TextOp: Real-time Interactive Text-Driven Humanoid Robot Motion Generation and Control

Weiji Xie[1,2*] Jiakun Zheng[1,3*] Jinrui Han[1,2] Jiyuan Shi[1]
Weinan Zhang[2†] Chenjia Bai[1†] Xuelong Li[1]
[1]Institute of Artificial Intelligence (TeleAI), China Telecom [2]Shanghai Jiao Tong University
[3]East China University of Science and Technology
[*]Equal contribution [†]Corresponding author

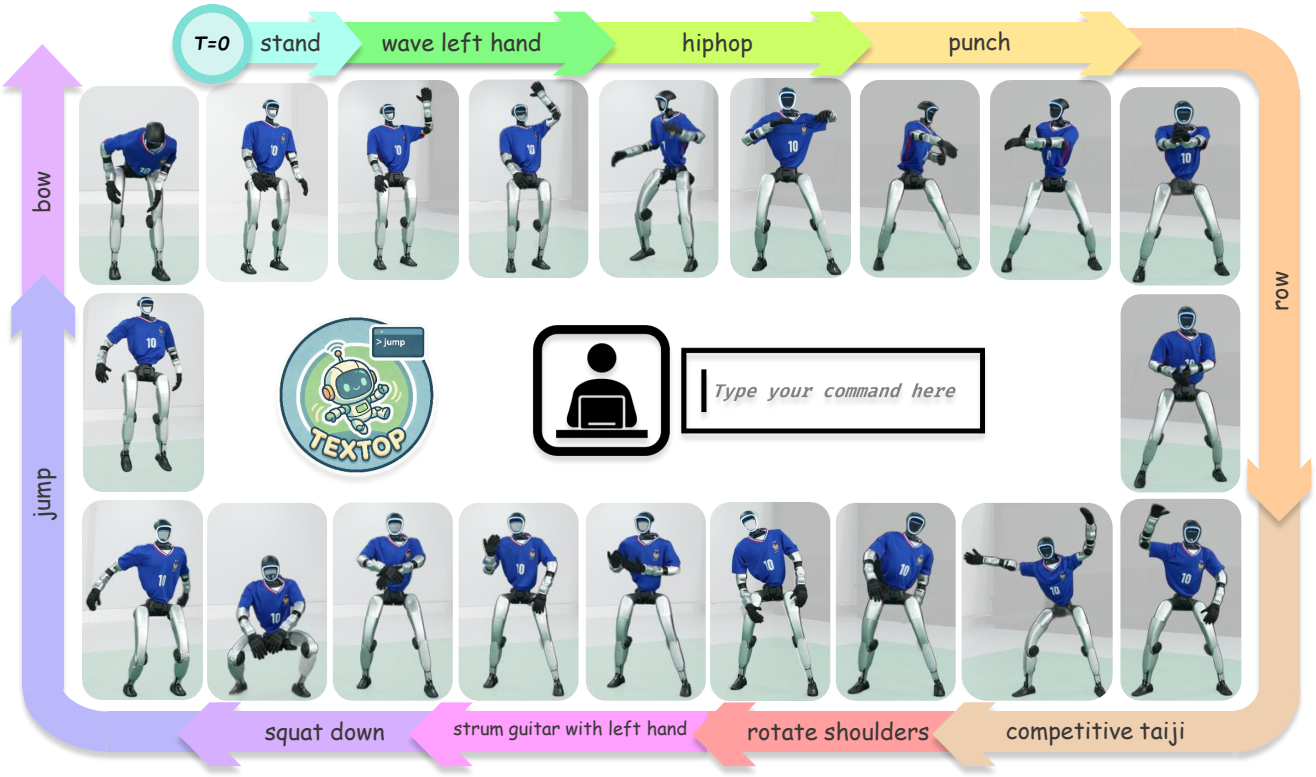

Fig. 1: **TextOp enables a humanoid robot to execute a seamless sequence of diverse skills—ranging from expressive gestures to complex physical tasks—driven by real-time, interactive text commands from the user in a single continuous trial.**

***Abstract*—Recent advances in humanoid whole-body motion tracking have enabled the execution of diverse and highly coordinated motions on real hardware. However, existing controllers are commonly driven either by predefined motion trajectories, which offer limited flexibility when user intent changes, or by continuous human teleoperation, which requires constant human involvement and limits autonomy. This work addresses the problem of how to drive a universal humanoid controller in a real-time and interactive manner. We present TextOp, a real-time text-driven humanoid motion generation and control framework that supports streaming language commands and on-the-fly instruction modification during execution. TextOp adopts a two-level architecture in which a high-level autoregressive motion diffusion model continuously generates short-horizon kinematic trajectories conditioned on the current text input, while a low-level motion tracking policy executes these trajectories on a physical humanoid robot. By bridging interactive motion generation with robust whole-body control, TextOp unlocks free-form intent expression and enables smooth transitions across multiple challenging behaviors such as dancing and jumping, within a single continuous motion execution. Extensive real-robot experiments and offline evaluations demonstrate instant responsiveness, smooth whole-body motion, and precise control. The project page and the open-source code are available at https://text-op.github.io/.**

I. Introduction

Recent years have witnessed rapid progress in universal humanoid whole-body motion tracking systems [32, 18, 12, 17, 64, 26], capable of robustly executing a wide range of highly coordinated behaviors on real hardware. Leveraging advances in reinforcement learning [3, 43] and large-scale physical simulation [34, 50], modern humanoid platforms can reliably track diverse kinematic references and perform complex full-body motions under a unified control framework. From a control perspective, the problem of executing rich humanoid behaviors is increasingly well-addressed.

However, **how to flexibly and interactively drive such universal controllers remains an open challenge**. In existing systems, universal humanoid controllers are commonly driven by either predefined motion trajectories [12, 17, 70] or continuous human teleoperation [65, 66, 18]. Predefined trajectories, while effective for reproducing specific motions, offer little flexibility at runtime: once execution begins, the robot's behavior is mainly determined by the trajectory and cannot easily adapt to changes in intent. Teleoperation, on the other hand, enables direct guidance but relies heavily on sustained human involvement, making long-term operation labor-intensive and limiting the autonomy of the robot. As a result, a gap persists between high-level human intent expression and low-level humanoid execution, despite significant advances in the underlying control technology.

In this context, natural language has emerged as a powerful modality for guiding robot behavior. A growing body of work on text-conditioned motion generation demonstrates that language can flexibly describe complex, semantically meaningful motions [11, 49, 25]. Inspired by this idea, some approaches first explicitly generate full motion sequences and then employ a tracking policy to track them on the robot [18, 44, 64, 17], while others embed motion generation implicitly within end-to-end models that map language directly to control signals [45, 55, 27]. Although differing in formulation, these methods share a common characteristic: behavior is determined largely upfront, with limited or no support for revising commands during execution. This limits their applicability to real-world humanoid control, where interaction is continuous and intent may evolve over time.

Separately, the humanoid animation community has explored interactive motion generation techniques that allow users to steer motions on the fly [8, 47, 71, 56, 7]. A key characteristic is that both the conditioning signal and the generated motion are treated as time-varying signal streams, rather than producing a complete trajectory from a static prompt. These approaches show the feasibility of responsive, user-in-the-loop motion synthesis, but are primarily developed for kinematic animation and virtual characters, without integration into real-robot whole-body control pipelines, due to missing dynamic constraints and the unaddressed sim-to-real gap.

Taken together, these observations reveal a missing link between interactive language-based intent expression and real-time, physically executable humanoid control. Based on this insight, we propose **TextOp**, a real-time text-driven humanoid motion generation and control framework that drives a universal whole-body controller using streaming language commands with on-the-fly instruction modification. Rather than committing to a fixed motion plan, **TextOp** continuously interprets evolving text input and generates short-horizon motion references suitable for immediate execution on real hardware.

TextOp adopts a two-level architecture. At the high level, an autoregressive, text-conditioned motion diffusion model incrementally generates short-horizon kinematic trajectories conditioned on the current language input and recent motion context. At the low level, a robust whole-body motion tracking policy executes these trajectories on a physical humanoid robot, converting kinematic references into joint-level commands at high frequency. This separation allows human intent to be updated flexibly without compromising control stability.

To effectively integrate interactive motion generation with real-world humanoid control, **TextOp** incorporates two critical design choices tailored to robotic execution. First, we adopt a robot-skeleton motion representation that reflects the constrained, single-DoF joint structure of humanoid robots, enabling compact and effective kinematic motion representation for the motion generation module. Second, to mitigate the distribution gap between motion datasets and the reference motions produced online by the generator, we augment the training data of the tracking policy with generator-produced motions under realistic text streams. Together, these designs align the high-level generator with the low-level controller and improve robustness during real-robot deployment.

By bridging interactive motion generation with whole-body control, **TextOp** enables the robot to execute long-horizon, continuous motions that smoothly transition across multiple challenging behaviors—such as dancing and jumping—while dynamically responding to user-provided language commands in real time, as validated on a real robot. Offline evaluations further demonstrate that carefully designed robot-skeleton motion representations, together with augmenting tracker training with generator-produced motions, enable the system to reliably translate streaming language commands into real-world robot motion.

In summary, this work makes the following contributions:

- We present **TextOp**, a real-time text-driven humanoid motion generation and control system that realizes a novel control mode in which the robot is driven by streaming natural language commands.
- We introduce two key design choices: (i) a robot-skeleton motion representation to improve generation quality for robot structure, and (ii) a data-augmentation strategy via motion generation to reduce the distribution gap between motion datasets and generator-produced reference motions.
- We validate **TextOp** through extensive real-robot experiments and offline evaluations, demonstrating instant responsiveness, smooth whole-body motion, and precise control.

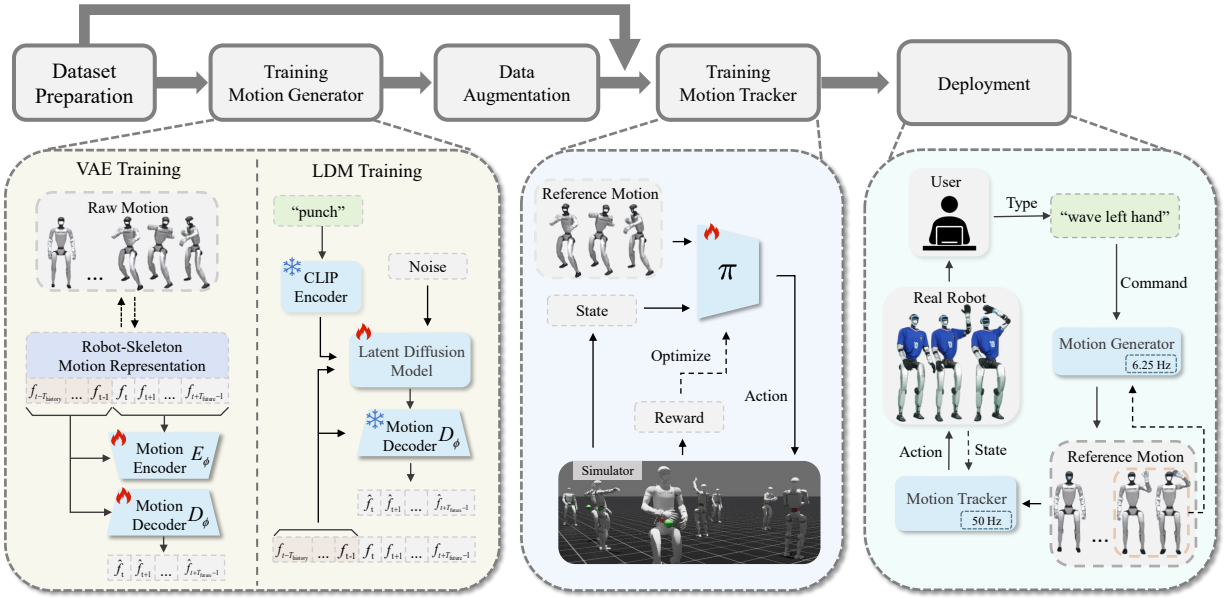

Fig. 2: **Overview of TextOp's framework.** The framework consists of three main parts: (a) **Interactive Motion Generation**, including VAE training and LDM training, which together model future reference motion sequences conditioned on history motion and text prompt in an autoregressive style; (b) **Dynamic Motion Tracking**, where the MLP-based policy π takes reference motions and robot states to generate joint actions, trained in the simulation for stable execution; (c) **Deployment**, where the real-time user text prompt is converted into motions by the generator, translated into actions by the tracking policy based on the robot state, and executed on the physical robot.

II. Related Work

A. Whole-body Humanoid Control

Humanoid whole-body control has advanced significantly, enabling realistic, diverse, and coordinated motion execution on physical robots. From the perspective of the driving mechanism, humanoid robot controllers are mainly driven by three command paradigms.

Simple Task Command encompasses control methods that enable the robot to execute straightforward high-level objectives, such as locomotion [14, 13, 73, 52, 61, 57], navigation [5, 6, 10], object manipulation [20, 59], sports [48, 16, 42], and standing up [23, 21]. This paradigm often provides fast response and reliability for single tasks, but it offers limited expressiveness for complex, semantically rich motions.

Static motion tracking relies on predefined reference trajectories to achieve agile and high-fidelity whole-body motion reproduction, but commits to a fixed motion plan that is difficult to adjust in real time. This paradigm leverages motion data from motion capture [33, 24], video-based extraction [1, 46, 54], and multimodal generation models in both explicit [18, 44] and implicit [45, 55, 27, 60] ways, together with motion retargeting [2, 62] and reinforcement learning methods [28, 32, 58, 17, 70, 18, 12, 38, 64, 9, 74] for whole-body motion tracking. We note that a few prior methods [45, 32] technically allow modifying text inputs at runtime; however, this capability is not a design focus and is only weakly supported in practice, with no systematic empirical validation.

Teleoperation provides flexible real-time control through motion capture suits [65], VR devices [66, 30], camera-based systems [19], or exoskeletons [35, 63, 4], but requires continuous human involvement, limiting autonomy and scalability for long-term operation.

This work goes beyond both static motion tracking and teleoperation paradigms, achieving flexible semantic intent expression and continuous real-time control with little reliance on manual labor.

B. Interactive Motion Generation

Text-driven motion generation has advanced significantly, with natural language emerging as a powerful modality for describing complex, semantically rich motions. Diffusion-based approaches [68, 49, 11], followed by language-model-integrated methods [67, 25, 53, 72], have demonstrated that compact textual descriptions can be effectively translated into diverse and high-quality human motions. Nevertheless, most existing motion generation systems remain offline, producing complete motion sequences from static inputs.

Interactive motion generation has been proposed to address this limitation. Early approaches enable interactive control using low-dimensional signals such as target position and direction [8, 47], while later work incorporate text as control signals. [71] introduces a diffusion-based autoregressive motion primitive model that conditions on historical frames and text prompts for real-time generation. [69] uses a two-stage method that first pretrain an autoregressive diffusion model to capture short-term body movements and then adapts it for

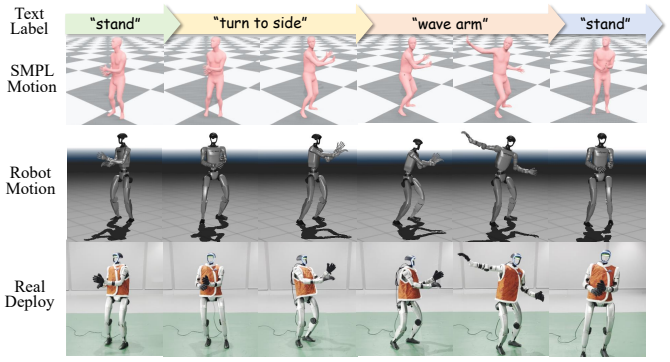

Fig. 3: **Illustration of the time-aligned data format including text labels, SMPL motions, and robot motions.**

real-time interactive control. [56] performs continuous causal latent autoregression, alleviating error accumulation caused by discrete tokenization. [7] proposes a diffusion-forcing framework with tailored temporal attention and scheduling to enable seamless real-time synthesis.

Despite these advances, existing methods focus on virtual character animation and lack physical robot deployment. Our work bridges this gap by integrating autoregressive motion generation with a whole-body tracking policy for real humanoid control.

III. Method

A. Overview

We consider a humanoid robot operating in a real-time streaming control setting. At each discrete time step t, the system receives a language command l_t together with the previous robot state $x^{\text{robot}}_{<t}$. The goal of the system is to generate a control signal, i.e., target joint commands a_t, such that the resulting physical robot behavior x^{robot}_t remains semantically consistent with the language command while maintaining stable and smooth whole-body motion.

TextOp decomposes this task into two levels: Interactive Motion Generation and Dynamic Motion Tracking. Specifically, the high-level motion generator G is implemented as an autoregressive motion latent diffusion model. At each time step, it produces a reference motion sequence $x^{\text{ref}}_{t:t+T_{\text{future}}-1}$ of length $T_{\text{future}} = 8$, conditioned on the past $T_{\text{history}} = 2$ frames of reference motion $x^{\text{ref}}_{t-T_{\text{history}}:t-1}$ and the current language command l_t, i.e.,

$$x^{\text{ref}}_{t:t+T_{\text{future}}-1} = G\big(x^{\text{ref}}_{t-T_{\text{history}}:t-1}, l_t\big).$$

The low-level tracking policy π then converts the kinematic reference motion into executable control signals based on the previous robot states, i.e.,

$$a_t = \pi\big(x^{\text{robot}}_{t-1}, a_{t-1}, x^{\text{ref}}_{t:t+T_{\text{ref}}-1}\big),$$

ensuring that the executed motion closely follows the reference trajectory while remaining stable and smooth.

B. Data Preparation

Our training data is built on the large-scale public motion capture dataset AMASS [33], supplemented with a small amount of private motion data featuring complex dance and martial arts motions. All motions are retargeted from the SMPL [29] skeleton to the robot skeleton using GMR [2], extracting the feet contact indicator and resampling to a unified frequency of 50 Hz. We further filter the retargeted motions by pretraining a universal motion tracking policy with privileged information and discarding motion sequences that cannot be reliably tracked. Fig. 3 illustrates our data format across different modalities.

For training the motion tracking policy, the filtered motions are further segmented into short clips of less than 40 seconds, resulting in a total of 12,296 motion clips with an aggregate duration of 40.67 hours from AMASS. The private dataset contributes an additional 403 clips totalling 3.12 hours.

For motion generation, we pair filtered AMASS motions with language annotations from BABEL [40], an action-centric dataset in which long motion sequences are segmented, each annotated with a text description, e.g., "walk", "jump". Such frame-level annotations are essential for effective language guidance and natural switching during generation, as demonstrated in prior work [71, 7]. We perform mirror augmentation on both the motion sequences and their corresponding textual descriptions, resulting in a dataset of 83,478 segment–text pairs. For complex long-horizon motions in the private dataset, the entire motion is assigned a unique label wrapped with "$\langle\cdot\rangle$" markers.

C. Interactive Motion Generation

Robot-Skeleton Motion Representation. In contrast to humanoid skeletons like SMPL, where joints are modeled as 3-DoF ball joints to represent full human articulation, robotic skeletons typically employ single-DoF rotational joints, yielding more constrained yet tractable kinematics. Therefore, we propose a DoF-based local incremental motion representation for the robot skeleton, instead of HumanML3D [15]-style representation. This choice naturally enforces robot kinematic constraints and ensures invariance to global pose, providing a compact yet sufficient feature space for the motion generator.

For a robot with n_q degrees of freedom, we define the per-frame motion feature $f_t \in \mathbb{R}^{d_{\text{feat}}}$ as:

$$f_t = \Big[\phi(r_t),\ \Delta\psi_t,\ c_t,\ \Delta p^{\text{local}}_t,\ h_t,\ q_t,\ \Delta q_t\Big],$$

where $r_t = (\text{roll}_t, \text{pitch}_t, \text{yaw}_t)$ denotes the root orientation in intrinsic Euler angles and $\phi(r_t) = [\sin(\text{roll}_t), \cos(\text{roll}_t) - 1, \sin(\text{pitch}_t), \cos(\text{pitch}_t) - 1]$ is a continuous trigonometric encoding of roll and pitch. $\Delta\psi_t = \text{yaw}_{t+1} - \text{yaw}_t$ denotes the incremental change of the root yaw, $c_t \in \{0,1\}^{n_c}$ represents binary contact indicators of feet, and $\Delta p^{\text{local}}_t = R_z(\text{yaw}_t)^\top (p_{t+1} - p_t)$ is the root translation increment expressed in a yaw-aligned local frame. Furthermore, h_t denotes the root height, $q_t \in \mathbb{R}^{n_q}$ the joint positions, and $\Delta q_t = q_{t+1} - q_t$ the corresponding joint-wise increments.

Model Architecture. The motion generator G models the text-conditional probabilistic distribution of the robot motion in an autoregressive style. It follows the DART [71] architecture and is implemented as a VAE combined with a latent diffusion model (LDM), both are Transformer-based networks. The VAE defines a motion latent space by encoding future motion frames into a latent variable conditioned on the motion history, $z \sim E_\phi(\cdot \mid f_{t-T_{\text{history}}:t+T_{\text{future}}-1})$, which can be decoded to reconstruct the future frames $\hat{f}_{t:t+T_{\text{future}}-1} = D_\phi(f_{t-T_{\text{history}}:t-1}, z)$. The LDM models the text-conditional distribution of future motion latents, where language commands are embedded using a pretrained CLIP encoder [41] as $e_t = \text{CLIP}(l_t)$, and a diffusion transformer predicts clean latents from noise via $\hat{z} = F_\theta(z_k, k, f_{t-T_{\text{history}}:t-1}, e_t)$, trained in a DDPM [22] style with $n_{\text{denoise}} = 5$ steps. During inference, classifier-free guidance is applied with scale $\sigma_{\text{CFG}} = 5$ to enhance semantic alignment.

Training Process. The VAE is first trained on motion features reconstruction, and its parameters are fixed during subsequent LDM training. The LDM is then trained to denoise motion latents by minimizing a combination of feature reconstruction, latent reconstruction, and geometric losses. To reduce the distribution gap between training and deployment, a self-rollout strategy is adopted, where N consecutive overlapping motion segments are processed sequentially and the history of each segment is randomly replaced by the predicted future of the previous segment. For classifier-free guidance, the text embedding is randomly set to zero during training.

D. Dynamic Motion Tracking

Training Universal Motion Tracker. The low-level universal motion tracker is trained through goal-conditioned RL in large-scale physical simulation with IsaacLab [34]. We adopt a one-stage RL pipeline to train the MLP-based tracking policy, discarding more complex training pipelines and network designs to maintain simplicity while ensuring balanced performance.

The policy takes as input the robot's proprioceptive state together with a reference motion over $T_{\text{ref}} = 5$ future frames. During training, an episode starts with a motion segment randomly sampled from the dataset and the robot initialized to the state of the first frame, and ends upon timeout or large tracking deviations. The policy is optimized via PPO [43] with a reward signal that combines tracking objectives and regularization terms. The training setup, including observation structure, reward formulation, and domain randomization, mainly follows established methods such as BeyondMimic [28] and is detailed in the appendix.

Data Augmentation via Motion Generation. To reduce the distribution gap between motion datasets and the reference motions induced by the high-level generator, we augment the training data with generated motions. Specifically, we randomly sample text annotations from the BABEL training set to form 20-second text streams and feed them into the high-level generator, producing 5,368 motion clips totalling 31.48 hours. This approach exposes the policy to realistic variability and noise from generator outputs, enhancing its tracking performance during deployment.

E. Deployment

The system is deployed on a Unitree G1 humanoid robot [51] with 29 degrees of freedom. The motion tracking policy executes on the robot's onboard computer at 50 Hz using ONNX Runtime [37], while the motion generator runs at 6.25 Hz on an external workstation equipped with an NVIDIA RTX 4090 GPU using TensorRT [36]. Communication between the two components occurs over a wired or wireless network, with a motion buffer to synchronize their execution. User textual commands are streamed to the external workstation in real time, encoded via CLIP, and incorporated into subsequent motion generation, enabling interactive control.

IV. Experiments

We assess the effectiveness of **TextOp** through a combination of real-world deployment and large-scale simulation studies. Our experiments are designed to evaluate both the practical performance of the system and the contributions of its key components.

Specifically, the evaluation aims to answer the following research questions:

- **Q1:** Can **TextOp** achieve precise, stable, and responsive whole-body behaviors in the real world?
- **Q2:** Can the motion generator produce high-quality and semantically consistent motions from text commands in an interactive setting?
- **Q3:** Can the motion tracking policy robustly execute diverse reference motions, especially those produced by the generator?

A. Experimental Setup

To systematically evaluate **TextOp** system, we adopt a comprehensive set of evaluation metrics covering **Motion Quality** and **Tracking Fidelity**. Formal definitions of these metrics are provided in the appendix, and different system components are evaluated using the corresponding subsets of these metrics.

- **Motion Quality.** Evaluates the overall quality of text-conditioned motion following prior work [71, 7], measured by FID and Diversity for distribution similarity to the dataset, R-precision (R@K) and multimodal distance (MM-Dist) for semantic alignment with language commands, and PJ and AUJ for temporal smoothness across command transitions.
- **Tracking Fidelity.** Evaluates tracking fidelity of the motion tracker under physical constraints by reporting the success rate (Succ) and motion discrepancy metrics between executed and reference trajectories, including global MPJPE ($E_{\text{g-mpjpe}}$, mm), root-relative MPJPE (E_{mpjpe}, mm), acceleration error (E_{acc}, mm/frame^2), and velocity error (E_{vel}, mm/frame), following [31, 32].

For real-world deployment, the motion tracking policy is trained on a server equipped with an NVIDIA A100 GPU for

Fig. 4: **Continuous diverse skill execution in the real robot.** The robot seamlessly performs a wide range of tasks, including multiple dance styles, dynamic jumping behaviors, instrument-playing motions, and expressive gestures. For complex long-horizon motions in the private dataset, the entire motion is assigned a unique label wrapped with "$\langle\cdot\rangle$" markers.

approximately one week, using 150,000 training steps, while for simulation studies, the policy is trained for 50,000 steps due to computational resource constraints. The motion generation model is trained on the same hardware for approximately two days, with 200,000 steps for the VAE and 300,000 steps for the LDM. The hyperparameter setup is detailed in the appendix.

B. Real-world Deployment

To address **Q1**, we evaluate **TextOp** in the real world through a combination of qualitative demonstrations, quantitative robustness tests, and real-time performance measurements.

Complex Skill Demonstrations and Interactive Control. We evaluate the expressiveness and robustness of **TextOp** system through a series of real-world demonstrations performed on the physical robot, as shown in Fig. 1 and Fig. 4. The robot executes a diverse set of skills, including locomotion, dance motions, martial arts sequences, and gestures. Beyond isolated skill execution, a distinctive feature of **TextOp** is its ability to perform continuous long-horizon tasks: the robot seamlessly executes a sequence of varied commands in a single, unbroken trial. This "one-take" demonstration underscores the system's capacity to maintain motion coherence and control stability. Throughout the entire process, the interactive control capability of **TextOp** allows users to modify commands on the fly, guiding the robot's behavior and enabling smooth composition and transitions between skills.

TABLE I: **Quantitative evaluation on 30-second long-horizon real-robot executions with diverse text command streams.** **TextOp** achieves a strong tracking fidelity across all evaluated tasks.

Text Command	Succ ↑	$E_{\text{g-mpjpe}}$ ↓	E_{mpjpe} ↓	E_{acc} ↓	E_{vel} ↓
Random	16 / 20	337.169	153.920	1.452	2.621
Looping					
"punch"	10 / 10	355.560	123.598	2.705	4.891
"wave right hand"	8 / 10	316.480	72.184	2.757	2.906
"strum guitar with left hand"	10 / 10	191.562	88.662	1.418	1.974
"play the violin"	10 / 10	107.584	33.941	0.518	0.750

Robustness Analysis. We evaluate the system's robustness through long-horizon real-robot experiments under varying text command streams. In each 30-s trial, the robot executes either randomly sampled commands or fixed loopable instructions, such as waving or punching. As shown in Table I, **TextOp** maintains high success rates and tracking accuracy across both settings. Robustness to disturbances is further demonstrated by manually perturbing the robot during execution, where it recovers gracefully while preserving ongoing motions, as illustrated in Fig. 5. These experiments collectively show that the system as a whole exhibits stable and reliable performance.

Real-Time Performance. We evaluate the real-time per-

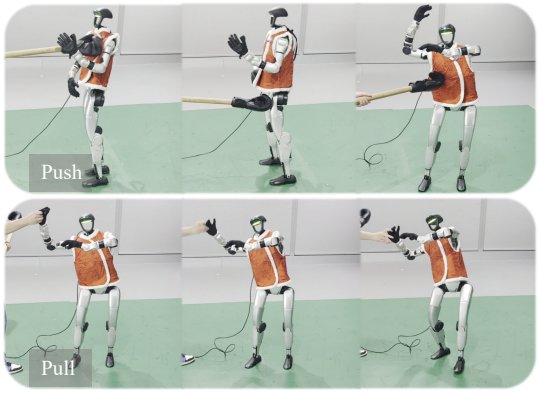

Fig. 5: **Real-time recovery under external perturbations.** The robot dynamically adjusts its actions based on perturbed states to preserve stability and fulfill text-driven commands.

formance of the system to assess its interactive capabilities. Latency is measured at key stages, from language command input to motion generation and low-level tracking. As summarized in Table II, the system exhibits fast and consistent response across all stages. In particular, the user interaction latency, defined as the time from typing a new command to the robot's physical response, is $0.73\,\mathrm{s}$ in average, which is sufficiently low to support responsive real-time interaction.

TABLE II: **Real-time performance of different stages in TextOp.**

Stage	Latency
Text encoding	$7.64_{\pm 2.56}$ ms
Motion generator	$29.63_{\pm 3.56}$ ms
Tracking policy	$2.15_{\pm 0.11}$ ms
User Interaction	$0.73_{\pm 0.10}$ s

C. Offline Evaluation

To quantitatively evaluate the components of TextOp, we conduct offline evaluations, isolating the contributions of the motion generator and the motion tracking policy. All models in these experiments are trained based on public datasets to facilitate reproducibility.

1) Motion Generation Evaluation: To answer **Q2**, we evaluate the motion generator independently, focusing on the effectiveness of the proposed robot-skeleton motion representation.

Baselines: We compare the generator based on our proposed representation against several established baselines:

- **DART+Retarget**: Uses the DART [71] model trained on the SMPL human skeleton, with the generated human motions subsequently retargeted to the robot skeleton via GMR [2].
- **BeyondMimic** [28] Diffusion State: Replicates the state representation of BeyondMimic, which comprises root position, root linear velocity, root rotation, and root angular velocity relative to the initial character-yaw frame, along with local body positions and velocities relative to each character-yaw frame.
- **HumanML3D**-style Representation [15]: Adapts the HumanML3D human-motion representation for the robot, including root angular velocity along the Z-axis, root linear velocity in the XY-plane, root height and rotation, body positions and velocities relative to the root, joint angles, and foot contact states. Global invariance processing is applied to enable autoregressive motion generation.
- **RobotMDM** [44]: Reproduces the motion representation proposed in RobotMDM, including root height, root linear velocity in the XY-plane, root angular velocity along the Z-axis, root rotation, and joint angles and velocities. Global invariance processing is applied to facilitate autoregressive generation.

Experimental Setup: Following DART [71], we evaluate text-conditioned motion generation on the BABEL validation set [40]. Text streams are extracted from the dataset and used as conditions to generate rollout trajectories. Evaluation focuses on two aspects: (i) the complete motion segments corresponding to each text command, and (ii) the transitions between consecutive segments.

Results: Table III summarizes the evaluation results. Our proposed motion representation achieves state-of-the-art performance both in generating motions for individual text commands and in maintaining high-quality transitions between consecutive segments. The observed improvements over alternative representations can be attributed to two key design elements: First, the compact DoF-based motion representation reduces the inherent modeling complexity relative to HumanML3D-style representations. Second, our carefully designed incremental root-state representation improves data efficiency compared with RobotMDM and BeyondMimic. In terms of transition smoothness, our approach is slightly below the DART+Retarget baseline, as the retargeting step introduces smoothness in joint displacements.

2) Motion Tracking Evaluation: To address **Q3**, we evaluate the ability of the motion tracking policy to robustly execute diverse reference motions, with a particular focus on trajectories produced by the motion generator. This evaluation also examines the effect of augmenting the training data with generated motions on tracking robustness and generalization.

Baselines and Ablations: We consider the following configurations:

- **TextOp-M+G (Ours):** Tracker trained on the combined dataset of motion capture and generated motions.
- **TextOp-G (Ours):** Tracker trained exclusively on generator-produced motions.
- **TextOp-M:** Tracker trained exclusively on motion capture data.
- Pretrained general motion tracker policies from prior work: **GMT** [12], **Any2Track** [70], **TWIST2** [66]. These models are trained on different datasets and simulation platforms; for fair comparison, we adopt the officially released pretrained checkpoints.

TABLE III: **Comparison of motion generation with different motion representations.** Our method achieves the best overall performance across most segment- and transition-level metrics. Rightarrow "→" denotes that closer alignment with the dataset reference is better. Bold and underlined values indicate the best and second-best results, respectively, excluding the dataset. All results are reported as mean ± standard deviation over three generator rollout seeds.

Method	Segment						Transition			
	FID ↓	Diversity →	R@1 ↑	R@2 ↑	R@3 ↑	MM-dist ↓	FID ↓	Diversity →	PJ →	AUJ ↓
Dataset	0.000 ± 0.000	9.150 ± 0.158	0.357 ± 0.004	0.556 ± 0.006	0.684 ± 0.005	3.729 ± 0.000	0.000 ± 0.000	8.138 ± 0.013	0.005 ± 0.000	0.000 ± 0.000
DART+Retarget	4.837 ± 0.070	8.012 ± 0.082	0.230 ± 0.007	0.372 ± 0.020	0.486 ± 0.018	5.746 ± 0.026	5.249 ± 0.245	7.939 ± 0.023	**0.007** ± 0.000	**0.123** ± 0.000
BeyondMimic	20.550 ± 0.089	5.930 ± 0.146	0.134 ± 0.009	0.231 ± 0.006	0.320 ± 0.013	7.578 ± 0.014	41.414 ± 0.339	4.209 ± 0.045	0.077 ± 0.003	0.231 ± 0.007
HumanML3D	6.599 ± 0.031	8.236 ± 0.046	0.197 ± 0.005	0.344 ± 0.005	0.453 ± 0.003	6.292 ± 0.023	9.310 ± 0.076	7.676 ± 0.178	0.134 ± 0.014	0.340 ± 0.019
RobotMDM	5.134 ± 0.131	8.210 ± 0.082	0.262 ± 0.006	0.427 ± 0.004	0.556 ± 0.010	5.076 ± 0.031	6.514 ± 0.178	7.602 ± 0.095	0.032 ± 0.001	0.150 ± 0.001
TextOp	**3.072** ± 0.199	**9.220** ± 0.151	**0.300** ± 0.007	**0.481** ± 0.004	**0.607** ± 0.008	**4.272** ± 0.058	**3.238** ± 0.430	**8.226** ± 0.079	0.015 ± 0.001	0.125 ± 0.001

TABLE IV: **Simulation evaluation of the motion tracker on generator-produced motion data.** Incorporating generator-produced motions during training leads to improved performance across both tracking and semantic metrics. Rightarrow "→" denotes that closer alignment with the dataset reference is better. Bold and underlined values indicate the best and second-best results, respectively, excluding the dataset. All results are reported as mean ± standard deviation over three generator rollout seeds.

Method	Tracking Fidelity					Motion Quality					
	Succ ↑	$E_{\text{g-mpjpe}}$ ↓	E_{mpjpe} ↓	E_{acc} ↓	E_{vel} ↓	FID ↓	Diversity →	R@1 ↑	R@2 ↑	R@3 ↑	MM-dist ↓
Dataset	–	–	–	–	–	0.000± 0.000	9.150± 0.158	0.357± 0.004	0.556± 0.006	0.684± 0.005	3.729± 0.000
TextOp-M+G (Ours)	**0.993** ± 0.003	**113.552** ± 1.996	34.665 ± 0.543	1.906 ± 0.014	4.872 ± 0.024	3.454 ± 0.225	8.386 ± 0.113	0.273 ± 0.005	0.453 ± 0.002	0.574 ± 0.007	4.645 ± 0.020
TextOp-G (Ours)	0.991 ± 0.002	121.690 ± 5.950	34.570 ± 0.747	**1.857** ± 0.015	**4.786** ± 0.021	**3.371** ± 0.145	**8.409** ± 0.100	**0.281** ± 0.006	**0.464** ± 0.004	**0.577** ± 0.001	**4.642** ± 0.034
TextOp-M	0.992 ± 0.005	130.532 ± 14.267	**34.526** ± 1.050	1.935 ± 0.007	4.979 ± 0.009	3.615 ± 0.175	8.381 ± 0.101	0.281 ± 0.004	0.452 ± 0.007	0.575 ± 0.006	4.648 ± 0.033
TWIST2	0.922 ± 0.011	847.203 ± 34.405	58.529 ± 1.659	1.999 ± 0.018	7.439 ± 0.081	4.092 ± 0.094	8.187 ± 0.087	0.255 ± 0.007	0.420 ± 0.005	0.543 ± 0.002	5.137 ± 0.058
GMT	0.834 ± 0.018	1252.235 ± 35.132	103.138 ± 1.046	1.934 ± 0.021	8.776 ± 0.085	5.142 ± 0.432	8.010 ± 0.040	0.240 ± 0.012	0.397 ± 0.014	0.503 ± 0.016	5.441 ± 0.125
Any2Track	0.714 ± 0.016	1489.394 ± 51.433	139.178 ± 1.502	2.425 ± 0.019	11.476 ± 0.171	6.135 ± 0.215	7.719 ± 0.013	0.222 ± 0.003	0.367 ± 0.009	0.473 ± 0.006	5.876 ± 0.036

Experimental Setup: All evaluations are conducted in the MuJoCo simulator in a Sim2Sim setting. Reference trajectories are drawn from two sources: (i) **Generator-produced motion:** motions synthesized by our generator, conditioned on text streams the same as Sec. IV-C1, to approximate the deploy-time distribution; and (ii) **SnapMoGen** [24], a publicly available motion capture dataset, which is unseen to all baselines, to evaluate generalization. Only the test split of SnapMoGen is used for evaluation.

TABLE V: **Simulation evaluation of the motion tracker on SnapMoGen data.** Exclusive training on generated motions shows reduced generalization, whereas training on both motion capture and generated data offers a better trade-off between deployment performance and generalization. Bold and underlined values indicate the best and second-best results, respectively.

Method	Succ ↑	$E_{\text{g-mpjpe}}$ ↓	E_{mpjpe} ↓	E_{acc} ↓	E_{vel} ↓
TextOp-M+G (Ours)	0.814	394.705	79.057	**0.892**	**3.859**
TextOp-G (Ours)	0.772	565.374	87.755	0.928	4.310
TextOp-M	**0.823**	**340.145**	**74.761**	0.960	4.020
TWIST2	0.634	1019.826	131.292	1.008	4.629
GMT	0.542	1095.242	150.067	0.968	4.806
Any2Track	0.374	1674.080	203.293	1.253	6.226

Results: Tables IV and V summarize the motion tracking performance on generator-produced trajectories and unseen motion capture data, respectively. On generator-produced motion data, all variants of **TextOp** achieve consistently high success rates and low tracking errors, indicating stable execution under text-conditioned reference motions. Among them, **TextOp**-G performs best overall, while **TextOp**-M+G also improves upon **TextOp**-M across several tracking fidelity and motion quality metrics, suggesting that incorporating generator-produced motions into training is beneficial for aligning the tracker with the deployment-time motion distribution.

On the unseen SnapMoGen dataset, **TextOp**-M achieves the strongest generalization performance, whereas **TextOp**-G exhibits a noticeable performance drop, indicating limited robustness when evaluated beyond the generator's motion distribution. The combined model, **TextOp**-M+G, maintains competitive performance across both settings, balancing robustness on generator-produced trajectories and generalization to unseen motion capture data. These results reflect a trade-off between deployment-oriented robustness and generalization, motivating the use of mixed motion capture and generated data as a practical training strategy.

V. Conclusion

TextOp represents a new control paradigm for humanoid robots, where natural language serves as a continuously revisable control signal rather than a one-shot task specification. This paradigm is realized by combining a high-level motion generator with a low-level universal motion tracker, enabling smooth, whole-body motions that respond instantaneously to language commands. Real-world experiments on the Unitree G1 robot demonstrate the system's ability to execute behaviors ranging from simple gestures to complex skills with high fidelity, robustness, and responsiveness. Offline evaluations

further indicate that carefully designed robot-skeleton motion features improve the quality of generated motions, and incorporating generator-produced motions into the training data effectively aligns the motion tracker with the generator.

Despite these strengths, **TextOp** currently lacks explicit environment perception and interactive physical reasoning. Consequently, the robot cannot adapt its motions to obstacles, objects, or dynamic changes in its surroundings. Overcoming this limitation by integrating environment-aware sensing and interactive planning would enable the robot to respond flexibly to dynamic real-world scenarios. Furthermore, combining such capabilities with high-level reasoning in language models into an agent system could provide a solid foundation for autonomous, instruction-driven behavior, paving the way toward fully autonomous, general-purpose humanoid robots.

Acknowledgments

We are grateful to Zehao Yu, Yiran Wang, Xingyi Wang, and Lei Kuang for their assistance with the real-robot experiments, and to Yiyi Cai, Yang Tang, and Yixuan Pan for their insightful discussions during the preparation of this paper.

This work was supported by the National Natural Science Foundation of China (Grant No. 62306242), the Young Elite Scientists Sponsorship Program by CAST (Grant No. 2024QNRC001), and the Yangfan Project of Shanghai (Grant No. 23YF11462200). The SJTU team was partially supported by the National Natural Science Foundation of China (Grant No. 62322603).

References

[1] Arthur Allshire, Hongsuk Choi, Junyi Zhang, David McAllister, Anthony Zhang, Chung Min Kim, Trevor Darrell, Pieter Abbeel, Jitendra Malik, and Angjoo Kanazawa. Visual imitation enables contextual humanoid control. *arXiv preprint arXiv:2505.03729*, 2025.

[2] Joao Pedro Araujo, Yanjie Ze, Pei Xu, Jiajun Wu, and C Karen Liu. Retargeting matters: General motion retargeting for humanoid motion tracking. *arXiv preprint arXiv:2510.02252*, 2025.

[3] Andrew G Barto. Reinforcement learning: An introduction. by richard's sutton. *SIAM Rev*, 6(2):423, 2021.

[4] Qingwei Ben, Feiyu Jia, Jia Zeng, Junting Dong, Dahua Lin, and Jiangmiao Pang. Homie: Humanoid loco-manipulation with isomorphic exoskeleton cockpit. *arXiv preprint arXiv:2502.13013*, 2025.

[5] Qingwei Ben, Botian Xu, Kailin Li, Feiyu Jia, Wentao Zhang, Jingping Wang, Jingbo Wang, Dahua Lin, and Jiangmiao Pang. Gallant: Voxel grid-based humanoid locomotion and local-navigation across 3d constrained terrains. *arXiv preprint arXiv:2511.14625*, 2025.

[6] Wenzhe Cai, Jiaqi Peng, Yuqiang Yang, Yujian Zhang, Meng Wei, Hanqing Wang, Yilun Chen, Tai Wang, and Jiangmiao Pang. Navdp: Learning sim-to-real navigation diffusion policy with privileged information guidance, 2025. URL https://arxiv.org/abs/2505.08712.

[7] Yiyi Cai, Yuhan Wu, Kunhang Li, You Zhou, Bo Zheng, and Haiyang Liu. Flooddiffusion: Tailored diffusion forcing for streaming motion generation. *arXiv preprint arXiv:2512.03520*, 2025.

[8] Rui Chen, Mingyi Shi, Shaoli Huang, Ping Tan, Taku Komura, and Xuelin Chen. Taming diffusion probabilistic models for character control. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–10, 2024.

[9] Sirui Chen, Zi-ang Cao, Zhengyi Luo, Fernando Castañeda, Chenran Li, Tingwu Wang, Ye Yuan, Linxi Fan, C Karen Liu, Yuke Zhu, et al. Chip: Adaptive compliance for humanoid control through hindsight perturbation. *arXiv preprint arXiv:2512.14689*, 2025.

[10] Sirui Chen, Yufei Ye, Zi-Ang Cao, Jennifer Lew, Pei Xu, and C. Karen Liu. Hand-eye autonomous delivery: Learning humanoid navigation, locomotion and reaching, 2025. URL https://arxiv.org/abs/2508.03068.

[11] Xin Chen, Biao Jiang, Wen Liu, Zilong Huang, Bin Fu, Tao Chen, and Gang Yu. Executing your commands via motion diffusion in latent space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 18000–18010, 2023.

[12] Zixuan Chen, Mazeyu Ji, Xuxin Cheng, Xuanbin Peng, Xue Bin Peng, and Xiaolong Wang. Gmt: General motion tracking for humanoid whole-body control. *arXiv preprint arXiv:2506.14770*, 2025.

[13] Alejandro Escontrela, Justin Kerr, Arthur Allshire, Jonas Frey, Rocky Duan, Carmelo Sferrazza, and Pieter Abbeel. Gaussgym: An open-source real-to-sim framework for learning locomotion from pixels, 2025. URL https://arxiv.org/abs/2510.15352.

[14] Xinyang Gu, Yen-Jen Wang, and Jianyu Chen. Humanoid-gym: Reinforcement learning for humanoid robot with zero-shot sim2real transfer. *arXiv preprint arXiv:2404.05695*, 2024.

[15] Chuan Guo, Shihao Zou, Xinxin Zuo, Sen Wang, Wei Ji, Xingyu Li, and Li Cheng. Generating diverse and natural 3d human motions from text. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5152–5161, June 2022.

[16] Tuomas Haarnoja, Ben Moran, Guy Lever, Sandy H Huang, Dhruva Tirumala, Jan Humplik, Markus Wulfmeier, Saran Tunyasuvunakool, Noah Y Siegel, Roland Hafner, et al. Learning agile soccer skills for a bipedal robot with deep reinforcement learning. *Science Robotics*, 9(89):eadi8022, 2024.

[17] Jinrui Han, Weiji Xie, Jiakun Zheng, Jiyuan Shi, Weinan Zhang, Ting Xiao, and Chenjia Bai. Kungfubot2: Learning versatile motion skills for humanoid whole-body control. *arXiv preprint arXiv:2509.16638*, 2025.

[18] Tairan He, Zhengyi Luo, Xialin He, Wenli Xiao, Chong Zhang, Weinan Zhang, Kris Kitani, Changliu Liu, and Guanya Shi. Omnih2o: Universal and dexterous human-to-humanoid whole-body teleoperation and learning. *arXiv preprint arXiv:2406.08858*, 2024.

[19] Tairan He, Zhengyi Luo, Wenli Xiao, Chong Zhang, Kris

Kitani, Changliu Liu, and Guanya Shi. Learning human-to-humanoid real-time whole-body teleoperation. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8944–8951. IEEE, 2024.

[20] Tairan He, Zi Wang, Haoru Xue, Qingwei Ben, Zhengyi Luo, Wenli Xiao, Ye Yuan, Xingye Da, Fernando Castañeda, Shankar Sastry, et al. Viral: Visual sim-to-real at scale for humanoid loco-manipulation. *arXiv preprint arXiv:2511.15200*, 2025.

[21] Xialin He, Runpei Dong, Zixuan Chen, and Saurabh Gupta. Learning getting-up policies for real-world humanoid robots. *arXiv preprint arXiv:2502.12152*, 2025.

[22] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[23] Tao Huang, Junli Ren, Huayi Wang, Zirui Wang, Qingwei Ben, Muning Wen, Xiao Chen, Jianan Li, and Jiangmiao Pang. Learning humanoid standing-up control across diverse postures. *arXiv preprint arXiv:2502.08378*, 2025.

[24] Inwoo Hwang, Jian Wang, Bing Zhou, et al. Snapmogen: Human motion generation from expressive texts. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[25] Biao Jiang, Xin Chen, Wen Liu, Jingyi Yu, Gang Yu, and Tao Chen. Motiongpt: Human motion as a foreign language. *Advances in Neural Information Processing Systems*, 36:20067–20079, 2023.

[26] Yitang Li, Zhengyi Luo, Tonghe Zhang, Cunxi Dai, Anssi Kanervisto, Andrea Tirinzoni, Haoyang Weng, Kris Kitani, Mateusz Guzek, Ahmed Touati, et al. Bfm-zero: A promptable behavioral foundation model for humanoid control using unsupervised reinforcement learning. *arXiv preprint arXiv:2511.04131*, 2025.

[27] Zhe Li, Cheng Chi, Yangyang Wei, Boan Zhu, Yibo Peng, Tao Huang, Pengwei Wang, Zhongyuan Wang, Shanghang Zhang, and Chang Xu. From language to locomotion: Retargeting-free humanoid control via motion latent guidance. *arXiv preprint arXiv:2510.14952*, 2025.

[28] Qiayuan Liao, Takara E Truong, Xiaoyu Huang, Yuman Gao, Guy Tevet, Koushil Sreenath, and C Karen Liu. Beyondmimic: From motion tracking to versatile humanoid control via guided diffusion. *arXiv preprint arXiv:2508.08241*, 2025.

[29] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. Smpl: A skinned multi-person linear model. In *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*, pages 851–866. 2023.

[30] Chenhao Lu, Xuxin Cheng, Jialong Li, Shiqi Yang, Mazeyu Ji, Chengjing Yuan, Ge Yang, Sha Yi, and Xiaolong Wang. Mobile-television: Predictive motion priors for humanoid whole-body control. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5364–5371. IEEE, 2025.

[31] Zhengyi Luo, Jinkun Cao, Kris Kitani, Weipeng Xu, et al. Perpetual humanoid control for real-time simulated avatars. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10895–10904, 2023.

[32] Zhengyi Luo, Ye Yuan, Tingwu Wang, Chenran Li, Sirui Chen, Fernando Castañeda, Zi-Ang Cao, Jiefeng Li, David Minor, Qingwei Ben, et al. Sonic: Supersizing motion tracking for natural humanoid whole-body control. *arXiv preprint arXiv:2511.07820*, 2025.

[33] Naureen Mahmood, Nima Ghorbani, Nikolaus F Troje, Gerard Pons-Moll, and Michael J Black. Amass: Archive of motion capture as surface shapes. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5442–5451, 2019.

[34] Mayank Mittal, Pascal Roth, James Tigue, Antoine Richard, Octi Zhang, Peter Du, Antonio Serrano-Muñoz, Xinjie Yao, René Zurbrügg, Nikita Rudin, Lukasz Wawrzyniak, Milad Rakhsha, Alain Denzler, Eric Heiden, Ales Borovicka, Ossama Ahmed, Iretiayo Akinola, Abrar Anwar, Mark T. Carlson, Ji Yuan Feng, Animesh Garg, Renato Gasoto, Lionel Gulich, Yijie Guo, M. Gussert, Alex Hansen, Mihir Kulkarni, Chenran Li, Wei Liu, Viktor Makoviychuk, Grzegorz Malczyk, Hammad Mazhar, Masoud Moghani, Adithyavairavan Murali, Michael Noseworthy, Alexander Poddubny, Nathan Ratliff, Welf Rehberg, Clemens Schwarke, Ritvik Singh, James Latham Smith, Bingjie Tang, Ruchik Thaker, Matthew Trepte, Karl Van Wyk, Fangzhou Yu, Alex Millane, Vikram Ramasamy, Remo Steiner, Sangeeta Subramanian, Clemens Volk, CY Chen, Neel Jawale, Ashwin Varghese Kuruttukulam, Michael A. Lin, Ajay Mandlekar, Karsten Patzwaldt, John Welsh, Huihua Zhao, Fatima Anes, Jean-Francois Lafleche, Nicolas Moënne-Loccoz, Soowan Park, Rob Stepinski, Dirk Van Gelder, Chris Amevor, Jan Carius, Jumyung Chang, Anka He Chen, Pablo de Heras Ciechomski, Gilles Daviet, Mohammad Mohajerani, Julia von Muralt, Viktor Reutskyy, Michael Sauter, Simon Schirm, Eric L. Shi, Pierre Terdiman, Kenny Vilella, Tobias Widmer, Gordon Yeoman, Tiffany Chen, Sergey Grizan, Cathy Li, Lotus Li, Connor Smith, Rafael Wiltz, Kostas Alexis, Yan Chang, David Chu, Linxi "Jim" Fan, Farbod Farshidian, Ankur Handa, Spencer Huang, Marco Hutter, Yashraj Narang, Soha Pouya, Shiwei Sheng, Yuke Zhu, Miles Macklin, Adam Moravanszky, Philipp Reist, Yunrong Guo, David Hoeller, and Gavriel State. Isaac lab: A gpu-accelerated simulation framework for multi-modal robot learning. *arXiv preprint arXiv:2511.04831*, 2025. URL https://arxiv.org/abs/2511.04831.

[35] Noboru Myers, Obin Kwon, Sankalp Yamsani, and Joohyung Kim. Child (controller for humanoid imitation and live demonstration): A whole-body humanoid teleoperation system. In *2025 IEEE-RAS 24th International Conference on Humanoid Robots (Humanoids)*, pages 1–6, 2025. doi: 10.1109/Humanoids65713.2025.11203119.

[36] NVIDIA. NVIDIA TensorRT, 2026. URL https://github.

com/NVIDIA/TensorRT. Accessed: 2026-01-25.

[37] ONNX Runtime developers. ONNX Runtime, November 2018. URL https://github.com/microsoft/onnxruntime.

[38] Yixuan Pan, Ruoyi Qiao, Li Chen, Kashyap Chitta, Liang Pan, Haoguang Mai, Qingwen Bu, Hao Zhao, Cunyuan Zheng, Ping Luo, et al. Agility meets stability: Versatile humanoid control with heterogeneous data. *arXiv preprint arXiv:2511.17373*, 2025.

[39] Mathis Petrovich, Michael J Black, and Gül Varol. Tmr: Text-to-motion retrieval using contrastive 3d human motion synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9488–9497, 2023.

[40] Abhinanda R. Punnakkal, Arjun Chandrasekaran, Nikos Athanasiou, Alejandra Quiros-Ramirez, and Michael J. Black. BABEL: Bodies, action and behavior with english labels. In *Proceedings IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 722–731, June 2021.

[41] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[42] Junli Ren, Junfeng Long, Tao Huang, Huayi Wang, Zirui Wang, Feiyu Jia, Wentao Zhang, Jingbo Wang, Ping Luo, and Jiangmiao Pang. Humanoid goalkeeper: Learning from position conditioned task-motion constraints. *arXiv preprint arXiv:2510.18002*, 2025.

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

[44] Agon Serifi, Ruben Grandia, Espen Knoop, Markus Gross, and Moritz Bächer. Robot motion diffusion model: Motion generation for robotic characters. In *SIGGRAPH asia 2024 conference papers*, pages 1–9, 2024.

[45] Yiyang Shao, Xiaoyu Huang, Bike Zhang, Qiayuan Liao, Yuman Gao, Yufeng Chi, Zhongyu Li, Sophia Shao, and Koushil Sreenath. Langwbc: Language-directed humanoid whole-body control via end-to-end learning. *arXiv preprint arXiv:2504.21738*, 2025.

[46] Zehong Shen, Huaijin Pi, Yan Xia, Zhi Cen, Sida Peng, Zechen Hu, Hujun Bao, Ruizhen Hu, and Xiaowei Zhou. World-grounded human motion recovery via gravity-view coordinates. In *SIGGRAPH Asia 2024 Conference Papers*, pages 1–11, 2024.

[47] Yi Shi, Jingbo Wang, Xuekun Jiang, Bingkun Lin, Bo Dai, and Xue Bin Peng. Interactive character control with auto-regressive motion diffusion models. *ACM Transactions on Graphics (TOG)*, 43(4):1–14, 2024.

[48] Zhi Su, Bike Zhang, Nima Rahmanian, Yuman Gao, Qiayuan Liao, Caitlin Regan, Koushil Sreenath, and S Shankar Sastry. Hitter: A humanoid table tennis robot via hierarchical planning and learning. *arXiv preprint arXiv:2508.21043*, 2025.

[49] Guy Tevet, Sigal Raab, Brian Gordon, Yonatan Shafir, Daniel Cohen-Or, and Amit H Bermano. Human motion diffusion model. *arXiv preprint arXiv:2209.14916*, 2022.

[50] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012. doi: 10.1109/IROS.2012.6386109.

[51] Unitree Robotics. Unitree G1: Humanoid agent AI avatar. https://www.unitree.com/g1, 2024. Accessed: 2026-01-25.

[52] Dewei Wang, Xinmiao Wang, Xinzhe Liu, Jiyuan Shi, Yingnan Zhao, Chenjia Bai, and Xuelong Li. More: Mixture of residual experts for humanoid lifelike gaits learning on complex terrains, 2025. URL https://arxiv.org/abs/2506.08840.

[53] Yuan Wang, Di Huang, Yaqi Zhang, Wanli Ouyang, Jile Jiao, Xuetao Feng, Yan Zhou, Pengfei Wan, Shixiang Tang, and Dan Xu. Motiongpt-2: A general-purpose motion-language model for motion generation and understanding. *arXiv preprint arXiv:2410.21747*, 2024.

[54] Yufu Wang, Yu Sun, Priyanka Patel, Kostas Daniilidis, Michael J Black, and Muhammed Kocabas. Prompthmr: Promptable human mesh recovery. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 1148–1159, 2025.

[55] Yuxuan Wang, Haobin Jiang, Shiqing Yao, Ziluo Ding, and Zongqing Lu. Sentinel: A fully end-to-end language-action model for humanoid whole body control. *arXiv preprint arXiv:2511.19236*, 2025.

[56] Lixing Xiao, Shunlin Lu, Huaijin Pi, Ke Fan, Liang Pan, Yueer Zhou, Ziyong Feng, Xiaowei Zhou, Sida Peng, and Jingbo Wang. Motionstreamer: Streaming motion generation via diffusion-based autoregressive model in causal latent space. *arXiv preprint arXiv:2503.15451*, 2025.

[57] Weiji Xie, Chenjia Bai, Jiyuan Shi, Junkai Yang, Yunfei Ge, Weinan Zhang, and Xuelong Li. Humanoid whole-body locomotion on narrow terrain via dynamic balance and reinforcement learning, 2025. URL https://arxiv.org/abs/2502.17219.

[58] Weiji Xie, Jinrui Han, Jiakun Zheng, Huanyu Li, Xinzhe Liu, Jiyuan Shi, Weinan Zhang, Chenjia Bai, and Xuelong Li. Kungfubot: Physics-based humanoid whole-body control for learning highly-dynamic skills. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=LCPoXt0pzm.

[59] Haoru Xue, Tairan He, Zi Wang, Qingwei Ben, Wenli Xiao, Zhengyi Luo, Xingye Da, Fernando Castañeda, Guanya Shi, Shankar Sastry, et al. Opening the sim-to-real door for humanoid pixel-to-action policy transfer. *arXiv preprint arXiv:2512.01061*, 2025.

[60] Haoru Xue, Xiaoyu Huang, Dantong Niu, Qiayuan Liao, Thomas Kragerud, Jan Tommy Gravdahl, Xue Bin Peng,

Guanya Shi, Trevor Darrell, Koushil Sreenath, et al. Leverb: Humanoid whole-body control with latent vision-language instruction. *arXiv preprint arXiv:2506.13751*, 2025.

[61] Yufei Xue, Wentao Dong, Minghuan Liu, Weinan Zhang, and Jiangmiao Pang. A unified and general humanoid whole-body controller for fine-grained locomotion. *arXiv e-prints*, pages arXiv–2502, 2025.

[62] Lujie Yang, Xiaoyu Huang, Zhen Wu, Angjoo Kanazawa, Pieter Abbeel, Carmelo Sferrazza, C Karen Liu, Rocky Duan, and Guanya Shi. Omniretarget: Interaction-preserving data generation for humanoid whole-body loco-manipulation and scene interaction. *arXiv preprint arXiv:2509.26633*, 2025.

[63] Shiqi Yang. Ace: A cross-platform visual-exoskeleton system for low-cost dexterous teleoperation. Master's thesis, University of California, San Diego, 2025.

[64] Kangning Yin, Weishuai Zeng, Ke Fan, Minyue Dai, Zirui Wang, Qiang Zhang, Zheng Tian, Jingbo Wang, Jiangmiao Pang, and Weinan Zhang. Unitracker: Learning universal whole-body motion tracker for humanoid robots. *arXiv preprint arXiv:2507.07356*, 2025.

[65] Yanjie Ze, Zixuan Chen, João Pedro Araújo, Zi ang Cao, Xue Bin Peng, Jiajun Wu, and C. Karen Liu. Twist: Teleoperated whole-body imitation system, 2025. URL https://arxiv.org/abs/2505.02833.

[66] Yanjie Ze, Siheng Zhao, Weizhuo Wang, Angjoo Kanazawa, Rocky Duan, Pieter Abbeel, Guanya Shi, Jiajun Wu, and C Karen Liu. Twist2: Scalable, portable, and holistic humanoid data collection system. *arXiv preprint arXiv:2511.02832*, 2025.

[67] Jianrong Zhang, Yangsong Zhang, Xiaodong Cun, Yong Zhang, Hongwei Zhao, Hongtao Lu, Xi Shen, and Ying Shan. Generating human motion from textual descriptions with discrete representations. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14730–14740, 2023.

[68] Mingyuan Zhang, Zhongang Cai, Liang Pan, Fangzhou Hong, Xinying Guo, Lei Yang, and Ziwei Liu. Motiondiffuse: Text-driven human motion generation with diffusion model. *IEEE transactions on pattern analysis and machine intelligence*, 46(6):4115–4128, 2024.

[69] Yan Zhang, Yao Feng, Alpár Cseke, Nitin Saini, Nathan Bajandas, Nicolas Heron, and Michael J. Black. PRIMAL: physically reactive and interactive motor model for avatar learning. October 2025.

[70] Zhikai Zhang, Jun Guo, Chao Chen, Jilong Wang, Chenghuai Lin, Yunrui Lian, Han Xue, Zhenrong Wang, Maoqi Liu, Jiangran Lyu, et al. Track any motions under any disturbances. *arXiv preprint arXiv:2509.13833*, 2025.

[71] Kaifeng Zhao, Gen Li, and Siyu Tang. Dartcontrol: A diffusion-based autoregressive motion model for real-time text-driven motion control. *arXiv preprint arXiv:2410.05260*, 2024.

[72] Bingfan Zhu, Biao Jiang, Sunyi Wang, Shixiang Tang, Tao Chen, Linjie Luo, Youyi Zheng, and Xin Chen. Motiongpt3: Human motion as a second modality. *arXiv preprint arXiv:2506.24086*, 2025.

[73] Shaoting Zhu, Ziwen Zhuang, Mengjie Zhao, Kun-Ying Lee, and Hang Zhao. Hiking in the wild: A scalable perceptive parkour framework for humanoids, 2026. URL https://arxiv.org/abs/2601.07718.

[74] Ziwen Zhuang, Shaoting Zhu, Mengjie Zhao, and Hang Zhao. Deep whole-body parkour. *arXiv preprint arXiv:2601.07701*, 2026.

APPENDIX A
ADDITIONAL IMPLEMENTATION DETAILS

This appendix provides supplementary implementation details that are omitted from the main text due to space constraints.

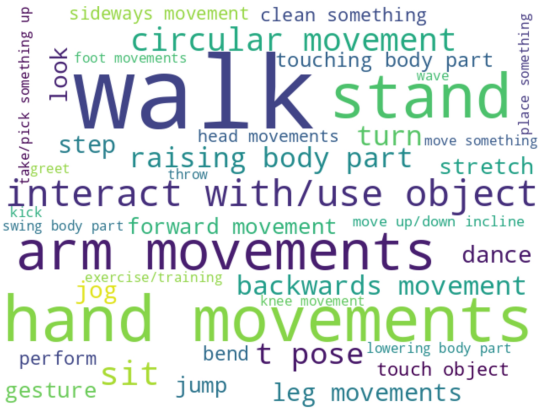

Fig. 6: **Word cloud of text annotations in the BABEL dataset.**

A. Data Preprocessing and Dataset Construction

We describe the complete data preprocessing pipeline and the construction of datasets used for training the motion generator and motion tracker. All datasets share a common preprocessing procedure, followed by task-specific preparation for motion generation and motion tracking.

Shared Preprocessing Pipeline.

1) All motion sequences are first retargeted from the SMPL [29] human skeleton to the 29-DoF G1 humanoid robot using GMR [2].
2) The retargeted motions are then temporally resampled from 30 Hz to a unified frequency of 50 Hz.
3) Following PBHC [58], feet contact indicators are extracted from the retargeted motions using a threshold-based rule:

$$c_t^{\text{left}} = \mathbb{I}[\|\boldsymbol{p}_{t+1}^{\text{l-ankle}} - \boldsymbol{p}_t^{\text{l-ankle}}\|_2^2 < \epsilon_{\text{vel}}] \cdot \mathbb{I}[p_{t,z}^{\text{l-ankle}} < \epsilon_{\text{height}}], \tag{1}$$

where c_t^{left} represents the contact mask of the left foot and $\boldsymbol{p}_t^{\text{l-ankle}}$ denotes the position of the left ankle joint at time t. $\epsilon_{\text{vel}} = 0.002$ and $\epsilon_{\text{height}} = 0.2$ are empirically chosen thresholds. Similarly for the right foot.
4) Filtering: (1) Based on the unfiltered dataset, we train a universal motion tracking policy with privileged information, without domain randomization, and with relaxed termination conditions by removing End-Effector Height Deviation and Pelvis Height Deviation. The policy is trained for 40,000 steps, and we record the failure rate of each motion using a sliding average method. (2) We remove those motions with a tracking failure probability greater than 0.05, using the relaxed termination conditions. (3) The remaining motions are re-evaluated using standard termination conditions, and any motion that fails is removed.

Dataset Construction for Motion Generation. For the AMASS dataset:

1) Motion segments are paired with textual annotations from the BABEL dataset [40] via filename-based indexing and temporal alignment. The distribution of text annotations is shown in Fig. 6.
2) Mirror augmentation is applied to both motion sequences and their corresponding textual labels, effectively doubling the dataset size.

For the proprietary dataset, which mainly contains complex long-horizon motions, we manually assign a unique label to each sequence. To distinguish such long-form descriptions from segment-level annotations, we wrap these labels with "$\langle\cdot\rangle$" markers, e.g., multiple hip hop dance segments of several tens of seconds will be named as "$\langle$hiphop 1$\rangle$", "$\langle$hiphop 2$\rangle$", and so on. Since this labeling is coarse-grained, we found it allows the model to associate each label with a distinct motion style and generate stylistically consistent motions, rather than reproducing an entire reference sequence.

Dataset Construction for Motion Tracking.

1) For data undergoing the **Shared Preprocessing Pipeline**, we further segment motions into short clips with a maximum duration of 40 seconds (i.e., 2000 frames). Specifically, each motion sequence is partitioned into multiple clips whose lengths are constrained to lie within $[100, 2000]$ frames, and adjacent clips are allowed to temporally overlap, with the overlap length constrained to lie within $[50, 200]$ frames.
2) Data Augmentation via Motion Generation: We sample text annotations from the BABEL training set to construct 20-second text streams. For each motion sequence, all associated text annotations are extracted, and the generation length assigned to each annotation is proportional to its temporal duration within the original motion. Using these constructed text streams, we then generate synthetic motions with our trained motion generation model.

B. Motion Generation

Robot Skeleton Motion Representation. We define a pair of deterministic forward and inverse algorithms that transform between a raw robot motion trajectory and a per-frame, local incremental motion representation. The forward algorithm maps a raw motion sequence $\{p_t, R_t, q_t, c_t\}_{t=0}^{T}$ to a feature sequence $\{f_t\}_{t=0}^{T-1}$ and an initial pose (p_0, R_0). The inverse algorithm reconstructs the motion sequence from the feature sequence given the initial pose. Both transformations are invariant to global translation and yaw and are exactly invertible up to numerical precision. In these algorithms, $p_t \in \mathbb{R}^3$ denotes the root translation, R_t the root rotation, q_t the joint positions, and c_t the feet contact indicators. The motion feature f_t consists of: ϕ_t representing the roll and pitch angles as $(\sin\theta, \cos\theta - 1)$, $\Delta\psi_t$ the yaw increment, $\Delta p_t^{\text{local}}$

Algorithm 1 Forward Transformation: Raw Motion $\rightarrow$ Motion Features

Require: Raw Motion $\{p_t, R_t, q_t, c_t\}_{t=0}^{T}$
Ensure: Motion Features $\{f_t\}_{t=0}^{T-1}$, Initial Pose (p_0, R_0)

$(p_0, R_0) \leftarrow (p_0, R_0)$
for $t = 0$ **to** $T-1$ **do**
  $(\mathrm{roll}_t, \mathrm{pitch}_t, \mathrm{yaw}_t) \leftarrow \textsc{QuatToEuler}(R_t)$
  $(\_, \_, \mathrm{yaw}_{t+1}) \leftarrow \textsc{QuatToEuler}(R_{t+1})$
  $s_r \leftarrow \sin(\mathrm{roll}_t), \quad c_r \leftarrow \cos(\mathrm{roll}_t) - 1$
  $s_p \leftarrow \sin(\mathrm{pitch}_t), \quad c_p \leftarrow \cos(\mathrm{pitch}_t) - 1$
  $\phi_t \leftarrow [s_r, c_r, s_p, c_p]$
  $\Delta\psi_t \leftarrow \mathrm{yaw}_{t+1} - \mathrm{yaw}_t$
  $\Delta p_t^{\text{local}} \leftarrow R_z(\mathrm{yaw}_t)^\top (p_{t+1} - p_t)$
  $\Delta q_t \leftarrow q_{t+1} - q_t$
  $h_t \leftarrow (p_t)_z$
  $f_t \leftarrow [\phi_t, \Delta\psi_t, c_t, \Delta p_t^{\text{local}}, h_t, q_t, \Delta q_t]$
end for

Algorithm 2 Inverse Transformation: Motion Features $\rightarrow$ Raw Motion

Require: Motion features $\{f_t\}_{t=0}^{T-1}$, Initial pose $(p_{\text{init}}, R_{\text{init}})$
Ensure: Reconstructed raw motion $\{p_t, R_t, q_t, c_t\}_{t=0}^{T-1}$

$(\_, \_, \mathrm{yaw}_0) \leftarrow \textsc{QuatToEuler}(R_{\text{init}})$
$p'_0 \leftarrow p_{\text{init}}$
for $t = 0$ **to** $T-1$ **do**
  $[\phi_t, \Delta\psi_t, c_t, \Delta p_t^{\text{local}}, h_t, q_t, \Delta q_t] \leftarrow f_t$
  $(s_r, c_r, s_p, c_p) \leftarrow \phi_t$
  $\mathrm{roll}_t \leftarrow \arctan 2(s_r, c_r + 1)$
  $\mathrm{pitch}_t \leftarrow \arctan 2(s_p, c_p + 1)$
  $R_t \leftarrow \textsc{EulerToQuat}(\mathrm{roll}_t, \mathrm{pitch}_t, \mathrm{yaw}_t)$
  $p_t \leftarrow [(p'_t)_x, (p'_t)_y, h_t]^\top$
  if $t < T-1$ **then**
    $\mathrm{yaw}_{t+1} \leftarrow \mathrm{yaw}_t + \Delta\psi_t$
    $\Delta p_t \leftarrow R_z(\mathrm{yaw}_t)\Delta p_t^{\text{local}}$
    $p'_{t+1} \leftarrow p'_t + \Delta p_t$
  end if
end for

the local translation increment, h_t the absolute root height, and Δq_t the joint velocities. In the inverse transformation, p'_t serves as an intermediate integration variable for translation, while p_t represents the final reconstructed position with its z-component assigned by h_t.

Architecture. We design the VAE and LDM modules following the DART [71]. The VAE adopts a transformer-based structure, where both the encoder and decoder are implemented as stacks of transformer encoder layers augmented with skip connections. In the encoding stage, the historical and future motion sequences are concatenated, linearly projected into a high-dimensional space, and prepended with a set of learnable distribution parameters. After incorporating positional encodings, the combined sequences are processed by the transformer encoder. The output corresponding to the distribution parameters is used to parameterize a Gaussian distribution in the latent space, from which latent variables are sampled. During decoding, the sampled latent variables, together with the historical motion sequences, are used to reconstruct the future motion sequences through another transformer stack and a final linear projection layer.

The LDM is implemented as a transformer encoder that operates in the latent space. It takes four types of inputs: noisy latent variables, the diffusion timesteps, historical motion sequences, and text embeddings. Each input modality is independently embedded into a shared hidden dimension. To support classifier-free guidance, the text embeddings are randomly masked during training with a fixed probability of 0.1. All embedded representations are concatenated into a sequence of tokens, enhanced with sinusoidal positional encodings, and fed into multiple transformer encoder layers. The output corresponding to the noisy latent token is then mapped back to the noise dimension, predicting the cleaned latent variables for the diffusion process. The detailed Motion Generator hyperparameters are shown in Table VI.

Training Details.

When training the VAE and LDM, we organize motion data into fixed-length motion primitives to ensure temporal continuity and stable learning.

Specifically, each motion primitive $\mathbf{P}^i$ consists of two parts: (1) T_{history} frames of historical motion, and (2) T_{future} frames of future motion to be predicted. The historical part of $\mathbf{P}^i$ is the same as the last T_{history} frames of the preceding primitive $\mathbf{P}^{i-1}$, which enforces continuity between adjacent primitives.

During data loading, we sample N_{prim} consecutive primitives from a full motion sequence to form one training sample. This results in a motion clip of length $T_{\text{history}} + N_{\text{prim}} \cdot T_{\text{future}}$ frames. The model is therefore trained on an ordered sequence of primitives that together represent a coherent motion segment.

To reduce distribution mismatch between training and inference, we further adopt a self-rollout curriculum. As training progresses, with a linearly increasing probability, the history frames of a primitive are replaced by the last T_{history} frames generated by the model from the previous step, instead of being directly sampled from the dataset. The replacement probability is capped at 0.8 to maintain training stability and avoid excessive error accumulation.

For the LDM, we adopt the DDPM framework. The diffusion process gradually adds Gaussian noise to the latent motion latent z_0 over K timesteps. The forward noising process is defined as a Markov chain:

$$\begin{aligned} q(z_{1:K}|z_0) &= \prod_{k=1}^{K} q(z_k|z_{k-1}), \\ q(z_k|z_{k-1}) &= \mathcal{N}(z_k; \sqrt{1-\beta_k} z_{k-1}, \beta_k I). \end{aligned} \tag{2}$$

Here $\beta_k \in (0,1)$ is the noise schedule determined by a cosine scheduler. Let $\alpha_k = 1 - \beta_k$ and $\bar{\alpha}_k = \prod_{s=1}^{k} \alpha_s$. Then the noisy latent z_k at an arbitrary timestep k can be directly sampled from z_0 via:

$$z_k = \sqrt{\bar{\alpha}_k} z_0 + \sqrt{1-\bar{\alpha}_k}\epsilon, \quad \epsilon \sim \mathcal{N}(0, I). \tag{3}$$

During training, we randomly sample a timestep $k \sim \mathcal{U}\{1, K\}$ and compute z_k from z_0 using Eq. 3. The LDM model F_θ then takes z_k, the timestep k, the historical motion features $f_{t-T_{\text{history}}:t-1}$, and the text embedding e_t as inputs, and predicts the original z_0.

Losses. The VAE is optimized with a combined loss function consisting of a reconstruction term, a lightweight KL divergence term, and a comprehensive geometric loss that ensures physical plausibility. The overall objective is defined as follows:

$$\mathcal{L}_{\text{VAE}} = \lambda_{\text{rec}} \cdot \mathcal{L}_{\text{rec}} + \lambda_{\text{KL}} \cdot \mathcal{L}_{\text{KL}} + \mathcal{L}_{\text{geo}}, \tag{4}$$

where $\mathcal{L}_{\text{rec}}$ is the reconstruction loss. We adopt the Huber loss (smooth L1 loss) as the reconstruction objective to measure the discrepancy between the predicted future motion features $\hat{f}$ and the ground truth f:

$$\mathcal{L}_{\text{rec}} = \text{Huber}(\hat{f}_{t:t+T_{\text{future}}-1}, f_{t:t+T_{\text{future}}-1}) \tag{5}$$

This encourages the decoder to accurately reconstruct the motion sequences. To maintain a well-structured latent space without sacrificing expressiveness, we impose a KL penalty toward a standard Gaussian distribution:

$$\mathcal{L}_{\text{KL}} = \text{KL}\left(E_\phi(z \mid f_{t-T_{\text{history}}:t+T_{\text{future}}-1})) \parallel \mathcal{N}(0, I)\right) \tag{6}$$

To enhance physical coherence and motion quality, we introduce a multi-term geometric loss evaluated in forward-kinematics space:

$$\begin{aligned}\mathcal{L}_{\text{geo}} =& \lambda_{\text{trans}} \cdot \mathcal{L}_{\text{body\_trans}} + \lambda_{\text{rot}} \cdot \mathcal{L}_{\text{body\_rot}} \\ &+ \lambda_{\text{dof}} \cdot \mathcal{L}_{\text{dof}} + \lambda_{\text{vel}} \cdot \mathcal{L}_{\text{dof\_vel}} \\ &+ \lambda_{\text{contact}} \cdot \mathcal{L}_{\text{contact}},\end{aligned} \tag{7}$$

where $\mathcal{L}_{\text{body\_trans}}$ and $\mathcal{L}_{\text{body\_rot}}$ penalize errors in body global translation and rotation. $\mathcal{L}_{\text{dof}}$ and $\mathcal{L}_{\text{dof\_vel}}$ penalize errors in dof angles and angular velocities. $\mathcal{L}_{\text{contact}}$ focuses on foot positions during contact phases to improve grounding. Each geometric term is computed using the Huber loss between predicted and ground-truth kinematic quantities, which ensemble significantly improves motion smoothness and physical realism.

The LDM is trained with the following losses:

$$L_{\text{LDM}} = \lambda_{\text{simple}} \cdot L_{\text{simple}} + \lambda_{\text{rec}} \cdot L_{\text{rec}} + L_{\text{geo}}, \tag{8}$$

where the definitions of L_{rec} and L_{geo} are consistent with the losses used for training the VAE and L_{simple} is defined as

$$\mathcal{L}_{\text{simple}} = \text{Huber}(F_\theta(z_k, k, f_{t-T_{\text{history}}:t-1}, e_t), z_0) \tag{9}$$

Inference Details. During inference, we first sample a Gaussian noise $z_K \sim \mathcal{N}(0, I)$, and then iteratively denoise for timestep $k = K, K-1, \ldots, 1$ to obtain the clean latent z_0. At each denoising step k, classifier-free guidance is applied to make motion latent predictions:

$$\begin{aligned}\hat{z}_0 =& F_\theta(z_k, k, f_{t-T_{\text{history}}:t-1}, \varnothing) \\ &+ \sigma_{\text{CFG}} \cdot \big(F_\theta(z_k, k, f_{t-T_{\text{history}}:t-1}, e_t) \\ &- F_\theta(z_k, k, f_{t-T_{\text{history}}:t-1}, \varnothing)\big),\end{aligned} \tag{10}$$

where σ_{CFG} is the guidance scale. Then, the predicted $\hat{z}_0$ is used to compute the estimated noise ϵ_θ:

$$\epsilon_\theta = \frac{z_k - \sqrt{\bar{\alpha}_k}\hat{z}_0}{\sqrt{1-\bar{\alpha}_k}}. \tag{11}$$

Finally, we sample z_{k-1} from the reverse transition distribution:

$$z_{k-1} = \frac{1}{\sqrt{\alpha_k}}\left(z_k - \frac{1-\alpha_k}{\sqrt{1-\bar{\alpha}_k}}\epsilon_\theta\right) + \sigma_k \epsilon, \tag{12}$$

where $\sigma_k^2 = \beta_k$. The final clean latent z_0 is then passed through the VAE decoder to produce the future motion features.

TABLE VI: **Hyperparameters for Motion Generator.**

Hyperparameter	**Value**
History Length T_{history}	2
Future Length T_{future}	8
Primitive Number N_{prim}	4
Batch Size	512
Learning Rate	$1e-4$
Feature Dim d_{feat}	69
λ_{rec}	1.0
λ_{KL}	$1e-4$
λ_{simple}	1.0
λ_{trans}	0.05
λ_{rot}	$1e-2$
λ_{dof}	0.03
λ_{vel}	$1e-5$
λ_{contact}	0.01
VAE Training	
Hidden Dim	512
FFN Dim	1024
Latent Dim	128
Transformer Layers Number	9
Heads Number	4
Maximum Rollout Probability	1.0
LDM Training	
Noise Dim	128
Hidden Dim	512
FFN Dim	1024
Transformer Layers Number	8
Heads Number	4
Diffusion Steps n_{denoise}	5
Max Rollout Probability	0.8

C. Motion Tracking Policy

The training of the motion tracking policy adopts an asymmetric actor-critic style, where the critic network takes additional privileged information as input. The observation space, reward functions, termination conditions, and domain randomization setting are listed in Table VII, Table VIII, Table IX, and Table X, respectively.

APPENDIX B
ADDITIONAL EXPERIMENTAL DETAILS

This appendix provides supplementary descriptions of the experimental setup, evaluation protocols, and additional results.

TABLE VII: **Observation space for Motion Tracker.** The **Type** column distinguishes between policy inputs (Base) and critic-only inputs (Privileged). The 14 key bodies include: Pelvis, Hips (2), Knees (2), Ankles (2), Torso, Shoulders (2), Elbows (2), and Wrists (2). The total dimensionality is 431 for Base observations and 557 when including Privileged inputs.

Term	Description	Dim.	Type
Reference Motion	Target joint positions and velocities over a 5-step future horizon	$290 = 2 \times 29 \times 5$	Base
Reference Pelvis Position	Relative pelvis positions in the base frame over a 5-step horizon	$15 = 3 \times 5$	Base
Reference Pelvis Orientation	Pelvis orientations in the base frame using a 6D representation over a 5-step horizon	$30 = 6 \times 5$	Base
Projected Gravity	Gravity vector expressed in the robot base frame	3	Base
Base Velocity	Linear and angular velocities of the pelvis link.	$6 = 3 + 3$	Base
Joint State	Joint positions and velocities	$58 = 2 \times 29$	Base
Last Action	Previous joint position commands	29	Base
Key Body Position	Relative positions of 14 key bodies in the base frame	$42 = 3 \times 14$	Priv.
Key Body Orientation	Relative orientations of 14 key bodies in the base frame using a 6D representation	$84 = 6 \times 14$	Priv.

TABLE VIII: **Reward functions for Motion Tracker.** Tracking terms adopt a negative exponential form reward, $r_i = \exp(-\|e\|^2/\sigma^2)$, where e denotes the corresponding tracking error and σ is the scale factor.

Term	Weight	Description
Tracking Terms		
Pelvis Position	0.5	Global pelvis position error ($\sigma = 0.3$ m)
Pelvis Orientation	0.5	Global pelvis orientation error ($\sigma = 0.4$ rad)
Key Body Position	1.0	Relative body position error in the base frame ($\sigma = 0.3$ m)
Key Body Orientation	1.0	Relative body orientation error in the base frame ($\sigma = 0.4$ rad)
Body Linear Velocity	1.0	Global body linear velocity error in the base frame ($\sigma = 1.0$ m/s)
Body Angular Velocity	1.0	Global body angular velocity error in the base frame ($\sigma = 3.14$ rad/s)
Regularization		
Action Rate	-0.1	L2 penalty on action differences
Joint Limit	-10.0	Penalty for joint position limit violations
Undesired Contact	-0.1	Penalty for contacts on non-end-effector links
Feet Slide	-0.3	Penalty on foot velocity during contact
Soft Landing	$-3.0e-4$	Penalty on impact forces for smooth footfalls
Overspeed	-1.0	Penalty for joint velocities exceeding 20.0 rad/s
Overeffort	-1.0	Penalty for joint torques exceeding actuator limits

TABLE IX: **Termination conditions for Motion Tracker.**

Term	Description
Timeout	Episode terminates when reaching the maximum duration ($T = 10$ s) or when the reference motion sequence is completed.
Pelvis Orientation Deviation	Deviation in gravity-aligned orientation between the reference and robot pelvis exceeds a threshold of 0.8, measured by the difference in the projected gravity vector along the pelvis z-axis.
Pelvis Height Deviation	z-axis tracking error of the robot pelvis relative to the reference motion exceeds 0.25 m.
End-Effector Height Deviation	z-axis tracking error of the robot end-effectors (ankles and wrists) relative to the reference motion exceeds 0.5 m.

A. Experimental Procedure

Real-world Deployment. In the quantitative evaluation of real-robot executions, we record the real-world robot trajectory by combining joint angles from motor sensors and the global position and orientation of the pelvis provided by the built-in odometry of the G1 robot.

In the "Random" command experiments, we construct a 30-second text stream by randomly sampling action descriptions from the BABEL training set. Specifically, 3–5 words are first sampled from the dataset vocabulary, and each word is assigned a random duration uniformly selected from $6, 7, 8, 9, 10$ seconds. To ensure a well-defined start and termination, the sequence is prefixed and suffixed with a fixed 2-second "stand" command. Based on this text stream, reference motions are generated offline using our model. Owing to the limited physical space of the experimental environment, we discard any generated motion whose maximum displacement from the initial position exceeds 2 meters. The remaining motion trajectories are then executed on the real robot.

To evaluate real-time performance, we deploy the entire system on the physical robot with motor command execution disabled and measure the latency of each module independently. The latency of the text encoding and motion generator modules is evaluated over 100 runs, whereas the tracking policy latency is measured over 1000 runs. To assess the latency of user interaction, we repeatedly command the real robot to switch from a "stand" command to a "wave left hand" command, and manually record the elapsed time between the moment a new text command is entered by the user and the moment the user perceives the robot initiating the corresponding motion. This latency is measured over 10 runs.

Motion Generation Evaluation. We evaluate motion generation quality through a two-stage procedure consisting of

TABLE X: **Domain randomization settings for Motion Tracker.**

Term	Value
Static friction	$\mathcal{U}(0.3, 1.6)$
Dynamic friction	$\mathcal{U}(0.3, 1.2)$
Restitution	$\mathcal{U}(0.0, 0.5)$
Joint default position offset	$\mathcal{U}(-0.01, 0.01)$ rad
Torso CoM offset (x, y, z)	$\mathcal{U}(-0.025, 0.025)$ m, $\mathcal{U}(-0.05, 0.05)$ m, $\mathcal{U}(-0.05, 0.05)$ m
External Push	Interval in $\mathcal{U}(1.0, 3.0)$ s

TABLE XI: **Training and simulation hyperparameters for Motion Tracker.**

Parameter	Value
Number of Environments	8192
Actor MLP Size	$[2048, 1024, 512]$
Critic MLP Size	$[2048, 1024, 512]$
Activation Function	ELU
Step per Environment	24
PPO Init Learning Rate	$1.0e-3$
PPO Mini-batches	4
PPO Learning Epochs	5
PPO Clip Parameter	0.2
Simulation Frequency	200 Hz
Policy Frequency	50 Hz

motion rollout and metric-based assessment.

We first construct a evaluation set from the BABEL validation split. Each motion sequence, together with its associated textual annotations and temporal durations, is segmented into chunks with a maximum length of 200 frames. Given these reconstructed textual prompts, the trained motion generation model is used to generate corresponding motion sequences. The generated motions are then evaluated from two complementary perspectives: segment-level quality and transition-level quality.

Segment-level quality assesses how well each generated motion segment matches its corresponding textual description, using standard motion–text consistency metrics computed independently for each segment.

Transition-level quality focuses on the temporal smoothness and coherence at textual transition boundaries. For each transition point, we extract a short motion clip consisting of 15 frames before and after the boundary, centered at the transition. Metrics are then computed on these clips to specifically assess the quality of motion transitions.

Motion Tracking Evaluation. The explanation of experimental procedure in the main text is detailed enough.

B. Evaluation Metrics

We provide detailed definitions of the evaluation metrics used in our experiments.

Motion Quality. We train the motion and text feature extractors based on TMR [39] to evaluate the quality of the generated motions. Specifically, we construct the BABEL dataset into individual text-motion pairs, with a maximum motion length of 200 frames. Through contrastive learning, we train motion and text feature extractors to produce closely aligned embeddings for semantically matching text-motion pairs.

- **Frechet Inception Distance (FID).** FID is the principal metric for assessing the distributional similarity between generated and real motions in the extracted feature space. Let P_g represent the empirical distribution of generated motion features and P_r the empirical distribution of real motion features, with μ_g, Σ_g and μ_r, Σ_r as their respective means and covariances. The FID is computed as:

$$\text{FID} = \|\mu_g - \mu_r\|^2 + \text{Tr}\left(\Sigma_g + \Sigma_r - 2(\Sigma_g\Sigma_r)^{1/2}\right) \quad (13)$$

- **Diversity (DIV).** Diversity measures the variance across the entire set of generated motions. To compute DIV, all generated motions are randomly split into two subsets of equal size D, with feature vectors $\{x_i\}_{i=1}^{D}$ and $\{x'_i\}_{i=1}^{D}$. The metric is defined as:

$$\text{Diversity} = \frac{1}{X_d}\sum_{i=1}^{X_d}\|x_i - x'_i\| \quad (14)$$

- **R-precision (R@K).** R@K evaluates text-to-motion semantic alignment by measuring retrieval accuracy within the feature space. Each time, a batch of $M = 32$ motion-text pairs is sampled from the generated motion. For each motion, its corresponding text is used as a query to retrieve the most similar motions from the batch, by ranking motions based on Euclidean distance in the extracted feature space learned by the text and motion extractors. R@K is then defined as the percentage of queries for which the correct motion appears among the top-K retrieved results.
- **Multimodal distance (MM-Dist).** MM-Dist directly measures the average proximity between a generated motion and its corresponding text in the feature space. For a batch of $M = 32$ generated motions with their

corresponding text descriptions, let $\{h_i\}_{i=1}^M$ and $\{x_i\}_{i=1}^M$ be their respective text and motion feature vectors. MM-Dist is computed as:

$$\text{MM-Dist} = \frac{1}{M}\sum_{i=1}^{M}\|x_i - h_i\| \tag{15}$$

- **Peak Jerk (PJ).** PJ captures the most extreme instantaneous jerk occurring during a motion transition. It is defined as the maximum absolute jerk value observed across all joints K and all time steps within the transition sequence of length L_{tr}:

$$\text{PJ} = \max_{\substack{1 \le i \le K \\ 1 \le t \le L_{tr}}} |j_i(t)|_1, \tag{16}$$

where $j_i(t)$ is the jerk of joint i at time t.

- **Area Under the Jerk (AUJ).** AUJ measures the cumulative deviation from natural smoothness over the entire transition. It aggregates the absolute difference between the instantaneous jerk at each frame and the dataset's jerk level j_{avg}. The metric is computed as:

$$\text{AUJ} = \sum_{t=1}^{L_{tr}} \max_{1 \le i \le K} |j_i(t) - j_{avg}|_1 \tag{17}$$

Tracking Fidelity.

- **Success Rate (Succ).** In the real-world experiments, a trial is considered successful if the robot completes the full motion execution without falling, encountering hardware malfunctions, or exceeding the predefined safety boundaries of the workspace.

  In the simulation experiments, success rate measures the proportion of motion sequences whose root-relative tracking error remains below a predefined threshold over the entire trajectory. For each sequence i, we define the maximum root-relative mean per-link position error as

$$e_i = \max_t \frac{1}{J}\sum_{j=1}^{J}\|(\mathbf{p}_t^{\text{pol}}[j] - \mathbf{p}_t^{\text{pol}}[0]) - (\mathbf{p}_t^{\text{ref}}[j] - \mathbf{p}_t^{\text{ref}}[0])\|_2. \tag{18}$$

  The success rate is then computed as

$$\text{Succ} = \frac{1}{N}\sum_{i=1}^{N}\mathbb{I}(e_i \le \theta), \tag{19}$$

  where N is the number of test sequences and $\theta = 0.3\,\text{m}$.

- **Global MPJPE ($E_{\text{g-mpjpe}}$).** Global Mean Per Joint Position Error measures the average 3D position error across all links:

$$E_{\text{g-mpjpe}} = \frac{1000}{T \cdot J}\sum_{t=1}^{T}\sum_{j=1}^{J}\|\mathbf{p}_t^{\text{pol}}[j] - \mathbf{p}_t^{\text{ref}}[j]\|_2 \tag{20}$$

  where T is the number of time steps, J is the total number of links, and the factor of 1000 converts from meters to millimeters.

  Although termed *Per Joint*, the name follows conventions established in prior virtual character animation literature; in our robot setting, the metric is computed over all corresponding links.

- **Root-relative MPJPE (E_{mpjpe}).** Root-relative MPJPE measures position errors relative to the root link, eliminating systematic translation errors:

$$E_{\text{mpjpe}} = \frac{1000}{T \cdot J}\sum_{t=1}^{T}\sum_{j=1}^{J}\|(\mathbf{p}_t^{\text{pol}}[j] - \mathbf{p}_t^{\text{pol}}[0]) - (\mathbf{p}_t^{\text{ref}}[j] - \mathbf{p}_t^{\text{ref}}[0])\|_2 \tag{21}$$

  where joint 0 represents the root (pelvis) link.

- **Velocity Error (E_{vel}).** Velocity error measures temporal consistency of motion execution:

$$E_{\text{vel}} = \frac{1000}{T-1}\sum_{t=1}^{T-1}\|\mathbf{v}_t^{\text{pol}} - \mathbf{v}_t^{\text{ref}}\|_2 \tag{22}$$

  where velocities are computed as $\mathbf{v}_t = \mathbf{p}_{t+1} - \mathbf{p}_t$.

- **Acceleration Error (E_{acc}).** Acceleration error captures higher-order temporal dynamics:

$$E_{\text{acc}} = \frac{1000}{T-2}\sum_{t=1}^{T-2}\|\mathbf{a}_t^{\text{pol}} - \mathbf{a}_t^{\text{ref}}\|_2 \tag{23}$$

  where accelerations are computed as $\mathbf{a}_t = \mathbf{v}_{t+1} - \mathbf{v}_t$.

C. Baseline Methods

This subsection describes the baseline methods used for comparison.

Motion Representation. Table XII illustrates the differences in motion feature representations between **TextOp** and the baselines.

Motion Tracking.

- TWIST2 [66] is a scalable and holistic humanoid data collection system designed to generate high-quality, whole-body demonstrations. Within its hierarchical framework, the low-level controller is implemented as a task-agnostic motion tracking engine. We adopt the officially released checkpoint of the low-level controller to conduct the experiment.
- GMT [12] trains a single unified policy for general humanoid motion tracking through two core components: an adaptive sampling strategy that balances easy and difficult motions during training, and a Motion Mixture-of-Experts architecture that specializes across different regions of the motion manifold.
- Any2Track [70] employs a two-stage reinforcement learning framework that reformulates dynamics adaptability as an additional capability on top of basic action execution. The system consists of AnyTracker, a general motion tracker for handling diverse, highly dynamic motions within a single policy, and AnyAdapter, a history-informed adaptation module that provides online dynamics adaptability. We adopt the officially released

TABLE XII: **Comparison of motion representation components used by different methods.** ✓ indicates the representation is used, × indicates it is not used. Components are reported based on the actual feature vectors used by each method, regardless of semantic overlap between each other. For the following tables, in the Unitree G1 skeleton, the term "Body" refers to 29 body links, excluding the root link, whereas in the SMPL skeleton it corresponds to 22 spherical joints. The "Base Frame" denotes the coordinate frame attached to the root link. In contrast, the "Character Frame" is a local frame aligned with the root's yaw: its origin coincides with the root position, its yaw matches the root yaw, and its roll and pitch are set to zero.

Components	**Dimension**	**DART+Retarget**	**BeyondMimic**	**HumanML3D**	**RobotMDM**	**TextOp**
Is Robot-Skeleton Generation	-	×	✓	✓	✓	✓
Root Trigonometric Encoding of Roll and Pitch	4	×	×	×	×	✓
Root Rotation 6D Representation	6	✓	×	×	×	×
Root Rotation Quaternion Representation	4	×	✓	✓	✓	×
Difference of Root Rotation 6D Representation	6	✓	×	×	×	×
Root Yaw Velocity	1	×	×	×	×	✓
Root Angular Velocity in the Z-axis	1	×	×	✓	✓	×
Root Angular Velocity	3	×	✓	×	×	×
Root Position	3	✓	✓	×	×	×
Root Height	1	×	×	✓	✓	✓
Root Linear Velocity in the Character Frame	3	✓	✓	×	×	✓
Root Linear Velocity in the XY-plane in the Character Frame	2	×	×	✓	✓	×
Dof Angle	29	×	×	✓	✓	✓
Dof Velocity	29	×	×	×	✓	✓
Body Rotation 6D Representation in the Base Frame	126 for SMPL	✓	×	×	×	×
Body Position in the Character Frame	87 (66 for SMPL)	✓	✓	✓	×	×
Body Linear Velocity in the Character Frame	87 (66 for SMPL)	✓	✓	✓	×	×
Foot Contact	2	×	×	✓	×	✓
Total Dim	-	276	187	213	66	69

checkpoint of AnyTracker, the model trained in the first stage to conduct the experiment, since the second stage is not open-source.

D. Additional Ablation Studies

As shown in Table XIII, we conduct additional ablation studies to analyze the impact of key design choices in the motion generation module, including history length T_{history}, future length T_{future}, number of primitives N_{prim}, hidden dimension and layer count of LDM, guidance scale σ_{CFG}, number of denoising steps, and maximum rollout probability. Through a trade-off analysis considering distribution matching, semantic alignment, and motion smoothness, we ultimately select a set of parameters that achieves balanced performance across all metrics.

E. Additional Real-World Results

Additional real-world robot results are shown in Fig. 7.

F. Computational Resources

Model training is conducted on a server equipped with an NVIDIA A100-PCIE GPU with 40 GB memory and an Intel Xeon Gold 6348 CPU operating at 2.60 GHz. The training system runs Ubuntu 22.04.3 LTS.

For deployment, we use a separate workstation with an NVIDIA GeForce RTX 4090 GPU and a 13th Gen Intel Core i7-13700 CPU. The deployment environment runs Ubuntu 20.04.6 LTS.

TABLE XIII: **Ablation study on key hyperparameters for Motion Generation.** All results are reported as mean ± standard deviation over three generator rollout seeds.

Setting	Segment						Transition			
	FID ↓	Diversity →	R@1 ↑	R@2 ↑	R@3 ↑	MM-dist ↓	FID ↓	Diversity →	PJ →	AUJ ↓
Dataset	0.000 ± 0.000	9.150 ± 0.158	0.357 ± 0.004	0.556 ± 0.006	0.684 ± 0.005	3.729 ± 0.000	0.000 ± 0.000	8.138 ± 0.013	0.005 ± 0.000	0.000 ± 0.000
$T_{\text{history}} = 1$	2.972 ± 0.032	9.082 ± 0.139	0.283 ± 0.011	0.461 ± 0.002	0.588 ± 0.004	4.321 ± 0.107	4.107 ± 0.268	8.528 ± 0.139	0.017 ± 0.002	0.128 ± 0.002
$T_{\text{history}} = 2$ (Ours)	3.072 ± 0.199	9.220 ± 0.151	0.300 ± 0.007	0.481 ± 0.004	0.607 ± 0.008	4.272 ± 0.058	3.238 ± 0.430	8.226 ± 0.079	0.015 ± 0.001	0.125 ± 0.001
$T_{\text{history}} = 3$	3.718 ± 0.091	9.218 ± 0.094	0.287 ± 0.012	0.467 ± 0.009	0.594 ± 0.007	4.366 ± 0.062	3.695 ± 0.167	8.470 ± 0.085	0.016 ± 0.001	0.126 ± 0.001
$T_{\text{history}} = 5$	4.713 ± 0.046	9.060 ± 0.079	0.279 ± 0.012	0.453 ± 0.001	0.566 ± 0.006	4.508 ± 0.049	4.750 ± 0.121	8.432 ± 0.154	0.019 ± 0.000	0.127 ± 0.001
$T_{\text{future}} = 4$	3.339 ± 0.089	8.808 ± 0.104	0.256 ± 0.010	0.431 ± 0.011	0.549 ± 0.003	4.747 ± 0.023	2.904 ± 0.140	8.301 ± 0.136	0.013 ± 0.000	0.127 ± 0.000
$T_{\text{future}} = 6$	2.930 ± 0.098	9.096 ± 0.027	0.283 ± 0.010	0.467 ± 0.011	0.584 ± 0.006	4.481 ± 0.027	2.910 ± 0.193	8.453 ± 0.135	0.016 ± 0.001	0.126 ± 0.001
$T_{\text{future}} = 8$ (Ours)	3.072 ± 0.199	9.220 ± 0.151	0.300 ± 0.007	0.481 ± 0.004	0.607 ± 0.008	4.272 ± 0.058	3.238 ± 0.430	8.226 ± 0.079	0.015 ± 0.001	0.125 ± 0.001
$T_{\text{future}} = 10$	4.126 ± 0.073	9.206 ± 0.070	0.326 ± 0.004	0.501 ± 0.004	0.624 ± 0.004	3.854 ± 0.015	5.467 ± 0.192	8.606 ± 0.163	0.028 ± 0.002	0.186 ± 0.002
$T_{\text{future}} = 12$	3.161 ± 0.062	9.213 ± 0.175	0.300 ± 0.003	0.478 ± 0.013	0.603 ± 0.010	4.101 ± 0.025	4.080 ± 0.244	8.496 ± 0.125	0.016 ± 0.001	0.127 ± 0.002
$N_{\text{prim}} = 1$	3.044 ± 0.204	8.391 ± 0.066	0.196 ± 0.005	0.346 ± 0.011	0.452 ± 0.014	6.245 ± 0.058	3.074 ± 0.183	8.047 ± 0.275	0.013 ± 0.001	0.123 ± 0.001
$N_{\text{prim}} = 2$	2.578 ± 0.037	8.861 ± 0.117	0.256 ± 0.011	0.419 ± 0.011	0.547 ± 0.015	4.990 ± 0.038	2.277 ± 0.197	8.265 ± 0.087	0.011 ± 0.000	0.122 ± 0.000
$N_{\text{prim}} = 4$ (Ours)	3.072 ± 0.199	9.220 ± 0.151	0.300 ± 0.007	0.481 ± 0.004	0.607 ± 0.008	4.272 ± 0.058	3.238 ± 0.430	8.226 ± 0.079	0.015 ± 0.001	0.125 ± 0.001
$N_{\text{prim}} = 6$	3.733 ± 0.088	9.141 ± 0.092	0.307 ± 0.008	0.492 ± 0.010	0.616 ± 0.012	4.011 ± 0.012	4.802 ± 0.205	8.421 ± 0.054	0.022 ± 0.002	0.133 ± 0.002
$N_{\text{prim}} = 10$	3.344 ± 0.098	9.111 ± 0.102	0.314 ± 0.010	0.482 ± 0.014	0.611 ± 0.005	3.897 ± 0.060	5.131 ± 0.272	8.315 ± 0.064	0.023 ± 0.001	0.134 ± 0.001
Hidden = 128	4.030 ± 0.242	9.493 ± 0.126	0.255 ± 0.006	0.435 ± 0.005	0.558 ± 0.006	4.640 ± 0.032	5.353 ± 0.208	8.903 ± 0.126	0.015 ± 0.001	0.126 ± 0.002
Hidden = 256	3.110 ± 0.082	9.089 ± 0.123	0.273 ± 0.009	0.448 ± 0.004	0.582 ± 0.009	4.487 ± 0.051	2.506 ± 0.197	8.279 ± 0.159	0.011 ± 0.000	0.123 ± 0.000
Hidden = 512 (Ours)	3.072 ± 0.199	9.220 ± 0.151	0.300 ± 0.007	0.481 ± 0.004	0.607 ± 0.008	4.272 ± 0.058	3.238 ± 0.430	8.226 ± 0.079	0.015 ± 0.001	0.125 ± 0.001
Hidden = 1024	3.552 ± 0.194	9.028 ± 0.167	0.277 ± 0.009	0.464 ± 0.012	0.598 ± 0.007	4.324 ± 0.040	3.607 ± 0.076	8.451 ± 0.100	0.014 ± 0.000	0.124 ± 0.001
Hidden = 2048	3.688 ± 0.131	9.154 ± 0.211	0.285 ± 0.010	0.449 ± 0.009	0.581 ± 0.004	4.509 ± 0.038	4.326 ± 0.385	8.633 ± 0.161	0.013 ± 0.000	0.125 ± 0.000
Layers = 4	4.186 ± 0.098	9.315 ± 0.089	0.303 ± 0.010	0.477 ± 0.006	0.597 ± 0.004	4.225 ± 0.024	5.151 ± 0.280	8.584 ± 0.117	0.014 ± 0.000	0.124 ± 0.000
Layers = 6	3.868 ± 0.203	9.213 ± 0.077	0.291 ± 0.005	0.462 ± 0.009	0.596 ± 0.004	4.275 ± 0.034	4.609 ± 0.365	8.701 ± 0.187	0.015 ± 0.001	0.127 ± 0.004
Layers = 8 (Ours)	3.072 ± 0.199	9.220 ± 0.151	0.300 ± 0.007	0.481 ± 0.004	0.607 ± 0.008	4.272 ± 0.058	3.238 ± 0.430	8.226 ± 0.079	0.015 ± 0.001	0.125 ± 0.001
Layers = 10	3.342 ± 0.082	9.030 ± 0.051	0.296 ± 0.003	0.476 ± 0.002	0.601 ± 0.007	4.254 ± 0.021	3.272 ± 0.199	8.205 ± 0.056	0.012 ± 0.000	0.123 ± 0.001
$\sigma_{\text{CFG}} = 3$	2.537 ± 0.056	8.919 ± 0.227	0.259 ± 0.003	0.441 ± 0.004	0.567 ± 0.006	4.667 ± 0.048	1.493 ± 0.168	8.036 ± 0.092	0.013 ± 0.001	0.124 ± 0.001
$\sigma_{\text{CFG}} = 4$	2.818 ± 0.118	9.058 ± 0.133	0.295 ± 0.008	0.480 ± 0.007	0.597 ± 0.005	4.384 ± 0.052	2.499 ± 0.199	8.258 ± 0.150	0.014 ± 0.000	0.124 ± 0.001
$\sigma_{\text{CFG}} = 5$ (Ours)	3.072 ± 0.199	9.220 ± 0.151	0.300 ± 0.007	0.481 ± 0.004	0.607 ± 0.008	4.272 ± 0.058	3.238 ± 0.430	8.226 ± 0.079	0.015 ± 0.001	0.125 ± 0.001
$\sigma_{\text{CFG}} = 6$	3.588 ± 0.148	9.197 ± 0.071	0.304 ± 0.001	0.486 ± 0.002	0.610 ± 0.010	4.141 ± 0.016	4.015 ± 0.282	8.279 ± 0.090	0.015 ± 0.000	0.125 ± 0.000
$\sigma_{\text{CFG}} = 7$	3.702 ± 0.118	9.195 ± 0.109	0.306 ± 0.013	0.489 ± 0.009	0.608 ± 0.010	4.094 ± 0.037	4.365 ± 0.369	8.323 ± 0.138	0.016 ± 0.000	0.127 ± 0.001
Steps = 3	3.755 ± 0.057	9.111 ± 0.164	0.309 ± 0.010	0.481 ± 0.011	0.608 ± 0.007	4.078 ± 0.017	4.786 ± 0.193	8.542 ± 0.144	0.017 ± 0.000	0.384 ± 0.001
Steps = 5 (Ours)	3.072 ± 0.199	9.220 ± 0.151	0.300 ± 0.007	0.481 ± 0.004	0.607 ± 0.008	4.272 ± 0.058	3.238 ± 0.430	8.226 ± 0.079	0.015 ± 0.001	0.125 ± 0.001
Steps = 7	3.451 ± 0.047	9.230 ± 0.078	0.308 ± 0.014	0.483 ± 0.007	0.616 ± 0.006	3.855 ± 0.012	4.913 ± 0.212	8.427 ± 0.114	0.017 ± 0.001	0.127 ± 0.002
Steps = 10	3.640 ± 0.072	8.951 ± 0.106	0.317 ± 0.008	0.497 ± 0.007	0.623 ± 0.007	3.858 ± 0.022	5.132 ± 0.309	8.350 ± 0.117	0.021 ± 0.000	0.133 ± 0.001
Rollout = 0	2.310 ± 0.112	8.540 ± 0.044	0.208 ± 0.003	0.349 ± 0.005	0.474 ± 0.015	5.936 ± 0.015	1.129 ± 0.159	8.087 ± 0.214	0.011 ± 0.000	0.123 ± 0.001
Rollout = 0.4	2.461 ± 0.089	8.831 ± 0.074	0.262 ± 0.010	0.419 ± 0.017	0.543 ± 0.008	4.960 ± 0.040	2.525 ± 0.202	8.541 ± 0.102	0.015 ± 0.000	0.125 ± 0.000
Rollout = 0.6	3.323 ± 0.016	9.198 ± 0.075	0.305 ± 0.017	0.479 ± 0.013	0.610 ± 0.006	4.176 ± 0.045	3.537 ± 0.201	8.386 ± 0.074	0.013 ± 0.001	0.124 ± 0.001
Rollout = 0.8 (Ours)	3.072 ± 0.199	9.220 ± 0.151	0.300 ± 0.007	0.481 ± 0.004	0.607 ± 0.008	4.272 ± 0.058	3.238 ± 0.430	8.226 ± 0.079	0.015 ± 0.001	0.125 ± 0.001
Rollout = 1	4.008 ± 0.026	9.185 ± 0.137	0.315 ± 0.005	0.500 ± 0.008	0.624 ± 0.008	3.847 ± 0.006	4.363 ± 0.253	8.192 ± 0.062	0.018 ± 0.001	0.129 ± 0.003

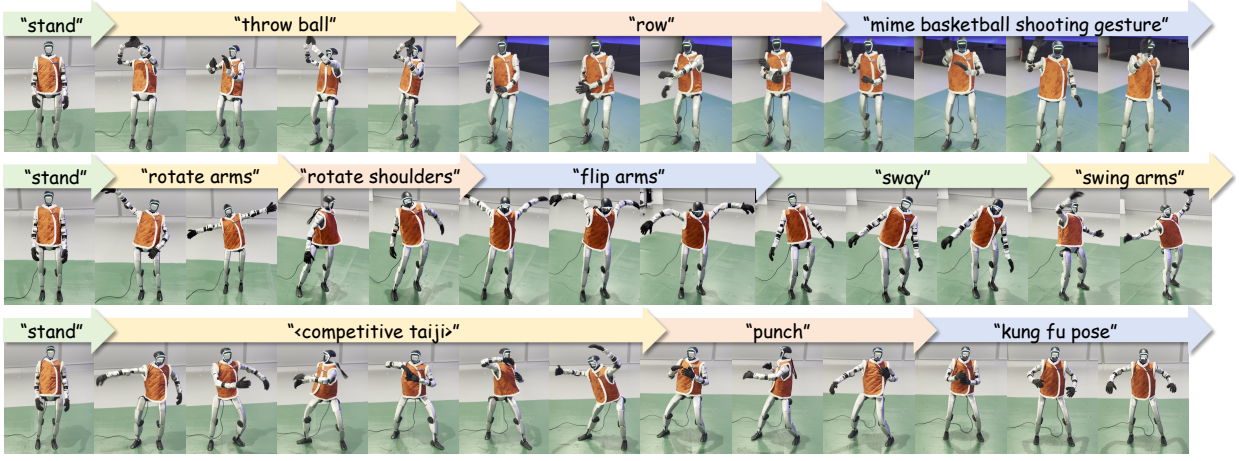

Fig. 7: **Additional results of continuous diverse skill execution in the real robot.**