```
%Load Files
clear
close
load dataset2.mat

%Variances values
T = 0.1;
%Q_k = [v_var, 0, 0;0, v_var,0;0, 0, om_var];
Q_k = diag([v_var;om_var])*10;

%Initialization of arrays
sizet = size(t);
iterations = 1000;%sizet(1,1);
F_kprev = [];
w_kprev = [];
x_kprev = zeros(iterations,3);
p_kprev = [];
x_kpost = zeros(iterations,3);
p_kpost = [];
r_thresh = 1;
variances = zeros(3,iterations);
cov_matrix = zeros(iterations,4);

for i = 1:iterations

    if i ==1

        % Prev Covariance matrix
        F_kprev = [1, 0 , -T*sin(th_true(1))*v(i); 0,1, T*cos(th_true(1))*v(i); 0,0,1];
        w_kprev = [cos(th_true(1)),0;sin(th_true(1)),0;0,1]*T;
        p_kprev = F_kprev * (diag([1,1,0.1])) * F_kprev' + w_kprev * Q_k * w_kprev';

        % Prev State
        vec1 = [x_true(1); y_true(1); th_true(1)] + T*[cos(th_true(1)),0;sin(th_true(1)),0;0,1] * [v(i);om(i)];
        x_kprev(i,1:3) = vec1';
        x_kprev(i,3) = wrapToPi(x_kprev(i,3));

        a = r(i,:);
        a = a(a~=0);

        a = a(:, ~(any(a > r_thresh,1)));

        if isempty(a) == 1

            x_kpost(i,1:3) = x_kprev(i,1:3);
            p_kpost = p_kprev;
```

```
 47              variances(1:3,i) = diag(p_kpost);
 48              cov_matrix(i,1:2) = p_kpost(1,1:2);
 49              cov_matrix(i,3:4) = p_kpost(2,1:2);
 50
 51
 52          else
 53
 54              %Kalman Gain & correction
 55              [K, G, y_kmeas, g] = kalmanGain2(x_kprev(i,1:3),p_kprev, r(i,:), b
(i,:), r_thresh, l, d,r_var,b_var);
 56              %Posterior state
 57              vec2 = x_kprev(i,1:3)' + K*(y_kmeas - g);
 58              x_kpost(i,1:3) = vec2';
 59              x_kpost(i,3) = wrapToPi(x_kpost(i,3));
 60              %Posterior covariance
 61              p_kpost = (eye(3)-K*G)*p_kprev;
 62              variances(1:3,i) = diag(p_kpost);
 63              cov_matrix(i,1:2) = p_kpost(1,1:2);
 64              cov_matrix(i,3:4) = p_kpost(2,1:2);
 65
 66          end
 67
 68      else
 69          % Prev Covariance matrix
 70          F_kprev = [1, 0 , -T*sin(x_kpost(i-1,3))*v(i); 0,1, T*cos(x_kpost(i-1,3))
*v(i); 0,0,1];
 71          w_kprev = [cos(x_kpost(i-1,3)),0;sin(x_kpost(i-1,3)),0;0,1]*T;
 72          p_kprev = F_kprev * p_kpost * F_kprev' + w_kprev * Q_k * w_kprev';
 73
 74
 75          % Prev State
 76          vec1 = [x_kpost(i-1,1); x_kpost(i-1,2); x_kpost(i-1,3)] + T*[cos(x_kpost
(i-1,3)),0;sin(x_kpost(i-1,3)),0;0,1] * [v(i);om(i)];
 77          x_kprev(i,1:3) = vec1';
 78          x_kprev(i,3) = wrapToPi(x_kprev(i,3));
 79
 80          a = r(i,:);
 81          a = a(a~=0);
 82
 83          a = a(:, ~(any(a > r_thresh,1)));
 84
 85          if isempty(a) == 1
 86
 87              x_kpost(i,1:3) = x_kprev(i,1:3);
 88              p_kpost = p_kprev;
 89              variances(1:3,i) = diag(p_kpost);
 90              cov_matrix(i,1:2) = p_kpost(1,1:2);
 91              cov_matrix(i,3:4) = p_kpost(2,1:2);
 92
```

```
 93         else
 94
 95             %Kalman Gain & correction
 96             [K, G, y_kmeas, g] = kalmanGain2(x_kprev(i,1:3),p_kprev, r(i,:), b ↙
(i,:), r_thresh, l, d,r_var,b_var);
 97             %Posterior state
 98             vec2 = x_kprev(i,1:3)' + K*(y_kmeas - g);
 99             x_kpost(i,1:3) = vec2';
100             x_kpost(i,3) = wrapToPi(x_kpost(i,3));
101             %Posterior covariance
102             p_kpost = (eye(3)-K*G)*p_kprev;
103             variances(1:3,i) = diag(p_kpost);
104             cov_matrix(i,1:2) = p_kpost(1,1:2);
105             cov_matrix(i,3:4) = p_kpost(2,1:2);
106
107         end
108
109     end
110 end
```