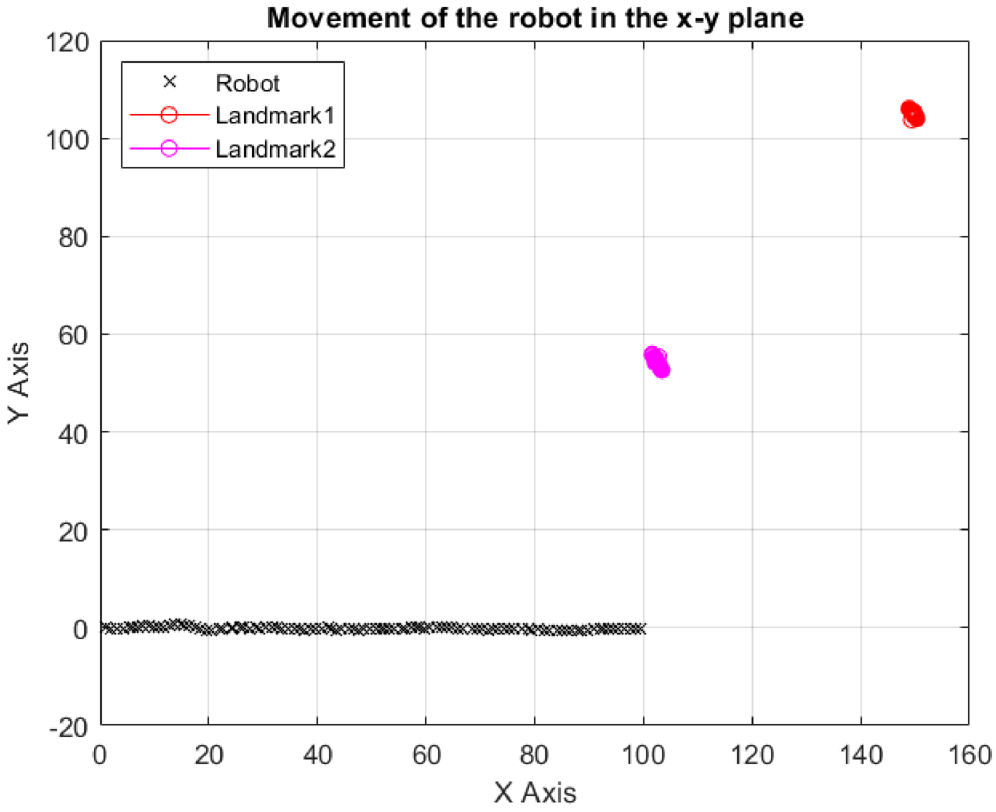
```
%HW3: EKF SLAM
clc;
clear all;
close all;
%% INITIALIZE
mU_tm1tm1=[0,0,0,147,102,98,53]';              % Initial state
sig_tm1tm1=[0 0 0 0 0 0 0;                      % Tuned intial covariance matrix
            0 0 0 0 0 0 0
            0 0 0 0 0 0 0
            0 0 0 10 0 0 0
            0 0 0 0 10 0 0
            0 0 0 0 0 10 0
            0 0 0 0 0 0 10];
% sig_tm1tm1=[zeros(7)];
dT=1;                                          % Sampling time
vt=1;                                          % Linear velocity
wt=0;                                          % Negligible non-zero value
load('s1.mat')
load('s2.mat')
z(:,:,1)= s1;                               % Measurement for landmark1
z(:,:,2)=s2;                                % Measurement for landmark2
R=[0.1  0  0  0  0  0  0;
    0  0.1 0  0  0  0  0;
    0  0  0.1 0  0  0  0;
    0  0  0   0  0  0  0;
    0  0  0   0  0  0  0;
    0  0  0   0  0  0  0;
    0  0  0   0  0  0  0];
Q=[ 0  0  0   0    0   0   0;
    0  0  0   0    0   0   0;
    0  0  0   0    0   0   0;
    0  0  0   0.1  0   0   0;
    0  0  0   0   0.01  0   0;
    0  0  0   0    0   0.1 0;
    0  0  0   0    0   0  0.01];
ln=length(z);
r=vt/wt;                                        %for use later
%% EKF SLAM
mU_storage=zeros(7,ln);                         % to store state updates
for i=1:1:ln                                    %Number of of observations
    Fx=[1 0 0 0 0 0 0;
        0 1 0 0 0 0 0;
        0 0 1 0 0 0 0];
    temp=[vt*dT*cos((mU_tm1tm1(3)+0.5*wt*dT));vt*dT*sin((mU_tm1tm1(3)+0.5*wt*dT));wt*dT];
    mU_ttm1=mU_tm1tm1+(Fx'*temp);
    Gt=[eye(7) + Fx' *[0 0 (-vt*dT*sin(mU_ttm1(3)+0.5*wt*dT)); 0 0 (vt*dT*cos(mU_ttm1(3)+0.5*wt
    sig_ttm1=Gt*sig_tm1tm1*Gt'+R;                         %sigma t,t-1
    k=1;                                        %index to acess sub-matrices
    for j=1:1:2                                 %there are two landmarks
        del = [mU_ttm1(k+3)-mU_ttm1(1);mU_ttm1(k+4)-mU_ttm1(2)];
        q=del'*del;                                     %For expected observation
        z_ti_hat=[sqrt(q);                              % Expected observation
                  atan2(del(2),del(1)) - mU_ttm1(3)];    % predicted measurement
        low_ht = 1/q*[-(sqrt(q)*del(1)) -(sqrt(q)*del(2)) 0  (sqrt(q)*del(1)) (sqrt(q)*del(2))
                        del(2)          -del(1)          -q     -del(2)           del(1)
        Fxj=[1 0 0 0 0 0 0;
             0 1 0 0 0 0 0;
             0 0 1 0 0 0 0];                               %generating Fxj matrix
        temp=[zeros(2,3) eye(2) zeros(2,2)];
        temp=circshift(temp,[0,2*(j-1)]);
        if j==1
            Fxj = [Fxj;temp;zeros(2,7)];
        else
            Fxj = [Fxj;zeros(2,7);temp];
        end
        Hti= low_ht*Fxj;
        Q_t = [0.1 0;
                0           0.01];
        Kti= sig_ttm1*Hti'*((Hti*sig_ttm1*Hti'+Q_t))^-1;       %Kalman gain
        mU_ttm1= mU_ttm1 + Kti*([z(i,1,j);z(i,2,j)]-z_ti_hat); %Corrected meu value
        sig_ttm1=(eye(7)- Kti*Hti)*sig_ttm1;                   %Corrected sigma value
        k=k+2;
    end
    mU_tm1tm1=mU_ttm1;                          %state update
    mU_storage(:,i)=mU_tm1tm1;
    sig_tm1tm1=sig_ttm1;                       %state covariance update
end
%% PLOT THE RESULT
figure(1);
title('Movement of the robot in the x-y plane');
for i=1:1:ln
    plot(mU_storage(1,i),mU_storage(2,i),'x','color','k')
    title('Movement of the robot in the x-y plane');
    xlim([0 160]);
    ylim([-20 120])
    xlabel('X Axis');
    ylabel('Y Axis');
    hold on
    grid on
    plot(mU_storage(4,i),mU_storage(5,i),'-mo','color','r');
    plot(mU_storage(6,i),mU_storage(7,i),'-mo','color','m');
    pause(0.05);
end
legend('Robot','Landmark1','Landmark2','location','northwest');
```


Movement of the robot in the x-y plane
Robot
Landmark1
Landmark2
Y Axis
X Axis
120
100
80
60
40
20
0
-20
0
20
40
60
80
100
120
140
160