MASTER THESIS

VALIDATION OF MACHINE LEARNING COMPUTED MUSCLE ACTIVATION WITH EXPERIMENTAL EMG

Yaela M. Amsing

FACULTY OF ENGINEERING TECHNOLOGY
DEPARTMENT OF BIOMECHANICAL ENGINEERING

EXAMINATION COMMITTEE
M. Sartori
M. Vlutters
G. Durandau
H. Wang

DOCUMENT NUMBER
BE - 910

UNIVERSITY OF TWENTE.

DEC 2022

Contents

1 Abstract

Objective In biomechanics, muscle-actuated control is researched through the use of neuromusculoskeletal models, and movement tasks can be performed with help of optimization techniques or machine learning. However, the accuracy of the muscle control that is generated by the simulation methods is still in question. ***Approach*** In this study, the muscle control that is simulated using a Myosuite musculoskeletal model during an elbow flexion/extension movement is validated with the help of experimentally collected EMG data. This paper describes how the experimental data is recorded, and the method for obtaining simulated muscle activation. The accuracy of the simulated muscle activation is evaluated by comparing it to the experimental data. In addition, the position data that is achieved by the muscle activation is evaluated for different reinforcement learning policies. ***Main results*** Results demonstrate the possibility of simulating realistic muscle activation when an optimized reward function is used. The approach was shown to be effective to some extent for extrapolating movements with different speeds. However, more experimental data, in the form of more subjects and different movements, is needed for further validating the used models and policies. ***Significance*** The ability to successfully simulate biomechanical tasks with accompanying realistic muscle control creates possibilities in developing and testing prosthetics and exoskeletons, along with rehabilitation opportunities. Simulation of biomechanical tasks enables us to better understand human-machine interaction.

2 Introduction

Nowadays, the study of biomechanics is used in rehabilitation, for the design of prosthetics and exoskeletons, and to help solve general complex motor neural problems like moving the fingers or picking up an object.
The current approach for the use of prosthetics and exoskeletons is mostly based on experience and trial and error. Lower arm prosthetics are commercially available in a variety of forms but have not been tested clinically on a large scale. This leads to contrasting results for different types of users and therefore makes user abandonment a great issue [19].
Similarly, clinical studies with lower arm exoskeletons designed for rehabilitation have been preliminary. The studies that assess rehabilitating capabilities of arm exoskeletons have only used a limited number of test subjects, and results have not all been positive [6]. This is due to a lack of a generally accepted protocol and suitable tools to assess the coordination between an exoskeleton and the human body. It results in a time-consuming trial-and-error approach before actual design decisions can be made.

A tool that can be used to improve this approach is simulation. Researchers simulate the prosthetic or exoskeleton working together with the human body. In this way, the effect that it has on motor control and musculoskeletal tissue can be studied. The simulation approach has several benefits.
Firstly, it saves money on the hardware of a prosthetic or exoskeleton as its effectiveness can be proven using simulation and, therefore, no physical experiments are needed. Adaptations in the design can simply be added to the simulation, saving on production costs.
Secondly, simulation saves time. Conducting real-life experiments is time-consuming and can be mostly avoided by simulation. Experiments that are simulated do not require intensive supervision because there are no real-life subjects involved. Also, it is possible to run multiple experiments in parallel to each other.
Finally, prosthetics and exoskeletons have the potential of being dangerous, as they often contain hard parts that can hurt or injure the user. Simulation offers a safer way to test these solutions.
Because it is possible to test the prosthetic or exoskeleton with simulation before it is made, there is an opportunity to personalize it. Specific needs and wishes of the user, like product size and weight, can easily be implemented into the product.

Simulating human movement also has its challenges. One of these challenges is generating realistic muscle control. Different methods have been presented to do this, including optimization techniques such as computed muscle control (CMC) and central pattern generators (CPGs). CMC uses static optimization to determine individual muscle excitation from joint moments, which leads to the desired acceleration needed to track a movement [11]. In this method, muscle activation is minimized. The CMC method also compensates for delays between muscle excitation and force development.
A CPG is capable of automatically generating a rhythmic muscle contraction, without the need for rhythmic input. An example is a motor control system that simulates muscle activation to produce a natural swimming movement, based on CPGs, which was presented by Weiguang Si et al. [15].
Another way to simulate human movement is to use muscle reflexes for biomechanical control. An example of this is a human gait model that is controlled by muscle reflexes [3]. It was found that the model is capable of simulating stable walking dynamics, based on its interaction with the ground. The model was also able to predict muscle activation patterns that were seen in other experiments. Unfortunately, due to the limited computational speed and accuracy of physics engines, all the proposed methods have their limitations.

In addition to the methods described above, natural human movements can be simulated using artificial intelligence. Machine learning helps to create neuromusculoskeletal models that generate their own movements. As a result, there is no need for experimental data, unlike the optimization methods which rely on real data. Only a few models are needed to simulate the entire range of human movements.

A way of using neuromusculoskeletal models in combination with machine learning to perform biomechanical tasks is the Myosuite framework, developed by Caggiano et al. [1]. This is a platform that contains physiologically accurate musculoskeletal models based on validated OpenSim models. OpenSim is a software package that can be used to build biomechanical models [14]. OpenSim models can be manipulated, and their dynamics can be simulated to get a better understanding of human biomechanics without performing experiments. The Myosuite models reflect the human muscular and skeletal systems, which are controlled by the nervous system. Myosuite allows for a more realistic simulation of the human body because changes to the musculoskeletal geometry, such as tendon transfers or exoskeleton assistance, can be included. This facilitates the research in human-device interaction. In addition, movements containing physical contact with the external environment are supported. Myosuite models are equal to their OpenSim original but can be simulated over two orders of magnitude faster because the muscle actuators in Myosuite are simplified. This makes Myosuite a suitable tool for training models to simulate human biomechanical movements with the help of reinforcement learning. Together with a set of realistic, biomechanical tasks which are simulated with the MuJoCo physics engine, Myosuite is designed to study human motor control. The MuJoCo physics engine has been used for more muscle-actuated simulation projects, such as OstrichRL, a project where a musculoskeletal model for the bipedal locomotion of an ostrich was built, together with reinforcement learning tasks [7].

When testing a biomechanical solution such as a prosthesis or an exoskeleton using a simulation, the simulated motor control has to be realistic. This is necessary to see what the impact of the solution is on the body, as well as the human adaptation to the device. In addition to a realistic movement, it is important to simulate human-like muscle activation. This makes results from human-device coordination experiments natural and therefore reliable.

In this study, the extent to which a model is capable of simulating realistic muscle activation by using reinforcement learning is analyzed. For this, a validation of the neural control, used by Myosuite to solve complex biomechanical problems, is performed. A functional simulation is made using Myosuite, that can generate a movement with muscle activation. Next, the model is trained using reinforcement learning in such a way that it is capable of tracking real human movement. After that, the simulated muscle activation, which is used by the model to generate that movement, is analyzed. The study will focus on elbow flexion/extension in particular, and the muscles that cause this movement. The goal of the study is to get an understanding of the extent to which Myosuite is able to generate realistic movement with natural muscle activation. For this study, two hypotheses can be distinguished:

1. By using reinforcement learning, a Myosuite model is capable of tracking the position of the elbow accurately, using muscle activation of the biceps (long), triceps (lateral), triceps (long), and brachialis.

2. By modifying the reward function in the reinforcement learning algorithm, more natural muscle activation can be simulated.

In the following sections, the method for obtaining experimental data will be discussed, followed by the tools that were used for generating the simulated data. The results will show a comparison of the experimental muscle activation data and the simulated muscle activation data, along with a comparison of the experimental position data and the simulated position data, corresponding to different policies. The results are analyzed in the discussion. In the conclusion, the performance of the general model will be discussed.

3 Experiment

The goal of the first part of this study is to get an insight into the human biomechanics during a movement. To do this, the muscle activation of fourteen different muscles was recorded using EMG. This was done for a total of eleven different movements, involving joints in the lower arm and hand. In addition to muscle activation, the position of the arm and hand was recorded using motion capture. Since only a lower arm movement was recorded, all other body segments were placed to be as relaxed as possible during the experiment. Hence, the subject was placed on a chair with his legs and left arm relaxed. Data from one subject was gathered using this protocol. A comprehensive version of the experimental protocol can be found in Appendix A. For the validation of the simulated data, only the experimental data of the elbow flexion/extension movement was used.

3.1 Muscle activation

Muscle activation of muscles on the right arm was measured using EMG. The EMG data was recorded using a Delsys software system and a Delsys Trigno sensor system, with a frame rate of 2048 frames per second. Surface electrodes were placed on the body surface, right on top of the muscle, and recorded the electrical signals produced by that specific muscle. During eleven different movements, EMG activation of a total of fourteen different muscles was collected: biceps brachii (long head), triceps (lateral head), flexor carpi ulnaris, brachialis, triceps (long head), flexor digitorum profundus, extensor carpi ulnaris, flexor carpi radialis, flexor digitorum superficialis, pronator teres, abductor pollicis longus, opponens pollicis, extensor carpi radialis and opponens digiti minimi.

The raw EMG data that was collected in this experiment, was processed using several BioMechPro modules in Matlab. For each movement, the EMG activation of every muscle was plotted against time. A module was used to detrend the data, making the vertical shift zero. A 25 Hz high pass filter was used to remove the DC offset and low-frequency artifacts. Next, the data was rectified, leaving the absolute value of the signal. This makes it possible to compute an average. Then, a 6 Hz low pass filter was applied to the EMG data, removing high frequencies from the signal and making it more smooth with a moving average to obtain the linear envelope. The EMG data was normalized by using the maximum voluntary contraction data from the experiment. The processed EMG data was visualized using Matlab.

3.2 Motion capture

In addition to muscle activation, the 3D position and orientation of the arm and hand during the movements were recorded. To attain this information, 33 passive reflective markers were placed on the arm and hand of the subject, shown in Figure 1. The position of each of the individual markers during the movements was tracked by a Qualisys motion capture system and visualized for analysis. Twelve Oqus cameras were used to record the motion capture data and two Miqus video cameras were used to film the movements, both with a frequency of 128 Hz. The Qualisys Track Manager was used for analyzing the movement of the markers, which were individually identified using the Automatic Identification of Markers (AIM) system in Qualisys. The locations on the body where all the reflective markers were placed are: acromion (top and bottom), clavicle (lateral and medial), C7, back (three markers), upper arm (three markers), elbow (lateral and medial), lower arm (three markers), wrist (radial and ulnar), tip of every finger, metacarpophalangeal (MCP) joint of every finger, and proximal interphalangeal (PIP) joint of every finger.

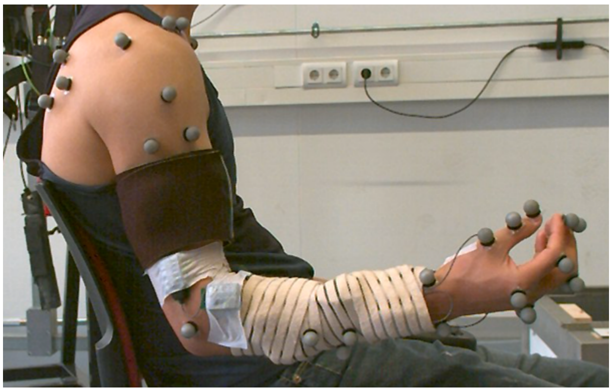


Figure 1: Placement of the passive reflective markers on the subject.

To analyze the motion capture data recorded during the experiment, an upper limb musculoskeletal OpenSim model was used that contains bones and muscles of body segments in the arm and hand [12]. For the analysis, all the joints from the shoulder to the distal fingers are regarded. In the OpenSim model, virtual markers are placed at the same anatomical locations as the reflective experimental markers.
After the virtual model markers were placed, the dimensions of the body segments of the model were scaled using OpenSim, as well as their mass properties. This was done by matching the distance between the virtual markers to the distance between the experimental markers. After this, the virtual model markers were moved to the experimental markers' position in a static pose. For this, the model markers needed to be linked to the body segment that they were placed on, to keep them in the same position after scaling.
Using the motion capture data from the experiment, it was possible to determine the coordinates of the arm during movement. This was done using inverse kinematics (IK), in which joint parameters are calculated from a given position and orientation of the body segment. In this study, the angle of the elbow joint was calculated.

Inverse kinematics was performed by the IK tool in OpenSim. This tool uses three different inputs: the scaled model including the adjusted virtual markers, the experimental marker trajectories, and a settings file containing marker weightings. The output is a motion file that contains coordinate trajectories in the form of joint angles. In Biomechpro, inverse kinematics can also be carried out by a specific module, which is opened via Matlab. It runs inverse kinematics in OpenSim, and stores the data in the correct structure right away.

3.3 Protocol

In total, data from eleven movements was recorded, involving several joints. Elbow movements include flexion/extension of the elbow, with and without additional weight of 2.5 kg at a specified speed, flexion/extension of the elbow with and without an additional weight of 2.5 kg at random speed, and holding a weight in a flexed position. Wrist movements include wrist deviation, wrist flexion/extension, and forearm pronation/supination. Hand movements include hand opening/closing, and touching the thumb with every finger separately. All movements were performed in repetitions of ten. Additionally, a movement was performed that combines the previously mentioned joints, which consists of grabbing a stick from a plateau, twisting it around, and putting it back. Also the maximum voluntary contraction of all muscles was recorded, and two static recordings were made. During the static recording, the subject held their arm still at a 0-degree and 90-degree angle.

For the validation of the simulated data that was generated using Myosuite, the experimental data of the flexion/extension movement of the elbow was used as a reference. In this movement, the subject holds their upper arm alongside the body, while lifting their lower arm up and down, changing the elbow angle. The two maximum positions are 0 degrees when the elbow is in a fully extended position, and 140 degrees when the elbow is in a fully flexed position. This movement was performed and analyzed at two speeds; one at 40 BPM, and one at unspecified random speed. For this second movement, the subject changed the speed randomly throughout the recording. Four of the fourteen muscles that were recorded are analyzed for this movement: biceps (long head), triceps (lateral head), triceps (long head), and brachialis.

4 Methods

In this section, the approach for simulating the elbow flexion/extension movement will be discussed. A Myosuite elbow model with 1 degree of freedom (elbow flexion/extension) and six muscles was used for this. The MyoSim pipeline was used to create this model, which is dynamically equivalent to the validated OpenSim model, and supports contact-rich simulation [20]. With this model, muscle activation of six muscles was simulated: biceps (long head), biceps (short head), triceps (lateral head), triceps (long head), triceps (medial head), and brachialis. Because muscle activation of the triceps (medial head) and the biceps (short head) was not experimentally recorded, only the muscle activation of the other four muscles was regarded in this study. The effect that the combination of activation of these muscles has on elbow flexion/extension was simulated.

4.1 Reinforcement learning

The Myosuite model uses reinforcement learning to generate a physiologically correct movement [16]. This is done by teaching the model to find the best solution to a given problem. A training policy was used to show which actions lead to a desired behavior.

In reinforcement learning, an agent is taught what to do in specific situations through the use of the Markov Decision Process [18]. In this process, the goal of the agent is to receive the highest total reward, and it must explore the actions which lead to that. At every time step, it takes an action that is based on the value of the reward it gets from that action. But in addition to the immediate reward, the agent also has to consider the possible rewards in subsequent actions. Before the agent knows what the best sequence of decisions is, it has to make decisions randomly. This trial and error approach continues until the agent has tried enough possibilities to find the sequence of decisions that leads to the highest reward. So before the agent knows how to perform a specific task, it has already tried out all the possibilities to do it. In the end, it chooses the sequence of actions that leads to the highest reward.

The reinforcement learning process used in this study took place in a test environment called OpenAI gym [10]. This is where the algorithm for a training policy can be written and tested. The framework for the algorithm that was used for this study was created for the Myosuite project, prior to this study. The OpenAI gym uses standardized methods and tools that make it easier to reproduce simulation experiments. The standard approach that was used in this study is similar to that for other environments; an agent interacts with the environment through an action, which in this study was muscle activation. Then, information is collected with the help of observation. This information includes the state of the environment after an action. The state of the environment for the model in this study consisted of position data and position error. The environment is reset to a certain state when the goal is achieved.

The performance of the agent in terms of training reward was evaluated periodically. Using the evaluation of the performance of the model after every 10000 calls, the best model is saved.

The algorithm that is used for training the agent consists of a neural network. This network is made up of an input layer, multiple hidden layers, and an output layer. Some of the hidden layers are shared between the policy network and the value network [9]. A policy network learns which specific output should be linked to which input during training. This network produces action-state combinations. A value network assigns a value to a particular state, corresponding to the reward that that state has acquired until that moment. The neural network that was used for the reinforcement learning policy in this study consists of two non-shared layers for the value network, both of size 64, and two non-shared layers for the policy network, also of size 64. Each layer uses a sigmoid activation function.
The algorithm makes use of Proximal Policy Optimization (PPO) to ensure that the best policy is accurate [13]. In each step within a training, it updates the policy to a new best policy. However, the old policy is not simply replaced by a new policy. PPO makes sure that the new policy does not vary too much from the old policy, in order to decrease the sensitivity for certain parameters and ensure a low variance during training.

In reinforcement learning, an agent is trained for a certain amount of time steps, in order to find out the optimal strategy that leads to the highest reward. Training the model after this has no effect because the reward value has reached a plateau and can not increase any more. Using Tensorboard, it was possible to find out how long it took the model to reach this plateau. For the model in this study, it took four million time steps to reach this final state and, thus, its maximum reward. The model was trained for a total number of five million steps, to make sure that the final state was always reached.
Because the model was trained on the same data set multiple times, there was a possibility that it learned the sinusoidal shape of the position data and extrapolate it without tracking the actual movement. To avoid this, a window was used. The window is a random piece of the signal that is 100 steps long. The model starts to track the signal from within that window. After that, a different random piece of the signal is chosen, which the model tries to follow. This continues over and over again, eventually covering the entire signal.

A problem that complicates simulating human-like muscle activation, is muscle redundancy. Muscle redundancy is applicable for many tasks, including elbow flexion/extension using four muscles. Because there are more muscles than degrees of freedom during this movement, there are theoretically infinitely many possibilities of combinations of muscle activation to perform the task. In the human body, it causes the central nervous system to select the best combination of muscle activation to perform a task. One hypothesis states that the body chooses this combination by using optimization; the combination of muscle activation of different muscles that leads to the lowest value for cost function is used [4]. The cost function is defined as the sum of squared muscle activities of all muscles involved in the task. Thus, the body performs a movement in a way that uses the least amount of energy. In this study, muscle activation is simulated in a similar way, but without optimization. Instead, using an EMG component in the reward function, the Myosuite model is taught to minimize muscle activation of each muscle while still having to perform the elbow flexion/extension task. This results in the lowest total muscle activation of every muscle, and therefore the lowest cost function. Muscle activation simulated by the Myosuite model can therefore be similar to natural muscle activation.

4.2 Reward function

The most important thing to teach an agent is what action leads to a high reward. This is done using a reward function. A change in this function may cause the agent to take different actions, leading to different behaviors. In the algorithm framework provided by the Myosuite project, the reward function was modified to suit the data that was used. In order to simulate more natural muscle activation, the reward function was modified. This was done by adding different components to construct the total reward function. Each component consists of a separate reward, which is added to another component's reward to form the total reward function. In this way, the effect of each of the components can be analyzed, and the results can be used to construct the optimal reward function. The total reward function is given by

$$Total\ reward = position\ reward + velocity\ reward + EMG\ reward + acceleration\ reward. \tag{1}$$

The reward function may consist of four different components; a position component, a velocity component, an EMG component, and an acceleration component.
Every component of the reward function is weighted by two scaling factors; a linear scaling factor $\boldsymbol{w}$ and an exponential scaling factor $\boldsymbol{s}$. Increasing the linear scaling factor emphasizes the importance of the component. The exponential scaling factor determines the slope of the curve of the reward gradient. A further explanation of the design of the reward function can be found in Appendix F. The values that were used for the individual scaling factors in every component were found by trial and error and can be seen in Table 1.

Table 1: Values for the linear- and exponential scaling factors for the four different components in the reward function.

Component	Linear scaling factor $\boldsymbol{w}$	Exponential scaling factor $\boldsymbol{s}$
Position reward	1.0	40.0
Velocity reward	0.5	20.0
EMG reward	0.0001	10.0
Acceleration reward	0.001	10.0

The total reward function is built up of four different components, which are added subsequently.
First, a ***position reward***, defined as

$$position\ reward = w_{\text{pos}} * e^{-s_{\text{pos}} * err1} \tag{2}$$

was added to the reward function. This component causes the agent to receive a high reward if the elbow angle of the model is close to the experimental elbow angle after completing an action. The closer the two values and, therefore, the smaller the position error ***err1***, the higher the reward, see Equation 2. In addition to this, the agent received a fixed negative reward of -500 when the error was above the threshold of 0.25. The linear scaling factor $\boldsymbol{w_{pos}}$ and exponential scaling factor $\boldsymbol{s_{pos}}$ were used for the position reward and can be seen in Table 1.

In addition to a similar position after an action, it was necessary that the simulated velocity of the elbow angle matched the experimental velocity, in order to create a smooth movement and natural muscle activation. In an effort to realize this, a ***velocity reward*** was added to the reward function, which is defined as

$$velocity\ reward = w_{\text{vel}} * e^{-s_{\text{vel}} * err2}. \tag{3}$$

After every action, the smaller the error between the simulated velocity and the experimental velocity, described as ***err2***, the higher the reward, see Equation 3. For the velocity reward, linear scaling factor $\boldsymbol{w_{vel}}$ and exponential scaling factor $\boldsymbol{s_{vel}}$ were used. The values for these factors are stated in Table 1.

To simulate more natural EMG data, the muscle activation should be as low as possible. Experimental muscle activation is also minimal because the human body tries to consume the least amount of energy. To include this in the reward function, an ***EMG reward*** was added, which is defined as

$$EMG\ reward = w_{\text{emg}} * e^{-s_{\text{emg}} * sum(act^2)}. \tag{4}$$

The lower the value for the muscle activation ***act***, the higher the reward, as can be seen in Equation 4. In Table 1, the values for the linear scaling factor $\boldsymbol{w_{emg}}$ and exponential scaling factor $\boldsymbol{s_{emg}}$ used for the EMG reward can be seen.

A way that helps generating smooth muscle activation is to minimize acceleration. For a natural movement, the signal should have a relatively constant speed that only changes when necessary. An ***acceleration reward***, defined as

$$acceleration\ reward = w_{\text{acc}} * e^{-s_{\text{acc}} * sum(acc^2)}, \tag{5}$$

was therefore added to the reward function. A low value for acceleration ***acc*** leads to a high reward, which can be seen in Equation 5. The linear scaling factor $\boldsymbol{w_{acc}}$ and exponential scaling factor $\boldsymbol{s_{acc}}$ which were used for the acceleration reward, can be seen in Table 1.

4.3 Data analysis

Two tools were used to measure the relation and the difference between the experimental data and the simulated data. The first is the Pearson correlation coefficient, which can be used to determine how strong the correlation is between two variables. The value of the Pearson coefficient is a number between -1 and 1. A negative value corresponds to a negative correlation, which means that if one of the variables increases, the other decreases. The negative correlation grows stronger when the value moves closer to -1. A value of 0 means that there is no relationship between the two variables at all. A change in one variable does not affect the other variable. A positive value corresponds to a positive correlation. If one of the variables increases, the other does too. The positive correlation grows stronger when the value moves closer to 1. The Pearson coefficient was determined for the simulated EMG data and the experimental EMG data of the biceps long and triceps long. The Pearson coefficient was also determined for the simulated elbow angle and the experimental elbow angle for different reward functions.

Another tool that was used to say something about the accuracy of the results is the root mean square error (RMSE). It measures the error between predicted and observed data. In this case, it was used to find out how spread out the simulated elbow angle data was around the line of the experimental elbow angle data. The standard deviation of the error was determined for every point in the data. For a good fit, the RMSE is as small as possible. A large RMSE means that there are elements that change the outcome for which the model does not account.

5 Results

The results in this section consist of experimental position data and experimental EMG data. In addition, outcomes of the best neural network models that were trained with different reward functions are displayed. In this study, there is a focus on the tracking accuracy of joint motions as well as the similarity of muscle activation of four different muscles between experimental recordings and simulation outcomes. To evaluate the similarity quantitatively, Pearson correlation coefficients and RMSE values were calculated. Also, an extrapolation with the trained policy on another motion trial is investigated. For this, the tracking accuracy and similarities in muscle activation are discussed for an elbow flexion/extension movement with random speed.

5.1 Experimental position data

In Figure 2, the results for the experimental position data can be seen. The output that is focused on is the elbow angle, which is depicted on the vertical axis in radians. It can be seen that the graph for the elbow angle has a sinusoidal shape, in which the maximums represent a fully flexed elbow, and the minimums represent a fully extended elbow. A total of 10 repetitions of elbow flexion/extension movement were recorded. In Figure 2, the first two repetitions are depicted, corresponding to a length of 7 seconds.

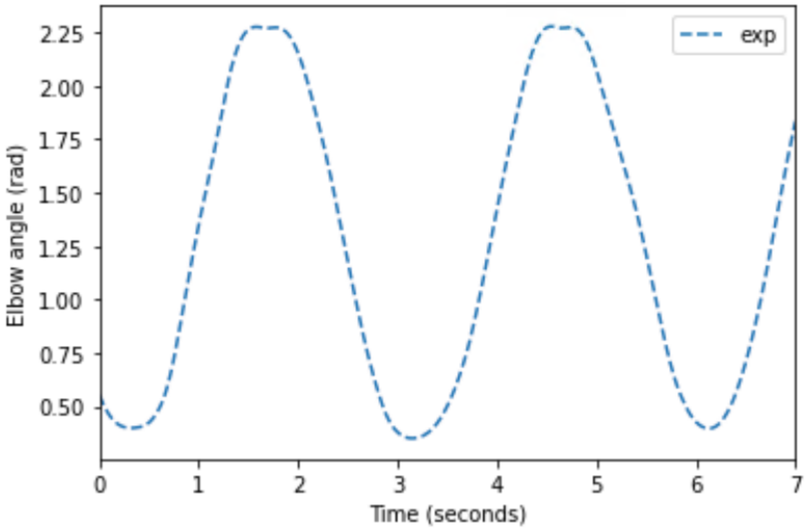


Figure 2: Experimental elbow angle in radians during two cycles of elbow flexion/extension.

5.2 Experimental EMG data

In this section, the experimental EMG data for four different muscles is shown. Two of the ten recorded repetitions are regarded. This consists of two full cycles of elbow flexion/extension, starting in a fully extended position when the elbow angle is at its minimum. The elbow is in a maximally flexed position after 1.7 seconds in the first repetition, and after 4.6 seconds in the second repetition. It is fully extended at 0.0 seconds, after 3.0 seconds, and after 6.1 seconds.

The values for the muscle activation are normalized, meaning they are scaled in a range between 0 and 1 in which zero represents no activation and one represents maximum activation.

In Figure 3 below, the experimental EMG data of the biceps (long head), triceps (lateral head), triceps (long head), and brachialis during two repetitions of elbow flexion/extension can be seen.

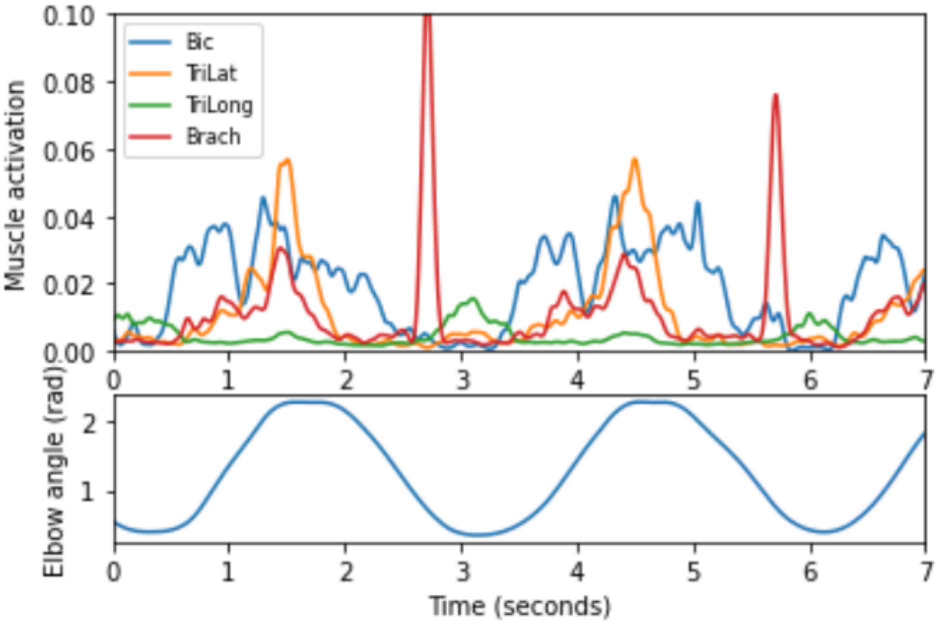
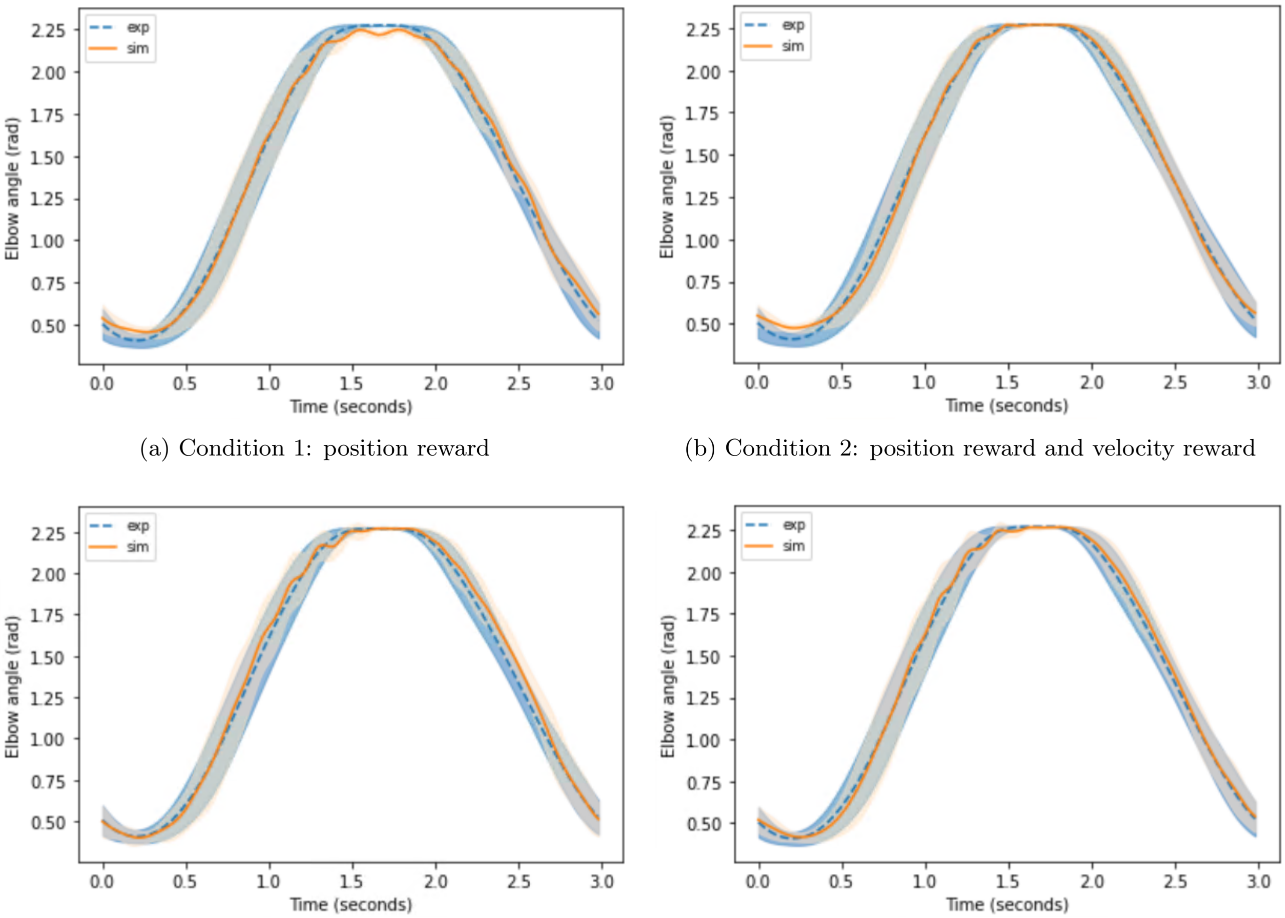


Figure 3: Experimental muscle activation (normalized) for the biceps long (blue), triceps lateral (orange), triceps long (green), and brachialis (red) during two cycles of elbow flexion/extension.

In Figure 3, the experimental EMG data for the biceps long can be seen in blue during two full cycles of elbow flexion/extension. An increase in activation during elbow flexion, between 0.3 seconds and 1.7 seconds can be seen. Then, activation decreases during elbow extension until 3.2 seconds. After that, activation increases and decreases again, with a maximum when the elbow is in a fully flexed position. This pattern matches the muscle activation for the biceps found in literature [2].
Figure 3 shows muscle activation for the triceps lateral in orange during the same elbow flexion/extension movement. Here, an exponential increase in activation during elbow flexion can be seen, as well as a decrease during elbow extension. This is the opposite of what would be expected since the function of the triceps lateral is to extend the elbow. The expected behavior for the triceps lateral is to show an increase in muscle activation during elbow extension, and a decrease in muscle activation during elbow flexion, with a maximum when the elbow is in a fully extended position [2].
The experimental EMG data for the triceps long can be seen in green in Figure 3. The graph shows the expected activation of the triceps long; a decrease in activation during elbow flexion and an increase in activation during elbow extension, with a maximum when the elbow is fully extended. In addition to this, small unexpected increases in activation can be seen in the data when the elbow is in a maximally flexed position.
Figure 3 shows the experimental data for the brachialis in red, which is not similar to the expected behavior for the brachialis [17]. Large activation values can be seen in the later phase of elbow extension, which is a moment in which the activation of the brachialis should not be increased. The smaller peaks in the graph match the expected muscle activation, showing an increase during elbow flexion and a decrease during elbow extension with a maximum when the elbow is in a fully flexed position.

5.3 Simulated position data

The main output that is analyzed is the position data, which is portrayed by the elbow angle. The simulated elbow angle gives an insight in how well the natural motion of the elbow angle can be tracked by the model. The results for this output vary depending on the reward function that is used during training. In this paragraph, four different conditions for the reward function will be regarded, and their corresponding result in the form of position data. For each condition, the result of the best training out of ten different trainings is shown. The best result was elected using the RMSE value of the experimental and simulated position data. The model was trained on experimental data from the first two repetitions. Then, it extrapolates for the other 8 repetitions of the recording using the same policy. Figure 4 below shows the mean result and standard deviation for the simulated elbow angle of the total of ten repetitions within one training. The result for the first two repetitions for position data can be seen in Appendix E.



(a) Condition 1: position reward

(b) Condition 2: position reward and velocity reward

(c) Condition 3: position reward, velocity reward, and EMG reward

(d) Condition 4: position reward, velocity reward, EMG reward, and acceleration reward

Figure 4: Mean experimental- and simulated elbow angle and standard deviation for condition 1 (a), 2 (b), 3 (c), and 4 (d) during one cycle of elbow flexion/extension, represented in blue and orange, respectively.

In condition 1, the reward function only contains a position component. That means that the agent receives a reward when the position error is small, see Equation 2. The simulated position data, using only a position component in the reward function, can be seen in Figure 4a. The position of the elbow is displayed on the vertical axis as the elbow angle in radians. On the horizontal axis, the time is shown in seconds. The 3 seconds correspond to 300 time steps because the simulation runs at 100 Hz with 10 ms between steps. Within the figure, two different lines can be distinguished. The blue dotted line resembles the mean of the experimentally obtained position data of ten repetitions. In orange, the mean simulated position data can be seen. It can be seen that for condition 1, the orange line tracks the sinusoidal shape of the blue line relatively accurately. However, there are some spikes visible for a larger angle, when the elbow is in a fully flexed position. Also, the simulated data seems to be incapable of reaching the lowest experimental elbow angle values.
Condition 2 consists of a reward function with a position- and a velocity component, as seen in Equation 2 and 3. Figure 4b shows the results for the simulated elbow angle using condition 2. The result is similar to the result for condition 1; the experimental elbow angle is tracked accurately, but the low values at a fully extended

elbow position are not reached. The spikes that were seen at high elbow angle values in condition 1 are not visible in condition 2. Here, the experimental elbow angle is accurately tracked for a fully flexed elbow position. In Figure 4c, the results for the position data can be seen for condition 3, when, in addition to a position- and velocity component, an EMG component is added to the reward function, see Equation 4. It can be seen that the simulated data is now able to track the low elbow angle values. The simulated data seems to be less accurate in tracking the experimental data during both elbow flexion and elbow extension.
Figure 4d shows the results for the simulated position data for condition 4. In this condition, the reward function consists of a position-, velocity-, EMG-, and acceleration component, see Equation 2, 3, 4, and 5. The results are similar to those for condition 3 but slightly more accurate. The experimental elbow angle is accurately tracked during elbow flexion and elbow extension. Some small spikes are visible when the elbow is almost fully flexed.

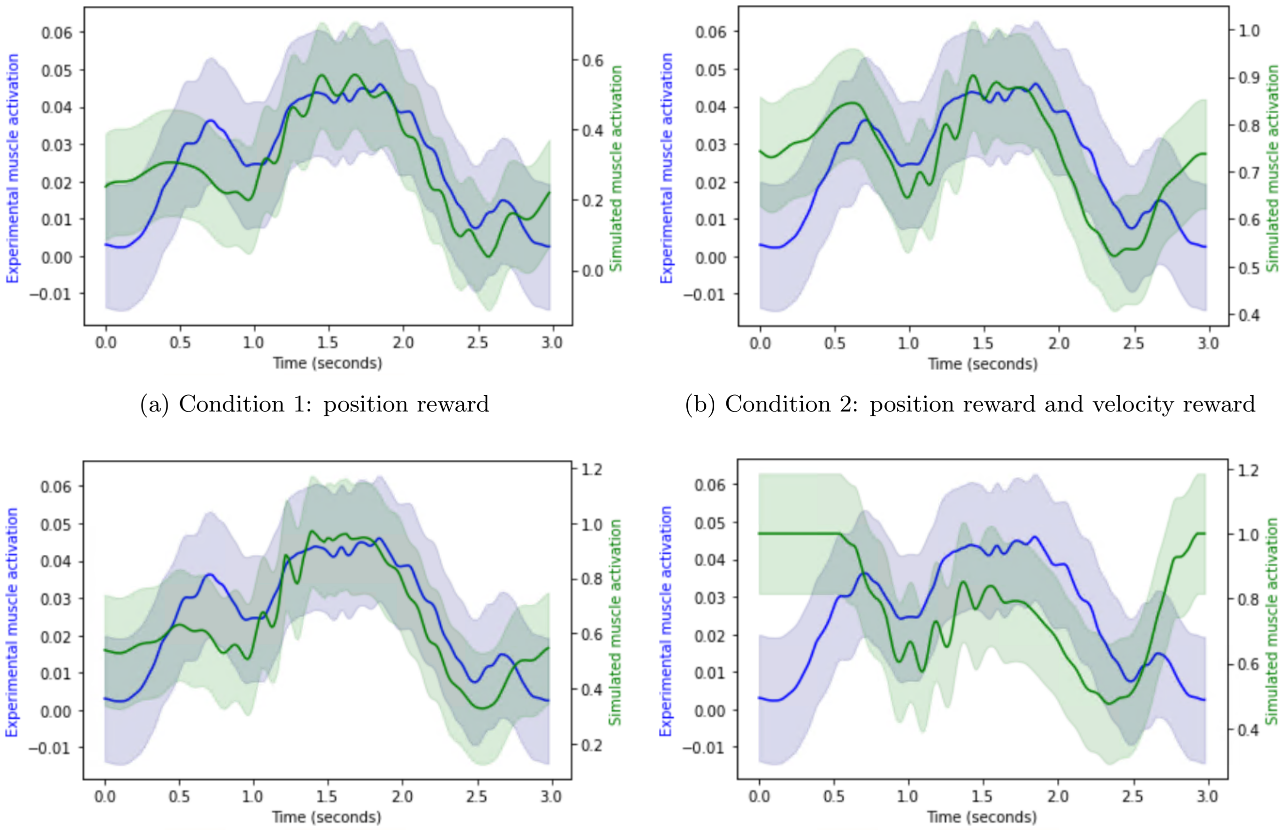
In Table 2 below, the Pearson coefficient and RMSE value can be seen for the experimental and simulated position data in all four conditions. The values in the table result from the mean of data of 10 repetitions, shown in Figure 4. From this table, it can be seen that the Pearson coefficient is the closest to one for condition 4. This means that the correlation between experimental and simulated elbow angle is the strongest for the best training in which the reward function includes a position-, velocity-, EMG-, and acceleration component. From Table 2 it can also be seen that the RMSE value is the lowest for condition 4. Therefore, it can be concluded that the simulated position data best fits the experimental position data in a training where a reward function is used that includes all four components. Using condition 4 leads to the smallest total position error. It should be noted that differences in Pearson coefficients and RMSE values are small, indicating that the extent to which the simulated data fits the experimental data is close between the best results for all four conditions.

Table 2: Pearson coefficients and RMSE values for the best result of four different conditions, indicating the relation between the experimental position data and the simulated position data.

Condition	Pearson coefficient	RMSE [rad]
1	0.9992	0.0337
2	0.9990	0.0317
3	0.9984	0.0456
4	0.9993	0.0294

5.4 Simulated EMG data

Muscle activation during elbow flexion/extension was simulated for the biceps (long head), triceps (lateral head), triceps (long head), and brachialis using the four conditions. In this paragraph, the EMG results for each muscle in all four conditions will be discussed. The simulated EMG data shown here is a result of the training that generated the lowest RMSE value for the position data in each condition, as seen in Figure 4. The simulated muscle activation of the biceps long and triceps long is compared to the corresponding experimental EMG data by means of the Pearson coefficient. The experimental muscle activation for the triceps lateral and the brachialis is not realistic, which can be seen in Figure 3, and is therefore not used for validation. Instead, the simulated muscle activation for these two muscles is evaluated using data from literature as a reference. Just as the experimental EMG data, the simulated EMG data for every muscle is also normalized. For each muscle, the simulated EMG data that is shown is the mean of ten repetitions. The repetition can be seen, from a fully extended elbow at 0.0 seconds, to a fully flexed elbow at 1.7 seconds, and extended again at 3.0 seconds. The simulated and experimental EMG results for the biceps (long), in every condition, can be seen in Figure 5 below.



(a) Condition 1: position reward

(b) Condition 2: position reward and velocity reward

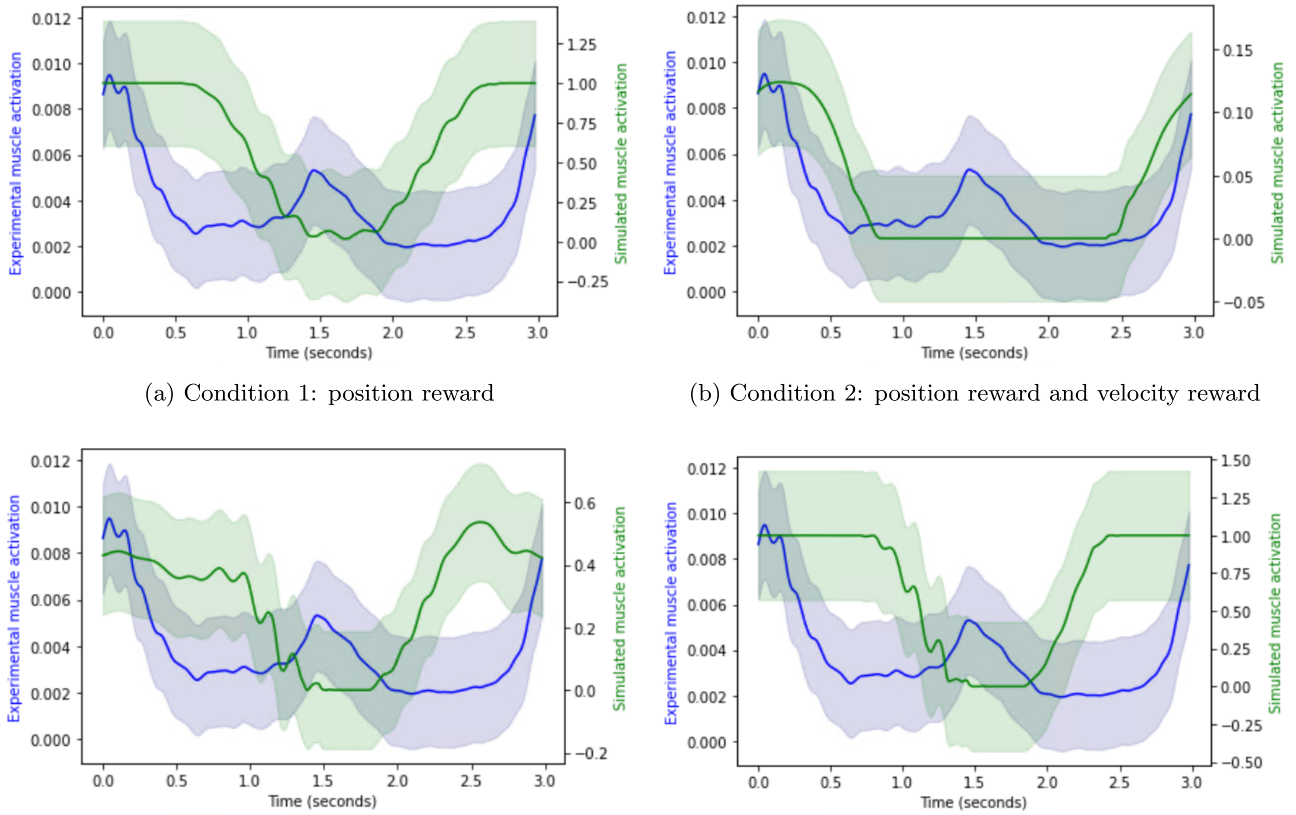
(c) Condition 3: position reward, velocity reward, and EMG reward

(d) Condition 4: position reward, velocity reward, EMG reward, and acceleration reward

Figure 5: Mean experimental (blue)- and simulated (green) muscle activation and standard deviation of the biceps (long) of one cycle of elbow flexion/extension for conditions 1 (a), 2 (b), 3 (c), and 4 (d).

It can be seen that for condition 1, in Figure 5a, the simulated muscle activation is comparable to the experimental muscle activation. Simulated muscle activation shows a maximum during elbow flexion after 0.7 seconds, and another maximum when the elbow is fully flexed, after 1.7 seconds. This is also seen in the experimental muscle activation. When the elbow starts extending, after 1.9 seconds, both experimental and simulated muscle activation decrease. The experimental values for muscle activation are different from the simulated values. This is because the model is not personalized, which means that the muscle forces do not coincide.
Simulated muscle activation for condition 2, which can be seen in Figure 5b, is similar to condition 1, with the difference that the maximum after 0.7 seconds is higher and overall activation values are higher throughout the movement. For condition 2, simulated muscle activation also increases during flexion, and decreases during extension.
The results for condition 3, which can be seen in Figure 5c, are comparable to the results for condition 1. The shape of the simulated muscle activation graph is similar to the shape of the experimental muscle activation graph. The greatest difference with the results for condition 1 is the value for simulated muscle activation, which are twice as high as for condition 1.
Figure 5d shows the results for condition 4. The simulated muscle activation is not similar to the experimental muscle activation when the elbow is in an extended position. Peaks in simulated muscle activation can be seen before 0.5 seconds and after 2.7 seconds. At these times, experimental muscle activation is minimal. For a flexed elbow, the results are more accurate. A smaller maximum can be seen after 1.5 seconds when the elbow is in a fully flexed position.

In addition to the biceps, the simulated EMG results for the triceps (long) are compared to the corresponding experimental EMG results. This can be seen for every condition in Figure 6. The results for the triceps (long) also consist of the mean of all repetitions of the best training, and the standard deviation. Figure 6 shows the same part of the movement as Figure 5; elbow flexion in the first 1.7 seconds, and elbow extension after that.



(a) Condition 1: position reward

(b) Condition 2: position reward and velocity reward

(c) Condition 3: position reward, velocity reward, and EMG reward

(d) Condition 4: position reward, velocity reward, EMG reward, and acceleration reward

Figure 6: Mean experimental (blue)- and simulated (green) muscle activation and standard deviation of the triceps (long) of one cycle of elbow flexion/extension for conditions 1 (a), 2 (b), 3 (c), and 4 (d).

In Figure 6a, the results for experimental and simulated muscle activation for the triceps long in condition 1 can be seen. A clear minimum in the simulated data can be seen when the elbow is fully flexed, after 1.7 seconds. Before this point, simulated data shows a steep decrease in muscle activation. After this point, when the elbow is extending, the simulated data shows a steep increase in activation. The simulated data shows an earlier increase and a later decrease in muscle activation than the experimental data. The increase in experimental activation which is seen at 1.7 seconds might be caused by co-contraction; this is not included in the simulated data.

Figure 6b shows that for condition 2 the simulated data is similar to the experimental data. Peaks can be seen in muscle activation when the elbow is extending and there is no activation during elbow flexion. Also, the increase in simulated muscle activation begins later, and the decrease starts earlier than in condition 1. This timing matches the experimental data more than in condition 1. The values for simulated muscle activation in this condition are around 10 times lower than the values in condition 1.

The results for simulated muscle activation for condition 3 can be seen in Figure 6c. A shape similar to that for condition 1 is visible. There is a decrease in simulated muscle activation during elbow flexion and an increase during elbow extension. Again, the period of minimal activation is smaller than that of the experimental data. Values for the simulated muscle activation are higher than for condition 2 but not as high as for condition 1.

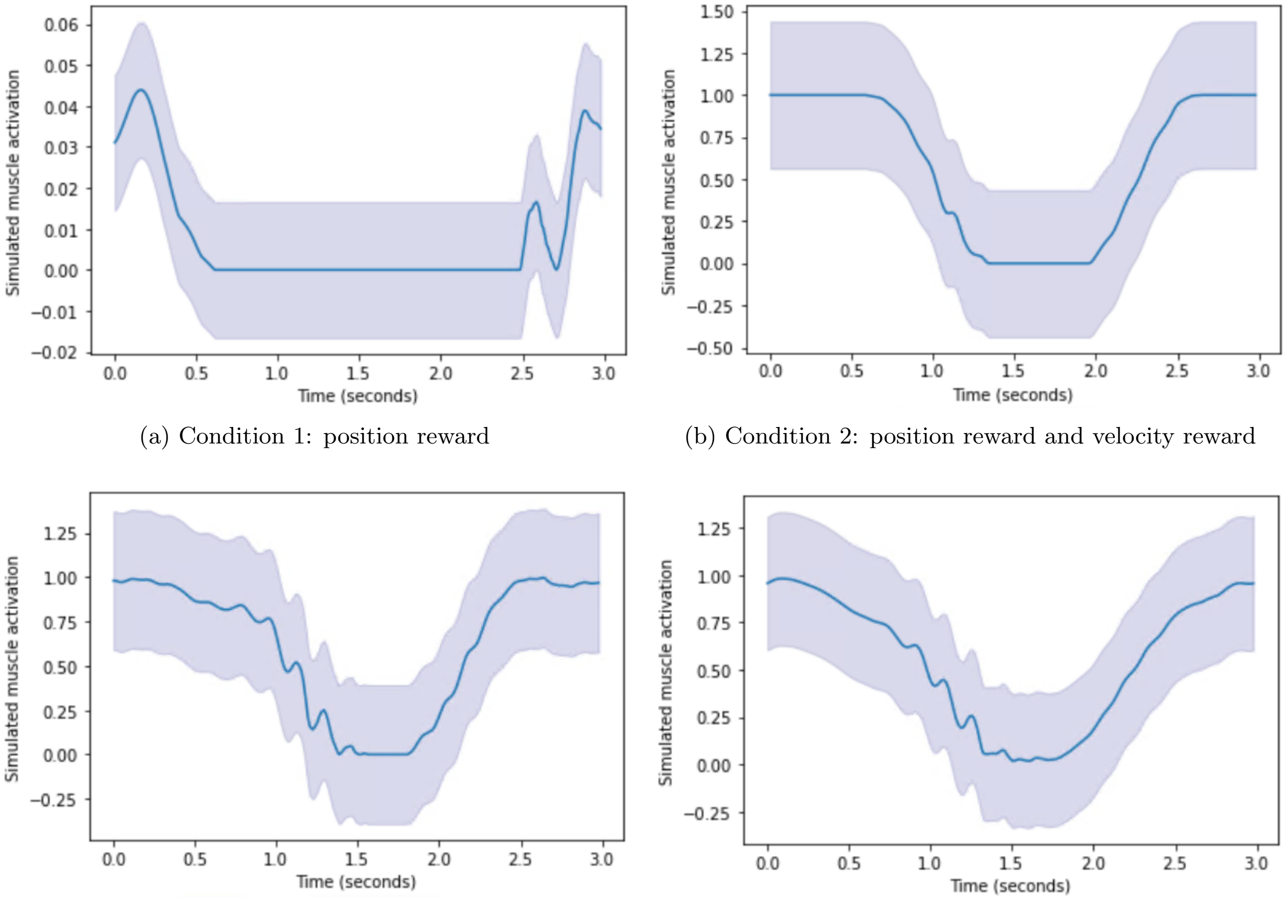
The simulated data for condition 4, which can be seen in Figure 6d, is comparable to condition 1. Minimums are seen when the elbow is flexed, and wide maximums when the elbow is extended. Values for simulated muscle activation are as high as for condition 1, reaching maximum activation when the elbow is extending.

To evaluate the accuracy of the simulated muscle activation in each condition, the Pearson coefficient for EMG data of the biceps (long) and triceps (long) was calculated for each condition. The Pearson coefficients can be seen in Table 3 below. The values in this table are based on the results of Figure 5 and 6, so the mean of ten repetitions. From this table, it can be seen that for the biceps (long) the highest Pearson coefficient is found for condition 3. The Pearson coefficient for the mean of all repetitions in the best training of condition 3 is 0.829. This means that there is a strong positive correlation between experimental muscle activation and simulation muscle activation of the biceps for condition 3. For the triceps (long), the highest Pearson coefficient is found for condition 2. The Pearson coefficient for condition 2 is 0.683, indicating a strong positive correlation between experimental and simulated muscle activation.

Table 3: Pearson coefficients for the biceps long and triceps long, in the four different conditions, indicating the correlation between the experimental EMG data and the simulated EMG data.

Condition	Biceps long	triceps long
1	0.804	0.232
2	0.578	0.683
3	0.829	0.0959
4	-0.220	0.130

For the triceps (lateral) and the brachialis, it was not possible to evaluate the accuracy of the simulated muscle activation by using the experimental data as a reference. Therefore, the simulated data was be compared to data from literature. The normalized, simulated muscle activation for the triceps (lateral) and brachialis can be seen in Figure 7 and 8 below for each condition. For these two muscles, the results also show the mean of all repetitions, with a flexed elbow after 1.7 seconds.



(a) Condition 1: position reward

(b) Condition 2: position reward and velocity reward

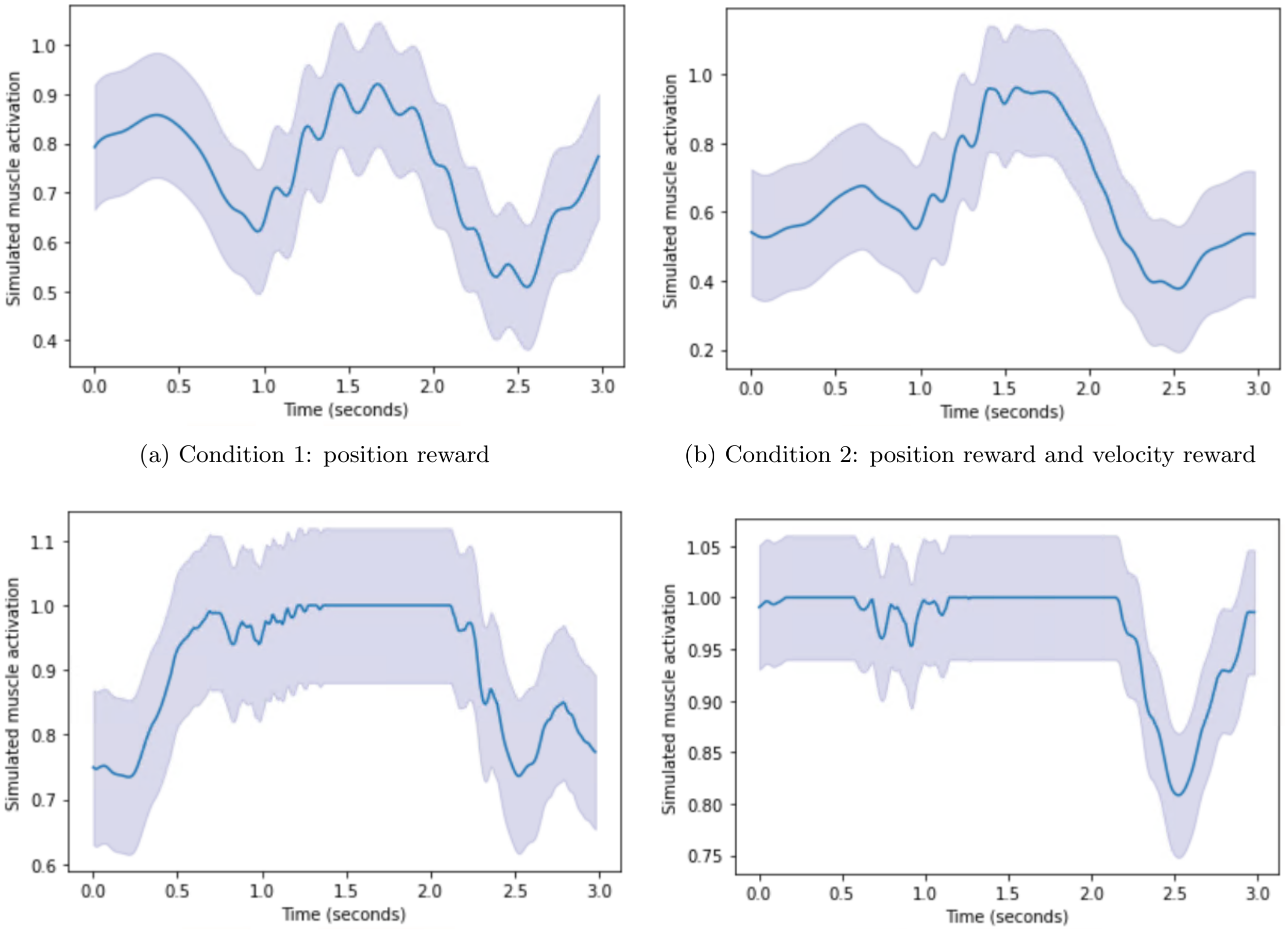
(c) Condition 3: position reward, velocity reward, and EMG reward

(d) Condition 4: position reward, velocity reward, EMG reward, and acceleration reward

Figure 7: Mean simulated muscle activation of the triceps (lateral) and standard deviation during one cycle of elbow flexion/extension for condition 1 (a), 2 (b), 3 (c), and 4 (d).

Figure 7a shows the simulated muscle activation for the triceps lateral for condition 1. Zero activation is seen when the elbow is in a flexed position. Clear peaks are visible when the elbow is fully extended, at 0.0 seconds and after 3.0 seconds. For condition 2, a similar pattern is visible, see Figure 7b. However, the decrease in muscle activation starts later and the increase starts earlier. Also, a plateau of maximal activation is reached for this condition, when the elbow is extended. In Figure 7c, the simulated data for condition 3 can be seen. This result is comparable to that of condition 2, with a decrease in muscle activation during flexion and an increase during extension. The decrease in muscle activation starts slightly earlier than in condition 2. The simulated data for condition 4 is similar to the results for condition 3, but without a plateau of maximal activation, see Figure 7d. The decrease in muscle activation during elbow flexion and the increase during elbow extension is less steep.

The results for the mean simulated muscle activation and standard deviation for the four conditions can be seen in Figure 8 below for the brachialis.



(a) Condition 1: position reward

(b) Condition 2: position reward and velocity reward

(c) Condition 3: position reward, velocity reward, and EMG reward

(d) Condition 4: position reward, velocity reward, EMG reward, and acceleration reward

Figure 8: Mean simulated muscle activation and standard deviation of the brachialis during one cycle of elbow flexion/extension for conditions 1 (a), 2 (b), 3 (c), and 4 (d).

Figure 8a shows the simulated muscle activation of the brachialis for condition 1. The simulated muscle activation for condition 1 shows a maximum when the elbow is in a fully flexed position, with additional maximums when the elbow is in an extended position. This is also the case for condition 2, which can be seen in 8b, but to a lesser extent. An increase in muscle activation can be seen during flexion and a decrease during elbow extension. This is also the case for condition 3, in Figure 8c, with the difference that the maximum activation extends over a longer time. The simulated muscle activation for condition 4 is high during the entire movement, only showing a decrease during elbow extension, after 2.5 seconds. This can be seen in Figure 8d.

If data from literature is used as a reference, the results for the triceps lateral are most accurate in condition 4 [2]. Here, the shape of the graph looks the most natural and there are no long plateaus of maximal activation. However, the value for muscle activation is high, reaching maximum activation when the elbow is fully extended. For the brachialis, the most accurate results are generated using condition 2 [17]. For this condition, muscle activation increases during elbow flexion and decreases during elbow extension in the most natural way.

5.5 Elbow flexion/extension at random speed

The results in the previous paragraph are all the outcomes of the elbow flexion/extension movement with a speed of 40 BPM. In Figure 9 the mean of all repetitions for the simulated elbow angle and the experimental elbow angle can be seen for an elbow flexion/extension movement with random speed. The standard deviation of the result can be seen in Appendix D. The model used to generate this result was trained using the same policy as condition 4 for the movement at 40 BPM, but now for a movement with random speed. It can be seen that the general shape of the simulated elbow angle is similar to that of the experimental elbow angle. However, there are a lot of spikes, indicating a big error in position. The error is the largest for high elbow angle values. The Pearson coefficient is 0.988, which indicates that despite the large error, there is still a strong correlation between the simulated and experimental elbow angle.
The results for muscle activation for elbow flexion/extension at a random speed are not accurate. Experimental muscle activation and simulated muscle activation for the biceps can be seen in Figure 10. There is no similarity between the two graphs. The simulated data shows high activation and a lot of spikes. This was also seen for the other muscles.

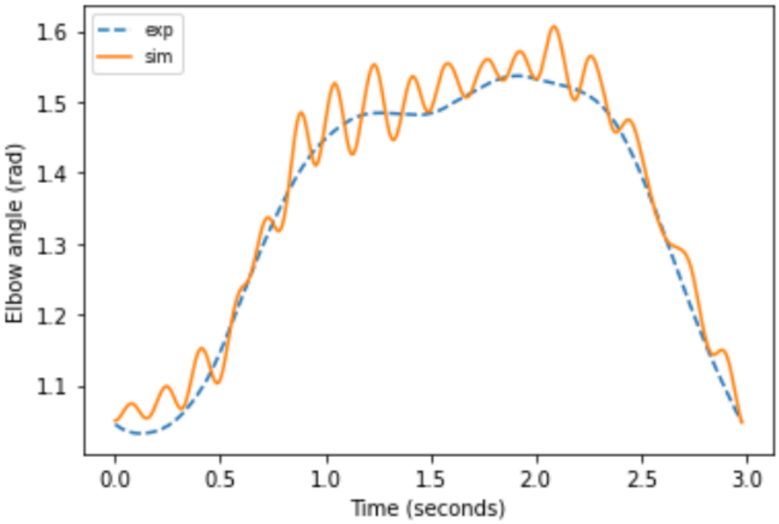


Figure 9: Mean simulated and experimental elbow angle (rad) for elbow flexion/extension at random speed, in orange and blue, respectively.

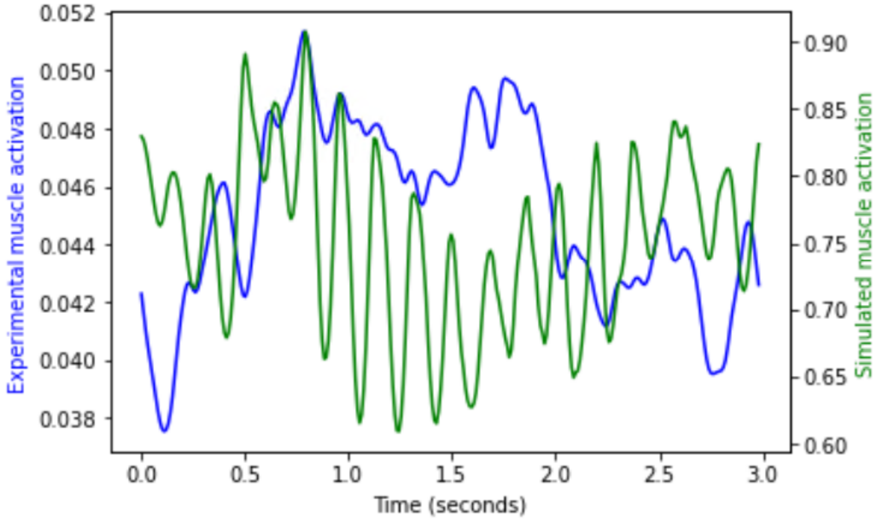


Figure 10: Mean experimental (blue)- and simulated (green) EMG data for the biceps long for elbow flexion/extension at random speed.

6 Discussion

The results show that the used method allows for realistic results. The Myosuite model is capable of generating realistic position data and EMG data, confirming the first hypothesis. The simulated EMG data of the triceps lateral and the brachialis could not be validated due to unrealistic experimental EMG data. The neural network is able to extrapolate similar data accurately and with realistic muscle activation. Extrapolation for a movement at a different speed is less accurate and corresponding muscle activation is not realistic. Simulating more natural EMG data by modifying the reward function, as the second hypothesis states, could not be confirmed. This was caused by the different components not being optimized.

While the outcome seems to be accurate, the results are based on limited data. The experimental data originates from a single subject. Therefore, optimizing the model was also based on the data from this subject. To generalize the results, data from more subjects should be used. A proposition for this is to use subjects with different sports backgrounds, to include different body types and moving abilities. More recordings per subject should be made, and the average of these recordings should be used for the analysis. Recording data from more subjects also helps to identify mistakes in electrode placement, avoiding misleading experimental EMG data.

Despite the high computational speed of Myosuite, time was still a limiting factor during this study. Training the model takes several hours, which limits the possibilities of trying out various policies. Also, the results from different trainings, within a single condition, varied because the starting point was chosen at random. The best result was chosen for every condition out of ten trainings. This choice was made based on the RMSE value for the position data. Because of the limited amount of times that a training could be performed, and the fact that the result is partly based on luck, the result might not always be the best. The best result based on RMSE of the position data was used for analysis in this study. Next to this, also the mean RMSE out of ten trainings and standard deviation for every condition was determined, which can be seen in Table 6 in Appendix B. Here it can be seen that, while condition 4 has the lowest individual RMSE, the mean RMSE for condition 4 is the highest. The mean RMSE of condition 2 is the lowest, with also the lowest standard deviation. This could mean that the scaling factors of the components in the reward function are not optimized and therefore do not have the desired effect.

In the results, it can be seen that there is a difference in amplitude between the experimental EMG data and the simulated EMG data. This is because the model is not personalized. It is therefore important to look at the shape of the EMG data, rather than the quantitative values, which are different for every individual. Changing the muscle force of the model could help the simulated data to match the muscle activation values of the experimental data.

The simulated EMG results show that the neural network model is capable of generating muscle activation for the biceps and the triceps long which is similar to the experimental data. However, the simulated muscle activation is different from what is seen in the experimental muscle activation for the triceps lateral and the brachialis. The probable cause for this is that the experimental data for these muscles is not correct. An explanation for this could be that during the experiment, the co-contracting activity of the muscle was measured. This is the muscle activation of the antagonist muscle, which causes a more accurate movement, and provides stability in the joint [5]. Co-activation of the triceps lateral takes place during elbow flexion, which is seen in the experimental EMG data. The small unexpected peaks in the experimental EMG data for the triceps long may also be caused by this. Another cause for the unrealistic results could be noise from the EMG surface electrodes. It is also possible that the electrode is misplaced, which causes it to record data from a different muscle instead. Also, the large peaks that are seen in the experimental data for the brachialis could be due to movement artifacts. Because the experimental data for the triceps lateral and the brachialis is not realistic, validation using this data is not possible. Therefore, only the simulated muscle activation from the biceps and the triceps long was validated using experimental EMG data. For the triceps (lateral) and the brachialis, the most realistic simulated EMG data was determined using known muscle activation behavior [17] [2].

In addition to the method that uses RMSE to decide which is the best result, the best results for the simulated muscle activation were also chosen manually for each of the four muscles. These results can be seen in Figures 15, 20, 17, and 18 in Appendix C. For the triceps long, the triceps lat, and the brachialis, the best result for muscle activation corresponds to one of the graphs in the results. For the biceps, the best EMG result was generated with a training that did not lead to the lowest RMSE for the position data.

The neural network model sets positive rewards for different components. The most important component is the position component. From the results, it can be seen that this component causes the basic sine shape that the simulated data should make. Adding a velocity component to the reward function helps to make the position data smoother. For the triceps (long), adding a velocity component also helped to create more accurate muscle activation. Adding an EMG component in the reward function was done to minimize the simulated muscle activation to generate more natural EMG data. However, it was concluded that in condition 3, when the EMG component is added to the reward function, the EMG values are not decreased. The mean activation values for each muscle in every condition can be seen in Table 5 in Appendix B. A reason for this could be that the scaling factors that are used for the EMG component in the reward function are not optimized. To increase the effect of the EMG component on muscle activation, the linear scaling factor of this component could be increased. Adding an acceleration component to the reward function generates somewhat smoother position data, but decreases the accuracy in the EMG data significantly for both the biceps and the triceps long, which can be seen by the lower Pearson coefficient for this condition. This might be caused by the design of the reward function of this component, see Appendix F. It should be noted that all the weights that are used for the components in the reward function are not optimized. The best results are generated by trial and error, but not all values were tested. This could explain why the results of condition 4 are not better than those for the other conditions. Optimizing the components' weights and running more trainings per condition could give a clear insight into the effect of the individual components of the reward function.

The simulated position data is similar in shape to the experimental data for movement at a random speed but is not smooth at all. An explanation for this could be the window that is used. The simulated data might be inaccurate at the window starting point. To solve this, the length of the window could be increased, decreasing the number of spikes in the data. Also, the simulated muscle activation for the biceps during a movement with random speed is not accurate, showing a large difference in activation values throughout the movement. A solution for this could be to increase the weight of the EMG component in the reward function, to further minimize the activation. Another solution could be to increase the muscle force that is generated by the model. In this way, lower activation values are needed to perform the same movements.

For future work, it will be useful to see experimental results when more subjects are used, because that will lead to more realistic and general results. It will also be useful to see how applicable the model is for generating other types of movements, using other joints besides the elbow. To do this, more muscles can be added to the model. In this way, more complex and smaller movements can be generated, such as moving individual fingers or picking up an object. Also, movements that require more force can be added and validated, such as lifting a heavy object using elbow flexion. In that way, it can be investigated if the model is capable of generating a higher muscle activation and more force when needed. When simulating more muscles and movements, it can be effective to use a supercomputer. In this way, more trainings can be performed in parallel. To further test the effects of the individual components in the reward function, an optimization of the scaling factors could be carried out. Lastly, a validation of simulated contact-rich movements can be performed. If the results are positive, and a realistic result can be generated consistently, many real-life biomechanical experiments can be replaced with simulations.

7 Conclusion

The results from this study show that with the help of reinforcement learning, Myosuite models can estimate the neural command that is needed to perform tasks. The algorithm is able to generate realistic EMG data for elbow flexion/extension movement.

In the neural network training of this study, the most important reward components are the position reward and the velocity reward, because of their contribution to accurate tracking of the movement with natural EMG.

In the results, it can be seen that it is possible to generate realistic muscle activation using reinforcement learning. The experimental EMG data of the biceps and triceps long is accurate and is in line with the simulated EMG data of the biceps and triceps long. The simulated EMG for the other two muscles did not coincide with the experimental EMG data but made sense physiologically. The model is capable of accurately extrapolating for the movement it was trained on.

In addition, the results show that the policy can extrapolate for other data, but to a lesser extent. The policy is capable of simulating somewhat realistic elbow flexion/extension movement at random speeds, using the same training. Muscle activation data was, however, not accurate for a different speed.

References

[1] Vittorio Caggiano, Huawei Wang, Guillaume Durandau, Massimo Sartori, and Vikash Kumar. Myosuite–a contact-rich simulation suite for musculoskeletal motor control. *arXiv preprint arXiv:2205.13600*, 2022.

[2] Ulysses F Ervilha, Dario Farina, Lars Arendt-Nielsen, and Thomas Graven-Nielsen. Experimental muscle pain changes motor control strategies in dynamic contractions. *Experimental brain research*, 164(2):215–224, 2005.

[3] Hartmut Geyer and Hugh Herr. A muscle-reflex model that encodes principles of legged mechanics produces human walking dynamics and muscle activities. *IEEE Transactions on neural systems and rehabilitation engineering*, 18(3):263–273, 2010.

[4] Masaya Hirashima and Tomomichi Oya. How does the brain solve muscle redundancy? filling the gap between optimization and muscle synergy hypotheses. *Neuroscience research*, 104:80–87, 2016.

[5] S Hirokawa, M Solomonow, Z Luo, Y Lu, and R D'ambrosia. Muscular co-contraction and control of knee stability. *Journal of Electromyography and Kinesiology*, 1(3):199–208, 1991.

[6] Nathanaël Jarrassé, Tommaso Proietti, Vincent Crocher, Johanna Robertson, Anis Sahbani, Guillaume Morel, and Agnes Roby-Brami. Robotic exoskeletons: a perspective for the rehabilitation of arm coordination in stroke patients. *Frontiers in human neuroscience*, 8:947, 2014.

[7] Vittorio La Barbera, Fabio Pardo, Yuval Tassa, Monica Daley, Christopher Richards, Petar Kormushev, and John Hutchinson. Ostrichrl: A musculoskeletal ostrich simulation to study bio-mechanical locomotion. *arXiv preprint arXiv:2112.06061*, 2021.

[8] Laëtitia Matignon, Guillaume J Laurent, and Nadine Le Fort-Piat. Reward function and initial values: Better choices for accelerated goal-directed reinforcement learning. In *International Conference on Artificial Neural Networks*, pages 840–849. Springer, 2006.

[9] Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 2021.

[10] Ashish Rana. Introduction: Reinforcement learning with openai gym. *Towards Data Science*, 2018.

[11] Sarah A Roelker, Elena J Caruthers, Rachel K Hall, Nicholas C Pelz, Ajit MW Chaudhari, and Robert A Siston. Effects of optimization technique on simulated muscle activations and forces. *Journal of Applied Biomechanics*, 36(4):259–278, 2020.

[12] Katherine R Saul, Xiao Hu, Craig M Goehler, Meghan E Vidt, Melissa Daly, Anca Velisar, and Wendy M Murray. Benchmarking of dynamic simulation predictions in two software platforms using an upper limb musculoskeletal model. *Computer methods in biomechanics and biomedical engineering*, 18(13):1445–1458, 2015.

[13] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

[14] Ajay Seth, Jennifer L Hicks, Thomas K Uchida, Ayman Habib, Christopher L Dembia, James J Dunne, Carmichael F Ong, Matthew S DeMers, Apoorva Rajagopal, Matthew Millard, et al. Opensim: Simulating musculoskeletal dynamics and neuromuscular control to study human and animal movement. *PLoS computational biology*, 14(7):e1006223, 2018.

[15] Weiguang Si, Sung-Hee Lee, Eftychios Sifakis, and Demetri Terzopoulos. Realistic biomechanical simulation and control of human swimming. *ACM Transactions on Graphics (TOG)*, 34(1):1–15, 2014.

[16] Seungmoon Song, Łukasz Kidziński, Xue Bin Peng, Carmichael Ong, Jennifer Hicks, Sergey Levine, Christopher G Atkeson, and Scott L Delp. Deep reinforcement learning for modeling human locomotion control in neuromechanical simulation. *Journal of neuroengineering and rehabilitation*, 18(1):1–17, 2021.

[17] Didier Staudenmann and Wolfgang Taube. Brachialis muscle activity can be assessed with surface electromyography. *Journal of Electromyography and Kinesiology*, 25(2):199–204, 2015.

[18] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction.* MIT press, 2018.

[19] Ivan Vujaklija, Dario Farina, and Oskar C Aszmann. New developments in prosthetic arm systems. *Orthopedic research and reviews*, 8:31, 2016.

[20] Huawei Wang, Vittorio Caggiano, Guillaume Durandau, Massimo Sartori, and Vikash Kumar. Myosim: Fast and physiologically realistic mujoco models for musculoskeletal and exoskeletal studies. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 8104–8111. IEEE, 2022.

A Appendix: Experimental protocol

Below, the protocol that was used during the experiment can be seen.

Preparation

Turn on the motion capture system (there are 3 on-switches). Put the chair in the middle of the camera space. Install 2 video cameras pointed at the chair. (Front- and side view). Connect pc to the big screen (black cord). Select in Qualisys: new project. Locate the two video cameras.

Calibration

Place (0,0) on the floor. Use the calibration set for this. Use the wand; move it around the area where the markers will be. Click the wand button on pc. Turn on the EMG system (Delsys). Put reflective markers on tape + separate them. Put stickers on EMG sensors (black boxes with cables attached to them).

Skin preparation

Before the EMG sensors are placed, the hair on the arm needs to be removed. The arm also needs to be cleaned. Remove dead skin cells (with gel and paper, hard as possible).

Find the muscles you want to measure on the arm, and mark them with a pen. After that, put EMG sensors on the body on the corresponding muscles.

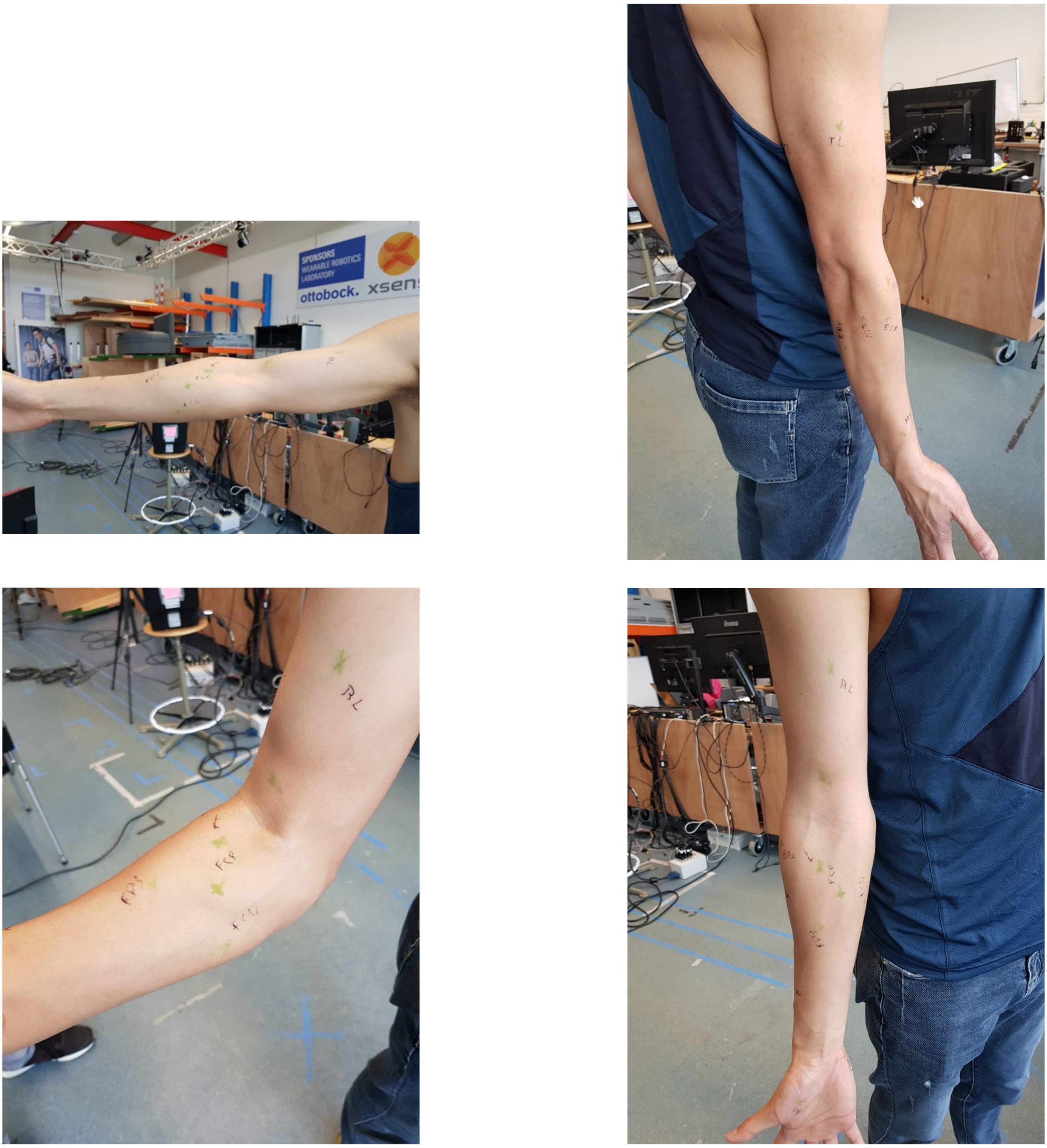
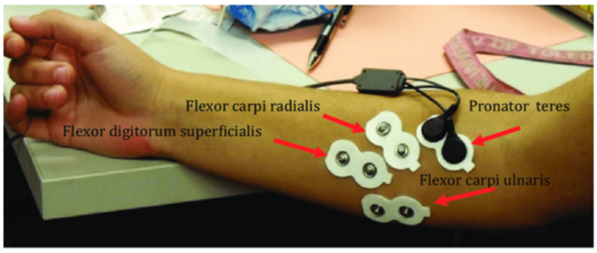


Figure 11: Muscle positions for EMG surface electrodes marked with a pen.

EMG

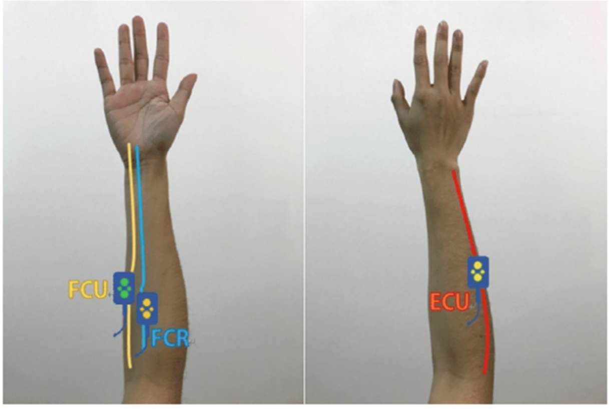
Muscles to measure:
biceps (long),
triceps (x2: lateral+ long),
brachialis,
extensor carpi ulnaris,
extensor carpi radialis,
apl (abductor pollicis longus),
fdp (flexor digitorum profundus),
fcr (flexor carpi radialis),
fcu (flexor carpi ulnaris),
fds (flexor digitorum superficialis),
opponens pollicis,
odm (opponens digi minimi),
pronator teres

EMG sensor placement



Placement of bipolar surface electrodes over the muscle bellies of the muscles of interest.

Figure 12: Placement of surface electrodes over muscle bellies



The EMG electrodes placed at the specific muscles including extensor carpi ulnaris (ECU), flexor carpi ulnaris (FCU), and flexor carpi radialis (FCR).

Figure 13: EMG electrodes placement ECU, FCU, FCR

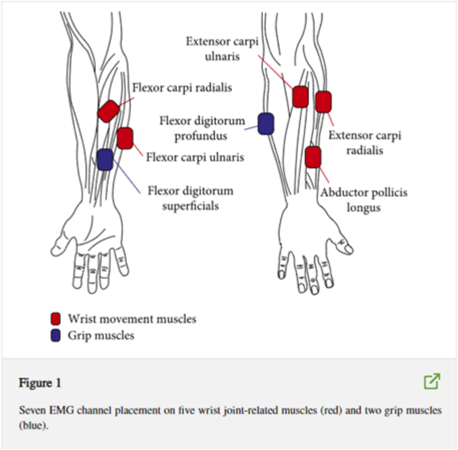


Figure 1

Seven EMG channel placement on five wrist joint-related muscles (red) and two grip muscles (blue).

Figure 14: EMG placement on wrist movement muscles and grip muscles

Use bigger sensors for larger muscles. The box attached to the little sensors should not be placed on a muscle. The number on the EMG sensor corresponds with the following muscle:
1. Biceps
2. Triceps lateral
3. FCU
4. Brachialis
6. Triceps long

Sensors 5 and 7 are broken.

8. EDP
9. ECU
10. FCR
11. FDS
12. PT
13. APL
14. OP
15. ECR
16. ODM

Marker placement
After the EMG sensors are placed on the body, the reflective markers are placed in the following locations:

Acromion (top and bottom),
Clavicle (lateral and medial),
C7,
Back (three markers),
Upper arm (three markers),
Elbow (lateral and medial),
Lower arm (three markers),
Wrist (radial and ulnar),

Nail of every finger,
MCP joint of every finger,
PIP joint of every finger.

Start analog running

Plot every muscle separately to see if the corresponding EMG sensor is working. If one is not working, try turning the little box off and on again. Then: pair it again. Check if all movements correspond to the EMG activation Tape all the cables and sensors to the arm (tight) with a rubber band or tape around the entire arm. To record data, click capture. First: static recording (2 positions: elbow in 90 degrees and 0 degrees (fully extended along body)). Record the MVC of every muscle

Data recording

For the entire recording, the right hand/arm was used. The subject sits in a chair.

Movements that are recorded:

Table 4: Movements that were recorded during the experiment.

Joint	Movement	Speed	Time	Comment
Elbow	flexion - extension	40 BPM	10 reps	-
Elbow	flexion - extension	40 BPM	10 reps	with 2.5 kg held in hand
Elbow	flexion - extension	-	40 sec	random speeds
Elbow	flexion - extension	-	40 sec	random speeds, with 2.5 kg
Wrist	flexion - extension	40 BPM	10 reps	
Wrist	deviation	50 BPM	10 reps	-
Wrist	rotation	50 BPM	10 reps	-
Hand	opening - closing	50 BPM	10 reps	-
Fingers	touching every finger with thumb	55 BPM	10 reps (each finger)	-
All	grab stick. Lift it, put it back	-	10 reps	-
Elbow	Flexion. Hold weight (5 kg) in hand	-	2 min, 3 reps	2 min rest in between

Data processing

Link the names (trajectories) to the markers:
C7, clavicleMed, clavLat, back1, back2, back3, acroniumTop, acroniumBot, up1, up2, up3 etc.

Make a new model in Qualisys, so that you can apply it to every recording.

Trajectory names

Tip1 : fingernail of thumb
Tip2 : fingernail of index finger
Tip3 : fingernail of middle finger
Tip4 : fingernail of ring finger
Tip5 : fingernail of little finger
MCP1 : knuckle of the thumb
MCP2 : knuckle of the index finger (MCP joint)
MCP3 : knuckle of the middle finger
MCP4 : knuckle of the ring finger
MCP5 : knuckle of the little finger

PIP1 : PIP joint of the thumb
PIP2 : PIP joint of the index finger
PIP3 : PIP joint of the middle finger
PIP4 : PIP joint of the ring finger
PIP5 : PIP joint of the little finger
Back1 : upper back marker
Back2 : middle back marker
Back 3 : lower back marker
UP2 : upper upper arm marker
UP3 : middle upper arm marker
UP1 : lower upper arm marker
LOW1 : lower lower arm radial marker
LOW2 : lower lower arm ulnar marker
LOW3 : upper lower arm marker

B Appendix: Tables

Table 5: Mean simulated EMG values of the four muscles for each reward function condition.

	Condition 1	Condition 2	Condition 3	Condition 4
Biceps	358	445	479	463
Triceps lateral	335	314	312	324
Triceps long	347	226	432	374
Brachialis	572	577	634	522

Table 6: Mean RMSE values and standard deviation of the experimental and simulated position data for each reward function condition.

	Condition 1	Condition 2	Condition 3	Condition 4
Mean RMSE	0.0860	0.0776	0.0968	0.106
STD	0.0344	0.0259	0.0430	0.0486

C Appendix: Best simulated EMG results

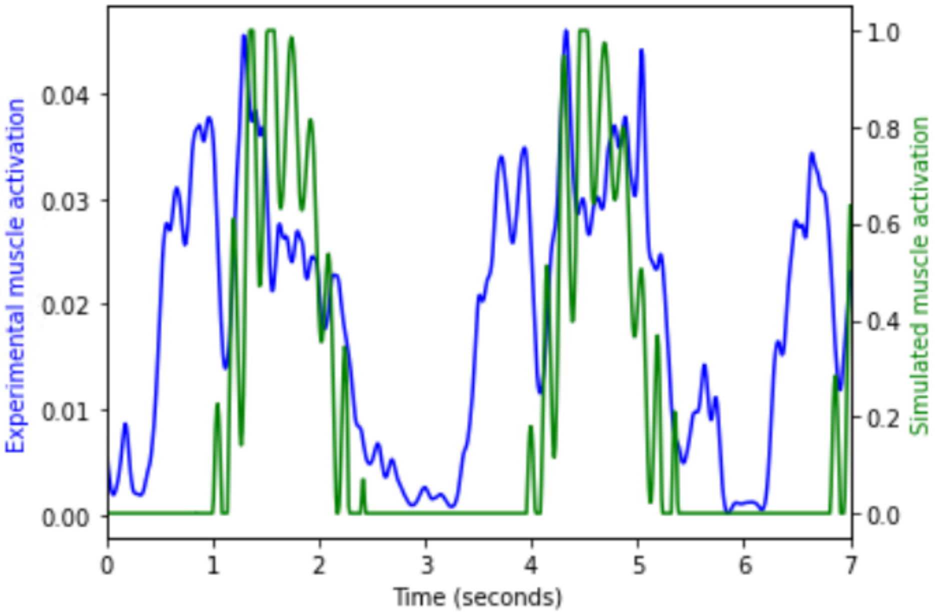


Figure 15: Best simulated muscle activation for the biceps, corresponding to training 10, using condition 2.

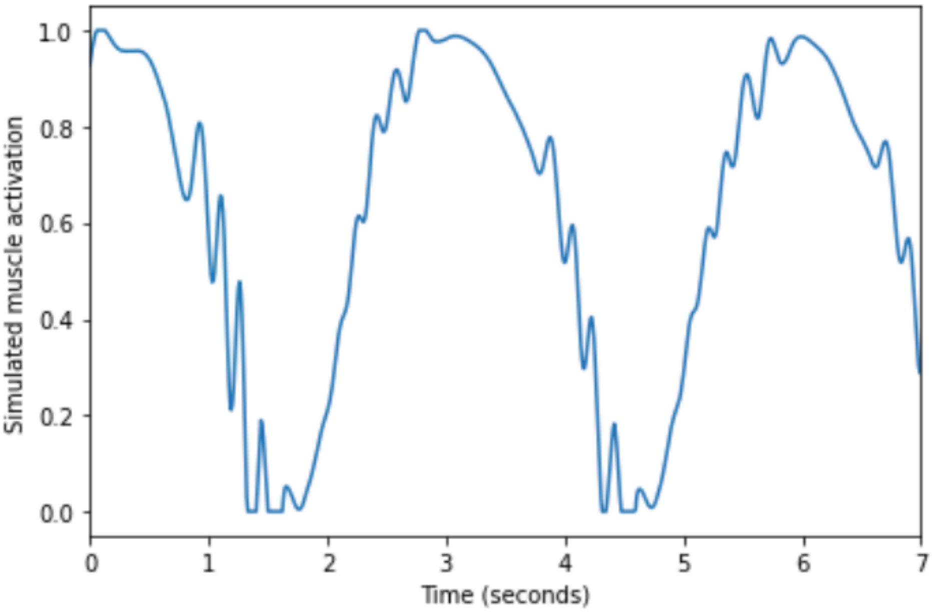


Figure 16: Best simulated muscle activation for the triceps lat, corresponding to training 2, using condition 4.

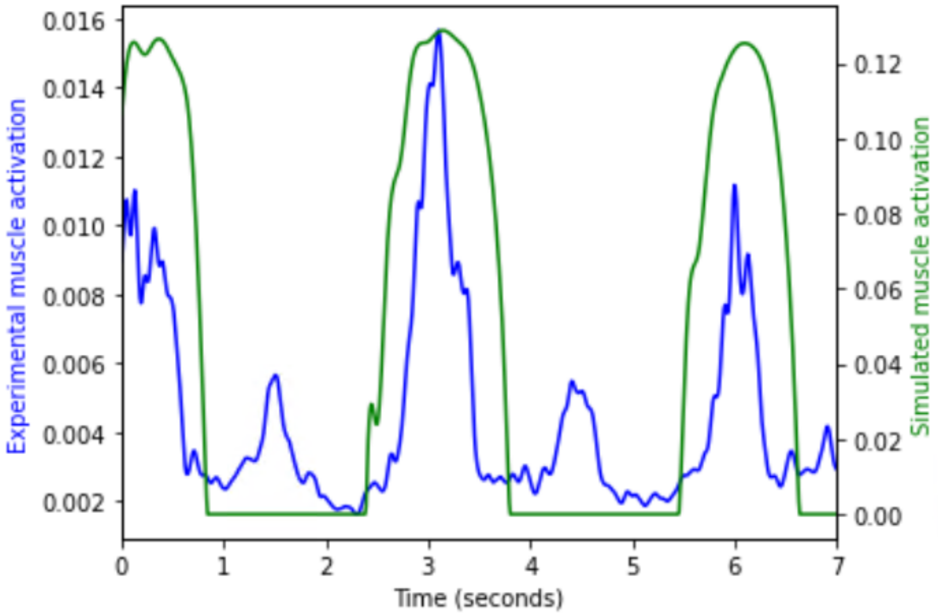


Figure 17: Best simulated muscle activation for the triceps long, corresponding to training 2, using condition 2.

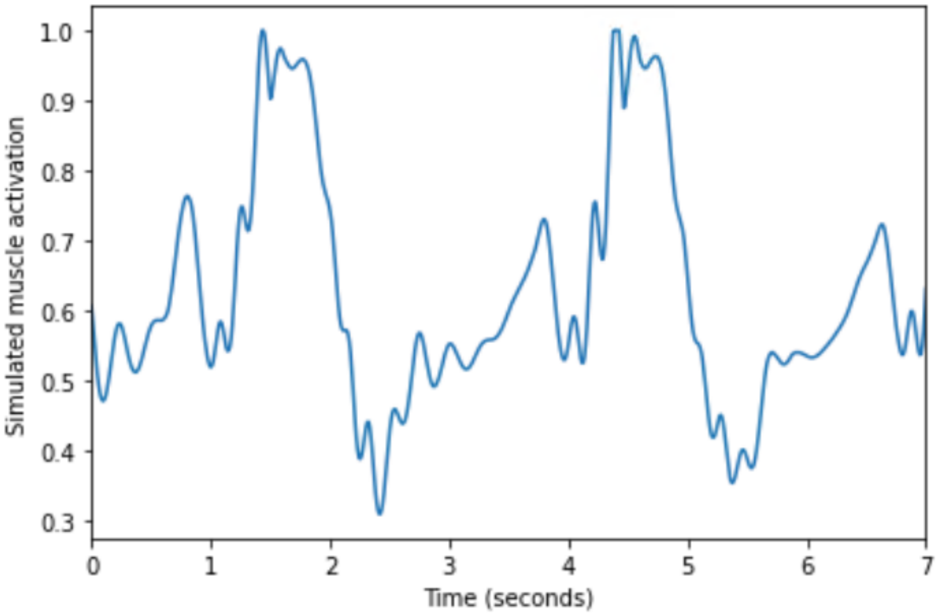


Figure 18: Best simulated muscle activation for the brachialis, corresponding to training 2, using condition 2.

D Appendix: Results for a movement at random speed

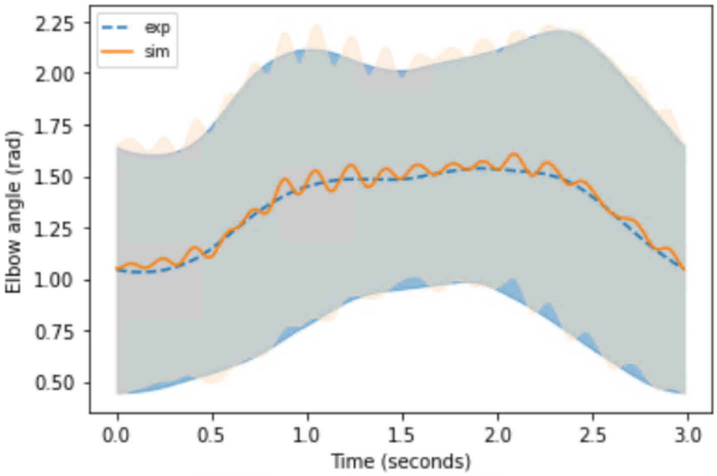


Figure 19: Position results for an elbow flexion/extension movement at random speed, displayed with standard deviation. The result is generated using condition 4.

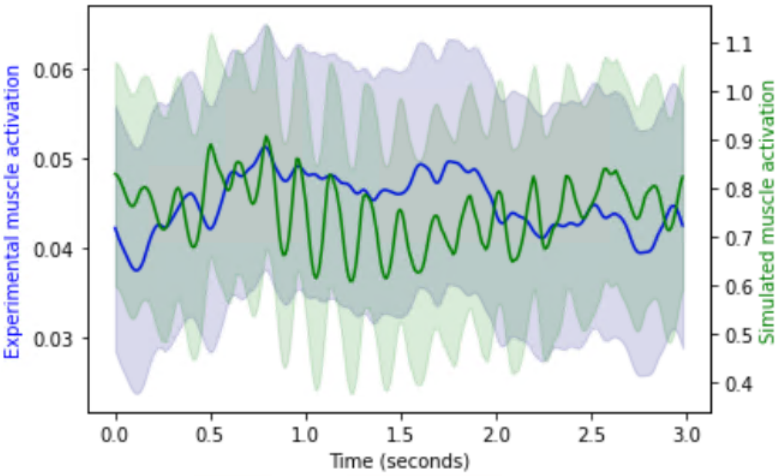
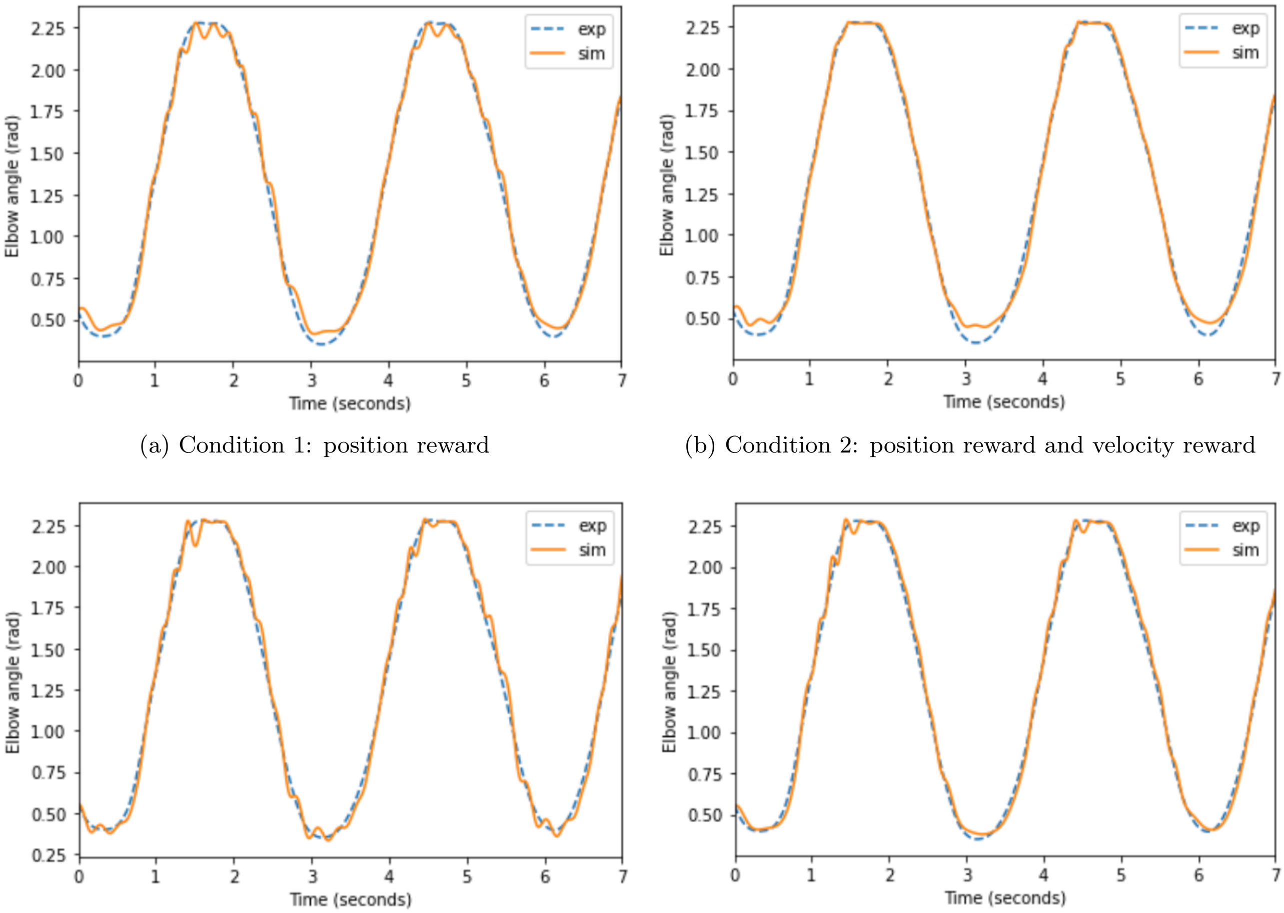


Figure 20: Simulated muscle activation of the Biceps (long) for an elbow flexion/extension movement at a random speed. The result is generated using condition 4.

E Appendix: Results for first two repetitions



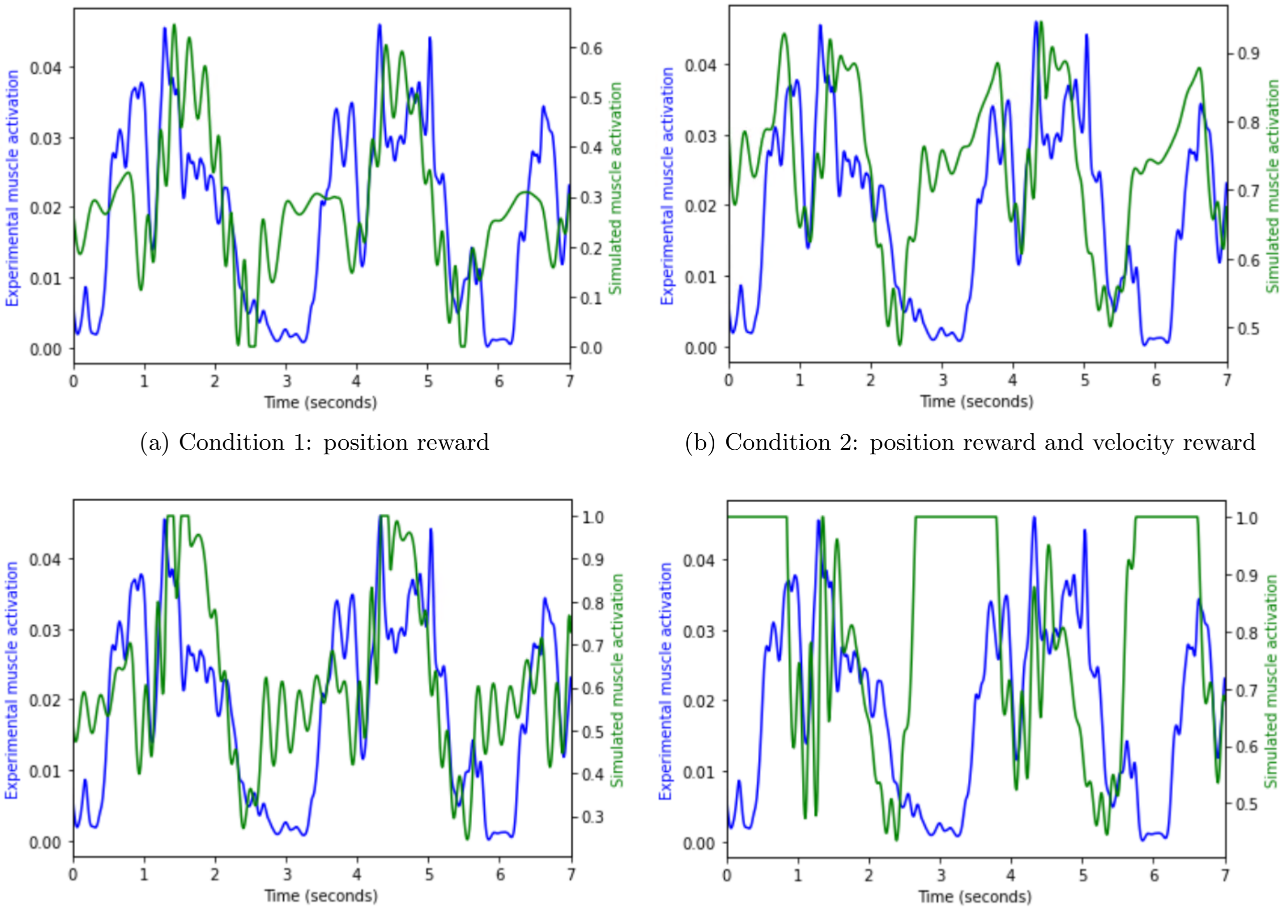
(a) Condition 1: position reward

(b) Condition 2: position reward and velocity reward

(c) Condition 3: position reward, velocity reward, and EMG reward

(d) Condition 4: position reward, velocity reward, EMG reward, and acceleration reward

Figure 21: Experimental- and simulated elbow angle for condition 1 (a), 2 (b), 3 (c), and 4 (d) during first two cycles of elbow flexion/extension, represented in blue and orange, respectively.



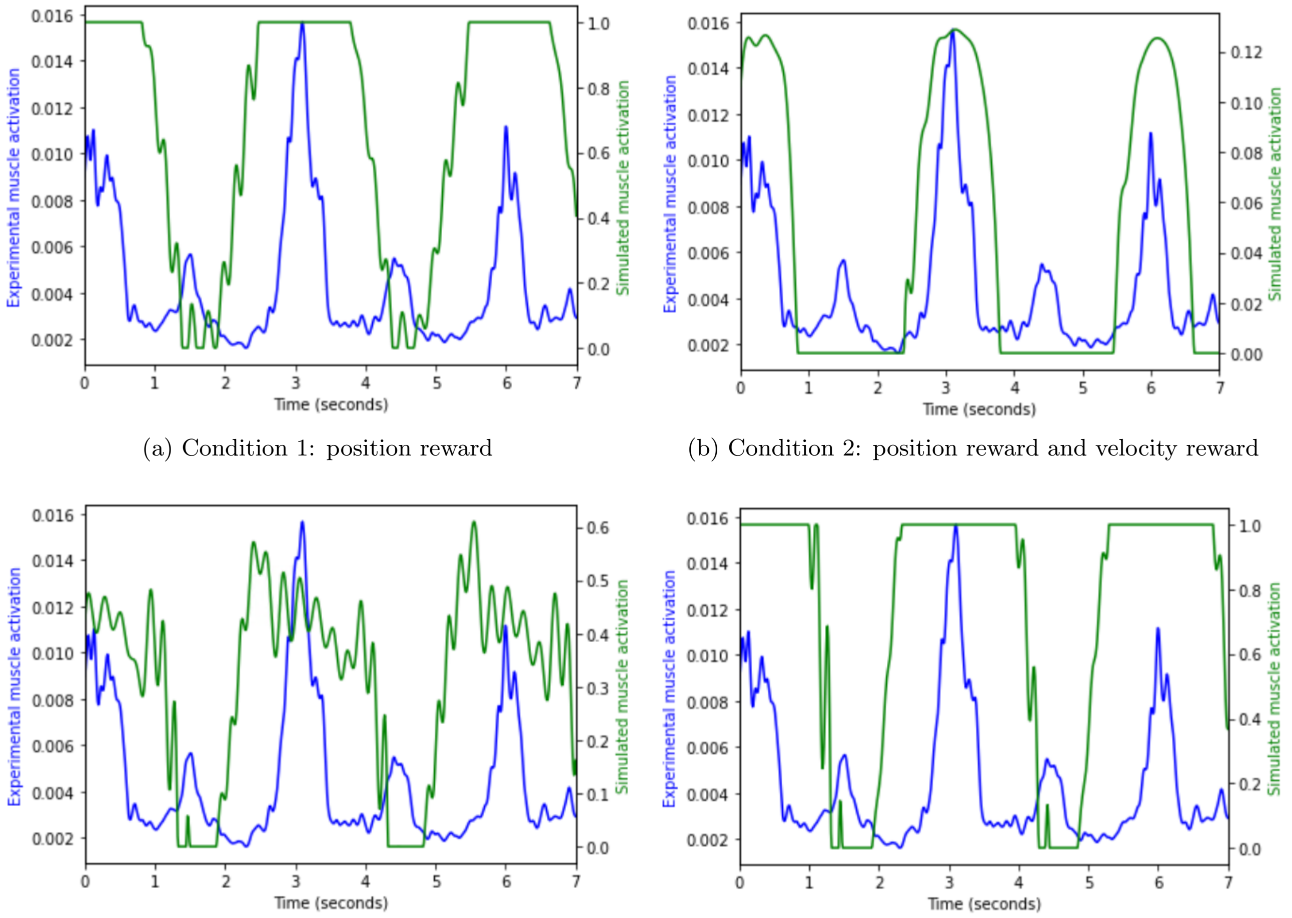
(a) Condition 1: position reward

(b) Condition 2: position reward and velocity reward

(c) Condition 3: position reward, velocity reward, and EMG reward

(d) Condition 4: position reward, velocity reward, EMG reward, and acceleration reward

Figure 22: Experimental (blue)- and simulated (green) muscle activation of the biceps (long) of the first two cycles of elbow flexion/extension for conditions 1 (a), 2 (b), 3 (c), and 4 (d).



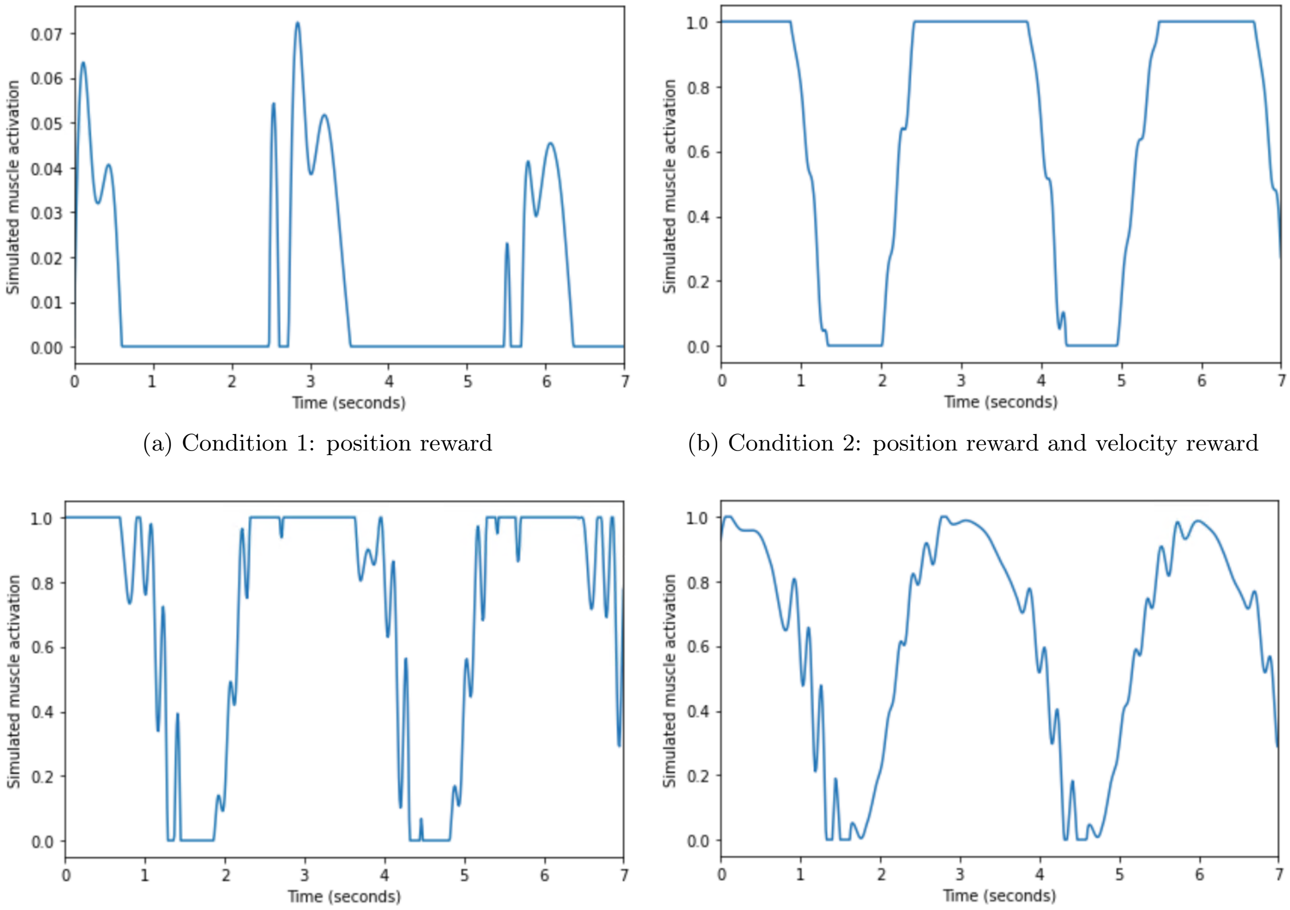
(a) Condition 1: position reward

(b) Condition 2: position reward and velocity reward

(c) Condition 3: position reward, velocity reward, and EMG reward

(d) Condition 4: position reward, velocity reward, EMG reward, and acceleration reward

Figure 23: Experimental (blue)- and simulated (green) muscle activation of the triceps (long) of the first two cycles of elbow flexion/extension for conditions 1 (a), 2 (b), 3 (c), and 4 (d).



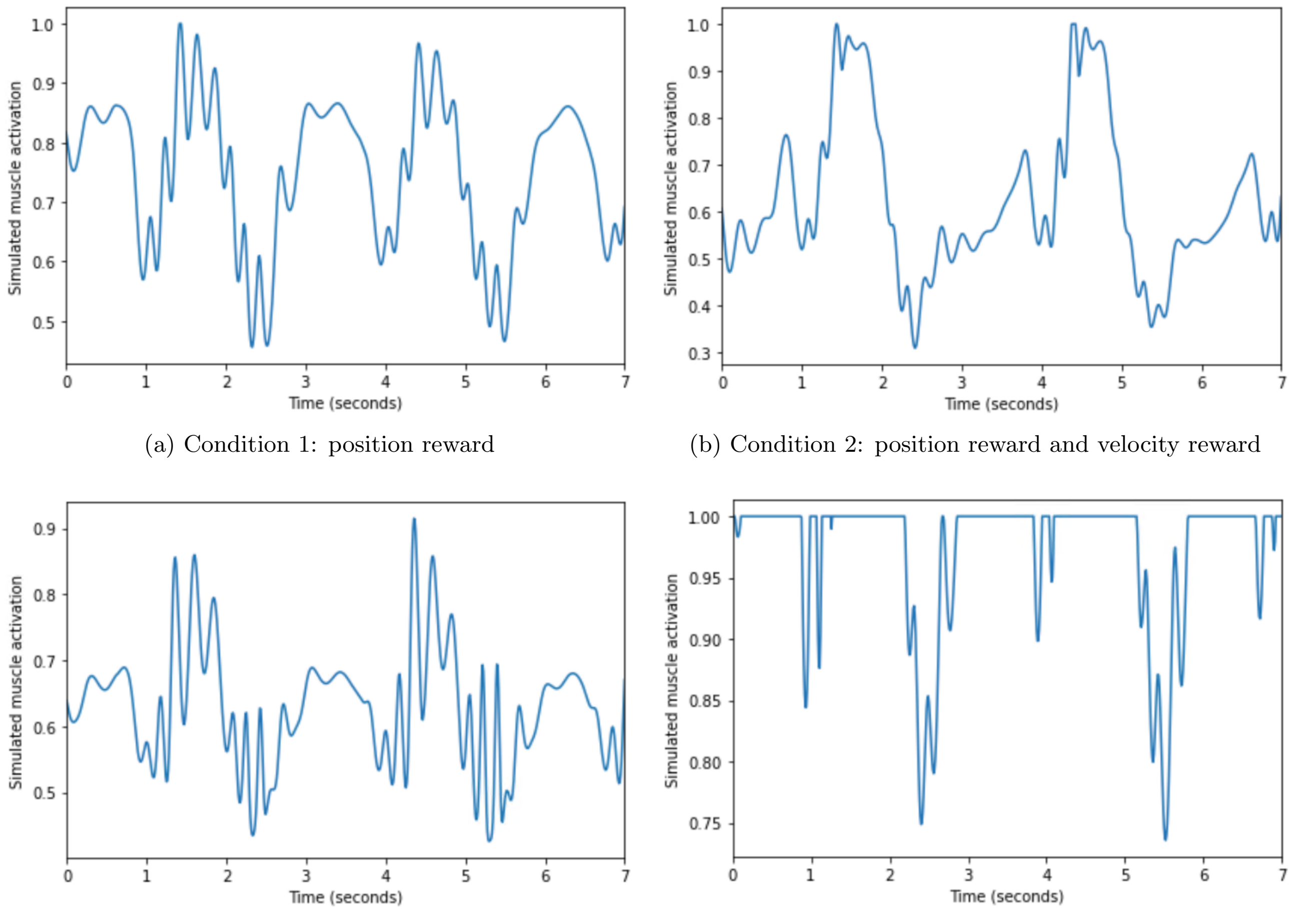
(a) Condition 1: position reward

(b) Condition 2: position reward and velocity reward

(c) Condition 3: position reward, velocity reward, and EMG reward

(d) Condition 4: position reward, velocity reward, EMG reward, and acceleration reward

Figure 24: Simulated muscle activation of the triceps (lateral) during the first two cycles of elbow flexion/extension for condition 1 (a), 2 (b), 3 (c), and 4 (d).



(a) Condition 1: position reward

(b) Condition 2: position reward and velocity reward

(c) Condition 3: position reward, velocity reward, and EMG reward

(d) Condition 4: position reward, velocity reward, EMG reward, and acceleration reward

Figure 25: Simulated muscle activation of the brachialis during the first two cycles of elbow flexion/extension for conditions 1 (a), 2 (b), 3 (c), and 4 (d).

F Appendix: Reward function design

Each of the four reward functions has the same design. The design uses an exponential function, which was suggested for a reward function by Matignon et al [8]. This function leads to low reward values if the agent is far from the goal, and shows a reward gradient around the goal. It causes the learning process to accelerate because there are no dead ends.

F.1 Position reward

The reward function for the position component is designed as:

$$position\ reward = w_{\text{pos}} * e^{-s_{\text{pos}} * err1}. \tag{6}$$

The function of the position component is to minimize the position error between the simulated and experimental data. The linear scaling factor $\mathrm{w_{pos}}$ has a value of 1. This is to indicate that position is the most significant component for tracking the elbow angle. The reward increases if the position error decreases. This relation is exponential, which means that the reward value increases at a higher rate as the error is getting smaller. This motivates the agent to decrease the error as much as possible, even if it is already small. At high position error values (higher than 0.15), the reward gradient is almost uniform. This would not be the case if the reward function only contains a linear component.
The shape of the curve is determined by the exponential scaling factor $\mathrm{s_{pos}}$, which is set at 40. The curve that arises from these scaling factors can be seen in Figure 26 below. In this figure, the reward value is depicted on the vertical axis and the position error on the horizontal axis. It can be seen that the reward value starts to increase significantly when the error is smaller than 0.10. The maximum reward value of 1 is reached if the position error is zero.

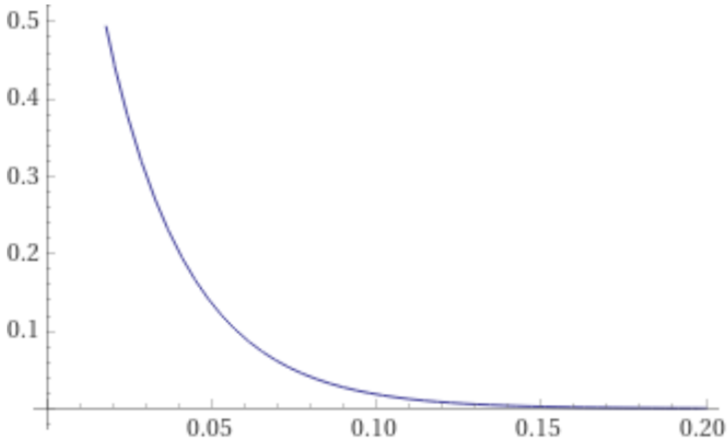
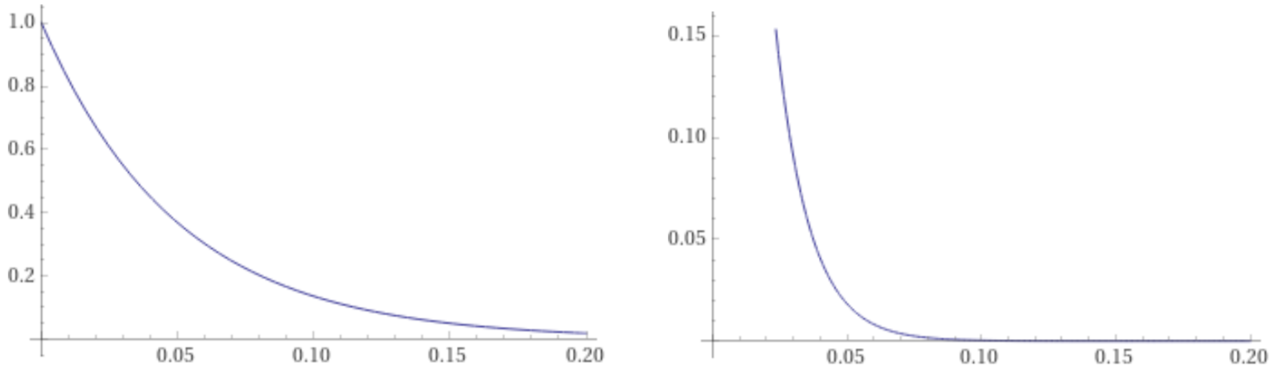


Figure 26: Curve for the reward value for the position component, using $\mathrm{s_{pos}}$= 40.

Changing the value for $\mathrm{s_{pos}}$ changes the speed with which the reward value increases as the position error decreases. The maximum reward value for the position component is always 1. In Figure 27 below, the curve for reward value can be seen if $\mathrm{s_{pos}} = 20$ and $\mathrm{s_{pos}} = 80$ would be used.



(a) $\mathrm{s}_{\mathrm{pos}}$= 20.

(b) $\mathrm{s}_{\mathrm{pos}}$= 80.

Figure 27: Curve for the reward value for the position component, using $\mathrm{s}_{\mathrm{pos}}$= 20 (a) and $\mathrm{s}_{\mathrm{pos}}$= 80.

F.2 Velocity reward

The equation for the reward function for the velocity component is defined as:

$$velocity\ reward = w_{\mathrm{vel}} * e^{-s_{\mathrm{vel}} * err2}. \tag{7}$$

The purpose of the velocity component is to minimize the velocity error between the simulated and experimental data to make the simulated muscle activation more natural.

Just as the reward function for the position component, Equation 7 consists of a linear and an exponential scaling factor. Since the position component is the most significant, the linear scaling factor for the velocity component $\mathrm{w}_{\mathrm{vel}}$ was set to 0.5. The exponential scaling factor for the velocity component $\mathrm{s}_{\mathrm{vel}}$ was set to 20. This was done to have the agent reach significant reward values if the velocity error was close to zero, but not zero. The result of the curve for the velocity reward value can be seen in Figure 28 below. The reward value is depicted on the vertical axis, and the velocity error on the horizontal axis. It can be seen that the maximum reward value is 0.5, which is reached when the velocity error is zero.

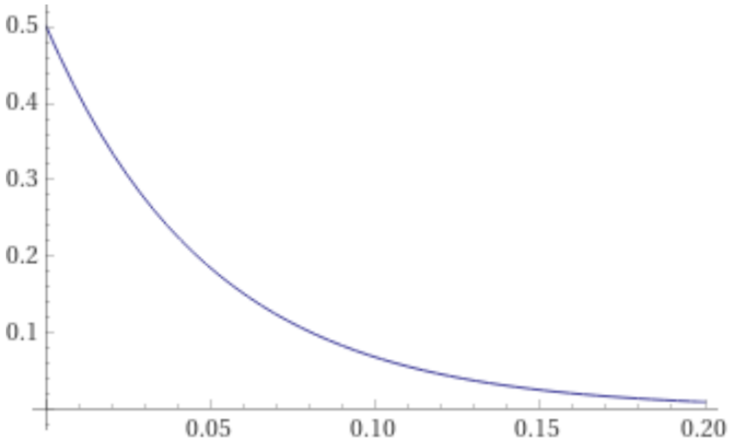


Figure 28: Curve for the reward value for the velocity component, using $\mathrm{s}_{\mathrm{vel}}$= 20.

F.3 EMG reward

The reward function for the EMG component is defined as:

$$EMG\ reward = w_{\text{emg}} * e^{-s_{\text{emg}} * sum(act^2)}. \qquad (8)$$

The value for the linear scaling factor w_{emg} was set to 0.0001. It was found that for higher values for the linear scaling factor, the results for the position data were not accurate. The purpose of the EMG component was to minimize muscle activation, given that the position tracking is still accurate. Therefore, the linear scaling factor is small. The value for the exponential scaling factor s_{emg} was set to 10.

In Figure 29, the reward value curve for the EMG reward can be seen. The reward value is depicted on the vertical axis, and the total activation value on the horizontal axis. It can be seen that the reward value for the EMG component increases as the activation decreases. However, the rewards values are small, even for low activation values. This might cause the agent to prioritize other components over the EMG component. For this component, the shape of the curve could be adjusted, using higher reward values for higher activation and showing a gradient around the goal EMG value instead of zero. This makes the gradient clear to the agent at higher activation values.

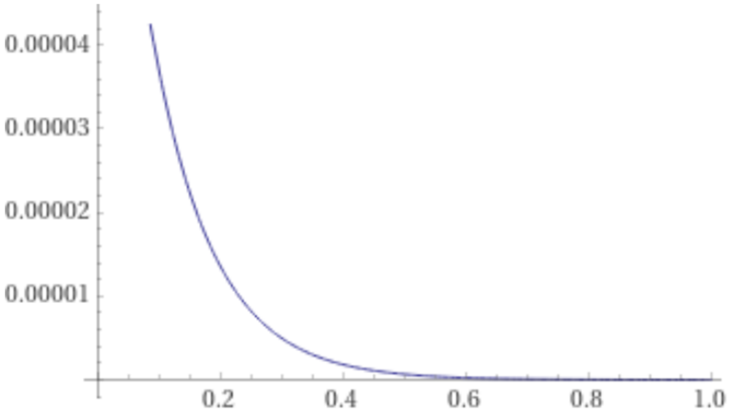


Figure 29: Curve for the reward value for the EMG component, using s_{emg}= 10.

F.4 Acceleration reward

The reward function for the acceleration component is defined as:

$$acceleration\ reward = w_{\text{acc}} * e^{-s_{\text{acc}} * sum(acc^2)}. \tag{9}$$

The function of the acceleration component is to minimize acceleration values to make the simulated muscle activation smoother and therefore more natural.
The linear scaling factor w_{acc} was set to 0.001 to indicate that the importance is significantly less than that of the velocity component. The exponential scaling factor s_{acc} was set to 10. This results in a reward value curve that is similar to that for the EMG component, but with higher reward values, see Figure 30. The reward value is depicted on the vertical axis, and the total acceleration value on the horizontal axis.

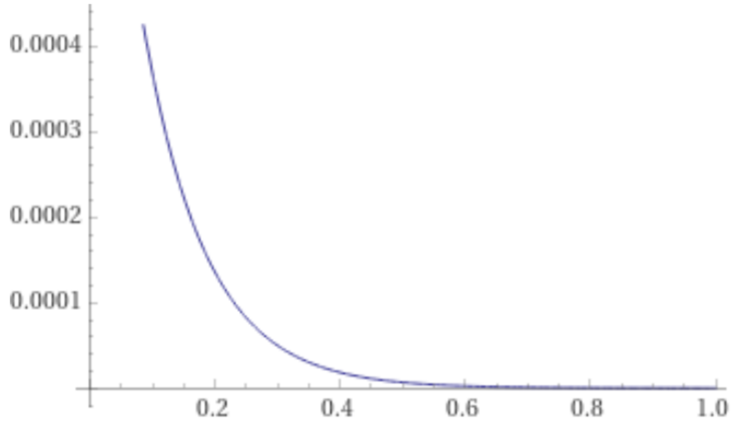


Figure 30: Curve for the reward value for the acceleration component, using s_{acc}= 10.

The maximum acceleration reward is 0.001 and is reached when the acceleration value is 0. However, for most of the simulation, the acceleration values will be higher than 1, which means that the reward value is almost negligible. To solve this, the exponential scaling factor could be decreased to 0.05. This causes the reward value for the acceleration component to increase at higher acceleration values, as can be seen in Figure 31 below. During elbow flexion/extension, accelerating the forearm is necessary. This is because the velocity of the elbow is zero when the elbow is both fully flexed and fully extended. Therefore, increasing the reward value as acceleration values decrease might not be the most accurate method. Minimizing the acceleration error between simulated data and experimental data in a similar way as for the position- and velocity component could lead to more natural simulated data.

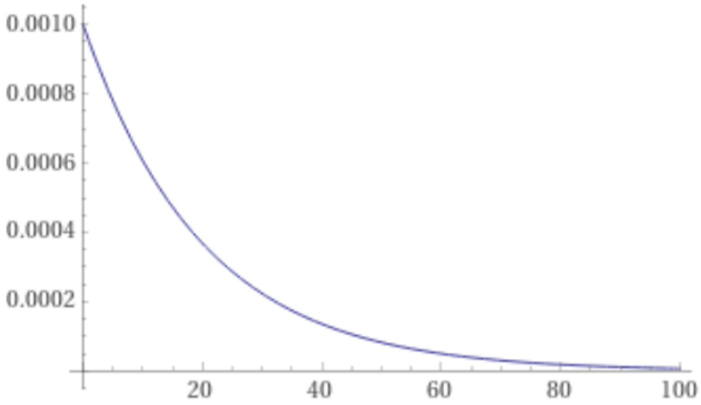


Figure 31: Curve for the reward value for the acceleration component, using $\mathrm{s}_{\mathrm{acc}}$= 0.05.