Paint-it: Text-to-Texture Synthesis via Deep Convolutional Texture Map Optimization and Physically-Based Rendering
— Supplementary Material —

Kim Youwang[1,2,4*] Tae-Hyun Oh[4,5,6] Gerard Pons-Moll[1,2,3]

[1]University of Tübingen [2]Tübingen AI Center, Germany [3]Max Planck Institute for Informatics, Germany
[4]Dept. of Electrical Engineering, POSTECH [5]Grad. School of AI, POSTECH
[6]Institute for Convergence Research and Education in Advanced Technology, Yonsei University

In this supplementary material, we provide additional details and results that are not included in the main paper due to the space limit. The attached video includes a brief introduction and interesting qualitative results of *Paint-it*.

Contents

A. Details of *Paint-it*

A.1. DC-PBR: Network Design

The main contribution of our work is the proposed DC-PBR parameterization for optimizing the physically-based rendering (PBR) texture maps. Instead of pixel-based parameterization of the PBR texture maps, we introduce the fixed random noise input $\mathbf{z} \sim \mathcal{N}(0, \mathbf{I}) \in \mathbb{R}^{H \times W \times 3}$, and a randomly initialized U-Net with skip connections, $\mathcal{T}_\theta$. We obtain the PBR texture map $[\mathbf{K}_\theta^{\text{d}}, \mathbf{K}_\theta^{\text{rm}}, \mathbf{K}_\theta^{\text{n}}] = \mathcal{T}_\theta(\mathbf{z}) \in \mathbb{R}^{H \times W \times (3+2+3)}$, for every iteration of the synthesis optimization.

Our design choice of DC-PBR is inspired by the Deep Image Prior [22], and we extended it to re-parameterize the PBR texture maps for the text-driven texture map synthesis task. We use an encoder-decoder ("hourglass") architecture with skip connections between encoder and decoder features for our neural re-parameterization, DC-PBR $\mathcal{T}_\theta$. For the network hyperparameters, we used the *default architecture* of the Deep Image Prior, *i.e.*, five levels of downsampling and upsampling layers for the encoder and decoder. We encourage readers to refer to the details in Fig. 21 of Deep Image Prior. We empirically set the learning rate as $5 \cdot 10^{-4}$ and the total iteration for PBR texture synthesis as 1000.

A.2. Details of SDS Loss

Recall that we optimize the DC-PBR given the text with the Score-Distillation Sampling (SDS).

SDS Loss for Multi-view Mesh Images. We adopt some engineering to obtain high-fidelity and multi-view consistent PBR texture maps. When computing the SDS loss, we need the rendered image of the textured mesh. We randomly sample camera poses in multi-view and render N view images. We sample the elevation angle as $\varphi_{\text{elev}} \sim \mathcal{U}(-\frac{\pi}{3}, \frac{\pi}{3})$, and the azimuth angle as $\varphi_{\text{azim}} \sim \mathcal{U}(0, 2\pi)$. We set $N = 4$ for most cases. When optimizing DC-PBR for humans and animals, we increase the generation quality of the face regions by additionally rendering the face-focused images. We translate the mesh so that the head can be the center of the world coordinate and render it with $\varphi_{\text{elev}} \sim \mathcal{U}(-\frac{\pi}{6}, \frac{\pi}{3})$ and $\varphi_{\text{azim}} \sim \mathcal{U}(0, 2\pi)$. For human and animal cases, we use a total $N = 8$ views for computing SDS loss, where four views are for the full body, and the others are for face regions. We also use the directional text prompt engineering as in prior arts [4, 16] to mitigate the "Janus problem".

When computing the SDS loss, at each iteration, we syn-

*Work done during a visiting research period at the University of Tübingen.

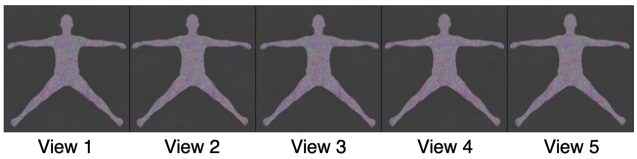


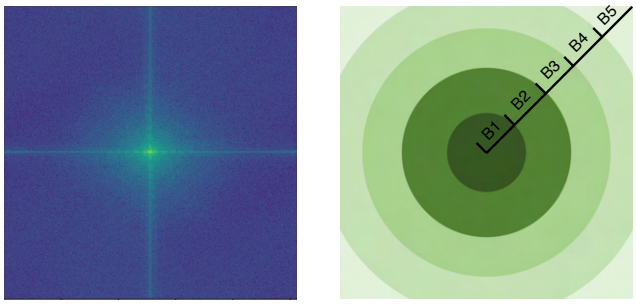
Figure S1. (For SDS analysis) Textured mesh image rendered from adjacent camera views $-2^\circ < \varphi_{\text{elev}} < 2^\circ$ and $-2^\circ < \varphi_{\text{azim}} < 2^\circ$. The images are slightly different but look almost identical.

chronize the noise ϵ and the noising timestep t for multi-view rendered images, *i.e.*, we randomly sample a single ϵ and t for each synthesis iteration and add the same amount of noise to the multi-view mesh images. Finally, we sample the noising timestep t, from the distribution $\mathcal{U}(t_{\text{min}}, t_{\text{max}})$. We start by $[t_{\text{min}}, t_{\text{max}}] = [0.2, 0.98]$, and linearly narrows down the distribution so that it become $[t_{\text{min}}, t_{\text{max}}] = [0.3, 0.5]$ by the end. We empirically set the ranges for t_{min} and t_{max}.

Why is SDS Loss a Noisy Signal?. In the main paper, we denote that the SDS loss is a noisy signal. By noisy, we refer to the incoherent nature of the SDS loss. From Sec. 3.1 and Eq. 1 in the main paper, we notice the SDS loss is dependent on the randomly sampled Gaussian noise $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ and the noising timestep $t \sim \mathcal{U}(t_{\text{min}}, t_{\text{max}})$. We sample ϵ and t for every iteration of the PBR texture map synthesis; thus, the SDS loss is highly likely to give incoherent direction for updating the DC-PBR $\mathcal{T}_\theta$. Moreover, as aforementioned, we use multi-view rendered images. Multi-view images contain different visible mesh parts, potentially providing incoherent update directions for $\mathcal{T}_\theta$.

To show the incoherent SDS gradient, we design a toy example. Given a mesh and a text input, we render the mesh in N adjacent views, *i.e.*, we sample elevation and azimuth in the range $-2^\circ < \varphi_{\text{elev}} < 2^\circ$ and $-2^\circ < \varphi_{\text{azim}} < 2^\circ$. Since the camera views are closely distributed, the rendered mesh images would look almost identical (see Fig. S1).

We investigated the backward gradients $\nabla \mathcal{L}_{\text{SDS}}$ applied on the diffuse map $\mathbf{K}^{\text{d}}_\theta$, computed from each view, with the identical text prompt and ϵ and t. We obtain a flattened, stacked gradient matrix from per-view SDS loss gradients on the diffuse map. Formally, we obtain the gradient matrix $\mathbf{G} \in \mathbb{R}^{N \times F}$, where N denotes the number of rendered views, and F denotes the flattened dimension of the gradient. Even though we compute the SDS loss with the same text prompt, same ϵ and t, and the almost identical rendered images indistinguishable in eyes, we observe that the gradient matrix $\mathbf{G}$ is in high rank. We first obtain the singular values of the gradient matrix $\mathbf{G}$ using the Singular Value Decomposition (SVD) and investigate the ratio of all the singular values with the smallest singular value. From a toy example, where $N = 5$, we obtain the singular value ratio as $[2.1868, 1.7937, 1.7838, 1.6689, 1.0000]$, *i.e.*, the highest and lowest singular values do not deviate too much in terms of the scale, showing $\mathbf{G}$ is high rank. In other words, the SDS loss, computed with a pre-trained text-to-image diffusion model, provides incoherent guidance from multiple views and guides the optimization in incoherent directions.



(a) FFT of texture map (b) Frequency bands

Figure S2. (a) Visualization of the FFT image of the diffuse map. (b) Visualization of the pre-defined non-overlapping frequency bands. Center: low frequency, Outer: high frequency.

We claim the SDS loss is a noisy signal from this observation. In Sec. 4.2, we empirically show our DC-PBR filters out the high-frequency noisy signal by inherently scheduling the optimization curriculum. Thus, DC-PBR is effective when combined with the noisy SDS loss.

A.3. Details of Frequency-based Analysis

We plot the energy-iteration plot in our frequency-based analysis of the proposed DC-PBR (Fig. 4 in the main paper). We first performed the Fast Fourier Transform (FFT) of the diffuse texture map obtained in each iteration. See Fig. S2a for the FFT result. Then, we define five non-overlapping frequency bands, depending on the radius from the center of the FFT image, as in Fig. S2b. Note that the ranges of the frequency bands are fixed during the optimization. Finally, we compute the energy of each frequency band by summing all the frequency response magnitudes (either absolute value or square works) in each frequency band.

In Fig. 4-left, the optimization starts from a monotonic gray image, which has high zero-frequency energies, *i.e.*, DC component. However, note that we visualize each frequency band's 'average' energy. DC component occupies only a tiny area ($\ll 1\%$) in the lowest frequency band; thus, the pixel brightness hardly affects the band's energy.

During *Paint-it* optimization, our DC-PBR representation automatically schedules the curriculum to learn low frequency first, then mid frequency, and filters out the highest frequency, *i.e.*, noise.

A.4. Details of User Study

We conducted a user study to assess the realism of the synthesized PBR texture maps. We showed ten untextured meshes from the Objaverse [6] dataset and showed five different

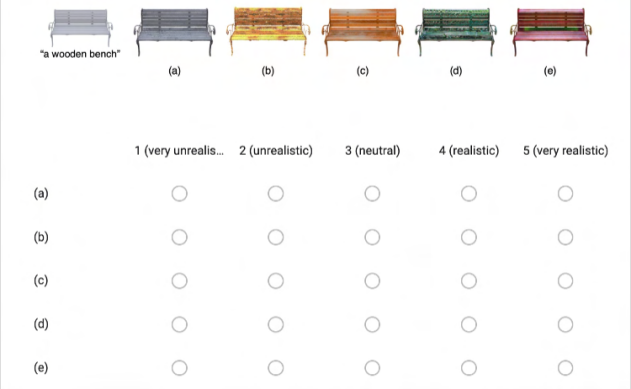


Figure S3. The users were asked to rate the realism of the rendered mesh images, textured with five different methods. The order of the methods was randomly shuffled for each question.

Figure S4. Feature grid+MLP (baseline) vs. DC-PBR (ours)

results obtained from the methods: Latent-Paint [13], Fantasia3D [4], Text2Tex [3], TEXTure [18], and *Paint-it* (ours). We asked 30 users with engineering/non-engineering backgrounds to rate the realism of the rendered images in the score range 1 to 5, *i.e.*, 1: very unrealistic, 2: unrealistic, 3: neutral, 4: realistic, and 5: very realistic. The order of the methods was randomly shuffled for fairness. The interface of the user study is shown in Fig. S3.

A.5. Details & Discussion about Optimization Time

The optimization time for synthesizing PBR texture maps takes about 15 min. for general object meshes in Objaverse [6]. For more complex cases, such as 3D humans or animals, we additionally use face-focused mesh renderings; thus, it takes about 30 min. to complete. We optimized both the baseline (Eq. (4) in the main paper, pixel optimization) and our DC-PBR until convergence, and there's no significant time difference. Instead, DC-PBR helps the optimization with noisy SDS loss find a better solution than the vanilla pixel optimization. We use a single NVIDIA RTX A6000 GPU for the optimization.

Extending *Paint-it* to large-scale 3D scenes would be an interesting future direction. However, *Paint-it*'s PBR texture generation for large-scale scenes would take a longer optimization time. As suggested in FPRF [10], we may try semantic feature distillation to accelerate the optimization when stylizing large-scale 3D scenes using SDS.

B. Additional Experiment

B.1. More Baseline for PBR Representation

For a PBR representation baseline, other than direct pixel optimization (Eq. (4) in the main paper), we implement a multi-resolution hash encoding of grid features and subsequent MLP [14] to model the disentangled PBR texture maps. In Fig. S4, our DC-PBR representation yields smoother and more vivid texture results than the new baseline. In Table S1, our DC-PBR obtains a better FID score than other PBR representation baselines (Pixel optim., & Feat. grid+MLP). We postulate that the SDS gradients only propagate to local hash grids in the new baseline, lacking non-local texture smoothness, as supported in [12]. On the other hand, the spatial CNN kernels of our DC-PBR are beneficial in naturally imposing texture smoothness.

B.2. Effects of DC-PBR Design Choice

In this section, we investigate the effect of our DC-PBR design choice. Specifically, we study the effect of our U-Net with skip connections in synthesizing the PBR texture maps.

As discussed in the main paper and Sec. A.1, we use a *randomly initialized* U-Net with skip connections, shortly, U-Net+skip. Deep Image Prior [22] used U-Net+skip and claimed skip connections inherently promote self-similarity across multi-scales, which is beneficial for inverse problems. We wanted to investigate how the skip connections affect the DC-PBR optimization with SDS loss. Thus, we conduct the same experiment as in Sec. 4 of the main paper, but with U-Net, without (*w/o*) skip connections.

Fitting behavior. Following the experiment in Sec. 4.1, we fit a randomly initialized U-Net *w/o* skip connections given a ground-truth texture map. In Fig. S5b, the energy-iteration plot shows that parameterizing a texture map with U-Net *w/o* skip connection fails to fit high frequency.

DC-PBR synthesis behavior. Similarly, for our task, *i.e.*, text-driven DC-PBR optimization, U-Net *w/o* skip connection fails to increase the mid-to-high frequency band (purple line), resulting in blurry texture maps. In contrast, our DC-PBR, parameterized in U-Net+skip, successfully increases the mid-to-high frequency band, synthesizing fine-grained texture maps. We conclude that the skip connections are in charge of synthesizing fine-grained, high-frequency details of texture maps. This observation aligns with the Deep Image Prior's claim, where skip connections benefit the inverse problems with multi-scale feature awareness. We additionally showed the frequency level behavior of the skip connection through the experiments (see Fig. S6).

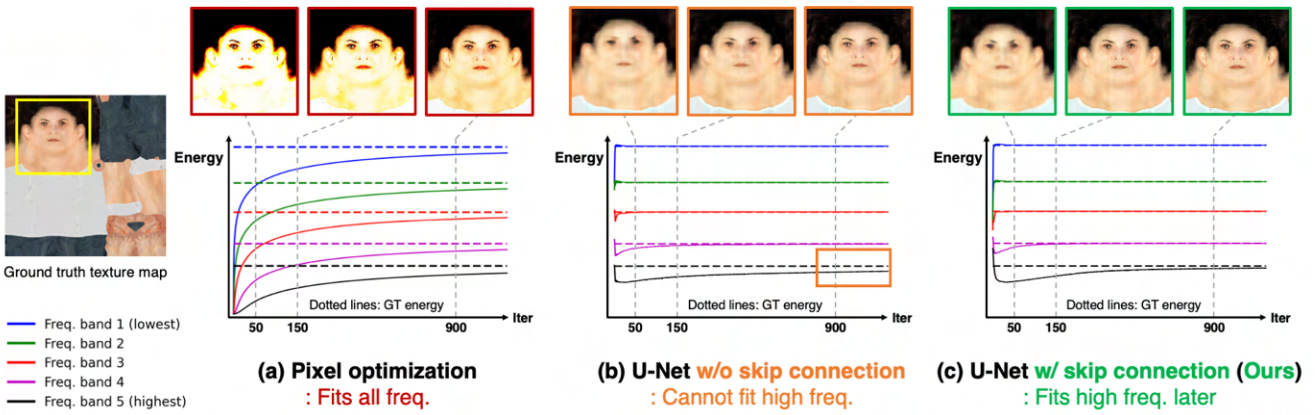


Figure S5. **Effect of skip connection: Texture map fitting.** When fitting a texture map with different parameterizations, U-Net without skip connections fails to fit the highest frequency band. This hints that the skip connections are responsible for representing fine-grained details.

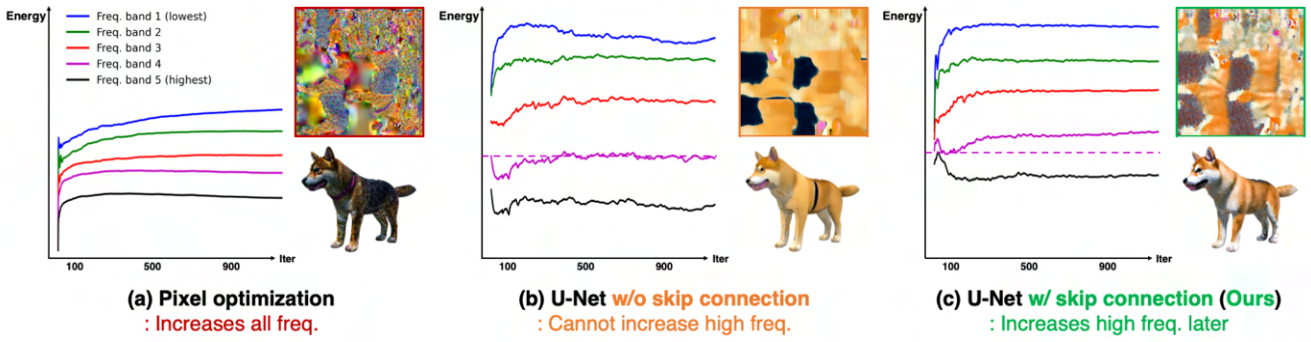


Figure S6. **Effect of skip connection: Text-driven texture synthesis.** When synthesizing PBR texture maps with different parameterizations, U-Net without skip connections cannot increase the mid-to-high frequency band (purple). Proposed DC-PBR with U-Net and skip connections can synthesize fine-grained details in texture maps, resulting in high-fidelity synthesis results.

	Pixel optim.	Only $\mathbf{K}^{\mathrm{d}}$	Feat. grid+MLP	**DC-PBR (Ours)**
FID (↓)	216.6	59.39	55.98	**34.46**

Table S1. FID for ablation study

C. Additional Results

C.1. More Quantitative Comparison

In Tab. S1, we report FID scores for the ablation study (in Sec. 5.3 in the main paper). We used 410 meshes from Objaverse to get the real sample set and added 50 meshes for the generated sample set. The results support that our full DC-PBR enhances the realism of the generated texture.

C.2. More Qualitative Results

We provide more qualitative results of our *Paint-it*. Given untextured meshes from Objaverse [6] and RenderPeople [1], we obtain the text prompts from 1) manually writing the requirements, *e.g.*, a Spiderman lego minifigure, or 2) generating an automatic text caption using multi-modal large-language models, *e.g.*, GPT-4. Then, we conduct *Paint-it* optimization to synthesize PBR texture maps and render the textured meshes (see Fig. S7 and Fig. S8).

C.3. More Comparison Results

In Figs. S9 and S10, we provide more qualitative results that compare *Paint-it* and recent competing methods [3, 4, 13, 18]. As in Fig. 6 of the main paper, we synthesize texture maps using each method for the same untextured meshes and text prompts. Overall, *Paint-it* synthesizes much realistic and vivid texture on the meshes, thanks to the PBR texture representation and texture smoothness induced by our proposed DC-PBR. Note that Fantasia3D [4] also synthesizes PBR materials, but in a per-point independent manner; thus, it lacks texture smoothness and yields substantial jitterings. Moreover, given an untextured mesh, Fantasia3D converts the mesh into a signed distance function (SDF) representation, DMTet [19]. Such auxiliary re-meshing optimization leads to severe geometric quality degradation, *e.g.*, floating artifacts on a toy bicycle example, in Fig. S9.

Figure S7. **Qualitative results of *Paint-it*: Objaverse dataset [6]**. Given any untextured mesh from the existing mesh database, *Paint-it* synthesizes high-fidelity, locally smooth, and realistic object PBR texture maps.

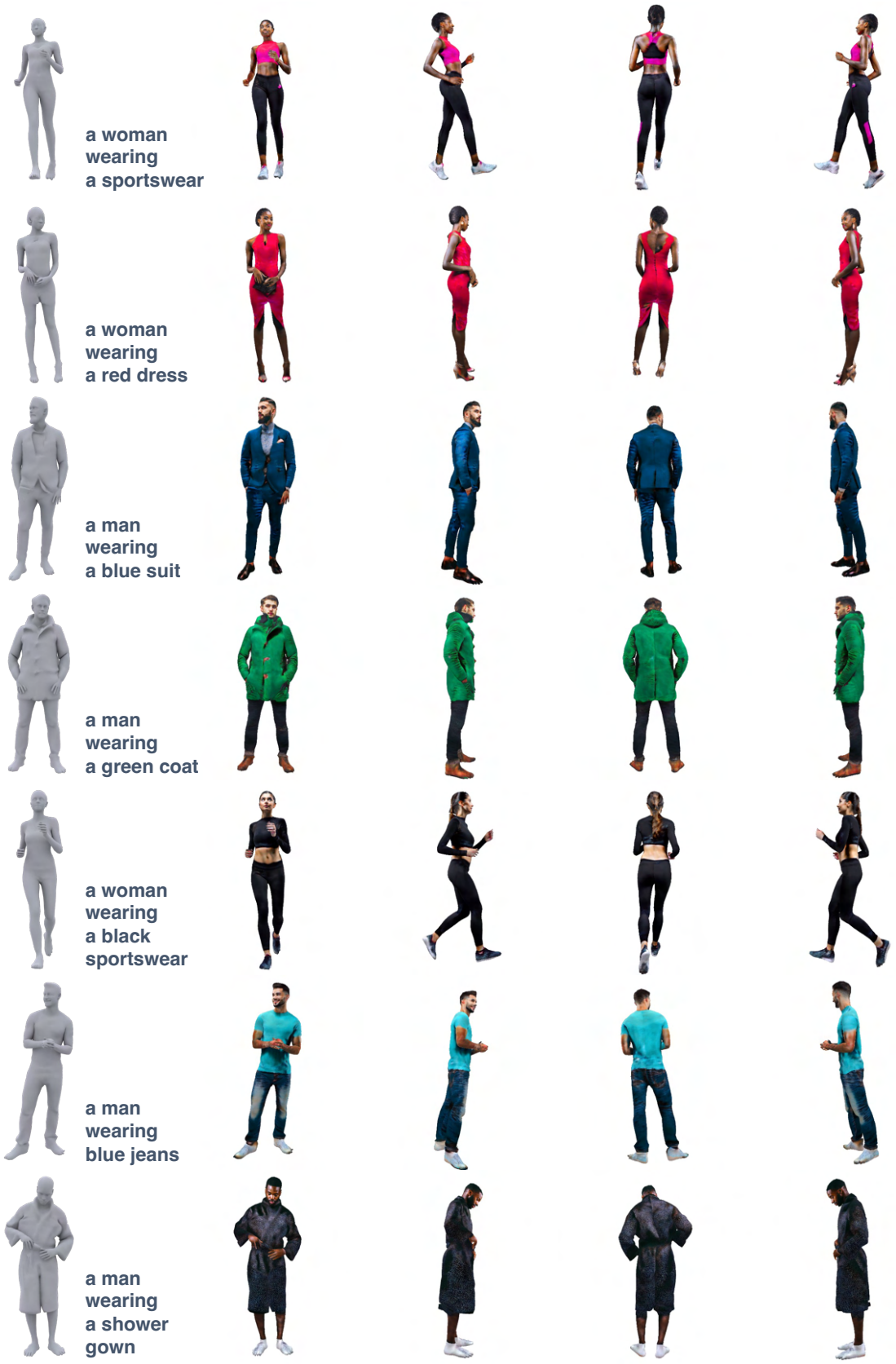


Figure S8. **Qualitative results of *Paint-it*: RenderPeople dataset [1].** Given untextured clothed human meshes, *Paint-it* synthesizes high-fidelity, vivid, and multi-view consistent human and cloth PBR textures. We render four different views of the textured mesh.

Figure S9. **Comparison results: Objaverse dataset [6].**

Figure S10. **Comparison results: RenderPeople dataset [1].**

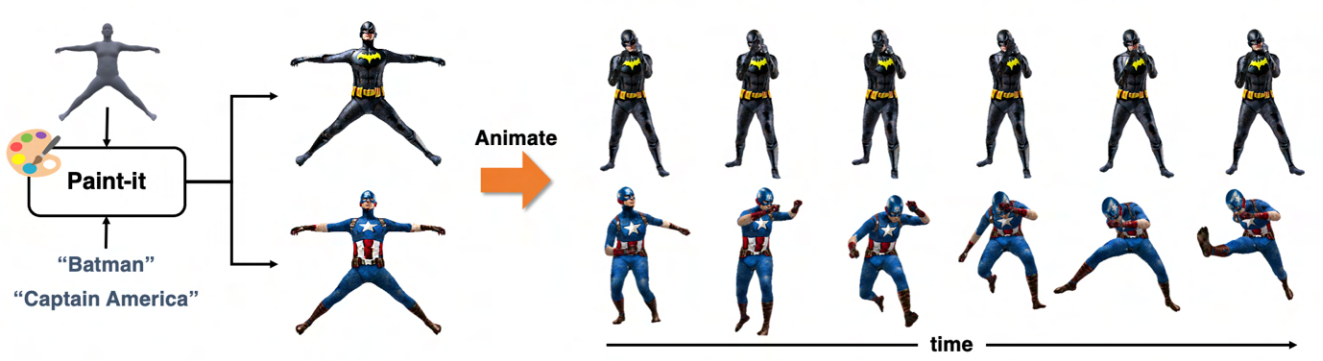


Figure S11. ***Paint-it*: Dynamic virtual 3D humans.** We synthesize PBR texture maps given text and rigged mesh, *e.g.*, SMPL-X [15], using *Paint-it*. Then, we animate the textured 3D humans using sequential pose parameters (can be retrieved [26] or generated [11]).

C.4. *Paint-it* for Animated Meshes

Paint-it can also synthesize high-quality PBR texture maps for animatable meshes and generate dynamic 3D assets. Since *Paint-it* does not perform a re-meshing process and preserves the original UV texture coordinates, we can synthesize maps for any rigged meshes, *e.g.*, T-posed human mesh, and animate with any motion sequences.

In this paper, we used SMPL-X [15]. We first take the canonical posed SMPL-X mesh and synthesize PBR texture maps using *Paint-it*. To animate the textured avatars, one may use the motion captured mesh sequences of 3D human bodies [5, 8, 9, 23, 24], faces [7, 27, 28] or even animals [2, 25]. Generative models for natural body or facial motions [17, 20, 21] could also be applied for animation. We may also use the posed meshes and perform advanced augmentations as proposed in CLIP-Actor [26]. We visualize the synthesized animated meshes in Fig. S11.

References

[1] https://renderpeople.com/, 2023. 4, 6, 8

[2] Benjamin Biggs, Oliver Boyne, James Charles, Andrew Fitzgibbon, and Roberto Cipolla. Who left the dogs out?: 3D animal reconstruction with expectation maximization in the loop. In *European Conference on Computer Vision (ECCV)*, 2020. 9

[3] Dave Zhenyu Chen, Yawar Siddiqui, Hsin-Ying Lee, Sergey Tulyakov, and Matthias Nießner. Text2tex: Text-driven texture synthesis via diffusion models. In *IEEE International Conference on Computer Vision (ICCV)*, 2023. 3, 4

[4] Rui Chen, Yongwei Chen, Ningxin Jiao, and Kui Jia. Fantasia3d: Disentangling geometry and appearance for high-quality text-to-3d content creation. In *IEEE International Conference on Computer Vision (ICCV)*, 2023. 1, 3, 4

[5] Junhyeong Cho, Kim Youwang, and Tae-Hyun Oh. Cross-attention of disentangled modalities for 3d human mesh recovery with transformers. In *European Conference on Computer Vision (ECCV)*, 2022. 9

[6] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 2, 3, 4, 5, 7

[7] Yao Feng, Haiwen Feng, Michael J. Black, and Timo Bolkart. Learning an animatable detailed 3D face model from in-the-wild images. *ACM Transactions on Graphics (SIGGRAPH)*, 40(8), 2021. 9

[8] Shubham Goel, Georgios Pavlakos, Jathushan Rajasegaran, Angjoo Kanazawa*, and Jitendra Malik*. Humans in 4D: Reconstructing and tracking humans with transformers. In *IEEE International Conference on Computer Vision (ICCV)*, 2023. 9

[9] Vladimir Guzov, Aymen Mir, Torsten Sattler, and Gerard Pons-Moll. Human poseitioning system (hps): 3d human pose estimation and self-localization in large scenes from body-mounted sensors. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 9

[10] GeonU Kim, Kim Youwang, and Tae-Hyun Oh. FPRF: Feed-forward photorealistic style transfer of large-scale 3D neural radiance fields. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2024. 3

[11] Jihoon Kim, Jiseob Kim, and Sungjoon Choi. Flame: Free-form language-based motion synthesis & editing. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2022. 9

[12] Zhaoshuo Li, Thomas Müller, Alex Evans, Russell H Taylor, Mathias Unberath, Ming-Yu Liu, and Chen-Hsuan Lin. Neuralangelo: High-fidelity neural surface reconstruction. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 3

[13] Gal Metzer, Elad Richardson, Or Patashnik, Raja Giryes, and Daniel Cohen-Or. Latent-nerf for shape-guided generation of 3d shapes and textures. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 3, 4

[14] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (SIGGRAPH)*, 41(4):102:1–102:15, 2022. 3

[15] Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed A. A. Osman, Dimitrios Tzionas, and Michael J. Black. Expressive body capture: 3D hands, face,

and body from a single image. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 9

[16] Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. In *International Conference on Learning Representations (ICLR)*, 2022. 1

[17] Davis Rempe, Tolga Birdal, Aaron Hertzmann, Jimei Yang, Srinath Sridhar, and Leonidas J. Guibas. Humor: 3d human motion model for robust pose estimation. In *IEEE International Conference on Computer Vision (ICCV)*, 2021. 9

[18] Elad Richardson, Gal Metzer, Yuval Alaluf, Raja Giryes, and Daniel Cohen-Or. Texture: Text-guided texturing of 3d shapes. *ACM Transactions on Graphics (SIGGRAPH)*, 2023. 3, 4

[19] Tianchang Shen, Jun Gao, Kangxue Yin, Ming-Yu Liu, and Sanja Fidler. Deep marching tetrahedra: a hybrid representation for high-resolution 3d shape synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 4

[20] Kim Sung-Bin, Lee Hyun, Da Hye Hong, Suekyeong Nam, Janghoon Ju, and Tae-Hyun Oh. Laughtalk: Expressive 3d talking head generation with laughter, 2023. 9

[21] Guy Tevet, Sigal Raab, Brian Gordon, Yoni Shafir, Daniel Cohen-or, and Amit Haim Bermano. Human motion diffusion model. In *International Conference on Learning Representations (ICLR)*, 2023. 9

[22] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1, 3

[23] Xianghui Xie, Bharat Lal Bhatnagar, and Gerard Pons-Moll. Visibility aware human-object interaction tracking from single rgb camera. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 9

[24] Yuxuan Xue, Bharat Lal Bhatnagar, Riccardo Marin, Nikolaos Sarafianos, Yuanlu Xu, Gerard Pons-Moll, and Tony Tung. Nsf: Neural surface fields for human modeling from monocular depth. In *IEEE International Conference on Computer Vision (ICCV)*, 2023. 9

[25] Kim Youwang, Kim Ji-Yeon, Kyungdon Joo, and Tae-Hyun Oh. Unified 3d mesh recovery of humans and animals by learning animal exercise. In *British Machine Vision Conference (BMVC)*, 2021. 9

[26] Kim Youwang, Kim Ji-Yeon, and Tae-Hyun Oh. CLIP-Actor: Text-driven recommendation and stylization for animating human meshes. In *European Conference on Computer Vision (ECCV)*, 2022. 9

[27] Kim Youwang, Lee Hyun, Kim Sung-Bin, Suekyeong Nam, Janghoon Ju, and Tae-Hyun Oh. A large-scale 3D face mesh video dataset via neural re-parameterized optimization, 2023. 9

[28] Wojciech Zielonka, Timo Bolkart, and Justus Thies. Towards metrical reconstruction of human faces. In *European Conference on Computer Vision (ECCV)*, 2022. 9