GarNet++: Improving Fast and Accurate Static 3D Cloth Draping by Curvature Loss

Erhan Gundogdu, Victor Constantin, Shaifali Parashar, Amrollah Seifoddini, Minh Dang, Mathieu Salzmann, Pascal Fua

GarNet++: Improving Fast and Accurate Static 3D Cloth Draping by Curvature Loss

Erhan Gundogdu, Victor Constantin, Shaifali Parashar,
Amrollah Seifoddini, Minh Dang, Mathieu Salzmann, and Pascal Fua

Abstract—In this paper, we tackle the problem of static 3D cloth draping on virtual human bodies. We introduce a two-stream deep network model that produces a visually plausible draping of a template cloth on virtual 3D bodies by extracting features from both the body and garment shapes. Our network learns to mimic a Physics-Based Simulation (PBS) method while requiring two orders of magnitude less computation time. To train the network, we introduce loss terms inspired by PBS to produce plausible results and make the model collision-aware. To increase the details of the draped garment, we introduce two loss functions that penalize the difference between the curvature of the predicted cloth and PBS. Particularly, we study the impact of mean curvature normal and a novel detail-preserving loss both qualitatively and quantitatively. Our new curvature loss computes the local covariance matrices of the 3D points, and compares the Rayleigh quotients of the prediction and PBS. This leads to more details while performing favorably or comparably against the loss that considers mean curvature normal vectors in the 3D triangulated meshes. We validate our framework on four garment types for various body shapes and poses. Finally, we achieve superior performance against a recently proposed data-driven method.

1 Introduction

Shopping for clothes is time-consuming due to the amount of time customers spend trying to determine if their purchases will fit. Online shopping can streamline this process, but only if it provides realistic and easy-to-use simulations that enable potential buyers to view a draped version of the garment on a 3D model of their own body. Ideally, this model should rely on a simple parametrization [30] that can be obtained from a few images, as in [7]. Recent Physics-Based Simulation (PBS) software [14], [37], [38], [48] can deliver highly realistic draping results on virtual 3D bodies, but at the cost of much computation, which makes it unsuitable for real-time and web-based applications. We propose to train a deep neural network to produce 3D draping results of the similar quality to PBS ones but much faster, as shown in Fig. 1.

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 cloth and body. 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 [43] 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 the simpler version of our approach depicted by Fig. 2a, the local body features are only used *implicitly* to compute the global ones. In the more sophisticated implementation depicted by Fig. 2b, 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 sub-network. We will see that this second version performs better than the simpler one, which indicates that local feature pooling is valuable.

By incorporating appropriate loss terms in the objective function that we minimize during training, we avoid the need for extra post-processing steps to minimize cloth-body interpenetration and undue tightness that PBS tools [14], [37], [38], [48], optimization-based [9] and data-driven [17], [54] methods often require at inference time. In addition to terms that discourage interpenetration, our loss function includes terms that favor surface predictions whose curvatures exhibit the same statistics as those of target shapes, thereby helping the network to infer 3D surfaces with local deformations similar to those of real clothes. Furthermore, because it relies only on convolution and pooling operations, our approach naturally scales to point clouds of arbitrary resolution. This is in contrast to earlier data-driven methods [17], [47], [54] that rely on low-dimensional subspaces whose size would typically have to grow to model that level of detail, thus adversely affecting their memory requirements.

Our main contribution is therefore a novel architecture for garment simulation that drapes clothes on virtual 3D bodies in real-time by properly modeling the body and garment interaction and was first reported in conference proceedings [18]. To further increase the level of details it delivers, we have since incorporated a novel curvature term in our training loss. It relies on Rayleigh Quotient [50] bounds to approximate eigenvalues in a manner that can be backpropagated and yields more realistic results than when using more traditional differentiable approximations of curvature [49].

We ran extensive experiments on a dataset that comprises a pair of jeans, a t-shirt, a dress, and a sweater draped over 600 dif-

- *E. Gundogdu, V. Constantin, S. Parashar, M. Salzmann and P. Fua were with the Computer Vision Laboratory (CVLab), Ecole Polytechnique Federale de Lausanne, Switzerland. E-mail: erhanguendogdu@gmail.com (work done when he was with CVLab), shaifali.parashar@gmail.com, {victor.constantin,mathieu.salzmann,pascal.fua}@epfl.ch*
- *A. Seifoddini and M. Dang were with Fision Technologies. E-mail: {as, minh.dang}@fision-technologies.com*

This work was supported in part under a grant from Innosuisse, the Swiss Innovation Agency.

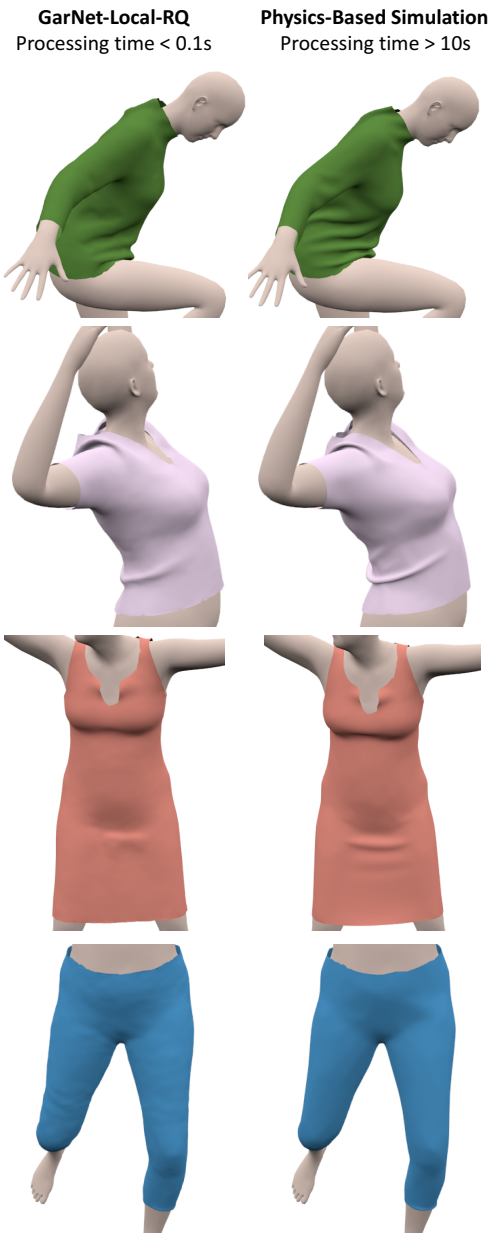


Fig. 1: Draping a sweater, a T-shirt, a dress and a pairs of jeans. Our method produces results as plausible as those of a PBS method, but runs 100× faster.

ferent bodies from the SMPL model [30] in various poses and that we will make publicly available at https://www.epfl.ch/labs/cvlab/. They show that our network can effectively handle many body poses and shapes. Furthermore, we can exploit additional information, such as cutting patterns, when available. To demonstrate this, we use the recently-published dataset of [54], which contains different garment types with varying cutting patterns and show that our method outperforms the most recent state-of-the-art one of [54], which adresses the same problem as our method does.

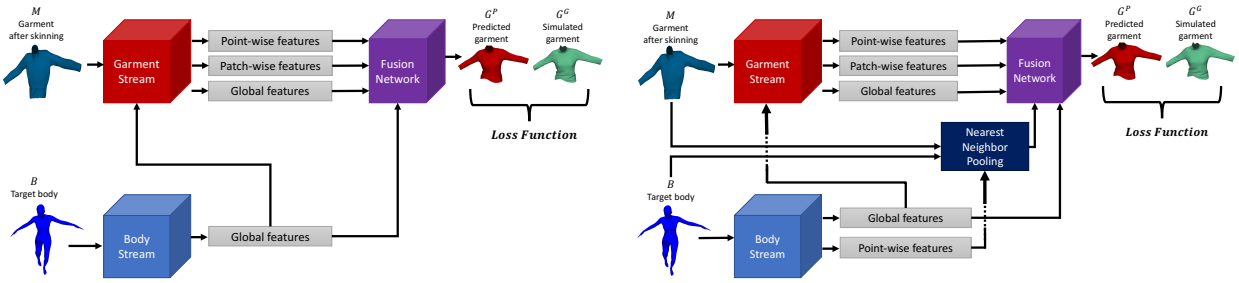
2 Related Work

Many professional tools can model cloth deformations realistically using Physics-Based Simulation (PBS) [14], [37], [38], [48]. However, they are computationally expensive, which precludes real-time use. Some of these can operate in near real-time. For example, the algorithm of [51] achieves more than 10 fps with high-resolution meshes using an incremental approach in motion sequences. By contrast, static cloth draping over a body in an arbitrary pose remains too slow for real-time performance. The speed of static cloth draping is directly proportional to the distance between the reference pose and the one for which the draping is required. For instance, it would take about 1s for a simulation operating at 10 fps to output the simulation result when the frame distance between the target pose and the initial one is only 10 frames. Furthermore, manual parameter tuning is often necessary and cumbersome. 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 [26] 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 [34] proposes a data-driven estimation of the physical parameters of the cloth material, while [25] constructs a finite motion graph for detailed cloth effects. In [22], 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 [12]. [16] also models the relation between 2D cloth deformations and corresponding bodies in a low-dimensional space. [17] 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. [47] trains an MLP and an RNN to model the cloth deformations by decomposing them as static and dynamic wrinkles. Both [17] and [47], however, require an *a posteriori* refinement to prevent cloth-body interpenetration. In a recent approach, [54] relies on a deep encoder-decoder model to create a joint representation for bodies, garment sewing patterns, 2D sketches and garment shapes. This defines a mapping between any pair of such entities, for example body-garment shape. However, it relies on a Principal Component Analysis (PCA) representation of the garment shape, thus reducing the accuracy. In contrast to [54], our method operates directly on the body and garment meshes, removing the need for such a limiting representation. We will show that our predictions are more accurate as a result.

Cloth fitting has been performed using 4D data scans as in [28], [41]. In [41], [57], 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 [41], we aim not only to obtain visually plausible results but also to emulate PBS for cloth fitting. In [28], 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 [24].

Image/Video-Based Approaches. Apart from methods requiring 3D scans [63], depth data [61], [62] or markers [31], there is a huge amount of literature on image/video-based reconstruction of



(a) **GarNet-Global** (b) **GarNet-Local**

Fig. 2: **Two versions of our GarNet.** Both take as input a target body and the garment mesh roughly aligned with the body pose by using [24]. **GarNet-Global**: We fuse the global body features with the garment features both early and late. **GarNet-Local**: In addition, we use a nearest neighbor approach to pooling local body features and feed the result to the fusion network whose job is to combine the body and garment features.

3D human shapes [1], [2], [3], [5], [15], [19], [46], [56], [64], and deformable 3D surfaces [4], [11], [42], to only quote some of the most recent papers. Moreover, conditional generative models [20], [29] are also proposed to predict clothed human bodies. Some of these include cloth modeling but none of them directly focus on mimicking PBS and cloth draping. By contrast, our method learns to drape a 3D cloth on different 3D target bodies in a visually plausible manner by incorporating notions of physical draping, such as interpenetration, bending and curvature.

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 [43], [44] and meshes [52]. PointNet [43], [44] 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 [60], unsupervised representation learning [58], 3D descriptor matching [13], and finding 2D correspondences [59]. 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 representative power, the point-wise operations in PointNet [43] are not sufficient to produce visually plausible garment fitting results, as we experimentally demonstrate via 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 [43], can produce local features, such as those of [8], [32], [35] that rely on hand-crafted patch operators. FeastNet [52] 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. Similarly to [52], 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 [43], [44] or 3D shape correspondence [8], [32], [35], [52], 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 novel two-stream architecture to account for both shapes.

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 [24] that produces a rough initial garment shape that depends only on body pose. In this section, we introduce two variants of our **GarNet** deep neural network that refine this initial shape and produce the final garment. Fig. 2 depicts these two variants.

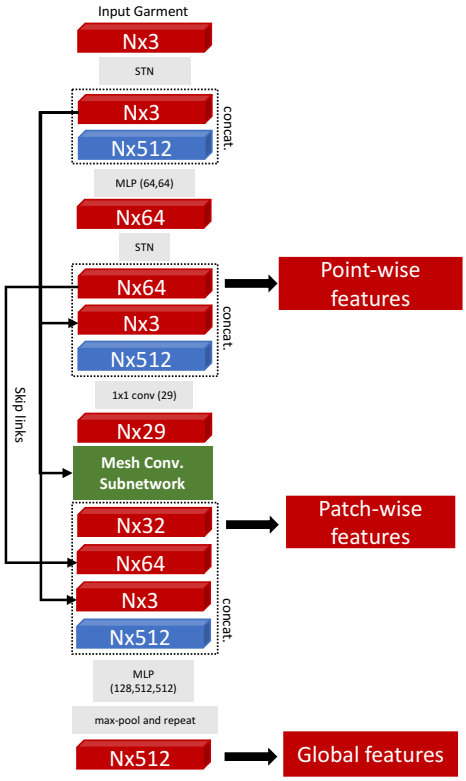


Fig. 3: **Garment branch of our network.** The grey boxes and the numbers in parenthesis denote network layers and their output channel dimensions. Red and blue ones represent garment and global body features, respectively. The green box is the mesh convolution subnetwork and depicted in more detail in Fig. 4. STN stands for a Spatial Transformer Network used in PointNet [43]. MLP blocks are shared by all N points.

3.1 Problem Formulation

Let $\mathcal{M}^0$ be the template garment mesh in the rest pose and let $\mathcal{M} = \text{dqs}(\mathcal{M}^0, \mathcal{B}, \mathcal{J}^0_{\mathcal{M}}, \mathcal{J}_{\mathcal{B}}, \mathcal{W})$ be the garment after skinning to the body $\mathcal{B}$, also modeled as a mesh, by the method [24]. Here, $\mathcal{J}^0_{\mathcal{M}}$ and $\mathcal{J}_{\mathcal{B}}$ are the joints of $\mathcal{M}^0$ and $\mathcal{B}$, respectively. $\mathcal{W}$ is

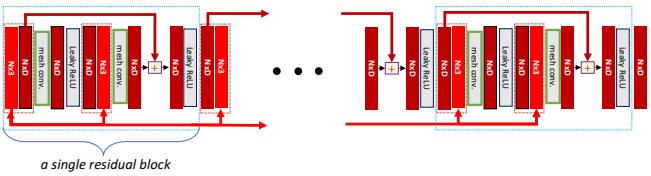


Fig. 4: **Mesh convolution subnetwork.** The residual block is repeated 6 times. Dashed red rectangles indicate channel-wise concatenation. The $N \times 3$-dimensional tensors contain the 3D vertex locations of the input garment, which are passed at different stages via skip connections.

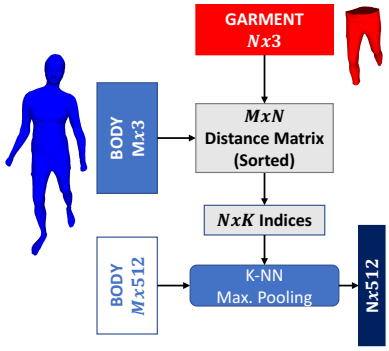


Fig. 5: K **nearest neighbor pooling in Fig. 2b.** We compute the K nearest neighbor body vertices of each garment vertex and max-pool their local features.

the skinning weight matrix for $\mathcal{M}^0$. Let f_θ be the network with weights θ chosen so that the predicted garment $\mathcal{G}^P$ given $\mathcal{M}$ and $\mathcal{B}$ is as close as possible to the ground-truth shape $\mathcal{G}^G$. We denote the i^{th} vertex of $\mathcal{M}$, $\mathcal{B}$, $\mathcal{G}^G$ and $\mathcal{G}^P$ by $\mathbf{M}_i$, $\mathbf{B}_i$, $\mathbf{G}_i^G$ and $\mathbf{G}_i^P \in \mathbb{R}^3$, respectively. Finally, let N be the number of vertices in $\mathcal{M}$, $\mathcal{G}^G$ and $\mathcal{G}^P$.

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

$$\mathcal{T}^P = f_\theta(\mathcal{M}, \mathcal{B}) \approx \mathcal{T}^G \ , \tag{1}$$

where $\mathcal{T}^P$ and $\mathcal{T}^G$ correspond to translation vectors from the skinned garment $\mathcal{M}$ to the predicted and ground-truth mesh, respectively, that is $\mathbf{G}_i^P - \mathbf{M}_i$ and $\mathbf{G}_i^G - \mathbf{M}_i$. Therefore, the final shape of the garment 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 $f_\theta(\mathcal{M}, \mathcal{B})$. The *body stream* takes as input the 3D point cloud representing the body while the *garment stream* takes as input the triangulated 3D mesh of the garment. 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 more detail below.

Body Stream. The *body stream* processes the body $\mathcal{B}$ in a manner similar to that of PointNet [43] (see Sec. 3.5 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 *garment stream* takes as input the warped garment $\mathcal{M}$ and the global body features extracted by the body stream to also compute the 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*, by using mesh convolution operations [52] that account for the local neighborhood of each garment vertex. 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 the purple box in Fig. 2. It consists of four MLP blocks shared by all the points, as done in the segmentation network of PointNet [43]. The final MLP block outputs the 3D translations $\mathcal{T}^P$ of Eq. (1) from the warped garment shape $\mathcal{M}$.

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 $\mathcal{M}$ to $\mathcal{B}$ and feeds them into the fusion network, along with the body and garment features. This additional 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 $\mathcal{L}(\mathcal{G}^G, \mathcal{G}^P, \mathcal{B}, \mathcal{M})$. We designed it to reduce the distance of the prediction $\mathcal{G}^P$ to the ground truth $\mathcal{G}^G$ while also incorporating regularization terms derived from physical constraints. We write

$$\mathcal{L} = \lambda_p \mathcal{L}_p + \lambda_{curv} \mathcal{L}_{curv} \ , \tag{2}$$

where L_{curv} is the curvature loss that considers our proposed curvature or mean curvature normals. This will be detailed in Sec. 3.4. In the equation above, $\mathcal{L}_p$ penalizes the descrepancy

between vertices of PBS and network predictions either independently or pair-wise using the following individual terms:

$$\mathcal{L}_p = \mathcal{L}_{vert} + \lambda_{pen}\mathcal{L}_{pen} + \lambda_{norm}\mathcal{L}_{norm} + \lambda_{bend}\mathcal{L}_{bend} \ . \quad (3)$$

Here, λ_{pen}, λ_{norm}, and λ_{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 $\mathcal{L}_{vert}$ to be the average L^2 distance between the vertices of $\mathcal{G}^G$ and $\mathcal{G}^P$,

$$\frac{1}{N}\sum_{i=1}^{N}\left\|\mathbf{G}_i^G - \mathbf{G}_i^P\right\|^2, \quad (4)$$

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 $\mathcal{C}(\mathcal{B}, \mathcal{G}^P)$, a set of garment-body index pairs. We write $\mathcal{L}_{pen}$ as

$$\sum_{\{i,j\}\in\mathcal{C}(\mathcal{B},\mathcal{G}^P)} \mathbb{1}_{\{\|\mathbf{G}_j^P - \mathbf{G}_j^G\| < d_{tol}\}} ReLU(-\mathbf{N}_{B_i}^T(\mathbf{G}_j^P - \mathbf{B}_i))/N, \quad (5)$$

to penalize the presence of garment vertices inside the body. Here, $\mathbf{N}_{B_i}$ is the normal vector at the i^{th} body vertex, as depicted by Fig 6a. This formulation penalizes the garment vertex G_j^P for not being on the green subspace of its corresponding body vertex B_i, provided that it is less than a distance d_{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 traingle-triangle intesections between the body and the garment, which could happen when two neighboring garment vertices are close to the same body vertex. Unlike in [17], 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, *e.g.* in the lower parts of a dress or in wrinkles for most garment types.

Normal Term. We write $\mathcal{L}_{norm}$ as

$$\frac{1}{N_F}\sum_{i=1}^{N_F}\left(1 - \left(\mathbf{F}_i^G\right)^T\mathbf{F}_i^P\right)^2, \quad (6)$$

to penalize the angle difference between the ground-truth and predicted facet normals. Here, N_F, $\mathbf{F}_i^G$ and $\mathbf{F}_i^P$ 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 $\mathcal{L}_{bend}$ to be

$$\frac{1}{|\mathcal{N}_2|}\sum_{\{i,k\}\in\mathcal{N}_2} \left|\ \|\mathbf{G}_i^P - \mathbf{G}_k^P\| - \|\mathbf{G}_i^G - \mathbf{G}_k^G\|\ \right|, \quad (7)$$

to emulate the bending constraint of NvCloth [37], the PBS method we use, which is an approximation of the one in [36]. Here, $\mathcal{N}_2$ 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.

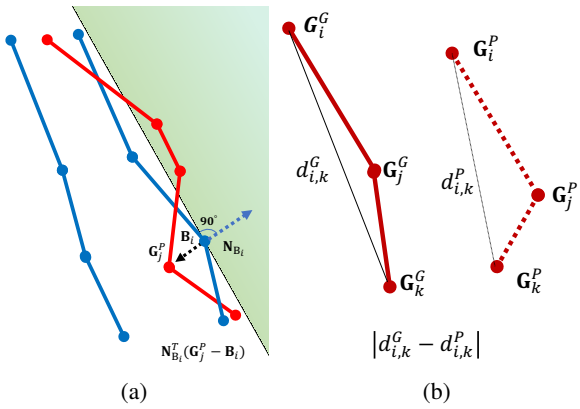


Fig. 6: **Interpenetration and Bending loss terms. (a)** The interpenetration term L_{pen} penalizes a garment vertex $\mathbf{G}_j^P$ for being on the wrong side of the corresponding body point $\mathbf{B}_i$. **(b)** The bending term L_{bend} penalizes the distance between two neighbors of $\mathbf{G}_j^P$ to differ from that in the ground truth.

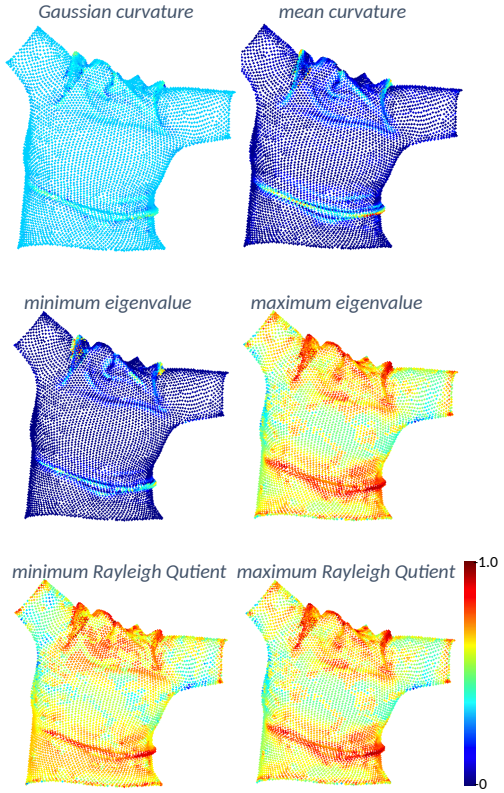


Fig. 7: **Visualizing different kinds of curvature.** All of the metrics above are normalized between 0 and 1.

3.4 Increasing the Level of Detail

As will be shown in Sec. 4, the network trained by minimizing the loss function of Eq. (3) delivers visually plausible draping results. However, they tend to be smoother than the PBS ones, especially in places where wrinkles are prominent. There are two reasons for this. First, even small body shape and pose dissimilarities can make the simulation engine yield visibly different results. Such significant variations in output stemming from small input changes are hard for the network to learn as the terms in Eq. (3) might not penalize smooth predictions. Second, the loss terms in Eq. (3) all account for the predicted 3D point locations either independently

from each other or in pairs. In other words, no neighborhood information is used to produce realistic local statistics.

In this section, we add to the loss function of Eq. (3) curvature terms designed to remedy this by forcing the local statistics of the predicted surface to be similar to that of the ground truth. We now discuss several ways to estimate curvature and argue that the one based on the Rayleigh quotient [23] is the most appropriate one for our purpose. We then present our approach to incorporating this quotient into a loss term.

3.4.1 Curvature Metrics

The curvature of a 3D surface represented by a triangulation can be estimated in several ways. Fig. 7 depicts four of them, which we discuss in more detail below.

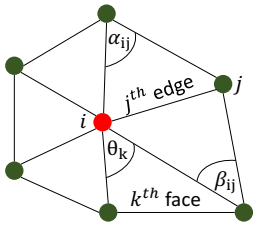


Fig. 8: Neighborhood convention for the Gaussian and mean curvature.

Gaussian curvature: We define the discrete Gaussian curvature as in [33]. That is, we write it as

$$\kappa^i_{GC} = \frac{1}{A^i_{Mixed}}\big(2\pi - \sum_{k\in\mathcal{N}^F_i} \theta_k\big), \tag{8}$$

where θ_k is the angle of the k^{th} facet, $\mathcal{N}^F_i$ is the face neighborhood depicted by Fig. 8 and A^i_{Mixed} is the finite-volume area of vertex i approximating the continuous formula. Although the Gaussian curvature can be high in wrinkly areas, it can also vanish in the presence of ridge-like wrinkles, as shown in Fig. 7. This is because one of the the principal curvatures can be so small that the product of the two curvatures is small as well, even though the other principal curvature is large.

Mean Curvature Normal: Also as in [33], we take the mean curvature normal to be

$$\kappa^i_{MC} = \frac{1}{A^i_{Mixed}} \sum_{j\in\mathcal{N}^E_i} (cot(\alpha_{ij}) + cot(\beta_{ij}))(\mathbf{G}_j - \mathbf{G}_i)\ , \tag{9}$$

where $\mathbf{G}_i$ is the 3D vertex position and $\mathcal{N}^E_i$ is the set of vertices that share an edge with vertex i. In Fig. 8, α_{ij} and β_{ij} are the opposite angles of the j^{th} edge of vertex i. This computation only involves vertices within a one-ring neighborhood of the center vertex and ignores those from the larger neighborhood. Furthermore, the cotangents in Eq. (9) can produce arbitrarily high values for small or wide facet angles. Therefore including κ_{MC} in a loss function could result in numerical instabilities.

Eigen Curvature: As suggested in [40], eigenanalysis can be used for curvature estimation. To this end, one can compute a covariance matrix for each vertex given its local neighborhood $\mathcal{N}^K_{Eig}$ that comprises its K nearest neighbors. The minimum and maximum eigenvalues of these matrices convey the curvature information. For instance, the ratio of the minimum eigenvalue to the sum of the three is related to the deviation from the tangent plane [40]. More formally, we write the covariance matrix for vertex i as

$$\Sigma^K_i = \frac{1}{K} \sum_{j\in\mathcal{N}^K_{Eig}} (\mathbf{G}_j - \hat{\mathbf{G}}_i)(\mathbf{G}_j - \hat{\mathbf{G}}_i)^T\ , \tag{10}$$

where $\hat{\mathbf{G}}_i$ is the mean of the vertices neighboring the i^{th} one. Then, we compute the smallest and largest eigenvalues as σ^i_{min} and σ^i_{max}. For an almost planar surface, σ_{max} is significantly larger than σ_{min}. For ridge-like regions, σ_{max} is also larger than σ_{min}, but not as much as in the planar case. Finally, hemispherical regions have three large eigenvalues. Therefore, in each one of these three cases, the eigenvalue distribution is different, which makes them easily distinguishable. This may not be true at saddle points where σ_{max} and σ_{min} get closer to each other as in the hemispherical case. Fortunately, minimizing L_{vert} and L_{norm} tends to prevent this from occurring. We illustrate this in the supplementary material.

Even though they are closely related to the minimum and maximum local curvature, directly incorporating these eigenvalues in a loss term is problematic because eigenvalue decomposition of a 3×3 matrix in closed-form is numerically unstable, especially in the presence of flat surfaces whose covariance matrices have high condition numbers. This can be mitigated using an iterative approach to eigenvalue decomposition [55] but results in a substantial increase in complexity.

Rayleigh quotient (*RQ*) Curvature: To overcome the limitations of the three above-mentioned curvature metrics, we propose to use the Rayleigh quotient [23] of the local neighborhood instead. We write it as

$$RQ(\Sigma^K_i, \mathbf{G}_j) = \frac{\mathbf{G}^T_j \Sigma^K_i \mathbf{G}_j}{\mathbf{G}^T_j \mathbf{G}_j}\ , \tag{11}$$

where we assume all the 3D vertex locations $\mathbf{G}_j$ to have been normalized so that they are zero-mean, and Σ^K_i is the covariance matrix of Eq. (10). As Σ^K_i is positive semi-definite, the inequality $\sigma_{min} \leq RQ(\Sigma^K_i, \mathbf{G}_j) \leq \sigma_{max}$, where σ_{min} and σ_{max} are the smallest and largest eigenvalues of Σ^K_i, is true. Computing *RQ* in Eq. (11) does not require an eigendecomposition. Moreover, existing deep learning libraries can efficiently carry out this computation by using batch matrix multiplication, which also features closed-form gradients. Using the property in Eq. (11) [23], we therefore use the smallest and largest values of $RQ(\Sigma^K_i, G_j)$ for different choices of G_j as an estimate for these curvatures. That is, we write

$$\begin{aligned} RQ^{P_i}_{min} &= \min_{j\in\mathcal{N}^K_{Eig}} RQ(\Sigma^K_i, \mathbf{G}_j)\ , \\ RQ^{P_i}_{max} &= \min_{j\in\mathcal{N}^K_{Eig}} RQ(\Sigma^K_i, \mathbf{G}_j)\ , \end{aligned} \tag{12}$$

for a given vertex P_i of the mesh. Fig. 7 shows that the *RQ* metric differentiates the flat and curved regions better than the Gaussian one while it highlights particular details, such as the wrinkles around the shoulders more strongly than the mean and eigen curvature metrics. Please see the redness of wrinkly regions in Fig. 7. Moreover, both the eigen and *RQ* metrics can operate at different scales by varying K in Eq. (10) or Eq. (12). We will study the effectiveness of the *RQ* metric for different choices of K in Sec. 4.4.

3.4.2 Curvature Loss Terms

As discussed earlier, the Rayleigh quotient fluctuates between the minimum and maximum eigen curvatures, but there is no guarantee that its smallest and largest values are strictly equal to them. In practice, this is not an issue because we are more interested in the similarity between our network's predictions and the ground truth than in exact curvature estimates. Therefore, let $RQ_{min}^{P_i}$ and $RQ_{max}^{P_i}$ be the minimum and maximum values of Eq. (12) for vertex i of the predicted mesh and $RQ_{min}^{G_i}$ and $RQ_{max}^{G_i}$ the corresponding values for the ground truth. We write the curvature loss term as

$$\mathcal{L}_{RQ} = \frac{1}{N}\sum_{i=1}^{N}(RQ_{min}^{P_i} - RQ_{min}^{G_i})^2 + (RQ_{max}^{P_i} - RQ_{max}^{G_i})^2 \ . \tag{13}$$

It is minimized when the curvature statistics of both meshes are the same, which is what we are trying to achieve. We also tried using the ratio of RQ_{max} and RQ_{min}, which yielded similar results but made the training numerically less stable.

Note that the proposed curvature loss term's main purpose is not to align the predicted surface normal with that of the ground truth. This task is already carried out by the normal term of Eq. 6 that improves angular accuracy but without increasing the level of details. The latter is a function of how the vertices are distributed in their local neighborhood. This distribution is not well-captured by the normal loss as our experiments will show. This requires using larger neighborhoods, which is precisely what our RQ-based loss function does.

For comparison purposes, we also implemented a loss term based on the mean curvature normals instead of the Rayleigh quotient, similar to the one used in [53]. We take it to be

$$\mathcal{L}_{MC} = \frac{1}{N}\sum_{i=1}^{N}\left\|\kappa_{MC}^{P_i} - \kappa_{MC}^{G_i}\right\|^2 \ , \tag{14}$$

where κ_{MC} is the mean curvature normal vector of Eq. (9). Fig. 7 shows that the norm of the mean curvature normal is high around the ridge-like wrinkles. However, its region of interest per-vertex is limited to the one-ring neighborhood, and extension to larger neighborhoods would require re-triangulation and extra computation. By contrast, *RQ* and eigen curvatures can be easily adapted to larger regions by changing the neighborhood size.

As mentioned earlier, another drawback of the mean curvature is that the cotangent function used to compute κ_{MC} can return very large values, and special care must be taken when minimizing $\mathcal{L}_{MC}$ to prevent divergence. Concretely, the weight of the loss term based on mean curature normals should be altered when a high cotangent value is computed. In practice, when using the mean curvature normal computation, we observed instabilities for some garments during the training stage even though we tried *ad hoc* modifications in the loss function where the loss term is turned-off if it is higher than a predefined threshold. By contrast, our *RQ*-based loss term does not need such a careful supervision. To confirm that the difficulties we encountered when using the mean curvature loss are not due only to numerical instabilities, we conducted an ablation study in Sec. 4.4.2.

3.4.3 Overall Loss Terms for Fine-Tuning the Network

To further increase the level of details of the draped cloth on target bodies, we refine the weights of **GarNet-Local** by minimizing an extended loss that incorporates the curvature terms $\mathcal{L}_{RQ}$ and $\mathcal{L}_{MC}$ introduced in Sec. 3.4.2. Thus, we define two complete loss function $\mathcal{L}_{MC}$ and $\mathcal{L}_{RQ}$, which we write as

$$\mathcal{L}_{MC}^{tot} = \lambda_p\mathcal{L}_p + \lambda_{MC}\mathcal{L}_{MC}, \tag{15}$$

$$\mathcal{L}_{RQ}^{tot} = \lambda_p\mathcal{L}_p + \lambda_{RQ}^{8}\mathcal{L}_{RQ}^{8} + \lambda_{RQ}^{16}\mathcal{L}_{RQ}^{16} + \lambda_{RQ}^{32}\mathcal{L}_{RQ}^{32}, \tag{16}$$

where $\mathcal{L}_p$ is the loss function of Eq. (3). The superscripts denote the number of neighboring vertices K in Eq. (10) and the λs are scalar weights that we will specify in Sec. 3.5. For completeness, we also introduce a third complete loss function

$$\mathcal{L}_{MCRQ}^{tot} = \lambda_p\mathcal{L}_p + \lambda_{MC}\mathcal{L}_{MC} + \lambda_{RQ}^{8}\mathcal{L}_{RQ}^{8} + \lambda_{RQ}^{16}\mathcal{L}_{RQ}^{16} + \lambda_{RQ}^{32}\mathcal{L}_{RQ}^{32}$$

that incorporates both $\mathcal{L}_{MC}$ and $\mathcal{L}_{RQ}$.

3.5 Implementation Details

To apply the skinning method of [24], we compute the skinning weight matrix $\mathcal{W}$ using Blender [6] given the pose information of the garment mesh.

The garment stream employs six residual blocks depicted in Fig. 4 following the common practice of ResNet [21]. In each block, we adopt the mesh convolution layer proposed in [52], which uses one-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 [43] 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 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. (5), 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 [54].

To train the network, we use the PyTorch [39] implementation of the Adam optimizer [27] with a learning rate of 0.001. In all the experiments reported in the following section, we empirically set the weights of Eq. (3), λ_{normal}, λ_{pen} and λ_{bend} to 0.3, 1.0 and 0.5, respectively.

In Eq. (15) and (16), λ_{MC} and λ_p are set to 10.0 and 0.1, respectively. We fixed the *RQ* loss term weights so that all terms have roughly the same overall loss value at the beginning of the training. Hence, λ_{RQ}^{8}, λ_{RQ}^{16}, λ_{RQ}^{32} are set to 500, 50 and 10.0. As evidenced by the results of Table 6, including all three loss terms achieves the best trade-off between distance and angle error. In Fig. 13, we show that different features are captured at different neighborhood sizes.

4 EXPERIMENTS

In this section, we evaluate the performance of our framework both qualitatively and quantitatively. In Sec. 4.1, we first introduce the evaluation metrics we use, and conduct extensive experiments on our dataset to validate our architecture design. Then, in Sec. 4.2, we compare our method against the only state-of-the art method [54] for which the training and testing data is publicly available. We also perform an ablation study to demonstrate the impact of our loss terms. In Sec. 4.4, to show the effectiveness of using curvature, we present qualitative and quantitative comparison of the network predictions when using either our RQ-based measure, or the one based on mean curvature normals. Finally, in Sec. 4.5, we demonstrate that our approach generalizes to body shapes generated using models other than the SMPL model [30] we used in previous experiments.

4.1 Evaluation Metrics

We introduce the following two quality measures:

$$\mathcal{E}_{dist} = \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{G}_i^G - \mathbf{G}_i^P\| \,, \tag{17}$$

$$\mathcal{E}_{norm} = \frac{1}{N_F}\sum_{i=1}^{N_F} \arccos\left(\frac{(\mathbf{F}_i^G)^T\mathbf{F}_i^P}{\|\mathbf{F}_i^G\|\|\mathbf{F}_i^P\|}\right) \,. \tag{18}$$

$\mathcal{E}_{dist}$ is the average vertex-to-vertex distance between the predicted mesh and the ground-truth one, while $\mathcal{E}_{norm}$ is the average angular deviation of the predicted facet normals to the ground-truth ones. As discussed in [9], 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 [37] to fit a T-shirt, a sweater, a dress and a pair of jeans represented by 3D triangulated meshes with 10k vertices on synthetic bodies generated by the SMPL body model [30], 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 [10] dataset. Using this parameterization was a natural choice because it is easy to use and handles skinning well, which is probably why it currently is one of the most popular approach to human body modeling. However, we will show in Section 4.5 that our approach to animation does not depend on this.

The training, validation and test sets consist of 500, 20 and 80 bodies, respectively. The T-shirt, the sweater, the dress and the jeans have, on average, 40, 23, 26 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 Sec. 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 a 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 (*i.e.* 1×1 conv.) and max-pooling operations, and we dub it **GarNet-Naive**, which can also be interpreted as a two-stream PointNet [43] with extra skip connections. We also compare against the garment warped by dual quaternion skinning (DQS) [24], which only depends on the body pose.

In Table 1, we report our results in terms of the $\mathcal{E}_{dist}$ and $\mathcal{E}_{norm}$ of Sec. 4.1. In Fig. 9, we plot the corresponding average precision curves for T-shirts, jeans, dress and sweaters. The average precision is the percentage of vertices/normals 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**. We provide additional qualitative comparisons between **GarNet-Local** and **GarNet-Global** in the supplementary material. Finally, in Table 2, we report the computation times of our networks and of the employed PBS software. Note that both variants of our approach yield a 100 times speed-up.

Using SMPL Parameters as Input: Unlike that of [54], our network does not depend on a specific body model, such as SMPL. If we only used SMPL body parameters as input to our network, it would be impossible to model interpenetration explicitly during training as we do, which would result in severe interpenetrations at test time and would require post-processing. This can of course be remedied by computing the 3D locations of enough body surface points to write an interpenetration loss term. To test this, we implemented a variant of our approach that does this and that we will refer to as **GarNet-Global-Params**. For a fair comparison, we replaced the global body features of **GarNet-Global** by those of the SMPL shape and its pose parameters but kept on using our garment stream, with its carefully designed mesh convolution layers and skip connections. We compare our **GarNet-Local** approach against **GarNet-Global-Params** in Table 3. The results are similar but our unmodified method has the advantage of being generic and not limited to a specific body parameterization.

Qualitative Results: Fig. 10 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. 11 for the four different garment types.

4.3 Results on the Dataset of [54]

The work in [54] 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 [54] 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.

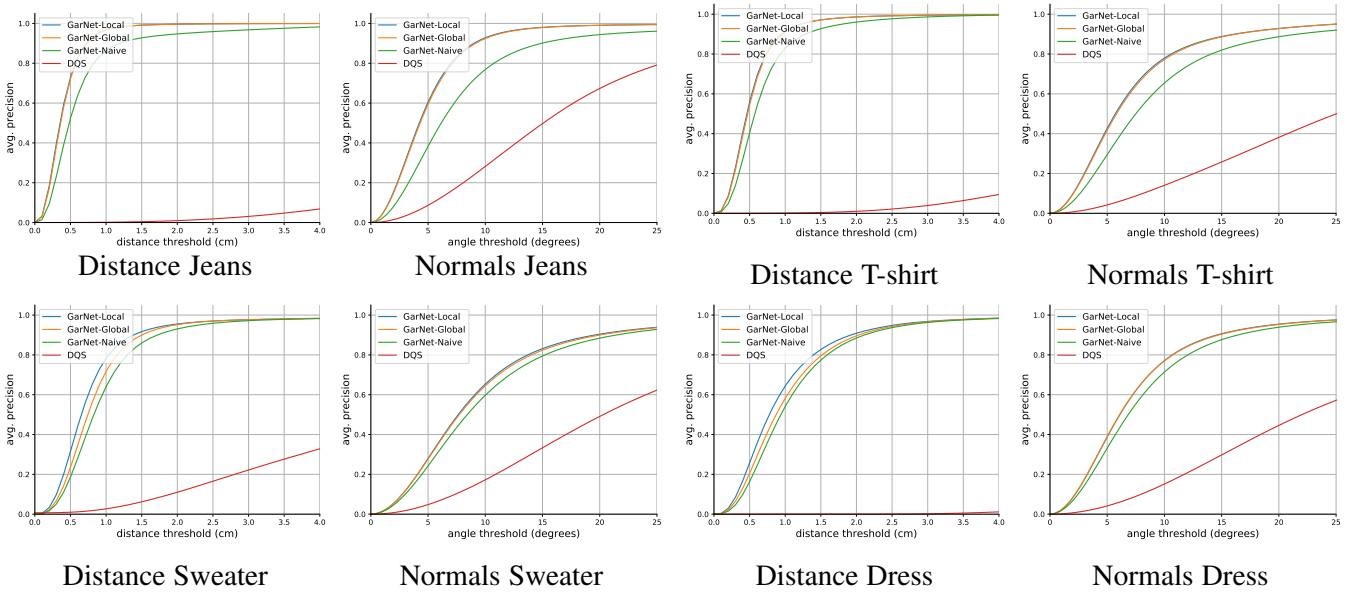


Fig. 9: Average precision curves for the vertex distance and the facet normal angle error.

	Jeans	**T-shirt**	**Sweater**	**Dress**
	$\mathcal{E}_{dist}/\mathcal{E}_{norm}$	$\mathcal{E}_{dist}/\mathcal{E}_{norm}$	$\mathcal{E}_{dist}/\mathcal{E}_{norm}$	$\mathcal{E}_{dist}/\mathcal{E}_{norm}$
GarNet-Local	**0.41** ($\pm$ 0.29) / **5.10** ($\pm$ 4.81)	**0.56** ($\pm$ 0.48) /**8.34** ($\pm$ 9.76)	**0.85** ($\pm$ 0.95) / **10.39** ($\pm$ 10.95)	**1.06** ($\pm$ 1.49) / **7.8** ($\pm$ 7.70)
GarNet-Global	0.42 ($\pm$ 0.29) / 5.18 ($\pm$ 4.85)	0.57 ($\pm$ 0.49) / 8.44 ($\pm$ 9.80)	0.93 ($\pm$ 0.93) / 10.54 ($\pm$ 11.04)	1.13 ($\pm$ 1.49) / 7.83 ($\pm$ 7.79)
GarNet-Naive	0.70 ($\pm$ 0.80) / 8.83 ($\pm$ 12.67)	0.73 ($\pm$ 0.63) / 11.28 ($\pm$ 14.05)	1.03 ($\pm$ 0.96) / 11.3 ($\pm$ 11.46)	1.19 ($\pm$ 1.49) / 8.92 ($\pm$ 9.61)
DQS	11.43 ($\pm$ 5.16) / 22.0 ($\pm$ 27.64)	9.98 ($\pm$ 4.49) / 30.74 ($\pm$ 24.90)	6.47 ($\pm$ 3.97) / 24.64 ($\pm$ 19.85)	14.79 ($\pm$ 4.52) / 28.21($\pm$ 22.27)

TABLE 1: **Architecture comparison:** Average distance in cm and face normal angle error in degrees between the PBS and predicted vertices in our dataset that uses SMPL body models. Numbers in paranthesis indicate standard deviation.

GarNet-Naive **GarNet-Global** **GarNet-Local** **PBS**

Fig. 10: **Comparison on the T-shirt**. **GarNet-Naive** produces artifacts near the shoulder while **GarNet-Local**, **GarNet-Global** and **PBS** yield similar results.

	GarNet-Local	**GarNet-Global**	**GarNet-Naive**	PBS	PBS†
time (ms)	68	59	0.2	$>$ 19000	$>$7200

TABLE 2: Comparison of the computation time. We used a single Nvidia TITAN X GPU for PBS and for our networks. In our case, forward propagation was done with a batch size of 16. PBS† stands for PBS computation excluding the time spent during the warping of template garment onto the target body pose.

	Jeans	**T-shirt**	**Sweater**	**Dress**
	$\mathcal{E}_{dist}/\mathcal{E}_{norm}$	$\mathcal{E}_{dist}/\mathcal{E}_{norm}$	$\mathcal{E}_{dist}/\mathcal{E}_{norm}$	$\mathcal{E}_{dist}/\mathcal{E}_{norm}$
GarNet-Local	0.41 / 5.10	0.56 / 8.34	0.85 / 10.39	1.06 / 7.80
GarNet-Global-Params	0.44 / 5.36	0.54 / 8.40	0.77 / 9.76	1.05 / 7.47

TABLE 3: Comparison between **GarNet-Local** and **GarNet-Global-Params** in our dataset that uses SMPL body models. The latter uses the ground truth SMPL shape and pose parameters.

We use the same training (95%) and test (5%) splits as in [54] and compare our results with theirs in terms of the normalized L^2 distance percentage, that is, $100 \times \frac{\|G^G - G^P\|}{\|G^G\|}$, 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 [54]. As evidenced by Table 4, our framework generalizes to making use of garment parameters, such as sewing patterns, and outperforms the state-of-the-art one of [54].

	GarNet-Local	**GarNet-Global**	[54]
Dist. %	**0.43**	0.48	3.01
Angle. $\sphericalangle$	**7.34**	7.75	N/A

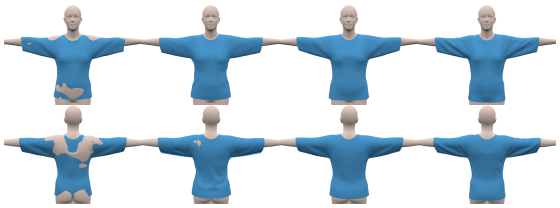
TABLE 4: Distance % and angle error on the shirt dataset of [54].

Loss Function	$\mathcal{E}_{dist}$	$\mathcal{E}_{normal}$
$\mathcal{L}_{vert} + \mathcal{L}_{pen}$	0.49	10.01
$\mathcal{L}_{vert} + \mathcal{L}_{pen} + \mathcal{L}_{bend}$	0.51	9.06
$\mathcal{L}_{vert} + \mathcal{L}_{norm} + \mathcal{L}_{bend}$	0.48	7.70
$\mathcal{L}_{vert} + \mathcal{L}_{pen} + \mathcal{L}_{norm}$	0.48	7.91
$\mathcal{L}_{vert} + \mathcal{L}_{pen} + \mathcal{L}_{norm} + \mathcal{L}_{bend}$	**0.42**	7.34
$\mathcal{L}_{vert} + \mathcal{L}_{pen} + \mathcal{L}_{norm} + \mathcal{L}_{bend} + \mathcal{L}_{RQ}$	0.45	**7.2**

TABLE 5: Ablation study on the dataset of [54] with **GarNet-Local**. The term $\mathcal{L}_{RQ}$ is our proposed loss term described in Sec. 3.4.

Ablation study. We also conducted an ablation study on the dataset of [54] to highlight the influence of the different terms in our loss function. To this end, we trained the network while removing the penetration term, the bending term and the normal term one at at time. We also report results without both the normal and bending terms. As shown in Table 5, using the normal and bending terms significantly improves the angle accuracy. This can be seen in Fig. 12 where the normal term helps remove spurious wrinkles. While turning off the penetration term only has only limited impact on the quantitative results, it causes substantial interpenetrations that can also be seen clearly in the figure.

Fig. 11: **GarNet-Local (top) vs PBS (bottom) results for several poses.** Note how similar they are, even though the former were computed in approx. 70ms instead of 20s. Our method successfully predicts the overall shape and details with intermediate frequency.



No penet. No norm. Full Loss PBS

Fig. 12: **Ablation study**. Reconstruction without some of the loss terms results in interpenetration (left) or different wrinkles at the back (second from left). By contrast, using the full loss yields a result very similar to the PBS one (two images on the right).

4.4 Fine-Tuning using the Curvature Losses

In this section, we show that fine-tuning the network using the curvature losses introduced in Sec. 3.4.3 increases the level of detail and plausibility of the predicted shapes.

	RQ 8-NN	*RQ* 16-NN	*RQ* 32-NN	*RQ* $\{8+16\}$-NN	*RQ* $\{8+16+32\}$-NN
$\mathcal{E}_{dist}$	**0.43**	0.44	0.47	0.43	0.45
$\mathcal{E}_{norm}$	7.51	7.44	7.42	7.38	**7.2**

TABLE 6: Distance and angle errors for different neighborhood combinations in our dataset that uses SMPL body models.

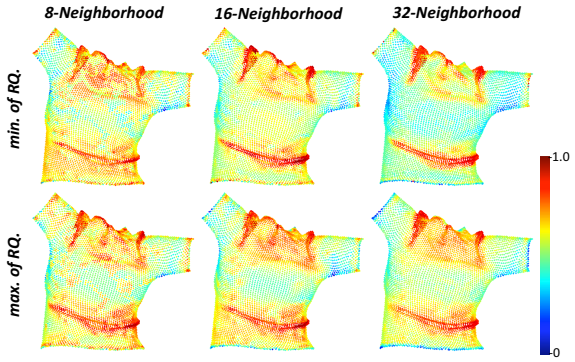


Fig. 13: **Varying the neighborhood in the *RQ* curvature metric.** The metrics above are normalized between 0 and 1.

4.4.1 Qualitative Results

Fig. 14 highlights the improved realism that our curvature loss terms deliver. Adding the *RQ* loss terms prevent oversmoothing and increases the level of detail. Furthermore, it removes noise from the predicted garment while preserving its wrinkles, as this can be observed from a qualitative comparison of the first and third column. Moreover, using **GarNet-Local-RQ** yields predictions that look more similar to the PBS ones than those using **GarNet-Local-MC**. We attribute this to the following two factors. First, the *RQ* loss, which is based on covariance matrices, accounts for the second-order statistics of the local neighborhood of each point, while the mean curvature one does not. Second, the *RQ* loss has a multi-scale impact in the local neighborhood, while the mean curvature only uses the one-ring neighborhood and does not penalize the absence of wrinkles covering a larger region than that neighborhood. We also compare **GarNet-Local** and **GarNet-Local-RQ** in Fig. 15 on the dataset of [54]. Using the *RQ* metric again helps produce more wrinkles while eliminating undesirable noise. Moreover, the second-order statistics in our proposed loss helps to mimic the statistics but not the exact 3D vertex locations since the simulation results might be inconsistent due to numerical instabilities for very similar body shapes and poses as also pointed out in [53].

Admittedly, wave-like patterns cannot be reconstructed solely by minimizing out curvature loss. However, it is not the only loss term. In theory, the normal loss term should penalize cases in which the network predicts a large wrinkle instead of higher frequency but smaller wrinkles. This can be seen in Fig. 15 where our network produces multiple consecutive wrinkles as it should.

4.4.2 Quantitative Results

	Jeans	**T-shirt**	**Sweater**	**Dress**
	$\mathcal{L}_{MC}/\mathcal{L}_{RQ}$	$\mathcal{L}_{MC}/\mathcal{L}_{RQ}$	$\mathcal{L}_{MC}/\mathcal{L}_{RQ}$	$\mathcal{L}_{MC}/\mathcal{L}_{RQ}$
GarNet-Local	0.29 / 0.070	0.14 / 0.23	0.13 / 0.82	0.06 / 0.52
GarNet-Local-RQ	0.29 / 0.070	0.13 / **0.09**	0.11 / **0.27**	0.04 / **0.15**
GarNet-Local-MC	**0.28** / 0.067	**0.11** / 0.26	**0.08** / 0.64	**0.03** / 0.46
GarNet-Local-MCRQ	0.30 / **0.066**	0.12 / 0.10	0.09 / 0.28	0.037 / 0.17

TABLE 7: Average loss values for *RQ* and mean curvature in our dataset that uses SMPL body models.

To quantify the improvement on the wrinkle details and the curvature deviation from the PBS results, in Table 7, we first report the average loss values for both the mean curvature normal $\mathcal{L}_{MC}$ and the sum of all *RQ* curvature terms in Eq. (16). We denote the results obtained using the mean curvature normal loss of Eq. (15)

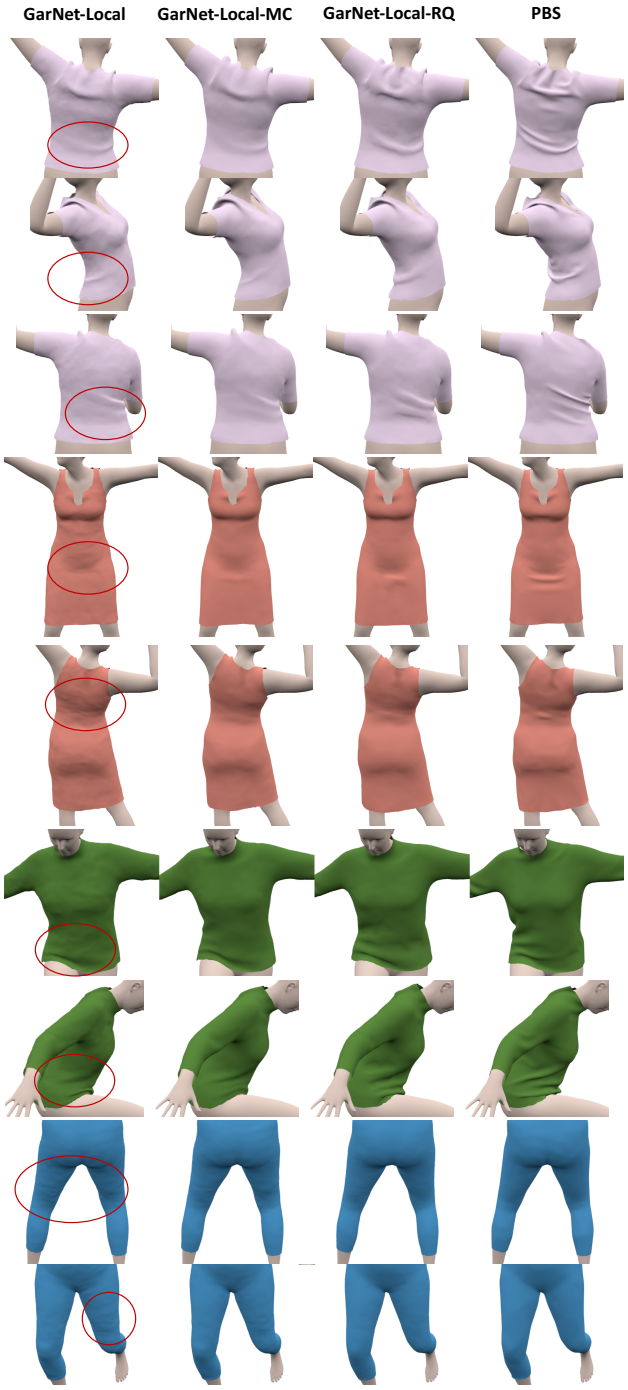


Fig. 14: **Using the curvature metrics to improve the level of detail on our dataset.** The red ellipses in the first column denote regions that are much smoother in the **GarNet-Local** prediction than in the **PBS** one. More realistic wrinkles are produced in those areas by **GarNet-Local-MC** and **GarNet-Local-RQ**.

as **GarNet-Local-MC**, those obtained using the Rayleigh quotient loss of Eq. (16) as **GarNet-Local-RQ**, and using both curvature losses as **GarNet-Local-MCRQ**, which we compare to those of **GarNet-Local**, that is, without fine-tuning. Introducing the curvature loss terms significantly improves over **GarNet-Local** in terms of average mean and *RQ* curvature loss values in the test dataset. Note that **GarNet-Local-MCRQ** and **GarNet-Local-RQ** most decrease the sum of these two loss values except for the jeans

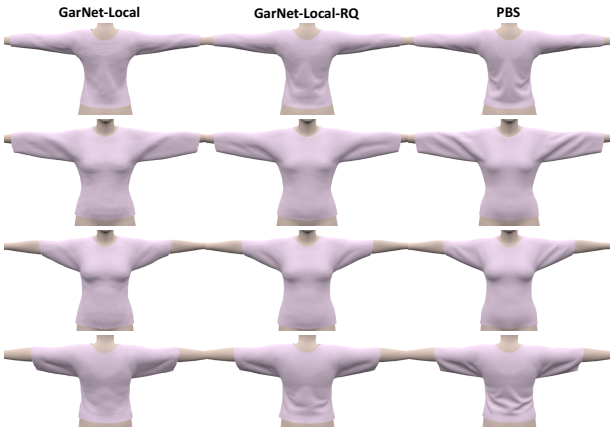


Fig. 15: **Using the curvature metrics to improve the level of detail on the dataset of [54].** Each row depicts a single body shape dressed by different method in each column. On the nearly planar part at the front of the T-shirt and on its back, **GarNet-Local-RQ** delivers both more realistic wrinkles and less noise than **GarNet-Local**.

	Jeans	**T-shirt**	**Sweater**	**Dress**
	$\mathcal{E}_{dist}/\mathcal{E}_{norm}$	$\mathcal{E}_{dist}/\mathcal{E}_{norm}$	$\mathcal{E}_{dist}/\mathcal{E}_{norm}$	$\mathcal{E}_{dist}/\mathcal{E}_{norm}$
GarNet-Naive	0.7/8.83	0.73/11.28	1.03/11.3	1.19/ 8.92
GarNet-Global	0.42/5.18	0.57/8.44	0.93/10.54	1.13 / 7.83
GarNet-Local	**0.41** / 5.1	**0.56** / 8.34	**0.85** / 10.39	**1.06** / 7.8
GarNet-Local-RQ	0.53 / 5.08	0.63 / 7.79	1.1 / 9.7	1.15 / 7.15
GarNet-Local-MC	0.45 / **4.79**	0.69 / 7.66	0.88 / **8.88**	1.16 / **6.85**
GarNet-Local-MCRQ	0.56 / 4.85	0.65/ **7.47**	1.14 / 9.54	1.21 / 7.18

TABLE 8: Average distance in cm and face normal angle difference in degrees between the PBS and predicted vertices in our dataset that uses SMPL body models.

because they lack wrinkles.

For the sake of completeness, we now turn to the error metrics introduced in Sec. 4.1 to show that the increase in realism does not significantly compromise the distance and angle accuracy. We report our results on our dataset in Table 8. In terms of lowest distance and angle errors, **GarNet-Local** and **GarNet-Local-MC** are virtually equivalent while **GarNet-Local-RQ** is slightly worse. This decrease in the angle and distance accuracy was expected because **GarNet-Local-RQ** puts more emphasis on generating local statistics that are closer to those of the **PBS** ground at the potential expense of the other loss terms.

When we compare **GarNet-Local-RQ** and **GarNet-Local** on the dataset of [54], the average distance error increases from 0.42cm (**GarNet-Local**) to 0.45cm (**GarNet-Local-RQ**). However, the additional *RQ* loss helps decrease the average angle error from 7.34 (**GarNet-Local**) to 7.2 (**GarNet-Local-RQ**). Moreover, **GarNet-Local-RQ** reduces the *RQ* curvature loss from 0.21 to 0.11.

By contrast, when the network is fine-tuned using the curvature loss term based on mean curvature normals, the training becomes unstable and diverges because of very high cotangent values of

	$\mathcal{E}_{dist}$	$\mathcal{E}_{norm}$	$\mathcal{L}_{MC}$	$\mathcal{L}_{RQ}$
GarNet-Local-UMC	**0.61**	8.25	0.14	0.23
GarNet-Local-MC	0.69	**7.66**	**0.11**	0.26
GarNet-Local-RQ	0.63	7.79	0.13	**0.09**

TABLE 9: Comparison between the proposed *RQ*-based curvature loss, the discrete mean curvature operator of Eq. 9, and the uniformly weighted mean curvature operator of Eq, 19 in our T-shirt dataset (SMPL body model).

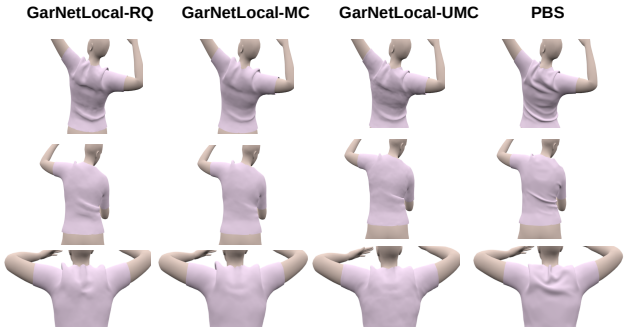


Fig. 16: Comparison of different curvature loss functions in our T-shirt dataset with SMPL body models to improve the details and reconstruct wrinkles and folds.

the facet angles, as discussed in Sec. 3.4. Hence, we do not report any result based on that loss here. To probe this further, we have also experimented on our T-shirt dataset with the Laplace Operator used in [49]. By analogy to the Mean Curvature Normal Operator in Eq. (9), we write it as

$$\kappa^i_{UMC} = \frac{1}{|\mathcal{N}^E_i|} \sum_{j \in \mathcal{N}^E_i} (\mathbf{G}_j - \mathbf{G}_i) \tag{19}$$

where $\mathbf{G}_i$ is the i^{th} vertex of the garment as in Sec. 3.1. This gives uniform weights to all vertices in a neighborhood. We use the same loss as in (14), where we replace κ_{MC} with κ_{UMC}. We will refer to to training using this operator as **GarNelLocal-UMC**. In Table 9 and Figure 16, we compare it to **GarNet-Local-RQ** and **GarNet-Local-MC**. Our proposed RQ-based curvature loss delivers the best trade-off between angle and distance accuracy: The distance accuracy decreases slightly but allows for a significant increase in angle accuracy. Moreover, it still yields the smallest total loss value for both the mean curvature and RQ. This is evidence that our RQ-based loss has a quantitative impact. Looking at the predicted 3D shapes such as those of Fig. 16, we can see that using our proposed loss results in more wrinkles and/or folds that make the result perceptually more similar to the PBS than the other two.

4.5 Moving away from SMPL Bodies.

The experiments described above were conducted on the dataset we generated and on the one of [54], both of which rely on the popular SMPL body models. It is currently the most widely used but this might change sooner or later. Because our method is generic and does not depend on the SMPL parameterization, it will remain relevant when and if this happens.

To demonstrate this, we generated another T-shirt and Sweater datasets based on the CAESAR female body shapes [45] in a single pose, as depicted by Fig. 17. We split the body shapes into train, validation and test sets comprising 1376, 344, and 432 shapes, respectively and run PBS simulations for all of them. In Table 10, we report our proposed *RQ*-based detail loss value, distance and angle errors for both **GarNet-Local** and **GarNet-Local-RQ** in the test set. Both configurations deliver good accuracy but with a clear advantage to **GarNet-Local-RQ** in terms of angle error and detail loss value.

	T-shirt	Sweater
	$\mathcal{E}_{dist}$ / $\mathcal{E}_{norm}$ /$\mathcal{L}_{RQ}$	$\mathcal{E}_{dist}$ / $\mathcal{E}_{norm}$ /$\mathcal{L}_{RQ}$
GarNet-Local-RQ	**0.46 / 6.56 / 0.025**	**0.56 / 6.66 / 0.031**
GarNet-Local	0.53 / 8.23 / 0.044	0.61 / 8.11 / 0.047

TABLE 10: Quantitative comparison between **GarNet-Local-RQ** and **GarNet-Local** in our dataset curated from CAESAR female body shapes.

Fig. 17: Qualitative results of **GarNet-Local-RQ** in our dataset curated from CAESAR female body shapes for T-shirt and sweater.

5 Conclusion

In this work, we have introduced a novel 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. To reduce the tendency of the network to remove high-frequency details, which can also be visually observed in [17] and [47], we have proposed two curvature loss functions that consider the local interactions between vertices and faces. This, in turn, has led to higher similarity to the Physics-Based Simulation while reducing the noise of the prediction of the network when trained without these loss terms.

References

[1] Thiemo Alldieck, Marcus Magnor, Bharat Lal Bhatnagar, Christian Theobalt, and Gerard Pons-Moll. Learning to reconstruct people in clothing from a single rgb camera. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[2] Thiemo Alldieck, Marcus Magnor, Weipeng Xu, Christian Theobalt, and Gerard Pons-Moll. Video Based Reconstruction of 3D People Models. In *Conference on Computer Vision and Pattern Recognition*, 2018.

[3] Thiemo Alldieck, Gerard Pons-Moll, Christian Theobalt, and Marcus Magnor. Tex2shape: Detailed full human body geometry from a single image. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[4] Jan Bednarík, Mathieu Salzmann, and Pascal Fua. Learning to Reconstruct Texture-Less Deformable Surfaces. In *International Conference on 3D Vision*, 2018.

[5] Bharat Lal Bhatnagar, Garvita Tiwari, Christian Theobalt, and Gerard Pons-Moll. Multi-garment net: Learning to dress 3d people from images. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[6] Blender, 2018. https://www.blender.org/.

[7] F. Bogo, A. Kanazawa, C. Lassner, P. Gehler, J. Romero, and M. J. Black. Keep It SMPL: Automatic Estimation of 3D Human Pose and Shape from a Single Image. In *European Conference on Computer Vision*, 2016.

[8] Davide Boscaini, Jonathan Masci, Emanuele Rodolà, and Michael Bronstein. Learning Shape Correspondence with Anisotropic Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*, pages 3189–3197, 2016.

[9] Remi Brouet, Alla Sheffer, Laurance Boissieux, and Marie-Paule Cani. Design Preserving Garment Transfer. *ACM Transactions on Graphics*, 31(4):361–3611, July 2012.

[10] CMU Graphics Lab Motion Capture Database, 2010. http://mocap.cs.cmu.edu/.

[11] Radek Danerek, Endri Dibra, Cengiz öztireli, Remo Ziegler, and Markus Gross. DeepGarment : 3D Garment Shape Estimation from a Single Image. *Eurographics*, 2017.

[12] Edilson de Aguiar, Leonid Sigal, Adrien Treuille, and Jessica K. Hodgins. Stable spaces for real-time clothing. *ACM Transactions on Graphics (SIGGRAPH 2010)*, 29(3), 2010.

[13] Haowen Deng, Tolga Birdal, and Slobodan Ilic. PPFNet: Global Context Aware Local Features for Robust 3D Point Matching. In *Conference on Computer Vision and Pattern Recognition*, 2018.

[14] Marvelous Designer, 2018. https://www.marvelousdesigner.com.

[15] Valentin Gabeur, Jean-Sebastien Franco, Xavier Martin, Cordelia Schmid, and Gregory Rogez. Moulding humans: Non-parametric 3d human shape estimation from single images. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[16] Peng Guan, Oren Freifeld, and Michael J. Black. A 2D Human Body Model Dressed in Eigen Clothing. In *European Conference on Computer Vision*, pages 285–298, 2010.

[17] Peng Guan, Loretta Reiss, David Hirshberg., Alexander Weiss, and Michael J. Black. DRAPE: Dressing Any Person. *ACM SIGGRAPH*, 31(4):351–3510, July 2012.

[18] Erhan Gundogdu, Victor Constantin, Amrollah Seifoddini, Minh Dang, Mathieu Salzmann, and Pascal Fua. Garnet: A two-stream network for fast and accurate 3d cloth draping. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2019.

[19] Marc Habermann, Weipeng Xu, Michael Zollhöfer, Gerard Pons-Moll, and Christian Theobalt. Livecap: Real-time human performance capture from monocular video. *ACM Trans. Graph.*, 38(2):14:1–14:17, Mar. 2019.

[20] Xintong Han, Zuxuan Wu, Zhe Wu, Ruichi Yu, and Larry S. Davis. Viton: An image-based virtual try-on network. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[22] Huamin Wangand Florian Hecht, Ravi Ramamoorthi, and James O'Brien. Example-Based Wrinkle Synthesis for Clothing Animation. In *ACM SIGGRAPH*, 2010.

[23] Roger A Horn and Charles R Johnson. *Matrix analysis*. Cambridge university press, 2012.

[24] Ladislav Kavan, Steven Collins, Jiri Žára, and Carol O'Sullivan. Skinning with Dual Quaternions. In *Proceedings of the 2007 Symposium on Interactive 3D Graphics and Games*, pages 39–46, 2007.

[25] Doyub Kim, Woojong Koh, Rahul Narain, Kayvon Fatahalian, Adrien Treuille, and James F. O'Brien. Near-exhaustive Precomputation of Secondary Cloth Effects. *ACM Trans. Graph.*, 32(4):87:1–87:8, July 2013.

[26] Tae-Yong Kim and Eugene Vendrovsky. DrivenShape - A Data-Driven Approach for Shape Deformation. In *Eurographics/SIGGRAPH Symposium on Computer Animation*, 2008.

[27] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *CoRR*, 2014. cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015.

[28] Zorah Lahner, Daniel Cremers, and Tony Tung. Deepwrinkles: Accurate and Realistic Clothing Modeling. In *European Conference on Computer Vision*, September 2018.

[29] Christoph Lassner, Gerard Pons-Moll, and Peter V. Gehler. A generative model of people in clothing. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[30] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A Skinned Multi-Person Linear Model. *ACM SIGGRAPH Asia*, 34(6), 2015.

[31] Naureen Mahmood, Nima Ghorbani, Nikolaus F. Troje, Gerard Pons-Moll, and Michael J. Black. AMASS: Archive of Motion Capture as Surface Shapes. *CoRR*, abs/1904.03278, 2019.

[32] Jonathan Masci, Davide Boscaini, Michael M. Bronstein, and Pierre Vandergheynst. Geodesic Convolutional Neural Networks on Riemannian Manifolds. In *International Conference on Computer Vision*, pages 832–840, December 2015.

[33] Mark Meyer, Mathieu Desbrun, Peter Schröder, and Alan H. Barr. Discrete differential-geometry operators for triangulated 2-manifolds. In Hans-Christian Hege and Konrad Polthier, editors, *Visualization and Mathematics III*, pages 35–57, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.

[34] Eder Miguel, Derek Bradley, Bernhard Thomaszewski, Bernd Bickel, Wojciech Matusik, Miguel A. Otaduy, and Steve Marschner. Data-driven estimation of cloth simulation models. *Comput. Graph. Forum*, 31(2pt2):519–528, May 2012.

[35] Federico Monti, Davide Boscaini, Jonathan Masci, Emanuele Rodolà, Jan Svoboda, and Michael M. Bronstein. Geometric Deep Learning on Graphs and Manifolds Using Mixture Model CNNs. In *Conference on Computer Vision and Pattern Recognition*, pages 5425–5434, 2017.

[36] Matthias Muller, Bruno Heidelberger, Marcus Hennix, and John Ratcliff. Position Based Dynamics. *Journal of Visual Communication and Image Representation*, 18(2):109–118, 2007.

[37] Nvidia. Nvcloth, 2018. https://docs.nvidia.com/gameworks/content/gameworkslibrary/physx/nvCloth/index.html.

[38] Nvidia. Nvidia flex, 2018. https://developer.nvidia.com/flex.

[39] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary Devito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic Differentiation in Pytorch. In *Advances in Neural Information Processing Systems*, 2017.

[40] M. Pauly, M. Gross, and L. P. Kobbelt. Efficient simplification of point-sampled surfaces. In *IEEE Visualization, 2002. VIS 2002.*, pages 163–170, Oct 2002.

[41] Gerard Pons-Moll, Sergi Pujades, Sonny Hu, and Michael J. Black. Clothcap: Seamless 4D Clothing Capture and Retargeting. *ACM SIGGRAPH*, 36(4):731–7315, July 2017.

[42] Tiberiu Popa, Qingnan Zhou, Derek Bradley, Vladislav Kraevoy, Hongbo Fu, Alla Sheffer, and Wolfgang Heidrich. Wrinkling Captured Garments Using Space-Time Data-Driven Deformation. *Computer Graphics Forum (Proc. Eurographics)*, 28(2):427–435, 2009.

[43] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep Learning on Point Sets for 3D Classification and Segmentation. In *Conference on Computer Vision and Pattern Recognition*, 2017.

[44] Charles R. Qi, Li Yi, Hao Su, and Leonidas J. Guibas. Pointnet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *Advances in Neural Information Processing Systems*, 2017.

[45] Kathleen Robinette, Sherri Blackwell, Hein Daanen, Mark Boehmer, Scott Fleming, Tina Brill, David Hoeferlin, and Dennis Burnsides. Anthropometry Resource (CAESAR) Final Report, US Air Force Research Laboratory. *Tech. Rep. AFRL-HEWP-TR-2002-0169*, 01 2002.

[46] Shunsuke Saito, Zeng Huang, Ryota Natsume, Shigeo Morishima, Angjoo Kanazawa, and Hao Li. Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[47] Igor Santesteban, Miguel A. Otaduy, and Dan Casas. Learning-Based Animation of Clothing for Virtual Try-On. *Computer Graphics Forum (Proc. of Eurographics)*, 33(2), 2019.

[48] Optitext Fashoin Design Software, 2018. https://optitex.com/.

[49] Olga Sorkine. Laplacian Mesh Processing. In *Eurographics 2005 - State of the Art Reports*. The Eurographics Association, 2005.

[50] Gilbert Strang. *Linear algebra and its applications*. Thomson, Brooks/Cole, Belmont, CA, 2006.

[51] Min Tang, Tongtong Wang, Zhongyuan Liu, Ruofeng Tong, and Dinesh Manocha. I-Cloth: Incremental collision handling for GPU-based interactive cloth simulation. *ACM Transaction on Graphics (Proceedings of SIGGRAPH Asia)*, 37(6):204:1–10, November 2018.

[52] Nitika Verma, Edmond Boyer, and Jakob Verbeek. Feastnet: Feature-Steered Graph Convolutions for 3D Shape Analysis. In *Conference on Computer Vision and Pattern Recognition*, 2018.

[53] Tuanfeng Wang, Tianjia Shao, Kai Fu, and Niloy Mitra. Learning an intrinsic garment space for interactive authoring of garment animation. *ACM Trans. Graph.*, 38(6), 2019.

[54] 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.

[55] Wei Wang, Zheng Dang, Yinlin Hu, Pascal Fua, and Mathieu Salzmann. Backpropagation-Friendly Eigendecomposition. In *Advances in Neural Information Processing Systems*, 2019.

[56] Weipeng Xu, Avishek Chatterjee, Michael Zollhöfer, Helge Rhodin, Dushyant Mehta, Hans-Peter Seidel, and Christian Theobalt. Monoperfcap: Human performance capture from monocular video. *ACM Trans.*

Graph., 37(2):27:1–27:15, May 2018.
[57] Jinlong Yang, Jean-Sebastien Franco, Franck Hetroy-Wheeler, and Stefanie Wuhrer. Analyzing clothing layer deformation statistics of 3d human motions. In *The European Conference on Computer Vision (ECCV)*, September 2018.
[58] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. FoldingNet: Point Cloud Auto-Encoder via Deep Grid Deformation. In *Conference on Computer Vision and Pattern Recognition*, June 2018.
[59] Kwang M. Yi, Eduard Trulls, Yuki Ono, Vincent Lepetit, Mathieu Salzmann, and Pascal Fua. Learning to Find Good Correspondences. In *Conference on Computer Vision and Pattern Recognition*, 2018.
[60] Lequan Yu, Xianzhi Li, Chi-Wung Fu, Daniel Cohen-Or, and Pheng-Ann Heng. Pu-Net: Point Cloud Upsampling Network. In *Conference on Computer Vision and Pattern Recognition*, pages 2790–2799, 2018.
[61] Tao Yu, Zerong Zheng, Kaiwen Guo, Jianhui Zhao, Qionghai Dai, Hao Li, Gerard Pons-Moll, and Yebin Liu. Doublefusion: Real-time capture of human performances with inner body shapes from a single depth sensor. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
[62] Tao Yu, Zerong Zheng, Yuan Zhong, Jianhui Zhao, Qionghai Dai, Gerard Pons-Moll, and Yebin Liu. Simulcap : Single-view human performance capture with cloth simulation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.
[63] Chao Zhang, Sergi Pujades, Michael J. Black, and Gerard Pons-Moll. Detailed, accurate, human shape estimation from clothed 3d scan sequences. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.
[64] Zerong Zheng, Tao Yu, Yixuan Wei, Qionghai Dai, and Yebin Liu. Deephuman: 3d human reconstruction from a single image. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

Erhan Gundogdu received the Ph.D. degree in Electrical and Electronics Engineering Dept. from Middle East Technical University (METU), Ankara, Turkey in 2017. He was with Aselsan Inc., Ankara, Turkey between 2013 and 2017. He worked as a Postdoctoral Researcher in CVLab, EPFL, Switzerland until October 2019. Now, he is with Amazon in Berlin, Germany. His current research interests include 3D shape learning, cloth draping, visual tracking, object recognition and detection.

Victor Constantin received his M.Sc. in Computer Science from EPFL in 2016. He is currently working as a Research Engineer at CVLab at EPFL. He's research interest include deep learning, 3D pose estimation and garment draping.

Shaifali Parashar received her Ph.D. in Computer Vision from the Université d'Auvergne in 2017. She is currently a PostDoc researcher at CVLab, EPFL. Her research interest are 3D computer vision including non- rigid 3D reconstruction and deformable SLAM.

Amrollah Seifoddini received his M.Sc. in Visual Computing from ETH Zurich in 2016. Currently, he is a research engineer at meepl, a Zurich-based tech company offering smartphone-based 3D body scanning, made-to-measure, size recommendation, and 3D virtual dressing room services for the apparel industry. His main research interests include human-centered visual computing and 3D reconstruction.

Minh Dang Dr. Minh Dang received his Ph.D. in 3D Computer Graphics from EPFL in 2016. Currently, he worked at Fision Technologies in Zurich where he leads a R&D team developing a smart phone-based 3D body scanning platform and the applications. His main research interests include geometry processing, 3D reconstruction, and 3D machine learning. He is now with Facebook in Zurich, Switzerland.

Mathieu Salzmann received the M.Sc. and Ph.D. degrees from EPFL, in 2004 and 2009, respectively. He then joined the International Computer Science Institute and the EECS Department with the University of California at Berkeley as a postdoctoral fellow, later the Toyota Technical Institute at Chicago as a research assistant professor and a senior researcher with the NICTA in Canberra. He is now a senior researcher in Computer Vision Lab, EPFL.

Pascal Fua received an engineering degree from Ecole Polytechnique, Paris, in 1984 and a Ph.D. in Computer Science from the University of Orsay in 1989. He joined EPFL in 1996 as a Professor in the School of Computer and Communication Science He is the head of the Computer Vision Lab. He has (co)authored over 300 publications in refereed journals and conferences and received several ERC grants. He is an IEEE Fellow and has been an Associate Editor of IEEE Transactions for Pattern Analysis and Machine Intelligence.