New York University

Unscented Kalman Filter for State Estimation of Robots

ROBOT LOCALISATION AND NAVIGATION

ROB-6213 Spring 2021

PROJECT-3

Professor: Giuseppe Loianno

Student name: - Rishi Eswar Varma Vegesna (N17479594)

Student Netid: - rv2210

INTRODUCTION

The project was implementation of unscented Kalman filter for sate estimation of a robot. The various state estimates given by the unscented Kalman filter were position, velocity, orientation, gyroscopic bias and accelerometer bias. In this project we use the IMU driven model from project 1 of this course and combine it with inertial data from camera pose and velocity obtained in project 2 of this course.

The given project consists of 2 parts orientation pose sensor part 1 and velocity sensor part 2. In the part 1 of the project, we use measurement from IMU for visual pose estimation. Since the pose was in world frame the part 1 of the project uses same linear measurement model from project 1. In the part 2 of the project, we only use the velocity from optical flow for pose estimation. In part 2 since the velocity was expressed in camera frame of the aerial vehicle the measurement model was nonlinear and was converted into body frame of the aerial vehicle. The outline of the code for unscented Kalman filter was given and we were only needed to update in Kalman filter For loop, prediction step and update step. These sections will be explained further in the report

PROJECT PART 1 – ORIENT POSE SENSOR OUT

In this part of the project the state of the robot was estimated from the IMU data and since the pose of the robot was in world frame the measurement model was linear. This part of the project contains 5 sections

Init function

This function was used to initialise the data and read the sample Data, Vicon Data and times from the given data. This was already given as the part of the code and should not be modified

Unscented Kalman filter

This is the main function of the code, and it takes the initialised data from init function and runs the functions prediction step, update step, and also runs the plot function.

Prediction step

This function calculates estimated mean and estimated covariance required for the update step. The inputs of this function are mean previous, angular velocity, covariance previous, acceleration, sample time. In this step we need to calculate X dot matrix and then compute the sigma points with augmented state and propagating that sigma point through nonlinear function. Then using the sigma points calculating predicted mean and covariance matrix. The following steps were followed in the prediction step

Initially the f(x, u, n) matrix was calculated from previous mean given to the function. This involves calculating Rotation matrix in Z-Y-X euler angle orientation and G matrix of same orientation

$X=[\ x\ y\ z\ roll\ pitch\ yaw\ v_x\ v_y\ v_z\ b_{g1}\ b_{g2}\ b_{g3}\ b_{a1}\ b_{a2}\ b_{a3}]^T$

$X\_dot= [\ X_3\quad G(X_2)^{-1}R(w_m-X4-n_g)\quad g+R(X_2)(a_m-X_5-n_a)\quad n_{bg}\quad n_{ba}]^T = f(\mu_{t-1},u_t,0)$

Next, we define n-prime lambda-prime alpha beta. N prime was the n-dimensional gaussian with mean u and covariance . This n-prime gives us the resulting sigma points. Alpha, k were scaling parameters that determine how far the sigma points were spread from the mean. Beta was chosen to encode additional information regarding the gaussian distribution.

Once we define these parameters, we calculate weights associated with each sigma points and also lambda-prime values. There were two types of weights one was w_m and other was w_c these were used when calculating mean and covariance. After getting all parameters we then define P-aug and U-aug matrices. It is important to define the size of this matrices correctly.

Now to calculate square root of P-aug Cholesky decomposition was done, this was done using inbuilt function in MATLAB called Chol which computes the sqrt(P-aug) and gives us definite lower triangular matrix. After computing Cholesky decomposition the sigma point were calculated. There are a total number of 2n-prime+1 sigma points. Since we can't give plus and minus symbol in the formula at the same time, we consider two cases positive case and negative case and compute the respective X_t function matrices. Once this calculation was done for all sigma points, we need to discreditise the function X_t, this can be done by multiplying the function with dt. Finally, we have to calculate mean estimate and covar estimate from the function X_t by multiplying them with their respective weights w_m and w_c

Alpha= 0.001
K=1 , alpha and K were sigma points spread from mean
Beta= 2 , beta tells about gaussian, optimal choice value 2 (according to probabilistic robotics book)
n'= 15 +12 , n=15 and n_q=12 were dimensionalities of X and q

$\boldsymbol{\mu}_{aug} = \begin{pmatrix} \boldsymbol{\mu} \\ \mathbf{0} \end{pmatrix}, \boldsymbol{\Sigma}_{aug} = \begin{pmatrix} \boldsymbol{\Sigma} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix}$,mean and covariance augmented

$\boldsymbol{\lambda}' = \boldsymbol{\alpha}^2(\boldsymbol{n}' + \boldsymbol{k}) - \boldsymbol{n}'$ **,** calculating lambda prime value

$W_0^{(m)'} = \frac{\lambda'}{n'+\lambda'}$, $W_i^{(m)'} = \frac{1}{2(n'+\lambda')}$, weight for mean calculation

$W_0^{(c)'} = \frac{\lambda'}{n'+\lambda'} + (1 - \alpha^2 + \beta)$, $W_i^{(c)'} = \frac{1}{2(n'+\lambda')}$, weight for covariance calculation

$\mathcal{X}^{(i)} = \boldsymbol{\mu}_{aug} \pm \sqrt{n' + \lambda'}\left[\sqrt{\boldsymbol{\Sigma}_{aug}}\right]_i$, computing sigma points

$\chi_t^{(i)} = f\left(\chi_{\text{aug,t-1}}^{(i),x}, u_t, \chi_{\text{aug,t-1}}^{(i),n}\right)$, sigma points through non-linear function

$\bar{\boldsymbol{\mu}}_t = \sum_{i=0}^{2n'} W_i^{(m)} \chi_t^{(i)}$, predicted mean

$\bar{\boldsymbol{\Sigma}}_t = \sum_{i=0}^{2n'} W_i^{(c)'} \left(\chi_t^{(i)} - \bar{\boldsymbol{\mu}}_t\right)\left(\chi_t^{(i)} - \bar{\boldsymbol{\mu}}_t\right)^T$, predicted covariance

Update step

This function was used to calculate current mean and current covariance. The inputs for the function were measurement model z_t, estimated mean and estimated covariance from prediction step. Initially in the measurement model we have to calculate ct and wt to linearise the system. The ct was given by the formula

$$C_t = \begin{bmatrix} p \\ q \end{bmatrix} + v = \begin{bmatrix} I & 0 & 0 & 0 & 0 \\ 0 & I & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} p \\ q \\ p_{dot} \\ bg \\ ba \end{bmatrix} + v = CX + v$$

In this case the measurement model depends on position and velocity so the ct matrix was an identity matrix with dimensions 6x15. Next, we define R which was as scaling parameter that depends on how much we believe in our sensor readings. Once these parameters were defined, we calculate Kalman gain. Using the Kalman gain we calculate current mean and current covariance

$z_t = Cx + \eta$,z_t was measurement model

$\boldsymbol{K_t = \bar{\Sigma}_t C^T \left(C \bar{\Sigma}_t C^T + R \right)^{-1}}$, Kalman gain

$\boldsymbol{\mu_t = \bar{\mu}_t + K_t \left(z_t - C \bar{\mu}_t \right)}$,current mean

$\boldsymbol{\Sigma_t = \bar{\Sigma}_t - K_t C \bar{\Sigma}_t}$,current covariance

Plot function

This function takes in the sampled data and data from current mean and covariance and plots the graph between predicted pose and estimated pose. This function was already given in the project file and should not be changed.

PROJECT PART 2 – VELOCITY SENSOR OUT

This part of the project estimates the pose of the aerial vehicle by using the velocity from the optical flow. Since the velocity was expressed in camera frame the model will be non-linear and should be converted to body frame that was coincident with the IMU frame. This part of the project also contains 5 sections

The init function, plot function, and prediction step functions were same as part 1 of the project

Unscented Kalman filter

This function was also similar to the one we used in part 1 of the project. The only change was in the measurement model z_t that we pass to the updated step function. In part 1 we pass position and pose in the measurement model but in part 2 we pass velocity and angular velocity as z_t into the update step.

Prediction step

This function was same as prediction step in part 1 of the project. In this step we calculate u-augmented, covar-augmented and then calculate the sigma points. Once the sigma points were calculated the mean estimate and covariance estimate were calculated by multiplying the weight to Xt function.

Update step

This update step was different from the update step in part1 because in this case the system was non-linear. Initially we calculate the transformation matrix from estimated mean and also the transformation matrix of

camera and body frames. After this we define the k, alpha, beta, n values which are scaling parameters of sigma point distribution and n is the dimension of the gaussian matrix. Mean and covariance weights were calculated from these values. Now the Z_t matrix was calculated and Cholesky decomposition was applied on augmented covariance matrix. Once this was done the sigma points were computer and are propagate through nonlinear function g. After this the z was calculated by multiplying respective weights. From z the uncertainty matrix S_t and predicted mean and covariance c_t were calculated. From this the Kalman filter gain was calculated which was used to calculate current mean and current covariance. These resulting filtered mean and filtered covariance were then returned to unscented Kalman filter function which were then plotted. The formulas used in this step were as follows

Alpha= 0.001

K=1 , alpha and K were sigma points spread from mean

Beta= 2 , beta tells about gaussian, optimal choice value 2 (according to probabilistic robotics book)

n'= 15 +12 , n=15 and n_q=12 were dimensionalities of X and q

$W_0^{(m)'} = \frac{\lambda'}{n'+\lambda'}$, $W_i^{(m)'} = \frac{1}{2(n'+\lambda')}$, weight for mean calculation

$W_0^{(c)'} = \frac{\lambda'}{n'+\lambda'} + (1 - \alpha^2 + \beta)$, $W_i^{(c)'} = \frac{1}{2(n'+\lambda')}$, weight for covariance calculation

$\boldsymbol{\mu}_{aug,t} = \begin{pmatrix} \bar{\boldsymbol{\mu}}_t \\ \mathbf{0} \end{pmatrix}$, mean augmented from predicted mean

$\boldsymbol{\Sigma}_{t,aug} = \begin{pmatrix} \bar{\boldsymbol{\Sigma}}_t & \mathbf{0} \\ \mathbf{0} & \boldsymbol{V}_t \end{pmatrix}$, covar augmented from predicted covar

$\chi_{aug,t}^{(i)} = \mu_{\text{aug,t}} \pm \sqrt{n'' + \lambda''}\left[\sqrt{\boldsymbol{\Sigma}_{t,aug}}\right]_i$, sigma points

$Z_t^{(i)} = g\left(\chi_{aug,t}^{(i),x}, \chi_{aug,t}^{(i),v}\right)$, propagating sigma points through g

$\boldsymbol{Z}_{\mu,t} = \sum_{i=0}^{2n''} W_i^{(m)''} Z_t^{(i)}$, cross covariance

$z_t = \quad {}^{C}\boldsymbol{v}_{\boldsymbol{C}}^{W} = g(x_2, x_3, \quad {}^{B}\omega_B^W) + \eta$, measurement model

$\boldsymbol{C}_t = \sum_{i=0}^{2n''} W_i^{(c)''}\left(\chi_{aug,t}^{(i),x} - \bar{\boldsymbol{\mu}}_t\right)\left(Z_t^{(i)} - \boldsymbol{z}_{\mu,t}\right)^T$

$\boldsymbol{S}_t = \sum_{i=0}^{2n''} W_i^{(c)''}\left(Z_t^{(i)} - \boldsymbol{z}_{\mu,t}\right)\left(Z_t^{(i)} - \boldsymbol{z}_{\mu,t}\right)^T$, uncertainty matrix

$\boldsymbol{K}_t = \boldsymbol{C}_t \boldsymbol{S}_t^{-1}$, Kalman filter gain

$\boldsymbol{\mu}_t = \bar{\boldsymbol{\mu}}_t + \boldsymbol{K}_t\left(\boldsymbol{z}_t - \boldsymbol{z}_{\mu,t}\right)$, current mean

$\boldsymbol{\Sigma}_t = \bar{\boldsymbol{\Sigma}}_t - \boldsymbol{K}_t \boldsymbol{S}_t \boldsymbol{K}_t^T$, current covariance

PLOTS AND RESULTS

There was a total of 4 plots. Datasets 1 and dataset 4 for project part 1 and part2. The plots show differences between the predicted model and actual model. The plots for the datasets were as follows

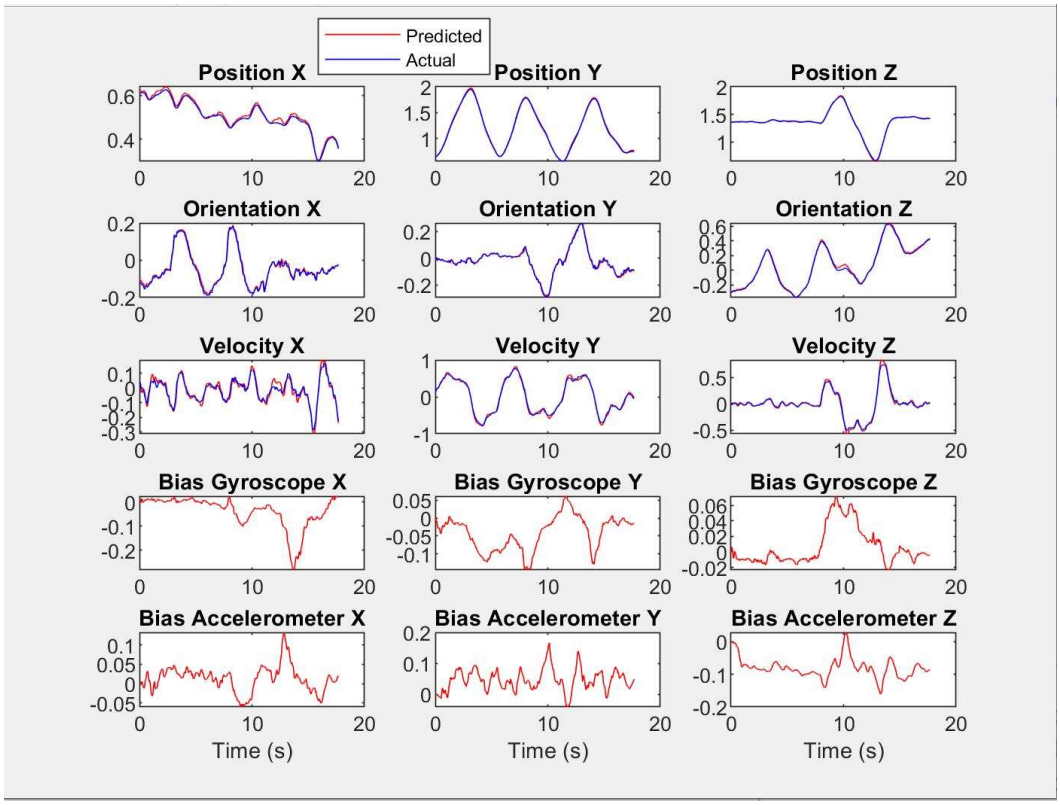

Figure 1: UKF part 1 Dataset 1

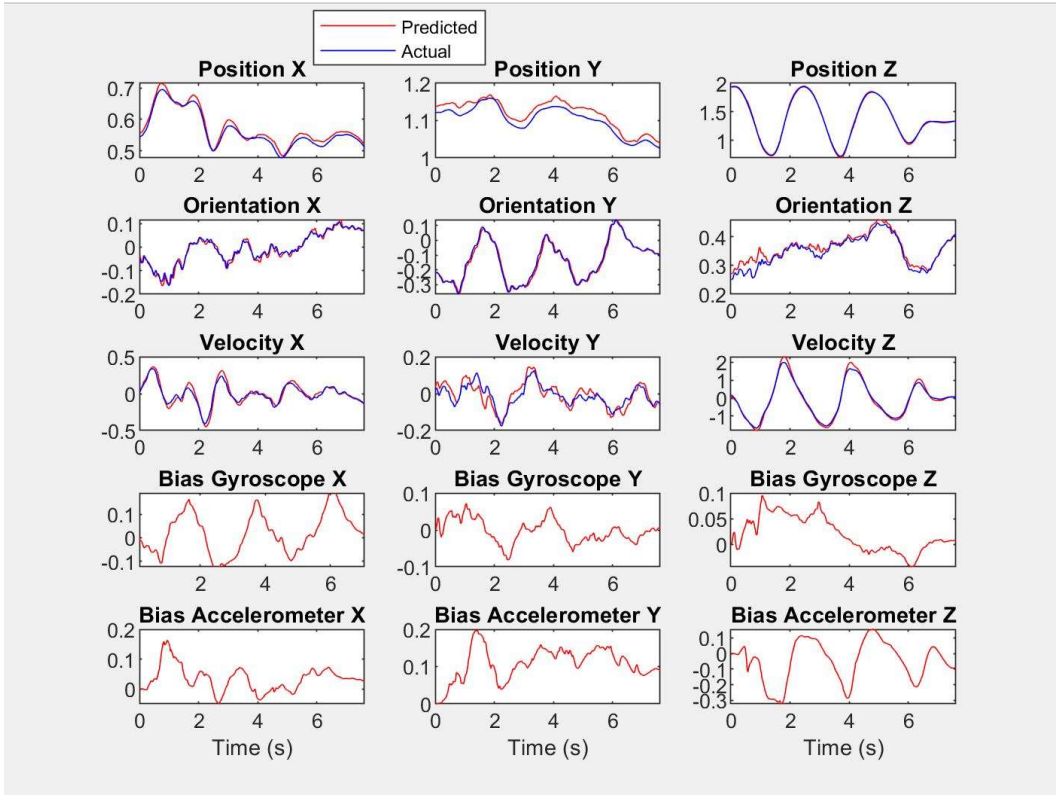

Figure 2: UKF part 1 Dataset 4

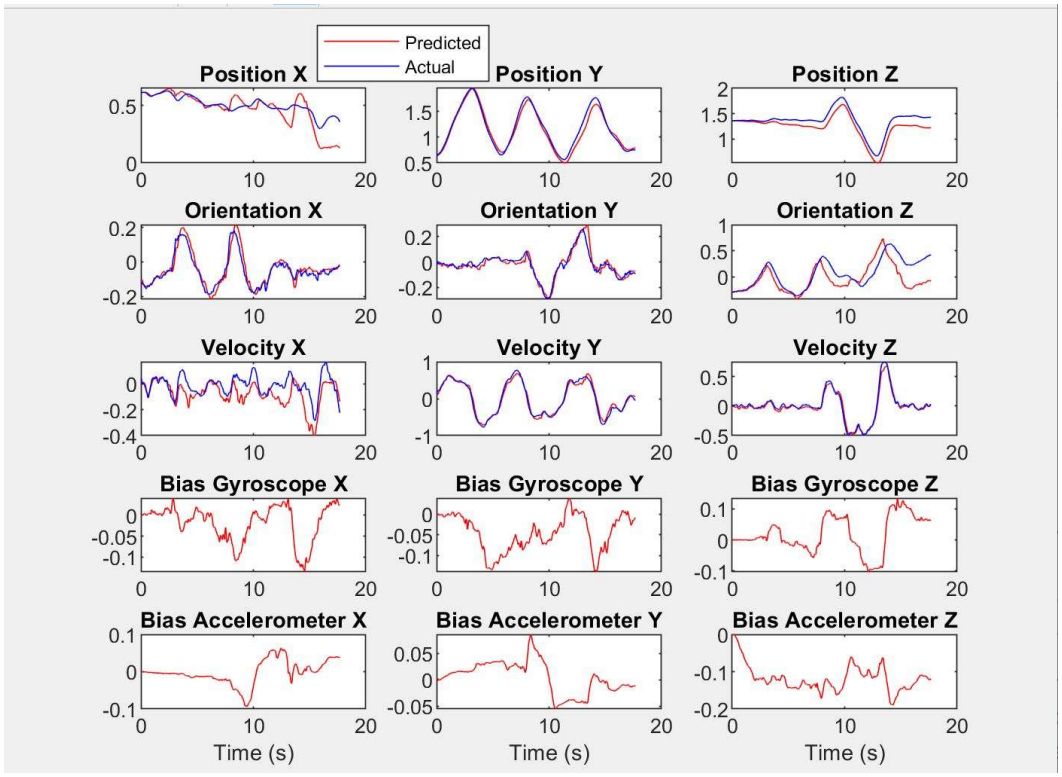

Figure 3: UKF part 2 Dataset 1

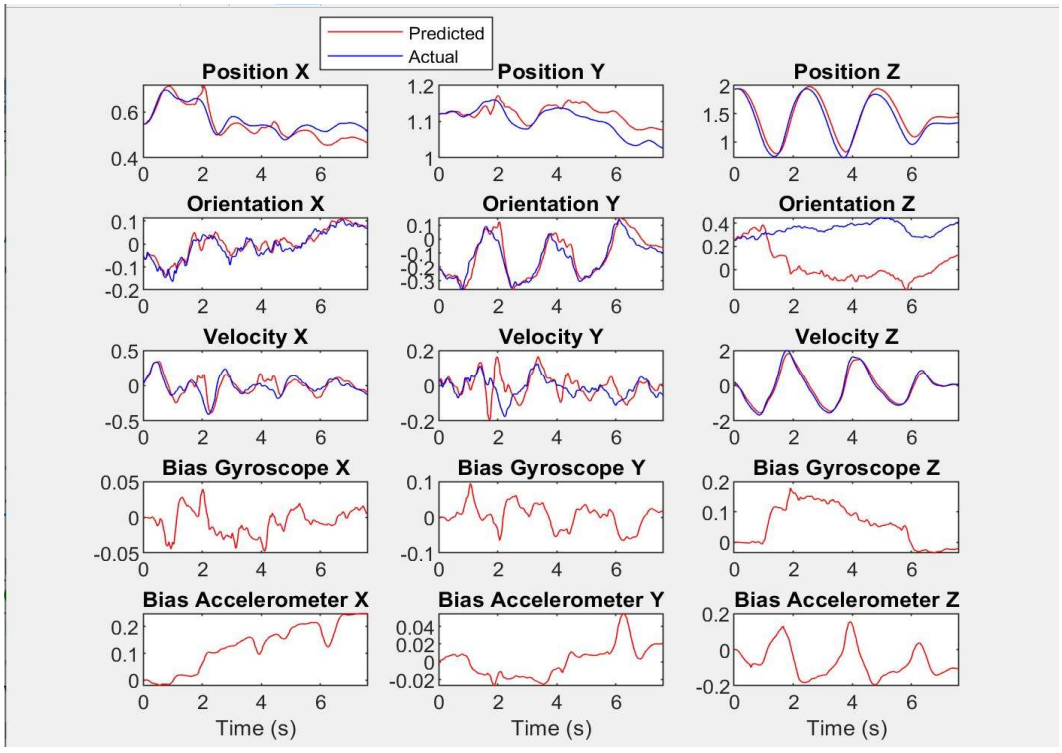

Figure 4: UKF part 2 Dataset 4

The plots of unscented Kalman filter were almost perfectly coinciding with the actual plots. However, in part 1 dataset number 4 there were little noises in the plots position x, position y, and velocity y plots this can be minimised by scaling the parameter values in prediction and update steps. The plots for part 2 of the project were almost coinciding with the actual values. But in part 2 plots the slight offset of the predicted plot from actual plot was visible this may be due to various factors like the values we considered for alpha, beta, k, and the scaling factor values we considered in update step of part 2. These discrepancies may also be due to inconsistencies in the sampled data providing with project. Overall, the plots were almost accurate with little offset in project part 2, which could be improved by tuning the scaling factor values in prediction and update step and also by checking for any inconsistencies in sampled data provided with the project

REFERENCES

1. Professor Giuseppe Loianno's Lecture notes

2. GRASP Lab report by Professor Vijay Kumar, university of Pennsylvania

3. Thrun S Burgard (2006), "Probabilistic Robotics", Kybernetes, Vol. 35 No. 7/8, pp

4. State Estimation for Robotics Chapter 3