University of São Paulo
São Carlos School of Engineering

Jhon Paul Feliciano Charaja Casas

Motor rehabilitation of human elbow flexion and extension movements using electrical stimuli

São Carlos
2023

Jhon Paul Feliciano Charaja Casas

Motor rehabilitation of human elbow flexion and extension movements using electrical stimuli

Dissertation presented to the São Carlos School of Engineering of the University of São Paulo in partial fulfillment of the requirements for the degree of Master of Science in Graduate Program in Mechanical Engineering.

Subject Area: Dynamics and Mechatronics

Advisor: Prof. Dr. Adriano A. G. Siqueira

São Carlos

2023

Ficha catalográfica elaborada pela Biblioteca Prof. Dr. Sérgio Rodrigues Fontes da EESC/USP com os dados inseridos pelo(a) autor(a).

Charaja Casas, Jhon Paul Feliciano
C469m Motor rehabilitation of human elbow flexion and extension movements using electrical stimuli / Jhon Paul Feliciano Charaja Casas; orientador Adriano Almeida Goncalves Siqueira. São Carlos, 2023.

Dissertação (Mestrado) - Programa de Pós-Graduação em Engenharia Mecânica e Área de Concentração em Dinâmica e Mecatrônica -- Escola de Engenharia de São Carlos da Universidade de São Paulo, 2023.

1. Deep reinforcement learning. 2. Functional electrical stimulation. 3. Elbow flexion and extension movements. I. Título.

Eduardo Graziosi Silva - CRB - 8/8907

FOLHA DE JULGAMENTO

Candidato: Bacharel **JHON PAUL FELICIANO CHARAJA CASAS.**

Título da dissertação: "Reabilitação motora de movimentos de flexão e extensão do cotovelo humano utilizando estímulos elétricos"

Data da defesa: 06/03/2023.

Comissão Julgadora	**Resultado**
Prof. Associado **Adriano Almeida Gonçalves Siqueira (Orientador)** (Escola de Engenharia de São Carlos – EESC/USP)	APROVADO
Prof. Dr. **Leonardo Abdala Elias** (Universidade de Campinas/UNICAMP)	APROVADO
Prof. Dr. **Roberto Santos Inoue** (Universidade Federal de São Carlos/UFSCar)	APROVADO

Coordenador do Programa de Pós-Graduação em Engenharia Mecânica:
Prof. Associado **Adriano Almeida Gonçalves Siqueira**

Presidente da Comissão de Pós-Graduação:
Prof. Titular **Carlos De Marqui Junior**

DEDICATION

With much appreciation to my family and all the people who supported me during the process.

ACKNOWLEDGMENT

to CNPq for the financial support in the development of my master's thesis.

ABSTRACT

Charaja, J. **Motor rehabilitation of human elbow flexion and extension movements using electrical stimuli**. 2022. Dissertation - São Carlos School of Engineering, University of São Paulo, São Carlos, 2022.

Clinical studies indicate that by performing repetitive exercises, patients gradually recover motor control of the upper limb. For this reason, diverse robotic rehabilitation systems have been developed to automatize the rehabilitation exercises, compensate for the lack of muscle strength and assist in the recovery of motor control. However, exoskeletons generate passive movements when the patients cannot coordinate their muscle contractions. Given this drawback, novel rehabilitation procedures use electrical pulses to generate muscle contraction and perform the exercise. The general objective of this work is to train an intelligent agent that determines the amplitude of electrical stimuli to generate controlled elbow flexion and extension movements. On the one hand, the intelligent agent will use deep reinforcement learning and soft actor-critic algorithm to determine the amplitude of the electrical pulses for the biceps and triceps. On the other hand, the reinforcement learning environment will use the OpenSim libraries to simulate how the biceps and triceps activation change the elbow's angular position. Finally, the performance of the intelligent agent to generate controlled elbow movements is evaluated in healthy volunteers with different arm characteristics.

Keywords: Deep Reinforcement Learning. Functional Electrical Stimulation. Elbow Flexion and Extension Movements.

RESUMO

Charaja, J. **Reabilitação motora de movimentos de flexão e extensão do cotovelo humano utilizando estímulos elétricos**. 2022. Dissertação - Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 2022.

Estudos clínicos indicam que, ao realizar exercícios repetitivos, os pacientes recuperam gradualmente o controle motor do membro superior. Por essa razão, foram desenvolvidos diversos sistemas de reabilitação robótica para automatizar os exercícios de reabilitação, compensar a falta de força muscular, e ajudar na recuperação do controle motor. Exoesqueletos, no entanto, geram movimentos passivos quando os pacientes não conseguem coordenar as suas contrações musculares. Dado esse inconveniente, novos procedimentos de reabilitação utilizam impulsos elétricos para gerar contrações musculares e realizar o exercício. O objetivo geral deste trabalho é treinar um agente inteligente que determina a amplitude dos estímulos elétricos para gerar movimentos controlados de flexão e extensão do cotovelo. Por um lado, o agente inteligente utilizará uma profunda aprendizagem de reforço e um algoritmo ator-crítico suave para determinar a amplitude dos impulsos elétricos para os bíceps e tríceps. Por outro lado, o ambiente de aprendizagem do reforço utilizará as bibliotecas OpenSim para simular como a ativação do bíceps e tríceps altera a posição angular do cotovelo. E por fim, o desempenho do agente inteligente para gerar movimentos controlados do cotovelo é avaliado em voluntários saudáveis com diferentes características de braço.

Palavras Chaves: Aprendizagem por Reforço Profundo. Estimulação elétrica funcional. Movimentos de Flexão e Extensão do Cotovelo.

List of Figures

List of Tables

Contents

Chapter 1

INTRODUCTION

1.1 Motivation

A stroke occurs when blood flow to the brain stops suddenly or gradually until patient's death (COUPLAND et al., 2017). The interruption in blood flow could be caused by blockage or rupture of a blood vessel (COUPLAND et al., 2017). The prolonged suspension of blood flow in the brain damages the nervous system through the death of neurons due to lack of oxygen (JOHNSON et al., 2016). This catastrophic event degenerates the nervous system and the generation of neural commands that lead to problems coordinating muscle contraction and controlling the body movements (CANNING; ADA; O'DWYER, 2000).

The Global Burden of Disease and Injury study, conducted in 2015, mentions that stroke is the second most common cause of death for people worldwide, with more than six million deaths in that year (WANG et al., 2016) and the second most common cause of permanent disability with more than one-hundred million people affected by stroke sequelae worldwide in that year (KASSEBAUM et al., 2016). In Brazil, stroke is the third cause of death, with more than thirty in-hospital deaths yearly (DANTAS et al., 2019). Likewise, from 2019 to 2016, more than one million stroke hospitalizations were registered in health centers in Brazil (DANTAS et al., 2019).

Patients with sequelae after a stroke usually cannot coordinate their muscles contraction to generates functional movements (DEWALD et al., 1995; CANNING; ADA; O'DWYER, 1999). This condition encourages sedentary behaviors that gradually reduce the life quality of the patients (FITZSIMONS et al., 2022). Reduced elbow range of motion drastically reduces upper limb performance for eating, dressing, and personal care activities (GROOT et al., 2011). For this reason, in occupational therapy, patients perform repetitive exercises that involve flexion and extension of the

elbow (LÓPEZ; AYUSO, 2010). An exercise in the feeding category is based on placing a spoon in the patient's hand and making elbow movements to move the spoon closer to and away from the patient's mouth. In this exercise, the elbow flexion movement brings the spoon closer and the elbow extension movement moves the spoon away from the patient's mouth.

Clinical studies indicate that by performing repetitive exercises, patients gradually recover motor control of the upper limb (FRENCH et al., 2016). For this reason, diverse robotic rehabilitation systems have been developed to automatize the rehabilitation exercises, compensate for the lack of muscle strength and assist in the recovery of motor control (SHEN; FERGUSON; ROSEN, 2020). However, the exoskeleton will generate passive movements when the patients cannot coordinate their muscle contractions (LOOZE et al., 2016; GORGEY, 2018). Given this drawback, other rehabilitation procedures use electrical pulses to generate muscle contraction and perform the exercise (PECKHAM; KNUTSON et al., 2005). At the end of this innovative rehabilitation therapy, patients increased motor control of the upper limb to perform activities of daily living (HOWLETT et al., 2015).

1.2 State of the Art

Functional electrical stimulation is a promising rehabilitation technique that uses skin-surface electrodes and electrical stimuli to generate muscle contractions (PECKHAM; KNUTSON et al., 2005). However, the upper limb muscles excited by electrical stimulation generate a dynamic system with challenging characteristics for motion control algorithms. On the one hand, the muscles present a nonlinear relationship between contraction force and electrical stimulus (MILLARD et al., 2013). On the other hand, the electrical pulses are applied to the surface of the skin and not directly to the muscle, so there are unmodeled dynamics that change with the physical characteristics of each patient (MAFFIULETTI, 2010). Finally, throughout the rehabilitation sessions, the response of the patient's muscles changes, and with it, the value of the muscle parameters (MAFFIULETTI, 2010).

The control methods should address the muscle complex behavior to achieve a good performance generating controlled elbow movements. Most of the reviewed

works stimulate the biceps and triceps to generate elbow extension and flexion motions. Kitamura, Sakaino e Tsuji (2015) used the Proportional-Integral-Derivative (PID) control method to computes the amplitude of the electrical pulses. The results showed a regular tracking performance due to slow response and no adaptation from the control method.

Since model accuracy limits the performance of linear control methods, researchers used nonlinear control approaches to improve the results. Barbouch et al. (2017) compared the Sliding Mode (SM) with the Proportional-Derivative (PD) control method to perform elbow movements. Authors used a Hill-based muscle model to describe the muscle forces during elbow movements; more details of Hill muscle model in (WINTERS, 1990). Likewise, authors used those muscle equations to design both control methods. The performance of the control methods was evaluated in a MATLAB simulation that did not cover uncertainties and time-varying parameters. The results indicated that SM is lightly better than PD in generating controlled elbow flexion and extension movements.

Despite the good results in simulation environments, successful implementation in the real world should overcome the variation of the dynamic model between each patient. For this reason, some authors use machine learning methods to estimate the muscle model and its parameters. On the one hand, Wolf, Hall e Schearer (2020) used a Gaussian process regression to estimate joint torques when muscles are exited with electrical pulses. The controller uses the muscle model and a proportional-integral formulation to compute the muscle stimulation commands to achieve the desired position. The experimental setup considers a robotic device to support the patient arm and to guarantee elbow motions in the transversal plane. The experimental results (95 volunteers) indicate a maximum error of 6 cm with low frequency oscillations; which were acceptable metrics to perform daily living activities. On the other hand, Koushki et al. (2021) uses deep neural networks and reinforcement learning method to control the elbow position without a mathematical model. Moreover, the authors used Deep Deterministic Policy Gradient (DDPG) to compute the optimal neural network parameters. Deep reinforcement learning (DRL) method learned the muscle behavior through interaction with a simulation environment that considers a nonlinear model with time-varying parameters. The trained model achieved a position root-squared-error of 2° for trajectory tracking

task in the simulator. Similarly, Febbo et al. (2018) used DRL with Proximal Policy Optimization (PPO) to generate controlled elbow movements. The experimental results (10 volunteers) indicates that DRL with PPO outperforms PID for trajectory tracking tasks and adaptation to system dynamics. Finally, Haarnoja et al. (2018) developed a novel reinforcement learning algorithm called Soft Actor-Critic that combines the best of DDPG and PPO. Wannawas, Shafti e Faisal (2022) uses SAC to generate elbow flexion and extension movements using only the biceps muscle. The authors used inertial units of measurement to estimate the elbow's angle and the Rehamove1 electrical stimulator. The experimental results indicate that DRL with SAC is an excellent method to generate controlled elbow flexion and extension movements.

1.3 Objective

The general objective of this work is to train an intelligent agent that determines the amplitude of electrical stimuli to generate controlled elbow flexion and extension movements. As a result, rehabilitation exercises will be performed by the patient's muscular contractions rather than an external mechanical system. The proposed rehabilitation system consists of three elements. First, a RehaMove3 brand electrical stimulator safely generates electrical stimuli. This device uses two adhesive electrodes to send electrical stimuli to each target muscle. Second, a mechanical system with an incremental encoder that is placed on the patient's elbow to measure the angle of flexion. Third, an intelligent agent that computes the amplitude of each electrical pulses to generate the desired movement. In addition, the rehabilitation system has been designed for patients with muscle weakness and motor coordination problems. Consequently, the rehabilitation system is not recommended in patients with muscle spasms or pain during elbow joint movement. Finally, some specific objectives are listed

- Define the reward system to indicate the ideal behavior of the intelligent agent.
- Create a reinforcement learning environment that simulate the muscle behavior under electrical stimuli and return the necessary observations to train the intelligent agent.

- Implement the soft actor-critic algorithm to compute the optimal neural network parameters.
- Build a mechanical system to place an encoder that will measure the angular position of the elbow.
- Implement a Kalman filter to estimate the angular position and velocity of the elbow.
- Carry out experimental tests on volunteers to evaluate the performance of the intelligent agent generating controlled elbow flexion and extension movements

1.4 Structure of the Work

This work comprises five chapters. The first chapter describes the problem to be addressed, presents the previous works that have focused on generating controlled elbow movements with electrical stimulation and the objective of the work. The second chapter shows biomechanics concepts and machine learning theory that are used in the development of this work. The third chapter presents the methods that will be used for the development of this work. The fourth chapter presents results of the intelligent agent to generate controlled movements of the elbow. Finally, conclusions and future work is presented.

Chapter 2

BACKGROUND

This chapter presents the concepts of biomechanics, electronic devices and machine learning theory that will be used for the development of this work. First, the main tools to describe the movement of a part of the body are described. Following this, the muscles involved in the generation of flexion and extension movements of the elbow joint are described. Second, technical details of the RehaMove3 electrical stimulation are described. Finally, the mathematical foundations of the reinforcement learning method are described, as well as its variant with deep learning and soft actor-critic algorithm.

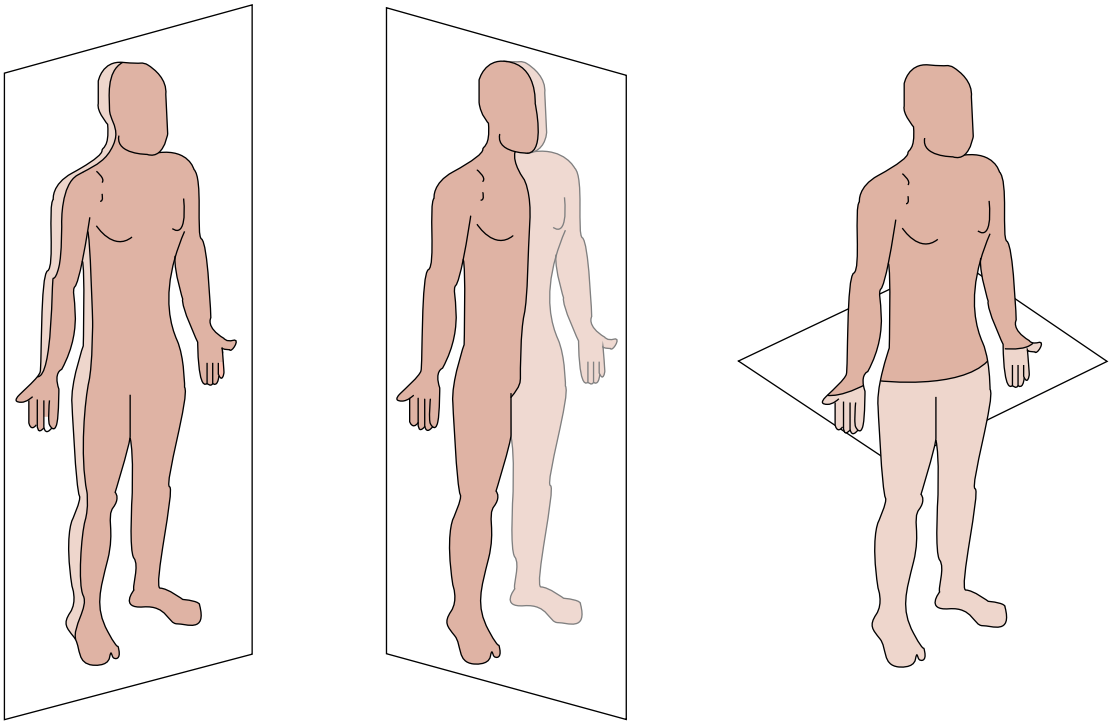
2.1 Basic Concepts of Biomechanics

Biomechanics studies the dynamics of biological systems; how the musculoskeletal model generates movements by applying excitation signals (e.g. external forces and electrical stimuli) (KNUDSON; KNUDSON, 2007). Biomechanics knowledge can be used to improve an athlete's performance and rehabilitate a physical injury (KNUDSON; KNUDSON, 2007). Anatomical planes and anatomical position are used to describe the position and axis of motion of a body part (HAMILL; KNUTZEN, 2006).

2.1.1 Anatomical Planes

Anatomical planes are imaginary two-dimensional sections used to separate parts of the human body (HAMILL; KNUTZEN, 2006; JARMEY, 2008). These two-dimensional planes establish spatial references that facilitate the description, location and study of the parts of the human body. The anatomical planes are sagittal, frontal and transverse (HAMILL; KNUTZEN, 2006; JARMEY, 2008). On the one hand, the sagittal plane is a vertical plane that divides the human body into the right half and left half. On the other hand, the frontal plane is a vertical plane that

divides the human body into the front and back. Finally, the transversal plane is a horizontal plane that divides the human body into the upper part and the lower part (HAMILL; KNUTZEN, 2006; JARMEY, 2008). Figure 2.1 shows the three anatomical planes of the human body.

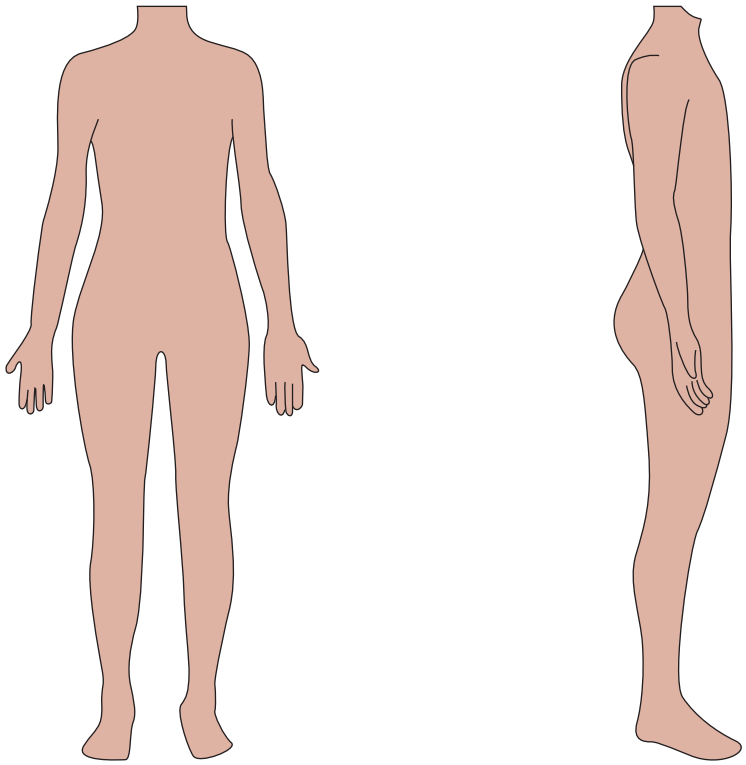


(a) frontal (b) sagittal (c) transversal

Figure 2.1: Anatomical planes of the human body.

2.1.2 Anatomical Position

Anatomical position is the reference body configuration to describe the relative position of human body parts (HAMILL; KNUTZEN, 2006; JARMEY, 2008). This reference position is defined as a standing human body with limbs hanging along the trunk and open hands directed forward (HAMILL; KNUTZEN, 2006; JARMEY, 2008). The body position described is illustrated in Figure 2.2.



(a) frontal (b) profile

Figure 2.2: Anatomical position of the human body.

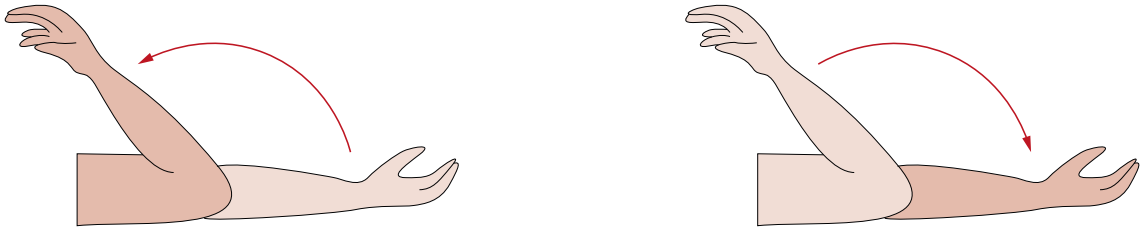
2.1.3 Anatomical Movements

Anatomical movements are the group of activities performed by each joint in the body. These movements can be: flexion and extension, abduction and adduction, pronation and supination, etc. (HAMILL; KNUTZEN, 2006; JARMEY, 2008). Likewise, each anatomical movement activates a sequence of muscles around the joint.

2.2 Anatomical Movements of the Elbow

The human elbow connects the humerus with the proximal ends of the ulna and radius. This joint has one degree of freedom and can perform the anatomical movement of flexion and extension (STAUGAARD-JONES, 2014). On the one hand, the elbow flexion movement decreases the angle between the bones connecting to the elbow joint. On the other hand, the elbow extension movement increases the angle

between the bones that connect to the elbow joint. Figure 2.3, shows the flexion and extension movement of the elbow.



(a) flexion (b) extension

Figure 2.3: Anatomical movements of the elbow joint.

The biomechanics of a human arm indicate that three muscles are activated to flex the arm and one muscle to extend it (STAUGAARD-JONES, 2014). On the one hand, the brachialis and biceps brachii are activated to perform the elbow flexion movement. On the other hand, the triceps brachii muscle is activated to perform the elbow extension movement. Figure 2.4 shows the muscles that are activated during elbow flexion and extension movements (STAUGAARD-JONES, 2014).

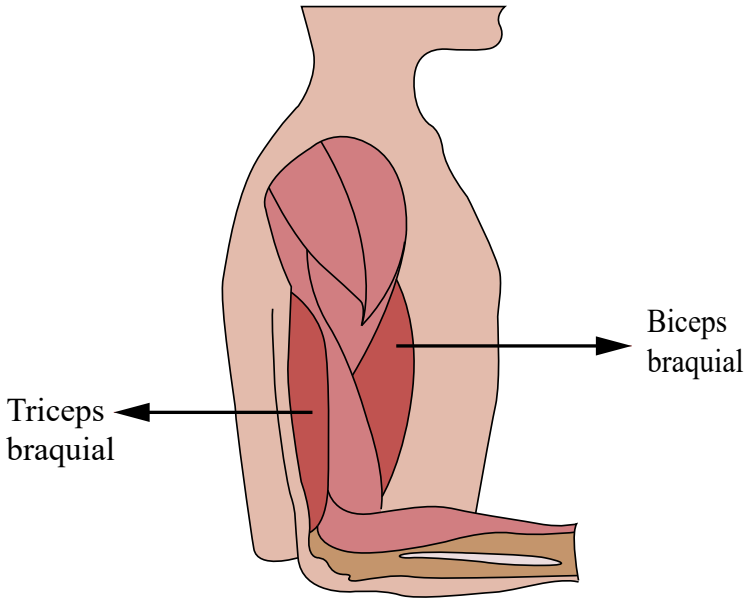


Figure 2.4: Arm muscles responsible for generating elbow flexion and extension movements.

2.3 RehaMove3

The Hasomed GmbH company developed the RehaMove3 electrical stimulator to assist in electrical stimulation therapies for patients with problems controlling their body movements (HASOMED GmbH, 2022). RehaMove3 generates electrical pulses considering the safety protocols for a person; likewise, the device requires placing two electrodes on the surface of the skin to send electrical pulses to each muscle. The user manual indicates that RehaMove3 can generates electrical pulses with frequency from 1 Hz to 500 Hz, duration from 10 us to 4000 us and maximum amplitude of 130 mA (HASOMED GmbH, 2022). Finally, the electrical stimulator with its skin-surface electrodes are shown in Figure 2.5.

Figure 2.5: RehaMove3 electrical stimulator with electrodes.

2.4 Reinforcement Learning

The reinforcement learning method trains an agent to take the optimal sequence of decisions (SUTTON; BARTO, 2018). The learning process uses positive and negative reinforcement to increase or decrease the probability of choosing an specific action for a given state. In this sense, the agent learns, through an iterative process, to

make the sequence of decisions that maximizes the amount of positive reinforcement that he will receive (SUTTON; BARTO, 2018).

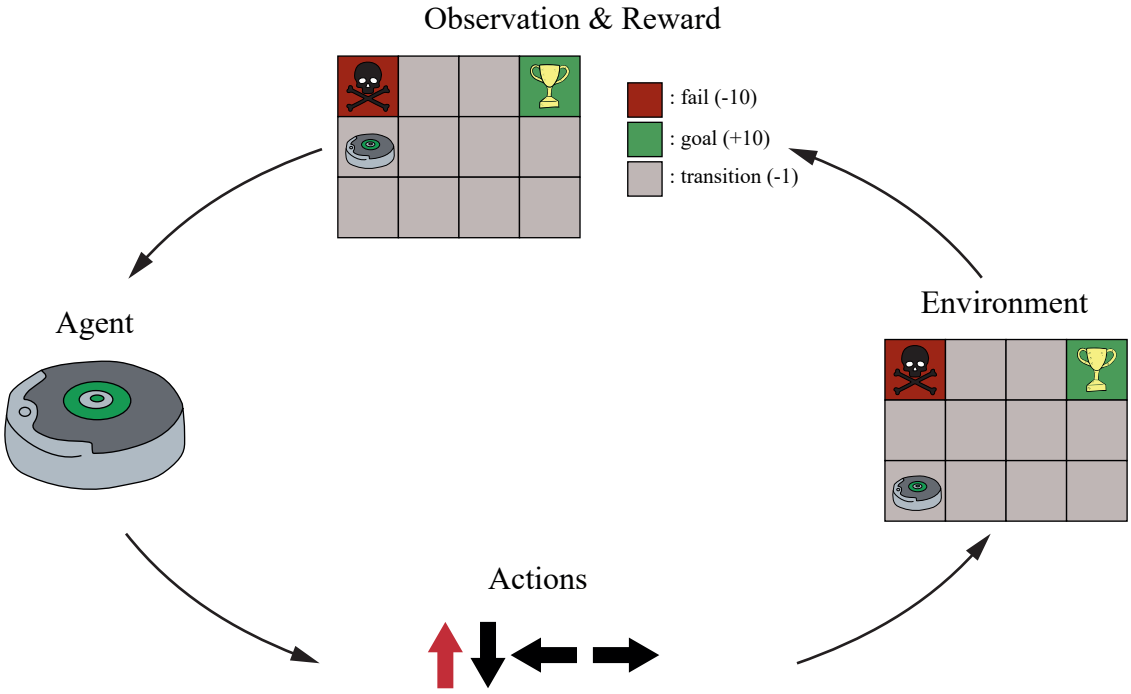


Figure 2.6: Reinforcing learning framework for the navigation control application of a mobile robot. The available actions are go up, down, right and left. Finally, the green and red blocks represent the goal and fail condition.

2.4.1 Framework

The framework of reinforcement learning comprises four elements: (i) agent, (ii) actions, (iii) environment, and (iv) observation and reward (SUTTON; BARTO, 2018). Figure 2.6 describes the reinforcement learning framework for the navigation control application of a mobile robot. First, an agent who makes decisions based on the reward and punishment that he will receive. Second, action space are all the available actions that the agent could use to interact with the environment and generate changes. Third, the environment where the agent lives and interact. Fourth, observations that describe the new state of the agent and its environment after applying an action; as well as the reward associated with the transition of states. Finally, this

process will be repeated several times until the agent learns a successful strategy to interact with the environment and maximize the reward.

2.4.2 Policy

The agent's strategy is called policy ($\pi(a|s)$) and indicates which action (a) the agent should choose in the current state (s) (SUTTON; BARTO, 2018). Policies have four fundamental characteristics, which are assigned according to the efficient use of the data collected and the probabilistic behavior of the decision-making process. On the one hand, the policy is offline if the method reuses much of the data collected; otherwise, the policy is online. On the other hand, the policy is stochastic if the method considers probabilistic processes during decision-making; otherwise, it is deterministic (SUTTON; BARTO, 2018).

In general, each type of policy has advantages and disadvantages. On the one hand, online policies present high rates of convergence but require the generation of new training data for each update (SINGH et al., 2000). These characteristics limit its implementation in systems that allow constant interaction with the environment. On the other hand, offline policies use a memory buffer that allows the reuse of collected data; however, it usually shows many oscillations during training (THOMAS; BRUNSKILL, 2016). Finally, the objective of reinforcement learning algorithms is to find the optimal policy that maximizes the cumulative sum of rewards.

2.4.3 Value Function

The reward indicates how good or bad the agent's decision was for a given state (s_t). However, just considering the immediate reward (r_t) does not guarantee that the agent will maintain a good performance until the episode ends. For this reason, the standard way considers the discounted sum of all the rewards that will be obtained starting in state s_t and then following the policy until the episode ends (SUTTON; BARTO, 2018). Besides, the cumulative sum of rewards can have two meanings for the agent training: (i) state value function and (ii) state-action value function.

State Value Function

The state value function indicates how good or bad it is to be in the state s_t and then follow the policy π. This can be computed as

$$V^{\pi}(s_t) = \mathbb{E}_{\pi}\left[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + ...|s_t = s\right]$$

where r_t represents the immediate reward at time t, $\gamma \in [0, 1]$ is an hyperparameter that indicates how important future rewards are going to be for the agent training, and s_t denotes the initial agent's state. Likewise, the state value function can be computed for iterative processes as $V^{\pi}(s_t) = E_{\pi}\left[r_t + \gamma V^{\pi}(s_{t+1})|s_t = s\right]$; in this formulation, the state value function is separated in the immediate reward and the state value function of the next state.

State-Action Value Function

The state-action value function indicates how good or bad it was to take action a_t for state s_t and then follow the policy π. This can be computed as

$$Q^{\pi}(s_t, a_t) = \mathbb{E}_{\pi}\left[r_t^a + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \gamma^4 r_{t+4} + ...|s_t = s, a_t = a\right],$$

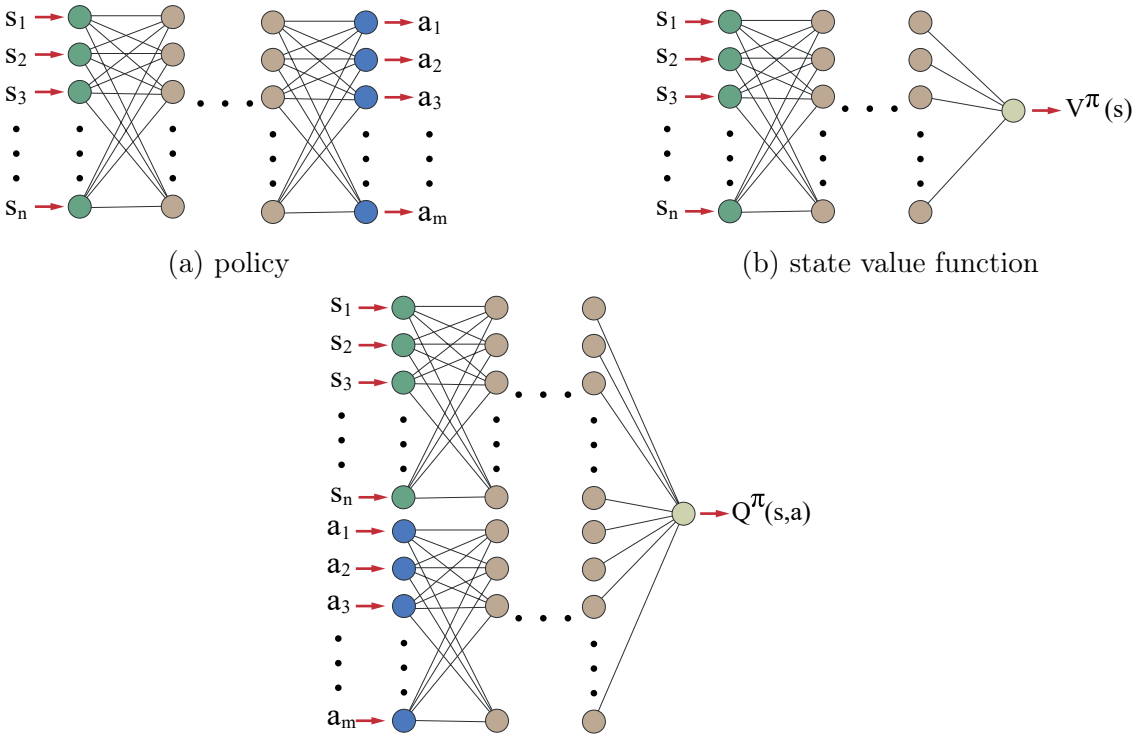
where r_t^a represents the immediate reward after choosing action a_t at time t, $\gamma \in [0, 1]$ is an hyperparameter that indicates how important future rewards are going to be for the agent training, and s_t denotes the initial agent's state. Likewise, the state-action value function can be computed for iterative processes as $Q^{\pi}(s_t, a_t) = E_{\pi}\left[r_t + \gamma Q^{\pi}(s_{t+1}, a_{t+1})|s_t = s, a_t = a\right]$; in this formulation, the state-action value function is separated in the immediate reward and the state-action value function of the next state and action.

2.5 Deep Reinforcement Learning

The main disadvantage of reinforcement learning is the exponential increase in computational resources due to the number of possible states. This characteristic makes

its application impossible in continuous systems that have infinite possible states. This problem encourages the use of approximation functions (e.g deep neural networks) to generalize the information in a system with a large number of possible states. The combination of reinforcement learning with deep learning is known as deep reinforcement learning (DRL) (FRANÇOIS-LAVET et al., 2018).

DRL represents the value functions and policy with deep neural networks, and use the environment observations as input to estimate value of an state, state-action and predict the best action. Figure 2.7 describes the policy and value functions parameterized with a deep neural networks. DRL techniques optimize the neural network parameters with two objectives: (i) find the optimal policy that maximizes the cumulative sum of rewards and (ii) reduce the approximation error of the state-action value function and then choose the action that guarantees the maximum reward.



(a) policy

(b) state value function

(c) state-action value function

Figure 2.7: Reinforcement learning elements parameterized with deep neuronal networks. The quantify of actions and observations are represented with m and n, respectively.

2.5.1 Policy Gradient

Policy gradient algorithms optimize the parameters of the policy neural network to maximize the cumulative sum of rewards (SUTTON; BARTO, 2018). In a general way, policy gradient algorithms formulate an objective function based on an expression of the reward accumulated along the trajectory; then, they estimate the gradient's value and use the gradient ascent method to modify the parameters of the policy neural network. The objective functions can be defined as (SUTTON; BARTO, 2018)

$$L(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^{T-1} r_t \right],$$

where θ represent the parameters of the policy neural network, π_θ denotes the policy parameterized as a function of θ, $\tau = (s_0, a_0, r_0, s_1, a_2, \cdots, s_T)$ represents the trajectory generated by following the policy π_θ, s_T is a general terminal state and r_t is the immediate reward at time t.

The gradient function can be computed as (SUTTON; BARTO, 2018)

$$\nabla_\theta L(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^{T-1} r_t \sum_{t=0}^{T-1} \nabla_\theta \ln \pi_\theta(a_t|s_t) \right],$$

where ∇_θ denotes the gradient operator with respect to θ and $ln(\cdot)$ denotes the natural logarithm.

2.5.2 Soft Actor-Critic

The soft actor-critic (SAC) is a policy gradient algorithm that uses the maximum entropy framework to maximize the accumulative sum of rewards while encouraging high exploration (HAARNOJA et al., 2018). For this purpose, SAC considers the entropy of the policy distribution in the objective function; high entropy implies almost the same probability for each action and random behavior that encourages exploration. SAC formulates the objective function as (HAARNOJA et al., 2018)

$$L(\pi_\theta) = \sum_{t=0}^{T} \mathbb{E}_{(s_t, a_t) \sim \rho_{\pi_\theta}} \left[r(s_t, a_t) + \alpha \mathcal{H}(\pi(\cdot|s_t)) \right],$$

where $r(s_t, a_t)$ represents the reward for being in state s_t and choosing the action a_t, α indicates the influence of entropy in agent training and $\mathcal{H}(\pi_\theta(\cdot|s_t)) = \mathbb{E}_{a\sim\pi_\theta(\cdot|s_t)}\left[-\log\pi_\theta(a|s_t)\right]$ represent entropy of the policy distribution.

SAC algorithm comprises three mean mechanism with specific objectives. First, the actor element takes decisions; so it represents the policy and indicates which action a_t the agent should take in the state s_t. Second, the critic element evaluates the agent decision; so it represents the state-action value function and indicates how good or bad the action a_t was for the state s_t. Finally, the soft transfer learning between state-action neural networks.

The actor element uses a deep neural network to computes the mean (μ_θ) and standard deviation (σ_θ) that will be use to compute the policy distribution $\pi(a|s, \mu_\theta, \sigma_\theta)$. Thus, the action is computed as

$$a = \tanh(\mu_\theta + \eta\sigma_\theta),$$

where $\tanh(\cdot)$ denotes the hyperbolic tangent function and η is random sample from a Gaussian distribution with $\mu = 0$ and $\sigma = I$. Finally, loss function of the policy network is computed as

$$J_\pi(\theta) = \mathbb{E}_{s\sim\mathcal{D}}\left[\mathbb{E}_{a\sim\pi_\theta(\cdot|s)}\left[\alpha\log\pi_\theta(a|s) - \min_{i=1,2} Q_{\phi,i}(s,a)\right]\right], \tag{2.1}$$

where $\mathcal{D}$ denotes the collected state transitions (s_t, a_t, r_t, s_{t+1}) and Q_ϕ represents the a neural network that predicts the state-action value function.

The critic element uses four deep neural networks to evaluate how good or bad was the the action a_t for the state s_t. On the one hand, two deep neural networks are used to predict the state-action value function and are called predict networks, $Q_{\text{predict},\phi}$. On the other hand, other two deep neural networks are used to estimate the real state-action value function and are called target networks, $Q_{\text{target},\phi}$. Likewise, every certain number of trajectories, the parameters of target networks will be updated using the following equation

$$\phi_{\text{target},i} := \rho\phi_{\text{target},i} + (1-\rho)\phi_{\text{predict},i},$$

where $i = 1, 2$ represent the number of predict and target deep neural network, ρ is a hyperparameter that set the transfer learning from the networks. Finally, loss function of the predict networks is computed as

$$J_Q(\theta_i) = \mathbb{E}_{(s_t, a_t, s_{t+1}) \sim \mathcal{D}} \left[\left(Q_{\theta,i}(s, a) - y \right)^2 \right], \text{ with} \tag{2.2}$$

$$y = \begin{cases} r & \text{if } s_t \text{ is terminal state} \\ r + \gamma \left(\min Q_{\text{target,i}}(s_{t+1}, a_{t+1}) - \alpha \log \pi_\theta(a_{t+1} | s_{t+1}) \right) & \text{for other cases} \end{cases}$$

2.6 OpenSim

OpenSim is an open-source software to simulate the dynamic behavior of musculoskeletal models (DELP et al., 2007). Hence, users can analyze muscle activation throughout the desired movement or the position and velocity of the links when the muscles are activated. Opensim's musculoskeletal models are very accurate and there are libraries for C++, MATLAB and Python (SimTK, 2023, January 31.).

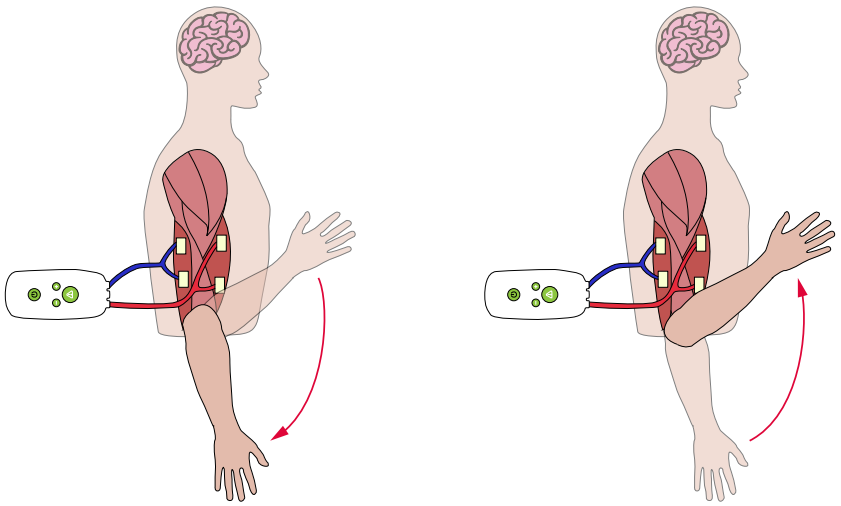
Chapter 3

METHODOLOGY

This chapter presents the methods that will be used for the development of this work. First, considerations to adequately stimulate the biceps and triceps muscles, location of the electrodes, and configuration of the electrical stimulator. Second, the reinforcement learning framework for the application of generating controlled elbow flexion and extension movements using electrical stimuli. Third, considerations for implementing the system on the arm of a volunteer.

3.1 Setup of the Rehamove3 electrical stimulator

The RehaMove3 electrical stimulator will generate elbow flexion and extension movements in the sagittal plane. For this purpose, two electrodes will be placed on the biceps and triceps; each electrode will be placed at the beginning and end of the targeted muscle, as shown in Figure 3.1.



(a) extension (b) flexion

Figure 3.1: Configuration of the experimental setup to generate the elbow flexion and extension movements with the Rehamove3 electrical stimulator.

The device uses electrical pulses to generate the contraction of the muscles. The electrical pulses are defined with three parameters: (i) frequency, (ii) width, and (iii) amplitude; Figure 3.2 graphically shows the three parameters of the electrical pulses. These three parameters set the muscle contraction level, movement consistency, and user comfort.

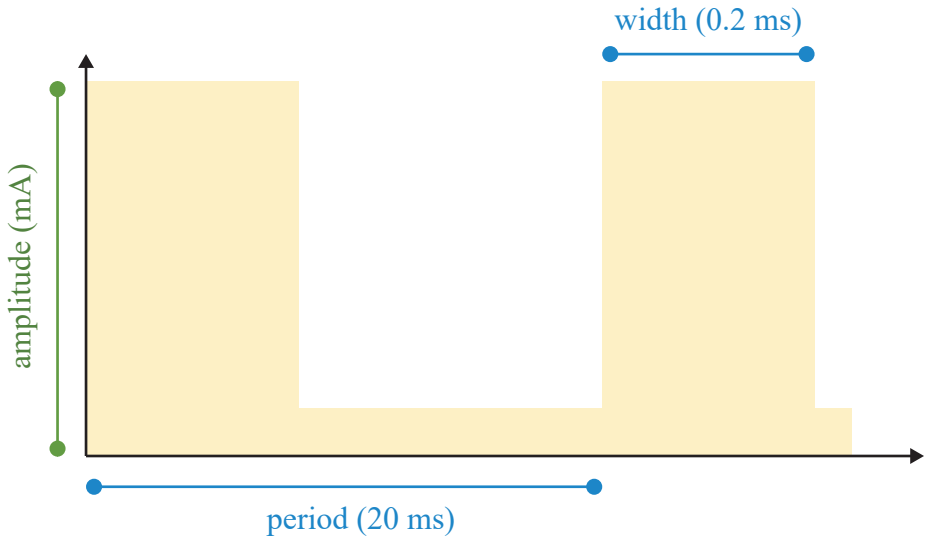


Figure 3.2: Main parameters of the electrical pulses generated by Rehamove3. Likewise, the period and width are the recommended values from the user manual of Rehamove3. Image adapted from (HASOMED GmbH, 2022).

The frequency of the electrical pulses is related to the oscillations of the generated force and nervous contractions (i.e., twitch) (UCHIDA; DELP, 2021). An electrical pulse generates a peak force that decays over time until it reaches 0. The goal of using a sequence of pulses is for the peak forces to accumulate and generate a constant signal (UCHIDA; DELP, 2021). Figure 3.3 shows muscle force generated for different stimulation frequencies. So, for low frequency (5 Hz - 10 Hz), the generated force presents high amplitude oscillations, for medium frequency (20 Hz - 40 Hz) the generated force presents low amplitude oscillations and for high frequency (50 Hz - 500 Hz) the generated force presents oscillations that can be negligible.

The above suggests that applying high-frequency pulses to generate smooth and consistent movements would be ideal. However, the level of user discomfort (i.e., pain) increases with the frequency; the same happens with the electrical pulse width. For this reason, in this work, the frequency and width of each electric pulse

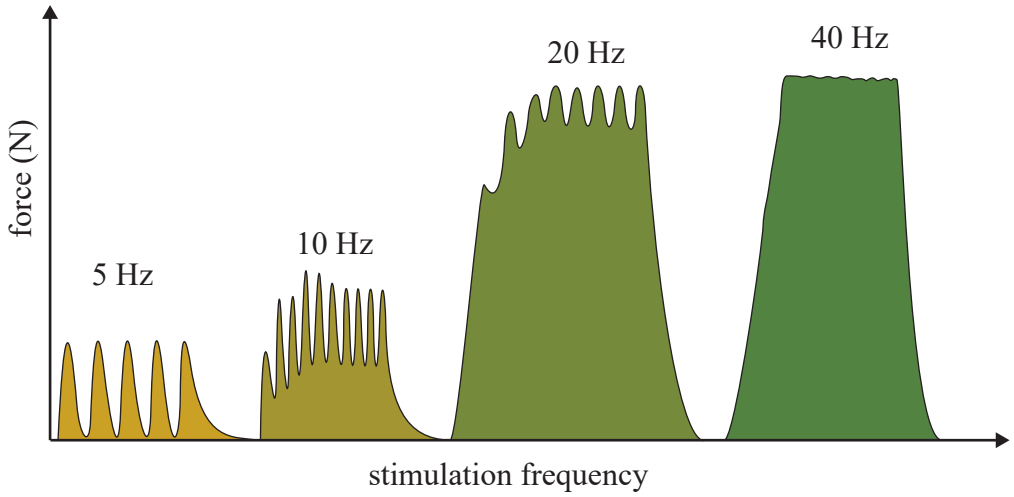


Figure 3.3: Muscle force behavior with respect to stimulation frequency. Image adapted from (UCHIDA; DELP, 2021) and (WAKELING et al., 2012).

will be chosen based on the recommendations of previous works in the area. Finally, the Rehamove3 user manual recommends using a frequency of 50 Hz, and a pulse width of 200 us was selected based on the user's discomfort level; the level of muscle contraction will be controlled by the amplitude of the electrical pulse (HASOMED GmbH, 2022). Hence, the objective of the reinforcement learning agent is to determine the amplitude of the electrical pulses to generate the controlled movements of elbow flexion and extension.

3.2 Reinforcement learning framework

Reinforcement learning algorithms do not need a detailed description of the environment (i.e., a mathematical model) or the task to be solved. In general, the intelligent agent learns iteratively how the environment around it works and what skills it needs to solve the assigned task. To do this, the agent interacts in a simulation environment where they can make decisions and observe the effect of their decisions. Similarly, a reward system encourages the sequence of decisions that best solves the assigned task.

Figure 2.6 describes the reinforcement learning framework for the application of generating controlled elbow flexion and extension movements with electrical

pulses. First, the intelligent agent will use the SAC algorithm to learn to make decisions because it is a start-of-art reinforcement learning algorithm. Second, Rehamove3 electrical stimulator has two outputs, and the agent will determine the normalized muscle activation (from 0 to 1) for the biceps and triceps. Third, OpenSim software will be used to create a reinforcement learning environment that allows the agent to learn how the level of muscle contraction of the biceps and triceps affects the position of the elbow angle.

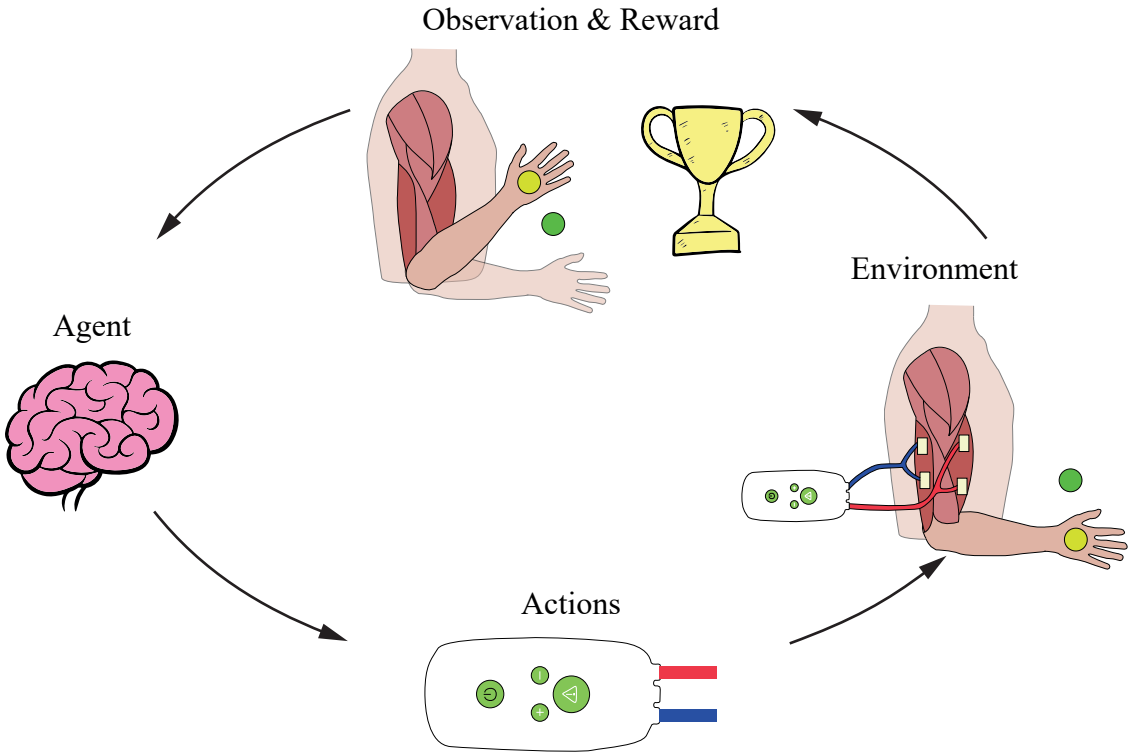
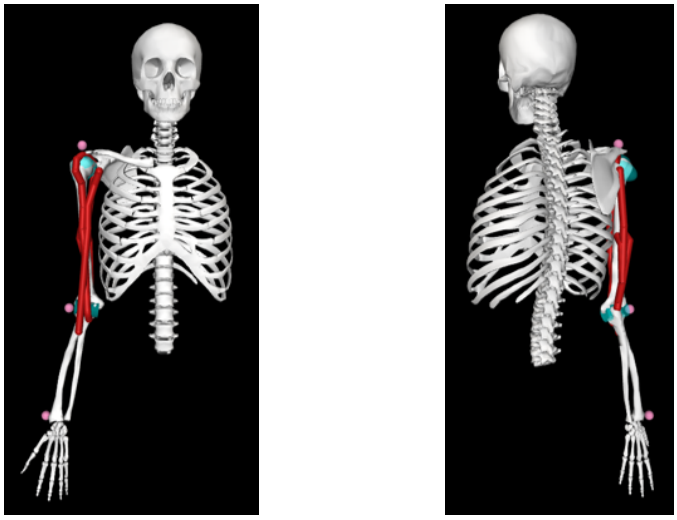


Figure 3.4: The reinforcement learning framework for the application of generating controlled elbow flexion and extension movements by electrical pulses. The agent's actions are the amplitude of each electrical stimulus for the biceps (red) and triceps (blue) muscles. Finally, green and yellow circles represent the desired and measured position.

3.2.1 Reinforcement learning environment

OpenSim has available musculoskeletal models of different parts of the human body (e.g., wrist, leg, arm). The `arm26.osim` is a musculoskeletal model which describes the upper right human arm with the shoulder and elbow joints and biceps and triceps

muscles (SimTK, 2023, January 31.). Figure 3.5 shows a front and back view of the `arm26.osim` musculoskeletal model.



(a) front view

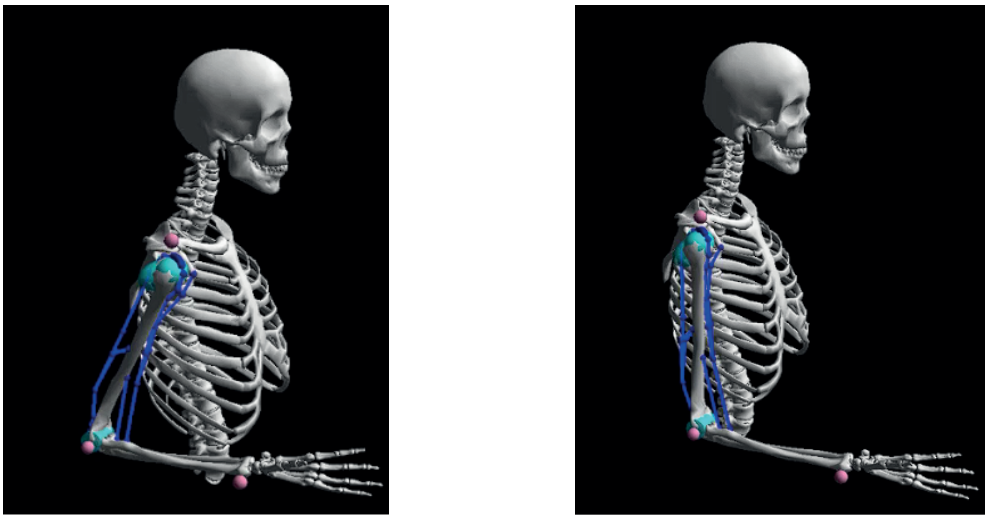
(b) back view

Figure 3.5: Human upper right limb model in OpenSim.

OpenSim and the `arm26.osim` model is used to create a reinforcement environment that allows the agent to understand how the activation of each muscle affects the elbow's angular position. First, the shoulder joint is fixed to concentrate all the movement in the elbow; and avoid weird arm configurations, as shown in Figure 3.6. Second, configure the model to activate all the biceps with one signal and all the triceps with another signal because Rehamove3 has two outputs (channel red and blue). Third, add human joint limits to the musculoskeletal model. Finally, the most relevant information for the agent to learn to perform controlled elbow flexion and extension movements are (i) elbow position, (ii) elbow angular velocity and (iii) muscle activation. Therefore, these measurements are considered the agent's observations during his training; likewise, these measurements will be used to calculate the reward after each action.

3.2.2 Reward system

The reward system guides the intelligent agent's learning and establishes the most important skills to solve the assigned task. The formulation of the reward system



(a) free (b) fixed

Figure 3.6: The shoulder joint's effect on the arm's final configuration; in both cases, the same muscle activation is used.

should be general to avoid limiting the agent's curiosity; similarly, the reward system should contain a performance metric based on the assigned task. For the application of generating controlled elbow flexion and extension movements, the agent must reduce the angle between the desired and measured position; while performing safe movements for the user's arm. For this reason, an element of the reward system must be a function of angular position error, β. Hence, an exponential function is used to avoid positive and negative reward values without a defined range; in this way, the reward system will generate values between 1 and 0. Likewise, the second element of the reward system penalizes elbow speed to avoid high speeds and the third element penalize high muscle activation to encourage low-energy movements.

The desired behavior is defined with the following reward system

$$R(\beta, \dot{\theta}) = \alpha_1 \exp\left(-\left(\frac{\beta}{\sqrt{2}\sigma}\right)^2\right) - \alpha_2\dot{\theta} - \alpha_3 u, \tag{3.1}$$

where α_1, α_2, α_3 are weighting coefficients, β is the angular position error, $\dot{\theta}$ is the elbow's angular velocity, $u = \sum_{i=1}^{2} a_i$ represent the total muscle activation and σ represent the dispersion of data.

The parameter σ is related to the reward (R) and the angular position error (β). When $\beta = 3\sigma$, the reward is approximately 0.1; which represents 10% of the maximum reward. Hence, the value of σ is defined as $\frac{\text{max error}}{3}$. Figure 3.7 describes the reward system and the objective of the reinforcement learning agent for the application of performing controlled elbow movements.

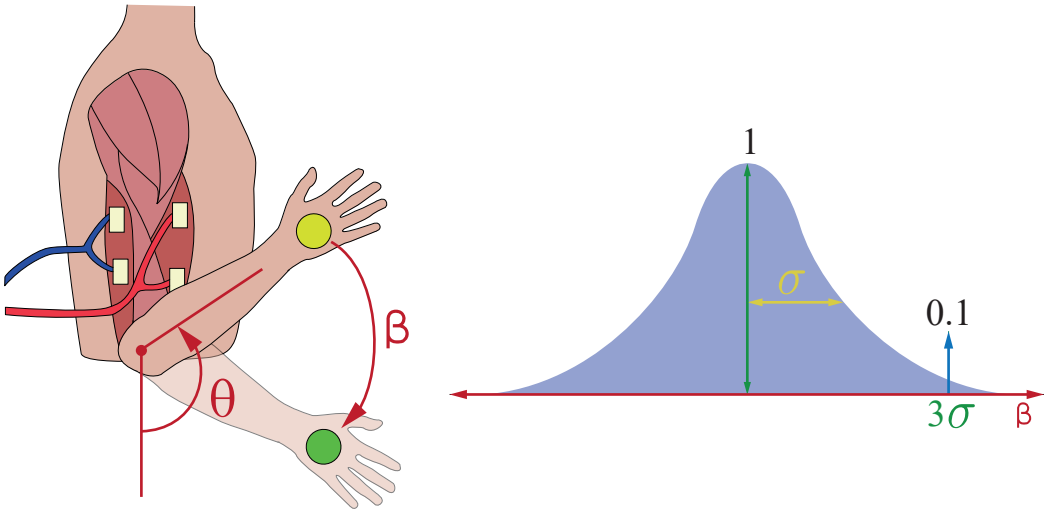


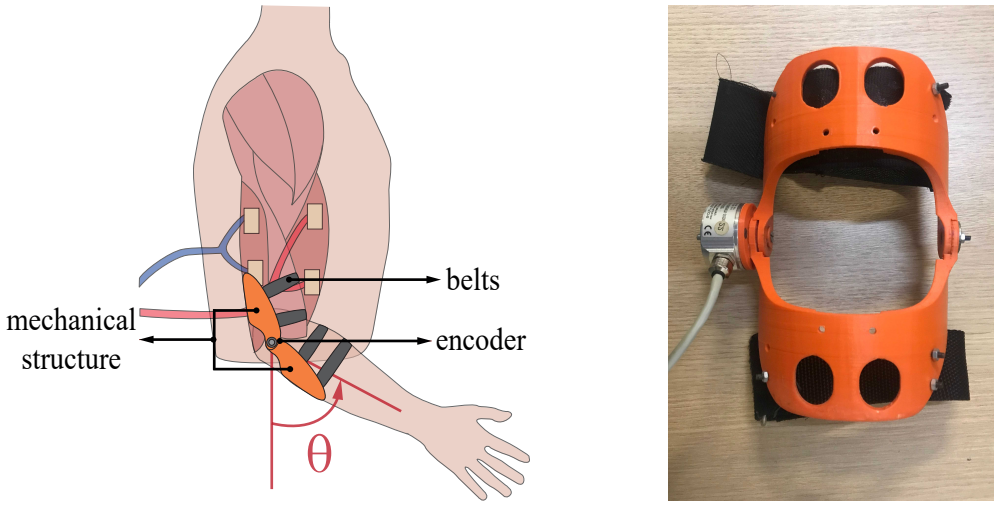
Figure 3.7: Graphical representation of the activity of reducing the angle between the desired (green) and measured position (yellow).

3.3 Real world implementation

An essential stage in all research work is to evaluate the system's performance (e.g., algorithm, mechanical structure) in a real environment, where working conditions differ from those in a simulation environment. The experimental tests allow observing the system's limitations to the noise of the measurements and the difference between the modeled and the real dynamics. In the same way, generally, it implies the development of additional mechanisms to obtain all the relevant data for the correct system functioning.

3.3.1 Estimation of elbow's position and velocity

The intelligent agent needs the position and angular velocity of the elbow to determine the amplitude of each electrical pulse. In the simulation environment, it is easy to access these measurements; however, devices capable of measuring these values are necessary for real-world implementation. For this reason, a mechanical system capable of measuring the angular position of the elbow concerning the anatomical position of the arm was designed; the anatomical position was described in Figure 2.2. On the one hand, the angular displacement is measured using an incremental encoder with a precision of 2048 counts per revolution. On the other hand, a mechanical structure is designed that allows the encoder to be secured to the user's arm. The mechanical system to measure the elbow's angle is shown in Figure 3.8.



(a) sketch

(b) physical prototype

Figure 3.8: Mechanical system to measure elbow's angle; The mechanical system consists of a mechanical structure to place the encoder and belts to secure it to the user's arm.

A widely used method to estimate speed is through a finite difference between the current measurement and a previous time instant, $\dot{\theta} \approx \frac{\theta_t - \theta_{t-1}}{\Delta t}$. However, the finite difference method is susceptible to environmental noise, and the estimated velocity usually presents undesired oscillations. Therefore, a Kalman filter will be used to estimate the velocity. Figure 3.9 shows a comparison of velocity estimation

between the finite element method and the Kalman filter ($R = 0.001\mathbb{I}_{2\times2}$ and $Q = 0.1$).

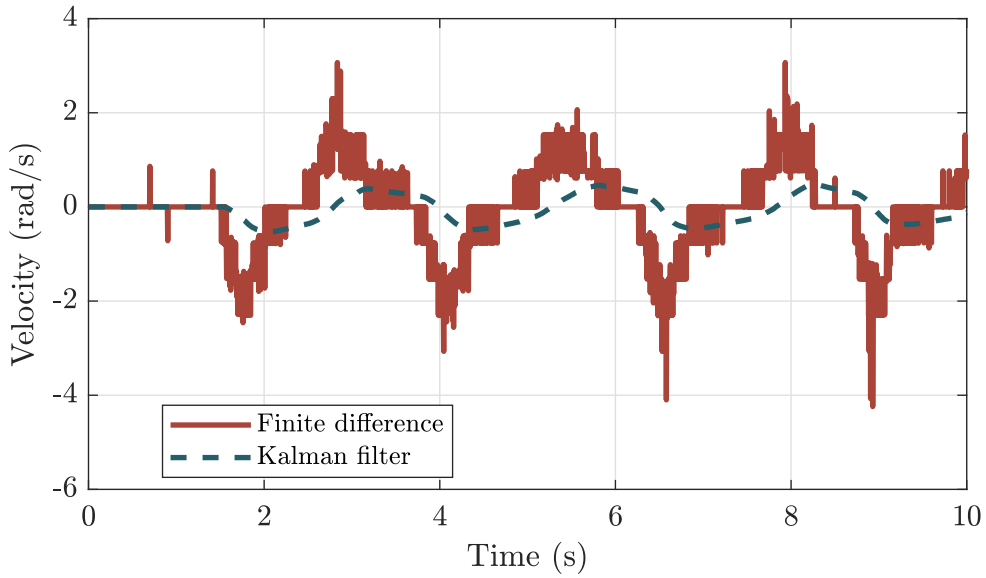


Figure 3.9: Velocity estimation comparison between finite element method and Kalman filter.

3.3.2 Testing protocol

In the simulation environment, the calculated electrical pulse generates muscle contraction with no change in the amplitude or shape of the electrical stimulus. However, in a real-world implementation, the electrical pulse must penetrate the layers of the skin to generate muscle contraction; this process involves changes in the amplitude and shape of the electrical signal. For this reason, it is necessary to condition the user's arm to improve electrical transmission and reduce signal loss. The procedure begins with estimating the start and end position of the target muscles (biceps and triceps). Afterward, the patient's arm is cleaned with isopropyl alcohol, and conductive fluid is placed on the electrodes. Finally, the user's current range calibration begins manually.

The last step before generating the controlled elbow flexion and extension movements is to relate the muscle activation (from 0 to 1) with the electrical amplitude range of the user's arm. For this purpose, two methods are proposed to

recognize the minimum and maximum electrical amplitude. On the one hand, the first method is based on gradually increasing the amplitude of the electrical pulse; consider as a minimum the amplitude of current that begins to raise the hand and as a maximum the amplitude that raises the hand 130 degrees. On the other hand, the second method is based on applying a random amplitude of the electrical pulse and iteratively finding the muscle activation that generates approximately the same angular position.

Chapter 4

RESULTS AND DISCUSSIONS

This chapter presents the results of the work with the methodology described in chapter 3. The first section describes the architecture of the neural networks and the hyperparameters of the optimization algorithm for deep neural networks, reinforcement learning and soft actor critic algorithms. The second section presents the performance of the intelligent agent to generate controlled elbow movements in the simulation environment and the real world.

4.1 Training setup

The intelligent agent's decision-making process involves deep learning and reinforcement learning algorithms. Both methods require tuning parameters that influence the speed and success of the training. On the one hand, the performance of neural networks depends on the number of hidden layers, the number of neurons per each, and the learning rate. On the other hand, reinforcement learning uses the discount factor (γ) to establish the influence of immediate or future rewards on the agent's training. Finally, SAC uses parameter ρ to establish the transfer of learning between predict and target neural networks.

4.1.1 Training parameters of deep neural networks

Deep reinforcement learning uses deep neural networks to estimate the value function and predict the best action. The architecture of these networks consists of three fully connected hidden layers. Likewise, the number of neurons per layer is halved with each level of depth in the neural network. The number of input, output and hidden layer neurons for each reinforcement learning element is shown in Table 4.1.

Table 4.1: Parameters of the deep neural networks to estimate the value function and predict the best action.

Element	Parameter	Value
Policy	number of inputs	5
	number of outputs	4
	hidden layers architecture	(64, 32, 16)
Value function	number of inputs	7
	number of outputs	1
	hidden layers architecture	(64, 32, 16)

The learning process of neural networks consists of transmitting the input data throughout the neural network and generating output data (e.g., estimation, prediction, classification). From there, compute the gradient (output relative to the neural network parameters) and use an optimization algorithm to update the neural network parameters. Most optimization algorithms modify the gradient descent method to increase the convergence speed and obtain optimal solutions (CHOI et al., 2019).

Adam is an optimization algorithm that uses the first and second moments of the gradient to adapt the learning rate; in this way, Adam overcomes local minima and increases convergence speed (KINGMA; BA, 2014). The optimizer requires tuning three parameters: the default learning rate and the gains of the two moments of the gradient. Tabla 4.2 shows the values used for training deep neural networks.

Table 4.2: Parameters of Adam optimization algorithm.

Parameter	Definition	Value
α	default learning rate	1e-4
β_1	exponential decay rate for first momentum	0.9
β_2	exponential decay rate for second momentum	0.99

4.1.2 Training parameters of deep reinforcement learning

Reinforcement learning uses a reward mechanism to guide the training of the intelligent agent; at each iteration, the agent receives a reward based on how much its decision contributed to solving the task. The reward system to encourage the learning of controlled elbow flexion and extension movements was described in (3.1) and the parameters used during training are described in Table 4.3. However, getting a high reward in the t iteration does not guarantee the highest cumulative sum of rewards at the end of the episode; in some cases, it can generate unfavorable conditions for the following iterations. For this reason, reinforcement learning regulates the influence of immediate and future rewards with the parameter γ; in general, $\gamma = 0.99$ gives good results for control tasks (DUAN et al., 2016). From there, the SAC cost functions, described in (2.1) and (2.2), can be used to modify the parameters of the neural networks.

Table 4.3: Parameters of the reward system.

Parameter	Definition	Value
α_1	influence of position error	1
α_2	influence of velocity penalty	0.01
α_3	influence of high activation penalty	0.01
max_error	position error for 10% of the maximum reward	20°
σ	standard deviation	6.3

The soft actor-critic requires adjusting two parameters. The first parameter is ρ and represents the learning transfer between the neural networks for the prediction and the target. In order to avoid instability during training, $\rho = 0.1$ was used; this implies that the parameters of the prediction networks ($Q_{\text{prediction}}$) influence 10% of the new parameters of the target networks (Q_{target}). The second parameter is the entropy coefficient and is automating computed as (HAARNOJA et al., 2018)

$$J(\alpha) = \mathbb{E}_{a_t \sim \pi_t} \left[-\alpha \log \pi_t(a_t|s_t) - \alpha \bar{\mathcal{H}} \right],$$

where α represent the entropy coefficient and $\bar{\mathcal{H}}$ is the expected minimum entropy of the policy; during training was consider $\bar{\mathcal{H}} = -2$.

4.2 Performance of the intelligent agent

The performance of the intelligent agent to generate controlled elbow movements was evaluated with two angular position steps; each step lasted 10 seconds. In this way, the intelligent agent demonstrated his ability to reach and maintain a desired angular position. The angular position of the steps was chosen based on the physical limitations of the mechanical system to measure the elbow angle. The biceps muscle grows with the elbow flexion angle; hence the belts of the mechanical system detach the skin-surface electrodes when the flexion angle is higher than 70°. Therefore, angles of 30° and 60° were chosen to compare results in a simulation environment and the real world.

Both in the simulation and experimental tests, the right arm starts out extended (i.e., elbow angle ≈ 0) and without obstacles around it. From there, the intelligent agent must calculate the muscle activations that move the arm to the angular position of each step and maintain the position for 10 seconds. Finally, the exercise of reaching and maintaining the position of the steps is repeated 6 times to validate the repeatability of the results. In the case of experimental tests, 5 minutes are waited between each repetition to relax the volunteer muscles; the experiment considers 5 volunteers.

4.2.1 Results in the simulation environment

Figure 4.1 shows the performance of the intelligent agent to reach and maintain the two desired angular positions in the simulation environment. In this figure, the agent reaches 98% of the first step with 0.48 seconds, 97% of the second step with 0.55 seconds, and the root mean squared error is 4.96°. However, despite the rapid reduction of the error in angular position, the agent maintains an average steady-state error of 1.4° and low-frequency oscillations with an amplitude of 1°.

Figure 4.2 shows the electrical stimuli calculated by the intelligent agent to perform the exercise of reaching and maintaining the desired angular positions; results are analyzed in four stages. In the first 0.3 seconds of both step (yellow wall), the agent started with high biceps activation ($\approx 80\%$) to reduce the high angular

position error and low triceps activation ($\approx 25\%$) to moderate elbow's velocity. In the second 0.3 seconds of both steps (green wall), the agent reduces biceps activation ($\approx 30\%$) and increases triceps activation ($\approx 70\%$) to reduce elbow's velocity and avoid overshoot. During middle of both steps, muscle activation of biceps and triceps have oscillations with amplitude of 5%. Likewise, unexpected behavior occurred on the second step. After the first second, the agent decides to reduce the biceps activation from 60% to 47%. Consequently, the root mean squared error during the second step is 0.8° greater than first step. Finally, the agent's behavior results from training using the reward system described in (3.1); therefore, analyzing the reward system is essential for a more detailed examination of the agent's abilities and limitations.

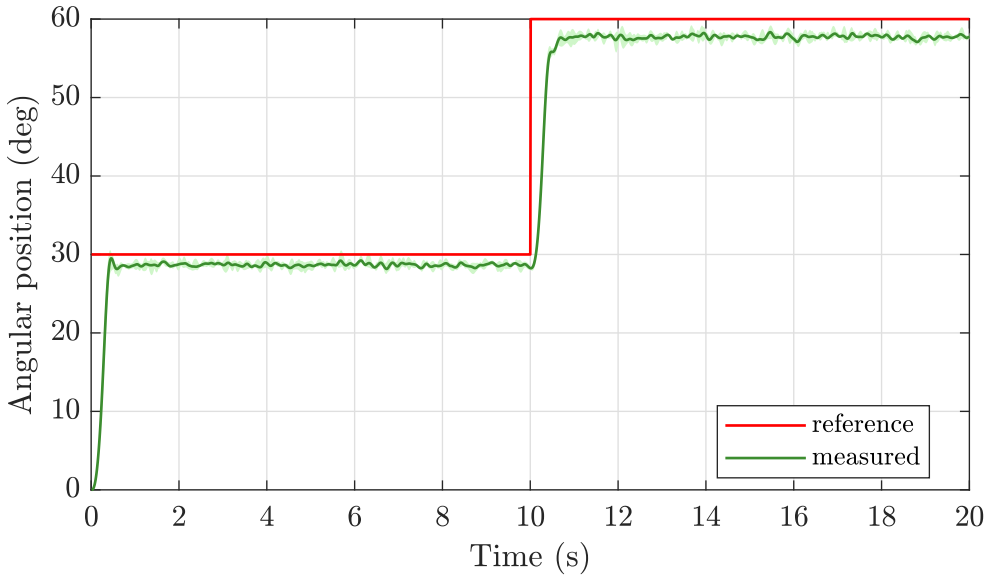
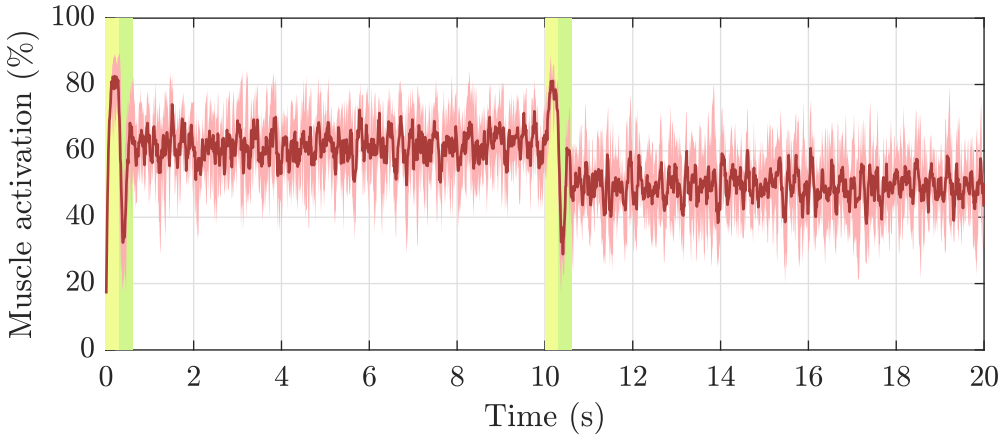
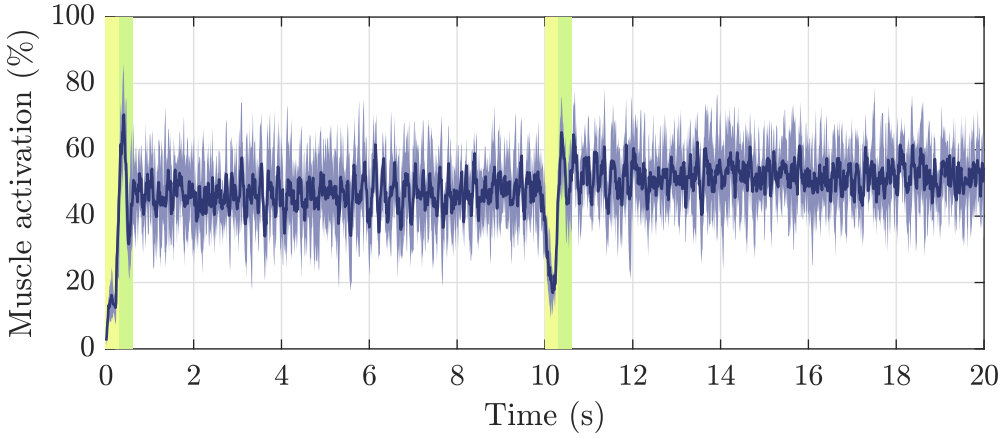


Figure 4.1: Performance of the intelligent agent to generate controlled elbow movements in a simulation environment. The reference trajectory comprises two angular position steps; each step lasts 10 seconds.

The reward system, described in (3.1), encourages the reduction of angular position error while maintaining low speed and energy consumption. Therefore, at some point, the agent must decide between increasing muscle activation to reduce the error or maintaining the position to avoid consuming more energy. Consequently, the agent has a nonzero steady-state error ($\approx 1.4°$) and reduces the bicep activation in the second step.



(a) biceps



(b) triceps

Figure 4.2: Electrical stimuli to reach and maintain the desired angular position. The reference trajectory consists of two angular position steps; each step lasts 10 seconds.

4.2.2 Results in the real world

Figure 4.3 shows the performance of the intelligent agent to reach and maintain the two desired angular positions in the real world. In this figure, the agent reaches 98% of the first step with 1.44 seconds and 97% of the second step with 5 seconds. Likewise, the angular position of the elbow presents oscillations of 5° after reaching the desired position, overshoot of 8%, and root mean squared error of 8.6°. The intelligent agent obtained better performance metrics (e.g., settling time, overshoot) in the simulation environment than in the real world. The performance reduction

can be due to external factors that were not considered during the training of the intelligent agent.

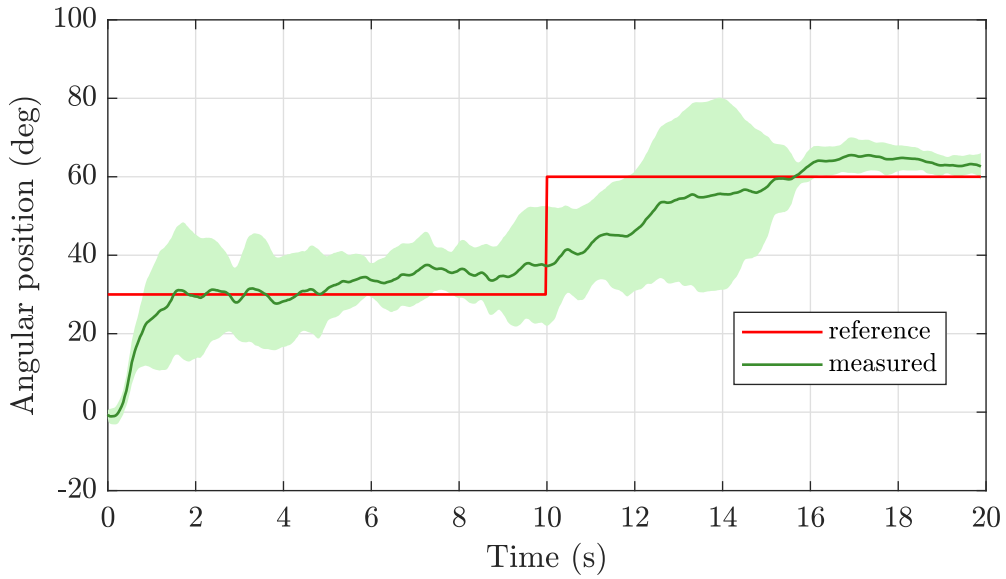


Figure 4.3: Performance of the intelligent agent to generate controlled elbow movements in a simulation environment. The reference trajectory comprises two angular position steps; each step lasts 10 seconds.

During the experimental tests, the volunteers had to relax their muscles and let the electrical stimulator apply electrical pulses to generate the elbow movements. However, most of the volunteers showed surprise every time the experiment started; due to the increase in amplitude from 0 mA to 8 mA. Hence, the users' involuntary contractions affected the intelligent agent's performance. In addition, the latency of the sensors and the electrical stimulator exceeded the established 20 ms to obtain sustainable muscle activations. Figure 4.4 shows the time delays generated by the electrical stimulator.

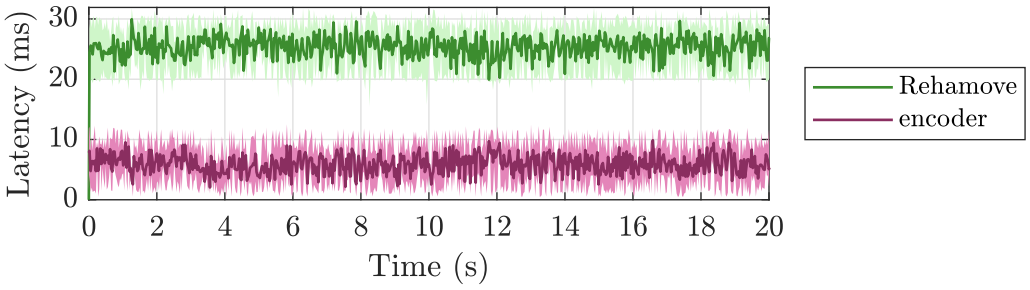
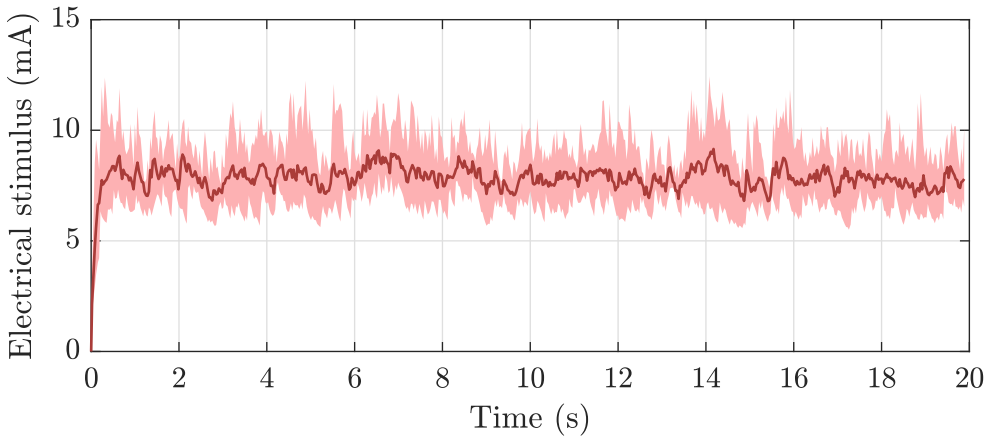
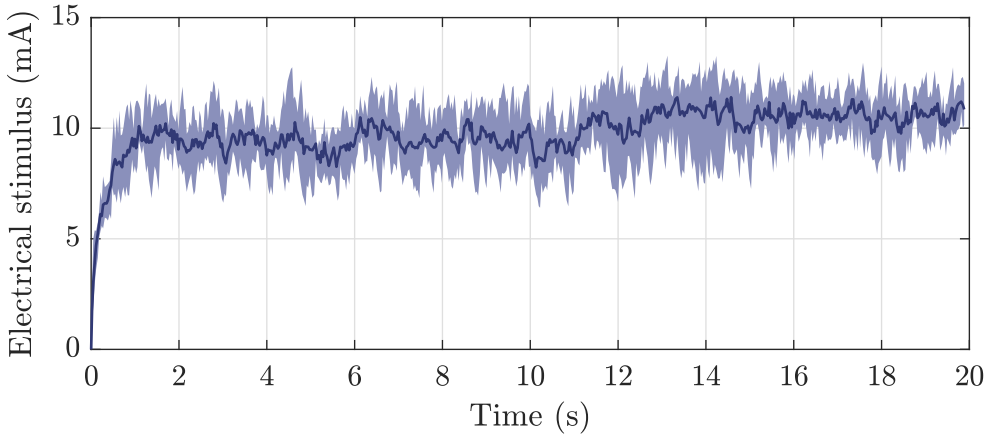


Figure 4.4: Device latency during experimental tests.

After the calibration process, 15 mA corresponds to the maximum activation while 5 mA corresponds to the minimum activation; all experimental tests used this range of electrical amplitude. Figure 4.5 shows the electrical stimuli calculated by the intelligent agent to reach and maintain the two desired positions. In Figure 4.5a, the agent rapidly increased ($\approx$ 0.5 seconds) biceps amplitude to 8 mA ($\approx$ 0.32 muscle activation) to reduce the error in angular position. Hence, it maintains that amplitude value with low amplitude oscillations ($\approx$ 2.3 mA). In Figure 4.5b, the agent slowly ($\approx$ 1.1 seconds) increased triceps amplitude to 10 mA ($\approx$ 0.5 muscle activation) to reduce velocity and overdrive. Hence, it maintains that amplitude value with low-amplitude oscillations ($\approx$ 2.2 mA).



(a) biceps



(b) triceps

Figure 4.5: Electrical stimuli to reach and maintain the desired angular position. The exercise consists of two angular position steps; each step lasts 10 seconds.

Chapter 5

CONCLUSIONS AND FUTURE WORK

In this work, an intelligent agent was trained to generate controlled elbow flexion and extension movements. The OpenSim program was used to create a reinforcement learning environment that allows the agent to understand how the activation of each muscle (i.e., biceps and triceps) affects the angular position of the elbow. Besides, the intelligent agent uses the soft actor-critic algorithm to learn to make the best sequence of decisions (i.e., muscle activation level). Finally, a reward system was formulated that incentivizes the reduction of angular position error while maintaining low speeds and low energy movements.

The performance of the intelligent agent to generate controlled movements of the elbow was evaluated with two steps of angular position; each step lasted 10 seconds. In this way, the intelligent agent demonstrated its ability to reach and maintain a desired angular position. The exercise was performed 6 times to validate the repeatability of the results. In the case of the experimental tests, 5 minutes were waited between each repetition to relax the volunteer's muscles; the experiment considers 5 volunteers. Likewise, the agent determines muscle activation, and the Rehamove electrical stimulator sends electrical pulses to generate muscle contractions. The relationship between muscle activation and the electrical amplitude range of the arm muscles was determined experimentally.

The agent showed better performance metrics (e.g., settling time and overshoot) in the simulation environment than in the real world; settling time of 1.44 seconds, overshoot of 8% and root mean squared error of 8.6°. The main reasons are: (i) the latency of the angular position sensor and the electrical stimulator and (ii) involuntary muscle contractions of the user. On the one hand, the atmega328p microcontroller took $\approx$ 7 ms to send each measurement, and Rehamove took $\approx$ 25 ms to generate each electrical pulse; hence, the total latency time was $\approx$ 32 ms which exceeded the ideal condition of 20 ms. On the other hand, users are not used to receiving electrical pulses and often experience involuntary contractions when

the agent rapidly increases the amplitude of the electrical pulse. Both factors contributed to reduced agent performance in the real world.

As for future work, it would be good to estimate the most important parameters of the user's muscles to improve the performance of the intelligent agent. Likewise, use inertial measurement sensors to estimate the angular position of the elbow; the current mechanical system is awkward and tends to slide off the arm slowly. Finally, consider external forces (e.g., involuntary contractions) during agent training.

REFERENCES

BARBOUCH, H. et al. Sliding mode control for functional electrical stimulation of a musculoskeletal model. In: IEEE. *2017 International Conference on Advanced Systems and Electric Technologies (IC_ ASET)*. [S.l.], 2017. p. 366–371.

CANNING, C. G.; ADA, L.; O'DWYER, N. Slowness to develop force contributes to weakness after stroke. *Archives of physical medicine and rehabilitation*, Elsevier, v. 80, n. 1, p. 66–70, 1999.

CANNING, C. G.; ADA, L.; O'DWYER, N. J. Abnormal muscle activation characteristics associated with loss of dexterity after stroke. *Journal of the neurological sciences*, Elsevier, v. 176, n. 1, p. 45–56, 2000.

CHOI, D. et al. On empirical comparisons of optimizers for deep learning. *arXiv preprint arXiv:1910.05446*, 2019.

COUPLAND, A. P. et al. The definition of stroke. *Journal of the Royal Society of Medicine*, SAGE Publications Sage UK: London, England, v. 110, n. 1, p. 9–12, 2017.

DANTAS, L. F. et al. Public hospitalizations for stroke in brazil from 2009 to 2016. *PLoS One*, Public Library of Science San Francisco, CA USA, v. 14, n. 3, p. e0213837, 2019.

DELP, S. L. et al. Opensim: open-source software to create and analyze dynamic simulations of movement. *IEEE transactions on biomedical engineering*, IEEE, v. 54, n. 11, p. 1940–1950, 2007.

DEWALD, J. P. et al. Abnormal muscle coactivation patterns during isometric torque generation at the elbow and shoulder in hemiparetic subjects. *Brain*, Oxford University Press, v. 118, n. 2, p. 495–510, 1995.

DUAN, Y. et al. Benchmarking deep reinforcement learning for continuous control. In: PMLR. *International conference on machine learning*. [S.l.], 2016. p. 1329–1338.

FEBBO, D. D. et al. Does reinforcement learning outperform pid in the control of fes-induced elbow flex-extension? In: IEEE. *2018 IEEE International Symposium on Medical Measurements and Applications (MeMeA)*. [S.l.], 2018. p. 1–6.

FITZSIMONS, C. F. et al. Stroke survivors' perceptions of their sedentary behaviours three months after stroke. *Disability and rehabilitation*, Taylor & Francis, v. 44, n. 3, p. 382–394, 2022.

FRANÇOIS-LAVET, V. et al. An introduction to deep reinforcement learning. *Foundations and Trends® in Machine Learning*, Now Publishers, Inc., v. 11, n. 3-4, p. 219–354, 2018.

FRENCH, B. et al. Repetitive task training for improving functional ability after stroke. *Cochrane database of systematic reviews*, John Wiley & Sons, Ltd, n. 11, 2016.

GORGEY, A. S. Robotic exoskeletons: The current pros and cons. *World journal of orthopedics*, Baishideng Publishing Group Inc, v. 9, n. 9, p. 112, 2018.

GROOT, J. H. de et al. Reduced elbow mobility affects the flexion or extension domain in activities of daily living. *Clinical Biomechanics*, Elsevier, v. 26, n. 7, p. 713–717, 2011.

HAARNOJA, T. et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018.

HAMILL, J.; KNUTZEN, K. M. *Biomechanical basis of human movement*. [S.l.]: Lippincott Williams & Wilkins, 2006.

HASOMED GmbH. *Functional Electrical Stimulation - HASOMED GmbH*. [S.l.], 2022.

HOWLETT, O. A. et al. Functional electrical stimulation improves activity after stroke: a systematic review with meta-analysis. *Archives of physical medicine and rehabilitation*, Elsevier, v. 96, n. 5, p. 934–943, 2015.

JARMEY, C. *LIBRO CONCISO DEL CUERPO EN MOVIMIENTO, EL (Color)*. [S.l.]: Editorial Paidotribo, 2008.

JOHNSON, W. et al. Stroke: a global response is needed. *Bulletin of the World Health Organization*, World Health Organization, v. 94, n. 9, p. 634, 2016.

KASSEBAUM, N. J. et al. Global, regional, and national disability-adjusted life-years (dalys) for 315 diseases and injuries and healthy life expectancy (hale), 1990–2015: a systematic analysis for the global burden of disease study 2015. *The Lancet*, Elsevier, v. 388, n. 10053, p. 1603–1658, 2016.

KINGMA, D. P.; BA, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

KITAMURA, T.; SAKAINO, S.; TSUJI, T. Bilateral control using functional electrical stimulation. In: IEEE. *IECON 2015-41st Annual Conference of the IEEE Industrial Electronics Society*. [S.l.], 2015. p. 002336–002341.

KNUDSON, D. V.; KNUDSON, D. *Fundamentals of biomechanics*. [S.l.]: Springer, 2007. v. 183.

KOUSHKI, A. et al. Deep reinforcement learning control of elbow motion under functional electrical stimulation. In: IEEE. *2021 9th RSI International Conference on Robotics and Mechatronics (ICRoM)*. [S.l.], 2021. p. 132–137.

LOOZE, M. P. D. et al. Exoskeletons for industrial application and their potential effects on physical work load. *Ergonomics*, Taylor & Francis, v. 59, n. 5, p. 671–681, 2016.

LÓPEZ, B. P.; AYUSO, D. M. R. *Terapia Ocupacional aplicada al Daño Cerebral Adquirido:(Colección Terapia Ocupacional)*. [S.l.]: Ed. Médica Panamericana, 2010.

MAFFIULETTI, N. A. Physiological and methodological considerations for the use of neuromuscular electrical stimulation. *European journal of applied physiology*, Springer, v. 110, n. 2, p. 223–234, 2010.

MILLARD, M. et al. Flexing computational muscle: modeling and simulation of musculotendon dynamics. *Journal of biomechanical engineering*, American Society of Mechanical Engineers Digital Collection, v. 135, n. 2, 2013.

PECKHAM, P. H.; KNUTSON, J. S. et al. Functional electrical stimulation for neuromuscular applications. *Annual review of biomedical engineering*, Palo Alto, Calif.: Annual Reviews, c1999-, v. 7, n. 1, p. 327–360, 2005.

SHEN, Y.; FERGUSON, P. W.; ROSEN, J. Upper limb exoskeleton systems—overview. *Wearable Robotics*, Elsevier, p. 1–22, 2020.

SimTK. *OpenSim Documentation*. 2023, January 31. Disponível em: <https://simtk-confluence.stanford.edu:8443/display/OpenSim/Documentation>.

SINGH, S. et al. Convergence results for single-step on-policy reinforcement-learning algorithms. *Machine learning*, Springer, v. 38, n. 3, p. 287–308, 2000.

STAUGAARD-JONES, J. A. *Anatomía del ejercicio y el movimiento*. [S.l.]: Paidotribo, 2014.

SUTTON, R. S.; BARTO, A. G. *Reinforcement learning: An introduction*. [S.l.]: MIT press, 2018.

THOMAS, P.; BRUNSKILL, E. Data-efficient off-policy policy evaluation for reinforcement learning. In: PMLR. *International Conference on Machine Learning*. [S.l.], 2016. p. 2139–2148.

UCHIDA, T. K.; DELP, S. L. *Biomechanics of movement: the science of sports, robotics, and rehabilitation*. [S.l.]: Mit Press, 2021.

WAKELING, J. M. et al. A muscle's force depends on the recruitment patterns of its fibers. *Annals of biomedical engineering*, Springer, v. 40, p. 1708–1720, 2012.

WANG, H. et al. Global, regional, and national life expectancy, all-cause mortality, and cause-specific mortality for 249 causes of death, 1980–2015: a systematic analysis for the global burden of disease study 2015. *The lancet*, Elsevier, v. 388, n. 10053, p. 1459–1544, 2016.

WANNAWAS, N.; SHAFTI, A.; FAISAL, A. A. Neuromuscular reinforcement learning to actuate human limbs through fes. *arXiv preprint arXiv:2209.07849*, 2022.

WINTERS, J. M. Hill-based muscle models: a systems engineering perspective. In: *Multiple muscle systems*. [S.l.]: Springer, 1990. p. 69–93.

WOLF, D. N.; HALL, Z. A.; SCHEARER, E. M. Model learning for control of a paralyzed human arm with functional electrical stimulation. In: IEEE. *2020 IEEE International Conference on Robotics and Automation (ICRA)*. [S.l.], 2020. p. 10148–10154.