Prelab 07 - 12.05.2020

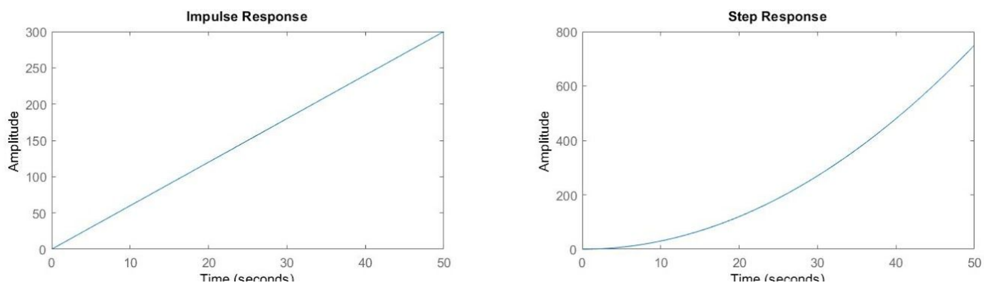
Constantin Gemmingen, Luke Peppard, Fabian Neumueller - Group 4.3

Q1)

- **Calculate the open-loop transfer function of this plant. You can use the linearized differential equation to calculate the transfer function.**
- **Show the plots as subplots next to each other and comment and explain the difference between step and impulse response.**

```
x’’= ⅗ *g*(alpha) = 30/5*(alpha)
-> Laplace => s^2*X = 30/5*(Alpha) with initial conditions set to zero
[x(0)=x’(0)=0]
Input: Alpha; Output: position X
=> Transfer Function G = X/Alpha = 30/(5*s^2)
```

Because the plate was tilted for an infinitely short time, the ball is put into motion and its velocity stays constant (no acceleration) because the plate is back in a neutral position. That’s why the impulse response is linear. With the step response the plate stays at an angle, thereby accelerating the ball and resulting in the step response seen below.



Impulse response and Step response of the ball dynamics system

Prelab 07 - 12.05.2020

Constantin Gemmingen, Luke Peppard, Fabian Neumueller - Group 4.3

Q2)

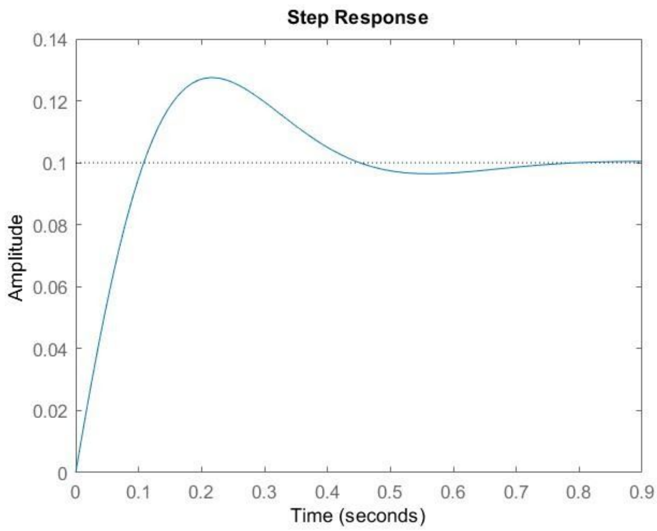
- **Is this response underdamped, critically damped or overdamped? What happens to your response if you increase Kd? If you consider our real system with visual Feedback, what could be the risk of increasing Kd too high?**

```
(X_des-X_meas)(K_p+1/s * K_I + s*K_D)=(Alpha)
X_meas = 30/(5*s^2) * (Alpha)

T = K_p + 1/s*K_I + s*K_D

(X_des-30/(5*s^2)*(Alpha))*T = Alpha
X_des * T = (30/(5*s^2) * T + 1) * (Alpha)
(Alpha)/X_des= T/((30/5*s^2)*T+1) = G
```

- **Hand in the plot for the two different Kd**

The system is underdamped. If we increase Kd, there is less overshoot and the response heads more gradually to a steady-state.

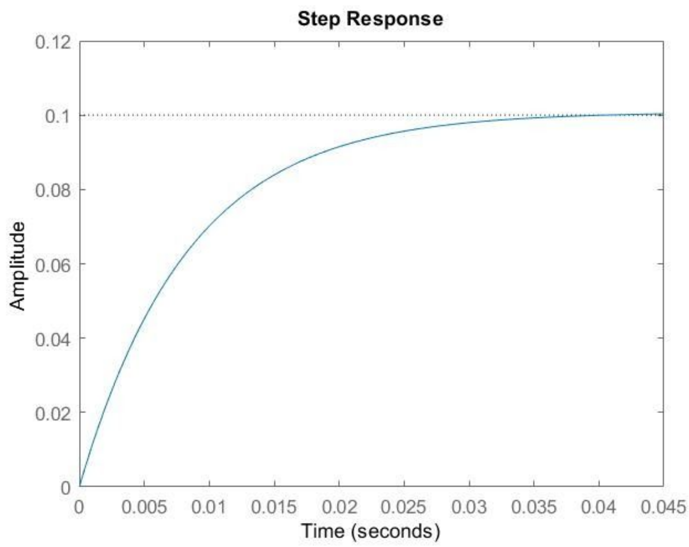
If we choose Kd too big, it takes too long to reach the steady-state and by then the ball would have already fallen off the plate.



Underdamped at Kd = 2 (Kp = 20, Ki = 1)

Prelab 07 - 12.05.2020

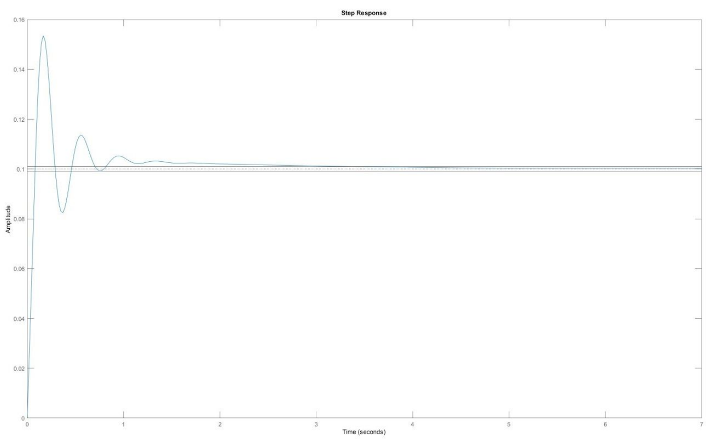
Constantin Gemmingen, Luke Peppard, Fabian Neumueller - Group 4.3



Critically damped at Kd = 20 (Kp = 20, Ki = 1)

Q3)

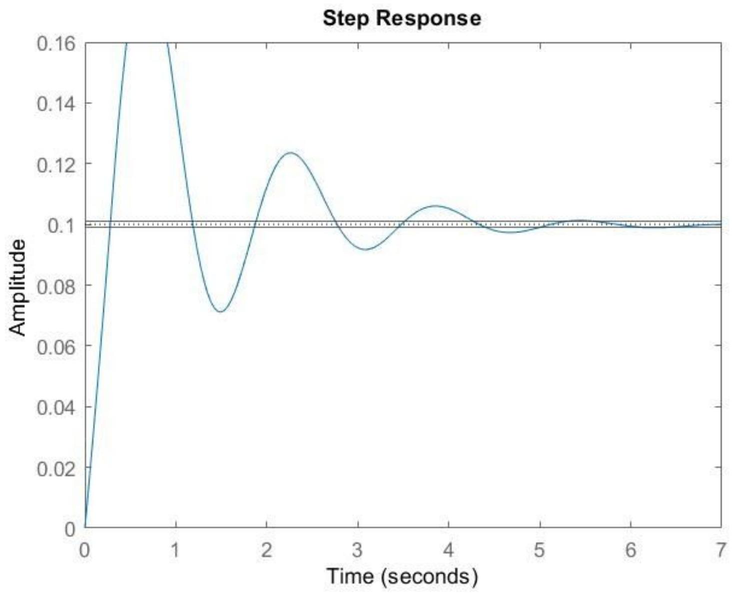
- **Hand in the plot for your report. What happens if the delay increases to 300 ms? Are you able to stabilize the system within 5 seconds?**



Feedback loop with delay time constant = 0.08 s (80 ms), Kp = 1.1, Ki = 0.2, Kd = 2;

Prelab 07 - 12.05.2020

Constantin Gemmingen, Luke Peppard, Fabian Neumueller - Group 4.3



Feedback loop delay time constant = 0.3 s (300 ms)

With the controller parameters from above the amplitude at 5 seconds too high for the controller to compensate. Using Kp = 0.4; Ki = 0.05; Kd = 0.54; we can stabilise the system at 5 s.

Q4)

- **How could you use this constraint to see if the PID controller's output exceeds the capabilities of our system?**

If the returned angle alpha exceeds 25° the system is saturated and it cannot be realised physically.

Q5)

- **When analyzing the system further you notice that the PID controller's output also exceeds the speed limits of your motors. What could be one of the problems you face?**

If the initial offset is too high and the counteracting speed exceeds the motor speed limit we cannot stabilise the system.

Lab 07

Table of Contents

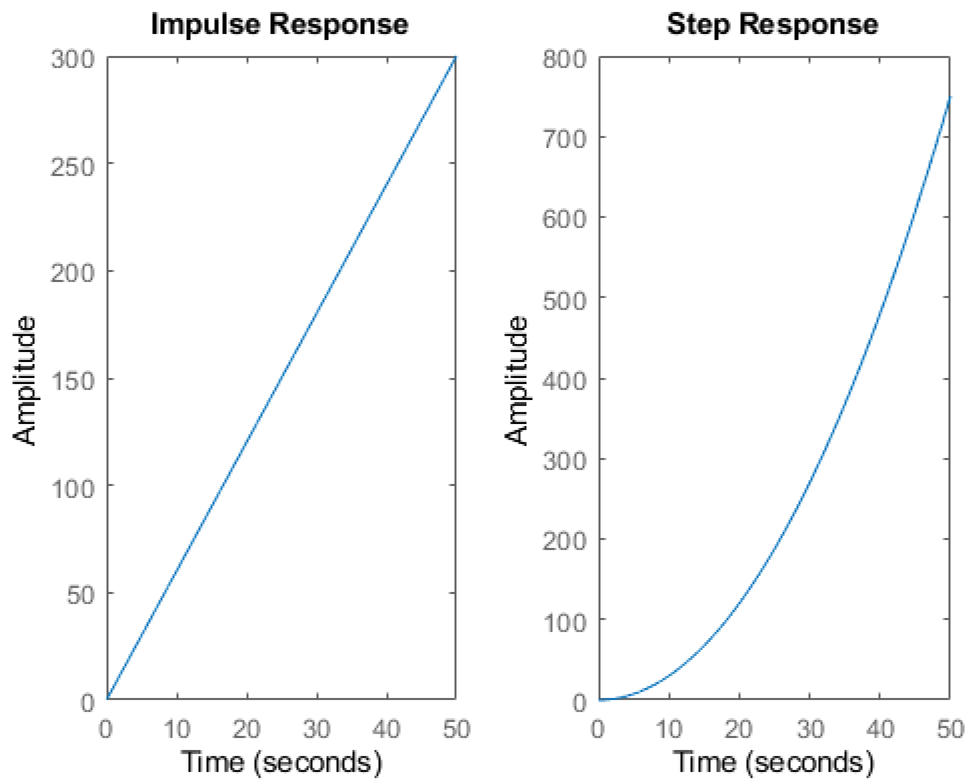
Constantin Gemmingen, Fabian Neumueller, Luke Peppard - Group 4.3

Prelab Q1

create the transfer function of the system (you should use the tf() function):

```
transBallDynamics = tf([30], [5 0 0])

% set step options to have a step-size of 0.1 instead of 1 (standard),
 use
% stepDataOptions()
opt_StepAmplitude01 = stepDataOptions('StepAmplitude', 0.1);

% open loop impulse and step response in one figure (two subplots):
figure(1)
subplot(1,2,1)
impulse(transBallDynamics)
subplot(1,2,2)
step(transBallDynamics, opt_StepAmplitude01)
```

```
transBallDynamics =

   30
  -----
  5 s^2

Continuous-time transfer function.
```

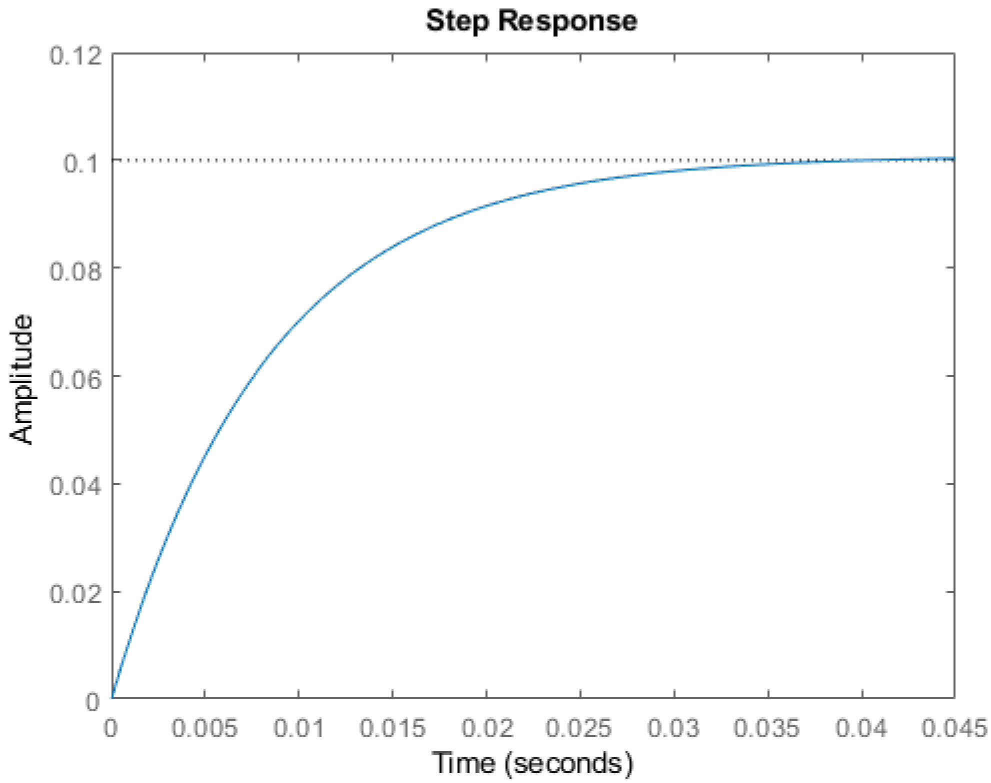


Prelab 02

Controller Parameters:

```
K_P = 20;
K_I = 1;
K_D = 2;
K_D_ = 20;

% create the transfer function of the Controller:
transPID_Controller = tf([K_D_ K_P K_I], [1 0])
transForward = transPID_Controller * transBallDynamics;

% create the transfer function of the feedback loop
% trans_Q2_feedback = tf([6*K_D 6*K_P 6*K_I], [1 6*K_D 6*K_P 6*K_I]);
trans_Q2_feedback = feedback(transForward, 1)

% Plot the closed loop step response
figure(2)
step(trans_Q2_feedback, opt_StepAmplitude01)
% under damped using K_d = 2, and critically damped using K_d = 20;


transPID_Controller =

  20 s^2 + 20 s + 1
```

```
  -----------------
          s

Continuous-time transfer function.


trans_Q2_feedback =

      600 s^2 + 600 s + 30
  ----------------------------
  5 s^3 + 600 s^2 + 600 s + 30

Continuous-time transfer function.
```



Prelab 03

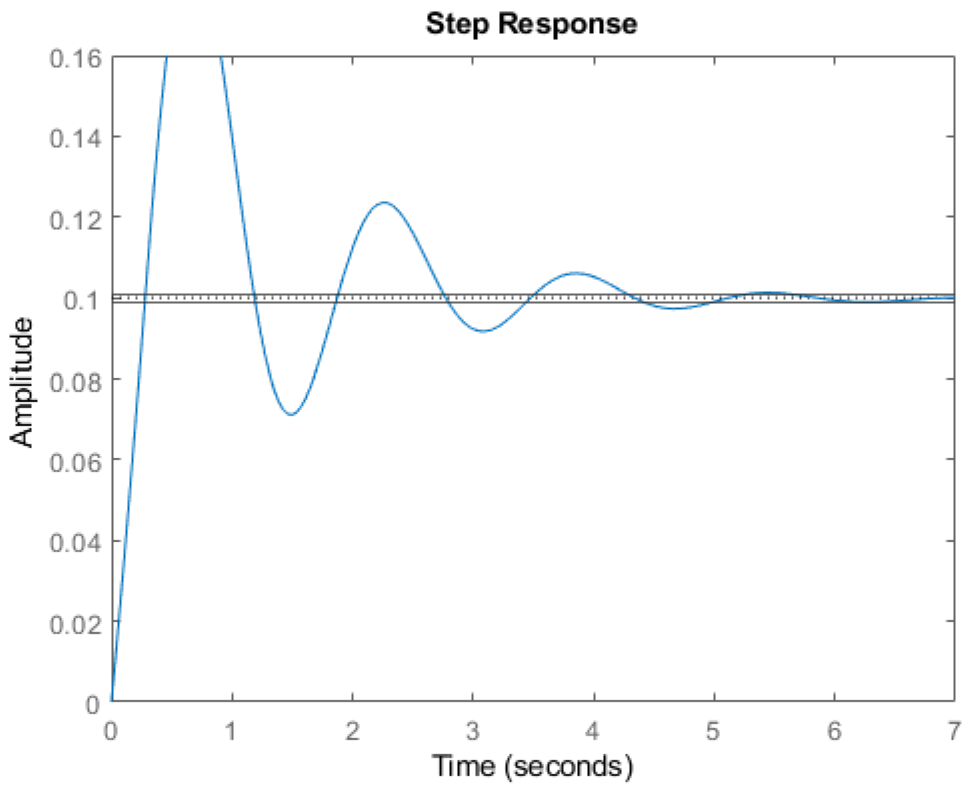
controller constants:

```
Kp = 0.4;
Ki = 0.05;
Kd = 0.54;

% create the transfer function of the Controller
transPID_ControllerDelay = tf([Kd Kp Ki], [1 0])
```

```
% Variables for delay:
DELAY=0.08; % 80ms
DELAY_=0.3; % 300ms

% create the transfer function of the delay in the system
% trans_Q3_delay = exp(-DELAY * s);
transDelay = tf(1, 1, 'InputDelay', DELAY_)


% create the transfer function of the feedback loop with delay
transForwardDelay = transPID_ControllerDelay * transBallDynamics;
transFeedbackDelay = feedback(transForwardDelay, transDelay)

% Plot the closed loop step response of the system with delay
y_top = 0.16;
y_bot = 0;
time_bot = 0;
time_top = 7;

figure(3)
step(transFeedbackDelay, opt_StepAmplitude01)
axis([time_bot time_top y_bot y_top])
yline(0.1+0.001)
yline(0.1-0.001)
% check design criteria for "steady state" error at 5 seconds
```

```
transPID_ControllerDelay =

  0.54 s^2 + 0.4 s + 0.05
  -----------------------
             s

Continuous-time transfer function.


transDelay =

  exp(-0.3*s) * (1)

Continuous-time transfer function.


transFeedbackDelay =

  A =
           x1      x2      x3
   x1   -3.24    -2.4    -0.3
   x2       1       0       0
   x3       0       1       0

  B =
        u1
   x1    2
```

```
 x2   0
 x3   0

C =
       x1    x2    x3
 y1  1.62   1.2  0.15

D =
     u1
 y1   0

(values computed with all internal delays set to zero)

Internal delays (seconds): 0.3

Continuous-time state-space model.
```



Prelab 04

```
% Hint:
% - Make use of the linearized differential equations
GRAVITY = 9.81;

syms x(alpha)
eqation = diff(x, alpha) == 3/5 * GRAVITY;
```

```
solution = dsolve(eqation);
% - Filter out the first data points from the time delay to get rid of
 the
% incontinouity at the beginning of the sytem-response (otherwise the
 derivative explodes)
[x_output, t_output] = step(transFeedbackDelay, opt_StepAmplitude01);


% - Make sure to use the right units (rad, degree, ...) --> check if
 your
%  values make sense if you are not sure. E.g. a free-falling ball
% has 9.81 m/s^2 acceleration
```

Published with MATLAB® R2020a