




Homework 1 in EL2450 Hybrid and Embedded Control Systems

Gustav Gybäck
950214-0537
gyback@kth.se

Albin Larsson Forsberg
960527-8671
albinfor@kth.se

January 29, 2021

Task 1

The purpose of the tap is to model any existing leakage. By changing the tap gain, water will be removed from the tank that is the  not fed into tank two, but instead gets removed from the system. This way of modeling the leakage is possible at small gain values, the model would otherwise lose its integrity.

Task 2

```
1 s = tf('s');  
2 uppertank = (k_tank)/(1+Tau*s);  
3 lowertank = (gamma_tank)/(1+gamma_tank*Tau*s);  
4 G = uppertank*lowertank; % Transfer function to lower tank
```

Task 3

The reference block starts a step function, which is the reference signal, at time $t = 25$ and sets the reference value to 10. The purpose of the u_{ss} and y_{ss} blocks is to set the linearization point around which the system has been linearized. The ss stands for steady state. It is a heaviside function with a step of 10 at $t = 25$.

Task 4

```
1 [K_pid, Ti, Td, N] = polePlacePID(chi, omega0, zeta, Tau, gamma_tank,  
    k_tank);  
2 F = K_pid*(1+(1/(Ti*s))+((Td*N*s)/(s+N)));  
3 sys = F*G/(1+F*G); % Closed Loop system
```

Task 5

Both setting 2 and 3 provide ok performance. However, setting 2 fails in the setting time. Setting 3 does provide slightly worse rise time but due to the dampening being harder,

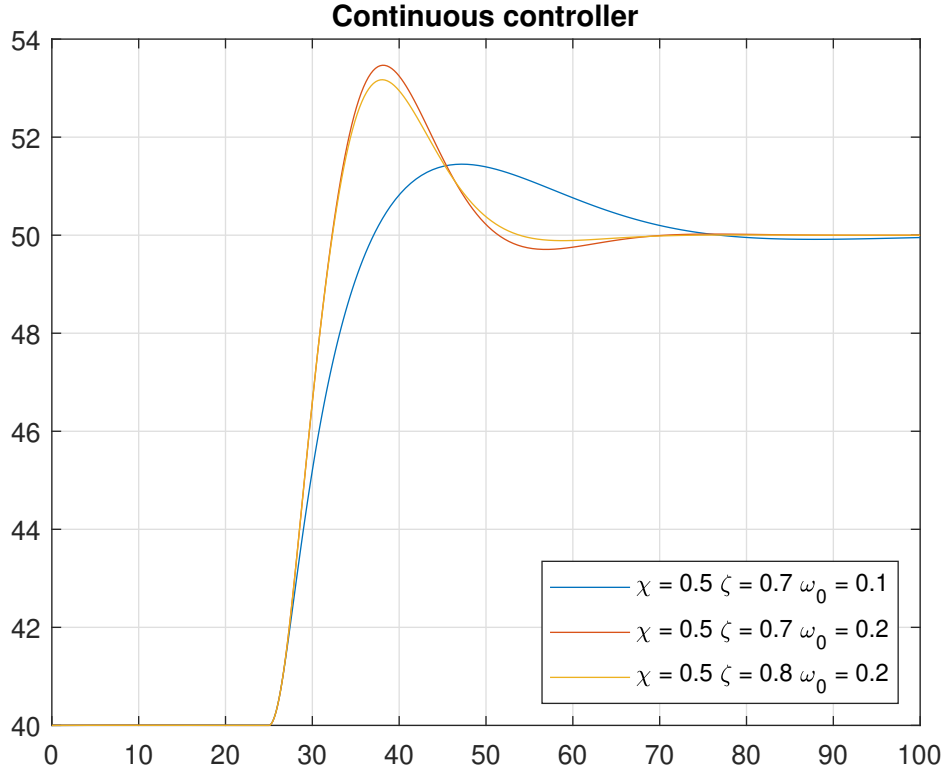


Figure 1: Performance of a continuous controller

it has a lower settling time. The result from the different runs is seen in table 1. The rise time is mainly affected by the choice of natural frequency of the system. Plots of the step response is seen in figure 1

params	χ	ζ	ω_0	T_r	M	T_{set}
1	0.5	0.7	0.1	8.11 s	15.04 %	46.27 s
2	0.5	0.7	0.2	4.92 s	34.68 %	36.57 s
3	0.5	0.8	0.2	4.94 s	31.72 %	26.39 s

Table 1: Results from simulation with different parameters

Task 6

The crossover frequency is where $|G(s)F(s)| = 1$. The value that we are looking for can easily be solved for by doing the substitution $s = \omega i$ and solve for the correct value of ω_c . We get three different values of cross over frequency. In this assignment they are solved by the command "margin" in matlab. The crossover frequencies are:

- For setting 1: $\omega_c = 0.224$ rad/s
- For setting 2: $\omega_c = 0.343$ rad/s
- For setting 3: $\omega_c = 0.362$ rad/s

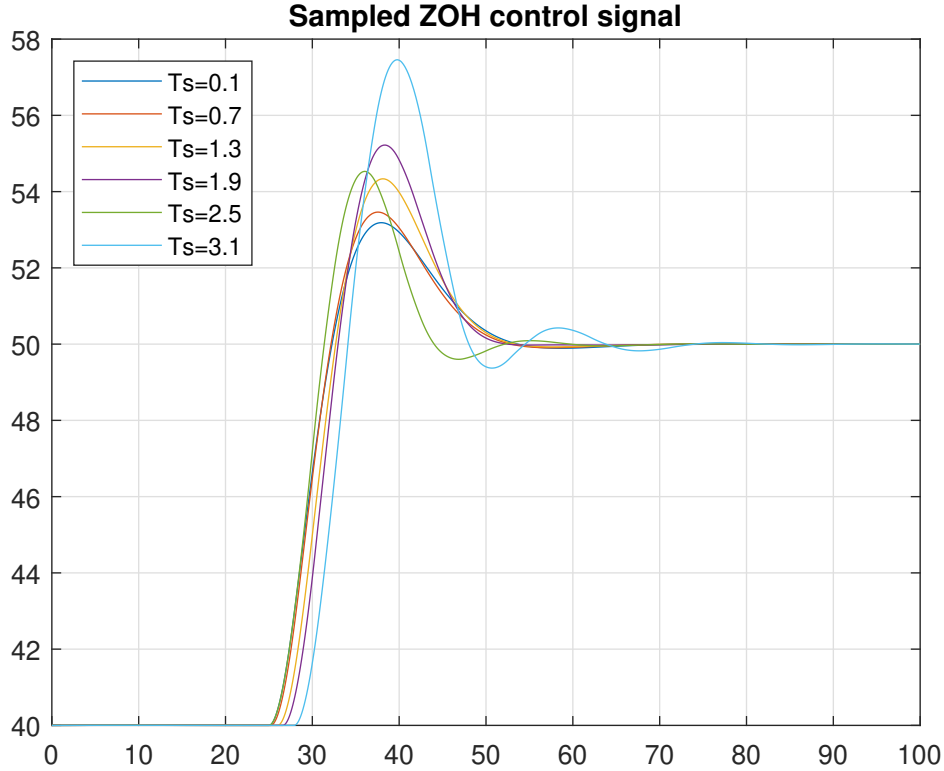


Figure 2: Performance of a sampled control signal at different sampling periods

Task 7

According to [WAA] it is recommended to keep the sampling time at most at $\frac{0.1}{\omega_0}$. For the natural frequency of ω_0 , this suggest that the sampling period should not exceed $T_s = 0.5$. This is verified by showing that the system maintains a satisfying performance until about a sampling period of 0.7 seconds for the third scenario. The overshoot gets too large at that sampling period. Results can be seen in figure 2 and 3.

Task 8

The discretized system does not perform similar to the sampled continuous controller. As described in [WAA] the discretized controller will have a higher overshoot than the continuous sampled controller, which is a fact that we can observe in the simulation. This happens at larger values of sampling time. The difference could be because the controller in this case is purely discrete, while it in the continuous case has access straight away to the continuous signal and it is only the output from the controller that is being sampled. The settling time makes a big jump at around $T_s = 0.7$, but fails already at $T_s = 0.3$. An interesting observation is that the overshoot actually gets better until it catastrophically fails at higher sampling periods. The limiting factor in this case is not the overshoot, as it was in the zero order hold continuous controller, but is instead the settling time. Results can be seen in figure 4 and 5.

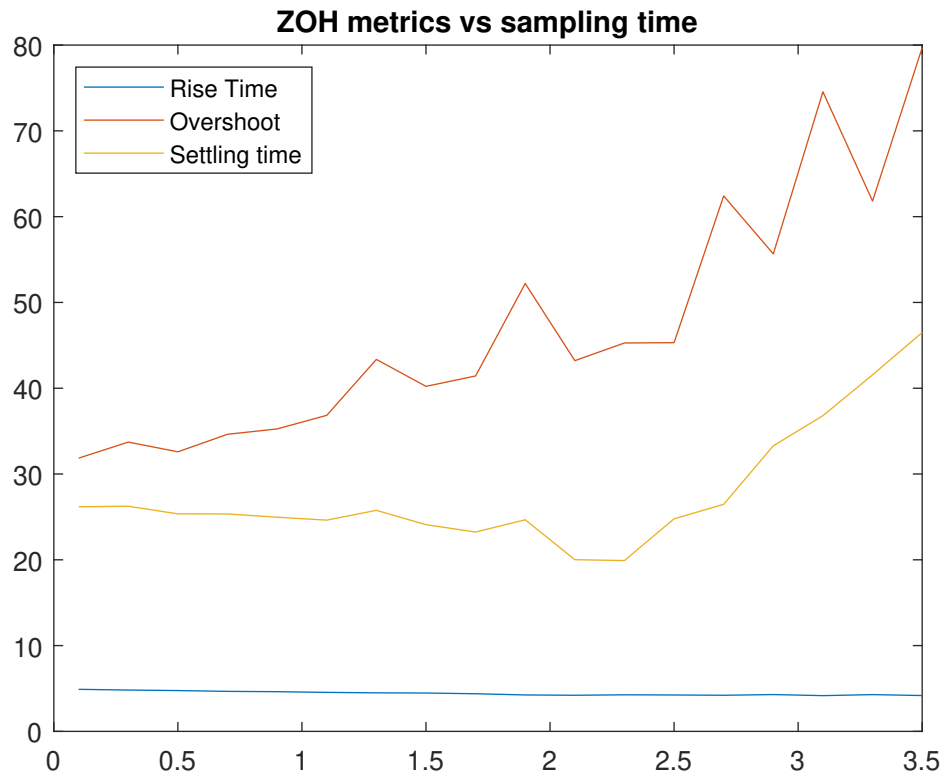


Figure 3: Different performance metrics at different sampling periods

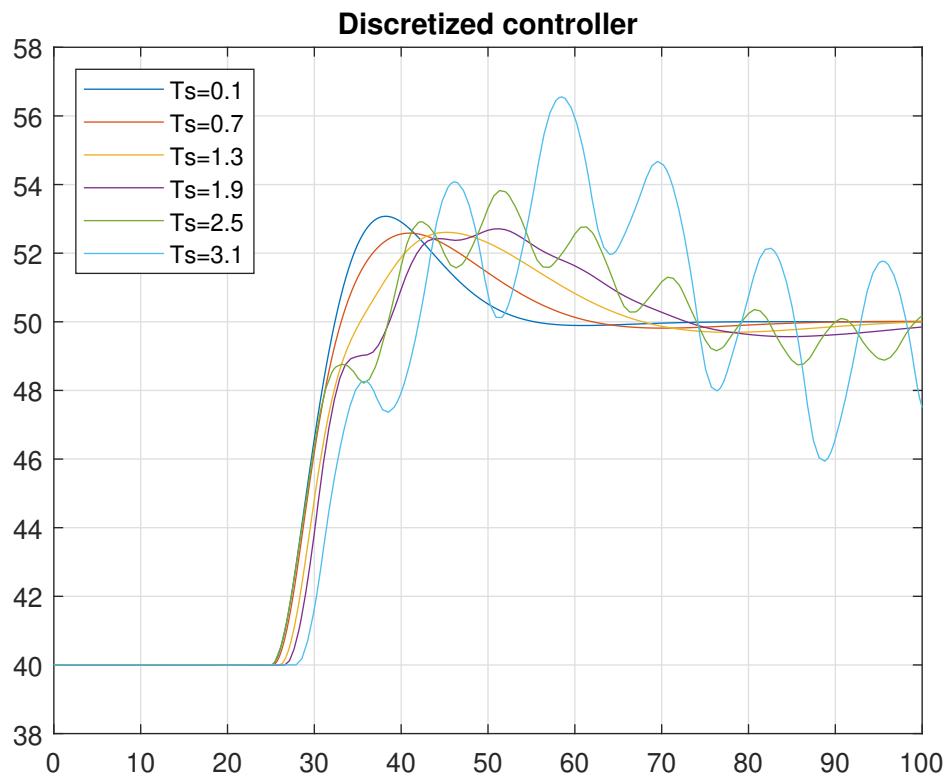


Figure 4: Performance of a discrete controller at different sampling periods

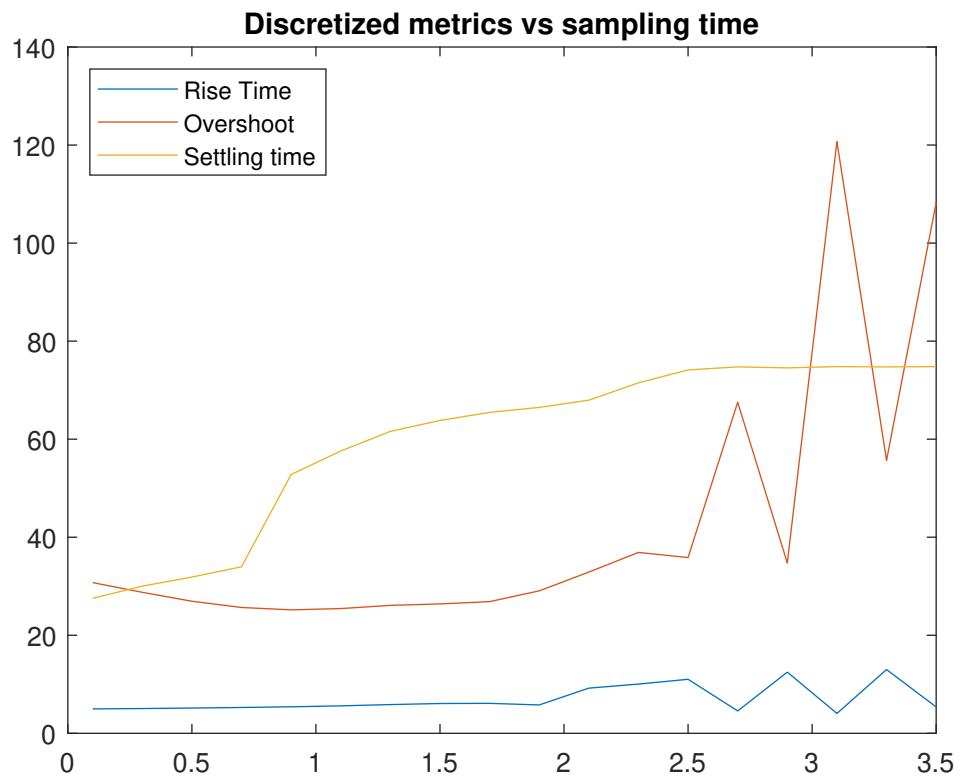


Figure 5: Different performance metrics at different sampling periods for the discretized controller

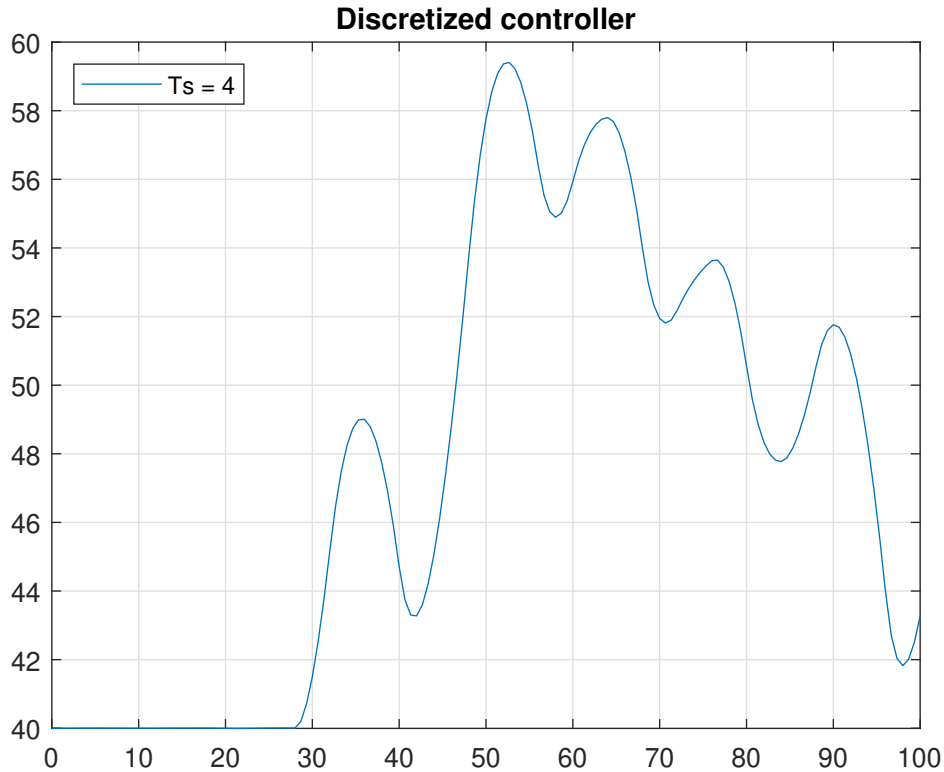


Figure 6: Performance of a discretized controller at sampling time 4 seconds.

Task 9

The sampling time should be chosen such that $\frac{0.05}{\omega_c} < h < \frac{0.14}{\omega_c}$. Focusing on the third scenario with the best controller, this gives the interval: $0.138 < h < 0.386$. A choice of $T_s = 0.25$ does seemingly look like a good performing choice. This can be confirmed in both the sampled control signal and discretized controller both work well at this sampling period.

Task 10

The case specific requirements in this scenario only allows us a sampling time of $T_s = 0.3$ seconds. The highest values in the suggested range still gives close to ok performance. As a rule of thumb, the choice of sampling time, as described in question 9, is a better choice as that provides a signal profile that is more similar to the continuous version. The general case might not allow the same amount of overshoot as is allowed in our case, for example.

Task 11

Running the simulation with a sampling time of $T_s = 4$ provides a bad controller. It is not stable at all. It provides a very oscillating behaviour in the measured output. The results can be seen in figure 6

Task 12

The two matrices can be calculated by $\Phi = e^{4A}$ and $\Gamma = \int_0^4 e^{sA} ds B$. For a piece-wise constant input the Γ value is given by $\frac{b}{a}(e^{4a} - 1)$. C stays constant in the discretized system and is the same as in the continuous case. In a multi state model, and the A matrix being invertible, the matrix Γ is equal to $A^{-1}(e^{4A} - I)B$.

The matrix A in system G is given as:

$$A = \begin{bmatrix} -\frac{1}{\tau} & 0 \\ \frac{1}{\tau} & -\frac{1}{\gamma\tau} \end{bmatrix} \quad B = \begin{bmatrix} \frac{k}{\tau} \\ 0 \end{bmatrix} \quad (1)$$

After discretization we have,

$$\Phi = \begin{bmatrix} 0.7249 & 0 \\ 0.2332 & 0.7249 \end{bmatrix} \quad \Gamma = \begin{bmatrix} 0.6017 \\ 0.0916 \end{bmatrix} \quad (2)$$

Task 13

The discretized model is fully controllable as the W_c matrix is invertible and of full rank. It is also fully observable as the matrix W_o is also both invertible and of full rank. The matrices are given as:

$$W_c = [\Gamma \quad \Phi\Gamma] \quad W_o = \begin{bmatrix} C \\ C\Phi \end{bmatrix} \quad (3)$$

Task 14

The choice of l_r behaves as a feed forward filter and makes sure that the reference signal has a steady state error of zero. Adding this filter detaches the design process of choosing a good performing controller from the reference following part.

Task 15

$$\hat{x}(k+1|k) = \Phi\hat{x}(k|k-1) + \Gamma u(k) + K[y(k) - C\hat{x}(k|k-1)] \quad (4)$$

$$u(k) = -L\hat{x}(k) + l_r r(k) \quad (5)$$

$$y(k) = Cx(k) \quad (6)$$

$$\hat{x}(k+1|k) = (\Phi - KC - \Gamma L)\hat{x}(k|k-1) + KCx(k) + \Gamma l_r r(k) \quad (7)$$

After some matrix algebra, the final augmented matrices are as follows:

$$A_a = \begin{bmatrix} \Phi & -\Gamma L \\ KC & \Phi - KC - \Gamma L \end{bmatrix} \quad B_a = \begin{bmatrix} \Gamma l_r \\ \Gamma l_r \end{bmatrix} \quad (8)$$

Task 16

$$\hat{x}(k|k-1) = x(k) - \tilde{x}(k|k-1) \quad (9)$$

We solve for the non-estimated system first,

$$x(k+1) = \Phi x(k) + \Gamma u(k) \quad (10)$$

$$x(k+1) = \Phi x(k) - \Gamma L \hat{x}(k|k-1) + \Gamma l_r r(k) \quad (11)$$

$$x(k+1) = \Phi x(k) - \Gamma L (x(k) - \tilde{x}(k|k-1)) + \Gamma l_r r(k) \quad (12)$$

$$x(k+1) = (\Phi - \Gamma L) x(k) + \Gamma L \tilde{x}(k|k-1) + \Gamma l_r r(k) \quad (13)$$

$$(14)$$

Next we solve the estimated system and substitute all \hat{x} with $x - \tilde{x}$.

$$\hat{x}(k+1|k) = (\Phi - KC - \Gamma L) \hat{x}(k|k-1) + KCx(k) + \Gamma l_r r(k) \quad (15)$$

$$\hat{x}(k+1|k) = (\Phi - KC - \Gamma L) (x(k) - \tilde{x}(k|k-1)) + KCx(k) + \Gamma l_r r(k) \quad (16)$$

$$\hat{x}(k+1|k) = (\Phi - \Gamma L) x(k) - (\Phi - KC - \Gamma L) \tilde{x}(k|k-1) + \Gamma l_r r(k) \quad (17)$$

$$(18)$$

Finally, we merge these two expressions

$$\tilde{x}(k+1|k) = x(k+1) - \hat{x}(k+1|k) \quad (19)$$

$$\tilde{x}(k+1|k) = (\Phi - \Gamma L) x(k) + \Gamma L \tilde{x}(k|k-1) + \Gamma l_r r(k) - \quad (20)$$

$$((\Phi - \Gamma L) x(k) - (\Phi - KC - \Gamma L) \tilde{x}(k|k-1) + \Gamma l_r r(k))$$

$$\tilde{x}(k+1|k) = (\Phi - KC) \tilde{x}(k|k-1) \quad (21)$$

$$A_z = \begin{bmatrix} \Phi - \Gamma L & \Gamma L \\ 0 & \Phi - KC \end{bmatrix} \quad B_z = \begin{bmatrix} \Gamma l_r \\ 0 \end{bmatrix} \quad (22)$$

The separation principle holds as the error reduction is handled separately from the system dynamics. It is possible to select K , which controls the change in error rate, independently from Φ , Γ , L , and l_r , which controls the system dynamics.

Task 17

The closed loop continuous system is given by $G_c = \frac{GF}{1 + GF}$. This system has poles at $s_{1,2} = -0.5$, and $s_{3,4} = -0.19 \pm j0.12i$. Mapping this to the z-plane gives poles at the locations $z_{1,2} = 0.1353$ and $z_{3,4} = 0.4677 \pm 0.2435i$. Next step is to decide which of these poles should be used to calculate the system controlling part and the error suppressing part. A fast system is desirable since that will give a rapid response to a change in reference, a quick error suppressing part is good, as that will make any errors from the observer disappear fast. Having the error go away faster is better as the observability error will otherwise propagate through the system and get reinforced. It is advisory to have a faster observer than controller. Thus the poles closer to zero are chosen to calculate K , while the two poles further away are used for L . The eigenvalues of A_a map exactly to the desired discrete poles, meaning that the poles are placed as intended.

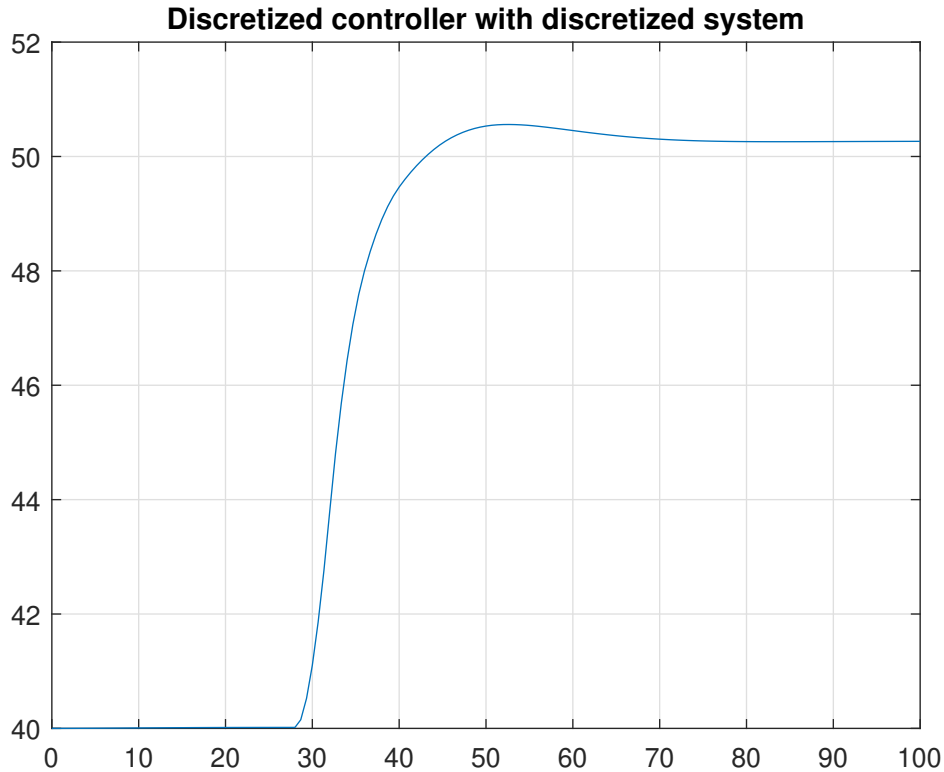


Figure 7: A fully discrete controller designed with pole placement.


Task 18

Simulating the fully discrete controller gives a response as can be seen in figure 7. The offset at the end is not perfect as the steady state error is not zero. The error could be because of the fact that we are getting outside of the range where the linearization assumption that we used in the beginning does not hold true any more. Since the underlying physical process is non-linear, the linearization only works in a proximity to the linearization point. Otherwise there will be errors present, that will grow larger the further from this point the evaluation goes.

Task 19

A 10 bit resolution gives $2^{10} = 1024$ different values. A range between 0 and 100 thus gets a resolution of $\frac{100}{1024} = 0.0977$.

Task 20

The saturation is needed to limit the signal to lay between physically realizable boundaries. If these boundaries did not exist  the controller could try to output values that would in worst case damage the machinery that it is connected to.

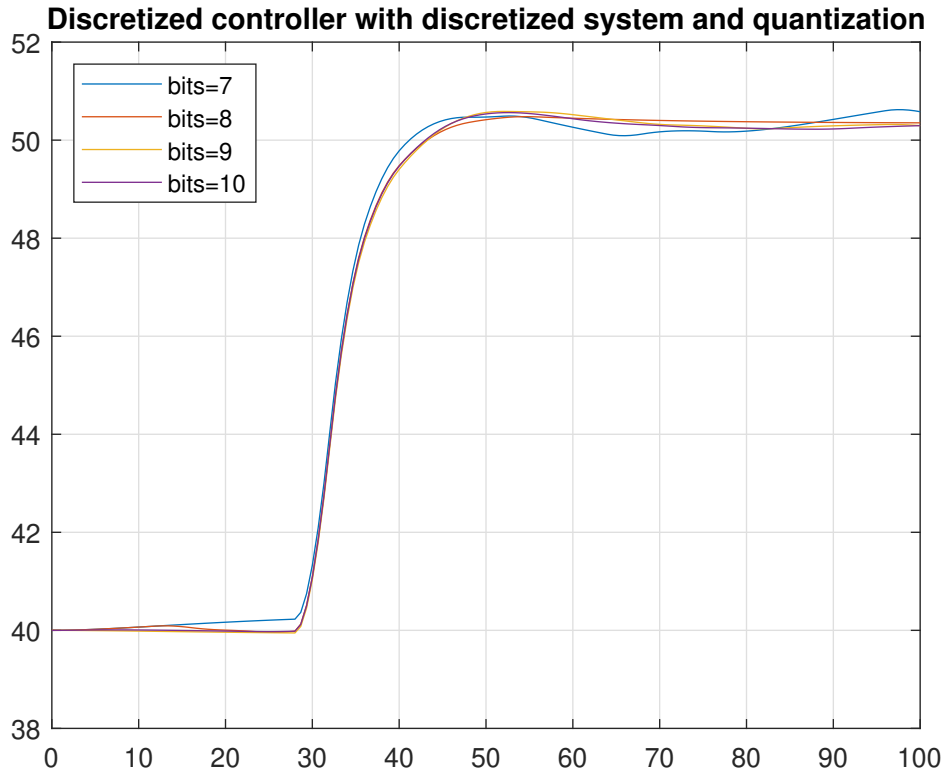


Figure 8: Controller performance at different quantization levels

Task 21

The controller fails at a bit resolution of 7 or lower, as can be seen in figure 8. This corresponds to failure at bit depth of 128 with a resolution of 0.78. This is a result in the magnitude that the controller is working in, thus it is not giving an acceptable performance. At this quantization it is impossible to do fine adjustments.

References

[WAA] Björn Wittenmark, Karl Johan Åström, and Karl-Erik Årzén. *Computer Control: An Overview*. IFAC Professional Brief 1 (2002): 2.