

Human Geometry Distribution for 3D Animation Generation

Supplementary Material

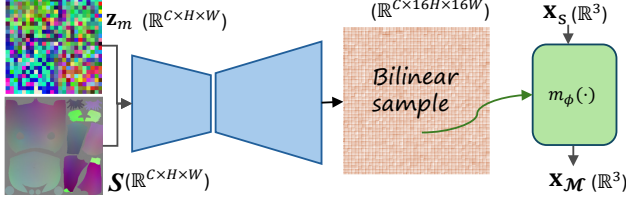


Figure 6. The illustration of the supervised model.

9. Implementation Details

9.1. Low-cost Mapping Construction

We adopt a similar structure as $u_\theta(\mathbf{x}_t \mid t, \mathbf{x}_S, \mathcal{S}, \mathbf{z})$ to train m_ϕ . Instead of using a diffusion model, we train m_ϕ as a supervised model for efficiency. As shown in Fig. 6, we replace the input of u_θ with \mathbf{x}_S , and output the point \mathbf{x}_M on the target geometry directly. Accordingly, we discard the condition branch of \mathbf{x}_S and t .

After training m_ϕ , we construct \mathcal{T} via three steps.

Step 1. Coarse mapping. We first uniformly sample millions of points from both SMPL mesh $\mathbf{x}_S \sim \mathcal{S}$ and 3D human geometry $\mathbf{x}_M \sim \mathcal{M}$, and use the trained model to obtain a coarse prediction $m_\phi(\mathbf{x}_S)$ for each \mathbf{x}_S .

Step 2. Initial transitions. For each \mathbf{x}_M , we find its nearest $m_\phi(\mathbf{x}_S)$ using KNN to determine its corresponding \mathbf{x}_S , yields the initial transitions:

$$\mathcal{T}_0 = \{(\mathbf{x}_S, \mathbf{x}_M) \mid \mathbf{x}_S = \arg \min_{\mathbf{x}'_S} \|\mathbf{x}_M - m_\phi(\mathbf{x}'_S)\|\} \quad (10)$$

However, since not all $\mathbf{x}_S \in \mathcal{S}$ are selected as nearest neighbors, some SMPL mesh regions remain sparsely covered or entirely unsampled. This incomplete coverage leads to under-trained regions on the SMPL mesh and inaccurate estimation during inference.

Step 3. Coverage refinement. To address this issue, we first identify triangle faces on the SMPL mesh that lack any sampled points from \mathcal{T}_0 . Let $\mathcal{F}_{\text{covered}}$ denote faces that contain at least one \mathbf{x}_S in \mathcal{T}_0 , and $\mathcal{F}_{\text{uncovered}} = \mathcal{F} \setminus \mathcal{F}_{\text{covered}}$ the remaining faces. We uniformly sample new points on $\mathcal{F}_{\text{uncovered}}$, estimate their target locations $m_\phi(\mathbf{x}_S)$, and find their nearest \mathbf{x}_M on the human surface. These additional transitions, \mathcal{T}_{add} , are merged with \mathcal{T}_0 to produce the final set $\mathcal{T} = \mathcal{T}_0 \cup \mathcal{T}_{\text{add}}$, ensuring better coverage and a more uniform transitions.

9.2. Training Configurations

During training of the latent space, we adopt a network architecture similar to HuGeoDis [40]. Specifically, we sample 2^{18} training transitions for each geometry, and the latent

feature \mathbf{z} is represented as $\mathbb{R}^{6 \times 24 \times 24}$. The model employs both linear and convolutional layers from EDM2 [16], with 256 channels used in all layers.

For the generative animation model \mathcal{G} , we employ a U-Net architecture following EDM2 [16]. The network performs three downsampling operations, with channel dimensions of 384, 768, and 1536 for each respective stage. Attention layers are applied at every resolution level. The conditioning input of animation model is provided by a convolutional network that takes the latent feature \mathbf{z} as input. Specifically, $\phi(\cdot)$ has a channel dimension of 32 and spatial size 24×24 , which is progressively downsampled through three convolutional layers to a resolution of 3×3 , yielding $\phi(\mathbf{z}) \in \mathbb{R}^{32 \times 3 \times 3}$.

For optimization, we use AdamW [22] with a learning rate of 8×10^{-3} , and apply a cosine learning rate scheduler across all models. A small L2 regularization of 10^{-6} is applied to the latent features. We observe that stronger regularization tends to make the network rely excessively on the SMPL condition, leading to potential overfitting to SMPL-driven patterns.

The latent model is trained for approximately five days using four NVIDIA A100 GPUs, while the generative animation model is trained for two days on the same hardware configuration. The batch size is set to 16 per GPU.

On an A100, our animation model takes 2.6s/frame (5,704MB peak memory), and the latent diffusion model takes 10.3s (3,511MB) for 500k-points. Costs scale linearly with frame count.

9.2.1. Dataset Split

The Dress4D dataset consists of various digital human avatars, each performing multiple distinct animation sequences (e.g., walking, dancing, stretching). For each unique avatar-garment combination, we utilize the majority of the available animation sequences to train the model. To evaluate the model’s temporal consistency and generalization to unseen motions, we withhold one full animation sequence per avatar exclusively for testing. This split ensures that while the model may be familiar with the avatar’s geometry and the specific garment’s material properties, it has never encountered the specific motion trajectories or poses present in the test sequence.

9.3. First Frame Initialization

To initialize the animation generation process, our method requires an initial latent vector \mathbf{z}^0 , which can be sourced through flexible strategies. For instance, \mathbf{z}^0 can be synthesized via a pre-trained 3D Human Generation network

(Sec. 5.2), whose strong performance based on our representation is demonstrated in Sec. 5.2. For the animation-related experiments, we specifically employ auto-decoder optimization to obtain \mathbf{z}^0 to ensure the highest fidelity.

9.4. Dino-based Identity Classifier

To evaluate identity similarity, we train an identity classifier based on the DINO network. Given a normal image as input, the DINO encoder converts it into a sequence of tokens, consisting of one [CLS] token and multiple image feature tokens. These tokens are then processed by 6 Transformer blocks, followed by an MLP head that maps the [CLS] token into a 64-dimensional embedding. The network is trained with a contrastive learning objective using the NT-Xent loss, which minimizes the distance between embeddings of the same identity and maximizes the distance between embeddings of different identities.

9.5. Long-term Supervision Model

We build a non-diffusion baseline following an auto-regressive framework commonly used in the motion domain [39] for long-term supervision. Unlike the diffusion-based model, which learns denoising at each step, this supervised model learns to predict clean latents directly, enabling efficient long-range generation. Specifically, the model directly predicts the next latent state \mathbf{z}^{s+1} conditioned on the known context $\{\mathbf{z}^{s-i:s}, \mathcal{S}^{s-i:s+1}, c\}$, rather than predicting noise or score.

During training, the model auto-regressively generates a sequence of $n = 8$ animation steps, and supervision is applied over all generated frames:

$$\mathcal{L}_{\text{sup}} = \sum_{s=1}^{n=8} \|\hat{\mathbf{z}}^s - \mathbf{z}^s\|_2^2. \quad (11)$$

This design allows direct long-horizon supervision and helps mitigate the accumulation of single-step prediction errors when sufficient training data is available.

10. Additional Results

We provide additional results, including the comparison with the “w/o augment” (Fig. 8), “w/o condition” (Fig. 9, Fig. 10), and “long-term” (Fig. 11, Fig 12) models, covering different characters and motion types to validate the generalization ability of our method.

10.1. User Study Details

Our user study involved 14 researchers (unpaid) specialized in computer graphics and vision. Each participant blindly rated 12 random animations on a Likert scale (1-5) across three metrics: Dynamic Quality, Naturalness, and Garment-Motion Conformance. We report the user study score with

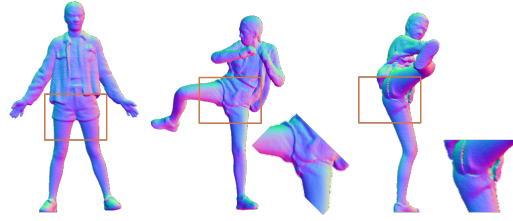


Figure 7. Failure case.

the standard deviations (SD) in Tab. 4, which demonstrates: 1) **High Consensus on Our Method:** The low SD (around 0.65) confirms participants consistently preferred our high-fidelity results. 2) **Consensus on Baselines:** Low SD for LHM’s geometry indicates a shared expert opinion on its limited dynamic naturalness. 3) **Analysis of w/o Condition:** This variant exhibited higher SD. While maintaining per-frame quality, it suffers from identity drifting. The higher variance suggests participants were divided on whether such identity shifts are acceptable.

Table 4. User study metrics (SD shown in brackets). “g” and “r” denote raw geometry and rendering results, respectively.

Method	Quality	Naturalness	Comformance
LHM (r)	3.3 (1.09)	2.7(1.00)	2.0(1.13)
LHM (g)	1.3 (0.95)	1.7 (0.62)	1.5 (1.00)
Long-term	3.0 (1.38)	2.2 (1.19)	2.5 (1.13)
w/o Cond.	3.1 (1.38)	2.2 (1.34)	2.6 (1.45)
w/o Aug.	3.7 (1.06)	3.1 (0.97)	3.5 (1.10)
Ours	4.4 (0.64)	4.5 (0.65)	4.4 (0.65)

10.2. Failure Case

In Fig. 7, we show a failure case involving an outer cloth in an extreme pose. While the upper body drapes naturally as the avatar leans, the pants exhibit unnatural deformations (middle) and seam artifacts (right). This is primarily because the training set lacks data for this specific clothing style performing such extreme movements.

10.3. Trade-off Between Sampling Density and Quality

Beyond Tab. 1 and Fig. 4 in the main paper, we evaluate the impact of sampling density on visual quality via FID (Tab. 5). While performance improves with more points, the FID gain diminishes significantly beyond 400k points (improving only from 26.93 to 26.36 at 500k).

Table 5. FID under different sampling points.

Points	100k	200k	300k	350k	400k	450k	500k
FID ↓	61.11	37.49	29.33	27.80	26.93	26.55	26.36

10.4. Other Comparisons

In addition to methods already mentioned in the main paper, we add a more recent method “Idol [64]” for comparison. Our method outperforms others across all metrics with strong user consensus (indicated by the low standard deviations).

Table 6. Quantitative comparison. For user study (Quality, Naturalness, Conformance \uparrow), we report mean (std) across 12 users. “g” and “r” denote geometry and rendering variants.

Method	FID \downarrow	Quality \uparrow	Naturalness \uparrow	Conformance \uparrow
E3Gen (g)	72	2.38 (1.19)	2.38 (1.26)	2.31 (1.11)
E3Gen (r)	67	2.62 (0.87)	2.46 (1.20)	2.31 (1.18)
GetAvatar (g)	91	1.08 (0.28)	1.31 (0.85)	1.38 (0.65)
GetAvatar (r)	61	2.08 (0.86)	1.85 (0.80)	1.92 (0.76)
gDNA (g)	55	2.31 (0.75)	2.23 (0.93)	1.92 (0.76)
gDNA (r)	51	3.00 (1.08)	2.69 (1.32)	2.62 (1.39)
Idol (g)	74	2.38 (1.04)	2.54 (1.13)	2.54 (1.13)
Idol (r)	68	2.77 (1.01)	2.62 (1.12)	2.69 (1.11)
HuGeoDis	37	3.00 (1.08)	3.15 (1.07)	3.46 (1.20)
Ours	25	4.38 (0.77)	4.62 (0.65)	4.42 (0.66)

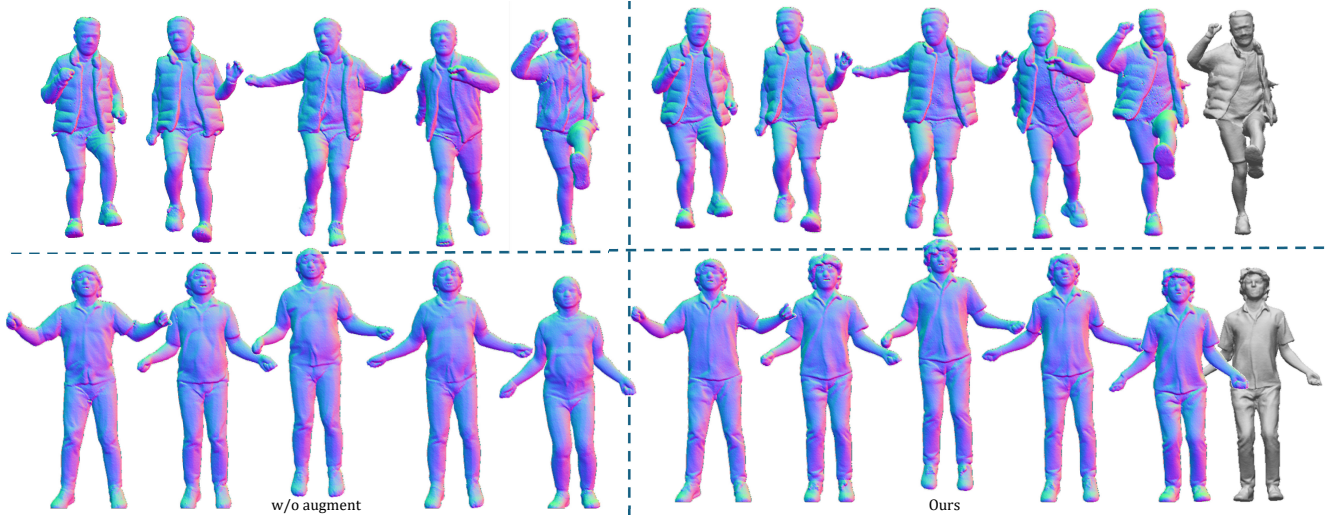


Figure 8. Comparison between our model and the “w/o augment” model. The first row shows a leg-kicking motion, where the outer garment naturally follows the body movement, while the second row depicts a rope-jumping motion. The “w/o augment” model tends to generate garments associated with other character identities, indicating identity leakage across sequences.

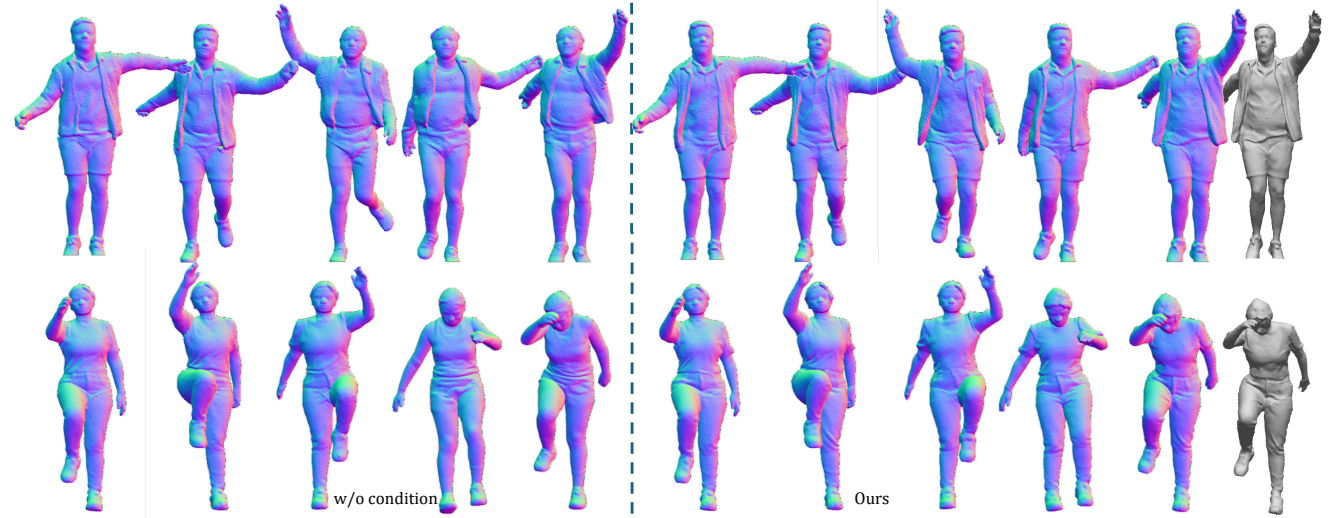


Figure 9. Comparison between our model and the “w/o condition” model. The first row shows running with swinging arms, and the second row shows leg lifts with arm swings. Without long-term consistency, the generated character identity gradually changes over time.

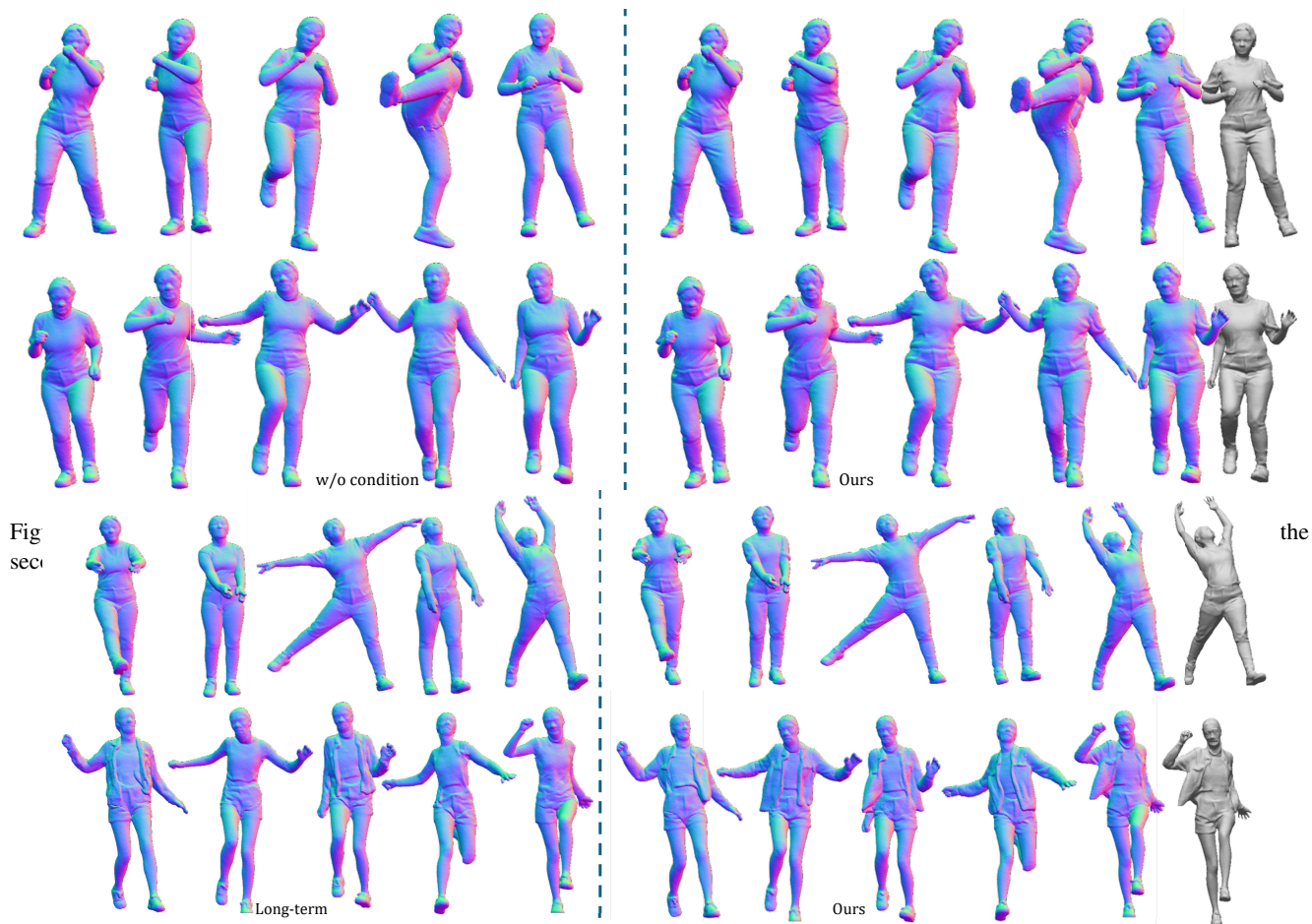


Figure 11. Comparison between our model and the “long-term” model. The first row shows a dance stretching motion and the second row shows a kicking motion. “Long-term” model generates low-quality and unnatural geometric details for unseen motions.

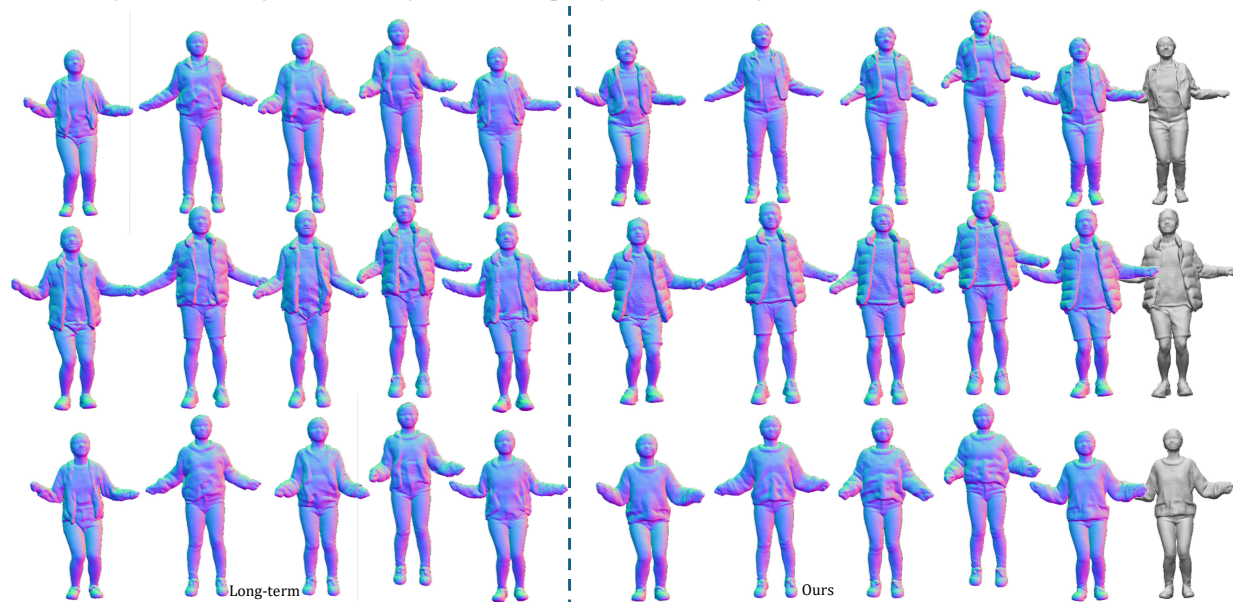


Figure 12. Comparison between our model and the “long-term” model. Diverse avatars with a rope-jumping motion are demonstrated. “Long-term” model generates low-quality and unnatural geometric details for unseen motions.