C-3PO: Cyclic-Three-Phase Optimization for Human-Robot Motion Retargeting based on Reinforcement Learning

Taewoo Kim† and Joo-Haeng Lee§

***Abstract*— Motion retargeting between heterogeneous polymorphs with different sizes and kinematic configurations requires a comprehensive knowledge of kinematics and inverse kinematics. Moreover, it is non-trivial to provide a kinematic independent general solution. In this study, we developed a cyclic three-phase optimization method based on deep reinforcement learning for human-robot motion retargeting. The motion retargeting and reward calculations were performed using refined data in a latent space by the cyclic and filtering paths of our method. In addition, the human-in-the-loop based three-phase approach provides a framework for the improvement of the motion retargeting policy by both quantitative and qualitative manners. Using the proposed C-3PO method, we were successfully able to learn the motion retargeting skill between the human skeleton and the real NAO, Pepper, Baxter and C-3PO robot motions.**

I. INTRODUCTION

Humans can effortlessly imitate the motions of others with different body sizes or even animals because of the humans' extraordinary motion retargeting skill that grasps the target's motion attributes from visual information and connects it with their joints appropriately. There have been several attempts to teach this motion retargeting skill to robots for motion imitation. Direct joint mapping [1]–[3] and inverse kinematics (IK)-solver-based methods [4], [5] require expertise in robot kinematics and are difficult to generalize due to their different kinematic configurations. They also have a singular position problem [6] and high IK calculation cost [5]. Recent machine-learning-based approaches learn imitation skills from demonstration, where they are collected by visual sensors [2], [3], [7], [8], motion capture (MoCap) [9]–[12] and virtual reality (VR) devices [13], [14]. However, visual-sensor-based sampling (e.g., human skeleton) is very noisy and unstable. MoCap and VR methods require additional cost and are not convenient to wear. Direct teaching (DT) methods [15]–[18] are difficult to collect a large number of demonstrations while they are intuitive.

In this study, we propose to advance the three-phase framework developed in our previous work[1] for learning human-robot motion retargeting skills. The robot agent learns a mapping policy between the human skeleton and the robot motion using reinforcement learning. Mapping is performed in the latent space, which is trained in phase 1. In phase 2, quantitative learning is performed based on a reward function and a simulator. Because the skeleton cannot provide the yaw angles of the wrist or the neck joint, we learn such motion details with a small set of direct teachings in phase 3. In our improved framework, we attempt to remove the noise in the skeleton data and utilize the latent space more actively using filtering and cyclic paths.

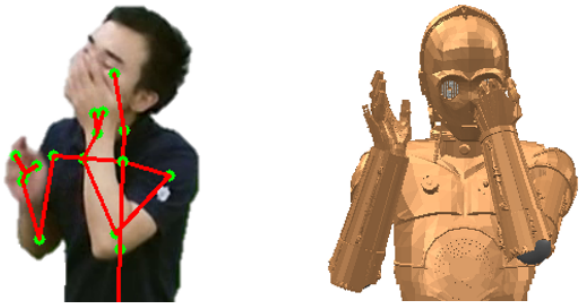

Fig. 1: Motion retargeting from humans to the C-3PO robot* using the C-3PO algorithm.

* https://www.turbosquid.com/3d-models/c-3po-star-wars-3d-obj/903731

In reinforcement learning, the widely used temporal-difference (TD) method works effectively in the Markovian environment. If a robotic task is in the Markovian environment, the state of the robot agent should include not only the angular position but also the rate of the position difference (angular velocity) to predict the next state based on the current state. However, robots using low-cost motors such as Dynamixel [19] may not provide accurate angular velocity due to sensor errors and delays in the control system [20]. Because our goal is to build a model that can be applied to such low-cost systems, we modeled our motion retargeting as a non-Markovian problem where the state of the agent has only positional information without velocity. We attempted to learn the motion retargeting policy based on the Monte-Carlo (MC) method that more effectively works in this non-Markovian environment than the TD method.

Our main contributions can be summarized as follows:

1) We propose a novel architecture by reusing the network neglected in the previous work.
2) Based on the newly proposed cyclic and filtering path, we define extended state in a latent space and a refined reward function. This method shows higher performance than developed in the previous work.
3) Based on a unified policy and an encoder-decoder network, which embraces all motion classes, we show that our model can sufficiently perform social robot motion retargeting using the MC method in the non-Markovian environment.

†Taewoo Kim is with the Department of Computer Software and Engineering, Korea University of Science and Technology, Daejeon, Republic of Korea twkim0812@gmail.com

§Joo-Haeng Lee with the Human-Robot Interaction Research Group, Electronics and Telecommunications Research Institute, Daejeon, Republic of Korea joohaeng@etri.re.kr

[1]Our previous work "*TeachMe: Three-phase learning framework for robotic motion imitation based on interactive teaching and reinforcement learning*" was accepted in Ro-Man 2019.

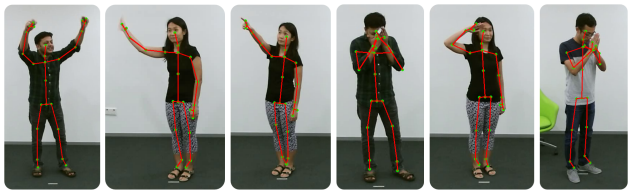

Fig. 2: Our target motion classes chosen from NTU-DB.

II. RELATED WORK

Motion retargeting has been attracting significant attention in many research fields including robotics and computer graphics [21]. In this section, we review the related studies on motion retargeting and reinforcement learning.

A. Motion Retargeting

Michael [22] proposed a method of motion retargeting on a new character with an identical kinematic structure and a different segment length using geometrically constrained optimization and a simple objective function. For online motion retargeting, *Choi et al.* [23] improved offline motion retargeting by space-time constraints and inverse rate control. *Jean-Sébastien et al.* [4] exploited an intermediate skeleton and an IK solver for retargeting from a character's motion to a geometrically and topologically different one. Another study attempted to retarget a motion between characters with different skeleton configurations such as humans and dogs [24]. *Ilya et al.* [25] proposed an automatic rigging and modeling algorithm from 3D character shapes, called *Pinocchio*. *Chris et al.* [26] proposed a real-time motion retargeting method for highly varied user-created characters using a particle IK solver. *Park et al.* [27] proposed an example-based motion cloning. In their work, using scattered data interpolation, the animator clones the behavior of the source example motion by specifying the key-posture between the source and the target with dynamic time-warping. They solved the time misalignment between the source and the target animation by fine-tuning the main algorithm process.

In the robotics field, there are many studies on motion retargeting between human motions and humanoid robots. *Behzad et al.* [28], [29] proposed an online motion retargeting method, which transfers human motions obtained from depth sensors to the humanoid robot ASIMO based on a constrained IK solver. *Sen et al.* [30] estimated a human pose from the 3D point cloud of a depth sensor and retargeted its pose to a humanoid robot without any skeleton and joint limitations. *Ko et al.* [31] presented a motion retargeting method, which solves the geometric parameter identification for motion morphing and motion optimization simultaneously. With MoCap sensors, the IK-solver-based motion retargeting methods from humans to robots have been widely studied in recent years [32], [33].

Although most motion retargeting studies have used IK-solver-based approaches, in this study, we applied reinforcement learning to motion retargeting without using any IK solvers. We also exploited the fine tuning approach for pose correction after the main learning phase as in [27].

B. Reinforcement Learning

In recent years, reinforcement learning (RL) has been used in various research areas including computer games [34]–[36], robotics [37] and animation [38] and outperformed previous approaches. Many studies in robotics used RL for a specific task such as ball throwing [39], pick & place [40], vision-based robotic grasping [41], robotic navigation [42], and other robotic tasks in daily life [43]. *Peng* [38] demonstrated learning skills such as locomotion, acrobatics, and martial arts on animation characters based on the reference motion and proximal policy optimization (PPO) [44] RL algorithm. We adopted the reference motion and the PPO algorithm with variational auto-encoder (VAE)-based [45] network architecture [39] in our learning model.

III. PRELIMINARIES

In this section, we describe the background knowledge and preliminary processes for a better understand of the method.

A. Deep Reinforcement Learning

We model motion retargeting as an infinite-horizon discounted partially observable Markov decision process (POMDP) as a tuple $\mathcal{M} = \{\mathcal{S}, \mathcal{O}, \mathcal{A}, \mathcal{T}, r, \gamma, \mathbb{S}\}$, with a state space $\mathcal{S}$, partial observation space $\mathcal{O}$, action space $\mathcal{A}$, state transition probability function $\mathcal{T}$, where $\mathcal{T}(s_{t+1}|s_t, a_t)$, reward function $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, discount factor $\gamma \in (0, 1]$, and initial state distribution $\mathbb{S}$. The goal of the agent is to learn a deterministic policy $\pi : \mathcal{O} \to \mathcal{A}$ that maximizes the expected discounted reward over an infinite-horizon:

$$J = \mathbb{E}_{\mathbb{S}}[R_0|\mathbb{S}] \tag{1}$$

where the return is defined as follows:

$$R_t = \sum_{i=t}^{\infty} \gamma^{i-t} r(s_i, a_i) \tag{2}$$

We adopted PPO-based [44] actor-critic algorithm [46] to learn the policy parameters of ω for the actor and ζ for the critic network, respectively. The critic network evaluates the action-value of the policy. We define a Q-function, which describes the expected return under policy π with parameter ζ from action a_t at state s_t as follows:

$$\begin{aligned} Q^{\pi}(s_t, a_t) &= \mathbb{E}_{\pi}[R_t|s_t, a_t] \\ &= \mathbb{E}_{\pi}[r(s_t, a_t) + \gamma Q^{\pi}(s_{t+1}, a_{t+1})|s_t, a_t] \end{aligned} \tag{3}$$

During training, the agent's experience data represented by a set of tuples $(o_t, s_t, a_t, q_t, r_t)$ are stored in a rollout memory, where $q_t = Q^{\pi}(s_t, a_t)$ and $o_t = z_t^s, s_t = (z_t^s \cup z_t^r), a_t = z_t^r$, which indicate the encoded latent representations of a skeleton z_t^s and a robot posture z_t^r at time t. The experience tuples stored in the rollout memory are then used to optimize the actor and the critic network.

B. Source Dataset

For human-robot motion retargeting, we utilized the public human motion dataset NTU-DB [47]. From initially chosen 12 motion classes among a total of 60, 6 classes

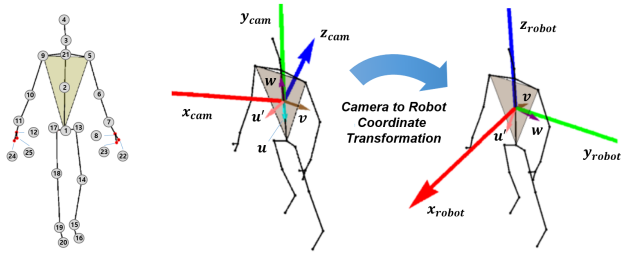

Fig. 3: **[Left]** NTU-DB skeleton and each joint number. **[Right]** Transformation from camera to robot coordinates.

TABLE I: NTU-DB Data Refinement Statistics.

Class Name	Refined Scene (Filtered / Total)	Use Rate	Total No. of Frames
Cheer up	533 / 948	56.2%	37,613
Hand waving	522 / 948	55.0%	37,228
Pointing with finger	500 / 948	52.7%	28,296
Wipe face	389 / 948	41.0%	42,172
Salute	508 / 948	53.5%	29,258
Put the palms together	493 / 948	52.0%	27,994

such as `shake head` were ruled out because they cannot easily capture the precise motions of the skeleton only. The selected final six motion classes are {`cheer up`, `hand waving`, `pointing with finger`, `wipe face`, `salute`, `and put the palms together`} (Fig. 2). We also excluded the data with severe noise and used 90% of the data for training and the remaining 10% for evaluation (Table I). The NTU-DB data manipulation code can be found in our repository: `https://github.com/gd-goblin/NTU_DB_Data_Loader`.

C. Data Pre-Processing

The skeleton data of the NTU-DB are given in camera coordinates while the robot data are given based on its torso coordinates. Because the reward in phase 2 is calculated using direction vector similarities, the coordinate alignment process between the skeleton and the robot is essential. For proper alignment, we made following assumptions.
At least within the selected motion classes:

- No bending posture at the waist exists.
- Therefore, shoulder, torso, and pelvis center joint in the skeleton are coplanar.
- The vector from the left shoulder joint to the right is always parallel to the ground.

Based on these assumptions, we performed coordinate alignment in two steps: 1) normalization with respect to (w.r.t) the skeleton torso frame, and 2) rotation w.r.t the robot basis frame. In the first step, each skeleton joint position $x_i^s = \{x, y, z\}$ is normalized by subtracting the torso position for all skeleton joints $x_i^{s'} = x_i^s - x_{torso}^s \ \forall i \in U$, where $U = \{1, 2, \cdots, 25\}$. For the second step, we first need to make an identical local coordinate to the robot torso frame. To do this, we get a vector u by $u = x_{c.pelvis}^{s'} - x_{torso}^{s'}$, each of which corresponds to joint number 1 and 2 respectively in Fig. 3, and $v = x_{RShoulder}^{s'} - x_{LShoulder}^{s'}$, which corresponds to the 5 and 9. We can then calculate the anterior axis by $u' = u \times v$ and obtain the cranial vector by $w = u' \times v$. From the normalized local coordinate frame, we create a direction cosine matrix (DCM) and transform the skeleton in camera coordinate using the DCM transpose, which is identical to the robot basis frame matrix I:

$$\text{DCM} = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{bmatrix} \begin{bmatrix} u'_x & v_x & w_x \\ u'_y & v_y & w_y \\ u'_z & v_z & w_z \end{bmatrix} \quad (4)$$

$$x_i^{s''} = \text{DCM}^T x_i^{s'} \quad \forall i \in U \quad (5)$$

where $x_i^{s'}$ and $x_i^{s''}$ are normalized joint positions and transformed positions to the robot coordinates, respectively.

IV. METHOD

We designed a three-phase framework to learn a human-robot motion retargeting skill. The policy evolves through learning by (in)direct human guidance at each phase. In addition to this, we applied filtering and cyclic paths and n-step MC method to our policy network for better performance.

A. Problem Formulation

The skeleton generation function f^s takes an image of human posture D_t at time t and generate a skeleton vector $x_t^s = f^s(D_t)$ corresponding to the input human posture, where the raw skeleton data contain x,y,z positions for all joints $x_t^s = \{x_1, y_1, z_1, \cdots, x_{25}, y_{25}, z_{25}\}$. The skeleton encoder ρ^s then takes a transformed skeleton $x_t^{s''}$ (Eq.(5)) from the raw skeleton data and generates a seven-dimensional latent representation $z_t^s = \rho^s(x_t^{s''})$. The skeleton latent vector z_t^s can be decoded by the skeleton decoder ψ^s as $\hat{x}_t^s = \psi^s(z_t^s)$ for later use in skeleton reconstruction and latent representation learning. Similarly, robot motion $x_t^{rj} = \{\theta_1, \theta_2, \cdots, \theta_{14}\}$ defined by joint angles (rad) at time t is encoded by the robot motion encoder ρ^r as $z_t^r = \rho^r(x_t^{rj})$. This latent vector of the robot motion z_t^r can also be decoded by the robot motion decoder ψ^r as $\hat{x}_t^{rj} = \psi^r(z_t^r)$ for future use in robot motion reconstruction and latent representation learning. Our mapping policy π^ω performs motion retargeting by mapping between the latent representations of the skeleton z_t^s and the robot motion z_t^r as $z_t^r = \pi^\omega(z_t^s)$.

B. Phase 1: Learning Latent Manifold

In the first phase, we learn the latent manifold of the skeleton and the robot motion using VAE [45] (Fig. 4). The skeleton encoder ρ^s consists of four fully connected (FC) layers including 512, 256, 128, 64 with ReLU, and encodes the transformed skeleton $x_t^{s''}$ in the seven-dimensional latent vector z_t^s. The skeleton decoder ψ^s has an identical structure to the encoder but in reverse order. We created a unified skeleton encoder-decoder by learning from all six motion class data at once. As described in Table I, the training data are randomly selected from 90% of the refined NTU-DB.

In order to learn the robot motion encoder-decoder, we first need to sample a set of reference robot motion trajectories corresponding to a motion in each class. We generated reference motion trajectories τ_i for all classes using the V-REP [48] and Choregraphe [49] simulator, where $\tau_i = \{x_t^{rj}, x_{t+1}^{rj}, \cdots, x_{t+T}^{rj}\}$ for $\forall i$, and $i \in H = \{\texttt{cheer up}, \cdots\}$. The reference motion generation took a

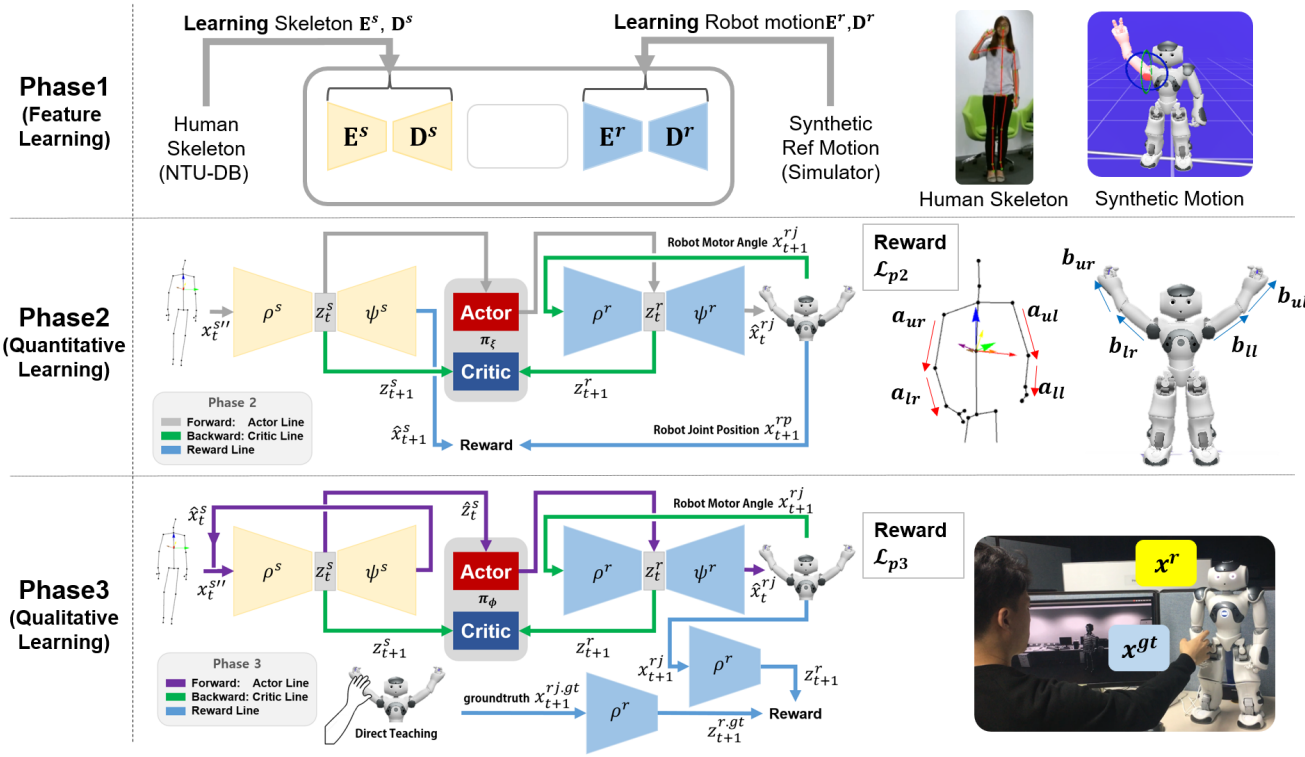

Fig. 4: Cyclic-three-phase optimization framework for human-robot motion retargeting. In phase 1, latent manifold for the skeleton and the robot motion are trained using the NTU-DB and the robot reference motion. Quantitative learning is performed using a simulator and a reward function in phase 2. The policy is optimized by DT-based fine-tuning in phase 3.

few minutes per class on average. Based on the reference motion, augmented training dataset were generated by adding uniform noise to τ_i iteratively as $x_t^{rj'} = x_t^{rj} + \epsilon$, where $\epsilon = [-0.05, 0.05]$. We augmented our reference motion trajectories up to 20k frames per class and combined them to learn a unified robot motion encoder-decoder from a total of 120k augmented datasets. The robot motion network consists of three FC layers with Tanh including 256, 128, 64 for the encoder, 7 for latent representation, and identical but in reverse order for the decoder. Both the skeleton and robot motion networks are learned using an MSE loss, learning rate=1e-4, weight decay=1e-6, and batch size=128.

C. Phase 2: Learning Mapping Function

In the second phase, we learn mapping policy π_ξ for proper motion retargeting based on a simulator and a reward function. In the forward step represented by the gray line in the second row of Fig.4, ρ^s encodes a raw skeleton to a latent vector at time t, where $z_t^s = \rho^s(x_t^{s''})$. The actor then performs mapping to generate a robot motion latent vector $z_t^r = \pi_\xi^\omega(z_t^s)$, and the decoded vector $\hat{x}_t^{rj} = \psi^r(z_t^r)$ is transferred to the robot in the simulator. After processing one time step (dt=50ms), the simulator outputs the next states x_{t+1}^{rj} with x_{t+1}^{rp}, which contains relative x,y,z positions of the robot arm w.r.t the torso frame $x_t^{rp} = \{x_{LS}, y_{LS}, z_{LS}, x_{LE}, y_{LE}, z_{LE}, x_{LW}, y_{LW}, z_{LW}, x_{RS}, y_{RS}, z_{RS}, x_{RE}, y_{RE}, z_{RE}, x_{RW}, y_{RW}, z_{RW}\}$; the subscripts represent the (left and right) shoulders, elbows and wrist joints respectively. This position vector is used in the following reward function:

$$\delta_i = \arccos\left(\frac{a_i \cdot b_i}{||a_i||||b_i||}\right), \; i \in \mathrm{U} = \{ur, lr, ul, ll\} \quad (6)$$

$$\mathcal{L}_{p2} = \frac{1}{n}\sum_{i \in S} \exp(-2.0 \cdot \delta_i) \quad (7)$$

where Eq.(6) describes the reward based on the similarities in the arm vector between the skeleton and the robot. Vectors a_i and b_i represent the direction vector of the upper and lower left and right arms for both the skeleton and the robot (see the right side of the second row in Fig. 4). These vectors can be obtained by taking the vector difference; e.g., the upper right robot arm vector is given by $b_{ur} = (x_{RE} - x_{RS}, y_{RE} - y_{RS}, z_{RE} - z_{RS})$. Even though we ruled out the skeletons with severe noise, there still remain noisy data in our dataset. Thus, in case of the skeleton, we calculated the arm vectors from the reconstructed skeleton $\hat{x}_t^s = \psi^s \circ \rho^s(x_t^{s''})$ for denoising [45] in the reward calculation. The cosine similarity-based reward δ_i is then normalized by multiplying the error amplitude constant (-2.0) and the exponential function, where $\mathcal{L}_{p2} \in [0, 1]$. In phase 2, there is a cyclic structure for learning the critic network based on the latent representation. The joint angle in the next step x_{t+1}^{rj} is encoded again and combined with the next latent vector of the skeleton as $s_{t+1} = z_{t+1}^s \cup \rho^r(x_{t+1}^{rj})$. The critic network π^ζ evaluates the action-value of the agent based on this full state, which contains the information of the skeleton and the robot motion at time t. In section V, we demonstrated the comparative analysis results, which

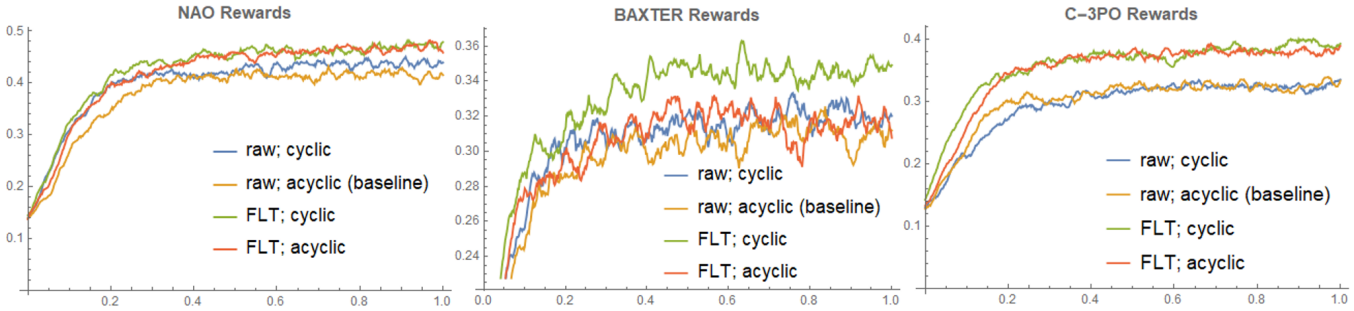

Fig. 5: Training results on the NAO, Baxter, and C-3PO robot. Policies using FLT and cyclic show the best performance.

showed that this cyclic architecture can improve the motion retargeting performance. The objective function of phase 2 can be defined as:

$$\xi^* = \arg\max_{\xi} \mathbb{E}_\pi[\mathcal{L}_{p2}(\hat{x}^s, x^{rp})|\mathbb{S}] \tag{8}$$

where $\hat{x}^s = \psi^s \circ \rho^s(x^{s''})$ and x^{rp} is obtained by applying $\hat{x}^{rj} = \psi^r \circ \pi^\omega \circ \rho^s(x^{s''})$ to the simulator. The goal of phase 2 is to find the optimal policy parameters $\xi^* = \{\omega^*, \zeta^*\}$ that maximizes the expected reward $\mathcal{L}_{p2}$. Our unified policy network consists of three 512 FC layers with the ReLU.

D. Phase 3: Policy Optimization by Fine Tuning

Even though the policy performs the mapping between the latent manifolds that are learned by the reference motion in phase 1, false retargeting can possibly occur because the reward of phase 2 does not consider the posture of the head or the wrist. In the last phase, we attempted to correct this false retargeting using DT-based fine tuning. First, we collected a ground truth dataset of 512 frames per class for about ten minutes using DT. The ground truth consists of a set of transformed skeleton frames $\boldsymbol{x^s_{gt}} = \{x_t^{s''}, x_{t+1}^{s''}, \cdots\}$, corresponding robot joint angles $\boldsymbol{x^{rj}_{gt}} = \{x_t^{rj}, x_{t+1}^{rj}, \cdots\}$ and robot joint positions $\boldsymbol{x^{rp}_{gt}} = \{x_t^{rp}, x_{t+1}^{rp}, \cdots\}$. Because the reward of phase 2 was calculated using the reconstructed skeleton $\hat{x}_t^s$, we constructed a cyclic structure that encoded the reconstructed skeleton $\hat{z}_t^s = \rho^s \circ \psi^s \circ \rho^s(x_t^{s''})$ (see phase 3 in Fig. 4). In the forward pass, the actor retargets the latent vector of a ground truth skeleton to generate a robot motion prediction. After one step simulation, the observed next robot state and the corresponding ground truth are encoded to calculate the reward function of phase 3:

$$e = ||\rho^r(x^{rj}) - \rho^r(x^{rj.gt})||_2 + \mathcal{N}(\mu, \sigma^2) \tag{9}$$

$$\mathcal{L}_{p3} = \exp(-1.0 \cdot e) \tag{10}$$

where e is calculated using the ℓ_2-norm between the robot motion prediction and the corresponding ground truth in the latent space with the human teaching error (μ=0, σ^2=1e-3) and then normalized between 0 and 1. The following equation shows the objective function of phase 3 to determine the optimal parameter $\phi^* = \{\omega^*, \zeta^*\}$.

$$\phi^* = \arg\max_{\phi} \mathbb{E}_\pi[\mathcal{L}_{p3}(\hat{z}^r, z^{r.gt})|\mathbb{S}(\phi) = \mathbb{S}(\xi^*)] \tag{11}$$

E. n-Step Monte-Carlo Learning

In general, the MC method has unbiased, high variance estimates, while the TD has biased and low variance estimates. This is because MC empirically updates the policy with the actual return, whereas the TD estimates the expected rewards by inference using bootstrapping [50]. MC usually works in episodic environments; however, it can be applied to our motion retargeting because we modeled our problem as a non-episodic task and the reward can be obtained at each time frame. Owing to the continuing and every-reward environment, we can apply the n-step MC to our problem:

$$\begin{aligned} Q^\pi(s_t, a_t) &= \mathbb{E}_\pi[G_t|s_t, a_t] \\ &= \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \cdots +, \gamma^{T-1} R_T|s_t, a_t] \end{aligned} \tag{12}$$

where T represents the number of steps in n-step MC. We present the comparative results on the n-step MC and TD (Eq.(3)) method in the next section.

V. EXPERIMENTS

Intuitive Motion Retargeting. Our C-3PO algorithm can be applied to various robots with different kinematics and sizes. This method is more intuitive than the other methods such as direct joint mapping or IK-solver-based methods because it does not require knowledge about mathematical modeling of kinematics. Through this method, we can learn the motion retargeting skill by manually appointing major joints (e.g., shoulder) and generating simple reference motion. We were successfully able to teach motion retargeting skills to the real NAO, Pepper, Baxter and C-3PO robots (Fig.6).

Learning Details. We used the learning rates of 1e-4 for the actor and 2e-4 for the critic. The rest of the hyper-parameters were set as follows: rollout steps=2048, PPO epoch=5, mini batch size=32, γ=0.98, entropy coefficient=5e-3. V-REP and Choregraphe run at 20Hz. Each unified policy learning for 1 million frames takes about 8 h on i7-8700K and Titan Xp.

A. Ablation Study on Network Architecture

To verify the performance in terms of network architecture, we evaluated them by combining the raw skeleton x_s, the filtered skeleton $\hat{x}_s$, and the cyclic $s_t = z_t^s \cup z_t^r$ and acyclic $s_t = z_t^s$ path-based reward function calculations. Fig. 5 represents the training results in these four cases. Due to the effects of noise filtering, policies using the filtering path (FLT) show far better performance than the raw skeleton method. The cyclic path is also shown to assist the policy to output better action. This ablation study shows that the proposed method is effective in improving the latent space-based motion retargeting task in various types of robots with different kinematic configurations and sizes.

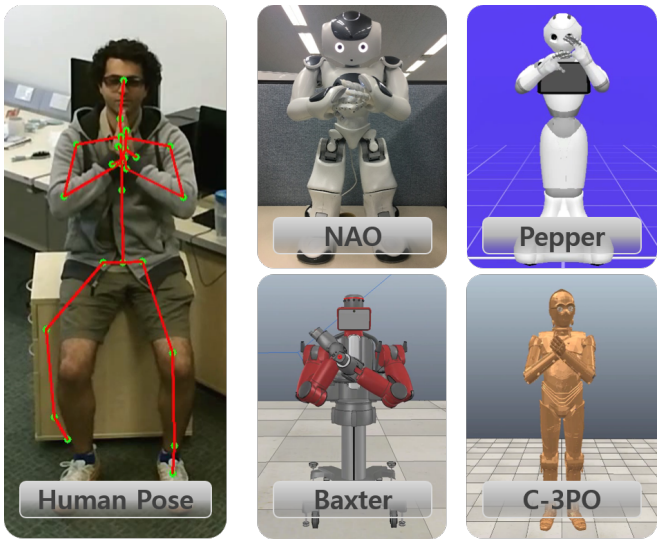

Fig. 6: Motion Retargeting result using our C-3PO algorithm. The Pepper was controlled using the NAO's policy because they have identical kinematic configurations.

TABLE II: Performance Comparison Result of TD and n-step MC Learning Methods. Evaluated by average reward and standard deviation during 5k frames.

	TD (FLT; cyclic)	MC (FLT; cyclic)		
		1-step	3-step	5-step
NAO	$\overline{r}$: 0.4463 σ : **0.1578**	$\overline{r}$: 0.5884 σ : 0.1722	$\overline{r}$: **0.5911** σ : 0.1725	$\overline{r}$: 0.5841 σ : 0.1716
Baxter	$\overline{r}$: 0.2797 σ : **0.1227**	$\overline{r}$: **0.4574** σ : 0.1633	$\overline{r}$: 0.4389 σ : 0.1523	$\overline{r}$: 0.4535 σ : 0.1647
C-3PO	$\overline{r}$: 0.4207 σ : 0.1543	$\overline{r}$: 0.4818 σ : 0.1692	$\overline{r}$: **0.4969** σ : 0.1769	$\overline{r}$: 0.4609 σ : **0.1532**

B. Temporal Difference and n-step Monte-Carlo Learning

We evaluated the performance of TD and MC w.r.t the number of steps during 5k frames. As shown in the previous subsection, because the policy with the filtering and the cyclic path showed the best performance, we only considered the FLT and the cyclic poliy in the TD method, and the n-step of MC was set to 1, 3, and 5. The experimental results evaluated by the average mean $\overline{r}$ and the standard deviation. σ in Table II suggest that MC ourperforms TD method in the non-Markovian motion retargeting problem. In MC, step-3 outperforms the others, but the overall performance is similar, and there is no dramatic performance improvement in more than three steps.

C. Policy Optimization Experiments on Phase 3

In our previous study, policy optimization by fine tuning was performed through differences in the joint space. We were able to learn the motion details; however, there was a significant loss of retargeting skill that is learned in the previous phase. We estimated that the rapid collapse of the policy is caused by the reward space mismatch; *i.e.*, the phase 3 reward is obtained in the joint space while the reward in phase 2 is based on the Cartesian space. To learn the motion details while retaining the learned skill as much as possible, we optimized the policy by fine tuning in the common latent space using a cyclic path. The ground truth

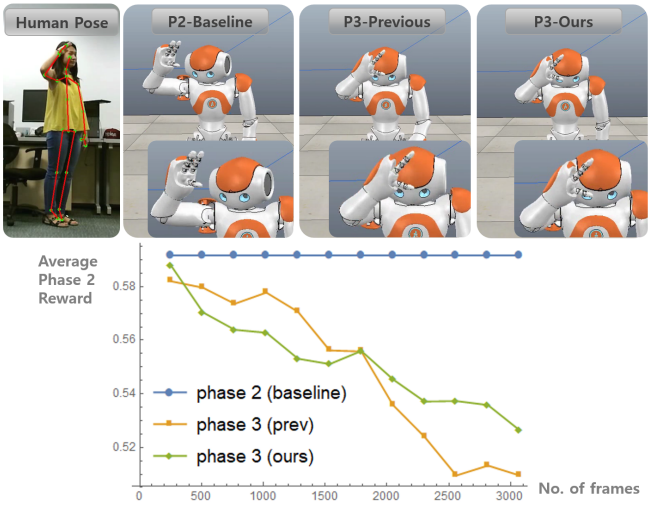

Fig. 7: Policy optimization by DT-based fine tuning. Through our method, the policy was able to learn motion details while retaining much of the retargeting skills learned from phase 2 than our previous work.

datasets of 3k frames for all six motion classes were sampled and shuffled scene-by-scene. Except for the learning rate of the actor (2e-4) and the rollout steps (256), the remainder of the other learning parameters were identical to those in phase 2. Based on this experimental environment, we were successfully able to correct our policy as shown in Fig. 7. As the fine tuning progresses, we lose the motion retargeting skill of phase 2. However, learning by cyclic path where the reward is calculated by the distance in the latent space is helpful to keep the motion retargeting skill of phase 2. Compared to our previous work, we achieved great advances in our motion retargeting task; we used a smaller training dataset with the unified encoder-decoder and policy while retaining the pre-learned skill more than our previous work. Qualitative results can be found in our supplementary video: `https://youtu.be/hXEQWXWDpTQ`.

VI. DISCUSSION AND CONCLUSION

In this study, we developed the C-3PO method for human-robot motion retargeting. In comparison with the previous work, we achieved a significant improvement in performance through the cyclic and filtering paths. In addition, we set a non-Markovian environment for our task to be applied to a low-cost system and solved it using the n-step MC method. Even though we could train the motion details using a small set of direct teaching, the motion ambiguity problem rarely occurred because we exploited frame-by-frame motion retargeting. In our future work, we will conduct a study on trajectory-based motion retargeting. We expect that this human-in-the- loop based learning framework can be extended to other robotic tasks such as human-robot interactions, specially when objects are involved.

ACKNOWLEDGMENT

This work was supported by UST Young Scientist Research Program through the University of Science and Technology. (No. [18AS1810])

This work was supported by the ICT R&D program of MSIP/IITP. [2017-0-00162, Development of Human-care Robot Technology for Aging Society.

REFERENCES

[1] P. Shahverdi and M. T. Masouleh, "A simple and fast geometric kinematic solution for imitation of human arms by a nao humanoid robot," in *2016 4th International Conference on Robotics and Mechatronics (ICROM)*, pp. 572–577, IEEE, 2016.

[2] F. Zuher and R. Romero, "Recognition of human motions for imitation and control of a humanoid robot," in *2012 Brazilian Robotics Symposium and Latin American Robotics Symposium*, pp. 190–195, IEEE, 2012.

[3] J.-H. Lee *et al.*, "Full-body imitation of human motions with kinect and heterogeneous kinematic structure of humanoid robot," in *2012 IEEE/SICE International Symposium on System Integration (SII)*, pp. 93–98, IEEE, 2012.

[4] J.-S. Monzani, P. Baerlocher, R. Boulic, and D. Thalmann, "Using an intermediate skeleton and inverse kinematics for motion retargeting," in *Computer Graphics Forum*, vol. 19, pp. 11–19, Wiley Online Library, 2000.

[5] S. Mukherjee, D. Paramkusam, and S. K. Dwivedy, "Inverse kinematics of a nao humanoid robot using kinect to track and imitate human motion," in *2015 International Conference on Robotics, Automation, Control and Embedded Systems (RACE)*, pp. 1–7, IEEE, 2015.

[6] J. J. Craig, *Introduction to robotics: mechanics and control, 3/E*. Pearson Education India, 2009.

[7] J. Lei, M. Song, Z.-N. Li, and C. Chen, "Whole-body humanoid robot imitation with pose similarity evaluation," *Signal Processing*, vol. 108, pp. 136–146, 2015.

[8] J. B. Cole, D. B. Grimes, and R. P. Rao, "Learning full-body motions from monocular vision: Dynamic imitation in a humanoid robot," in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 240–246, IEEE, 2007.

[9] J. Koenemann, F. Burget, and M. Bennewitz, "Real-time imitation of human whole-body motions by humanoids," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2806–2812, IEEE, 2014.

[10] C. Ott, D. Lee, and Y. Nakamura, "Motion capture based human motion recognition and imitation by direct marker control," in *Humanoids 2008-8th IEEE-RAS International Conference on Humanoid Robots*, pp. 399–405, IEEE, 2008.

[11] K. Yamane and J. Hodgins, "Simultaneous tracking and balancing of humanoid robots for imitating human motion capture data," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2510–2517, IEEE, 2009.

[12] S. Kim, C. Kim, B. You, and S. Oh, "Stable whole-body motion generation for humanoid robots to imitate human motions," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2518–2524, IEEE, 2009.

[13] T. Zhang, Z. McCarthy, O. Jowl, D. Lee, X. Chen, K. Goldberg, and P. Abbeel, "Deep imitation learning for complex manipulation tasks from virtual reality teleoperation," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1–8, IEEE, 2018.

[14] S. Baek, S. Lee, and G. J. Kim, "Motion retargeting and evaluation for vr-based training of free motions," *The Visual Computer*, vol. 19, no. 4, pp. 222–242, 2003.

[15] G. Grunwald, G. Schreiber, A. Albu-Schaffer, and G. Hirzinger, "Programming by touch: The different way of human-robot interaction," *IEEE Transactions on Industrial Electronics*, vol. 50, no. 4, pp. 659–666, 2003.

[16] D. Kushida, M. Nakamura, S. Goto, and N. Kyura, "Human direct teaching of industrial articulated robot arms based on force-free control," *Artificial Life and Robotics*, vol. 5, no. 1, pp. 26–32, 2001.

[17] T. Tsumugiwa, R. Yokogawa, and K. Hara, "Variable impedance control based on estimation of human arm stiffness for human-robot cooperative calligraphic task," in *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*, vol. 1, pp. 644–650, IEEE, 2002.

[18] R. D. Schraft, C. Meyer, C. Parlitz, and E. Helms, "Powermate-a safe and intuitive robot assistant for handling and assembly tasks," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pp. 4074–4079, IEEE, 2005.

[19] A. Mensink, "Characterization and modeling of a dynamixel servo," *Trabajo Individual de Investigación en el Electrical Engineering Control Engineeringde la University of Twente*, 2008.

[20] A. Bubnov, V. Emashov, and A. Chudinov, "Iterative method of measurement with a given accuracy for angular velocity errors," in *2015 International Siberian Conference on Control and Communications (SIBCON)*, pp. 1–4, IEEE, 2015.

[21] J. Bandera, J. Rodriguez, L. Molina-Tanco, and A. Bandera, "A survey of vision-based architectures for robot learning by imitation," *International Journal of Humanoid Robotics*, vol. 9, no. 01, p. 1250006, 2012.

[22] M. Gleicher, "Retargetting motion to new characters," in *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pp. 33–42, ACM, 1998.

[23] K.-J. Choi and H.-S. Ko, "Online motion retargetting," *The Journal of Visualization and Computer Animation*, vol. 11, no. 5, pp. 223–235, 2000.

[24] M.-K. Hsieh, B.-Y. Chen, and M. Ouhyoung, "Motion retargeting and transition in different articulated figures," in *Ninth International Conference on Computer Aided Design and Computer Graphics (CAD-CG'05)*, pp. 6–pp, IEEE, 2005.

[25] I. Baran and J. Popović, "Automatic rigging and animation of 3d characters," in *ACM Transactions on graphics (TOG)*, vol. 26, p. 72, ACM, 2007.

[26] C. Hecker, B. Raabe, R. W. Enslow, J. DeWeese, J. Maynard, and K. van Prooijen, "Real-time motion retargeting to highly varied user-created morphologies," in *ACM Transactions on Graphics (TOG)*, vol. 27, p. 27, ACM, 2008.

[27] M. J. Park and S. Y. Shin, "Example-based motion cloning," *Computer Animation and Virtual Worlds*, vol. 15, no. 3-4, pp. 245–257, 2004.

[28] B. Dariush, M. Gienger, A. Arumbakkam, C. Goerick, Y. Zhu, and K. Fujimura, "Online and markerless motion retargeting with kinematic constraints," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 191–198, IEEE, 2008.

[29] B. Dariush, M. Gienger, A. Arumbakkam, Y. Zhu, B. Jian, K. Fujimura, and C. Goerick, "Online transfer of human motion to humanoids," *International Journal of Humanoid Robotics*, vol. 6, no. 02, pp. 265–289, 2009.

[30] S. Wang, X. Zuo, R. Wang, F. Cheng, and R. Yang, "A generative human-robot motion retargeting approach using a single depth sensor," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5369–5376, IEEE, 2017.

[31] K. Ayusawa and E. Yoshida, "Motion retargeting for humanoid robots based on simultaneous morphing parameter identification and motion optimization," *IEEE Transactions on Robotics*, vol. 33, no. 6, pp. 1343–1357, 2017.

[32] A. E. Vijayan, S. Alexanderson, J. Beskow, and I. Leite, "Using constrained optimization for real-time synchronization of verbal and nonverbal robot behavior," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1955–1961, IEEE, 2018.

[33] L. Penco, B. Clément, V. Modugno, E. M. Hoffman, G. Nava, D. Pucci, N. G. Tsagarakis, J.-B. Mouret, and S. Ivaldi, "Robust real-time whole-body motion retargeting from human to humanoid," in *2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)*, pp. 425–432, IEEE, 2018.

[34] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[35] G. Lample and D. S. Chaplot, "Playing fps games with deep reinforcement learning," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[36] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, *et al.*, "Mastering the game of go with deep neural networks and tree search," *nature*, vol. 529, no. 7587, p. 484, 2016.

[37] Y. Liu, A. Gupta, P. Abbeel, and S. Levine, "Imitation from observation: Learning to imitate behaviors from raw video via context translation," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1118–1125, IEEE, 2018.

[38] X. B. Peng, P. Abbeel, S. Levine, and M. van de Panne, "Deepmimic: Example-guided deep reinforcement learning of physics-based character skills," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 4, p. 143, 2018.

[39] A. Ghadirzadeh, A. Maki, D. Kragic, and M. Björkman, "Deep predictive policy training using reinforcement learning," in *2017 IEEE/RSJ*

International Conference on Intelligent Robots and Systems (IROS), pp. 2351–2358, IEEE, 2017.

[40] S. James, A. J. Davison, and E. Johns, "Transferring end-to-end visuomotor control from simulation to real world for a multi-stage task," *arXiv preprint arXiv:1707.02267*, 2017.

[41] D. Quillen, E. Jang, O. Nachum, C. Finn, J. Ibarz, and S. Levine, "Deep reinforcement learning for vision-based robotic grasping: A simulated comparative evaluation of off-policy methods," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6284–6291, IEEE, 2018.

[42] A. Faust, K. Oslund, O. Ramirez, A. Francis, L. Tapia, M. Fiser, and J. Davidson, "Prm-rl: Long-range robotic navigation tasks by combining reinforcement learning and sampling-based planning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5113–5120, IEEE, 2018.

[43] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 1334–1373, 2016.

[44] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[45] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[46] V. R. Konda and J. N. Tsitsiklis, "Actor-critic algorithms," in *Advances in neural information processing systems*, pp. 1008–1014, 2000.

[47] A. Shahroudy, J. Liu, T.-T. Ng, and G. Wang, "Ntu rgb+ d: A large scale dataset for 3d human activity analysis," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1010–1019, 2016.

[48] E. Rohmer, S. P. Singh, and M. Freese, "V-rep: A versatile and scalable robot simulation framework," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1321–1326, IEEE, 2013.

[49] E. Pot, J. Monceaux, R. Gelin, and B. Maisonnier, "Choregraphe: a graphical tool for humanoid robot programming," in *RO-MAN 2009-The 18th IEEE International Symposium on Robot and Human Interactive Communication*, pp. 46–51, IEEE, 2009.

[50] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.