Simulation of Assisted Human Walking Using Musculoskeletal Model Coupled with Exoskeleton via Deep Reinforcement Learning

2021 International Conference on Computer, Control and Robotics (ICCCR) | 978-1-7281-9035-8/20/$31.00 ©2021 IEEE | DOI: 10.1109/ICCCR49711.2021.9349390

Ruiming Luo
College of Computer Science and Technology
Zhejiang University
Hangzhou, China
e-mail: joeluo@zju.edu.cn

Shouqian Sun
College of Computer Science and Technology
Zhejiang University
Hangzhou, China
e-mail: ssq@zju.edu.cn

***Abstract*—This paper presents a novel approach to simulate assisted human walking to evaluate the assistance of the powered lower-limb exoskeleton. We first construct a subject specific musculoskeletal model as an agent that generates muscle forces according to internal and external states, and describe the exoskeleton as a rigid multi-link structure with compensatory torques applied at the joints. Then we train the agent to produce walking motion using a deep reinforcement learning algorithm given recorded experimental data on a biomechanical simulator. Next, we combine the pre-trained musculoskeletal agent with the exoskeleton and perform the assisted walking simulation which takes into account the human exoskeleton dynamics. Simulated energy expenditures under different experimental conditions are compared to evaluate the assistance effectiveness of the exoskeleton. Results show that the proposed method has great potential in providing insights into the human-exoskeleton interaction.**

Keywords-exoskeleton; simulation; reinforcement learning

I. INTRODUCTION

Increasingly, lower limb exoskeletons are being developed as assistive devices to decrease the metabolic cost of human walking, especially in military and industrial scenarios. Although advanced materials and design paradigms of assistive exoskeletons have continually released in the literature [1]-[3], the evaluation of human-robot collaboration is little investigated. Since the exoskeleton is highly customized and expensive, evaluating its assistive efficiency by means of computer simulation before manufacturing is necessary and need to be studied.

One of the challenges is how to model the walking human inside the exoskeleton. Since the leading work by Geyer and Herr [4], various muscle-reflex controllers for human locomotion have been developed [5], [6]. These controllers are extremely sophisticated and require a lot of biomechanics expertise. To generate legged locomotion in a more general way, deep reinforcement learning (RL) techniques have been proposed [7]–[9]. Controllers generated using these techniques need less hand-tuned inputs compared to the aforementioned biomechanical controllers, and require no domain knowledge. However, the training is quite time-consuming and resulting motions are not anthropomorphic and exhibit disturbing artifacts. To tackle this problem, motion capture data are used to accelerate the training process and generate desirable outcomes [10], [11]. Unfortunately, these algorithms are applied to articulated multi-link models without biologically accurate actuators, which fails to reveal muscular properties that is non-trivial in human-exoskeleton interaction.

In recent years, researchers from exoskeleton community have utilized results of biologically muscle-driven models and movement simulation [12], [13] to improve insights of human-exoskeleton interaction. For example, Torricelli et al. tracked the joint angles of the human and the exoskeleton to predict human motion [14]. Uchida et al. [15] simulated human running to generate assistive torques required to decrease energy expenditure when running. And discoveries found from that study were used to decrease the metabolic cost of running with a tethered soft exosuit [16]. Yet exoskeletons dramatically change human's kinematics, these analyses cannot reflect the neuromuscular and kinematic adaptations of the human induced by assistive devices. Thus, we need a better approach that can emulate human locomotion that realistically adapt to various perturbations, including internal physiological variations and externally applied forces.

In this paper, we propose a novel approach of simulating subject-specific assisted walking to evaluate the human-exoskeleton interaction. First, kinematics data of a subject in a walking trial are collected, in order to calculate the corresponding muscle states of the scaled musculoskeletal model. Then we use DDPG algorithm to train the muscle driven agent to generate muscle forces in accordance with internal and external states, aiming at producing the walking motion. Next we perform the assisted walking simulation of the musculoskeletal agent with a lower limb exoskeleton, which is modeled as a rigid multi-link structure with compensatory torques applied at hip/knee joints. In final, experimental results show that the exoskeleton is able to decrease the energy expenditure of human walking.

Contributions of our work are summarized as: (1) We model the human-exoskeleton system as an adaptive muscle-driven agent in the context of RL; (2) We use DDPG algorithm to train the subject-specific musulockeletal agent to produce walking motion with reference trajectory; (3) Assisted human walking simulation is performed on a physics-based simulator to evaluate the effectiveness of the assistance by exoskeleton.

The remaining parts of this paper are organized as follows:

978-1-7281-9035-8/21/$31.00 ©2021 IEEE

Section II mathematically formulate the musculoskeletal agent and its behavior. Section III presents the details of the simulation framework and the agent training method. IV describes the experiment and discusses the results. Section V draws the conclusion and describes future work.

II. Problem Formulation

A. Muscoluskeletal Agent

A musculoskeletal model is composed of rigid skeleton fragments wrapped by geometrically stretchable muscles. The contraction and relaxation of these muscles lead to the rotation of joints, and therefore the motion of the whole body. Details of the muscle dynamics is in [17]. In short, muscle activations that regulated by nervous system determine the length change of the muscle fibers, and further, the change of muscle force.

The musculoskeletal model walks on level ground, and is ceaselessly applied with ground reaction force (GRF). Hunt-Crossley contact model is implemented and uses Hertz contact theory to represents the interaction between the feet and the ground.

To mimic the motion control of human being and formulate the human model with or without assistance by exoskeleton in a unified structure, we describe the musculoskeletal model as an agent governed by a black-box controller, which takes internal physiological states (e.g. fiber force, fiber length) as well as external kinematic data (e.g. joint angle, GRF) as inputs, and takes muscle activations as outputs. In general, the agent is actuated by muscles, GRF and if there is, assistive torque provided by the exoskeleton.

Specifically, the total states for the agent include joint angle q, joint angular velocity $\dot{q}$, trunk tilt φ, forward velocity v, pelvis height h, fiber force f_{fb}, fiber length l_{fb}, normal component of GRF f_c, and state of the exoskeleton s_e. In the interest of brevity, we denote the state $s = \{s_h, s_e\}$, where $s_h = \{q, \dot{q}, f_{fb}, l_{fb}, f_c, \varphi, v, h\}$ is the musculoskeletal state. Note that s_e is constantly set to zero during training since there is no exoskeleton. In this case, it can be considered that an intangible, massless exoskeleton is worn.

B. Agent Behavior

The behavior of the preprocessed musculoskeletal model in the simulation environment is modeled as the Markov Decision Process (MDP), where an agent (muscle-driven model) interacts with an environment (physics-based simulator) by taking actions (muscle activations) based on observations (a function of the state of the agent). A parametric policy $\pi(a|s)$ represents the distribution $a \in A$ given a state $s \in S$. At each timestep, the agent observes the current state s_t and performs an action $a_t = \pi(s_t)$. Then the environment responds with a new state s_{t+1} sampled from the dynamics $p(s_{t+1}|s_t, a_t)$, and an instant reward $r_t \in R$. As the process continues, the agent generates a trajectory $\{s_0, a_0, s_1, a_1, \cdots, s_T\}$ until a termination condition is met.

C. Reward Definition

The reward r at each timestep reflects the desirability of agent movement. It is composed of two terms

$$r = \omega_K r_K + \omega_D r_D \quad (1)$$

where r_K and r_D encourage the musculoskeletal model to match the experimental kinematics and dynamics, with ω_K and ω_D being respective weights.

The former part r_K is further decomposed into three terms:

$$r_K = \omega_{angle} r_{angle} + \omega_{vel} r_{vel} + \omega_{body} r_{body} \quad (2)$$

where r_{angle} evaluates the tracking error of joint angle and is computed from the quadratic sum of different between the joint angles of the simulated agent q and those of the reference motion $\hat{q}$:

$$r_{angle} = exp\left[-\sum \|q - \hat{q}\|^2\right] \quad (3)$$

r_{vel} evaluates the tracking error of the joint angular velocity:

$$r_{vel} = exp\left[-\sum \left\|\dot{q} - \hat{\dot{q}}\right\|^2\right] \quad (4)$$

Here $\dot{q}$ is computed from q using finite difference. r_{body} is derived from the quadratic sum of tracking errors of body properties:

$$r_{body} = exp\left[-(\varphi - \hat{\varphi})^2 - (v - \hat{v})^2 - \left(h - \hat{h}\right)^2\right] \quad (5)$$

The latter part r_D encourages the agent to follow the dynamics properties of the experimental dynamics, and comprises three parts:

$$r_D = \omega_{force} r_{force} + \omega_{fiber} r_{fiber} + \omega_{grf} r_{grf} \quad (6)$$

where r_{force} and r_{fiber} evaluate the tracking error of the calculated physiological states:

$$r_{force} = exp\left[-\sum \left\|f_{fb} - \widehat{f_{fb}}\right\|^2\right] \quad (7)$$

$$r_{fiber} = exp\left[-\sum \left\|l_{fb} - \widehat{l_{fb}}\right\|^2\right] \quad (8)$$

Finally, r_{grf} evaluates the tracking error of the GRF and is defined as:

$$r_{grf} = exp\left[-\sum \left\|f_c - \hat{f}_c\right\|^2\right] \quad (9)$$

D. Training Objective

The objective of training the agent is to find the optimal policy π^* that maximizes the expected sum of rewards along the trajectory:

$$J(\pi) = E\left[\sum_{t=0}^{T} \gamma^t r_t\right] \quad (10)$$

where $\sum_{t=0}^{T} \gamma^t r_t$ is the cumulative return of a trajectory with the length of T steps, and $\gamma \in [0,1]$ is a discounting factor.

To maximize the objective function, an actor-critic structure is used, where the actor implements the policy π and the critic evaluates the policy. The aim of the actor is to maximize the Q-value, which is used to describe the expected return after taking action a_t in state s_t following the policy π:

$$Q^{\pi}(s_t, a_t) = E\left[r_t + \gamma E_{a_{t+1} \sim \pi}[Q^{\pi}(s_{t+1}, a_{t+1})]\right] \quad (11)$$

The critic is considered as a function approximator, which aims to minimize the estimation error of Q-value.

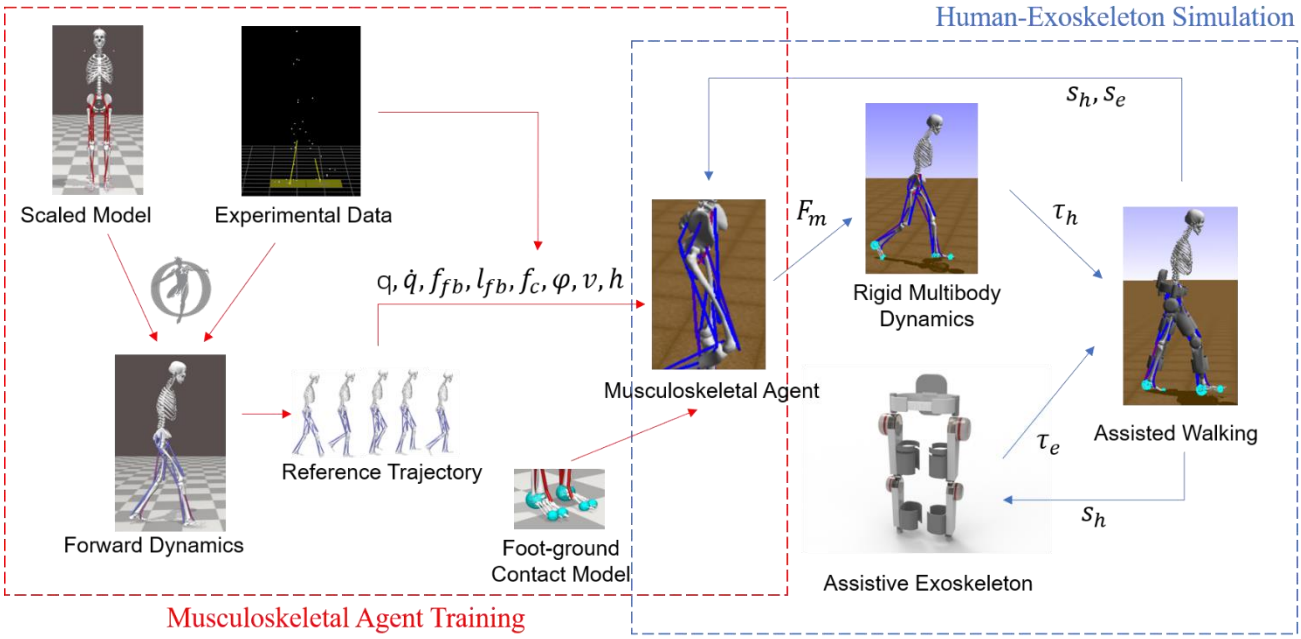


Figure 1. " The workflow of the human-exoskeleton simulation.

III. METHODOLOGY

A. Simulation Framework

The overall flow diagram of the proposed simulation method is shown in Figure 1. It includes two stages: musculoskeletal agent training stage and human-exoskeleton simulation stage.

In musculoskeletal agent training stage, a parametric muscle controller is trained to drive the musculoskeletal agent to generate walking motion. First an experimental subject conducts a walking trial, and the kinematics as well as the GRF data are recorded. Then we choose a pre-existing, available musculoskeletal model as the generic model and scale it to a subject-specific model. We use OpenSim [18], [19], an opensource biomechanical modeling and simulation software, to implement the forward dynamic simulation (see Figure 2). Note that the Computed Muscle Control (CMC) algorithm [20] calculates the muscle properties that required to drive the scaled model towards the recorded trajectory. After that, a simulated motion clip is attained, in which each frame contains musculoskeletal state s_h. This motion clip is then used as the reference trajectory to accelerate the training of the agent policy.

In human-exoskeleton simulation stage, the pre-trained agent is attached with a lower limb exoskeleton model, which is described as multiple fragments rigidly connected to the lower limbs of the body while adding torques at the revolute joints under feedback control strategy. The outputs of the musculoskeletal agent depend on the musculoskeletal state s_h and the exoskeleton state s_e observed from the human-exoskeleton system. The muscle forces F_m generated by the musculoskeletal agent are transformed into joint torques τ_h through rigid multibody dynamics. The assistive torque τ_e is generated based on the s_h. Thereupon, the human-exoskeleton agent is actuated by the cumulative torques of τ_h and τ_e. The assisted walking is simulated in a physics-based environment to test the effectiveness of the exoskeleton.

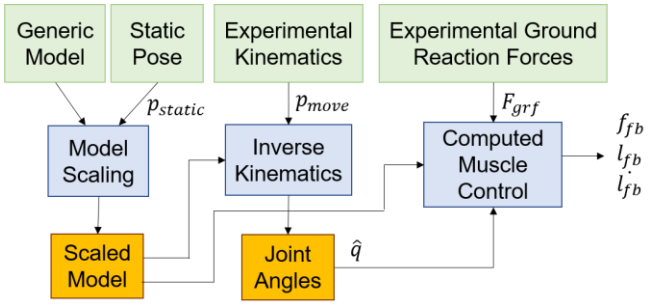
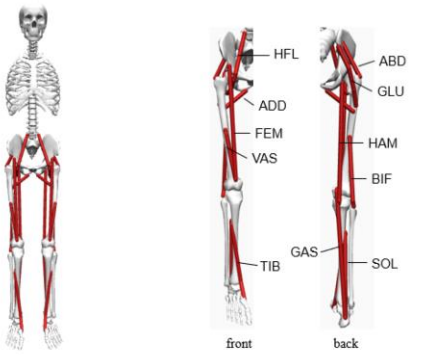


Figure 2. " The workflow for generating a muscle-driven forward dynamic simulation in OpenSim. Input data are shown in green; OpenSim tools are shown in blue; intermediate data are shown in orange.

B. Subject-Specific Model

We select the musculoskeletal model that broadly used in biomechanics community1 as the generic model. It is produced in accordance with an adult subject with the mass of about 75 kg and the height of about 1.8 m, and has 8 active degrees of freedom (4 for each leg, i.e. hip abduction/extension, knee extension and plantar flexion) to be actuated by 22 muscles (11 for each leg), as shown in Figure 3. Then we scale it to match the size of the experimental subject.



(a) the full model (b) 11 muscles for each leg

Figure 3. " Generic musculoskeletal model.

Our subject is a healthy young man who is 1.73 m tall and weigh 62 kg. To extract the geometrical profiles, 39 virtual markers are marked in the skeletal keypoints of the generic model, and the same number of reflective markers are placed on the same positions on the experimental subject. Then the motion capture system is used to collect the spatial trajectories of these markers for a static trial. Experimental marker trajectories and generic model files are imported into OpenSim to execute the scaling.

C. Neural Networks and Learning Algorithm

In deep reinforcement learning, the actor and the critic are parameterized by neural networks with parameters θ^{μ} and θ^{Q} respectively. The network structure of the actor-critic is depicted in Figure 4. The actor network is a three-layer fully-connected network with tanh activation. The critic network takes the output of actor network and the observation as inputs, and concatenates them together to generate the Q-value, with selu activation used in each layer.

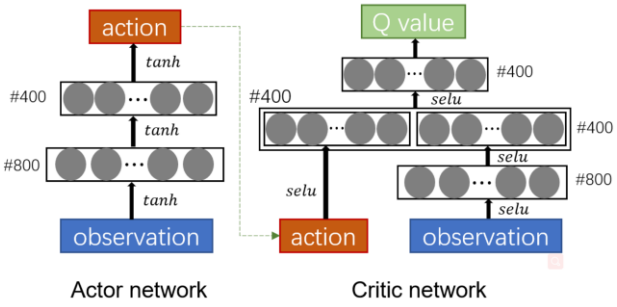


Figure 4. " The network structure of the actor-critic.

We use Deep Deterministic Policy Gradient (DDPG) [7] to train our policy. DDPG is an efficient off-policy, model-free reinforcement learning algorithm, which enhances the actor-critic approach with the success of Deep Q Network [21], and is proved to be practical for continuous action control.

The policy π is deterministic and described as function μ: S → A, and hence we get:

$$Q^{\mu}(s_t, a_t) = E\left[r_t + \gamma Q^{\mu}\left(s_{t+1}, \mu(s_{t+1})\right)\right] \quad (12)$$

In each training iteration, we first select action $a_t = \mu(s_t|\theta^{\mu}) + N_t$, where N_t is an Ornstein-Uhlenbeck process for action exploration. After a step is made, a transition (s_t, a_t, r_t, s_{t+1}) is stored into the replay buffer R. Next, N transitions from R are randomly sampled to update the networks.

The critic is optimized by minimizing the loss:

$$\mathrm{L}(\theta^{Q}) = \frac{1}{N}\sum_{i}\left(y_i - Q(s_t, a_t|\theta^{Q})\right)^2 \quad (13)$$

where $\mathrm{y_i} = \mathrm{r_t} + \gamma \mathrm{Q'}(\mathrm{s_{t+1}}, \mu'(\mathrm{s_{t+1}}|\theta^{\mu'})|\theta^{\mathrm{Q'}})$, $\theta^{\mu'}$ and $\theta^{\mathrm{Q'}}$ are parameters of target networks.

The actor policy is updated using the sampled gradient:

$$\nabla_{\theta^{\mu}}\mu \approx \frac{1}{N}\sum_{i}\nabla_a Q(s, a|\theta^{Q})|_{s=s_i, a=\mu(s_i)}\nabla_{\theta^{\mu}}\mu(s|\theta^{\mu})|_{s_i} \quad (14)$$

At last, the target networks are soft updated:

$$\theta^{\mu'} = \tau\theta^{\mu} + (1-\tau)\theta^{\mu'} \quad (15)$$

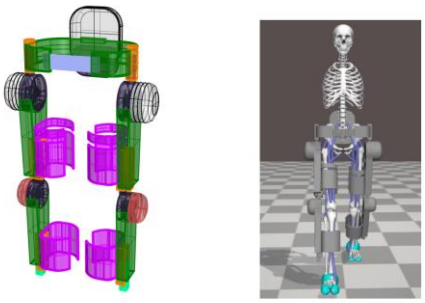
$$\theta^{Q'} = \tau\theta^{Q} + (1-\tau)\theta^{Q'} \quad (16)$$

This process iteratively continues until the maximal episode number is reached or a specified training goal is fulfilled.

D. Exoskeleton Assistance

We have designed a 4-DoFs walking assistance exoskeleton called Walking Enhancing Exoskeleton Robot (WEER), which has 4 DC motors and assists flexion/extension movement of the hip joint and flexion movement of the knee joint over the sagittal plane (see Figure. 5 (a)). Note that the length of each linkage is adjusted to match our experimental subject.

In order to attach the WEER to the musculoskeletal agent in a more convenient way, we simplify its CAD model and assemble it into five parts: a pelvis holder, two thigh modules and two shank modules. These exported 3D assemblies are attached to the musculoskeletal agent in the position of the strap via weld joint (see Figure 5 (b)).



(a) CAD model of the WEER (b) Musculoskeletal model coupled with the WEER

Figure 5. " Overview of the WEER.

Supplementary torques are applied at the hip joint and the knee joint. For simplicity of implementation as a preliminary study, we set the assistive torques as the scaled moments at each of the joints, which are calculated from the musculoskeletal state s_h via inverse dynamics:

$$\tau_e = \left[\tau_{hip}; \tau_{knee}\right] = \mathrm{f}(s_h) \quad (17)$$

This part is not included in this paper.

IV. Experimental Evaluation

A. Experimental Condition

OptiTrack motion capture system (NaturalPoint, Inc., USA) with 8 infrared cameras is used to record motion trajectory of the subject, and 2 AMTI 6-axis force plates (Advanced Mechanical Technology, Inc., USA) are used to measure GRF. The collected kinematics data as well as synchronous GRF data are processed in Motive.

The simulation is implemented in osim-rl [22], which is built upon the reinforcement learning environment OpenAI Gym [23] and integrated with OpenSim API. We implement our neural networks using PyTorch, and train the networks on an Ubuntu 16.04 server with 2 Intel Xeon(R) E5-2620 CPU and 4 Nvidia TITAN Xp GPU. The hyper-parameters used in the training are listed in Table I.

TABLE I. Hyper-Parameters for Training

Parameter	*Value*	*Description*
N	128	Number of transitions for each train
N_{rb}	1e6	Size of the replay buffer
α_π	3e-5	Learning rate of the actor
α_Q	3e-5	Learning rate of the critic
γ	0.96	Update coefficient in Bellman equation
τ	0.001	Soft update coefficient
ε	0.5	Probability of adding noise to action
ε_d	1e-6	Decay rate for noise process
σ_{ou}, θ_{ou}	0.2, 0.15	Ornstein-Uhlenbeck noise parameter

B. Experiment and Results

The motion trajectory collected from the experimental subject is tailored to a full gait cycle, starting from the heel-strike of right foot and ending with the toe-off of left foot. Since walking is a periodic motion, the reference trajectory is used cyclically. The total length of the reference trajectory is 1.42 s and the sampling rate is set to 100 Hz to lower the computational requirements.

Before training, we collect 10000 frames by sampling trajectories starting from random poses in the reference motion using random policy, in order to calculate the mean and the standard deviation of the states, which are used to normalize the network inputs during training. When agent training, each episode is terminated when the height of the pelvis is lower than 0.6 m or the maximum number of step T = 1000 is reached. The actor network and critic network are not updated until the replay buffer is full and the replay buffer is designed to be a cyclic queue which stores latest N_{rb} transitions.

The learning curve of musculoskeletal agent learning to walk is shown in Figure. 6. The performance is calculated as the average return over consecutive 50 episodes. We record outcomes from 3 training runs using random seeds, and the performance appears consistent across different runs. It takes approximately 4 hours wall clock time, corresponding to 250 episodes to reach the peak with a total reward of near 16.

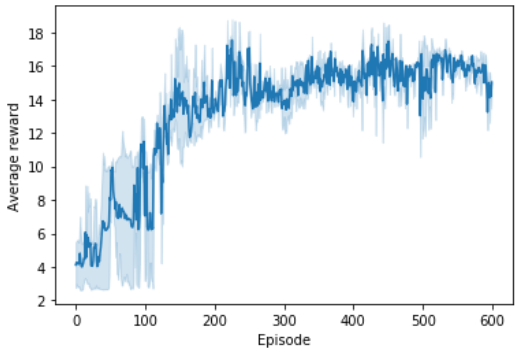


Figure 6. Learning curve for training the agent.

We conduct 2 experimental trials on the trained agent. One is walking solely, and another is walking with the assistance of the WEER. Both trials last a full gait cycle. To evaluate the assistance effectiveness, we record the normalized activations of 4 representative muscles of the right leg, which play major roles in walking motion (ADD, GLU for hip flexion/extension and GAS, VAS for knee flexion). Figure. 7 shows the result of comparing the muscle activation during walking under different conditions. Muscle activations of ADD and GLU are dramatically reduced, which validates the assistance of the WEER to some extent. Whereas, those of GAS and VAS are not so, and we believe this is because the knee motion is not so appropriate for being assisted by supplementary torques as that for the hip motion.

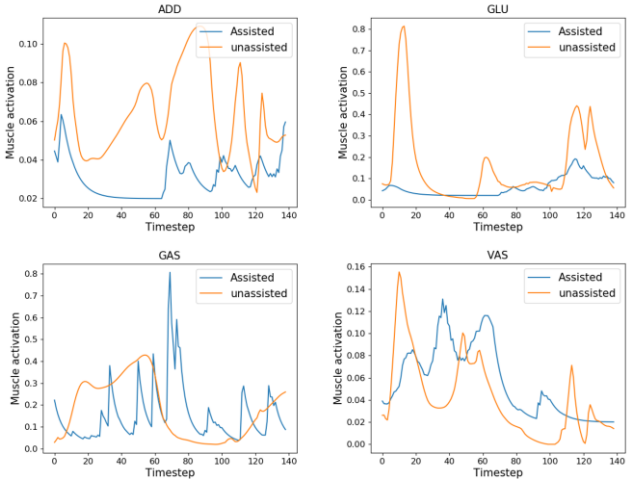


Figure 7. Muscle activations of 4 muscles (ADD, GLU, GAS and VAS).

C. Discussion

The assistance effectiveness of the WEER has not been comprehensively evaluated, since energy expenditures of agent walking are not proved to be significantly reduced. Note that the reliability of this analysis lies in the accuracy of the motion simulation. However, the performance of the musculoskeletal agent walking is now far from idealization, frankly, which is the bottleneck of the human-exoskeleton simulation.

On the other hand, although the agent with or without the exoskeleton is formulated in a uniform way, the dynamics followed in the assisted walking stage are not the same as that in the agent training stage, since the exoskeleton also has mass and inertia. And this may be another reason for the poor performance of agent assisted walking.

V. Conclusion and Future Work

This paper proposes a simulation framework for emulating the human walking with the assistance of lower limb exoskeleton. Experimentally, we collect the reference trajectory of a subject, train the subject-specific musculoskeletal agent to produce walking motion, and then simulate the assisted walking with a powered lower limb exoskeleton. The experimental results show the feasibility of the proposed method.

In future work, we are about to (1) train the musculoskeletal agent to walk at different speeds and across uneven terrain with the help of the scaled reference trajectory or advanced DRL training techniques; (2) develop more sound control strategy for exoskeleton; (3) explicitly formulate the exoskeleton noumenon considering the impact on human motion due to the gravity and inertia of the exoskeleton.

Acknowledgment

We thank Yufei Huang, Yongchuan Tang and Xianfu Zhang for helpful discussions. This research is funded by the National Key Research and Development Program of China under Grant 2017YFD0700102.

References

[1] Park Y L, Chen B, Pérez-Arancibia N O, et al. Design and control of a bio-inspired soft wearable robotic device for ankle–foot rehabilitation[J]. Bioinspiration & biomimetics, 2014, 9(1): 016007.

[2] K. Elissa, “Title of paper if known,” unpublished. Aguilar-Sierra H, Lopez R, Yu W, et al. A lower limb exoskeleton with hybrid actuation[C]//5th IEEE RAS/EMBS International Conference on Biomedical Robotics and Biomechatronics. IEEE, 2014: 695-700.

[3] Bartenbach V, Gort M, Riener R. Concept and design of a modular lower limb exoskeleton[C]//2016 6th IEEE International Conference on Biomedical Robotics and Biomechatronics (BioRob). IEEE, 2016: 649-654.

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

[5] Song S, Geyer H. Generalization of a muscle-reflex control model to 3d walking[C]//2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC). IEEE, 2013: 7463-7466.

[6] Song S, Geyer H. A neural circuitry that emphasizes spinal feedback generates diverse behaviours of human locomotion[J]. The Journal of physiology, 2015, 593(16): 3493-3511.

[7] Lillicrap T P, Hunt J J, Pritzel A, et al. Continuous control with deep reinforcement learning[J]. arXiv preprint arXiv:1509.02971, 2015.

[8] Schulman J, Levine S, Abbeel P, et al. Trust region policy optimization[C]//International conference on machine learning. 2015: 1889-1897.

[9] Schulman J, Wolski F, Dhariwal P, et al. Proximal policy optimization algorithms[J]. arXiv preprint arXiv:1707.06347, 2017.

[10] Peng X B, Abbeel P, Levine S, et al. Deepmimic: Example-guided deep reinforcement learning of physics-based character skills[J]. ACM Transactions on Graphics (TOG), 2018, 37(4): 1-14.

[11] Merel J, Tassa Y, TB D, et al. Learning human behaviors from motion capture by adversarial imitation[J]. arXiv preprint arXiv:1707.02201, 2017.

[12] Arnold E M, Hamner S R, Seth A, et al. How muscle fiber lengths and velocities affect muscle force generation as humans walk and run at different speeds[J]. Journal of Experimental Biology, 2013, 216(11): 2150-2160.

[13] Hicks J L, Uchida T K, Seth A, et al. Is my model good enough? Best practices for verification and validation of musculoskeletal models and simulations of movement[J]. Journal of biomechanical engineering, 2015, 137(2).

[14] Torricelli D, Cortés C, Lete N, et al. A subject-specific kinematic model to predict human motion in exoskeleton-assisted gait[J]. Frontiers in neurorobotics, 2018, 12: 18.

[15] Uchida T K, Seth A, Pouya S, et al. Simulating ideal assistive devices to reduce the metabolic cost of running[J]. PloS one, 2016, 11(9): e0163417.

[16] Lee G, Kim J, Panizzolo F A, et al. Reducing the metabolic cost of running with a tethered soft exosuit[J]. Sci. Robot, 2017, 2(6): 6708-31.

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

[18] Delp S L, Anderson F C, Arnold A S, et al. OpenSim: open-source software to create and analyze dynamic simulations of movement[J]. IEEE transactions on biomedical engineering, 2007, 54(11): 1940-1950.

[19] Seth A, Hicks J L, Uchida T K, et al. OpenSim: Simulating musculoskeletal dynamics and neuromuscular control to study human and animal movement[J]. PLoS computational biology, 2018, 14(7): e1006223.

[20] Thelen D G, Anderson F C. Using computed muscle control to generate forward dynamic simulations of human walking from experimental data[J]. Journal of biomechanics, 2006, 39(6): 1107-1115.

[21] Mnih V, Kavukcuoglu K, Silver D, et al. Human-level control through deep reinforcement learning[J]. nature, 2015, 518(7540): 529-533.

[22] Kidziński Ł, Mohanty S P, Ong C F, et al. Learning to run challenge: Synthesizing physiologically accurate motion using deep reinforcement learning[M]//The NIPS'17 Competition: Building Intelligent Systems. Springer, Cham, 2018: 101-120.

[23] Brockman G, Cheung V, Pettersson L, et al. Openai gym[J]. arXiv preprint arXiv:1606.01540, 2016.