.1 Environment Details

.1.1 Real-world Deployment

Our real robot is built on the Unitree H1-2 platform, with two onboard Jetson AGX serving as the primary computing and communication unit. The control policy outputs the desired joint positions for each motor and transmits commands to the robot's low-level interface. The policy runs at an inference frequency of 50 Hz. The low-level interface operates at 500 Hz, ensuring smooth real-time control. Communication between the control policy and the low-level interface is facilitated via Lightweight Communications and Marshalling (LCM)

.1.2 Input and Output Information

In this section, we will provide detailed input and output information of the controller. Note that the Unitree H1 is a full-size general-purpose humanoid robot developed by Unitree Robotics, featuring multiple high-performance joints. Its legs have 10 DoF, the torso has 1 DoF, and each arm has 8 DoF. The legs H1-2 have 12 DoF and the other DoF is the same as H1.

Controller Input: As shown in Table 8, the controller's observation of each timestep comprises several components: the scaled base angular velocity (3 dimensions), the projected gravity vector (3 dimensions), the tracking command information (5 dimensions), the scaled difference between the current and default joint positions (21 dimensions for H1-2 and 19 dimensions for H1), the scaled joint velocity (21 dimensions for H1-2 and 19 dimensions for H1), and the previous actions executed by the controller (21 dimensions for H1-2 and 19 dimensions for H1). Lastly, we have reference pose (joint angles) information (21 dimensions for H1-2 and 19 dimensions for H1).

For the lower-body controller, we take 15 timesteps of observation as input. For the upper-body controller, we take 3 timesteps of observation as input.

Table 8: Controller Observation Components and Their Dimensions

Input Category	H1-2 Dimensions	H1 Dimensions
Base Angular Velocity	3	3
Projected Gravity Vector	3	3
Tracking Command	5	5
Joint Positions	21	19
Joint Velocity	21	19
Previous Actions	21	19
Reference Joint Angles	21	19

Critic Input: The critic's input includes one additional component, the base linear velocity (3 dimensions), compared to the controller's input. All other components remain the same as the controller's input.

Tracking Command Information: For the root-based mode, there are two kinds of input, root velocity command and upper-body reference joint angles. The former one is 3 dimensions, including the linear forward velocity, linear sideways velocity, and the angular velocity. For the training range, we set the linear forward velocity as $[-0.5, 1]$ m/s, linear sideways velocity as $[-0.5, 0.5]$ m/s, and the angular velocity as $[-0.5, 0.5]$ rad/s.

The latter one is 9-dimension reference joint angle information for both H1-2 and H1. The reference joint angle for the lower body is set to zero. Since the requirements for locomotion capabilities are high, we additionally introduce two dimensions to represent the clock signal, which is encoded using sine and cosine functions following the previous work

For the pose mode, the main target information is the reference joint angles. The reference joint angles are for the whole body of the humanoid. For H1 and H1-2, it is 19 dimensions and 21 dimensions respectively. In addition, the root velocity command is also given but it is extracted from the reference motion trajectory. Since the clock signal is meaningless for pose mode, we change it to the target root pitch and roll information which are also from the reference motion sequences.

Action Space: The action is the target position of joint proportional derivative (PD) controllers, which is 21 dimensions for Unitree H1-2 and 19 dimensions for Unitree H1.

.2 Model Details

.2.1 Upper-body Controller

The upper-body controller is a 3-layer MLP, which is the same as the Unitree codebase. More details can be found in https://github.com/unitreerobotics. The main difference is that the unitree codebase only take the observation of current timestep as input, while we concatenate 3-timestep observation as input.

.2.2 Lower-body Controller

The lower-body controller adopts the Gated Transformer-XL architecture. For code details, please refer to https://github.com/datvodinh/ppo-transformer

The Gated Transformer XL is designed for processing sequential data by integrating Transformer mechanisms with gated GRUs to enhance information flow. Below is an overview of its main components: 1. SinusoidalPE (Relative Positional Encoding) computes sinusoidal-based relative positional embeddings and uses fixed frequency transformations to provide relative positional awareness to the model. 2. TransformerBlock (Core Unit) consists of Multi-Head Attention, Layer Normalization, GRU-based Gating Mechanisms (GRUGate), and a Feedforward Network. The two GRUGates allow dynamic information flow between layers, improving memory retention. 3. GatedTransformerXL (Overall Architecture) incorporates SinusoidalPE for position encoding to maintain temporal dependencies and employs multiple TransformerBlocks for feature extraction while maintaining long-term memory. For our controller, the configuration of model can be found in Table 9. In addition, we take 15-timestep observation as input.

This architecture is well-suited for tasks requiring long-range dependency modeling, such as trajectory modeling in reinforcement learning, language modeling, or sequential decision-making tasks.

Table 9: Transformer Architecture Parameters

Parameter	Value
Number of Blocks (`num_blocks`)	2
Number of Heads (`num_heads`)	6
GRU Bias (`gru_bias`)	0.0
Hidden Size (`hidden_size`)	128
Embedding Dimension (`embed_dim`)	384
Memory Length (`memory_length`)	15

.3 Training Details

Supervised Initalization: For supervised learning, we use the standard MSE loss. It measures the mean squared error (squared L2 norm) between the action and target. The target is the reference joint angles.

For KL regularization, we minimize the Kullback-Leibler divergence (KL divergence) between two policies π^{lower} and π^{base} is given by:

$$\begin{aligned} & D_{\text{KL}}(\pi^{\text{lower}} \parallel \pi^{\text{base}}) \\ =& \mathbb{E}_{s\sim\rho_{\pi^{\text{lower}}}} \left[\sum_a \pi^{\text{lower}}(a|s) \log \frac{\pi^{\text{lower}}(a|s)}{\pi^{\text{base}}(a|s)} \right] \end{aligned} \tag{1}$$

RL training: The controller is further trained with standard PPO

Reward Design:In the main text, we provided an overview of our tracking-based reward design. Additionally, our reward function includes various penalties and regularization terms. The components of the regularization reward, along with their respective weights for calculating the final rewards, are detailed in Table **??**. The final reward integrates both the regularization terms and the tracking-based reward to effectively train a robust RL policy.

Domain Randomizations: Detailed domain randomization setups are summarized in Table 13 following the similar setups in OmniH2O

.4 Additional Results

More demonstrations of JAEGER, including retargeting results, root-based mode and pose-based mode, can be found in the link: **https://anonymous.4open.science/r/JAEGER/**.

Table 10: Hyperparameters for RL Controller

Hyperparameter	**Value**
Optimizer	Adam
β_1, β_2	0.9, 0.999
Learning Rate	1×10^{-4}
Batch Size	4096
Discount factor (γ)	0.99
Clip Param	0.2
Entropy Coef	0.01
Max Gradient Norm	1
Learning Epochs	2
Mini Batches	4
Value Loss Coef	1

Table 13: The range of dynamics randomization for simulated dynamics randomization, external perturbation, and terrain, which are important for sim-to-real transfer, robustness, and generalizability.

Term	**Value**
Dynamics Randomization	
Friction	$\mathcal{U}(0.2, 1.1)$
Base CoM offset	$\mathcal{U}(-0.1, 0)$ m
Link mass	$\mathcal{U}(0.7, 1.3)\times$ default kg
P Gain	$\mathcal{U}(0.75, 1.25)\times$ default
D Gain	$\mathcal{U}(0.75, 1.25)\times$ default
Control delay	$\mathcal{U}(20, 60)$ ms
External Perturbation	
Push robot	interval = 5s, $v_{xy} = 1$m/s
Randomized Terrain	
Terrain type	flat, rough