ROB GY-6213 Robotic Localization and Navigation project I

Yash Khanolkar yak9424 N12169803

15 March 2024

1 Introduction

The crux of Robot localization and Navigation is to be able to estimate the states of the system which is our primary goal in this project. The vicon is basically a motion capture system which can track the movement of our robot which is measured by using the onboard IMU sensor. The vicon data obtained during measurement is decoded in MATLAB. On this data we perform our main objective of this project,i.e, implementing the Extended Kalman Filter(EKF) to estimate and plot the position, orientation, velocity and the sensor biases of a micro Aerial vehicle for different datasets.

2 Problem Statement

We will use the Extended Kalman Filter (EKF) to estimated the position, velocity, and orientation, and sensor biases of an Micro Aerial Vehicle. To simplify the overall process, we have already been given a skeleton code that is able to synchronize the data that you need to use. The Vicon velocity is given in the world frame, whereas the angular rate in the body frame of the robot. We will use the body frame acceleration and angular velocity from the on board IMU as your inputs. We will have to present two filter versions. For this reason the project has been dived in two parts in separate folders. In 1 the first one, the measurement update will be given by the position and orientation from vicon, whereas in the second one we will use only the velocity from the Vicon. In the released code, two separate folders have been created to accommodate each part. The function init.m is called within the KalmanFilt Part1.m and KalmanFilt Part2.m functions to initialize the process and select the specific dataset. The data is already parsed. In this way, the sensor data has already been parsed to facilitate the process. The Euler angles convention to use is ZYX.We should have two measurement models, one for each part of the project. The first one for the position and orientation (part 1) and the second one for the velocity (part 2). In both parts, the process model is the same.

3 Procedure

Firstly, we have to perform Z-Y-X Euler angle parameterization as given in the problem statement.This can be computed by,

$$R_{ZYX} = R_Z.R_Y.R_X$$

We know that the angular velocity in the body frame is given by,

$$\omega = G(q).q^{\cdot}$$

where we have determined from Euler angles to be, $G(q) = \begin{bmatrix} 1 & 0 & -sin(\theta) \\ 0 & cos(\psi) & cos(\theta)sin(\psi) \\ 0 & -sin(\psi) & cos(\theta)cos(\psi) \end{bmatrix}$

and $q^{\cdot} = \begin{bmatrix} \psi^{\cdot} \\ \theta^{\cdot} \\ \phi^{\cdot} \end{bmatrix}$

Now the ω that we get is actually with respect to the world frame (ω_w) but we want ω in the body frame which we can get using the formula

$$\omega_w = R_b^w.\omega_b$$

Hence, ω_b =$R_w^b.\omega_w$ Putting the value of ω_w we get $\omega_b = {R_b^w}^T$.G(q).$q^{\cdot}$

Now, we can start obtaining our **process model**.

The noisy estimate of the angular velocity of the vicon system can be given by

$$\omega_m = \omega + b_g + n_g$$

where we have assumed that drift in the gyroscope bias is defined by a gaussian white noise process given by,

$$b_g^{\cdot} = n_{bg}$$

$$n_{bg} \sim N(0, Q_g)$$

Similarly, the noisy estimate of the angular acceleration can be given by,

$$a_m = R(q)^T(p^{\cdot\cdot} - g) + b_a + n_a$$

where we have assumed that drift in the accelerometer bias is defined by a gaussian white noise process given by,

$$b_a^{\cdot} = n_{ba}$$

$$n_{ba} \sim N(0, Q_{ba})$$

In the above formula b_g, b_a are the biases and n_g, n_{ba} are the noise parameters which can be tuned according our requirement. g is the acceleration due to gravity variable defined as g = *9.81*.

Now,the state matrix can be represented as $x = \begin{bmatrix} p \\ q \\ p^{\cdot} \\ b_g \\ b_a \end{bmatrix} \epsilon = R^{15}$

where p = position, q = orientation, $p^{\cdot}$ = linear velocity, b_g = gyrosope bias and b_a = accelerometer bias. Similarly, the **process model** can be given by,

$x^{\cdot} = \begin{bmatrix} p. \\ G(x2)^(-1)(\omega_m - b_g - n_g) \\ g + r(x_2)(a_m - b_a - n_a) \\ n_{bg} \\ n_{ba} \end{bmatrix}$

We assume that $x_{.}$ = f(x,u,n) which, for our case, is, f($\mu_{t-1}, u_t, 0$).

Now, the data which we have been provided is not linear, hence our first plan of action will be to linearize the data by using jacobian functions, after which we will get $A_t = \frac{\partial f}{\partial x} |_{\mu_{t-1}, u_t, 0}$ and $U_t = \frac{\partial f}{\partial n} |_{\mu_{t-1}, u_t, 0}$. We can substitute these values to get the components of the prediction step.

$$f_t = I + \partial t A_t, V_t = U_t, Q_d = Q\partial t$$

.This process is called Discretization.

Now, we can finally derive the **prediction step** and finding the estimated mean and covariance values using the following equations.

$$\overline{\mu}_t = \mu_{t-1} + \partial t f(\mu_{t-1}, u_t, 0).$$

$$\overline{\epsilon}_t = F_t \epsilon_{t-1} f_t^T + V_t Q_d V_t^T$$

This completes our prediction step. Now, we have to update our variable according to what was predicted by our model.

In the measurement part, our model is linear, hence the **update step** is calculated by using the linear **measurement model** which is given by

$$z_t = C_t x_t + v_t, where\, v_t \sim N(0, R_t)$$

Now, the updated values are derived easily after setting the value of R_t, which should be as low as possible to make a good update.

$$\mu = \overline{\mu}_t \; + K_t(z_t - C_t \overline{\mu}_t)$$

$$\epsilon_t = \overline{\epsilon}_t - K_t C_t \overline{\epsilon}_t$$

$$K_t = \overline{\epsilon}_t C_t^T (C_t \overline{\epsilon}_t C_t^T + R_t)^{-1}$$

These are the result of our prediction which we run for the whole function and try to pinpoint the movement of our aerial robot.

4 Code

With the skeletal code provided to us, our main concern in the file Kalmanfilter Part1 is to call the prediction and update step functions in the for loop and ensure that all processes required for that are performed beforehand.

Firstly, we call the angular velocity and acceleration values from our data file. Then, we define the variables and matrices as symbolic to facilitate easy application of jacobian for linearization.

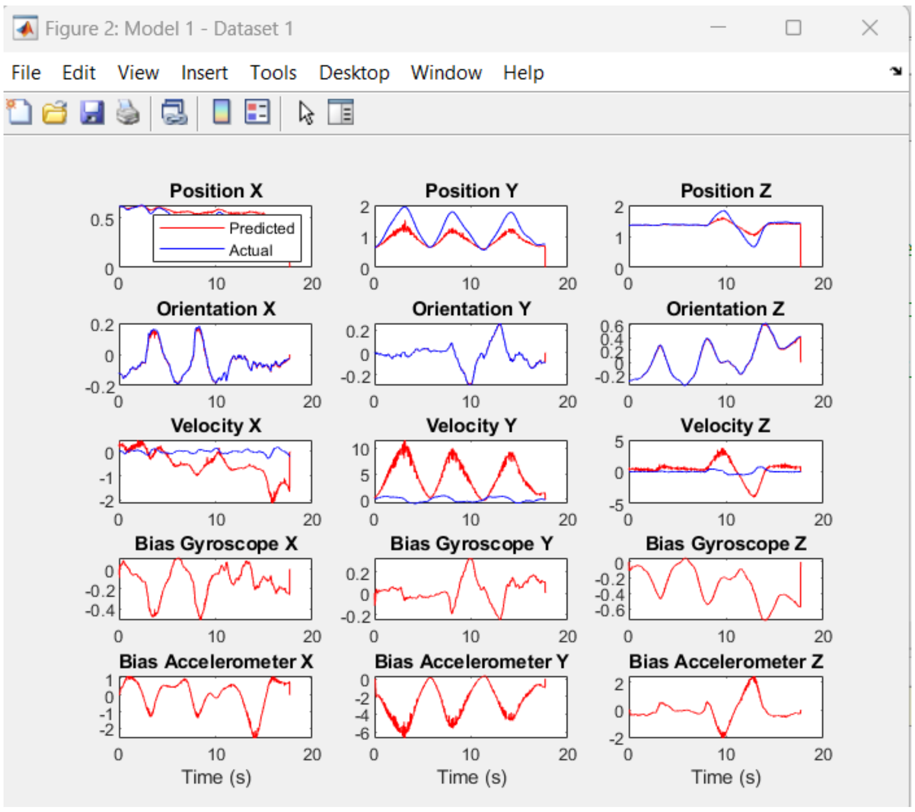
In the for loop, we need to assign the numeric values of the data to the variable as we perform the same function as in the symbolic case, again. This time though, we have to call the functions from the prediction and update part with the help of linear measurement variable z and save the state of each iteration in savedStates variable. Kalmanfilter Part22 performs the exact same function as part 1, only in case of velocity.

In the prediction part, we use the inputs assigned in the function definition to estimate the μ and ϵ values denoted by $\overline{\mu}$ and $\overline{\epsilon}$ respectively. We have to tune the noise to bring the result as close to the expected value as possible. Similar function is performed by the prediction part of the other dataset.

In the update part, both the datasets are different in the positions of C_t that represent position and orientation or velocity and is given by identity element. R value changes accordingly to remove inaccuracies and to fit the matrix size of the calculation. Here, the current value of mean and variance is predicted based on the estimated value we found in the prediction part.

Our aim was to include as much of the code in the main function as possible to reduce the work done in the prediction and update part were the steps were kept as minimum as possible. Also, most of the variables introduced are single to reduce the computation time as much as possible.

5 Results



In part 1, in the first dataset, we were asked to link the position and orientation of our expected output to the one provided to us. As we were expected to use the position and orientation information, the goal was to predict the position and orientation of the robot as accurately as possible while undermining the velocity whose info was not applied in our expression .Hence, we try to reduce the value of R as much as possible to reduce the inaccuracy in our system while simultaneously toggling the Q and W to improve noise efficiency. A similar strategy was followed by the second dataset result shown below. There is a little inaccuracy in the y axis but other than that the noise is tuned to perfection to give position and orientation of the robot as interpreted from dataset 1. In the second dataset, the function performed is the same with a little inaccuracy being in the z - axis.

In part 2, in the second dataset, we were expected to align the velocity graph with what was expected . For this, we try to determine velocity as close as possible. This means skewed graphs in case of position and orientation as they were not taken into consideration for any of the datasets.

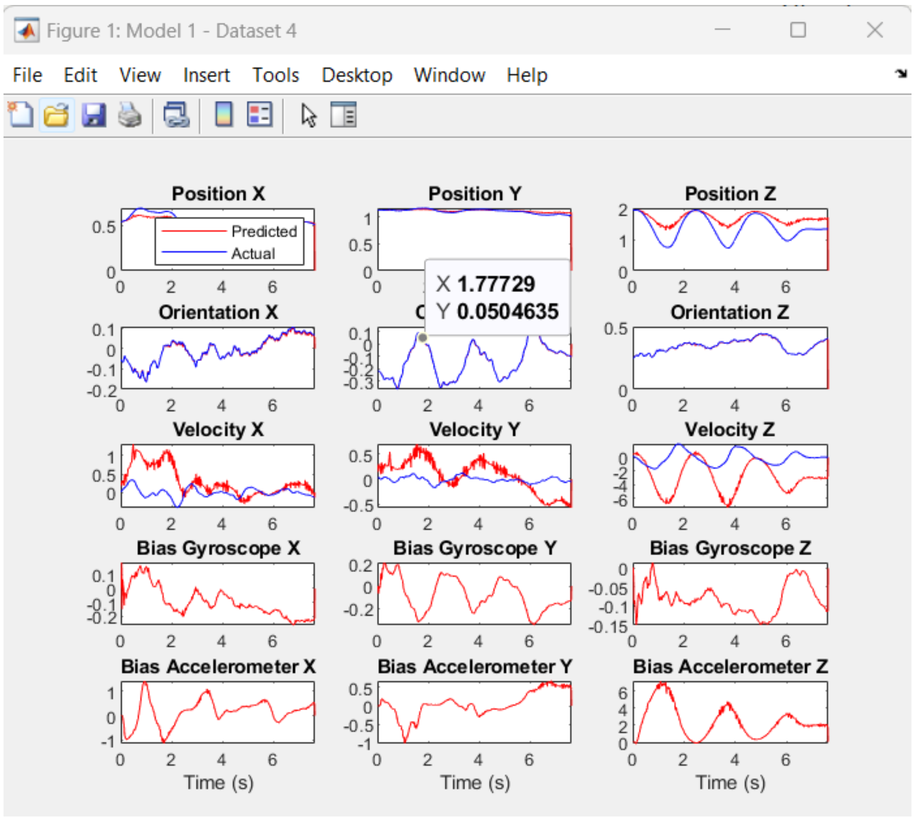


Figure 1: Model 1 - Dataset 4
File Edit View Insert Tools Desktop Window Help
Position X
Predicted
Actual
Position Y
Position Z
Orientation X
Orientation Z
X 1.77729
Y 0.0504635
Velocity X
Velocity Y
Velocity Z
Bias Gyroscope X
Bias Gyroscope Y
Bias Gyroscope Z
Bias Accelerometer X
Bias Accelerometer Y
Bias Accelerometer Z
Time (s)
Time (s)
Time (s)

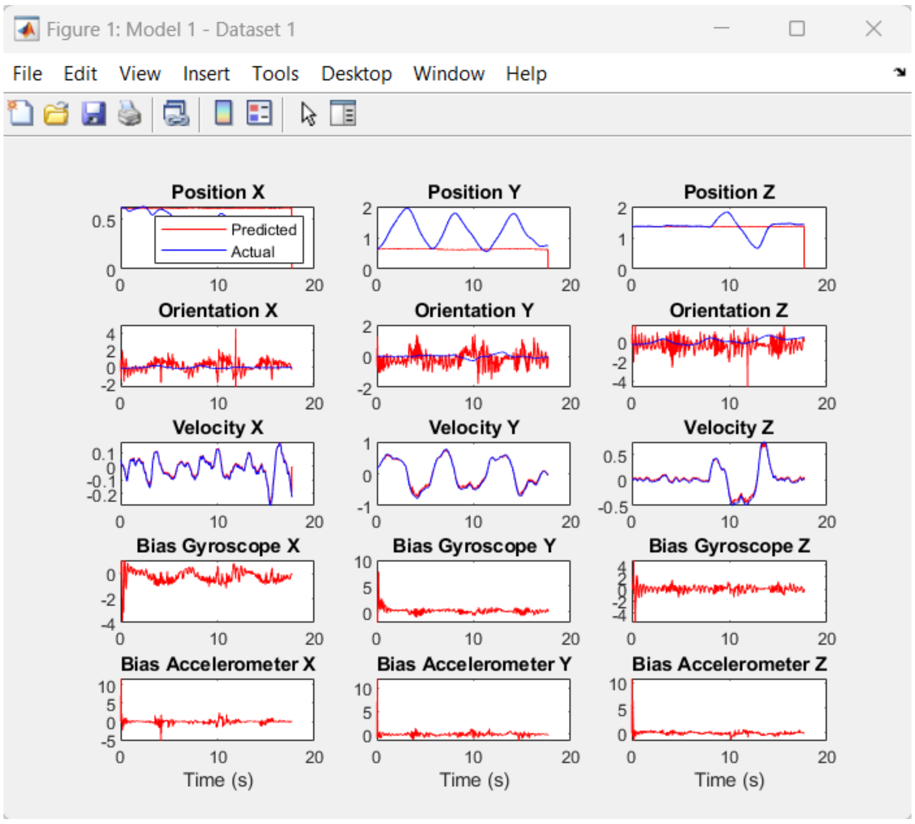
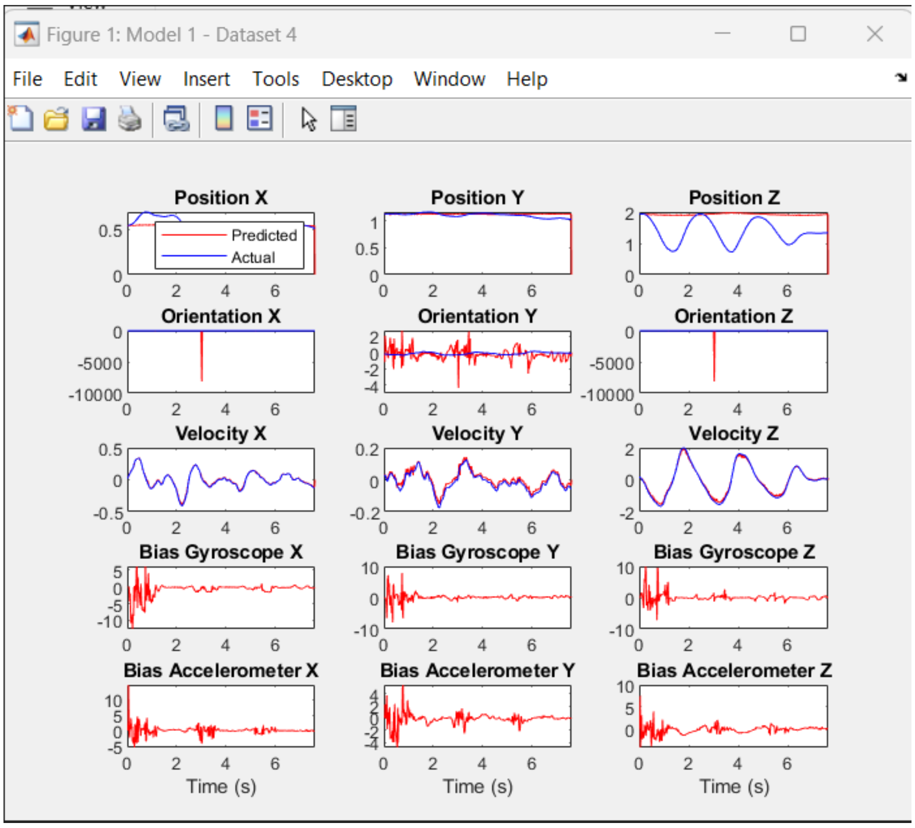


Figure 1: Model 1 - Dataset 1
File Edit View Insert Tools Desktop Window Help
Position X
Position Y
Position Z
Predicted
Actual
Orientation X
Orientation Y
Orientation Z
Velocity X
Velocity Y
Velocity Z
Bias Gyroscope X
Bias Gyroscope Y
Bias Gyroscope Z
Bias Accelerometer X
Bias Accelerometer Y
Bias Accelerometer Z
Time (s)
Time (s)
Time (s)



6 Conclusion and References

In conclusion, we can say that Kalman Decomposition and EKF can be very useful in the activity of state estimation of a robot. It is vital to recognise the exact position, orientation, velocity, acceleration and bias of the robot so that the robot can be accurately navigated and controlled. Also, we learnt about how MATLAB can be helpful to parse data from the sensor of the robot and as a display software for graphical use. My main source of reference for the project was Dr. Vijay Kumar's paper distributed in our class, Professor Loianno's notes and some guidance from my friend Abhimanyu Suthar.