UCLouvain

École polytechnique de Louvain

Control in a human-inspired model of reaching movements:

Comparison between optimal feedback and trajectory based controllers

Author: **Cédric TOUSSAINT**
Supervisor: **Frédéric CREVECOEUR**
Readers: **Philippe LEFÈVRE, Julien HENDRICKX, Hari KALIDINDI**
Academic year 2021–2022
Master [120] in Mathematical Engineering

Abstract

This master thesis investigates the capacities of two controllers, whose operating principles are significantly different, to mimic the human behavior when performing reaching movements. The first controller is purely based on the optimal feedback theory while the second one is a mix of this theory and another that supposes that the motor commands are based on a desired trajectory. Several simulations, based on two real experiments, are carried out to compare the trajectories obtained by each of the controllers with the trajectories realized by humans. It appears that the controller purely based on the optimal feedback theory computes trajectories much more similar to those realized by humans than the other controller.

Acknowledgements

I would first like to thank my thesis director, Professor F. Crevecoeur, for his patience, his availability and above all his judicious advice, which guided me and contributed to my reflection.

I would also like to thank Professor P. Lefèvre, Professor J. Hendrickx and Dr H. Kalindi for agreeing to be part of my thesis jury as readers.

And finally, I would like to thank my parents for their support and valuable help in the proofreading this thesis.

Contents

Abstract **i**

Acknowledgments **ii**

1 Introduction **1**
1.1 Motivation . . . 1
1.2 Structure of the thesis . . . 1
1.3 Linear dynamical system notations . . . 2

2 Sensorimotor control in humans **3**
2.1 The structure of the sensorimotor control . . . 4
2.2 Experiments with a long-latency response . . . 6

3 State of the art **9**
3.1 LQG controller . . . 9
3.2 2-link 2-muscle model . . . 11
3.3 Linearization . . . 12
3.4 Delay . . . 15
3.5 iLQR . . . 15
3.6 Main differences . . . 18

4 Implementation **19**
4.1 State vector augmentations . . . 19
4.1.1 Affine term . . . 19
4.1.2 Perturbations . . . 20
4.2 Wide target . . . 20

5 Results **23**
5.1 Initial control sequence for iLQR . . . 24
5.2 Hyperparameters . . . 25
5.2.1 iLQR . . . 25
5.2.2 LQG$_t$ . . . 26
5.3 Delay . . . 28
5.4 Noisy system . . . 29
5.4.1 LQG$_t$ . . . 29

5.4.2 iLQR 32
5.5 Perturbed movement 35
5.5.1 Target point 35
5.5.2 Target axis 41

6 Discussion **47**
6.1 Similarity to the human behavior 47
6.2 Sensitivity 50
6.3 Possible improvements 51

7 Conclusion **53**

Chapter 1

Introduction

1.1 Motivation

Human beings are capable of doing simple and complex movements every day, like reaching for a glass of water or hitting a ball with a tennis racket. But even though we are capable of doing those movements, it is still very hard for us to understand how the brain manages to produce motor commands to realize those movements.

The nonlinearities present in the model used for the human arm make it very difficult to study the structure of the human control.
Indeed, in the literature two opposing ideas are present [1]. The first one is that the brain relies on a virtual trajectory and on the inverse dynamics of the human body to find the appropriate motor commands corresponding to the desired trajectory [2]. The second one is that the motor commands are chosen, based on forward dynamics, to minimize a certain cost function that depends on the behavioral goal [3].

The purpose of this thesis will be to compare two different controllers with the human behavior. One of the two controllers relies purely on the second idea and the other one is a mix of both.
To do so, two experiments will be reproduced with those controllers and the obtained trajectories will be compared to real trajectories realized by humans in the same conditions. The trajectories will be compared in a qualitative way on the basis of two criteria: flexibility and adaptability.

1.2 Structure of the thesis

The thesis is divided in five different parts:

- Firstly, a summary of the sensorimotor control in humans is given as well as the experiments that will be used to compare the two controllers to the human behavior.
- Then, the model used to reproduce the dynamics of the human arm and the two

controllers used are presented.

- After this, the features added to the model and controllers as well as their implementations are detailed.
- Next, a lot of simulations are performed to test the controllers in a wide variety of situations: when a delay is introduced, when the target is small or large, when the process or observation noise is high and when one of the arm parts is mechanically disturbed.
- Afterwards, the different results obtained are compared and discussed.
- Finally, some leads are given for future works in the continuity of this one.

1.3 Linear dynamical system notations

Since the purpose of this thesis is about investigating how well controllers mimic the human behavior of reaching movements, a quick reminder of the notations used for linear dynamical systems is given in this section.

The system of equations 1.1 is a representation in continuous time of a linear dynamical system similar to the one used in this work.

$$\begin{cases} \dot{x}(t) = A_c x(t) + B_c u(t) \\ y(t) = H x(t) \end{cases} \tag{1.1}$$

where the vector x is the state vector, u is the input vector and y is the output vector.

In order to implement this system, it is necessary to work with a discrete system and that can be achieved with the explicit Euler integration over one time step:

$$\begin{aligned} x(t+\delta t) &\approx x(t) + \delta t \dot{x}(t) \\ &= x(t) + \delta t (A_c x(t) + B_c u(t)) \\ &= (\mathcal{I} + \delta t A_c)\, x(t) + \delta t B_c u(t) \end{aligned} \tag{1.2}$$

All that is left to do is to define $A = \mathcal{I} + \delta t A_c$ and $B = \delta t B_c$ and to set the index k equal to t and the time step equal to δt. This gives us the following system:

$$\begin{cases} x_{k+1} = A x_k + B u_k \\ y_k = H x_k \end{cases} \tag{1.3}$$

Because the arm movements that will be simulated have stochastic components, we still have to make a little modification to the system 1.3 by adding noises:

$$\begin{cases} x_{k+1} = A x_k + B u_k + \xi_k \\ y_k = H x_k + \omega_k \end{cases} \tag{1.4}$$

with ξ and ω being the process and the observation noise vectors. In all this work, the covariance matrices of those random variables will be diagonal and their values are constants.

Chapter 2

Sensorimotor control in humans

The purpose of this work is to compare two different controllers that mimic the human behavior when performing reaching movements. To do so, it is interesting to first have a look at how the control of the arms works in humans.
As mentioned earlier in the introduction, two opposing ideas of how the nervous system works exist:

- The first one is that the control realized by the nervous system is based on a desired trajectory. In this theory, sensory feedback is considered as an additional feature that can be used if it is necessary.
  This theory of a controller based on a desired trajectory is supported by some behaviors observed in the way people are performing reaching movement. One of them is that the hand's trajectories realized by humans tend to be straight even though this is not optimal. Another one occurs when a movement is perturbed, people tend to bring their hand back to the original trajectory made when there was no perturbation [4]. But we will see in the next point that this last behavior can be associated with some specific tasks that would explain why the motor control acts as a trajectory based controller.

- The second idea is that the nervous system works as an optimal feedback controller (OFC). The OFC is a very interesting approach to explain how the sensorimotor control works in humans because it identifies the best way to realize a movement while minimizing a cost function [5] and it deals very well with noisy systems as well as movements subject to an external mechanical perturbation. It has also been shown that the OFC is compatible with the experiment where people bring their hand back to the original path when facing a perturbation. Indeed, this behavior could be linked to the integration of a timing constraint in the movement and because of this timing constraint, the sensorimotor control starts to behave like a trajectory based controller. A similar behavior from the sensorimotor control is observed when the person is asked to follow a precise trajectory.
  Furthermore, some studies have related the neural activity in the primary motor cortex to the movements of the hand, whereas others have related the neural activity of the same region of the brain to the generation of motor commands [6]. It appears

that the framework of optimal control theory combines those two observations very well [7][8].

As the two controllers that we will work with are based on the OFC theory because this approach has been at the center of interest over the past two decades, this chapter will summarize the structure of the sensorimotor control in humans seen as an optimal feedback controller.

2.1 The structure of the sensorimotor control

The sensorimotor control can be decomposed into three different blocks that interact with each other [9]. Those blocks are represented on figure 2.1 as "What", "How" and "Where". Each one of those blocks represents a specific part of the sensorimotor control.

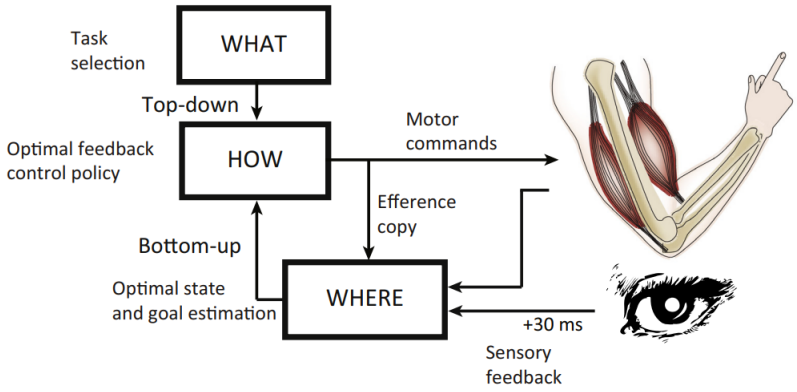

Figure 2.1: Figure from [9]
representing a simplified structure of the sensorimotor control in humans

What

This block represents all the voluntary control, it can be something simple like holding your arm in front of you or more complex like driving a car.
In this work, the simulated tasks will be to reach for a target in front of the person. This target will either be a point or a horizontal bar. We will see how that can change the control made by the person.
It also important to notice that this block is directly connected to the block "How" through a Top-down connection. This connection has quite a long delay, especially compared to the Bottum-up connection that is responsible for the sensory feedback.

Where

On figure 2.1, we can see that there are actually three different arrows that go into the "Where" block. Each of them corresponds to a different sensory feedback:

- The first one called the "efference copy" corresponds to a copy of the input signal sent to the muscles by the nerves. This copy is very important because as we will discuss later, the different types of feedback have different delays and those delays can induce instabilities especially for rapid movements. This path of feedback allows to have a first useful feedback with the lowest delay possible.

- The second type of feedback corresponds to the afferent nerves, i.e. those who carry the sensory information from the skin and the muscles back to the central nervous system. The delay of a goal-dependent feedback using information from afferent nerves is at least 60ms and that delay can increase depending on the task wanted.

- The last type of feedback is the visual one. Even though this type of feedback is very important for motor control, we can see on figure 2.1 that it has a 30ms delay in addition of the delay of the previous type of feedback.

As stated above, all the feedbacks' delays can induce a lot of instabilities for fast movements. To prevent this from happening, the feedback from the efferent nerves, that has a very short delay compared to the others, is used for a motor prediction based on an internal model of the body's dynamics and the copy of the motor commands[10]. This allows to make an estimate about the position of the body.
But because this single method of forward dynamics cannot predict the impact of external perturbations, there is a need for other types of feedback that will react to those perturbations. This is where the feedbacks through afferent nerves and vision start to be useful, those two types of feedback are combined to get an estimate of the actual position of the body.

How

This is the most important block of the sensorimotor control in this work. It is the part that determines how the feedback from the block "Where" will be used in order to compute the new motor commands sent to the muscles to achieve the goal determined by the block "What". That is exactly what we want to explore in this work by comparing two different candidates of controllers that are both based on the OFC theory but that differ in the way they compute the feedback gains.

The optimal feedback theory is very convenient because as said earlier, it allows to deal with noisy systems and external perturbations. But the actual organization of the sensorimotor control could be very different from figure 2.1 if the feedback was only taken into account for voluntary movements. The Bottom-up link would go to the block "What" and the block "How" would not be based on any feedback except through the Top-down link that was mentioned earlier.
To ensure that this is not the case, some experiments have been done where the motor responses to external perturbations during reaching movements have been measured. It appeared that there are two different epochs that were each related to different types of reflexes [11][12]:

- The first epoch corresponds to the short-latency responses, also called R1, and is contained in a 20-45ms interval. The responses contained in that interval are generally non-goal oriented. They are generated by the spinal cord and it has been shown that those responses are not very flexible.

- The second epoch corresponds to long-latency responses and is actually divided in two different parts. The first, called R2, occurs in a 45-75ms interval. and the second, called R3, occurs in a 75-105ms interval.
  Those responses unlike the short-latency ones contain a transcortical pathway that goes through the primary motor cortex.

In the context of this work, we are more interested in the second type of responses because we want to identify which mathematical controller will better mimic the human one in a qualitative approach. And in order to compare the two controllers, we will reproduce experiments where a long-latency response was observed.
It is also important to highlight that even though it does not appear on figure 2.1, the block "How" is actually much more complex. A same sensory feedback can either have an impact on the optimal way to reach an object, on the choice of target when the person is given multiple possible targets or even on the next task that the person wants to achieve. The delay between the sensory feedback and the corresponding response will be different for each one of these situations.
However this goes beyond the scope of this work and we will not deal with those varying delays since all the experiments we will work with have an impact on the same thing, i.e. the optimal way to reach the target.

2.2 Experiments with a long-latency response

In order to compare the behavior of the two controllers that we have implemented, we will perform simulations in several different conditions corresponding to two experiments that have been performed with humans [13]. Then, we compare the trajectories obtained with the human ones.

The first experiment consists in asking people to reach for a small circular target situated in front of them and no external perturbation is applied. The same people are then asked to reach again for a target in front of them, but this time the target is no longer a small one but rather a large horizontal bar. Again no external perturbation is applied during the reaching movement.
As we can see on figure 2.2, the end points of the trajectories when the target is a large bar are much more spread along that bar than when the target is small [11][13]. This highlights the fact that the corrective movements to compensate for the noises take the shape of the target into account. We will compare those results with the ones obtained with the controllers.

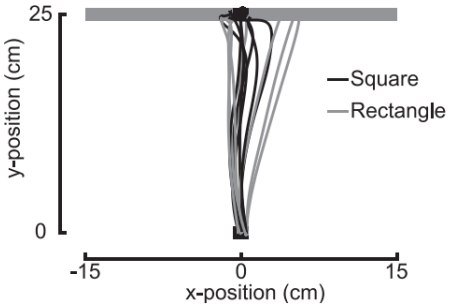

Figure 2.2: Figure from [13]

The second experiment is a little bit different. The people are still asked to reach for a small target in front of them but during the reaching movement, a mechanical perturbation is applied on the arm. Then, the people do the same experiment but with a large horizontal bar like in the first experiment.
We can see on figure 2.3 the results of that experiment. The solid lines correspond to the group's mean and the dashed ones are the standard errors. We can see that when the target is a square, the perturbation is corrected and that most of the trajectories seem to reach the target very well. When the target is a large bar, the perturbation is not compensated as much as before and people let their hands being carried further away from the original target in front of them.

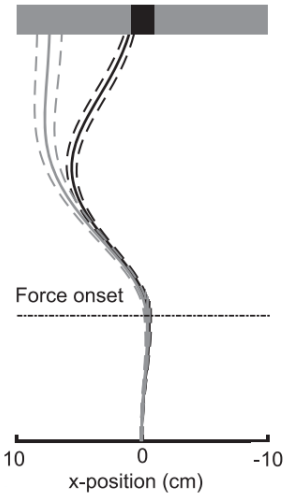

Figure 2.3: Modified figure from [13]

Notice that even though the trajectories arrive more to the left when the target is a rectangle than when it is a square, the trajectories are perpendicular to the rectangle at the end. This might be surprising because OFC theory is based on the minimum intervention principle which says that corrective movements only occur when a deviation interferes with the cost function. We might then wonder why people don't just let their hand move to the left during the movement, since they will hit the target anyway. But it is in fact

due to the need to stop the hand when arriving on the target. If the people just let their hand being carried to the left without correcting that perturbation before arriving to the target, it would require them much more energy to stop it. Therefore, they start to correct them as soon as possible.

Chapter 3

State of the art

This chapter first describes the LQG controller. Then, the model used to reproduce the human's arm dynamics is detailed. After, two different ways to linearize it are given and motivated. Next, two other controllers are described, the iLQR and the Minimum Jerk Trajectory generator. Finally, the way the iLQR is adapted to work with a noisy system like the one used in this work is presented.

3.1 LQG controller

In the previous chapter, we talked about how the motor control could be seen as an optimal feedback control, and one of the most famous optimal feedback controller is the LQG controller. This controller is divided into two different parts:

- a Kalman filter that computes the optimal estimates of the state vector of stochastic systems
- a feedback controller LQR that minimizes a quadratic cost function for deterministic systems

Kalman filter

The first part of the LQG controller is the Kalman filter. As said above, this filter allows to extract an optimal estimate of the state vector from the noisy output of the system. This is very interesting in this case because there are two sources of noise in the model: the process noise and the observation noise.

The Kalman filter computes a series of Kalman gains K_k in a forward recursion as follows [14]:

$$\begin{aligned} K_k &= A\Sigma_k H^T \left(H\Sigma_K H^T + \Omega_\omega\right) \\ \Sigma_{k+1} &= \Omega_\xi + (A - K_k H)\,\Sigma_k A^T \end{aligned} \tag{3.1}$$

where A and H are the matrices defined in the equation 1.4, Σ_1 is the covariance matrix of the initial estimate $\hat{x}_1$ and, Ω_ξ and Ω_ω are respectively the covariance matrices of the

process noise and the observation noise.
The optimal estimate is then extracted at each iteration with the following equation:

$$\hat{x}_{k+1} = A\hat{x}_k + Bu_k + K_k\left(y_k - H\hat{x}_k\right) \tag{3.2}$$

where $\hat{x}_1$ is a random vector with a normal distribution centered in x_1 and with covariance matrix Σ_1.

LQR

The second part of the LQG controller is the Linear Quadratic Regulator. The LQR is a feedback controller that minimizes a quadratic cost function of the following form [14]:

$$J(x,u) = x_N^T Q_N x_N + \sum_{k=1}^{N-1}\left(x_k^T Q_k x_k + u_k^T R u_k\right) \tag{3.3}$$

In this work, the matrices Q_k will always be equal to zero except for Q_N because the positions and the velocities of the hand during the movement are not penalized.
The matrix R is a diagonal matrix whose elements are the coefficients that will give more or less importance to the penalization of the command input sent to the muscles.
The matrix Q_N is designed to minimize a final cost function of the form:

$$J_f(\theta,\dot{\theta}) = C_{pos}\left\|\theta_N - \theta^*\right\|^2 + C_{speed}\left\|\dot{\theta}_N\right\|^2 \tag{3.4}$$

with the vector θ_N containing the elbow and the shoulder angles of the final position of the hand, θ^* the vector of target angles and $\dot{\theta}_N$ the final angular speeds of the joints.

We will see later in this chapter that the state vector we are working with include the target angles and it allows to express the function 3.4 above in a quadratic form $x_N^T Q_N x_N$.
In this case, the matrix Q_N is the following:

$$Q_N = \begin{pmatrix} C_{pos} & 0 & 0 & 0 & 0 & 0 & -C_{pos} & 0 \\ 0 & C_{pos} & 0 & 0 & 0 & 0 & 0 & -C_{pos} \\ 0 & 0 & C_{speed} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & C_{speed} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -C_{pos} & 0 & 0 & 0 & 0 & 0 & C_{pos} & 0 \\ 0 & -C_{pos} & 0 & 0 & 0 & 0 & 0 & C_{pos} \end{pmatrix} \tag{3.5}$$

Now that the matrices included in the cost function have been detailed, the following

backward recursion can be computed:

$$\begin{aligned} S_N &= Q_N, \quad s_N = 0, \\ L_k &= \left(R + B^T S_{k+1} B\right)^{-1} B^T S_{k+1} A, \\ S_k &= Q_k + A^T S_{k+1} \left(A - B L_k\right), \\ s_k &= s_{k+1} + \mathrm{tr}\left(S_{k+1} + \Omega_\xi\right). \end{aligned} \tag{3.6}$$

where L_k is the feedback gain matrix that is used online to determine the input command at the step k:

$$u_k = -L_k \hat{x}_k \tag{3.7}$$

with $\hat{x}_k$ being the optimal estimate computed with the equation 3.2.
Thanks to a very nice property of the LQG, called the separation principle, the coefficients of the Kalman filter and the feedback gains of the LQ command can be computed independently and offline.

3.2 2-link 2-muscle model

Now that we have seen a first optimal feedback controller, we need to find a model that will reproduce the human's arm dynamics. Such a model would require an input corresponding to the command sent by the nerves to the muscles, an equation that would link a nerve impulse with a corresponding muscle contraction, and finally an output that would be the angles of the shoulder and the elbow in function of the muscle contractions.

A possible candidate for such a model is a 2-link and 2-muscle biomechanical model. This model is, as represent in figure 3.1, composed of two links: the forearm and the upper arm. Those two links are connected to each other by the elbow and the upper arm is connected to the body by the shoulder.
Please note that all the body, except for the arm, is assumed to be completely static.

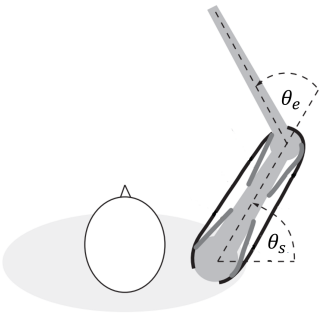

Figure 3.1: 2-link 2-muscle model (figure from [15])

At each of those two joints, there is one muscle that allows to apply a torque that can be both positive and negative. The muscle located on the shoulder will apply a torque to the upper arm and the one located on the elbow will apply a torque on the forearm.

The dynamic of those muscles is represented by a linear differential equation which is a low-pass filter:

$$\tau\dot{\mathcal{T}} = u - \mathcal{T} \tag{3.8}$$

with $\mathcal{T}$ being the torque applied by the muscle, u the signal sent by the brain to the muscles and τ a time constant. That time constant characterizes the speed taken by the muscle to contract when a signal is sent by the brain. In agreement with physiological studies [9], the value of that time constant is 60ms.

The inverse dynamics of this system, which link the angles of the joints with the torques applied, are described by this differential equation [15]:

$$\mathcal{M}(\theta)\ddot{\theta} + \mathcal{C}(\theta, \dot{\theta}) + \mathcal{B}\dot{\theta} = \tau \tag{3.9}$$

which, once inverted, becomes:

$$\ddot{\theta} = \mathcal{M}^{-1}(\theta)(\mathcal{T} - \mathcal{C}(\theta, \dot{\theta}) - \mathcal{B}\dot{\theta}) \tag{3.10}$$

where θ is the vector that contains both angles θ_s and θ_e of the figure 3.1, $M(\theta)$ is a positive definite inertia matrix that depends on the vector θ, $C(\theta, \dot{\theta})$ is a vector containing the nonlinear effects of the centripetal and the Coriolis forces, and finally $\mathcal{B}$ is a joint-friction matrix that captures the viscous forces[16].
The values of those matrices are the following:

$$\begin{aligned} \mathcal{M} &= \begin{pmatrix} a_1 + 2a_2\cos\theta_e & a_3 + a_2\cos\theta_e \\ a_3 + a_2\cos\theta_e & a_3 \end{pmatrix} \\ \mathcal{C} &= \begin{pmatrix} -\dot{\theta}_e\left(2\dot{\theta}_s + \dot{\theta}_e\right) \\ \dot{\theta}_s^2 \end{pmatrix} a_2\sin\theta_e, \quad \mathcal{B} = \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} \end{aligned} \tag{3.11}$$

where $a_1 = I_1 + I_2 + m_2 l_1^2$, $a_2 = m_2 l_1 s_2$, $a_3 = I_2$ and $b_{11} = b_{22} = 0.05$, $b_{12} = b_{21} = 0.025$. The constants m_1 and m_2 are respectively the masses of the upper arm and the forearm, l_1 and l_2 are their lengths and I_1 and I_2 are their moments of inertia. Finally, s_2 is the distance between the elbow and the forearm's center of mass.

3.3 Linearization

We have since previously in this chapter that the LQG controller only works with linear systems and since the equation 3.9 that model the arm dynamics is not linear because of the terms $\mathcal{M}$ and $\mathcal{C}$, we need to find a way to linearize it.
To do so, two different methods are presented: linearization around a point and around a path.

Linearization around a point

When it comes to linearize a non-linear system, linearizing around an equilibrium point is a very common thing. This equilibrium point in this situation of reaching movements

might be the target point that must be attained for example[16].

This technique has the advantage of being really simple and straight forward. However, it completely ignores the vector $\mathcal{C}$ of our model because both terms of that vector are multiplied by angular speeds and those speeds will be equal to zero if we linearize around a point.
Furthermore, it also limits the possible simulated movement of the arm to a really close area around the equilibrium point, otherwise the approximations made during the linearization become too bad and the dynamics of the linear system are then too far from the dynamics of the original one.

The linearization around a point is represented in the figure 3.2. The black point in the middle is the equilibrium point around which the linearization is done and the different colored areas around that point represent the accuracy of the approximations qualitatively. The lighter the color is, the worst is the approximation and at some point, it becomes irrelevant to use the linearization.

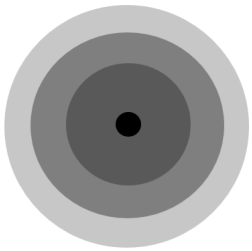

Figure 3.2: Equilibrium point

By doing such a linearization, the model can be rewritten in a form similar to the equation 1.4 where the state vector x and the input vector u are defined as follow:

$$\begin{aligned} x &= [\theta_s, \theta_e, \dot{\theta}_s, \dot{\theta}_e, \mathcal{T}_s, \mathcal{T}_e, \theta_s^*, \theta_e^*] \\ u &= [n_s, n_e] \end{aligned} \tag{3.12}$$

where θ_s^* and θ_e^* are the target angles and the elements n_s and n_e of u are respectively the neural signals to the shoulder and the elbow.
As we saw earlier in this chapter, it is necessary to include the target angles in the state vector in order to be able to write the cost function in a quadratic form.
The matrix A and B are the following:

$$A = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\mathcal{M}^{-1}B_{11} & -\mathcal{M}^{-1}B_{12} & \mathcal{M}_{11}^{-1} & \mathcal{M}_{12}^{-1} & 0 & 0 \\ 0 & 0 & -\mathcal{M}^{-1}B_{21} & -\mathcal{M}^{-1}B_{22} & \mathcal{M}_{21}^{-1} & \mathcal{M}_{22}^{-1} & 0 & 0 \\ 0 & 0 & 0 & 0 & -1/\tau & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1/\tau & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} B = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1/\tau & 0 \\ 0 & 1/\tau \\ 0 & 0 \\ 0 & 0 \end{pmatrix} \tag{3.13}$$

and the matrix H is the identity matrix.

Linearization around a path

Linearize around a point is quite a good approximation for movement that are slow and not too large. But, as soon as you start doing large movement or moving fast, the approximations about the angular positions equal to the target angles and the angular speeds equal to zero do not hold and the approximations are so far from the original dynamics of the system that the controller does not work anymore.

To avoid that, we can use another linearization technique which no longer consists in linearizing around a single point, but rather along a path as shown in figure 3.3.

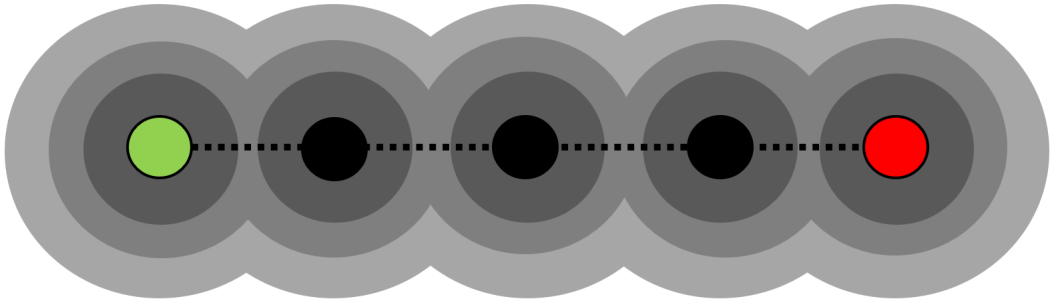

Figure 3.3: Equilibrium path

This figure represents a movement along a line with the green point being the starting point and the red point the ending point. The path of the movement is represented by the dashed points and the black points are the points where the linearization of the system is updated in the same way than when linearizing around an equilibrium point.
Like in figure 3.2, the darker the color is, the better the approximation of the linearization is. We can see that here, thanks to the multiple updates of the linearization along the path of the movement, the approximations are still good even if the ending point is far away from the starting point.
In fact, here the quality of the approximation will mainly depend on the distance between the black circles and thus on the rate of update of the linearization. This impact will be investigated in the results chapter.
Now that the linearization is done around several points in the trajectory, we can linearize with angular speeds different from zero and equal to the angular speeds of the joints at the positions used for the linearization. This will allow us to take the vector $\mathcal{C}(\theta, \dot{\theta})$ into account.

Finally, when using this type of linearization with the LQG controller, each time the linearization is updated, the backward recursion 3.6 of the LQR and the forward recursion 3.1 of the kalman filter are performed again with the new matrices of the updated linearization. This new controller updated during the movement will be denoted as the LQG_t in the rest of this thesis.

3.4 Delay

Defining the system as in the previous sections supposes that there is no delay between a change in the state and the command input from the brain reacting to that change. In fact, as we have seen in chapter 2, we know that there is a delay between the sensory feedback and the corresponding response contained in a 45-75ms interval. We take the R2 long-latency interval because the simulations that will be made should trigger a response in that interval. For the sake of simplicity, we will take the mean of that interval and work with a 60ms delay.
Notice that this delay has the same value as the constant in the equation 3.8 modelling the muscle but this is just a coincidence. Indeed, the former takes the delay between the stimuli and the reaction of the brain into account, and the latter model the progressive contraction of a muscle in reaction to a step input signal from the brain.
An easy way to implement this delay for linear systems is through the augmentation of its matrices and its state vector. [16].
The matrices A, B and H are modified to become the following matrices:

$$\bar{A} = \begin{pmatrix} A & 0 & \dots & 0 \\ I & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \dots & I & 0 \end{pmatrix}, \bar{B} = \begin{pmatrix} B \\ 0 \\ \vdots \\ 0 \end{pmatrix} \tag{3.14}$$

$$\bar{H} = (0 \dots 0H) \tag{3.15}$$

and the new state vector is defined as:

$$z_k^T := \left[x_k^T, x_{k-1}^T, \dots, x_{k-h}^T\right] \tag{3.16}$$

with h being the delay in number of time steps. The delay induced by the augmentation of the system is thus $h * \delta t$.

3.5 iLQR

The other controller that we will use in this thesis to investigate how good it mimics the human behavior is the iterative Linear Quadratic Regulator. This controller is based on the LQR that has been defined in the beginning of this chapter. The main difference is that it works as an iterative algorithm that converges towards an optimal trajectory for the system [17][18].
One might wonder why an iterative algorithm is required when the LQR is already supposed to minimize the cost function of equation 3.3. But as we saw previously in this chapter, the model we use is not linear and therefore we need to make approximations by linearizing it. Because of that, the solution found by the LQR is not necessarily the best to minimize the cost function.

The main steps of the algorithm are the following [19]:

1. Get an initial state x_0 and an initial control sequence $U = (u_0, ..., u_{N-1})$.
2. In a forward pass, perform a simulation based on the initial state and the control sequence U to get the corresponding trajectory X.
3. In a backward pass, estimate the Value Function (defined below) and dynamics for each (x, u) in the state-space and control signal trajectories.
4. Compute an updated control sequence U'.
5. Evaluate the cost of the trajectory X' simulated with the new control sequence and compare it with the cost of the previous control sequence:
   - if |cost(X',U')-cost(X,U)|$\leq$ threshold $\implies$ Exit, the algorithm converged.
   - if cost(X',U') < cost(X,U) $\implies$ $U = U'$ and $X = X'$ and change update size to be more aggressive. Go back to step 2.
   - if cost(X',U') $\geq$ cost(X,U) $\implies$ change update size to be more modest. Go back to step 3.

Three things present in this algorithm must still be detailed: the Value Function, the way the control sequence U is updated and what the update size is.

The first one is the Value Function but to define it, the cost-to-go function $G_t(x, U_t)$ must be defined first:

$$G_t(x, U_t) = \sum_{i=t}^{N-1} \ell\left(x_i, u_i\right) + \ell_f\left(x_N\right) \tag{3.17}$$

with $\ell\left(x_i, u_i\right)$ and $\ell_f\left(x_N\right)$ being the immediate and final costs. In our case, the final cost will be the equation 3.4 and the immediate cost is just the result of $u_i^T R u_i$.
Based on that, the Value Function can be defined as:

$$V_t(x) = \min_{U_t} G_t\left(x, U_t\right) \tag{3.18}$$

with $V(x_N) = \ell_f\left(x_N\right)$.

The second thing that must be explained is the procedure to update the control sequence. This part is the longest part of the algorithm because it requires to compute a lot of different matrices but the details of those computations will not be given in this work. They can be found in details on this blog as well as the implementation of this algorithm [19][20].
The idea is to use the gradient and hessian of the system dynamics as well as the immediate and final cost functions to compute two series of terms: feed-forward terms l_k and feedback terms L_k.
Those terms are then used to compute a new control sequence with the following equation:

$$u'_k = u_k + l_k + L_k(x'_k - x_k) \tag{3.19}$$

with x_k being the state of the system at the time step k of the trajectory simulated with the previous control sequence U.

The last thing that needs to be clarified is the update size. This thing is actually a constant λ that is used during the computations of the terms l_k and L_k when a matrix must be inverted.
To make sure that this matrix stays positive definite, which is important to guarantee a descent direction, the matrix is regularized. To do that, the eigenvalues and eigenvectors are calculated, all the negative eigenvalues are set equal to zero and then the term λ is added to every eigenvalue. The inverse matrix is then computed with those new eigenvalues and the eigenvectors found previously.

Now, one might wonder what impact on the algorithm can the value of the constant λ have. To understand that, one needs first to understand the intuition behind the value of an eigenvalue. The values of the eigenvalues represent the magnitudes of their corresponding eigenvectors. When the inverse of the matrix is computed, the eigenvalues are inverted and thus the contributions of the eigenvectors are flipped.
Adding a term λ to the eigenvalues before inverting them ensures that the inverted eigenvalues will never be greater than $1/\lambda$. The greater lambda will be, the lower will be that limit and this is equivalent to throw out information.
If the approximations of the system dynamics and the Value Function are really good, this is not interesting and the value of λ is decreased. When those approximations are not very accurate, the value of λ is increased in order to avoid the presence of eigenvalues with really high values that could induce instabilities.
The value of λ is increased by multiplying it by a constant named λ_{ratio} greater than 1 and is decreased by dividing it by the same constant. The initial value of λ when starting the algorithm is named λ_{start}.

One last thing that must be highlighted is that since this algorithm is based on the LQR, it is not designed to work with a stochastic system. Consequently, it will be explained in the end of this chapter how the algorithm was adapted to work with such a system.

Minimum jerk trajectory

We just saw in the previous section that the iLQR algorithm needed in its first step an initial control sequence.
In this work, we will compare the performances of the iLQR with two different initial control sequences. The first one will be computed with the LQG controller combined with a linearization around the target point. The second one will be obtained with the minimum jerk trajectory generation.

The minimum jerk trajectory is a technique that generates trajectories while minimizing the jerk, i.e. the derivative of the acceleration. This approach has the advantage of

generating straight line trajectories which is a behavior that is also found in human's reaching movements as mentioned in chapter 2.
The computation of the trajectory is pretty quick and is given by [21]:

$$\begin{aligned} x(t) &= x_0 + (x_f - x_0)\left(6\tau^5 - 15\tau^4 + 10\tau^3\right) \\ y(t) &= y_0 + (y_f - y_0)\left(6\tau^5 - 15\tau^4 + 10\tau^3\right) \end{aligned} \tag{3.20}$$

with $\tau = t/t_f$ and t_f the duration of the movement.
Notice that this time, the trajectory is not expressed in terms of angles but in Cartesian coordinates because it is the third derivative of the cartesian position that is minimized in order to get straight line trajectories. Nevertheless, it is possible to get the corresponding angular positions with a little bit of geometry and then compute the reversed dynamics of the system with the equation 3.9 and get the control sequence corresponding to the calculated trajectory.

iLQR with noise

The iLQR algorithm that was described just above is designed to work with deterministic systems. Since our system is stochastic, the algorithm needed to be adapted to work with the Gaussian noises.
The idea is to make it work the same way as the LQG, which as explained before, is composed of a kalman filter and a LQ command. Those two parts could be considered as respectively the blocks "Where" and "How" of chapter 2, the former doing the state estimation and the latter doing the optimal feedback control. The idea is to reproduce that organization by splitting the state estimation that will be done with the Kalman filter and the computation of the feedback gains that will be achieved by the iLQR algorithm. This is of course not optimal like it was for the LQG but as we will see in the results chapter, it works quite well.

3.6 Main differences

Two different controllers have been described in this section to simulate trajectories: the LQG_t and the iLQR. The utility of comparing those controllers is due to their differences in the way they control the system. It is therefore interesting to explore how they respectively behave when facing some experiments that have been done with humans.
As explained in chapter 2, there exist two opposing families of controllers in this field: the optimal feedback controllers and the trajectory based controllers. The LQG_t is obviously a member of the first category and the minimum jerk trajectory belongs to the latter. But the iLQR is a little bit between the two because it computes an ideal trajectory by iterating, thus that makes it a trajectory based controller but the optimization performed at each iteration is based on the LQR which is an optimal feedback controller. Therefore, the iLQR cannot be categorized as clearly as the two other controllers.
The purpose of this work will be to compare the LQG_t controller and the iLQR in the different situations presented in chapter 2 and see how they behave.

Chapter 4

Implementation

This chapter covers all the things that have been added or modified to the model presented in chapter 3 as well as how the cost function for a wide target was put into a quadratic form.
The first section is dedicated to the modifications made to the state vector in order to take into account the affine term that the 2-link model contains and also to permit the application of mechanical perturbations on the arm.
The second section details how to change the shape of the target from a single point to an axis.

4.1 State vector augmentations

4.1.1 Affine term

Since the term $\mathcal{C}(\theta, \dot{\theta})$ of the equation 3.9 is not multiplied by any variables contained in the state vector defined in the equation 3.12, it is for the moment impossible to take it into account even when linearizing the model along a path. That is because the only form of equation that we can use to model our system is the following:

$$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u} \tag{4.1}$$

and since the vector $\mathbf{C}(\theta, \dot{\theta})$ is not multiplied by any variables, it gives an equation of the following form:

$$\dot{\mathbf{x}} = A\mathbf{x} + \mathbf{C} + B\mathbf{u} \tag{4.2}$$

In order to get rid of the term $\mathbf{C}$, we need to build an augmented state space to keep the system in the form of the equation 4.1 [22]. Indeed, it is a common trick that allows to transform an affine function into a linear one. This is done by adding a constant equal to 1 in the state space vector and augmenting the other matrices accordingly:

$$\dot{\mathbf{x}}' = A'\mathbf{x}' + B'\mathbf{u} \tag{4.3}$$

with

$$\mathbf{x}' = \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix} \quad A' = \left[\begin{array}{ccc|c} & A & & \mathbf{C} \\ 0 & \cdots & 0 & 1 \end{array}\right] \quad B' = \begin{bmatrix} B \\ 0 \end{bmatrix} \tag{4.4}$$

$\mathbf{x}, \mathbf{u}, A$ and B being the vectors and matrices from the equations 3.12-3.13.
This way, we can keep a convenient form of system to work with the LQG controller and also take the term $\mathbf{C}(\theta, \dot{\theta})$ into account.
This trick will also be useful in one of the following sections when developing a new cost function corresponding to reaching movements with wide targets.

4.1.2 Perturbations

Since we want to be able to simulate reaching movements and see how the controllers react to different situations, it can be useful to enable the system to be perturbed by external mechanical torques. This way, we can observe what happens when a controller is facing an unexpected dynamic and compare it to how people react in real life.

There are several ways of implementing such a thing but the one that has been done here is to augment again the state space vector with 2 entries, each one corresponding to a torque applied to one of the two links.
This gives us a new augmented system where:

$$\dot{\mathbf{z}} = A''\mathbf{z} + B''\mathbf{u} \tag{4.5}$$

with

$$\mathbf{z} = \begin{bmatrix} \mathbf{x}' \\ \mathbf{T_{s,pert}} \\ \mathbf{T_{e,pert}} \end{bmatrix} \quad B'' = \begin{bmatrix} B' \\ 0 \\ 0 \end{bmatrix} \tag{4.6}$$

$$A'' = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\mathcal{M}^{-1}B_{11} & -\mathcal{M}^{-1}B_{12} & \mathcal{M}_{11}^{-1} & \mathcal{M}_{12}^{-1} & 0 & 0 & -\mathcal{M}^{-1}\mathcal{C}_1 & 0 & 0 \\ 0 & 0 & -\mathcal{M}^{-1}B_{21} & -\mathcal{M}^{-1}B_{22} & \mathcal{M}_{21}^{-1} & \mathcal{M}_{22}^{-1} & 0 & 0 & -\mathcal{M}^{-1}\mathcal{C}_2 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1/\tau & 0 & 0 & 0 & 0 & 1/dt & 0 \\ 0 & 0 & 0 & 0 & 0 & -1/\tau & 0 & 0 & 0 & 0 & 1/dt \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{4.7}$$

$\mathbf{T_{e,pert}}$ and $\mathbf{T_{s,pert}}$ being the external perturbations on the elbow and the shoulder.
The new elements $1/dt$ of the matrix A'' are present because as explained in the introduction, the matrix A'' is multiplied by a factor dt because of the euler explicit integration. Thanks to the factors $1/dt$ the elements $\mathbf{T_{s,pert}}$ and $\mathbf{T_{e,pert}}$ have the same unit as the elements $\mathbf{T_s}$ and $\mathbf{T_e}$ (i.e. Newton meter).

4.2 Wide target

So far, the only experiment that we can simulate with the controllers presented, is asking a person to do a movement with its arm to reach a small circular target. In order to

reproduce the experiments presented in chapter 2, we need to find a way to express a wide target in a quadratic cost function.

In order to simulate that shape of target with the LQG_t controller, we need to change the expression of the final cost function of the equation 3.4 into another one that will not penalize the distance between two points but rather between a point and an axis.
The expression of that cost function is:

$$\begin{aligned} Cost_{wide}(\theta, \dot{\theta}, u) = & C_{pos} \left[(l_1 sin(\theta_s) + l_2 sin(\theta_s + \theta_e)) - (l_1 sin(\theta_s^*) + l_2 sin(\theta_s^* + \theta_e^*)) \right]^2 \\ & + C_{vel} \left[\dot{\theta_s^2} + \dot{\theta_e^2} \right] \\ & + C_{command} \left[\sum_{k=1}^{N-1} u_{k,s}^2 + u_{k,e}^2 \right] \end{aligned} \tag{4.8}$$

where C_{pos}, C_{vel} and $C_{command}$ denotes respectively the weights of the penalization of the position, speed and input command.

As we have seen in chapter 3, the LQG controller requires a quadratic cost function of the following form:

$$J(x, u) = x_N^T Q_N x_N + \sum_{k=1}^{N-1} \left(x_k^T Q_k x_k + u_k^T R u_k \right) \tag{4.9}$$

Unfortunately, one can quickly see that equation 4.8 cannot be written in the quadratic form above because of the term multiplied by C_{pos}. The idea to still be able to use it with the LQG controller is to perform a quadratic taylor approximation of that term. Notice that even though θ_s^* and θ_e^* are contained in the state vector, they are both constants. Thus, we can just use the quadratic taylor approximation for a function of two variables which are θ_s and θ_e in our case.
The quadratic taylor approximation around a point $(\theta_s = \alpha, \theta_e = \beta)$ for such a function is the following [23]:

$$\begin{aligned} f(\theta_s, \theta_e) \approx h(\theta_s, \theta_e) = & f(\alpha, \beta) + f_{\theta_s}(\alpha, \beta)(\theta_s - \alpha) + f_{\theta_e}(\alpha, \beta)(\theta_e - \beta) \\ & + \frac{f_{\theta_s\theta_s}(\alpha, \beta)}{2}(\theta_s - \alpha)^2 \\ & + f_{\theta_s\theta_e}(\alpha, \beta)(\theta_s - \alpha)(\theta_e - \beta) \\ & + \frac{f_{\theta_e\theta_e}(\alpha, \beta)}{2}(\theta_e - \beta)^2 \end{aligned} \tag{4.10}$$

where f_{θ_s} and $f_{\theta_s\theta_s}$ denote the first-order and the second-order partial derivative of f with respect to θ_s and similarly for f_{θ_e}, $f_{\theta_s\theta_e}$ and $f_{\theta_e\theta_e}$.
Since we want to minimize that new function $h(\theta_s, \theta_e)$, we are only interested in the non-constant part of that function:

$$\begin{aligned} g(\theta_s, \theta_e) = & \theta_s \left[f_{\theta_s}(\alpha, \beta) - \alpha f_{\theta_s\theta_s}(\alpha, \beta) - \beta f_{\theta_s\theta_e}(\alpha, \beta) \right] \\ & + \theta_e \left[f_{\theta_e}(\alpha, \beta) - \alpha f_{\theta_s\theta_e}(\alpha, \beta) - \beta f_{\theta_e\theta_e}(\alpha, \beta) \right] \\ & + \theta_s^2 \frac{f_{\theta_s\theta_s}(\alpha, \beta)}{2} + \theta_e^2 \frac{f_{\theta_e\theta_e}(\alpha, \beta)}{2} + \theta_s \theta_e f_{\theta_s\theta_e}(\alpha, \beta) \end{aligned} \tag{4.11}$$

Now, thanks to the constant equal to 1 that we have added previously to the state vector, we can formulate this last function $g(\theta_s, \theta_e)$ in the form $x^T Q x$ even if that function contains linear terms.
For the iLQR controller, it is not required to work with such a restrictive formulation of the cost function, so those steps are not necessary. We just have to compute the gradient and the hessian of the cost function of the equation 4.8. It is important to highlight that even though the terms θ_s^* and θ_e^* are also in the state vector, it is useless to compute the partial derivatives of the cost function with respect to them because their values cannot be impacted by the values of the input of the system. Thus those terms will not have any impact on the control law computed by the iLQR.

Chapter 5

Results

Now that the details of the controllers and their implementations have been explained, this chapter explores how those controllers behave in different situations.
Five aspects of those controllers are explored in five distinct sections:

- The first section compares the performances of the iLQR algorithm when it is combined with the LQG controller or the minimization jerk trajectory generation to get the initial control sequence U.
- The second section is dedicated to the choice of hyperparameters. For the LQG_t controller, we only investigate the impact of the frequency of update of the linearization. For the iLQR, we explore two different hyperparameters: λ_{start} and λ_{rate}.
- The third section highlights the impact of introducing a delay such as the one described in chapter 3.
- The fourth section investigates the impact of the noise by making simulation of the same movement with various levels of noises.
- The fifth section contains simulations of movements to imitate the two experiments described in chapter 2.

In order to represent the performances of the controllers and their behaviors, three different kinds of graphs are used in this chapter:

- The first one is the box plot. It is composed of a box that represents the three quartiles, vertical lines called the whiskers and at the end of those whiskers, there are horizontal lines called caps that represent the maximum and minimum values.
- The second type of plots shows the paths of the simulated movements on a plan. On those graphs, the target is either a black circle in the case of a small target or a gray rectangle in the case of a wide target. The starting point is situated on the bottom of the figure and the target is on the top.

- Finally, the third kind of graphs shows the input commands in Newton meter [Nm] of the system that the controller computes. The lines show the average inputs of all the simulations and the colored regions around those lines represent the standard deviations of the signals.

Each situation described in this chapter is simulated twenty times in order to observe the stochastic influence over the system.
Note that for simplicity, we will refer to the value of the cost function for a trajectory performed by a controller in certain conditions as the cost of that controller in that conditions.

5.1 Initial control sequence for iLQR

We have seen in chapter 3 that the algorithm of the iLQR needed an initial control sequence U and an initial state in order to work. But it has not been said yet how those things were obtained. In this section, two candidates for this are compared. The first one is the minimum jerk trajectory generator and the second one is the LQG controller with a linearization around the target point.

The minimum jerk trajectory generator provides an initial state sequence X, which combined with the inverse dynamics 3.9 of the system, can be used to deduce a corresponding control sequence U.
On the other hand, the LQG allows to directly get an initial control sequence U and an initial state sequence X without using an inverse model.

The figure 5.1 below illustrates the costs of the movements performed with the iLQR with the two different candidates.

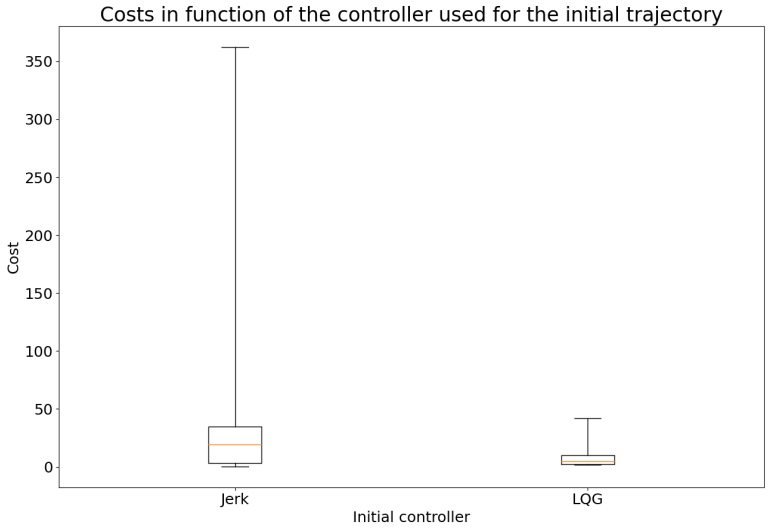

Figure 5.1: Costs of the iLQR in function of the controller used to find the initial control sequence when no perturbation is applied

It is already clear in that figure that using the LQG to find an initial trajectory works better than using the minimum jerk trajectory generator.

The choice of the hyperparameters used for iLQR combined with the minimum jerk controller will not be discussed since this combination will not be used but the method used was the same as the one described in the next section, the only difference being that the initial control sequence was computed with the minimum jerk controller instead of the LQG.

5.2 Hyperparameters

This second section is composed of two subsections, each one exploring the choice of hyperparameters of either the LQG$_t$ or the iLQR.

5.2.1 iLQR

In this subsection, we explore the choices of the two hyperparameters that are λ_{ratio} and λ_{start}. To do so, we simulate twenty times a movement in straight line without any mechanical perturbation and with both noises having the non-zero elements of their covariance matrices equal to 10^{-3}.

Since we want to explore the impact of two hyperparameters, we should make them vary both at the same time and see which combination works the best. But because the simulation of movements controlled by the iLQR is quite long, it is not feasible and it would require to simulate too many movements.
The idea here is therefore to find the best values for those hyperparameters by making them vary one at the time.

First, we begin by making the value of λ_{ratio} vary as illustrated in figure 5.2. We see in that figure that there is no clear convergence towards an ideal value.

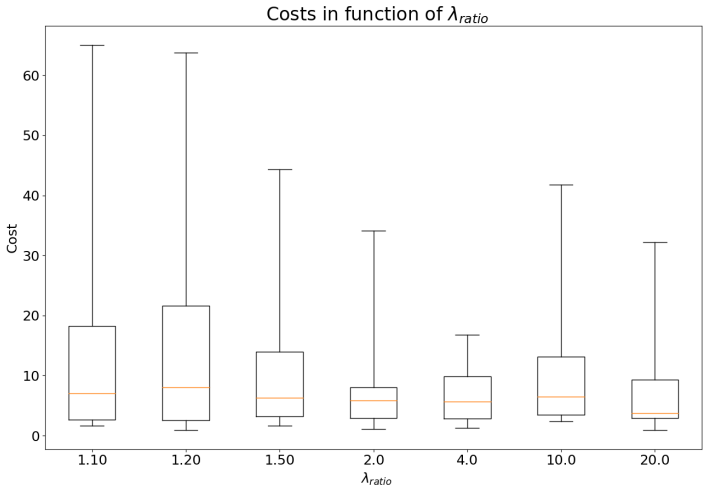

Figure 5.2: Costs of the iLQR for simulations performed with various values of λ_{ratio}

The value of 4 seems to be the best one in this situation because its highest cost is the lowest for all the values of λ_{ratio} tested. Also, the corresponding quartiles are low even though the ones computed for λ_{ratio} equal 2 are lower. From now on, the value of λ_{ratio} will always be set to 4 in the future simulations.

Now that we have fixed a value for λ_{ratio}, we want to see for which value of λ_{start} the simulated costs are the lowest. The figure 5.3 shows that again there is no convergence to a certain value but we see that one of the best average convergence is achieved with λ_{start} equal 2, thus we will make the simulations with that value.

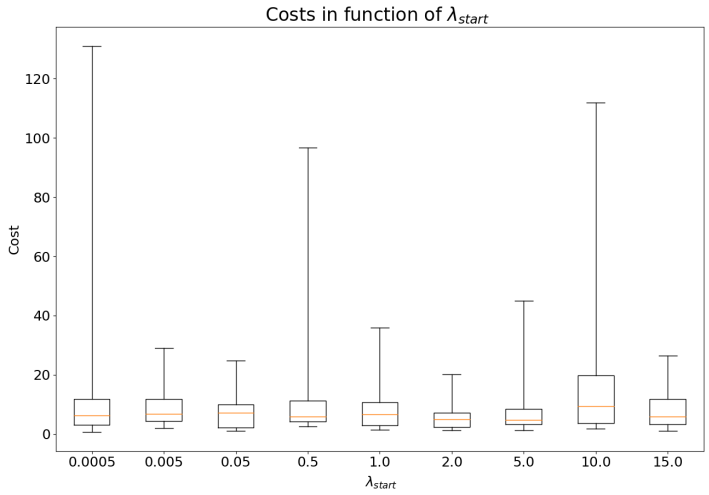

Figure 5.3: Costs of the iLQR for simulations performed with various values of λ_{start}

5.2.2 LQG$_t$

In order to explore the impact of the frequency of update of the linearization for the LQG$_t$ controller, we use two different metrics: the cost of the movement and the final distance between the hand and the target.
The cost of a movement is a weighted sum composed of the final distance between the hand and the target, the final hand's speed and the sum of the input commands squared. This is the quantity that the controller tries to minimize.

The figure 5.4 shows the quartiles and the extreme values of the costs of a movement simulated 20 times in 3 different situations. The plot on the left corresponds to a movement done with no external perturbation, the movements represented in the middle are subjected to a 1.5Nm perturbation on the shoulder and the plot on the right is done with movements that are also subject to a 1.5Nm perturbation but on the elbow.
The value of 143Hz corresponds to an update of the linearization at each step of the simulation since the duration of the movement is 0.7 second and that there are 100 iterations.

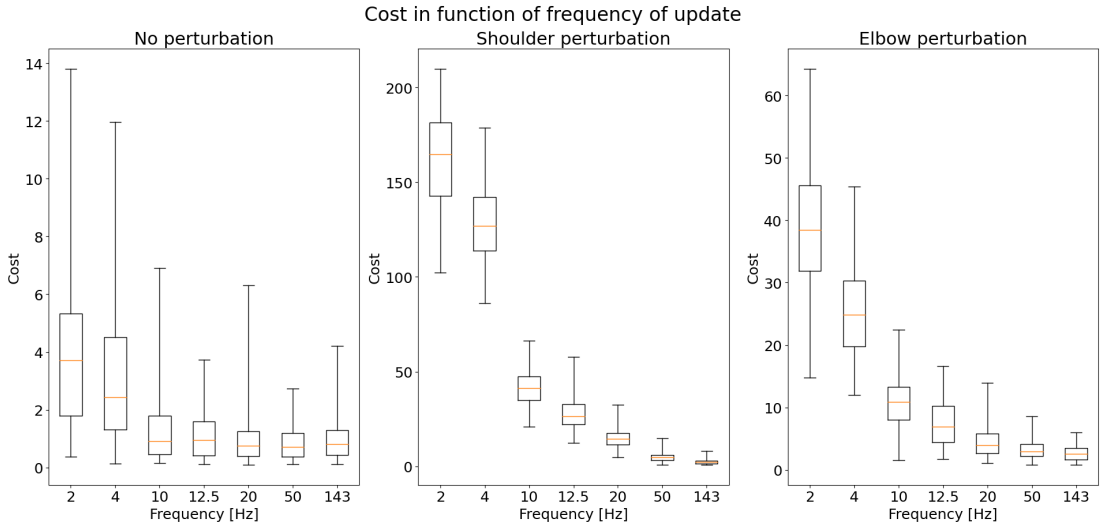

Figure 5.4: Costs of the LQG$_t$ controller in function of the frequency of linearization update with mechanical perturbations of 1.5Nm

We can see in that figure that when there is no perturbation, having a high rate of update does not decrease that much the cost of the movement, but as soon as we start applying a perturbation on the arm and especially on the shoulder, the difference in cost between a high rate and a low rate explodes.

The figure 5.5 shows the means and the standard deviations of the final distances between the hand and the target of 20 movements simulated with the same disturbances as the previous plot.

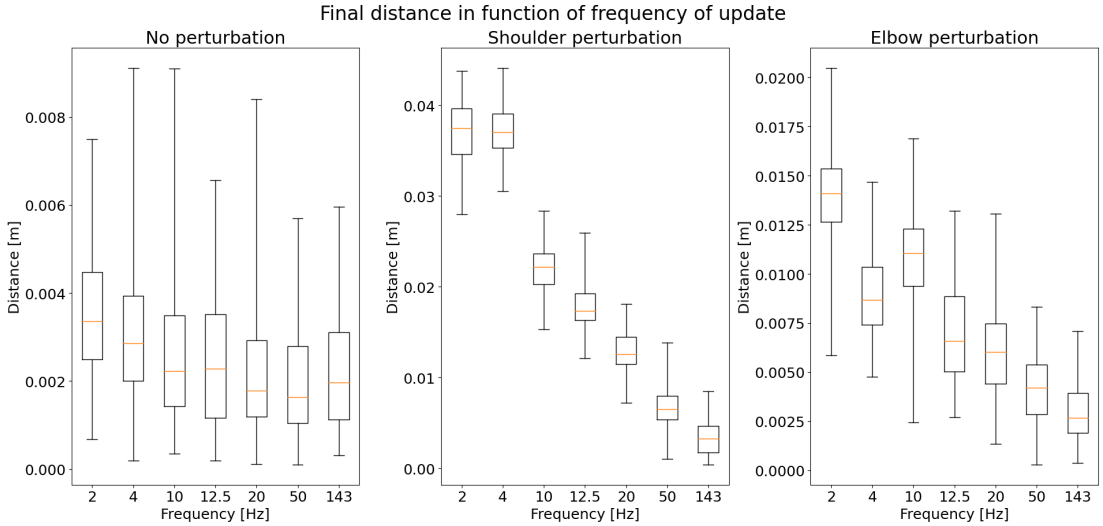

Figure 5.5: Final distance between the target and the hand's position in function of the frequency of linearization update with perturbations of 1.5Nm

We can see that there is also a correlation between the final distance and the update rate. This is not really surprising since the coefficient multiplying the final distance in the cost function is much bigger than the two others. This is because the first purpose of the reaching movement is to attain the target.
Nevertheless, we can still observe some differences between the cost and the final distance. For example, we see that when there is a perturbation on the elbow, the cost always decreases when the frequency increases but the distance increases between 4 to 10 Hz.

5.3 Delay

The way to introduce some delay in the system was presented in chapter 3. In this section, we observe what impact inducing such a delay has on the LQG$_t$ by simulating movements with a perturbation on the upper arm to highlight the delay in the reaction.

The figure 5.6 shows how the LQG$_t$ is performing when a perturbation of 1.5Nm on the upp

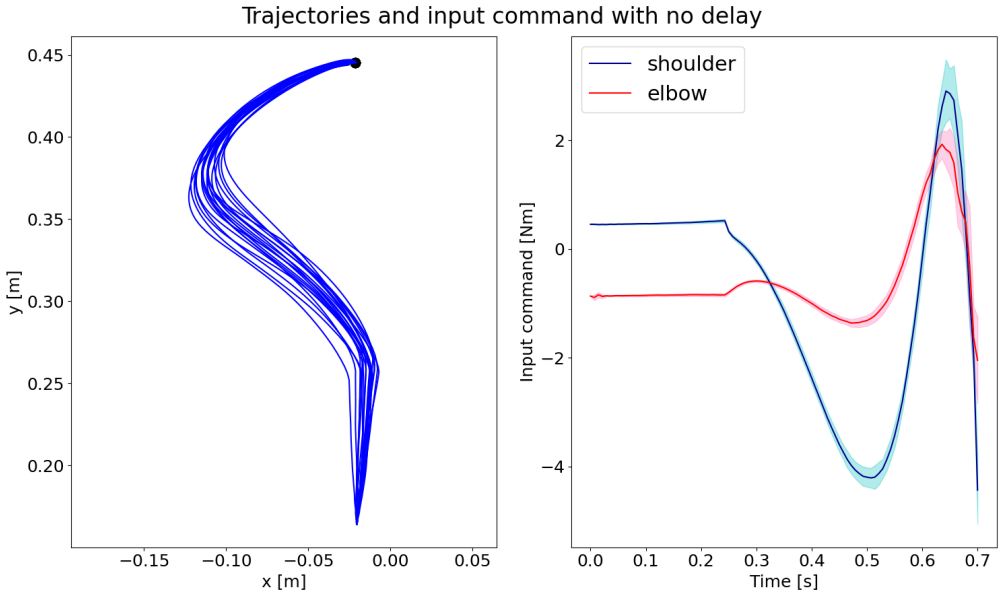

Figure 5.6: Trajectories and input of the system with no delay and a perturbation of 1.5 Nm on the upper arm

On the other hand, the figure 5.7 illustrates the trajectories and the input signals when a delay of 60ms is introduced in the model. We observe that when there is a delay, the trajectories are deviated a little bit more on the left and that the final position is not on the target unlike when there is no delay.
The observed behavior is quite logical, the control is a little bit less precise and the time to correct a perturbation is a little bit longer but apart from that, the control seems to work exactly the same way.

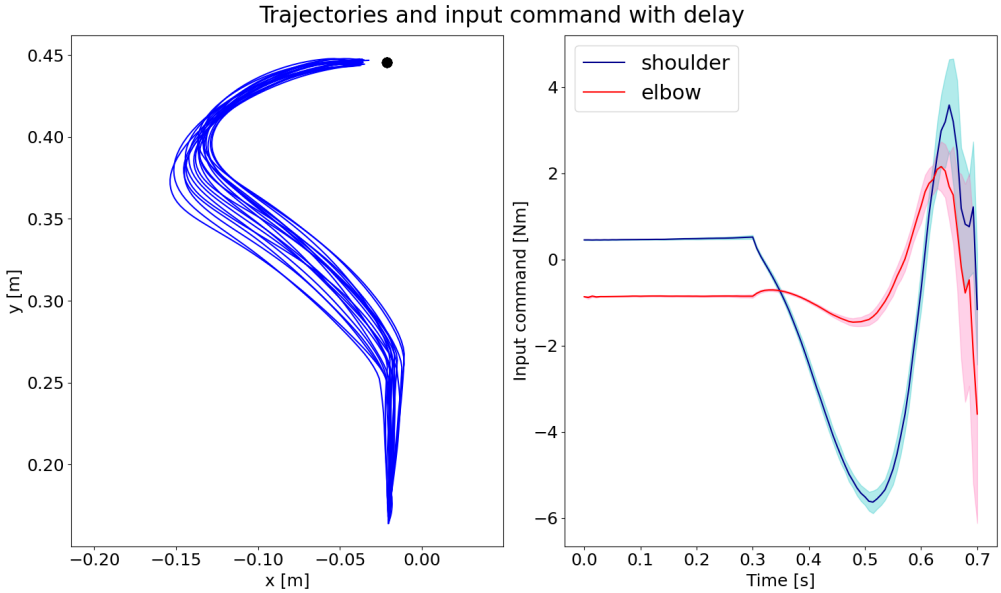

Figure 5.7: Trajectories and input of the system with 60ms delay and a perturbation of 1.5 Nm on the upper arm

Unfortunately, such a delay was not implemented for the iLQR because it required to modify a lot of matrices and because of that all the simulations in the following sections will be realized without delay so that the results of the iLQR and the LQG$_t$ can be compared.

5.4 Noisy system

Now that we have investigated the choice of hyperparameters, we are going to perform simulations of movements with both controllers with various levels of process and observation noises. For the sake of simplicity, we will say that the value of a noise is changed when the values inside its covariance matrix are changed.
The value used as standard for both noises is 10^{-3}. This means that the non-zero values of the covariance matrices are all equal to that value. When it is said that the process or the observation noise is changed, it means that they take the value 10^{-2}.

5.4.1 LQG$_t$

The figure 5.8 shows the quartiles and extreme values of the costs of movements performed with different levels of noises. From left to right, the bars represent trajectories with: standard values of noises, higher process noise and higher observation noise. It is clear that the observation noise has the bigger impact on the cost.

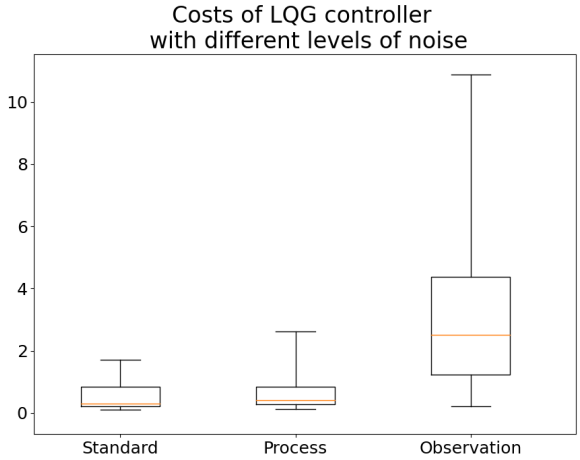

Figure 5.8: Costs of the LQG$_t$ controller with different levels of noises

In figure 5.9 below, we see on the left the trajectories simulated with standard values of noises, and on the right the input commands sent by the brain to the shoulder and the elbow during the movement.
This input signal is quite similar for all the 20 movements simulated in the beginning because the transparent surfaces do not expand that far from the mean line but after 0.4 seconds the signals start to be much noisier and some oscillations appear.
We will see that this behavior is very common and can be found in almost all the performed simulations. This is in fact due to the need of stopping the hand on the target and stabilizing it.

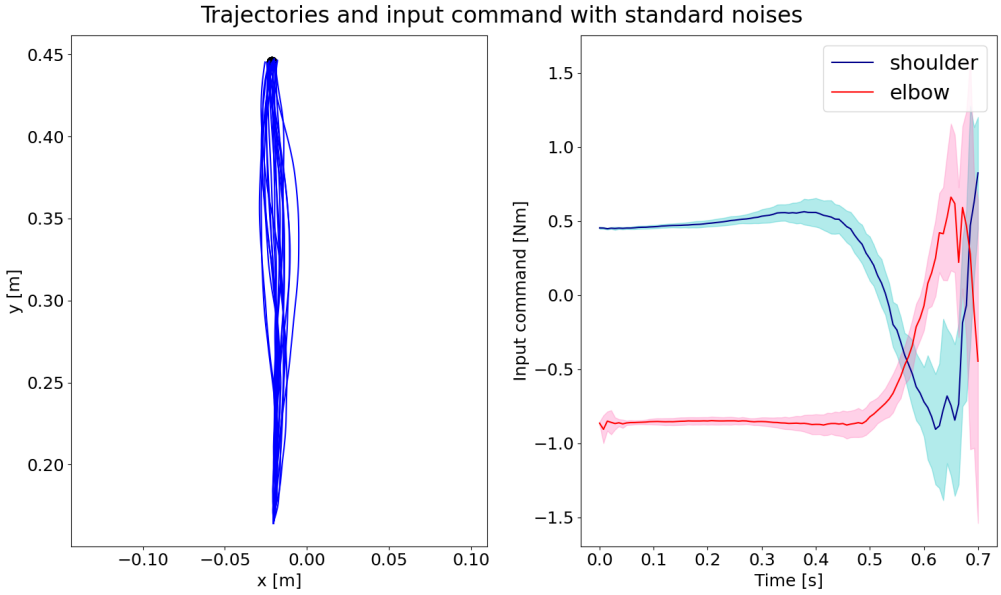

Figure 5.9: Trajectories and input of the system with standard values of noises with LQG$_t$

Now that we have seen the behavior defined as standard, we can compare it to other

results obtained with different values of noises.
The first one that we will compare is represented in figure 5.10. The simulations in that figure are done with a higher value of process noise. Even though the trajectories are much more spread along the x-axis than in the previous figure, they still converge very well towards the target point.
Also, the input commands are even noisier than in the standard case after 0.4 seconds. Finally, the oscillations of those signals are bigger than previously.
Those results are consistent with what is observed in the trajectories of the hand. The hand deviates more from the straight path to the target than before and in a random way. The corrective responses to get back to the target are thus more random and also bigger.

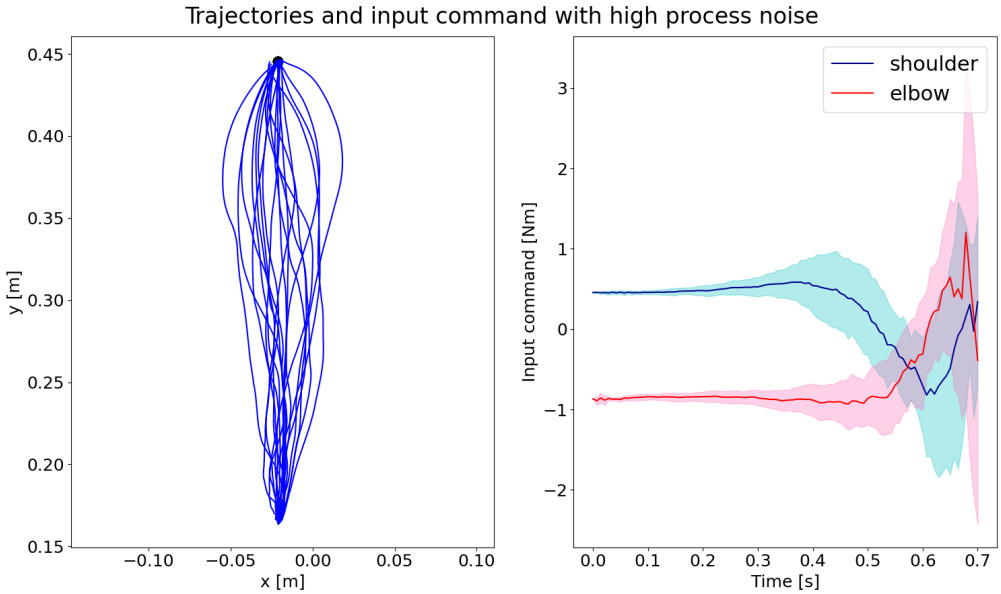

Figure 5.10: Trajectories and input of the system with high value of process noise with LQG_t

In figure 5.11, we can see the impact of the observation noise on the movements. This time the trajectories are not spread anymore along the x-axis but they do not reach the target as well as before.
Regarding the command inputs, they are a little bit less noisy than in the previous figure but the amplitude of the oscillations at the end of the movement is similar to the one observed previously.
Again, those results make sense because the trajectories are not that much spread anymore, thus the input signals are more similar. But at the end, the oscillations are still observed in order to stabilize the hand on the target.

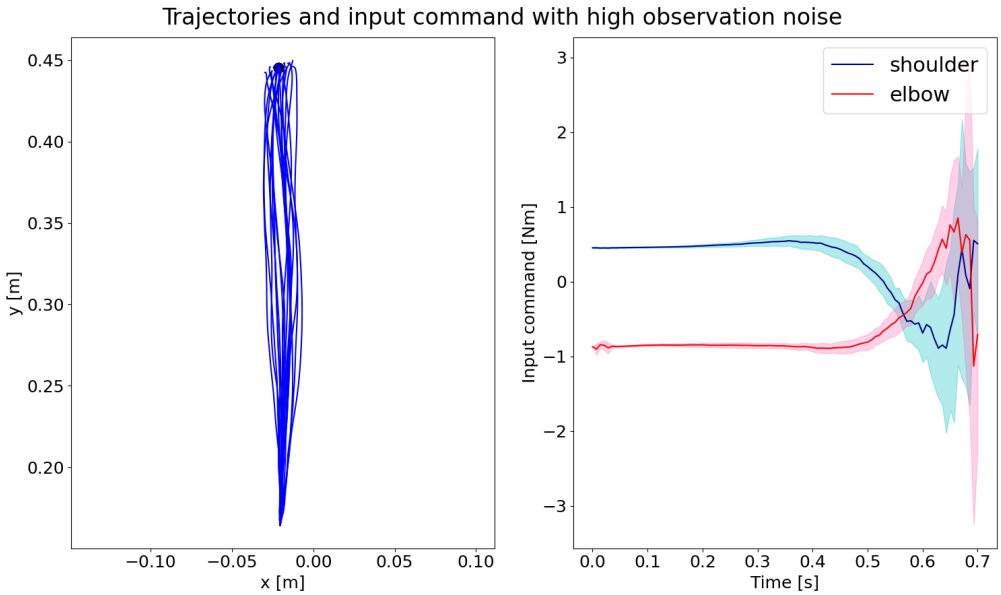

Figure 5.11: Trajectories and input of the system with high value of observation noise with LQG$_t$

5.4.2 iLQR

Now that we have seen the impact of a noisy system on the LQG$_t$ controller, we will see how the iLQR reacts to similar conditions.
The figure 5.12 is a box plot of the costs of the trajectories simulated with the iLQR with different levels of noises. This time, the impacts of the two noises seem to be more similar even though the 3 quartiles for the observation noise are slightly bigger than the ones for the process noise.
However, as we will see on the figure 5.14, some simulations did not converge towards the target for the high value of process noise. The trajectories of those outliers are in red on that same figure and the costs of those trajectories were not taken into account to make the box plot of the figure 5.15. Thus the process noise has an impact on the convergence of the algorithm.

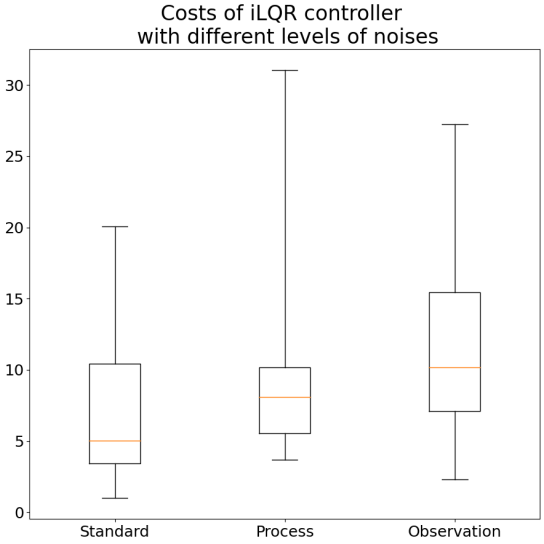

Figure 5.12: Costs of the iLQR with different levels of noises

The figure 5.13 provides an overview of how well the iLQR is doing in a standard situation. Most of the trajectories reach well the target but there are already some of them that do not. The input signals are really not very noisy, this is the result of the convergence by the algorithm to a certain trajectory. On the other hand, there are still oscillations at the end of the movement and they are much bigger than the ones observed with the LQG$_t$.

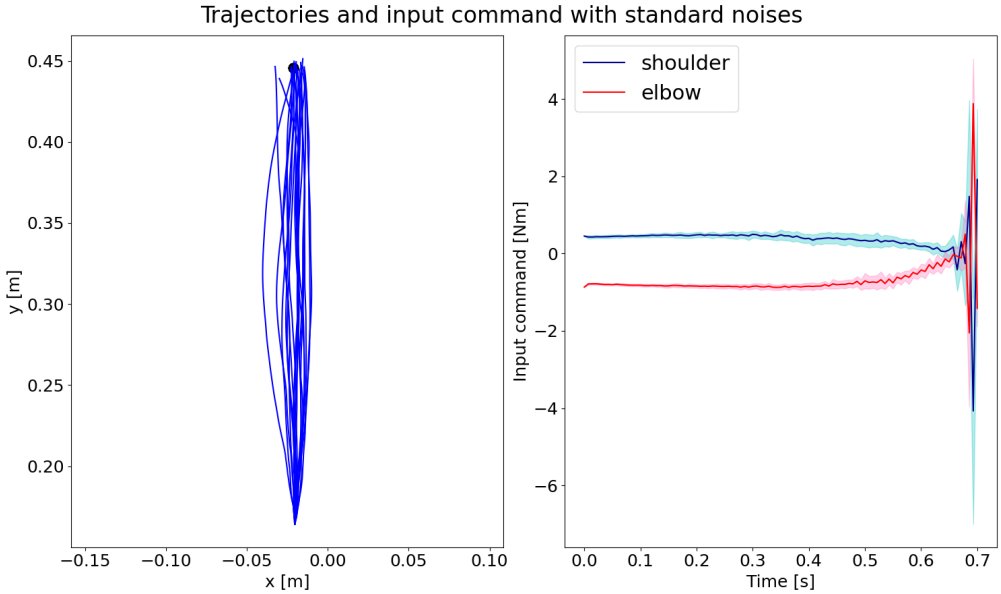

Figure 5.13: Trajectories and input of the system with standard values of noises with iLQR

The figure 5.14 shows the impact of a higher value of process noise on the iLQR. First, we can see that as for the LQG$_t$, the trajectories are more spread along the x-axis when the process noise is higher. And again like for the LQG$_t$, most of the trajectories are still reaching quite well the target except for some trajectories in red.

As explained above, the red lines on the left of the image correspond to the trajectories that were classified as outliers. This classification is purely based on a qualitative approach that combine the path of the trajectory and its final cost. Indeed, 16 trajectories out of 20 reach the target in a very similar way but for some reason, 4 of them behave completely differently. In fact, we will observe several times this behavior with the iLQR where some trajectories seem to reach for another target point. The reason of that lack of accuracy will be explained in the next chapter.
The input signals are not very noisy except at the end of the movement where we observe big oscillations like in the standard case.

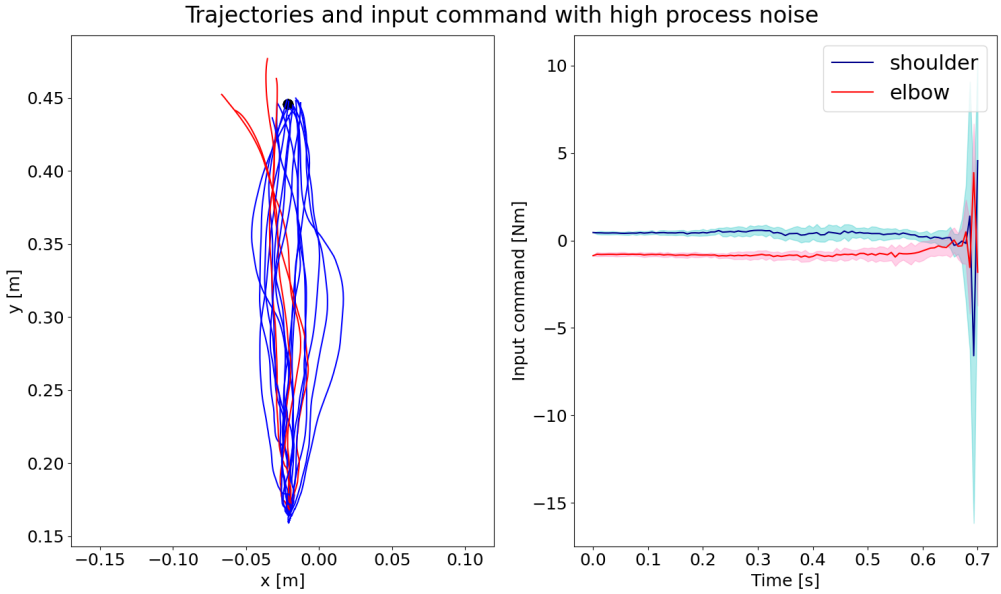

Figure 5.14: Trajectories and input of the system with high value of process noise with iLQR

The impact of the observation noise is illustrated in figure 5.15. It seems that the trajectories are less spread around the x-axis than when the process noise was high, but most of them don't reach the target as close.
Regarding the input commands, they are very similar to the ones of the standard case.

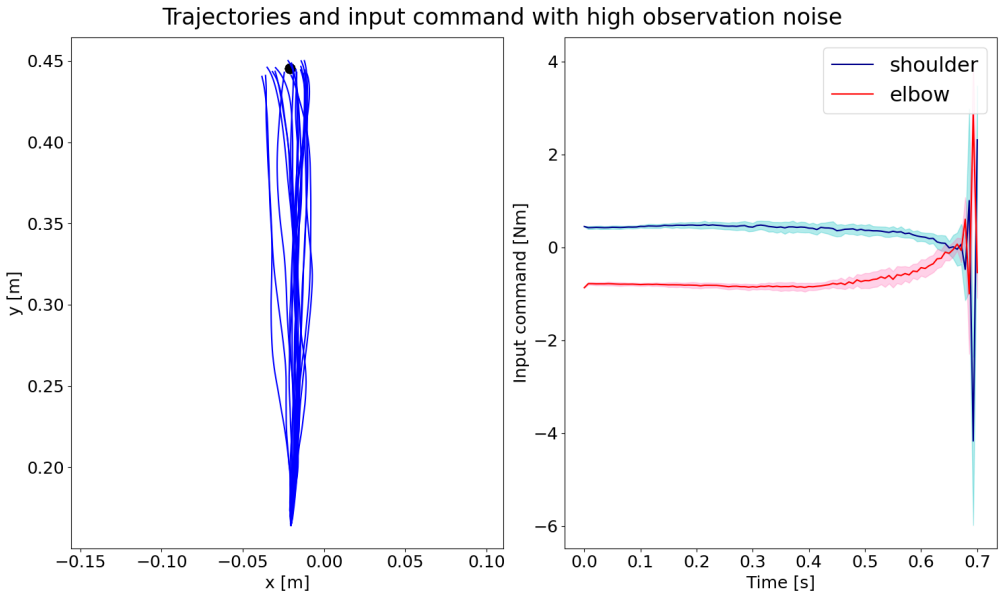

Figure 5.15: Trajectories and input of the system with high value of observation noise with iLQR

All these figures show that the observation noise seems to have the bigger impact on the costs of the trajectories but sometimes, the process noise can cause the iLQR to not converge towards the target at all.

5.5 Perturbed movement

Now that we have analyzed the impact that the noises have on the two controllers, we will do the same thing but with mechanical perturbations. As in the previous subsection, the trajectories will be simulated for the two different controllers, but this time for both small and wide targets.
The perturbations have a magnitude of 1.5Nm and are either applied on the lower arm or the upper arm. For the sake of simplicity, the perturbations applied on the lower arm will be denoted as perturbations on the elbow and the ones applied on the upper arm as perturbations on the shoulder.
Notice that in the case of the iLQR, the perturbation is not applied during the iterations of the algorithm but only when the algorithm has converged and that the trajectory is simulated with the obtained feedback gains.

5.5.1 Target point

We begin by investigating the impact of the perturbations on both controllers when a small target is reached.

$\mathbf{LQG}_t$

The figure 5.16 shows that the mechanical perturbations do not have much impact on the trajectories cost. Indeed, even though the values of the costs do not mean anything taken alone, the median costs when there is a perturbation are lower than the maximum cost simulated without one.

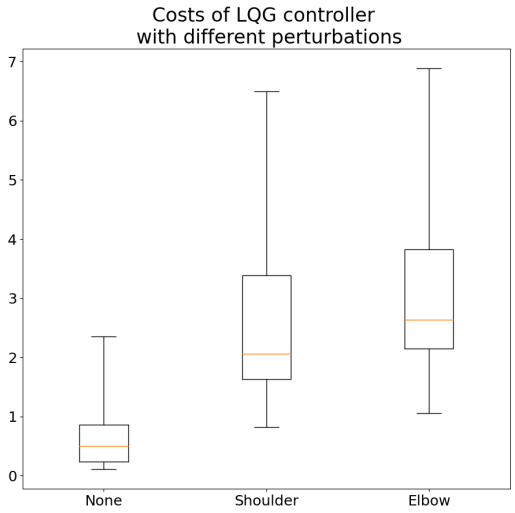

Figure 5.16: Costs of the LQG_t with different perturbations

The figure 5.17 gives a reminder of how the controller behaves in standard conditions and is just there to ease the comparison with the other graphs of this section. Indeed, the trajectories in figure 5.9 were simulated using the same parameters.

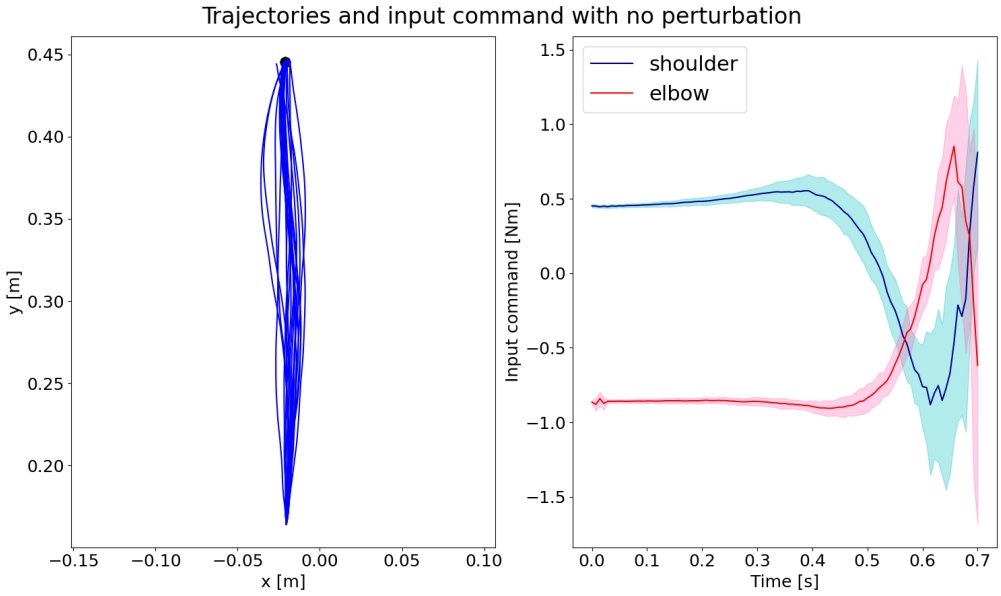

Figure 5.17: Trajectories and input of the system with no perturbation with LQG_t

The trajectories in figure 5.18 show that a perturbation on the elbow is well corrected and that the hand is still capable of reaching the target. Moreover, the input signals are much less noisy than in the standard case. This is probably because the reaction needed to counter the perturbation is roughly the same for every trajectory.
The signals also oscillate less than before at the end of the movement but the amplitude is greater.
Finally, the responses in the input signal to the perturbation are very clear, especially for the signal of the elbow. Interestingly, we observe that even though there is a step in the input signal corresponding to the perturbation, the trajectory is not instantly corrected and makes a circular arc. This is due to the time constant of the equation 3.8 which makes the contraction of the muscle progressive even though the input signal presents a step.

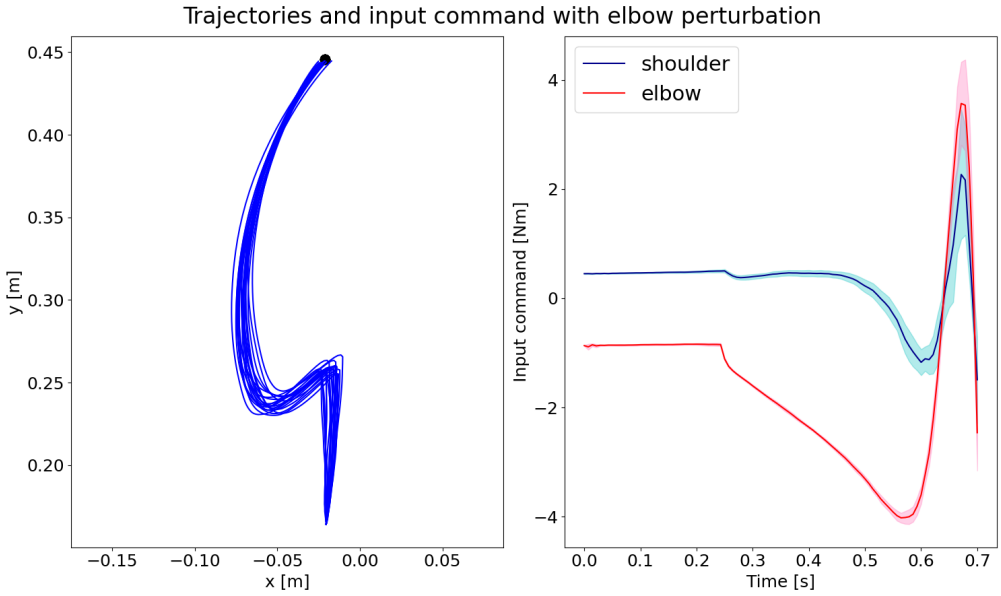

Figure 5.18: Trajectories and input of the system with a perturbation on the elbow with LQG_t

The figure 5.19 presents a lot of similarities with the previous figure. This time, the perturbation is on the shoulder but the trajectories still reach the target very well. The input signals are again much less noisy than when there is no disturbance.
In contrast, this time it is more the signal corresponding to the shoulder that has the biggest oscillations where in the previous case it was more the one corresponding to the elbow. This makes sense since the perturbation is not longer applied to the same part of the arm.

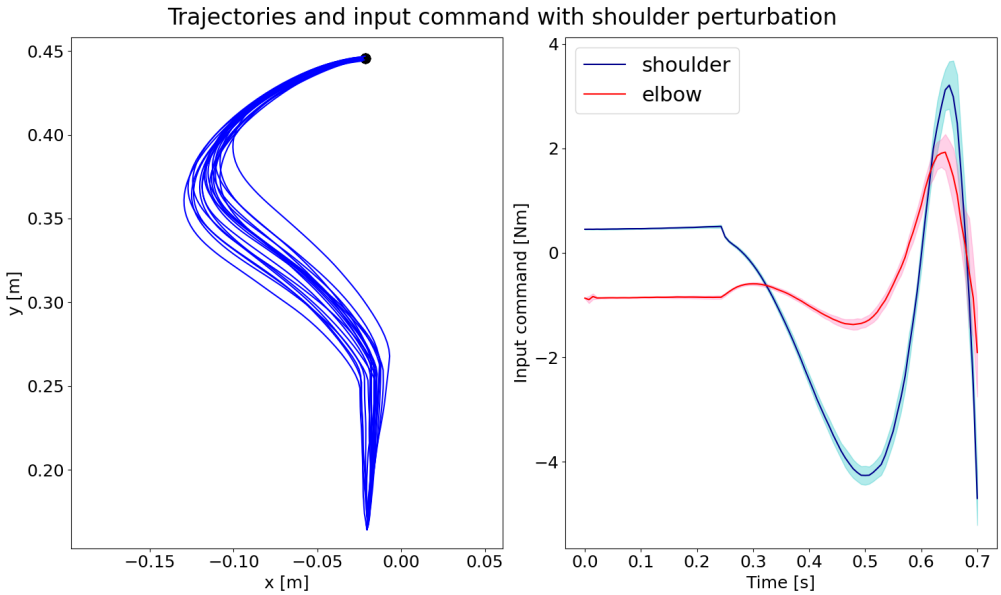

Figure 5.19: Trajectories and input of the system with a perturbation on the shoulder with LQG_t

iLQR

Now, we are interested in how the iLQR is dealing with disturbances.
In figure 5.20, one can see by looking at the scale of the y-axis that this controller is not able to adapt as well as the LQG_t does. Indeed, the median costs are very high, particularly the one corresponding to the shoulder perturbation. We can also see on the figures 5.22 and 5.23 that some of the trajectories are in red which means that they do not converge towards the target and therefore are classified as outliers which are not taken into account to make this box plot.

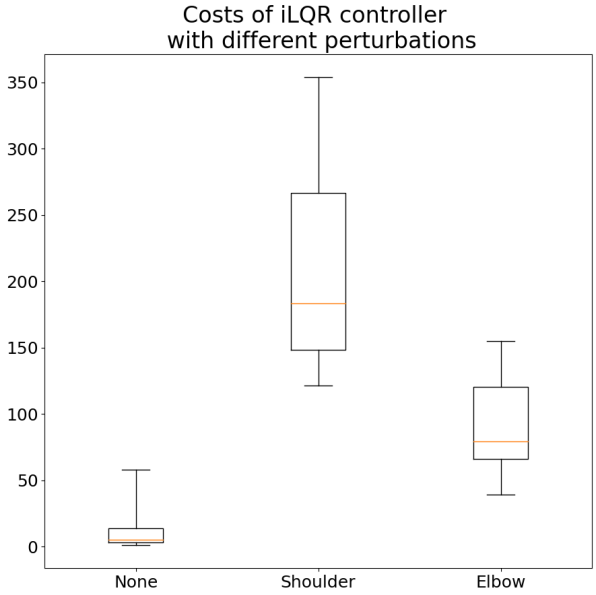

Figure 5.20: Costs of the iLQR with different perturbations

In figure 5.21, one can see that the trajectories reach quite well the target except for a few ones that did not converge well and that explains why the maximum value of cost, when there is no perturbation, is much higher than the third quartile.

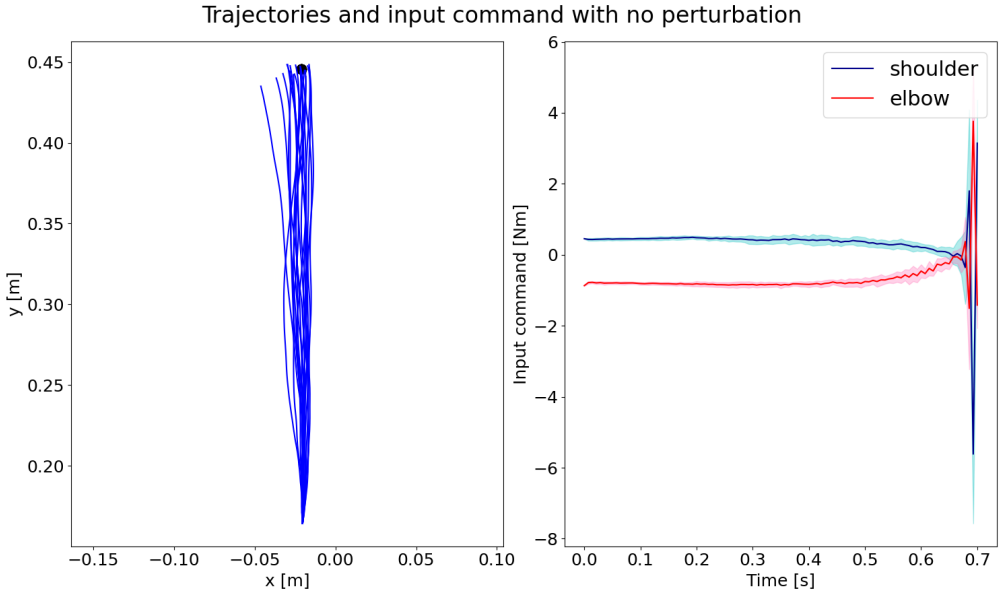

Figure 5.21: Trajectories and input of the system with no perturbation with iLQR

As shown in figure 5.22, the iLQR is never capable of correcting the trajectories back to the target when an elbow perturbation is applied, nevertheless most of them converge to a point that is at approximately 2 centimeters from the target which is quite close

considering the amplitude of the movement. This lack of precision has already been observed with the iLQR in the context of a high process noise and the trajectories that behaved like that were classified as outliers in that context. As mentioned at the time, the reason of that lack of precision will be given in the next chapter.
We also observe that on twenty simulations, four of them do not converge at all towards the target and just deviate completely after that the perturbation is applied. The possible explanations of the present of those outliers will be given in the next chapter.
The input signal corresponding to the elbow presents some oscillations with a very high frequency at the end of the movement and those oscillations seem to be common to all the trajectories because the red region around the line is not visible.

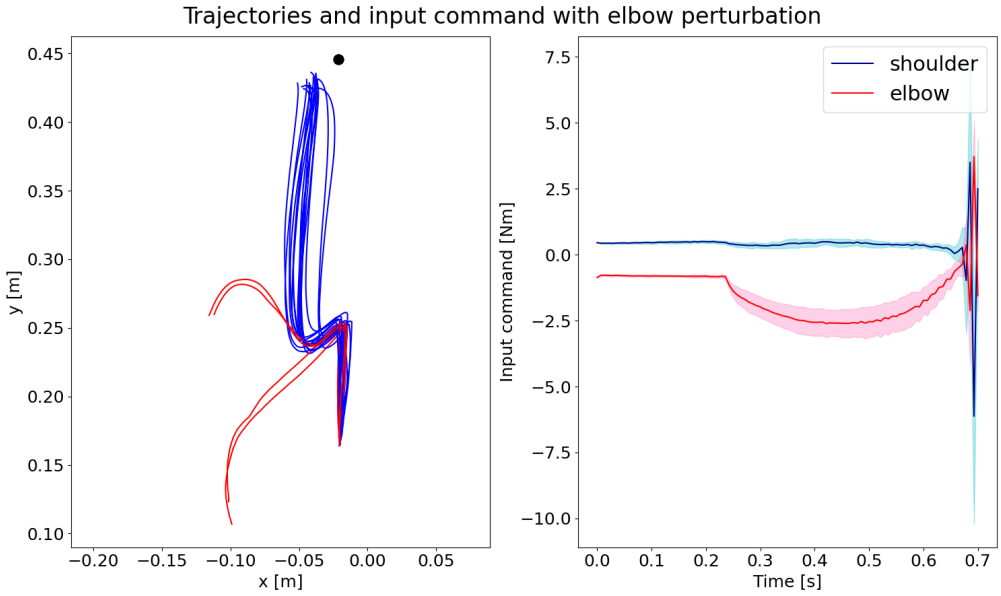

Figure 5.22: Trajectories and input of the system with a perturbation on the elbow with iLQR

On figure 5.23, we can see that again the trajectories do not reach the target as well as in the case when there is no perturbation but most of them reach for a point at approximately 5 centimeters on the left of the target point. Like in the case of an elbow perturbation, we observe four outliers in red that do not converge at all like the rest of the simulations.

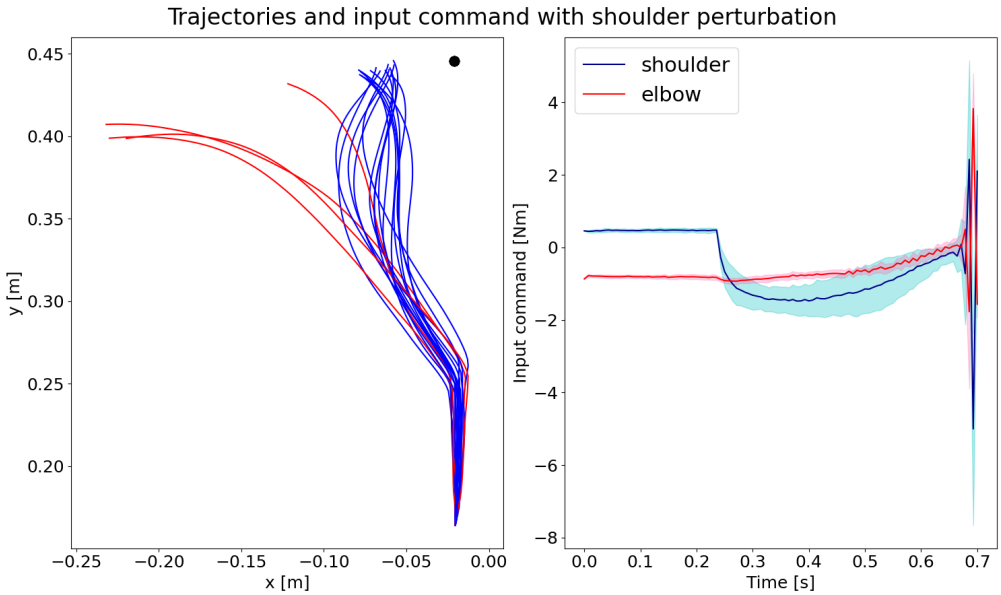

Figure 5.23: Trajectories and input of the system with a perturbation on the shoulder with iLQR

5.5.2 Target axis

Now, we will see if the controllers that we are using are reacting like people would do when they experience a mechanical perturbation while they reach for a wide target.

$\mathbf{LQG}_t$

The figure 5.24 shows that the LQG_t handles the perturbations on the shoulder way better than the ones on the elbow. Notice that comparing the values of the cost when reaching for a small or a wide target does not make much sense. The cost functions are indeed not the same and therefore being at the same distance from the target with the same final speed and the same value of penalization of input command would not result in the same cost.

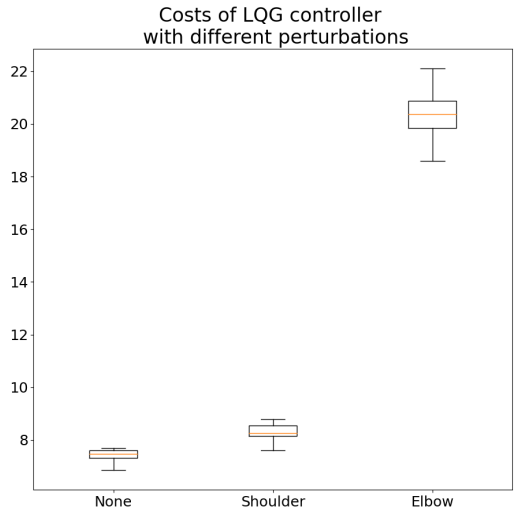

Figure 5.24: Costs of the LQG_t with different perturbations for a wide target

The figure 5.25 shows how the LQG_t is doing when reaching for a wide target without any disturbance. The trajectories are in a straight line and go to the nearest point of the axis. Regarding the input signals, one can observe that the signals are noisier around the minima and maxima of the oscillations.
Furthermore, we observe that those signals oscillate less at the end of the movement than when reaching for a small target. This is logical because there is one more degree of freedom to reach the target. Thus, less corrective responses are needed to stabilize the hand on the target.

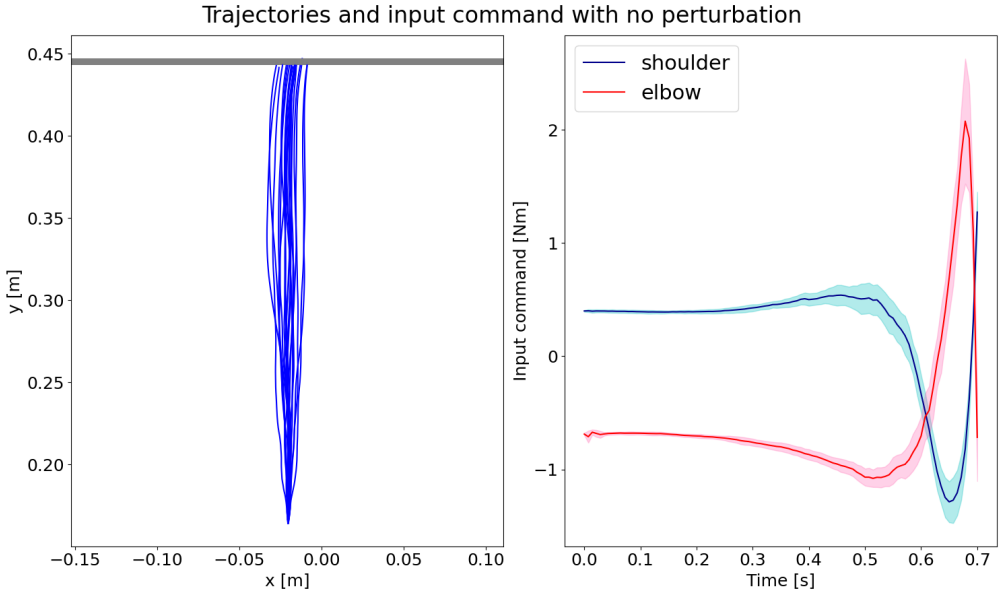

Figure 5.25: Trajectories and input of the system with no perturbation with LQG_t for a wide target

The figure 5.26 confirms the intuition the figure 5.24 gave, namely that the trajectories do not reach the target when a disturbance on the shoulder is applied. We see here that they converge towards the axis but they stop before reaching it. This is the first time that we observe such a behavior.
It seems that the disturbance on the elbow delays the arrival of the hand on the target. As the duration of the experiment is limited, the hand does not have enough time to reach the target and therefore stops before it does in order to avoid a penalty on the final speed. It appears that the input signals are even less noisy than in the standard case.

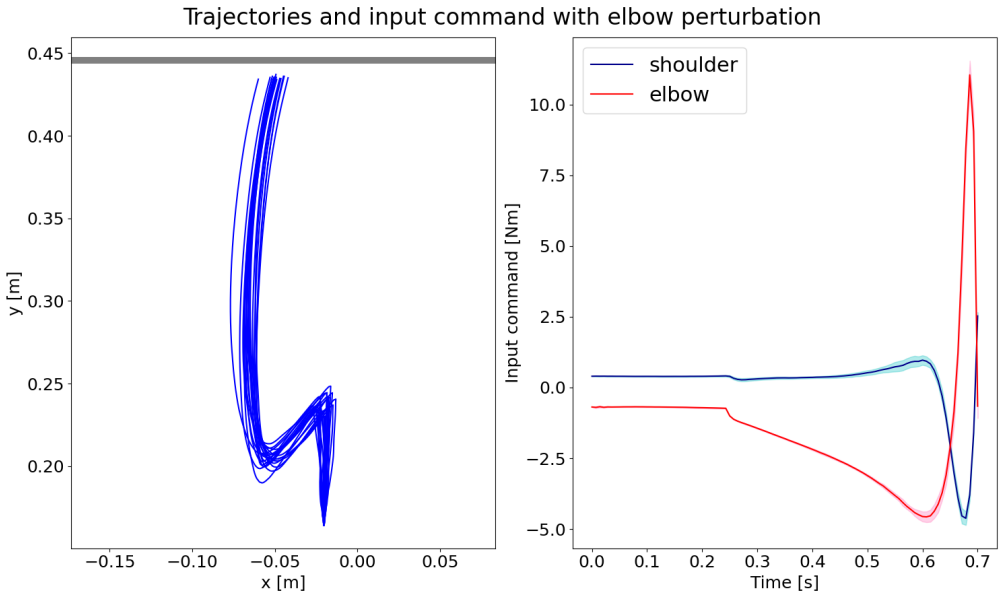

Figure 5.26: Trajectories and input of the system with a perturbation on the elbow with LQG_t for a wide target

The figure 5.27 shows that the LQG_t is doing very well with perturbations on the shoulder. We see that it behaves in a very similar way to the one observed in humans. Nevertheless, we see that at some point the trajectories are going straight forward to the target and stop their deviations and this is not what is observed with humans.

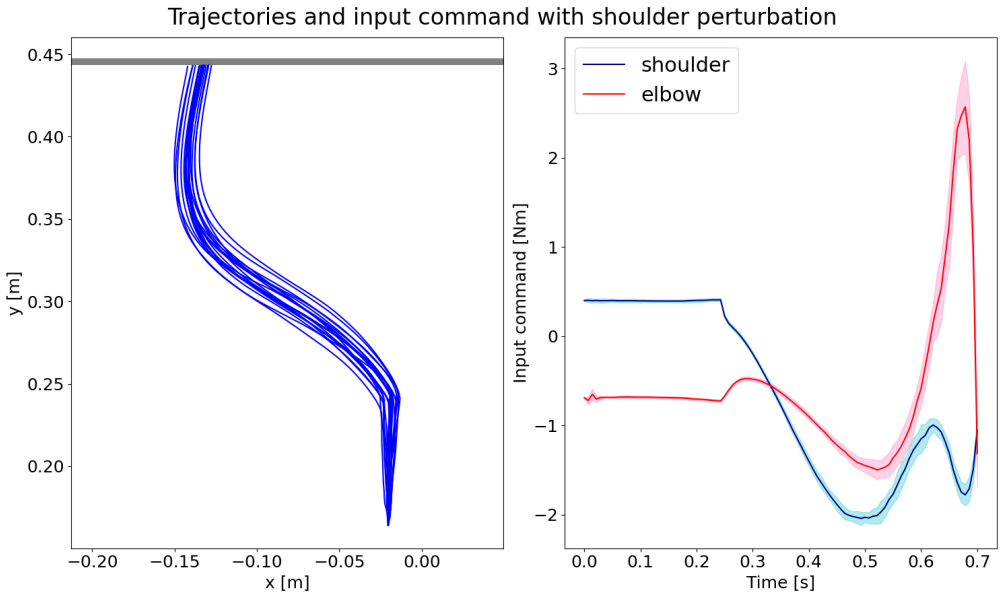

Figure 5.27: Trajectories and input of the system with a perturbation on the shoulder with LQG_t for a wide target

iLQR

By looking at the scale of the figure 5.28, one can already see that the iLQR is not doing as well as the LQG_t when facing mechanical disturbances. It seems that the iLQR is doing slightly better when facing shoulder perturbations. Moreover, if we look at the figure 5.30 corresponding to the elbow perturbation, we can quickly see that the majority of the simulations do not even converge towards the target when on the figure 5.31, we see that every trajectory converges pretty much the same way towards the target.

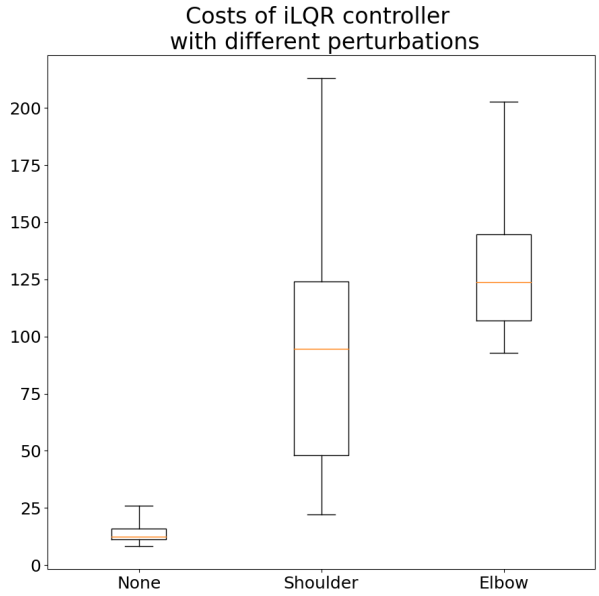

Figure 5.28: Costs of the iLQR with different perturbations with a wide target

The figure 5.29 is very surprising because we can see that the trajectories do not converge well to the target even though there is no perturbation applied yet. We also see that the trajectories are very spread and that the input signals are noisy at the end of the movement.

The reason of that lack of precision is not obvious but if we take a step back, we see that the LQG_t converges very well with a quadratic approximation of the same cost function and that the iLQR converged better when the target was small and thus when the cost function was different. Having said this, it seems that the problem would come from the capacity of the iLQR to use this new cost function and not a simple quadratic function. This is quite logical since the iLQR makes approximations of the cost function all along the path found when iterating but as soon as the simulated trajectories move away from this path, those approximations are less and less accurate and so are the feedback terms computed.

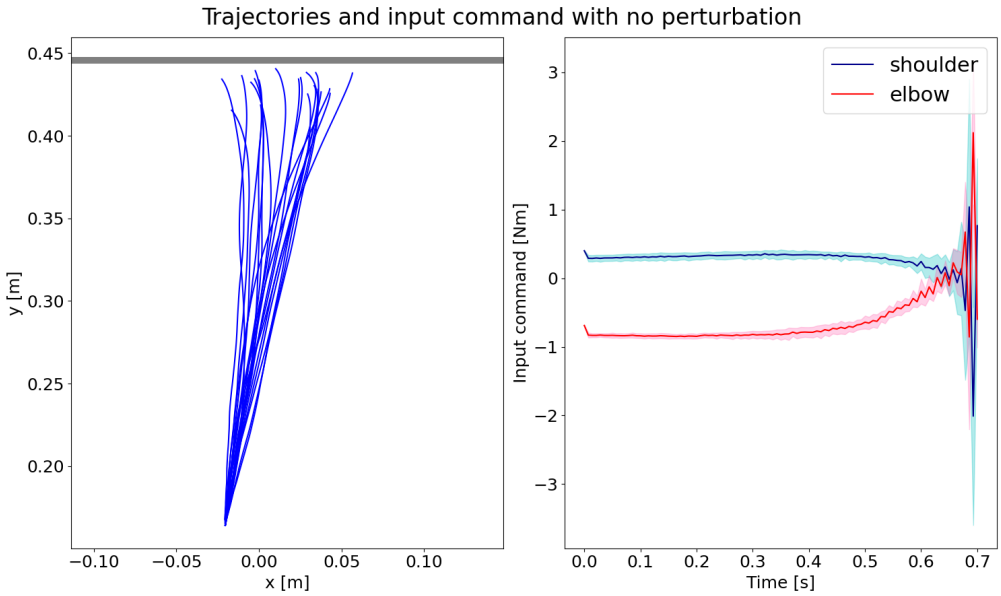

Figure 5.29: Trajectories and input of the system with no perturbation with iLQR for a wide target

If we look at the trajectories in figure 5.30, we can see on the left all the trajectories simulated and on the right a zoom in the same image.
On the figure, we see something that was not observed so far. Indeed, approximately half of them not only fail to converge but also explode due to numerical instabilities.
For those which converge, we observe a similar situation as in the case of the LQG_t where the trajectories do not have enough time to reach the target and stop before.

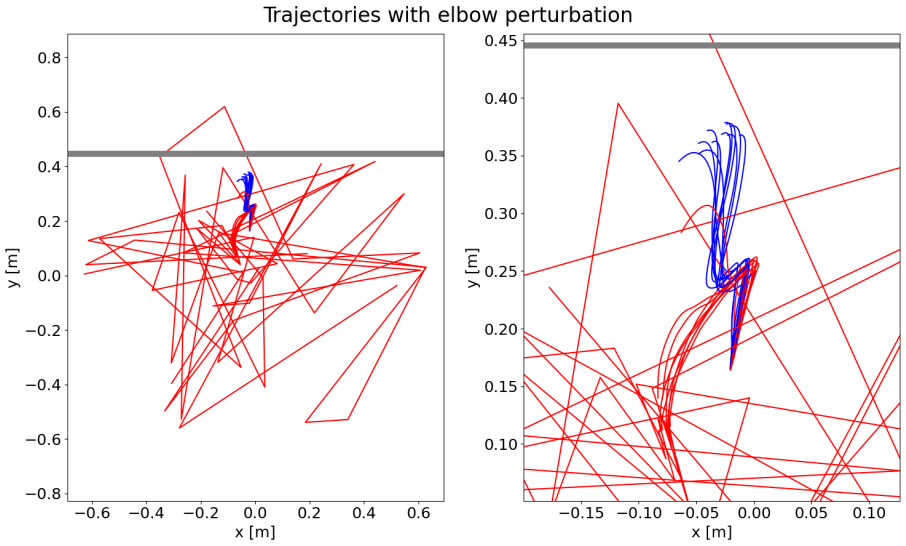

Figure 5.30: Trajectories and input of the system with a perturbation on the elbow with iLQR for a wide target

Finally, the trajectories in figure 5.31 are still not reaching the target very well. However, we observe the human behavior of letting the arm being drifted away by the disturbance. The reason why the trajectories do not reach well the target is very likely the same as in figure 5.29. Indeed, the trajectories being completely deviated from the original path computed by the algorithm of the iLQR, the approximations made about the cost function are not very accurate anymore. Therefore, the feedback gains that the iLQR found do not bring the arm back to the target very precisely.

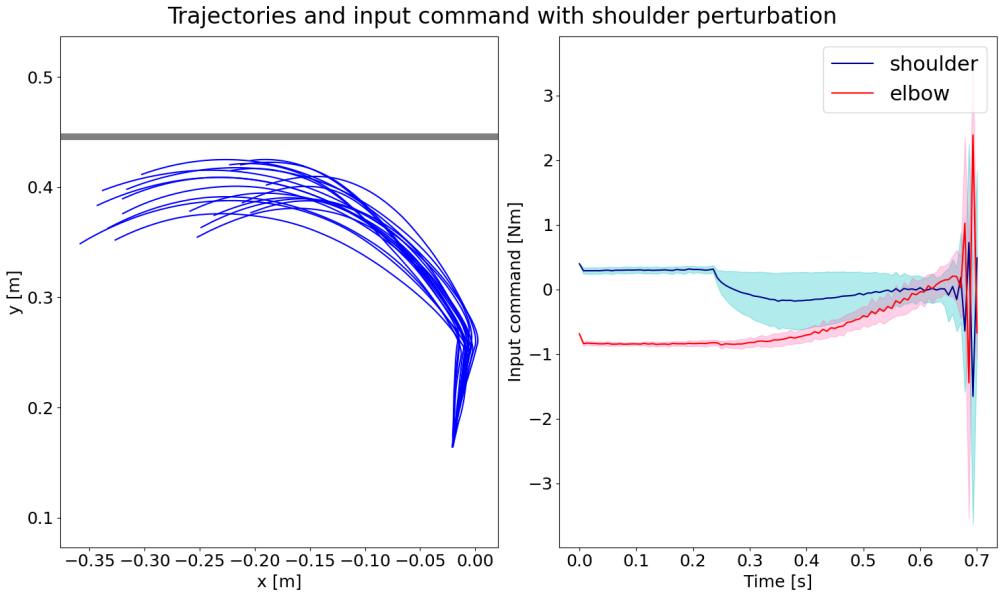

Figure 5.31: Trajectories and input of the system with a perturbation on the shoulder with iLQR for a wide target

Chapter 6

Discussion

In this chapter, we will discuss the similarities between the trajectories simulated by the controllers with the ones realized by humans.
Then, the sensitivity to both types of noise of the controllers will be compared.
Finally, we will discuss the potential improvements that can be made to the model and the iLQR controller.

6.1 Similarity to the human behavior

Now that we have performed all the simulations in the various situations corresponding to the experiments that were presented in chapter 2, we can compare them with the results obtained when people were doing those experiments. This way, we will hopefully see which of the two controllers better mimics the human behavior.
Two different properties of the controllers will be compared: the adaptability and the flexibility. The former is the capacity of the controller to adapt to an external mechanical perturbation and still be able to reach the target. The latter is the capacity of the controller to take into account a change in the shape of the target.
To identify the capacity of adaptability of the controllers, we will use the experiment where people are facing a mechanical perturbation on their arms when reaching a small target. And for the flexibility, we will use both the experiments with an external perturbation and without it, when reaching a large target.

Adaptability

In the case of the LQG_t, it is pretty clear by looking at the figures 5.18 and 5.19 that the controller is dealing very well with external perturbations both on the elbow and the shoulder.
Now, if we look on figure 2.3 at the paths taken by the hand when humans are facing similar conditions, we can see that the trajectories are very similar to those simulated. First, the arm is deviating to the left from its original trajectory and then, the muscles are progressively contracting in order to correct that perturbation. When correcting for the

perturbation, the simulated trajectories do not go back to the original path taken when there was no perturbation and we observe the same behavior for the humans.

When it comes to the iLQR, it is more difficult to draw such good conclusions about its adaptability.
First, if we take a look at the figures 5.22 and 5.23, we can see that all the trajectories do not converge towards the target. This is something that is observed neither in real life nor with the LQG$_t$. A possible explanation for the presence of those outliers is the fact that this algorithm is iterative and makes local approximations of the gradient and the hessian of the cost function during those iterations. Because of those local approximations that are different for each simulation, the algorithm when iterating can "fall" into a local minimum and thus fail to find a proper convergence like it does most of the time.
Now, if we look at the trajectories that do converge, we see that in both cases they do not converge towards the actual target but rather to a point a little bit to the left. That can be explained with the fact that the iLQR is using approximations of the cost function. Based on those approximations, it computes feedback gains that are supposed to minimize the cost function. But those approximations are only valid on the path that the iLQR converged to because our model of the arm is not linear and therefore the dynamics of the arm change in function of its position. Because of the perturbation, the trajectory of the arm is not the same as the one find by the iLQR when iterating and thus the local approximations of the cost function are not accurate anymore. Since the approximations are not accurate, neither are the feedback gains and the trajectories are converging towards another point.

Flexibility

In this section, we focus on the capacity of flexibility of both controllers. To do so, we look at how well the controllers take advantage of a change in the shape of the target and we compare that to the human behavior in the two experiments presented in Figure 2.1.

The first experiment was to reach for a small target and afterwards for a wide one without any perturbation and it was observed that even though the target was still very well reached, the trajectories' end points were more spread when the target was wider than when it was small, showing that people took advantage of the new target's shape.
In the case of the LQG$_t$, when looking at the figures 5.17 and 5.25, we can see that the trajectories follow a similar pattern to the human ones. Indeed, in the case of a small target, we observe that the trajectories that deviate from the straight line between the start point and the target are brought back to the small target by the end of the movement. But when the target is large, the deviating trajectories are not brought back towards the original small target but keep going towards another point on the target in a straight line. This illustrates very well that the LQG$_t$ mimics very well the human behavior in terms of flexibility.

If we look at the same experiment with the iLQR as controller, we can see that even when

the target is small, the trajectories that deviate are not very well corrected and they don't reach as well the target as with the LQG_t. The reason why has already been explained, it is because of the non-linearity of the model and the fact that the feedback gains are computed with the approximations of the cost function on the trajectory found by the iLQR. Therefore, if the trajectories deviate from the original wanted path because of the noise or of a perturbation, the feedback gains are not accurate anymore.
Despite this lack of precision of the feedback gains, if we look at the figure 5.29, we can clearly see that the iLQR does also take advantage of the new shape of the target. Even though the trajectories are not reaching very well the target for the same reasons as before, they clearly diverge more than when the target is small. This shows that despite its lack of accuracy to reach the target, the iLQR seems to still have a good capacity of adaptability in this situation.

Now, to see if in different conditions the controllers still have a good adaptability, we will look at the second experiment where people were asked to reach for a small target and after that for a wide one, with a mechanical perturbation applied on their arm during the movement. That perturbation is important in our case because otherwise, the optimal path to reach the wide target is the same as for the small one.
If we look at the trajectories simulated by the LQG_t when the perturbation is on the elbow, it is not very clear if the controller is really using the new shape of the target as an advantage because the trajectories are not deviated that much from the original trajectories and therefore, their end-points are not so far from the original small target.
But if we now look at the trajectories simulated when the perturbation is applied on the shoulder, we can observe a behavior very similar to the human one. The trajectories are deviated to the left and they are not brought back to the original aimed point on the target.
On top of that, by looking at the input signals and to the trajectories that are perpendicular to the target at the end of the movement, we can deduce that the perturbation starts to be countered before arriving on the target even though the arm could just be deviated more to the left and still arrive on the target. This is something that people also do when doing the same experiment as we can see on figure 2.3. This is due to the fact that they have to stop their hand on the target. Because of that, they start to compensate for the external force before arriving on the target to better control the speed of their hand.
Like the first experiment, the second one seems to show that the LQG_t has a very good capacity of flexibility and similar to that observed in human sensorimotor control.

Now, if we look at the results of the iLQR for the same experiment, we have very different results. First, we can observe a very special phenomenon when the perturbation is applied on the elbow. Indeed, approximately half of the simulated trajectories do not converge at all and, even worse, almost all the ones which don't converge are completely unstable and unrealistic.
In the case of the shoulder disturbance, the trajectories all converge and are realistic. But as always when they deviate from the original path designed by the iLQR, the end-points do not arrive on the target. However, if we only focus on the capacity of adaptability, we

can clearly say that the algorithm takes the new shape of the target into account and does not correct the perturbation when it is useless. But even though the iLQR takes the shape of the target into account and therefore has a good flexibility, the way it does it, is completely different from how humans do it and it gives completely different trajectories. For example, the trajectories do not arrive perpendicularly to the target as we observed before. Since our purpose is to try to find a controller that would give similar results to the human sensorimotor control, this experience highlights the fact that this controller does not mimic well the human behavior even if it has a good capacity of adaptability.

Summary

It appears that the LQG$_t$ has a closer way to control the human arm and perform reaching movement like humans do than the iLQR. For the adaptability, it seems pretty clear that having a convergence towards a certain trajectory to compute feedback gains offline, like the iLQR does, is not a good thing and does not correspond to the human behavior, which can very well adapt to external perturbations. Because, in humans' trajectories, we do not observe that lack of precision to reach the target as soon as a perturbation is applied. For the flexibility, both controllers seem to take the shape of the target into account, but again the LQG$_t$ does it in a closer way to the human one than the iLQR.

6.2 Sensitivity

The last property that we want to investigate in our controllers is their sensitivity to the noises. In this case, we have two different kinds of noises: the process and the observation noise. The former will put some randomness on the dynamics of the biomechanical model and the latter on the estimation of the state by the person.

Process noise

For the LQG$_t$ controller, it appears clearly that the process noise does not have that much impact on its capacity to reach the target, but as mentioned in the previous chapter, it causes the trajectories to be more spread and therefore the input signals are very noisy. This seems logical since this controller is based on the feedback and increasing the process noise will make the dynamics of the system more random but the observation of the state used for the feedback will not be altered. So there is no reason for the feedback to be less appropriate.

On the other hand, if we look at the impact that a high process noise has on the iLQR, we can see that it is much more important. Indeed, the trajectories are also much more spread than with a "standard" process noise but unlike the LQG$_t$, the iLQR does not manage to bring them back to the target.
This is due to the fact that the iLQR is an mixture of an optimal feedback controller and a trajectory based one. As a matter of fact, the convergence towards a trajectory

is made harder by the process noise because at each iteration, even if the input signals do not change, the resulting simulated trajectory will be much more different from the previous one computed. For that reason, the algorithm does not converge successfully because measuring the cost of a control sequence is not precise enough.

To sum up, the LQG_t handles better that kind of noise that the iLQR because it is based on feedback and not on a precise trajectory.

Observation noise

Now, if we look at the impact of the increase of observation noise on the LQG_t, we see that it is much more important than the one of the process noise. Again, based on what has been said above, it makes sense because this controller is based on feedback and that feedback is based on the estimate of the state. If the observation noise is increased, it means that the estimation of the state is less good and therefore, the feedback cannot be optimal.
This seems to be in line with what we would observe in a human being. If a human cannot distinguish the exact position of his hand, he will have more difficulties in reaching a target.

For the iLQR, it is the opposite. The observation noise has a smaller impact on the costs of the movements than the process noise, even if the median cost of the iLQR with a high observation noise is higher than the median cost of the LQG_t in the same conditions. We observe, nevertheless, that the respective differences with the standard cases are completely different.
The observation noise does not impact that much the trajectories simulated with the iLQR because this controller is computing an optimal trajectory. Even though it still has some impact on the control because the iLQR is based on the LQR and there is obviously still some feedback based on an estimate of the state, the observation noise does not deflect the trajectories as much as the process noise and thus the convergence can be better achieved.

6.3 Possible improvements

As a conclusion to this work, we will see two possible improvements that can be made to the model used and to the iLQR controller in order to get more realistic results.

2-link 6-muscle

As detailed in chapter 3, the model we use in this thesis is a 2-link 2-muscle model. One improvement that can be made to this model, and would therefore enable us to have a more realistic analysis of the contraction of the muscles in the arm, is to transform it into a 2-link 6-muscle model.
In this work the muscles were simply modeled with the equation 3.8 as a low pass filter but in the literature, another model composed of 6 muscles instead of only 2 can be found[15].

The full development will not be detailed here but it is important to emphasize that this modelling does not simply add 4 more muscles, it also takes into account biomechanical properties of the muscles that we have ignored.
Furthermore, in this work we allowed the muscles to apply both a negative and a positive torque, which does not make sense from a biological point of view because a muscle can only make contractions and therefore apply a torque in only one direction of rotation.

iLQG

Another possible improvement concerns the iLQR. As indicated in chapter 3, the iLQR is not originally designed to work with stochastic systems. Nevertheless, in order to be able to use it with our noisy model, we combined it with a Kalman filter to get an estimate of the state. The idea was to combine it with the iLQR the same way the Kalman filter and the LQR are combined to form the LQG.

In fact, there exists another algorithm very similar to the iLQR named the iLQG[15]. This algorithm is essentially the same as the iLQR but it is designed to work with stochastic systems and even with state and input dependent noises which we have simplify as additive noises in this thesis, even though we know that the noise on the command signals to the muscle is signal-dependent[24].
It has been tried during the writing of this thesis to implement the iLQG controller, but unfortunately it has never worked. The reason why is still not very clear because the principle is very close to the iLQR and the latter worked well. It may be due to the fact that the choice of a hyperparameter similar to the λ of the iLQR was more sensitive and this induced too many instabilities.
Notice that the word sensitive here is not used like in the previous section to denote a tendency to be impacted by noisy signals, but rather a tendency of the algorithm to be impacted by the choice of value of its hyperparameters and in our case prevent convergence.

Chapter 7

Conclusion

In this work we have seen how the sensorimotor control in humans is organized, how the human arm can be modeled. We have also seen the structure of two controllers that can be used to simulate reaching movements, how each one reacts to two different experiments and finally we compared the simulated trajectories of each controller with the human ones to identify two properties of the controllers: their adaptability and flexibility.

Thanks to the simulations with a mechanical perturbation on the arm, it appeared that the controller based on the optimal feedback control theory, i.e. the LQG$_t$, had a capacity of adaptability far closer to the human one than the iLQR did.
The simulations with a wide target highlighted the differences in flexibility between the two controllers. Even though both of them took advantage of the change of shape of the target in the first experiment where no perturbation was applied, the trajectories simulated for the second experiment by the LQG$_t$ were far more similar to the ones realized by humans than the ones obtained with the iLQR.
We also saw that as one could have expected, the controller based on optimal feedback was more affected by the observation noise than the trajectory based one. And conversely with the process noise where the LQG$_t$ was less affected than the iLQR.

To sum up, it appears clearly that the LQG$_t$ controller, the one purely based on the optimal feedback theory, has a better capacity to mimic the human behavior than the iLQR that converges towards an optimal trajectory. This reinforces the idea that sensorimotor control in humans behaves like an optimal feedback controller and that it is not based on a predefined trajectory.

Bibliography

[1] T. Cluff and S. H. Scott. "Apparent and Actual Trajectory Control Depend on the Behavioral Context in Upper Limb Motor Tasks". In: *Journal of Neuroscience* 35.36 (Sept. 2015), pp. 12465–12476.

[2] Tamar Flash, Yaron Meirovitch, and Avi Barliya. "Models of Human Movement: Trajectory Planning and Inverse Kinematics Studies". In: *Robotics and Autonomous Systems* 61.4 (Apr. 2013), pp. 330–339.

[3] Emanuel Todorov and Michael I. Jordan. "Optimal Feedback Control as a Theory of Motor Coordination". In: *Nature Neuroscience* 5.11 (Nov. 2002), pp. 1226–1235.

[4] Justin Won and Neville Hogan. "Stability Properties of Human Reaching Movements". In: *Experimental Brain Research* 107.1 (Nov. 1995).

[5] Jörn Diedrichsen, Reza Shadmehr, and Richard B. Ivry. "The Coordination of Movement: Optimal Feedback Control and Beyond". In: *Trends in Cognitive Sciences* 14.1 (Jan. 2010), pp. 31–39.

[6] Stephen H Scott. "Role of Motor Cortex in Coordinating Multi-Joint Movements: Is It Time for a New Paradigm?" In: *Canadian Journal of Physiology and Pharmacology* 78.11 (Nov. 2000), pp. 923–933.

[7] Stephen H. Scott. "Optimal Feedback Control and the Neural Basis of Volitional Motor Control". In: *Nature Reviews Neuroscience* 5.7 (July 2004), pp. 532–545.

[8] Frédéric Crevecoeur, Tyler Cluff, and Stephen Scott. "Computational Approaches for Goal-Directed Movement Planning and Execution". In: *The Cognitive Neurosciences*. Oct. 2014, pp. 461–475.

[9] Stephen H. Scott. "A Functional Taxonomy of Bottom-Up Sensory Feedback Processing for Motor Actions". In: *Trends in Neurosciences* 39.8 (Aug. 2016), pp. 512–526.

[10] Frédéric Crevecoeur, Jean-Louis Thonnard, and Philippe Lefèvre. "A Very Fast Time Scale of Human Motor Adaptation: Within Movement Adjustments of Internal Representations during Reaching". In: *eneuro* 7.1 (Jan. 2020).

[11] Joseph Y. Nashed, Frédéric Crevecoeur, and Stephen H. Scott. "Influence of the Behavioral Goal and Environmental Obstacles on Rapid Feedback Responses". In: *Journal of Neurophysiology* 108.4 (Aug. 2012), pp. 999–1009.

[12] Joseph Y Nashed, Frederic Crevecoeur, and Stephen H Scott. “Rapid Online Selection between Multiple Motor Plans”. In: (2014).

[13] Antoine De Comite, Frédéric Crevecoeur, and Philippe Lefèvre. “Online Modification of Goal-Directed Control in Human Reaching Movements”. In: *Journal of Neurophysiology* 125.5 (May 2021), pp. 1883–1898.

[14] P M Mäkilä. “Kalman Filtering and Linear Quadratic Gaussian Control”. In: (2004).

[15] W. Li and E. Todorov. “Iterative Linearization Methods for Approximately Optimal Control and Estimation of Non-Linear Stochastic System”. In: *International Journal of Control* 80.9 (Sept. 2007), pp. 1439–1453.

[16] Frederic Crevecoeur. “Online Parameter Tracking in Human Reaching Adaptation and Control”. In: ().

[17] Yuval Tassa, Nicolas Mansard, and Emo Todorov. “Control-Limited Differential Dynamic Programming”. In: IEEE, May 2014, pp. 1168–1175.

[18] Emanuel Todorov and Weiwei Li. “Iterative Linear Quadratic Regulator Design for Nonlinear Biological Movement Systems”. In: SciTePress - Science and and Technology Publications, 2004, pp. 222–229.

[19] Travis DeWolf. *The Iterative Linear Quadratic Regulator Algorithm*. Feb. 2016.

[20] Travis DeWolf. *iLQR Implementation*. `https://github.com/studywolf/control`.

[21] Abdel-Nasser Sharkawy. “Minimum Jerk Trajectory Generation for Straight and Curved Movements: Mathematical Analysis”. In: *Mathematical Analysis*. Vol. 2, pp. 187–201.

[22] Nghia Minh and Levente Hajder. “Affine Transformation from Fundamental Matrix and Two Directions:” in: SCITEPRESS - Science and Technology Publications, 2020, pp. 819–826.

[23] Paul Seeburger. *Taylor Polynomials of Functions of Two Variables*. `https://math.libretexts.org/Bookshelves/Calculus/Supplemental_Modules_(Calculus)/Multivariable_Calculus/3%3A_Topics_in_Partial_Derivatives/Taylor__Polynomials_of_Functions_of_Two_Variables`. Dec. 2020.

[24] Kelvin E. Jones, Antonia F. de C. Hamilton, and Daniel M. Wolpert. “Sources of Signal-Dependent Noise During Isometric Force Production”. In: *Journal of Neurophysiology* 88.3 (Sept. 2002), pp. 1533–1544.

UNIVERSITÉ CATHOLIQUE DE LOUVAIN
École polytechnique de Louvain
Rue Archimède, 1 bte L6.11.01, 1348 Louvain-la-Neuve, Belgique | **www.uclouvain.be/epl**