Question 1a

Wednesday, September 28, 2022 8:08 PM

1. Given an open-loop transfer function $G(s) = \frac{200}{s^3+22s^2+141s+2}$, answer the following questions (5 points each):

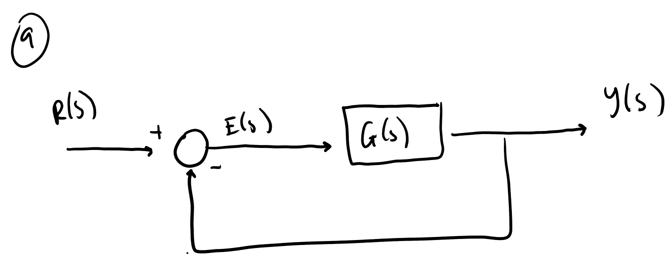
   (a) (5 points) Determine the closed-loop transfer function $T(s) = \frac{Y(s)}{R(s)}$ with unity negative feedback.

   (b) (5 points) Determine poles and zeros of $T(s)$.

   (c) (5 points) Plot $y(t)$ using MATLAB's *step* function and discuss which poles of $T(s)$ dominate the response and why?

   (d) (5 points) Find the steady state value using Final Value Theorem.

(a)



EQs.

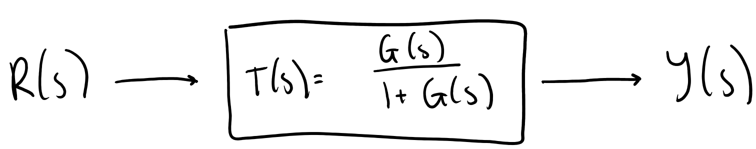
(1) $E(s) = R(s) - Y(s)$

(2) $Y(s) = G(s)E(s)$

(1) IN (2):

(DROP "s")

$$Y = G(R - Y)$$
$$Y = GR - GY$$
$$Y + GY = GR$$
$$Y(1+G) = GR$$
$$T(s) = \frac{Y(s)}{R(s)} = \frac{G(s)}{1+G(s)}$$



Question 1b, 1c

Wednesday, September 28, 2022 9:08 PM

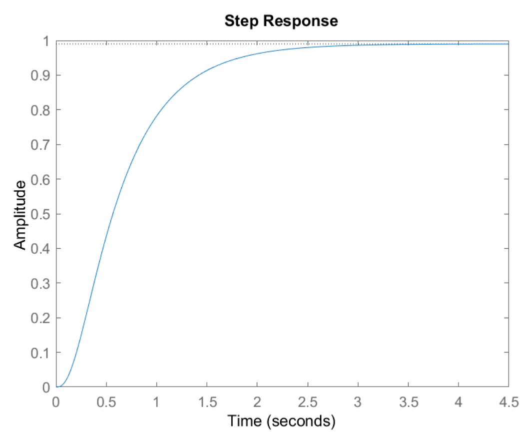
1b: List the Poles and Zeros of the Closed Loop Transfer Function

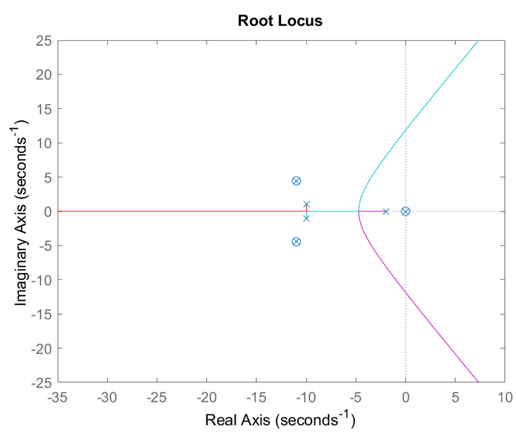
RED ZEROS/POLES CANCEL

```
The zeros of T_cl are:
 -10.9929 + 4.4546i
 -10.9929 - 4.4546i
  -0.0142 + 0.0000i

The poles of the T_cl are:
 -10.9929 + 4.4546i
 -10.9929 - 4.4546i
 -10.0000 + 1.0000i
 -10.0000 - 1.0000i
  -2.0000 + 0.0000i
  -0.0142 + 0.0000i
```

DOMINANT POLE

1c: Plot the step response and discuss which poles are dominant & why





Dominant Poles:

The dominant pole of the system is the last pole given in part 1b: -0.0142. Visually we can see this from the root locus plot, which shows this pole as being the one closest to the imaginary axis.

Question 1.d

Wednesday, September 28, 2022 9:58 PM

(D)

$$T(s) = \frac{200s^3 + 4400s^2 + 28200s + 400}{s^6 + 44s^5 + 766s^4 + 6408s^3 + 24369s^2 + 28764s + 404}$$

$$\lim_{s \to 0} T(s) = \frac{\cancel{200s^3} + \cancel{4400s^2} + \cancel{28200s} + 400}{\cancel{s^6} + \cancel{44s^5} + \cancel{766s^4} + \cancel{6408s^3} + \cancel{24369s^2} + \cancel{28764s} + 404} = \frac{400}{404}$$

$$\boxed{y_s = \frac{400}{404} = \frac{100}{101} \cong 0.99}$$

Question 2

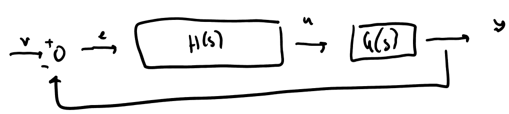
Tuesday, October 4, 2022 12:03 AM

2. (40 points) Implement a PID controller in MATLAB and use it to control the plant with transfer function

$$G(s) = \frac{s + 10}{s^4 + 71s^3 + 1070s^2 + 1000s}.$$

Your design objectives are to have a rise time of 0.5 seconds, a maximum percent overshoot of less than 5%, and a steady state error of zero. Tune the gains manually to achieve these objectives. List the final gains you choose and provide a plot of the resulting closed loop step response.

Find the Closed Loop Transfer Function:



$$H(s) = \frac{K_d s^2 + K_p s + K_i}{s}$$

$V - y = E$

$EH = u$

$Gu = y$

$y = GEH$

$y = G(V - y)H$

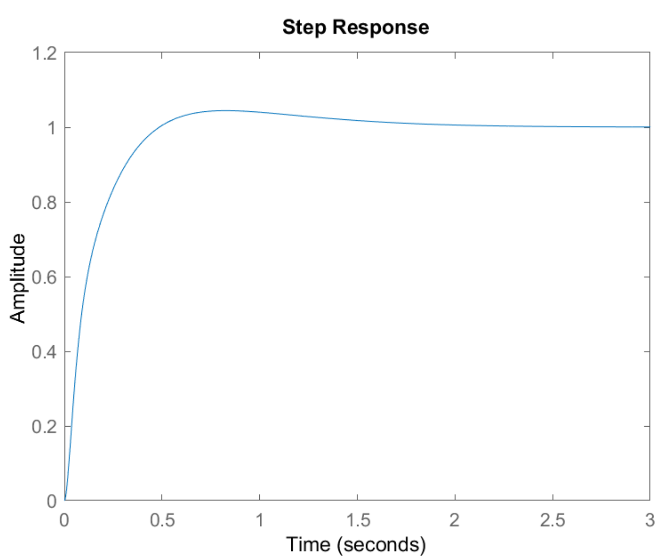
$y = GHV - GHy$

$y + GHy = GHV$

$y(1 + GH) = GHV$

$$\boxed{\frac{y}{V} = \frac{GH}{1 + GH}}$$

Plot the step response & list gains:



Question 2

Sunday, October 9, 2022 7:26 PM

List the Gains:

```
% Create PID transfer function
kd = 587;
kp = 1108;
ki = 0.001;
H = tf([kd, kp, ki], [1, 0]);

% create the Tf from the problem set
G_2 = tf([1, 10], [1, 71, 1070, 1000, 0]);
```

List the Transient Parameters

```
Transient Response Parameters:
        RiseTime: 0.2913
    TransientTime: 1.4431
     SettlingTime: 1.4431
      SettlingMin: 0.9020
      SettlingMax: 1.0443
        Overshoot: 4.4321
       Undershoot: 0
             Peak: 1.0443
         PeakTime: 0.8255

Steady State Error:
   5.1517e-04
```

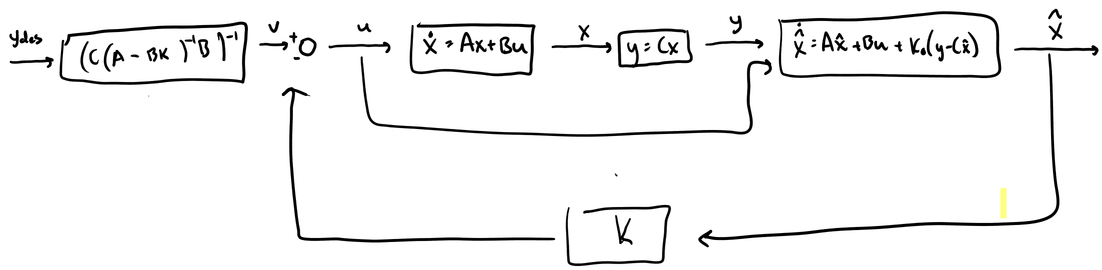
Steady state Error was calculated by finding the output of the system at the end of the 1 second

Question 3

Wednesday, October 5, 2022 9:44 PM

3. (40 points) Picking up from Problem 2(h) in Problem Set 1, build an observer for the cart-pendulum system and repeat your most successful tracking controller simulation replacing the state feedback you used in PS1 with the state estimate from your observer.

New Block Diagram



Plot the Actual state with the Observed State

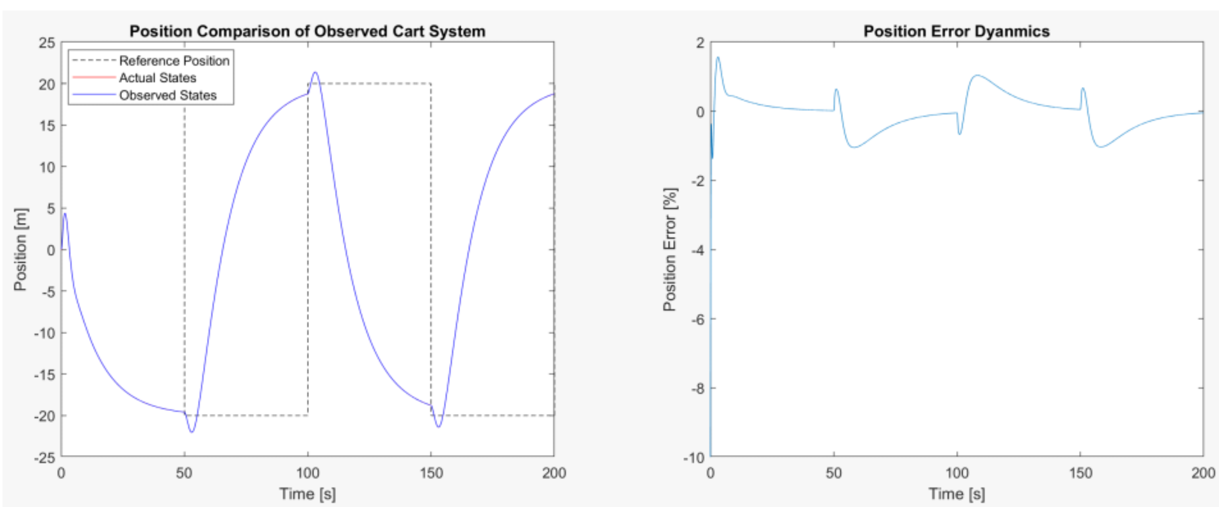


Fig. *Observer estimate of the cart system x position (left), and the error dynamics of the position (right). The initial conditions of the states were x0 = [0; 0; 0; 0] and the initial conditions of the estimated states were xhat0 = [0.1; 0.8; 0.5; 0.6].*

Question 3

Sunday, October 9, 2022 7:16 PM

Prove Observability

```
%% Find the observability of the system
% Create the matrix W = [C | CA | CA^2]
CA = C * A;
CA2 = C * A ^ 2;
CA3 = C * A ^ 3;
W = [C;  CA;  CA2; CA3];

obsv = rank(W) == 4;

disp("The system is observable: ")
disp(obsv)
```

Command Window

```
The system is observable:
   1
```

Plot the States

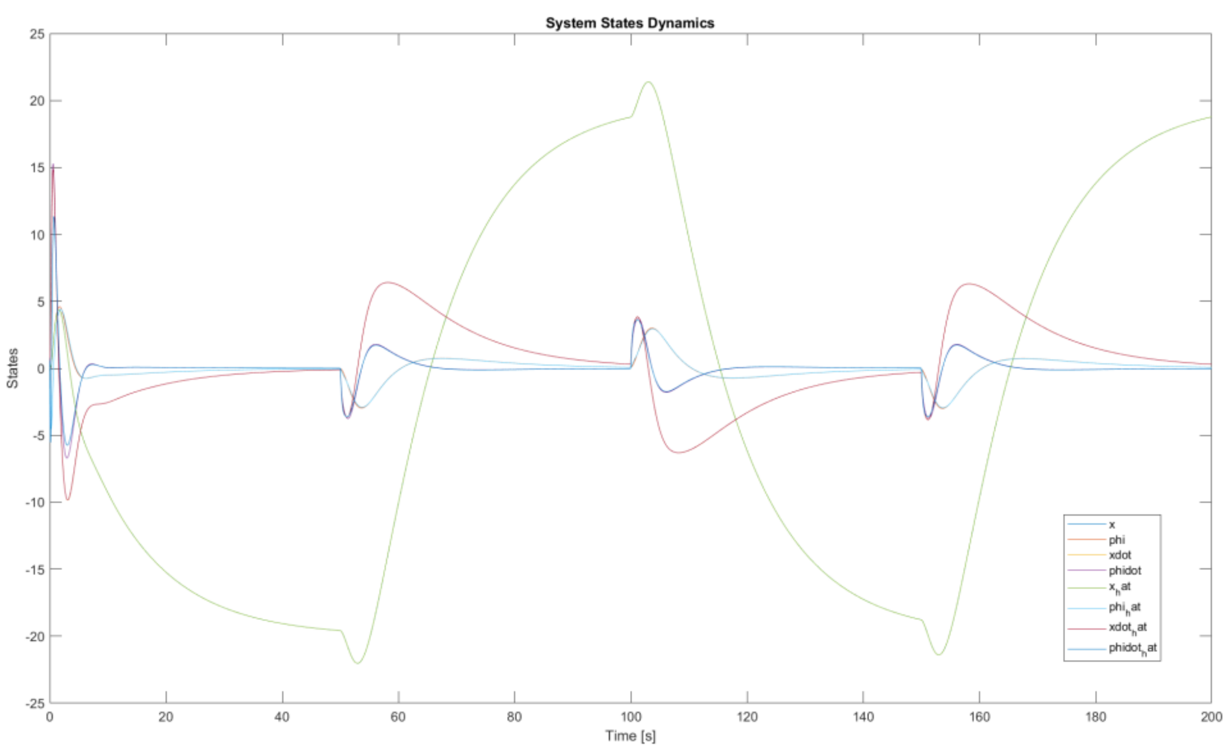


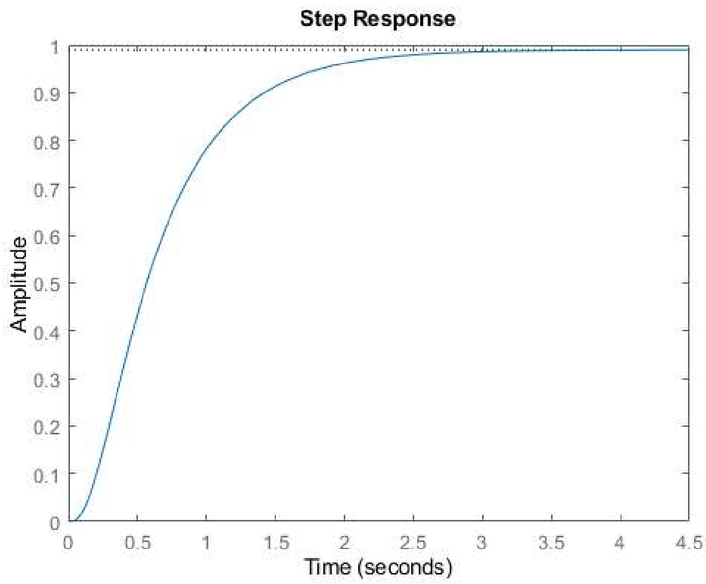
Fig. *Observer states of the cart system, and the actual states. The initial conditions of the states were x0 = [0; 0; 0; 0] and the initial conditions of the estimated states were xhat0 = [0.1; 0.8; 0.5; 0.6].*

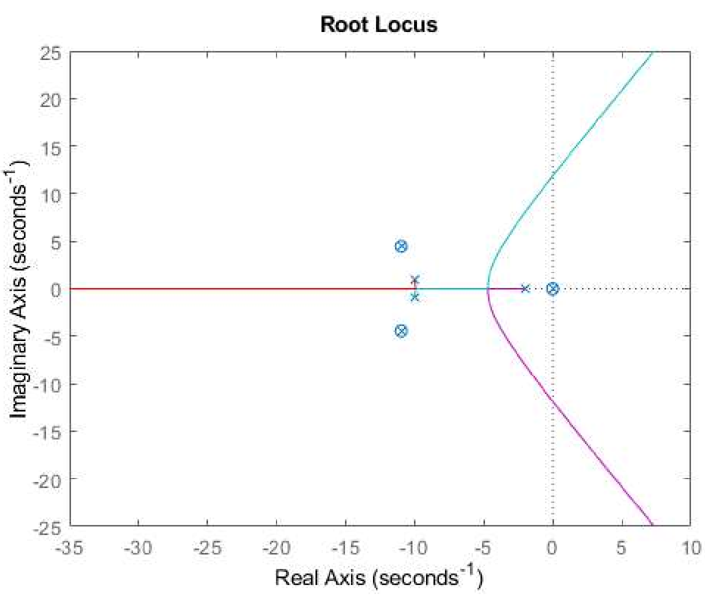
Contents

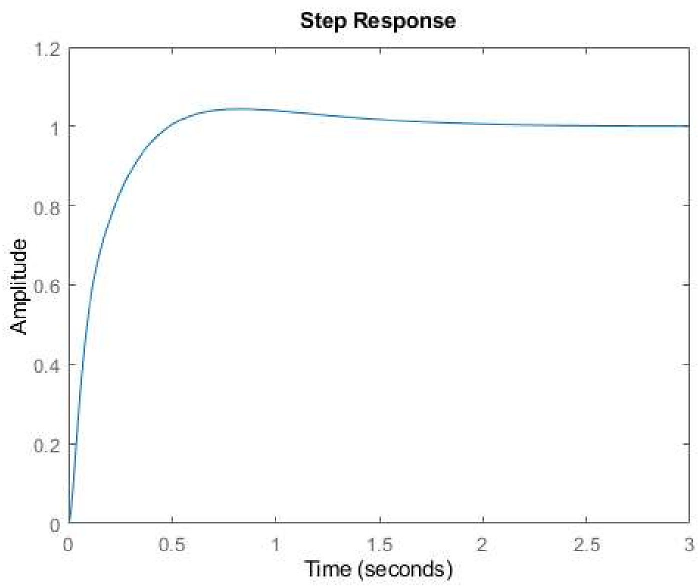

```
% Problem Set 2 | Alec Trela | Sept. 28, 2022
clc
clear
% Question 1.b
clc

% Find the Poles and Zeros of T(s)

% Create the tf G(s)
G = tf(200, [1, 22, 141, 2]);

% Create T, noting that T = G / (1 + G) during unity feedback
T = G / (1 + G);

% get the num, den for the tf2zp function
T_num = cell2mat(T.Numerator);
T_den = cell2mat(T.Denominator);

% Find the Poles and Zeros of the TF
[Z, P] = tf2zp(T_num, T_den);

% Print
disp("The zeros of T_cl are:")
disp(Z)
disp("The poles of the T_cl are:")
disp(P)

%Question 1.c
clc

% Create the step response of T
figure(1)
step(T)

% Plot the root locus to visualize poles, zeros
figure(2)
rlocus(T)

% Question 2
clc
% Create PID transfer function
kd = 587;
kp = 1108;
ki = 0.001;
H = tf([kd, kp, ki], [1, 0]);

% create the Tf from the problem set
G_2 = tf([1, 10], [1, 71, 1070, 1000, 0]);

% Create the closed loop tf
T_cl_2 = (G_2 * H) / (1 + (G_2 * H));

% plot the step response and get the transient response params
figure(3);
step(T_cl_2);
[y, t] = step(T_cl_2);
params = stepinfo(T_cl_2);
sserror = abs(1 - y(end));

disp("Transient Response Parameters:")
disp(params)
disp("Steady State Error:")
disp(sserror)
```

```
The zeros of T_cl are:
 -10.9929 + 4.4546i
 -10.9929 - 4.4546i
  -0.0142 + 0.0000i

The poles of the T_cl are:
 -10.9929 + 4.4546i
 -10.9929 - 4.4546i
 -10.0000 + 1.0000i
 -10.0000 - 1.0000i
```

```
   -2.0000 + 0.0000i
   -0.0142 + 0.0000i

Transient Response Parameters:
         RiseTime: 0.2913
    TransientTime: 1.4431
     SettlingTime: 1.4431
      SettlingMin: 0.9020
      SettlingMax: 1.0443
        Overshoot: 4.4321
       Undershoot: 0
             Peak: 1.0443
         PeakTime: 0.8255

Steady State Error:
   5.1517e-04
```







Question 3

```
clc
clear

% create the reference square wave
t200 = 0:0.01:200;
ref_pos = (20) * square(2*pi*(1/100)*t200);

A = [0 0 1 0;
    0 0 0 1;
    0 1 -3 0;
    0 2 -3 0];


B = [0;
    0;
    1;
    1];

C = [1 0 0 0];

Q = [15 0 0 0; 0 1 0 0; 0 0 15 0; 0 0 0 1];
R = 25;
```

Find the observability of the system

Create the matrix W = [C | CA | CA^2]

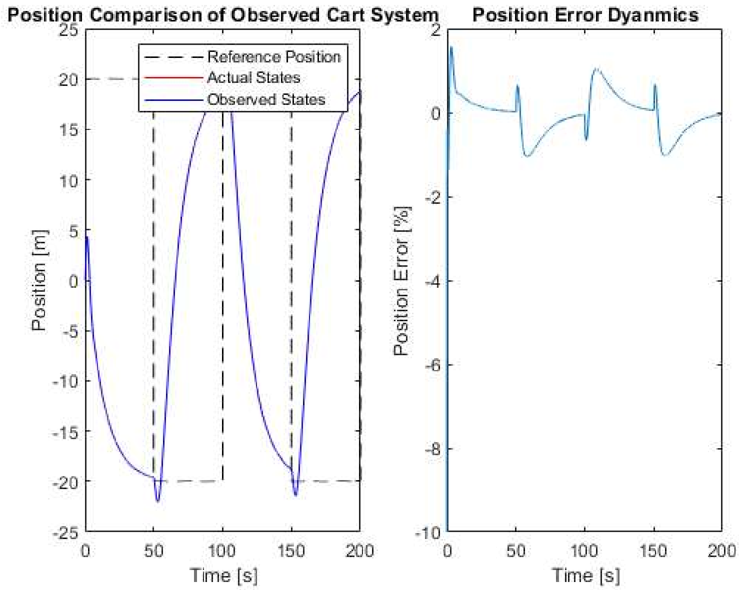
```
CA = C * A;
CA2 = C * A ^ 2;
CA3 = C * A ^ 3;
W = [C;  CA;  CA2; CA3];

obsv = rank(W) == 4;

disp("The system is observable: ")
disp(obsv)
```

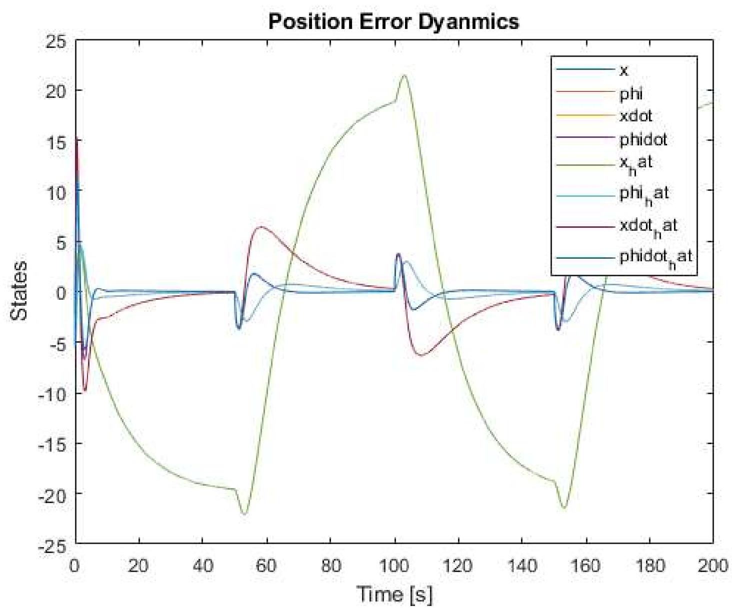
```
The system is observable:
   1
```

```
% initial conditions
x0 = [0; 0; 0; 0];
x0hat = [0.1; 0.8; 0.5; 0.6];

K = lqr(A, B, Q, R);

% place poles for observer
```

```
poles = [-6, -7 -8, -9];
Ktmp = place(transpose(A), transpose(C), poles);
K0 = transpose(Ktmp);

[xhatL, xL] = CartSys(0, 200, 0.01, x0, x0hat, K, K0, C);


figure(4)
subplot(121)
plot(t200, ref_pos, "Color", "black", "LineStyle","--")
hold on
plot(t200, xL(1, :), "Color", "red")
hold on
plot(t200, xhatL(1, :), "Color", "blue")
xlabel("Time [s]")
ylabel("Position [m]")
title("Position Comparison of Observed Cart System")
legend(["Reference Position", "Actual States", "Observed States"])
hold off
subplot(122)
plot(t200, (xL(1, :)-xhatL(1, :)) * 100)
title("Position Error Dyanmics")
xlabel("Time [s]")
ylabel("Position Error [%]")

figure(5)
plot(t200, xL)
hold on
plot(t200, xhatL)
hold off
title("Position Error Dyanmics")
xlabel("Time [s]")
ylabel("States")
legend(["x", "phi", "xdot", "phidot", "x_hat", "phi_hat", "xdot_hat", "phidot_hat"])

function [xhatL, xL] = CartSys(t0, tf, step, x0, xhat0, K, K0, C)

    % create the time inc of the whole model -> start:step:end
    time_inc = t0:step:tf;

    % init 3 steps to take in the smaller
    inner_step = step/3;

    % init x, xhat, u
    xL(:, 1) = x0;
    xhatL(:, 1) = xhat0;
    negKhat(1) = -K * xhat0;

    for t = 1:length(time_inc)-1

    % find time step of the outer loop
    time_at_step = time_inc(t);

    % create a small time interval to find x, xhat
    inner_inc = time_at_step:inner_step:(time_at_step+inner_step);

    % find the current state of
    x_curr = xL(:, t);

    % get the current input from the list of inputs
    negKhat_curr = negKhat(t);

    % integrate to find x across this step
    [~, step_x] = ode45(@(inner_inc, x_curr) cartController(inner_inc, x_curr, negKhat_curr, K), inner_inc, x_curr);

    % get the state at the end of the ode function
    new_x = step_x(end, :);

    % update x
    xL(:, t+1) = new_x;

    % get the output of the system at the end of the step
    curr_y = C * new_x.';

    % get the xhat for this current time step
    xhat_curr = xhatL(:, t);

    % integrate to find new xhat for th
    [~, step_xhat] = ode45(@(inner_inc, step_obs_state) cartObserver(inner_inc, step_obs_state, curr_y, negKhat_curr, K, K0), inner_inc, xhat_curr);

    % get the xhat at the end of the step
    new_xhat = step_xhat(end, :);
```

```
    % update xhat
    xhatL(:, t+1) = new_xhat;

    % update - k * xhat;
    negKhat(t+1) = -K * new_xhat.';

    end
end

function xhatdot = cartObserver(t, xhat, y, negKhat, K, K0)

    A = [0 0 1 0;
    0 0 0 1;
    0 1 -3 0;
    0 2 -3 0];


    B = [0;
        0;
        1;
        1];

    C = [1 0 0 0];

    ydes = (20) * square(2 * pi * (1/100) * t);

    v = inv(C * inv(A - B * K) * B) * ydes;

    y_des_last = (20) * square(2 * pi * (1/100) * (t - 0.001));

    u = v + negKhat;

    xhatdot = (A * xhat) + (B * u) + (K0 * (y - C * xhat));
end

% Cart Sys, change to car state
function xdot = cartController (t, x, negKhat, K)

    A = [0 0 1 0;
    0 0 0 1;
    0 1 -3 0;
    0 2 -3 0];

    B = [0;
        0;
        1;
        1];

    C = [1 0 0 0];

    ydes = (20) * square(2 * pi * (1/100) * (t));

    v = inv(C * inv(A - B * K) * B) * ydes;


    u = v + negKhat;

    xdot = (A * x) + (B * u);

end
```





Published with MATLAB® R2022a