Design and Control of a Single-Axis Robotic Arm for Precise and Accurate Motion

Moamen Ali 202000437, Abdelateef Khaled Abdelateef 202001344, Seif Islam Raslan 20200079

Department of Engineering, University of Science and Technology at Zewail City, Egypt 2022

Contents

1 abstract **3**

2 Introduction **3**

3 System Modeling and Properties **4**
3.1 Mathematical Modelling 4
3.2 Definition of the system parameters 5
3.3 Simulink modelling 6
3.4 The open-loop Transfer Function 7
3.5 System Stability 7

4 Controller Design **8**
4.1 Control Design objective for the closed-loop response 8
4.1.1 Control Design Objective: 9
4.2 Design of a PID controller 10
4.3 PID controller using the basic building blocks 12
4.4 PID controller block 12

5 Results **13**
5.1 open-loop response 13
5.2 closed-loop response 14
5.3 tuned closed-loop response 17

6 conclusion **18**

7 References **18**

8 Appendix **18**

1 abstract

Many industrial and research settings use robotic arms to carry out tasks that are challenging or impossible for humans to complete. The design and control strategy for a single-axis robotic arm that can move precisely and accurately in lab and research environments is presented in this study. A PMDC motor uses a closed-loop feedback system to control the gearbox and the shaft that together make up the robotic arm. A variety of characteristics are incorporated into the design to improve the robotic arm's precision and repeatability, including sensor feedback for position and velocity control and a strong mechanical construction to reduce bending and vibration. We use experimental testing to show the effectiveness of the design, Through software tests, we establish the design's effectiveness and the robotic arm's ability to move precisely and accurately under a variety of operational circumstances. Our method offers a practical foundation for the design and control of single-axis robotic arms for a range of laboratory and research applications by MATLAB/Simulink modelling.

2 Introduction

Robotic arms are now an essential component of industrial automation, carrying out a variety of risky, repetitive, or highly precise jobs. Researchers became very dependent on these devices as they serve their purposes in research and laboratory settings in addition to industry settings, especially with having a small margin of error and pretty accurate results. They also are very economically effiecient as these devices don't require a salary. As a result, there has been a lot of advancement in the development and operation of robotic arms recently, leading to improvements in their functionality, dependability, and adaptability.

These tools have considerably increased the productivity and efficiency of numerous industries and are capable of carrying out a broad variety of tasks, from straightforward pick-and-place operations to intricate manufacturing processes. Robotic arms are utilised in research and laboratory settings where great precision and accuracy are crucial for experimentation and data collecting. It takes a multidisciplinary approach to develop and control robotic arms, combining knowledge from mechanical engineering, electrical engineering, control theory, and computer science. The design and control strategy for a single-axis robotic arm that can move precisely and accurately in lab and research environments is presented in this study. Our method offers a practical framework for the design and control of robotic arms for a variety of applications by combining mechanical and electrical design with control algorithms and simulation testing.

We used MATLAB/Simulink models, which allowed for simulation and testing of the system behaviour, to create and optimise the design and control algorithms. The simulation findings show that the robotic arm can move precisely and accurately under a variety of operating situations, making it appropriate for

a range of laboratory and research applications.

The remainder of the work is structured as follows: We describe the mathematical modelling of the system in Section 3, illustrating the potency of the design and control strategy. in Section 4, we provide a detailed description of the control approach, the control algorithm, and the technique for simulation and testing. Finally, we provide our conclusion in Section 6. Overall, our method offers a helpful foundation for the creation and management of single-axis robotic arms and has the potential to lead to novel insights and advancements in the field of laboratory and research activity.

3 System Modeling and Properties

3.1 Mathematical Modelling

We will start with the basic equation that states that in the PMDC motor there is a resistor, inductance and voltage source. Hence,

$$e_a(t) = V_b(t) + L_a \frac{di_a(t)}{dt} + \mathrm{R}_a \, i_a(t)$$

Now taking the Laplace transform to transform it to the s-domain yields,

$$E_a(s) = R_u I(s) + L_a s I(s) + V_b(s) \tag{1}$$

However, we know that the voltage in that motor is nothing but k_b multiplied by the derivative of the angular position with respect to the time. Taking the laplace transform of that equation will give us,

$$V_b(s) = K_b s \theta_m(s)$$

Now in this design there is no damping on the shaft of the arm we depend fully on the power of the motor to stop the motion and that will be illustrated in later sections. Now from the properties of the design and the concept of the gears we can extract more equations to help us obtain our transfer function. One of them is

$$\tau_m(s) = k_t I_a(s)$$

By knowing the previous 2 informations we can return to equation (1) and substitute in these equations. By doing so this yields,

$$E_a(s) = R_u I(s) + L_a s I(s) + K_b s \theta_m(s)$$

Now taking I(s) as a common factor and substituting it will bring us to,

$$E_a(s) = \frac{(R_u + L_a s)\, \tau_m}{K_t} + K_b s \theta_m(s)$$

After obtaining this important formula we must now account the gears as the gears play an important factor on the shaft's angular displacement and torque.

So, relating them with the angular displacement and the torque of the motor will give us two more equations

$$\theta_m = \frac{N_2}{N_1}\theta_L \tag{2}$$

$$\tau_m = \frac{N_1}{N_2} J_L s^2 \theta_m(s) \tag{3}$$

Now substituting equation (2) and (3) in the previous relation and factoring out yields,

$$E_a(s) = \frac{(R_a + L_a s)\left(J_L s^2 \theta_L(s)\right)}{k_t} + k_b s \frac{N_2}{N_1}\theta_L(s)$$

And this is our final equation, for the purpose of simulating and using Simulink we must obtain the transfer function of this system. Such function can be obtained by taking $\theta_L(s)$ as a common factor and dividing it by the voltage. Hence,

$$\frac{\theta_L(s)}{E_a(s)} = \frac{K_t}{J_L s^2 (R_a + L_a s) + K_b K_t s \frac{N_2}{N_1}} \tag{4}$$

3.2 Definition of the system parameters

For a PMDC motor, the following are the typical ranges of values for the motor constants and parameters:

1. Motor voltage constant, K_b: This constant is a measure of the voltage generated in the motor per unit of angular velocity. For a PMDC motor, K_b have the same value, typically in the range of 0.01 to 0.1 V/rad/s

2. Torque constant, K_t: This constant is the measure of the torque produced by the motor per unit of current flowing through it. For a PMDC motor, K_t have the same value, typically in the range of 0.01 to 0.1 Nm/A

3. Armature resistance, R_a: This is the resistance of the winding of the motor. It is typically in the range of 0.01 to 10 Ohms for a PMDC motor.

4. Inductance, L_a: This is a measure of the ability of the winding of the motor to store energy in the form of a magnetic field. It is typically in the range of 10 to 1000 microHenries for a PMDC motor

The range of gear ratios for a gearbox in a single-axis robotic arm can vary widely depending on the specific arm's design and intended use. Generally, the gear ratios in a robotic arm gearbox fall within the following range:

1. Low Range: 1:1 to 10:1

2. Medium Range: 10:1 to 100:1

3. High Range: 100:1 and above

The acceptable range of moment of inertia for a single-axis robotic arm can vary depending on the specific application and design requirements. However, as a general guideline, the moment of inertia for a single-axis robotic arm typically falls within the following range:

1. Low Range: 0.1 kg·m^2 to 10 kg·m^2
2. Medium Range: 10 kg·m^2 to 100 kg·m^2
3. High Range: 100 kg·m^2 and above

For the final decision for medium range and typical PMDC motor:

$$\begin{array}{ll} K_t = 0.05\,Nm/A, & K_b = 0.05\,V/rad/s \\ R_a = 5\,\Omega, & L_a = 500\,\mu H \\ N_2/N_1 = 50, & J_L = 1\,kg \cdot m^2 \end{array}$$

3.3 Simulink modelling

The simulink modelling of the previous transfer function is clearly shown below

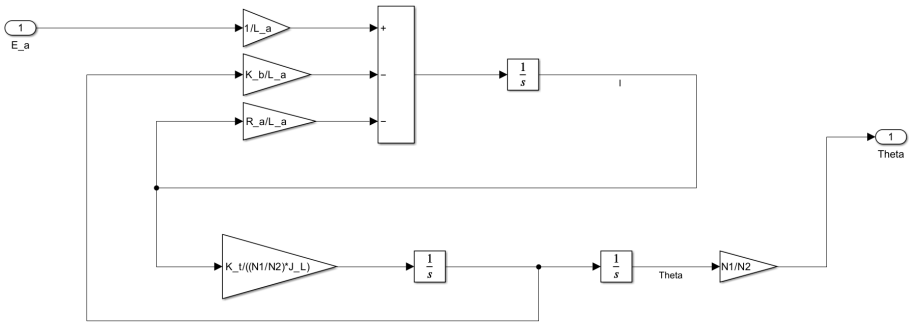

Figure 1: Simulink Model

3.4 The open-loop Transfer Function

Using Model Linearize to get the open-loop Transfer Function:

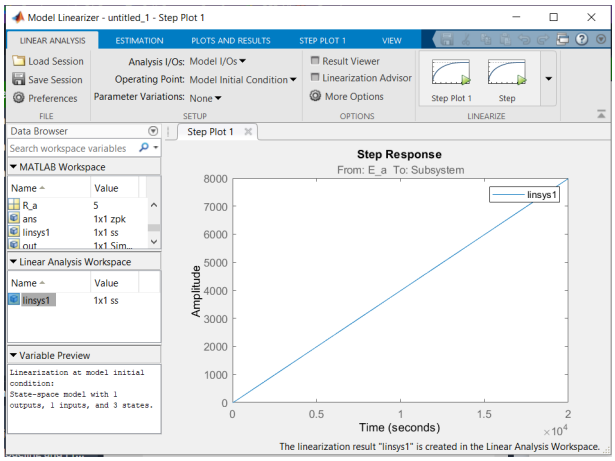

Figure 2: Model Linearizer

and the transfer function is to be:

```
>> zpk(linsys1)

ans =

  From input "E_a" to output "Subsystem":
          100
  --------------------
  s (s+0.025) (s+1e04)

Name: Linearization at model initial condition
Continuous-time zero/pole/gain model.
```

Figure 3: Transfer Function

3.5 System Stability

We got the transfer Function

$$T(s) = \frac{100K}{s^3 + 10000.025s^2 + 250s + 100K}$$

Make a Routh table:

s^3	1	250
s^2	10000.025	$100k$
s^1	$250 - (9.9975 * 10^{-3}k)$	0
s^0	0	0

The system is stable as every bounded input yields a bounded output for a gain of $0 < k < 25006.25$

4 Controller Design

4.1 Control Design objective for the closed-loop response

Transfer function for Unity Feedback and unity gain is

$$T(s) = \frac{100}{s^3 + 10000.025s^2 + 250s + 100}$$

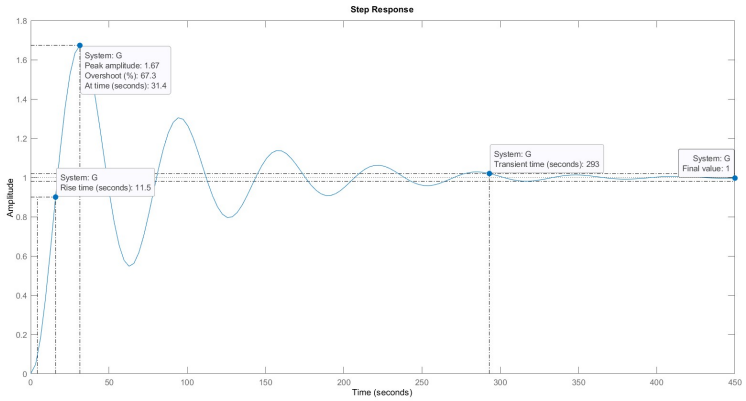

Figure 4: Step Response

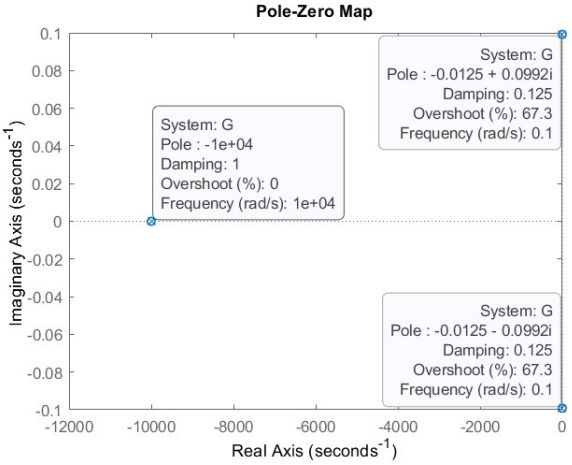

Figure 5: Pole-Zero Map

From Figure 4 and Figure 5 we can estimate the values of:

$$T_p = 31.4\ s, \quad OS\% = 67.3\%$$
$$T_r = 11.5\ s, \quad T_s = 293\ s$$
$$\zeta = 0.125, \quad w_n = 0.1\ rad/s$$

Steady-State Error of the system are shown in the below table :

Input	Steady-state error formula	Type 1	
		Static error constant	Error
Step, $u(t)$	$\frac{1}{1+K_p}$	$K_p = \infty$	0
Ramp, $tu(t)$	$\frac{1}{K_v}$	$K_v = 0.4$	2.5
Parabola, $\frac{1}{2}t^2u(t)$	$\frac{1}{K_a}$	$K_a = 0$	∞

4.1.1 Control Design Objective:

1. **Overshoot:** In control systems, a common goal is to minimize overshoot to achieve a more precise and stable response. Typical values for overshoot can range from 0% (no overshoot) to around 20% or higher, depending on the specific application requirements. We choose OS% = 5%

2. **Settling time:** The settling time depends on the desired level of accuracy and stability required for the robotic arm's movements. Typical settling times can range from a few milliseconds to a few seconds. $5\ s < T_s < 10\ s$

3. **Rising Time:** Typical values for rising time can vary significantly depending on the specific application and control system requirements. In some

applications, a fast response is desired, and thus the rising time can be in the range of milliseconds. However, in other cases where stability and smoothness are prioritized, the rising time may be longer, ranging from seconds to even minutes. $0\ s < T_r < 5\ s$

4.2 Design of a PID controller

First we sketched the Root Locus of our Transfer Function

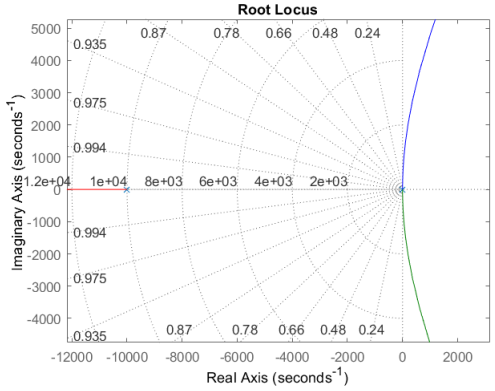

Figure 6: Root Locus Graph

we take a closer look at our damping line and how it intersect with the root locus.

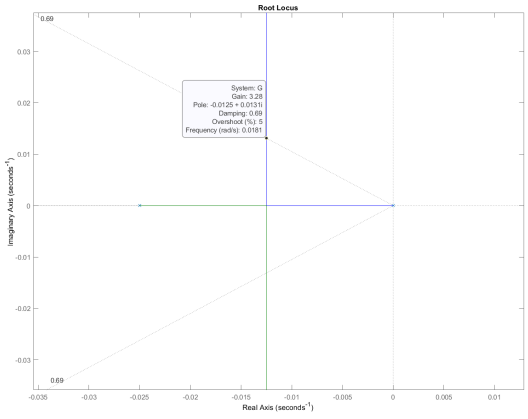

Figure 7: Root Locus Graph with damping line

we designed a matlab code to get the location of the compensator zero that makes the system has an over shoot of 5% and settling time of 10 s.

```
zeta = 0.69010673; %at 5% overshoot
wn = 0.0125/zeta; %uncompensated system frequancy
ang = 180- (acos(zeta)*(180/pi)); %damping angle
sigma = 4/10; %real part of the compensated dominant pole
wd = (sigma/zeta)*sqrt(1-zeta^2); %imaginary part of the
    compensated dominant pole
pole1_angles = 180- (atan(wd/abs(sigma-0.025))*(180/pi)); %open
    system pole angle
pole2_angles = (atan(wd/abs(10^4-sigma))*(180/pi)); %open system
    pole angle
sum_angles = -(pole2_angles +pole1_angles +ang);
zero_ang = 180 - sum_angles -360; % The angle of the compensator
    zero
comp_zero = (wd/tan(zero_ang*(pi/180))) +sigma; %The location of
    the compensator zero
```

we get a compensator zero at -0.4335; then sketched the root locus after compensating.

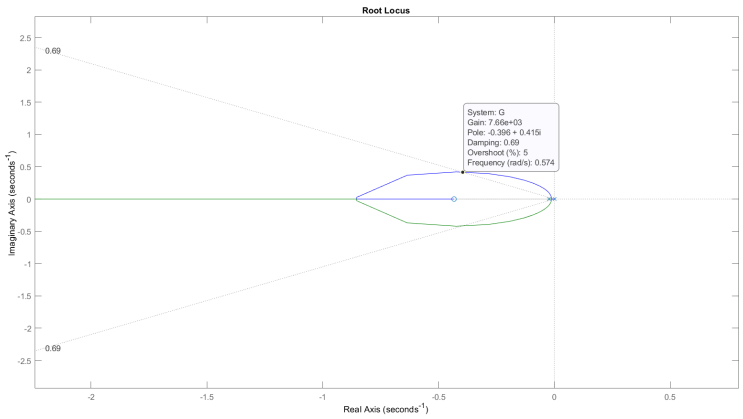

Figure 8: Root Locus Graph after compensating

Then analysing the system before and after compensating;

Parameter	Uncompensated	compensated
Plant and compensator	$\frac{K}{s(s+0.025)(s+10000)}$	$\frac{K(s+0.4335)}{s(s+0.025)(s+10000)}$
Dominant second-order poles	$-0.0125 \pm j0.0131$	$-0.396 \pm j0.415$
K	3.25	879.55
ζ	0.69	0.69
$\%OS$	5%	5%
T_s	320	10.1
T_p	239.81	7.57
K_p	∞	∞
$e(\infty)$	0	0
Third pole	-10000	-9999
Zero	None	-0.4335

4.3 PID controller using the basic building blocks

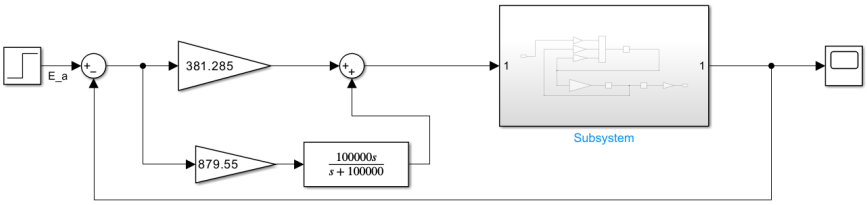

Figure 9: PID controller using the basic building blocks

4.4 PID controller block

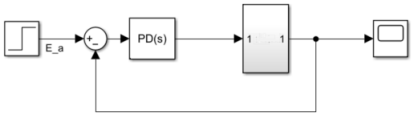

Figure 10: PID controller

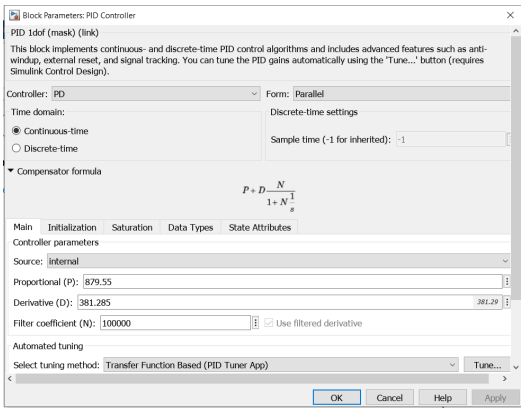

Figure 11: PID controller

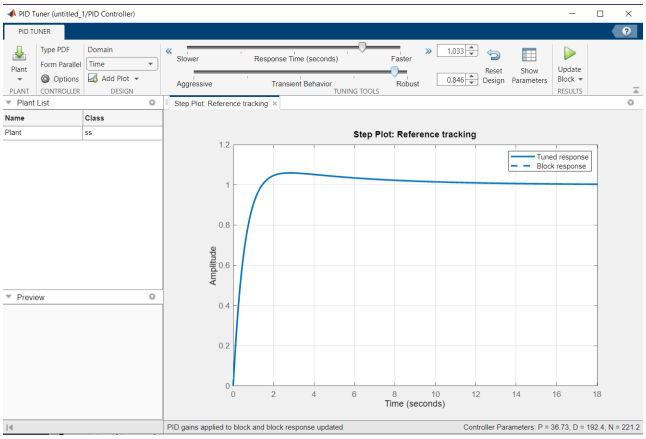

Figure 12: PID controller

5 Results

To get more vision on how our system developed, we are going to analyse and revise each process that we have made and plot their step responses to see how each process contributed to the output of the system

5.1 open-loop response

The open-loop step response is the response we get when we add a step voltage to our G(s). This gives us a better understanding of how our system naturally behaves. Referring to the equations presented in the Mathematical modelling section, we plotted the step response using MATLAB.

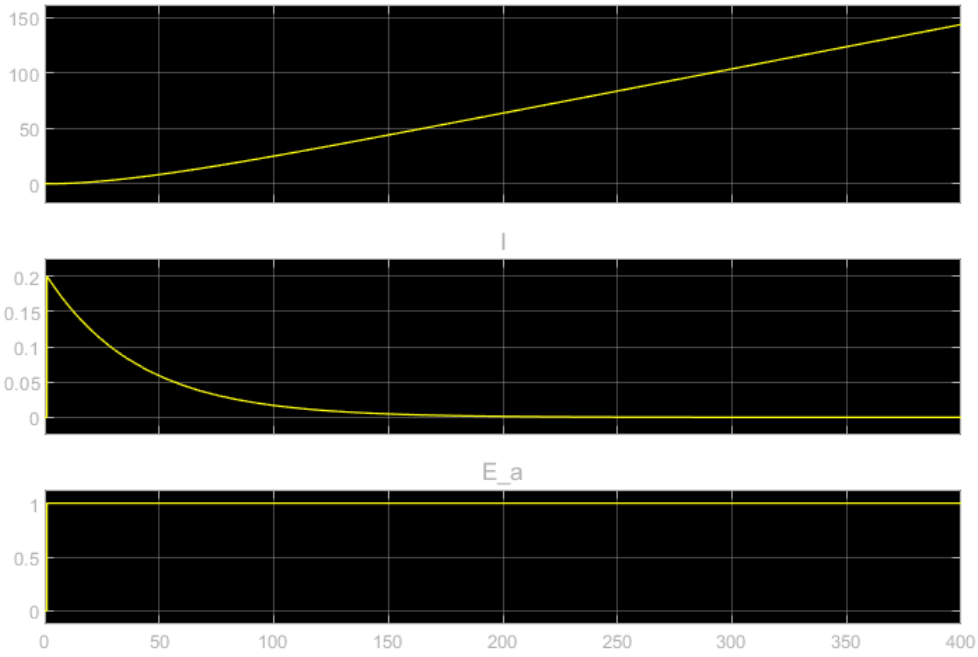

figure 12: open-loop response

The first figure represents the relation between angular displacement (θ) and time and it is clear that theta increases in somewhat an exponential form. The secound figure represents the current with respect to time and it is reasonable that the current approaches zero because at some point the arm will stop moving. The third figure represents the voltage with time and for sure it is constant (step response)

5.2 closed-loop response

The closed-loop response is the more idealistic figure of our design modelling. Bellow is the step responses at different voltages from 1v to 5v to 10v.

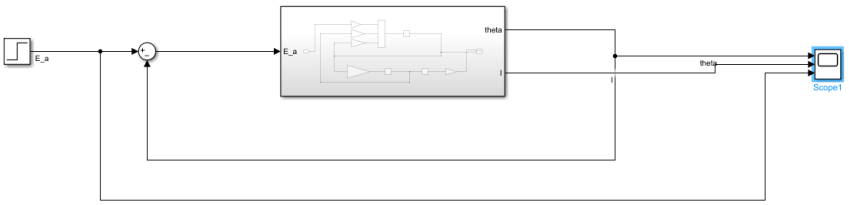

Figure 13: closed loop response

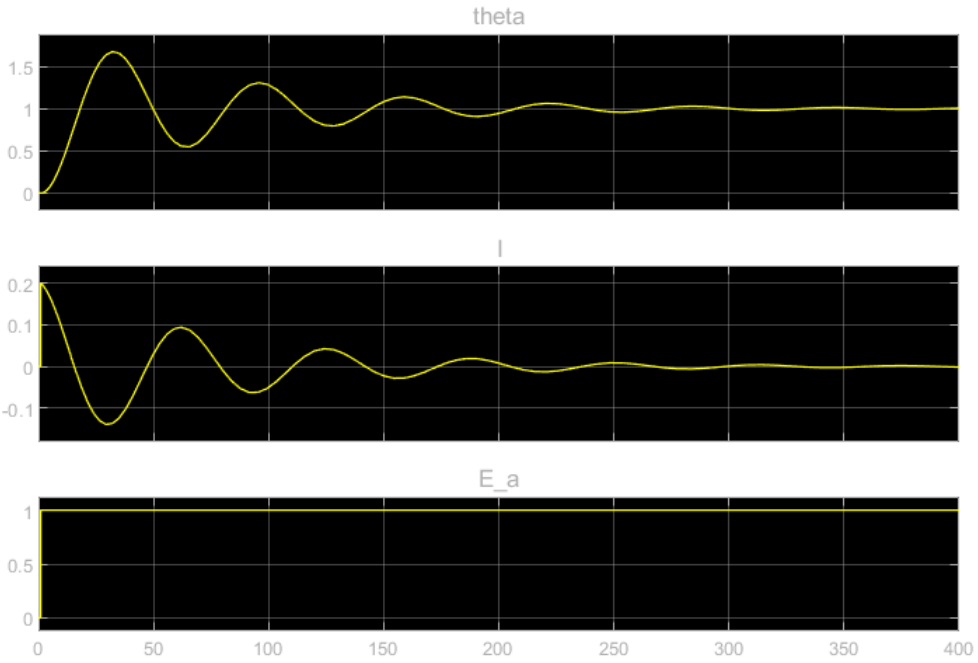

figure 14: closed loop response with voltage of 1v

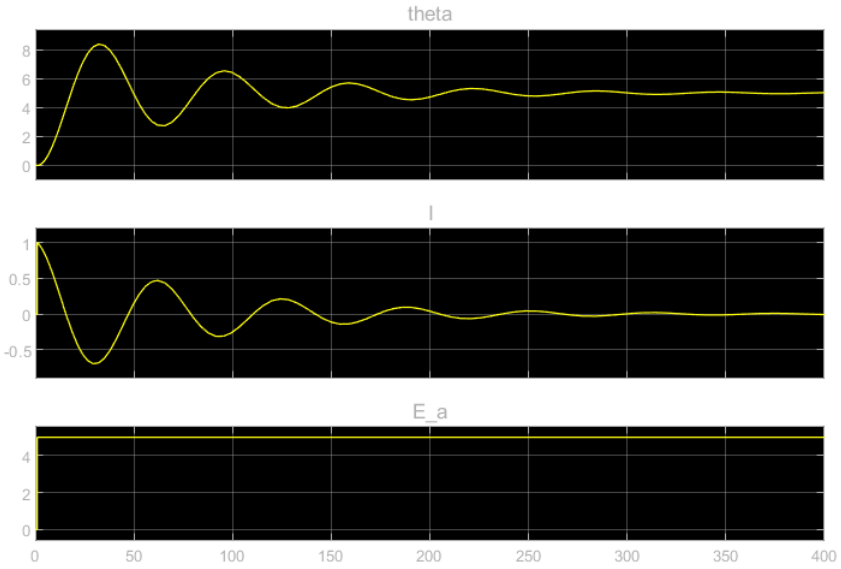

figure 15: closed loop response with voltage of 5v

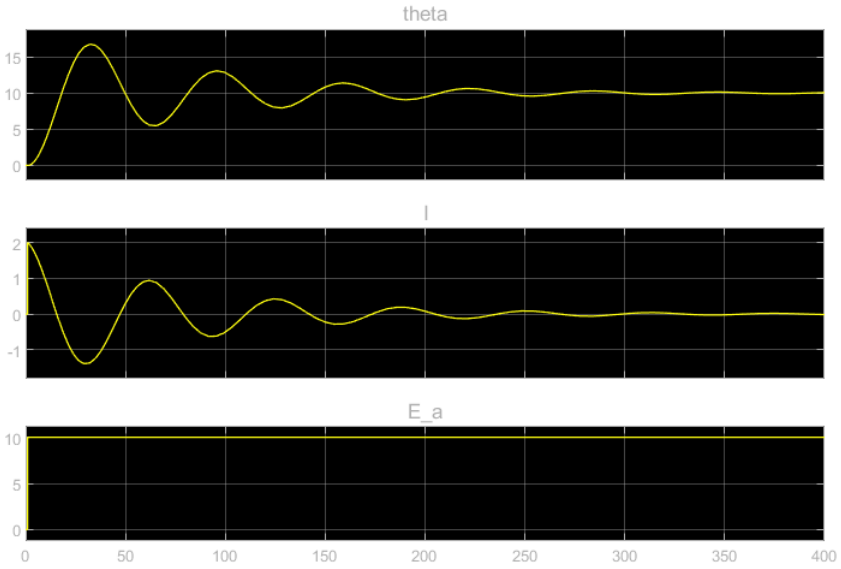

figure 16: closed loop response with voltage of 10v

We observe that by comparing the three states the graphs are quite similar the only difference is the peak theta, current and voltage. This is very expected as we are using the same transfer function with no cancelled factors.

5.3 tuned closed-loop response

After designing the PD controller and applying it to the system and plotting the results are as follows

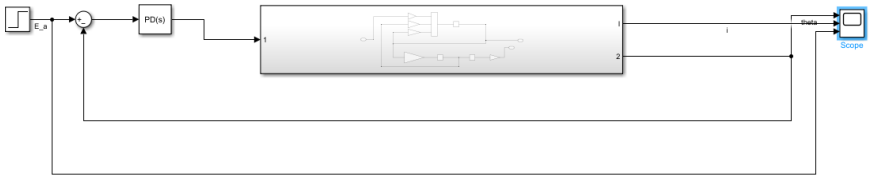

figure 17: closed-loop tuned

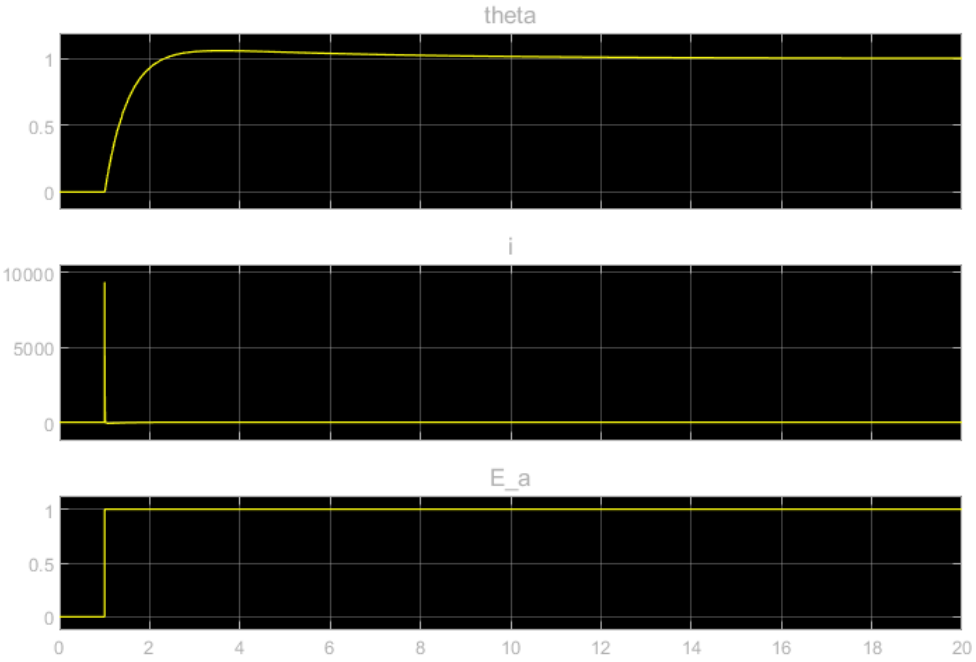

figure 18: closed-loop response

Of course the results are different from the previous plots due to and additional zero and most importantly an additional pole to the system. We see that the angular displacement responds to the input voltage that means at the desired rotation all we have to do is give the system a significant voltage input and the robotic arm will correspond to that input and stops at it. This can be more clarified by the secound graph as all that is needed is some input at the moment

of movement then it returns back to zero. The voltage also initiates at the time of the movement and reaches a constant value that is of course to keep the arm from snapping back.

6 conclusion

In conclusion, this study presents a comprehensive design and control strategy for a single-axis robotic arm that is capable of precise and accurate movement . The use of a PMDC motor and closed-loop feedback system, along with sensor feedback for position and velocity control and a strong mechanical construction, results in improved precision and repeatability. The effectiveness of the design is demonstrated through experimental testing and software simulations, showing the robotic arm's ability to perform reliably under various operational circumstances. This project offers a practical foundation for the development of single-axis robotic arms for a range of laboratory and research applications. With the use of MATLAB/Simulink modelling.

It is seen that by implementing a PD control unit we have decreased the settling time massively and the accuracy of the movement is now controlled and only by the voltage source which makes it very reliable to use. The system before the PD controller also achieved the mission theoretically but in a very long time as seen in this project. Hence, we believe through this project that PD/PID controllers are a must and serve a really importnant purpose in controlling units with in one dimensional planes or more.

7 References

NISE, N. S. (2020). Control Systems Engineering. JOHN WILEY.

8 Appendix

```
zeta = 0.69010673; %at 5% overshoot
wn = 0.0125/zeta; %uncompensated system frequancy
ang = 180- (acos(zeta)*(180/pi)); %damping angle
sigma = 4/10; %real part of the compensated dominant pole
wd = (sigma/zeta)*sqrt(1-zeta^2); %imaginary part of the
    compensated dominant pole
pole1_angles = 180- (atan(wd/abs(sigma-0.025))*(180/pi)); %open
    system pole angle
pole2_angles = (atan(wd/abs(10^4-sigma))*(180/pi)); %open system
    pole angle
sum_angles = -(pole2_angles +pole1_angles +ang);
zero_ang = 180 - sum_angles -360; % The angle of the compensator
    zero
comp_zero = (wd/tan(zero_ang*(pi/180))) +sigma; %The location of
    the compensator zero
```

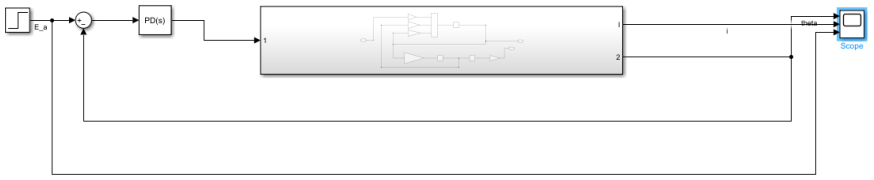

figure 19: closed-loop tuned

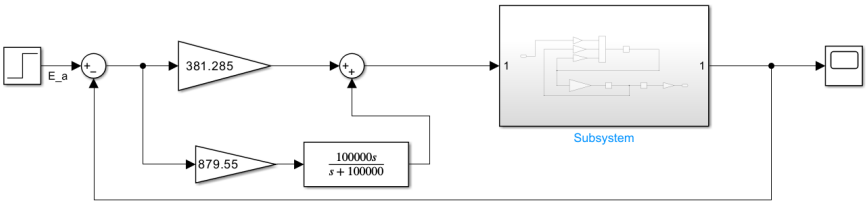

Figure 14: PID controller using the basic building blocks