HAHA: Highly Articulated Gaussian Human Avatars with Textured Mesh Prior

David Svitov[1,2], Pietro Morerio[2], Lourdes Agapito[3], and Alessio Del Bue[2]

[1] Università degli Studi di Genova, Italy
[2] Istituto Italiano di Tecnologia (IIT) Genoa, Italy
{david.svitov, pietro.morerio, alessio.delbue}@iit.it
[3] Department of Computer Science, University College London
l.agapito@cs.ucl.ac.uk

Abstract. We present HAHA - a novel approach for animatable human avatar generation from monocular input videos. The proposed method relies on learning the trade-off between the use of Gaussian splatting and a textured mesh for efficient and high fidelity rendering. We demonstrate its efficiency to animate and render full-body human avatars controlled via the SMPL-X parametric model. Our model learns to apply Gaussian splatting only in areas of the SMPL-X mesh where it is necessary, like hair and out-of-mesh clothing. This results in a minimal number of Gaussians being used to represent the full avatar and reduced rendering artifacts. This allows us to handle the animation of small body parts, such as fingers, that are traditionally disregarded. We demonstrate the effectiveness of our approach on two open datasets: SnapshotPeople and X-Humans. Our method demonstrates on par reconstruction quality to the state-of-the-art on SnapshotPeople, while using less than a third of Gaussians. HAHA outperforms previous state-of-the-art on novel poses from X-Humans both quantitatively and qualitatively.

Keywords: Human avatar · Full-body · Gaussian splatting · Textures

1 Introduction

The task of creating photo-realistic animated objects has always been of paramount importance in 3D computer vision. High-fidelity animated objects are widely used in real-time applications, ranging from computer games to online telepresence systems [3, 12]. In recent years the interest in the field has increased due to the emergence of devices for virtual [1] and augmented [2] reality. Traditionally, the central aspect of the task is the creation of a human avatar as it has a wide range of uses and digital replicas are essential for online human-to-human interaction. Therefore, our work concentrates on rendering animated photo-realistic human avatars.

In this work, we introduce *Highly Articulated Gaussian Human Avatars with Textured Mesh Prior (HAHA)*. While existing approaches focus on using the

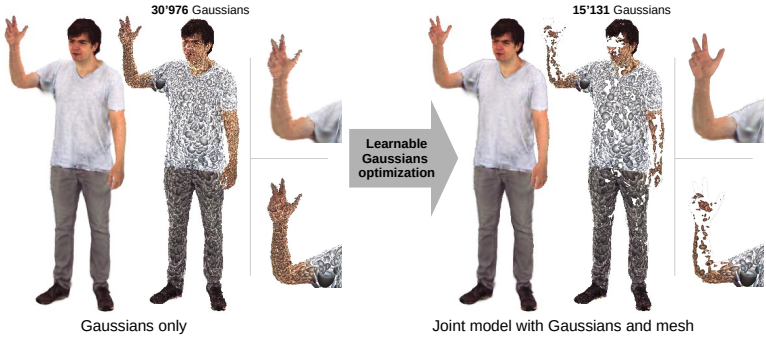

Fig. 1: Optimizing the number of Gaussians. HAHA jointly optimizes a Gaussian splatting model with a textured mesh to improve the photometric quality of the avatars. The method filters out superfluous Gaussians in a learnable, unsupervised manner. As a result, we can more efficiently and better animate highly articulated parts of a body.

mesh-based approach [27] or Gaussian-based approach [19], we target to take the best from both representations. Our main idea is to **learn to use the appropriate number of Gaussians relying on a textured mesh where possible** (Fig. 1). We attach Gaussians to the mesh surface only at the points where it is necessary to represent out-of-mesh details. For the mesh, we use SMPL-X [20] parametric human model with articulated fingers and face, and in contrast with previous approaches that use SMPL [17] we aim to control fingers animation as well as the bigger joints. Areas not covered with the Gaussians are represented as a textured mesh surface that is more efficient to store. Using such a mesh, we significantly reduce the number of Gaussians in the areas of the hands and face (Fig. 1). Overall, we reduce the amount of Gaussians **up to three times** for the whole avatar, resulting in $\times 2.3$ reduced storage costs.

We obtain an avatar with a three-stage pipeline. During the first two stages, we learn Gaussian and textured mesh representation of the avatar. In the final stage, we estimate which Gaussians to remove in an unsupervised manner. We proposed the mechanism for the combined differentiable rendering of Gaussians and a mesh, which allows us to adjust Gaussians' parameters based on the final rendering of the avatar.

We propose several regularization techniques to encourage *HAHA* to remove as many Gaussians as possible without affecting the quality of the avatar. Following 3DGS [14] our Gaussians have trainable opacity and we delete them when it is lower than a threshold. We use two regularizations balancing each other to control Gaussians' opacity during training. While the first pushes opacity down, the second controls out-of-mesh detail preservation. This way, we find a learnable trade-off in using Gaussians and a textured mesh. To train *HAHA* in such a manner, we only need input video frames with the provided SMPL-X fits without any additional labels.

In our experiments, we show that *HAHA* reaches quantitative metrics on par with state-of-the-art methods [11,16,22] on the open SnapshotPeople dataset [6], while better generalizing to novel poses and views. Using videos from the X-Humans dataset [23], we demonstrated that *HAHA* allows us to animate fingers with higher quality than state-of-the-art. We demonstrate that our method, both qualitatively and quantitatively, outperforms state-of-the-art methods on agile X-Humans data, while at the same time, it allows us to reduce the number of Gaussians.

The main contributions of the work are the following:

- We first propose the use of Gaussians in combination with textured mesh to increase the efficiency of rendering human avatars;
- We develop an unsupervised method for significantly reducing the amount of Gaussians in the scene through the use of textured mesh;
- We demonstrate that our method can efficiently handle the animation of hands and other highly articulated parts without the need for any additional engineering.

2 Method

Our pipeline comprises three stages. In the first (Fig. 2 (a)) stage, we learn a full-Gaussian representation of the avatar and fine-tune SMPL-X's poses and shapes for training frames. As a result, we get an avatar represented with Gaussians as in previous state-of-the-art approaches having a fixed initial set of Gaussians (*i.e.* $N = 20908$). In the second stage (Fig. 2 (b)), we use resulting SMPL-X meshes with the provided UV-map to learn RGB texture. Thus we obtain textured avatars without any out-of-mesh details but efficient to render and store. In the last stage, we merge these two avatars and learn to remove some Gaussians without losing quality (Fig. 2 (c)). To figure out which Gaussians to delete we perform combined rendering of the avatar and fine-tune Gaussians opacity. Further in this section, we describe these three stages in more detail.

2.1 Gaussian Avatar Preliminaries

First, we describe how we set Gaussians on the SMPL-X mesh surface. For each mesh's polygon, we calculate the coordinates of its center T_i, the quaternion rotation R_i, and the scale k_i (Fig. 2 (a)). Then we calculate the parameters of the N Gaussians $\Delta_i = \{\mu_i, r_i, s_i, c_i, o_i\}$ attached to each SMPL-X's polygon referred as i. Here μ_i, r_i, s_i are the Gaussian's translation, rotation, and scale offsets relative to i-th polygon parameters $\{T_i, R_i, k_i\}$, while c_i and o_i are the color and opacity properties, respectively. Similar to [21] we perform a subdivision of Gaussians while maintaining the attachment to the parent polygon: $\Delta_i = \{\mu_i^j, r_i^j, s_i^j, c_i^j, o_i^j\}_{j=0}^{M_i}$ (Fig. 2 (a)). Thus, the final Gaussians pose and shape parameters are calculated as offsets to the corresponding i-th polygon parameters $\{T_i, R_i, k_i\}$ as follows:

$$r' = Rr \qquad \mu' = kR\mu + T \qquad s' = ks. \tag{1}$$

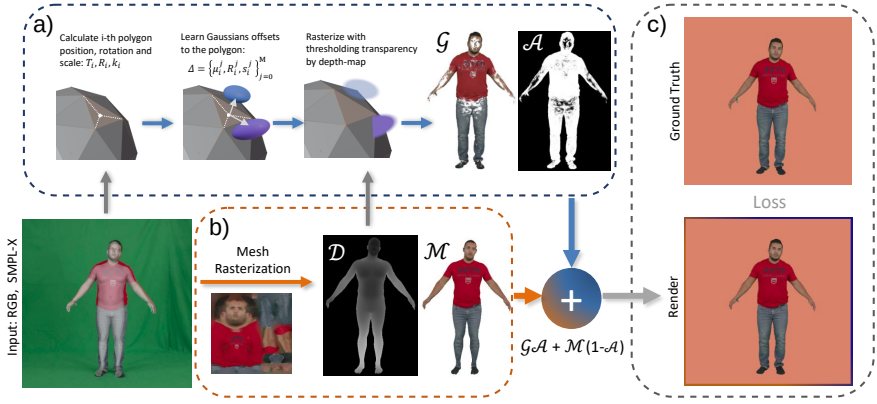

Fig. 2: Scheme of our approach. a) We attach Gaussians to mesh polygons as described in Section 2.1 and rasterize them conditioned on depth map $\mathcal{D}$ into RGB image $\mathcal{G}$ and alpha map $\mathcal{A}$. b) We train RGB texture for SMPL-X and rasterize mesh to RGB image $\mathcal{M}$ and depth map $\mathcal{D}$. c) During training and inference we merge rasterizations of Gaussians $\mathcal{G}$ and mesh $\mathcal{M}$, based on the trainable transparency map $\mathcal{A}$ of Gaussians.

2.2 Monocular Avatar Training

First stage: Gaussian avatar training. In the first stage (Fig. 2 (a)), we train the 3DGS representation of an avatar by optimizing only local Gaussians transformations μ_i^j, r_i^j, s_i^j and color c_i^j. Opacity o_i^j is fixed to 1 during this stage as we keep all Gaussians untransparent to efficiently back-propagate image space losses to the SMPL-X parameters. Thus, we force the model to optimize the pose and shape of the underlying mesh rather than deleting Gaussians. We use randomly colored backgrounds in this stage to prevent Gaussians from learning background color.

To optimize Gaussians we use several image space losses as L_2 loss, L_{LPIPS} perceptual loss [25], L_{SSIM} structure similarity loss, and L_{Sobel} loss to get sharper edges. To calculate L_{Sobel} loss we measure L_2 between results of applying the Sobel operator [13] to rendered and ground truth images. In other words, we calculate the distance between discrete derivatives of images to account for high-frequency details. We follow [16] and apply L_{KNN}, a KNN-based regularization to get smoother results with fewer artifacts. In KNN-regularization we minimize the standard deviation of properties of neighboring Gaussians. The final loss is as follows:

$$\mathcal{L}_{\text{Gaussian}} = L_2 + \lambda_{LPIPS} L_{\text{LPIPS}} + \lambda_{SSIM} L_{\text{SSIM}} + \lambda_{Sobel} L_{\text{Sobel}} + \lambda_{KNN} L_{\text{KNN}}. \quad (2)$$

Second stage: RGB texture training. In the second stage we render an avatar as rasterized SMPL-X mesh with a texture (Fig. 2 (b)). We disable 3DGS and rasterize SMPL-X mesh with trainable texture using Nvdiffrast [15]. The use of the differentiable rasterizer lets us back-propagate to the avatar's parameters. We optimize only the texture keeping SMPL-X's parameters frozen during the

whole stage. Similar to classic avatar approaches [4–6] we utilize three-channeled RGB texture.

Following [7], we utilize TV-regularization (L_{TV}) [8] to produce smoother results. But we apply L_{TV} in the texture space instead of the image space as we aim to reduce texture artifacts. The final loss for this stage is as follows:

$$\mathcal{L}_{\text{texture}} = L_2 + \lambda_{LPIPS} L_{\text{LPIPS}} + \lambda_{SSIM} L_{\text{SSIM}} + \lambda_{TV} L_{\text{TV}}. \tag{3}$$

Third stage: Filtering out Gaussians. Textured mesh from the previous stage can replace close-to-surface Gaussians on the avatar (*e.g.* hands and face). Therefore, we can learn which Gaussians to remove in an unsupervised manner and reduce rendering and storage costs. To achieve this, we merge the differentiable rendering of the textured mesh and the differentiable 3DGS process.

In Figure 2 (c), we render the merged SMPL-X mesh-based and Gaussian avatar (Section 2.3) and train Gaussians opacity o_i^j and color c_i^j. We delete all Gaussians with transparency lower than a threshold (0.1). We use two regularizations to encourage optimization to find a trade-off between Gaussians amount and image quality. One reduces the transparency of Gaussians to remove as much of them as possible, while the second preserves Gaussians with a segmentation loss. Using both of them allows us to remove only unnecessary Gaussians.

The transparency regularization pushes opacity o_i of Gaussians down as follows:

$$L_{\text{opacity}} = \sum_{i=0}^{N} \sum_{j=0}^{M_i} \|o_i^j\|_2^2. \tag{4}$$

Optimising only this loss would aggressively remove several Gaussians, and for this reason, we add a "counterweight". We propose to use silhouette Dice loss (L_{dice}) [18] to encourage the training to preserve out-of-mesh details. As ground truth, we use human silhouettes S_{GT} that can be predicted by from-the-shelf segmentation models [10, 24]. We summarize alpha map $\mathcal{A}$ and binarized depth map $bin(\mathcal{D})$ to generate silhouette masks for L_{dice}. With these terms, the loss for the third stage is the following:

$$\mathcal{L}_{\text{filtering}} = \mathcal{L}_{\text{Gaussian}} + \lambda_{opacity} L_{\text{opacity}} + \lambda_{dice} L_{\text{dice}}(S_{\text{GT}}, bin(\mathcal{D}) || \mathcal{A}). \tag{5}$$

As a result of such training, we remove only Gaussians that could be replaced with the underlying mesh.

2.3 Merging Gaussians with Mesh Representation

Here we describe how to simultaneously render 3DGS and textured mesh in a differentiable way. When rendering the textured mesh in Figure 2 (b), we calculate its depth map $\mathcal{D}$ as the distance from the camera. We use this depth map as additional input to our modified 3DGS rasterizer $G_{2D}(\mathcal{D}, K, M, \{r', \mu', s', c, o\})$,

that also accepts camera intrinsic K and extrinsic M matrices and optimized Gaussians parameters.

During rasterization, we calculate the distance D_i from the camera to each point of i-th Gaussian in the scene. The corresponding value of D_i can be addressed via its screen space coordinates $[x, y]$. Our modification of splatting takes into account the distance D_i in each pixel and compares it to the depth map $\mathcal{D}$ *i.e.* we check if the Gaussians are under the mesh or behind it. We set Gaussian's transparency at each pixel to zero if the distance to Gaussian at this point is more than the depth map value:

$$\alpha_i'[x, y] = \begin{cases} 0 & , if D_i[x, y] > \mathcal{D}[x, y] \\ \alpha_i[x, y] & , else \end{cases}, \tag{6}$$

where $\alpha_i[x, y]$ initially calculates based on the opacity o_i^j and the Gaussian attenuation. We also store the final Gaussians transparency map for each pixel to the alpha map $\mathcal{A}$. To do this, we accumulate transparency at each $[x, y]$ pixel during 3DGS rasterization [14]:

$$\mathcal{A}[x, y] = 1 - \prod_{i=0}^{N[x,y]} (1 - \alpha_i'[x, y]). \tag{7}$$

We then use alpha map $\mathcal{A}$ to mix rasterization $\mathcal{M}$ of the textured mesh with Gaussians rasterization $\mathcal{G}$ to get final avatar. To obtain the final rasterization, we mix them as shown in Figure 2:

$$\mathcal{I} = \mathcal{G}\mathcal{A} + \mathcal{M}(1 - \mathcal{A}). \tag{8}$$

3 Experiments

In our experiments, we compared HAHA to the state-of-the-art Gaussian methods, namely: GART [16], 3DGS-Avatar [22], and GaussianAvatar [11]. All these methods represent the human body as a set of Gaussians. We used two open datasets to evaluate our approach: X-Humans [23] and SnapshotPeople [6]. From both datasets, we used monocular RGB videos as input to our method. In the following section, we show both qualitative and quantitative results.

3.1 X-Humans

We report the following metrics: PSNR, SSIM, and LPIPS [26] (Table 1). PSNR and SSIM measure the fidelity of the signal and structural similarity, respectively, while LPIPS correlates with human perception of the image using neural network features to compare with ground truth. We evaluated metrics on the renderings with a black background as in 3DGS-Avatar [22] experiments to set the background value to zero. During inference, we used *test time pose optimization* following GART [16] to reduce the impact of SMPL-X fitting inaccuracies.

Fig. 3: Comparison on X-Humans dataset. We provide results for three different poses and views to demonstrate hands animation. HAHA allows us to animate hands while we use much fewer Gaussians, and it is more robust to the input data while producing fewer artifacts. While GaussianAvatar [11] also benefits from using SMPL-X to animate hands, HAHA produces more realistic-looking results.

X-Humans [23] dataset provides a sequence of frames with rendered 3D scans of a person doing complex movements. The movements are diverse for both training and testing videos, therefore it is a challenging task to train on such a dataset. In Table 1 we compare our method with previous state-of-the-art methods: GART [16], 3DGS-Avatar [22], and GaussianAvatar [11]. The last one, similar to us, uses SMPL-X and can control fingers animation so we can compare the animation of hands. We provide metrics for both male and female avatars.

HAHA is more robust and gets better metrics for these complex training and testing sequences. Besides, our method requires fewer Gaussians. We also provide qualitative results in Figure 3 demonstrating overall quality and how our approach handles hands animation.

3.2 PeopleSnapshot

Following the previous literature, we also provide quantitative metrics for the SnapshotPeople [6] dataset (Table 2). However, SnapshotPeople does not allow assess quality for novel views and poses since train and test sequences are very similar-looking.

In all experiments for SnapshotPeople we used SMPL provided by AnimNerf [9]. As our method requires a parametric model to have articulated fingers, we converted the provided SMPL to SMPL-X using a converter from the SMPL official repository. Then we fine-tuned the resulting SMPL-X hand's pose and shape using SMPLify-X [20] to match ground truth frames.

SnapshotPeople evaluation methodology is challenging for our method because we strongly rely on the underlying mesh geometry. Therefore, in cases

Table 1: Quantitative metrics for X-Humans [23] dataset. The dataset lets one evaluate metrics values for novel poses.

	00016 (male)				00019 (female)			
	Gaussians↓	PSNR↑	SSIM↑	LPIPS↓	Gaussians↓	PSNR↑	SSIM↑	LPIPS↓
3DGS-Avatar [22]	42,77k	25.44	0.9315	0.0409	41,12k	27.63	0.9539	0.0471
GART [16]	55.85k	25.71	0.9295	0.0598	55.61k	27.78	0.9512	0.0668
GaussianAvatar [11]	191.58k	25.58	0.9328	0.0518	191.58k	27.54	0.9574	0.0647
HAHA(**Ours**)	15.13k	25.49	0.9339	0.0507	12.26k	28.49	0.9593	0.0501
	00018 (male)				00027 (female)			
	Gaussians↓	PSNR↑	SSIM↑	LPIPS↓	Gaussians↓	PSNR↑	SSIM↑	LPIPS↓
3DGS-Avatar [22]	26,78k	28.71	0.9521	0.0580	36,82k	26.84	0.9477	0.0445
GART [16]	50,47k	30.98	0.9595	0.0683	47,18k	26.56	0.9449	0.0595
GaussianAvatar [11]	191,58k	29.92	0.9588	0.0744	191,58k	25.69	0.9481	0.0543
HAHA(**Ours**)	18,57k	31.10	0.9630	0.0579	15,50k	27.26	0.9513	0.0473

Table 2: Quantitative metrics for SnapshotPeople [6] dataset. Our method gets metrics on par with state-of-the-art approaches while using much fewer Gaussians.

	female-3-casual				male-3-casual			
	Gaussians↓	PSNR↑	SSIM↑	LPIPS↓	Gaussians↓	PSNR↑	SSIM↑	LPIPS↓
3DGS-Avatar [22]	53.78k	30.57	0.9581	0.0208	37.22k	34.28	0.9724	0.0149
GART [16]	19.67k	32.73	0.9672	0.0459	21.88k	35.93	0.9767	0.0294
GaussianAvatar [11]	202.73k	25.94	0.9673	0.0434	202.73k	33.59	0.9697	0.0243
HAHA(**Ours**)	13.67k	32.53	0.9633	0.0403	13.60k	31.46	0.9619	0.0277

when train and test views and poses are similar, we could face metrics value reduction on the opposite to the methods where rendering does not strongly depend on the mesh surface. Nevertheless, we demonstrate metrics on par with state-of-the-art approaches for this dataset while using almost two times fewer Gaussians (Table 2).

4 Conclusion

We have presented a new method for modeling human avatars using joint representation with RGB textured mesh and Gaussian splatting. We use a textured SMPL-X parametric model to portray the avatar's areas near the human body surface while using Gaussians to render out-of-mesh details. Our methods allow us to significantly reduce the number of Guassians and memory required to store avatars. Using textured SMPL-X for body parts representation allows us to animate small details such as fingers. We demonstrated the efficiency of our approach both quantitatively and qualitatively on the open datasets. HAHA outperforms the previous state-of-the-art on challenging X-Humans dataset.

References

1. Expand your world with Meta Quest. https://www.meta.com/it/en/quest/, [Online; accessed 27-February-2024] 1
2. Introducing Apple Vision Pro: Apple's first spatial computer. https://www.apple.com/newsroom/2023/06/introducing-apple-vision-pro/, [Online; accessed 27-February-2024] 1
3. Mark zuckerberg: First interview in the metaverse. https://lexfridman.com/mark-zuckerberg-3/, online; accessed 27-February-2024 1
4. Alldieck, T., Magnor, M., Bhatnagar, B.L., Theobalt, C., Pons-Moll, G.: Learning to reconstruct people in clothing from a single rgb camera. In: CVPR. pp. 1175–1186 (2019) 5
5. Alldieck, T., Magnor, M., Xu, W., Theobalt, C., Pons-Moll, G.: Detailed human avatars from monocular video. In: International Conference on 3D Vision (3DV). pp. 98–109. IEEE (2018) 5
6. Alldieck, T., Magnor, M., Xu, W., Theobalt, C., Pons-Moll, G.: Video based reconstruction of 3d people models. In: CVPR. pp. 8387–8397 (Jun 2018). https://doi.org/10.1109/{CVPR}.2018.00875, CVPR Spotlight Paper 3, 5, 6, 7, 8
7. Bashirov, R., Larionov, A., Ustinova, E., Sidorenko, M., Svitov, D., Zakharkin, I., Lempitsky, V.: Morf: Mobile realistic fullbody avatars from a monocular video. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 3545–3555 (2024) 5
8. Chambolle, A.: An algorithm for total variation minimization and applications. Journal of Mathematical imaging and vision **20**, 89–97 (2004) 5
9. Chen, J., Zhang, Y., Kang, D., Zhe, X., Bao, L., Jia, X., Lu, H.: Animatable neural radiance fields from monocular rgb videos. arXiv preprint arXiv:2106.13629 (2021) 7
10. Gong, K., Gao, Y., Liang, X., Shen, X., Wang, M., Lin, L.: Graphonomy: Universal human parsing via graph transfer learning. In: CVPR (2019) 5
11. Hu, L., Zhang, H., Zhang, Y., Zhou, B., Liu, B., Zhang, S., Nie, L.: Gaussianavatar: Towards realistic human avatar modeling from a single video via animatable 3d gaussians. arXiv preprint arXiv:2312.02134 (2023) 3, 6, 7, 8
12. Jones, B., Zhang, Y., Wong, P.N., Rintel, S.: Belonging there: Vroom-ing into the uncanny valley of xr telepresence. Proceedings of the ACM on Human-Computer Interaction **5**(CSCW1), 1–31 (2021) 1
13. Kanopoulos, N., Vasanthavada, N., Baker, R.L.: Design of an image edge detection filter using the sobel operator. IEEE Journal of solid-state circuits **23**(2), 358–367 (1988) 4
14. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. ACM TOG **42**(4) (2023) 2, 6
15. Laine, S., Hellsten, J., Karras, T., Seol, Y., Lehtinen, J., Aila, T.: Modular primitives for high-performance differentiable rendering. ACM TOG **39**(6), 1–14 (2020) 4
16. Lei, J., Wang, Y., Pavlakos, G., Liu, L., Daniilidis, K.: Gart: Gaussian articulated template models. arXiv preprint arXiv:2311.16099 (2023) 3, 4, 6, 7, 8
17. Loper, M., Mahmood, N., Romero, J., Pons-Moll, G., Black, M.J.: Smpl: A skinned multi-person linear model. In: Seminal Graphics Papers: Pushing the Boundaries, Volume 2, pp. 851–866 (2023) 2
18. Milletari, F., Navab, N., Ahmadi, S.A.: V-net: Fully convolutional neural networks for volumetric medical image segmentation. In: International conference on 3D vision (3DV). pp. 565–571. Ieee (2016) 5

19. Moreau, A., Song, J., Dhamo, H., Shaw, R., Zhou, Y., Pérez-Pellitero, E.: Human gaussian splatting: Real-time rendering of animatable avatars. In: CVPR (2024) 2
20. Pavlakos, G., Choutas, V., Ghorbani, N., Bolkart, T., Osman, A.A., Tzionas, D., Black, M.J.: Expressive body capture: 3d hands, face, and body from a single image. In: CVPR. pp. 10975–10985 (2019) 2, 7
21. Qian, S., Kirschstein, T., Schoneveld, L., Davoli, D., Giebenhain, S., Nießner, M.: Gaussianavatars: Photorealistic head avatars with rigged 3d gaussians. arXiv preprint arXiv:2312.02069 (2023) 3
22. Qian, Z., Wang, S., Mihajlovic, M., Geiger, A., Tang, S.: 3dgs-avatar: Animatable avatars via deformable 3d gaussian splatting. arXiv preprint arXiv:2312.09228 (2023) 3, 6, 7, 8
23. Shen, K., Guo, C., Kaufmann, M., Zarate, J., Valentin, J., Song, J., Hilliges, O.: X-avatar: Expressive human avatars. CVPR (2023) 3, 6, 7, 8
24. Yang, L., Song, Q., Wang, Z., Hu, M., Liu, C., Xin, X., Jia, W., Xu, S.: Renovating parsing r-cnn for accurate multiple human parsing. In: ECCV. pp. 421–437. Springer (2020) 5
25. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: CVPR (2018) 4
26. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: CVPR. pp. 586–595 (2018) 6
27. Zheng, Z., Zhao, X., Zhang, H., Liu, B., Liu, Y.: Avatarrex: Real-time expressive full-body avatars. ACM TOG **42**, 1 – 19 (2023), https://api.semanticscholar.org/CorpusID:258557606 2