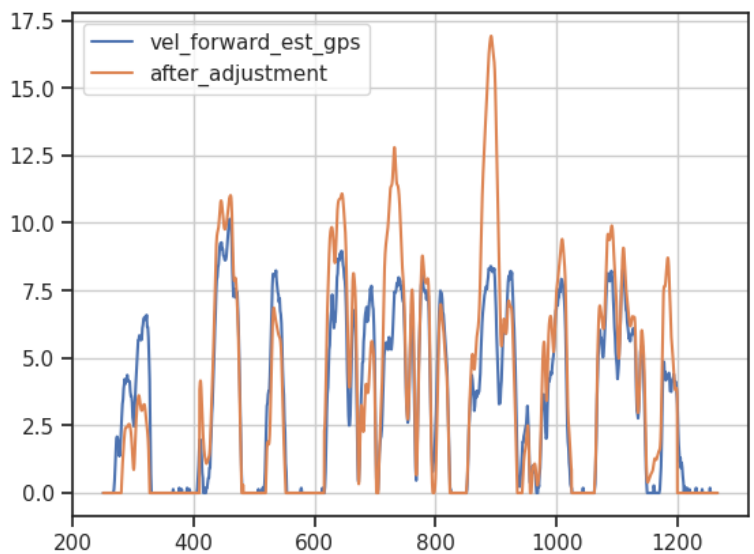


Figure-8

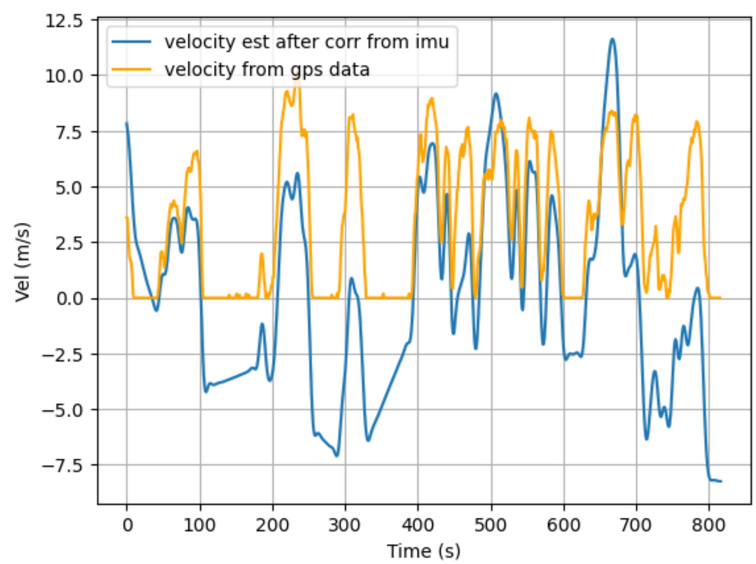
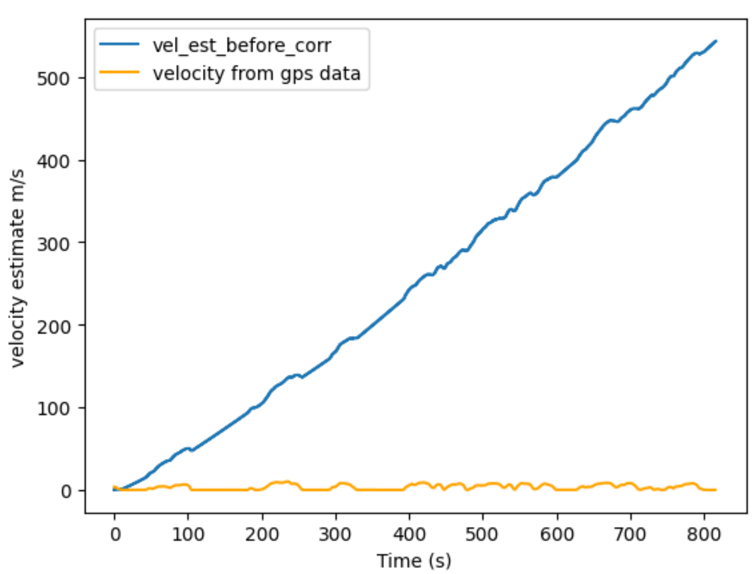


Figure - 7

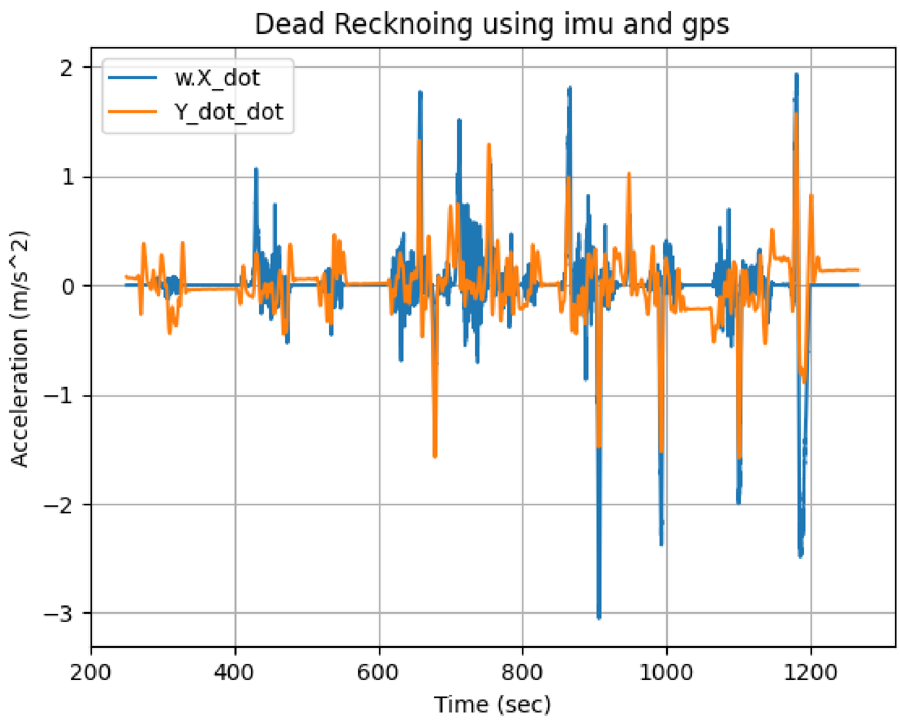


Q5. What discrepancies are present in the velocity estimate between accel and GPS. Why?

The only discrepancy observed in velocity estimated from IMU acceleration is the presence of negative velocity at certain regions of the graph which isn't observed in the velocity estimated from GPS. The one main reason to take the velocity estimate from gps to be promising is the perfect zero velocity at points where there is halt or a stop in the motion of the car. The velocity estimate from IMU shows some error which needs a lot more high level filtering, but would be helpful in approximate indoor navigation.

Q6 6. Compute ωX and compare it to $\ddot{y}$obs . How well do they agree? If there is a difference, what is it due to?

Fig-8 ωX and $\ddot{y}obs$ plotted together



The ωX and $\ddot{y}o$bserved are obtained from the formula

$$\ddot{x}_{obs} = \ddot{X} - \omega\dot{Y} - \omega^2 x_c$$

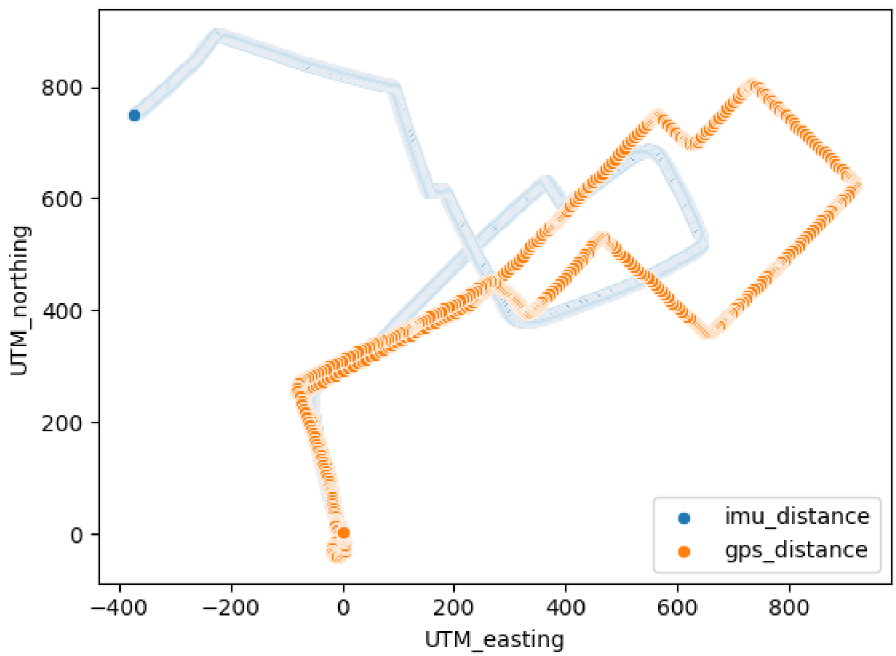
$$\ddot{y}_{obs} = \ddot{Y} + \omega\dot{X} + \dot{\omega}x_c$$

Notation taken for the position of the center-of-mass (CM) of the vehicle by (X,Y,0) and its rotation rate about the CM by (0,0,ω). We denote the position of the inertial sensor in space by (x,y,0) and its position in the vehicle frame by (xc,0,0) and assuming that $\dot{Y}$ = 0 (that is, the vehicle is not skidding sideways) and ignore the offset by setting xc = 0 (meaning that the IMU is on the center of mass of the vehicle, i.e. the point about which the car rotates). Then the first equation above reduces to $\ddot{X}$= $\ddot{x}obs$. So now by integrating $\ddot{X}$ we get $\dot{X}$ which is the velocity of the vehicle w.r.t the center of mass, we can see that the noise in y bserved is much greater than to that of the ωX. This is because when integrating the error also gets integrated with the original data causing more noise. But for ωX, the X velocity obtained when multiplied with ω will make the angular rotation compensation and reduces the error .

- As we can see that the general trend is being followed but w.X has a lot of errors that are high frequency we can use a low pass filter to get rid of them and get a more accurate correlation between w.X. and Y.. , this difference is because xc is not zero and also it can be due to the vibrations in the car as well as having the orientation of the sensor slightly off.

Q7 . Estimate the trajectory of the vehicle (xe,xn) from inertial data and compare with GPS. (adjust heading so that the first straight line from both are oriented in the same direction). Report any scaling factor used for comparing the tracks

The trajectories xe and xn are calculated by integrating the velocities estimated from gps and imu.Respective graphs are plotted. A very good trajectory was plotted with gps data, nevertheless trajectory using the complementary filter output was coinciding with gps data with slight variation.



Q8 . Given the specifications of the VectorNav, how long would you expect that it is able to navigate without a position fix? For what period of time did your GPS and IMU estimates of position match closely? (within 2 m) Did the stated performance for dead reckoning match actual measurements? Why or why not?

Given the specifications, I don't expect it to perform that well as there is no feedback to get it back on track once an error is generated. Even though GPS and IMU estimates of position match closely for just a few seconds under a minute we can see that if position fix is applied we would have got almost identical results as the general trend is followed throughout the trajectory.

Q9. Estimate xc and explain your calculations (bonus up to 100%)

Xc calculations:
Using the equations in the below figure

$$v^U_{sensor} = v^U_{car} + \omega \times \rho^R_{sensor}$$

$$a^U_{sensor} = a^U_{car} + \dot{\omega} \times \rho^R_{sensor} + \omega \times (\omega \times \rho^R_{sensor})$$

$${}^R_T R^T a^R_{sensor} = a^U_{car} + \dot{\omega} \times \rho^R_{sensor} + \omega \times (\omega \times \rho^R_{sensor})$$

$${}^R_T R^T \begin{bmatrix} a_{imux} \\ a_{imuy} \\ 0 \end{bmatrix} = \begin{bmatrix} a_{utmx} \\ a_{utmy} \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \dot{\omega} \end{bmatrix} \times \begin{bmatrix} x_c \\ 0 \\ 0 \end{bmatrix}^R + \begin{bmatrix} 0 \\ 0 \\ \omega \end{bmatrix} \times (\begin{bmatrix} 0 \\ 0 \\ \omega \end{bmatrix} \times \begin{bmatrix} x_c \\ 0 \\ 0 \end{bmatrix}^R)$$

$$\begin{bmatrix} a_{imux} \\ a_{imuy} \\ 0 \end{bmatrix}^R = {}^R_U R \begin{bmatrix} a_{utmx} \\ a_{utmy} \\ 0 \end{bmatrix}^U + \begin{bmatrix} 0 \\ \dot{\omega} x_c \\ 0 \end{bmatrix}^R + \begin{bmatrix} -\omega^2 x_c \\ 0 \\ 0 \end{bmatrix}^R$$

$$\begin{bmatrix} a_{imux} \\ a_{imuy} \\ 0 \end{bmatrix}^R - {}^R_U R \begin{bmatrix} a_{utmx} \\ a_{utmy} \\ 0 \end{bmatrix}^U = \begin{bmatrix} -\omega^2 \\ \dot{\omega} \\ 0 \end{bmatrix}^R x_c$$

Then solving for Xc the obtained value is approximately **Xc = 0.5891492207064902**