Intel - Take home assignment - Documentation - Karthik Ram

Assignment Summary

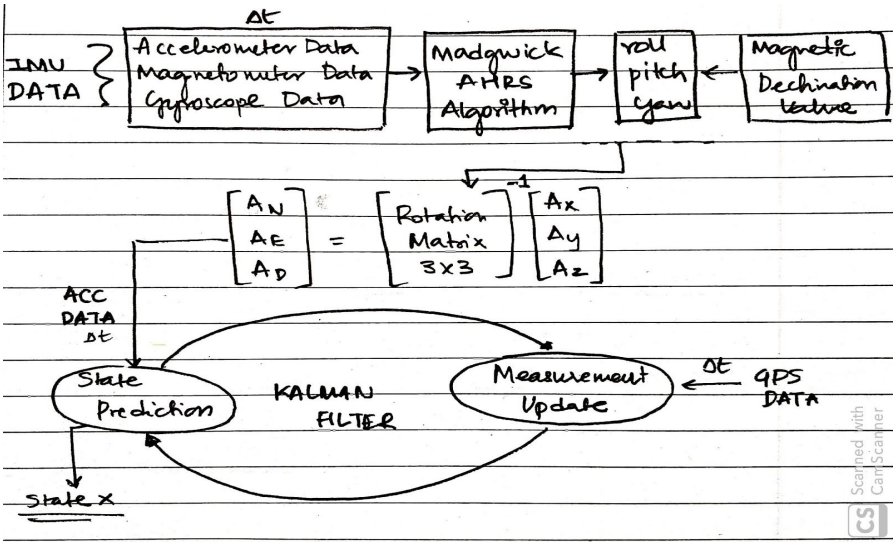
The task was to estimate accurately the trajectory of a robot carrying a smartphone mounted in a landscape mode with noisy timestamped GPS and IMU data and Images from the phone.

Tasks Completed according to the requirements sheet

1. Read sensor data from text files into python lists for processing data
2. Performed linear Interpolation of GPS, Accelerometer, Magnetometer and Gyroscope data, taking GPS timestamp as a reference. The Accelerometer, Magnetometer and Gyroscope values were further processed to calculate Absolute East-North-Down Absolute Acceleration Vector in order to fuse the data with GPS
3. Implemented Kalman filter on 30000 interpolated data points between Acceleration and GPS data to obtain a better-estimated trajectory
4. Plotted the estimated trajectory on Google maps using **gmplot** package
5. Implemented function to return distance (in meters) and bearing data between any two frames of the estimated trajectory

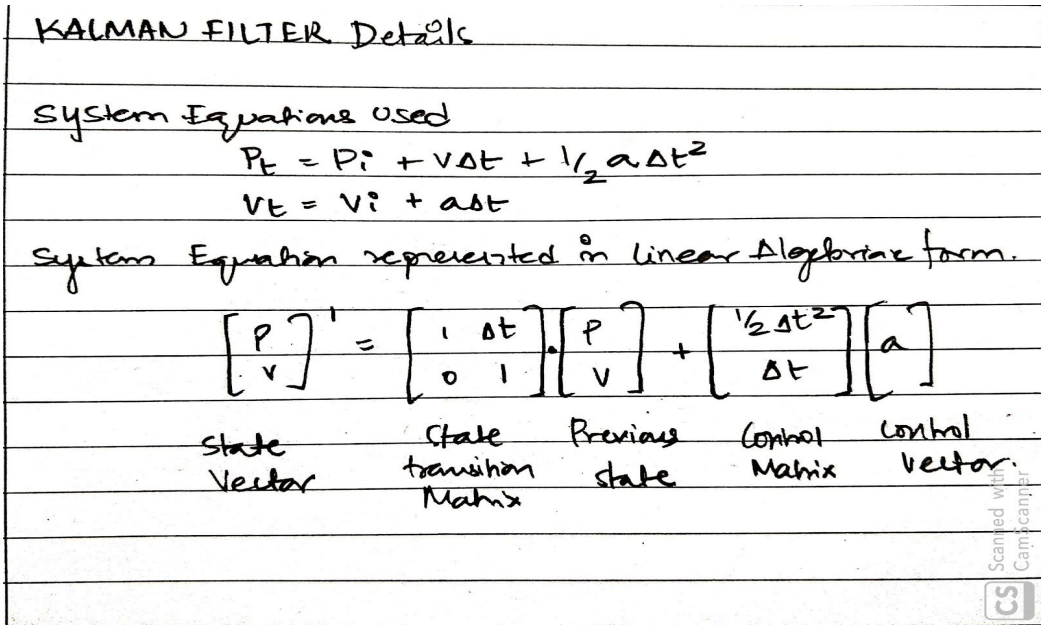
Assignment Notes

Python was the language of choice to solve the assignment. The data in the text files were read into python lists The numpy linspace function and interp function were used to interpolate data.
In order to fuse GPS and IMU data, I had to first convert the raw IMU data to absolute Acceleration in terms of true North-East-Down coordinates, for I which madgwickAHRS algorithm was used, which converts Accelerometer, Magnetometer and Gyroscope values to [roll, pitch, yaw] format. After adding the magnetic declination to the yaw values, a [3X3] rotation matrix was formed and the inverse of it was multiplied to the original Accelerometer data to get Absolute Acceleration values.

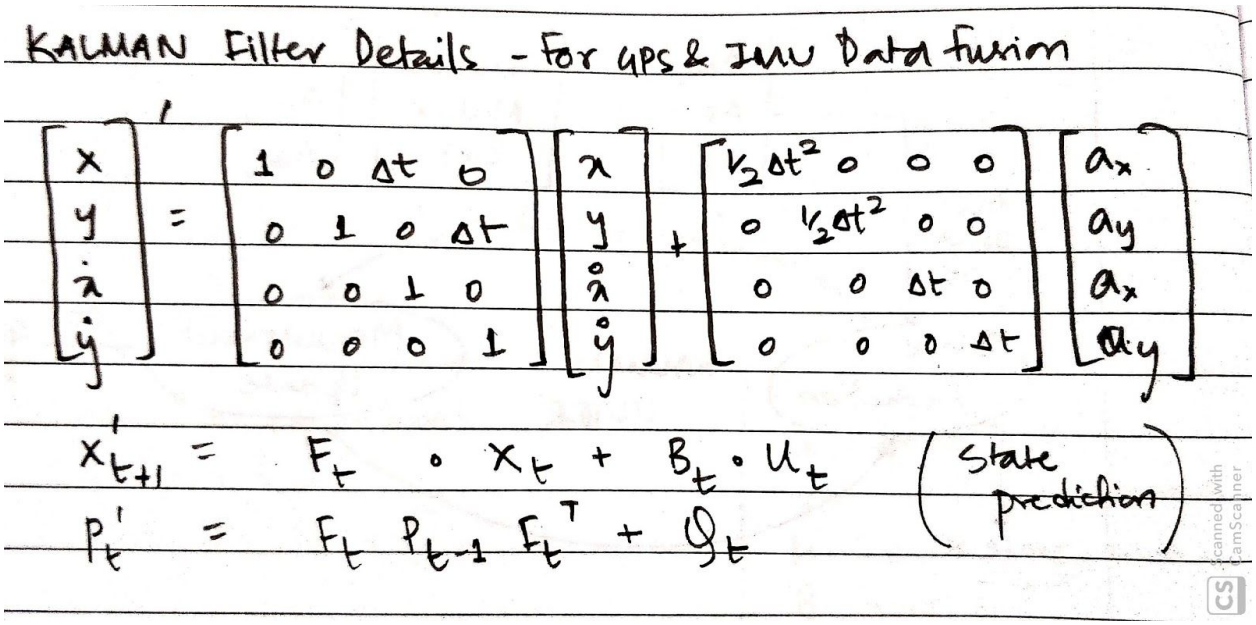


Algorithm Block Diagram

Kalman filter was used to fuse GPS and Absolute Acceleration Data. The following are the details of the Kalman filter. Since GPS data is at low frequency, low accuracy and good absolute position data points, and IMU has high frequency, high accuracy and but drifts over time, the covariance noise for State prediction(IMU data) is kept high and the measurement covariance for Measurement Update is low (GPS) allowing the algorithm to estimate the trajectory accurately



System Equations used for prediction step



System Equations for prediction step for GPS - IMU Data Fusion

$$z_t = \begin{bmatrix} \text{GPS\_longitude} \\ \text{GPS\_latitude} \end{bmatrix} \qquad H_t = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

$$y_k = z_t - H_t \cdot x'_{t+1}$$

$$S_t = H_t P_t H_t^T + R_t$$

$$K_t = P_t H_t^T S_t^{-1}$$

$$x_t = x'_{t+1} + K_t (z_t - H_t x'_{t+1})$$

$$P_t = (P'_t - K_t H_t P'_t)$$

Measurement Update Step, uses GPS Data

Where...

F - State transition Matrix

X - State Vector

B - Control Matrix

U - Control Vector

P - State Covariance

Q - Process Covariance

Z - Measurement Vector

R - Measurement Covariance

K - Kalman Gain

Notations used

Result

Estimating Image Trajectory from Images (possible solutions/algorithms)

- Optical flow using the Lucas Kanade Method
- Feature tracking and point correspondences (OpenCV)

References

- Kalman Filter for IMU and GPS data fusion references
  https://www.youtube.com/watch?v=6M6wSLD-8M8
  https://www.youtube.com/watch?v=T9jXoG0QYIA
- Kalman filter code snippet
  http://ros-developer.com/2019/04/10/kalman-filter-explained-with-python-code-from-scratch/
- AHRS library to get Absolute East-North-Down Acceleration Vector
  https://github.com/morgil/madgwick_py
- Calculating magnetic declination: https://www.geomag.nrcan.gc.ca/calc/mdcal-en.php
  Latitude: 48.0831° North
  Longitude: 11.6354° East
  Date: 2019-11-26
  Magnetic declination: 3° 25.20' East
- Other code snippets borrowed from StackOverflow for distance calculations between Latitude and Longitude.

Python Modules used

- math
- matplotlib
- gmplot
- statistics
- numpy
- madgwickahrs