This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

3D Gaussian Head Avatars with Expressive Dynamic Appearances by Compact Tensorial Representations

Yating Wang[1] Xuan Wang[2] Ran Yi[1*] Yanbo Fan[2] Jichen Hu[1] Jingcheng Zhu[1]
Lizhuang Ma[1*]
[1]Shanghai Jiao Tong University [2]AntGroup Research

Abstract

Recent studies have combined 3D Gaussian and 3D Morphable Models (3DMM) to construct high-quality 3D head avatars. In this line of research, existing methods either fail to capture the dynamic textures or incur significant overhead in terms of runtime speed or storage space. To this end, we propose a novel method that addresses all the aforementioned demands. In specific, we introduce an expressive and compact representation that encodes texture-related attributes of the 3D Gaussians in the tensorial format. We store appearance of neutral expression in static tri-planes, and represents dynamic texture details for different expressions using lightweight 1D feature lines, which are then decoded into opacity offset relative to the neutral face. We further propose adaptive truncated opacity penalty and class-balanced sampling to improve generalization across different expressions. Experiments show this design enables accurate face dynamic details capturing while maintains real-time rendering and significantly reduces storage costs, thus broadening the applicability to more scenarios.

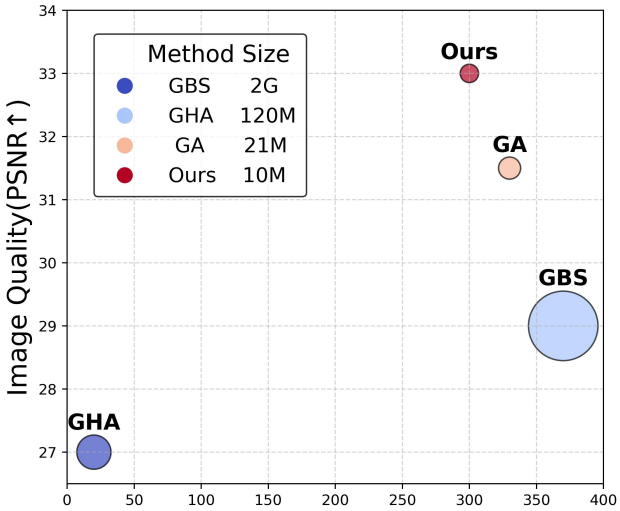


Figure 1. Our method improves rendering quality while ensuring real-time performance and minimum storage. The points radius in the figure is proportional to the square root of the storage.

1. Introduction

Photorealistic 3D head avatars reconstruction is a fundamental research topic in the field of computer graphics and vision, encompassing a variety of applications including films, gaming, and AR/VR etc. A highly practical 3D head avatar needs to meet various technical requirements: 1) It needs to faithfully render the expressions and movements depicted in the driving signal into high-quality rendered images, including clear texture details, consistent geometric structures, and vivid texture variations under dynamic expressions; 2) It requires extremely high inference and rendering efficiency to meet the demands of tasks that require real-time performance; 3) It needs to meet the requirements of lightweight design to facilitate easy distribution and deployment. However, meeting all of the aforementioned requirements remains a significantly challenge.

Early methods are based on 3D face Morphable Models (3DMM) [2, 32] to achieve 3D avatar reconstruction, which can easily control facial expressions using low-dimensional PCA coefficients, but the rendering effect lacks realism. With the development of NeRF [25], some approaches combine NeRF and 3DMM to achieve high-quality, animatable head avatars, but the volumetric rendering process of NeRF requires substantial computation, making it difficult to achieve real-time performance. Recently, 3D Gaussian Splatting (3DGS) [18] has gained widespread attention for its high-quality rendering and real-time performance, with some works attempting to combine 3DGS with 3DMM to achieve fast, realistic, and animatable head avatars.

To enhance realism of head avatar rendering, it is important to model the dynamic facial details that change with

*Corresponding author

expressions, but capturing such details inevitably introduces additional time and memory costs. Some methods [24] explicitly store a set of Gaussian splats for each blendshape, resulting in significant storage requirements. Alternatively, other methods [37] employ MLPs to implicitly model dynamic textures, followed by super-resolution techniques to enhance details rendering, but at the cost of not being able to render in real-time. Additionally, more parameters and complex networks often lead to overfitting, resulting in poorer generalization ability on novel expressions.

To address these issues, we propose a novel 3D head avatar modeling method that takes into account both dynamic texture modeling and spatiotemporal efficiency. First the efficient 3D Gaussian Splatting rendering pipeline is employed. Then the 3D Gaussian splats are bound to a parametric head mesh, enabling the splats to depict the base motion of the face along with the mesh. Observing that neighboring Gaussian splats share similar appearances and dynamics, hence we store the static texture and dynamic properties in the involved compact tensorial features to reduce spatial redundancy. To be specific, the static texture in neutral expression is stored in **tri-planes** within the canonical space, replacing spherical harmonics of 3DGS. For facial dynamics modeling, we store a neural grid representing opacity offsets relative to the neutral face for each blendshape. Then we exploit **1D feature lines** to depict the dynamic part of facial textures, as experiments show that those features can adequately capture the texture changes caused by facial dynamics, and further reduce the occupied storage. These feature lines are interpolated by blendshape coefficients, and a non-linear MLP decoder outputs the opacity offset. Benefited from compact architecture, our method enables high-fidelity head avatar with dynamic textures while maintaining storage efficiency and real-time inference.

Compact tensorial representations also reduce the risk of overfitting to training expressions. We further propose two training strategies to improve generalization ability to novel expressions. First, as areas which remain consistent to the neutral expression should not have opacity offsets, we propose an adaptive truncated penalty on opacity offset which identifies relatively static mesh triangles in each frame and constrains opacity offset of their corresponding splats to be minimal. Second, as large-scale expressions are underrepresented in the training set, we propose a class-balanced resampling method: training expressions are clustered, and samples are drawn uniformly from each cluster.

We conduct experiments on the Nersemble [19] dataset, showing that our method accurately reconstructs dynamic facial details and improves rendering metrics. Our compact representations require no more than 10MB per subject, making it the most storage-efficient method compared to the state-of-the-art competitors. We achieve 300 FPS, ensuring real-time performance. The spatial and temporal efficiency of our approach allows it to be extended to broader application scenarios, such as fast network transmission and real-time rendering in mobile video conferencing.

2. Related Works

2.1. Animatable Head Avatar

Traditional head avatar methods rely on **3D M**orphable face **M**odel [2, 32] to control face motion by low-dimensional coefficients but lacks rendering realism. Recently, advances in neural rendering has led to combining neural scene representations like NeRF [25] and 3DGS [18] with 3DMM to achieve high-quality animatable head avatars [1, 10, 11, 29, 40] by conditioning NeRF or 3DGS on 3DMM coefficients or meshes. Further approaches attempt to improve head avatar reconstruction from various perspectives, such as inference speed [34, 45], statistical face model [12, 14, 38, 43], dynamic details [5, 35], sparse-view robustness [7, 42] and texture-material disentanglement [30, 36]. Below, we primarily review 3DGS head avatar methods focused on dynamic facial textures. Some approaches [6, 34] use MLPs to capture dynamic offsets of gaussian attributes, while others employ CNNs [11, 30]. Super-resolution modules are also incorporated to improve the quality of dynamic details rendering [13, 37]. Accurately capturing dynamic details often requires multiple large networks, which highly increase time costs. Methods such as [8, 24] store expression-related dynamic information as additional attributes per Gaussian splat, leading to significant memory usage. By contrast, our approach achieves high-fidelity rendering with dynamic facial details with 10M storage and 300FPS rendering speed.

2.2. Compact 3D Scene Representations for Novel View Synthesis.

NeRF employs deep MLPs to model scenes, but suffers from slow inference and scalability issues for large, unbounded areas. Some works [3, 4, 26] address this by replacing one large MLP with spatially discrete features and small MLP decoders, improving rendering efficiency. Other methods reduce computation by decomposing large NeRFs into smaller sub-networks [39]. 3DGS is efficient in rendering speed but requires storing a large number of splats for large or complex scene. Recent approaches address this issue using region-based vector quantization [28], K-means-based codebooks [27], or learned binary masks for each Gaussian. The memory-intensive spherical harmonics in 3DGS are replaced with more compact neural networks [20, 46] for view-dependent radiance. Furthermore, some approaches attempt to model dynamic scenes using compact format, such as 2D plane [9, 31], sparse control points [15, 17] and parametric curves [23, 33]. In this paper, we leverage tensorial representations(tri-planes and 1D feature lines) to represent static and dynamic appearances.

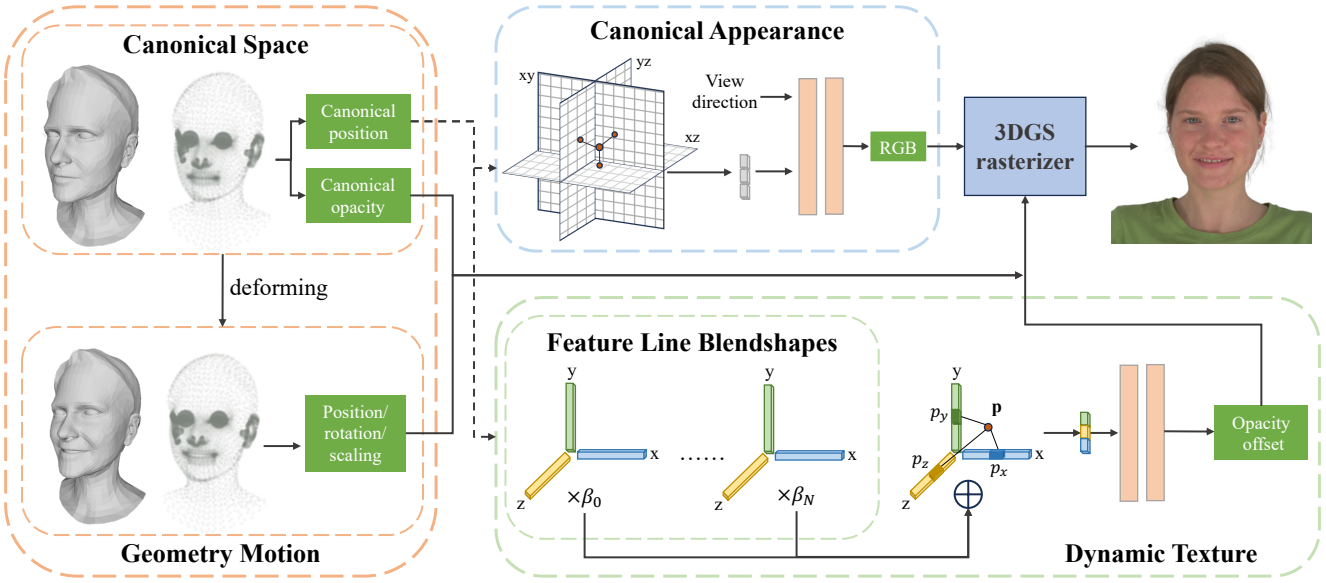


Figure 2. Our goal is to reconstruct 3DGS head avatar with dynamic details, ensuring real-time rendering and minimized storage. We use a **parametric face mesh** to describe large-scale geometry motions, moving the bound Gaussian splats accordingly. A **triplane** stores view-dependent appearance in canonical space, while **1D feature lines** are used for dynamic details per blendshape, allowing interpolation with expression coefficients. Finally, the geometry attributes of the splats, along with the canonical appearance and dynamic details, are combined to render the face image.

3. Method

Our method takes multi-view face videos as inputs and outputs an animatable head avatar with dynamic textures. As shown in Fig. 2, our method stores large-scale head motion, head appearance in the canonical space, and dynamic texture variations for each blendshape using three different structures, *i.e.,* mesh, triplane, and feature lines, respectively. 1) Geometry motion bound to mesh: We follow [29] to track FLAME [21] mesh for each frame from multi-view images and known camera parameters and calculate splats geometry attributes (position/rotation/scaling) by tracked meshes. 2) Appearance of neutral face in triplanes: Tri-planes are used to store view-dependent radiance of the canonical 3D face. The features sampled from the tri-planes, and view direction transformed into the canonical space, are fed into a tiny MLP decoder to obtain RGB color. 3) Dynamic details in feature lines: We utilize a feature line per blendshape to store dynamic textures, which can be interpolated by tracked blendshape coefficients. An MLP decoder then maps the features sampled from the interpolated feature line into opacity offset, which is added to canonical opacity. Finally, the aforementioned gaussian attributes are combined to render the image.

3.1. Preliminaries

3DGS. 3D Gaussian Splatting [18] enables novel view synthesis of a static scene from multi-view images and camera parameters. A scene is represented by a collection of 3D Gaussian splats, and each 3D Gaussian contains the following attributes: position $\mu \in \mathbb{R}^3$, scaling $\mathbf{s} \in \mathbb{R}^3$, quaternion $\mathbf{q} \in \mathbb{R}^4$, opacity $\alpha \in \mathbb{R}$ and spherical harmonics $\mathbf{SH} \in \mathbb{R}^{(k+1)^2 \times 3}$ to represent view-dependent color (where k is $\mathbf{SH}$ degree). In our paper, we use $\mathbf{r} \in \mathbb{R}^{3\times3}$ to notate the corresponding rotation matrix to $\mathbf{q}$. Finally the final color for a given pixel is calculated by sorting and blending the overlapped Gaussians.

Gaussian Avatars [29] extends 3DGS from static scene to dynamic avatar by binding Gaussian splats to a tracked head mesh, which is obtained by fitting FLAME (a face shape prior model) parameters to multi-view observations. Each 3D Gaussian splat is paired with a mesh triangle, with its geometric attributes$(\mu', \mathbf{r}', \mathbf{s}')$ defined in the triangle's local coordinate system. Once optimized, these attributes are fixed, making each splat's relative position within its triangle fixed, yet allowing global movement as the triangle moves. For a triangle with three vertices, the average vertex position T is set to origin of local coordinate system. Rotation matrix R is constructed using one edge direction, the triangle normal, and their cross product to represent the orientation transformation from local to global space. A scalar k is computed as the mean length of one of the edges and its perpendicular. Then the transformation from the local space to the global space is conducted as:

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

3.2. Appearance in Canonical Space via Triplane

In 3DGS, 48 out of the total 59 parameters for each Gaussian are used for **SH** (3 degrees) to capture view-dependent color. Noticing that neighboring splats should have similar appearance, instead of storing 48 **SH** parameters per splat, we use a triplane to store implicit encodings, and a tiny MLP decoder to decode the encodings along with view direction to RGB colors, which compresses the model size. The triplane T consists of three orthogonal feature planes aligned with the axes: $\{T_{xy}, T_{xz}, T_{yz}\} \in \mathbb{R}^{3\times n_f \times n_f \times n_{d1}}$, where $n_f \times n_f$ is spatial resolution of the 2D feature planes, and n_{d1} is the feature dimension. For any given position $\mathbf{p}$ in canonical space, the corresponding feature is obtained by projecting $\mathbf{p}$ onto the axis-aligned planes (the $x-y, x-z$ and $y-z$ planes), interpolating to obtain the features on each feature plane and concatenating the interpolated features, which is formulated as:

$$t(\mathbf{p}) = \textit{interp}(T_{xy}, \mathbf{p}_{xy}) \oplus \textit{interp}(T_{xz}, \mathbf{p}_{xz}) \oplus \textit{interp}(T_{yz}, \mathbf{p}_{yz}),$$

where *interp* represents bilinear interpolation, $\oplus$ represents concatenation, and $\mathbf{p}_{xy}, \mathbf{p}_{xz}, \mathbf{p}_{yz}$ refer to the projected position on each plane.

Note that the triplane is defined in the *canonical space*, which corresponds to the global space of the neutral expression. In contrast, we refer to the global space of the non-neutral expression in each frame as the *deformed space*. Given a splat with the position μ', rotation $\mathbf{r}'$, and scale $\mathbf{s}'$ defined in the local space, it can be transformed into the canonical space and deformed space of the current frame based on the canonical transformation (R_c, T_c, k_c) and the deformed transformation (R_d, T_d, k_d) respectively. The local coordinate system can also serve as a bridge to transform between the deformed space and the canonical space. The transformation from view direction in the deformed space (denoted as $\mathbf{v}_d$) to view direction in the canonical space (denoted as $\mathbf{v}_c$) can be formulated as:

$$\mathbf{v}_c = R_c R_d^{-1} \mathbf{v}_d.$$

Finally a tiny MLP decodes $t(\mathbf{p})$ and $\mathbf{v}_c$ into RGB value $\mathbf{c}$, which can be used as the first three components of degree-1 **SH** for 3DGS rendering. Additionally, since the majority of facial information is concentrated in the frontal view, with less information available from the side views, we utilize larger feature dimension for T_{xy} and lower dimension for T_{xz}, T_{yz} to achieve storage compression.

3.3. Dynamic Texture via Feature Line Blendshapes

Existing researches show that 3D Gaussian texture attributes can be effectively compressed using tri-planes, confirming color consistency in local neighborhoods. We extend this local consistency to dynamic avatars for efficient dynamic details representation. For each expression blendshape, we store a separate representation that describes the texture changes (opacity offsets) relative to the neutral expression induced by that expression. Since the FLAME model has 100 PCA blendshapes, storing 2D planes or 3D tensors for each blendshape incurs excessive memory consumption. To enable a compact representation for dynamic textures, we use lightweight 1D feature lines to encode the texture changes for blendshape i, denoted as $(L_x^i, L_y^i, L_z^i) \in \mathbb{R}^{3\times n_{d2}\times n_s}$, where n_s represents the length of the 1D feature line, and n_{d2} represents the feature dimension. We interpolate feature lines by tracked blendshape coefficients $\beta_j \in \mathbb{R}^{n_b}$ to obtain specific feature line for frame j, where n_b is blendshape count, which can be formulated as:

$$l_b^j = \sum_{i=0}^{n_b} \beta_j^i * (L_x^i, L_y^i, L_z^i).$$

In addition to the linear blendshapes, FLAME incorporates a nonlinear quaternion jaw rotation to describe large-scale jaw movements. To unify the linear basis with the nonlinear rotation, we follow the method proposed in [22], extracting linear jaw rotation bases $\{\mathbf{q}_k : k \in \{0, \ldots, n_j\}\}$ from jaw rotations in training videos via farthest point sampling. And we store a feature line (L_x^k, L_y^k, L_z^k) for each jaw basis. We follow [16] to calculate the distance between jaw rotation of frame j and the k-th jaw rotation basis which can be formulated as $d(j,k) = 1 - |\mathbf{q}_j^T \mathbf{q}_k|$($\mathbf{q}_j$ and $\mathbf{q_k}$ are unit quaternion). And the jaw feature lines are interpolated using inverse distance weighting to calculate the feature line of frame j, formulated as:

$$l_r^j = \sum_{k=0}^{n_j} \beta_j^k * (L_x^k, L_y^k, L_z^k), \qquad \beta_j^k = \frac{1 - d(j,k)}{\sum_{k=0}^{n_j}(1 - d(j,k))}.$$

Similar to the triplane, the opacity offset features of a given position $\mathbf{p}$ in frame j is calculated by projecting $\mathbf{p}$ onto the x, y and z axes, interpolating and concatenating the interpolated features, formulated as:

$$l^j(\mathbf{p}) = \textit{interp}(l_x^j, \mathbf{p}_x) \oplus \textit{interp}(l_y^j, \mathbf{p}_y) \oplus \textit{interp}(l_z^j, \mathbf{p}_z).$$

This projection and interpolation process is applied to both the expression blendshape feature line l_b^j and the jaw rotation feature line l_r^j.

We finally utilize a tiny MLP θ to decode interpolated $l_b^j(\mathbf{p})$ and $l_r^j(\mathbf{p})$ into opacity offset $\Delta\alpha$, which will be added to the canonical opacity α_c (opacity of the neutral expersion). The final opacity is calculated as:

$$\alpha = \alpha_c + \Delta\alpha = \alpha_c + \theta(l_b^j(\mathbf{p}), l_r^j(\mathbf{p})).$$

The facial motion caused by facial expressions are primarily concentrated in the leading components of FLAME

PCA blendshapes. Using only the leading expression coefficients achieves similar results, allowing us to reduce the number of feature lines, thereby reducing storage and accelerating calculation.

3.4. Training

Class-balanced Sampling. Most images in the training set show minor facial movements, with only few displaying large expressions, making it challenging to accurately reconstruct these. Since the data distribution is unknown, simply oversampling large expressions may bias against smaller ones. We propose a resampling method to ensure that various expressions are sampled evenly during training.

First, we measure the similarity between two frames by comparing the differences in vertex displacements of the FLAME mesh. Given that different regions of the face exhibit varying degrees of motion, *e.g.,* the lips moving significantly more than the eyes, we empirically increase weights of the eyes vertices. We denote the FLAME mesh for frame i as M_i, and the similarity score between frame i and frame j is calculated as $dist(i,j) = ||M_i - M_j||_2^2 * \mathbf{w}$, where $\mathbf{w}$ denotes the weight of each vertex. We set $\mathbf{w} = 2$ for eye vertices and $\mathbf{w} = 1$ for other vertices. Next, we perform spectral clustering on the similarity matrix to categorize all frames into $n = 16$ classes. Finally, we utilize a uniform sampling probability distribution between categories and equal probability within each category.

Adaptive Truncated Opacity Offset Penalty. The feature lines store the opacity offset of each blendshape relative to the neutral face, hence the offsets in relatively static regions are ideally zero. Therefore, we calculate triangle translations $\bar{\mathbf{t}}$, *i.e.*, the displacement between the deformed mesh and the neutral face mesh, and empirically set a threshold τ to identify whether the triangle is "static" or "dynamic". Then opacity offsets of splats bound to "static" triangles are constrained to be close to 0. This penalty helps decouple static textures from dynamic details, enhancing generalization to unseen expressions, formulated as:

$$\mathcal{L}_{op} = \lambda_{op}|\Delta\alpha| * w_{op}, \quad w_{op} = \begin{cases} 0, & \text{if } |\bar{\mathbf{t}}| > \tau, \\ 1, & \text{if } |\bar{\mathbf{t}}| <= \tau. \end{cases} \tag{2}$$

Loss Function. We use the L1 loss and D-SSIM loss between the rendered images and the ground truth images as image supervision, which can be formualted as:

$$\mathcal{L}_{image} = (1-\lambda)\mathcal{L}_1 + \lambda\mathcal{L}_{\text{D-SSIM}}.$$

Assuming that the Gaussian splats should roughly conform to the mesh and be similar in size with the bound triangles, we follow methods proposed by [29] to employ a position loss and a scale loss to prevent splats from being excessively far from the mesh or excessively large:

$$\mathcal{L}_{geom} = \lambda_{pos}\mathcal{L}_{pos} + \lambda_{scale}\mathcal{L}_{scale}. \tag{3}$$

The total training loss can be formulated as:

$$\mathcal{L} = \mathcal{L}_{image} + \mathcal{L}_{geom} + \mathcal{L}_{op},$$

where $\lambda = 0.2$, $\lambda_{pos} = 0.01$, $\lambda_{scale} = 1$, and $\lambda_{op} = 1$.

4. Experiments

4.1. Settings and Dataset

We conduct experiments on nine individuals from the nersemble dataset, collecting a total of 11 video segments of each subject from 16 different viewpoints. Each participant performs 10 distinct expressions and emotions as instructed, followed by a free performance in the last video segment. The videos are downsampled to a resolution of 802 × 550. We utilize the FLAME coefficients and camera parameters provided in GA [29], including shape β, translation $\mathbf{t}$, pose θ, expression ψ, and vertex offset $\Delta\mathbf{v}$ in the canonical space.

We compare the experimental results across three tasks: 1) Novel View Synthesis: 15 out of 16 viewpoints are used for training, while the remaining viewpoint is reserved for testing; 2) Self-Reenactment: Testing is conducted using videos of the same individual showcasing unseen poses and expressions from all 16 viewpoints; 3) Cross-Identity Reenactment: An avatar is driven by the motions and expressions of other individuals. We use free performance sequences for testing of task 2 and task 3.

4.2. Implementation Details

We implemented our approach using PyTorch and trained for 600,000 iterations using the Adam optimizer for each subject. Both the triplane and feature lines consist of two components: a neural grid feature and a MLP decoder. We train triplane and feature line blendshapes with the same learning rates, setting learning rate of the feature to $2e-3$ and the MLP learning rate to $1e-4$. Each plane of the triplane is 128×128 in size, with the feature dimension of the T_{xy} being 32, and the feature dimensions of the T_{xz}, T_{yz} planes set to 16. The MLP decoder following triplane consists two-layer with 128 dimensions per layer, using ReLU as the activation function. We apply position encoding to improve the resolution of view direction. We assign feature lines with a spatial resolution of 64 and feature dimension of 32 to the first 80 PCA expression bases and 16 key jaw rotation bases. The decoder for the feature lines is a two-layer MLP with 128 dimensions per layer. The triplane requires 4.05M of storage, the feature line requires 2.41M, and the other Gaussian attributes (including position/rotation/scaling and canonical opacity) average 3.25M per subject.

4.3. Comparison

We compare three baselines with our method.

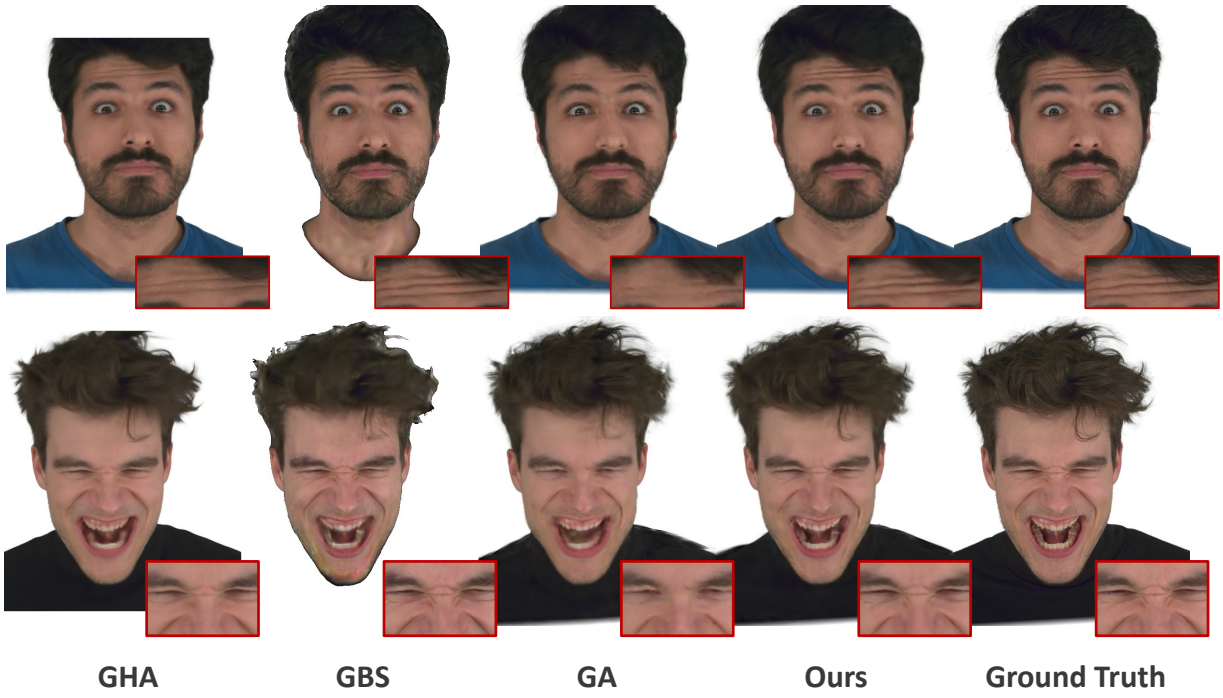


Figure 3. Qualitative comparison with baseline methods on novel view synthesis task.

	Novel View Synthesis			Self-Reenactment			Performance	
Method	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	Storage	FPS
GA	31.4702	0.9489	0.05144	27.2678	0.9230	0.06668	21M	330
GHA	26.9932	0.9347	0.04905	22.7397	0.8895	0.07995	120M	20
GBS	28.8966	0.9500	0.06311	25.9797	0.9270	0.08099	2G	370
Ours	32.9688	0.9506	0.05940	28.0688	0.9259	0.07724	10M	300

Table 1. Quantitative comparison with baselines. GREEN indicates the best of all methods. YELLOW indicates the second.

1) GA [29] defines Gaussian splats in the relative coordinate system of the mesh triangles, allowing splats to move with the mesh. However, GA uses the same set of Gaussian splats for all expressions, limiting its ability to capture expression-specific details.

2) GBS [24] optimizes a set of 3D Gaussians splats per blendshape, which can be interpolated linearly using 3DMM coefficients, but at the cost of significant storage. Moreover, extensive parameters may lead to overfitting to training expressions and simple linear interpolation of splats attributes may introduces artifacts in large expressions.

3) GHA [37] uses coarse guide meshes and two MLPs to predict geometry and color offsets, and converts 3D Gaussian features into RGB images via super-resolution. GHA achieve highly detailed rendering, but fail to achieve real-time rendering speed.

Different from GA, GHA and our method, GBS inputs monocular videos and reconstructs the head but excluding the clothing and shoulders. To ensure a fair comparison, we use multi-view videos and the FLAME parameters tracked following GA to serve as input of GBS, and segment the clothing area using face parsing, excluding it from the quantitative metrics. We employ PSNR, SSIM, and LPIPS [41] as quantitative metrics while listing the required storage and FPS, which can be found in Table 1. The storage size does not include pre-tracked 3DMM parameters. The FPS is tested on NVIDIA RTX4090 GPU. Our method occupies the minimum storage, ensuring real-time rendering speed, while also improving PSNR metrics on the tasks of novel view synthesis and self-reenactment.

Fig. 3 demonstrates the effectiveness of our method on novel view synthesis task. Compared to GA and GHA, our approach better reconstructs dynamic textures generated by significant expressions, such as the forehead wrinkles in the first row and the wrinkles around the right eye in the second row. GBS linearly interpolates all Gaussian attributes using

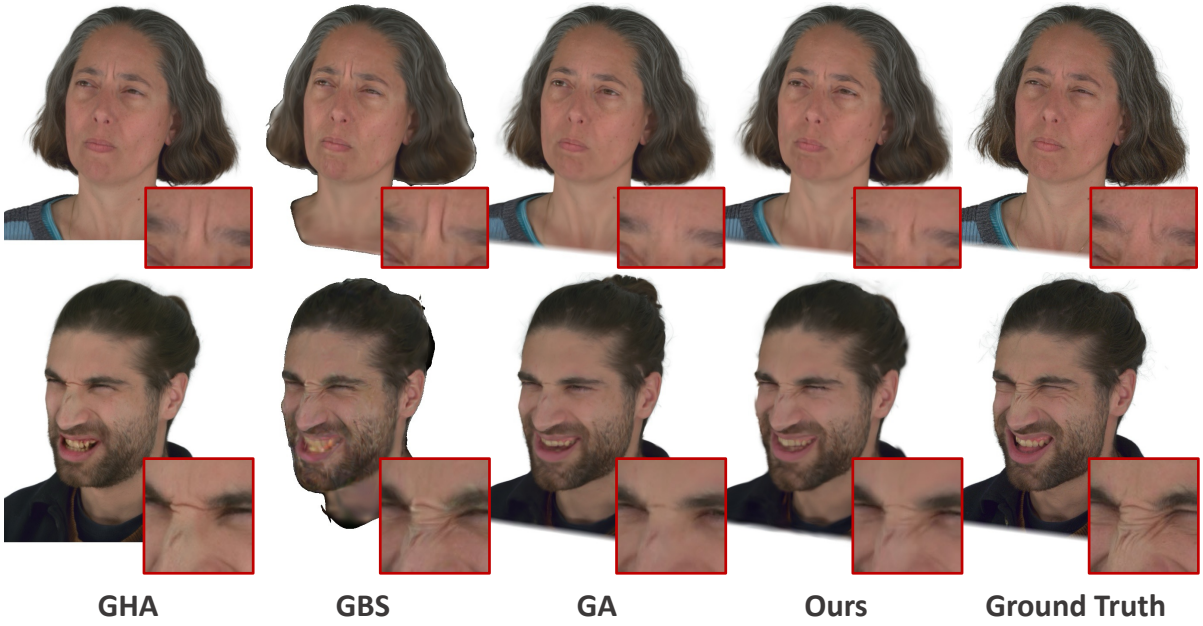


Figure 4. Qualitative comparison with baseline methods on self-reenactment task.

Components				Novel View Synthesis			Self-Reenactment		
trip	fl	penalty	resample	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
			✓	33.4206	0.9684	0.02968	31.2487	0.9561	0.03578
✓			✓	33.4337	0.9687	0.03090	31.3246	0.9585	0.03792
	✓	✓	✓	35.3876	0.9739	0.02497	31.4366	0.9577	0.03564
✓	✓	✓		35.2652	0.9737	0.02590	31.4288	0.9580	0.03585
✓	✓		✓	35.5509	0.9734	0.02597	31.4042	0.9580	0.03790
✓	✓	✓	✓	35.1644	0.9715	0.02605	31.6446	0.9591	0.03580

Table 2. Ablation Study on subject #306. "trip" and "fl" refer to neutral texture triplane and feature line blendshapes respectively. "penalty" indicates opacity offset penalty. "resample" indicates class-balanced sampling. GREEN indicates the best and YELLOW indicates the second.

blendshape coefficients, which can result in artifacts such as the left chin in the second row where exists large-scale non-linear motion and neck in the first row. In contrast, our method provides more stable modeling while maintaining comparable dynamic details. Noted that GBS requires 2GB storage, whereas ours requires only 10MB.

Fig. 4 shows the qualitative comparison results for the self-reenactment task. Our method enables avatar rendering with dynamic textures and less artifacts while avoiding overfitting of the wrinkles, such as the frown lines in the first row of Fig. 4. Fig. 5 illustrates the performance of our method in the cross-reenactment task, showcasing the generation of distinct wrinkle effects and identity preservation.

4.4. Ablation Study

We conduct ablations on subject #306 to evaluate the proposed components. Noted that novel view synthesis task involves novel view on training expressions, while self-reenactment involves novel expressions on 15 training views and one novel view.

Tensorial representations. In Table 2, the first two rows show compact triplanes help avoid overfitting, improving metrics on unseen expressions, while it may blur high-frequency details, leading to a worse LPIPS score. The first and third rows show that involving feature lines to model dynamic textures enhances both novel view and novel expressions rendering.

Class-balanced Sampling. We conduct ablations on the class-balanced sampling, as shown in the 4th and 6th rows of Table 2. Frames with large expressions are less frequent in the training data compared to frames with small motion. The sampling prevents the model from overfitting to relatively static frames, thereby improving generalization

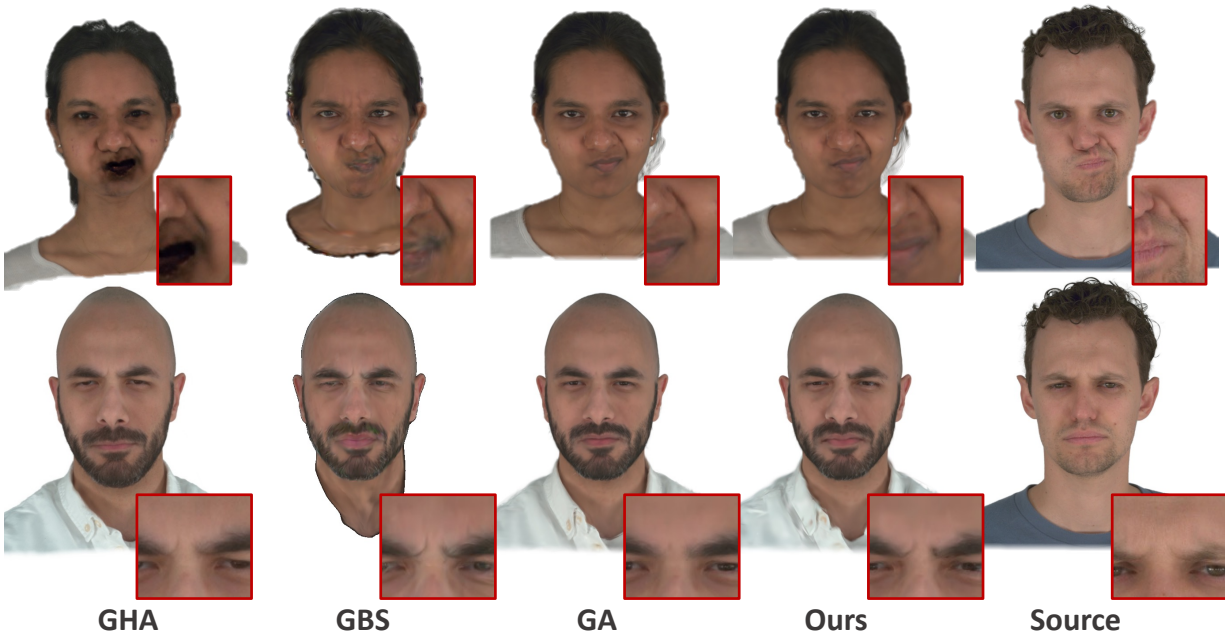


Figure 5. Cross-identity reenactment of head avatars. We use the expression and pose of the source subject on the far right to drive the character on the left.

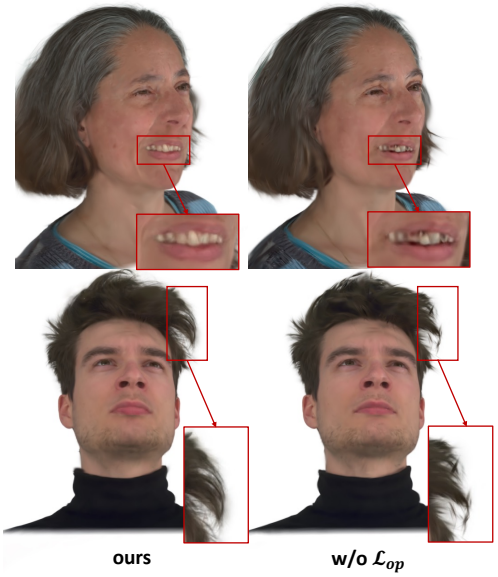


Figure 6. L_{op} helps prevent artifacts in self-reenactment.

to unknown expressions distribution, achieving better self-reenactment performance.

Adaptive truncated opacity offset penalty. Ablation studies are conducted to evaluate the adaptive truncated opacity offset penalty, as shown in 5th and 6th rows of Table 2. This penalty helps disentangle canonical appearance with dynamic details, thus improving robustness on unseen expressions. Moreover, without this penalty, floaters may appear around the hair, and artifacts may occur in the teeth area during expression changes, as shown in Fig. 6.

4.5. Conclusion and Limitations

In this paper, we propose a 3DGS head avatar modeling method that balances dynamic details capturing with real-time performance and low storage. Compact tensorial features (2D triplane for canonical appearance, and lightweight 1D feature lines for dynamic details) allow for accurate appearance modeling. Adaptive opacity offset penalty and class-balanced training helps prevent from overfitting to the training expressions. Experiments demonstrate that our approach not only enhances the rendering quality but also maintains real-time rendering speed and minimal storage, making it suitable for a wide range of practical applications.

Our method has some limitations, such as reliance on the tracked mesh thus unable to handle complex hairstyles or topological changes in the mouth interior. Our method do not decouple material and lighting, so avatar relighting is not feasible, which we intend to explore in the future.

Acknowledgements

This work was supported by National Natural Science Foundation of China (72192821, 62302297 and 62472282), Shanghai Sailing Program(22YF1420300), Young Elite Scientists Sponsorship Program by CAST (2022QNRC001), YuCaiKe [2023](231111310300) and the Fundamental Research Funds for the Central Universities(YG2023QNA35). This work was also supported by AntGroup Research Intern Program.

References

[1] Yunpeng Bai, Yanbo Fan, Xuan Wang, Yong Zhang, Jingxiang Sun, Chun Yuan, and Ying Shan. High-fidelity facial avatar reconstruction from monocular video with generative priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4541–4551, 2023. 2

[2] Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3d faces. 1999. 1, 2

[3] Eric R. Chan, Connor Z. Lin, Matthew A. Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas Guibas, Jonathan Tremblay, Sameh Khamis, Tero Karras, and Gordon Wetzstein. Efficient geometry-aware 3D generative adversarial networks. In *CVPR*, 2022. 2

[4] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *European conference on computer vision*, pages 333–350. Springer, 2022. 2

[5] Chuhan Chen, Matthew O'Toole, Gaurav Bharaj, and Pablo Garrido. Implicit neural head synthesis via controllable local deformation fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 416–426, 2023. 2

[6] Yufan Chen, Lizhen Wang, Qijing Li, Hongjiang Xiao, Shengping Zhang, Hongxun Yao, and Yebin Liu. Monogaussianavatar: Monocular gaussian point-based head avatar. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–9, 2024. 2

[7] Xuangeng Chu and Tatsuya Harada. Generalizable and animatable gaussian head avatar. *arXiv preprint arXiv:2410.07971*, 2024. 2

[8] Helisa Dhamo, Yinyu Nie, Arthur Moreau, Jifei Song, Richard Shaw, Yiren Zhou, and Eduardo Pérez-Pellitero. Headgas: Real-time animatable head avatars via 3d gaussian splatting. In *European Conference on Computer Vision*, pages 459–476. Springer, 2025. 2

[9] Sara Fridovich-Keil, Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. K-planes: Explicit radiance fields in space, time, and appearance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12479–12488, 2023. 2

[10] Guy Gafni, Justus Thies, Michael Zollhofer, and Matthias Nießner. Dynamic neural radiance fields for monocular 4d facial avatar reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8649–8658, 2021. 2

[11] Xuan Gao, Chenglai Zhong, Jun Xiang, Yang Hong, Yudong Guo, and Juyong Zhang. Reconstructing personalized semantic facial nerf models from monocular video. *ACM Transactions on Graphics (TOG)*, 41(6):1–12, 2022. 2

[12] Simon Giebenhain, Tobias Kirschstein, Markos Georgopoulos, Martin Rünz, Lourdes Agapito, and Matthias Nießner. Learning neural parametric head models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21003–21012, 2023. 2

[13] Simon Giebenhain, Tobias Kirschstein, Martin Rünz, Lourdes Agapito, and Matthias Nießner. Npga: Neural parametric gaussian avatars. *arXiv preprint arXiv:2405.19331*, 2024. 2

[14] Yang Hong, Bo Peng, Haiyao Xiao, Ligang Liu, and Juyong Zhang. Headnerf: A real-time nerf-based parametric head model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20374–20384, 2022. 2

[15] Yi-Hua Huang, Yang-Tian Sun, Ziyi Yang, Xiaoyang Lyu, Yan-Pei Cao, and Xiaojuan Qi. Sc-gs: Sparse-controlled gaussian splatting for editable dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4220–4230, 2024. 2

[16] Du Q Huynh. Metrics for 3d rotations: Comparison and analysis. *Journal of Mathematical Imaging and Vision*, 2009. 4

[17] Yuheng Jiang, Zhehao Shen, Penghao Wang, Zhuo Su, Yu Hong, Yingliang Zhang, Jingyi Yu, and Lan Xu. Hifi4g: High-fidelity human performance rendering via compact gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19734–19745, 2024. 2

[18] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023. 1, 2, 3

[19] Tobias Kirschstein, Shenhan Qian, Simon Giebenhain, Tim Walter, and Matthias Nießner. Nersemble: Multi-view radiance field reconstruction of human heads. *ACM Transactions on Graphics (TOG)*, 42(4):1–14, 2023. 2

[20] Joo Chan Lee, Daniel Rho, Xiangyu Sun, Jong Hwan Ko, and Eunbyung Park. Compact 3d gaussian representation for radiance field. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21719–21728, 2024. 2

[21] Tianye Li, Timo Bolkart, Michael J Black, Hao Li, and Javier Romero. Learning a model of facial shape and expression from 4d scans. *ACM Trans. Graph.*, 36(6):194–1, 2017. 3

[22] Zhe Li, Zerong Zheng, Yuxiao Liu, Boyao Zhou, and Yebin Liu. Posevocab: Learning joint-structured pose embeddings for human avatar modeling. In *ACM SIGGRAPH 2023 Conference Proceedings*, pages 1–11, 2023. 4

[23] Zhan Li, Zhang Chen, Zhong Li, and Yi Xu. Spacetime gaussian feature splatting for real-time dynamic view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8508–8520, 2024. 2

[24] Shengjie Ma, Yanlin Weng, Tianjia Shao, and Kun Zhou. 3d gaussian blendshapes for head avatar animation. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–10, 2024. 2, 6

[25] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. 1, 2

[26] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM transactions on graphics (TOG)*, 41(4):1–15, 2022. 2

[27] KL Navaneet, Kossar Pourahmadi Meibodi, Soroush Abbasi Koohpayegani, and Hamed Pirsiavash. Compact3d: Compressing gaussian splat radiance field models with vector quantization. *arXiv preprint arXiv:2311.18159*, 2023. 2

[28] Simon Niedermayr, Josef Stumpfegger, and Rüdiger Westermann. Compressed 3d gaussian splatting for accelerated novel view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10349–10358, 2024. 2

[29] Shenhan Qian, Tobias Kirschstein, Liam Schoneveld, Davide Davoli, Simon Giebenhain, and Matthias Nießner. Gaussianavatars: Photorealistic head avatars with rigged 3d gaussians. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20299–20309, 2024. 2, 3, 5, 6

[30] Shunsuke Saito, Gabriel Schwartz, Tomas Simon, Junxuan Li, and Giljoo Nam. Relightable gaussian codec avatars. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 130–141, 2024. 2

[31] Ruizhi Shao, Zerong Zheng, Hanzhang Tu, Boning Liu, Hongwen Zhang, and Yebin Liu. Tensor4d: Efficient neural 4d decomposition for high-fidelity dynamic reconstruction and rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16632–16642, 2023. 2

[32] Justus Thies, Michael Zollhofer, Marc Stamminger, Christian Theobalt, and Matthias Nießner. Face2face: Real-time face capture and reenactment of rgb videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2387–2395, 2016. 1, 2

[33] Chaoyang Wang, Ben Eckart, Simon Lucey, and Orazio Gallo. Neural trajectory fields for dynamic novel view synthesis. *arXiv preprint arXiv:2105.05994*, 2021. 2

[34] Jun Xiang, Xuan Gao, Yudong Guo, and Juyong Zhang. Flashavatar: High-fidelity head avatar with efficient gaussian embedding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1802–1812, 2024. 2

[35] Yuelang Xu, Lizhen Wang, Xiaochen Zhao, Hongwen Zhang, and Yebin Liu. Avatarmav: Fast 3d head avatar reconstruction using motion-aware neural voxels. In *ACM SIGGRAPH 2023 Conference Proceedings*, pages 1–10, 2023. 2

[36] Yingyan Xu, Prashanth Chandran, Sebastian Weiss, Markus Gross, Gaspard Zoss, and Derek Bradley. Artist-friendly relightable and animatable neural heads. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2457–2467, 2024. 2

[37] Yuelang Xu, Benwang Chen, Zhe Li, Hongwen Zhang, Lizhen Wang, Zerong Zheng, and Yebin Liu. Gaussian head avatar: Ultra high-fidelity head avatar via dynamic gaussians. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1931–1941, 2024. 2, 6

[38] Yuelang Xu, Lizhen Wang, Zerong Zheng, Zhaoqi Su, and Yebin Liu. 3d gaussian parametric head model. In *European Conference on Computer Vision*, pages 129–147. Springer, 2025. 2

[39] Bangbang Yang, Yinda Zhang, Yinghao Xu, Yijin Li, Han Zhou, Hujun Bao, Guofeng Zhang, and Zhaopeng Cui. Learning object-compositional neural radiance field for editable scene rendering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13779–13788, 2021. 2

[40] Wangbo Yu, Yanbo Fan, Yong Zhang, Xuan Wang, Fei Yin, Yunpeng Bai, Yan-Pei Cao, Ying Shan, Yang Wu, Zhongqian Sun, et al. Nofa: Nerf-based one-shot facial avatar reconstruction. In *ACM SIGGRAPH 2023 conference proceedings*, pages 1–12, 2023. 2

[41] 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. 6

[42] Xiaozheng Zheng, Chao Wen, Zhaohu Li, Weiyi Zhang, Zhuo Su, Xu Chang, Yang Zhao, Zheng Lv, Xiaoyuan Zhang, Yongjie Zhang, et al. Headgap: Few-shot 3d head avatar via generalizable gaussian priors. *arXiv preprint arXiv:2408.06019*, 2024. 2

[43] Yiyu Zhuang, Hao Zhu, Xusen Sun, and Xun Cao. Mofanerf: Morphable facial neural radiance field. In *European conference on computer vision*, pages 268–285. Springer, 2022. 2

[44] Wojciech Zielonka, Timo Bolkart, and Justus Thies. Towards metrical reconstruction of human faces. In *European conference on computer vision*, pages 250–269. Springer, 2022. 1

[45] Wojciech Zielonka, Timo Bolkart, and Justus Thies. Instant volumetric head avatars. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4574–4584, 2023. 2

[46] Zi-Xin Zou, Zhipeng Yu, Yuan-Chen Guo, Yangguang Li, Ding Liang, Yan-Pei Cao, and Song-Hai Zhang. Triplane meets gaussian splatting: Fast and generalizable single-view 3d reconstruction with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10324–10335, 2024. 2