ECEN426 Advanced Mechatronic Systems

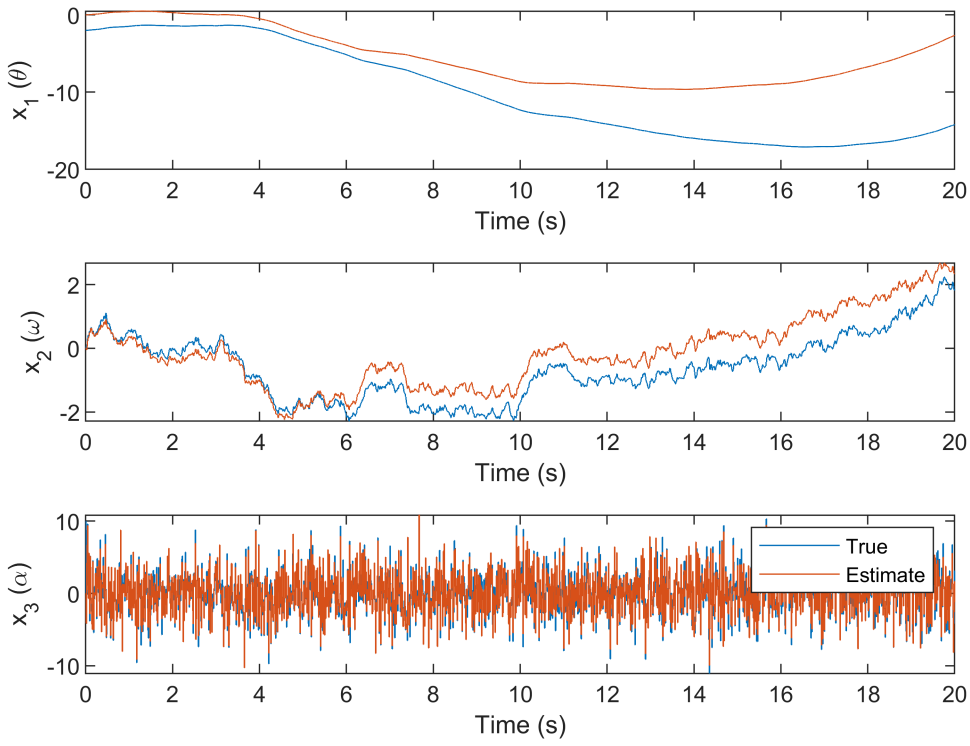
Assignment 3

Daniel Eisen - 300447549

(a) [5 marks] Choose appropriate initial conditions $\boldsymbol{x}(0|0)$ *and* $\boldsymbol{P}(0|0)$ *for the filter. At* $t = 0$ *the robot head rests against a stop, which corresponds to a starting angle of* $(0 \pm 5)^{\circ}$

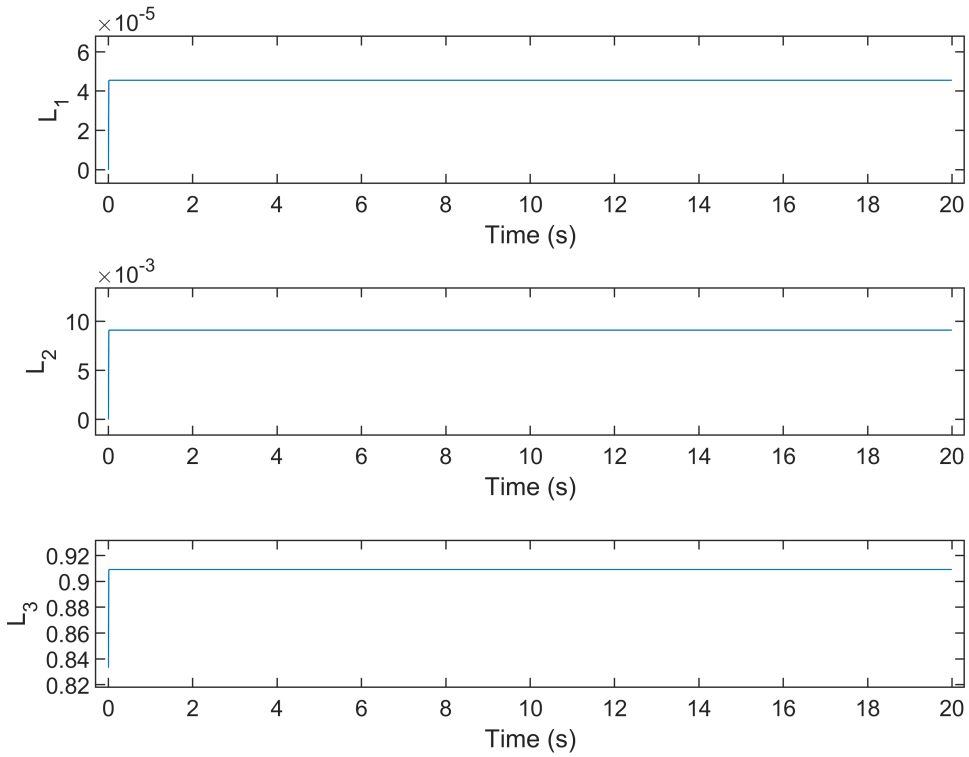
Demonstrate the operation of the filter using only sensor one.

```
clear;clc;

% Our initial guess at the neck position.
x_post(:,1) = [0,0,0]';     % Put your initial estimate here.
P_post = diag([20,5,5]);     % Put your initial estimate here.
run("robot_head_one_sensor.m")
figure();
labels = [" (\theta)"," (\omega)"," (\alpha)"];
for plot_num = 1:3
  subplot(3,1,plot_num)
  stairs(t,[x(plot_num,:);x_post(plot_num,:)]')
  ylabel(['x_' num2str(plot_num)] + labels(plot_num))
  xlabel('Time (s)')
end
legend("True","Estimate")
```



(b) [5 marks] Demonstrate the start up transient of the Kalman filter by plotting $\mathbf{L}$ as a function of time.

```
figure()
for plt_n = 1:3
    subplot(3,1,plt_n)
    plot(t,L_acc(:,plt_n))
    ylabel(['L_' num2str(plt_n)])
    xlabel('Time (s)')
end

subplot(3,1,1)
xlim([-0.3 20.3])
ylim([-0.000007 0.000068])
subplot(3,1,2)
xlim([-0.3 20.3])
ylim([-0.0016 0.0134])
subplot(3,1,3)
xlim([-0.3 20.3])
ylim([0.818 0.932])
```

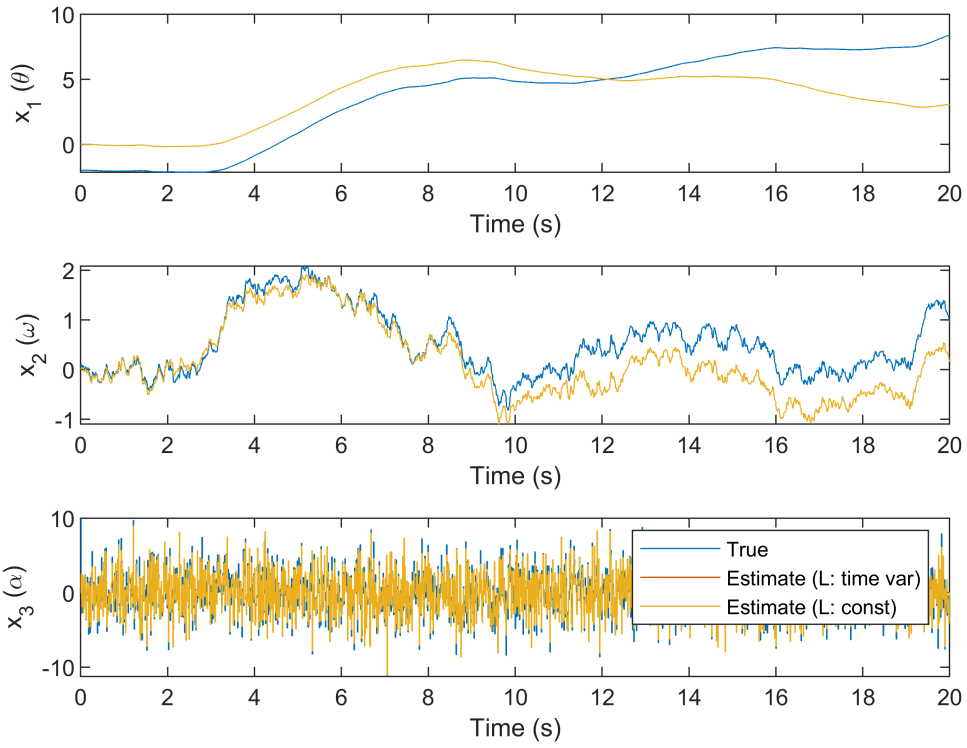


(c) [10 marks] Contrast the performance of the filter with a steady state Kalman filter.

You should use Matlab to design the steady state Kalman gain.

```
L_s = L;
```

```
% Our initial guess at the neck position.
x_post(:,1) = [0,0,0]';     % Put your initial estimate here.
P_post = diag([20,5,5]);     % Put your initial estimate here.
x_post_s = x_post;

run("robot_head_compare_SS.m")

figure();
labels = [" (\theta)"," (\omega)"," (\alpha)"];

for plot_num = 1:3
  subplot(3,1,plot_num)
  stairs(t,[x(plot_num,:);x_post(plot_num,:);x_post_s(plot_num,:)]')
  ylabel(['x_' num2str(plot_num)] + labels(plot_num))
  xlabel('Time (s)')
end
legend("True","Estimate (L: time var)", "Estimate (L: const)")
```



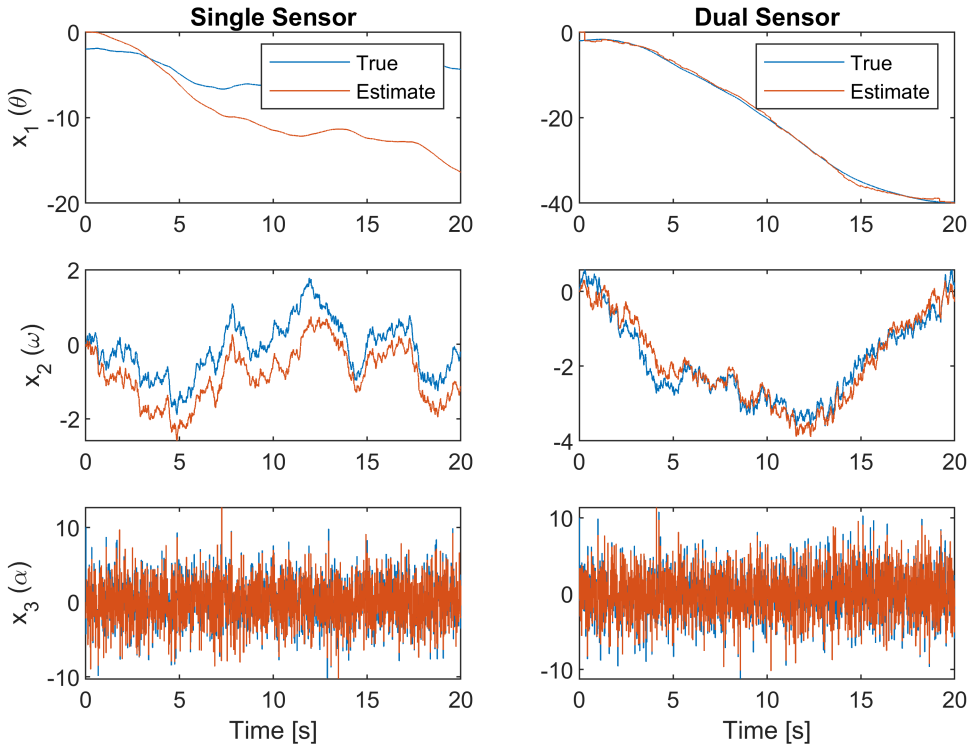
Ans. This shows that the steady state Kalman behaves identically to the Kalman filter with a time varient L. This is to be expect as c) showed that L reached a steady state pretty much immediatly.

(d) [10 marks] Add sensor two to the system and demonstrate its effect on the state estimate. Contrast the tracking performance with and without the extra sensor for a variety of initial conditions (that is consider examples of initial estimates that are both near and far from the robot's true state).

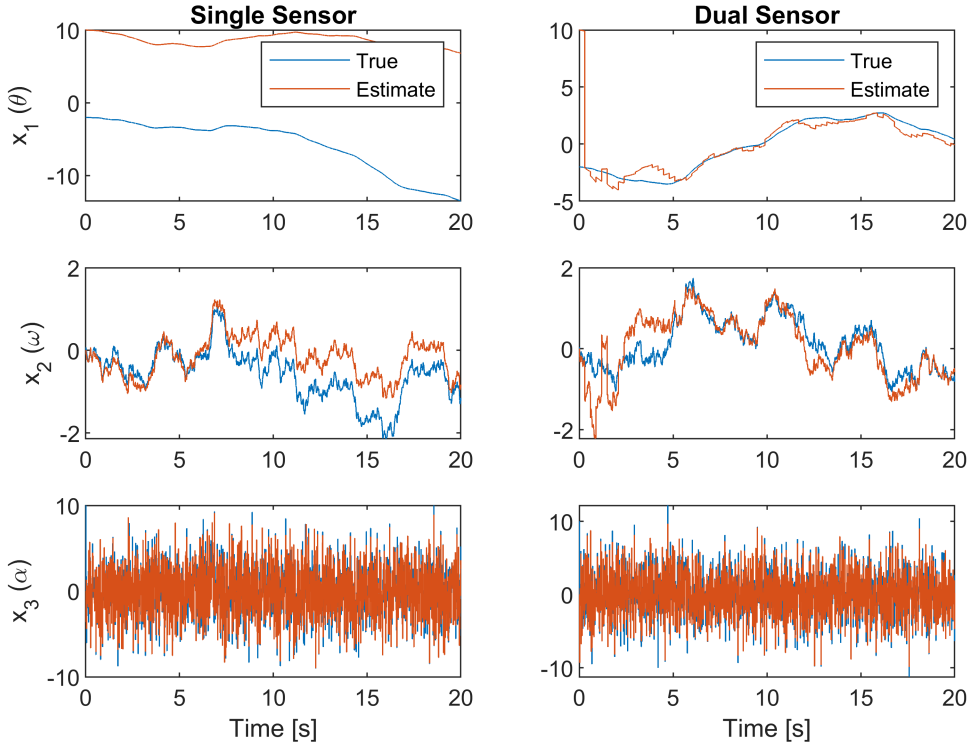
Hint: To do this you will need to add a section to the loop in the code so that a correction is only applied from the second sensor when appropriate. You may find it useful to know that the Matlab code `mod(k,8)==0` *will be one only when* k *is a multiple of eight for example.*

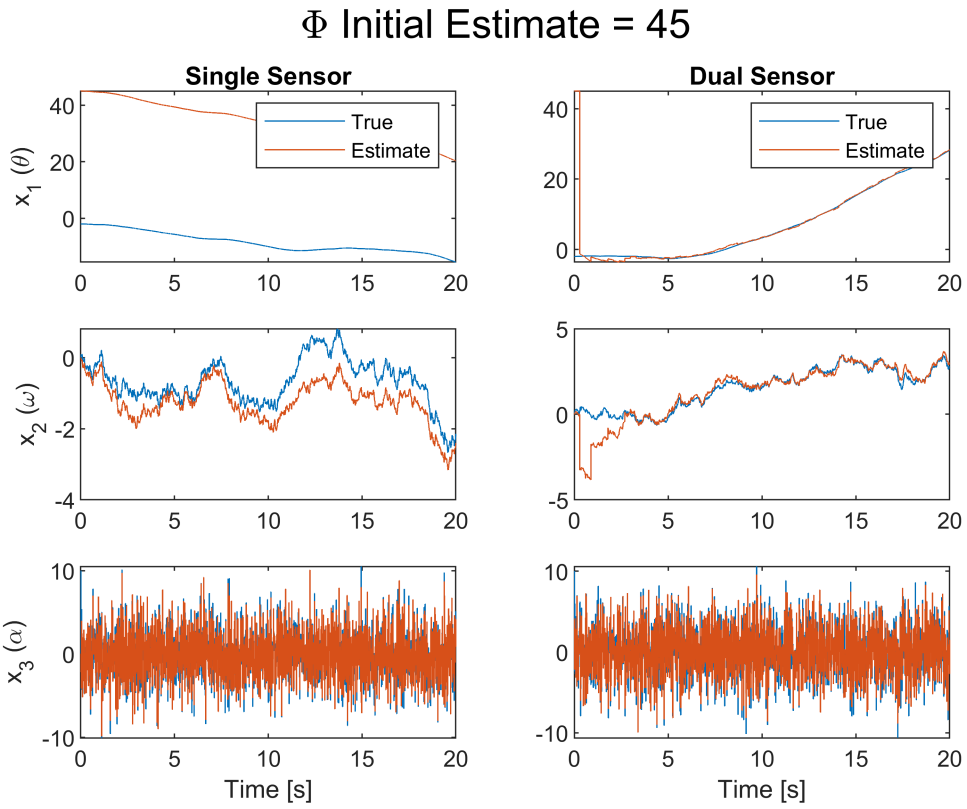
```
clear;clc;

for est = [0,10,45]
    x_post(:,1) = [est,0,0]';      % Put your initial estimate here.
    P_post = diag([20,5,5]);       % Put your initial estimate here.
    run("robot_head_one_sensor.m")

    figure();
    labels = [" (\theta)"," (\omega)"," (\alpha)"];
    for plot_num = [1, 3, 5]
        subplot(3,2,plot_num)
        stairs(t,[x(ceil(plot_num/2),:); x_post(ceil(plot_num/2),:)]')
        ylabel(['x_' num2str(ceil(plot_num/2))] + labels(ceil(plot_num/2)))
    end
    x_post(:,1) = [est,0,0]';      % Put your initial estimate here.
    P_post = diag([20,5,5]);       % Put your initial estimate here.
    run("robot_head_two_sensors.m")
    for plot_num = [2, 4, 6]
        subplot(3,2,plot_num)
        stairs(t,[x( plot_num/2,:); x_post(plot_num/2,:)]')
    end
    %% Add titles and axis
    subplot(3,2,1)
    legend("True", "Estimate")
    title("Single Sensor")
    subplot(3,2,2)
    legend("True", "Estimate")
    title("Dual Sensor")
    subplot(3,2,5)
    xlabel("Time [s]")
    subplot(3,2,6)
    xlabel("Time [s]")
    sgtitle(['\Phi Initial Estimate = ' num2str(est)])
end
```

Φ Initial Estimate = 0



Φ Initial Estimate = 10





Ans. These figures demonstrate that the addition of a the virtual sensor estimating θ quite drastically improves the systems ability to accurately estimate the true value of θ and quickly recover from an incorrect first guess.

(e) [10 marks] Extension – Validation Gating

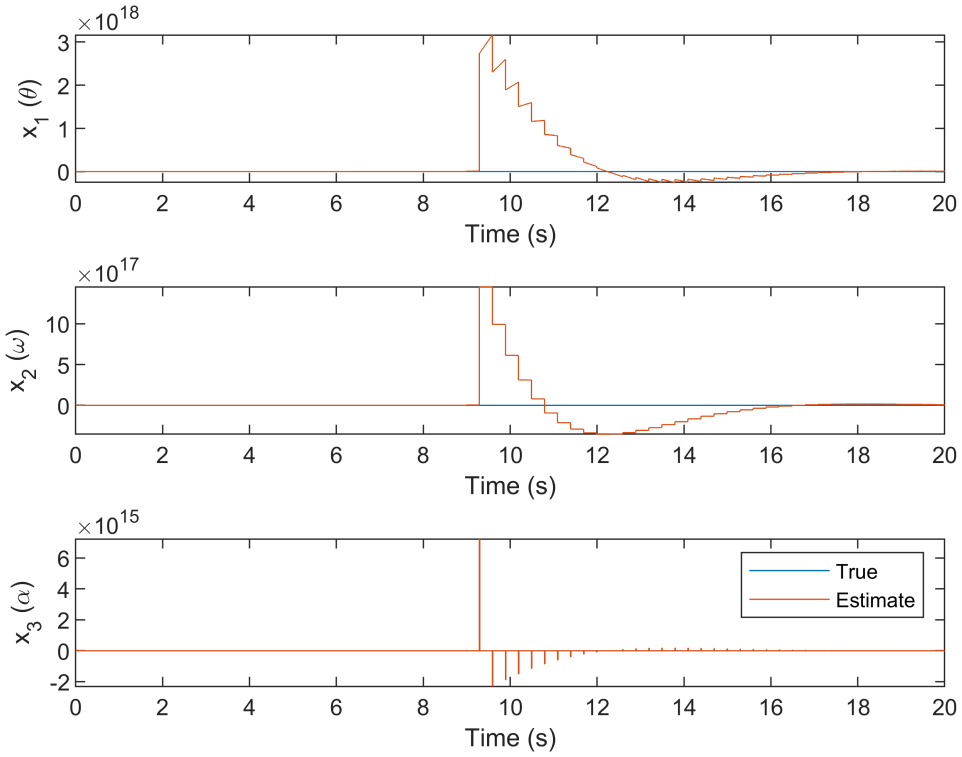
One common addition that is often made to the Kalman filter is a validation gate. The idea is that when a new sensor reading is obtained it is compared to the value that was expected (the prior estimate of the measurement). Measurements that are too far away from the expected value are rejected as errors; that is, they are not used to update the agent's estimate of its state.

Typically the Mahalanobis distance is used to measure whether the measurement is too far from its expected distance. Modify your code to include validation gating of some form and demonstrate it working by injecting occasional errors into the measurement stream. Be sure to explain your technique.

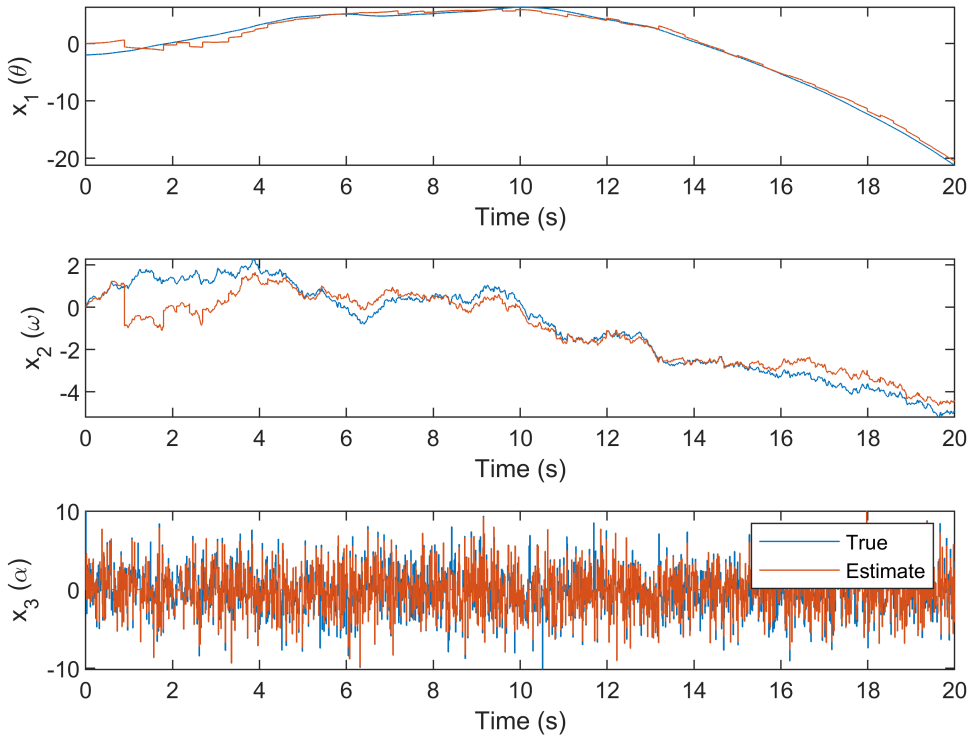
Hint: This question is deliberately open ended. Do not spend too long on it!

```
clear;clc;

error_ks = [30, 60, 120, 150, 300, 600, 660, 900, 930]; %point to add errors
threshold = 500; %gate threshold

% Our initial guess at the neck position.
x_post(:,1) = [0,0,0]';     % Put your initial estimate here.
P_post = diag([20,5,5]);     % Put your initial estimate here.
% run("robot_head_two_sensors.m")
run("robot_head_mes_error.m")
```

```
figure();
labels = [" (\theta)"," (\omega)"," (\alpha)"];
for plot_num = 1:3
  subplot(3,1,plot_num)
  stairs(t,[x(plot_num,:);x_post(plot_num,:)]')
  ylabel(['x_' num2str(plot_num)] + labels(plot_num))
  xlabel('Time (s)')
end
legend("True","Estimate")
```




```
% Our initial guess at the neck position.
x_post(:,1) = [0,0,0]';     % Put your initial estimate here.
P_post = diag([20,5,5]);     % Put your initial estimate here.
% run("robot_head_two_sensors.m")
run("robot_head_valid_gate.m")
figure();
labels = [" (\theta)"," (\omega)"," (\alpha)"];
for plot_num = 1:3
  subplot(3,1,plot_num)
  stairs(t,[x(plot_num,:);x_post(plot_num,:)]')
  ylabel(['x_' num2str(plot_num)] + labels(plot_num))
  xlabel('Time (s)')
end
legend("True","Estimate")
```



Technique: When a sensor takes a measurement, this 'distance' between it and the prior measurement is computed. If this exeeds some threshold, the current measurent is rejected and over written with the prior. This happens inside the `if` blocks that handle sensor updates.

To inject occasional error the measurement `(y)` is occationally multiplied by a random number.

```
% introduce error

if ismember(k,error_ks)

y(:,k) = C_2*x_post(:,k-1)*threshold*2;

end

%gate them

if k>2 %ignore first 0

%is new measurement 2 mag away from the last value

expect = C_2*x_post(:,k-1);

if (y(:,k)/expect >= threshold) || (expect/y(:,k)) >= threshold

y(:,k) = expect;

end

end
```