```
u=[1;0;0];
mut=[2;2;0];
covt=zeros(3,3);
velocity=[1 0;1 0;1 0;pi/2 pi/2;pi/2 pi/2;1 0;1 0; 1 0];
signature=[2;3;3;4;5;6;6;6];
pmx=[2];
pmy=[2];
cmx=[];
cmy=[];

hold on
 for i = 1:8
     u(1)=velocity(i,1);
     u(2)=velocity(i,2);
     c=signature(i);
     z_mes=i;
     %plot(mut(1),mut(2))
     %hold on
     scatter(2,2)
     [mutbar,covtbar] = motion_model_ekf(mut,covt,u);
     plot(mutbar(1),mutbar(2))
     hold on
     plot_error(mutbar(1),mutbar(2),covtbar)
     pmx(end+1) = mutbar(1);
     pmy(end+1) = mutbar(2);
     hold on
%      pmx(end+1) = mutbar(1);
%      pmy(end+1) =mutbar(2);
     %pred_cov(end+1)=covtbar;
     [mut,covt] = correction_ekf(mutbar,covtbar,z_mes,c);
     plot(mut(1),mut(2))
     hold on
     plot_error(mut(1),mut(2),covt)
     hold on
     cmx(end+1) = mut(1);
     cmy(end+1) = mut(2);
%      cmx(end+1) = mut(1);
%      cmy(end+1) =mut(2);

     %cor_mean(end+1)=mut;
     %cor_cov(end+1)=covt;
 end
%[0.0;0.0],[4.0;0.0],[8.0;0.0],[8.0;6.0],[4.0;6.0],[0.0;6.0]

plot(pmx,pmy)
hold on
plot(cmx,cmy)
plot_lines(cmx,cmy)
function plots = plot_lines(cmx,cmy)
title("EKF filter localization")
scatter(0,0)
hold on
```

```
scatter(4,0)
hold on
scatter(8,0)
hold on
scatter(8,6)
hold on
scatter(4,6)
hold on
scatter(0,6)
hold on

plot([cmx(1) 4],[cmy(1) 0],'--')
hold on
plot([cmx(2) 8],[cmy(2) 0],'--')
hold on
plot([cmx(3) 8],[cmy(3) 0],'--')
hold on
plot([cmx(4) 8],[cmy(4) 6],'--')
hold on
plot([cmx(5) 4],[cmy(5) 6],'--')
hold on
plot([cmx(6) 0],[cmy(6) 6],'--')
hold on
plot([cmx(7) 0],[cmy(7) 6],'--')
hold on
plot([cmx(8) 0],[cmy(8) 6],'--')



end


 function [mutbar,covtbar] = motion_model_ekf(mut_1,covt_1,u)
 plot(mut_1(1),mut_1(2))
 hold on
%UNTITLED3 Summary of this function goes here
%   Detailed explanation goes here
%u=[4;0;0];
%mut_1=[2;2;0];
alpha=[0.0001; 0.0001; 0.01; 0.0001; 0.0001; 0.0001];
Gt = eye(3);
Vt = zeros(3,2);
mt = zeros(2,2);
%mutbar=zeros(3,1);
%covtbar=zeros(3,3);
%covt_1=zeros(3,3);
dt=1;
v=u(1);
w=u(2);
mt(1,1)=alpha(1)*v^2 +alpha(2)*w^2;
mt(2,2)=alpha(3)*v^2+alpha(4)*w^2;
eps=1e-4;
theta=mut_1(3);
if abs(w)>eps
```

```
    k=v/w;
    Gt(1,3)=-k*cos(theta)+k*cos(theta+w); %not multiplying with dt as
 its 1
    Gt(2,3)=-k*sin(theta)+k*sin(theta+w);
    Vt(1,1)= -(sin(theta) + sin(theta + w*dt))/w;
    Vt(2,1) = (cos(theta) - cos(theta + w*dt))/w;
    Vt(1,2) =  v*(sin(theta) - sin(theta + w*dt))/(w*w) + v*cos(theta
 + w*dt)*dt/w;
    Vt(2,2) = -v*(cos(theta) - cos(theta + w*dt))/(w*w) + v*sin(theta
 + w*dt)*dt/w;
    Vt(3,1)= 0;
    Vt(3,2) = dt;
    mutbar(1)= mut_1(1) - k*sin(theta) + k*sin(theta + w*dt);
    mutbar(2) = mut_1(2) + k*cos(theta) - k*cos(theta + w*dt);
    mutbar(3) = mut_1(3) + w*dt;
    mutbar=mutbar.';
    %predicted_mean.append(mutbar)
    covtbar = Gt*covt_1*Gt.'+Vt*mt*Vt.';
    %predicted_cov.append(covtbar)
else
    Gt(1,3)=-v*sin(theta)*dt; %not multiplying with dt as its 1
    Gt(2,3)=v*cos(theta)*dt;
    Vt(1,1)= cos(theta)*dt;
    Vt(2,1) = sin(theta)*dt;
    Vt(1,2) = -v*sin(theta)*dt*dt*0.5;
    Vt(2,2) = v*cos(theta)*dt*dt*0.5;
    Vt(3,1)= 0;
    Vt(3,2) = dt;
    mutbar(1)= mut_1(1)+v*cos(theta)*dt;
    mutbar(2) = mut_1(2)+v*sin(theta)*dt;
    mutbar(3) = mut_1(3);
    mutbar=mutbar.';
    %predicted_mean.append(mutbar)
    covtbar = Gt*covt_1*Gt.'+Vt*mt*Vt.';
    %predicted_cov.append(covtbar)
end

 end

 function [mut,covt] = correction_ekf(mutbar,covtbar,z_mes,c)
%UNTITLED4 Summary of this function goes here
%   Detailed explanation goes here

mapObj = containers.Map({1,2,3,4,5,6},{[0.0;0.0],[4.0;0.0],[8.0;0.0],
[8.0;6.0],[4.0;6.0],[0.0;6.0]});
measurement=[2.276 5.249 2;4.321 5.834 3;3.418 5.869 3;3.774 5.911
 4;2.631 5.140 5;4.770 5.791 6;3.828 5.742 6;3.153 5.739 6];
sigmar=0.1;
sigmaphi=0.09;
Qt=zeros(2,2);
Ht=zeros(2,3);
%St=zeros(3,2);
I=eye(3);
Qt(1,1)=sigmar*sigmar;
```

```
Qt(2,2)=sigmaphi*sigmaphi;
landmark=mapObj(c);
mx=landmark(1);
my=landmark(2);
q=(mx-mutbar(1))^2 + (my-mutbar(2))^2;

zthat(1)= sqrt(q);
temp_angle=atan2(my-mutbar(2),mx-mutbar(1))-mutbar(3);
zthat(2)=check_angle(temp_angle);
Ht(1,1)=-(mx-mutbar(1))/sqrt(q);
Ht(1,2)=-(my-mutbar(2))/sqrt(q);
Ht(2,1)=(my-mutbar(2))/(q);
Ht(2,2)=-(mx-mutbar(1))/(q);
Ht(2,3)=-1;

St=(Ht*covtbar)*Ht.'+Qt;

inverseS = inv(St);
Kt=(covtbar*Ht.')*inverseS;

z_measurement(1) = measurement(z_mes,1);
z_measurement(2) = measurement(z_mes,2);

mut=mutbar+Kt*((z_measurement-zthat).');
covt=(I-Kt*Ht)*covtbar;

 end

 function angle = check_angle(rad)
  if rad<0
      angle = rad + 2*pi;
  elseif rad >2*pi
       angle = rad- 2*pi;
  else
      angle = rad;
  end
 end

 function [error_plot] = plot_error(x,y,cov_mat)
    cov_mat=cov_mat(1:2,1:2);
    disp(cov_mat)
    disp(x);
    disp(y);
    [eigenvec, eigenval ] = eig(cov_mat);
    max_values =  max(eigenval);
    [argvalue1, argmin] = min(max_values);
    [argvalue2, argmax] = max(max_values);
    largest_eigenvec = eigenvec(:, argmax);
    smallest_eigenvec=eigenvec(:,argmin);
    % Calculate the angle between the x-axis and the largest
 eigenvector
    angle = atan2(largest_eigenvec(2), largest_eigenvec(1));
```

```
    % This angle is between -pi and pi.
    % Let's shift it such that the angle is between 0 and 2pi
    if(angle < 0)
        angle = angle + 2*pi;
    end

    chisquare_val = 2.4477;
    theta_grid = linspace(0,2*pi);
    phi = angle;
    a=chisquare_val*sqrt(argvalue2);
    b=chisquare_val*sqrt(argvalue1);

% the ellipse in x and y coordinates
    ellipse_x_r  = a*cos( theta_grid );
    ellipse_y_r  = b*sin( theta_grid );

    %Define a rotation matrix
    R = [ cos(phi) sin(phi); -sin(phi) cos(phi) ];

    %let's rotate the ellipse to some angle phi
    r_ellipse = [ellipse_x_r;ellipse_y_r]' * R;

    % Draw the error ellipse
    plot(r_ellipse(:,1) + x,r_ellipse(:,2) + y,'-')
    hold on;

    % Plot the original data
    plot(x, y, '.');
    % mindata = min(min(data));
    % maxdata = max(max(data));
    % Xlim([mindata-3, maxdata+3]);
    % Ylim([mindata-3, maxdata+3]);
    hold on;

    % Plot the eigenvectors
    quiver(x,y, largest_eigenvec(1)*sqrt(argvalue2),
 largest_eigenvec(2)*sqrt(argvalue2), '-m', 'LineWidth',4);
    quiver(x,y, smallest_eigenvec(1)*sqrt(argvalue1),
 smallest_eigenvec(2)*sqrt(argvalue1), '-g', 'LineWidth',4);
    hold on;

    % Set the axis labels
    hXLabel = xlabel('x');
    hYLabel = ylabel('y');
   % error_plot=true;
 end

    0.0001         0
         0    0.0025

     3

     2
```

```
1.0e-03 *

 0.0996   -0.0063
-0.0063    0.9285

 2.9996

 1.9850

 0.0002    0.0002
 0.0002    0.0108

 3.9991

 1.9549

 0.0002    0.0001
 0.0001    0.0043

 4.0013

 1.9419

 0.0003    0.0005
 0.0005    0.0164

 5.0008

 1.9112

 0.0003    0.0002
 0.0002    0.0064

 5.0000

 1.8103

 0.0121   -0.0120
-0.0120    0.0209

 6.1071

 2.6902

 0.0023   -0.0017
-0.0017    0.0067

 6.1201

 2.6973

 0.0197    0.0095
 0.0095    0.0159
```

*5.2654*

*3.8240*

| | |
|---|---|
| *0.0061* | *0.0007* |
| *0.0007* | *0.0047* |

*5.1818*

*3.7018*

| | |
|---|---|
| *0.0061* | *0.0021* |
| *0.0021* | *0.0146* |

*4.1819*

*3.6826*

| | |
|---|---|
| *0.0045* | *0.0018* |
| *0.0018* | *0.0071* |

*4.1832*

*3.7003*

| | |
|---|---|
| *0.0046* | *0.0022* |
| *0.0022* | *0.0186* |

*3.1832*

*3.7014*

| | |
|---|---|
| *0.0039* | *0.0022* |
| *0.0022* | *0.0084* |

*3.1734*

*3.7709*

| | |
|---|---|
| *0.0040* | *0.0025* |
| *0.0025* | *0.0185* |

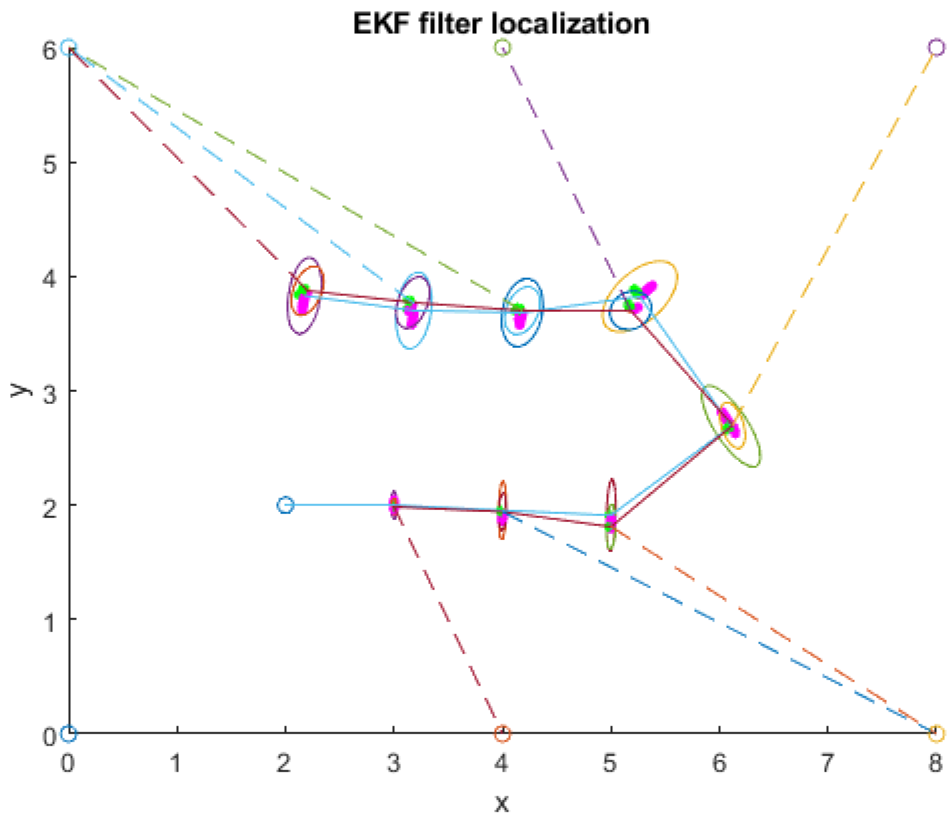
*2.1751*

*3.8299*

| | |
|---|---|
| *0.0037* | *0.0022* |
| *0.0022* | *0.0076* |

*2.2026*

*3.8738*




*Published with MATLAB® R2021a*