

Realizing Text-Driven Motion Generation on NAO Robot: A Reinforcement Learning-Optimized Control Pipeline

Zihan Xu[†], Mengxian Hu[†], Kaiyan Xiao, Qin Fang, Chengju Liu^{*} and Qijun Chen, *Senior Member, IEEE*

Abstract—Human motion retargeting for humanoid robots, transferring human motion data to robots for imitation, presents significant challenges but offers considerable potential for real-world applications. Traditionally, this process relies on human demonstrations captured through pose estimation or motion capture systems. In this paper, we explore a text-driven approach to mapping human motion to humanoids. To address the inherent discrepancies between the generated motion representations and the kinematic constraints of humanoid robots, we propose an angle signal network based on norm-position and rotation loss (NPR Loss). It generates joint angles, which serve as inputs to a reinforcement learning-based whole-body joint motion control policy. The policy ensures tracking of the generated motions while maintaining the robot’s stability during execution. Our experimental results demonstrate the efficacy of this approach, successfully transferring text-driven human motion to a real humanoid robot NAO.

I. INTRODUCTION

Humanoid robots have long been recognized for their potential to mimic human actions due to their anthropomorphic structure. While their design naturally lends itself to imitation, achieving the seamless transfer of human motion to robots remains a complex challenge [1]. Recent advancements in text-driven diffusion models offer a promising solution by generating human motion from textual descriptions [2]–[4], paving the way for more flexible and intuitive motion generation. Applying these human motions to humanoid robots requires overcoming the structural and kinematic discrepancies between humans and humanoids.

Directly applying human motion data to robotic control is far from straightforward while matching the end-effector pose with the target’s pose yields higher similarity and practical applicability. From this perspective, end-effector tracking is not only more effective but also more aligned with real-world applications. To achieve reliable end-effector tracking, human motion data must first be processed in a way that is compatible with the robot’s kinematics. Traditional human pose estimation methods typically infer joint orientations from captured human poses, which are then used to compute robot joint angles via inverse kinematics [5]. Alternatively, using parametric human models to parameterize the robot’s structure provides more consistent motion transfer, reformulating the imitation task as a keypoint tracking problem [6].

From the robot’s perspective, human motion provides a high-dimensional solution space $S \subset \mathbb{R}^n$. The goal of

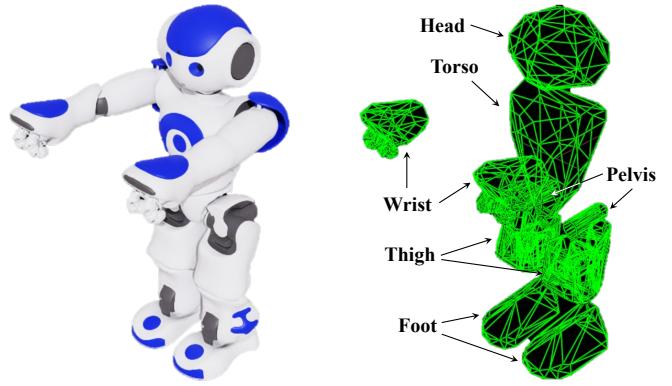


Fig. 1. Visualization of NAO robot: Appearance (left) and collision model (right). Except for the thigh link, which uses the convex decomposition method to generate the colliders, all other links adopt the convex hull method, generating colliders with up to 64 faces.

retargeting human motion to a humanoid is to identify a subset $S' \subseteq S$ that satisfies a set of predefined constraints or conditions. Fundamentally, we aim to encode human movements into a mathematical representation that captures essential parameters: the relative position and orientation of the end-effector with respect to the root joint of the articulated chain. These relative parameters are dimensionless, ensuring that they can generalize across both human and humanoid motions. This parameterization allows us to decode the posture regardless of whether the source is human or humanoid motion.

Instead of directly transferring human motion to humanoid robots, we employ a norm-based description of motion to define as the basis for our normalized mapping loss function, as discussed in Section III-B. This description encompasses both norm-position and rotational components. For human movements, we compute the normalized parameters. For humanoid robots, the objective is to determine the joint configurations that best replicate the posture under the described parameters. This approach not only enhances the fidelity of imitation but also ensures that the robot’s motion closely adheres to the intended end-effector trajectory, making the solution adaptable and scalable to different motion tasks.

In this paper, we use text-driven diffusion model to generate reference joint commands through angle signal net and optimized the joint commands to reliable joint output based on the IsaacLab reinforcement learning frameworks as illustrated in Fig. 2. The whole-body control is achieved and applied in a real-world robot. The main contributions include: **(I)** A text-driven diffusion model was leveraged to generate

[†] are equal contributors, ^{*} is the corresponding author.

Authors are with the College of Electronic and Information Engineering, Tongji University, Shanghai, China. {xuzihan, humengxian, xiaokaiyan, tongji_fq, liuchengju, qjchen}@tongji.edu.cn

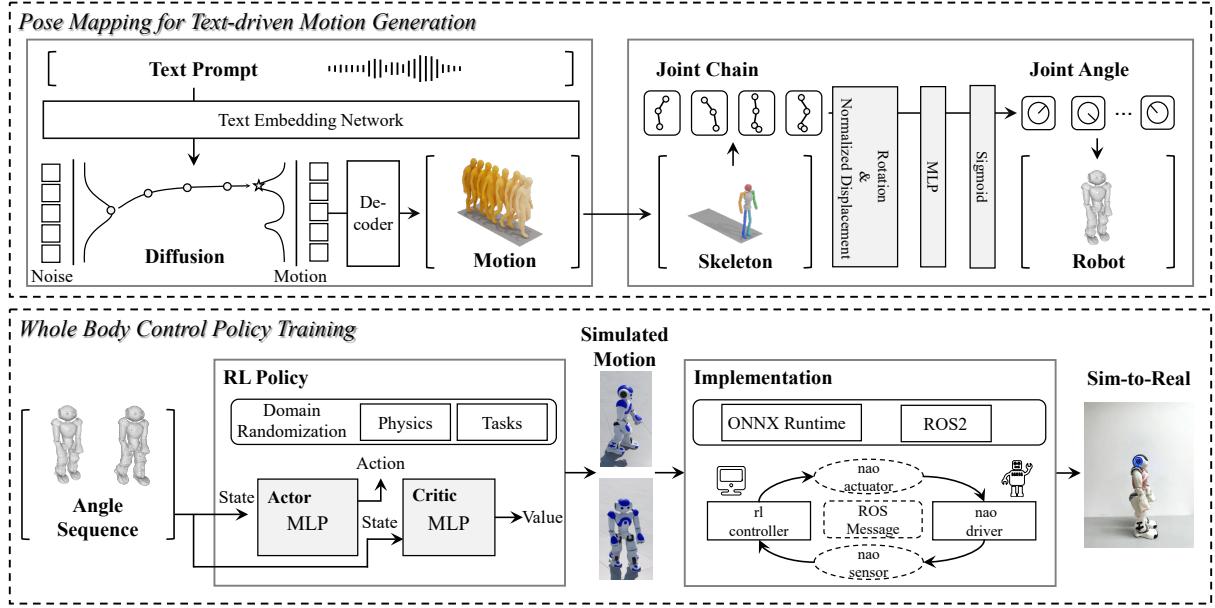


Fig. 2. Overview of mapping text-driven human motion to humanoid robots: (a) **Pose Mapping** (Section III): Mapping the text-driven human motion generation to the robot joint angles by angle signal network. (b) **Whole Body Control** (Section IV): Training the reinforcement-learning based controller and implement the policy using ONNX Runtime and run the system in ROS2 framework.

motion reference sequences, and employed an angle signal network based on NPR Loss to achieve precise pose mapping. **(II)** The detailed NAO robot simulation model, derived from its URDF, was developed, with its collision volumes fine-tuned to ensure full compatibility with the IsaacSim 4.5 platform. Furthermore, we identified practical joint actuator parameters tailored for real-world NAO applications. **(III)** The robot motion control system was designed using the ROS2 framework, enabling the successful deployment of reinforcement learning policies for controlling a physical NAO robot. All components of this work, including the simulation model and control system, are fully open-sourced to facilitate further research and development in the field.

II. RELATED WORKS

Human motion imitation by humanoid robots has been a longstanding research focus, aiming to enable robots to perform tasks in a human-like manner. Previous works often rely on human pose estimation or motion capture data to generate control commands for robots [7]–[9]. Khalil *et al.* [10] employed methods based on inverse kinematics to compute joint angles from estimated human poses and apply to robots. These techniques primarily focus on upper-body motion transfer, neglecting leg movements and limiting the diversity of imitated actions. Directly mapping human joint angles to robot joints is challenging due to differences in kinematic structures and degrees of freedom (DoFs) between humans and robots. To address these challenges, some researchers have parameterized the robot’s structure using human models [6], defining keypoint correspondences to frame the imitation task as a keypoint tracking problem [11], [12]. While this approach provides a structured framework that efficiently aligns human and robot kinematics, it still depends heavily

on human-provided motion data, limiting the robot’s ability to perform novel or unanticipated tasks without additional human input.

Text-driven motion generation has emerged as a promising solution to reduce reliance on human motion data by allowing robots to generate motions based on natural language descriptions. Early discussions focused on variational autoencoders (VAEs) [13]–[15] and generative adversarial networks (GANs) [16], [17]. The former imposes a KL-divergence constraint on the distribution of latent representations and samples from a standard normal distribution, but often yields suboptimal results. The latter avoids explicit modeling of the target distribution through adversarial training, achieving good generative performance, though adversarial training is unstable and prone to mode collapse. Recent advancements include motion diffusion models [4], [18], [19], which convert data into a standard normal distribution by progressively adding noise, and use multiple reverse diffusion steps to recover motion representations during inference. Compared to traditional single-step inference methods, the diffusion framework exhibits higher controllability and generative capability. Early related works, such as Motiondiffuse [4] and MDM [20], demonstrated the feasibility of the motion diffusion framework. Subsequently, MLD [18] and M2DM [21] extended the diffusion process to the motion latent space, improving inference speed. The latest progress, MLCT [22], precomputes diffusion trajectories during the training phase, achieving high-quality, controllable motion generation with lower training and inference costs. These developments provide convenient data tools for imitation learning in humanoid robotics.

Building upon these developments, the subsequent challenge lies in ensuring that humanoid robots can execute the

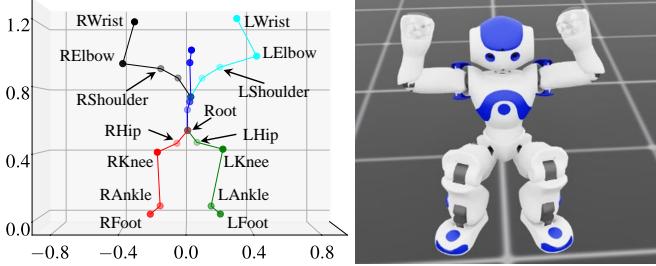


Fig. 3. Skeletal keypoints derived from human motion data and mapped for joints for NAO.

generated motions with stability in real-world environments. Reinforcement learning (RL) provides a framework for robots to learn complex motor skills through trial and error, control policies based on feedback from the environment [23] and enables robots to adapt to dynamic environments [24]. RL-based approaches have demonstrated remarkable success in bipedal locomotion, enabling robots to learn efficient walking and running strategies [25], recover from external disturbances [26], and generalize across different terrains and environments [27]. A crucial aspect of deploying RL-trained policies in real-world settings is the sim-to-real transfer, which addresses the discrepancies between simulation and physical execution. Various techniques, such as domain randomization and adaptation strategies, have been proposed to bridge this gap [28], [29]. These approaches ensure that policies learned in simulation can generalize effectively to unstructured and unpredictable real-world scenarios. Benefiting from the development of large-scale high-performance simulation frameworks [30], we can leverage GPU acceleration to enable massively parallelized training of robot policies. By integrating RL with text-driven motion generation, robots can learn to execute a wide range of motions specified by natural language, adapting them to their unique kinematic structures and environmental constraints.

III. POSE MAPPING FOR TEXT-GENERATED MOTION

A. Synthesizing Text-driven Human Motion

We focus on text-driven humanoid robot imitation strategies to assess the potential of imitation learning in human-robot interaction scenarios. To enable robust motion imitation, diverse and text-driven human motion sequences are the data cornerstone. Relying solely on a single motion capture dataset for imitation is suboptimal due to the high cost of motion capture equipment, which restricts sample diversity, and the semantic ambiguity arising from varying natural language descriptions of the same motion. To address these issues, we introduce the conditional generative model that estimates the human motion distribution and synthesize samples based on fine-grained semantic parsing.

Motion diffusion models are a typical class of generative models with powerful distribution matching properties. It gradually injects noise into the human motion representation through a Gaussian perturbation kernel until the perturbed state approximates the standard normal distribution. Recent

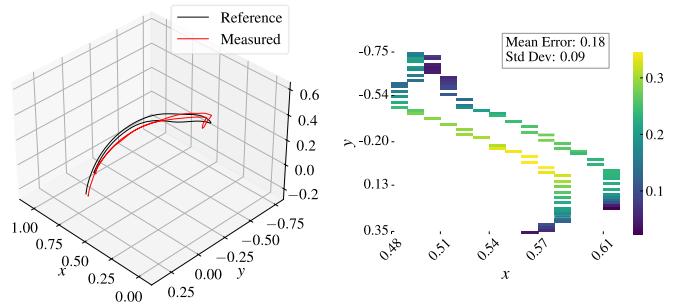


Fig. 4. Comparison of reference and measured spatial trajectories for a waving motion, accompanied by a heatmap visualization of error distribution with mean and standard deviation.

work indicates that the forward diffusion process can be described as the stochastic partial differential equation,

$$dx_t = f(t)x_t dt + g(t)dw_t \quad (1)$$

where $t \in [\epsilon, T]$, ϵ and T are the fixed positive constant, w_t denotes the standard Brownian motion, f and g are the drift and diffusion coefficients respectively. The perturbation parameters are related as follows,

$$f(t) = \frac{d \log \alpha_t}{dt}, \quad g^2(t) = \frac{d \sigma_t^2}{dt} - 2 \frac{d \log \alpha_t}{dt} \sigma_t^2 \quad (2)$$

where α_t and σ_t are noise schedules.

It is further demonstrated that the inverse diffusion process of Eq. (1) is approximated in solving the *probabilistic flow ordinary differential equation*:

$$dx_t = [f(t)x_t - \frac{1}{2}g^2(t)\nabla_{x_t} \log p(x_t)]dt \quad (3)$$

where $\nabla_{x_t} \log p(x_t)$ is named the *score function*, fitted with a neural network \mathcal{F} . The traditional diffusion approach utilizes higher-order ODE numerical solvers to iteratively solve Eq. (3), which involves expensive time and computational costs. To avoid such efficiency limitations, we utilize the motion consistency training framework, which precomputes the diffusion trajectories in the training phase to enable large-scale skip-step sampling in the inference phase.

Specifically, given a textual instruction \mathcal{I} , the CLIP model is employed as the textual feature extractor to parse fine-grained textual semantics. Then, x_T is sampled from the prior distribution $\mathcal{N}(0, I)$ and solved with reference the text embedding features following pre-computed reverse diffusion trajectories from Eq. (3) up to the motion latent representation x_ϵ . It's transformed with the motion decoder \mathcal{D} into the human motion sequence $x = \mathcal{D}(x_\epsilon) \in \mathcal{R}^{f \times 22 \times 3}$ matching the instruction semantics, which contains the 3D coordinates of the 22 joints of f motion frames. The generated human pose data follows the format of the HumanML3D [31] dataset.

B. Normalized Mapping of Humans to Humanoids

Given the structural differences between humans and humanoids, generated human motion sequences often require preprocessing to align with the kinematic characteristics of robots. While fine-tuning the SMPL model [32] for robot

Procedure 1 Computing Rotation Matrix Between Two Vectors

Require: $v_1, v_2 \in \mathbb{R}^3$ (non-zero vectors)

Ensure: 3×3 rotation matrix R

- 1: Normalize input vectors: $\hat{v}_1 = \frac{v_1}{\|v_1\|}$, $\hat{v}_2 = \frac{v_2}{\|v_2\|}$
- 2: Compute rotation axis: $k = \hat{v}_1 \times \hat{v}_2$
- 3: Calculate rotation angle: $\theta = \arccos(\hat{v}_1 \cdot \hat{v}_2)$
- 4: **if** $\|k\| < 10^{-2}$ **then**
- 5: **return** Identity matrix $I_{3 \times 3}$
- 6: **end if**
- 7: Normalize rotation axis: $\hat{k} = \frac{k}{\|k\|}$
- 8: Construct skew-symmetric matrix K :
$$K = \begin{bmatrix} 0 & -\hat{k}_z & \hat{k}_y \\ \hat{k}_z & 0 & -\hat{k}_x \\ -\hat{k}_y & \hat{k}_x & 0 \end{bmatrix}$$
- 9: Compute Rodrigues rotation matrix:

$$R = I_{3 \times 3} + \sin \theta K + (1 - \cos \theta) K^2$$
- 10: **return** R

morphology alignment has shown promise on fixed datasets, this approach becomes labor-intensive with diverse generated data. To address this, we propose a normalization method that simplifies human motion sequences into four basic joint chains (arms and legs) for efficient mapping of human postures to robot joint states.

The generated human motion sequences are represented in world coordinates using skeleton point data obtained from the diffusion model, as shown in Fig. 3. Focusing on limb movements, we express the 3D poses of limbs in robot coordinates, including both position and rotation. The robot coordinate system is defined with the positive x -axis as forward, the positive y -axis as leftward, and the positive z -axis as upward.

To compute the transformation matrix between the robot and the world coordinates, we use the root forward vector v_{RF} derived as the cross product of the vector from the root to the right hip v_{RRH} and the vector from the root to the left hip v_{RLH} . For each frame in the motion sequence, the rotation matrix is computed using the root forward vector of the first frame, denoted as v_{RF0} , and the current root forward vector v_{RF} , as described in Procedure 1.

For limb movements, we consider the left arm and left leg as example. The position of the left arm is represented by the vector from the left shoulder to the left wrist v_{LSW} , normalized by the arm length to yield the norm-position. The rotation of the left arm is described by the vector v_{LEW} , pointing from the left elbow to the left wrist. For the NAO robot, the default arm posture is horizontal, represented by $v_{LEW0} = [1, 0, 0]$. The rotation matrix between v_{LEW0} and v_{LEW} is computed and converted to a quaternion, providing the left arm's rotation. Similarly, the left leg's position is represented by the vector v_{LHA} , normalized by the leg length, and its rotation is described by v_{LAF} , pointing from the left ankle to the left foot. The default foot direction is

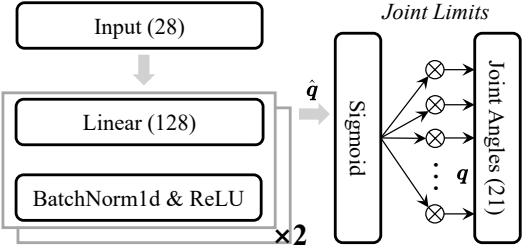


Fig. 5. The architecture of the angle signal net.

$v_{LAF0} = [1, 0, 0]$, and the corresponding quaternion is derived analogously.

The angle signal net maps input feature vectors, formed by concatenating the end-effector quaternion $Q \in \mathbb{R}^4$ and displacement $D \in \mathbb{R}^3$, to joint-space configurations while respecting the robot's physical joint limits as illustrated in Fig. 5. The network has an input dimension of $D_{in} = 28$ and an output dimension of $D_{out} = 21$, representing the predicted joint angles. It consists of two hidden layers with 128 units each, followed by batch normalization and ReLU activation. To ensure predictions remain within feasible joint limits, the raw output \hat{q} is passed through a sigmoid function:

$$q = \sigma(\hat{q}) \odot (q_{\max} - q_{\min}) + q_{\min} \quad (4)$$

where $\sigma(\cdot)$ is the sigmoid function, \odot denotes the element-wise multiplication, and q_{\max} and q_{\min} are the upper and lower joint limits, respectively.

The proposed loss function, referred to as NPR Loss, evaluates discrepancies between predicted and target kinematic configurations by jointly optimizing translational and rotational accuracy. The rotational error is measured using a quaternion-based loss, which computes the angular discrepancy between predicted and target orientations:

$$\begin{aligned} e &= q_{\text{target}} \otimes q_{\text{predict}}^{-1} \\ \mathcal{L}_{\text{quat}} &= 2 \cdot \arccos(w) \end{aligned} \quad (5)$$

where \otimes denotes quaternion multiplication, q_{predict}^{-1} is the inverse of the predicted quaternion, and w is the real part of e . The translational loss is computed as the mean squared error (MSE) between predicted and target positions:

$$\mathcal{L}_{\text{trans}} = \text{MSE}(p_{\text{target}}, p_{\text{predicted}}) \quad (6)$$

To account for differences in limb scale, the translational loss is weighted by limb lengths:

$$\mathcal{L}_{\text{trans}}^{\text{total}} = l_{\text{arm}} \cdot \mathcal{L}_{\text{trans}}^{\text{arm}} + l_{\text{leg}} \cdot \mathcal{L}_{\text{trans}}^{\text{leg}} \quad (7)$$

where l_{arm} and l_{leg} are the real robot's arm and leg lengths. The NPR Loss is a weighted combination of the translational and rotational losses:

$$\mathcal{L} = \omega_{\text{trans}} \cdot \mathcal{L}_{\text{trans}}^{\text{total}} + \omega_{\text{quat}} \cdot \mathcal{L}_{\text{quat}} \quad (8)$$

Fig. 4 illustrates the reference and the measured trajectory of the robot's arm endpoint while performing a waving motion. The reference trajectory is obtained by mapping generated motion data through the angle signal network,

TABLE I
NAO ROBOT'S JOINTS PARAMETERS.

Type	Max torque	Max speed	Stiffness	damping
Head	10.0	7.0	150.0	5.0
Arm	10.0	7.0	150.0	5.0
Leg pitch	20.0	6.4	200.0	5.0
Leg roll	20.0	4.0	150.0	5.0
Leg yaw pitch	30.0	4.0	200.0	5.0

TABLE II
REWARD TERMS, EXPRESSIONS AND WEIGHTS

Term	Expression	Weight
DoF torque	$\sum \ \tau_t\ ^2$	$-8.0e^{-4}$
Ankle torque	$\ \tau_{ankle}\ ^2$	$-2.0e^{-3}$
DoF acceleration	$\sum \ \ddot{q}_t\ ^2$	$-2.5e^{-7}$
Action rate	$\sum \ \dot{a}_t - \dot{a}_{t-1}\ $	$-2.0e^{-2}$
Action	$\sum \ a_t\ $	$-6.5e^{-4}$
Torso flat	$\sum (g_x^2 + g_y^2)$	-1.2
Feet flat	$\sum \exp(-(a_{x,i}^2 + a_{y,i}^2)/3.046e^{-4})$	0.3
Undesired contacts	$\sum (f_i > 10.0)$	-1.0
DoF target	$\exp(-\sum \dot{q}_t - \dot{q}_{target} /5.0)$	10.0

while the measured trajectory is captured from the physical robot. Notably, the proposed NPR Loss accurately captures the spatial features of the mapped motion, enabling precise joint sequence commands that enhance the performance of reinforcement learning.

IV. WHOLE BODY CONTROL POLICY TRAINING

A. State Space and Action Space

In the whole body control policy, the state representation plays a critical role in describing the current configuration of the humanoid robot. The state includes information about the robot's joint angles, joint velocities \dot{q}_t , and the global orientation and position of the robot's base. These inputs allow the control policy to understand the robot's current posture and anticipate future movements. In our approach, we encode the state as a concatenation of projected gravity g , joint position targets q_{target} , joint positions q_t and self-actions a_t .

The action space defines the possible commands that the policy can output to control the robot. It consists of 21-dim joint targets. The joints are controlled by a PD controller for precise position regulation, utilizing two parameters: stiffness K_p and damping K_d . The PD controller is formulated as:

$$\tau = K_p(\hat{q}_t - q_t) + K_d(\dot{\hat{q}}_t - \dot{q}_t) + \tau_0 \quad (9)$$

The robot's actuator are driven by four types of motors. To the best of our knowledge, there are currently no publicly available simulation information for the NAO robot in Isaac-Sim. Therefor, we provide a set of fine-tuned joint controller parameters, where the torque is measured in N m and speed in rad/s, as shown in Table I.

B. Reward Formulation

The reward function is crucial for guiding the learning process of the whole body control policy. In this work, we de-

TABLE III
RANGES FOR EACH PARAMETER BASED ON ESTIMATES OF UNCERTAINTY.

Parameter	Range
Static friction	$\mathcal{U}(0.3, 1.1)$
Dynamic friction	$\mathcal{U}(0.2, 0.7)$
Base mass	$\mathcal{U}(0.0, 0.2) + \text{default kg}$
Hand mass	$\mathcal{U}(0.0, 0.1) + \text{default kg}$
Push robot	interval $\in \mathcal{U}(4, 6)$ s $v_{xy} = 0.2$ m/s

sign a reward function that encourages the robot to accurately imitate the target human motion while maintaining stability. The reward function r_t consists of several components:

- 1) **Pose mapping reward.** Encourages the robot to align its end-effector positions (hands and feet) with the target positions generated from the text-driven human motion. Specifically, we compute this reward using the L_2 -norm of the difference between the joint commands output by the angle signal net and the actual executed joint positions of the robot.
- 2) **Security reward.** Ensure the robot does not incur self-inflicted damage during motion execution. The safety reward penalizes excessive contact forces at critical locations—specifically, the wrists, torso, and thigh links. By constraining the contact forces at these points to remain below predefined thresholds.
- 3) **Joint inhibition reward.** Penalizes large or abrupt changes in joint positions, torques, velocities, and accelerations to ensure smooth and practically applicable joint actions.

Certain components of the reward function are assigned positive weights, indicating that these actions are desirable and should be encouraged. Conversely, other components are assigned negative weights, signifying that these actions are undesirable or prohibited, and should be penalized accordingly. These defined reward functions are summarized in detail in Table II.

C. Termination Terms

Termination terms define the conditions that end a training episode. These terms are essential for providing meaningful feedback to the learning process. An episode terminates if any of the following conditions are met:

- 1) **Falling.** The robot is considered to have fallen if its base moves beyond a certain threshold from its original orientation, indicating a loss of balance. In our case, the robot has the unacceptable orientation if it rotates more than 60 degrees about the x or y axis, upon which the robot's state will terminate and reset.
- 2) **Timeout Condition.** Timeout occurs if the robot fails to complete the action within the designated time frame, resulting in the termination and reset of its state.

D. Sim-to-real Transfer

One of the key challenges in humanoid robot control is transferring policies trained in simulation to the real

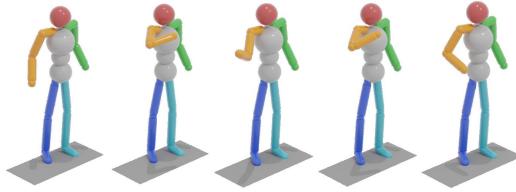


Fig. 6. Human motion animation using processed outputs of the diffusion model, with the textual description “wave right hand”.

world. Domain randomization involves introducing random variations in simulation parameters. Once the weights of the reward function are determined and effective motion can be trained in the simulation, domain randomization is incrementally introduced. For the NAO robot, training with domain randomization effectively reduces instability during real-world motion execution. Significantly, the joint controller of the physical NAO robot is highly sensitive to simulation parameters, and improper settings can lead to severe joint oscillations. Therefore, we chose not to randomize actuator gains. All the domain randomization we used are listed in Table III.

E. Training Details

Although the angle signal network ensures the correct alignment of the HipYaw rotation, the structural design of the NAO robot introduces a unique constraint: the left and right HipYaw joints are mechanically interconnected. To further ensure accuracy during reinforcement learning, the action of the right HipYaw joint is explicitly set to mirror that of the left HipYaw joint. The policy network architecture, trained with Proximal Policy Optimization (PPO) [33], consists of a MLP with three hidden layers containing 128, 128, and 128 neurons, respectively, and exponential linear unit (ELU) as the activation function. The training was conducted on a machine equipped with an NVIDIA RTX 3080TI GPU, using IsaacLab 2.0.0 as the primary RL framework.

V. EXPERIMENTAL RESULTS

A. Simulated Experiments

Text-driven motion generation. We utilize a well-trained MLCT model [22] as motion data generator, leveraging its robust capabilities to produce high-quality motion sequences, which is trained on the HumanML3D dataset. Text prompts adhere to the annotation guidelines established by HumanML3D, employing third-person perspectives as the primary subject. The original motion data is encoded into a 263-dimensional representation and can be visualized as a human animation, as shown in Fig. 6. This representation includes crucial components such as 3D joint rotations, which describe the orientation of each joint in space, displacements that capture changes in joint positions, velocities that indicate the speed of these movements, and foot contact information that provides details on the interaction between the feet and the ground. To facilitate accurate orientation and visualization, we transform the encoded motion representations back into 3D joint coordinates.

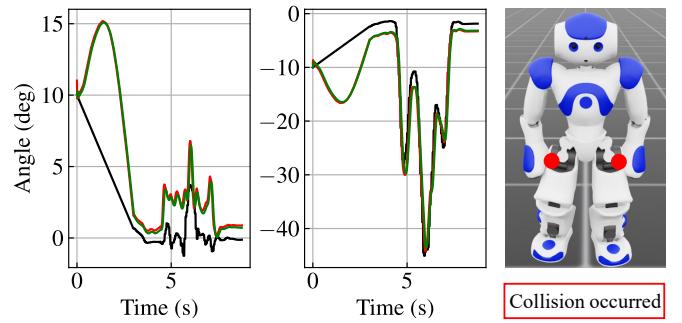


Fig. 7. Joint commands (black), actions (red), and actual joint position measurements (green) for the left shoulder roll and right shoulder roll during the transition from the default pose to the reference motion. At the start of the motion, the robot adjusts its joint positions to mitigate collisions, as reflected in the significant differences between the actions and commands.

Simulated robot. IsaacSim supports importing custom robot models. We integrate the URDF description file and STL model files obtained from the official ROS wiki into the environment and export them as USD files. We modify the STL models to address issues of unrealistic self-collisions inherent in the original files. The robot’s head, torso, wrists, pelvis and feet links are selected to generate collision bodies, as illustrated in Fig. 1. The NAO robot begins simulation in a crouched stance, with the hip height set to 230 mm. During the resampling phase, each robot randomly samples a single frame of joint sequences from the motion sequences as the learning target for the current episode. Additionally, we explored the use of sequential joint sequences as the learning objective for an episode, which demonstrated strong performance in simulation but encountered significant challenges when applied to the physical robot, failing to achieve effective transfer.

In the simulator, we enable self-collision detection and add contact sensors to all collision bodies of the robot. Particularly, there is a significant discrepancy between the default orientation of the robot’s arms and the initial position of the reference motion. Fig. 7 illustrates the noticeable collisions that occur between the robot’s hands and thighs when transitioning from the default pose to the reference motion. After optimization with the collision avoidance reward function, the robot adjusts its joint positions to prevent such collisions.

B. Real-world Implementations

Hardware platform. We validated our proposed joint motion control method on the NAO robot. The NAO robot stands 57.4 cm tall and features 23 DoFs. In this work, we utilized 21 DoFs for controlling the robot’s arms and legs, excluding the 2 DoFs in the head. The motion policy was exported in ONNX format and executed using ONNX Runtime for inference. The inference engine was deployed locally on NAO, with each inference taking approximately 1 ms. Data communication between the inference engine and the NAO robot was handled via ROS2 topics. As part of the policy input, the projected gravity vector was directly provided by the simulation environment during training. However,

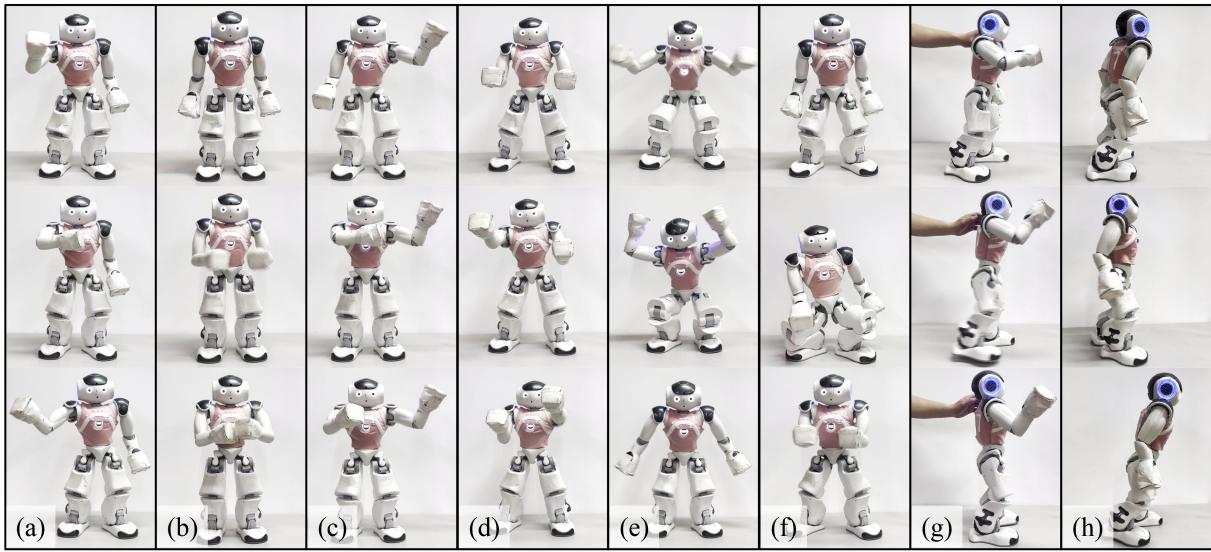


Fig. 8. Time sequence snapshots of real NAO motion control, including (a) waving right hand, (b) touching the left hand with right hand, (c) playing the violin, (d) boxing, (e) squatting while raising both hands, (f) squatting to pick up an object, (g) recovering balance after being pushed, and (h) forward walking.

this cannot be directly measured on the physical robot. To address this, we employed the Madgwick Orientation Filter [34] to obtain a quaternion representation of the robot’s base orientation. This orientation was then applied to a unit z-axis vector to estimate the projected gravity. Other observations, such as joint positions, were gathered from the NAO robot’s joint position sensors.

Motion control results. The real-world motion control results are shown in Fig. 8, where the robot demonstrates the ability to replicate upper-body motions, as shown in (a) to (e), which primarily focus on the reproduction of various upper-limb actions. These results indicate the system’s capability to map human-like upper-body movements onto the robot’s joint commands. For lower-body motions, we intentionally relaxed the tracking of joint commands and disregarded ankle joint commands to allow for greater flexibility in leg movements. This design choice is illustrated in Fig. 8(g), where the robot adjusts its balance after being pushed, demonstrating adaptive behavior rather than strict adherence to predefined joint trajectories. Additionally, Fig. 8(f) illustrates a limitation in the reproduction of the reference arm motion. While the intended motion was designed to position the both hands in front of the chest, only one hand remaining in front of the torso. This discrepancy arises from the reward function’s emphasis on collision avoidance, as the robot’s legs occupy significant space and restrict feasible arm trajectories. Finally, Fig. 8(h) depicts the robot’s attempt at forward walking. Although the robot mimics certain characteristics of human walking, such as leg swings and a wider stance, it does not lift its feet off the ground during the motion. This incomplete replication is considered a failure case in achieving fully dynamic bipedal locomotion.

Discussions and limitations. To the best of our knowledge, we have thoroughly investigated the current method-

ologies for implementing motion control on the NAO robot using reinforcement learning policies. One notable limitation of our approach is that it samples actions from individual motion frames without accounting for temporal coherence between consecutive frames. Although we attempted to incorporate complete motion sequences as commands during reinforcement learning training, successful transfer from simulation to the physical robot has not yet been achieved.

In the case of walking motions, a failure scenario, our method does not consider the robot’s velocity information. Consequently, while our approach effectively maps human motions to robot actions for stationary tasks, it struggles with dynamic motions that inherently involve self-locomotion. Furthermore, the NAO robot’s limited torso degrees of freedom pose additional challenges. Specifically, motions requiring flexibility in the waist, such as bending or turning, are constrained and reduced to mappings of upper-body movements.

VI. CONCLUSIONS AND FUTURE WORKS

This paper introduces a framework that enables humanoid robots to imitate human motions generated by text-driven diffusion models. The framework employs a normalized motion representation based on NPR Loss and leverages an angle signal network to map human motion data into robot joint commands. A reinforcement learning-based joint motion controller is used to optimize these commands, which are trained in simulation and successfully deployed on a physical NAO robot, demonstrating effective sim-to-real transfer. This approach enhances the flexibility of motion generation without relying on external motion capture systems and includes an NAO simulation model fully compatible with IsaacSim.

Currently, the proposed method primarily focuses on imitating joint position commands and does not account for temporal dependencies between these commands, as the current

NAO simulation parameters do not support such advanced modeling. To address this limitation, further exploration will be conducted to enable the learning of continuous motion commands, thereby developing a more versatile and broadly applicable robotic simulation model.

REFERENCES

- [1] J. Koenemann, F. Burget, and M. Bennewitz, “Real-time imitation of human whole-body motions by humanoids,” in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 2806–2812.
- [2] G. Tevet, B. Gordon, A. Hertz, A. H. Bermano, and D. Cohen-Or, “Motionclip: Exposing human motion generation to clip space,” in *European Conference on Computer Vision*. Springer, 2022, pp. 358–374.
- [3] M. Petrovich, M. J. Black, and G. Varol, “Action-conditioned 3d human motion synthesis with transformer vae,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 10985–10995.
- [4] M. Zhang, Z. Cai, L. Pan, F. Hong, X. Guo, L. Yang, and Z. Liu, “Motondiffuse: Text-driven human motion generation with diffusion model,” *arXiv preprint arXiv:2208.15001*, 2022.
- [5] J. Li, C. Xu, Z. Chen, S. Bian, L. Yang, and C. Lu, “Hybrik: A hybrid analytical-neural inverse kinematics solution for 3d human pose and shape estimation,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 3383–3393.
- [6] T. He, Z. Luo, W. Xiao, C. Zhang, K. Kitani, C. Liu, and G. Shi, “Learning human-to-humanoid real-time whole-body teleoperation,” *arXiv preprint arXiv:2403.04436*, 2024.
- [7] S. Mukherjee, D. Paramkusam, and S. K. Dwivedy, “Inverse kinematics of a nao humanoid robot using kinect to track and imitate human motion,” in *2015 International Conference on Robotics, Automation, Control and Embedded Systems (RACE)*. IEEE, 2015, pp. 1–7.
- [8] K. Otani and K. Bouyarmane, “Adaptive whole-body manipulation in human-to-humanoid multi-contact motion retargeting,” in *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*. IEEE, 2017, pp. 446–453.
- [9] P. M. Viceconte, R. Camoriano, G. Romualdi, D. Ferigo, S. Dafarra, S. Traversaro, G. Oriolo, L. Rosasco, and D. Pucci, “Adherent: Learning human-like trajectory generators for whole-body control of humanoid robots,” *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2779–2786, 2022.
- [10] H. Khalil, E. Coronado, and G. Venture, “Human motion retargeting to pepper humanoid robot from uncalibrated videos using human pose estimation,” in *2021 30th IEEE International Conference on Robot & Human Interactive Communication (RO-MAN)*. IEEE, 2021, pp. 1145–1152.
- [11] S. M. Khansari-Zadeh and A. Billard, “Learning stable nonlinear dynamical systems with gaussian mixture models,” *IEEE Transactions on Robotics*, vol. 27, no. 5, pp. 943–957, 2011.
- [12] D. Holden, J. Saito, and T. Komura, “A deep learning framework for character motion synthesis and editing,” *ACM Transactions on Graphics (TOG)*, vol. 35, no. 4, pp. 1–11, 2016.
- [13] C. Guo, X. Zuo, S. Wang, S. Zou, Q. Sun, A. Deng, M. Gong, and L. Cheng, “Action2motion: Conditioned generation of 3d human motions,” in *Proceedings of the 28th ACM International Conference on Multimedia*, 2020, pp. 2021–2029.
- [14] M. Petrovich, M. J. Black, and G. Varol, “Action-conditioned 3d human motion synthesis with transformer vae,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 10985–10995.
- [15] C. Guo, S. Zou, X. Zuo, S. Wang, W. Ji, X. Li, and L. Cheng, “Generating diverse and natural 3d human motions from text,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 5152–5161.
- [16] Z. Wang, P. Yu, Y. Zhao, R. Zhang, Y. Zhou, J. Yuan, and C. Chen, “Learning diverse stochastic human-action generators by learning smooth latent transitions,” in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 07, 2020, pp. 12281–12288.
- [17] H. Cai, C. Bai, Y.-W. Tai, and C.-K. Tang, “Deep video generation, prediction and completion of human action sequences,” in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 366–382.
- [18] X. Chen, B. Jiang, W. Liu, Z. Huang, B. Fu, T. Chen, and G. Yu, “Executing your commands via motion diffusion in latent space,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 18 000–18 010.
- [19] P. Jin, Y. Wu, Y. Fan, Z. Sun, Y. Wei, and L. Yuan, “Act as you wish: Fine-grained control of motion diffusion model with hierarchical semantic graphs,” *arXiv preprint arXiv:2311.01015*, 2023.
- [20] G. Tevet, S. Raab, B. Gordon, Y. Shafir, D. Cohen-Or, and A. H. Bermano, “Human motion diffusion model,” in *International Conference on Learning Representations*, 2023.
- [21] H. Kong, K. Gong, D. Lian, M. B. Mi, and X. Wang, “Priority-centric human motion generation in discrete latent space,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 14 806–14 816.
- [22] M. Hu, M. Zhu, X. Zhou, Q. Yan, S. Li, C. Liu, and Q. Chen, “Efficient text-driven motion generation via latent consistency training,” *arXiv preprint arXiv:2405.02791*, 2024.
- [23] X. B. Peng, P. Abbeel, S. Levine, and M. Van de Panne, “Deepmimic: Example-guided deep reinforcement learning of physics-based character skills,” *ACM Transactions On Graphics (TOG)*, vol. 37, no. 4, pp. 1–14, 2018.
- [24] J. Tan, T. Zhang, E. Coumans, A. Iscen, Y. Bai, D. Hafner, S. Bohez, and V. Vanhoucke, “Sim-to-real: Learning agile locomotion for quadruped robots,” *arXiv preprint arXiv:1804.10332*, 2018.
- [25] G. Li, A. Ijspeert, and M. Hayashibe, “Ai-cpg: Adaptive imitated central pattern generators for bipedal locomotion learned through reinforced reflex neural networks,” *IEEE Robotics and Automation Letters*, 2024.
- [26] G. A. Castillo, B. Weng, W. Zhang, and A. Hereid, “Robust feedback motion policy design using reinforcement learning on a 3d digit bipedal robot,” in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 5136–5143.
- [27] Y. Li, L. Zheng, Y. Wang, E. Dong, and S. Zhang, “Impedance learning-based adaptive force tracking for robot on unknown terrains,” *IEEE Transactions on Robotics*, 2025.
- [28] W. Zhao, J. P. Queralta, and T. Westerlund, “Sim-to-real transfer in deep reinforcement learning for robotics: a survey,” in *2020 IEEE symposium series on computational intelligence (SSCI)*. IEEE, 2020, pp. 737–744.
- [29] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter, “Learning agile and dynamic motor skills for legged robots,” *Science Robotics*, vol. 4, no. 26, p. eaau5872, 2019.
- [30] M. Mittal, C. Yu, Q. Yu, J. Liu, N. Rudin, D. Hoeller, J. L. Yuan, R. Singh, Y. Guo, H. Mazhar, A. Mandlekar, B. Babich, G. State, M. Hutter, and A. Garg, “Orbit: A unified simulation framework for interactive robot learning environments,” *IEEE Robotics and Automation Letters*, vol. 8, no. 6, pp. 3740–3747, 2023.
- [31] C. Guo, S. Zou, X. Zuo, S. Wang, W. Ji, X. Li, and L. Cheng, “Generating diverse and natural 3d human motions from text,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022, pp. 5152–5161.
- [32] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black, “Smpl: A skinned multi-person linear model,” in *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*, 2023, pp. 851–866.
- [33] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms,” *arXiv preprint arXiv:1707.06347*, 2017.
- [34] S. Madgwick *et al.*, “An efficient orientation filter for inertial and inertial/magnetic sensor arrays,” *Report x-io and University of Bristol (UK)*, vol. 25, pp. 113–118, 2010.