ROBOT LOCALIZATION AND NAVIGATION (ROB-6213) PROJECT 3 REPORT

Rushi Bhavesh Shah (rs7236)

1. Introduction

The extended Kalman filter can perform poorly when the state transition and observation models, i.e., the predict and update functions f and h, are extremely nonlinear. This is because the underlying nonlinear model's covariance is communicated through linearization. The unscented Kalman filter (UKF) selects a limited number of sample points (called sigma points) around the mean using a deterministic sampling technique known as the unscented transformation (UT). Following that, the sigma points are propagated through the nonlinear functions, yielding a new mean and covariance estimate. The filter that results is determined by how the UT's modified statistics are produced and which set of sigma points are used. It should be noted that building new UKFs in a consistent manner is always possible. The resulting UKF more precisely calculates the true mean and covariance for specific systems. Monte Carlo sampling or Taylor series expansion of the posterior statistics can be used to verify this. Furthermore, this technique eliminates the need to directly calculate Jacobians, which can be a challenging, if not impossible, process for complex functions (i.e., requiring sophisticated derivatives if done analytically or being computationally expensive if done numerically) (if those functions are not differentiable).

2. Problem Statement

The aim of this project is to program an Unscented Kalman Filter (UKF) to fuse the inertial data already used in project 1 and the vision-based pose and velocity estimation developed in project 2. The UKF may capture the non-linearity of the system better, but it might require more runtime. The IMU-driven model from the project 1 is to be used and fused with the inertial data with from camera pose and velocity obtained in project 2. In the first one, visual pose estimation is used as measurement, whereas in the second one only the velocity is used from the optical flow.

3. Solution for Part 1

For the the entire project, we follow the model for non-additive noise that is incorporated in the function model itself. So, for Part one, I first startedoff by developing the prediction step and then the update step.

3.1 The Prediction Step

In the prediction step, the first thing I did was calculate the value if ‘lambda’, following the equation :

$$\lambda' = \alpha^2(n' + k) - n'$$

Then, I defined the augmented state space, along with augment mean and covariance. Then I used the Cholesky decomposition to find the matrix $\sqrt{\Sigma_{aug}}$. Later on, I computed the sigma point by using a for-loop, and stored all the sigma points in a 27 x 55 sized matrix. Later, on I ran all the sigma points through the process model, the one similar to the process model used in project one, except this time. Lateron, I computed the weights in order to compute the estimated mean and covariance. So, to calculate the weights, the following equations were used:

$$W_0^{(m)\prime} = \frac{\lambda'}{n' + \lambda'} \qquad W_i^{(m)\prime} = \frac{1}{2(n' + \lambda')}. \qquad i = 1, \dots 2n'$$

$$W_0^{(c)\prime} = \frac{\lambda'}{n' + \lambda'} + (1 - \alpha^2 + \beta) \quad W_i^{(c)\prime} = \frac{1}{2(n' + \lambda')}$$

Further, the *uEst* and *CovarEst* calculated by running the below eqautions in a for loop and taking th overall summation.

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

3.2 The Update Step

This step takes the predicted values from the predicted step to update it using the measured data from the sensors. Update step starts by identifying the observation model g(xt, vt) where xt is the predicted state and vt is the noise with zero mean. Further we need to calculate the Kalman Gain, updated mean and the covariance using the following equations. These equations are the same as the ones in project one.

$$\boldsymbol{\mu}_t = \bar{\boldsymbol{\mu}}_t + \boldsymbol{K}_t\,(\boldsymbol{z}_t - \boldsymbol{z}_{\mu,t})$$
$$\boldsymbol{\Sigma}_t = \bar{\boldsymbol{\Sigma}}_t - \boldsymbol{K}_t\,\boldsymbol{S}_t \boldsymbol{K}_t^{\,T}$$
$$\boldsymbol{K}_t = \boldsymbol{C}_t\,\boldsymbol{S}_t^{\,-1}$$

3.3 The Main Program

To obtain Unscented Kalman Filter, we need to integrate both the steps viz. prediction and the update step. To do that, we first initialize the data files and parse the data to the main script. We take the observed data for the position and the orientation from the vicon. Also, we'll initialize the values of the mean by taking the initial values of the vicon. While writing the main loop, we run the for loop for the number of timestamps available in the dataset. We need to calculate the time-difference i.e. dt. We can do this by defining two variables t2 and t1. For the first loop, t1 = 0, and the first element in the array will be t2. Then we'll calculate dt = t2- t1. Further, we'll extract the values for angular velocity and acceleration from the IMU data, corresponding to the iteration variable. We'll pass values of uPrev, covarPrev, angVel, acc and dt to the pred_step function; from which we'll get covarEst and uEst i.e. estimated covariance and mean. Then, assign t2, the value of t1. Next, for the update step, we'll extract the observed data z_t from the project 2 data for position and orientation of the robot, again, corresponding to the iteration variable. We'll pass the output from the pred_step function as input to the upd_step function, along with the observed data z_t. Here, we'll get updated or current mean and covariance (uCurr & covar_curr) as outputs. Then, to continue the loop, we'll do covarPrev = covar_curr uPrev = uCurr and save the updated/current mean in the avedStates matrix, which will help in plotting the data.

4. Solution for Part 2

4.1 The Prediction step

The prediction step will be followed in the similar way as that in Part 1.

4.2 The Update Step

For this part, we'll tak ethe data from the optical flow, that we did in the project 2. Next, similar to previous part, we start of by calculating the lambda and then the augmented state, mean and the covariance. Furthermore, we'll have to transform the frames in order to get everything in the world frame since from project 2 we have data in the camera frame. Further, we calculate the sigma points as we did in the previous part. Further, I ran Sigma points throught the non-linear function. Then, I calculated the weights for the present values of n, lambda, alpha and beta. Then I calculated predicted mean, using the equation

$$\mathbf{z}_{\mu,t} = \sum_{i=0}^{2n''} W_i^{(m)''} Z_t^{(i)}$$

Further, I calculated the predicted cross-covariance C_t and predicted covariance of the measurement S_t using the equations:

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

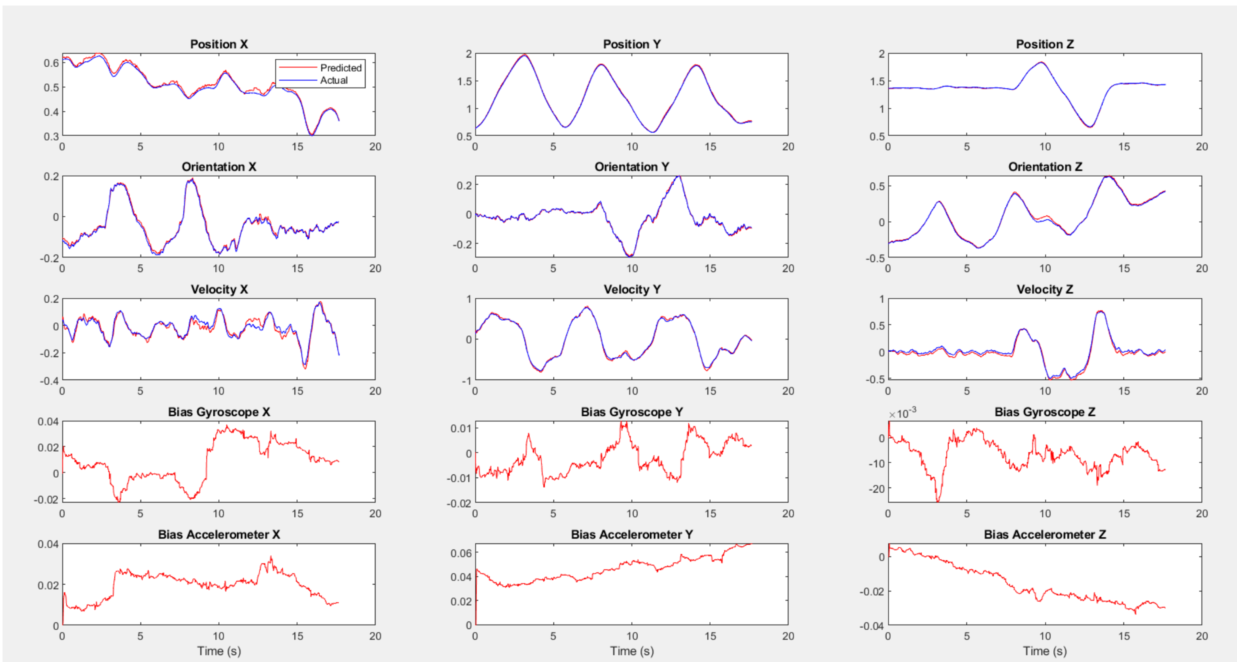
$$\boldsymbol{S}_t = \sum_{i=0}^{2n''} W_i^{(c)''} \left(Z_t^{(i)} - \mathbf{z}_{\mu,t}\right)\left(Z_t^{(i)} - \mathbf{z}_{\mu,t}\right)^T$$

And then used the equations shown below, to compute the Kalman Gain and the filtered state mean and covariance, conditional to the measurement.

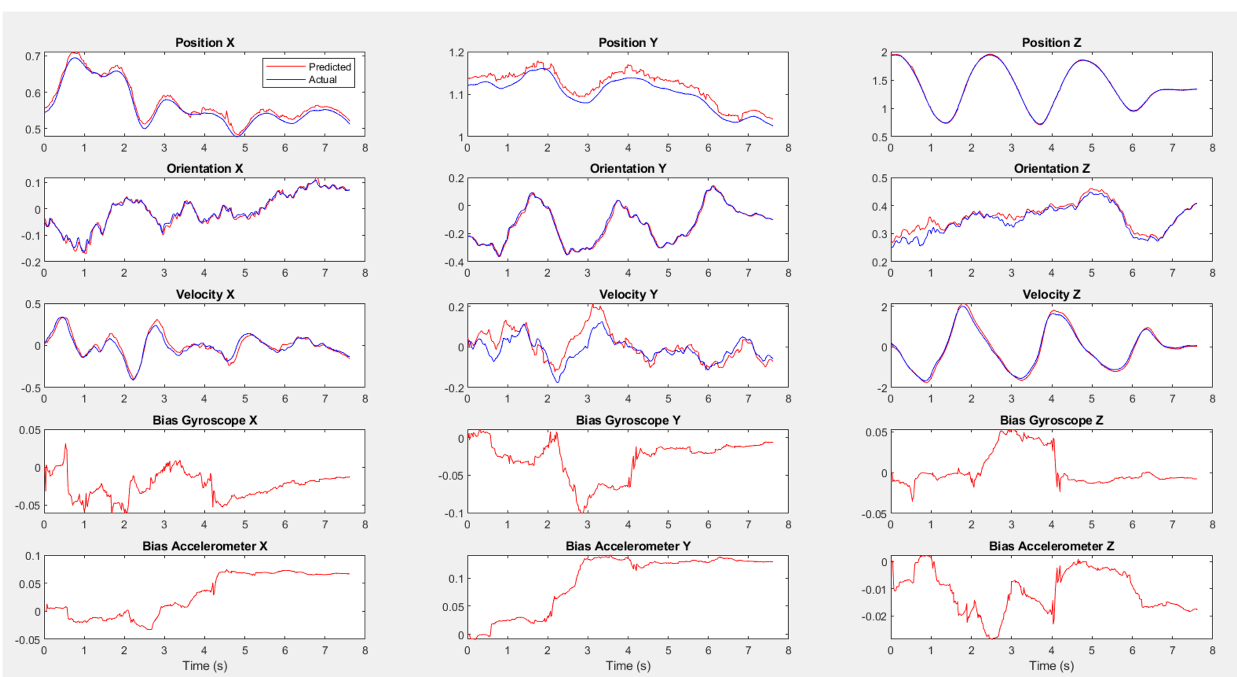
4.3 The Main Program

The programis similar to that in the previous part, although, the data z_t, this time is the velocity data from the project 2.

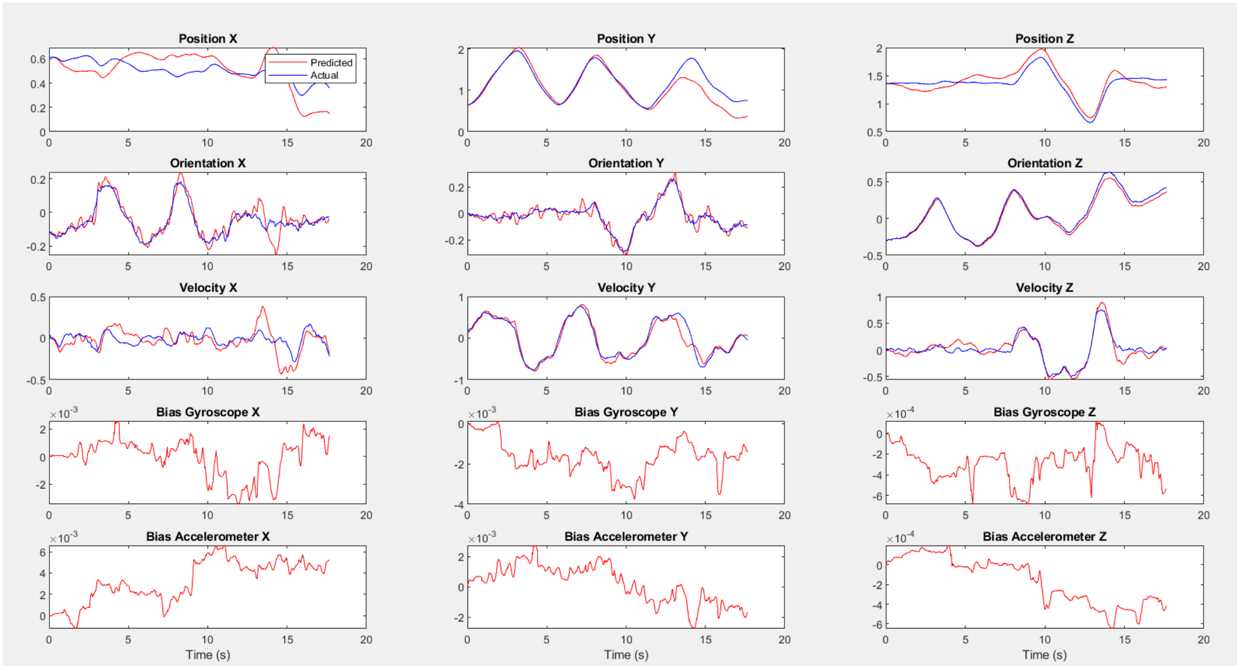
5. Plots



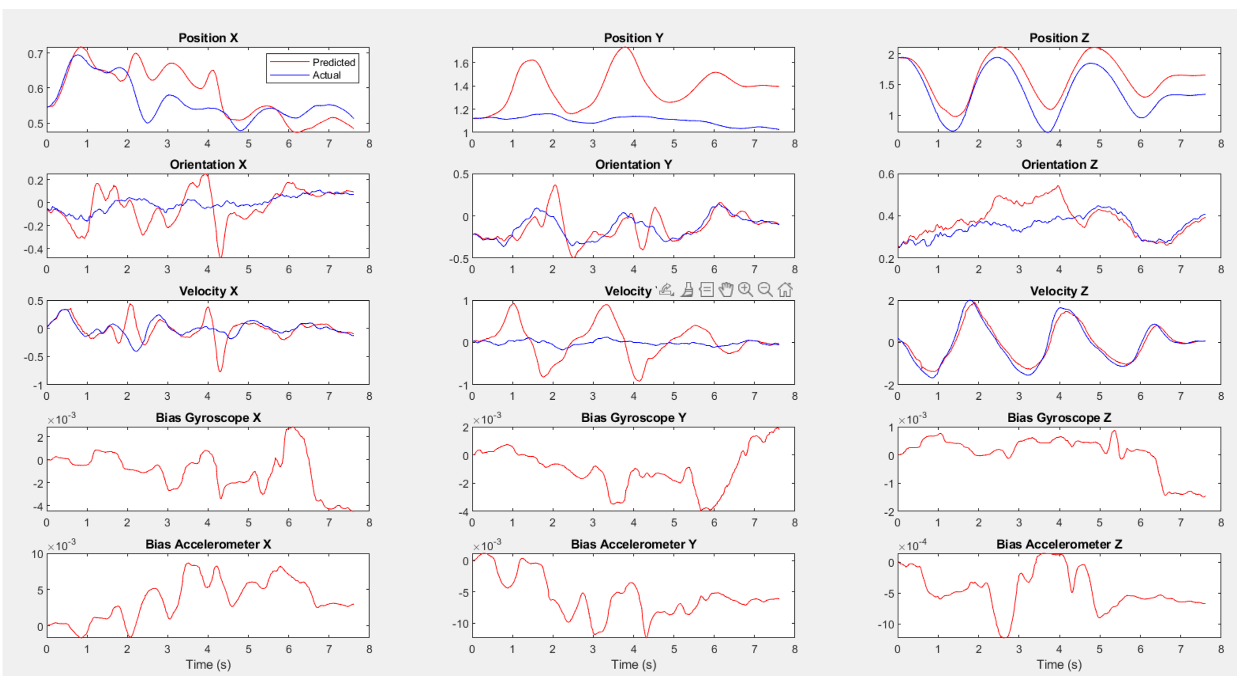
Part1 DataSet1



Part1 DataSet4



Part2 Dataset1



Part2 Dataset4

6. Conclusion

The Unscented Kalman filter better approximated the distribution than that done by the EKF. The noise values need to be tuned in order to ake the plots overlap with each other. However, a drift has been observed in the Part2 when Dataset4 is used. That is due to the untuned noise values.