École Centrale de Nantes

Statistical Signal Processing and Estimation Theory Lab Report

Student Name: Tailei Wang, Hao Deng

Date: 26/12/2021

1. Aims/Objectives

The aim of this lab is to enable us to acquire a better understanding of the constructions of Kalman Filter and Stationary Kalman Filter. Kalman filter and Stationary Kalman filter are specified cases of the Bayes filter. Kalman filter is a recursive implementation of the MMSE (Minimum Mean Square Estimator) applied to the Gaussian linear model. This lab indicates the application of Kalman filter and Stationary Kalman filter in terms of velocity control on a DC motor for a given input voltage $u(t)$. The angular position $\theta(t)$ of the rotor and the angular velocity $\Omega(t)$ are estimated in the Kalman filters based on $u(t)$ and $y(t)$. The estimated results of velocity and position are plotted, and the estimated results are compared under rough model and perfect model in the simulation section.

2. Introduction

One specific case of the Bayes filter is Kalman filter, which is a recursive implementation of the MMSE applied to the Gaussian linear model. If the Gaussian assumption on the sequences v, w or the initial state is removed, the Kalman filter becomes an implementation of the LMMSE (Linear Minimum Mean Square Estimator) estimator. In the given model, it is shown that the initial state and sequence w is distributed uniformly. Thus, the Kalman filter in this motor case can be treated as an implementation of LMMSE estimator. That is defined as:

$$\hat{x}_{MMSE}(Y) = E(X|Y)$$

$$\hat{x}_{LMMSE}(Y) = m_X + C_{X,Y}C_{Y,Y}^{-1}(Y - m_Y)$$

In which the Kalman gain is defined as $C_{X,Y}C_{Y,Y}^{-1}$.

And the Linear model is given as below:

$$Y[n] = H_nX[n] + h_n + W[n]$$
$$X[n+1] = F_nX[n] + f_n + V[n]$$

One important step during the estimator prediction is the recursion, the recursion inside the Kalman filter can be construed as below.

$$Input:\ \hat{x}^{|n-1}[n]\ with\ P^{|n-1}[n]$$
$$Inside\ the\ kalman\ filter:\ \hat{x}^{n}[n]\ with\ P^{|n}[n]$$
$$Output:\ \hat{x}^{|n}[n+1]\ with\ P^{|n}[n+1]$$

Stationary Kalman filter is a highly simplified filter, which is obtained by replacing the time varying Kalman gain K[n] with its limit K[∞].

3. Simulations and Results

3.1 Input voltage

The input voltage is defined as a zero-mean square wave with period 100ms and the peak-to-peak amplitude 0.1V. Figure 1 is the generated waveform of the input voltage to the DC motor model.

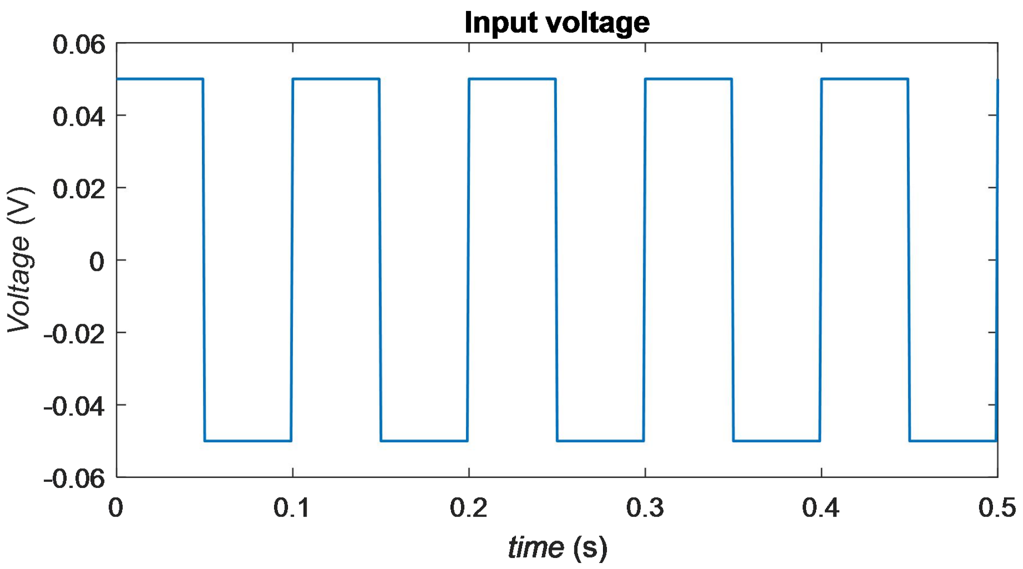

Figure 1 Input voltage waveform

3.2 System modelling and simulation

In this lab, the DC motor is treated as a linear model. With a state vector $x(t) = \begin{bmatrix} \theta(t) \\ \Omega(t) \end{bmatrix}$, the linear model is defined as below:

$$\begin{cases} \dot{x} = \begin{bmatrix} 0 & 1 \\ 0 & -1/T \end{bmatrix} x + \begin{bmatrix} 0 \\ G/T \end{bmatrix} u \\ \theta = \begin{bmatrix} 1 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \end{bmatrix} u \end{cases}$$

The input-output signals are sampled with a sample time T_s, and the input voltage $u(t)$ is a square wave, which means it is constant between two sampling times. Therefore, the zero-order-hold method is applied here to convert the model from continuous time domain into discretized domain.
The angular position is measured with an incremental encoder (precision L = 512 angles per lap), which provides the measure $y(t)$ of $\theta(t)$. The output y can be defined as $y = round(teta * L/2/pi) * 2 * pi/L$. After the simulation function is built, the simulator is then tested with $G = 50\, rad \cdot s^{-1} \cdot V^{-1}$ and $T = 20ms$.
Figure 2 shows the simulation results of the angular position (actual and measured) and the angular velocity of the rotor. It can be observed that the measure $y(t)$ from the incremental encoder is a quantization of the actual angular position $\theta(t)$.

3.3 Kalman filter manufacture

The system uses an incremental encoder to measure the angular position. During the measurement, a quantization noise occurs (noted as ω_n). The variance of the quantization noise is r. To take into account of the modelling errors, the actual input is assumed to be $u_n + v_n$. Here, v_n is a white noise with variance q.

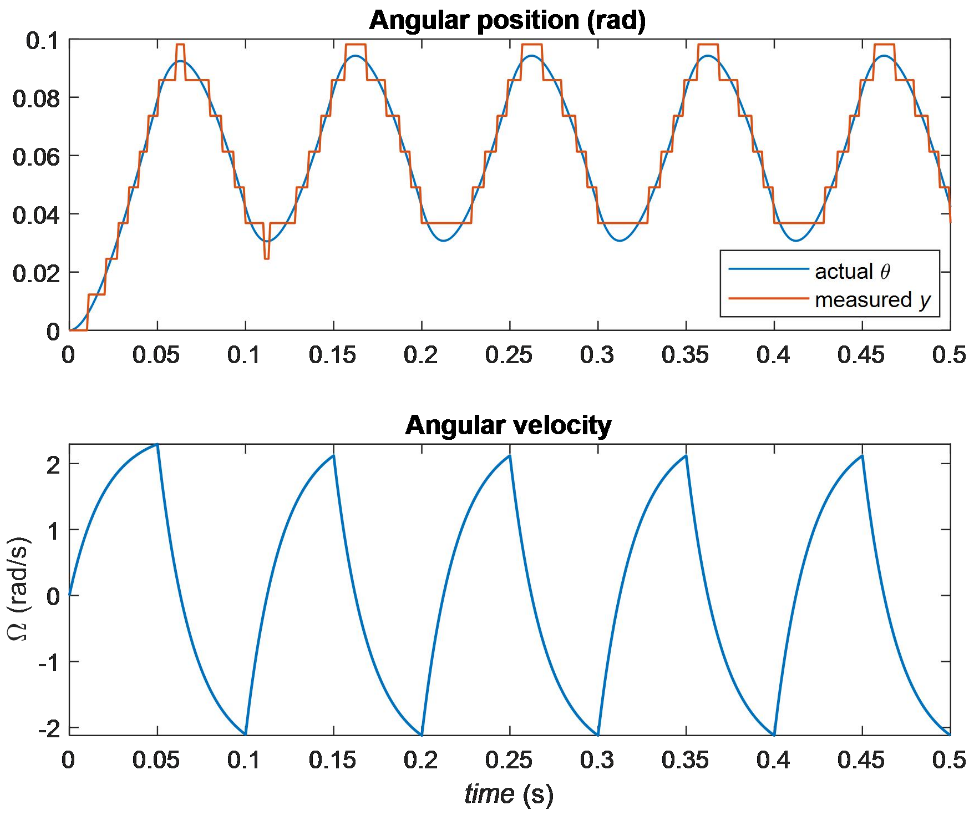

Figure 2 System simulation result and the measure provided by incremental encoder

To begin with, the model can be represented by equations listed below.

$$Y[n] = H_n X[n] + h_n + W[n]$$

$$X[n+1] = F_n X[n] + f_n + V[n]$$

The variance of the quantization noise of the incremental encoder *r* can be noted as:

$$r = (2 * \mathrm{pi/L})\text{^}2/12$$

The prior information $\widehat{X}_{1/0}$ is initiated as 0 and the variance of $P_{1/0}$ is defined as:

$$\begin{bmatrix} Var\_teta & 0 \\ 0 & 0 \end{bmatrix}$$

With the prior information and modelling noise defined, the Kalman filter and Stationary Kalman filter can be constructed. In the next session, those two built filters are simulated under different initial conditions of $\widehat{\theta}_{1/0}$.

3.4 Simulations

In this section, the previous built Kalman filter and Stationary Kalman filter is simulated under different conditions. The conditions vary from the model perfection to the initialization of $\widehat{\theta}_{1/0}$.

3.4.1 Use perfect model with $\widehat{\boldsymbol{\theta}}_{1/0} = \boldsymbol{\theta}_1$

In this part, it is assumed that the model of the system is perfect. Thus, $G_{actual} =$

$G_{filter} = 50\,rad \cdot s^{-1} \cdot V^{-1}$. And $T_{actual} = T_{filter} = 20ms$. The results are in figure 3 and figure 4 below.

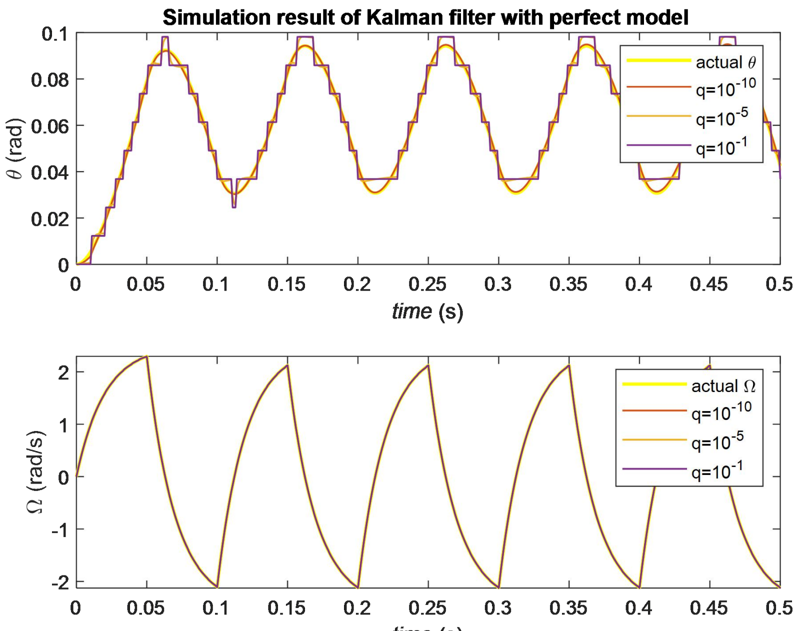

Figure 3 Simulation result of Kalman filter with perfect model

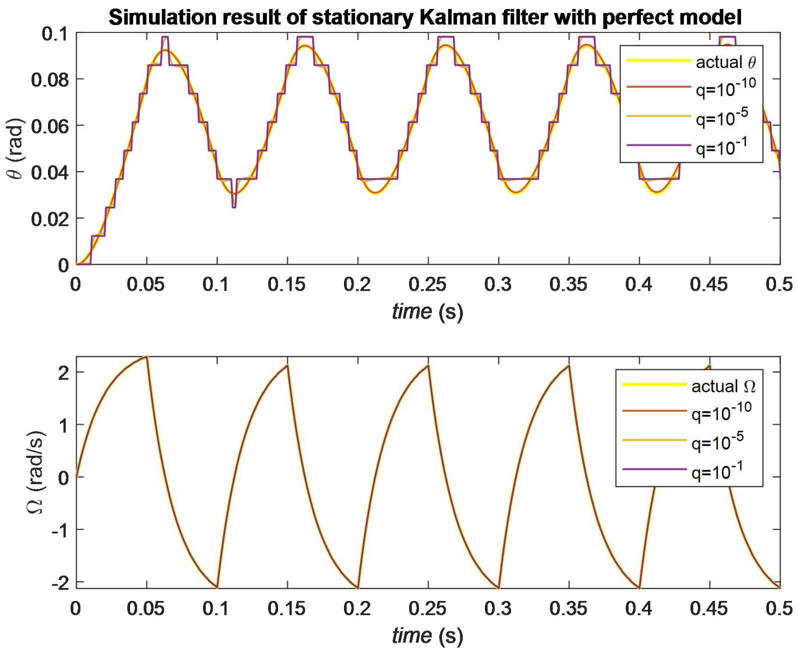

Figure 4 Simulation result of Stationary Kalman filter with perfect model

From these two figures, it can be obtained that using the perfect model, both Kalman filter and Stationary Kalman filter can get an accurate estimation of the two states of the system. However, if the variance q of the white noise in the actual input u is non-negligible, the estimation will approach the measure value y instead of actual value. That means the Kalman filters have a bad behavior.

3.4.2 Use rough model with $\hat{\theta}_{1/0} = \theta_1$

In reality, it is observed that the model of the system is never perfect due to the inaccuracy of the chosen parameters values. Therefore, the value of the T_{filter} is now changed to 25ms to test the effect of applying a rough model compared to a perfect model. The results are shown in figure 5 and figure 6.

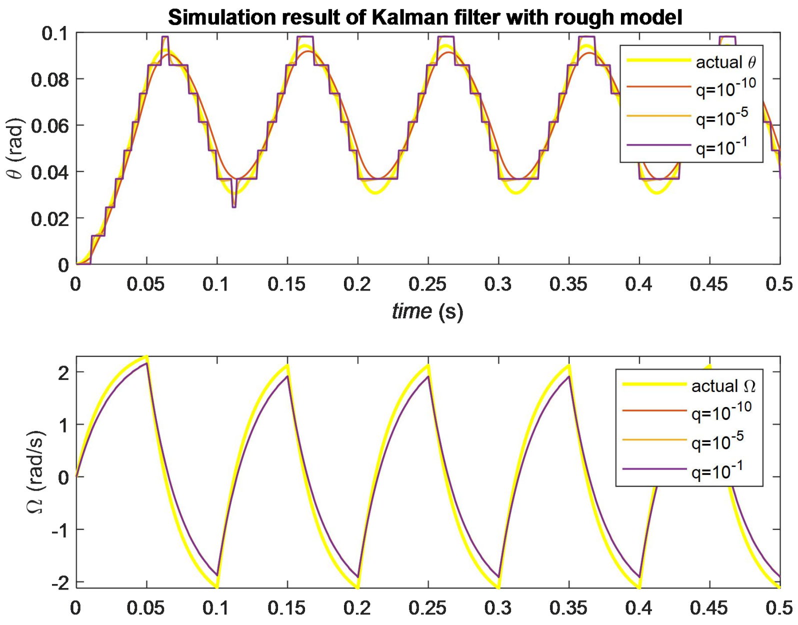

Figure 5 Simulation result of Kalman filter with perfect model

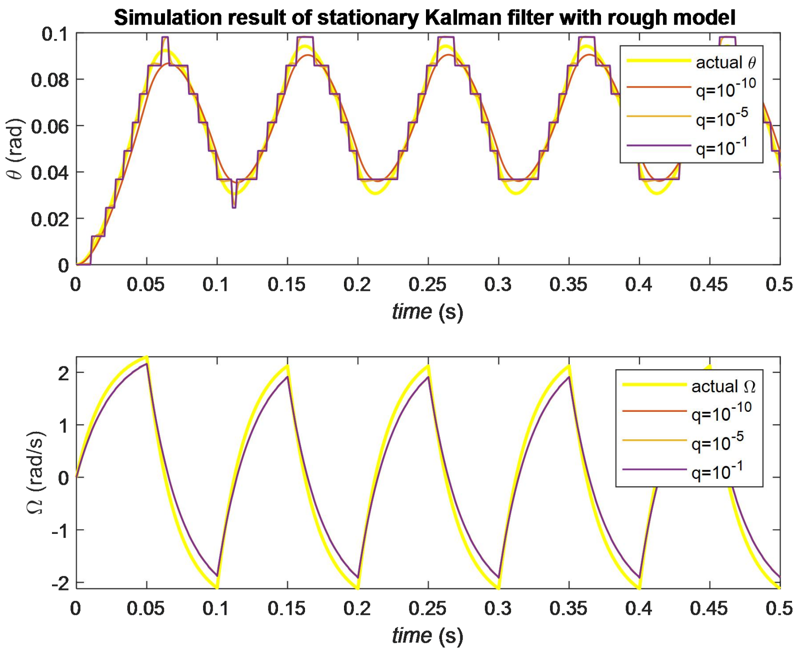

Figure 6 Simulation result of Stationary Kalman filter with perfect model

Compared with figure 3 and figure 4, it can be observed that the accuracy of the estimation of either $\theta(t)$ or $\Omega(t)$ reduces because of the defect in model parameters. In addition, the two Kalman filters have similar behavior and reach a good performance when q is tiny.

3.4.3 $\boldsymbol{\hat{\theta}_{1/0} = \theta_1 \pm 0.05}$

The textbook suggests that the Stationary Kalman filter have a slower transience response after the initialization. To verify this, the initial value of the angular positions is changed to $\theta_1 \pm 0.05$. The system is simulated with $\hat{\theta}_{1/0} = 0.05$ and $\hat{\theta}_{1/0} = -0.05$ respectively. The results are shown in the figures below.

(1) $\hat{\theta}_{1/0} = 0.05$

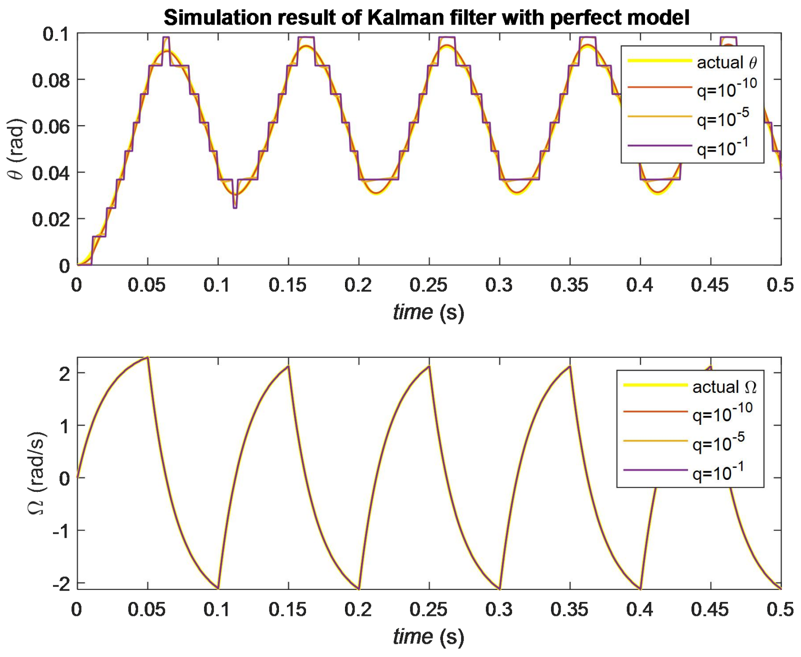

Figure 7 Simulation result of Kalman filter with perfect model

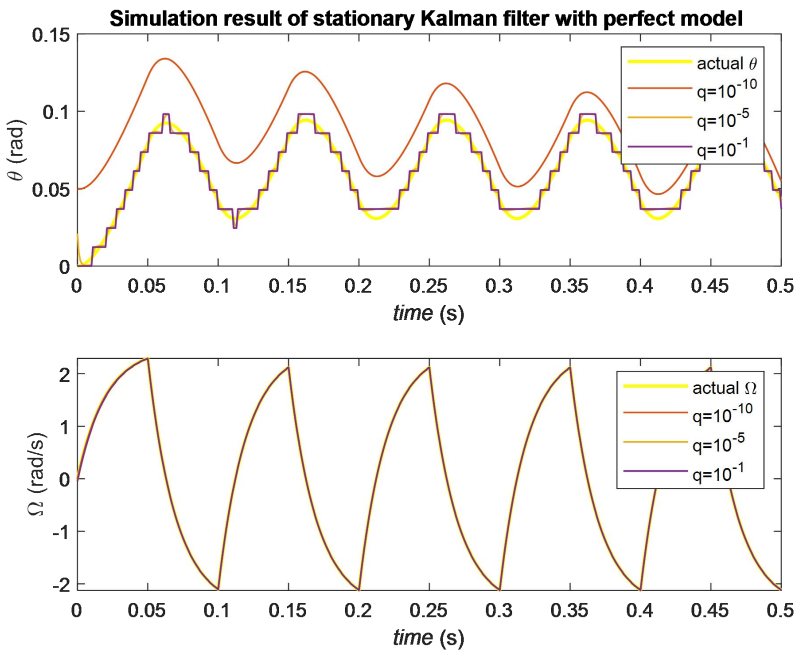

Figure 8 Simulation result of Stationary Kalman filter with perfect model

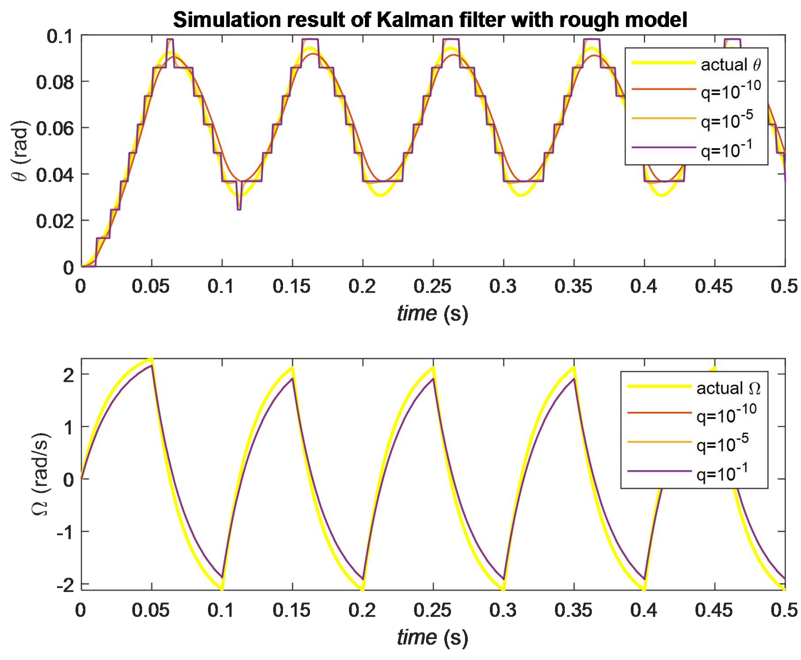

Figure 9 Simulation result of Kalman filter with rough model

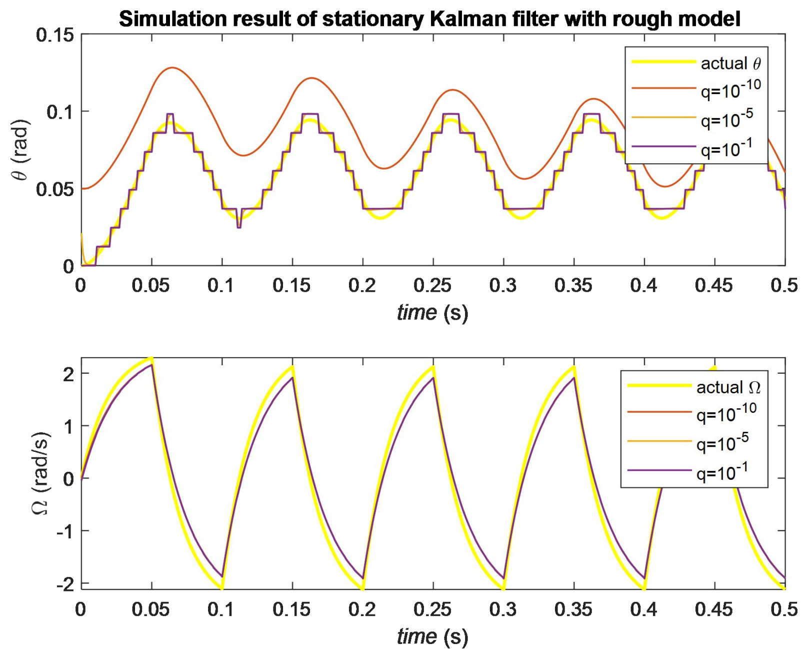

Figure 10 Simulation result of Stationary Kalman filter with rough model

(2) $\hat{\theta}_{1/0} = \ -0.05$

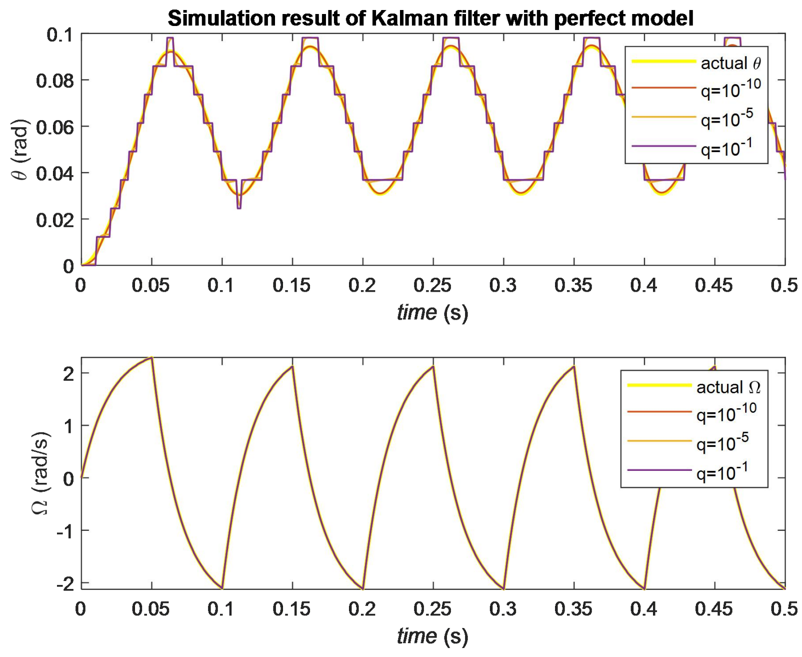

Figure 11 Simulation result of Kalman filter with perfect model

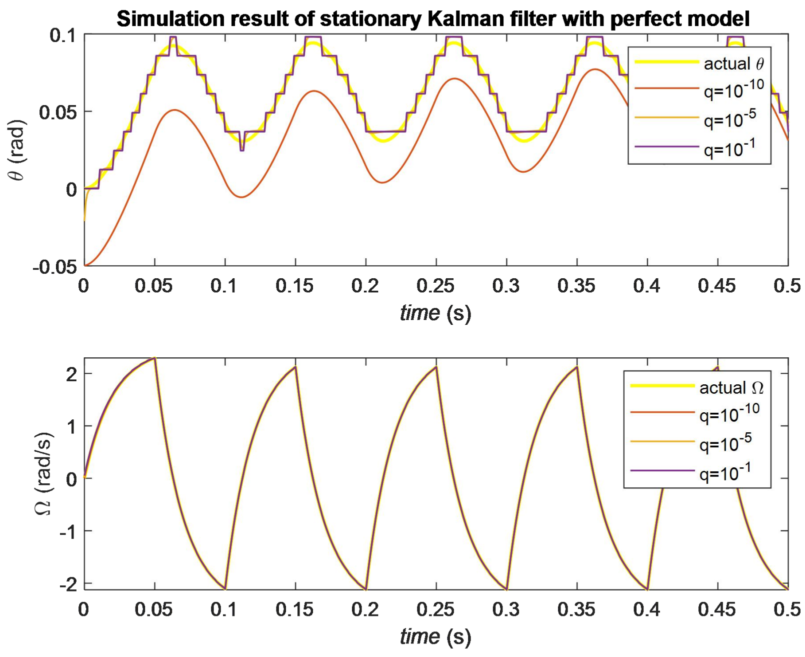

Figure 12 Simulation result of Stationary Kalman filter with perfect model

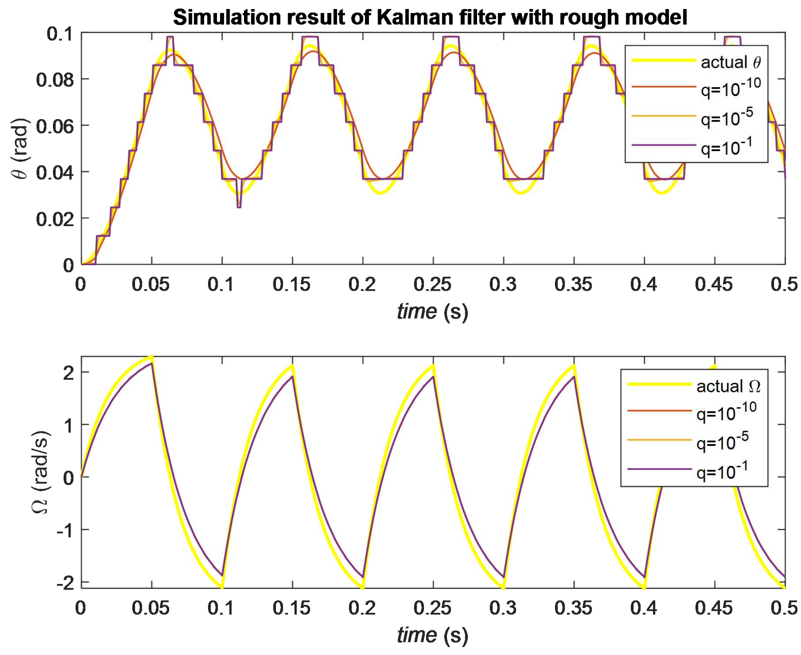

Figure 13 Simulation result of Kalman filter with rough model

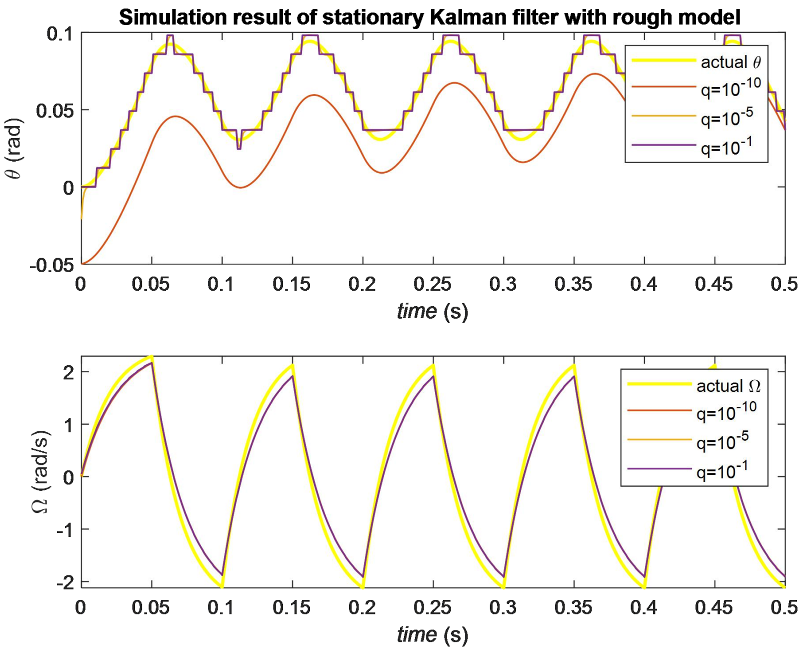

Figure 14 Simulation result of Stationary Kalman filter with rough model

From the figures above, it is clear to see that variation of initial state does not affect the estimation of Kalman filter but has a negative influence on the accuracy of Stationary Kalman filter. It verifies that there is a slower transience in Stationary Kalman filter. In another word, Stationary Kalman filter eliminates the initialization of the variance matrix in recursion. This increases its dependence on the reliability of initial values.

4. Conclusion

(1) The noise introduced in the process of input signal acquisition will affect the estimation of both Kalman filter and Stationary Kalman filter.
(2) Both Kalman filters depend on the accuracy of the system model.
(3) Compared with Kalman filter, Stationary Kalman filter is more affected by false initial values since it simplifies its recursion.

Appendix: Code

A. Matlab Script

```
close all
clear; clc;

%% 2.1 Input voltage
D = 0.5;         % duration
A = 0.1;         % peak-to-peak amplitude
Delta = 100e-3; % period
Ts = 1e-3;       % sample time

u = inputvoltage(D, A, Delta, Ts);

% Plot
t = 0:Ts:D;
figure(1)
plot(t,u,'LineWidth',1);
title('Input voltage');
xlabel('\ittime \rm(s)');
ylabel('\itVoltage \rm(V)');
ylim([-0.06 0.06]);

%% 2.2 System modeling and simulation
G = 50;
T = 20e-3;
L = 512;     % L angles per lap
x1 = [0;0];
[y,x] = simulate(u,G,T,Ts,L,x1);

%% Plot
figure(2)
subplot('211');
plot(t, x(:,1), t, y,'LineWidth',0.8);
title('Angular position (rad)');
legend('actual \theta','measured \ity');
subplot('212')
plot(t, x(:,2),'LineWidth',1);
title('Angular velocity');
xlabel('\ittime \rm(s)');
ylabel('\Omega (rad/s)');

%% 2.3 Kalman filter
r = (2*pi/L)^2/12;
```

```matlab
x1_0 = zeros(2,1);
x1_0(1) = -0.05;  % estimation of theta_0
x1_0(2) = 0;
var_theta = (2*pi)^2/12;
var_Omega= 0;
P1_0 = [var_theta, 0; 0, var_Omega];

%% 2.4 Simulations
T = 20e-3;
xe_1 = kal(y, u, G, T, Ts, L, x1_0, P1_0, 1e-10);
xe_2 = kal(y, u, G, T, Ts, L, x1_0, P1_0, 1e-5);
xe_3 = kal(y, u, G, T, Ts, L, x1_0, P1_0, 1e-1);
t = 0:Ts:D;
figure(1);
subplot(211);
plot(t, x(:,1),'y','LineWidth',1.5); hold on
plot(t, xe_1(:,1),'LineWidth',0.8)
plot(t, xe_2(:,1),'LineWidth',0.8)
plot(t, xe_3(:,1),'LineWidth',0.8)
hold off
legend('actual \theta','q=10^{-10}','q=10^{-5}','q=10^{-1}')
title('Simulation result of Kalman filter with perfect model')
xlabel('\ittime \rm(s)');
ylabel('\theta (rad)')
subplot(212);
plot(t, x(:,2),'y','LineWidth',1.5); hold on
plot(t, xe_1(:,2),'LineWidth',0.8)
plot(t, xe_2(:,2),'LineWidth',0.8)
plot(t, xe_3(:,2),'LineWidth',0.8)
hold off
legend('actual \Omega','q=10^{-10}','q=10^{-5}','q=10^{-1}')
xlabel('\ittime \rm(s)');
ylabel('\Omega (rad/s)');
```

B. inputvoltage function

```matlab
function u = inputvoltage(D,A,Delta,Ts)
t = 0:Ts:D;
u = A/2*square(2*pi*t/Delta);
u = u';
end
```

C. simulate function

```matlab
function [y,x] = simulate(u,G,T,Ts,L,x1)

```

```
%% Initialization
length_u = length(u);
y = zeros(length_u,1);
x = zeros(length_u,2);
x(1,:) = x1;

%% Modeling
A = [0 1; 0 -1/T];
B = [0; G/T];
C = [1 0];
D = 0;

%  Discretization
[Ad,Bd,~,~] = c2dm(A,B,C,D,Ts,'zoh');

for n = 2:length_u
    x_temp = Ad*x(n-1,:)' + Bd*u(n-1);
    x(n,:) = x_temp';
end

% Outputs
teta = x(:,1);
y = round(teta*L/2/pi)*2*pi/L;

end
```

D. kal function

```
function xe = kal(y, u, G, T, Ts, L, x1_0, P1_0, q)
A = [0 1; 0 -1/T];
B = [0; G/T];
C = [1 0];
D = 0;
[Ad,Bd,~,~] = c2dm(A,B,C,D,Ts,'zoh');

xe = zeros(length(u), 2);

H = C;
h = 0;
r = (2*pi/L)^2/12;
F = Ad;

%% Loop n = 1
y_est = H * x1_0 + h;
C_xy = P1_0 * H';
```

```
C_yy = H * P1_0 * H' + r;

x1_1 = x1_0 + C_xy / C_yy * (y(1) - y_est);
xe(1,:) = x1_1';
P1_1 = P1_0 - C_xy / C_yy * C_xy';

f = Bd * u(1);
x_est = F * x1_1 + f;
P = F * P1_1 * F' + q;

%% Loop n > 1
for n = 2 : length(u)
    y_est = H * x_est + h;
    C_xy = P * H';
    C_yy = H * P * H' + r;

    x_estx_est = x_est + C_xy / C_yy * (y(n) - y_est);
    xe(n,:) = x_est';
    PP = P - C_xy / C_yy * C_xy';

    f = Bd * u(n);
    x_est = F * x_est + f;
    P = F * P * F' + q;
end

end
```

E. skal function

```
function xe = skal(y, u, G, T, Ts, L, x1_0, P1_0, q)
A = [0 1; 0 -1/T];
B = [0; G/T];
C = [1 0];
D = 0;
[Ad,Bd,~,~] = c2dm(A,B,C,D,Ts,'zoh');

xe = zeros(length(u), 2);

H = C;
h = 0;
r = (2*pi/L)^2/12;
F = Ad;
% Kalman gain
[K,~,~,~] = dlqe(Ad, [1;1], C, q, r);

```

```
%% Loop n = 1
y_est = H * x1_0 + h;

x1_1 = x1_0 + K * (y(1) - y_est);
xe(1,:) = x1_1';
f = Bd * u(1);
x_est = F * x1_1 + f;

%% Loop n > 1
for n = 2 : length(u)
    y_est = H * x_est + h;

    x_estx_est = x_est + K * (y(n) - y_est);
    xe(n,:) = x_est';
    f = Bd * u(n);
    x_est = F * x_est + f;
end

end
```