TexVocab: Texture Vocabulary-conditioned Human Avatars

Supplementary Material

A. OverView

In Sec. B, we display more results of the synthesized avatar and geometric avatar. In Sec. C and Sec. D, we present more implementation details and how we conduct experiments, respectively.

B. More Results

B.1. Animation Results

Fig. A4 shows more animation results of different datasets. Fig. A5 displays animation results of different method on AIST++ [5] dataset. The comparisons against results from PoseVocab and results from NeRF MLP with pose further demonstrate the generalization of our method. Fig. A6 displays more anination results of different encoding methods(global-pose embedding, joint-structured embedding, body-part-wise embedding) on AMASS Dataset [11]. For more animation results and comparisons against other methods such as ARAH [15], TAVA [6] and PoseVocab [7], please refer to the supplementary videos.

B.2. Geometric Results

Benefiting from VolSDF [17], our method can also produce detailed geometric results as shown in Fig A1. The geometry results can also serve as priors to sample points near the surface in rendering to accelerate the training procedure.

C. Implementation Details

C.1. Network Architecture

In our method, we use a CNN as the image encoder to extract features of the texture images, and we use a NeRF MLP to decode the 3D character. The architecture of the NeRF MLP is shown in Fig. A2. For the image encoder, following PixelNeRF [18], we use the ResNet34 [3] backbone pre-trained on ImageNet for our experiments. Features are extracted prior to the first 3 pooling layers, upsampled using bilinear interpolation, and concatenated to form latent vectors of size 256 aligned to each pixel. Actually, changing the backbone of the network does not make much difference to the rendering results.

The number of nearest key body parts in KNN is $K = 4$, the number of key body parts equals to $0.3 \times T_1$, which can cover most of the appearances of training poses. And the resolution of the obtained texture images is 512×512. For the amount of training frames T_1, please refer to Sec. D.1.

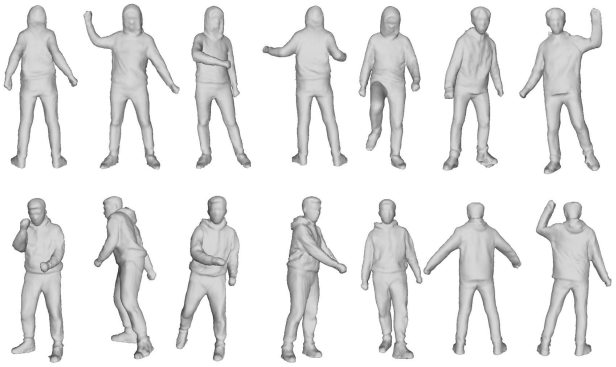

Figure A1. Geometric results of our method.

C.2. Comparisons

For animation results of ARAH [15], TAVA [6], AniNeRF [13] and PoseVocab [7], we use the code released by authors. For results of NeuralActor [9], we borrow the results reported in the technical paper and presented in the video.

D. Experimental Details

D.1. Datasets

In our experiments, we use 6 sequences of multi-view videos from THUman4.0 dataset [20], ZJU-MoCap dataset [14] and DeepCap dataset [2] for training and evaluation. We also use AMASS dataset [11] and AIST++ dataset [5] for novel pose syhthesis.

THUman4.0 Dataset. We use sequence 'subject00', 'subject01' and 'subject02' in THUman4.0 Dataset [20], which are all captured from 24 cameras, and contain 2500, 5060 and 3110 frames respectively. Each frame also provides SMPL-X [12] registration. We use the first 2000 frames of all the cameras for training, and the rest are for testing.

ZJU-MoCap Dataset. We use sequence 'CoreView_313', 'CoreView_394' in ZJU-MoCap Dataset [14]. . 'CoreView_313' is captured by 21 cameras, while 'CoreView_394' is captured by 23 cameras. Each frame provides SMPL [10] registration. We both use the former 500 frames of all the cameras for training.

DeepCap Dataset. We use sequence 'lan' in DeepCap Dataset [2], which is captured from 11 cameras and contains around 33600 training frames and around 14200 testing frames. Each frame provides SMPL-X [12] registration. We use all the cameras and sample every 10 frames in the

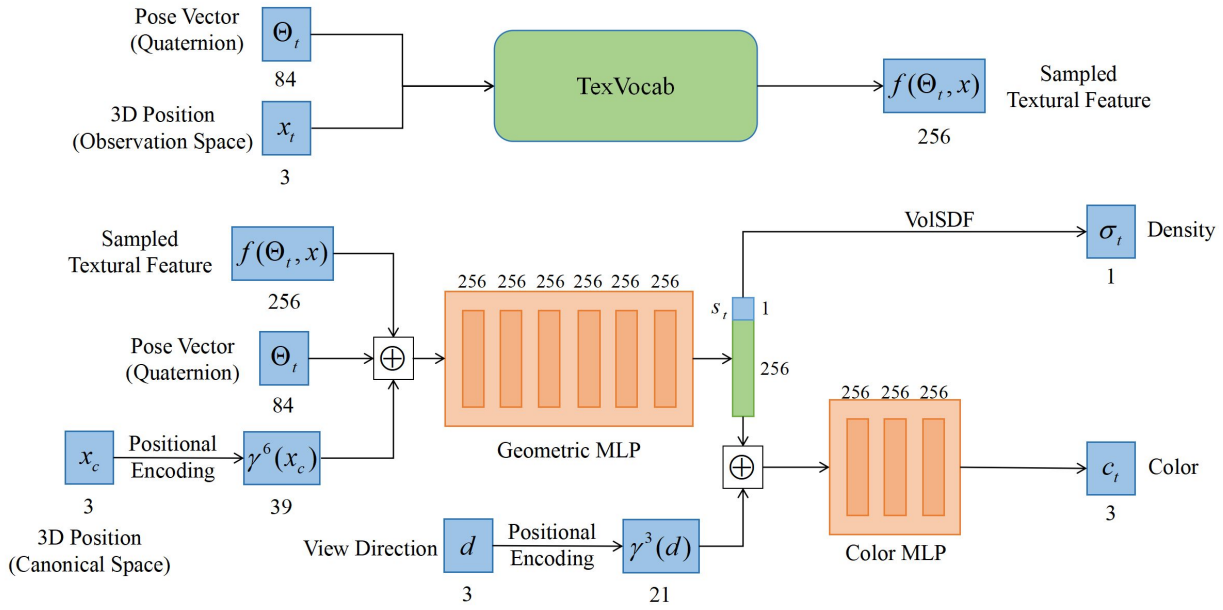

Figure A2. Architecture of our network.

training set for training, i.e., the training dataset contains about 3360 poses.

Testing Dataset. We use several sequences from AMASS [11] and AIST++ [5] dataset for novel pose synthesis to evaluate the generalization of our method.

D.2. Training Loss

The training losses include a color loss, a perceptual loss, a mask loss and the Eikonal loss.

Color Loss. Color loss measures the MSE loss between the rendered image and ground-truth pixel colors:

$$\mathcal{L}_{color} = \sum\nolimits_{r\in\mathcal{R}} \|\mathcal{C}(r) - \mathcal{C}^*(r)\|_2^2 \tag{1}$$

where$\mathcal{R}$ is the set of sampled rays from the rendered view, and $\mathcal{C}(r)$ and $\mathcal{C}^*(r)$ are the rendered and true pixel colors, respectively.

Mask Loss. Mask loss measures the MAE loss between the mask of rendered image and ground-truth:

$$\mathcal{L}_{mask} = \sum\nolimits_{r\in\mathcal{R}} \|\mathcal{M}(r) - \mathcal{M}^*(r)\|_2^2 \tag{2}$$

where $\mathcal{M}(r)$ and $\mathcal{M}^*(r)$ are the rendered and ground-truth mask values, respectively. The mask loss supervises the occupancy values of the rendered pixel.

Eikonal Loss. Eikonal loss [1] encourages the geometry fields to approximate a true signed distance function [17]:

$$\mathcal{L}_{eikonal} = \mathbb{E}(\|\nabla_x s(x, f(\Theta, x))\|_2^2 - 1) \tag{3}$$

where $s(\cdot)$ is the MLP-based function that maps a 3D position x_c and pose feature $f(\Theta, x_c)$ to the SDF value.

Perceptual Loss. The perceptual loss $\mathcal{L}_{perceptual}$ is widely used in NeRF training [7, 20, 21], which leads to better recovery of high-frequency details like the clothed wrinkles and thin lines [19]. We choose VGGNet as the backbone to calculate the learned perceptual image patch similarity (LPIPS):

$$\mathcal{L}_{color} = \sum\nolimits_{p\in\mathcal{P}} \|VGG(\mathcal{C}(r)) - VGG(\mathcal{C}^*(r))\|_2^2 \tag{4}$$

where $\mathcal{P}$ is the set of sampled patches from the rendered view, and $\mathcal{C}(r)$ and $\mathcal{C}^*(r)$ are the rendered and true patch colors, respectively.

Finally, we calculate the weight sum to optimize the network:

$$\begin{aligned}\mathcal{L} = \lambda_{color}\mathcal{L}_{color} + \lambda_{mask}\mathcal{L}_{mask} \\ + \lambda_{perceptual}\mathcal{L}_{perceptual} + \lambda_{eikonal}\mathcal{L}_{eikonal}\end{aligned} \tag{5}$$

where the λs are loss weights.

D.3. Training Details

We train the network using the Adam [4] optimizer with a batch size of 4 for 40 epochs. The initial learning rate is 5×10^{-4} and decays by multiplying 0.9 every 20k iterations. The training procedure takes about $1 \sim 2.5$ days on 4 RTX 3090 cards varying different multi-view video sequences. The training procedure contains three stages. In the first stage, we set $\lambda_{color} = 1$, $\lambda_{mask} = 1$, $\lambda_{eikonal} = 0.1$ and $\lambda_{perceptual} = 0$ during the first 6 epochs. We randomly sample 1024 rays on the training views, 80% of which are inside the mask image. For each ray, we sample 64 points within the SMPL bounding box. Under the supervision of the mask and color losses, a plausible geometry for each training frame can be learned. Then we extract 3D meshes using Marching Cubes [16] for all the training frames by querying the SDF value of each voxel in a coarse 3D volume that contains the posed human, and then render depth maps of all the training frames. In the following training procedure, we use depth-guided sampling by only sampling 32 points near the surface based on the rendered depth map. The depth-guided sampling strategy not only accelerates the training process, but also encourages the network to focus on modeling the dynamic appearance of valid regions. In the second stage, from the 6th to the 20th epoch, we set Eikonal loss to 0 for faster training. In the third stage, we sample patches with a resolution of 64×64 on the training views and enable the perceptual loss with the perceptual set to 0.1 until the end of the training procedure.

D.4. Animation Details

We utilize depth-guided sampling benefiting from the results of geometric avatars. Given a novel pose, we first obtain the SMPL model and allocate a sparse volume with a resolution of $128 \times 128 \times 128$ that contains the posed SMPL. Then, we predict SDF values of voxels near the SMPL surface to extract the geometric avatar using Marching Cubes [16]. With the explicit geometric result, we can render it to the camera view to obtain a depth map. In the following volume rendering, we can use depth-guided sampling, which means we only need to evaluate the colors of pixels inside the body mask on the depth map. However, the depth maps may be inaccurate on boundaries of self-occluded regions, producing background colors. For these pixels, we re-sample points within the balls generated by SMPL vertices similar to [9]. Such a strategy can remarkably improve the rendering speed and it can take around 6 seconds to render the human appearance at a resolution of 1150×1330.

For sequential animation, e.g., the animation results shown in the supplementary video, we use a sliding window of length 5 to consider embeddings of adjacent frames in order to guarantee the temporal consistency of the results.

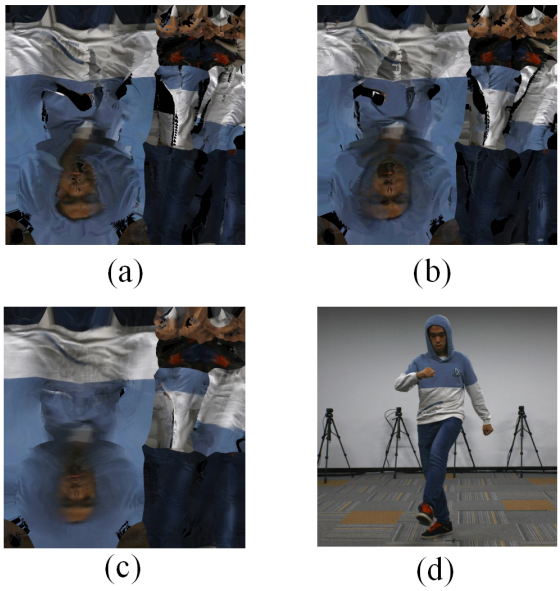

Figure A3. Image (d) and corresponding texture maps using 3 views(a), 4 views(b) and dense views(c).

E. Limitations and Future Work

In this section, we further discuss the limitations of our method and the future work.

Temporal robustness. Actually, our method loses some temporal consistency compared with PoseVocab [7]. This is because the KNN neighbors in the query cannot be guaranteed to be temporally continuous. Our method and PoseVocab both suffer from this, but the values of feature lines of PoseVocab can be optimized to lie in a small range, thus producing more stable temporal animation. In contrast, the texture features in our method are of much higher frequency, yielding occasional temporal flickers sometimes.

Sparse Views. The back-projection in Sec.4.2 relies on dense views of multi-view videos. As shown in Fig. A3, the texture maps might be incomplete using 3 or 4 views because of the occlusion. For instance, when the arm is placed in front of the body, part of the body may be occluded.

Body Template and Loose Clothes. When we back-project the image evidence, we use a naked SMPL instead of a person-specific template. Since there are differences between SMPL and clothed humans, as shown in Fig. A3(c), the cloth pattern (specifically, the "D" letter on the hoodie) may be distorted after back-projecting. Also, the usage of SMPL UV parameterization in back-projection constrains the character to wear tight clothes, our approach cannot handle the loose clothes like long dresses. The problem may be dealt with a person-specific parametric template like [8] instead of a sole SMPL mesh. And we leave it for future work.

Figure A4. Animation examples on 6 sequences of multi-view videos.

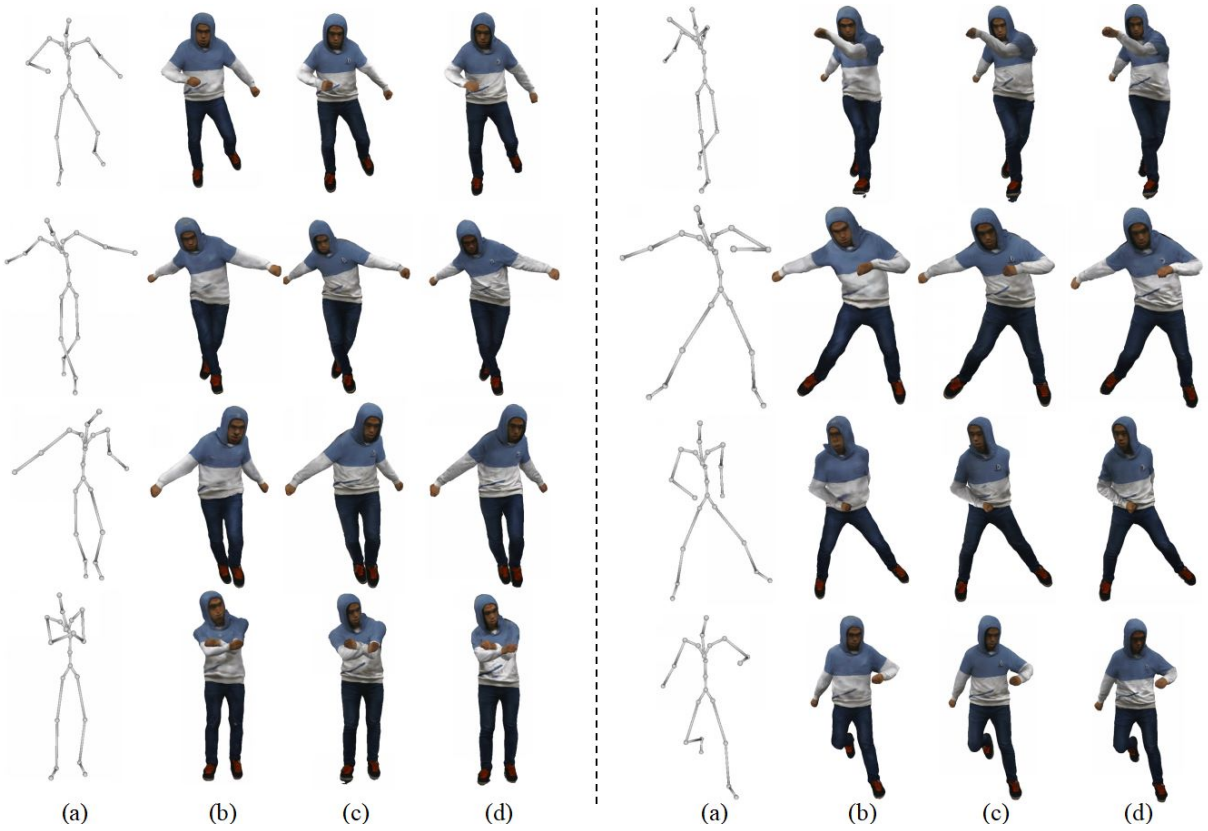

Figure A5. Animation results on AIST++ [5] dataset. We compare our animation results against results produced by NeRF with Pose, and PoseVocab.(a) Driving poses, (b) NeRF MLP with pose, (c) Posevocab, (d) ours.

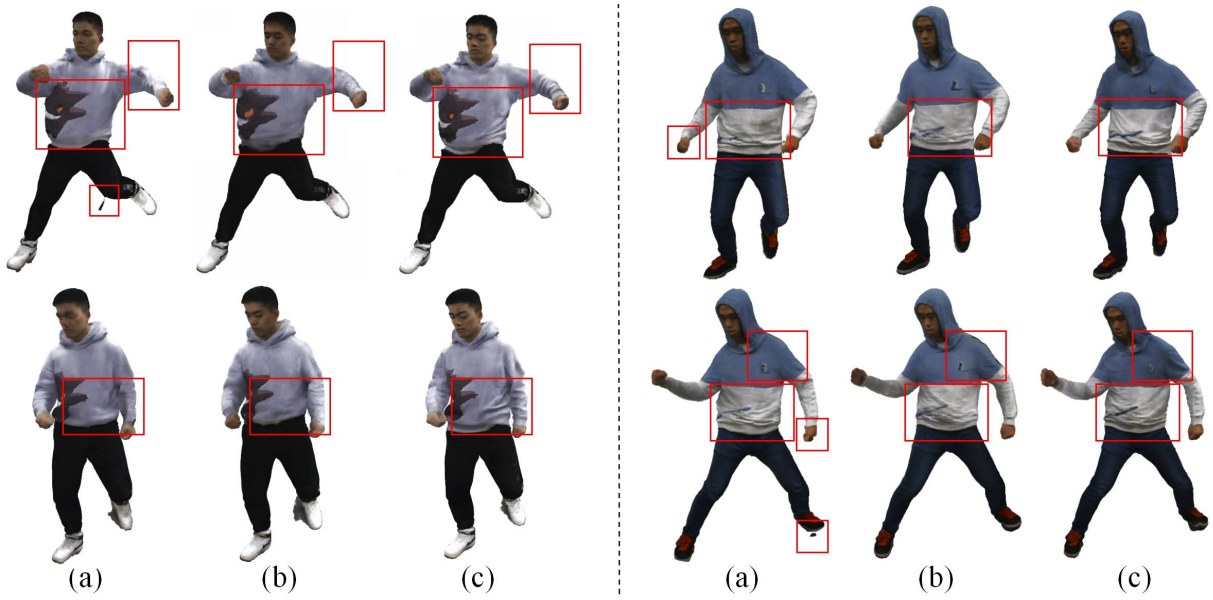

Figure A6. Animation results on AMASS [5] dataset. We compare our animation results against different encoding strategy. We show synthesized images of (a) global pose, (b) joint-structured, and (c) body-part-wise embedding.

References

[1] Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. Implicit geometric regularization for learning shapes. *arXiv preprint arXiv:2002.10099*, 2020. 2

[2] Marc Habermann, Weipeng Xu, Michael Zollhofer, Gerard Pons-Moll, and Christian Theobalt. Deepcap: Monocular human performance capture using weak supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5052–5063, 2020. 1

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1

[4] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 3

[5] Ruilong Li, Shan Yang, David A Ross, and Angjoo Kanazawa. Ai choreographer: Music conditioned 3d dance generation with aist++. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13401–13412, 2021. 1, 2, 5

[6] Ruilong Li, Julian Tanke, Minh Vo, Michael Zollhöfer, Jürgen Gall, Angjoo Kanazawa, and Christoph Lassner. Tava: Template-free animatable volumetric actors. In *European Conference on Computer Vision*, pages 419–436. Springer, 2022. 1

[7] Zhe Li, Zerong Zheng, Yuxiao Liu, Boyao Zhou, and Yebin Liu. Posevocab: Learning joint-structured pose embeddings for human avatar modeling. *arXiv preprint arXiv:2304.13006*, 2023. 1, 2, 3

[8] Zhe Li, Zerong Zheng, Lizhen Wang, and Yebin Liu. Animatable gaussians: Learning pose-dependent gaussian maps for high-fidelity human avatar modeling. *arXiv preprint arXiv:2311.16096*, 2023. 3

[9] Lingjie Liu, Marc Habermann, Viktor Rudnev, Kripasindhu Sarkar, Jiatao Gu, and Christian Theobalt. Neural actor: Neural free-view synthesis of human actors with pose control. *ACM transactions on graphics (TOG)*, 40(6):1–16, 2021. 1, 3

[10] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. Smpl: A skinned multi-person linear model. *Acm Transactions on Graphics*, 34 (6cd):248, 2015. 1

[11] Naureen Mahmood, Nima Ghorbani, Nikolaus F Troje, Gerard Pons-Moll, and Michael J Black. Amass: Archive of motion capture as surface shapes. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5442–5451, 2019. 1, 2

[12] Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed AA Osman, Dimitrios Tzionas, and Michael J Black. Expressive body capture: 3d hands, face, and body from a single image. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10975–10985, 2019. 1

[13] Sida Peng, Junting Dong, Qianqian Wang, Shangzhan Zhang, Qing Shuai, Xiaowei Zhou, and Hujun Bao. Animatable neural radiance fields for modeling dynamic human bodies. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14314–14323, 2021. 1

[14] Sida Peng, Yuanqing Zhang, Yinghao Xu, Qianqian Wang, Qing Shuai, Hujun Bao, and Xiaowei Zhou. Neural body: Implicit neural representations with structured latent codes for novel view synthesis of dynamic humans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9054–9063, 2021. 1

[15] Shaofei Wang, Katja Schwarz, Andreas Geiger, and Siyu Tang. Arah: Animatable volume rendering of articulated human sdfs. In *European conference on computer vision*, pages 1–19. Springer, 2022. 1

[16] LorenSen We. Marching cubes: A high resolution 3d surface construction algorithm. *Comput Graph*, 21:163–169, 1987. 3

[17] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. *Advances in Neural Information Processing Systems*, 34:4805–4815, 2021. 1, 2

[18] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4578–4587, 2021. 1

[19] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018. 2

[20] Zerong Zheng, Han Huang, Tao Yu, Hongwen Zhang, Yandong Guo, and Yebin Liu. Structured local radiance fields for human avatar modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15893–15903, 2022. 1, 2

[21] Zerong Zheng, Xiaochen Zhao, Hongwen Zhang, Boning Liu, and Yebin Liu. Avatarrex: Real-time expressive full-body avatars. *arXiv preprint arXiv:2305.04789*, 2023. 2