Cloth2Tex: A Customized Cloth Texture Generation Pipeline for 3D Virtual Try-On

Daiheng Gao[1*] **Xu Chen**[2,3*] **Xindi Zhang**[1] **Qi Wang**[1]
Ke Sun[1] **Bang Zhang**[1] **Liefeng Bo**[1] **Qixing Huang**[4]
[1]Alibaba XR Lab [2]ETH Zurich, Department of Computer Science
[3]Max Planck Institute for Intelligent Systems [4]The University of Texas at Austin

Figure 1. We propose **Cloth2Tex**, a novel pipeline for converting 2D images of clothing to high-quality 3D textured meshes that can be draped onto 3D humans. In contrast to previous methods, Cloth2Tex supports a variety of clothing types. Results of 3D textured meshes produced by our method as well as the corresponding input images are shown above.

Abstract

Fabricating and designing 3D garments has become extremely demanding with the increasing need for synthesizing realistic dressed persons for a variety of applications, e.g. *3D virtual try-on, digitalization of 2D clothes into 3D apparel, and cloth animation. It thus necessitates a simple and straightforward pipeline to obtain high-quality texture from simple input, such as 2D reference images. Since traditional warping-based texture generation methods require a significant number of control points to be manually selected for each type of garment, which can be a time-consuming and tedious process. We propose a novel method, called* ***Cloth2Tex****, which eliminates the human burden in this process. Cloth2Tex is a self-supervised method that generates texture maps with reasonable layout and structural consistency. Another key feature of Cloth2Tex is that it can be used to support high-fidelity texture inpainting. This is done by combining Cloth2Tex with a prevailing latent diffusion model. We evaluate our approach both qualitatively and quantitatively and demonstrate that Cloth2Tex can generate high-quality texture maps and achieve the best visual effects in comparison to other methods. Project page: xxx*

1. Introduction

The advancement of AR/VR and 3D graphics has opened up new possibilities for the fashion e-commerce industry. Customers can now virtually try on clothes on their avatars in 3D, which can help them make more informed purchase decisions. However, most clothing assets are currently presented in 2D catalog images, which are incompatible with 3D graphics pipelines. Thus it is critical to produce 3D clothing assets automatically from these existing 2D images, aiming at making 3D virtual try-on accessible to everyone.

Towards this goal, the research community has been developing algorithms [19, 20, 37] that can transfer 2D images into 3D textures of clothing mesh models. The key

Figure 2. Problem of warping-based texture generation algorithm: partially filled UV texture maps with large missing holes as highlighted in yellow.

to producing 3D textures from 2D images is to determine the correspondences between the catalog images and the UV textures. Conventionally, this is achieved via the Thin-Plate-Spline (TPS) method [3], which approximates the dense correspondences from a small set of corresponding key points. In industrial applications, these key points are annotated manually and densely for each clothing instance to achieve good quality. With deep learning models, automatic key point detectors [19, 35] have been proposed to detect key points automatically for clothing. However, as seen in Fig. 2, the inherent self-occlusions (*e.g.* sleeves occluded by the main fabric) of TPS warping-based approaches are intractable, leading to erroneous and incomplete texture maps. Several works have attempted to use generative models to refine texture maps. However, such a refinement strategy has demonstrated success only in a small set of clothing types, *i.e.* T-shirts, pants, and shorts. This is because TPS cannot produce satisfactory initial texture maps on all clothing types, and a large training dataset covering high-quality texture maps of diverse clothing types is missing. Pix2Surf [20], a SMPL [18]-based virtual try-on algorithm, has automated the process of texture generation with no apparent cavity or void. However, due to its clothing-specific model, Pix2Surf is limited in its ability to generalize to clothes with arbitrary shapes.

This paper aims to automatically convert 2D reference clothing images into 3D textured clothing meshes for a larger diversity of clothing types. To this end, we first contribute template mesh models for 10+ different clothing types (well beyond current SOTAs: Pix2Surf (**4**) and [19] (**2**)). Next, instead of using the Thin-Plate-Spline (TPS) method as previous methods, we incorporate neural mesh rendering [17] to directly establish dense correspondences between 2D catalog images and the UV textures of the meshes. This results in higher-quality initial texture maps for all clothing types. We achieve this by optimizing the 3D clothing mesh models and textures to align with the catalog images' color, silhouette, and key points.

Although the texture maps from neural rendering are of higher quality, they still need refinement due to missing regions. Learning to refine these texture maps across different clothing types requires a large dataset of high-quality 3D textures, which is infeasible to acquire. We tackle this problem by leveraging the recently emerging latent diffusion model (LDM) [24] as a data simulator. Specifically, we use ControlNet [39] to generate large-scale, high-quality texture maps with various patterns and colors based on its *canny edge* version. In addition to the high-quality ground-truth textures, the refinement network requires the corresponding initial defective texture maps obtained from neural rendering. To get such data, we render the high-quality texture maps into catalog images and then run our neural rendering pipeline to re-obtain the texture map from the catalog images, which now contain defects as desired. With these pairs of high-quality complete texture maps and the defective texture maps from the neural renderer, we train a high-resolution image translation model that refines the defective texture maps.

Our method can produce high-quality 3D textured clothing from 2D catalog images of various clothing types. In our experiments, we compare our approach with state-of-the-art techniques of inferring 3D clothing textures and find that our method supports more clothing types and demonstrates superior texture quality. In addition, we carefully verify the effectiveness of individual components via a thorough ablation study.

In summary, we contribute **Cloth2Tex**, a pipeline that can produce high-quality 3D textured clothing in various types based on 2D catalog images, which is achieved via

- *a)* 3D parametric clothing mesh models of 10+ different categories that will be publicly available,
- *b)* an approach based on neural mesh rendering to transferring 2D catalog images into texture maps of clothing meshes,
- *c)* data simulation approach for training a texture refinement network built on top of blendshape-driven mesh and LDM-based texture.

2. Related Works

Learning 3D Textures. Our method is related to learning texture maps for 3D meshes. Texturify [27] learns to generate high-fidelity texture maps by rendering multiple 2D images from different viewpoints and aligning the distribution of rendered images and real image observations. Yu *et al.* [38] adopt a similar method, rendering images from different viewpoints and then discriminating the images by separate discriminators. With the emergence of diffusion

models [7, 31], recent work Text2Tex [5] exploits 2D diffusion models for 3D texture synthesis. Due to the mighty generalization ability of the diffusion model [11, 24] trained on the largest corpus LAION-5B [26], *i.e*. stable diffusion [24], the textured meshes generated by Text2Tex are of superior quality and contain rich details. Our method is related to these approaches in that we also utilize diffusion models for 3D texture learning. However, different from previous approaches, we use latent diffusion models only to generate synthetic texture maps to train our texture inpainting model, and our focus lies in learning 3D texture corresponding to a specific pair of 2D reference images instead of random or text-guided generation.

Texture-based 3D Virtual Try-On. Wang *et al*. [34] provide a sketch-based network that infers both 2D garment sewing patterns and the draped 3D garment mesh from 2D sketches. In real applications, however, many applications require inferring 3D garments and the texture from 2D catalog images. To achieve this goal, Pix2Surf [20] is the first work that creates textured 3D garments automatically from front/back view images of a garment. This is achieved by predicting dense correspondences between the 2D images and the 3D mesh template using a trained network. However, due to the erroneous correspondence prediction, particularly on unseen test samples, Pix2Surf has difficulty in preserving high-frequency details and tends to blur out fine-grained details such as thin lines and logos.

To avoid such a problem, Sahib *et al*. [19] propose to use a warping-based method (TPS) [3] instead and to use further a deep texture inpainting network built upon MADFNet [40]. However, as mentioned in the introduction, warping-based methods generally require dense and accurate corresponding key points in images and UV maps and have only demonstrated successful results on two simple clothing categories, T-shirts and trousers. In contrast to previous work, Cloth2Tex aims to achieve automatic high-quality texture learning for a broader range of garment categories. To this end, we use neural rendering instead of warping, which yields better texture quality on more complex garment categories. We further utilize latent diffusion models (LDMs) to synthesize high-quality texture maps of various clothing categories to train the inpainting network.

3. Method

We propose Cloth2Tex, a two-stage approach that converts 2D images into textured 3D garments. The garments are represented as polygon meshes, which can be draped and simulated on 3D human bodies. The overall pipeline is illustrated in Fig. 3. The pipeline's first stage (Phase I) is to determine the 3D garment shape and coarse texture. We do this by registering our parametric garment meshes onto catalog images using a neural mesh renderer. The pipeline's second stage (Phase II) is to recover fine textures from the coarse estimate. We use image translation networks trained on large-scale data synthesized by pre-trained latent diffusion models. The mesh templates for individual clothing categories are a pre-requirement for our pipeline. We obtain these templates by manual artist design and will make them publicly available.

Implementation details are placed in the supp. material due to the page limit.

3.1. Pre-requirement: Template Meshes

For the sake of both practicality and convenience, we design cloth template mesh (with fixed topology) $\mathcal{M}$ for common garment types (*e.g*., T-shirts, sweatshirts, baseball jackets, trousers, shorts, skirts, and *etc*.). We then build a deformation graph $\mathcal{D}$ [29] to optimize the template mesh vertices. This is because per-vertex image-based optimization is subject to errors and artifacts due to the high degrees of freedom. Specifically, we construct $\mathcal{D}$ with k nodes, which are parameterized with axis angles $\mathbf{A} \in \mathbb{R}^3$ and translations $\mathbf{T} \in \mathbb{R}^3$. The vertex displacements are then derived from the deformation nodes (the number of nodes k is dependent on the garment type since different templates have different numbers of vertices and faces). We also manually select several vertices on the mesh templates as landmarks $\mathcal{K}$. The specific requirements of the template mesh are as follows: vertices V less than 10,000, uniform mesh topology, and integrity of UV. The vertex number of all templates ranges between **skirt** (*6,116*) to **windbreaker** (*9,881*). For uniformity, we set the downsampling factor of $\mathcal{D}$ for all templates to 20 (details of template meshes are placed in the supp. material). The integrity of UV means that the UV should be placed as a whole in terms of front and back, without further subdivision, as used in traditional computer graphics. Fabricating integral UV is not complicated and can be a super-duper candidate for later diffusion-based texture generation. See Sec. 3.3.1 for more details.

3.2. Phase I: Shape and Coarse Texture Generation

The goal of Phase I is to determine the garment shape and a coarse estimate of the UV textures $\mathcal{T}$ from the input catalog (*Front & Back* view). We adopt a differentiable rendering approach [17] to determine the UV textures in a self-supervised way without involving trained neural networks. Precisely, we fit our template model to the catalog images by minimizing the difference between the 2D rendering of our mesh model and the target images. The fitting procedure consists of two stages, namely *Silhouette Matching* and *Image-based Optimization*. We will now elaborate on these stages below.

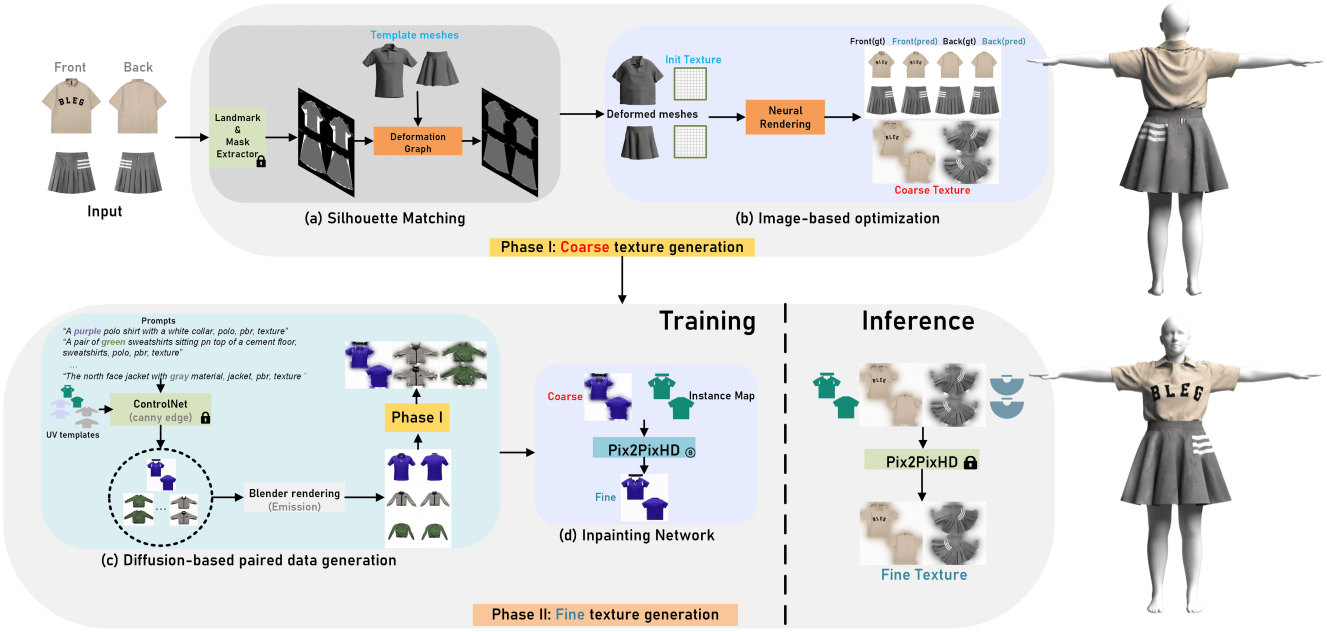

Figure 3. **Method overview**: Cloth2Tex consists of two stages. In Phase I, we determine the 3D garment shape and coarse texture by registering our parametric garment meshes onto catalog images using a neural mesh renderer. Next, in Phase II, we refine the coarse estimate of the texture to obtain high-quality fine textures using image translation networks trained on large-scale data synthesized by pre-trained latent diffusion models. Note that the only component that requires training is the inpainting network. Please watch our video on the project page for an animated explanation of Cloth2Tex.

3.2.1 Silhouette Matching

We first align the corresponding template mesh to the 2D images based on the 2D landmarks and silhouette. Here, we use BCRNN [35] to detect landmarks L_{2d} and DenseCLIP [22] to extract the silhouette M. To fit our various types of garments, we finetune BCRNN with 2,000+ manually annotated clothing images per type.

After the mask and landmarks of the input images are obtained, we first perform a global rigid alignment by an automatic cloth scaling method to adjust the scaling factor of mesh vertices according to the overlap of the initial silhouette between mesh and input images, which ensures a rough agreement of the yielded texture map (See Fig. 8). Specifically, we implement this mechanism by checking the silhouette between the rendered and reference images, and then enlarging or shrinking the scale of mesh vertices accordingly. After an optimum **Intersection over Union(IoU)** has been achieved, we fix the coefficient and send the scaled template to the next step.

We then fit the silhouette and the landmarks of the template mesh (the landmarks on the template mesh are predefined as described in Sec. 3.1) to those detected from the 2D catalog images. To this end, we optimize the deformations of the nodes in the deformation graph by minimizing the following energy terms:

2D Landmark Alignment E_{lmk} measures the distance between 2D landmarks $L_{2\text{d}}$ detected by BRCNN and the 2D projection of 3D template mesh keypoints:

$$E_{\text{lmk}} = \| \prod \mathcal{K} - L_{2\text{d}} \|_2 \tag{1}$$

where $\prod$ denotes the 2D projection of 3D keypoints.

2D Silhouette Alignment E_{sil} measures the overlap between the silhouette of $\mathcal{M}$ and the predicted M from DenseCLIP:

$$E_{\text{sil}} = \text{MaskIoU}(S_{\text{proj}}(\mathcal{M}), M) \tag{2}$$

where $S_{\text{proj}}(\mathcal{M})$ is the silhouette rendered by the differentiable mesh renderer SoftRas [17] and *MaskIoU* loss is derived from Kaolin [9].

Merely minimizing E_{lmk} and E_{sil} does not lead to satisfactory results, and optimization procedure can easily get trapped into local minimums. To alleviate this issue, we introduce a couple of regularization terms. We first regularize the deformation using the as-rigid-as-possible loss E_{arap} [28] which penalizes the deviation of estimated local surface deformations from rigid transformations. Moreover, we further enforce the normal consistency E_{norm}, which measures normal consistency for each pair of neighboring faces). The overall optimization objective is given as

$$w_{\text{sil}} E_{\text{sil}} + w_{\text{lmk}} E_{\text{lmk}} + w_{\text{arap}} E_{\text{arap}} + w_{\text{norm}} E_{norm} \tag{3}$$

where w_* are the respective weights of the losses.

We set large regularization weights w_{arap}, w_{norm} at the initial iterations. We then reduce their values progressively during the optimization procedure, so that the final rendered texture aligns with the input images. Please refer to the supp. material for more details.

3.2.2 Image-based Optimization

After the shape of the template mesh is aligned with the image silhouette, we then optimize the UV texture map $\mathcal{T}$ to minimize the difference between the rendered image $I_{\text{rend}} = S_{\text{rend}}(\mathcal{M}, \mathcal{T})$ and the given input catalog images I_{in} from both sides simultaneously. To avoid any outside interference during the optimization, we only preserve the ambient color and set both diffuse and specular components to be zero in the settings of SoftRas [17], PyTorch3D [23].

Since the front and back views do not cover the full clothing texture, *e.g.* the seams between the front and back bodice can not be recovered well due to the occlusions, we use the total variation method [25] to fill in the blank of seam-affected UV areas. The total variation loss E_{tv} is defined as the norm of the spatial gradients of the rendered image $\nabla_x I_{\text{rend}}$ and $\nabla_y I_{\text{rend}}$:

$$E_{tv} = \|\nabla_x I_{\text{rend}}\|_2 + \|\nabla_y I_{\text{rend}}\|_2 \tag{4}$$

In summary, the energy function for the image-based optimization is defined as below:

$$w_{\text{img}}\|I_{\text{in}} - I_{\text{rend}}\|_2 + w_{\text{tv}} E_{\text{tv}} \tag{5}$$

where I_{in} and I_{rend} are the reference and rendered image. As shown in Fig. 3, $\mathcal{T}$ implicitly changes towards the final coarse texture $\mathcal{T}_{coarse}$, which ensures the final rendering is as similar as possible with the input. Please refer to our attached video for a vivid illustration.

3.3. Phase II: Fine texture generation

In Phase II, we refine the coarse texture from Sec. 3.2 and fill in the missing regions. Our approach takes inspiration from the strong and comprehensive capacity of Stable Diffusion (SD), which is a terrific model to have by itself in image inpainting, completion, and text2image tasks. In fact, there's also an entire, growing ecosystem around it: LoRA [12], ControlNet [39], textual inversion [10] and Stable Diffusion WebUI [1]. Therefore, a straightforward idea is to resolve our texture completion via SD.

However, we find poor content consistency between the inpainted blank and original textured UV. This is because UV data in our setting rarely appears in the training dataset LAION-5B [26] of SD. In other words, the semantic composition of LAION-5B and UV texture (cloth) are quite different and challenging for SD to generalize.

To address this issue, we first leverage ControlNet [39] to generate $\sim 2,000+$ HQ complete textures per template and render emission-only images under the front and back view. Next, we use Phase I again to recover the corresponding coarse textures. After collecting the pairs of coarse and fine textures, we train an inpainting network to fill the missing regions in the coarse texture maps.

3.3.1 Diffusion-based Data Generation

We employ diffusion models [7, 24, 39] to generate realistic and diverse training data.

We generate texture maps following the UV template configuration, adopting the pre-trained ControlNet with edge map as input conditions. ControlNet finetunes text-to-image diffusion models to incorporate additional structural conditions as input. The input edge maps are obtained through canny edge detection on clothing-specific UV, and the input text prompts are generated by applying image captioning models, namely Lavis-BLIP [16], OFA [32] and MPlug [15], on tens of thousands of clothes crawled from Amazon and Taobao.

After generating the fine UV texture maps, we are already able to generate synthetic front and back 2D catalog images, which will be used to train the impainting network. We leverage the rendering power of Blender native EEVEE engine to get the best visual result. A critical step of our approach is to perform data augmentation so that the impainting network captures invariant features instead of details that differ between synthetic images and testing images, which do not generalize. To this end, we vary the blend shape parameters of the template mesh to generate 2D catalog images in different shapes and pose configurations and simulate self-occlusions, which frequently exist in reality and lead to erroneous textures as shown in Fig. 2. We handcraft three common blendshapes (Fig. 4) that are enough to simulate the diverse cloth-sleeve correlation/layout in reality.

Next, we run Phase I to produce coarse textures from the rendered synthetic 2D catalog images, yielding the coarse, defect textures corresponding to the fine textures. These pairs of coarse-fine textures serve as the training data for the subsequent inpainting network.

3.3.2 Texture Inpainting

Given the training data simulated by LDMs, we then train our inpainting network. Note that we train a single network for all clothing categories, making it general-purpose.

Regarding the impainting work, we choose Pix2PixHD [33], which shows better results than alternative approaches such as conditional TransUNet [6], ControlNet. One issue of Pix2PixHD is that produces color-consistent output $\mathcal{T}_o$ in contrast to prompt-guided ControlNet (please check our supp. material for visualization comparison). These results are compared with the

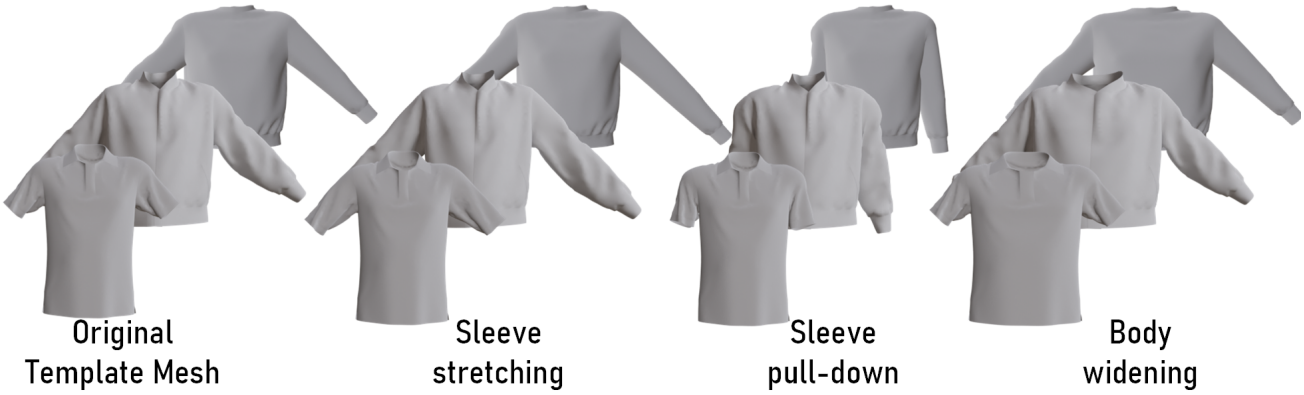

Figure 4. Illustration of the three sleeve-related blendshapes of our template mesh model. These blendshapes allow rendering clothing images in diverse pose configurations to facilitate simulating real-world clothing image layouts.

input full UV as the condition. To address this issue, we first locate the missing holes, continuous edges and lines in the coarse UV as the residual mask M_r (left corner at the bottom line of Fig. 9). We then linearly blend those blank areas with the model's output during texture repairing. Formally speaking, we compute the output as below:

$$\mathcal{T}_{\text{fine}} = \text{BilateralFilter}(\mathcal{T}_{\text{coarse}} + M_r * \mathcal{T}_o) \qquad (6)$$

where BilateralFilter is non-linear filter that can blur the irregular and rough seaming between $\mathcal{T}_{\text{coarse}}$ and $\mathcal{T}_o$ very well while keeping edges fairly sharp. More details can be seen in our attached video.

4. Experiments

Our goal is to generate 3D garments from 2D catalog images. We verify the effectiveness of Cloth2Tex via thorough evaluation and comparison with state-of-the-art baselines. Furthermore, we conduct a detailed ablation study to demonstrate the effects of individual components.

4.1. Comparison with SOTA

We first compare our method with SOTA virtual try-on algorithms, both 3D and 2D approaches.

Comparison with 3D SOTA: We compare Cloth2Tex with SOTA methods that produce 3D mesh textures from 2D clothing images, including model-based Pix2Surf [20] and TPS-based Warping [19] (We replace the original MADF with locally changed UV-constrained Naiver Stokes method, differences between our UV-constrained naiver-stokes and original version is described in the suppl. material). As shown in Fig. 5, our method produces high-fidelity 3D textures with sharp, high-frequency details of the patterns on clothing, such as the leaves and characters on the top row. In addition, our method accurately preserves the spatial configuration of the garment, particularly the overall aspect ratio of the patterns and the relative locations of the

Figure 5. Comparison with Pix2Surf [20] and Warping [19] on T-shirts. Please zoom in for more details.

logos. In contrast, the baseline method Pix2Surf [20] tends to produce blurry textures due to a smooth mapping network, and the Warping [19] baseline introduces undesired spatial distortions (e.g., second row in Fig. 5) due to sparse correspondences.

Comparison with 2D SOTA: We further compare Cloth2Tex with 2D virtual try-on methods: Flow-based DAFlow [2] and StyleGAN-enhanced Deep-Generative-Projection (DGP) [8]. As shown in Fig. 6, Cloth2Tex achieves better quality than 2D virtual try-on methods in sharpness and semantic consistency. More importantly, our outputs, namely 3D textured clothing meshes, are naturally compatible with cloth physics simulation, allowing the synthesis of realistic try-on effects in various body poses. In contrast, 2D methods rely on prior learned from training

images and are hence limited in their generalization ability to extreme poses outside the training distribution.

Figure 6. Comparison with 2D Virtual Try-One methods, including DAFlow [2] and DGP [8].

User Study: Finally, we conduct a user study to evaluate the overall perceptual quality and consistency with our methods' provided input catalog images and 2D and 3D baselines. We consider DGP the 2D baseline and TPS the 3D baseline due to their best performance among existing work. Each participant is shown three randomly selected pairs of results, one produced by our method and the other made by one of the baseline methods. The participant is requested to choose the one that appears more realistic and matches the reference clothing image better. In total, we received 643 responses from 72 users aged between 15 and 60. The results are reported in Fig. 7. Compared to DGP [8] and TPS, Cloth2Tex is favored by the participants with preference rates of 74.60% and 81.65%, respectively. This user study result verified the quality and consistency of our method.

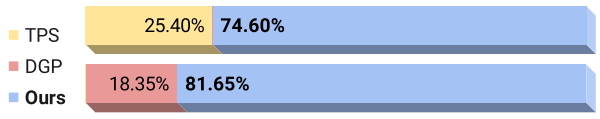

Figure 7. User preferences among 643 responses from 72 participants. Our method is favored by significantly more users.

4.2. Ablation Study

To demonstrate the effect of individual components in our pipeline, we perform an ablation study for both stages in our pipeline.

Neural Rendering vs. TPS Warping: TPS warping has been widely used in previous work on generating 3D garment textures. However, we found that it suffers from challenging cases illustrated in Fig. 2, so we propose a new pipeline based on neural rendering. We compare our method with TPS warping quantitatively to verify this design choice. Our test set consists of 10+ clothing categories, including T-shirts, Polos, sweatshirts, jackets, hoodies, shorts, trousers, and skirts, with 500 samples per category. We report the structure similarity (SSIM [36]) and peak signal-to-noise ratio (PSNR) between the recovered textures and the ground truth textures.

As shown in Tab. 1, our neural rendering-based pipeline achieves superior SSIM and PSNR compared to TPS warping. This improvement is also preserved after inpainting and refinement, leading to a much better quality of the final texture. We conduct a comprehensive comparison study on various inpainting methods in the supp. material, and please check it if needed.

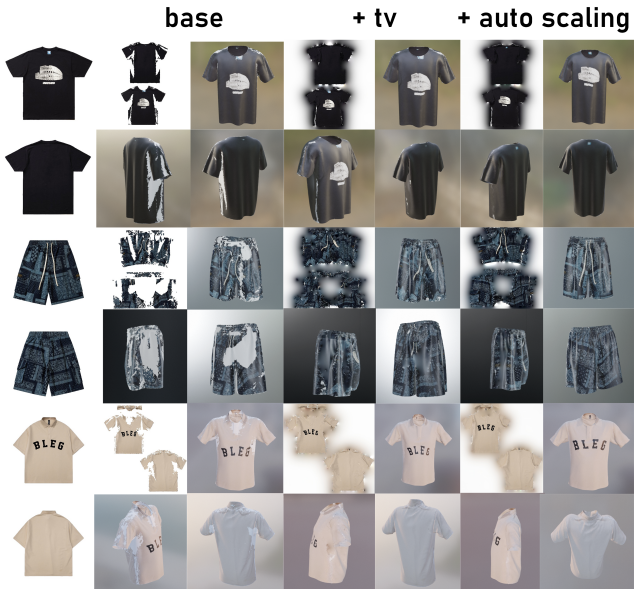

Figure 8. Ablation Study on Phase I. From left to right: base, base + total variation loss E_{tv}, base + E_{tv} + automatic scaling.

Table 1. Neural Rendering vs. TPS Warping. We evaluate the texture quality of neural rendering and TPS-based warping, with and without inpainting.

Baseline	Inpainting	SSIM ↑	PSNR ↑
TPS	*None*	0.70	20.29
TPS	*Pix2PixHD*	0.76	23.81
Phase I	*None*	0.80	21.72
Phase I	*Pix2PixHD*	**0.83**	**24.56**

Total Variation Loss & Automatic Scaling (Phase I) As shown in Fig. 8, dropping the total variation loss E_{tv} and automatic scaling, the textures are incomplete and cannot maintain a semantically correct layout. With E_{tv}, Cloth2Tex produces more complete textures by exploiting

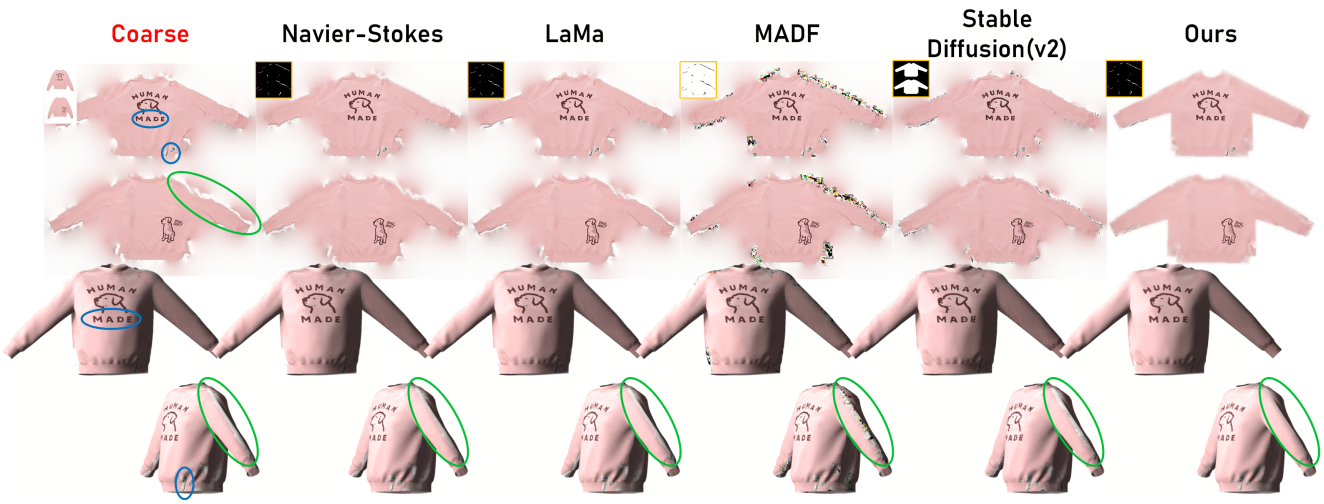

Figure 9. Comparison with SOTA inpainting methods (Naiver-Stokes [4], LaMa [30], MADF [40] and Stable Diffusion v2 [24]) on texture inpainting. The upper left corners of each column are the conditional mask input. **Blue** in the first column shows that our method is capable of maintaining consistent boundary and curvature *w.r.t* reference image while **Green** highlights the blank regions that need inpainting.

the local consistency of textures. Further applying automatic scaling results in better alignment between the template mesh and the input images, resulting in a more semantically correct texture map.

Inpainting Methods (Phase II) Next, to demonstrate the need for training an inpainting model specifically for UV clothing textures, we compare our task-specific inpainting model with general-purpose inpainting algorithms, including Navier-Stokes [4] algorithm and off-the-shelf deep learning models including LaMa [30], MADF [40] and Stable Diffusion v2 [24] with pre-trained checkpoints. Here, we modify the traditional Navier-Stokes [4] algorithm to a UV-constrained version because a texture map is only part of the whole squared image grid, where plenty of non-UV regions produce an adverse effect for texture in-painting (please see supp. material for comparison).

As shown in Fig. 9, our method, trained on our synthetic dataset generated by the diffusion model, outperforms general-purpose inpainting methods in the task of refining and completing clothing textures, especially in terms of the color consistency between inpainted regions and the original image.

4.3. Limitations

As shown in Fig. 10, Cloth2Tex can produce high-quality textures for common garments, *e.g*. T-shirt, Shorts, Trousers and *etc*. (blue bounding box (bbox)). However, we have observed that it is having difficulty in recovering textures for garments with complex patterns: *e.g*. inaccurate and inconsistent local texture (belt, collarband) occurred in windbreaker (red bbox). We regard this as the extra accessories occurred in the garment, which inevitably add on the partial texture in addition to the main UV.

Another imperfection is that our method cannot maintain the uniformity of checked shirts with densely assembled grids: As shown in the second row of Fig. 6, our method inferior to 2D VTON methods in preserving texture among which comprised of thousands of fine and tiny checkerboard-like grids, checked shirts and pleated skirts are representative type of such garments.

We boil this down to the subtle position changes during deformation graph optimization period, which leads to the template mesh becomes less uniform eventually as the regularization terms, *i.e*. as-rigid-as-possible is not a very strong constraint energy terms in obtaining a conformal mesh. We acknowledge this challenge and leave it to future work to explore the possibility in generating a homogeneous mesh with uniformly-spaced triangles.

5. Conclusion

This paper presents a novel pipeline, Cloth2Tex, for synthesizing high-quality textures for 3D meshes from the pictures taken from only front and back views. Cloth2Tex adopts a two-stage process in obtaining visually appealing textures, where phase I offers coarse texture generation and phase II performs texture refinement. Training a generalized texture inpainting network is non-trivial due to the high topological variability of UV space. Therefore, obtaining paired data under such circumstances is important. To the best of our knowledge, this is the first study to combine a diffusion model with a 3D engine (Blender) in collecting coarse-fine paired textures in 3D texturing tasks. We show the generalizability of this approach in a variety of examples.

To avoid distortion and stretched artifacts across clothes,

Figure 10. Visualization of 3D virtual try-on. We obtain textured 3D meshes from 2D reference images shown on the left. The 3D meshes are then draped onto 3D humans.

we automatically adjust the scale of vertices of template meshes and thus best prepare them for later image-based optimization, which effectively guides the implicitly learned texture with a complete and distortion-free structure. Extensive experiments demonstrate that our method can effectively synthesize consistent and highly detailed textures for typical clothes without extra manual effort.

In summary, we hope our work can inspire more future research in 3D texture synthesis and shed some light on this area.

References

[1] AUTOMATIC1111. Stable diffusion web ui. https://github.com/AUTOMATIC1111/stable-diffusion-webui, 2022. 5

[2] Shuai Bai, Huiling Zhou, Zhikang Li, Chang Zhou, and Hongxia Yang. Single stage virtual try-on via deformable attention flows. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XV*, pages 409–425. Springer, 2022. 6, 7

[3] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(4):509–522, 2002. 2, 3

[4] Marcelo Bertalmio, Andrea L Bertozzi, and Guillermo Sapiro. Navier-stokes, fluid dynamics, and image and video inpainting. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, pages I–I. IEEE, 2001. 8, 1

[5] Dave Zhenyu Chen, Yawar Siddiqui, Hsin-Ying Lee, Sergey Tulyakov, and Matthias Nießner. Text2tex: Text-driven texture synthesis via diffusion models. *arXiv preprint arXiv:2303.11396*, 2023. 3, 1

[6] Jieneng Chen, Yongyi Lu, Qihang Yu, Xiangde Luo, Ehsan Adeli, Yan Wang, Le Lu, Alan L. Yuille, and Yuyin Zhou. Transunet: Transformers make strong encoders for medical image segmentation. *arXiv preprint arXiv:2102.04306*, 2021. 5, 1, 4

[7] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021. 3, 5

[8] Ruili Feng, Cheng Ma, Chengji Shen, Xin Gao, Zhenjiang Liu, Xiaobo Li, Kairi Ou, Deli Zhao, and Zheng-Jun Zha. Weakly supervised high-fidelity clothing model generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3440–3449, 2022. 6, 7

[9] Clement Fuji Tsang, Maria Shugrina, Jean Francois Lafleche, Towaki Takikawa, Jiehan Wang, Charles Loop, Wenzheng Chen, Krishna Murthy Jatavallabhula, Edward Smith, Artem Rozantsev, Or Perel, Tianchang Shen, Jun Gao, Sanja Fidler, Gavriel State, Jason Gorski, Tommy Xiang, Jianing Li, Michael Li, and Rev Lebaredian. Kaolin: A pytorch library for accelerating 3d deep learning research. https://github.com/NVIDIAGameWorks/kaolin, 2022. 4

[10] Rinon Gal, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit H Bermano, Gal Chechik, and Daniel Cohen-Or. An image is worth one word: Personalizing text-to-image generation using textual inversion. *arXiv preprint arXiv:2208.01618*, 2022. 5

[11] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020. 3

[12] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021. 5

[13] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 1

[14] Mikhail Konstantinov, Alex Shonenkov, Daria Bakshandaeva, and Ksenia Ivanova. Deepfloyd: Text-to-image model with a high degree of photorealism and language understanding. https://deepfloyd.ai/, 2023. 1

[15] Chenliang Li, Haiyang Xu, Junfeng Tian, Wei Wang, Ming Yan, Bin Bi, Jiabo Ye, Hehong Chen, Guohai Xu, Zheng Cao, et al. mplug: Effective and efficient vision-language learning by cross-modal skip-connections. *arXiv preprint arXiv:2205.12005*, 2022. 5, 1

[16] Dongxu Li, Junnan Li, Hung Le, Guangsen Wang, Silvio Savarese, and Steven C. H. Hoi. Lavis: A library for language-vision intelligence, 2022. 5, 1

[17] Shichen Liu, Tianye Li, Weikai Chen, and Hao Li. Soft rasterizer: A differentiable renderer for image-based 3d reasoning. *The IEEE International Conference on Computer Vision (ICCV)*, 2019. 2, 3, 4, 5

[18] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. Smpl: A skinned multi-person linear model. *ACM transactions on graphics (TOG)*, 34(6):1–16, 2015. 2

[19] Sahib Majithia, Sandeep N Parameswaran, Sadbhavana Babar, Vikram Garg, Astitva Srivastava, and Avinash Sharma. Robust 3d garment digitization from monocular 2d images for 3d virtual try-on systems. In *Proceedings of the IEEE/CVF Winter Conference on*

Applications of Computer Vision, pages 3428–3438, 2022. 1, 2, 3, 6

[20] Aymen Mir, Thiemo Alldieck, and Gerard Pons-Moll. Learning to transfer texture from clothing images to 3d humans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7023–7034, 2020. 1, 2, 3, 6

[21] 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. 2017. 1

[22] Yongming Rao, Wenliang Zhao, Guangyi Chen, Yansong Tang, Zheng Zhu, Guan Huang, Jie Zhou, and Jiwen Lu. Denseclip: Language-guided dense prediction with context-aware prompting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 4

[23] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3d deep learning with pytorch3d. *arXiv:2007.08501*, 2020. 5, 1

[24] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022. 2, 3, 5, 8, 1

[25] Leonid I Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: nonlinear phenomena*, 60(1-4): 259–268, 1992. 5

[26] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *arXiv preprint arXiv:2210.08402*, 2022. 3, 5

[27] Yawar Siddiqui, Justus Thies, Fangchang Ma, Qi Shan, Matthias Nießner, and Angela Dai. Texturify: Generating textures on 3d shape surfaces. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part III*, pages 72–88. Springer, 2022. 2

[28] Olga Sorkine and Marc Alexa. As-rigid-as-possible surface modeling. In *Symposium on Geometry processing*, pages 109–116, 2007. 4

[29] Robert W Sumner, Johannes Schmid, and Mark Pauly. Embedded deformation for shape manipulation. In *ACM siggraph 2007 papers*, pages 80–es. 2007. 3

[30] Roman Suvorov, Elizaveta Logacheva, Anton Mashikhin, Anastasia Remizova, Arsenii Ashukha, Aleksei Silvestrov, Naejin Kong, Harshith Goka, Kiwoong Park, and Victor Lempitsky. Resolution-robust large mask inpainting with fourier convolutions. *arXiv preprint arXiv:2109.07161*, 2021. 8, 1

[31] Brandon Trabucco, Kyle Doherty, Max Gurinas, and Ruslan Salakhutdinov. Effective data augmentation with diffusion models. *arXiv preprint arXiv:2302.07944*, 2023. 3

[32] Peng Wang, An Yang, Rui Men, Junyang Lin, Shuai Bai, Zhikang Li, Jianxin Ma, Chang Zhou, Jingren Zhou, and Hongxia Yang. Ofa: Unifying architectures, tasks, and modalities through a simple sequence-to-sequence learning framework. In *International Conference on Machine Learning*, pages 23318–23340. PMLR, 2022. 5, 1

[33] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8798–8807, 2018. 5, 1, 4

[34] Tuanfeng Y. Wang, Duygu Ceylan, Jovan Popovic, and Niloy J. Mitra. Learning a shared shape space for multimodal garment design. *ACM Trans. Graph.*, 37(6):1:1–1:14, 2018. 3

[35] Wenguan Wang, Yuanlu Xu, Jianbing Shen, and Song-Chun Zhu. Attentive fashion grammar network for fashion landmark detection and clothing category classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4271–4280, 2018. 2, 4

[36] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004. 7

[37] Yi Xu, Shanglin Yang, Wei Sun, Li Tan, Kefeng Li, and Hui Zhou. 3d virtual garment modeling from rgb images. In *2019 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 37–45. IEEE, 2019. 1

[38] Rui Yu, Yue Dong, Pieter Peers, and Xin Tong. Learning texture generators for 3d shape collections from internet photo sets. In *British Machine Vision Conference*, 2021. 2

[39] Lvmin Zhang and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models, 2023. 2, 5, 1, 4

[40] Manyu Zhu, Dongliang He, Xin Li, Chao Li, Fu Li, Xiao Liu, Errui Ding, and Zhaoxiang Zhang. Image inpainting by end-to-end cascaded refinement with mask awareness. *IEEE Transactions on Image Processing*, 30:4855–4866, 2021. 3, 8

Cloth2Tex: A Customized Cloth Texture Generation Pipeline for 3D Virtual Try-On

Supplementary Material

6. Implementation Details

In phase I, we fix the optimization steps of both silhouette matching and image-based optimization to 1,000, which makes each coarse texture generation process takes less than 1 minute to complete on an NVIDIA Ampere A100 (80GB VRAM). The initial weights of each energy term are $w_{sil} = 50, w_{lmk} = 0.01, w_{arap} = 50, w_{norm} = 10, w_{img} = 100, w_{tv} = 1$, we then use cosine scheduler for decaying w_{arap}, w_{norm} to $5, 1$.

During the blender-enhanced rendering process, we augment the data by random sampling blendshapes of upper cloth by a range of $[0.1, 1.0]$. The synthetic images were rendered using Blender **EEVEE** engine at a resolution of 512^2, emission only (disentangle from the impact of shading, which is the notoriously difficult puzzle as dissected in Text2Tex [5]).

The synthetic data used for training texture inpainting network are yielded from pretrained ControlNet through prompts (generates from Lavis-BLIP [16], OFA [32] and MPlug [15]) and UV templates (manually crafted UV maps by artists) can be shown in Fig. 14, which contains more garment types than previous methods, *e.g.* Pix2Surf [20] (**4**) and Warping [19] (**2**).

The only existing trainable Pix2PixHD in phase II is optimized by Adam [13] with $lr = 2e - 4$ for 200 epochs. Our implementation is build on top of PyTorch [21] alongside PyTorch3D [23] for silhouette matching, rendering and inpainting.

Table 2. SOTA inpainting methods act on our synthetic data.

Baseline	Inpainting	SSIM ↑
Phase I	*None*	0.80
Phase I	*Navier-Stokes* [4]	0.80
Phase I	*LaMa* [30]	0.78
Phase I	*Stable Diffusion (v2)* [24]	0.77
Phase I	*Deep Floyd* [14]	0.80

Table 3. Inpainting methods trained on our synthetic data.

Baseline	Inpainting	SSIM ↑
Phase I	*None*	0.80
Phase I	*Cond-TransUNet* [6]	0.78
Phase I	*ControlNet* [39]	0.77
Phase I	*Pix2PixHD* [33]	**0.83**

Figure 11. Visualization of Navier-stokes method on UV template. Our locally constrained NS method fills the blanks thoroughly (though lack of precision) compared to the original global counterpart.

The detailed parameters of template meshes in Cloth2Tex are summarized in Tab. 4, sketch of all template meshes and UV maps are shown in Fig. 12 and Fig. 13 respectively.

7. Self-modified UV-constrained Naiver-Stokes Method

As shown in Fig. 11, we display the results between our self-modified UV-constrained Navier-Stokes (NS) method (*local*) and original NS (*global*) method. Specifically, we add a reference branch (UV template) for NS and thus confine the inpainting-affected region to the given UV template for each garment, thus contributing directly to the interpolation result. Our locally constrained NS method allows blanks to be filled thoroughly compared to the original global NS method.

The sole aim of modifying the original global NS method is to conduct a fair comparison with deep learning based methods as depicted in the main paper.

The noteworthy thing is that for small blank areas (*e.g.* Column *1,3* of Fig. 11), the texture uniformity and consistency are well-persevered thus capable of producing plausi-

Figure 12. Visualization of all template meshes used in Cloth2Tex.

Figure 13. All UV maps of template meshes used in Cloth2Tex.

Table 4. Detailed parameters of template mesh in Cloth2Tex. As shown in the table, each template's vertex is less than 10,000 and all are animatable by means of Style3D, which is the best fit software for clothing animation.

Category	Vertices	Faces	Key Nodes (Deformation Graph)	Animatable
T-shirts	8,523	16,039	427	✓
Polo	8,922	16,968	447	✓
Shorts	8,767	14,845	435	✓
Trousers	9,323	16,995	466	✓
Dress	7,752	14,959	388	✓
Skirt	6,116	11,764	306	✓
Windbreaker	9,881	17,341	494	✓
Jacket	8,168	15,184	409	✓
Hoodie (Zipup)	8,537	15,874	427	✓
Sweatshirt	9,648	18,209	483	✓
One-piece Dress	9,102	17,111	455	✓

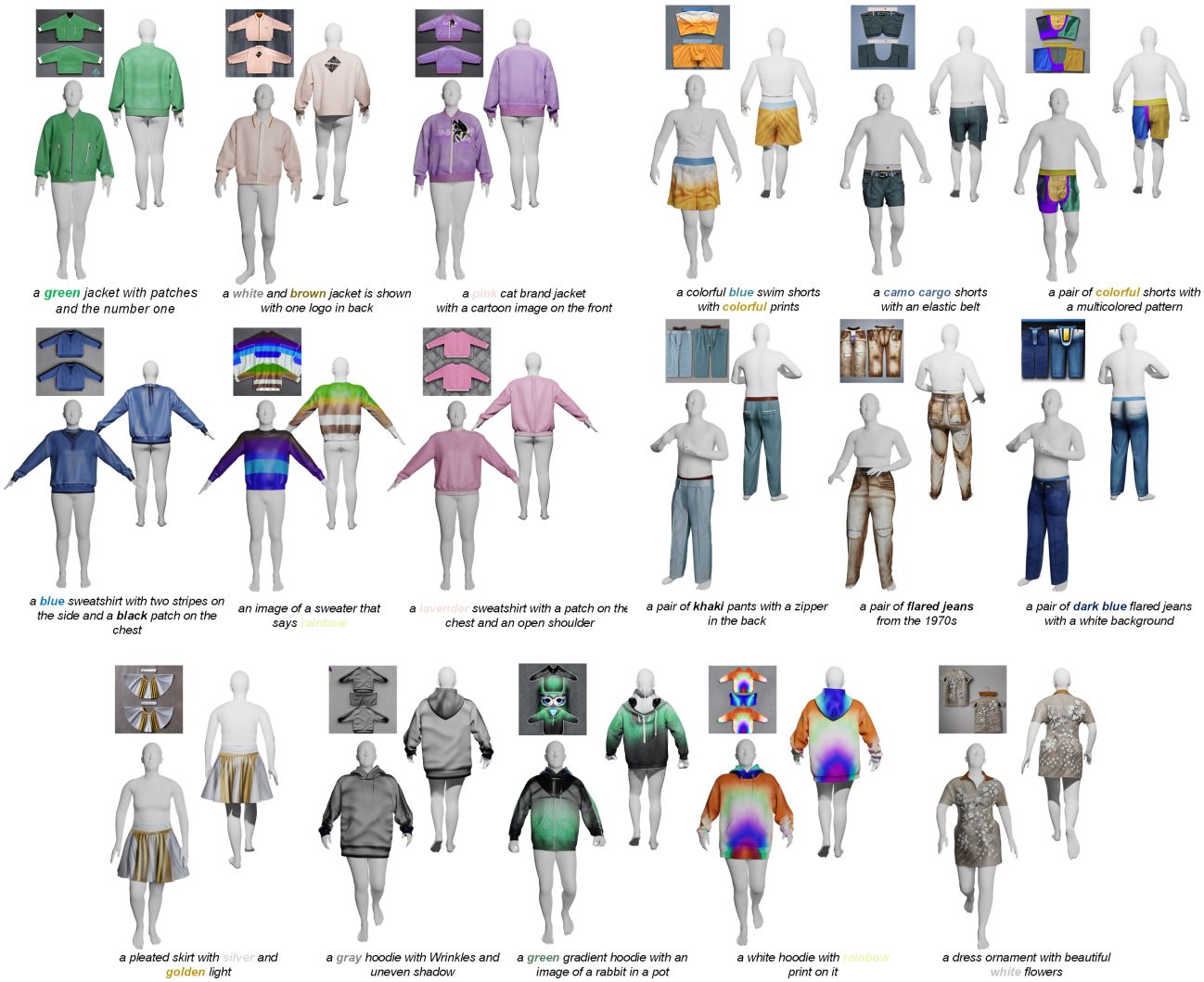

Figure 14. Texture maps for training instance map guided Pix2PixHD, synthesized by ControlNet *canny edge*.

Figure 15. Comparison with representative image2image methods with conditional input: autoencoder-based TransUNet [6] (we modify the base model and add an extra branch for UV map, aims to train it with all types of garment together), diffusion-based ControlNet [39] and GAN-based Pix2PixHD [33]. It is rather obvious that prompts-sensitive ControlNet limited in recover a globally color-consistent texture maps. Upper right corner of each method is the conditional input.

ble textures.

8. Efficiency of mainstream Inpainting methods

As depicted in the main paper, our neural rendering based pipeline achieves superior SSIM compared to TPS warping. This improvement is also preserved after inpainting and refinement, leading to a much better quality of the final texture.

Free from the page limit in the main paper, here we conduct a comprehensive comparison study on various inpainting methods act upon the coarse texture maps derived from Phase I directly, to demonstrate the efficiency of mainstream inpainting methods.

First, we compare the state-of-the-art inpainting methods quantitatively on our synthetic coarse-fine paired dataset. One thing to note is that checkpoints derived from all deep learning based inpainting methods are open and free. No finetune or modification is involved in this comparison. As described in Tab. 2, none of such methods produce a noticeable positive impact in boosting the SSIM score compared to the original coarse texture (*None* version).

Next, we revise TransUNet [6] with input a conditional UV map for the unity of the input and output with ControlNet [39] and Pix2PixHD [33]. Then we train *cond-TransUNet*, ControlNet, and Pix2PixHD on the synthetic data for a fair comparison. We input all these three with original input coarse texture maps, conditional input UV maps, and output fine texture maps. The selective basis of TransUNet, ControlNet, and Pix2PixHD originates from the generative paradigm: TransUNet is a basic autoencoder-based supervised learning image2image model, ControlNet is a diffusion-based generative model and Pix2PixHD is a GAN-based generative model. We want to explore the feasibility of these methods in our task, as depicted in Tab. 3 and Fig. 15, Pix2PixHD is superior in obtaining satisfactory texture maps in terms of both qualitative and quantitative views.