Avatars Grow Legs: Generating Smooth Human Motion from Sparse Tracking Inputs with Diffusion Model

Yuming Du* Robin Kips Albert Pumarola Sebastian Starke Ali Thabet Artsiom Sanakoyeu
Meta AI

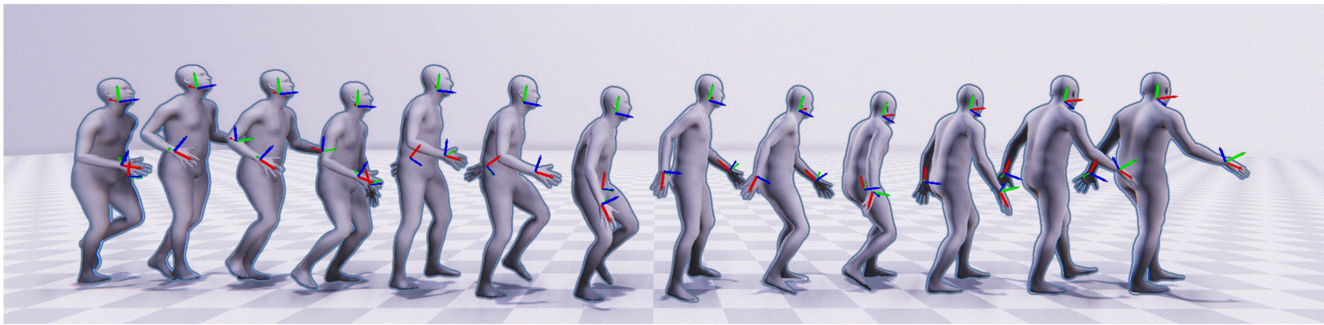

Figure 1. Full body motion synthesis based on HMD and hand controllers input. We show synthesis results of the proposed AGRoL method. RGB axes illustrate the orientation of the head and hands which serves as the input to to our model.

Abstract

With the recent surge in popularity of AR/VR applications, realistic and accurate control of 3D full-body avatars has become a highly demanded feature. A particular challenge is that only a sparse tracking signal is available from standalone HMDs (Head Mounted Devices), often limited to tracking the user's head and wrists. While this signal is resourceful for reconstructing the upper body motion, the lower body is not tracked and must be synthesized from the limited information provided by the upper body joints. In this paper, we present AGRoL, a novel conditional diffusion model specifically designed to track full bodies given sparse upper-body tracking signals. Our model is based on a simple multi-layer perceptron (MLP) architecture and a novel conditioning scheme for motion data. It can predict accurate and smooth full-body motion, particularly the challenging lower body movement. Unlike common diffusion architectures, our compact architecture can run in real-time, making it suitable for online body-tracking applications. We train and evaluate our model on AMASS motion capture dataset, and demonstrate that our approach outperforms state-of-the-art methods in generated motion accuracy and smoothness. We further justify our design choices through extensive experiments and ablation studies.

1. Introduction

Humans are the primary actors in AR/VR applications. As such, being able to track full-body movement is in high demand for these applications. Common approaches are able to accurately track upper bodies only [25, 56]. Moving to full-body tracking unlocks engaging experiences where users can interact with the virtual environment with an increased sense of presence. However, in the typical AR/VR setting there is no strong tracking signal for the entire human body – only the head and hands are usually tracked by means of Inertial Measurement Unit (IMU) sensors embedded in Head Mounted Displays (HMD) and hand controllers. Some works suggest adding additional IMUs to track the lower body joints [22, 25], those additions come at higher costs and the expense of the user's comfort [24, 27]. In an ideal setting, we want to enable high-fidelity full-body tracking using the standard three inputs (head and hands) provided by most HMDs.

Given the position and orientation information of the head and both hands, predicting full-body pose, especially the lower body, is inherently an underconstrained problem. To address this challenge, different methods rely on generative models such as normalizing flows [44] and Variational Autoencoders (VAE) [11] to synthesize lower body motions. In the realm of generative models, diffusion models have recently shown impressive results in image and video generation [21, 39, 46], especially for conditional generation. This inspires us to employ the diffusion model to generate the fully-body poses conditioned on the sparse track-

[*] Work done during an internship at Meta AI.
Code is available at github.com/facebookresearch/AGRoL.

ing signals. To the best of our knowledge, there is no existing work leveraging the diffusion model solely for motion reconstruction from sparse tracking information.

However, it is not trivial to employ the diffusion model in this task. Existing approaches for conditional generation with diffusion models are widely used for cross-modal conditional generation. Unfortunately, these methods can not be directly applied to the task of motion synthesis, given the disparity in data representations, *e.g*. human body joints feature *vs*. images.

In this paper, we propose a novel diffusion architecture – *Avatars Grow Legs* (AGRoL), which is specifically tailored for the task of conditional motion synthesis. Inspired by recent work in future motion prediction [18], which uses an MLP-based architecture, we find that a carefully designed MLP network can achieve comparable performance to the state-of-the-art methods. However, we discovered that the predicted motions of MLP networks may contain jittering artifacts. To address this issue and generate smooth realistic full body motion from sparse tracking signals, we design a novel lightweight diffusion model powered by our MLP architecture. Diffusion models require time step embedding [21, 38] to be injected in the network during training and inference; however, we found that our MLP architecture is not sensitive to the positional embedding in the input. To tackle this problem, we propose a novel strategy to effectively inject the time step embedding during the diffusion process. With the proposed strategy, we can significantly mitigate the jittering issues and further improve the model's performance and robustness against the loss of tracking signal. Our model accurately predicts full-body motions, outperforming state-of-the-art methods as demonstrated by the experiments on AMASS [35], large motion capture dataset.

We summarize our contributions as follows:

- We propose AGRoL, a conditional diffusion model specifically designed for full-body motion synthesis based on sparse IMU tracking signals. AGRoL is a simple and yet efficient MLP-based diffusion model with a lightweight architecture. To enable gradual denoising and produce smooth motion sequences we propose a block-wise injection scheme that adds diffusion timestep embedding before every intermediate block of the neural network. With this timestep embedding strategy, AGRoL achieves state-of-the-art performance on the full-body motion synthesis task without any extra losses that are commonly used in other motion prediction methods.
- We show that our lightweight diffusion-based model AGRoL can generate realistic smooth motions while achieving real-time inference speed, making it suitable for online applications. Moreover, it is more robust against tracking signals loss then existing approaches.

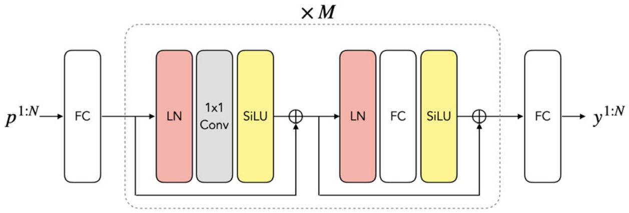

Figure 2. The architecture of our MLP-based network. *FC*, *LN*, and *SiLU* denote the fully connected layer, the layer normalization, and the SiLU activation layer respectively. *1 × 1 Conv* denotes the 1D convolution layer with kernel size 1. Note that *1 × 1 Conv* here is equivalent to a fully connected layer operating on the first dimension of the input tensor $\mathbb{R}^{N\times D}$, while the *FC* layers operate on the last dimension. N denotes the temporal dimension and D denotes the dimension of the latent space. The middle block is repeated M times. The first *FC* layer projects input data to a latent space $\mathbb{R}^{N\times D}$ and the last one converts from latent space to the output space of full-body poses $\mathbb{R}^{N\times S}$.

2. Related Work

2.1. Motion Tracking from Sparse Tracking Inputs

The generation of full-body poses from sparse tracking signals of body joints has become an area of considerable interest within the research community. For instance, recent works such as [22] have demonstrated the ability to track full bodies using only 6 IMU inputs and employing a bi-directional LSTM to predict SMPL body joints. Additionally, in [56], a similar approach is used to track with 4 IMU inputs, specifically the head, wrists, and pelvis. However, in the practical HMD setting, only 3 tracking signals are typically available: the head and 2 wrists. In this context, AvatarPoser [24] provides a solution to the 3-point problem through the use of a transformer-based architecture. Other methods attempt to solve sparse input body tracking as a synthesis problem. To that extent, Aliakbarian *et al*. [4] proposed a flow-based architecture derived from [10], while Dittadi *et al*. [11] opted for a Variational Autoencoder (VAE) method. While more complex methods have been developed that involve Reinforcement Learning, as seen in [55, 57], these approaches may struggle to simultaneously maintain accurate upper-body tracking while generating physically realistic motions.

In summary, all methods presented in this section either require more than three joints input or face difficulties in accurately predicting full body pose, particularly in the lower body region. Our proposed method, on the other hand, utilizes a custom diffusion model and employs a straightforward MLP-based architecture to predict full body pose with a high degree of accuracy, while utilizing only three IMU inputs.

2.2. Diffusion Models and Motion Synthesis

Diffusion models [21, 39, 46] are a class of likelihood-based generative models based on learning progressive noising and denoising of data. Diffusion models have recently have garnered significant attention in the field of image generation [9] due to their ability to significantly outperform popular GAN architectures [7, 26] and is better suited for handling a large amount of data. Furthermore, diffusion models can support conditional generation, as evidenced by the classifier guidance approach presented in [9] and the CLIP-based text conditional synthesis for diffusion models proposed in [38].

More recently, concurrent works have also extended diffusion models to motion synthesis, with particular focus on the text-to-motion task [28, 49, 59]. However, these models are both complex in architecture and require multiple iterations at inference time. This hinders them unsuitable for real-time applications like VR body tracking. We circumvent this problem by designing a custom and efficient diffusion model. To the best of our knowledge, we present the first diffusion model solely purposed for solving motion reconstruction from sparse inputs. Our model leverages a simple MLP architecture, runs in real-time, and provides accurate pose predictions, particularly for lower bodies.

2.3. Human Motion Synthesis

Early works in human motion synthesis rose under the task of future motion prediction. Works around this task saw various modeling approaches ranging from sequence to sequence models [14] to graph modeling of each body part [23]. These supervised models were later replaced by generative methods [17, 31] based on Generative Adversarial Networks (GANs) [16]. Despite their leap forward, these approaches tend to diverge from realistic motion and require access to all body joint positions, making them impractical for avatar animation in VR [19].

A second family of motion synthesis methods revolves around character control. In this setting, character motion must be generated according to user inputs and environmental constraints, such as the virtual environment properties. This research direction has practical applications in the field of computer gaming, where controller input is used to guide character motion. Taking inspiration from these constraints, Wang et al. [54] formulated motion synthesis as a control problem by using a GAN architecture that takes direction and speed input into account. Similar efforts are found in [48], where the method learns fast and dynamic character interactions that involve contacts between the body and other objects, given user input from a controller. These methods are impractical in a VR setting, where users want to drive motion using their real body pose instead of a controller.

3. Method

3.1. Problem Formulation

Our goal is to predict the whole body motion given sparse tracking signals, i.e. the orientation and translation of the headset and two hand controllers. To achieve this, we use a sequence of N observed joint features $p^{1:N} = \{p^i\}_{i=1}^N \in \mathbb{R}^{N \times C}$ and aim to predict the corresponding whole-body poses $y^{1:N} = \{y^i\}_{i=1}^N \in \mathbb{R}^{N \times S}$ for each frame. The dimensions of the input/output joint features are represented by C and S, respectively. We utilize the SMPL [33] model in this paper to represent human poses and follow the approach outlined in [11, 24] to consider the first 22 joints of the SMPL model and disregard the joints on the hands and face. Thus, $y^{1:N}$ reflects the global orientation of the pelvis and the relative rotation of each joint. Following [24], during inference, we initially pose the human model using the predicted rotations. Next, we calculate the character's global translation by accounting for the known head translation and subtracting the offset between the root joint and the head joint.

In the following section, we first introduce a simple MLP-based network for full-body motion synthesis based on sparse tracking signals. Then, we show how we further improve the performance by leveraging the proposed MLP-based architecture to power the conditional generative diffusion model, termed AGRoL.

3.2. MLP-based Network

Our network architecture comprises only four types of components commonly employed in the realm of deep learning: fully connected layers (FC), SiLU activation layers [41], 1D convolutional layers [30] with kernel size 1 and an equal number of input and output channels, as well as layer normalization (LN) [5]. It is worth noting that the 1D convolutional layer with a kernel size of 1 can also be interpreted as a fully connected layer operating along a different dimension. The details of our network architecture are demonstrated in Figure 2. Each block of the MLP network contains one convolutional and one fully connected layer, which is responsible for temporal and spatial information merging respectively. We use skip-connections as in ResNets [20] with Layer Norm [6] as pre-normalization of the layers. First, we project the input data $p^{1:N}$ to a higher dimensional latent space using a linear layer. And the last layer of the network projects from the latent space to the output space of full-body poses $y^{1:N}$.

3.3. Diffusion Model

Diffusion model [21, 46] is a type of generative model which learns to reverse random Gaussian noise added by a Markov chain to recover desired data samples from the noise. In the forward diffusion process, given a sample mo-

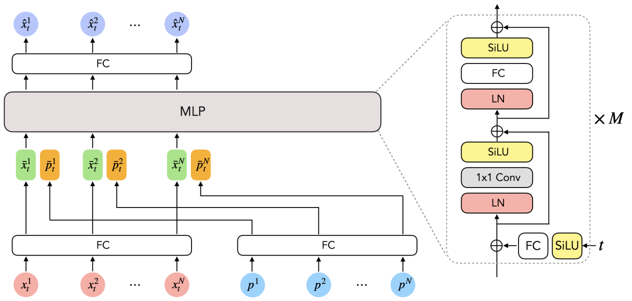

Figure 3. The architecture of our MLP-based diffusion model. t is the noising step. $x_t^{1:N}$ denotes the motion sequence of length N at step t, which is pure Gaussian noises when $t = 0$. $p^{1:N}$ denotes the sparse upper body signals of length N. $\hat{x}_t^{1:N}$ denotes the denoised motion sequence at step t.

tion sequence $x_0^{1:N} \sim q(x_0^{1:N})$ from the data distribution, the Markovian noising process can be written as:

$$q(x_t^{1:N}|x_{t-1}^{1:N}) := \mathcal{N}(x_t^{1:N}; \sqrt{\alpha_t}x_{t-1}^{1:N}, (1-\alpha_t)I), \quad (1)$$

where $\alpha_t \in (0,1)$ is constant hyper-parameter and I is the identity matrix. $x_T^{1:N}$ tends to an isotropic Gaussian distribution when $T \rightarrow \infty$. Then, in the reverse diffusion process, a model p_θ with parameters θ is trained to generate samples from input Gaussian noise $x_T \sim \mathcal{N}(0, I)$ with variance σ_t^2 that follows a fixed schedule. Formally,

$$p_\theta(x_{t-1}^{1:N}|x_t^{1:N}) := \mathcal{N}(x_{t-1}^{1:N}; \mu_\theta(x_t, t), \sigma_t^2 I), \quad (2)$$

where μ_θ could be reformulated [21] as

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t, t)), \quad (3)$$

where $\bar{\alpha}_t = \alpha_1 \cdot \alpha_2 ... \cdot \alpha_t$. So the model has to learn to predict noise $\epsilon_\theta(x_t, t)$ from x_t and timestep t.

In our case, we want to use the diffusion model to generate sequences of full-body poses conditioned on the sparse tracking of joint features $p^{1:N}$. Thus, the reverse diffusion process becomes conditional: $p_\theta(x_{t-1}^{1:N}|x_t^{1:N}, p^{1:N})$. Moreover, we follow [42] to directly predict the clean body poses $x_0^{1:N}$ instead of predicting the residual noise $\epsilon_\theta(x_t, t)$. The objective function is then formulated as

$$\mathcal{L}_{dm} = \mathbb{E}_{x_0^{1:N} \sim q(x_0^{1:N}), t}\left[\| x_0^{1:N} - \hat{x}_0^{1:N} \|_2^2\right] \quad (4)$$

where the $\hat{x}_0^{1:N} = f_\theta(x^{1:N}, p^{1:N}, t)$ denotes the output of our model f_θ.

We use the MLP architecture proposed in Sect. 3.2 as the backbone for the model f_θ that predicts the full-body poses. At time step t, the motion features $x_t^{1:N}$ and the observed joints feature $p^{1:N}$ are first passed separately through a fully connected layer to obtain the latent features $\bar{x}_t^{1:N}$ and $\bar{p}^{1:N}$:

$$\bar{x}_t^{1:N} = \text{FC}_0(x_t^{1:N}), \quad (5)$$

$$\bar{p}^{1:N} = \text{FC}_1(p^{1:N}). \quad (6)$$

Then these features are concatenated together and fed to the MLP backbone: $\hat{x}_0^{1:N} = \text{MLP}(\text{Concat}(\bar{x}_t^{1:N}, \bar{p}^{1:N}), t)$.

Block-wise Timestep Embedding. When utilizing diffusion models, the embedding of the timestep t is often included as an additional input to the network. To achieve this, a common approach is to concatenate the timestep embedding with the input, similar to positional embedding used in transformer-based methods [12,53]. However, since our network is mainly composed of FC layers, that mix the input features indiscriminately [50], the time step embedding information can easily be lost after several layers, which hinders learning the denoising process and results in predicted motions with severe jittering artifacts, as shown in Section 4.4.2. In order to address the issue of losing time step embedding information in our network, we introduce a novel strategy that repetitively injects the time step embedding into every block of the MLP network. This process involves projecting the timestep embedding to match the input feature dimensions through a fully connected layer and a SiLU activation layer. The details of our pipeline are shown in Figure 3. Unlike previous work, such as [21], which predicts a scale and shift factor for each block from the timestep embedding,our proposed approach directly adds the timestep embedding projections to the input activations of each block. Our experiments in Sect. 4 validate that this approach significantly reduces jittering issues and enables the synthesis of smooth motions.

4. Experiments

Our models are trained and evaluated on the AMASS dataset [35]. To compare with previous methods, we use two different settings for training and testing. In the first setting, we follow the approach of [24], which utilizes three subsets of AMASS: CMU [8], BMLr [51], and HDM05 [37]. In the second setting, we adopt the data split employed in several recent works, including [4,11,43]. This approach employs a larger set of training data, including CMU [8], MPI Limits [3], Total Capture [52], Eyes Japn [13], KIT [36], BioMotionLab [51], BMLMovi [15], EKUT [36], ACCAD [1], MPI Mosh [32], SFU [2], and HDM05 [37] as training data, while HumanEval [45] and Transition [35] serve as testing data.

In both settings, we adopt the SMPL [33] human model for the human pose representation and train our model to predict the global orientation of the root joint and relative rotation of the other joints.

Method	MPJRE	MPJPE	MPJVE	Hand PE	Upper PE	Lower PE	Root PE	Jitter	Upper Jitter	Lower Jitter
Final IK	16.77	18.09	59.24	-	-	-	-	-	-	-
LoBSTr	10.69	9.02	44.97	-	-	-	-	-	-	-
VAE-HMD	4.11	6.83	37.99	-	-	-	-	-	-	-
AvatarPoser*	3.08	4.18	27.70	2.12	1.81	7.59	**3.34**	14.49	7.36	24.81
MLP (Ours)	2.69	3.93	22.85	2.62	1.89	6.88	3.35	13.01	9.13	18.61
AGRoL (Ours)	**2.66**	**3.71**	**18.59**	**1.31**	**1.55**	**6.84**	3.36	**7.26**	**5.88**	**9.27**
GT	0	0	0	0	0	0	0	4.00	3.65	4.52

Table 1. Comparison of our approach with state-of-the-art methods on a subset of AMASS dataset following [24]. We report *MPJPE* [cm], *MPJRE* [deg], *MPJVE* [cm/s], Jitter [10^2m/s^3] metrics. AGRoL achieves the best performance on *MPJPE*, *MPJRE* and *MPJVE*, and outperforms other models, especially on the *Lower PE* (Lower body Position Error) and *Jitter* metrics, which shows that our model generates accurate lower body movement and smooth motions.

Method	MPJRE	MPJPE	MPJVE	Jitter
VAE-HMD† [11]	-	7.45	-	-
HUMOR† [43]	-	5.50	-	-
FLAG† [4]	-	4.96	-	-
AvatarPoser*	4.70	6.38	34.05	10.21
MLP (Ours)	4.33	6.66	33.58	21.74
AGRoL (Ours)	**4.30**	**6.17**	**24.40**	**8.32**
GT	0	0	0	2.93

Table 2. Comparison of our approach with state-of-the-art methods on AMASS dataset following the protocol of [4, 11, 43]. We report the MPJPE [cm], MPJRE [deg], MPJVE [cm/s], and Jitter [10^2m/s^3] metrics. The * denotes that we retrained the AvatarPoser using public code. † denotes methods that use pelvis location and rotation during inference, which are not directly comparable to our method, as we assume that the pelvis information is not available during the training and the testing. The best results are in bold, and the second-best results are underlined.

4.1. Implementation Details

We represent the joint rotations by the 6D reparametrization [60] due to its simplicity and continuity. Thus, for the sequences of body poses $y^{1:N} \in \mathbb{R}^{N\times S}$, $S = 22 \times 6$. The observed joint features $p^{1:N} \in \mathbb{R}^{N\times C}$ consists of the orientation, translation, orientation velocity and translation velocity of the head and hands in global coordinate system. Additionally, we adopt 6D reparametrization for the orientation and orientation velocity, thus $C = 18 \times 3$. Unless otherwise stated, we set the frame number N to 196.

MLP Network We build our MLP network using 12 blocks ($M = 12$). All latent features in the MLP network have the same shape of $N \times 512$. The network is trained with batch size 256 and Adam optimizer [29]. The learning rate is set to 3e-4 at the beginning and drops to 1e-5 after 200000 iterations. The weight decay is set to 1e-4 for the entire training. During inference, we apply our model in an auto-regressive manner for the longer sequences.

MLP-based Diffusion Model (AGRoL) We keep the MLP network architecture unchanged in the diffusion model. To inject the time step embedding used in the diffusion process in the network, in each MLP block, we pass the time step embedding to a fully connected layer and a SiLU activation layer [41] and sum it with the input feature. The network is trained with exactly the same hyperparameters as the MLP network, with the exception of using the AdamW [34] as optimizer. During training, we set the sampling step to 1000 and employ a cosine noise schedule [39]. However, to expedite the inference speed, we leverage the DDIM [47] technique, which allows us to sample only 5 steps instead of 1000 during inference.

All experiments were carried out on a single NVIDIA V100 graphics card, using the PyTorch framework [40].

4.2. Evaluation Metrics

In line with previous works [11,24,43,58], we adopt nine evaluation metrics that we group into three categories.

Rotation-related metric: Mean Per Joint Rotation Error [degrees] (*MPJRE*) measures the average relative rotation error for all joints.

Velocity-related metrics: These include Mean Per Joint Velocity Error [cm/s] (*MPJVE*) and *Jitter*. *MPJVE* measures the average velocity error for all joints, while *Jitter* [58] evaluates the mean jerk (change in acceleration over time) of all body joints in global space, expressed in 10^2m/s^3. *Jitter* is an indicator of motion smoothness.

Position-related metrics. Mean Per Joint Position Error [cm] (*MPJPE*) quantifies the average position error across all joints. *Root PE* assesses the position error of the root joint, whereas *Hand PE* calculates the average position error for both hands. *Upper PE* and *Lower PE* estimate the average position error for joints in the upper and lower body, respectively.

4.3. Evaluation Results

We evaluate our method on the AMASS dataset with two different protocols. As shown in Table 1 and Table 2,

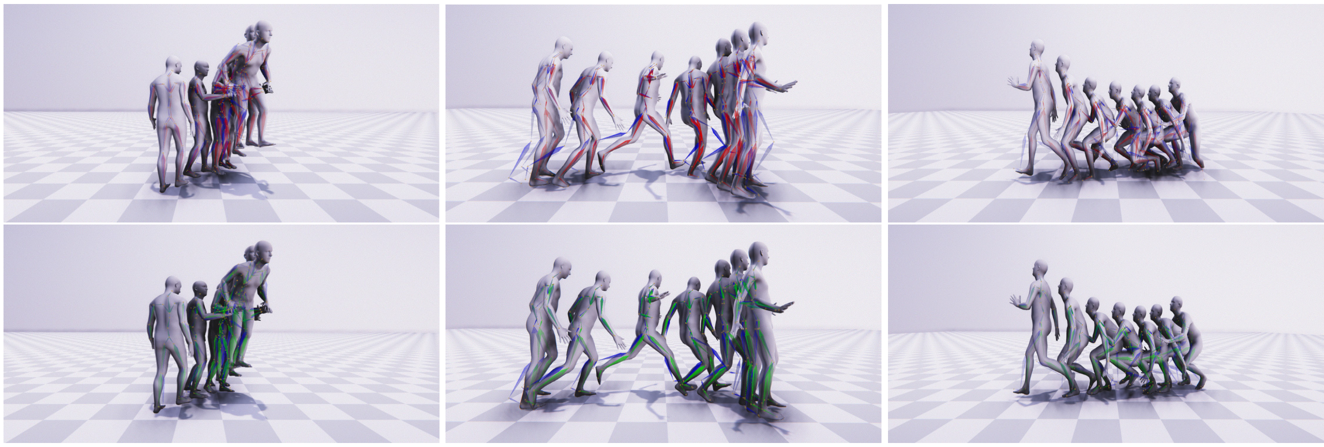

Figure 4. Qualitative comparison between AGRoL (top) and AvatarPoser [24] (bottom) on test sequences from AMASS dataset. We visualize the predicted skeletons and render human body meshes. **Top:** AvatarPoser predictions in red. **Bottom:** AGRoL predictions in green. In both rows, the blue skeletons denote the ground truth motion. We observe that motions predicted by AGRoL are closer to ground truth compared to the predictions of AvatarPoser.

Method	#Params	MPJRE	MPJPE	MPJVE	Hand PE	Upper PE	Lower PE	Root PE	Jitter
AGRoL-AvatarPoser	2.89M	4.31	6.71	27.65	1.47	2.56	12.69	6.69	9.57
AGRoL-AvatarPoser-Large	7.63M	2.86	4.04	21.90	**1.29**	1.62	7.53	3.64	9.94
AGRoL-Transformer	7.03M	3.01	4.41	20.33	2.97	2.13	7.71	3.88	**6.45**
AGRoL (Ours)	7.48M	**2.66**	**3.71**	**18.59**	1.31	**1.55**	**6.84**	**3.36**	7.26

Table 3. Ablation study of network architectures in our diffusion model. We replace the proposed MLP backbone with other architectures and train several versions of the diffusion model with the same hyperparameters. The *AvatarPoser-Large* denotes the backbone with the same architecture as AvatarPoser [24] but with more transformer layers. *AGRoL-Transformer* is the AGRoL version with the transformer backbone from [49]. The *AGRol (ours)* with our MLP backbone outperforms all other backbones on most of the metrics.

our MLP network can already surpass most of the previous methods and achieves comparable results with the state-of-the-art method [24], demonstrating the effectiveness of the proposed simple MLP architecture. By leveraging the diffusion process and the proposed MLP backbone, the AGRoL model remarkably boosts the performance of the MLP network, surpassing all previous methods in all metrics[*]. Moreover, our proposed AGRoL model significantly improves the smoothness of the generated motion, as reflected by the reduced *Jitter* error compared to other methods. We visualize some examples in Figure 4 and Figure 5. In Figure 4 we show the comparison of the reconstruction error between AGRoL and AvatarPoser. In Figure 5, by visualizing the pose trajectories, we demonstrate the comparison of the smoothness and foot contact quality between AGRoL and AvatarPoser.[†]

4.4. Ablation Studies

In this section, we ablate our methods on AMASS dataset. We first compare our proposed MLP architecture with other backbones in the context of the diffusion model in Section 4.4.1 to highlight the superiority of our MLP network. Then we investigate the importance of time step embedding for our diffusion model and evaluate different strategies for adding the time step embedding in Section 4.4.2. Finally, we analyze the impact of the number of sampling steps used during inference in Section 4.4.3.

4.4.1 Architecture

To validate the effectiveness of our proposed MLP backbone in the diffusion model setup, we conduct experiments where we replace our MLP network in AGRoL with other types of backbones and compare them. Specifically, we consider two alternative backbone architectures: the network from AvatarPoser [24] and the transformer network from Tevet et al. [49]. In transformer networks, instead of repetitively injecting the time positional embedding to every block, we concatenate the time positional embedding with the input features $\bar{x}^{1:N}$ and $\bar{p}^{1:N}$ before being fed to transformer layers. We apply the same technique to the AvatarPoser backbone as this model is also based on transformer layers. To ensure a fair comparison, we train

[*] Except for insignificant 0.2 mm difference in Root PE in Tab. 1.

[†] Please visit https://dulucas.github.io/agrol for more visual examples.

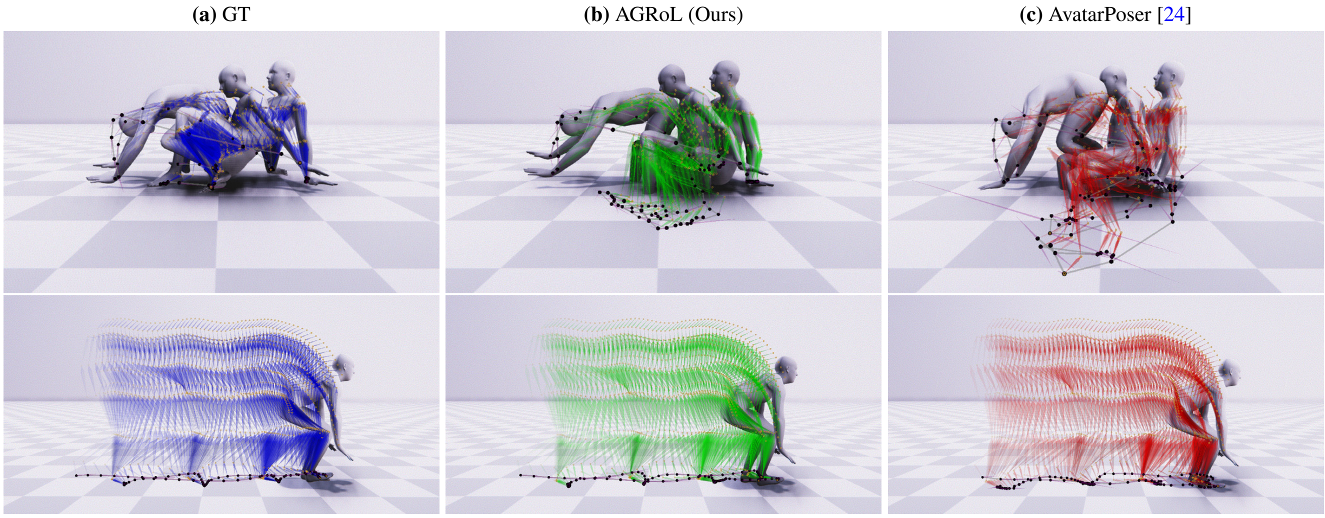

Figure 5. Motion trajectory visualization for predicted motions. **(a)** The ground truth motion with blue skeletons; **(b)** motion predicted by AGRoL with green skeletons; **(c)** motion predicted by AvatarPoser with red skeletons. The purple vectors denote the velocity vectors of the corresponding joints. Observing the motion trajectories, we can see jittering and foot sliding issues more clearly. Smooth motion typically exhibits regular pose trajectories with the velocity vector of each joint changing steadily. The density of joint trajectories varies with walking speed; trajectories become denser as the individual slows down. Therefore, in the absence of foot sliding, we should observe a significantly high density of points when a foot makes contact with the ground. The black dots in the bottom row represent the trajectories of the foot joints. We notice more pronounced spikes in the density of foot trajectories for AGRoL compared to AvatarPoser.

AGRoL with two versions of AvatarPoser backbone. The first one, AGRoL-AvatarPoser, uses the architecture that follows exactly the same settings as described in the original paper [24], while the second one, AGRoL-AvatarPoser-Large, incorporates additional transformer layers to achieve a comparable size to our AGRoL model with MLP backbone. Similarly, we increase the number of layers in the transformer backbone [49] and train AGRoL-Transformer. As shown in Tab. 3, the AGRoL diffusion model with the proposed MLP backbone achieves superior results compared to the versions with other backbones.

4.4.2 Diffusion Time Step Embedding

In this section, we study the importance of time step embedding. Time step embedding is often used in diffusion-based models [9, 59] to indicate the noise level t during the diffusion process. We use the sinusoidal positional embedding [53] as the time step embedding. Although the AGRoL without time step embedding (see Tab. 4) can still attain reasonable performance on metrics related to position errors and rotation errors, the performance on metrics related to velocity errors (*MPJVE* and *Jitter*) is severely degraded. This outcome is expected as the absence of the time step embedding implies that the model is not aware of the current denoising step, rendering it unable to denoise accurately.

We now ablate three strategies for utilizing the time step embedding in our network: *Add*, *Concat*, and *RepIn*. *RepIn* (Repetitive Injection) repetitively passes the time step embedding through a linear layer and injects the results into every block of the MLP network. In contrast, *Add* and *Concat* inject the time step embedding only once at the beginning of the network.

Before inputting it into the network, the time step embedding is passed through a fully connected layer and a SiLU activation to obtain a latent feature $u_t \in \mathbb{R}^{1\times D}$. *Add* sums the u_t with the input features $\bar{x}_t^{1:N}$ and $\bar{p}^{1:N}$, the output of the network then becomes $\hat{x}_0^{1:N} = \text{MLP}(\text{Concat}(\bar{x}_t^{1:N}, \bar{p}^{1:N}) + u_t)$, where vector u_t is broadcasted along the first dimension. *Concat* concatenates the u_t with the input features $\bar{x}_t^{1:N}$ and $\bar{p}^{1:N}$, resulting in $\hat{x}_0^{1:N} = \text{MLP}(\text{Concat}(\bar{x}_t^{1:N}, \bar{p}^{1:N}, u_t))$.

RepIn is our proposed strategy for injecting the time step embedding. Specifically, for each block of the MLP network, we project the time step embedding separately using a fully connected layer and a SiLU activation, then we add the obtained features $u_{t,j}$ to the input features of the correspondent block, where $j \in [0, ..M]$ and M is the number of blocks. As shown in Table 4, our proposed strategy can largely improve the velocity-related metrics and alleviate the jittering issues and generate smooth motion.

4.4.3 Number of Sampling Steps during Inference

We ablate the number of sampling steps that we used during inference. In Tab. 5, we take the AGRoL model trained

Method	MPJRE	MPJPE	MPJVE	Hand PE	Upper PE	Lower PE	Root PE	Jitter
w/o Time	2.68	**3.63**	22.80	1.36	**1.54**	**6.67**	**3.25**	15.23
Add	2.80	4.01	23.60	1.40	1.64	7.44	3.59	15.02
Concat	2.72	3.79	21.99	1.31	1.57	7.00	3.43	13.30
RepIn (Ours)	**2.66**	3.71	**18.59**	**1.31**	1.55	6.84	3.36	**7.26**

Table 4. Ablation of the time step embedding. *w/o Time* denotes the results of AGRoL without time step embedding. *Add* sums up the features from time step embedding with the input features. *Concat* concatenates the features from time step embedding with the input features. In *Add* and *Concat*, the time step embedding is only fed once at the top of the network. *RepIn* (Repetitive Injection) denotes our strategy to inject the time step embedding into every block of the network. The time step embedding mainly affects the *MPJVE* and *Jitter* metrics. Omiting the timestep embedding or adding it improperly results in high MPJVE and causes severe jittering issues.

# Sampling Steps	MPJRE	MPJPE	MPJVE	Hand PE	Upper PE	Lower PE	Root PE	Jitter
2	3.17	4.93	20.03	2.19	2.12	8.98	4.61	**6.90**
5	**2.66**	3.71	**18.59**	**1.31**	**1.55**	6.84	3.36	7.26
10	2.68	**3.69**	19.55	1.39	**1.55**	**6.77**	**3.31**	7.51
100	2.84	3.93	23.50	1.62	1.67	7.19	3.51	9.64
1000	2.97	4.14	27.25	1.82	1.78	7.55	3.66	12.79

Table 5. Ablation of the number of DDIM [47] sampling steps during inference. The input and output length is fixed to $N = 196$.

Methods	MPJRE	MPJPE	MPJVE	Hand PE	Upper PE	Lower PE	Root PE	Jitter
AvatarPoser	5.69	10.34	572.58	8.98	5.49	17.34	8.83	762.79
MLP	5.37	10.76	107.82	12.43	6.48	16.94	8.74	92.51
Transformer	4.44	8.62	135.99	7.29	5.28	13.44	10.32	147.09
AGRoL (Ours)	**4.20**	**6.38**	**96.85**	**5.27**	**3.86**	**10.03**	**6.67**	**33.35**

Table 6. Robustness of the models to joints tracking loss. We evaluate the methods by randomly masking a portion (10%) of input frames during the inference on AMASS dataset. We test each method 5 times and take the average results. AGRoL achieves the best performance among all the methods, which shows the robustness of our method against joint tracking loss.

with 1000 sampling steps and test it with a subset of diffusion steps during inference. We opted to use 5 DDIM [47] sampling steps as it enabled our model to achieve superior performance on most of the metrics while also being faster.

4.5. Robustness to Tracking Loss

In this section, we study the robustness of our model against input joint tracking loss. In VR applications, it is common for the joint tracking signal to be lost on some frames when hands or controllers move out of the field of view, causing temporal discontinuity in the input signals. We evaluate the performance of all available methods on tracking loss by randomly masking 10% of input frames during inference, and present results in Tab. 6. We observe that the performance of all previous methods is significantly degraded, indicating their lack of robustness against tracking loss. In comparison, AGRoL shows less degradation in accuracy, suggesting that our approach can accurately model motion even with highly sparse tracking inputs.

4.6. Inference Speed

Our AGRoL model achieves real-time inference speed due to a lightweight architecture combined with DDIM sampling. A single AGRoL generation, that runs 5 DDIM sampling steps, produces 196 output frames in 35 ms on a single NVIDIA V100 GPU. Our predictive MLP model takes 196 frames as input and predicts a final result of 196 frames in a single forward pass. It is even faster and requires only 6 ms on a single NVIDIA V100 GPU.

5. Conclusion and Limitations

In this paper, we presented a simple yet efficient MLP-based architecture with carefully designed building blocks which achieves competitive performance on the full-body motion synthesis task. Then we introduced AGRoL, a conditional diffusion model for full-body motion synthesis based on sparse tracking signal. AGRoL leverages a simple yet efficient conditioning scheme for structured human motion data. We demonstrated that our lightweight diffusion-based model generates realistic and smooth human motions while achieving real-time inference speed, making it suitable for online AR/VR applications. A notable limitation of our and related approaches is occasional floor penetration artifacts. Future work involves investigating this issue and integrating additional physical constraints into the model.

6. Acknowledgements

We thank Federica Bogo, Jonas Kohler, Wen Guo, Xi Shen for inspiring discussions and valuable feedback.

References

[1] Osu accad. https://accad.osu.edu/research/motion-lab/system-data.

[2] Sfu motion capture database. https://mocap.cs.sfu.ca/.

[3] Ijaz Akhter and Michael J Black. Pose-conditioned joint angle limits for 3d human pose reconstruction. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1446–1455, 2015.

[4] Sadegh Aliakbarian, Pashmina Cameron, Federica Bogo, Andrew Fitzgibbon, and Thomas J Cashman. Flag: Flow-based 3d avatar generation from sparse observations. In *CVPR*, pages 13253–13262, 2022.

[5] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[6] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[7] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

[8] Carnegie Mellon University. CMU MoCap Dataset.

[9] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *NeurIPS*, 34:8780–8794, 2021.

[10] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *ICLR*, 2016.

[11] Andrea Dittadi, Sebastian Dziadzio, Darren Cosker, Ben Lundell, Thomas J Cashman, and Jamie Shotton. Full-body motion from a single head-mounted device: Generating smpl poses from partial observations. In *ICCV*, pages 11687–11697, 2021.

[12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[13] Eyes, JAPAN Co. Ltd. Eyes, Jappan.

[14] Katerina Fragkiadaki, Sergey Levine, Panna Felsen, and Jitendra Malik. Recurrent network models for human dynamics. In *ICCV*, pages 4346–4354, 2015.

[15] Saeed Ghorbani, Kimia Mahdaviani, Anne Thaler, Konrad Kording, Douglas James Cook, Gunnar Blohm, and Nikolaus F Troje. Movi: A large multipurpose motion and video dataset. *arXiv preprint arXiv:2003.01888*, 2020.

[16] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[17] Liang-Yan Gui, Yu-Xiong Wang, Xiaodan Liang, and José MF Moura. Adversarial geometry-aware human motion prediction. In *ECCV*, pages 786–803, 2018.

[18] Wen Guo, Yuming Du, Xi Shen, Vincent Lepetit, Alameda-Pineda Xavier, and Moreno-Noguer Francesc. Back to mlp: A simple baseline for human motion prediction. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2023.

[19] Ikhsanul Habibie, Daniel Holden, Jonathan Schwarz, Joe Yearsley, and Taku Komura. A recurrent variational autoencoder for human motion synthesis. In *British Machine Vision Conference*, 2017.

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[21] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *NeurIPS*, 33:6840–6851, 2020.

[22] Yinghao Huang, Manuel Kaufmann, Emre Aksan, Michael J Black, Otmar Hilliges, and Gerard Pons-Moll. Deep inertial poser: Learning to reconstruct human pose from sparse inertial measurements in real time. *ACM TOG*, 37(6):1–15, 2018.

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

[24] Jiaxi Jiang, Paul Streli, Huajian Qiu, Andreas Fender, Larissa Laich, Patrick Snape, and Christian Holz. Avatarposer: Articulated full-body pose tracking from sparse motion sensing. *ECCV*, 2022.

[25] Yifeng Jiang, Yuting Ye, Deepak Gopinath, Jungdam Won, Alexander W Winkler, and C Karen Liu. Transformer inertial poser: Attention-based real-time human motion reconstruction from sparse imus. *arXiv preprint arXiv:2203.15720*, 2022.

[26] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, pages 4401–4410, 2019.

[27] Manuel Kaufmann, Yi Zhao, Chengcheng Tang, Lingling Tao, Christopher Twigg, Jie Song, Robert Wang, and Otmar Hilliges. Em-pose: 3d human pose estimation from sparse electromagnetic trackers. In *ICCV*, pages 11510–11520, 2021.

[28] Jihoon Kim, Jiseob Kim, and Sungjoon Choi. Flame: Free-form language-based motion synthesis & editing. *arXiv preprint arXiv:2209.00349*, 2022.

[29] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[30] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[31] Chen Li, Zhen Zhang, Wee Sun Lee, and Gim Hee Lee. Convolutional sequence to sequence model for human dynamics. In *CVPR*, pages 5226–5234, 2018.

[32] Matthew Loper, Naureen Mahmood, and Michael J Black. Mosh: Motion and shape capture from sparse markers. *ACM Transactions on Graphics (ToG)*, 33(6):1–13, 2014.

[33] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. Smpl: A skinned multi-person linear model. *ACM TOG*, 34(6):1–16, 2015.

[34] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[35] Naureen Mahmood, Nima Ghorbani, Nikolaus F. Troje, Gerard Pons-Moll, and Michael J. Black. AMASS: Archive of motion capture as surface shapes. In *ICCV*, pages 5442–5451, Oct. 2019.

[36] Christian Mandery, Ömer Terlemez, Martin Do, Nikolaus Vahrenkamp, and Tamim Asfour. The kit whole-body human motion database. In *2015 International Conference on Advanced Robotics (ICAR)*, pages 329–336. IEEE, 2015.

[37] 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.

[38] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021.

[39] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pages 8162–8171. PMLR, 2021.

[40] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[41] Prajit Ramachandran, Barret Zoph, and Quoc V Le. Swish: a self-gated activation function. *arXiv preprint arXiv:1710.05941*, 7(1):5, 2017.

[42] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

[43] Davis Rempe, Tolga Birdal, Aaron Hertzmann, Jimei Yang, Srinath Sridhar, and Leonidas J Guibas. Humor: 3d human motion model for robust pose estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11488–11499, 2021.

[44] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR, 2015.

[45] Leonid Sigal, Alexandru O Balan, and Michael J Black. Humaneva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion. *International journal of computer vision*, 87(1):4–27, 2010.

[46] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015.

[47] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[48] Sebastian Starke, Yiwei Zhao, Taku Komura, and Kazi Zaman. Local motion phases for learning multi-contact character movements. *ACM TOG*, 39(4):54–1, 2020.

[49] Guy Tevet, Sigal Raab, Brian Gordon, Yonatan Shafir, Amit H Bermano, and Daniel Cohen-Or. Human motion diffusion model. *arXiv preprint arXiv:2209.14916*, 2022.

[50] Ilya O Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, et al. Mlp-mixer: An all-mlp architecture for vision. *Advances in neural information processing systems*, 34:24261–24272, 2021.

[51] Nikolaus F. Troje. Decomposing biological motion: A framework for analysis and synthesis of human gait patterns. *Journal of Vision*, 2(5):2–2, Sept. 2002.

[52] 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 28th British Machine Vision Conference*, pages 1–13, 2017.

[53] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[54] Zhiyong Wang, Jinxiang Chai, and Shihong 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, 2019.

[55] Alexander Winkler, Jungdam Won, and Yuting Ye. Questsim: Human motion tracking from sparse sensors with simulated avatars. *ACM TOG*, 2022.

[56] Dongseok Yang, Doyeon Kim, and Sung-Hee Lee. Lobstr: Real-time lower-body pose prediction from sparse upper-body tracking signals. In *Comput. Graph. Forum*, volume 40, pages 265–275. Wiley Online Library, 2021.

[57] Yongjing Ye, Libin Liu, Lei Hu, and Shihong Xia. Neural3Points: Learning to Generate Physically Realistic Full-body Motion for Virtual Reality Users. 2022.

[58] Xinyu Yi, Yuxiao Zhou, Marc Habermann, Soshi Shimada, Vladislav Golyanik, Christian Theobalt, and Feng Xu. Physical inertial poser (pip): Physics-aware real-time human motion tracking from sparse inertial sensors. In *CVPR*, pages 13167–13178, 2022.

[59] Mingyuan Zhang, Zhongang Cai, Liang Pan, Fangzhou Hong, Xinying Guo, Lei Yang, and Ziwei Liu. Motiondiffuse: Text-driven human motion generation with diffusion model. *arXiv preprint arXiv:2208.15001*, 2022.

[60] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *CVPR*, pages 5745–5753, 2019.