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, 29]. 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.

To date, several options are available to generate human avatars for what concerns input data. To get the best quality many methods rely on multi-view

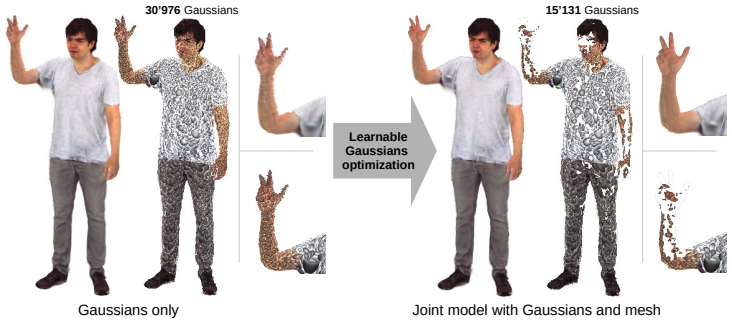

Fig. 1: Optimizing the number of Gaussians. HAHA jointly optimizes a Gaussian splatting model with a textured mesh to improve the photometric quality of the avatars. The method filters out superfluous Gaussians in a learnable, unsupervised manner. As a result, we can more efficiently and better animate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

data [9, 11, 18, 25, 65]. However, complex acquisition systems such as a multi-camera capture setup [23] or a 3D scanner [4] are required to collect such data. On the other hand, some methods [8, 19, 51, 55] use a single image of a person as input, which overcomplicates the task with the necessity to restore unobserved regions of the body. Eventually, the most convenient way is to generate avatars from **monocular videos**. Using a monocular video provides a trade-off between the complexity of obtaining input data and the quality of the avatar.

In the last few years, monocular video avatars have been represented using explicit [5–7] or implicit [11, 26, 47, 62] geometry. Recently, a novel method for representing 3D objects has appeared - **Gaussian splatting** (3DGS) [31] - where the scene is represented as a set of parametrized Gaussians, which are projected onto the screen surface during rendering. The most recent methods for human avatars [12, 13, 20, 21, 24, 38, 48, 49, 58, 68] indeed utilize 3DGS for rendering. These works cover an extensive range of tasks, from head avatars to multi-view full-body avatars. With this representation, temporal consistency is improved over implicit methods, and out-of-mesh details are more accurately conveyed than with traditional explicit methods. To drive the animation, previous works traditionally employ parametric models [39, 41, 46] of the human body. This way, they can control the shape and pose of the body via learnable parameters.

A common drawback of existing Gaussian-based methods is that they require **a large number of Gaussians** to represent a human avatar. Especially if we need to animate high-frequency details such as fingers. These regions of the body could require a tremendous amount of Gaussians to look realistic enough. Up to 200'000 for previous approaches [20] to represent an avatar. This in turn leads to an increase in the required memory. Moreover, if we need to improve the details of the resulting avatar, we can only increase the number of Gaussians. This could be a bottleneck when we want to render a scene with many avatars (*e.g.* for a game or a movie). Another issue with monocular video-based Gaussian avatars is that video frames from a single camera are often insufficient to generalize to

novel views and poses efficiently. Mesh-based explicit methods [5–7] circumvent this issue by strongly relying on mesh geometry, whereas Gaussians tend to overfit. However, these methods struggle to reconstruct loose clothes and hair accurately.

In this work, we introduce *Highly Articulated Gaussian Human Avatars with Textured Mesh Prior (HAHA)*. While existing approaches focus on using the mesh-based approach [67] or Gaussian-based approach [44], 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 [46] parametric human model with articulated fingers and face, and in contrast with previous approaches that use SMPL [41] 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 [31] 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 [20,38,49] on the open SnapshotPeople dataset [7], while better generalizing to novel poses and views. Using videos from the X-Humans dataset [53], 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 Related Work

Human parametric models. Parametric models such as SMPL [41] or FLAME [39] are widely used in human avatars [5, 6, 11, 14, 17, 26, 38, 49] to control pose and shape. The parametric model gets as input vectors of the pose and shape parameters and produces the mesh. Such a mesh is posed using linear blend skinning (LBS) when the pose vector controls pose-dependent body transformation. The resulting mesh may be used to transform an avatar to the canonical pose [11, 26] or to directly form an avatar appearance [5, 6].

Researchers traditionally use SMPL to get avatars, while the most flexible parametric model is SMPL-X [46]. This model allows one to additionally control finger joints and facial expressions. Therefore, it is more useful for practical use cases of avatars. *HAHA* uses as input SMPL-X's parameters corresponding to video frames. One can get SMPL-X's pose and shape parameters from input images using SMPLify-X [46] method or one of the recent feed-forward methods [34, 54].

Gaussian splatting avatars. 3DGS [31] appeared recently as a novel method for explicit scene representation. The method represents a scene as a collection of 3D Gaussians and their associated photometric information. These Gaussian splats on the camera image surface produce a rendered image during rendering. 3DGS demonstrated its efficiency for static scene representation [22,27,37,61] as well as for dynamic scenes [15,28,35,42]. Recent methods [12,13,20,21,24,38,48, 49,58,68] use 3DGS for rendering photo-realistic human avatar in different operational scenarios. They generate avatars based on the multi-view data [40,45,66] or a monocular-video [20, 24, 38, 49] input. Using 3DGS for avatar rendering allows authors to obtain temporally consistent animated rendering with better metrics value.

Current state-of-the-art methods use SMPL to drive animation in Gaussian-based human body rendering. For instance, GART [38] represents Gaussians in the canonical space and uses skeletons with learnable LBS weights to animate them. To handle out-of-mesh details, they proposed to create additional bones. 3DGS-Avatar [49] sets Gaussians in the canonical space and models non-rigid deformations with a learnable MLP network. The authors also applied as-isometric-as-possible regularization [32] to the Gaussians to preserve geometric consistency. GaussianAvatar [20] enforces inductive bias by using CNN to generate Pose Features in SMPL's texture space. GaussianAvatar optimizes this model and the SMPL pose to compensate for SMPL's inaccuracy. SplatArmor [24] embeds Gaussians to the SMPL surface in the canonical space. They use Neural Color Field to preserve inductive bias and use MLP to predict non-rigid transformation.

Another approach [45, 48, 52, 57, 59] is to attach Gaussians to the mesh's polygons. But so far, such methods focus on mostly rigid objects (*e.g.* heads) or use multi-view data as input. This work demonstrates the efficiency of such an approach for the monocular video-based full-body human avatars.

Several previous works [53, 67] solve the task of generating human avatars with articulated finders using multi-view data. AvatarReX [67] uses a separate parametric model to process hands. As input, they accept a multi-view video of a person. X-Avatar [53] uses a part-specific deformer network to handle the hands. As input X-Avatar gets 3D scans or RGB-D video with depth information. Both methods reconstruct an avatar as a textured mesh that in general leads to more blurred results than 3DGS. In contrast to the previous works, we use only monocular RGB data.

Texture-based avatars. The classical approaches generate video-based human avatars using textured meshes [5–7]. A mesh with RGB texture allows faster rendering with minimal artifacts, but the drawback of such an approach is the lack of out-of-mesh details. Existing methods try to circumvent this issue by predicting offsets to the SMPL mesh vertices. However, such an approach is limited by mesh topology as we can not represent enough details where the mesh grid is sparse.

To improve the textured mesh approach, researchers proposed the neural-texture rendering technique Deferred Neural Rendering (DNR) [56]. In this approach, textures contain an arbitrary number of channels and can be interpreted as matrices of features. After rasterization, the method applies U-Net-like architecture to transform image channels to RGB. Several methods [9,18,50,65] build avatars using neural-textures. This allows them to represent more details than the RGB texture, especially out-of-mesh ones (*e.g.* loose clothing). However, such methods are prone to temporal inconsistency and flickering during animation.

In this work, we research the new task of merging a novel Gaussian-based approach with a classical RGB texture-based. This allows us to reduce the number of Gaussians and, therefore, reduce memory requirements to store an avatar. Utilizing textured mesh where possible helps us reduce the number of artifacts connected with redundant Gaussians while remaining Gaussians represent out-of-the-mesh elements of avatars. Thus, we leverage the pros from both representations.

3 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

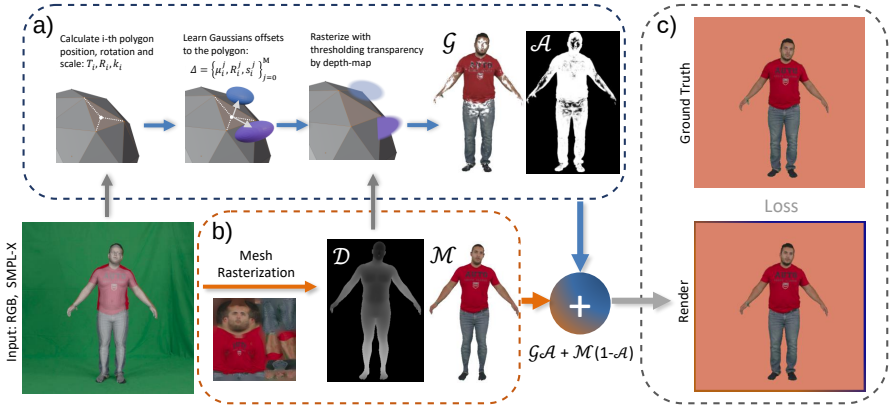

Fig. 2: Scheme of our approach. a) We attach Gaussians to mesh polygons as described in Section 3.1 and rasterize them conditioned on depth map $\mathcal{D}$ into RGB image $\mathcal{G}$ and alpha map $\mathcal{A}$. b) We train RGB texture for SMPL-X and rasterize mesh to RGB image $\mathcal{M}$ and 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.

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.

3.1 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 [48] 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:

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

Several works [20, 38, 40, 52] demonstrated the effectiveness of using neighboring Gaussians information, similar to convolution inductive bias in the Convolutional Neural Networks. Such a technique increases the similarity between neighboring Gaussians and reduces the number of artifacts. Following [38], we apply KNN regularization for Gaussians to constrain the transformation and

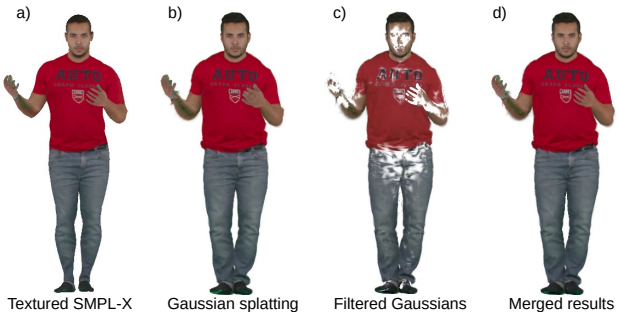

Fig. 3: Stages of training. a) SMPL-X with optimizable RGB texture fitted on input video frames. b) 3DGS trained as described in Section 3.1. c) All unnecessary Gaussians are deleted (Section 3.3) to merge this step with (a) and get (d).

appearance of neighbors. To further improve the avatars' quality, we use back-propagation in the SMPL-X to adjust pose and shape parameters. The effectiveness of such optimization of parameters was demonstrated in [20].

3.2 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 not transparent 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 [63], 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 [30] 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 [38] 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:

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

As a result of this stage, we get a full-body animatable human avatar (Fig. 3 (b)) with about 25k Gaussians.

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 [36]. 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 [5–7] we utilize three-channeled RGB texture.

Following [9], we utilize TV-regularization (L_{TV}) [10] 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:

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

As a result of such training, we get textured mesh (Fig. 3 (a)) that is fast to render and efficient to store. Although such a representation lacks out-of-mesh details.

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 (Fig. 3 (c)) 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 3.3) 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}) [43] 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 [16, 60]. 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. Finally, in the inference stage, we utilize preserved Gaussians and trained texture to render an avatar driven by SMPL-X pose parameters.

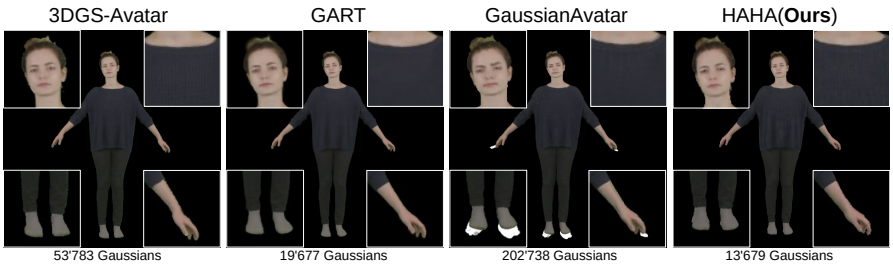

Fig. 4: Reconstruction for test frames from SnapshotPeople dataset (*female-3-casual*). Our method demonstrates the same subjective quality of reconstruction as state-of-the-art [20,38,49] while using fewer Gaussians to represent an avatar. For some sequences, GaussianAvatar [20] tends to include the white background color used in the training while the overall quality of the method is high.

3.3 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 (For more details, please refer to the supplementary materials). 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 [31]:

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

We then use alpha map $\mathcal{A}$ to mix rasterization $\mathcal{M}$ of the textured mesh (Fig. 3 (a)) with Gaussians rasterization $\mathcal{G}$ (Fig. 3 (c)) to get final avatar (Fig. 3 (d)). To obtain the final rasterization, we mix them as shown in Figure 2:

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

Fig. 5: 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 [20] also benefits from using SMPL-X to animate hands, HAHA produces more realistic-looking results.

The result is coherent because we already set transparency in $\mathcal{A}$ to zero for the Gaussians inside or behind the mesh. This formalization is a fully differentiable pipeline for rendering a mixture of Gaussians and textured mesh. The only issue with such an approach is that 3DGS accumulates color along the ray, taking into account the background color of the scene. We use rasterization $\mathcal{M}$ pixel values as background colors to circumvent color artifacts. So for half-transparent Gaussians we calculate the final color correctly.

4 Experiments

In our experiments, we compared HAHA to the state-of-the-art Gaussian methods, namely: GART [38], 3DGS-Avatar [49], and GaussianAvatar [20]. All these methods represent the human body as a set of Gaussians. We used two open datasets to evaluate our approach: X-Humans [53] and SnapshotPeople [7]. From both datasets, we used monocular RGB videos as input to our method. In the following section, we show both qualitative and quantitative results.

4.1 Implementation Details

In our experiments, we set loss weight values as follows. We set $\lambda_{LPIPS} = 0.01, \lambda_{SSIM} = 0.1, \lambda_{Sobel} = 1.0, \lambda_{KNN} = 0.01, L_{\text{TV}}$ to $\lambda_{TV} = 0.01$ for all stages' losses *i.e.* $\mathcal{L}_{\text{Gaussian}}, \mathcal{L}_{\text{texture}}$ and $\mathcal{L}_{\text{filtering}}$. We get the best trade-off in quality and number of Gaussians for learnable removing with the following regularization weights: $\lambda_{opacity} = 0.001, \lambda_{dice} = 0.1$. We trained all avatars using batch size equal to 4 using Adam [33] optimizer. In the first stage of training, we optimize

Table 1: Quantitative metrics for X-Humans [53] 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 [49]	42.77k	25.44	0.9315	0.0409	41.12k	27.63	0.9539	0.0471
GART [38]	55.85k	25.71	0.9295	0.0598	55.61k	27.78	0.9512	0.0668
GaussianAvatar [20]	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 [49]	26.78k	28.71	0.9521	0.0580	36.82k	26.84	0.9477	0.0445
GART [38]	50.47k	30.98	0.9595	0.0683	47.18k	26.56	0.9449	0.0595
GaussianAvatar [20]	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 [7] 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 [49]	53.78k	30.57	0.9581	0.0208	37.22k	34.28	0.9724	0.0149
GART [38]	19.67k	32.73	0.9672	0.0459	21.88k	35.93	0.9767	0.0294
GaussianAvatar [20]	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

Gaussian parameters for 3000 iterations. Then we optimize texture in the second stage for 2500 iterations. In the last stage, we fine-tuned Gaussians' color and opacity for 5000 iterations. We set other hyperparameters (such as learning rates) similar to GART [38]. For more details, please refer to supplementary materials.

4.2 X-Humans

We report the following metrics: PSNR, SSIM, and LPIPS [64] (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 [49] experiments to set the background value to zero. During inference, we used *test time pose optimization* following GART [38] to reduce the impact of SMPL-X fitting inaccuracies.

X-Humans [53] 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 [38], 3DGS-Avatar [49], and GaussianAvatar [20]. 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.

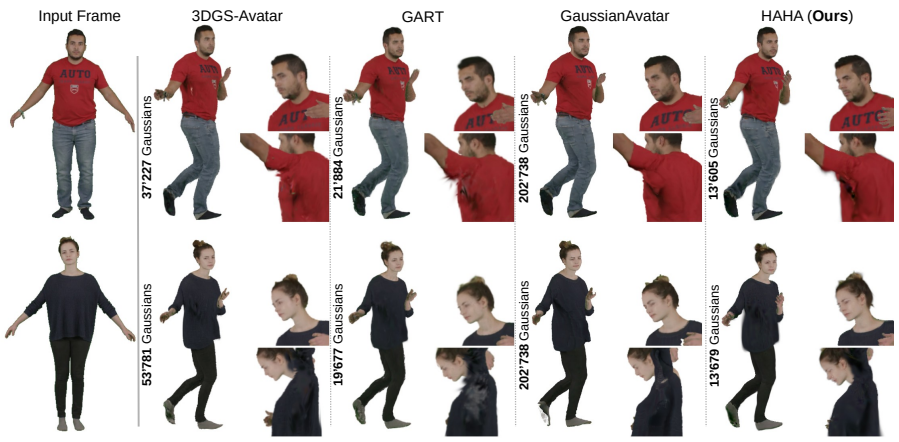

Fig. 6: Novel poses for SnapshotPeople dataset. HAHA reduces the number of artifacts for novel views and body regions unseen during training. At the same time, we use fewer Gaussians for rendering.

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 5 demonstrating overall quality and how our approach handles hands animation. Additional visualizations for more people from the dataset can be found in the supplementary materials.

4.3 PeopleSnapshot

Following the previous literature, we also provide quantitative metrics for the SnapshotPeople [7] 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 [11]. 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 [46] to match ground truth frames. Similar to X-Humans experiments, we evaluate metrics with a black background and use *test time pose optimization* during inference.

SnapshotPeople evaluation methodology is challenging for our method because we strongly rely on the underlying mesh geometry. Therefore, in cases 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). In Figure 4 we provide a qualitative comparison of avatar reconstruction for test frames from the PeopleSnapshot dataset. Our method

Table 3: Ablation study of losses and regularizations on *female-4-casual* from SnapshotPeople.

	PSNR ↑	SSIM ↑	LPIPS ↓	Gaussians ↓
No Sobel loss	30.74	0.9564	0.0331	6.89k
No opacity regularization	30.95	0.9582	0.0289	22.56k
No segmentation regularization	28.80	0.9450	0.0364	2.00k
Full pipeline	31.15	0.9589	0.0283	11.96k

demonstrates on par quality of the avatar using fewer Gaussians. Additionally, in some regions that are difficult to represent with Gausssians based on limited input data, our method reduces the number of artifacts (Fig. 6).

The qualitative improvement to the state-of-the-art is noticeable for novel poses and viewpoints (Fig. 6). To demonstrate how our and competitors' methods handle novel poses, we provide a comparison with reposed results. The use of textured mesh prior not only allows us to reduce the number of Gaussians but also reduces artifacts, especially in areas not sufficiently represented in the training frames. Additional visualizations and metrics for more people from the dataset can be found in the supplementary materials.

4.4 Ablation Study.

First, we ablate our loss choices: Sobel loss and the two proposed opacity regularizations. In Table 3, we provide quantitative metrics to evaluate the impact of each design choice. Figure 7 shows avatars corresponding to each table's row. According to our experiments, Sobel loss acts as an additional regularizer that prevents the deletion of Gaussians as it restricts the preserving of high-frequency details. So it could be switched off if one needs even fewer Gaussians but this will affect the sharpness of the edges and values of the quantitative metrics.

Both opacity and segmentation regularizations are essential to control the Gaussians amount. Without opacity regularization the method tends to inefficient Gaussians removal. The removal of segmentation regularization results in deleting too many Gaussians causing severe artifacts. We conclude that one should use both these regularizations simultaneously to get the best result.

We also evaluated the quality of avatars after each stage of training. We provided quantitative metrics in Table 4 while qualitative comparison can be found in Figure 3 (a, b, d). In this experiment, we also ablated the effectiveness of unnecessary Gaussians removal. *Naive merging* is a baseline method that merges Gaussians with a textured mesh without filtering them out. Using more Gaussians leads to higher PSNR and SSIM, while LPIPS decreases. As LPIPS is known for better correlation with human perception, we claim that removing part of the Gaussians leads to better quality.

In Table 4 we demonstrate how the reduction in the number of Gaussians affects the storage space consumption and rendering speed. Using fewer Gaussians with 256×256 RGB texture lets us use more than 2.3x less memory to store an

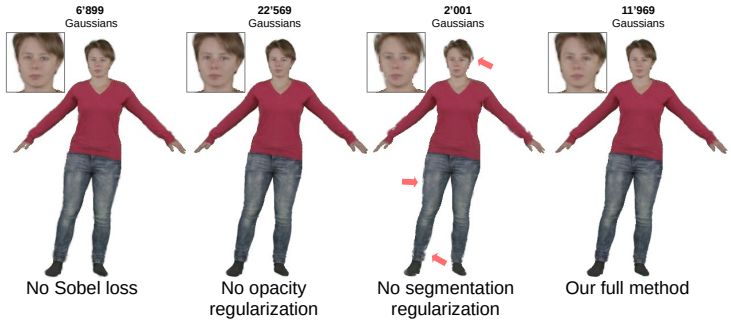

Fig. 7: Losses ablation study. HAHA gets the best trade-off between image quality and the number of Gaussians with the full set of proposed losses. Also removing Sobel loss could help to additionally reduce the amount of Gaussians but it leads to blurred edges.

Table 4: Metrics for each stage of avatar training pipeline. Metrics evaluated for *00019* subject from X-Humans dataset.

	Gaussians ↓	Storage ↓ (Mb)	FPS↑ (Inference)	Train ↓ (min:sec)	PSNR ↑	SSIM ↑	LPIPS ↓
Textured mesh	——	0.196	463.58 ± 17.46	5:24	27.26	0.9554	0.0539
Full-Gaussian	37.25k	2.086	240.09 ± 7.65	7:58	28.66	0.9601	0.0572
Naive merging	37.25k	2.282	238.17 ± 7.81	——	28.51	0.9599	0.0510
Finetuning	12.26k	0.883	247.87 ± 4.62	12:19	28.49	0.9593	0.0501

avatar. Such storage space reduction could be useful for industrial applications when it is necessary to store avatars for millions of users. At the same time, we demonstrate total convergence speed (25 min 41 sec) on par with other methods: GaussianAvatar (28 min 37 sec), 3DGS-Avatar (26 min 03 sec).

5 Discussion and 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.

Our method's limitation is the difficulty of getting an accurate SMPL-X mesh for an input video. As we strongly depend on how accurate mesh projection matches the input frames. The task of getting SMPL-X parameters from a monocular video is long-standing but still has room for improvement.

References

1. Expand your world with Meta Quest. https://www.meta.com/it/en/quest/, [Online; accessed 27-June-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-June-2024] 1
3. Mark zuckerberg: First interview in the metaverse. https://lexfridman.com/mark-zuckerberg-3/, online; accessed 27-February-2024 1
4. Texel 3d body model dataset. https://texel.graphics/texel-3d-body-model-dataset/, online; accessed 27-June-2024 2
5. 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) 2, 3, 4, 5, 8
6. 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) 2, 3, 4, 5, 8
7. 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 2, 3, 5, 8, 10, 11, 12
8. Alldieck, T., Zanfir, M., Sminchisescu, C.: Photorealistic monocular 3d reconstruction of humans wearing clothing. In: CVPR. pp. 1506–1515 (2022) 2
9. 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) 2, 5, 8
10. Chambolle, A.: An algorithm for total variation minimization and applications. Journal of Mathematical imaging and vision **20**, 89–97 (2004) 8
11. 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) 2, 4, 12
12. Chen, Y., Wang, L., Li, Q., Xiao, H., Zhang, S., Yao, H., Liu, Y.: Monogaussianavatar: Monocular gaussian point-based head avatar. arXiv preprint arXiv:2312.04558 (2023) 2, 4
13. Dhamo, H., Nie, Y., Moreau, A., Song, J., Shaw, R., Zhou, Y., Pérez-Pellitero, E.: Headgas: Real-time animatable head avatars via 3d gaussian splatting. arXiv preprint arXiv:2312.02902 (2023) 2, 4
14. Duan, H.B., Wang, M., Shi, J.C., Chen, X.C., Cao, Y.P.: Bakedavatar: Baking neural fields for real-time head avatar synthesis. ACM TOG **42**(6), 1–17 (2023) 4
15. Duan, Y., Wei, F., Dai, Q., He, Y., Chen, W., Chen, B.: 4d gaussian splatting: Towards efficient novel view synthesis for dynamic scenes (2024) 4
16. Gong, K., Gao, Y., Liang, X., Shen, X., Wang, M., Lin, L.: Graphonomy: Universal human parsing via graph transfer learning. In: CVPR (2019) 8
17. Grassal, P.W., Prinzler, M., Leistner, T., Rother, C., Nießner, M., Thies, J.: Neural head avatars from monocular rgb videos. In: CVPR. pp. 18653–18664 (2022) 4
18. Grigorev, A., Iskakov, K., Ianina, A., Bashirov, R., Zakharkin, I., Vakhitov, A., Lempitsky, V.: Stylepeople: A generative model of fullbody human avatars. In: CVPR. pp. 5151–5160 (2021) 2, 5

19. He, T., Xu, Y., Saito, S., Soatto, S., Tung, T.: Arch++: Animation-ready clothed human reconstruction revisited. In: ICCV. pp. 11046–11056 (2021) 2
20. 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. CVPR pp. 634–644 (2024) 2, 3, 4, 6, 7, 9, 10, 11
21. Hu, S., Liu, Z.: Gauhuman: Articulated gaussian splatting from monocular human videos. CVPR pp. 20418–20431 (2024) 2, 4
22. Huang, L., Bai, J., Guo, J., Li, Y., Guo, Y.: On the error analysis of 3d gaussian splatting and an optimal projection strategy (2024) 4
23. Işık, M., Rünz, M., Georgopoulos, M., Khakhulin, T., Starck, J., Agapito, L., Nießner, M.: Humanrf: High-fidelity neural radiance fields for humans in motion. ACM TOG **42**(4), 1–12 (2023). https://doi.org/10.1145/3592415, https://doi.org/10.1145/3592415 2
24. Jena, R., Iyer, G.S., Choudhary, S., Smith, B., Chaudhari, P., Gee, J.: Splatarmor: Articulated gaussian splatting for animatable humans from monocular rgb videos. arXiv preprint arXiv:2311.10812 (2023) 2, 4
25. Jiang, T., Chen, X., Song, J., Hilliges, O.: Instantavatar: Learning avatars from monocular video in 60 seconds. CVPR pp. 16922–16932 (2022) 2
26. Jiang, T., Chen, X., Song, J., Hilliges, O.: Instantavatar: Learning avatars from monocular video in 60 seconds. In: CVPR. pp. 16922–16932 (2023) 2, 4
27. Jiang, Y., Tu, J., Liu, Y., Gao, X., Long, X., Wang, W., Ma, Y.: Gaussianshader: 3d gaussian splatting with shading functions for reflective surfaces. CVPR pp. 5322–5332 (2024) 4
28. Jiang, Y., Shen, Z., Wang, P., Su, Z., Hong, Y., Zhang, Y., Yu, J., Xu, L.: Hifi4g: High-fidelity human performance rendering via compact gaussian splatting. CVPR pp. 19734–19745 (2024) 4
29. 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
30. 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) 7
31. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. ACM TOG **42**(4) (2023) 2, 3, 4, 9
32. Kilian, M., Mitra, N.J., Pottmann, H.: Geometric modeling in shape space. In: ACM TOG, pp. 64–es (2007) 4
33. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. CoRR **abs/1412.6980** (2014), https://api.semanticscholar.org/CorpusID:6628106 10
34. Kocabas, M., Athanasiou, N., Black, M.J.: Vibe: Video inference for human body pose and shape estimation. In: CVPR. pp. 5253–5263 (2020) 4
35. Kratimenos, A., Lei, J., Daniilidis, K.: Dynmf: Neural motion factorization for real-time dynamic view synthesis with 3d gaussian splatting. arXiv preprint arXiv:2312.00112 (2023) 4
36. 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) 7
37. Lee, B., Lee, H., Sun, X., Ali, U., Park, E.: Deblurring 3d gaussian splatting (2024) 4
38. Lei, J., Wang, Y., Pavlakos, G., Liu, L., Daniilidis, K.: Gart: Gaussian articulated template models. CVPR pp. 19876–19887 (2024) 2, 3, 4, 6, 7, 9, 10, 11

39. Li, T., Bolkart, T., Black, M.J., Li, H., Romero, J.: Learning a model of facial shape and expression from 4D scans. ACM TOG **36**(6), 194:1–194:17 (2017), https://doi.org/10.1145/3130800.3130813 2, 4
40. Li, Z., Zheng, Z., Wang, L., Liu, Y.: Animatable gaussians: Learning pose-dependent gaussian maps for high-fidelity human avatar modeling. CVPR pp. 19711–19722 (2024) 4, 6
41. 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, 3, 4
42. Luiten, J., Kopanas, G., Leibe, B., Ramanan, D.: Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis pp. 800–809 (2024) 4
43. 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) 8
44. 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) 3
45. Pang, H., Zhu, H., Kortylewski, A., Theobalt, C., Habermann, M.: Ash: Animatable gaussian splats for efficient and photoreal human rendering. CVPR pp. 1165–1175 (2024) 4, 5
46. 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, 3, 4, 12
47. Peng, S., Zhang, Y., Xu, Y., Wang, Q., Shuai, Q., Bao, H., Zhou, X.: Neural body: Implicit neural representations with structured latent codes for novel view synthesis of dynamic humans. In: CVPR. pp. 9054–9063 (2021) 2
48. Qian, S., Kirschstein, T., Schoneveld, L., Davoli, D., Giebenhain, S., Nießner, M.: Gaussianavatars: Photorealistic head avatars with rigged 3d gaussians. CVPR pp. 20299–20309 (2024) 2, 4, 5, 6
49. Qian, Z., Wang, S., Mihajlovic, M., Geiger, A., Tang, S.: 3dgs-avatar: Animatable avatars via deformable 3d gaussian splatting. CVPR pp. 5020–5030 (2024) 2, 3, 4, 9, 10, 11
50. Raj, A., Tanke, J., Hays, J., Vo, M., Stoll, C., Lassner, C.: Anr: Articulated neural rendering for virtual avatars. In: CVPR. pp. 3722–3731 (2021) 5
51. Saito, S., Huang, Z., Natsume, R., Morishima, S., Kanazawa, A., Li, H.: Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. In: ICCV. pp. 2304–2314 (2019) 2
52. Saito, S., Schwartz, G., Simon, T., Li, J., Nam, G.: Relightable gaussian codec avatars. CVPR pp. 130–141 (2024) 5, 6
53. Shen, K., Guo, C., Kaufmann, M., Zarate, J., Valentin, J., Song, J., Hilliges, O.: X-avatar: Expressive human avatars. CVPR (2023) 3, 5, 10, 11
54. Sun, Y., Bao, Q., Liu, W., Fu, Y., Black, M.J., Mei, T.: Monocular, one-stage, regression of multiple 3d people. In: ICCV. pp. 11179–11188 (2021) 4
55. Svitov, D., Gudkov, D., Bashirov, R., Lempitsky, V.: Dinar: Diffusion inpainting of neural textures for one-shot human avatars. In: ICCV. pp. 7062–7072 (2023) 2
56. Thies, J., Zollhöfer, M., Nießner, M.: Deferred neural rendering: Image synthesis using neural textures. ACM TOG **38**(4), 1–12 (2019) 5
57. Waczyńska, J., Borycki, P., Tadeja, S., Tabor, J., Spurek, P.: Games: Mesh-based adapting and modification of gaussian splatting (2024) 5
58. Wang, J., Li, X., Xie, J., Xu, F., Gao, H.: Gaussianhead: Impressive 3d gaussian-based head avatars with dynamic hybrid neural field. arXiv e-prints pp. arXiv–2312 (2023) 2, 4

59. Xiang, J., Gao, X., Guo, Y., Zhang, J.: Flashavatar: High-fidelity digital avatar rendering at 300fps. arXiv preprint arXiv:2312.02214 (2023) 5
60. 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) 8
61. Yu, Z., Chen, A., Huang, B., Sattler, T., Geiger, A.: Mip-splatting: Alias-free 3d gaussian splatting. CVPR pp. 19447–19456 (2024) 4
62. Yu, Z., Cheng, W., Liu, X., Wu, W., Lin, K.Y.: Monohuman: Animatable human neural field from monocular video. In: CVPR. pp. 16943–16953 (2023) 2
63. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: CVPR (2018) 7
64. 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) 11
65. Zhao, H., Zhang, J., Lai, Y.K., Zheng, Z., Xie, Y., Liu, Y., Li, K.: High-fidelity human avatars from a single rgb camera. In: CVPR. pp. 15904–15913 (2022) 2, 5
66. Zheng, S., Zhou, B., Shao, R., Liu, B., Zhang, S., Nie, L., Liu, Y.: Gps-gaussian: Generalizable pixel-wise 3d gaussian splatting for real-time human novel view synthesis. CVPR pp. 19680–19690 (2024) 4
67. 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 3, 5
68. Zielonka, W., Bagautdinov, T., Saito, S., Zollhöfer, M., Thies, J., Romero, J.: Drivable 3d gaussian avatars. arXiv preprint arXiv:2311.08581 (2023) 2, 4