GarNet: A Two-Stream Network for Fast and Accurate 3D Cloth Draping

Erhan Gundogdu¹, Victor Constantin¹, Amrollah Seifoddini²
Minh Dang², Mathieu Salzmann¹, Pascal Fua¹
¹CVLab, EPFL, Switzerland
²Fision Technologies, Zurich, Switzerland
{erhan.gundogdu, victor.constantin, mathieu.salzmann, pascal.fua}@epfl.ch
{amrollah.seifoddini, minh.dang}@fision-technologies.com

Abstract

While Physics-Based Simulation (PBS) can accurately drape a 3D garment on a 3D body, it remains too costly for real-time applications, such as virtual try-on. By contrast, inference in a deep network, requiring a single forward pass, is much faster. Taking advantage of this, we propose a novel architecture to fit a 3D garment template to a 3D body. Specifically, we build upon the recent progress in 3D point cloud processing with deep networks to extract garment features at varying levels of detail, including point-wise, patch-wise and global features. We fuse these features with those extracted in parallel from the 3D body, so as to model the cloth-body interactions. The resulting two-stream architecture, which we call as GarNet, is trained using a loss function inspired by physics-based modeling, and delivers visually plausible garment shapes whose 3D points are, on average, less than 1 cm away from those of a PBS method, while running 100 times faster. Moreover, the proposed method can model various garment types with different cutting patterns when parameters of those patterns are given as input to the network.

1. Introduction

Garment simulation is useful for many purposes such as virtual try-on, online shopping, gaming, and virtual reality. Physics-Based Simulation (PBS) can deliver highly realistic results, but at the cost of heavy computation, which makes it unsuitable for real-time and web-based applications. As shown in Fig. 1, in this paper, we propose to train a deep network to produce visually plausible 3D draping results, as achieved by PBS, but much faster.

Realistic simulation of cloth draping over the human body requires accounting for the global 3D pose of the person and for the local interactions between skin and cloth caused by the body shape. To this end, we introduce the architecture depicted by Fig. 2. It consists of a garment stream and a body stream. The body stream uses a PointNet [36] inspired architecture to extract local and global information about the 3D body. The garment stream exploits the global body features to compute point-wise, patch-wise and global features for the garment mesh. These features, along with the global ones obtained from the body, are then fed to a fusion subnetwork to predict the shape of the fitted garment. In one implementation of our approach, shown in Fig. 2a, the local body features are only used implicitly to compute the global ones. In a more sophisticated implementation, we explicitly take them into account to further model the skin-cloth interactions. To this end, we introduce an auxiliary stream that first computes the $K$ nearest body vertices for each garment vertex, performs feature pooling on point-wise body features and finally feeds them to the fusion subnetwork. This process is depicted by Fig. 2b. We will see that it performs better than the simpler one, indicating that local feature pooling is valuable.
By incorporating appropriate loss terms in the objective function that we minimize during training, at test time, we avoid the need for extra post-processing steps to minimize cloth-body interpenetration and undue tightness that PBS tools rely on convolution and pooling operations, our approach naturally scales to point clouds of arbitrary resolution. This is in contrast to data-driven methods that rely on a low-dimensional subspace whose size would typically need to grow as the resolution increases, thus strongly affecting these models' memory requirements.

Our contribution is therefore a novel architecture for static garment simulation that delivers fitting results in real-time by properly modeling the body and garment interaction, thus reducing cloth and body interpenetration. For training purposes, we built a dataset that will be made public. It comprises a pair of jeans, a t-shirt and a sweater worn by 600 bodies from the SMPL dataset in various poses. Experiments on our dataset show that our network can effectively handle many body poses and shapes. Moreover, our approach can incorporate additional information, such as cutting patterns, when available. To illustrate this, we make use of the recently-published data of [41], which contains different garment types with varying cutting patterns. Our experiments demonstrate that our method outperforms the state-of-the-art one of [41] on this dataset. Finally, whereas the PBS approach that we take as reference takes more than 10 seconds to predict the shape of a garment, ours takes less than 70 ms, thus being practical for real-time applications.

2. Related Work

Many professional tools can model cloth deformations realistically using Physics-Based Simulation (PBS) [31, 39, 32, 12]. However, they are computationally expensive, which precludes real-time use. Furthermore, manual parameter tuning is often required. First, we briefly review recent approaches to overcoming these limitations. Then, we summarize the deep network architectures for 3D point cloud and mesh processing, and the related works for 3D human/cloth modeling.

Data-Driven Approaches. They are computationally less intensive and memory demanding, at least at run-time, and have emerged as viable competitors to PBS. One of the early methods relies on generating a set of garment-body pairs. At test time, the garment shape in an unseen pose is predicted by linearly interpolating the garments in the database. An earlier work proposes a data-driven estimation of the physical parameters of the cloth material while constructs a finite motion graph for detailed cloth effects. In [18], potential wrinkles for each body joint are stored in a database so as to model fine details in various body poses. However, it requires performing this operation for each body-garment pair. To speed up the computation, the cloth simulation is modeled in a low-dimensional linear subspace as a function of 3D body shape, pose and motion in [16]. [13] also models the relation between 2D cloth deformations and corresponding bodies in a low-dimensional space. [14] extends this idea to 3D shapes by factorizing the cloth deformations according to what causes them, which is mostly shape and pose. The factorized model is trained to predict the garment’s final shape. [38] trains an MLP and an RNN to model the cloth deformations by decomposing them as static and dynamic wrinkles. Both [14] and [38], however, require an a posteriori refinement to prevent cloth-body interpenetration. In a recent approach, [41] trains an MLP and an RNN to model the cloth deformations by decomposing them as static and dynamic wrinkles. Both [14] and [38], however, require an a posteriori refinement to prevent cloth-body interpenetration. In a recent approach, [41] trains an MLP and an RNN to model the cloth deformations by decomposing them as static and dynamic wrinkles. Both [14] and [38], however, require an a posteriori refinement to prevent cloth-body interpenetration. In a recent approach, [41] trains an MLP and an RNN to model the cloth deformations by decomposing them as static and dynamic wrinkles. Both [14] and [38], however, require an a posteriori refinement to prevent cloth-body interpenetration. In a recent approach, [41] trains an MLP and an RNN to model the cloth deformations by decomposing them as static and dynamic wrinkles.
Cloth fitting has been performed using 4D data scans as in [24, 34]. In [34], garments deforming over time are reconstructed using 4D data scans and the reconstructions are then retargeted to other bodies without accounting for physics-based clothing dynamics. Unlike in [34], we aim not only to obtain visually plausible results but also to emulate PBS for cloth fitting. In [24], fine wrinkles are generated by a conditional Generative Adversarial Network (GAN) that takes as input predicted, low-resolution normal maps. This method, however, requires a computationally demanding step to register the template cloth to the captured 4D scan, while ours needs only to perform skinning of the template garment shape using the efficient method of [20].

**Point Cloud and Mesh Processing.** A key innovation that has made our approach practical is the recent emergence of deep architectures that allow for the processing of point clouds [36, 37] and meshes [40]. PointNet [36, 37] was the first to efficiently represent and use unordered point clouds for 3D object classification and segmentation. It has spawned several approaches to point-cloud upsampling [46], unsupervised representation learning [44], 3D descriptor matching [11], and finding 2D correspondences [45]. In our architecture, as in PointNet, we use Multilayer Perceptrons (MLPs) for point-wise processing and max-pooling for global feature generation. However, despite its simplicity and representational power, point-wise operations in PointNet [36] is not sufficient to produce visually plausible garment fitting results, as we experimentally demonstrate by qualitative and quantitative analysis.

Given the topology of the point clouds, for example in the form of a triangulated mesh, graph convolution methods, unlike PointNet [36], can produce local features, such as those of [6, 27, 29] that rely on hand-crafted patch operators. FeastNet [40] generalizes this approach by learning how to dynamically associate convolutional filter weights with features at the vertices of the mesh, and demonstrates state-of-the-art performance on the 3D shape correspondence problem. Similar to [40], we also use mesh convolutions to extract patch-wise garment features that encode the neighborhood geometry. However, in contrast to the methods whose tasks are 3D shape segmentation [36, 37] or 3D shape correspondence [40, 6, 27, 29], we do not work with a single point cloud or mesh as input, but with two: one for the body and the other for the garment, which are combined in our two-stream architecture to account for both shapes.

**3D human body/cloth reconstruction.** 3D body shapes/cloth are modeled from RGB/RGBD cameras in [49, 43, 42, 15, 2, 1, 47, 48] while garment and surface reconstruction methods from images are addressed in surface/wrinkle reconstruction from images [9, 3, 35]. Moreover, generative models reconstruct cloths in [25, 16].

---

**Figure 3:** Garment branch of our network. The grey boxes and the numbers in parenthesis denote network layers and their output channel dimensions. Red and blue ones represent garment and global body features, respectively. The green box is the mesh convolution subnetwork and depicted in more detail in Fig. 4. STN stands for a Spatial Transformer Network used in PointNet [36]. MLP blocks are shared by all $N$ points.

**Figure 4:** Mesh conv. subnetwork. The residual block is repeated 6 times. Dashed red rectangles indicate channel-wise concatenation. The $N \times 3$-dimensional tensors contain the 3D vertex locations of the input garment, which are passed at different stages via skip connections.

---

**3. 3D Garment Fitting**

To fit a garment to a body in a specific pose, we start by using a dual quaternion skinning (DQS) method [20] that produces a rough initial garment shape that depends on body pose. In this section, we introduce two variants of our GarNet deep network to refine this initial shape and produce the final garment. Fig. 2 depicts these two variants.

**3.1. Problem Formulation**

Let $M^0$ be the template garment mesh in the rest pose and let $M = \text{dqs}(M^0, B, J_{M}^{\text{dqs}}, J_{B}, W)$ be the garment after skinning to the body $B$, also modeled as a mesh, by the method [20]. Here, $J_{M}^{\text{dqs}}$ and $J_{B}$ are the joints of $M^0$ and
\( B \), respectively. \( W \) is the skinning weight matrix for \( M^0 \).

Let \( f_0 \) be the network with weights \( \theta \) chosen so that the predicted garment \( G^P \) given \( M \) and \( B \) is as close as possible to the ground-truth shape \( G^G \). We denote the \( i^{th} \) vertex of \( M \), \( B \), \( G^G \) and \( G^P \) by \( M_i \), \( B_i \), \( G^G_i \) and \( G^P_i \in \mathbb{R}^3 \), respectively. Finally, let \( N \) be the number of vertices in \( M \), \( G^G \) and \( G^P \).

Since predicting deformations from a reasonable initial shape is more convenient than predicting absolute 3D locations, we train \( f_0 \) to predict a translation vector for each vertex of the warped garment \( M \) that brings it as close as possible to the corresponding ground-truth vertex. In other words, we optimize with respect to \( \theta \) so that

\[
\mathcal{T}^P = f_0(M, B) \approx \mathcal{T}^G ,
\]

where \( \mathcal{T}^P \) and \( \mathcal{T}^G \) correspond to translation vectors from the skinned garment \( M \) to the predicted and ground-truth mesh, respectively, that is \( G^P_i - M_i \) and \( G^G_i - M_i \). Therefore, the final shape of the garment mesh is obtained by adding the translation vectors predicted by the network to the vertex positions after skinning.

### 3.2. Network Architecture

We rely on a two-stream architecture to compute \( f_0(M, B) \). The first stream, or body stream, takes as input the body represented by a 3D point cloud while the second, or garment stream, takes as input the garment represented by a triangulated 3D mesh. Their respective outputs are fed to a fusion network that relies on a set of MLP blocks to produce the predicted translations \( \mathcal{T}^P \) of Eq. 1. To not only produce a rough garment shape, but also predict fine details such as wrinkles and folds, we include early connections between the two streams, allowing the garment stream to account for the body shape even when processing local information. As shown in Fig. 2, we implemented two different versions of the full architecture and discuss them in detail below.

**Body Stream.** The first stream processes the body \( B \) in a manner similar to that of PointNet [36] (see Sec. 3.4 for details). It efficiently produces point-wise and global features that adequately represent body pose and shape. Since there are no direct correspondences between 3D body points and 3D garment vertices, the global body features are key to incorporating such information while processing the garment. We observed no improvement by using mesh convolution layers in this stream.

**Garment Stream.** The second stream takes as input the warped garment \( M \) and the global body features extracted by the body stream to also compute point-wise and global features. As we will see in the results section, this suffices for a rough approximation of the garment shape but not to predict wrinkles and folds. We therefore use the garment mesh to create patch-wise features, that account for the local neighborhood of each garment vertex by using mesh convolution operations [40]. In other words, instead of using a standard PointNet architecture, we use the more sophisticated one depicted by Fig. 3 to compute point-wise, patch-wise, and global features. As shown in Fig. 3, the features extracted at each stage are forwarded to the later stages via skip connections. Thus, we directly exploit the low-level information while extracting higher-level representations.

**Fusion Network.** Once the features are produced by the garment and body streams, they are concatenated and given as input to the fusion network shown as a purple box in Fig. 2. It consists of four MLP blocks shared by all the points, as done in the segmentation network of PointNet [36]. The final MLP block outputs the 3D translations \( \mathcal{T}^P \) of Eq. 1 from the warped garment shape \( M \).

**Global and Local Variants.** Fig. 2a depicts the GarNet-Global version of our architecture. It discards the point-wise body features produced by the body stream and exclusively relies on the global body ones. Note, however, that the local body features are still implicitly used because the global ones depend on them. This enables the network to handle the garment/body dependencies without requiring explicit correspondences between body points and mesh vertices. In the more sophisticated GarNet-Local architecture depicted by Fig. 2b, we explicitly exploit the point-wise body features by introducing a nearest neighbor pooling step to compute separate local body features for each garment vertex. It takes as input the point-wise body features and uses a nearest neighbor approach to compute additional features that capture the proximity of \( M \) to \( B \) and feeds them into the fusion network, along with the body and garment features. This step shown in Fig. 5 improves the prediction accuracy due to the explicit use of local body features.

### 3.3. Loss Function

To learn the network weights, we minimize the loss function \( \mathcal{L}(G^G, G^P, B, M) \). We designed it to reduce the distance of the prediction \( G^P \) to the ground truth \( G^G \) while...
also incorporating regularizaton terms derived from physical constraints. The latter also depend on the body \( B \) and the garment \( M \). We therefore write \( \mathcal{L} \) as
\[
\mathcal{L}_{\text{vertex}} + \lambda_{\text{pen}} \mathcal{L}_{\text{pen}} + \lambda_{\text{norm}} \mathcal{L}_{\text{norm}} + \lambda_{\text{bend}} \mathcal{L}_{\text{bend}},
\]
where \( \lambda_{\text{pen}}, \lambda_{\text{norm}}, \) and \( \lambda_{\text{bend}} \) are weights associated with the individual terms described below. We will study the individual impact of these terms in the results section.

**Data Term.** We take \( \mathcal{L}_{\text{vertex}} \) to be the average \( L^2 \) distance between the vertices of \( G^G \) and \( G^P \),
\[
\frac{1}{N} \sum_{i=1}^{N} \left\| G^G_i - G^P_i \right\|^2,
\]
where \( N \) is the total number of vertices.

**Interpenetration Term.** To assess whether a garment vertex is inside the body, we first find the nearest body vertex. At each iteration of the training process, we perform this search for all garment vertices. This yields \( \mathcal{C}(B, G^P) \), a set of garment-body index pairs. We write \( \mathcal{L}_{\text{pen}} \) as
\[
\sum_{(i,j) \in \mathcal{C}(B, G^P)} \| \| G^G_i - G^P_j \| < d_{\text{tol}} \| \| \text{ReLU}(-(N^P_B_i(G^P_j - B_i))/N, \quad (4)
\]
to penalize the presence of garment vertices inside the body. Here, \( N^P_B_i \) is the normal vector at the \( i^{th} \) body vertex, as depicted by Fig 6a. This formulation penalizes garment vertex \( G^P_j \) for not being on the green subspace of its corresponding body vertex \( B_i \), provided that it is less than a distance \( d_{\text{tol}} \), from its ground-truth position. In other words, the constraint only comes into play when the vertex is sufficiently close to its true position to avoid imposing spurious constraints at the beginning of the optimization. The loss term also penalizes triangle-triangle intersections between the body and the garment, which could happen when two neighboring garment vertices are close to the same body vertex. Unlike in [14], we do not force the garment vertex to be within a predefined distance of the body because, in some cases, garment vertices can legitimately be far from it.

**Normal Term.** We write \( \mathcal{L}_{\text{norm}} \) as
\[
\frac{1}{N_F} \sum_{i=1}^{N_F} \left( 1 - (F^G_i)^T F^P_i \right)^2,
\]
to penalize the angle difference between the ground-truth and predicted facet normals. Here, \( N_F, F^G \) and \( F^P \) are the number of facets, the normal vector of the \( i^{th} \) ground-truth facet and of the corresponding predicted one, respectively.

**Bending Term.** We take \( \mathcal{L}_{\text{bend}} \) to be
\[
\frac{1}{|N_2|} \sum_{(i,k) \in N_2} \| G^P_k - G^P_i \| - \| G^P_i - G^G_i \|, \quad (6)
\]
to emulate the bending constraint of NvCloth [31], the PBS method we use, which is an approximation of the one in [30]. Here, \( N_2 \) denotes a set of pairs of vertices connected by a shortest path of two edges. This term helps preserve the distance between neighboring vertices of a given vertex, as shown in Fig. 6b. Although it is theoretically possible to consider larger neighborhoods, the number of pairs would grow exponentially.

**3.4. Implementation Details**

To apply the skinning method of [20], we compute the skinning weight matrix \( W \) using Blender [5] given the pose information of the garment mesh. The garment stream employs 6 residual blocks depicted in Fig. 4 following the common practice of ResNet [17]. In each block, we adopt the mesh convolution layer proposed in [40], which uses 1-ring neighbors to learn patch-wise features at each convolution layer. As the mesh convolution operators rely on trainable parameters to weigh the contribution of neighbors, we always concatenate the input vertex 3D locations to their input vectors so that the network can learn topology-dependent convolutions. While using the exact PointNet architecture of [36] in the body stream, we observed that all point-wise body features converged to the same feature vector, which seems to be due to ReLU saturation. To prevent this, we use leaky ReLUs with a slope of 0.1 and add a skip connection from the output of the first Spatial Transformer Network (STN) to the input of the second MLP block. To use the body features in the garment stream as shown in Fig. 3, the 512-dimensional global body features are repeated for each garment vertex. For the local body pooling depicted by Fig. 5, we downscale the 3D body points along with their point-wise features by a factor 10. This is done by average pooling applied to the point-wise body features with a 16 neighborhood size. For the local max-pooling of body features in Fig. 5, the number of neighbors is 15.
To increase the effectiveness of the interpenetration term in Eq. (4), each matched body point $B_i$ is extended in the direction of its normal vector by 20% of average edge length of the mesh to ensure that penetrations are well-penalized, and the tolerance parameter $d_{tol}$ is set to 0.05 for both our dataset and that of [41]. Additional details are given in the supplementary material. To train the network, we use the PyTorch [32] implementation of the Adam optimizer [23] with a learning rate of 0.001. In all the experiments reported in the following section, we empirically set the weights of Eq. 2, $\lambda_{\text{normal}}$, $\lambda_{\text{pen}}$ and $\lambda_{\text{bend}}$ to 0.3, 1.0 and 0.5.

4. Experiments

In this section, we evaluate the performance of our framework both qualitatively and quantitatively. We first introduce the evaluation metrics we use, and conduct extensive experiments on our dataset to validate our architecture design. Then, we compare our method against the only state-of-the-art method [41] for which the training and testing data is publicly available. Finally, we perform an ablation study to demonstrate the impact of our loss terms.

4.1. Evaluation Metrics

We introduce the following two quality measures:

$$\mathcal{E}_{\text{dist}} = \frac{1}{N} \sum_{i=1}^{N} \| G_i^G - G_i^P \| , \quad (7)$$

$$\mathcal{E}_{\text{norm}} = \frac{1}{N_F} \sum_{i=1}^{N_F} \arccos \left( \frac{(F_i^G)^T F_i^P}{\| F_i^G \| \| F_i^P \|} \right) . \quad (8)$$

$\mathcal{E}_{\text{dist}}$ is the average vertex-to-vertex distance between the predicted mesh and the ground-truth one, while $\mathcal{E}_{\text{norm}}$ is the average angular deviation of the predicted facet normals to the ground-truth ones. As discussed in [7], the latter is important because the normals are key to the appearance of the rendered garment.

4.2. Analysis on our Dataset

We created a large dataset featuring various poses and body shapes. We first explain how we built it and then test various aspects of our framework on it.

Dataset Creation. We used the Nvidia physics-based simulator NvCloth [31] to fit a T-shirt, a sweater and a pair of jeans represented by 3D triangulated meshes with 10k vertices on synthetic bodies generated by the SMPL body model [26], represented as meshes with 6890 vertices. To incorporate a variety of poses, we animated the SMPL bodies using the yoga, dance and walking motions from the CMU mocap [8] dataset. The training, validation and test sets consist of 500, 20 and 80 bodies, respectively. The T-shirt, the sweater and the jeans have, on average, 40, 23 and 31 poses, respectively. To guarantee repeatability for similar body shapes and poses, each simulation was performed by starting from the initial pose of the input garment.

Quantitative Results. Recall from Section 3.2 that we implemented two variants of our network, GarNet-Global that relies solely on global body-features and GarNet-Local that also exploits local body-features by performing nearest neighbor pooling as shown in Fig. 5. As the third variant, we implemented a simplified version of GarNet-Global in which we removed the mesh convolution layers that produce patch-wise garment features. It therefore performs only point-wise operations (i.e. $1 \times 1$ conv.) and max-pooling layer, and we dub it GarNet-Naive, which can also be interpreted as a two-stream PointNet [36] with extra skip connections. We also compare against the garment warped by dual quaternion skinning (DQS) [20], which only depends on the body pose.

![Figure 7: Average precision curves for the vertex distance and the facet normal angle error.](image)

In Table 1, we report our results in terms of the $\mathcal{E}_{\text{dist}}$ and $\mathcal{E}_{\text{norm}}$ of Section 4.1. In Fig. 7, we plot the corresponding average precision curves for T-shirts, jeans and sweaters. The average precision is the percentage of vertices/normal of all test samples whose error is below a given threshold. GarNet-Naive does worse than the two others, which
underlines the importance of patch-wise garment features. GarNet-Global and GarNet-Local yield comparable results with an overall advantage to GarNet-Local. Finally, in Table 2, we report the computation times of our networks and the employed PBS software. Note that both variants of our approach yield a 100× speedup.

Tests on unseen poses. The T-shirt dataset is split such that 50% (25%) of the poses (uniformly sampled within each motion) are in the training set; the rest are in the test set. The distance and angle errors increase to 1.16 (1.68) cm and 9.71 (11.88)°. Since our poses are carefully sampled to ensure diversity, the performance on the splits above indicate generalization ability.

Qualitative Results. Fig. 8 depicts the results of the GarNet-Local, GarNet-Global and GarNet-Naive architectures. The GarNet-Global results are visually similar to the GarNet-Local ones on the printed page; however, GarNet-Global produces a visible gap between the body and the garment while the garment draped by GarNet-Local is more similar to the PBS one. GarNet-Naive generates some clearly visible artifacts, such as spurious wrinkles near the right shoulder. By contrast, the predictions of GarNet-Local closely match those of the PBS method while being much faster. We provide further evidence of this in Fig. 9 for three different garment types. Additional visual results are provided in the supplementary material.

4.3. Results on the Dataset of [41]

As discussed in Section 2, [41] is the only non-PBS method that addresses a problem similar to ours and for which the data is publicly available. Specifically, the main focus of [41] is to drape a garment on several body shapes for different garment sewing patterns. Their dataset contains 7000 samples consisting of a body shape in the T-pose, sewing parameters, and the fitted garment. Hence, the inputs to the network are the body shape and the garment sewing parameters. To use GarNet for this purpose, we take one of the fitted garments from the training set to be the template input to our network, and concatenate the sewing parameters to each vertex feature before feeding them to the MLP layers of our network. The modified architecture is described in more detail in the supplementary material. We use the same training (95%) and test (5%) splits as in [41] and compare our results with theirs in terms of the normalized $L^2$ distance percentage, that is, $100 \times \|G^G - G^P\|/\|G^G\|$, where $G^G$ and $G^P$ are the vectorized ground-truth and predicted vertex locations normalized to the range $[0, 1]$. We use this metric here because it is the one reported in [41]. As evidenced by Table 3, our framework generalizes to making use of garment parameters, such as sewing patterns, and significantly outperforms the state-of-the-art one of [41].

Ablation study. We conducted an ablation study on the dataset of [41] to highlight the influence of the different terms in our loss function. We trained the network by individually removing the penetration, bending, and normal term. We also report results without both the normal and bending terms. As shown in Table 4, using the normal and bending terms significantly improves the angle accuracy. This is depicted in Fig. 10 where the normal term helps remove the spurious wrinkles. While turning off the penetration term has limited impact on the quantitative results, it causes more severe interpenetration, as shown in Fig. 10.

5. Conclusion

In this work, we have introduced a new two-stream network architecture that can drape a 3D garment shape on different target bodies in many different poses, while running 100 times faster than a physics-based simulator. Its
key elements are an approach to jointly exploiting body and garment features and a loss function that promotes the satisfaction of physical constraints. By also taking as input different garment sewing patterns, our method generalizes to accurately draping different styles of garments.

Our model can drape the garment shapes to within 1 cm average distance from those of a PBS method while limiting interpenetrations and other artifacts. However, it still has a tendency to remove high-frequency details, as also observed in [14, 38], because regression tends to smooth. In future work, we will explore conditional Generative Adversarial Networks [19] to add subtle wrinkles to further increase the realism of our reconstructions, as in [24]. Another avenue of research we intend to investigate is mesoscopic-scale augmentation, as was done in [4], to enhance the reconstructed faces.
References


Tuanfeng Y. Wang, Duygu Ceylan, Jovan Popovic, and Niloy Jyoti Mitra. Learning a shared shape space for multimodal garment design. In ACM SIGGRAPH Asia, 2018.


