EE 406
Laboratory of Feedback Control Systems

Experiment 7
Position Control in Discrete Time

May 2022

1 Objectives

In this laboratory session, you will become familiar with the fundamentals of discrete time control system design. The challenge of this lab is to control the position of the IP02 linear motion servo plant with specified control performance by making use various discrete time controllers.

At the end of this lab, you are expected to

- comprehend the differences between various continuous to discrete transformation methods,
- analyze the effect of sampling period on the system response,
- to be able to design and implement a deadbeat controller.

2 Prerequisites

To successfully carry out this laboratory, the prerequisites are

- to be familiar with your IP02 main components (e.g., actuator, sensors), your power amplifier (e.g., UPM), and your data acquisition card (e.g., Q2 or Q8), as described in References [1], [2], [3], and [4],
- to have successfully completed the Experiment #2. In this experiment, we will make use of the controller that was previously designed during Experiment #2,
- to be familiar with the complete wiring of your IP02 servo plant, as per dictated in References [1],
- to be familiar with the fundamentals of discrete time controller design.

3 References

[1] Experiment #1: Control Hardware and Software Setup, Signal Interfaces
[2] IP02 User Manual.
[3] DAQ User Manual.
[4] Universal Power Module User Manual
[5] QuaRC Installation Guide.
[6] QuaRC User Manual (type doc quarc in Matlab to access).
[7] Experiment #2: Proportional Derivative Position Control
[8] G.F. Franklin and J. D. Powell, *Digital control of dynamic systems*. Reading, Mass.: Addison-Wesley Pub. Co., 1980.
[9] K. Ogata, *Discrete-time control systems*, *2nd edition*. Englewood Cliffs, N.J.: Prentice-Hall, 1987.

4 Experimental Setup

4.1 Main Components

This laboratory is composed of the following hardware and software components:

- **Power Module:** Quanser UPM 1503 or UPM 2405
- **Data Acquisition Board:** Quanser Q2 or Q8
- **Linear Motion Servo Plant:** Quanser IP02
- **Real Time Control Software:** The QuaRC-Simulink configuration, as detailed in the References [5].
- **A Computer to Run Matlab-Simulink and the QuaRC software**

The setup is the same as the one in Experiment #2.

4.2 Wirings

To wire up the system, please follow the default wiring procedure for your IP02 as fully described in References [1]. When you are confident with your connections, you can power up the UPM. (If you are uncertain of any parts of the hardware, seek help from your lab assistant.)

5 Controller Design Specifications

We have already designed and implemented a continuous time controller based on Proportional-Velocity (PV) controller scheme which is a modified implementation of Proportional-Derivative (PD) control approach. The following requirements of IP02 closed-loop system were satisfied under the control of this controller.

i) The percent overshoot was less than 10%,

$$PO \leq 10\%.$$

ii) The time to first peak was 150 ms,

$$t_p = 0.15 \text{ sec}$$

ii) The steady state error to unit step was zero,

$$e_{ss} = 0.$$

In the present laboratory, you will firstly build discrete time controllers with various sampling frequencies considering the PD controller whose parameters were previously acquired.

Subsequently, you will design and implement a deadbeat controller. You should revise your EE402 notes to remember deadbeat design procedure.

Student Name: Özgür Gülsuna
Student ID: 2307668

6 Preliminary Work

6.1 Transformations from s plane to z plane

Transfer functions in s domain can be transformed into z domain using different techniques. Here, we will be dealing with backward rectangular rule, forward rectangular rule, Tustin's method (bilinear transformation) and zero-pole matching.

The first three transformation rules are derived from the numerical integration methods. A transfer function in s domain describes a differential equation in continuous time and this differential equation can be solved by numerical integration methods. For example, suppose that we have the following transfer function

$$\frac{C(s)}{R(s)} = \frac{b}{s+a}, \tag{1}$$

which defines the differential equation in (2).

$$\dot{c} + ac = br \tag{2}$$

This equation can be solved for *c* by the following integration.

$$c(t) = \int_0^t [br(\tau) - ac(\tau)]d\tau \tag{3}$$

Assuming fixed-step size of *T*, the integration can be approximated using various numerical approaches.

$$c(kT) = \int_0^{k(T-1)} [br(\tau) - ac(\tau)]d\tau + \int_{k(T-1)}^{kT} [br(\tau) - ac(\tau)]d\tau \tag{4}$$

$$= c((k-1)T) + \int_{k(T-1)}^{kT} [br(\tau) - ac(\tau)]d\tau \tag{5}$$

The second term on the right-hand side of (5) can be computed using one of the following methods

- Backward rectangular rule,
- Forward rectangular rule,
- Trapezoidal rule.

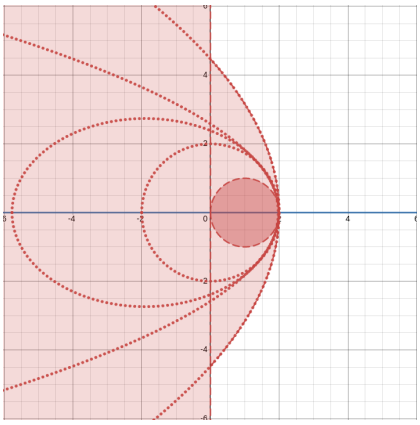
Table 1: Transformation methods

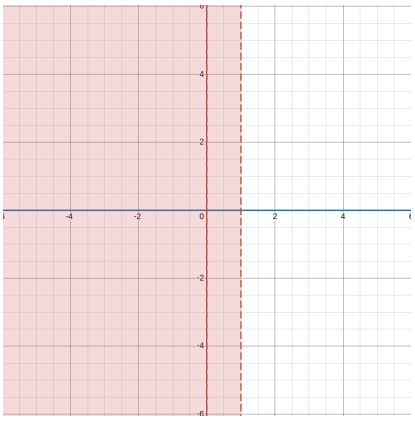
Rule	Approximation
Backward rectangular rule	$\frac{z-1}{Tz}$
Forward rectangular rule	$\frac{z-1}{T}$
Trapezoidal rule	$\frac{z-1}{Tz}$

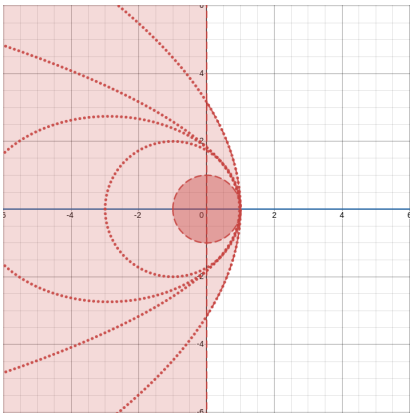
Using these methods, you may solve the numerical integration problem yourself, and then, you should be able to observe that these rules result in different transformations from s plane to z plane which are summarized in Table 1 as substitutions of approximate expressions for the variable *s*. Note that the trapezoidal rule is also known as Tustin's method or bilinear transformation. For detailed information, you can read Chapter 3, pages 53-66 of Franklin's book which is reference [8].

Using three methods explained above, transform the left-half of s plane to z plane. Plot your results indicating especially the mapping for $s = jw$ axis.

Backward Rectangular Rule | **Forward Rectangular Rule** | **Trapezoidal Rule**







Compare your results commenting on any possible advantages or disadvantages of these approaches. Which one would you prefer? Why?

- Backward rectangular mapping does not cover the stable region of the Z plane. Also stable poles of the S plane turns into Astable poles.
- Forward rectangular mapping have disadvantage of again having possibility of changing the stability of the poles.
- Trapezoidal mapping have possibility that can preserve the stability of the poles.

I would prefer the trapezoidal rule but it is little bit more calculation heavy.

Another method that we will cover in this experiment is pole-zero matching. In this approach, discrete time equivalent of a continuous time system is achieved by mapping all of the poles and zeros of the system using the following equality:

$$z = e^{Ts}, \tag{6}$$

where T is the sampling period.

Additionally, one may transform the system using hold equivalents. If a zero-order hold device is employed, then the equivalent discrete time system is computed using (7).

$$H(z) = (1 - z^{-1})\mathrm{Z}\left\{\frac{H(s)}{s}\right\}$$

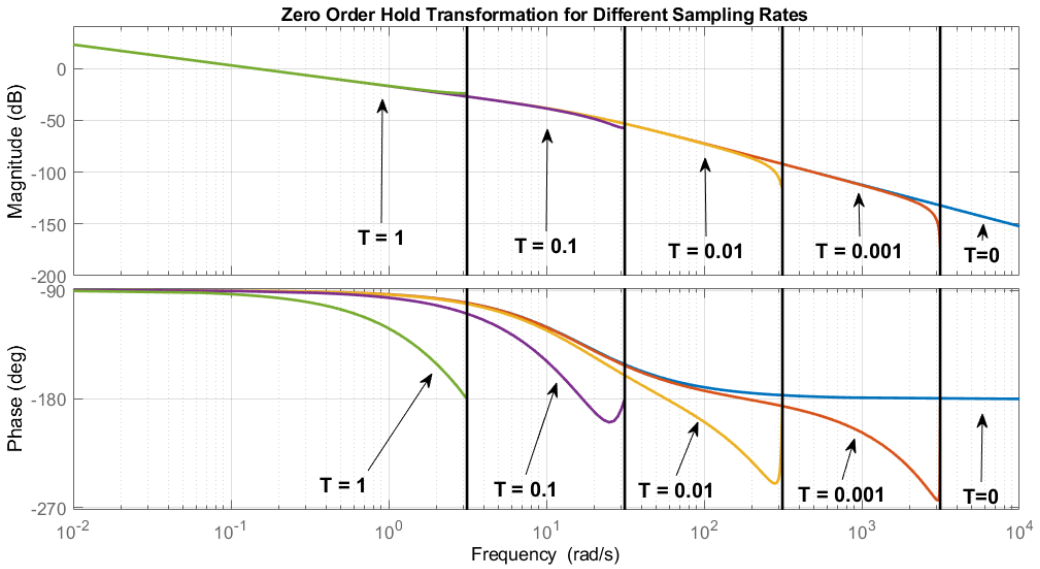
In previous experiments, we derived the transfer function of the plant as in (8).

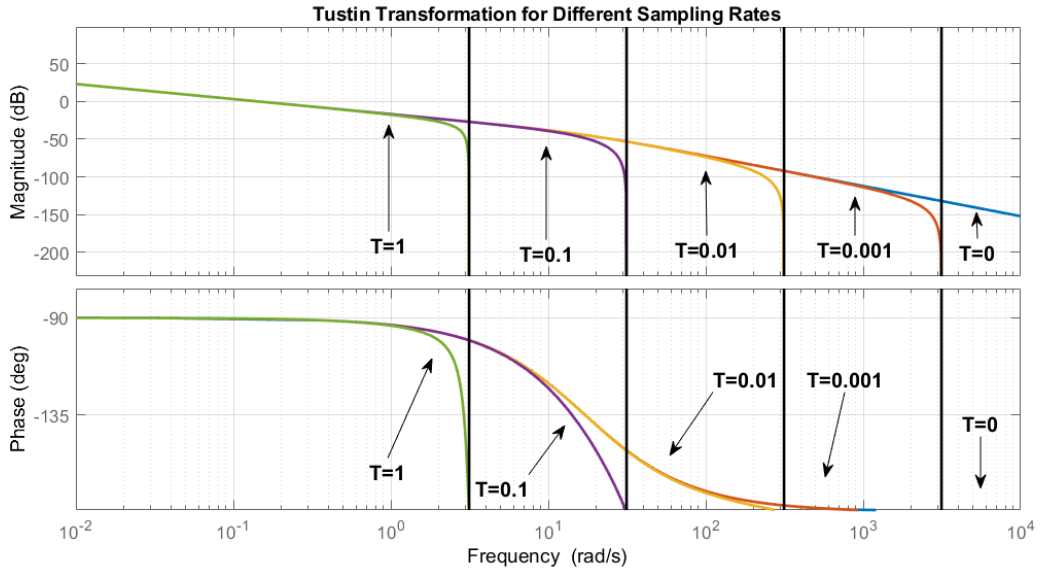
$$H(s) = \frac{2.45}{s(s + 17.13)} \tag{8}$$

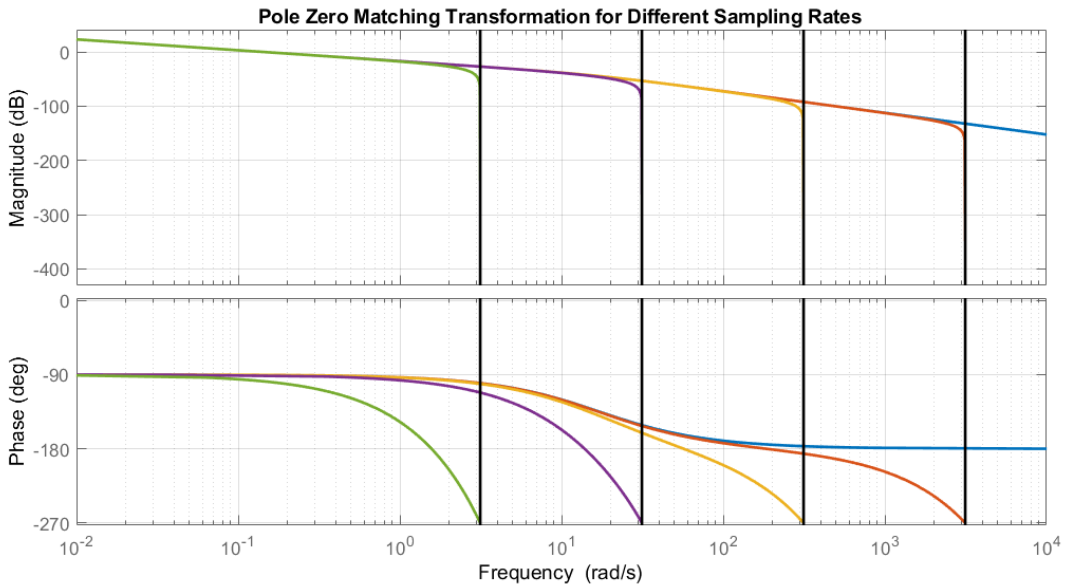
Obtain the Bode plot of this transfer function by writing a MATLAB script. Then on the same figure, plot Bode plots of discrete equivalents of the system when transformation is achieved by zero order hold, Tustin's method, and pole-zero matching for sampling period of $T = 0.001$ sec.

Repeat the above step for T = 0.01 sec, T = 0.1 sec and T = 1 sec. Take the printouts of your results and attach them to end of your preliminary work.

Comment on your results.







We observe that for different methods, lower the sampling period better the discrete representation which means it is more similar to the continious time response.

In terms of differences between the transformation methods, Zero Order Hold and Pole Zero Matching methods introduce unwanted phase shift which is not visible in the continious time couterpart.

I would selec the Tustin Method because it is the one that has the best frequency response and have no unwanted extra phase shift.

6.2 Transforming the Continuous Time Controller using Different Sampling Periods

In Experiment #2, a PV controller that satisfies the design requirements was achieved, and the corresponding gains were attained as

$$K_p = 275.6,\ Kv = 5.55. \tag{9}$$

In that experiment, a PV controller was preferred instead of a PD controller for computational purposes. Since they yield to the same characteristic equation for the closed loop transfer function, we will now implement a PD controller using the same gain values.

$$G_c(s) = K_p + K_d s, \tag{10}$$

where $K_p = 275.6$, $K_d = 5.55$.

Transform this transfer function in (10) into z plane analytically using zero-order hold method for sampling frequencies $f = 500$Hz, $f = 100$Hz, $f = 50$Hz and $f = 10$Hz.

$$G_c(s) = K_p + K_d s$$

$$G_c(z) = \left(1 - \frac{1}{z}\right) Z\left\{\frac{K_p}{s} + K_d\right\} = \left(1 - \frac{1}{z}\right)\left[\frac{K_p}{1 - \frac{1}{z}} + K_d\right]$$

$$G_c(z) = K_p + \left(1 - \frac{1}{z}\right) K_d$$

independent of sampling frequency !

6.2.1 Building Hybrid Structured Simulink Models

The Simulink model that is created to simulate the behavior of the discrete controllers would have a hybrid structure; since the plant would be represented by a *continuous transfer function*, while the controller should be realized by a *discrete transfer function*. Therefore, we need to make use of particular Simulink blocks such as *Zero Order Hold (ZOH)* and *Rate Transition* to enable reliable data transfer between continuous time and discrete time systems.

1. Examine closely the documentation for *ZOH* and *Rate Transition* blocks in Simulink.
2. Discuss the functionality of each block and indicate the discrepancies between formal purpose of usage which is stated in the documentation and your expectations from your theoretical background.

ZOH Block takes in sampling frequency and converts the continuous signal into discrete output signal.

Rate Transition Block on the other hand changes the rate of different blocks to equate them.

ZOH does not change rate between two discrete rated systems, it might fail since the input is not continuous time signal. But using Rate Transition Block to transform continuous time to discrete would work since continuous time signal have enough information to be discritisized.

6.2.2 Simulating Discrete Time PD Controllers

At this stage, we will implement a Simulink model to simulate the control loop utilizing discrete time PD controller scheme and investigate corresponding performances of the controllers with different sampling frequencies. Your model should look like the one depicted in Figure 1. Throughout the implementation of the model, you should be careful about the following points.

- The solver of the model is to operate at 1 kHz. Therefore, the type of the solver is to be selected as *fixed-step-size*. Please check *Model Configuration Parameters* pane.
- The sample time for the discrete time blocks such as ZOH should be adjusted in accordance with the sampling frequency of the discrete time controller you investigate.
- The sample time of *Rate Transition* blocks is to be same with the fundamental sample time of the model.

- You can use a *Saturation* block to limit the output of the controller considering the limits of the UPM which are ±13V.
- You are free to use any suitable discrete time block to represent the discrete time controller, but usage of *Discrete Transfer Function* is suggested as illustrated in Figure 1.
- You are encouraged to use *Goto* and *From* blocks to render your model more coherent and organized.
- The position command should be a square wave with 100 mm peak-to-peak amplitude and a period of 4 seconds.
- The position tracking performances of the systems obtained by all controllers, including the continuous one, are to be fed to the same scope.

Note: You are expected to bring your models to the lab with a flash memory. Yourmodels should have the extension '.mdl' instead of '.slx'.

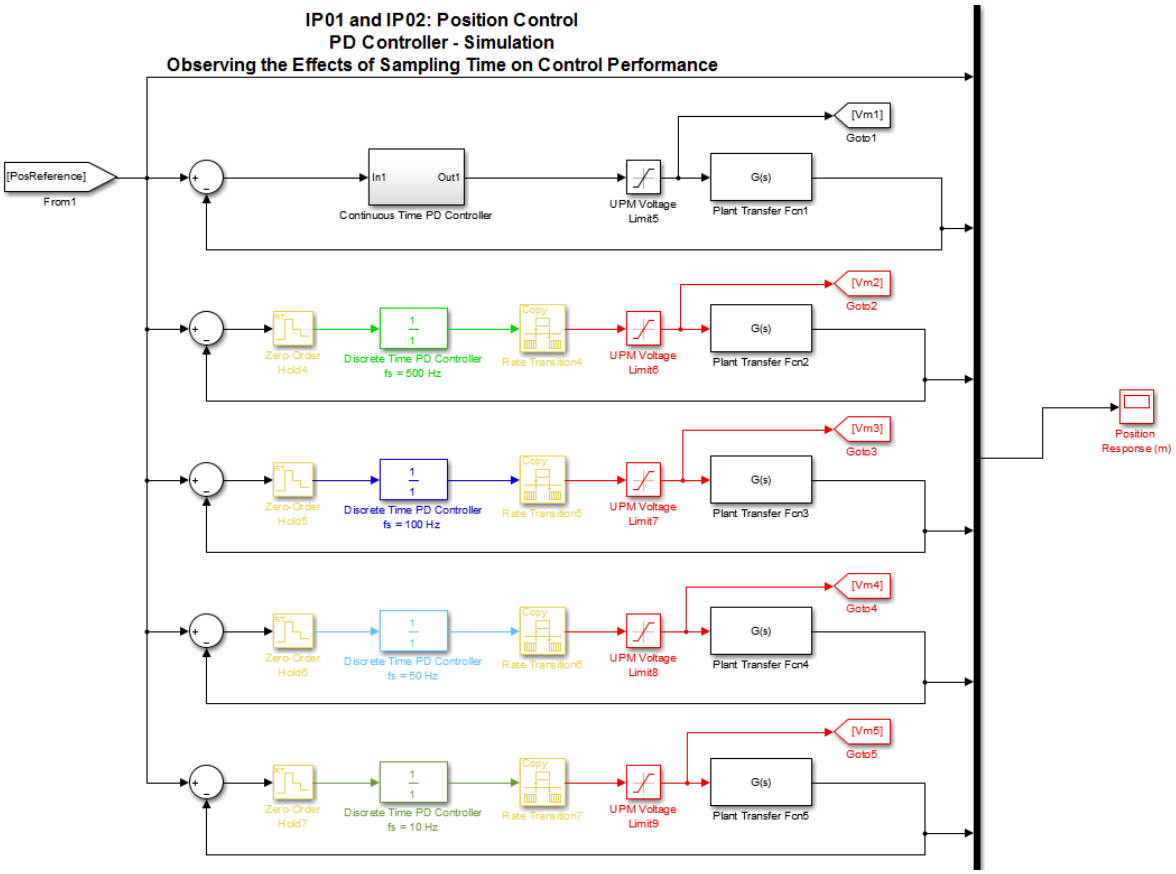


Figure 1: Simulink model to observe the effect of sampling time for discrete time control

Provide your simulation results by attaching them to your preliminary work. Comment on the simulation results comparing the effect of sampling frequency.

6.3 Deadbeat Controller Design

Deadbeat controller is an optimal discrete time controller that drives the output of the system to the desired constant value with zero steady state error at minimum number of steps. For an N^{th} order system, it takes N number of steps at most to settle. You are expected to go over *Section 4.7 - Analytical Design Method* in Discrete-Time Control Systems, Ogata. You will benefit a lot from the examples given in this section considering the rest of the Preliminary Work.

Note: In this part, represent your final results in the following form.

$$D(z) = K\frac{1 - z_0 z^{-1}}{1 - p_0 z^{-1}}$$

Design a deadbeat controller for a unit step input with 10 milliseconds of sampling time in order to control the position of the cart, i.e., $T_{controller}$ = 10 *msec*. Use ZOH equivalent of the transfer function of the plant.

Design another deadbeat controller for a unit step input with 250 milliseconds of sampling time, i.e., Tcontroller = 250 msec. Use ZOH equivalent of the transfer function of the plant.

```
s = tf('s');
z = tf('z');
H = 2.45/(s*(s+17.13));

for T = [1/100 1/25]
    G_z = c2d(H,T,'zoh');
    [num, den]=tfdata(G_z);
    den=cell2mat(den);
    num=cell2mat(num);
    A= den;
    B= num;
    Gc_z1=tf(A,B,T);
    Gc_z2=tf(1,[1 -1],T);
    Gc_z=Gc_z1*Gc_z2
end
```

```
Gc_z =

          z^2 - 1.843 z + 0.8426
  ----------------------------------------
  0.0001158 z^2 - 6.425e-06 z - 0.0001094

Sample time: 0.01 seconds
Discrete-time transfer function.


Gc_z =

         z^2 - 1.504 z + 0.504
  -----------------------------------
  0.00158 z^2 - 0.0003216 z - 0.001258

Sample time: 0.04 seconds
Discrete-time transfer function.
```