Deep Reinforcement Learning for Physics-Based Musculoskeletal Simulations of Healthy Subjects and Transfemoral Prostheses' Users During Normal Walking

Leanne de Vree and Raffaella Carloni, *Member, IEEE*

***Abstract*—This paper proposes to use deep reinforcement learning for the simulation of physics-based musculoskeletal models of both healthy subjects and transfemoral prostheses' users during normal level-ground walking. The deep reinforcement learning algorithm is based on the proximal policy optimization approach in combination with imitation learning to guarantee a natural walking gait while reducing the computational time of the training. Firstly, the optimization algorithm is implemented for the OpenSim model of a healthy subject and validated with experimental data from a public data-set. Afterwards, the optimization algorithm is implemented for the OpenSim model of a generic transfemoral prosthesis' user, which has been obtained by reducing the number of muscles around the knee and ankle joints and, specifically, by keeping only the uniarticular ones. The model of the transfemoral prosthesis' user shows a stable gait, with a forward dynamic comparable to the healthy subject's, yet using higher muscles' forces. Even though the computed muscles' forces could not be directly used as control inputs for muscle-like linear actuators due to their pattern, this study paves the way for using deep reinforcement learning for the design of the control architecture of transfemoral prostheses.**

***Index Terms*—Deep Reinforcement Learning (DRL), computer simulation, prosthetics.**

I. Introduction

COMPUTER simulations are used to analyze the biomechanics of both healthy and impaired gait patterns [1], [2], and to understand how assistive devices and prostheses can provide a valuable support to compensate for abnormalities [3], [4].

In this study, we implement computer simulations on two musculoskeletal models, i.e., the model of a healthy human subject and the model of a transfemoral (above-knee) amputee. We use a Deep Reinforcement Learning (DRL) algorithm for both models to optimize the muscle activation so that the models can learn how to walk forward on a flat surface at a normal speed. Specifically, two optimization algorithms are compared, i.e., a Proximal Policy Optimization (PPO) [5] and PPO with imitation learning [6], which we propose in this paper. PPO with imitation learning is validated on the healthy subject model with experimental data from a public data-set and, then, applied to the transfemoral amputee model. After training, the resulting gait patterns of the two models are compared to study the effects of a generic transfemoral prosthesis on the gait patterns and the muscles' forces, and to analyze the required actuators' forces of the prosthesis.

This study consists of three main steps. Firstly, this paper brings DRL to the field of healthy gait analysis via computer simulations. A state-of-the-art DRL algorithm (PPO) is implemented and a DRL algorithm (PPO with imitation learning) is proposed and implemented on a musculoskeletal model of a healthy subject in the open-source simulation software OpenSim [7]. This paper uses the model presented in [8], which consists of two healthy legs including 18 healthy muscles (9 per leg) to control 10 degrees of freedom. The DRL algorithms (PPO and PPO with imitation learning) are validated on a public data-set [9].

Secondly, this study introduces a generic musculoskeletal model of a transfemoral amputee. This model is built in OpenSim, and consists of 19 muscles to control 12 degrees of freedom. The muscles are: 11 healthy muscles in the sound leg, 4 muscles at the hip joint of the amputated leg, and 4 muscle-like actuators in the transfemoral prosthesis, i.e., a generic prosthetic device with two agonist/antagonist muscle-like actuators at the knee joint and two at the ankle joint, which are lost following the transfemoral amputation. Previous research has shown that it is possible to build realistic human models with several deficits [2], transtibial (below-knee) amputation [8], [10], and transfemoral amputation [11] in OpenSim. However, there is not yet a transfemoral amputees model with muscle-like actuators on a prosthetic leg,

Manuscript received August 17, 2020; revised December 12, 2020 and January 27, 2021; accepted February 25, 2021. Date of publication March 1, 2021; date of current version March 9, 2021. This work was supported by the European Commission's Horizon 2020 Program as a part of the Project MyLeg under Grant 780871. *(Corresponding author: Raffaella Carloni.)*

The authors are with the Faculty of Science and Engineering, Bernoulli Institute for Mathematics, Computer Science and Artificial Intelligence, University of Groningen, 9747 AG Groningen, The Netherlands (e-mail: l.de.vree@student.rug.nl; r.carloni@rug.nl).

Digital Object Identifier 10.1109/TNSRE.2021.3063015

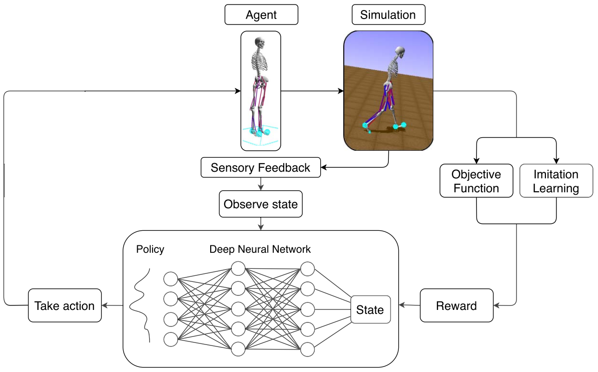


Fig. 1. The proposed DRL algorithm for the dynamic optimization of the forward dynamic of the agent (either the musculoskeletal model of a healthy subject or of a transfemoral amputee) during normal level-ground walking.

specifically designed for implementing DRL, as proposed in this paper.

Thirdly, this study applies the DRL algorithms (PPO and PPO with imitation learning) to the transfemoral amputee model. This paper exploits DRL to control the muscles of the transfemoral amputees and the actuators' forces of the transfemoral prosthesis to achieve normal level-ground walking with a comparable gait pattern as the one of healthy subjects.

To summarize, the contributions of this paper are:

- To bring a state-of-the-art DRL algorithm (PPO) to the analysis of gait patterns of healthy subjects.
- To propose to use a DRL algorithm (PPO with imitation learning) on a healthy subject model in OpenSim.
- To validate the DRL algorithms (PPO and PPO with imitation learning) on a public data-set of healthy subjects during normal level-ground walking.
- To introduce a generic musculoskeletal transfemoral amputee model in OpenSim.
- To use the DRL algorithms (PPO and PPO with imitation learning) to control the healthy muscles of the transfemoral amputees and the actuators' forces of the transfemoral prosthesis to achieve gait patterns with a forward dynamics comparable to healthy subjects'.
- To analyze and evaluate the forces of the healthy muscles and of the muscle-like actuators of the transfemoral amputee model, and to compare them to the muscles' forces of the healthy model and of healthy subjects.

This study, with reference to Figure 1, proposes to use a novel DRL algorithm (PPO with imitation learning) to simulate the forward dynamics of an agent (either the musculoskeletal model of a healthy subject or of a transfemoral amputee) during level-ground walking at a normal speed. The agent is trained by a DRL algorithm, which is based on a deep neural network that receives a reward (computed on an objective function and an imitation learning term) and the observed muscles' and joints' states of the agent as inputs, and outputs an action, i.e., the activation of the muscles and of the muscle-like actuators in the transfemoral prosthesis, that generates the forward dynamic simulation. Sensory feedback, based on the muscles' and joints' states of the agent, is used in the network.

The remainder of the paper is organized as follows. Section II describes the theoretical background on optimization strategies and DRL. Section III presents the methodology of this study, i.e., the deep neural network and the optimizer for training the two models to walk in a normal gait pattern. In Section IV, the implementation in OpenSim is described. The empirical results are presented and discussed in Section V. Finally, concluding remarks are drawn in Section VI.

II. Theoretical Background

This Section discusses the current state of the art of optimization strategies in bipedal locomotion and the motivation behind the choice of using DRL in this study.

A. OpenSim

OpenSim is an open-source software for modeling, simulating, controlling, and analyzing the human neuromusculoskeletal system. By performing inverse and forward dynamics, OpenSim allows for simulations of human locomotion and is used in a wide range of studies [7].

We have several justifications for using OpenSim in this paper. First of all, OpenSim allows to run musculoskeletal simulations, which are an accurate tool to study gait patterns in the absence of human participants [12]. Moreover, OpenSim provides a flexible platform that can be used in combination with optimization routines based on reinforcement learning. Furthermore, OpenSim is widely used in current research on locomotion. Krogt *et al.* [13] use OpenSim to generate muscle-driven simulations of normal walking and then, progressively, weaken the muscle groups to examine how much weakness could be tolerated in order to maintain a normal gait pattern. Steele *et al.* [14] compare the contributions of mass center accelerations and joint angular accelerations during single-limb stance in crouch gait to those in unimpaired gaits. LaPre *et al.* [15] analyze the interaction between residual limb and prostheses' sockets in transtibial amputees using OpenSim as a simulation tool. OpenSim is also used in transfemoral prosthetics research, for instance to examine the influence of limb alignment and surgical technique on a muscle's capacity to generate a force [3], to study gait compensatory mechanisms [4], and to understand the biomechanics of osseointegrated transfemoral amputees [11].

B. Optimization Strategies

Optimization strategies are algorithms that aim at optimizing a policy in order to perform a task. Algorithms have been designed to find the global minimum of an objective function by efficiently using computational power. Hereafter, we discuss three different evolutionary algorithms (i.e., covariance matrix adaptation, biogeography-based optimization, and particle swarm optimization) and DRL algorithms that have been used for dynamic optimization, as summarized in Table I. The choice for using evolutionary algorithms methods is often based upon their resemblance with natural selection, thus assumed to produce more natural behaviors [16]. As in evolution, the chances of reproduction for individuals in a population are determined by the fitness of its genes. In these

TABLE I
STATE-OF-THE-ART OF EVOLUTIONARY ALGORITHMS AND DRL ALGORITHMS FOR DYNAMIC OPTIMIZATION

Article	Inputs	Parameters	Outputs	Objective function	Kind of validation	Application
Covariance Matrix Adaptation (CMA-ES, Hansen (2006) [17])						
[1] (2013)	Activation state, total muscle length	Muscle routing, physiological properties, attachment points	Muscle excitation	Minimize error: speed, head orientation / velocity, slide, effort	Finding a stable controller within 300 generations	Model 3D bipedal characters
[18] (2018)	Step signal, torque approximation	Peak torque, peak torque time, rise times, fall times	Muscle excitation	Proportional / integral / differential gain	Lower overshoot compared to original PID	Powered ankle-foot prosthesis
[2] (2019)	Initial state, simulation performance	Gains, offsets, and transition	Muscle excitation	Minimize cost, avoid falling, head stability	RMSE and normalized cross-correlation experimental data	Healthy human and ankle weakness model
Biogeography-Based Optimization (BBO, Simon (2008) [19])						
[20] (2014)	Clinical human gait data	Ground reaction forces	Motor torques	Minimize error: difference GRF's robot vs. reference	RMSE GRF's robot data vs. reference data	Prosthetic leg testing robot
[22] (2017)	Reference velocity	Motor controller: center of mass, shank length	Angular velocity	Minimize error: system identification vs. control	Integral square error system identification vs. control	Active prosthetic knee
Particle Swarm Optimization (PSO)						
[25] (2019)	Muscle, joint over-extension prevention, ground sensory data, stability feedback	Stance and swing time	Sensor and joint outputs	Distance travelled, penalty for falling / velocity	Ankle angles and torques model vs. experimental data	Simulated transtibial amputee model
[26] (2015)	Generalized joint displacements	Joint displacements, velocities, accelerations	Optimal design parameters	Control signal magnitudes and tracking errors	RMSE control signal magnitudes and tracking errors	Active transfemoral prosthetic leg
Deep Reinforcement Learning (DRL)						
[28] (2016)	Raw image pixels from simulator	Linear and angular velocities of ball	Eight possible actions	Positive rotation of ball along the z-axis	Increase in average reward over time	Robotic hand
[29] (2017)	Electromyographic signals from the user	The system's weight vectors	Continuous actions	Minimize error in joint angles	Ability to achieve desired joint angles	Powered prosthetic arm
[30] (2018)	Position and velocity of the joints and object position	The neural network's weight vectors	Continuous actions	Minimize distance between the hand and the object	Increase in average return over time	Anthropomorphic hand
[8] (2019)	Joint angles and positions	The neural network's weight vectors	Continuous actions	Amount of meters traveled, penalty for overusing ligaments	Increase in average reward over time	Transtibial amputee simulations

applications, the population would be represented by candidate solutions and its genes are the gait parameters, such as the activation for each muscle. After each time-step, individuals reproduce based on their fitness. Candidate solutions with a higher fitness have a higher chance of reproduction, i.e., finding the optimal solution for the task.

1) Covariance Matrix Adaptation: Covariance Matrix Adaptation Evolution Strategy (CMA-ES) is an evolutionary algorithm often used as optimizer in robotic applications [17]. By continuously updating a matrix, the algorithm searches for optimal solutions to optimize an objective function. An optimization strategy based on CMA-ES for the simulated motion of 3D bipedal characters has been developed in [1]. Physics-based characters have been created after which the geometry is optimized to allow for natural walking. Parameters, such as muscle routing, physiological properties, and attachment points, are optimized following an objective function to minimize errors based on speed, head orientation, head velocity, and effort. Yin *et al.* [18] use CMA-ES for the torque control of a powered ankle-foot prosthesis. Optimization of the control parameters demonstrates that the overshoot of the CMA-ES controller is lower than that in the original control parameters. Similarly, Ong *et al.* [2] use CMA-ES as an optimizer to create gait patterns. Two simulation models in OpenSim are trained, i.e., one healthy subject and a subject with specific ankle weaknesses, and their gait patterns are compared.

2) Biogeography-Based Optimization: Biogeography-Based Optimization (BBO) [19] is an evolutionary algorithm similar to CMA-ES. Instead of updating a matrix, BBO uses the migration behavior of candidate solutions that are divided into a population where the sharing process of decision variables is analogous. Across generations, each candidate solution immigrates decision variables from and emigrates them to others. Davis *et al.* [20] use BBO to train a robot to walk with a prosthesis. Their results show that BBO could achieve a 62% decrease in the ground reaction force error with regard to gait data. Thomas *et al.* [21] train artificial neural networks with BBO for the control of a prosthetic knee. It is demonstrated that BBO improves the average performance of the powered knee prosthesis by 8%. Abdelhady *et al.* [22] use BBO to optimize the parameters of an active prosthetic knee and demonstrate good performances in the prediction of these parameters. Ammu *et al.* [23] list several disadvantages of using BBO as an optimizer. First, although BBO is designed to converge to a solution, it often ends up in local minima. This could be solved by increasing the rate of mutation next to migration to increase randomness. However, high rates may complicate the algorithm and it may not be able to find a solution. Second, there is no provision that requires to always select the best members. Whether an individual reproduces in BBO is dependent upon the rates of migration. To prevent local minima, these probabilities are rarely set to 100%. Therefore, useful information in the best members can be lost while using BBO. Third, taking over features from the previous generation is independent of the individual's resultant fitness, hence infeasible solutions are generated. This unnecessarily increases the required computational resources, making the algorithm less efficient.

3) Particle Swarm Optimization: Particle Swarm Optimization (PSO) is an evolutionary algorithm that focuses more on swarm behavior rather than on information sharing, and is one of the most commonly used optimization techniques [24]. PSO is defined by self-organization, i.e., it is not controlled by any factor inside or outside the system. It performs searching on a swarm of particles that updates each time-step. To find the global minimum, each particle moves towards its previously best solution and the global best solution in the swarm. Ferreira *et al.* [25] propose a simulated transtibial amputee model with both a passive and an active prosthesis optimized by means of PSO. The fitness was calculated in terms of distance traveled, minus a penalty for falling and not adhering to a desired velocity. It is found that, with an asymmetric gait pattern, amputees consume around 20-30% more metabolic energy compared to healthy individuals. Azimi *et al.* [26] introduce a robust model as an adaptive controller for an active transfemoral prosthetic leg. In their framework, PSO is used to optimize the design parameters of the controller where the cost function consists of control signal magnitudes and tracking errors. They showed that PSO obtains an 8% improvement in the objective function. Kameyama [27] reports several issues with regard to the stability and convergence of PSO, e.g., the trajectory of each particle might become an oscillation, hence leading to instability of the algorithm. In addition, the algorithm rarely leads to convergence after a local minimum.

4) Deep Reinforcement Learning: Although there exists a rich literature on studying transfemoral prostheses using various optimization strategies, research that has examined the effects of transfemoral prostheses on walking patterns using DRL as an optimization strategy is relatively sparse.

DRL consists of two parts, i.e., reinforcement learning and deep learning. Reinforcement learning is a branch of machine learning in which the learner is a decision-making agent that takes actions in an environment receiving feedback for its actions when judged in terms of solving a certain problem [31]. Hence, the agent can take actions in an environment and, based on the reward that action provides, the agent decides whether this action should be taken again when being in the same state.

Most of the current contributions to prosthetics research that use DRL are applications to arm prostheses. Katyal *et al.* [28] use a shallow neural network in combination with reinforcement learning to learn a policy for in-hand manipulation from raw images. This method requires a discrete action space, which would not work for walking, where a method that can solve for a continuous action space (i.e., non-binary values for muscles activation) is required. This paper extends this approach by allowing the DRL algorithm to learn in a continuous action space. Vasan and Pilarski [29] use reinforcement learning to teach a powered prosthetic arm to perform tasks as an intact arm. This method, called learning-from-demonstration, requires experimental data for muscle activations. Since in this paper, we do not intend to use muscle activations data for a transfemoral prosthetic leg, we propose to exploit imitation learning which uses data on the model's state, e.g., the joint angles, rather than muscle activations. This allows the algorithm to find the optimal muscle activations to generate a walking pattern without information of what those activations should be. Mudigonda *et al.* [30] demonstrate that it is possible to learn robust grasp policies for anthropomorphic hands by means of DRL. This method uses the trust-region-policy optimization, i.e., an algorithm similar to the one used in this paper. PPO, as used in this paper, optimizes the trust-region-policy optimization by modifying the surrogate objective function such that it prevents large policy updates. This adaptation improves the algorithm's performance and decreases the complexity of implementation and computation [5].

C. Transfemoral Prostheses and DRL

Previous research has demonstrated that DRL may provide an effective tool in studying arm prostheses. However, so far, DRL has not yet been applied to transfemoral prostheses. This paper argues that DRL can be effectively applied to transfemoral prostheses for the following reasons.

Firstly, DRL has proven to perform well for studying transtibial amputees so it provides a promising method for transfemoral amputees. In the NeurIPS 2018 Artificial Intelligence for Prosthetics challenge, participants were asked to build a controller for a transtibial amputee model with the goal of moving it forward, and were encouraged to use DRL. Kidzinski *et al.* [8] use a musculoskeletal model of 19 muscles with one leg having the below-knee leg replaced by a prosthesis. Results have shown that DRL can find solutions in which the agent learns a policy to efficiently move forward.

Secondly, a musculoskeletal model of a transfemoral amputee performs its motions in a continuous action space and DRL is specialized to deal with continuous action spaces. Muscles cannot only activate or deactivate, they can also partly activate by imposing less or more power to the muscle. Combined with the degrees of freedom, this creates a large action space which makes reinforcement learning unsuitable. Therefore, DRL can provide solutions allowing an agent to learn how to walk with different transfemoral prostheses.

Thirdly, using DRL limits the need of experimental data. In transfemoral prostheses research, collecting experimental data can be complicated due to the costs of finding and testing suitable participants. The advantage of using computer simulations in combination with DRL techniques is that it does not require the agent to have knowledge about the environment. The agent purely acts on the rewards and the penalties it receives from trying actions, which are designed

TABLE II
THE STATE VARIABLES OF THE AGENT (EITHER THE MUSCULOSKELETAL MODEL OF THE HEALTHY SUBJECT OR OF THE TRANSFEMORAL AMPUTEE)

	Healthy	**Amputee**
Position+Rotations of body segments	13+13	13+13
Linear+Rotational vel. of body segments	13+13	13+13
Linear+Rotational acc. of body segments	13+13	13+13
Positions+Velocities+Accelerations of joints	17+17+17	17+17+17
Muscle forces	72	76
Miscellaneous forces	13	13
Total size of the state vector	**214**	**218**

based on an objective, such as moving forward. Therefore, little experimental data is needed to find an efficient solution.

Fourthly, DRL allows for making and studying adaptations of a prosthetic device quickly. Due to material, time, and participant constraints, experimental methods for transfemoral prosthetics complicate the adaptation of a prosthetic device and testing it directly. With computer simulations, changes to the device can be made quickly and specifically. This allows for testing several variations of, e.g., joints' motors, allowing for finding the optimal combination of the device's specifications that lead to highest performance. DRL is flexible, meaning that the same algorithm can be applied to an agent with a different prosthetic device and still converge to a solution.

III. METHOD

This paper proposes to use a DRL algorithm (PPO with imitation learning) so that the agent (either the musculoskeletal model of a healthy subject or of a transfemoral amputee) learns how to walk forward on a flat surface at a normal speed, as shown in Figure 1. The remainder of this Section details the components of the DRL algorithm.

A. Deep Neural Network

DRL algorithms are based on deep neural networks. In this study, we propose to use a multi-layer perceptron, a feed-forward artificial neural network that consists of 4 layers, i.e., an input layer with 214 neurons (for the healthy subject model) or 218 (for the transfemoral amputee model), two hidden layers with 312 neurons each, and an output layer with 18 neurons (for the healthy subject model) or 19 neurons (for the transfemoral amputee model). For each neuron υ_i, the output y is calculated using a general output function, i.e., $y(\upsilon_i) = \tanh\left(b + \sum_{i=1}^{n} x_i \omega_i\right)$, where n is the number of inputs from the previous layer, x the input to the neuron, ω the weight between the current and the previous neuron, b the bias, and tanh the activation function.

The input to the deep neural network (see Figure 1) is the state of the agent (i.e., a vector of positions/rotations, linear/rotational velocities, and linear/rotational accelerations of the joints' angles and of the body segments either of the musculoskeletal model of the healthy subject or of the transfemoral amputee), ground reaction forces, muscle activities, muscle fiber lengths, muscle velocities, and tendon forces. Table II summarizes the state variables of the agent.

The output of the deep neural network (see Figure 1) is an action, i.e., an 18-dimensional for the healthy subject model or a 19-dimensional vector for the transfemoral amputee model, where each variable represents the activation of one of the muscles in the models. Both the state vector and action vector are continuous variables, which entails that the values they contain are neither binary nor binned to a certain distribution. Our goal is to optimize the weights in the neural network such that, for each given state, the deep neural network outputs the optimal action to allow the agent (either the model of the healthy subject or of the transfemoral amputee) to walk.

B. The Learning Algorithms

The neural network is trained to obtain input-output behaviors that satisfy the task of normal walking. During training, the weights of the connections between the neurons are optimized such that the network outputs desirable actions based on the state input. The following subsections discuss the learning algorithms used to train the neural network.

1) Proximal Policy Optimization: The optimization used in this paper is PPO, i.e., an algorithm introduced in [5] as an alternative to policy gradient methods. By modifying the surrogate objective function of the trust-region-policy optimization method, PPO becomes more effective yet simple to implement. In PPO, the agent alternates between sampling trajectories with the newest policy, and performs optimization on the objective function using the earlier sampled trajectories. It aims to not allow for large updates of the policy, i.e., the mapping from states to actions. This is done by keeping the Kullback-Leiber divergence, which measures how one probability distribution is different from another, between the new and the old policy within the trust region. To achieve this, PPO clips the probability ratio and adds the Kullback-Leiber divergence term to the loss. The ratio $r_t(\theta)$ denotes the probability ratio between the probabilities of the new and the old policy, i.e.: $r_t(\theta) = \frac{\pi_\theta(\alpha_t|s_t)}{\pi_{\theta_{old}}(\alpha_t|s_t)}$, where θ and θ_{old} are the new and the old parameters, π is the policy, π_θ is the policy corresponding to the parameter θ, α_t and s_t are the action vector and the state vector at the time-step t, respectively. PPO uses the following objective function: $L^{CLIP}(\theta) = \mathbb{E}\left[min(r_t(\theta)\hat{A}_t, clip(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)\right]$, where $\mathbb{E}$ is the expected value, $\hat{A}_t$ is the advantage estimation, i.e., the difference between the expected and the real reward from an action, and ϵ is the clip value. If the probability ratio falls outside the range $[(1-\epsilon), \cdots, (1+\epsilon)]$, the advantage function is clipped to prevent too large policy updates. Intuitively, a high advantage means that the difference between the expected and the estimated reward is positively turned to the estimated reward. Hence, the new action in this state gives more reward than expected by the current policy. Since a higher reward is better, we want to ensure that the probability of taking the new action in this state increases to increase future reward. Therefore, we update the weights in the neural network such that there is a higher probability of taking this action in this state. The clip surrogate makes sure that the update is not too large, as disruptions in the current policy may have a negative influence on previous updates made. On the contrary, for a low advantage the new action in this state gives

TABLE III
THE HYPERPARAMETERS AND SPECIFICATIONS USED IN PPO LEARNING ALGORITHM DURING TRAINING

Parameter	Value	Parameter	Value
Parallel runs	4	Parameter noise	yes
Iterations	895	PPO clip parameter (ϵ)	0.2
Episodes	30.000	PPO batch size	512
Steps	5.000.000	PPO optimiz. per epoch	4
Training time	12 [hours]	PPO entropy coefficient	0.01
Workers	4	PPO δ	0.9
Steps per worker	1,536	PPO γ	0.999
Steps per action	4	Neural network amount	2
Entropy coeff.	0.01	Neural network size	312
Policy network(s)	1	Activation function	tanh

less reward than expected by the current policy. Therefore, we want to decrease the probability of taking this action in the current state. Again, we update the weights in the neural network such that there is a lower probability of taking this action in this state, while the clip surrogate ensures that the update is not too large. The advantage is calculated using the expected and the real reward from an action. The expected reward is based on the old policy, while the real reward is the output of the reward function, as discussed in the next section.

Table III summarizes the hyperparameters and specifications used in PPO learning algorithm. Note that δ controls the size of the allowed policy updates (i.e., the Kullback-Leiber divergence should be smaller than δ) and γ is the discount factor ($0 \leq \gamma \leq 1$), i.e., a meta-parameter that determines to what extent the agent considers future rewards. If the discount factor γ is set to 0, then the agent only considers the current reward. A discount factor close to 1 is preferred when rewards in the distant future are to be prioritized.

2) Reward Function: The reward function provides the agent (either the model of the healthy subject or of the transfemoral amputee) with information about the value of its actions. In reinforcement learning, the agent does not have any information about the environment. The only information it gets is that, after each action, it receives a reward for that action. This reward can be either positive or negative, and the agents' behaviors are based upon maximizing the reward.

The reward function has a large influence on the agents' behaviors and, in this study, we want the agent to walk similarly to a healthy subject. In this paper, the reward function $reward_{goal,t}$ consists of four parts: the amount of distance travelled, the adherence to a desired velocity, the muscle fatigue, and the time-steps the agent manages not to fall, i.e.:

$$\begin{aligned} & reward_{goal,t} \\ & \quad = \sum_t (reward_{distance} - penalty_{velocity} - penalty_{cost}) \\ & \qquad + \sum_t (reward_{alive}) \end{aligned} \tag{1}$$

where t is the time-step. The first part of the reward function concerns the distance the agent has travelled. This is expressed by adding a goal reward that calculates the difference in the x-position of the agent's pelvis between the current and the previous time-step. The higher this difference, the more distance the agent has travelled during a time-step and the higher is its reward. The second part rewards the agent for adhering to a target velocity. During each of the simulations, the agent gets as input a certain target velocity. This velocity changes two or three times for each simulation. The goal of the agent is to develop a walking pattern in the specific target velocity, such as 4.5 km/h, which is a reasonable speed for human walking. During each time-step, the difference between the velocity of the agent's pelvis and the target velocity is calculated. The sum of squared error of this difference denotes a penalty for deviating from the target velocity. The third element of the reward function gives a penalty for muscle fatigue. The transport costs the amount of energy that the muscles use for moving from point to point. Certain gait patterns, such as one-legged walking, may also allow the agent for reaching its goal. However, this gait pattern puts more pressure on the muscles compared to a more natural, symmetric, two-leg gait pattern. Since we are looking for the most efficient gait for reaching the task, we add a penalty for muscle fatigue. For each time-step, the squared sum of the muscle activations serves as a penalty for the agent. Lastly, the fourth part of the function regards the number of time-steps the agent manages not to fall. The simulation stops every time the pelvis of the agents drops below 0.7 m, and the agent is explicitly rewarded for every additional time-step it walks without falling down.

3) PPO With Imitation Learning: The reward function is designed to lead to realistic motions, but it does not ensure that the motions are human-like. Hence, inspired by the work in [8] on transtibial amputee models in OpenSim, this paper proposes to use PPO with imitation learning.

PPO with imitation learning has proven to decrease training times as well as to increase performance. To implement PPO with imitation learning, we introduce an imitation term next to the original reward function. The added imitation term uses experimental data (from the NeurIPS 2018 for transtibial amputees [8]) to ensure that the algorithm converges to a solution and that the agent develops a natural walking pattern. For each time-step, both the position and the velocity loss of the pelvis, knee, hip, and ankle joints are calculated. This is done by taking the sum of the squared error of the difference between the current angles of the agent's joints and the joints' angles in the data in [8] at a specific time-step. The same is done for the velocities, where their losses are calculated by the difference between the current velocities of the agent's joints and the joint's velocities in the data in [8] at a specific time-step. These losses serve as a penalty to the reward. The higher the losses, the lower the reward and the other way around. This encourages the agent to keep its states as close to the ones in the data as possible, hence encouraging a more natural walking pattern. It should be noted that the data-set used for training [8] required no scaling because it was specifically designed for OpenSim and for the healthy subject model used in this study. However, before using the data for the training of the imitation learning part of both models, they have been processed to guarantee a symmetric gait pattern of the hip, knee, and ankle angles.

The final reward consists of the sum of the goal reward term and the imitation reward term. For the selection of the

TABLE IV

Some Trials for Finding the Optimal Weights of the Goal and Imitation Learning Terms of the Reward Function. The Reward Column Shows the Average After 30.000 Episodes of Training. The Best Results Are in Bold

Goal - Imitation	Reward	Observations
25 - 75	3300	Mostly standing still
30 - 70	2000	Moving forward, falls often
35 - 65	3200	Moving forward, falls often
40 - 60	**2900**	**Moving forward, rarely falling**
45 - 55	2600	Moving forward, falls often

weights of each terms, several simulations were conducted which showed that 40% for the goal reward and 60% for the imitation reward are the optimal values. Table IV reports some trials, but more were conducted around 40-60 as they showed high average result. In combination with Equation 1, the proposed final reward function is:

$$Reward_t = 0.4 \cdot reward_{goal,t} + 0.6 \cdot reward_{imitation,t} \qquad (2)$$

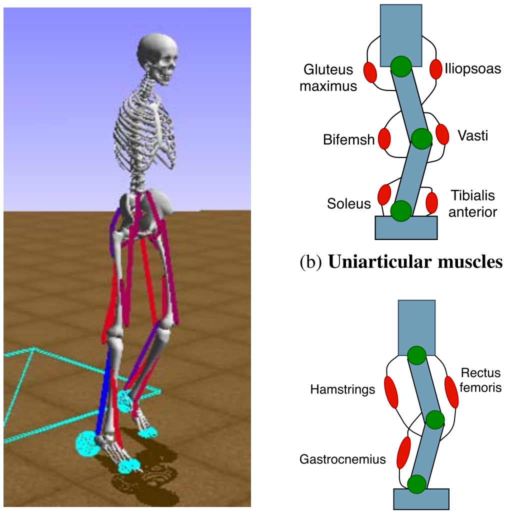
IV. Implementation

In this Section, we implement the DRL algorithm presented in Section III on the two OpenSim musculoskeletal models (healthy subject and transfemoral amputee). Specifically, the healthy subject model is used to implement the DRL algorithm (PPO and PPO with imitation learning) and to validate the emerging gait patterns with experimental data from a public data-set [9]. The transfemoral amputee model uses the validated DRL algorithm (PPO and PPO with imitation learning) to analyze the effects of a transfemoral prosthetic leg on gait patterns. Figure 2 visualizes the models and Table VIII (see Appendix) details the included muscles.

A. Two Models

In OpenSim, a musculoskeletal model consists of rigid body segments connected by joints. Hill-type muscles connect these joints and produce forces and motion [32]. Therefore, once a musculoskeletal model is created, OpenSim allows users to investigate a wide range of topics, such as the effects of geometry, the joint kinematics, and the properties on the muscle-tendons' forces and joints' moments. Figure 2 (a) shows an example of an animation in OpenSim, where the red lines represent activated muscles, while the blue lines are deactivated muscles. For the implementation of the DRL algorithm, this paper makes use of the environment in [33], which provides a link between the OpenSim software and the Python programming language (www.python.org).

1) Healthy Subject Model: The healthy model is the musculoskeletal model of a human. For this study, we use the model provided in the NeurIPS 2018 with two healthy legs [33]. The model has 18 muscles to control 10 degrees of freedom. Specifically, each leg has 6 uniarticular muscles and 3 biarticular muscles, as shown in Figure 2 (b) and (c). The 10 degrees of freedom are: 3 between the pelvis and the ground (2 transl.and 1 rot.), 1 at each hip joint, 1 at each knee joint, 1 at each ankle joint, and 1 to move the pelvis freely.



(a) Healthy subject model

(b) **Uniarticular muscles**

(c) **Biarticular muscles**

Fig. 2. Overview of the muscles used in the two models. (a) Shows the healthy subject model with 18 Hill-type muscles [32], shown in red and blue. (b) Displays in red the six uniarticular muscles in each leg that produce flexion or extension torques at single joints. (c) Shows the three biarticular muscles in red that generate torques at pairs of joints. The healthy subject model contains both the uniarticular muscles from (b) and the biarticular muscles from (c) in each leg. The transfemoral amputee model contains both the uniarticular and the biarticular muscles in the left leg and only the uniarticular muscles from (b) in the right leg (i.e., the transfemoral prosthesis).

2) Transfemoral Amputee Model: The simulated transfemoral amputee model proposed in this study consists of 19 muscles to control 12 degrees of freedom. Specifically, the left healthy leg has 11 muscles, i.e., 6 uniarticular muscles and 3 biarticular muscles (as shown in Figure 2 (b) and (c)), and 2 uniarticular muscles for the hip adduction and abduction. The right transfemoral prosthetic leg has 8 uniarticular muscles, i.e., two agonist/antagonist muscles at the hip joint, two agonist/antagonist muscles at the knee joint, and two agonist/antagonist muscles at the ankle joint (as shown in Figure 2 (b)), and 2 muscles for the hip adduction and abduction. The 12 degrees of freedom include the 10 of the healthy subject model, plus one additional degree of freedom for each leg allowing for the hip adduction/abduction. Hip adductors/abductors were added to both legs to make the model more realistic and allow it for maintaining stability.

This transfemoral amputee model has been realized in OpenSim by replacing the OpenSim model input file in [33] with an adjusted model such that the right leg does no longer include three biarticular muscles, i.e., gastrocnemius, hamstrings, and rectus femoris. The short head of the biceps femoris, vasti, soleus, and tibialis anterior muscles were kept so they can simulate muscle-like linear actuators, i.e., sources to power the prosthetic leg. The hip adductors and abductors were taken from the transtibial model in [8] and added to the OpenSim model file.

3) Muscle Actuation: The two models as presented above contain simulated biological muscles based on a first-order dynamic Hill-type muscle model between excitation and activation [34]. Figure 3 shows the Hill-type muscle model including a contractile element (CE), a parallel elastic element (PE) and a series elastic element (SE). The generated muscle force

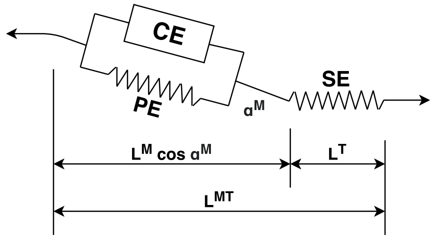


Fig. 3. Hill-type muscle model that describes the musculo-tendon contraction mechanics in the two models [32]. It includes a contractile element (CE), a parallel elastic element (PE), and a series elastic element (SE). The elements generate a force on the tendon [34].

is a function of three factors: the length, the velocity, and the activation level, which can range between 0% and 100%. The muscle activations generate a movement as a function of muscle properties, such as, the maximum isometric force, the muscle fiber length L^M, the tendon slack length L^T, the maximum contraction velocity, and the pennation angle α^M.

Based on the observation of the state vector, the DRL algorithm outputs a vector of muscle excitations. OpenSim calculates the muscle activations from the excitations by using first-order dynamics equations of a Hill-type muscle model. To summarize, during each time-step of 10 ms, the simulation: (i) computes the activations of the muscles based on the provided excitation vector; (ii) actuates the muscles; (iii) computes the torques based on the activations; (iv) computes the ground reaction forces; (v) computes the positions and the velocities of the joints and the bodies' segments; (vi) generates a new state based on the forces, velocities, and positions of the joints.

B. Validation Data-Set

For validating the proposed DRL algorithms and, specifically, for implementing the imitation learning reward term, we use the experimental data taken from a public data-set [9]. The data was collected on 83 typically developing children by measuring the kinematics and kinetics of the hip, knee, and ankle joints, the surface electromyographic signals, and the spatio-temporal data. It contains the means of all subjects over one gait cycle at speeds ranging from very slow to very fast (>3 of the standard deviation below or above the mean of the free speed). This paper uses the pelvis, hip, knee, and ankle joints' angles, velocities, and the ground reaction forces.

V. Results and Discussion

This Section presents the results of the DRL algorithm (PPO and PPO with imitation learning). The first part shows the results on the healthy model and demonstrates that it can be validated against experimental data by comparing the kinematic results of the simulation to the experimental data [9]. The second part shows the results of the transfemoral amputee model. The third part discusses the performances of the two models and compares them against experimental data [9].

A. Healthy Subject Model

1) Algorithms' Performance: Figure 4 (left) shows the performance of the DRL algorithm on the healthy model.

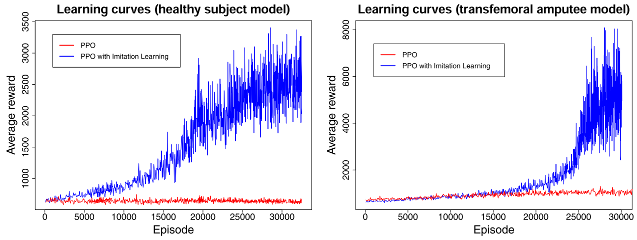


Fig. 4. The learning curves of the DRL algorithm (PPO and PPO with imitation learning) applied to the healthy subject model (left) and to the transfemoral amputee model (right). The y-axis shows the reward and the x-axis the amount of episodes. It can be seen that, in both cases, PPO with imitation learning performs better as over time it learns to maximize rewards.

TABLE V
The Mean Total Reward Received and the Standard Deviation for the Combinations of DRL Algorithm and Models

	PPO		PPO with imitation learning	
	mean	SD	mean	SD
Healthy	641.74	23.82	1784.69	717.68
Amputee	940.07	123.92	2724.36	1919.42

TABLE VI
Similarity Metrics Between Experimental and Simulated Data of the Emerging Gait Pattern of the Healthy Model. The RMSE Is Reported in Units of the SD. The Z-Score Denotes the Mean Taken Over All Scores for Each 2% of the Gait Cycle

	Angles			GRFs	
	Hip	Knee	Ankle	Hor	Vert
RMSE	1.24	1.44	2.20	0.15	0.55
Z-score	-0.45	-1.05	1.77	1.15	-0.26

The red curve shows the average reward for PPO over a total of 30.000 episodes (one simulation), and the blue curve shows the same results for PPO with imitation learning. It can be noted that PPO with imitation learning allows for a better learning compared to PPO, with a reward mean of 1784.69 compared to 641.74 (see Table V). These numbers indicate that the healthy subject model learns to optimize the objective function around four times better when PPO is combined with imitation learning. After around 15.000 episodes, there is a steep rise in rewards, which means that the agent has learned a policy, i.e., a division of weights in the neural network, that allows it to maximize returns based on the reward function.

2) Kinematics: Figure 5 (black line) shows the kinematics for the emerging gait pattern of the healthy model, which has been learned using PPO with imitation learning. It compares the experimental data (gray area), including the angles of the hip, knee and ankle joints, and the horizontal and vertical ground reaction forces of the healthy model. The figure shows that the simulated kinematic and kinetic trajectories of the emerging gait pattern are similar both in value and shape to the experimental data.

Table VI summarizes the kinematic results in terms of the root-mean-squared error (RMSE) and the Z-score. The RMSE compares simulation mean trajectories to those of experimental data. It is computed by taking the square root

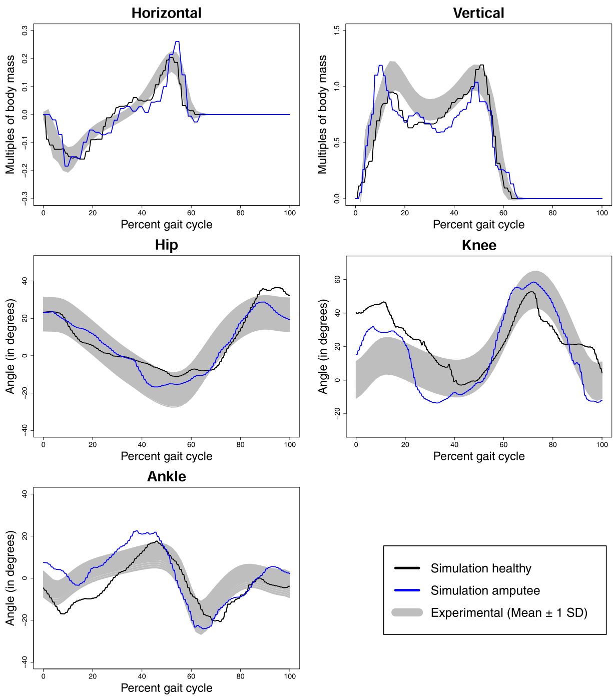


Fig. 5. The horizontal/vertical ground reaction forces and the hip/knee/ankle joints angles of the emerging gait pattern using PPO with imitation learning of the healthy model (black line) and of the transfemoral amputee model (blue line) compared to the experimental data in [9] (the grey area is the healthy subject experimental data varying 1 standard deviation from the mean).

of the difference in errors squared and reported in units of standard deviation (SD). The Z-score denotes how close the simulation data is to the mean of the experimental data. The closer it is to 0, the closer the simulation is to the experimental data. It is computed for the simulation data at every 2% of the gait cycle. It can be noted that, for the majority of the gait cycle, the kinematics and kinetics of the emerging gait pattern are within 1 SD of the experimental data describing a natural walking pattern. Furthermore, for each of the measures, the RMSE between the simulated and the experimental data is no more than 2.20 SD and the Z-score is at most 1.77 away from the mean. The ground reaction forces were found to match well with experimental data with Z-scores of 1.15 and -0.26.

B. Transfemoral Amputee Model

1) Algorithms' Performance: Figure 4 (right) shows the performance of the DRL algorithm on the transfemoral amputee model. The red curve shows the average reward for PPO over a total of 30.000 episodes (one simulation), and the blue curve shows the same results for PPO with imitation learning. It can be noted that PPO with imitation learning is able to learn to maximize its rewards, with a steep increase in average reward around the 20.000th episode. Moreover, the results show that PPO with imitation learning allows for the transfemoral amputee model to learn to walk forward with a reward mean of 2724.36 compared to 940.07 of PPO (see Table V). These numbers indicate that the transfemoral amputee agent has learned a policy that allows it to maximize returns based on the given reward function.

2) Kinematics: Figure 5 (blue line) shows the kinematic data for the emerging gait pattern of the transfemoral amputee model. It compares the angles of the hip, knee and ankle and the horizontal and vertical ground reaction forces of the transfemoral amputee model with the healthy subject model and the healthy subject experimental data along the gait cycle.

The emerging gait pattern was found to have a similar shape and value for the hip and ankle, while it differs for the knee. The hip angles were found to be not much diverging from before, hence there is not much influence of the prosthesis on the movements of the hips. For the ankle, the resulting angles were found to have a similar shape and value compared to experimental data, yet being less smooth compared to the healthy model. This could be explained by that having less muscles in both the lower and upper leg allows for less control over the ankle hence making it more prone to uncontrolled and larger movements. The angles for the knee were found to be most diverging from both the healthy and the experimental data. The shape along the gait cycle is maintained, yet the angles are more extreme compared to the other data. The ground reaction forces were also found to have a similar shape for the transfemoral amputee model. However, for both forces the general value is higher compared to the healthy subject model. Hence, the ground exerted more force to the left leg of the amputee model compared to the left leg of the healthy model. This finding implies that the amputee model's left leg consumes more strength than the healthy model's left leg, as higher strength can be related to increased ground reaction forces [35].

C. Comparison of the Two Models

One of the goals of this study is to find the difference in the power used by the muscles around the knee and ankle between the healthy subject and the transfemoral amputee. Table VII summarizes the results of our simulations with regard to the used muscle power. It shows the mean in activation and fiber force values for the four muscles around the knee and ankle joints taken over 500 episodes of the trained model (~3 gait cycles). For each model, the difference in the left and right leg were computed and added to return a total difference in means. It demonstrates that the difference in activation for the amputee model (0.542) is 49.8% higher compared to the healthy model (0.272). The difference in fiber force is 17.9% higher for the amputee model (2292.782 vs. 1943.878). The results confirm our hypothesis that for the transfemoral amputee model, both activation and fiber force are higher for the right leg (i.e, the prosthesis) and the difference between legs is higher compared to the healthy model. The same conclusion is reached from the Figures 6 and 7 in the Appendix, which show the fiber force for each of the four muscles over time, and from the Figures 8 and 9 in the Appendix, which show the distribution and the mean fiber forces for each of the four muscles.

D. Limitations and Future Outlook

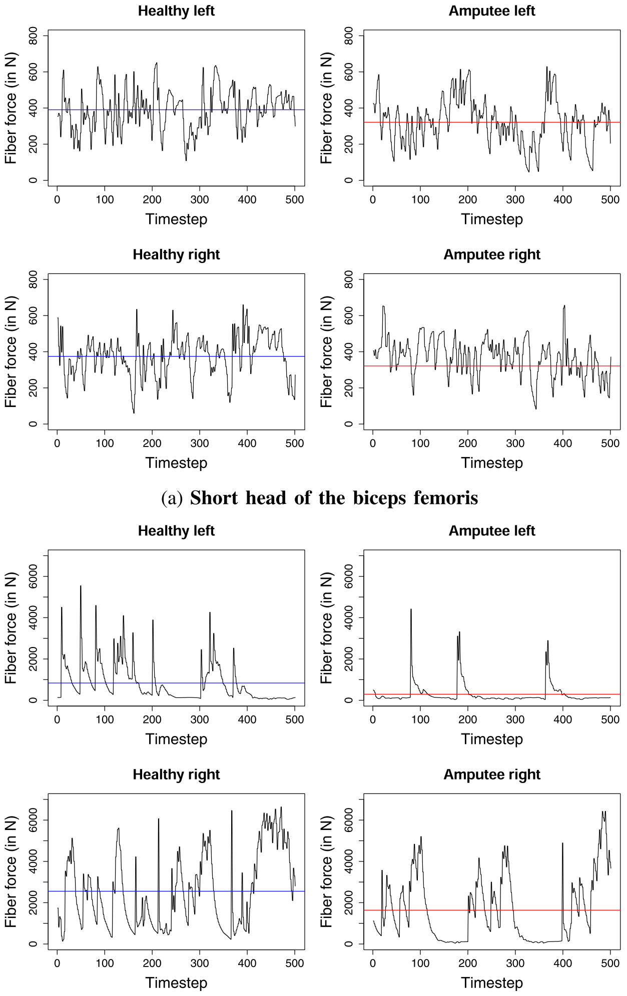
As shown in the previous sections, deep reinforcement learning is able to generate a stable gait with a forward dynamic comparable to the healthy subjects' for physics-based

TABLE VII
RESULTS FOR THE COMPARISON OF THE DIFFERENCE IN MUSCLE USAGE BETWEEN THE HEALTHY AND TRANSFEMORAL AMPUTEE MODELS

		Healthy		
	Muscle	**Left**	**Right**	**Δ**
Activation mean	Bifemsh	0.726	0.705	**0.021**
	Vasti	0.130	0.369	**0.239**
	Soleus	0.205	0.194	**0.011**
	Tibialis anterior	0.780	0.778	**0.001**
Total difference in means		**0.272**		
Fiber force mean	Bifemsh	390.424	374.504	**15.921**
	Vasti	836.544	2554.886	**1718.342**
	Soleus	1001.269	849.358	**151.912**
	Tibialis anterior	1372.850	1430.554	**57.704**
Total difference in means		**1943.878**		
		Transfemoral		
	Muscle	**Left**	**Right**	**Δ**
Activation mean	Bifemsh	0.594	0.699	**0.106**
	Vasti	0.056	0.224	**0.168**
	Soleus	0.364	0.480	**0.116**
	Tibialis anterior	0.740	0.587	**0.153**
Total difference in means		**0.542**		
Fiber force mean	Bifemsh	321.469	369.401	**47.933**
	Vasti	295.214	1639.328	**1344.114**
	Soleus	1440.968	2002.211	**561.243**
	Tibialis anterior	1388.679	1049.187	**339.492**
Total difference in means		**2292.782**		

TABLE VIII
LIST OF MUSCLES IN BOTH THE HEALTHY SUBJECT MODEL (FROM KIDZINSKI *et al.* [33]) AND THE PROPOSED TRANSFEMORAL AMPUTEE MODEL

		Healthy		**Prosthesis**	
Muscle Name	**Primary function**	**Left**	**Right**	**Left**	**Right**
Biceps femoris	Knee flexion	Yes	Yes	Yes	Actuat.
Gastrocnemius	Knee flexion, Ankle extension	Yes	Yes	Yes	No
Gluteus max.	Hip extension	Yes	Yes	Yes	Yes
Biart. hamst.	Hip extension, Knee flexion	Yes	Yes	Yes	No
Iliopsoas	Hip flexion	Yes	Yes	Yes	Yes
Rectus fem.	Hip flexion, Knee extension	Yes	Yes	Yes	No
Soleus	Ankle extension (plantarflexion)	Yes	Yes	Yes	Actuat.
Tibialis ant.	Ankle flexion (dorsiflexion)	Yes	Yes	Yes	Actuat.
Vasti	Knee extension	Yes	Yes	Yes	Actuat.
Hip abductor	Hip abduction	No	No	Yes	Yes
Hip adductor	Hip adduction	No	No	Yes	Yes



(a) **Short head of the biceps femoris**

(b) **Vasti**

Fig. 6. Fiber forces for the short head of the biceps (a) and vasti (b) over 500 time-steps (~3 gait cycles). The blue and red lines denote the mean fiber force over the time period.

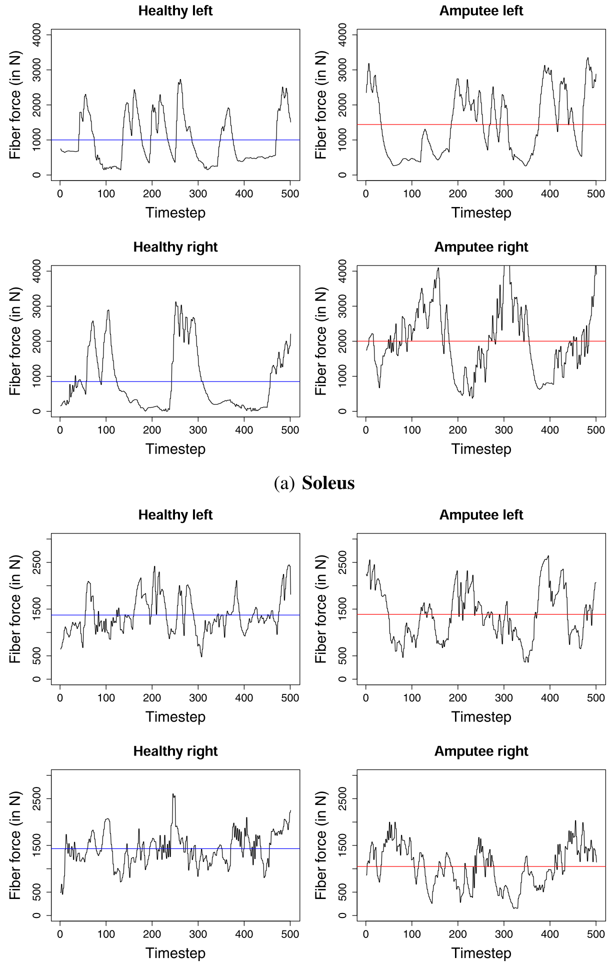
musculoskeletal model of both healthy subjects and transfemoral prostheses' users. However, the computed muscles' forces and, specifically, the fiber forces of the four muscle-like actuators (two agonist/antagonist at the knee joint and two agonist/antagonist at the ankle joint) in the model of the transfemoral amputee have an erratic pattern (see Figures 6 and 7 in the Appendix). As a consequence, these forces could not be directly used as control inputs for the muscle-like linear actuators in the control architecture of a prosthesis.

Future research should focus on investigating how the forces could be less erratic, such that they can be used as control inputs for the muscle-like linear actuators. A possible solution could be found by using the output of the deep neural network (specifically the activation forces of the four muscle-like actuators) to define a reward term that penalizes erratic patterns. This can be achieved by adding the penalty term to the total reward function described by Equation 2. By feeding the muscle activations (i.e., the control inputs to the muscle-like linear actuators) directly back into the reward term, the system will learn that to improve the reward, it needs to output less erratic patterns. This way the training of the deep neural network would account for additional requirements on the action to take onto the agent and, if defined correctly, compute smoother control inputs for the muscle-like linear actuators.

VI. CONCLUSION

By examining the usage of computer simulations to study transfemoral prostheses and gait patterns, this paper contributes to an expanding research field. Having large amounts of literature on healthy walking patterns as well as transtibial prostheses, this paper hypothesized that computer simulations could be used for studying transfemoral prostheses as well.

Testing these predictions, we presented two main contributions of this research to the existing literature. Firstly, we



(a) **Soleus**

(b) **Tibialis anterior**

Fig. 7. Fiber forces for the soleus (a) and tibialis anterior (b) over 500 time-steps (~3 gait cycles). The blue and red lines denote the mean fiber force over the time period.

found that the use of deep reinforcement learning is useful for studying gait patterns for transfemoral prostheses. The presented modification of the proximal policy optimization that includes imitation learning allowed for an optimization of the algorithm and a 2.8 times increased mean reward compared to regular proximal policy optimization. Secondly, the emerging gait pattern was validated against experimental data from a public data-set, with close to natural ground reaction forces as well as joint angles for the hip, knee, and ankle. Our results suggest that a transfemoral prosthesis would need around 49% more activation around the knee and ankle compared to the healthy leg. Moreover, there should be an increase of 17% in force that the replacing muscle-like linear actuators would have to provide to generate a natural gait pattern. The forces computed with deep reinforcement learning and, specifically, with proximal policy optimization combined with imitation learning, show an erratic pattern. As a consequence, it would not be possible to directly use them as control inputs for muscle-like linear actuators in the control architecture of a transfemoral prosthesis. Future work will focus on the generated muscles' patters that can be used as direct control of transfemoral prostheses.

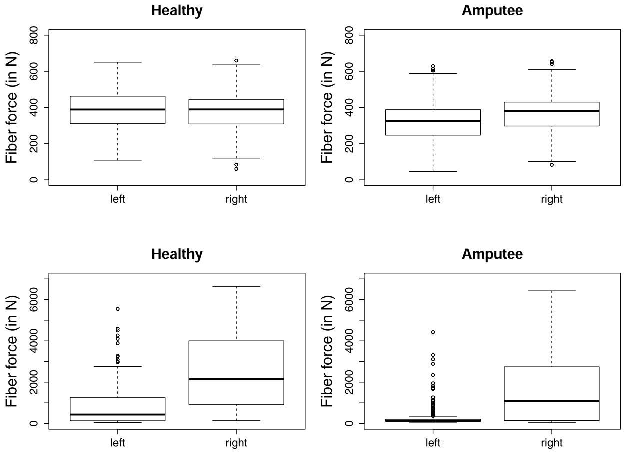


Fig. 8. Comparison of distribution and mean fiber forces for the short head of the biceps femoris (above) and vasti (below) over 500 time-steps (~3 gait cycles).

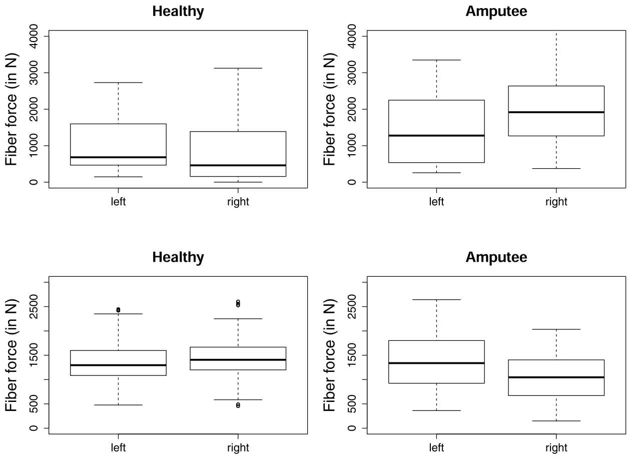


Fig. 9. Comparison of distribution and mean fiber forces for the soleus (above) and of the tibialis anterior (below) over 500 time-steps (~3 gait cycles).

Appendix

Table VIII reports the list of muscles in both the healthy subject model in [33]) and the proposed transfemoral amputee model. It can be noted that the hip adductors/abductors are not present in the healthy subject model and that the biarticular muscles are not present in the right leg of the amputee model (i.e., in the transfemoral prosthesis).

Figure 6 reports the fiber forces of the short head of the biceps femoris (a) and of the vasti (b) over 500 time-steps (~3 gait cycles). The blue and red lines denote the mean fiber force over the time period. For the biceps femoris, the fiber forces of both the healthy subject and the transfemoral amputee are comparable. For the vasti, it can be seen that the difference in mean fiber force between the left and right leg is larger for the transfemoral amputee's model than for the healthy subject's. Figure 7 reports the fiber forces for the soleus (a) and of the tibialis anterior (b) over 500 time-steps (~3 gait cycles). The blue and red lines denote the mean fiber force over the time period. It can be seen that, in general, the muscles in the amputee model's right leg (i.e., the prosthetic leg) have a higher fiber force compared to the healthy model's right leg. Moreover, it can be seen that the difference in mean fiber force between the left and right leg is larger for the amputee model compared to the healthy model.

Figure 8 compares the distribution and mean fiber forces for the short head of the biceps femoris (above) and of the vasti (below) over 500 time-steps (∼3 gait cycles). Figure 9 compares the distribution and mean fiber forces for the soleus (above) and of the tibialis anterior (below) over 500 time-steps (∼3 gait cycles). From both figures, it can be seen that, the difference in mean fiber force between the left- and right leg is higher in the amputee model compared to the healthy model.

Acknowledgment

The authors would like to thank Vishal Raveendranathan (Doctoral candidate, University of Groningen) and Aurelien J.C. Adriaenssens (BSc student, University of Groningen) for the discussions on the OpenSim models, and Rik P. Timmers (Research assistant, University of Groningen) for the advices in the implementation of the proposed method.

References

[1] T. Geijtenbeek, M. van de Panne, and A. F. van der Stappen, "Flexible muscle-based locomotion for bipedal creatures," *ACM Trans. Graph.*, vol. 32, no. 6, pp. 1–11, Nov. 2013.

[2] C. Ong, T. Geijtenbeek, J. Hicks, and S. Delp, "Predicting gait adaptations due to ankle plantarflexor muscle weakness and contracture using physics-based musculoskeletal simulations," *PLoS Comput. Biol.*, vol. 15, no. 10, 2019, Art. no. e1006993.

[3] E. C. Ranz, J. M. Wilken, D. A. Gajewski, and R. R. Neptune, "The influence of limb alignment and transfemoral amputation technique on muscle capacity during gait," *Comput. Methods Biomech. Biomed. Eng.*, vol. 20, no. 11, pp. 1167–1174, Aug. 2017.

[4] V. J. Harandi *et al.*, "Gait compensatory mechanisms in unilateral transfemoral amputees," *Med. Eng. Phys.*, vol. 77, pp. 95–106, Mar. 2020.

[5] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," in *Proc. OpenAI*, 2017, pp. 1–12.

[6] A. Hussein, M. M. Gaber, E. Elyan, and C. Jayne, "Imitation learning: A survey of learning methods," *ACM Comput. Surv.*, vol. 50, no. 2, p. 21, 2017.

[7] S. Delp *et al.*, "OpenSim: Open-source software to create and analyze dynamic simulations of movement," *IEEE Trans. Biomed. Eng.*, vol. 54, no. 11, pp. 1940–1950, Nov. 2007.

[8] Ł. Kidziński *et al.*, "Artificial intelligence for prosthetics: Challenge solutions," in *The NeurIPS'18 Competition: From Machine Learning to Intelligent Conversations*, vol. 69. Cham, Switzerland: Springer, 2019, pp. 1–49.

[9] M. H. Schwartz, A. Rozumalski, and J. P. Trost, "The effect of walking speed on the gait of typically developing children," *J. Biomech.*, vol. 41, no. 8, pp. 1639–1650, 2008.

[10] A. Willson, "A quasi-passive biarticular prosthesis and novel musculoskeletal model for transtibial amputees," Ph.D. dissertation, Dept. Mech. Eng., Univ. Washington, Seattle, WA, USA, 2017.

[11] V. Raveendranathan and R. Carloni, "Musculoskeletal model of an osseointegrated transfemoral amputee in OpenSim," in *Proc. 8th IEEE RAS/EMBS Int. Conf. Biomed. Robot. Biomechatronics (BioRob)*, Nov. 2020, pp. 1196–1201.

[12] A. Galloy, K. Frisch, and D. Schmitz, "Planning an OpenSim simulation," in *ASB: Pipeline Shared by Dordt College UW Madison*. 2018. [Online]. Available: https://simtk-conuence.stanford.edu/display/OpenSim/Planning+an+OpenSim+Simulation

[13] M. M. van der Krogt, S. L. Delp, and M. H. Schwartz, "How robust is human gait to muscle weakness?" *Gait Posture*, vol. 36, no. 1, pp. 113–119, May 2012.

[14] K. M. Steele, A. Seth, J. L. Hicks, M. S. Schwartz, and S. L. Delp, "Muscle contributions to support and progression during single-limb stance in crouch gait," *J. Biomech.*, vol. 43, no. 11, pp. 2099–2105, Aug. 2010.

[15] A. LaPre, M. Price, R. Wedge, B. Umberger, and F. Sup, "Approach for gait analysis in persons with limb loss including residuum and prosthesis socket dynamics," *Int. J. Numer. Methods Biomed. Eng.*, vol. 34, no. 4, 2018, Art. no. e02936.

[16] R. Alexander, "Design by numbers," *Nature*, vol. 412, p. 591, Aug. 2001.

[17] H. Hansen, "The CMA evolution strategy: A comparing review," in *StudFuzz*, vol. 192. Berlin, Germany: Springer, 2006, pp. 75–102.

[18] K. Yin, M. Pang, K. Xiang, and C. Jing, "Optimization parameters of PID controller for powered ankle-foot prosthesis based on CMA evolution strategy," in *Proc. IEEE 7th Data Driven Control Learn. Syst. Conf. (DDCLS)*, May 2018, pp. 175–179.

[19] D. Simon, "Biography-based optimization," *IEEE Trans. Evol. Comput.*, vol. 12, pp. 702–713, 2008.

[20] R. Davis, H. Richter, D. Simon, and A. van den Bogert, "Evolutionary optimization of ground reaction force for a prosthetic leg testing robot," in *Proc. Amer. Control Conf.*, Jun. 2014, pp. 4081–4086.

[21] G. Thomas, S. Szatmary, T. Wilmot, and D. Simon, "Evolutionary optimization of artificial neural networks for prosthetic knee control," in *Efficiency and Scalability Methods for Computational Intellect*, vol. 1. Hershey, PA, USA: IGI Global, 2013, pp. 142–161.

[22] M. Abdelhady, A. Rashvand, H. Richter, and D. Simon, "System identification and control optimization of an active prosthetic knee in swing phase," in *Proc. Amer. Control Conf. (ACC)*, May 2017, pp. 857–862.

[23] P. Ammu, K. Sivakumar, and R. Rejimoan, "Biogeography-based optimization–a survey," *Int. J. Electron. Comput. Sci. Eng.*, vol. 2, no. 1, pp. 154–160, 2013.

[24] E. Elbeltagi, T. Hegazy, and D. Grierson, "Comparison among five evolutionary-based optimization algorithms," *Adv. Eng. Informat.*, vol. 19, no. 1, pp. 43–53, Jan. 2005.

[25] C. Ferreira, F. Dzeladini, A. Ijspeert, L. P. Reis, and C. P. Santos, "Development of a simulated transtibial amputee model," in *Proc. IEEE Int. Conf. Auto. Robot Syst. Competitions (ICARSC)*, Apr. 2019, pp. 1–6.

[26] V. Azimi, D. Simon, and H. Richter, "Stable robust adaptive impedance control of a prosthetic leg," in *Proc. ASME Dyn. Syst. Control Conf.*, 2015, pp. 1–10, Art. no. V001T09A003.

[27] K. Kameyama, "Particle swarm optimization-a survey," *IEICE Trans. Inf. Syst.*, vol. 92, no. 7, pp. 1354–1361, 2009.

[28] K. D. Katyal *et al.*, "In-hand robotic manipulation via deep reinforcement learning," in *Proc. Conf. Neural Inf. Process. Syst.*, vol. 1, 2016, pp. 1–5.

[29] G. Vasan and P. M. Pilarski, "Learning from demonstration: Teaching a myoelectric prosthesis with an intact limb via reinforcement learning," in *Proc. Int. Conf. Rehabil. Robot. (ICORR)*, Jul. 2017, pp. 1457–1464.

[30] M. Mudigonda, P. Agrawal, M. Deweese, and J. Malik, "Investigating deep reinforcement learning for grasping objects with an anthropomorphic hand," in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–5.

[31] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, Bradford Books, 2018.

[32] A. V. Hill, "The heat of shortening and the dynamic constants of muscle," *Proc. Roy. Soc. London. B-Biol. Sci.*, vol. 126, no. 843, pp. 136–195, 1938.

[33] L. Kidzinski *et al.*, "Learning to run challenge: Synthesizing physiologically accurate motion using deep reinforcement learning," in *Proc. NIPS*. Cham, Switzerland: Springer, 2018, pp. 101–120.

[34] D. G. Thelen, "Adjustment of muscle mechanics model parameters to simulate dynamic contractions in older adults," *J. Biomech. Eng.*, vol. 125, no. 1, pp. 70–77, Feb. 2003.

[35] D. P. LaRoche, E. D. Millett, and R. J. Kralian, "Low strength is related to diminished ground reaction forces and walking performance in older women," *Gait Posture*, vol. 33, no. 4, pp. 668–672, Apr. 2011.