SSY130 - Hand in Problem 3

Alfred Aronsson

December 12, 2023

Student ID: 20000729, Secret Key: 'Landorus'

1

We are given a state space defined by:

$$s(t) = \begin{bmatrix} s_1(t) \\ s_2(t) \\ s_3(t) \\ s_4(t) \end{bmatrix} = \begin{bmatrix} x(t) \\ \dot{x}(t) \\ y(t) \\ \dot{y}(t) \end{bmatrix}$$

Further we that velocities are constant leading to:

$$\dot{s}_1(t) = s_2(t) \tag{1}$$
$$\dot{s}_2(t) = 0 \tag{2}$$
$$\dot{s}_3(t) = s_4(t) \tag{3}$$
$$\dot{s}_4(t) = 0 \tag{4}$$

The first task in this question is to derive a discrete state space model looking like:

$$s(k+1) = As(k) + w(k)$$

According to the problem definition this should be done by applying the finite difference approximation,

$$\dot{x}(t)|_{t=kT} \approx \frac{x(kT+Y) - x(kT)}{T}, \text{ where } T \text{ is the sampling time}$$

to equation (1) through (4). Producing the following results in accordance with equation (12.2) in the lecture notes:

$$s_1(k+1) - s_1(k) = s_2(k) \cdot T \tag{5}$$
$$s_2(k+1) - s_2(k) = 0 \tag{6}$$
$$s_3(k+1) - s_3(k) = s_4(k) \cdot T \tag{7}$$
$$s_4(k+1) - s_4(k) = 0 \tag{8}$$

From equations (5) through (8) we can extrapolate matrix A.

$$A = \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The second part of this question is to derive the measurement equation:

$$z(k) = Cs(k) + v(k)$$

From the problem definition states s_2 and s_3 can be recognized as the additive noises $v_x(k)$ and v_y respectively. Building on this the C matrix can once again be identified with equations (5) through (8):

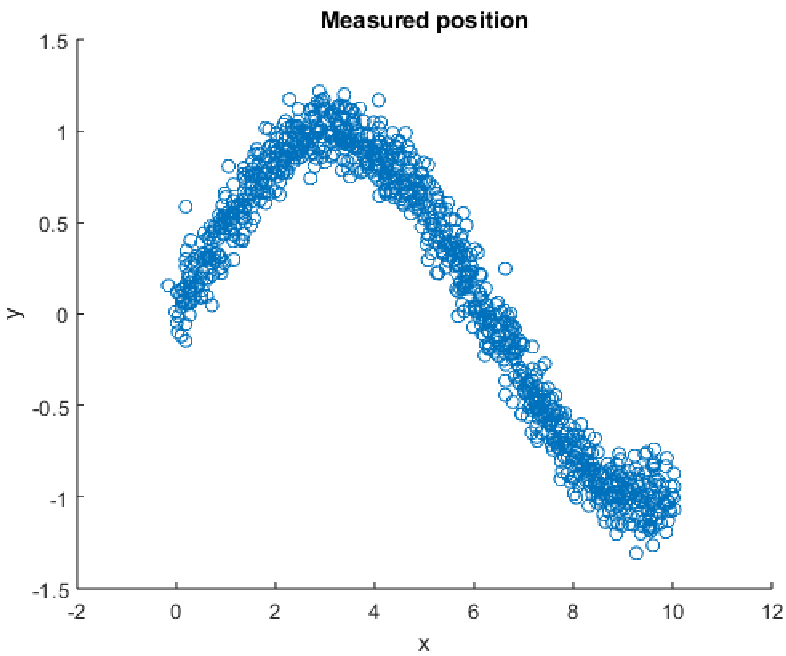
$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

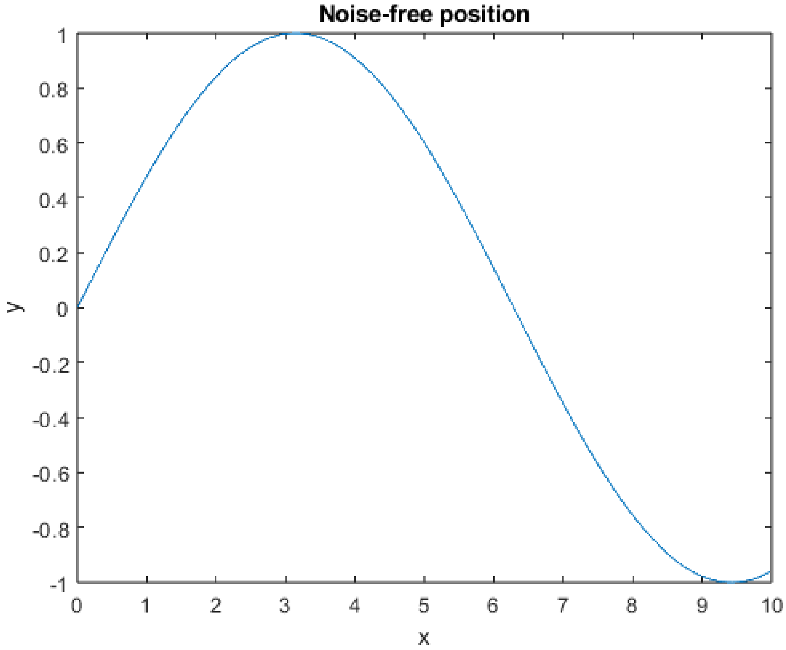
All in all we now have:

$$s(k+1) = As(k) + w(k)$$
$$z(k) = Cs(k) + v(k)$$
$$\text{where, } w(k) \sim WGN(0, Q) \text{ and } v(k) \sim WGN(0, R)$$

2

The plots for measured and true position respectively:





3

Kalman filter implementation:

```
function [Xfilt,Pplus] = kalm_filt(Y,A,C,Q,R,x0,P0)

        % [Xfilt,Pplus] = kalmfilt(Y,A,C,Q,R,x0,P0)
        % Matlab function for Kalman filtering
        % Inputs:
        %   Y    Measured signal, matrix of size [p,N] where N is the number
        %        of samples and p the number of outputs
        %   A    System dynamics matrix, size [n,n]
        %   C    Measurement matrix, size [p,n]
        %   Q    Covariance matrix of process noise, size [n,n]
        %   R    Covariance matrix of measurement noise, size [p,p]
        %   x0   Estimate of x(0), size [n,1]. Defaults to a zero vector if
        %        not supplied.
        %   P0   Error covariance for x(0), size [n,n]. Defaults to the
        %        identity matrix if not supplied.
        % Outputs:
        % Xfilt Kalman-filtered estimate of the state, size [n,N]
        % Pplus Covariance matrix for last sample, size [n,n]

        [p,N] = size(Y);         % N = number of samples, p = number of "sensors"
        n = length(A);           % n = system order
```

```
    Xpred = zeros(n,N+1);    % Kalman predicted states
    Xfilt = zeros(n,N);      % Kalman filtered states (after using the meeasurment)

    if nargin < 7
        P0=eye(n);           % Default initial covariance
    end
    if nargin < 6
        x0=zeros(n,1);       % Default initial states
    end

    % Filter initialization:
    Xpred(:,1) = x0;         % Index 1 means time 0
    P = P0;                  % Initial covariance matrix (uncertainty)

    % Kalman filter iterations:
    for t=1:N
        % Filter update based on measurement
        % Xfilt(:,t) = Xpred(:,t) + ...
        % Implementation according to lecture notes
        K = P * C' / (C * P * C' + R);
        Xfilt(:, t) = Xpred(:, t) + K * (Y(:, t) - C * Xpred(:, t)); %TODO: This line is m

        % Uncertainty update
        Pplus = (eye(n) - K * C) * P; %TODO: This line is missing some code!

        % Prediction
        Xpred(:, t + 1) = A * Xfilt(:, t); %TODO: This line is missing some code!

        % Uncertainty propagation
        P = A * Pplus * A' + Q; %TODO: This line is missing some code!
    end
end
```

4

The last task is to implement the Kalman filter and produce some plots. To determine the R matrix I found the variance of the measured x and y positions respectively using Matlab. With the variances i defined R as:

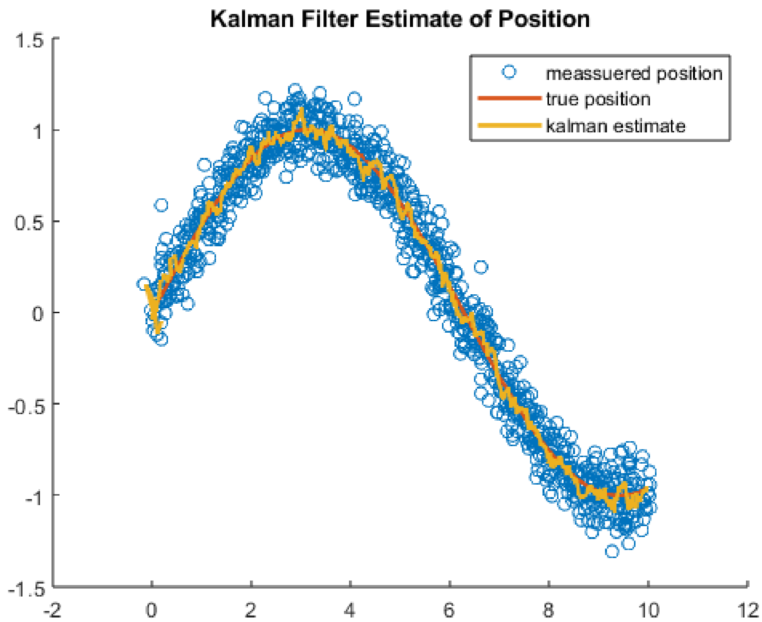
$$R = \begin{bmatrix} 8.315 & 0 \\ 0 & 0.521 \end{bmatrix} \tag{9}$$

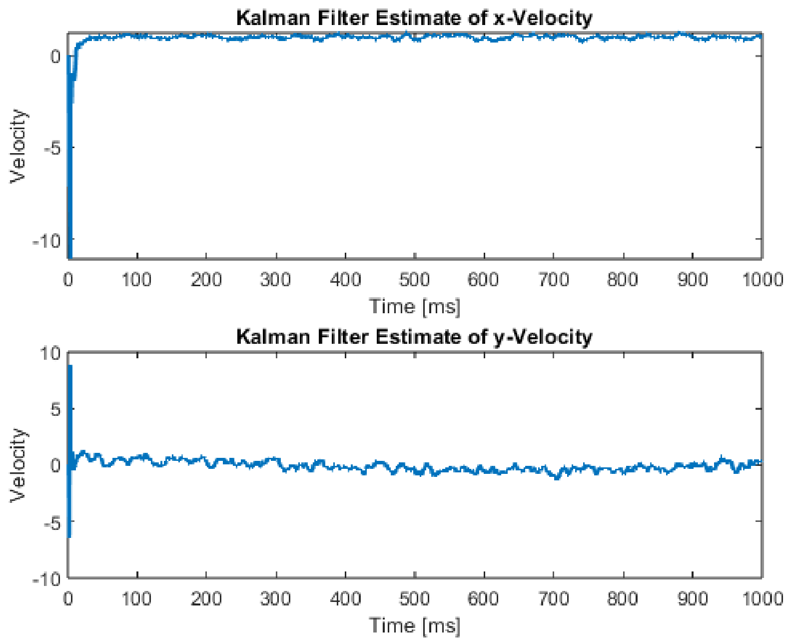
When it comes to the Q matrix there isn't very much information to use. We know that the positions are very trustworthy and therefore their elements in the diagonal matrix are set to 0. With states s_2 and s_4 however, it is not obvious what the values should be. I will begin with setting them to 1 and then evaluate the results and tune there after.

Q is defined as:

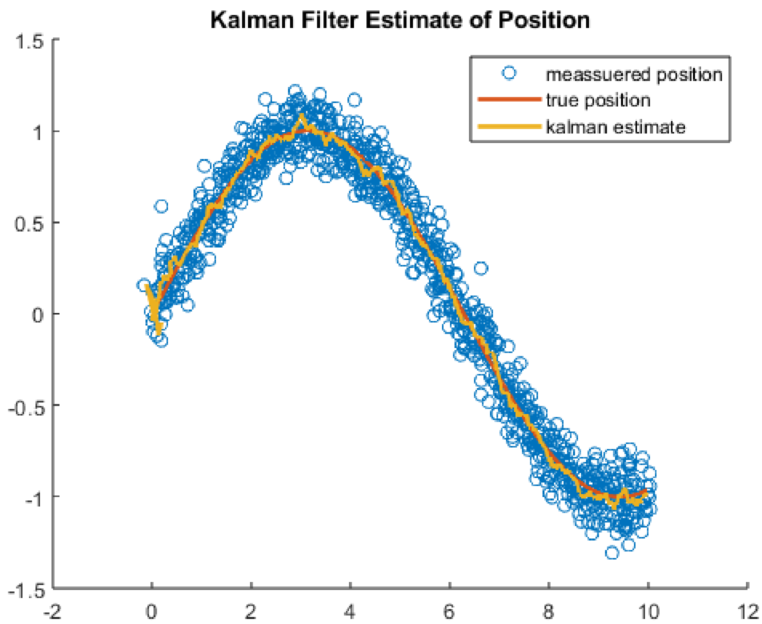
$$Q = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{10}$$

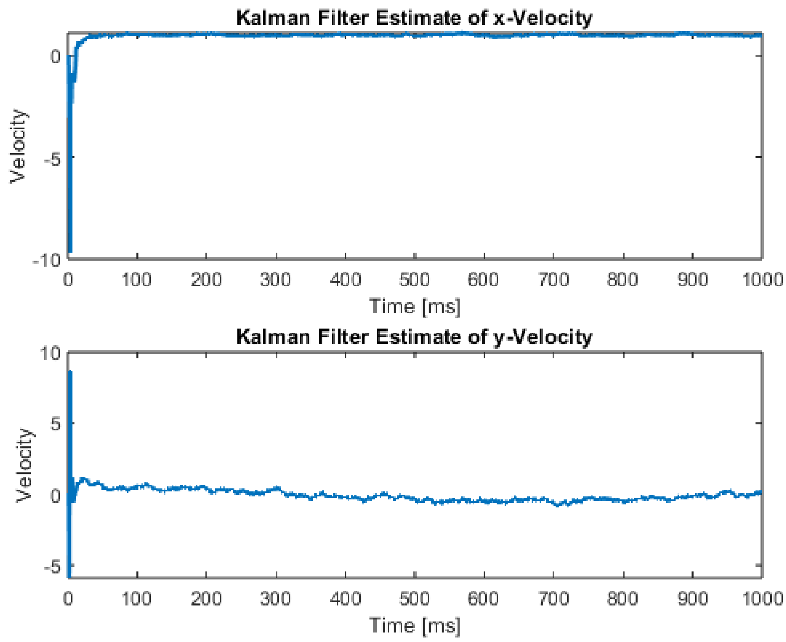
The problem now states that the Kalman estimator now should be tuned with multiplying the R matrix with different powers of 10.
These are the plots with scaling 10^0:



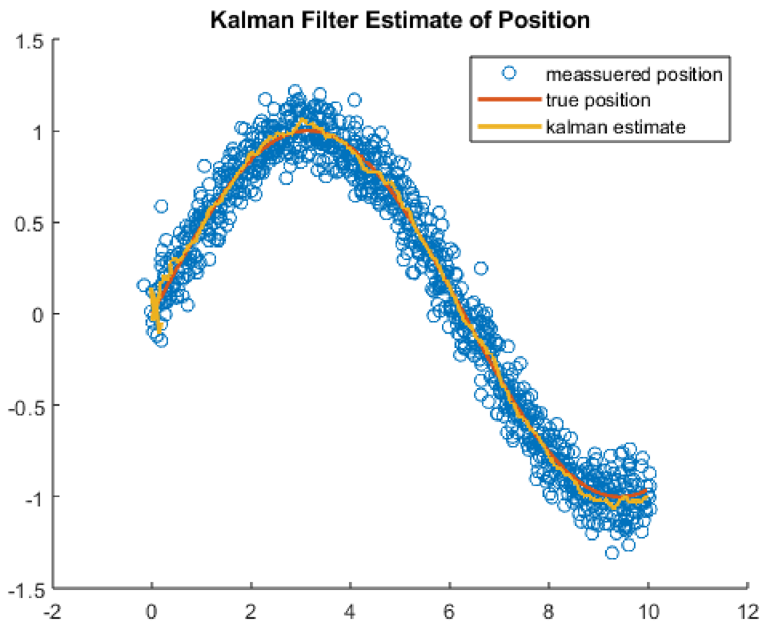


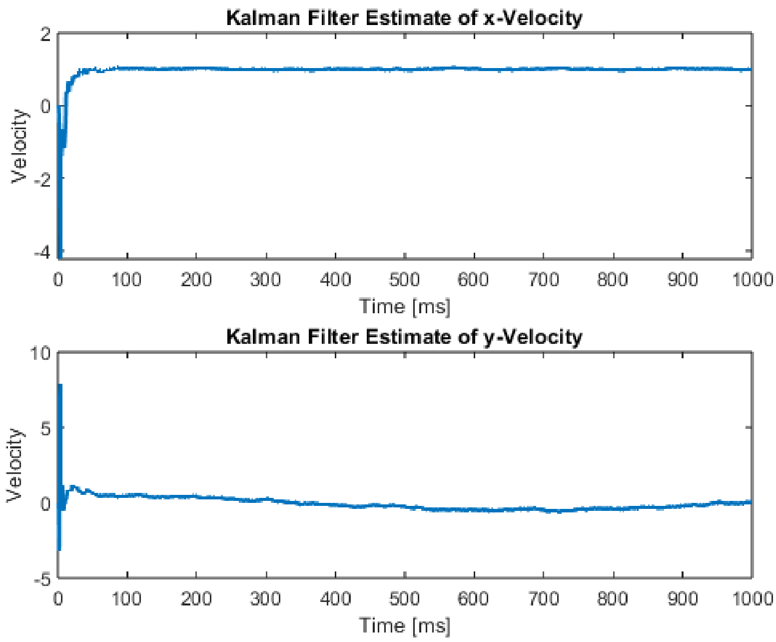
These are the plots with scaling 10^1:



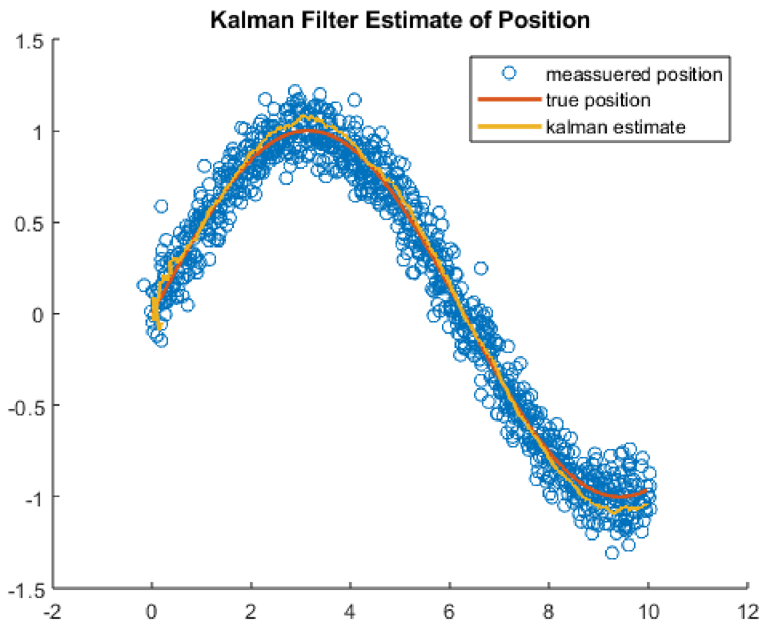


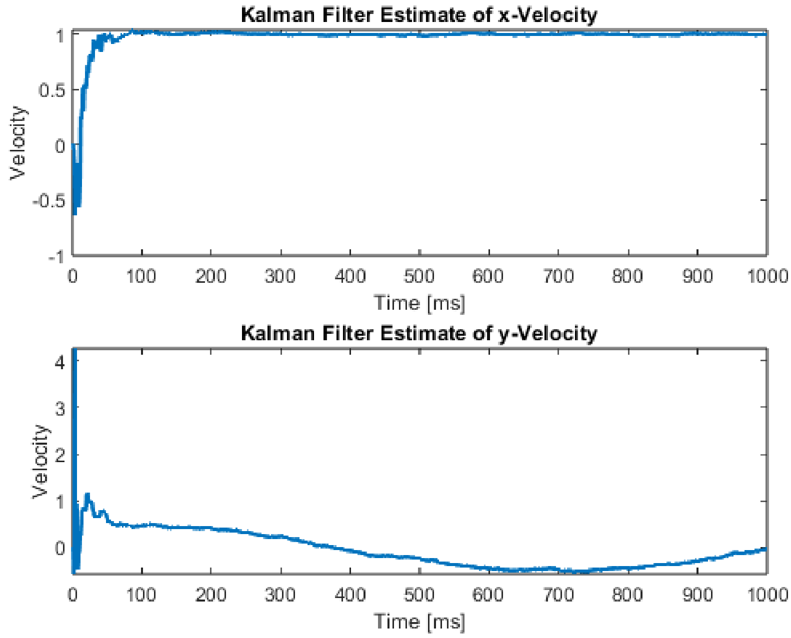
These are the plots with scaling 10^2:





These are the plots with scaling 10^3:





As can be observed, the Kalman filter is best tuned with the scaling 10^2.