Spatio-Temporal Motion Retargeting for Quadruped Robots

Taerim Yoon, *Graduate Student Member, IEEE*, Dongho Kang, *Graduate Student Member, IEEE*, Seungmin Kim, *Graduate Student Member, IEEE*, Jin Cheng, *Graduate Student Member, IEEE*, Min Sung Ahn, Stelian Coros, *Member, IEEE*, and Sungjoon Choi, *Member, IEEE*

***Abstract*—This work presents a motion retargeting approach for legged robots, aimed at transferring the dynamic and agile movements to robots from source motions. In particular, we guide the imitation learning procedures by transferring motions from source to target, effectively bridging the morphological disparities while ensuring the physical feasibility of the target system. In the first stage, we focus on motion retargeting at the kinematic level by generating kinematically feasible whole-body motions from keypoint trajectories. Following this, we refine the motion at the dynamic level by adjusting it in the temporal domain while adhering to physical constraints. This process facilitates policy training via reinforcement learning, enabling precise and robust motion tracking. We demonstrate that our approach successfully transforms noisy motion sources, such as hand-held camera videos, into robot-specific motions that align with the morphology and physical properties of the target robots. Moreover, we demonstrate terrain-aware motion retargeting to perform BackFlip on top of a box. We successfully deployed these skills to four robots with different dimensions and physical properties in the real world through hardware experiments.**

***Index Terms*—Legged robots, learning from demonstration, motion control, motion retargeting.**

Received 26 February 2025; revised 30 June 2025; accepted 11 July 2025. Date of publication 19 August 2025; date of current version 23 September 2025. This work was supported in part by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT), in part by the Artificial Intelligence Graduate School Program Korea University, 12.5% under Grant RS-2019-II190079, in part by the AI Research Hub Project, 12.5% under Grant RS-2024-00457882, in part by the AI Autonomy and Knowledge Enhancement for AI Agent Collaboration, 12.5% under Grant RS-2022-II220871, in part by the Development of Training and Inference Methods for Goal-Oriented Artificial Intelligence Agent, 12.5% under Grant RS-2022-II220480, in part by the Development of Complex Task Planning Technologies for Autonomous Agents, 50% under Grant RS-2024-00336738, and in part by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme under Grant 866480. This article was recommended for publication by Associate Editor M. Khadiv and Editor P. M. Wensing upon evaluation of the reviewers' comments. *(Corresponding author: Sungjoon Choi.)*

Taerim Yoon, Seungmin Kim, and Sungjoon Choi are with the Department of Artificial Intelligence, Korea University, Seoul, Seongbuk-gu 02841, South Korea (e-mail: taerimyoon@korea.ac.kr; dkslanjdi96@korea.ac.kr; sungjoon–choi@korea.ac.kr).

Dongho Kang, Jin Cheng, and Stelian Coros are with the Department of Computer Science, ETH Zurich, 8092 Zurich, Switzerland (e-mail: kangd@ethz.ch; jicheng@ethz.ch; scoros@ethz.ch).

Min Sung Ahn is with the Department of Mechanical and Aerospace Engineering, UCLA, Los Angeles, CA 90095 USA (e-mail: aminsung@ucla.edu).

This article has supplementary downloadable material available at https://doi.org/10.1109/TRO.2025.3600123, provided by the authors.

Digital Object Identifier 10.1109/TRO.2025.3600123

I. Introduction

LEGGED robots are steadily making their way into human society for their ability to walk alongside humans. As these robots become more prevalent in everyday settings, there is growing interest in generating natural and subtle motions beyond standard walking [1], [2]. In this context, *imitation learning* (IL) has emerged as an effective tool for generating natural motion by imitating prerecorded or hand-crafted motions [3]. For example, safe and attentive behaviors of robotic assistance dogs can be developed by imitating the motions of a service dog captured in a video, where the dog carefully approaches the elderly without causing disruption.

The main challenge of motion imitation lies in overcoming the morphological and dimensional differences between source and target systems [4], [5]. Specifically, when imitating prerecorded animal motion, the disparity between the animal actor and the robot in morphological and physical properties hampers the direct transfer of the motion at the joint trajectory levels. To address this issue, motion imitation involves a process known as *motion retargeting*, which adjusts the target motion to ensure compatibility with the size and morphology of the target robotic system.

Existing motion retargeting methods can transfer and adapt source motions for target systems but often produce kinodynamically infeasible motions. These infeasible motions can result in suboptimal mimicking behaviors or even complete failure in imitation [6], [7]. In addition, the application of these methods is generally limited to motion data that includes whole-body movements with global body pose information. Consequently, this limitation restricts the use of motion data with an unknown coordinate frame, such as animal movements captured by a hand-held camera.

Our work aims to generate physically feasible reference motions to facilitate streamlined and successful learning of control policies that imitate the expressiveness and agility in source movements, as shown in Fig. 1. More specifically, we aim to develop a motion retargeting method that enables transferring motion data lacking global body pose information and with an unknown origin point to target robotic systems.

To this end, we propose *spatio-temporal motion retargeting* (STMR), which transfers baseless keypoint trajectories to the target robot, as shown in Fig. 2(a). The motivation of our approach is to break down motion retargeting problems into

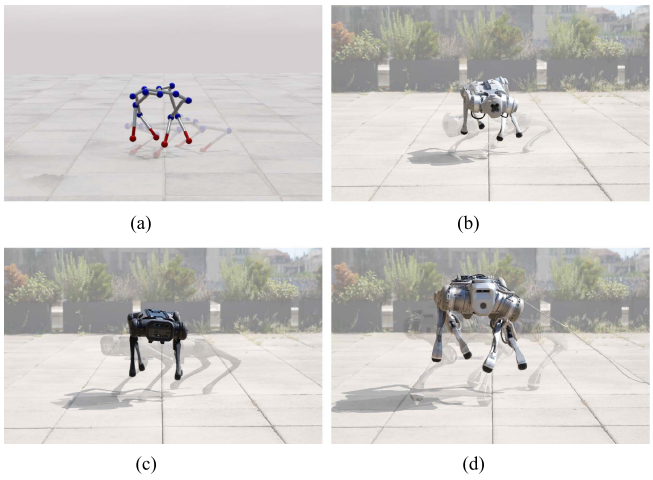


Fig. 1. (a) Hand-crafted HopTurn motion was kino-dynamically retargeted using our method and executed on the (b) *Unitree Go1*, (c) *AlienGo*, and (d) *B2* robots, each with different dimensions and physical properties. (a) Original motion. (b) Unitree Go1. (c) Unitree AlienGo. (d) Unitree B2.

two subproblems in space and time domains, respectively. In more detail, STMR generates whole-body motion with two sequential processes, namely *spatial motion retargeting* (SMR) and *temporal motion retargeting* (TMR). SMR retargets motion at a kinematic level. By regulating kinematic artifacts of foot sliding and foot penetrations, SMR enables the generation of whole-body motion from videos by reconstructing it from baseless keypoint trajectories, as depicted in Fig. 2(b). On the other hand, TMR, as illustrated in Fig. 2(c), focuses on refining the motion subject to dynamics constraints and further deforms the motion in the temporal domain to generate a dynamically feasible motion. This step is particularly crucial for motions that involve flight phases, as motions in Fig. 1, where variations in robot size and actuation power should lead to differences in mid-air duration. In the final step, a feedback control policy is trained through reinforcement learning (RL), guided by a kino-dynamically feasible reference motion to ensure accurate and robust tracking when deployed on real robots.

To demonstrate the efficacy of our approach, we conducted extensive experiments with distinct motions across various quadrupedal robot platforms and compared the results with three baseline methods for motion imitation. In addition, we quantitatively show that motions generated by our STMR method are free of foot sliding and preserve contact schedules. We also showcase that STMR can generate whole-body motion from the relative movement of keypoints and contact schedules, which we refer to as baseless motion. Finally, we demonstrate that a learned control policy can be successfully deployed in the real world on four robots: 1) *Unitree Go1*; 2) *Unitree Go2*; 3) *AlienGo*; and 4) *B2*, each with different dynamic properties and dimensions.

In summary, we propose STMR, which generates a kino-dynamically feasible motion from keypoint trajectories described relative to an unknown reference coordinate frame and facilitates successful IL. The key contributions of our work are summarized as follows.

1) We introduce STMR that transfers motion by adjusting in both spatial and temporal dimensions to ensure the physical feasibility of motion imitation.
2) We present a novel nested optimization framework for TMR, which integrates a model-based controller as an internal process to optimize motion timing.
3) We experimentally show that the motion optimized by STMR leads to successful policy learning for real-world deployment.

II. Related Work

A. Motion Imitation for Quadruped Robots

Developing a legged locomotion controller capable of replicating the agile and natural movements of legged animals has been a longstanding aspiration. To realize this ambition, a body of research has explored the approach of incorporating prerecorded animal motion or hand-crafted motion animation into a legged locomotion control pipeline.

Several studies have demonstrated motion imitation with a model-based legged locomotion control pipeline. Kang et al. [1] employed a simplified dynamics model and a gradients-based optimization method to search for the control sequence, including the footholds of robots, in order to transfer the gait sequences that maintain the nonperiodic and irregular patterns of animal motion. Grandia et al. [8] introduced a nested optimization approach for the retargeting of animal motion by deriving sensitivities in the retargeting parameters, with the goal of creating dynamically feasible target motions. These motions were then executed on a quadruped robot using model predictive control. Zhang et al. [9] presented a motion imitation pipeline that transfers source motions obtained from videos using a pose estimator. In addition, they adopted the contact implicit trajectory optimization (TO) technique to remove the requirement for explicit contact information in the motions. Notably, Kang et al. [10] demonstrated a model-based motion controller that follows high-level joystick commands while preserving animal-like walking styles by incorporating a data-driven motion planning algorithm into the legged locomotion control pipeline.

In another vein, IL has also been a vigorously explored area of research and represents a promising strategy for imitating animal motions. Peng et al. [11] introduced an RL approach that uses a reward function to align the state of a character with prerecorded motion data, enabling the character to perform actions, such as walking, running, and dancing. This methodology was further extended to real-world robotics in later work, showcasing the execution of agile animal movements on a quadruped robot [12]. More recently, the adversarial motion prior (AMP) [13] method was introduced for improved generality and applied for a quadruped robot to walk in a real-world scenario [14]. Inspired by this, Li et al. [15] adopted a similar approach to imitate rough and physically infeasible reference torso motions.

Shifting the focus to animal motion imitation, some studies combine model-based optimal control (MBOC) and IL by leveraging MBOC demonstrations to train RL policies, resulting in dynamic and agile quadruped motions [6], [7], [16], [17], [18]. Notably, Fuchioka et al. [17] employed offline TO to generate

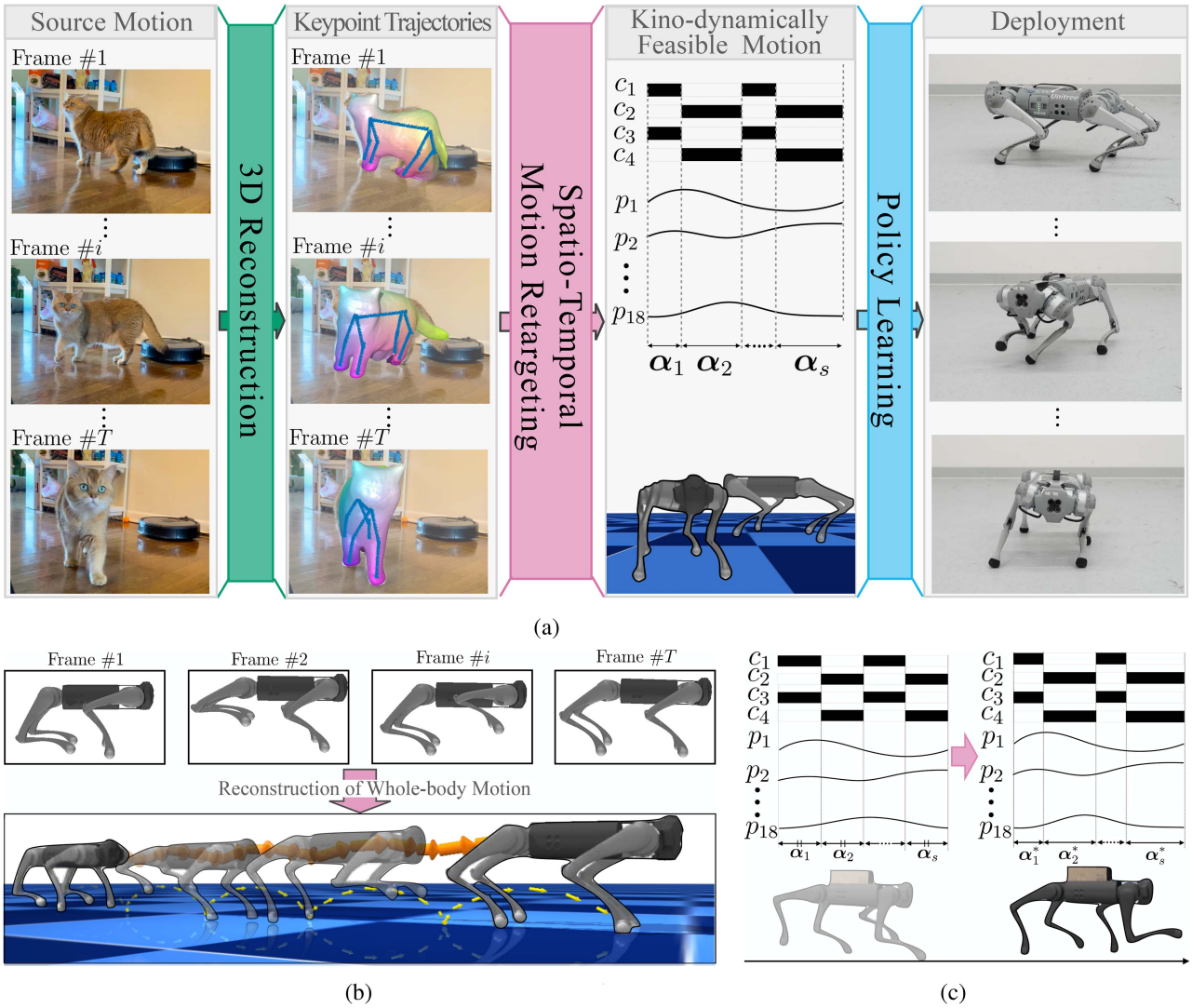


Fig. 2. (a) Our STMR method consists of SMR and TMR stages to generate kino-dynamically feasible motion. (b) In the SMR stage, a kinematically feasible whole-body motion in absolute coordinates is generated, but only keypoint motion is given in local coordinates. (c) In the TMR stage, the temporal aspect of the motion is optimized, and a dynamically feasible motion is generated. As the resulting motion is physically feasible, it can guide the training of control policy toward successful deployment in the real world. (a) Overview reconstruction of whole-body motion. (b) Spatial motion retargeting. (c) Temporal motion retargeting.

complex reference motions, such as quadruped backflipping and executed these motions using a feedback controller trained with IL. Similarly, Kang et al. [18] introduced on-demand reference motion generation through optimal control for efficient and robust IL across various quadruped gait patterns. Liu et al. [19] utilized differential dynamic programming (DDP) to refine motion skills led by policy training.

In this work, we utilize prerecorded animal motion data and hand-crafted animations to replicate the agile and natural movements observed in animals. Similar to the work by Fuchioka et al. [17], Kang et al. [18], and Liu et al. [19], our approach enhances motion imitation by using MBOC to generate optimal control and state data for streamlined IL. However, the difference is that we not only optimize the control and states of the robot but also adjust the target motion, including temporal deformation.

In addition, we demonstrate that our method can retarget motions obtained from videos. This is similar to the work of Zhang et al. [9] in that we refine motions obtained from pose-estimators according to the robot's dynamics. However, our proposed method differs in that it reconstructs the base trajectory of the robot in the global frame by incorporating contact plans, instead of relying on noisy base positions given by the pose-estimator.

B. *Motion Retargeting*

In the context of motion imitation, overcoming the morphological differences between the source and target systems is essential to replicate motion from a system with distinct configurations. In this regard, motion retargeting plays a crucial role by adapting the source motion to be compatible with the target system.

The most intuitive motion retargeting methods often involve utilizing supervised learning with paired motion data between

two source and target configurations. In line with this methodology, Yamane et al. [20] employed a Gaussian process latent model to map human motions to a character directly. Seol et al. [21] presented a technique of blending the retargeted motion with the nearest motion data point to efficiently learn the mapping for motion retargeting. More recently, a mixture of experts were trained on a paired dataset to generate real-time quadruped motions [22]. Meanwhile, Choi et al. [23] proposed a semisupervised learning approach that constructs a latent space of collision-free poses and uses nonparametric regression to enable real-time motion retargeting in the real world. While these methods may appear straightforward, it is important to note that they often require a laborious data collection process, posing challenges in scaling the data for numerous configurations.

Alternatively, another line of work focuses on retargeting motions at the kinematic level by transferring the movement of keypoint trajectories. The work by Choi and Ko [24] is among the first to utilize inverse kinematics (IK) for motion retargeting by following the keypoint trajectories. Choi and Kim [25] further advanced this approach by modifying keypoint trajectories and employing IK to evaluate the deformed motion.

Building upon this foundation, the transfer of keypoint trajectories was further explored using unsupervised learning approaches. Villegas et al. [26] leveraged the differentiability of forward kinematics to transfer motions between human-like characters by matching keypoint movements with adversarial loss. Similarly, the work by Li et al. [27] utilizes keypointwise feature loss and adversarial loss to retarget humanoid motions to nonhumanoid characters. Aberman et al. [28] introduced a concept of the common skeleton to construct an intermediate latent space shared among different kinematic structures using unsupervised learning. Choi et al. [29] proposed a self-supervised learning framework to ensure a safe motion retargeting process, wherein pseudo-labels were acquired through optimization-based motion retargeting approaches.

While the kinematic motion retargeting methods mentioned previously can generate visually convincing motions, incorporating system dynamics can significantly enhance the physical feasibility of motions and streamline their deployment to real robots. Tak and Ko [30] introduced the dynamic motion retargeting filter to regularize motion with zero-moment point constraints for legged figures. Following this, Al Borno et al. [31] employed linear quadratic regulator search trees, and Rouxel et al. [32] used a whole-body optimization under kino-dynamic constraints to track keypoint trajectories of the source motion. As previously mentioned, Grandia et al. [8] employed a nested optimization approach with MBOC to ensure the dynamical feasibility of the retargeted motions.

In this article, we transfer trajectories of local positions of keypoints from a source motion while preserving the contact schedule of the motion. Throughout this process, we account for both the kinematics and dynamics of the target system to generate physically feasible motions. Specifically, our method prevents kinematic artifacts (e.g., foot sliding) while adhering to the system's dynamics and physical constraints. Notably, we refine the motion in the temporal domain by adjusting the time scale. In our experiments, we demonstrate that this temporal optimization generates dynamically feasible motions, successfully transferring motions to a target system with significantly different dimensions.

III. Preliminaries

This section describes two established methods, namely the unit vector method (UVM) [25] and DDP [33]. Since our proposed method produces physically feasible movements by refining a target motion at kinematic and dynamic levels, these techniques are utilized as subprocesses: UVM for kinematic-level motion retargeting and DDP for dynamic-level motion retargeting.

A. Unit Vector Method

The UVM retargets a source motion by preserving the directional unit vector between two adjacent keypoints that move along with the target robot. While this method does not guarantee the kinematic feasibility of the retargeted motion, we use it as a subprocess to generate an initial reference for whole-body motion.

Consider a robot whose joint position is denoted as $\boldsymbol{\theta} \in \mathbb{R}^M$ and keypoint position as $\mathbf{p} \in \mathbb{R}^{N\times 3}$, where M and N are the numbers of joints and keypoints, respectively. The UVM aims to obtain joint position $\boldsymbol{\theta}$ given the keypoint positions of the source system, denoted as ${}^{\text{src}}\mathbf{p}$.

Let us define a parent index $\mathcal{P}(j)$ for the jth keypoint with respect to the kinematic tree. The unit directional vector between the jth keypoint and its parent can be described as $\mathbf{e}_j = (\mathbf{p}_j - \mathbf{p}_{\mathcal{P}(j)})/d_j$ where $d_j := \|\mathbf{p}_j - \mathbf{p}_{\mathcal{P}(j)}\|$ is constant as two keypoints lies on rigid link. By scaling the directional vector $\mathbf{e}_j$ with target link length ${}^{\text{trg}}d_j$, the keypoint position of the target system ${}^{\text{trg}}\mathbf{p}$ is obtained as

$$ {}^{\text{trg}}\mathbf{p}_j = {}^{\text{trg}}\mathbf{p}_{\mathcal{P}(j)} + {}^{\text{trg}}d_j\,{}^{\text{src}}\mathbf{e}_\mathbf{j}. \tag{1} $$

Subsequently, the joint position $\boldsymbol{\theta}$ is obtained by solving IK

$$ \boldsymbol{\theta} = \text{IK}({}^{\text{trg}}\mathbf{p}_{1:N}). $$

B. Differential Dynamic Programming

DDP is an effective approach for finding control inputs that achieve user-defined objectives while satisfying the system dynamics model and physical constraints. As the proposed retargeting problem involves finding dynamically feasible motions, we utilize DDP as an internal subprocess for dynamic-level motion retargeting. Specifically, we utilize an iterative linear quadratic Gaussian [34], a variant of DDP.

We denote the state of the robot as $\mathbf{x}$, the control input as $\mathbf{u}$, and the dynamics of the robot as f. The discrete-time dynamics at the ith step is described as $\mathbf{x}^{i+1} = f(\mathbf{x}^i, \mathbf{u}^i)$. The primary goal of DDP is to find the optimal control inputs $\mathbf{u}^{0:h-1}$ and states $\mathbf{x}^{0:h}$ given the target states $\bar{\mathbf{x}}^{0:T}$ under the system dynamics f. This can be represented as (2), where we minimize the objective function comprising the sum of the running cost l_i and the final

cost l_f

$$\min_{\mathbf{x}^{0:h},\,\mathbf{u}^{0:h-1}} \quad \sum_{i=0}^{h-1} l_i(\mathbf{x}^i,\mathbf{u}^i;\bar{\mathbf{x}}^i) + l_f(\mathbf{x}^h;\bar{\mathbf{x}}^h)$$
$$\text{s.t.} \quad \mathbf{x}^{i+1} = f(\mathbf{x}^i,\mathbf{u}^i). \tag{2}$$

We define the optimal value function (also known as the optimal cost to go) at the ith step as follows:

$${}^*V^i(\mathbf{x}) = \min_{\mathbf{x}^{i:h},\,\mathbf{u}^{i:h-1}} \quad \sum_{j=i}^{h-1} l(\mathbf{x}^j,{}^*\mathbf{u}^j;\bar{\mathbf{x}}^j) + l_f(\mathbf{x}^h;\bar{\mathbf{x}}^h)$$
$$\text{s.t.} \quad \mathbf{x}^{j+1} = f(\mathbf{x}^j,\mathbf{u}^j)$$
$$\mathbf{x}^i = \mathbf{x}.$$

According to the Bellman optimality principle, the relationship between the optimal value function at i and $i+1$ steps can be expressed as

$${}^*V^i(\mathbf{x}) = \min_{\mathbf{u}} \left[l_i(\mathbf{x},\mathbf{u}) + {}^*V^{i+1}\left(f(\mathbf{x},\mathbf{u})\right)\right]. \tag{3}$$

Furthermore, we define the state-action value function Q^i, which is derived by perturbing the state-action pair $(\mathbf{x}^i,\mathbf{u}^i)$ around the minimum

$$Q^i(\delta\mathbf{x},\delta\mathbf{u}) = l_i(\mathbf{x}+\delta\mathbf{x},\mathbf{u}+\delta\mathbf{u}) - l_i(\mathbf{x},\mathbf{u}) + V^{i+1}(f(\mathbf{x}+\delta\mathbf{x},\mathbf{u}+\delta\mathbf{u})) - V^{i+1}(f(\mathbf{x},\mathbf{u})).$$

For brevity, we drop the step-index i and use V' for V^{i+1}. Then, we expand Q with second-order approximation with the coefficients

$$Q_{\mathbf{x}} = l_{\mathbf{x}} + f_{\mathbf{x}}^T V'_{\mathbf{x}} \tag{4a}$$
$$Q_{\mathbf{u}} = l_{\mathbf{u}} + f_{\mathbf{u}}^T V'_{\mathbf{x}} \tag{4b}$$
$$Q_{\mathbf{xx}} = l_{\mathbf{xx}} + f_{\mathbf{x}}^T V'_{\mathbf{xx}} f_{\mathbf{x}} + V'_{\mathbf{x}} \cdot f_{\mathbf{xx}} \tag{4c}$$
$$Q_{\mathbf{uu}} = l_{\mathbf{uu}} + f_{\mathbf{u}}^T V'_{\mathbf{uu}} f_{\mathbf{u}} + V'_{\mathbf{x}} \cdot f_{\mathbf{uu}} \tag{4d}$$
$$Q_{\mathbf{ux}} = l_{\mathbf{ux}} + f_{\mathbf{u}}^T V'_{\mathbf{xx}} f_{\mathbf{x}} + V'_{\mathbf{x}} \cdot f_{\mathbf{ux}} \tag{4e}$$

where the subscripts denote differentiation. With this approximation, the optimal control modification ${}^*\delta\mathbf{u}$ is obtained as

$${}^*\delta\mathbf{u} = -Q_{\mathbf{uu}}^{-1}(Q_{\mathbf{u}} + Q_{\mathbf{ux}}\delta\mathbf{x}) \tag{5}$$

and by ignoring the second-order derivative of the dynamics (i.e., $f_{\mathbf{xx}}, f_{\mathbf{uu}}, f_{\mathbf{ux}}$), each coefficient can be recursively obtained with

$$\Delta V(i) = -\frac{1}{2} Q_{\mathbf{u}} Q_{\mathbf{uu}}^{-1} Q_{\mathbf{u}} \tag{6a}$$
$$V_{\mathbf{x}}(i) = Q_{\mathbf{x}} - Q_{\mathbf{u}} Q_{\mathbf{uu}}^{-1} Q_{\mathbf{ux}} \tag{6b}$$
$$V_{\mathbf{xx}}(i) = Q_{\mathbf{xx}} - Q_{\mathbf{xu}} Q_{\mathbf{uu}}^{-1} Q_{\mathbf{ux}}. \tag{6c}$$

For more detailed derivations, see the previous work by Mayn [33].

IV. Problem Formulation

In this section, we introduce essential notations to formulate the STMR problem. In the latter part of this section, we effectively solve the proposed STMR problem by decoupling it into two stages: 1) SMR; and 2) TMR. In essence, the SMR problem addresses only the kinematic aspects, while the TMR problem considers the dynamics of the target robotic system.

A. Notations

Consider a robot with M joints whose joint position is denoted as $\boldsymbol{\theta} \in \mathbb{R}^M$. The base position of the robot is denoted as $\mathbf{p}_{\text{b}} \in \mathbb{R}^3$, and base orientation represented with quaternion is denoted as $\mathbf{h} \in \mathbb{H}$, where $\mathbb{H}$ is unit quaternion space. The generalized coordinate of the robot is defined by stacking these values denoted as $\mathbf{q} = [\mathbf{p}_{\text{b}}, \mathbf{h}, \boldsymbol{\theta}]$. Similarly, we write linear, angular, and joint velocity as $\mathbf{v} \in \mathbb{R}^3, \mathbf{w} \in \mathbb{R}^3$, and $\dot{\boldsymbol{\theta}} \in \mathbb{R}^M$, respectively. The time derivative of generalized coordinate is accordingly defined as $\dot{\mathbf{q}} = [\mathbf{v}, \mathbf{w}, \dot{\boldsymbol{\theta}}]$ and the state of the robot is defined as $\mathbf{x} = [\mathbf{q}, \dot{\mathbf{q}}]^T$.

We denote the keypoint positions as $\mathbf{p} \in \mathbb{R}^{N\times 3}$ and N is the number of keypoints. Specifically, we focus on $N = 16$ keypoints consisting of four hips, thighs, knees, and feet. These values are specified for each frame, with the maximum frame index denoted by T. For instance, the position of the jth keypoint in the ith frame is denoted as $\mathbf{p}_j^i$, where $j \in [1, 2, \ldots, N]$ and $i \in [0, 1, 2, \ldots, T]$. Furthermore, we define foot index $\kappa(\cdot)$ to represent position of four feet as $\mathbf{p}_{\kappa(1)}, \mathbf{p}_{\kappa(2)}, \mathbf{p}_{\kappa(3)}, \mathbf{p}_{\kappa(4)}$.

We write forward kinematics as FK_j, which maps $\mathbf{q}$ to the global position of the jth keypoint as $\mathbf{p}_j = \text{FK}_j(\mathbf{q})$, and its Jacobian matrix as $\mathbf{J}_j$, where $\mathbf{J}_j = \frac{\partial(\text{FK}_j(\mathbf{q}))}{\partial \mathbf{q}} \in \mathbb{R}^{(M+6)\times 3}$. We note that, in the subsequent chapter, we abuse the forward kinematic notation without the subscript as $\mathbf{p} = \text{FK}(\mathbf{x})$ to denote a mapping between the full state $\mathbf{x}$ and the concatenated keypoint positions $\mathbf{p}$, for brevity.

B. Spatio-Temporal Motion Retargeting

As the keypoint trajectory $\mathbf{p}$ is acquired from an arbitrary quadruped actor, it can be physically infeasible for the target robot to track. Furthermore, the keypoint trajectory $\mathbf{p}$ may not contain the base movement required for whole-body imitation. To this end, we propose STMR, which regenerates the physically feasible whole-body motion by optimizing both spatial and temporal dimensions.

Let us define the temporal parameters as $\boldsymbol{\alpha} \in \mathbb{R}^S$ that scale the keypoint motion along the time axis, which we refer to as *temporal deformation*, as shown in Fig. 3(a). More specifically, we divide the source motion into S segments and temporally scale each segment by the factor corresponding to each component of $\boldsymbol{\alpha}$. We define this operation as the temporal deformation function $s_{\boldsymbol{\alpha}}(t)$, which maps control time to the corresponding time in the source motion. Through this operation, T frames of keypoint trajectories correspond to $T_{\boldsymbol{\alpha}}$ control time steps. In addition, we define the linear interpolation function $\text{LI}(t; \mathbf{p}^{0:T})$ which computes the keypoint positions at continuous time $t \in \mathbb{R}$

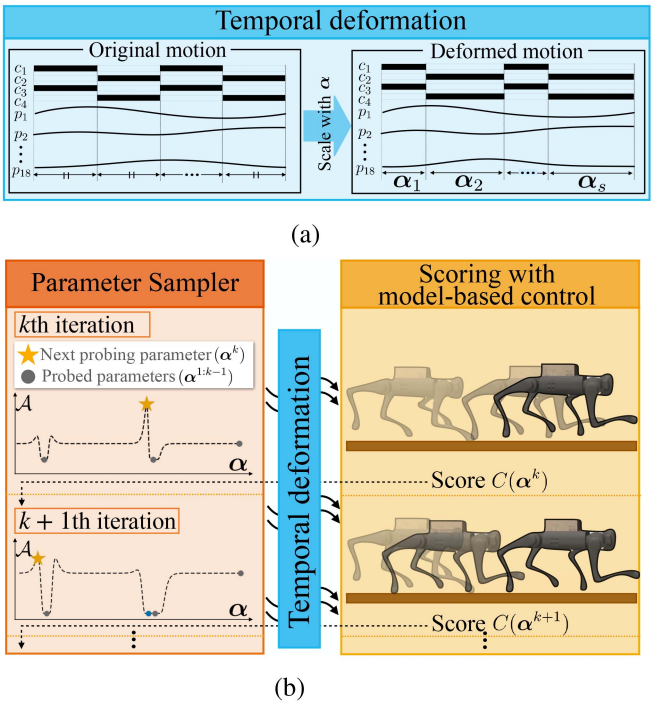


Fig. 3. Overview of TMR is illustrated. (a) Temporal deformation involves splitting the motion into equal segments and scaling with temporal parameter $\boldsymbol{\alpha}$. (b) Using acquisition function $\mathcal{A}$, the temporal parameter $\boldsymbol{\alpha}$ is sampled for temporal deformation and scored by model-based control. (a) Temporal deformation. (b) Temporal motion retargeting.

by linearly interpolating the keypoint trajectories. Therefore, $\mathrm{LI}(s_{\boldsymbol{\alpha}}(t_k); \mathbf{p}^{0:T})$ gives the keypoint positions at the kth control time step, which serves as control targets.

Then, we formulate an optimization problem as described in the following equation, where search for the optimal temporal parameters ${}^*\boldsymbol{\alpha}$, control sequence ${}^*\mathbf{u}^{0:T_{*\boldsymbol{\alpha}}}$, and robot states ${}^*\mathbf{x}^{0:T_{*\boldsymbol{\alpha}}}$:

$$\begin{aligned}\min_{\mathbf{u}^{0:T_{\boldsymbol{\alpha}}},\mathbf{x}^{0:T_{\boldsymbol{\alpha}}},\boldsymbol{\alpha}\in\mathcal{I}} \quad & \frac{1}{2T_{\boldsymbol{\alpha}}}\sum_{k=0}^{T_{\boldsymbol{\alpha}}} \left\|\mathrm{FK}(\mathbf{x}^k) - \mathrm{LI}\left(s_{\boldsymbol{\alpha}}(t_k); \mathbf{p}^{0:T}\right)\right\|_Q^2 \\ \text{s.t.} \quad & \mathbf{x}^{k+1} = f(\mathbf{x}^k, \mathbf{u}^k) \\ & h(\mathbf{x}^k, \mathbf{u}^k) \le 0 \\ & g_{\text{eq}}(\mathbf{x}^k) = 0 \\ & g_{\text{in}}(\mathbf{x}^k) \le 0 \end{aligned} \tag{7}$$

where t_k is the control time at the kth time step and $\mathcal{I}$ is the predefined bound for the temporal parameters $\boldsymbol{\alpha}$.

In the STMR problem, the system is constrained by the dynamics f and other physical constraints h, such as torque limit, Coulomb friction cone constraints, so on. In addition, we introduce foot constraints g_{eq} and g_{in} that prevent foot sliding and ground penetration, while enforcing identical contact timing to the original motion. The details of the foot constraints will be described in Section V-A.

It is worth noting that the STMR problem involves transforming the target motion to make it feasible for the robot to imitate rather than simply tracking the motion. Therefore, the retargeted motion should be constructed to preserve the semantic meaning of the original motion. For that reason, we deform the motion in the bounded temporal regions $\mathcal{I}$ to preserve the overall expressiveness of the motion.

Solving this optimization problem is challenging due to its complexity, which involves nonconvex objectives and constraints. To address this, we divide this problem into two subproblems: 1) SMR; and 2) TMR. This decomposition simplifies the problem, even though it remains nonconvex, as detailed in the following sections.

C. Spatio-Temporal Decoupling

Due to the challenges mentioned earlier, we decompose the STMR problem into separate spatial and temporal components. With this approach, we sequentially determine mappings for each component through a two-stage optimization process.

In the first stage, the SMR stage, we focus on the kinematics of the motion. Starting from the STMR problem in (7), we exclude the dynamics f and temporal parameters $\boldsymbol{\alpha}$, concentrating on the kinematic feasibility of the position-level state $\mathbf{q}$ by enforcing the foot constraints g. We search for kinematic feasible states $\bar{\mathbf{q}}$ by minimizing the objective function under the foot constraints g, as shown in (8). Specifically, foot constraints are applied to prevent foot sliding and penetration while ensuring identical contact timing. See Section V-A for more details on these foot constraints.

$$\begin{aligned}\min_{\mathbf{q}^{0:T}} \quad & \frac{1}{2T}\sum_{i=0}^{T} \|\mathrm{FK}(\mathbf{q}^i) - \mathbf{p}^i\|_Q^2 \\ \text{s.t.} \quad & g_{\text{eq}}(\mathbf{q}^i) = 0 \\ & g_{\text{in}}(\mathbf{q}^i) \le 0 \end{aligned} \tag{8}$$

Following this, we compute the trajectory of keypoints corresponding to $\bar{\mathbf{q}}$ as $\bar{\mathbf{p}} = \mathrm{FK}(\bar{\mathbf{q}})$. We then perform temporal deformation with $s_{\boldsymbol{\alpha}}$ on the newly obtained keypoint trajectory. Finally, we solve TMR problem, presented in the following equation, to search for the optimal temporal parameters ${}^*\boldsymbol{\alpha}$, control sequence ${}^*\mathbf{u}^{0:T_{*\boldsymbol{\alpha}}}$, and resulting states ${}^*\mathbf{x}^{0:T_{*\boldsymbol{\alpha}}}$:

$$\begin{aligned}\min_{\mathbf{u}^{0:T_{\boldsymbol{\alpha}}},\mathbf{x}^{0:T_{\boldsymbol{\alpha}}},\boldsymbol{\alpha}\in\mathcal{I}} \quad & \frac{1}{2T_{\boldsymbol{\alpha}}}\sum_{k=0}^{T_{\boldsymbol{\alpha}}} \left\|\mathrm{FK}(\mathbf{x}^k) - \mathrm{LI}\left(s_{\boldsymbol{\alpha}}(t_k); \bar{\mathbf{p}}^{0:T}\right)\right\|_Q^2 \\ \text{s.t.} \quad & \mathbf{x}^{i+1} = f(\mathbf{x}^k, \mathbf{u}^k) \\ & h(\mathbf{x}^k, \mathbf{u}^k) \le 0 \\ & g_{\text{eq}}(\mathbf{x}^k) = 0 \\ & g_{\text{in}}(\mathbf{x}^k) \le 0. \end{aligned} \tag{9}$$

Note that the TMR problem encompasses a finite-horizon optimal control problem (OCP), which finds the optimal control sequence to track the given reference states under dynamics. By ignoring the temporal parameter $\boldsymbol{\alpha}$, the TMR problem reduces to OCP with the goal of tracking the reference keypoint trajectory $\mathbf{p}^{0:T}$. From this insight, we reformulate this problem as a nested

Algorithm 1: Spatial Motion Retargeting.

$\mathbf{q}^0 \leftarrow {}^{\text{IK}}\mathbf{q}^0$
$\tilde{\mathbf{p}} \leftarrow ProjectGround(\text{FK}(\mathbf{q}))$
${}^{\text{IK}}\dot{\mathbf{q}} \leftarrow Differentiate({}^{\text{IK}}\mathbf{q}^1, {}^{\text{IK}}\mathbf{q}^0)$
for $i = 1$ **to** N **do**
  ${}^{\text{IK}}\dot{\mathbf{q}} \leftarrow Differentiate({}^{\text{IK}}\mathbf{q}^i, {}^{\text{IK}}\mathbf{q}^{i+1}, \Delta t)$
  if not $\text{ANY}(\mathbf{c}^i)$ **then** ▷ Flight phase
    if $\text{ANY}(\mathbf{c}^{i-1})$ **then**
      $\mathbf{v}_{\text{exit}} \leftarrow Polyfit(BasePosition(\mathbf{q}^{0:i}))$
    $BaseVelocity({}^{\text{IK}}\dot{\mathbf{q}}) \leftarrow \mathbf{v}_{\text{exit}}$
    $\mathbf{v}_{\text{exit}} \leftarrow \mathbf{v}_{\text{exit}} + \mathbf{g}\Delta t$
  $\tilde{\mathbf{q}} \leftarrow Integrate(\mathbf{q}^{i-1}, {}^{\text{IK}}\dot{\mathbf{q}}, \Delta t)$
  repeat
  $\mathbf{J} \leftarrow GetJacobian(\mathbf{q})$
  $\dot{\mathbf{q}} \leftarrow SolveQP(\mathbf{J}, \mathbf{q}, \tilde{\mathbf{q}}, \mathbf{c}, K)$ ▷ (14)
  $\mathbf{q}^i \leftarrow \mathbf{q}^i + \eta\dot{\mathbf{q}}$
  until $\dot{\mathbf{q}} < \dot{\mathbf{q}}_{\text{thres}}$
  for $j = 1$ **to** 4 **do**
    $\mathbf{p}_{\kappa(j)} = \text{FK}(\mathbf{q})_{\kappa(j)}$
    if $\mathbf{p}_{\kappa(j)^z} < HeightMap(\mathbf{p}^{xy}_{\kappa(j)})$ **then**
      $\mathbf{c}^i_j \leftarrow$ True
    if $\mathbf{c}^i_j$ **and not** $\mathbf{c}^{i-1}_j$ **then** ▷ At the new contact segment
      $\tilde{\mathbf{p}}_j \leftarrow ProjectGround(\mathbf{p}_{\kappa(j)})$

optimization problem to reduce the complexity:

$$\min_{\boldsymbol{\alpha}\in\mathcal{I}} \left(\begin{array}{ll} \min\limits_{\mathbf{u}^{0:T_{\boldsymbol{\alpha}}}, \mathbf{x}^{0:T_{\boldsymbol{\alpha}}}} & \frac{1}{2T_{\boldsymbol{\alpha}}} \sum\limits_{k=0}^{T_{\boldsymbol{\alpha}}} \left\| \text{FK}(\mathbf{x}^k) - \text{LI}\left(s_{\boldsymbol{\alpha}}(t_k); \bar{\mathbf{p}}^{0:T}\right) \right\|^2_Q \\ \text{s.t.} & \mathbf{x}^{i+1} = f(\mathbf{x}^k, \mathbf{u}^k), \\ & h(\mathbf{x}^k, \mathbf{u}^k) \leq 0, \\ & g_{\text{eq}}(\mathbf{x}^k) = 0, \\ & g_{\text{in}}(\mathbf{x}^k) \leq 0 \end{array} \right).$$

As illustrated in Fig. 3(b), we iteratively search for the optimal set of ${}^*\boldsymbol{\alpha}$, ${}^*\mathbf{x}^{0:T_{*\alpha}}$, and ${}^*\mathbf{u}^{0:T_{*\alpha}}$. For each iteration of the outer loop, we begin with a given value of $\boldsymbol{\alpha}$ and solve the inner optimization for the ${}^*\mathbf{x}^{0:T_{\boldsymbol{\alpha}}}$ and ${}^*\mathbf{u}^{0:T_{\boldsymbol{\alpha}}}$. In the next iteration, we update $\boldsymbol{\alpha}$ and repeat the process until a minimum is reached. The resulting robot state ${}^*\mathbf{x}^{0:T_{*\alpha}}$ serves as the kino-dynamically retarged motion. More details on TMR are discussed in Section VI.

Furthermore, it is worth noting that we solve this internal OCP using the off-the-shelf simulator [35] using full dynamics model f instead of the reduced model. This approach makes introducing a new target robot straightforward, enabling motion retargeting to arbitrary quadruped robots. However, solving optimal control with full dynamics model f can be challenging because of high-dimensional state space. In this article, we utilize DDP from Section III-B, which is known to be effective in such scenarios [36].

V. Spatial Motion Retargeting

In this section, we provide more details on SMR that retargets motion at the kinematic level. One simple way for kinematic motion retargeting is the UVM, as outlined in Section III-A, that maintains the directional unit vector between adjacent keypoints. However, this approach may introduce undesired artifacts, such as foot sliding or foot penetration. For instance, Fig. 4(a) illustrates that transferring a walking trajectory by a short-legged robot to a long-legged robot leads to unnatural foot sliding and penetration. In addition, the UVM cannot generate a base trajectory beyond direct transfer, making it unable to adjust the base trajectory to fit the robot's kinematic configuration. More critically, this naive transfer using the UVM can alter contact schedules, failing to preserve the semantic meaning of the original motion.

On the other hand, our SMR, illustrated in Fig. 4(b), eliminates foot sliding and foot penetration, adjusts base trajectory and heights according to the target robot's kinematics, and preserves contact schedules of the source motion. As briefly mentioned in Section IV, we introduce the foot constraints, formulated as two constraints: 1) the *contact preservation constraint*, which ensures that the contact schedule of the source and retargeted motion remain identical without any foot penetration; and 2) the *foot locking constraint*, which prevents foot sliding during contact phases. By enforcing these constraints, SMR aims to obtain a refined generalized coordinate trajectory of robot $\bar{\mathbf{q}}^{0:T}$ that mimics the source motion. The overall algorithm of SMR is summarized in Algorithm 1.

Let us elaborate more on the intuition of SMR in whole-body motion reconstruction from baseless motion. A key idea is minimizing the positional difference of the locally defined baseless motion while optimizing the whole-body motion. However, without additional constraints, this often results in a trivial solution where the robot flounders in the air. Introducing foot constraints helps mitigate this issue by effectively regularizing the resulting motions. Intuitively speaking, we are anchoring the robot's feet while adjusting its joint angles to induce base movement. The whole-body motion is reconstructed by repeatedly detaching and reanchoring the feet according to a given contact schedule while adjusting each joint. In essence, *foot constraints are not just an outcome of* SMR*but a fundamental component in reconstructing whole-body motion.*

In practice, SMR can be used to retarget motion in two scenarios: 1) when the base trajectory is provided; and 2) when it is not. In the first case, SMR efficiently suppresses kinematic artifacts when given whole-body motion, as evaluated in Section VIII-D. In the second case, SMR reconstructs whole-body motion by incorporating contact schedules, even when only the local movements of keypoints are known, which we evaluate in Section VIII-E. Moreover, we demonstrate that the base trajectory generation is powerful enough to adapt the motion to the given terrain, as shown in Section IX-C.

A. Foot Constraints

Let the jth foot of a robot be in a contact phase. We introduce the anchor position of the jth foot as $\tilde{\mathbf{p}}_j$, which is a projection of $\text{FK}_j(\mathbf{q})$ to the ground. During the contact phase, the height of the foot should match the elevation of the projection point, and in a swing phase, it should be positioned above this point.

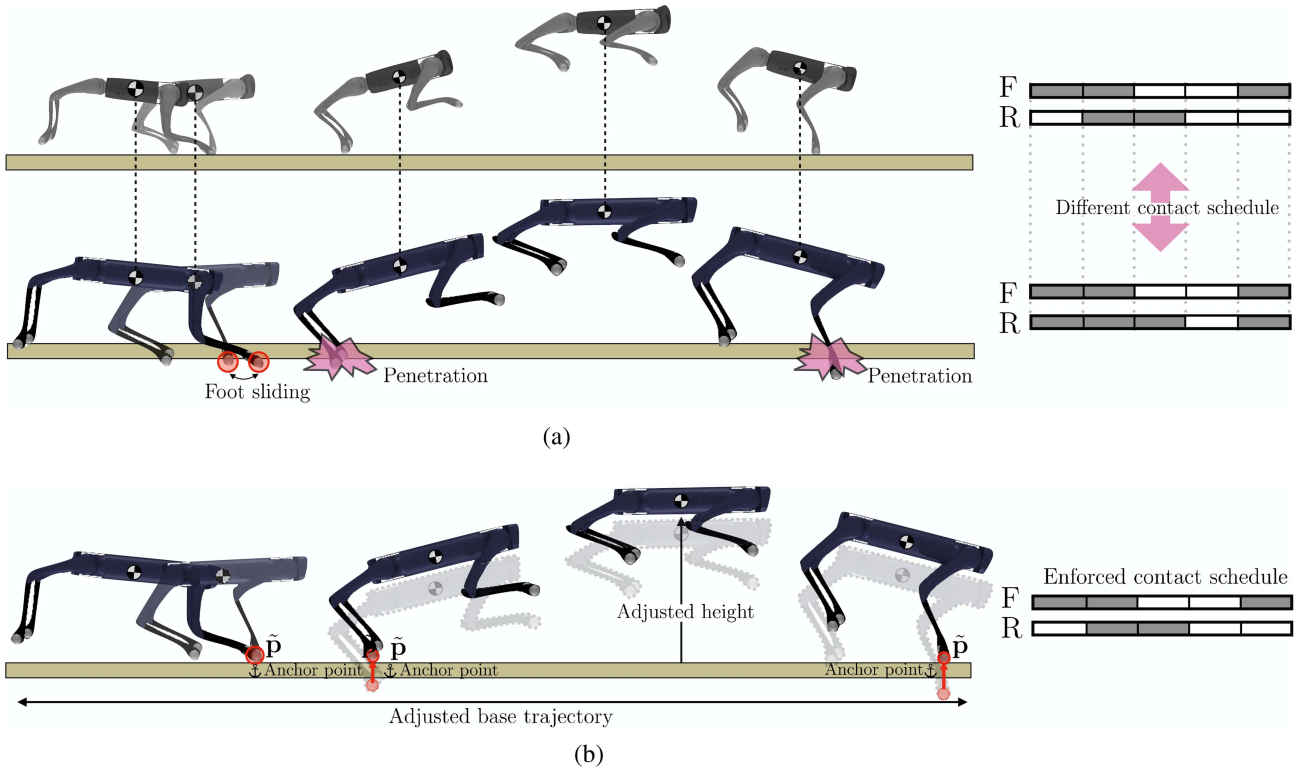


Fig. 4. Illustration of baseline method and SMR. (a) The baseline method (i.e., UVM) can lead to kinematic artifacts, such as foot sliding, foot penetration, and mismatched contact timing. (B) On the other hand, SMR generates kinematically feasible motion by appropriately anchoring the foot position as $\tilde{\mathbf{p}}$. (a) Baseline method. (b) Spatial motion retargeting.

This condition can be expressed as

$$\begin{cases} \mathrm{FK}_j(\mathbf{q})^z = \tilde{\mathbf{p}}_j^z, & \text{if } c_j \\ \mathrm{FK}_j(\mathbf{q})^z > \tilde{\mathbf{p}}_j^z, & \text{else} \end{cases} \tag{10}$$

where z represents the height component of the position vector, and c_j is a contact Boolean for the jth foot.

In addition, we fix the x and y coordinates of the foot position to the anchor point during the contact phase to eliminate foot sliding as follows:

$$\mathrm{FK}_j(\mathbf{q})^{xy} = \tilde{\mathbf{p}}_j^{xy} \quad \text{if } c_j \tag{11}$$

we combine (10) and (11), and relax them as follows, which we refer to as $g(\mathbf{q})$:

$$c_j(\mathrm{FK}_j(\mathbf{q}) - \tilde{\mathbf{p}}_j) = 0. \tag{12}$$

B. Objective Function

On top of the relaxation in the foot constraints, we make the objective function in (8) convex. We start by incorporating the relaxed constraints g obtained in (12) into (8) and search for the optimal generalized coordinate sequence $\bar{\mathbf{q}}^{0:T}$ that minimize the positional distance to keypoints as described in the following equation:

$$\begin{aligned} \bar{\mathbf{q}}^{0:T} = \arg\min_{\mathbf{q}^{0:T}} \quad & \frac{1}{2T}\sum_{i=0}^{T} \|\mathrm{FK}(\mathbf{q}^i) - \mathbf{p}^i\|_Q^2 \\ \text{s.t.} \quad & g(\mathbf{q}^i) = 0. \end{aligned} \tag{13}$$

Without the foot constraints g, (13) becomes a typical unconstrained IK problem. Instead of solving (13) directly, we solve the unconstrained IK problem to obtain the generalized coordinate solution ${}^{\mathrm{IK}}\mathbf{q}^{0:T}$ and then compute its time derivative ${}^{\mathrm{IK}}\dot{\mathbf{q}}^{0:T}$ by finite difference. In the subsequent steps, the time derivative of IK solutions serves as the velocity target for the final optimization problem. This allows us to transform (13) into a velocity-level problem, which is more straightforward to solve.

More specifically, we build the reference $\tilde{\mathbf{q}}^i$, which is obtained by time-integrating the current coordinate $\mathbf{q}^i$ with ${}^{\mathrm{IK}}\dot{\mathbf{q}}^i$. In addition, we write scaled error between reference coordinates $\tilde{\mathbf{q}}^i$ and current coordinate $\mathbf{q}^i$ as $\delta\tilde{\mathbf{q}}^i = K_q(\tilde{\mathbf{q}}^i - \mathbf{q}^i)$, where K_q is a tunable parameter. Finally, we form the optimization problem mimicking the reference $\tilde{\mathbf{q}}^i$ for a single frame, subject to the linearized foot constraints, as follows:

$$\begin{aligned} \dot{\tilde{\mathbf{q}}}^i = \arg\min_{\dot{\mathbf{q}}^i} \quad & \frac{1}{2}\|\dot{\mathbf{q}}^i - \delta\tilde{\mathbf{q}}^i\|_Q^2 \\ \text{s.t.} \quad & \mathbf{J}_j^i\dot{\mathbf{q}}^i = c_j K_p(\tilde{\mathbf{p}}_j - \mathrm{FK}_j(\mathbf{q}^i)) \quad j \in [1,2,3,4]. \end{aligned} \tag{14}$$

We solve this problem sequentially, starting from the initial frame to the last. This approach has the additional benefit of simplifying the determination of the foot anchor point $\tilde{\mathbf{p}}$. Together with the linearized foot constraints, this approach ensures that the system effectively satisfies the foot constraints.

The final form of the problem is a convex optimization problem, which allows us to use off-the-shelf solvers. In this

work, we utilize the alternating direction method of multipliers (ADMM) [37] method implemented by Stellato et al. [38] .

C. Handling Flight Phases

As SMR heavily relies on contact schedules, it is crucial to handle scenarios where all of a robot's feet are in swing phases (i.e., flight phases). In such cases, we assume the robot's base follows a ballistic trajectory.

As described in Algorithm 1, we fit a polynomial function to the history of base trajectories when a flight phase is detected. Specifically, choose the degree of the polynomial based on the current time step clipped by the maximum horizon of h. Then, we calculate the exit velocity $\mathbf{v}_{\text{exit}}$ by taking the derivative of the polynomial function. We update the velocity by integrating gravitational acceleration $\mathbf{g}$, allowing the robot to follow the ballistic trajectory until the contact schedule is set or new contact occurs between the robot and the terrain.

In addition, note that such a ballistic trajectory is set for the base trajectory of reference motion $\tilde{\mathbf{q}}$. Since we solve the optimization process to determine the final kinematic posture $\mathbf{q}$, the whole-body motion will be adjusted to correspond to the ballistic base trajectory of the reference motion $\tilde{\mathbf{q}}$.

VI. Temporal Motion Retargeting

In this section, we elaborate on TMR, which generates dynamically feasible motions for the target robot by determining the temporal parameters $\boldsymbol{\alpha}$ and control sequence $\mathbf{u}^{0:T_\alpha}$. The main challenge of TMR lies in jointly optimizing both temporal parameters $\boldsymbol{\alpha}$ and control sequence $\mathbf{u}^{0:T_\alpha}$. As outlined in Section IV-B, we approach this as a nested optimization problem, where we iteratively search for the optimal temporal parameters ${}^*\boldsymbol{\alpha}$ using Bayesian optimization (BO) [39]. In detail, TMR involves the repetition of three processes, as shown in Fig. 3(b): 1) parameter sampling; 2) temporal deformation; and 3) scoring.

For a warm start, we begin by randomly sampling the temporal parameter within the interval $\mathcal{I} = [\boldsymbol{\alpha}_{\min}, \boldsymbol{\alpha}_{\max}]$. Subsequently, we perform temporal deformation, denoted as $s = s_{\boldsymbol{\alpha}}(t)$, which divides the time frame into S equal intervals and scales the motion according to the temporal parameters $\boldsymbol{\alpha}$, as illustrated in Fig. 3(a).

Subsequently, we employ MBOC to track the deformed motion and evaluate the result with the scoring function denoted as $C(\cdot)$. The key intuition is that appropriately deforming the motion makes it easier for MBOC to track, leading to improved tracking performance. Specifically, we design the scoring function to evaluate not only keypoint tracking and contact matching but also to regularize extreme values of the temporal parameters, enhancing motion imitation performance in practice. Equation (15) presents the metric $C(\boldsymbol{\alpha})$ for evaluating the tracking performance which incorporates measurements of contact differences using Intersection over Union (IoU), base positional error (L1 distance d_b), and base orientation error (L1 distance of Euler angles d_E), where $\mathbf{p}_\text{b}$ and $\mathbf{h}$ represent base position and orientation, respectively.

$$C(\boldsymbol{\alpha}) = -d_b(\mathbf{p}_\text{b}, \bar{\mathbf{p}}_\text{b}) - d_E(\mathbf{h}, \bar{\mathbf{h}}_\text{b}) + w_c \text{IoU}(\mathbf{c}, \bar{\mathbf{c}}). \tag{15}$$

Denoting the temporal parameter at the kth iteration as $\boldsymbol{\alpha}^k$ and its corresponding score as C^k, the next probing point $\boldsymbol{\alpha}^{k+1}$ is sampled based on the previous probing points $(\boldsymbol{\alpha}^{1:k}, C^{1:k})$. To achieve this, we first fit a surrogate function s using Gaussian process regression

$$\begin{gathered} s(\boldsymbol{\alpha}) \sim \mathcal{GP}(s_\mu(\boldsymbol{\alpha}), s_\sigma(\boldsymbol{\alpha})) \\ s_\mu = \mathbf{K}_*^T \mathbf{K}^{-1} C^{1:k}, \quad s_\sigma = \mathbf{K}_{**} - \mathbf{K}_*^T \mathbf{K}^{-1} \mathbf{K}_* \end{gathered} \tag{16}$$

where $\mathbf{K}$, $\mathbf{K}_*$, and $\mathbf{K}_{**}$ are defined as follows, using a Matern kernel $\mathbf{K}(\cdot,\cdot)$:

$$\begin{gathered} \mathbf{K} = \mathbf{K}(\boldsymbol{\alpha}^{1:k}, \boldsymbol{\alpha}^{1:k}) \\ \mathbf{K}_* = \mathbf{K}(\boldsymbol{\alpha}, \boldsymbol{\alpha}^{1:k}), \qquad \mathbf{K}_{**} = \mathbf{K}(\boldsymbol{\alpha}, \boldsymbol{\alpha}). \end{gathered} \tag{17}$$

We then construct an acquisition function $\mathcal{A}$ using expected improvement [39], where $\hat{\boldsymbol{\alpha}}$ represents the best parameter found in the history, ϕ denotes the Gaussian distribution, Φ is the cumulative distribution function (CDF) of ϕ, and ξ controls the degree of exploration

$$\begin{aligned} \mathcal{A} &= \Delta s \Phi(\Delta s / s_\sigma) + s_\sigma \phi(\Delta s / s_\sigma) \\ \Delta s &= s_\mu(\boldsymbol{\alpha}) - s_\mu(\hat{\boldsymbol{\alpha}}) + \xi, \\ \hat{\boldsymbol{\alpha}} &= \underset{\boldsymbol{\alpha} \in \{\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \ldots, \boldsymbol{\alpha}_k\}}{\arg\max} \; C(\boldsymbol{\alpha}). \end{aligned} \tag{18}$$

Finally, we sample the next probing parameter $\boldsymbol{\alpha}^{k+1}$ by maximizing the acquisition function $\boldsymbol{\alpha}^{k+1} = \arg\max_{\boldsymbol{\alpha}} \mathcal{A}(\boldsymbol{\alpha})$. This process is repeated until a value of $\boldsymbol{\alpha}$ converges.

We note that any dynamics model capable of producing whole-body motion can be used for MBOC. In our experiments, we used a full-body model implemented through the MuJoCo engine [35], [36], as MuJoCo provides the derivative information required for DDP described in Section III-B. Since we use a full-body model, it only requires a universal robot description file or its extensions. This allows us to avoid the tedious modeling process typically required for reduced models.

VII. Residual Policy Learning

The optimal control sequence identified in the STMR stage can successfully produce the retargeted motion in simulation with open-loop control execution. However, to deploy the motions robustly in the real world, a feedback policy is necessary to overcome uncertainties and model mismatches. Therefore, we adopt residual policy learning [40] to guide the feedback policy learning process. The residual policy denoted as π, computes a closed-loop control signal that is added to a base control ${}^*\mathbf{x}^{0:T_\alpha}$ obtained through STMR. As illustrated in Fig. 5, the joint values of feasible motion ${}^*\boldsymbol{\theta}$ are obtained directly from indexing the feasible motion ${}^*\mathbf{x}^{0:T_\alpha}$ where the residual control policy outputs the joint correction denoted as $\Delta\boldsymbol{\theta}$. Both terms are fed to the PD controller to yield a motor torque command as $\tau = K_p({}^*\boldsymbol{\theta} + \Delta\boldsymbol{\theta} - \boldsymbol{\theta}) - K_d\dot{\boldsymbol{\theta}}$, where K_p and K_d are the proportional gains and derivative gains, respectively, and $\boldsymbol{\theta}$ and $\dot{\boldsymbol{\theta}}$ are the current robot's joint angles and velocities.

The residual policy is trained with RL, where the rewards at frame i are defined as a summation of tracking measures for joint position $\boldsymbol{\theta}^i_{1:M}$, base position $\mathbf{q}^i_\text{b}$, base orientation $\mathbf{h}^i$, the

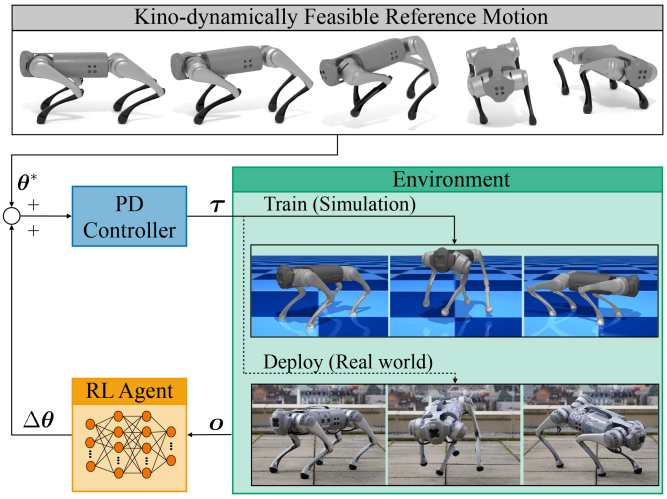


Fig. 5. Control policy with residual learning

TABLE I
LIST OF THE HYPERPARAMETERS USED FOR RL TRAINING

Hidden dimensions (Actor)	[512, 256, 128]
Hidden dimensions (Critic)	[512, 256, 128]
Activation function	Exponential Linear Unit (ELU)
Weight coefficient for Entropy term	0.01
Learning rate	1.0e-3
Discount factor	0.99
Optimizer	ADAM [41]
Number of policy iterations	10,000
Joint reward (w_q, $\beta_{\mathbf{q_j}}$)	(3.0, 2.0)
Base position reward (w_b, β_b)	(3.0, 5.0)
Base orientation reward ($w_\mathbf{h}$, $\beta_\mathbf{h}$)	(3.0, 1.0)
Keypoints reward($w_\mathbf{p}$, $\beta_\mathbf{p}$)	(30.0, 10.0)
K_p, K_d (A1)	(30.0, 1.0)
K_p, K_d (Go1)	(30.0, 1.0)
K_p, K_d (AlienGo)	(50.0, 1.0)
K_p, K_d (B2)	(200.0, 10.0)

position of keypoints $\mathbf{p}^i_{1:N}$

$$\begin{aligned} r^i_t = w_q \exp[\beta_{\mathbf{q_j}} \|\bar{\boldsymbol{\theta}}^i_{1:M} - \boldsymbol{\theta}^i_{1:M}\|] \\ + w_\mathrm{b} \exp[\beta_\mathrm{b} \|\bar{\mathbf{p}_\mathrm{b}}^i - {\mathbf{p}_\mathrm{b}}^i\|] \\ + w_\mathbf{h} \exp[\beta_\mathbf{h} \|\bar{\mathbf{h}}^i \ominus \mathbf{h}^i\|] \\ + w_\mathbf{p} \exp[\beta_\mathbf{p} \|\bar{\mathbf{p}}^i_{1:N} - \mathbf{p}^i_{1:N}\|]. \end{aligned}$$

The values of the hyperparameters used in our experiments are presented in Table I.

The observation space, denoted as $\boldsymbol{o}$, is defined in (19) which consists of projected gravity $\mathbf{g}_\mathrm{proj}$, joint positions $\boldsymbol{\theta}$, joint velocity $\dot{\boldsymbol{\theta}}$, last deployed torque $\boldsymbol{\tau}$, phase variable ϕ, and base height $\mathbf{p}^z_\mathrm{b}$:

$$\boldsymbol{o} = [\mathbf{g}_\mathrm{proj}, \boldsymbol{\theta}, \dot{\boldsymbol{\theta}}, \boldsymbol{\tau}, \phi, \mathbf{p}^z_\mathrm{b}]. \tag{19}$$

To facilitate learning dynamic motions involving flying phases, we adopt the *reference state initialization* scheme from DeepMimic [11], which determines the initial state of each episode through uniform sampling from the reference motion.

VIII. Simulational Experiments

We conducted a series of simulation experiments to verify the efficacy of our approach. In the first experiment, we evaluated the tracking performance of the final motion execution on simulated robots and demonstrated the superiority of our method by comparing it to baseline IL methodologies. In the second experiment, we validated that the motion transferred by our method is free of foot sliding and preserves the contact schedules. We then demonstrate that our method can reconstruct the base movements from baseless motions and quantify the recovery rate. Finally, we highlight the importance of temporal optimization using the BackFlip motion, a highly dynamic movement that includes a significant flight phase.

A. Training Details

All RL control policies used in our experiments were trained with simulational data generated by Isaac Gym [42]. We employed proximal policy optimization [43] with 10 000 iterations, equivalent to approximately 50 million data samples requiring one hour of training with NVIDIA RTX A6000 GPU. For each experiment, we trained five policies with five different random seeds and reported the mean and standard deviation of the metrics. More details on training configuration are summarized in Table I.

B. Details of Baseline Methods

We selected three other motion retargeting baselines, including downstream IL methods, to evaluate how the motions retargeted in both space and time domains lead to more successful IL.

1) UVM (DeepMimic): We aim to show that motion retargeted by the proposed method leads to better policy learning than the UVM. In detail, we employ the UVM and direct base position transfer for motion retargeting and train the control policy to imitate the retargeted motions. The downstream control policy learning is the same as STMR, including the reward function from Section VII. Therefore, the main difference between this baseline method and STMR is using the UVM for motion retargeting. This retargeting schema is essentially equivalent to DeepMimic [11], which employs motion retargeting through the direct transfer of joint angles, except that we do not scale the base position manually for each robot.

2) Adverserial Motion Prior (AMP): We use the AMP [14] approach as a second baseline method to evaluate whether motion retargeting plays a critical role in successful motion imitation or if improved control policy learning alone is sufficient. By leveraging a reward derived from a learned discriminator, the AMP approach can partially circumvent the need for dynamically feasible reference motions, as demonstrated by Li et al. [15]. Consequently, we apply the motion obtained through the UVM in conjunction with AMP's control policy learning and compare the results to those of the proposed method.

For AMP, we trained a discriminator D to classify whether the transition between the current state s and the next state s' is generated by the agent or from a reference motion. The output

TABLE II
NORMALIZED KEYPOINT L1 DISTANCE WITH DYNAMIC TIME WARPING (%)

Motion	Robot	UVM (DeepMimic) [11]	UVM (AMP) [13]	TO [17]	SMR (ours)	STMR (ours)
SideSteps	Go1	4.0(0.6)	6.4(6.1)	3.5(0.6)	5.3(1.3)	**1.9**(0.3)
	A1	7.2(1.9)	5.3(3.8)	5.0(0.7)	5.4(0.2)	**2.7**(0.7)
	AlienGo	3.7(0.3)	6.9(2.9)	2.7(0.4)	3.7(0.6)	**1.7**(0.3)
HopTurn	Go1	6.5(1.2)	7.9(0.2)	6.1(0.3)	6.4(1.0)	**2.1**(0.2)
	A1	8.2(1.5)	12.2(5.2)	4.4(0.8)	8.2(0.7)	**1.6**(0.2)
	AlienGo	5.3(1.9)	8.5(0.7)	2.1(0.3)	5.2(1.4)	**1.7**(0.2)

TABLE III
SOURCE AND DURATION OF THE MOTION DATASET ARE ILLUSTRATED

	Trot0	Trot1	Pace0	Pace1	SideSteps	HopTurn
Source	Mcp	Mcp	Mcp	Mcp	Anim	Anim
Duration	1.65	1.65	1.95	2.50	14.50	9.10

Mcp refers to motion capture data, and Anim refers to handmade animation data. The durations are given in seconds.

from the discriminator D contributes an additional reward term with the corresponding weight w_{ad} as

$$r = r_t + w_{\text{ad}} \log D(s, s'). \tag{20}$$

As a side note, we extended the training iterations to 25 000 for AMP policies to achieve convergence.

3) TO: Our method is comparable to *OptMimic* [17], as both approaches use MBOC to obtain a dense description of dynamic motion. Specifically, OptMimic employs a single rigid body model [44] and contact-implicit TO with a nonconvex optimizer [45] to refine the reference motion. In more detail, they incorporate constraints to avoid foot penetration, enforce friction cones, and complementary conditions for contacts. Lastly, this baseline follows the same control policy learning framework, as described in Section VIII-B1, making a fair comparison.

A key distinction between our proposed method and this baseline lies in the scope of optimization: while OptMimic focuses on spatial dimensions, our approach incorporates optimization in both spatial and temporal dimensions. Therefore, we demonstrate the impact of temporal optimization by comparing the performance of our method against this baseline.

4) STMR (Ours): As mentioned in Section V, STMR can retarget motions with and without the global base pose trajectory. In the simulation experiments, we used the base pose trajectories in Section VIII-C and Section VIII-D, while we reconstructed the base pose trajectory solely from local movement in Section VIII-E.

We set the number of time segments to $S = 1$, as the tested motions for Section VIII-C are relatively short, as shown in Table III. This is possible because DDP, which serves as the internal process for TMR, can provide more fine-grained temporal adjustments. Note that in Section IX-B, we set the number of time segments to $S = 3$ as the reference motions include different phases of motions, such as slow walking led by quick turn. Lastly, the bounds for the temporal parameters are set as $\log_2 \boldsymbol{\alpha} \in [-1, 1]$, meaning the motion can be scaled by a factor of up to two in either direction.

C. Evaluating Motion Tracking Performance

We compare the tracking performance of baseline methods and the proposed method to evaluate how appropriate the retargeted motions are. The intuition of measuring tracking performance is that if retarget motions are dynamically feasible for the target robot, it will be straightforward for the controller to follow them, resulting in a smaller tracking error. Similarly, better tracking performance indicates a reduced likelihood of imitation failure, as a policy that fails to reproduce a source motion on a robot results in significantly lower tracking performance. We utilize dynamic time warping (DTW) [46] to measure distance irrelevant to temporal deformation, avoiding the risk of underevaluating baseline methods. Details on the motion retargeting method used for each IL approach are presented in Section VIII-B.

We used motion clips collected from real dogs from Zhang et al. [47] and quadruped motions crafted manually by animators from Peng et al. [12]. As these motions are collected from diverse sources, we demonstrate the versatility of our method. Regarding the complexity of these motions, they can be ranked in ascending order as follows: Trot, Pace, SideSteps, and HopTurn. Trot motion is comparatively more straightforward to replicate than Pace as it involves cross-arranging two feet for enhanced stability. Both Pace and SideSteps involve statically unstable postures. However, SideSteps poses a greater challenge due to the need to balance against lateral momentum. Lastly, HopTurn is the most complex, as it requires jumping and a mid-air maneuver to turn. In terms of the target robot, we employ three different sizes of quadrupedal robots (Unitree A1, Unitree Go1, and Unitree AlienGo). In addition, we elongated the duration of each motion with a scale of two to highlight the impact of temporal deformation. More details on each motion are summarized in Table II.

Fig. 6 illustrates tracking performance in terms of mean and standard deviation over five random seeds. STMR exhibits exceptional performance across all six motions, achieving an average tracking error of 48.7 mm. In comparison, the average errors for the three baseline methods: 1) UVM (DeepMimic); 2) UVM (AMP); 3) TO; and 4) SMR, are 168.3, 275.3, 88.4, and 154.2 mm, respectively. Thus, our method's improvement in tracking error corresponds to 71.1%, 82.3%, 44.9%, and 68.4% reductions for each baseline method, corresponding to an average improvement of 66.7% . Moreover, the result shows that STMR can perform the two most challenging motions with flight phases (i.e., SideSteps and HopTurn), whereas other

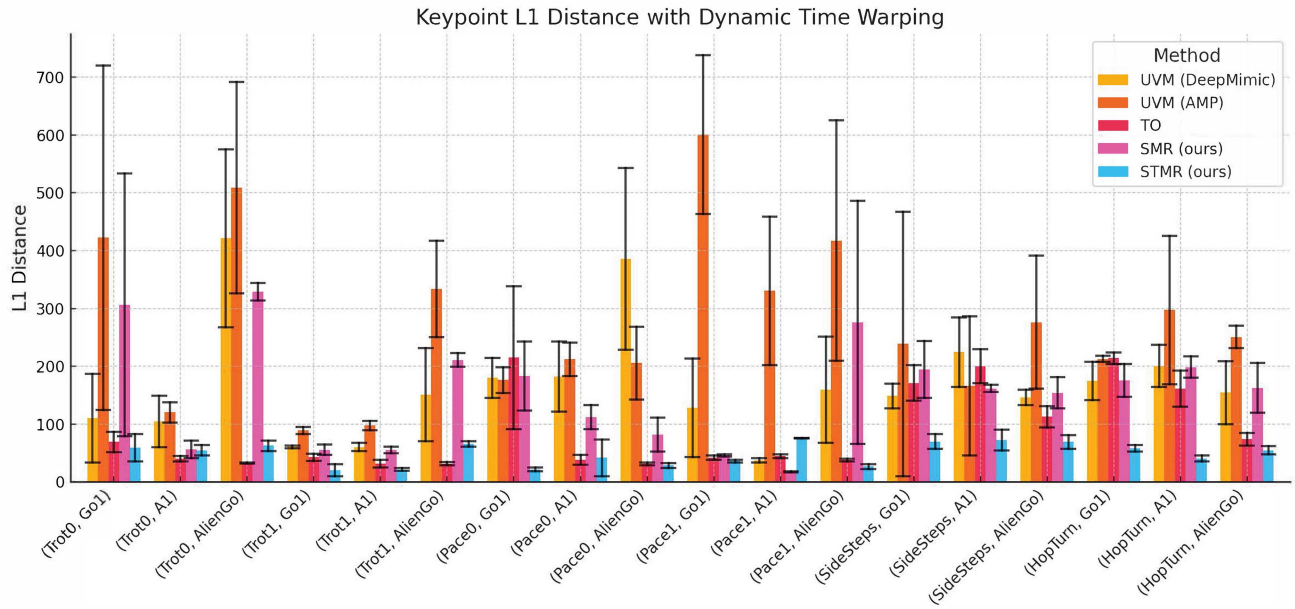


Fig. 6. Tracking performance for each robot and motion is illustrated in terms of the mean and standard deviation.

baseline methods fail.[1] We also report the normalized keypoint tracking error (L1 distance) expressed as a percentage of the total trajectory length (measured in meters), computed using DTW. Bold values indicate the best performance. Values in parentheses denote standard deviations.

The TMR process requires significantly more computation than SMR. Therefore, it is essential to evaluate how each component contributes to performance. Therefore, we also conduct ablation study of STMR by evaluating only the SMR stage. As shown in Fig. 6, TMR enhances the overall dynamic feasibility of the motions. In particular, we highlight that SMR fails for HopTurn and SideSteps, suggesting that TMR plays a crucial role in the flight phase by adjusting motion in the temporal dimension.

D. Evaluating Foot Constraints Enforcement

SMR enforces foot constraints to generate kinematically feasible motions that preserve the original contact schedules and avoid foot sliding. Therefore, we quantitatively evaluate foot sliding and contact preservation using the same six motions from the previous Section VIII-C. We measure foot sliding by calculating the L1 distance of position between the beginning and end of the contact segment, where continuous contact longer than 0.5 s is marked as the contact segment. In addition, we measure the contact preservation through the IoU between the contact schedules of the original and retargeted motion. An IoU of 1.0 between two motions signifies identical contact schedules, indicating successful contact preservation, whereas an IoU of 0.0 indicates completely divergent contact schedules.

To evaluate foot slide regularization and contact preservation, we compare against the UVM as the baseline method, where the results are summarized in Table IV. The proposed method shows an average foot sliding of 0.34 mm, whereas that of the baseline method is 73.92 mm across six motions and three robots. Furthermore, the proposed method also shows significant improvement in contact preservation, with an average IoU of 0.998 compared to 0.513 for the baseline method. Given that the foot sliding error and the IoU achieve near-optimal values, the results strongly suggest that the motions generated by our method effectively eliminate foot sliding and accurately preserve contact schedules.

Moreover, our method outperforms TO in both foot sliding prevention and contact preservation. This is because foot sliding constraints are not explicitly enforced in the TO framework, as doing so would distort the motion's semantic meaning. For

TABLE IV
EVALUATION OF FOOT SLIDING AND CONTACT PRESERVATION

Robot	Motion	Foot slide (mm)↓			IoU ↑		
		UVM [25]	TO [17]	Ours	UVM [25]	TO [17]	Ours
Go1	Trot0	110.19	51.72	**0.15**	0.46	0.40	**1.00**
	Trot1	72.93	82.71	**0.09**	0.48	0.38	**1.00**
	Pace0	88.90	205.46	**0.14**	0.47	0.62	**1.00**
	Pace1	61.33	73.71	**0.09**	0.53	0.89	**0.99**
	SideSteps	34.74	6.12	**0.03**	0.60	0.95	**1.00**
	HopTurn	33.35	11.87	**0.05**	0.59	0.82	**1.00**
A1	Trot0	101.39	118.31	**0.15**	0.44	0.47	**1.00**
	Trot1	86.29	135.82	**0.12**	0.50	0.39	**1.00**
	Pace0	83.29	68.54	**0.12**	0.47	0.75	**1.00**
	Pace1	63.58	100.06	**0.11**	0.53	0.82	**1.00**
	SideSteps	41.37	13.00	**0.04**	0.58	0.84	**1.00**
	HopTurn	37.19	12.01	**0.05**	0.49	0.79	**1.00**
AlienGo	Trot0	147.01	65.27	**0.07**	0.46	0.48	**1.00**
	Trot1	112.22	141.84	**2.89**	0.54	0.40	**0.98**
	Pace0	114.51	24.38	**0.10**	0.49	0.76	**1.00**
	Pace1	71.88	59.56	**0.09**	0.53	0.85	**1.00**
	SideSteps	26.34	0.32	**0.04**	0.50	0.84	**1.00**
	HopTurn	44.11	14.44	**1.83**	0.58	0.79	**1.00**

[1]The footage of each comparison experiment can be found in the attached video.

instance, when retargeting a casual walking motion from a large robot to a smaller one, enforcing foot sliding constraints locks the feet to the ground, forcing unnatural leg spreading to match the global motion. This alteration fundamentally changes the motion, making it no longer resemble casual walking. In addition, TO employs contact-implicit TO, allowing contact Booleans to change for improved tracking. However, this flexibility leads to deviations from the original contact schedules, resulting in a lower IoU. In summary, STMR outperforms TO by generating a base trajectory tailored to each robot, enabling direct incorporation of constraints without conflicting with the global motion.

E. Evaluating Reconstruction of Whole-Body Motion

Our STMR method is capable of generating the whole-body motion from the baseless keypoint trajectories, as described in Fig. 2(b). In this experiment, we quantitatively evaluate this capability by removing base motions from a set of source motions, reconstructing them, and measuring the recovery rate. Moreover, we highlight that this reconstruction process generates base trajectories tailored to the kinematics of each of the three different robots, allowing for the efficient transfer of motions while overcoming morphological differences. [2]

1) Data Collection: We collected Go1's motion data of walking forward with various gait patterns and velocities using MBOC [34]. Specifically, the gaits include Pace (1.0 m/s), Bound (1.0 m/s), and Trot (0.5 m/s, 1.0 m/s, 1.5 m/s). Each motion sequence includes 3 s of locomotion with 0.5 s of standing at the beginning and end, respectively. Furthermore, we selected forward walking motions for this experiment to evaluate the reconstruction capability by comparing the travel distance.

2) Motion Reconstruction: We evaluate the motion reconstruction capability by removing the base trajectory $\mathbf{p}_{\mathrm{b}}^{0:T}$ and reconstructing it from the keypoint trajectories $\mathbf{p}^{0:T}$. In detail, we focus on the distance traveled and measure positional differences with respect to the motions before clearing the base position.

Table V shows the distance traveled for each motion generated by SMR from whole-body motions and baseless motions. In detail, SideStep travel distance is measured laterally, while the rest are measured longitudinally. Setting the motions generated from whole-body motion as ground truth, we calculate the recovery rate. The average recovery rate for Go1, A1, and AlienGo was 75.19%, 74.40%, and 78.46%, showing that SMR can reconstruct the base trajectory regardless of configurations.

3) Motion Retargeting by Reconstruction: As illustrated in Fig. 7, SMR adjusts the base trajectory and subsequent local movements appropriately for the target robot. We evaluate how the base's travel distance changed accordingly, as shown in Table V. On average, AlienGo exhibited a travel distance that was 13.15% and 17.68% longer than Go1 for the motions retargeted from whole-body motion and baseless motion, respectively. This increase roughly corresponds to the size difference between Go1 and AlienGo, where the horizontal lengths of Go1 and AlienGo are 540 and 610 mm, respectively, with a 12.96% difference. Similarly, A1 showed a travel distance that was 4.67% and 6.22% shorter than Go1. This decrease also roughly corresponds to the size difference, where the horizontal length of A1 is 7.41% shorter than that of Go1.

[2]The footage of motion reconstruction can be found in the attached video.

TABLE V
EVALUATION OF SPATIAL MOTION RETARGETING WITH AND WITHOUT THE BASE MOTION

Robot	Motion	Distance Traveled by Retargeted Motions(m)		
		With Base Motion	Without Base Motion	Recovery Rate (%)
Go1 (Source)	Trot (0.5m/s)	0.97	0.76	78.35
	Trot (1.0m/s)	1.93	1.49	77.20
	Trot (1.5m/s)	2.70	1.93	71.48
	Pace (1.0m/s)	1.91	1.47	76.96
	Bound (1.0m/s)	1.92	1.42	73.96
	SideSteps	0.243	0.125	51.44
A1 (Target)	Trot (0.5m/s)	0.93	0.72	77.42
	Trot (1.0m/s)	1.84	1.40	76.09
	Trot (1.5m/s)	2.59	1.82	70.27
	Pace (1.0m/s)	1.80	1.36	75.56
	Bound (1.0m/s)	1.83	1.33	72.68
	SideSteps	0.234	0.167	71.80
AlienGo (Target)	Trot (0.5m/s)	1.12	0.91	81.25
	Trot (1.0m/s)	2.18	1.74	79.82
	Trot (1.5m/s)	3.05	2.28	74.75
	Pace (1.0m/s)	2.16	1.72	79.63
	Bound (1.0m/s)	2.16	1.67	76.85
	SideSteps	0.254	0.171	67.32

F. Evaluating Flight-Phase Dynamic Motion

TMR employs temporal optimization to determine the optimal timing of motion, enabling the dynamic execution of HopTurn and SideSteps, as demonstrated in Section VIII-C. To further emphasize the importance of temporal optimization, we experiment with the BackFlip motion, which is even more dynamic and includes a longer flight phase. Specifically, we manually create 12 kinematic frames using the Unitree Go1 and apply temporal optimization to generate the whole-body motion, as shown in Fig. 8(a).

We retarget this motion to B2 with and without temporal optimization to clarify its impact. As shown in Fig. 8(b), B2 fails to perform a BackFlip without temporal optimization, resulting in a collision with the ground. In contrast, Fig. 8(c) illustrates that B2 successfully performs the BackFlip when temporal optimization is applied. More specifically, the original motion has a duration of 1.48s, whereas the temporally optimized motion for B2 extends to 2.18s, representing a 47% increase.

IX. REAL-WORLD EXPERIMENTS

We deployed a set of motions retargeted with our STMR method for four robot hardware platforms: 1) Go1; 2) Go2; 3) AlienGo; and 4) B2, using the learned feedback control policies. The control policy runs at 33.33 Hz, successfully responding in real time to execute the dynamic motions of HopTurn and

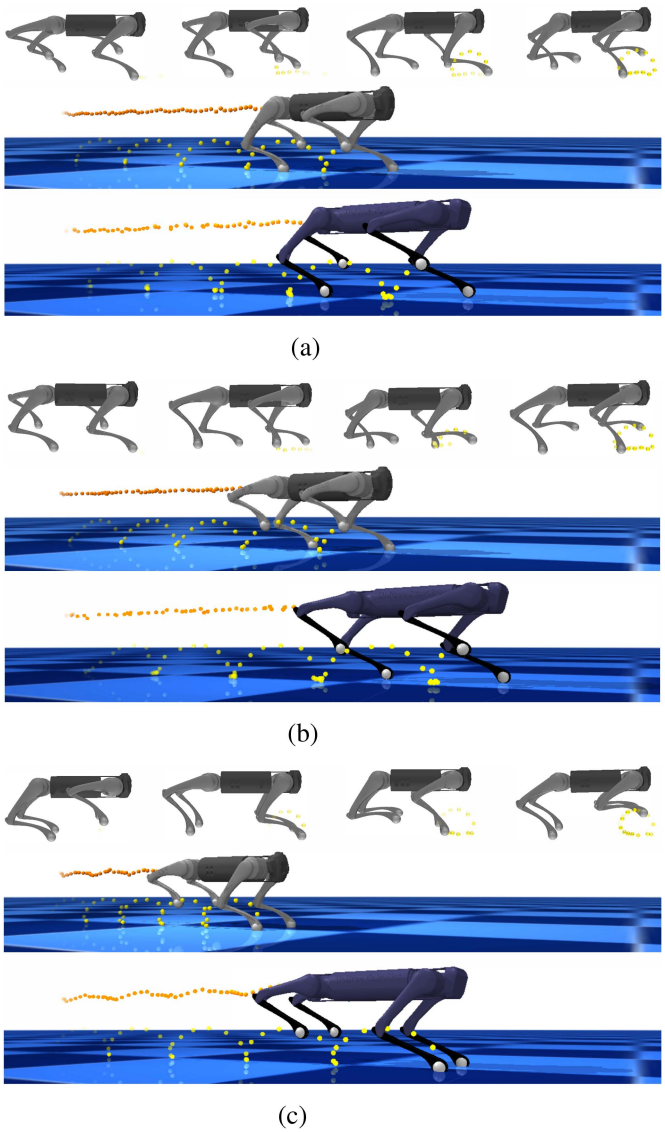


Fig. 7. Differences between original motion and retargeted motion are shown for: (a) Trot ($1.0\mathrm{m/s}$), (b) Pace ($1.0\mathrm{m/s}$), and (c) Bound ($1.0\mathrm{m/s}$). The baseless motion of the small robot (Go1, Above) can reconstructed as whole-body motion (Go1, Middle). Similarly, it can be retargeted to the large robot (AlienGo, Below) without base motion. The figure also shows that SMR can generate naturally elongated base trajectories for robots of different sizes. The orange and yellow dots indicate base and foot trajectory, respectively. (a) Trot. (b) Pace. (c) Bound.

SideSteps on three robots, as illustrated in Fig. 9, overcoming the difference in kinematic and dynamic properties.[3]

To bridge the sim-to-real gap, we applied domain randomization [48] and introduced additional observations for the control policy. Specifically, we randomized controller gains, mass, inertia, friction, and floor restitution, and we also applied random velocity impulses to push the robot. The details of the randomization are provided in Table VI. In addition, we use a base state estimator [49] to provide the linear velocity, angular velocity, and height of the base as additional observations to the policies. Since we have the contact schedule of the target

[3]The footage from the real-world experiments can be found in the attached video.

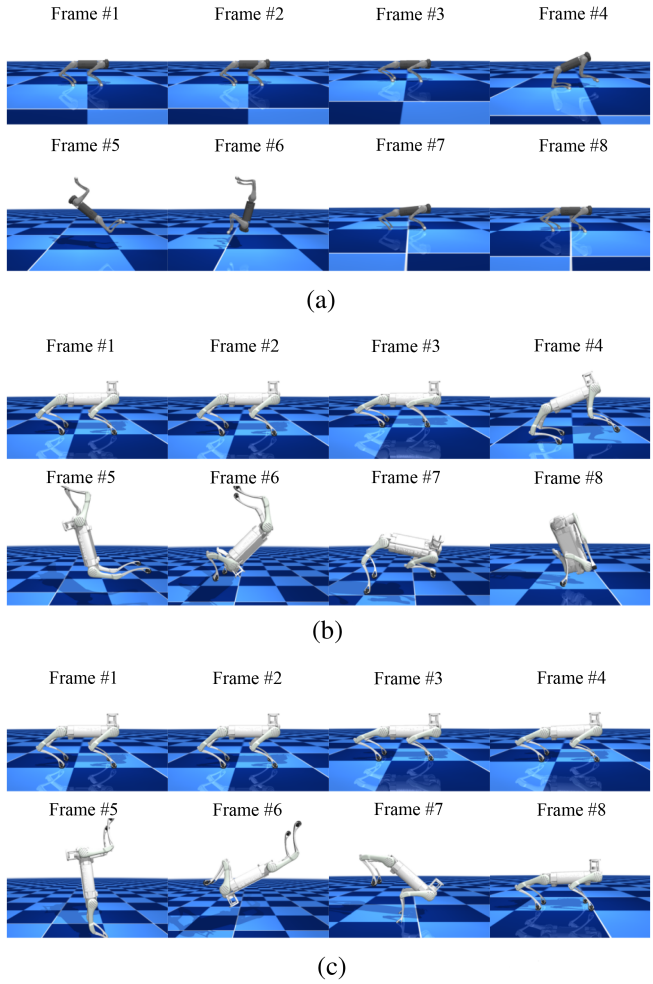


Fig. 8. We experiment with BackFlip to highlight the importance of temporal optimization in dynamic motions with a flight phase. (a) The original motion of Go1 is transferred to B2 (b) without temporal optimization and (c) with temporal optimization. (a) Go1 BackFlip. (b) B2 BackFlip. (c) B2 BackFlip with temporal optimization

TABLE VI
SAMPLING RANGES FOR DOMAIN RANDOMIZATION

Values	Minimum	Maximum
Friction	0.75	1.00
Mass(kg)	-1.0	1.0
Proportional Gain Multiplier	0.9	1.1
Damping Gain Multiplier	0.9	1.1
Restitution	0.0	0.5
COM displacement (mm)	-100	100

motion in hand, we also utilize it when estimating the base state. Introducing velocity observations helps in reducing the motion drifting over time and enables the production of more accurate jumping motions in HopTurn.

A. Analysis on Motion Tracking

To examine the motion deployed in the real world, we used motion capture devices (Opti-Track Prime $\times 22$) to record the base position $\mathbf{p}_b$ and orientation $\mathbf{h}$ of the robot. In addition, we recorded the joint angle values θ using the motor encoders, thereby collecting the entire generalized coordinate

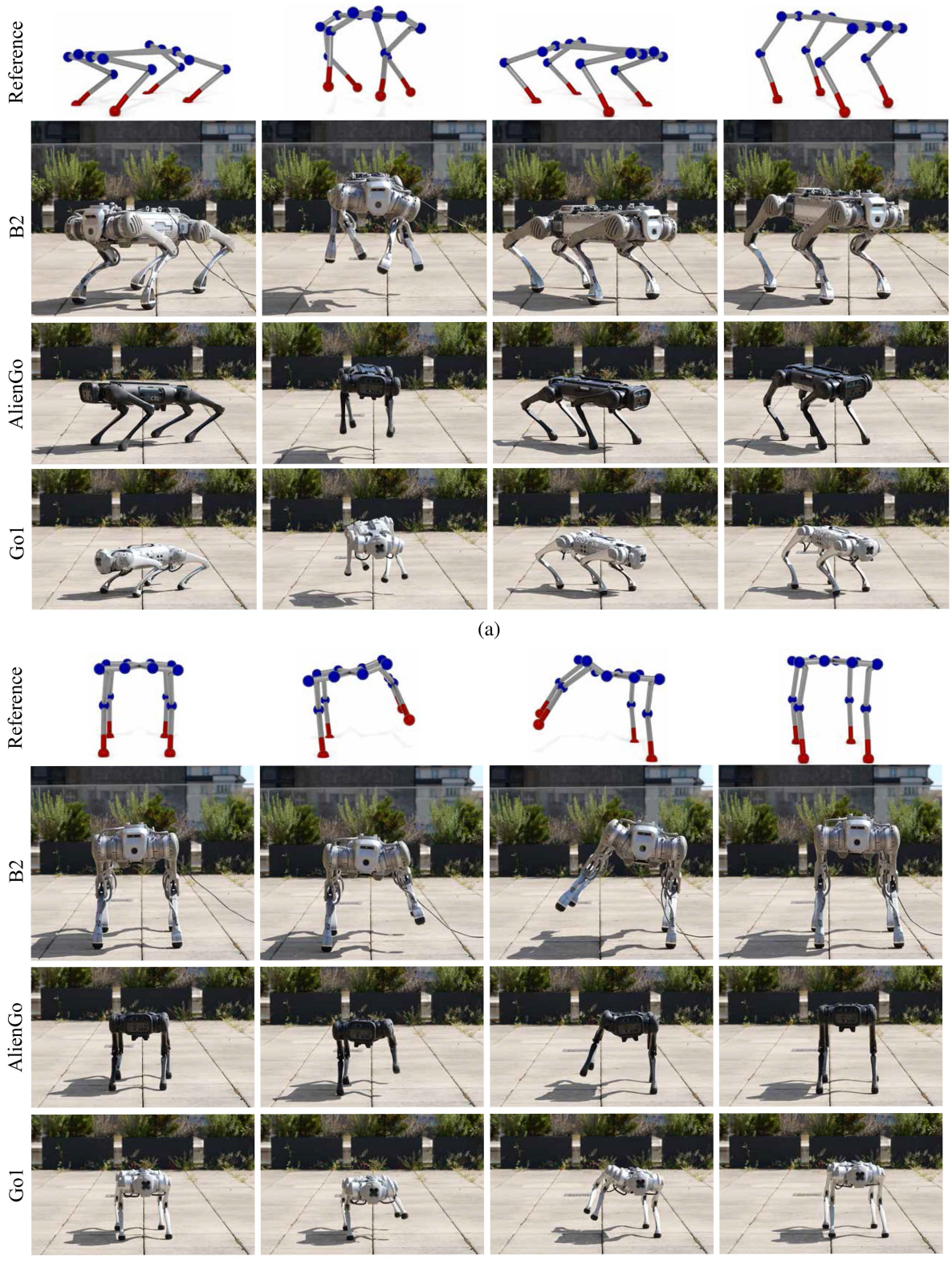


Fig. 9. Real-world deployment of control policy for two motions. (a) HopTurn and (b) SideSteps. From top to bottom, the motions for the reference, Go1, AlienGo, and B2 are illustrated.

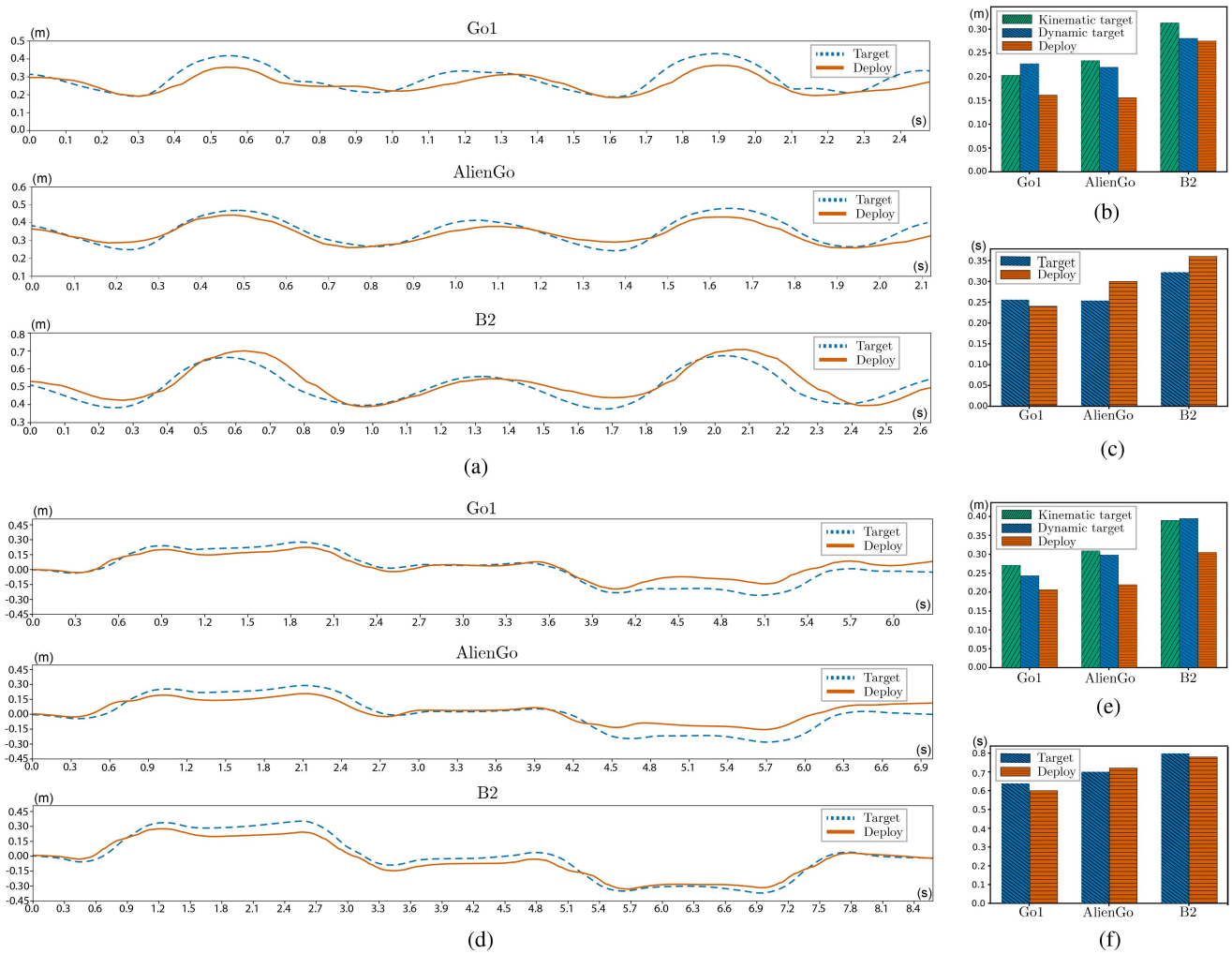


Fig. 10. Motion tracking results for HopTurn and SideSteps are illustrated to highlight the deformation of target motion by STMR in both spatial and temporal dimensions, as well as the tracking performance. For HopTurn, (a) the vertical base trajectory, (b) the height of the first jump, and (c) the elapsed time of the first jump are illustrated. For SideSteps, (d) the horizontal base trajectory, (e) the horizontal distance of the first step, and (f) the elapsed time of the first step are shown. The kinematic and dynamic targets, illustrated in (b) and (e), represent the motions refined by SMR and STMR, while Deploy indicates the motion recorded in the real world using motion capture devices. (a) Vertical base trajectory of HopTurn. (b) Jump height. (c) Jump time. (d) Horizontal base trajectory of SideSteps. (e) Stepstep distance. (f) Stepstep time.

$\mathbf{q} = [\mathbf{p}_b, \mathbf{h}, \theta]$ of the robot. Using this data, we analyzed the tracking performance and motion deformation in both spatial and temporal dimensions.

For HopTurn motion, the mean error per frame and keypoints was 30.5, 41.2, and 42.2 mm for Go1, AlienGo, and B2, respectively. For the case with SideSteps, the mean error was 28.0, 35.4, and 34.2 mm. On average, the error across both motions was 35.3 mm, demonstrating that the closed-loop control policies accurately produce the motions on the robots, effectively overcoming uncertainties and sim-to-real gaps.

Shifting our attention to retargeted motions, we analyze how our method appropriately adjusts the motion in both the space and time domains for each robot. Fig. 10(a) illustrates the tracking result of the base height trajectory for HopTurn with the three robots. From the line plots, we can observe that each robot's motion execution time changed, with AlienGo being 13.95% shorter than Go1 and B2 being 5.81% longer than Go1. We also measured the time and height of the first jump, which is defined as the interval between the minima and maxima of the base height trajectory. As shown in Fig. 10(b), SMR retargets the source motion appropriately at the kinematic level so that the jump height increases with the robot's scale.

Furthermore, it is evident that STMR tailors the HopTurn motion for each robot based on its physical properties, such as weight and actuation power. Notably, the maximum mechanical power per mass for AlienGo is 368.1 W/kg, which is 48.4% smaller than that of Go1. This results in AlienGo jumping lower than Go1 by 2.95% in simulation and 3.77% in the real world, despite its larger dimensions. We note that AlienGo nearly reached its maximum mechanical power during the execution of this motion. The maximum mechanical power per mass for B2 is 9.6% higher than that of Go1. Consequently, it easily managed to jump 23.80% higher than Go1. Meanwhile, we observe the jump time increases with the robot's scale, as observed in Fig. 10(c),

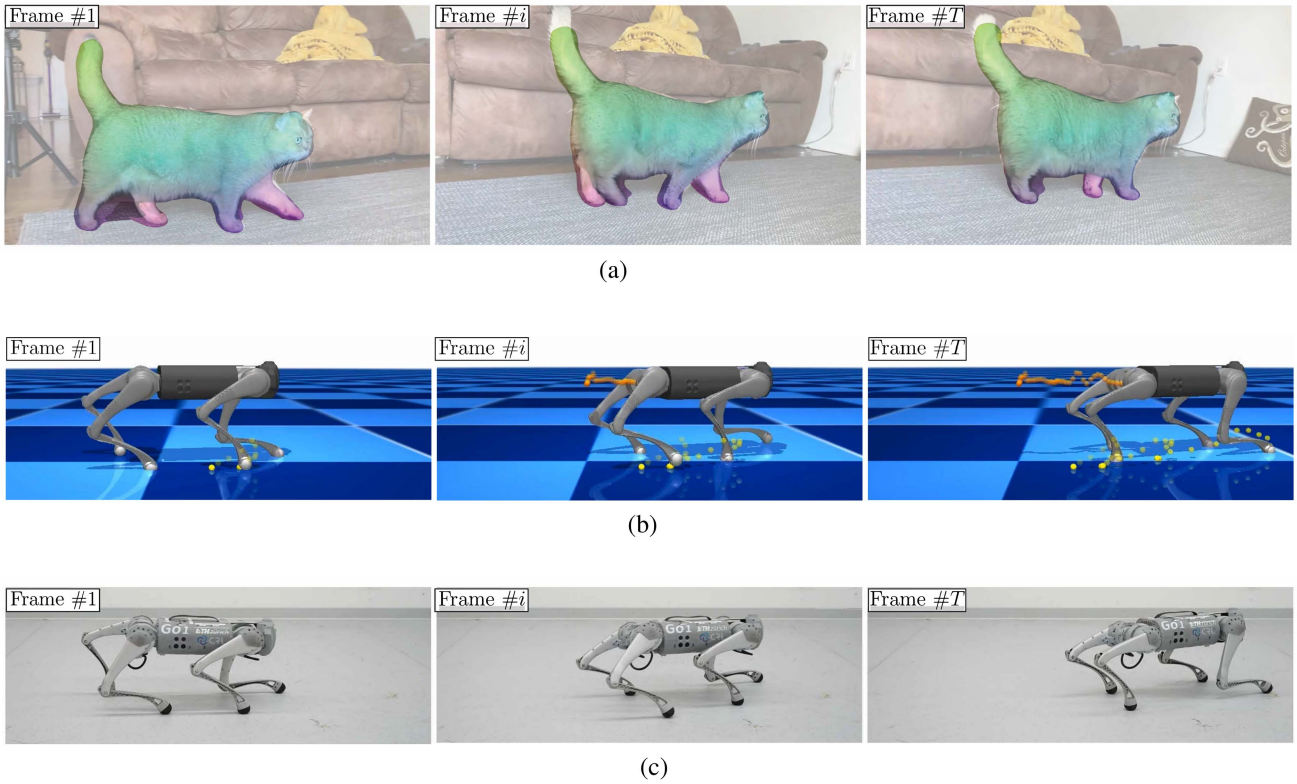


Fig. 11. Keypoint trajectories are (a) extracted using pose-estimator [50] and (b) reconstructed as a whole-body motion. Subsequently, the motion is utilized as a reference motion to train control policy and (c) deploy in the real world. (a) Estimated pose from raw video. (b) Reconstructed whole-body motion. (c) Real-world deployment.

with AlienGo jumping 10% longer and B2 jumping 19.12% longer than Go1.

For SideSteps, we focus on horizontal movement, as illustrated in Fig. 10(d). Similar to HopTurn, each robot's total elapsed time changes where the time duration of AlienGo and B2 is extended by 10.96% and 35.16% compared to that of Go1. Likewise, we compare the time and traveled distance of the first step for each robot. As shown in Fig. 10(e), the sidestep distance increases according to the robots' scale for all kinematic targets, dynamic targets, and real-world deployments. To be specific, the step distance for AlienGo and B2 is larger by 14.70% and 41.50% than Go1. Similarly, sidestep time also increased by 2.82% and 8.45% for AlienGo and B2, as illustrated in Fig. 10(f). Again, we highlight that such deformation is caused by STMR optimizing motion in both space and time domains, resulting in successful deployment.

B. Motion Retargeting From Videos

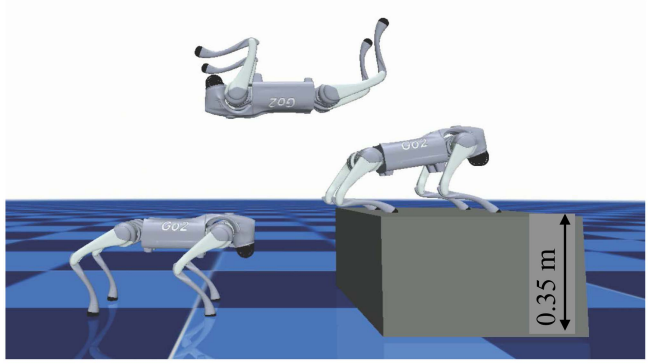
Leveraging the reconstruction capability of the proposed method, we demonstrate motion retargeting with keypoint trajectories obtained from the video pose estimator [50]. Although we utilize the pose estimator that provides the base position, the estimation is considerably noisy because the global pose of the camera is unknown. Therefore, we removed and reconstructed the base position, as shown in Fig. 11(a) . Subsequently, we temporally optimized the kinematic motion to consider the dynamic properties of the robot. Finally, we trained residual policy as in Section VIII-C, where the resulting motion is illustrated in Fig. 11(a).

Our proposed method requires contact Booleans for each foot to reconstruct whole-body motion. Therefore, we calculate foot velocity by applying finite differencing to the foot position provided by the pose estimator, followed by thresholding the foot velocity. However, these values are noisy due to the inaccuracy of the pose estimator. Particularly, high-frequency noise in the obtained contact leads to jerky kinematic motions. To overcome this, we apply a low-pass filter to the foot position, making the reconstructed motion smoother.

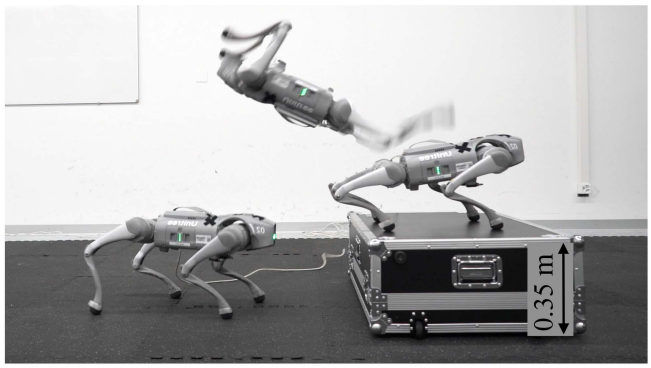
Compared to the HopTurn and SideSteps, the motions obtained from videos involve more diverse phases, such as walking slowly followed by sharp turning. To better handle these varying motion phases, we set the number of time segments to $S = 3$. In more detail, the tracking errors for time segments $S = 1$, $S = 2$, and $S = 3$ were 410.8, 405.5, and 378.8 mm, respectively, in simulation.

C. Terrain-Aware Retargeting of Dynamic Flight-Phase Motions

We deploy the BackFlip motion on a real robot to demonstrate the effectiveness of our framework in refining motion both spatially and temporally. Specifically, we highlight two key capabilities: 1) spatial retargeting adapts the base trajectory to the terrain; and 2) temporal retargeting adjusts timing to ensure dynamic feasibility for real-world deployment.



(a)



(b)

Fig. 12. (a) BackFlip is retargeted to Go2 by adapting to the terrain and ensuring the feasibility of imitation. (b) The trained policy successfully performs the BackFlip in the real world. (a) Terrain-aware BackFlip. (b) BackFlip deployed in the real world.

Starting from the flat-ground BackFlip motion introduced in Section VIII-F, we retarget it to the Unitree Go2 robot jumping off a box with a height of 0.35 m. To achieve this, we first generate a kinematically feasible motion for the new terrain using SMR, based on the contact schedule from the original motion. Since SMR adapts the base trajectory to the terrain, it produces a kinematic motion that enables jumping from the box. We then apply temporal retargeting to refine the timing and satisfy dynamic constraints. As shown in Fig. 12(a), the resulting motion successfully performs a BackFlip in simulation.

Similar to previous experiments, we train the control policy to handle real-world uncertainties. A key difference, however, is that we do not provide height and linear velocity information from the state estimator, as it assumes flat terrain. The policy is successfully deployed in the real world and performs BackFlip, as shown in Fig. 12(b).

X. Limitation and Future Work

The proposed method involves generating whole-body motion from baseless motion coherent to the contact sequence in a source motion. This process heavily relies on accurate contact estimation, whereas wrong estimation can cause irregular movements that propagate during the construction of whole-body motions. In particular, the estimated contact phase can be noisy as we obtain the contact Boolean by thresholding the foot's velocity. This can lead to jerky motions or failure of imitation, as robots cannot follow such a quick contact transition. In our experiments, we partially address this issue by applying a low-pass filter that regularizes high-frequency change of contact phases. In future work, we plan to explore more advanced contact estimation methods [51] to obtain a more accurate contact sequence from the motion.

We utilized BO to find the optimal temporal parameters for the TMR problem, formulated as a nested optimization problem. It is worth noting that optimizing over the temporal dimension is a very challenging problem, and BO offers a reasonable solution to this. However, BO has limitations when scaling to high-dimensional space [52]. Thus, it can be problematic if the source motions are long and require a larger number of time segments. In our experiments, the duration of the motions is relatively short (< 15 s); therefore, we set the number of time segments to a maximum of three to mitigate this issue. Although we can mitigate this issue by dividing motion into smaller clips, exploring more scalable optimization techniques for the TMR problem remains an area for future research.

XI. Conclusion

This article introduces the problem of STMR that aims to generate kino-dynamically feasible motion to guide the IL process. The STMR ensures that the transferred motions are kino-dynamically feasible for the target system. Furthermore, it facilitates the use of motion data with the unknown origin of reference by generating whole-body motions that closely mimic the agility and expressiveness of natural animal movements.

Our comprehensive simulation experiments demonstrate STMR's efficacy in control policy learning, which facilitates dynamic motion imitation with more accurate motion tracking performance compared to baseline methods. Furthermore, our method effectively preserves the contact schedule of the source motion while eliminating foot slips. We successfully deployed the retargeted motions on four different robotic hardware platforms with varying dimensions and physical properties, highlighting the practical applicability of STMR in generating natural and dynamic motions for general quadruped robots.

Although we complemented our STMR framework with a control policy trained via RL in this work, we note that any feedback motion controller for quadrupedal robots capable of producing whole-body movement can be used to execute the retargeted motion. We look forward to seeing potential extensions of this work by integrating our STMR framework with various model-based or learning-based control methods and applying it to more challenging locomotion tasks, such as bipedal locomotion or whole-body loco-manipulation tasks for legged robots.

Moreover, STMR demonstrated the ability to retarget motions from noisy sources, such as raw video data. This suggests that web-scale motion datasets with physical guarantees for imitation could be realized. Such an approach may serve as an efficient data collection pipeline, enabling scalable and adaptable motion transfer across diverse robotic platforms. We leave the exploration of this direction as future work.

REFERENCES

[1] D. Kang, F. De Vincenti, N. C. Adami, and S. Coros, "Animal motions on legged robots using nonlinear model predictive control," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Kyoto, Japan, Oct. 2022, pp. 11955–11962.

[2] N. Rudin, H. Kolvenbach, V. Tsounis, and M. Hutter, "Cat-like jumping and landing of legged robots in low gravity using deep reinforcement learning," *IEEE Trans. Robot.*, vol. 38, no. 1, pp. 317–328, Feb. 2022.

[3] A. Serifi, R. Grandia, E. Knoop, M. Gross, and M. Bächer, "VMP: Versatile motion priors for robustly tracking motion on physical characters," in *Proc. ACM SIGGRAPH/Eurographics Symp. Comput. Animation*, 2024, Art. no. e15175.

[4] G. Feng et al., "GenLoco: Generalized locomotion controllers for quadrupedal robots," in *Proc. Conf. Robot Learn.*, 2023, pp. 1893–1903.

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

[6] A. Tagliabue and J. P. How, "Efficient deep learning of robust policies from MPC using imitation and tube-guided data augmentation," *IEEE Trans. Robot.*, vol. 40, pp. 4301–4321, 2024.

[7] W.-S. Yang, W.-C. Lu, and P.-C. Lin, "Legged robot running using a physics-data hybrid motion template," *IEEE Trans. Robot.*, vol. 37, no. 5, pp. 1680–1695, Oct. 2021.

[8] R. Grandia, F. Farshidian, E. Knoop, C. Schumacher, M. Hutter, and M. Bächer, "DOC: Differentiable optimal control for retargeting motions onto legged robots," *ACM Trans. Graph.*, vol. 42, no. 4, pp. 1–14, Aug. 2023.

[9] J. Z. Zhang et al., "SLoMo: A general system for legged robot motion imitation from casual videos," *IEEE Robot. Automat. Lett.*, vol. 8, no. 11, pp. 7154–7161, Nov. 2023.

[10] D. Kang, S. Zimmermann, and S. Coros, "Animal gaits on quadrupedal robots using motion matching and model-based control," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2021, pp. 8500–8507.

[11] X. B. Peng, P. Abbeel, S. Levine, and M. Van De Panne, "DeepMimic: Example-guided deep reinforcement learning of physics-based character skills," *ACM Trans. Graph.*, vol. 37, no. 4, pp. 1–14, Aug. 2018.

[12] X. B. Peng, E. Coumans, T. Zhang, T.-W. E. Lee, J. Tan, and S. Levine, "Learning agile robotic locomotion skills by imitating animals," in *Proc. Robot.: Sci. Syst.*, 2020, pp. 1–12.

[13] X. B. Peng, Z. Ma, P. Abbeel, S. Levine, and A. Kanazawa, "AMP: Adversarial motion priors for stylized physics-based character control," *ACM Trans. Graph.*, vol. 40, no. 4, pp. 1–20, Aug. 2021.

[14] A. Escontrela et al., "Adversarial motion priors make good substitutes for complex reward functions," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Kyoto, Japan, IEEE, Oct. 2022, pp. 25–32.

[15] C. Li, M. Vlastelica, S. Blaes, J. Frey, F. Grimminger, and G. Martius, "Learning agile skills via adversarial imitation of rough partial demonstrations," in *Proc. Conf. Robot Learn.*, 2023, pp. 342–352.

[16] S. Levine and V. Koltun, "Guided policy search," in *Proc. Int. Conf. Mach. Learn.*, 2013, pp. 1–9.

[17] Y. Fuchioka, Z. Xie, and M. Van De Panne, "OPT-Mimic: Imitation of optimized trajectories for dynamic quadruped behaviors," in *Proc. IEEE Int. Conf. Robot. Automat.*, May 2023, pp. 5092–5098.

[18] D. Kang, J. Cheng, M. Zamora, F. Zargarbashi, and S. Coros, "RL+ model-based control: Using on-demand optimal control to learn versatile legged locomotion," *IEEE Robot. Automat. Lett.*, vol. 8, no. 10, pp. 6619–6626, Oct. 2023.

[19] F. Liu et al., "Opt2Skill: Imitating dynamically-feasible whole-body trajectories for versatile humanoid loco-manipulation," 2024, *arXiv:2409.20514*.

[20] K. Yamane, Y. Ariki, and J. Hodgins, "Animating non-humanoid characters with human motion data," in *Proc. ACM SIGGRAPH/Eurographics Symp. Comput. Animation*, 2010, pp. 169–178.

[21] Y. Seol, C. O'Sullivan, and J. Lee, "Creature features: Online motion puppetry for non-human characters," in *Proc. 12th ACM SIGGRAPH/Eurographics Symp. Comput. Animation*, Jul. 2013, pp. 213–221.

[22] S. Kim, M. Sorokin, J. Lee, and S. Ha, "HumanConQuad: Human motion control of quadrupedal robots using deep reinforcement learning," in *Proc. SIGGRAPH Asia Emerg. Technol.*, Dec. 2022, pp. 1–2.

[23] S. Choi, M. Pan, and J. Kim, "Nonparametric motion retargeting for humanoid robots on shared latent space," in *Proc. 16th Robot.: Sci. Syst.*, 2020, pp. 1–9.

[24] K.-J. Choi and H.-S. Ko, "On-line motion retargetting," in *Proc. 7th Pacific Conf. Comput. Graph. Appl.*, 1999, pp. 32–42.

[25] S. Choi and J. Kim, "Towards a natural motion generator: A pipeline to control a humanoid based on motion data," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 4373–4380.

[26] R. Villegas, J. Yang, D. Ceylan, and H. Lee, "Neural kinematic networks for unsupervised motion retargetting," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8639–8648.

[27] T. Li, J. Won, A. Clegg, J. Kim, A. Rai, and S. Ha, "ACE: Adversarial correspondence embedding for cross morphology motion retargeting from human to nonhuman characters," in *Proc. SIGGRAPH Asia Conf. Papers*, New York, NY, USA, 2023, pp. 1–11.

[28] K. Aberman, P. Li, D. Lischinski, O. Sorkine-Hornung, D. Cohen-Or, and B. Chen, "Skeleton-aware networks for deep motion retargeting," *ACM Trans. Graph.*, vol. 39, no. 4, Aug. 2020, Art. no. 62.

[29] S. Choi, M. J. Song, H. Ahn, and J. Kim, "Self-supervised motion retargeting with safety guarantee," in *Proc. IEEE Int. Conf. Robot. Automat.*, May 2021, pp. 8097–8103.

[30] S. Tak and H.-S. Ko, "A physically-based motion retargeting filter," *ACM Trans. Graph.*, vol. 24, no. 1, pp. 98–117, Jan. 2005.

[31] M. Al Borno, L. Righetti, M. J. Black, S. L. Delp, E. Fiume, and J. Romero, "Robust physics-based motion retargeting with realistic body shapes," *Comput. Graph. Forum*, vol. 37, no. 8, pp. 81–92, Dec. 2018.

[32] Q. Rouxel, K. Yuan, R. Wen, and Z. Li, "Multicontact motion retargeting using whole-body optimization of full kinematics and sequential force equilibrium," *IEEE/ASME Trans. Mechatron.*, vol. 27, no. 5, pp. 4188–4198, Oct. 2022.

[33] D. Q. Mayne, "Differential dynamic programming—A unified approach to the optimization of dynamic systems," in *Control and Dynamic Systems*, vol. 10. Amsterdam, The Netherlands: Elsevier, 1973, pp. 179–254.

[34] Y. Tassa, T. Erez, and E. Todorov, "Synthesis and stabilization of complex behaviors through online trajectory optimization," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2012, pp. 4906–4913.

[35] E. Todorov, T. Erez, and Y. Tassa, "MuJoCo: A physics engine for model-based control," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst*, 2012, pp. 5026–5033.

[36] T. Howell, N. Gileadi, S. Tunyasuvunakool, K. Zakka, T. Erez, and Y. Tassa, "Predictive sampling: Real-time behaviour synthesis with MuJoCo," Dec. 2022, *arXiv:2212.00541*.

[37] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, 2011.

[38] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, "OSQP: An operator splitting solver for quadratic programs," *Math. Program. Computation*, vol. 12, no. 4, pp. 637–672, 2020.

[39] J. Snoek, H. Larochelle, and R. P. Adams, "Practical Bayesian optimization of machine learning algorithms," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2012, pp. 2951–2959.

[40] T. Johannink et al., "Residual reinforcement learning for robot control," in *Proc. Int. Conf. Robot. Automat.*, 2019, pp. 6023–6029.

[41] D. Kinga et al., "A method for stochastic optimization," in *Proc. Int. Conf. Learn. Representations*, 2015, vol. 5, no. 6.

[42] V. Makoviychuk et al., "Isaac gym: High performance gpu-based physics simulation for robot learning," in *Proc. Neural Inf. Process. Syst. (NeurIPS) Track Datasets Benchmarks*, 2021.

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

[44] Y. Ding, A. Pandala, and H.-W. Park, "Real-time model predictive control for versatile dynamic motions in quadrupedal robots," in *Proc. Int. Conf. Robot. Automat.*, 2019, pp. 8484–8490.

[45] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Math. Program.*, vol. 106, pp. 25–57, 2006.

[46] P. Senin, "Dynamic time warping algorithm review," *Information and Computer Science Department University of Hawaii Manoa: Honolulu, HI, USA*, 2008, vol. 855, pp. 1–23.

[47] H. Zhang, S. Starke, T. Komura, and J. Saito, "Mode-adaptive neural networks for quadruped motion control," *ACM Trans. Graph.*, vol. 37, no. 4, pp. 1–11, Jul. 2018.

[48] W. Zhao, J. P. Queralta, and T. Westerlund, "Sim-to-real transfer in deep reinforcement learning for robotics: A survey," in *Proc. IEEE Symp. Ser. Comput. Intell.*, 2020, pp. 737–744.

[49] M. Bloesch et al., "State estimation for legged robots - consistent fusion of leg kinematics and IMU," in *Proc. Robot.: Sci. Syst.*, 2012, pp. 17–24.

[50] G. Yang et al., "BANMo: Building animatable 3 D neural models from many casual videos," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 2853–2863.

[51] S. Zhang et al., "RoHM: Robust human motion reconstruction via diffusion," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2024, pp. 14606–14617.

[52] R. Moriconi, M. P. Deisenroth, and K. Sesh Kumar, "High-dimensional Bayesian optimization using low-dimensional feature spaces," *Mach. Learn.*, vol. 109, pp. 1925–1943, 2020.

Taerim Yoon (Graduate Student Member, IEEE) received the B.S. degree in mechanical engineering from Korea University, Seoul, South Korea, in 2022, where he is currently working toward the integrated M.S./Ph.D. degree in artificial intelligence.

His research interests include legged robotics, companion robots, reinforcement learning, and optimal control.

Dongho Kang (Graduate Student Member, IEEE) received the B.S. degree, double-majoring in mechanical and aerospace engineering, and computer science and engineering, from Seoul National University, Seoul, South Korea, in 2016, and the M.S. degree in mechanical engineering, in 2019, from ETH Zurich, Zurich, Switzerland, where he is currently working toward the Ph.D. degree in computer science.

His research interests include legged robotics, character animation, optimal control, and reinforcement learning.

Seungmin Kim (Graduate Student Member, IEEE) received the B.S. degree in electrical engineering, in 2014, from Korea University, Seoul, South Korea, where he is currently working toward the Ph.D. degree in artificial intelligence.

His research interests include trajectory optimization, task and motion planning, and dexterous manipulation.

Jin Cheng (Graduate Student Member, IEEE) received the B.Eng. degree in vehicle engineering from Tsinghua University, Beijing, China, in 2020, and the M.Sc. degree in mechanical engineering, in 2023, from ETH Zurich, Zurich, Switzerland, where he is currently working toward the Ph.D. degree in computer science.

His research interests include legged robots, model-based control, reinforcement learning, locomotion, and loco-manipulation.

Min Sung Ahn received the Ph.D. degree in mechanical engineering from the University of California, Los Angeles, Los Angeles, CA, USA, in 2023.

His research interests include humanoid robots, model-based control, and reinforcement learning.

Stelian Coros (Member, IEEE) received the Ph.D. degree in computer science from the University of British Columbia, Vancouver, BC, Canada, in 2011.

He leads the Computational Robotics Lab, a research group within the Institute for Intelligent Interactive Systems, ETH Zurich, Zurich, Switzerland. His research group draws insights from computer science, applied mathematics, and control theory to establish the foundations for algorithms that address a variety of computational problems in robotics. Applications of the work range from studying the principles of dexterous manipulation and legged locomotion to computation-driven design of novel types of robots. His research interests include computer graphics, robotics, and computational fabrication.

Sungjoon Choi (Member, IEEE) received the B.S. and Ph.D. degrees in electrical and computer engineering from Seoul National University, Seoul, Korea, in 2012 and 2018, respectively.

In 2018, he was a Research Scientist with Kakao Brain, Pangyo, South Korea. From 2018 to 2020, he was a Postdoctoral Associate with Disney Research, Los Angeles, CA, USA. He is currently an Assistant Professor with the Department of Artificial Intelligence, Korea University, Seoul, South Korea. His research interests include human–robot interaction and dexterous robot manipuation.