Challenge Deliverable 4
Answer form

4DB00 Dynamics and Control of mechanical systems
2019-2020

Name:	Jelle Langedijk
Student number:	1315358

Be aware of the TU/e Code of Scientific conduct. See: https://www.tue.nl/en/our-university/about-the-university/organization/integrity/scientific-integrity/.

Question a):

Is the transfer function $G_l(s)$ that you derived in deliverable 3 stable? Please explain how you came to this conclusion.

When running the command `P = pole(Gl)` in the command window, MATLAB saves all poles of $G_l(s)$ to the variable P. When requesting P, MATLAB outputs:

```
  -0.1250 + 1.3942i
  -0.1250 - 1.3942i
  -0.1250 + 1.3942i
  -0.1250 - 1.3942i
  -0.0501 + 0.0000i
  -0.0501 + 0.0000i
   0.0000 + 0.0000i
   0.0000 + 0.0000i
```

Since there are no poles that have a real part “>” zero, but there are poles with a real part equal 0, the transfer function can be classified as not stable.

NOTE: some poles are shown twice above, this is due to the fact that I did not use the `minreal()` command in MATLAB.

Question b):

Generate the step response of the transfer function $40 \cdot G_l(s)$ on the requested time interval, by utilizing the "step" function in Matlab. Add the requested plot here.

```
figure()
step(40*Gl);
xlim([0 100]);
```

Gives us the plot:

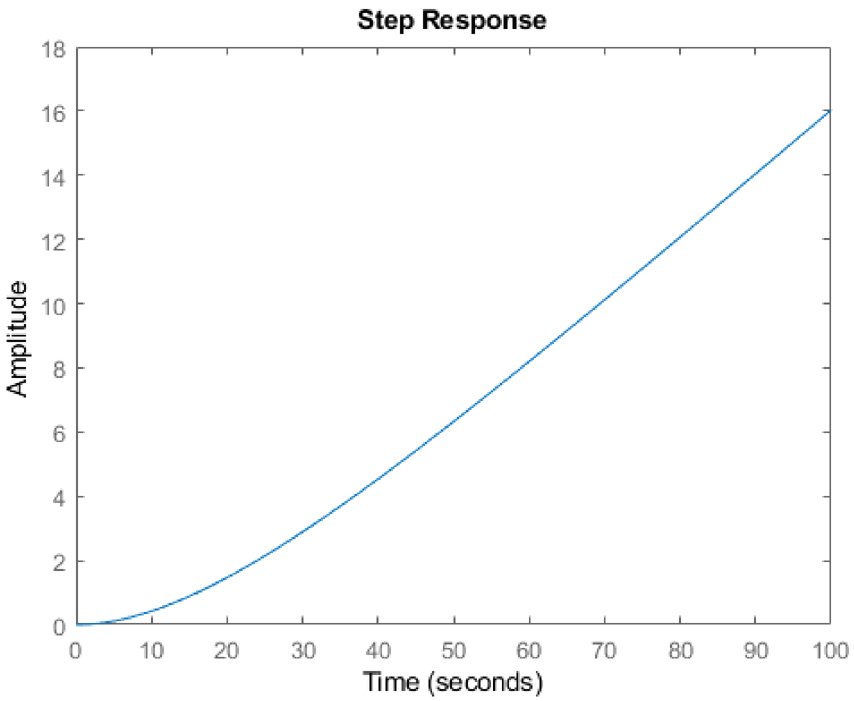

Figure 1: Step response of $40 \cdot G_l(s)$

Question c):

Determine a similar step response for the "virtual test setup", by utilizing the "Sim_Setup" request within the DOMS toolbox. Use the simulation conditions as described in the deliverable and add the requested plot here.

The code

```
t_sim      = [100];  % Simulation time
q_init     = [12.5 ; pi/2];  % Initial value for q
dq_init    = [0;0];  % Initial value for dq
s_pd_of_t = [0];  % Prescribed displacement a function of time t (e.g.
sin(t))
FA_of_t    = [40];  % VARIATE PER QUESTION

% Simulate the virtual test setup in Simulink (statement is complete):
Sim_Setup  = DOMS('Sim_Setup',Components,t_sim,...
                                    q_init,dq_init,s_pd_of_t,FA_of_t);

str4 = "$$ x_{l} [m] $$";

figure()
plot(Sim_Setup.t, Sim_Setup.qt(:,1) -
Components.parameters.L2.value*cos(Sim_Setup.qt(:,2)))
xlim([0 100]);
ylim([10 25]);
title("Position of the load" , 'Interpreter','latex')
xlabel(str1,'Interpreter','latex')
ylabel(str4,'Interpreter','latex')
```

Results in the following plot:

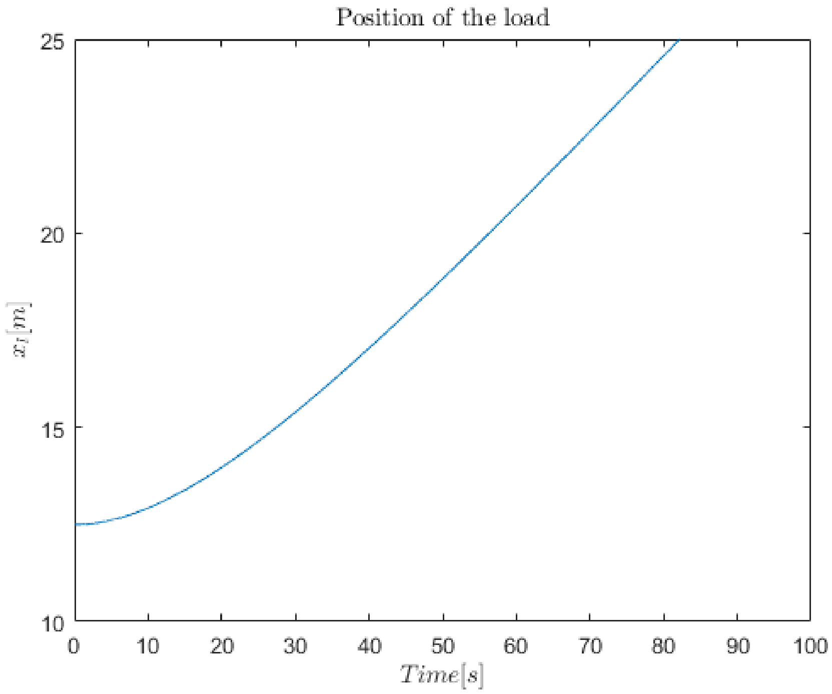

Figure 2: Position of the load, x_l

Question d):

Is there any difference between the responses that you observed in questions b and c? If so, can you explain this?

Since there is no transfer function of Sim_Setup, one cannot produce a step response of it. Nonetheless, the plot of x_l against time gives the same final slope and show a identical trajectory compared to figure 1, but is starts at a “height” of 12.5, determined by q(0), instead of 0, the starting “height” (by definition) of a step function.

Question e):

Consider the closed-loop interconnection between the controller that is described by $C(s) = 40$ and the crane that is described by the transfer function $G_l(s)$ that you derived in deliverable 3. Is the resulting closed-loop transfer function stable? Please explain how you came to this conclusion.

The transfer function for this closed-loop system is:

$$H_a = \frac{C(s)G_l(s)}{1 + C(s)G_l(s)}$$

First, to remove the cancelling pole/zero pairs or non minimal state dynamics, I use the `minreal(…)` command.

Then, using the `pole(…)` command in MATLAB, I obtain

```
-0.1250 + 1.3943i
-0.1250 - 1.3943i
-0.1249 + 1.3942i
-0.1249 - 1.3942i
-0.1250 + 1.3942i
-0.1250 - 1.3942i
-0.1250 + 1.3906i
-0.1250 - 1.3906i
-0.0250 + 0.0971i
-0.0250 - 0.0971i
-0.0501 + 0.0000i
-0.0501 + 0.0000i
-0.0501 - 0.0000i
```

Since there are no poles that have a real part “>” zero, the transfer function can be classified as stable.

Question f):

Generate the step response of the closed-loop transfer function as described in question e on the requested time interval, by utilizing the “step” function in Matlab. Add the requested plot here.

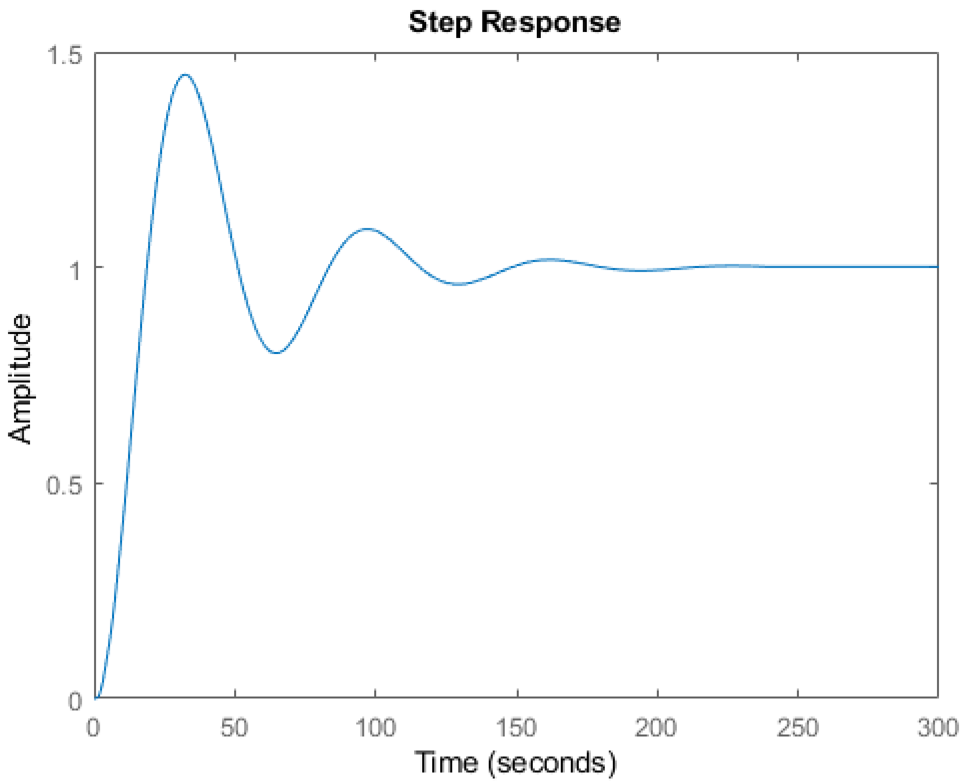

Figure 3: Step response of H_a .

Question g):

Simulate the three step responses of the closed-loop interconnection between the “virtual test setup” and a controller of the form $C(s) = P$, by utilizing the “Sim_Control” request within the DOMS toolbox. Use the simulation conditions as described in the deliverable and add the requested plot here.

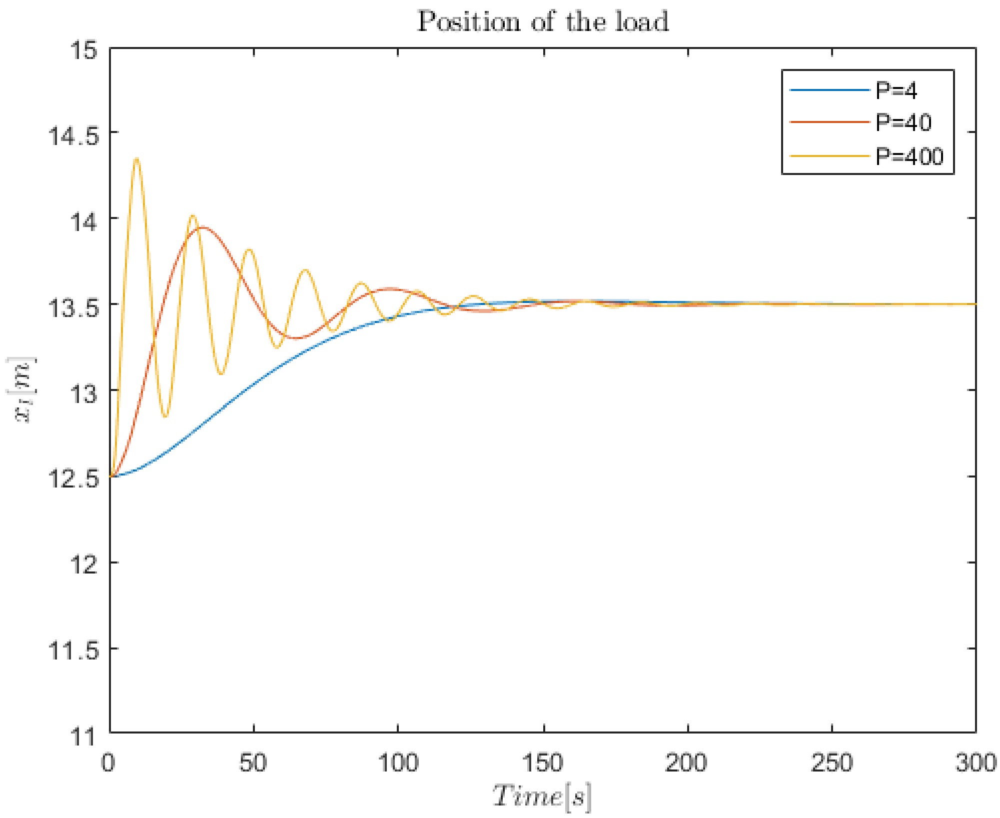

Figure 4: Step Responses for x_l for P = [4 ,40, 400].

Question h):

What is the effect of increasing and decreasing the gain P in question g? Based on your findings in this deliverable, would you recommend the use of a proportional controller $C(s) = P$ for this application?

A P-controller ($C(s) = P$) can be seen as (and technically *is*) a gain. The higher the constant value of the controller, the higher the excitation on the load (in this specific instance).

For P=4, we see that the signal finds its reference position in one smooth go, while for higher values of P, it finds this point far earlier, but keeps fluctuating around it. These fluctuations seem to be heavier with increasing P (and less heavy with decreasing P).

Of course, one does not want that the load fluctuates heavily around its desired position, the entire goal for a sway control is that this **not** happens.

With this statement in our mind, P=4 seems to be the most ideal controller, since the load finds its desired position without fluctuations. However, this seems to take for the P-controller more than two minutes, which is very impractical in real life applications.

Hence, the P-controller is not a suitable controller for our sway control.