Design and Control of a Muscle-skeleton Robot Elbow based on Reinforcement Learning

Jianyin Fan, Haoran Xu, Yuwei Du, Jing Jin and Qiang Wang
Harbin Institute of Technology, Harbin, China
E-mail: fanjianyin_q@foxmail.com, haoran2400@outlook.com, hitduyw@163.com,
jinjinghit@hit.edu.cn, wangqiang@hit.edu.cn

***Abstract*—The muscle-skeleton body structure and learning ability allow natural creatures to adapt to the complex environment. These can also make robots more adaptive in human-robot interaction scenarios. In this work, we implement a humanoid muscle-skeleton robot elbow joint actuated by two antagonistic pneumatic artificial muscles (PAMs). A reinforcement learning algorithm based on soft actor-critic (SAC) is adopted to learn the control policy of the proposed elbow joint. Lower action space and hindsight experience replay (HER) further reduce training time, and the temperature factor is fixed during the training process for small steady-state error. An elbow model is implemented in the simulation to verify the training procedure for our real robot elbow platform. The experimental results show that the RL learning procedure can learn control policies in the robot elbow prototype, and the steady-state error is within 0.64% after 1 s of control time.**

I. Introduction

In the past few decades, humanoid robots have received more and more attention in the robot research field. Compared with traditional robots, the ability to use tools and infrastructure designed for humans makes humanoid robots more flexible in human-robot interaction environments.

Currently, most robots are actuated by electric motors, while some inherent characteristics of the motors restrict further development of the robot design. For example, low power-to-weight ratios of electric motors result in a large mass, making it dangerous in a collision when high stiffness control performance is desired [1]. Besides, the motors installed on the joint rotation axis lead to an unnatural appearance.

Inspired by the muscle-skeleton system of natural creatures living in complex and variable environments, we want to implement a muscle-skeleton robot with skeleton links and pneumatic muscles. PAMs share a similar appearance and character with bionic muscles, and the muscles produce tractive force through an axial contraction when the pressures are applied. The PAM's high power-to-weight ratio significantly reduces the robot's inertia. Robots driven by PAMs show more natural appearances due to the similar driving method to biological creatures. In addition, PAM provides compliance even after applying pressure due to the air compressibility, which can reduce the impact force in collisions [1].

Except for the muscle-skeleton body structures and appearance, continuously evolving motor skills also play a crucial role for natural creatures. Therefore, a humanoid robot elbow not only shares a similar struct and appearance with humans but also has similar evolving body control skills. Existing humanoid pneumatic robots, like the 7 degree-of-freedoms (DoFs) robot arm actuated by antagonistic PAMs in [2] and [3], stuck on traditional PID control or reproducing postures via fix pressure sets.

The applications of reinforcement learning (RL) algorithms in robotic fields have achieved tremendous success, like dexterous manipulation [4], Ball-in-a-Cup[5] and locomotion[6]. RL algorithms enable robots to learn policies for specific tasks by exploring environments, and this kind of learning ability is an indispensable part of humanoid robots.

In this paper, we implement a 1-DoF joint with a similar structure to the human elbow, and the RL algorithm is adopted to control the proposed robot elbow.

Our robot elbow consists of skeleton links and two antagonistic PAMs. 3D printed joint and carbon fiber tube construct the skeleton links, and the contraction of PAMs leads to the movement of the joint. The RL algorithm based on SAC [7] is adopted to learn a control policy for our robot elbow. We have demonstrated that RL can be used to control a muscle-skeleton robot arm in the simulation environment [8]. However, the algorithm still requires a large number of data to obtain the optimal policy. Reducing the environment complexity is vital for real robot control applications.

The main contributions of this paper are two folds. The first is the construction of a humanoid muscle-skeleton robot elbow actuated by PAMs, and the second is an RL learning framework for controlling the proposed joint. The rest of this paper is organized as follows: A survey of related works about robot arms driven by PAMs and off-policy RL algorithms is presented in Section II. The design and implementation of the robot elbow and RL learning procedure are illustrated in Section III. Experimental results are presented and discussed in Section V, and conclusions are summarized in Section VI.

II. Related Works

A. Pneumatic Robot Arm

While there have been many research attempts about pneumatic robots, our literature review mainly focuses on humanoid robot arms driven by PAMs.

Pneumatic humanoid robots can be grouped into two categories based on their muscle configurations: antagonistic and humanoid configurations. For the robot arms with antagonistic muscle configuration, every DoF is actuated by a pair of

antagonistic PAMs. This muscle configuration leads to an easy-to-control structure because of the decoupling DoFs.

Tondu[2] designed a 7-DoFs robot arm actuated by antagonistic PAMs, which share a human-like appearance and joint range. ISAC[9] consisted of two 6-DoFs arms with antagonistic PAMs. Encoders installed on the arms provided angle feedback while the target object was located via vision. Subsequently, a hyper controller was developed for ISAC[10]. A neural network was first adopted to bring the end-effector into the neighborhood of the target object, and then PID control with non-contact impedance was employed for precise position control. Shin[1], [11] implemented a hybrid actuation for a 2-DoFs arm. The combination of PAMs and small on-joint electrical motors significantly improved control performance. Andrikopoulos[12], [13] designed a 10 DoFs two-arm robot, and antagonistic PAMs drove all joints. ANPID was adopted for joint angles feedback control, and the whole two-arm robot weighs only 7.32kg.

For the robot arms with humanoid muscle configuration, each joint may contain multiple DoFs and was driven by redundant muscles similarly arranged to human muscles. These robot arms are more humanoid in appearance and function, but they are much more complex to control than their antagonistic muscle configuration counterparts.

To control the posture of a 7 DoFs robot driven by 17 humanoid arranged PAMs, the desired internal pressures or axial tensions of the PAMs were calculated for reproducing posture from the teaching phase [3]. Ikemotot[14] proposed a robot arm actuated by 25 PAMs, where the mechanical structures of the shoulder joint achieve larger ranges of movement. And fixed sets of desired pressures were provided to generate motions. While in [15], this shoulder joint was controlled by an input signal converted from the surface electromyogram signals, based on the muscle placement similar to human anatomy.

Kengoro in [16] was a full-size musculoskeletal humanoid robot driven by 116 PAMs, and it shared similar body proportions, joint DoFs as well as muscle arrangement with humans. A PID controller was employed for tension and length control of each muscle. Hitzmann introduced a 10-DoFs humanoid arm with metal, carbon fiber skeleton, and 28 PAMs with tension sensors [17]. Pressure and tension sensors' feedback combined with motion tracking information were used to train a neural network for forward and inverse kinematics. The robot arm mentioned above had mature mechanical structures. However, there was no sophisticated control method due to the coupling between DoFs and muscles.

B. Reinforcement Learning Algorithm

An RL agent learns desired policy by interacting with the environment, which can solve complex non-linear problems. Here we focus on sample-efficient off-policy RL algorithms due to the limitations of data collection in real robot platforms.

DDPG, an off-policy RL algorithm proposed by Lillicrap, introduced neural networks to DPG, and utilized target networks and replay buffer to ensure the stability of the training process [18]. In [19], multiple value networks were adopted to alleviate the overestimation of the value function, combined with delaying policy update to improve performance further. RL algorithms mentioned above aim for optimal deterministic policy, while the deterministic policy is brittle with respect to the environment parameters. To obtain a more robust policy, the maximum entropy RL methods maximize the cumulative discount reward and an entropy term, leading to policies that can solve tasks while acting randomly. SAC was a maximum entropy RL algorithm that could handle high dimension input and continuous, in which multiple value networks and delayed policy update from TD3 were adopted [20]. In [7], Haarnoja implemented automatically tuning of the temperature hyper-parameter for training stability.

Replay buffer was an essential part of off-policy RL algorithms. Schaul developed prioritizing replay buffer, which samples important experiences more frequently [21]. Moreover, Andrychowicz proposed hindsight experience replay (HER), which augmented the replay buffer for increasing learning speed in sparse and binary reward tasks [22]. We have demonstrated that the RL method can learn control policy for humanoid robot arms with biological muscle configuration in simulation environment[8]. In this work, we will apply the RL method to our designed muscle-skeleton elbow joint platform.

III. System Configuration

This section introduces our humanoid muscle-skeleton robot elbow joint platform and the policy learning framework. PAMs drive the elbow joint and can achieve similar movement ability as the human elbow joint. The outline of the system is shown in Fig. 1. First, the hardware platform of the muscle-skeleton robot elbow is proposed, including the design of the joint and air supply of the muscles. Then, we introduce the control framework based on SAC, where HER is employed to reduce training time.

A. Robot Elbow Implementation and Muscle Configuration

As we aim to implement an easy-to-control humanoid robot arm, several guidelines are followed. First, the designed robot elbow should have a similar movement ability to the human elbow joint. Then, sliding friction should be avoided to reduce the possibility of damage. Moreover, encoders should be installed at every DoF to provide feedback signals for the control system.

The human elbow joint can be treated as a single DoF hinge joint. Four bearings are adopted for smooth movement, and the encoder is fixed on the rotation axis. The joint's movement range is $0 \sim 100^{\circ}$. Carbon-fiber tubers are used as skeletons, and two Festo PAMs are fixed on the skeleton. Non-elastic and durable Dyneema acts as biological tendons that connect muscles and joint.

B. Air Pressure Control System

The air supply system, marked as red arrows in Fig. 1, consists of an air compressor, a reducing valve, and two proportional valves (ITV2030-212S-X154, SMC). The air

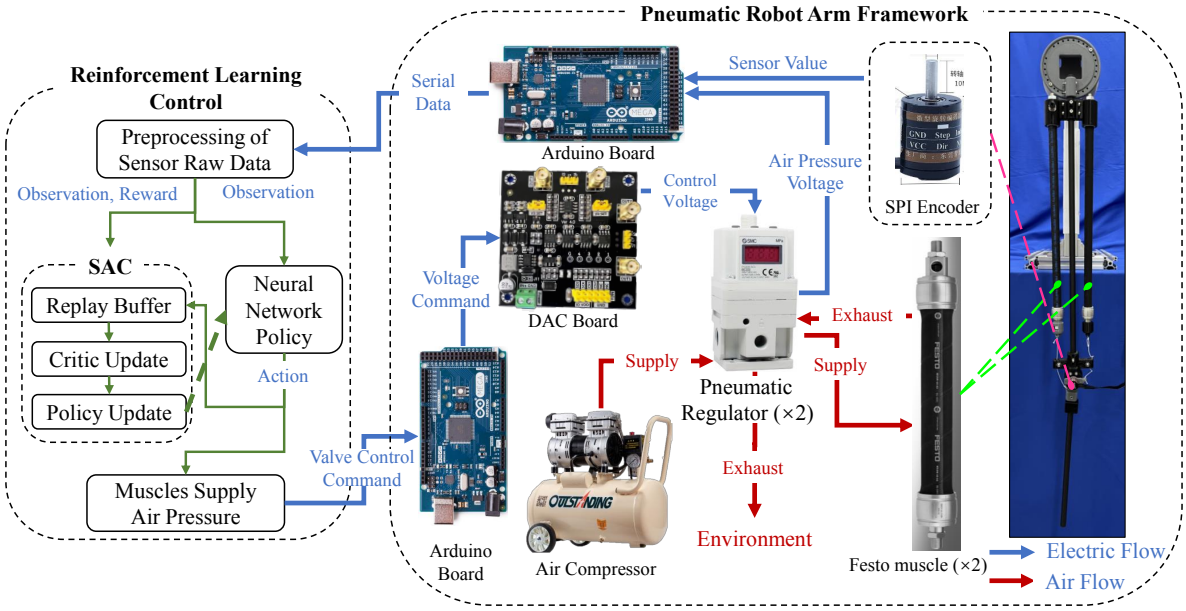


Fig. 1. System of the humanoid muscle-skeleton robot elbow.

compressor provides compressed air of 0.7 MPa, and the proportional valves regulate the air pressure according to the control commands before actuating the PAMs. The input signal of the proportional valves is a voltage analog signal between $0 \sim 5$ V, corresponding to $0.005 \sim 0.5$ MPa output air pressure. The proportional valves can also output a voltage analog signal between $1 \sim 5$ V, indicating current output air pressure.

Blue arrows in Fig. 1 represent electric signals. After the control policy inferencing muscle supply air pressure according to system observation, an Arduino Mega 2560 board receives valve control commands and sends them to a DAC board via SPI. The DAC board output control voltage to the proportional valves to change muscle supply air pressure.

Due to the antagonistic configuration of the PAMs, we use a single control variable for the pressure of both PAMs to reduce the complexity. The control signal is between $-1 \sim 1$, and the supply pressures are

$$\begin{cases} P_1 = P_{1,0} + \Delta P \\ P_2 = P_{2,0} - \Delta P \end{cases} \tag{1}$$

where P_1, P_2 are the pressures of the PAMs, and $P_{1,0} = P_{2,0} = 0.25$ MPa is the initial air pressure. ΔP is as follows:

$$\Delta P = \beta \cdot c \tag{2}$$

where c is the control signal and $\beta = 0.25$ is a scale factor.

C. Reinforcement Learning Training Framework

An RL agent learns optimal policy of Markov decision process $(\mathcal{S}, \mathcal{A}, p, r)$ via interacting with the environment. $\mathcal{S}$ is state space, $\mathcal{A}$ is action space, p is state transition probability and r is reward. A maximum entropy optimal policy is obtained through maximizing cumulative discount rewards augmented with an extra entropy term:

$$\pi^* = \arg\max_{\pi} \sum_{t=0}^{\infty} \mathbb{E}_{(\mathbf{s}_t, \mathbf{a}_t) \sim \rho_\pi} \Bigg[\sum_{l=t}^{\infty} \gamma^{l-t} \mathbb{E}_{\mathbf{s}_l \sim p, \mathbf{a}_l \sim \pi} [r(\mathbf{s}_t, \mathbf{a}_t) + \alpha \mathcal{H}(\pi(\cdot \mid \mathbf{s}_t)) \mid \mathbf{s}_t, \mathbf{a}_t] \Bigg] \tag{3}$$

where ρ_π is the state-action marginal of the trajectory distribution under the policy π, α is the temperature parameter, and γ is the discount factor. The extra entropy term makes the learned policy inferences actions as random as possible while solving the task, which encourages exploration during the training stage and shows better robustness in the face of environment drift and observation errors.

As for the control system, we want the learned policy to control our designed arm to given positions. That is, given the current state of the elbow and target elbow angle, the control policy outputs pressure values of corresponding muscles, and the muscles drive the elbow joint to reach the target joint angle. Therefore, we adopt SAC to obtain such a control policy. SAC is a maximum entropy off-policy actor-critic algorithm that utilizes an experience replay buffer, target networks, and double value network for stable and fast training. We recommend [7] for more details about SAC. The objective of updating the policy is

$$J_\pi(\phi) = \mathbb{E}_{\mathbf{s}_t \sim \mathcal{D}} \left[\mathbb{E}_{\mathbf{a}_t \sim \pi_\phi} [\alpha \log(\pi_\phi(\mathbf{a}_t \mid \mathbf{s}_t)) - Q_\theta(\mathbf{s}_t, \mathbf{s}_t)] \right] \tag{4}$$

where $\mathcal{D}$ is replay buffer storing environment transition tuples, π_ϕ is current policy with parameters ϕ, Q_θ is the action-value function with parameters θ. The action-value function Q can

Algorithm 1 SAC with HER

Randomly initialize network $Q_\theta(s,a)$, $\pi_\phi(s)$, initialize target network $Q_{\bar{\theta}}(s,a)$ with weights $\bar{\theta} \leftarrow \theta$
Initialize replay buffer $\mathcal{R}$
for Episode = 1, M **do**
  Reset environment and receive observation s_0
  Reset path buffer $\mathcal{P}$
  for Step = 1, T **do**
    Sample action $a_t \sim \pi_\phi(a_t|s_t)$
    Execute action a_t, receive reward r_t and next observation s_{t+1}
    Store transition (s_t, a_t, r_t, s_{t+1}) in $\mathcal{P}$
  end for
  Store transitions and HER augmented transitions in $\mathcal{R}$
  for Update = 1, U **do**
    Update $Q_\theta(s,a)$ according to Eq.5
    Update $\pi_\phi(s)$ according to Eq.4
    Update target network $\bar{\theta} \leftarrow \tau\theta + (1-\tau)\bar{\theta}$
  end for
end for

be trained though minimize

$$J_Q(\theta) = \mathbb{E}_{(\mathbf{s}_t,\mathbf{a}_t)\sim\mathcal{D}}\Bigg[\frac{1}{2}\Big(Q_\theta(\mathbf{s}_t,\mathbf{a}_t) - \big(r(\mathbf{s}_t,\mathbf{a}_t) + \gamma\left(Q_{\bar{\theta}}(\mathbf{s}_{t+1},\mathbf{a}_{t+1}) - \alpha\log\left(\pi_\phi(\mathbf{a}_{t+1}\mid\mathbf{s}_{t+1})\right)\right)\big)\Big)^2\Bigg] \tag{5}$$

where $Q_{\bar{\theta}}$ is the target action-value network with parameters $\bar{\theta}$.

HER is a data augmentation method for off-policy RL algorithm using replay buffer. For the environments where specific goals need to be achieved, like a robot arm reaching a given position, HER samples new goals from future states in the trajectory and combines new goals with original states to form a new transition tuple. HER increases data accumulation and further accelerate the training process. Our training algorithm is described in Alg. 1.

Each episode has a data collection phase and a network updating phase. In the data collection phase, after sampling a new target angle from the joint movement range, the policy outputs control action according to current observation. Then, the action is executed on the pneumatic elbow platform, and the learning agent receives the reward and the next observation. This process repeats until the episode finish. The collected transitions are augmented via HER and stored in the replay buffer. In the network updating phase, policy and action-value networks are updated according to their objectives. After that, the target network is 'soft' updated.

Observations include current joint angle and angular velocity, action from the last step, and target joint angle. These give rise to a 4-dimensional vector. The joint angle and angular velocity are normalized using history training data. Policy and action-value networks are MLP with two hidden layers of 512 units and ReLU activation. And the output layer of the policy was a $tanh$ layer to bound the action between $-1 \sim 1$. And the reward r is the clipped negative distance between current joint angle θ_{cur} and target joint angle θ_{tar}

$$r = \begin{cases} -|\theta_{\text{tar}} - \theta_{\text{cur}}| & \text{if } -|\theta_{\text{tar}} - \theta_{\text{cur}}| > -1 \\ -1 & \text{other} \end{cases} \tag{6}$$

In the training process, the max training episode is $M = 1000$. For each episode, we choose $T = 40$ and $U = 80$ as the number of environment steps and the number of network update per episode. HER augmented ratio is 1, which means 40 new transitions are generated per episode. The discount factor $\gamma = 0.99$, and the 'soft' update factor $\tau = 0.01$. We set the temperature parameter $\alpha = 0.002$ instead of automatically adjusting its value during the training process. There are two reasons we directly fix the value of α: First, we find a fixed value of α can slightly speed up training in our experiments as a single DoF joint leads to a plain environment. Second, a small value of α significantly reduces steady-state error. The objective in Eq. 3 tries to maximize cumulative discount rewards and an entropy term. If α does not converge to a small value, the entropy term will overwhelm the cumulative discount rewards, and this leads to a non-neglectable steady-state error.

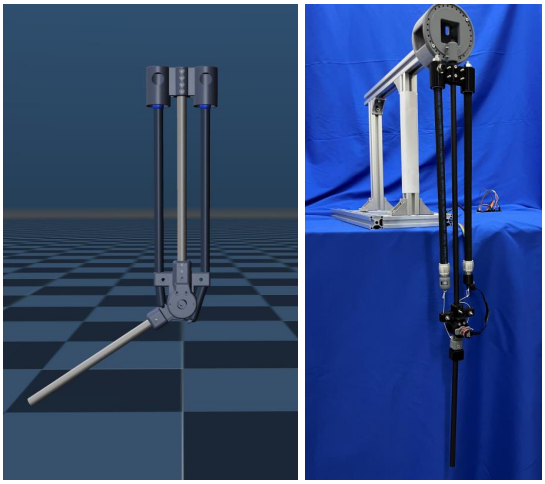


Fig. 2. Muscle-skeleton elbow joint MuJoCo simulation environment (left) and the robot elbow prototype (right).

IV. Experimental Results

We test our robot elbow joint design and training procedure in MuJoCo[23] simulation environment. Then, we build the hardware platform of the robot elbow and evaluate the RL control framework on the hardware platform. Policies are trained and tested on a PC with 16 GB memory and an NVIDIA 3080 graphics card.

A. Robot Elbow in Simulation Environment

To verify our RL learning procedure and find suitable learning parameters for muscle-actuated robot elbow, we conceptually implement our designed elbow joint in the MuJoCo

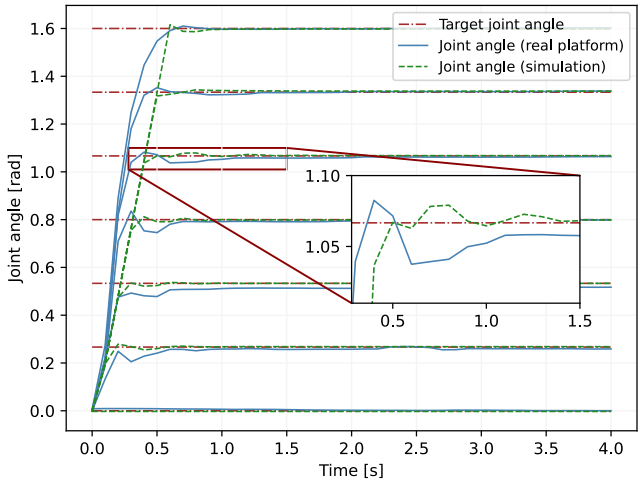


Fig. 3. Fixed angle step response of the humanoid robot elbow in the simulation environment and real platform.

TABLE I
AVERAGE STEADY-STATE ERRORS OF THE FIXED ANGLE STEP RESPONSE

Environment	**Steady-state errors** ($\times 10^{-3}$ rad)				
	Times (s)	1	2	3	4
Simulation		2.60	2.07	1.98	1.98
Real platform		10.28	6.98	5.25	5.17

simulation environment as shown in the left Fig. 2. Since the main purpose of using simulation is to establish a smooth learning procedure, we directly utilize the muscle actuator of MuJoCo rather than laboriously modeling Festo PAMs. In the simulation environment, every control step is 0.1 s consisting of 20 simulation steps.

In our experiments, we find that observations consisting of current joint angle, joint angular velocity, and target joint angle have enough information for training control policy. Moreover, the temperature factor of SAC converges to around 0.005 during the training process. Control policy with such a small temperature factor generally leads to a very small steady-state error. We evaluate the control policy learned by the RL algorithm in terms of fixed angle step response and tracking a sinusoidal signal.

Seven target angles in the joint movement range are averagely chosen for fixed angle step response. For each target angle, the policy outputs 40 actions representing a control time of 4 s, then the joint returns to the start angle of 0° before evaluating the next target angle. Step response in the simulation is illustrated in Fig. 3 as green dash lines. It is shown that the rising time of all target angles is within 1 s before stabilizing around the target angles. The average steady-state errors of fixed angle step responses are illustrated in TABLE I. We can say that the RL algorithm can learn a policy that controls muscles to drive the elbow joint to fixed target angles and achieve a neglectable steady-state error in the simulation environment.

Subsequently, the dynamic performance of tracking a sinusoidal signal is tested. The sinusoidal signal has a mean of 0.8 rad, amplitude of 0.64 rad, and period of 3 s. For every control step, the target angle is calculated according to the current time, then the action inferred by the control policy is executed in the environment. This process is repeated until the end of the dynamic performance evaluation. In the simulation environment, the current time for calculating the target angle is obtained by considering every step lasting 0.1 s. As can be seen from Fig. 4 (a) and (b), the dynamic error is within 0.05 rad after the joint successfully tracks the reference sinusoidal signal. Moreover, the average error is 0.0168 rad after the first period.

B. Robot elbow Prototype

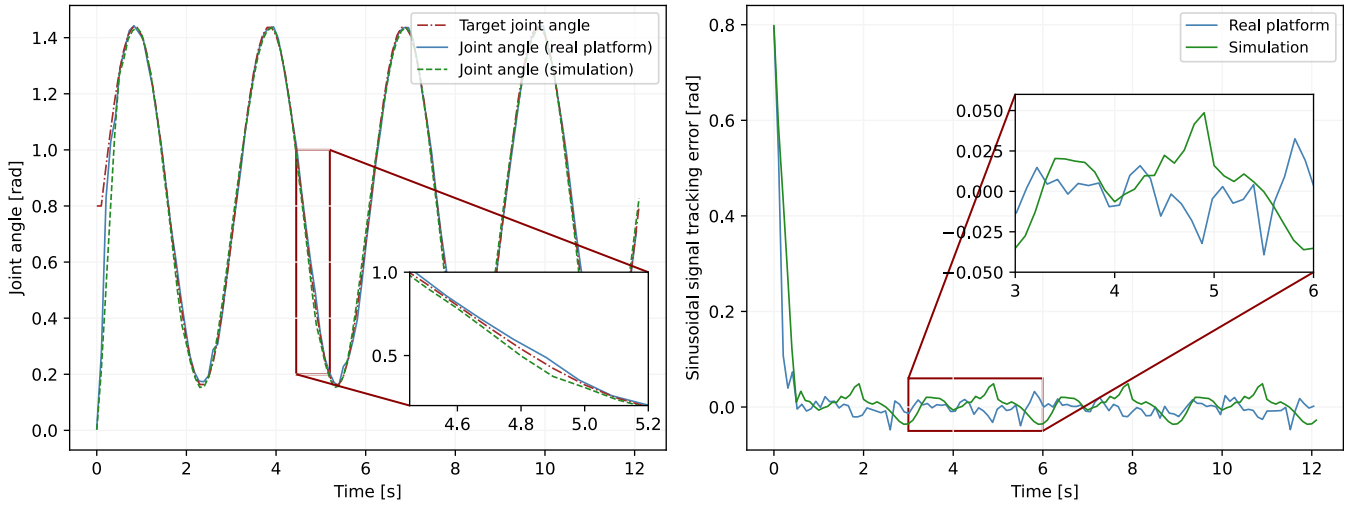
Our robot elbow prototype is shown on the right of Fig. 2. There are two major differences between the simulation and the real platform. For observations, joint states can be read directly in the simulation environment, while sensors must be installed on our robot elbow prototype to receive feedback information. An encoder is mounted on the joint rotation axis for the joint angle signal, which can be further processed to the joint angular velocity. The proportional valves can output the pressures of each PAMs, but we found that pressure information has no contribution to our experiments. Following the training configuration mentioned in Section III, observations are current joint angle, joint angular velocity, and target joint angle. It is worth noting that action from the last control step is also necessary on the real platform for stable training. The temperature factor only converges to around 0.02 in the real platform, which results in a non-negligible steady-state error. So, we directly set the temperature factor to 0.002.

Like the simulated robot elbow, we also test the learned policy in terms of step response and tracking sinusoidal signal on our robot elbow prototype. As shown in Fig. 3, for step response, all target angles have a rising time within 1 s, and the elbow can stay near the target angle without oscillation. The average steady-state error is in TABLE I. The steady-state errors decrease with running time. We can see that the control performance of the real robot elbow is deteriorated compared with that of the simulated joint, and it is reasonable because of the disturbances and uncertainty in the real environment. Furthermore, the average steady-state error is less than 0.64% after 1 s.

The results of tracking the sinusoidal signal are different from that of the step response, as shown in Fig. 4. The average tracking error is 0.0102 rad after the first period, slightly better than the simulation one. It is reasonable since the dynamic performance of the real platform is slightly better than that of the simulation one.

V. CONCLUSION

In this paper, we present a single DoF robot elbow actuated by a pair of antagonistic PAMs, and the control policy is learned via the RL algorithm. The proposed muscle-skeleton robot elbow consists of a 3D printing joint and PAMs where an encoder is installed on the rotation axis to provide necessary



(a) (b)

Fig. 4. Experiment results of the humanoid robot elbow in the simulation environment and real platform. (a) Tracking a sinusoidal signal. (b) The error of tracking the sinusoidal signal.

information for the control system. The supply pressures of two PAMs are controlled through a single variable to reduce system complexity. Moreover, a modified RL training procedure base on SAC and HER is proposed, where the temperature factor is fixed to reduce steady-state error. Experiments show that the training procedure can learn control policy both in the simulation and real-world platform. And the control precision of the fixed angle step response is within 0.64% after 1 s of control time.

Acknowledgment

This research was supported by the National Natural Science Foundation of China under Grant No. 61876054.

References

[1] D. Shin, I. Sardellitti, and O. Khatib, "A hybrid actuation approach for human-friendly robot design," in *2008 IEEE international conference on robotics and automation*, pp. 1747–1752, IEEE, 2008.

[2] B. Tondu, S. Ippolito, J. Guiochet, and A. Daidie, "A seven-degrees-of-freedom robot-arm driven by pneumatic artificial muscles for humanoid robots," *The International Journal of Robotics Research*, vol. 24, no. 4, pp. 257–274, 2005.

[3] S. Ikemoto, Y. Nishigori, and K. Hosoda, "Direct teaching method for musculoskeletal robots driven by pneumatic artificial muscles," in *2012 IEEE International Conference on Robotics and Automation*, pp. 3185–3191, IEEE, 2012.

[4] H. Zhu, A. Gupta, A. Rajeswaran, S. Levine, and V. Kumar, "Dexterous manipulation with deep reinforcement learning: Efficient, general, and low-cost," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 3651–3657, IEEE, 2019.

[5] D. Schwab, T. Springenberg, M. F. Martins, T. Lampe, M. Neunert, A. Abdolmaleki, T. Hertweck, R. Hafner, F. Nori, and M. Riedmiller, "Simultaneously learning vision and feature-based control policies for real-world ball-in-a-cup," *arXiv preprint arXiv:1902.04706*, 2019.

[6] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Science robotics*, vol. 5, no. 47, p. eabc5986, 2020.

[7] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel, *et al.*, "Soft actor-critic algorithms and applications," *arXiv preprint arXiv:1812.05905*, 2018.

[8] J. Fan, J. Jin, and Q. Wang, "Humanoid muscle-skeleton robot arm design and control based on reinforcement learning," in *2020 15th IEEE Conference on Industrial Electronics and Applications (ICIEA)*, pp. 541–546, IEEE, 2020.

[9] K. Kawamura, R. A. Peters, D. M. Wilkes, W. A. Alford, and T. E. Rogers, "Isac: foundations in human-humanoid interaction," *IEEE Intelligent Systems and their Applications*, vol. 15, no. 4, pp. 38–45, 2000.

[10] B. Ulutas, E. Erdemir, and K. Kawamura, "Application of a hybrid controller with non-contact impedance to a humanoid robot," in *2008 International Workshop on Variable Structure Systems*, pp. 378–383, IEEE, 2008.

[11] D. Shin, I. Sardellitti, Y.-L. Park, O. Khatib, and M. Cutkosky, "Design and control of a bio-inspired human-friendly robot," *The International Journal of Robotics Research*, vol. 29, no. 5, pp. 571–584, 2010.

[12] G. Andrikopoulos, G. Nikolakopoulos, D. Kominiak, and Å. Unander-Scharin, "Towards the development of a novel upper-body pneumatic humanoid: Design and implementation," in *2016 European Control Conference (ECC)*, pp. 395–400, IEEE, 2016.

[13] G. Andrikopoulos and G. Nikolakopoulos, "Design, development and control of a human-inspired two-arm robot via pneumatic artificial muscles," in *2017 25th Mediterranean Conference on Control and Automation (MED)*, pp. 241–246, IEEE, 2017.

[14] S. Ikemoto, F. Kannou, and K. Hosoda, "Humanlike shoulder complex for musculoskeletal robot arms," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4892–4897, IEEE, 2012.

[15] S. Ikemoto, Y. Kimoto, and K. Hosoda, "Shoulder complex linkage mechanism for humanlike musculoskeletal robot arms," *Bioinspiration & biomimetics*, vol. 10, no. 6, p. 066009, 2015.

[16] Y. Asano, T. Kozuki, S. Ookubo, M. Kawamura, S. Nakashima, T. Katayama, I. Yanokura, T. Hirose, K. Kawaharazuka, S. Makino, *et al.*, "Human mimetic musculoskeletal humanoid kengoro toward real world physically interactive actions," in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, pp. 876–883, IEEE, 2016.

[17] A. Hitzmann, H. Masuda, S. Ikemoto, and K. Hosoda, "Anthropomorphic musculoskeletal 10 degrees-of-freedom robot arm driven by pneumatic artificial muscles," *Advanced Robotics*, vol. 32, no. 15, pp. 865–878, 2018.

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

[19] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *International conference on machine learning*, pp. 1587–1596, PMLR, 2018.

[20] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic

actor," in *International conference on machine learning*, pp. 1861–1870, PMLR, 2018.
[21] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," *arXiv preprint arXiv:1511.05952*, 2015.
[22] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, O. Pieter Abbeel, and W. Zaremba, "Hindsight experience replay," *Advances in neural information processing systems*, vol. 30, 2017.
[23] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ international conference on intelligent robots and systems*, pp. 5026–5033, IEEE, 2012.