Computational Visual Media
https://doi.org/10.1007/s41095-022-0324-2 Vol. 10, No. 2, April 2024, 261–277

Research Article

Generating diverse clothed 3D human animations via a generative model

Min Shi[1], Wenke Feng[1], Lin Gao[2], and Dengming Zhu[2] (✉)

Abstract Data-driven garment animation is a current topic of interest in the computer graphics industry. Existing approaches generally establish the mapping between a single human pose or a temporal pose sequence, and garment deformation, but it is difficult to quickly generate diverse clothed human animations. We address this problem with a method to automatically synthesize dressed human animations with temporal consistency from a specified human motion label. At the heart of our method is a two-stage strategy. Specifically, we first learn a latent space encoding the sequence-level distribution of human motions utilizing a transformer-based conditional variational autoencoder (Transformer-CVAE). Then a garment simulator synthesizes dynamic garment shapes using a transformer encoder–decoder architecture. Since the learned latent space comes from varied human motions, our method can generate a variety of styles of motions given a specific motion label. By means of a novel beginning of sequence (BOS) learning strategy and a self-supervised refinement procedure, our garment simulator is capable of efficiently synthesizing garment deformation sequences corresponding to the generated human motions while maintaining temporal and spatial consistency. We verify our ideas experimentally. This is the first generative model that directly dresses human animation.

Keywords Transformer; garment animation; conditional variational autoencoder (CVAE); computer graphics

1 School of Control and Computer Engineering, North China Electric Power University, Beijing 102206, China. E-mail: M. Shi, shi_min@ncepu.edu.cn; W. Feng, fwk@ncepu.edu.cn.

2 Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China. E-mail: L. Gao, gaolin@ict.ac.cn; D. Zhu, mdzhu@ict.ac.cn (✉).

Manuscript received: 2022-05-30; accepted: 2022-11-06

1 Introduction

Rapidly synthesizing diverse 3D clothed human animations given a motion label is a significant and challenging task in computer graphics. It can be easily used in visual effects, video games, virtual reality, etc. To generate desired clothed human animations in a motion category, two kinds of approaches are normal: physics-based simulation (PBS) and data-driven techniques. As the predominant technique, PBS achieves realistic, rich wrinkles in dynamic garments, but it comes with expensive computing costs and complex authoring procedures, making it unsuitable for real-time and web-based applications. An alternative is to train a data-driven model that produces the plausible garment shapes from the deformation sources [1–3], such as human shape, pose, garment style, etc. However, such approaches generally regard it as a translation task that largely relies on the prior information. This is an overly constrained scenario and many applications such as virtual reality and garment animation authoring require animating 3D humans of a given type (e.g., given a semantic motion label). Other methods [4–6] focus on generating garment deformation directly conditioned by given information such as garment type, body shape, etc. However, they consider the problem for a single frame, with the result that the synthesized garments are not temporally and spatially consistent with the underlying bodies.

Our goal differs: we aim to directly generate a body motion sequence with detailed garment shapes. We observe that garment deformations are dependent on human motion, and various types of garments present different wrinkle details for the same motion sequence. Hence, the key challenge is to generate motion sequences and then synthesize corresponding dynamic garment shapes with temporal and spatial coherence.

In this paper, we present a two-stage strategy to train a body motion generation network and a dynamic garment regressor separately. Concretely, we train a body motion generator based on a Transformer-CVAE that is conditioned by semantic motion labels. We aim to learn a sequence-level embedding latent space for every kind of motion. After training, our motion generator is capable of generating diverse realistic human motions given a motion label. We formulate garment shape synthesis as a pure translation task following the usual approach [1–3]. Since the two branches are trained separately, a few unexpected visual effects such as interpenetration between body and garment may arise. To alleviate this phenomenon, we employ a novel strategy to learn BOS and further introduce a self-supervised collision removal procedure, which provides superior predicted garment shapes with temporal and spatial consistency. Sample instances generated by our method are presented in Fig. 1.

In summary, the main contributions of our work are:

- the first temporal generative model that is capable of generating a variety of clothed human animations by specifying a motion label,
- learning a sequence-level latent space encoding the distribution of human motions, which is conditioned by motion labels, and
- learning a plug-and-play garment simulator which can automatically synthesize dynamic garment shapes with the help of a novel learnt beginning-of-sequence strategy and a refinement procedure.

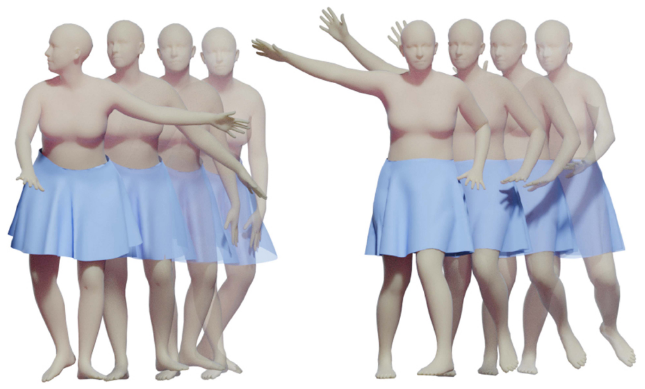
Fig. 1 Sample output from our model, which is capable of synthesizing clothed 3D human animations with temporal and spatial coherence.

2 Related work

2.1 Human motion synthesis

As a fundamental procedure in the digital world, human motion synthesis has received significant attention. Previous studies have conditioned motion synthesis on text [7], language [8], action [9, 10], music [11, 12], or motion style [13]. Action2Motion [9] and ACTOR [10] are most similar to our motion generator; both are able to generate diverse motions by specifying motion labels. However, Action2Motion [9] uses an autoregressive approach and the latent representation is at frame level. In contrast, ACTOR [10] and our method both encode a sequence-level latent space by combining Transformer and VAE, which provides significant advantages, as shown in our experiments. However, there are various differences between ACTOR [10] and our motion generator. To obtain the sequence-level embedding, we both pool the time dimension. Doing so fails to capture various features including temporal continuity. When restoring a motion sequence from a latent point, the results generated by ACTOR [10] are only derived from that latent point and the time information only acts as a query condition. Instead, our motion generator first merges the distributional latent point with time information, giving fused features acting as the key, value, and query condition at the same time, which helps maintain the temporal consistency.

While the above previous studies are able to generate human animations with realism, they still miss an important component: clothing. Our plug-and-play clothing simulator goes further and can synthesize rich and dynamic garment details according to the underlying body motions.

2.2 Garment animation

2.2.1 Physics-based simulation

PBS has been widely used in computer graphics. Three main processes are involved: modeling of the internal cloth forces [14], collision detection, and response [15, 16]. This complex modeling procedure is slow and computationally expensive. To accelerate the simulation, some approaches use adaptive remeshing [17], or GPU acceleration [18] to achieve a trade-off between speed and simulation accuracy. However, PBS is still resource-hungry and cannot be used in many applications such as mobile games.

2.2.2 Data-driven clothing models

There is a vast literature on data-driven clothing models. A common approach in earlier studies is to train a data-driven model that captures linear deformations [19, 20]. However, folds usually deform in detail in a nonlinear way, so the resulting wrinkles lack realism, and even present blending artifacts. To synthesize folds as richly as possible, a number of works [1–3, 21–25] learn nonlinear deformation over different variables, including body shape, body pose, garment style, etc. Lähner et al. [21] and Xu et al. [22] predicted pose-dependent garment deformations for a fixed body shape and garment style. To model shape-dependent and pose-dependent deformations jointly, Santesteban et al. [1] and Wu et al. [23] learned a clothing shape and clothing pose model separately and synthesized the resulting clothing deformation by blending the results from the shape-dependent and pose-dependent model. Differing from them, Gundogdu et al. [24] presented a method that directly retains geometric features of garments by adopting curvature losses, which can further be extended to different body shapes and poses. As garment style is strongly associated with perception of draping of the clothing, Wang et al. [25] learned a shared shape space for multimodal garment design. To model the multi-source deformation, Patel et al. [2] presented TailorNet, a neural model which predicts clothing deformations in 3D as a function of human shape, pose, and garment style. Going further, Tiwari and Bhowmick [3] learned to predict garment deformation as a function of human shape, pose, measurement, and garment style, which makes prediction more accurate. However, these studies [1–3, 21–25] are most suitable for tight garments such as T-shirts and pants. Pan et al. [26] presented a learning algorithm to predict deformations of loose-fitting garments; it uses bone-driven motion networks. Like the work of Wang et al. [27], their work can estimate garment deformations with varying simulation parameters (e.g., cloth material properties). Nevertheless, for a different garment, it is necessary to re-train the network, since it is template-dependent [26, 27]. To design a method for general uses, e.g., without assumptions about the topology of the cloth or the shape or topology of obstacles, Li et al. [28] proposed N-Cloth for 3D cloth deformation prediction, which works for a variety of clothing of different topologies. Zhang et al. [29] presented a novel neural rendering pipeline to simulate and render dynamic garments based on a coarse-to-fine strategy, which is also suitable for loose garments. To alleviate the expense of a supervised scheme, Bertiche et al. [30] and Santesteban et al. [31] used neural garment simulators trained using a set of physics-based losses.

2.3 Generative models based on Transformers

With the success of the Transformer [32] in NLP, several lines of research have exploited a combination of Transformer with generative models such as VAE [33]. Example uses include story completion [34], speech synthesis [35], and music generation [36]. Following this line, we propose a Transformer-CVAE to synthesize clothed 3D human animations of a specific motion category in an immediate way.

3 Approach

In this section, we introduce our learning-based workflow. Figure 2 shows the overall structure of the proposed method. In Section 3.1, we present a formal definition of the problem. There are three essential components in our method: (i) a body motion generator, which allows us to generate realistic human motion sequences from a motion label (see Section 3.2), (ii) a dynamic garment regressor, which infers corresponding dynamic garment shapes with detailed wrinkles (see Section 3.3), and (iii) a self-supervised body-garment collision handler that penalizes interpenetration between body and garment without extra prior information (see Section 3.4).

3.1 Overview

3.1.1 Problem definition

Given a motion label l, our goal is to synthesize a body motion sequence $\Theta = \{\theta_1, \cdots, \theta_T\}$ with dynamic garment shapes $\boldsymbol{G} = \{G_1, \cdots, G_T\}$. We propose a two-stage strategy for this task, including human motion generation and garment shape synthesis. To generate human motions in a specific motion category, we train a Transformer-CVAE to infer a latent space that encodes the sequence-level probabilistic distribution of human motions, which takes Θ and the motion label $l \in \mathbb{R}^m$ (encoded as a one-hot vector) as inputs. The learned latent vector Z can be applied to generate various motion sequences. For garment shape synthesis, we construct a network to establish

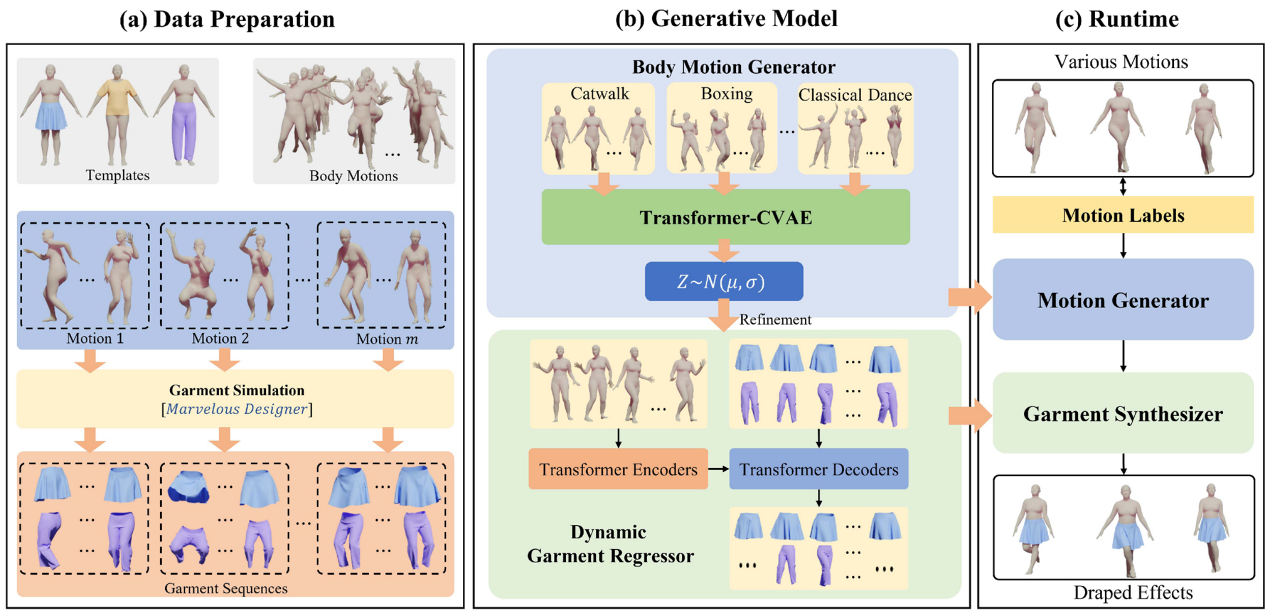

Fig. 2 Architecture of our model. The data preprocessing stage generates physics-based simulations for multiple garment types, for various body motions. The training uses a two-level strategy to construct a human motion generator and a dynamic garment regressor. After training, the workflow is capable of synthesizing realistic, temporally coherent, clothed 3D human motions from a given motion label.

the mapping between Θ and $\boldsymbol{G}$. At runtime, by inputting a motion label l, our workflow is able to generate various motion sequences from the latent space and further synthesize dynamic garment shapes through the garment synthesis network.

3.1.2 Data representation

Instead of treating human motion as the transformations of a sparse set of joints [37, 38], we incorporate both joints and vertices to represent body motions. Specifically, we adopt the SMPL [39] body model that is a disentangled body representation with parameterized shape (denoted β) and pose (denoted θ), in which $\beta \in \mathbb{R}^{10}$ is derived using principal component analysis and $\theta \in \mathbb{R}^{24\times3}$ is the axis-angle representation of the relative rotations of joints with respect to their parents in a kinematic tree. To make training easier, we employ a continuous 6D rotation representation [40], so $\theta \in \mathbb{R}^{24\times6}$. As for the body mesh vertices, we take the mean shape to get the mesh vertices sequence $V_1^B, \cdots, V_T^B$. Further, we utilize PBS to generate the initial garment instances with a common template $G = (V_{\text{tmp}}^G, F_{\text{tmp}}^G)$ where V_{tmp}^G is a $|V^G| \times 3$ matrix storing coordinates of the vertices while F_{tmp}^G stores the faces of the triangular mesh, and we take the vertex coordinates V_t^G $(\forall t \in \{1, \cdots, T\})$ to represent deformation while garment shape topology remains unchanged from F_{tmp}^G.

3.2 Body motion generator

3.2.1 Approach

Inspired by ACTOR [10], we construct our body motion generator based on a Transformer-CVAE. However, our network only consists of Transformer encoder layers while ACTOR [10] has Transformer encoder and decoder layers. From our perspective, in ACTOR [10], the learned latent vector Z acts as the key and value when it is fed into the Transformer decoder, and the temporal input $\{1, 2, \cdots, T\}$ of the Transformer decoder serves as the query. In other words, the reconstructed motions are fully restored from the latent vector by ignoring temporal consistency. However, in our implementation, we add Z to the temporal information element-wise, and regard the results as queries, keys, and values at the same time, thereby maintaining the sequential consistency of the reconstruction. We further adopt a multilayer perception (MLP) to learn the classification tokens that facilitate learning the posterior distribution for every kind of motion. The overall architecture of our motion generator is shown in Fig. 3.

3.2.2 Body motion encoder

To learn the probabilistic distribution of the i-th kind of motion, our motion encoder M_E takes a sequence of pose parameters $\theta_1, \cdots, \theta_T$ and a

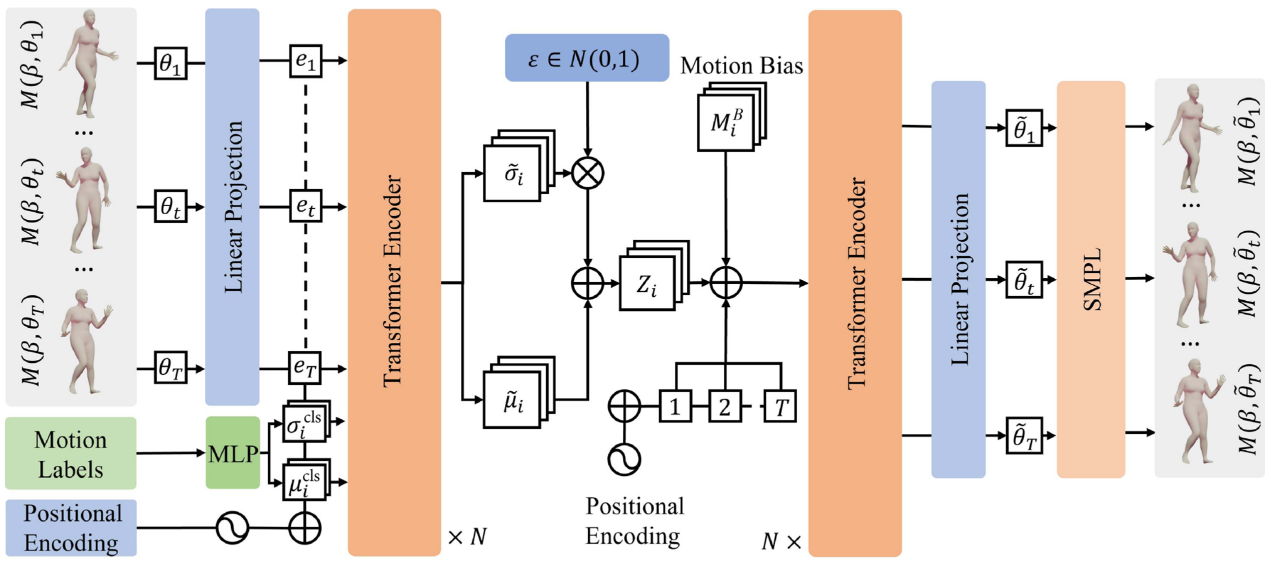

Fig. 3 Detailed architecture of our body motion generator. To learn the distribution parameters for each type of motion, we first employ a fully connected network to get the motion category tokens, and concatenate them to the pose embeddings. Through the encoder, we take the first two items of the latent embeddings as the posterior distributional features and discard the rest. We also employ a learnable motion bias for each motion and add it element-wise to the latent vector. To enable the decoder to capture distributional and temporal features simultaneously, we add the learned latent vector to the temporal input of the decoder element-by-element. Finally, through the decoder and an SMPL layer, we synthesize the sequenced meshes.

motion label as inputs, and outputs the posterior distribution parameters $\tilde{\mu}_i$ and $\tilde{\sigma}_i$. With the help of a reparameterization trick [33], we get the final latent vector $Z_i \in \mathbb{R}^{d_{\text{model}}}$. Specifically, following BERT [41] and ViT [42], we first construct a label encoder L_E using a fully connected network to learn the motion class tokens μ_i^{cls} and σ_t^{cls} for the i-th kind of motion. Then we append these class distribution parameters to the pose embeddings $e_1, \cdots, e_T$, which are the outputs of the linear projection module, and then add the positional information to the sequence based on a sinusoidal function. Finally, with N stacked Transformer encoder layers, the motion encoder learns the sequence-level distribution characterized by the parameters $\tilde{\mu}_i$ and $\tilde{\sigma}_i$ for the i-th kind of motion. Note that, in our method, the encoder only focuses on creating a sequence-level embedding in the latent space, while the decoder is devoted to restoring sequences by aggregating temporal and distributional features. Hence, we only take the first two parameters of the output as distributional parameters and discard the rest.

3.2.3 *Body motion decoder*

Unlike Ref. [10], our motion decoder M_D is also composed of Transformer encoder layers. To enable our body motion generator to gather both distributional and temporal features, we add the latent vector Z to the temporal information element-wise. We further adopt a learnable motion bias for every motion latent space. Then, through N stacked Transformer encoder layers, the motion decoder outputs the predicted pose sequence $\tilde{\theta}_1, \cdots, \tilde{\theta}_T$. Finally, we employ an SMPL layer to obtain the vertices $\tilde{V}_1^B, \cdots, \tilde{V}_T^B$ of the predicted body meshes, which participate in the loss calculation.

3.2.4 *Loss function*

The training loss functions used are:

- Reconstruction loss. We adopt joints and surface vertices to represent the transformation, so the reconstruction loss has two terms: L_P for pose reconstruction and L_V^B for vertex reconstruction. In both cases we use the L2 loss between the ground-truth and the prediction:

$$L_{\text{recon}} = \sum_{t=1}^{T} \| \theta_t - \tilde{\theta}_t \|_2^2 + \sum_{t=1}^{T} \| V_t^B - \tilde{V}_t^B \|_2^2 \quad (1)$$

  where T is the sequence length, $\tilde{\theta}_t$ is the t-th pose parameter in the predicted pose sequence, and $\tilde{V}_t^B$ represents corresponding mesh vertices.
- Distribution loss. We employ a standard VAE [33] with the temporal network. From the hypothesis that all samples follow a normalized

Gaussian distribution, we minimize the discrepancy between the prior distribution and the posterior characterized by the learned parameters $\tilde{\mu}$ and $\tilde{\sigma}$, which is represented by the Kullback–Leibler (KL) divergence.

Overall, the training loss of our motion generator is

$$L_B = L_P + L_V^B + \lambda_{\mathrm{KL}} L_{\mathrm{KL}} \tag{2}$$

where λ_{KL} is the weight of the distribution loss, also known as β-VAE [43].

3.3 Dynamic garment regressor

In this section, we describe our garment regressor used to predict detailed garment shapes. The basic architecture comprises M Transformer pose encoder and garment decoder layers, as depicted in Fig. 4.

3.3.1 *Pose encoder*

Differing from the sequence-level embedding of the body motion generator, our garment regressor aims to learn a frame-level mapping from body motion to garment deformation. Specifically, given a sequence of body poses $\theta_1, \cdots, \theta_T$, the pose encoder P_E first transforms them to pose embeddings through a linear projection block. Next, using a multi-head attention block with h heads, our pose encoder captures the frame-level dependencies from different subspaces. Then, the multi-level temporally-dependent features are integrated into an intermediate vector through a position-wise feed-forward network. Finally, using M stacked Transformer encoder layers, the pose encoder outputs the latent sequence embeddings, which are used by the garment decoder for keys and values.

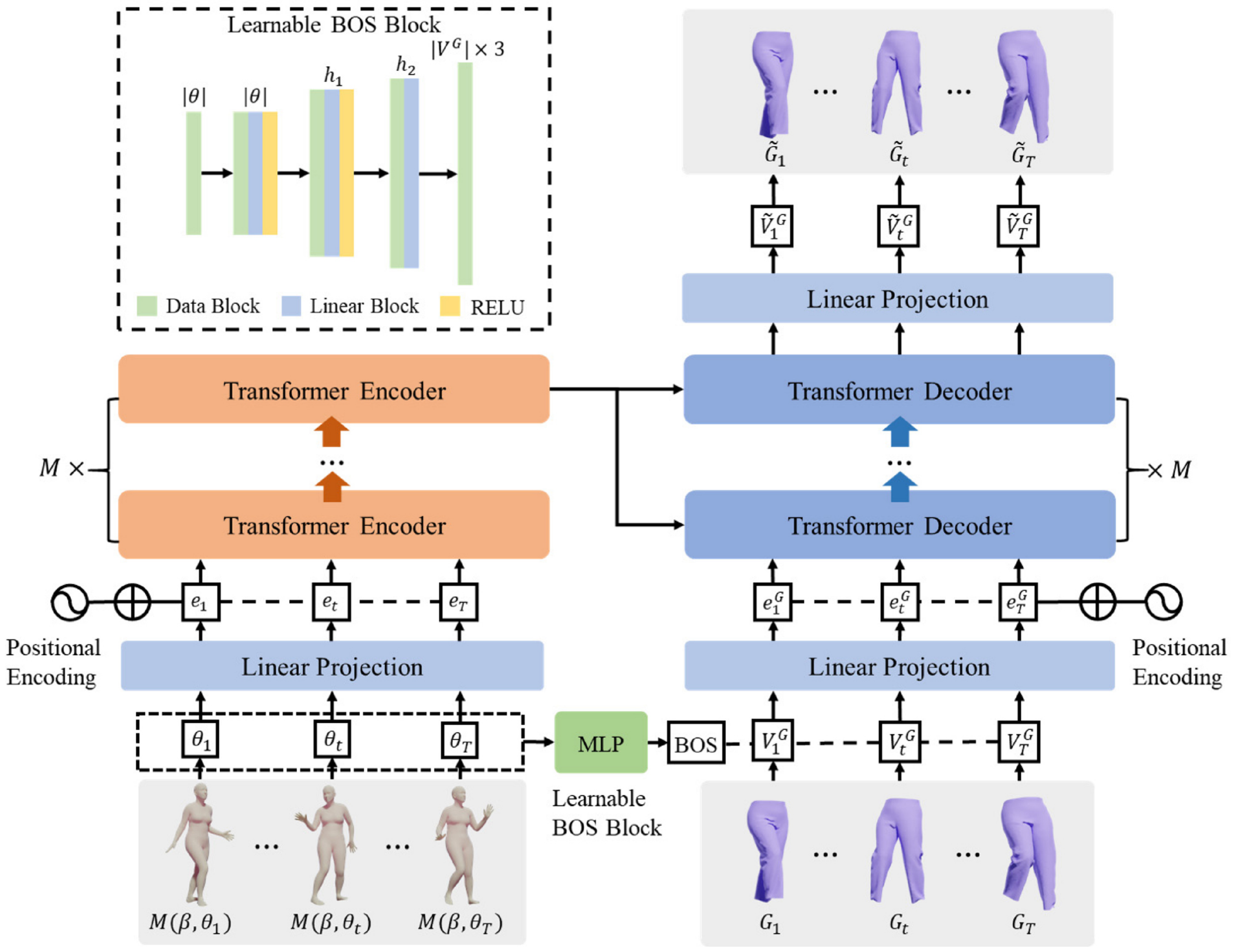

Fig. 4 Architecture of our garment regressor. We feed a sequence of body poses and garment vertex sequences obtained by PBS into Transformer encoders and decoders separately. Then the frame-level latent embeddings obtained by the encoder are provided to the decoder as keys and values, which is essential to maintain the temporal consistency between body motion and garment deformation. We further employ a learnable BOS to reduce error accumulation during prediction. Our method can generate garment dynamics temporally and spatially consistent with the underlying bodies.

3.3.2 Garment decoder

The garment decoder G_D takes the vertices $V_t^G (\forall t \in 1, \cdots, T)$ of the original garments and the output of the pose encoder as inputs. As in the pose encoder, the vertices are first projected to the garment embeddings. Since the garment shapes are generated sequentially, the garment embeddings are firstly fed into the masked multi-head attention block, which masks the remaining frames. The masked multi-head attention module produces sequential queries which are then fed into the following multi-head attention block. The final garment embeddings output by M stacked Transformer decoder layers are projected back to the garment vertices $\tilde{V}_1^G, \cdots, \tilde{V}_T^G$. Note that the length of the input garment vertex sequence is $T+1$, with first element a beginning-of-sequence (BOS) symbol. In our approach, we employ a learnable BOS strategy to learn a specific BOS from the input poses. Compared to using a fixed symbol, our learnable BOS strategy helps reduce deviations significantly.

3.3.3 Training loss

We adopt two types of losses to train the garment regressor: the reconstruction loss L_V^G and the mesh Laplacian loss on vertices L_{Lap}. The former penalizes global shape differences and the latter aims to preserve wrinkle details. Formally,

$$L_G = \sum_{t=1}^{T} \|V_t^G - \tilde{V}_t^G\|_2^2 + \lambda_{\text{Lap}} \sum_{t=1}^{T} \|\Delta(V_t^G) - \Delta(\tilde{V}_t^G)\|_2^2 \tag{3}$$

where $\Delta(\cdot)$ is the Laplacian operator [44], and λ_{Lap} represents its weight.

3.4 Self-supervised collision handler

Our body motion generator and garment regressor are trained separately. However, our goal is to synthesize clothed 3D human animations in an end-to-end manner. In other words, the garment regressor needs to be integrated into our generative model. To achieve this, we introduce self-supervised collision removal handling to refine the garment regressor, following established work [45]. Specifically, we randomly sample a number of human motions from the latent space of the body motion generator and synthesize the corresponding garment sequences. Then, we fine-tune the garment regressor in a self-supervised manner without extra ground-truth data. The collision loss term is defined as

$$L_{\text{collision}} = \sum_{t=1}^{T} \max(-n_t^B \cdot (\tilde{V}_t^G - V_t^B), 0) \tag{4}$$

where $\tilde{V}_t^G$ is the t-th set of predicted garment mesh vertices, V_t^B is the set of closest body mesh vertices, and n_t^B represents the normals of these body vertices. Since the refinement process relies on synthesized garments driven by randomly sampled motions, for the uncovered area in the latent space, we apply further post-processing to remove garment intersections with the body. Furthermore, our collision handler cannot cope with some cases of deep interpenetration, e.g., when the closest body vertices are on the opposite of the body.

4 Implementation

In this section, we first introduce the details of the dataset used, and then present the implementation details of the overall network.

4.1 Dataset

We employ AMASS [46] to generate the human motion set. This is a huge synthetic human motion set that provides detailed body parameters, such as shape and pose parameters in SMPL [39]. We first extracted motions with semantic labels, like dancing, boxing, etc., and then discarded invalid sequences, for example, those with interpenetration between the torso and limbs, as the corresponding garment is also invalid. Finally, we utilize SMPL to generate the human motion data by ignoring shape variation (setting $\beta = (0, \cdots, 0)^{\mathrm{T}}$). Overall, our motion dataset consists of $m = 66$ types of motions with semantic labels encoded as one-hot vectors. There are 108,705 frames in total; each body mesh has $|V^B| = 6890$ vertices and 13,776 faces.

Using the motion sequences, we construct 3 garment datasets: skirt, T-shirt, and pants. For each garment type, we first designed a template in an A-pose using Marvelous Designer (https://www.marvelousdesigner.com), and then all garment shapes were simulated using on the same template. Details for each garment template are presented in Table 1. We use vertex coordinates to represent garment deformations, so our model can be easily extended to other garments, even those with complex typologies. Figure 5 illustrates our garment dataset.

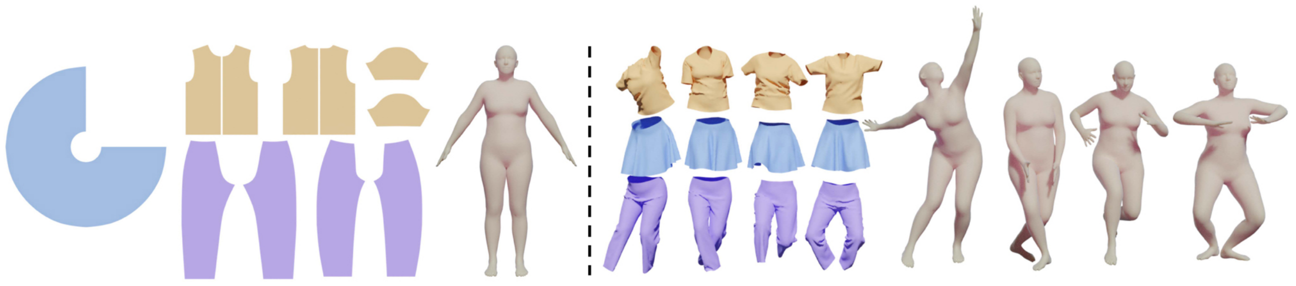

Fig. 5 Testing our generative model on three garment types, both loose-fitting (a skirt), and close-fitting (a T-shirt and pants). We first design templates based on a standard body in A-pose (left), and then generate numbers of garment shapes in different poses (right).

Table 1 Garment templates

Type	Vertices ($\lvert V^G_{\text{tmp}}\rvert$)	Faces ($\lvert F^G_{\text{tmp}}\rvert$)
Skirt	3326	6490
T-shirt	3170	6234
Pants	4473	8870

In our experiments, we set the sequence length $T = 60$; longer sequences are split into multiple segments.

We take 90% of the samples as our training set, and the other 10% as the test set.

4.2 Network details

We have implemented our approach using PyTorch [47] and PyTorch3D [48]. We now provide details of each branch.

For the motion branch, the label encoder L_E contains a hidden layer with 128 neurons followed by ReLU [49] activation. The motion encoder M_E and motion decoder M_D both consist of $N = 8$ Transformer encoder layers, while the hidden size of the feed-forward network is 1024. For the garment branch, the learnable BOS block takes the mean value of the pose sequence as input and outputs the BOS for the corresponding garment sequence. There are two hidden layers which contain $h_1 = 1024$ and $h_2 = 4096$ neurons. Each hidden layer is followed by an activation layer. P_E and G_D have $M = 2$ Transformer encoder and decoder layers separately as their backbone architecture, while the hidden layer of the feed-forward network comprises 512 neurons. Other common settings exist for these branches: both have 4 heads in the multi-head attention blocks, and for both, embedding size and latent dimension is $d_{\text{model}} = 256$, with ReLU activation functions in the Transformer blocks. The collision handler has the same architecture as the garment branch.

During training, we set $\lambda_{\text{KL}} = 10^{-5}$, and $\lambda_{\text{Lap}} = 0.1$. We adopt the Adam [50] optimizer to train our model. For the motion branch, we use 5000 epochs with a fixed batch size of 20 and an initial learning rate of 10^{-4}. For the garment branch, we use 2000 epochs, an initial batch size of 10, and a learning rate of 10^{-3}. We increase the batch size by 5 after every 200 epochs and fix it at 25 for the remaining period. Both branches employ a dynamic learning rate strategy. We reduce the learning rate by a factor 0.1 if the loss fails to decrease after every 30 epochs. To avoid overfitting, we employ L2 normalization and randomly disable 10% of the hidden neurons. In the garment branch, we pretrain the MLP to learn the BOS, supervised by the last frame of the current sequence, and then freeze its weights when training the garment synthesizer.

In the refinement stage, we first sample numbers of motions and synthesize the corresponding deformed garment instances. Then, we eliminate invalid sequences (e.g., with heavy interpenetration) and feed the valid ones to the garment regressor. During the refinement stage, we set the learning rate to the same magnitude as the final value for the garment branch.

5 Experiments

5.1 Simulation speed

We have implemented our method on an Intel Core i9-10900K CPU with an Nvidia RTX 3090 GPU with 24 GB of RAM. Table 2 shows the average per-frame execution time of our integrated pipeline. We take ARCSim [17] as a reference; it is a CPU-based implementation of a full physics-based simulation. Taking the T-shirt example, for a motion sequence of 60 frames, ARCSim consumes about 3.9 s per frame, while the inference time per frame of our method is just 12.3 ms, about 300 times faster than ARCSim.

Table 2 Per-frame simulation time for ARCSim—a physics-based method, and our method

Clothing	ARCSim	Our method	Speedup
Skirt	3.2841 s	0.0117 s	281×
T-shirt	3.9047 s	0.0123 s	317×
Pants	8.9763 s	0.0132 s	680×

5.2 Evaluation of body motion generator

In this section, we evaluate the body motion generator.

5.2.1 Reconstruction accuracy

We evaluated the reconstruction accuracy using three metrics: RMSE (root mean square error), Hausdorff distance, and STED (spatio-temporal edge distance) [51]. RMSE measures the distance between the prediction and the ground truth for both pose parameters and mesh vertices. Hausdorff distance is used to estimate the similarity of the predicted mesh vertex set and the ground truth mesh vertex set, while STED is used to measure the distance of the dynamic mesh sequence. We compare our results to those from Action2Motion [9] and ACTOR [10] in Table 3. The methods using sequence-level latent embedding (ACTOR and ours) achieve superior reconstruction results compared to the method using frame-level embedding (Action2Motion). However, compared to ACTOR, our motion generator provides more accurate motion reconstruction. Quantitatively, our motion generator achieves an improvement of 17% and 15% in RMSE of pose and vertex respectively, 4% in Hausdorff distance, and 21% in STED compared to ACTOR.

Table 3 Reconstruction errors for unseen motion sequences for Action2Motion, ACTOR, and our method. We employ three metrics: RMSE (for pose parameters, and mesh vertices), Hausdorff distance, and STED

Method	RMSE ↓ ($\times 10^{-2}$)		Hausdorff↓ ($\times 10^{-1}$)	STED ↓ ($\times 10^{-2}$)
	Pose	Vertex		
Action2Motion	19.500	15.366	2.653	3.269
ACTOR	9.634	7.155	2.510	1.965
Ours	**7.933**	**6.038**	**2.405**	**1.533**

5.2.2 Generative performance

To evaluate generative performance, we use the measures employed in Action2Motion [9], comparing our method to Action2Motion and ACTOR [10]. Thus, we employ four metrics: FID (Frechet inception distance), recognition accuracy, overall diversity, and multimodality, which represents per-motion diversity. When calculating these metrics, a pre-trained motion recognition model is applied: we re-train the motion recognition model described in Ref. [9] using our dataset. The results are presented in Table 4. Like Action2Motion [9] and ACTOR [10], we generate sets of sequences 20 times and average the results. Note that metrics for the real motion and various methods are calculated based on the motions from the test set and the generated motions. Compared to Action2Motion and ACTOR, our motion generator achieves superior generative performance. However, we note that the recognition accuracy given in Table 4 is lower than the results provided in Action2Motion [9] and ACTOR [10]. We believe the reason is that the dataset in our paper is split using higher-level semantic motion labels only (e.g., dance, but not left-kick), which means that some similar sub-sequences may appear in different kinds of motions. Such sub-sequences may be categorized as the wrong motion type.

Table 4 Generative performance of various methods, as measured by FID, recognition accuracy, diversity, and multimodality. → means motions are better when the metric is closer to that for real motion

Method	FID ↓	Accuracy↑	Diversity→	Multimodality→
Real motion	0.039±0.003	94.0±0.1	7.06±0.02	2.91±0.01
Action2Motion	0.182±0.022	89.9±0.3	6.99±0.02	2.59±0.06
ACTOR	0.062±0.004	90.3±0.2	7.01±0.01	3.07±0.05
Ours	0.042±0.002	93.3±0.2	7.05±0.02	2.89±0.01

5.2.3 Motion consistency

Since our goal is to generate clothed 3D human motions with temporal-coherence, motion consistency is one of the most important indicators in evaluating the generated output. However, the generated body motions are randomly sampled from the latent space, and lack corresponding real motions. Thus, we conducted a qualitative visual evaluation of temporal consistency, for simplicity assessed by counting the number of sequences considered visually discontinuous by volunteers (i.e., showing a sudden transformation over a few frames when playing at 25 fps). Overall, we sampled 100 sequences and employed 24 volunteers divided into 3 groups, with results presented in Table 5. Our motion generator has better motion consistency.

Table 5 User evaluation of the generated motion sequences, giving the proportion of sequences judged to be discontinuous. The 24 volunteers were divided into 3 groups

Method	Group 1	Group 2	Group 3	Overall
ACTOR	8.25%	10.43%	13.75%	10.81%
Ours	**6.75%**	**7.86%**	**7.63%**	**7.41%**

5.3 Evaluation of garment regressor

5.3.1 Comparison to state-of-the-art methods

We now quantitatively and qualitatively compare our garment regressor to prior learning-based methods. For quantitative evaluation, we employ four metrics: RMSE, Hausdorff distance, STED, and cosine similarity, the last being used to evaluate vertex normal similarity between predicted meshes and the ground truth. Firstly, we make comparisons to the methods of Pan et al. [26] and Wang et al. [27]. Since code for generating virtual bones and skinning weights is not available, we used dress03 from the dataset of Pan et al. to re-train our network and that of Wang et al. We make the comparison using a fixed pair of simulation parameters; results are given in Table 6. The results demonstrate that our method achieves significant improvements in garment synthesis. We also conducted a visual evaluation, with predicted meshes shown in Fig. 6(a) and a per-vertex error map shown in Fig. 6(b). It can be seen that our method is capable of generating more accurate results.

Table 6 Comparison to state-of-the-art methods on the dress03 dataset from Pan et al., in terms of RMSE, Hausdorff distance, STED, and cosine similarity

Method	RMSE↓ ($\times 10^{-3}$)	Hausdorff↓ ($\times 10^{-2}$)	STED↓ ($\times 10^{-2}$)	Cosine similarity↓
Wang et al.	17.825	5.103	10.730	0.3383
Pan et al.	6.487	4.165	7.805	0.1904
Ours	**4.834**	**2.521**	**6.394**	**0.1695**

We further compared our garment regressor to the methods of Santesteban et al. [1], TailorNet [2], and PBNS [30], using the simulation result for female old-t-shirt for a medium body and a neutral style ($\beta = 0$, $\gamma = 0$) from TailorNet to re-train our garment regressor. To re-train the garment fit regressor and the garment wrinkle regressor of Santesteban et al., we picked simulation results with neutral style for different shapes and poses as training data, and

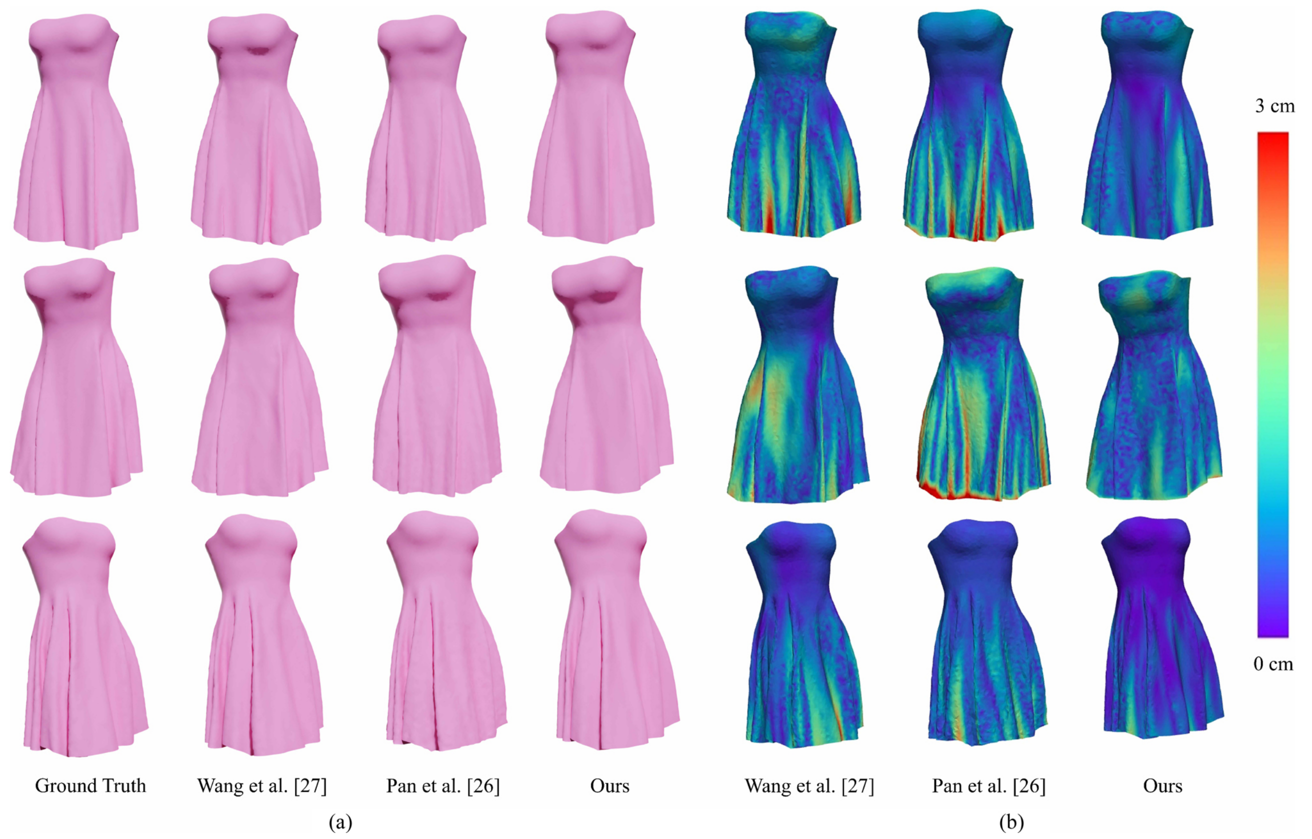

Fig. 6 Visual comparison of generated meshes using the dress03 dataset of Pan et al. [26]. (a) Ground truth and predictions from various methods. (b) Per-vertex error map; warmer colors represents larger errors.

set the mean style garment mesh for the medium body as the garment template. For PBNS, we used the garment template and the skinning weights of different poses of the mean body to re-train the network. Again, we adopted RMSE, Hausdorff distance, STED, and cosine similarity for evaluation. Table 7 gives the results; a few instances generated by the different methods are shown in Fig. 7. While the method of Santesteban et al. and TailorNet can generate the overall shape of the garment, detailed wrinkles are missing. For PBNS, while the physics-based neural clothing simulator is able to synthesize plausible deformations, the overall shape and style (e.g., size) may change, which is undesirable in applications like virtual try-on. In contrast, our garment regressor is capable of synthesizing garment deformations with overall shape and detailed wrinkles both in line with the ground truth.

5.4 Ablation study

We conducted an ablation study in a controlled setup for several components of our approach, to validate our choice of loss function and architecture.

Table 7 Comparison of various methods on the old-t-shirt dataset from TailorNet

Method	RMSE↓ ($\times10^{-3}$)	Hausdorff↓ ($\times10^{-2}$)	STED↓ ($\times10^{-2}$)	Cosine similarity↓
Santesteban et al.	10.054	3.231	4.788	0.2280
TailorNet	10.361	4.512	6.531	0.2114
PBNS	16.179	5.109	9.186	0.3087
Ours	**4.169**	**2.554**	**2.746**	**0.1265**

5.4.1 Loss function

We investigated the influence of the body mesh vertex loss L_V^B in the motion reconstruction loss by omitting it. We adopted RMSE of reconstructed mesh vertex positions, Hausdorff distance, and STED to assess the results relative to the full loss, as shown in Table 8. We also visualize a few instances of both strategies in Fig. 8. We observe that pose loss by itself is insufficient

Table 8 Effect on the motion generator of omitting the mesh vertex loss

Loss	RMSE ↓ ($\times10^{-2}$)	Hausdorff ↓ ($\times10^{-2}$)	STED ↓ ($\times10^{-3}$)
L_P	2.263	6.245	6.230
$L_P + L_V^B$	**1.447**	**6.128**	**5.830**

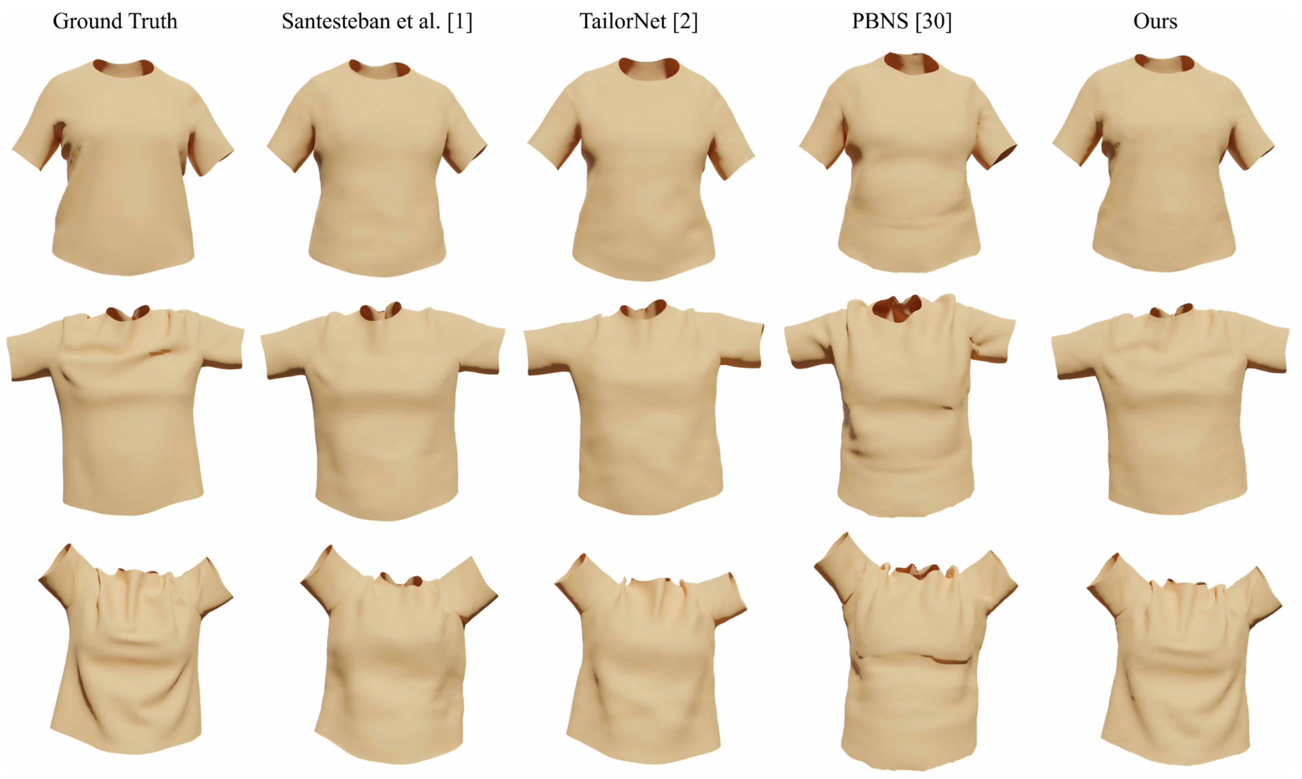

Fig. 7 Visual evaluation of different methods on the old-t-shirt-female dataset from TailorNet [2]. Left to right: ground truth, results from Santestban et al. [1], TailorNet [2], PBNS [30], and our method.

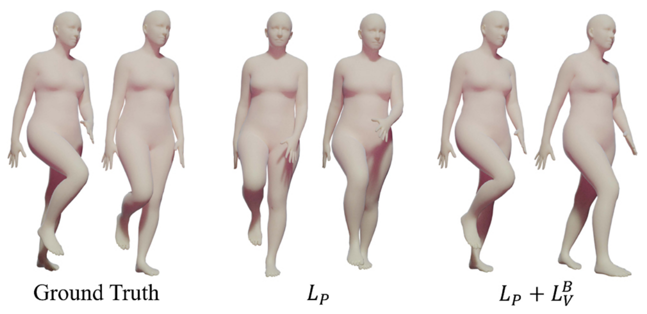

Fig. 8 Visual effect on output of omitting reconstruction loss. L_P and $L_P + L_V^B$ are our method without the mesh vertex loss and with the full reconstruction loss, respectively.

to provide good reconstruction, and omitting vertex loss can lead to unexpected visual effects, such as the rotation seen in Fig. 8.

5.4.2 *Learnable BOS*

Since our inference stage is based on a greedy decoding strategy, prediction errors accumulate as time progresses. To alleviate this problem, we adopt a learnable BOS strategy; here, we compare it to using a fixed BOS (zero in our experiments). Figure 9 shows example results, which represent the first frame of the prediction. Using a fixed strategy, there are serious intersections between body and predicted garment, which is detrimental to the overall prediction. Our learnable scheme mostly avoids such interpenetration. A quantitative assessment is provided in Table 9, taking the skirt as an example, employing the same four metrics as before; it demonstrates the benefit of using a learnable BOS.

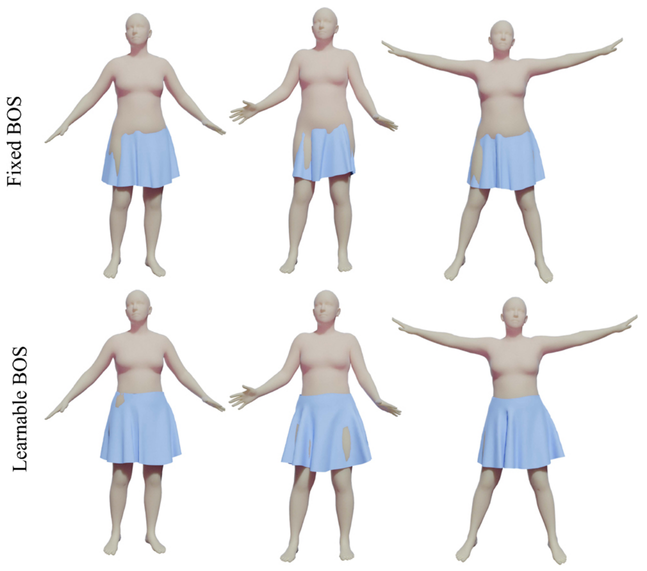

Fig. 9 Fixed and learnable BOS strategies: the learnable BOS greatly reduces intersections between the body and garment.

Table 9 Comparison of fixed and learnable BOS strategies for unseen data

BOS	RMSE ↓ ($\times10^{-3}$)	Hausdorff ↓ ($\times10^{-2}$)	STED ↓ ($\times10^{-2}$)	Cosine similarity ↓
Fixed	9.535	7.168	6.263	0.1197
Learnable	**3.384**	**3.858**	**6.158**	**0.1073**

5.4.3 *Refinement*

Our method includes a self-supervised collision removal procedure to refine our garment regressor's results. Figure 10 presents a few examples with and without refinement. Interpenetrating regions are improved, as our self-supervised collision handler succeeds in pushing out those garment vertices that lie within the underlying body, which significantly helps to keeping results temporally and spatially consistent with body motions.

5.5 Applications

5.5.1 *Clothed human animation synthesis*

To demonstrate the generative performance of our overall pipeline, we make a comparison to the state-

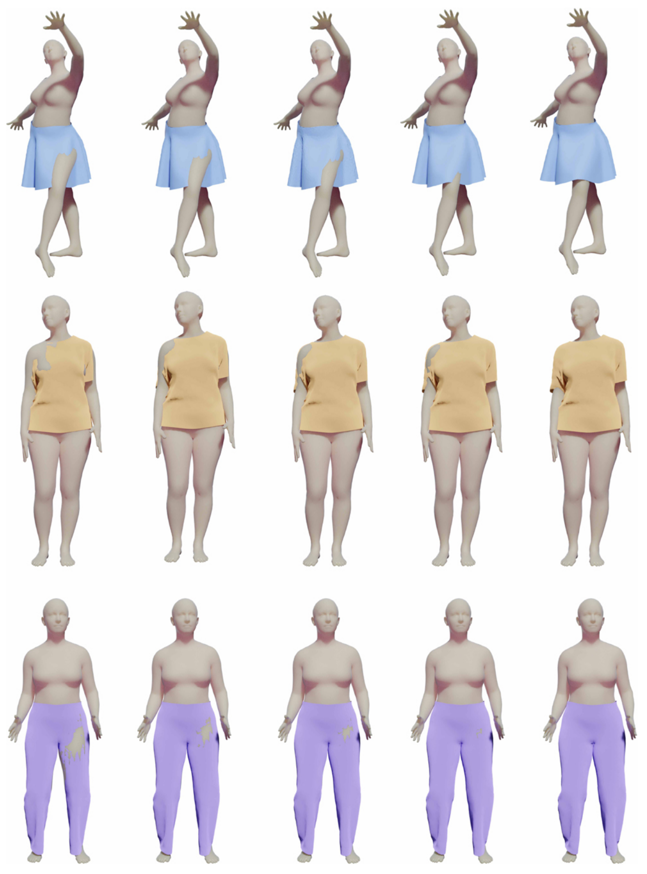

Fig. 10 Effectiveness of our self-supervised refinement strategy. Left to right: our collision handler progressively reduces interpenetration between the garment and body as time progress.

of-the-art method: CAPE [4], which is also capable of generating clothed 3D human body meshes directly. However, CAPE only uses a frame-level latent representation, while our approach creates a sequence-level embedding in the latent space that is able to synthesize clothed 3D human animations directly. CAPE generates single clothed 3D body meshes, ignoring temporal consistency between frames, which results in discontinuities between adjacent frames. These can be seen in the red rectangles in the example in Fig. 11(top). In comparison the sequences generated by our method maintain are continuous and coherent. Furthermore, CAPE treats the body and garment as a single mesh, making it difficult for animators to re-author animations. CAPE is also powerless to tackle the loose garment types like skirts and dresses.

5.5.2 Generation of new sequences

We have also explored the representational power of the learned latent space. We can produce reasonable latent vectors with the same motion and feed them to the decoder to generate diverse but plausible motion sequences, even though they are not present in the dataset. These still retain the characteristics of the original data: Fig. 12 shows several examples. We observe that there is a great diversity in the way a given motion is performed. For example, when generating new Chinese classical dance sequences, our model retains as many essential features as possible while creating a few new movements (e.g., via rotation angles of the limbs). Our model is capable of generalizing to new sequences in different ways for a given motion.

Ultimately, while our workflow aims to provide a clothed human animation synthesizer in an end-to-end manner, the body motion generator and the garment regressor can be separately applied in different domains since they are trained separately. For example, like Action2Motion [9] and ACTOR [10], our motion generator could also be used for action recognition. The garment simulator could be used to aid regular studies, as it is able to automatically

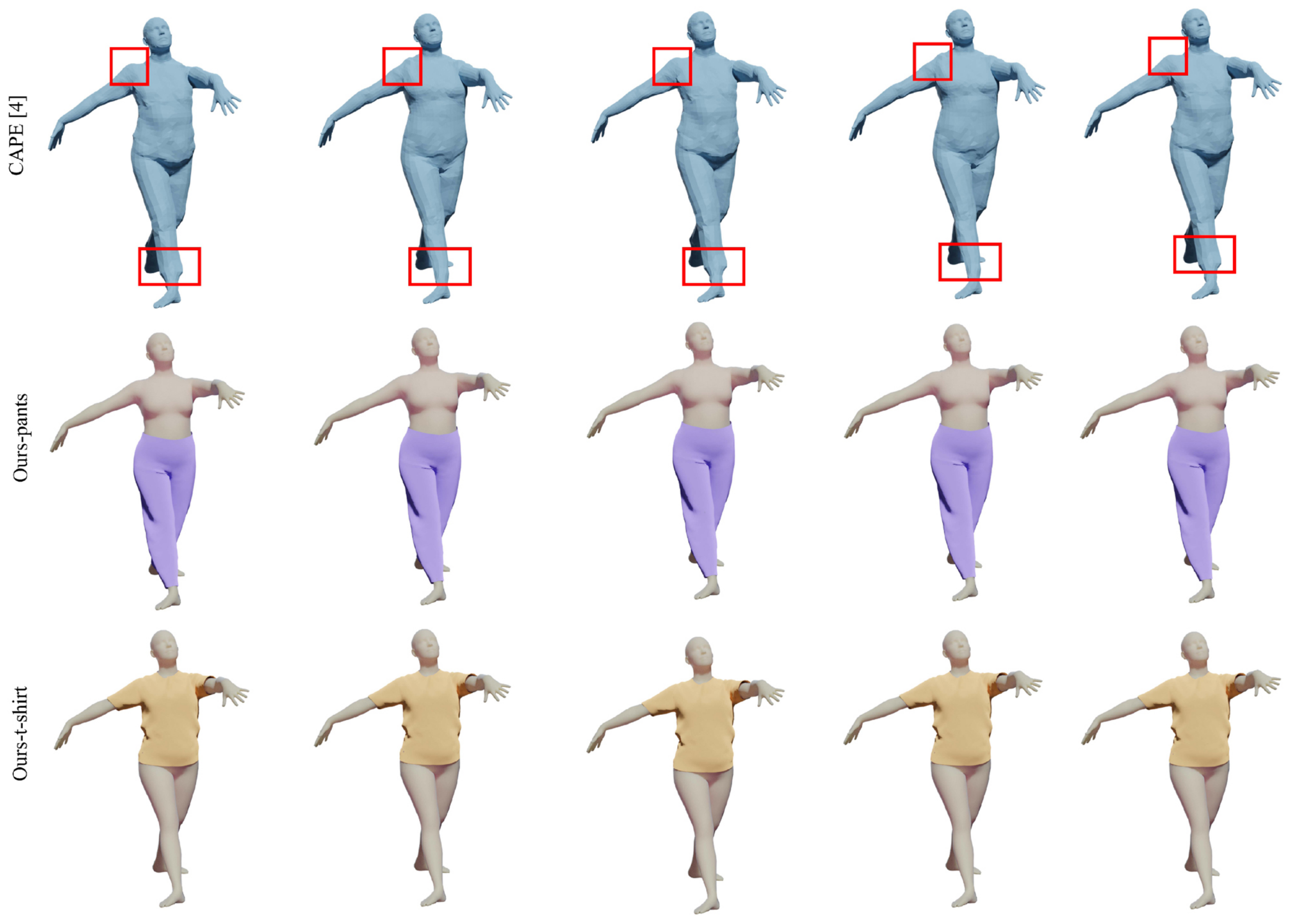

Fig. 11 Comparison of our generated results directly sampled from the latent space by giving a specific motion label, and those from CAPE.

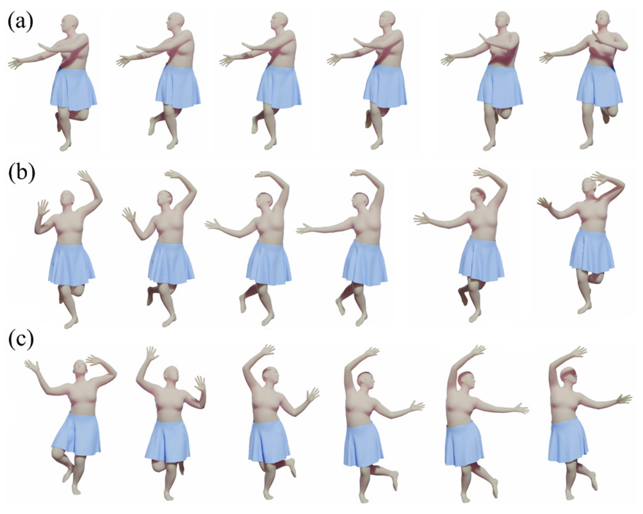

Fig. 12 Three diverse Chinese classical dance animations, newly synthesized from the latent space. Left to right: 6 equally spaced frames from 60 frame generated sequences.

synthesize garment deformations according to given motion parameters.

6 Conclusions and future work

In this paper, we have introduced a novel temporal generative model for generating clothed 3D human animations in real time. To realize the goal, we proposed a two-stage strategy, training a body motion generator with a sequence-level latent representation, and a garment regressor that can synthesize realistic variable garment shapes. To eliminate interpenetration between body and garment, we employ a novel learnable BOS strategy and further employ a self-supervised collision removal handler, which efficiently reduce interpenetrations at runtime. We have constructed three garment datasets to evaluate the proposed workflow, and showed that our approach achieves superior results for both motion generation and garment synthesis.

However, we hope to address two remaining limitations. Firstly, although we provide a plug-and-play garment synthesizer, shape-dependent input parameters are not considered in the proposed workflow. To consider more variables, it is necessary to retrain or re-design the model to consider these configurations. Secondly, our model is based on Transformer blocks, which means that it only makes predictions segment-by-segment during inferencing. Further consideration is needed to design a network capable of synthesizing long sequences (say more than 1000 frames) of clothed human animations as a whole.

Acknowledgements

We thank the volunteers for the user study. This work was supported by the National Natural Science Foundation of China (Grant No. 61972379).

Declaration of competing interest

The authors have no competing interests to declare that are relevant to the content of this article.

References

[1] Santesteban, I.; Otaduy, M. A.; Casas, D. Learning-based animation of clothing for virtual try-on. *Computer Graphics Forum* Vol. 38, No. 2, 355–366, 2019.

[2] Patel, C.; Liao, Z.; Pons-Moll, G. TailorNet: Predicting clothing in 3D as a function of human pose, shape and garment style. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 7363–7373, 2020.

[3] Tiwari, L.; Bhowmick, B. DeepDraper: Fast and accurate 3D garment draping over a 3D human body. In: Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops, 1416–1426, 2021.

[4] Ma, Q. L.; Yang, J. L.; Ranjan, A.; Pujades, S.; Pons-Moll, G.; Tang, S. Y.; Black, M. J. Learning to dress 3D people in generative clothing. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 6468–6477, 2020.

[5] Bertiche, H.; Madadi, M.; Escalera, S. CLOTH3D: Clothed 3D humans. In: *Computer Vision – ECCV 2020. Lecture Notes in Computer Science, Vol. 12365*. Vedaldi, A.; Bischof, H.; Brox, T.; Frahm, J. M. Eds. Springer Cham, 344–359, 2020.

[6] Santesteban, I.; Thuerey, N.; Otaduy, M. A.; Casas, D. Self-supervised collision handling via generative 3D garment models for virtual try-on. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 11758–11768, 2021.

[7] Ahn, H.; Ha, T.; Choi, Y.; Yoo, H.; Oh, S. Text2Action: Generative adversarial synthesis from language to action. In: Proceedings of the IEEE International Conference on Robotics and Automation, 5915–5920, 2018.

[8] Ahuja, C.; Morency, L. P. Language2Pose: Natural language grounded pose forecasting. In: Proceedings of the International Conference on 3D Vision, 719–728, 2019.

[9] Guo, C.; Zuo, X. X.; Wang, S.; Zou, S. H.; Sun, Q. Y.; Deng, A. N.; Gong, M. L.; Cheng, L. Action2Motion: Conditioned generation of 3D human motions. In: Proceedings of the 28th ACM International Conference on Multimedia, 2021–2029, 2020.

[10] Petrovich, M.; Black, M. J.; Varol, G. Action-conditioned 3D human motion synthesis with transformer VAE. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, 10965–10975, 2021.

[11] Lee, H. Y.; Yang, X.; Liu, M. Y.; Wang, T. C.; Lu, Y. D.; Yang, M. H.; Kautz, J. Dancing to music. In: Proceedings of the 33rd International Conference on Neural Information Processing Systems, Article No. 322, 3586–3596, 2019.

[12] Li, J. M.; Yin, Y. H.; Chu, H.; Zhou, Y.; Wang, T. W.; Fidler, S.; Li, H. Learning to generate diverse dance motions with transformer. *arXiv preprint* arXiv:2008.08171, 2020.

[13] Wen, Y. H.; Yang, Z. P.; Fu, H. B.; Gao, L.; Sun, Y. N.; Liu, Y. J. Autoregressive stylized motion synthesis with generative flow. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 13607–13607, 2021.

[14] Baraff, D.; Witkin, A. Large steps in cloth simulation. In: Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques, 43–54, 1998.

[15] Provot, X. Collision and self-collision handling in cloth model dedicated to design garments. In: *Computer Animation and Simulation '97. Eurographics.* Thalmann, D.; van de Panne, M. Eds. Springer Vienna, 177–189, 1997.

[16] Volino, P.; Magnenat Thalmann, N. Collision and self-collision detection: Efficient and robust solutions for highly deformable surfaces. In: *Computer Animation and Simulation '95. Eurographics.* Terzopoulos, D.; Thalmann, D. Eds. Springer Vienna, 55–65, 1995.

[17] Narain, R.; Samii, A.; O'Brien, J. F. Adaptive anisotropic remeshing for cloth simulation. *ACM Transactions on Graphics* Vol. 31, No. 6, Article No. 152, 2012.

[18] Li, C.; Tang, M.; Tong, R. F.; Cai, M.; Zhao, J. Y.; Manocha, D. P-cloth: Interactive complex cloth simulation on multi-GPU systems using dynamic matrix assembly and pipelined implicit integrators. *ACM Transactions on Graphics* Vol. 39, No. 6, Article No. 180, 2020.

[19] Guan, P.; Reiss, L.; Hirshberg, D. A.; Weiss, A.; Black, M. J. Drape. *ACM Transactions on Graphics* Vol. 31, No. 4, Article No. 35, 2012.

[20] Wang, H. M.; Hecht, F.; Ramamoorthi, R.; O'Brien, J. F. Example-based wrinkle synthesis for clothing animation. *ACM Transactions on Graphics* Vol. 29, No. 4, Article No. 107, 2010.

[21] Lähner, Z.; Cremers, D.; Tung, T. DeepWrinkles: Accurate and realistic clothing modeling. In: *Computer Vision – ECCV 2018. Lecture Notes in Computer Science, Vol. 11208.* Ferrari, V.; Hebert, M.; Sminchisescu, C.; Weiss, Y. Eds. Springer Cham, 698–715, 2018.

[22] Xu, W. W.; Umentani, N.; Chao, Q. W.; Mao, J.; Jin, X. G.; Tong, X. Sensitivity-optimized rigging for example-based real-time clothing synthesis. *ACM Transactions on Graphics* Vol. 33, No. 4, Article No. 107, 2014.

[23] Wu, N. N.; Chao, Q. W.; Chen, Y. Z.; Xu, W. W.; Liu, C.; Manocha, D.; Sun, W. X.; Han, Y.; Yao, X. R.; Jin, X. G. AgentDress: Realtime clothing synthesis for virtual agents using plausible deformations. *IEEE Transactions on Visualization and Computer Graphics* Vol. 27, No. 11, 4107–4118, 2021.

[24] Gundogdu, E.; Constantin, V.; Seifoddini, A.; Dang, M.; Salzmann, M.; Fua, P. GarNet: A two-stream network for fast and accurate 3D cloth draping. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, 8738–8747, 2019.

[25] Wang, T. Y.; Ceylan, D.; Popovic, J.; Mitra, N. J. Learning a shared shape space for multimodal garment design. *arXiv preprint* arXiv:1806.11335, 2018.

[26] Pan, X. Y.; Mai, J. M.; Jiang, X. W.; Tang, D. X.; Li, J. X.; Shao, T. J.; Zhou, K.; Jin, X. G.; Manocha, D. Predicting loose-fitting garment deformations using bone-driven motion networks. In: Proceedings of the ACM SIGGRAPH Conference, Article No. 11, 2022.

[27] Wang, Y. T.; Shao, T.; Fu, K.; Mitra, N. Learning an intrinsic garment space for interactive authoring of garment animation. *ACM Transactions on Graphics* Vol. 38, No. 6, Article No. 220, 2019.

[28] Li, Y. D.; Tang, M.; Yang, Y.; Huang, Z.; Tong, R. F.; Yang, S. C.; Li, Y.; Manocha, D. N-cloth: Predicting 3D cloth deformation with mesh-based networks. *Computer Graphics Forum* Vol. 41, No. 2, 547–558, 2022.

[29] Zhang, M.; Wang, T. Y.; Ceylan, D.; Mitra, N. J. Dynamic neural garments. *ACM Transactions on Graphics* Vol. 40, No. 6, Article No. 235, 2021.

[30] Bertiche, H.; Madadi, M.; Escalera, S. PBNS: Physically based neural simulator for unsupervised garment pose space deformation. *ACM Transactions on Graphics* Vol. 40, No. 6, Article No. 198, 2021.

[31] Santesteban, I.; Otaduy, M. A.; Casas, D. SNUG: Self-supervised neural dynamic garments. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 8130–8140, 2022.

[32] Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, ?; Polosukhin, I. Attention is all you need. In: Proceedings of the 31st International Conference on Neural Information Processing Systems, 6000–6010, 2017.

[33] Kingma, D. P.; Welling, M. Auto-encoding variational Bayes. *arXiv preprint* arXiv:1312.6114, 2013.

[34] Wang, T. M.; Wan, X. J. T-CVAE: Transformer-based conditioned variational autoencoder for story completion. In: Proceedings of the 28th International Joint Conference on Artificial Intelligence, 5233–5239, 2019.

[35] Kumar, S.; Pradeep, J.; Zaidi, H. Learning robust latent representations for controllable speech synthesis. *arXiv preprint* arXiv:2105.04458, 2021.

[36] Jiang, J. Y.; Xia, G. G.; Carlton, D. B.; Anderson, C. N.; Miyakawa, R. H. Transformer VAE: A hierarchical model for structure-aware and interpretable music representation learning. In: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, 516–520, 2020.

[37] Barsoum, E.; Kender, J.; Liu, Z. C. HP-GAN: Probabilistic 3D human motion prediction via GAN. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, 1499–149909, 2018.

[38] Habibie, I.; Holden, D.; Schwarz, J.; Yearsley, J.; Komura, T. A recurrent variational autoencoder for human motion synthesis. In: Proceedings of the 28th British Machine Vision Conference, 119.1–119.12, 2017.

[39] Loper, M.; Mahmood, N.; Romero, J.; Pons-Moll, G.; Black, M. J. Smpl. *ACM Transactions on Graphics* Vol. 34, No. 6, Article No. 248, 2015.

[40] Zhou, Y.; Barnes, C.; Lu, J. W.; Yang, J. M.; Li, H. On the continuity of rotation representations in neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 5738–5746, 2019.

[41] Devlin, J.; Chang, M. W.; Lee, K.; Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint* arXiv:1810.04805, 2018.

[42] Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X. H.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint* arXiv:2010.11929, 2020.

[43] Higgins, I.; Matthey, L.; Pal, A.; Burgess, C.; Glorot, X.; Botvinick, M.; Mohamed, S.; Lerchner, A. beta-VAE: Learning basic visual concepts with a constrained variational framework. In: Proceedings of the International Conference on Learning Representations, 2017.

[44] Taubin, G. A signal processing approach to fair surface design. In: Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques, 351–358, 1995.

[45] Vidaurre, R.; Santesteban, I.; Garces, E.; Casas, D. Fully convolutional graph neural networks for parametric virtual try-on. *Computer Graphics Forum* Vol. 39, No. 8, 145–156, 2020.

[46] Mahmood, N.; Ghorbani, N.; Troje, N. F.; Pons-Moll, G.; Black, M. AMASS: Archive of motion capture as surface shapes. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, 5441–5450, 2019.

[47] Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z. M.; Desmaison, A.; Antiga, L.; Lerer, A. Automatic differentiation in PyTorch. In: Proceedings of the NIPS Workshop Autodiff, 2017.

[48] Ravi, N.; Reizenstein, J.; Novotny, D.; Gordon, T.; Lo, W. Y.; Johnson, J.; Gkioxari, G. Accelerating 3D deep learning with PyTorch3D. *arXiv preprint* arXiv:2007.08501, 2020.

[49] Agarap, A. F. Deep learning using rectified linear units (ReLU). *arXiv preprint* arXiv:1803.08375, 2018.

[50] Kingma, D. P.; Ba, J. Adam: A method for stochastic optimization. *arXiv preprint* arXiv:1412.6980, 2014.

[51] Vasa, L.; Skala, V. A perception correlated comparison method for dynamic meshes. *IEEE Transactions on Visualization and Computer Graphics* Vol. 17, No. 2, 220–230, 2011.

Min Shi is an associate professor in the School of Control and Computer Engineering, North China Electric Power University. She received her Ph.D. degree in computer science and technology from the Chinese Academy of Sciences in 2013. Her research interests include cloth simulation, computer vision, and virtual reality.

Wenke Feng received his B.S. degree in software engineering from North China Electric Power University in 2018, where he is currently pursuing an M.S. degree in computer science and technology. His research interests include computer graphics and garment animation.

Lin Gao received his Ph.D. degree in computer science from Tsinghua University. He is currently an associate professor at the Institute of Computing Technology, Chinese Academy of Sciences. He has held a Newton Advanced Fellowship from the Royal Society and an AG Young Researcher Award. His research interests include computer graphics and geometric processing.

Dengming Zhu received his B.S. degree from Ningbo University, China, in 1996, his M.S. degree in theoretical physics from Shanghai Jiao Tong University, China, in 2001, and his Ph.D. degree in computer science from the Institute of Computing Technology, Chinese Academy of Sciences, in 2008. He is currently an associate professor in the Institute of Computing Technology. His research interests include computer graphics and virtual reality.