ECE 8540 Analysis of Tracking System

Lab 4 report
Kalmann Filter

Harshal B. Varpe
October 13, 2022

1 Introduction

In his lab report, we deal with implementing the Kalman filter. A Kalman filter is an optimal filter used to optimally estimate the variables of interest that cannot be measured directly. We use model fitting techniques to mitigate the noise in sensor data. However, model fitting techniques can only be used when there is no change in the system's behavior. In the case of tracking systems, the object of interest is not expected to follow the same behavior. Otherwise, there will be no need for a tracking system, and the object's behavior can be perfectly modeled. The Kalman filter weights the accuracy of predictions and measurements to produce the best results.
For this lab, students were given two sets of data. One set of data was 1D, and the second set of data was 2D. The report will briefly discuss the derivation of the Kalman filter, the implementation of the Kalman filter, and the effects of dynamic noise and measurement noise on the accuracy of results.

2 Methods

The Kalman filter functions in a continuous loop of predictions and updates. The whole cycle is explained below in terms of equations that can be implemented.

1. Predict a new state:

$$X_{t,t-1} = \Phi X_{t-1,t-1} \tag{1}$$

2. Predict next state covariance:

$$S_{t,t-1} = \Phi S_{t-1,t-1} \Phi^T + Q \tag{2}$$

3. Obtain measurement(s): Y_t
4. Calculate the Kalman gain:

$$K_t = S_{t,t-1} M^T [M S_{t,t-1} M^T + R]^{-1} \tag{3}$$

5. Update the state

$$X_{t,t} = X_{t,t-1} + K_t (Y_t - M X_{t,t-1}) \tag{4}$$

6. Update state covariance

$$S_{t,t} = [I - K_t M] S_{t,t-1} \tag{5}$$

7. Loop (now t becomes $t+1$)

Where,
Φ: State transition matrix
S_t : State estimate covariance
Y_t : Sensor measurement
M : Observation matrix
K_t : Kalman gain
Q : Dynamic noise covariance
R : Measurement noise covariance

In the first step, initial values for states, state covariances, Dynamic noise covariance, and measurement noise covariance need to be defined. These values are to be chosen based on available information.

2.1 1D model

For the first problem, we are to develop a Kalman filter for the 1D model.

1. State variables:

$$X_t = \begin{bmatrix} x_t \\ \dot{x}_t \end{bmatrix} \tag{6}$$

We are interested in both the position and the velocity. Hence, the state variables are x_t representing position and $\dot{x_t}$ representing velocity.
For the 1D model, the state transition equations are modeled as follows.

2. State transition equation:

$$x_{t,t} = X_{t,t-1} + \dot{X}_{t,t-1}T \tag{7}$$

$$\dot{x}_{t,t} = \dot{X}_{t,t-1} \tag{8}$$

Here T represents the time interval between new measurements. We can write a state transition matrix Φ from the above state transition equations.

$$X_{t+1} = \Phi X_t + A_t \tag{9}$$

The above equation is another way to write a state transition equation. In this equation, the X_t is the current actual state, X_{t+1} is the next actual state, and A_t is the random dynamics during the sensing interval. This equation in matrix form is given below. Here Φ is a four-by-four matrix called a state transition matrix.

$$\begin{bmatrix} x_{t+1} \\ \dot{x}_{t+1} \end{bmatrix} = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_t \\ \dot{x}_t \end{bmatrix} + \begin{bmatrix} 0 \\ N(0, \sigma_a^2) \end{bmatrix} \tag{10}$$

The value of T is set to be one since the time between each new measurement is one second. Here $N(0, \sigma_a^2)$ represents the dynamic noise. We can rewrite this noise as dynamic noise covariance Q. There is no dynamic noise on the position portion of the transition equation; its variance is zero. Therefore, only the velocity portion of dynamic noise has a covariance term.

$$Q = \begin{bmatrix} 0 & 0 \\ 0 & \sigma_a^2 \end{bmatrix} \tag{11}$$

Now that we have defined state variable equations, Dynamic Noise Covariance, and state transition matrix, we describe the state estimate covariance matrix. This matrix represents the level of uncertainty in the state variables in equation 6. The state estimate covariance matrix S_t is given by:

$$S_t = \begin{bmatrix} \sigma_x^2 & \sigma_{x,\dot{x}} \\ \sigma_{x,\dot{x}} & \sigma_{\dot{x}}^2 \end{bmatrix} \tag{12}$$

The diagonal elements represent the uncertainty in the position and velocity estimates, and the off-diagonal elements represent their covariances.

3. Measurement equation:

$$Y_t = MX_t + N_t \tag{13}$$

where Y_t is the measurements, X_t is the actual state of the systems at time t, and M is the observation matrix.N_t is the measurement noise covariance. For a 1D problem, this equation can be written as:

$$\begin{bmatrix} Y_t \end{bmatrix} = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_t \\ \dot{x}_t \end{bmatrix} + \begin{bmatrix} N(0, \sigma_n^2 \end{bmatrix} \tag{14}$$

We can write the measurement noise covariance R from the measurement equation. Since there is only one measurement, the matrix contains only the variance of the measurement noise of position.

$$R = \begin{bmatrix}\sigma_n^2\end{bmatrix} \tag{15}$$

4. State prediction equation:

$$X_{t+1} = \Phi X_t \tag{16}$$

We have already derived the value for Φ.

5. State update equation:

$$X_{t,t} = X_{t,t-1} + K_t(Y_t - MX_{t,t-1}) \tag{17}$$

where K_t is Kalman gain.

These two matrices defined above control how the filter behaves. The ratio of the dynamic noise covariance matrix to the measurement noise covariance matrix control how closely the filter follows the measurements or the prediction.

2.2 2D Kalman filter

For the second problem, we use a 2D constant velocity model.

1. State variables:

$$X_t = \begin{bmatrix} x_t \\ y_t \\ \dot{x}_t \\ \dot{y}_t \end{bmatrix} \tag{18}$$

where x_t and y_t are the positions and $\dot{x}_t$ and $\dot{y}_t$ are the velocities.

2. State transition equation:

$$x_{t+1} = x_t + T\dot{x}_t \tag{19}$$

$$y_{t+1} = y_t + T\dot{y}_t \tag{20}$$

$$\dot{x}_{t+1} = \dot{x}_t \tag{21}$$

$$\dot{y}_{t+1} = \dot{y}_t \tag{22}$$

From these equations, we can define the state variable covariance matrix S_t.

$$S_t = \begin{bmatrix} \sigma_{x_t}^2 & \sigma_{x_t,y_t} & \sigma_{x_t,\dot{x}_t} & \sigma_{x_t,\dot{y}_t} \\ \sigma_{x_t,y_t} & \sigma_{y_t}^2 & \sigma_{y_t,\dot{x}_t} & \sigma_{y_t,\dot{y}_t} \\ \sigma_{x_t,\dot{x}_t} & \sigma_{y_t,\dot{x}_t} & \sigma_{\dot{x}_t}^2 & \sigma_{\dot{x}_t,\dot{y}_t} \\ \sigma_{x_t,\dot{y}_t} & \sigma_{y_t,\dot{y}_t} & \sigma_{\dot{x}_t,\dot{y}_t} & \sigma_{\dot{y}_t}^2 \end{bmatrix} \tag{23}$$

3. State Transition matrix:

From the state transition equation, we can write the state transition matrix Φ as follows:

$$\Phi = \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{24}$$

4. Measurement Equations:
We know that data was collected in the x and y direction. Hence, we define the observation variables as follows:

$$Y_t = \begin{bmatrix} \tilde{x}_t \\ \tilde{y}_t \end{bmatrix} \tag{25}$$

We write the observation equation as

$$\tilde{x}_t = x_t \tag{26}$$

$$\tilde{y}_t = y_t \tag{27}$$

From the above equation, we can write the observation matrix M.

$$M = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{28}$$

5. Dynamic Noise covariance (Q) and Measurement covariance (R):

For this problem, we write the dynamic noise covariance Q as :

$$Q = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_{a1}^2 & \sigma_{a1,a2} \\ 0 & 0 & \sigma_{a1,a2} & \sigma_{a2}^2 \end{bmatrix} \tag{29}$$

We write the measurement covariance matrix R as:

$$R = \begin{bmatrix} \sigma_{n1}^2 & \sigma_{n1,n2} \\ \sigma_{n1,n2} & \sigma_{n2}^2 \end{bmatrix} \tag{30}$$

In this problem, we try to tune the behavior of the Kalman filter by adjusting the ratio of Dynamic Noise Covariance to Measurement Covariance.

3 Result

Given data for both problems was iterated over in a for loop, equations 1 through 5 were used to create a Kalman filter in MATLAB. The MATLAB code can be seen in the appendix section of the report.

3.1 Results for 1D data set

We were to test the filter for three different ratios of Dynamic Noise Covariance to Measurement noise covariance. I kept the measurement noise covariance R fixed at one and changed the dynamic noise covariance Q to 1000, 0.0001, and 0.1. We can see the graphs for each of the ratios below.

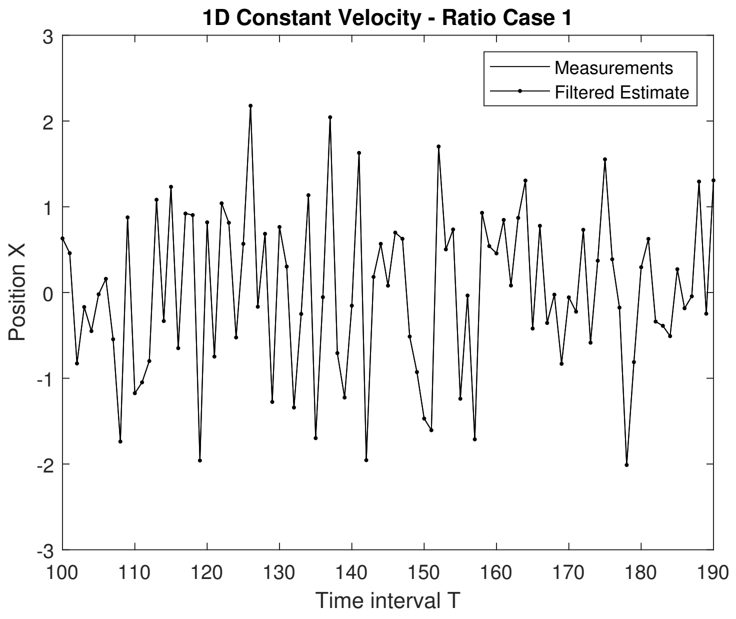

Figure 1: Plot of measurements and filtered estimate for Q=1000 and R=1

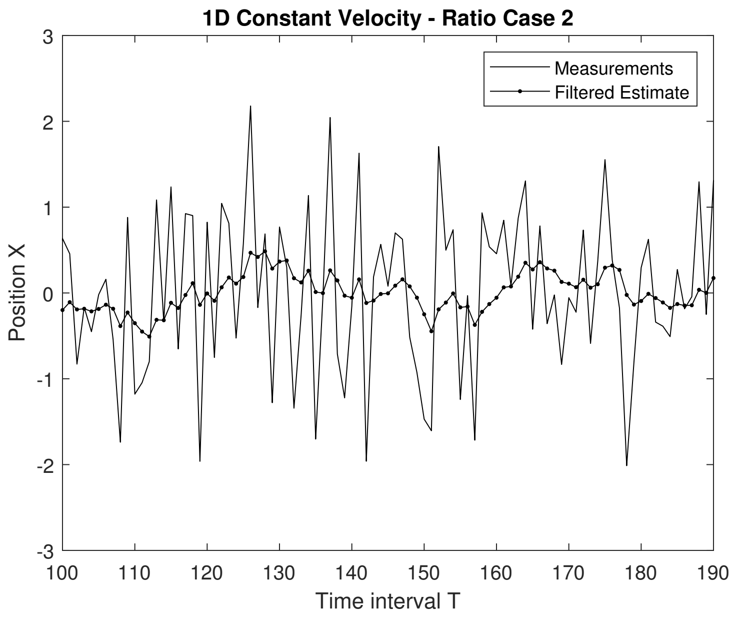

Figure 2: Plot of measurements and filtered estimate for Q=0.0001 and R=1

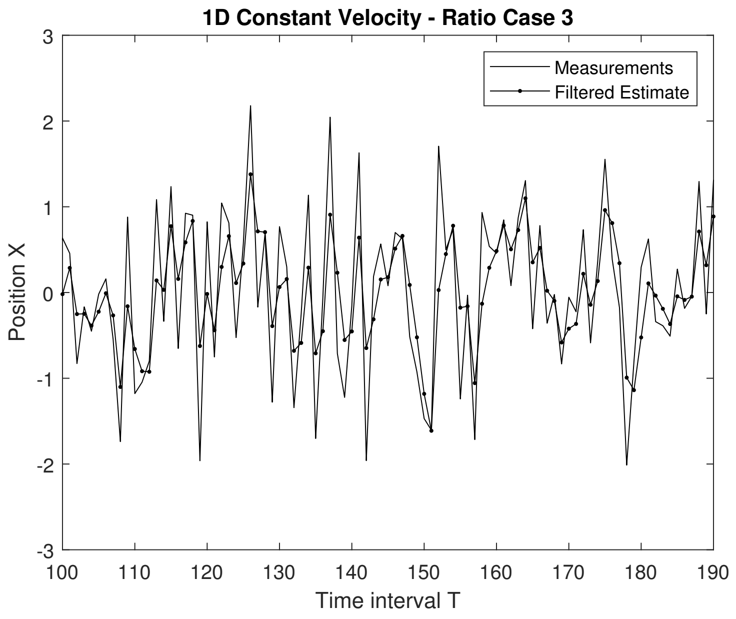

Figure 3: Plot of measurements and filtered estimate for Q=0.1 and R=1

Let us start with figure 1. In figure 1, we can see that the filtered estimate follows the measurements very closely. It is evident that we can only see one line in the plot. This behavior is as expected. Since the value of Q is very high compared to the value of R, we are deciding to trust the measurements more than the predictions.
In figure 2, We have a very low value of Q compared to the value of R. Here, we are deciding to trust the predictions more than the measurements. Therefore, the behavior in the plot is as expected; the filtered estimate does not follow the measurements as it did in figure 1.
In figure 3, We have $Q = 0.1$ and $R = 1$. Here the filtered estimate does not follow measurements closely, nor does it become flat.

3.2 Result for 2D data set

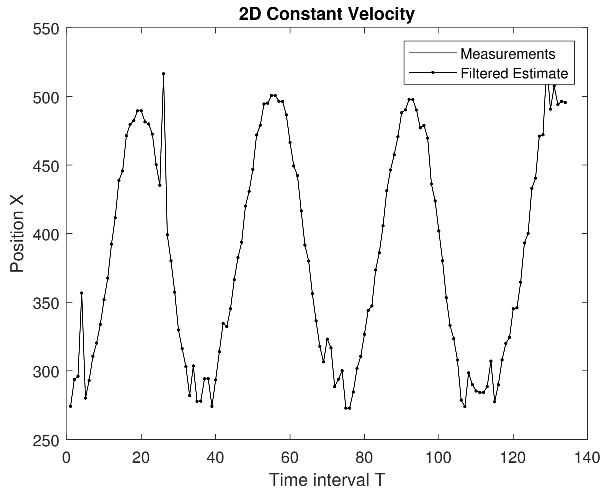

Figure 4: Plot of measurements and filtered estimate for Q=0.1 and R=0.001 of X position

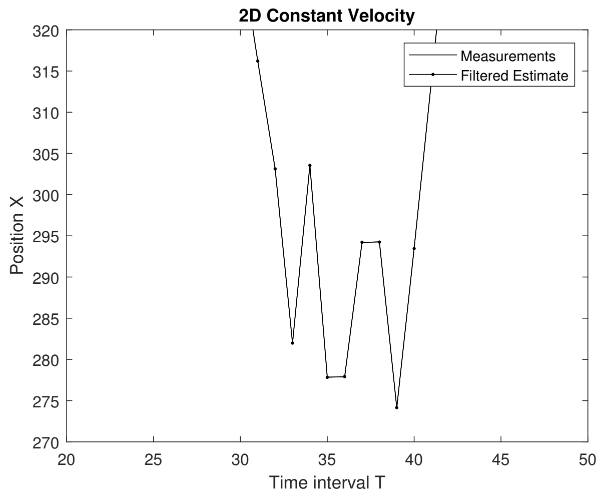

Figure 5: Zoomed in Plot of measurements and filtered estimate for Q=0.1 and R=0.001 of X position

Figures 4 and 5 show the same plot of measurement against the filtered estimate of the X position for 2D data. However, Figure 5 is zoomed in to show that the filtered estimate follows the measurements very closely. For this plot, the value of Dynamic Noise Covariance is 100 times higher than the Measurement Noise Covariance. Hence the behavior of the plot is justified. Figures 6 and 7 show the same observation for the plot of the Y position.

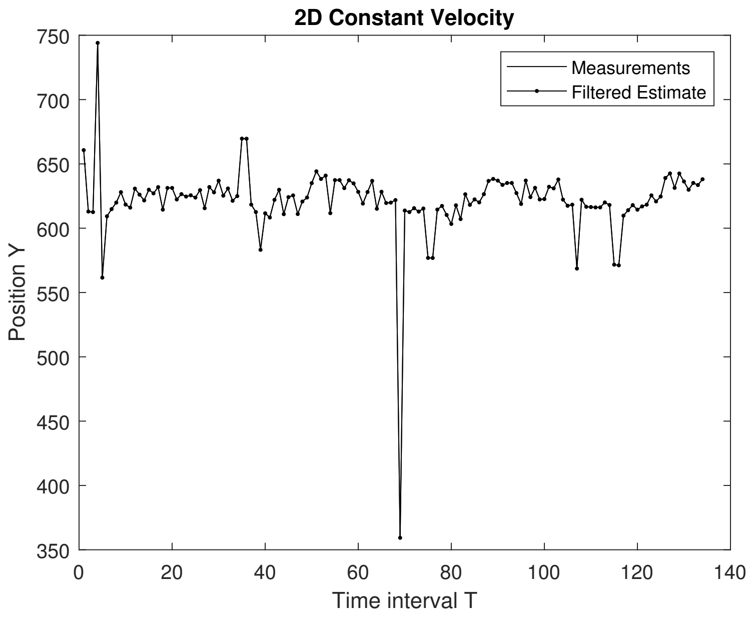

Figure 6: Plot of measurements and filtered estimate for Q=0.1 and R=0.001 of Y position

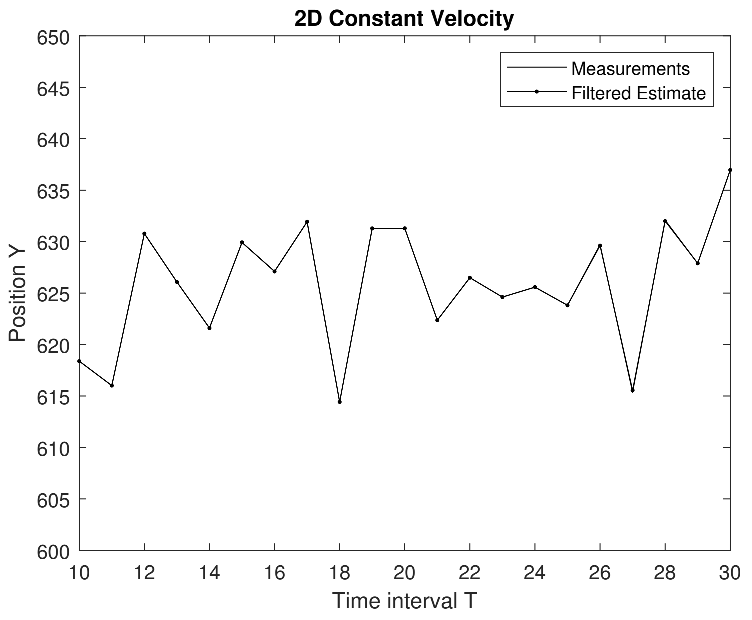

Figure 7: Zoomed in Plot of measurements and filtered estimate for Q=0.1 and R=0.001 of Y position

Figures 8 and 9 show the behavior of the Kalman filter when the Q and R values are set equal. However, when I set the values of Dynamic Noise Covariance and Measurement noise covariance values equal for this data set, the filtered estimate jumps around a lot. Despite the jumping around, it still tries to follow the trend shown in the measurements. This behavior is observed for both

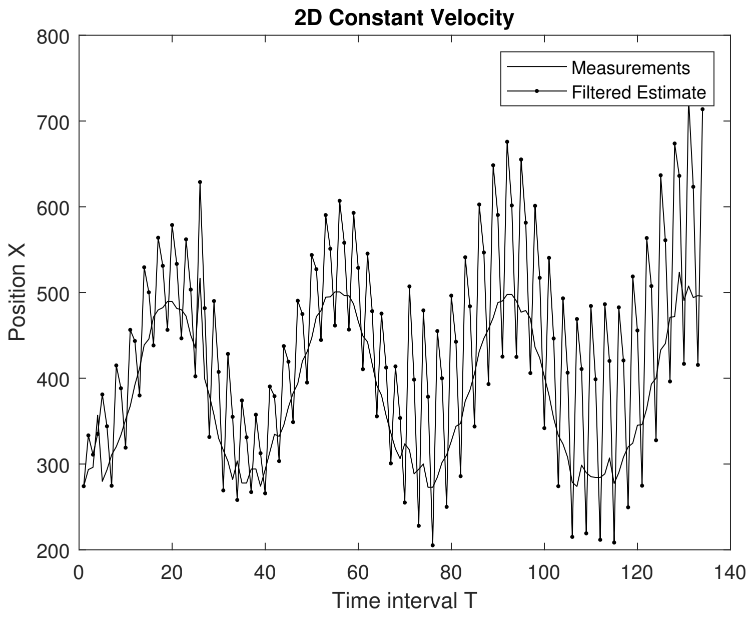

Figure 8: Plot of measurements and filtered estimate for Q=0.1 and R=0.1 of X position of Y position

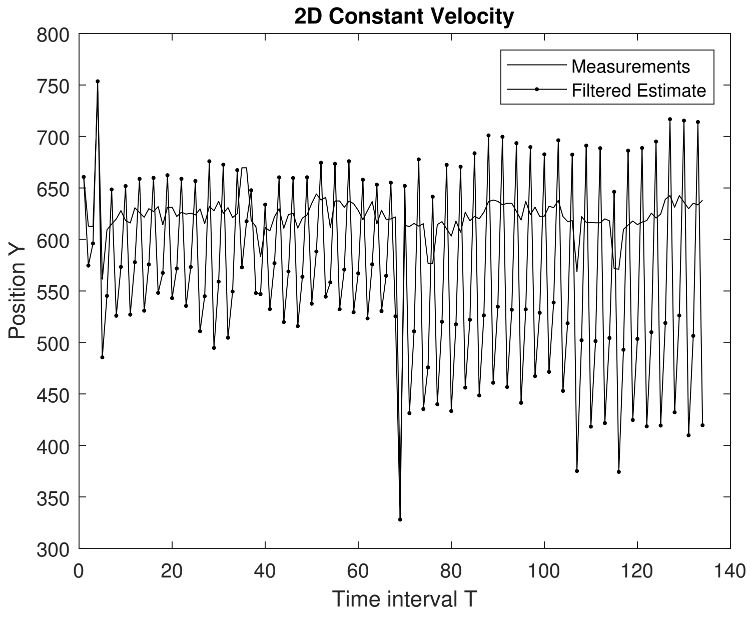

Figure 9: Plot of measurements and filtered estimate for Q=0.1 and R=0.1

X-position and Y-position plots.

4 Conclusion

Kalman filter is an optimal filter that relies on weights to produce the best results. These weights are the Dynamic Noise Covariance matrix Q and the Measurement Noise Covariance matrix R. The higher the value of Dynamic noise, the less weight is given to the predictions made, and more weight is given to measurements. Thus, the filtered output will follow the measurements closely. Similarly, if the measurement noise is higher than the dynamic noise, then the filtered estimate will follow predictions made more closely. A proper ratio is needed for better performance of the Kalman Filter.

5 Apendix

MATLAB code for part 1

```
clear all
clc
%% Read data files
data_1d = importdata('1D-data.txt');


%%Part 1 - 1D data
T = 1; % time interval of 1 second
phi = [1 T;0 1]; % State transition matrix
Q = [0 0;0 0.0001] ; % Dynamic Noise covariance
R = 1 ; % Measurement noise
M = [1 0]; % Observation Matrix
X_p = [0;0]; % Previous state matrix / initial state matrix
S_p = [1 0;0 1]; % state covariance
plt_data = zeros(1,length(data_1d)); % output data

for t = 1:1:length(data_1d)
    Yt(t) = data_1d(t);
    X_n = phi * X_p ;
    S_n = (phi * S_p * phi') + Q ;
    Kt = S_n * M'/((M*S_n*M')+ R);
    X_p = X_n + Kt * (Yt(t) - M*X_n);
    S_p = (eye(2) - Kt*M) * S_n ;
    plt_data(t) = X_p(1);
end

figure

plot((1:length(data_1d)),data_1d,'k')
hold on;
plot((1:length(data_1d)),plt_data, 'k.-','linewidth',0.5)
axis([100 190 -3 3])
xlabel("Time interval T");
ylabel("Position X");
legend("Measurements","Filtered Estimate");
title("1D Constant Velocity - Ratio Case 2")
```

Matlab code for part 2 -

```
clear all
clc
%% Read data files
data_2d = importdata('2D-UWB-data.txt');

%%Part 1 - 1D data
T = 1; % time interval of 1 second
phi = [1 0 T 0;0 1 0 T; 0 0 1 0;0 0 0 1]; % State transition matrix

Q = [0 0 0 0;0 0 0 0;
     0 0 0.1 1; 0 0 1 0.1] ; % Dynamic Noise covariance

R = [10 0; 0 10] ; % Measurement noise
M = [1 0 0 0; 0 1 0 0]; % Observation Matrix

X_p = [data_2d(1,1);data_2d(1,2);0;0]; % Previous state matrix / initial state matrix
S_p = eye(4); % state covariance
plt_data = zeros(length(data_2d(:,1)),2); % output data

for t = 1:1:length(data_2d(:,1))
     Yt(1,1) = data_2d(t,1);
     Yt(2,1) = data_2d(t,2);
     X_n = phi * X_p ;
     S_n = (phi * S_p * phi') + Q ;
     Kt = S_n * M'/((M*S_n*M')+ R);
     X_p = X_n + Kt * (Yt - M*X_n);
     S_p = (eye(4) - Kt*M) * S_n ;
     plt_data(t,1) = X_p(1,1);
     plt_data(t,2) = X_p(2,1);
end
%
figure
plot((1:length(data_2d(:,1))),data_2d(:,1),'k')
hold on;
plot((1:length(data_2d)),plt_data(:,1) , 'k.-','linewidth',0.5)
% axis([20 50 270 320])
xlabel("Time interval T");
ylabel("Position X");
legend("Measurements","Filtered Estimate");
title("2D Constant Velocity")


figure
plot((1:length(data_2d(:,1))),data_2d(:,2),'k')
hold on;
plot((1:length(data_2d)),plt_data(:,2) , 'k.-','linewidth',0.5)
% axis([10 30 600 650])
xlabel("Time interval T");
ylabel("Position Y");
```

```
legend("Measurements","Filtered Estimate");
title("2D Constant Velocity")
```