Delft University of Technology

SC42095 Control Engineering

Surface grinding wheel control

Abstract

This report contains multiple designs for controllers to control a fictitious grinding machine. In the first part, two continuous-time controllers are discussed. Next, different discrete-time controllers are created. After creating multiple different controllers, these controllers are recalibrated such that the control input stays between given bounds. Finally, the effect of a delay is studied and adjusted for.

Author:
Willem van der Linden (4902815)

Assignment 22 - Surface grinding wheel control

January 9, 2023

Delft University of Technology

Contents

1 Introduction

This report is part of the SC42095 Control Engineering course taught, at the Delft Center for Systems and Control. In this report, questions are answered on both continuous-time and discrete-time control, with an emphasis on discrete-time control.
This assignment contains the answers to thirteen questions. The questions are not explicitly included in this report.

The system, that this report is about, is given by the transfer function in equation (1). This equation describes how the input to an axis of a grinding machine changes the position of the grinding wheel.

$$G(s) = \frac{5}{s(0.5s+1)(0.2s+1)} \tag{1}$$

In the figure below (figure 1) an example grinding machine is given. A fast-rotating grinding wheel made from abrasive material is pressed against and moved along a workpiece. The figure shows a cylindrical grinding machine which means that the workpiece itself is also rotating. Machines like this are used to make high-precision parts with tolerances usually below 0.01mm.

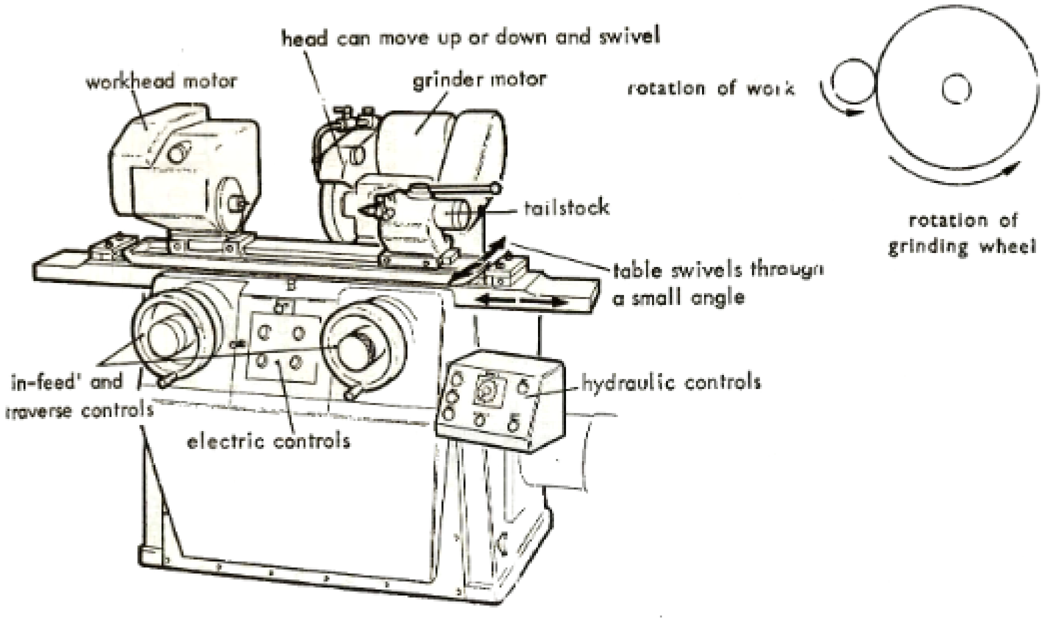


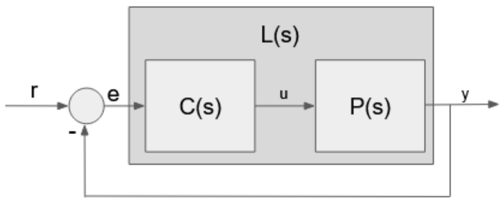
Figure 1: Example of a cylindrical grinding machine

2 Continuous-time Control

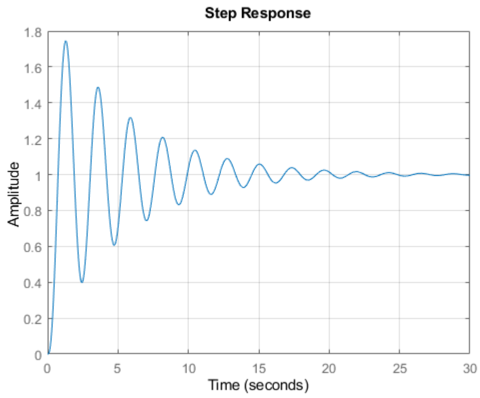
This first section deals with control of the system in continious time. The continious time controllers will serve as a baseline for the discrete time controllers.

2.1 Question 1: Reference tracking

In continuous time a first feedback controller is used with a control scheme as given in figure fig. 2a. C(s) denotes the controller that will be synthesized later, P(s) is the system with the transfer function given in 1. To ease notation L(s) denotes the combination of the controller and plant. Figure 3b shows the step response of the closed loop system with no controller.



(a)



(b)

Figure 2: (a): Control scheme used in this exercise, (b): Step response of the closed loop system with no control.

The transfer function of a closed loop system as depicted in Figure 2a is given in equation (2). The poles of the closed-loop system without controller are given in 1

$$G_1(s) = \frac{C(s)P(s)}{1 + C(s)P(s)} \tag{2}$$

0.00
-5.00
-2.00
-6.63
-0.19 ± 2.74i

Table 1: Poles of the uncontrolled closed loop system

In this first part, a PID-like controller will be created to make the closed loop continuous time behave in a specified way. The goal is to minimize the settling time while the overshoot stays under 5%. The task is specified as follows:

1. Minimal settling time, where the settling time is the time it takes for the system to reach the final state with a 1% (and not deviate from the final state with more than 1%).
2. Overshoot less than 5%
3. Zero steady-state error

2.1.1 Integral term

To check for steady state error the final value theorem is used (3):

$$\lim_{t\to\infty}(y(t)) = \lim_{s\to 0} sP(s) \tag{3}$$

Applying a step response to the system in the Laplace domain corresponds to multiplying the Laplace domain closed-loop transfer function with an integrator $\frac{1}{s}$. This shows that the system has no steady-state error if no control is applied (4):

$$\lim_{s\to 0} sG_1(s)\frac{1}{s} = \frac{s}{s}\frac{5}{s(0.5s+1)(0.2s+1)+5} = 1 \tag{4}$$

Since the controller is not yet defined the final value theorem is used in a parametric way. In which the numerator (denoted by subscript N) and denominator (denoted by subscript D of the controller (C) and plant(P) are specified separately:

$$G_1 = \frac{\frac{C_N}{C_D}\frac{P_N}{P_D}}{1+\frac{C_N}{C_D}\frac{P_N}{P_D}} = \frac{C_N P_N}{C_D P_D + C_N P_N} \tag{5}$$

The limit of s to 0 gives $P_D = s(0.5s+1)(0.2s+1)$ goes to zero. Therefor equation (5) reduces to the following expression:

$$\lim_{s\to 0} sG_1(s)\frac{1}{s} = \frac{s}{s}\frac{C_N P_N}{C_N P_N} = 1 \tag{6}$$

In equation (4) the final theorem shows that the closed loop will converge to 1 if a unit step function is applied. This means that no steady-state error has to be corrected and no integral action is needed to correct for steady-state error.

2.1.2 PDD-controller

To make the step response faster the two stable poles, located at -2 and -5 are canceled and replaced with faster poles (poles with a more negative real part). This is done with a PDD-controller, a controller with a proportional component and two derivative components. A standard derivative controller component is improper. Therefore both derivative terms are filtered using a low pass filter with a time constant τ. The expression of the controller is given as the following expression (7):

$$\begin{aligned} C(s) &= K_p + \frac{K_{d1}s}{\tau_1 s+1} + \frac{K_{d2}s}{\tau_2 s+1} \\ &= \frac{K_p(\tau_1 s+1)(\tau_2 s+1) + K_{d1}s(\tau_2 s+1) + K_{d2}s(\tau_1 s+1)}{(\tau_1 s+1)(\tau_2 s+1)} \\ &= \frac{(K_p\tau_1\tau_2 + K_{d1}\tau_2 + K_{d2}\tau_1)s^2 + (K_p\tau_1 + K_p\tau_2 + K_{d2} + K_{d1})s + K_p}{(\tau_1 s+1)(\tau_2 s+1)} \end{aligned} \tag{7}$$

Applying this controller to the system is the same as multiplying the controller with the plant (8):

$$\begin{aligned} C(s)P(s) &= \frac{(K_p\tau_1\tau_2 + K_{d1}\tau_2 + K_{d2}\tau_1)s^2 + (K_p\tau_1 + K_p\tau_2 + K_{d2} + K_{d1})s + K_p}{(\tau_1 s+1)(\tau_2 s+1)}\frac{5}{s(0.5s+1)(0.2s+1)} \\ &= \frac{(K_p\tau_1\tau_2 + K_{d1}\tau_2 + K_{d2}\tau_1)s^2 + (K_p\tau_1 + K_p\tau_2 + K_{d2} + K_{d1})s + K_p}{(0.5s+1)(0.2s+1)}\frac{5}{s(\tau_1 s+1)(\tau_2 s+1)} \end{aligned} \tag{8}$$

The dominant poles of the system will depend on τ_1 and τ_2 which have to be chosen way smaller than 0.5 and 0.2 for the system response to become faster.

To cancel the stable poles $(K_p\tau_1\tau_2 + K_{d1}\tau_2 + K_{d2}\tau_1)s^2 + (K_p\tau_1 + K_p\tau_2 + K_{d2} + K_{d1})s + K_p$ has to be equal to $(0.5s + 1)(0.2s + 1)$. By rearranging the equations K_{d1} and K_{d2} can be written as:

$$K_{d1} = \frac{0.1 - K_p\tau_1\tau_2 - 0.7\tau_1}{\tau_2 - \tau_1} \tag{9}$$

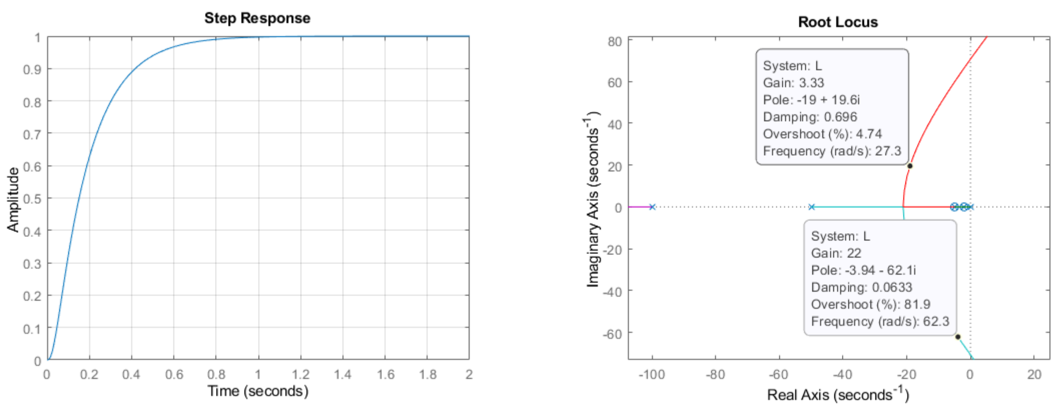
$$K_{d2} = 0.7 - K_p\tau_1 - K_p\tau_2 - K_{d1} \tag{10}$$

By choosing τ_i lower and lower the collective gain can be increased and the system can be made arbitrarily fast. However, as $\tau \to 0$, $\frac{K_d s}{\tau s+1} \to K_d s$ which is the not realizable D-term. For this reason, τ is chosen 10 times smaller than the smallest time constant of the system. From equation 9 it is clear that τ_1 and τ_2 can not be chosen to be equal. The intial controlleraparameters are biven in table 2

Parameter	Value
K_p	1
τ_1	0.02
τ_2	0.01
K_{d1}	-8.64
K_{d2}	9.31

Table 2: Parameters without collective gain

The parameters given in table 2 give a stepresponse with a settlingtime of 0.80 seconds and an overshoot of 0%. The stepresponse is given in fig. 3a.



(a) (b)

Figure 3: (a): Stepresponse with controllervalues as in table 2, (b): Root locus of openloop controlled system

2.1.3 Collective gain

Since 5% overshoot is allowed we can increase all gains (K_p, K_{d1} and K_{d2}) as long as 9 and 10 are still satisfied. To do this a collective gain is applied to the controller. A initial guess[1] for this gain is found by looking at the rootlocus trajectory of the poles. As seen in figure 3.b a gain of 3.33 should give an overshoot of 4.74%. The estimate from rootlocus is further improved with a few tries. The final gain K_c is 3.41. This gain is applied to the controller in the following way:

$$C(s) = K_c(K_p + \frac{K_{d1}s}{\tau_1 s + 1} + \frac{K_{d2}s}{\tau_2 s + 1}) \tag{11}$$

[1]The rootlocus method is not completely accurate for the given controllaw and transferfunctions

Note that multiplying the controller with a gain K_c is equal to multiplying all gains (K_p, K_{d1} and K_{d2}) with K_c. Applying this gain to the controller results in the step response shown below (figure 4). The final controller gains as well as relevant information on the step response are given in table 3

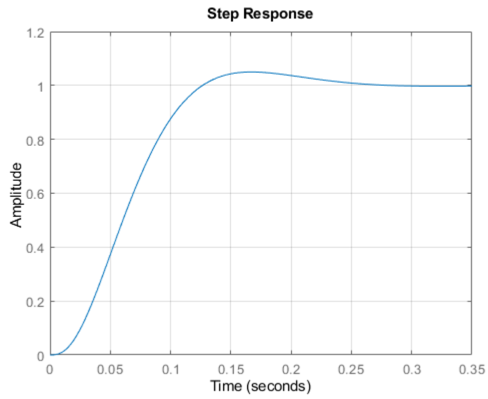


Figure 4: Step Response of final closed loop system

Step information		Controller settings	
Rise time:	0.08 s	K_p	3.41
Settling time (1%):	0.25 s	K_{d1}	-29.46
Overshoot:	5.00%	K_{d2}	31.75
Peaktime:	0.17 s	τ_1	0.02
		τ_2	0.01

Table 3: Information on controller and step response

In the bode plot of the controller applied to the system, it can be seen that the phase margin is still 60 degrees and a gain margin of approximately 20 dB. This means that the closed-loop system will definitely be stable. Also, the margins are not exceptionally high, meaning that the response is somewhat optimal.

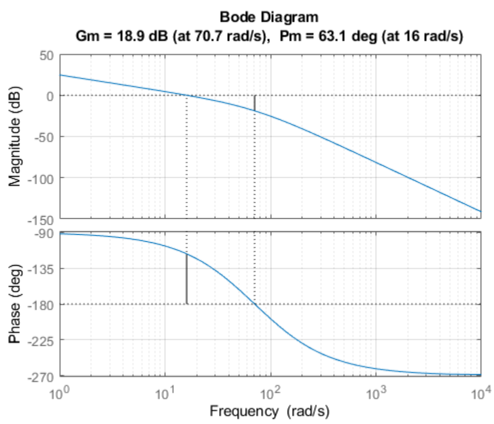


Figure 5: Bode plot of the controlled system.

If the values for τ are made smaller the gain can be increased further and the step response becomes faster. The assignment sets a goal of minimal settling time, however, an even faster response is not desirable on two main counts

Firstly, under the assumption that the output corresponds to the position of the grinding wheel in meters, the velocity that the step response corresponds to already dwarfs the maximal rapid traverse of state-of-the-art machines. For example, this [1] computer numerical controlled grinding machine of has a maximal X-axis travel speed of 40m/min. (The step response shows the movement of 1 meter in approximately one-tenth of a second which would correspond to 600m/min without taking acceleration into account.

Secondly, assuming that the control input of the system is in Volt the controller as designed above might give too much voltage to the motor. The input given by the controller to get the step response as shown in figure 4 is plotted in figure 6. For a very short period of time a voltage, that is likely higher than the rating of the motor, is applied. For a short time, applying a higher voltage to the motor is common practice but increasing the gain (and with it the input) even more can not be recommended.

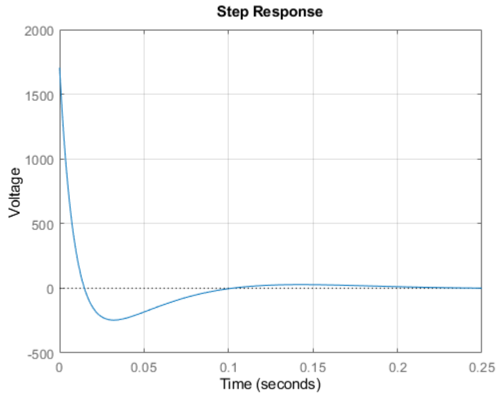
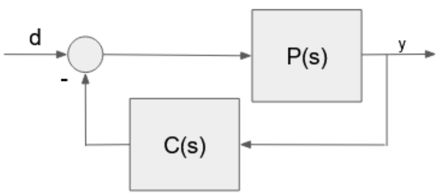


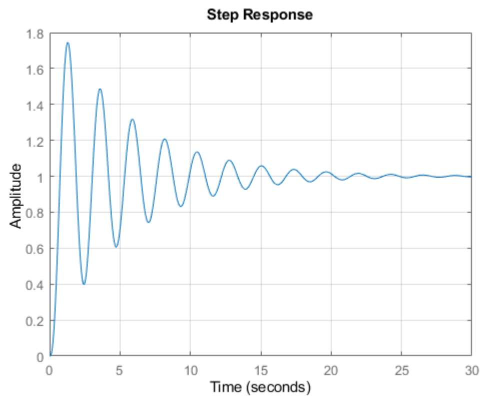
Figure 6: Input of step response calculated by $\frac{C}{1+CP}$

2.2 Question 2: Disturbance rejection

With a step disturbance on the input instead of a reference, the system can be drawn as in fig. 7a. The equation for the closed-loop system is given in equation (12). Note that with no control the closed loop transfer function is the same as in question 1. The step response of the uncontrolled system with the disturbance input gives the same result as in the previous case. The step response is given in fig. 7b. Note that the disturbance step input converges to 1, meaning that the disturbance is not rejected.



(a)



(b)

Figure 7: (a): Control scheme used in this exercise, (b): Response of the uncontrolled closed loop system to a disturbance step input.

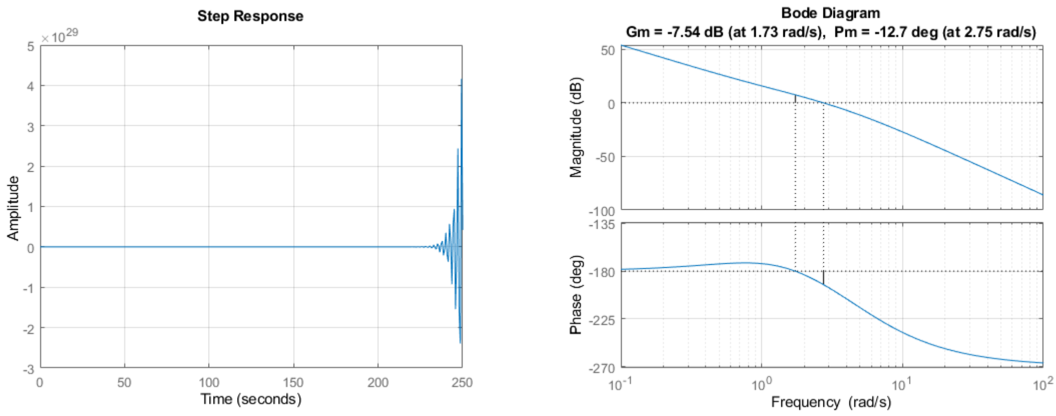
$$G_2(s) = \frac{P(s)}{1 + C(s)P(s)} \tag{12}$$

To create a controller the same stragegy as in question 1 is used. First the final value theorem is used. By writing out the closed loop equation as a function of numerators and denominators. From this function (13) it is clear that a zero steady-state error is not achieved for all controllers.

$$G_2 = \frac{\frac{P_N}{P_D}}{1 + \frac{C_N}{C_D}\frac{P_N}{P_D}} = \frac{C_D P_N}{C_D P_D + C_N P_N} \tag{13}$$

In (13) it is clear that if taking the limit of the controller denominator to zero results in zero the final value will become zero. This can be achieved by using an integrator term ($\frac{1}{s}$) in the controller. To find a good controller that uses an integrator term, first an PI controller is tested.

$$C_{PI} = K_p + \frac{K_i}{s} = 1 + \frac{1}{s} \tag{14}$$

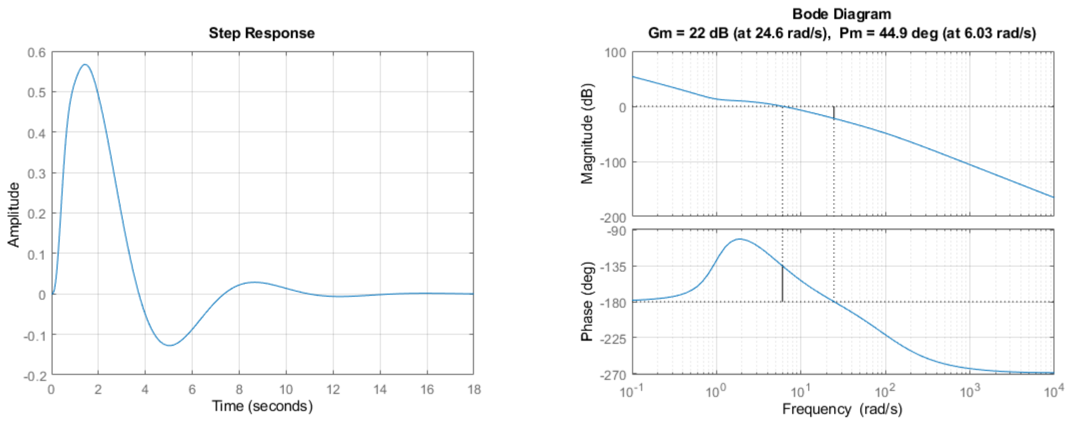


(a) (b)

Figure 8: PI

results in a negative phase and gain margin and also an unstable response. To add phase a realizable derivative term is added to the controller. The lowpass filter added to the derivative term has the same time constant as in 2.1.

$$C_{PID} = K_p + \frac{K_i}{s} + \frac{K_d s}{\tau s + 1} = 1 + \frac{1}{s} + \frac{s}{0.01s + 1} \tag{15}$$

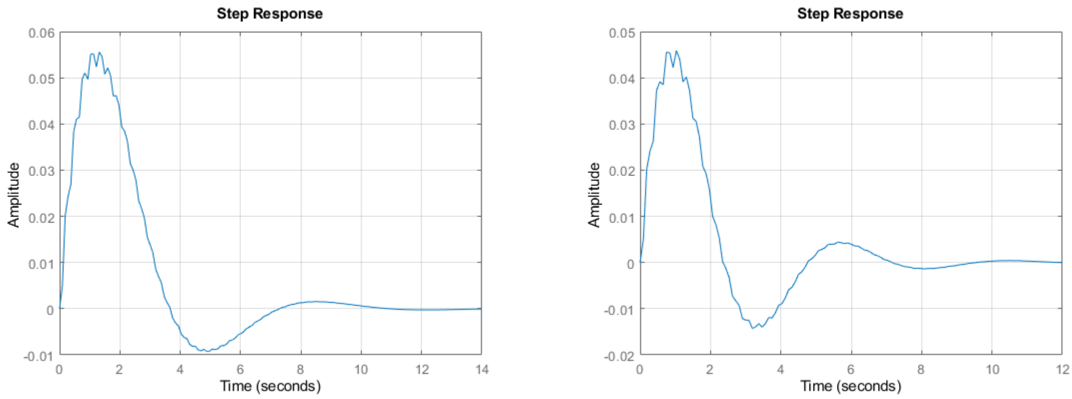


(a) (b)

Figure 9: PID controlled disturbance rejection, in (a) the disturbance step response, in (b) the bode plot of the open loop controlled system.

Through an iterative process, the controller is optimized. By looking at the gain and phase margin the controller gains are changed such that the margins stay positive and the step response becomes faster.
Increasing all the gains evenly by adding a collective gain to the system results in a response that has a similar rise time. The deviation caused by the disturbance becomes significantly smaller. The

settling that has to minimized is defined as the time it takes for the output to stay in a 1% bound of the steady state. Where the 1% threshold is 1% from the highest peak. A lower maximum value with a similar behaviour therefor results in a longer settling time. Next to minimizing the settling time the amplitude also has to be minimized.
By increasing the collective gain K_c to 10 the settling time is 58.87 seconds but the maximum value is 0.06. shown in figure 10a



(a) (b)

Figure 10: (a): Step response with collective gain. (b): Step response with final parameters.

Finally, the parameters K_p, K_d and K_i are changed to reduce the settling time. The resulting final step response is shown in figure 10b. The corresponding bode plot is shown in figure 11 and the information of the gains and are given in 4. Note that this controller results in small ossicilations in the stepresponse. This might be an issue for the real system. In section 3.2 this will be discused more.

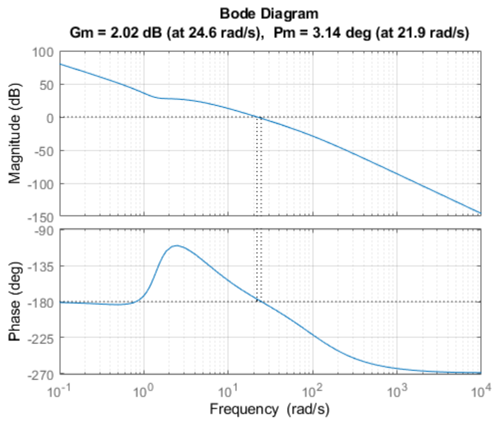


Figure 11: Bode plot of final controlled system.

Step information		Controller settings	
Peak	$5.7e-3$	K_p	10
Settling time (1%):	29 s	K_i	20
Transient time	2.95 s	K_d	10
Peaktime:	0.6 s	τ	0.01

Table 4: Information on controller and step response

3 Discrete-time Control

In this section, the system will be looked at in discrete time. First, the system is discretized. Then different kinds of discrete controllers are created and compared to the continious time controllers designed in section 2.

3.1 Question 3: Discretization

3.1.1 State space description

A linear system in continuous time can be written in the following form:

$$\dot{x}(t) = Ax(t) + Bu(t), \qquad y(t) = Cx(t) + Du(t) \tag{16}$$

To write the given system in the standard form given in (16) the following steps are taken.

$$G = \frac{Y(s)}{U(s)} = \frac{5}{s(0.5s+1)(0.2s+1)} \tag{17}$$

$$5u(t) = 0.1\dddot{y}(t) + 0.7\ddot{y}(t) + \dot{y}(t) \tag{18}$$

$$\dddot{y}(t) = -7\ddot{y}(t) - 10\dot{y}(t) + 50u(t) \tag{19}$$

By setting $y(t) = x_1(t)$ and $\dot{x}_1(t) = x_2(t)$, $\dot{x}_2(t) = x_3(t)$ the system can written in the following matrix equation.

$$\begin{aligned} \begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \dot{x}_3(t) \end{bmatrix} &= \underbrace{\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & -10 & -7 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{bmatrix}}_{x(t)} + \underbrace{\begin{bmatrix} 0 \\ 0 \\ 50 \end{bmatrix}}_{B} u(t) \\ y(t) &= \underbrace{\begin{bmatrix} 1 & 0 & 0 \end{bmatrix}}_{C} x(t) + \underbrace{\begin{bmatrix} 0 \end{bmatrix}}_{D} u(t) \end{aligned} \tag{20}$$

3.1.2 Physical meaning of the states

In the assignment, it is given that the output of the transfer function is the position of the grinding wheel. With this in mind, the output $y(t) = x_1(t)$ is set to be the grinding wheel position in meters[2]. Given this assumption, other values can be easily deduced from the state space description given in (16). The description of states and inputs is given in table 5

Variable	Unit	Description
$x_1(t)$	m	Position of grinding wheel at time t
$x_2(t)$	m/s	Velocity of the grinding wheel at time t
$x_3(t)$	m/s^2	Acceleration of the grinding wheel at time t
$u(t)$	V	Input voltage at time t

Table 5: Physical meaning of states and input

3.1.3 Discretization

From here the discrete-time system will be considered. To get this discrete-time model zero-order hold is used through the *c2d* command in matlab. The discrete-time state space description is given in the following form (21):

$$x_{k+1} = \Phi x_k + \Gamma u_k, \qquad y_k = Cx_k + Du_k \tag{21}$$

[2]Meters is chosen as unit as SI units is a reasonable assumption if no more information is available.

To choose an adequate sampling time rule of thumb is used that there should be 8-10 samples in the rise time. This corresponds to a sampling frequency

Rise time found in question 1 is 0.08 seconds. Taking 10 samples per rise-time results in a sampling time of 0.008 seconds. This sampling time results in the following discrete-time system[3] (22):

$$\begin{aligned}\begin{bmatrix} x_1[k+1] \\ x_2[k+1] \\ x_3[k+1] \end{bmatrix} &= \underbrace{\begin{bmatrix} 1 & 0.01 & 0.00 \\ 0 & 1.00 & 0.01 \\ 0 & -0.08 & 0.95 \end{bmatrix}}_{\Phi} \underbrace{\begin{bmatrix} x_1[k] \\ x_2[k] \\ x_3[k] \end{bmatrix}}_{x[k]} + \underbrace{\begin{bmatrix} 0.00 \\ 0.00 \\ 0.39 \end{bmatrix}}_{\Gamma} u(t) \\ y[k] &= \underbrace{\begin{bmatrix} 1 & 0 & 0 \end{bmatrix}}_{C} x[k] + \underbrace{\begin{bmatrix} 0 \end{bmatrix}}_{D} u[k]\end{aligned} \tag{22}$$

This discretization will be changed throughout the assignment if needed.

3.2 Question 4: Discretized controllers

3.2.1 Set point step response

In the previous question a sampling time was chosen based on stepresponse from question 1. To discretize the controller created in question one the sampling interval is used again.
The system is sampled using Zero Order Hold (ZOH). In the figure below (figure 12) both the continious time controlled system as the discritized version is given. The disctization with ZOH has a worse performance; more overshoot and a longer settling time. The explanation for the big difference in performance can be found in the method used to create the PDD controller. The PDD controller works trough pole-zero cancelation. This cancellation does not work in the discretized case since the zeros and poles do not exactly overlap.

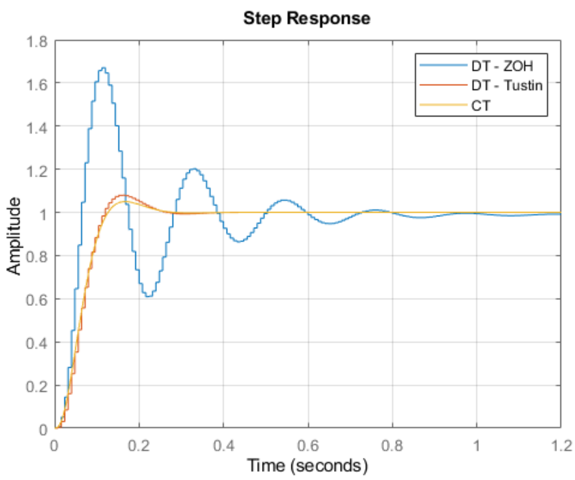


Figure 12: Step response of the continuous system (CT), the system discretized with Zero Order Hold (DT-ZOH) and the system discretized with Tustin's method (DT-Tustin)

As shown in figure 12 a better discrete version of the controlsystem can be made using Tustins approximation. Note that the system is still discritized with ZOH but the controller is discritized with Tustin with the same sampling interval.

[3]Note that all values are rounded to two decimals to increase readability

Step information		
	ZOH	Tustin
Rise time:	0.04 s	0.07 s
Settling time (1%):	1.11 s	0.25 s
Overshoot:	67.53%	8.00%
Peaktime:	0.11 s	0.16 s

Table 6: Resulting step response of discretized system with discretized controllers.

3.2.2 Disturbance step response

Using the same sampling time as before the PID controller designed in section 2.2 destabilizes the system by a disturbance input. Using the controller designed in 2.2 with a sampling time of 0.008 seconds the response looks as follows fig. 13:

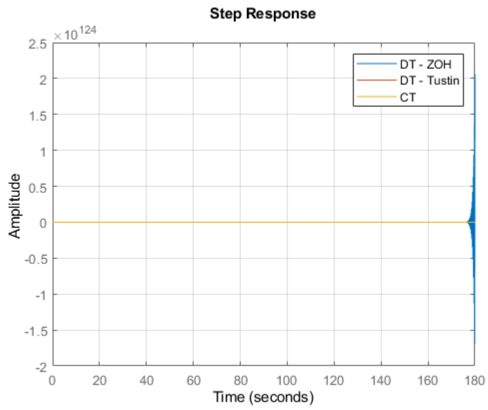
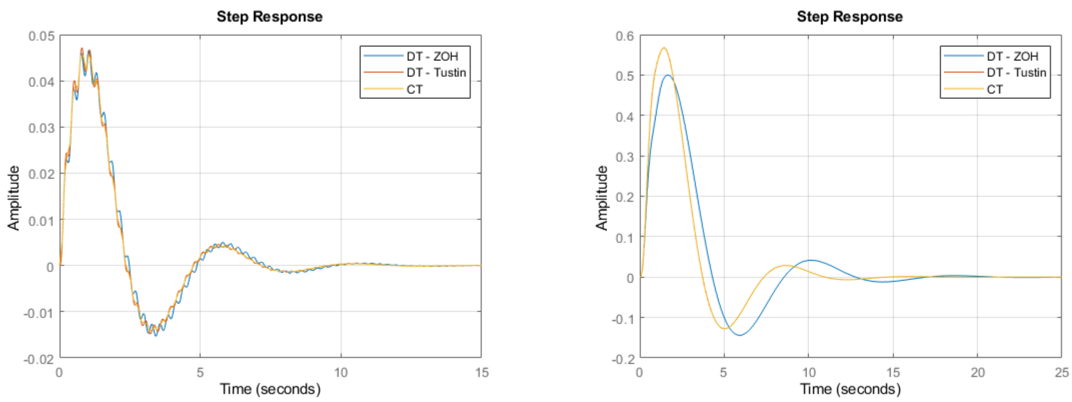


Figure 13: Disturbance rejection of the continuous system (CT), the system discretized with Zero Order Hold (DT-ZOH) and the system discretized with Tustin's method (DT-Tustin)

This issue is caused by the oscillations that are introduced by the controller. In 2.2 this is not addressed. To make the discritezed system and controller stable we can reduce the oscillations by modifying the controller gains or by choosing a faster sampling time.
Below in figure 14 the effect of the reduced sampling time and the effect of a modified PID controller is shown.



(a) (b)

Figure 14: (a): Response with sample time of 0.001 seconds (b): Response with $K_p = 1$, $K_i = 1$, $K_d = 1$ and $\tau = 0.01$

	Modified PID		Modified Sampling	
	ZOH	Tustin	ZOH	Tustin
Rise time:	0 s	0 s	0 s	0.00 s
Settling time (1%):	34.86 s	29.74 s	24.68	20.97 s
Peak:	0.50	0.57	0.05	0.05
Peaktime:	1.66 s	1.43 s	1.04 s	0.79 s

Table 7: Resulting step response of the discretized system with discretized controllers.

Comparing the step responses given above with the continuous-time from 2.2 (refer to table 4) we see that the discretized controllers lead to a higher peak value. However, with the faster sampling time of 0.001 seconds, the discretized controller has a slightly faster settling time.

3.3 Question 5: Poleplacement

Now instead of a PID controller a full state feedback gain is used. To find the gain the poles of the system are placed in a way such that the response of the system is satisfactory. This is possible because the discrete-time system is reachable which can be shown by comparing the rank of the controllability matrix W_c to the number of states. Note that the controllability matrix is dependent on the sampling interval. The rank mentioned below is calculated based on the sampling time used.

$$\begin{aligned} W_c &= \begin{bmatrix} \Gamma & \Phi\Gamma & \Phi^2\Gamma \end{bmatrix} \\ Rank(W_c) &= 3 \end{aligned} \tag{23}$$

To find the pole locations in a structured manner an estimate a second order system is created as shown in (24).

$$G(s) = \frac{k\omega_0^2}{s^2 + 2\zeta\omega_0 s + \omega_0^2} \tag{24}$$

The overshoot M_p of the step response of a second-order system can be calculated by using the following equation (25) [2].

$$\begin{aligned} M_p &= e^{-\pi\zeta/\sqrt{1-\zeta^2}} \\ \implies \zeta &= \frac{|\ln(M_p)|}{\sqrt{\pi^2 + \ln^2(M_p)}} \end{aligned} \tag{25}$$

Since a maximum overshoot is given this function can be used to calculate the damping ratio ζ

$$\zeta = \frac{|\ln(0.05)|}{\sqrt{pi^2 + \ln^2(0.05)}} = 0.69 \tag{26}$$

Using an iterative process a ω_0 is found such that the step response of the second order system has similar characteristics as the PDD controller designed in section 2.1.

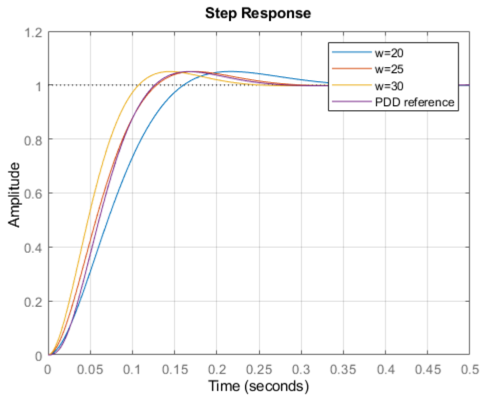


Figure 15: Different stepresponses with different ω_0 compared to the PDD control form 2.1

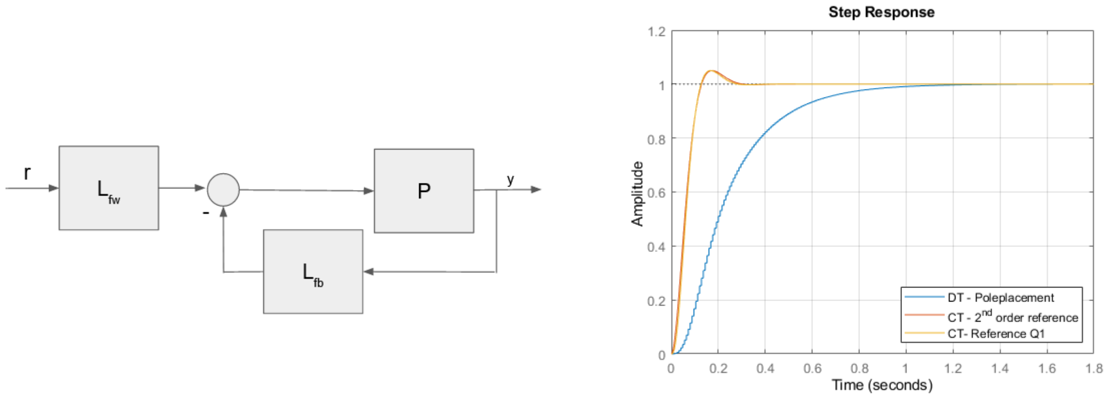
The plot above (figure 15) shows that a second order transferfunction with $\zeta = 0.69$ and $\omega_0 = 25$ gives a very similar result to the previously designed controller. Next, from the the transferfunction with $\zeta = 0.69$ and $\omega_0 = 25$ the pole locations are extracted using the procedure given in (27) and (28)

$$\begin{aligned} &z^2 + p_1 z + p_2 \\ \text{with } &p_1 = -2e^{-\zeta\omega_0 h} cos(\omega_0 h\sqrt{1-\zeta^2}) \\ &p_2 = e^{2\zeta\omega_0 h} \end{aligned} \tag{27}$$

$$\lambda_{1,2} = -\frac{1}{2}p_1 \pm \frac{1}{2}\sqrt{p_1^2 - 4p_2} \tag{28}$$

These eigenvalues give the two new poles. The goal to improve performance can be met by replacing the two slowest poles of the system with the newly calculated poles. The system is of third order, meaning that the desired pole locations are the two calculated pole locations and the pole of the system with the lowest magnitude. This is implemented in matlab using the place command to calculate a feedback gain K: *K=Place(*Φ,Γ,λ_1,λ_2,*min(pole(sys)))*.

This feedback gain is then applied to the system using $\Phi_{fb} = \Phi - K\Gamma$. To get rid of steady-state error a feedforward gain is applied to to the system using the scheme given in 16a This feedforward gain is calculated by 1 over the dcgain of the closed loop system.



(a) (b)

Figure 16: (a): Controlscheme used with L_{fb} the feedback gain calculated with the *Place* command and L_{fw} the feedforward gain defined as the inverse of the gain of the closed loop system. (b): The step response using of the feedback gain with two new pole locations.

As seen in figure 16b the resulting step response is not at all similar to the baseline controllers. This is because the PDD controller is very aggressive and the former fastest pole is now dominant. To further increase performance this third pole is changed to be faster as well. With the sample time of 0.008 seconds used the smallest pole of the system is 0.96. The complex polepair from the second order system is $0.86 \pm 0.13i$. By iteratively lowering the third pole magnitude a pole location of -0.75 gives a response very similar to the continuous time reference.

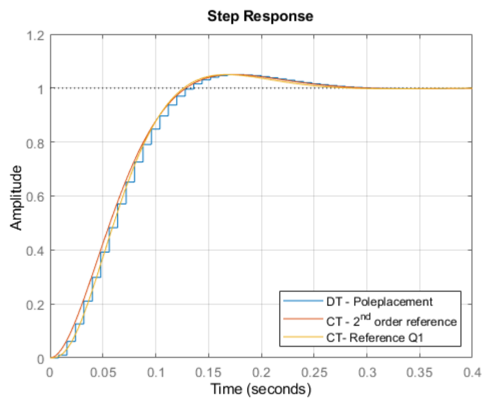


Figure 17: Step response resulting from the selected polepair

Step informations		Pole locations
Rise time:	0.09 s	$0.86 \pm 0.13i$
Settling time (1%):	0.25 s	-0.75
Overshoot:	5.01%	
Peaktime:	0.18 s	

Table 8: Information on the closed loop poles and step response

Comparing table 12 with table 3 it can be concluded that the pole placement controller almost exactly replicates the effect of the PDD controller. By bringing the pole closer to the origin, the PDD controller can be easily outperformed. The poles of the original system and the new poles are plotted in the *pzmap* shown below in fig. 18.

3.4 Question 6: Observer

In the previous question, the assumption was made that all states are available for feedback. However, as shown in 3.1 only the first state is measured. To use full-state feedback control even though only 1 state is measured an observer is constructed. Next to tracking a reference the control scheme is also used to reject a disturbance rejection. This is done by adding an integrator to the control loop

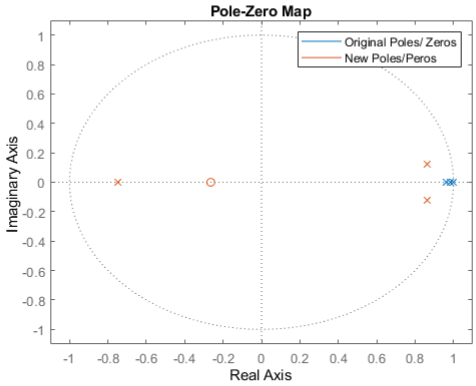


Figure 18: Pole-zero map of original and new poles zoomed in on the unit circle

that acts as a timedelay on the disturbance estimate.
To let the system track an reference trajectory while rejecting a disturbance the following control scheme below (fig. 19) is used. In this figure, the L_i block is the integrator with gain L_i. ϵ is the error between the measurement and the observer output. $\hat{d}$ is the estimate of the disturbance d and $\hat{x}$ is the estimate of the states x.
Note that, as discussed before a feedforward gain L_{fw} is added to the reference signal to get rid of the steady-state error.

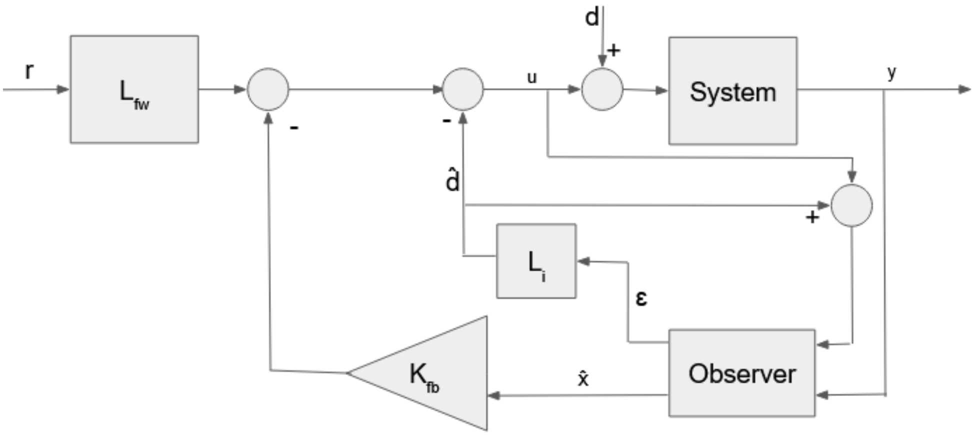


Figure 19: Observer scheme

For the feedback gain K_{fb} the same gain is used as in 3.3. The observer poles have to be faster as the estimation error should decay faster as the state error. To ensure faster observer poles the observer poles are taken 5 times smaller than the "control poles".
The disturbance estimate $\hat{d}_k$ evolves as the integral of the error between the system output and the observer output:

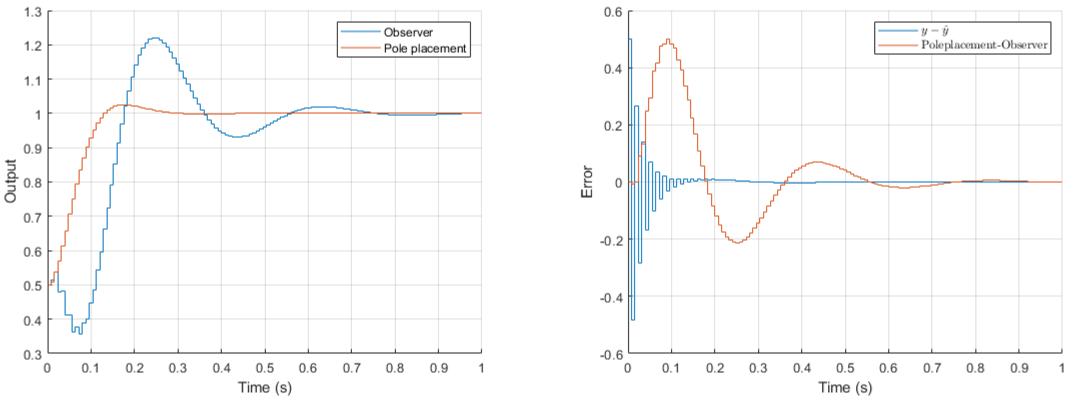
$$\hat{d}_{k+1} = \hat{d}_k + L_i(y_k - C\hat{x}_k) \tag{29}$$

The complete closed loop state space description is given below in (30).

$$\begin{bmatrix} x_{k+1} \\ \hat{x}_{k+1} \\ \hat{d}_{k+1} \end{bmatrix} = \begin{bmatrix} \Phi & \Gamma L_c & -\Gamma \\ CL_o & \Phi - L_oC - \Gamma L_c & 0 \\ L_iC & -L_iC & 1 \end{bmatrix} \begin{bmatrix} x_k \\ \hat{x}_k \\ \hat{d}_k \end{bmatrix} + \begin{bmatrix} \Gamma L_{fw} & \Gamma \\ 0 & \Gamma L_{fw} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} d \\ r \end{bmatrix} \tag{30}$$

Using no disturbance (the first input is zero) the system is given a step reference input. To compare the outputfeedback controller to the pole placement controller designed in 3.3 a initial state different than zero is used. (The observer state $\hat{x}$ are zero.) In figure fig. 20a both the poleplacement

controller and the controller with observer are given an initial condition of all states $x = 0.5$. The poleplacement controller without observer behaves the same as with zero initial condition. The controller with observer however, behaves different before converging to zero steady state error. This is because the initial condition of the observer state ($\hat{x}$) is different from the real initial state. Setting the observerstates equal to the states results in no observer error and a behaviour equal to the pole placement controller.
The observer error and the error (difference) between the full state and output feedback are given in fig. 20b.



(a) (b)

Figure 20: (a): Stepresponse of the observer system (output feedback system) and the (full state feedback) controller with the same non-zero initial condition. (b): The error between the observer states and the real states and the difference between the full state feedback controller and the output-feedback controller

Next to the reference tracking the observer constructed can be used to reject input disturbance. To make the effect of a step disturbance more visible a delayed step function with an amplitude of 10 is used as an additional input d. The disturbance is taken away to show that the system will handle a changing disturbance well. In the figure below the systemoutput is shown in the top plot. In the bottom plot the disturbance estimate $\hat{d}$ is shown.

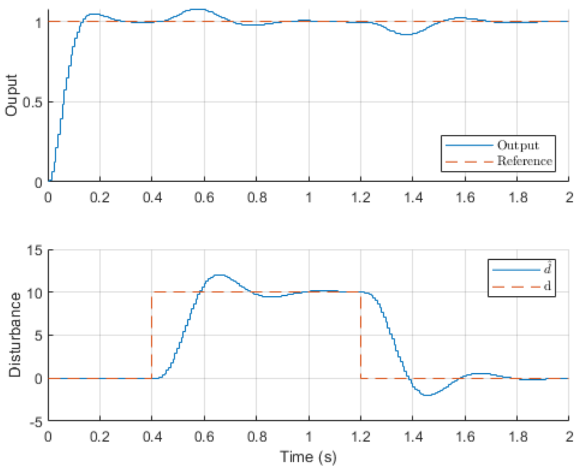


Figure 21: Disturbance rejection while tracking a reference. $\hat{d}$ is the estimated disturbance, d is the actual disturbance applied to the input.

Parameters	
Controller poles	$0.86 \pm 0.13i$ -0.75
Observer poles	$0.17 \pm 0.03i$ -0.15
Integral gain	200

Table 9: Parameters used for controller and observer.

In the the figure above (fig. 21) it can be seen that the reference is reached quickly similar to the poleplacement controller. Note that here the initial condition for all states is 0. This is not nescasry for the observer and controller to work but it gives a clearer image. In the bottom plot it can be seen that the disturbance estimate has a small overshoot. This can be changed by changing the integral gain. The poles of the controller and the observer and the integral gain used for the simulations are given in table 9.

3.5 Question 7: LQR

Now, a feedback gain is chosen using a more structured aproach. A minimization of a defined costfunction is used to get the optimal feedback gain. The cost function used is an infinite horizon quadratic cost function as given in (31).

$$\begin{aligned} J &= \min_{\underline{u}} \sum_{k=0}^{\infty} (x_k^\top Q x_k + u_k R u_k) \\ s.t. \quad x_{k+1} &= \Phi x_k + \Gamma u_k \end{aligned} \tag{31}$$

Where x_k is the state vector at timestep k, Q is the weight matrix that determines the cost of the states, u_k is the input (scaler) at timestep k and R is the weight in the input. The minimizing argument $\underline{u}$ is the sequence of inputs. The minimization is subject to the state evolution given by x_{k+1}.
To find a gain that minimizes the cost function above a discrete algebraic Riccati (DARE) is solved.

$$\begin{aligned} K_{lqr} &= (\Gamma^\top P \Gamma + R)^{-1}(\Gamma^\top P \Phi) \\ with \quad P &= \Phi^\top P \Phi - \Phi^\top P \Gamma(\Gamma^\top P \Gamma + R)\Gamma^\top P^\top \Phi + Q \end{aligned} \tag{32}$$

In the Matlab implementation the the gain K_{lqr} is found using the *idare* function. Steady-state error is removed by dividing the reference by the dcgain of the closed loop system. Using the following implementation scheme:

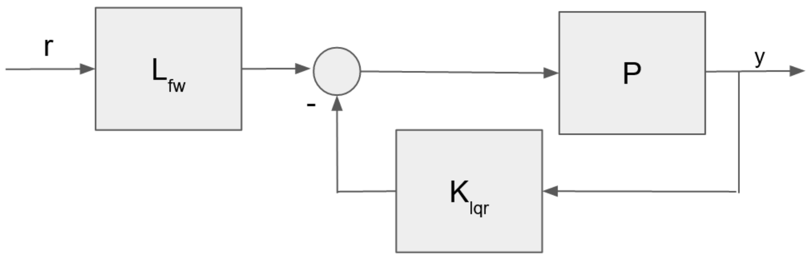
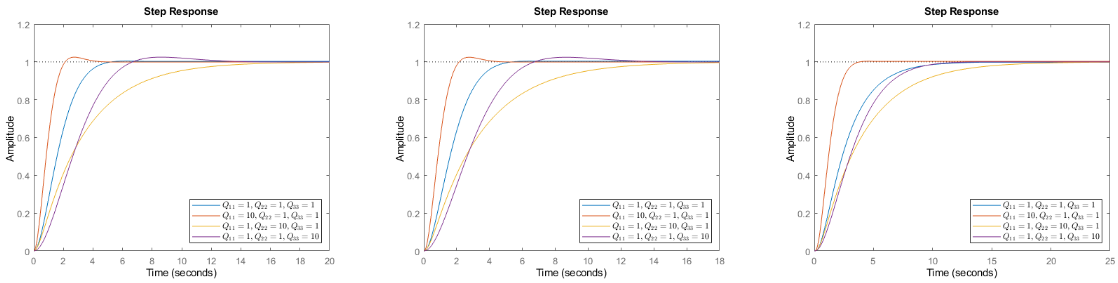


Figure 22: LQR control scheme

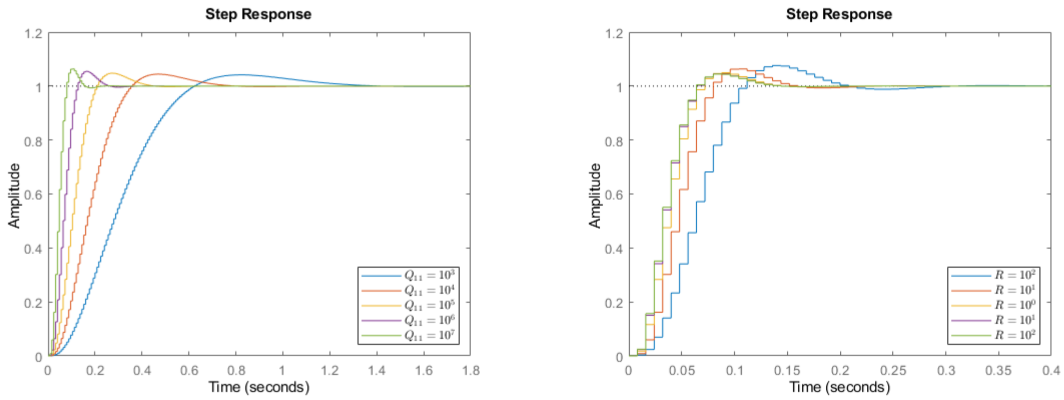
To find the optimal Q and R matrices an iterative method is used. First, the Q matrix is initialized as identity, then one entry on the diagonal is changed to 10. The different resulting step responses are given in fig. 23. To investigate the influence of the input weight R the different Q matrices are used with 3 different values of R.



(a) (b) (c)

Figure 23: Step response with different LQR-tuning weight Q. In (a): R=0.01, (b): R=1, (c): R=100

From these figures, it can be concluded that the first entry in the Q matrix has the most influence on the step response. This is expected as the first entry (Q_{11}) influences the cost resulting from the error of the first state. The output of the system is only the first state as well.
The influence from the input weight R is most clear in figures 23a and 23c A smaller value leads to a faster response, which is to be expected as a higher input (inputgain) leads to a higher cost but also a faster response.



(a) (b)

Figure 24: (a): R=1, Q_{11}= varying, (b):$Q_{11} = 10^7$, R= varying

After veryfying the results of changing weights in fig. 23 the choice is made to only change Q_{11} and R. To find an optimal value for these two weights different step responses are given above in fig. 24. The final selection of weights and the infromation of the resulting step response are given in table 10. The stepresponse, shown in fig. 25 shows the stepresponse. With the given parameters it is slightly faster than the pole placement approach. In a later section, the control input will be discussed as an additional comparison.

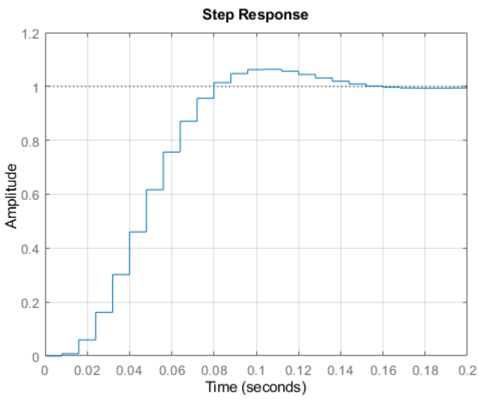


Figure 25: Step Response of final closed loop system with LQR control

Step information		Weights	
Rise time:	0.05 s	Q_{11}	10^7
Settling time (1%):	0.14 s	Q_{22}	1
Overshoot:	6.37%	Q_{33}	1
Peaktime:	0.10 s	R	1

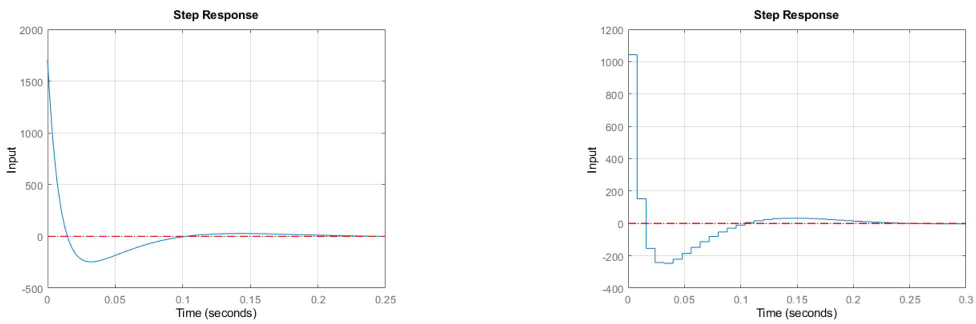
Table 10: Information on the controller and step response

3.6 Question 8: Input saturation

Usually, the input that can be applied to the system is limited, either because of safety precautions or because the actuator is limited. In the following part the input will be limited to ± 1. First, the controllers designed in previous questions are tested to see if the input is outside of the ± 1 tolerance.

3.6.1 PID controller reference tracking

The PID-type controller for reference tracking is optimized to a quick response. As already discussed at the end of section 2.1 this means that an enormous input is needed. In the figure below (figure 38) it is confirmed that the input does break the $\pm$ 1 bound for both the continuous time and its discrete-time counterpart.



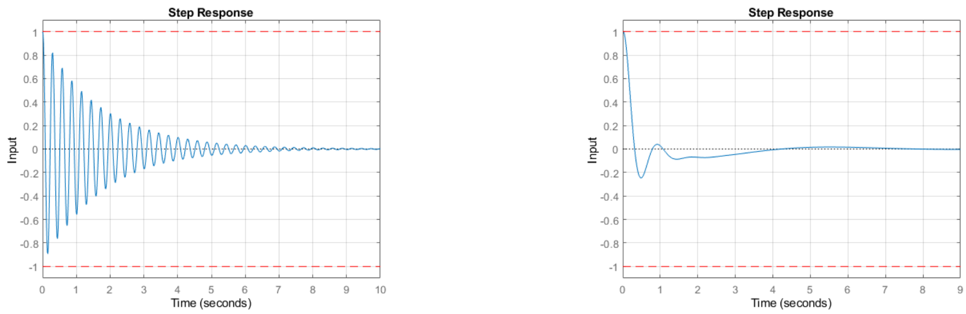
(a)

(b)

Figure 26: (a): Continuous-time PDD controller output, (b): Discrete-time PDD controller output

3.6.2 PID controller disturbance rejection

The PID controller for disturbance rejection designed in question 2.2 is not as fast as the reference tracking controller. As can be seen in 2.2 the response of the controlled system has small but fast oscillations these oscillations are clearly present in the control input (figure 27a). In figure 27b the control input resulting from the modified discrete time PID controller is shown. (This control input corresponds to the response given in figure 14b). The redesigned PID controller does not result in oscillations of the control input which is likely preferred for the system at hand.



(a) (b)

Figure 27: (a): Continuous time PID controller output, (b): Discrete time PDD controller output

3.6.3 Pole placement

The poleplacement controller designed in section 3.3 is created to mimic the effect of the fast PDD controller for reference tracking. The control input given in figure 28 shows that the control input violates the given bounds of ± 1. For the output feedback controller designed in question 3.4 the same pole placement controller is used. The control input resulting from the output feedback controller with the observer will be very similar to the pole placement control input (depending on initial conditions and disturbance).

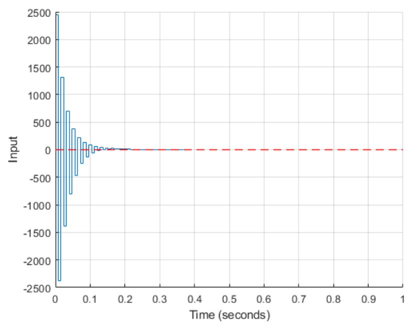


Figure 28: Control input from pole placement controller

3.6.4 LQR

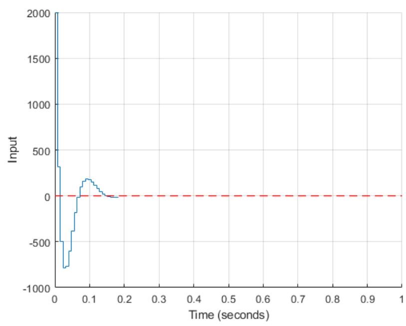
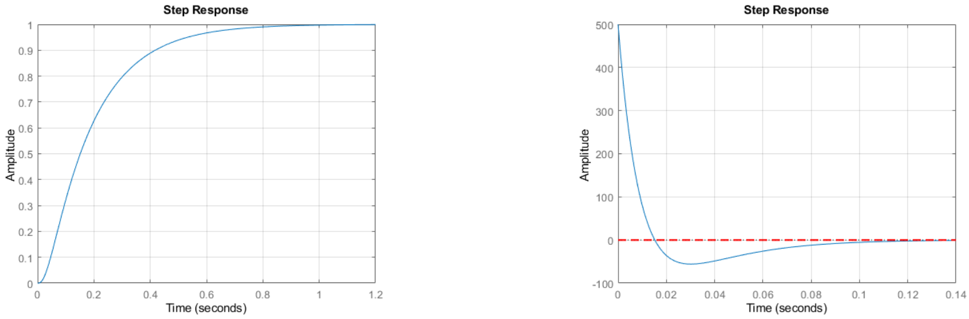


Figure 29: Caption

The LQR controller designed in question 2.2 has a similar response time as the pole-placement controller. However, since the cost function includes the input cost the poles placed by the LQR controller are better optimized to also limit the control action over time. This can be seen in figure 29 as the input exceeds the bounds in a similar way as the pole placement controller, however, the control input does not oscillate and follows a more smooth curve before settling to zero.

3.7 Question 9: Discrete PID with saturation

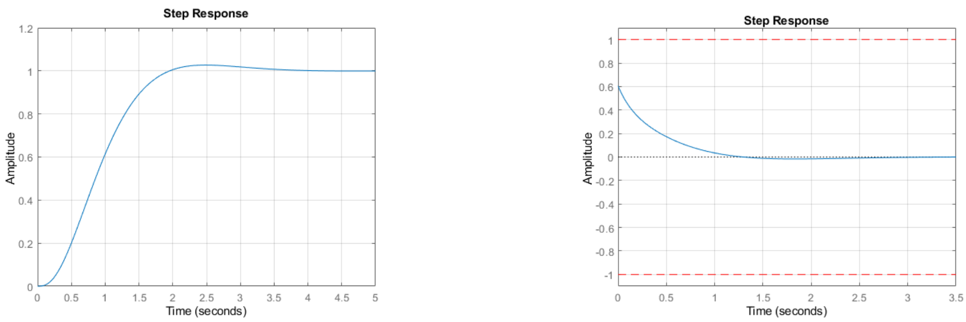
To redesign the PDD controller for reference tracking the same procedure as in question 2.1. The controller gains found previously are used as starting point. To reduce the control input that the PDD controller generates, first the collective gain is removed. The continuous time step response is given in figure 30a. The corrosponding control input is given in figure 30b.



(a) (b)

Figure 30: PDD controller without the collective gain. In (a): the stepresponse, in (b): The corresponding control input.

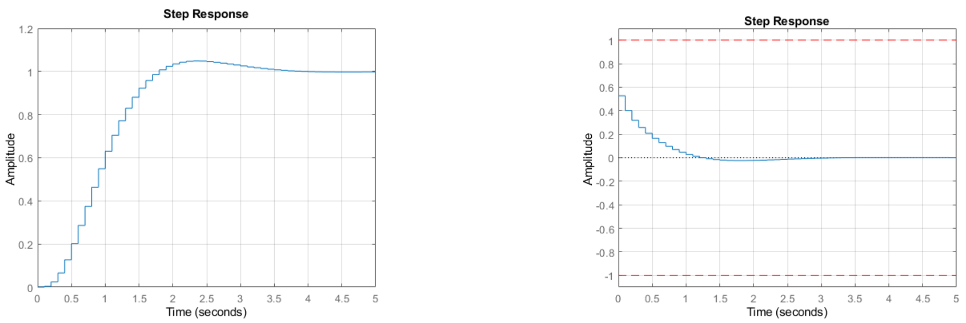
The plot above clearly shows more change is needed to get the controlinput in to the saturation. The next step is changing the filterterms that give the location of the new poles of the system. (Refer to (8)). The new time constants of the derrivative componants are chosen as $\tau_1 = 0.2$ and $\tau_2 = 0.19$. Then the collective gain is reduced below 1 to make the final adjustments such that the overshoot is below 5% and the input is within the given limit.



(a) (b)

Figure 31: Controller with changed filterterms and a collective gain below 1. In (a): the stepresponse, in (b): The corresponding control input.

Similar to section 3.2 the continuous time PDD controller is discretized using Tustin's method. Since the response is slower a sampling time of 0.1 seconds is used. Below the discrete stepresponse and discrete control input are given.



(a) (b)

Figure 32: (a): Discrete time stepresponse, (b) Control input of discrete-time system.

Step information		Controller settings	
Rise time:	1.10 s	K_p	0.23
Settling time (1%):	3.50 s	K_{d1}	-2.78e-15
Overshoot:	4.85%	K_{d2}	0.31
Peaktime:	2.30 s	τ_1	0.2
		τ_2	0.19

Table 11: Information on controller and step response

As seen in section 3.6.2 the PID controller for disturbance rejection does not generate control inputs that are outside of the given limit. Therefor this controller is not redesigned in this section.

3.8 Question 10: Poleplacement with saturation

For the poleplacement controller the same design method is used as before. The step continious time step response given in figure 32a is used as reference example.
Using the equations previously used in section 3.3 a second-order reference function is created to generate two of the three poles. Since the response found in question 3.7 is slower a sampling time of 0.1 seconds is used for the discrete-time pole-placement controller. The reachability condition given in (23) is still met with this adjusted sampling time.

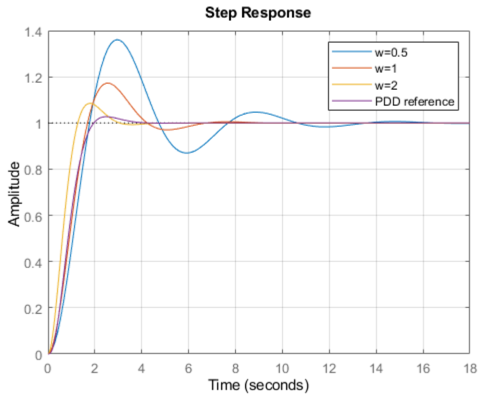
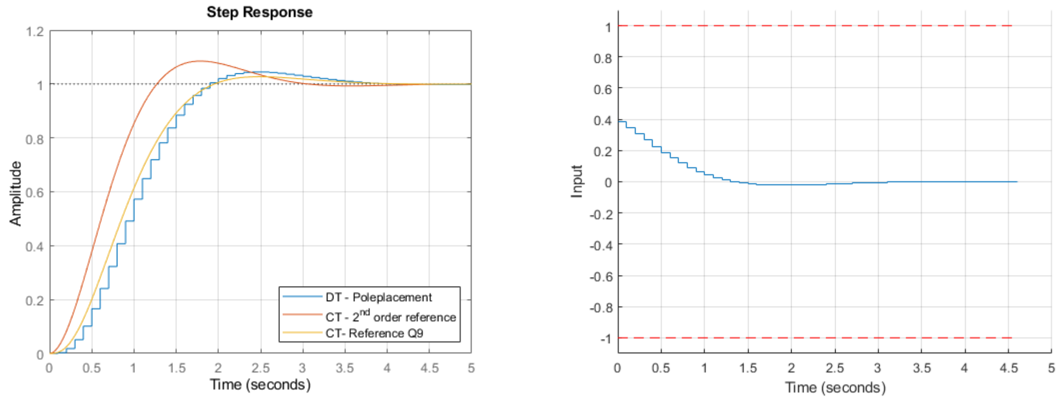


Figure 33: Different ω_0 values.

In the figure above (figure 33) an value for ω_0 is found such that the second-order system step response behaves similarly to the PDD controller redesigned in the previous question. Using (27) two pole locations are calculated. The third pole location is the fastest pole of the original system. Without changing this last pole location the third-order system behaves very similarly to the PDD-controlled reference system. The input (given in fig. 34b) stays within the given bounds.



(a) (b)

Figure 34: Step response resulting from the selected polepair

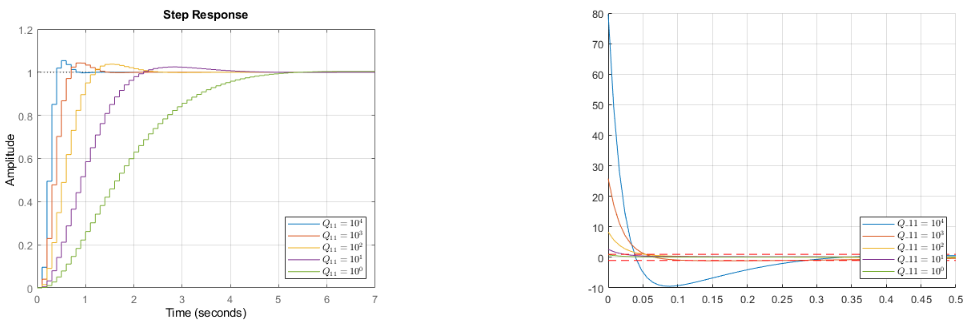
In table 12 the inormation on step response is given next to the pole locations of the controlled system. The settling time is slightly faster than the PDD-control baseline. The rise time however, is slightly higher as well.

Step informations		Pole locations
Rise time:	1.20 s	$0.86 \pm 0.13i$
Settling time (1%):	3.30 s	0.61
Overshoot:	4.45%	
Peaktime:	1.04 s	

Table 12: Information on the closed loop poles and step response

3.9 Question 11: LQR with saturation

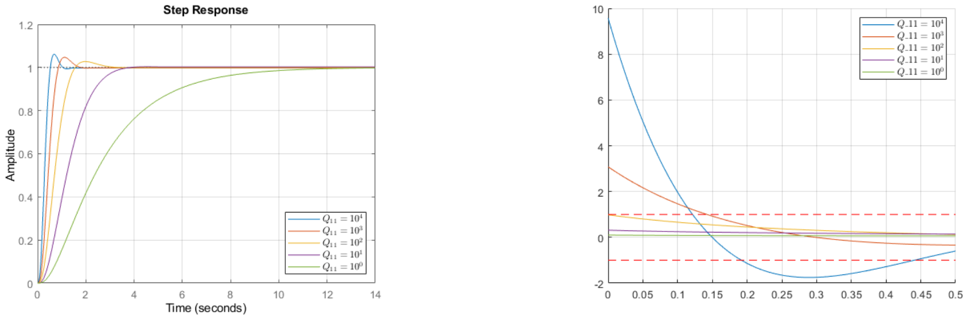
To limit the controlinput for the lqr controller between -1 and 1 an iterative process is used. In the figure below (figure 35) input weight R is set to 1 and similar to section 3.5 the weight on the first state Q_{11} is changed. We see that a lower weight on the state makes the step response slower, but also decreases the control input.



(a) (b)

Figure 35: In (a): the stepresponse and in (b): the corrosponding control input. Inputweight $R = 1$, Q_{11} is varying.

The same test is done using an inputweight R of 100. In the figure below (fig. 36a) a similar but slightly slower step response is shown. In fig. 36b a we see that for $Q_{11} = 100$, $Q_{11} = 10$ and $Q_{11} = 1$ the control input stays bounded between -1 and 1.



(a) (b)

Figure 36: In (a): the stepresponse and in (b): the corrosponding control input. Inputweight $R = 100$, Q_{11} is varying.

The conclusion drawn from the figure above is used to make final adjustments to the weights. The final weights are given in table 13. The stepresponse with the corrosponding control input of LQR controller are given in fig. 37.

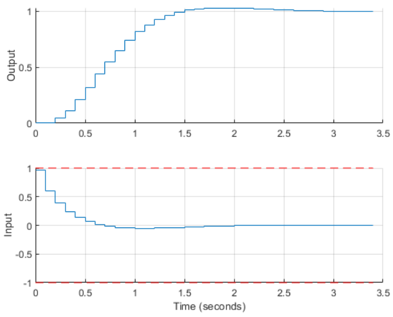


Figure 37: Resulting Step response

Step information		Weights	
Rise time:	0.90 s	Q_{11}	100
Settling time (1%):	2.60 s	Q_{22}	1
Overshoot:	3.10%	Q_{33}	1
Peaktime:	1.80 s	R	65

Table 13: Information on controller and step response

As this final response is significiantly slower than before the rise samplerate is decreased to 0.1 seconds. This change is made before the final LQR controller is designed. In table 13 the information of the stepresponse of the final system is given. Comparing this stepresponse to previous designed controllers it can be concluded that the settling time is slightly faster as previous designed controllers, even though the overshoot is well below the maximum of 5%.

3.10 Question 12: Steady state error

As seen in previous questions, all controllers achieve zero steady state error. This is done in two different ways. For the PID-like controllers an integrator term is used to get rid of steady state error if it occurs. The need for the integrator term can be checked by using the final value theorem or just by checking if the system has a pure integrator from itself (which is the case in the first question).

Next to adding an integrator term another way used to get rid of steady state error is by using a feedforward gain. This is used in the poleplacement controller, observer and LQR-controller. The feedforward term is the inverse of the DC gain of the closed-loop system.

For the disturbance rejection in the observer design a integrator is used to get an estimate for the disturbance such that the disturbance can be adjusted.

3.11 Question 13: Time delay

Finally, a time delay on the controlinput is added to the system. This delay is exactly equal to one sample. Throughout the exercises the samplingtime is adjusted multiple times. The added delay is adjusted accordingly.
Adding the delay to the system can be done in mulitple different ways. One way is to include the delay in the given continious time transferfunction by adding a time shift. This results in the following expression (33).

$$G_d = e^{-\tau s} \frac{5}{s(0.5s+1)(0.2s+1)} \tag{33}$$

Where τ is the time delay, equal to the sampling time.
Discritizing this expression using the metholodgy from section 3.1 results in a discrete time system with a delay of one sample. Another way of incorporating the time delay is by using a state space description (34):

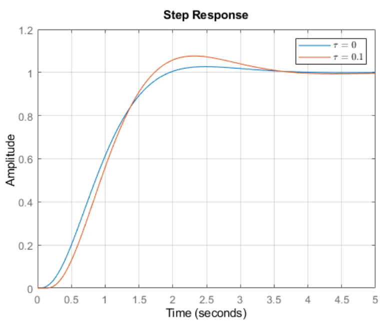
$$\begin{aligned} x_{k+1} &= \Phi x_k + \Gamma u_{k-1} \\ y_k &= C x_k + D u_{k-1} \end{aligned} \tag{34}$$

This state space description can be rewritten such that the timedelay on the input happens internally using an augmented system (35):

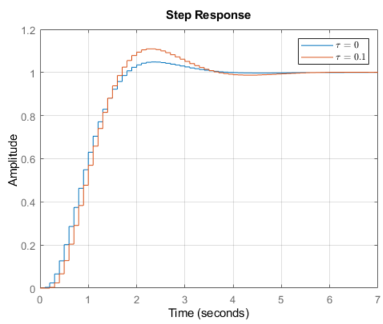
$$\begin{bmatrix} x_{k+1} \\ u_k \end{bmatrix} = \begin{bmatrix} \Phi & \Gamma \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_k \\ u_{k-1} \end{bmatrix} + \begin{bmatrix} 0 \\ I \end{bmatrix} u_k \tag{35}$$

3.11.1 PDD reference tracking

Applying the input delay to the controller for tracking designed in question 9. Results in a slightly different response. In the figure below it can be seen that both in continious and discrete time the overshoot of the delayed system is a bit higher than the original. The sampling interval used in question 9 is 0.1 seconds, meaning that for this controller a delay of 0.1 seconds is used.



(a)



(b)

Figure 38: (a): Continuous-time PDD controller, (b): Discrete-time PDD controller, both with ($\tau = 0.1$) and without ($\tau = 0$) delay

The overshoot seen in the figures above is approximatly 10%. The controller is modified such that the overshoot stays below the required 5%.

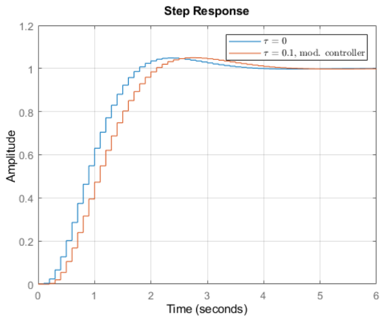


Figure 39: Discrete time PDD controller, with delay and modified controller in red, original in blue

Step information		Controller settings	
Rise time:	1.30 s	K_p	0.19
Settling time (1%):	4 s	K_{d1}	-5.25e-16
Overshoot:	5.00%	K_{d2}	0.06
Peaktime:	2.70 s	τ_1	0.2
		τ_2	0.19

Table 14: Information on controller and step response

3.11.2 PID Distrubance rejection

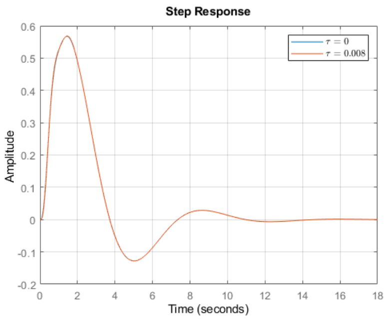


Figure 40: Discrete time PID controller, with delay and modified controller in red, original in blue

The discrete time distrubance rejection controller designed in section 3.2 has now apparent caused by the time delay. However, on closer inspection, it can be seen that the delayed system does in fact lags behind one sample.

3.11.3 Poleplacement

To apply the input delay to designed poleplacement controller the same gain is used as in section 3.8. Instead of the original system, this gain is applied to the system with the delay of one sample. This results in the poles being placed at completely different locations than before and the system not reacting in the same way at all.

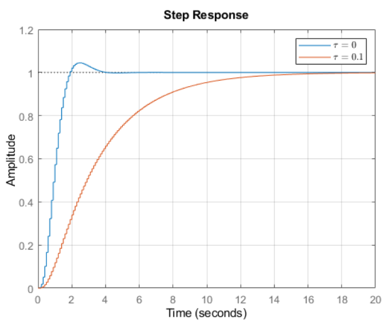


Figure 41: Step response of the delayed system with the same gain as before.

To adjust for the time delay the same poles as in section 3.8 are used to calculate the new feedback gain K. This way the poles of the delayed closed-loop system will be the same as the poles of the closed-loop system without delay. The resulting gain is shown below in table 15. In fig. 42 the original and adjusted step responses are shown. Note that because the closed loop poles are the same, the response of the delayed system is almost identical to the original.

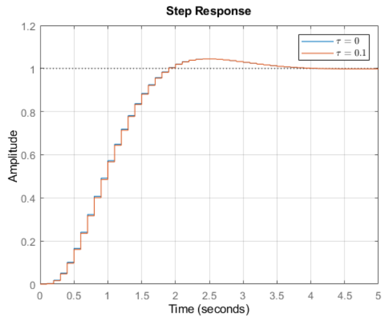


Figure 42: Step response with adjusted gain

Orginal gain	Adjusted gain
0.38	2.40
0.16	0.80
0.02	0.11

Table 15: Original and adjusted Gain.

3.11.4 LQR

To add the delay to the lqr-controlled system the same procedure is used for pole-placement. The LQR gain is applied to the delayed system resulting in a very different response as the response without delay.

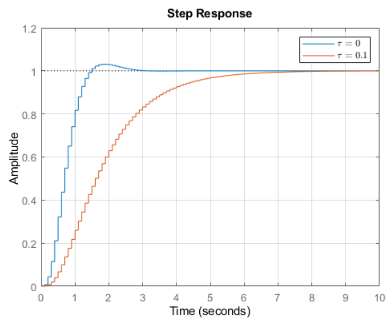


Figure 43: Step response with the same gain as in section 3.9

To get the delay system to behave like the previously system the weight on the states is changed. The final values of the used to generate the LQR gain are given in table 16. The original and redtuned stepresponse are given in fig. 44

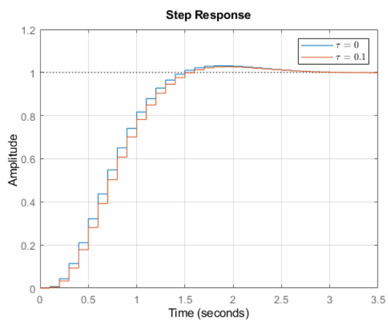


Figure 44: Step response with adjusted lqr gain

Adjusted weights	
Q_{11}	10^5
Q_{22}	1
Q_{33}	1
R	65

Table 16: Adjusted weight matrices.

4 Conclusion and final notes

Attached in a separate file is the complete Matlab script. Note that all Matlab code is in just two files. One very large file contains all questions, controller designs, and plot functions back to back. The reason that everything except the function to create the PID-like controller is that almost all questions use information from the questions that came before it. Running the entire script will open over 50 figures. These will automatically close but it might take a few seconds. After running the file all questions can be run separately by using *Run Section*.

Comparing all designed controllers it is difficult to say which performs better because all are tuned to behave in a similar way. All controllers were designed to match the given tasks. It should be mentioned that all controllers can be optimized almost indefinitely to perform even better within the given bounds.

All controllers achieve zero steady-state error. This is because the feedforward gain is applied to the reference in most of the discrete-time controllers as described in section 3.10. This means that the grinding wheel will reach the exact intended location over time. However, we should note that it reaches this location only up to limited accuracy (of 1%) in the settling time. 1% Seems way to large for the accuracy of a grinding machine which means additional design parameters are needed before a final controller can be integrated into the real system.

References

[1] United Grinding. *PLANOMAT HP 408 technical data.* Accessed: 29-11-2022, Available: https://www.grinding.com/en/products-and-services/all-products/detailed-view/product/planomat-hp/.

[2] Karl Johan Astrom Richard M. Murray. *Feedback Systems: An introduction for Scientists and Engineers.* Princeton University Press, Princeton, 1988.