Human Motion Control of Quadrupedal Robots using Deep Reinforcement Learning

Sunwoo Kim*, Maks Sorokin†, Jehee Lee*, Sehoon Ha†
*Seoul National University, †Georgia Institute of Technology
Email: sunwoo@mrl.snu.ac.kr, maks@gatech.edu, jehee@mrl.snu.ac.kr, sehoonha@gatech.edu

***Abstract*—A motion-based control interface promises flexible robot operations in dangerous environments by combining user intuitions with the robot's motor capabilities. However, designing a motion interface for non-humanoid robots, such as quadrupeds or hexapods, is not straightforward because different dynamics and control strategies govern their movements. We propose a novel motion control system that allows a human user to operate various motor tasks seamlessly on a quadrupedal robot. We first retarget the captured human motion into the corresponding robot motion with proper semantics using supervised learning and post-processing techniques. Then we apply the motion imitation learning with curriculum learning to develop a control policy that can track the given retargeted reference. We further improve the performance of both motion retargeting and motion imitation by training a set of experts. As we demonstrate, a user can execute various motor tasks using our system, including standing, sitting, tilting, manipulating, walking, and turning, on simulated and real quadrupeds. We also conduct a set of studies to analyze the performance gain induced by each component.(Video[1])**

I. Introduction

A tireless and invulnerable robotic worker entering dangerous environments has long been a dream for roboticists. Many researchers have approached this goal by developing autonomous robotic systems from various perspectives, such as model-based control or learning algorithms. However, a fully autonomous agent may not work in unforseen scenarios, such as disasters, where information is lacking. This limitation motivates the need for a more flexible control system that can be applied to novel scenarios.

We propose a human motion control interface that allows users to control robots using intuitive motions. This approach has great potential to overcome completely novel scenarios by combining humans' intuition with robots' motor capabilities. Traditionally, this problem has been approached with a model-based algorithm, such as the work of Ramos and Kim [57] that projects human centroidal dynamics to the robot's space. Instead, our key idea is to exploit the recent advances in motion imitation learning [49, 53, 41] that achieve realistic motion control on simulated characters or robotic creatures. We investigate quadrupedal robots as the target platform inspired by the recent success [29, 53, 37, 36].

Our motion control system consists of two main components: the motion retargeting module and the imitation control policy. The motion retargeting module takes a live human motion as input and translates it into the corresponding robot motion with proper semantics and dynamics. Then, the imitation policy tracks the retargeted motion based on onboard sensor information. We develop the motion retargeting module in a supervised learning fashion while training the imitation policy with deep reinforcement learning.

Fig. 1. We develop a novel control system that allows a user to control a quadrupedal robot on various tasks.

However, we must address a few unique challenges to achieve the goal of developing a general motion control framework. First, we must deal with the ambiguous human motion that makes retargeting and control difficult. We mitigate this issue by adopting a hierarchical approach that learns a set of experts for both motion retargeting networks and control policies. We also develop a couple of post-processing techniques to improve the contact and temporal consistencies of the retargeted motion. Another key challenge is that our robot must imitate the target motion without accessing the

[1]Supplementary Video: https://sites.google.com/view/humanconquad

future reference trajectory, often resulting in a conservative policy. We improve the training of the motion imitation policy by adopting curriculum learning, which gradually increases the difficulty over multiple tasks.

We demonstrate that our system allows a human user to execute various motor tasks with simulated and real quadrupedal robots using a consumer-grade motion capture system, Microsoft Kinect [45]. For instance, a user can control an A1 robot to approach the target and manipulate the object with both standing and sitting postures. A user can also tilt the robot's body to reach out to a distant object or avoid incoming obstacles. We evaluate our system by conducting an ablation study of essential components, including consistency corrections, curriculum learning, and domain randomization. We list our technical contributions as follows:

- We design a novel human motion interface for a quadrupedal robot that requires minimal information about the task or the model.
- We develop an effective motion retargeting algorithm with contact and temporal consistency corrections.
- We improve the performance of motion imitation with curriculum learning and a hierarchical formulation.
- We demonstrate that a user can execute various motor tasks seamlessly on simulated and real robots.

II. Related Work

A. Legged Robot Control

Legged robot control. Striving for robust and dynamics robotic systems has enormously advanced the state of the hardware and software of the legged robots. By virtue of these advancements, legged robots can exhibit diverse, dynamic, and robust motion behaviors, allowing the robots to traverse challenging terrains or exhibit highly agile movements. The development of the hardware has enabled quadrupedal robots to perform agile motor skills while maintaining high stability [55, 27, 7]. On the other hand, the development of bipedal robots has focused on the robustness of locomotion [6, 75, 33]. Traditionally, designing effective motion controllers involves a lot of manual engineering and domain expertise. On the contrary, mathematical approaches like trajectory optimization [54, 20] and model predictive control (MPC) [26, 19, 13, 14], leverage the optimization techniques to generating robot motions while alleviating the human-powered efforts in controller design process. Such optimization methods have enabled legged robots to complete the challenging control tasks such as locomoting on a slippery floor [32, 8], traversing the rough terrain [17], recovering from the slip [16] and even keeping the balance on a large ball in a physics simulation [78]. However, the complexity of the real-legged robot dynamics usually forces these algorithms to either operate with a simplified robot model or design a task-specific controller. Our algorithm, on the contrary, allows the robot to learn a wide range of tasks without any task-specific dynamics modeling.

Learning-based control. Reinforcement learning (RL) control of physically simulated characters has led to a great performance in sophisticated motor skills such as walking, jumping, cart-wheel, and skating [49, 38, 48, 80]. However, while controllers behave well in idealized simulated environments, they often struggle when transferred to the real world, exhibiting infeasible motor-control behaviors due to the difference between simulation and real-world, which is often referred to as the *reality gap*. Some approaches propose to address the reality gap with conventional optimization methods such as MPC, allowing the policy to adjust on the real-robot [30, 68, 40, 76]. On the other hand, others have investigated methods that leverage real-world data, such as learning on real robots [22, 21, 62], identifying system parameters [29], or adapting policy behaviors [53, 83, 36]. Instead, we leverage a Domain Randomization (DR) technique [50, 82, 47, 70, 12, 58, 41], which randomizes domain parameters like mass, friction or PD gain during training in simulation to obtain more robust control policies while training only in simulation.

B. Motion Imitation

Data-driven motion controllers have been proven effective for generating a wide range of physically plausible motions by leveraging motion capture data. Although kinematic approaches can provide interactive motion control [24, 5, 63, 25, 64, 65], they cannot be directly transferred to real-world due to the lack of physical plausibility. On the other hand, physics-based motion trackers [42, 43] allow us to obtain natural motions in simulation, but its control design requires additional manual efforts, such as feature selection and motion processing. The recent RL-based formulation [49] provides an automated pipeline for developing effective motion imitation control policies from simple reward descriptions, which is capable of learning various motions on simulated characters [23, 72, 73, 49, 11, 38, 46, 48, 80, 44, 39], or even on a real quadrupedal robot [53] with manual motion retargeting. We adopt the concept of imitation objective to gain both physically correct motion and interactive control.

C. Motion-based Control

Human motion control allows for direct control of the robot body based on the human body motion. Motion control schemes can liberate the human operator from the means of the commonly used control mechanisms (e.g. joysticks, keyboards), and allows the operator to better convey his or her intents to the robot controller. In this reason, human posture-based control has been widely studied in the fields of computer animation and robotics [59, 66, 4, 77, 1, 61, 84, 71, 56, 31, 35, 57, 34, 9, 10].

Human motion control for humanoid robots. Humanoid robots that inherit many human body features are seemingly a suitable platform for mimicking human body motions. Application of such anthropomorphic creatures ranges from human interaction [71] to housekeeping teleoperation [3] or even hazardous disaster rescue [56]. With further expansions to smaller-sized humanoid control via motion imitation [35], which highlights the challenges of projecting the human posture to a new morphology. A number of methods [59, 66, 1]

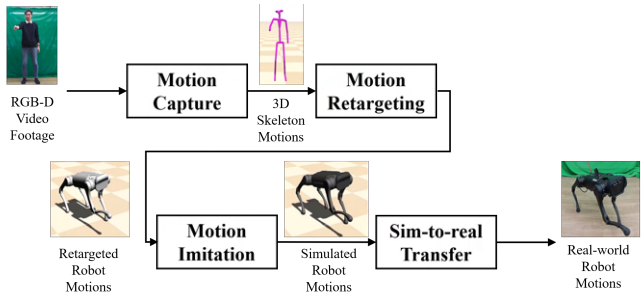

Fig. 2. Overview diagram. Our system takes a human motion as inputs and controls the robot via motion retargeting and motion imitation.

have been proposed to address the differences in the morphology configuration space, such as link length, joint limits, and degrees of freedom. Aside from morphology differences, timing issues emerge when expanding the controller from simple posture mapping to more dynamic motions, such as maintaining a balance. Zheng and Yamane [84] propose to integrate a time-warping objective to obtain a smoother motion-to-motion mapping control. Balancing has been further addressed by several approaches, such as Linear Inverted Pendulum (LIP) model safety constraints [31] or the balance feedback to the human [57]. In addition, safety has also been another main challenge during teleoperation. For instance, Choi et al. [10] proposed a shared-latent embedding retargeting algorithm to avoid self-collisions. Arduengo et al. [2] proposed a technique for adapting the dynamics of the end-effector to switch between stiffness and compliance to obtain better safety.

Motion retargeting to non-humanoid characters. Although some non-human-like characters as animals or alphabet shape characters have a different configuration from human beings, it might be possible to convey the semantics of the human posture to the character's posture [4, 77, 61] with proper motion retargeting. Researchers have proposed various pose-to-pose motion retargeting algorithms with probabilistic pose-to-pose mapping [77], using semantic deformation transfer as mapping [4], or a feature selecting method [61]. However, these methods focused mainly on posture mapping, which is hard to be extended for robot control. On the other hand, Kim et al. [34] proposed the embedding of cyclic motion on the shared latent space, which a user to control an ostrich character in a 2D physics simulation [34]. In this work, we propose a new control framework that allows a user to operate a quadrupedal robot with motions, which has a different morphology from a human. We achieve real-time motion control for a variety of tasks, including walking, tilting, manipulation, and sitting, with a minimal amount of information about each task.

III. Overview

We develop a system for controlling a quadrupedal robot with a human operator's motions. Our system receives human motions from any motion capture system, which is Microsoft Azure Kinect [45] in our case. Then the motion retargeting module (Section IV) converts the captured human into the corresponding robot reference motion that is physically valid and conveys proper semantics. To achieve this goal, we adopt a hierarchical approach of learning a set of experts mappers while applying optimization-based post-processing techniques. Then we learn a control policy that can imitate the given retargeted robot motion using deep reinforcement learning (Section V). For more robust and flexible control, we develop robust expert policies using curriculum learning and combine them as a state machine with additional transition controllers. We illustrate the system overview in Figure 2.

IV. Motion Retargeting

The first component of our motion-based control system is a motion retargeting module, which converts the user's motion into the corresponding robot motion. Many prior works have demonstrated successful human-to-humanoid motion mapping [59, 66, 1, 84, 71, 35, 56, 31, 57, 3, 10]. However, our problem is unique in the sense that we have to find a mapping function between two very different morphologies without leveraging hand-engineered motion features, such as contact states or centroidal dynamics. Even worse, we have to address additional issues, such as the sparsity of the data and the required interactivity.

We tackle this problem by learning a set of expert networks and applying post-processing. Traditional techniques [15, 4, 77] typically approach motion retargeting by solving optimization, but they tend to exhibit a slow turnaround time that is not suitable for interactive applications. And they also often require task-specific formulation [61], which makes the system hard to handle a wide variety of motions. On the other hand, learning-based approaches [10] show impressive inference capabilities at interactive rates, but they are known to be data-hungry. In addition, they can also generate inconsistent or unexpected motions that can cause severe damage to the robot. We propose a motion retargeting algorithm that first infers the motion in a supervised learning fashion from a sparse dataset and corrects the inconsistency using optimization at the post-processing stage. The set of experts will be managed by an additional selector. Our framework allows us to build a fast and robust motion mapper that can be applied to various tasks.

A. Motion Retargeting Network

In this section, we will explain how to learn a motion retargeting network for a single task. We aim to develop a motion mapper f takes a human pose $\mathbf{q}$ as inputs and maps it into the corresponding robot pose $\mathbf{p}$. However, a simple pose-to-pose mapping can be ambiguous in periodic motions because a single pose does not contain any temporal information. For instance, we can interpret the same pose in an in-place marching motion as two different phases: *swing up* or *swing down*, which must be mapped to different quadruped poses. Therefore, our model learns to map a triplet of the human pose, velocity, and acceleration $(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})$ to those of

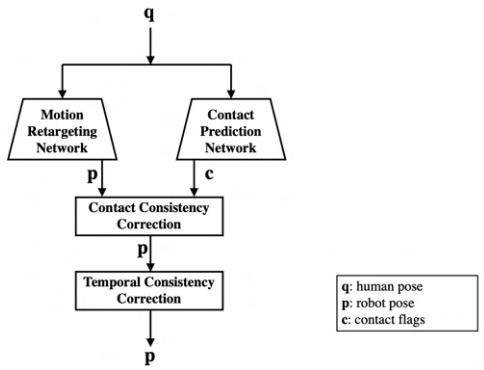

Fig. 3. The illustration of the motion retargeting module. The motion retargeting networks converts the given human motion $\mathbf{q}$ into the robot motion $\mathbf{p}$. The contact and temporal consistencies are maintained based on the inferred contact flags and previous history.

robots $(\mathbf{p}, \dot{\mathbf{p}}, \ddot{\mathbf{p}})$. We omit the derivatives in some figures and equations for brevity.

Data preparation. We prepare the dataset $\mathcal{D}$ by collecting matching pairs of human and robot motions. First, we generate robot motions for sampled tasks. For example, tilting or manipulation tasks are synthesized by generating random goals and solving robot poses using inverse kinematics. Then we interpolate these key poses with a random time interval ranging from 1 to 3 seconds. For locomotion tasks, we generate a set of walking motions with various gait parameters including body heights, foot clearance heights, and swing angles using a trajectory generator [30]. Note that these motions can be reused for imitation policy training in Section V.

Once we have the robot motions, we collect the matching human motion sequences. While showing robot motions, we ask a human user to act the "corresponding motions" based on the user's own intuition and record the motions using a motion capture system. We further manually process the motions to clean up noisy segments and fix asynchronous actions based on contact flags. Once we obtain the motions, we compute the pose derivatives for both the user and the robot using finite differences with $\Delta t = 0.1$.

Learning process. We use multi-layer perceptron (MLP) for training a mapper from the given dataset $\mathcal{D}$ (Figure 3). Our MLP consists of three leaky ReLU layers and one final hyperbolic tangent layer. Since a hyperbolic tangent function outputs the value between [-1, 1], we shift and scale the outputs using the robot joint limit vector to finalize the joint angle of the robot. Our loss function is defined as follows:

$$L_{map} = w_{ori}L_{ori} + w_{jnt}L_{jnt} + w_{dx}L_{dx} + w_{ddx}L_{ddx}. \quad (1)$$

We omit the function arguments $\mathbf{p}, \dot{\mathbf{p}}$, $\ddot{\mathbf{p}}$, $\bar{\mathbf{p}}, \bar{\dot{\mathbf{p}}}$, and $\bar{\ddot{\mathbf{p}}}$ for brevity, where the former three are the outputs from the networks and the latter three are the target values. The orientation loss $L_{ori} = d(\mathbf{p}^{root}, \bar{\mathbf{p}}^{root})$ compares the root orientation $\mathbf{p}^{root}$ in quaternion and its target value $\bar{\mathbf{p}}^{root}$ via a quaternion distance function d. The joint angle loss $L_{jnt} = \|\mathbf{p}^{jnt} - \bar{\mathbf{p}}^{jnt}\|^2$ is designed to match the joint angles $\mathbf{p}^{jnt}$ and their target values $\bar{\mathbf{p}}^{jnt}$. Two end-effector terms, $L_{dx} = \|\dot{\mathbf{x}} - \bar{\dot{\mathbf{x}}}\|^2$ and $L_{ddx} = \|\ddot{\mathbf{x}} - \bar{\ddot{\mathbf{x}}}\|^2$ compares the end effector velocities $\dot{\mathbf{x}}$ and accelerations $\ddot{\mathbf{x}}$ against their target values, $\bar{\dot{\mathbf{x}}}$ and $\bar{\ddot{\mathbf{x}}}$, respectively. Please note that these values can be derived from $\dot{\mathbf{p}}$ and $\ddot{\mathbf{p}}$. We set the weights w_{ori}, w_{jnt}, w_{dx}, and w_{ddx} as 0.3, 1, 0.001, and 0.001 for all the experiments, respectively.

B. Post-processing for Consistency

The learned function often generates physically invalid motions in practice. This inconsistency slows the learning of a control policy and degrades the final motion's quality. To this end, we further clean up the motion at the post-processing stage to maintain contact and temporal consistency.

Contact consistency correction. Contact consistency without foot skating is crucial to obtain physical plausibility. The violation of contact consistency destabilizes the robot balancing, hence leading to learning failure. To this end, we estimate four dimensional contact flags $\mathbf{c}_t$ and fix undesirable movements if they supposed to be in contact phases.

One possible approach to estimate $\mathbf{c}_t$ is to simply compare the current robot's foot heights against a certain threshold. However, we found that this approach yields undesirable discontinuous motions. Instead, we learn an auxiliary network that can predict smooth contact probabilities directly from human motions. We train this contact consistent network from the same training data using the following loss function:

$$L_{cp} = \|\bar{\mathbf{c}}_t(\bar{\mathbf{q}}, \bar{\dot{\mathbf{q}}}) - \mathbf{c}_t(\mathbf{q}, \dot{\mathbf{q}})\|^2, \quad (2)$$

where the contact probability $\bar{\mathbf{c}}_t$ is continuously estimated from the foot height and velocity: 1.0 if the height is 0cm and the velocity is 0.0cm/s, while 0.0 if the height is above 2cm and the velocity is more than 60.0cm/s. We apply inverse kinematics when the contact probability is greater than 0.5 to correct the contact feet to the previous frame's positions.

Temporal consistency correction. We also invent an additional procedure to ensure the temporal consistency of the retargeted motions over multiple frames because abrupt movements on the real robot often cause dangerous situations. In our experiments, this was critical to deploy the proposed system to the real world by making the entire system more stable. To this end, we clip the joint angles with respect to the velocity limits, which is set to $120^\circ/s$.

C. A Set of Experts for Multi-task Support

In our experiments, it becomes more difficult to obtain an accurate motion mapping when the human motions for multiple tasks are close to each other. To address this issue, we propose to use a hierarchical learning approach that manages a set of expert networks.[79, 74, 52, 18] We first learn three different motion retargeting networks for three robot states, *stand*, *walk*, and *sit* (Figure 4). Each network work can handle multiple tasks, such as *manipulation-at-stand* or *tilting-at-stand*. We query k-Nearest neighbors (kNN) over the input data to identify the expert associated with the closest data set; if one expert's training data is closer to the current one, we

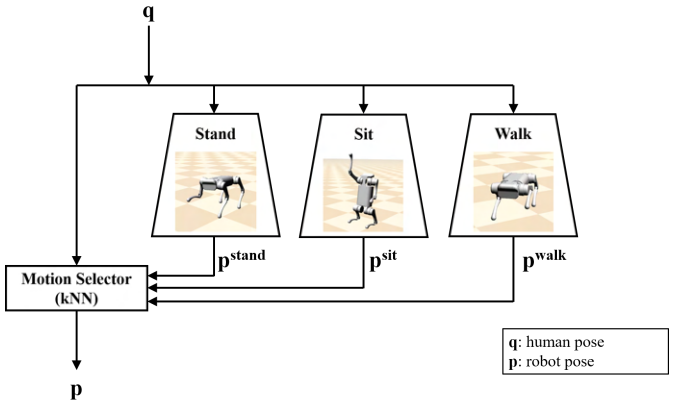

Fig. 4. We learn a set of expert motion retargeting networks for better accuracy in multi-task scenarios.

switch to the corresponding expert. We found that this results in an accurate mapping function while greatly reducing the time for manual engineering, such as hyperparameter tuning and data curation.

V. Motion Imitation

Once we generate the kinematic robot motions, the next step is to develop a control policy to imitate the given reference. We employ the motion imitation learning framework of Peng et al. [49] that allows natural and diverse motions in simulation, which has also been applied to a quadrupedal robot [53].

Our problem is more ambitious than the others because we have to track a wide range of motions on real robots. In addition, our problem is facing additional unique challenges that make learning more difficult. First, we must imitate noisier references because they are from live human movements. Because a human cannot reproduce the exact same motion, our controller must be able to imitate similar motions with spatial and temporal noises. Second, our controller does not have access to the "future" reference motions. Indeed, this absence of future information often puts a robot into conservative states rather than actively tracking the references.

We aim to maximize the performance of motion imitation by introducing the following techniques. First, we design a hierarchical controller that learns three expert controllers for robot states, *stand*, *sit*, and *walk*, while manually designing transition controllers between states. Second, we obtain effective expert controllers with curriculum learning, which is arranged over various difficulties and tasks. These inventions allow us to develop a practical controller that handles a wide range of motions on both simulated and real robots.

A. Background: Reinforcement Learning

We formulate our problem as Partially Observable Markov Decision Processes (PoMDP) to utilize the reinforcement learning [67]. An each time step, an agent observes an observation $\mathbf{o}_t \sim \mathcal{O}(\mathbf{s}_t)$ emitted from the current state $\mathbf{s}_t$ and takes an action $\mathbf{a}_t \sim \pi(\mathbf{a}_t|\mathbf{o}_t)$ from its policy π. This results in the trajectory of the states and actions $\tau = \{(\mathbf{s}_0, \mathbf{a}_0), (\mathbf{s}_1, \mathbf{a}_1), \cdots (\mathbf{s}_T, \mathbf{a}_T)\}$ where T is the episode length. Our goal is to find the optimal policy that maximizes the expected return:

$$J(\pi) = E_{\tau \sim p(\tau|\pi)}[\sum_{t=0}^{T-1} \gamma^t r(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1})], \tag{3}$$

where $p(\tau|\pi)$ is a probability of the given trajectory τ.

B. Formulation of Motion Imitation

We formulate the problem of imitating the given reference motion as PoMDP.

Reference Motions. We take the generated robot trajectories that are used for training a mapping function in the previous section and use them as example reference motions for motion imitation learning. We injected a noise vector into reference motions to improve the robustness of the learned policy.

Observation. The observation $\mathbf{o}_t = [\mathbf{z}_{t-3:t}, \mathbf{a}_{t-3:t-1}, \bar{\mathbf{p}}_{t-3:t}]$ consists of three components: robot sensor data, previous actions, and reference poses, with their corresponding histories. Each robot sensor data $\mathbf{z}_t$ is a 16 dimensional vector from 12 joint motor encoders and 4 IMU orientation and angular velocity readings in pitch and roll axes. A history of previous actions $\mathbf{a}_{t-3:t-1}$ are also stored to make the problem more Markovian in the real world. The previous reference poses $\bar{\mathbf{p}}_{t-3:t}$ are also given to the robot. Please note that we do not have *future* reference motions due to the nature of our problem, which makes the tracking task more difficult.

Action. The action $\mathbf{a}_t$ defines as the PD target for the twelve joint motors of a robot. We apply the Butterworth low-pass filter with the cut-off frequency at 5Hz to actions to generate smoother motions.

Reward function. The reward function encourages the agent to imitate the given reference motion while adapting to the physics simulation:

$$r_t = w^{main} r_t^p \cdot r_t^e \cdot r_t^{rp} \cdot r_t^{ro} \cdot r_t^{sp} + w^{acc} r_t^{acc}, \tag{4}$$

which adopts the multiplicative form inspired by previous works [38, 48]. The term r_t^p refers to a joint imitation reward:

$$r_t^p = \exp\left(s_p \sum_j \|\bar{\mathbf{p}}_t^j - \mathbf{p}_t^j\|^2\right), \tag{5}$$

where $\bar{\mathbf{p}}$ and $\mathbf{p}$ are the target and current joint angles. The end-effector reward r_t^e drives the robot to track the end-effector of the reference:

$$r_t^e = \exp\left(s_e \sum_e \|\bar{\mathbf{x}}_t^e - \mathbf{x}_t^e\|^2\right), \tag{6}$$

where $\bar{\mathbf{x}}_t^e$ and $\mathbf{x}_t^e$ are the target and current end effector positions. Similarly, the root position reward r_t^{rp} and the root orientation reward r_t^{ro} measures the differences in root position and orientation:

$$\begin{aligned} r_t^{rp} &= \exp(s_{rp} \|\bar{\mathbf{x}}_t^{\text{root}} - \mathbf{x}_t^{\text{root}}\|^2) \\ r_t^{ro} &= \exp(s_{ro} d(\bar{\mathbf{p}}_t^{\text{root}}, \mathbf{p}_t^{\text{root}})^2) \end{aligned} \tag{7}$$

by comparing the current root position $\mathbf{x}^{\text{root}}$ and orientation and $\mathbf{p}^{\text{root}}$ with respect to their target values, $\bar{\mathbf{x}}_t^{\text{root}}$ and $\bar{\mathbf{p}}_t^{\text{root}}$. Finally, we penalize the deviation from support polygon

$$r_t^{sp} = \exp\left(s_{sp} d_{sp}(\mathbf{x}^{\text{root}}, \mathbf{p}^{\text{root}}, \mathbf{p})^2\right), \tag{8}$$

where d_{sp} is the minimal distance to the support polygon. We only measure d_{sp} when the robot is required to make at least three contacts: otherwise, d_{sp} is defined as zero. Finally, we penalize excessive motions with the acceleration penalty term:

$$r_t^{acc} = \exp\left(s_{acc} \sum_j \|\ddot{\mathbf{p}}_t\|^2\right). \tag{9}$$

For all experiments, we set weight terms $w^{main} = 0.9, w^{acc} = 0.1$ to emphasize the main mimicking term. The scaling coefficients are set to $s_p = 1.0$, $s_e = 20.0$, $s_{rp} = 20.0$, $s_{ro} = 5.0$ and $s_{sp} = 10.0$ respectively.

Early termination. The early termination accelerates the learning speed which is proven by many works [49, 51, 73, 81]. We trigger the early termination when the trunk of the robot touches the ground and self penetrating contact happens.

Learning process. We optimize policies with Proximal Policy Optimization [60]. The policies are represented as feedforward networks that consists of two hidden layers with 256 ReLu neurons. The PPO has a clipping range of 0.2, learning rate of 0.00005, the discount factor is $\gamma = 0.95$, and the GAE parameter is $\lambda = 0.95$. The minibatch size is 128 for policy and value network. The max gradient norm is set to 0.5.

C. Curriculum Learning over Tasks and Difficulties

While the above formulation works well for a single motion clip, our goal of learning a versatile policy for multiple tasks remains a challenging problem. In our experience, naive learning will result in a conservative policy that is stuck in a steady position to avoid falls while not trying to follow the target motion. To address this issue, we train an expert policy for the given state with a curriculum that expands the range of the motion and also expands the number of tasks.

For this purpose, we sort all the robot reference motions based on two criteria: a task type as a primary and difficulty of the task as a secondary. For instance, we train a control policy by training on the *tilting-at-stand* task first and expanding the task set by adding the *manipulation-at-stand* task. For both tasks, we gradually increase the difficulty of the task by measuring the range of reference motions. Similarly, we train an *walking* expert by expanding the curriculum from the *walking forward* task to the *turning left/right* task, with increasing turning rates.

D. Hierarchical Control with States

Our motion includes multiple tasks, *tilting*, *manipulation*, and *locomotion*, over three different robot states, *stand*, *sit*, and *walk*, which yields different combinations such as *tilting-at-stand* or *manipulation-at-sit*. Instead of learning one monolithic policy, we learn three experts for three robot states and develop special transition controller that are called when the motion selector of the motion retargeting detects transitions. These transition controllers can be developed in multiple ways, such as model-based control or reinforcement learning, but we choose to reuse the existing motion imitation framework. The transition takes 1 to 3 seconds depending on the tasks and robot's state. During transitions, the robot executes the predefined policy while ignoring human motions.

Parameters	Range	Unit
Link Mass	[0.75, 1.25] X default	kg
Ground Friction Coefficients	[0.5, 1.5]	1
Proportional Gain	[0.7, 1.3] X default	N/rad
Derivative Gain	[0.7, 1.3] X default	N·s/rad
Communication Delay	[0, 0.016]	sec
Ground Slope	[0, 0.14]	rad

TABLE I
DOMAIN RANDOMIZATION PARAMETERS

E. Domain Randomization

The gap between the dynamics of the simulation and the real world decreases the performance of policies trained in simulation that are deployed on a real physics system. We introduce Domain Randomization [50], that randomizes dynamics parameters during the training to obtain more robust control policies. The randomized dynamic parameters and their ranges are specified in Table I. We gradually increase the range of dynamic parameters with a curriculum similar to the method mentioned in subsection C. Detailed procedures are well mentioned in previous works [49, 41].

VI. RESULTS

We design experiments to evaluate our framework from three perspectives. First, we evaluate the proposed system on a set of tasks in simulated and real environments. Second, we conduct an ablation study to evaluate the effectiveness of important components. Finally, we qualitatively compare our system with the previous motion-based interfaces.

A. Experimental Setup

We test our system on an A1 quadrupedal robot [69], which has three degrees of freedom for each leg and six under-actuated degrees of freedom for the root. We prepared 76 to 522 matching data pairs for training the mapper varying by the tasks. We train each expert policy using 1.2 billion samples in the RaiSim [28] physics simulator. We conducted all the experiments with a desktop computer with Intel 16 core 3.60GHz i9-9900K CPU and GeForce RTX 2070 SUPER GPU. We capture a human motion using a Kinect [45].

While all simulation demos are controlled interactively, we conduct real-world experiments in two modes: (1) the live mode that controls a real robot in an end-to-end fashion and (2) the replay mode that controls the robot to the prerecorded human motion trajectories. This is because fluctuating control delays could harm real robots. However, we do not feed the future trajectory information even in the replay mode, which is not available in the live mode. We annotate the experiment modes in the manuscript and supplemental videos.

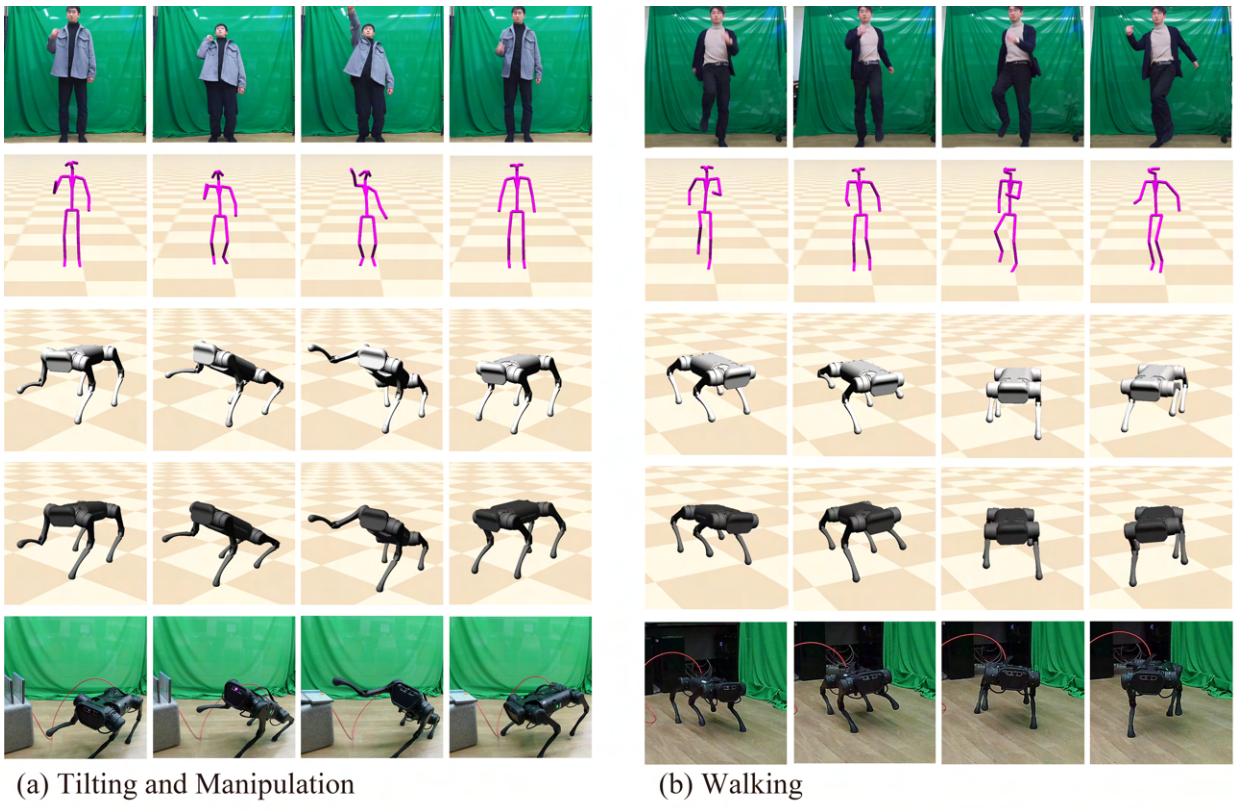

Fig. 5. Motions of the (**a**) *tilting and manipulation* and (**b**) *walking* tasks. From the top row, we illustrate human video footage, human skeleton, retargeted robot motion, simulated motion, and real robot motion, at the corresponding time frames.

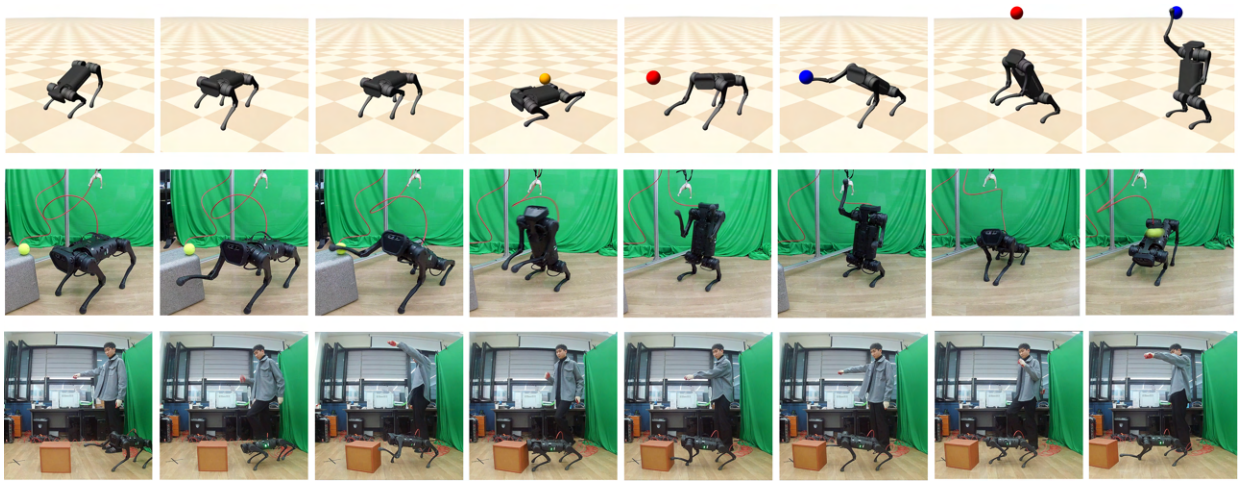

Fig. 6. Composite task demos in simulation (**top**), real-world (**middle** (replay mode) and **bottom** (live mode)).

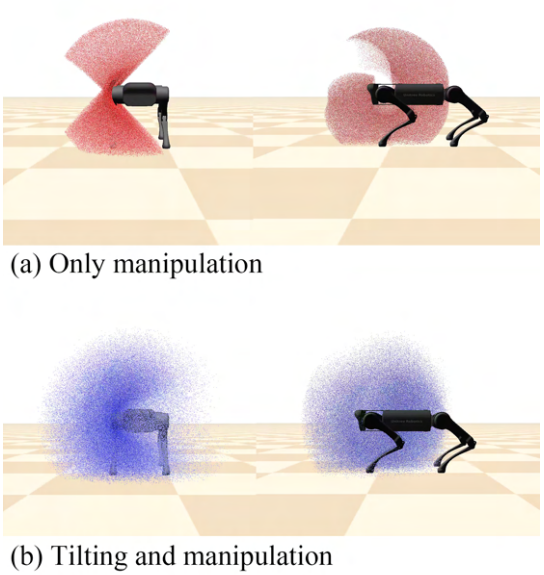

Fig. 7. Point clouds that illustrate the workspace of the right front leg when performing only manipulation(red) and manipulation with tilting(bottom) during *standing*. The robot can reach approximately 2.7 larger areas by simultaneously tilting its body.

B. Motion Performance

Individual tasks. Our system enables a user to control a diverse set of motor skills for A1 using human motions. In the *stand* state, a robot can move its end-effector while simultaneously tilting its body. A robot can tilt its body $-40°$ to $40°$ for all x, y, z axes, which is larger than the tilting range of the manufacturer's controller: $-20°$ to $20°$ for pitch and roll and $-28°$ to $28°$ for yaw. Similarly, a robot can transit to the *sit* state that allows the robot to use both arms and reach higher targets. During sitting, the robot can tilt $30°$, $15°$, and $7°$ in pitch, roll, and yaw axes, respectively. Finally, a robot can walk at the speed of 0.0m/s to 0.97 m/s with the maximum turning rate of $15°$ per second. The motion is less stable than a regular walking controller to prepare abrupt change of the speed at any moment. We illustrate the motions in Figure 5 and the supplemental video.

We also found that simultaneous manipulation and tilting provide a broader workspace, which is approximately 2.7 times larger than manipulation without tilting. We compare the workspaces as point clouds in Figure 7.

Composite tasks. Our system allows a user to switch between tasks seamlessly. In the simulation, we conduct an experiment with a sequence of the following tasks: (1) tilting the body to express a greeting, (2) walking forward to reach a target in 3m, (3) dodging a thrown orange ball by crouching, (4) manipulating the target and (5) touching another target high in the air (Figure 6 top). Similarly, we control a real robot to execute the following tasks: (1) hitting a tennis ball located at 0.42m height, (2) touching a bone hanging high at 0.8m height, and (3) dodging a thrown tennis ball (Figure 6 middle). The robot must sit to achieve the second task because it cannot reach the bone while *standing*. This long sequence is originally executed in simulation in the live mode and replayed on the hardware. These scenarios demonstrate the seamless transition capability of our control system. Finally, we control a real robot to push the box to the target position(X-mark on the floor) in real-time. This task is challenging because the box is located far from the initial position. To complete the task, we control the robot by repeating the following control tasks: (1) walk to the box and (2) push the box toward the target (Figure 6 bottom).

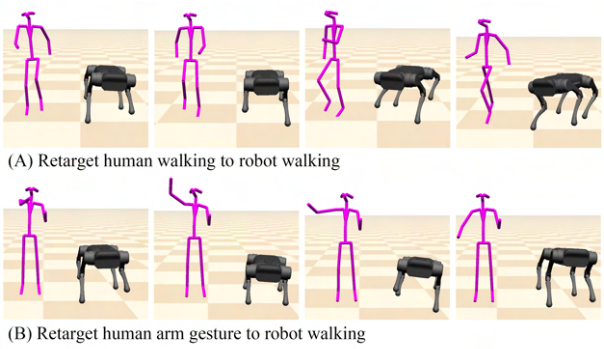

Fig. 8. Different styles of mapping for *walking*. The **top** shows a mapping with *in-place marching* and the **bottom** shows a mapping with *hand gestures*.

Control responsiveness. In real-time control, responsiveness is one of the most important criteria, which is even more critical for a quadrupedal robot with a floating base. Our control loop mainly consists of two stages: *motion reconstruction* and *control inference*. Control inference includes the main technical components, such as motion retargeting, post-processing, and querying the control policy, while motion reconstruction is inferring human 3D poses from the Kinect. In our experience, the entire control inference takes less than 0.01 second, which is fast enough for our 30 Hz control loop. We stabilize the control frequency by skipping Kinect reading when the delay is significant and reusing the existing human motions from the previous frame.

Different mapping styles. Our system is flexible enough to support different styles of mapping for the same task. To demonstrate this, we generate two motion retargeting functions with (1) in-place marching motions and (2) circular hand gestures. Both mapping styles generate successful marching motions in the simulation (Figure 8).

Semantic mapping with manual features. We can also manually tune a mapping by selecting features for retargeting. For instance, we can build a mapping for the *walking* task with explicit notions of the target velocities by extracting them from both human and robot motions. Although this explicit mapping offers slightly better motion quality, this requires domain-specific knowledge of the task.

C. Analysis

Contact and temporal consistency. We evaluate the importance of *contact consistency* and *temporal consistency* corrections by conducting an ablation study. We generate

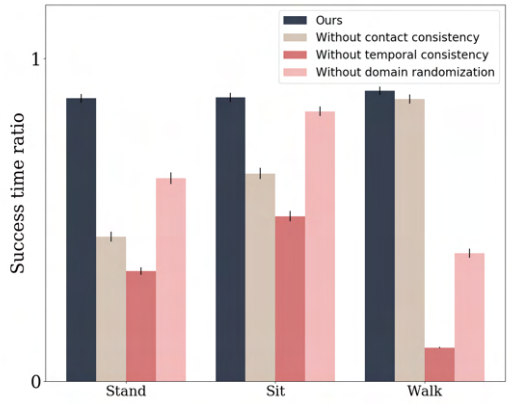

Fig. 9. Average success time ratio, which is the ratio of the termination time to the maximum episode duration. We conduct an ablation study with contact consistency, temporal consistency, and domain randomization to evaluate their effectiveness.

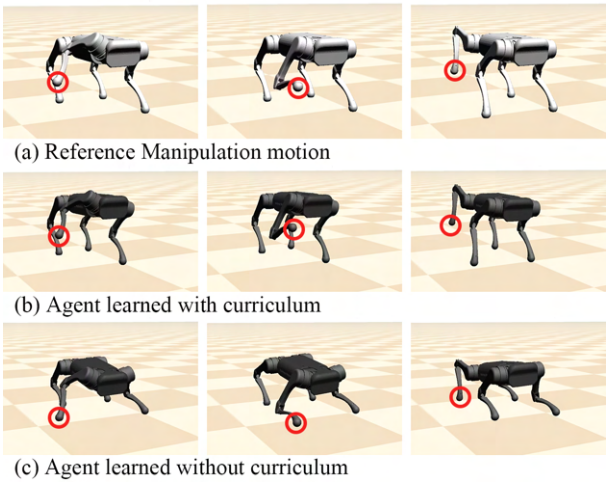

Fig. 10. Snapshots of the robot control to show the effectiveness of curriculum learning. The physically simulated agent tries to mimic the motion of reference (**Top**). While the policy trained with curriculum successfully mimic the reference (**Middle**), the policy without curriculum is stuck in local optimum (**Bottom**).

5120 test episodes of tracking 10 seconds of noisy trajectories perturbed from the ground truth robot motions. We compare methods based on the success time ratio, which is the ratio of the termination time to the maximum episode length. As illustrated in Figure 9, both components are essential for achieving the best performance. While contact consistency correction is more important for *standing* and *sitting* motions, temporal consistency seems crucial for *walking* motions.

Curriculum learning. In our experience, curriculum learning is essential for obtaining the best motion imitation performance. We do not evaluate the performance based on the success time ratio because policies without curriculum learning tend to survive until the last frame while showing conservative behaviors. Instead, we compare the quality of motions in Figure 10 and the supplemental video. They illustrate the conservative behaviors of the policies without curriculum, which put all the feet on the ground and do not attempt to reach the target.

Criteria	(A)	(B)	(C)	(D)	Ours
Mapping dimension	2D	3D	3D	3D	3D
Real-Time	Y	N	N	Y	Y
Dynamics	Y	N	Y	N	Y
Mapping Flexibility	N	N	N	Y	Y
Sim2Real	N	N	N	N	Y

TABLE II
COMPARISON WITH PREVIOUS HUMAN TO NON-HUMANOID CONTROL METHODS (A)KIM ET AL. [34], (B)DONTCHEVA ET AL. [15], (C)YAMANE ET AL. [77] (D)SEOL ET AL. [61]

Domain adaptation. Domain randomization (DR) has been one of the most effective techniques for overcoming the sim-to-real gap. We evaluate its effectiveness by measuring the success time ratio over 5120 test cases with randomized dynamics. Figure 9 shows that DR is essential for all the tasks. In addition, we conduct the sim-to-real experiment, where the policy without DR cannot complete the given motion: please refer to the supplemental video.

Importance of future reference. We often observe that the quality of the motion imitation is not as good as we expected. We hypothesize that the poor tracking performance is because our real-time motion tracking does not have information about the future reference trajectory. We verify this hypothesis by training an additional policy to track the fixed trajectory with future information and comparing the motion quality with the original agent. In our experience, the policy with future information generates more stable motions: please refer to the supplemental video for visual comparisons.

D. Comparison to Other Methods

We compare our method with the previous human to non-humanoid character control approaches based on criteria that are meaningful for control in Table II. Kim et al. [34] showed the mapping that corresponds to the dynamical systems of two different morphologies, but it is limited to 2D cyclic motions. Dontcheva et al. [15] proposed the concept of detecting the human gesture to search the matched motion pair of characters. This method only provides kinematic animation without the notion of dynamics. Yamane et al. [77] successfully mapped human motions to non-humanoid characters with natural movements. However, this approach didn't aim to get real-time puppetry. Seol et al. [61] showed a flexible retargeting scheme that contains both agility and semantics but is limited to kinematic animations.

Our framework has advantages over flexible mapping and real-time control compared to the other methods. Our framework supports various tasks without explicitly modeling task-specific dynamics due to the flexibility of motion retargeting and control schemes. We also achieve robust control by adopting the motion imitation learning with domain randomization.

VII. CONCLUSION AND FUTURE WORK

We presented a human motion control system that allows a user to control quadrupedal robots using motion capture.

The system has two main components: a motion retargeting module and a motion imitation policy. The motion retargeting module translates the captured human motion into robot motion with proper semantics through supervised learning and post-processing techniques. Then we train a control policy that can imitate the given retargeted motion using deep reinforcement learning. We further improve the control performance by leveraging a set of experts and curriculum learning. We evaluate the proposed motion control system on simulated and real-world quadrupedal robots by conducting various tasks, including standing, tilting, sitting, manipulating, walking, or their combinations.

Our work has a few limitations. First, we found that the Kinect's delay of 0.01s to 0.06s is significant for real-time control, particularly for a real robot, preventing us from conducting more real-world experiments in the live mode. While we mitigated this issue by training imitation policies with randomized control frequencies, it could not solve all the raised issues. Moreover, instability of the Kinect estimation system occasionally observes unexpected operator motions which often leads to control failure. We believe a motion capture system with more stability and higher rates, will be more suitable for real-time motion control applications.

Another key observation is that the lack of future trajectory severely degrades the quality of motion imitation. We train policies to track versatile reference motions with frequent changes in target velocities and turning rates, often resulting in conservative policies with poor motion quality. One notable research direction will be to predict user intentions from history and leverage them to improve tracking performance.

We assume that the user and the robot are in the same space. However, we must release this assumption to achieve the goal of developing robotic workers in dangerous environments. We plan to combine the proposed system with virtual reality devices to provide more immersive experiences. This extension will raise new research questions, such as which robot sensory information is essential for users to operate the robot properly and how to deal with increased delays.

VIII. Acknowledgements

We would like to thank Phil Sik Chang and Visak Kumar for their contributions to this work.

References

[1] Sebastian Albrecht, Karinne Ramirez-Amaro, Federico Ruiz-Ugalde, David Weikersdorfer, Marion Leibold, Michael Ulbrich, and Michael Beetz. Imitating human reaching motions using physically inspired optimization principles. In *2011 11th IEEE-RAS International Conference on Humanoid Robots*, pages 602–607. IEEE, 2011.

[2] Miguel Arduengo, Ana Arduengo, Adrià Colomé, Joan Lobo-Prat, and Carme Torras. A robot teleoperation framework for human motion transfer, 2019.

[3] Max Bajracharya, James Borders, Dan Helmick, Thomas Kollar, Michael Laskey, John Leichty, Jeremy Ma, Umashankar Nagarajan, Akiyoshi Ochiai, Josh Petersen, Krishna Shankar, Kevin Stone, and Yutaka Takaoka. A mobile manipulation system for one-shot teaching of complex tasks in homes. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11039–11045, 2020. doi: 10.1109/ICRA40945.2020.9196677.

[4] Ilya Baran, Daniel Vlasic, Eitan Grinspun, and Jovan Popović. Semantic deformation transfer. *ACM Trans. Graph.*, 28(3), jul 2009. ISSN 0730-0301. doi: 10.1145/1531326.1531342. URL https://doi.org/10.1145/1531326.1531342.

[5] Kevin Bergamin, Simon Clavet, Daniel Holden, and James Richard Forbes. Drecon: Data-driven responsive control of physics-based characters. *ACM Trans. Graph.*, 38(6), November 2019. ISSN 0730-0301. doi: 10.1145/3355089.3356536. URL https://doi.org/10.1145/3355089.3356536.

[6] Rainer Bischoff and Volker Graefe. Demonstrating the humanoid robot hermes at an exhibition: A long-term dependability test. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems; Workshop on Robots at Exhibitions*. Lausanne, Switzerland, 2002.

[7] G. Bledt, M. J. Powell, B. Katz, J. Di Carlo, P. M. Wensing, and S. Kim. Mit cheetah 3: Design and control of a robust, dynamic quadruped robot. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2245–2252, 2018.

[8] Jan Carius, René Ranftl, Vladlen Koltun, and Marco Hutter. Trajectory optimization for legged robots with slipping motions. *IEEE Robotics and Automation Letters*, 4(3):3013–3020, 2019. doi: 10.1109/LRA.2019.2923967.

[9] Sungjoon Choi, Matt Pan, and Joohyung Kim. Nonparametric motion retargeting for humanoid robots on shared latent space. *Proceedings of Robotics: Science and Systems (R: SS)*, 2020.

[10] Sungjoon Choi, Min Jae Song, Hyemin Ahn, and Joohyung Kim. Self-supervised motion retargeting with safety guarantee. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8097–8103, 2021. doi: 10.1109/ICRA48506.2021.9560860.

[11] Alexander Clegg, Wenhao Yu, Jie Tan, C. Karen Liu, and Greg Turk. Learning to dress: Synthesizing human dressing motion via deep reinforcement learning. *ACM Trans. Graph.*, 37(6), dec 2018. ISSN 0730-0301. doi: 10.1145/3272127.3275048. URL https://doi.org/10.1145/3272127.3275048.

[12] Ekin D. Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V. Le. Autoaugment: Learning augmentation policies from data, 2019.

[13] Jared Di Carlo, Patrick M. Wensing, Benjamin Katz, Gerardo Bledt, and Sangbae Kim. Dynamic locomotion in the mit cheetah 3 through convex model-predictive control. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–9, 2018. doi: 10.1109/IROS.2018.8594448.

[14] Yanran Ding, Abhishek Pandala, Chuanzheng Li, Young-

Ha Shin, and Hae-Won Park. Representation-free model predictive control for dynamic motions in quadrupeds. *IEEE Transactions on Robotics*, 37(4):1154–1171, 2021. doi: 10.1109/TRO.2020.3046415.

[15] Mira Dontcheva, Gary Yngve, and Zoran Popović. Layered acting for character animation. *ACM Trans. Graph.*, 22(3):409–416, jul 2003. ISSN 0730-0301. doi: 10.1145/882262.882285. URL https://doi.org/10.1145/882262.882285.

[16] Michele Focchi, Victor Barasuol, Marco Frigerio, Darwin G. Caldwell, and Claudio Semini. *Slip Detection and Recovery for Quadruped Robots*, pages 185–199. Springer International Publishing, Cham, 2018. ISBN 978-3-319-60916-4. doi: 10.1007/978-3-319-60916-4_11. URL https://doi.org/10.1007/978-3-319-60916-4_11.

[17] Michele Focchi, Romeo Orsolino, Marco Camurri, Victor Barasuol, Carlos Mastalli, Darwin G. Caldwell, and Claudio Semini. *Heuristic Planning for Rough Terrain Locomotion in Presence of External Disturbances and Variable Perception Quality*, pages 165–209. Springer International Publishing, Cham, 2020. ISBN 978-3-030-22327-4. doi: 10.1007/978-3-030-22327-4_9. URL https://doi.org/10.1007/978-3-030-22327-4_9.

[18] Kevin Frans, Jonathan Ho, Xi Chen, Pieter Abbeel, and John Schulman. Meta learning shared hierarchies. *CoRR*, abs/1710.09767, 2017. URL http://arxiv.org/abs/1710.09767.

[19] Christian Gehring, Stelian Coros, Marco Hutter, Michael Bloesch, Markus A. Hoepflinger, and Roland Siegwart. Control of dynamic gaits for a quadrupedal robot. In *2013 IEEE International Conference on Robotics and Automation*, pages 3287–3292, 2013. doi: 10.1109/ICRA.2013.6631035.

[20] Hartmut Geyer, Andre Seyfarth, and Reinhard Blickhan. Positive force feedback in bouncing gaits? *Proceedings. Biological sciences / The Royal Society*, 270:2173–83, 11 2003. doi: 10.1098/rspb.2003.2454.

[21] Sehoon Ha, Peng Xu, Zhenyu Tan, Sergey Levine, and Jie Tan. Learning to walk in the real world with minimal human effort. *arXiv preprint arXiv:2002.08550*, 2020.

[22] Tuomas Haarnoja, Sehoon Ha, Aurick Zhou, Jie Tan, George Tucker, and Sergey Levine. Learning to walk via deep reinforcement learning. *arXiv preprint arXiv:1812.11103*, 2018.

[23] Nicolas Heess, Dhruva TB, Srinivasan Sriram, Jay Lemmon, Josh Merel, Greg Wayne, Yuval Tassa, Tom Erez, Ziyu Wang, S. M. Ali Eslami, Martin Riedmiller, and David Silver. Emergence of locomotion behaviours in rich environments, 2017.

[24] Daniel Holden, Taku Komura, and Jun Saito. Phase-functioned neural networks for character control. *ACM Trans. Graph.*, 36(4), jul 2017. ISSN 0730-0301. doi: 10.1145/3072959.3073663. URL https://doi.org/10.1145/3072959.3073663.

[25] Daniel Holden, Oussama Kanoun, Maksym Perepichka, and Tiberiu Popa. Learned motion matching. *ACM Trans. Graph.*, 39(4), jul 2020. ISSN 0730-0301. doi: 10.1145/3386569.3392440. URL https://doi.org/10.1145/3386569.3392440.

[26] Tomislav Horvat, Kamilo Melo, and Auke J. Ijspeert. Model predictive control based framework for com control of a quadruped robot. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3372–3378, 2017. doi: 10.1109/IROS.2017.8206176.

[27] Marco Hutter, Christian Gehring, Dominic Jud, Andreas Lauber, C Dario Bellicoso, Vassilios Tsounis, Jemin Hwangbo, Karen Bodie, Peter Fankhauser, Michael Bloesch, et al. Anymal-a highly mobile and dynamic quadrupedal robot. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 38–44. IEEE, 2016.

[28] Jemin Hwangbo, Joonho Lee, and Marco Hutter. Per-contact iteration method for solving contact dynamics. *IEEE Robotics and Automation Letters*, 3(2):895–902, 2018.

[29] Jemin Hwangbo, Joonho Lee, Alexey Dosovitskiy, Dario Bellicoso, Vassilios Tsounis, Vladlen Koltun, and Marco Hutter. Learning agile and dynamic motor skills for legged robots. *Science Robotics*, 4(26):eaau5872, 2019.

[30] Atil Iscen, Ken Caluwaerts, Jie Tan, Tingnan Zhang, Erwin Coumans, Vikas Sindhwani, and Vincent Vanhoucke. Policies modulating trajectory generators. In *2nd Annual Conference on Robot Learning, CoRL 2018*, pages 916–926, 2018. URL https://storage.googleapis.com/pub-tools-public-publication-data/pdf/9c90117f324d3b1bc53779713cb1c800841c926a.pdf.

[31] Yasuhiro Ishiguro, Kunio Kojima, Fumihito Sugai, Shunichi Nozawa, Yohei Kakiuchi, Kei Okada, and Masayuki Inaba. Bipedal oriented whole body master-slave system for dynamic secured locomotion with lip safety constraints. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 376–382, 2017. doi: 10.1109/IROS.2017.8202182.

[32] Fabian Jenelten, Jemin Hwangbo, Fabian Tresoldi, C. Dario Bellicoso, and Marco Hutter. Dynamic locomotion on slippery ground. *IEEE Robotics and Automation Letters*, 4(4):4170–4176, 2019. doi: 10.1109/LRA.2019.2931284.

[33] Kyunam Kim, Patrick Spieler, Elena-Sorina Lupu, Alireza Ramezani, and Soon-Jo Chung. A bipedal walking robot that can fly, slackline, and skateboard. *Science Robotics*, 6(59):eabf8136, 2021.

[34] Nam Hee Kim, Zhaoming Xie, and Michiel Panne. Learning to correspond dynamical systems. In *Learning for Dynamics and Control*, pages 105–117. PMLR, 2020.

[35] Jonas Koenemann and Maren Bennewitz. Whole-body imitation of human motions with a nao humanoid. In *2012 7th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 425–425, 2012. doi: 10.1145/2157689.2157830.

[36] Ashish Kumar, Zipeng Fu, Deepak Pathak, and Jitendra

Malik. Rma: Rapid motor adaptation for legged robots. *arXiv preprint arXiv:2107.04034*, 2021.

[37] Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. Learning quadrupedal locomotion over challenging terrain. *Science Robotics*, 5(47), 2020. doi: 10.1126/scirobotics.abc5986. URL https://robotics.sciencemag.org/content/5/47/eabc5986.

[38] Seunghwan Lee, Moonseok Park, Kyoungmin Lee, and Jehee Lee. Scalable muscle-actuated human simulation and control. *ACM Trans. Graph.*, 38(4), jul 2019. ISSN 0730-0301. doi: 10.1145/3306346.3322972. URL https://doi.org/10.1145/3306346.3322972.

[39] Seyoung Lee, Sunmin Lee, Yongwoo Lee, and Jehee Lee. Learning a family of motor skills from a single motion clip. *ACM Trans. Graph.*, 40(4), 2021.

[40] Tianyu Li, Jungdam Won, Sehoon Ha, and Akshara Rai. Model-based motion imitation for agile, diverse and generalizable quadupedal locomotion. *arXiv preprint arXiv:2109.13362*, 2021.

[41] Zhongyu Li, Xuxin Cheng, Xue Bin Peng, Pieter Abbeel, Sergey Levine, Glen Berseth, and Koushil Sreenath. Reinforcement learning for robust parameterized locomotion control of bipedal robots. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–7, 2021.

[42] Libin Liu and Jessica Hodgins. Learning to schedule control fragments for physics-based characters using deep q-learning. *ACM Trans. Graph.*, 36(4), jun 2017. ISSN 0730-0301. doi: 10.1145/3072959.3083723. URL https://doi.org/10.1145/3072959.3083723.

[43] Libin Liu and Jessica Hodgins. Learning basketball dribbling skills using trajectory optimization and deep reinforcement learning. *ACM Trans. Graph.*, 37(4), jul 2018. ISSN 0730-0301. doi: 10.1145/3197517.3201315. URL https://doi.org/10.1145/3197517.3201315.

[44] Ying-Sheng Luo, Jonathan Hans Soeseno, Trista Pei-Chun Chen, and Wei-Chao Chen. Carl: Controllable agent with reinforcement learning for quadruped locomotion. *ACM Trans. Graph.*, 39(4), jul 2020. ISSN 0730-0301. doi: 10.1145/3386569.3392433. URL https://doi.org/10.1145/3386569.3392433.

[45] Microsoft. Azure kinect dk – develop ai models: Microsoft azure, 2018. URL https://azure.microsoft.com/services/kinect-dk/.

[46] Sehee Min, Jungdam Won, Seunghwan Lee, Jungnam Park, and Jehee Lee. Softcon: Simulation and control of soft-bodied animals with biomimetic actuators. *ACM Trans. Graph.*, 38(6), 2019.

[47] OpenAI, Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Jozefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, Jonas Schneider, Szymon Sidor, Josh Tobin, Peter Welinder, Lilian Weng, and Wojciech Zaremba. Learning dexterous in-hand manipulation, 2019.

[48] Soohwan Park, Hoseok Ryu, Seyoung Lee, Sunmin Lee, and Jehee Lee. Learning predict-and-simulate policies from unorganized human motion data. *ACM Trans. Graph.*, 38(6), 2019.

[49] Xue Bin Peng, Pieter Abbeel, Sergey Levine, and Michiel van de Panne. Deepmimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM Trans. Graph.*, 37(4):143:1–143:14, July 2018. ISSN 0730-0301. doi: 10.1145/3197517.3201311. URL http://doi.acm.org/10.1145/3197517.3201311.

[50] Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018. doi: 10.1109/icra.2018.8460528. URL http://dx.doi.org/10.1109/ICRA.2018.8460528.

[51] Xue Bin Peng, Angjoo Kanazawa, Jitendra Malik, Pieter Abbeel, and Sergey Levine. Sfv: Reinforcement learning of physical skills from videos. *ACM Trans. Graph.*, 37 (6), November 2018.

[52] Xue Bin Peng, Michael Chang, Grace Zhang, Pieter Abbeel, and Sergey Levine. MCP: learning composable hierarchical control with multiplicative compositional policies. *CoRR*, abs/1905.09808, 2019. URL http://arxiv.org/abs/1905.09808.

[53] Xue Bin Peng, Erwin Coumans, Tingnan Zhang, Tsang-Wei Edward Lee, Jie Tan, and Sergey Levine. Learning agile robotic locomotion skills by imitating animals. In *Robotics: Science and Systems*, 07 2020. doi: 10.15607/RSS.2020.XVI.064.

[54] M. H. Raibert. Hopping in legged systems — modeling and simulation for the two-dimensional one-legged case. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-14(3):451–463, 1984.

[55] Marc Raibert, Kevin Blankespoor, Gabriel Nelson, and Rob Playter. Bigdog, the rough-terrain quadruped robot. *IFAC Proceedings Volumes*, 41(2):10822–10825, 2008.

[56] Joao Ramos and Sangbae Kim. Improving humanoid posture teleoperation by dynamic synchronization through operator motion anticipation. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5350–5356, 2017. doi: 10.1109/ICRA.2017.7989629.

[57] Joao Ramos and Sangbae Kim. Dynamic locomotion synchronization of bipedal robot and human operator via bilateral feedback teleoperation. *Science Robotics*, 4(35):eaav4282, 2019. doi: 10.1126/scirobotics.aav4282. URL https://www.science.org/doi/abs/10.1126/scirobotics.aav4282.

[58] Nataniel Ruiz, Samuel Schulter, and Manmohan Chandraker. Learning to simulate, 2019.

[59] Alla Safonova, Nancy Pollard, and Jessica K Hodgins. Optimizing human motion for the control of a humanoid robot. *Proc. Applied Mathematics and Applications of Mathematics*, 78:18–55, 2003.

[60] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[61] Yeongho Seol, Carol O'Sullivan, and Jehee Lee. Creature features: online motion puppetry for non-human characters. In *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 213–221, 2013.

[62] Laura Smith, J. Chase Kew, Xue Bin Peng, Sehoon Ha, Jie Tan, and Sergey Levine. Legged robots that keep on learning: Fine-tuning locomotion policies in the real world, 2021.

[63] Sebastian Starke, He Zhang, Taku Komura, and Jun Saito. Neural state machine for character-scene interactions. *ACM Trans. Graph.*, 38(6), nov 2019. ISSN 0730-0301. doi: 10.1145/3355089.3356505. URL https://doi.org/10.1145/3355089.3356505.

[64] Sebastian Starke, Yiwei Zhao, Taku Komura, and Kazi Zaman. Local motion phases for learning multi-contact character movements. *ACM Trans. Graph.*, 39(4), jul 2020. ISSN 0730-0301. doi: 10.1145/3386569.3392450. URL https://doi.org/10.1145/3386569.3392450.

[65] Sebastian Starke, Yiwei Zhao, Fabio Zinno, and Taku Komura. Neural animation layering for synthesizing martial arts movements. *ACM Trans. Graph.*, 40(4), jul 2021. ISSN 0730-0301. doi: 10.1145/3450626.3459881. URL https://doi.org/10.1145/3450626.3459881.

[66] Wael Suleiman, Eiichi Yoshida, Fumio Kanehiro, Jean-Paul Laumond, and Andre Monin. On human motion imitation by humanoid robot. In *2008 IEEE International Conference on Robotics and Automation*, pages 2697–2704, 2008. doi: 10.1109/ROBOT.2008.4543619.

[67] Richard S. Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Proceedings of the 12th International Conference on Neural Information Processing Systems*, NIPS'99, page 1057–1063, Cambridge, MA, USA, 1999. MIT Press.

[68] Jie Tan, Tingnan Zhang, Erwin Coumans, Atil Iscen, Yunfei Bai, Danijar Hafner, Steven Bohez, and Vincent Vanhoucke. Sim-to-real: Learning agile locomotion for quadruped robots. *arXiv preprint arXiv:1804.10332*, 2018.

[69] Unitree. A1, 2020. URL https://www.unitree.com/products/a1/.

[70] Quan Vuong, Sharad Vikram, Hao Su, Sicun Gao, and Henrik I. Christensen. How to pick the domain randomization parameters for sim-to-real transfer of reinforcement learning policies?, 2019.

[71] John P. Whitney, Tianyao Chen, John Mars, and Jessica K. Hodgins. A hybrid hydrostatic transmission and human-safe haptic telepresence robot. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 690–695, 2016. doi: 10.1109/ICRA.2016.7487195.

[72] Jungdam Won, Jongho Park, Kwanyu Kim, and Jehee Lee. How to train your dragon: Example-guided control of flapping flight. *ACM Trans. Graph.*, 36(6), nov 2017. ISSN 0730-0301. doi: 10.1145/3130800.3130833. URL https://doi.org/10.1145/3130800.3130833.

[73] Jungdam Won, Jungnam Park, and Jehee Lee. Aerobatics control of flying creatures via self-regulated learning. *ACM Trans. Graph.*, 37(6), dec 2018. ISSN 0730-0301. doi: 10.1145/3272127.3275023. URL https://doi.org/10.1145/3272127.3275023.

[74] Jungdam Won, Deepak Gopinath, and Jessica Hodgins. A scalable approach to control diverse behaviors for physically simulated characters. *ACM Trans. Graph.*, 39(4), jul 2020. ISSN 0730-0301. doi: 10.1145/3386569.3392381. URL https://doi.org/10.1145/3386569.3392381.

[75] Zhaoming Xie, Glen Berseth, Patrick Clary, Jonathan Hurst, and Michiel van de Panne. Feedback control for cassie with deep reinforcement learning. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1241–1246. IEEE, 2018.

[76] Zhaoming Xie, Xingye Da, Buck Babich, Animesh Garg, and Michiel van de Panne. Glide: Generalizable quadrupedal locomotion in diverse environments with a centroidal model. *CoRR*, abs/2104.09771, 2021. URL https://arxiv.org/abs/2104.09771.

[77] Katsu Yamane, Yuka Ariki, and Jessica Hodgins. Animating non-humanoid characters with human motion data. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 169–178, 2010.

[78] Chenyu Yang, Bike Zhang, Jun Zeng, Ayush Agrawal, and Koushil Sreenath. Dynamic legged manipulation of a ball through multi-contact optimization. *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7513–7520, 2020.

[79] Chuanyu Yang, Kai Yuan, Qiuguo Zhu, Wanming Yu, and Zhibin Li. Multi-expert learning of adaptive legged locomotion. *Science Robotics*, 5(49), 2020.

[80] Ri Yu, Hwangpil Park, and Jehee Lee. Figure skating simulation from video. In *Computer graphics forum*, volume 38, pages 225–234. Wiley Online Library, 2019.

[81] Wenhao Yu, Greg Turk, and C. Karen Liu. Learning symmetric and low-energy locomotion. *ACM Transactions on Graphics*, 37(4):1–12, Aug 2018. ISSN 1557-7368. doi: 10.1145/3197517.3201397. URL http://dx.doi.org/10.1145/3197517.3201397.

[82] Wenhao Yu, C. Karen Liu, and Greg Turk. Policy transfer with strategy optimization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=H1g6osRcFQ.

[83] Wenhao Yu, Jie Tan, Yunfei Bai, Erwin Coumans, and Sehoon Ha. Learning fast adaptation with meta strategy optimization, 2020.

[84] Yu Zheng and Katsu Yamane. Adapting human motions to humanoid robots through time warping based on a general motion feasibility index. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6281–6288, 2015. doi: 10.1109/ICRA.2015.7140081.