Assignment for EE5101/ME5401 Linear Systems

Multivariable Controller Design for Diesel Engine Air System Control

Submitted by

Wang Yutong

A0225480J

E0576114@u.nus.edu

Department of Mechanical Engineering

National University of Singapore

23/11/2020

Session:2020/2021

Abstract

As diesel engine emissions standards become increasingly stringent, engine manufacturers need design engine air controllers to balance the engine's customer performance against the mandates of government agencies to reduce emissions. In this project, we will model the engine air system as a MIMO linear system and try to design some control methods to stabilize the system or track the reference signal. The control methods used in this project were taught in the lecture of Linear Systems ME5401. Finally, the validity of these methods is proved by the simulations in MATLAB/SIMULINK.

Contents

1.Introduction

1.1 Problem statement

With the increasing awareness of environmental protection, in recent years, governments of various countries have imposed stricter limits on vehicle exhaust emissions. It results in engine manufacturers need design engine air controllers to achieve the desired reduction in emissions. One of the most commonly employed method is active control of inducted air and recirculated exhaust gas (EGR) ,but it will form a complex, interactive, multivariable system. In this mini project, we will model this engine air system as a MIMO linear system and try to control it using the techniques I have learned in Linear Systems.

1.2 Model

For model-based control, first we need to build an effective dynamic model for our target plant. In this project, we use a linear fourth order state-space model to represent the target plant.

$$\dot{x} = Ax + Bu \qquad (1\text{-}1)$$

$$y = Cx \qquad (1\text{-}2)$$

$$A = \begin{bmatrix} -8.6487 & -0.0399 & -4.7500 & 3.5846 \\ -4.5740 & 0.9619 & -4.3662 & -0.9683 \\ 3.7698 & 14.5212 & -17.5853 & 4.4936 \\ -8.6645 & 8.3742 & -4.4331 & -11.1484 \end{bmatrix}$$

$$B = \begin{bmatrix} 0.2786 & 0.0319 \\ 0.0138 & -0.0200 \\ 4.4939 & 1.4986 \\ -1.4269 & -0.2730 \end{bmatrix}$$

$$C = \begin{bmatrix} -3.2988 & -1.5532 & 0.0370 & -0.0109 \\ 0.2522 & -2.1506 & -0.0104 & 0.0163 \end{bmatrix}$$

where the manipulated inputs $u = [u_1 \quad u_2]^T$ are the VGT Vane position VGT_{vane} and EGR valve position EGR_{value}respectively, which are common actuators used on diesel engines for air system management and emissions reduction. The outputs $y = [y_1 \quad y_2]^T$represent the in-cylinder air/fuel ratio (AFR) and intake manifold EGR percentage respectively. The states $x = [x_1 \quad x_2 \quad x_3 \quad x_4]^T$ have no direct physical meaning since they are determined by system identification.

2. design and analysis

2.1 Task 1: Pole placement

In this part, a state feedback controller will be designed by pole place method. Assume all the four state variables could be measured and both the disturbance and set point are zero. For this scenario, the feedback matrix K is a 2 by 4 matrix defined by the dimension of both A and B, which means that there are 8 unknows in K matrix. From the characteristic polynomial 5 equation can be constructed. The number of unknowns is larger than equations, making direct comparison method or unity rank method not a plausible way to solve the task. So I converted matrices into controllable canonical form to solve this task.

2.1.1 Choosing Reference Model

To meet the performance criteria (overshoot less than 10% and 2% settling time less than 20 seconds) for all the below-mentioned tasks, A reference model should be designed firstly. To simplify the task, started with a standard 2nd system:

$$H(s) = \frac{\omega_n^2}{s^2+2\zeta\omega_n s+\omega_n^2} \qquad (2\text{-}1)$$

settling time:$t_s = \frac{4}{\zeta\omega_n} < 20 \ \rightarrow \zeta\omega_n > 0.2$ (2-2)

Overshoot: $M_p = e^{-\frac{\pi\zeta}{\sqrt{1-\zeta^2}}} < 0.1 \rightarrow \zeta > 0.5912$ (2-3)

According to the equation (2-2) and (2-3),choose $\zeta = 0.8$,$\omega_n = 0.4$, The poles of H(s) will be $\lambda_{1,2} = -0.3200 \pm 0.2400i$.Because the system is a four-order system, two extra poles should be added and usually locate extra poles to be 2-5 times faster than dominant ones. Set $\lambda_3 = -1.28$, $\lambda_4 = -1.5$. Therefore, we can get the transfer function

$$G(s) = \frac{1}{(s+0.3200+0.2400i)(s+0.3200-0.2400i)(s+1.28)(s+1.5)} \qquad (2\text{-}4)$$

Then a possible desired closed-loop matrix is

$$A_d = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -0.3072 & -1.6736 & -3.8592 & -3.4200 \end{bmatrix}$$

The simulation result of the reference model is shown in Figure 2.1. By inspection, the reference

model can satisfy performance requirement.

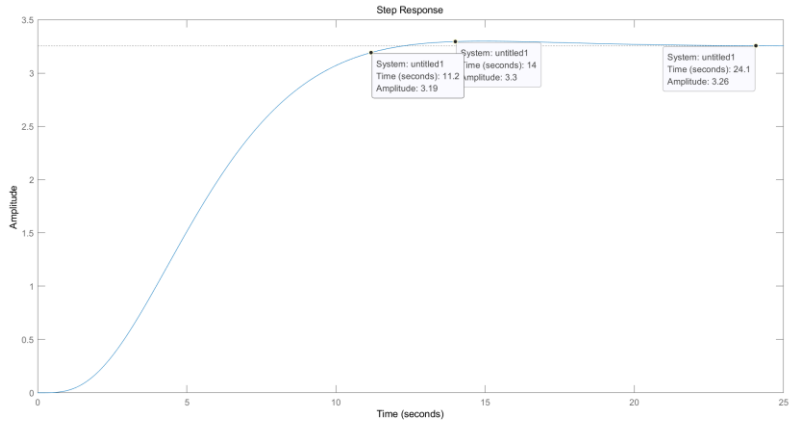

Figure 2.1 reference model step response

2.1.2 Computation Feedback Gain K

Then design the feedback control $u = -Kx + Fr$, where r is the reference input, K and F are constant matrices to be designed.

Firstly, we need to calculate the controllability matrix Wc before carrying out any solution.

$$W_c = [B\ AB\ A^2B\ A^3B] \qquad (2\text{-}5)$$

It can be verified by MATLAB that Wc is a full-rank matrix, namely the process is controllable. After that, we need to select n independent vectors out of Wc , strictly following the order from left to right. In my case the first four vectors are independent with each other, group them together as C in this sequence:

$$C = \{b_1 \quad Ab_2 \quad b_2 \quad Ab_2\} \qquad (2\text{-}6)$$

$$C^{-1} = \begin{bmatrix} q_1^T \\ q_2^T \\ q_3^T \\ q_4^T \end{bmatrix} \qquad (1\text{-}7)$$

For each input, there will be one row to be taken out to contrast T, in this case, the second and fourth rows should be taken out from C^{-1} corresponding to 2 inputs.

$$T = \begin{bmatrix} q_2^T \\ q_2^T A \\ q_4^T \\ q_4^T A \end{bmatrix} \qquad (2\text{-}8)$$

Therefore, the MIMO canonical form is obtained as

$$\dot{x} = \bar{A}x + \bar{B}u \qquad (2\text{-}9)$$

$$y = Cx$$

Where $\bar{A} = TAT^{-1}$,$\bar{B} = TB$

Set $\overline{K} = \begin{bmatrix} k_{11} & k_{12} & k_{13} & k_{14} \\ k_{21} & k_{22} & k_{23} & k_{24} \end{bmatrix}$

The closed loop matrix $\overline{A_C}$ is

$$\overline{A_c} = \overline{A} - \overline{BK} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -k_{11} - 201.7607 & -k_{12} - 14.9750 & -k_{13} - 55.3043 & -k_{14} - 0.9560 \\ 0 & 0 & 0 & 1 \\ -k_{21} - 91.8336 & -k_{22} - 30.4032 & -k_{23} - 54.9727 & -k_{24} - 21.4455 \end{bmatrix}$$

$\overline{K}$ can be obtained by solving equation $A_d = \overline{A_c}$,then the real feedback gain is obtained by transforming the canonical feedback gain $\overline{K}$ into unit coordinate using

$$K = \overline{K}T = \begin{bmatrix} -4.4400 & -12.0287 & 2.4134 & 17.1122 \\ 0.1465 & 43.9714 & -18.0495 & -36.2574 \end{bmatrix}$$

2.1.3 Simulation results

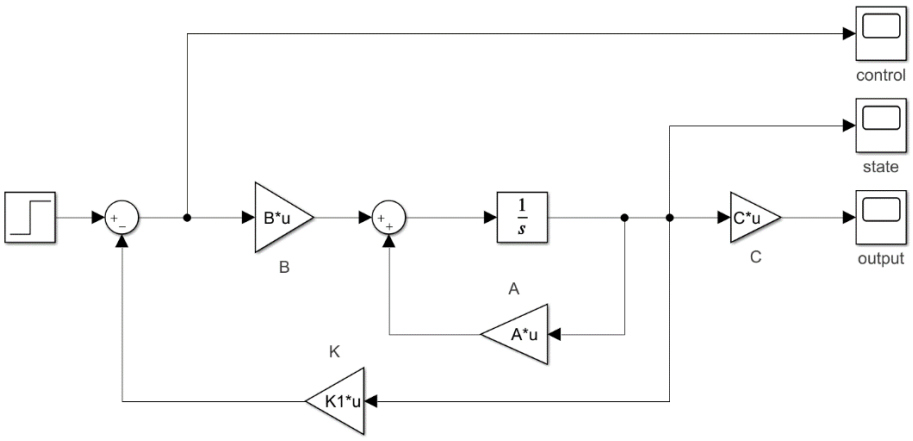

Figure 2.2 state feedback Block Diagram

This step response is checked by giving a step reference signal for each input channel, with zero initial condition, the result is shown in Figure2.3 and Figure2.5.When r=[1 0] ,overshoot of y_1 and y_2 are 1.48% and 1.37% respectively, When r=[0 1] ,overshoot of y_1 and y_2 are 1.39% and 0.085% respectively. They settling time all less than 20 seconds, so the model can satisfy performance requirement and the control signal also in an acceptable range.

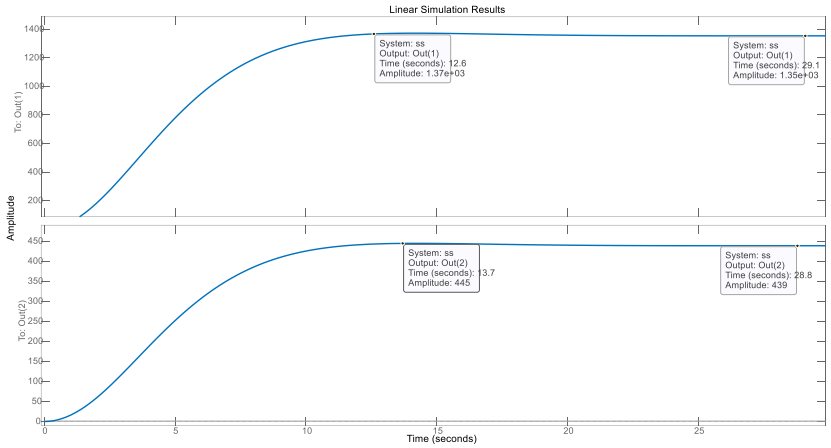

Figure 2.3 step response when r=[1 0]

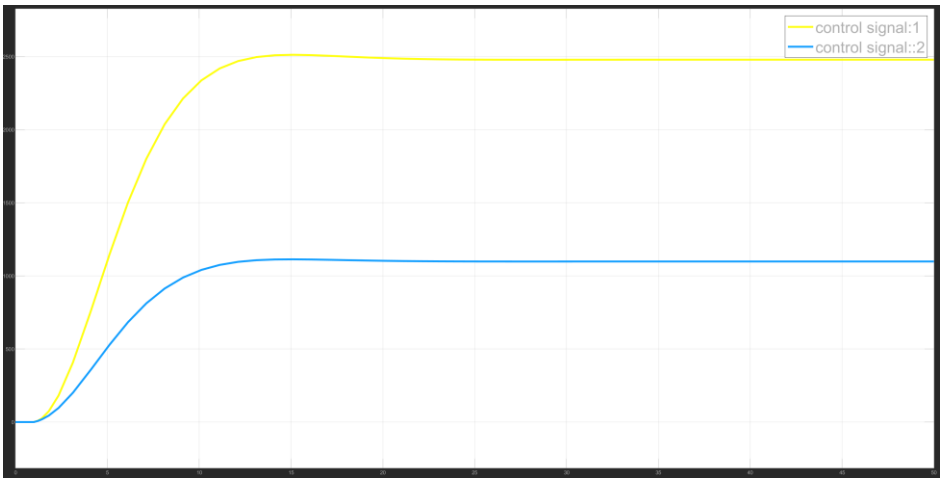

Figure 2.4 control signal when r=[1 0]

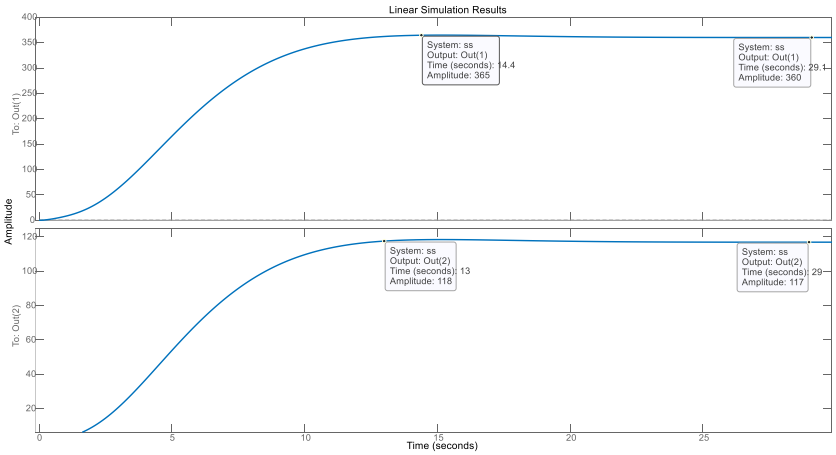

Figure 2.5 step response when r=[0 1]

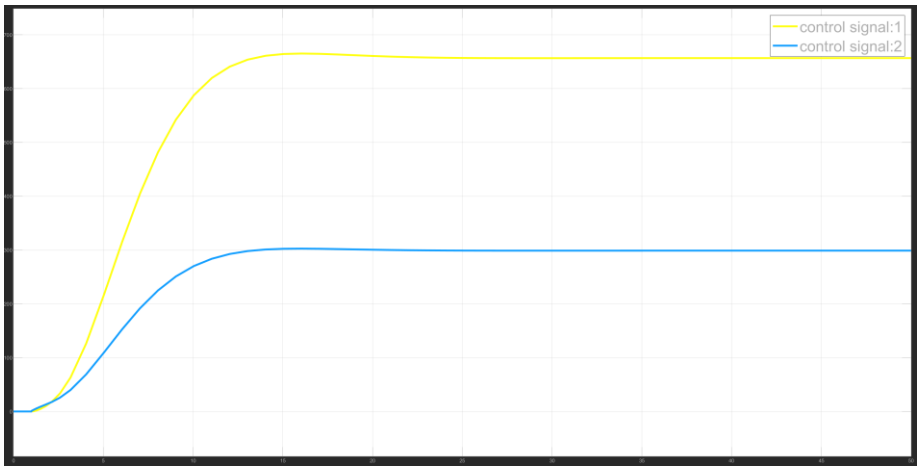

Figure 2.6 control signal when r=[0 1]

State changing with respect to time under zero external input and non-zero initial state is shown in Figure 2.7. From the we can inspect that states vibrate dramatically before 15 second. More seriously,x_4 has a -69.8 overshot at around 2.8 second, it is unbearable to real system even if all the states can finally converge to zero.

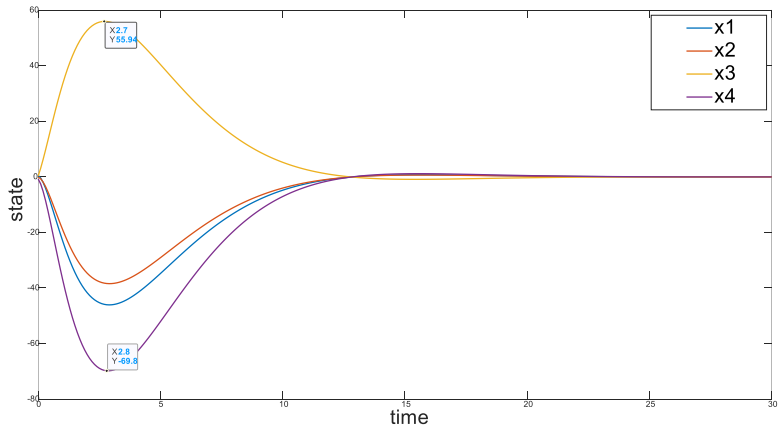

Figure 2.7 state trend under zero-input and non-zero initial state

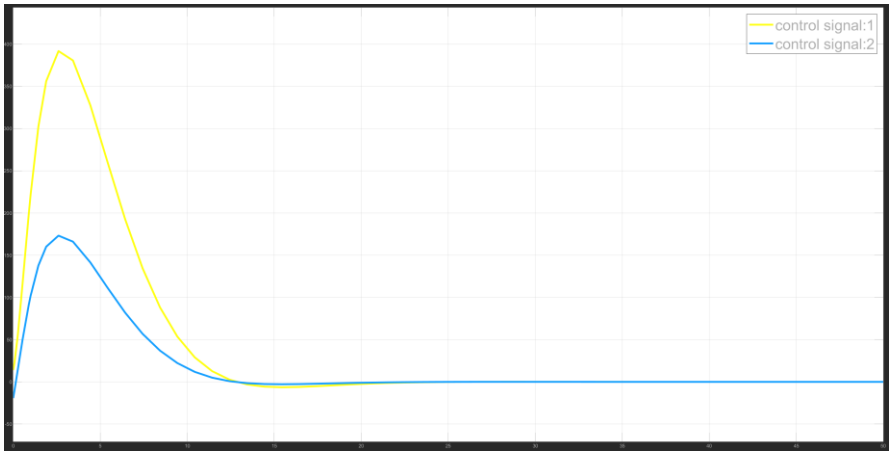

Figure 2.8 control signal under zero-input and non-zero initial state

2.1.4 effects of the positions of the poles on system performance

- Stability

As figure 2.9 show, When the real part of the pole is greater than 0, the system is unstable, when the real part is less than 0, the system is stable, and when the real part is equal to 0, the system is critically stable.

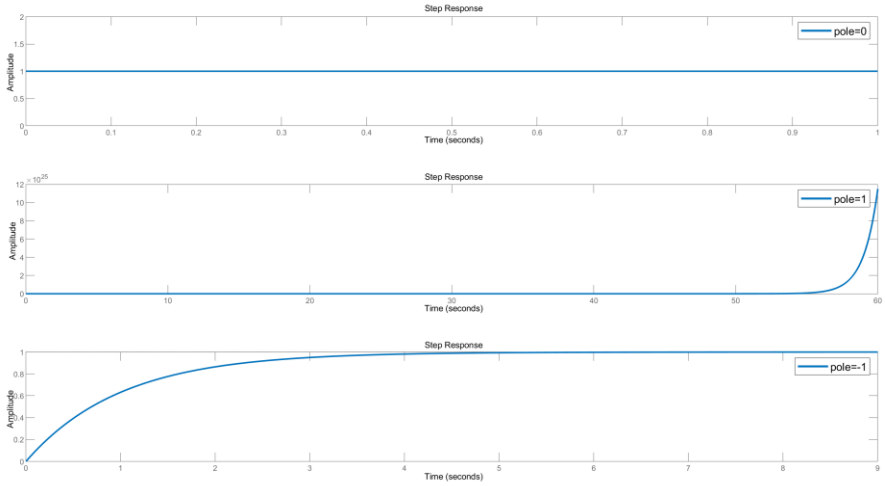

Figure 2.9 output of different pole

- oscillation amplitude

  As figure 2.10 show, The greater the imaginary part of the pole, the greater the amplitude of the oscillation

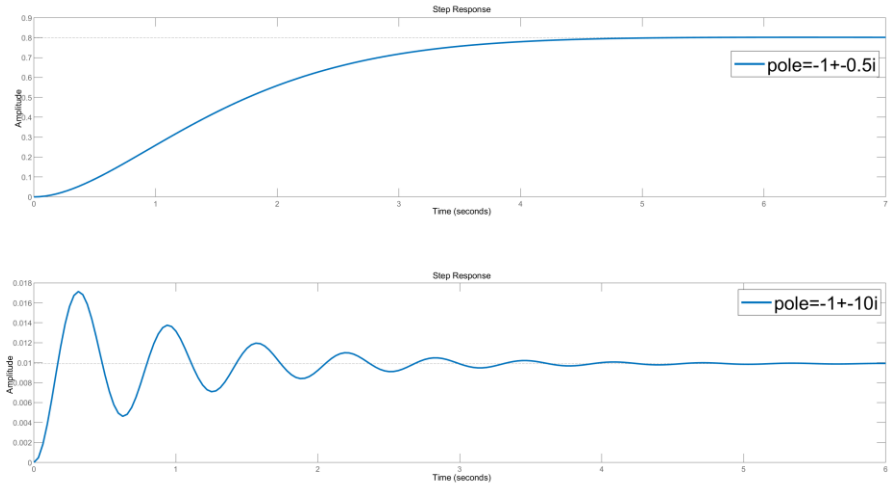

Figure 2.10 output of different pole

- dominant pole

  Set pole is $\lambda = -a$+bi$\rightarrow \frac{1}{s+a-bj} \rightarrow e^{-at}e^{bjt}$, the bigger the absolute value of the real part of poles, the faster it goes to zero, which means the response speed of system is faster, so the dynamics of the system is dominated by the slow mode, and the fast model can be ignored.

2.2 Task2: LQR method

2.2.1 Computation Feedback Gain K

In this part, a state feedback controller will be designed by LQR method. Before implementing LQR, system controllability has been checked in section 2.1.

The LQR optimal control is to find the control law that minimizes

$$J = \frac{1}{2}\int_0^\infty (x^T Q x + u^T R u)dt \qquad (2\text{-}10)$$

The optimal control law turns out to be in the form of linear state feedback:

$$u = -Kx \qquad (2\text{-}11)$$

The matrix R is positive definite, while the matrix Q is semi-positive definite. They appear most often in diagonal form. Though there is no inherent restriction to such a form..

I choose

$$Q = \begin{bmatrix} 10 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 10 \end{bmatrix} \text{ and } R = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}$$

The minimized cost is derived by taking derivative of J, such that the optimal feedback gain K follows

$$\frac{J}{dK} = 0 \qquad (2\text{-}12)$$

Since taking the derivative of J is difficult in MIMO system, we can use the concept of Lyapunov equation.

$$J = \frac{1}{2}x^T(0)Px(0) + \frac{1}{2}\int_0^\infty (x^T PB + u^T R)R^{-1}(B^T Px + Ru)dt \qquad (2\text{-}13)$$

where P is a positive definite matrix satisfying ARE:

$$A^T P + PA + Q = PBR^{-1}B^T P \qquad (2\text{-}14)$$

P can be solved by constructing a 8 by 8 matrix

$$\Gamma = \begin{bmatrix} A & -BR^{-1}B^T \\ -Q & -A^T \end{bmatrix} \qquad (2\text{-}15)$$

Then find its eigenvectors corresponding to its 4 stable eigenvalues to be

$$\begin{pmatrix} v_i \\ \mu_i \end{pmatrix} \text{ i=1,2,3,4} \qquad (2\text{-}16)$$

Finally, P is

$$P = [\mu_1 \quad \mu_2 \quad \mu_3 \quad \mu_4][v_1 \quad v_2 \quad v_3 \quad v_4]^{-1} \qquad (2\text{-}17)$$

The feedback controller K is

$$K = R^{-1}B^T P \qquad (2\text{-}18)$$

$$K = \begin{bmatrix} 0.0419 & -0.0731 & 0.1363 & -0.0615 \\ 0.0155 & -0.0390 & 0.0464 & -0.0081 \end{bmatrix}$$

2.2.2 Simulation results

The simulation model same as figure 2.2 The step responses are respectively shown in Figure 2.11 and Figure 2.13. When r=[1 0] ,overshoot of y_1 and y_2 are 0.44% and 3.82% respectively, When r=[0 1] ,overshoot of y_1 and y_2 are 0.00% and 0.00% respectively. They settling time all less than 20 seconds, so the model can satisfy performance requirement and the control signal also in an acceptable range. The state trend under non-zero initial state is shown in Figure 2.15.

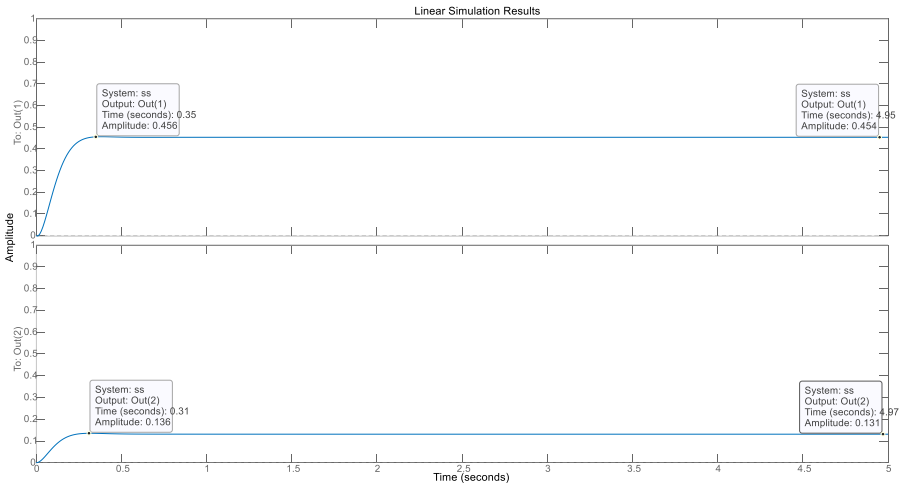

Figure 2.11 step response when r=[1 0]

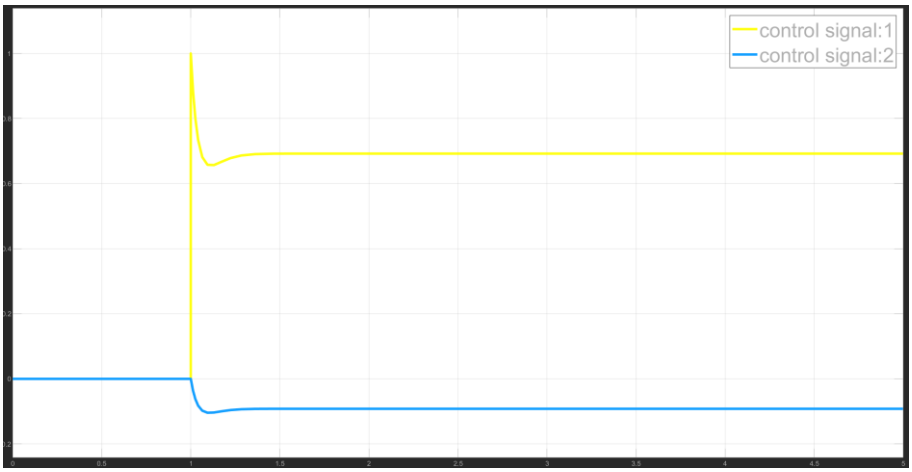

Figure 2.12 control signal when r=[1 0]

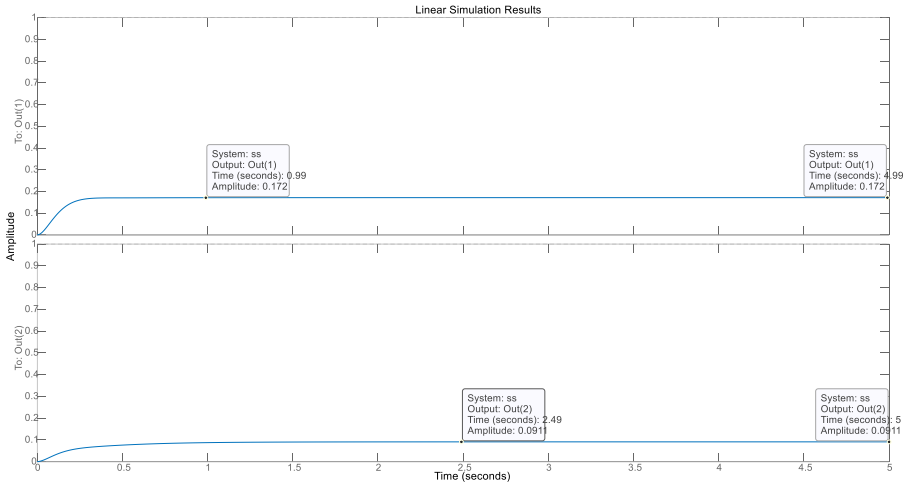

Figure 2.13 step response when r=[0 1]

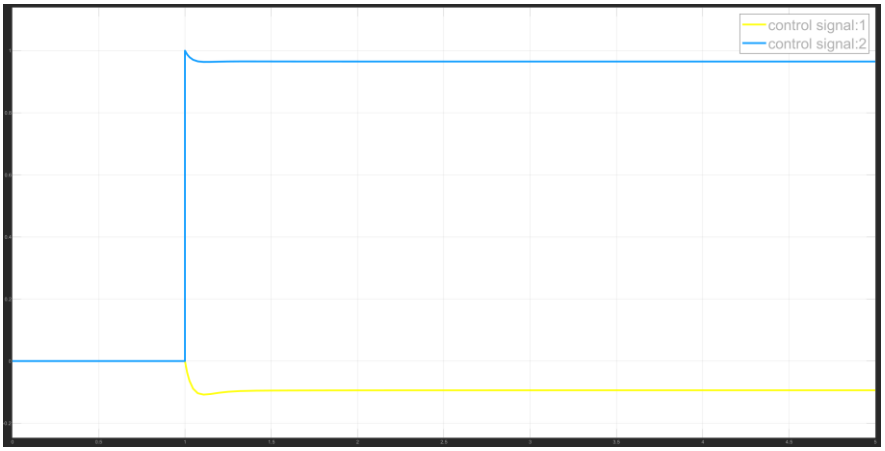

Figure 2.14 control signal when r=[0 1]

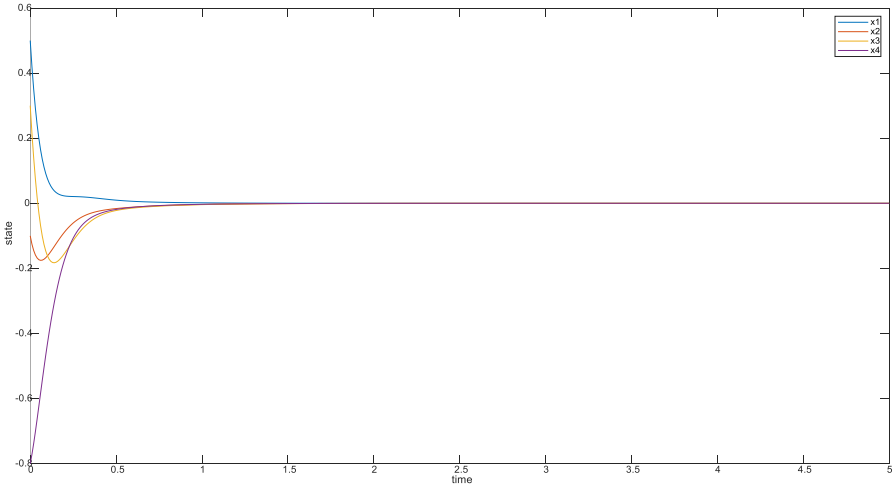

Figure 2.15 state trend under zero-input and non-zero initial state

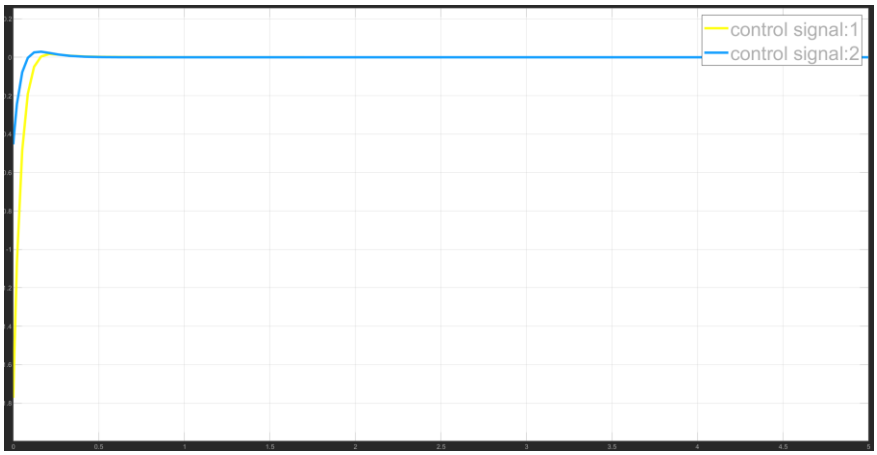

Figure 2.16 control signal under zero-input and non-zero initial state

2.2.3 effects of weightings Q and R on system performance

Q is relative weightings among state x, R is relative weightings among feedback u. the absolute value of Q and R has no effect on the feedback gain ,only the proportion of Q and R influence the computation of K. Therefore, scaling up (or down) Q and R at the same time will lead to the same K. Increasing Q/R will lead to larger elements of the gain matrix K, and the faster the state variables approach zero. Decreasing Q/R will lead to smaller elements of the gain matrix K, which leads to slower response but smaller energy cost. State trend comparison and control signal comparison between setting Q=I R=I and Q=100I R=I is shown in following figures. Increasing Q/R results in faster the state variables approach zero but larger energy cost, vice versa.

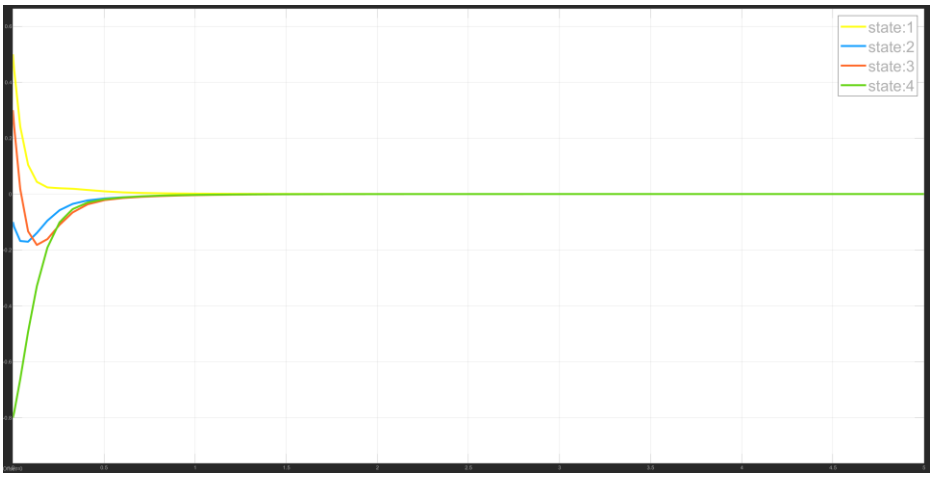

Figure 2.17 state trend under zero-input and non-zero initial state when Q=I

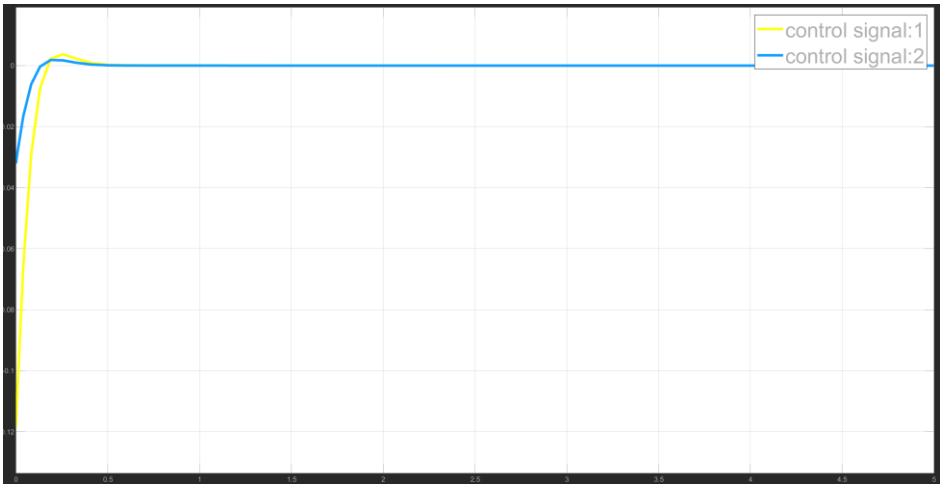

Figure 2.18 control signal under zero-input and non-zero initial state when Q=I

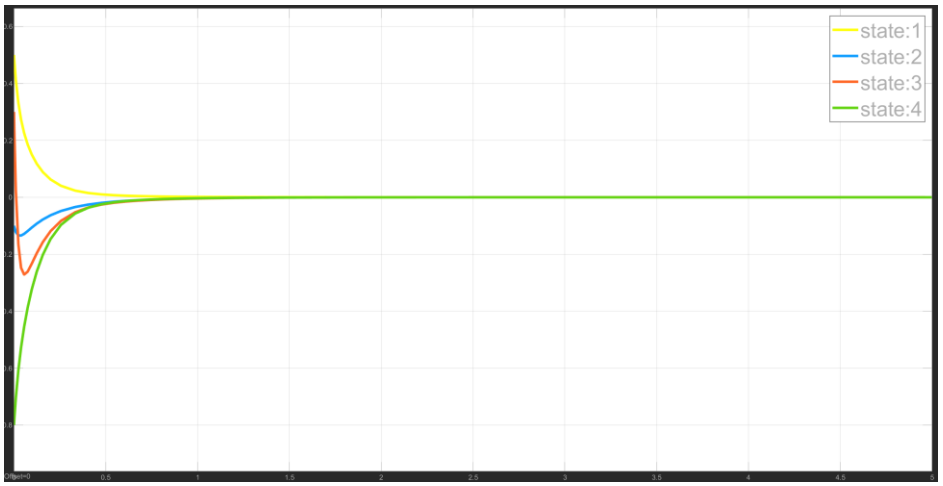

Figure 2.19 state trend under zero-input and non-zero initial state when Q=100I

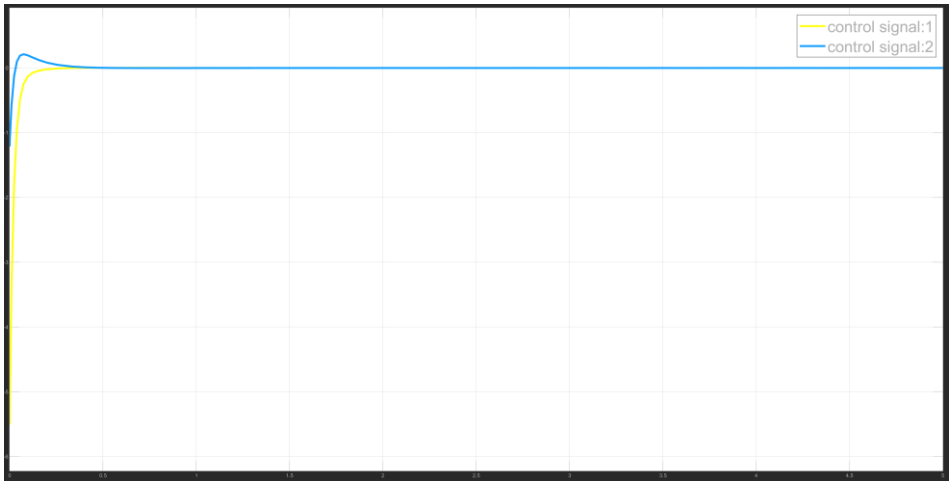

Figure 2.20 control signal under zero-input and non-zero initial state when Q=100I

2.3 Task3: State Observer

2.3.1 Full-state observer

The Full-state observer model **is**

$$\dot{\hat{x}}(t) = A\hat{x}(t) + Bu(t) + L\left[y - \hat{y}\right] \qquad (2\text{-}19)$$

$$\hat{y} = C\hat{x} \qquad (2\text{-}20)$$

The estimate error is $\tilde{x} = x - \hat{x}$, Substituting observer model into estimate error can get $\dot{\tilde{x}} = (A - LC)\tilde{x}$, The purpose of the observer is to make the error approach 0 when time approaches infinity. To achieve this goal, we need to design L to make $(A - LC)$ stable. Because $det[sI - (A - LC)] = det\left[sI - \left(\tilde{A} - \tilde{B}\tilde{K}\right)\right]$,where $\tilde{A} = A^T, \tilde{B} = C^T, \tilde{K} = L^T$,L can be calculated using either pole placement method or LQR method.

2.3.1.1 pole placement method

First ,we should check the observability of the system,

$$W_o = \begin{bmatrix} C \\ CA \\ CA^2 \\ CA^3 \end{bmatrix} \qquad (2\text{-}21)$$

rank(W_o)=4,then we can use the same procedure as 2.1 the pole place method, just change the A to A^T,B to C^T.

The new controllability matrix Wc .is

$$W_c = [C^T \; A^T C^T \; A^{T^2} C^T \; A^{T^3} C^T] \qquad (2\text{-}22)$$

the first four vectors are independent with each other, group them together as C in this sequence:

$$C = \{c_1 \;\; A^T c_2 \;\; c_2 \;\; A^T c_2 \} \qquad (2\text{-}23)$$

$$C^{-1} = \begin{bmatrix} q_1^T \\ q_2^T \\ q_3^T \\ q_4^T \end{bmatrix} \qquad (2\text{-}24)$$

$d_1 = 2, d_2 = 2$ so T is

$$T = \begin{bmatrix} q_2^T \\ q_2^T A^T \\ q_4^T \\ q_4^T A^T \end{bmatrix} \qquad (2\text{-}25)$$

Then we can get $\bar{A} = TA^T T^{-1}$,$\bar{C} = TC^T$

Set $\overline{L}^T = \begin{bmatrix} l_{11} & l_{12} & l_{13} & l_{14} \\ l_{21} & l_{22} & l_{23} & l_{24} \end{bmatrix}$,

Because the dominate pole of the controller is -2.9085, choose observer poles as -12 ,-11.6,-10 ,-13,they are 3-5 times faster than the closed loop poles designed by the controller.$\overline{L}^T$ can be obtained by solving equation $A_d = \overline{A} - \overline{C}\overline{L}^T$ where A_d is desired closed-loop matrix calculate from observer poles, then transform $\overline{L}^T$ into unit coordinate and transpose it, we can get the real L

$$L = (\overline{L}^T T)^T = 1.0e{+}03 * \begin{bmatrix} 0.0073 & -1.8561 \\ 0.0034 & -0.2013 \\ 0.0112 & -3.3125 \\ -0.0013 & 2.4537 \end{bmatrix}$$

2.3.1.2 LQR method

We also can use the LQR method to design L. Different from the pole placement, LQR can have a better energy result, It is also necessary to make the observer poles 3-5times faster than the closed loop poles designed by the controller. Choose Q=3.4I,K=0.5I,use the same procedure as 2.2, just change the A to A^T,B to C^T,finally we can get

$$L = \begin{bmatrix} -1.5915 & 0.2397 \\ 0.3946 & -3.7171 \\ 0.4702 & 4.8706 \\ 2.0189 & -2.9128 \end{bmatrix}$$

The poles of the observer is -13.7257 +11.4643i, -13.7257 -11.4643i, -12.4370, -9.1211 they 3-5 times faster than the closed loop poles, so the poles we designed for observer satisfied the

requirement.

2.3.2 Simulation results

The full-state Observer diagram is shown in Figure2.21,where the state feedback gain K1 is same as 2.2.As we can see from below pictures , the state errors of the two observers can all converge to 0.The step response also satisfy performance requirement $t_s < 20$ and $M_p < 0.1$.

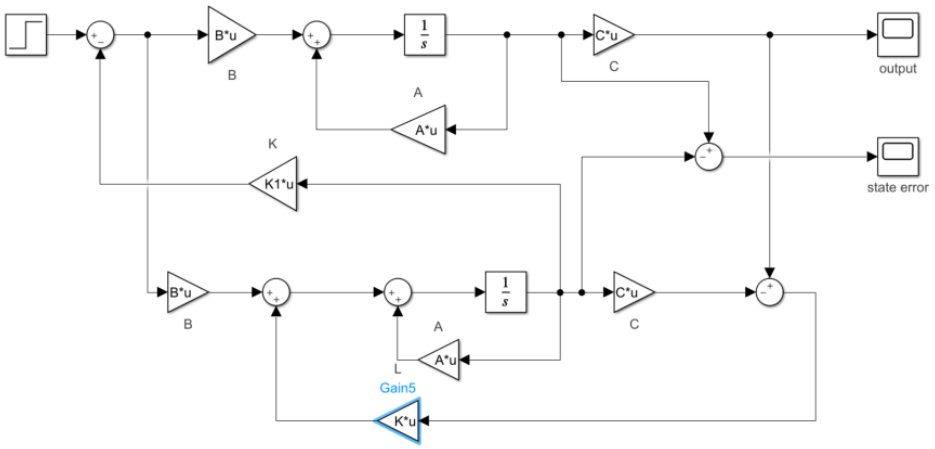

Figure 2.21 Full-state Observer diagram

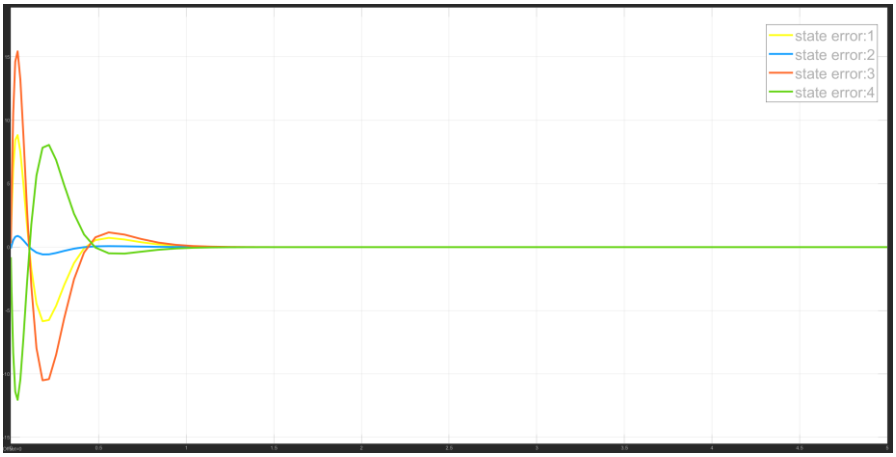

Figure 2.22 state error of Full-state Observer design by pole placement method

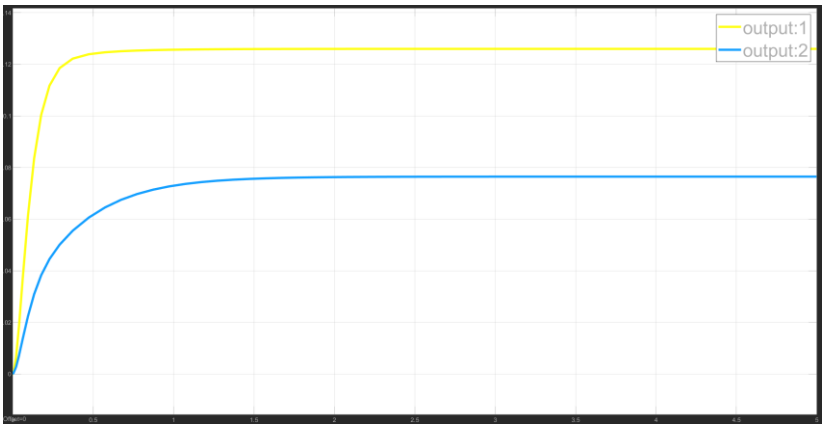

Figure 2.23 step response when r=[0 1] with Full-state Observer design by pole placement method

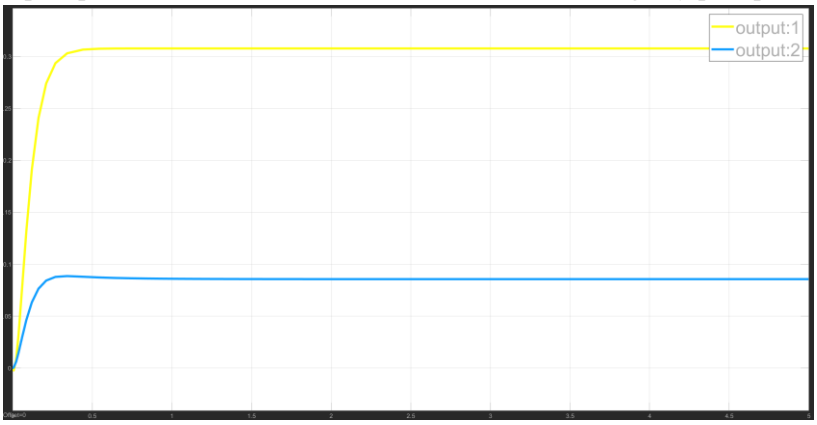

Figure2.24 step response when r=[1 0] with Full-state Observer design by pole placement method

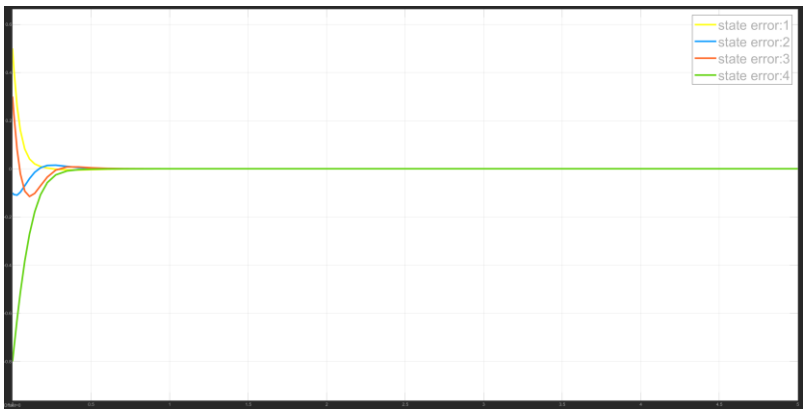

Figure2.25 state error of Full-state Observer design by LQR method

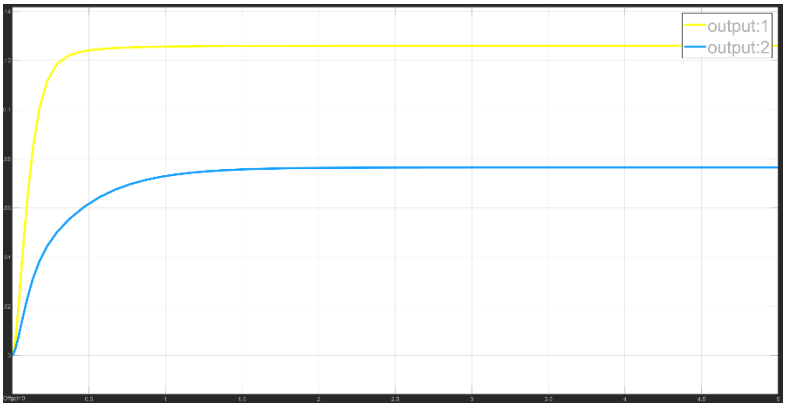

Figure2.26 step response when r=[0 1] with Full-state Observer design by LQR method

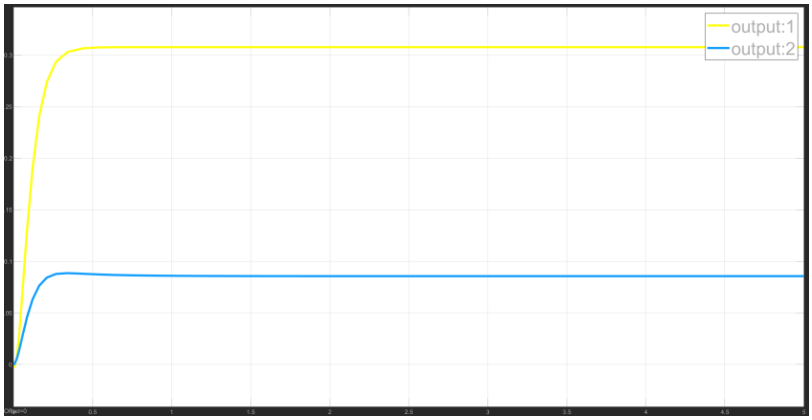

Figure2.27 step response when r=[1 0] with Full-state Observer design by LQR method

2.3.3 effects of observer poles

The rate of estimation error convergence to zero will be controlled by how negative the real parts of the pole of the observer are. The larger the negative real part of the observer pole, the faster the state error will converge to 0. As figure 2.5 show, when all observer poles=-14,the state error converge to 0 at t=1s, but when all observer poles=-9,the state error converge to 0 at t=1.5s But If the observation y has been corrupted by noise, say $y = cx + d$, then the error equation becomes $\dot{\tilde{x}} = (A - LC)\tilde{x} - Ld$ Therefore, a large L will lead to large estimation error if d is persistent. So the choice of observer poles is a tradeoff between speed and constraints imposed by noise, saturation and nonlinearity. In practice we can use The Kalman filter where L is designed to be time-varying to design the observer.

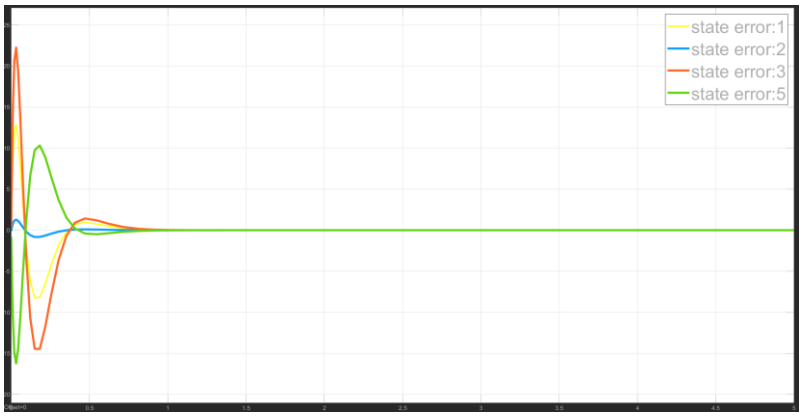

Figure2.28 state error when all observer poles=-14

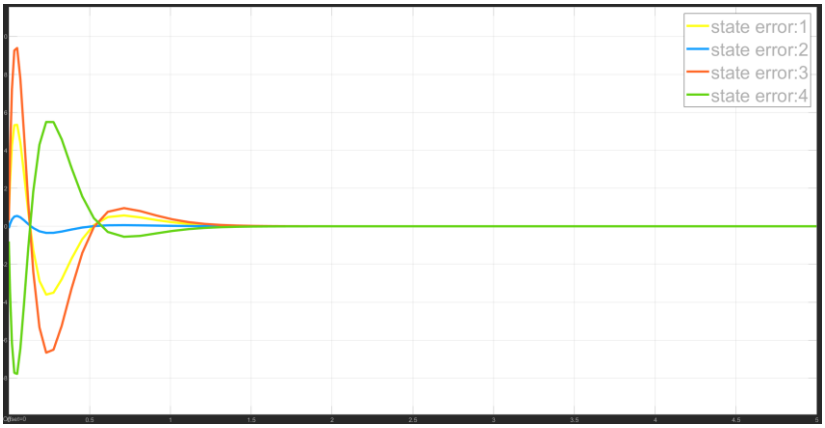

Figure2.29 state error when all observer poles=-9

2.4 Task4: decoupling controller

2.4.1 decoupling control by state feedback

For the MIMO system in this project, the transfer function matrix is

$$Y(s) = C(sI - A)^{-1}BU(s) = G(s)U(s) \qquad (2\text{-}26)$$

Where G is supposed to be an 2 by 2 square matrix,if $g_{ii} \neq 0$ and $g_{ij} = 0$ for all $i \neq j$, the system is called decoupled. In this part,we will use state feedback $u = -Kx + Fr$ to design a decoupling controller.

First , check the relative degree use

$$\sigma_i = \begin{cases} \min\,(j | c_i^T A^{j-1} B \neq 0^T, j = 1,2\cdots,n) \\ n, if\ c_i^T A^{j-1} B = 0^T, j = 1,2\cdots,n \end{cases} \qquad (2\text{-}27)$$

The relative degree of the system is $\sigma_1 = 1,\ \sigma_2 = 1,$

Then the whole transfer function G can be written as

$$G(s) = \begin{pmatrix} \frac{1}{s} & 0 \\ 0 & \frac{1}{s} \end{pmatrix} [B^* + C^*(sI - A)^{-1}B] \qquad (2\text{-}28)$$

Where $B^* = \begin{bmatrix} c_1^T A^0 B \\ c_2^T A^0 B \end{bmatrix}, C^* = \begin{bmatrix} c_1^T A \\ c_2^T A \end{bmatrix}$, finally F and K can be calculate through

$$F = (B^*)^{-1} = \begin{bmatrix} -1.2927 & -0.6561 \\ -1.2255 & 31.6134 \end{bmatrix},$$

$$K = (B^*)^{-1} \begin{bmatrix} c_1^T \phi_{f1}(A) \\ c_2^T \phi_{f2}(A) \end{bmatrix} = \begin{bmatrix} -48.6480 & 4.7459 & -33.7167 & 11.1625 \\ 200.0480 & -107.3454 & 235.4583 & 99.8244 \end{bmatrix}$$

$\phi_{f1}(A) = A + \gamma_{11} I, \phi_{f2}(A) = A + \gamma_{21} I$ are the desired characteristic polynomial, I choose $\gamma_{11} =$

$\gamma_{21} = 0.64$,

Check the closed-loop transfer function matrix when s=0, We can find that it is a diagonal matrix so the system is successfully decoupled.

2.4.2 decoupling control by output feedback

In this part, we will use output feedback to design a decoupling controller. The transfer function of this system is

$$H(s) = [I + G(s)K(s)]^{-1}G(s)K(s) \quad (2\text{-}29)$$

$$H^{-1} = (GK)^{-1}[I + GK] = (GK)^{-1} + I \quad (2\text{-}30)$$

If GK is decoupled, H will also be decoupled. Now our aim become design a K to decouple GK without unstable pole-zero cancellation. At this time, the decoupled loop G(s)K(s) can be stabilized loop by loop, so that the resultant system is decoupled and stable.

First of all ,we should design $K_d(s)$ to make G(s) $K_d(s)$ diagonal and nonsingular. There are many ways to design $K_d(s)$, we choose $K_d(s) = \text{adj}\,(N(s))$.so that $G(s)K_d(s) = \frac{N(s)\cdot\text{adj}\,(N(s))}{d(s)} = \frac{\det(N(s))}{d(s)} I_m$ is decoupled.However,in my case, $K_d(s)$ is not proper,so we should design a $K_s(s)$ to make $K(s) = K_d(s)K_s(s)$ is proper and assure that there is no unstable pole-zero cancellation between G(s) and K(s).here We use the reciprocal of the greatest common divisor of the $G(s)K_d(s)$ to form $K_s(s)$.The final K is

$$K = K_d(s)K_S(s) = \begin{bmatrix} 4.7821 & -9.1105 \\ -7.0271 & 24.1943 \end{bmatrix}$$

Check the closed-loop transfer function matrix when s=0, We can find that it is a diagonal matrix so the system is successfully decoupled.

2.4.3 Simulation results

The decoupling controller with state feedback diagram is shown in Figure 2.30

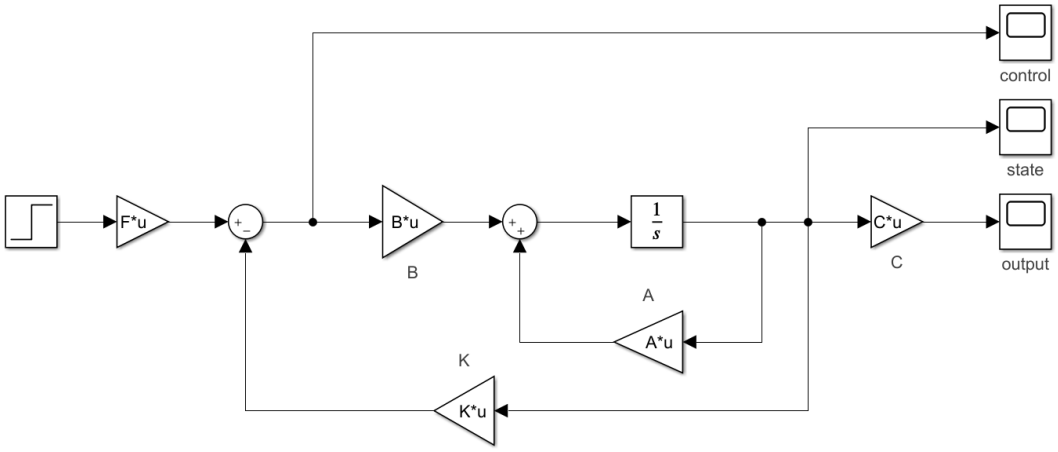

Figure 2.30 decoupling controller with state feedback diagram

As shown below ,the system with state feedback is neither BIBO stable or internal stable in the sense of Lyapunov , because the inputs, outputs and all the state variables are not bounded when time goes to infinity, By checking the poles of the system, we can find that the system has a pole with a positive real part . This pole that cannot be placed by decoupling controller. Because the relative degree is $\sigma 1=1$, $\sigma 2 = 1$,so the desired characteristic polynomial is $\phi_{fi}(A) = A + \gamma_{i1} I$, This means we can only place two poles and the unstable pole can not be moved. Therefore, there does not hold a stable decoupling solution for this system, the step response also can not satisfy performance requirement.

Figure 2.31 output step response when r=[1 0]

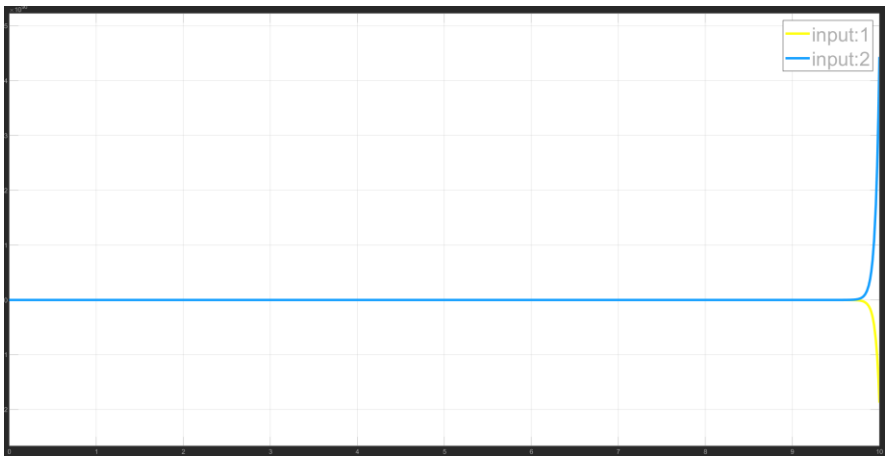

Figure 2.32 input step response when r=[1 0]

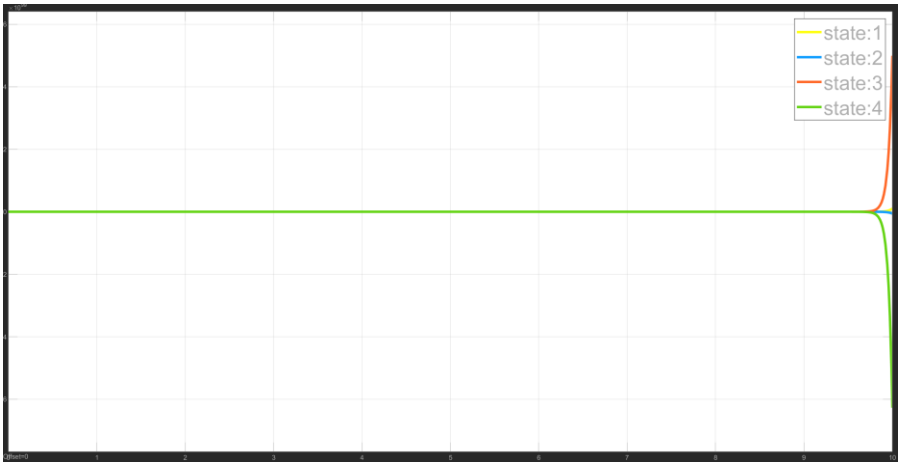

Figure 2.33 state trend under zero-input and non-zero initial state

The decoupling controller with output feedback diagram is shown in Figure 2.34

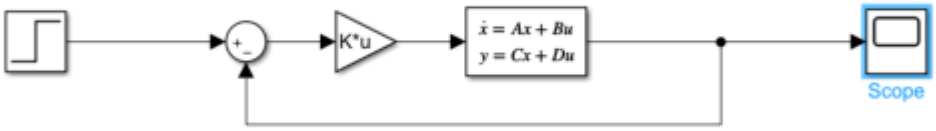

Figure 2.34 decoupling controller with output feedback diagram

As shown in the figures below, the decoupling controller with output feedback can not only decouple the system but also stabilize the output, and meet the system requirements.

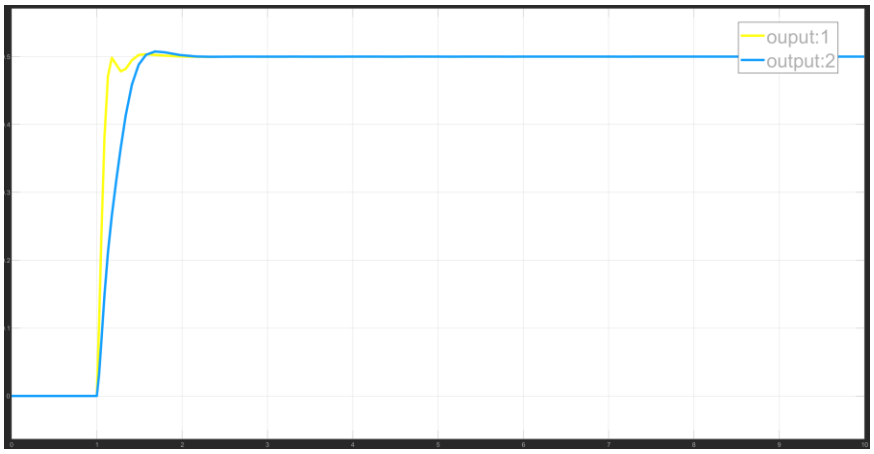

Figure 2.35 output step response when r=[1 1]

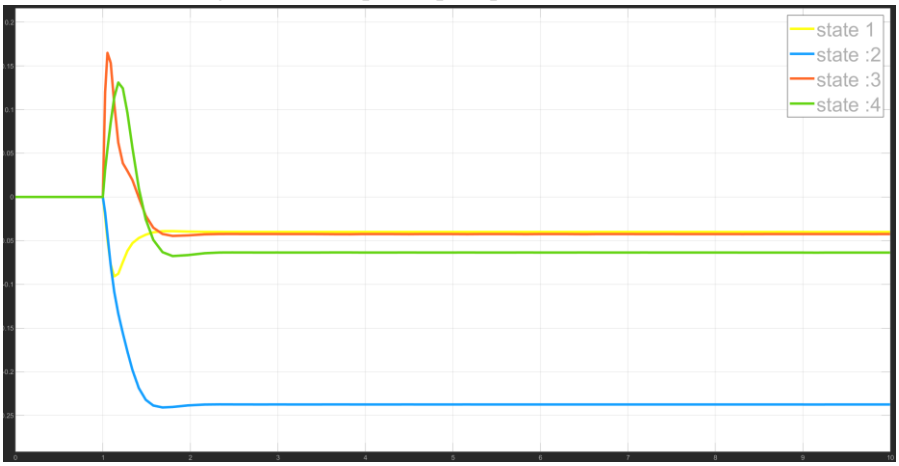

Figure 2.36 state trend when r=[1 1]

2.5 Task 5 Servo Control

2.5.1 feedback and observer design

A MIMO system with disturbance can be written as

$$\dot{x} = Ax + Bu + B_w w \qquad (2\text{-}29)$$

$$y = Cx$$

The error vector is defined as

$$e = r - y \qquad (2\text{-}30)$$

If the disturbance w and reference r are both of step type then we can use the following integral control to cancel out unstable factor in input and disturbance signal then achieve zero steady state error for all the outputs .

$$v(t) = \int_0^t e(t)\, dt \qquad (2\text{-}31)$$

$$\dot{v}(t) = e(t) = r - y(t) = r - Cx(t) \qquad (2\text{-}32)$$

by putting x and v together we can form a new dynamic system,

$$\begin{pmatrix} \dot{x} \\ \dot{v} \end{pmatrix} = \begin{pmatrix} A & 0 \\ -C & 0 \end{pmatrix} \begin{pmatrix} x \\ v \end{pmatrix} + \begin{pmatrix} B \\ 0 \end{pmatrix} u + \begin{pmatrix} B_w \\ 0 \end{pmatrix} w + \begin{pmatrix} 0 \\ I \end{pmatrix} r \qquad (2\text{-}33)$$

$$\dot{\overline{x}} = \overline{A}\overline{x} + \overline{B}u + \overline{B}_W W + \overline{B}_r r \qquad (2\text{-}34)$$

$$y = [C \quad 0] \begin{bmatrix} x \\ v \end{bmatrix} = \overline{C}\overline{x} \qquad (2\text{-}35)$$

In order to design a controller, first ,we need to check the controllability of the system

$rank \begin{pmatrix} A & B \\ -C & 0 \end{pmatrix}$ = 6=n+m and the plant is controllable, so the new system is controllable, it can be stabilized by state feedback control law:

$$u = -K\overline{x} = -[K_1 \quad K_2] \begin{bmatrix} x \\ v \end{bmatrix} \qquad (2\text{-}36)$$

Where $K1$ is the feedback control gain which is responsible for stabilization, $K2$ is the integral gain in charge of set point tracking. Once the feedback system is stable, each signal in the system in response to step inputs will be constant in the steady state, and so $e(t) = \dot{v}(t) = 0$,as $t \to \infty$.

The feedback gain K can be determined by pole placement procedure or LQR .here we choose LQR method.

$$\text{Setting} Q = \begin{bmatrix} 10 & 0 & 0 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 & 0 & 0 \\ 0 & 0 & 10 & 0 & 0 & 0 \\ 0 & 0 & 0 & 10 & 0 & 0 \\ 0 & 0 & 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 0 & 0 & 10 \end{bmatrix} \text{ and } R = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}$$

the gain will be $K = \begin{bmatrix} -0.8047 & -1.5312 & 2.3718 & -1.1707 & -4.4535 & 0.4078 \\ 1.4368 & -4.6048 & 1.0475 & 0.8164 & -0.4078 & -4.4535 \end{bmatrix}$

The first 4 columns are K_1,5-6 columns are K_2.Because we only can measure two output so we need also design an observer to measure 4 state. Using the same method in 2.3 to design a full-state observer, we can get

$$L = 1.0e{+}03 * \begin{bmatrix} 0.0073 & -1.8561 \\ 0.0034 & -0.2013 \\ 0.0112 & -3.3125 \\ -0.0013 & 2.4537 \end{bmatrix}$$

2.5.2 Simulation results

The Simulink model is :

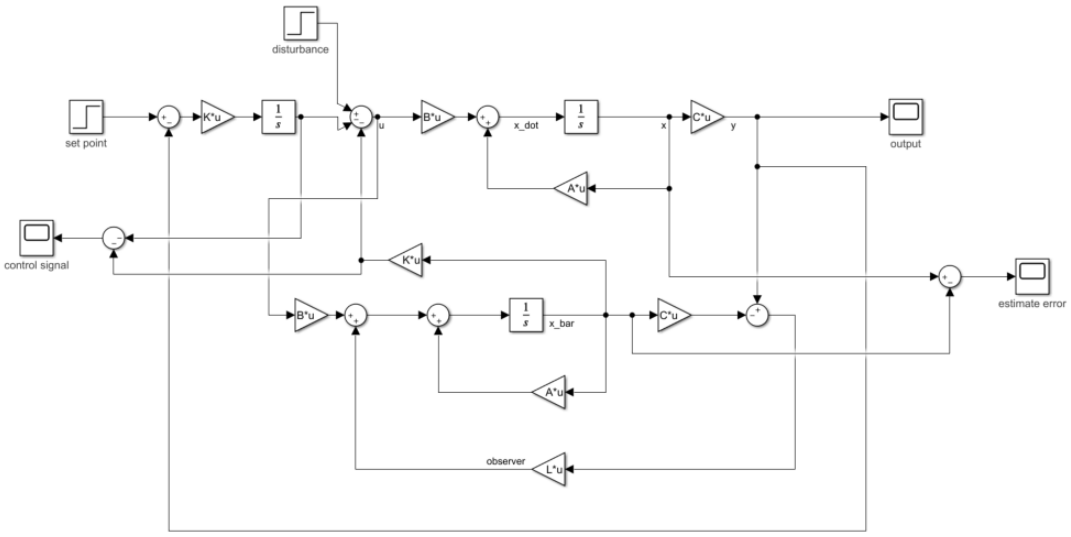

Figure 2.37 Servo Control diagram

When there is no disturbance, we can get the result as show in below figure, the output can converge to set point when t=30s

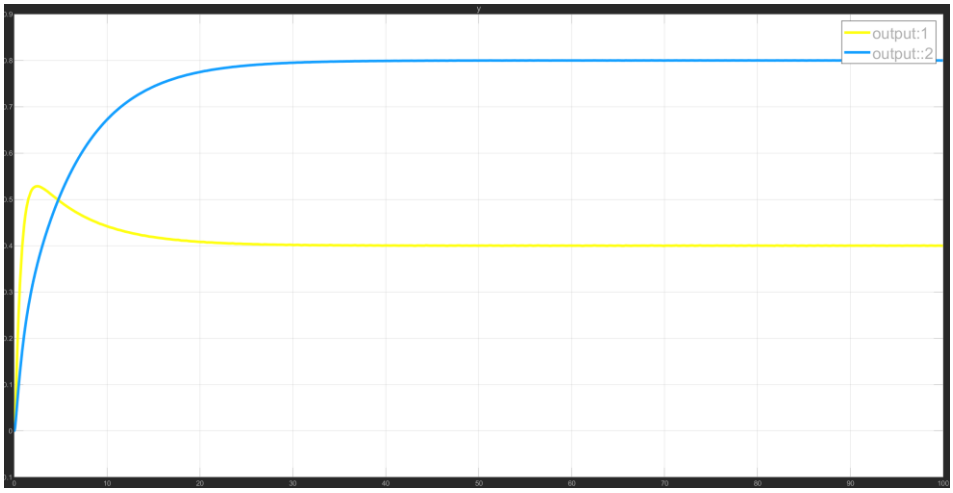

Figure 2.38 output without disturbance

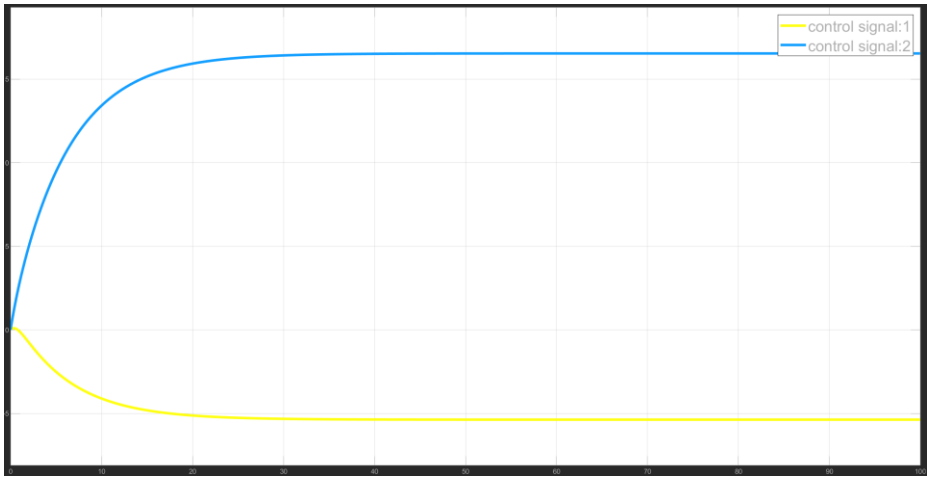

Figure 2.36 control signal without disturbance

When we add a disturbance in t=10s, the output will also converge to set point when t=30s, But compared with no disturbance, there will be a fluctuation in the control signal and output at 10s, but this fluctuation will be quickly eliminated

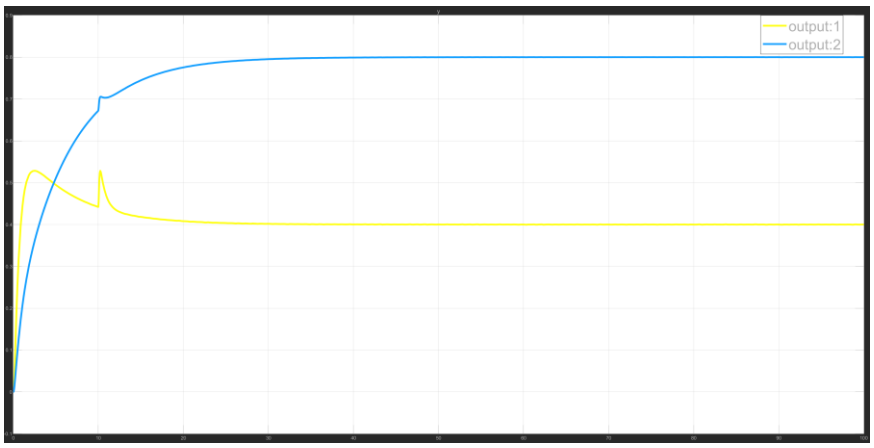

Figure 2.39 output with disturbance

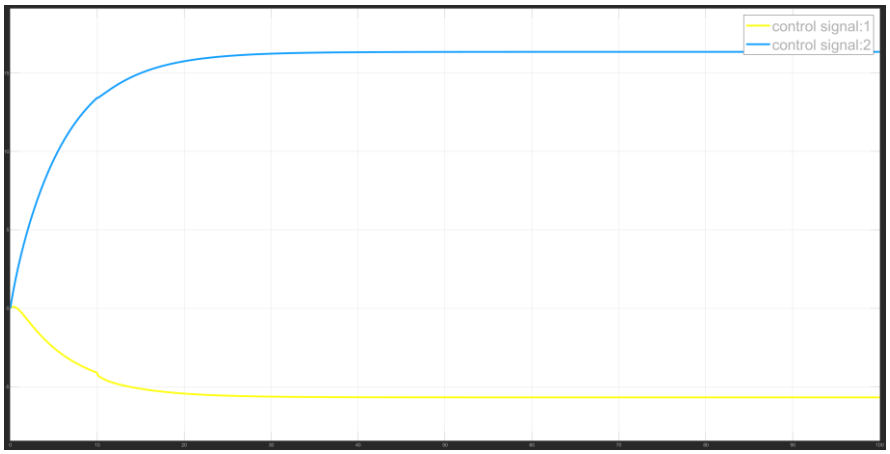

Figure 2.40 control signal with disturbance

Finally check the step response of the system we find it satisfy the performance requirement $t_s < 20$ and $M_p < 0.1$.

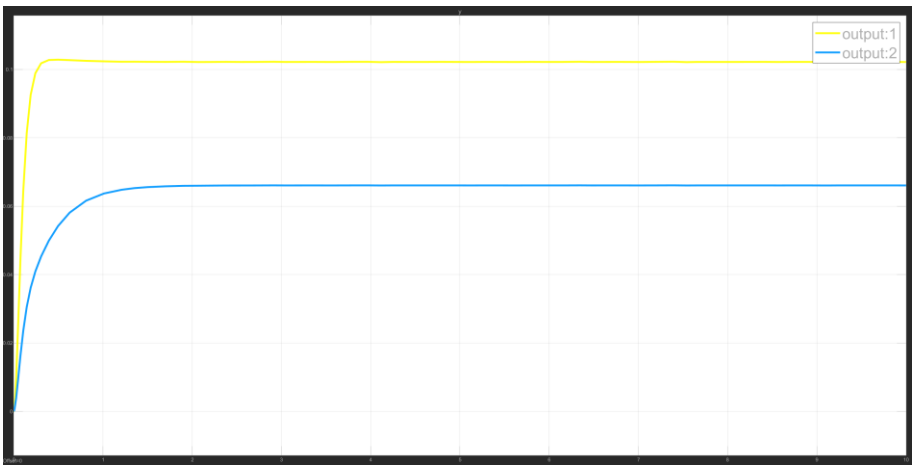

Figure 2.41 step response when r=[0 1]

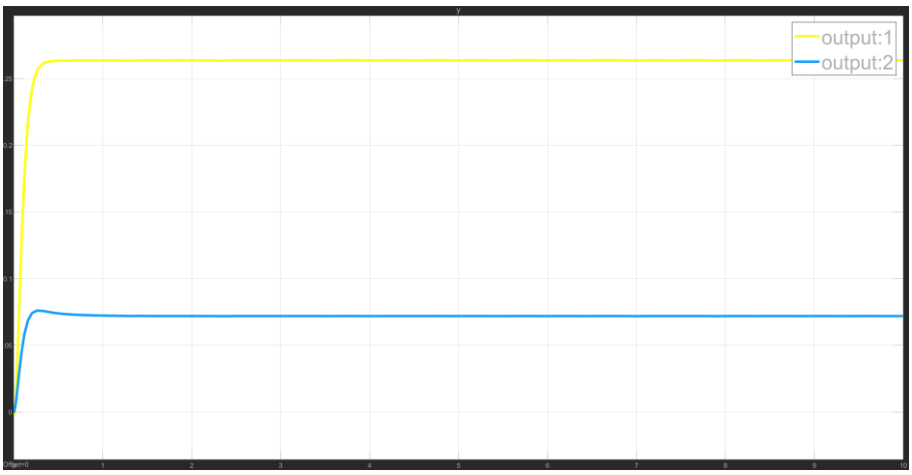

Figure 2.42 step response when r=[1 0]

2.6 Task 6

（1）because $\dot{x} = Ax + Bu$ though Laplace transform we have

$$sx(s)=Ax(s)+Bu(s) \qquad (2\text{-}37)$$

$$x(s) = (sI - A)^{-1}Bu(s) \qquad (2\text{-}38)$$

However, u is a 2 by 1 vector and x is a 4 by 1 vector. When $t \to \infty, s \to 0$, $x_{sp}(s) = (sI - A)^{-1}Bu(s)$ will have 2 unknown variables but 4 equations, so equations have no solution. Therefore, if we keep the original system unchanged, we cannot maintain the states x around a given set point at steady state. But if we expand u to a 4 by 1 vector by add a new input u_{new} and design a B_{new} ,the system will become

$$x(s) = (sI - A)^{-1}[B \quad B_{new}]\begin{bmatrix} u \\ u_{new} \end{bmatrix} \qquad (2\text{-}39)$$

At this time, we will have 4 unknown variables and 4 equations, we can find a definite solution.

Here I choose $B_{new} = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 1 \\ 1 & 1 \end{bmatrix}$,the input will be $u = \begin{bmatrix} -6.9719 \\ 15.9472 \\ -1.5218 \\ -6.6886 \end{bmatrix}$,as figure show ,the states x can maintain around $x_{sp} = [0,0.5,-0.4,0.3]^T$ at steady state

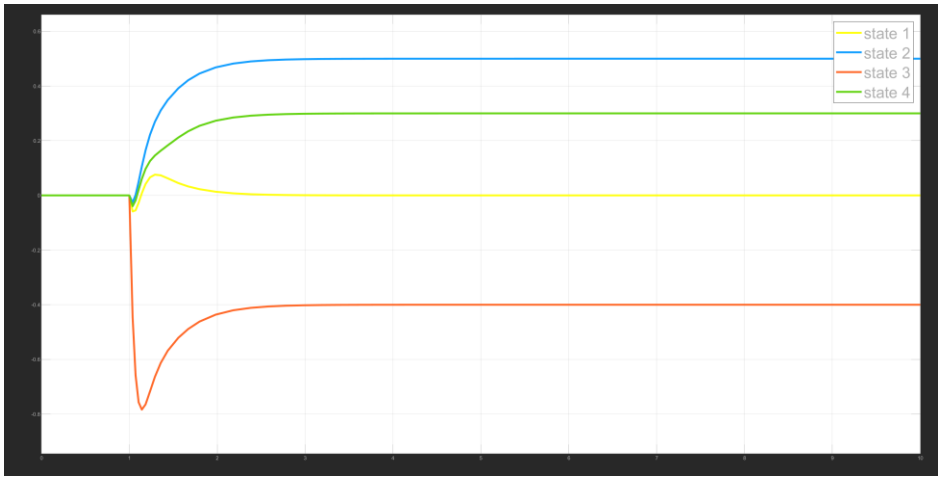

Figure 2.43 state with new input

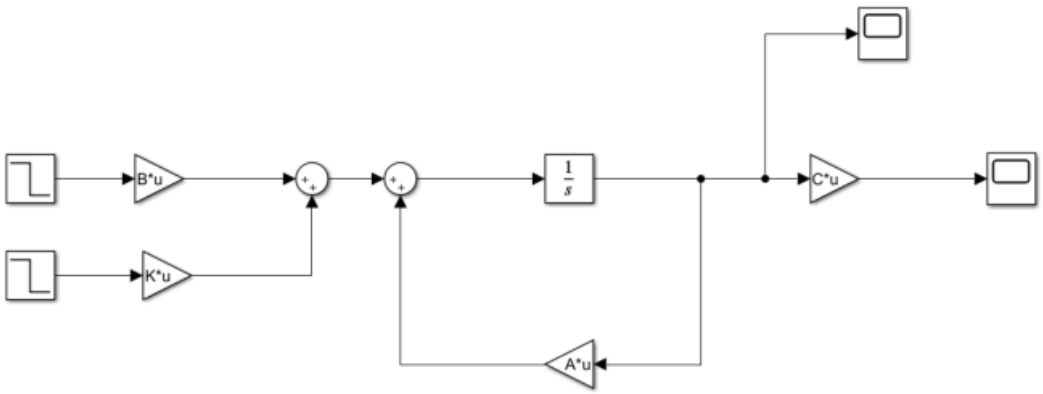

Figure 2.44 new input diagram

(2)in my case $W = \begin{bmatrix} 6 & 0 & 0 & 0 \\ 0 & 5 & 0 & 0 \\ 0 & 0 & 9 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$,same as last question, through Laplace transform we have $x_s(s) = (sI - A)^{-1}B\mathrm{u}(s)$ when $t \to \infty, s \to 0$,x will at steady state .At this time

$$J(x_s) = \frac{1}{2}(x_s - x_{sp})^T W(x_s - x_{sp}) \qquad (2\text{-}40)$$

will have two unknown variables $u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$.In order to minimize J, we need differentiate J with respect to u and make the derivative equal to 0,namely

$$\frac{dJ}{du_1} = \frac{dJ}{du_2} = 0 \qquad (2\text{-}41)$$

The solutions are u_1=-0.4356 , u_2=-7.6480,at this time J= 0.3504.Input solutions into system ,we can get result as show in figure 2.42 ,where $x_1 = 0.2805, x_2 = 0.3941, x_3 = 0.-2683, x_4 = 0.4277$, The state cannot be accurately maintained at the set point, it can only close enough to the set point.

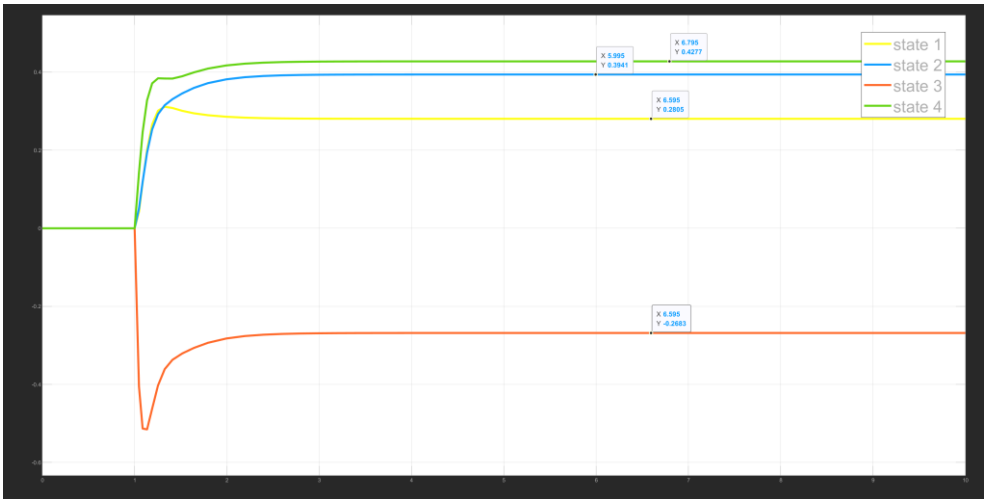

Figure 2.45 state with new input

3.conclusion

In this project, we model the engine air system as a MIMO linear system and try to design some control methods to stabilize the system or track the reference signal. We use pole placement, LQR , server control to stabilize system. Using state observer and decoupling to analyze the system more realistically. The results of simulink show the effectiveness of the controller.

Reference

[1]Luenberger, D. G. (1964). Observing the state of a linear system. *IEEE transactions on military electronics*, *8*(2), 74-80.

[2] Callier, F. M., & Desoer, C. A. (2012). *Linear system theory*. Springer Science & Business Media.

[3] Zadeh, L., & Desoer, C. (2008). *Linear system theory: the state space approach*. Courier Dover Publications.

Appendix

Matrix

```
A=[-8.6487 -0.0399 -4.7500 3.5846;
      -4.5740 0.9619 -4.3662 -0.9683;
      3.7698 14.5212 -17.5853 4.4936;
      -8.6645 8.3742 -4.4331 -11.1484];
B=[0.2786 0.0319;
      0.0138 -0.02;
      4.4939 1.4986;
      -1.4269 -0.2730];
C=[-3.2988 -1.5532 0.0370 -0.0109;
    0.2522 -2.1506 -0.0104 0.0163];
x0=[0.5;-0.1;0.3;-0.8]
D=[0 0;0 0]
```

Task1

```
clc;
clear;
load('matrix.mat');
%% reference model
syms s;
ts=20;%settling time
mp=0.1;%overshoot
pos_damp=log(1/mp)/sqrt(pi^2+(log(1/mp))^2);
pos=4/(ts);
damp=0.8;
wn=0.4;
lamda1=-damp*wn+wn*sqrt(1-damp^2)*i;
lamda2=-damp*wn-wn*sqrt(1-damp^2)*i;
lamda3=-1.5;
lamda4=-1.28;
polynomial=(s-lamda1)*(s-lamda2)*(s-lamda3)*(s-lamda4);
Ad_cof=double(coeffs(polynomial));
step(tf([1],fliplr([Ad_cof])));
%% CCF
syms k11 k12 k13 k14 k21 k22 k23 k24 ;
K=[k11 k12 k13 k14; k21 k22 k23 k24];
W=[B A*B A^2*B A^3*B];
assert(rank(W(:,1:4))==4);
CC=W(:,1:4);
CC=[CC(:,1),CC(:,3),CC(:,2),CC(:,4)];
inv_C=inv(CC);
```

```
d1=2;
d2=2;
T=[inv_C(d1,:);
       inv_C(d1,:)*A;
       inv_C(d1+d2,:);
       inv_C(d1+d2,:)*A];
Aba=T*A/(T);
Bba=T*B;
Aba(abs(Aba)<10^(-10))=0;
Bba(abs(Bba)<10^(-10))=0;
Am=Aba-Bba*K;
Ad=[ 0    1    0    0 ;
        0    0    1    0 ;
        0    0    0    1 ;
        -Ad_cof(1:4)];
 %% solve equetion
equation=Am==Ad;
K_num=solve(equation);
K_ans=struct2array(K_num);
K_ans=double(K_ans);
Kba=[K_ans(1:4);
       K_ans(5:8)];
K1=Kba*T;
```

Task2

```
clc;
clear;
load('matrix.mat');
%% lqr
Q=[1 0 0 0
     0 1 0 0
     0 0 1 0
     0 0 0 1 ]*100;
R=[1 0
     0 1];
gamma=[A -B/R*B'
       -Q -A'];
[vector,value]=eig(gamma);
value=sum(value);
v=vector(:,find(real(value)<0));
P=v(5:8,:)/v(1:4,:);
K1=real(inv(R)*B'*P);
```

Task 3

```
clc;
clear;
load('matrix.mat');
%% LQR
syms s
Q=[1 0 0 0
    0 1 0 0
    0 0 1 0
    0 0 0 1 ].*10;
R=[1 0
    0 1].*0.5;
gamma=[A -B/R*B'
      -Q -A'];
[vector,value]=eig(gamma);
value=sum(value);
v=vector(:,find(real(value)<0));
P=v(5:8,:)/v(1:4,:);
K1=inv(R)*B'*P;
K1=real(K1)
Af=A-B*K1;
ss=ss(Af,B,C,D);
orig_pole=pole(ss);
dir_pole=[-9,-9,-9,-9];
polynomial=(s-dir_pole(1))*(s-dir_pole(2))*(s-dir_pole(3))*(s-dir_pole(4));
Ad_cof=double(coeffs(polynomial));
%% full order Observer LQR method
Q1=[1 0 0 0
    0 1 0 0
    0 0 1 0
    0 0 0 1 ].*3.4;
R1=[1 0
      0 1 ]*0.5;

phi1 = [A' -C'/R1*C;
                -Q1 -A];
[vector,value]=eig(phi1);
value=sum(value);
v=real(vector(:,find(real(value)<0)));
P=v(5:8,:)/v(1:4,:);
L1=inv(R1)*C*P;
L1=L1';
L1=real(L1)
A1=A-L1*C;
[vector,value]=eig(A1)
```

```
%% full order pole placement
syms l11 l12 l13 l14 l21 l22 l23 l24    ;
L=[l11 l12 l13 l14 ;l21 l22 l23 l24];
W=[C' A'*C' A'^2*C' A'^3*C'];
assert(rank(W(:,1:4))==4);
CC=W(:,1:4);
CC=[CC(:,1),CC(:,3),CC(:,2),CC(:,4)];
inv_C=inv(CC);
d1=2;
d2=2;
T=[inv_C(d1,:);
      inv_C(d1,:)*A';
      inv_C(d1+d2,:);
      inv_C(d1+d2,:)*A'];
Aba=T*A'/(T);
Bba=T*C';
Aba(abs(Aba)<10^(-10))=0;
Bba(abs(Bba)<10^(-10))=0;
Am=Aba-Bba*L;
Ad=[ 0    1    0    0 ;
       0    0    1    0 ;
       0    0    0    1 ;
      -Ad_cof(1:4) ];
 %% solve equetion
equation=Am==Ad;
L_num=solve(equation);
L_ans=struct2array(L_num);
L_ans=double(L_ans);
Lba=[L_ans(1:4);
      L_ans(5:8)];
L2=Lba*T;
L2=L2'
L2=real(L2)
```

Task 4

```
clc;
clear;
load('matrix.mat');
%%
syms s
for i=1:4
      if C(1,:)*A^(i-1)*B~=0
            degree1=i;
            break
```

```
        end
    end
    for i=1:4
        if C(2,:)*A^(i-1)*B~=0
            degree2=i;
            break
        end
    end
    Bstar=[C(1,:)*A^(degree1-1)*B
            C(2,:)*A^(degree2-1)*B];
    F=inv(Bstar);
    damp=0.8
    wn=0.4
    A_d=(s^2+2*damp*wn*s+wn^2);
    A_d_den=double(fliplr(coeffs(A_d)));
    % fA1=A+3*eye(4)
    fA1=A+A_d_den(2)*eye(4)
    K=F*[C(1,:)*fA1
         C(2,:)*fA1];
    Bf=B*F;
    Af = A-B*K
    decouple_model=ss(Af,Bf,C,D)
    % step(decouple_model)
    p=pole(decouple_model)
    H=C*inv(s*eye(4)-Af)*Bf

    clc;
    clear;
    load('matrix.mat');
    system=ss(A,B,C,D)
    G=tf(system)
    syms s
    N=[-0.7587*s^3 + 94.85*s^2 + 1251*s + 2815,-0.01574*s^3 + 36.07*s^2 + 465.8*s + 1060;-0.02941*s^3 +
    34.36*s^2 + 382.8*s + 817.6,0.03102*s^3 + 13.1*s^2 + 162.3*s + 556.4]
    Kd=inv(N)*det(N)
    ds=s^4 + 36.42*s^3 + 548.8*s^2 + 3381*s + 6013
    res=(det(N)/ds)*eye(2) %GKd

    thisnow=res;
    answer=gcd(thisnow(1,1),thisnow(2,2));
    Ks=[1/answer,0;0,1/answer]
    K=Kd*Ks;
    H=inv(eye(2)+ res*Ks)*res*Ks
    s = 0
```

```
subs(s)
K1=eval(K)
```

Task 5

```
clc;
clear;
load('matrix.mat');
%%
y_sp=[0.4;0.8];
disturbance=[0.3;0.2];
rank([A B;C zeros(2,2)]);
A_bar=[A zeros(4,2);-C zeros(2,2)];
B_bar=[B;zeros(2,2)];
B_w_bar=[B;zeros(2,2)];
B_r_bar=[zeros(4,2);eye(2)];
C_bar=[C zeros(2,2)];
Q=[1 0 0 0 0 0;
    0 1 0 0 0 0;
    0 0 1 0 0 0;
    0 0 0 1 0 0;
    0 0 0 0 1 0;
    0 0 0 0 0 1]*10;
R=[1 0
    0 1]*0.5;
gamma=[A_bar -B_bar/R*B_bar'
      -Q -A_bar'];
[vector,value]=eig(gamma);
value=sum(value);
v=vector(:,find(real(value)<0));
P=v(7:12,:)/v(1:6,:);
K=real(inv(R)*B_bar'*P);
K1=K(:,1:4)
K2=K(:,5:6)
%% observe
syms l11 l12 l13 l14 l21 l22 l23 l24 s ;
dir_pole=[-12,-11.6,-10,-13];
polynomial=(s-dir_pole(1))*(s-dir_pole(2))*(s-dir_pole(3))*(s-dir_pole(4));
Ad_cof=double(coeffs(polynomial));
L=[l11 l12 l13 l14 ;l21 l22 l23 l24];
W=[C' A'*C' A'^2*C' A'^3*C'];
assert(rank(W(:,1:4))==4);
CC=W(:,1:4);
CC=[CC(:,1),CC(:,3),CC(:,2),CC(:,4)];
```

```
inv_C=inv(CC);
d1=2;
d2=2;
T=[inv_C(d1,:);
      inv_C(d1,:)*A';
      inv_C(d1+d2,:);
      inv_C(d1+d2,:)*A'];
Aba=T*A'/(T);
Bba=T*C';
Aba(abs(Aba)<10^(-10))=0;
Bba(abs(Bba)<10^(-10))=0;
Am=Aba-Bba*L;
Ad=[ 0    1    0    0 ;
       0    0    1    0 ;
       0    0    0    1 ;
      -Ad_cof(1:4) ];
  %% solve equetion
equation=Am==Ad;
L_num=solve(equation);
L_ans=struct2array(L_num);
L_ans=double(L_ans);
Lba=[L_ans(1:4);
      L_ans(5:8)];
L=Lba*T;
L=L'
L=real(L)
```

Task 6

```
clc;
clear;
load('matrix.mat');
B1=[1 0;1 0;1 1;1 1];
xsp=[0;0.5;-0.4;0.3];
syms u1 u2 u3 u4 s
U=[u1;u2;u3;u4]
B2=[B B1];
eq1=inv(s*eye(4)-A)*B2*U;
eq2=subs(eq1,s,0);
ans=solve(eq2==xsp)
u1=double(ans.u1)
u2=double(ans.u2)
u3=double(ans.u3)
u4=double(ans.u4)
U1=[u1,u2]
```

```
U2=[u3,u4]

clc;
clear;
load('matrix.mat');
syms s u1 u2
U=[u1;u2]
a=5
b=4
c=8
d=0
w=[a+1 0 0 0;
      0 b+1 0 0;
      0 0 c+1 0;
      0 0 0 d+1]
xsp=[0;0.5;-0.4;0.3];
xs=inv(s*eye(4)-A)*B*U;
xs2=subs(xs,s,0);
J=1/2*(xs2-xsp)'*w*(xs2-xsp)
jacob = jacobian(J, [u1 u2])
ans=solve(jacob==0)
u1=double(ans.u1)
u2=double(ans.u2)
u=[u1;u2]
```