Article

Inertia-Constrained Reinforcement Learning to Enhance Human Motor Control Modeling

Soroush Korivand [1,2,*], Nader Jalili [1,*] and Jiaqi Gong [2,*]

[1] The University of Alabama, Mechanical Engineering Department; Tuscaloosa, AL, USA
[2] The University of Alabama, Computer Science Department; Tuscaloosa, AL, USA
* Correspondence: skorivand@crimson.ua.edu, jiaqi.gong@ua.edu, njalili@ua.edu

Abstract: Locomotor impairment is a high-prevalent and significant source of disability and significantly impacts a large population's quality of life. Despite decades of research in human locomotion, the challenges of simulating human movement to study the features of musculoskeletal drivers and clinical conditions remain. Most recent efforts in utilizing reinforcement learning (RL) techniques are promising to simulate human locomotion and reveal musculoskeletal drives. However, these simulations often failed to mimic natural human locomotion because most reinforcement strategies have yet to consider any reference data regarding human movement. To address these challenges, in this study, we designed a reward function based on the trajectory optimization rewards (TOR), and bio-inspired rewards, which includes the rewards obtained from reference motion data captured by a single Intertial Moment Unit (IMU) sensor. The sensor was equipped on the participants' pelvis to capture reference motion data. Also, we adapted the reward function by leveraging previous research in walking simulation for TOR. The experimental results showed that the simulated agents with the modified reward function performed better in mimicking the collected IMU data from participants, which means the simulated human locomotion was more realistic. Also, as this bio-inspired defined cost, IMU data enhanced the agent's capacity to converge during the training process. As a result, the models' convergence is faster than those developed without reference motion data. Consequently, human locomotion can be simulated more quicker and in a broader range of environments with a better simulation performance.

Keywords: Reinforcement Learning, Locomotion Disorder, IMU Sensor, Musculoskeletal simulation

1. Introduction

An accurate model and simulation of human locomotion are highly desirable for many applications, such as identifying musculoskeletal features, assessing clinical conditions, and preventing aging and locomotor diseases. Although separated human muscles and limbs have been modeled accurately [1], a holistic and reliable simulation of human locomotion is still under development. Most recent research has shown that reinforcement learning techniques are promising for training human locomotion controllers in simulation environments. These controllers have been validated to produce human-like musculoskeletal simulations that could be useful in predicting responses to assistive devices or therapies like targeted strength training. Moreover, researchers could gain insight into human motor control by training a controller with deep RL in certain conditions (i.e., objective functions, simulation environment, etc.) and by analyzing the controller. One could also train controllers to mimic human motion (e.g., using imitation learning, where a controller is trained to replicate behaviors demonstrated by an expert [2]) or integrate an existing neuromechanical control model with artificial neural networks to study certain aspects of human motor control [3].

Musculoskeletal simulations using deep reinforcement learning (RL) can help us overcome current control models' limitations. Furthermore, deep learning advances have enabled the development of controllers with high-dimensional inputs and outputs for

 doi:10.20944/preprints202212.0167.v1

human musculoskeletal models. The results of versatile controllers could help investigate human motor controls, despite the discrepancy between artificial and biological neural networks [3,4]. To explain the relevant research, we first review the studies pertinent to the musculoskeletal simulation and then the appropriate reinforcement algorithms that have been used for the simulation purpose.

1.1. Musculoskeletal simulations

Musculoskeletal models typically have rigid segments, and muscle-tendon actuators [5–7] (Figure 3). In skeletal systems, rigid segments are usually connected by rotational joints. It is common to actuate the joints using Hill-type muscle models [8], which incorporate both active and passive contractile elements [3,7,9,10] (Figure 1). In simulations, Hill-type muscle models can be used to estimate metabolic energy consumption and muscle fatigue. The musculoskeletal parameter values could be derived from measurements taken from a large number of people, and cadavers [11–13] and can be customized based on the height and weight of an individual, as well as CT and MRI scan data [14,15]. OpenSim [16], which is the basis of the OpenSim-RL package [1] used in the Learn to Move competition, is a widely used open-source software package in biomechanics to simulate musculoskeletal dynamics [3].

A wide variety of human motion recordings have been analyzed using musculoskeletal simulations. Through a variety of computational methods, muscles are found to be activated in one common approach, allowing for the tracking of reference motion data, such as motion capture data and ground reaction forces, while minimizing muscle effort [17,18]. Using a simulation, we can estimate body states, such as individual muscle forces, that are difficult to measure directly. Using this approach, human walking and running have been validated by comparing the simulated muscle activation to recorded electromyography data [19,20]. It has been shown that motion-tracking approaches are useful for predicting locomotion diseases [21,22], analyzing human locomotion [17,23], controlling assistive devices [24–26], and predicting how exoskeleton assistance and surgical interventions will affect muscle coordination [27,28]. Despite being able to analyze recorded motions with these simulations, they cannot predict movement in novel scenarios since they do not produce new motions [3].

It is also possible to create musculoskeletal motions without reference motion data using trajectory optimization methods [29]. This approach finds muscles that produce the target motion through muscle activation patterns and musculoskeletal model optimization, assuming that the target motion is well optimized. In this way, this method has produced well-practiced motor tasks, such as walking and running [30,31], as well as insights into the optimal gait for different objectives [32,33], biomechanical features [34], and assistive devices [35]. If a behavior has not been trained well and is, therefore, functionally suboptimal, the application of this approach is not straightforward. For example, when wearing lower leg exoskeletons, people initially walk inefficiently and adapt to more energy-efficient gaits over days and weeks [36], so trajectory optimization based on energy minimization likely would not predict the initial gait. Physiological control constraints, such as neural transmission delays and limited sensory information, limit human brain function by producing functionally suboptimal behaviors due to the nervous system being optimized for typical motions, such as walking. A better representation of the underlying controller might be necessary to predict emergent behaviors that deviate from minimum effort optimal behavior.

1.2. Reinforcement learning for simulation of human locomotion

A reinforcement learning paradigm is an approach to solving decision-making problems using machine learning. Hence, through interactions with its environment, an agent tries to optimize its policy pi to maximize its cumulative reward [37] (Figure. 2). Higher cumulative rewards can be obtained with better-followed target velocities and lower muscle effort in this study's musculoskeletal model and physics-based simulation environment. A

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 December 2022 doi:10.20944/preprints202212.0167.v1

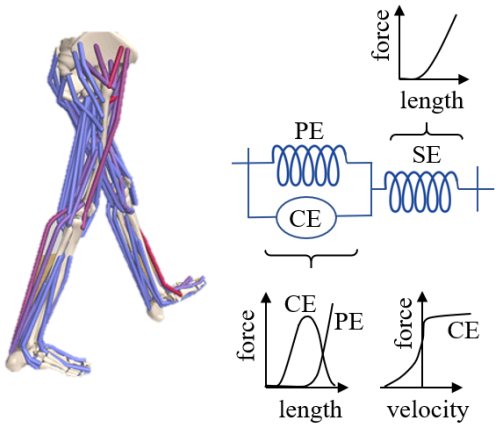

Figure 1. Hill-type muscle models consist of contractile elements (CE), parallel elastic elements (PE), and series elastic elements (SE). Depending on the length and velocity of the contractile element, it produces contractile forces proportional to the excitation signal. Passive elements act as non-linear springs with length-dependent forces.

general RL problem involves receiving observations o_t at timestep t and querying its policy for the action a_t (excitation values of the muscles in the model) at timestep t. Observations are full or partial descriptions of the state of the environment at timestep t. $\pi(a_t|o_t)$ can be either stochastic or deterministic, with a stochastic policy defining a distribution over actions at timestep t [38–40]. It is possible to calculate gradients from non-differentiable objective functions [41], such as those generated from neuromechanical simulations, and then use the gradients as a basis for updating the policies. After applying the action in the environment, the agent transitions to a new state s_{t+1} and receives a scalar reward $r_t = r(s_t, a_t, s_{t+1})$. Using a dynamics model, we determine the state transition $\rho(s_{t+1}|s_t, a_t)$. A policy should be learned that maximizes the agent's cumulative reward.

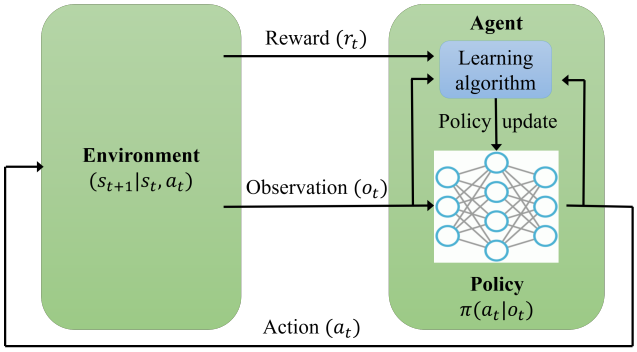

Figure 2. Reinforcement learning algorithm

One of the crucial design decisions in applying RL to a particular problem is the choice of policy representation. Deep RL is the combination of RL with deep neural network function approximators. While a policy can be modeled by any class of functions that maps observations to actions, the use of deep neural networks to model policies demonstrated promising results in complex problems and has led to the emergence of the field of deep RL. The policies trained with deep RL methods achieved high performance [42–45].

In the OpenSim-RL environment [1], where the actions are continuous values of muscle excitations, model-free deep RL algorithms are widely used for continuous control tasks, such as learning to walk the agent using continuous values of muscle excitations. Model-free algorithms do not learn an explicit dynamics model of state transitions; instead, they directly learn a policy to maximize the expected return, or reward. In these continuous control tasks, the policy specifies actions that represent continuous quantities such as

 doi:10.20944/preprints202212.0167.v1

control forces or muscle excitations. Policy gradient algorithms incrementally improve a policy by first estimating the gradient of the expected return using trajectories collected from forward simulations of the policy, and then updating the policy via gradient ascent [46].

Even though the standard policy gradient update is simple, it has several drawbacks, such as instability and inefficiency in sampling. Gradient estimators may have a high variance, leading to unstable learning, and it may take a great number of training samples to get a good gradient estimate. The stability of policy gradient methods has been improved using algorithms such as TRPO [47] and PPO [48], which limit policy behavior changes after each step, as measured by relative entropy between policies.

Low sample efficiency is another limitation of policy gradient methods. At each iteration of updating the current policy, standard policy gradient algorithms estimate the gradient from a new batch of data collected with the current policy. As a result, each batch of data is used for a relatively small number of updates, then discarded, and millions of sample data are often needed for a relatively simple problem. In off-policy gradient algorithms, new data from previous iterations of the algorithm can be reused when updating the most recent policy, which greatly reduces the number of samples required to learn effective policies [49–51]. An off-policy algorithm, such as DDPG [49], approximates the policy gradient by fitting a Q-function, $Q(s, a)$, which is the expected return for performing an action a in the current state. Using these methodologies, the learned Q-function is differentiated to approximate a policy gradient, and the policy is then updated using the learned gradient. SAC and TD3 are recent off-policy methods that improve both sample efficiency and stability with a number of modifications.

The application of deep RL to high-dimensional parameter controllers has also been shown to yield promising results [48,49]. An advantage of deep RL is that it enables the learning of controllers based on low-level, high-dimensional representations of the underlying system, reducing the need to design compact control representations manually and having a deeper understanding of motion. As a result of the development of deep RL models, controllers for complex environments as well as complex musculoskeletal models have been trained [52–54]. Moreover, deep RL is compatible with cases where reference motion data can be used to develop the controller [2,53,55]. In this regard, IMU sensor data due to being inexpensive and easy to collect in various environments (except being exposed to an environment with extensive varying magnetic fields) is very desirable to be employed as the reference motions [56].

2. Materials and Methods

Our method combines continuous and discrete action space reinforcement learning by using Soft Actor-Critic (SAC) and Recurrent Experience Replay in Distributed Reinforcement Learning while we use a combination of bio-inspired and TOR for training. We used the L2M2019 environment of Opensim-rl [1] for our simulation. This environment provides a physiologically plausible 3D human model to move following velocity commands with minimum effort [1]. This human model consists of a pelvis segment, a single segment showing the upper section of the body, and several segments for legs (Figure 3).

2.1. Simulation environment

The simulation environment provides observation or the input to the controller consists of a local target velocity map V and the body state S. The states consist of a vector with 97 values for pelvis state, ground reaction forces joint angles and rates and muscle states as observation [1]. In addition, the environment provides a local target velocity field in a $2 \times 11 \times 11$ matrix, representing a 2D vector field on an 11×11 grid (Figure 4.a). The 2D vectors are target velocities, and the 11×11 grid is for every 0.5 meters back-to-front and left-to-side. The agent starts at the coordination of [0,0] and the target coordination is [5,0] (Figure 4.b). The action space consists of a vector with 22 values showing the activation of 22 muscles (11 per leg).

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 December 2022 doi:10.20944/preprints202212.0167.v1

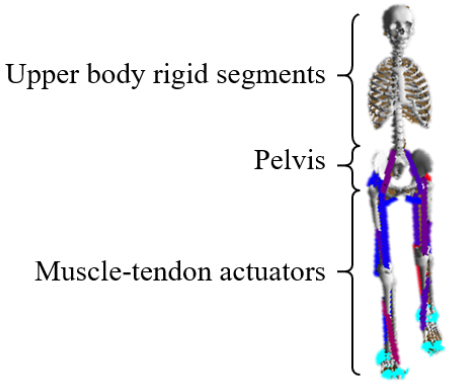

Figure 3. The body sections of the musculoskeletal, which consist of one upper body segment, a pelvis, and several segments for legs

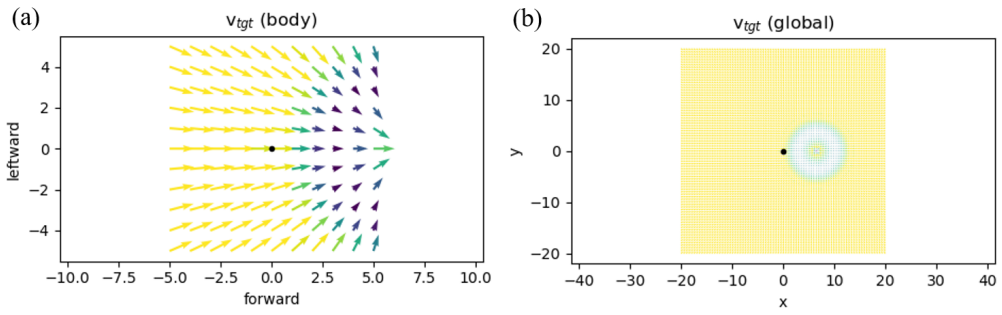

Figure 4. The path that the agent should walk toward the target

Moreover, the environment offers difficulty levels in this environment. Although in difficulty=2 the environment assigns a random location as the target for the agent, in difficulty=1, the target is located at the coordination of [5,0] (Figure 4). This scenario is the same as our data collection. In our data collection, one 35-year old male participant with 170 cm height and 75 kg weight walked in a straight line for 5 meters. As shown in Figure 5, the IMU sensor, Shimmer, was connected to the pelvis of the participant. Then, a penalty was considered for deviation of the observed IMU values of the agent in the environment from the collected IMU values. This constrain builds the bio-inspired reward of our algorithm. For TOR, we used the defined reward at [54]. The following equations show all the defined reward functions:

The total reward J(π) is high when the human model locomotes at desired velocities with minimum effort:

$$J(\pi) = R_{alive} + R_{step} = \sum_i r_{alive} + \sum_i r_{step}(w_{step}.r_{step}\text{-}w_{vel}.c_{vel}\text{-}w_{eff}.c_{eff}) \quad (1)$$

where R_{alive} prevents the agent from falling and step term urges the agent to move toward the target, which is here coordination of [5,0] on Figure 4.a. Also, in the OpenSim-rl [1], r_{alive}, r_{step}, c_{vel}, and c_{eff} are defined as:

Figure 5. The attached IMU to the participant's shimmer for data collection during straight walking

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 December 2022 doi:10.20944/preprints202212.0167.v1

$$
\begin{aligned}
r_{alive} &= 0.1 \\
r_{step} &= \sum_{i\, in\, step_i} \Delta t_i = \Delta t_{step_i} \\
c_{vel} &= || \sum_{i\, in\, step_i} (v_{vel} - v_{tgt}) \Delta t_i || \\
c_{vel} &= \sum_{i\, in\, step_i} \sum_{m}^{muscles} A_m^2 \Delta t_i
\end{aligned}
\tag{2}
$$

in equation 2, Δ = 0.01 second is the simulation timestep, v_vel is the velocity of the pelvis, v_tgt is the target velocity, A_ms are the muscle activations, and w_step, w_vel, and w_eff are the weights for the stepping reward and velocity and effort cost.

The objective of this paper is to simulate the musculoskeletal agent walking similar to the participant that IMU data has been collected from using reinforcement learning. To this end, the RL is utilized to develop a policy $\pi(a_t \mid s_t)$ to maximize the discounted sum of the expected rewards:

$$
J(\pi) = \sum_t \mathbb{E}_{(s_t, a_t) \sim \rho_\pi} [\gamma^t r(s_t, a_t)] \tag{3}
$$

where $s_t \in S$ is state, $a_t \in A$ is action, $r : S \times A \rightarrow [r_{min}, r_{max}]$ is reward function and ρ_π presents the state-action marginals of the trajectory distribution induced by the policy $\pi(a_t \mid s_t)$. As the main RL procedure as described in [54], Soft Actor-Critic algorithm, [51,57] is used. SAC is the current state-of-the-art DDPG improvement. Deep Deterministic Policy Gradient (DDPG) is an off-policy RL method, which is used for continuous action spaces. The off-policy methods allow data re-usage for policy optimization. SAC method relies on maximizing the entropy while the agent maximize the expected reward. SAC algorithm has shown its data efficiency, learning stability, and hyper-parameter robustness. Maximum entropy RL framework augments reward term in equation 3 with an entropy term:

$$
J(\pi) = \sum_t \mathbb{E}_{(s_t, a_t) \sim \rho_\pi} [\gamma^t (r(s_t, a_t) + \alpha \mathcal{H}(\pi(\cdot \mid s_t)))] \tag{4}
$$

where α is a trade-off between the entropy and reward and thus controls the stochasticity of the optimal policy. Reinforcement learning methods using off-policy continuous action spaces are based on the actor-critic pair, where the critic estimates Q-value:

$$
Q_\pi(s_t, a_t) = r(s_t, a_t) + \sum_{k=t+1} \mathbb{E}_{(s_k, a_k) \sim \rho_\pi} [\gamma^k (r(s_k, a_k) + \alpha \mathcal{H}(\pi(\cdot \mid s_t)))] \tag{5}
$$

In practice actor and critic are represented by neural networks π_ϕ $(a_t \mid s_t)$ and Q_θ $(s_t \mid a_t)$ with parameters ϕ and θ. Standard practice is to estimate mean and variance of factorized Gaussian distribution, $\pi_\phi(a_t \mid s_t) = \mathcal{N}(\mu_\phi(s_t), \sum_\phi(s_t))$. A distribution like this allows for reparametrization and policy training through backpropagation. Using such parametrization, learning objectives for actor, critic, and entropy parameters read as follows:

$$
\begin{aligned}
J_\pi(\phi) &= \mathbb{E}_{s_t \sim \mathcal{D}} [\mathbb{E}_{a_t \sim \pi_\phi} [\alpha \log(\pi_\phi(a_t \mid s_t)) - Q_\theta(s_t, a_t)]], \\
J_Q(\theta) &= \mathbb{E}_{(s_t, a_t) \sim \mathcal{D}} [\frac{1}{2} (Q_\theta(s_t, a_t) - (r(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1} \sim p} [V_{\bar{\theta}}(s_{t+1})]))^2], \\
J(\alpha) &= \mathbb{E}_{a_t \sim \pi_t} [-\alpha \log \pi_t(a_t \mid s_t) - \alpha \bar{\mathcal{H}}]
\end{aligned}
\tag{6}
$$

The experience replay is denoted by D, and the objectives can be optimized by using any stochastic gradient descent method. Aside from the previously mentioned properties, the policy has another good point: it constantly investigates promising actions and abandons the ones that obviously don't work.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 December 2022 doi:10.20944/preprints202212.0167.v1

Experience Replay (ER) is the traditional data storage method for off-policy reinforcement learning algorithms. During training, an agent sends transactions (s, a, r, s') while collecting data from the environment. A Prioritized Experience Replay [58] adjusts its sample probabilities based on the loss value associated with a transaction, so training is more likely to include transactions with higher loss values. In the R2D2 approach, the transaction itself is not stored in ER, but overlapping sequences of consecutive (s, a, r) transactions. Sequences never cross episode boundaries and overlap by half-time steps. These sequences are referred to as segments. To compute prioritization weights over segments, R2D2 pipeline uses n-step prioritization based on n-step TD-errors δ_i over the sequence: $p = \eta\ max_i\ \delta_i + (1\text{-}\eta)\ \bar{\delta}$, where η is set to 0.9.

2.2. Reward Shaping

The reward function is pivotal to RL agents' behavior: they are motivated to maximize the returns from the reward function, so the optimal policy is determined by the reward function [54]. Sparse and/or delayed rewards can make learning difficult in many real-world application domains. RL agents are typically guided by reward signals when interacting with their environment. Learning speed and converged performance can be improved by adding a shaping reward to the reward naturally received from the environment, which is called reward shaping principle. Nonetheless, there are two main problems in using reward shaping in RL [54]: 1) interference of rewards. For example, moving with minimum effort is desired, however, to define the reward function a velocity bonus sums up with effort penalty. 2) difficulty of modifying the existing rewards. When the agent learned through a reward to take an action but the action is not enough to achieve a purpose, e.g., moving a leg but not moving forward. Hence, modifying a reward function is needed, which can cause to forget the previous learned action.

To address these two issues, a Q-function split technique called multivariate reward representation is introduced [54], in which the scalar reward function is weighted as sum of the n terms:

$$r_t = \sum_{i=1}^{n} w_i \cdot r_{i,t} \tag{7}$$

In this approach, the reward terms do not interfere with each other as it uses each term separately and optimize the corresponding Q-function of each reward term. Accordingly, if more physiological reward is collected, more reward functions based on realistic human locomotion can be added to this reward function, which makes this algorithm a suitable choice to use a combination of TOR and bio-inspired physiological data. Also, this multivariate reward approach allows the critic pretraining to add new or remove the existing reward terms. The critic is in fact, represented by the neural network. To remove a reward, the parameters assigned to the reward can be set to zero, or to add a new reward, the matrix should be extended by adding a new row. To train the actor and critic with multivariate reward representation, the vector of critic loss should be optimized and the actor should optimize its policy with the scalar representation of the Q-function:

$$Q(s_t, a_t) = \sum_{i=1}^{n} w_i \cdot Q_i(s_t, a_t) \tag{8}$$

The reward function used here is:

$$\bar{r} = [r_{env}, r_{clp}, r_{vdp}, r_{pvb}, r_{dep}, r_{tab}, r_{entropy}, r_{IMU}^{roll}, r_{IMU}^{pitch}, r_{IMU}^{yaw}] \tag{9}$$

To evaluate addition of the bio-inspired rewards, we kept the reward function used in [54]; however, three terms, $r_{IMU}^{roll}, r_{IMU}^{pitch}, r_{IMU}^{yaw}$, have been added to the reward function and thanks to the multivariate reward representation, they don't interfere with the other rewards in the training process. $r_{IMU}^{roll}, r_{IMU}^{pitch}, r_{IMU}^{yaw}$ are defined as the deviation of the collected IMU data IMU_{col} from the observed IMU data IMU_{obs} of the environment during the training.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 December 2022 doi:10.20944/preprints202212.0167.v1

$$\begin{aligned} r_{IMU}^{roll} &= -|IMU_{col}^{roll} - IMU_{obs}^{roll}| \\ r_{IMU}^{pitch} &= -|IMU_{col}^{pitch} - IMU_{obs}^{pitch}| \\ r_{IMU}^{yaw} &= -|IMU_{col}^{yaw} - IMU_{obs}^{yaw}| \end{aligned} \tag{10}$$

Also, the weight of 1 was assigned to these rewards (equation 7). Another benefit is that if more physiological data (e.g., more IMU data from other parts of the body) were collected, the reward function can extend and thus, more bio-inspired constrains can be added to the reward function. Consequently, the musculoskeletal mimicking the human's locomotion tasks can get closer to the real scenarios. The other rewards are defined as follow:

crossing legs penalty (r_{clp}) is defined to stop the tendency of the agent to cross its legs.

$$r_{clp} = min(0, (r^{head} - r^{pelvis}, r^{left} - r^{pelvis}, r^{right} - r^{pelvis})) \tag{11}$$

where r is a radius vector. To encourage the agent to move at the early stages, r_{pvb}, pelvis velocity bonus is used:

$$r_{pvb} = ||v_{body}|| \tag{12}$$

Velocity deviation penalty r_{vdp} is defined to guide the agent toward the target.

$$r_{vdp} = -\sum_{i\, in\, step_i} ||v_{body} - v_{tgt}|| \tag{13}$$

r_{dep}, dense effort penalty, is to move the agent with minimal effort

$$r_{dep} = -||action_t|| \tag{14}$$

to force the agent stop at the target, the reward of target achieve bonus is added (r_{tab}):

$$r_{tab} = \begin{cases} 0, & 0.7 < ||v_{tgt}|| \\ 0.1, & 0.5 < ||v_{tgt}|| \leq 0.7 \\ 1 - 3.5||v_{tgt}||^2, & ||v_{tgt}|| \leq 0.5 \end{cases} \tag{15}$$

The last reward coordinate is entropy bonus from SAC:

$$r_{entropy} = \alpha * \mathcal{H}(\pi(\cdot|s_t)) \tag{16}$$

Finally, in our method, the described multivariate reward function, which is the combination of bio-inspired inertial-constrained and TOR, loss functions and networks from SAC, parallel data collection and prioritization, also n-step Q-learning and invertible value function rescaling from R2D2 were used to train the agent.

3. Results

The process of learning in this agent is first starting to walk in any direction and then walking 5 meters straight. To achieve the first end, 4-layer perception for both policy and critic networks, with input size of dim(S) = 97 for policy and dim(S) + dim(A) = 97 + 22 = 119 for the critic, hidden size of 256, layer norm before activation function, 'ELU' activation for policy and 'ReLU' for the critic, and residual connections. The discount factor,γ, is set to 0.99. Experience replay size is 250.000 with a segment length of 10. Also, 30 data sampler was set to provide a fair judgment in the case that no IMU-constrained is used. Adam optimizer with the learning rate of 3×10^{-5} for policy and 10^{-4} for critic. The batch size was equal to 256 and the segment length to 10. Priority exponents α and β were set to 0.1 at the beginning of training and linearly increased to 0.9 in 3000 training steps.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 December 2022 doi:10.20944/preprints202212.0167.v1

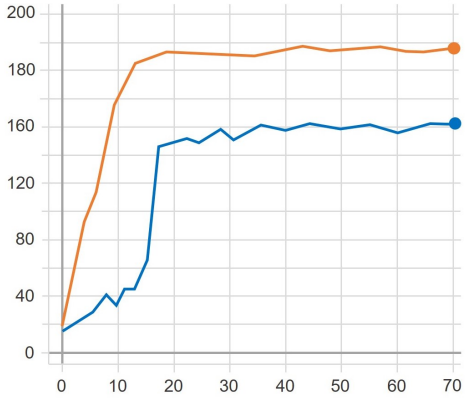

Figure 6. Comparison of the reward obtained by the agent with the same training condition, when no IMU sensor is used (blue) with using the IMU-constrain (orange). The horizontal axis shows the training time (hr) and the vertical axis shows the reward.

The second step of learning, after starting to walk in any direction, is to walk forward toward the target. To this end, a new model with $\pi^s_\phi(a_t|s_t, v_t)$ and $Q^s_\theta(s_t, v_t, a_t)$ are trained by minimizing Kullback-Leibler divergence between policies and mean squared error between critics on data from previously saved experience replay:

$$\begin{aligned} J\pi^s(\phi) &= \mathbb{E}_{st\sim\mathcal{D}}\mathbb{E}_{v_t\sim\mathcal{N}(0,0.1)}[D_{KL}(\pi^s_\phi(a_t|s_t, v_t)||\pi^t_{\phi'}(a_t|s_t))] \\ J_{Q^s}(\theta) &= \mathbb{E}_{st\sim\mathcal{D}}\mathbb{E}_{v_t\sim\mathcal{N}(0,0.1)}(Q^s_\theta(s_t, v_t, a\sim\pi^s(\cdot|s_t, v_t)) - Q^t_{\theta'}(s_t, a\sim\pi^t(\cdot|s_t)))^2 \end{aligned} \tag{17}$$

Models $\pi^s(a_t|s_t, v_t)$ and $Q^s(s_t, v_t, a_t)$ share the same architecture with $\pi^t(a_t, s_t)$ and $Q^t(a_t, s_t)$ except input dim is now dim(S) + dim(V) = $97 + 2 \cdot 11 \cdot 11 = 339$ for policy and dim(S) + dim(V)+dim(A) = 339+22 = 361 for critic and hidden size equals to 1024 for both. The Adam optimizer [59] with the learning rate 10^{-4} is used to optimize the distillation losses for policy and critic networks and batch size 128.

The explained hyperparameters and steps were taken for both cases where no IMU-constrained reward is used and when the IMU-constrained reward is used and the results are shown in Figure 6. Besides the faster training and higher reward, according to the deviation of IMU data from the observation in the environment and the recorded data from the participant approaches zero. In this regard, Figure 7.a shows the path that musculoskeletal walk to reach to be at the target spot, which is straightly 5 meters away from the start point, when IMU data is used for training the agent. Compared to Figure 7.c, where no IMU-constrain has been used to guide the agent to the target, the latter case, Figure 7.c, shows some deviations from the straight path; however, in Figure 7.a the agent walks in a more straight way. This roots in the fact that IMU-constrains provide an accurate guideline for the agent to achieve its goal. For further demonstration, the musculoskeletal walking frames when IMU-constrain is used (Figure 7.b) and when no IMU constrain is used (Figure 7.d) demonstrate the deviation of the agent from straight walking by showing its effect on the way that agent takes the steps and body direction (e.g., the agent's head direction) when no IMU-constrain is used. To investigate the deviation of the agent from the locomotion behavior of the participant the Root Mean Square Error (RMSE) has been calculated and RMSE for roll, pitch, and yaw data are 0.8824, 0.5825, and 1.5908; respectively (Figure 8.a,c,e). Moreover, 8.b,d,f compare the observed IMU data in the simulation environment when IMU data is used for training (orange) and when no IMU data is used for training (green). There is an increasing trend in the observed IMU data from the simulation environment when no IMU data is used for training.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 December 2022 doi:10.20944/preprints202212.0167.v1

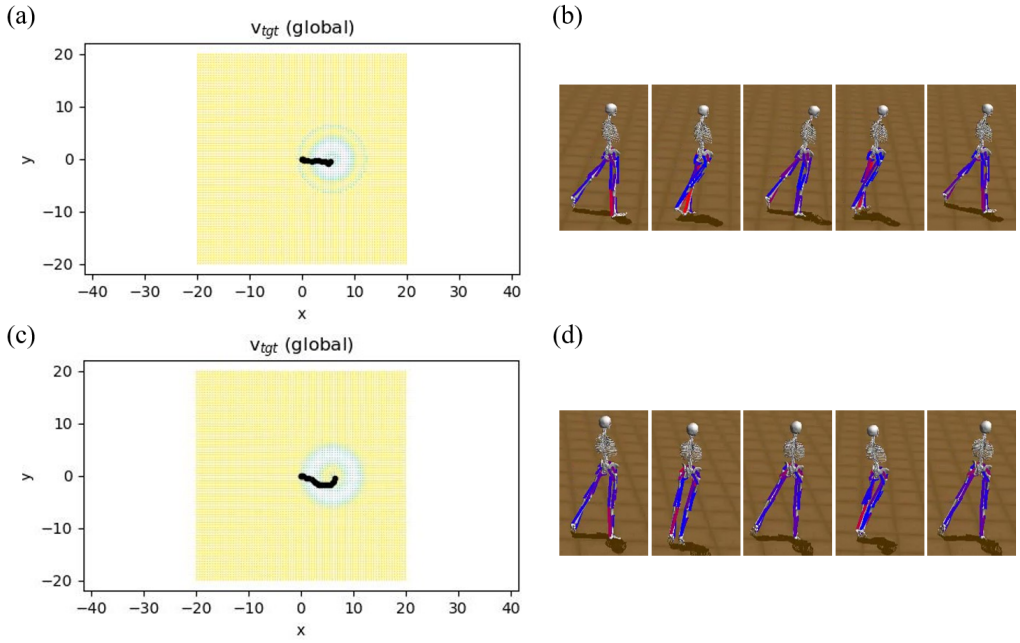

Figure 7. a) The musculoskeletal locomotion trajectory and b) frames for a 5-meter straight walk when no IMU constrain is used for training. c) The musculoskeletal locomotion trajectory and d) frames for a 5-meter straight walk when IMU constrain is used for training.

4. Discussion

This study developed an integrative framework for designing a novel reward function of both TOR and bio-inspiration to develop RL techniques to model human motor control. Experimental results have demonstrated the improvement of the models in training time reduction and increased rewards in simulating human locomotion, compared with previous work that did not consider human motion data. Our contributions are specifically 1) to introduce the novel reward function combining TOR and bio-inspired reward function and 2) to demonstrate a computational framework for redesigning reward function to improve human locomotion simulation models.

As shown in Figure 6, the RL model with IMU-constrained reward function showed a faster learning rate and could achieve a higher reward. With this approach, the IMU constraints provide a reference guideline for the agent to walk similarly to natural human movements. These rewards could help the agent achieve a higher reward and faster learning than the model when no bio-inspiration is being used. Noteworthy is that this finding suggests that integrating real-world data into the reward function of the RL techniques could help the simulation models escape some anomalies or saddle points during the training process. This observation is consistent with other theoretical analyses of deep neural networks' convergence processes [60].

According to the results shown in Figure 8, the participants' pelvis motion for a straight walk can be replicated through an acceptable training process of the RL models. Hence, with accurate models of the human anatomy (e.g., OpenSim [61]), without any invasive procedure, the participants' locomotion disorders can be investigated, or at least the pelvis part of the participant can be analyzed accurately. This function will be beneficial for medical applications. For instance, a rehabilitation therapist could import their patients' IMU sensor data to the RL models. The models could provide estimates of the patients' musculoskeletal mechanisms to assist the therapist in identifying the potential issues during rehabilitation and determining better strategies.

Another significant implication of our research is that the experimental results only rely on one single IMU sensor on partipants' pelvis. In the last decade, despite the advances of wearable and mobile techniques, the affordability and acceptance of sensing techniques

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 December 2022 doi:10.20944/preprints202212.0167.v1

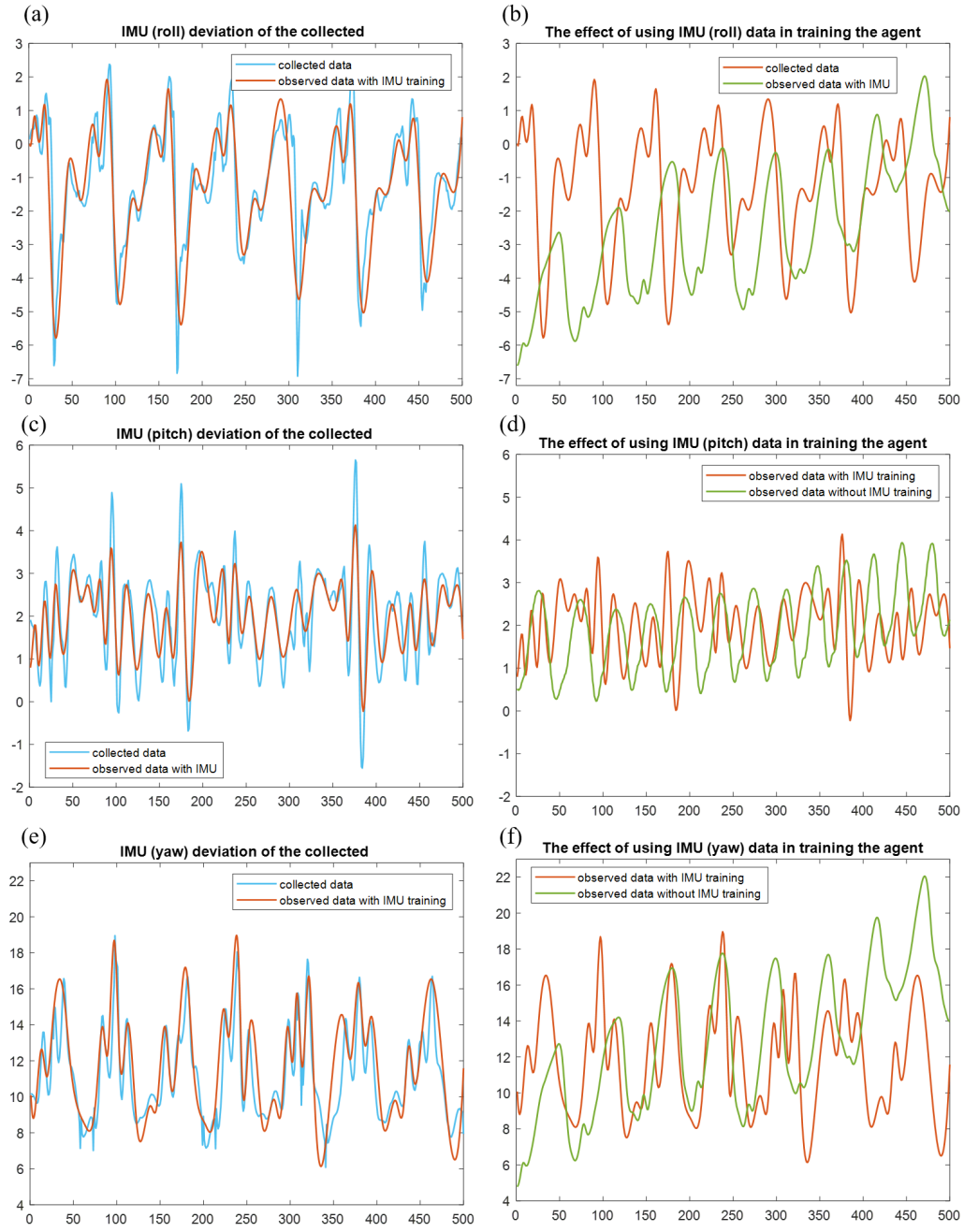

Figure 8. The IMU deviation of the collected IMU from the observed IMU (m/s^2) for the first five seconds of the simulation is shown on the vertical axes and the horizontal axes show the 500 frames of the time steps: a) roll, RMSE = 0.8824 C) pitch, RMSE = 0.5825 e) yaw, RMSE = 1.5908.

have constrained existing studies on human locomotion. Using much of the bio-sensor data is challenging as it is costly to collect. Also, not all of the sensors can be used in all environments. For example, high-resolution cameras in an open environment or Electromyography (EMG) data in a laboratory environment. In addition, by raising the need for remote health monitoring and reducing the need for patients to attend doctors' clinics, it is very desirable that patients use some easy-to-wear sensors such as IMU sensors and send the data to the doctors. Previous efforts in data collection of human locomotion for healthcare research mostly instructed the participants to wear a wearable IMU sensor, such as Fitbit, close to the pelvis [62]. With the robust framework developed in this research, their locomotion could be simulated and assessed remotely and retrospectively. This is

 doi:10.20944/preprints202212.0167.v1

the original motivation of this research to equip researchers with a simulation model to reinvestigate the existing human locomotion dataset.

The limitations of this study are the relatively small sample size and the small number of musculoskeletal tasks in the experiment, the inaccuracy of the computational models, and other factors during the data collection and preprocessing. Because the RL framework is designed for personalized human locomotion modeling, the experiments focused on training, testing, and validating individual participants' data. Further work will explore the characteristics of the models across participants and quantify the uncertainties during the generalization process. In addition, interpreting the RL results and training process is still a challenging problem, primarily how to ensure its clinical meaningness. Furthermore, in future work, we will explore other musculoskeletal tasks, such as jogging, jumping, and running, to embrace the knowledge learned from this study.

5. Conclusions and future work

Our study showed that integrating IMU data into the RL framework reward functions could improve human locomotion simulation. In our experiments, IMU data collected from a participant walking in a straightforward way for 5 meters were used to train a musculoskeletal model in a simulation environment. Consequently, this bio-inspired constraint could help the agent move its pelvis like the human from which IMU data was collected. This concept was shown through a comparison of the trajectory, walking frames, obtained reward in RL, and comparison of the collected IMU data with the observed IMU data in the simulation environment. The comparison was conducted between musculoskeletal agents trained with IMU data and those trained without IMU data. The results demonstrated the improved performance of musculoskeletal agents trained with IMU data, including faster convergence, higher reward, and better simulated human locomotion. These findings are consistent with the existing theoretical work in escaping saddle points of deep neural networks. Furthermore, we discussed the implications and potential medical applications of these findings. In future work, we will implement the RL models with more than one IMU constraint on different parts of the body to replicate more complicated locomotion tasks such as jogging, jumping, and running.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Kidziński, Ł.; Mohanty, S.P.; Ong, C.F.; Hicks, J.L.; Carroll, S.F.; Levine, S.; Salathé, M.; Delp, S.L. Learning to run challenge: Synthesizing physiologically accurate motion using deep reinforcement learning. In *The NIPS'17 Competition: Building Intelligent Systems*; Springer, 2018; pp. 101–120.
2. Peng, X.B.; Abbeel, P.; Levine, S.; Van de Panne, M. Deepmimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM Transactions On Graphics (TOG)* **2018**, *37*, 1–14.
3. Song, S.; Kidziński, Ł.; Peng, X.B.; Ong, C.; Hicks, J.; Levine, S.; Atkeson, C.G.; Delp, S.L. Deep reinforcement learning for modeling human locomotion control in neuromechanical simulation. *Journal of neuroengineering and rehabilitation* **2021**, *18*, 1–17.
4. Richards, B.A.; Lillicrap, T.P.; Beaudoin, P.; Bengio, Y.; Bogacz, R.; Christensen, A.; Clopath, C.; Costa, R.P.; de Berker, A.; Ganguli, S.; et al. A deep learning framework for neuroscience. *Nature neuroscience* **2019**, *22*, 1761–1770.
5. Seth, A.; Dong, M.; Matias, R.; Delp, S. Muscle contributions to upper-extremity movement and work from a musculoskeletal model of the human shoulder. *Frontiers in neurorobotics* **2019**, *13*, 90.
6. Rajagopal, A.; Dembia, C.L.; DeMers, M.S.; Delp, D.D.; Hicks, J.L.; Delp, S.L. Full-body musculoskeletal model for muscle-driven simulation of human gait. *IEEE transactions on biomedical engineering* **2016**, *63*, 2068–2079.
7. Haeufle, D.; Günther, M.; Bayer, A.; Schmitt, S. Hill-type muscle model with serial damping and eccentric force–velocity relation. *Journal of biomechanics* **2014**, *47*, 1531–1536.
8. Hill, A.V. The heat of shortening and the dynamic constants of muscle. *Proceedings of the Royal Society of London. Series B-Biological Sciences* **1938**, *126*, 136–195.
9. Geyer, H.; Herr, H. A muscle-reflex model that encodes principles of legged mechanics produces human walking dynamics and muscle activities. *IEEE Transactions on neural systems and rehabilitation engineering* **2010**, *18*, 263–273.
10. Millard, M.; Uchida, T.; Seth, A.; Delp, S.L. Flexing computational muscle: modeling and simulation of musculotendon dynamics. *Journal of biomechanical engineering* **2013**, *135*.
11. Chandler, R.; Clauser, C.E.; McConville, J.T.; Reynolds, H.; Young, J.W. Investigation of inertial properties of the human body. Technical report, Air Force Aerospace Medical Research Lab Wright-Patterson AFB OH, 1975.
12. Visser, J.; Hoogkamer, J.; Bobbert, M.; Huijing, P. Length and moment arm of human leg muscles as a function of knee and hip-joint angles. *European journal of applied physiology and occupational physiology* **1990**, *61*, 453–460.
13. Ward, S.R.; Eng, C.M.; Smallwood, L.H.; Lieber, R.L. Are current measurements of lower extremity muscle architecture accurate? *Clinical orthopaedics and related research* **2009**, *467*, 1074–1082.
14. Scheys, L.; Loeckx, D.; Spaepen, A.; Suetens, P.; Jonkers, I. Atlas-based non-rigid image registration to automatically define line-of-action muscle models: a validation study. *Journal of biomechanics* **2009**, *42*, 565–572.
15. Fregly, B.J.; Boninger, M.L.; Reinkensmeyer, D.J. Personalized neuromusculoskeletal modeling to improve treatment of mobility impairments: a perspective from European research sites. *Journal of neuroengineering and rehabilitation* **2012**, *9*, 1–11.
16. Seth, A.; Hicks, J.L.; Uchida, T.K.; Habib, A.; Dembia, C.L.; Dunne, J.J.; Ong, C.F.; DeMers, M.S.; Rajagopal, A.; Millard, M.; et al. OpenSim: Simulating musculoskeletal dynamics and neuromuscular control to study human and animal movement. *PLoS computational biology* **2018**, *14*, e1006223.
17. De Groote, F.; Van Campen, A.; Jonkers, I.; De Schutter, J. Sensitivity of dynamic simulations of gait and dynamometer experiments to hill muscle model parameters of knee flexors and extensors. *Journal of biomechanics* **2010**, *43*, 1876–1883.
18. Thelen, D.G.; Anderson, F.C.; Delp, S.L. Generating dynamic simulations of movement using computed muscle control. *Journal of biomechanics* **2003**, *36*, 321–328.
19. Liu, M.Q.; Anderson, F.C.; Schwartz, M.H.; Delp, S.L. Muscle contributions to support and progression over a range of walking speeds. *Journal of biomechanics* **2008**, *41*, 3243–3252.
20. Hamner, S.R.; Seth, A.; Delp, S.L. Muscle contributions to propulsion and support during running. *Journal of biomechanics* **2010**, *43*, 2709–2716.

21. Wang, C.; Zeng, L.; Li, Y.; Shi, C.; Peng, Y.; Pan, R.; Huang, M.; Wang, S.; Zhang, J.; Li, H. Decabromodiphenyl ethane induces locomotion neurotoxicity and potential Alzheimer's disease risks through intensifying amyloid-beta deposition by inhibiting transthyretin/transthyretin-like proteins. *Environment International* **2022**, *168*, 107482.
22. Wong, Y.B.; Chen, Y.; Tsang, K.F.E.; Leung, W.S.W.; Shi, L. Upper extremity load reduction for lower limb exoskeleton trajectory generation using ankle torque minimization. In Proceedings of the 2020 16th International Conference on Control, Automation, Robotics and Vision (ICARCV). IEEE, 2020, pp. 773–778.
23. De Groote, F.; Kinney, A.L.; Rao, A.V.; Fregly, B.J. Evaluation of direct collocation optimal control problem formulations for solving the muscle redundancy problem. *Annals of biomedical engineering* **2016**, *44*, 2922–2936.
24. Cavallaro, E.E.; Rosen, J.; Perry, J.C.; Burns, S. Real-time myoprocessors for a neural controlled powered exoskeleton arm. *IEEE Transactions on Biomedical Engineering* **2006**, *53*, 2387–2396.
25. Bassiri, Z.; Austin, C.; Cousin, C.; Martelli, D. Subsensory electrical noise stimulation applied to the lower trunk improves postural control during visual perturbations. *Gait & Posture* **2022**, *96*, 22–28.
26. Lotti, N.; Xiloyannis, M.; Durandau, G.; Galofaro, E.; Sanguineti, V.; Masia, L.; Sartori, M. Adaptive model-based myoelectric control for a soft wearable arm exosuit: A new generation of wearable robot control. *IEEE Robotics & Automation Magazine* **2020**, *27*, 43–53.
27. Uchida, T.K.; Seth, A.; Pouya, S.; Dembia, C.L.; Hicks, J.L.; Delp, S.L. Simulating ideal assistive devices to reduce the metabolic cost of running. *PloS one* **2016**, *11*, e0163417.
28. Fox, M.D.; Reinbolt, J.A.; Õunpuu, S.; Delp, S.L. Mechanisms of improved knee flexion after rectus femoris transfer surgery. *Journal of biomechanics* **2009**, *42*, 614–619.
29. De Groote, F.; Falisse, A. Perspective on musculoskeletal modelling and predictive simulations of human movement to assess the neuromechanics of gait. *Proceedings of the Royal Society B* **2021**, *288*, 20202432.
30. Anderson, F.C.; Pandy, M.G. Dynamic optimization of human walking. *J. Biomech. Eng.* **2001**, *123*, 381–390.
31. Falisse, A.; Serrancolí, G.; Dembia, C.L.; Gillis, J.; Jonkers, I.; De Groote, F. Rapid predictive simulations with complex musculoskeletal models suggest that diverse healthy and pathological human gaits can emerge from similar control strategies. *Journal of The Royal Society Interface* **2019**, *16*, 20190402.
32. Ackermann, M.; Van den Bogert, A.J. Optimality principles for model-based prediction of human gait. *Journal of biomechanics* **2010**, *43*, 1055–1060.
33. Miller, R.H.; Umberger, B.R.; Hamill, J.; Caldwell, G.E. Evaluation of the minimum energy hypothesis and other potential optimality criteria for human running. *Proceedings of the Royal Society B: Biological Sciences* **2012**, *279*, 1498–1505.
34. Miller, R.H.; Umberger, B.R.; Caldwell, G.E. Limitations to maximum sprinting speed imposed by muscle mechanical properties. *Journal of biomechanics* **2012**, *45*, 1092–1097.
35. Handford, M.L.; Srinivasan, M. Energy-optimal human walking with feedback-controlled robotic prostheses: a computational study. *IEEE Transactions on Neural Systems and Rehabilitation Engineering* **2018**, *26*, 1773–1782.
36. Zhang, J.; Fiers, P.; Witte, K.A.; Jackson, R.W.; Poggensee, K.L.; Atkeson, C.G.; Collins, S.H. Human-in-the-loop optimization of exoskeleton assistance during walking. *Science* **2017**, *356*, 1280–1284.
37. Sutton, R.S.; Barto, A.G. *Reinforcement learning: An introduction*; MIT press, 2018.
38. Levine, S. Reinforcement learning and control as probabilistic inference: Tutorial and review. *arXiv preprint arXiv:1805.00909* **2018**.
39. Kuang, N.L.; Leung, C.H.; Sung, V.W. Stochastic reinforcement learning. In Proceedings of the 2018 IEEE First International Conference on Artificial Intelligence and Knowledge Engineering (AIKE). IEEE, 2018, pp. 244–248.
40. Azimirad, V.; Ramezanlou, M.T.; Sotubadi, S.V.; Janabi-Sharifi, F. A consecutive hybrid spiking-convolutional (CHSC) neural controller for sequential decision making in robots. *Neurocomputing* **2022**, *490*, 319–336.
41. Schulman, J.; Heess, N.; Weber, T.; Abbeel, P. Gradient estimation using stochastic computation graphs. *Advances in Neural Information Processing Systems* **2015**, *28*.
42. Vinyals, O.; Babuschkin, I.; Czarnecki, W.M.; Mathieu, M.; Dudzik, A.; Chung, J.; Choi, D.H.; Powell, R.; Ewalds, T.; Georgiev, P.; et al. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature* **2019**, *575*, 350–354.

43. Badnava, B.; Kim, T.; Cheung, K.; Ali, Z.; Hashemi, M. Spectrum-Aware Mobile Edge Computing for UAVs Using Reinforcement Learning. In Proceedings of the 2021 IEEE/ACM Symposium on Edge Computing (SEC). IEEE, 2021, pp. 376–380.
44. Akhavan, Z.; Esmaeili, M.; Badnava, B.; Yousefi, M.; Sun, X.; Devetsikiotis, M.; Zarkesh-Ha, P. Deep Reinforcement Learning for Online Latency Aware Workload Offloading in Mobile Edge Computing. *arXiv preprint arXiv:2209.05191* **2022**.
45. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.A.; Veness, J.; Bellemare, M.G.; Graves, A.; Riedmiller, M.; Fidjeland, A.K.; Ostrovski, G.; et al. Human-level control through deep reinforcement learning. *nature* **2015**, *518*, 529–533.
46. Sutton, R.S.; McAllester, D.; Singh, S.; Mansour, Y. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems* **1999**, *12*.
47. Schulman, J.; Levine, S.; Abbeel, P.; Jordan, M.; Moritz, P. Trust region policy optimization. In Proceedings of the International conference on machine learning. PMLR, 2015, pp. 1889–1897.
48. Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* **2017**.
49. Lillicrap, T.P.; Hunt, J.J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; Wierstra, D. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971* **2015**.
50. Fujimoto, S.; Hoof, H.; Meger, D. Addressing function approximation error in actor-critic methods. In Proceedings of the International conference on machine learning. PMLR, 2018, pp. 1587–1596.
51. Haarnoja, T.; Zhou, A.; Abbeel, P.; Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Proceedings of the International conference on machine learning. PMLR, 2018, pp. 1861–1870.
52. Peng, X.B.; van de Panne, M. Learning locomotion skills using deeprl: Does the choice of action space matter? In Proceedings of the Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation, 2017, pp. 1–13.
53. Lee, S.; Park, M.; Lee, K.; Lee, J. Scalable muscle-actuated human simulation and control. *ACM Transactions On Graphics (TOG)* **2019**, *38*, 1–13.
54. Akimov, D. Distributed soft actor-critic with multivariate reward representation and knowledge distillation. *arXiv preprint arXiv:1911.13056* **2019**.
55. Liu, L.; Hodgins, J. Learning basketball dribbling skills using trajectory optimization and deep reinforcement learning. *ACM Transactions on Graphics (TOG)* **2018**, *37*, 1–14.
56. Uhlenberg, L.; Amft, O. Comparison of Surface Models and Skeletal Models for Inertial Sensor Data Synthesis. In Proceedings of the 2022 IEEE-EMBS International Conference on Wearable and Implantable Body Sensor Networks (BSN). IEEE, 2022, pp. 1–5.
57. Haarnoja, T.; Zhou, A.; Hartikainen, K.; Tucker, G.; Ha, S.; Tan, J.; Kumar, V.; Zhu, H.; Gupta, A.; Abbeel, P.; et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905* **2018**.
58. Schaul, T.; Quan, J.; Antonoglou, I.; Silver, D. Prioritized experience replay. *arXiv preprint arXiv:1511.05952* **2015**.
59. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* **2014**.
60. Sun, R.; Li, D.; Liang, S.; Ding, T.; Srikant, R. The global landscape of neural networks: An overview. *IEEE Signal Processing Magazine* **2020**, *37*, 95–108.
61. Delp, S.L.; Anderson, F.C.; Arnold, A.S.; Loan, P.; Habib, A.; John, C.T.; Guendelman, E.; Thelen, D.G. OpenSim: open-source software to create and analyze dynamic simulations of movement. *IEEE transactions on biomedical engineering* **2007**, *54*, 1940–1950.
62. Johnson, A.; Bulgarelli, L.; Pollard, T.; Horng, S.; Celi, L.A.; Mark, R. Mimic-iv. *PhysioNet. Available online at: https://physionet. org/content/mimiciv/1.0/(accessed August 23, 2021)* **2020**.