EUROGRAPHICS 2023 / K. Myszkowski and M. Nießner
(Guest Editors)

COMPUTER GRAPHICS *forum*
Volume 42 (2023), Number 2

Variational Pose Prediction with Dynamic Sample Selection from Sparse Tracking Signals

N. Milef, S. Sueda, and N. Khademi Kalantari

Texas A&M University, United States

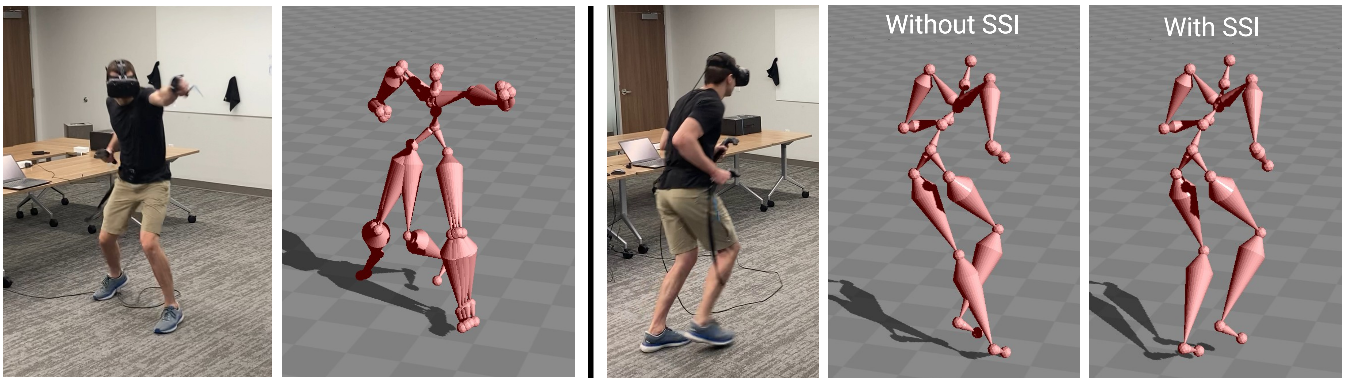


Figure 1: *(Left) We show the ability of our conditional VAE to produce diverse poses from a set of 4 tracking points obtained by a VR device. (Right) We demonstrate the effectiveness of our novel sample selection and interpolation (SSI) strategy. Specifically, we show that our system without SSI produces a result with an implausible pose. Our method with SSI is able to intelligently select a better sample and produce a result with a natural pose.*

Abstract

We propose a learning-based approach for full-body pose reconstruction from extremely sparse upper body tracking data, obtained from a virtual reality (VR) device. We leverage a conditional variational autoencoder with gated recurrent units to synthesize plausible and temporally coherent motions from 4-point tracking (head, hands, and waist positions and orientations). To avoid synthesizing implausible poses, we propose a novel sample selection and interpolation strategy along with an anomaly detection algorithm. Specifically, we monitor the quality of our generated poses using the anomaly detection algorithm and smoothly transition to better samples when the quality falls below a statistically defined threshold. Moreover, we demonstrate that our sample selection and interpolation method can be used for other applications, such as target hitting and collision avoidance, where the generated motions should adhere to the constraints of the virtual environment. Our system is lightweight, operates in real-time, and is able to produce temporally coherent and realistic motions.

CCS Concepts

• ***Computing methodologies*** → ***Neural networks; Motion processing; Virtual reality;***

1. Introduction

As virtual reality (VR) systems become more ubiquitous, there exists a growing desire for more immersive systems. Accurately reconstructing the full-body pose is useful for visualizing other players in multiplayer contexts, performing cheap motion capture, and providing self-embodiment. Unfortunately, VR systems such as the HTC Vive typically only provide sparse positions and orientations for the upper body (e.g., head and hands) via optical and inertial measurement unit tracking. With such sparse inputs, the problem of reconstructing the entire user's body motion becomes extremely challenging due to pose ambiguity.

Recently, a few methods [YKL21; DDC*21; ACB*22; JSQ*22] have been specifically designed to handle extremely sparse VR inputs using deep learning. The approaches by Yang et al. [YKL21] and Jiang et al. [JSQ*22] attempt to solve this problem using a recurrent and a transformer-based neural network, respectively. Unfortunately, these techniques train their model in a supervised manner and as such produce averaged poses for ambiguous motions. A couple of methods [DDC*21; ACB*22] try to address these problems by leveraging generative models. These approaches train an encoder that maps the sparse input to a latent space and a decoder that maps the latent space to full-body poses. By randomly sampling the latent space, these techniques are able to generate different results for the same input tracking points and avoid producing a mean pose. However, because of the randomness of the generative models, they sometimes produce sequences with implausible poses. Moreover, their results suffer from temporal instability as they estimate the results from only the current [ACB*22] or a few previous [DDC*21] observations.

In this paper, we propose a novel framework that addresses these issues and produces temporally stable, realistic, and diverse motions in real time. Specifically, we leverage a conditional variational autoencoder (CVAE) [SLY15] and incorporate gated recurrent units (GRUs) to ensure the generated sequences are temporally coherent. In our system, we use the 4 tracking points (head, hands, and waist) as the input to both the encoder and decoder to produce the full-body pose. Our network also predicts the foot contact labels that are used within a post-process inverse kinematic (IK) stage to prevent foot sliding.

While our system is able to produce diverse and temporally coherent results, the generated results in some cases can have implausible poses because of the IK post-processing. To address this issue, we make a key observation that different samples of the latent space are imperfect at different times. Armed with this observation, we propose a novel sample selection and interpolation strategy along with an anomaly detection algorithm that allows us to evaluate many samples in parallel and transition to the most natural one, once an anomaly is detected with the current sample. In addition to avoiding implausible poses, we demonstrate other applications of our sample selection and interpolation algorithm such as target hitting and collision avoidance.

Through extensive experimental results, we show that our system is able to produce sequences that are better than the state of the art on various examples. In summary, we make the following contributions in this paper:

- We propose a CVAE framework with GRUs for temporally stable full-body pose reconstruction from 4 tracking points (Sec. 3.1).
- We present a novel sample selection and interpolation framework with an anomaly detection algorithm to avoid generating implausible poses (Sec. 3.2).
- We show that our approach produces state-of-the-art results and demonstrate other applications of our sample selection and interpolation framework (Sec. 5).

2. Related Work

With the rise of deep learning, there has been an increased interest in motion synthesis in the research community. Here, we briefly discuss the closely related work to our approach. We refer interested readers to the survey by Mourot et al. [MHL*21] for a thorough review of the subject.

2.1. Generative Motion Synthesis

A large number of approaches use variational autoencoders (VAEs) [KW13] to synthesize motions due to their relative training stability and the ability to generate diverse results. Habibie et al. [HHS*17] show that recurrent VAEs can synthesize realistic human poses with a pose prior. Guo et al. [GZW*20] propose an autoregressive approach that incorporates a VAE with gated recurrent units (GRU) conditioned on the action label and trained with a learned prior [DF18]. Petrovich et al. [PBV21] also generate action-based animation sequences using a transformer-based VAE, but they propose a sequence-level embedding instead of per-frame embedding as used by Guo et al. [GZW*20]. Ghorbani et al. [GWE*20] demonstrate the effectiveness of a hierarchical recurrent character animation VAE that operates on weak signals such as action class and style to synthesize motion sequences. Ling et al. [LZCV20] propose motion VAEs, which allow for realistic generation of primarily locomotion-based sequences such as walking and running. We also leverage a VAE to handle the relevant, but different problem of reconstructing full-body pose from extremely sparse inputs.

2.2. Sparse Pose Prediction

Commercial full-body motion capture systems can accurately record poses, but are expensive and require a complex setup. One way to minimize the costs and increase accessibility to motion capture is to use a small number of sensors and infer the full-body pose from this sparse data. Chai et al. [CH05] present an approach that uses local principal component analysis (PCA) to estimate pose using only a camera and a small number of markers. However, the usage of images or videos limits portability as the camera needs to be calibrated. Liu et al. [LZWM06] use PCA to map a full motion capture setup to a reduced set of tracker points. Liu et al. [LWC*11] compute full-body poses from inertial measurement unit (IMU) sensors using local dynamic models. Von Marcard et al. [VRBP17] use a statistical body model to derive a plausible human pose using 6 IMU sensors. A couple of methods [HKA*18; JYG*22] use supervised temporal neural networks to tackle the 6 IMU pose reconstruction problem. Yi et al. [YZH*22] estimate motion, joint torques, and ground reaction forces, which they show leads to better pose reconstruction accuracy and plausibility for 6 IMU pose reconstruction. Rempe et al. [RBH*21] leverage a VAE as a motion prior for finding the optimal pose from sparse or occluded tracking points. These approaches require 6 or more tracking points, often covering the entire body. In contrast, our approach reconstructs the full-body pose from 4 upper body tracking points.

2.3. VR Pose Prediction

Pose reconstruction in a VR setting is a challenging problem as the VR trackers provide extremely sparse inputs including hands (controllers), head (headset), and optionally waist (only 3 or 4 tracking points). While motion matching [AOG*21; PYAP22] can produce plausible motions, it is constrained by the motions in the

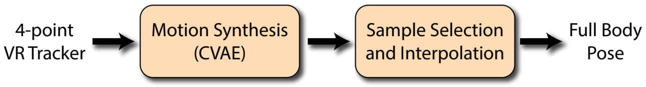


Figure 2: *Our system consists of two main components: motion synthesis using a conditional VAE (Sec. 3.1), and sample selection and interpolation (Sec. 3.2). The 4-point tracking data are used as the input to the CVAE to synthesize motion. The sample selection and interpolation component is responsible for detecting the quality of the sample and transitioning to a better one if the quality falls below a certain threshold.*

dataset, which is a significant limitation in practice. Additionally, this approach requires storing a large number of animations. Yang et al. [YKL21] propose an approach for handling 4-point tracking by breaking the problem into two smaller sub-problems of estimating the upper and lower body poses. Specifically, the upper body is predicted using an inverse kinematics (IK) solver [Mot] and the lower half is estimated using a GRU-based neural network. Most recently, Jiang et al. [JSQ*22] use a transformer architecture to generate poses for both 3 and 4 points tracking. In 3-point tracking, the root orientation and position are estimated using the input data, transforming 3-point tracking into a 4-point tracking problem. Ye et al. [YLHX22] predict full-body pose from 3 tracking points and then feed the estimated pose into a reinforcement learning module to generate the final pose in a physical simulation.Unfortunately, these approaches train their neural network in a deterministic manner, and thus produce average poses for highly ambiguous cases.

To address this issue, a few methods [DDC*21; ACB*22] use generative models to estimate the full-body pose. Specifically, Dittadi et al. [DDC*21] use a VAE for 4-point tracking, but they only provide information from a few previous poses as the input, and thus their results lack temporal coherency. Aliakbarian et al. [ACB*22] introduce a flow-based generative model to produce full-body pose from 3 tracked points. Optimization is performed at test time to find a pose corresponding to given head and hands positions. However, their approach works on a per-frame basis and suffers from temporal inconsistency. Additionally, these generative methods could produce sequences with implausible poses in difficult cases. We also use a generative model in the form of a conditional VAE, but incorporate a GRU to produce temporally consistent results. We also propose a novel sample selection and interpolation strategy to avoid producing results with implausible poses.

3. Methodology

Our goal is to reconstruct full body positions for each frame $p_t \in \mathbb{R}^{19\times3}$ from 4 tracking points. Our system is composed of two main components, as shown in Fig. 2. The first component (Sec. 3.1) is a conditional variational autoencoder (CVAE) [SLY15] with an encoder that maps the sparse inputs to a latent vector and a decoder that converts the sparse input and a latent vector to the full body pose, followed by motion post-processing driven by inverse kinematics (IK). The second component of our system, called sample selection and interpolation (Sec. 3.2), monitors the quality of the generated poses and transitions to better samples when it detects problems with the current sample.

VR systems provide the position and orientation of the tracking points in the global coordinate system. To make it easier for our network to learn a mapping from the input to the output, we transform the positions of the 4 tracking points to the local coordinate space of the character. Specifically, the points are transformed by the inverse waist yaw rotation (Y axis) and normalized to the X and Z of the waist position. The positions along with the velocities of the 4 points in the local coordinate $x_t \in \mathbb{R}^{4\times6}$ serve as the input to our network. Given these inputs, our network estimates the rotations for each joint $r_t \in \mathbb{R}^{18\times6}$ using a 6D rotation representation [ZBL*19] and a contact label for each foot. We use the position and orientation of the root (waist) from the VR system along with the estimated rotations (18 joints) to obtain the position of each joint in world space p_t using forward kinematics. We use the contact labels in an IK post-processing to prevent foot sliding.

3.1. Conditional VAE Motion Synthesis

We propose a conditional VAE (CVAE) to estimate the rotation of each joint r_t from the positions and velocities of the 4 tracking points x_t at each frame t (see Fig. 3). In addition to the rotations, we also estimate the foot contact labels $c_t \in \mathbb{R}^2$, which determine whether the feet are contacting the ground. We use these contact labels to alleviate foot sliding in a post-process, as discussed later.

As is common in VAEs, our system is composed of an encoder and a decoder. Specifically, our encoder estimates the mean μ_t and standard deviation σ_t of a Gaussian distribution from the input. These are then used to obtain the latent vector z_t through the reparametrization trick [KW13], i.e., $z_t = \mu_t + \sigma_t\varepsilon$ where $\varepsilon \sim \mathcal{N}(0,1)$. Our decoder is responsible for mapping this latent vector z_t to the output (joint rotations r_t). Unlike the approach by Dittadi et al. [DDC*21] which only uses the latent vector as the input to the decoder, we condition the decoder on the sparse tracking data x_t, as is common in CVAEs [SLY15]. In addition to providing more constraints for synthesizing high-quality motions, using a CVAE eliminates the need for the encoder during inference [SLY15], and thus improves the speed.

While a CVAE operating on a single frame can produce reasonable poses, the predicted motions suffer from temporal instability. Additionally, since pose reconstruction from sparse trackers is highly under-constrained, a single frame often does not contain all the necessary information for reconstructing high-quality motions. To address this problem, we incorporate gated recurrent units (GRUs) in both the encoder and decoder. Each GRU takes the features from the previous layer, as well as the hidden states (the output of the GRU) of the previous frame h_{t-1} to estimate the hidden states of the current frame h_t. This is in contrast to the approaches by Dittadi et al. [DDC*21] and Jiang et al. [JSQ*22] which use a window of previous sparse tracking points as the input to the encoder. With the GRUs, we intelligently accumulate the information of *all* the past observations to reduce the ill-posedness of the problem and produce temporally coherent motions.

We train the network on a large dataset of diverse motions, by minimizing the following objective:

$$\mathcal{L} = \mathcal{L}_{\text{MSE}}(p_t, \hat{p}_t) + \lambda\mathcal{L}_{KL} + \gamma\mathcal{L}_{\text{BCE}}(c_t, \hat{c}_t). \qquad (1)$$

Here, $\mathcal{L}_{\text{MSE}}$ ensures that the reconstructed poses $\hat{p}_t$ are close to

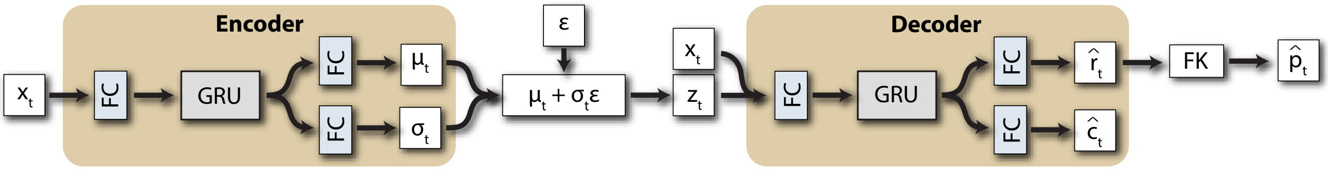


Figure 3: *We show the architecture of our conditional VAE. The positions and velocities of the tracking points (x_t) are passed to the encoder, consisting of a few fully connected (FC) layers and a GRU, to estimate the mean μ_t and standard deviation σ_t of a Gaussian distribution. These are then used along with a random vector ε (from a normal distribution) to generate our random latent vector z_t. The decoder uses the latent vector along with the tracking points and outputs the rotation of each joint $\hat{r}_t$ and the foot contact labels $\hat{c}_t$. The rotations are used along with the position and orientation of the root (waist) to obtain the world positions $\hat{p}_t$ via forward kinematics (FK).*

the ground truth p_t. The KL divergence loss $\mathcal{L}_{KL}$ enforces the latent space to be close to normal distribution. Moreover, we use the binary cross-entropy loss $\mathcal{L}_{\text{BCE}}$ to supervise the foot contact estimates. Finally, λ and γ are the weights of the KL divergence and binary cross-entropy losses, respectively. In our implementation we use $\lambda = 5$ and $\gamma = 0.1$.

Motion Post-Processing: Following existing methods [HSK16; HKS17; YKL21], we improve our synthesized results by applying a post-processing approach using the foot contact estimates from the network. Specifically, when we detect foot contact with the ground (contact estimate above 0.5), we lock down the foot position and use a two-bone IK-solver [HKS17] on the thigh and calf bones. To prevent jerkiness when the foot is lifted, we linearly interpolate between the contact position and the output of the neural network.

Discussion: By using different random vectors ε to generate a latent vector, our system is able to generate a diverse set of poses given the 4 input tracking points. As shown in Fig. 4 our results with different random vectors are drastically different for the less constrained lower body, but are similar for the upper body. This is one of the major advantages of our system compared to the approaches using deterministic training [YKL21; JSQ*22]. In their case, the network produces an average pose for the highly ambiguous poses, which is often implausible.

3.2. Sample Selection and Interpolation

While our CVAE with IK post-processing can produce temporally coherent sequences without foot sliding, the results for highly ambiguous cases may still contain implausible poses. In these cases, the predicted foot contact labels may not match the generated pose by the network. This mismatch causes our IK post-processing to produce implausible poses, as shown in Fig. 5 (left).

To address this problem, we observe that *not every sample* of the latent space produces imperfect poses *at the same time*. Therefore, as shown in Fig. 5 (right), there usually exist other samples that can produce plausible poses. Guided by this observation, we propose a novel sample selection and interpolation strategy, as illustrated in Fig. 6. Specifically, during inference, we use a set of N random vectors $\mathcal{E} = \{\varepsilon^1, \cdots, \varepsilon^N\}$ to generate N poses at each time step in parallel (i.e., in a batch). Note that each ε in the initial sampling pass remains the same for every frame. Randomly resampling ε each frame causes jitter in the resulting reconstructed animation. At the beginning, we choose one of these vectors as the current sample ε^c to generate our sequence. Once an error is detected with the current sample ε^c, we find the best sample ε^b among the N samples and transition to it.

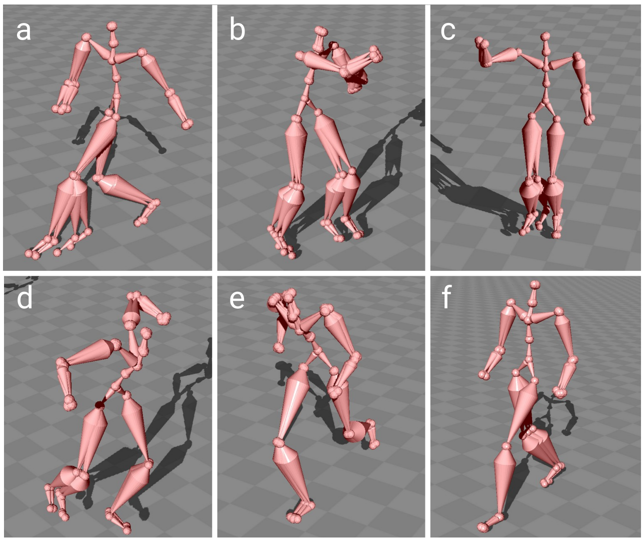


Figure 4: *Our system is able to generate diverse motions for tracker sequences. More ambiguous tracker sequences such as dancing (**a**) or boxing (**b**) tend to produce more diverse outputs while animations such as running (**e**) or walking (**f**) in a straight line tends to produce more deterministic outputs due to high correlation of the lower and upper body.*

As shown in Fig. 6, there are three major challenges with this system that need to be addressed: **1)** how to detect problem with the current sample, **2)** how to select the new best sample, and **3)** how to transition to this sample. In the next sections, we discuss our proposed method to address each challenge.

3.2.1. Anomaly Detection

Our goal here is to quantitatively measure the quality of a generated pose using the current sample ε^c and detect whether it is implausible. We propose to do so by projecting the generated pose to the closest plausible pose. We then compare the distance between the joints in the two poses against a statistically defined threshold to determine if the generated pose is implausible. Specifically, inspired by a category of anomaly detection approaches [PSCH21; HHWB02], we use an autoencoder to learn the space of plausible poses by training it on a large motion capture dataset. During train-

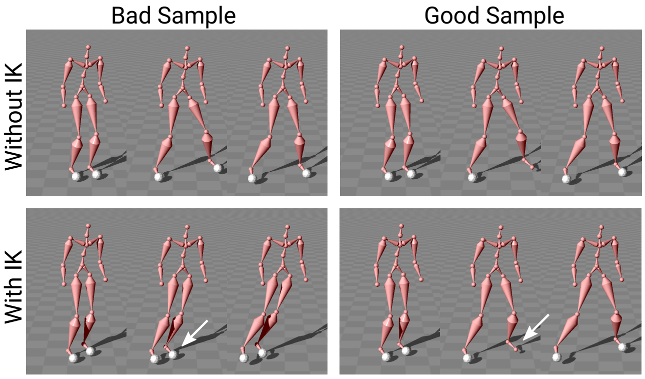


Figure 5: *On the top-left, we show the output of our CVAE (before IK post-processing) for three frames of a sequence. Note that in addition to the pose, we also visualize the foot contact labels with white spheres; i.e., if the network detects contact with the ground for a foot, we show a sphere on that foot. In this sequence, the character is moving its left foot to the right side of the screen. However, our network detects the left foot to be in contact with the ground at all times. This mismatch between the generated poses and contact labels causes our IK post-processing to lock down both feet to the ground and produce implausible poses (bottom-left). On the right, we show the results before and after IK post-processing for a different sample. In this case, the estimated contact labels are consistent with the generated motion (top-right), and thus our generated poses after IK post-processing are plausible.*

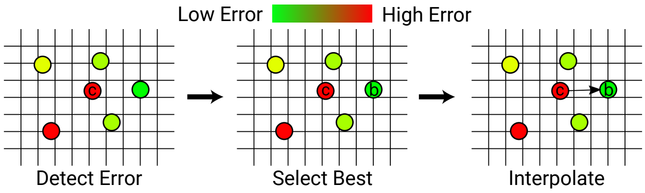


Figure 6: *We show an overview of our sample selection and interpolation process (illustrated in 2D for simplicity). We run a set of N samples with different random vectors in parallel, and choose one of them as the current sample c at the beginning. Once the error of the current sample goes above a statistically defined threshold, we find the best sample b in a pool of $N-1$ samples and transition to it over a period of T frames. Once the transition is complete, the best sample b is used as the current sample.*

ing, the encoder takes the input pose and maps it to a latent vector, while the decoder takes this vector as the input and attempts to reconstruct the same input pose. Since this autoencoder is trained on a dataset of natural poses, the latent space corresponds to all the plausible poses. Similar to our CVAE, we use the joint positions in the local coordinate space of the character as the input and estimate the rotations as the output. These rotations are used to obtain the position of the joints in world space using forward kinematics.

During testing, we pass the generated pose after IK post-processing (using the current sample) to this trained autoencoder and measure the reconstruction error for each joint (see Fig. 7), i.e., the error between the input and output positions for each joint. If the generated pose is in-distribution (plausible), the output of the autoencoder will be close to the input and the reconstruction error will be small. However, for an out-of-distribution (implausible) pose, the difference between the output and input will be large. Since implausible poses are mainly because of the the lower body joints (due to their ambiguity), we measure the quality of the generated pose using the current sample as the average of reconstruction error across all the lower body joint positions $\mathcal{R}(\varepsilon^c)$.

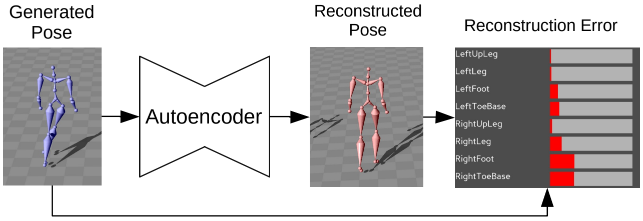


Figure 7: *We train an autoencoder on a large motion capture dataset. During testing, we pass the generated pose after IK post processing to the trained network. If the generated pose is out-of-distribution (implausible), the reconstructed pose using this autoencoder will be significantly different from the generated pose. We use the reconstruction error (difference of the lower body joints position between the generated and reconstructed poses) as a metric to measure the quality.*

To detect whether the estimated pose by the current sample is implausible, we can compare the average reconstruction error against a pre-defined threshold. However, reconstruction error varies for different poses, making it difficult to establish a universal threshold. Therefore, we propose to define our threshold in a statistical manner based on the reconstruction error across all the samples for the current frame. Specifically, we determine whether the current sample is an outlier, by comparing its average reconstruction error to all the samples. We do this using a trimmed estimator as follows:

$$r = \operatorname{median}(\mathcal{R}(\mathcal{E})) - \min(\mathcal{R}(\mathcal{E})), \tag{2}$$

where $\mathcal{E} = \{\varepsilon^1, \cdots, \varepsilon^N\}$ represents the set of all the samples. In our implementation, we consider the current pose an outlier (implausible) if its average reconstruction error is greater than $2r$, i.e., $\mathcal{R}(\varepsilon^c) > 2r$.

3.2.2. Sample Selection

Once the current sample ε^c is flagged as implausible, we need to identify the best sample ε^b among the $N-1$ samples. To do so, we look through all the other $N-1$ samples and find the sample that generate the pose with the smallest average reconstruction error.

3.2.3. Sample Interpolation

Once the best sample is identified, we need to transition to this sample ε^b from the current sample ε^c. Immediately jumping over to the best sample creates discontinuity in the motion as different samples could potentially generate drastically different poses (see Fig. 4). Poses differ in both temporal phase (i.e., some joints start moving before others) and rotation of joints. Therefore, we propose to smoothly transition from the current sample to the best one over the period of T frames (T equal to 10 in our implementation). We do this by linearly interpolating the random vectors for the current

ε^c and best ε^b samples. We also linearly interpolate the input hidden states of the GRU unit in our decoder between the current h^c_{t-1} and the best h^b_{t-1} samples because the hidden states accumulate information over the synthesized motions up to the current frame and are different for these two samples. Once the interpolated random vector and input hidden states are obtained, we use them as the input to our CVAE to synthesize the interpolated pose at the current time. Continuing this process for T frames provides a smooth transition from the current to the best sample. Once the transition is complete, the best sample becomes our current sample.

3.2.4. Look Ahead Transition

Switching to a better sample once the current sample is detected as implausible is often too late because the implausible pose will then appear. Therefore, we favor a small amount of latency in exchange for the ability to "look ahead" at future frames to prevent producing an implausible pose. For multiplayer pose reconstruction [CC06; VSS19; MRQ*21] or motion capture applications, such a latency is acceptable. Specifically, we perform the anomaly (error) detection and best sample selection (see Fig. 6) at frame $t+\tau$. This is done by keeping a buffer of τ frames (GRU inputs, predicted pose, etc.) for all the samples. If an anomaly is detected for frame $t+\tau$, we find the best sample at that frame (ε_b). We then transition from ε_c at frame t to ε_b at frame $t+\tau$. In our implementation, we use a look ahead of $\tau = 10$ frames at 60Hz, or 167ms.

3.2.5. Application-Dependent Error-Detection Module

Beyond preventing implausible poses, there are other scenarios where our sample selection and interpolation is beneficial. Specifically, we can use our framework to synthesize motions that match the constraints of the virtual environment. For example, in kickboxing, it would not make sense to perform a low kick when a coach is holding the kicking pads near head height. To synthesize motion according to the environment constraints, we replace our anomaly detection module with an application-dependent error-detection module. We discuss two specific applications of our framework (target hitting and collision avoidance) in Sec. 5.3.

4. Implementation

In this section, we describe the necessary details for implementing our approach.

4.1. Data Preprocessing

To construct our dataset, we create a combination of the Edinburgh [HSK16; HKS17], CMU [CMU00], and HDM05 [MRC*07] motion capture datasets. All of the training data share the same kinematic tree and bone length. Our training set is composed of 2,317 sequences, which covers over 4 hours of animation. We train our CVAE network on sequences of 60 frames sampled at 60Hz.

Since the motion capture datasets do not contain ground truth foot contact labels, we need to compute these labels according to the placement of the feet with respect to the ground. Specifically, we consider a foot to be in contact with the ground if both its position and velocity are below a certain threshold. Note that we use velocity to determine contact to avoid incorrectly detecting cases where the foot is moving quickly at a small height as in contact with the ground. To set the height threshold for each sequence, we first calculate the 10 percentile height of each toe and select the smaller value. We then add 5cm to this value and use it as our contact threshold height for that sequence. For velocity, we use a fix threshold of $1cm$ per frame. Note that since the velocities are usually noisy, we smooth them using a Savitzky-Golay filter (window size of 11 and polynomial order of 3) before comparing them against this threshold. We found this approach works well for labeling foot contacts after manually inspecting the labels for the motion capture data.

Table 1: *We numerically compare our technique against state-of-the-art approaches on a test set of 398 sequences in terms of three metrics. We also show the results of the autoencoder version of our method.*

Method	MPJPE (cm)	MPJVE (cm/s)	MPJAE (cm/s^2)
LoBSTr [YKL21]	8.95	126.0	7593
VAE-HMD [DDC*21]	10.85	62.3	3171
AvatarPoser [JSQ*22]	9.43	**53.5**	3465
Ours (autoencoder)	**8.62**	59.4	2984
Ours	9.24	54.4	**2724**

4.2. Architecture

CVAE: The encoder consists of a fully connected layer (24 inputs and 128 outputs), a GRU (128 inputs and a hidden size of 128), and finally two independent fully connected layers (128 inputs and 30 outputs each) for μ and σ. Note that both μ and σ are vectors with 30 elements as our latent vector is of size 30. The decoder consists of a fully connected layer (54 inputs and 128 outputs), a GRU (128 inputs and a hidden size of 128) followed by a ReLU activation function on the hidden state, and two independent fully connected layers that use the GRU's hidden state as input. The first fully connected layer predicts the pose (128 inputs and 54 outputs) while the second one is followed by a sigmoid function and estimates the foot contact labels (128 inputs and 2 outputs).

Anomaly detection network The error detection network is an autoencoder with 7 layers. The inputs are the local positions $p_t \in \mathbb{R}^{19\times3}$ for all 19 joints of the skeleton (same normalization as the CVAE), and the outputs are the rotations for forward kinematics $\mathbb{R}^{18\times6}$. The layer sizes are as follows: $57 \rightarrow 28 \rightarrow 14 \rightarrow 7 \rightarrow 7 \rightarrow 14 \rightarrow 28 \rightarrow 108$. All layers except for the last layer are followed by a ReLU activation.

4.3. Training

We implement our approach in PyTorch [PGM*19] and use Adam optimizer [KB14] to train all our networks. Specifically, we train our CVAE with a learning rate of 0.0002, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\lambda = 0.0001$ for 500,000 iterations, while we use a learning rate of 0.0002, $\beta_1 = 0.5$, $\beta_2 = 0.999$, and $\lambda = 0.0001$ for 100,000 iterations, to train our anomaly detection network.

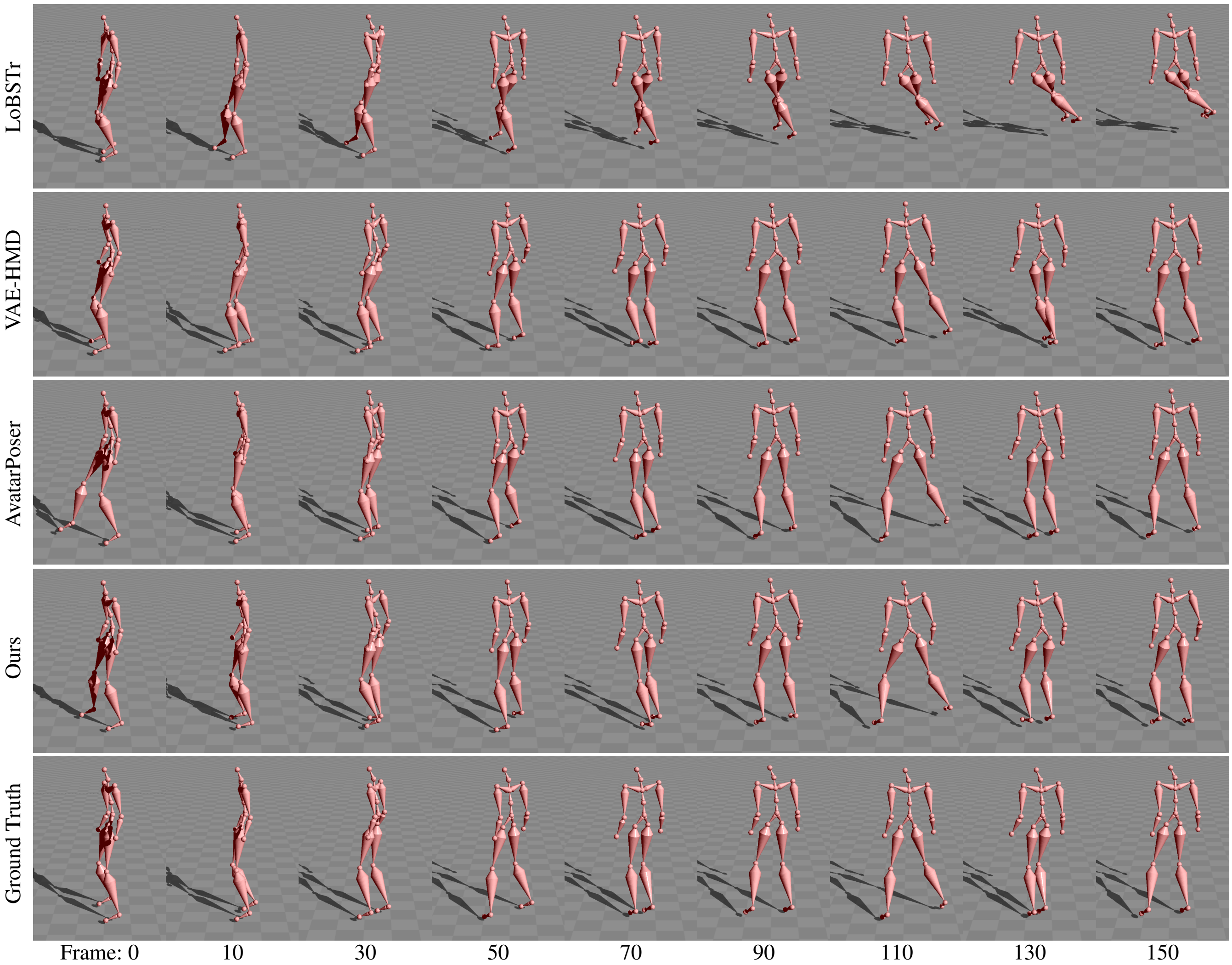


Figure 8: *We show visual comparison against the other methods on a challenging test sequence. Both AvatarPoser and LoBSTr (supervised approaches) produce results with implausible poses. The right foot in AvatarPoser is locked to the ground because of incorrect contact labels in frame 0, generating an unnatural motion. Moreover, VAE-HMD produces results with intersecting legs in frame 130. For our approach, we purposefully select a bad sample at start, but our method successfully transitions to better samples and produces plausible poses.*

4.4. VR System Setup

In our real-world data setup, we use the HTC Vive tracking system and read the tracking data at 60Hz. While the HTC Vive trackers provide fairly accurate tracking points [BSC*18], we observed that they can be further improved through post-processing. Specifically, the waist tracker generally provides the least stable position and particularly orientation. This becomes especially problematic in motions that involve explosive movements, such as jumping and running, where the tracker can oscillate due to large forces and unstable anchoring. To reduce this, we filter the waist tracker's orientation by applying a typical lowpass Butterworth filter (order 6, cutoff frequency=100Hz) to the quaternion components independently. We normalize the quaternion after filtering is performed.

Another problem is matching a reasonable skeleton from the motion capture dataset. First, the waist tracker tends to have an offset vertically and in the forward direction of the user. Second, the actors in the training data tend to have a certain range of height. The network can handle some differences in height, but it is best to normalize the tracker data to a more common height from the dataset. We correct for both of these issues with a calibration stage where the user stands in a T-pose.

5. Results

In this section we extensively compare our approach against the other methods and perform ablation experiments to show the effectiveness of our proposed components. Note that we show individual frames in the paper, but the full sequences are shown in the supplementary video.

Table 2: *Comparisons against the other methods in terms of the execution time*

Method	Time (ms)
LoBSTr [YKL21]	0.20
VAE-HMD [DDC*21]	1.47
AvatarPoser [JSQ*22]	1.08
Ours (interpolation)	0.55
Ours	0.39

5.1. Comparisons to Other Methods

We compare our method against three state-of-the-art techniques by Yang et al. [YKL21] (LoBSTr), Dittadi et al. [DDC*21] (VAE-HMD), and Jiang et al. [JSQ*22] (AvatarPoser). For LoBSTr [YKL21], we use the authors' code for the lower-body pose estimation and reproduce the FinalIK [Mot] driven setup for the upper body pose estimation. Since the code for VAE-HMD [DDC*21] is not publicly available, we use our implementation as the representation of their approach. For AvatarPoser [JSQ*22] we use the publicly available source code, provided by the authors. Moreover, we improve VAE-HMD and AvatarPoser by adding foot contact label prediction to their networks and implementing the IK post-processing. This significantly improves their results and mitigates the foot sliding artifacts. Note that LoBSTr, similar to ours, performs IK post-processing using the predicted foot contact labels. All the approaches use the same 4 tracking points as the input and we train them on our dataset to ensure fair comparison. Throughout this section we use $N = 100$ samples for our sample selection and interpolation strategy, unless otherwise stated.

Quantitative: We numerically compare our approach against the other approaches on a test set consisting of 398 sequences in Table 1. We also show the results of the autoencoder version of our approach, which we discuss later in Sec. 5.4. We use common metrics to evaluate the quality of estimated poses: mean per joint position error (MPJPE), mean per joint velocity error (MPJVE), and mean per joint acceleration error (MPJAE). While our approach produces reasonable results, it produces slightly higher error for some metrics compared to the deterministic approaches (LoBSTr and AvatarPoser). This is, however, expected as our approach is based on CVAE which aims to produce results that are plausible, but not necessarily match the ground truth.

Qualitative: In Fig. 8, we show visual comparison against the other methods on one of the test sequences. LoBSTr produces results with implausible poses throughout the sequence, while VAE-HMD produces a pose with intersecting legs at frame 130. Moreover, AvatarPoser produces results with an unnatural motion around frame 0 (see the supplementary video). Additionally, VAE-HMD and AvatarPoser produce results that lack temporal coherency as they only rely on a limited number of previous poses to estimate the current one.

Inference Speed: We compare our timings against the other methods in Table 2. All the timings are obtained on an NVidia GeForce RTX 2070 SUPER GPU. Moreover, we exclude the timings for forward and inverse kinematics for all the approaches, as the implementation is the same for all methods and can be implemented in various ways (e.g., CPU vs. CUDA kernel). For our approach, we report the timings for when the current sample is plausible, "Ours", and during transition from the current to the best sample, "Ours (interpolation)". During a transition, which is performed in a small number of frames (10 in our implementation), our system requires an additional network evaluation using interpolated latent vector and hidden states. Overall, our method is slightly slower than LoBSTr (because of additional network evaluations), but significantly faster than VAE-HMD and AvatarPoser that use large neural networks. Note that, unlike VAE-HMD, our VAE is conditional, and thus we do not need to use our encoder during inference. This significantly saves performance, which is critical for production VR applications and VR platforms on limited hardware such as the Meta Quest 2. Our timing for "Ours" consists of 0.17ms for CVAE and 0.22 ms for the anomaly detection networks with 100 samples, while the number for "Ours (interpolation)" contains an additional 0.16 ms for evaluating our CVAE with the interpolated sample.

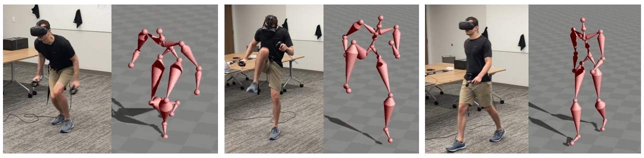
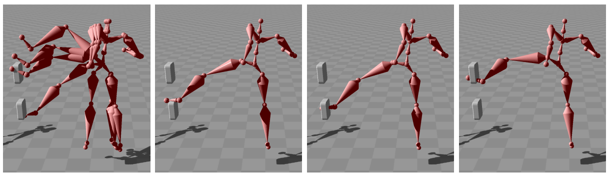


Figure 9: *Pose prediction for real VR device. Our network is capable of generating difficult motions such as knee strikes or jumps.*



(a) five random (b) w/o SSI (c) low target (d) high target

Figure 10: *We show one frame of a sequence where the goal is to synthesize motions of kicking a specific target.* ***(a)*** *we show the results of our CVAE using five random samples from the same 4 tracking points.* ***(b)*** *Without sample selection and interpolation (SSI) we can generate a sequence with a random sample and do not have any control over the kicking position. With SSI, we can synthesize motions of kicking a low* ***(c)*** *or a high* ***(d)*** *target.*

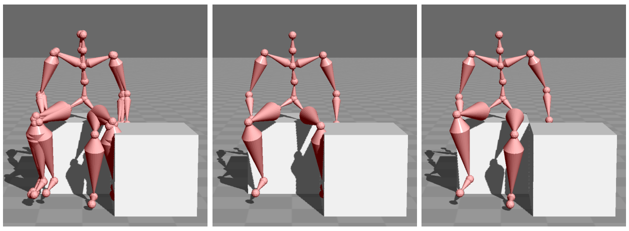
5.2. Real VR Device

In Fig. 9, we show the results of our approach using 4 tracking points obtained via a HTC Vive VR system. We use the approach described in Sec. 4.4 to process and calibrate the tracking points before using them as the input to our system. As seen, our system is able to generate plausible poses for difficult sequences such as knee strikes and jumps.

5.3. Other Applications of Sample Selection and Interpolation

In this section, we present the two applications of our sample selection and interpolation strategy, where the goal is to produce motions that satisfy the contextual constraints of the environment.

Target Hitting: Kicking or kneeing an object is common in sports such as martial arts. In these situations, there often exists a certain target that needs to be hit. For example, in kick boxing, the goal



(a) five random samples (b) w/o SSI (c) with SSI

Figure 11: *Motion sequence of an actor sitting next to another seat.* ***(a)*** *Five random samples producing different motions.* ***(b)*** *Our method without SSI could produce results where the leg penetrates into the right box.* ***(c)*** *Our method with SSI produces results that avoid leg penetration.*

may be to hit a kicking pad that is held near head height. Unfortunately, the target position of the end-effector (e.g., knee or foot) is often ambiguous from the upper body movement alone. So by simply estimating the full pose from the sparse inputs, we may not be able to properly position the end-effector with respect to the target.

We use our sample selection and interpolation to address this application. Specifically, we use $L2$ distance between target position p_T and estimated end-effector joint location $p(\varepsilon^i)$ using each sample ε^i as our sample selection criteria:

$$\varepsilon^b = \operatorname*{argmin}_{\varepsilon^i}(\left\| p_T - p(\varepsilon^i) \right\|_2). \tag{3}$$

As shown in Fig. 10, using our sample selection and interpolation (SSI), our approach is able to synthesize motions that kick both the low and high targets from the same 4 tracking points. In Table 3, we evaluate the effect of number of samples on the accuracy of the low and high target kicks. As seen, the mean and standard deviation of the error decreases by increasing the number of samples, as our approach will have more flexibility in choosing the appropriate sample. We found that 100 samples provides the best trade-off between accuracy and speed.

Collision Avoidance: In cases where there are objects surrounding the character (see Fig. 11), we would like to synthesize poses that do not penetrate the environmental objects. Our sample selection and interpolation strategy can be used to address this application. In this case, our sample selection criteria is the maximum distance between the predicted joint position $p(\varepsilon^i)$ and the environmental object position p_E:

$$\varepsilon^b = \operatorname*{argmax}_{\varepsilon^i}(\left\| p_E - p(\varepsilon^i) \right\|_2). \tag{4}$$

As shown in Fig. 11, our approach with sample selection and interpolation avoids estimating a pose that collides with the surrounding objects. We also numerically evaluate the effect of number of samples for this application in Table 3. As seen, by increasing the number of samples, our approach is able to produce poses with fewer number of collisions.

Table 3: *We show the mean and standard deviation of errors for sample selection and interpolation applications across 100 runs. The error for the high and low kick tasks are the distance (in meters) from the target point. The error for the sitting task is the ratio between the number of left knee collisions to the total number of frames.*

# samples	high kick (m)		low kick (m)		sit (# collisions)	
	μ	σ	μ	σ	μ	σ
1	0.24	0.16	0.27	0.19	0.22	0.33
10	0.12	0.04	0.12	0.04	0.11	0.25
100	0.10	0.02	0.10	0.02	0.07	0.19
1000	0.08	0.02	0.08	0.02	0.09	0.23

5.4. Ablations

VAE vs Autoencoder: We begin by numerically comparing our approach against the autoencoder version of our technique in Table 1. For the autoencoder version of our model, we use the same network architecture as ours, but replace the μ and σ outputs with a fully-connected layer. As seen, our autoencoder model produces better results in terms of MPJPE which is essentially the metric it is trained on. However, as shown in the supplementary video, it produces results with implausible poses.

Sample Selection and Interpolation: We show the impact of our sample selection and interpolation (SSI) strategy in Fig. 12. For a fair comparison, we start at the same sample using the same seed in both cases. As seen, without SSI, our approach produces poses with crossed legs. On the other hand, our approach with SSI avoids producing such implausible poses.

6. Limitations and Future Work

Our network's ability to generalize outside of the training distribution depends on whether there are aspects of the motion captured in the training data. For example, our training data does not include motions such as kicking a soccer ball. For such sequences, our network produces results where the character does not fully extend its leg (see Fig. 13). While this can be addressed by adding these types of sequences to the training data, handling out-of-distribution motions would be an interesting problem to address in the future. Moreover, our trimmed estimator can fail to detect an implausible pose if the majority of the samples contain similar implausible poses (see Fig. 14). This could be improved with a better estimator, such as clustering algorithms, but we leave this as future work.

While our method produces realistic poses, the hand and head positions do not exactly line up with the trackers. In applications, such as self-embodiment, where matching the exact tracking positions is important, we can use an additional IK pass on the arms to improve the accuracy of the estimated upper body poses. Finally, calibration of the user is still a coarse process and could be improved. We fit the user's tracking positions to a skeleton from the training data, but it would be useful to be able to determine the bone lengths with a more complex calibration phase.

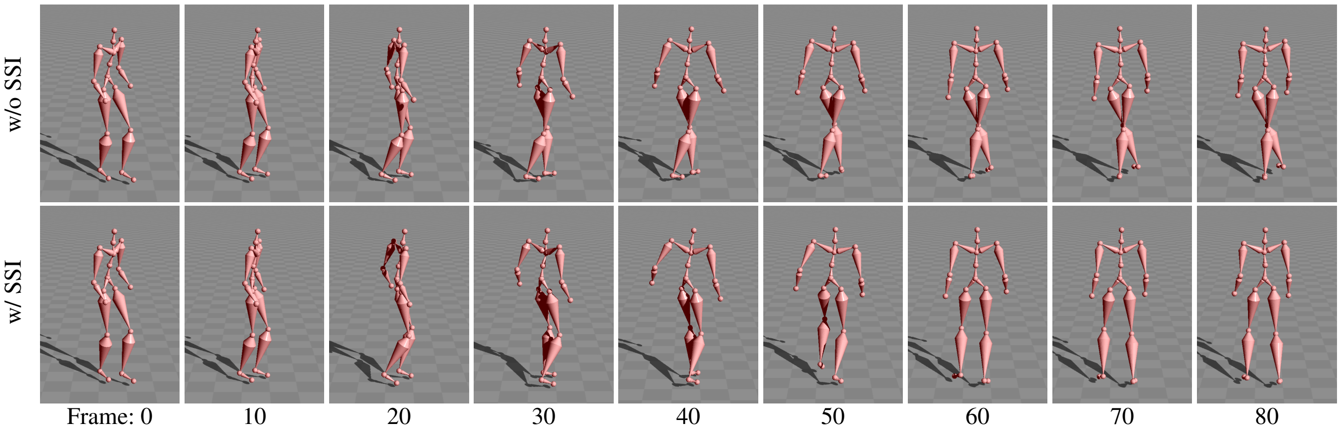
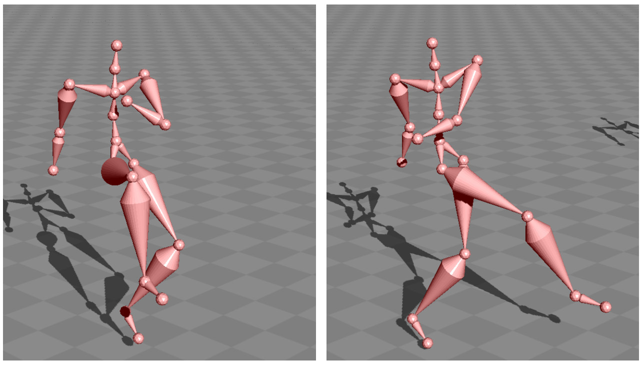
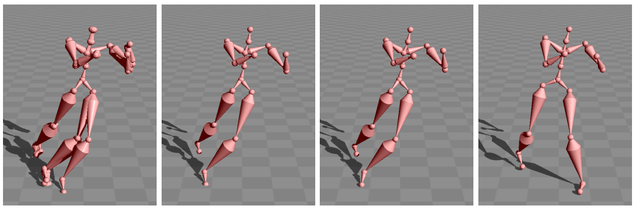


Figure 12: *Without sample selection, the legs cross because the left leg gets caught in the ground during the turn due to an incorrect contact label.*



(a) ours (b) ground truth

Figure 13: *Kicking a ball is an animation outside of our training distribution, so it is difficult to reconstruct this pose.*



(a) five random (b) w/o SSI (c) with SSI (d) ground truth

Figure 14: *If the majority of samples are of low quality, our anomaly detection algorithm will not flag the current sample as implausible.*

7. Conclusion

We have presented a novel pose prediction approach from 4 tracking points. Specifically, we leverage a conditional VAE with GRUs to produce diverse and temporally coherent motions. Moreover, we propose a novel sample selection and interpolation strategy to avoid producing implausible poses. Moreover, we demonstrate that this strategy can be used to synthesize motions that adhere to the constraints of the virtual environment. Through extensive experimental results, we show that our approach produces better results than the state-of-the-art techniques.

Acknowledgements

This work was funded in part by the National Science Foundation (CAREER-1846368) and a generous gift from Adobe. The data used in this project was obtained from mocap.cs.cmu.edu [CMU00], HDM05 [MRC*07], and Edinburgh University [HSKJ15; HSK16]. The CMU database was created with funding from NSF EIA-0196217.

References

[ACB*22] Aliakbarian, Sadegh, Cameron, Pashmina, Bogo, Federica, et al. *FLAG: Flow-based 3D Avatar Generation from Sparse Observations*. 2022 1–3.

[AOG*21] Ahuja, Karan, Ofek, Eyal, Gonzalez-Franco, Mar, et al. "Coolmoves: User motion accentuation in virtual reality". *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 5.2 (2021), 1–23 2.

[BSC*18] Borges, Miguel, Symington, Andrew, Coltin, Brian, et al. "HTC vive: Analysis and accuracy improvement". *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, 2610–2615 7.

[CC06] Claypool, Mark and Claypool, Kajal. "Latency and player actions in online games". *Communications of the ACM* 49.11 (2006), 40–45 6.

[CH05] Chai, Jinxiang and Hodgins, Jessica K. "Performance animation from low-dimensional control signals". *ACM SIGGRAPH 2005 Papers*. 2005, 686–696 2.

[CMU00] CMU Graphics Lab. *CMU Graphics Lab Motion Capture Database*. `http://mocap.cs.cmu.edu/`. 2000 6, 10.

[DDC*21] Dittadi, Andrea, Dziadzio, Sebastian, Cosker, Darren, et al. "Full-Body Motion From a Single Head-Mounted Device: Generating SMPL Poses From Partial Observations". *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, 11687–11697 1–3, 6, 8.

[DF18] Denton, Emily and Fergus, Rob. "Stochastic video generation with a learned prior". *International conference on machine learning*. PMLR. 2018, 1174–1183 2.

[GWE*20] Ghorbani, Saeed, Wloka, Calden, Etemad, Ali, et al. "Probabilistic character motion synthesis using a hierarchical deep latent variable model". *Computer Graphics Forum*. Vol. 39. 8. Wiley Online Library. 2020, 225–239 2.

[GZW*20] Guo, Chuan, Zuo, Xinxin, Wang, Sen, et al. "Action2motion: Conditioned generation of 3d human motions". *Proceedings of the 28th ACM International Conference on Multimedia*. 2020, 2021–2029 2.

[HHS*17] Habibie, Ikhsanul, Holden, Daniel, Schwarz, Jonathan, et al. “A recurrent variational autoencoder for human motion synthesis”. *28th British Machine Vision Conference*. 2017 2.

[HHWB02] Hawkins, Simon, He, Hongxing, Williams, Graham, and Baxter, Rohan. “Outlier detection using replicator neural networks”. *International Conference on Data Warehousing and Knowledge Discovery*. Springer. 2002, 170–180 4.

[HKA*18] Huang, Yinghao, Kaufmann, Manuel, Aksan, Emre, et al. “Deep inertial poser: Learning to reconstruct human pose from sparse inertial measurements in real time”. *ACM Transactions on Graphics (TOG)* 37.6 (2018), 1–15 2.

[HKS17] Holden, Daniel, Komura, Taku, and Saito, Jun. “Phase-functioned neural networks for character control”. *ACM Transactions on Graphics (TOG)* 36.4 (2017), 1–13 4, 6.

[HSK16] Holden, Daniel, Saito, Jun, and Komura, Taku. “A deep learning framework for character motion synthesis and editing”. *ACM Transactions on Graphics (TOG)* 35.4 (2016), 1–11 4, 6, 10.

[HSKJ15] Holden, Daniel, Saito, Jun, Komura, Taku, and Joyce, Thomas. “Learning motion manifolds with convolutional autoencoders”. *SIGGRAPH Asia 2015 technical briefs*. 2015, 1–4 10.

[JSQ*22] Jiang, Jiaxi, Streli, Paul, Qiu, Huajian, et al. “Avatar-Poser: Articulated Full-Body Pose Tracking from Sparse Motion Sensing”. *Proceedings of European Conference on Computer Vision*. Springer. 2022 1–4, 6, 8.

[JYG*22] Jiang, Yifeng, Ye, Yuting, Gopinath, Deepak, et al. “Transformer Inertial Poser: Attention-based Real-time Human Motion Reconstruction from Sparse IMUs”. *arXiv preprint arXiv:2203.15720* (2022) 2.

[KB14] Kingma, Diederik P and Ba, Jimmy. “Adam: A method for stochastic optimization”. *arXiv preprint arXiv:1412.6980* (2014) 6.

[KW13] Kingma, Diederik P and Welling, Max. “Auto-encoding variational bayes”. *arXiv preprint arXiv:1312.6114* (2013) 2, 3.

[LWC*11] Liu, Huajun, Wei, Xiaolin, Chai, Jinxiang, et al. “Realtime human motion control with a small number of inertial sensors”. *Symposium on interactive 3D graphics and games*. 2011, 133–140 2.

[LZCV20] Ling, Hung Yu, Zinno, Fabio, Cheng, George, and Van De Panne, Michiel. “Character controllers using motion vaes”. *ACM Transactions on Graphics (TOG)* 39.4 (2020), 40–1 2.

[LZWM06] Liu, Guodong, Zhang, Jingdan, Wang, Wei, and McMillan, Leonard. “Human motion estimation from a reduced marker set”. *Proceedings of the 2006 symposium on Interactive 3D graphics and games*. 2006, 35–42 2.

[MHL*21] Mourot, Lucas, Hoyet, Ludovic, Le Clerc, François, et al. “A Survey on Deep Learning for Skeleton-Based Human Animation”. *Computer Graphics Forum*. Wiley Online Library. 2021 2.

[Mot] Motion, Root. *Final IK* 3, 8.

[MRC*07] Müller, M., Röder, T., Clausen, M., et al. *Documentation Mocap Database HDM05*. Tech. rep. CG-2007-2. Universität Bonn, June 2007 6, 10.

[MRQ*21] Milef, Nicholas, Ryason, Adam, Qi, Di, et al. “Disruptions to shared mental models from poor quality of service in collaborative virtual environments”. *Scientific reports* 11.1 (2021), 1–12 6.

[PBV21] Petrovich, Mathis, Black, Michael J, and Varol, Gül. “Action-Conditioned 3D Human Motion Synthesis with Transformer VAE”. *arXiv preprint arXiv:2104.05670* (2021) 2.

[PGM*19] Paszke, Adam, Gross, Sam, Massa, Francisco, et al. “Pytorch: An imperative style, high-performance deep learning library”. *Advances in neural information processing systems* 32 (2019) 6.

[PSCH21] Pang, Guansong, Shen, Chunhua, Cao, Longbing, and Hengel, Anton Van Den. “Deep learning for anomaly detection: A review”. *ACM Computing Surveys (CSUR)* 54.2 (2021), 1–38 4.

[PYAP22] Ponton, Jose Luis, Yun, Haoran, Andujar, Carlos, and Pelechano, Nuria. “Combining Motion Matching and Orientation Prediction to Animate Avatars for Consumer-Grade VR Devices”. *Computer Graphics Forum* (2022) 2.

[RBH*21] Rempe, Davis, Birdal, Tolga, Hertzmann, Aaron, et al. “Humor: 3d human motion model for robust pose estimation”. *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, 11488–11499 2.

[SLY15] Sohn, Kihyuk, Lee, Honglak, and Yan, Xinchen. “Learning structured output representation using deep conditional generative models”. *Advances in neural information processing systems* 28 (2015) 2, 3.

[VRBP17] Von Marcard, Timo, Rosenhahn, Bodo, Black, Michael J, and Pons-Moll, Gerard. “Sparse inertial poser: Automatic 3d human pose estimation from sparse imus”. *Computer Graphics Forum*. Vol. 36. 2. Wiley Online Library. 2017, 349–360 2.

[VSS19] Vlahovic, Sara, Suznjevic, Mirko, and Skorin-Kapov, Lea. “The impact of network latency on gaming QoE for an FPS VR game”. *2019 Eleventh International Conference on Quality of Multimedia Experience (QoMEX)*. IEEE. 2019, 1–3 6.

[YKL21] Yang, Dongseok, Kim, Doyeon, and Lee, Sung-Hee. “LoBSTr: Real-time Lower-body Pose Prediction from Sparse Upper-body Tracking Signals”. *Computer Graphics Forum*. Vol. 40. 2. Wiley Online Library. 2021, 265–275 1, 3, 4, 6, 8.

[YLHX22] Ye, Yongjing, Liu, Libin, Hu, Lei, and Xia, Shihong. “Neural3Points: Learning to Generate Physically Realistic Full-body Motion for Virtual Reality Users”. *Computer Graphics Forum* (2022). ISSN: 1467-8659. DOI: `10.1111/cgf.14634` 3.

[YZH*22] Yi, Xinyu, Zhou, Yuxiao, Habermann, Marc, et al. “Physical Inertial Poser (PIP): Physics-aware Real-time Human Motion Tracking from Sparse Inertial Sensors”. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, 13167–13178 2.

[ZBL*19] Zhou, Yi, Barnes, Connelly, Lu, Jingwan, et al. “On the continuity of rotation representations in neural networks”. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, 5745–5753 3.