ULC: A Unified and Fine-Grained Controller for Humanoid Loco-Manipulation

Wandong Sun[1*], Luying Feng[2*], Baoshi Cao[1], Yang Liu[1], Yaochu Jin[2†], Zongwu Xie[1†]

***Abstract*—Loco-Manipulation for humanoid robots aims to enable robots to integrate mobility with upper-body tracking capabilities. Most existing approaches adopt hierarchical architectures that decompose control into isolated upper-body (manipulation) and lower-body (locomotion) policies. While this decomposition reduces training complexity, it inherently limits coordination between subsystems and contradicts the unified whole-body control exhibited by humans. We demonstrate that a single unified policy can achieve a combination of tracking accuracy, large workspace, and robustness for humanoid loco-manipulation. We propose the Unified Loco-Manipulation Controller (ULC), a single-policy framework that simultaneously tracks root velocity, root height, torso rotation, and dual-arm joint positions in an end-to-end manner, proving the feasibility of unified control without sacrificing performance. We achieve this unified control through key technologies: sequence skill acquisition for progressive learning complexity, residual action modeling for fine-grained control adjustments, command polynomial interpolation for smooth motion transitions, random delay release for robustness to deploy variations, load randomization for generalization to external disturbances, and center-of-gravity tracking for providing explicit policy gradients to maintain stability. We validate our method on the Unitree G1 humanoid robot with 3-DOF (degrees-of-freedom) waist. Compared with strong baselines, ULC shows better tracking performance to disentangled methods and demonstrating larger workspace coverage. The unified dual-arm tracking enables precise manipulation under external loads while maintaining coordinated whole-body control for complex loco-manipulation tasks. The code and videos are available on our project website at https://ulc-humanoid.github.io.**

***Index Terms*—Humanoid Robots, Loco-Manipulation, Reinforcement Learning, Whole-Body Control**

I. INTRODUCTION

Humanoid robots, with their human-like morphology, represent a promising paradigm for versatile robotic systems capable of operating in human-designed environments. Recent years have witnessed remarkable advances in locomotion [1, 2, 3, 4, 5, 6] and autonomous manipulation [7, 8, 9, 10, 11] capabilities for humanoid platforms. These achievements are enabled by the synergistic integration of high-level decision-making layers, powered by Imitation Learning (IL) models [12, 13] or Vision-Language-Action (VLA) models [14, 15, 16], with sophisticated Loco-Manipulation Controllers (LMCs) [17, 18, 19, 20, 21, 22] that translate high-level commands into precise whole-body motions for autonomous locomotion and dexterous dual-arm manipulation.

*These authors contributed equally to this work
† denotes the corresponding author
[1]State Key Laboratory of Robotics and Systems, Harbin Institute of Technology.
[2]School of Engineering, Westlake University.

An ideal Loco-Manipulation Controller (LMC) should seamlessly translate whole-body motion commands into precise joint-level actions, minimizing the discrepancy between commanded and executed motions while guaranteeing dynamic stability. However, designing effective LMCs involves several critical architectural decisions that significantly impact performance:

- **Command Space Selection**. Robot actions can be parameterized through various representations [20], including joint positions, Cartesian pose targets, root velocity, and root height. An effective command space should eliminate potential conflicts between different command modalities while enabling full exploration of the robot's kinematic and dynamic capabilities.
- **Unified vs. Decoupled Control Architecture**. Whole-body controllers [23, 20, 24, 25, 21] can theoretically achieve superior performance by coordinating all degrees of freedom simultaneously, but are commonly perceived as more challenging to train effectively compared to specialized controllers. Alternatively, some approaches decouple the LMC into separate upper and lower body controllers [19, 22, 18, 17], which accelerates learning but may compromise performance in scenarios requiring tight coordination between locomotion and manipulation.
- **Motion Capture vs. Procedural Training Data**. Motion capture data provides physically plausible whole-body movement patterns, but inherent noise and kinematic infeasibilities can significantly degrade tracking accuracy [23, 20, 24, 25, 26]. Additionally, distribution bias in motion capture datasets poses risks when encountering out-of-distribution movements during deployment. Procedurally sampled command spaces can mitigate distribution bias but are primarily limited to upper-body motions, as the inherent instability of humanoid platforms precludes obtaining stable leg references through random sampling [17, 18, 19, 21].

These design choices present fundamental trade-offs that significantly impact the practical deployment of humanoid loco-manipulation systems. Each decision involves balancing competing objectives: command space design must reconcile expressiveness with feasibility, control architecture must weigh coordination benefits against training complexity, and data generation must balance physical plausibility with distribution coverage. The challenge lies not in optimizing any single aspect, but in finding the optimal combination of design decisions that collectively enable robust, versatile, and deployable loco-manipulation capabilities.

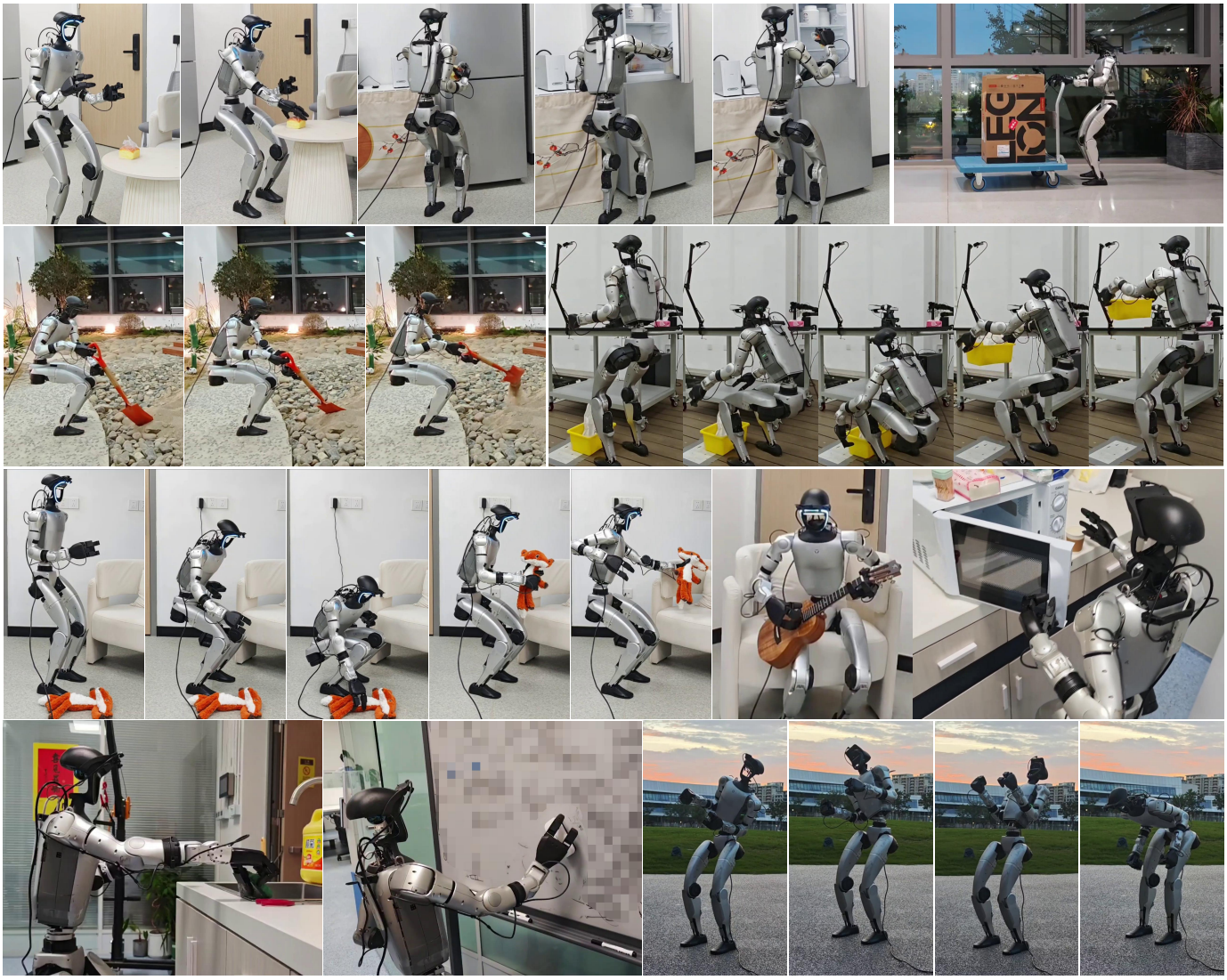

Fig. 1: Diverse loco-manipulation capabilities enabled by ULC. The humanoid robot demonstrates various coordinated whole-body actions including: picking up bread from a table and placing it in a refrigerator, pushing a cart with coordinated locomotion, squatting to shovel sand from the ground, lifting boxes from the floor to table height with dual-arm coordination, picking up dolls from the ground with hand switching and placing them on a sofa, sitting and playing ukulele with fine motor control, placing items in a microwave with precise manipulation, cleaning kitchen surfaces with wiping motions, erasing blackboards with arm coordination, and performing torso rotation in outdoor environments.

A. *Objective of this Work*

The objective of this work is to design a practical and versatile loco-manipulation controller that balances design decisions to best fit real-world scenarios. Specifically, we aim to develop a command space that comprehensively covers the majority of human mobility and manipulation scenarios while maintaining physical feasibility and eliminating command conflicts. We pursue a unified control architecture that coordinates the entire body simultaneously to maximize workspace coverage and enable tight coupling between locomotion and manipulation, challenging the conventional wisdom that decoupled approaches are necessary for practical deployment. To ensure robustness and generalization, we adopt procedural command generation rather than motion capture data, developing novel training methodologies that mitigate the inherent challenges of random sampling for humanoid locomotion while preserving the distribution coverage advantages. Our goal is to demonstrate that unified whole-body control can achieve both the precision of specialized controllers and the coordination benefits of integrated approaches, creating a system that is not only theoretically sound but practically deployable in real-world scenarios.

B. *Contributions*

To achieve these objectives, we propose the Unified Loco-Manipulation Controller (ULC), a unified control framework that employs massively parallel reinforcement learning (RL) to accurately track procedurally sampled commands including root velocity, root height, torso orientation, and arm joint positions. This design choice deliberately simplifies leg commands compared to full motion capture approaches, but enables comprehensive coverage of the feasible command

space through principled random sampling while preserving the coordination benefits of unified control. To realize these theoretical advantages in practice, we identify and address three fundamental technical challenges in developing ULC, presenting novel solutions for each:

- **Multi-Task Learning in Unified Control**. Single-model multi-task tracking for humanoid robots often suffers from reduced single-task performance due to potential conflicts between heterogeneous command modalities and gradient interference across tasks [27]. We address these issues through: (1) careful command space design to ensure physical feasibility of command combinations, (2) progressive command curriculum learning from simple to difficult to enable systematic capability exploration, (3) residual action modeling [28, 29, 5] for both arms to enhance tracking precision, and (4) sequential skill acquisition [30] where training on subsequent skills begins only after achieving mastery of current capabilities, ensuring comprehensive skill development without catastrophic forgetting.
- **Deployment-Realistic Command Generation**. Out-of-distribution commands during deployment can lead to catastrophic failures [31]. While random sampling mitigates distribution bias, naive interval-based sampling creates target discontinuities, whereas continuous interpolation produces overly smooth trajectories inconsistent with real deployment scenarios. We develop a novel sampling strategy that combines fixed-interval random sampling with fifth-degree polynomial interpolation [32, 33] to generate smooth command transitions. To simulate deployment-realistic command variations, we introduce stochastic command release mechanisms where commands may be buffered or released with certain probabilities, ensuring all released commands remain within the feasible sampling range while introducing instruction mutations that may occur in actual deployment to enhance robustness.
- **Loaded Balance and Generalization**. For arm position tracking, controllers must maintain consistent performance under varying payload conditions while preserving whole-body stability. While randomizing end-effector masses [18] addresses dual-arm tracking accuracy under load to a certain extent, maintaining stability requires explicit balance considerations. We incorporate center-of-mass tracking rewards [34] by computing the robot's center of mass with loaded body mass distributions and encouraging the xy-plane projection to remain within the support polygon defined by the feet. This approach provides clear gradient signals for balance optimization and demonstrably enhances stability under varying load conditions.

Extensive experiments in both simulation and real-world settings demonstrate that our approach achieves state-of-the-art performance across a wide range of loco-manipulation tasks, outperforming existing baselines in tracking accuracy, workspace coverage, and robustness. Ablation studies further confirm that each component of our framework is essential. These results validate the effectiveness of our unified design for robust, high-precision loco-manipulation.

II. RELATED WORK

A. Learning Legged Locomotion

Reinforcement learning has emerged as the dominant paradigm for humanoid locomotion control, demonstrating remarkable capabilities in learning complex walking gaits and dynamic behaviors [35, 36, 37, 38, 33, 39, 40, 41, 42, 43]. The evolution from traditional model-based approaches to learning-based methods has been driven by the need to handle high-dimensional control spaces, environmental uncertainties, and the complexity of bipedal dynamics.

Early RL applications focused on basic locomotion tasks. [42] established foundational principles for learning locomotion policies to address the challenge of stable quadrupedal gait generation. [41] introduced Multiplicity of Behavior (MoB) to encode diverse locomotion strategies within a single policy, enabling real-time strategy selection for different tasks without retraining, though focused on quadrupedal locomotion.

[37] pioneered practical RL deployment on real humanoid platforms to solve sim-to-real transfer challenges, establishing fundamental principles including actuator modeling and environmental robustness, though limited to basic walking gaits. [35] advanced the field through sophisticated training frameworks to address multi-task learning complexity, incorporating curriculum learning and multi-objective optimization, but requires careful hyperparameter tuning. [38] explored perception-locomotion integration to solve visual navigation challenges with lidar height map. [33] provided standardized simulation environments to address reproducibility issues. [39] addresses motion smoothness challenges through Lipschitz-constrained policies to eliminate jerky movements. Several other works have explored balance recovery, energy efficiency, and adaptive gait generation, each addressing specific locomotion limitations.

Despite significant progress in locomotion, these RL-based approaches primarily focus on walking capabilities and lack integrated manipulation functionalities. The fundamental limitation is that they only enable basic locomotion without the ability to perform meaningful manipulation tasks, limiting their practical applicability in real-world scenarios that require coordinated loco-manipulation behaviors. Additional challenges remain in sample efficiency, safety guarantees, and generalization across diverse environments and tasks.

B. Humanoid Whole-Body Motion Capture Tracking

Humanoid whole-body motion tracking aims to enable robots to reproduce complex human motions from diverse datasets [1, 34, 5, 6, 45, 26, 46, 23, 47, 48, 24, 49, 50, 51, 52, 53, 4]. Key challenges include morphological differences, noise handling, and sim-to-real transfer.

Traditional approaches have primarily relied on model-based methods such as inverse kinematics, trajectory optimization, and model predictive control (MPC). While MPC can handle stability constraints and dynamics to some extent, these

Method	Architecture	Legs	Torso Yaw	Torso Pitch	Torso Roll	Dual Arms	Workspace	Precision
HOMIE [19]	Decoupled	RL-1	PD	-	-	PD	Medium	Medium
FALCON [18]	Decoupled	RL-1	RL-1	RL-1	RL-1	RL-2	Medium	High
JAEGER [22]	Decoupled	RL-1	RL-1	-	-	RL-2	Medium	High
AMO [21]	Decoupled	RL	RL	RL	RL	PD	Large	Medium
SoFTA [44]	Decoupled	RL-1	RL-1	-	-	RL-2	Medium	Medium
R^2S^2 [2]	Unified	RL	RL	RL	RL	RL	Medium	Medium
ULC (Ours)	Unified	RL	RL	RL	RL	RL	Large	High

TABLE I: Comparison of humanoid loco-manipulation controllers. Colors indicate control types: Blue/Red: RL, Orange: PD, Purple: Unified RL, Gray: Not controlled.

methods face significant limitations when dealing with complex whole-body motion tracking from human demonstrations. They struggle with the high-dimensional nature of humanoid systems, require accurate dynamic models that are difficult to obtain, and cannot easily adapt to the nuanced coordination patterns present in human motion data. The shift to deep RL enabled learning-based approaches that can directly learn complex coordination patterns from data without requiring explicit dynamic models. [51] pioneered adversarial motion priors to solve natural movement generation problems, though requires careful discriminator design and can be unstable.

[23] trains whole-body policies using large-scale motion capture datasets to address dexterous manipulation challenges, but suffers from noise and kinematic inconsistencies in captured data. [26] addresses data quality issues through teacher-student distillation to improve motion expression, yet exhibits tracking errors in fine-grained movements due to distribution mismatch. [48] relaxes leg constraints while requiring upper body tracking to enable natural social interactions, but there is still much room for improvement in tracking accuracy and workspace. [20] proposes neural architectures unifying both through shared representations to address coordination issues, though faces deployment challenges due to state space complexity.

[34] enables extreme motion reproduction through advanced processing pipelines to solve dynamic motion challenges, but lacks generalizability across different robots and requires extensive retraining. [6] explores visual imitation from video demonstrations to reduce motion capture dependency, but faces challenges in visual perception accuracy.

Our ULC deliberately avoids motion capture dependency, eliminating inherent noise and artifacts while training from scratch with carefully designed mechanisms for superior tracking and generalization.

C. Humanoid Loco-Manipulation Controller

Humanoid loco-manipulation requires coordinating locomotion and manipulation while maintaining tracking accuracy and robustness [17, 18, 19, 20, 21, 22, 54, 55, 56, 57, 11, 58, 59, 60, 61, 62, 2, 44]. The complexity involves simultaneous optimization of balance, end-effector positioning, and environmental adaptation.

Traditional decoupled approaches separate leg and arm control to simplify training complexity. [19] uses RL for legs and PD for arms to address basic loco-manipulation coordination, but results in poor arm tracking under gravitational loads and limited torso workspace. [18] jointly trains upper body policies with force curriculum to solve force adaptation challenges, but remains limited by restricted torso rotation and coordination deficiencies. [22] presents JAEGER with separate upper and lower body controllers supporting both coarse-grained root velocity tracking and fine-grained joint angle tracking, though relies on motion capture data retargeting that can introduce artifacts. [44] introduces SoFTA framework with separate upper-body and lower-body agents at different frequencies to solve end-effector stabilization during locomotion, but it has limited working space. [17] treats locomotion and manipulation as manifestations of the same control problem to solve architectural limitations, but the dual-arm tracking performance of PD control needs to be further improved. [21] combines trajectory optimization with RL through hierarchical design to address motion planning challenges, achieving better performance but introducing computational overhead and both arms are still controlled by PD. [2] proposes Real-world-Ready Skill Space (R^2S^2) to address large-space reaching through skill library ensembling, enabling diverse whole-body skills but requiring careful primitive skill design. Tab. I shows a horizontal comparison of various methods.

The fundamental trade-off remains between training complexity and performance. Decoupled designs offer simplicity but sacrifice coordination. Unified approaches promise better performance but face training complexity challenges. Our ULC addresses these limitations through end-to-end policy architecture trained specifically for loco-manipulation. By eschewing decoupled paradigms, ULC enables natural whole-body coordination while optimizing tracking accuracy, workspace coverage, and robustness.

III. PROBLEM FORMULATION

We formulate the humanoid loco-manipulation task as a goal-conditioned Markov Decision Process (MDP) defined by the tuple $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{G}, P, R, \gamma \rangle$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{G}$ is the goal (command) space, $P : \mathcal{S} \times \mathcal{A} \times \mathcal{G} \to \Delta(\mathcal{S})$ is the transition probability function, $R : \mathcal{S} \times \mathcal{A} \times \mathcal{G} \to \mathbb{R}$ is the reward function, and $\gamma \in [0, 1)$ is the discount factor.

The policy $\pi_{\boldsymbol{\theta}} : \mathcal{S} \times \mathcal{G} \to \Delta(\mathcal{A})$ is parameterized by neural network weights $\boldsymbol{\theta}$ and maps the concatenated state-goal observations to a probability distribution over actions:

Parameter	Unit	Range
Linear Velocity X	m/s	[-0.45, 0.55]
Linear Velocity Y	m/s	[-0.45, 0.45]
Angular Velocity Z	rad/s	[-1.2, 1.2]
Root Height	m	[0.3, 0.75]
Torso Rotation Yaw	rad	[-2.62, 2.62]
Torso Rotation Roll	rad	[-0.52, 0.52]
Torso Rotation Pitch	rad	[-0.52, 1.57]
Arm Joint Positions	-	Robot Design Limits

TABLE II: Components and ranges of command space.

$$\pi_{\boldsymbol{\theta}}(\boldsymbol{a}_t|\boldsymbol{s}_t,\boldsymbol{g}_t)=\mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{s}_t,\boldsymbol{g}_t),\boldsymbol{\Sigma}_{\boldsymbol{\theta}}(\boldsymbol{s}_t,\boldsymbol{g}_t)) \tag{1}$$

where $\boldsymbol{\mu}_{\boldsymbol{\theta}}$ and $\boldsymbol{\Sigma}_{\boldsymbol{\theta}}$ represent the mean and covariance matrix of the Gaussian policy distribution.

A. State Space and Observation Design

The state space $\mathcal{S}$ consists of proprioceptive observations $\boldsymbol{o}_{prop} \in \mathbb{R}^{n_s}$ that capture the robot's internal state without external sensing modalities. The proprioceptive observation vector at time t is defined as:

$$\boldsymbol{o}_{prop}^{(t)}=\begin{bmatrix}\boldsymbol{q}_{joint}^{(t)}\\ \dot{\boldsymbol{q}}_{joint}^{(t)}\\ \boldsymbol{\omega}_{base}^{(t)}\\ \boldsymbol{g}_{proj}^{(t)}\\ \boldsymbol{a}_{t-1}\\ \boldsymbol{g}_t\end{bmatrix} \tag{2}$$

where:

- $\boldsymbol{q}_{joint}^{(t)} \in \mathbb{R}^{n_j}$ represents joint positions for n_j actuated joints
- $\dot{\boldsymbol{q}}_{joint}^{(t)} \in \mathbb{R}^{n_j}$ denotes joint velocities
- $\boldsymbol{\omega}_{base}^{(t)} \in \mathbb{R}^3$ represents base angular velocity
- $\boldsymbol{g}_{proj}^{(t)} \in \mathbb{R}^3$ denotes gravity projection vector in base frame
- $\boldsymbol{a}_{t-1} \in \mathbb{R}^{n_j}$ is the previous timestep's action
- $\boldsymbol{g}_t \in \mathbb{R}^{n_g}$ is the current command

To enhance temporal reasoning and enable smooth control transitions, the policy observation incorporates both current and historical observations. The complete policy observation is formed by concatenating multiple timesteps:

$$\boldsymbol{s}_t=\left[\boldsymbol{o}_{prop}^{(t)},\boldsymbol{o}_{prop}^{(t-1)},\ldots,\boldsymbol{o}_{prop}^{(t-k+1)}\right]^T \tag{3}$$

where k is the number of historical timesteps included in the policy observation.

B. Command Space Design and Mathematical Formulation

To enable efficient curriculum learning and prevent interference between different skill components, we design a hierarchically structured command space without mutual dependencies. The command space $\boldsymbol{g} \in \mathcal{G} \subset \mathbb{R}^{n_g}$ is factorized into independent subspaces:

$$\boldsymbol{g}=[\boldsymbol{g}_{loco},\boldsymbol{g}_{torso},\boldsymbol{g}_{arms}]^T \in \mathcal{G}_{loco}\times\mathcal{G}_{torso}\times\mathcal{G}_{arms} \tag{4}$$

where each subspace is defined as follows:

Locomotion Commands: The locomotion commands are represented as $\boldsymbol{g}_{loco}=[\boldsymbol{v}_{xy},\omega_z,h_{pelvis}]^T \in \mathbb{R}^4$, where the components are defined as:

$$\boldsymbol{v}_{xy}=[v_x,v_y]^T \in [-v_{max},v_{max}]^2 \quad \text{(planar velocities)} \tag{5}$$

$$\omega_z \in [-\omega_{max},\omega_{max}] \quad \text{(yaw angular velocity)} \tag{6}$$

$$h_{pelvis} \in [h_{min},h_{max}] \quad \text{(pelvis height)} \tag{7}$$

Torso Orientation Commands: The torso orientation commands are specified as $\boldsymbol{g}_{torso}=[\theta_z,\theta_x,\theta_y]^T \in \mathbb{R}^3$

The torso orientation follows the ZXY Euler angle convention (yaw-roll-pitch), ensuring that the rotation sequence remains within the robot's kinematic constraints. The rotation matrix is computed as:

$$\boldsymbol{R}_{torso}^{cmd}=\boldsymbol{R}_z(\theta_z)\boldsymbol{R}_x(\theta_x)\boldsymbol{R}_y(\theta_y) \tag{8}$$

where:

$$\boldsymbol{R}_z(\theta_z)=\begin{bmatrix}\cos\theta_z & -\sin\theta_z & 0\\ \sin\theta_z & \cos\theta_z & 0\\ 0 & 0 & 1\end{bmatrix} \tag{9}$$

$$\boldsymbol{R}_y(\theta_y)=\begin{bmatrix}\cos\theta_y & 0 & \sin\theta_y\\ 0 & 1 & 0\\ -\sin\theta_y & 0 & \cos\theta_y\end{bmatrix} \tag{10}$$

$$\boldsymbol{R}_x(\theta_x)=\begin{bmatrix}1 & 0 & 0\\ 0 & \cos\theta_x & -\sin\theta_x\\ 0 & \sin\theta_x & \cos\theta_x\end{bmatrix} \tag{11}$$

Arm Joint Commands: The arm joint commands are defined as $\boldsymbol{g}_{arms}=[\boldsymbol{q}_{left},\boldsymbol{q}_{right}]^T \in \mathbb{R}^{n_{arm}}$

Each arm configuration is constrained by joint limits as follows:

$$\boldsymbol{q}_{left},\boldsymbol{q}_{right} \in \prod_{i=1}^{n_{arm}/2}[q_{i,min},q_{i,max}] \tag{12}$$

C. Action Space and Control Interface

The action space $\mathcal{A}$ consists of target joint positions for all actuated degrees of freedom, which can be expressed as follows:

$$\boldsymbol{a}_t=([\boldsymbol{q}_{legs}^{target},\boldsymbol{q}_{torso}^{target},\boldsymbol{q}_{arms}^{target}]^T\cdot\alpha_{scale}+\boldsymbol{q}_{default}) \in \mathbb{R}^{n_j} \tag{13}$$

where $\alpha_{scale}=0.25$ is the action scaling factor and $\boldsymbol{q}_{default}$ represents the default joint positions. For arm control, the policy outputs are combined with desired positions through residual modeling as follows:

$$\boldsymbol{q}_{arms}^{final}=\boldsymbol{a}_{arms}+\boldsymbol{q}_{arms}^{desired} \tag{14}$$

where $\boldsymbol{q}_{arms}^{desired}$ represents the desired arm positions. This residual approach enables fine-grained control adjustments while maintaining stability (detailed in Sect. IV-D).

The actions are executed through a PD controller with feedforward torque compensation, formulated as follows:

$$\boldsymbol{\tau}_t=\boldsymbol{K}_p(\boldsymbol{q}_t^{target}-\boldsymbol{q}_t)-\boldsymbol{K}_d\dot{\boldsymbol{q}}_t \tag{15}$$

where $\boldsymbol{K}_p$ and $\boldsymbol{K}_d$ are diagonal gain matrices.

Fig. 2: Visualization of random sampling of torso rotations and upper body joint positions.

D. Command Space Constraints and Operational Ranges

Each command component operates within carefully defined bounds that ensure physical realizability and safe operation while maximizing the robot's operational capabilities.

Velocity Command Constraints: The planar velocities are constrained within stable limits as follows:

$$v_x \in [-v_{x,max}, v_{x,max}] = [-0.45, 0.55] \text{ m/s} \quad (16)$$

$$v_y \in [-v_{y,max}, v_{y,max}] = [-0.45, 0.45] \text{ m/s} \quad (17)$$

Angular Velocity Constraints: The yaw angular velocity is bounded as follows:

$$\omega_z \in [-\omega_{max}, \omega_{max}] = [-1.2, 1.2] \text{ rad/s} \quad (18)$$

Height Command Range: The pelvis height operates within the kinematic workspace as follows:

$$h_{pelvis} \in [h_{min}, h_{max}] = [0.3, 0.75] \text{ m} \quad (19)$$

The lower bound h_{min} corresponds to the maximum crouch position, while h_{max} represents the fully extended standing height, h_{min} and h_{max} are determined by the leg kinematics and stability considerations.

Torso Orientation Bounds: The torso orientation angles are constrained to maintain balance and prevent kinematic singularities as follows:

$$\theta_z \in [-2.62, 2.62] \quad \text{(yaw)} \quad (20)$$

$$\theta_x \in [-0.52, 0.52] \quad \text{(roll)} \quad (21)$$

$$\theta_y \in [-0.52, 1.57] \quad \text{(pitch)} \quad (22)$$

The asymmetric pitch range reflects the robot's ability to lean forward more than backward due to biomechanical considerations.

Arm Joint Constraints: Each arm joint $q_{arm,i}$ is bounded by mechanical limits, which can be expressed as follows:

$$q_{arm,i} \in [q_{i,min}, q_{i,max}], \quad i = 1, \ldots, n_{arm} \quad (23)$$

where the specific bounds vary per joint according to the robot's mechanical design.

The complete command specifications and operational ranges are detailed in Tab. II and visually displayed in Fig. 2.

IV. UNIFIED LOCO-MANIPULATION CONTROL

We present ULC, a *unified* and *fine-grained* controller for humanoid loco-manipulation that leverages massive parallel reinforcement learning to train a single policy from scratch. Our framework systematically addresses the fundamental challenges of high-dimensional exploration and skill coordination through four key technical innovations:

1) **Sequential skill acquisition** with adaptive curriculum
2) **Command interpolation** with stochastic delay modeling
3) **Load generalization** through dynamic mass distribution and center-of-mass tracking
4) **Residual action modeling** for stable training and precise upper body tracking

The core innovation of ULC lies in its systematic decomposition of the complex loco-manipulation problem into a hierarchy of manageable sub-skills, each governed by carefully designed curriculum learning strategies. This principled approach enables stable learning of high-dimensional behaviors while maintaining robustness to real-world deployment conditions.

A. Sequential Skill Acquisition and Adaptive Curriculum Learning

To address the fundamental challenge of inefficient exploration in high-dimensional command spaces, ULC employs a *sequential skill acquisition strategy* with adaptive command curriculum. The policy progressively masters skills following a carefully designed hierarchical sequence. This sequential approach prevents catastrophic forgetting and ensures robust acquisition of fundamental capabilities before advancing to more complex behaviors.

1) Mathematical Framework for Curriculum Progression: We formalize the curriculum learning process through a structured progression system with rigorous mathematical foundations. Let $\mathcal{T} = \{T_1, T_2, T_3\}$ represent the ordered set of skills to be learned sequentially, where:

$$T_1: \text{ Base velocity tracking} \quad (\boldsymbol{v}_{xy}, \omega_z) \quad (24)$$

$$T_2: \text{ Base height tracking} \quad (h_{\text{pelvis}}) \quad (25)$$

$$T_3: \text{ Torso and arm tracking} \quad (\boldsymbol{g}_{\text{torso}}, \boldsymbol{g}_{\text{arms}}) \quad (26)$$

For each skill T_i, we define a *curriculum parameter* $\alpha_i(t) \in [0, 1]$ that controls the difficulty progression over training time t. The curriculum advancement follows a reward-based gating mechanism that evaluates multiple performance metrics simultaneously:

$$\alpha_i(t+1) = \begin{cases} \min\{1, \alpha_i(t) + \Delta\alpha\} & \text{if } \mathcal{C}_i(t) = \texttt{True} \\ \alpha_i(t) & \text{otherwise} \end{cases} \quad (27)$$

where $\Delta\alpha = 0.05$ represents the curriculum increment. The advancement conditions $\mathcal{C}_i(t)$ are specifically designed based on empirical validation:

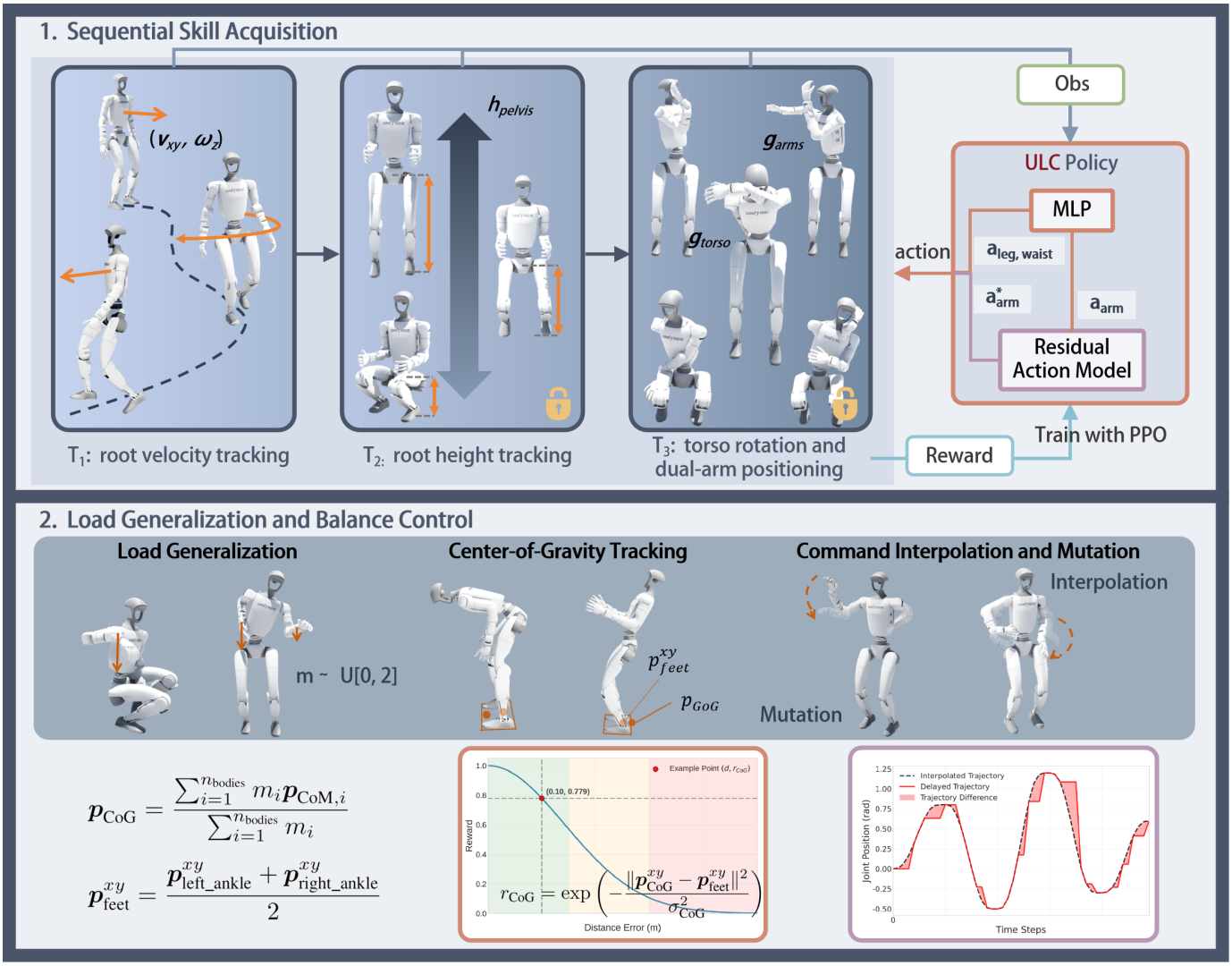

Fig. 3: Method overview of the Unified Loco-Manipulation Controller (ULC). Our approach employs massively parallel reinforcement learning to train a single unified policy that tracks procedurally sampled commands including root velocity, root height, torso orientation, and arm joint positions. The framework addresses multi-task learning challenges through sequential skill acquisition with adaptive curriculum, deployment-realistic command generation with interpolation and random delay, and loaded balance optimization with center-of-mass tracking.

a) Height Curriculum Advancement ($\mathcal{C}_2$): The height curriculum advancement condition implements a multi-criteria evaluation that ensures the robot has mastered fundamental locomotion skills before introducing height variation challenges. The condition is mathematically defined as:

$$\mathcal{C}_2(t) = \mathcal{C}_{\text{height}}(t) \wedge \mathcal{C}_{\text{velocity}}(t) \wedge \mathcal{C}_{\text{hip}}(t) \tag{28}$$

where each component evaluates specific performance metrics with empirically-tuned thresholds:

$$\mathcal{C}_{\text{height}}(t) = R^{\text{avg}}_{\text{height}}(t) \geq 0.85 \cdot w_{\text{height}} \tag{29}$$
$$\mathcal{C}_{\text{velocity}}(t) = R^{\text{avg}}_{\text{vel}}(t) \geq 0.8 \cdot w_{\text{vel}} \tag{30}$$
$$\mathcal{C}_{\text{hip}}(t) = R^{\text{avg}}_{\text{hip}}(t) \geq 0.2 \cdot |w_{\text{hip}}| \tag{31}$$

Here, $R^{\text{avg}}_{\text{height}}(t) = \exp(-|h - h^*|^2/\sigma^2_{\text{height}})$ represents the height tracking reward with weight $w_{\text{height}} = 1.0$, $R^{\text{avg}}_{\text{vel}}(t) = \exp(-\|\boldsymbol{v}_{xy} - \boldsymbol{v}^*_{xy}\|^2/\sigma^2_{\text{vel}})$ denotes the velocity tracking reward with weight $w_{\text{vel}} = 1.0$, and $R^{\text{avg}}_{\text{hip}}(t)$ is the hip deviation penalty with weight $w_{\text{hip}} = -0.15$.

b) Upper Body Curriculum Advancement ($\mathcal{C}_3$): The upper body curriculum advancement implements a comprehensive condition that requires mastery of both arm tracking and torso control capabilities, while simultaneously maintaining all previously acquired skills. The advancement criterion is:

$$\mathcal{C}_3(t) = \mathcal{C}_{\text{upper}}(t) \wedge \mathcal{C}_{\text{torso}}(t) \wedge \mathcal{C}_{\text{prev}}(t) \wedge \mathcal{C}_{\text{complete}}(t) \tag{32}$$

The individual components are rigorously defined as:

$$\mathcal{C}_{\text{upper}}(t) = R^{\text{avg}}_{\text{upper}}(t) \geq 0.7 \cdot w_{\text{upper}} \tag{33}$$
$$\mathcal{C}_{\text{torso}}(t) = R^{\text{avg}}_{\text{torso}}(t) \geq 0.8 \cdot w_{\text{torso}} \tag{34}$$
$$\mathcal{C}_{\text{prev}}(t) = \mathcal{C}_{\text{height}}(t) \wedge \mathcal{C}_{\text{velocity}}(t) \wedge \mathcal{C}_{\text{hip}}(t) \tag{35}$$
$$\mathcal{C}_{\text{complete}}(t) = \alpha_2 \geq 0.98 \tag{36}$$

where $R^{\text{avg}}_{\text{upper}}(t) = \exp(-\|\boldsymbol{q}_{\text{upper}} - \boldsymbol{q}^*_{\text{upper}}\|^2/\sigma^2_{\text{upper}})$ denotes the upper body joint tracking reward with weight $w_{\text{upper}} = 1.0$, and $R^{\text{avg}}_{\text{torso}}(t) = \frac{1}{4}(\exp(-e^2_{\text{yaw}}/\sigma^2_{\text{torso}}) + \exp(-e^2_{\text{roll}}/\sigma^2_{\text{torso}}) + 2 \cdot \exp(-e^2_{\text{pitch}}/\sigma^2_{\text{torso}}))$ represents the torso orientation tracking reward with weight $w_{\text{torso}} = 1.0$.

This multi-criteria gating mechanism ensures that curriculum progression occurs only when all prerequisite skills are sufficiently mastered, thereby preventing catastrophic forgetting and maintaining stable performance across all learned capabilities.

c) Skill-Specific Curriculum Design: **Base Velocity Tracking Curriculum:** The velocity command sampling employs the full range throughout training. The curriculum parameter α_1 does not directly constrain the velocity ranges, but rather the policy gradually learns to handle increasingly complex velocity commands through the reward-based progression system.

Height Tracking Curriculum: The height command range is dynamically adjusted based on curriculum progress. The effective range evolves according to:

$$h^{\text{curr}}_{\text{range}} = \left(h^{\text{orig}}_{\text{min}} + (1 - \alpha_2) \cdot (h^{\text{orig}}_{\text{max}} - h^{\text{orig}}_{\text{min}}), h^{\text{orig}}_{\text{max}}\right) \tag{37}$$

The curriculum starts from the maximum height $h^{\text{orig}}_{\text{max}} = 0.75$ m (standing position) and gradually expands to include crouching positions with minimum height $h^{\text{orig}}_{\text{min}} = 0.3$ m as α_2 increases.

Upper Body Tracking Curriculum: For torso rotation and arm joint position commands, we employ exponential distribution sampling to control movement complexity [19]. The curriculum-adapted sampling is based on inverse transform sampling:

$$r_{\text{upper}} = -\frac{1}{\lambda(\alpha_3)} \ln(1 - u + u \cdot e^{-\lambda(\alpha_3)}) \tag{38}$$

where $u \sim \mathcal{U}(0, 1)$ and the curriculum parameter is:

$$\lambda(\alpha_3) = 20(1 - \alpha_3 \cdot 0.99) \tag{39}$$

The final joint commands are computed as:

$$q_{\text{upper}} = q_{\text{bound}} \cdot r_{\text{upper}} \cdot \text{sign}(\mathcal{N}(0, 1)) \tag{40}$$

where q_{bound} represents the joint limit bounds. This exponential sampling strategy ensures conservative movements initially ($\alpha_3 = 0.05$), progressively expanding to the full joint space as $\alpha_3 \to 1.0$.

d) Curriculum Advancement Algorithm: The complete curriculum learning algorithm integrates all the above components into a unified framework that systematically progresses through skill acquisition stages. The algorithm is formalized in Algorithm 1, which demonstrates the precise interaction between curriculum parameters, reward evaluation, and skill progression logic:

Algorithm 1 ULC Sequential Skill Acquisition with Adaptive Curriculum

Input: Skills $\mathcal{T} = \{T_1, T_2, T_3\}$, reward weights $\{w_{vel}, w_{height}, w_{upper}, w_{torso}, w_{hip}\}$
Initialize: $\alpha_1 \leftarrow 0.05$, $\alpha_2 \leftarrow 0.0$, $\alpha_3 \leftarrow 0.0$, $t \leftarrow 0$
Initialize: Active skills $\mathcal{A} \leftarrow \{T_1\}$, curriculum update interval $I \leftarrow 1000$ steps
while training not converged **do**
  $t \leftarrow t + 1$
  // Sample commands based on current curriculum
  for each skill $T_i \in \mathcal{A}$ **do**
    Sample commands $\boldsymbol{g}_i$ using curriculum parameter α_i
  end for
  // Execute training step
  $\boldsymbol{g} \leftarrow$ Concatenate sampled commands from active skills
  Execute policy $\pi_{\boldsymbol{\theta}}(\boldsymbol{a}_t|\boldsymbol{s}_t, \boldsymbol{g})$
  Compute episode rewards and track running averages
  Update policy parameters $\boldsymbol{\theta}$ using PPO
  // Evaluate curriculum advancement every I steps
  if $t \bmod I = 0$ **then**
    // Height curriculum advancement
    if $\mathcal{C}_2(t)$ and $\alpha_2 < 0.98$ **then**
      $\alpha_2 \leftarrow \min(0.98, \alpha_2 + 0.05)$
      Reset tracked rewards for next evaluation
    end if
    // Upper body curriculum advancement
    if $\mathcal{C}_3(t)$ and $\alpha_3 < 0.98$ **then**
      $\alpha_3 \leftarrow \min(0.98, \alpha_3 + 0.05)$
      Reset tracked rewards for next evaluation
    end if
    // Activate terrain curriculum when both skills mastered
    if $\alpha_2 > 0.98$ and $\alpha_3 > 0.98$ **then**
      Enable terrain level progression
    end if
  end if
end while

B. Stochastic Delay Mechanism and Command Interpolation

To ensure stable arm movements and enhance training robustness, we implement sophisticated command processing mechanisms including quintic polynomial interpolation and stochastic delay modeling that accurately reflects the actual implementation.

a) Quintic Polynomial Interpolation: Upper body commands are smoothly interpolated using quintic polynomial transitions between randomly sampled target positions. The interpolation is executed over a fixed interval $T_{\text{interval}} = 1.0$ s, with the instantaneous target position determined by:

$$\boldsymbol{q}_{\text{target}}(t) = \boldsymbol{q}_{\text{start}} + (\boldsymbol{q}_{\text{goal}} - \boldsymbol{q}_{\text{start}}) \cdot s(t) \tag{41}$$

where $s(t)$ is the quintic smoothing factor:

$$s(t) = 10t^3 - 15t^4 + 6t^5, \quad t \in [0, 1] \tag{42}$$

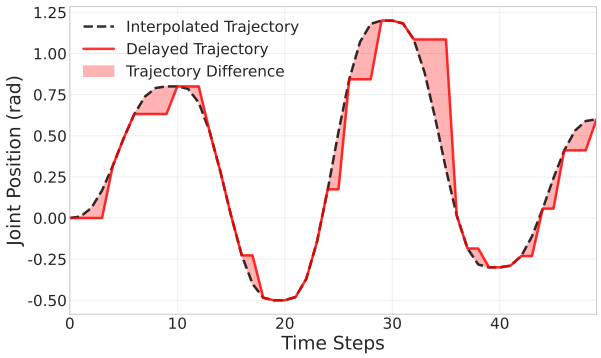

Fig. 4: Illustration of the stochastic delay mechanism for upper body command processing.

The movement step counter t_{step} is normalized as $t = \min(t_{\text{step}}/T_{\text{interval}}, 1.0)$ to ensure smooth transitions. This quintic polynomial ensures C^2 continuity with zero velocity and acceleration at the endpoints, providing natural arm movement characteristics.

b) Stochastic Delay Mechanism: The delay mechanism is implemented through a sophisticated accumulation and release system that operates on the incremental commands between consecutive timesteps. Let $\Delta \boldsymbol{q}^{(t)}$ represent the incremental change in target position at timestep t:

$$\Delta \boldsymbol{q}^{(t)} = \boldsymbol{q}^{(t)}_{target} - \boldsymbol{q}^{(t-1)}_{theoretical} \tag{43}$$

where $\boldsymbol{q}^{(t-1)}_{theoretical}$ is the theoretical position from the previous timestep's interpolation.

Delay Mask and Accumulation: At each timestep, a random delay mask $\boldsymbol{d}^{(t)} \in \{0,1\}^{n_j}$ is generated:

$$d^{(t)}_j \sim \text{Bernoulli}(p_{delay}), \quad j = 1, \ldots, n_j \tag{44}$$

where $p_{delay} = 0.5$ is the fixed delay probability. The accumulation buffer $\boldsymbol{A}^{(t)}$ stores delayed commands:

$$\boldsymbol{A}^{(t)} = \boldsymbol{A}^{(t-1)} \odot \boldsymbol{d}^{(t)} + \Delta \boldsymbol{q}^{(t)} \odot \boldsymbol{d}^{(t)} \tag{45}$$

Command Release: The effective command executed at timestep t combines immediate and delayed components:

$$\Delta \boldsymbol{q}^{(t)}_{\text{effective}} = \Delta \boldsymbol{q}^{(t)} \odot (1 - \boldsymbol{d}^{(t)}) + \boldsymbol{A}^{(t-1)} \odot (1 - \boldsymbol{d}^{(t)}) \tag{46}$$

After execution, the accumulation buffer retains only the still-delayed commands:

$$\boldsymbol{A}^{(t)} := \boldsymbol{A}^{(t-1)} \odot \boldsymbol{d}^{(t)} \tag{47}$$

This mechanism ensures that delayed commands are released as soon as the delay mask permits, maintaining command fidelity while introducing beneficial temporal disturbances. The final desired upper body actions are updated as:

$$\boldsymbol{q}^{(t)}_{\text{desired}} = \boldsymbol{q}^{(t-1)}_{\text{desired}} + \Delta \boldsymbol{q}^{(t)}_{\text{effective}} \tag{48}$$

Fig. 4 shows the characteristic curve of the random delay system intuitively.

C. Load Generalization and Balance Control

To improve robustness to varying payload conditions and maintain dynamic stability, we implement comprehensive load generalization and advanced balance control mechanisms based on the actual implementation.

a) Random Load Distribution: During training, we apply random masses to the robot's wrist to simulate diverse payload conditions. The mass randomization is applied to the robot's wrist masses during environment reset, with the total wrist mass distribution modified to simulate carrying loads.

b) Center-of-Gravity Tracking: We implement a sophisticated center-of-gravity tracking reward that maintains stability across all motion phases. The reward function is formulated as:

$$r_{\text{CoG}} = \exp\left(-\frac{\|\boldsymbol{p}^{xy}_{\text{CoG}} - \boldsymbol{p}^{xy}_{\text{feet}}\|^2}{\sigma^2_{\text{CoG}}}\right) \tag{49}$$

where $\boldsymbol{p}^{xy}_{\text{CoG}}$ is the horizontal projection of the whole-body center of gravity, and $\boldsymbol{p}^{xy}_{\text{feet}}$ represents the midpoint between the ankle positions.

Center-of-Gravity Computation: The whole-body center of gravity is computed using the mass-weighted average of all body segments:

$$\boldsymbol{p}_{\text{CoG}} = \frac{\sum_{i=1}^{n_{\text{bodies}}} m_i \boldsymbol{p}_{\text{CoM},i}}{\sum_{i=1}^{n_{\text{bodies}}} m_i} \tag{50}$$

where m_i and $\boldsymbol{p}_{\text{CoM},i}$ are the mass and center-of-mass position of body segment i, respectively.

Feet Support Reference: The support reference is computed as the midpoint between the ankle positions:

$$\boldsymbol{p}^{xy}_{\text{feet}} = \frac{\boldsymbol{p}^{xy}_{\text{left\_ankle}} + \boldsymbol{p}^{xy}_{\text{right\_ankle}}}{2} \tag{51}$$

This provides a consistent reference point for balance control that accounts for the robot's current stance configuration.

Reward Function Integration: The balance control is integrated into the reward function with a weight of $w_{CoG} = 0.5$ and standard deviation $\sigma_{CoG} = 0.2$ m. This encourages the policy to maintain the center of gravity close to the support base, promoting stable locomotion and manipulation behaviors even under varying load conditions. Fig. 5 illustrates the center-of-gravity tracking reward mechanism.

D. Residual Action Modeling for Arm Control

We introduce a residual action modeling approach for arm joints that enables precise tracking while maintaining training stability. This approach draws inspiration from residual learning principles in robotics [28, 29, 5], providing a principled method for dynamics compensation and stable training.

1) Mathematical Framework: The residual action modeling framework is grounded in additive decomposition, where the final control command combines a base policy output with a residual correction term.

The mathematical formulation distinguishes between the unified policy output and the residual application mechanism. Let $\pi_{\boldsymbol{\theta}}(\boldsymbol{s}, \boldsymbol{g})$ represent the unified policy that outputs actions for all joints. The residual addition is applied post-processing:

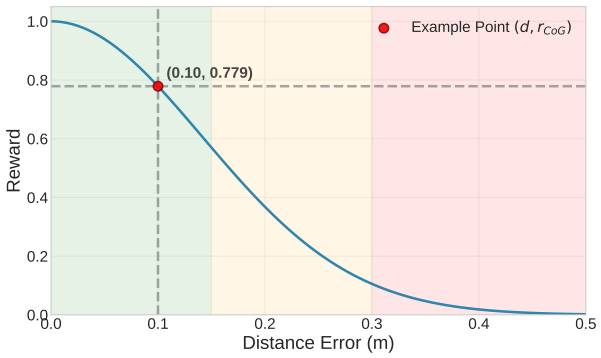

Fig. 5: Illustration of the center-of-gravity tracking reward mechanism.

$$\boldsymbol{q}_{\text{processed}} = \alpha_{scale} \cdot \pi_{\boldsymbol{\theta}}(\boldsymbol{s}, \boldsymbol{g}) + \boldsymbol{q}_{\text{default}} \tag{52}$$

$$\boldsymbol{q}_{\text{final}}[\mathcal{J}_{\text{upper}}] = \boldsymbol{q}_{\text{processed}}[\mathcal{J}_{\text{upper}}] + \boldsymbol{q}_{\text{desired}}[\mathcal{J}_{\text{upper}}] \tag{53}$$

where $\boldsymbol{q}_{\text{default}}$ represents the robot's default joint positions, and $\boldsymbol{q}_{\text{desired}}$ is generated through the command interpolation and delay mechanism described in previous sections.

2) Theoretical Advantages of Residual Action Modeling: The residual term $\boldsymbol{q}_{\text{desired}}$ acts as a feedforward component that compensates for predictable dynamics, particularly gravitational effects on arm joints. This decomposition allows the policy network $\pi_{\boldsymbol{\theta}}$ to focus on learning corrective adjustments rather than reconstructing the entire control signal, significantly reducing the learning complexity.

E. Implementation and Training Details

We implement ULC using massively parallel reinforcement learning and train the policy using Proximal Policy Optimization (PPO). For comprehensive implementation details, including domain randomization parameters, detailed reward function formulation, and complete hyperparameter specifications with network architecture details, please refer to the Appendix. These implementation choices are validated through extensive ablation studies and real-world deployment experiments.

V. TELEOPERATION SYSTEM FOR ULC

The teleoperation system serves as the critical interface between human operators and the humanoid robot, enabling intuitive real-time control through virtual reality and remote control inputs. As shown in Fig. 6, the system acquires data from VR headsets [63, 64] and remote controllers, then processes this data through custom algorithms to generate structured robot commands at 100Hz frequency.

The teleoperation pipeline processes multiple input modalities through five key subsystems: (1) head rotation mapping to torso orientation, (2) head height variation to base height control, (3) wrist position to dual-arm inverse kinematics, (4) finger position to dexterous hand control, and (5) remote controller input to base locomotion commands. This systematic decomposition enables precise and safe human-robot motion transfer while maintaining real-time responsiveness.

A. Head Rotation to Torso Orientation Mapping

The operator's head orientation serves as the primary input for controlling the robot's torso rotation. VR head pose data is acquired and processed through coordinate transformations and safety constraints.

1) Head Pose Processing: The VR system provides head pose data as a 4x4 transformation matrix $\boldsymbol{H}_{\text{head}} \in SE(3)$. The raw head matrix is processed through coordinate transformations:

$$\boldsymbol{H}_{\text{robot}} = \boldsymbol{T}_{\text{robot}}^{\text{openxr}} \boldsymbol{H}_{\text{head}} (\boldsymbol{T}_{\text{robot}}^{\text{openxr}})^{-1} \tag{54}$$

where $\boldsymbol{T}_{\text{robot}}^{\text{openxr}}$ is the calibration transformation matrix between OpenXR and robot coordinate systems.

2) Torso Orientation Extraction: The torso orientation is extracted from the head rotation matrix and constrained for safety:

$$\boldsymbol{R}_{\text{head}} = \boldsymbol{H}_{\text{robot}}[0:3, 0:3] \tag{55}$$

$$[\text{yaw}, \text{pitch}, \text{roll}] = \text{euler}(\boldsymbol{R}_{\text{head}}, \text{'zyx'}) \tag{56}$$

$$\text{yaw}_{\text{clipped}} = \text{clamp}(\text{yaw}, -2.62, 2.62) \tag{57}$$

$$\text{pitch}_{\text{clipped}} = \text{clamp}(\text{pitch}, -0.52, 1.57) \tag{58}$$

$$\text{roll}_{\text{clipped}} = \text{clamp}(\text{roll}, -0.52, 0.52) \tag{59}$$

B. Head Height Variation to Base Height Control

The robot's base height is controlled through VR head height variation to provide intuitive vertical motion control:

$$\Delta h_{\text{head}}(t) = \boldsymbol{H}_{\text{head}}[2,3] - h_{\text{head}}^{\text{ref}} \tag{60}$$

$$h_{\text{pelvis}}(t) = h_{\text{nominal}} + \kappa \cdot \Delta h_{\text{head}}(t) \tag{61}$$

where $h_{\text{nominal}} = 0.75$ m is the nominal pelvis height, $\kappa = 0.5$ is the scaling factor, and the pelvis height is constrained within $[0.3, 0.75]$ m to match training.

C. Wrist Position to Dual-Arm Inverse Kinematics

Hand controller poses are captured as 4x4 transformation matrices and processed through coordinate transformations to compute robot wrist positions.

1) Hand Controller Data Processing: VR hand controllers provide left and right hand poses $\boldsymbol{H}_{\text{hand}}^{L/R} \in SE(3)$. These are transformed through the same coordinate transformation:

$$\boldsymbol{H}_{\text{wrist}}^{L/R} = \boldsymbol{T}_{\text{robot}}^{\text{openxr}} \boldsymbol{H}_{\text{hand}}^{L/R} (\boldsymbol{T}_{\text{robot}}^{\text{openxr}})^{-1} \tag{62}$$

2) Wrist Position Relative to Head Frame: The wrist positions are computed relative to the head frame with specific offsets for left and right arms:

$$\boldsymbol{H}_{\text{rel}}^{L/R} = \boldsymbol{H}_{\text{head}}^{-1} \boldsymbol{H}_{\text{wrist}}^{L/R} \boldsymbol{T}_{\text{unitree}}^{L/R} \tag{63}$$

where $\boldsymbol{T}_{\text{unitree}}^{L/R}$ are the left/right arm-specific transformation matrices. Additional workspace offsets are applied:

$$\boldsymbol{H}_{\text{rel}}^{L/R}[0,3] += 0.15 \text{ m} \tag{64}$$

$$\boldsymbol{H}_{\text{rel}}^{L/R}[2,3] += 0.45 \text{ m} \tag{65}$$

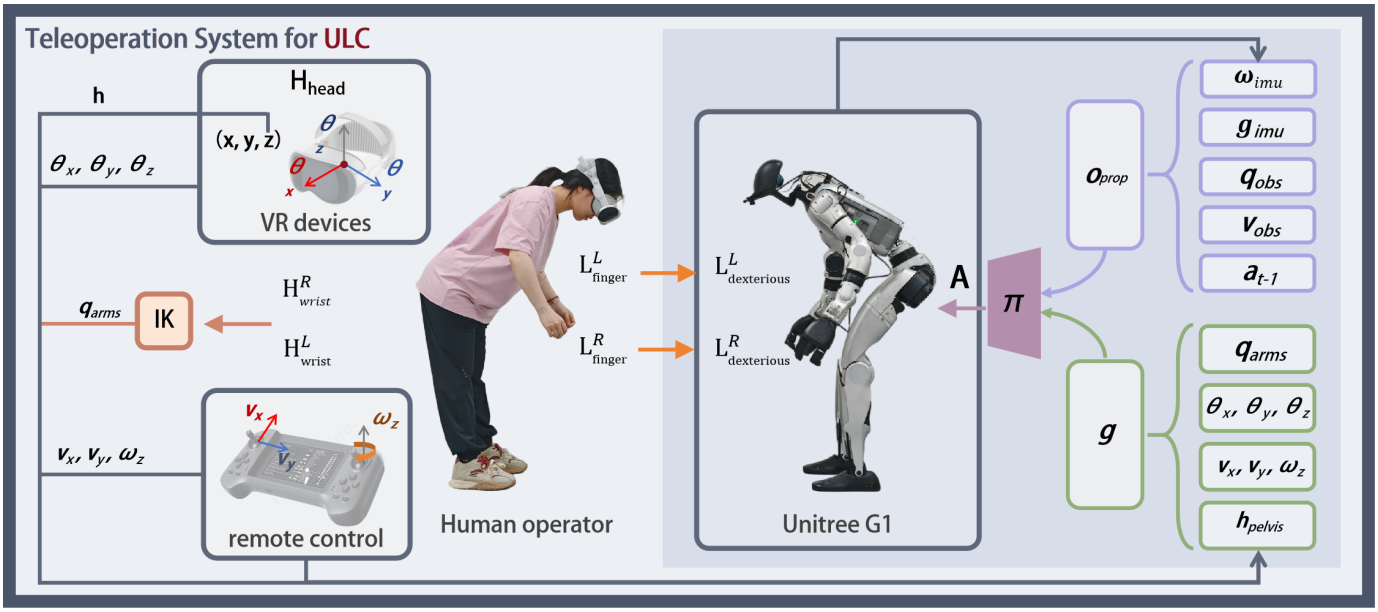

Fig. 6: Schematic of the teleoperation system. The system acquires multimodal inputs from VR headsets and remote controllers, and processes them through custom algorithms to generate structured robot control commands in real time. The pipeline consists of five key subsystems: (1) head rotation to torso orientation mapping, (2) head height variation to base height control, (3) wrist position to dual-arm inverse kinematics, (4) finger position to dexterous hand control, and (5) remote controller input to base locomotion commands. This modular design enables intuitive, precise, and responsive human-robot interaction.

3) Inverse Kinematics: The processed wrist poses are fed to the inverse kinematics solver to compute joint angles for the dual-arm system.

D. Finger Position to Dexterous Hand Control

Finger landmarks for both hands are captured and processed to control the dexterous hand joints through coordinate transformations and retargeting algorithms.

1) Finger Landmark Processing: VR hand tracking provides finger landmarks $\boldsymbol{L}^{L/R}_{\text{finger}} \in \mathbb{R}^{25\times 3}$ for each hand. These landmarks are transformed to robot coordinates and computed relative to the wrist frame:

$$\boldsymbol{L}^{L/R}_{\text{robot}} = \boldsymbol{T}^{\text{openxr}}_{\text{robot}} \boldsymbol{L}^{L/R}_{\text{finger}} \tag{66}$$

$$\boldsymbol{L}^{L/R}_{\text{rel}} = (\boldsymbol{H}^{L/R}_{\text{wrist}})^{-1} \boldsymbol{L}^{L/R}_{\text{robot}} \tag{67}$$

$$\boldsymbol{L}^{L/R}_{\text{dexterous}} = (\boldsymbol{T}_{\text{hand2dexterous}})^T \boldsymbol{L}^{L/R}_{\text{rel}} \tag{68}$$

where $\boldsymbol{T}_{\text{hand2dexterous}}$ transforms from hand coordinate frame to the dexterous hand coordinate frame.

E. Remote Controller Input to Base Locomotion Commands

Remote controller joystick inputs provide intuitive base locomotion control. The controller provides analog stick inputs that are processed and mapped to robot base velocities:

$$\boldsymbol{u}_{\text{joystick}} = [u_x, u_y, u_{\text{rot}}] \in [-1, 1]^3 \tag{69}$$

$$v_x = \text{deadband}(u_x, 0.1) \cdot v_{x,\max} \tag{70}$$

$$v_y = \text{deadband}(u_y, 0.1) \cdot v_{y,\max} \tag{71}$$

$$\omega_z = \text{deadband}(u_{\text{rot}}, 0.1) \cdot \omega_{\max} \cdot 1.2 \tag{72}$$

where deadband$(\cdot)$ eliminates small unintentional inputs. The x and y axes are independently deadbanded and mapped to $v_{x,\max} = 0.55$ m/s and $v_{y,\max} = 0.45$ m/s, respectively, enabling precise omnidirectional velocity control. The angular velocity ω_z is mapped similarly with $\omega_{\max} = 1.2$ rad/s.

F. System Integration and Real-Time Performance

All teleoperation commands are integrated at 100Hz and transmitted to the ULC policy:

$$\boldsymbol{u}_{\text{teleop}} = [v_x, v_y, \omega_z, h_{\text{pelvis}}, \boldsymbol{\theta}_{\text{torso}}, \boldsymbol{q}^L_{\text{arm}}, \boldsymbol{q}^R_{\text{arm}}, \boldsymbol{q}^L_{\text{hand}}, \boldsymbol{q}^R_{\text{hand}}]^T \tag{73}$$

where $\boldsymbol{\theta}_{\text{torso}} = [\text{yaw}_{\text{clipped}}, \text{roll}_{\text{clipped}}, \text{pitch}_{\text{clipped}}]^T \in \mathbb{R}^3$ represents the constrained torso orientation vector.

VI. EXPERIMENT

A. Experiments Setup

We compare the proposed ULC method with state-of-the-art loco-manipulation controllers. The baselines include:

- **HOMIE** [19] A decoupled controller that uses reinforcement learning for leg control and PD control for waist yaw joint and arms.
- **HOMIE-3-DoF-Waist** An evolution of the HOMIE pipeline, with unlocked three waist DoF for PD control
- **FALCON** [18] A decoupled controller that uses dual policy for lower and upper body control with adaptive force curriculum.
- **AMO** [21] A hierarchical humanoid loco-manipulation controller that combines trajectory optimization with reinforcement learning, using a motion adaptation module

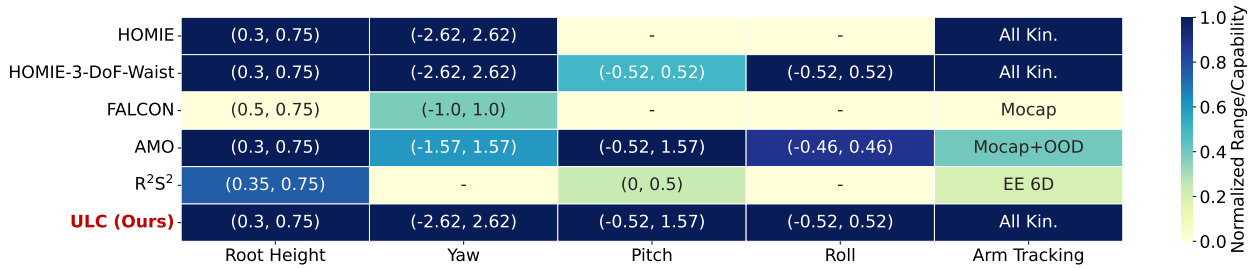

Fig. 7: Comparison of reachable command ranges for different methods. Values show operational limits for root height (m), orientations (rad), and arm control capabilities.

and a tracking controller for leg and waist control, and the arms are controlled by PD.

- **R²S²** [2] A skill-based whole-body controller that uses a pre-trained skill library of primitive motions ensembled into a unified latent space for efficient goal-reaching tasks.

Our metrics include:

- **Root Linear Velocity Tracking Error** E_v
- **Root Angular Velocity Tracking Error** E_ω
- **Root Height Tracking Error** E_h
- **Root Yaw Orientation Tracking Error** E_y
- **Root Pitch Orientation Tracking Error** E_p
- **Root Roll Orientation Tracking Error** E_r
- **Arm Joint Position Tracking Error** E_a

All metrics are computed by rolling out 1024 parallel environments in Isaaclab [65] for 50,000 steps and averaging the tracking errors across all timesteps and environments. This extensive evaluation ensures statistical significance and captures the long-term tracking performance under diverse operational conditions.

B. Comparison of Reachable Workspace

Fig 7 compares the reachable command ranges across different methods, revealing significant variations in workspace capabilities.

HOMIE [19] and **HOMIE-3-DoF-Waist** employ PD control for waist joints, creating a fundamental decoupling between torso rotation and leg control. While this design enables full yaw rotation capability ($\pm$2.62 rad) and maximum root height range (0.3, 0.75 m), the legs cannot actively participate in torso orientation control, severely constraining the integrated whole-body workspace. HOMIE-3-DoF-Waist extends to pitch and roll control but remains limited to symmetric ranges ($\pm$0.52 rad) with PD contorled waist joints. **FALCON** [18] prioritizes adaptive force curriculum learning for precision under external loads, but this focus comes at the cost of workspace coverage. The method suffers from restricted yaw range ($\pm$1.0 rad) and elevated minimum root height (0.5 m), limiting low-reaching capabilities. More critically, FALCON's dual-arm control relies entirely on motion capture data, creating a fundamental bottleneck for generalization. This mocap dependency severely limits robustness under out-of-distribution (OOD) commands that deviate from pre-recorded human demonstrations. **AMO** [21] addresses the OOD limitation through its motion adaptation module, demonstrating superior robustness beyond mocap constraints. By unifying waist and leg control in a single model, AMO successfully unlocks torso rotation capabilities and achieves asymmetric pitch control (-0.52, 1.57 rad), enabling both downward manipulation and upward reaching. However, the method's workspace remains constrained in roll orientation ($\pm$0.46 rad) and arm control versatility compared to kinematic-based approaches. **R²S²** [2] utilizes a pre-defined skill library compressed into a unified latent space for systematic goal-reaching. While this approach enables efficient skill composition and maintains reasonable root height control (0.35, 0.75 m), the reliance on pre-defined primitives severely constrains torso rotation capabilities. The method achieves only minimal pitch control (0, 0.5 rad) and completely lacks yaw and roll capabilities, fundamentally limiting whole-body coordination. ULC overcomes these limitations through unified coordinated control that integrates all degrees of freedom. Our approach achieves maximum root height range (0.3, 0.75 m) enabling both low-squatting and high-reaching motions, complete torso rotation tracking across all axes including full yaw rotation ($\pm$2.62 rad) matching the best existing capability, asymmetric pitch control (-0.52, 1.57 rad) optimized for both downward looking and upward reaching tasks, and enhanced roll stability ($\pm$0.52 rad) for lateral manipulation. The procedurally sampled dual-arm control strategy unlocks all kinematic degrees of freedom without mocap constraints, providing unprecedented manipulation versatility.

C. Comparison of Tracking Accuracy

Fig. 8 evaluates tracking performance across four scenarios with commands sampled within each method's operational ranges from Tab. 7: (1) **Whole command space**: entire operational workspace; (2) **Edge command space**: extreme torso rotation cases; (3) **Wrist loaded**: 2kg external loads on both wrists; (4) **Command mutation**: random command delays IV-B (probability 0.5) during execution.

Locomotion Control Analysis: AMO demonstrates exceptional linear velocity tracking ($E_v = 0.039 \pm 0.008$ m/s) and angular velocity control ($E_\omega = 0.061 \pm 0.011$ rad/s) in the whole command space, attributed to its hierarchical design combining trajectory optimization with RL tracking controllers. This hybrid approach enables precise motion planning that optimizes locomotion dynamics. ULC achieves competitive performance ($E_v = 0.068 \pm 0.012$ m/s, $E_\omega = 0.127 \pm 0.018$ rad/s) through unified whole-body control, while

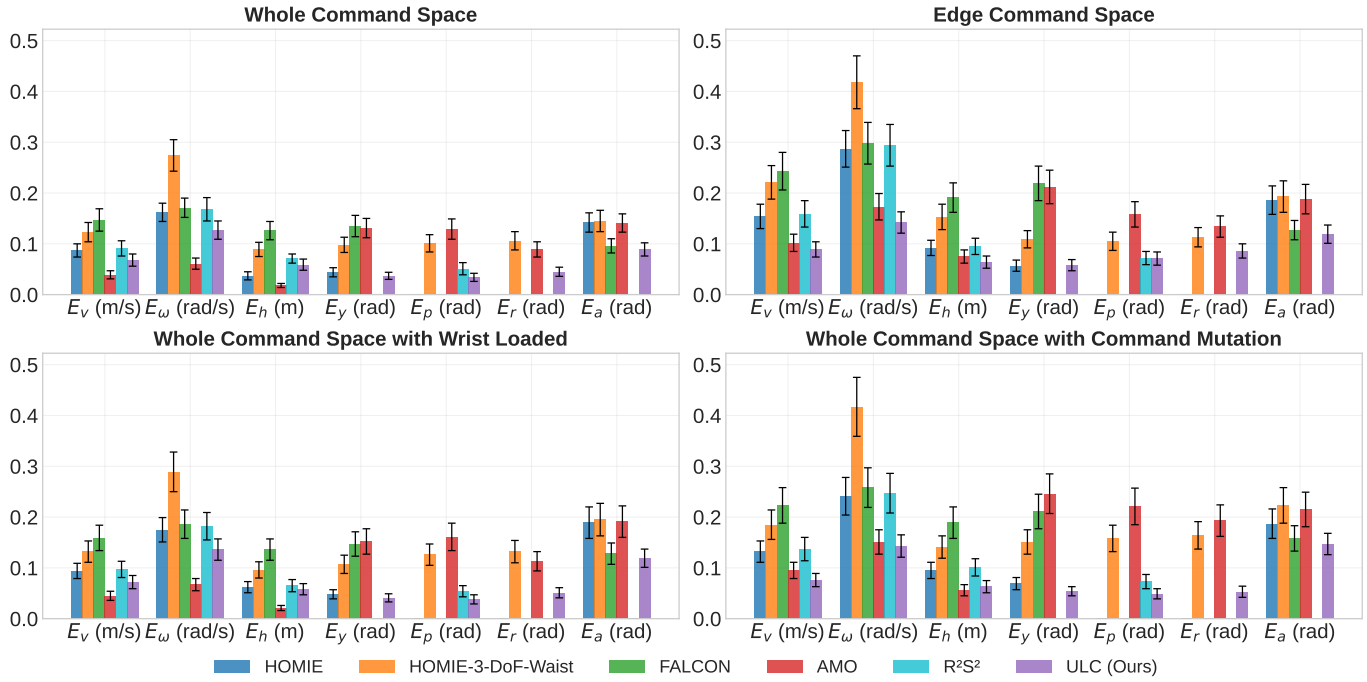

Fig. 8: Comparison of tracking accuracy for different methods across four representative scenarios. The figure shows the main tracking errors and standard deviations under whole command space, edge command space, wrist loaded (2kg), and command mutation (random delay) conditions. ULC consistently achieves superior overall tracking performance and robustness compared to state-of-the-art baselines.

HOMIE and FALCON show moderate performance due to their decoupled leg-arm architectures.

Torso Orientation Control: ULC, HOMIE-3-DoF-Waist and AMO support full 3-DoF torso control capabilities. ULC excels in yaw tracking ($E_y = 0.037 \pm 0.007$ rad) and achieves superior pitch ($E_p = 0.034 \pm 0.008$ rad) and roll ($E_r = 0.045 \pm 0.009$ rad) control. AMO shows competitive yaw performance ($E_y = 0.131 \pm 0.019$ rad) but exhibits higher pitch ($E_p = 0.129 \pm 0.020$ rad) and roll ($E_r = 0.089 \pm 0.015$ rad) errors due to the complexity of coordinating trajectory optimization with RL tracking. The waist degree of freedom of HOMIE-3-DoF-Waist is completely controlled by PD, so it performs poorly in tracking accuracy. HOMIE and FALCON completely lack pitch/roll control capabilities, while R²S² provides only limited pitch control.

Dual-Arm Tracking Performance: HOMIE and AMO rely on PD controllers for arm tracking, resulting in similar performance levels (HOMIE: $E_a = 0.142 \pm 0.019$ rad, AMO: $E_a = 0.141 \pm 0.018$ rad). FALCON's dedicated upper-body RL policy achieves better arm tracking ($E_a = 0.096 \pm 0.014$ rad) through learned force adaptation, but remains constrained by mocap dependency. ULC outperforms all methods ($E_a = 0.089 \pm 0.013$ rad) through residual action modeling and sequential skill acquisition, enabling precise arm control without mocap constraints.

Robustness Under Extreme Conditions: In edge command space scenarios, architectural differences become pronounced. HOMIE-3-DoF-Waist suffers severe degradation ($E_v = 0.221 \pm 0.033$ m/s, $E_\omega = 0.418 \pm 0.052$ rad/s) due to inadequate coordination between PD-controlled torso and RL-controlled legs. FALCON's dual-policy architecture struggles with extreme conditions ($E_v = 0.243 \pm 0.037$ m/s). ULC maintains robust performance across all metrics through unified control architecture.

External Load Adaptation: Under 2kg wrist loads, AMO maintains its locomotion advantage ($E_v = 0.045 \pm 0.009$ m/s) due to trajectory optimization's ability to adapt to changing dynamics. However, PD-based arm control in HOMIE and AMO shows degradation under external loads, with HOMIE's arm tracking error increasing to $E_a = 0.189 \pm 0.031$ rad. FALCON's force-adaptive curriculum provides some load robustness ($E_a = 0.128 \pm 0.021$ rad), but ULC's residual action modeling achieves superior load adaptation across all metrics ($E_v = 0.072 \pm 0.013$ m/s, $E_a = 0.119 \pm 0.018$ rad) while maintaining precise orientation control ($E_y = 0.041 \pm 0.008$ rad, $E_p = 0.038 \pm 0.009$ rad).

Command Mutation Robustness: Under stochastic command delays, ULC demonstrates superior robustness with minimal performance degradation ($E_v = 0.076 \pm 0.013$ m/s, $E_a = 0.147 \pm 0.021$ rad), while other methods show significant deterioration. AMO experiences substantial torso control degradation ($E_y = 0.246 \pm 0.039$ rad, $E_p = 0.221 \pm 0.036$ rad) due to trajectory optimization sensitivity to timing variations. HOMIE and HOMIE-3-DoF-Waist suffers severe performance loss across all metrics, confirming that the PD controller is susceptible to sudden disturbances.

D. Ablation on Policy Training

Ablation studies are conducted to systematically evaluate the contribution of each key component in the ULC training pipeline. We consider four variants by removing one module

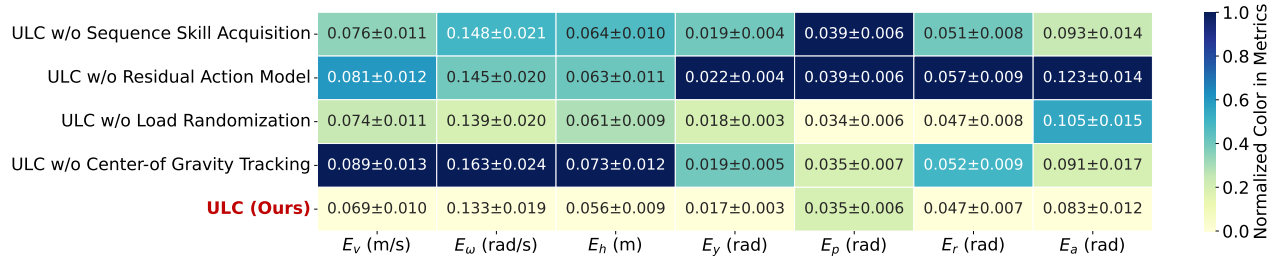

Fig. 9: Quantitative results of ablation studies for key components in the ULC training pipeline. The column-normalized heatmap shows that lower values indicate better tracking performance across all metrics.

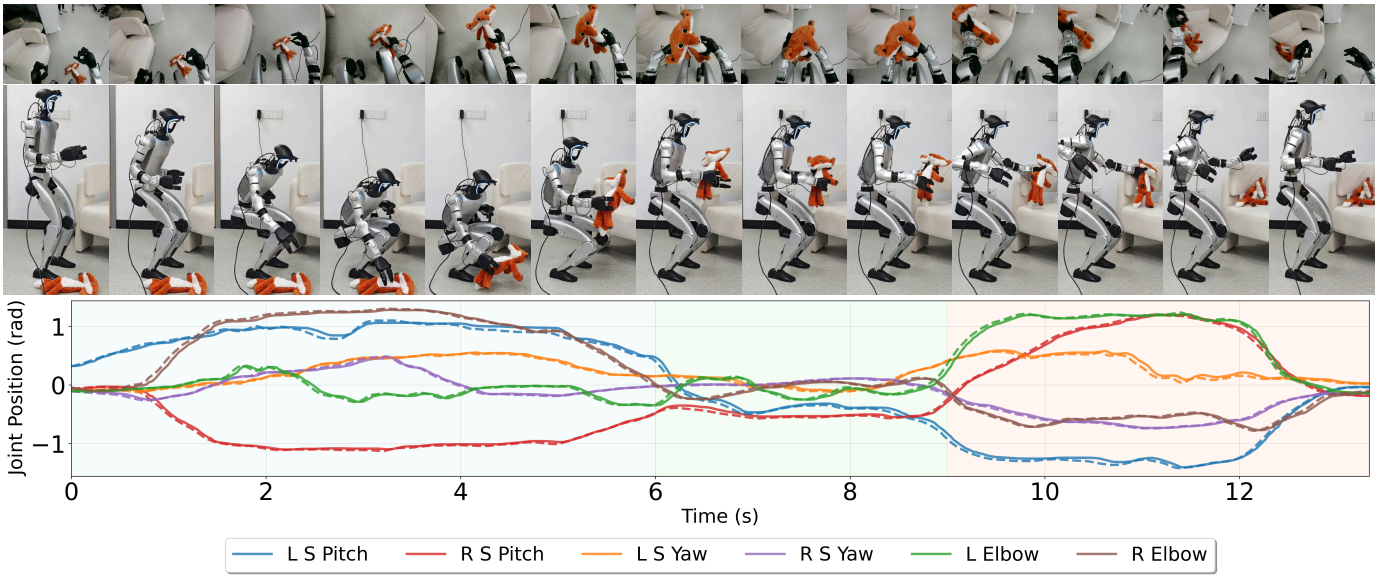

Fig. 10: Time-series visualization of the doll pick-and-place task. Multiple sub-images illustrate the sequential execution process, covering all key stages: squatting to pick up the doll, hand switching, and placing the doll at the target location.

at a time: Sequence Skill Acquisition, Residual Action Model, Load Randomization, and Center-of-Gravity Tracking. The quantitative results are visualized in Fig. 9 as a column-normalized heatmap, where lower values indicate better tracking performance across all metrics. Evaluation was performed with a wrist load of 2 kg.

Removing Sequence Skill Acquisition leads to a noticeable increase in all metrics, highlighting the importance of progressive skill composition for precise whole-body coordination. Excluding the Residual Action Model results in higher errors in both arm ($E_a = 0.123 \pm 0.014$ rad) and root tracking ($E_v = 0.081 \pm 0.012$ m/s), confirming that residual learning is critical for fine-grained motion adaptation. Without Load Randomization, the model exhibits increased sensitivity to external disturbances, as reflected by higher errors in all metrics, especially in arm tracking ($E_a = 0.105 \pm 0.015$ rad), demonstrating the necessity of diverse training for generalization. Omitting Center-of-Gravity Tracking causes the most significant performance drop in root-related metrics ($E_v = 0.089 \pm 0.013$ m/s, $E_\omega = 0.163 \pm 0.024$ rad/s), indicating that explicit CoG supervision is essential for stable locomotion and orientation control.

The full ULC model (Ours) consistently achieves the lowest errors across all metrics ($E_v = 0.069 \pm 0.010$ m/s, $E_\omega = 0.133 \pm 0.019$ rad/s, $E_h = 0.056 \pm 0.009$ m, $E_y = 0.017 \pm 0.003$ rad, $E_p = 0.035 \pm 0.006$ rad, $E_r = 0.047 \pm 0.007$ rad, $E_a = 0.083 \pm 0.012$ rad), validating the effectiveness of the unified training strategy. These results demonstrate that each component is indispensable for achieving robust, high-precision loco-manipulation, and their integration yields significant performance gains over ablated variants.

E. Real World Results

We evaluate the performance of ULC in real-world scenarios to validate how its height control, torso rotation capabilities, and dual-arm tracking precision contribute to practical task performance. We design a series of challenging manipulation scenarios that require precise whole-body coordination.

1) ***Teleoperation Results****:* To demonstrate ULC's effectiveness as a low-level controller in practical applications, we evaluate two representative teleoperation scenarios that require coordinated locomotion and manipulation, as illustrated in Fig. 10 and Fig. 11.

Pick and place the doll on the sofa: This task evaluates ULC's ability to perform coordinated whole-body manipulation through three key steps: (1) squatting down and

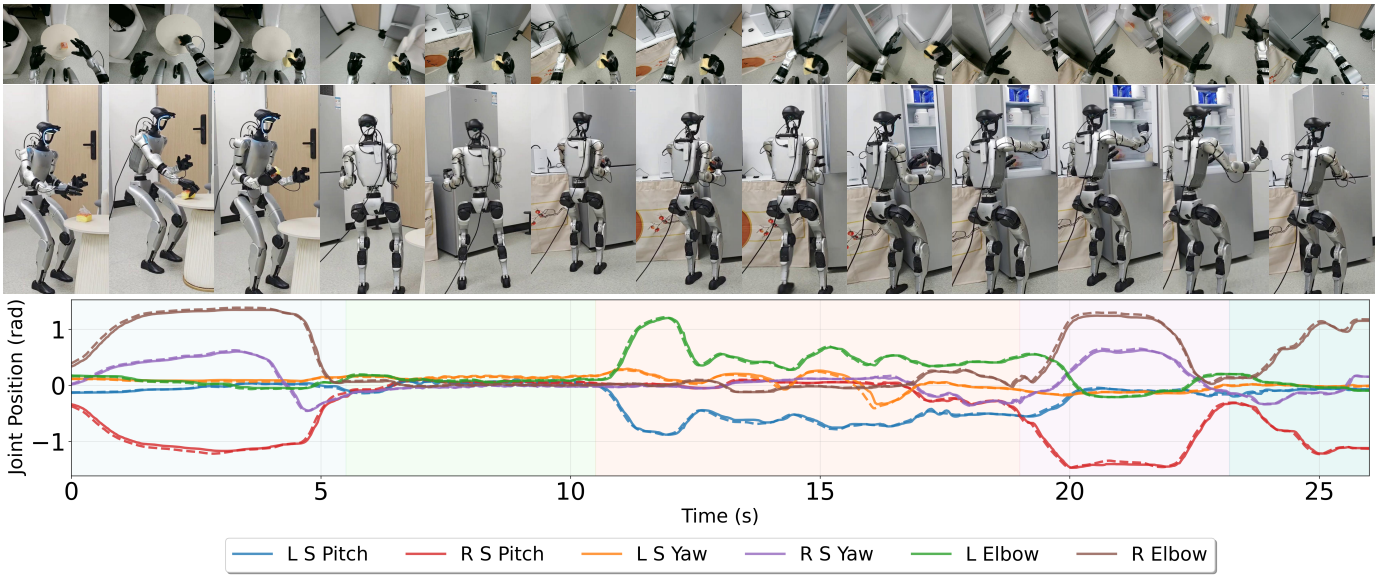

Fig. 11: Time-series visualization of the refrigerator task. Multiple sub-images illustrate the sequential execution process, covering all five stages: picking up the bread, walking to the refrigerator, opening the door, placing the bread inside, and closing the door.

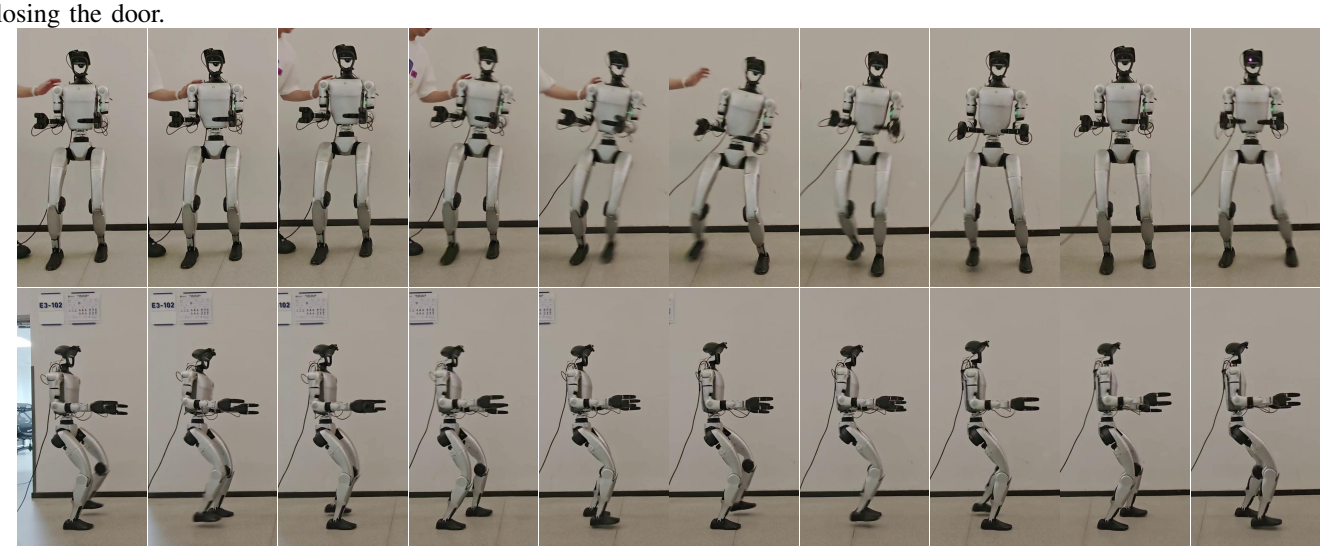

Fig. 12: Time-series visualization of the push and walk experiments. The images illustrate the robot's performance under lateral push disturbances and during walking, demonstrating the robustness and stability of the proposed controller in dynamic real-world scenarios.

grasping the doll on the floor using precise height control and torso pitch adjustment; (2) standing up and pass the doll to the other hand; (3) placing the doll onto the sofa at a designated target location. As shown in Fig. 10, the execution sequence demonstrates ULC's locomotion stability during complex height transitions and its ability to maintain dynamic balance while executing coordinated arm movements across different body configurations. The figure also presents the tracking curves for selected joints, where the solid line represents the actual state and the dashed line represents the expected state from IK, with the dual-arm tracking error E_a of 0.092 rad throughout the entire task execution phase.

Put the bread in the refrigerator: This task exemplifies ULC's ability to execute a complex, multi-step manipulation sequence that integrates spatial navigation, dual-arm coordination, and dynamic interaction with the environment. The robot must complete the following five steps: (1) use its right hand to grasp the bread from the table, requiring precise arm positioning and stable grasp control; (2) maintain a secure hold on the bread while turning and walking to the refrigerator, demonstrating robust locomotion and object stability during whole-body movement; (3) open the refrigerator door with the left hand, which involves coordinated dual-arm manipulation and balance maintenance as one arm continues to hold the bread; (4) place the bread inside the refrigerator, a step that demands accurate and careful placement in a confined space; (5) after releasing the bread, use the right hand to close the refrigerator door, requiring the robot to re-engage the right arm for environmental interaction. Throughout this process, ULC demonstrates highly coordinated dual-arm and whole-

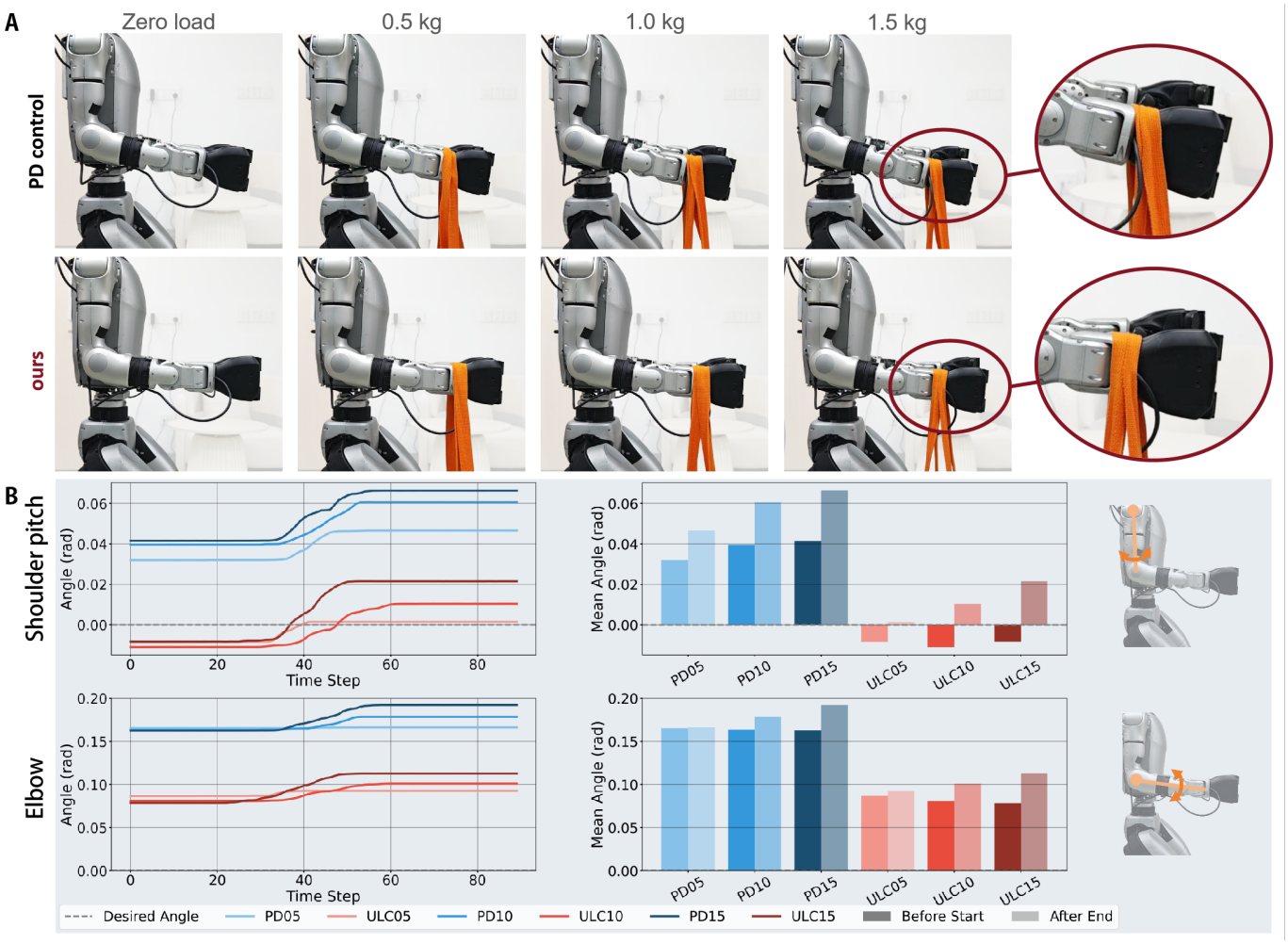

Fig. 13: Comparison of joint angle tracking errors for ULC and traditional PD control (gains: $K_p = 80$, $K_d = 3$) under different external loads (0.5 kg, 1.0 kg, 1.5 kg) in real-world experiments. ULC consistently achieves lower errors than PD control at all load levels, demonstrating superior force adaptation and robustness to external disturbances.

body control, maintaining stable manipulation and precise object handling, real-world environments. Fig. 11 presents the full teleoperated process, showcasing the system's ability to achieve smooth, accurate, and robust execution across all steps. The consistently low arm tracking error E_a (0.103 rad) throughout the task further highlights ULC's precision and reliability in practical force interactive loco-manipulation scenarios.

*2) **Case Study: Robust Stand and Walk**:* To qualitatively evaluate the robustness and stability of ULC in real-world scenarios, we conduct a series of stand and walk experiments under challenging conditions. During the standing tests, the robot is subjected to strong pushes from the left, right, front, and back while maintaining an upright posture with the upper body target angles set to zero. Notably, even when the robot is pushed forcefully from any direction, it is able to maintain balance and quickly recover to its original pose, demonstrating the effectiveness of the center-of-gravity (CoG) tracking module. The robot's ability to resist disturbances without excessive upper body sway or instability highlights the superior whole-body coordination enabled by ULC.

For the walking experiments, the robot is commanded to walk long distances in a straight line. Importantly, at the moment of gait initiation, the robot does not exhibit any forward lean in the upper body, which is a direct benefit of explicit CoG tracking. This allows for natural and stable walking without the need for pre-leaning or compensatory motions.

Fig. 12 presents time-series visualizations of the robot being pushed from the side and walking forward. The images illustrate the robot's rapid recovery from external disturbances and its stable, confident gait during long-distance walking. These results confirm that ULC provides reliable and robust whole-body control for both static and dynamic tasks, instilling strong confidence in its real-world deployment.

*3) **Real world Loaded Comparison**:* We conduct a controlled experiment comparing ULC with traditional PD control under external wrist loads of 0.5 kg, 1.0 kg, and 1.5 kg. ULC and PD controller share the same PD parameters (PD gains: $K_p = 80$, $K_d = 3$). Both methods are required to maintain dual-arm poses with target joint angles set to zero (forearms parallel to the ground). The tracking errors under each load

condition are visualized in Fig. 13.

Across all load levels, ULC consistently achieves lower joint angle deviations than PD control. Notably, even at the highest load of 1.5 kg, ULC maintains high tracking accuracy, while PD control exhibits significant errors due to inadequate gravity compensation. This performance gap is evident at every tested load (0.5 kg, 1.0 kg, 1.5 kg), where ULC's learned dynamics naturally incorporate force adaptation, resulting in superior robustness and precision. In contrast, PD control struggles to maintain parallel positioning even without load, and its errors increase substantially as the load increases. These results validate the advantage of ULC in real-world manipulation tasks requiring reliable force adaptation and precise tracking under varying external disturbances.

VII. Conclusions and Limitations

We presented ULC, a unified controller for humanoid loco-manipulation that, to the best of our knowledge, is the first to simultaneously achieve unified whole-body control, large operational workspace, and high-precision tracking. By integrating all degrees of freedom in a single controller and leveraging principled procedural command sampling, ULC enables robust and versatile performance across a diverse set of tasks and challenging scenarios. Extensive experiments demonstrate that ULC outperforms prior decoupled or mocap-based methods in tracking accuracy, workspace coverage, and robustness, establishing a new foundation for practical, deployable humanoid loco-manipulation systems. Ablation studies further confirm that each component of our framework is essential.

Despite these advances, ULC still has a key limitation: the use of simplified locomotion commands precludes the generation of complex leg patterns achievable through motion capture approaches. Addressing this limitation and further enhancing locomotion expressiveness and generalization to even more complex real-world tasks will be the focus of future work.

References

[1] Z. Zhuang and H. Zhao, *Embrace Collisions: Humanoid Shadowing for Deployable Contact-Agnostics Motions*, Feb. 2025. arXiv: 2502.01465 [cs].

[2] Z. Zhang *et al.*, *Unleashing Humanoid Reaching Potential via Real-world-Ready Skill Space*, May 2025. arXiv: 2505.10918 [cs].

[3] W. Sun, B. Cao, L. Chen, Y. Su, Y. Liu, and Z. Xie, "Learning Perceptive Humanoid Locomotion over Challenging Terrain,"

[4] T. Huang *et al.*, *Learning Humanoid Standing-up Control across Diverse Postures*, Feb. 2025. arXiv: 2502.08378 [cs].

[5] T. He *et al.*, *ASAP: Aligning Simulation and Real-World Physics for Learning Agile Humanoid Whole-Body Skills*, Feb. 2025. arXiv: 2502.01143 [cs].

[6] A. Allshire *et al.*, *Visual Imitation Enables Contextual Humanoid Control*, May 2025. arXiv: 2505.03729 [cs].

[7] R.-Z. Qiu *et al.*, *Humanoid Policy ˜ Human Policy*, Mar. 2025. arXiv: 2503.13441 [cs].

[8] T. Lin, K. Sachdev, L. Fan, J. Malik, and Y. Zhu, *Sim-to-Real Reinforcement Learning for Vision-Based Dexterous Manipulation on Humanoids*, Feb. 2025. arXiv: 2502.20396 [cs].

[9] Y. Jiang *et al.*, *BEHAVIOR Robot Suite: Streamlining Real-World Whole-Body Manipulation for Everyday Household Activities*, Mar. 2025. arXiv: 2503.05652 [cs].

[10] Z. Wei *et al.*, "D(R, O) Grasp: A Unified Representation of Robot and Object Interaction for Cross-Embodiment Dexterous Grasping,"

[11] C. Chen, Z. Yu, H. Choi, M. Cutkosky, and J. Bohg, *DexForce: Extracting Force-informed Actions from Kinesthetic Demonstrations for Dexterous Manipulation*, Jan. 2025. arXiv: 2501.10356 [cs].

[12] C. Chi *et al.*, *Diffusion policy: Visuomotor policy learning via action diffusion*, 2024. arXiv: 2303.04137 [cs.RO].

[13] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn, *Learning fine-grained bimanual manipulation with low-cost hardware*, 2023. arXiv: 2304.13705 [cs.RO].

[14] K. Black *et al.*, π_0*: A vision-language-action flow model for general robot control*, 2024. arXiv: 2410.24164 [cs.LG].

[15] P. Intelligence *et al.*, $\pi_{0.5}$*: A vision-language-action model with open-world generalization*, 2025. arXiv: 2504.16054 [cs.LG].

[16] NVIDIA *et al.*, *Gr00t n1: An open foundation model for generalist humanoid robots*, 2025. arXiv: 2503.14734 [cs.RO].

[17] Y. Xue, W. Dong, M. Liu, W. Zhang, and J. Pang, *A Unified and General Humanoid Whole-Body Controller for Fine-Grained Locomotion*, Feb. 2025. arXiv: 2502.03206 [cs].

[18] Y. Zhang *et al.*, *FALCON: Learning Force-Adaptive Humanoid Loco-Manipulation*, May 2025. arXiv: 2505.06776 [cs].

[19] Q. Ben, F. Jia, J. Zeng, J. Dong, D. Lin, and J. Pang, *HOMIE: Humanoid Loco-Manipulation with Isomorphic Exoskeleton Cockpit*, Feb. 2025. arXiv: 2502.13013 [cs].

[20] T. He *et al.*, "HOVER: Versatile Neural Whole-Body Controller for Humanoid Robots," *arXiv preprint arXiv:2410.21229*, 2024. arXiv: 2410.21229.

[21] J. Li, X. Cheng, T. Huang, S. Yang, R.-Z. Qiu, and X. Wang, *AMO: Adaptive Motion Optimization for Hyper-Dexterous Humanoid Whole-Body Control*, May 2025. arXiv: 2505.03738 [cs].

[22] Z. Ding *et al.*, *JAEGER: Dual-Level Humanoid Whole-Body Controller*, May 2025. arXiv: 2505.06584 [cs].

[23] T. He *et al.*, "OmniH2O: Universal and Dexterous Human-to-Humanoid Whole-Body Teleoperation and Learning," *arXiv preprint arXiv:2406.08858*, 2024. arXiv: 2406.08858.

[24] Z. Fu, Q. Zhao, Q. Wu, G. Wetzstein, and C. Finn, "HumanPlus: Humanoid Shadowing and Imitation from Humans," *arXiv preprint arXiv:2406.10454*, 2024. arXiv: 2406.10454.

[25] X. Cheng, Y. Ji, J. Chen, R. Yang, G. Yang, and X. Wang, "Expressive whole-body control for humanoid robots," *arXiv preprint arXiv:2402.16796*, 2024. arXiv: 2402.16796.

[26] M. Ji *et al.*, *ExBody2: Advanced Expressive Humanoid Whole-Body Control*, Dec. 2024. arXiv: 2412.13196 [cs].

[27] O. Sener and V. Koltun, "Multi-task learning as multi-objective optimization," *CoRR*, vol. abs/1810.04650, 2018. arXiv: 1810.04650.

[28] T. Silver, K. Allen, J. Tenenbaum, and L. Kaelbling, "Residual policy learning," *arXiv preprint arXiv:1812.06298*, 2018.

[29] T. Johannink *et al.*, "Residual reinforcement learning for robot control," in *2019 international conference on robotics and automation (ICRA)*, IEEE, 2019, pp. 6023–6029.

[30] Z. Luo, J. Cao, A. Winkler, K. Kitani, and W. Xu, *Perpetual Humanoid Control for Real-time Simulated Avatars*, Sep. 2023. arXiv: 2305.06456 [cs].

[31] C. Packer, K. Gao, J. Kos, P. Krähenbühl, V. Koltun, and D. Song, *Assessing generalization in deep reinforcement learning*, 2019. arXiv: 1810.12282 [cs.LG].

[32] X. Gu *et al.*, "Advancing humanoid locomotion: Mastering challenging terrains with denoising world model learning," *arXiv preprint arXiv:2408.14472*, 2024. arXiv: 2408.14472.

[33] X. Gu, Y.-J. Wang, and J. Chen, "Humanoid-Gym: Reinforcement Learning for Humanoid Robot with Zero-Shot Sim2Real Transfer," *arXiv preprint arXiv:2404.05695*, 2024. arXiv: 2404.05695.

[34] T. Zhang *et al.*, *HuB: Learning Extreme Humanoid Balance*, May 2025. arXiv: 2505.07294 [cs].

[35] I. Radosavovic, S. Kamat, T. Darrell, and J. Malik, "Learning Humanoid Locomotion over Challenging Terrain," *arXiv:2410.03654*, 2024. arXiv: 2410.03654.

[36] I. Radosavovic *et al.*, "Humanoid locomotion as next token prediction," *arXiv preprint arXiv:2402.19469*, 2024. arXiv: 2402.19469.

[37] I. Radosavovic, T. Xiao, B. Zhang, T. Darrell, J. Malik, and K. Sreenath, "Real-World Humanoid Locomotion with Reinforcement Learning," *arXiv:2303.03381*, 2023. arXiv: 2303.03381.

[38] J. Long *et al.*, "Learning Humanoid Locomotion with Perceptive Internal Model," *arXiv preprint arXiv:2411.14386*, 2024. arXiv: 2411.14386.

[39] Z. Chen *et al.*, *Learning Smooth Humanoid Locomotion through Lipschitz-Constrained Policies*, Oct. 2024. arXiv: 2410.11825 [cs].

[40] N. Rudin, D. Hoeller, P. Reist, and M. Hutter, "Learning to walk in minutes using massively parallel deep reinforcement learning," in *Conference on Robot Learning*, PMLR, 2022, pp. 91–100.
[41] G. B. Margolis and P. Agrawal, "Walk these ways: Tuning robot control for generalization with multiplicity of behavior," in *Conference on Robot Learning*, PMLR, 2023, pp. 22–31.
[42] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Science robotics*, vol. 5, no. 47, eabc5986, 2020.
[43] Z. Li *et al.*, "Reinforcement learning for robust parameterized locomotion control of bipedal robots," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2021, pp. 2811–2817.
[44] Y. Li *et al.* "Hold My Beer: Learning Gentle Humanoid Locomotion and End-Effector Stabilization Control." arXiv: 2505.24198 [cs]. (Jun. 3, 2025), [Online]. Available: http://arxiv.org/abs/2505.24198 (visited on 07/04/2025), pre-published.
[45] J. Mao *et al.*, *Learning from Massive Human Videos for Universal Humanoid Pose Control*, Dec. 2024. arXiv: 2412.14172 [cs].
[46] T. He *et al.*, "Learning human-to-humanoid real-time whole-body teleoperation," *arXiv preprint arXiv:2403.04436*, 2024. arXiv: 2403.04436.
[47] P. Dugar, A. Shrestha, F. Yu, B. van Marum, and A. Fern, "Learning multi-modal whole-body control for real-world humanoid robots," *arXiv preprint arXiv:2408.07295*, 2024. arXiv: 2408.07295.
[48] X. Cheng, Y. Ji, J. Chen, R. Yang, G. Yang, and X. Wang, *Expressive Whole-Body Control for Humanoid Robots*, Mar. 2024. arXiv: 2402.16796 [cs].
[49] Z. Luo *et al.*, *Universal Humanoid Motion Representations for Physics-Based Control*, Apr. 2024. arXiv: 2310.04582 [cs].
[50] C. Tessler, Y. Guo, O. Nabati, G. Chechik, and X. B. Peng, *MaskedMimic: Unified Physics-Based Character Control Through Masked Motion Inpainting*, https://arxiv.org/abs/2409.14393v1, Sep. 2024.
[51] X. B. Peng, Z. Ma, P. Abbeel, S. Levine, and A. Kanazawa, "AMP: Adversarial Motion Priors for Stylized Physics-Based Character Control," *ACM Trans. Graph.*, vol. 40, no. 4, Jul. 2021.
[52] S. Xu, H. Y. Ling, Y.-X. Wang, and L.-Y. Gui, *InterMimic: Towards Universal Whole-Body Control for Physics-Based Human-Object Interactions*, Feb. 2025. arXiv: 2502.20390 [cs].
[53] J. Liu *et al.*, *Human-Humanoid Robots Cross-Embodiment Behavior-Skill Transfer Using Decomposed Adversarial Learning from Demonstration*, Dec. 2024. arXiv: 2412.15166 [cs].
[54] S. Lin, G. Qiao, Y. Tai, A. Li, K. Jia, and G. Liu, *HWC-Loco: A Hierarchical Whole-Body Control Approach to Robust Humanoid Locomotion*, Mar. 2025. arXiv: 2503.00923 [cs].
[55] B. Xu, H. Weng, Q. Lu, Y. Gao, and H. Xu, *FACET: Force-Adaptive Control via Impedance Reference Tracking for Legged Robots*, May 2025. arXiv: 2505.06883 [cs].
[56] J. Dao, H. Duan, and A. Fern, "Sim-to-real learning for humanoid box loco-manipulation," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2024, pp. 16 930–16 936.
[57] T. Lin, Z.-H. Yin, H. Qi, P. Abbeel, and J. Malik, *Twisting Lids Off with Two Hands*, Oct. 2024. arXiv: 2403.02338 [cs].
[58] Y. Ze *et al.*, *TWIST: Teleoperated Whole-Body Imitation System*, May 2025. arXiv: 2505.02833 [cs].
[59] P. Roth, J. Nubert, F. Yang, M. Mittal, and M. Hutter, *ViPlanner: Visual Semantic Imperative Learning for Local Navigation*, May 2024. arXiv: 2310.00982 [cs].
[60] H. Qi, B. Yi, M. Lambeta, Y. Ma, R. Calandra, and J. Malik, *From Simple to Complex Skills: The Case of In-Hand Object Reorientation*, Jan. 2025. arXiv: 2501.05439 [cs].
[61] Z. Wang, J. Zhou, and Q. Wu, *Dribble Master: Learning Agile Humanoid Dribbling Through Legged Locomotion*, May 2025. arXiv: 2505.12679 [cs].
[62] Z. Xiao *et al.*, *Unified Human-Scene Interaction via Prompted Chain-of-Contacts*, Nov. 2024. arXiv: 2309.07918 [cs].
[63] G. Yang, *Vuer*, 2024.
[64] X. Cheng, J. Li, S. Yang, G. Yang, and X. Wang, "Open-television: Teleoperation with immersive active visual feedback," *arXiv preprint arXiv:2407.01512*, 2024.
[65] M. Mittal *et al.*, "Orbit: A unified simulation framework for interactive robot learning environments," *arXiv preprint arXiv:2301.04195*, 2023. arXiv: 2301.04195.

APPENDIX

A. Communication Architecture

The teleoperation system employs a distributed communication architecture that coordinates multiple components across different computational nodes. Dual cameras mounted on the robot's onboard computer transmit stereo images to the host computer via TCP and ZeroMQ protocols. The host computer processes these images for VR visualization while receiving operator commands through a network router connection. Robot actuators (dexterous hands and joints) communicate bidirectionally with the host computer using DDS protocol, transmitting states and receiving action commands. The system uses an asynchronous architecture where the teleoperation solver processes VR inputs to generate robot commands, while a separate deployment module runs at 50Hz to continuously read and execute the latest commands. This design ensures responsive control while maintaining system modularity.

The complete communication architecture can be summarized as follows:

$$\text{Cameras} \xrightarrow{\text{TCP/ZeroMQ}} \text{Host} \xrightarrow{\text{Network}} \text{VR Headset} \tag{74}$$

$$\text{VR Headset} \xrightarrow{\text{Network}} \text{Host} \xrightarrow{\text{DDS}} \text{Solver} \tag{75}$$

$$\text{Solver} \xrightarrow{\text{DDS}} \text{Deployment} \xrightarrow{\text{DDS}} \text{Robot Actuators} \tag{76}$$

This distributed architecture enables scalable and responsive teleoperation while maintaining the modularity necessary for system development and debugging.

B. Domain Randomization

We use domain randomization to simulate the sensor noise and physical variations in the real-world. The randomization parameters are shown in Table III.

Parameter	Unit	Range	Operator
Angular Velocity	rad/s	± 0.2	scaling
Projected Gravity	-	± 0.05	scaling
Joint Position	rad	± 0.01	scaling
Joint Velocity	rad/s	± 1.5	scaling
Static Friction	-	[0.7, 1.0]	uniform
Dynamic Friction	-	[0.4, 0.7]	uniform
Restitution	-	[0.0, 0.005]	uniform
Wrist Mass	kg	[0.0, 2.0]	additive
Base Mass	kg	[-5.0, 5.0]	additive

TABLE III: Domain randomization parameters. Additive randomization adds a random value within a specified range to the parameter, while scaling randomization adjusts the parameter by a random multiplication factor within the range.

C. Reward Function

Our reward function is a sum of the following terms:

- **Tracking Linear Velocity Reward** (r_{vel}): This term encourages the robot to track the commanded linear velocity in the xy-plane.

$$r_{vel} := \exp(-\|v_{xy} - v^*_{xy}\|_2^2 / \sigma^2_{vel}),$$

where v_{xy} and v_{xy}^* represent the actual and commanded linear velocities, respectively. σ_{vel} is set to 0.5. Weight: 1.0.

- **Tracking Angular Velocity Reward** (r_{ang}): This term encourages the robot to track the commanded angular velocity.

$$r_{ang} := \exp(-\|\omega_z - \omega_z^*\|_2^2/\sigma_{ang}^2),$$

where ω_z and ω_z^* represent the actual and commanded angular velocities, respectively. σ_{ang} is set to 0.5. Weight: 1.25.

- **Root Height Tracking Reward** (r_{height}): This term encourages tracking of the commanded pelvis height.

$$r_{height} := \exp(-|h - h^*|^2/\sigma_{height}^2),$$

where h and h^* are the actual and commanded root heights. σ_{height} is set to 0.4. Weight: 1.0.

- **Upper Body Position Tracking Reward** (r_{upper}): This term encourages tracking of arm joint positions.

$$r_{upper} := \exp(-\|\boldsymbol{q}_{upper} - \boldsymbol{q}_{upper}^*\|_2^2/\sigma_{upper}^2),$$

where $\boldsymbol{q}_{upper}$ and $\boldsymbol{q}_{upper}^*$ are the actual and desired upper body joint positions. σ_{upper} is set to 0.35. Weight: 1.0.

- **Torso Yaw Tracking Reward** (r_{yaw}): This term encourages tracking of torso yaw orientation commands.

$$r_{yaw} := \exp(-e_{yaw}^2/\sigma_{torso}^2),$$

where e_{yaw} is the yaw orientation error. σ_{torso} is set to 0.2. Weight: 0.25.

- **Torso Roll Tracking Reward** (r_{roll}): This term encourages tracking of torso roll orientation commands.

$$r_{roll} := \exp(-e_{roll}^2/\sigma_{torso}^2),$$

where e_{roll} is the roll orientation error. σ_{torso} is set to 0.2. Weight: 0.25.

- **Torso Pitch Tracking Reward** (r_{pitch}): This term encourages tracking of torso pitch orientation commands with higher weight.

$$r_{pitch} := \exp(-e_{pitch}^2/\sigma_{torso}^2),$$

where e_{pitch} is the pitch orientation error. σ_{torso} is set to 0.2. Weight: 0.5.

- **Center-of-Gravity Tracking Reward** (r_{CoG}): This term maintains stability by keeping the center of gravity near the support base.

$$r_{CoG} := \exp(-\|\boldsymbol{p}_{CoG}^{xy} - \boldsymbol{p}_{feet}^{xy}\|_2^2/\sigma_{CoG}^2),$$

where $\boldsymbol{p}_{CoG}^{xy}$ is the horizontal center of gravity projection and $\boldsymbol{p}_{feet}^{xy}$ is the midpoint between ankles. σ_{CoG} is set to 0.2. Weight: 0.5.

- **Termination Reward**: This term penalizes episode termination.

$$r_{ter} := -200.0 \cdot \mathbb{I}_{\text{terminated}}$$

where $\mathbb{I}_{\text{terminated}}$ is 1 if the episode terminates, otherwise 0.

- **Z-axis Linear Velocity Reward**: This term penalizes the robot for moving along the z-axis.

$$r_z := -1.0 \cdot (v_z)^2$$

where v_z is the z-axis linear velocity.

- **Energy Reward**: This term penalizes output torques to reduce energy consumption.

$$r_e := -0.001 \cdot \sum_i |\tau_i \cdot \dot{q}_i|$$

where τ represents the joint torques and $\dot{q}$ represents the joint velocities.

- **Joint Acceleration Reward**: This term penalizes excessive joint accelerations to promote smooth motions.

$$r_{ja} := -2.5 \times 10^{-7} \cdot \|\ddot{q}\|_2^2$$

where $\ddot{q}$ represents the joint accelerations of the configured joints.

- **Action Rate Reward**: This term penalizes rapid changes in actions to encourage smooth control.

$$r_{ar} := -0.1 \cdot \|a_t - a_{t-1}\|_2^2$$

where a_t represents the current action and a_{t-1} represents the previous action.

- **Base Orientation Reward**: This term penalizes non-flat base orientation to maintain an upright posture.

$$r_{ori} := -5.0 \cdot (\text{roll}^2 \cdot \text{mask}_{roll} + \text{pitch}^2 \cdot \text{mask}_{pitch})$$

where mask_{roll} and mask_{pitch} are adaptive masks based on torso command magnitudes.

- **Joint Position Limit Reward**: This term penalizes joint positions that exceed their soft limits.

$$r_{\text{jpl}} := -2.0 \cdot \sum_i \max(|q_i| - q_{i,\text{limit}}, 0)$$

where q_i represents the position of joint i, and $q_{i,\text{limit}}$ is the soft limit.

- **Joint Effort Limit Reward**: This term penalizes excessive torques on waist joints.

$$r_{\text{jel}} := -2.0 \cdot \sum_i \max(|\tau_i| - 0.999 \cdot \tau_{i,\max}, 0)$$

where τ_i is the torque and $\tau_{i,\max}$ is the maximum torque limit.

- **Joint Deviation Reward**: This term penalizes joint positions that deviate from their default positions.

$$r_{jd} := -0.15 \cdot \sum_i |q_i - q_{i,\text{default}}| - 0.3 \cdot \sum_j |q_j - q_{j,\text{default}}|$$

where i represents hip yaw and ankle roll joints, and j represents hip roll joints.

- **Feet Air Time Reward**: This term rewards appropriate stepping behavior for bipedal locomotion.

$$r_{\text{fat}} := 0.3 \cdot \min(t_{\text{air}}, 0.4)$$

where t_{air} is the air time when exactly one foot is in contact and velocity command is above 0.1 m/s.

- **Feet Slide Reward**: This term penalizes feet sliding during ground contact.

$$r_{\text{sl}} := -0.25 \cdot \sum_i \|v_{i,xy}\|_2 \cdot \mathbb{I}(\text{contact}_i)$$

where $v_{i,xy}$ is the horizontal velocity of foot i, and $\mathbb{I}(\text{contact}_i)$ indicates if the foot is in contact.

- **Feet Force Reward**: This term encourages maintaining appropriate ground reaction forces.

$$r_{\text{ff}} := -3 \times 10^{-3} \cdot \sum_i \min(\max(f_{z,i} - 500, 0), 400)$$

where $f_{z,i}$ is the vertical ground reaction force on foot i.

- **Feet Stumble Reward**: This term penalizes lateral forces that indicate stumbling.

$$r_{\text{fs}} := -2.0 \cdot \sum_i \mathbb{I}(\|f_{xy,i}\|_2 > 5|f_{z,i}|)$$

where $f_{xy,i}$ represents the horizontal ground reaction forces.

- **Flying State Reward**: This term penalizes the robot when it is airborne.

$$r_{\text{fly}} := -1.0 \cdot \mathbb{I}(\text{all feet off ground})$$

- **Undesired Contacts Reward**: This term penalizes undesired contacts with the environment.

$$r_{\text{uc}} := -1.0 \cdot \sum_{i \in \mathcal{C}} \mathbb{I}\left(\|\mathbf{F}_i\|_2 > 1.0\right)$$

where $\mathcal{C}$ represents the set of contact points excluding ankle contacts.

- **Ankle Orientation Reward**: This term penalizes excessive ankle roll orientations.

$$r_{\text{ankle}} := -0.5 \cdot \sum_i \|\text{gravity}_{xy,i}\|_2^2$$

where $\text{gravity}_{xy,i}$ is the projected gravity vector in each ankle frame.

D. ULC Hyperparameters

We illustrate the hyperparameters of ULC in Table IV.

E. Architecture Details

We illustrate the network architecture of ULC in Table V.

Parameter	**Value**
Number of Environments	8192
Training Iteration	100000
Environment Steps	24
Number of Training Epochs	5
Mini Batch Size	4
Max Clip Value Loss	0.2
Discount Factor	0.99
GAE discount factor	0.95
Entropy Regularization Coefficient	0.006
Learning rate	1.0e-3
Schedule	adaptive
Desired KL	0.01
Max Grad Norm	1.0
Value Loss Coefficient	1.0
Observation History Length	6
Action Scale	0.25
Episode Length	20.0 s
Simulation Timestep	0.005 s
Control Decimation	4

TABLE IV: Hyperparameters of ULC.

Component	Configuration
Actor Network	
Input Layer	Observation (History × Features)
Hidden Layer 1	Linear(Input → 1024) + ELU
Hidden Layer 2	Linear(1024 → 512) + ELU
Hidden Layer 3	Linear(512 → 512) + ELU
Hidden Layer 4	Linear(512 → 256) + ELU
Output Layer	Linear(256 → 29)
Critic Network	
Input Layer	Observation (History × Features)
Hidden Layer 1	Linear(Input → 1024) + ELU
Hidden Layer 2	Linear(1024 → 512) + ELU
Hidden Layer 3	Linear(512 → 512) + ELU
Hidden Layer 4	Linear(512 → 256) + ELU
Output Layer	Linear(256 → 1)
Policy Distribution	
Distribution Type	Gaussian
Initial Noise Std	1.0
Noise Type	log

TABLE V: ULC network architecture details. The table shows the configuration of both actor and critic networks with identical architectures except for output dimensions.