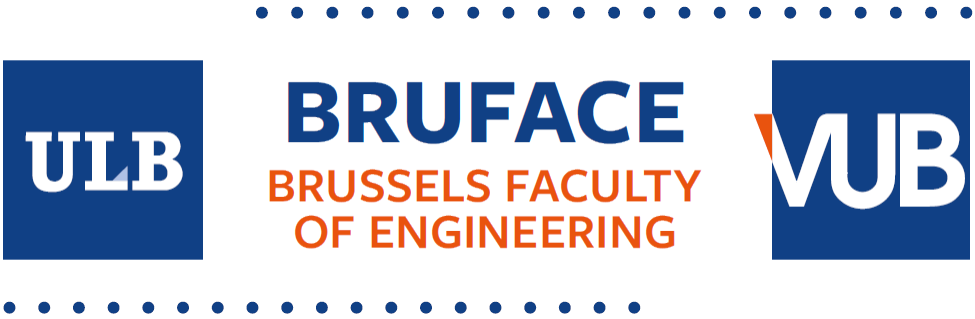
Université libre de Bruxelles



MATH-H407

Control System Design

Laboratory Report

Autor :
Livin Aurélien
Matin Pacha Joël
Moli David

Professor :
Emanuele Garone

Academic Year 2023-2024

Contents

1 Preliminaries

The laboratories aim to create a controller that fulfills specific criteria for an actual system. These criteria encompass both fundamental and more challenging advanced requirements. The system in focus here is referred to as "The ball in the tube," which involves a table tennis ball suspended in the air using an air fan. To regulate this system, the approach taken will follow the following methodology:

1. Analyse the system
    - Determine the sensor and actuator
    - Requirements of the control
2. Identify the system
    - Black box approach
    - Gray box approach
3. The control design

Multiple simulations and experimental trials were conducted to authenticate the suggested solutions, ensuring compliance with the outlined requirements. This report will encompass the activities carried out during the lab sessions, the suggested remedies, and the comprehensive implementation designed to regulate the plant.

2 Description of the system

The focus of this project centers around a system known as "the ball in the tube." Fundamentally, it involves a table tennis ball positioned within a tube, propelled upwards by a fan powered by a DC motor. The airflow generated by the fan elevates the ball within the tube, with the project's objective being the precise control of the ball's vertical position.

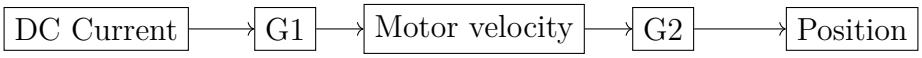


Figure 1: Diagram of the system

2.1 Sensors and actuators

The available sensors consist of a tachometer, responsible for measuring velocity (velocity sensor), and an infrared sensor designed to measure the ball's position. The system's actuator is the fan, operated by the current-driven DC motor. An interface facilitates the connection between measurements and actuators to Matlab. Existing Matlab codes were provided for system identification, and they will undergo modifications to meet the specified requirements.

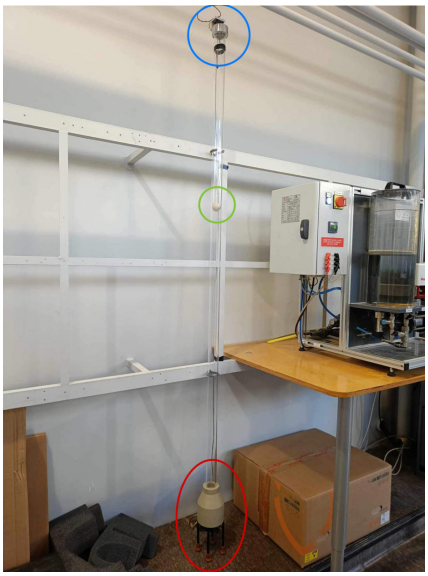


Figure 2: System with infrared sensor in blue, ball in green, fan and tachometer in red

2.2 Requirements

The first requirement entails achieving ball stabilization at a fixed position without any steady state error. Additionally, there exist more sophisticated requirements: transitioning between two distinct points with a maximum overshoot of 1 cm and switching between these points within a maximum time of 2 seconds, also with a maximum overshoot of 1 cm.

2.3 Global controller regulation

Our system will be a cascade control. We will then have a inner and outerloop with one controller each. The innerloop will control the DC motor and then the fan speed, while the outerloop will control the ball position. Indeed, there is no point in controlling the ball position if the fan speed is not correct. We must ensure that the fan speed is correct before aiming to control the ball position. Here is a scheme of our global system controller.

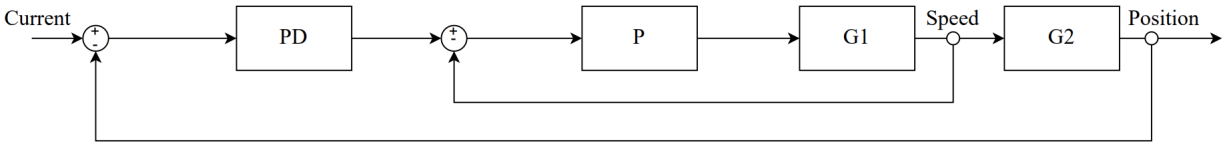


Figure 3: Global regulation

3 Inner Loop

The initial phase involves regulating the inner loop i.e the motor system. Its input and output correspond to a reference voltage and the motor's velocity, respectively, which will subsequently translate into the fan's speed.

3.1 Identification

The black box approach was utilized to identify our transfer function G_1 due to the complexity of its mathematical modeling, enabling us to save time. The representation of the black box can be depicted as follows:

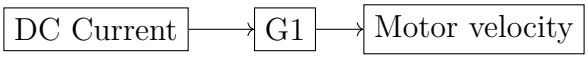


Figure 4: Inner loop system diagram

And the current-velocity static characteristic is as follows:

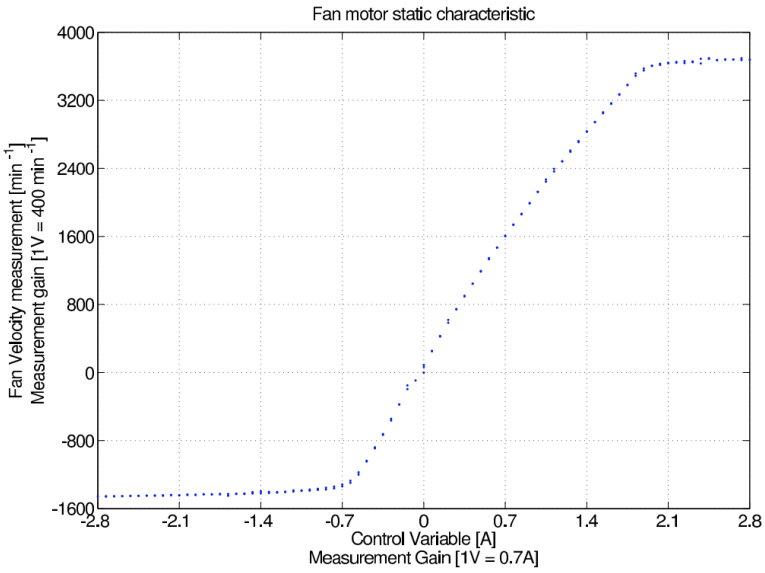


Figure 5: Ball in the tube plant: Current - velocity static characteristic

To determine the transfer function, it's essential to conduct our tests within a linear segment of the static characteristic. In addition, given that the data are sampled, it is necessary to ensure that the Nyquist-Shannon criterion is respected:

$$f_s \geq 2 * f_{sys}$$

Hence, when operating around the point $[1A; 1900min^{-1}]$, it's imperative to ensure that f_s remains equal to or greater than $2 \times 31.6Hz = 63.2Hz$.

Transfer Function

Taking these precautions into account, we may proceed with our testing to derive the transfer function of our G_1 system. As we're employing the black box approach, one method to determine this is by creating a step or impulse and observing the system's response. This observation helps us discern whether we're dealing with a second-order or first-order system. The decision to send a step or an impulse hinges on the nature of our system, whether it's integrating or not. If it's an integrating system, an impulse is necessary; otherwise, a step is more appropriate. In our case, we will send a step. The answer to this step is as follows:

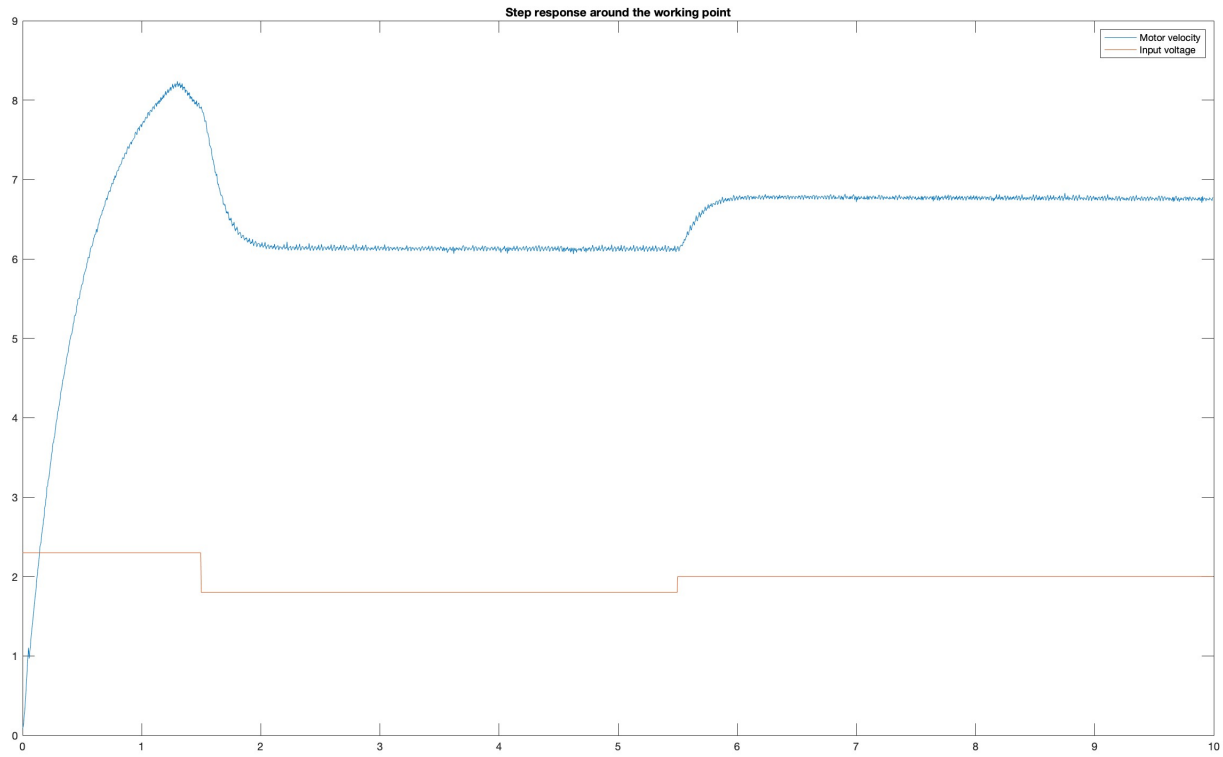


Figure 6: Step response around the working point

The conclusion that our G1 system is non-integrating stems from the observed step response. If the system were indeed an integrator, the response would have been divergent.

In our context, as previously indicated, we position ourselves within a linear operational range before initiating the step signal. The obtained response aligns with that of a first-order system, as there's no observable overshoot.

After computation, we find the following transfer function:

$$G_1 = \frac{3.37}{0.1626s + 1}$$

This transfer function as been validated by conducting this test at different ball position and by comparing it with the empirical one with a plot as follow:

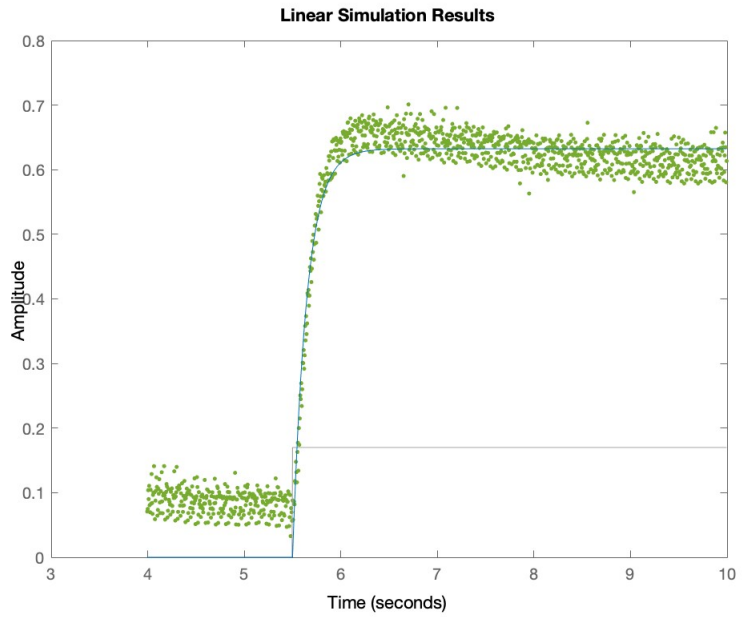


Figure 7: Impulse Response of system G_1 real and linear simulation

3.2 Regulation of the inner loop

Given the objective of positioning the ball at a specific location and considering our system G_1 as a first-order system, this entails the following implications:

- Absence of oscillation indicates the absence of a necessity to incorporate derivative action.
- Since our system is in the inner loop, the primary focus is on achieving rapid and straightforward regulation[1].
- The presence of an integrator in our outer loop system implies the absence of static error in our final regulation.

Hence, opting for a proportional regulation seems ideal due to its simplicity and rapid response.

Proportionnal controller

To determine the gain of our regulator, we have to take into account the gain margin which is computed by taking into account the fact that we use a Zero Order Holder (ZOH) in order to manipulate our discrete data.

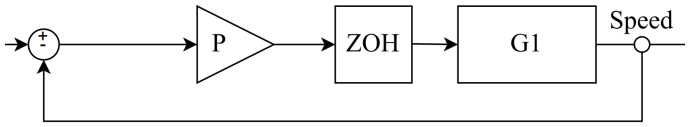


Figure 8: Inner loop diagram with ZOH

The expression of the ZOH is the following:

$$ZOH = \frac{1 - e^{T_s s}}{s}$$

[1]This is due to the direct dependency of the position on the fan's velocity.

This leads us to this Bode Diagram:

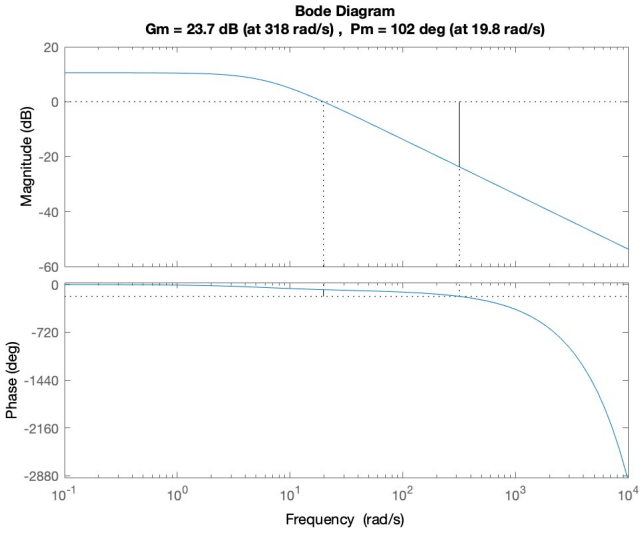


Figure 9: Bode diagram of the series connection of a ZOH and G1

This implies that our gain K has to be $< 23.7\ dB \iff K < 15.31$. Following heuristic tests, we arrived at a proportional gain value of 10 based on empirical observations and assessments.

4 Outer loop

We will now regulate the outerloop which control the position of the ball. We assume that the inner loop operates much faster than the outer loop, allowing us to disregard its effects in this section but still allowing us to have a much more precise DC-engine speed.

4.1 Identification

For some constant speed of the motor, the position of the ball will increase. So, we have an integrator in our system but we still need to determine the order of our system. Since we can use the Newton's second law, the mathematics, considering some assumptions, aren't too complex. We will then use the gray box approch.

Gray box modelling

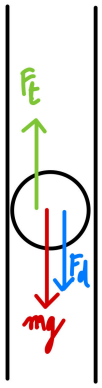


Figure 10: Physical situation

We use Newton's second law, with F_t, the thrust force and F_d, the drag force.

$$\begin{aligned} m\ddot{x} &= F_t + F_d - mg \\ &= F_t - C(v_{air} - \dot{x}) - mg \end{aligned} \tag{1}$$

Introducing small variations at the equilibrium $\tilde{x} = x - x_0$ and $\tilde{v}_{air} = v_{air} - v_{air_0}$:

$$\begin{aligned} m\ddot{\tilde{x}} &= F_t + F_d \\ &= K\tilde{v}_{air} - C\dot{\tilde{x}} \\ &= K\lambda v_{fair} - C\dot{\tilde{x}} \end{aligned} \tag{2}$$

In the Laplace domain:

$$\begin{aligned} ms^2x(s) + Csx(s) &= K_2 v_{fair}(s) \\ (ms^2 + Cs)x(s) &= K_2 v_{fair}(s) \\ G(s) = \frac{x(s)}{V_{fair}(s)} &= \frac{K_2}{s(ms + C)} \end{aligned} \tag{3}$$

We can assume that we have a second system order.

Transfer function

Now that the form of the transfer function is known, its expression can be found. Because the inner loop was previously controlled to have a step at its output which correspond to the input of the outerloop, such a step can be sent to it after positioning the ball at a point of equilibrium. The infra-red sensor is more accurate the closer to it. It is then appropriate to work in that zone to determine the transfer function. The figure 12 displays the step response of the outer loop.

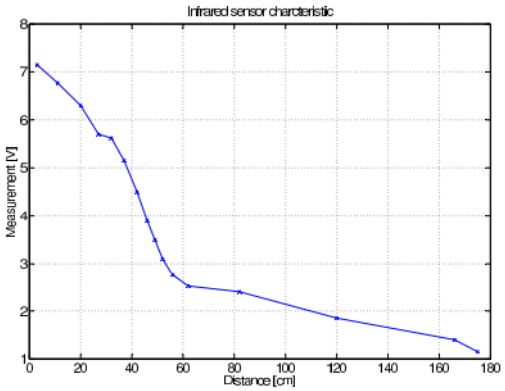


Figure 11: Infrad-red distance sensor

After computation, we find the transfer function:

$$G_2 = \frac{1}{s} \frac{6.72}{1.054s + 1} \tag{4}$$

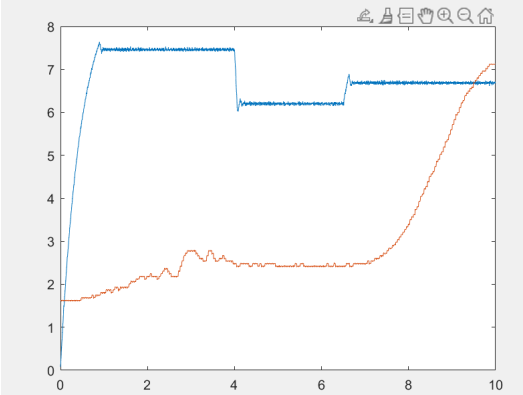


Figure 12: Step response of the system in tension [V][s]

We can confirm that we have a second order system with an integrator. As in the innerloop, this transfer function as been validated by conducting this test at different ball position.

Moreover, the following graph compares the reponse of our real system to the step generated from the inner loop and the response of the theoretical system computed with matlab's $IdentificationCode$ script to the same step. The superposition of both curves is almost complete.

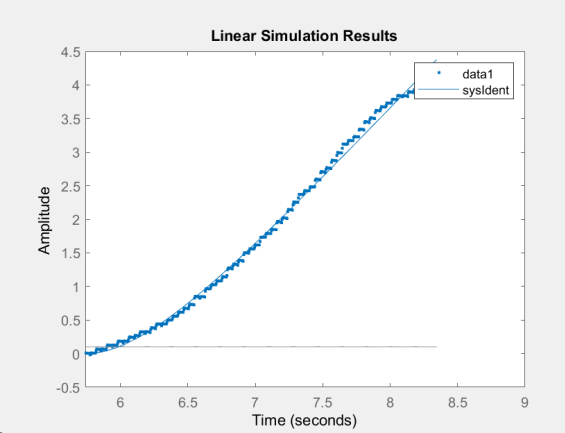


Figure 13: Transfer function

.

4.2 Regulation of the outer loop

Just like the first part of the overall system, the outer loop of the system has to be controlled to make sure the movement of the ball follows the adequate trajectory.

4.2.1 Proportionnal controller

As previously, a proportionnal gain K is used to close the second loop. Below is the diagram of the simulink graph where the left gain component is the proportionnal controller of the outer loop.

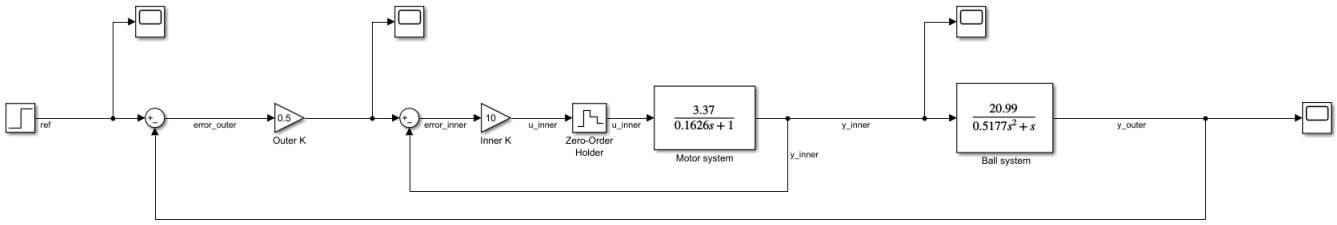


Figure 14: Regulation diagram of the outer loop

In order to find the maximum value of K, the gain margin was computed in Matlab while still taking into account the effect of the Zero Order Holder on the open loop transfer function of the system. The graph below shows the bode diagram with the gain margin obtained with the command $margin(sysIdent)$. Our gain K has to be $< 19.6\ dB \iff K < 9.55$.

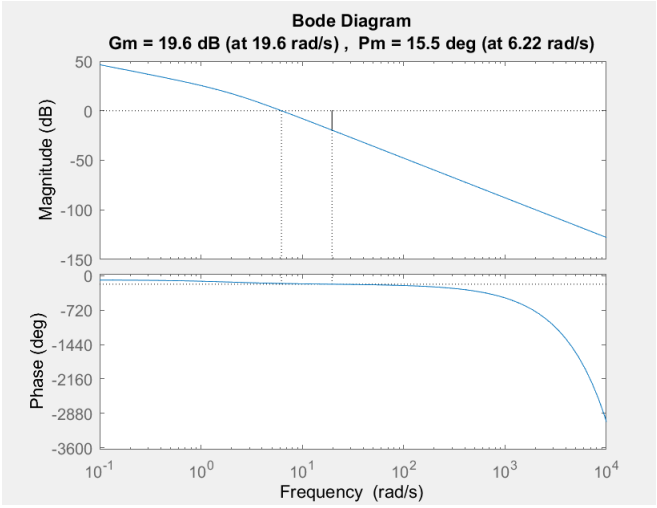


Figure 15: Bode diagramm of the ball system with ZOH

The following illustrates the output of the outer loop simulation controlled with $K = 0.5$ for a unitary step as the reference. The value of K was chosen based on a trial and error

method with bigger values showing more oscillations and smaller values showing a larger rising time. Although this method stabilizes the system, a more advanced controller must be designed to reduce the oscillations and the overshoot.

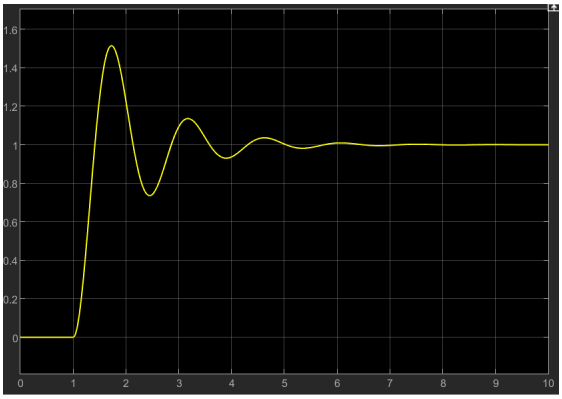


Figure 16: Proportionnal control of the outer loop

4.2.2 PD Lead-phase controller

Due to the aforementionned reasons, a lead-phase controller was chosen to improve the results. It approximates the behavior of a classical PD controller and has the advantages of reducing the rising time and the overshoot. The general transfer function for such a controller is given by:

$$D(p) = k_c \frac{s + z_c}{s + p_c}$$

with $p_c < z_c$. Both the pole and the zero of the controller have to be determined. The zero should be chosen to compensate the non-null pole of the system which is in -1.9316. On the other hand, the pole needs to be computed using the root locus of the ball system which is displayed below.

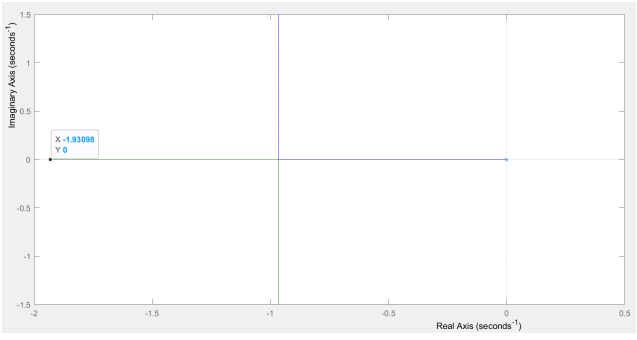


Figure 17: Root locus of the ball system

Using the MATH-H304 course, let's use the specifications of a settling time t_s of less than 2s and an overshoot M_p of less than 2% to find the damping coefficient ζ and the natural frequency ω_n.

$$M_p = e^{\frac{-\pi\zeta}{\sqrt{1-\zeta^2}}} \rightarrow \zeta > 2.95$$
$$t_s = \frac{4.6}{\zeta\omega_n} \rightarrow \omega_n > 0.78$$

The denominator of the closed loop transfer function can then be written:

$$p^2 + 2\zeta\omega_n p + \omega_n = p^2 + 4.6p + 2.95^2 \rightarrow s_{1.2} = -2.3 \pm 1.846i$$

The two poles found above should belong to the root locus of the system for the closed loop to posess the demanded specifications. Therefore, the argument rule as seen in MATH-H304 can be computed to find the pole p_c which should be on the real axis.

$$arg(G(s)|_{-2.3+1.846i}) + arg(D(s)|_{-2.3+1.846i}) = 180°$$

$$arg(s + z_c) - arg(s + p_c) = 86.4°$$
$$arg(s + p_c) = 39.8°$$
$$p_c = \frac{1.846}{tan(39.8°)} + 2.3 = 4.57$$

Finally, the gain of the controller can be computed using the condition on the module.

$$K = 6.72k_c = |\frac{s(1.054s + 1)}{1}|_{-2.3+1.846i} = 7.11$$
$$k_c = 1.058$$

Implementing this controller in simulink, the following step response was obtained. Its clear the overshoot is rendered null and that the settling time is exactly of 2s.

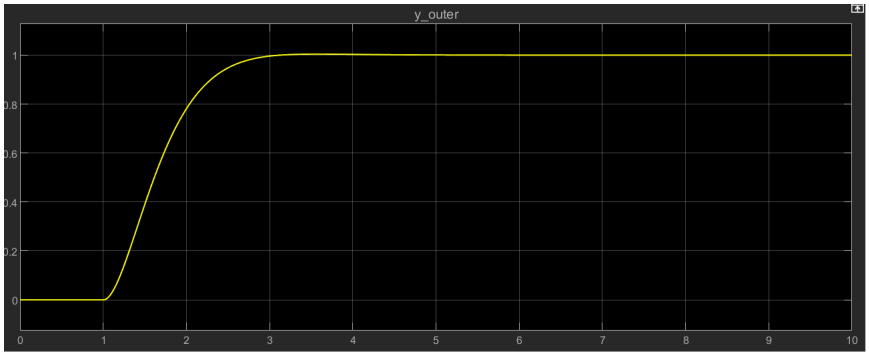


Figure 18: PD control of the outer loop

5 Requirements

5.1 Basic

Based of the previous simulations and experiments, the initial requirement of attaining a certain position without a steady error should be successfully accomplished. In the following experiment,a graphical representation is presented, depicting the behavior of the motor and ball. The orange curve represents the motor velocity, while the blue curve illustrates the position of the ball. The targeted position is set at 4 volts. This data was obtained by positioning the ball at a position close to the sensor and by subsequently sending a step. We can validate that our controller is reaching the basic requirements. The distance is computed in volts as always given the infra-red sensor caracteristic.

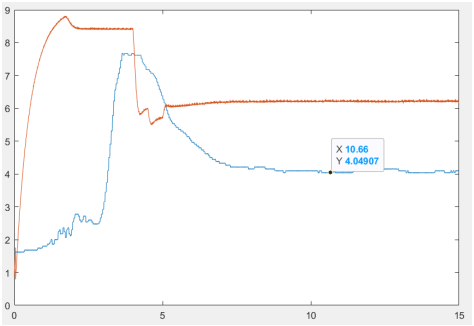


Figure 19: Controlled trajectory at 4volts [v]vs[t]

5.2 Advanced

There were 2 advanced requirements. The first one is to switch between two different points with a maximum of 1cm overshoot. The second one is to switch between this two points in maximum 2s and still 1cm of overshoot.

The following graph represents our test from a point at 3v to 6V. The orange curve represents the motor velocity, while the blue curve illustrates the position of the ball.

We can see that the switch of position occurs in less than 2 seconds. For the overshoot, we can see that the peak is at 6.33 Volts. Using the infra-red characteristic, we conclude that we have an overshoot of more of less 3,5cm. We are then only completing half of the advanced requirement.

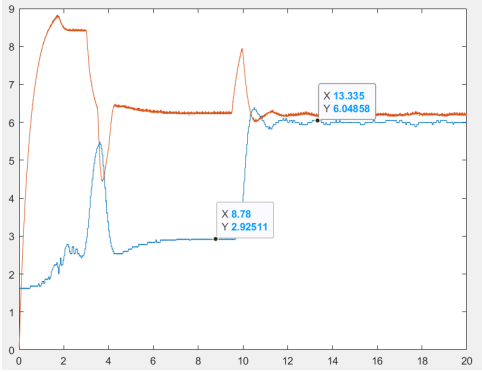


Figure 20: Controlled trajectory from 3V to 6V [V]vs[t]

6 Conclusion

In conclusion, the control system design for the "ball in the tube" project involved a comprehensive approach, addressing both the inner and outer loops to achieve precise regulation of the ball's position. The inner loop, controlling the DC motor and fan speed, was successfully modeled as a first-order system using a black box identification approach. Proportional control was implemented to achieve rapid and straightforward regulation.

For the outer loop, the system's position control was modeled as a second-order system with an integrator using a gray box approach. A proportional controller was initially employed, but to improve the system's performance, a PD lead-phase controller was introduced. This controller effectively reduced overshoot and settling time, meeting the desired specifications for the system.

Simulations and experimental trials were conducted to validate the proposed control solutions, ensuring compliance with the specified requirements. The overall control strategy, with both inner and outer loops working in tandem, demonstrated successful regulation of the ball's position in response to various scenarios.

The implemented controllers provided a balance between stability and performance, showcasing the effectiveness of the designed control system. Further refinement and optimization may be possible through advanced control strategies or additional tuning, but the current design fulfills the primary objectives.

A Matlab code

A.1 Samplecodeouter.m

```
%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Setup
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

openinout; %Open the ports of the analog computer.
Ts=1/200;%Set the sampling time.
lengthExp=10; %Set the length of the experiment (in seconds).
N0=lengthExp/Ts; %Compute the number of points to save the datas.
Data=zeros(N0,1); %Vector saving the datas. If there are several datas to save, change "
sample_debut = 800;
sample_stabilisation = 500;
sample_step = 190;
sample_end=N0 - sample_debut - sample_stabilisation-sample_step;

voltage_stabilisation = 6.2; %6.2V Pr la position on est en zone linéaire 2600tours / mi
voltage_debut = 7.5;
voltage_step = 6.7; % 8 V = 3600 tours / min
DataCommands= [ones(sample_debut,1)* voltage_debut ; ones(sample_stabilisation, 1) * vol
cond=1; %Set the condition variable to 1.
i=1; %Set the counter to 1.
tic %Begins the first strike of the clock.
time=0:Ts:(N0-1)*Ts; %Vector saving the time steps.
omega_ref = 10;
u0 = 1.83;
%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Loop
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
K =10;
e_motor = 0;
u_motor = 0;
velocity_old = 0;
% Ajouté par moi
ref = [ones(sample_debut,1)* voltage_debut ; ones(sample_stabilisation, 1) * voltage_sta

while cond==1
    [velocity,position,in3,in4,in5,in6,in7,in8]=anain; %Acquisition of the measurements.
```

```
    %ref=omega_ref; %Input of the system.
    Data(i,1)=velocity; %Save one of the measurements (in1).
    Data(i,2)=position;
    e_motor = ref(i) - velocity;
    u_motor = K*e_motor + u0;


    anaout(u_motor,0); %Command to send the input to the analog computer.
    DataCommands(i,:) = u_motor;


    t=toc; %Second strike of the clock.
    if t>i*Ts
        disp('Sampling time too small');%Test if the sampling time is too small.
    else
        while toc<=i*Ts %Does nothing until the second strike of the clock reaches the s
        end
    end
    if i==N0 %Stop condition.
        cond=0;
    end
    i=i+1;
end

%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Plots
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

closeinout %Close the ports.

figure %Open a new window for plot.
plot(time,Data(:,1),time,Data(:,2)); %Plot the experiment (input and output).
```

A.2 Identificationouterv2

```
u=Data(StartingPoint:end, 1).';
y=Data(StartingPoint:end,2).';
time2 = time(1, StartingPoint:end);
offsetu=6.246; %Operating point
```

```
offsety = 2.545;
SystemOrder=[0 2]; %Number of zeros and of poles (0 and 1), respectively.
sysIdent=IdentifySystem_outer((u-offsetu),y-offsety,SystemOrder,Ts)
%sysRequi = tf([0, 0, 20.99*1.44],[1, 1.872, 1.44] );
figure
plot(time2(),y-offsety,'.');
hold on;
lsim(sysIdent,u-offsetu,time2);
hold on;
%lsim(sysRequi, u-offsetu, time2);
```

A.3 Regulationsamplecodeouter

```
openinout; %Open the ports of the analog computer.
Ts=1/200;%Set the sampling time.
lengthExp=20; %Set the length of the experiment (in seconds).
N0=lengthExp/Ts; %Compute the number of points to save the datas.
Data_ball=zeros(N0,2); %Vector saving the datas. If there are several datas to save, cha
Data_motor =zeros(N0,1);
sample_debut = 600;
sample_stabilisation = 100;
sample_step = 1100;
sample_step2 = N0-sample_debut-sample_stabilisation;



voltage_debut = 5.5;
voltage_step = 3;
voltage_step2 = 6;
voltage_stabilisation = 1.83;
DataCommands= [ones(sample_debut+sample_stabilisation,1)*voltage_debut; ones(sample_stab
cond=1; %Set the condition variable to 1.
i=1; %Set the counter to 1.
tic %Begins the first strike of the clock.
time=0:Ts:(N0-1)*Ts; %Vector saving the time steps.

K =0.352;
u0 = 1.83;
e_motor = 0;
e_ball = 0;
u_motor = 0;
u_ball = 0;
```

```
position_old = 0;
velocity_old = 0;
e_ball_old = 0;
u_ball_old = 0;
e_motor_old = 0;
u_motor_old = 0;
%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Loop
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

while cond==1
    [velocity,position,in3,in4,in5,in6,in7,in8]=anain; %Acquisition of the measurements.
    ref=DataCommands(i); %Input of the system.
            e_ball = ref - position;
    u_ball = 0.7949*u_ball_old + 0.9720*e_ball -0.9270*e_ball_old;

    e_motor = u_ball - velocity;


    if i < sample_debut
        u_motor = 2.5;
    else
        u_motor = K*e_motor + u0;
    end
    anaout(u_motor,0); %Command to send the input to the analog computer.

    Data_motor(i,1)=velocity; %Save one of the measurements (in1).
    Data_ball(i,1)=position;
    t=toc; %Second strike of the clock.
    if t>i*Ts
        disp('Sampling time too small');%Test if the sampling time is too small.
    else
        while toc<=i*Ts %Does nothing until the second strike of the clock reaches the s
        end
    end
    if i==N0 %Stop condition.
        cond=0;
    end
    position_old = Data_ball(i);
    velocity_old = Data_motor(i);
    e_ball_old = e_ball;
```

```
    u_ball_old = u_ball;
    e_motor_old = e_motor;
    u_motor_old = u_motor;
    i=i+1;
end

%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Plots
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

closeinout %Close the ports.

figure %Open a new window for plot.
plot(time,Data_ball(:,1),time,Data_motor(:,1)); %Plot the experiment (input and output).
```

A.4 Identifysystemouter

```
    function sys = IdentifySystem_outer(input,output,S_Order,Ts)

global u y SystemOrder k;

u = input;
y = output;
SystemOrder = S_Order;
k = 0:Ts:(length(u)-1)*Ts;

theta_0 = rand(1,SystemOrder(1) + SystemOrder(2));
% Nb parameters = Nb poles + Nb zeros
theta = fminsearch('cost_outer',theta_0);
Num = [];
Den = [];
for i = 1 : SystemOrder(1)+1
    Num = [Num theta(1,i)];
end
for j = i+1 : SystemOrder(2)+SystemOrder(1)
    Den = [Den theta(1,j)];
end
Den = [Den 1 0];
sys = tf(Num,Den);
%sys = c2d(tf(Num,Den),Ts,'matched');
```

A.5 costouter

```
    function J = cost_outer(theta)

global u y SystemOrder k
Num = [];
Den = [];
for i = 1 : SystemOrder(1)+1
    Num = [Num theta(1,i)];
end
for j = i+1 : SystemOrder(2)+SystemOrder(1)
    Den = [Den theta(1,j)];
end
Den = [Den 1 0];

sys = tf(Num,Den);
hatY = lsim(sys,u,k)';

epsilon = y - hatY;

J = epsilon*epsilon';
```