DOI: 10.2478/sbeef-2023-0017

DESIGN OF MATLAB/OPENSIM ROBUST CONTROLLER OF AN EXOSKELETON FOR DISUSE MUSCULAR ATROPHY REHABILITATION OF A HUMAN ARM

Dalila DJOUDI* and, Zakaria BOUZID**

*Applied Automation and Industrial Diagnostic Laboratory, University of Djelfa, 1st Elmoudjbara road, Zip code 17000, Djelfa, Algeria.

**Department of electrical engineering, University of Djelfa, 1st Elmoudjbara road, Zip code 17000, Djelfa, Algeria.

Abstract : ***The mean goal of this paper is to describe the design and control approach of an exoskeleton for rehabilitation of the disuse muscular atrophy of a human arm. This work includes three main parts: Firstly, the exoskeleton model wes design by Autodesk Inventor 3D software. Secondly, the dynamic simulation of the designed exoskeleton model attached to the human arm was performed using OpenSim software and its Matlab API extension, and finally a robust control law was simulated in order to ensure tracking of the rehabilitation trajectories applied by the exoskeleton to the human arm. OpenSim software makes it possible to simulate movements with musculoskeletal models, namely, the human arm. Rehabilitation in this case consists in a precise exercises given by the therapist. In our case, it is the repetitive trajectories given to the exoskeleton that must be controlled. A sliding mode controller was used since it is a robust control and ensures the best solutions to uncertainty issues. Through simulation, we tested some rehabilitation reference trajectories for the elbow and shoulder. The controller ensures high performances in terms of trajectory tracking in the presence of initial errors and also in the presence of model parameter errors. That showed the effectiveness of the exoskeleton control.***

Keywords: Robust control, Sliding mode control, exoskeleton design, disuse muscular atrophy, therapeutic exercise, initial conditions errors, model parameters errors.

1. INTRODUCTION

The exoskeleton is an electromechanical structure that can be worn by an operator to match the shape and functions of the human body. It can augment the ability of human body parts to treat muscles, joints, or skeletal parts that are weak, ineffective, or injured because of a disease or a neurological condition. Many studies have been conducted to enhance the performance of exoskeletons, and many designs have been proposed [1]-[6]. In [2], a review of the lower limb exoskeleton designs has been conducted. The authors in [5] review the recent progress of upper-limb exoskeleton robots. [7]-[10] were interested in finger control or knee control. Exoskeletons are sometimes complicated devices that are relatively expensive. Each exoskeleton is designed for a predefined purpose; they can be used for military, industrial, or medical purposes. In our work, we are interested in an exoskeleton that is used to rehabilitate patient muscles. Rehabilitation exercises which are quite simple and repetitive, strengthen the missed connections between the mind and muscles, adjusting the pathways and preventing the spasms[9]. The exoskeleton helps patients perform their prescribed rehabilitation movements without resorting to a practitioner, some patients may avoid exercises because of pain or simply, because of laziness.

We focus on the atrophy of the brachialis muscle flexion [11]. For this purpose, a complete exoskeleton for the whole limb, like in [5], is not necessary; a simpler exoskeleton with 2 degrees of freedom can be used and can be cheaper than a complete limb exoskeleton, consequently making it more accessible for a wider range of patients for home use.

In this paper, we are interested in an exoskeleton that we design for muscle atrophy rehabilitation in the human arm. Disuse muscular atrophy requires rehabilitation exercises to effectively restore mass and function. These exercises are conducted by the therapist; in our case, an exoskeleton is designed and used to do these exercises. So in this paper, we present the exoskeleton model and the human arm model to be attached to. In order to control this system, many control laws exist in the literature, but the sliding mode seems to be one of the most used theoretically for its robustness against uncertainties [12] [13], and its proven global stability, which makes it possible to avoid tracking errors with or without model errors or initial state errors. The computed torque control law [14] can predict the exact torque value needed to ensure perfect tracking, but with model parameter's errors, the calculated torque is not suitable anymore, because it uses a wrong parameter's values.

Many authors adopt the sliding mode control to converge in finite time, either by back-stepping methods [10], adaptive methods [15], or learning methods. The finite-time sliding mode control is quite difficult to implement and tune since it has many parameters and gains. In our work, we will apply the sliding mode control strategy and test its robustness against initial condition errors and model parameter errors.

The problem with the sliding mode, especially in practical implementation, is its discontinuous part, which creates the chattering problem; the actuators have to cope with the high-frequency control actions. This problem can be avoided simply by approximating the discontinuous part, but this may alter tracking performances. In this work the approximation of the discontinuous function has been but tracking errors was avoided by suitable tuning of the control gains.

2. DISUSE MUSCLE ATROPHY

Before talking about exoskeleton design and control, we must first define what disuse muscle atrophy is. Muscle atrophy is defined as the wasting or thinning of the

muscle mass and the decrease of the isometric force. It can be caused by disuse of the muscles or neurogenic conditions[16]. We are interested in brachialis muscle atrophy. Figure 1 shows the difference between atrophied and healthy brachialis muscles[17].

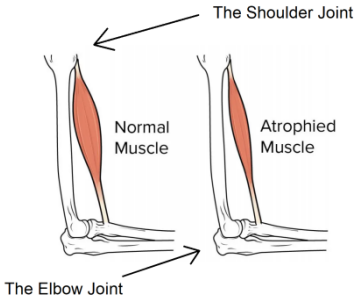


Figure 1. The brachialis muscle: Atrophied muscles are smaller than healthy muscles [16].

3. 3D EXOSKELETON MODEL

The suitable exoskeleton to rehabilitate the brachialis muscle does not need a lot of degrees of freedom. 2 degrees of freedom are enough to help the human arm perform the needed rehabilitation exercises [18]. The shoulder joint and the elbow joint are exactly parallel to the real joints; see Figure 2. The exoskeleton model was designed using 3D mechanical solid modelling software, AutoDesk Inventor. Table 1 lists the exoskeleton dynamic parameters.

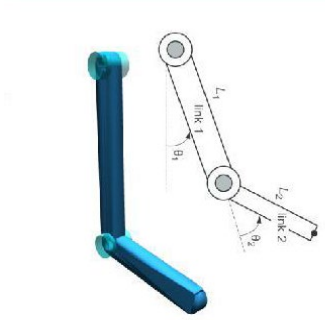


Figure 2. 2 Degrees of Freedom exoskeleton designed in Autodesk inventor software.

Table 1. Dynamic parameters of the exoskeleton links

Exo Parts	Mass(kg)	Length(cm)	Width(cm)	Heigh(cm)
Upper Arm	1	29	4	2
Forearm	0.8	18	4	2

Let's note that the ranges within which the shoulder and elbow joints can evolve are, respectively, [0 180°] and [0 145°]. See Table 2.

Table 2. Joint's stops of the exoskeleton

Exo Joints	Lower limit(rad)	Upper limit(**rad**)
Shoulder	0	3.14
Elbow	0	2.53

4. OPENSIM API FOR MATLAB

OpenSim [19], is open-source software that allows users to develop, analyze, and visualize models of the musculoskeletal system and to generate dynamic simulations of movement [20][21]. OpenSim can be extended by using its API, Application Programming Interface, through C++, Matlab, or Python [22]. In our work, we have used Matlab.

In order to control the exoskeleton and test its matching to the human arm shape, we have used OpenSim to design the musculoskeletal arm model and to import the exoskeleton parts. Figure 3 shows the whole model, including the exoskeleton and the human arm model.

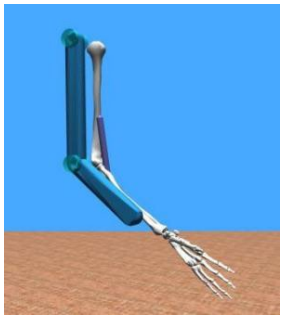


Figure 3. Imported Exoskeleton Model by OpenSim to be Attached to the Arm Musculoskeletal Model

This system makes it possible to apply our control approach, which produces external forces on the human arm. The accuracy of the control law in tracking the reference trajectory and the effect of the exoskeleton on the atrophied muscle rehabilitate it by changing the rate of change in the length of the fibers **Erreur ! Source du renvoi introuvable.**.

To train the human arm, some specific reference trajectories have to be tracked by the exoskeleton. They are inspired by real rehabilitation exercises [18] and interpolated using Matlab. A control law must be chosen and designed in order to have good tracking of reference trajectories. Sliding mode control is chosen for its robustness against external disturbances and unpredictable parameter variations [25],[26].

Figure 4 represents the simulation scheme for the exoskeleton. The direct dynamic model is given by Opensim, and the control design and reference trajectories are generated in Matlab and exported to Opensim via the API. OpenSim requires the dynamic parameters of the exoskeleton, such as masses and lengths, but it does not require the inertia tensors of the links; it calculates them internally.

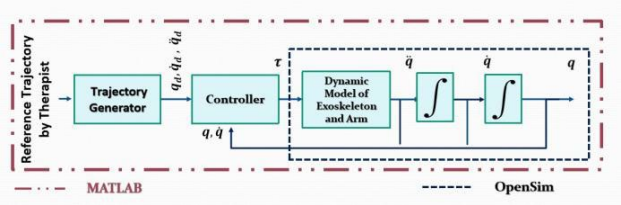


Figure 4.Simulation scheme between Matlab and Opensim

5. THE EXOSKELETON DYNAMIC MODEL

The arm motion, consequently the exoskeleton motion, is limited in the sagittal plane, and the task of control consists of controlling the shoulder and elbow joint's flexions. See Figure 3. The general form of the dynamic model generated by Opensim is given in the following form [27]:

$$M(q)\ddot{q} + C(q,\dot{q}) + G(q) = \Gamma \quad (1)$$

Where:
n is the number of degrees of freedom: n=2.
$q, \dot{q}, \ddot{q} \in R^n$ are the vectors of generalized joint positions, velocities and accelerations, respectively.
$M(q) \in R^{n,n}$ is the system mass matrix.
$C(q,\dot{q}) \in R^n$ is the vector of coriolis and centrifugal forces.$C(q,\dot{q})$=$\boldsymbol{C}(q,\dot{q})\dot{q}$, according to conventional robots dynamic model [14].
$G(q) \in R^n$ is the vector of gravitational forces.
$\Gamma \in R^n$ is the vector of generalized forces: in our case they are torques. It has the same dimension as the number of degrees of freedom since the exoskeleton is completely actuated.
OpenSim has a package called Simbody which provides the dynamic model matrices and vectors. OpenSim uses them to construct the multibody system that underlies an OpenSim dynamic model -[20], [27].

5.1. State Space model of the Exoskeleton

In this section, we give the state-space formulation of the exoskeleton so that it will be possible to calculate our control law designed in this state-space representation.

By defining the state vector $x \in R^n$ as:

$$\begin{cases} x_1 = q \\ x_2 = \dot{q} \end{cases} \quad (2)$$

Calculating time derivative of equation (2), taking into account equation (1) yields to:

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = -M^{-1}(q)(C(q,\dot{q}) + G(q)) + M^{-1}(q)\tau \end{cases} \quad (3)$$

Thus, the state space representation is given by:

$$\begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \end{pmatrix} = \begin{pmatrix} x_2 \\ -M^{-1}(x_1)\big(C(x_1,x_2) + G(x_1)\big) \end{pmatrix} + \begin{pmatrix} 0 \\ M^{-1}(x_1) \end{pmatrix}\tau \quad (4)$$

In matrix form we write:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & \boldsymbol{I} \\ 0 & 0 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ -M^{-1}(x_1)\big(C(x_1,x_2) + G(x_1)\big) \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ M^{-1}(x_1) \end{bmatrix}\tau \quad (5)$$

Where :
0 is a (2×2) zeros matrix;
I a (2×2) identity matrix;

6. SLIDING MODE CONTROL OF THE EXOSKELETON

In this section, we design the SMC to control the dynamic model of the exoskeleton.
Let's consider that the output is the joint position q. Thus, the dynamic model of the exoskeleton is given as:

$$\begin{cases} \dot{X} = \begin{bmatrix} \mathbf{0} & \boldsymbol{I} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} X + \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ -M^{-1}\big(C(X) + G(X)\big) \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ M^{-1}(X) \end{bmatrix}\tau \\ y = [\boldsymbol{I} \quad \mathbf{0}]\, X \end{cases} \quad (6)$$

Where
0 is a (2×2) zeros matrix;
I a (2×2) identity matrix;

The purpose of the control law is to track the reference trajectory$x_d = [q_d \ \dot{q}_d]'$. Thus, The suggested control law is [26] :

$$\tau(X) = \tau_{eq}(X) + \tau_d(X) \quad (7)$$

And the vector of sliding surface is defined as:

$$s(X) = \dot{e} + \Lambda e \quad (8)$$

Where:
e : is the error vector $e = x - x_d$.
Λ: is a (2×2) positive definite diagonal matrix.

Thus equation (8) becomes:

$$s(X) = (\dot{x} - \dot{x}_d) + \boldsymbol{\Lambda}(x - x_d) \quad (9)$$

In matrix form, we get:

$$s(X) = [\boldsymbol{\Lambda} \quad \mathbf{I}](X - X_d) \quad (10)$$

To ensure that all trajectories of the system track the desired ones, the sliding mode dynamics are null, it means that: $s(X) = \dot{s}(X) = 0_{2,1}$.

Now, let's calculate the equivalent control law τ_{eq}, which satisfies $\dot{s}(X) = 0_{2,1}$. We have:

$$\dot{s}(X) = \frac{\partial s(X)}{\partial X}\frac{dX}{dt} + \frac{\partial s(X)}{\partial X_d}\frac{dX_d}{dt} \quad (11)$$

Thus:

$$\dot{s}(X) = [\boldsymbol{\Lambda} \quad I]\frac{dX}{dt} - [\boldsymbol{\Lambda} \quad I]\frac{dX_d}{dt} \quad (12)$$

Substituting $\frac{dX}{dt}$ in (6) in equation (12), yields to:

$$\dot{s}(X) = [\boldsymbol{\Lambda} \quad I]\left(\begin{bmatrix}\mathbf{0} & \boldsymbol{I}\\ \mathbf{0} & \mathbf{0}\end{bmatrix}X + \begin{bmatrix}\mathbf{0}\\ \mathbf{0}\\ -M^{-1}(C(X)+G(X))\end{bmatrix} + \begin{bmatrix}\mathbf{0}\\ M^{-1}(X)\end{bmatrix}\tau_{eq}\right) - [\boldsymbol{\Lambda} \quad I]\dot{X}_d$$

But, $\dot{s}(X) = 0_{21}$, so:

$$[\boldsymbol{\Lambda} \quad I]\left(\begin{bmatrix}\mathbf{0} & \boldsymbol{I}\\ \mathbf{0} & \mathbf{0}\end{bmatrix}X + \begin{bmatrix}\mathbf{0}\\ \mathbf{0}\\ -M^{-1}(C(X)+G(X))\end{bmatrix} + \begin{bmatrix}\mathbf{0}\\ M^{-1}(X)\end{bmatrix}\tau_{eq}(X)\right) - [\boldsymbol{\Lambda} \quad I]\dot{X}_d = \mathbf{0}_{21} \qquad (13)$$

Now, the equivalent control law is:

$$\tau_{eq}(X) = \left([\boldsymbol{\Lambda} \quad I]\begin{bmatrix}\mathbf{0}\\ M^{-1}(X)\end{bmatrix}\right)^{-1}\left\{[\boldsymbol{\Lambda} \quad I]\dot{X}_d - [\boldsymbol{\Lambda} \quad I]\left(\begin{bmatrix}\mathbf{0} & \boldsymbol{I}\\ \mathbf{0} & \mathbf{0}\end{bmatrix}X + \begin{bmatrix}\mathbf{0}\\ \mathbf{0}\\ -M^{-1}(C(X)+G(X))\end{bmatrix}\right)\right\} \qquad (14)$$

And the formula of $\tau_d(X)$ which is the discontinuous part will be:

$$\tau_d(X) = -\boldsymbol{K}\, sign(s) \qquad (15)$$

The final equivalent control form is:

$$\tau(X) = \left([\boldsymbol{\Lambda} \quad \boldsymbol{I}]\begin{bmatrix}0\\ M^{-1}(X)\end{bmatrix}\right)^{-1}\left\{[\boldsymbol{\Lambda} \quad \boldsymbol{I}]\dot{X}_d - [\boldsymbol{\Lambda} \quad \boldsymbol{I}]\left(\begin{bmatrix}\mathbf{0} & \boldsymbol{I}\\ \mathbf{0} & \mathbf{0}\end{bmatrix}[X + \begin{bmatrix}0\\ 0\\ -M^{-1}(C(X)+G(X))\end{bmatrix}\right)\right\} - Ksign(s) \qquad (16)$$

$\boldsymbol{K}$ is a (2×2) positive definite diagonal matrix.

The stability of this control law is proven using Lyapunov function [26].

7. SIMULATION RESULTS

7.1. Trajectory generation

In real life, we train brachialis muscle without a shoulder movement which can diminish the effectiveness of the training of the Brachialis muscle. The flexion of the elbow is the only target movement, but for the sake of generality, both movements of the shoulder and the Elbow are considered.
The desired trajectory of the exoskeleton is illustrated in Figure 5. It is defined such that the joint positions vary from an initial position to a final position through an intermediate position as a function of time until a final time of tf = 12s. The initial joint position is $q_i = [0 \quad 0]^t$ and the final one is $q_f = [\pi/6 \quad \pi/4]^t$. The intermediate position is: $q_{int1} = [\pi/6 \quad \pi/2]^t$.

These positions correspond to the time points: $t = [0\ 1\ 2]^t$



Figure 5. The repetitive reference trajectory

Starting with the initial position, the exoskeleton keeps oscillating between the intermediate and final positions with a period of T = 2s until the final time tf. The desired trajectory for joint positions is depicted in Figure 6. The final time tf has been chosen arbitrarily for testing the simulation. The necessary time for rehabilitation depends on many factors related to the patient's health status and muscle status. In order to avoid discontinuities in the exoskeleton's motion, joints' velocities and accelerations in initial, intermediate, and final positions are null.

Matlab function "quinticpolytraj" was used in interpolating the trajectory points, with sampling time Td. The generated trajectories are then exported as '.sto' extension files recognizable by OpenSim in order to run the simulation tests for the whole system's exoskeleton and human arm.

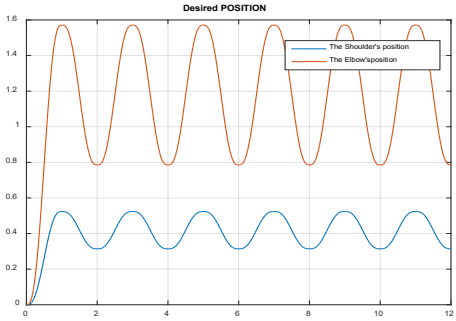


Figure 6. The shoulder and elbow's desired joint positions

The required torques to accomplish the desired motion are given in Figure 7. They are evolving in the range [3, 5] Nm for the elbow and [7, 10] Nm for the shoulder. These values are acceptable for an exoskeleton.

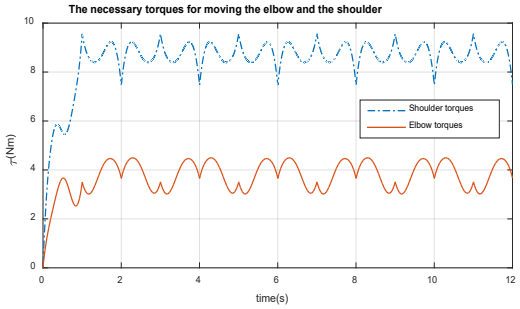


Figure 7. Required torques

7.2. Sliding Mode Control in perfect conditions

The control law in equation (16) is applied to the exoskeleton attached to the human arm with the dynamic model in equation (1). The values of the diagonal matrix elements, which are the gains of the control law, have been tuned in order to have good tracking errors for both joint positions and velocities and also to have minimal shuttering behavior. The error tolerance boundary is within $\pm 10^{-3}$.

For a perfect model without any parameter errors and without initial tracking errors, the following values are taken:

$$\boldsymbol{\Lambda} = \begin{bmatrix} 0.01 & 0 \\ 0 & 12 \end{bmatrix} \text{ and } \mathbf{K} = \begin{bmatrix} 150 & 0 \\ 0 & 200 \end{bmatrix}$$

These parameters are the best ones obtained by trial-and-error tests in order to have good performances in terms of tracking errors and shuttering.

Figure 8 and Figure 9 show that the exoskeleton tracks perfectly the reference trajectory for both the desired joint positions and velocities.

All tracking errors are about $\pm 10^{-5}$ without shuttering. The torques are also the same as those given in Figure 7, since the exoskeleton tracks perfectly on the desired trajectory.

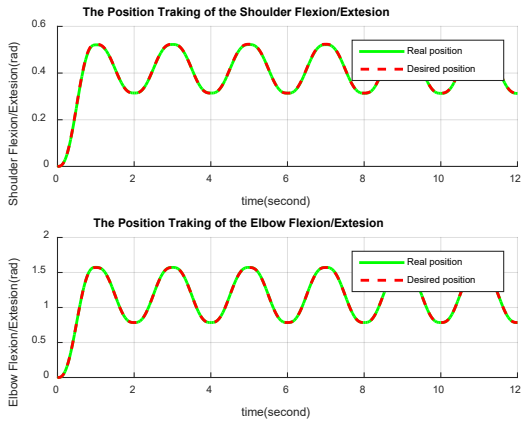


Figure 8. The joint positions tracking of the elbow and shoulder

7.3. Robustness tests:

7.3.1 Exoskeleton control in the presence of initial errors:

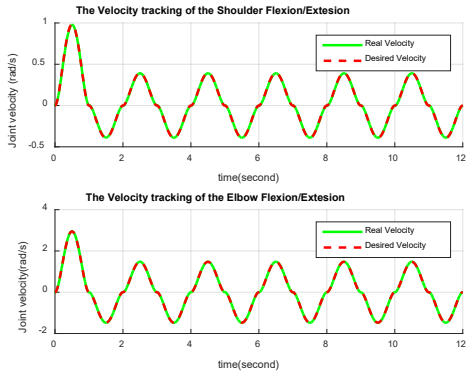


Figure 9. The joint velocities tracking of the elbow and shoulder

We assume that the exoskeleton starts up its motion with an initial error of 10° both for the shoulder and the elbow. The tuning of the control gains is:

$$\boldsymbol{\Lambda} = \begin{bmatrix} 100 & 0 \\ 0 & 50 \end{bmatrix} \text{ and } \boldsymbol{K} = \begin{bmatrix} 150 & 0 \\ 0 & 200 \end{bmatrix}$$

These gains tuning makes it possible for the torques to make the tracking errors converge rapidly toward the error tolerance boundary in about 1 second. Figures 10 and 11 depict, respectively, the convergence of the real position and velocity towards the desired ones.

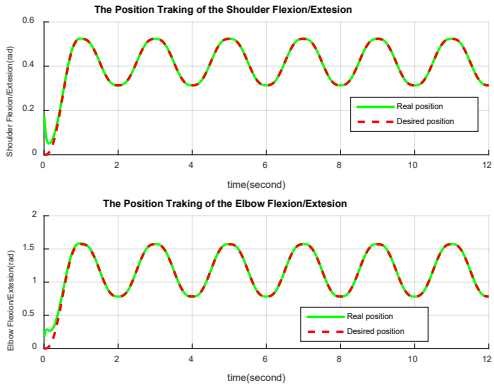


Figure 10. Joint's position tracking in the presence of initial errors

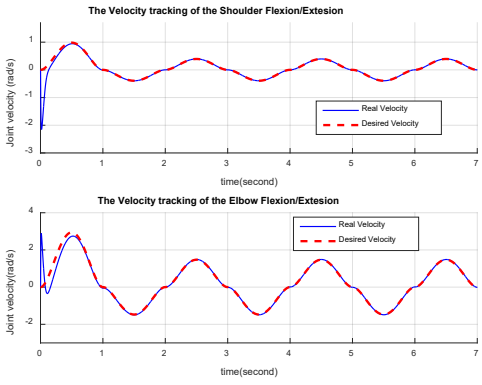


Figure 11. Joint's velocity tracking in the presence of initial errors

This convergence requires quite an important torque at the beginning for approximately 0.01 seconds. The initial torque required by the control law for the cancellation of tracking errors is about -100 Nm; see Figure 12. Saturation on the control law is needed in order to limit the maximum acceptable torque; this depends on the actuator's limitations.

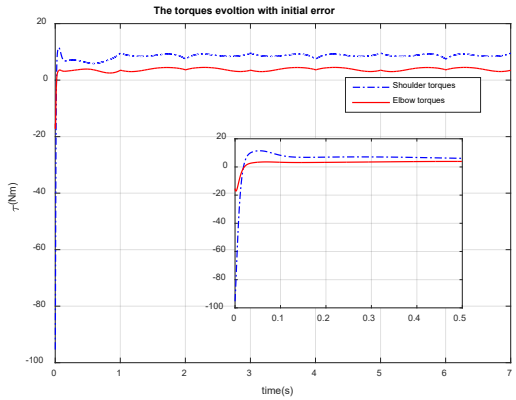


Figure 12. The torques evolution in the presence of initial errors

With the same simulation parameters, but taking into account a saturation to have torques within the range [-30 +30]Nm. The results are almost the same since the needed torques in Figure 12 exceed the limit for a very short time period, less than 0.5 s. The position tracking errors for both the shoulder and elbow and the velocity tracking error for the elbow settles at about $\pm 10^{-3}$, while the velocity tracking error for the shoulder settles at about $\pm 10^{-2}$.

The errors can be accepted as long as the position joints for the elbow are within the range of the exoskeleton joints stops, see Table 2, and within the maximum actuator torques, velocities, and accelerations.

7.3.2 Exoskeleton Control in the presence of model parameters errors.

In practice, the exoskeleton parameters are not perfectly known. We assume that we have some errors in the masses and consequently in the inertia moments of the exoskeleton links. This leads to a global error on the inertia matrix, which has the most effect on the exoskeleton dynamics; see equation (1). The robustness of the SMC against model errors is studied by introducing an arbitrary error on the inertia matrix diagonal. To do so, we recorded the maximum values of the inertia matrix diagonal over the whole reference trajectory. The maximum value of M for the trajectory given in Section 8.1 is: *M_max=[0.6465 0.2200]*.

The error is considered to be 100% of M_max on the inertia matrix diagonal.

$$M = M_n + Diag(M_max))$$

This percentage is the maximum error that the SMC control can handle under the constraints of maximum torques with the following tuned gains:

$$\boldsymbol{\Lambda} = \begin{bmatrix} 150 & 0 \\ 0 & 200 \end{bmatrix} \text{ and } \boldsymbol{K} = \begin{bmatrix} 100 & 0 \\ 0 & 500 \end{bmatrix}$$

In this case, the control law ensures perfect tracking of the reference trajectory even if we have important model errors. Figure 13 depicts the evolution of the tracking errors, which are within $\pm 10^{-3}$ and $\pm 10^{-4}$, over the motion.

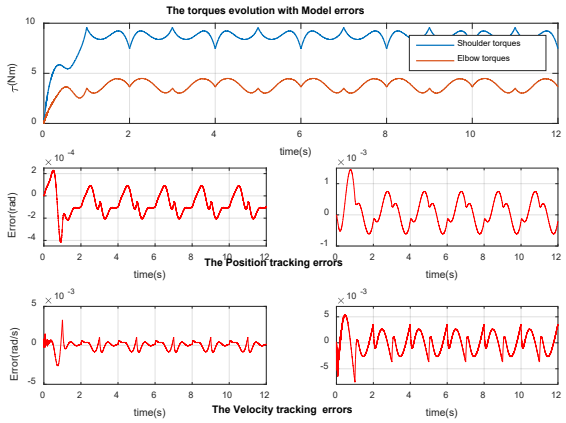


Figure 13. Torques evolution and tracking errorsevolution for both the joint's positions and velocities in the presence of model errors

Beyond the value of the error simulated in Figure 12, the necessary torques exceed the acceptable limitation, and the saturation is applied; for a saturation of +-50 Nm, the torques oscillate rapidly for about 0.5 seconds in order to cancel the error. Such oscillations are not suitable for the actuator, even for a limited amount of time, but even with limitations, the control law converges and cancels the tracking errors after 1 second. See Figure 14.

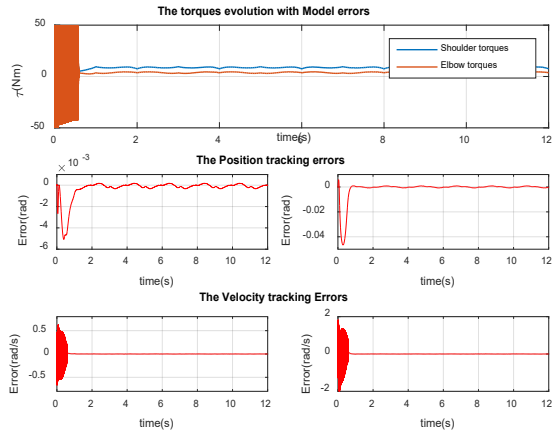


Figure 14. Torques and tracking errors evolution for more important errors, the torques are saturated, they oscillate rapidly during 0.5 s and cancel the errors after 1second.

8. CONCLUSIONS

In this paper we have presented an exoskeleton

design and control strategy in order to move the human arm for rehabilitation purpose. The brachialis muscle was to be trained to recover from disuse atrophy an adapted exoskeleton was designed using Autodesk Inventor 3D software and with two degrees of freedom which are independent joints parallel to the real human arm's joints. OpenSim software which is an open source software for simulating the dynamic of the musculoskeletal system is used to verify the ergonomy of the exoskeleton and to simulate its motions when attached to the human arm.

The API interface of OpenSim was used to connect with Matlab. The reference trajectories were calculated as an interpolation of several joint positions and are exported to OpenSim. The sliding mode control law is calculated in Matlab and tested with normal conditions. A robustness tests was ran according to initial errors and model parameters errors. The simulations show that Sliding mode control has very good performances when used with our model, even with a constrained state in terms of maximum actuator torques. The control converges rapidly so there is no need to use the sliding mode with finite time convergence because tuning of their parameters are difficult and they don't really give faster convergence compared to their complexity and difficulty of implementation.

9. REFERENCES

[1] Guan X., Linhong Ji., Wang R. Development of Exoskeletons and Applications on Rehabilitation. MATEC Web of Conferences 2016; 40: 02004, 2016. doi: 10.1051/matecconf/20164002004.

[2] Somer M. N., Nebras H G., Mays M. M. A. A review of lower limb exoskeletons. Innovative Systems Design and Engineering 2016; 7(11). ISSN 2222-1727 (Paper) ISSN 2222-2871 (Online).

[3] Sungjun Y., Jinyeong H., Hongchul K., Yongjin J. K., Technical analysis of exoskeleton robot, World Journal of Engineering and Technology 2018; 7(1): 68-79. ISSN: 2945-316X (Online).

[4] Nikhil, GVPG., Yedukondalu, G.; Rao, SS., Robotic exoskeletons: a review on development. International Journal of Mechanical and Production Engineering Research and Development 2019; 9(4): 529-542.

[5] Shing,. L. H., Quan X. S,. Exoskeleton robots for upper-limb rehabilitation: State of the art and future prospects. Medical Engineering & Physics 2012. 34(3): 61-268, ISSN:1350-4533, https://doi.org/10.1016/j.medengphy.2011.10.004.

[6] Pineda-Rico. Z. , Sanchez De Lucio. J. A., Martinez Lopez. F. J., Cruz, P. Design of an exoskeleton for upper limb robot-assisted rehabilitation based on co-simulation. Journal Of Vibroengineering 2016; 18(5): 3269-3278. ISSN 1392-8716.

[7] Saul. KR., Hu. X., Goehler. CM., Vidt. ME., Daly. M., Velisar. A., Murray. WM., Benchmarking of dynamic simulation predictions in two software platforms using an upper limb musculoskeletal model. Comput Methods Biomech Biomed Engin. 2015; 18(13):1445-58. doi: 10.1080/10255842.2014.916698. Epub 2014 Jul 4. PMID: 24995410; PMCID: PMC4282829.

[8] Zhihong. M, Paplinski. A. P., Wu. H. R., A robust MIMO terminal sliding mode control scheme for rigid robotic manipulators, *IEEE Transactions on Automatic Control* 1994, 39(12): 2464-2469. doi: 10.1109/9.362847.

[9] Hoffman H. (2022, 11 8). Guide To Recovering Hand Function After A Stroke. Retrieved from saebo.com: https://www.saebo.com/blog/guide-recovering-hand-function-stroke/

[10] KHAMAR. M., EDRISI. M., Designing a backstepping sliding mode controller for an assistant human knee exoskeleton based on nonlinear disturbance observer, Mechatronics 2018; 54: 121-132. ISSN 0957-4158. https://www.sciencedirect.com/science/article/pii/S0957415818301259.

[11] GORMAN, MSc,N. *Brachialis muscle*, viewed on 30 september 2022, retrieved from: https://www.kenhub.com/en/library/anatomy/brachialis-muscle

[12] GAMBHIRE, S.J., KISHORE, D.R., LONDHE, P.S. et al. Review of sliding mode based control techniques for control system applications. Int. J. Dynam. Control 2021; 9: 363–378. https://doi.org/10.1007/s40435-020-00638-7.

[13] Gambhire, S.J., Sri Kanth, K.S., Malvatkar, G.M. et al. Robust fast finite-time sliding mode control for industrial robot manipulators. Int. J. Dynam. Control 2019; 7: 607–618. https://doi.org/10.1007/s40435-018-0476-1.

[14] Khalil W. Dombre E. Modeling Identification and Control of Robots. CRC Press., 2002.

[15] Abooee A, Khorasani M, HAERI M, Free-chattering robust finite time tracking for connected double integrator nonlinear systems. In: Proceedings of the 4th International Conference on Control, Instrumentation, and Automation, Qazvin, Iran, 2016. doi: 10.1109/ICCIAutom.2016.7483178.

[16] ESKE, J. What to know about muscle atrophy, muscle, viewed on 02 september 2022, retrieved from: https://www.medicalnewstoday.com/articles/325316

[17] Muscle Atrophy: Causes, Symptoms Treatment, viewed on 02 september 2022, retrieved from: https://my.clevelandclinic.org/health/diseases/22310muscle-atrophy

[18] De Carlo. MS., Misamore. GW., Carrell. KR., Sell. KE., Rehabilitation of myositis ossificans in the brachialis muscle. Journal of Athletic Training 1992; 27(1): 76-9. PMID: 16558137; PMCID: PMC1317135.

[19] Delp. SL., Anderson. FC., Arnold. AS., Loan. P., Habib. A., John. CT., Guendelman. E., Thelen. DG., OpenSim: open-source software to create and analyze dynamic simulations of movement. IEEE Trans Biomed Eng 2007; 54(11):1940-50. doi: 10.1109/TBME.2007.901024. PMID: 18018689..

[20] Multibody Dynamics Concepts (Simbody) -OpenSim Documentation. viewed on 02 september 2022, retrieved from:https://simtk-confluence.stanford.edu:8443/display/OpenSim/Scripting+and+Development. https://simtk-confluence.stanford.edu:8443/pages/viewpage.action?pageId=9165421

[21] Seth. A., Hicks. JL., Uchida. TK., Habib. A., Dembia. CL., Dunne. JJ., et al. OpenSim: Simulating musculoskeletal dynamics and neuromuscular control to study human and animal movement. PLoS Comput Biol 2018; 14(7): e1006223. https://doi.org/10.1371/journal.pcbi.1006223.

[22] Introduction to the OpenSim API - OpenSim Documentation, viewed on 02 september 2022, retrieved from:

[23] https://simtk-confluence.stanford.edu:8443/display/OpenSim/Introduction+to+the+OpenSim+API
[24] Millard. M., Uchida. T., Seth. A., Delp. SL., Flexing computational muscle: modeling and simulation of musculotendon dynamics. J Biomech Eng 2013; 135(2): 021005. doi: 10.1115/1.4023390. PMID: 23445050; PMCID: PMC3705831.
[25] Vadim IU. Sliding Modes in Control and Optimization, Berlin: Springer-Verlag., 1992.
[26] Slotine, J.-J. E., Li. W. Applied Nonlinear control, Englewood Cliffs, N.J: Prentice Hall.,1991.
[27] How Inverse Dynamics Works - OpenSim Documentation. viewed on 02 september 2022, retrieved from: https:/simtk-confluence.stanford.edu:8443/display/Opensim/How+Inverse+Dynamics+Works.