TokenHSI: Unified Synthesis of Physical Human-Scene Interactions through Task Tokenization

Supplementary Material

A. Simulated Character

Character Model Creation. We apply a custom simulated character model, with 32 degrees-of-freedom (DoF). This custom model is based on the character model used in AMP [13], which comprises 15 rigid bodies, 12 controllable joints, and 28 DoF, as depicted in Fig. A (a). While retaining most designs of AMP's character model, we introduce three improvements:

- (1) The 3D relative positions of the lower body joints, including the hips, knees, and ankles, are adjusted to match those in the SMPL [8] human body model configured with a neutral gender and default shape parameters.
- (2) The collision shapes of the foot rigid bodies are modified from rectangular boxes to realistic foot meshes using the method proposed by SimPoE [17].
- (3) The knee joints are upgraded from 1-DoF revolute joints to 3-DoF spherical joints.

An illustration of our custom character model is available in Fig. A (b). The primary motivation for building this custom model is two fold: the reference motion datasets are represented using SMPL parameters, and the kinematic structure of AMP's character model is different from that of SMPL. Consequently, directly copying rotation parameters to retarget these motions onto AMP's character model leads to unnatural lower body motions. In contrast, using our improved character model, which features a lower body structure consistent with SMPL, can significantly reduce retargeting errors and ensure more natural character motions.

The designed simulated character is used for most tasks, except those involving stairs terrain, as illustrated in Fig. 4 (e) and (f). This difference is due to inaccurate contact simulation between the meshed foot rigid bodies and the terrain in IsaacGym [10]. To address this issue, we revert the collision shapes of foot rigid bodies back to rectangular boxes, which are more simulation-friendly, as shown in Fig. A (c).

The Proprioception s_t and Action a_t. The proprioception s_t describes the simulated state of the character at each time step t. Following ASE [14], s_t is constructed using a set of features, including the positions, rotations, linear velocities, and angular velocities of all rigid bodies. All features are expressed in the character's local coordinate frame, except for the root joint rotation, which is represented in the world coordinate frame. The 6D rotation representation [18] is employed. Notably, the root joint position is excluded from the proprioception. Combined, these features

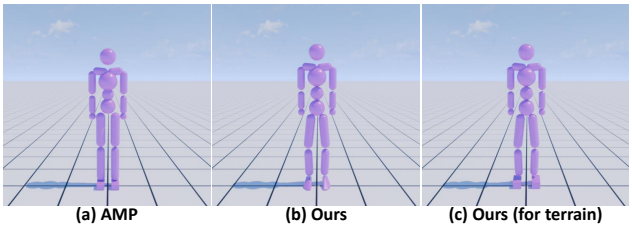

Figure A. Different simulated character models. Building on (a) AMP's model, we devise two improved versions: (b) and (c), which are used for tasks on flat ground and tasks on stairs terrain, respectively.

define a 222D humanoid proprioception $s_t \in \mathbb{R}^{222}$. At each time step t, the control policy generates an action a_t, representing the target rotations for the PD controllers at each of the character's degrees-of-freedom. The target rotations for 3D spherical joints are encoded using a 3D exponential map [3]. Our character model has ten 3-DoF spherical joints and two 1-DoF revolute joints (*i.e.*, left and right elbows), resulting in a 32D action space $a_t \in \mathbb{R}^{32}$. No external forces are applied to any rigid body of the simulated character.

B. Tasks

In this section, we provide the implementation details about all tasks involved in this paper. Tab. A presents an overview of all 12 tasks, including 4 foundational HSI tasks, 3 skill composition tasks, 4 object/terrain shape variation tasks, and 1 long-horizon task. We begin by introducing the common settings shared across tasks in Section B.1. Task-specific settings, such as task observations and reward functions, are detailed in the subsequent sections.

B.1. Preliminaries

Reference Motion Dataset. To encourage the character to perform tasks in a realistic and life-like manner, we manually construct a comprehensive reference motion dataset encompassing a wide variety of behavior categories associated with the four foundational HSI tasks. The dataset is divided into five distinct subsets:

- **Loco:** This subset includes 12 motion sequences from the AMASS [9] dataset, covering basic locomotion behaviors such as standing, walking, and turning around on flat ground. Since every task involves a walking stage, this subset is used for all tasks.
- **Stair:** This subset is used for tasks on stairs terrain and

Task	Num. of Task Tokens	Num. of Obj.		Reference Motion Dataset					Epis. Len. (s)	Early Termination Condition			
		Train	Test	Loco	Stair	Climb	Carry	Sit		Char. Fall	Obj. Fall	Path Dist.	IET
Follow	1	/	/	✓					10	✓		✓	
Sit	1	49	26	✓				✓	10	✓			✓
Climb	1	38	26	✓		✓			10	✓			✓
Carry	1	9	9	✓			✓		20	✓			
Follow + Carry	3	/ + 5	/ + 9	✓			✓		10	✓	✓	✓	
Sit + Carry	3	49 + 5	26 + 9	✓			✓	✓	10	✓	✓		✓
Climb + Carry	3	38 + 5	26 + 9	✓		✓	✓		10	✓	✓		✓
Obj. Shap. Var. (Chair)	1	63	27	✓			✓		20	✓	✓		
Obj. Shap. Var. (Table)	1	21	9	✓			✓		20	✓	✓		
Terr. Shap. Var. (Follow)	2	/	/	✓	✓				10	✓		✓	
Terr. Shap. Var. (Carry)	2	9	9	✓	✓		✓		20	✓			
Long-horizon Task	5	/	/	✓		✓	✓	✓	40	✓			✓

Table A. The overview of all 12 tasks implemented in this paper. Key settings for each task are summarized, including the number of task tokens, the construction of reference motion and object datasets, the episode length, and early termination conditions. The available termination conditions contain character fall, object fall, path distance, and interaction early termination (IET). A slash (/) indicates that the specific configuration is not applicable.

consists of 20 motion sequences for ascending and descending stairs.

- **Climb**: To support the training of the climbing task, we collect 11 motion sequences from the AMASS [9] dataset, where characters climb onto a high platform from the ground.
- **Carry:** We collect the carrying motions from hybrid sources, with 17 sequences from the OMOMO [7] dataset and 4 sequences from the AMASS [9] dataset.
- **Sit:** This subset consists of 20 sitting motions collected from the SAMP [4] dataset.

The usage of these five motion datasets in each task's training process is summarized in Tab. A. For skill composition tasks, such as sitting down while carrying an object, no post-processing is applied to merge the two corresponding subsets (*i.e.*, Carry and Sit) to obtain composite kinematic reference motions. Therefore, the reference motion dataset does not include any motions of sitting down while carrying a box. The policy learns these composite skills primarily through the guidance of task rewards. As the policy learns composite tasks, the style reward decreases while the task reward increases, leading to an overall increase in total reward and improved task completion.

Object Dataset. To make learned interaction skills effectively generalize to diverse unseen objects, we construct an object training dataset and a corresponding testing dataset to evaluate the generalization capabilities of these skills. The high-quality 3D object models are collected from the 3D-Front [2] object dataset, while the 3D models of boxes are procedurally generated using Trimesh. The number of objects used for training and testing is described in Table A. We carefully ensure all test are conducted on the unseen objects.

Early Termination Condition. Early termination is an effective technique for improving the reinforcement learning (RL) training process by preventing negative samples from adversely affecting the policy gradient [12]. A fundamental and widely applicable termination condition is humanoid fall detection, which is utilized for all tasks in our implementation. To further facilitate the learning of dynamic object-carrying skills, we introduce a similar condition called object fall detection. If the object's height drops below a specified threshold, the trial will be terminated. For the path following task, we adopt the path distance detection condition proposed in Trace&Pace [15]. If the 2D distance between the root of the simulated character and the target point on the trajectory at the current moment exceeds a specified threshold, the trial will be terminated. For tasks such as sitting and climbing, where the physical character enters a static interaction state with the object upon task completion, we introduce the Interaction Early Termination (IET) condition proposed by InterScene [11]. This effectively enhances smoothness and increases the success rate performance of these particular tasks.

B.2. Foundational HSI Tasks

B.2.1. Path Following

Definition. This task requires the simulated character to move along a target 2D trajectory. We follow the prior work [15] to procedurally generate the trajectory dataset. A whole trajectory is formulated as $\tau = \{x^\tau_{0.1}, x^\tau_{0.2}, ..., x^\tau_{T-0.1}, x^\tau_T\}$, where $x^\tau_{0.1}$ denotes a 2D waypoint of the trajectory τ at the simulation time $0.1s$, and T is the episode length. According to Tab. A, the path following task's episode length T is $10s$. The character needs to follow this trajectory τ accurately.

Task Observation. At each simulation moment t second, we query 10 future waypoints $\{x^\tau_t, x^\tau_{t+0.1}, ..., x^\tau_{t+0.8}, x^\tau_{t+0.9}\}$ in the future $1.0s$ from

the whole trajectory τ by linear interpolation. The sampling time interval is $0.1s$. We use the 2D coordinates of the sampled waypoints as the task observation $g_t^f \in \mathbb{R}^{2\times 10}$.

Task Reward. The task reward r_t^f calculates the distance between the current character 2D root position $x_t^{root\_2d}$ and the desired target waypoint x_t^τ:

$$r_t^f = \exp\big(-2.0 \left\| x_t^{root\_2d} - x_t^\tau \right\|^2\big). \tag{1}$$

B.2.2. Sitting

Definition. The task objective is for the character to move its root joint to a target 3D sitting position located on the object surface. The target position is placed 10 cm above the center of the top surface of the chair seat.

Task Observation. The sitting task observation $g_t^s \in \mathbb{R}^{38}$ includes the 3D target sitting position $\in \mathbb{R}^3$ and the 3D information of the interacting object, *i.e.*, the root position $\in \mathbb{R}^3$, the root rotation $\in \mathbb{R}^6$, and the 2D front-facing direction $\in \mathbb{R}^2$, as well as the positions of eight corner points on the object's bounding box $\in \mathbb{R}^{3\times 8}$.

Task Reward. The sitting policy is trained by minimizing the distance between the character's 3D root position x_t^{root} and the target 3D sitting position x_t^{tar}. The task reward r_t^s is defined as:

$$r_t^s = \begin{cases} 0.7\, r_t^{near} + 0.3\, r_t^{far}, \left\| x_t^{obj\_2d} - x_t^{root\_2d} \right\| > 0.5 \\ 0.7\, r_t^{near} + 0.3, \text{otherwise} \end{cases} \tag{2}$$

$$r_t^{far} = \exp\big(-2.0 \left\| 1.5 - d_t^* \cdot \dot{x}_t^{root\_2d} \right\|^2\big) \tag{3}$$

$$r_t^{near} = \exp\big(-10.0 \left\| x_t^{tar} - x_t^{root} \right\|^2\big), \tag{4}$$

where x_t^{root} is the 3D coordinates of the character's root, $\dot{x}_t^{root\_2d}$ is the 2D linear velocity of the character's root, $x_t^{obj\_2d}$ is the 2D position of the object root, d_t^* is a horizontal unit vector pointing from $x_t^{root\_2d}$ to $x_t^{obj\_2d}$, $a \cdot b$ represents vector dot product.

B.2.3. Climbing

Definition. In this work, we introduce a new contact-based interaction task similar to the sitting task. The goal is for the character to stand on a given object, placing its root joint at a target 3D climbing position. We place the target position 94 cm above the center of the top surface of the object.

Task Observation. The task observation $g_t^m \in \mathbb{R}^{27}$ includes the target root position $\in \mathbb{R}^3$ and the 3D coordinates of eight corner points on the object's bounding box $\in \mathbb{R}^{3\times 8}$.

Task Reward. This task is also optimized through minimizing the 3D distance between the character's root x_t^{root} and its target location x_t^{tar}. We formulate the task reward r_t^m, as follows:

$$r_t^m = \begin{cases} 0.5\, r_t^{near} + 0.2\, r_t^{far}, \left\| x_t^{obj\_2d} - x_t^{root\_2d} \right\| > 0.7 \\ 0.5\, r_t^{near} + 0.2 + 0.3\, r_t^{foot}, \text{otherwise} \end{cases} \tag{5}$$

$$r_t^{far} = \exp\big(-2.0 \left\| 1.5 - d_t^* \cdot \dot{x}_t^{root\_2d} \right\|^2\big) \tag{6}$$

$$r_t^{near} = \exp\big(-10.0 \left\| x_t^{tar} - x_t^{root} \right\|^2\big) \tag{7}$$

$$r_t^{foot} = \exp\big(-50.0 \left\| (x_t^{tar\_h} - 0.94) - x_t^{foot\_h} \right\|^2\big), \tag{8}$$

where $x_t^{tar\_h}$ denotes the height component of the 3D target root position, x_t^{tar}, $(x_t^{tar\_h} - 0.94)$ represent the height of the top surface of the target object in the world coordinate, and $x_t^{foot\_h}$ denotes the mean height of the two foot rigid bodies. The reward function r_t^{foot} is introduced to encourage the character to lift its feet, which is applied when the character is close enough to the target object. We find it is crucial for the successful training of the climbing task.

B.2.4. Carrying

Definition. The character is directed to move a box from a randomly initial 3D location $x_t^{box\_init}$ to a target 3D location $x_t^{box\_tar}$. We use two thin platforms to support the box since the its initial and target heights are randomly generated.

Task Observation. The task observation $g_t^c \in \mathbb{R}^{42}$ comprises the following properties of the target box:

- Target location of the box $\in \mathbb{R}^3$
- Root position $\in \mathbb{R}^3$
- Root rotation $\in \mathbb{R}^6$
- Root linear velocity $\in \mathbb{R}^3$
- Root angular velocity $\in \mathbb{R}^3$
- Positions of 8 corner points on the bounding box $\in \mathbb{R}^{3\times 8}$

Task Reward. We implement the multi-stage task reward function proposed by InterPhys [5]. The first stage aims to encourage the character to walk to the initial box. The corresponding reward $r_t^{c\_walk}$ is defined as:

$$r_t^{c\_walk} = \begin{cases} 0.2, \left\| x_t^{obj\_2d} - x_t^{root\_2d} \right\| < 0.5 \\ 0.2 \exp\big(-5.0 \left\| 1.5 - d_t^* \cdot \dot{x}_t^{root\_2d} \right\|^2\big), \\ \qquad\qquad \text{otherwise} \end{cases} \tag{9}$$

where d_t^* is a horizontal unit vector pointing from $x_t^{root\_2d}$ to $x_t^{obj\_2d}$, $a \cdot b$ represents vector dot product. The second stage is to encourage the character to pick up and move the box to its target location. We utilize two reward functions to achieve this stage, *i.e.*, $r_t^{c\_carry}$ to calculate the 3D distance between the box current root position x_t^{obj} and its target location x_t^{tar}, and $r_t^{c\_pick}$ to calculate the 3D distance between the box position x_t^{obj} and the mean 3D position of the character's two hands x^{hand}. We define $r_t^{c\_carry}$ as follows:

$$r_t^{c\_carry} = \begin{cases} 0.2\, r_t^{near} + 0.2\, r_t^{far}, \\ \qquad \left\| x_t^{obj\_2d} - x_t^{tar\_2d} \right\| > 0.5 \\ 0.2\, r_t^{near} + 0.2, \text{otherwise} \end{cases} \tag{10}$$

$$r_t^{far} = \exp\big(-5.0 \left\| 1.5 - d_t^{\#} \cdot \dot{x}_t^{obj\_2d} \right\|^2\big) \tag{11}$$

$$r_t^{near} = \exp\big(-10.0 \left\| x_t^{tar} - x_t^{obj} \right\|^2\big), \tag{12}$$

where x_t^{obj} is the 3D coordinates of the box's root, $\dot{x}_t^{obj\_2d}$ is the 2D linear velocity of the box's root, $x_t^{obj\_2d}$ is the 2D position of the object root, $x_t^{tar\_2d}$ is the 2D coordinates of the box's target location, $d_t^{\#}$ is a horizontal unit vector pointing from $x_t^{obj\_2d}$ to $x_t^{tar\_2d}$, $a \cdot b$ represents vector dot product. The task reward $r_t^{c\_pick}$ to incentivize the character pick up the box using its hands, defined as follows:

$$r_t^{c\_pick} = \begin{cases} 0.0, \left\| x_t^{obj\_2d} - x_t^{root\_2d} \right\| > 0.7 \\ 0.2 \exp\big(-5.0 \left\| x_t^{obj} - x_t^{hand} \right\|^2\big), \\ \qquad\qquad \text{otherwise} \end{cases} \tag{13}$$

where x_t^{hand} denotes the mean 3D coordinates of the character's two hands. Additionally, we further design a reward function $r_t^{c\_put}$ to incentivize the character to put down the box at its target location accurately, which is formulated as:

$$r_t^{c\_put} = \begin{cases} 0.0, \left\| x_t^{obj\_2d} - x_t^{tar\_2d} \right\| > 0.1 \\ 0.2 \exp\big(-10.0 \left\| x_t^{obj\_h} - x_t^{tar\_h} \right\|^2\big), \\ \qquad\qquad \text{otherwise} \end{cases} \tag{14}$$

where $x_t^{obj\_h}$ denotes the hight of the current box and $x_t^{tar_h}$ represents the height of the target placing position. Therefore, the total reward function r_t^c for training the carrying skill can be formulated as:

$$r_t^c = r_t^{c\_walk} + r_t^{c\_carry} + r_t^{c\_pick} + r_t^{c\_put}. \tag{15}$$

B.3. Downstream HSI Tasks

In this section, we provide the details about how we implement the task observations and rewards used for training these more challenging HSI tasks.

Skill Composition. When learning the composite tasks using our policy adaptation, we reuse and freeze two relevant task tokenizers of foundational skills. Their task observations are illustrated in Sec B.2. Therefore, we mainly focus on describing how we construct the task configurations for these composite tasks.

Follow + Carry. The task observation g_t^{f+c} contains two parts: (1) the primary following task observation $g_t^f \in \mathbb{R}^{2\times10}$ and (2) the revised carrying task observation $g_t^{c\_revised} \in \mathbb{R}^{39}$, which excludes the target location of the box $\in \mathbb{R}^3$ because the carrying task is no longer the primary task. The final composite task observation is $g_t^{f+c} \in \mathbb{R}^{2\times10+39}$ that considers both the primary task states and the carrying box states. We define the task reward for this composite task r_t^{f+c} as follows:

$$r_t^{f+c} = \begin{cases} 0.0, \left\| x_t^{obj\_2d} - x_t^{root\_2d} \right\| > 0.7 \\ 0.5\, r_t^f + 0.5\, r_t^{c\_pick}, \text{otherwise} \end{cases} \tag{16}$$

where the r_t^f is the same as Equ. 1 and the $r_t^{c\_pick}$ is equal to Equ. 13.

Sit + Carry. The task observation g_t^{s+c} also includes two parts: (1) the primary sitting task observation $g_t^s \in \mathbb{R}^{38}$ and (2) the revised carrying task observation $g_t^{c\_revised} \in \mathbb{R}^{39}$, which have been introduced before. The final composite task observation is $g_t^{s+c} \in \mathbb{R}^{38+39}$. We define the task reward for this composite task r_t^{s+c} as follows:

$$r_t^{s+c} = \begin{cases} 0.0, \left\| x_t^{obj\_2d} - x_t^{root\_2d} \right\| > 0.7 \\ 0.7\, r_t^s + 0.3\, r_t^{c\_pick}, \text{otherwise} \end{cases} \tag{17}$$

where the r_t^s is the same as Equ. 2 and the $r_t^{c\_pick}$ is equal to Equ. 13.

Climb + Carry. The task observation g_t^{m+c} also includes two parts: (1) the primary climbing task observation $g_t^m \in \mathbb{R}^{27}$ and (2) the revised carrying task observation $g_t^{c\_revised} \in \mathbb{R}^{39}$, which have been introduced before. The final composite task observation is $g_t^{m+c} \in \mathbb{R}^{27+39}$. We define the task reward for this composite task r_t^{m+c} as follows:

$$r_t^{m+c} = \begin{cases} 0.0, \left\| x_t^{obj\_2d} - x_t^{root\_2d} \right\| > 0.7 \\ 0.7\, r_t^m + 0.3\, r_t^{c\_pick}, \text{otherwise} \end{cases} \tag{18}$$

where the r_t^m is the same as Equ. 5 and the $r_t^{c\_pick}$ is equal to Equ. 13.

Object/Terrain Shape Variation. For object shape variation, we directly fine-tune the pre-trained box-carrying task tokenizer $\mathbb{T}^c$. That is, the task observation is still $g_t^c \in \mathbb{R}^{42}$. And we reuse the box-carrying reward function Equ. 15. For terrain shape variation, we introduce an additional task tokenizer for perceiving the surrounding height map, which use 1024 sensor points to represent the height values in a $2\times2\,m^2$ square area centered at the humanoid root position. Thus, the new height map observation is $g_t^{new} \in \mathbb{R}^{1024}$. We also reuse the task observation and reward function of the box-carrying task for terrain shape variation.

Long-horizon Task Completion. As illustrated in Fig. 4 (g), we sequence the four learned foundational skills to perform a long-horizon task in a complex environment. We reuse all observations of foundational skills and introduce a new height map observation $g_t^{new} \in \mathbb{R}^{625}$, which utilizes 625 sensor points to observe the heights in a $1\times1\,m^2$ square area. We design a step-by-step task reward mechanism. For each step in the task sequence, we reuse the task reward from the corresponding task ($(r_t^f, r_t^s, r_t^m$, or $r_t^c)$). Once a step is completed, the reward value for that sub-task is set to its maximal value, indicating the task have been accomplished. Then, the task reward for the next step in the sequence will be activated for reward calculation. Please refer to our publicly released code for more details.

C. Implementation Details of CML

In Sec. 4.2.1, we compare our transformer-based policy adaptation with CML [16] and CML (dual) on the skill composition tasks. The network structure of CML (dual) is an improved version based on the original CML framework.

CML [16] employs a hierarchical framework consisting of a pre-trained, fixed meta policy π^{meta} as the low-level controller and a newly introduced, trainable policy π^{new} as the high-level controller. The high-level policy π^{new} observes the humanoid proprioception s_t and the new task observation g_t^{new}. The low-level policy π^{meta} observes the humanoid proprioception s_t and the base task observation g_t^{base}. Take the *Climb + Carry* task as an example. We use a specialist policy trained on the climbing task as the $\pi^{meta}(a_t^{meta}|s_t, g_t^m)$, which possesses a joint character-goal state space. Then, we introduce a new policy $\pi^{new}(a_t^{new}, w_t^{new}|s_t, g_t^{m+c})$, which generates a new action a_t^{new} and a group of joint-wise weights $w_t^{new} \in \mathbb{R}^{32}$, each value is $\in [0, 1]$. The high-level policy π^{new} is trained to cooperate with the low-level policy π^{meta} to quickly learn the composite tasks. The composition process is conducted in the action space as follows:

$$a_t = a_t^{new} + w_t^{new} a_t^{meta}, \tag{19}$$

which is called post-composition in the main paper.

However, the original CML framework supports only a single meta policy. To ensure a fair comparison, we develop CML (dual), an improved version that can simultaneously utilize two meta policies π_1^{meta} and π_2^{meta}. To handle the two sets of actions $a_t^{meta_1}$ and $a_t^{meta_2}$, generated by the two meta policies, the high-level policy π^{new} outputs an additional set of weights. In this way, we obtain $w_t^{new_1}$ and $w_t^{new_2}$. This results in the following post-composition process:

$$a_t = a_t^{new} + w_t^{new_1} a_t^{meta_1} + w_t^{new_2} a_t^{meta_2}, \tag{20}$$

where $w_t^{new_1}$ and $w_t^{new_2}$ are joint-wise weights applied to the two sets of meta actions, $a_t^{meta_1}$ and $a_t^{meta_2}$, respectively. All weights are processed using sigmoid activations, transforming their values to $[0, 1]$.

D. Quantitative Evaluation on Long-horizon Task Completion

Experimental Setup. We first describe the construction of the long-horizon task shown in Fig. 4 (g). The long task comprises four sequential sub-tasks: follow a target trajectory → carry a box to its target location → climb onto the box → sit on a chair located on the high platform. Each subtask should have a sub-goal. The finite state machine monitors the task executing process using the spatial relationship between the reference point (the humanoid root joint or the box centroid) and the sub-goal. Each sub-goal is procedurally generated—follow: the trajectory is planned by A*; carry: the target box position is placed close to the platform using a rule-based method; climb and sit: the character's target root position is pre-defined on the object geometry. We use the completed sub-task count as the evaluation metric. The maximal value is 4 in our case. We collect 512 trials to statistic the metrics.

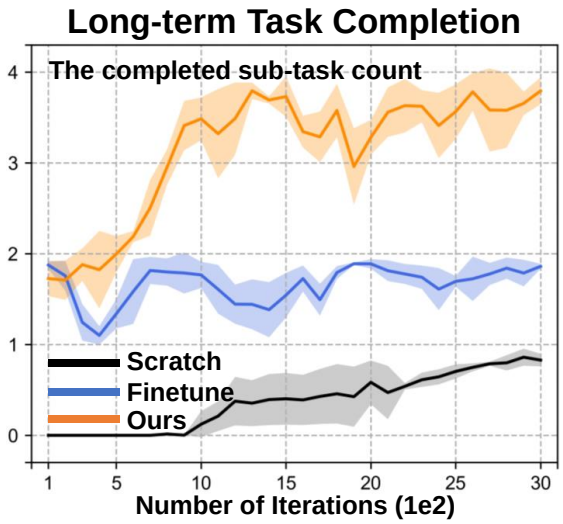

Figure B. Learning curves comparing the efficiency on long-horizon task completion using TokenHSI, Scratch [13], and iterative fine-tuning of multiple pre-trained specialist policies, namely Finetune.

Baselines. We compare our approach with two baseline methods: (1) Scratch [13]: training a policy to learn the whole long-term task from scratch; (2) Finetune [1, 6]: iterative fine-tuning multiple specialist policies in the environment to improve skill transitions and collision avoidance. We use our transformer policy as the policy architecture when conducting the experiment Scratch. Both Scratch and TokenHSI can observe height map. The difference between these two approaches is that our approach utilized pre-trained parameters. Due to the limited flexibility of MLP-based policies used by the experiment Finetune, we cannot make them environment-aware. The training adopts $1,024$ parallel simulation environments and $3k$ PPO iterations.

Quantitative Results. Our method achieves the highest value of the completed sub-task count of 3.79 ± 0.14, significantly outperforming Scratch 0.82 ± 0.06 and Finetune 1.86 ± 0.02. We also illustrate the convergence curves in Fig. B, which shows that TokenHSI still maintains its efficiency advantage in the long-horizon task completion.

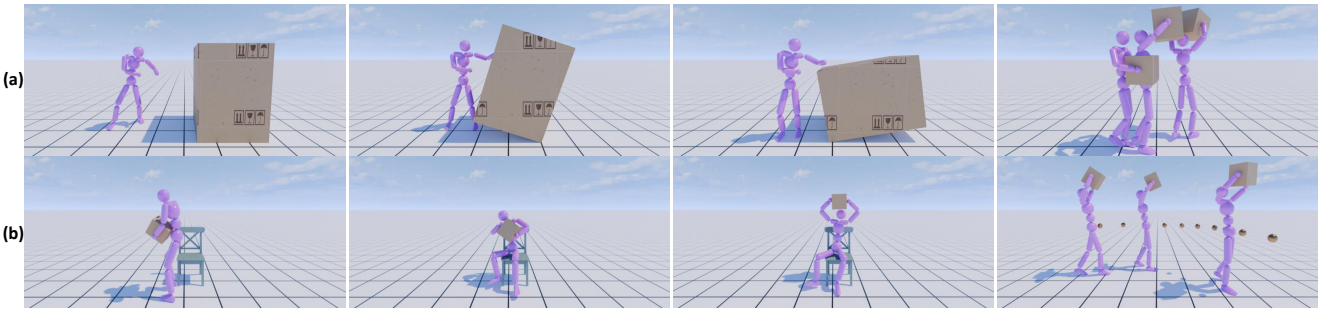

Figure C. Qualitative results of new skills learned by our policy adaptation. (a) We first learn two out-of-domain interaction skills, *i.e.*, pushing down a large object and walking to a target location while lifting up a box using two hands. (b) We then combine the new lifting skill with previously learned sitting and path-following skills. These results demonstrate the good extensibility of our transformer policy.

E. Extensibility

In the main paper, we mainly focus on adapting skills learned in the first stage (*i.e.*, foundational skill learning) to address more challenging HSI tasks through policy adaptation. In this section, we want to evaluate the extensibility of our approach to more HSI skills. We attempt to answer two questions: (1) Can we insert out-of-domain skills into the pre-trained transformer policy? (2) Can we further combine the newly-added skills with previously learned foundational skills to create more compositional cases? The flexibility of the transformer policy allows us to explore these problems.

Q1: Out-of-domain Skill Insertion. We consider two more types of manipulation skills, including pushing down a large object and walking to a target location while lifting up a box using two hands. To prepare the training, we collect the reference motions from the AMASS dataset [9], design the task observations, rewards, and other environmental configurations. During training, we introduce a randomly initialized task tokenizer $\mathbb{T}^{new}$ and zero-initialized adapter layers to the action head $\mathbb{H}$. The rest network parameters are frozen. For the pushing task, we declare it to be successful if the object falls. For the lifting task, we determine a testing trial to be successful if the pelvis is within 20 cm (XY-planar distance) of the target location while maintaining the box in a lifted position. As shown in Fig. C (a), our approach successfully synthesizes these out-of-domain manipulation skills. Specifically, the pushing task attains a success rate of 100% and the lifting task receives $80.6\% \pm 2.6$.

Q2: More Compositional Cases. Moreover, we combine the newly learned box-lifting skill with previous sitting and following skills. The training method is the same as skill composition. Through policy adaptation, we create more compositional cases shown in Fig. C (b). The success rates are $72.1\% \pm 2.8$ and $91.1\% \pm 0.8$, respectively.

References

[1] Alexander Clegg, Wenhao Yu, Jie Tan, C Karen Liu, and Greg Turk. Learning to dress: Synthesizing human dressing motion via deep reinforcement learning. *ACM Transactions on Graphics (TOG)*, 37(6), 2018. 5

[2] Huan Fu, Bowen Cai, Lin Gao, Ling-Xiao Zhang, Jiaming Wang, Cao Li, Qixun Zeng, Chengyue Sun, Rongfei Jia, Binqiang Zhao, et al. 3d-front: 3d furnished rooms with layouts and semantics. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10933–10942, 2021. 2

[3] F Sebastian Grassia. Practical parameterization of rotations using the exponential map. *Journal of graphics tools*, 3(3): 29–48, 1998. 1

[4] Mohamed Hassan, Duygu Ceylan, Ruben Villegas, Jun Saito, Jimei Yang, Yi Zhou, and Michael J Black. Stochastic scene-aware motion prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11374–11384, 2021. 2

[5] Mohamed Hassan, Yunrong Guo, Tingwu Wang, Michael Black, Sanja Fidler, and Xue Bin Peng. Synthesizing physical character-scene interactions. In *ACM SIGGRAPH 2023 Conference Proceedings*, pages 1–9, 2023. 3

[6] Youngwoon Lee, Joseph J Lim, Anima Anandkumar, and Yuke Zhu. Adversarial skill chaining for long-horizon robot manipulation via terminal state regularization. In *5th Annual Conference on Robot Learning*, 2021. 5

[7] Jiaman Li, Jiajun Wu, and C Karen Liu. Object motion guided human motion synthesis. *ACM Transactions on Graphics (TOG)*, 42(6):1–11, 2023. 2

[8] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A skinned multi-person linear model. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, 34(6):248:1–248:16, 2015. 1

[9] Naureen Mahmood, Nima Ghorbani, Nikolaus F Troje, Gerard Pons-Moll, and Michael J Black. Amass: Archive of motion capture as surface shapes. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5442–5451, 2019. 1, 2, 6

[10] Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller,

Nikita Rudin, Arthur Allshire, Ankur Handa, et al. Isaac gym: High performance gpu-based physics simulation for robot learning. *arXiv preprint arXiv:2108.10470*, 2021. 1

[11] Liang Pan, Jingbo Wang, Buzhen Huang, Junyu Zhang, Haofan Wang, Xu Tang, and Yangang Wang. Synthesizing physically plausible human motions in 3d scenes. In *2024 International Conference on 3D Vision (3DV)*, pages 1498–1507. IEEE, 2024. 2

[12] Xue Bin Peng, Pieter Abbeel, Sergey Levine, and Michiel van de Panne. Deepmimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM Transactions on Graphics (TOG)*, 37(4), 2018. 2

[13] Xue Bin Peng, Ze Ma, Pieter Abbeel, Sergey Levine, and Angjoo Kanazawa. Amp: Adversarial motion priors for stylized physics-based character control. *ACM Transactions on Graphics (ToG)*, 40(4):1–20, 2021. 1, 5

[14] Xue Bin Peng, Yunrong Guo, Lina Halper, Sergey Levine, and Sanja Fidler. Ase: Large-scale reusable adversarial skill embeddings for physically simulated characters. *ACM Transactions On Graphics (TOG)*, 41(4):1–17, 2022. 1

[15] Davis Rempe, Zhengyi Luo, Xue Bin Peng, Ye Yuan, Kris Kitani, Karsten Kreis, Sanja Fidler, and Or Litany. Trace and pace: Controllable pedestrian animation via guided trajectory diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13756–13766, 2023. 2

[16] Pei Xu, Xiumin Shang, Victor Zordan, and Ioannis Karamouzas. Composite motion learning with task control. *ACM Transactions on Graphics (TOG)*, 42(4):1–16, 2023. 5

[17] Ye Yuan, Shih-En Wei, Tomas Simon, Kris Kitani, and Jason Saragih. Simpoe: Simulated character control for 3d human pose estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7159–7169, 2021. 1

[18] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5745–5753, 2019. 1