Assignment B

By: Quinty van Dijk and Mitchel Mulder

Problem 1 **1) for the first link, plan a linear segments with parabolic blends (LSPB) trajectory with a maximum velocity of 10π/17 rad/s that satisfies the given constraints.**

The created trajectory of q1 is shown in figure 1

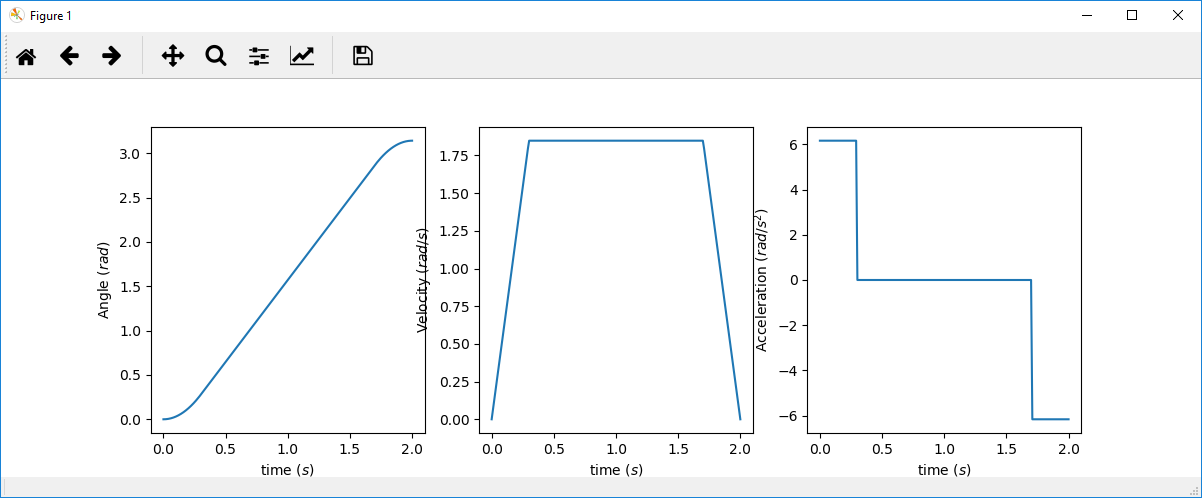


Figure 1 LSPB trajectory of the first link

**2) For the second link, we consider a minimum-time trajectory that satisfies all given constraints.**

Compute the acceleration of this minimum-time trajectory:

tf = 2s q(0) = 0 q(tf) = rad v(0) = v(tf) = 0 rad/s

The trajectory is generated in python and the plot is shown in figure 2

**3) Plot the obtained trajectories for both joints**

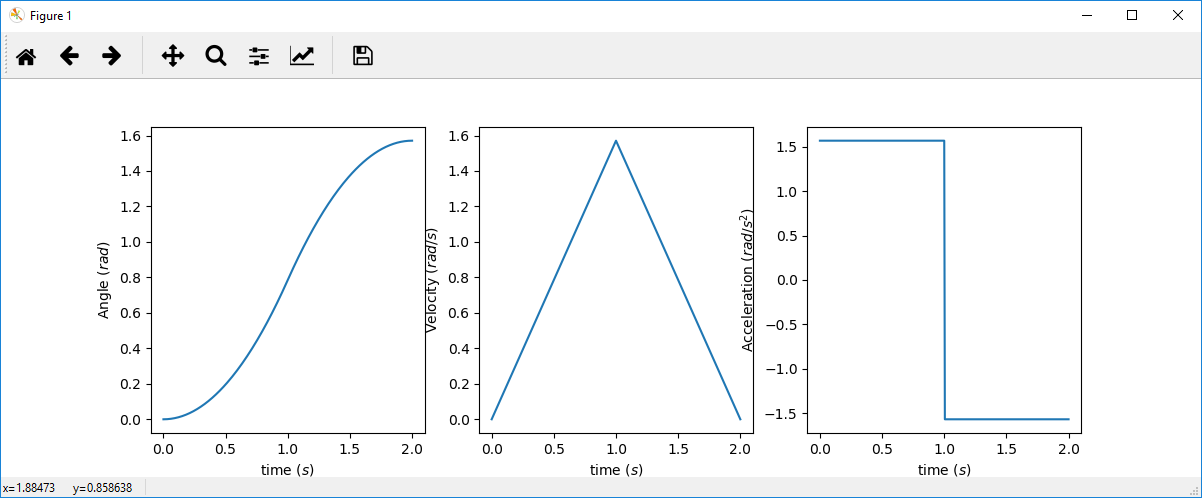


Figure 2 Minimum time trajectory of the second link

# Problem 2

**1) Plan two different trajectories using "via points" that avoid the obstacle. Ensure that the trajectory is continuous in both velocity and acceleration. Explain and motivate how you have constructed these trajectories.**

Viapoint 1 is causing link1 to go to -pi/2 and link2 to pi/2  
Viapoint 2 is causing link1 to go to pi/2 and link2 is already at its qf.

The first movement is moving link1 down and link2 to its qf, This way link1 can move to its qf without the arm of link2 hitting the dot. The other way (first link1 to its qf and then link2 to its qf) is not possible because the dot is very close to link2, so link1 has to make space for link2 to move. The points q0, qvp and qf can be seen in figure 3, 4 and 5 respectively

**Viapoints animation**

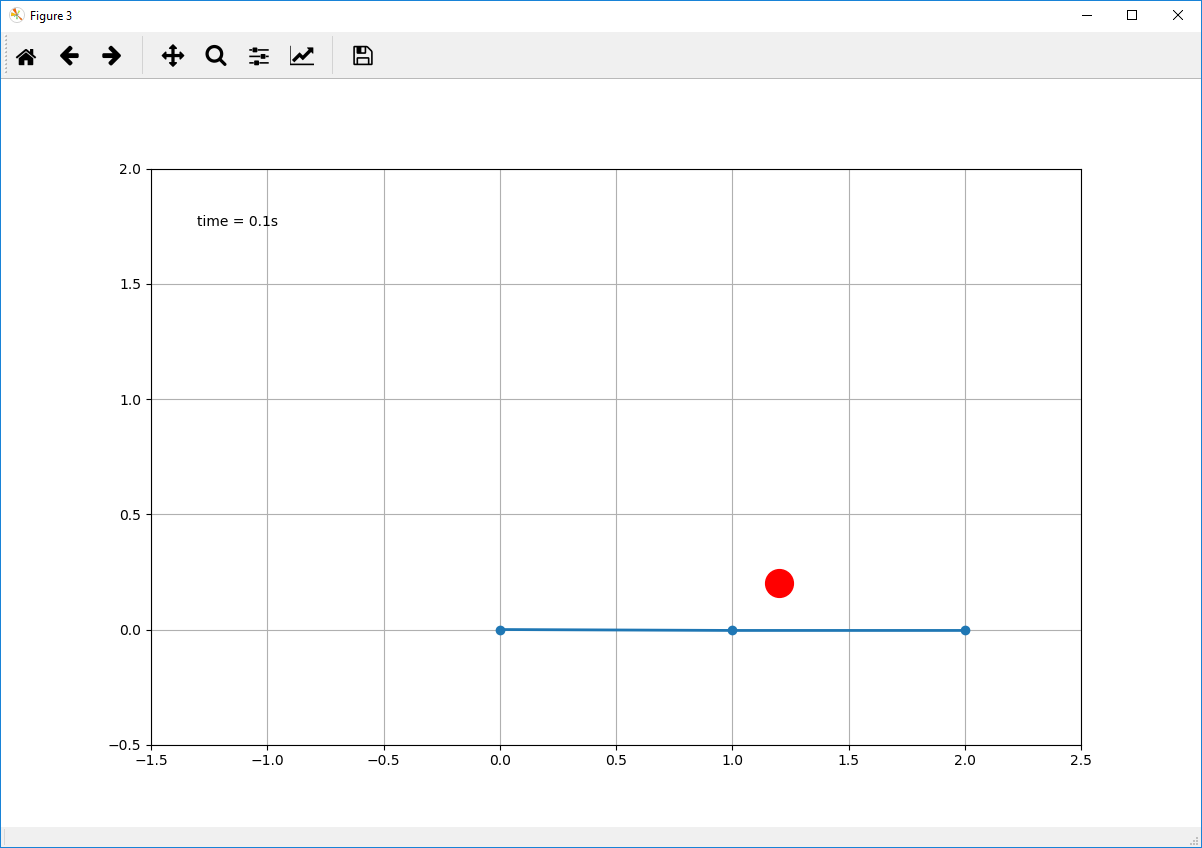


Figure 3 Animation at q0

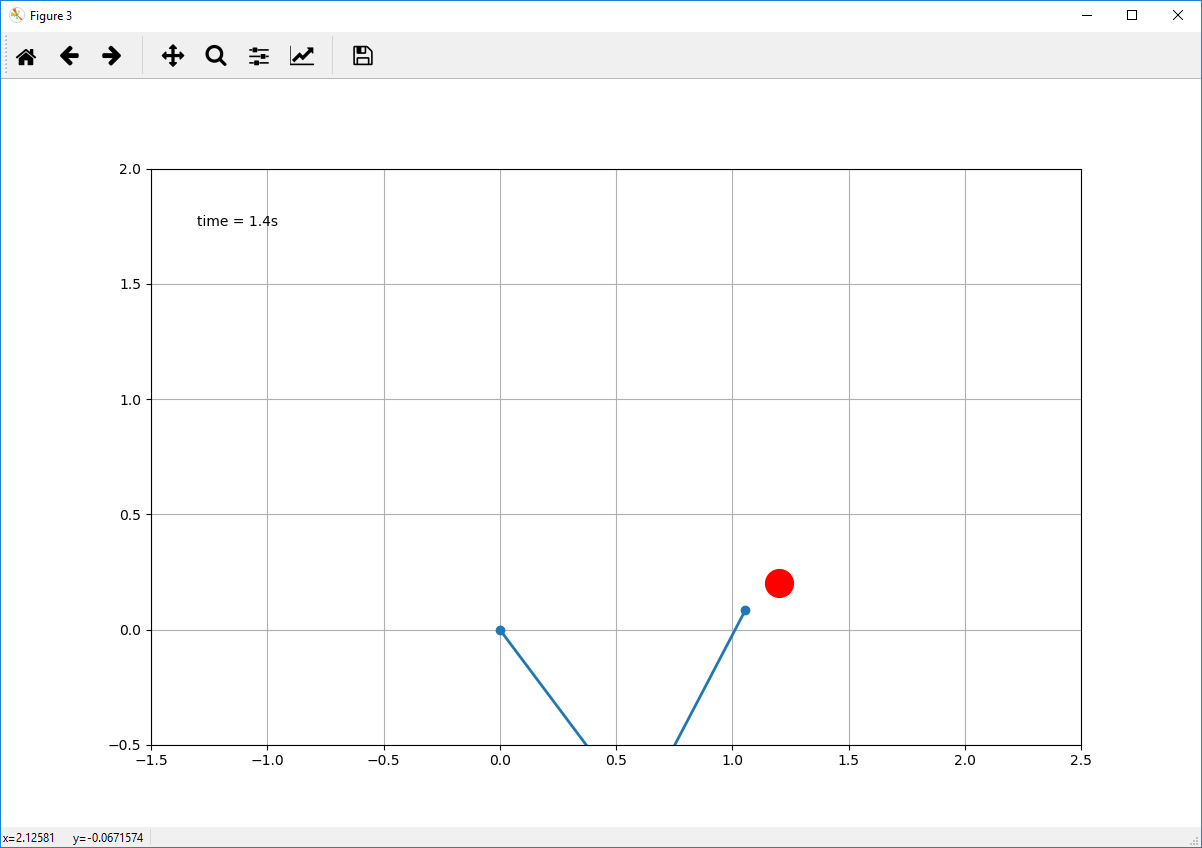


Figure 4 Animation at qvp

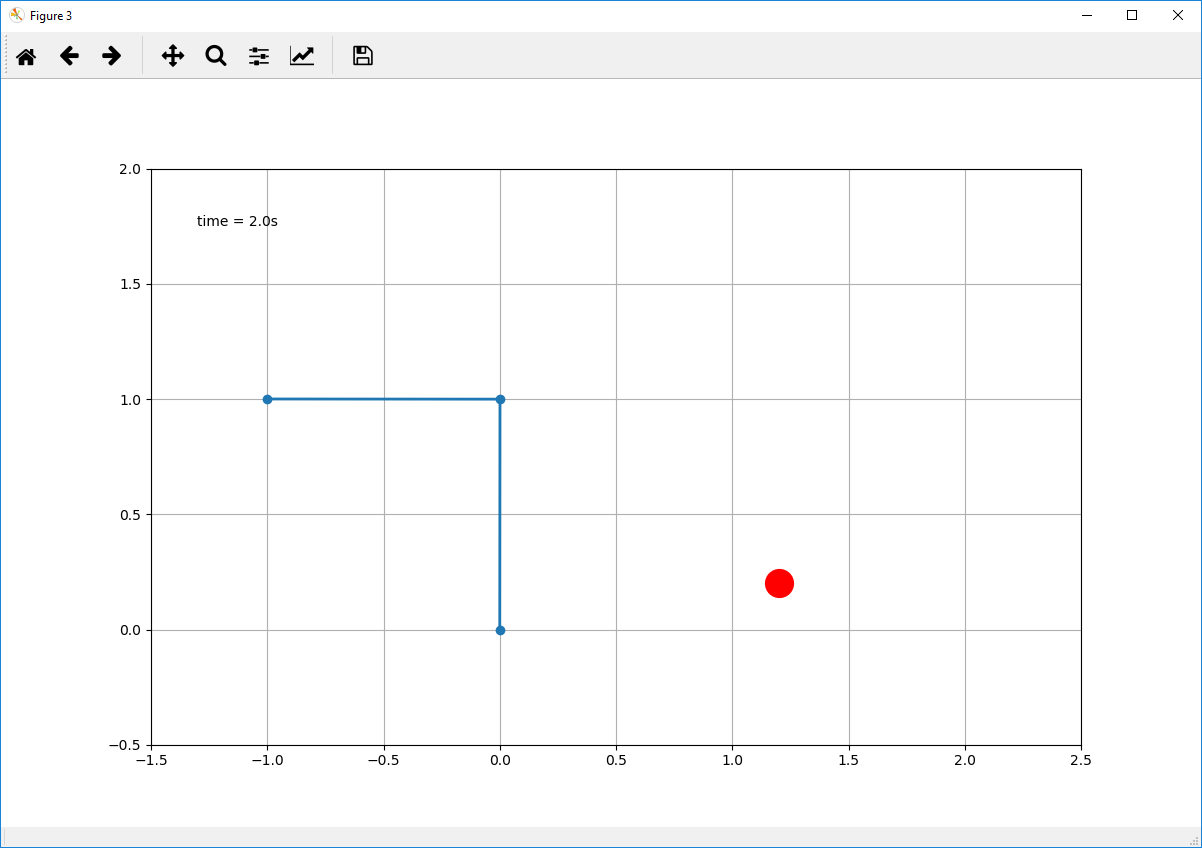


Figure 5 Animation at qf

**2) Plot the obtained trajectories for both joints and animate the robot with the planned trajectories**

Three points of the animation can be seen at the end of 1)

The trajectory of the first link is plotted in figure 6 and the trajectory of the second link in figure 7

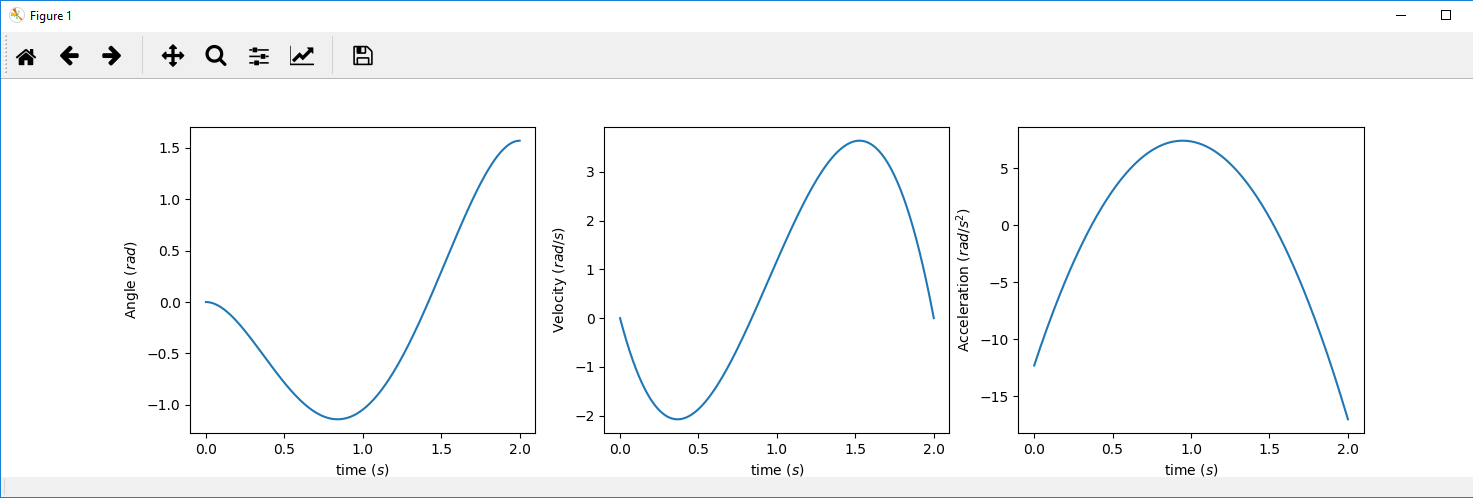


Figure 6 Viapoint trajectory of the first link

The first link has five constraints:

* q0, qf, qvp, v0 and vf

This means we need a 4th grade polynomial. (Quartic)

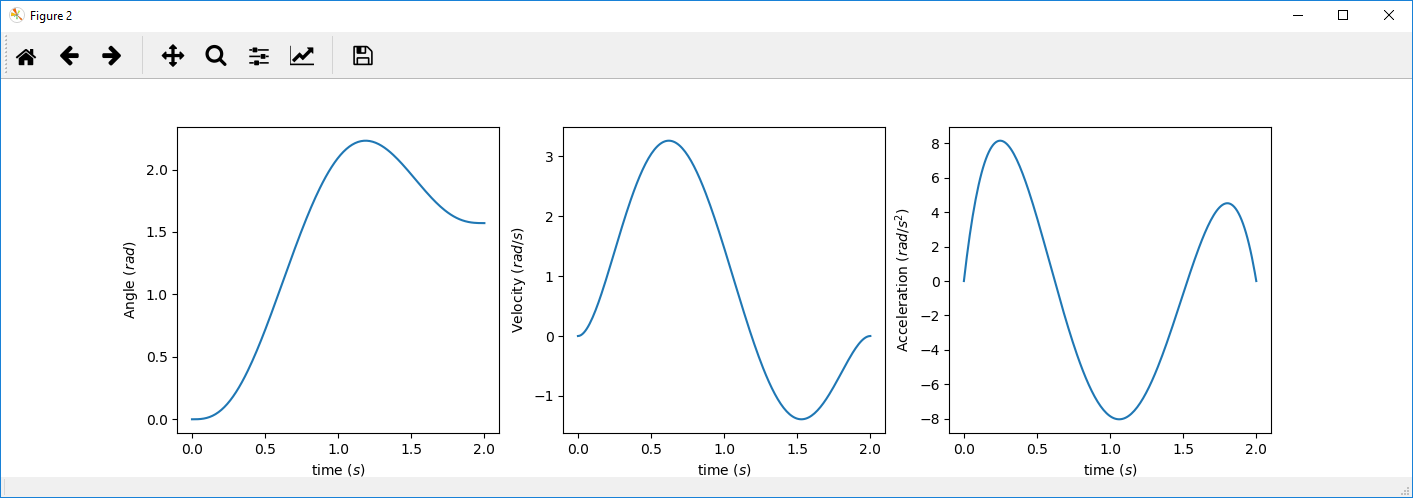


Figure 7 Viapoint trajectory of the second link

The first link has seven constraints:

* q0, qf, qvp, v0, vf, a0 and af

This means we need a 6th grade polynomial. (Sextic)

**3) Compare the two generated trajectories.**

The biggest difference between the two trajectories is that the 2nd link had two extra constraints, that being the begin and ending acceleration being 0. They both overshot at their via point, because there is no velocity constraint at the via point

# Problem 3

**1) Design three PD controllers for system G(s) = 1 / (s(3s+3))**

**The first one with Kp = 16 and Kd = 7**

**The second one placing the closed-loop system poles in s = −3 and s = −1  
PD2:**G(s) =   
Y(s) = => J=3, B=3

Poles at S = -1 and S = 3

3(S+1)(S+3) = 3(S2 + 4S + 3) = 3S2 + 12S + 9

3S2 + 12S + 9 = 3S2 + (3+KdS) + Kp

Kd = 12-3 = 9, Kp = 9

**The third one resulting in a closed-loop response with ω = 20 and ζ = 1  
PD3:**  
 = 20  
ζ = 1

3\*(2ζ𝜔S) = 3+KdS  
120S = 3+KdS  
Kd = 117  
Kp = 3\*(𝜔2)= 3\*400 = 1200

**2) Compare the different responses of the three compensators for a constant reference input.**

The output of the three controllers with a step input of 10 and no disturbance are plotted in figure 8.  
The higher Kp causes a steeper climb and more overshoot, but not necessarily a faster stable state. The critically dampened controller has no overshoot

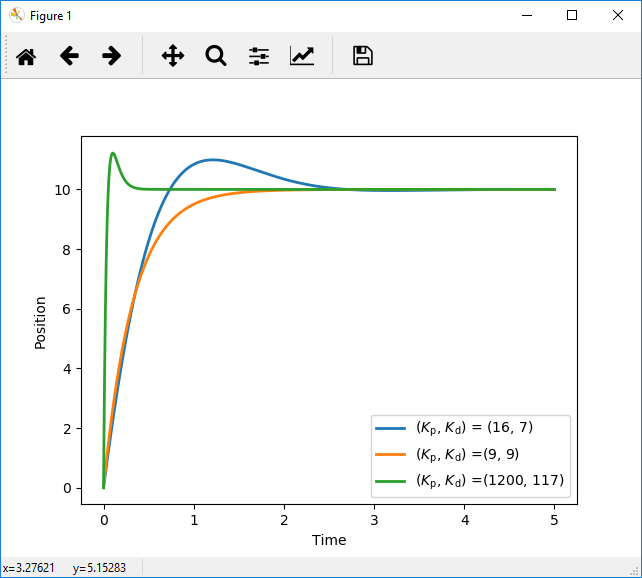


Figure 8 PD controller with three different configurations

**3) Test the performance of the previously designed PD controllers in the presence of a constant disturbance d = 20.**

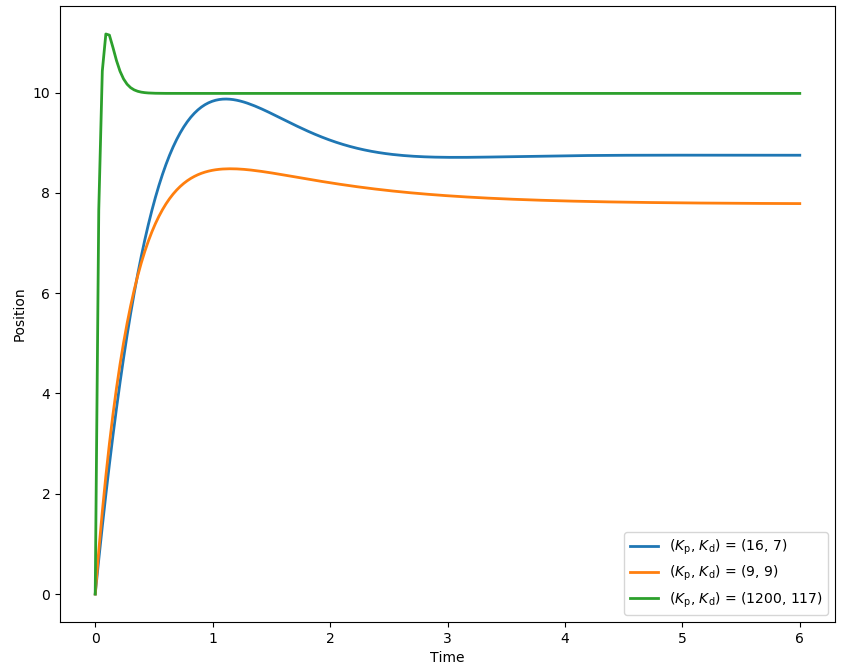


Figure 9 The same PD controllers, but now with disturbance d=20

Since the steady-state error for a PD controller is calculated by d/Kp. The signal with the higher Kp values are less affected by the disturbance as shown in figure 9.

**4) Compute for each of the controllers the steady-state error Ess in the presence of this  
disturbance.**

* Blue signal: Ess = -d/Kp = -20/16 = -1.25
* Orange signal: Ess = -d/Kp = -20/9 = -2.22
* Green signal: Ess = -d/Kp = -20/1200 = -0.01667

**5) Design a new controller that completely removes the steady-state error.**

A PID controller would be sufficient to eliminate the Ess over time.

Y(s) =

Kp = 16, Kd = 7, Ki = 16

# Problem 4

**Analyse the performance of each of the following control strategies:**

**1) PD control**

**2) PD + feedforward control**

**3) PID control**

**4) PID + feedforward control**

* Constant reference, no disturbance

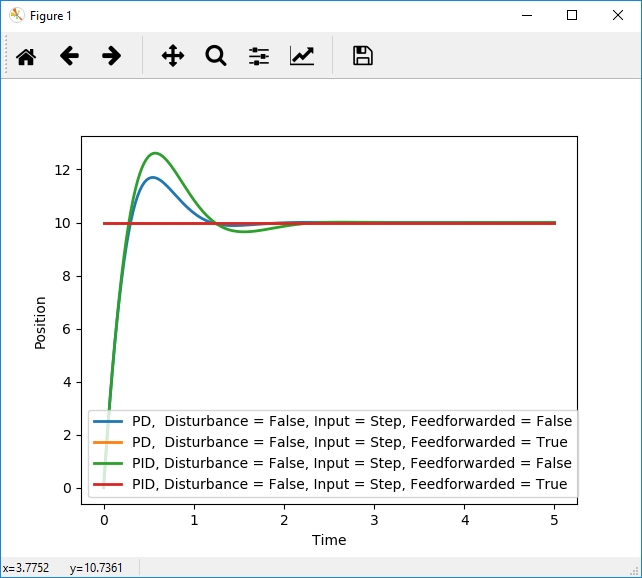


Figure 10 Constant reference, no disturbance

The four controllers are plotted in the same graph (figure 10).

Since there are no limitations to the acceleration and the speed, the feedforward controllers(both PID and PD because of the absence of a disturbance)follows the reference signal perfectly and instantly. The non-feedforwarded controllers need to ramp up their errors so that takes a bit. The PID controller is a bit slower with the same Kp and Kd Ki = Kp. Ki could have been a more suitable value.

* Constant reference, constant disturbance

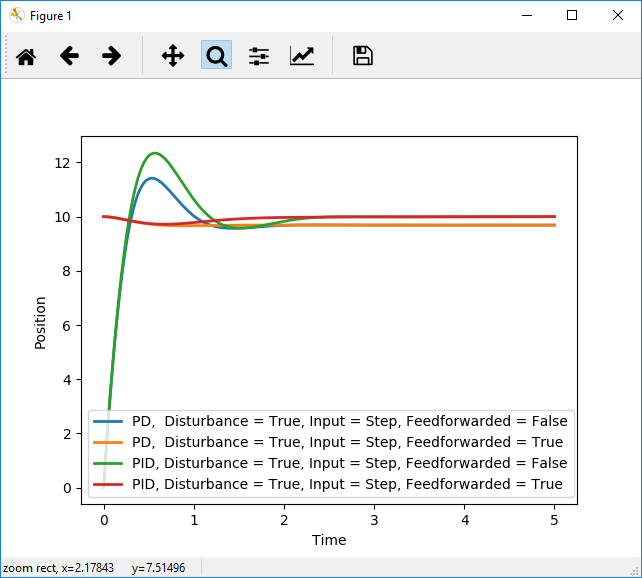


Figure 11 Constant reference, constant disturbance

The four controllers are plotted in the same graph (figure 11).

Just like the signal without disturbance. The feedforwarded controllers follow the signal instantly. The PD however follows it with a steady-state error. Which is to be expected. The feedforwarded PID can be seen that it also dips, but the integrating action fixes the steady-state error after a while

* Constant reference, ramp disturbance

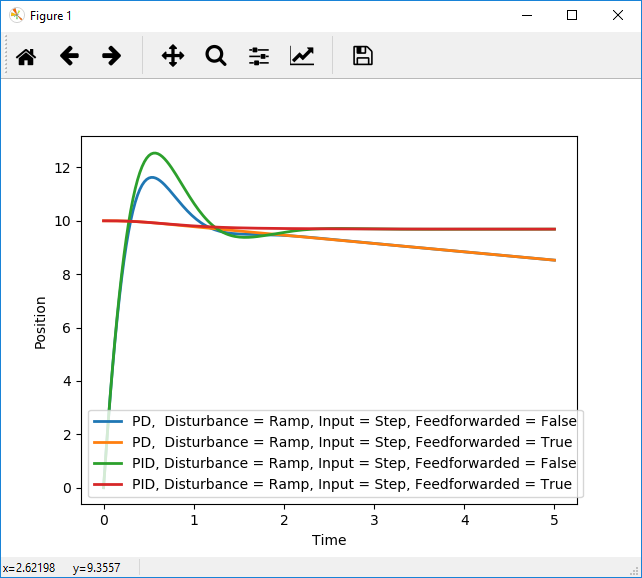


Figure 12 Constant reference, ramp disturbance

The four controllers are plotted in the same graph (figure 12).

Here the steady-state error increases overtime due to the ramping disturbance, so the PD controllers will drift further and further away. The PID controllers fix themselves over time

* Ramp reference, no disturbance

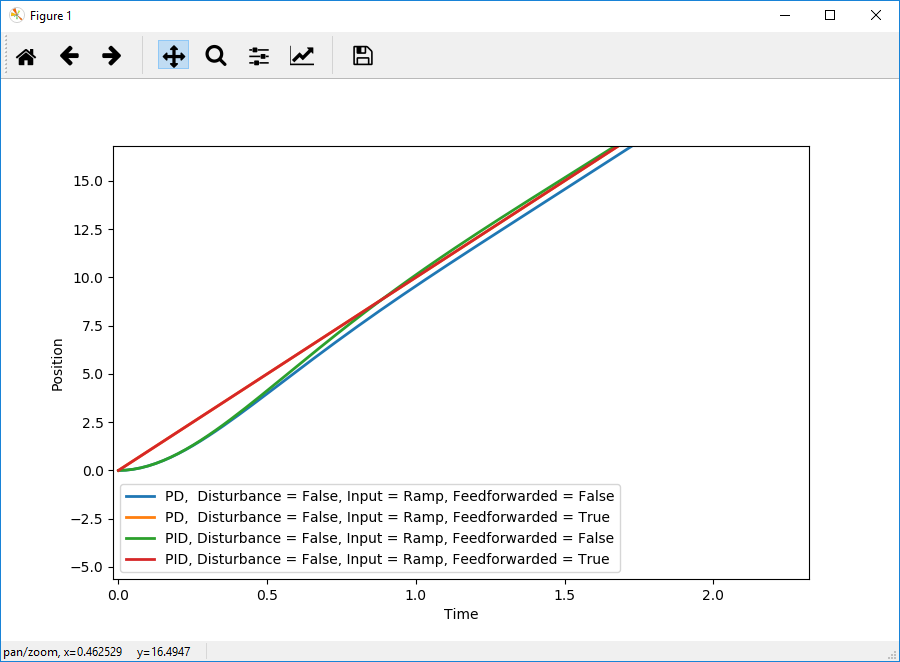


Figure 13 Ramp reference, no disturbance

The four controllers are plotted in the same graph (figure 13).

Non-feedforwarded PD controller can’t keep up and creates a steady-state error.  
Non-feedforwarded PID controller fixes that steady-state error due to the integrating action.  
Both feedforwarded controllers follow the input perfectly from the beginning because of the predicting action

* Ramp reference, constant disturbance

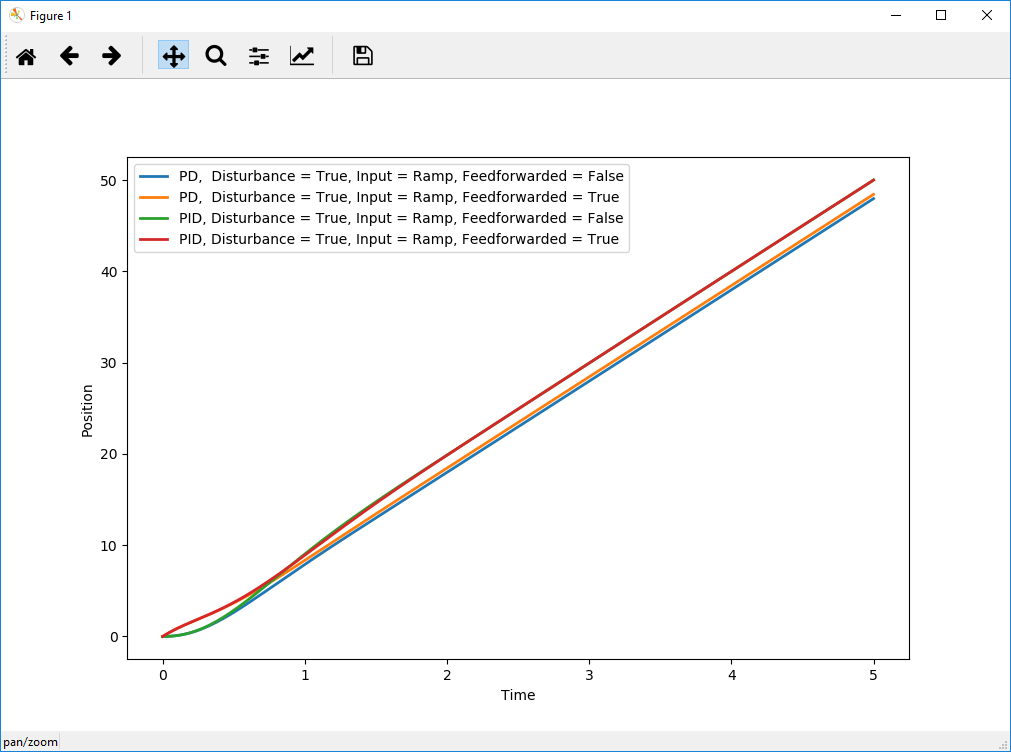


Figure 14 Ramp reference, constant disturbance

The four controllers are plotted in the same graph (figure 14).

The non-feedforwarded PD controller now has two issues with this signal and will add up both ramping and steady-state errors.  
The feedforward PD controller still suffers from the steady-state error from the disturbance, but the error is a bit less, because of the predicting capability of the feedforward controller.

Both PID controllers can still handle this signal. Although the non-feedforwarded PID controller oscillated and lagged quite a bit in the beginning

* Second-order polynomial reference signal, no disturbance

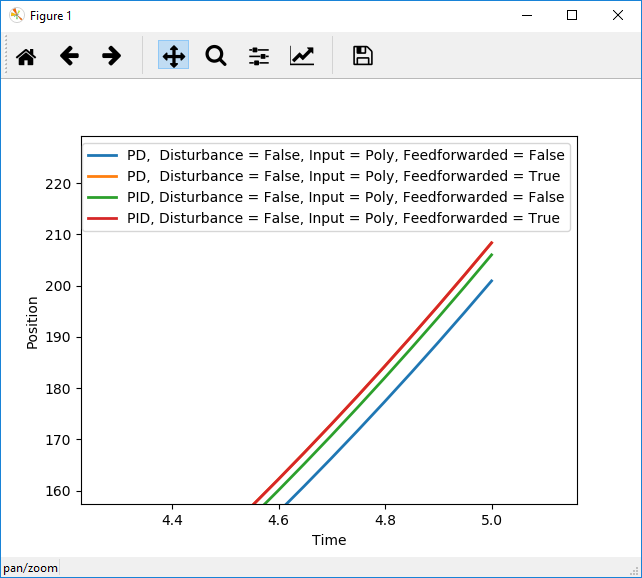


Figure 15 Second-order polynomial reference signal, no disturbance

The four controllers are plotted in the same graph (figure 15).

Here it is visible that, just like the ramped reference signal that the both non-feedforwarded controllers are starting to lag behind, but more and move over time because the signal accelerates. The feedforwarded PD controller follows the feedforwarded PID controller because of the absence of a disturbance.

* Second-order polynomial reference signal, constant disturbance

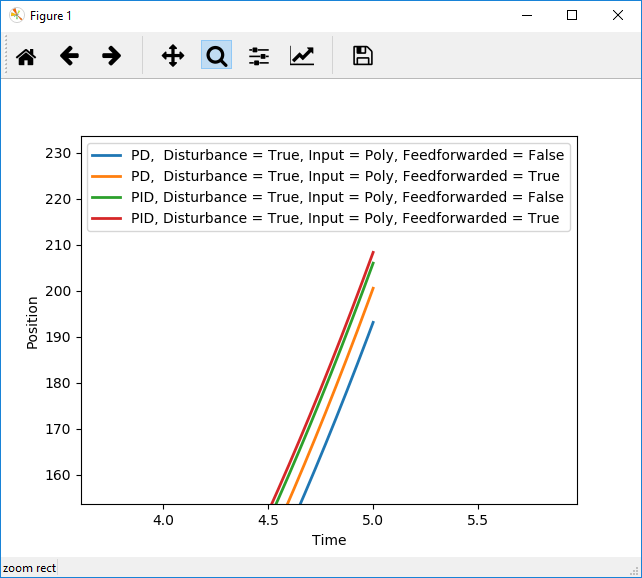


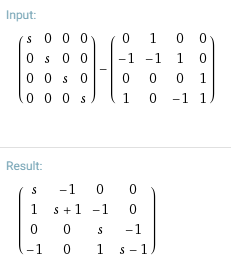
Figure 16 Second-order polynomial reference signal, constant disturbance

The four controllers are plotted in the same graph (figure 16).

Increased the disturbance to 500 to get a better look at the steady-state errors of the PD controllers and the lagging of the non-feedforwarded controllers. Just like the ramped input, because the input is changing over time. The non-feedforwarded PID controller will lag as well as the PD controller. The non-feedforwarded PD controller also has the steady-state error added to the lagging. The Feedforwarded PID controller catches up after a while to eliminate the steady state error.

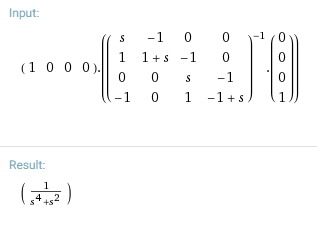
## Problem 5

**1): Derive a state-space model describing the relation between u(t) and y(t).**



<https://www.wolframalpha.com/input/?i=(%7B%7Bs,+0,+0,+0%7D,+%7B0,+s,+0,+0%7D,+%7B0,+0,+s,+0%7D,+%7B0,+0,+0,+s%7D%7D+-+%7B%7B0,+1,+0,+0%7D,+%7B-1,+-1,+1,+0%7D,+%7B0,+0,+0,+1%7D,+%7B1,+0,+-1,+1%7D%7D)>

:



[https://www.wolframalpha.com/input/?i=%7B1,0,0,0%7D+\*+%7B%7Bs,+-1,+0,+0%7D,+%7B1,+1+%2B+s,+-1,+0%7D,+%7B0,+0,+s,+-1%7D,+%7B-1,+0,+1,+-1+%2B+s%7D%7D%5E-1\*%7B%7B0%7D,%7B0%7D,%7B0%7D,%7B1%7D%7D](https://www.wolframalpha.com/input/?i=%7B1,0,0,0%7D+*+%7B%7Bs,+-1,+0,+0%7D,+%7B1,+1+%2B+s,+-1,+0%7D,+%7B0,+0,+s,+-1%7D,+%7B-1,+0,+1,+-1+%2B+s%7D%7D%5E-1*%7B%7B0%7D,%7B0%7D,%7B0%7D,%7B1%7D%7D)

**2) Apply a step input to both the model in state-space form and the model in the Laplace domain. Are both responses identical?**

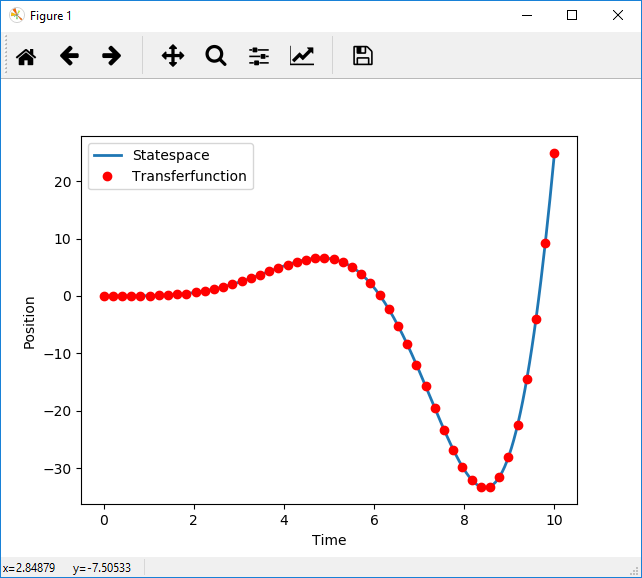


Figure 17 Both statespace and the derived transferfunction reacting to a step input

According to the graph in figure 17, you can see that both controllers react the same to the step input. Both are a bit instable though.

# Problem 6

We solved problem 6 with handwriting to avoid having to create the many matrices in word. The figures 18 until figure 22 are the answers from the notebook

**1) Derive a state-space model describing the relation between u(t) and y(t)**

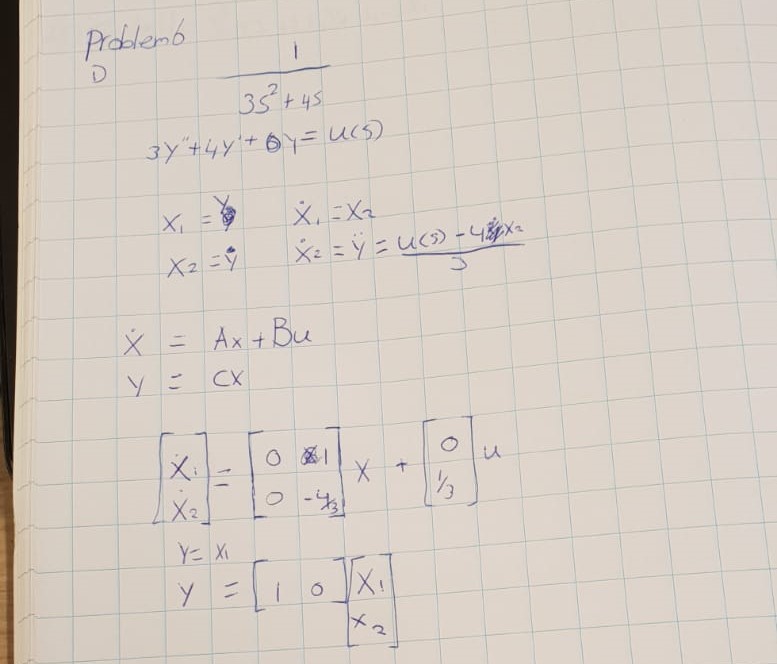


Figure 18 Answer to 1)

**2) Is the system controllable?**

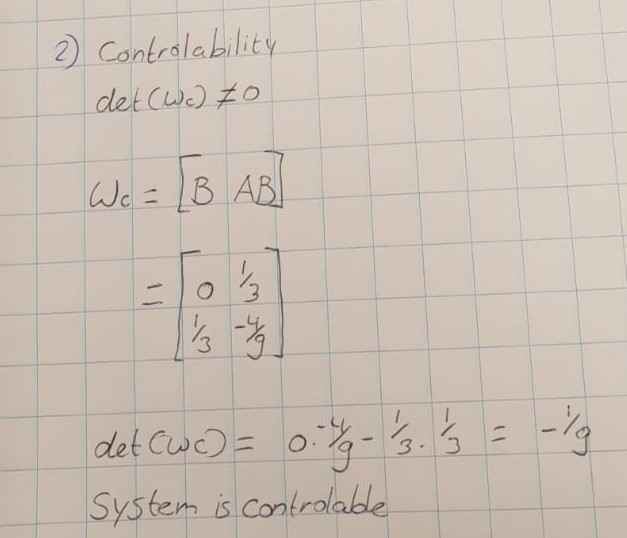


Figure 19 Answer to 2)

**3) Is the system observable?**

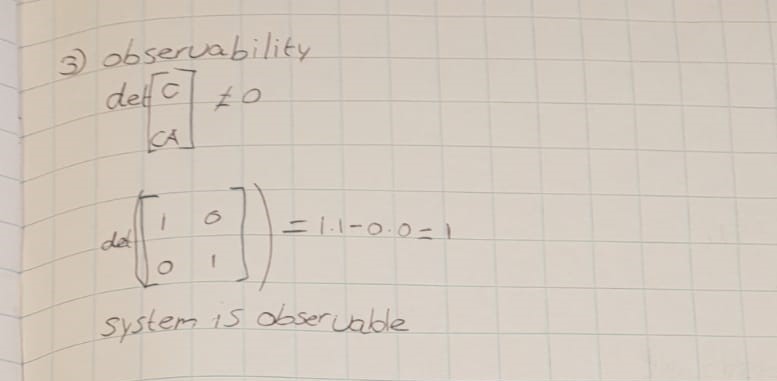


Figure 20 Answer to 3)

**4) Design a state-feedback controller u(t) = −k T \* x(t) + kr\*r(t) such that the closed-loop system has poles in s = −1 and s = −3 and tracks the constant reference input r(t) = 3 with zero steady-state error.**

The rest of the answer (figure 22) is on the next page

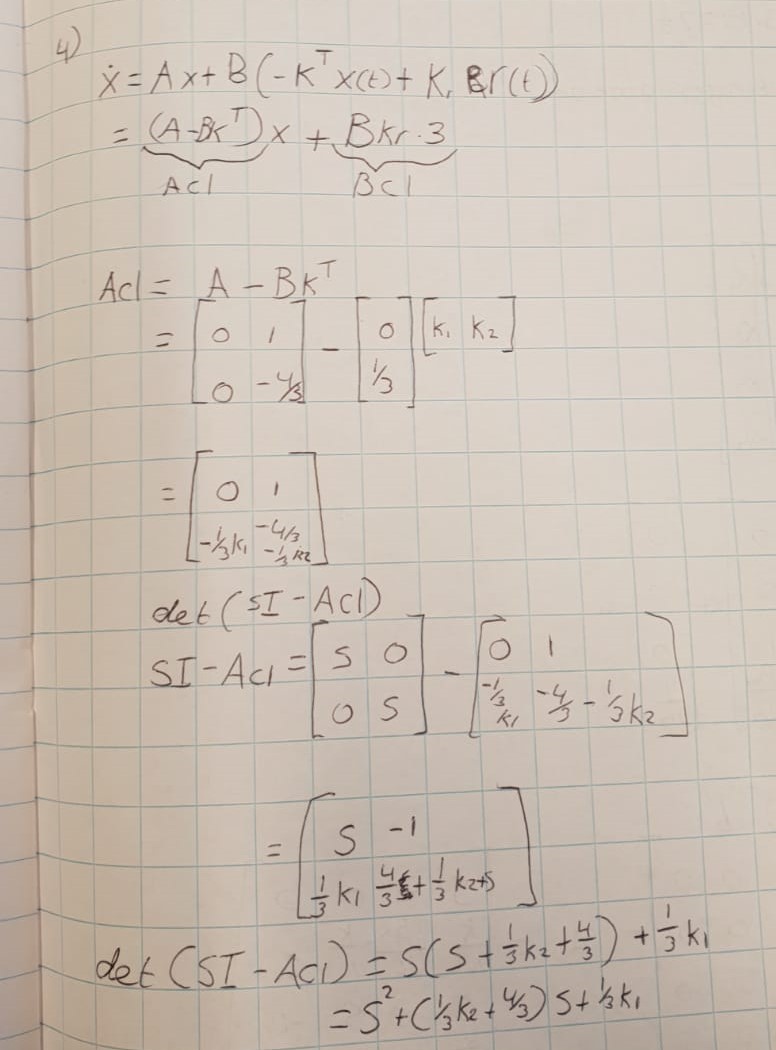


Figure 21 Answer to 4) 1/2

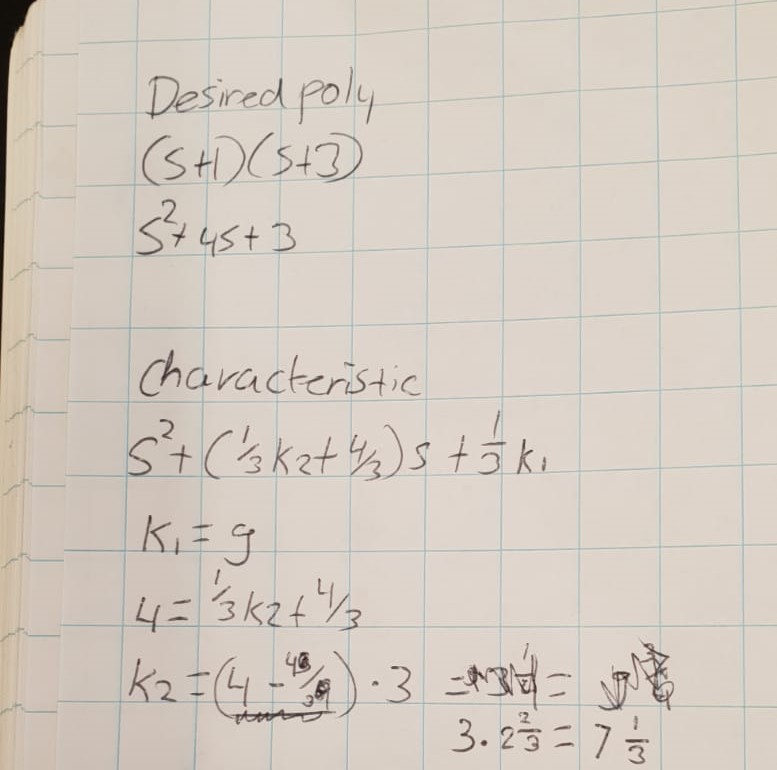


Figure 22 Continued answer to 4) 2/2