

## SV Control of a Realistic Second-order Plant

## TV Beheer van 'n Realistiese Tweede-orde Aanleg

- |  |  |
|--|--|
| <ul style="list-style-type: none"> <li>• <b>Preparation:</b> See “What should I do in Week 6?” on SUNLearn.</li> <li>• <b>Instructions:</b> This practical has to be done individually – no group work. Write your own practical report and hand it in on SUNLearn by Friday at 17:00. See SUNLearn for instructions.</li> <li>• <b>Note:</b> Study the Matlab/Simulink documentation by clicking on the <i>Help</i> menu if you do not know how to use a certain function/block.</li> </ul> | <ul style="list-style-type: none"> <li>• <b>Voorbereiding:</b> Sien “What should I do in Week 6?” op SUNLearn.</li> <li>• <b>Instruksies:</b> Hierdie prakties moet individueel gedoen word – geen groepwerk. Skryf jou eie praktiese verslag en handig dit in op SUNLearn teen Vrydag 17:00. Sien SUNLearn vir instruksies.</li> <li>• <b>Neem kennis:</b> Bestudeer die Matlab/Simulink-dokumentasie deur na die <i>Help</i>-kieslys te gaan indien jy nie weet hoe om 'n sekere funksie/blok te gebruik nie.</li> </ul> |
|--|--|

## Background / Agtergrond

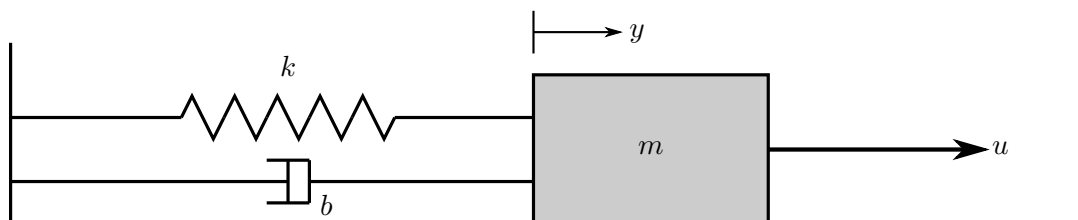
We have now looked at all the components of a SV compensator, including the state feedback, reference feedforward, and observer. In today's practical, we put all these components together and apply a compensator to a realistic plant simulation<sup>1</sup>. We also apply an alternative compensator – integral control – that ensures robust steady-state tracking.

Ons het nou na al die dele van 'n TV kompenseerder gekyk, insluitend toestand-terugvoer, verwysing-vorentoevoer, en waarnemer. Vandag kombineer ons alles en wend 'n kompenseerder aan op 'n realistiese simulatie van 'n aanleg<sup>2</sup>. Ons wend ook 'n alternatiewe kompenseerder – integraalbeheer – aan wat robuuste gestadigde-toestand-volging verseker.

## Assignment / Voorskrif

The plant that we use for today's practical has second-order dynamics, which could have been generated by the spring-damper setup shown below, with object mass  $m$ , force input  $u$ , measured displacement  $y$ , spring constant  $k$  and damping coefficient  $b$ .

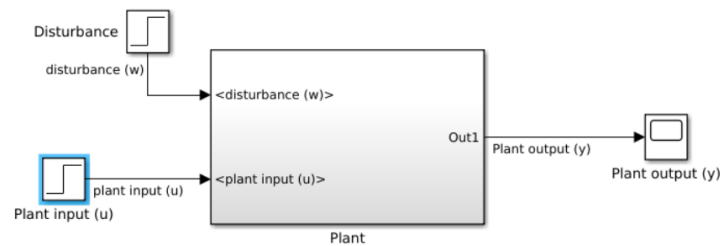
Die aanleg wat ons gebruik vir vandag se prakties het tweede-orde dinamika, wat gegenereer kan word deur die veer-demper opstelling hieronder, met objekmassa  $m$ , kragintree  $u$ , gemete verplasing  $y$ , veerkonstante  $k$  en demperkonstante  $b$ .



<sup>1</sup>We use Matlab for today's design, but make sure you can get the same values using hand calculations.

<sup>2</sup>Ons gebruik Matlab vir vandag se ontwerp, maar maak seker dat jy dieselfde waardes kan kry met die hand.

The “actual” plant for today’s practical is implemented in Simulink as a block as shown below.



Die “werklike” aanleg vir vandag se prakties is geïmplementeer as ’n Simulink blok soos getoon hieronder.

The plant has an input,  $u$ , and measured output,  $y$ , that contains noise. There is also a disturbance input,  $w$ , that models a disturbance force. Note that for a plant in the real world, you would know the input  $u$  and output  $y$ , but not the disturbance  $w$ . The disturbance is initially set to zero.

Die aanleg het ’n intree,  $u$ , en gemete uitree,  $y$ , wat ruis bevat. Daar is ook ’n steur-intree,  $w$ , wat ’n steur-krag modelleer. Neem kennis dat vir ’n aanleg in die regte wêreld sal jy weet wat die intree  $u$  en uitree  $y$  is, maar nie die steursein  $w$  nie. Die steur-intree is aanvanklik gestel op nul.

To set up this Simulink model, download the Simulink file `Prac3_setup.slx` as well as the Matlab data file `Prac3_parameters***.mat` and store them in the same folder<sup>3</sup>. In the Matlab file explorer, double-click on `Prac3_parameters***.mat` to load the parameters, open `Prac3_setup.slx`, and run the simulation.

Om hierdie Simulink-model op te stel, laai die Simulink-lêer `Prac3_setup.slx` asook die Matlab-datalêer `Prac3_parameters***.mat` af en stoor dit in dieselfde “folder”<sup>4</sup>. In die Matlab “file explorer”, dubbelklik op `Prac3_parameters***.mat` om die parameters te laai, maak `Prac3_setup.slx` oop, en hardloop die simulاسie.

<sup>3</sup>If you use Matlab/Simulink online, upload the files to one of your Matlab online folders.

<sup>4</sup>Indien jy Matlab/Simulink online gebruik, laai die lêers op na een van jou Matlab online “folders”.

# 1 System Identification / *Stelselidentifikasie*

## Problem statement: / *Probleemstelling:*

Create and analyse a SV model of the plant. Skep en analiseer 'n TV model van die aanleg.

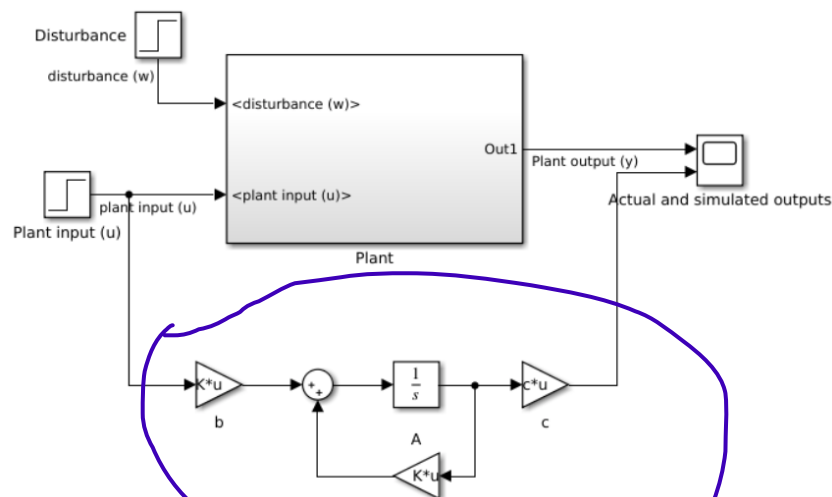
## Solution development: / *Ontwikkeling van oplossing:*

- (a) The second-order plant dynamics can be described by the transfer function Die tweede-orde aanlegdinamika kan beskryf word deur die oordragsfunksie

$$\frac{Y(s)}{U(s)} = \frac{b_2}{s^2 + a_1s + a_2} = \frac{b_2}{s^2 + 2\zeta\omega_n s + \omega_n^2}. \quad (1)$$

Apply a step to the input of the plant and identify the damping  $\zeta$ , natural frequency  $\omega_n$ , and parameter  $b_2$  from the response. Convert the transfer function to the control canonical SV form, build the SV plant model as a vector block diagram in Simulink, and compare this model to the actual plant. Adjust the parameters until the model responds similarly to the actual plant. Your Simulink model should look similar to the one below.

Wend 'n trap op die intree van die aanleg aan en identifiseer die demping  $\zeta$ , natuurlike frekwensie  $\omega_n$ , en parameter  $b_2$  van die uittree. Transformeer die oordragsfunksie na die beheer-kanoniese TV vorm, bou die TV aanlegmodel as 'n vektor-blokdiagram in Simulink, en vergelyk die model met die werklike aanleg. Verstel die parameters totdat die model se gedrag soortgelyk is aan die werklike aanleg. Jou Simulink diagram behoort soortgelyk te lyk as die een hieronder.




- (b) Determine the poles of the plant. Bepaal die pole van die aanleg.
- (c) Determine the controllability and observability of the plant by using a Matlab m-file (use the command `det`). Bepaal die beheerbaarheid en waarneembaarheid van die aanleg deur 'n Matlab m-lêer te gebruik (gebruik die `det`-bevel).

### 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:*

- 
- (a) How accurate is the plant model? In which aspects do the actual plant differ from the model?      Hoe akkuraat is die aanlegmodel? In watter aspekte verskil die werklike aanleg van die model?
- (b) How would you characterise the plant dynamics (from the step response and plant poles)?      Hoe sal jy die aanlegdinamika karakteriseer (vanaf die trapweergawe en aanlegpole)?
- (c) What can you conclude from the controllability and observability of the plant about the design of a compensator?      Wat kan jy sê vanaf die beheerbaarheid en waarneembaarheid van die aanleg oor die ontwerp van 'n kompenseerder?

## 2 Compensator with Reference Feedforward / *Kompenseerder met Verwysingsvorentoevoer*

### Problem statement: / *Probleemstelling:*

Design, implement and test a compensator with reference feedforward for this plant.      Ontwerp, implementeer en toets 'n kompenseerder met verwysing-vorentoevoer vir hierdie aanleg.

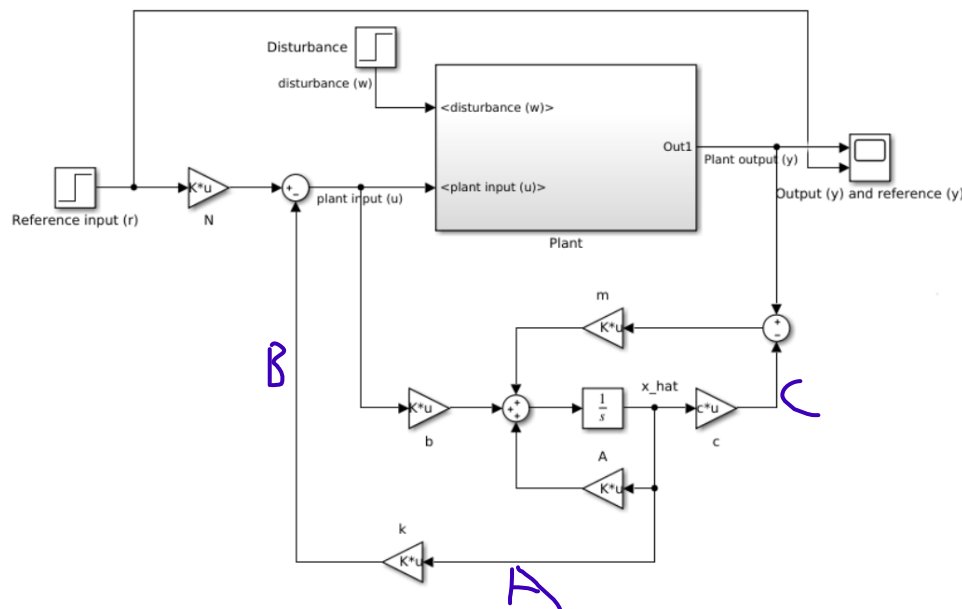
### Solution development: / *Ontwikkeling van oplossing:*

- (a) Design a SV compensator (i.e. calculate the state feedback gain  $\mathbf{k}$ , the observer gain  $\mathbf{m}$  and the reference feedforward gain  $N$ ) to control the plant; calculate these gains using a Matlab m-file. Satisfy the following closed-loop specifications:
- Closed-loop damping:  $\zeta = 0.5$
  - Closed-loop 2% settling time:  $t_s = 0.5$  s
  - Full state observer (estimator), where the observer's error dynamics will have 2 repeated poles with natural frequency 5 times faster than that of the desired closed-loop poles
  - Accurate tracking of a constant reference input  $r(t)$
- Ontwerp 'n TV kompenseerder (d.w.s. bereken die toestands-terugvoer-aanwins  $\mathbf{k}$ , die waarnemer-aanwins  $\mathbf{m}$  en die verwysing-vorentoevoer-aanwins  $N$ ) om die aanleg te beheer; bereken hierdie aanwinste deur 'n Matlab m-lêer te gebruik. Voldoen aan die volgende geslotelus-spesifikasies:
- Geslotelus damping:  $\zeta = 0.5$
  - Gelotelus 2% wegsterftyd:  $t_s = 0.5$  s
  - Volle toestandwaarnemer (afskatter) waar die waarnemer se foutdinamika herhalende pole sal besit waarvan die natuurlike frekwensie 5 keer vinniger is as dié van die verlangde geslotelus-pole
  - Akkurate volging van 'n konstante verwysings-intree  $r(t)$

- (b) Build the compensator in Simulink to control the *SV plant model* (adapt the Simulink model of Question 1; do not create a new model from scratch). Apply a step to the reference input  $r(t)$  and check that the desired specifications are satisfied by reading the overshoot and 2% settling time from the plot of the output  $y(t)$ . To double-check your work, look at the observer's error dynamics using an initial condition on the observer's integrator (e.g.  $[0.5 \ 0.0]^T$ ) while keeping the simulated plant's initial states at zero. Is the state error settling time as expected?
- (c) Now apply the compensator to control the *actual* plant; your Simulink model should look something like the diagram below.

Bou die kompenseerder in Simulink om die *TV aanlegmodel* te beheer (pas die Simulink-model van Vraag 1 aan; moenie 'n nuwe model skep nie). Wend 'n trap op die verwysings-intree  $r(t)$  aan en kyk of die verlangde spesifikasies bevredig word deur die oorskiet en 2% wegsterftyd te lees vanaf die plot van die uittree  $y(t)$ . Toets jou waarnemer se foutdinamika deur 'n begintoestand binne die waarnemer se integreerder te plaas (bv.  $[0.5 \ 0.0]^T$ ) terwyl jy die gesimuleerde aanleg se begintoestande hou by nul. Is die toestandsfout se wegsterftyd soos verwag?

Pas nou die kompenseerder toe om die *werklike* aanleg te beheer; jou Simulink-model behoort soos die diagram hieronder te lyk.



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

- (a) Apply a step to the reference input  $r(t)$  and plot the response of the compensator applied to the SV plant model as well as that of the compensator applied to the actual plant on the same graph.
- Wend 'n trap aan op die verwysingsintree  $r(t)$  aan en plot die gedrag van die kompenseerder toegepas op die TV aanlegmodel asook dié van die kompenseerder toegepas op die werklike aanleg op dieselfde grafiek.

- |  |  |
|--|--|
| (b) Apply a unit step to the disturbance input $w(t)$ (use a “step time” parameter of 5 second to distinguish its effect from the reference input). Plot the response of the compensator applied to both the SV plant model as well as the actual plant on the same graph. | Wend 'n eenheidstrap aan op die steurintree $w(t)$ (gebruik 'n “step time” parameter van 5 sekondes om die effek te skei van die verwysingsintree). Plot die gedrag van die kompenseerder toegepas op beide die TV aanlegmodel asook die werklike aanleg op dieselfde grafiek. |
|--|--|

### Conclusions: / *Gevolgtrekkings:*

- |  |  |
|--|--|
| (a) Does the compensator with the actual plant (without disturbance) comply with the specifications? Explain any differences between the expected and observed response. | Voldoen die kompenseerder met die werklike aanleg (sonder steursein) aan die spesifikasies? Verklaar enige verskille tussen die verwagte en waargenome gedrag. |
| (b) When the plant experiences a constant disturbance, does the system comply with the steady-state specifications? Explain the reason for this.                         | Wanneer die aanleg 'n konstante steursein ervaar, voldoen die stelsel aan die gestadigde-toestand spesifikasies? Verduidelik die rede hiervoor.                |

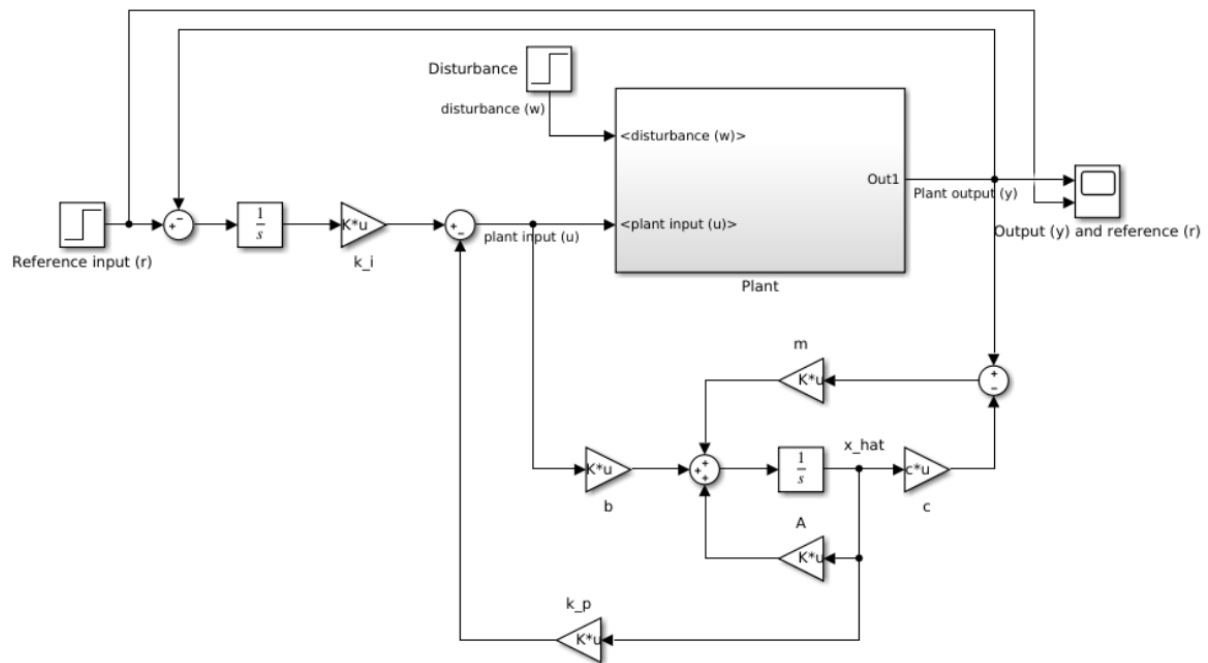
## 3 Compensator with Integral Control / *Kompenseerder met Integraalbeheer*

### Problem statement: / *Probleemstelling:*

Design, implement and test a compensator with integral control for this plant.	Ontwerp, implementeer en toets 'n kompenseerder met integraalbeheer vir die aanleg.
--	---

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

Repeat Question 2, but now use integral control. Use the same closed-loop poles as for Question 2 and place the <b>third closed-loop pole at <math>s = -12</math>. Calculate the state feedback gain <math>\mathbf{k}</math></b> for the augmented system using a Matlab m-file. <b>Keep the same observer as designed in Question 2.</b> Your Simulink model should look something like the diagram below.	Herhaal Vraag 2, maar gebruik nou integraalbeheer. Gebruik dieselfde geslotelus-pole as vir Vraag 2 en plaas die derde geslotelus-pool by $s = -12$ . Bereken die toestands-terugvoeraanwys $\mathbf{k}$ vir die uitgebreide stelsel deur 'n Matlab m-lêer te gebruik. Gebruik dieselfde waarnemer soos ontwerp in Vraag 2. Jou Simulink-model behoort iets soos die onderstaande diagram te lyk.
---	---



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

Repeat the experiments of Question 2.

Herhaal die eksperimente van Vraag 2.

Conclusions: / *Gevolgtrekkings:*

What can you conclude about integral control after comparing the results of this question with that of Question 2?

Watter gevolgtrekking kan jy maak oor integraalbeheer vanaf 'n vergelyking tussen die resultate van hierdie vraag en Vraag 2?