Assignment 2

Data-intensive Control

Jiaxuan Zhang (1551957)

April 9, 2022

Course:	Embedded Control Systems
Professor:	Prof. Dip Goswami
Date:	April 9, 2022

1 Part 0 | Background

In this assignment, we focused on designing the controller for the LKAS system.

2 Part 1 | Case 1

The τ and h can be calculated by the following equations:

$$\begin{aligned} &\tau = t_{isp1} + t_{isp2} \cdot 8 + t_{isp3} + t_{RoID} + t_{RoIP} \cdot 8 + t_{RoIM} + t_{Controller} + t_{Actuator} = 96.7ms \\ &h = \lceil \frac{\tau}{10} \rceil \cdot 10 = 100ms; \end{aligned} \tag{1}$$

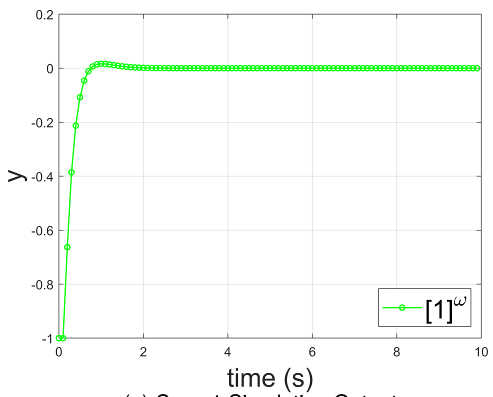
For simplicity, we can add a delay module so that we can make $\tau = h = 100ms$.

For Case 1, I tried the LQR controller. Because the original state-space model of the system is actually not stable, Kalman decomposition is used. We cannot directly set each entry in Q because we cannot make sure which state is directly related to our desired output after Kalman decomposition. So set Q by $Q = Q_{parameter} C_d C_d^T$, where the C_d is the C matrix after Kalman decomposition. Intuitively, the higher ratio between $Q_{parameter}$ and R, the shorter the settling time. However, when I tried to use an LQR controller with higher $Q_{parameter}$ (with R fixed), it is not always stable (in the `vrep` simulation, the fluctuation gradually increase). The performance of several different Q and R values are shown in Table 1. ($Q = Q_{parameter} C_d C_d^T$).

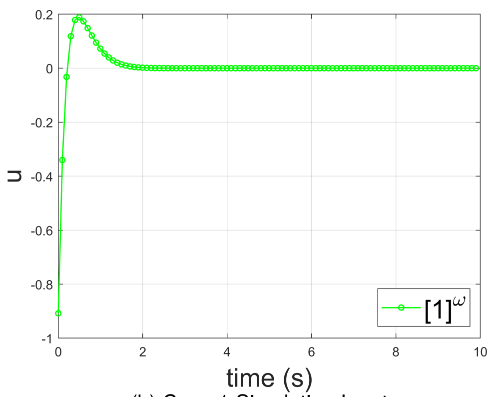
Table 1: Case 1 Parameters Test Result

	$Q_{parameter}$	R	Settling Time (s)	Simulation Stable
1	5	3	0.675	yes
2	10	3	0.557	yes
3	20	3	0.459	no
4	100	3	0.294	no

Finally, I choose the $Q_{parameter} = 5, R = 3$ for designing the LQR controller. The result of the simulation of the MATLAB Script is shown in Figure 1. The scope figure of `vrep` simulation is shown in Figure 2. It can be seen that the car can stay around the expected line and the fluctuation gradually decreases.



(a) Case 1 Simulation Output



(b) Case 1 Simulation Input

Figure 1: Case 1 Simulation Result

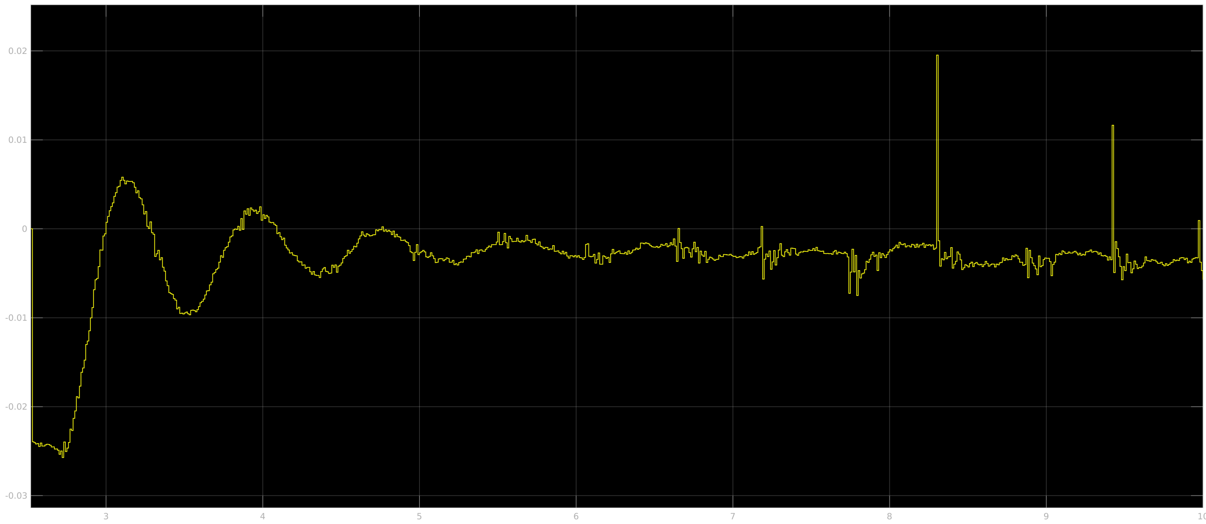


Figure 2: Case 1 $Q_{parameter} = 5$

3 Part 2 | Case 2

The τ and h can be calculated by the following equations:

$$\begin{aligned} &\tau = t_{isp1} + t_{isp2} \cdot 8/8 + t_{isp3} + t_{RoID} + t_{RoIP} \cdot 8/8 + t_{RoIM} + t_{Controller} + t_{Actuator} = 26.3ms \\ &h = \lceil \frac{\tau}{10} \rceil \cdot 10 = 30ms; \end{aligned} \tag{2}$$

For simplicity, we can add a delay module so that we can make $\tau = h = 30ms$

For Case 1, I tried the LQR controller. Because the original state-space model of the system is actually not stable, Kalman decomposition is used. We cannot directly set each entry in Q because we cannot make sure which state is directly related to our desired output after Kalman decomposition. So set Q by $Q = Q_{parameter} C_d C_d^T$, where the C_d is the C matrix after Kalman decomposition. Intuitively, the higher ratio between $Q_{parameter}$ and R, the shorter the settling time. However, when I tried to use an LQR controller with higher $Q_{parameter}$ (with R fixed), it is not always better in the `VREP` simulation. The performance of several different Q and R values are shown in Table 2. ($Q = Q_{parameter} C_d C_d^T$).

Table 2: Case 2 Parameters Test Result

	$Q_{parameter}$	R	Settling Time (s)	Simulation Stable
1	5	3	0.595	yes
2	10	3	0.475	yes
3	20	3	0.375	yes
4	100	3	0.206	yes

All the $Q_{parameter}$ settings will lead to an stable controller, however, they perform differently, a larger $Q_{parameter}$ always means more fluctuation. For example, the Figure 3 shows the `vrep` simulation result for $Q_{parameter} = 100$, the Figure 4 shows the `vrep` simulation result for $Q_{parameter} = 20$ the Figure 5 shows the `vrep` simulation result for $Q_{parameter} = 5$.

Finally, I choose the $Q_{parameter} = 20$. Because it has a shorter settling time in the MATLAB simulation and has no large fluctuation in `vrep` simulation. The result of the simulation of the MATLAB Script is shown in Figure 6. The scope figure is shown in Figure 4. It can be seen that the car can stay around the expected line and the fluctuation gradually decreases.

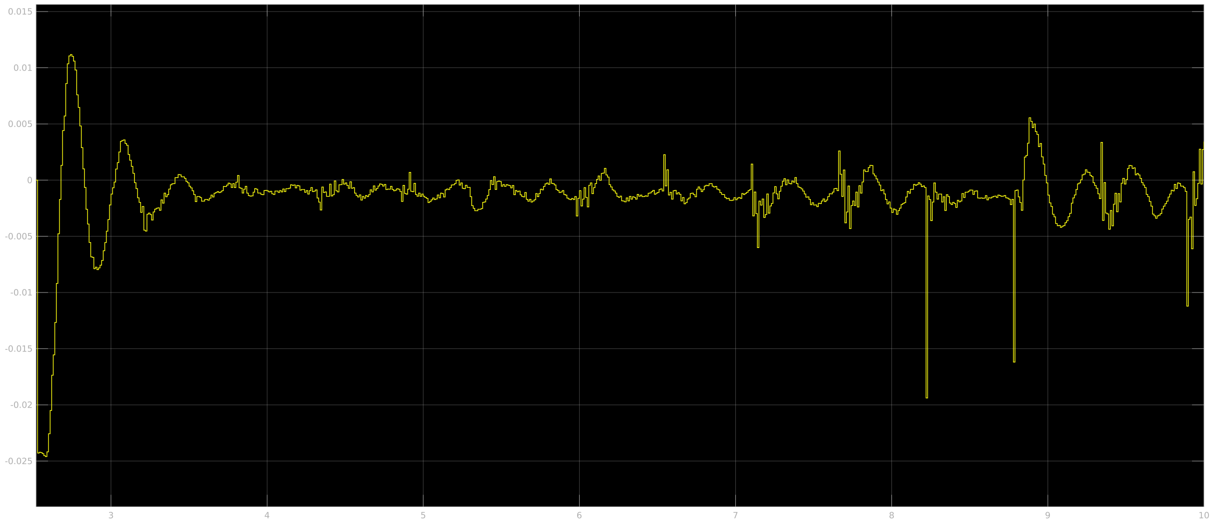


Figure 3: Case 2 $Q_{parameter} = 100$

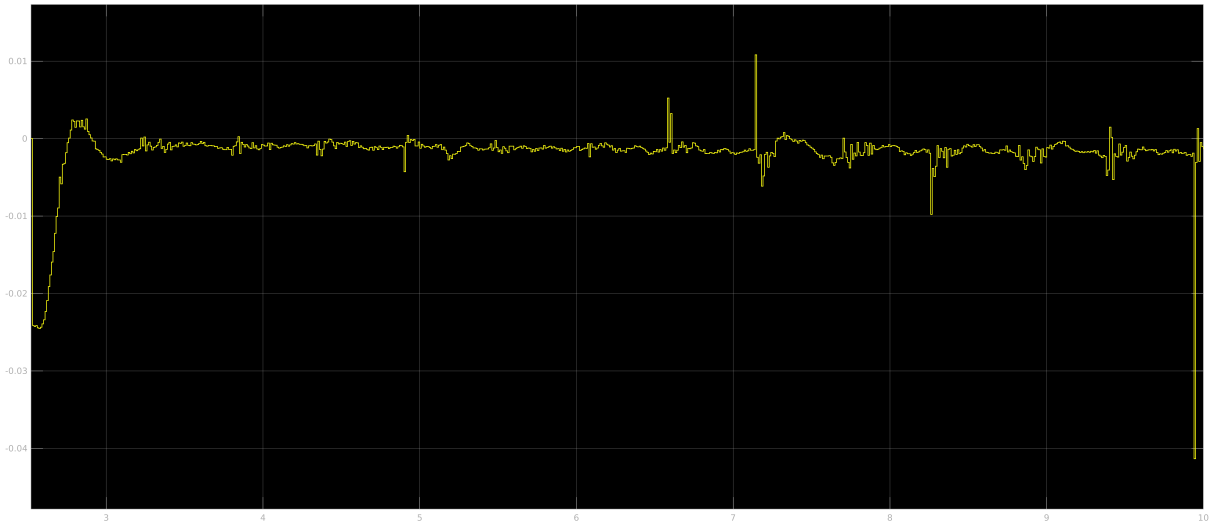


Figure 4: Case 2 $Q_{parameter} = 20$

3.1 Comparison Between Case 1 and Case 2

Two observations can be made from the MATLAB simulation and the VREP simulation.

1. After adding parallelization, the system can achieve a shorter settling time. It may be because parallelization shortens the sampling period. This means in a given time period, the LQR controller will take more states into consideration.
2. After adding parallelization, the system becomes more stable even under larger Q. The possible explanation is that the smaller the sampling time, the system can achieve a smoother control process.

4 Part 3 | Case 3

The τ and h can be calculated by the following equations:

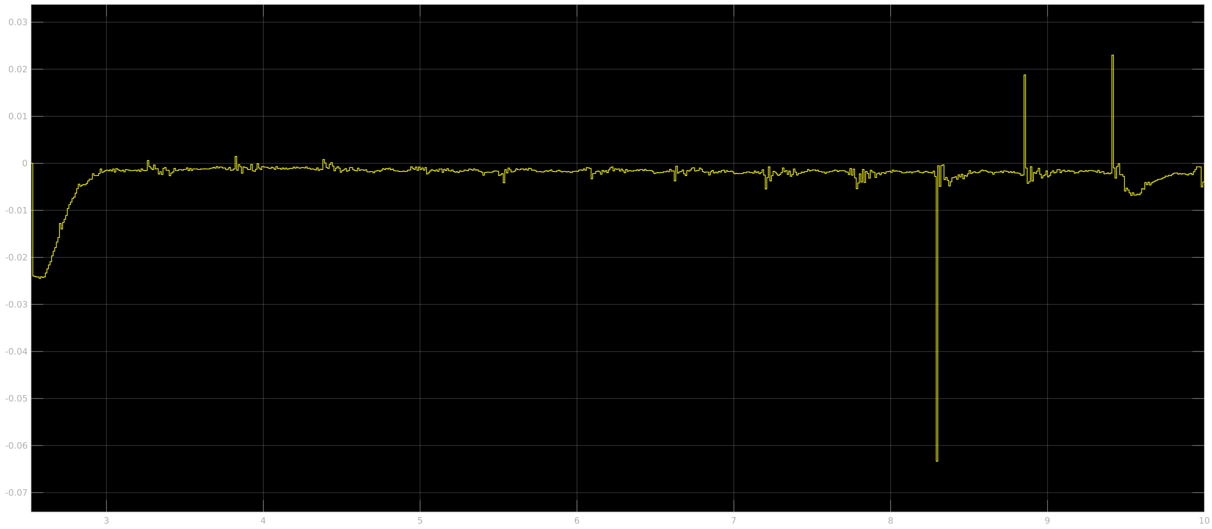
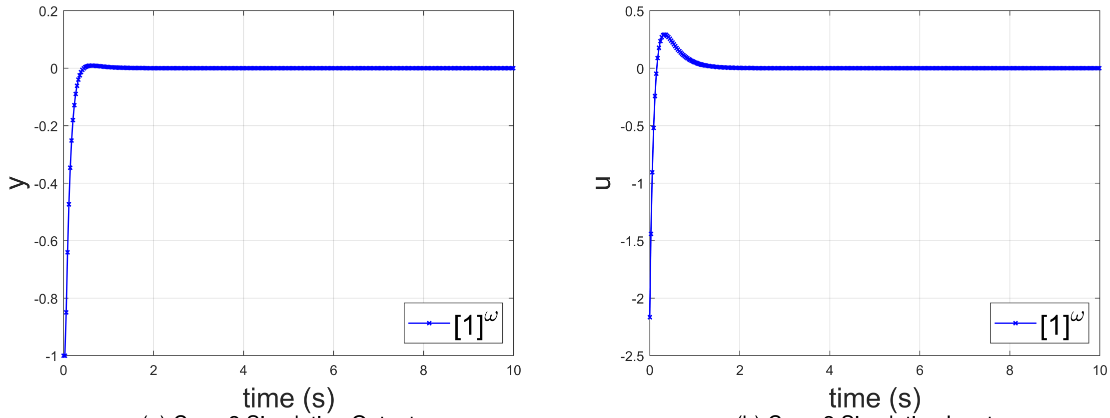


Figure 5: Case 2 $Q_{parameter} = 5$



(a) Case 2 Simulation Output

(b) Case 2 Simulation Input

Figure 6: Case 2 Simulation Result

$$\begin{aligned} &\tau = t_{isp1} + t_{isp2} \cdot 8 + t_{isp3} + t_{RoID} + t_{RoIP} \cdot 8 + t_{RoIM} + t_{Controller} + t_{Actuator} = 96.7ms \\ &h = \lceil \frac{\tau}{\text{core number} \cdot 10} \rceil \cdot 10 = 20ms; \end{aligned} \tag{3}$$

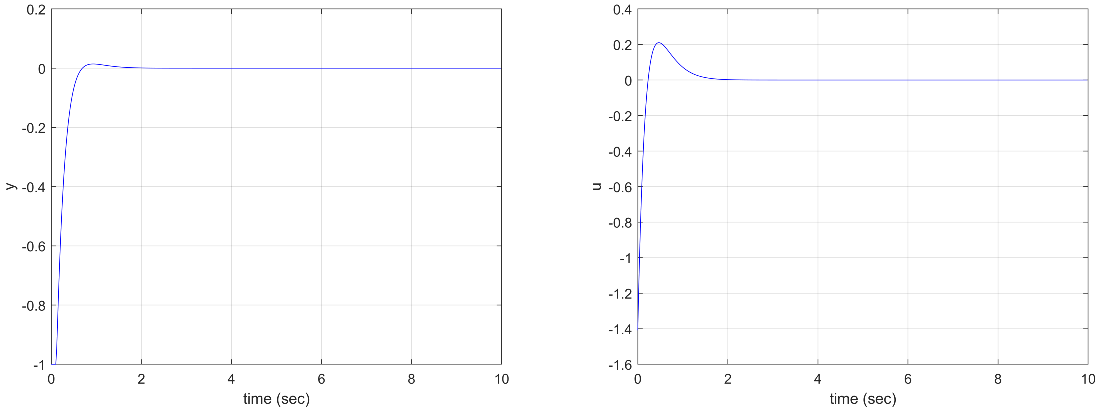
For simplicity, we can add a delay module so that we can make $\tau = 100ms$

For Case 3, I tried the LQR controller. Because the original state-space model of the system is actually not stable, Kalman decomposition is used. We cannot directly set each entry in Q because we cannot make sure which state is directly related to our desired output after Kalman decomposition. So set Q by $Q = Q_{parameter} C_d C_d^T$, where the C_d is the C matrix after Kalman decomposition. Intuitively, the higher ratio between $Q_{parameter}$ and R, the shorter the settling time. However, when I tried to use an LQR controller with higher $Q_{parameter}$ (with R fixed), it is not always stable (in the `vrep` simulation, the fluctuation gradually increase). The performance of several different Q and R values are shown in Table 3. ($Q = Q_{parameter} C_d C_d^T$).

Then I decided to choose $Q_{parameter} = 7$ as the controller parameter. The result of the simulation of the MATLAB Script is shown in Figure 7. The scope figure is shown in Figure 8. It can be seen that the car can stay around the expected line and the fluctuation gradually decreases.

Table 3: Case 3 Parameters Test Result

	$Q_{parameter}$	R	Settling Time (s)	Simulation Stable
1	5	3	0.55	yes
2	7	3	0.605	yes
3	10	3	0.545	not
4	20	3	0.44	not
5	100	3	0.275	not



(a) Case 3 Simulation Output (b) Case 3 Simulation Input

Figure 7: Case 3 Simulation Result

5 Part 4 | Case 4

The τ and h can be calculated by the following equations:

$$\begin{aligned} \tau &= t_{isp1} + t_{isp2}\cdot 8 + t_{isp3} + t_{RoID} + t_{RoIP}\cdot 8 + t_{RoIM} + t_{Controller} + t_{Actuator} = 96.7ms \\ h &= \lceil \frac{\tau}{\text{core number}\cdot 10} \rceil \cdot 10; \end{aligned} \tag{4}$$

For simplicity, we can add a delay module so that we can make $\tau = 100ms$. It can be easily computed the different values of h with a different number of pipelines. The h values are shown in Table 4. It can be found that, when the core number is larger than 5, the h will be the same as the h of 5 cores.

Table 4: Influence of Core Number in Parallelization

	Core Number	tau(ms)	h(ms)
1	1	96.7	100
2	2	96.7	50
3	3	96.7	40
4	4	96.7	30
5	5	96.7	20
6	6	96.7	20
7	7	96.7	20
8	8	96.7	20

For Case 4, the same $Q_{parameter} = 7$ is chosen as the parameter in Case 3. Besides, the performance of 3 and 5 cores are tested. The result of the simulation of the MATLAB Script is shown in Figure 9. The scope figure is shown in the Figure 10 and the Figure 11. It can be seen that the car can stay around the expected line and the fluctuation gradually decrease.

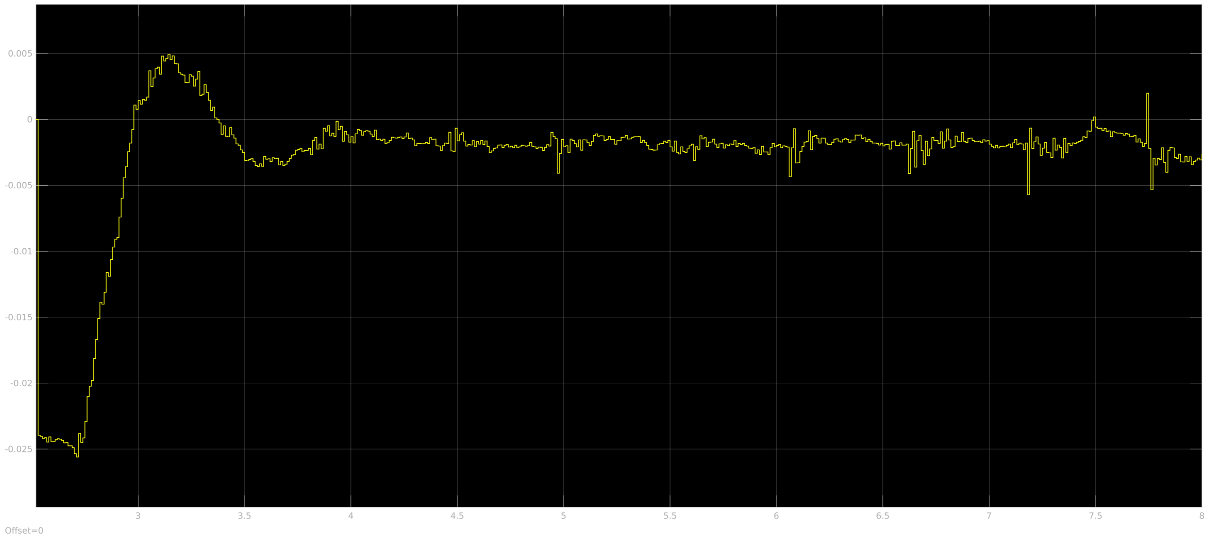
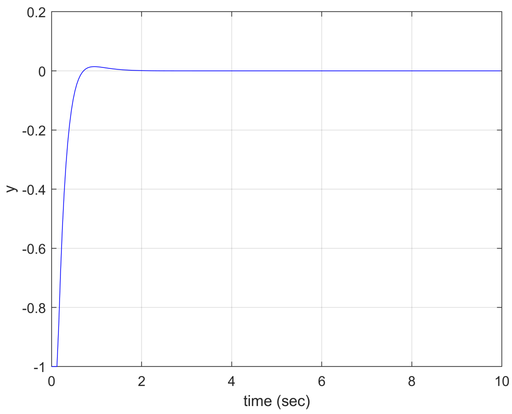
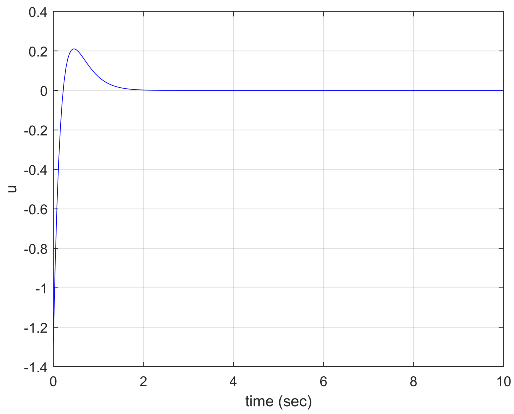


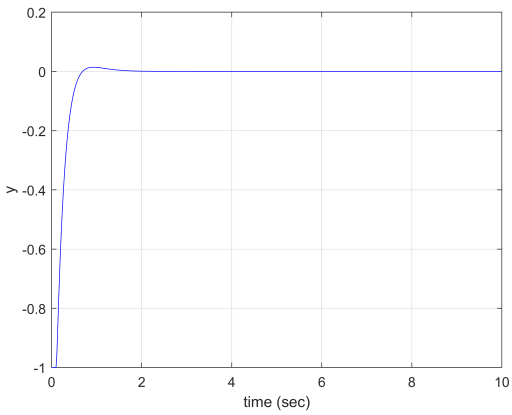
Figure 8: Case 3 VREP Simulation Result



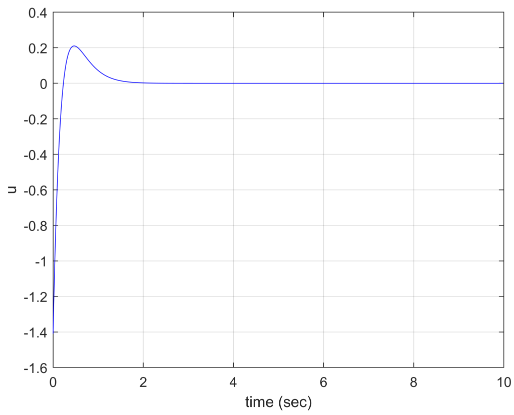
(a) Case 4 Simulation Output: 3 Cores



(b) Case 4 Simulation Input: 3 Cores



(c) Case 4 Simulation Input: 5 Cores



(d) Case 4 Simulation Input: 5 Cores

Figure 9: Case 4 Simulation Result

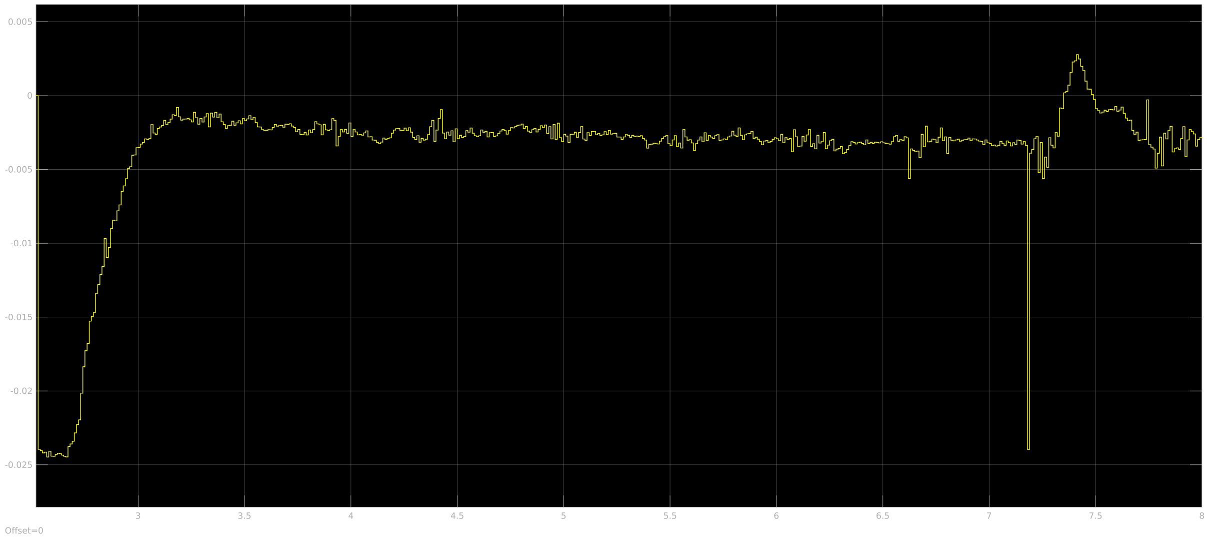


Figure 10: Case 4 VREP result: 3 Cores

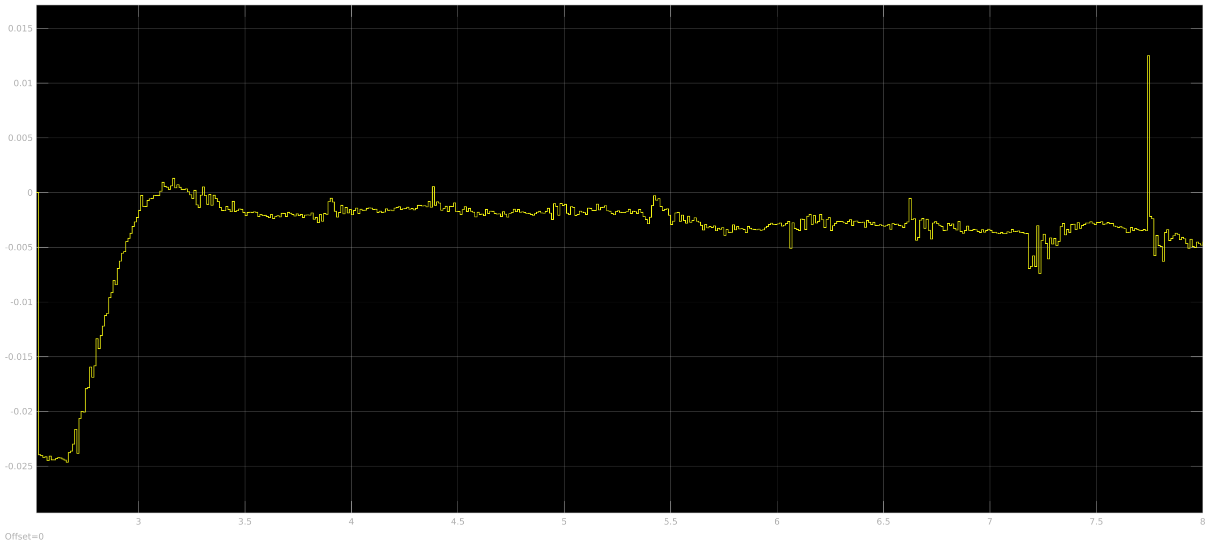


Figure 11: Case 4 VREP result: 5 Cores

5.1 Analysis of Different Core Numbers

Based on the `vrep` simulation result of Case 3 and Case 4. Several observations can be made:

1. With more pipelines, the system can achieve a better steady-state. This happens not only in the comparison of 5 pipelines and 3 pipelines cases, this phenomenon can also be observed in the comparison of 8 pipelines and 5 pipelines, although not obvious.

2. With more pipelines, a higher overshooting can be observed. It may be because the delay between feedback input and the sensor information. The more pipeline, the larger the delay.

6 Part 5 | Case 5

For Design 5, I first calculated the τ and h for several combinations of parallelization number and pipeline number. The result is shown in Table 5

Table 5: Performance of Combinations of Parallelization and Pipeline

	No. Parallel	No. Pipeline	Total Core	tau(s)	h(s)
1	2	2	4	0.0565	0.03
2	2	3	6	0.0565	0.02
3	2	4	8	0.0565	0.02
4	4	2	8	0.0364	0.02

Based on the Table, I decide to choose the combination of 8 cores. The parallelization number is set to 4 and the pipeline number is set to 2. From the table 5, the τ is $0.0364s$ and the h is $0.02s$. I then add some delay module to make $\tau = 0.04s$

For the 4 parallelization + 2 pipeline structure, I tried different $Q_{parameter}$. The performance of different $Q_{parameter}$ is shown in the Table 6.

Table 6: Case 5 Parameters Test Result

	$Q_{parameter}$	R	Settling Time (s)	Simulation Stable
1	5	3	0.607	yes
2	7	3	0.545	yes
3	10	3	0.48	yes
4	20	3	0.382	yes
5	100	3	0.217	not

Two examples of VREP simulation resulSt are shown. The result of $Q_{parameter} = 20$ is shown in Figure 12. The result of $Q_{parameter} = 7$ is shown in Figure 13. It can be seen that, with a higher $Q_{parameter}$, the system can achieve a better steady-state result while a higher $Q_{parameter}$ will lead to the overshooting phenomenon.

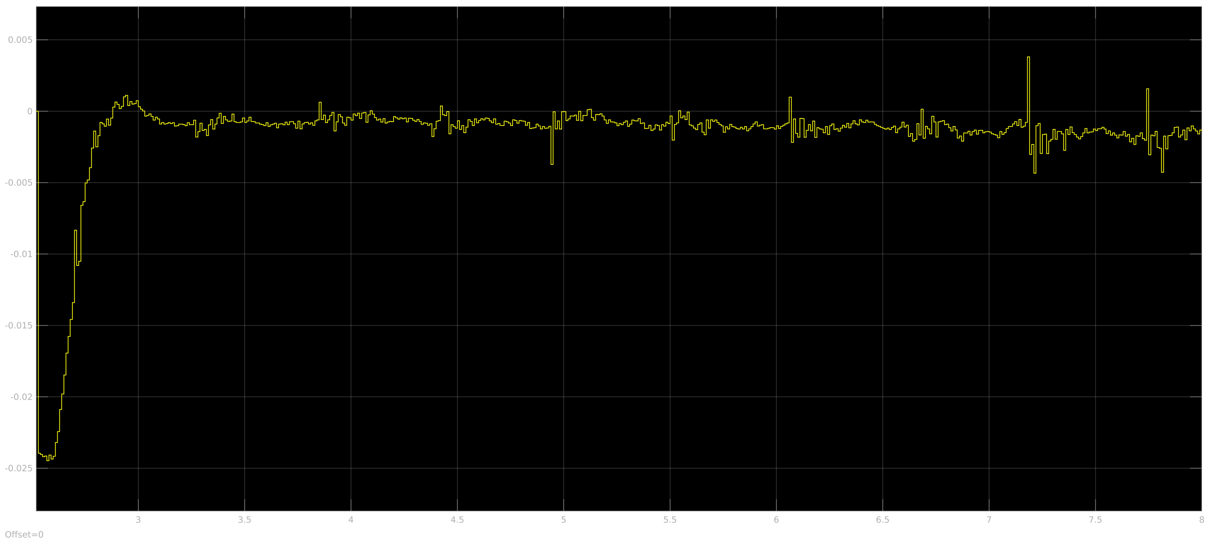


Figure 12: Case 5 VREP simulation result: $Q_{parameter=20}$

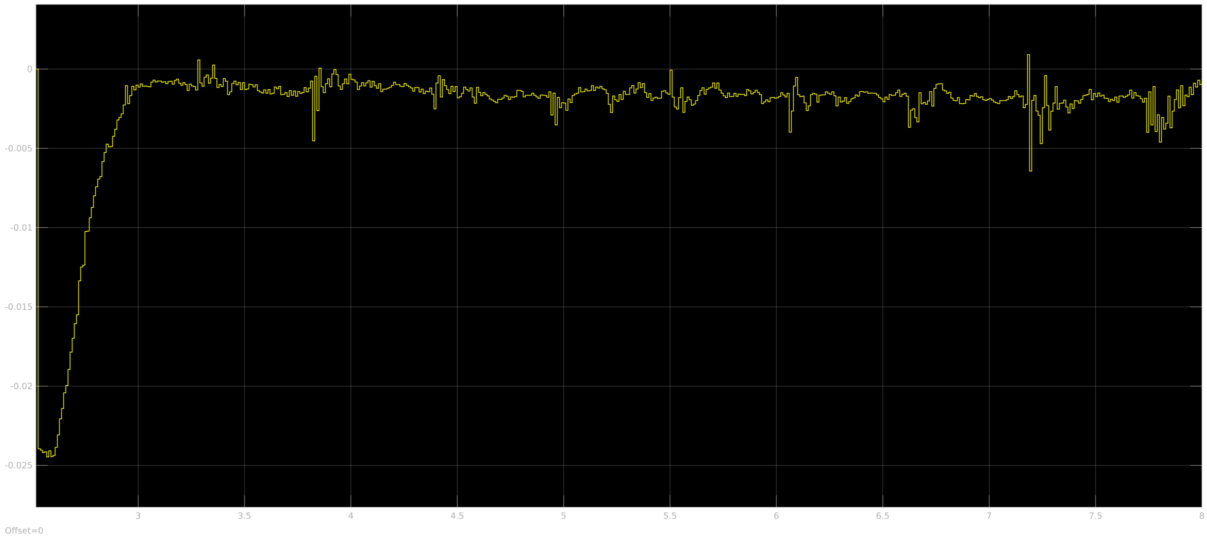
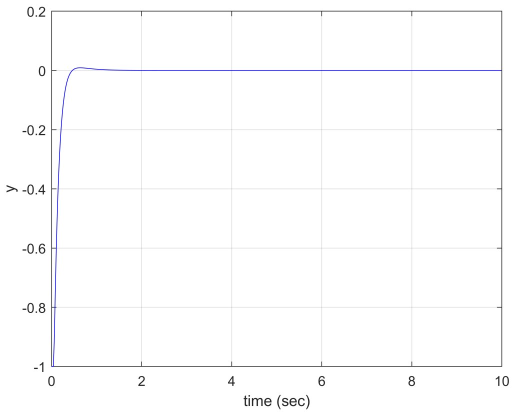
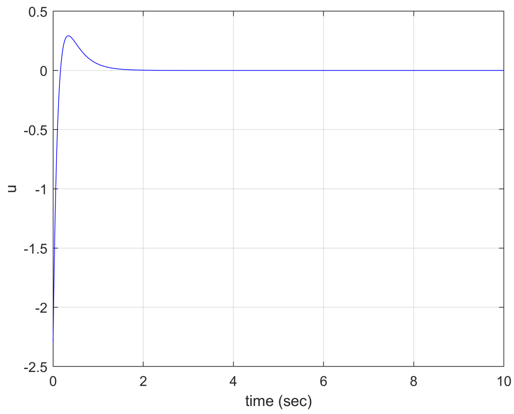


Figure 13: Case 5 VREP simulation result: $Q_{parameter=7}$

Finally, I choose $Q_{parameter} = 20$ as the controller parameter, because in VREP simulation, it can achieve a better steady-state result with a very small overshooting. The result of the simulation of the MATLAB Script is shown in Figure 14. The scope figure is shown in Figure 15. It can be seen that our controller can achieve a pretty good steady-state with a small overshooting.



(a) Case 5 Simulation Output



(b) Case 5 Simulation Input

Figure 14: Case 5 Simulation Result

7 Part 6 | Comparison and Conclusion

Based on the result of Case 1 to Case 5, we can generate several conclusions.

- A large Q theoretically has better settling time, but the system may not perform stably in `vrep` simulation.
- By adding pipeline and parallelization, the system can achieve better settling time in MATLAB simulation when we fixed the Q. It may be because adding pipeline and parallelization can reduce the sampling time.
- Pipeline can sometimes result in overshooting phenomenon, the more pipeline we use, the larger overshooting it will cause.

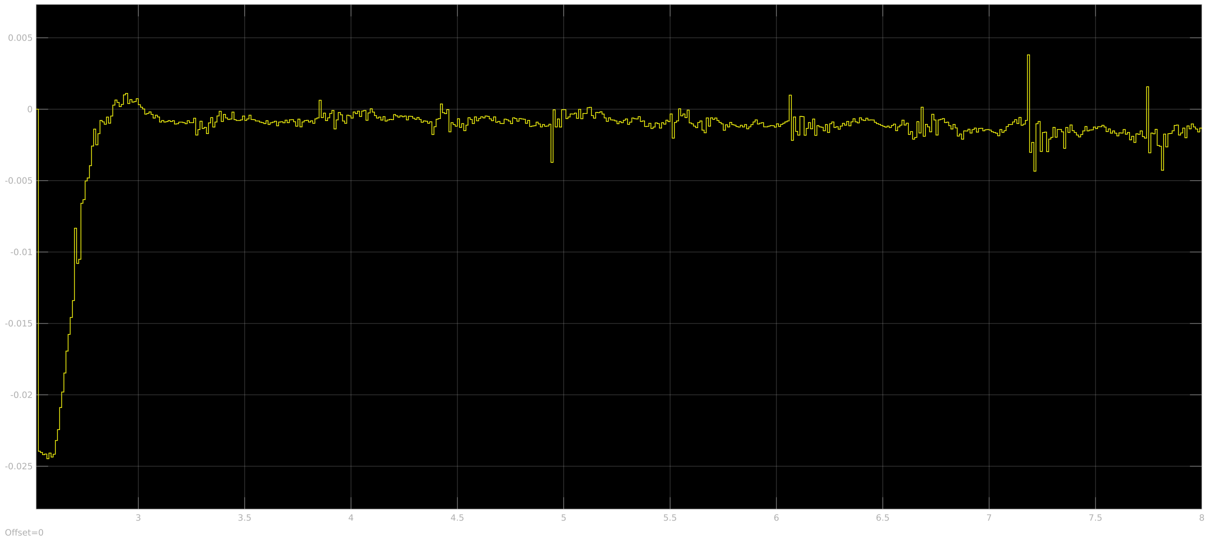


Figure 15: Case 5 VREP simulation result: $Q_{parameter=20}$

Based on these conclusions and the simulation result, we can propose two general rules when designing the controller by considering h and τ:

- We should try to shorten the sampling time as much as possible under the premise of ensuring stability.
- When considering the method of parallelization and pipeline, parallelization should be preferred. Because parallelization has a smaller overshooting phenomenon and will use fewer hardware resources. However, in reality, we may not know the detail of the processing task.