Generating Texture for 3D Human Avatar from a Single Image using Sampling and Refinement Networks

Sihun Cha¹, Kwanggyoon Seo¹, Amirsaman Ashtari¹, Junyong Noh¹

¹Visual Media Lab, KAIST, Republic of Korea

Abstract

There has been significant progress in generating an animatable 3D human avatar from a single image. However, recovering texture for the 3D human avatar from a single image has been relatively less addressed. Because the generated 3D human avatar reveals the occluded texture of the given image as it moves, it is critical to synthesize the occluded texture pattern that is unseen from the source image. To generate a plausible texture map for 3D human avatars, the occluded texture pattern needs to be synthesized with respect to the visible texture from the given image. Moreover, the generated texture should align with the surface of the target 3D mesh. In this paper, we propose a texture synthesis method for a 3D human avatar that incorporates geometry information. The proposed method consists of two convolutional networks for the sampling and refining process. The sampler network fills in the occluded regions of the source image and aligns the texture with the surface of the target 3D mesh using the geometry information. The sampled texture is further refined and adjusted by the refiner network. To maintain the clear details in the given image, both sampled and refined texture is blended to produce the final texture map. To effectively guide the sampler network to achieve its goal, we designed a curriculum learning scheme that starts from a simple sampling task and gradually progresses to the task where the alignment needs to be considered. We conducted experiments to show that our method outperforms previous methods qualitatively and quantitatively.

CCS Concepts

• Computing methodologies → Computer graphics; Computer vision; Image manipulation;

© 2023 Eurographics - The European Association for Computer Graphics and John Wiley & Sons Ltd.
1. Introduction

The demand for animatable 3D human avatars is increasing in various VR/AR applications such as virtual try-on, metaverse, and games. To create an animatable 3D human avatar, it is essential to produce a 3D model that resembles the shape and appearance of the source human appearance. Furthermore, the 3D model should be rigged for animation. These processes often require manual work from artists or rely on a special capture system such as multi-view camera sets or 3D scanners. To alleviate these conditions, numerous methods have been proposed to reconstruct a 3D human avatar [KPD19; APTM19; LCL*19; LIP19; XCM*21; NSH*19; SHN*19; SSSJ20; HCJS20; HXL*20; HXS*21; AZS22] from a single image. In contrast to the reported successes in reconstructing body shapes and poses, restoring the occluded texture for 3D human avatars has been relatively less studied.

Generating a texture map for a 3D human avatar from a single image is challenging due to the following two reasons. First, only portions of the texture information are available from the source image. This is caused by various poses and shapes of the human body and the diverse camera positions. Second, the generated human texture map needs to be semantically aligned with the surface of the target 3D human mesh. As the texture coordinates correspond to the surface geometry of a 3D human mesh, misalignment of the texture can produce a distorted human appearance in a rendered image.

Due to these challenges, generating a 3D human texture map cannot be simply posed as an image inpainting task [LRS*18]. Unlike image painting, where inputs and outputs are spatially aligned, the alignment of inputs and outputs is not guaranteed for 3D human texture generation tasks. Because UV alignment is essential, a process of correcting the spatial structure of the input during the 3D human texture generation task is required. The difference between the image inpainting and 3D human texture generation is highlighted in Figure 2. Therefore, direct application of image inpainting methods to 3D human texture generation tends to result in a misaligned texture map. On the other hand, methods that utilize image-to-image translation [IZZE17] may successfully produce a texture map that is semantically aligned with the UV space of the target mesh [LIP19; WZL*19]. Unfortunately, these CNN-based models tend to learn an average texture from the training data, leading to a blurry result.

In this paper, we propose a method that generates a complete human texture map from a single image while synthesizing the occluded texture with relevance to the given visible appearance. Using a neural network based on sampling and refinement strategies, our method preserves the details given in the source image in the generated texture while retaining the structural alignment with the surface of the target mesh. Similar to previous methods [APTM19; LIP19], we convert the source image to a partial texture map and use it as input to our method. We also predict a 3D human mesh based on the SMPL model [LMR*15] from the source image using off-the-shelf method [APTM19] and utilize the surface normal information of the 3D human mesh in the sampling process. Given the partial texture map and a normal map, SamplerNet completes the missing details by sampling the visible region of the texture and re-arranging them.

The proposed sampler network overcomes the limitation of image inpainting methods [GSVL19; YLY*19], such as the generation of misaligned textures, by learning to align the texture to the corresponding surface of the target 3D human mesh. To guide the SamplerNet for effective sampling, we adopt a curriculum learning scheme. Given the sampled texture map produced by SamplerNet, RefinerNet generates a refined texture and a blending mask. The mask is used for blending the refined texture with the sampled texture to produce the final result. The blending mask helps to preserve the appearance detail presented in the source image and therefore allows to generate the final texture map with improved quality by removing artifacts from the sampled texture. Example results of the generated texture map and rendered images are shown in Figure 1. We conducted a set of experiments to show that our approach outperforms the various baseline methods [IZZE17; WLZ*18; LIP19; XL21; ALY*21] in reproducing the details present in the source image and aligning textures to the surface of the target 3D human mesh.

2. Related Work

2.1. 3D Human Texture Generation with Multi-View Images

There have been many studies that allow to generate 3D human avatars with texture from multi-view images [AMX*18a; AMX*18b; BTTP19; ZLT*20; MAP20; BTTP19]. These methods project the given multi-view images or video frames back to the predicted mesh to create the partial texture maps and combine them with blending techniques to produce the final texture map. However, with only a single view, it is not clear how to extend the methods to capture all the detailed information and fill in the invisible regions.
2.2. 3D Human Texture Generation from a Single Image

Progress in generating 3D human avatars from a single image has shown a big leap in reconstructing body shapes and poses as much as that interest. However, synthesizing a texture that maintains given appearance details and restoring the occluded region with the relevance of the visible region is still challenging. Similar to 3D human texture generation with multi-view images, some methods [NSH*19; SHN*19; HXL*20; HXS*21] acquire the texture by predicting the back view from the given frontal view image followed by projecting the both front and back view images back to the predicted 3D human mesh. While these methods are effective in utilizing the visible textures in the source image, the results are often blurry and poorly reconstructed in the occluded area.

Some studies [GSVL19; LIP19] employ a predefined mapping process that converts the visible human appearance in the source image to the UV space of the SMPL model using DensePose [GNK18]. Lazova et al. [LIP19] utilized the partial texture map produced by a predefined mapping and takes an image-to-image translation approach to generate a full texture map. Because this method learns to generate the missing information based on training data, the generated results can often be blurry and detailed texture patterns are not recovered well when the given garment texture is unseen from the training data.

Instead of using a texture explicitly for supervision, some methods learn the human texture generation in an unsupervised manner, directly producing a texture map from the given image [WZL*19; XL21; COO22]. These methods can generate a texture map directly from the source image by eliminating the pose variant features and minimizing the identity difference. Xu et al. [XL21] maps the given image to the texture space by predicting flow field and blends it with the generated texture map to eliminate the artifacts and maintain detailed appearance. Inspired from this approach, our method also incorporates blending process with sampled texture and refined texture.

### 2.3. Pose-guided Image and Video Synthesis

The goal of pose-guided image synthesis is to transfer the person’s appearance from a source image to the desired pose. Some approaches [SMX*20; SLGT21; SGLT21] utilize the UV space of the SMPL model as an intermediate representation to achieve the task. These methods generate the latent features, which lie in the SMPL’s UV space, and use them to synthesize the image with the given target pose. Other methods [NGK18; LXH*20; YLG*21] directly predict the texture in the UV space of the target human mesh while reproducing the detailed appearance presented in the source image.

In this section, we describe the proposed networks and training process. We first preprocess the human appearance in the source image to be mapped into the UV space of a 3D human mesh [LMR*15] and produce a partial texture map and a corresponding visibility mask from the partial texture map, we obtain the geometry information using Tex2shape [APTM19], which generates the 3D human mesh from a single image by predicting the surface normal and vertex displacements in the UV space of the SMPL model. Given the partial texture map, visibility mask, and normal map, SamplerNet samples the missing appearance details from the visible regions of the source image and completes the texture map. In the following process, RefinerNet generates a blending mask and a refined texture. Insufficient details and distortion artifacts in the sampled texture produced by SamplerNet are adjusted in the refined texture. The final texture is then generated by blending the sampled texture with the refined texture using the blending mask. An overview of the proposed method is shown in Figure 3.

### 3.1. Preprocessing

Similar to previous methods [LIP19; APTM19; GSVL19; ALY*21], we map the source image into the UV space of the SMPL model. We denote the mapped partial texture as $T_{source}$. Following Albahar et al. [ALY*21], we combine $T_{source}$ and $T_{mirror}$, a semantically mirrored texture map of $T_{source}$, to produce a symmetric texture map $T_{input}$.

\[
T_{input} = T_{source} + (T_{mirror} \odot (1 - M_{source})),
\]

where $\odot$ is the Hadamard product. $M_{source}$ is a binary mask in which 1 indicates a valid pixel in $T_{source}$ and 0 indicates an invalid pixel.
Figure 3: Overview of the proposed method. The source image is processed to create partial texture, visibility mask, and normal map which are given as an input to SamplerNet. SamplerNet predicts a sampling grid that is used for producing the sampled texture. RefinerNet receives the sampled texture and occlusion mask as input, and generates a refined texture and blending mask. The final output is produced by alpha blending the sampled texture with the refined texture using the blending mask.

For the mapping, we use the indexed UV coordinates (IUV) predicted by DensePose [GNK18] to map the source image to the UV space of the SMPL model. The IUV establishes the correspondence between the human appearance in the image and the surface of the SMPL model. The surface is labeled with an index which indicates a predefined body part. Using the IUV and a lookup table provided by DensePose, the visible appearance in the source image can map to the UV space. Because DensePose divides the surface of the SMPL model to exploit the left-right symmetry, the mirrored texture map $T_{\text{source}}^\text{mirror}$ can be created by switching the index for the IUV of the symmetrical body parts during the mapping process.

### 3.2. Sampler Network

Given $T_{\text{input}}$, visibility mask, and normal map as input, SamplerNet predicts a sampling grid to produce a complete texture map, $T_{\text{sample}}$. The visibility mask is a binary mask that indicates the valid pixel in $T_{\text{input}}$. Using the predicted sampling grid, the visible region of $T_{\text{input}}$ is sampled to synthesize the occluded region in $T_{\text{input}}$. Furthermore, the texture in the visible region is re-sampled to be structurally aligned with the semantic meaning of the UV space. SamplerNet consists of two encoders and a single decoder which is similar to the network proposed in Yoon et al. [YLGi21]. The geometry and appearance features are extracted from each dedicated encoder and are fed to the decoder. The appearance features extracted from $T_{\text{input}}$ and visibility mask at each layer of the encoder are skip-connected to the corresponding layer in the decoder.

Both encoders used for SamplerNet consist of one convolutional layer followed by five layers of residual blocks. The decoder uses the same number of residual block layers with upsampling and one convolutional layer at the end. All activations are LeakyReLU with instance normalization. The residual blocks consist of two convolutional layer with gated convolutions [LYL19].

When training SamplerNet, we observed that SamplerNet often fails to preserve the given appearance detail in $T_{\text{input}}$. To overcome this, we apply the curriculum learning strategy. Before describing the details of the proposed curriculum learning scheme, we will first address the data preparation process for the training.

#### 3.2.1. Training Data Preparation

SamplerNet is trained with a curriculum learning scheme, which begins from the simplest case where the given input texture is perfectly aligned with the UV space, to the hard case where the input texture is misaligned with the UV space. For the simplest case, if the predicted IUV perfectly aligns with the UV space, $T_{\text{source}}$ can be acquired by masking out the ground truth texture map $T_{\text{GT}}$ with $M_{\text{source}}$:

$$T_{\text{source}} = T_{\text{GT}} \odot M_{\text{source}}.$$  \hfill (2)

We denote $T_{\text{source}}$ obtained by Equation 2 as $T_{\text{source}}^M$.

Due to imperfect prediction, however, DensePose often fails to locate the exact pixel position that corresponds to the geometric position. This prevents the partial texture from being semantically aligned with $T_{\text{GT}}$. We approximate this misalignment using a geometric transformation function $f(\cdot)$, and thus $T_{\text{source}}$ can be expressed as augmentation as follows:

$$T_{\text{source}} \approx T_{\text{Augment}} = f(T_{\text{GT}}^M, \alpha),$$  \hfill (3)

where $\alpha$ is a control parameter for $f(\cdot)$. By changing $\alpha$, interpolation from $T_{\text{GT}}^M$ to $T_{\text{source}}$ is approximated which enables the curriculum learning scheme.

#### 3.2.2. Region-wise Augmentation

For the augmentation $f(\cdot)$, we use the thin-plate-spline (TPS) transformation in a region-wise manner (Figure 4). There are two reasons why naive augmentation techniques are inadequate for texture generation. First, unlike general images, texture maps have a...
unique structure that corresponds to a 3D geometry. Second, although $T_{\text{source}}$ is non-linearly deformed from $T_{\text{GT}}$, the deformation is limited to the UV structure of the SMPL model. Therefore, augmentation should be performed in consideration of this structure.

We divide the UV space into six different regions which correspond to the head, body, legs, arms, feet, and hands. We crop each region with the bounding box and apply the TPS transform individually. To prevent unintentional cropping, each body part is masked before the process. Then, we merge all of the transformed regions back to form a single texture followed by multiplying it by the UV mask, $M_{uv}$, to produce the final deformed texture for training. $M_{uv}$ is a binary mask, which represents valid UV coordinates in the UV space. For the transformation, we assign control points to each region and shift these points with a random value determined from a uniform distribution $U(0, \alpha)$. $\alpha$ is expressed as follows:

$$\alpha = \begin{cases} 0, & \text{step} = 0, \\ 0.1 + (\text{step} \times \delta), & \text{step} > 0. \end{cases}$$

$\delta$ is a hyperparameter.

### 3.2.3. Curriculum Learning

The goal of curriculum learning is to guide SamplerNet to maintain the given appearance detail in $T_{\text{input}}$ while enforcing the semantic alignment. To this end, we divide the training into two curriculum steps: sampling the occluded texture from the visible region and rearranging the given visible region. First we will describe the data preparation for the curriculum learning.

In the initial step, $\text{step} = 0$, the objective is to complete the given partial texture map without considering the alignment in the visible region. Hence, $\alpha$ in Equation 3 is set to 0, which makes $f(\cdot)$ an identity mapping function, resulting in $T_{\text{Augment}}$ identical to $T_{\text{GT}}$. As $T_{\text{GT}}$ is semantically aligned with the UV space, SamplerNet is encouraged to sample for the missing regions only. A single $\text{step}$ is set to 4,000 iterations, which is sufficient for the model to see the whole training data set twice.

After the initial step, $\alpha$ is set to 0.1 and increased by $\delta$ after every single step. Here, the goal is to encourage the model to begin re-sampling the visible region to enforce the structural alignment. For the steps equal to or greater than 3, we additionally use the partial texture produced by DensePose to reduce the domain gap between training data and inference data.

The data used in the curriculum learning can be expressed as follows:

$$T_{\text{source}} = \begin{cases} T_{\text{Augment}}, & \text{step} < 3, \\ T_{\text{Augment}} \cup T_{\text{DensePose}}, & \text{step} \geq 3. \end{cases}$$

where $T_{\text{DensePose}}$ is a partial texture produced by DensePose. The variation of the partial texture is visualized in the supplementary material.

The effect of using this augmentation is verified in Section 4.2. With the curriculum learning and the region-wise augmentation, SamplerNet effectively samples a partial texture map to synthesize all the occluded region while retaining the given visible part and aligning texture to the surface of the target 3D mesh.

### 3.2.4. Loss Functions

To enforce the alignment and proper sampling, we train SamplerNet by minimizing a reconstruction loss and a perceptual loss between $T_{\text{sample}}$ and $T_{\text{GT}}$. The reconstruction loss is expressed as follows:

$$\mathcal{L}_{\text{Recon}} = \sum_{i=1}^{N} \left| \| w_i \cdot M^i_{\text{body}} \odot (T_{\text{sample}} - T_{\text{GT}}) \|_1 \right|,$$

where $M^i_{\text{body}}$ is a set of binary masks for the six body parts in the UV space and $w_i$ is its corresponding weights. $w_i$ is set to $0, 0, 1, 1, 1,$ and 1 for each face, body, leg, arm, foot, and hand.

For the perceptual distance, we use LPIPS [ZIE*18], which can be expressed as follows:

$$\mathcal{L}_{\text{LPIPS}} = \text{LPIPS\left(T_{\text{sample}}, T_{\text{GT}}\right)}.$$

where $\text{LPIPS}(\cdot)$ extracts features from the two inputs using AlexNet [Kri14] and calculates the cosine distance between the extracted features.

The total loss function for SamplerNet is expressed as follows:

$$\mathcal{L}_{\text{Sampler}} = \lambda_{\text{Recon}} \mathcal{L}_{\text{Recon}} + \lambda_{\text{LPIPS}} \mathcal{L}_{\text{LPIPS}},$$

where $\lambda_{\text{Recon}}$ and $\lambda_{\text{LPIPS}}$ are set to 1 and 1, respectively.

### 3.3. Refiner Network

After SamplerNet completes the partial texture, the resulting texture map is refined. SamplerNet implicitly learns to sample the missing texture information from the visible region by following the guidance of given geometry information and minimizing the loss in the training process. During the process of sampling the given information according to the geometry information, some artifacts are accompanied as shown in Figure 5.

RefinerNet alleviates this problem by refining the details in $T_{\text{sample}}$. RefinerNet employs a U-Net-like architecture with three down and up sampling layers, and 9 residual blocks for the bottleneck. RefinerNet receives $T_{\text{sample}}$ and occlusion mask as input and produces a refined texture map $T_{\text{refine}}$ and a blending mask $M_{\text{blend}}$. The occlusion mask is a binary mask, which is acquired by subtracting $T_{\text{source}}$ from $M_{uv}$.

As described in Section 2.2, Xu et al. [XL21] proposed a mask-fusion strategy that blends the two texture outputs using a predicted

![Diagram](image-url)
mask to preserve the fine texture details. Inspired by this, we adapt a texture blending process that exploits details in $T_{\text{sample}}$. The blending mask helps to preserve the given appearance detail by replacing the artifacts in the sampled texture with the refined texture. The final output is computed as follows:

$$T_{\text{final}} = T_{\text{sample}} \odot M_{\text{blend}} + T_{\text{refine}} \odot (1 - M_{\text{blend}}).$$

Despite the well reconstructed human appearance features, the fine details observed in $T_{\text{sample}}$ can sometimes be lost and become blurry in $T_{\text{refine}}$. The blending of $T_{\text{refine}}$ and $T_{\text{sample}}$ using $M_{\text{blend}}$ leverages the advantages of each generated texture. The effect of texture blending is demonstrated in Figure 11.

### 3.3.1. Loss Functions

To produce a texture map of perceptually plausible quality with minimal artifacts, we minimize the following objective terms when training RefinerNet:

$$\mathcal{L}_{\text{Refiner}} = \lambda_{\text{Recon}} \mathcal{L}_{\text{Recon}} + \lambda_{\text{VGG}} \mathcal{L}_{\text{VGG}} + \lambda_{\text{GAN}} \mathcal{L}_{\text{GAN}} + \lambda_{\text{FM}} \mathcal{L}_{\text{FM}},$$

where $\mathcal{L}_{\text{Recon}}$, $\mathcal{L}_{\text{VGG}}$, $\mathcal{L}_{\text{GAN}}$, and $\mathcal{L}_{\text{FM}}$ are the reconstruction loss, perceptual loss, adversarial loss, and feature matching loss, respectively.

For $\mathcal{L}_{\text{Recon}}$, instead of calculating the loss between $T_{\text{final}}$ and $T_{\text{GT}}$, we calculate the loss between $T_{\text{final}}$ and $T_{\text{sample}}$. Because there can be multiple texture maps corresponding to one source image, calculating the loss directly from the ground truth map is restrictive. For example, in the source image, a person wearing a T-shirt with a pattern on the front, may have the same pattern, different pattern, or even no pattern on the back. Thus, a pixel-wise loss with $T_{\text{GT}}$ will lead to a texture with the averaged color output (Figure 6). We used the loss between $T_{\text{final}}$ and $T_{\text{sample}}$ to guide the model in a direction that more respects the estimated $T_{\text{sample}}$.

$$\mathcal{L}_{\text{Recon}} = ||T_{\text{sample}} - T_{\text{final}}||_1.$$  \hfill (11)

For $\mathcal{L}_{\text{VGG}}$, we use the pre-trained VGG-19 [SZ14] to calculate the perceptual distance between $T_{\text{final}}$ and $T_{\text{GT}}$ by extracting features from each layer $l$ as performed in Wang et al. [WLZ*18].

$$\mathcal{L}_{\text{VGG}} = \sum_{i=1}^{N} \frac{1}{w_i} ||VGG_i(T_{\text{GT}}) - VGG_i(T_{\text{final}})||_1.$$  \hfill (12)

Here, $VGG_i$ denotes the layer of the VGG-19 network, where $i \in \{1, 6, 11, 20, 29\}$, $w_i$ is set to 32, 16, 8, 4, and 1 for each layer.

We use an adversarial loss with the PatchGAN discriminator [IZZE17]. The objective function can be expressed as follows:

$$\mathcal{L}_{\text{GAN}} = \mathbb{E}_{T_{\text{GT}}} \left[ \log D(T_{\text{GT}}) \right] + \mathbb{E}_{T_{\text{sample}}} \left[ \log (1 - D(T_{\text{final}})) \right],$$

where $D$ denotes the discriminator. We additionally use a feature matching loss $\mathcal{L}_{\text{FM}}$ [WLZ*18], which minimizes the feature distance between $T_{\text{final}}$ and $T_{\text{GT}}$ of the discriminator to stabilize the training. $\mathcal{L}_{\text{FM}}$ is expressed as follows:

$$\mathcal{L}_{\text{FM}} = \sum_{i=1}^{N} ||D_i(T_{\text{GT}}) - D_i(T_{\text{final}})||_1,$$

where $l_i$ represents a set of layers after the activation function and $i \in \{1, 2, 3\}$. The weights used in the total loss function are $\lambda_{\text{Recon}} = 10$, $\lambda_{\text{VGG}} = 10$, $\lambda_{\text{GAN}} = 1$, and $\lambda_{\text{FM}} = 10$.

### 3.4. Training Details

We used the Adam optimizer [KB14] with a learning rate of 0.0002 and beta parameters set to 0.9 and 0.999 for both networks, SamplerNet and RefinerNet. The region-wise augmentation for training SamplerNet is applied with the probability of 0.8 and is set to 0.25. We used color augmentation for training RefinerNet with the probability of 0.5. The batch size is set to 8, and each model is trained separately for 30,000 iterations on an NVIDIA GTX 1080 Ti GPU.

### 4. Experiments

In this section, we compare our method with previous approaches. Furthermore, several experiments were carried out to evaluate the effect of region-wise augmentation, curriculum learning, and texture blending. A total of 1,441 texture maps were used for training the networks: 929 from SURREAL [VRM*17] dataset, 512 from our newly gathered dataset. The textures were rendered with SMPL [LMR*15] in various human poses to generate partial texture maps for training. To obtain various poses, we randomly sampled 100 poses in the collected animations from Mixamo [Ado]. The textures and poses were randomly paired and rendered in the range of $[-90^\circ, 90^\circ]$ with an interval of $10^\circ$. The texture maps were resized to $256 \times 256$ for both training and testing. For the evaluation, we used Digital Wardrobe [BTTTP19] dataset, which contains 96 textures that are different from the training set. Additionally, we
generated texture using samples from SHHQ [FLJ*22b] dataset to assess the generalizability of our method to real human images. The results are shown in Figure 9.

4.1. Comparisons

We compared our method with previous methods in both generated texture map and rendered image to quantitatively measure the quality of the generated texture map. To evaluate the quality of the generated texture map, we used the following metrics: Structural Similarity (SSIM) [WBSS04], Peak Signal-to-Noise Ratio (PSNR), and Learned Perceptual Image Patch Similarity (LPIPS) [ZIE*18]. SSIM and PSNR measure the reconstruction quality and LPIPS measures the perceptual quality using VGG-16 [SZ14] as a backbone. We further evaluated the results after applying the textures to the 3D human mesh. For this, we employed LPIPS, and cosine similarity (CosSim) of features extracted from PCB network [SZY*18]. PCB network [SZY*18] is used for a person re-identification task, which aims to find an identical person from the different cameras. The similarity between the source image and the rendered image using the generated texture is high when the CosSim is close to one. To evaluate the quality of the synthesized tex-
tured in the occluded region, CosSim was measured in two different images: one rendered with a pose identical to that of the person in the given image and the other rendered with A-pose in the frontal view. We denote the cosine similarity to the image rendered in the identical pose as CosSim-I and to the image rendered with A-pose as CosSim-A. LPIPS was calculated with the image rendered in the identical pose. To render the image using a generated texture, we obtained a 3D human mesh from the given image using Tex2Shape [APT19]. We also used RSC-Net [XC21] to predict the 3D human pose and the camera parameters from the given image.

We evaluate our method with previous approaches that can generate texture maps using image-to-image translation: Iso et al. [IZZE17], Wang et al. [WLZ18], and Lazova et al. [LIP19]. Additionally, we also compared our methods with approaches that utilize pixel coordinate information: Xu et al. [XL21] and Albahar et al. [ALY*21]. For Albahar et al. [ALY*21], we exploited their coordinate completion model (CCM) as this can be used for texture generation. We used the available implementations provided by the authors except for Lazova et al. [LIP19], in which we tried to replicate the implementation following the descriptions in the paper. All methods were trained with the same data as our methods except for Xu et al. [XL21]. Because our data was not compatible to train the method, we used the pretrained model released with their code. The input for all baseline methods was the symmetric partial texture map which was produced using Equation 1 except for Xu et al. [XL21] which can generate texture map directly from the given image. To evaluate the robustness of each method against various view angles, we generated texture maps using the images viewed from different perspectives by horizontally rotating the 3D human mesh in the range of $[-90^\circ, 90^\circ]$.

The quantitative results for texture map and rendered image are shown in Table 1 and Table 2, respectively. The methods that generate a texture map using image-to-image translation [IZZE17; WLZ18; LIP19] achieved high scores in PSNR. Nonetheless, the produced texture lacked the detailed patterns present in the source image and failed to align with the surface of the target 3D mesh as shown in Figure 7(b)-(d) and Figure 8(b)-(d).

The methods that utilize pixel coordinates [XL21; ALY*21] tend to preserve visible texture pattern in the given image better compared to image-to-image translation methods [IZZE17; WLZ18; LIP19] as shown in Figure 8(e)-(f). Xu et al. [XL21] directly generate a texture map from the given image by predicting the flow field from the source image to the UV space of the SMPL model. This allows the preservation of appearance detail. However, some artifacts are apparent when the side view image is given as shown in Figure 8(e). Albahar et al. [ALY*21] synthesized the texture pattern that is relevant to the given details but often with severe artifacts which defect the alignment with the surface of the target 3D mesh as shown in Figure 8(f). SamplerNet was able to preserve the given details and sample the occluded texture patterns while maintaining the alignment with the surface of the target 3D mesh. This is reflected by higher values of both CosSim-A and CosSim-I reported.

**Table 1:** Quantitative evaluation of the generated texture map. The texture maps were generated using the input images with various views. For the "Avg", we used all the images in the range of $[-90^\circ, 90^\circ]$ with $10^\circ$ interval as input and averaged the calculated scores from the generated texture maps.

<table>
<thead>
<tr>
<th>Angle</th>
<th>0° (front view)</th>
<th>90° (side view)</th>
<th>*Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method</td>
<td>LPIPS↑</td>
<td>PSNR↑</td>
<td>SSIM↑</td>
</tr>
<tr>
<td>Isola et al. [IZZE17]</td>
<td>0.3142</td>
<td>18.49</td>
<td>0.5305</td>
</tr>
<tr>
<td>Wang et al. [WLZ*18]</td>
<td>0.3085</td>
<td>18.21</td>
<td>0.5323</td>
</tr>
<tr>
<td>Lazova et al. [LIP19]</td>
<td>0.3047</td>
<td>18.91</td>
<td>0.5604</td>
</tr>
<tr>
<td>Xu et al. [XL21]</td>
<td>0.3713</td>
<td>15.70</td>
<td>0.4941</td>
</tr>
<tr>
<td>Albahar et al. [ALY*21] (CCM)</td>
<td>0.2884</td>
<td>16.57</td>
<td>0.5144</td>
</tr>
<tr>
<td>Ours (SamplerNet)</td>
<td>0.2435</td>
<td>17.53</td>
<td>0.5501</td>
</tr>
<tr>
<td>Ours (SamplerNet + RefinerNet)</td>
<td>0.2230</td>
<td>18.04</td>
<td>0.5853</td>
</tr>
</tbody>
</table>

**Table 2:** Quantitative evaluation of the rendered image. We compared the methods using the input images with various views and rendered the generated texture maps on the target 3D mesh. We used Tex2Shape [APT19] to estimate the target mesh from the input image and applied it for rendering in all methods. For the "Avg", we used all the rendered images in the range of $[-90^\circ, 90^\circ]$ with $10^\circ$ interval as input and averaged the calculated scores.

<table>
<thead>
<tr>
<th>Angle</th>
<th>0° (front view)</th>
<th>90° (side view)</th>
<th>*Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method</td>
<td>CosSim-A↑</td>
<td>CosSim-F↑</td>
<td>LPIPS↑</td>
</tr>
<tr>
<td>Isola et al. [IZZE17]</td>
<td>0.7421</td>
<td>0.7685</td>
<td>0.2017</td>
</tr>
<tr>
<td>Wang et al. [WLZ*18]</td>
<td>0.7266</td>
<td>0.7542</td>
<td>0.2058</td>
</tr>
<tr>
<td>Lazova et al. [LIP19]</td>
<td>0.7510</td>
<td>0.7786</td>
<td>0.1965</td>
</tr>
<tr>
<td>Xu et al. [XL21]</td>
<td>0.7480</td>
<td>0.7711</td>
<td>0.1923</td>
</tr>
<tr>
<td>Albahar et al. [ALY*21] (CCM)</td>
<td>0.7638</td>
<td>0.7881</td>
<td>0.1944</td>
</tr>
<tr>
<td>Ours (SamplerNet)</td>
<td>0.7673</td>
<td>0.7944</td>
<td>0.1931</td>
</tr>
<tr>
<td>Ours (SamplerNet + RefinerNet)</td>
<td>0.7678</td>
<td>0.7938</td>
<td>0.1868</td>
</tr>
</tbody>
</table>
in Table 2. Given the sampled texture with improved alignment, our full method produced visually better results compared to those produced by previous methods in terms of maintaining the appearance details present in the source image and synthesizing missing details exploiting the information from the visible region as shown in Figure 8(h).

![Figure 9: Rendered images of 3D human avatars produced using real human images. The images are from SHHQ [FLJ*22b] dataset. The 3D human pose and the camera parameters were predicted from the images using RSC-Net [XCM*21].](image)

### 4.2. Ablation study

#### 4.2.1. Region-wise Augmentation

The goal of region-wise augmentation is to approximate the transformation caused by DensePose [GNK18]. To evaluate the similarity of the proposed augmentation with DensePose, we compared the results from SamplerNet trained with several different augmentation alternatives that are applied in a region-wise manner. The curriculum learning was excluded in this experiment to examine the effect of augmentation. \( \alpha \) and the probability for the augmentation is fixed to 0.25 and 0.999, respectively. We also compared SamplerNet trained with and without region-wise augmentation using TPS. For the case without it, the control points were set at an equispaced 6 \( \times \) 6 grid followed by shifting them according to a random value from a uniform distribution \( U(0, \alpha) \). We used same \( \alpha \) value for both. As shown in Table 3, SamplerNet trained with TPS in a region-wise manner resulted in the scores closest to that produced by the model trained with DensePose.

#### 4.2.2. Curriculum Learning

We evaluated the effectiveness of the curriculum learning by comparing our networks trained with and without it. The comparison results are shown in Table 4. The results from the networks trained with curriculum learning achieved better scores for all of the evaluation metrics. Moreover, the network trained with curriculum learning produced qualitatively superior results in terms of reproducing the given appearance as shown in Figure 10.

### 4.2.3. Texture Blending

We evaluate the effect of blending the refined texture with the sampled texture using a blending mask. We trained RefinerNet with two different settings: producing the final texture map using the blending mask and directly generating the final texture map. The results are shown in Table 5 and Figure 11. Although the scores of PSNR and SSIM are similar, our qualitative evaluation (Figure 11) shows improvements in maintaining texture details present in input images, which is reflected in the LPIPS score (Table 5). The model trained without blending fails to produce clear details of the logo or patterns present in the source image as indicated in the orange box.

### 5. Discussion

#### 5.1. Limitations

Even though our method achieves better results compared to previous methods, we have limitations that need to be addressed. Our method generates textures based on the surface of the target 3D human mesh which is predicted by Tex2Shape [APTM19]. Because Tex2Shape deforms the SMPL [LMR*15] model to obtain the 3D

---

**Table 3: Comparison between various augmentation techniques.**

<table>
<thead>
<tr>
<th>Augmentation</th>
<th>Region-wise</th>
<th>LPIPS↓</th>
<th>PSNR↑</th>
<th>SSIM↑</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rotate ✓</td>
<td>0.2348</td>
<td>17.99</td>
<td>0.5657</td>
<td></td>
</tr>
<tr>
<td>Translate ✓</td>
<td>0.2468</td>
<td>18.01</td>
<td>0.5669</td>
<td></td>
</tr>
<tr>
<td>TPS x</td>
<td>0.2323</td>
<td>18.15</td>
<td>0.5758</td>
<td></td>
</tr>
<tr>
<td>TPS ✓</td>
<td>0.2270</td>
<td>18.26</td>
<td>0.5817</td>
<td></td>
</tr>
<tr>
<td>DensePose</td>
<td>-</td>
<td>0.2142</td>
<td>18.86</td>
<td>0.6071</td>
</tr>
</tbody>
</table>

**Table 4: Ablation study for the method trained with and without curriculum learning.**

<table>
<thead>
<tr>
<th>Method</th>
<th>LPIPS↓</th>
<th>PSNR↑</th>
<th>SSIM↑</th>
</tr>
</thead>
<tbody>
<tr>
<td>w/o curriculum learning</td>
<td>0.2238</td>
<td>18.09</td>
<td>0.5920</td>
</tr>
<tr>
<td>w/ curriculum learning</td>
<td><strong>0.2220</strong></td>
<td><strong>18.18</strong></td>
<td><strong>0.5935</strong></td>
</tr>
</tbody>
</table>

![Figure 10: Visual comparison of the results produced with and without curriculum learning.](image)
A texture map with improved quality by successfully retaining the textural patterns presented in the visible regions of the source image while maintaining spatial alignment with the surface of the target 3D human mesh. In addition, our method produces a plausible texture map from a non-frontal view image. We verified the effectiveness of curriculum learning and other design choices we made by ablation studies and showed significant quality improvement in the resulting texture by comparing our approach with previous methods.

Acknowledgements

We thank the anonymous reviewers for their invaluable comments; Gyeong Hun Im for collecting the various pose data; Jaedong Kim for his assistance in implementing the mapping function. This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2020-0-00460, A Deep Learning Based Immersive AR Content Creation Platform for Generating Interactive, Context and Geometry Aware Movement from a Single Image).

References


Cha et al. / Generating Texture for 3D Human Avatar from a Single Image using Sampling and Refinement Networks

395


