This ICCV paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Stochastic Scene-Aware Motion Prediction

Mohamed Hassan[1] Duygu Ceylan[2] Ruben Villegas[2] Jun Saito[2] Jimei Yang[2] Yi Zhou[2] Michael Black[1]
[1]Max Planck Institute for Intelligent Systems, Tübingen, Germany [2]Adobe Research
{mhassan, black}@tue.mpg.de {ceylan, villegas, jsaito, jimyang, yizho}@adobe.com

Figure 1: SAMP synthesizes virtual humans navigating complex scenes with realistic and diverse human-scene interactions.

Abstract

A long-standing goal in computer vision is to capture, model, and realistically synthesize human behavior. Specifically, by learning from data, our goal is to enable virtual humans to navigate within cluttered indoor scenes and naturally interact with objects. Such embodied behavior has applications in virtual reality, computer games, and robotics, while synthesized behavior can be used as training data. The problem is challenging because real human motion is diverse and adapts to the scene. For example, a person can sit or lie on a sofa in many places and with varying styles. We must model this diversity to synthesize virtual humans that realistically perform human-scene interactions. We present a novel data-driven, stochastic motion synthesis method that models different styles of performing a given action with a target object. Our Scene-Aware Motion Prediction method (SAMP) generalizes to target objects of various geometries while enabling the character to navigate in cluttered scenes. To train SAMP, we collected MoCap data covering various sitting, lying down, walking, and running styles. We demonstrate SAMP on complex indoor scenes and achieve superior performance than existing solutions. Code and data are available for research at `https://samp.is.tue.mpg.de`.

1. Introduction

The computer vision community has made substantial progress on 3D scene understanding and on capturing 3D human motion, but less work has focused on synthesizing 3D people in 3D scenes. The advances in these two subfields, however, have provided tools for, and have created interest in, embodied agents for virtual worlds (e.g. [35, 42, 55, 56]) and in placing humans into scenes (e.g. [6, 21]). Creating virtual humans that move and act like real people, however, is challenging and requires tackling many smaller but difficult problems such as perception of unseen environments, plausible human motion modeling, and embodied interaction with complex scenes. While advances have been made in human locomotion modeling [23, 32] thanks to the availability of large scale datasets [7, 33, 38, 45, 50], realistically synthesizing virtual humans moving and interacting with 3D scenes, remains largely unsolved.

Imagine instructing a virtual human to "sit on a couch" in a cluttered scene, as illustrated in Fig. 1. To achieve this goal, the character needs to perform a series of complex actions. First, it should navigate through the scene to reach the target object while avoiding collisions with other objects in the scene. Next, the character needs to choose a *contact point* on the couch that will result in a plausible sitting action facing the right direction. Finally, if the character performs this action multiple times, there should be natural variations in the motion, mimicking real-world human-scene interactions; e.g., sitting on different parts of the couch with different styles such as with crossed legs, arms in different poses, etc. Achieving these goals requires a system to jointly reason about the scene geometry, smoothly transition between cyclic (e.g., walking) and acyclic (e.g., sitting) motions, and to model the diversity of human-scene interactions.

To this end, we propose SAMP for *Scene-Aware Mo-*

tion Prediction. SAMP is a stochastic model that takes a 3D scene as input, samples valid interaction goals, and generates goal-conditioned and scene-aware motion sequences of a character depicting realistic dynamic character-scene interactions. At the core of SAMP is a novel autoregressive conditional variational autoencoder (cVAE) called MotionNet. Given a target object and an action, MotionNet samples a random latent vector at each frame to condition the next pose both on the previous pose of the character as well as the random vector. This enables MotionNet to model a wide range of styles while performing the target action. Given the geometry of the target object, SAMP further uses another novel neural network called GoalNet to generate multiple plausible contact points and orientations on the target object (e.g., different positions and sitting orientations on the cushions of a sofa). This component enables SAMP to generalize across objects with diverse geometry. Finally, to ensure the character avoids obstacles while reaching the goal in a cluttered scene, we use an explicit path planning algorithm (A* search) to pre-compute an obstacle-free path between the starting location of the character and the goal. This piecewise linear path consists of multiple way-points, which SAMP treats as intermediate goals to drive the character around the scene. SAMP runs in real-time at 30 fps. To the best of our knowledge, these individual components make SAMP the first system that addresses the problem of generating diverse dynamic motion sequences that depict realistic human-scene interactions in cluttered environments.

Training SAMP requires a dataset of rich and diverse character scene interactions. Existing large-scale MoCap datasets are largely dominated by locomotion and the few interaction examples lack diversity. Additionally, traditional MoCap focuses on the body and rarely captures the scene. Hence, we capture a new dataset covering various human-scene interactions with multiple objects. In each motion sequence, we track both the body motion and the object using a high resolution optical marker MoCap system. The dataset is available for research purposes.

Our contributions are: (1) A novel stochastic model for synthesizing varied goal-driven character-scene interactions in real-time. (2) A new method for modeling plausible action-dependent goal locations and orientations of the body given the target object geometry. (3) Incorporating explicit path planning into a variational motion synthesis network enabling navigation in cluttered scenes. (4) A new MoCap dataset with diverse human-scene interactions.

2. Related Work

Interaction Synthesis: Analyzing and synthesizing plausible human-scene interactions have received a lot of attention from the computer vision and graphics communities. Various algorithms have been proposed for predicting object functionalities [16, 66], affordance analysis [18, 53], and synthesizing static human-scene interactions [16, 18, 21, 27, 41, 62, 64].

A less explored area involves generating dynamic human-scene interactions. While earlier work [28] focuses on synthesizing motions of a character in the same environment in which the motion was captured, follow up work [2, 26, 29, 43] assembles motion sequences from a large database to synthesize interactions with new environments or characters. Such methods, however, require large databases and expensive nearest neighbor matching.

An important sub-category of human-scene interaction involves locomotion, where the character must respond to changes in terrain with appropriate foot placement. Phase-functioned neural networks [23] have shown impressive results by using a guiding signal representing the state of the motion cycle (i.e., phase). Zhang et al. [61] extend this idea to use a mixture of experts [13, 25, 60] as the motion prediction network. An additional gating network is used to predict the expert blending weights at run time. More closely related to our work is the Neural State Machine (NSM) [47], which extends the ideas of phase labels and expert networks to model human-scene interactions such as sit, carry, and open. While NSM is a powerful method, it does not generate variations in such interactions, which is one of our key contributions. Our experiments also demonstrate that NSM often fails to avoid intersections between the 3D character and objects in cluttered scenes (Sec. 5.2). Furthermore, training NSM requires time-consuming manual, and often ambiguous, labeling of phases for non-periodic actions. Starke et al. [48] propose a method to automatically extract local phase variables for each body part in the context of a two-player basketball game. Extending local phases to non-periodic actions is not trivial, however. We find that using scheduled sampling [5] provides an alternative to generate smooth transitions without phase labels. More recently, Wang et al. [52] introduce a hierarchical framework for synthesizing human-scene interactions. They generate sub-goal positions in the scene, predict the pose at each of these sub-goals, and synthesize the motion between such poses. This method requires a post-optimization framework to ensure smoothness and robust foot contact and to discourage penetration with the scene. Corona et al. [11] use a semantic graph to model human-object relationships followed by an RNN to predict human and object movements.

An alternative approach uses reinforcement learning (RL) to build a control policy that models interactions. Merel et al. [37] and Eom et al. [14] focus on ball catching from egocentric vision. Chao et al. [10] train sub-task controllers and a meta controller to execute the sub-tasks to complete a sitting task. However, in contrast to SAMP, their approach does not enable variations in the goal positions and directions. In addition, as with many RL-based approaches, generalizing the learned policies to new environments or actions is often challenging.

Motion Synthesis: Neural networks (feed-forward networks, LSTMs, or RNNs) have been extensively applied to the motion synthesis problem [1, 15, 19, 24, 36, 49, 51]. A typical approach predicts the future motion of a character based on previous frame(s). While showing impressive results when generating short sequences, many of these methods either converge to the mean pose or diverge when tested on long sequences. A common solution is to employ scheduled sampling [5] to ensure stable predictions at test time to generate long locomotion and dancing sequences [32, 65].

Several works have focused on modeling the stochastic nature of human motion, with a specific emphasis on trajectory prediction. Given the past trajectory of a character, they model multiple plausible future trajectories [4, 6, 8, 17, 34, 40, 44]. Recently, Cao et al. [6] sample multiple future goals and then use them to generate different future skeletal motions. This is similar in spirit to our use of GoalNet. The difference is that our goal is to predict various trajectories that always lead to the same target object (instead of predicting any plausible future trajectory).

Modeling the stochasticity of the full human motion is a less explored area [54, 58, 59]. Motion VAE [32] predicts a distribution of the next poses instead of one pose using the latent space of a conditional variational autoencoder. MoGlow is a controllable probabilistic generative model based on normalizing flows [22]. Generating diverse dance motions from music has also been recently explored [30, 31]. Xu et al. [57] generate diverse motions by blending short sequences from a database. To the best of our knowledge, no previous work has tackled the problem of generating diverse human-scene interactions.

3. Method

Generating dynamic human scene interactions in cluttered environments requires solutions to several subproblems. First and foremost, the synthesized motion of the character should be realistic and capture natural variations. Given a target object, it is important to sample plausible contact points and orientations for performing a specific action (e.g., where to sit on a chair and which direction to face). Finally, the motion needs to be synthesized such that it navigates to the goal location while avoiding penetrating objects in the scene. Our system consists of three main components that address each of these sub-problems: a *MotionNet*, *GoalNet*, and a *Path Planning Module*. At the core of our method is the MotionNet which predicts the pose of the character based on the previous pose as well as other factors such as the interaction object geometry and the target goal position and orientation. GoalNet predicts the goal position and orientation for the interaction on the desired object. The Path Planning Module computes an obstacle-free path between the starting location of the character and the goal location. The full pipeline is illustrated in Fig. 2.

3.1. MotionNet

MotionNet is an autoregressive conditional variational autoencoder (cVAE) [12, 46] that generates the pose of the character conditioned on its previous state (e.g., pose, trajectory, goal) as well as the geometry of the interaction object. MotionNet has two components: an encoder and a decoder. The encoder encodes the previous and current states of the character and the interaction object to a latent vector $\boldsymbol{Z}$. The decoder takes this latent vector, the character's previous state, and the interaction object to predict the character's next state. The pipeline is shown in Fig. 3. Note that, at test time, we only utilize the decoder of MotionNet and sample $\boldsymbol{Z}$ from a standard normal distribution.

Encoder: The encoder consists of two sub-encoders: *State Encoder* and *Interaction Encoder*. The State Encoder encodes the previous and current state of the character into a low-dimensional vector. Similarly, the Interaction Encoder encodes the object geometry into a different low-dimensional vector. Next, the two vectors are concatenated and passed through two identical fully connected layers to predict the mean μ and standard deviation σ of a Gaussian distribution representing a latent embedding space. We then sample a random latent code $\boldsymbol{Z}$, which is provided to the decoder when predicting the next state of the character.

State Representation: We use a representation similar to Starke et al. [47] to encode the state of the character. Specifically, the state at frame i is defined as $\boldsymbol{X}_i =$

$$\left\{\boldsymbol{j}_i^p, \boldsymbol{j}_i^r, \boldsymbol{j}_i^v, \tilde{\boldsymbol{j}}_i^p, \boldsymbol{t}_i^p, \boldsymbol{t}_i^d, \tilde{\boldsymbol{t}}_i^p, \tilde{\boldsymbol{t}}_i^d, \boldsymbol{t}_i^a, \boldsymbol{g}_i^p, \boldsymbol{g}_i^d, \boldsymbol{g}_i^a, \boldsymbol{c}_i\right\}, \quad (1)$$

where $\boldsymbol{j}_i^p \in \mathbb{R}^{3j}, \boldsymbol{j}_i^r \in \mathbb{R}^{6j}, \boldsymbol{j}_i^v \in \mathbb{R}^{3j}$ are the position, rotation, and velocity of each joint relative to the root. j is the number of joints in the skeleton which is 22 in our data. $\tilde{\boldsymbol{j}}_i^p \in \mathbb{R}^{3j}$ are the joint positions relative to future root 1 second ahead. $\boldsymbol{t}_i^p \in \mathbb{R}^{2t}$ and $\boldsymbol{t}_i^d \in \mathbb{R}^{2t}$ are the root positions and forward directions relative to the root of frame $i-1$. $\tilde{\boldsymbol{t}}_i^p \in \mathbb{R}^{2t}$ and $\tilde{\boldsymbol{t}}_i^d \in \mathbb{R}^{2t}$ are the root positions and forward directions relative to the goal of frame $i-1$. We define these inputs for t time steps sampled uniformly in a 2 second window between $[-1, 1]$ seconds. $\boldsymbol{t}_i^a \in \mathbb{R}^{n_a t}$ is a vector of continuous action labels on each of the t samples. In our experiments, n_a is 5, which is the total number of actions we model (i.e., idle, walk, run, sit, lie down). $\boldsymbol{g}_i^p \in \mathbb{R}^{3t}, \boldsymbol{g}_i^d \in \mathbb{R}^{3t}$ are the goal positions and directions, and $\boldsymbol{g}_i^a \in \mathbb{R}^{n_a t}$ is a one-hot action label describing the action to be performed at each of the t samples. $\boldsymbol{c}_i \in \mathbb{R}^5$ are contact labels for pelvis, feet, and hands.

State Encoder: The State Encoder takes the current $\boldsymbol{X}_i$ and previous state $\boldsymbol{X}_{i-1}$ and encodes them into a low-dimensional vector using three fully connected layers.

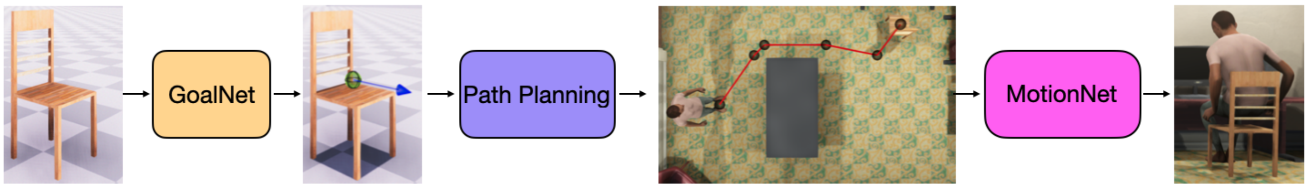

Figure 2: Our system consists of three main components. GoalNet predicts oriented goal locations (green sphere and blue arrow on the chair) given an interaction object. The *Path Planning Module* predicts an obstacle-free path from the starting position to the goal. *MotionNet* sequentially predicts the next character state until the desired action is executed.

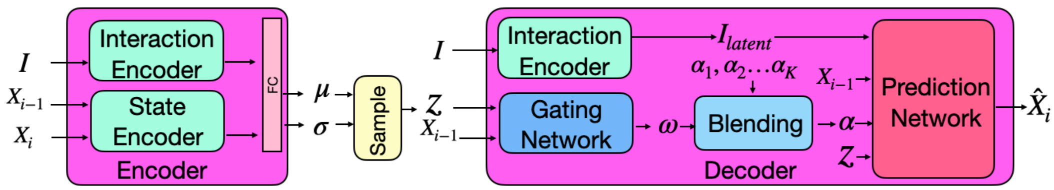

Figure 3: MotionNet consists of an encoder and a decoder. The encoder consists of two sub-encoders: State Encoder and Interaction Encoder. The decoder consists of a Prediction Network to predict the next character state and a gating network that predicts the blending weights of the Prediction Network. See Sec. 3.1.

Interaction Encoder: The Interaction Encoder takes a voxel representation of the interaction object $\boldsymbol{I}$ and encodes it into a low-dimensional vector. We use a voxel grid of size $8 \times 8 \times 8$. Each voxel stores a $4-$dimensional vector. The first three components refer to the position of the voxel center relative to the root of the character. The fourth element stores the real-valued occupancy (between 0 and 1) of the voxel. The architecture consists of three fully connected layers.

Decoder: The decoder takes the random latent code $\boldsymbol{Z}$, the interaction object representation $\boldsymbol{I}$, and the previous state $\boldsymbol{X}_{i-1}$, and predicts the next state $\hat{\boldsymbol{X}}_i$. Similar to recent work [32, 47], our decoder is built as a mixture-of-experts with two components: the Prediction Network and Gating Network.

The Prediction Network is responsible for predicting the next state $\hat{\boldsymbol{X}}_i$. The weights of the Prediction Network $\boldsymbol{\alpha}$ are computed by blending K expert weights:

$$\boldsymbol{\alpha} = \sum_{i=1}^{K} \omega_i \boldsymbol{\alpha}_i, \tag{2}$$

where the blending weights ω_i are predicted by the Gating Network. Each expert is a three-layer fully connected network. The Gating Network is also a three-layer fully connected network, which takes as input $\boldsymbol{Z}$ and $\boldsymbol{X}_{i-1}$.

MotionNet is trained end-to-end to minimize the loss $\mathcal{L}_{\text{motion}} =$

$$||\hat{\boldsymbol{X}}_i - \boldsymbol{X}_i||_2^2 + \beta_1 KL(Q(\boldsymbol{Z}|\boldsymbol{X}_i, \boldsymbol{X}_{i-1}, \boldsymbol{I})||p(\boldsymbol{Z})), \tag{3}$$

where the first term minimizes the difference between the ground truth and predicted states of the character and KL denotes the Kullback-Leibler divergence.

3.2. GoalNet

Given a target interaction object (which can be interactively defined by a user at test time or randomly sampled among the objects in the scene), the character is driven by the goal position $\boldsymbol{g}^p \in \mathbb{R}^3$ and direction $\boldsymbol{g}^d \in \mathbb{R}^3$ sampled on the object's surface. In order to perform realistic interactions; the character requires the ability to predict these goal positions and directions from the object geometry. For example, while a regular chair allows variation in terms of sitting direction, the direction of sitting on an armchair is restricted (see Fig. 7). We use GoalNet to model object-specific goal positions and directions. GoalNet is a conditional variational autoencoder (cVAE) that predicts plausible goal positions and directions given the voxel representation of the target interaction object $\boldsymbol{I}$ as shown in Fig. 4. The encoder encodes the interaction object $\boldsymbol{I}$, goal position $\boldsymbol{g}^p$, and direction $\boldsymbol{g}^d$, into a latent code $\boldsymbol{Z}_{goal}$. The decoder reconstructs the goal position $\hat{\boldsymbol{g}}^p$, and direction $\hat{\boldsymbol{g}}^d$ from $\boldsymbol{Z}_{goal}$ and $\boldsymbol{I}$. We represent the object using a voxel representation similar to the one used in MotionNet (Sec. 3.1). The only difference is that we compute the voxel position relative to the object center instead of the character root. In the encoder, we use an Interaction Encoder similar to the one used in MotionNet (see Sec. 3.1) to encode the object representation $\boldsymbol{I}$ to a low dimension vector. This vector is then concatenated with $\boldsymbol{g}^p$ and $\boldsymbol{g}^d$ and encoded further to the latent vector $\boldsymbol{Z}_{goal}$. The decoder has the same architecture as the encoder as shown in Fig. 4. The network is trained to minimize the loss:

$$\begin{aligned}\mathcal{L}_{\text{goal}} =& ||\hat{\boldsymbol{g}}^p - \boldsymbol{g}^p||_2^2 + ||\hat{\boldsymbol{g}}^d - \boldsymbol{g}^d||_2^2 \\ &+ \beta_2 KL(Q(\boldsymbol{Z}_{goal}|\boldsymbol{g}^p, \boldsymbol{g}^d, \boldsymbol{I})||p(\boldsymbol{Z}_{goal})).\end{aligned} \tag{4}$$

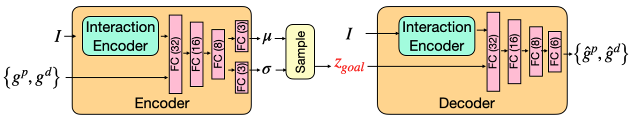

Figure 4: GoalNet generates multiple valid goal positions $\hat{\boldsymbol{g}}^p$ and directions $\hat{\boldsymbol{g}}^d$ given an object representation $\boldsymbol{I}$. FC(N) denotes a fully connected layer of size N.

At test time, given a target object $\boldsymbol{I}$, we randomly sample $\boldsymbol{Z}_{goal} \sim \mathcal{N}(0, I)$ and use the decoder to generate various goal positions $\boldsymbol{g}^p$ and directions $\boldsymbol{g}^d$.

3.3. Path Planning

To ensure the character can navigate inside cluttered environments while avoiding obstacles, we employ an explicit A* path planning algorithm [20]. Given the desired goal location, we use A* to compute an obstacle-free path from the starting position of the character to the goal. The path is defined as a series of waypoints $w_i = \{w_0, w_1, w_2, ...\}$ that define the locations where the path changes direction. We break the task of performing the final desired action into sub-tasks in which each sub-task requires the character to walk to the next waypoint. The final sub-task requires the character to perform the desired action at the final waypoint.

3.4. Training Strategy

Training MotionNet using standard supervised training produces poor quality predictions at run time (see Sup. Mat.). This is due to the accumulation of error at run time when the output of the network is fed back as input in the next step. To account for this, we train the network using *scheduled sampling* [5], which has been shown to result in long stable motion predictions [32]. During training, the current network prediction is used as input in the next training step with a probability $1 - P$. P is (see Sup. Mat.):

$$P = \begin{cases} 1 & epoch \leq C_1, \\ 1 - \frac{epoch - C_1}{C_2 - C_1} & C1 < epoch \leq C_2, \\ 0 & epoch > C2. \end{cases} \quad (5)$$

4. Data Preparation

4.1. Motion Data

To model variations in human-scene interactions, we capture a new dataset using an optical MoCap system with 54 Vicon cameras. We place seven different objects in the center of the MoCap area, namely two sofas, an armchair, a chair, a high bar chair, a low chair and a table. We record multiple clips of each interaction with different styles. In each sequence, the subject starts from an A-Pose in a random location in the MoCap space, walks towards the object, and performs the action for $20 - 40$ seconds. Finally, the subject gets up from the object and walks away. Our goal is to capture various styles of performing the same action, thus we ask the subject to change the style in each sequence. In addition to the subject, we also capture the object pose using attached markers. We also have the CAD model for each object. Finally, we capture running, walking, and idle sequences where the subject walks and runs in different directions with different speeds and stands in an idle state. Our dataset consists of $\sim$100 minutes of motion data recorded at 30 fps from a single subject, resulting in $\sim$185K frames. We use MoSh++ [33] to fit the SMPL-X [39] body model to the optical markers. More details about the data are available in the Sup. Mat.

4.2. Motion Data Augmentation

With only seven captured objects, MotionNet will fail to adapt to new unseen objects. Capturing MoCap with a wide range of objects requires a significant amount of effort and time. We address this issue by augmenting our data using an efficient augmentation pipeline similar to [3, 47]. Since we capture both the body motion as well as the object pose, we compute the contact between the body and the object. We detect the contacts of five key joints of the character skeleton. Namely, pelvis, hands, and feet. We then augment our data by randomly switching or scaling the object at each frame. When switching, we replace the original object with a random object of a similar size selected from ShapeNet [9]. For each new object (scaled or switched), we project the contacts detected from the ground truth data to the new object. Finally, we use an IK solver to recompute the full pose such that the contacts are maintained. Please refer to the Sup. Mat. for more details.

4.3. Goal Data

To train GoalNet, we label various goal positions $\boldsymbol{g}^p$ and directions $\boldsymbol{g}^d$ for different objects from ShapeNet [9]. These goals represent the position on the object surface where a character could sit and the forward direction of the character when sitting. We select 5 categories from ShapeNet namely, sofas, L-shaped sofas, chairs, armchairs, and tables. From each category, we select $15 - 20$ instances and we manually label $1 - 5$ goals for each instance. The number of goals labeled per instance depends on how many different goals an object can afford. For example, an L-shaped sofa offers more places to sit than a chair. In total, we use 80 objects as our training data. We augment our data by randomly scaling the objects across the xyz axes leading to $\sim$13K training samples.

5. Experiments & Evaluation

5.1. Qualitative Evaluation

In this section, we provide qualitative results and discuss the main points. We refer to the Sup. Mat. and the accompanying video for more results.

Generating Diverse Motion: In contrast to previous deterministic methods [47], SAMP generates a wide range of diverse styles of an action while ensuring realism. Several different sitting and lying down styles generated by SAMP are shown in Fig. 5. The use of the Interaction Encoder 3.1 and the data augmentation (Sec. 4.2) further ensures SAMP can adapt to different objects with varying geometry. Notice how the character naturally leans its head back on the sofa. The style of the action is also conditioned on the interacting object. The character lifts its legs when sitting on a high chair/table but extends its legs when sitting on a very low table. We observe that lying down is a harder task and several of baseline methods fail to execute this task (see Sec. 5.2). While SAMP synthesizes reasonable sequences, our results are not always perfect. The generated motion might involve some penetration with the object.

Goal Generation: When presented with a new object, the character needs to predict where and in which direction the action should be executed. In [47], the goal is computed as the object center. However, this heuristic fails for objects with complex geometries. In Fig. 6 we show that using the object center results in invalid actions whereas GoalNet allows our method to reason about where the action should be executed. As shown in Fig. 7, by sampling different latent codes $\boldsymbol{Z}_{goal}$, GoalNet generates multiple goal positions and directions for various objects. Notice how GoalNet captures that, while a person can sit sideways on a regular chair, this is not valid for an armchair.

Figure 8 shows how the different goals generated by GoalNet guide the motion of the character. Starting from the same position, direction, and initial pose, the virtual human follows two different paths to reach different goal positions when performing the "sit on the couch" action. The final pose of the character is also different in the two cases due to the stochastic nature of MotionNet.

Path Planning: When navigating to a particular goal location in a cluttered scene, it is critical to avoid obstacles. Our Path Planning Module achieves this goal by predicting the shortest obstacle-free path between the starting character position and the goal using a navigation mesh computed based on the 3D scene. The navigation mesh defines the walk-able areas in the scene and is computed once offline. In Fig. 9, we show an example path computed by the Path Planning Module. Without this module, the character often walks through objects in the scene. We observe a similar behaviour in the previous work of NSM [47], even though NSM uses a volumetric representation of the environment to help the character navigate.

	Walk	Run	Sit	Liedown
GT	5.95	7.74	5.18	7.52
SAMP	5.63	5.75	5.05	6.69

Table 1: Diversity metric. Higher values indicate more diversity.

5.2. Quantitative Evaluation

Deterministic vs. Stochastic: To quantify the diversity of the generated motion, we put the character in a fixed starting position and direction and we run our method ten times with the same goal. For example, we instruct the character to sit/lie down on the same object multiple times starting from the same initial state/position/direction. For walking and running, we instruct the character to run in each of the four directions for 15 seconds. We record the character motion for each run and then compute the Average Pairwise Distance (APD) [58, 63] as shown in Table. 1. The APD is defined as:

$$APD = \frac{1}{N(N-1)} \sum_{i=0}^{N} \sum_{\substack{j=0 \\ j\neq i}}^{N} ||\boldsymbol{X}_i' - \boldsymbol{X}_j'||_2^2. \tag{6}$$

$\boldsymbol{X}_i'$ represents the character's local pose features at frame i. $\boldsymbol{X}_i' = \{\boldsymbol{j}_i^p, \boldsymbol{j}_i^r, \boldsymbol{j}_i^v\}$. N is the total number of frames for all sequences. For comparison we also report the APD for the ground truth (GT) data in Table. 1.

GoalNet: Given 150 unseen goals sampled on test objects, we measure the average position and orientation reconstruction error of GoalNet to be $\mathbf{6.04}$ cm and $\mathbf{2.29}$ deg (we note that the objects have real-life measurements). To measure the diversity of the generated goals, we compute the Average Pairwise Distance (APD) among the generated goal positions g^p and directions g^d:

$$\text{APD-Pos} = \frac{1}{LN(N-1)} \sum_{k=0}^{L} \sum_{i=0}^{N} \sum_{\substack{j=0 \\ j\neq i}}^{N} |g_i^p - g_j^p| \tag{7}$$

$$\text{APD-Rot} = \frac{1}{LN(N-1)} \sum_{k=0}^{L} \sum_{i=0}^{N} \sum_{\substack{j=0 \\ j\neq i}}^{N} \arccos(g_i^d . g_j^d). \tag{8}$$

$L = 150$ is the number of objects and $N = 10$ is the number of goals generated for each object. We find APD-Pos and APD-Rot for our generated goals to be $\mathbf{16.42}$ cm and $\mathbf{41.27}$ deg compared to $\mathbf{16.18}$ cm and $\mathbf{90.23}$ deg for the ground truth (GT) data.

Path Planning Module: To quantitatively evaluate the effectiveness of our Path Planning Module, we test our method in a cluttered scene. We put the character in a random initial position and orientation and select a random

Figure 5: SAMP generates plausible and diverse action styles and adapts to different object geometries.

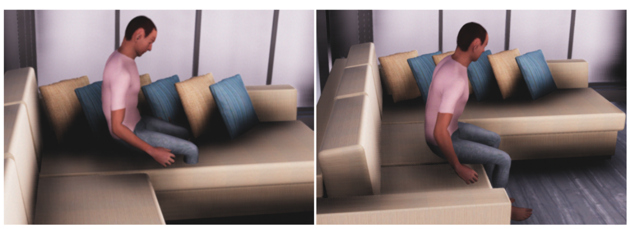

Figure 6: Without GoalNet (left), SAMP fails to sit on a valid place. SAMP with GoalNet is shown on the right.

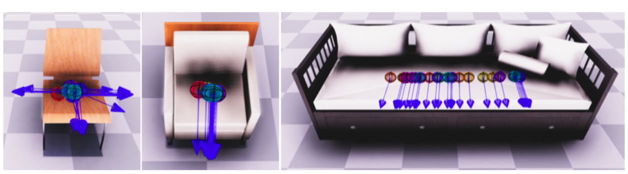

Figure 7: GoalNet generates diverse valid goals on different objects. Spheres indicate goal positions, and blue arrows indicate goal directions.

goal. We repeat this 10 times. We find the percentage of frames where a penetration happens is $\mathbf{3.8}\%$, $\mathbf{11.2}\%$, and $\mathbf{8.11}\%$ for SAMP with Path Planning Module, without Path Planning Module, and NSM [47], respectively. While NSM uses a volumetric sensor to detect collisions with the environment, it is not as effective as explicit path planning.

Comparison to Previous Models: We compare our model to baselines by measuring three metrics: average execution time, average precision, and Frèchet distance (FD) between the distribution of the generated motion and ground truth. Execution time is the time required to transition to the target action label from an idle state. Precision is the positional (PE) and rotational (RE) error at the goal. We measure FD on a subset of the state features which we call $\tilde{\boldsymbol{X}}$:

$$\tilde{\boldsymbol{X}} = \{\boldsymbol{j}^p, \boldsymbol{j}^r, \boldsymbol{j}^v, \tilde{\boldsymbol{t}}^p, \tilde{\boldsymbol{t}}^d\}. \quad (9)$$

As our baselines, we choose a feedforward network (MLP) as the motion prediction network, Mixture of Experts (MoE) [61], and NSM [47] (see Sup. Mat. for details).

	MLP	MoE	SAMP	GT
Sit	13.06	12.99	**12.53**	11.7
Liedown	∞	∞	**17.06**	15.49

Table 2: Average execution Time in seconds. ∞ means the method failed to reach the goal within 3 minutes.

SAMP vs. MLP vs. MoE: We re-trained the MLP and MoE using the same training strategy and data we used for SAMP. Both MLP and MoE take a longer time to execute the task and often fail to execute the "lie down" action (denoted ∞) as evidenced by the execution time in Table. 2 and precision in Table. 3. These architectures sometimes generate implausible poses as shown in Sup. Mat., which is reflected by the lower FD in Table. 4

SAMP vs. NSM: For NSM, we used the publicly available pre-trained model since retraining NSM on our data is infeasible due to the missing phase labels. We trained SAMP on the same data on which NSM was trained. In

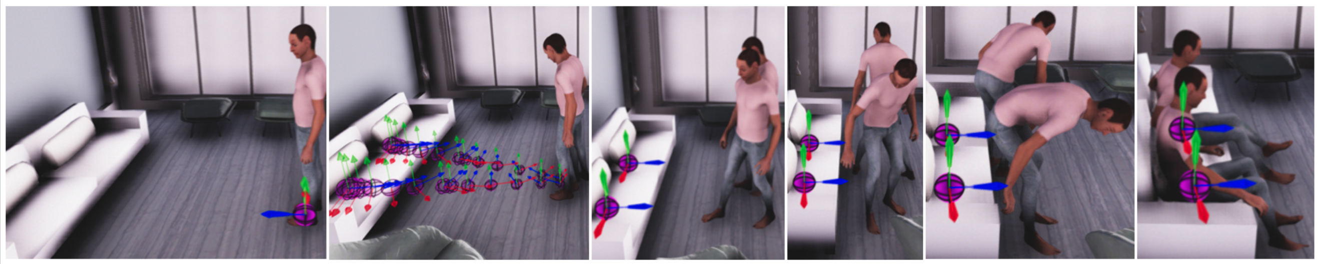

Figure 8: Goals generated by GoalNet (mesh spheres) are used by MotionNet to guide the motion of virtual characters.

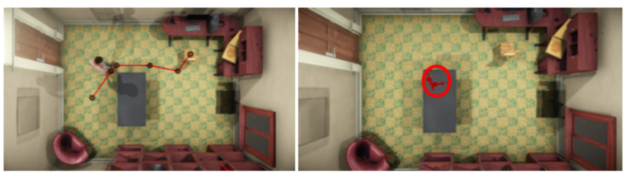

Figure 9: Our Path Planning Module helps SAMP to successfully navigate cluttered scenes (left). NSM [47] fails in such scenes (right).

Method	Sit		Liedown	
	PE(cm)	RE(deg)	PE(cm)	RE(deg)
MLP	9.27	3.99	∞	∞
MoE	7.99	5.73	∞	∞
SAMP	**6.09**	**3.55**	**5.76**	**6.45**

Table 3: Average precision in terms of positional and rotational errors (PE and RE). ∞ means the method failed to reach the goal within 3 minutes.

	Idle	Walk	Run	Sit	Liedown
MLP	102.85	121.18	150.56	105.87	36.85
MoE	102.91	114.17	151.14	105.10	35.79
SAMP	**102.72**	**111.09**	**141.11**	**104.68**	**17.30**

Table 4: Frèchet distance.

Table 5 we observe that our model is on par with NSM in terms of achieving goals without the need for phase labels, which are cumbersome and often ambiguous to annotate. In addition, our main focus is to model diverse motions via a stochastic model while NSM is deterministic. Our Path Planning Module module helps SAMP to safely navigate complex scenes where NSM fails as shown by the penetration amounts.

For all evaluations, all test objects are randomly selected from ShapeNet and none is part of our training set.

Limitations and Future Work: We observe that sometimes slight penetrations between the character and the interacting object can occur. A potential solution is to incorporate a post-processing step to optimize the pose of the character to avoid such intersections. In order to generalize SAMP to interacting objects that have significantly different geometry than those seen in training, in future work, we would like to explore methods to encode local object geometries.

Metric	Sit		Carry	
	SAMP	NSM	SAMP	NSM
Precision PE (cm) $\downarrow$	**15.97**	16.95	**4.58**	4.72
Precision RE (deg) $\downarrow$	5.38	**2.32**	1.78	**1.65**
Execution Time (sec) $\downarrow$	12.93	**10.26**	13.29	**12.82**
FD $\downarrow$	6.20	**4.21**	10.17	**7.31**
Diversity $\uparrow$	**0.44**	0.0	**0.26**	0.0
Penetration (%) $\downarrow$	**3.8**	8.11	**3.62**	8.45

Table 5: SAMP vs. NSM.

6. Conclusion

Here we have described SAMP, which makes several important steps toward creating lifelike avatars that move and act like real people in previously unseen and complex environments. Critically, we introduce three elements that must be part of a solution. First, characters must be able to navigate the world and avoid obstacles. For this, we use an existing path planning method. Second, characters can interact with objects in different ways. To address this, we train GoalNet to take an object and stochastically produce an interaction location and direction. Third, the character should produce motions achieving the goal that vary naturally. To that end, we train a novel MotionNet that incrementally generates body poses based on the past motion and the goal. We train SAMP using a novel dataset of motion capture data involving human-object interaction.

Acknowledgement This work was initiated while MH was an intern at Adobe. We are grateful to Sebastian Starke for inspiring work, helpful discussion, and making his code open-source. We thank Joachim Tesch for feedback on Unity and rendering, Nima Ghorbani for MoSH++, and Meshcapade for the character texture. For helping with the data collection, we are grateful to Tsvetelina Alexiadis, Galina Henz, Markus Höschle and Tobias Bauch.

Disclosure: MJB has received research funds from Adobe, Intel, Nvidia, Facebook, and Amazon. While MJB is a part-time employee of Amazon, his research was performed solely at, and funded solely by, Max Planck. MJB has financial interests in Amazon, Datagen Technologies, and Meshcapade GmbH.

References

[1] Vida Adeli, Mahsa Ehsanpour, Ian Reid, Juan Carlos Niebles, Silvio Savarese, Ehsan Adeli, and Hamid Rezatofighi. Tripod: Human trajectory and pose dynamics forecasting in the wild. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2021. 3

[2] Shailen Agrawal and Michiel van de Panne. Task-based locomotion. *ACM Trans. Graph.*, 35(4), 2016. 2

[3] Rami Ali Al-Asqhar, Taku Komura, and Myung Geol Choi. Relationship descriptors for interactive motion adaptation. In *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '13, page 45–53, New York, NY, USA, 2013. Association for Computing Machinery. 5

[4] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social lstm: Human trajectory prediction in crowded spaces. In *Proceedings IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 961–971, 2016. 3

[5] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, page 1171–1179, Cambridge, MA, USA, 2015. MIT Press. 2, 3, 5

[6] Zhe Cao, Hang Gao, Karttikeya Mangalam, Qi-Zhi Cai, Minh Vo, and Jitendra Malik. Long-term human motion prediction with scene context. In *European Conference on Computer Vision (ECCV)*, pages 387–404. Springer, 2020. 1, 3

[7] Carnegie Mellon University. CMU MoCap Dataset. 1

[8] Yuning Chai, Benjamin Sapp, Mayank Bansal, and Dragomir Anguelov. Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction. In *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pages 86–99. PMLR, 30 Oct–01 Nov 2020. 3

[9] 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. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015. 5

[10] Yu-Wei Chao, Jimei Yang, Weifeng Chen, and Jia Deng. Learning to sit: Synthesizing human-chair interactions via hierarchical control. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2019. 2

[11] Enric Corona, Albert Pumarola, Guillem Alenya, and Francesc Moreno-Noguer. Context-aware human motion prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6992–7001, 2020. 2

[12] P Kingma Diederik and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations ICLR*, 2014. 3

[13] David Eigen, Marc'Aurelio Ranzato, and Ilya Sutskever. Learning factored representations in a deep mixture of experts. In *International Conference on Learning Representations ICLR*, 2014. 2

[14] Haegwang Eom, Daseong Han, Joseph S. Shin, and Junyong Noh. Model predictive control with a visuomotor system for physics-based character animation. *ACM Trans. Graph.*, 39(1), Oct. 2019. 2

[15] Katerina Fragkiadaki, Sergey Levine, Panna Felsen, and Jitendra Malik. Recurrent network models for human dynamics. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, ICCV '15, page 4346–4354, USA, 2015. IEEE Computer Society. 3

[16] Helmut Grabner, Juergen Gall, and Luc Van Gool. What makes a chair a chair? In *Proceedings IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1529–1536, 2011. 2

[17] Agrim Gupta, Justin Johnson, Li Fei-Fei, Silvio Savarese, and Alexandre Alahi. Social gan: Socially acceptable trajectories with generative adversarial networks. In *Proceedings IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2255–2264, 2018. 3

[18] A. Gupta, S. Satkin, A. A. Efros, and M. Hebert. From 3D scene geometry to human workspace. In *Proceedings IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1961–1968, 2011. 2

[19] I. Habibie, Daniel Holden, Jonathan Schwarz, Joe Yearsley, and T. Komura. A recurrent variational autoencoder for human motion synthesis. In *BMVC*, 2017. 3

[20] P. E. Hart, N. J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968. 5

[21] Mohamed Hassan, Partha Ghosh, Joachim Tesch, Dimitrios Tzionas, and Michael J. Black. Populating 3D scenes by learning human-scene interaction. In *Proceedings IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2021. 1, 2

[22] Gustav Eje Henter, Simon Alexanderson, and Jonas Beskow. MoGlow: Probabilistic and controllable motion synthesis using normalising flows. *ACM Trans. Graph.*, 39(6), Nov. 2020. 3

[23] Daniel Holden, Taku Komura, and Jun Saito. Phase-functioned neural networks for character control. *ACM Trans. Graph.*, 36(4), July 2017. 1, 2

[24] Daniel Holden, Jun Saito, and Taku Komura. A deep learning framework for character motion synthesis and editing. *ACM Trans. Graph.*, 35(4):1–11, 2016. 3

[25] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87, 1991. 2

[26] Mubbasir Kapadia, Xu Xianghao, Maurizio Nitti, Marcelo Kallmann, Stelian Coros, Robert W. Sumner, and Markus Gross. Precision: Precomputing environment semantics for contact-rich character animation. In *Proceedings of the 20th ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, I3D '16, page 29–37, New York, NY, USA, 2016. Association for Computing Machinery. 2

[27] Vladimir G. Kim, Siddhartha Chaudhuri, Leonidas Guibas, and Thomas Funkhouser. Shape2Pose: Human-centric shape analysis. *ACM Trans. Graph.*, Aug. 2014. 2

[28] Jehee Lee, Jinxiang Chai, Paul S. A. Reitsma, Jessica K. Hodgins, and Nancy S. Pollard. Interactive control of avatars animated with human motion data. *ACM Trans. Graph.*, 21(3):491–500, July 2002. 2

[29] Kang Hoon Lee, Myung Geol Choi, and Jehee Lee. Motion patches: Building blocks for virtual environments annotated with motion data. In *ACM SIGGRAPH 2006 Papers*, SIGGRAPH '06, page 898–906, New York, NY, USA, 2006. Association for Computing Machinery. 2

[30] Jiaman Li, Yihang Yin, Hang Chu, Yi Zhou, Tingwu Wang, Sanja Fidler, and Hao Li. Learning to generate diverse dance motions with transformer. *arXiv preprint arXiv:2008.08171*, 2020. 3

[31] Ruilong Li, Shan Yang, David A Ross, and Angjoo Kanazawa. Learn to dance with AIST++: Music conditioned 3D dance generation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2021. 3

[32] Hung Yu Ling, Fabio Zinno, George Cheng, and Michiel Van De Panne. Character controllers using motion vaes. *ACM Trans. Graph.*, 39(4), July 2020. 1, 3, 4, 5

[33] 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 International Conference on Computer Vision (ICCV)*, pages 5442–5451, Oct. 2019. 1, 5

[34] O. Makansi, E. Ilg, Ö. Çiçek, and T. Brox. Overcoming limitations of mixture density networks: A sampling and fitting framework for multimodal future prediction. In *Proceedings IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019. 3

[35] Manolis Savva*, Abhishek Kadian*, Oleksandr Maksymets*, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A Platform for Embodied AI Research. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019. 1

[36] Julieta Martinez, Michael J Black, and Javier Romero. On human motion prediction using recurrent neural networks. In *Proceedings IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2891–2900, 2017. 3

[37] Josh Merel, Saran Tunyasuvunakool, Arun Ahuja, Yuval Tassa, Leonard Hasenclever, Vu Pham, Tom Erez, Greg Wayne, and Nicolas Heess. Catch & carry: Reusable neural controllers for vision-guided whole-body tasks. *ACM Trans. Graph.*, 39(4), July 2020. 2

[38] M. Müller, T. Röder, M. Clausen, B. Eberhardt, B. Krüger, and A. Weber. Documentation mocap database HDM05. Technical Report CG-2007-2, Universität Bonn, June 2007. 1

[39] Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed A. A. Osman, Dimitrios Tzionas, and Michael J. Black. Expressive body capture: 3D hands, face, and body from a single image. In *Proceedings IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019. 5

[40] Amir Sadeghian, Vineet Kosaraju, Ali Sadeghian, Noriaki Hirose, Hamid Rezatofighi, and Silvio Savarese. Sophie: An attentive gan for predicting paths compliant to social and physical constraints. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1349–1358, 2019. 3

[41] Manolis Savva, Angel X Chang, Pat Hanrahan, Matthew Fisher, and Matthias Nießner. PiGraphs: Learning interaction snapshots from observations. *ACM Trans. Graph.*, 35(4):1–12, 2016. 2

[42] Bokui Shen, Fei Xia, Chengshu Li, Roberto Martín-Martín, Linxi Fan, Guanzhi Wang, Claudia Pérez-D'Arpino, Shyamal Buch, Sanjana Srivastava, Lyne P. Tchapmi, Micael E. Tchapmi, Kent Vainio, Josiah Wong, Li Fei-Fei, and Silvio Savarese. iGibson 1.0: a simulation environment for interactive tasks in large realistic scenes. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems*, page accepted. IEEE, 2021. 1

[43] Hubert P. H. Shum, Taku Komura, Masashi Shiraishi, and Shuntaro Yamazaki. Interaction patches for multi-character animation. *ACM Trans. Graph.*, 27(5), Dec. 2008. 2

[44] H. Sidenbladh, M. J. Black, and L. Sigal. Implicit probabilistic models of human motion for synthesis and tracking. In *European Conf. on Computer Vision (ECCV)*, volume 1, pages 784–800, 2002. 3

[45] L. Sigal, A. Balan, and M. J. Black. HumanEva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion. *International Journal of Computer Vision*, 87(4):4–27, Mar. 2010. 1

[46] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems 28*, pages 3483–3491. Curran Associates, Inc., 2015. 3

[47] Sebastian Starke, He Zhang, Taku Komura, and Jun Saito. Neural state machine for character-scene interactions. *ACM Trans. Graph.*, 38(6), Nov. 2019. 2, 3, 4, 5, 6, 7, 8

[48] Sebastian Starke, Yiwei Zhao, Taku Komura, and Kazi Zaman. Local motion phases for learning multi-contact character movements. *ACM Trans. Graph.*, 39(4), July 2020. 2

[49] Graham W. Taylor and Geoffrey E. Hinton. Factored conditional restricted boltzmann machines for modeling motion style. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, page 1025–1032, New York, NY, USA, 2009. Association for Computing Machinery. 3

[50] Matthew Trumble, Andrew Gilbert, Charles Malleson, Adrian Hilton, and John Collomosse. Total capture: 3D human pose estimation fusing video and inertial sensors. In *Proceedings of the British Machine Vision Conference (BMVC)*, pages 14.1–14.13, Sept. 2017. 1

[51] Ruben Villegas, Jimei Yang, Yuliang Zou, Sungryull Sohn, Xunyu Lin, and Honglak Lee. Learning to generate long-term future via hierarchical prediction. In *international conference on machine learning*, pages 3560–3569. PMLR, 2017. 3

[52] Jiashun Wang, Huazhe Xu, Jingwei Xu, Sifei Liu, and Xiaolong Wang. Synthesizing long-term 3D human motion and interaction in 3D scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9401–9411, 2021. 2

[53] Xiaolong Wang, Rohit Girdhar, and Abhinav Gupta. Binge

watching: Scaling affordance learning from sitcoms. In *Proceedings IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[54] Z. Wang, J. Chai, and S. Xia. Combining recurrent neural networks and adversarial training for human motion synthesis and control. *IEEE Transactions on Visualization and Computer Graphics*, 27(1):14–28, 2021. 3

[55] Fei Xia, Amir R. Zamir, Zhi-Yang He, Alexander Sax, Jitendra Malik, and Silvio Savarese. Gibson env: real-world perception for embodied agents. In *Proceedings IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2018. 1

[56] Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, Li Yi, Angel X. Chang, Leonidas J. Guibas, and Hao Su. SAPIEN: A simulated part-based interactive environment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognitio)*, June 2020. 1

[57] Jingwei Xu, Huazhe Xu, Bingbing Ni, Xiaokang Yang, Xiaolong Wang, and Trevor Darrell. Hierarchical style-based networks for motion synthesis. In *European Conference on Computer Vision (ECCV)*, pages 178–194, Cham, 2020. Springer International Publishing. 3

[58] Ye Yuan and Kris Kitani. Dlow: Diversifying latent flows for diverse human motion prediction. In *European Conference on Computer Vision (ECCV)*, 2020. 3, 6

[59] Ye Yuan and Kris M. Kitani. Diverse trajectory forecasting with determinantal point processes. In *International Conference on Learning Representations*, 2020. 3

[60] S. E. Yuksel, J. N. Wilson, and P. D. Gader. Twenty years of mixture of experts. *IEEE Transactions on Neural Networks and Learning Systems*, 23(8):1177–1193, 2012. 2

[61] He Zhang, Sebastian Starke, Taku Komura, and Jun Saito. Mode-adaptive neural networks for quadruped motion control. *ACM Trans. Graph.*, 37(4), July 2018. 2, 7

[62] Siwei Zhang, Yan Zhang, Qianli Ma, Michael J Black, and Siyu Tang. Place: Proximity learning of articulation and contact in 3D environments. In *International Conference on 3D Vision (3DV)*, 2020. 2

[63] Yan Zhang, Michael J. Black, and Siyu Tang. We are more than our joints: Predicting how 3D bodies move. In *Proceedings IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, page 3372–3382, June 2021. 6

[64] Yan Zhang, Mohamed Hassan, Heiko Neumann, Michael J Black, and Siyu Tang. Generating 3D people in scenes without people. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6194–6204, 2020. 2

[65] Yi Zhou, Zimo Li, Shuangjiu Xiao, Chong He, Zeng Huang, and Hao Li. Auto-conditioned recurrent networks for extended complex human motion synthesis. In *International Conference on Learning Representations ICLR*, 2018. 3

[66] Yixin Zhu, Yibiao Zhao, and Song-Chun Zhu. Understanding tools: Task-oriented object modeling, learning and recognition. In *Proceedings IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2015. 2