LEARNING TO SIT: SYNTHESIZING HUMAN-CHAIR INTERACTIONS VIA HIERARCHICAL CONTROL

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent progress on physics-based character animation has shown impressive breakthroughs on human motion synthesis, through imitating motion capture data via deep reinforcement learning. However, results have mostly been demonstrated on imitating a single distinct motion pattern, and do not generalize to interactive tasks that require flexible motion patterns due to varying human-object spatial configurations. To bridge this gap, we focus on one class of interactive tasks—sitting onto a chair. We propose a hierarchical reinforcement learning framework which relies on a collection of subtask controllers trained to imitate simple, reusable mocap motions, and a meta controller trained to execute the subtasks properly to complete the main task. We experimentally demonstrate the strength of our approach over different single level and hierarchical baselines. We also show that our approach can be applied to motion prediction given an image input. A video highlight can be found at `https://youtu.be/XWU3wzz1ip8`.

1 INTRODUCTION

The capability of synthesizing realistic human-scene interactions is an important basis for simulating human living space, where robots can be trained to collaborate with humans, e.g. avoiding collisions or expediting the completion of assistive tasks.

Motion capture (mocap) data, by offering high quality recordings of articulated human pose, has provided a crucial resource for human motion synthesis. With large mocap datasets and deep learning algorithms, kinematics-based approaches have recently made rapid progress on motion synthesis and prediction (Fragkiadaki et al., 2015; Jain et al., 2016; Holden et al., 2016; Ghosh et al., 2017; Bütepage et al., 2017; Martinez et al., 2017; Holden et al., 2017; Zhou et al., 2018; Li et al., 2018; Gui et al., 2018a;b; Yan et al., 2018). However, the lack of physical interpretability in their synthesized motion has been a major limitation of these approaches. The problem becomes especially clear when it comes to motions that involve substantial human-object or human-human interactions. Without modeling the physics, the sythensized interactions are often physically unrealistic, e.g. body parts penetrating obstacles or not reacting to collision. This generally limits the use of these approaches to either non-interactive motions, or a carefully set up virtual scene with high fidelity to the captured one.

The graphics community has recently witnessed impressive progress on physics-based character animation (Peng et al., 2017; 2018a;b). These approaches, through imitating mocap examples via deep reinforcement learning, can synthesize realistic motions in physics simulated environments.

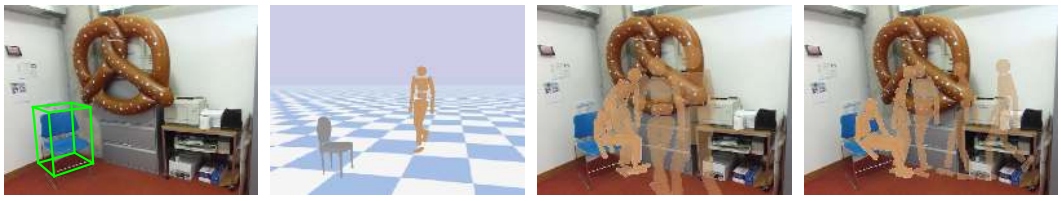

Figure 1: Synthesizing the motion of sitting. **Left:** Input image and 3D chair detection. **Middle:** Physics simulated environment for learning human-chair interactions. **Right:** Two examples of the synthesized motions.

Consequently, they can adapt to different physical contexts and thus attain a better generalization performance for interaction-based motions, e.g. walking on uneven terrain or stunt performance under obstacle disturbance. Nonetheless, these approaches still suffer from a drawback—a single model is trained for performing a single task with a distinct motion pattern (often time from a single mocap clip). As a result, they might not generalize to higher-level interactive tasks that require flexible motion patterns. Take the example of a person sitting down on a chair. A person can start in any location and orientation relative to the chair (Fig. 1). A fixed motion pattern (e.g. turn left and sit) will be incapable of handling such variations.

In this paper, we focus on one class of high-level interactive tasks—sitting onto a chair. As earlier mentioned, there are many possible human-chair configurations and different configurations may require different sequences of actions to accomplish the goal. For example, if the human is facing the chair, it needs to walk, turn either left or right, and sit; if the human is behind the chair, it needs to walk, side-walk and sit. To this end, we propose a hierarchical reinforcement learning (RL) method to address the challenge of generalization. Our key idea is the use of hierarchical control: (1) we assume the main task (e.g. sitting onto a chair) can be decomposed into several subtasks (e.g. walk, turn, sit, etc.), where the motion of each subtask can be reliably learned from mocap data, and (2) we train a meta controller using RL which can execute the subtasks properly to "complete" the main task from a given configuration. Such strategy is in line with the observation that humans have a repertoire of motion skills, and different subset of skills is selected and executed for different high-level tasks.

Our contributions are three folds: (1) we extend the prior work on physics-based motion imitation to the context of higher-level interactive tasks using a hierarchical approach; (2) we experimentally demonstrate the strength of our hierarchical approach over different single level and hierarchical baselines; (3) we show at the end that our approach can be applied to motion synthesis in human living space with the help of 3D scene reconstruction.

2 Related Work

Kinematics-based Models Kinematic modeling of human motions has a substantial literature in both vision and graphics domains. Conventional methods such as motion graphs (Kovar et al., 2002) require a large corpus of mocap data and face challenges in generalizing to new behaviors in new context. Recent progress in deep learning enables researchers to explore more efficient algorithms to model human motions, again, from large-scale mocap data. The focus in the vision community is often motion prediction (Fragkiadaki et al., 2015; Jain et al., 2016; Ghosh et al., 2017; Bütepage et al., 2017; Martinez et al., 2017; Zhou et al., 2018; Li et al., 2018; Gui et al., 2018a;b; Yan et al., 2018; Villegas et al., 2018), where a sequence of mocap poses is given as historical observation and the goal is to predict future poses. Recent work has even started to predict motions directly from a static image (Chao et al., 2017; Walker et al., 2017; Yao et al., 2018). In the graphics community, the focus has been primarily on motion synthesis, which aims to synthesis realistic motions from mocap examples (Yamane et al., 2004; Agrawal & van de Panne, 2016; Holden et al., 2016; 2017). Regardless of the focus, this class of approaches still faces the challenge of generalization due to the lack of physical plausibility in the synthesized motion, e.g. foot sliding and obstacle penetrations.

Physics-based Models Physics simulated character animation has a long history in computer graphics (Liu et al., 2016; Liu & Hodgins, 2017; Peng et al., 2017; Liu & Hodgins, 2018; Peng et al., 2018a; Clegg et al., 2018; Peng et al., 2018b). Our work is most related to the recent work by Peng et al. (Peng et al., 2017; 2018a), which trained a virtual character to imitate mocap data using deep reinforcement learning. They demonstrated robust and realistic looking motions on a wide array of skills including locomotion and acrobatic motions. Notably, they have used a hierarchical model for the task of navigating on irregular terrain (Peng et al., 2017). However, their meta task only requires a single subtask (i.e. walk), and the meta controller focuses solely on steering. We address a more complex task (i.e. sitting onto a chair) which requires the execution of diverse subtasks (e.g. walk, turn, and sit). Another recent work that is closely related to ours is that of Clegg et al. (2018), which addressed the task of dressing also with a hierarchical model. However, their subtasks are executed in a pre-defined order, and the completion of subtasks is determined by hand-coded rules. In contrast, our meta controller is trained and is free to select any subtask at any time point. This is crucial when the main task cannot always be completed by a fixed order of subtasks.

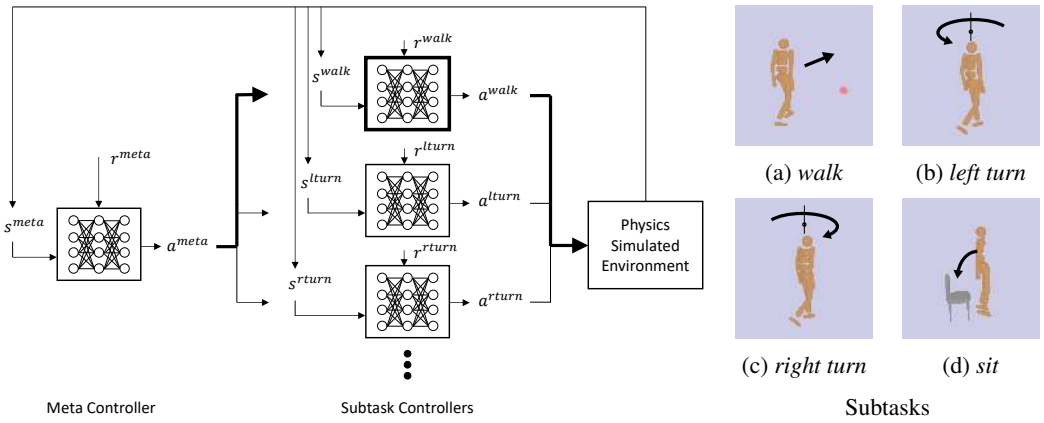

Figure 2: **Left:** Overview of the hierarchical system. **Right:** Illustration of the subtasks.

Note that humanoid control in physics simulated environments is also a widely-used benchmark task in the RL community, for example, to investigate how to ease the design of the reward function (Heess et al., 2017; Merel et al., 2017). However, work in this domain focuses less on realistic motions.

Hierarchical Reinforcement Learning Our model is inspired by a series of recent work on hierarchical control in deep reinforcement learning (Heess et al., 2016; Kulkarni et al., 2016; Tessler et al., 2017). Although in different contexts, they share the same attribute that the tasks of concern have high-dimensional action space, but can be decomposed into simpler, reusable subtasks. Such decomposition may even help in generalizing to new high-level tasks due to the shared subtasks.

Object Affordances Our work is connected to the learning of object affordances in the vision domain. Affordances express the functionality of objects and how humans can interact with them. Prior work attempted to detect affordances of a scene, represented as a set of plausible human poses, by training on large video corpora (Delaitre et al., 2012; Zhu et al., 2015; Wang et al., 2017). Instead, we learn the motion in a physics simulated environment using limited mocap examples and reinforcement learning. Another relevant work also detected affordances using mocap data (Gupta et al., 2011), but focused only on static pose rather than motion.

3 Overview

Our main task is the following: given a chair and a skeletal pose of a human in the 3D space, generate a sequence of skeletal poses that describes the motion of the human sitting onto the chair from the given pose (Fig. 1). Our system builds upon a physics simulated environment which contains an articulated structured humanoid and a rigid body chair model. Each joint of the humanoid (except the root) can receive a control signal and produce dynamics from the physics simulation. The goal is to learn a policy that controls the humanoid to successfully sit on the chair.

Fig. 2 (left) illustrates the hierarchical architecture of our policy. At the lower level is a set of subtask controllers, each responsible for generating the control input of a particular subtask. As illustrated in Fig. 2 (right), we consider four subtasks: *walk*, *left turn*, *right turn*, and *sit*. [1] To synthesize realistic motions, the subtask policies are trained on mocap data to imitate real human motions. At the higher level, a meta controller is responsible for controlling the execution of subtasks to ultimately accomplish the main task. The subtask controllers and meta controller generate control input at different timescales—60 Hz for the former and 2Hz for the latter. The physics simulation runs at 240 Hz. Each subtask as well as the meta controlling task is formulated as an independent reinforcement learning problem. We leverage recent progress in deep RL and approximate each policy using a neural network.

[1] We consider quick, in-place *turns*, which is distinct from moderate angled steering during walking. *sit* is also an in-place motion and should be distinguished from the main sitting task that involves locomotion.

4 SUBTASK CONTROLLER

A subtask controller is a policy network $\pi(a_t|s_t)$ that maps a state vector s_t to an action a_t at each timestep t. The state representation s is extracted from the current configuration of the simulation environment, and may vary for different subtasks. For example, *turn* requires only proprioceptive information of the humanoid, while *sit* requires not only such information, but also the pose of the chair relative to the humanoid. The action a is the signal for controlling the humanoid joints for each subtask. We use a humanoid model with 21 degrees of freedom, i.e. $a \in \mathbb{R}^{21}$. The network architecture is fixed across the subtasks: we use a multi-layer perceptron with two hidden layers of size 64. The output of the network parameterizes the probability distribution of a, modeled by a Gaussian distribution with a fixed diagonal covariance matrix, i.e. $\pi(a|s) = \mathcal{N}(\mu(s), \Sigma)$ and $\Sigma = diag(\{\sigma_i\})$. We can generate a_t at each timestep by sampling from $\pi(a_t|s_t)$.

Each subtask is formulated as an independent RL problem. At timestep t, the state s_t given by the simulation environment is fed into the policy network to output an action a_t. The action a_t is then fed back to the simulation environment to generates the state s_{t+1} at the next timestep and a reward signal r_t. The design of the reward function is crucial and plays a key role in shaping the style of the humanoid's motion. A heuristically crafted reward may yield a task achieving policy, but may result in unnatural looking motions and behaviors (Heess et al., 2017). Inspired by Peng et al. (2018a), we set the reward function of each subtask by a sum of two terms that simultaneously encourages the imitation of the mocap reference and the achievement of the task objectives:

$$r^{sub} = r^S + r^G. \tag{1}$$

r^S and r^G account for the similarly to the reference motion and the achievement of the subtask goals, respectively. We use a consistent similarity reward r^S across all subtasks:

$$r^S = \omega^p r^p + \omega^v r^v, \tag{2}$$

where r^p and r^v encourage the similarity of local joint angles q_j and velocities $\dot{q}_j$ between the humanoid and the reference motion, and ω^p and ω^v are the respective weights. Specifically,

$$\begin{aligned} r^p &= \exp\left(-\alpha^p \sum_j d(q_j, \hat{q}_j)^2\right) \\ r^v &= \exp\left(-\alpha^v \sum_j (\dot{q}_j - \hat{\dot{q}}_j)^2\right), \end{aligned} \tag{3}$$

where $d(\cdot,\cdot)$ computes the angular difference between two angles. We empirically set $\omega^p = 0.5$, $\omega^v = 0.05$, $\alpha^p = 1$, and $\alpha^v = 10$. Next, we detail the state representation s and task objective reward r^G for each subtask.

1) Walk The state $s^{walk} \in \mathbb{R}^{52}$ consists of a 50-d proprioceptive feature and a 2-d goal feature that specifies an intermediate walking target. The proprioceptive feature includes the local joint angles and velocities, the height and linear velocity of the root (i.e. torso) as well as its pitch and roll angles, and a 2-d binary vector indicating the contact of each foot with the ground (Fig. 3). Rather than walking in random directions, target-directed locomotion (Agrawal & van de Panne, 2016) is necessary for accomplishing high-level tasks. Assuming a target is given, represented by a 2D point on the ground plane, the 2-d goal feature is given by $[\sin(\psi), \cos(\psi)]^\top$, where ψ is the azimuth angle to the target in the humanoid centric coordinates. The generation of targets will be detailed in the meta controller section (Sec. 5).

We observe that it is challenging to directly train a target-directed walking policy with mocap examples. Therefore we adopt a two-stage training strategy where each stage uses a distinct task objective reward. In the first stage, we encourage similar steering patterns to the reference motion, i.e. the linear velocity of the root $v \in \mathbb{R}^3$ should be similar between the humanoid and reference motion:

$$r^G = 0.5 \cdot \exp\left(-10 \cdot \sum_i (v_i - \hat{v_i})^2\right). \tag{4}$$

In the second stage, we reward motion towards the target:

$$r^G = 0.1 \cdot \frac{1}{1 + \exp(10 \cdot V^{walk})}, \tag{5}$$

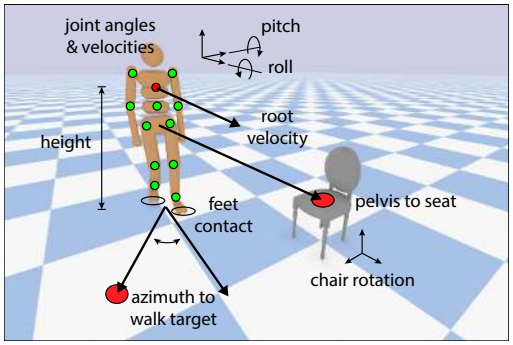

Figure 3: State representation of the humanoid and chair. The red and green dots on the humanoid denote the root and non-root joints. The red dots on the ground and chair denote the walk target and the center of the seat surface.

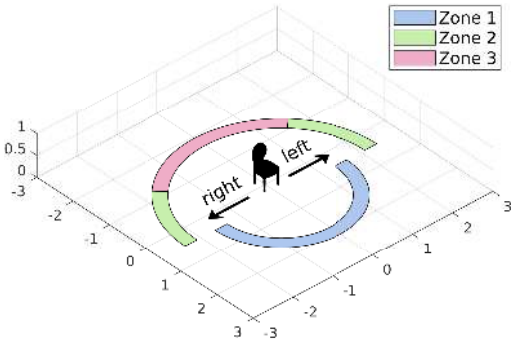

Figure 4: Curriculum learning for the meta controller. The humanoid spawn location is initially set to less challenging states (Zone 1), and later moved to more challenging states (Zone 2 and 3).

where $V^{walk} = (D^{walk}_{t+1} - D^{walk}_t)/\delta t$. D^{walk}_t denotes the horizontal distance between the root and the target, and δt is the length of the timestep.

2) Left/Right Turn The states s^{lturn}, $s^{rturn} \in \mathbb{R}^{50}$ reuse the 50-d proprioceptive feature from the walk subtask. The task objective reward encourages the rotation of the root to be matched between the humanoid and reference motion:

$$r^G = 0.1 \cdot \exp\left(-10 \cdot \sum_i d(\theta_i, \hat{\theta}_i)^2\right), \tag{6}$$

where $\theta \in \mathbb{R}^3$ consists of the root's pitch, yaw, and roll.

3) Sit The sit subtask assumes that the humanoid is initially standing roughly in front of the chair and facing away. The task is simply to lower the body and be seated. Different from *walk* and *turn*, the state for *sit* should capture the pose information of the chair. Our state $s^{sit} \in \mathbb{R}^{57}$ consists of the same 50-d proprioceptive feature used in *walk* and *turn*, and additionally a 7-d feature describing the state of the chair in the humanoid centric coordinates. The 7-d chair state includes the displacement vector from the pelvis to the center of the seat surface, and the rotation of the chair in the humanoid centric coordinates represented as a quaternion (Fig. 3). The task objective reward encourages the pelvis to move towards the center of the seat surface:

$$r^G = 0.5 \cdot (-V^{sit)}, \tag{7}$$

where $V^{sit} = (D^{sit}_{t+1} - D^{sit}_t)/\delta t$ and D^{sit}_t is the 3D distance between the pelvis and the center of the seat surface.

5 Meta Controller

The meta controller is also a policy network and shares the same architecture as the subtask controllers. As the goal now is to navigate the humanoid to sit on the chair, the input state s^{meta} should encode the pose information of the chair. We reuse the 57-d state representation from the sit subtask which contains both the proprioceptive and chair information. Rather than directly controlling the humanoid joints, the output action a^{meta} now controls the execution of subtasks. Specifically, $a^{meta} = \{a^{switch}, a^{target}\}$ consists of two components. $a^{switch} \in \{$*walk*, *left turn*, *right turn*, *sit*$\}$ is a discrete output which at each timestep picks a single subtask out of the four to execute. $a^{target} \in \mathbb{R}^2$ specifies the 2D target for the walk subtask, which is used to compute the goal state in s^{walk}. Note that a^{target} is only used when the walk subtask is picked for execution. The output of the policy network parameterizes the probability distributions of both a^{switch} and a^{target}, where a^{switch} is modeled by a categorical distribution as in standard classification problems, and a^{target} is modeled by a Gaussian distribution following the subtask controllers.

The meta task is also formulated as an independent RL problem. At timestep t, the policy network takes the state s^{meta}_t from the simulation environment and output an action a^{meta}_t. a^{meta}_t then

triggers one specific subtask controller to generate the control signal for the humanoid joints. The control signal is finally fed back to the simulation to generate the next state s_{t+1}^{meta} and a reward r_t^{meta}. Rather than evaluating the similarity to a mocap reference, the reward now should be providing feedback on the main task. We adopt a reward function that encourages the pelvis to move towards and be in contact with the seat surface:

$$r^{meta} = \begin{cases} 1 & \text{if } z_{\text{contact}} = 1 \\ 0.5 \cdot (-V^{sit}) & \text{otherwise.} \end{cases} \tag{8}$$

z_{contact} indicates whether the pelvis is in contact with the seat surface, which can be detected by the physics simulator. V^{sit} is defined as in Eq. 7.

6 TRAINING

Since the subtasks and meta task are formulated as independent RL problems, they can be trained independently using standard RL algorithms. We first train each subtask controllers separately, and then train the meta controller using the trained subtask controllers. All controllers are trained in a standard actor-critic framework using the proximal policy optimization (PPO) algorithm (Schulman et al., 2017).

1) Subtask Controller The training of the subtasks is also divided into two stages. First, in each episode, we initialize the pose of the humanoid to the first frame of the reference motion, and train the humanoid to execute the subtask by imitating the following frames. We apply the early termination strategy (Peng et al., 2018a): an episode is terminated immediately if the height of the root falls below 0.78 meters for *walk* and *turn*, and 0.54 meters for *sit*. These thresholds are chosen according to the height of the humanoid. For *turn*, the episode is also terminated when the root's yaw angle differs from the reference motion for more than $45°$. For *walk*, we adopt the two-stage training strategy described in Sec. 4. In target-directed walking, we randomly sample a new 2D target in the front of the humanoid every 2.5 seconds or when the target is reached. For *sit*, the chair is placed at a fixed location behind the humanoid, and we use reference state initialization (Peng et al., 2018a) to facilitate training.

The training above enables the humanoid to perform the subtasks from the initial pose of the reference motion. However, this does not guarantee successful transitions between subtasks (e.g. *walk*→*turn*), which is required for the main task. Therefore in the second stage, we fine-tune the controllers by setting the initial pose to a sampled ending pose of another subtask, similar to the policy sequencing method in Clegg et al. (2018). For *turn* and *sit*, the initial pose is sampled from the ending pose of *walk* and *turn*, respectively.

2) Meta Controller Recall that the task is to have the humanoid sit down regardless of where it starts in the environment. The task's difficulty highly depends on the initial state: if it is already facing the seat, it only needs to turn and sit, while if it is behind the chair, it needs to first walk to the front and then sit down. Training can be challenging when starting from a difficult state, since the humanoid needs to by chance execute a long sequence of correct actions to receive the reward for sitting down. To facilitate training, we propose a multi-stage training strategy inspired by curriculum learning (Zaremba & Sutskever, 2014). The idea is to begin the training from easier states, and progressively increase the difficulty when the training converges. As illustrated in Fig. 4, we begin by only spawning the humanoid on the front side of the chair (Zone 1). Once trained, we change the initial position to the lateral sides (Zone 2) and continue the training. Finally, we train the humanoid to start from the rear side (Zone 3).

7 RESULTS

Reference Motion We collect mocap data from the CMU Graphics Lab Motion Capture Database (CMU). Tab. 1 shows the mocap clips we used for each subtask. We extract relevant motion segments and retarget the motion to our humanoid model. We use a 21-DoF humanoid model provided by the Bullet Physics SDK (Bullet). Motion retargeting is performed using a Jacobian-based inverse kinematics method (Holden et al., 2016).

Implementation Details Our simulation environment is based on OpenAI Roboschool (Schulman et al., 2017; OpenAI), which uses the Bullet physics engine (Bullet). We use a randomly selected

Subtask	Subject #	Trial #
walk	8	1, 4
left / right turn	69	13
sit	143	18

Table 1: Mocap clips adopted from the CMU MoCap database (CMU).

	Subtasks	Meta Task
`nsteps`	8192	64
`nminibatches`	32	8
`noptepochs`	4	2
`lr`	1×10^{-4}	1×10^{-4}

Table 2: Hyperparamters for PPO training.

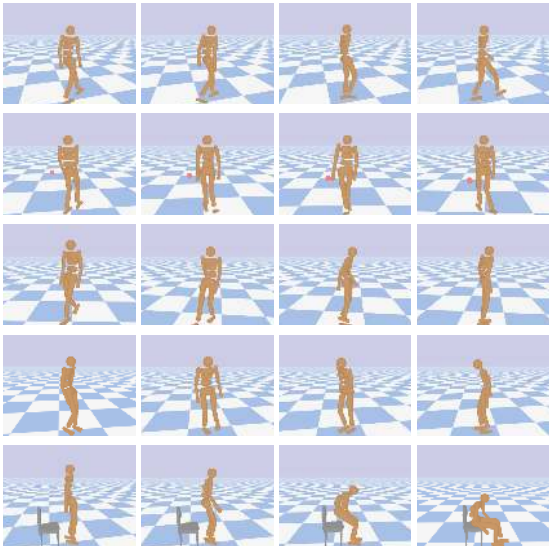

Figure 5: Execution of subtasks. From top to bottom: *forward walking*, *target directed walking*, *left turn*, *right turn*, and *sit*.

chair model from ShapeNet (Chang et al., 2015). The PPO algorithm for training is based on the implementation from OpenAI Baselines (Dhariwal et al., 2017). Tab. 2 shows the hyerparamters we used for the PPO training.

Subtask First we show qualitative results of the individual subtask controllers trained using their corresponding reference motions. Each row in Fig. 5 shows the humanoid performance of one particular subtask: walk in one direction (row 1), following a target (row 2), turn in place both left (row 3) and right (row 4), and sit on a chair (row 5).

Evaluation of Main Task We adopt two different metrics to quantitatively evaluate the main task: (1) *success rate* and (2) *minimum distance*. We declare a success whenever the pelvis of the humanoid has been continuously in contact with the seat surface for 3.0 seconds. We report the success rate over 10,000 trials by spawning the humanoid at random locations. Note that the success rate evaluates task completion with a hard constraint and does not reveal the progress when the humanoid fails. Therefore we also compute the per-trial minimum distance (in meters) between the pelvis and the center of the seat surface, and report the mean and standard deviation over the 10,000 trials.

As noted in Sec. 6, the task can be challenging when the initial position of the humanoid is unconstrained. To better analyze the performance, we consider two different initialization settings: (1) *Easy* and (2) *Hard*. In the Easy setting, the humanoid is initialized from roughly 2 meters away on the front half plane of the chair (i.e. Zone 1 in Fig. 4), with an orientation roughly towards the chair. The task is expected to be completed by simply walking forward, turning around, and sitting down. In the Hard setting, humanoid is initialized again from roughly 2 meters away but on the lateral and rear sides of the chair (i.e. Zone 2 and 3 in Fig. 4). It needs to walk around the chair to sit down successfully.

Easy Setting We benchmark our approach against various baselines in this setting. We start with two *non-hierarchical* (i.e. single-level) baselines. The first is a *kinematics*-based method: we select a mocap clip with a holistic motion sequence that successively performs walking, turning, and sitting on a chair. When a trial begins, we align the first frame of the sequence to the humanoid's initial pose by aligning the yaw of the root. Once aligned, we simply use the following frames of the sequence as the kinematic trajectory of the trial. Note that this method is purely kinematic and cannot reflect any physical interactions between the humanoid and chair. The second method extends the first one to a *physics*-based approach: we use the same kinematic sequence but now train a controller to imitate the motion. This is equivalent to training a subtask controller except the subtask is holistic (i.e. containing walk, turn, and sit in one reference motion). Both methods are considered non-hierarchical as neither performs task decomposition.

	Succ Rate (%)	Min Dist (m)
Kinematics	–	1.2656 ± 0.0938
Physics	0.00	1.3316 ± 0.1966
walk→left turn→sit	25.16	0.3790 ± 0.2326
walk→right turn→sit	0.92	0.7948 ± 0.2376
walk / left turn / sit	29.38	0.3913 ± 0.2847
walk / right turn / sit	23.01	0.3620 ± 0.2378
Full Model	**31.61**	$\mathbf{0.3303 \pm 0.2393}$

Table 3: Comparison of our approach with the hierarchical/non-hierarchical baselines in the Easy setting.

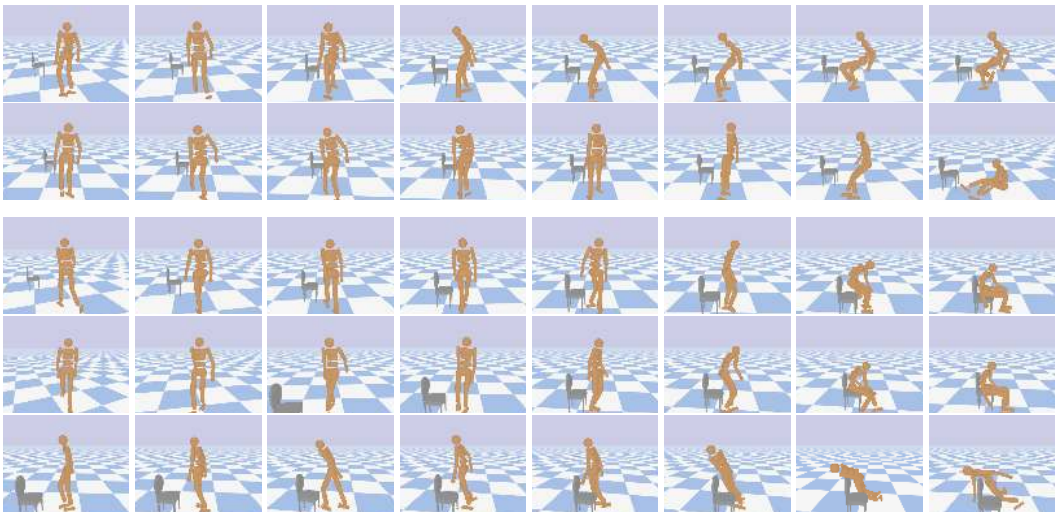

Figure 6: Qualitative results of our approach and the baselines. Row 1 and 2 show failure cases from the kinematics and physics baselines, respectively. The former violates physics rules (e.g. sitting in air), and both do not generalize to new human-chair configurations. Row 3 to 4 show two successful cases and row 5 shows one failure cases from our approach.

Tab. 3 shows the quantitative results. For the kinematics baseline, the success rate is not reported since we are unable to detect physical contact between the pelvis and chair. However, the 1.2656 mean minimum distance suggests that the humanoid on average remains far from the chair. For the physics baseline, we observe a similar mean minimum distance (i.e. 1.3316). The zero success rate is unsurprising given that the humanoid is unable to get close to the chair in most trials. As shown in the qualitative examples (Fig. 6), the motion generated by the kinematics baseline (row 1) is not physics realistic (e.g. sitting in air). The physics baseline (row 2), while following physics rules (e.g. falling on the ground eventually), still fails in approaching the chair. These holistic baselines perform poorly since they simply imitate the mocap example and repeat the same motion pattern regardless of their starting position.

We now turn to a set of *hierarchical* baselines and our approach. We also consider two baselines. The first one always executes the subtasks in a pre-defined order, and the meta controller is only used to trigger transitions (i.e. a binary classification). Note that this is in similar spirit to Clegg et al. (2018). We consider two particular orders: *walk→left turn→sit* and *walk→right turn→sit*. The second one is a degenerated version of our approach that uses either only *left turn* or *right turn*: *walk / left turn / sit* and *walk / right turn / sit*.

As shown in Tab. 3, hierarchical approaches outperform single level approaches, validating our hypothesis that hierarchical models, by breaking a task into reusable subtasks, can attain better generalization. Besides, our approach outperforms the pre-defined order baselines. This is because: (1) the main task cannot always be completed by a fixed order of subtasks, and (2) fixing the order increases training difficulty because certain missing transitions (e.g. *left turn→walk*) are necessary for recovery from mistakes. Finally, our full model outperforms the baselines that only allow turning in one direction. This suggests the two turning subtasks are complementary and being used in

Subtask	Initial Pose	Succ Rate (%)
left turn	mocap	87.02
right turn	mocap	67.59
sit	mocap	99.25

Subtask	Initial Pose	Succ Rate (%) w/o FT	Succ Rate (%) w/ FT
left turn	*walk*	0.09	51.12
right turn	*walk*	1.96	58.31
sit	*left* or *right turn*	32.94	87.41

Table 4: Evaluation of individual subtasks.

(a) left side (b) right side

Figure 7: Transition matrices of starting from different sides of the chair.

	Succ Rate (%)	Min Dist (m)
Zone 1	31.61	0.3303 ± 0.2393
Zone 2 w/o CL	0.00	0.5549 ± 0.2549
Zone 2	**10.01**	$\mathbf{0.5526 \pm 0.3303}$
Zone 3 w/o CL	4.05	0.5636 ± 0.2263
Zone 3 w/ CL	**7.05**	$\mathbf{0.5262 \pm 0.2602}$

Table 6: Comparison of the Easy and Hard settings. Applying the curriculum learning strategy improves the performance.

different scenarios, e.g. in Fig. 6, *walk→right turn→sit* when starting from the chair's right side (row 3), and *walk→left turn→sit* when starting from the chair's left side (row 4).

Analysis As can be seen in Tab. 3, the success rate is still low even with the full model (i.e. 31.61%). This can be attributed to three factors: (1) failures of subtask execution, (2) failures due to subtask transitions, and (3) an insufficient subtask repertoire. First, Tab. 4 (top) shows the success rate of individual subtasks, where the initial pose is set to the first frame of the reference motion (i.e. as in stage one of subtask training). We can see the execution does not always succeed (e.g. 67.59% for *right turn*). Second, Tab. 4 (bottom) shows the success rate for the same subtasks, but with the initial pose set to the last frame of the execution of another subtask (i.e. as in stage two of subtask training). With fine-tuning the success rate after transitions can be significantly improved, although still not perfect. Finally, Fig. 6 (row 5) shows a failure case where the humanoid needs a "back up" move when it is stuck in the state of directly confronting the chair. Building a more diverse subtask skill set is an interesting future research problem.

To analyze the meta controller's behavior, we look at the statistics on the switching between subtasks. Fig. 7 shows the subtask transition matrices when the humanoid is started either from the right or left side of the chair. We can see that certain transitions are more favored in certain starting areas, e.g. *walk→left turn* is favored over *walk→right turn* when started from the left side. This is in line with the earlier observation that the two turning subtasks are complementary.

Hard Setting We now increase the task's difficulty by initializing the humanoid in Zone 2 and 3 (Fig. 4), and show the effect of the proposed curriculum learning (CL) strategy. Tab. 6 shows the results from different initialization zones. First, we observe a severe drop in the success rate when the humanoid is spawned in Zone 2 and 3 (e.g. from 31.61% to 4.05% for "Zone 3 w/o CL"). However, the success rate is higher in both zones when the proposed curriculum learning strategy is applied (e.g. from 4.05% to 7.05% in Zone 3). This suggests that a carefully tailored curriculum can improve the training outcome of a challenging task. Note that the difference in the minimum distance is less significant (e.g. 0.5549 for "Zone 2 w/o CL' versus 0.5526 for "Zone 2"), since without CL the humanoid can still approach the chair, but will fail to turn and sit due to the difficulty in learning. Fig. 8 shows two successful examples when the humanoid is spawned from the rear side of the chair. Interestingly, the humanoid learns a slightly different behavior (e.g. *walk→sit* without *turn*) compared to when starting from the front side (row 3 and 4 in Fig. 6).

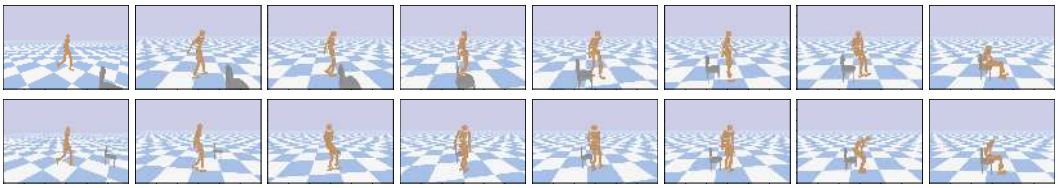

Figure 8: Qualitative results on the Hard setting. The humanoid can sit down successfully when starting from the back side of the chair.

Figure 9: Synthesizing sitting motions from a single image. The first column shows the 3D reconstruction output from Huang et al. (2018).

8 Motion Synthesis in Human Living Space

We show a vision-based application of our approach by synthesizing sitting motions from a single RGB image that depicts human living space with chairs. First, we recover the 3D scene configuration using the method of Huang et al. (2018). We then align the observed scene with the simulated environment using the detected chair and its estimated 3D position and orientation. This enables us to transfer the synthesized sitting motion to the observed scene. Fig. 9 shows two images rendered with synthesized humanoid motion. While the motion looks physically plausible in these examples, this is not always the case in general, since we do not model the other objects (e.g. tables) in the scene. An interesting future direction is to learn the motion by simulating scenes with cluttered objects. It is also possible to synthesize motions based on the humans observed in the image, given the recent advance on extracting 3D human pose from a single image (Peng et al., 2018b).

9 Conclusion

We address motion synthesis of an interactive task—sitting onto a chair. We introduce a hierarchical reinforcement learning approach which relies on a collection of subtask controllers trained to imitate reusable mocap motions, and a meta controller trained to execute the subtasks properly to complete the main task. We experimentally demonstrate the strength of our approach over different single level and hierarchical baselines, and show an application to motion prediction given an image.

References

Shailen Agrawal and Michiel van de Panne. Task-based locomotion. In *SIGGRAPH*, 2016.

Bullet. Bullet Physics SDK. `https://github.com/bulletphysics/bullet3`.

Judith Bütepage, Michael J. Black, Danica Kragic, and Hedvig Kjellström. Deep representation learning for human motion prediction and classification. In *CVPR*, 2017.

Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An information-rich 3D model repository. *arXiv preprint arXiv:1512.03012*, 2015.

Yu-Wei Chao, Jimei Yang, Brian Price, Scott Cohen, and Jia Deng. Forecasting human dynamics from static images. In *CVPR*, 2017.

Alex Clegg, Wenhao Yu, Jie Tan, C. Karen Liu, and Greg Turk. Learning to dress: Synthesizing human dressing motion via deep reinforcement learning. In *SIGGRAPH Asia*, 2018.

CMU. CMU Graphics Lab Motion Capture Database. http://mocap.cs.cmu.edu.

Vincent Delaitre, David F. Fouhey, Ivan Laptev, Josef Sivic, Abhinav Gupta, and Alexei A. Efros. Scene semantics from long-term observation of people. In *ECCV*, 2012.

Prafulla Dhariwal, Christopher Hesse, Oleg Klimov, Alex Nichol, Matthias Plappert, Alec Radford, John Schulman, Szymon Sidor, Yuhuai Wu, and Peter Zhokhov. OpenAI Baselines. https://github.com/openai/baselines, 2017.

Katerina Fragkiadaki, Sergey Levine, Panna Felsen, and Jitendra Malik. Recurrent network models for human dynamics. In *ICCV*, 2015.

Partha Ghosh, Jie Song, Emre Aksan, and Otmar Hilliges. Learning human motion models for long-term predictions. In *3DV*, 2017.

Liang-Yan Gui, Yu-Xiong Wang, Xiaodan Liang, and José M. F. Moura. Adversarial geometry-aware human motion prediction. In *ECCV*, 2018a.

Liang-Yan Gui, Yu-Xiong Wang, Deva Ramanan, and José M. F. Moura. Few-shot human motion prediction via meta-learning. In *ECCV*, 2018b.

Abhinav Gupta, Scott Satkin, Alexei A. Efros, and Martial Hebert. From 3D scene geometry to human workspace. In *CVPR*, 2011.

Nicolas Heess, Greg Wayne, Yuval Tassa, Timothy Lillicrap, Martin Riedmiller, and David Silver. Learning and transfer of modulated locomotor controllers. *arXiv preprint arXiv:1610.05182*, 2016.

Nicolas Heess, Dhruva TB, Srinivasan Sriram, Jay Lemmon, Josh Merel, Greg Wayne, Yuval Tassa, Tom Erez, Ziyu Wang, S. M. Ali Eslami, Martin Riedmiller, and David Silver. Emergence of locomotion behaviours in rich environments. *arXiv preprint arXiv:1707.02286*, 2017.

Daniel Holden, Jun Saito, and Taku Komura. A deep learning framework for character motion synthesis and editing. In *SIGGRAPH*, 2016.

Daniel Holden, Taku Komura, and Jun Saito. Phase-functioned neural networks for character control. In *SIGGRAPH*, 2017.

Siyuan Huang, Siyuan Qi, Yixin Zhu, Yinxue Xiao, Yuanlu Xu, and Song-Chun Zhu. Holistic 3D scene parsing and reconstruction from a single RGB image. In *ECCV*, 2018.

Ashesh Jain, Amir R. Zamir, Silvio Savarese, and Ashutosh Saxena. Structural-RNN: Deep learning on spatio-temporal graphs. In *CVPR*, 2016.

Lucas Kovar, Michael Gleicher, and Frédéric Pighin. Motion graphs. In *SIGGRAPH*, 2002.

Tejas D. Kulkarni, Karthik Narasimhan, Ardavan Saeedi, and Josh Tenenbaum. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. In *NIPS*. 2016.

Chen Li, Zhen Zhang, Wee Sun Lee, and Gim Hee Lee. Convolutional sequence to sequence model for human dynamics. In *CVPR*, 2018.

Libin Liu and Jessica Hodgins. Learning to schedule control fragments for physics-based characters using deep Q-learning. *ToG*, 36(3), Jun 2017.

Libin Liu and Jessica Hodgins. Learning basketball dribbling skills using trajectory optimization and deep reinforcement learning. In *SIGGRAPH*, 2018.

Libin Liu, Michiel van de Panne, and Kangkang Yin. Guided learning of control graphs for physics-based characters. *ToG*, 35(3):29:1–29:14, May 2016.

Julieta Martinez, Michael J. Black, , and Javier Romero. On human motion prediction using recurrent neural networks. In *CVPR*, 2017.

Josh Merel, Yuval Tassa, Dhruva TB, Sriram Srinivasan, Jay Lemmon, Ziyu Wang, Greg Wayne, and Nicolas Heess. Learning human behaviors from motion capture by adversarial imitation. *arXiv preprint arXiv:1707.02201*, 2017.

OpenAI. OpenAI Roboschool. `https://blog.openai.com/roboschool/`.

Xue Bin Peng, Glen Berseth, KangKang Yin, and Michiel van de Panne. DeepLoco: Developing locomotion skills using hierarchical deep reinforcement learning. In *SIGGRAPH*, 2017.

Xue Bin Peng, Pieter Abbeel, Sergey Levine, and Michiel van de Panne. DeepMimic: Example-guided deep reinforcement learning of physics-based character skills. In *SIGGRAPH*, 2018a.

Xue Bin Peng, Angjoo Kanazawa, Jitendra Malik, Pieter Abbeel, and Sergey Levine. SFV: Reinforcement learning of physical skills from videos. In *SIGGRAPH Asia*, 2018b.

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

Chen Tessler, Shahar Givony, Tom Zahavy, Daniel J. Mankowitz, and Shie Mannor. A deep hierarchical approach to lifelong learning in Minecraft. In *AAAI*, 2017.

Ruben Villegas, Jimei Yang, Duygu Ceylan, and Honglak Lee. Neural kinematic networks for unsupervised motion retargetting. In *CVPR*, 2018.

Jacob Walker, Kenneth Marino, Abhinav Gupta, and Martial Hebert. The pose knows: Video forecasting by generating pose futures. In *ICCV*, 2017.

Xiaolong Wang, Rohit Girdhar, and Abhinav Gupta. Binge watching: Scaling affordance learning from sitcoms. In *CVPR*, 2017.

Katsu Yamane, James J. Kuffner, and Jessica K. Hodgins. Synthesizing animations of human manipulation tasks. In *SIGGRAPH*, 2004.

Xinchen Yan, Akash Rastogi, Ruben Villegas, Kalyan Sunkavalli, Eli Shechtman, Sunil Hadap, Ersin Yumer, and Honglak Lee. MT-VAE: Learning motion transformations to generate multimodal human dynamics. In *ECCV*, 2018.

Taiping Yao, Minsi Wang, Bingbing Ni, Huawei Wei, and Xiaokang Yang. Multiple granularity group interaction prediction. In *CVPR*, 2018.

Wojciech Zaremba and Ilya Sutskever. Learning to execute. *arXiv preprint arXiv:1410.4615*, 2014.

Yi Zhou, Zimo Li, Shuangjiu Xiao, Chong He, Zeng Huang, and Hao Li. Auto-conditioned recurrent networks for extended complex human motion synthesis. In *ICLR*, 2018.

Yixin Zhu, Yibiao Zhao, and Song-Chun Zhu. Understanding tools: Task-oriented object modeling, learning and recognition. In *CVPR*, 2015.