ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA ĐIỆN – ĐIỆN TỬ
----------o0o----------

BÁO CÁO TIỂU LUẬN
XỬ LÝ SỐ TÍN HIỆU NÂNG CAO

STOCHASTIC SIGNAL PROCESSING

GVHD: TS. ĐỖ HỒNG TUẤN
HVTH: TIẾN HOÀNG TRÍ NGHĨA – 1870048

TP. HỒ CHÍ MINH, THÁNG 2 NĂM 2019

Contents

1. Problem 1 - Spectral estimation

1.1. Problem

Consider a sum of sinusoids in white Gaussian noise. The sinusoids are with frequencies of αf_s, βf_s and ηf_s, and with amplitudes of *a*, *b* and *c*, respectively. The f_s is sampling frequency. Number of signal samples is *d*, DFT size is *e*. The white Gaussian noise is with variance of 1.5 and zero mean.

a. Plot the estimated spectrum of sum of sinusoids in white Gaussian noise using *f*. Use the values given in Table 1.
b. Evaluate the accuracy of frequency estimation for different values of noise variance (using Monte Carlo simulations).

Table 1

Group	α	β	η	*a*	*b*	*c*	*d*	*e*	*f*
14	**0.20**	**0.25**	**0.4**	**2**	**5**	**2**	**360**	**1024**	**M2, M3**

M2: Modified periodogram (Hamming window)

M3: Modified periodogram (Hamming window)

1.2. Solution A

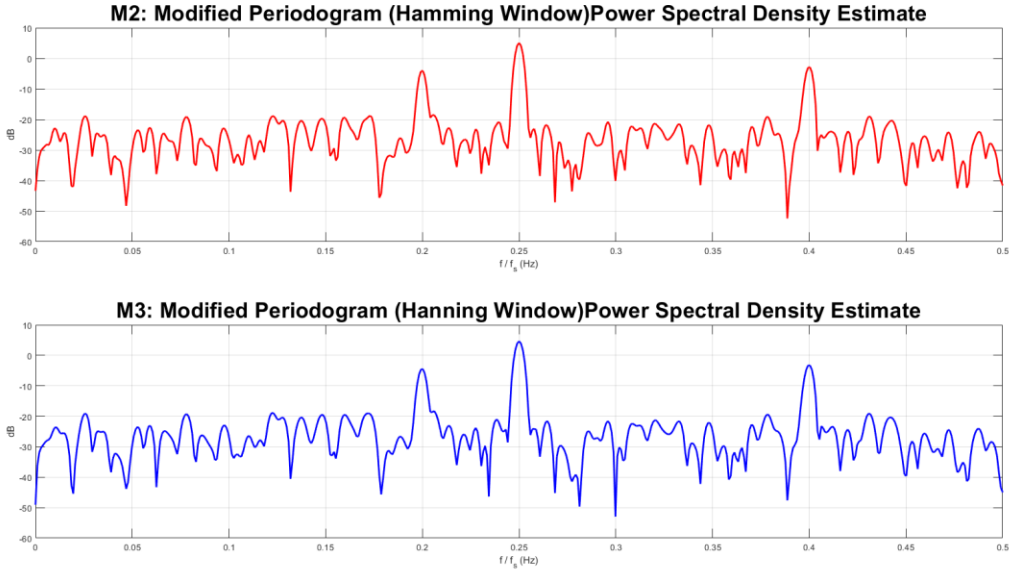
Plot the estimated spectrum of sum of sinusoids in white Gaussian noise using *f*. Use the values given in Table 1.

1.2.1. Matlab Code

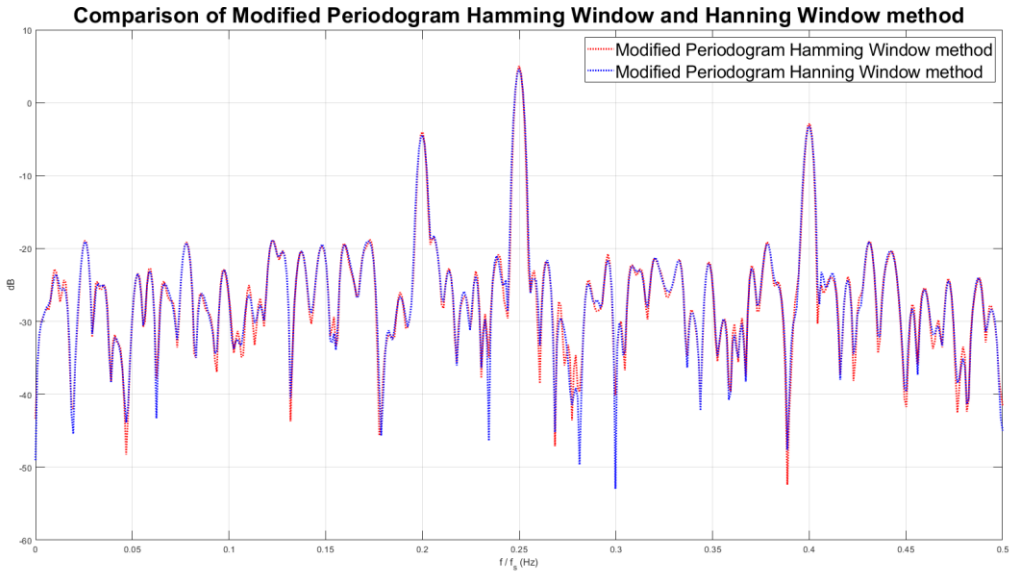
```
clc;
d = 360;                          % Number of signal samples
fs = 1000;                        % Sampling frequency (Hz)
t = (0:1:d-1)/fs;
nfft = 1024;                      % DFT size
A=[2 5 2];                        % Amplitude Matrix
F=fs*[0.20; 0.25; 0.40];          % Frequency Matrix
varriance_noise = 1.5;            % Variance of Gaussian noise

% Sum of sinusoids in white Gaussian noise formula
xn = A*sin(2*pi*F*t) +
sqrt(varriance_noise)*randn(size(t));

% Modified Periodogram with Hamming Window (M2)
[MP_M2,f_M2] = periodogram(xn,hamming(length(xn)),nfft,fs);
MP_M2_dB = 10*log10(MP_M2);
```

```
% Modified Periodogram with Hanning Window (M3)
[MP_M3,f_M3] = periodogram(xn,hanning(length(xn)),nfft,fs);
MP_M3_dB = 10*log10(MP_M3);

% Plot the estimated spectrum of sum of sinusoids in white
Gaussian noise
figure(1)
% Plot the Modified Periodogram with Hamming Window (M2)
subplot(2,1,1),plot(f_M2/fs,MP_M2_dB, 'r' , 'LineWidth',2 ,
'MarkerFaceColor','k')
xlabel('f / f_s (Hz)'); ylabel('dB');
title('\fontsize{25}M2: Modified Periodogram (Hamming
Window)Power Spectral Density Estimate')
grid on
hold on;
%Plot the Modified Periodogram with Hanning Window (M3)
subplot(2,1,2),plot(f_M3/fs,MP_M3_dB, 'b' , 'LineWidth',2 ,
'MarkerFaceColor','k')
xlabel('f / f_s (Hz)'); ylabel('dB');
title('\fontsize{25}M3: Modified Periodogram (Hanning
Window)Power Spectral Density Estimate')
grid on
hold on;

% Comparision of Modified Periodogram(Hamming Window) and
Modified Periodogram(Hanning Window) method
figure(2)
plot(f_M2/fs,MP_M2_dB, 'r:' , 'LineWidth',2 ,
'MarkerFaceColor','k')
hold on;
plot(f_M3/fs,MP_M3_dB, 'b:' , 'LineWidth',2 ,
'MarkerFaceColor','k')
hold on;
legend(' \fontsize{20}
Modified Periodogram Hamming Window method',' \fontsize{20}
Modified Periodogram Hanning Window method')
xlabel('f / f_s (Hz)'); ylabel('dB');
title('\fontsize{25}Comparison of Modified Periodogram
Hamming Window and Hanning Window method')
grid on
```

1.2.2. Results



Hình 1 - The estimated spectrum of sum of sinusoids in white Gaussian noise



Hình 2 – Comparison of modified periodogram Hamming Window and Hanning Window method

1.2.3. Comment the results

- Phương pháp modified Periodogram – Hamming window (M2) có độ rộng búp chính tương đối hẹp và các búp phụ đầu tiên có độ suy giảm công suất tốt hơn, giúp giảm nhiễu và cho phổ tần số chính xác hơn về phổ tần số của tín hiệu ban đầu.
- Phương pháp modified Periodogram – Hanning window (M3) có độ rộng búp chính rộng hơn phương pháp Hamming window nhưng các búp phụ ở xa búp chính có độ suy giảm công suất tốt hơn. Phương pháp Hanning window còn có ưu điểm là răng cưa rất thấp được đánh đổi bằng sự giảm độ phân giải làm độ rộng búp chính bị mở rộng.

1.3. Solution B

Evaluate the accuracy of frequency estimation for difference values of noise variance (using Monte Carlo simulations).

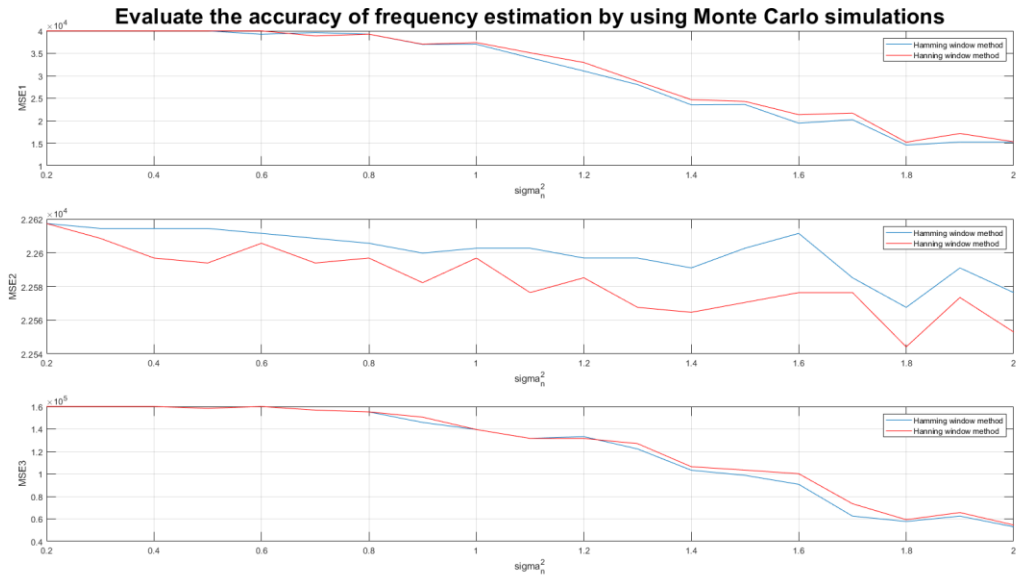
1.3.1. Matlab Code

```
clc;
d = 360;                                % Number of signal
samples
fs = 1000;                              % Sampling frequency
(Hz)
t = (0:1:d-1)/fs;
nfft = 1024;                            % DFT size
A = [2 5 2];                            % Amplitude matrix
F = fs*[0.20; 0.25 ;0.40];              % Frequency matrix

varriance_noise = 0.2:0.1:2;            % Variance of white
noise
K = size(varriance_noise);
N = 100;                                % Number of estimation
Err_hamming_periodogram = zeros(N,3);
Err_hanning_periodogram = zeros(N,3);
MSE_hamming_periodogram = zeros(K(2), 3);
MSE_hanning_periodogram = zeros(K(2), 3);
for i = 1:K(2)
    for k = 1:N
        xn = A*sin(2*pi*F*t) +
sqrt(varriance_noise(i))*randn(size(t));
    % Hamming Periodogram
        [M2Pxx,f] =
periodogram(xn,hamming(length(xn)),nfft,fs);
```

```
        [M2Pxx_dB] = 10*log10(M2Pxx);
    % Find max
        M2Pxx_dB_max = [ -20 -20 -20];
        f_hamming_estimate = [0 0 0];
        for j = 1:size(f)
            if ((f(j) < 170) && (f(j) > 130))
                if (M2Pxx_dB(j) > M2Pxx_dB_max(1))
                    M2Pxx_dB_max(1) = M2Pxx_dB(j);
                    f_hamming_estimate(1) = f(j);
                end
            end
            if ((f(j) < 420) && (f(j) > 380))
                if (M2Pxx_dB(j) > M2Pxx_dB_max(2))
                    M2Pxx_dB_max(2) = M2Pxx_dB(j);
                    f_hamming_estimate(2) = f(j);
                end
            end
            if ((f(j) < 370) && (f(j) > 330))
                if(M2Pxx_dB(j) > M2Pxx_dB_max(3))
                    M2Pxx_dB_max(3) = M2Pxx_dB(j);
                    f_hamming_estimate(3) = f(j);
                end
            end
        end
 % Calculate square error
 Err_hamming_periodogram(k,:) = (f_hamming_estimate -
F').^2;
 % Hanning Periodogram
 [M3Pxx,f] = periodogram(xn,hanning(length(xn)),nfft,fs);
 M3Pxx_dB = 10*log10(M3Pxx);
 %%% Find max
 M3Pxx_dB_max = [ -20 -20 -20];
 f_hanning_estimate = [0 0 0];
        for j = 1:size(f)
            if ((f(j) < 170) && (f(j) > 130))
                if(M3Pxx_dB(j) > M3Pxx_dB_max(1))
                    M3Pxx_dB_max(1) = M3Pxx_dB(j);
                    f_hanning_estimate(1) = f(j);
                end
            end
            if ((f(j) < 420) && (f(j) > 380))
                if(M3Pxx_dB(j) > M3Pxx_dB_max(2))
                   M3Pxx_dB_max(2) = M3Pxx_dB(j);
                   f_hanning_estimate(2) = f(j);
                end
            end
            if ((f(j) < 370) && (f(j) > 330))
```

```
                if(M3Pxx_dB(j) > M3Pxx_dB_max(3))
                   M3Pxx_dB_max(3) = M3Pxx_dB(j);
                   f_hanning_estimate(3) = f(j);
                end
            end
        end
% Calculate square error
Err_hanning_periodogram(k,:) = (f_hanning_estimate -
F').^2;
end
MSE_hamming_periodogram(i,:) =
mean(Err_hamming_periodogram);
MSE_hanning_periodogram(i,:) =
mean(Err_hanning_periodogram);
end
% PLot MSE of each frequency for both methods according to
varriance_noise
figure(1)
subplot(3,1,1),
plot(varriance_noise,MSE_hamming_periodogram(:,1),varriance
_noise,MSE_hanning_periodogram(:,1), 'r' ),
grid on,
xlabel('sigma_n^2'), ylabel('MSE1')
legend('Hamming window method','Hanning window method');
title('\fontsize{25}Evaluate the accuracy of frequency
estimation by using Monte Carlo simulations' )
subplot(3,1,2),
plot(varriance_noise,MSE_hamming_periodogram(:,2),varriance
_noise,MSE_hanning_periodogram(:,2), 'r' ),
grid on,
xlabel('sigma_n^2'), ylabel('MSE2'),
legend('Hamming window method','Hanning window method');
subplot(3,1,3),
plot(varriance_noise,MSE_hamming_periodogram(:,3),varriance
_noise,MSE_hanning_periodogram(:,3), 'r' ),
grid on,
xlabel('sigma_n^2'),ylabel('MSE3')
legend('Hamming window method','Hanning window method');
```

1.3.2. Results



Hình 3 – Evaluate the accuracy of frequency estimation by using Monte Carlo simulations

1.3.3. Comment the results

- Khi phương sai của nhiễu tăng lên thì sai số của các giá trị tần số ước tính so với giá trị tần số thật sẽ giảm xuống.

- Sai số của phương pháp Hamming Window trong trường hợp ước tính sóng sin có công suất nhỏ hơn thì tốt hơn phương pháp Hanning Window

2. Problem 2 - Kalman Filter

2.1. Problem

Formulate the state-space equations for an object moving with a constant velocity in **3-D coordinates** and apply the Kalman filter to track its trajectory. Assumed that the acceleration of the object is white Gaussian noise process with the process noise variance of σ_1^2, and the positioning error is white Gaussian noise process with the measurement noise variance of σ_2^2. The initial state vector of the object is assumed as vector **x**, other initial parameters can be selected arbitrarily.

a. Investigate the performance of the Kalman filter for this problem (using Matlab).
b. Evaluate the error between the measured position and the true position, compared with the error between the Kalman-estimated position and the true position. Give comments based on the evaluation of the errors. Use the values given in Table 2.

Table 2

Group	σ_1^2	σ_2^2	**x**
14	**0.100**	**120**	**[40, 10, 20, 1, 1, 0]**

2.2. Solution A

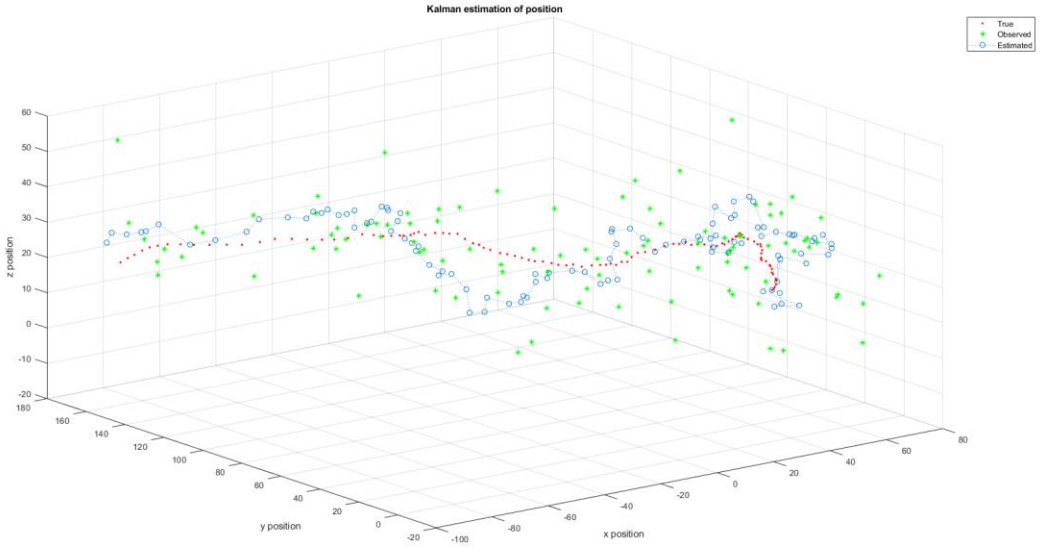
Investigate the performance of the Kalman filter for this problem (using Matlab).

2.2.1. Matlab Code

```
clc;
clear all;
close all;
 F = [1 0 0 1 0 0
      0 1 0 0 1 0
      0 0 1 0 0 1                    % Transition matrix
      0 0 0 1 0 0
      0 0 0 0 1 0
      0 0 0 0 0 1];
 C = [1 0 0 0 0 0
      0 1 0 0 0 0                  % Measurement matrix
      0 0 1 0 0 0];
sigma_v1 = 0.1;                     % Process noise variance
```

```
Q1 = sigma_v1*eye(6);              % Process noise correlation
matrix
sigma_v2 = 120;                    % Measurement noise variance
Q2 = sigma_v2*eye(3);              % Measurement noise
correlation matrix
N=100;                             % State
 %Let's generate some observations.
x(:,1)=[40 10 20 1 1 0]';
y = zeros(3,N);

for i=1:N
    x(:,i+1) = F * x(:,i) + sqrt(sigma_v1)*randn(6,1);
    y(:,i) = C*x(:,i) + sqrt(sigma_v2)*randn(3,1);
end
 % Now let's use the Kalman filter to track the state.
 % Initial estimate of the state
xhat(:,1)=[40 10 20 1 1 0]';
 % Initial estimate of the filtered error correlation matrix
Kn=10*eye(6);
 % Request of thread: The other initial parameters can be
selected arbitrarily
 % Onto Kalman filtering
 for i = 1:N
    G = F * Kn * C' * inv(C*Kn*C' + Q2);
    alpha(:,i) = y(:,i) - C*xhat(:,i);
    xhat(:,i+1) = F*xhat(:,i) + G*alpha(:,i);
    K = Kn - F * G * C * Kn;
    Kn = F * K * F' + Q1;
end;
plot3(x(1,:),x(2,:),x(3,:),'r.');hold on;
plot3(y(1,:),y(2,:),y(3,:),'g*'); hold on;
plot3(xhat(1,:),xhat(2,:),xhat(3,:),':o');hold on;
xlabel('x position');ylabel('y position');zlabel('z
position');title('Kalman estimation of position');
legend('True','Observed','Estimated');
grid on
```

2.2.2. Results



Hình 4 – Kalman estimation of position

2.2.3. Comment the results

- Nhìn vào đồ thị ta có thể thấy, các giá trị ước tính bằng bộ lọc Kalman bám sát các giá trị thật hơn là các giá trị quan sát được.
- Từ kết quả trên, bộ lọc Kalman đã chứng minh là tối ưu theo 2 nghĩa:

  + Giá trị trung bình của trạng thái ước lượng bằng với giá trị trung bình của trạng thái thực: kỳ vọng của trạng thái ước lượng bằng với kỳ vọng của trạng thái thực, theo ý nghĩa xác suất.

  + Độ lệch của trạng thái ước lượng so với trạng thái thực là nhỏ nhất: Phương sai của độ lệch giữa trạng thái thực và trạng thái ước lượng là nhỏ nhất.

2.3. Solution B

Evaluate the error between the measured position and the true position, compared with the error between the Kalman-estimated position and the true position. Give comments based on the evaluation of the errors. Use the values given in Table 2.

2.3.1. Matlab Code

```
clc;
clear all;
close all;
F = [1 0 0 1 0 0
     0 1 0 0 1 0
     0 0 1 0 0 1                % Transition matrix
     0 0 0 1 0 0
     0 0 0 0 1 0
     0 0 0 0 0 1];
C = [1 0 0 0 0 0
     0 1 0 0 0 0                % Measurement matrix
     0 0 1 0 0 0];

N = 100;                         % State
NN = 100;                        % Number of estimate

% Consider the change of sigma_v1
sigma_v1 = 0.02:0.02:0.20;       %process noise variance
sigma_v2 = 120;                  %measurement noise variance
KK = size(sigma_v1,2);
for l=1:KK
    for o=1:NN
        Q2 = sigma_v2*eye(3);    %measurement noise
correlation matrix
        Q1 = sigma_v1(l)*eye(6); %process noise correlation
matrix
% Let's generate some observations.
        x(:,1)=[40 10 20 1 1 0]';
        y = zeros(3,N);
            for i=1:N
             x(:,i+1) = F * x(:,i) +
sqrt(sigma_v1(l))*randn(6,1);
             y(:,i) = C*x(:,i) + sqrt(sigma_v2)*randn(3,1);
            end
% Now let's use the Kalman filter to track the state.
% Initial estimate of the state
        xhat(:,1)=[40 10 20 1 1 0]';

% Initial estimate of the filtered error correlation matrix
```

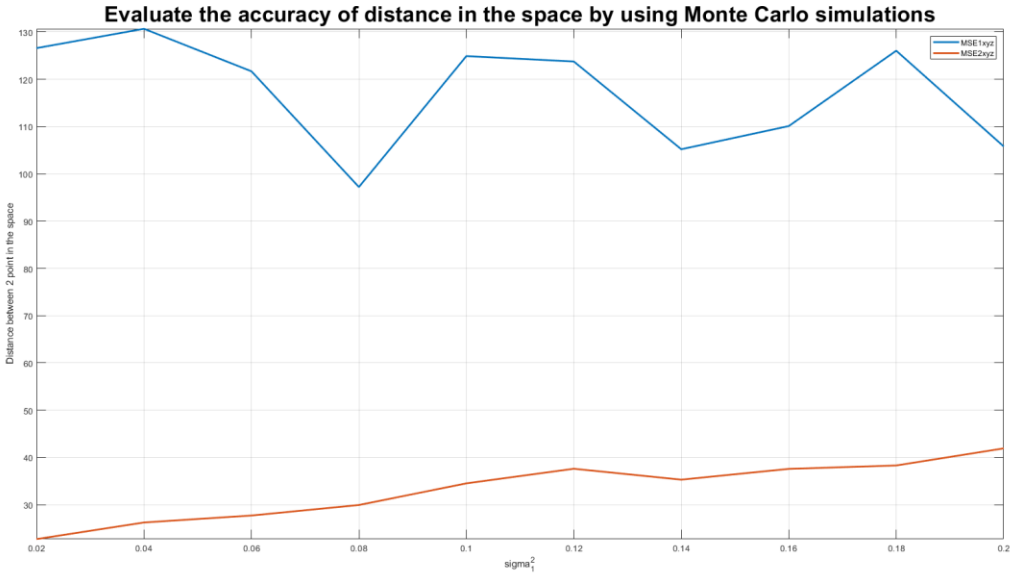
```
        Kn = 10*eye(6);             % Request of thread: The
other initial parameters can be selected arbitrarily
 % Onto Kalman filtering
        for i = 1:N
         G = F * Kn * C' * inv(C*Kn*C' + Q2);
         alpha(:,i) = y(:,i) - C*xhat(:,i);
         xhat(:,i+1) = F*xhat(:,i) + G*alpha(:,i);
         K = Kn - F * G * C * Kn;
         Kn = F * K * F' + Q1;
        end
        Xmod=x(1:3,100);
        Ymod=y(:,100);
        Xhatmod=xhat(1:3,100);
% Calculate Meassurement error acording to x
        Measerrx(:,o) = (Xmod(1,1)-Ymod(1,1)).^2;
% Calculate Meassurement error acording to y
        Measerry(:,o) = (Xmod(2,1)-Ymod(2,1))^2;
% Calculate Meassurement error acording to z
        Measerrz(:,o) = (Xmod(3,1)-Ymod(3,1))^2;
% Calculate estimate error acording to x
        Kalerrx(:,o) = (Xmod(1,1)-Xhatmod(1,1))^2;
% Calculate estimate error acording to y
        Kalerry(:,o) = (Xmod(2,1)-Xhatmod(2,1))^2;
% Calculate estimate error acording to z
        Kalerrz(:,o) = (Xmod(3,1)-Xhatmod(3,1))^2;
% Calculate Meassurement error acording to x,y,z (use formula
of distance between 2 points in the space)
        Measerr3d(:,o) = mean((Xmod-Ymod).^2);
% Calculate estimate error acording to x,y,z (use formula of
distance between 2 points in the space)
        Kalerr3d(:,o) = mean((Xmod-Xhatmod).^2);
    end
    MSE13d(l,:) = mean(Measerr3d,2);
    MSE23d(l,:) = mean(Kalerr3d,2);
    MSE2x(l,:) = mean(Kalerrx,2);
    MSE2y(l,:) = mean(Kalerry,2);
    MSE2z(l,:) = mean(Kalerrz,2);
    MSE1x(l,:) = mean(Measerrx,2);
    MSE1y(l,:) = mean(Measerry,2);
    MSE1z(l,:) = mean(Measerrz,2);
end
% Plot
figure(1)
plot(sigma_v1,MSE13d, sigma_v1, MSE23d, 'LineWidth',2 ,
'MarkerFaceColor','k')
hold on;grid on;
```

```
xlabel('sigma_1^2'), ylabel('Distance between 2 point in the
space')
axis tight;
legend('MSE1xyz','MSE2xyz');
title('\fontsize{25}Evaluate the accuracy of distance in the
space by using Monte Carlo simulations')
figure(2)
hold on,
subplot(3,1,1),plot(sigma_v1,MSE1x,sigma_v1, MSE2x ,
'LineWidth',2 , 'MarkerFaceColor','k');
hold on;
grid on;
legend('MSE1','MSE2');
xlabel('sigma_1^2'), ylabel('x')
axis tight;
title('\fontsize{25}Evaluate the accuracy of distance in each
direction in the space by using Monte Carlo simulations')
subplot(3,1,2),plot(sigma_v1,MSE1y, sigma_v1,MSE2y,
'LineWidth',2 , 'MarkerFaceColor','k');
hold on;
axis tight;
grid on;
xlabel('sigma_1^2'), ylabel('y')
subplot(3,1,3),plot(sigma_v1,MSE1z, sigma_v1, MSE2z ,
'LineWidth',2 , 'MarkerFaceColor','k');
hold on;
grid on;
xlabel('sigma_1^2'), ylabel('z')
axis tight;
%Consider the change of sigma_v2
sigma_v1 = 0.1;                      %process noise variance
sigma_v2 = 100:10:190;                %measurement noise
variance
KK=size(sigma_v2,2);
for l=1:KK
    for o=1:NN
        Q2 = sigma_v2(l)*eye(3);     %measurement noise
correlation matrix
        Q1 = sigma_v1*eye(6);        %process noise correlation
matrix
% Let's generate some observations.
        x(:,1)=[40 10 20 1 1 0]';
        y = zeros(3,N);
            for i=1:N
             x(:,i+1) = F * x(:,i) +
sqrt(sigma_v1)*randn(6,1);
             y(:,i) = C*x(:,i) + sqrt(sigma_v2(l))*randn(3,1);
```

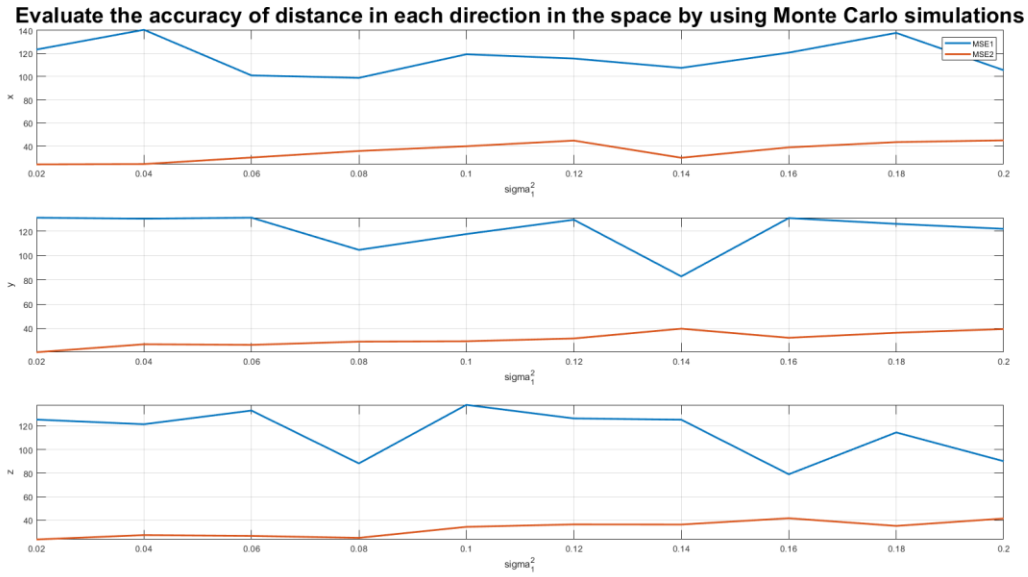
```
            end
% Now let's use the Kalman filter to track the state.
 % Initial estimate of the state
        xhat(:,1)=[40 10 20 1 1 0]';
 % Initial estimate of the filtered error correlation matrix
        Kn=10*eye(6);               % Request of thread: The
other initial parameters can be selected arbitrarily
 % Onto Kalman filtering
        for i = 1:N
         G = F * Kn * C' * inv(C*Kn*C' + Q2);
         alpha(:,i) = y(:,i) - C*xhat(:,i);
         xhat(:,i+1) = F*xhat(:,i) + G*alpha(:,i);
         K = Kn - F * G * C * Kn;
         Kn = F * K * F' + Q1;
        end
        Xmod=x(1:3,50);
        Ymod=y(:,50);
        Xhatmod=xhat(1:3,50);
% Calculate Meassurement error acording to x
        Measerrx(:,o) = (Xmod(1,1)-Ymod(1,1)).^2;
% Calculate Meassurement error acording to y
        Measerry(:,o) = (Xmod(2,1)-Ymod(2,1))^2;
% Calculate Meassurement error acording to z
        Measerrz(:,o) = (Xmod(3,1)-Ymod(3,1))^2;
% Calculate estimate error acording to x
        Kalerrx(:,o) = (Xmod(1,1)-Xhatmod(1,1))^2;
% Calculate estimate error acording to y
        Kalerry(:,o) = (Xmod(2,1)-Xhatmod(2,1))^2;
% Calculate estimate error acording to z
        Kalerrz(:,o) = (Xmod(3,1)-Xhatmod(3,1))^2;
% Calculate Meassurement error acording to x,y,z (use formula
of distance between 2 points in the space)
        Measerr3d(:,o) = mean((Xmod-Ymod).^2);
% Calculate Estimate error acording to x,y,z (use formula of
distance between 2 points in the space)
        Kalerr3d(:,o) = mean((Xmod-Xhatmod).^2);
    end
    MSE13d(l,:) = mean(Measerr3d,2);
    MSE23d(l,:) = mean(Kalerr3d,2);
    MSE2x(l,:) = mean(Kalerrx,2);
    MSE2y(l,:) = mean(Kalerry,2);
    MSE2z(l,:) = mean(Kalerrz,2);
    MSE1x(l,:) = mean(Measerrx,2);
    MSE1y(l,:) = mean(Measerry,2);
    MSE1z(l,:) = mean(Measerrz,2);
end
  %% plot
```

```
 figure(3)
plot(sigma_v2,MSE13d,sigma_v2,MSE23d , 'LineWidth',2 ,
'MarkerFaceColor','k')
hold on;grid on;
legend('MSE1xyz','MSE2xyz');
xlabel('sigma_2^2'), ylabel('Distance between 2 point in the
space')
title('\fontsize{25}Evaluate the accuracy of distance in the
space by using Monte Carlo simulations')
figure(4)
hold on,
subplot(3,1,1),plot(sigma_v2,MSE1x, sigma_v2,MSE2x
,'LineWidth',2 , 'MarkerFaceColor','k');
hold on;
grid on;
xlabel('sigma_2^2'), ylabel('x')
title('\fontsize{25}Evaluate the accuracy of distance in each
direction in the space by using Monte Carlo simulations')
subplot(3,1,2),plot(sigma_v2,MSE1y,sigma_v2,MSE2y,
'LineWidth',2 , 'MarkerFaceColor','k');
hold on;
grid on;
xlabel('sigma_2^2'), ylabel('y')
subplot(3,1,3),plot(sigma_v2,MSE1z,sigma_v2,MSE2z,'LineWidth',
2 , 'MarkerFaceColor','k');
hold on;
grid on;
legend('MSE1','MSE2');
xlabel('sigma_2^2'), ylabel('z')
```

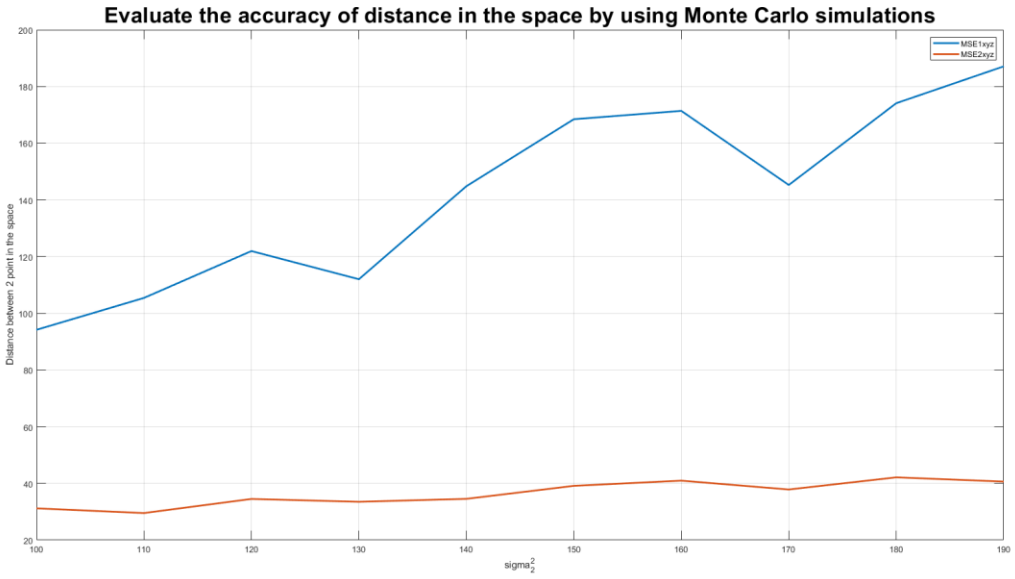
2.3.2. Results



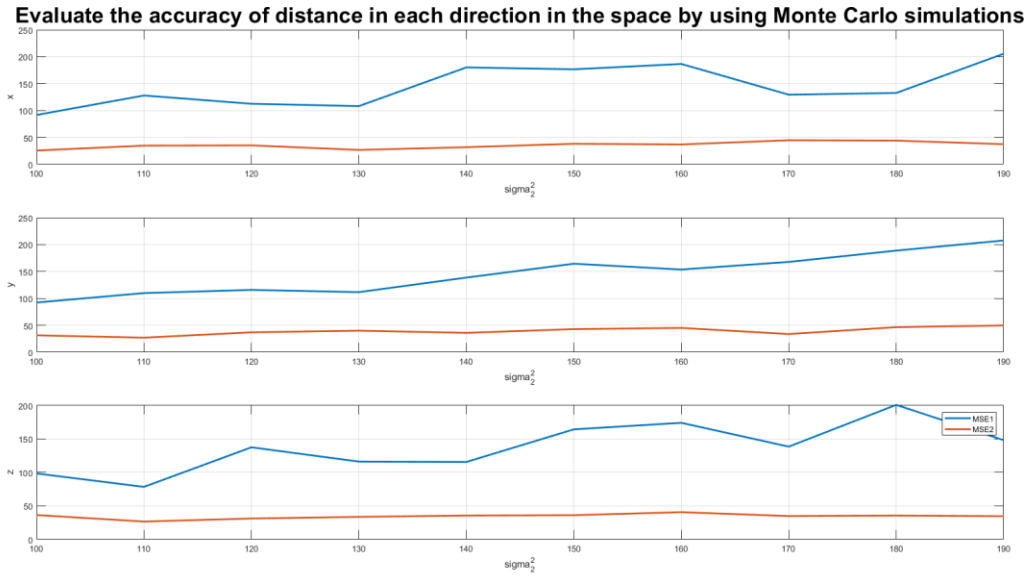
Hình 5 – The result of evaluate the accuracy of distance in the space when changing sigma 1



Hình 6 – The result of evaluate the accuracy of distance in each direction in the space when changing sigma 1



Hình 7 – The result of evaluate the accuracy of distance in the space when changing sigma 2



Hình 8 – The result of evaluate the accuracy of distance in each direction in the space when changing sigma 2

2.3.3. Comment the results

- Qua quan sát kết quả thu được, ta có thể nhận thấy khi phương sai của nhiễu tăng, sai số của các giá trị quan sát và giá trị ước lượng bằng bộ lọc Kalman cũng tăng theo tương ứng.
- Bộ lọc Kalman giúp cải thiện được độ chính xác của hệ thống định vị do dự đoán trạng thái có độ chính xác cao hơn so với giá trị quan sát được từ hệ thống định vị bằng cách ước đoán trạng thái tiên nghiệm và sau đó dựa vào kết quả đo để hiệu chỉnh lại ước đoán.

TÀI LIỆU THAM KHẢO

[1] Mounika S. K. Gudipati, APPLICATION OF KALMAN FILTER TO ESTIMATE POSITION OF A MOBILE NODE IN INDOOR ENVIRONMENTS, IJARCET, 2015.

[2] Sandeep Kaur, Design of FIR filter using hanning window, hamming window and modified hamming window, Prentice Hall, 2nd ed. 1997

[3] Prajoy Podder, Comparative Performance Analysis of Hamming, Hanning and Blackman Window, Final Thesis, International Journal of Computer Applications, 2014