DDQN-Learning of Hill-Type Musculoskeletal Arm Model for Elbow Motor Control

Mohammad-Reza Sayyed Noorani
dept. Mechatronics Engineering
University of Tabriz
Tabriz, Iran
smrs.noorani@tabrizu.ac.ir

Abbas Jafarpour Mahalleh
dept. Mechatronics Engineering
University of Tabriz
Tabriz, Iran
jafarpourabbas@googlemail.com

Kimiya Khojand
dept. Mechatronics Engineering
University of Tabriz
Tabriz, Iran
kimiya.khojand1999@gmail.com

***Abstract*— This study aimed to develop a model-based reinforcement learning (RL) framework designed to partially emulate central nervous system (CNS) learning processes for goal-directed motor control. The RL model, implemented using a double deep Q-learning (DDQN) algorithm, interacting with a biomechanical arm model served as the simulated environment. The environment comprised a Hill-type musculoskeletal representation of the biceps brachii and triceps brachii muscles, enabling elbow flexion–extension over a range of 10–135°. Within this setup, the RL agent received state information, including elbow joint angle and velocity, from the environment and generated muscle activation signals as control outputs. These signals acted on the Hill-based biomechanical model, allowing the agent to learn reaching toward specified target through iterative episodes. To validate biomechanical realism, forward dynamics simulations were performed in OpenSim using a customized arm model driven by the RL-generated excitations. Results demonstrated that the agent successfully acquired stable and biologically plausible motor strategies.**

***Keywords*— Double Deep Q-learning, Elbow Flexion Control, Hill-Type Musculoskeletal Model, Muscle Activation, OpenSim.**

I. Introduction

Accurate motor control is essential for effective interaction with the environment and others. It relies on the integration of sensory information by the central nervous system (CNS) and its translation into coordinated muscle activity, ultimately driving body motion. This motion emerges from the interplay between the musculoskeletal and neural systems. Recent neuromechanical modeling and computational studies have further emphasized the importance of such interactions [1].

Machine learning (ML) and reinforcement learning (RL) have revolutionized musculoskeletal (MSK) biomechanics by addressing limitations in traditional inverse dynamics, such as sensitivity to noise in electromyography or kinematics data, computational expense, and the need for subject-specific calibrations. These techniques enable efficient estimation of joint moments, reaction forces, and motion trajectories, with applications in rehabilitation, gait analysis, and functional electrical stimulation (FES) control. Integrating ML/RL with MSK simulators like OpenSim facilitates simulation of complex, high-dimensional systems, including fatigable muscles and pathological conditions [2-7].

Wu et al. [2] pioneered RL for joint moment estimation from EMG or kinematics, employing PPO on forearm/hand data from six subjects. RL agents predicted moments driving forward dynamics, achieving 84–98% correlations with measured kinematics using only 15 s of data, outperforming traditional methods in noise robustness and free-movement generalization. The study demonstrated that RL can serve as an effective substitute for conventional inverse dynamics in biomechanics research and EMG-driven human–machine interface applications.

Denizdurduran et al. [3] proposed a pipeline addressing the scarcity of human motion-capture experiments by using optimal-control trajectories as reference signals for RL, enabling musculoskeletal models to learn human-like behavior. This approach provides a basis for in-silico studies of human movement and can be extended to explore adaptive and personalized upper-arm rehabilitation with assistive robots. Jang et al. [4] highlighted that while physics-based musculoskeletal models provide valuable insights into estimating unmeasurable variables such as muscle forces and joint moments, their computational complexity limits real-time applicability. In contrast, data-driven methods offer speed and simplicity but lack physiological interpretability. To address these limitations, they proposed a physics-informed deep learning framework that integrates domain knowledge as soft constraints within a CNN-based model. Using sEMG to predict muscle forces and joint kinematics, validated on benchmark and self-collected datasets, the framework showed robust and effective performance. Kumar et al. [5] emphasized that while exoskeletons and rehabilitation systems rely on accurate estimation of joint dynamics, conventional ML approaches often fail to generalize across dynamic movements. To overcome this, they proposed the Physics-informed Gated Recurrent Network (PiGRN), which integrates biomechanical constraints into a GRU-based framework for predicting multi-joint kinematics and torques from sEMG. Validated on elbow flexion–extension tasks under different loads, PiGRN achieved high accuracy (RMSE 4–11% and correlations 0.87–0.98), highlighting its potential for real-time rehabilitation and assistive applications. Song et al. [6] highlighted the challenge of modeling human motor control, noting that traditional neuromechanical simulations reproduce basic locomotion but struggle with complex, higher-level controls. They introduced the "Learn to Move" competition, leveraging deep reinforcement learning to generate novel, physiologically plausible human motions, demonstrating RL's potential in advancing neuromechanical simulations for biomechanics and rehabilitation research. Dashkovets et al. [7] addressed the challenge of controlling robotic leg prostheses and exoskeletons by developing an efficient two-layer Q-learning algorithm with k-d trees for continuous action spaces. Their approach employed a reward model evaluating the similarity of muscle activation between the agent and human state-to-action pairs, and utilized a high-dimensional physics-based musculoskeletal simulation for training and evaluation. This study demonstrates a promising step toward bioinspired controllers for robotic prosthetic legs and exoskeletons, with potential for real-time embedded applications, and sets the stage for future improvements in prediction accuracy and extension to other locomotor tasks. Abreu et al. [8] examined RL-based control of FES for upper-limb rehabilitation and found that performance is more affected by flexor weakness and passive resistance than by tendon or muscle stiffness, emphasizing the need to reduce passive resistance and strengthen antagonistic muscles.

Despite notable advances in musculoskeletal modeling and machine learning, which have been extensively applied to study human motor control and inform rehabilitation or assistive technologies, a critical gap remains in developing computational frameworks that jointly capture both of the biomechanical realism of muscle-driven movement and the adaptive learning capability of the central nervous system. Physics-based models offer detailed insights into muscle dynamics but are computationally prohibitive for real-time applications, while data-driven methods provide efficiency but often neglect underlying neuromechanical constraints. Recent RL approaches show promise for modeling human motor control due to their adaptability; however, they typically lack physiological interpretability and have largely focused on lower-limb locomotion or simplified kinematic mappings without explicitly representing muscle activation dynamics. This limitation is particularly pronounced in upper-limb tasks, where fine motor control critically depends on realistic interactions between antagonistic muscles. To address this gap, the present study proposes an integrated framework by (1) developing a Hill-type two-muscle musculoskeletal model of the elbow joint, (2) designing a RL-based method namely Double Deep Q-learning (DDQN) controller to learn muscle activation strategies for goal-directed elbow movement, and (3) validating the learned control through biomechanical simulations in OpenSim.

II. HILL'S TYPE MUSCULOSKELETAL ARM MODEL

As illustrated in Fig. 1, a simplified two-muscle model of the elbow joint is developed, comprising two primary muscles responsible for flexion and extension: the biceps brachii and the triceps brachii. The contraction dynamics of each muscle are represented using a Hill-type model. According to [9], the modeling of each muscle consists of two main components: musculoskeletal geometry analysis, which determines the length–angle relationships and joint moment arms, and muscle force computation based on the contractile properties of the Hill-type formulation. These components are detailed in the following subsections, and the complete model is ultimately implemented as a computational script in Python.

TABLE I. VALUES OF MUSCLES PARAMETERS IN THE MODEL

Muscles	F_o^M	l_o^M	l_s^T	v_{max}^M	φ_o	d
BIC	300 N	13 cm	20 cm	85 cm/s	0.0	8.2 cm
TRC	300 N	19 cm	14 cm	95 cm/s	0.0	4.1 cm

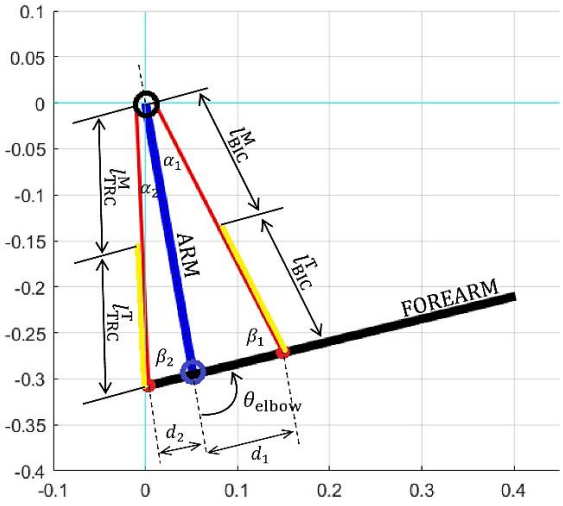

Fig 1. The two-muscle musculoskeletal model of the elbow joint

A. Muscle-Tendon Length

To analyze the musculoskeletal geometry, some simplifying assumptions are adopted. First, the shoulder joint is assumed to remain fixed, and consequently, the upper arm is stationary. Under this assumption, the model's kinematics reduce to a single degree of freedom at the elbow. Second, the tendons of the muscles are assumed to be rigid during muscle length changes. Although this physiological assumption is valid for short tendons, long tendons are affected by elastic dynamics; nevertheless, to reduce computational cost, while preserving architectural and physiological consistency, tendon lengths are considered constant and equal to their slack lengths [9]. Additionally, although the pennation angles of the biceps and triceps muscles are relatively small (typically 5–15°), they are assumed to be zero in this model. This simplification is justified because such small angles have a negligible effect on the overall force transmission along the muscle–tendon unit. Moreover, the contribution of passive element is neglected in modeling. Finally, the forward dynamics simulations neglect the effects of gravity. Under these assumptions and based on the musculoskeletal geometry shown in Fig. 1, the length of each muscle can be derived using the law of cosines as follows (The notations used here correspond to those shown in Fig. 1):

$$\begin{aligned} l_{\mathrm{BIC}}^{\mathrm{MT}} &= l_{\mathrm{ARM}}^2 + d_1^2 + 2d_1 l_{\mathrm{ARM}} \cos\theta \\ l_{\mathrm{TRC}}^{\mathrm{MT}} &= l_{\mathrm{ARM}}^2 + d_2^2 - 2d_2 l_{\mathrm{ARM}} \cos\theta \end{aligned} \tag{1}$$

Then, under the rigid tendon assumption, the muscle fiber length can be expressed as:

$$\begin{aligned} l_{\mathrm{BIC}}^{\mathrm{M}} &= l_{\mathrm{BIC}}^{\mathrm{MT}} - l_{\mathrm{BIC}}^{\mathrm{T-s}} \\ l_{\mathrm{TRC}}^{\mathrm{M}} &= l_{\mathrm{TRC}}^{\mathrm{MT}} - l_{\mathrm{TRC}}^{\mathrm{T-s}} \end{aligned} \tag{2}$$

where $l_{\mathrm{MUS}}^{\mathrm{T-s}}$ and stands for the slack length of the tendon.

In selecting the numerical values for the parameters of the musculoskeletal model, both geometric scaling in accordance with anatomical dimensions and key physiological facts must be considered. Notably, the biceps brachii reaches its optimal fiber length—and therefore its maximum isometric force—when the elbow is within the angular range of 80–100°, whereas the triceps achieves this in the range of 90–110°. Thus, by setting the upper-arm length to 32 cm, the resulting parameter values were determined as summarized in Table 1. Additional calculations verifying that these selected values satisfy the aforementioned physiological constraints are provided in the Appendix.

B. Muscle-Tendon Force

The Hill-type model represents the calculation of the force generated through muscle activation. As illustrated in Fig.2, schematically, this model consists of two main components: the muscle fiber and the tendon. The muscle fiber attaches to the tendon at an angle known as the pennation angle (φ). The muscle fiber itself comprises two elements: an active element, responsible for generating controlled muscle force, in parallel with a passive element, which represents the reactive force resulting from the stretching of connective tissues as well as the titin protein during muscle elongation [9-10]. According to the Hill model, the active force generated in the muscle fiber is the product of the muscle activation level a, the active force–length relationship, $f^L(\tilde{l}^M)$ depicted in Fig. 3-A, and the active force–velocity relationship, $f^V(\tilde{v}^M)$ depicted in Fig. 3-B. Thus, the total muscle fiber force is then calculated as the sum of the active and passive, $f^{PE}(\tilde{l}^M)$, fiber forces and is expressed w.r.t unidimensional quantities as follows:

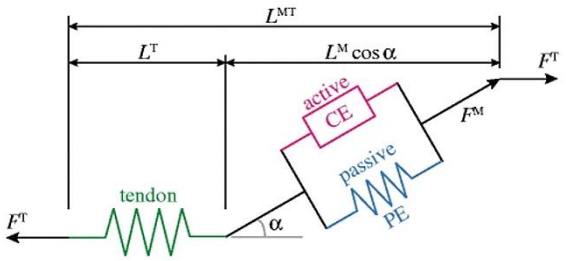

Fig 2. Shematic of the Hill-type model used for muscle force calculation [10]

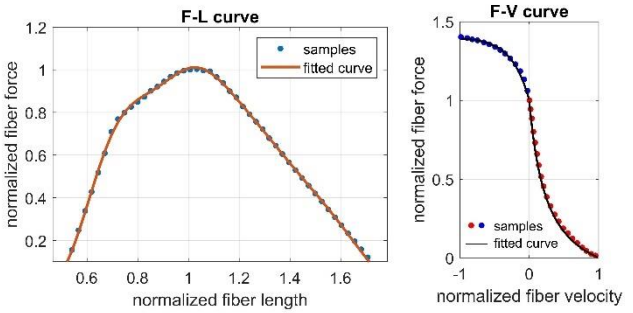

Fig 3. The normalized force–length and force–velocity curves, sampled from data in [9], along with their regenerated fitted curves

$$F^M = F_o^M\left[a\, f^L(\tilde{l}^M) f^V(\tilde{v}^M) + f^{PE}(\tilde{l}^M)\right] \quad (1)$$

The tendon force transmitted to the bone is also then calculated w.r.t the pennation angle, φ, as follows:

$$F^T = F^M \cos\varphi \quad (2)$$

where, F_o^M represents the maximum isometric force, while f^L and f^V denote the normalized active force components (relative to F_o^M) as functions of the normalized fiber length $\tilde{l}^M$ and normalized contraction velocity $\tilde{v}^M$, respectively.

Notably, in the Hill-type model, each muscle–tendon unit is characterized by five anatomical and physiological parameters, which are named in Table I (highlighted in bold). Hence, the normalized fiber length $\tilde{l}^M$ and velocity $\tilde{v}^M$ are expressed relative to the muscle-specific parameters l_o^M and $v_{\max}^M$, respectively. By knowing the normalized fiber quantities, the normalized active force components of $f^L$and f^V are read from the normalized force–length and force–velocity curves given by Fig. 3.

It is worth to mention that the curves depicted in Fig. 3 were generated by sampling data from the reference plots reported in [9] and then fitting the corresponding functions. This approach also improves the computational efficiency of forward dynamics simulations of muscle-driven systems, which is essential during RL agent training. Thus, the fiber length–force curve was approximated using a five-component Fourier series, as follows:

$$f^L(\tilde{l}^M) = \frac{a_0}{2} + \sum_{n=1}^{5} a_n \cos n\omega_0 \tilde{l}^M + b_n \sin n\omega_0 \tilde{l}^M \quad (3)$$

where, $a_0 = 1.10$, $a_n = \{-4.3, 6.0, -4.6, 2.0, -0.6\} \times 10^{-2}$, $b_n = \{-47.7, 7.2, 1.5, -1.0, 0.1\} \times 10^{-2}$, and $\omega_0 = 4.43$.

The fiber contraction velocity ($\tilde{v}^M$)–force curve was also computed using the following two-term relationship:

$$f^L(\tilde{v}^M) = \begin{cases} (b - a\tilde{v}^M)/(b + \tilde{v}^M) & \tilde{v}^M \geq 0 \\ s + (1 - s)\exp(|\tilde{v}^M|/b) & \tilde{v}^M < 0 \end{cases} \quad (4)$$

where, $a = b = 0.25$, and $s = 1.4$ is concentric coefficient.

III. Double Deep Q-Learning Network Design

Reinforcement learning trains agents to make sequential decisions. While Deep Q-Networks (DQN) gained human-level performance on Atari games, they suffered from overestimation bias caused by the max operator in Q-learning. To address this, Double Deep Q-Networks (DDQN) were introduced, significantly improving both performance and training stability. DDQN maintains two separate Q-functions: one for selecting actions and the other for evaluating them, thereby reducing over-optimism. As demonstrated in [11], this concept can be adapted to deep RL by integrating it with DQN's architecture. In DDQN, the online Q-network chooses the action that maximizes the Q-value, while the target Q-network evaluates the value of that action, resulting in more accurate Q-value estimates. The target Q value at time t is acquired as follows:

$$Y_t^{DDQN} = R_{t+1} + \gamma \cdot Q(s_{t+1}, \text{argmax}\, Q(s_{t+1}, a'; \theta); \theta^-) \quad (5)$$

where r_t, γ, s_{t+1}, θ, θ^- indicate rewards at time step t, discount factor, next state, parameters of the online network and the parameters of the target network, respectively.

In the training process, the agent starts by taking random actions with a probability of ε which decreases with a decay rate of 0.998, starting from 1 (exploration) in the beginning of training process, decreasing to a minimum of 0.01 (exploitation) as the training progresses. This gives the agent a more diverse experience set to learn from.

Here, the network takes angular states of the forearm $(\theta, \dot{\theta})$ as inputs. It is consisted of two fully connected hidden layers with 64 neurons, and a softmax output with 5 neurons which determine the Q value of the best action for s_{t+1}. The meaning of each action outputted by the network is as follows:

- Action 0: Do nothing
- Action 1: Increase the biceps activation by 0.5%
- Action 2: Decrease the biceps activation by 0.5%
- Action 3: Increase the triceps activation by 0.5%
- Action 4: Decrease the triceps activation by 0.5%

This approach for interpreting the network output was employed to discretize the continuous action space of the designed environment. The network was trained with a learning rate of 0.0002, soft updating every 4 time steps [12], to help a more robust convergence towards a local optimal point. The network updating proceeded up to a point of reaching 80%-win rate in previous 100 episodes (This point is indicated by a red vertical line in plots of Figs. 4 and 5).

In the custom environment, initially, agent starts from a random position and tries to move the arm to a target angle. In the following charts, training has been deployed with a target arm angle of 110 degrees. If the agent manages to stay within a ±5 degrees' tolerance of the target angle for 10 time steps, maintaining an angular velocity below ±1.57 rad/s, the agent has reached a successful iteration, for which, it receives +50 points; however, if the arm's angle gets out of the acceptable arm angle range 0 to 135 degrees, or the episode's time steps exceeds 1000, the agent is penalized by receiving a reward of −10 points. The distance and velocity reward are calculated by $-0.1\,(\theta - \theta_{\text{target}})^2$ and $-0.1\,(\dot{\theta})^2$ relations, respectively; penalizing the agent for every step that it is not in the target θ and $\dot{\theta}$ range. Furthermore, agent gets +1 point for every time steps that it is inside the suitable θ and $\dot{\theta}$ range, rewarding the agent to stay in the suitable range.

IV. Results and Discussion

This section presents 3 categories of results: (1) an analysis of the learning process, (2) an evaluation of the RL-agent's performance within the musculoskeletal model during a representative episode, and (3) a validation of the outcomes through muscle-driven simulations conducted in OpenSim.

A. Learning Process Analysis

Figure 4 shows the evolution of the relative success ratio across episodes, both during training (before the red dashed line) and after training was stopped (after the red dashed line), where the ratio of successful (or wined) episodes is presented for a moving window of 100 episodes (blue) and for the entire training duration (green). As mentioned earlier, the learning (i.e., updating the weights of the RL model) is stopped once the win ratio over a 100-episode horizon exceeds 80%, preventing instability and loss of acquired experience due to random exploration. As observed, after approximately 4500 episodes the learning process reaches this threshold and subsequently fluctuates around it, whereas the win ratio over all episodes increases monotonically as the influence of early episodes diminishes.

Figure 5 illustrates the cumulative values of each continuous reward component—those allocated step by step during an episode—namely distance, velocity, and in-range rewards, over successive episodes. The sparse distribution of scatter points at low reward values, followed by their concentration at higher values after the red dashed line, reflects the successful learning of the RL agent.

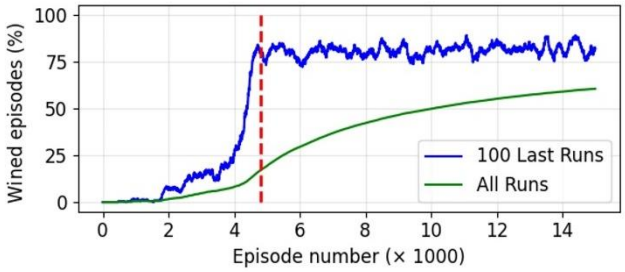

Fig. 4 Evolution of the relative success ratio across episodes during training (before the red dashed line) and after training was stopped (after the red dashed line)

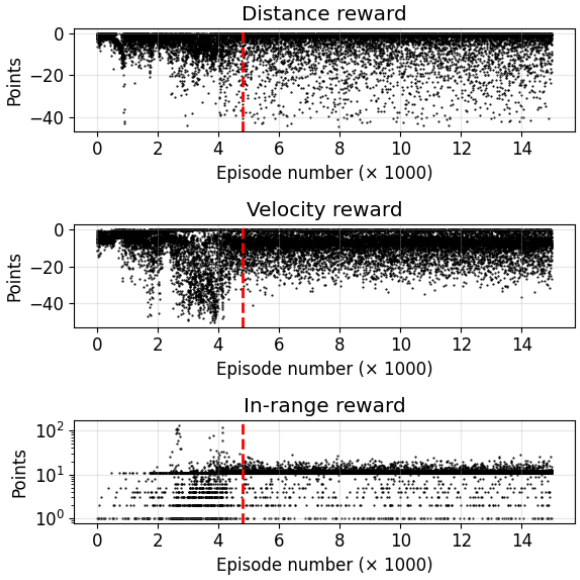

Fig. 5 The evolution of the three reward components across episodes

Finally, the frequency distribution of action selections across successive episodes is illustrated in Fig. 6. It is clearly observed that, after learning, action 0 (i.e., 'Do nothing' action) occurs with the highest frequency. This indicates the agent's tendency to avoid frequent oscillations in muscle activation signals, which could otherwise lead to unstable movements.

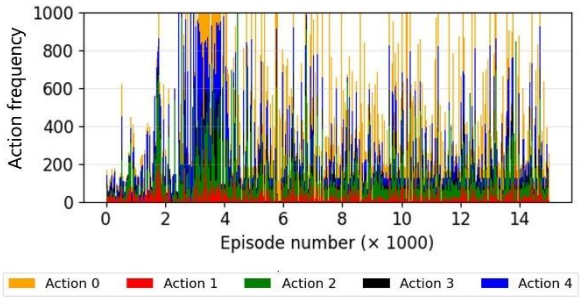

Fig. 6 Frequency of the actions executed in each episode

B. Evaluation of a Sampled Post-Trained Episode

Over 10,000 episodes following the termination of training were evaluated to confirm the sustained 80% success ratio achieved during the learning process, as illustrated in Fig. 4. Among them, a successful episode was randomly selected as a representative sample to demonstrate the musculoskeletal model's capability in controlling the arm from its initial configuration to the target. It should be noted that the initial conditions for each episode were assigned randomly.

Figure 7 presents the time series of muscle activations, which were generated by the RL agent after training. Based on the Hill-type model and the RL-generated activations, the length–force and velocity–force components of the active muscle force for both muscles are illustrated in Figs. 8 and 9, respectively. Furthermore, the profiles of active muscle force for both the biceps (BIC) and triceps (TRC) muscles are presented in Fig. 10. By multiplying the muscle forces by their corresponding moment arms and summing them algebraically, the net joint torque around the elbow has been calculated, which its profile depicted in Fig. 11.

At this stage, by applying the joint torque to the single-degree-of-freedom dynamic model and performing double integration according to Euler's law, the angular velocity and displacement trajectories of the elbow joint are obtained, as shown in Fig. 12. For a clearer visualization of the movement, a series of snapshots depicting the kinematic evolution of the elbow joint are provided in Fig. 13. The results confirm the RL agent's success in both reaching the target position and stably maintaining the hand at that location.

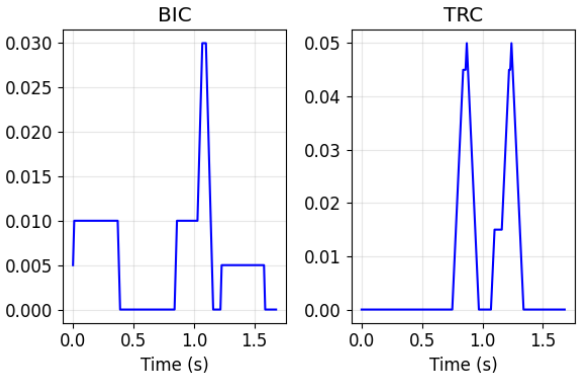

Fig. 7 The generated muscle activations in the sampled post-trained episode

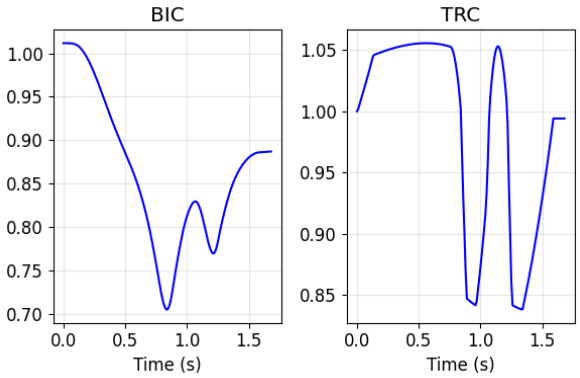

Fig. 8 The fiber length coefficient in the sampled post-trained episode

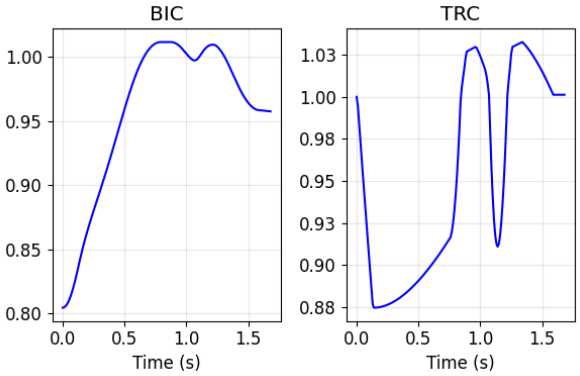

Fig. 9 The fiber velocity coefficient in the sampled post-trained episode

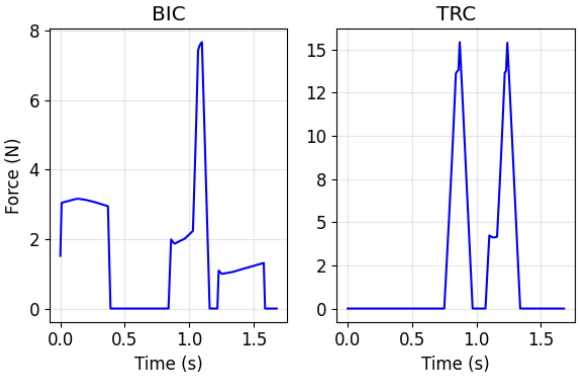

Fig. 10 Hill's model-based muscle force in the sampled post-trained episode

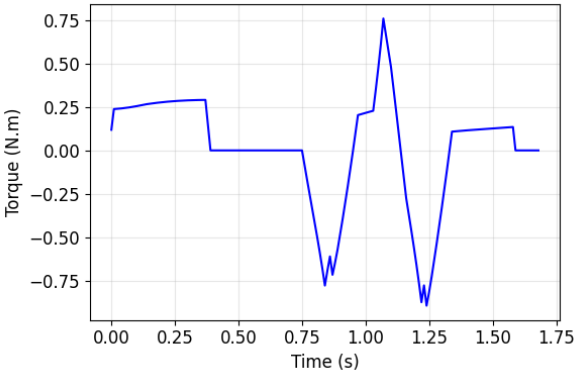

Fig. 11 The elbow joint torque in the sampled post-trained episode

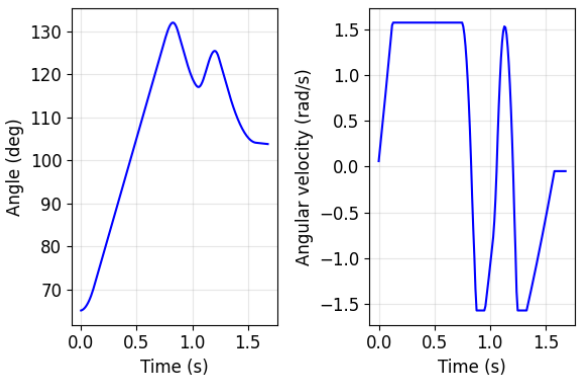

Fig. 12 Kinematic response of the elbow in the sampled post-trained episode

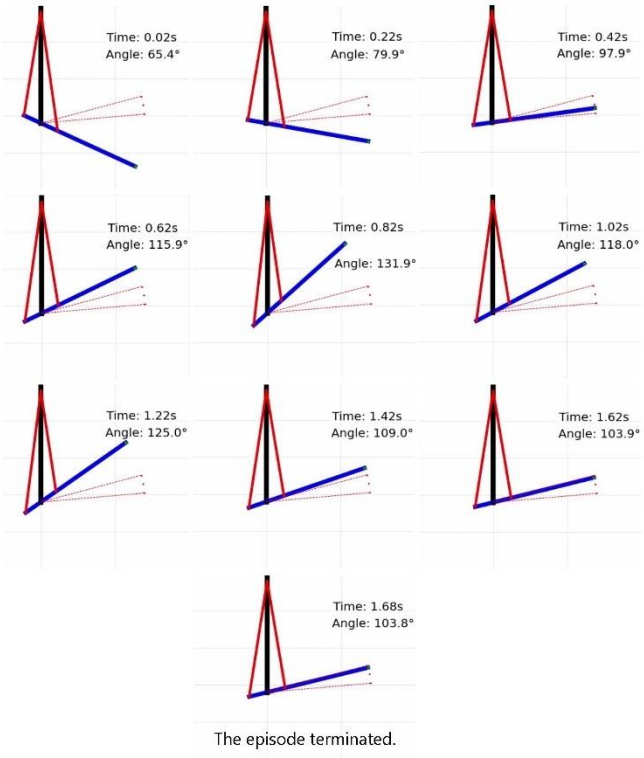

Fig. 13 Snapshots of the movement of the sampled post-trained episode

C. OpenSim Muscle Driven Validation

To validate the musculoskeletal model, a two-muscle elbow joint customized-model was implemented in OpenSim, and a muscle-driven simulation using the RL-generated activations was conducted, as shown in Fig. 14.

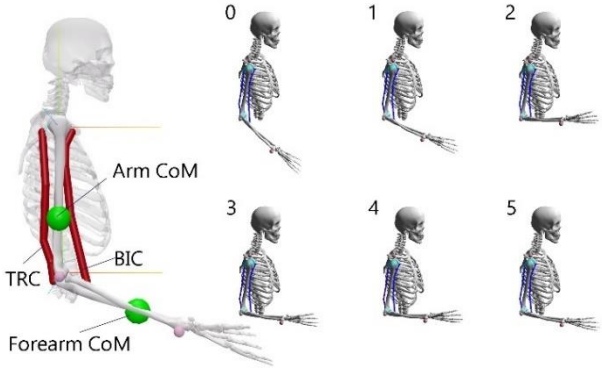

Fig. 14 OpenSim arm model and snapshots of the muscle-driven simulation

In this customized model, derived from tailoring the existing OpenSim arm26 model, the muscle-specific parameters were assigned proportional with those used in the Hill-type model applied during the learning process. Figure 15 presents the comparison of kinematic trajectories between the two models. While an exact overlap of the trajectories is not expected, the substantial consistency between them—especially at the peaks and troughs—provides strong evidence supporting the validity the validity of our developed musculoskeletal model, which is simple, accurate, and computationally efficient.

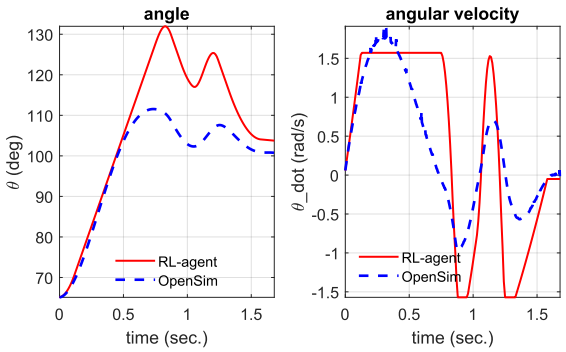

Fig. 15 Comparison of elbow kinematic response between the OpenSim and our custom-built musculoskeletal models for sampled post-trained episode

V. Conclusion

In this study, a two-muscle Hill-type musculoskeletal model of the elbow joint was developed and integrated with a Double Deep Q-Network reinforcement learning framework to investigate goal-directed motor control. The RL agent successfully learned to generate physiologically plausible muscle activation patterns to drive the elbow from various initial positions to target angles. The learned control strategies were validated through forward dynamics simulations in OpenSim using a customized arm model, demonstrating close agreement in kinematic trajectories. These results confirm that the proposed model is accurate, simple, and computationally efficient, capable of reproducing stable and realistic elbow movements. Overall, the framework provides a practical approach for studying neuromechanical control, offering potential applications in rehabilitation, assistive robotics, and human–machine interface design.

Authors' Contribution — MSN: Conceptualization, Developing Hill's type musculoskeletal arm model, OpenSim. AJM: Python scripting, developing and training the RL-agent. KKh: Literature review, original draft and final revise.

References

[1] Bruel, A., Abadía, I., Collin, T., Sakr, I., Lorach, H., Luque, N.R., Ros, E. and Ijspeert, A., 2024. The spinal cord facilitates cerebellar upper limb motor learning and control; inputs from neuromusculoskeletal simulation. *PLOS Computational Biology*, *20*(1), p.e1011008.

[2] Wu, W., Saul, K.R. and Huang, H., 2021. Using reinforcement learning to estimate human joint moments from electromyography or joint kinematics: An alternative solution to musculoskeletal-based biomechanics. *Journal of biomechanical engineering*, *143*(4), p.044502.

[3] Denizdurduran, B., Markram, H. and Gewaltig, M.O., 2022. Optimum trajectory learning in musculoskeletal systems with model predictive control and deep reinforcement learning. *Biological cybernetics*, *116*(5), pp.711-726.

[4] Zhang, J., Zhao, Y., Shone, F., Li, Z., Frangi, A.F., Xie, S.Q. and Zhang, Z.Q., 2022. Physics-informed deep learning for musculoskeletal modeling: Predicting muscle forces and joint kinematics from surface EMG. IEEE Transactions on Neural Systems and Rehabilitation Engineering, 31, pp.484-493.

[5] Kumar, R., Gupta, A., Muthukrishnan, S.P., Kumar, L. and Roy, S., 2024. sEMG-Driven Physics-Informed Gated Recurrent Networks for Modeling Upper Limb Multi-Joint Movement Dynamics. *arXiv preprint arXiv:2408.16599*.

[6] Song, S., Kidziński, Ł., Peng, X.B., Ong, C., Hicks, J., Levine, S., Atkeson, C.G. and Delp, S.L., 2021. Deep reinforcement learning for modeling human locomotion control in neuromechanical simulation. *Journal of neuroengineering and rehabilitation*, *18*(1), p.126.

[7] Dashkovets, A. and Laschowski, B., 2024, September. Reinforcement learning for control of human locomotion in simulation. In *2024 10th IEEE RAS/EMBS International Conference for Biomedical Robotics and Biomechatronics (BioRob)* (pp. 43-48). IEEE.

[8] Abreu, J., Crowder, D.C. and Kirsch, R.F., 2023. The impact of hill-type actuator components on the performance of reinforcement learning controllers to reverse upper-limb paralysis. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, *31*, pp.2883-2892.

[9] Uchida, T. K., & Delp, S. L. (2021). 'Biomechanics of movement: the science of sports, robotics', and rehabilitation. Mit Press.

[10] Hassanzadeh Kh., Sh., and Sayyed Noorani, M., 2025. 'OpenSim/MATLAB interface to musculoskeletal-based simulations for control of a motion assistive exoskeleton robot in elbow flexion/extension. Iranian Journal of Biomedical Engineering.' 18(3), pp. 211-220. [in Persian] doi: 10.22041/ijbme.2025.2048707.1942

[11] Van Hasselt, H., Guez, A. and Silver, D., 2016, March. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence* (Vol. 30, No. 1).

[12] Lillicrap, T.P., Hunt, J.J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D. and Wierstra, D., 2015. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*.

Appendix

Here, we demonstrate that the selected values for the geometric parameters (i.e., arm length and muscle insertion site on the forearm) and the muscle-specific parameters (i.e., optimal fiber length and tendon slack length) are consistent with the physiological observation that the biceps muscle reaches its optimal fiber length—and thus its maximum isomeric force—when the elbow is about the angular range of 80–100°, whereas the triceps muscle achieves this in the range of 90–110°.

For BIC:

$$l_{\mathrm{BIC}}^{\mathrm{MT-o}} = l_{\mathrm{BIC}}^{\mathrm{M-o}} + l_{\mathrm{BIC}}^{\mathrm{T-s}} = 13 + 20 = 33 \text{ cm}$$

$$\theta_{\mathrm{BIC}}^{\mathrm{MT-o}} = \operatorname{acos}\frac{l_{\mathrm{MT-o}}^2 - l_{\mathrm{ARM}}^2 - d^2}{2dl_{\mathrm{ARM}}} = \operatorname{acos}\frac{33^2 - 32^2 - 8.1^2}{2(32)} \cong 90.1°$$

For TRC:

$$l_{\mathrm{TRC}}^{\mathrm{MT-o}} = l_{\mathrm{TRC}}^{\mathrm{M-o}} + l_{\mathrm{TRC}}^{\mathrm{T-s}} = 19 + 14 = 33 \text{ cm}$$

$$\theta_{\mathrm{TRC}}^{\mathrm{MT-o}} = \operatorname{acos}\frac{l_{\mathrm{MT-o}}^2 - l_{\mathrm{ARM}}^2 - d^2}{-2dl_{\mathrm{ARM}}} = \operatorname{acos}\frac{33^2 - 32^2 - 4.2^2}{-2(4.2)(32)} \cong 100.1°$$

where, $l_{\mathrm{MUS}}^{\mathrm{M-o}}$ and $l_{\mathrm{MUS}}^{\mathrm{T-s}}$ denote the optimal fiber length and the tendon slake length, and $l_{\mathrm{MUS}}^{\mathrm{MT-o}}$ and $\theta_{\mathrm{MUS}}^{\mathrm{MT-o}}$ represent length of muscle-tendon unit length and the elbow angle at which the fiber reaches its optimal length, respectively.