Robotica (2022), **40**, pp. 2189–2214
doi:10.1017/S0263574721001600

CAMBRIDGE
UNIVERSITY PRESS

RESEARCH ARTICLE

A model-free deep reinforcement learning approach for control of exoskeleton gait patterns

Lowell Rose[1], Michael C. F. Bazzocchi[1,2,*] and Goldie Nejat[1,3]

[1]Autonomous Systems and Biomechatronics Laboratory, Department of Mechanical and Industrial Engineering, University of Toronto, Toronto, Canada,[2]Department of Mechanical and Aeronautical Engineering, Clarkson University, Potsdam, NY, USA and [3]Toronto Rehabilitation Institute, Toronto, Canada
*Corresponding author. E-mail: mbazzocc@clarkson.edu

Received: 16 December 2020; **Revised:** 25 August 2021; **Accepted:** 12 October 2021;
First published online: 15 December 2021

Keywords: exoskeletons, human-exoskeleton interaction, deep reinforcement learning, over-ground gait rehabilitation, model-free control

Abstract

Lower-body exoskeleton control that adapts to users and provides assistance-as-needed can increase user participation and motor learning and allow for more effective gait rehabilitation. Adaptive model-based control methods have previously been developed to consider a user's interaction with an exoskeleton; however, the predefined dynamics models required are challenging to define accurately, due to the complex dynamics and nonlinearities of the human-exoskeleton interaction. Model-free deep reinforcement learning (DRL) approaches can provide accurate and robust control in robotics applications and have shown potential for lower-body exoskeletons. In this paper, we present a new model-free DRL method for end-to-end learning of desired gait patterns for over-ground gait rehabilitation with an exoskeleton. This control technique is the first to accurately track any gait pattern desired in physiotherapy without requiring a predefined dynamics model and is robust to varying post-stroke individuals' baseline gait patterns and their interactions and perturbations. Simulated experiments of an exoskeleton paired to a musculoskeletal model show that the DRL method is robust to different post-stroke users and is able to accurately track desired gait pattern trajectories both seen and unseen in training.

1. Introduction

Stroke is a leading cause of disability, with over 80 million individuals living with stroke worldwide [1]. Lower-body exoskeletons have been recently used to aid in post-stroke gait recovery [2], namely they have been used in physiotherapy for post-stroke individuals [3, 4], to deliver the task-specific, repetitive movements required for effective gait rehabilitation [5]. The use of exoskeletons in rehabilitation can assist in helping post-stroke individuals to increase their walking speed, restore their gait symmetry, and increase their range of motion, as well as aid in restoring mobility and overall functionality [2, 6, 7].

In the development of control strategies for lower-body exoskeletons, common areas of research include increasing the effectiveness, accuracy, and adaptability of an exoskeleton between different patients and levels of functions [8–10]. Existing control strategies have often focused on controlling predetermined gait patterns for users through trajectory-tracking control [4, 11–13]. While this can permit a moderate level of recovery through gait rehabilitation, it can result in less engagement and participation from patients, leading to insufficient improvement in gait recovery and motor learning, as well as potential user discomfort [11, 13–16]. Personalized gait training that is tailored for individual users has been shown to lead to greater gait recovery [10, 17–20]. Furthermore, exoskeleton control methods that offer assist-as-needed control [4, 6, 11, 21, 22], that is, where only the level of torque or power required to assist a user in reaching a goal is provided, have also shown increased gait improvements through active user engagement and participation [13, 23].

These aforementioned control strategies require predefined dynamics models to characterize the equations of motion of the user and exoskeleton and determine the required actuator torques needed to control the exoskeleton while interacting with the user [14, 21, 24–29]. For example, in ref. [26], exoskeleton trajectories were predicted and generated using spatiotemporal features of a users' gait pattern, through a dual unscented Kalman filter. A dynamics model of an exoskeleton and user was used, where the user's limb and exoskeleton were modelled as one body. An impedance-based supervisory controller was incorporated to follow a trajectory and synchronize it with the user during locomotion with a defined friction and stiffness-based torque model. Models such as these can adapt online to users and their variations in performance as well as correct for any nonlinearities and dynamic uncertainties that arise from the interaction. However, accurately defining these dynamics models is a challenge due to the complexity of the human-exoskeleton interaction, and the resulting dynamic uncertainties, nonlinearities, and perturbations that exist. Exoskeleton-user interactions can result in additional unmodelled nonlinearities such as friction and user perturbations [51–53]. These complex dynamics models have often been simplified by considering: (1) only the forces resulting from the interaction of the user and exoskeleton [28], or (2) combining the exoskeleton and user as a single body [26, 27, 44, 46]. A simplified dynamics model, however, is not able to represent fully the user-exoskeleton interactions, which can in turn affect the accuracy of the controller [24, 53, 54]. Therefore, these dynamics models are often unable to precisely model the interactions between the user and exoskeleton [26, 30–34].

Reinforcement learning (RL) techniques have been developed more recently to learn the dynamics models of different users, by learning user-specific optimal control parameters through automated selection and tuning [35, 36]. For example, in ref. [35], human motion trajectories for an exoskeleton were modelled with dynamic movement primitives (DMPs) to adapt to movements from the users. A coupled cooperative primitive (CCP) was used, where a user-exoskeleton interaction term was incorporated into a conventional DMP. An impedance-based spring-damper model was then used for modelling the interaction. Using imitation learning, the CCPs were first learned by observing user movement patterns, and RL was then used to solve for the scaling, damping and stiffness parameters of the coupled primitives. By learning these parameters through RL, the dynamic uncertainties and forces due to the human-exoskeleton interaction were reduced. However, traditional RL techniques such as these are constrained to discrete observation and action spaces, which are often represented as controller parameters (e.g., damping, stiffness) [8, 35, 36, 38, 39] or joint torque actions in a discrete system [40]. Storing continuous observation or action values as discrete state-action pairs can become infeasible due to the curse of dimensionality [41].

Model-free deep reinforcement learning (DRL) methods address this limitation by directly using high-dimensional, continuous observation and action spaces needed by exoskeletons to learn from the observations of joints, including position, velocity, and acceleration, and provide continuous joint torques actions. In general, DRL has been shown to be effective in learning locomotion skills [30, 57–62] and assisting in controlling certain exoskeleton joint-based actions such as maintaining user stability and knee joint movements [47–50]. However, they have not yet addressed the specific challenge of learning exoskeleton control of hip-knee-ankle gait patterns desired in physiotherapy, for use in over-ground gait rehabilitation. Applying DRL to exoskeleton control problems remains very challenging as classical RL algorithms require large numbers of training samples and from numerous people, which are impractical for robots working in human-robot interaction scenarios in clinical settings especially with patients. Furthermore, the controller must be designed to adapt to different user needs as well as changing needs of the same user during gait rehabilitation. The advantage of model-free DRL is that it does not require a dynamics model and instead only learns through the direct interaction of the exoskeleton with its environment [32]. More information on related works for exoskeleton control is presented in detail in Section 2.

In this paper, we present the development of a generic model-free Deep Deterministic Policy Gradient (DDPG) DRL approach in order to account for varying post-stroke individuals and desired gait patterns. The controller is the first end-to-end generic DRL-based approach for over-ground exoskeleton control of multiple desired gait patterns with multiple users with varying movement functionality. In contrast

https://doi.org/10.1017/S0263574721001600 Published online by Cambridge University Press

to existing controllers, it does not require individualized or predefined dynamics models as needed in adaptive control strategies. It further is not constrained to learn the discrete controller parameters that are required in existing RL-based exoskeleton control techniques. The controller can adapt to the limited movements of multiple post-stroke individuals through different stages of physiotherapy by learning the torque actions for the hip, knee, and ankle joints of exoskeleton actuators. The approach is able to train on different desired gait patterns and post-stroke baseline gait patterns with varying levels of movement functionality, their user-exoskeleton interactions and perturbations, namely the DRL controller is able to actively detect the deviations from the trained desired gait pattern due to the post-stroke individual's perturbations and existing movements and applies the necessary corrective actuator torque actions to the exoskeleton joints to minimize the joint angle error. This allows for accurate control of desired gait patterns both seen and unseen in training, and between different exoskeleton users.

In our previous work [42], we developed a DRL technique for user-specific exoskeleton control, which was trained on only one user's baseline gait pattern and one desired gait pattern obtained from the OpenSim Gait2354 model. The controller was limited in that it required training for each exoskeleton-user pair, when the movements of the post-stroke individual improved during physiotherapy sessions. Therefore, the procedure was not adaptable to changes. In contrast, this paper extends our previous DRL approach to incorporate various desired and baseline gait patterns from the same user or multiple users. These gait patterns allow the DRL controller to adapt to multiple users and multiple desired gait patterns with varying levels of movement functionality (both seen and unseen in training). This is accomplished through training the modified controller on a large dataset of gait patterns from both healthy and post-stroke individuals, as well as through the redesign of the exploration process, reward function, and deep neural network architecture. Details of the proposed DRL control scheme and the adaptations employed in this work are presented in Sections 3 and 4.

The paper is organized as follows: Section 2 provides an overview of related works on exoskeleton control, particularly model-based adaptive control techniques, RL approaches, and DRL control methods. In Section 3, the DRL controller employed in this work is developed, with the user-exoskeleton simulation environment presented in Section 4. In Section 5, the user data for training the exoskeleton controller are described. Section 6 presents and discusses test results, with concluding remarks presented in Section 7.

2. Related works

Previous research that has investigated assistive exoskeleton control techniques for adapting to different users can be categorized into: (1) model-based adaptive control techniques [26–28, 43–46], and (2) RL approaches [35, 37–40]. More recently, DRL control methods have been investigated for learning lower and upper limb exoskeleton-based joint control tasks such as assisting in fall prevention with a hip exoskeleton or augmenting knee movements with a knee-based exoskeleton [47–50]. To-date, however, a DRL-based approach has not yet been applied to end-to-end exoskeleton control for over-ground gait training of desired gait patterns.

2.1. Adaptive model-based control approaches

Traditional adaptive control techniques rely on predefined dynamics models of both the exoskeleton and user to adapt to user-specific characteristics such as walking speed or step length [18, 44], and to synchronize with the movements of exoskeleton users through feedback from exoskeleton-user interactions [27, 28, 45, 46].

For example, in ref. [27], a compliance-based control technique was used for exoskeleton motion by tracking a user's trunk inclination, foot pressure, and joint angles. Parameterized trajectories from an inverted pendulum model were first generated through inverse kinematics, which were controlled on spring-based exoskeleton joints with specified stiffness parameters. The generated trajectory was then

https://doi.org/10.1017/S0263574721001600 Published online by Cambridge University Press

used as equilibrium points for the compliance controller. This allowed the exoskeleton to slightly deviate from the equilibrium when perturbed by the user.

In ref. [44], endpoint model predictive control (MPC) was used to provide torque control on an exoskeleton modelled as a double pendulum and generate online joint trajectories. To create endpoint references for the MPC controller, walking speed, swing duration, and step length were used as inputs. The MPC controller then computed optimized joint torque actions for a full prediction horizon from a cost function based on the reference trajectory. A computed torque from the first time step was then used as the current control action. Exoskeleton joint angles and velocities were obtained as feedback for the controller, and user-specific gait assistance was used for the swing phase of a user gait pattern.

In ref. [28], an impedance controller for an exoskeleton for assist-as-needed was demonstrated. The exoskeleton only provided active force when a user had difficulty reaching a joint angle goal on their own. The controller observed the interaction forces between the exoskeleton and their limbs using a force-field and computed the corresponding joint torques through model-based compensation. The impedance parameters and level of assistance were then adapted based on a user's position within the force field. An impedance controller was also presented in ref. [43] for providing assist-as-needed control. The endpoint stiffness of an exoskeleton modelled as a two-link pendulum was specified based on a user's deviations throughout the phases of their gait and was altered in direction and magnitude as the exoskeleton followed a model-reference foot trajectory.

In ref. [45], a feedback controller with virtual constraints was developed for a lower-body exoskeleton. The controller was intended for users with spinal cord injury to walk in the exoskeleton without additional support to maintain their balance. This dynamics model of the exoskeleton and user was represented as a single body. The controller provided virtual constraint parameters to regulate both the exoskeleton's forward hip velocity and posture to synchronize with the other exoskeleton joints. This approach was then used for offline trajectory optimization for generating gait patterns that could be followed accurately online.

In ref. [46], trajectory tracking control on an exoskeleton was optimized through the use of a linear quadratic regulator (LQR), where human-exoskeleton interaction-based disturbances were also addressed. The exoskeleton was modelled as a double pendulum, with the inputs to the LQR controller composed of feedback proportional, derivative, and integral-based actions, and a feedforward reference torque. The cost function for the LQR contained weighting matrices for the state and input. Least squares estimation was then used to experimentally derive the exoskeleton dynamics model parameters for use in the controller.

The adaptive control approaches described above all require dynamics models of the user and exoskeleton to determine appropriate control actions. However, these complex dynamics models can be difficult to accurately model due to the interactions of the user and exoskeleton and their resulting nonlinear characteristics.

2.2. Reinforcement learning in exoskeleton control

Reinforcement learning for exoskeleton control has been used in a variety of tasks where an optimized or user-specific controller is desired, namely RL has been applied in exoskeleton control to: (1) find optimal parameters of adaptive controllers [35, 38], (2) account for forces resulting from the human-exoskeleton interactions [39, 40], or (3) allow for assist-as-needed exoskeleton control [37].

In ref. [38], user-specific parameters of an admittance controller were learned with RL. The user-exoskeleton interaction forces were minimized when the exoskeleton was deploying a reference trajectory. This RL approach was able to tune and find optimal damping and stiffness parameters of the controller based on the user's performance, where user-exoskeleton contact forces and joint angle error were used as observations. A similar approach with an impedance controller using RL was demonstrated in ref. [55] for finding optimal impedance parameters for use on an upper limb exoskeleton.

Reinforcement learning was also used in ref. [40] to implement a model-reference compliance controller for tracking a reference trajectory on a lower limb exoskeleton. Q-learning was used to find optimal joint torque values for a joint-based compliance controller while tracking a reference trajectory

https://doi.org/10.1017/S0263574721001600 Published online by Cambridge University Press

by observing exoskeleton joint angles and velocities from a discrete-time system. A compliance controller was used to allow for user safety with the exoskeleton by permitting the user's perturbations. The controller represented the exoskeleton joints as second-order mass-spring-damper systems. These mass-spring-damper joints were then able to comply with the user-exoskeleton interaction forces.

In ref. [39], upper-limb exoskeleton assistive strategies were proposed for learning from user-exoskeleton interactions. A dynamics model and control policies were learned from limited user muscle activation data using model-based RL, where a user moved a simulated arm through electromyography (EMG)-based muscle-activation recordings. A policy search approach then found the optimal assistive torque needed to help the user in accomplishing a reaching task, while reducing their observed EMG signals and thus muscle effort.

In ref. [37], an impedance controller was modelled using an actor-critic-based approach to provide assist-as-needed control on an ankle exoskeleton. This RL approach changed the amount of assistance provided to the user by adjusting the stiffness parameters of the impedance controller's force-field, depending on the performance of the user. User performance was determined by observing the resulting tracking errors when following a reference trajectory. The controller was then able to assist users in following a sinusoidal reference pattern that was displayed on a screen for ankle movements.

The aforementioned RL-based control methods were either used for finding optimal controller parameters [35, 37–39] or for following a reference trajectory through spring-mass-damper-based joints [40]. However, they still required a model of the exoskeleton or its joints to determine the control torque or reference trajectory. They also only learned discrete actions such as impedance or admittance controller parameters. As RL is also typically constrained to discrete low-dimensional observation and action spaces, it is unable to learn in high-dimensional or continuous systems [55]. However, to accurately control an exoskeleton, continuous high-dimensional observation and action spaces are needed for representing joint angles, velocities, and accelerations as well as actuator torque outputs. Discretizing these spaces would be infeasible as the required state-action pairs would become too numerous to store [41].

2.3. Deep reinforcement learning control approaches

A key benefit of DRL for exoskeleton control is its ability to learn and provide control in high-dimensional continuous joint observations and continuous actuator torque actions. This is achieved by mapping states to actions through deep neural networks approximating the policy and value functions of RL [32]. The use of these continuous observation spaces allows for actuator torque actions to be learned in an end-to-end nature directly from sensory observations of joint parameters [30, 56]. In particular, the advantage of model-free DRL is that it does not require a predefined dynamics model. To date, it has been used in the literature to learn: (1) human locomotion [57–59], (2) control of bipedal robots [30, 60–62], and (3) control of upper and lower limb exoskeleton joints [47–50].

In refs. [57]–[59], locomotion skills for musculoskeletal models were learned with DRL, using observations of joint angles, velocities, and accelerations, and muscle activations and joint torques as actions. DRL was also used to learn motor and locomotion skills for legged and bipedal robots, with speed, distance traversed, and stability as goals in learning [30, 60–62]. However, unlike in the case of rehabilitative lower-body exoskeletons, bipedal robots trained with DRL tend to define rewards for robot balance and remaining upright in order for the robots to be able to walk independently. In contrast, rehabilitative lower-body exoskeletons are used to provide assistance/support as needed based on the resistance of the user wearing them and the need to consider user-exoskeleton interactions, as opposed to performing the walking task completely for the user.

In applications for exoskeletons, DRL was used for learning an optimal control method for upper limb functional electrical stimulation (FES), while a user wore a passive elbow exoskeleton [48]. A Proximal Policy Optimization (PPO) algorithm was used to learn the control policy, where the elbow's joint position, velocity, and acceleration were used as observations with the error from the joint angle goal. Continuous actions of electrical stimulations to the arm muscles were, thus, able to be learned to provide elbow extension motions for the user to reach desired joint angles.

https://doi.org/10.1017/S0263574721001600 Published online by Cambridge University Press

In ref. [50], a normalized advantage function (NAF) algorithm was used to provide mirror therapy for a lower-limb exoskeleton. The approach allowed a seated patient's impaired leg to follow their functional limb through exoskeleton joint torque actions, while maximizing the impaired leg's EMG recorded muscle activations. This approach also aimed to minimize tracking error between a model-reference impedance controller on the functional limb and the follower controller on the impaired leg, by observing hip and knee joint angles, velocities, and a user's muscle activations.

In ref. [47], DRL was used for learning a policy to predict and recover from falls with an exoskeleton for fall prevention in older adults. Torque commands were sent to a hip exoskeleton to augment a user's locomotion when a possible fall was detected. A human locomotion policy was first learned using PPO, which was used for training the fall recovery policy. The PPO algorithm was used to observe the angular position and velocity of the hip joints and learn joint torque actions for the exoskeleton hip actuator. A classifier using a support vector machine (SVM) was then trained to predict observations that could result in a fall, and the hip exoskeleton was used to validate the recovery policy by providing a torque to help the user in maintaining their balance.

In ref. [49], a DRL-based rehabilitation game was developed, where a user wearing an EMG-controlled exoskeleton on their knee controlled a game character. A thigh worn EMG sensor was used to record a user's muscle activations, and this was then transferred into movements on the exoskeleton to control the character in the game. A deep-Q network algorithm (DQN) used a convolutional neural network (CNN) to observe images from the game to determine the character's position. The DQN controller then outputted discrete translational movement actions to control the game character. The user was then assisted in the game through the DRL agent augmenting their EMG-based exoskeleton control by providing assistive joint movements on the exoskeleton when needed based on the character movements in the game.

This paper demonstrates the advantages of using DRL to assist users as they progress through different stages of physiotherapy. In the following section, we present the development of a novel model-free generic end-to-end DRL-based control, which uses DDPG to uniquely learn different gait patterns to assist multiple users as they progress through different stages of physiotherapy and recovery. Subsequently, the DRL controller is trained on several post-stroke individuals' existing baseline gait patterns at varying levels of movement functionality, as described in Section 4. Training on these gait patterns within the simulation environment (Section 5) will minimize the need to re-train the exoskeleton for each new user or changes in gait patterns over time. The DDPG controller can observe continuous and high-dimensional hip, knee, and ankle joint parameters of the exoskeleton user and provide the necessary exoskeleton actuator torque actions needed to accurately follow the desired gait patterns, as demonstrated by the experiments and results presented in Section 6.

3. Exoskeleton control with DRL

Our DRL approach uses the DDPG [41] algorithm in the Keras-RL library, [63] to enable end-to-end learning of a torque controller to accurately follow any desired gait pattern on a lower-body exoskeleton. We present the first use and adaptation of the DDPG algorithm for exoskeleton control of hip-knee-ankle gait patterns. The overall DRL architecture is presented Fig. 1 with respect to its main modules and information flow between them (Fig. 1(a)), as well as its corresponding control scheme block diagram (Fig. 1(b)). The DDPG framework is primarily composed of actor and critic neural networks to represent the policy and value function of RL, respectively [41]. The actor network takes actions based on state observations (joint parameters) obtained directly from the environment and its current policy, while the critic determines the value of these actions (actuator torques) based on the observations, and receives a reward. The simulation environment contains a musculoskeletal model with an exoskeleton model attached to it within OpenSim-RL [64]. As we use a model-free DRL approach, the policy only learns through the interaction of this model with the environment. The details of our DRL technique are discussed in the following sections.

https://doi.org/10.1017/S0263574721001600 Published online by Cambridge University Press

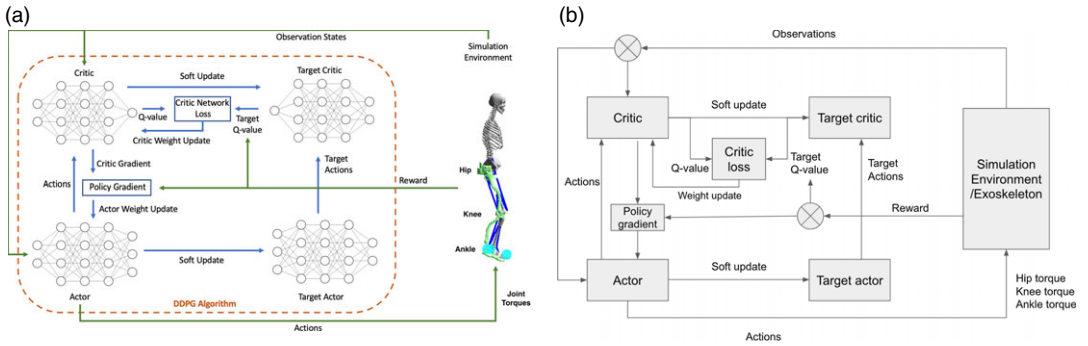

Figure 1. *Overview of DRL architecture for exoskeleton control. (a) Diagram of DRL architecture. (b) Block diagram representation of the DRL control scheme.*

3.1. Deep Deterministic Policy Gradient (DDPG)

DDPG is an off-policy, model-free, actor-critic DRL technique [41]. In this DRL framework, the agent first obtains an observation state of joint parameters from the environment, s_t, and takes an actuator torque action, a_t based on the actor's policy. It receives a reward from the defined reward function, r_t, as outlined in Section 3.2, below as well as joint parameter observations from the next state, s_{t+1}, in each time step, t, and maximizes the discounted future cumulative reward[41]:

$$R = \sum_{i=t}^{T} \gamma^{(i-t)} r_i \left(s_i, a_i\right), \tag{1}$$

where $\gamma \in [0, 1]$ is a discounting factor used to emphasize the influence of future rewards over those immediately obtained.

The critic, $Q\left(s, a\right)$, uses the Bellman equation to provide the Q-value, which approximates the expected return after taking exoskeleton actuator torque actions in a particular state. The actor policy, $\mu(s|\theta^{\mu})$, is parameterized deterministically by mapping specific joint observation states to torque actions and is updated by the policy gradient. The policy gradient approach works by updating the policy through gradient descent based on the Q-value estimated by the critic network and its gradient. In this DRL algorithm, deep neural networks are used to approximate the policy and value functions [41].

During the training process, the actor network, $\mu(s|\theta^{\mu})$, and the critic network, $Q(s, a|\theta^{Q})$, are first initialized with random weights θ^{μ} and θ^{Q}. With these initial random weights, the actor policy has no initial knowledge of the value of actuator torque actions or which will be highly rewarded. Through training, the actor policy learns this by updating its weights based on the critic's Q-value approximations.

In the DDPG algorithm, target networks are also initialized. This is based on the method used in the DQN algorithm [65], where target networks based on copies of the original network weights assisted in learning. The copied actor and critic target networks aid in maintaining stability in learning and prevent divergence when the target Q-value is determined, addressing issues that may arise from estimating the Q-value based on the same critic network that is being updated [41]. These networks, μ' and Q', have the following weights:

$$\theta^{\mu'} \leftarrow \theta^{\mu}, \tag{2}$$

$$\theta^{Q'} \leftarrow \theta^{Q}. \tag{3}$$

A replay buffer, B, is also provided in order to store the transitions of experience, including the current and subsequent time steps' state observations of joint parameter information, the actuator torque actions, and reward from the defined reward function (s_t, a_t, r_t, s_{t+1}). A random process $\mathcal{N}$ is used during training to encourage action exploration in the actuator torque space, based on a Gaussian white noise process.

https://doi.org/10.1017/S0263574721001600 Published online by Cambridge University Press

Improving on our previous work [42], we extended the DDPG implementation by adding Gaussian white noise (with $\sigma = 0.3$) in order to allow for greater exploration of the action space and to avoid convergence to local minima [67]. This type of action space noise is supported in more recent updates to the DDPG algorithm [67] and improves upon the Ornstein Uhlenbeck process used in ref. [42]. A higher maximum torque was used for each actuator to account for larger ranges of motion and joint speeds based on clinical data from ref. [78]. In the reward function, we introduced a new joint angle error linear penalization function to minimize the mean absolute error. In the actor neural network architecture of this work, a hyperbolic tangent activation function is introduced in the output layer of the network to replace the sigmoid activation function used in ref. [42]. The updated activation function allows for a range of normalized actions within -1 to 1; thereby increasing the allowable range of actuator torque outputs. It also provides greater stability and performance in training than the sigmoid activation function [89]. Furthermore, in this work, action clipping was added to the action outputs at each time step to ensure the maximum torque was not exceeded with the exploration noise, and the absolute error at each time step of an episode for each joint was added to the observation space as input to both the actor and critic networks, thereby improving the accuracy of the controller. This work is the first application of end-to-end generic DRL-based control of exoskeletons for multiple gait patterns and users.

In each time step of training, a joint observation state is received. The observations are composed of the: (1) exoskeleton hip, knee, and ankle actuator torque and velocity, (2) joint angles, velocities and accelerations for the hip, knee and ankle joints of the musculoskeletal model, (3) the current error for each joint, and (4) the goals at the current time step. These goals are composed of the desired joint angles to be reached for all three joints of the musculoskeletal model, which over a full episode of training represent one gait cycle of the desired gait pattern.

For each subsequent time step, torque action for each actuator of the exoskeleton, $a_t = \mu\left(s_t|\theta^{\mu}\right) + \mathcal{N}_t$, is performed based on the actor's current policy and added noise for exploration [41]. Observations from a new state and reward are then obtained. The state transition tuples, (s_t, a_t, r_t, s_{t+1}), are then stored in the replay buffer.

A target Q-value, y_t, is then determined based on the target networks, μ' and Q', and receives the reward r_t, using the Bellman equation [41]:

$$y_t = r_t + \gamma Q'\left(s_{t+1}, \mu'\left(s_{t+1}|\theta^{\mu'}\right)|\theta^{Q'}\right). \tag{4}$$

The critic network weights are then updated through minimizing the mean squared error loss, L, between the target Q-value and the current Q-value estimated by the critic network, over a batch of samples of state transitions gathered from the replay buffer:

$$L = \frac{1}{N}\sum_{t}\left(y_t - Q\left(s_t, a_t|\theta^{Q}\right)\right)^2. \tag{5}$$

The actor network weights are then updated by following the policy gradient based on the value estimated by the critic network [41]. The parameter optimizer for the actor and critic networks uses the Adam optimizer [68] to update the network weights. This is a commonly used approach for stochastic gradient-based optimization, where a learning rate of 0.001 is used for the actor and critic networks.

The target network weights are then updated based on the original actor and critic networks; however, a soft update term is used to allow the main networks weights to be slowly tracked, which encourages stability in learning [41, 65]:

$$\theta^{Q'} \leftarrow \tau\theta^{Q} + (1-\tau)\,\theta^{Q'}, \tag{6}$$

$$\theta^{\mu'} \leftarrow \tau\theta^{\mu} + (1-\tau)\,\theta^{\mu'}, \tag{7}$$

with $\tau \ll 1$.

As an episode progresses through each time step, the hip, knee, and ankle joint angle goals are updated in the gait patterns. Throughout a full episode, the DRL agent learns to follow all joint angles of a desired gait pattern in order to maximize the received reward and thus track the desired gait pattern.

https://doi.org/10.1017/S0263574721001600 Published online by Cambridge University Press

In this implementation of the DDPG algorithm, the first layer of the actor network takes state observations as inputs and contains three hidden layers with a rectified linear unit (ReLU) activation function in each. The output layer is composed of the torque actions, with a hyperbolic tangent (tanh) activation function. The critic network has the observations and actions as inputs in its first layer and is composed of three hidden layers with ReLU activation functions, with its output layer containing the estimated Q-value through a linear activation.

3.2. Reward function

To encourage the DRL agent to find optimal actions, a reward function is defined. The reward function is composed of the following penalties being applied: (1) when there is an offset between the desired joint angle and current joint angles of each joint, and (2) when a joint exceeds a minimum or maximum value. This reward function encourages the exoskeleton to track the desired gait patterns by minimizing the error between the desired joint angles of the musculoskeletal joint angles and also provides a penalty when a joint angle state is far from its goal. The reward function is composed of linear joint reward functions and ramp functions, improving on the Gaussian-based reward function designed in ref. [42].

In training, the desired gait pattern at each episode provides the joint angle goals at each time step for the hip, knee, and ankle joints and is updated at the start of each episode.

For all joints, the total cumulative reward R_E, in each episode, E, is the summation of individual joint rewards over all episode time steps, n, and is represented as:

$$R_E = \sum_{i=0}^{n} \left(w_h r_{i_h} + w_k r_{i_k} + w_a r_{i_a} \right), \tag{8}$$

where r_i is the individual reward for each joint, w_i is the weight for each joint, with h, k, a representing the hip, knee, and ankle joints, respectively.

The reward for a single joint at each time step, r_i, is defined as:

$$r_i = w_F \mathcal{F}(q_i) + w_G \mathcal{G}\,(q_i)\,, \tag{9}$$

where $\mathcal{F}$ and $\mathcal{G}$ are reward function components, and w_F and w_G their weights. $\mathcal{F}$ provides the penalty for the joint angle error at each time step, and $\mathcal{G}$ represents the penalty when a defined minimum or maximum joint angle is exceeded.

$\mathcal{F}(q_i)$ is defined as a linear function:

$$\mathcal{F}(q_i) = -\left| q_i - q_{d_i} \right|, \tag{10}$$

which is represented as the absolute difference between the joint angle, q_i, and the desired joint angle, q_{d_i} at the current time step.

To aid the agent in finding the range of ideal actions during exploration and learning, a penalty, $\mathcal{G}(q_i)$, is also applied if a joint exceeds a defined minimum or maximum joint angle:

$$\mathcal{G}(q_i) = -M\,(y_{\max}) - M\,(y_{\min})\,, \tag{11}$$

where

$$M\,(y) = \begin{cases} 0 & y \le 0 \\ y & y > 0 \end{cases}, \tag{12}$$

$$y_{\max} = q_i - q_{\max}, \tag{13}$$

and

$$y_{\min} = q_{\min} - q_i. \tag{14}$$

$q_{\min}$ and $q_{\max}$ above are the set minimum and maximum joint angles, respectively, and are defined as 20 degrees below and above the minimum and maximum desired joint angles. The penalty is then

https://doi.org/10.1017/S0263574721001600 Published online by Cambridge University Press

added only if the minimum or maximum joint angles are surpassed, by using the ramp function $M(y)$. Energy consumption by the agent in not considered in the reward function, as is commonly done in DRL exoskeleton control (e.g., refs. [47, 48, 50, 86]), since it would limit the assistive capability of the exoskeleton. Therefore, the only restriction on energy consumption, herein, is limiting the peak torque of the system. Furthermore, for our case, impedance added from the musculoskeletal model's muscles restrict sudden torque movements and the reward function penalizes the error that may be resulting from such sudden movements during training.

In the next section, the minimum and maximum joint angles for the exoskeleton system are defined as well as the simulation environment. The simulation environment, which includes a lower-body exoskeleton model and a human musculoskeletal model, is developed so that the DRL agent can explore and learn within a constrained action space. The exoskeleton joint angles are constrained to prevent the agent from taking actions that are unsafe or infeasible during exploration, rewarding, and penalizing joint actions according to Eq. (8).

4. User-exoskeleton simulation environment

A simulation environment in the OpenSim API platform [69] has been developed that includes a 3D exoskeleton model paired to a 3D musculoskeletal model. OpenSim-RL is used as the platform to provide the RL framework and environment [64]. It is based on OpenAI Gym [76], a widely used platform for performing DRL benchmarking and research, and utilizes the OpenSim API. To incorporate the dynamics, motion, and interaction forces between the musculoskeletal joints, muscles, bones, and exoskeleton components in the OpenSim-RL environment, the Simbody physics engine [77] is used. The DRL agent described in Section 3 explores and learns through the actuation of the hip, knee, and angle joints of the exoskeleton model within the simulation environment. The inputs to the controller are the environment observations, i.e., (i) exoskeleton hip, knee, and ankle actuator torques and velocities, (ii) joint angles, velocities, and accelerations for the hip, knee, and ankle joints of the musculoskeletal model, (iii) current error in the joint angles, and (iv) the desired joint angles at the current time step. The control outputs are the individual torque actions to each of the joints.

The 3D hip-knee-ankle exoskeleton model is adapted from ref. [70] and contains a hip attachment, and thigh, shank, and foot linkages, Fig. 2. The exoskeleton motors are added as torque actuators from the OpenSim API to each exoskeleton joint. In this work, a new exoskeleton model with 6 degrees of freedom (DOFs) is simulated based on the H2 Exoskeleton from Technaid [12] and used in OpenSim. Table I presents the kinematic and dynamic specifications for the exoskeleton model. An actuator at each joint of the exoskeleton model, namely hip, knee, and ankle, provides a degree of freedom to the system. The maximum torque the actuator can provide is 220 Nm, which is aligned with the maximum torque on the H2 [12]. The model used in this work improves upon the H2 exoskeleton modelled in ref. [71], which was represented with simplified geometry and dynamics, that is, links and inertia tensors were modelled as rectangular prisms. The simplified model of the H2 exoskeleton is extended to include the joint angle limits as well as the dimensions, masses, and moments of inertia for each link of the exoskeleton provided in ref. [70] in order to closely match the H2 exoskeleton in ref. [12], and more effectively model the exoskeleton and evaluate the control method proposed. Each exoskeleton joint is constrained to simulate a motion range similar to real exoskeleton joints [71], and prevents unrealistic and dangerous movements during exploration or execution, such as hyperextension in the knee. The forces in the simulated environment include acceleration due to gravity, ground reaction forces on the bottom of the exoskeleton foot plates at the heel and toe, and joint limit constraints. The ground reaction forces are modelled as Hunt–Crossley Forces [72] in the OpenSim environment. This contact model creates force interactions between contact spheres that are applied at the exoskeleton's foot plate on the heel and toe, as seen in blue on the footplate in Fig. 2, and the floor plane. The exoskeleton parameters incorporated in our simulation are summarized in Table I.

The musculoskeletal model used is adapted from OpenSim's Gait2392 model [73]. This model represents a human subject with a height of 1.8 m and mass of 76.16 kg [73]. The muscles on this

https://doi.org/10.1017/S0263574721001600 Published online by Cambridge University Press

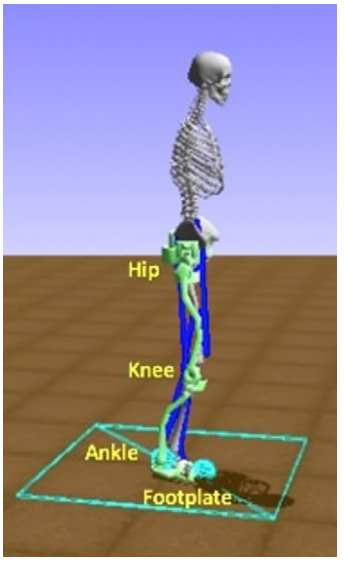

Figure 2. *OpenSim-RL-based Simulation environment, with a hip-knee-ankle exoskeleton attached to a musculoskeletal model.*

musculoskeletal model are based on a Hill-type muscle model [74], each containing a muscle-tendon unit with specific properties such as muscle fibre length, tendon slack length, and maximum isometric force. Passive forces from the muscle-tendon unit are incorporated into the simulation. To simulate a post-stroke individuals' baseline gait pattern, hip-knee-ankle joint torque patterns are applied to the joints of the musculoskeletal model through an additional torque actuator added to each musculoskeletal joint [75]. In following section, gait information data from post-stroke individuals are gathered to establish datasets of desired and baseline gait patterns.

5. User data for training

User data for both the desired gait patterns and baseline gait patterns are included in training. These datasets are collected from clinical studies of post-stroke individuals and healthy users. Testing sets of desired gait patterns both seen and unseen in training are then used to validate the controller's ability for on-the-fly adaptation of desired gait patterns, for use with multiple post-stroke exoskeleton users. The patient data provide the reference joint angle positions necessary to define the goal at each step of an episode, while the controller learns the torque values required to reach that goal based on the individual user simulated, their movements, and their interactions with the exoskeleton.

5.1. Desired gait dataset

The desired gait patterns dataset for the exoskeleton consists of ankle, knee, and hip joint angle data obtained from the walking gait of 20 healthy adults ranging in age from 22 to 72 [78]. These gait patterns corresponded to an average maximum range of motion in the sagittal plane of 45.6°, 61.5°, and 34.4° for the hip, knee, and ankle joints, respectively. This dataset is representative of a wide range of healthy gait patterns with similar joint angle range [79, 80]. The subjects were recorded using a motion capture system over five trials of walking at a self-selected natural walking speed. Then, they were instructed to walk at a slower speed and also an increased speed. The subjects in this study had the following mean (SD) gait characteristics in natural walking: a step length of 0.66 m (0.06 m), stride length of 1.3 m (0.12 m), stride time of 1.1 s (0.1 s), a double support time of 13% (1.68%) of their gait cycle, and a

https://doi.org/10.1017/S0263574721001600 Published online by Cambridge University Press

Table I. *Exoskeleton parameters.*

Link	**Foot**	**Shank**	**Thigh**
Length (m)	0.26	0.42	0.47
Mass (kg)	0.5	1.5	1.5
I_{xx} (kg m^2)*	0.337	0.059	0.015
I_{yy} (kg m^2)	0.009	0.003	0.005
I_{zz} (kg m^2)	0.338	0.057	0.014
Joint limits	**Hip**	**Knee**	**Ankle**
Minimum (°)	−80	−100	−40
Maximum (°)	80	10	40
Max. torque (Nm)	220	220	220

*The I_{xx}, I_{yy}, and I_{zz}, are the moment of inertia tensor's diagonal values.

swing time of 37% (1.97%) of their gait cycle. The joint angles sequenced from these gait patterns are defined as individual joint goals at each episode time step during training. Each episode during training consists of one full gait cycle of the desired gait pattern.

In ref. [78], the subjects' recorded gait patterns were sorted into five categories of walking speeds, v, normalized with respect to their body heights, h (v/h): (1) very slow ($v/h < 0.6$), (2) slow ($0.6 \leq v/h < 0.8$), (3) medium ($0.8 \leq v/h < 1$), (4) fast ($v/h > 1$), and (5) natural (mean $v/h = 0.71$). A mean gait pattern for each of these categories was then provided. These gait patterns are represented as one gait cycle stride from heel-strike to heel-strike. Figure 3 shows the mean gait patterns of the three angle joints used for these speeds in training. For training, the episode length and duration of the gait cycle is set based on the desired gait pattern speed. This dataset also provided gait patterns for each speed that were one standard deviation from the mean gait patterns. These were then also used in training to provide additional variation in the desired gait pattern goals.

To incorporate variations in desired gait, we also used scaling factors between 0.5 and 1.2, in order to simulate different stages of gait recovery towards a final healthy gait pattern, for a range of subjects with varying motor function. A healthy gait pattern typically involves increased flexion in the knee [81], increased stride length, and a general increased range of motion [82]. These scaling factors therefore allow for setting gait pattern goals between a post-stroke individual's existing gait pattern with limited function and a final healthy desired gait pattern. In a clinical scenario, a physiotherapist would choose the scale of desired gait pattern that best fits the current motor performance of the post-stroke individual and progressively increase this as their gait and range of motion recovers. As a post-stroke individual would have limited gait function in their affected leg [82, 83] and thus a more limited amplitude of flexion [83], the range of motion for each joint in these patterns is scaled down to represent possible motion limitations in their desired gait pattern. Furthermore, these gait patterns are also scaled up to provide a larger range of motion to further challenge the user if desired, by increasing the range of motion of the gait pattern goals, which also adds variability in the training set.

As an example, the range of scaled gait joint angles for the mean natural gait pattern speed is shown in Fig. 4, where the plotted line displays the provided mean gait pattern from ref. [78], with the shaded region representing the scaled desired gait patterns from 0.5 to 1.2. All additional gait pattern speeds are scaled in a similar manner.

5.2. Baseline gait dataset

To obtain the baseline motions for the musculoskeletal model, walking gait patterns in the sagittal plane were obtained from six chronic post-stroke individuals [84]. The subjects ranged in age from 54 to 70 and were between 12 months and 55 months past the date of their stroke. Selection criteria were defined

https://doi.org/10.1017/S0263574721001600 Published online by Cambridge University Press

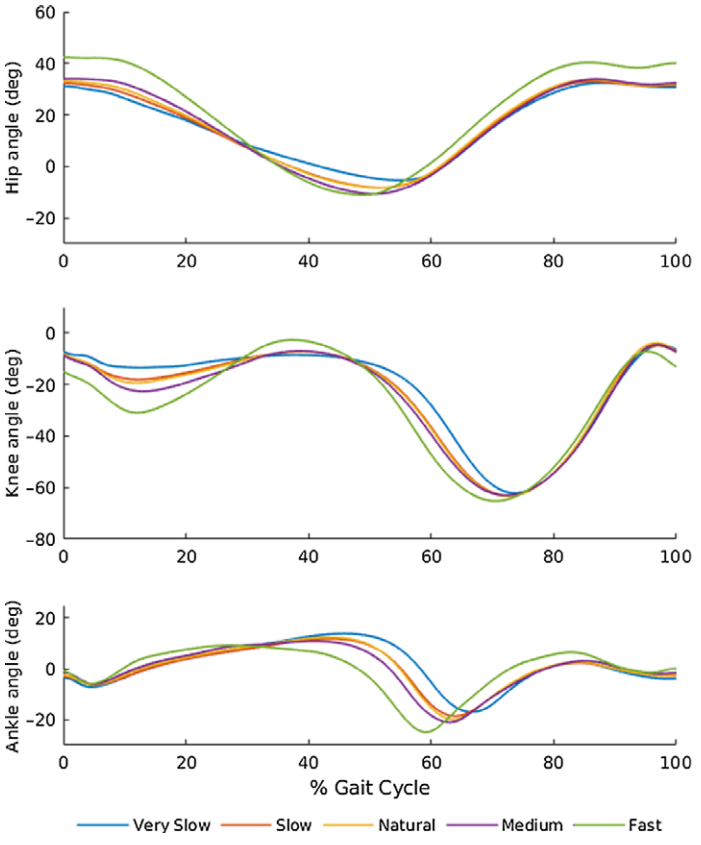

Figure 3. *Mean desired gait patterns used in training.*

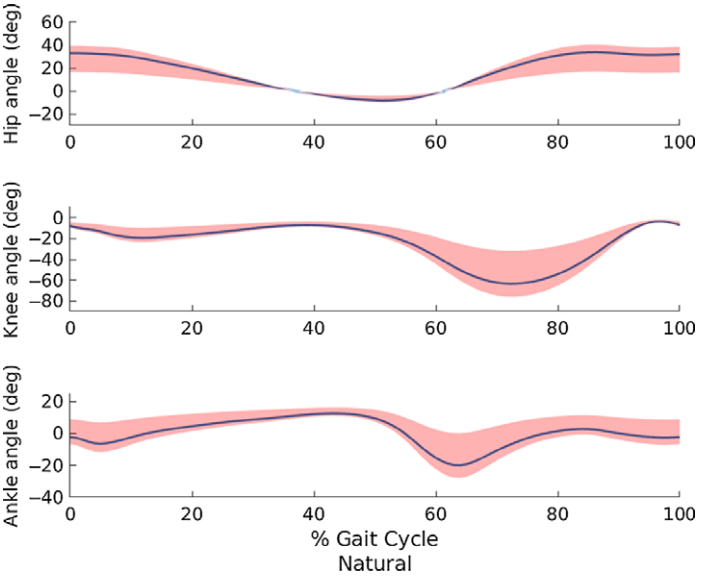

Figure 4. *Mean desired joint angle pattern for the natural gait pattern speed, with the shaded region indicating the scaling from 0.5 to 1.2.*

https://doi.org/10.1017/S0263574721001600 Published online by Cambridge University Press

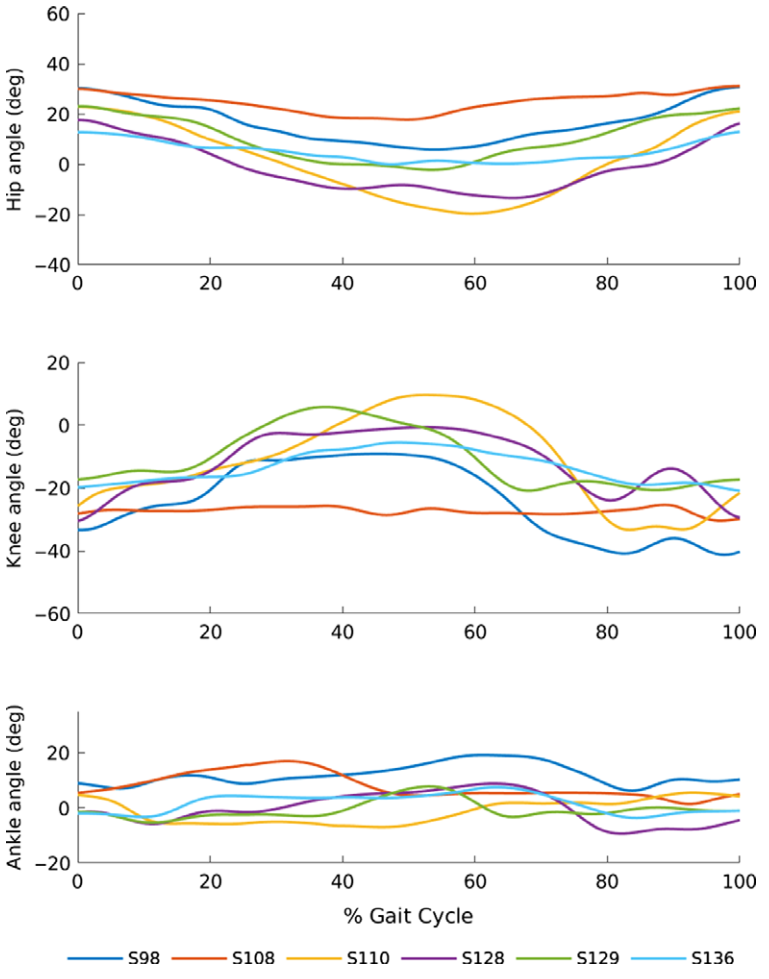

Figure 5. *Baseline gait patterns from each post-stroke individuals' affected leg, as named in ref. [84].*

as the capability to walk for 5 min at a self-selected speed in the presence of gait deficits. The post-stroke individuals had hemiparesis, or weakness, on one side of their body, which is indicated in the data. These gait patterns were captured by using a motion capture system as subjects walked at a self-selected walking speed on a split-belt treadmill. The collected motion capture data of the subjects' gait sequences was then converted to joint angles in an OpenSim motion file format [84].

Input into the DRL training consisted of the ankle, knee, and hip joint angle data from the affected leg for one gait cycle per subject, scaled similarly as the desired gait patterns, to simulate variability functionality and recovery levels for each post-stroke individual. These baseline gait patterns were then applied to the musculoskeletal model through a torque actuator added to each of the musculoskeletal model's hip, knee, and ankle joints using the OpenSim API. The baseline joint angle patterns for one cycle of each subject's gait from heel strike to heel strike are shown in Fig. 5 [84]. In the next section, the two datasets are employed to train the DDPG algorithm as well as to test the trained policy.

6. Experiments

To validate our DRL approach for exoskeleton control, we first trained the DDPG algorithm with the desired gait pattern and baseline gait pattern sets. The resulting trained policy was then verified on desired gait patterns both seen and unseen in training.

https://doi.org/10.1017/S0263574721001600 Published online by Cambridge University Press

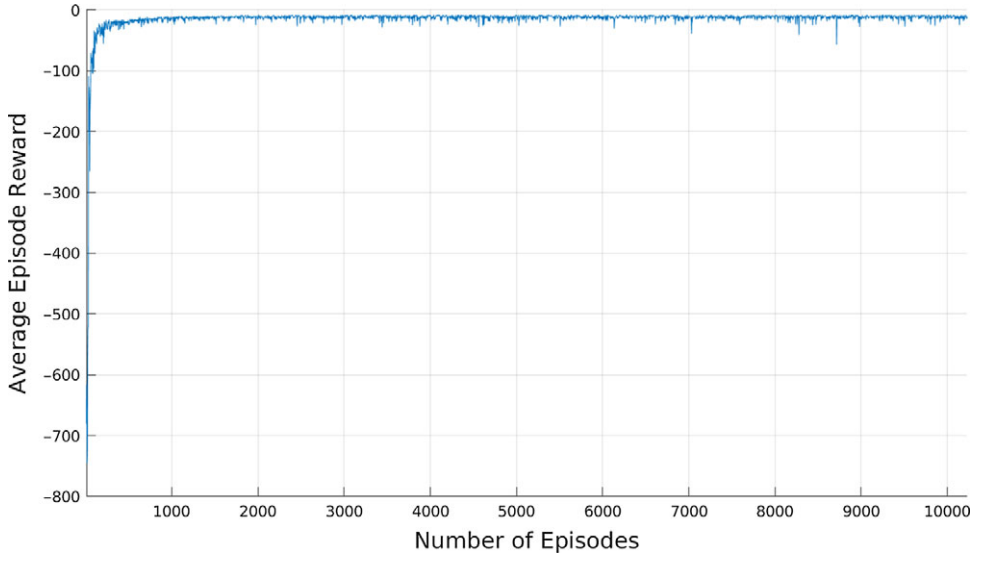

Figure 6. *Average reward per episode during training.*

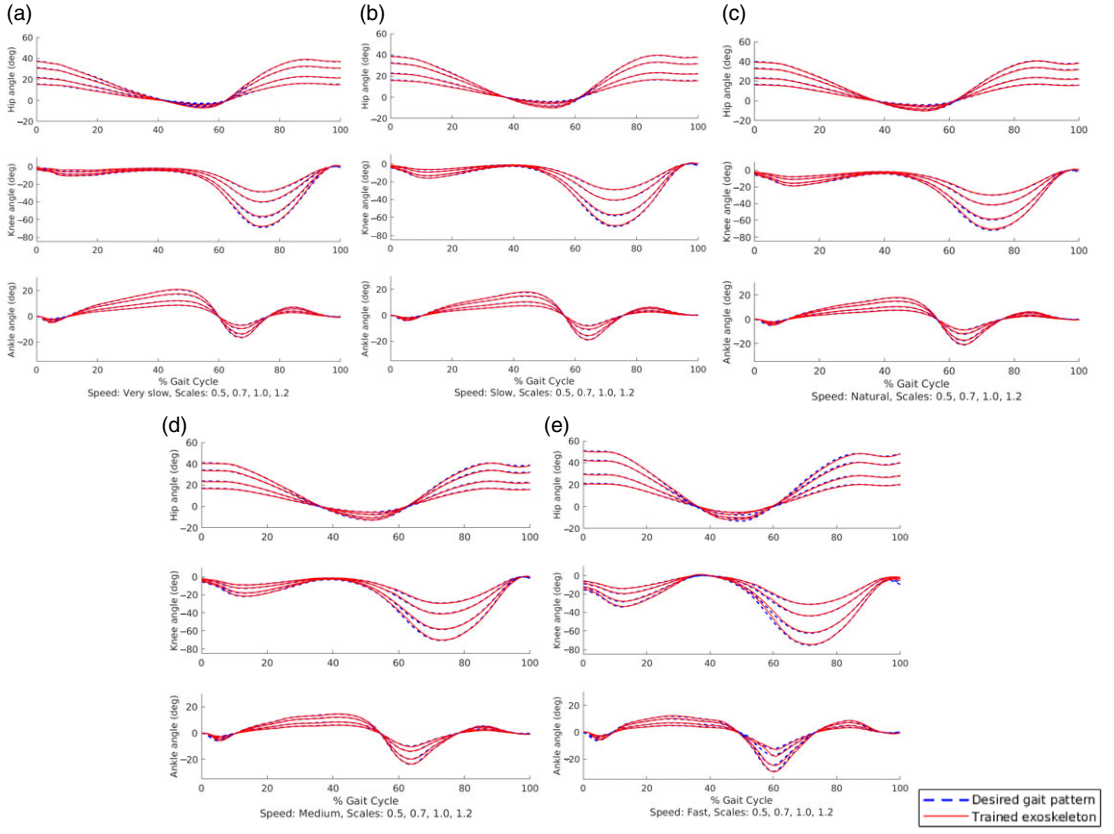

Figure 7. *Stage 1 mean desired gait patterns and trained exoskeleton gait patterns for the hip, knee, and ankle joint.*

https://doi.org/10.1017/S0263574721001600 Published online by Cambridge University Press

Table II. *Stage 1 gait pattern error.*

Joint	Mean absolute error (degrees)	Standard deviation (degrees)
Hip	0.36	0.14
Knee	0.47	0.17
Ankle	0.23	0.097

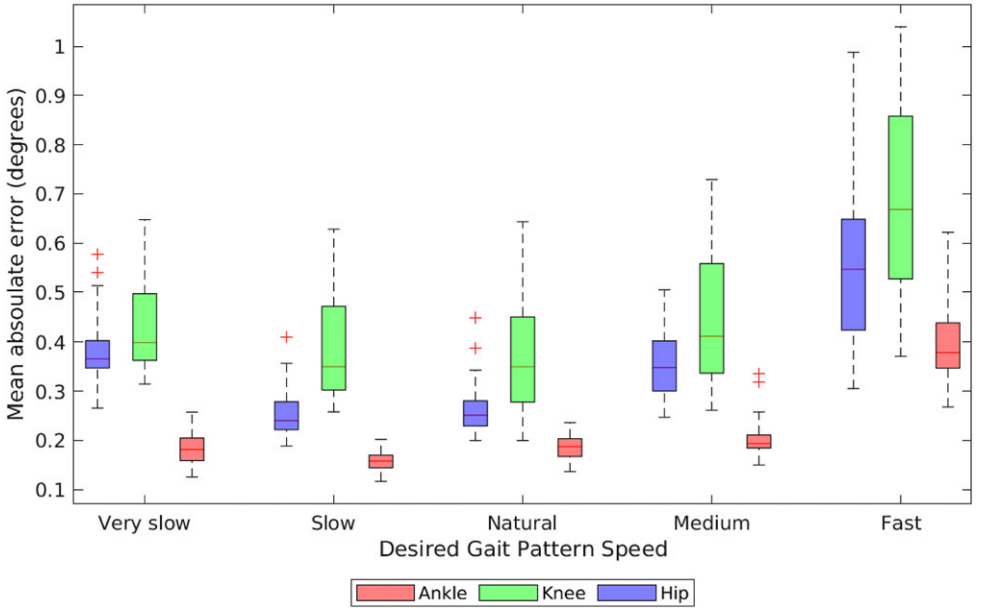

Figure 8. *Bloxplot for Stage 1 mean absolute error of each desired gait pattern speed for each hip, knee, and ankle joints.*

6.1. DRL training

During training, a desired gait pattern from the dataset in ref. [78], a baseline gait pattern from ref. [84], and a scaling factor for each were randomly chosen at each episode. The number of time steps and, thus, duration of each episode was based on the speed of the selected desired gait pattern. Training was completed for 2 million time steps, corresponding to over 10,000 episodes, on an Intel Core i7-8700 CPU, with each episode consisting of between 147 and 247 time steps. The reward function parameters and weights were defined as $\Sigma = 0.3$, $\mu = 0$, $w_F = 5$, $w_G = 1$, and $w_h, w_k, w_a = 1$. The training and DDPG algorithm hyperparameters are $\gamma = 0.99$, $\tau = 0.001$, and a learning rate of 0.001 as adapted from refs. [41, 63].

As can be seen in Fig. 6, the average reward per episode converged after approximately 1000 episodes, where a maximum average reward of approximately -7 was eventually reached. Prior to convergence, the controller learned a general torque pattern to follow the desired gait patterns. The reward curve fluctuates due to continual exploration by the agent while it is learning to follow the desired torque pattern. These fluctuations reduce from approximately 40 at 250 episodes to approximately 5 after 1000 episodes and remain within this later range for the duration of training. The number of steps required to control the baseline gait to reach the desired gait at each episode was equal to the episode length, which ranged from 147 to 247 steps. After convergence, the controller was able to follow the various gait patterns, and the remainder of training was used to find optimal actions for increasing the accuracy of tracking the desired gait patterns.

https://doi.org/10.1017/S0263574721001600 Published online by Cambridge University Press

Table III. *Stage 1 gait pattern error for each gait speed.*

	Mean absolute error (degrees)				
Joint \ Speed	**Very slow**	**Slow**	**Natural**	**Medium**	**Fast**
Hip	0.38	0.25	0.26	0.36	0.55
Knee	0.43	0.39	0.37	0.45	0.69
Ankle	0.18	0.16	0.19	0.19	0.40

Table IV. *Stage 1 error for each baseline subject.*

	Mean absolute error (degrees)					
Joint \ Subject	**S98**	**S108**	**S110**	**S128**	**S129**	**S136**
Hip	0.35	0.34	0.41	0.33	0.40	0.35
Knee	0.46	0.42	0.51	0.48	0.47	0.47
Ankle	0.21	0.20	0.23	0.22	0.25	0.24

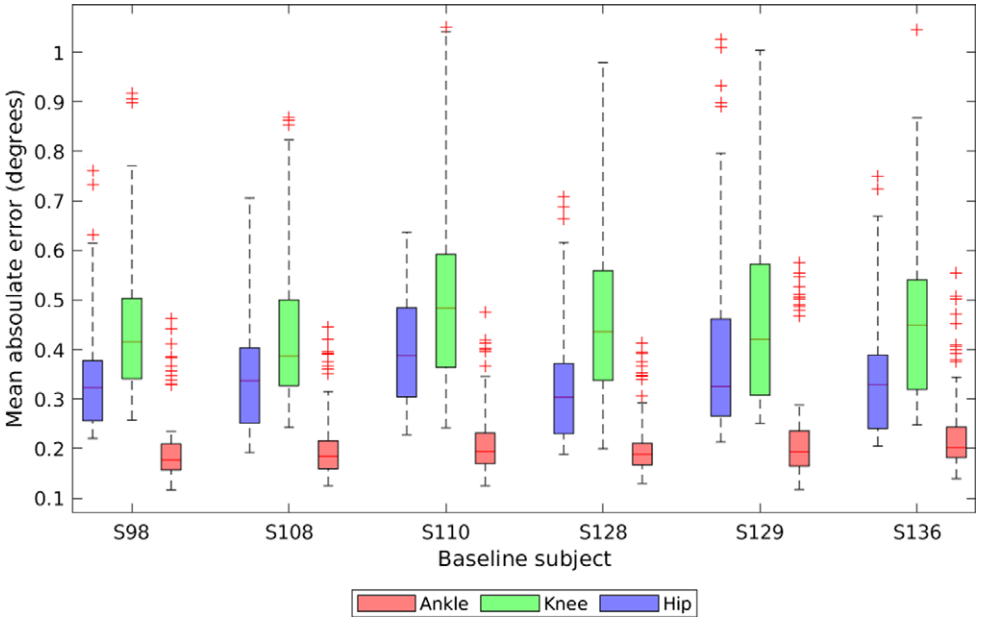

Figure 9. *Stage 1 mean absolute error for each baseline gait pattern for hip, knee, and ankle joints.*

6.2. Simulation runs: DRL testing

We conducted testing in two stages: (1) Stage 1: testing on desired gait patterns from the training, and (2) Stage 2: testing on a set of unseen desired gait patterns.

In Stage 1, the agent was tested for 400 episodes, corresponding to 400 gait cycles. Each desired gait pattern at each scale factor between 0.5 and 1.2 was tested 10 times. The average of 10 episodes is plotted for each gait pattern and scale factor and is presented in the results. In each episode, the baseline gait pattern from ref. [84] and its scale factor were randomly chosen.

In Stage 2, *five unseen* walking gait patterns were obtained from additional gait information from healthy adult subjects [85]. These gait patterns were not used in training. In this dataset, gait patterns with corresponding joint angles for one gait cycle from heel-strike to heel-strike were provided. Each

https://doi.org/10.1017/S0263574721001600 Published online by Cambridge University Press

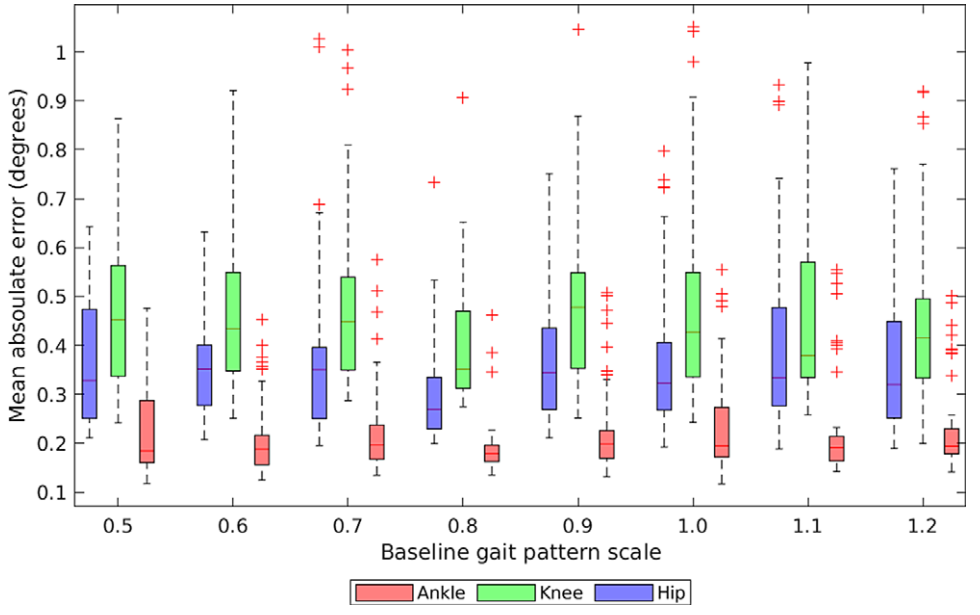

Figure 10. *Stage 1 mean absolute error of each baseline gait pattern scale factor for each hip, knee, and ankle joint.*

unseen gait pattern was set to a gait pattern speed as used in Stage 1, from very slow to fast. These gait patterns were then tested at each scale factor between 0.5 and 1.2 for 10 episodes each, corresponding to 400 gait cycles in total. The baseline gait patterns and scales, as used in stage 1, were randomly selected at each episode.

6.3. Simulation results

(1) Stage 1:

Figure 7 provides the tracking performance for the desired gait patterns at various scaling factors (0.5–1.2) for all the desired gait pattern speeds (very slow to fast). As can be seen in the figure, the desired gait patterns were closely followed by the exoskeleton within the full range of scaling, including at the extremities of 0.5 and 1.2. This shows the ability of the exoskeleton to reach the various gait pattern goals that would be required as a post-stroke individual recovers their range of motion and function.

Joint Errors: From these plots, it can be seen that the desired gait patterns were closely tracked by the exoskeleton, where the full range of motion of each desired gait pattern scale and speed was reached for all joints. This corresponds to a maximum range of motion of 64.1°, 75°, and 41° for the hip, knee, and ankle, respectively. Furthermore, it demonstrates that the gait pattern goals desired in physiotherapy, such as increased stride length and increased flexion in the knee and ankle, can be accurately reached by the trained exoskeleton. The mean absolute error for Stage 1 is also presented in Table II. The mean absolute error and standard deviations for each joint is determined from the mean absolute error of all episodes of testing. The overall mean absolute error ranged from 0.23 degrees for the ankle to 0.47 degrees for the knee. In comparison, overall mean error was found to be lower than the model-based approach presented in ref. [45] with an average of 0.74 degrees, and the mean error found in ref. [50] (2.87–4.5 degrees) for a DRL-based lower limb exoskeleton.

Gait Speed: The detailed boxplot for the mean absolute error for each gait pattern speed is presented in Fig. 8. The overall mean absolute error for the gait pattern speeds, as seen in Table III, ranged between 0.38, 0.43, and 0.18 degrees, for the hip, knee, and ankle joints for very slow to 0.55, 0.69, and 0.4

https://doi.org/10.1017/S0263574721001600 Published online by Cambridge University Press

Table V. *Stage 2 gait pattern error.*

Joint	Mean absolute error (degrees)	Standard deviation (degrees)
Hip	0.44	0.12
Knee	0.53	0.15
Ankle	0.37	0.14

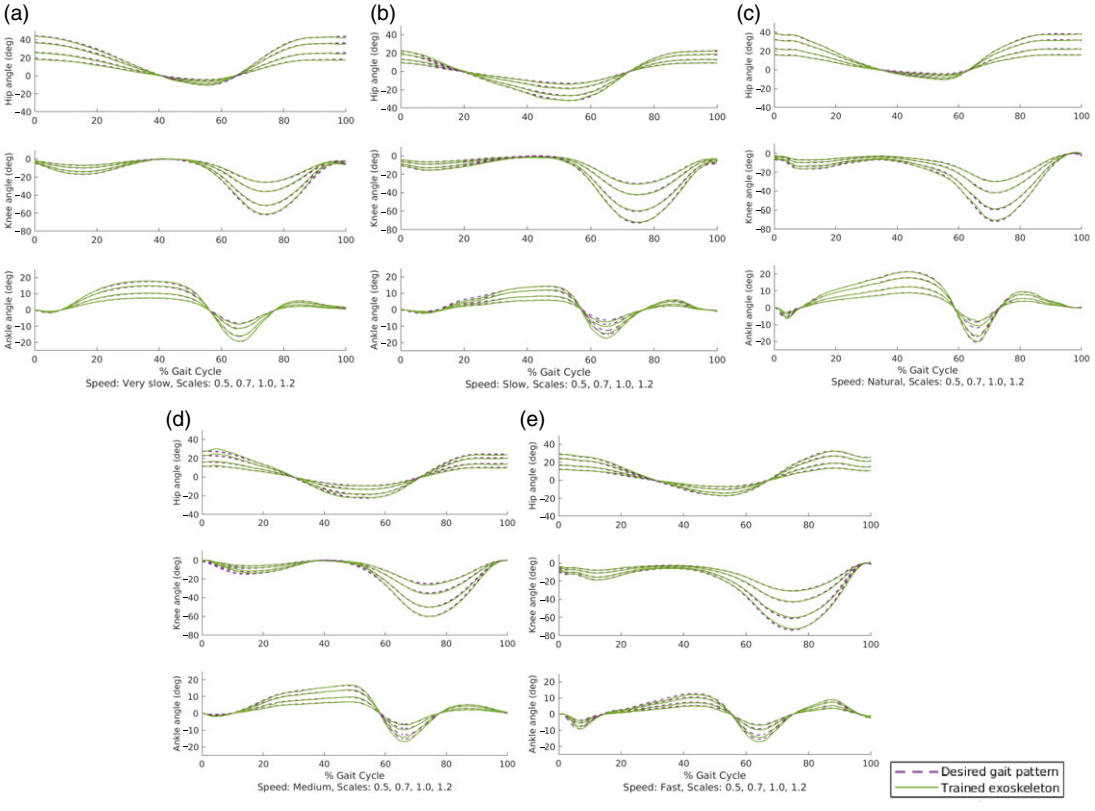

Figure 11. *Stage 2 desired gait patterns and trained exoskeleton gait patterns for the hip, knee, and ankle joint.*

degrees for fast, respectively. The low error range between the desired gait pattern speeds demonstrates the ability of this controller to be used at various walking speeds, which is advantageous for different post-stroke individuals at varying stages of functional recovery. The overall error is low for all desired gait pattern speeds but increases for the fastest gait speed. This is due in part to the larger range of motion of the fastest gait pattern as seen in Fig. 3, which introduces additional error as a larger range of joint angles must be reached in a shorter amount of time. However, the desired gait patterns were still able to be tracked at all speeds as shown in Fig. 7.

Baseline Gait: In Table IV, the mean absolute error across all tested baseline gait patterns for each joint can be seen, for each post-stroke subject (S98–S136) in ref. [84]. This demonstrates the controller's accuracy between varying exoskeleton users. A corresponding boxplot is shown in Fig. 9 of the various baseline gait patterns from the different subjects and in Fig. 10 of the baseline gait pattern scaling factors. Although it is evident that the overall mean error varied between baseline subjects and scales and was slightly higher for the subjects and joints with a larger range of motion, such as the knee joint for subject S110, the range of error was similar between the varying baseline gait patterns and

https://doi.org/10.1017/S0263574721001600 Published online by Cambridge University Press

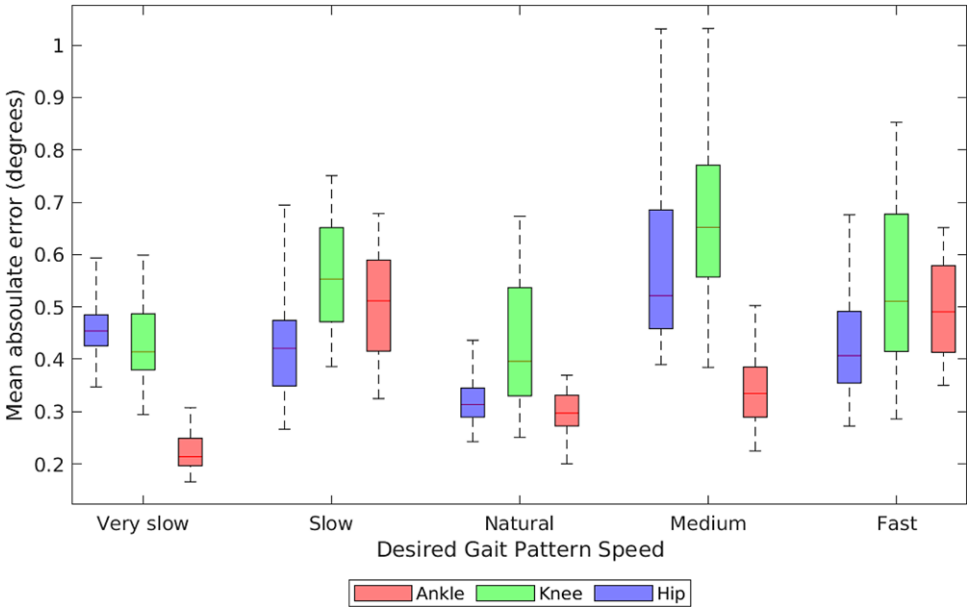

Figure 12. *Bloxplot for Stage 2 mean absolute error of each desired gait pattern speed for each hip, knee, and ankle joint.*

scale factors. Furthermore, the mean absolute error found for each baseline subject seen in Table IV was low overall, with error ranging between 0.35–0.41 degrees in the hip, 0.52–0.51 degrees in the knee, and 0.20–0.25 degrees in the ankle, for the different users. Therefore, the controller was able to learn multiple desired gait patterns with varying baseline gait pattern inputs, while maintaining low tracking error with different exoskeleton users' individual level of function, exoskeleton interaction, and perturbations.

(2) Stage 2:
In Stage 2, joint angle patterns from the *unseen* testing set were verified.

Joint Errors: The mean absolute error for the hip, knee, and ankle is presented in Table V. As expected, the overall mean absolute error was higher than in Stage 1 by 0.08, 0.06, and 0.14 degrees for the hip, knee, and ankle, respectively. However, the error was still low for all joints for these unseen gait patterns, with an average error ranging from 0.37 degrees for the ankle to 0.53 for the knee. Figure 11 presents the tracked joint angle plots of the unseen desired gait patterns, with various desired gait patterns scales from 0.5 to 1.2 for each speed. In the figure, the gait patterns were closely followed by the exoskeleton throughout the range of scale factors at each gait speed, although these gait patterns were unseen by the controller in training.

Gait Speed: Figure 12 presents the boxplots for the mean absolute error for each gait pattern speed in Stage 2. The overall mean absolute error for each speed, as seen in Table VI, ranged between 0.22 degrees in the ankle for the very slow speed and 0.67 degrees in the knee for the medium speed.

Baseline Gait: Figure 13 presents the boxplot of the mean absolute error from the various baseline gait patterns tested in Stage 2, for the different subjects (S98–S136) in ref. [84]. The corresponding mean absolute errors are also shown in Table VII. The mean absolute error ranged between 0.39–0.48 degrees in the hip, 0.50–0.57 degrees in the knee, and 0.36–0.40 degrees in the ankle for the different exoskeleton users. This demonstrates the DRL controller's ability to be used with multiple post-stroke individuals with varying levels of movement functionality, while being able to control completely unseen desired gait patterns.

https://doi.org/10.1017/S0263574721001600 Published online by Cambridge University Press

Table VI. *Stage 2 gait pattern error for each gait speed.*

	Mean absolute error (degrees)				
Joint \ Speed	**Very slow**	**Slow**	**Natural**	**Medium**	**Fast**
Hip	0.46	0.43	0.32	0.57	0.43
Knee	0.44	0.56	0.43	0.67	0.54
Ankle	0.22	0.51	0.30	0.34	0.50

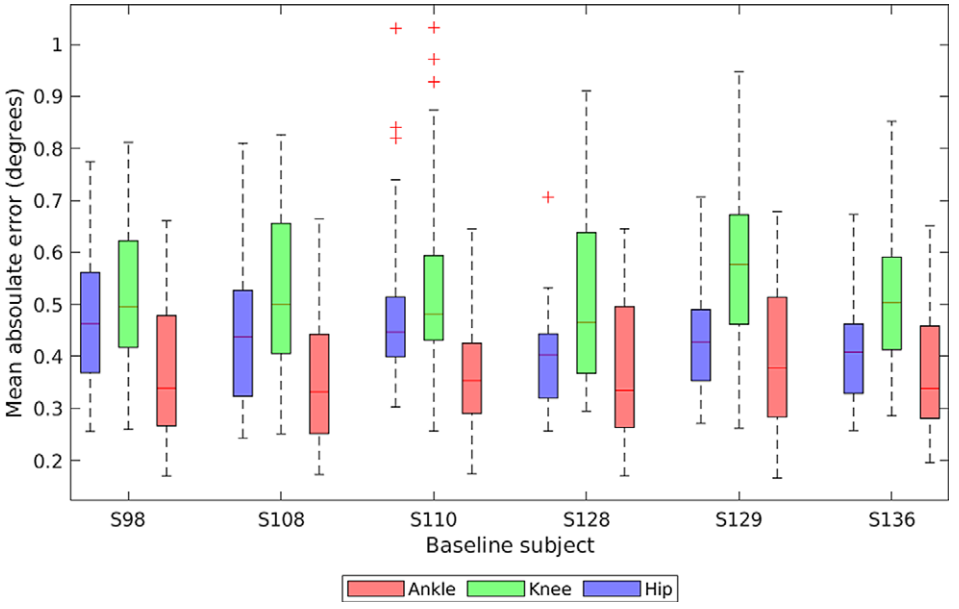

Figure 13. *Bloxplot for Stage 2 mean absolute error of each baseline gait pattern subject, as named in ref. [84], for each hip, knee, and ankle joint.*

We provide an overall general comparison of our mean absolute errors to those obtained from other exoskeleton control methods. It should be noted that these results were obtained using different exoskeleton systems with fewer degrees of freedom than the system employed in this work (i.e., three active DOFs or less). In general, our approach has better error than several existing adaptive model-based controllers, where their tracking error ranged from 1° to 5° [37, 46]. Our mean absolute error was also lower than those reported in ref. [75], where a PID controller was applied to an orthotic attached to an OpenSim human musculoskeletal model with an average tracking error of 3.8°. In comparison to other DRL-based exoskeleton control methods, which had mean absolute errors ranging from 0.12° to 0.91° [48], 2.86° to 4.6° [50], and 0.41° to 1.31° [86], respectively, our approach has comparable or better error (all our mean absolute errors were less than 0.67°), while uniquely allowing for model-free and accurate control for both seen and unseen gait patterns. This supports the ability of our DRL approach to learn and track additional desired gait patterns a physiotherapist may wish a subject to perform in a clinical session, while being robust to multiple post-stroke individuals' baseline gait patterns and individual level of functionality.

6.4. Simulation to real-world

In order to further adapt this control scheme for use on hardware, sim-to-real techniques and transfer learning approaches would need to be employed to account for the differences in the simulated and

https://doi.org/10.1017/S0263574721001600 Published online by Cambridge University Press

Table VII. *Stage 2 error for each baseline subject.*

	Mean absolute error (degrees)					
Joint \ Subject	**S98**	**S108**	**S110**	**S128**	**S129**	**S136**
Hip	0.48	0.45	0.47	0.39	0.44	0.40
Knee	0.52	0.52	0.52	0.50	0.57	0.52
Ankle	0.37	0.36	0.36	0.37	0.40	0.37

physical actuators, exoskeleton linkages, and human model. The research topics of sim-to-real and transfer learning have their own open challenges, and there are dedicated papers solely focusing on effectively transferring simulations to real-world robotic experiments without additional training, for example, refs. [90]–[94]. Current literature on this subject suggests including parameter, measurement, and physical characteristic noise into the model to provide robustness in training, in addition to the existing action space noise [30], to allow for an easier transfer to hardware. The proposed DRL controller could also be adjusted to optimize the network architecture and to fine-tune the parameters of the exploration function, in order to increase sample efficiency and potentially reduce training time. Moreover, add-ons to the OpenSim platform have been developed such as actuator classes that emulate the characteristics of DC motors [87], which would further assist in adding additional realism to the DRL simulation. Additional musculoskeletal model characteristics, for example, muscle and ligament models of greater anatomical complexity, as well as supplemental joint and muscle mechanic models, including muscle spasticity characteristics that are often found in post-stroke individuals [88], could be implemented to improve the fidelity of the simulated 3D musculoskeletal model. The inclusion of such models would result in baseline gait patterns that are more realistically simulated. We will consider the design and application of such sim-to-real techniques as part of our future research work. Moreover, in order to further validate the effectiveness of our developed control strategy, future work will include conducting a detailed comparative study of multiple control strategies, including learning-based and adaptive model-based methods, on a physical exoskeleton platform.

7. Conclusions

In this paper, a novel model-free, end-to-end DRL method for control of gait patterns on an exoskeleton for overground gait training was developed capable of accounting for the varying movement functionality of post-stroke individuals and different desired gait patterns for multiple users. The DDPG algorithm was adapted to learn hip, knee, and ankle exoskeleton actuator control torque actions in an end-to-end manner, directly from observations of joint parameters for multiple users. The DRL controller was trained and then tested in a simulated 3D physics environment, with clinical data from both healthy and post-stroke individuals. Results demonstrate that the controller was able to accurately follow the desired gait patterns both seen and unseen in training. Our controller can provide gait recovery needed in physiotherapy without requiring a predefined dynamics model and is robust to varying post-stroke individuals' baseline gait patterns and their interactions and perturbations as they progress through therapy. It can also be adapted for any lower-body exoskeleton model by adjusting the kinematic and dynamic system parameters in the simulation environment. Future work includes implementing the DRL controller on exoskeleton hardware and performing clinical user studies. In order to implement the DRL controller on a physical exoskeleton, we will explore the use of sim-to-real techniques to account for the differences that exist in the simulated and physical actuators, exoskeleton linkages, and human model. We will also design and integrate the necessary sensory, control, and communication hardware.

Funding. This research was supported by the EMHSeed Fund, the Natural Sciences and Engineering Research Council of Canada, and the Canada Research Chairs Program (CRC).

https://doi.org/10.1017/S0263574721001600 Published online by Cambridge University Press

Acknowledgments. The authors would like to thank Kara Patterson, Julie Vaughan-Graham, and Dina Brooks from the Department of Physical Therapy at the University of Toronto for their input in defining rehabilitation goals and outcomes.

Conflicts of interest. None of the authors report any potential conflict of interest in respect of this research.

References

[1] H. Krueger, J. Koot, R. E. Hall, C. O'Callaghan, M. Bayley and D. Corbett, "Prevalence of individuals experiencing the effects of stroke in Canada: Trends and projections," *Stroke* **46**(8), 2226–2231 (2015).
[2] K. Lo, M. Stephenson and C. Lockwood, "Effectiveness of robotic assisted rehabilitation for mobility and functional ability in adult stroke patients: A systematic review protocol," *JBI Database Syst. Rev. Implementation Rep.* **15**(12), 3049–3091 (2017).
[3] B. S. Rupal, S. Rafique, A. Singla, E. Singla, M. Isaksson and G. S. Virk, "Lower-limb exoskeletons: Research trends and regulatory guidelines in medical and non-medical applications," *Int. J. Adv. Robot. Syst*. **14**(6), 1–27 (2017).
[4] B. Hobbs and P. Artemiadis, "A review of robot-assisted lower-limb stroke therapy: Unexplored paths and future directions in gait rehabilitation," *Front. Neurorobot*. **14**, Article 19 (1–16) (2020).
[5] C. Selves, G. Stoquart and T. Lejeune, "Gait rehabilitation after stroke: Review of the evidence of predictors, clinical outcomes and timing for interventions," *Acta Neurol. Belg.* **120**(4), 783–790 (2020).
[6] D. R. Louie and J. J. Eng, "Powered robotic exoskeletons in post-stroke rehabilitation of gait: A scoping review," *J. Neuroeng. Rehabil.* **13**(1), 53 (2016).
[7] S. Federici, F. Meloni, M. Bracalenti and M. L. De Filippis, "The effectiveness of powered, active lower limb exoskeletons in neurorehabilitation: A systematic review," *NeuroRehabilitation* **37**(3), 321–340 (2015).
[8] X. Wu, D.-X. Liu, M. Liu, C. Chen and H. Guo, "Individualized gait pattern generation for sharing lower limb exoskeleton robot," *IEEE Trans. Autom. Sci. Eng*., **15**(4), 1459–1470 (2018).
[9] R. Mendoza-Crespo, D. Torricelli, J. C. Huegel, J. L. Gordillo, J. L. Pons and R. Soto, "An adaptable human-like gait pattern generator derived from a lower limb exoskeleton," *Front. Robot. AI* **6**, Article 36 (1–14) (2019).
[10] B. Chen et al., "Recent developments and challenges of lower extremity exoskeletons," *J. Orthop. Transl.* **5**, 26–37 (2016).
[11] A. J. Young and D. P. Ferris, "State of the art and future directions for lower limb robotic exoskeletons," *IEEE Trans. Neural Syst. Rehabil. Eng*. **25**(2), 171–182 (2017).
[12] M. Bortole et al., "The H2 robotic exoskeleton for gait rehabilitation after stroke: Early findings from a clinical study Wearable robotics in clinical testing," *J. Neuroeng. Rehabil.* **12**(1), 54 (2015).
[13] G. Lv, H. Zhu and R. D. Gregg, "On the design and control of highly backdrivable lower-limb exoskeletons: A discussion of past and ongoing work," *IEEE Control Syst.* **38**(6), 88–113 (2018).
[14] A. McDaid, K. Kora, S. Xie, J. Lutz and M. Battley, "Human-Inspired Robotic Exoskeleton (HuREx) for Lower Limb Rehabilitation," 2013 *IEEE International Conference on Mechatronics and Automation, IEEE ICMA 2013* (2013) pp. 19–24.
[15] J. Zhang et al., "Human-in-the-loop optimization of exoskeleton assistance during walking," *Science* **356**(6344), 1280–1284 (2017).
[16] G. Wu, C. Wang, X. Wu, Z. Wang, Y. Ma and T. Zhang, "Gait Phase Prediction for Lower Limb Exoskeleton Robots," 2016 IEEE International Conference on Information and Automation (2016) pp. 19–24.
[17] T. P. Luu, K. H. Low, X. Qu, H. B. Lim and K. H. Hoon, "An individual-specific gait pattern prediction model based on generalized regression neural networks," *Gait Posture* **39**(1), 443–448 (2014).
[18] F. Horst, S. Lapuschkin, W. Samek, K.-R. MÜller and W. I. SchÖllhorn, "Explaining the unique nature of individual gait patterns with deep learning," *Sci. Rep.* **9**(1), 2391 (2019).
[19] H. B. Lim, T. P. Luu, K. H. Hoon and K. H. Low, "Natural Gait Parameters Prediction for Gait Rehabilitation via Artificial Neural Network," 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (2010) pp. 5398–5403.
[20] M. R. Tucker et al., "Control strategies for active lower extremity prosthetics and orthotics: A review," *J. Neuroeng. Rehabil.* **12**(1), 1–29 (2015).
[21] J. C. Moreno, J. Figueiredo and J. L. Pons, "Chapter 7 - Exoskeletons for lower-limb rehabilitation," **In**: Rehabilitation Robotics (R. Colombo and V. Sanguineti, eds.) (Elsevier, Imprint: Academic Press), ISBN: 978-0-12-811995-2. Available: https://doi.org/10.1016/C2016-0-02285-4.
[22] M. Lotze, C. Braun, N. Birbaumer, S. Anders and L. G. Cohen, "Motor learning elicited by voluntary drive," *Brain* **126**(4), 866–872 (2003).
[23] T. Yan, M. Cempini, M. Oddo and N. Vitiello, "Review of assistive strategies in powered lower-limb orthoses and exoskeletons," *Rob. Auton. Syst.* **64**, 120–136 (2015).
[24] B. Brahmi, M. Saad, C. O. Luna, P. S. Archambault and M. H. Rahman, "Passive and active rehabilitation control of human upper-limb exoskeleton robot with dynamic uncertainties," *Robotica* **36**(11), 1757–1779 (2018).
[25] Y. Long, Z. J. Du, W. D. Wang and W. Dong, "Robust sliding mode control based on GA optimization and CMAC compensation for lower limb exoskeleton," Appl. Bionics Biomech. **2016**, Article 5017381 (1–13) (2016).
[26] F. Sado, H. J. Yap, R. Ariffin, R. A. R. Ghazilla and N. Ahmad, "Exoskeleton robot control for synchronous walking assistance in repetitive manual handling works based on dual unscented Kalman filter," *PLoS One* **13**(7), 1–36 (2018).
[27] D. Sanz-Merodio, M. Cestari, J. C. Arevalo, X. A. Carrillo and E. Garcia, "Generation and control of adaptive gaits in lower-limb exoskeletons for motion assistance," *Adv. Robot.* **28**(5), 329–338 (2014).

[28] S. K. Banala, S. K. Agrawal, S. H. Kim and J. P. Scholz, "Novel gait adaptation and neuromotor training results using an active leg exoskeleton," *IEEE/ASME Trans. Mechatron.* **15**(2), 216–225 (2010).
[29] X. Wang, X. Li, J. Wang, X. Fang and X. Zhu, "Data-driven model-free adaptive sliding mode control for the multi degree-of-freedom robotic exoskeleton," *Inf. Sci. (Ny)*. **327**, 246–257 (2016).
[30] J. Hwangbo et al., "Learning agile and dynamic motor skills for legged robots," *Sci. Robot.* **4**(26), 1–13 (2019).
[31] A. Rajeswaran et al., "Learning Complex Dexterous Manipulation with Deep Reinforcement Learning and Demonstrations," ArXiv1709.10087 (2018).
[32] Z. Qu et al., "Research on Fuzzy Adaptive Impedance Control of Lower Extremity Exoskeleton," *Proceedings of 2019 IEEE International Conference on Mechatronics and Automation* (2019) pp. 939–944.
[33] G. Serrancoli et al., "Subject-exoskeleton contact model calibration leads to accurate interaction force predictions," *IEEE Trans. Neural Syst. Rehabil. Eng*. **27**(8), 1597–1605 (2019).
[34] Y. Yuan, Z. Li, T. Zhao and D. Gan, "DMP-based motion generation for a walking exoskeleton robot using reinforcement learning," *IEEE Trans. Ind. Electron.*(2019).
[35] R. Huang, H. Cheng, J. Qiu and J. Zhang, "Learning physical human-robot interaction with coupled cooperative primitives for a lower exoskeleton," *IEEE Trans. Autom. Sci. Eng*. **16**(4), 1–9 (2019).
[36] V. Pong, S. Gu, M. Dalal and S. Levine, "Temporal Difference Models: Model-Free Deep RL for Model-based Control," *6th International Conference on Learning Representations. ICLR 2018 - Conference Track Proceedings* (2018) pp. 1–14.
[37] G. Bingjing, H. Jianhai, L. Xiangpan and Y. Lin, "Human–robot interactive control based on reinforcement learning for gait rehabilitation training robot," *Int. J. Adv. Robot. Syst.* **16**(2), 1–16 (2019).
[38] Y. Zhang, S. Li, K. J. Nolan and D. Zanotto, "Adaptive Assist-as-needed Control Based on Actor-Critic Reinforcement Learning," IEEE International Conference on Intelligent Robots and Systems (2019) pp. 4066–4071.
[39] M. Hamaya, T. Matsubara, T. Noda, T. Teramae and J. Morimoto, "Learning Assistive Strategies from a Few User-Robot Interactions: Model-based Reinforcement Learning Approach," *Proceedings - IEEE International Conference on Robotics and Automation* (2016) pp. 3346–3351.
[40] S. G. Khan, M. Tufail, S. H. Shah and I. Ullah, "Reinforcement learning based compliance control of a robotic walk assist device," *Adv. Robot.*, **33**(24), 1281–1292 (2019).
[41] T. P. Lillicrap et al., "Continuous Control with Deep Reinforcement Learning," *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings* (2016).
[42] L. Rose, M. C. F. Bazzocchi and G. Nejat, "End-to-End Deep Reinforcement Learning for Exoskeleton Control," *Conference Proceedings - IEEE International Conference on Systems, Man and Cybernetics* (2020).
[43] S. Maggioni, N. Reinert, L. LÜnenburger and A. Melendez-Calderon, "An adaptive and hybrid end-point/joint impedance controller for lower limb exoskeletons," *Front. Robot. AI* **5**, 104 (2018).
[44] L. Wang, E. H. F. Van Asseldonk and H. Van Der Kooij, "Model Predictive Control-based Gait Pattern Generation for Wearable Exoskeletons," IEEE International Conference on Rehabilitation Robotics (2011) pp. 1–6.
[45] O. Harib et al., "Feedback Control of an Exoskeleton for Paraplegics: Toward Robustly Stable Hands-free Dynamic Walking," ArXiv1802.08322 (2018).
[46] D. Lo Castro, C. H. Zhong, F. Braghin and W. H. Liao, "Lower Limb Exoskeleton Control via Linear Quadratic Regulator and Disturbance Observer," *2018 IEEE International Conference on Robotics and Biomimetics, ROBIO 2018* (2018) pp. 1743–1748.
[47] V. C. V. Kumar, S. Ha, G. Sawicki and C. K. Liu, "Learning a Control Policy for Fall Prevention on an Assistive Walking Device," ArXiv1909.10488 (2019).
[48] D. Di Febbo et al., "Reinforcement Learning Control of Functional Electrical Stimulation of the upper limb : a feasibility study," Annual Conference of the International Functional Electrical Stimulation Society (2018) pp. 111–114.
[49] M. Lyu, W. H. Chen, X. Ding and J. Wang, "Knee exoskeleton enhanced with artificial intelligence to provide assistance-as-needed," *Rev. Sci. Instrum.* **90**(9), 094101-1–094101-13 (2019).
[50] J. Xu et al., "A multi-channel reinforcement learning framework for robotic mirror therapy," *IEEE Robot. Autom. Lett.* **5**(4), 5385–5392 (2020).
[51] X. Zhang, H. Wang, Y. Tian, L. Peyrodie and X. Wang, "Model-free based neural network control with time-delay estimation for lower extremity exoskeleton," *Neurocomputing* **272**, 178–188 (2018). Available: https://doi.org/10.1016/j.neucom.2017.06.055.
[52] P. Yang, J. Sun, J. Wang, G. Zhang, and Y. Zhang, "Model-Free Based Back-Stepping Sliding Mode Control for Wearable Exoskeletons," 25th IEEE International Conference on Automation and Computing (2019).
[53] J. Zhang, C. C. Cheah and S. H. Collins, "Chapter 5 - Torque control in legged locomotion," In: Bioinspired Legged Locomotion: Models, Concepts, Control and Applications (Elsevier, Imprint: Butterworth-Heinemann, 2017) pp. 347–400, ISBN: 978-0-12-803766-9.
[54] D. Torricelli et al., "A subject-specific kinematic model to predict human motion in exoskeleton-assisted gait," *Front. Neurorobot.* **12**, Article 18 (1–11) (2018).
[55] H. van Hasselt, "Reinforcement Learning in Continuous State and Action Spaces," In: *Adaptation, Learning, and Optimization*, vol. 12 (2012) pp. 207–251.
[56] X. Bin Peng and M. van de Panne, "Learning Locomotion Skills Using Deep RL: Does the Choice of Action Space Matter?," *Proceedings - SCA 2017 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, vol. 1 (2017).
[57] S. Lee, M. Park, K. Lee and J. Lee, "Scalable muscle-actuated human simulation and control," *ACM Trans. Graph.* **38**(4) (2019).

https://doi.org/10.1017/S0263574721001600 Published online by Cambridge University Press

[58] X. Bin Peng, G. Berseth, K. Yin and M. Van De Panne, "DeepLoco: Dynamic locomotion skills using hierarchical deep reinforcement learning," *ACM Trans. Graph.* **36**(4) (2017).

[59] A. S. Anand, G. Zhao, H. Roth and A. Seyfarth, "A Deep Reinforcement Learning Based Approach Towards Generating Human Walking Behavior with a Neuromuscular Model," *2019 IEEE-RAS 19th International Conference on Humanoid Robots* (2020) pp. 537–543.

[60] C. R. Gil, H. Calvo and H. Sossa, "Learning an efficient gait cycle of a biped robot based on reinforcement learning and artificial neural networks," *Appl. Sci.* **9**(3), Article 502 (1–24) (2019).

[61] J. GarcÍa and D. Shafie, "Teaching a humanoid robot to walk faster through Safe Reinforcement Learning," *Eng. Appl. Artif. Intell*. **88**, Article 103360 (1–10) (2020). Available: https://doi.org/10.1016/j.engappai.2019.103360.

[62] C. Liu, A. Lonsberry, M. Nandor, M. Audu, A. Lonsberry and R. Quinn, "Implementation of deep deterministic policy gradients for controlling dynamic bipedal walking," *Biomimetics* **4**(1), 28 (2019).

[63] M. Plappert, "keras-rl," *GitHub*, 2016. [Online]. Available: https://github.com/keras-rl/keras-rl. [Accessed: 24-Apr-2020].

[64] Ł. KidziŃski et al., "Learning to Run Challenge: Synthesizing Physiologically Accurate Motion Using Deep Reinforcement Learning," In: *The NIPS '17 Competition: Building Intelligent Systems, The Springer Series on Challenges in Machine Learning* (S. Escalera and M. Weimer, eds.) (Springer, Cham, 2018) pp. 101–120.

[65] V. Mnih et al., "Playing Atari with Deep Reinforcement Learning," (2013), pp. 1–9. Available: https://arxiv.org/pdf/1312.5602.pdf?source=post_page

[66] G. E. Uhlenbeck and L. S. Ornstein, "On the theory of the Brownian motion," *Phys. Rev.* **36**(5), 823–841 (1930).

[67] S. Fujimoto, H. Van Hoof and D. Meger, "Addressing Function Approximation Error in Actor-Critic Methods," *35th International Conference on Machine Learning, ICML 2018*, vol. 4 (2018) pp. 2587–2601.

[68] D. P. Kingma and J. L. Ba, "Adam: A Method for Stochastic Optimization," *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings* (2015).

[69] A. Seth et al., "OpenSim: Simulating musculoskeletal dynamics and neuromuscular control to study human and animal movement," *PLOS Comput. Biol.* **14**(7), e1006223 (2018).

[70] D. Coll Pujals, *Simulation of the Assistance of an Exoskeleton on Lower Limbs Joints Using Opensim* (Polytechnic University of Catalonia, 2017).

[71] L. Rose, M. C. F. Bazzocchi, C. de Souza, J. Vaughan-Graham, K. Patterson and G. Nejat, "A Framework for Mapping and Controlling Exoskeleton Gait Patterns in both Simulation and Real -World," *Proceedings of the 2020 Design of Medical Devices Conference* (2020).

[72] K. H. Hunt and F. R. E. Crossley, "Coefficient of restitution interpreted as damping in vibroimpact," *J. Appl. Mech. Trans. ASME* **42**(2), 440–445 (1975).

[73] D. Thelen, A. Seth, F. C. Anderson and S. L. Delp, "OpenSim Models Gait 2392 and 2354 Documentation," *SimTK*. [Online]. Available: https://simtk-confluence.stanford.edu:8443/display/OpenSim/Gait+2392+and+2354+Models. [Accessed: 09-May-2020].

[74] D. G. Thelen, "Adjustment of muscle mechanics model parameters to simulate dynamic contractions in older adults," *J. Biomech. Eng.* **125**(1), 70–77 (2003).

[75] D. Mosconi, P. F. Nunes and A. A. G. Siqueira, "Modeling and control of an active knee orthosis using a computational model of the musculoskeletal system," *J. Mechatronics Eng.* **1**(3), 12 (2018).

[76] G. Brockman et al., "OpenAI Gym," ArXiv1606.01540 (2016).

[77] M. A. Sherman, A. Seth and S. L. Delp, "Simbody: Multibody dynamics for biomedical research," *Procedia IUTAM* **2**, 241–261 (2011). Available: https://doi.org/10.1016/j.piutam.2011.04.023.

[78] G. Bovi, M. Rabuffetti, P. Mazzoleni and M. Ferrarin, "A multiple-task gait analysis approach: Kinematic, kinetic and EMG reference data for healthy young and adult subjects," *Gait Posture* **33**(1), 6–13 (2011).

[79] F. Horst, S. Lapuschkin, W. Samek, K. R. MÜller and W. I. SchÖllhorn, "Explaining the unique nature of individual gait patterns with deep learning," *Sci. Rep.* **9**(1), Article 2391 (1–13) (2019).

[80] J. K. Moore, S. K. Hnat and A. J. van den Bogert, "An elaborate data set on human gait and the effect of mechanical perturbations," *PeerJ* **2015**(3), 1–21 (2015). Available: https://peerj.com/articles/918/#.

[81] W. Wang, J. Chen, Y. Ji, W. Jin, J. Liu and J. Zhang, "Evaluation of lower leg muscle activities during human walking assisted by an ankle exoskeleton," *IEEE Trans. Ind. Inf.* **16**(11), 7168–7176 (2020).

[82] N. D. Neckel, N. Blonien, D. Nichols and J. Hidler, "Abnormal joint torque patterns exhibited by chronic stroke subjects while walking with a prescribed physiological gait pattern," *J. Neuroeng. Rehabil.* **5**(1), 19 (2008).

[83] R. B. Huitema, A. L. Hof, T. Mulder, W. H. Brouwer, R. Dekker and K. Postema, "Functional recovery of gait and joint kinematics after right hemispheric stroke," *Arch. Phys. Med. Rehabil.* **85**(12), 1982–1988 (2004).

[84] B. A. Knarr, T. M. Kesar, D. S. Reisman, S. A. Binder-Macleod and J. S. Higginson, "Changes in the activation and function of the ankle plantar flexor muscles due to gait retraining in chronic stroke survivors.," *J. Neuroeng. Rehabil.* **10**, 12 (2013).

[85] T. Lencioni, I. Carpinella, M. Rabuffetti, A. Marzegan and M. Ferrarin, "Human kinematic, kinetic and EMG data during different walking and stair ascending and descending tasks," *Sci. Data* **6**(1), 1–10 (2019).

[86] D. Di Febbo et al., "Does Reinforcement Learning Outperform PID in the Control of FES-Induced Elbow Flex-Extension?," 2018 IEEE International Symposium on Medical Measurements and Applications Proceedings (2018) pp. 1–6.

[87] V. Q. Nguyen, A. K. LaPre, M. A. Price, B. R. Umberger and F. C. Sup, "Inclusion of actuator dynamics in simulations of assisted human movement," *Int. J. Numer. Methods Biomed. Eng.* **36**(5), 1–13 (2020). Available: https://doi.org/10.1002/cnm.3334.

[88] S. Li, G. E. Francisco and P. Zhou, "Post-stroke hemiplegic gait: New perspective and insights," *Front. Physiol.* **9**, Article 1021 (1–8) (2018). Available: https://doi.org/10.3389/fphys.2018.01021.

[89] C. Nwankpa, W. Ijomah, A. Gachagan and S. Marshall, "Activation functions: Comparison of trends in practice and research for deep learning," arXiv preprint arXiv:1811.03378 (2018).
[90] N. Liu, Y. Cai, T. Lu, R. Wang and S. Wang, "Real–sim–real transfer for real-world robot control policy learning with deep reinforcement learning," *Appl. Sci.* **10**(5), 1555 (2020).
[91] W. Yu, V. C. Kumar, G. Turk and C. K. Liu, "Sim-to-Real Transfer for Biped Locomotion" International Conference on Intelligent Robots and Systems (IROS) (2019).
[92] X. B. Peng, M. Andrychowicz, W. Zaremba and P. Abbeel, "Sim-to-Real Transfer Of Robotic Control with Dynamics Randomization," IEEE International Conference on Robotics and Automation (ICRA) (2018) pp. 3803–3810.
[93] W. Zhao, J. P. Queralta and T. Westerlund, "Sim-to-Real Transfer in Deep Reinforcement Learning for Robotics: A Survey," IEEE Symposium Series on Computational Intelligence (SSCI) (2020) pp. 737–744.
[94] R. C. Julian, E. Heiden, Z. He, H. Zhang, S. Schaal, J. J.Lim, G. S. Sukhatme and K. Hausman, "Scaling simulation-to-real transfer by learning a latent space of robot skills," *Int. J. Rob. Res.* **39**(10–11), 1259–1278 (2020).

Cite this article: L. Rose, M. C. F. Bazzocchi and G. Nejat (2022). "A model-free deep reinforcement learning approach for control of exoskeleton gait patterns", Robotica **40**, 2189–2214. https://doi.org/10.1017/S0263574721001600