Chapter 4: Joint Estimation of Human and Garment from Video

4.1 Introduction

Chapter 2 and 3 introduce novel methods to estimate the human body and garment material independently. It is not difficult to observe that the two tasks are closely related. Jointly estimating both in an end-to-end system can help removing certain ambiguity originating from the input image. Moreover, estimating the garment materials of a worn garment is a more challenging task than what Chapter 3 introduces; it does not assume known external forces or even garment geometry, making differentiable physics not applicable under such a circumstance.

In this chapter, I introduce an end-to-end learning model that achieves both garment geometry estimation and fabric material prediction at the same time. To handle the dynamic geometry and different topologies of the garments and to provide a unified parametric model for the garments, I propose a two-level auto-encoder network. The key observation is that classical point cloud encoders such as PointNet [99] are great for capturing global shapes, but not suitable for encoding the local details. Multi-scale feature extraction not only decomposes the problem into smaller partitions, but also decouples global and local features to enable larger coverage on local shape learning. It is also critical to construct a continuous space that includes different topological structures, since topology transitions often happen locally. During the estimation, I couple the human body inference with the garment recovery to maximize the estimation accuracy of the two correlated tasks. Other than traditional multi-tasking, I further introduce a closed-loop structure so that the garment features of different scales can guide the body estimation to improve the accuracy for both. Based on the temporal change of these garment features, I can then perform accurate material classification accordingly. To sum up, my key contributions include:

- The first end-to-end neural network that recovers fabric material(s) of a garment from one single RGB video (Section 4.3);
- A novel two-level auto-encoder for learning the latent space of garments through multi-scale feature coupling (Section 4.4);
- Joint estimation of human body and apparels through a close-loop iterative optimization (Section 4.5);
- A large dataset of garment motion sequences with wide variations of human body, fabric materials, textures, and lightings (Section 4.6);
- The first garment prediction model that can account for arbitrary topologies and use a feedback loop to the body estimation for prediction consistency (Section 4.7).

My experiments show that the proposed network structure effectively increases the capability and accuracy of the fabric material estimation. By using only a few frames of a person wearing a garment, my model can faithfully reconstruct the garment fabric material(s), using the recovered shape and motion of both the garment and the human body as the conditioning in the material estimation task.

4.2 Related Work

Fabric material estimation. Researchers have been tackling different inverse problems, including inverse cloth design [100], combinatorial material design [101], BRDF parameter capturing [102], weaving pattern reconstruction [103], human material perception [104, 105], and frictional coefficient estimation [106, 107]. As a typical example of inverse problems, physical material estimation is a challenging yet important component of scene understanding. Cloth material estimation is even more challenging than most due to its highly dynamic motions. Previous works study the task in a simplified and constrained scenario, and recover the materials using statistical observation [108, 109], optimization [110, 111], or learning [21, 112]. In contrast, my method learns the cloth material from videos of a human wearing garments, which is more general and widely applicable. More importantly, my method makes use of the estimated multi-scale garment latent codes as input signal as well, which is shown to be much more useful in recovering overall garment geometry with local details than the traditional image features.

Garment modeling and estimation. Garment geometry capturing or recovery has been studied in computer graphics and computer vision. To address this problem, non-learning methods employ symmetry with user input [113], optimization [110, 114], or binocular data [115]. Recently, methods using deep learning have been proposed for faster speed and more convenient usage [8, 10, 11, 12, 116, 117, 118, 119, 120, 121, 122]. In addition, direct garment modeling methods have also been proposed using spherical parameterization [7] for estimation or displacement map [123] for retargetting.

Different from displacement-based cloth representation [10], PCA-based models [9], and mesh-CNN-based methods [8], my garment model is universal to all topologies, applicable to those not homotopy to human surfaces (*e.g.* long dresses), and enabling semantic interpolation between different garments. Compared with [7], my model generates a stand-alone garment mesh, which is easy to export and retarget. More importantly, my method is the *first end-to-end network that jointly estimates the garment material together with its geometry.*

This method focuses on the *material* estimation of garments. Without any prior knowledge, I take an RGB video as input and recover the garment mesh, which is then used for fabric material estimation. My method is substantially different from most garment capturing or generation methods [90, 122, 124] regarding model input, output, and assumptions.

Point cloud encoder and decoder. PointNet [99] was among the first network model for encoding an unordered point set. Follow-on improvements include spatial partition [125, 126, 127], edge convolution [128], local region filtering [129, 130], and analogous convolutional operators [131].

Although these recent works have utilized hierarchical structure to some extent, their methods are not sufficient for auto-encoding the garment geometry or topology. The key difference between garment auto-encoding and rigid gadgets auto-encoding is that there are a large number of local details (*e.g.* wrinkles) due to cloth's highly deformable and dynamical nature. As a result, latent codes for local details are necessary. Methods using random sampling [131] or farthest point sampling [127] propose unstable representative points for similar input, thus not suitable for local feature encoding. [131] used a stabilized version of self-organized mapping (SOM) for representative point generation, but did not further encode the local features. So this method [131] is still a single-level auto-encoder.

In contrast, my two-level auto-encoder for garments successfully encodes local features, while capturing the global geometry. The trained latent code maintains similarity between similar input point clouds – otherwise not achievable using unstable representative points in prior works.

Human reconstruction from images. Human estimation using RGB images has been a popular research topic in deep learning for its importance in virtual reality and computer animation. While early works propose network models for only 2D/3D body skeletons [31, 132, 133], more recent works introduce techniques to regress the entire human body – either using a parametric human model [2, 3] or voxel-based representation [4, 5, 6]. Given the fact that the annotations in most real-world datasets contain only joint positions, the learning process has been refined in various ways [68, 134, 135, 136, 137].

In order to estimate the fabric material, I need to recover the garment shape on the human body, which is an important problem rarely addressed in human estimation tasks. In my pipeline, I use state-of-the-art human body predictions as a strong prior for the garment estimation module. Given the focus of this paper, I use video input solely for garment material estimation. Thereby, I do not review related works on video-based pose estimation here. Instead, I refer the interested readers to a recent survey [138].

4.3 Method Overview

I first give the formal problem definition below. Given a video clip showing a person moving (*e.g.* walking, jumping, bending, etc), I estimate the fabric materials of the garment worn by the person. By *fabric materials*, I refer to the physical material parameters used in cloth simulation. I adopt the same material parameter definition introduced in [93], which consists of 24 parameters for stretching stiffness and 15 parameters for bending stiffness.

Due to the fact that the differences of the material parameter values do not intuitively reflect the human visual perception, I follow the previous work [21] to discretize the material parameter space based on the amount of deformations due to external forces. Using sensitivity analysis [139], the stretching stiffness is split into 6 classes and 9 for the bending. Combining both dimensions will yield 54 different material classes. As confirmed by [21], these 54 classes cover most of the common materials, including polyester, cotton, nylon, rayon, and their combinations. For example, one type of materials named *'white-swim-solid'* consisting of 87% nylon and 13% spandex, as measured by [93], fits in the discrete classification model with the stretching label of 2 and the bending label of 3.

In this paper, I introduce an end-to-end deep neural network (Fig. 4.1) for simultaneously estimating the garment geometry and its material type(s), along with the human body. My key idea is that image features are not sufficient for inferring garment materials; it is necessary to extract the garment geometry as well for a more accurate estimation. To support different topologies of garments, I choose point clouds for its geometry representation. To better account for the highly dynamic garment surfaces, I train a two-level point cloud auto-encoder (Sec. 4.4) so that it can learn the global shapes and local features of the garment to reduce the total number of degrees of freedom. I use the SMPL model (see [30] for its rigorous math definition) to represent the human pose and shape.

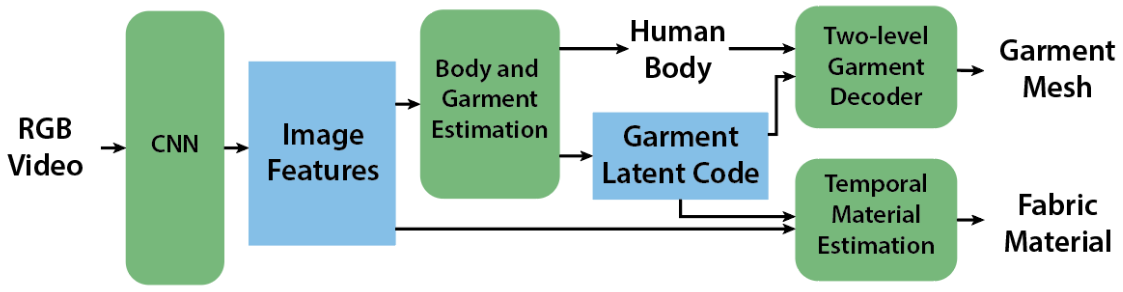


Figure 4.1: **Overall network structure**. Given an RGB video, I extract its image features and estimate the body and garment shape frame by frame (Sec. 4.5.1). The latter is decoded to obtain a garment mesh (Sec. 4.4). The temporal sequences of image and garment are fed to an LSTM for material classification (Sec. 4.5.2).

I divide the estimation pipeline into two phases. First, I estimate the human body and the cloth geometry in a frame-by-frame manner (Sec. 4.5.1). A closed-loop optimization structure is used to improve the estimation accuracy of these two correlated tasks. The garment geometry prediction module is conditioned on the human body parameters, and at the same time provides corrective feedback to the human body prediction module. I then feed the features of the image and the garment geometry from each frame together to a temporal neural network for the garment material estimation (Sec. 4.5.2). By sharing common features, providing corrective feedback, and conditioning on outputs of closely-related tasks, my network model can achieve higher estimation accuracy on all three tasks than independent estimation baselines.

4.4 Garment Auto-encoder

I first set up an auto-encoder for the cloth model. Since the model is designed

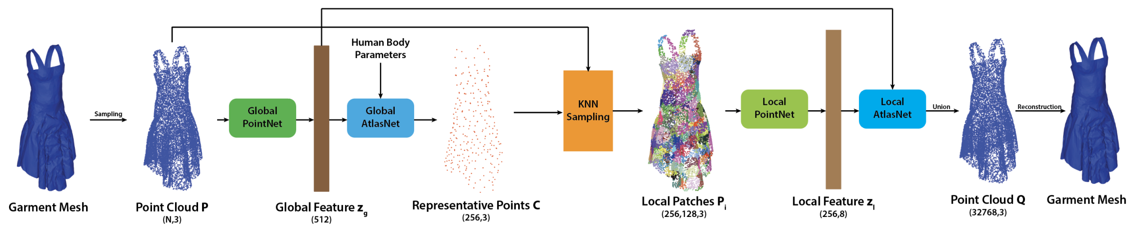


Figure 4.2: **The network structure of the garment auto-encoder**. The point cloud sampled from the original mesh is first fed to a global PointNet for coarse shape features. Its representative point set is then obtained by decoding the global features from an AtlasNet. From those points, I sample the local patches using K-nearest neighbor and pass them to a local PointNet for detailed shape features. The local decoder is then conditioned on the global latent code and the corresponding patch center to recover the patches that are stitched together to form the reconstructed point cloud.

not to assume fixed garment topology, I choose to use point clouds as the underlying representation. Other representations, such as graph-based [140] or displacement-based [10, 11], rely on either fixed graph structure, or fixed human surface, thus not applicable for generalization to different garments. The use of auto-encoder here is necessary because the degrees of freedom (DoF) for point clouds are too high for estimation. An encoder-decoder structure can effectively reduce the DoF and retain only the essential information, such as the global shape and the local details. More importantly, it clusters similar shapes to similar latent codes, which is beneficial to the estimation module. As later shown in the Appendix, my model provides smooth transitioning between different topologies by using simple interpolation between latent codes, which is never achieved in previous works.

4.4.1 Two-Level Encoder-Decoder Structure

Previous point cloud auto-encoders such as AtlasNet [141] use Multi Layer Perceptron (MLP) to transform a 2D patch to a set of 3D points in the space. Their method performs well in point cloud datasets that include rigid objects, such as airplanes or chairs, since the deformations presented in those objects are simple and regular. However, it cannot be directly applied to learn garment point clouds, since garments have a much larger variance in point cloud distribution due to its dynamic nature. For example, a simple dress can create different wrinkle structures under different external forces. As a result, one global auto-encoder cannot account for all detailed structures, resulting in overly smoothed point clouds. Recently, [142] proposes a method to resolve patch overlapping and collapsing occurred in AtlasNet, but it still cannot account for arbitrary topologies and detailed wrinkles.

I propose a two-level auto-encoder for learning the latent space of the cloth. As shown in Fig. 4.2, I use a set of representative points $\mathbf{C}$ to express the global shape of the garment, and sample around them to form local patches, which are encoded independently to account for local shapes. Specifically, given a point cloud $\mathbf{P}$, I first pass it through a global auto-encoder to form a representative point cloud:

$$\mathbf{C} = D_g(E_g(\mathbf{P}), \theta) \tag{4.1}$$

where E_g and D_g are the global encoder and decoder, and θ is the human body parameter. Next, I use K-nearest-neighbor to sample points around the representative ones:

$$\mathbf{P}_i = KNN(\mathbf{P}, \mathbf{c}_i) \tag{4.2}$$

where $\mathbf{c}_i$ is the i-th element in $\mathbf{C}$, and $\mathbf{P}_i$ is the i-th patch. This step forms local patches around the representative points. Finally, I pass each patch to the shared local auto-encoder, and do a union operation to obtain the reconstructed point cloud:

$$\mathbf{Q}_i = D_l(E_l(\mathbf{P}_i), \mathbf{z}_g, \mathbf{c}_i) \tag{4.3}$$

$$\mathbf{Q} = \bigcup_i \mathbf{Q}_i \tag{4.4}$$

where $\mathbf{Q}_i$ and $\mathbf{Q}$ are the reconstructed patches and point cloud, D_l and E_l are the local decoder and encoder, and $\mathbf{z}_g = E_g(\mathbf{P})$ is the global latent code.

4.4.2 Representative Point Set Extraction

Note that Eq. 4.3 and 4.4 imply that the representative points $\mathbf{C}$ have to be in the same order as the local latent codes $\mathbf{z}_l$. This is the key reason why traditional methods such as farthest point sampling [127] do not work: its ordering is very sensitive to the input, resulting in an unknown mapping between reconstructed patch centers $\mathbf{C}$ and the local patches $\mathbf{P}_i$ (thus the local latent code $\mathbf{z}_{l_i}$).

To resolve this issue, I encode the entire point cloud and compute the representative points using the decoder itself. Due to the continuous nature of the auto-encoder network, the continuity and consistency regarding similar point clouds are guaranteed, thus ensuring $\mathbf{c}_i$ to be exactly matched with $\mathbf{P}_i$.

4.4.3 Training Losses

During training, I use Chamfer Distance between two point clouds as the loss:

$$d(\mathbf{P}, \mathbf{Q}) = \frac{1}{|\mathbf{P}|} \sum_{\mathbf{p} \in \mathbf{P}} \min_{\mathbf{q} \in \mathbf{Q}} \|\mathbf{p} - \mathbf{q}\| + \frac{1}{|\mathbf{Q}|} \sum_{\mathbf{q} \in \mathbf{Q}} \min_{\mathbf{p} \in \mathbf{P}} \|\mathbf{q} - \mathbf{p}\| \tag{4.5}$$

I apply the Chamfer Distance loss between the representative point set and the point cloud to learn the global shape, and the one between the recovered and the original point cloud, both patch-wisely and globally, to capture the local details:

$$\mathcal{L}_{AE} = d(\mathbf{P}, \mathbf{C}) + \frac{1}{n} \sum_{i=1}^{n} d(\mathbf{P}_i, \mathbf{Q}_i) + d(\mathbf{P}, \mathbf{Q}) \tag{4.6}$$

4.4.4 Recovery from Point Clouds to Garment Meshes

The point cloud representation of a garment mesh does not explicitly store the connectivity information, so it is necessary to apply certain prior when recovering meshes from point clouds. One straightforward way is to connect each point with its neighbors, determined by a distance threshold. However, this method does not guarantee the resulting mesh to be a manifold. Instead, I use manifold surfaces to approximate the results.

The overall pipeline to recover point clouds to meshes is split into several steps. I first employ Screen Poisson algorithm [143] on the point cloud with estimated normals using neighboring points to recover the overall shape and topology of the garment. Next, I remove the reconstructed vertices that are too far away from the point cloud, since the first step tends to generate water-tight meshes. I focus on removing large clusters that forms holes for neck, arms, and legs. After upsampling on the resulting mesh, I deform the mesh by minimizing the distance $d_M(\mathbf{P}, \mathbf{F})$ between the point cloud $\mathbf{P}$ and the mesh $\mathbf{F}$ as defined below:

$$d_M(\mathbf{P}, \mathbf{F}) = \frac{1}{|\mathbf{P}|} \sum_{\mathbf{p} \in \mathbf{P}} \min_{\mathbf{f} \in \mathbf{F}} \hat{d}(\mathbf{p}, \mathbf{f}) + \frac{1}{|\mathbf{F}|} \sum_{\mathbf{f} \in \mathbf{F}} \min_{\mathbf{p} \in \mathbf{P}} \hat{d}(\mathbf{p}, \mathbf{f}) \tag{4.7}$$

where $\hat{d}$ is the point-to-face distance, $\mathbf{f} \in \mathbf{F}$ is a face in the garment mesh, $\mathbf{F}$ is the set of all faces of the mesh. I use other regularization terms similar to the mesh deformation demo in Pytorch3D [144].

4.5 Material Estimation

With the garment auto-encoder (Sec. 4.4) at hand, garment material estimation becomes tractable. I design my overall pipeline as shown in Fig. 4.3. Given the sequence of image frames, I first feed them one by one to a model for estimating the human body and cloth geometry. By predicting the latent vector instead of the exact positions of the point cloud, the single-frame estimation network avoids severe overfitting or producing irrational results, due to the reduction of the degree of freedom by the auto-encoder.

Next, I combine the image features as well as the estimated garment latent code as the temporal signals, which go through a canonical temporal network module (*i.e.* LSTM [145]) to predict the final material type. Since the latent space preserves similarity (*i.e.* positive correlation between distances of latent vectors and distances between the original point clouds), the motion of the estimated latent vector becomes a better indicator of garment motion than image features, which is beneficial to garment material learning. I do not include body features here because the garment material is directly related to the garment motion, which has already taken the human body as the condition (Sec. 4.5.1). I discuss more details of the network in the following sections.

4.5.1 Single Frame Closed-Loop Estimation

As shown in Fig. 4.4, I train a model to estimate the human body and cloth geometry given one single frame. Formally, in each frame, I am given the image

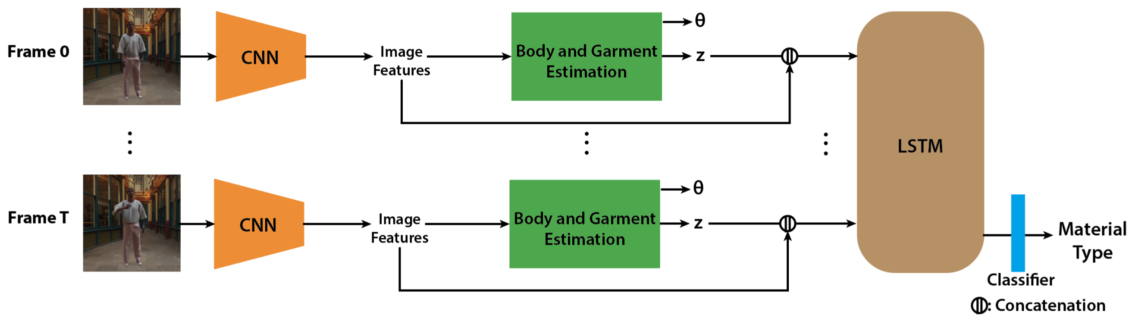
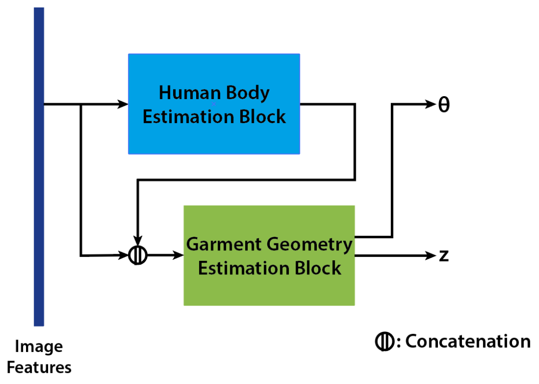
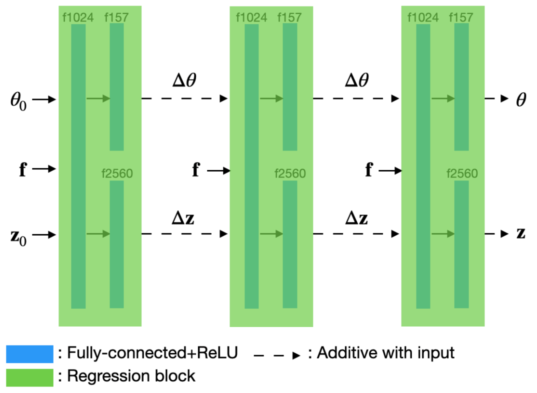


Figure 4.3: **My estimation pipeline**. Each video frame is first processed to obtain the image feature, the human body, and the garment shape. Then the image features are concatenated with the garment latent code as input to LSTM for material recovery.


Human Body Estimation Block
Garment Geometry Estimation Block
θ
z
Image Features
: Concatenation

(a) Closed loop structure for body & garment estimation



(b) Detailed structure in the garment estimation block

Figure 4.4: **The network structure for body and garment estimation in each frame**. (a) The garment shape estimation block takes the human body parameters as a prior, but also provides a feedback correction. (b) The garment estimation module consists of three identical, shared-weight blocks, each of which takes image features $\mathbf{f}$ and current predictions of the human body θ_0 and garment $\mathbf{z}_0$, and outputs the corrective values.

features $\mathbf{f}$. I first go through a state-of-the-art body estimation block, HB, to get a first-hand body estimation:

$$\hat{\theta} = HB(\mathbf{f}) \tag{4.8}$$

where $\hat{\theta} = [\theta, \beta]$ are the human body parameters including pose and shape. In the garment estimation block, I take as input $\mathbf{f}$ together with $\hat{\theta}$ and regress the garment latent code $\mathbf{z}$ and a final value of body parameters θ. Inside the garment estimation block, I use three shared-parameter small regression blocks, RB, to iteratively provide the correction, given the current estimation:

$$\theta_0 = \hat{\theta} \quad \mathbf{z}_0 = \mathbf{0} \tag{4.9}$$

$$\Delta\theta_i, \Delta\mathbf{z}_i = RB(\theta_{i-1}, \mathbf{z}_{i-1}) \tag{4.10}$$

$$\theta_i = \theta_{i-1} + \Delta\theta_i \quad \mathbf{z}_i = \mathbf{z}_{i-1} + \Delta\mathbf{z}_i \tag{4.11}$$

Overall, the garment estimation block forms a closed-loop structure, in which the human body parameters are required to predict the garment, and are later corrected back by the garment prediction as well.

The key insight of my module design is that the human body and garment shape are highly correlated at different scales and should be jointly learned using shared information. On the global scale, the detailed features of the garments restrict the variance of the human body and reduce ambiguity due to camera projection. On the local scale, the body pose and shape largely defines the valid distribution of the garment wrinkle positions. My proposed structure is also analogous to iterative optimization and feedback control in other areas, where two objectives serve as prior knowledge of each other and are improved iteratively. My work is the first to introduce this idea to the human and garment joint estimation task.

The loss function for the single-frame estimation is defined as:

$$\mathcal{L}_s = \mathcal{L}_{body} + \mathcal{L}_{AE} \tag{4.12}$$

$$\mathcal{L}_{body} = \mathcal{L}_{2D} + \mathcal{L}_{3D} + \mathcal{L}_{SMPL} \tag{4.13}$$

where $\mathcal{L}_{2D}$, $\mathcal{L}_{3D}$, and $\mathcal{L}_{SMPL}$ represent the 2D joint loss, the 3D joint loss, and the body parameter loss defined to supervise the human body estimation [136], and $\mathcal{L}_{AE}$ is the Chamfer distance defined in Eq. 4.6 to supervise the garment estimation.

4.5.2 Temporal Estimation for Garment Material

Garment material estimation is challenging since the visual difference of different materials is subtle and can easily be overwhelmed by disturbance, *e.g.* various directions or magnitudes of external forces. To tackle this problem, previous works often assume fixed environment settings and cloth shapes [21, 110]. While I follow a similar principle when training the material estimation module, I go one step further that I only assume common human motion for driving the garment instead of the whole external force field. While previous works [21, 110] can only handle videos of a piece of cloth hanging and dragged by the wind, my method possesses a wider applicability regarding the diversity of the garment shapes, sizes and human motions in the input video, which for the first time enables practical usages for garment material cloning.

As shown in Fig. 4.3, I collect and concatenate the image features and the estimated garment latent vector of each frame as the input signal, and feed the sequence of the signals to LSTM to produce a summary feature. Finally, I pass the summary feature to a fully-connected layer for material type classification. I use the cross-entropy loss for supervision.

I train the model in an end-to-end fashion, but deliberately fix the single-frame estimation module. While my training does not benefit from end-to-end fine-tuning (which I technically can), it poses an even more challenging task for the temporal estimation module. No matter how expressive the network is, the training results will not be great, if the input signal provided by the previous module is noisy. My experiments demonstrated in Sec. 4.7 indicate that the multi-scale garment features are not merely useful for detecting the fabric materials; they are the dominant features during the estimation and can boost the test accuracy compared to methods that only feed the image features.

4.6 Data Preparation and Training

In order to train my model, a large number of examples that contain ground truth human body parameters, garment meshes, and the corresponding material parameters are needed. These are very challenging to capture in real world. To supplement a very limited number of such real-world videos, I create a large dataset of videos generated with controlled variables with the corresponding ground-truth values for validation. I vary different conditions to generate this dataset:

Human motion. I sample common human motion sequences and shape parameters in the CMU Mocap dataset [65], including walking, sitting, boxing and climbing stairs.

Garment meshes. I use the garment dataset from [146].

Material space. I sample different materials uniformly in the discretized space mentioned in Sec. 4.3.

Human and garment textures. I use random human textures from SURREAL [147] and random garment textures from online images.

Lighting. I employ diverse outdoor and indoor environment maps downloaded from [148].

In total, I create a dataset of 250,000 images (10 motions * 100 garments * 250 frames) for single-frame human and garment geometry estimation, and 140,000 video sequences with each consisting of 25 images sampled at a frame rate of 5. Some examples of my training datasets are shown in Fig. 4.15.

4.6.1 Training Details

The split percentages for the train/validation/test sets are 85%/5%/10%. During training, I use Adam [149] to minimize the loss. The learning rate is 10^{-3} for the auto-encoder, and 10^{-4} for others. They are scaled down to 10^{-4} and 10^{-5} respectively after 10 epochs. All hyper-parameters are chosen empirically. As shown in Fig. 4.2, I use a representative point set of size 256, and generate 128 points for each patch. The sampling size for the ground-truth point set (N in Fig. 4.2) is 16,384.

I trained and tested my model on a machine with 8 CPUs (Intel Xeon, 3.60GHz) for data loading and 2 GPUs for computing (GeForce GTX 1080). I trained my auto-encoder, single-frame module, and the temporal module for 20 epochs, respectively.

4.7 Experiments

I demonstrate the performance of my model as follows. (1) To illustrate my model's generality and effectiveness *quantitatively* (Sec. 4.7.1) and *qualitatively* (Sec. 4.7.2), I compare the results of my material estimation pipeline with the baselines and previous works. (2) I conduct a user study to investigate material perception using this work and quantify the similarity between the measurements of real-world fabrics from lab experiments and predicted garment materials from videos (Sec. 4.7.3). (3) I conduct additional comparison with other related learning-based methods, including ablation studies and latent code for garment interpolation/design, and application to virtual try-on.

4.7.1 Quantitative Analysis

Mine vs. Image-Only [21]. Due to the difference regarding the input distribution (dressed garment on a human body in my method vs. hanging cloth in theirs), I re-train their model on my datasets for a fair comparison. I study the contribution of image-only features vs. garment-only features, as well as CNN vs. LSTM (that exploits the temporal coherence). Finally, I compare the overall performance difference between mine and [21]. The test classification accuracy is reported in Table 4.1.

Findings: (1) While all three models have learned the relationship between motion and materials and all three outperform random guess, *the garment feature signals are shown to be much more important than the image features.* This finding is not surprising, since the garment shape is directly affected by the material. (2) Combining the two features, as my model does, further improves the test accuracy. A possible reason is that an overall capturing of the garment shape (*e.g.* width and length of the entire piece), which is difficult to retrieve using garment latent codes, could be more easily extracted using image features. (3) By exploiting temporal coherence, unsurprisingly all three versions of the model achieve better accuracy than only using 1 image.

Method	Mean Accuracy	Temporal Gain	Garment Features Gain
Random guess	1.85%	-	-
Image only, CNN	5.11%	40.16%	-
Image only, LSTM [21]	45.27%		-
Garment only, CNN	11.85%	53.31%	6.74%
Garment only, LSTM	65.16%		19.89%
Image + Garment, CNN	12.62%	**57.52%**	7.51%
Image + Garment, LSTM **(mine)**	**70.14%**		**24.87%**

Table 4.1: **Comparison on material estimation**: my method achieves much higher accuracy (∼50%) in classification than [21].

Mine vs. Optimization-based [110]: An optimization-simulation framework to obtain the fabric material parameters using wrinkle density of the garment in a single image was proposed in [110]. In contrast, my method extracts both static image features and spatio-temporal *garment features* across frames. I generate the same set of test scenes as shown in Fig. 4.9. My model is tested on these sequences under varying lighting and visibility conditions; the average accuracy is reported in

Table 4.2. In this challenging case where the lighting condition and the textures are not seen in the training distribution, my method still achieves comparable accuracy with previous method [110], but it runs **more than 1,000x faster**. *Mine is the first end-to-end learning-based method to predict fabric materials directly from a video of garments worn on a human body.*

Method	Accuracy (%)						Speed
	Mid-day			Sunset			
	T-shirt	Pants	Skirt	T-shirt	Pants	Skirt	
[110]	80.2	80.2	**83.3**	81.6	79.9	**80.7**	4-6 hours
Mine	**86.5**	**91.6**	81.6	**82.4**	**91.6**	79.6	**8.7s**

Table 4.2: **Quantitative comparison with [110]**. My method achieves comparable or higher accuracy, but runs at least **three orders of magnitude** faster than the state-of-the-art [110].

4.7.2 Qualitative Results

I compare mine with the most relevant work of [110] for joint estimation of garment shapes and materials, as shown in Fig. 4.5. My method achieves similar reconstruction accuracy and visual quality as [110]. But, [110] uses semantic segmentation, thus suffering from tedious manual processing and long inference time. In contrast, my learning-based method is fully automatic and can compute the prediction in real time. Moreover, my method does not assume the sewing patterns as a prior.

I further compare with several learning methods [10, 118] in Sec. 4.7.6. Many often use additional information (*e.g.* mesh templates or known garment types) as priors, so direct visual comparison is not meaningful. Nonetheless, my model successfully generalizes to unseen real-world images/videos with comparable visual results, as shown in Fig. 4.13. During these experiments, my method is directly applied without any fine-tuning or post-optimization. Although trained using synthetic datasets, my model correctly identifies people and the garments from real-world images, and achieves similar visual results in all examples, when compared with previous works. The network is also capable of the predicting correct sizes of garments relative to the body, due to multiscale auto-encoders.

Material Cloning for Virtual Try-On: I show three application scenarios of my method. In Fig. 4.6, given an RGB video of a person wearing garments of different fabric materials, my method can identify the underlying material and *clone* it onto other garment models using cloth simulation. My method is the first to achieve fast and accurate material extraction from videos of dressed garments on a body. I further show the ability of my method to reconstruct the entire human appearance from the input video using one single network. I first estimate the body and the garment geometry frame by frame, and use the temporal information to infer the material. The three parts are combined using cloth simulation to generate the final output. Fig. 4.7 and Fig. 4.8 show the reconstruction results (also see the supplement video). My reconstructed garment shapes and wrinkles match those in the input video frames.

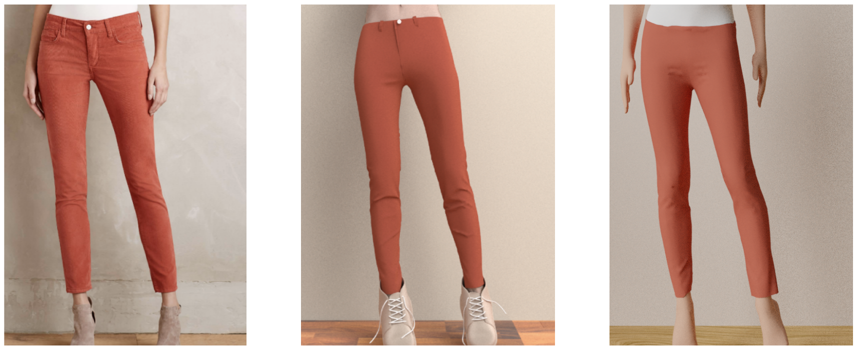
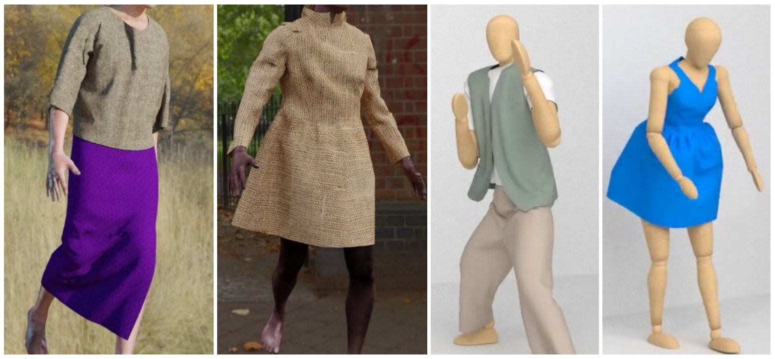

(a) Input image (b) Results from [110] (c) My results

Figure 4.5: **Qualitative comparison with** [110]: Mine is easier to use and achieves visually comparable reconstruction much faster without priors on garment patterns and topology.


(a) Soft silk (b) Stiff polyester (c) Soft silk (d) Stiff polyester

Input videos **Estimated materials**

Figure 4.6: **Material transfer between videos**. My method can take videos of a person wearing any garments (a, b) and clone the underlying fabric materials onto other simulated garments (c, d).

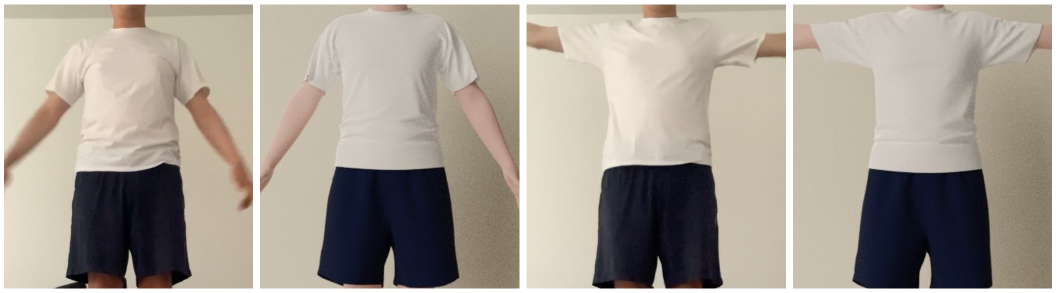

(a) Real (b) Virtual (c) Real (d) Virtual

Figure 4.7: **Qualitative results:** my method faithfully recovers the T-shirt materials (a, c) in video so that the wrinkles around the simulated t-shirt sleeves (b, d) appear similar under different poses.

4.7.3 Lab Experiments and User Study

In this experiment, I test the prediction accuracy of my method using real-world materials. I used five real-world materials measured from lab experiments [93], which are sampled from sweater, t-shirt, tablecloth, jeans, and blanket, respectively. The materials are used to create videos using the same pipeline as I generate the training data. The videos are then input to my network model for material estimation, which is used to generate the resulting videos with other conditions being held the same (see Fig. 4.16 for sample frames).

I first quantify my material recovery using the sensitivity analysis described in [21]. For each of the materials, I measure its stiffness ratio according to the deformations under a fixed amount of external forces. The measured values are then compared with the one predicted from my method, reported in Table 4.3. My method achieves a relatively small error between 9.5% to 16.7%.

Material Name [93]	Stretching Ratio (GT/Prediction)	Bending Ratio (GT/Prediction)	Mean Relative Error
gray-interlock	1.01/1	1.6/2	10.5%
navy-sparkle-sweat	0.56/0.5	1.7/2	12.8%
white-dots-on-blk	15.8/20	3.5/4	16.7%
11oz-black-denim	3.6/3	3.0/3	8.3%
pink-ribbon-brown	2.93/3	12/10	9.5%

Table 4.3: **Lab experiment results.** My material estimation achieves relatively small errors compared to lab measurements on all real-world materials tested.

To further validate and quantify the material similarity, I conduct a user study to examine how close my estimation results are to the ground-truth data in human perception. In the study, I place two videos side by side for each material; then ask each participant to rate the level of material similarity: from 0 (totally different) to 10 (identical). There are 25 subjects in my study group: 17 male and 8 female, with age ranging from 20 to 40. To further calibrate the subjective score range, I use a pair of videos generated from the *same material* but with *different mesh resolutions* and inform the participants that this example has a similarity score of 5 for calibration. My results show that the average similarity ratings for five tested materials vs. the ground-truth data are all larger than 5, ranging from 5.7 to 8.5, with an overall mean value of 7.1. These indicate that my method indeed can recover fabric materials with only minor perceptible differences to the real-world materials.

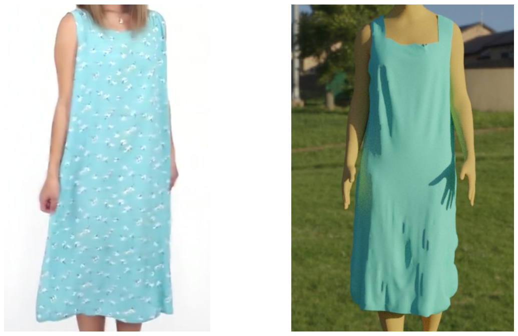
Besides the main results, I also conducted several other studies to investigate how people perceive the garment materials in different environmental conditions:

Garment color and texture. I changed the garment colors and textures; then asked participants the same questions. I found that by varying the garment colors, either brighter or darker, does not affect the similarity scores much – within a maximum difference of 0.4. On the other hand, changing the textures results in more perceptible effect – with similarity score differences of 0.5 to 0.9.

Lighting. I varied the lighting conditions in the rendered results to understand how shading on the garments affects material perception. The results show that the similarity scores are decreased by 1.1-1.3 when one of the videos has a different lighting angle than the other. These noticeable difference indicate the effects of lighting and shading on how humans perceive wrinkles and folds.

Stiffness range. I took the material called 'gray-interlock', consisting of 60% cotton and 40% polyester, and multiplied its material parameters by 1, 2, 5, and 10, respectively. The participants were asked to distinguish which of the two sampled materials is stiffer. My findings indicate that there is little perceptible visual difference between lower stiffness values till when the garment stiffness is increased to a certain threshold. This finding also reconfirms my design rationale of the material space discretization using sensitivity analysis in Sec. 4.3.

Additional Results. I present more comparisons with other related learning-based methods for garment shape and pose recovery in Appendices. Please note that none of these methods was designed to *recover the fabric materials*, the main focus and motivation of this work for simulation-based virtual try-on, as shown in Fig. 4.8.



(a) Input video (b) Reconstruction result

Figure 4.8: **Qualitative comparison with a real-world video.**

4.7.4 Ablation Study

Method	CD	SD
Single-scale	0.31	8.05
Multi-scale	**0.12**	**1.03**

(a) Auto-encoder

Method	MPJPE	CD
w/o feedback	62.93	0.89
w/ feedback	**55.20**	**0.88**

(b) Human and garment estimation

Table 4.4: **Ablation study for different parts of my proposed network.** My method is marked in Italic. CD stands for errors in Chamfer Distance; SD stands for Sinkhorn Divergence [150]; and MPJPE stands for Mean Per Joint Position Error – all in millimeters (mm). My method results in notably smaller errors than all baselines in the estimation and reconstruction tasks.

(a) T-shirt and pants (day)

(b) T-shirt and skirt (day)

(c) T-shirt and pants (night)

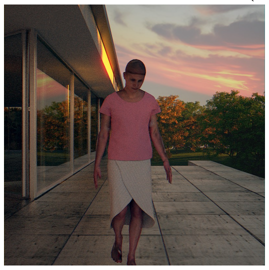

(d) T-shirt and skirt (night)

Figure 4.9: **Sample test images for the comparisons with [110].**

In the following ablation studies, I verify the effectiveness of my network model. I use the test errors to compare my model with the baselines, which include previous methods or their combinations. During the test, all other conditions are held the same, except for the network structure itself.

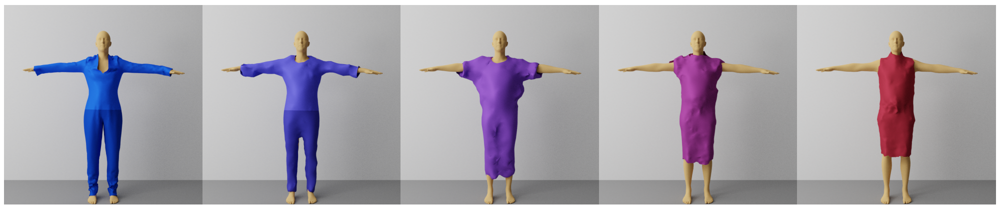


Figure 4.10: **Interpolation between different garments**. w is the interpolation weight. I extract the latent code of 2 different garments: a long-sleeve top & pants (leftmost) and a short-sleeve dress (rightmost). I use the decoder to produce new garments of linearly-interpolated latent codes, enabling smooth transitioning between topologically different garments not achieved before.

Garment auto-encoder. I use a two-level auto-encoder structure to address the highly dynamic geometry of garments, as discussed in Sec. 4.4. I demonstrate here

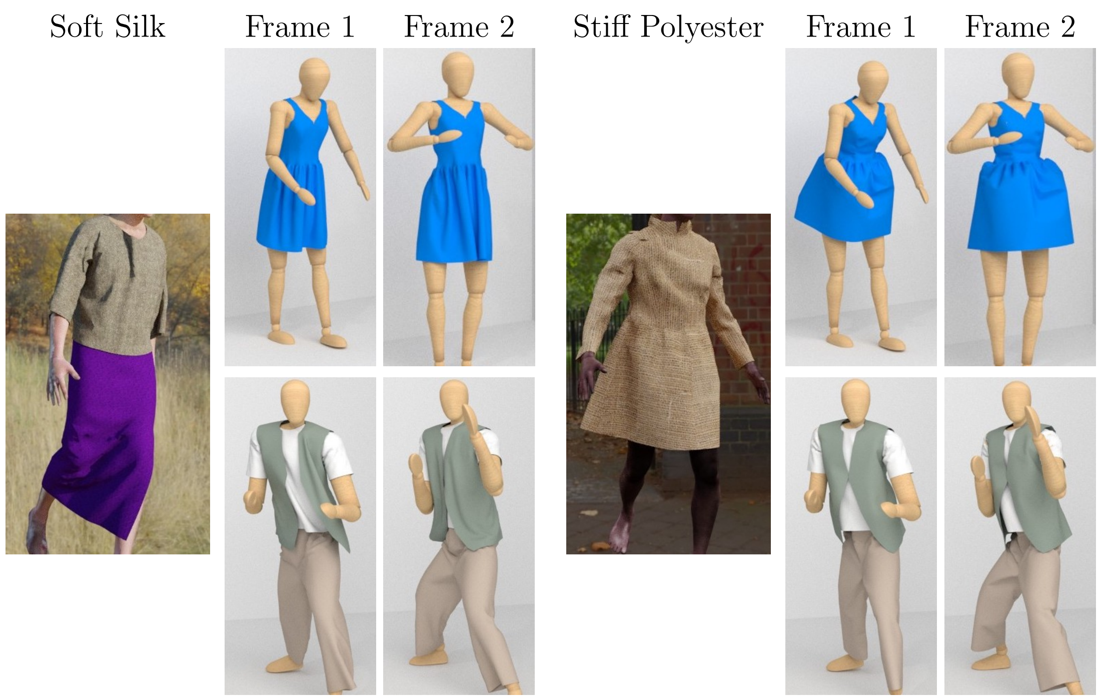


Figure 4.11: Material transfer examples. My method can accurately estimate the material parameter from the input video and replicate the same effect in other animations. Note that different estimated materials create significantly different visual appearance even for the same animation.

its effectiveness compared to the baseline, which simply consists of a PointNet [99] encoder and an AtlasNet [141] decoder. There are two theoretical advantages of my method against the baseline. First, the two-level structure partitions the entire point cloud by patches that only overlap by a small fraction. This approach makes the local features more focused on its local shape rather than on a more expanded surface. Next, the two-level network also offers more capability to express detailed features and prevents overly-smooth reconstructions.

In Table 4.4(a), I report my test errors compared to the baseline. In addition to Chamfer Distance as I used in the training, I also use Sinkhorn Divergence [150], which is a fast approximator for computing Earth Moving Distance between two distributions. While Chamfer Distance indicates the average distance between two point clouds, Sinkhorn Divergence captures the density difference across space. The numbers in the table show that my method not only reproduces point clouds closer to the ground-truth but also has more evenly-distributed points due to my patch-wise partition during training. The results align with my theoretical analysis.

Garment geometry estimation. In Sec. 4.5.1, I proposed a closed-loop feedback structure to improve the estimation of both the human body and garment shape. I conducted an ablation study to show the difference introduced by this structure. My baseline is a two-branch estimation block that predicts the human body and the garment in parallel but sharing the image features as input. Although it does not have the feedback correction, it already benefits from multi-tasking which can extract useful common image features. I use Mean Per Joint Position Error (MPJPE) for human estimation metric, and Chamfer Distance for garment estimation.

As shown in Table 4.4(b), by introducing an extra feedback loop, my model results in a much smaller error on human body estimation. Interestingly, the garment estimation error is not greatly improved, possibly because the error comes more from local shapes (especially those occluded by the human) than the global one.

I further compare my method with a recent work [10] since their task is the most relevant to mine. I use the public dataset from Multi-Garment-Net [10], which consists of 95 scans of people. Nearly half of them are not suitable due to incorrect or incomplete garment labeling. I tested my model in the dataset without any fine-tuning and used their reported numbers for comparison. I use the Chamfer Distance defined in Eqn. 4.5 as my test metric. As shown in Table 4.5, my model, without any fine-tuning or domain adaptation, achieves the smallest error – whether the ground-truth human body model is provided or not during reconstruction. Since my model is trained on a wider range of body poses and garment types than theirs while achieving better accuracy, it has shown to offer good generality to unseen inputs.

Methods	[68]	[10]		Mine	
	GT Pose	GT Pose	Full Pred.	GT Pose	Full Pred.
Pants (mm)	5.44	5.57	10.16	**1.58**	**3.08**
Short Pants (mm)	8.23	5.97	10.00	**4.92**	**5.69**
T-shirt (mm)	5.80	5.63	11.97	**1.67**	**3.08**
Shirt (mm)	5.71	6.33	9.05	**2.29**	**3.75**
Coat (mm)	5.85	5.66	9.09	**2.84**	**3.65**

Table 4.5: **Test errors on the Multi-Garment Net dataset [10].** My method achieves the best estimation accuracy across all garment types, without any fine-tuning or reference body.

4.7.5 Latent Code Interpolation

To showcase the expressiveness of my learned latent space, I conducted an experiment generating new garments by interpolation. I encode two different garments to obtain their latent codes and linearly interpolate in between. I generate the new point clouds using the interpolated results accordingly. The visual results are shown in Fig. 4.10. My results show that the interpolations represent a smooth transition between the two original garments, creating new garment styles that are not seen before during training. Note that modeling long dresses or garments with different mesh structures are not achieved in previous works, which either use displacement maps [10, 11] (thus not able to model dresses) or mesh-CNN for encoding local features [8] (thus not applicable to different mesh topologies). My method is the first to propose *a feature space that unifies garments of different topologies with different body poses and shapes*, which is the key component to accurate garment material estimation, as demonstrated in Table 4.1.

I provide more interpolation results in Fig. 4.12. As described in Sec. 4.7.5, my algorithm enables two garment meshes to be interpolated using their latent code to generate new garments. The first row shows an example where the dress is shorten at the bottom and extended at the sleeves gradually. I also show interpolation results in the point cloud form in Row 2-4. The second row is another example of transformation from dress to pants, and the last two examples demonstrate the ability to interpolate the garment with different body poses. In these two cases, the garment type is the same, but the body pose is changing. Although the poses in between are never seen in the training set, the interpolated garment point clouds follow the pose transition, showing very little interpenetrations with the body. Note that when the human legs are moving, the garment correctly deforms with the pose, close enough to be consistent with the body motion, while still being collision-free. See the supplementary video for the interpolation animation.

4.7.6 Additional Qualitative Results

I tested my model without any fine-tuning on the images provided in MGN [10] and DeepCap [118], as shown in Fig 4.13. Although my model is never exposed

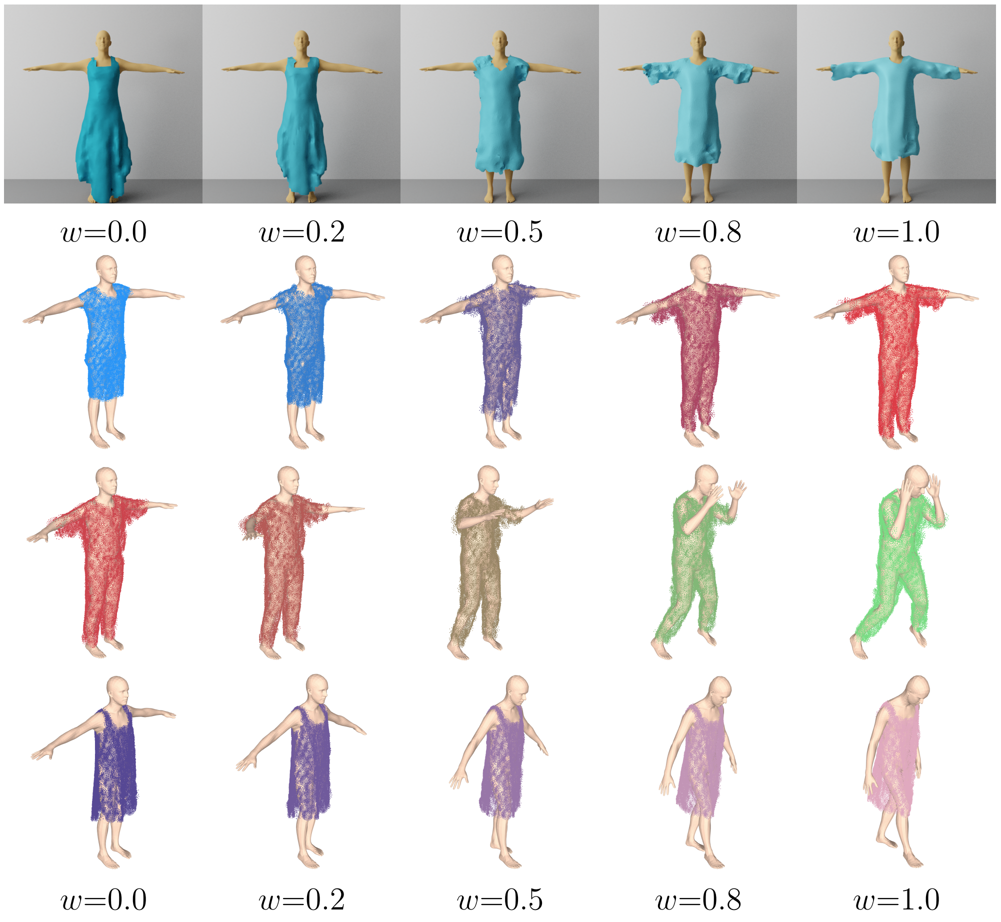


Figure 4.12: **Interpolation results**. My method can smoothly interpolate garments between different topologies and body poses.

to real-world images, it successfully predicts the human body pose and garment shape correctly. Note that my model is an end-to-end network that has no prior knowledge to any information about the garment shape, while MGN assumes that there is a one-to-one approximation mapping between body vertices and garment vertices. DeepCap has an initial ground-truth mesh that is to be optimized. My model can thereby easily generalize to unseen garments, which is not possible using these two works. In Table 4.6, I extensively compare my work with previous ones regarding different assumptions, functionalities, and abilities. I define 'one model per garment' in 'generality' as that the method needs to create extra templates or registrations to the body, or need to retrain part of the network in order to predict a different garment type. Although DeepFashion3D [120] and ARCH [119] also have generality to different topologies to some extent, there are still limitations in their pipeline. The output from DeepFashion3D has to be continuous in one body part, meaning that they cannot support all topologies (*e.g.* dresses with holes). ARCH does support different garments on the body, but the output is a water-tight mesh together with the body, which is not always convenient for certain applications like virtual try-on. In contrast, my method naturally supports all kinds of topologies, and predicts the body and the garment in separate meshes.

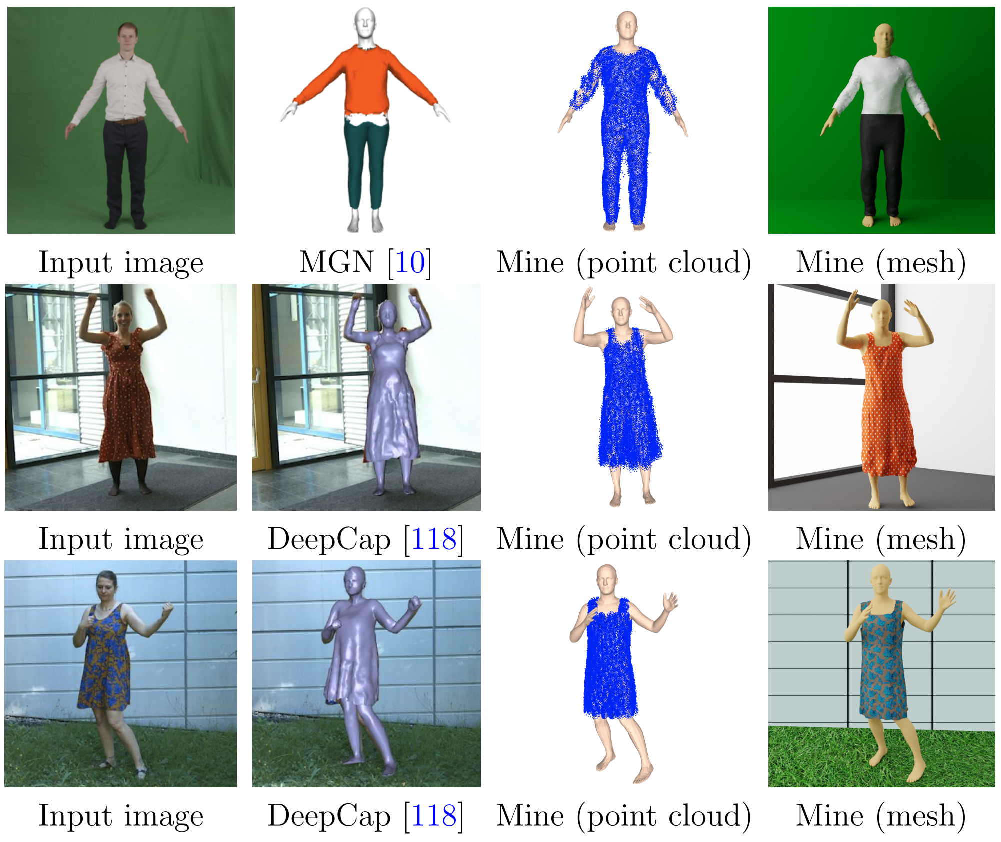


Figure 4.13: **Qualitative Results**. My model can achieve similar visual results with previous work without any knowledge of the target garment or any assumption of the topology (See Table 4.6).

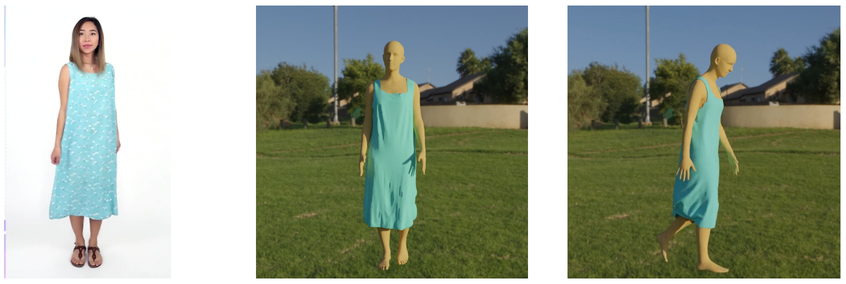
Method	Input	Dependencies	Generality	Dresses support	Separate mesh	Material Estimation
MGN [10]	Semantic seg. + 2D joints	Garment correspondences	One model per garment	No	Yes	No
DeepCap [118]	Foreground seg.	Template mesh	One model per garment	Yes (w/ known template)	No	No
[110]	Semantic seg.	Template mesh	One model per garment	Yes (w/ known template)	Yes	Yes
DeepFashion3D [120]	RGB frame (garment only)	None	One model for all	Yes (limited topologies)	Yes	No
Tailornet [11]	Body parameters	Garment correspondences	One model per garment	Yes (limited topologies)	Yes	No
BCNet [121]	RGB frame	Garment correspondences	One model per garment	Yes (limited topologies)	Yes	No
SIZER [123]	Body scan	Garment labels	One model per garment	No	Yes	No
ARCH [119]	RGB frame (foreground only)	None	One model for all	Yes (water-tight)	No	No
Mine	RGB frame	None	One model for all	Yes	Yes	Yes

Table 4.6: **Comparison with previous works**. My method can handle the largest set of garments, using fewest possible information (*i.e.* RGB image), in one standalone, end-to-end network.

4.7.7 Application: Virtual Try-On

To further showcase the strength of my network, I apply it to a virtual try-on application. An online video clip showing a person wearing a dress is taken as input to my network (Fig. 4.14a). The body and the dress are estimated in each frame, and the fabric material is inferred using the garment motion and the image features. As shown in Fig. 4.14b, my method successfully infers the correct human body and the garment.

I then simulate the garment in a different body motion, which is the key functionality in virtual try-on systems. The simulated results (Fig. 4.14c) show that the garment motion provides similar visual impression with the input dress (mostly from the wrinkle motions of the dress). This example shows that my method can effectively extract the correct type of fabric material and transfer the given fabric material in a video to a simulation-based virtual try-on system. More animation results can be found in Fig. 4.11 and the supplementary video.



(a) Input video clip (b) Estimated result (c) Simulation result

Figure 4.14: **Virtual try-on example**. My network model can clone a person's appearance from the physical world (in a video) to the virtual world, enabling simulation under different motions with accurate estimations of the body shape, garment geometry, and fabric materials.

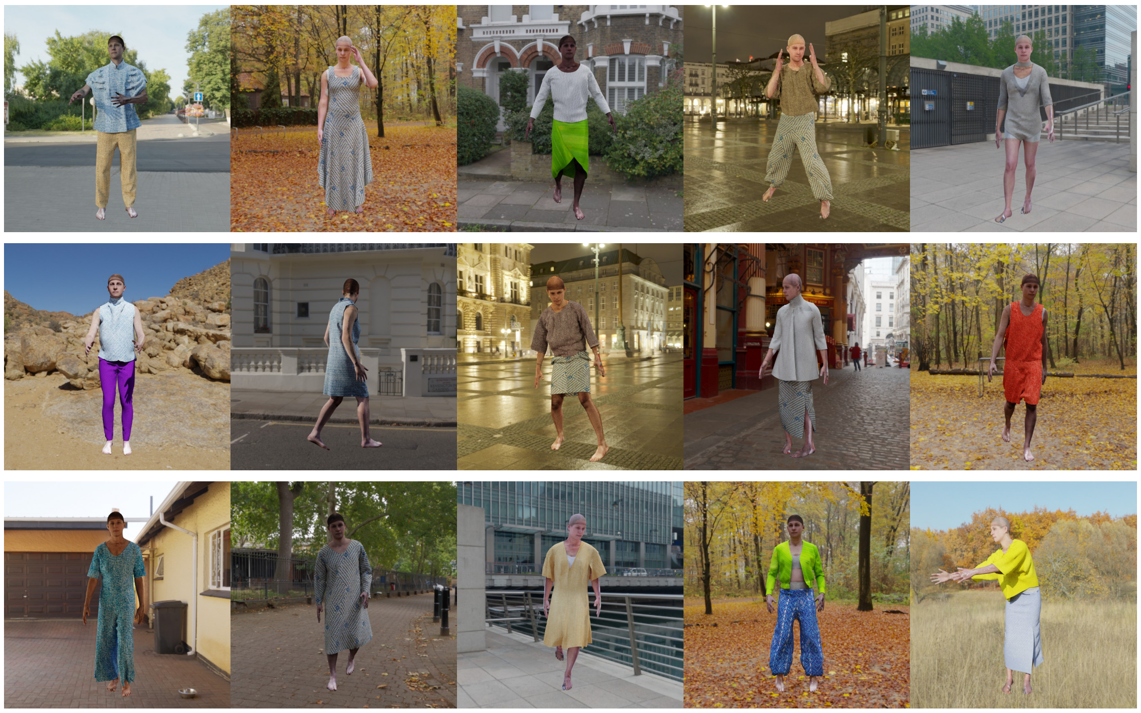


Figure 4.15: **Training data examples**. My dataset includes various garment topologies with rich body poses, textures, and background environments. Some examples are shown here.

4.8 Conclusion

In this chapter, I introduced an end-to-end learning model for garment material estimation using RGB videos. I do not assume other inputs (e.g. segmentation, 3D

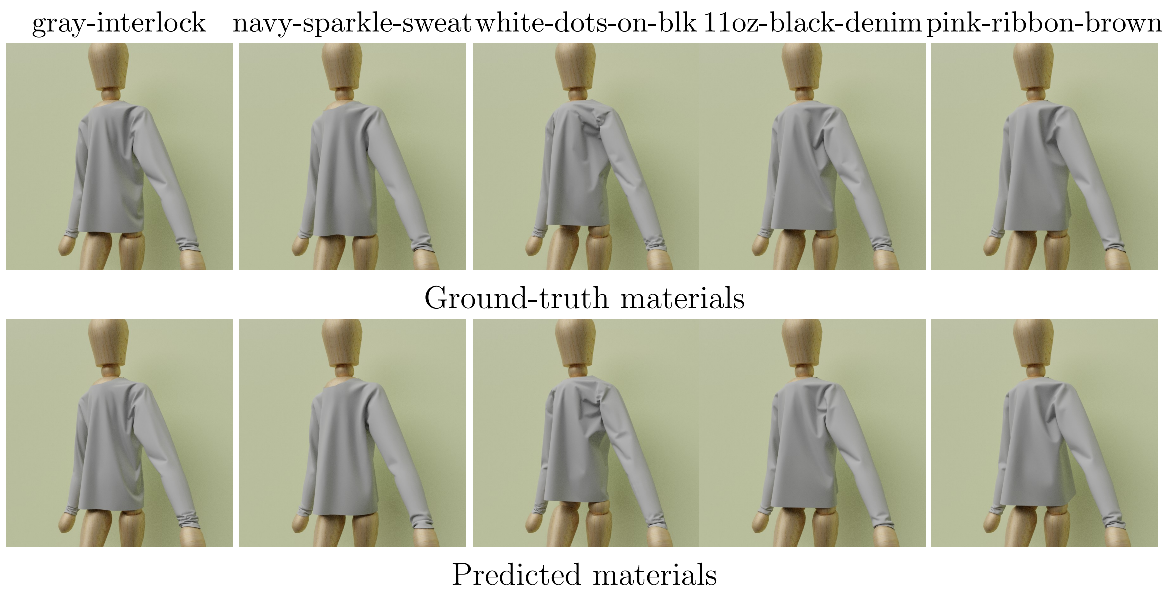


Figure 4.16: **User study examples**. My predictions received similarity scores of 5.7 to 8.5 in a 0-10 range.

scans, multi-views, etc.) or any prior knowledge on the garment shape/topology, design patterns/templates, or correspondences. I extract the multi-scale features to effectively represent the dynamic geometry structure of garments, which can be combined with image features to estimate fabric materials by learning their temporal patterns, while improving the human body reconstruction using a feedback loop. This approach is perhaps the first to introduce a unified parametric model for all garment types, and it can thereby support garments of different topologies without the need to retrain different models. Experiments show that my method achieves much higher accuracy up to 70.14% in estimating fabric materials than prior works, while offering capabilities in recovering garment types and topologies with generality and simplicity for an unification of multiple correlated tasks.

One limitation of this method is that the current representation does not support multi-layer or folded garments. These issues can be addressed by adding structural prior to the garment model to encode multi-layer clothing and curvature representation to support multi-fold features. I further postulate that the proposed multiscale garment auto-encoder can also be integrated with neural rendering [151] to synthesize photorealistic images of simulated garments.