Supplementary Material for Get3DHuman

Zhangyang Xiong[1,2] Di Kang[3] Derong Jin[2] Weikai Chen[4] Linchao Bao[3]
Shuguang Cui[2,1] Xiaoguang Han[2,1]*
[1]FNii, CUHKSZ [2]SSE, CUHKSZ [3]Tencent AI Lab [4]Tencent America

Contents

1. Summary

In this supplementary, we present additional experimental results and details of our Get3DHuman. Sec. 2 first visualizes rendering results of multi-view, sampling, re-texturing, interpolation of Get3DHuman, and our pseudo-GT. Then in Sec. 3, we further show some examples of GET3D trained with non-T-pose RenderPeople dataset and EVA3D [6]). Finally, we explain the relevant details of our Get3DHuman in Sec. 4.

2. Visualizations results of Get3DHuman

2.1. Multi-view results.

We randomly sampled some examples and then used Blender to render the geometry and appearances from different angles. Fig. 5 shows these results. From different views, our Get3DHuman is able to obtain high-quality geometry and texture results.

2.2. Sampling results.

Fig. 6 visualize the textured mesh sampling from a Gaussian distribution. The results show that our Get3DHuman is able to generate 3D textured human mesh with diverse geometries and appearances.

*Corresponding author: hanxiaoguang@cuhk.edu.cn

2.3. Re-texturing results.

Fig. 7 visualize the re-texturing results of the fixed geometry by different texture latent codes. Examples of re-texturing given shapes. We can see different textures are diverse, plausible, and suitable for the given shape since our texture branch is conditioned on shape branch features.

2.4. Interpolation results.

Fig. 8 and 9 visualize the interpolation results of two sets of shape/texture latent codes to generate the right-/left-most examples, then interpolate both the shape and texture latent codes to generate the in-between examples. More results show in the video.

2.5. Inversion results.

Fig. 8 and 9 visualize the interpolation results of two sets of shape/texture latent codes to generate the right-/left-most examples, then interpolate both the shape and texture latent codes to generate the in-between examples. More results show in the video.

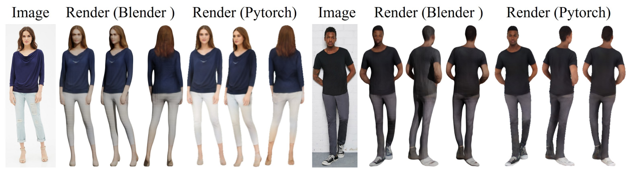


Figure 1: More inversion results with two rendering methods.

2.6. Visualizing our pesudo-GT.

We extract pseudo-GT by using two prior networks, a human image synthesis network [3] and a single-view reconstruction network [7], which are trained using our RenderPeople data. The images synthesized by [3] have a higher probability of some defects (limb deformities, incomplete limbs and etc.), and the reconstruction result of

[7] is insufficient to deal with uncommon postures and exotic clothes, and we cannot directly use their results. Therefore, we manually selected 69,069 reasonable results from 300,000 generated data as our pseudo-gt to train Get3DHuman.

Noted that although our pseudo-GT data is manually selected, it is definitely not comparable to commercial data sets (such as RenderPeopel) in terms of geometry and texture quality. Fig. 2 visualize several examples of our pseudo-GT.

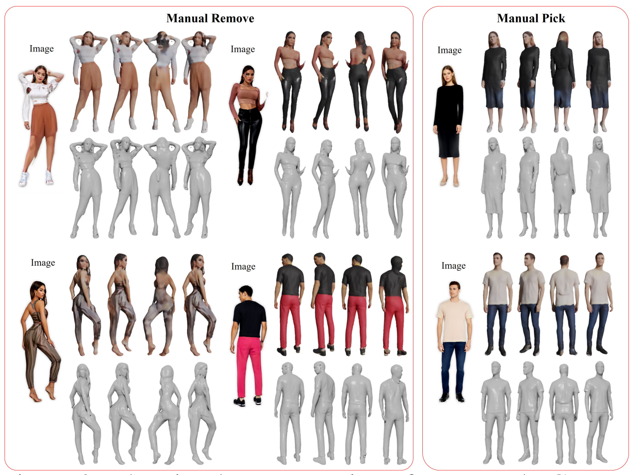


Figure 2: The visual representation of our pesudo-GT. We remove the poor quality results (left) through manual selection and only keep some high quality results (right).

3. Results of GET3D and EVA3D

3.1. Results of GET3D [4] (our Renderpeople data)

The original GET3D [4] paper is trained in a T-posed RenderPeople dataset. However, our purchased RenderPeople data contains 396 high-quality non-T-posed 3D human models with high-resolution texture maps that could be rendered into photorealistic images. We attempt to train GET3D with our RenderPeople data, but the results are poor in terms of geometry and textures, see Fig. 3, possibly due to the large pose space and limited training data. Its results could possibly be improved with more training data, but high-quality 3D data are expensive and difficult to obtain.

3.2. Results of EVA3D [6].

EVA3D has just opened its code recently, and Fig. 4 visualizes four sampling results of EVA3D generated from a pretrain model from Github [5]. It introduces SMPL prior to the 3D generation network. Observing its results, the image quality in the front view is far better than other views, and the overall geometry is too flat and has many artifacts. These shortcomings are also reasonable, because they only

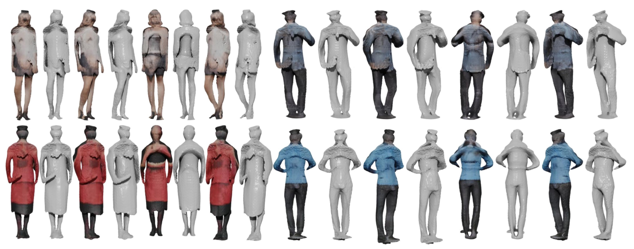


Figure 3: The results of GET3D trained with our non-T-pose Renderpeople data.

use SMPL prior and front view images for training, and lack data from non-front view information.

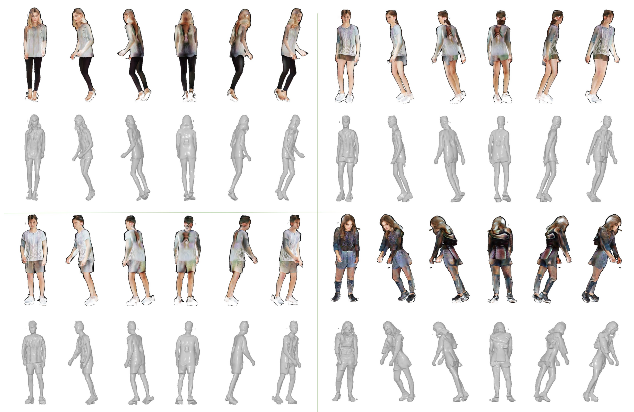


Figure 4: The results of EVA3D with the pretrain model $(512x256 - deepfashion)$ from Github [5]. Note that, in this figure, all images are generated from the source code of EVA3D, not rendered by Blender.

4. Notes about the Get3DHuman

In our Get3DHuman, We obtain 3D human meshes with diverse geometries and textures by sampling two latent codes (shape & texture) from a Gaussian distribution. Given a shape latent code Z_s, Get3DHuman generates a shape feature F_s, a shape feature volume F_{sv}, and produces a high-quality human shape by using a fixed PIFu shape decoder f_s. Given a shape feature F_s and a texture latent code Z_t, Get3DHuman generates a texture feature volume F_{tv}, and textures the shape by using a fixed PIFu texture decoder f_t.
Feature Volume and PIFu decoder In Get3DHuman, each shape feature volume and its corresponding texture feature volume represents a 3D textured human model. The size of shape & texture feature volumes are both $\{bs, 256, 128, 128\}$, bs means the number of batch size.

During the training process of Get3DHuman, the shape PIFu decoder and the texture PIFu decoder are fixed, without training.
Training data There are mainly two kinds of training data in our pseudo-GT: with latent code and without latent code. All pseudo-GT data are utilized in *Adv. losses*, the pseudo-

GT with latent codes which are manual pick and generated from StyleGAN-Huma is used in *latent prior losses* (See Fig.2 in paper),

Geometry evaluation settings. Similar to any GAN, we adapt Fréchet point cloud distance (FPD) [2] to evaluate the diversity and quality of the generated shapes. Specifically, we first sample a number of points (4096) on the mesh surface and pass this point cloud to a pre-trained shape classification network using DGCNN [8] as the backbone. The feature vectors from this classification network are used to calculate FPD for the generated shapes.

We also test Chamfer Distance based (d_{CD}) Coverage (COV) and Minimum Matching following [1, 4] to evaluate the similarity between a set of generated meshes and the reference set of pseudo-GT meshes. Our test set contains 2k textured model estimated from recon prior network as pseudo-GT. For FPD, we randomly generate 2k shapes for evaluation. For COV and MMD, Following GET3D, we randomly generate 5 times as many shapes as the test set (i.e. 10k).

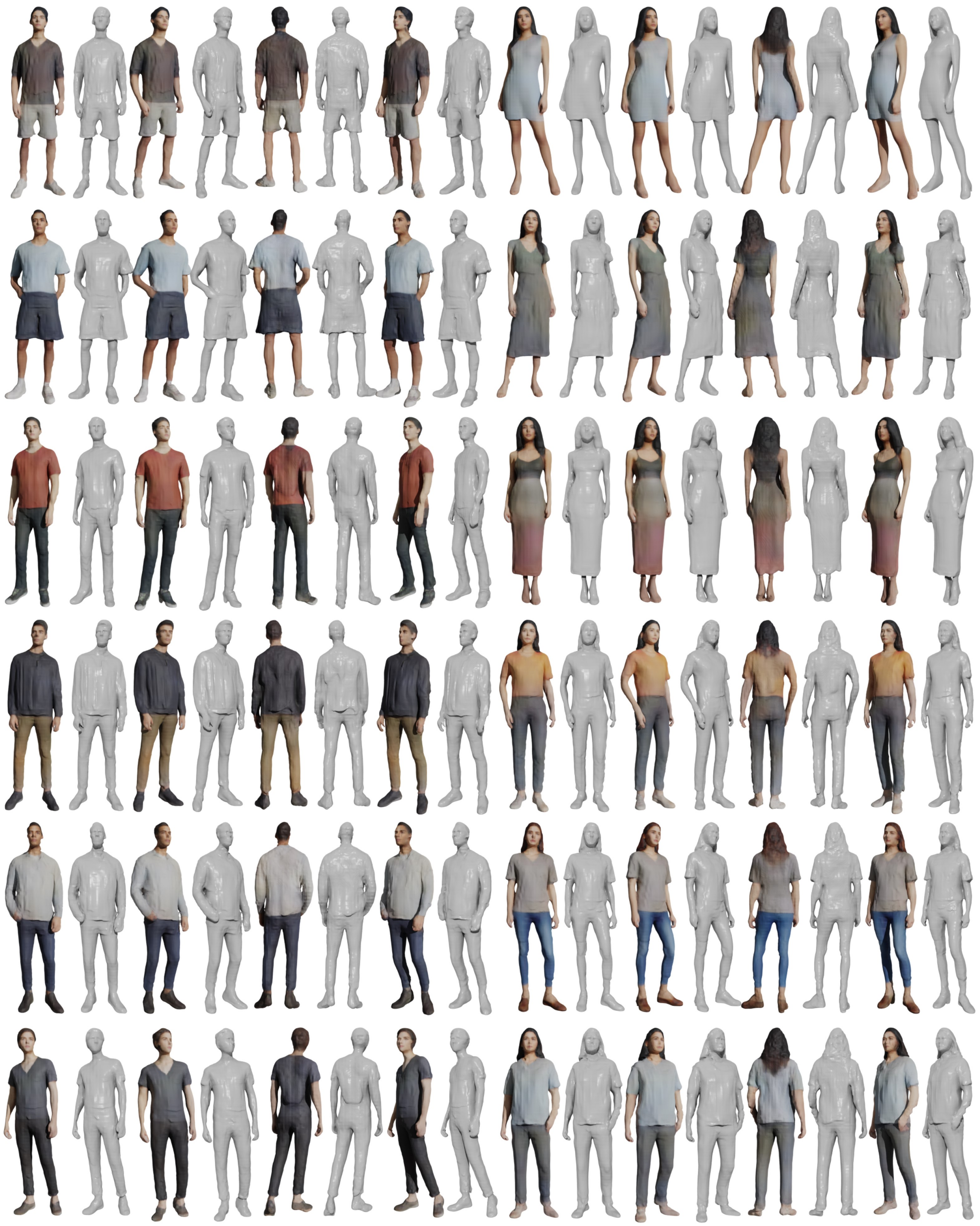


Figure 5: Visualization multi-view images rendering by blender.

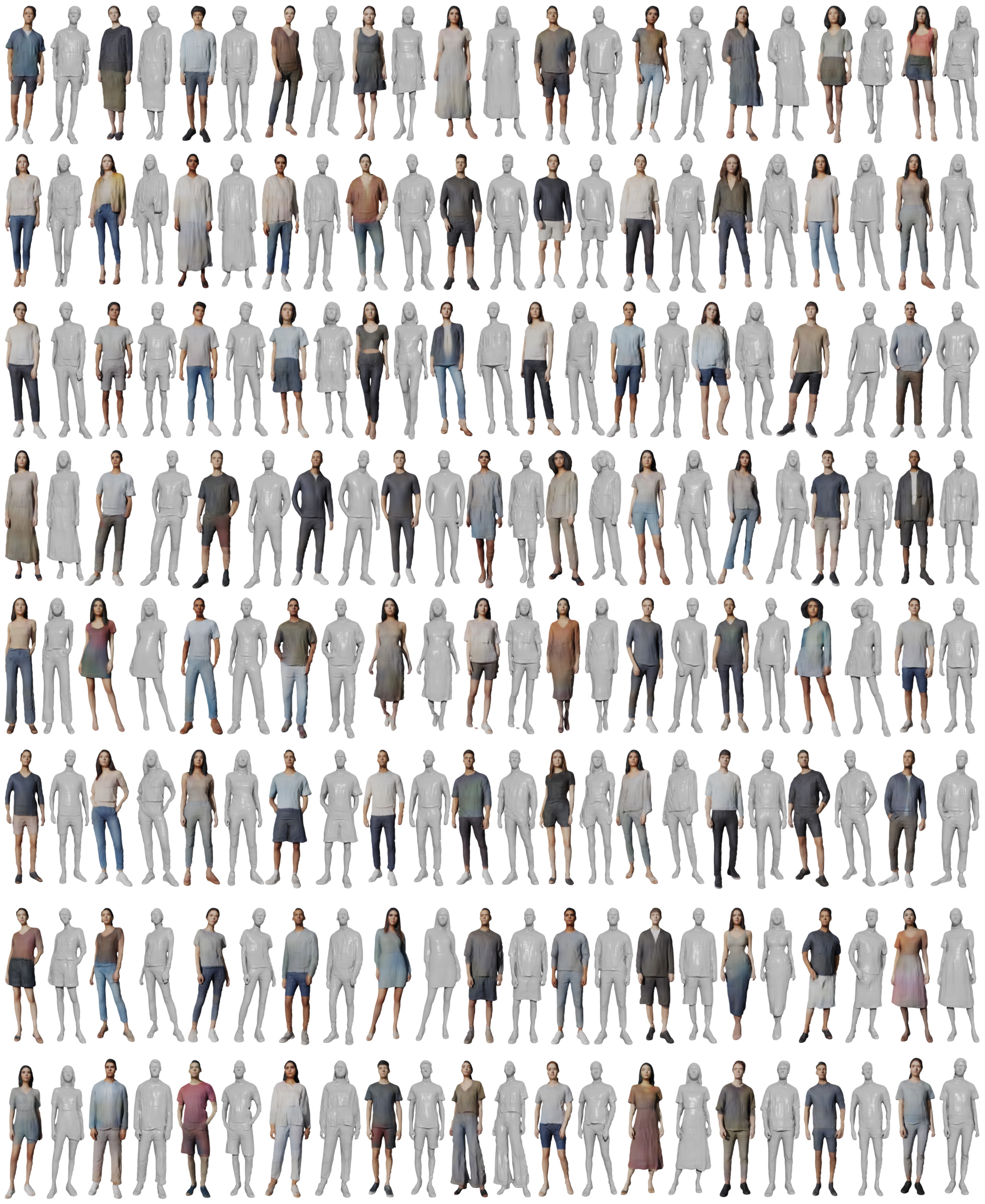


Figure 6: Visualize the results of generated textured mesh.

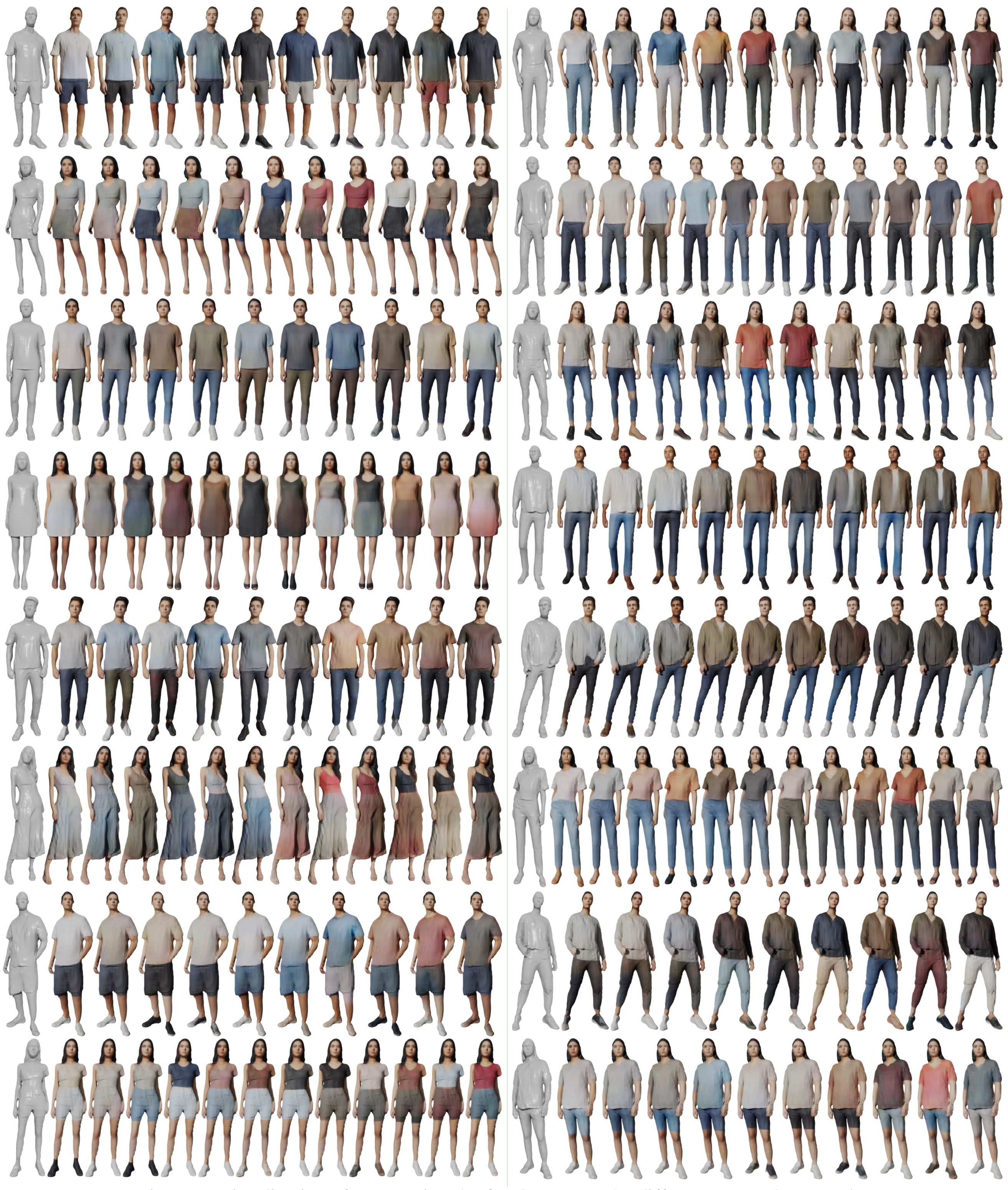


Figure 7: Visualization of re-texturing the fixed geometry by different texture latent code.

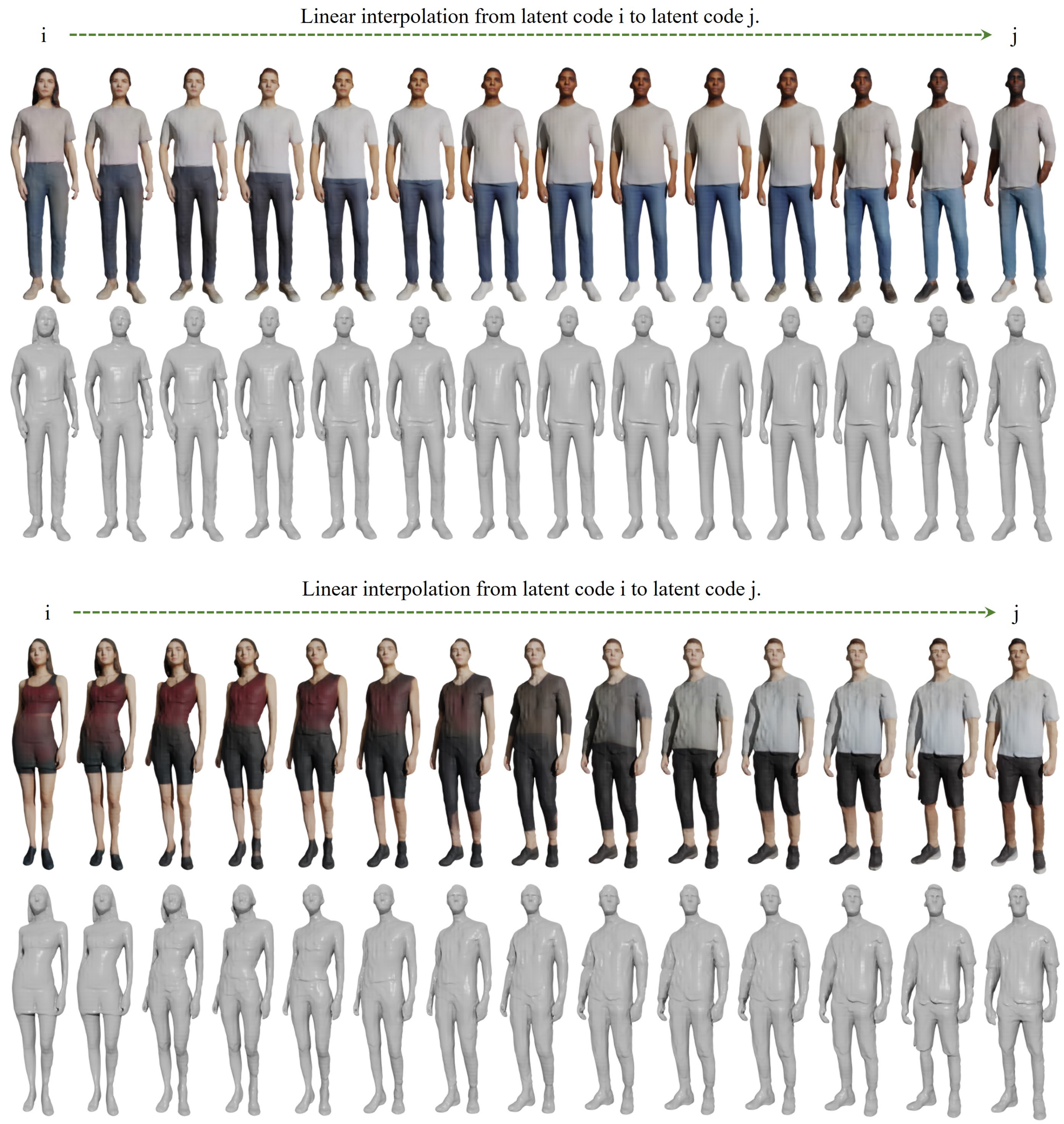
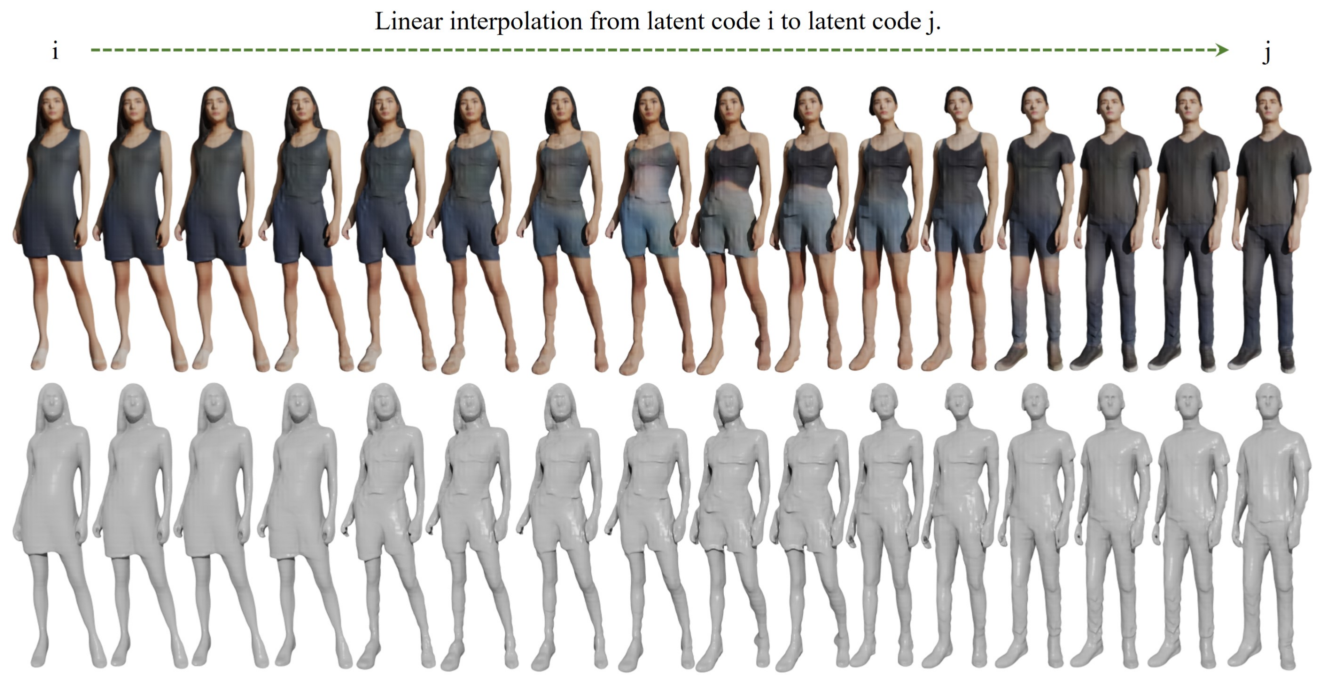

Linear interpolation from latent code i to latent code j.
i
j
Linear interpolation from latent code i to latent code j.
i
j

Figure 8: Interpolation examples. We randomly sample two sets of shape/texture latent codes to generate the right-/left-most examples, then interpolate both the shape and texture latent codes to generate the in- between examples.



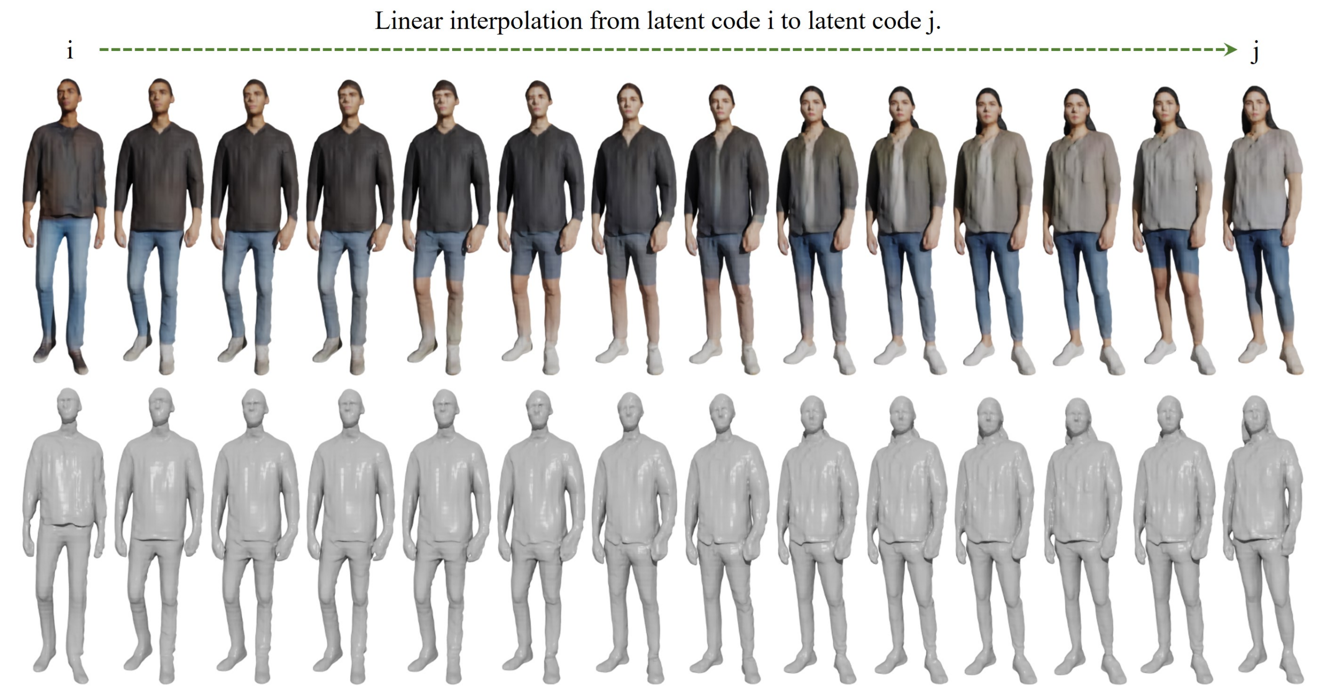


Figure 9: Interpolation examples. We randomly sample two sets of shape/texture latent codes to generate the right-/left-most examples, then interpolate both the shape and texture latent codes to generate the in- between examples.

References

[1] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3D point clouds. In *International conference on machine learning*, 2018.

[2] Junseok Kwon Dong Wook Shu, Sung Woo Park. 3d point cloud generative adversarial network based on tree structured graph convolutions. *arXiv*, 2019.

[3] Jianglin Fu, Shikai Li, Yuming Jiang, Kwan-Yee Lin, Chen Qian, Chen-Change Loy, Wayne Wu, and Ziwei Liu. StyleGAN-Human: A data-centric odyssey of human generation. *arXiv preprint*, 2022.

[4] Jun Gao, Tianchang Shen, Zian Wang, Wenzheng Chen, Kangxue Yin, Daiqing Li, Or Litany, Zan Gojcic, and Sanja Fidler. GET3D: A generative model of high quality 3D textured shapes learned from images. In *NeurIPS*, 2022.

[5] Fangzhou Hong. https://github.com/hongfz16/eva3d.

[6] Fangzhou Hong, Zhaoxi Chen, Yushi Lan, Liang Pan, and Ziwei Liu. EVA3D: Compositional 3D human generation from 2d image collections. *arXiv*, 2022.

[7] Shunsuke Saito, Zeng Huang, Ryota Natsume, Shigeo Morishima, Angjoo Kanazawa, and Hao Li. PIFu: Pixel-aligned implicit function for high-resolution clothed human digitization. *arXiv preprint arXiv:1905.05172*, 2019.

[8] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. Dynamic graph cnn for learning on point clouds. *TOG*, 2019.