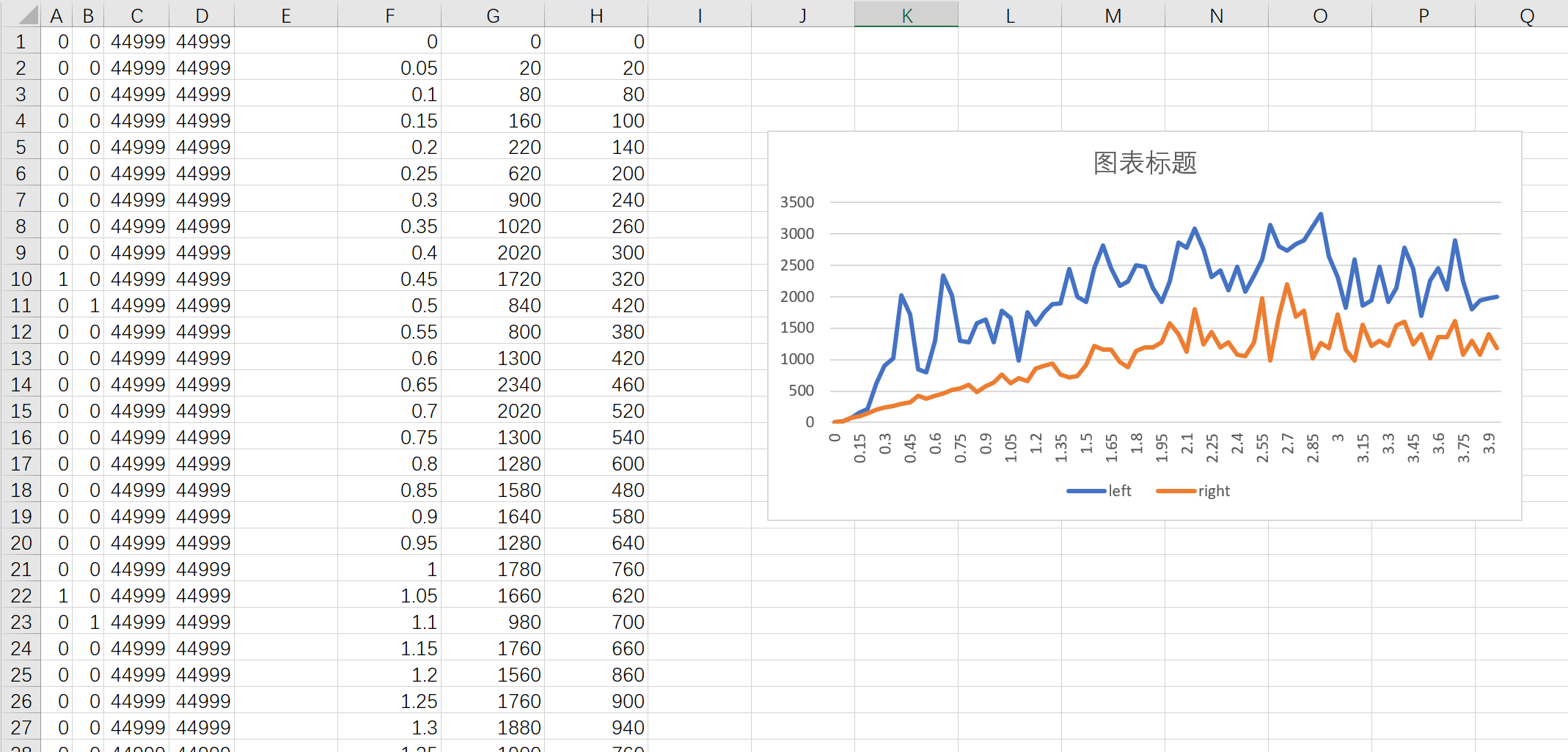
Joint lab report

Introductory

In this experiment, we need to combine the content of the two courses to control the car to go straight. The 3123 theory is used to design the control system, and the 3105 knowledge is used to put the designed control system into the microprocessor for execution.

Procedure

Using the counter value divided by the sampling period 2.5ms directly, the graph looks very strange. So, I try to look at this way: consider the sampling period to be 50ms, so I took 20 of the counter value as one set and then plot the graph. The graph was shown below:



As shown in the figure, the two wheels perform quite different, speed of left wheel is high than the right wheel in the same PWM 44999. So the robot car keep moving circle within the 4 seconds.

The first step is to construct the models for two motors.

The steady state value of left wheel and right wheel are 2500 and 1250 respectively.

The time constant can be found at around 63% of is final value.

For left wheel, 2500\*63%=1575, corresponding time constant=1, pole=1/time constant=1

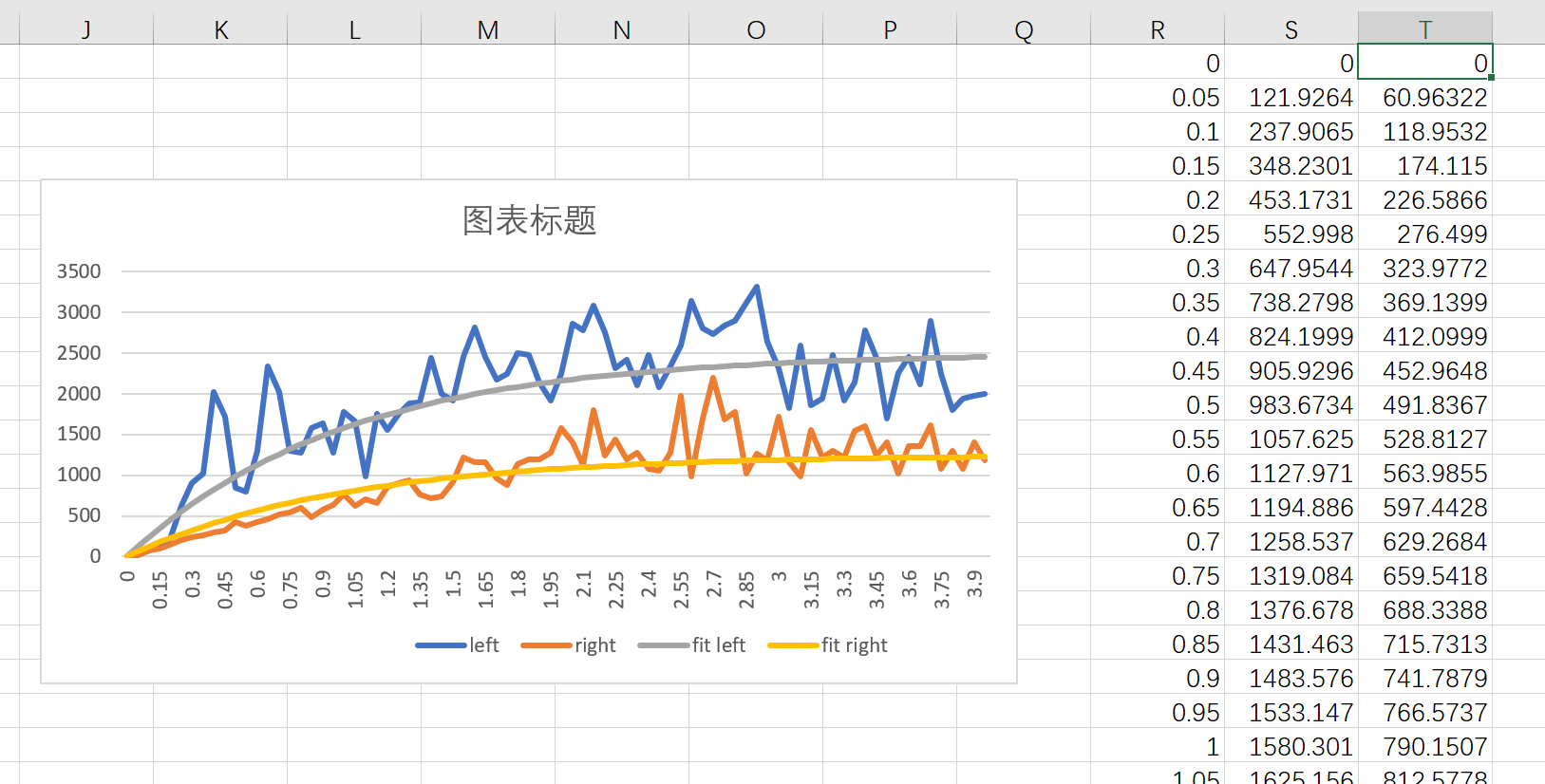
For left wheel, 1250\*63%=787.5, corresponding time constant=1, pole=1/time constant=1

The model of this two wheels can be estimated as:

2500\*(1-EXP(-1\*R1)) and

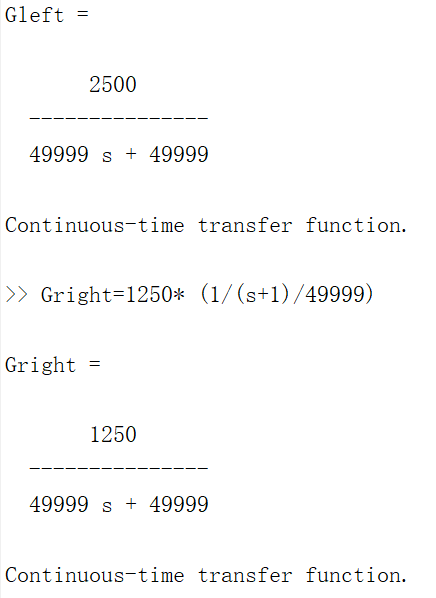
1250\*(1-EXP(-1\*R1))

Where R is a list of time{0, 0.05 0.1…}

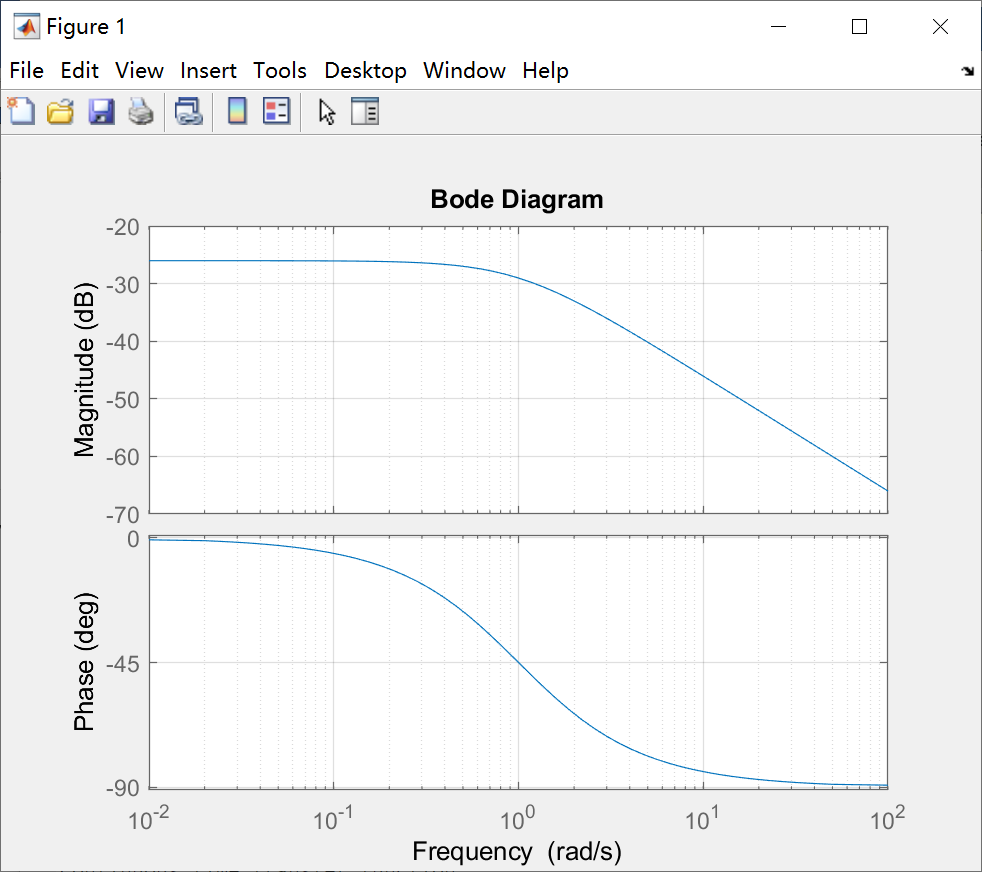


Gleft(s) =2500\* (1/(s+1))(1/49999)

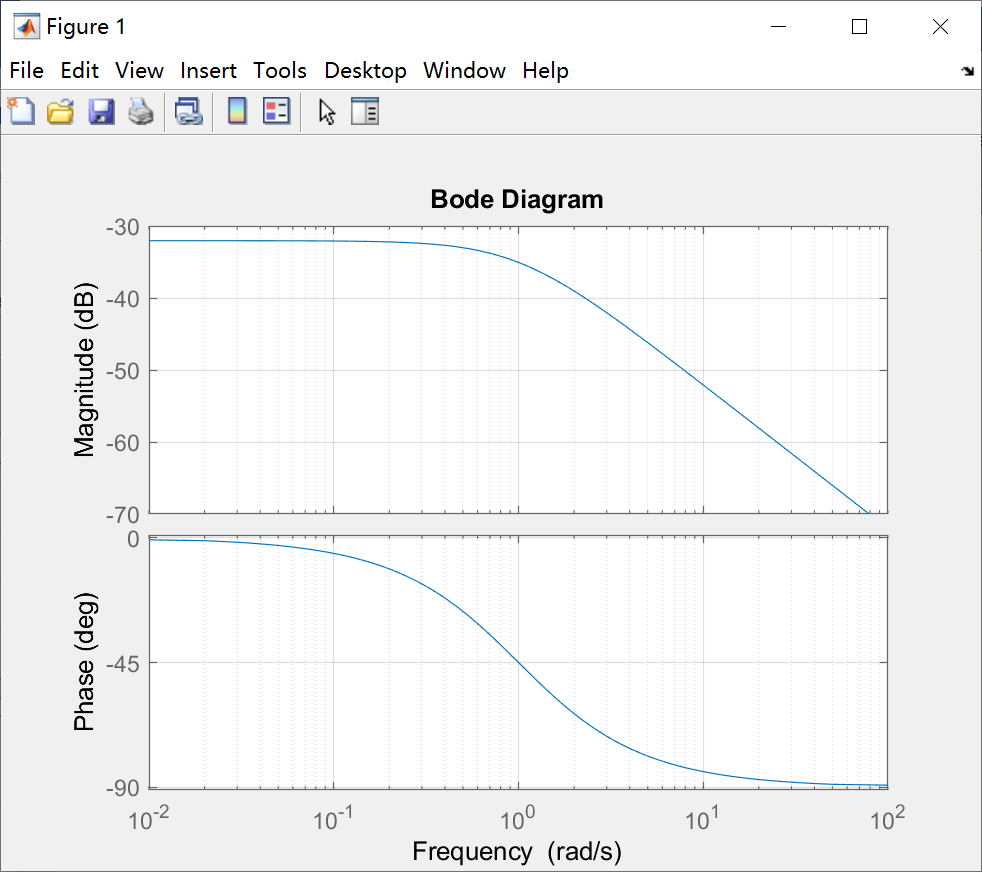
Gright(s) = 1250\*(1/(s+1))(1/49999)



Bode plot of left wheel:



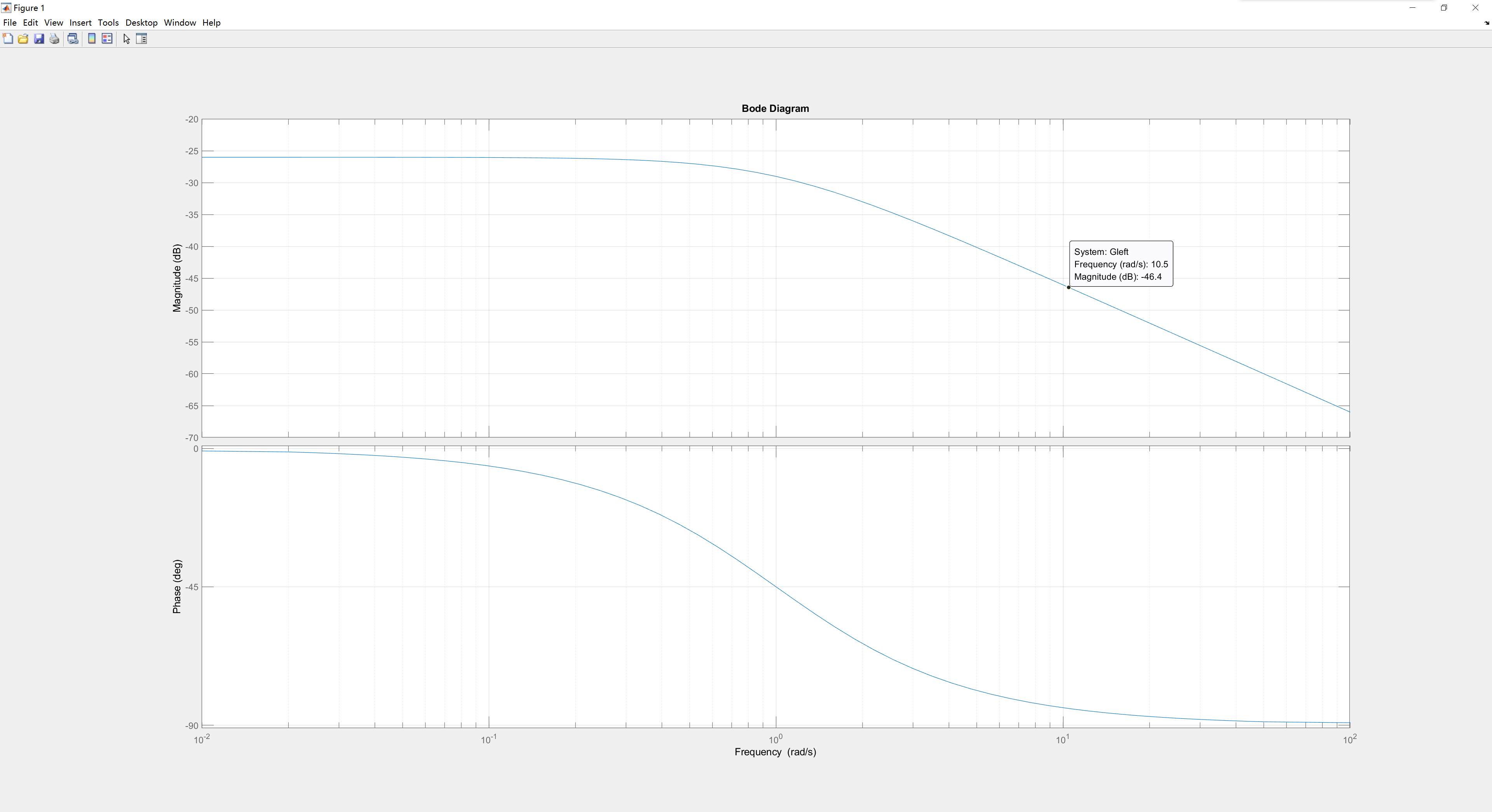
Bode plot of right wheel:



A PI controller is needed to obtain zero steady state error.

Gc(s)=Kp+KI/s=Kp(s+KI/Kp)/s

For left wheel:



Let desire 0db frequency=10

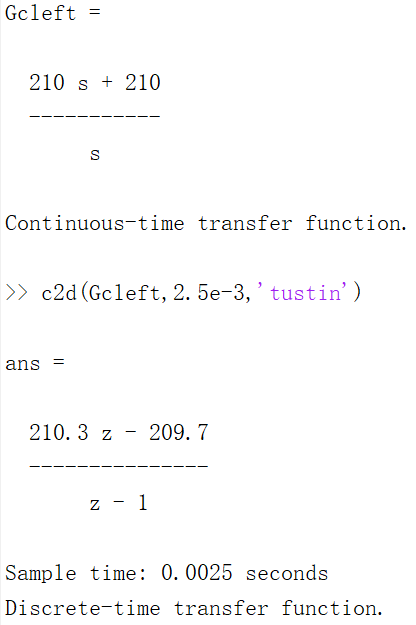
20logKp=46.4

Kp=210

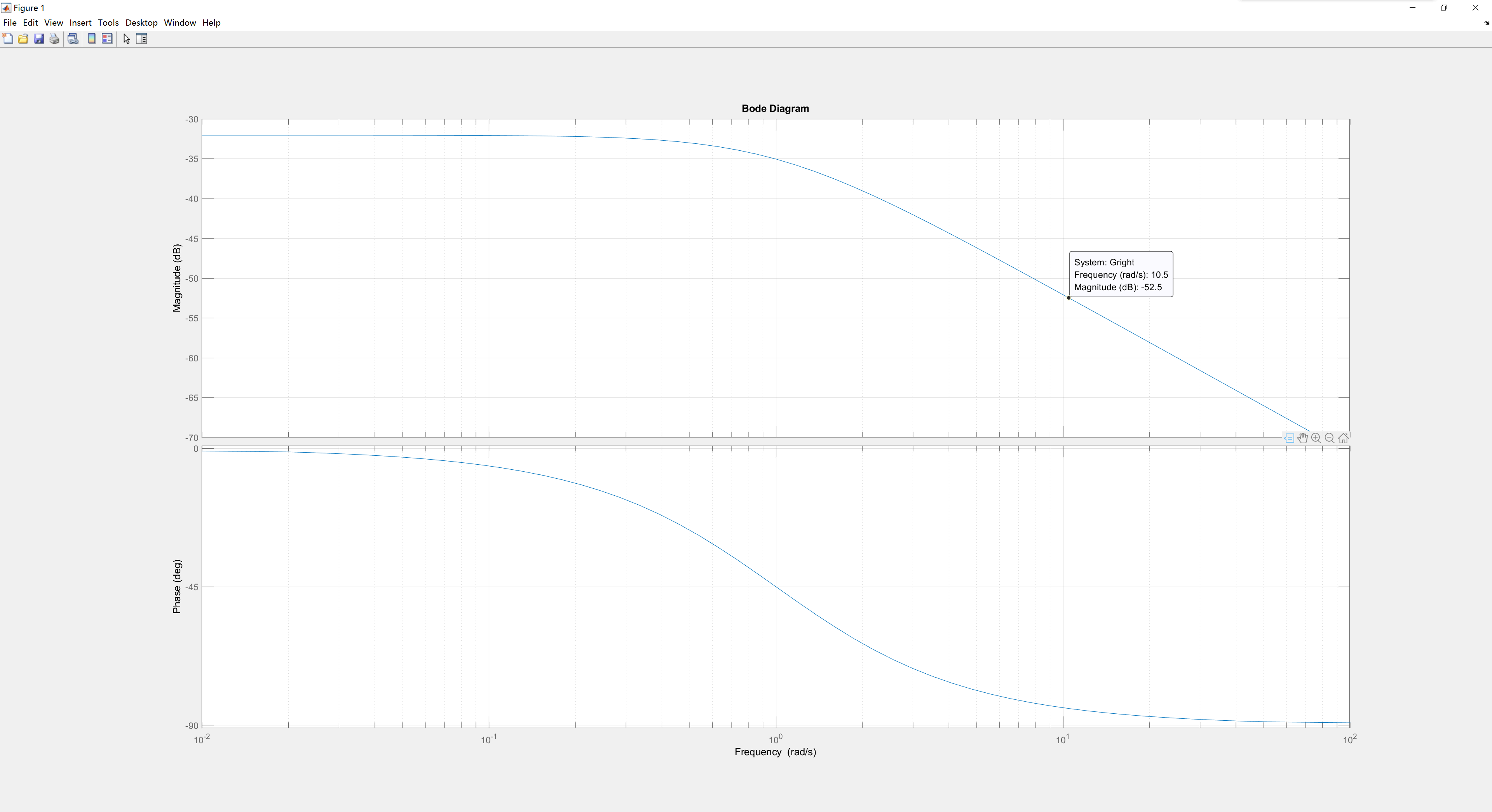
KI/Kp=1

KI­=210

Compensator equation Gc(s)=210(s+1)/s



For right wheel:



Let desire 0db frequency=10

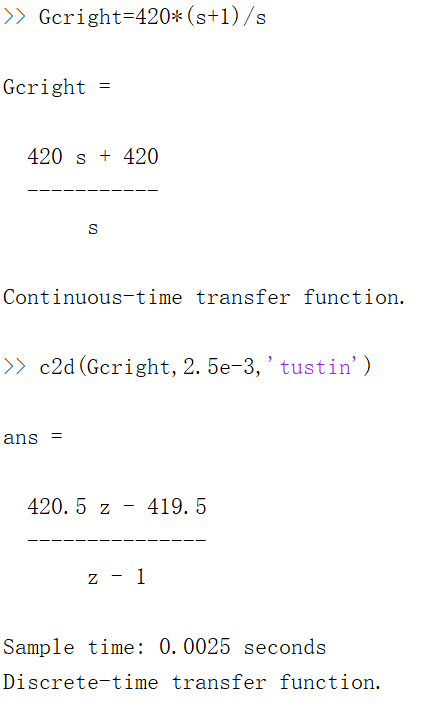
20logKp=52.5

Kp=420

KI/Kp=1

KI­=420

Compensator equation Gc(s)=420(s+1)/s



Note that G(z)=D(z)/E(z).

Difference equation are shown below:

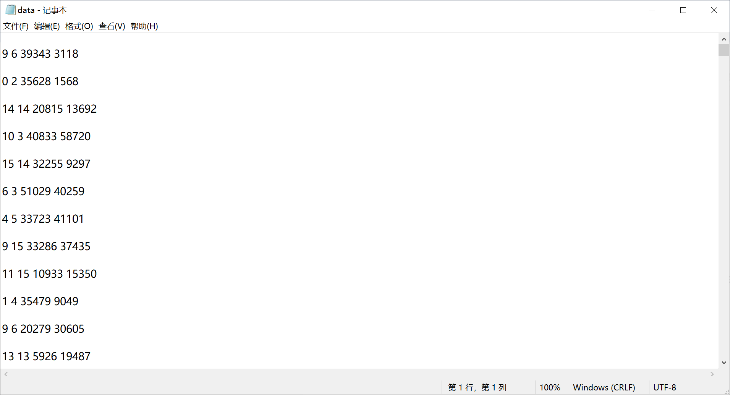
Dleft[k]= Dleft[k-1] +210.3\*e[k]-209.7\*e[k-1]

Dright[k]= Dright[k-1] +420.5\*e[k]-419.5\*e[k-1]

The design of control part is finished, now I need to implement the control system into the ARM controller.

Notice that the equation needs previous value, so it is needed to create some variable to store the previous value and the current value.

The difficulties in the implementation

At first, I didn't know where to start, so I could only follow the video instructions. The data of the car is quite strange. PWM will change on its own without a compensation system. I did it a few times, sometimes it changed, sometimes it didn’t. Finally, I chose the full duty cycle 44999 to get the data. The difference from the video is that my two wheels are a bit far apart, so I can't use one model to represent the two wheels like in the video. I need to design the system separately. In theory, I have two parts to complete. The first is to make the two wheels have the same speed at the end (walking in a straight line), and then to control them to follow the original path in a straight line. Unfortunately, I did not achieve the second goal. During the test, my car didn't stop after 4 seconds, which caused it to hit the wall at full speed and the batteries all came out. During the next test, I found that one of the wheels would not move. I started late. After Christmas, I went back to the laboratory and found that the laboratory was no longer open, so I couldn't change parts, and I couldn't shoot demo videos.