## Digital Controller Design by Emulation for a Rotor-Pendulum Model

- **Preparation**: See "What should I do in Week 10?" on SUNLearn.
- **Instructions**: You have to do this practical by yourself – no group work. Write your own practical report and hand it in on SUNLearn by Friday at 17:00. See SUNLearn for instructions.
- **Help**: Study the Matlab/Simulink documentation by clicking on the *Help* menu if you do not know how to use a certain function/block.

## Digitale Beheerder-Ontwerp met Emulasie vir 'n Rotor-Pendulum-Model

- **Voorbereiding**: Sien "What should I do in Week 10?" op SUNLearn.
- **Instruksies**: Jy moet hierdie prakties self doen – geen groepwerk. Skryf jou eie praktiese verslag en handig dit in op SUNLearn teen Vrydag 17:00. Sien SUNLearn vir instruksies.
- **Hulp**: Lees die Matlab/Simulink-dokumentasie deur na die *Help*-kieslys te gaan indien jy nie weet hoe om 'n sekere funksie/blok te gebruik nie.

# Background / *Agtergrond*

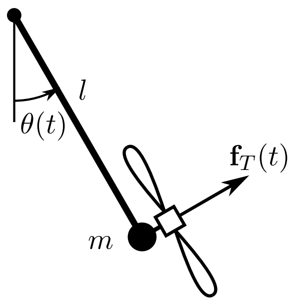
We look at two approaches for digital controller design: *emulation* and *direct digital design*. In today's practical, we apply the first approach – digital controller design by emulation – to a realistic rotor-pendulum simulation. Emulation consists of two steps: (a) designing an analogue controller to control the analogue plant, and (b) discretising the controller.

Ons kyk na twee benaderings vir digitale beheerderontwerp: *emulasie* en *direkte digitale ontwerp*. In vandag se prakties pas ons die eerste benadering – digitale beheerderontwerp met emulasie – toe op 'n realistiese rotor-pendulum simulasie. Emulasie bestaan uit twee stappe: (a) ontwerp van 'n analoog beheerder om die analoog aanleg te beheer, en (b) diskretisering van die beheerder.

# Assignment / *Voorskrif*

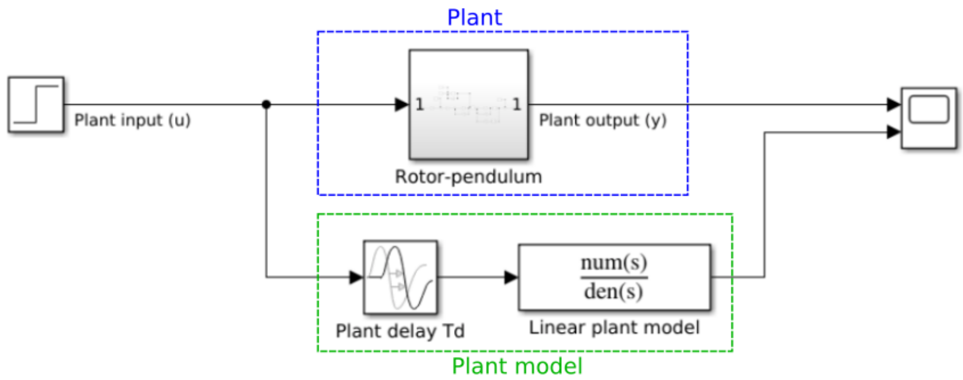
The rotor-pendulum tutor consists of a hanging pendulum with an electrically actuated rotor (propeller) at its tip. The voltage across the electrical motor that drives the rotor is the input (units: V), and the pendulum angle $\theta(t)$– measured with a potentiometer – is the output (units: rad). A diagram of the setup with pendulum length $l$, mass $m$ and rotor force $\mathbf{f}_T(t)$ is shown below.

Die rotor-pendulum-tutor bestaan uit 'n hangende pendulum met 'n elektries-gedrewe rotor (propeller) aan sy punt. Die spanning oor die elektriese motor wat die rotor aandryf is die intree (eenheid: V), en die pendulumhoek $\theta(t)$ – gemeet met 'n potentiometer – is die uittree (eenheid: rad). 'n Diagram van die opstelling met pendulum-lengte $l$, massa $m$ en rotorkrag $\mathbf{f}_T(t)$ word hieronder getoon.



The "actual" plant for today's practical is implemented in Simulink as a block as shown below (in blue).

Die "werklike" aanleg vir vandag se prakties is geïmplementeer as 'n Simulink blok soos getoon hieronder (in blou).




The plant input is the input voltage $u(t) = v_i(t)$ and the measured output is the pendulum angle $y(t) = \theta(t)$. The plant contains measurement noise as well as slight random disturbances (modelling aerodynamic disturbances).

The plant model we are going to use (see Question 1) is also shown (in green). This model requires the following Matlab variables to be defined: `Td` (the time delay $T_d$), `wn` (the natural frequency $\omega_n$), `zeta` (the damping $\zeta$), and `C` (the scaling constant $C$) – see Question 1 for their meanings.

To set up this Simulink model, download the Simulink file `Prac5_setup.slx` as well as the Matlab data file `Prac5_parameters.mat` and store them in the same folder[1]. In the Matlab file explorer, double-click on `Prac5_parameters.mat` to load the parameters, open `Prac5_setup.slx`, create the variables `Td`, `wn`, `zeta` and `C`[2], and run the simulation.

Die aanlegintree is die intreespanning $u(t) = v_i(t)$ en die gemete uittree is die pendulumhoek $y(t) = \theta(t)$. Die aanleg bevat meetruis asook klein lukrake steurseine (wat aerodinamiese steurings voorstel).

Die aanlegmodel wat ons gaan gebruik (sien Vraag 1) word ook gewys (in groen). Hierdie model vereis dat die volgende Matlab-veranderlikes gedefinieer word: `Td` (die tydvertraging $T_d$), `wn` (die natuurlike frekwensie $\omega_n$), `zeta` (die demping $\zeta$), en `C` (die skaleringskonstante $C$) – sien Vraag 1 vir hul betekenis.

Om hierdie Simulink-model op te stel, laai die Simulink-lêer `Prac5_setup.slx` asook die Matlab-datalêer `Prac5_parameters.mat` af en stoor dit in dieselfde "folder"[3]. In die Matlab "file explorer", dubbelkliek op `Prac5_parameters.mat` om die parameters te laai, maak `Prac5_setup.slx` oop, skep die veranderlikes `Td`, `wn`, `zeta` en `C`[4], en hardloop die simulasie.

[1] If you use Matlab/Simulink online, upload the files to one of your Matlab online folders.
[2] I suggest you use an m-file to create the variables.
[3] Indien jy Matlab/Simulink online gebruik, laai die lêers op na een van jou Matlab online "folders".
[4] Ek stel voor jy gebruik 'n m-lêer om die veranderlikes to skep.

# 1 System Identification / *Stelselidentifikasie*

## Problem statement: / *Probleemstelling:*

Set up a plant model of the rotor-pendulum.

Stel 'n aanlegmodel op van die rotor-pendulum.

## Solution development: / *Ontwikkeling van oplossing:*

The plant dynamics can be described as a lightly-damped second-order system (due to the pendulum) with a time delay (due to the response time of the rotor), with the transfer function given by

Die aanleg-dinamika kan beskryf word as 'n liggedempte tweede-orde stelsel (a.g.v. die pendulum) met 'n tydvertraging (a.g.v. die reaksietyd van die rotor), met die oordragsfunksie gegee deur

$$G(s) = \frac{Y(s)}{U(s)} = \frac{Ce^{-T_d s}}{s^2 + 2\zeta\omega_n s + \omega_n^2}, \tag{1}$$

where the damping is given by $\zeta$, natural frequency by $\omega_n$, constant scaling by $C$, and time delay by $T_d$. Typical parameter values are $T_d$ between 0.1 and 0.2 s, $\zeta$ between 0.01 and 0.1, $\omega_n$ between 4 and 5 rad/s, and $C$ about 1.

Apply a step input with an amplitude of 10 V to the input of both the actual plant and plant model, and adjust the parameters of the plant model until the responses agree fairly well (it will not be possible to match the responses exactly due to the non-idealities in the actual plant).

waar die demping gegee word deur $\zeta$, die natuurlike frekwensie deur $\omega_n$, die konstante skalering deur $C$, en die tydvertraging deur $T_d$. Tipiese parameter-waardes is $T_d$ tussen 0.1 en 0.2 s, $\zeta$ tussen 0.01 en 0.1, $\omega_n$ tussen 4 en 5 rad/s, en $C$ ongeveer 1.

Wend 'n trapintree met 'n amplitude van 10 V aan op die intree van beide die werklike aanleg en die aanlegmodel, en verstel die parameters van die aanlegmodel totdat die gedrag redelik goed ooreenstem (dit sal nie moontlik wees om hulle presies te laat ooreenstem nie a.g.v. die nie-idealiteite in die werklike aanleg).

## Experiments and results: / *Eksperimente en resultate:*

Compare the step response of the plant model with that of the actual plant.

Vergelyk die trapweergawe van die aanlegmodel met die van die werklike aanleg.

## Conclusions: / *Gevolgtrekkings:*

How accurate is the plant model? In which aspects do the response of the actual plant differ from that of the model?

Hoe akkuraat is die aanlegmodel? In watter aspekte verskil die gedrag van die werklike aanleg van die van die model?

## 2 Design of analogue controller / *Ontwerp van analoog beheerder*

### Problem statement: / *Probleemstelling:*

Design an analogue controller to control the analogue plant.

Ontwerp 'n analoog beheerder om die analoog aanleg te beheer.

### Solution development: / *Ontwikkeling van oplossing:*

Use the identified plant model and $s$-plane root locus design to obtain a controller $D(s)$ (by hand) that will satisfy the following closed-loop specifications:

- Optimally damped (overshoot of 4.3%) with a 2% settling time of 2 s
- Zero steady state tracking error for constant reference angles

Gebruik die geïdentifiseerde aanleg-model en $s$-vlak-wortellokus-ontwerp om 'n beheerder $D(s)$ te vind (met die hand) wat aan die volgende geslotelus-spesifikasies voldoen:

- Optimaal gedemp (oorskiet van 4.3%) met 'n 2%-wegsterftyd van 2 s
- Nul-gestadigde-toestand volgfout vir 'n konstante verwysings-hoek

Model the total time delay of the plant and the ZOH, $T_D = T_d + \frac{T_s}{2}$ [5], using the Padé approximation

Modelleer die totale tydvertraging van die aanleg en ZOH, $T_D = T_d + \frac{T_s}{2}$ [6], d.m.v. die Padé-benadering

$$G_{\text{delay}}(s) = e^{-T_D s} \approx \frac{\frac{1}{T_D}}{s + \frac{1}{T_D}}, \tag{2}$$

where the sampling period is given by $T_s = 0.15$ s. The plant model used for the controller design[7] is therefore given by

waar die monsterperiode gegee word deur $T_s = 0.15$ s. Die aanlegmodel wat gebruik word vir die beheerder-ontwerp[8] word dus gegee deur

$$G'(s) = \left( \frac{\frac{1}{T_D}}{s + \frac{1}{T_D}} \right) \left( \frac{C}{s^2 + 2\zeta\omega_n s + \omega_n^2} \right). \tag{3}$$

Use the following structure for the controller

Gebruik die volgende struktuur vir die beheerder

$$D_a(s) = K_D \frac{s^2 + 2\zeta\omega_n s + \omega_n^2}{s(s+a)}, \tag{4}$$

such that the controller zeros cancel the lightly damped plant poles, and the controller pole at the origin ensures a zero steady state tracking for a constant reference.

sodat die beheerder-zeros die liggedempte aanleg-pole uitkanselleer, en die beheerder-pool by die oorsprong 'n nul-gestadigde-toestand-volgfout vir 'n konstante verwysing verseker.

[5] In the lectures, we only modelled the effect of the time delay due to the ZOH; however, in this case the plant also contains a time delay, $T_d$, and we therefore model the *total* time delay, $T_D$, using the Padé approximation.

[6] In die lesings het ons slegs die effek van die vertraging a.g.v. die ZOH gemodelleer. In hierdie geval het die aanleg egter ook 'n vertraging, $T_d$, en ons modelleer dus die *totale* vertraging, $T_D$, d.m.v. die Padé-benadering.

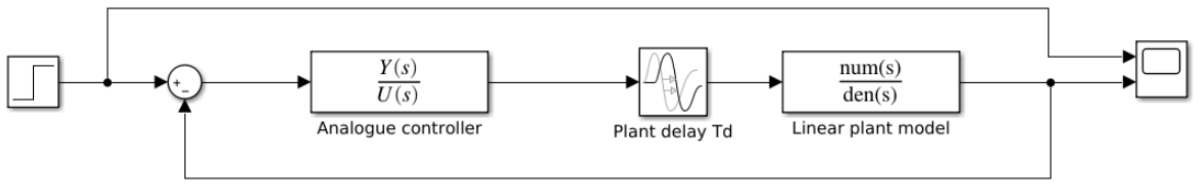
[7] Remember that we have simplified the plant and ZOH model so that we can use it for controller *design*. When *evaluating* the controller in simulation, we should use the model without simplifications, i.e. Equation 1.

[8] Onthou dat ons die aanleg- en ZOH-modelle vereenvoudig het sodat ons dit kan gebruik vir beheerder-*ontwerp*. As ons egter die beheerder wil *evalueer* in simulasie, moet ons die model sonder vereenvoudigings gebruik, d.w.s. Vergelyking 1.

## Experiments and results: / *Eksperimente en resultate:*

Implement the analogue controller $D_a(s)$ to control the rotor-pendulum *model* in Simulink, as shown in the diagram below. Apply a step signal with amplitude 0.5 rad (about 30°) to the reference input. Measure the overshoot and 2% settling time.

Implementeer die analoog beheerder $D_a(s)$ om die rotor-pendulum-*model* in Simulink te beheer soos getoon in die diagram hieronder. Wend 'n trapsein met amplitude 0.5 rad (omtrent 30°) aan op die verwysingsintree. Meet die oorskiet en 2%-wegsterftyd.




## Conclusions: / *Gevolgtrekkings:*

Does the analogue closed-loop system satisfy the specifications? Can you explain the deviations from the expected behaviour?

Voldoen die analoog geslotelus-stelsel aan die spesifikasies? Kan jy die afwykings vanaf die verwagte gedrag verduidelik?

# 3 Discretising the controller / *Diskretisering van die beheerder*

## Problem statement: / *Probleemstelling:*

Discretise the controller using the bilinear transform.

Diskretiseer die beheerder met die bilineêre transform

## Solution development: / *Ontwikkeling van oplossing:*

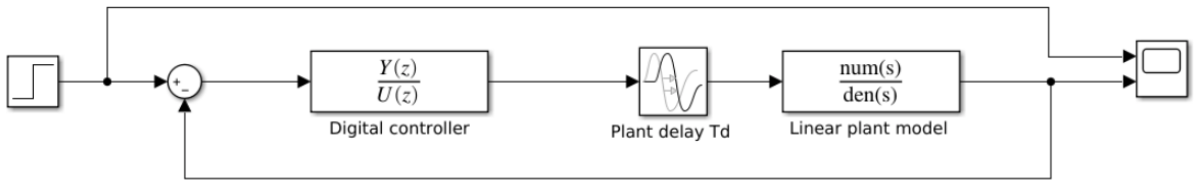
Discretise the controller $D(s)$ using the bilinear transform (Tustin's method) to obtain the discrete controller $D(z)$. Use a sampling period of $T_s = 0.15$ s. Use the Matlab commands `tf` and `c2d` (use the argument `'tustin'` with the latter) for the discretisation[9].

Diskretiseer die beheerder $D(s)$ d.m.v. die bilineêre transform (Tustin se metode) om die diskrete beheerder $D(z)$ te verkry. Gebruik 'n monsterperiode van $T_s = 0.15$ s. Gebruik die Matlab-funksies `tf` en `c2d` (gebruik die argument `'tustin'` vir lg.) vir die diskretisering[10].

## Experiments and results: / *Eksperimente en resultate:*

Implement the *discrete* controller $D(z)$ to control the rotor-pendulum *model* in Simulink, as shown in the diagram below. Use the *Discrete Transfer Fcn* block (in the *Discrete* folder) to implement the discrete controller[11] (remember to set the *Sample time* field to the correct value). Apply a step signal with amplitude 0.5 rad to the reference input. Compare the step response to that of the system with analogue controller.

Implementeer die *diskrete* beheerder $D(z)$ om die rotor-pendulum-*model* in Simulink te beheer, soos getoon in die diagram hieronder. Gebruik die *Discrete Transfer Fcn*-block (in die *Discrete*-onderafdeling) om die diskrete beheerder te implementeer[12] (onthou om die *Sample time*-veld na die regte waarde te stel). Wend 'n trap met amplitude 0.5 rad aan op die verwysingsintree. Vergelyk die trapweergawe met dié van die stelsel met analoog beheerder.




## Conclusions: / *Gevolgtrekkings:*

Does the closed-loop system with the digital controller $D(z)$ satisfy the specifications? Explain any differences in behaviour of the closed-loop systems with analogue and digital controllers. (Hint: Would the discretised controller's zeros still cancel the plant poles?)

Voldoen die geslotelus-stelsel met die digital beheerder $D(z)$ aan die spesifikasies? Verduidelik enige verskille in die gedrag van geslotelus-stelsels met die analoog en digitale beheerders. (Wenk: Sal die gediskretiseerde beheerder se zeros steeds die aanleg se pole kanselleer?)

[9] Make sure you can do the discretisation by hand as well (after the practical).

[10] Maak seker dat jy die diskretisering met die hand ook kan doen (na die prakties).

[11] If you have created the discrete transfer function in Matlab as e.g. `Dz`, you can extract the numerator and denominator coefficients by using the command `[num,den] = tfdata(Dz,'v')`. You can then put the variables `num` and `den` in the respective *Numerator* and *Denominator* fields of the *Discrete Transfer Fcn* block.

[12] As jy die diskrete oordragsfunksie in Matlab geskep het as bv. `Dz`, kan jy die noemer- en deler-koëffisiënte onttrek deur die bevel `[num,den] = tfdata(Dz,'v')` te gebruik. Jy kan dan die veranderlikes `num` en `den` in die onderskeie *Numerator*- en *Denominator*-velde van die *Discrete Transfer Fcn*-blok plaas.

# 4 Applying the discretised controller to the actual plant / *Aanwending van die gediskretiseerde beheerder op die werklike aanleg*

## Problem statement: / *Probleemstelling:*

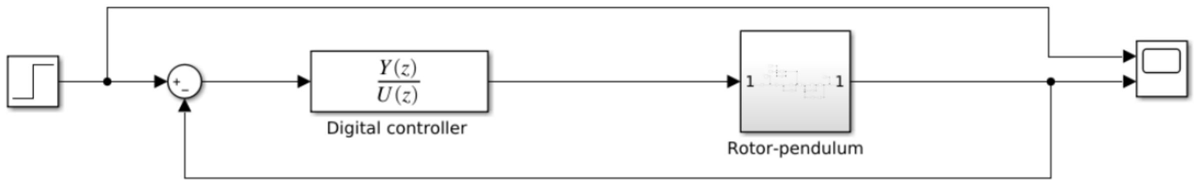
Apply discretised controller to the actual plant and analyse its behaviour.

Wend die gediskretiseerde beheerder aan op die werklike aanleg en analiseer die gedrag.

## Experiments and results: / *Eksperimente en resultate:*

Implement the *discrete* controller $D(z)$ to control the *actual* rotor-pendulum, as shown in the diagram below. Compare the step response to that of the control system with the simulated plant model (Question 3).

Implementeer die *diskrete* beheerder $D(z)$ om die *werklike* rotor-pendulum te beheer, soos in die onderste diagram getoon. Vergelyk die trapweergawe met dié van die beheerstelsel met die gesimuleerde aanlegmodel (Vraag 3).




## Conclusions: / *Gevolgtrekkings:*

Does the closed-loop system with the actual rotor-pendulum satisfy the specifications? Explain any difference in behaviour of the closed-loop systems with the simulated and actual plants.

Voldoen die geslotelus-stelsel met die werklike rotor-pendulum aan die spesifikasies? Verduidelik enige verskille in die gedrag van geslotelus-stelsels met die gesimuleerde en werklike aanlegte.