**Appendix D – Codes accompanying Question 4**

|  |
| --- |
| function [x\_est,y\_est,dx,P,S,e\_x,e\_y] = LKF(dx\_init,P\_init,x\_nom,y\_nom,x,y,Fk,Hk,Ok,Q,R, wrapOn)    n = size(Fk(:,:,1),2); % No. of States  p = size(Hk(:,:,1),1); % No. of Measurements  steps = length(x\_nom(1,:))-1; % No. of k steps  dx = zeros(n,steps+1);  P = zeros(n,n,steps+1);  S = zeros(p,p,steps+1);  K = zeros(n,p,steps+1);    dx(:,1) = dx\_init; % initial dx  dy = y - y\_nom; % ground truth dx  if wrapOn == true  dx = WrapX(dx);  dy = WrapY(dy);  end    P(:,:,1) = P\_init; % inital P  I = eye(n);    y\_est = zeros(5,steps+1);  y\_est(:,1) = y\_nom(:,1) + Hk(:,:,1)\*(dx(:,1));    e\_y = zeros(p,steps+1);  e\_y(:,1) = y(:,1) - y\_est(:,1);% error in measurement estimates  if wrapOn == true  e\_y = WrapY(e\_y);  end    S(:,:,1) = Hk(:,:,1)\*P(:,:,1)\*Hk(:,:,1)' + R;  K(:,:,1) = P(:,:,1)\*Hk(:,:,1)'\*inv(S(:,:,1));    for i=1:steps  % Prediction Step  dx(:,i+1) = Fk(:,:,i)\*dx(:,i);  if wrapOn == true  dx = WrapX(dx);  end  P(:,:,i+1) = Fk(:,:,i)\*P(:,:,i)\*Fk(:,:,i)' + Ok(:,:,i)\*Q\*Ok(:,:,i)';    % Correction Step  S(:,:,i+1) = Hk(:,:,i+1)\*P(:,:,i+1)\*Hk(:,:,i+1)' + R;  K(:,:,i+1) = P(:,:,i+1)\*Hk(:,:,i+1)'\*inv(S(:,:,i+1));  y\_est(:,i+1) = y\_nom(:,i+1) + Hk(:,:,i+1)\*(dx(:,i+1));    e\_y(:,i+1) = y(:,i+1) - y\_est(:,i+1);  if wrapOn == true  e\_y = WrapY(e\_y);  end  P(:,:,i+1) = (I - K(:,:,i+1)\*Hk(:,:,i+1))\*P(:,:,i+1);    dx(:,i+1) = dx(:,i+1) + K(:,:,i+1)\*(dy(:,i+1) - Hk(:,:,i+1)\*dx(:,i+1));  if wrapOn == true  dx = WrapX(dx);  end  end    x\_est = x\_nom + dx;  e\_x = x - x\_est;  if wrapOn == true  x\_est = WrapX(x\_est);  e\_x = WrapX(e\_x);  end  end |
| function [x\_truth,y\_truth,x\_est,y\_est,dx,P,S,e\_x,e\_y] = LKF\_MonteCarlo(Q,R,steps)  load("cooplocalization\_finalproj\_KFdata.mat");    x\_nom = [10 0 pi/2 -60 0 -pi/2]';  u\_nom = [2 -pi/18 12 pi/25]';  x\_pert = [0 1 0 0 0 0.1]';  Dt = 0.1;    n = size(x\_nom,1);  P\_init = diag([1 1 0.025 1 1 0.025]);    t = 0:Dt:steps\*Dt;  Rtrue(2,2)=1;Rtrue(4,4)=1;Rtrue(5,5)=1;  % Generate truth model outputs for nominal trajectories  [x\_truth, y\_truth] = GenerateTruth(x\_nom, u\_nom, P\_init, Qtrue, Rtrue, Dt, steps, true);    % Generate nominal trajectories  [~,x\_NL] = ode45(@(t,x) NL\_DynModel(t,x,u\_nom,zeros(6,1)),t,x\_nom);  x\_NL = x\_NL';  y\_NL = zeros(5,length(t));  for i=1:length(t)  y\_NL(:,i) = NL\_MeasModel(x\_NL(:,i),zeros(5,1));  end    % Generate DT matrices along nominal trajectory  x\_nominal = x\_NL;  y\_nominal = y\_NL;    x\_nominal = WrapX(x\_NL);  y\_nominal = WrapY(y\_NL);    Fk = zeros(6,6,length(t));  Hk = zeros(5,6,length(t));  Ok = zeros(6,6,length(t));    for i=1:length(t)  [A\_t, B\_t, C\_t] = Linearize(x\_nominal(:,i),u\_nom);  [Fk(:,:,i), ~ , Hk(:,:,i)] = Discretize(A\_t,B\_t,C\_t, Dt);  Ok(:,:,i) = eye(6);  end    % Run LKF on Data  dx\_init = x\_truth(:,1)-x\_nominal(:,1);  P\_init = eye(6);    [x\_est,y\_est,dx,P,S,e\_x,e\_y] = LKF(dx\_init,P\_init,x\_nominal,y\_nominal,x\_truth,y\_truth,Fk,Hk,Ok,Q,R, true);  end |
| clc  clear  load('cooplocalization\_finalproj\_KFdata.mat')  seed = 100;  rng(seed);  Dt = 0.1;  steps = 1000;  n = 6; p = 5; t = 0:Dt:steps\*Dt;    N = 50; % No. of Monte Carlo runs  NEES\_all = zeros(N,steps+1);  NIS\_all = zeros(N,steps+1);    % Tuning parameters  Q = diag([0.0001 0.0001 0.01 0.1 0.1 0.01]);  R = Rtrue;  for i=1:N  disp(i)  NEES = zeros(1,steps+1);  NIS = zeros(1,steps+1);    [x\_truth,y\_truth,x\_est,y\_est,dx,P,S,e\_x,e\_y] = LKF\_MonteCarlo(Q,R,steps);    for k=1:steps+1  NEES(:,k) = e\_x(:,k)'\*inv(P(:,:,k))\*e\_x(:,k);  NIS(:,k) = e\_y(:,k)'\*inv(S(:,:,k))\*e\_y(:,k);  end    NEES\_all(i,:) = NEES;  NIS\_all(i,:) = NIS;  end    NEES\_bar = zeros(1,steps+1);  NIS\_bar = zeros(1,steps+1);    for i=1:steps+1  NEES\_bar(1,i) = mean(NEES\_all(:,i));  NIS\_bar(1,i) = mean(NIS\_all(:,i));  end    alpha = 0.01;  rx1 = (chi2inv(alpha/2,N\*n))./N;  rx2 = (chi2inv(1-alpha/2,N\*n))./N;  ry1 = (chi2inv(alpha/2,N\*p))./N;  ry2 = (chi2inv(1-alpha/2,N\*p))./N; |