LAMPIRAN

List program

1. **“mono_pulse.m”**

```
function mono_pulse(phi0)
phi0=0.75;
eps = 0.0000001;
angle = -pi:0.01:pi;
y1 = sinc(angle + phi0);
y2 = sinc((angle - phi0));
ysum = y1 + y2;
ydif = -y1 + y2;
figure(1)
plot(angle,y1,'k',angle,y2,'k');
grid;
xlabel('Angle - radians')
ylabel('Squinted patterns')
figure(2)
plot(angle,ysum,'k');
grid;
xlabel('Angle - radians')
ylabel('Sum pattern')
figure(3)
plot(angle,ydif,'k');
grid;
xlabel('Angle - radians')
ylabel('Difference pattern')
angle = -pi/4:0.01:pi/4;
y1 = sinc(angle + phi0);
y2 = sinc((angle - phi0));
ydif = -y1 + y2;
ysum = y1 + y2;
dovrs = ydif ./ ysum;
figure(4)
plot(angle,dovrs,'k');
grid;
xlabel('Angle - radians')
ylabel('voltage gain')
```

2. **“maketraj.m”**

```
function [times , trajectory] = maketraj(start_loc, xvelocity,
yamp, yperiod, zamp, zperiod, samplingtime, deltat)

% CARA MENGGUNAKAN:  [times , trajectory] = maketraj(start_loc,
xvelocity, yamp, yperiod, zamp, zperiod, samplingtime, deltat)
%
% CATATAN: keterangan dari semua koordinat pada radar.
%
```

```
% INPUT
%
% nama          dimensi   keterangan
satuan
%------        ------    ---------------                          ---
----
% start_loc     3 X 1    lokasi awal target                       m
% xvelocity     1        kecepatan target                         m/s
% yamp          1        amplituda dari osilasi arah y            m
% yperiod       1        perioda dari osilasi arah y              m
% zamp          1        amplituda dari osilasi arah z            m
% zperiod       1        peroida dari osilasi arah z              m
% samplingtime  1        waktu jeda lintasan                      sec
% deltat        1        waktu antara sample                      sec
%
% OUTPUTS
%
% nama          dimensi                 keterangan
satuan
%------        ----------              ---------------         -----
-
% times        1 X samplingtime/deltat vektor dari kores-
%                                      ponding ke sample        sec
% trajectory   3 X samplingtime/deltat lintasan x,y,z           m
%
```

```
      times = 0: deltat: samplingtime ;

      x = start_loc(1)+xvelocity.*times ;
      if yperiod~=0
         y = start_loc(2)+yamp*cos(2*pi*(1/yperiod).*times) ;
      else
         y = ones(1, length(times))*start_loc(2) ;
      end
      if zperiod~=0
         z = start_loc(3)+zamp*cos(2*pi*(1/zperiod).*times)  ;
      else
         z = ones(1, length(times))*start_loc(3) ;
      end

      trajectory = [x ; y  ; z] ;
```

3. “addnoise.m”

```
      function [noisytraj ] = addnoise(trajectory, sigmaaz, sigmael,
      sigmarange )
      %
% CARA MENGGUNAKAN: [noisytraj ] = addnoise(trajectory, sigmaaz, sigmael,
sigmarange )
%
```

```
% INPUT
%
% namA          dimensi     keterangan                                satuan
%------         ------      ---------------                           -------
% trajectory    3 X POINTS  lintasan pada koordinat radar             [m;m;m]
% sigmaaz       1           standar deviasi dari eror azimuthr        radians
% sigmael       1           standar deviasi dari eror elevasi         radians
% sigmarange    1           standar deviatsi dari eror jarak           m
%
% OUTPUTS
%
% nama          dimensi      keterangan                               satuan
%------         ------       ---------------                          -------
% noisytraj     3 X POINTS   lintasan noise                           [m;m;m]

      noisytraj = zeros(3, size(trajectory,2)) ;

      for loop = 1 : size(trajectory,2)
         x = trajectory(1,loop);
         y = trajectory(2,loop);
         z = trajectory(3,loop);
         azimuth_corrupted =  atan2(y,x) + sigmaaz*randn(1) ;
         elevation_corrupted = atan2(z, sqrt(x^2+y^2)) + sigmael*randn(1)
      ;
         range_corrupted = sqrt(x^2+y^2+z^2)  + sigmarange*randn(1) ;
         x_corrupted =
      range_corrupted*cos(elevation_corrupted)*cos(azimuth_corrupted) ;
         y_corrupted =
      range_corrupted*cos(elevation_corrupted)*sin(azimuth_corrupted) ;
         z_corrupted = range_corrupted*sin(elevation_corrupted) ;
         noisytraj(:,loop) = [x_corrupted ; y_corrupted; z_corrupted ] ;
      end % loop berikutnya
```

4. “kalfit.m”

```
      function [filtered, residuals , covariances, kalmgains] =
      kalfilt(trajectory, x0, P0, phi, R, Q )

% CARA MENGGUNAKAN: [filtered, residuals , covariances, kalmgains] =
kalfilt(trajectory, x0, P0, phi, R, Q )
%
% INPUT
%
% nama          dimensi                      keterangan        satuan
%------         ------                       ---------------  -------
% trajectory    NUMMEASUREMENTS X NUMPOINTS  trajectory in radar
reference coords          [m;m;m]
% x0            NUMSTATES X 1                initial estimate of
state vector              m, m/s
```

```
% P0           NUMSTATES X NUMSTATES        initial estimate of
covariance matrix        m, m/s
% phi          NUMSTATES X NUMSTATES        state transition matrix
-
% R            NUMMEASUREMENTS X NUMMEASUREMENTS   measurement error
covariance matrix   m
% Q            NUMSTATES X NUMSTATES        state error covariance
matrix               m, m/s
%
% OUTPUT
%
% nama         dimensi                      keterangan
satuan
%------        ------                       ---------------
-------
% filtered     NUMSTATES X NUMPOINTS        filtered trajectory
x,y,z pos, vel    [m; m/s; m; m/s; m; m/s]
% residuals    NUMSTATES X NUMPOINTS        residuals of filtering
[m;m;m]
% covariances  NUMSTATES X NUMPOINTS        diagonal of covariance
matrix         [m;m;m]
% kalmgains    (NUMSTATES X NUMMEASUREMENTS)
%                 X NUMPOINTS               Kalman gain matrix
-
%


      NUMSTATES = 6 ;
      NUMMEASUREMENTS = 3 ;
      NUMPOINTS = size(trajectory, 2) ;

      % inisialisasi output matiks
      filtered = zeros(NUMSTATES, NUMPOINTS) ;
      residuals = zeros(NUMSTATES, NUMPOINTS) ;
      covariances = zeros(NUMSTATES, NUMPOINTS) ;
      kalmgains = zeros(NUMSTATES*NUMMEASUREMENTS, NUMPOINTS) ;

      % set matrix relating measurements to states
      H = [1 0 0 0 0 0 ; 0 0 1 0 0 0 ; 0 0 0 0 1 0];

      xhatminus = x0 ;
      Pminus = P0 ;

      for loop = 1: NUMPOINTS


         % menghitung penguatan kalman
         K = Pminus*H'*inv(H*Pminus*H' + R) ;
         kalmgains(:,loop) = reshape(K, NUMSTATES*NUMMEASUREMENTS,
      1) ;

         % update the estimate with the measurement z
         z = trajectory(:,loop) ;
```

```
   xhat = xhatminus + K*(z - H*xhatminus) ;
   filtered(:,loop) = xhat ;
   residuals(:,loop) = xhat - xhatminus ;

   % update the error covariance for the updated estimate
   P = ( eye(NUMSTATES, NUMSTATES) - K*H)*Pminus ;
   covariances(:,loop) = diag(P) ;  % only save diagonal of
covariance matrix

   % project ahead
   xhatminus_next = phi*xhat ;
   Pminus_next = phi*P*phi' + Q ;

   xhatminus = xhatminus_next ;
   Pminus = Pminus_next ;

end % loop berikutnya

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%
```

5. “kalm.m”

```
function kalm(start_loc, velocity, yamp, yperiod, zamp,
zperiod, samplingtime, deltat, sigmaaz, sigmael, sigmarange,
x0, P0, R, Q)



   phi = [1 deltat 0 0 0 0 ;     % transisi matrix keadaan
     0 1 0 0 0 0 ;
     0 0 1 deltat 0 0 ;
     0 0 0 1 0 0 ;
     0 0 0 0 1 deltat ;
     0 0 0 0 0 1];



% membuat lintasan berdasarkan parameter
[times trajectory] = maketraj(start_loc, velocity, yamp,
yperiod, zamp, zperiod, samplingtime, deltat) ;

figure(1)
plot3( trajectory(1,:), trajectory(2,:), trajectory(3,:) )
grid on
title('lintasan yang diinginkan')
xlabel('x  (meter)')
ylabel('y  (meter)')
zlabel('z  (meter)')
```

```
view(22,66)

% menambah niose untuk menggangu lintasan
[noisytraj] = addnoise(trajectory, sigmaaz, sigmael,
sigmarange ) ;

figure(2)
plot3( noisytraj(1,:), noisytraj(2,:), noisytraj(3,:) )
grid on
title('lintasan yang diberi noise')
xlabel('x  (meter)')
ylabel('y  (meter)')
zlabel('z  (meter)')
view(22,66)

figure(3)
plot(times, noisytraj(1,:), times, trajectory(1,:),'r')
title('posisi x')
xlabel('detik')
ylabel('meter')
legend('terganggu','tak terganggu')
grid on

figure(4)
plot(times, noisytraj(2,:), times, trajectory(2,:),'r')
title('posisi y')
xlabel('detik')
ylabel('meter')
legend('terganggu','tak terganggu')
grid on

figure(5)
plot(times, noisytraj(3,:), times, trajectory(3,:),'r')
title('posisi z')
xlabel('detik')
ylabel('meter')
legend('terganggu','tak terganggu')
grid on

% menampilkan pemfilteran kalamn pada lintasan yang diberi
noise
[filtered, residuals , covariances, kalmgains] =
kalfilt(noisytraj, x0, P0, phi, R, Q ) ;

% menampilkan lintasan yang telah difilter dan lintasan yang
memiliki noise
figure(6)
subplot(3,1,1)
plot(times, noisytraj(1,:), times, filtered(1,:),'r' ) ;
grid on
title('posisi x')
legend('lintasan','telah difilter')
xlabel('detik')
```

```
ylabel('meter')
subplot(3,1,2)
plot(times, noisytraj(2,:), times, filtered(3,:),'r' ) ;
grid on
title('posisi y')
legend('lintasan','telah difilter')
xlabel('detik')
ylabel('meter')
subplot(3,1,3)
plot(times, noisytraj(3,:), times, filtered(5,:),'r' ) ;
grid on
title('posisi z')
legend('lintasan','telah difilter')
xlabel('detik')
ylabel('meter')


% menampilkan kecepatan yang sudah difilter
figure(7)
subplot(3,1,1)
plot(times, filtered(2,:) ) ;
grid on
title('kecepatan pada sumbu x')
xlabel('detik')
ylabel('meter per detik')
axis([0 times(length(times)) min(filtered(2, 3:length(times)))
max(filtered(2, 3:length(times))) ]);
subplot(3,1,2)
plot(times, filtered(4,:) ) ;
grid on
title('kecepatan pada sumbu y')
xlabel('detik')
ylabel('meter per detik')
axis([0 times(length(times)) min(filtered(4, 3:length(times)))
max(filtered(4, 3:length(times))) ]);
subplot(3,1,3)
plot(times, filtered(6,:) ) ;
grid on
title('kecepatan pada sumbu z')
xlabel('detik')
ylabel('meter per detik')
axis([0 times(length(times)) min(filtered(6, 3:length(times)))
max(filtered(6, 3:length(times))) ]);

% Mencari sudut azimut dan sudut elevasi target
figure(8)
plot(times, 180/pi.*atan(filtered(1,:)./filtered(3,:)) ,'r' )
;
grid on
title('Sudut Azimut')

xlabel('detik')
ylabel('derajat')
```

```
figure(9)
plot( times,
180/pi.*atan(filtered(5,:)./sqrt(filtered(1,:).^2+
filtered(3,:).^2)) ) ;
grid on
title('Sudut Elevasi')

xlabel('detik')
ylabel('derajat')


% menampilkan residu dari posisi
figure(10)
subplot(3,1,1)
plot(times, residuals(1,:) ) ;
grid on
title('residual posisi pada sumbu x')
xlabel('detik')
ylabel('meter')
axis([0 times(length(times)) min(residuals(1,
3:length(times)))  max(residuals(1, 3:length(times))) ]);
subplot(3,1,2)
plot(times, residuals(3,:) ) ;
grid on
title('residual posisi pada sumbu y')
xlabel('detik')
ylabel('meter')
axis([0 times(length(times)) min(residuals(3,
3:length(times)))  max(residuals(3, 3:length(times))) ]);
subplot(3,1,3)
plot(times, residuals(5,:) ) ;
grid on
title('residual posisi pada sumbu z')
xlabel('detik')
ylabel('meter')
axis([0 times(length(times)) min(residuals(5,
3:length(times)))  max(residuals(5, 3:length(times))) ]);

% menampilkan residu dari kecepatan
figure(11)
subplot(3,1,1)
plot(times, residuals(2,:) ) ;
grid on
title('residual kecepatan pada sumbu x')
xlabel('detik')
ylabel('meter per sdetik')
axis([0 times(length(times)) min(residuals(2,
3:length(times)))  max(residuals(2, 3:length(times))) ]);
subplot(3,1,2)
plot(times, residuals(4,:) ) ;
grid on
title('residual kecepatan pada sumbu y')
```

```
xlabel('detik')
ylabel('meter per detik')
axis([0 times(length(times)) min(residuals(4,
3:length(times)))  max(residuals(4, 3:length(times))) ]);
subplot(3,1,3)
plot(times, residuals(6,:) ) ;
grid on
title('residual kecepatan pada sumbu z')
xlabel('detik')
ylabel('meter per detik')
axis([0 times(length(times)) min(residuals(6,
3:length(times)))  max(residuals(6, 3:length(times))) ]);




%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%
```