```
% xk = xk-1 + Tuk + wk
% yk = xc - rk = xk + nk

clear
%Load dataset
load('dataset1.mat')
deltas = [1,10,100,1000];
%Loop over deltas
for delta = deltas
    %Range readings used
    r_used = r(1+delta:delta:end);
    %Range reading indices used
    i_y = 10*t(1+delta:delta:end);

    if mod((length(r)-1),delta) ~= 0
        r_used = [r_used;r(end)];
        i_y = [i_y;10*t(end)];
    end

    %Measurements Used
    y_used = l - r_used;
    %Odometry Velocities Used
    v_used = v(2:end);
    %Z Vector
    z = [v_used;y_used];
    %Number of velocities used
    K = size(v_used,1);
    %Number of measurements used
    y_K = size(y_used,1);

    %Create A Inverse matrix
    I_K = eye(K);
    I_y = eye(y_K);
    O_K = zeros(K,1);
    pos = [O_K,I_K];
    neg = [-I_K,O_K];

    A_inv = pos + neg;

    %Create C Matrix with only taking a measurement every delta time steps
    C = zeros(y_K,K+1);
    acc = 1;
    for i = i_y'
        C(acc,uint32(i+1)) = 1;
        acc = acc + 1;
    end

    %Create H marix
    H = [A_inv;C];
```

```
%Create W Inverse matrix
Q_inv = (1/v_var)*I_K;
R_inv = (1/r_var)*I_y;
Os_K = zeros(K,y_K);
Os_yK = zeros(y_K,K);
W_inv = [Q_inv,Os_K;Os_yK,R_inv];

%%
% Left hand side of linear least squares problem
A = H'*W_inv*H;
% Right hand side of linear least squares problem
b = H'*W_inv*z;
%%

% Perform Cholesky factorization
[L,flag] = chol(A,"lower");

% Solve lower triangular linear system
opts.LT = true;
d = linsolve(L,b,opts);
opts.LT = false;

% Solve upper triangular linear system
opts.UT = true;
x_hat = linsolve(L',d,opts);
opts.UT = false;

% Calculate covariance by inverting left hand side of linear system
cov = inv(A);

% Take only the variance of each individual point
cov_entries = diag(cov);
% Get standard deviation
std_dev = sqrt(cov_entries);
% Difference between predicted and true state
x_diff = x_hat - x_true;

% Upper and lower bounds of standard deviation (+/- 3 std deviations)
x_upper = 3*std_dev;
x_lower = -3*std_dev;

% Plot difference between predicted and true state
figure
plot(t,x_upper,':',t,x_lower,':',t,x_diff,'-');
title(["xk* - xk as a Function of Time for Delta = ", num2str(delta)]);
xlabel("Time t (s)");
ylabel("xk* - xk (m)");
legend(["+3 Standard Deviations", "-3 Standard Deviations","xk* - xk"]);
fname = sprintf('Plot_%d.png', delta);
saveas(gcf,fname)
```

```
    % Plot histogram of errors between predicted and true state
    figure
    histogram(x_diff,'Normalization','probability');
    title(["Histogram of Errors xk* - xk for Delta = ", num2str(delta)]);
    xlabel("Position error (m)");
    ylabel("Frequency");
    fname = sprintf('Histogram_%d.png', delta);
    saveas(gcf,fname)

end
```