Control System Design

DC motor and brake

COLOT Emmeran, FOTSA DJUIFFO Gladys, IBRAHIM Awais, MUTLU Zinar

Prof. Emanuele Garone

2024

Contents

Introduction

Context

We have been tasked to design a numeric controller for a given system which has to meet some requirements. The system is called *DC motor and brake* and it consists of two current controlled motors connected to a brake through a shaft.

Our objective is to send the best signal to the motors to ensure that:

- the shaft can reach any reasonable speed in less than half a second
- the shaft's speed is maintained as smooth as possible
- the system can reject a step disturbance as fast as possible
- the shaft's speed can follow a feasible speed reference of $4Hz$ from any feasible speed

In addition to this, we could implement a strategy to reject various brake disturbances profile and magnitude, while maintaining the speed variations as smooth as possible.

Plant description

As explained in the introduction, the plant consists of two motors and one brake acting on the same shaft. There are three sensors available: one speed sensor and two position sensors.

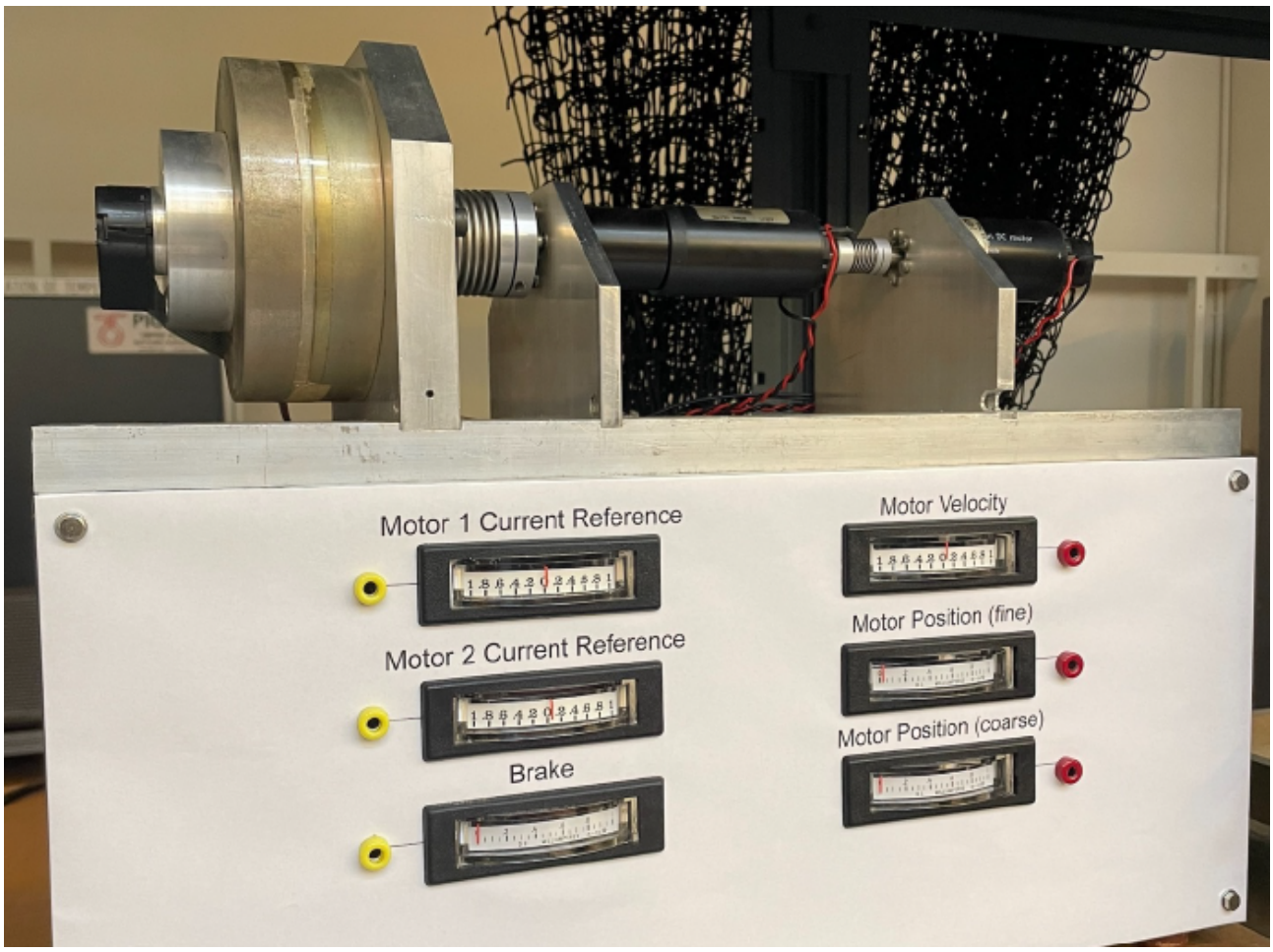


Figure 1: Picture of the plant

1 System identification

1.1 Derivation of the System Dynamics

To analyze the transfer function of a system with two DC motors and a brake using current-driven dynamics, we incorporate the relationships between motor current, torque, and angular velocity. Using Newton's Second Law for rotational dynamics, we determine the torque balance on the shaft. The system's dynamics are described by the following mechanical equations:

- **DC Motors:** The motors are current-driven, meaning the torque produced by each motor is proportional to the input current:

$$T_m = KI \tag{1.1}$$

- **Brake:** The brake applies a frictional torque that opposes the rotation of the shaft. This braking torque is often modeled as proportional to the shaft velocity:

$$T_b = c_b\omega \tag{1.2}$$

- **Torque Balance on the Shaft:** The shaft's angular velocity is proportional to the net torque on the shaft. The frictional torque is given by:

$$T_f = b\omega, \quad \text{where } T_f \text{ is the friction torque.} \tag{1.3}$$

  The total torque balance equation is:

$$T_s + T_f = T_m - T_b \tag{1.4}$$

  Substituting (1.1), (1.2), and (1.3) into (1.4), we obtain:

$$J\frac{d\omega}{dt} + b\omega + c_b\omega = KI \tag{1.5}$$

- **Electrical Dynamics:** Applying Kirchhoff's Voltage Law to the motor circuit:

$$L\frac{di}{dt} + Ri = V - K\omega \tag{1.6}$$

where $e = K\omega$ in (1.6) is the back EMF, proportional to the angular velocity of the motor shaft.

Taking the Laplace transform of the equations:

1. For the torque balance on the shaft:

$$Js\Omega(s) + (b + c_b)\Omega(s) = KI(s) \tag{1.7}$$

2. For the electrical dynamics:

$$LsI(s) + RI(s) = V(s) - K\Omega(s) \tag{1.8}$$

By eliminating $I(s)$, the transfer function relating the output (rotational speed of the shaft) $\Omega(s)$ to the input voltage $V(s)$ is derived:

$$G(s) = \frac{\Omega(s)}{V(s)} = \frac{K}{(Js + b + c_b)(Ls + R) + K^2} \tag{1.9}$$

This describes the dynamic response of the DC motor, where the rotational speed is expressed in rad/sec per volt of input.

1.2 Characteristics of the Transfer Function

Based on the mechanical equations of the DC motor, the 2nd order system can be simplified to a first order system to reduce complexity and mitigate the effects of noise and friction variations.

- In most DC motor systems, the electrical time constant (L/R) is much smaller than the mechanical time constant (J/b). Therefore, the electrical inductance (L) has a negligible effect on the overall system dynamics, and the term Ls can be ignored in the denominator of the transfer function.

$$G(s) = \frac{\Omega(s)}{V(s)} = \frac{K}{Js + b + c_b + \frac{K^2}{R}} \tag{1.10}$$

- The system can now be represented as a first order system with a time constant (τ) and a steady-state gain (A_0) .

$$Js + b + c_b + \frac{K^2}{R} = \tau s + 1, \quad \tau = \frac{J}{b + c_b + \frac{K^2}{R}} \quad A_0 = \frac{K}{b + c_b + \frac{K^2}{R}} \tag{1.11}$$

By making reasonable assumptions, the transfer function is linearized and approximated to match the expected form of a first order system transfer function, which is already known as:

$$G(s) = \frac{A_0}{\tau s + 1} \tag{1.12}$$

Because it has no pole in 0 (or equivalently it is a non-integrator system), the response of the system to a step command is enough to find both A_0 and τ.

As the plant is not perfect (the motors have some friction variations depending on the temperature, there is noise, ...), the objective is to establish a linearized model of it. In the following section, the determination of a transfer function for motor 1 will be explained in detail. This transfer function will link the speed of the shaft to the voltage applied to motor 1. We focus on speed and not position because all the requirements are based on speed, making it unnecessary to study the position.

1.3 Step response

As the internal workings of the motor were unknown, it was necessary to obtain its step response and perform system identification to understand its behavior. Since the documentation lacked details such as the motor's torque or friction, its dynamics could not be calculated. Therefore, a black box approach was used to determine the transfer function from input-output data.

The step response experiment began with analysing the static characteristic of the motor:

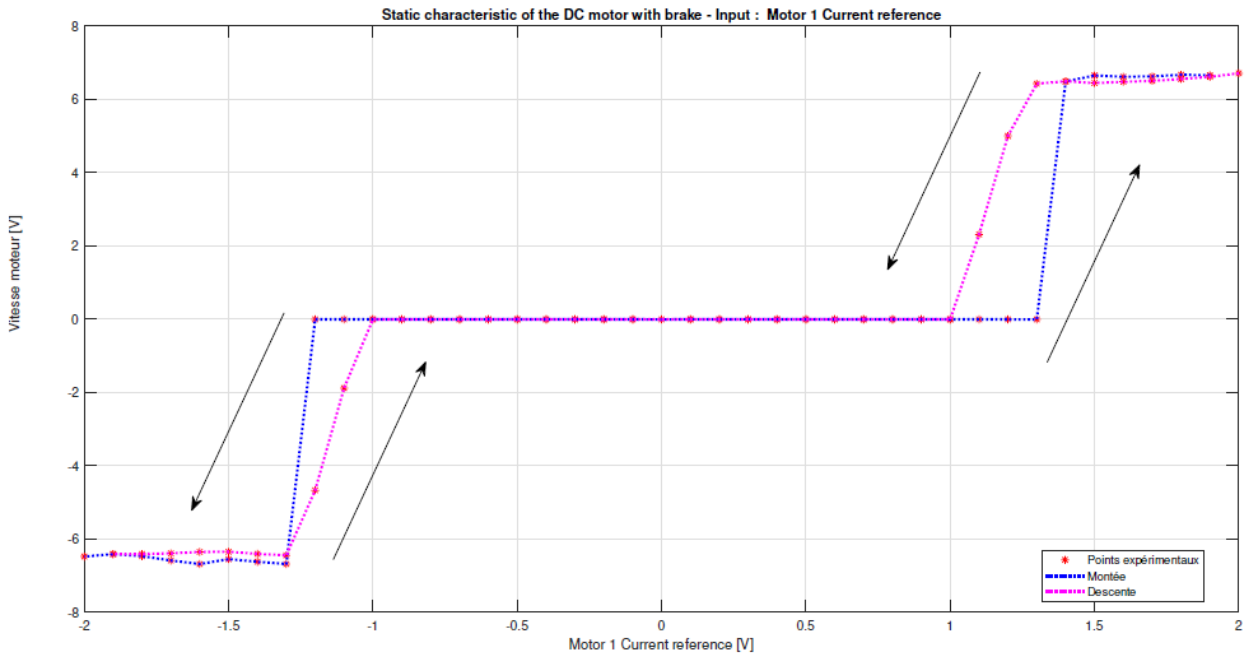


Figure 1.1: Static characteristic of the system driven by the motor 1

By analyzing figure 1.1, a first obvious deduction to be made is that the velocity cannot reach a value greater than approximately $6.5V$[1] (and $-6.5V$ is minimum value). This clarifies the use of " *feasible*" in the

[1] Any speed will be expressed in volts (V) as speed is measured by a sensor that returns a voltage

requirements as a speed of $7V$ could never been achieved, no matter what the input is.

What is really interesting to understand is that the static characteristic gives the speed of the shaft after a sufficient amount of time (in theory, after an infinite amount of time but in practice, once the transient response vanishes). By looking at it, it is clear that to have a speed that is different from $0V$ and that is outside of the saturated region, a high command should be sent to reach saturation and it must then be lowered to a point where the speed is no longer saturating (a command between $1.1V$ and $1.3V$ approximately).

With this in mind, the following step response has been recorded:

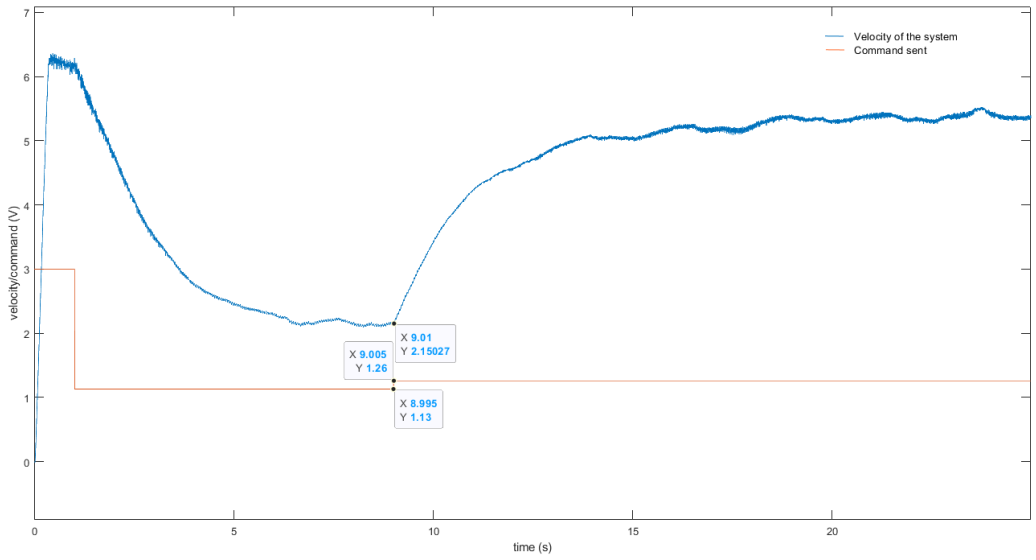


Figure 1.2: Step response of the system when the motor 1 has a step as command

Where the command is first set at a high enough voltage ($\geq 1.5V$, based on 1.1). After the velocity saturates, it lowers to a value where the velocity will stabilize ($1.13V$). This operating point OP_{1+} will correspond to the transfer function $G(s)$ that we are trying to estimate here. Then the step is introduced (with a magnitude of $0.13V$).

$$OP_{1+} = \begin{bmatrix} \text{command} = 1.13V \\ \text{velocity} = 2.15V \end{bmatrix} \tag{1.13}$$

With a coordinates change for easier visualization, the data plot 1.3 is used for the determination of A_0 and τ. It is indeed known that for a transfer function in the form of 1.12, the parameter A_0 is equal to the asymptotic value of the step response divided by the amplitude of the step. τ on the other hand is equal to the time after which the step response reaches $\frac{e-1}{e}$ (≈ 0.63) times the asymptotic step response. This gives as transfer function:

$$G_{1+}(s) = \frac{24.88}{1.915s+1} \tag{1.14}$$

The name $G_{1+}(s)$ has been chosen because the "1" indicates that it corresponds to the motor 1 and the "+" indicates that it is used for a positive speed. As shown on the static characteristic 1.1, the operating point at which the system will be operating to have a negative speed (OP_{1-}) is quite far from OP_{1+}, which means that another model will be needed to describe the behavior of the system in this zone (see discussion in section 1.4). To ensure accurate modeling and account for this difference, we use operating regions to avoid nonlinear regions.

With the model 1.14, the simulated step response can be computed and as shown in figure 1.3 it can be seen that it matches quite well with the experimental results.

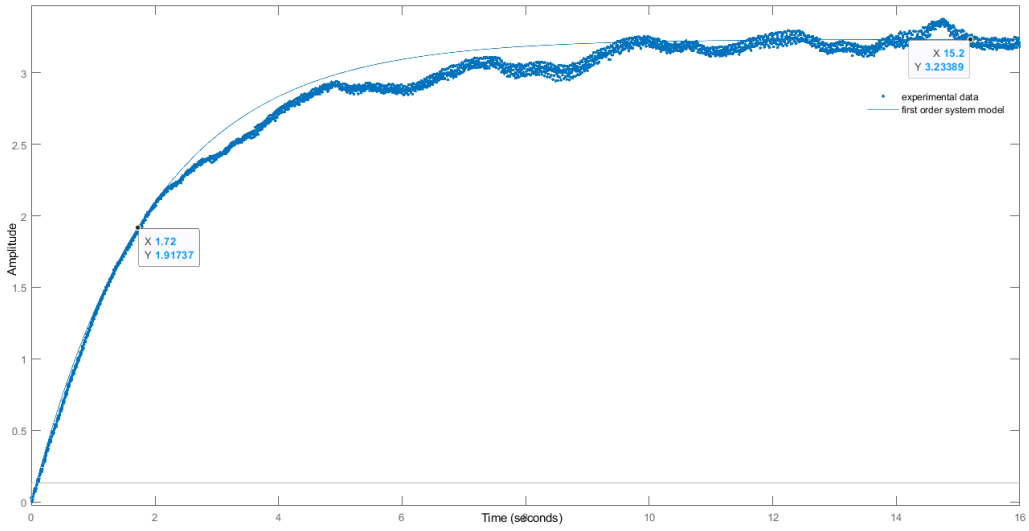


Figure 1.3: Estimation of the step response compared to the real one

By the way, the step response 1.3 can be used as a proof that the motor is a first order system: the initial value theorem states that if there are at least 2 more poles than zeros, the derivative of the step response is continuous around 0. As here $\lim_{t\to 0^+} s(t) \neq 0$[2], there is only one pole and no zero to the transfer function $G_{1+}(s)$.

1.4 Validation

The estimated transfer function established in section 1 must yet be tested. Indeed, as it is the result of the linearization of the real plant around the equilibrium point OP_{1+} we have to ensure that it has the same

[2] with $s(t)$ the step response of the system

behaviour as the plant even when the system is not in OP_{1+}.

The system has been controlled by a step similar to the one in figure 1.2 with slightly modified values. It started at $1.2V$ and the downward step has been chosen with an amplitude of $0.05V$. Figure 1.4 shows the step command with the real plant response and the response computed based on the transfer function 1.14.

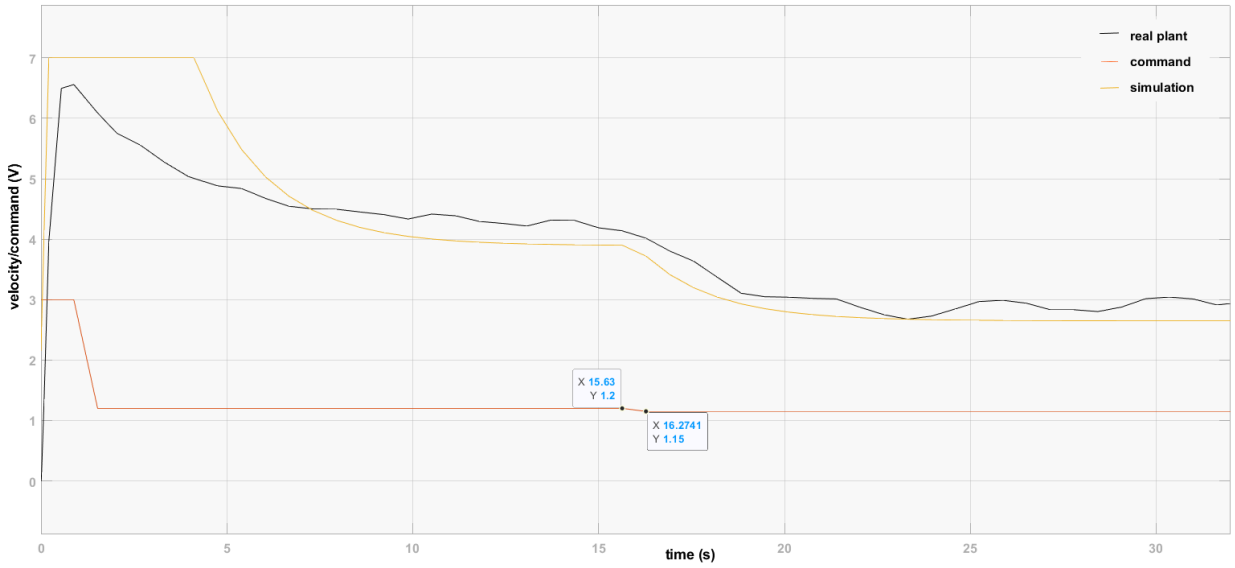


Figure 1.4: Validation of the model $G_{1+}(s)$

It appears quite clearly that the simulated response is not corresponding to the real one. Note that the first part of the response (until $\sim 10s$) is not interesting as the model does not have to take the velocity saturation into account. Based on the observation that the asymptotic value of the velocity was not corresponding between the real plant and the simulation, we tried changing A_0 to get a better matching. With $A_0 = 30$, the model was close to the reality as shown in figure 1.5, which leads to the transfer function:

$$\tilde{G}_{1+}(s) = \frac{30}{1.915s+1} \tag{1.15}$$

The conclusion that can be drawn from this whole section is that depending on the operating point, the transfer function needs to be modified *by a multiplicative factor* but the position of the pole does not change. This means that $G_{1+}(s)$ can be used if we keep in mind that the numerator can vary a little.

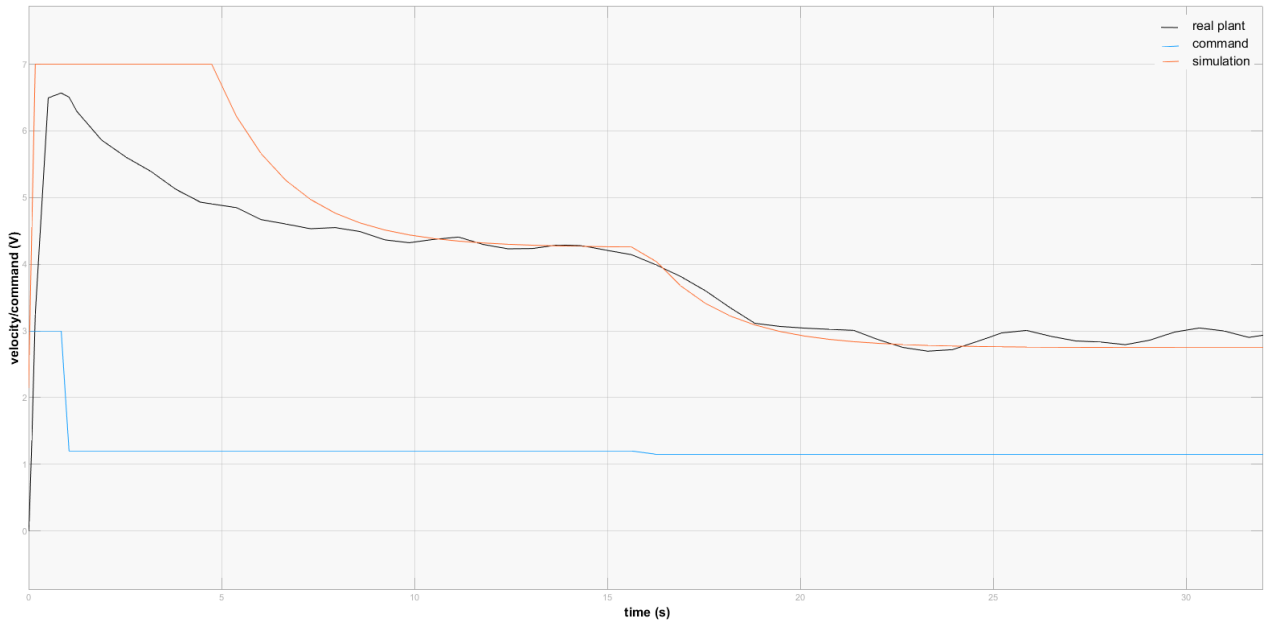


Figure 1.5: Validation of the modified model $\tilde{G}_{1+}(s)$

1.5 Other transfer functions

Based on the same experiment, we determined the other transfer functions[3]:

$$G_{1-}(s) = \frac{24.31}{1.85s+1} \qquad OP_{1-} = \begin{bmatrix} \text{command} & = & -1.1V \\ \text{velocity} & = & -1.494V \end{bmatrix} \tag{1.16}$$

$$G_{2+}(s) = \frac{19.51}{1.7s+1} \qquad OP_{2+} = \begin{bmatrix} \text{command} & = & 1.1V \\ \text{velocity} & = & 2.4V \end{bmatrix} \tag{1.17}$$

$$G_{2-}(s) = \frac{26.86}{2.09s+1} \qquad OP_{2-} = \begin{bmatrix} \text{command} & = & -1.1V \\ \text{velocity} & = & -2.51V \end{bmatrix} \tag{1.18}$$

[3]With the naming convention mentioned before

2 PID Controller

2.1 Requirements analysis

The first step of the controller design is to analyse the requirements that have been given in the introduction:

- the shaft can reach any reasonable speed in less than half a second
- the shaft's speed in maintained as smooth as possible
- the system can reject a step disturbance as fast as possible
- the shaft's speed can follow a feasible speed reference of $4Hz$ from any feasible speed

There is one reference tracking objective, one disturbance rejection objective and two dynamical response properties. The design of the controller will follow the following path:

1. Rejecting a step disturbance (with each motor separately)
2. Following a $4Hz$ reference (with each motor separately)
3. Merge the two controllers into a single digital one
4. Reach any speed in $< 0.5s$

The method used here is to design the controller in the Laplace domain in the first time. Once the continuous time controller has the desired behaviour, it is discretized using the Tustin method (without forgetting to take the ZOH into account).

2.2 Disturbance rejection and pole cancellation

From theory, it is known that a controller is able to asymptotically reject a disturbance if it has at the denominator of its transfer function the denominator of the disturbance.

As a step disturbance is of the form:

$$D(s) = \frac{Ae^{-\tau s}}{s} \tag{2.1}$$

The controller will need a pole in 0. This can be easily done by using a PI controller, which looks like:

$$C_{PI}(s) = k_p\left(1 + \frac{1}{T_i s}\right) \tag{2.2}$$

$$= k_p\left(\frac{T_i s + 1}{T_i s}\right) \tag{2.3}$$

The k_p will be chosen once the dynamic response characteristics will have to be met. However, T_i can already be chosen based on a simple criteria. As the open-loop transfer function equals the product $C_{PI}(s) \times G_{1+}(s)$, a solid choice is to use T_i to cancel the pole in the system. This way, the position of the poles of the closed-loop will be entirely based on the controller.
Of course this will be true in the case of a perfect model but as proved in the section 1.4, the parameter τ of the transfer function (which fixes the pole of it) does not seems to move a lot depending on the operating point. We can conclude from this that the best choice is:

$$T_i = \tau \tag{2.4}$$

for each controller, which leads to:

$$C_{PI}(s) = k_p\left(\frac{\tau s + 1}{\tau s}\right) \tag{2.5}$$

2.3 Frequential analysis

As the next objective is to perfectly track a $4Hz$ reference, the bandwidth of the closed loop has to be studied. If it is greater than $4 \times 2\pi \approx 25$, then there is no need to add a part to the controller's structure.
The bandwidth is determined using the Bode curves of the closed loop. This means that the structure of the regulator has to be defined. In continuous time, it looks like:

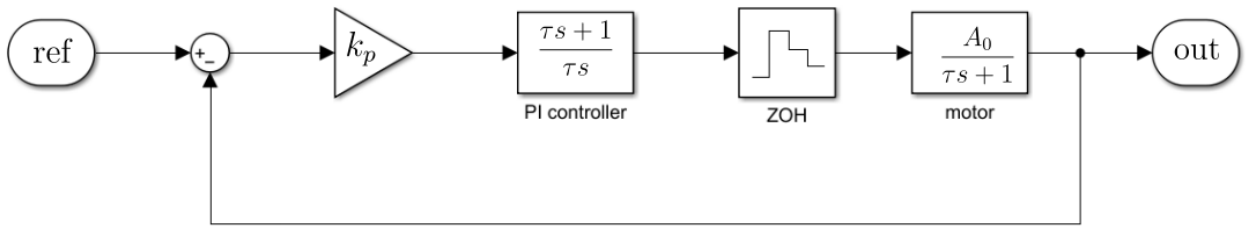


Figure 2.1: Structure of the closed loop system with a PI regulator

To determine the gain (= k_p) that can be put in the system, the `margin` method of matlab is used (fig 2.2) on the open-loop transfer function with k_p set to 1. A gain margin of $6dB$ has to be maintained [1] to ensure the stability of the system once the loop is closed.

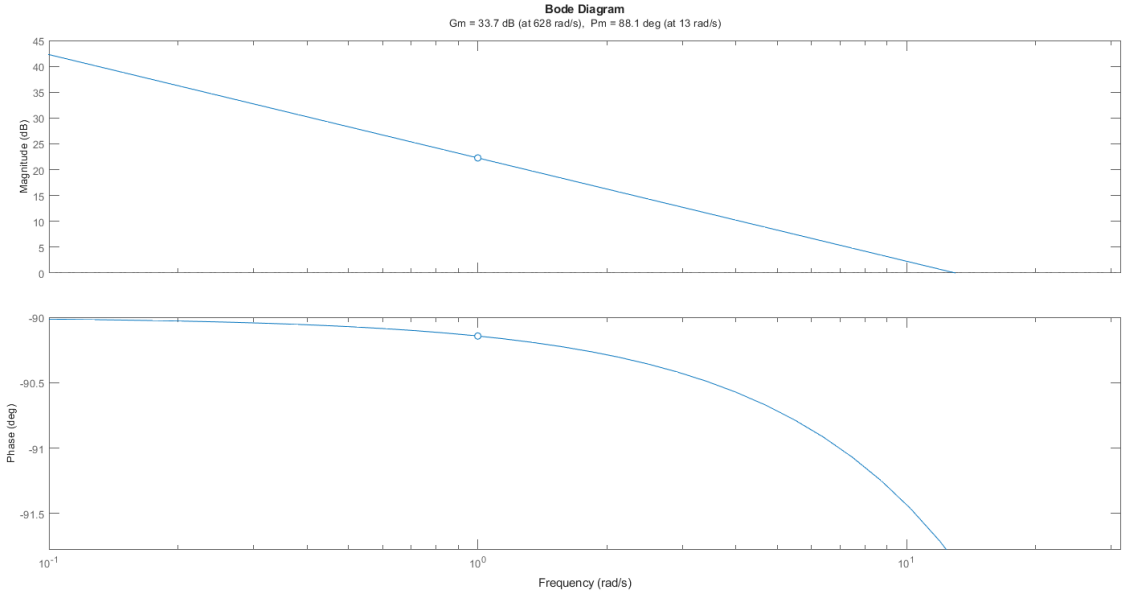


Figure 2.2: Bode diagram of the open loop system with the stability margins - motor 1

To keep a gain margin of at least $6dB$, we can multiply k_p (which was set to 1) by $27.7dB = 10^{27.7/20} = 24.26$. This means that if the loop is closed with $k_p = 24$, the closed loop system will be stable and it's bandwidth can be determined. This is done by taking the frequency at which the gain of the closed loop system falls under $-3dB$.

Figure 2.3 gives a bandwidth of around $709\text{rad}/s$. This proves that a simple PI controller is sufficient to perfectly track a signal at $4Hz$ with a single motor.

Finally, the simulink model of the controller is shown on figure 2.4.

The same method has been used to design a PI controller for the second motor too.

2.4 PI controller result

By analysing the response of each controller to a step reference (figures 2.5 and 2.6), we can see that both have a settling time smaller than $0.5s$ (respectively $0.135s$ and $0.23s$). An observation that will be interesting for the chapter 3 is that the second motor is weaker than the first one. It indeed takes $115ms$ for it to reach the reference instead of the $45ms$ it takes for motor 1.

The regulated system is also able to follow a sinusoïdal reference at $4Hz$ as expected with the k_p chosen in section 2.3 as proved on figure 2.7 for the first motor:

[1] by convention

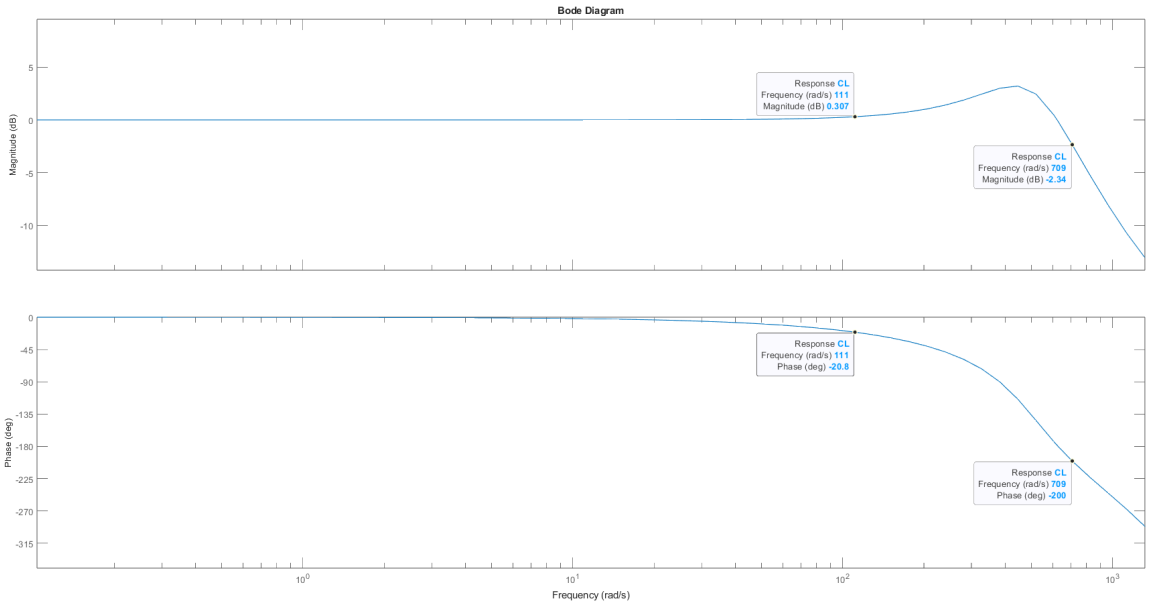


Figure 2.3: Bode diagram of the closed loop system for $k_p = 24$ - motor 1

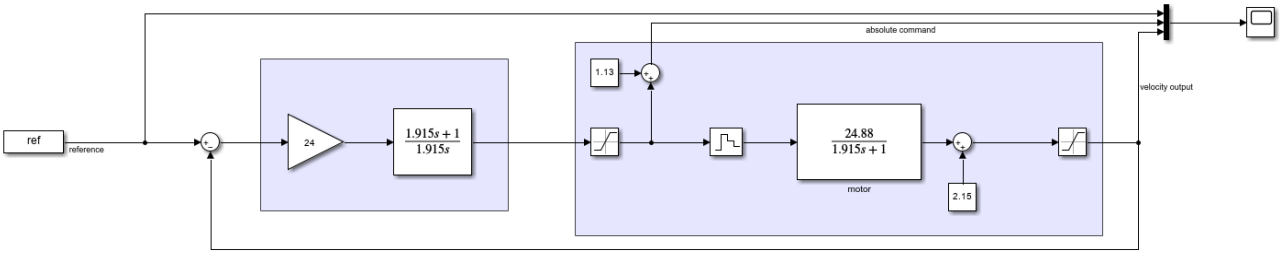


Figure 2.4: Simulink model of the PI controller - motor 1

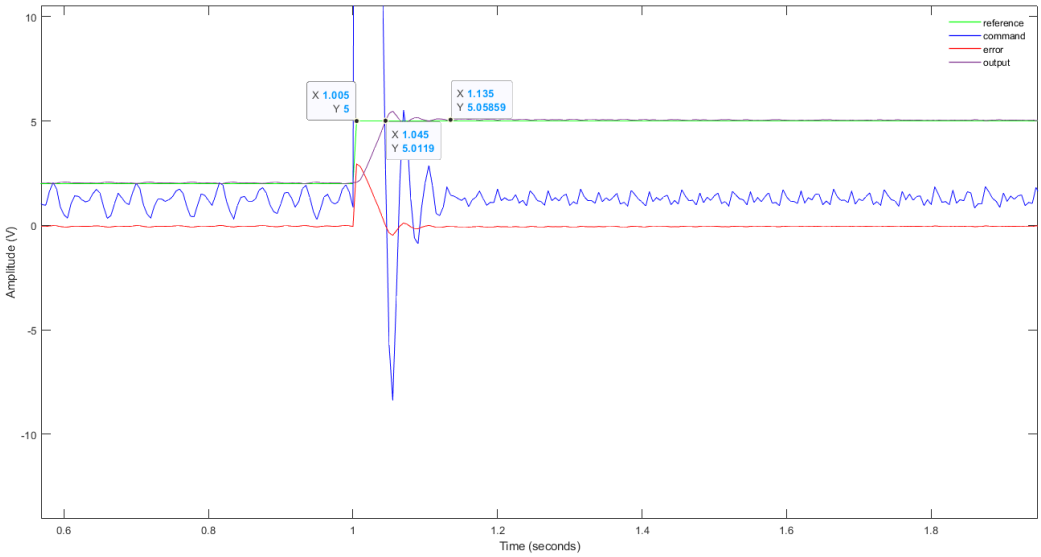


Figure 2.5: closed loop output with a PI controller for a step reference $[0;5]V$ - motor 1

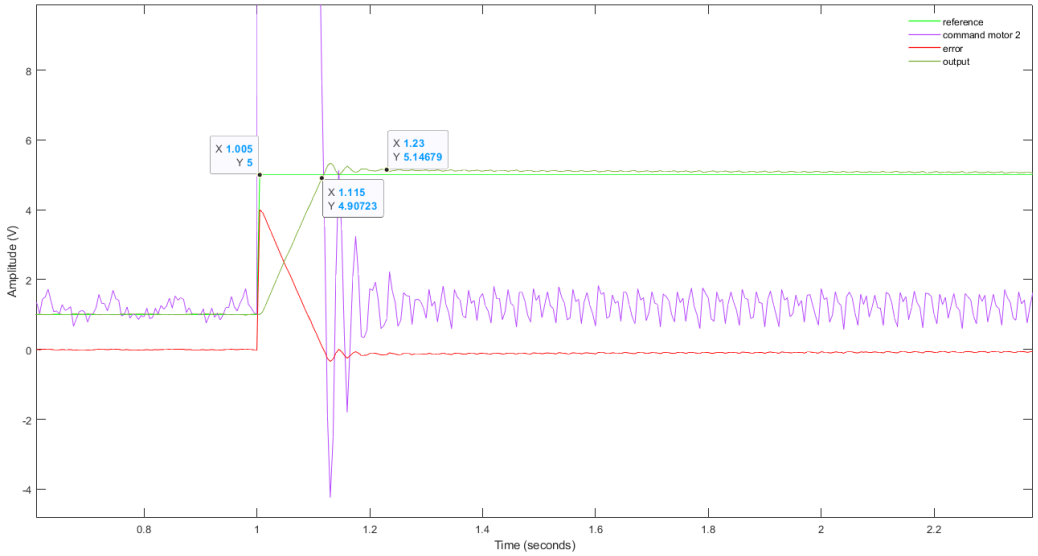


Figure 2.6: closed loop output with a PI controller for a step reference $[0;5]V$ - motor 2

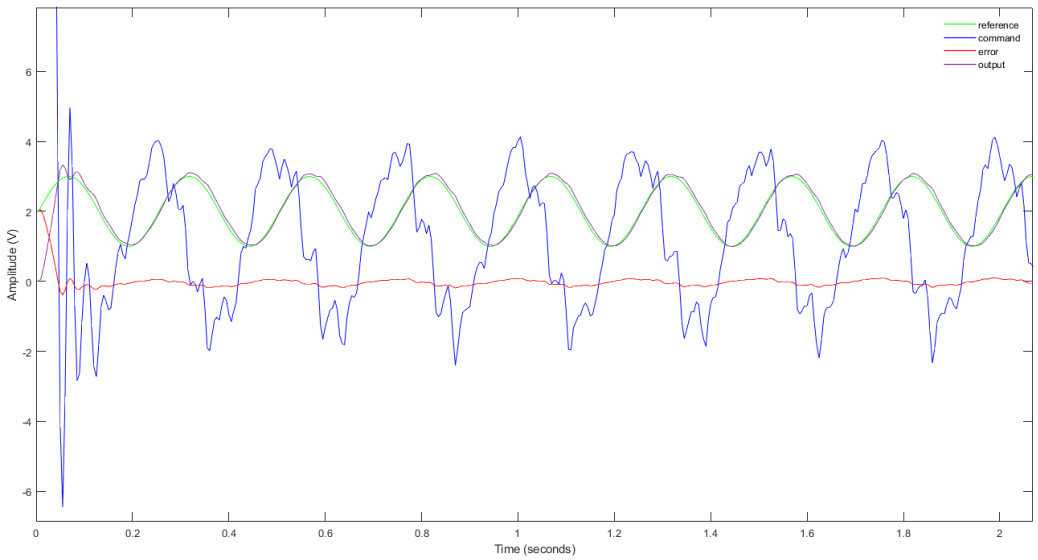


Figure 2.7: closed loop output with a PI controller for a sine reference $(4Hz)$ - motor 1

2.5 Combination of the two motors

The idea now is to use both controller designed from now to once again reduce the settling time. It is worth noting that putting both motors in parallel destabilized the system so a gain block ($k_p = 0.45$) was added to make the open loop phase margin greater than $6dB$ (methodology of section 2.3), leading to the control scheme:

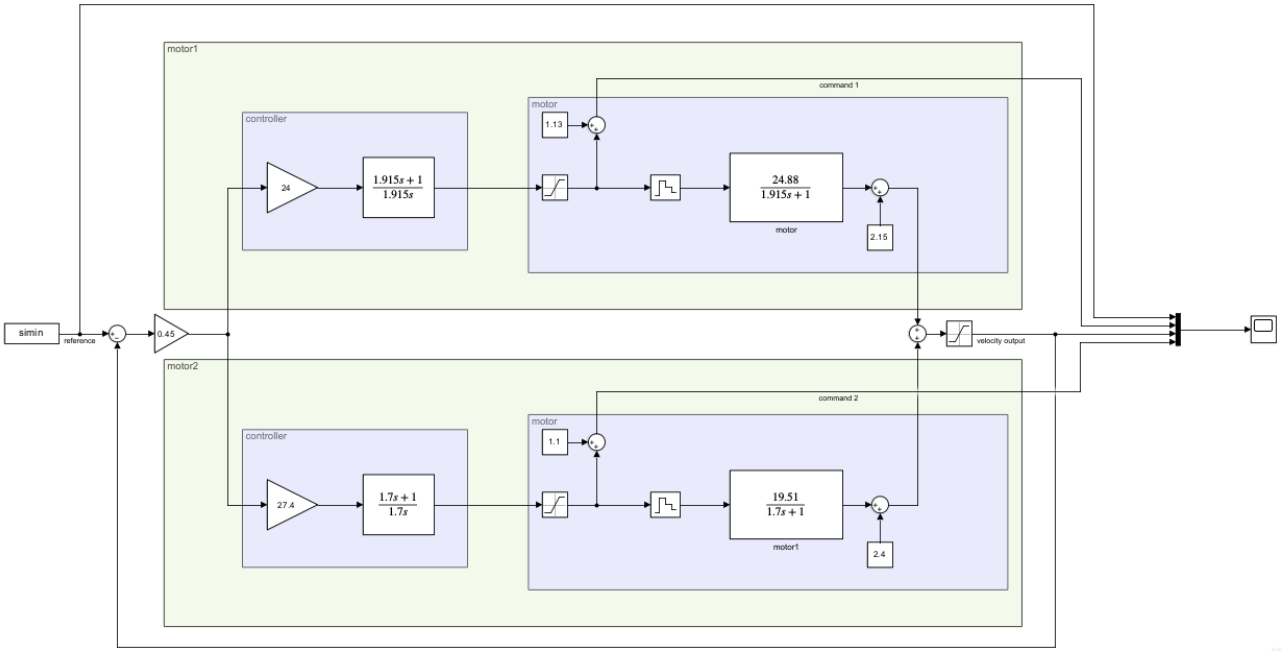


Figure 2.8: structure of the PI controller with both motors

This controller has then been tested to check the four requirements:

- **Settling time:** with the biggest reference step that could be imposed to the system, the settling time is $165ms$ as shown on fig 2.9. This time is greater than the $135ms$ measured with only the 1^{st} motor as the step was twice smaller there.
- **Reference tracking:** the output velocity perfectly follows the reference when it is at $4Hz$ (see figure 2.10). The bandwidth of the controller is $689Hz$ (the Bode plot of this controller has been removed to save space).
- **Disturbance rejection:** the analysys of figure 2.11 shows that the disturbation is fully rejected after $3.3s$. Even though this time is quite consequent, the velocity doesn't deviate more than 6.5% compared to the reference.
- **Speed smoothness:** the tree of the following figures have an output that is smooth.

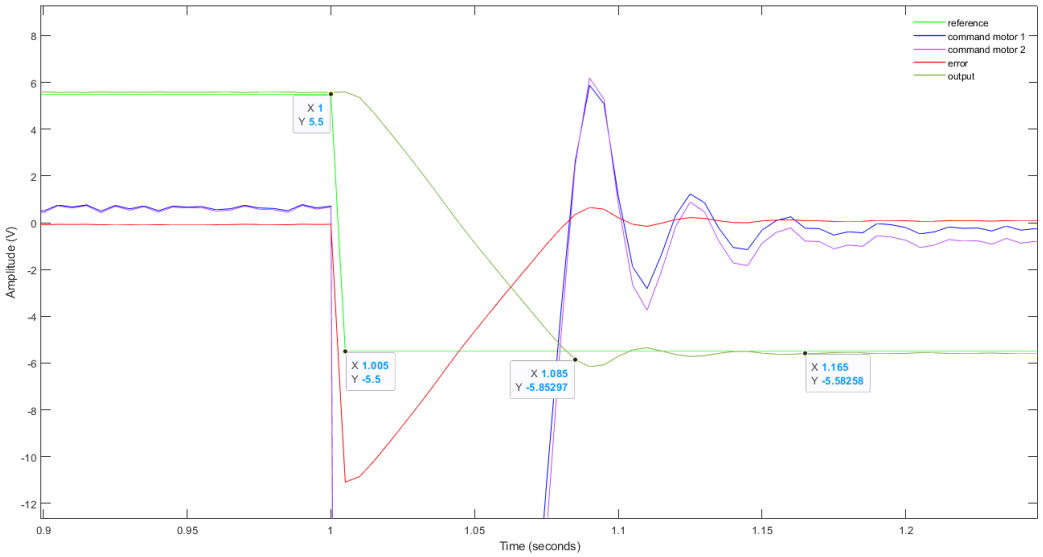


Figure 2.9: response of the PI controller with both motors for a step reference $[5.5; -5.5]V$

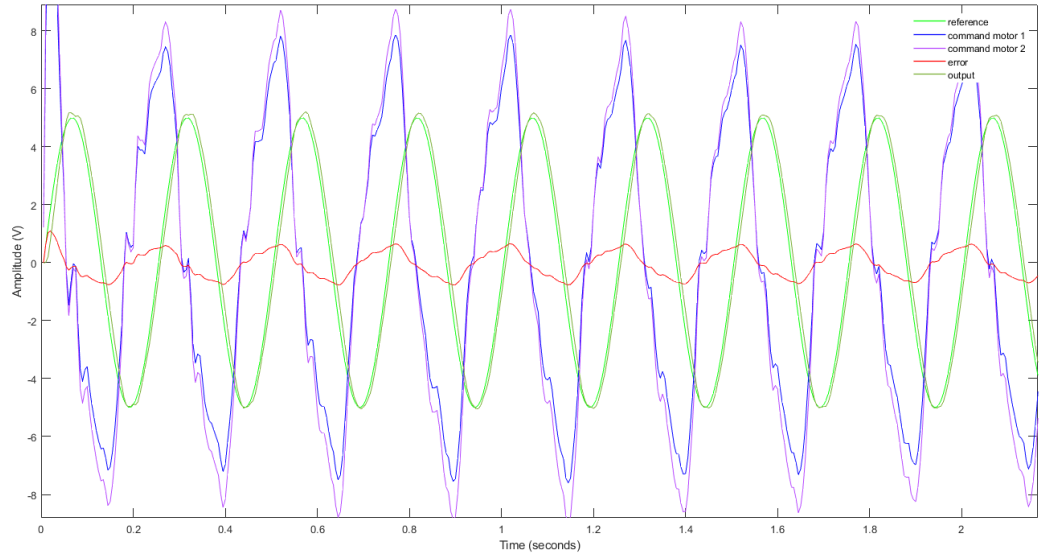


Figure 2.10: response of the PI controller with both motors for a sine reference at $4Hz$

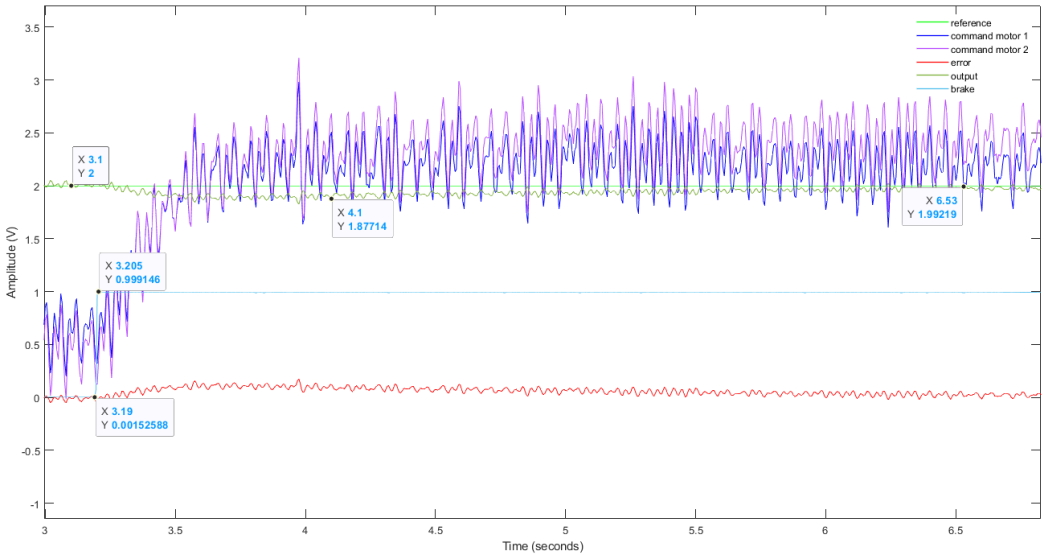


Figure 2.11: response of the PI controller with both motors for a step disturbance

3 LQR Controller

3.1 Introduction

After validating the PID controller, the TA asked if it would be possible to design another controller able to only use the second motor when the first one struggles to reach the desired speed. This desire suits to the system as the motor 2 is weaker than the first one: it can be seen easily as its time constant τ is greater than the one of motor 1, meaning that it needs more time to reach a set value.

3.2 State Space Model

In control system design, it is often easier to define a parameterized state-space model in continuous time because physical laws are typically described using differential equations. The linear state-space representation in continuous-time has the following form:

$$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u} \tag{3.1}$$

where $\mathbf{x}$ is the state vector, A is the state matrix that represents the system dynamics, B is the input matrix that represents the control input, and $\mathbf{u}$ is the input vector.
To get to the state space representation from the transfer function is quite easy as each motor is a 1^{st} order system. By using the following property of the Laplace transform:

$$X(s) = \mathscr{L}\{x(t)\} \tag{3.2}$$

$$sX(s) = \mathscr{L}\left\{\frac{dx(t)}{dt}\right\} \tag{3.3}$$

The transfer function 1.12 can be put in the temporal domain:

$$G(s) = \frac{Y(s)}{U(s)} = \frac{A_0}{\tau s + 1} \tag{3.4}$$

$$(\tau s + 1)\, Y(s) = A_0 U(s) \tag{3.5}$$

$$\tau \dot{y}(t) + y(t) = A_0 u(t) \tag{3.6}$$

$$\dot{y}(t) = \frac{-1}{\tau} y(t) + \frac{A_0}{\tau} u(t) \tag{3.7}$$

This can be done for both motors with respectively (u_1 / y_1) and (u_2 / y_2) as input and output. With the choice of state $x_i(t) = y_i(t)$ representing the contribution of each motor to the total velocity, the state space representation becomes:

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} \frac{-1}{\tau_1} & 0 \\ 0 & \frac{-1}{\tau_2} \end{bmatrix} \mathbf{x}(t) + \begin{bmatrix} \frac{A_{01}}{\tau_1} & 0 \\ 0 & \frac{A_{02}}{\tau_2} \end{bmatrix} \mathbf{u}(t)$$

$$y(t) = \begin{bmatrix} 1 & 1 \end{bmatrix} \mathbf{x}(t) \tag{3.8}$$

where y is the velocity output, which is just the sum of both imposed speeds $(y = y_1 + y_2)$. It is however known that the states are not really independent so this state space representation is not accurate.the following simplified model will be used from now where the state is a single value, namely the velocity of the shaft.

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} -\frac{\tau_1 + \tau_2}{2\tau_1 \tau_2} \end{bmatrix} \mathbf{x}(t) + \begin{bmatrix} \frac{A_{01}}{\tau_1} & \frac{A_{02}}{\tau_2} \end{bmatrix} \mathbf{u}(t) \tag{3.9}$$

$$y(t) = \mathbf{x}(t) \tag{3.10}$$

3.3 Discrete-Time State-Space Model

We need to convert the continuous-time state-space model (3.9) and (3.10) to a discrete-time state-space model to design the controller in discrete time. This is the other approach compared to the one used in section 2 as we were there building the controller in continuous time and then we discretized it. Using the parameters from the identification section (1):

$$\tau_1 = 1.915, \quad \tau_2 = 1.7, \quad A_{01} = 24.88, \quad A_{02} = 19.51$$

The continuous-time matrices are computed:

$$\mathbf{A} = \begin{bmatrix} -0.5552 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 12.9870 & 11.4765 \end{bmatrix}.$$

For a sampling time $T_s = 0.005$ seconds, the discrete-time matrices A_{T_s} and B_{T_s} are calculated as follows:

- **Discrete-Time State Matrix** (A_{T_s}): Using the matrix exponential formula:

$$A_{T_s} = e^{AT_s} = \begin{bmatrix} 1.0028 \end{bmatrix}$$

- **Discrete-Time Input Matrix** (B_{T_s}): Using the formula:

$$B_{T_s} = \int_0^{T_s} e^{A\zeta} B \, d\zeta = \begin{bmatrix} 0.0648 & 0.0573 \end{bmatrix}$$

Thus, the discrete-time state-space representation of the system is:

$$\mathbf{x}[k+1] = \begin{bmatrix} 1.0028 \end{bmatrix} \mathbf{x}[k] + \begin{bmatrix} 0.0648 & 0.0573 \end{bmatrix} \mathbf{u}[k]$$

$$\mathbf{y}[k] = \mathbf{x}[k]$$

Finally, the sampled system is equivalent to a discrete-time system with these matrices. We will now use this representation to design an LQR controller for the given discrete-time system.

3.4 Controller design

Simple state feedback

In the case of this system and because the model has been simplified to a single state representation in section 3.2, there is no need to build an observer as the state is the velocity measured by the speed sensor. This gives rise to a really simple block diagram:

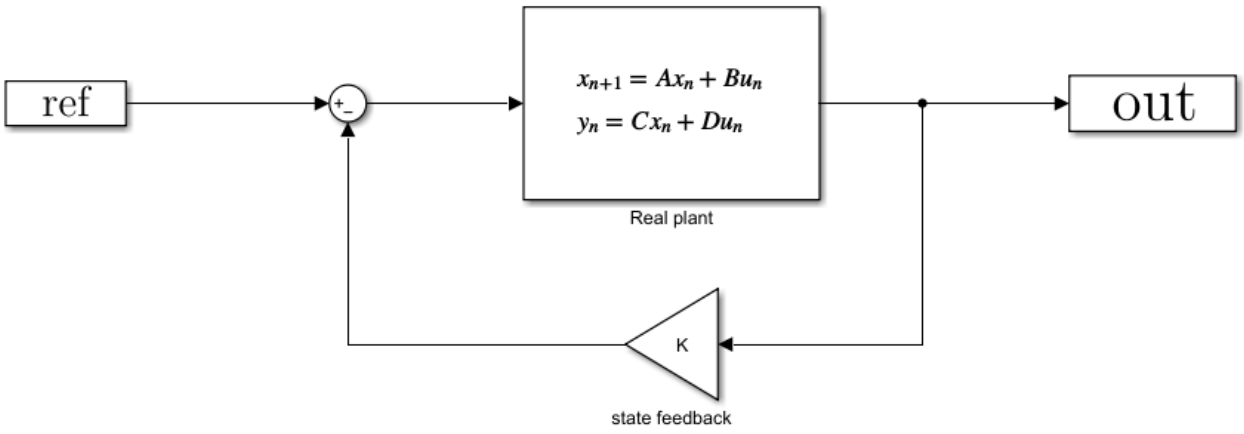


Figure 3.1: Structure of the LQR controller

The determination of the value for K has been done using the LQR algorithm.

The goal of the Linear Quadratic Regulator (LQR) is to minimize the quadratic cost function:

$$J = \sum_{k=0}^{\infty} \left(x[k]^\top Q x[k] + u[k]^\top R u[k] \right).$$

Where the weighting square matrices Q and R must be selected based on system requirements:

- Q: Penalizes state deviations
- R: Penalizes large control efforts

Because we have a single state, Q will have no impact. The interesting parameter to play with is the matrix R as the value on its diagonal will determine the "*cost*" of each input. The following choice has been made because, as said in the introduction 3.1, we want to use the second motor only when the first one cannot produce enough torque:

$$Q = 1, \quad R = \begin{bmatrix} 1 & 0 \\ 0 & 5 \end{bmatrix}.$$

We then computed the feedback matrix using the matlab function `dlqr`:

$$K = \begin{bmatrix} -0.0368 \\ -0.0065 \end{bmatrix}$$

Finally, the step response of the system with a state feedback has been measured (fig 3.2). We can conclude from the data below that with a properly designed state feedback, the controller can mainly use the 1^{st} motor and only send an input to the second motor when the reference is hard to reach (e.g. the step). However this is far from satisfying us as there is a steady state error. In the next section, we'll explain how to get rid of it.

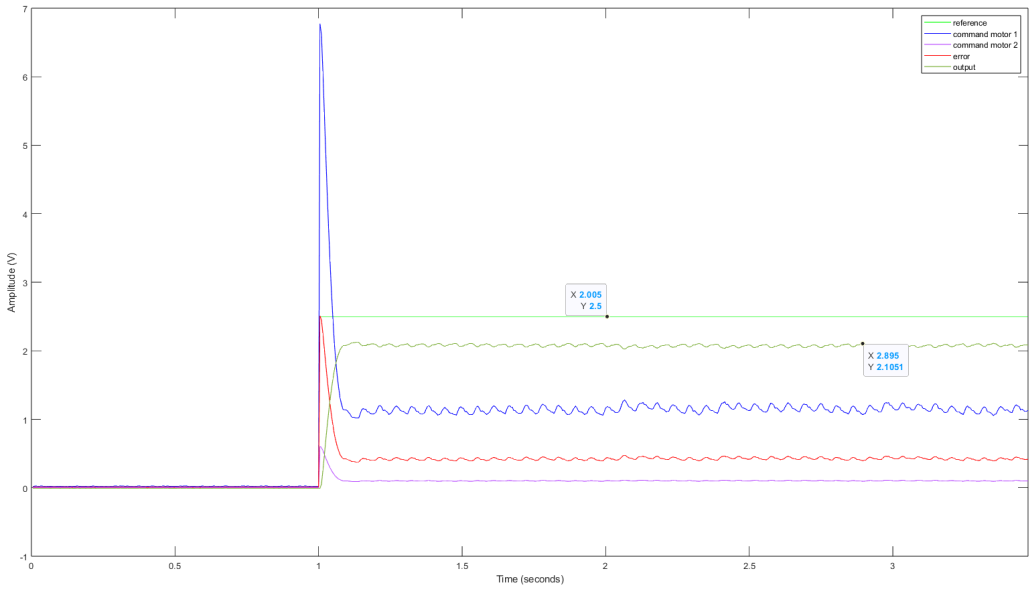


Figure 3.2: Step response with a state feedback

state feedback with an integrator

To ensure a zero steady-state error, the closed loop system must contain a pole at the origin. This can be very easily done by adding an integrator in front of the feedback. However, the matlab documentation gives a function specifically designed for an LQR state feedback with an integrator: `lqi`.

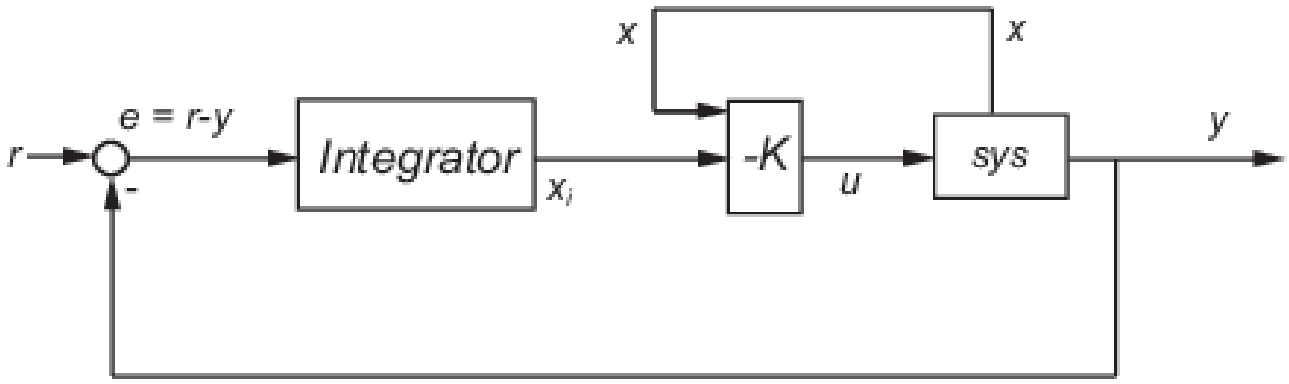
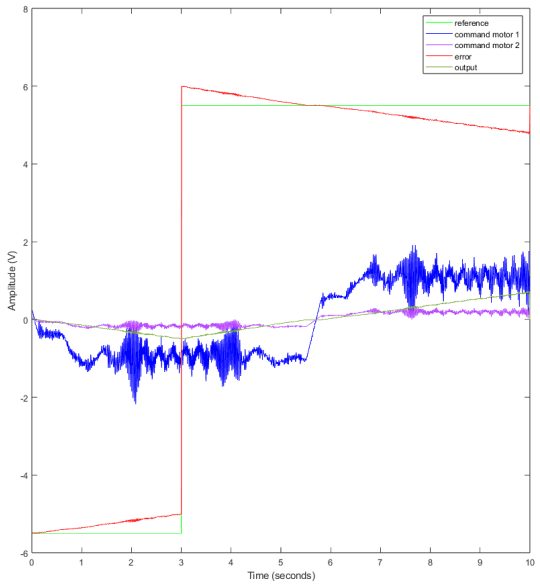
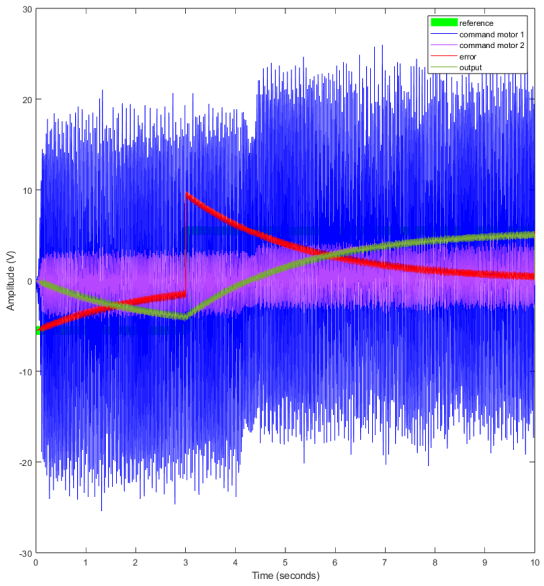


Figure 3.3: LQI structure, from matlab documentation

This structure implies that K becomes a $2x2$ matrix because its input is now the state x and the output of the integrator x_i. The integrator has been discretized using the Tustin equivalence and the new K has been computed using the same R matrix than before. After trying with $Q = 1$, it has been increased to reduce the settling time which gave the second graph.



(a) Step response of the LQI with $Q = 1$



(b) Step response of the LQI with $Q = 500$

Figure 3.4: Comparison of step responses for different Q values

Increasing Q indeed helped to make the system reach the reference faster but it comes with an increase in the command swing. A way to make it a better was to express Q as a $2x2$ matrix. Let's recall from the previous section that Q penalizes state deviation and because of the integrator, the state has become $\begin{bmatrix} y(t) & x_i(t) \end{bmatrix}^T$ which means that only the integrator state (which controls the settling time) could be increased.

$$Q = \begin{bmatrix} \frac{1}{100} & 0 \\ 0 & 100 \end{bmatrix}$$

Allowed indeed a settling time close to the one of $Q = 500$ shown above (3.4a) with less input fluctuations. The best way found was in fact to let $Q = 1$ and to give more impact to the integrator by adding a multiplier block K_i at its output, giving the control scheme:

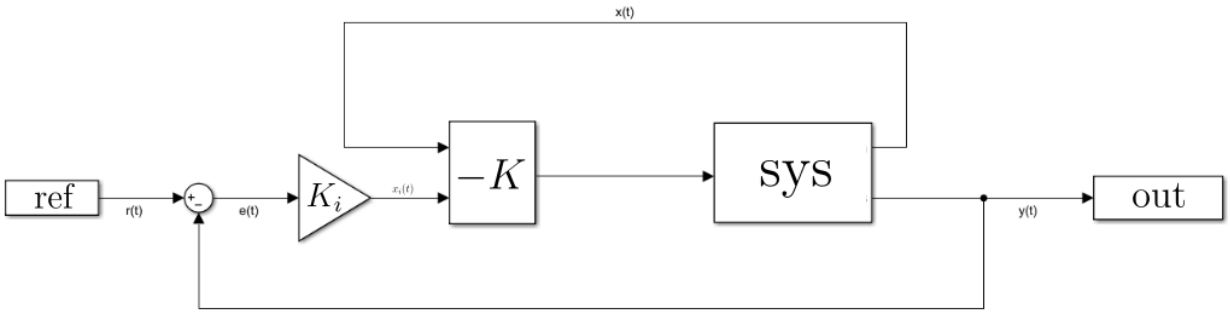
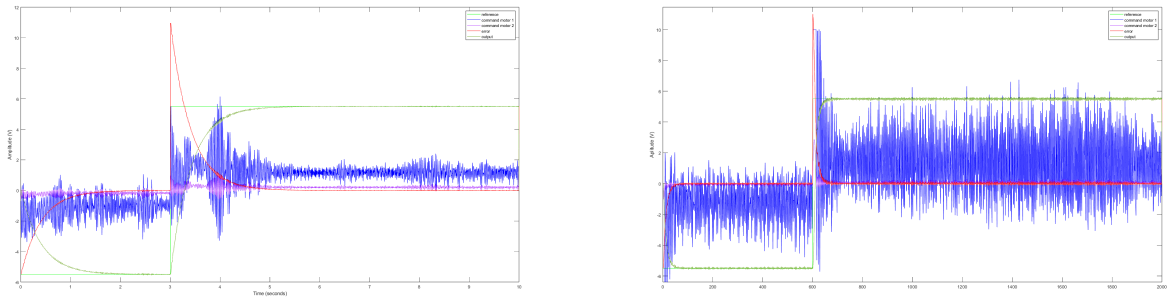


Figure 3.5: LQI adapted structure

Here again, two experiments have been made. The one on the left is with a low K_i (= 50) and $R = \begin{bmatrix} 1 & 0 \\ 0 & 10 \end{bmatrix}$ where the second test was conducted with $K_i = 500$ and R has been set to $\begin{bmatrix} 1 & 0 \\ 0 & 20 \end{bmatrix}$.



(a) Step response of the LQI with low K_i

(b) Step response of the LQI with high K_i

Figure 3.6: Comparison of step responses for different K_i values

The second one satisfying the settling time requirement, it has been chosen as the final LQI controller. A closer look at the step (done in figure 3.7) shows that the second motor is indeed used when the 1^{st} motor is not able to provide enough torque by itself (when a step reference is applied).

Due to the limited time we had in the lab, we were not able to conduct a disturbance test on this controller but as it contains an integrator, it is (*at least theoretically*) able to reject a step disturbance. The same thing applies for the frequency response where the limited time only allowed us to manually measure the amplitude and the phase of the closed loop at $4Hz$ thanks to figure 3.8.

$$\text{Amplitude}\Big|_{f=4Hz} \simeq \frac{2.858}{4.99} = -4.84dB$$
$$\text{Phase}\Big|_{f=4Hz} \simeq \frac{0.315 - 0.355}{0.25} = -57.6^{\circ}$$

So we are definitely not achieving the 4^{th} requirement which was to perfectly track a $4Hz$ signal.

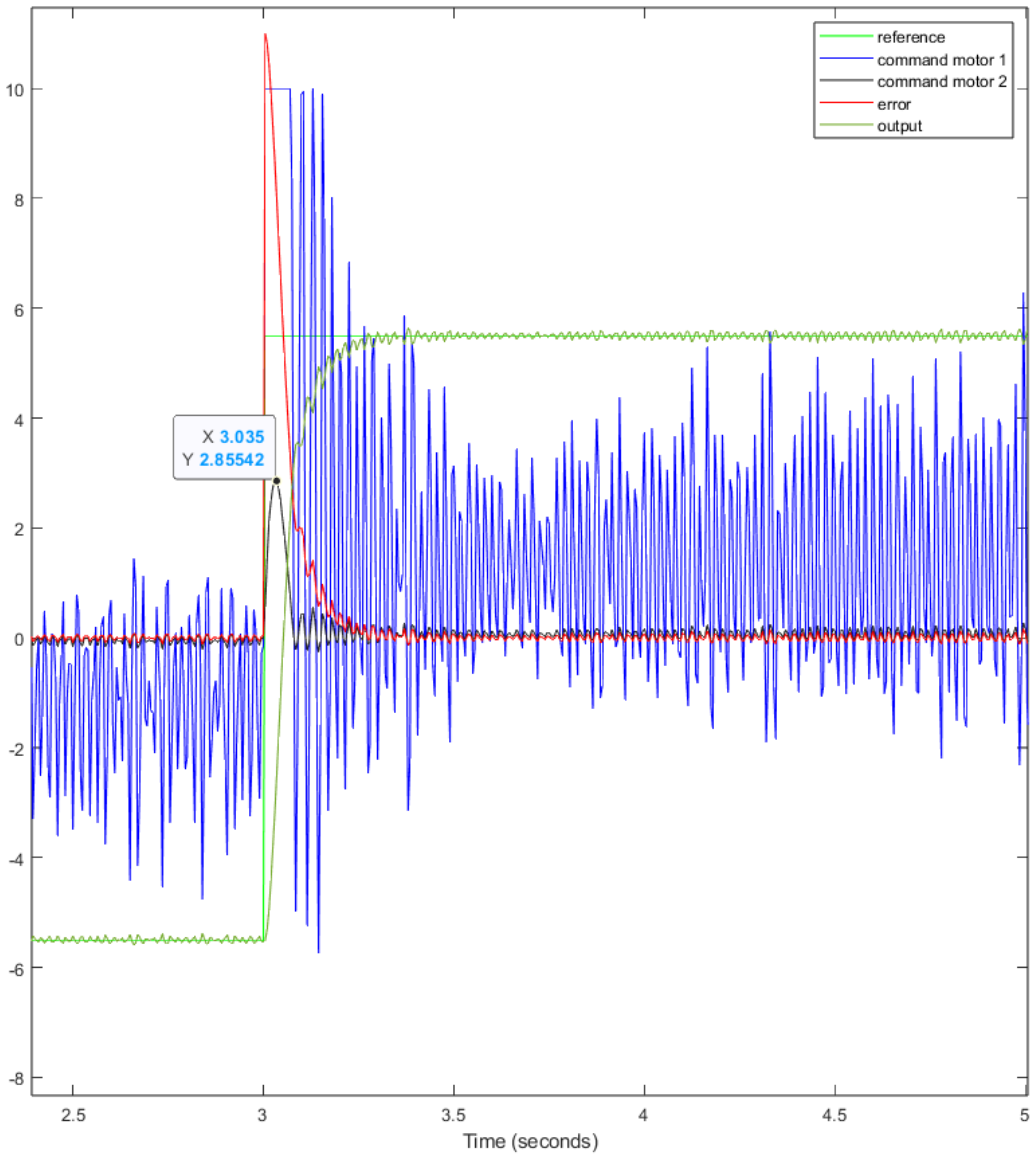


Figure 3.7: closer look to the step response of the final LQI controller

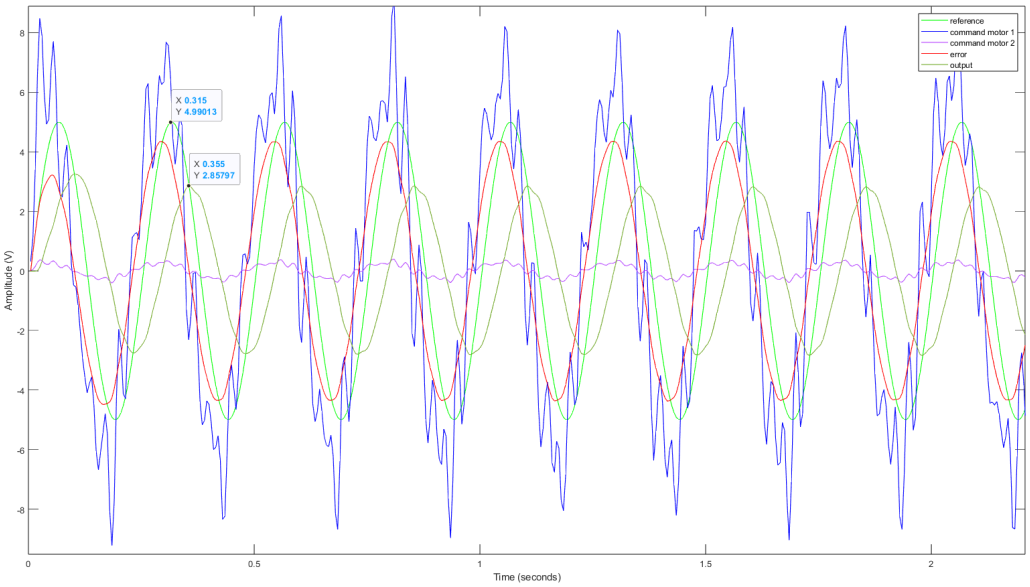


Figure 3.8: Response of the final LQI controller to a $4Hz$ sine reference

Conclusion

blabla