TOP: Time Optimization Policy for Humanoid Robot Standing Manipulation Stably and Accurately

Abstract— **Humanoid robots have the potential capability to perform a diverse range of manipulation tasks, but this is based on a robust and precise standing controller. Existing methods are either ill-suited to precisely control high-dimensional upper-body joints, or difficult to ensure both robustness and accuracy, especially when upper-body motions are fast. This paper proposes a novel time optimization policy (TOP), to train a standing manipulation control model that ensures balance, precision, and time efficiency simultaneously, with the idea of adjusting the time trajectory of upper-body motions but not only strengthening the disturbance resistance of the lower-body. Our approach consists of three parts. Firstly, we utilize motion prior to represent upper-body motions to enhance the coordination ability between the upper and lower-body by training a variational autoencoder (VAE). Then we decouple the whole-body control into an upper-body PD controller for precision and a lower-body RL controller to enhance robust stability. Finally, we train TOP method combined with the decoupling controller and VAE to reduce the balance burden resulting from fast upper-body motions that would destabilize the robot and exceed the capabilities of the lower-body RL policy. The effectiveness of the proposed approach is evaluated via both simulation and real world experiments, which demonstrate the superiority on standing manipulation tasks stably and accurately. The project page can be found at https://anonymous.4open.science/w/top-258F/.**

I. INTRODUCTION

Humanoid robots are the most potential embodied agents for the purpose of liberating human-level labors, as they are designed to perform anthropomorphic motions and various whole-body loco-manipulation tasks, including industrial parts assembly, home service, etc.[1]. Their anthropomorphism naturally makes them more suitable than other specific robots to interact with environments, objects and humans to complete various physical tasks. Although rapid growth has been achieved in the field of humanoid robots[2], it remains a challenge to execute various intricate tasks while maintaining balance and precision simultaneously due to the intrinsic instability characteristic of humanoid robot.

Existing methods can be broadly divided into two paradigms: whole-body controllers[3, 4, 5] and upper and lower-body decoupled controllers[6, 7, 8]. Traditional whole-body controllers, such as model predictive control (MPC), are capable of generating precise motions[3], but maintaining stability and robustness in real world remains a significant challenge. Whole-body RL-based controllers offer improved dynamic robustness[9, 10], but often struggle to accurately track complex reference trajectories, particularly involving high-dimensional upper-body joints[11], and are prone to overfitting to suboptimal behaviors or generating unpredictable actions[12]. To mitigate these limitations, a decoupled control architecture has been proposed[7], wherein the high-dimensional upper body is controlled by a PD controller to ensure precise trajectory tracking, while the lower body is governed by an RL policy to provide robust balance against external perturbations. This decoupled controller has shown promise in enhancing both the stability and precision required for standing manipulation tasks[8].

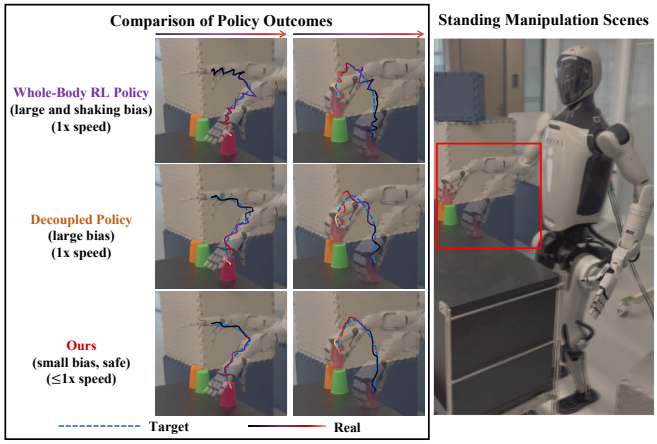

Fig. 1: Illustration of different methods. **A**: Whole-body RL policy, but it will be ill-suited to control high-dimensional upper-body joint, which cause large and shaking bias from target trajectory. **B**: Decoupled policy avoid shaking bias by using PD controller for upper-body, but still lack the consideration about momentum caused by rapid upper-body motions, remaining large bias. **C**: Our method adjust timestamp of motions aiming to reduce the impact of momentum and making standing safer, which gain smaller bias but need more time to achieve the goal.

However, all the aforementioned methods tend to execute upper-body motions without fully considering the robot's actual execution capabilities. This oversight often neglects the dynamic consequences of momentum changes caused by fast upper-body motions, which can lead to instability, loss of balance, or even collisions with the environment. A key issue lies in the momentum introduced by upper-body motions: fast upper-body motions may destabilize the robot and impact the tracking precision, while slower motions can certainly reduce momentum changes and improve stability and accuracy, but sacrifice time efficiency. In other words, whether the reference trajectory is generated by teleoperation[2], VLA[13], or other planners[14], determining the appropriate motion speed remains a tricky problem in whole-body standing manipulation scenarios.

Therefore, in this paper, we propose a novel learning-based framework to improve lower-body stability and upper-body

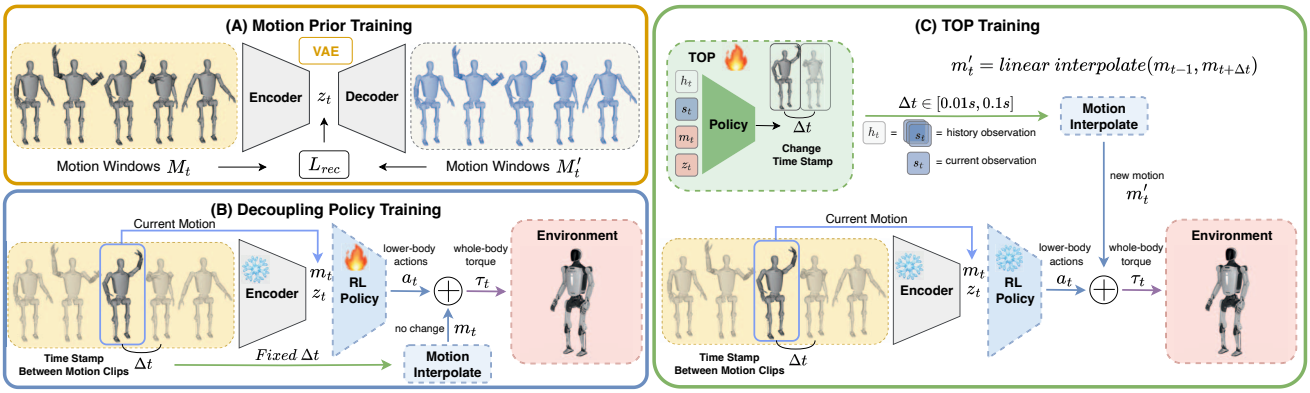

Fig. 2: The overall architecture. (A) Training a latent code z_t based on VAE structure to represent diverse upper-body motions. (B) A balance RL based policy to control the robot stay still, while upper-body use PD controller to execute current motion m_t, with a fixed time stamp Δt between motions. (C) Then we train the TOP to optimize the timestamp between the motion clips to reduce the speed of motions and the impact of the momentum changes, which will slow down the current motion m_t and execute new motion m'_t calculated by $linear\ interpolate(m_{t-1}, m_{t+\Delta t})$.

motion precision by using **TOP** to optimize the timestamps of upper-body motions, minimizing momentum-induced balance disturbances. Our approach aims to strike a balance between stability, accuracy, and time efficiency by optimizing the time trajectory of motions to determine motion speed. This reduces the effect of fast upper-body motions on overall momentum and enhances both balance and execution efficiency, while minimizing disturbances from lower-body shifts that could affect upper-body tracking. The framework consists of three stages: first, a structured representation of diverse upper-body motions is learned to capture motion clip features; second, training a RL-based lower-body policy to maintain balance while the upper body executes motion clips via a PD controller; finally, training TOP method via supervised reinforcement learning to optimize the timestamps between motion clips. This alleviates balance challenges caused by fast upper-body motions that often exceed the limits of balance controllers, preventing falls and improving performance in standing manipulation tasks. We validate our approach in both simulation and real-world environments, demonstrating its versatility and effectiveness. Our contributions are as follows:

- propose a novel framework that can realize standing manipulation stably and accurately with the consideration about time efficiency, achieving various complex upper-body motions and manipulation tasks.
- TOP is proposed for humanoid standing manipulation, leveraging supervised reinforcement learning to optimize motion timestamps for high accuracy, enhanced stability, and time efficiency.
- our method is tested both in simulation and the real world that can generalize well to practical standing manipulation tasks, which demonstrates the adaptability and efficiency of our approach.

II. RELATED WORKS

A. Whole-body Balance Controller

The primary challenge of humanoid robots in achieving diverse rich and precision motions is the whole-body controller that tracks accurate upper-body motions and stabilizes the entire body in real time[1, 15]. Previously, different attempts have been made in dynamics modeling and control[16, 17], but these methods sacrifice precise models and fail to track large-scale motions. Moreover, the dilemma of balancing computationally efficiency and model complexity restricts its practical application[3].

Data-driven approaches are gradually taking the lead in this field, especially reinforcement learning combined with some sophisticated methods [18, 19]. Zhang et al. [20] propose a method of whole-body humanoid robot locomotion by leveraging human motion references through imitation and optimization techniques, but it is difficult to generalize it to more whole-body motions. Fu et al. [10] introduce a framework based on transformer structures for humanoid robots to shadow and imitate human motions, improving natural interaction and adaptability. He et al. [21] successfully overcome the sim-to-real gap and train a universal and dexterous system for whole-body teleoperation and learning. However, these works focus on whole-body control with smaller arm DoFs and do not pay attention to the impact of lower-body stability on upper-body motion accuracy.

To address the limitations of RL in precisely controlling high-dimensional upper-body joints, researchers adopt a decoupled control architecture that combines a PD controller for accurate upper-body motion execution with an RL-based lower-body controller to ensure robust stability. Cheng et al. [22] decouples the upper and lower-body controllers to obtain precision, but still lacks a measurement of the accuracy of upper-body motion and may lead to instability when the robot is standing still. Lu et al. [8] propose a decoupled controller with CVAE to represent upper-body motions, aiming to maintain balance and precisely control in standing manipulation. On their basis, as a comparison, our approach integrates upper and lower-body separation control with time optimization policy that adjust the timestamp of upper-body motion, ensuring to improve standing stability, motion precision and time efficiency.

B. Motion Representation Learning

Motions from human or robots have spatial-temporal features in high-dimensional spaces[23], and an effective representation to extract these complex spatial-temporal structures is highly needed[24, 25]. Peng et al. [26] introduced adversarial training to learn motion priors, improving reinforcement learning policy efficiency and generalization. Hassan et al. [27] present a similar adversarial imitation learning framework to generate realistic interactions by integrating simulation with data-driven motion synthesis. However, when faced with large-scale data, it is difficult for such methods to learn a universal strategy, failed with mode collapse[28].

Another common way to represent complex motion is to filter and extract key motion information to represent smooth temporal and spatial composition[29, 30], but struggle with handling highly diverse or long-range motions. Recently, advances in generative models, like diffusion models[31] or variational autoencoder(VAE)[32], provided a good idea for representation of motions by creating a latent code to learn the distribution of multimodal motion sequences [4]. To provide a multimodal representation of upper-body motions, we decide to use VAE to extract a latent kinematic motion space and reconstruct them for training.

III. METHOD

A. Overview

Our framework as depicted in Figure 2. The motion prior represents the multimodal distribution of upper-body motions, aiming to enhance the the coordination ability between the upper and lower-body controllers. In the decoupling controller training stage, we use the curriculum schedule to reduce the exploration burden of RL policy, and we fixed the time stamps between motion clips combined with motion priors to train a robust lower-body RL controller. In the TOP training process, we introduce supervised reinforcement learning to optimize the time stamp of motion clips, in which the init guess of time stamps can be given by supervise learning. And the action chunking[14] can improve the smoothness of the timestamp, which we introduce in details in III-D.

B. Extract Motion Priors

To improve awareness of the lower-body controller for upper-body past and future motions, we provide prior knowledge of upper-body motion, which is important for training more robust lower-body controllers [11, 22]. Specifically, we train an encoder-decoder pair to reconstruct the sequence of upper-body motions and incorporate the latent space as a representation in the state space of the lower-body control.

To represent diversity kinematic motions of a human or robot and capture more fine-grained level of motions, we use variational autoencoder (VAE) structure, which has already been shown as an effective motion representation and has the ability to learn the motion distribution and similarities of adjacent motions. We extract kinematic state of upper-body motions, consisting of joint position, velocity for a few past and future window frames. Our VAE structures include an encoder E and a decoder D, and the latent space is modeled as a multivariate Gaussian distribution $\boldsymbol{z}_t \in \mathbb{R}^{d_z}$.

$$\boldsymbol{m}_t = \{\boldsymbol{r}_t, \boldsymbol{\theta}_t, \boldsymbol{q}_t^{upper}, \dot{\boldsymbol{q}}_t^{upper}\}. \tag{1}$$

where $\boldsymbol{r}_t \in \mathbb{R}^3$ is the position of the base relative to the world frame. Since we only need to encode the upper-body motion, setting the $\boldsymbol{r_t}$ as the constant is natural. $\boldsymbol{\theta}_t$ is the orientation of the base frame, represents as 6D vector. The joint angles and velocities are given by $\boldsymbol{q}_t^{upper} \in \mathbb{R}^{n_j}$ and $\dot{\boldsymbol{q}}_t^{upper} \in \mathbb{R}^{n_j}$, where $n_j = 15$ includes two 7-dof arms and one waist joint.

Formally, we extract the past and future frames to consist motion windows of length $2W + 1$ from the distinct motion clips. In order to ensure the stability of the training of VAE networks, it is common to normalize the input data, and we use the mean and standard deviation of all motion clips in dataset to normalize our motion window frames, except orientation.

$$\boldsymbol{M}_t = \{\boldsymbol{m}_{t-W}, \ldots, \boldsymbol{m}_t, \ldots, \boldsymbol{m}_{t+W}\}. \tag{2}$$

The encoder of our VAE $E_\phi(\boldsymbol{z}_t|\boldsymbol{M}_t)$ maps the motion window to latent space $\boldsymbol{z}_t \in \mathbb{R}^{d_z}$, $d_z = 64$, and the sampled latent variable is then mapped back to input space by the decoder $D_\theta(\boldsymbol{M}'_t|\boldsymbol{z}_t)$. And we decide to use β-VAE[33] with the reconstruction loss as follows:

$$L_{rec}(\boldsymbol{M}_t, \boldsymbol{M}'_t) = \frac{1}{2W+1} \sum_{i=t-W}^{t+W} l_{rec}(\boldsymbol{m}_t, \boldsymbol{m}'_t) \tag{3}$$

$$\begin{aligned} l_{rec}(\boldsymbol{m}_t, \boldsymbol{m}'_t) = &||R(\boldsymbol{\theta}_t) - R(\boldsymbol{\theta}'_t)|| + ||\boldsymbol{q}_t - \boldsymbol{q}'_t|| \\ &+ ||\dot{\boldsymbol{q}}_t - \dot{\boldsymbol{q}}'_t|| + ||\boldsymbol{p}_t - \boldsymbol{p}'_t|| \end{aligned} \tag{4}$$

where the $R(\cdot)$ represents that computing rotation matrices for orientations using the Gram-Schmidt process, and because we already normalize the quantities, no relative weights are needed here. It is worth noting that the window size of $\boldsymbol{M}_t$ should be short enough to achieve motion generalization that the latent space can capture the features of primitive motion blocks that may appear in unseen motion sequences.

C. Training Decoupling Policy

We train a balance policy using a Legged Gym-based reinforcement learning framework, where PPO was used to update our lower-body policy. During training, the motion sequence is randomly chosen from the dataset at the beginning of a new episode, and retrieving the motion pair $(\boldsymbol{m}_t, \boldsymbol{z}_t)$ from Encoder $E_\phi(\boldsymbol{z}_t|\boldsymbol{M}_t)$. Then, we feed the motion pair to our lower-body policy to provide the instantaneous kinematic reference motions of upper-body and the past and future information, which helps the comprehension of the policy about disturbances caused by upper motions on balance.

We consider our balance lower control policy as a goal-conditional $\pi_\phi(\boldsymbol{a}_t|\boldsymbol{s}_t, \boldsymbol{g}_t) : \mathbb{G}\times\mathbb{S} \to \mathbb{A}$, where $\boldsymbol{g}_t \triangleq (\boldsymbol{m}_t, \boldsymbol{z}_t) \in \mathbb{G}$ is the goal at the time t that indicates the target of upper-body motion clip and the latent code from Encoder $E_\phi(z_t|M_t)$ from dataset. $\boldsymbol{s}_t \triangleq \{\boldsymbol{q}_t, \dot{\boldsymbol{q}}_t, \boldsymbol{\theta}_t, \boldsymbol{\omega}_t, \boldsymbol{a}_{t-1}, \boldsymbol{g}_t\} \in \mathbb{S}$ is the current observation, where $\boldsymbol{q}_t \in \mathbb{R}^{27}$, $\dot{\boldsymbol{q}}_t \in \mathbb{R}^{27}$ are the position and velocity of whole body joints, $\boldsymbol{\omega}_t$ is the angular velocity of the base, $\boldsymbol{a}_{t-1} \in \mathbb{R}^{12}$ means the last action of lower-body joint.

$\boldsymbol{a}_t \in \mathbb{A}$ is the action of lower-body joints. Both of upper-body motions and lower-body actions are actuated by a PD torque controller $\boldsymbol{\tau}_t = k_p(\boldsymbol{a}_t - \boldsymbol{q}_t) + k_d\dot{\boldsymbol{q}}_t$ for each joint. The reward design is shown in Table II. It should be noted that the rewards for regularization of actions are used to shape the standing mode, and the input history will be encoded as a hybrid internal embedding[34], which improves the training efficiency and robustness of lower-body RL controller.

In order to reduce the exploration burden caused by upper-body motions and gain more stable training process, we introduce a training curriculum schedule that will change the amplitude of target motion clips[8]. For the PD controller of the joint position during the training, the target joint position is calculated by

$$\boldsymbol{q}_t^{upper} = \boldsymbol{q}_{default}^{upper} + \alpha_i(\boldsymbol{q}_{target}^{upper} - \boldsymbol{q}_{default}^{upper}) \tag{5}$$

where $\boldsymbol{q}_{default}^{upper}$ is the default joint position of upper-body. The $\alpha_i \in [0, 1]$ is the unique amplitude factors of motion i, which is changed during the training by the rules similar to the [8].

D. Time Optimization Policy

We design a reinforcement learning policy to optimize the timestamp between motion clips $\boldsymbol{m}_t, \boldsymbol{m}_{t+1}...\boldsymbol{m}_{t+N}$, considering the latent variable $\boldsymbol{z}_t$, the current observation $\boldsymbol{s}_t$ and the history observation $\boldsymbol{h}_t$. The structure of this net can be described as $\Delta t_t^{seq} = \pi_\theta(\boldsymbol{m}_t, \boldsymbol{z}_t, \boldsymbol{s}_t, \boldsymbol{h}_t)$, the θ is the learnable variable of TOP. The output of TOP ($\Delta t_t^{seq} = \Delta t_t, ..., \Delta t_{t+N}$) will feed into RL policy combined with the motion pair $(\boldsymbol{m}_t, \boldsymbol{z}_t)$. Once the timestamp of motion has been set to $t + \Delta t_t$, it is necessary to change the origin motion $\boldsymbol{m}_t$ to the new motion $\boldsymbol{m}'_t = linear\ interpolate(\boldsymbol{m}_{t-1}, \boldsymbol{m}_{t+\Delta t_t})$ by linear interpolation. Because the motions of the dataset $\mathcal{M}$ satisfy the kinematic and dynamic constraints of the robot, after linear interpolation, m'_t will not violate the constraints.

It should be noted that because of the rapid motion in the past few frames, the robot may lose its balance at the current moment, which exhibits a certain degree of lag. In other words, when we change the timestamp of the current upper-body motion to slow down the motion, it may be reflected in future multi frames feedback. This means that the output of the past policy (Δt_{t-N}) will affect the current balance performance of robot. Therefore, we are seeking to optimize timestamps for a period of time in the future, and inspired by *action chunking*[35], the policy model becomes $\pi_\theta(\Delta t_t^{seq}|\boldsymbol{m}_t, \boldsymbol{z}_t, \boldsymbol{s}_t, \boldsymbol{h}_t)$ instead of $\pi_\theta(\Delta t_t|\boldsymbol{m}_t, \boldsymbol{z}_t, \boldsymbol{s}_t, \boldsymbol{h}_t)$, $N = 10$ is the horizon step, which was obtained based on our testing.

A simple implementation of predicting a horizon of future timestamps like action chunking will be sub-optimal: if we directly shift by one timestamp until the end of trajectory, a new timestamps trajectory is incorporated abruptly every N steps and can result in the sudden action to slow down. To improve the action smoothness and avoid jerky discrete switching of timestamps, we query the output of policy at every timestamp, and give every prediction with an exponential weighting scheme $w_i = exp(-k * i)$, where w_0 represents the weight of the oldest action. Then we use the weighted average for the current predictions similar to [14].

Obviously, this kind of feedback lag brings an extra burden of policy exploration and it is more difficult to predict a short horizon of timestamp trajectory than a single timestamp. It is natural for us to think that the original timestamp of the motion clips can be set as the initial guess solution for the policy. Therefore, we introduce the supervise learning combined with reinforcement learning to reduce the need for inefficient random exploration, which also help steer the agent toward high-quality policies instead of suboptimal solutions [36]. Our PPO-based supervised reinforcement learning framework incorporates the following loss components:

$$L_{sup} = -\sum\nolimits_i \boldsymbol{\Delta} t_i^{log} \boldsymbol{\pi}_\theta(\boldsymbol{\Delta} t_i^*|\boldsymbol{s}_i) \tag{6}$$

$$L_{RL} = -\mathbb{E}_t[\min(r_t(\theta)A_t, clip(r_t(\theta), 1-\epsilon, 1+\epsilon)A_t)] \tag{7}$$

$$L_{total} = \lambda_{sup} L_{sup} + \lambda_{RL} L_{RL} \tag{8}$$

Where L_{sup}, L_{RL} are the loss of supervise and reinforcement learning. $r_t(\theta)$ is the policy ratio, and A_t is the advantage function. We combine both losses with weighting factors $\lambda_{sup} = 0.1$ and $\lambda_{RL} = 0.5$ to balance supervised learning and reinforcement learning. At the beginning of training, L_{sup} is relatively large and plays a dominant role. After L_{sup} decreases, the loss of L_{RL} plays a dominant role. And the rewards of TOP is shown in Table I.

We can easily observe that the reward design of TOP and the design of RL Rewards have similarities, as they both want to guide that robots can execute motions accurately while maintaining balance as much as possible. So why not train jointly? Because we want to make the TOP policy independent of the controller, we can retrain the top policy when the controller changes. In this way, we can freely change and update the controller without considering the top policy, and the performance of TOP policy will not impact the controller itself. During our training, we find that training both policy jointly will not affect the results of RL policy, but need more iteration to converge and the worse performance of TOP policy, which means separate training will bring more stable training results.

TABLE I: REWARDS OF TOP

Term	Expression	Weight
Gravity projection	$\exp(-20\|\mathbf{pg}_t^{xy}\|)$	2.5
Balance penalty	$\exp(20\sqrt{(\mathbf{pg}_t^x)^2 + (\mathbf{pg}_t^y)^2}) - 1$	-1.0
Support constraint	$\lg(7 * (\mathbf{p}_{feet}^{center} - \mathbf{p}_{com}^{project}))$	-5
Encourage small Δt	$\sum_{i=0}^{N} \exp\left(-\frac{(\Delta t_i)^2}{2\sigma^2}\right), \sigma = 0.5$	5.0
Δt smooth	$0.1 * \sum_{i=0}^{N-1}(\|\Delta t_{i+1} - \Delta t_i\|)$	-0.1
Δt norm	$\|\sum_{i=0}^{N} \Delta t_i\|$	0.1

IV. EXPERIMENTS

A. Experimental Setup

We perform our experiments on a full-size humanoid robot to validate our approach in both simulation and the real world. The robot stands 1.65m tall, weighs 60kg, and has 41 degrees of freedom, including two 6kg arms with 7-DoF each and a payload capacity of 3kg per arm. This combination of weight, high precision, and manipulation capability poses

a significant challenge to maintain balance during manipulation. The training dataset $\mathcal{M}$ is the large-scale human motion dataset GRAB[37]. Additionally, during testing, we also introduced more difficult motions with 16000 clips, and used it for algorithm testing, named it as dataset $\mathcal{T}$. All these motions are retargeted to humanoid robot via motion retargeting methods[38].

VAE We use a latent code dimension of d_z = 64 and a window size of W = 30, corresponding to a real time of 0.6s. The structure of our variational autoencoder is built with four 1D-convolutional layers, Layer Normalization, ReLU activations, and a final linear layer. At the bottleneck layer, we double the encoder output dimension and sample from a multivariate Gaussian distribution[4].Refer to Table II for details of the hyperparameters.

RL Policy We use domain randomization to provide robustness to object parameters, similar to the setting in [9]. It is worth noting that the base Center-of-Mass (CoM) position, motor delays and torque noise play a crucial role in solving the problem of sim-to-real. We also add some upper-body position and velocity noise during training in order to obtain a more robust lower-body balance control policy.

TOP The structure of our TOP network is built with three-layers MLPs. We also employ the actor-critic framework to enhance the stability of learning. Supervised learning provides an initial guess of Δt_t^{seq} to encourage smaller Δt_t, but in the event that the robot falls, the loss of supervision L_{sup} will not be added to L_{total}, which will guide the robot to slow down specific motions, ensuring a success rate. In practise, after testing the data distribution of Δt, we limit its range in $[0.01s, 0.1s]$, and we also find both action chunking with weighting scheme and the motion prior are important for the success of TOP, which produces precise and smooth motion. The results are shown in table III.

TABLE II: VAE PARAMETERS AND RL REWARDS

VAE Parameters		**VAE Training**	
Param.	**Value**	**Param.**	**Value**
$kl\ weight$	0.002	Batch size	512
W	30	Number of epochs	30 000
d_z	64	Learning rate	0.003
Param.	0.8M	KL-scheduler cycles/ratio	7 / 0.5
Term		**Expression**	**Weight**
Base linear Velocity^{xy}		$\exp -4(\mathbf{v}_{xy}^2)$	3.0
Base linear Velocity^{z}		$\Sigma \mathbf{v}_z^2$	-0.8
Base angular Velocity^{xy}		$\Sigma \boldsymbol{\omega}_{xy}^2$	-0.1
Base orientation		$\Sigma \mathbf{g}_{xy}^2$	-1.5
Base acceleration		$\exp(-3\|\mathbf{v}_t - \mathbf{v}_{t-1}\|_2)$	0.2
Stand still		$\exp(-10\|\mathbf{q}_t^{leg} - \mathbf{q}^{leg,ref}\|_2)$	1.0
Feet contact		$\mathbf{1}(F_{\text{feet}}^z \geq 5)$	0.5
Feet slip		$\mathbf{1}(F_{\text{feet}}^z \geq 5) \times \sqrt{\|\mathbf{v}_t^{\text{feet}}\|_2}$	0.2
Action rate		$\|\mathbf{a}_t - \mathbf{a}_{t-1}\|_2^2$	-0.2
Action acceleration		$\|\mathbf{a}_t + \mathbf{a}_{t-2} - 2\mathbf{a}_{t-1}\|_2^2$	-0.2
Torques		$\|\boldsymbol{\tau}_t\|_2^2$	-5e-6
DoF velocity		$\|\dot{\mathbf{q}}_t\|_2^2$	-5e-4
DoF acceleration		$\|\ddot{\mathbf{q}}_t\|_2^2$	-1e-7

B. Evaluation of Motion Priors and Decoupling Policy

Generalization of Motion Priors. To evaluate the efficacy of the VAE model and the latent space, we use linear interpolating between the latent space trajectories corresponding to two different motions, one motion comes from dateset $\mathcal{M}$, the other is an unseen motion from dataset $\mathcal{T}$. As shown in Figure 3, the result of 50% linear interpolation between two different latent code, and we reconstruct it by Decoder $D_\theta(\boldsymbol{M}_t'|\boldsymbol{z}_t)$. We also show the reconstructed results of motions on the website, which demonstrate that our latent space can capture the features in short-horizon motions and has the ability to represent and reconstruct unseen motions that are in proximity to those of similar motion windows within the dataset.

Fig. 3: **Latent Generalization.** Given two motion windows, one from the dataset $\mathcal{M}$ (row 1), the other from unseen dataset $\mathcal{T}$ (row 3), and the results of 50% linear interpolation show in (row 2).

Robustness of Decoupling policy. To thoroughly evaluate the performance of our lower-body balance policy without TOP, we test upper-body motions from dataset $\mathcal{M}$ and unseen dataset $\mathcal{T}$ in simulation and real world. It is worth noting that the amplitude and speed of the motions from dataset $\mathcal{T}$ are larger than those in dataset $\mathcal{M}$, which put higher demands on the robustness. As shown in Figure 4, our policy can maintain balance while tracking upper-body motions with a certain level of precision. The balance RL policy we trained demonstrates strong robustness. Although there are occasional failures on the unseen dataset $\mathcal{T}$, particularly in challenging cases where the motion is too fast or both arms are raised above the head resulting in significant momentum changes that destabilize the robot. By slowing down such difficult motions, our system achieves a success rate of over 80%, indicating good generalization to unseen scenarios. The quantitative data are available in Table III.

C. Evaluation of TOP.

The performance of TOP is evaluated in simulation by comparing it to other baselines:

- **Fixed Root (reference)**: This baseline is that we fixed the root of robot, and directly execute the upper-body motions, which means the motions can be perfectly executed with the highest control accuracy.
- **Exbody*** [22]: This baseline uses whole-body RL policy with the tracking rewards to control whole-body joints.

Method	Time Cost ↓	Success Rate ↑	$\mathbf{E}_{\text{jpe}}^{\text{upper}}$ ↓	$\mathbf{E}_{\text{eepe}}^{\text{upper}}$ ↓	$\mathbf{E}_{\text{eeoe}}^{\text{upper}}$ ↓	$\mathbf{E}_{\mathbf{g}}$ ↓	$\mathbf{E}_{\text{acc}}^{\text{lower}}$ ↓	$\mathbf{E}_{\text{action}}^{\text{lower}}$ ↓	$\mathbf{E}_{\text{acc}}^{\text{upper}}$ ↓	$\mathbf{E}_{\text{action}}^{\text{upper}}$ ↓
Comparative Results										
Fixed Root (reference)	$15.0s$	100.0%	0.0130	0.0164	0.0594	1.000	-	-	-	-
Exbody* [22]	$\mathbf{15.0s}$	92.46%	0.0376	0.0741	0.0923	3.432	21.65	**0.7323**	24.26	2.976
Ours (TOP)	$40.5s$	**95.30%**	**0.0269**	**0.0270**	**0.0827**	**2.729**	**13.45**	0.8592	**10.41**	**1.601**
Ablation Results										
Ours w/o TOP $\Delta t = 0.01s$	$\mathbf{15.0s}$	82.43%	0.0301	0.0541	0.0962	4.438	16.87	0.9329	12.04	1.738
Ours w/o TOP $\Delta t = 0.03s$	$45.0s$	87.27%	0.0303	0.0434	**0.0810**	3.797	16.52	0.9473	11.26	1.647
Ours w/o TOP $\Delta t = 0.05s$	$75.0s$	92.41%	0.0299	0.0362	0.0813	3.573	15.79	0.9340	**9.79**	**1.556**
Ours w/o motion prior	$34.5s$	89.33%	0.0304	**0.0313**	0.0824	3.682	14.76	0.9130	10.92	1.640
Ours w/o acting chunking	$28.5s$	**93.16%**	**0.0288**	0.0353	0.0817	**3.275**	**14.28**	**0.8835**	11.51	1.659

TABLE III: Comparative results and Ablation results. We execute more than 10000 motion clips from dataset $\mathcal{M}$ and unseen dataset $\mathcal{T}$, and report their time cost, success rate, mean metrics. Mean Metrics are only calculated based on the data of the robot standing in place, and the data of robot taking steps and falls will not be included in the calculation.

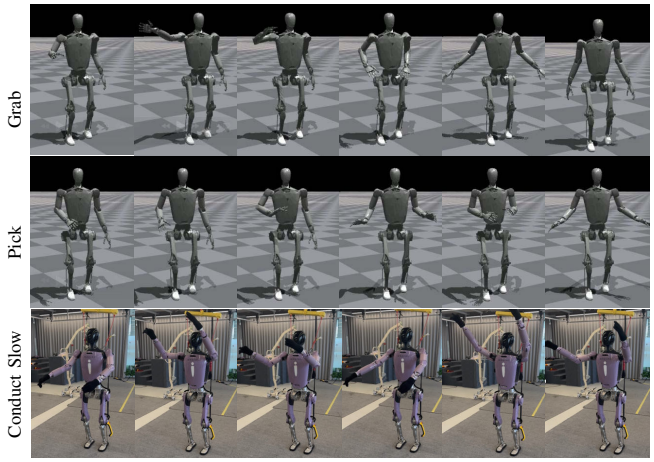

Fig. 4: **Robustness.** Given three motion windows in simulation and real world, grab and pick motions are sampled from the dataset $\mathcal{M}$ (row 1, row 2), and the conduct slower motions comes from unseen dataset $\mathcal{T}$(row 3).

The code comes from the Exbody* [22] and similar methods include OmniH2O[21] in terms of reward settings and RL training, and we apply this code to our own robot.

- **Ours (TOP)**: Our method of time optimization policy with balance lower-body RL policy.
- **Ours w/o TOP** $\Delta t = 0.01s/0.03s/0.05s$: These baselines only use the balance lower-body RL policy without TOP. The timestamp between motion clips is fixed as Δt.
- **Ours w/o motion priors**: This baseline uses TOP method but without the motion priors.
- **Ours w/o acting chunking**: Our method of TOP, but the time optimization policy only predicts single Δt_t not a sequence of Δt_t^{seq}.

The metrics are as follows:

- **Time Cost**: Average time cost of per 1500 motion clips (15s of origin data), which shows the time efficiency of executing the target motions.
- **Success Rate**: We define that the robot executes the upper body motion without falling, this motion clip will be labeled as successful motion.
- **Precision**: upper joint position error $\mathbf{E}_{\text{jpe}}^{\text{upper}}$, upper end effector position error in world frame $\mathbf{E}_{\text{eepe}}^{\text{upper}}$, upper end effort orientation error in world frame $\mathbf{E}_{\text{eeoe}}^{\text{upper}}$.
- **Stability**: projected gravity $\mathbf{E}_{\mathbf{g}}$, lower joint acceleration $\mathbf{E}_{\text{acc}}^{\text{lower}}$, lower action difference $\mathbf{E}_{\text{action}}^{\text{lower}}$.
- **Smoothness**: upper joint acceleration $\mathbf{E}_{\text{acc}}^{\text{upper}}$, upper action difference $\mathbf{E}_{\text{action}}^{\text{upper}}$.

Analysis of TOP methods. The results are shown in Table III. It is evident that, in comparison to the Exbody* method, the proposed method enhances the tracking accuracy of the joint position and end effector position, but has worse time efficiency. We would like to clarify, this because our method will execute the upper-body motion directly, while the Exbody* method involves executing upper-body motions through an RL policy, which changes the amplitude of the robot's upper-body motions to maintain balance. That is why the Exbody* method performs worse in precision. And our method will consider the impact of momentum changes that will slow down the motion, leading to more time cost.

Considering the ablation results, as shown in the Figure 7, our method helps reduce the balance burden and unpredicted inference caused by the lower-body, which enhances both the balance stability and the precision of upper-body motions. And with the help of TOP algorithm, we have achieved a good balance between time efficiency, and accuracy, demonstrating the outperfomance of our methods. Compare to **Ours w/o TOP** $\Delta t = 0.01s/0.03s/0.05s$, our method can automatically choose the appropriate $\Delta t \in [0.01s, 0.1s]$ in different situations: when motions seriously affect accuracy and balance, our algorithm will give a larger Δt even more than $0.05s$ to improve stability and precision; conversely, when motions are easy enough, a smaller Δt is chosen to maximize time efficiency. This adaptive TOP strategy results in improved overall precision and stability, while achieving higher average time efficiency.

We also measure the real-world end effector poses within accurate motion capture device, which proves that our method can still maintain balance and demonstrate high accuracy considering the time efficiency of motions. The quantitative results are shown in Table IV and Figure 6.

D. Manipulation Experiments

We evaluate our approach on manipulation tasks that require precision and robustness simultaneously, and we also use teleoperation combined with TOP for real-time motion slowing.

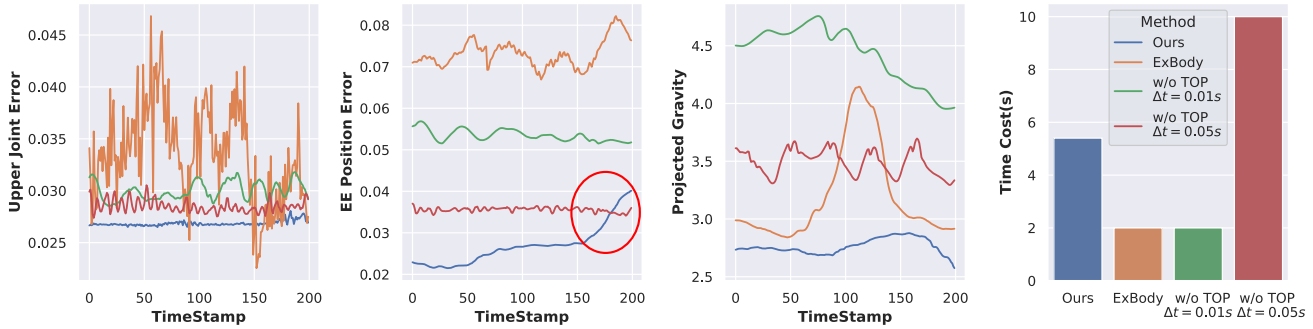

Fig. 5: Evaluation of precision(Upper Joint Error, EE position Error), stability(Projected Gravity) and time efficiency(Time Cost(s)) of motion clips. Because the upper-body motions sometimes cause small impact to whole-body balance, our method tends to execute the motion at origin speed rather than slow down, which which will increase the EE position error appropriately, which is a dynamic balance between precision, stability and time efficiency.

TABLE IV: Real-world Performance of serval motions.

Arm Tracking Error						
Task	x_{ee}(mm)	y_{ee}(mm)	z_{ee}(mm)	roll(rad)	pitch(rad)	yaw(rad)
motions in $\mathcal{M}$	5.0679	2.1918	0.5600	0.0213	0.0138	0.0048
motions in $\mathcal{T}$	33.735	3.0494	1.4913	0.0502	0.0127	0.0103

[1] The errors are averaged over twenty different motions.
[2] Motions from $\mathcal{M}$ have smaller amplitude and slower speed of the motions than motions from $\mathcal{T}$.

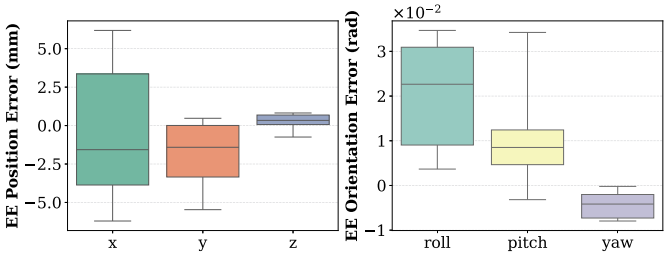

Fig. 6: **Evaluate of accuracy.** EE position and orientation error of our method. The motions are from $\mathcal{M}$ which includes the motion space and speed of the vast majority of manipulation tasks, and can well measure the accuracy of our algorithm in manipulation tasks.

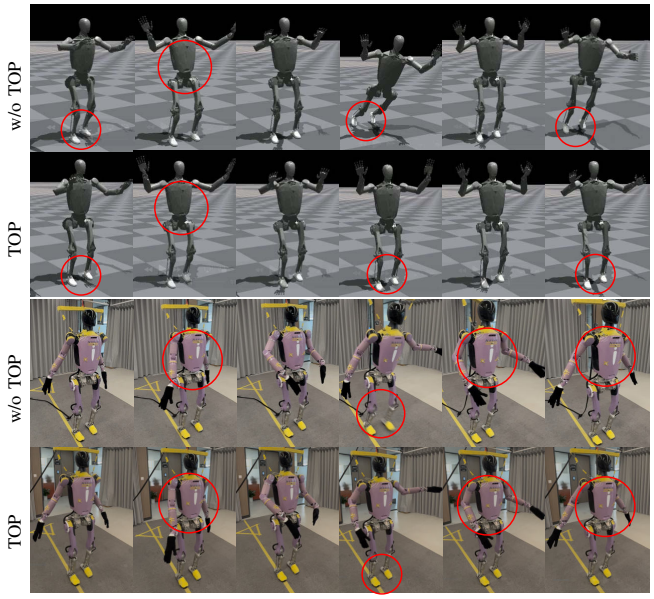

Fig. 7: **TOP Performance.** The result shows that TOP can improve the stability and reduce the occurrence of robot taking a step or falling both in simulation and real world.

In simple manipulation tasks, like grab a cup, policy tends to maintain the original speed to get higher time efficiency. When the amplitude and speed of the robot's motion affect balance, like dance with arms, it will actively slow down the motion, which can reduce the impact of momentum on the robot to control the robot stably and accurately. The results are shown in Figure 8.

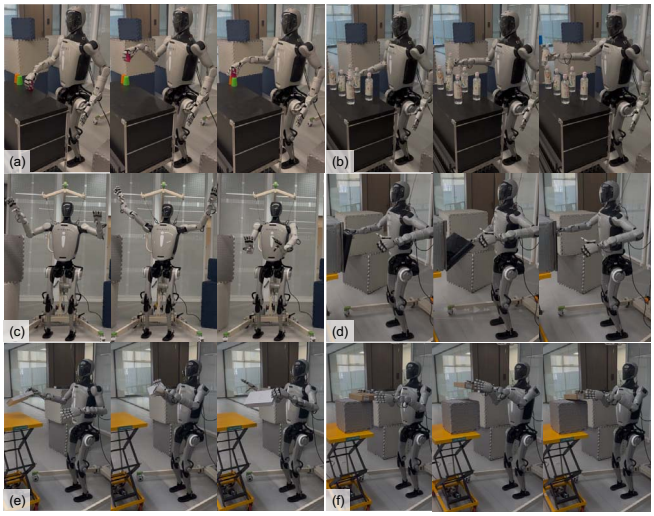

Fig. 8: **Manipulation tasks.** (a) grab the cup and put it onto cups, (b) take and deliver a bottle of water, (c) dance with arms but slow down the original motions while the lower body maintains balance, (d) open the oven, (e) Take the book and pass it from left hand to right hand, (f) carry a box.

V. CONCLUSIONS

In this paper, we propose a novel framework with the TOP method, which improves lower-body stability and upper-body precision by optimizing the timestamps of motion clips. By integrating a robust lower-body policy with precise upper-body tracking, and leveraging a VAE-based motion prior to capture upper-body features, our method achieves high success rates in both simulation and real-world experiments. However, current motions remain somewhat rigid, for instance, the robot tends to tilt backward rather than twisting its hips. In future work, we plan to incorporate motion generation modules to enable more natural standing adjustments and further enhance balance control.

REFERENCES

[1] Zhaoyuan Gu et al. "Humanoid locomotion and manipulation: Current progress and challenges in control, planning, and learning". In: *arXiv preprint arXiv:2501.02116* (2025).

[2] Qingwei Ben et al. "HOMIE: Humanoid Loco-Manipulation with Isomorphic Exoskeleton Cockpit". In: *arXiv preprint arXiv:2502.13013* (2025).

[3] Giulio Romualdi et al. "Online non-linear centroidal mpc for humanoid robot locomotion with step adjustment". In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 10412–10419.

[4] Agon Serifi et al. "Vmp: Versatile motion priors for robustly tracking motion on physical characters". In: *Computer Graphics Forum*. Vol. 43. 8. Wiley Online Library. 2024, e15175.

[5] Mazeyu Ji et al. "Exbody2: Advanced expressive humanoid whole-body control". In: *arXiv preprint arXiv:2412.13196* (2024).

[6] Kensuke Harada et al. "A humanoid robot carrying a heavy object". In: *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*. IEEE. 2005, pp. 1712–1717.

[7] Mingyo Seo et al. "Deep imitation learning for humanoid loco-manipulation through human teleoperation". In: *2023 IEEE-RAS 22nd International Conference on Humanoid Robots (Humanoids)*. IEEE. 2023, pp. 1–8.

[8] Chenhao Lu et al. "Mobile-television: Predictive motion priors for humanoid whole-body control". In: *arXiv preprint arXiv:2412.07773* (2024).

[9] Ilija Radosavovic et al. "Real-world humanoid locomotion with reinforcement learning". In: *Science Robotics* 9.89 (2024), eadi9579.

[10] Zipeng Fu et al. "HumanPlus: Humanoid Shadowing and Imitation from Humans". In: *arXiv preprint arXiv:2406.10454* (2024).

[11] Minghuan Liu et al. "Visual whole-body control for legged loco-manipulation". In: *arXiv preprint arXiv:2403.16967* (2024).

[12] Qisheng Zhang et al. "PPO-UE: Proximal Policy Optimization via Uncertainty-Aware Exploration". In: *arXiv preprint arXiv:2212.06343* (2022).

[13] Songming Liu et al. "Rdt-1b: a diffusion foundation model for bimanual manipulation". In: *arXiv preprint arXiv:2410.07864* (2024).

[14] Tony Z Zhao et al. "Learning fine-grained bimanual manipulation with low-cost hardware". In: *arXiv preprint arXiv:2304.13705* (2023).

[15] Kourosh Darvish et al. "Teleoperation of humanoid robots: A survey". In: *IEEE Transactions on Robotics* 39.3 (2023), pp. 1706–1727.

[16] Yasuhiro Ishiguro et al. "Bipedal oriented whole body master-slave system for dynamic secured locomotion with lip safety constraints". In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2017, pp. 376–382.

[17] Luigi Penco et al. "A multimode teleoperation framework for humanoid loco-manipulation: An application for the icub robot". In: *IEEE Robotics & Automation Magazine* 26.4 (2019), pp. 73–82.

[18] Zhengyi Luo et al. "Perpetual humanoid control for real-time simulated avatars". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 10895–10904.

[19] Nicklas Hansen et al. "Hierarchical World Models as Visual Whole-Body Humanoid Controllers". In: *arXiv preprint arXiv:2405.18418* (2024).

[20] Qiang Zhang et al. "Whole-body humanoid robot locomotion with human reference". In: *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2024, pp. 11225–11231.

[21] Tairan He et al. "OmniH2O: Universal and Dexterous Human-to-Humanoid Whole-Body Teleoperation and Learning". In: *arXiv preprint arXiv:2406.08858* (2024).

[22] Xuxin Cheng et al. "Expressive whole-body control for humanoid robots". In: *arXiv preprint arXiv:2402.16796* (2024).

[23] Sebastian Starke, Ian Mason, and Taku Komura. "Deepphase: Periodic autoencoders for learning motion phase manifolds". In: *ACM Transactions on Graphics (TOG)* 41.4 (2022), pp. 1–13.

[24] Xue Bin Peng et al. "Ase: Large-scale reusable adversarial skill embeddings for physically simulated characters". In: *ACM Transactions On Graphics (TOG)* 41.4 (2022), pp. 1–17.

[25] Chen Tessler et al. "Calm: Conditional adversarial latent models for directable virtual characters". In: *ACM SIGGRAPH 2023 Conference Proceedings*. 2023, pp. 1–9.

[26] Xue Bin Peng et al. "Amp: Adversarial motion priors for stylized physics-based character control". In: *ACM Transactions on Graphics (ToG)* 40.4 (2021), pp. 1–20.

[27] Mohamed Hassan et al. "Synthesizing physical character-scene interactions". In: *ACM SIGGRAPH 2023 Conference Proceedings*. 2023, pp. 1–9.

[28] Annan Tang et al. "Humanmimic: Learning natural locomotion and transitions for humanoid robot via wasserstein adversarial imitation". In: *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2024, pp. 13107–13114.

[29] Félix G Harvey et al. "Robust motion in-betweening". In: *ACM Transactions on Graphics (TOG)* 39.4 (2020), pp. 60–1.

[30] Sebastian Starke et al. "Local motion phases for learning multi-contact character movements". In: *ACM Transactions on Graphics (TOG)* 39.4 (2020), pp. 54–1.

[31] Wenyang Zhou et al. "Emdm: Efficient motion diffusion model for fast and high-quality motion generation". In: *European Conference on Computer Vision*. Springer. 2024, pp. 18–38.

[32] Jungdam Won, Deepak Gopinath, and Jessica Hodgins. "Physics-based character controllers using conditional vaes". In: *ACM Transactions on Graphics (TOG)* 41.4 (2022), pp. 1–12.

[33] Irina Higgins et al. "beta-vae: Learning basic visual concepts with a constrained variational framework." In: *ICLR (Poster)* 3 (2017).

[34] Junfeng Long et al. "Hybrid internal model: Learning agile legged locomotion with simulated robot response". In: *arXiv preprint arXiv:2312.11460* (2023).

[35] Lucy Lai, Ann Zixiang Huang, and Samuel J Gershman. "Action chunking as policy compression". In: (2022).

[36] Lu Wang et al. "Supervised reinforcement learning with recurrent neural network for dynamic treatment recommendation". In: *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 2018, pp. 2447–2456.

[37] Omid Taheri et al. "GRAB: A dataset of whole-body human grasping of objects". In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IV 16*. Springer. 2020, pp. 581–600.

[38] Haodong Zhang et al. "Kinematic motion retargeting via neural latent optimization for learning sign language". In: *IEEE Robotics and Automation Letters* 7.2 (2022), pp. 4582–4589.