MixedGaussianAvatar: Realistically and Geometrically Accurate Head Avatar via Mixed 2D-3D Gaussians

Peng Chen[1,2], **Xiaobao Wei**[1,2], **Qingpo Wuwu**[6], **Xinyi Wang**[4], **Xingyu Xiao**[3], **Ming Lu**[5†]

[1]Institute of Software, Chinese Academy of Sciences
[2]University of Chinese Academy of Sciences
[3]Tsinghua University [4]Nankai University
[5]Intel Labs China [6]Peking University
chenpeng.cp0225@gmail.com

Abstract

Reconstructing high-fidelity 3D head avatars is crucial in various applications such as virtual reality. The pioneering methods reconstruct realistic head avatars with Neural Radiance Fields (NeRF), which have been limited by training and rendering speed. Recent methods based on 3D Gaussian Splatting (3DGS) significantly improve the efficiency of training and rendering. However, the surface inconsistency of 3DGS results in subpar geometric accuracy; later, 2DGS uses 2D surfels to enhance geometric accuracy at the expense of rendering fidelity. To leverage the benefits of both 2DGS and 3DGS, we propose a novel method named MixedGaussianAvatar for realistically and geometrically accurate head avatar reconstruction. Our main idea is to utilize 2D Gaussians to reconstruct the surface of the 3D head, ensuring geometric accuracy. We attach the 2D Gaussians to the triangular mesh of the FLAME model and connect additional 3D Gaussians to those 2D Gaussians where the rendering quality of 2DGS is inadequate, creating a mixed 2D-3D Gaussian representation. These 2D-3D Gaussians can then be animated using FLAME parameters. We further introduce a progressive training strategy that first trains the 2D Gaussians and then fine-tunes the mixed 2D-3D Gaussians. We demonstrate the superiority of MixedGaussianAvatar through comprehensive experiments. The code will be released.

1. Introduction

Reconstructing high-quality 3D head avatars from multiple 2D images captured from different viewpoints is essential for various applications [2, 5, 24]. The key high-fidelity aspects include the geometric and realistic accuracy of 3D head avatars and rendered images. With the advancement of deep learning, techniques based on Neural Radiance Fields (NeRF) [15] and 3D Gaussian Splatting (3DGS) [12] have gained popularity, providing significant benefits for creating high-fidelity 3D head avatars.

NeRF introduced the idea of radiance fields, utilizing neural networks to store 3D information, which allows for reconstructing high-quality static scenes. When it comes to 3D head avatar reconstruction, the proposed methods primarily focus on reconstructing dynamic head avatars [6, 33]. For example, Neural Head Avatars (NHA) [7] process RGB video inputs that display various expressions and perspectives. NHA involves estimating and refining the low-dimensional shape, expression, and pose parameters of the FLAME head model from an RGB input frame, which allows for creating a neural head avatar that can be animated and rendered from novel viewpoints. However, the head avatars generated using NeRF-based methods have lower accuracy due to the implicit representation characteristics of NeRF, and the training and rendering processes are slow.

Recently, 3DGS effectively modeled a static scene using 3D Gaussians and rendered it through a differentiable rasterizer, greatly speeding up the 3D reconstruction process from multi-view RGB images. Because of 3DGS's explicit representation and efficient rasterization properties, it achieves high rendering quality and fast training and inference speeds. In 3D head avatar reconstruction, most methods attach 3D Gaussians to the FLAME triangular mesh to create dynamic facial expressions. For example, GaussianAvatars [20] employ a rigging approach to align the local positions, rotations, and scales of 3D Gaussians with the global parameters. FlashAvatar [26] and Gaussian Head Avatar [28] utilize neural networks to predict 3D Gaussian parameters. The advantage of these methods lies in their

†Corresponding Author.

Figure 1. MixedGaussianAvatar uses a mixed 2D-3D Gaussian Splatting method to reconstruct a realistically and geometrically accurate 3D head avatar mesh.

ability to dynamically represent 3D head avatars while producing high-quality images. However, due to the inherent multi-view inconsistency of 3DGS, it cannot accurately reconstruct geometric surfaces.

2D Gaussian Splatting [10] replaces 3D Gaussians with 2D Gaussians and employs a ray-splat intersection method for the splatting process, ensuring consistency across multiple views. The advantage lies in the high quality of reconstructed geometric surfaces, but it compromises the quality of rendered images. To summarize 3DGS and 2DGS, 1) 3DGS has achieved realistic results in novel view synthesis, but it struggles to capture accurate geometric structures. 2) 2DGS uses 2D surfels to accurately reconstruct geometric surfaces, but it cannot render realistic images as effectively as 3DGS.

To harness the advantages of both 3DGS and 2DGS, we introduce an innovative method called MixedGaussianAvatar. This method uses a mixed 2D-3D Gaussians representation, combining the color rendering advantages of 3DGS with the geometric reconstruction strengths of 2DGS to achieve a realistically and geometrically accurate 3D head avatar reconstruction. As shown in Fig. 1, our primary goal is to use 2DGS to preserve the geometric accuracy of the 3D head surface. We attach 2D Gaussians to the triangular mesh of the FLAME model and then connect additional 3D Gaussians to the corresponding 2D Gaussians in areas where the rendering quality is insufficient. The mixed 2D-3D Gaussians can be driven using FLAME parameters to form a dynamic 3D representation. To train the mixed 2D-3D Gaussians, we further propose a progressive training strategy that first trains the 2D Gaussians and then fine-tunes the mixed 2D-3D Gaussians. Our contributions are summarized as follows:

- We introduce MixedGaussianAvatar, an innovative approach that combines the rendering benefits of 3DGS with the reconstruction strengths of 2DGS, enabling the realistically and geometrically accurate reconstruction of 3D head avatars.
- We present a progressive training strategy to train mixed 2D-3D Gaussians for dynamic 3D head avatars.
- We conduct extensive experiments that show MixedGaussianAvatar achieves state-of-the-art color rendering and geometric reconstruction results.

2. Related Work

Dynamic Neural Fields The representation of dynamic scenes has attracted considerable attention. Previous methods, like NeRF [15] and its variants [3, 17, 23], focus on modeling static scenes, storing them in learnable neural networks to achieve competitive novel view synthesis quality. Recently, 3D Gaussian Splatting (3DGS) [12] and its variants [14] have introduced a much faster training and rendering pipeline. The methods mentioned above are only effective for modeling static scenes and have difficulty handling dynamic scenes. Based on NeRF and 3DGS, several approaches [11, 21, 25] introduce 4D coordinates to effectively represent dynamic scenes. Humans are very sensitive to facial details, making it challenging for general methods to create high-fidelity head avatars. Recently, numerous efforts [4, 7, 9] have been made to develop specialized techniques for representing 3D head avatars. AvatarMAV [27] and INSTA [34], both based on voxel representations, enable fast avatar rendering. PointAvatar [31] is the first to employ a deformable point-based representation for creating high-fidelity avatars. Inspired by 3DGS and PointAvatar, methods like GaussianAvatars [20], Gaussian Head Avatar [28], and FlashAvatar [26] attach 3D Gaussians to dynamic meshes or UV maps, achieving success in real-time 3D head avatar animation. Previous methods fail to consider geometric reconstruction, which makes them less appropriate for industrial applications. In contrast, MixedGaussianAvatar integrates the high-quality appearance of 3DGS with the consistent geometry of 2DGS.

Head Avatar Reconstruction. Compared to neural rendering, which primarily focuses on view synthesis, neural reconstruction emphasizes explicit surface extraction and textured mesh reconstruction. NeuS [22] is the first to propose a bias-free volume rendering method based on neural fields that leverages a signed distance function (SDF) to achieve high-quality surface reconstruction. UNISURF [18] presents a unified framework that combines neural implicit surfaces and radiance fields to enable accurate 3D surface reconstruction from multi-view images without requiring object masks. VolSDF [29] significantly improves geometry approximation and disentangles

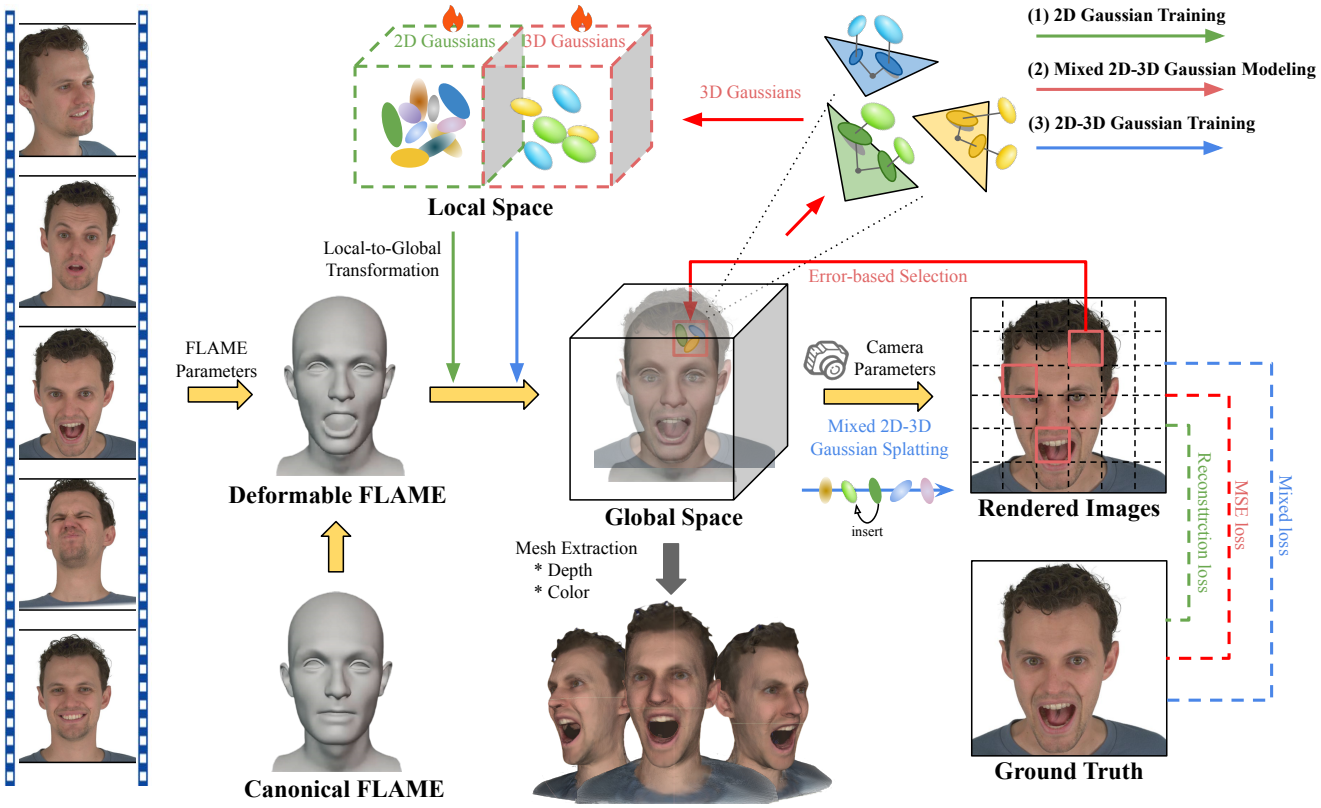

Figure 2. **Pipeline of MixedGaussianAvatar.** We propose a differentiable mixed 2D-3D Gaussian Splatting method for reconstructing realistic and geometrically accurate head avatars from multi-view 2D images. This approach uses 2D Gaussians for geometric precision and 3D Gaussians to correct color errors, resulting in high-quality 3D head mesh and realistic mesh textures or RGB images. The method is integrated with the FLAME model, enabling dynamic effects and animation driven by parameters through local-to-global transformation and the progressive training strategy.

shape and appearance compared to previous neural rendering methods. With the development of 3D Gaussian Splatting, SuGaR [8] introduces a method for fast and precise 3D mesh reconstruction and high-quality mesh rendering by aligning 3D Gaussian splats with scene surfaces, enabling efficient and detailed mesh extraction within minutes. 2D Gaussian Splatting (2DGS) [10] proposes a novel approach for geometrically accurate radiance field reconstruction by leveraging 2D Gaussian primitives, which significantly improves surface modeling and view-consistent rendering compared to 3D Gaussian splatting methods. These works focus on the general static object surface reconstruction. Within the head avatar reconstruction, NHA [7] presents a novel approach to reconstructing full human head geometry and photorealistic textures from a short monocular RGB video. Ming et al. [16] introduces a technique for reconstructing personalized, animation-ready mesh blendshapes from single or sparse multi-view videos using neural inverse rendering. VHAP [19] is a recent head mesh tracking tool, but it cannot render colors and textures.

To our knowledge, we are the first to utilize Gaussians to explicitly represent head avatar meshes. Compared to 3DGS-based methods like GaussianAvatars, which exhibit poorer surface reconstruction, our approach mixes the advantages of 2DGS and 3DGS to achieve an optimal balance between rendering and reconstruction quality.

3. Method

3.1. Preliminaries

3DGS represents 3D scenes using Gaussian primitives, where each primitive is described by a 3D position $\mathbf{p_k}$ and a 3D covariance matrix Σ. The Gaussian function is defined as:

$$G(\mathbf{p}) = \exp\left(-\frac{1}{2}(\mathbf{p}-\mathbf{p_k})^\top \Sigma^{-1}(\mathbf{p}-\mathbf{p_k})\right) \quad (1)$$

where the covariance matrix Σ is decomposed into a rotation matrix R and a scaling matrix S, given by $\Sigma = RSS^\top R^\top$. During the rendering process, these 3D Gaussian primitives are projected onto the image plane using a world-to-camera transformation matrix W and a local affine transformation matrix J. This projection yields projected 2D Gaussian primitives with a covariance matrix Σ^{2D}, derived from Σ after omitting its third row and column. Then we can obtain G^{3D} with Σ^{2D}. Finally, to render the scene, 3DGS uses volumetric alpha blending, where the contribu-

tions of each Gaussian primitives are combined sequentially from front to back to determine the color at a pixel $\mathbf{x}$:

$$\mathbf{c}(\mathbf{x}) = \sum_{k=1}^{K} \mathbf{c_k} \alpha_k G_k^{3D}(\mathbf{x}) \prod_{j=1}^{k-1} (1 - \alpha_j G_j^{3D}(\mathbf{x})) \quad (2)$$

where $\mathbf{c_k}$ represents the color, and α_k is the alpha value of each Gaussian primitive.

2DGS removes the third dimension of scaling from the 3D Gaussian primitives, turning them into 2D Gaussian primitives, which geometrically resemble discs. During the splatting process, 2DGS abandons the local affine transformation method used in 3DGS and introduces the ray-splat intersection method. This method directly computes the intersection position $\boldsymbol{u}(\boldsymbol{x})$ between the ray from the camera origin to the pixel and the 2D Gaussian primitives. The Gaussian value G^{2D} is then calculated using a low-pass filter, specifically $\max\left\{G\left(\mathbf{u}(\mathbf{x})\right), G\left(\frac{\mathbf{x}-\mathbf{c}}{\sigma}\right)\right\}$. The alpha blending formula is the same as in Eq. (2), with G^{3D} replaced by G^{2D}.

3.2. Overview

We propose a differentiable mixed 2D-3D Gaussian splatting method to accurately reconstruct a geometrically accurate head avatar from multi-view 2D images, as shown in Fig. 2. We begin by training 2D Gaussian representations of the 3D head avatar to ensure geometric accuracy. Next, we identify the 2D Gaussians for which the rendering quality of 2DGS is inadequate. For these identified 2D Gaussians, we bind additional 3D Gaussians to construct the mixed 2D-3D Gaussians in Sec. 3.3. We use a local-to-global transformation method to convert mixed 2D-3D Gaussians for creating dynamic 3D head avatars. Finally, we train the mixed 2D-3D Gaussians to create realistic and geometrically accurate head avatars. The progressive training strategy is introduced in Sec. 3.4.

3.3. Mixed 2D-3D Gaussian Avatar

3.3.1. Mixed 2D-3D Gaussian Modeling

We introduce mixed 2D-3D Gaussian modeling, utilizing the 2D Gaussian to accurately represent geometric surfaces and the 3D Gaussian to enhance color representation. Specifically, we first train the model using 2DGS. Initially, we assign a 2D Gaussian to each triangular mesh of FLAME. As the adaptive density control process progresses, more generated 2D Gaussians will be bound to the mesh triangles. After the 2DGS training, we take a rendered image $\hat{X}$ and a real image X and divide each of them into tiles of size $w \times h$. Each tile is represented as $\hat{x}_{ij}$ for the rendered image and x_{ij} for the real image, where i and j indicate the position indices of the tiles. The MSE loss value is defined as follows:

$$L_{ij} = \|\hat{x}_{ij} - x_{ij}\|_2^2 \quad (3)$$

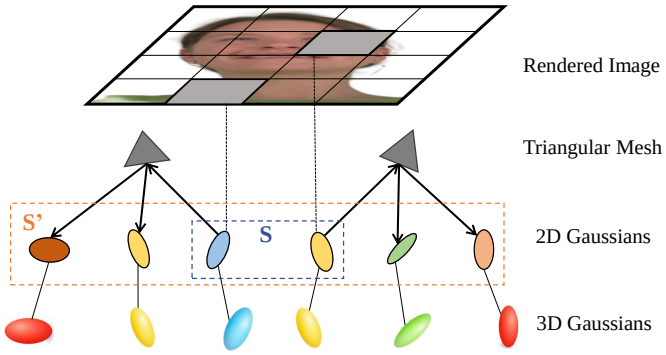

Figure 3. The tree structure of mixed 2D-3D Gaussians.

We select the top k tiles with the largest L_{ij} values and record the indices of the 2D Gaussians contributing to the colors of these tiles, forming a set S. Since a triangle consists of multiple bounded 2D Gaussians, a Gaussian tree structure is formed with the triangular mesh as the root node and the 2D Gaussians as its child nodes, as shown in Fig. 3. Based on the tree, for each 2D Gaussian in set S, we add its sibling nodes to collectively form a larger set S'. This process is repeated for n iterations, with each iteration using different FLAME parameters and camera parameters to generate 2D images from multiple angles and with various expressions. The set S' from each iteration is combined into a union set U. The whole pipeline is shown in Alg. 1.

$$U = \bigcup_{i=1}^{n} S_i' \quad (4)$$

To improve the rendering quality of 2D Gaussians, we add the additional 3D Gaussians to the scene based on the set U, as shown in Fig. 3. We treat each 2D Gaussian $g^{2D} \in U$ as a parent node and generate a corresponding 3D Gaussian g^{3D} as its child node. These 3D Gaussians inherit the position μ, spherical harmonics (SHs), opacity α, rotation r, and scaling s from corresponding parent g^{2D} in local space, with the third dimension of scaling initialized to an average value. In this way, we generate some 3D Gaussians in areas where the rendering quality of 2DGS is poor. No model parameter optimization is required at this stage.

3.3.2. Mixed 2D-3D Gaussian Splatting

The splatting methods of 3DGS and 2DGS are entirely different; however, both utilize the same alpha-blending approach. By combining these two splatting methods, we project 3D Gaussians onto image space using a local affine transformation to obtain Gaussian values, while 2D Gaussians are computed directly using the ray-splat intersection method.

During the rasterization process, for a given pixel $\mathbf{x}$ belonging to tile t, we first sort all the 2D Gaussians within tile t by depth and then perform alpha blending in order from

Algorithm 1 Error-based 2D Gaussian Selection

Require: Rendered image $\hat{X}$, Ground truth image X
Require: Number of tiles $w \times h$
Require: Number of iterations n
Require: Number of top tiles k
Require: Set of FLAME and camera parameters f, c
Initialize set $U \leftarrow \emptyset$
for iteration $= 1$ to n **do**
  Sample f and c
  Render image $\hat{X}$ using f and c
  Divide images $\hat{X}$ and X into $w \times h$ tiles
  Calculate MSE for each tile: $L_{ij} = \|\hat{x}_{ij} - x_{ij}\|_2^2$
  Select top k tiles with highest L_{ij} loss
  Record indices of 2D Gaussians contributing to these tiles in set S
  Based on the Gaussian tree, for each 2D Gaussian in set S, add its sibling nodes to form set S'
  Update $U \leftarrow U \cup S'$
end for
return U

front to back. Specifically, while traversing the 2D Gaussians, if the current 2D Gaussian g_k^{2D} has a 3D Gaussian g_j^{3D} child node, we insert the contribution of g_j^{3D} after that of g_k^{2D}, in order to emphasize the compensation of the 3D Gaussians to the color. The mixed alpha blending formula is as follows:

$$T^{2D} = \prod_{t=1}^{i-1} \left(1 - \alpha_t G_t^{2D}(\mathbf{x})\right) \tag{5}$$

$$\mathbf{c}_p^{2D}(\mathbf{x}) = \sum_{i=1}^{k} \mathbf{c_i} \alpha_i G_i^{2D}(\mathbf{x}) T^{2D} \tag{6}$$

$$\mathbf{c}^{3D}(\mathbf{x}) = \mathbf{c_j} \alpha_j G_j^{3D}(\mathbf{x}) \prod_{t=1}^{k} \left(1 - \alpha_t G_t^{2D}(\mathbf{x})\right) \tag{7}$$

$$\mathbf{c}_b^{2D}(\mathbf{x}) = \sum_{i=k+1}^{N} \mathbf{c_i} \alpha_i G_i^{2D}(\mathbf{x}) \left(1 - \alpha_j G_j^{3D}(\mathbf{x})\right) T^{2D} \tag{8}$$

where G^{2D} represents the Gaussian value of 2D Gaussian, and G^{3D} represents that of 3D Gaussian. $\mathbf{c}_p^{2D}(\mathbf{x})$ denotes the color contribution to pixel $\mathbf{x}$ from g_k^{2D} and all preceding 2D Gaussians, while $\mathbf{c}_b^{2D}(\mathbf{x})$ denotes the color contribution from the 2D Gaussians that follow g_k^{2D}. $\mathbf{c}^{3D}(\mathbf{x})$ represents the color contribution from g_j^{3D}. We add all color contribution together and obtain the final color of pixel $\mathbf{x}$.

$$\mathbf{c}^{mixed}(\mathbf{x}) = \mathbf{c}_p^{2D}(\mathbf{x}) + \mathbf{c}^{3D}(\mathbf{x}) + \mathbf{c}_b^{2D}(\mathbf{x}) \tag{9}$$

3.3.3. Local-to-Global Transformation

To create dynamic 3D head avatars, we propose a novel local-to-global transformation method designed for mixed 2D-3D Gaussians, inspired by the 3D Gaussian rigging approach in GaussianAvatars [20]. The core of the local-to-global transformation is to establish a mapping between the FLAME mesh and the mixed Gaussians. As shown in the Gaussian tree relation in Fig. 3, we associate each 2D Gaussian in local space with the triangular mesh in global space. Likewise, we connect each 3D Gaussian in local space to its corresponding 2D Gaussian. The FLAME parameters, such as expressions, enable the canonical FLAME to transform into a deformable FLAME. Through this mapping, the triangular mesh can adjust along with the mixed Gaussians, allowing these mixed Gaussians to be represented in a global space.

In local space, we initialize the position μ_l of the mixed Gaussians to 0, the rotation matrix r_l to the identity matrix, scaling s_l to the unit vector, and opacity α to 0. In global space, we define the mixed Gaussians' position as μ_g, rotation matrix as r_g, scaling as s_g, and opacity as α. For the transformation of mixed Gaussians through rotation and scaling, the formula is as follows:

$$r_g = Rr_l, s_g = \lambda s_l \tag{10}$$

where R is the rotation matrix formed by the direction vector of one edge of the triangle, the normal vector, and their cross-product. λ represents the average length of an edge of the triangle and its perpendicular, serving as a global position and size scaling factor.

For the mapping between the mesh and the 2D Gaussian, the transformation of positions is defined as follows:

$$\mu_g^{2D} = \lambda R \mu_l^{2D} + T + p_\theta^{2D} \tag{11}$$

where T denotes the average position of the three vertices of the corresponding triangle in the global space. We introduced a learnable perturbation term p_θ^{2D} to adaptively correct positional errors.

For the mapping between 2D and 3D Gaussians, we substituted T with μ^{2D} in the position transformation formula:

$$\mu_g^{3D} = \lambda R \mu_l^{3D} + \mu_g^{2D} + p_\theta^{3D} \tag{12}$$

3.4. Progressive Training Strategy

3.4.1. 2D Gaussian Training

At this stage, we will only use 2DGS to train the model, with the goal of achieving a highly accurate geometric surface. The loss functions in this stage align with those of GaussianAvatars and 2DGS. We calculate the L_1 loss between the rendered images and the ground truth, including a D-SSIM term. In this case, λ is set to 0.2.

$$L_{\text{rgb}} = (1 - \lambda) L_1 + \lambda L_{\text{D-SSIM}} \tag{13}$$

To ensure that the locations of the 2D Gaussians and the corresponding underlying mesh remain adequately aligned

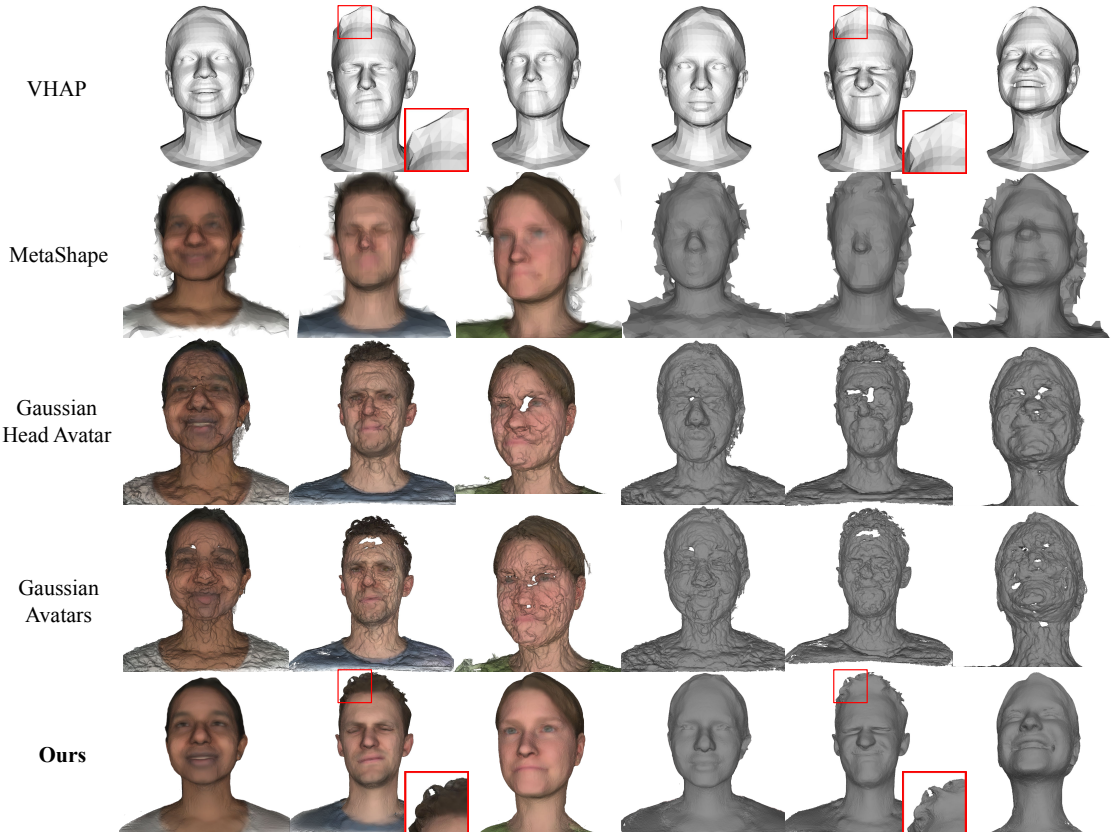

Figure 4. Qualitative comparison on mesh reconstruction.

in global space, we constrain the local position μ_l using L_{pos}. The threshold ϵ_{pos} is 1.

$$L_{\text{pos}} = \left\| \max\left(\mu_l^{2D}, \epsilon_{\text{pos}}\right) \right\|_2 \tag{14}$$

Likewise, the scaling of the 2D Gaussians should correspond to that of the triangular mesh. To prevent the local scaling s_l from becoming excessively large, we constrain it using L_{sca}, with ϵ_{sca} set to 0.6.

$$L_{\text{sca}} = \left\| \max\left(s_l^{2D}, \epsilon_{\text{sca}}\right) \right\|_2 \tag{15}$$

To improve the performance of 2DGS in geometric modeling, we utilize their depth distortion function L_{d} and normal consistency function L_{n} (details can be found in [10]). Therefore, the training loss in this stage is expressed as:

$$L_{\text{rec}} = L_{\text{c}} + \lambda_1 L_{\text{pos}} + \lambda_2 L_{\text{sca}} + \lambda_3 L_{\text{d}} + \lambda_4 L_{\text{n}} \tag{16}$$

3.4.2. Mixed 2D-3D Gaussian Training

After completing training for 2DGS in Sec. 3.4.1 and mixed 2D-3D Gaussian modeling in Sec. 3.3.1, the scene now includes 2D Gaussians distributed across the surface of the geometry and 3D Gaussians created for color compensation. During the mixed 2D-3D Gaussian training stage, we train all parameters of the 3D Gaussians as well as the color parameters (spherical harmonics) of the 2D Gaussians. However, mixed 2D-3D Gaussians require that each 3D Gaussian is positioned close to its corresponding parent 2D Gaussian in global space. This proximity ensures that when both are projected into image space, they can significantly contribute to the same tile's color. We propose a 2D-3D distance regularization term L_{dis} that limits the distance between 2D and 3D Gaussians to a specified threshold. ϵ_{dis} is 1.

$$L_{\text{dis}} = \left\| \max\left(\mu_l^{3D} + p_\theta^{3D}, \epsilon_{\text{dis}}\right) \right\|_2 \tag{17}$$

Finally, the training loss in this stage is expressed as:

$$L_{mixed} = L_{rgb} + \lambda_5 L_{dis} \tag{18}$$

where $\lambda_1 = \lambda_5 = 0.01$, $\lambda_2 = 1$, $\lambda_3 = 1000$, and $\lambda_4 = 0.05$.

4. Experiments

4.1. Experimental Settings

4.1.1. Dataset and Baselines

We use two challenging datasets: NeRSemble and INSTA. NeRSemble dataset [13] is used to conduct reconstruction and rendering experiments on 9 subjects, with each recording including 16 viewpoints. To ensure a fair comparison, we use the same dataset split as GaussianAvatars. INSTA dataset [34] is used for rendering experiments. It consists of monocular videos from ten subjects, with the final 350 frames of each sequence set aside for testing.

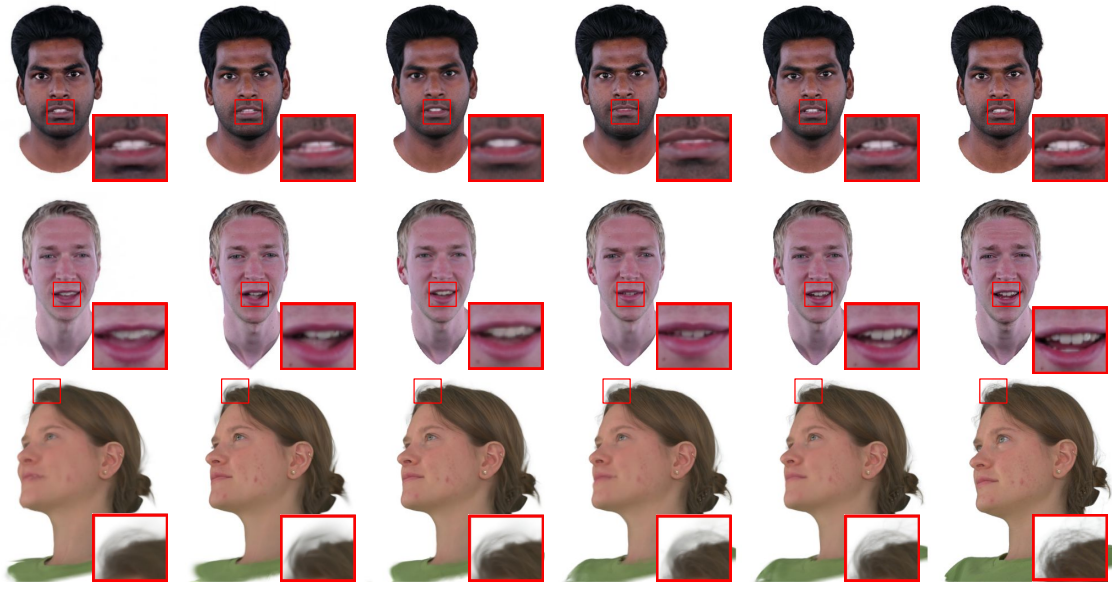

Figure 5. Qualitative comparison of image rendering.

For the NerSemble dataset, we selected NeRF-based methods, including PointAvatar [31], and INSTA [34]; and 3DGS-based methods, including FlashAvatar [26], Gaussian Head Avatar [28], and GaussianAvatars [20]. For the INSTA dataset, we selected NeRF-based methods, including NHA, IMAvatar [30], and INSTA; and 3DGS-based methods, including FlashAvatar, Gaussian Head Avatar, and GaussianAvatars. Additionally, we selected a 3D reconstruction industrial software, MetaShape [1], and a representative mesh tracking method, VHAP [19], as baselines.

4.1.2. Evaluation Metrics

Due to the absence of ground truth for 3D head meshes in the datasets, we are unable to utilize quantitative metrics to evaluate geometric reconstruction accuracy. Our method demonstrates significant advantages in the visual quality of geometric reconstruction compared to the baselines; therefore, we will evaluate it using visual comparison.

In terms of rendering, to assess the quality of the synthesized images, we utilized common metrics such as Mean Squared Error (L2), Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index (SSIM), and Learned Perceptual Image Patch Similarity (LPIPS).

4.1.3. Mesh Extraction and Texture Generation

To extract meshes from the reconstructed mixed Gaussians, we generate depth maps of the training views using the median depth values of the 2D Gaussians projected onto the pixels. To achieve a high-quality texture for the 3D head mesh, we utilize the RGB color from the mixed 2D-3D Gaussians projected onto the pixels. Then, we use Open3D [32] to apply truncated signed distance fusion (TSDF) to merge the depth maps for mesh extraction and use the RGB color to generate the texture. For a fair comparison, we apply the same mesh extraction and texture generation method to the 3DGS-based baselines.

4.2. Comparison Results

Comparison experiments are divided into quantitative and qualitative experiments. In our quantitative experiments, we assessed the quality of image rendering using two different datasets. In the qualitative experiments for reconstruction, we presented two visualization results: mesh without texture and mesh with texture. For rendering, we demonstrate the results of the generated RGB images.

4.2.1. Quantitative Evaluation

In this experiment, NeRSemble dataset was used to assess novel-view synthesis, while INSTA was used for evaluating self-reenactment. As shown in Tab. 1, it is clear that NeRF-based methods, such as PointAvatar and INSTA, generally yield lower image quality. In contrast, methods based on 3DGS tend to demonstrate stronger rendering capabilities. This observation highlights the inherent advantage of 3D Gaussians in color representation. FlashAvatar binds 3D Gaussians to the UV map, and controlling the number of Gaussians can lead to insufficient detail representation, resulting in lower rendering quality compared to other 3DGS-based methods. Tab. 2 presents the rendering results of our method on the NeRSemble dataset utilizing only 2D Gaussians. It is evident that, in comparison to methods based on 3DGS, the multi-view consistency of 2DGS greatly diminishes image quality. However, our method employs 3D

Table 1. Results on the NeRSemble [13] and INSTA [33] dataset. Green indicates the best and yellow indicates the second.

Method	dataset	L2 ↓	PSNR ↑	SSIM ↑	LPIPS ↓
PointAvatar [31]	NeRSemble	0.0020	25.9	0.887	0.096
INSTA [33]		0.0018	26.8	0.895	0.117
FlashAvatar [26]		0.0016	29.3	0.932	0.075
Gaussian Head Avatar [28]		0.0015	30.8	0.945	0.078
GaussianAvatars [20]		0.0013	31.4	0.941	0.064
(Ours)		0.0011	31.8	0.953	0.067
NHA [7]	INSTA	0.0024	27.0	0.942	0.043
IMAvatar [30]		0.0021	27.9	0.943	0.061
INSTA [33]		0.0017	28.6	0.944	0.047
FlashAvatar [26]		0.0017	29.2	0.949	0.040
Gaussian Head Avatar [28]		0.0016	29.7	0.958	0.043
GaussianAvatars [20]		0.0014	29.1	0.953	0.046
(Ours)		0.0012	30.4	0.962	0.039

Gaussians to overcome the limitations of 2D Gaussians in color rendering, resulting in higher-quality rendering than pure 2DGS. Our approach is highly competitive with 3DGS methods, but it significantly excels in geometric surface accuracy, outpacing 3DGS-based methods.

4.2.2. Qualitative Evaluation

For accurate mesh reconstruction, we assess visual results using the multi-view NeRSemble dataset, as our datasets lack ground truth for reconstructed meshes. We presented the results of visualizing 3D head avatar reconstruction using our method alongside representative baselines. As shown in Fig. 4, our reconstruction results emphasize the mesh and texture, depicted as mesh without and with texture, respectively. In the mesh w/o texture visualization, we mainly evaluate the accuracy of the geometric surface reconstruction. Methods based on 3DGS exhibit the poorest reconstruction quality, resulting in uneven mesh surfaces and significant noise. This is due to the inherent multi-view inconsistency of 3DGS and the contradiction between its three-dimensional shape and the two-dimensional mesh surface. The face reconstructed by MetaShape is blurry, and it produces many irrelevant floating artifacts. VHAP is the latest head mesh tracking tool, but it cannot model details such as hair strands, as shown in the red boxes in Fig. 4. Moreover, it has fewer mesh faces. Our approach creates detailed and noise-free surfaces by utilizing 2D Gaussians to preserve the 3D head surface. In the mesh w/ texture visualization, MetaShape and methods based on 3DGS struggle with texture color accuracy due to significant errors in the geometric surface reconstruction. VHAP cannot render colors, which is another limitation compared to our method.

Fig. 5 offers a qualitative comparison of the latest baselines for more accurate rendering. We have highlighted the comparative details using red boxes. Our method significantly outperforms INSTA in terms of facial detail representation. Mixed 2D-3D Gaussians show strong competitiveness compared to methods based on 3DGS, and in some cases, they even outperform these methods.

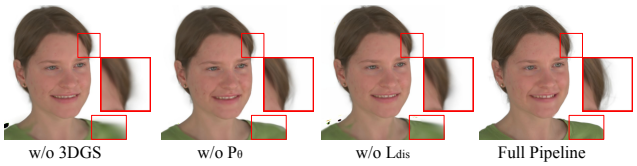

Figure 6. Qualitative ablation study on the NeRSemble dataset.

Table 2. Quantitative ablation study on NeRSemble dataset.

Method	L2 ↓	PSNR ↑	SSIM ↑	LPIPS ↓
Ours w/o 3DGS	0.0016	27.8	0.943	0.079
Ours w/o p_θ	0.0013	30.9	0.945	0.072
Ours w/o L_{dis}	0.0013	29.7	0.946	0.078
Our full pipeline	**0.0011**	**31.8**	**0.953**	**0.067**

4.3. Ablation Study

In this section, we conducted an ablation study by component to evaluate the role of each element on the NeRSemble dataset. We trained the model using only 2DGS for all iterations without using 3DGS, referred to as "w/o 3DGS," to evaluate the contribution of 3DGS to RGB rendering. We removed the perturbation term p_θ in local-to-global transformation, denoted as "w/o p_θ." Additionally, we removed the 2D-3D distance loss L_{dis}, denoted as "w/o L_{dis}".

The results are shown in Tab. 2 and Fig. 6. the absence of all components results in a decrease in rendering quality, indicating the effectiveness of these components. Removing 3D Gaussians resulted in a significant drop in rendering quality, demonstrating that mixed Gaussians have a considerable effect in maintaining high-precision color rendering. The learnable p_θ enables a trade-off between 2D Gaussians moving towards or away from the triangular mesh, allowing for adaptive correction of spatial positioning errors. The loss L_{dis} maintains the distance between 2D Gaussians and their corresponding 3D Gaussians in global space, ensuring that the projection of the 3D Gaussians after local affine

transformation lies within the same tile as the 2D splats, thereby compensating for color errors to the greatest extent.

5. Conclusion

In this work, we introduce MixedGaussianAvatar, a new method that employs mixed 2D-3D Gaussian Splatting for creating head avatars. We use 2DGS to maintain the surface geometry and employ 3DGS for color correction in areas where the rendering quality of 2DGS is insufficient, reconstructing a realistically and geometrically accurate 3D head avatar. We employ a local-to-global transformation technique to reconstruct dynamic head avatars and adopt a progressive training strategy to train mixed 2D-3D Gaussians. Our method demonstrates exceptional performance in both mesh reconstruction and texture rendering, setting a new standard in the field.

References

[1] Agisoft. Metashape professional (software), 2023. Accessed: 2023-11-16. 7

[2] Zechen Bai, Peng Chen, Xiaolan Peng, Lu Liu, Naiming Yao, and Hui Chen. Bring your own character: A holistic solution for automatic facial animation generation of customized characters. In *2024 IEEE Conference Virtual Reality and 3D User Interfaces (VR)*, pages 429–438. IEEE, 2024. 1

[3] 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

[4] Peng Chen, Xiaobao Wei, Ming Lu, Yitong Zhu, Naiming Yao, Xingyu Xiao, and Hui Chen. Diffusiontalker: Personalization and acceleration for speech-driven 3d face diffuser. *arXiv preprint arXiv:2311.16565*, 2023. 2

[5] Gérard Chollet, Anna Esposito, Annie Gentes, Patrick Horain, Walid Karam, Zhenbo Li, Catherine Pelachaud, Patrick Perrot, Dijana Petrovska-Delacrétaz, Dianle Zhou, et al. Multimodal human machine interactions in virtual and augmented reality. *Multimodal Signals: Cognitive and Algorithmic Issues: COST Action 2102 and euCognition International School Vietri sul Mare, Italy, April 21-26, 2008 Revised Selected and Invited Papers*, pages 1–23, 2009. 1

[6] 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 (Proceedings of SIGGRAPH Asia)*, 41(6), 2022. 1

[7] Philip-William Grassal, Malte Prinzler, Titus Leistner, Carsten Rother, Matthias Nießner, and Justus Thies. Neural head avatars from monocular rgb videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18653–18664, 2022. 1, 2, 3, 8

[8] Antoine Guédon and Vincent Lepetit. Sugar: Surface-aligned gaussian splatting for efficient 3d mesh reconstruction and high-quality mesh rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5354–5363, 2024. 3

[9] 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

[10] Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2d gaussian splatting for geometrically accurate radiance fields. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–11, 2024. 2, 3, 6, 1

[11] Nan Huang, Xiaobao Wei, Wenzhao Zheng, Pengju An, Ming Lu, Wei Zhan, Masayoshi Tomizuka, Kurt Keutzer, and Shanghang Zhang. S3gaussian: Self-supervised street gaussians for autonomous driving. *arXiv preprint arXiv:2405.20323*, 2024. 2

[12] 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

[13] 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. 6, 8, 4

[14] Tao Lu, Mulin Yu, Linning Xu, Yuanbo Xiangli, Limin Wang, Dahua Lin, and Bo Dai. Scaffold-gs: Structured 3d gaussians for view-adaptive rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20654–20664, 2024. 2

[15] 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

[16] Xin Ming, Jiawei Li, Jingwang Ling, Libo Zhang, and Feng Xu. High-quality mesh blendshape generation from face videos via neural inverse rendering. *arXiv preprint arXiv:2401.08398*, 2024. 3

[17] 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

[18] Michael Oechsle, Songyou Peng, and Andreas Geiger. Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5589–5599, 2021. 2

[19] Shenhan Qian. Versatile head alignment with adaptive appearance priors. 2024. 3, 7, 4

[20] 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. 1, 2, 5, 7, 8, 4

[21] 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

[22] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *arXiv preprint arXiv:2106.10689*, 2021. 2

[23] Xiaobao Wei, Renrui Zhang, Jiarui Wu, Jiaming Liu, Ming Lu, Yandong Guo, and Shanghang Zhang. Nto3d: Neural target object 3d reconstruction with segment anything. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20352–20362, 2024. 2

[24] Isabell Wohlgenannt, Alexander Simons, and Stefan Stieglitz. Virtual reality. *Business & Information Systems Engineering*, 62:455–461, 2020. 1

[25] Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4d gaussian splatting for real-time dynamic scene rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20310–20320, 2024. 2

[26] 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. 1, 2, 7, 8

[27] 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

[28] 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. 1, 2, 7, 8

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

[30] Yufeng Zheng, Victoria Fernández Abrevaya, Marcel C Bühler, Xu Chen, Michael J Black, and Otmar Hilliges. Im avatar: Implicit morphable head avatars from videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13545–13555, 2022. 7, 8

[31] Yufeng Zheng, Wang Yifan, Gordon Wetzstein, Michael J Black, and Otmar Hilliges. Pointavatar: Deformable point-based head avatars from videos. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 21057–21067, 2023. 2, 7, 8

[32] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3d: A modern library for 3d data processing. *arXiv preprint arXiv:1801.09847*, 2018. 7

[33] Wojciech Zielonka, Timo Bolkart, and Justus Thies. Instant volumetric head avatars. *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4574–4584, 2022. 1, 8

[34] 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, 6, 7, 4

MixedGaussianAvatar: Realistically and Geometrically Accurate Head Avatar via Mixed 2D-3D Gaussians

Supplementary Material

A. Overview

The supplementary material encompasses the following components. Please visit the anonymous website https://mixedgaussianavatar.github.io/ for visualized dynamic meshes of 3D head avatars.

- Supplementary Visualization Results
  - Mesh w/ Texture
  - Mesh w/o Texture
- Implementation Details
  - CUDA Implementation
  - Training Details
  - Dataset Details
- Limitaions and Prospects
- Ethical Consideration
- Demo Video

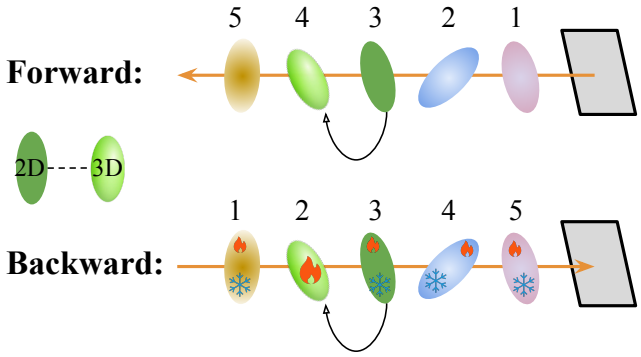

Figure 1. The CUDA implementation of mixed 2D-3D Gaussian Splatting.

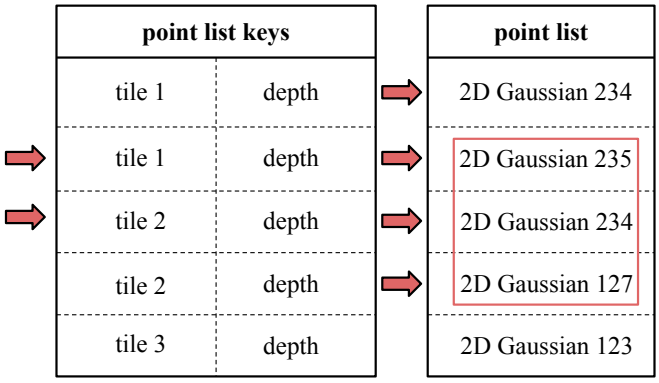

Figure 2. The CUDA implementation of error-based selection.

B. Supplementary Visualization Results

We have supplemented additional visualization results to demonstrate the realistically and geometrically accurate 3D head avatar reconstruction.

Mesh w/ Texture. In this section, we present the visualization results of the 3D head avatar mesh with texture generated using our method, as shown in Fig. 3.

Mesh w/o Texture. In this section, we only present the visualization results of the 3D head avatar mesh generated using our method, as shown in Fig. 4.

C. Implementation Details

CUDA Implementation We have implemented mixed 2D-3D Gaussian Splatting with CUDA kernels based on the 2DGS [10] and 3DGS [12] frameworks.

In the implementation of error-based selection in mixed 2D-3D Gaussian modeling, we extract the point list and point list keys from the 2DGS CUDA rasterizer. These two lists are one-to-one correspondences, with each element corresponding to the projection of a Gaussian in the image space, and they are ordered by depth. Each element in the point list keys is a 64-bit integer, where the first 32 bits store the tile's ID, and the last 32 bits store the depth value of the current Gaussian projection. Each element in the point list is a 32-bit integer, which stores the index of the 2D Gaussian corresponding to the Gaussian projection. Therefore, using these two lists, we can establish the mapping between the tiles and the 2D Gaussians. By using the tile's ID, we can identify the 2D Gaussians that contribute to it, as shown in line 8 of Alg.1 in the main text. For example, as shown in Fig. 2, after the 2D Gaussian training process, we calculate that tiles 1 and 2 have the largest MSE. Based on the point list keys and point list, we have identified the corresponding 2D Gaussian indices as 235, 234, and 127.

In the implementation of mixed 2D-3D Gaussian splatting, we pass the connection between 2D and 3D Gaussians to the CUDA rasterizer. Fig. 1 is a schematic diagram of this process. During the forward pass, for each pixel, we perform alpha blending of the projections of all 2D Gaussians in the current tile in depth order, from front to back. When a 2D Gaussian with a 3D Gaussian child node is encountered, the projection of the 3D Gaussian is inserted after the projection of the current 2D Gaussian, and alpha blending continues. In the backward pass, this order is reversed, meaning that alpha blending is performed from back to front. Therefore, the projection of the child 3D Gaussian will be inserted

Figure 3. Visualization results of mesh w/ texture.

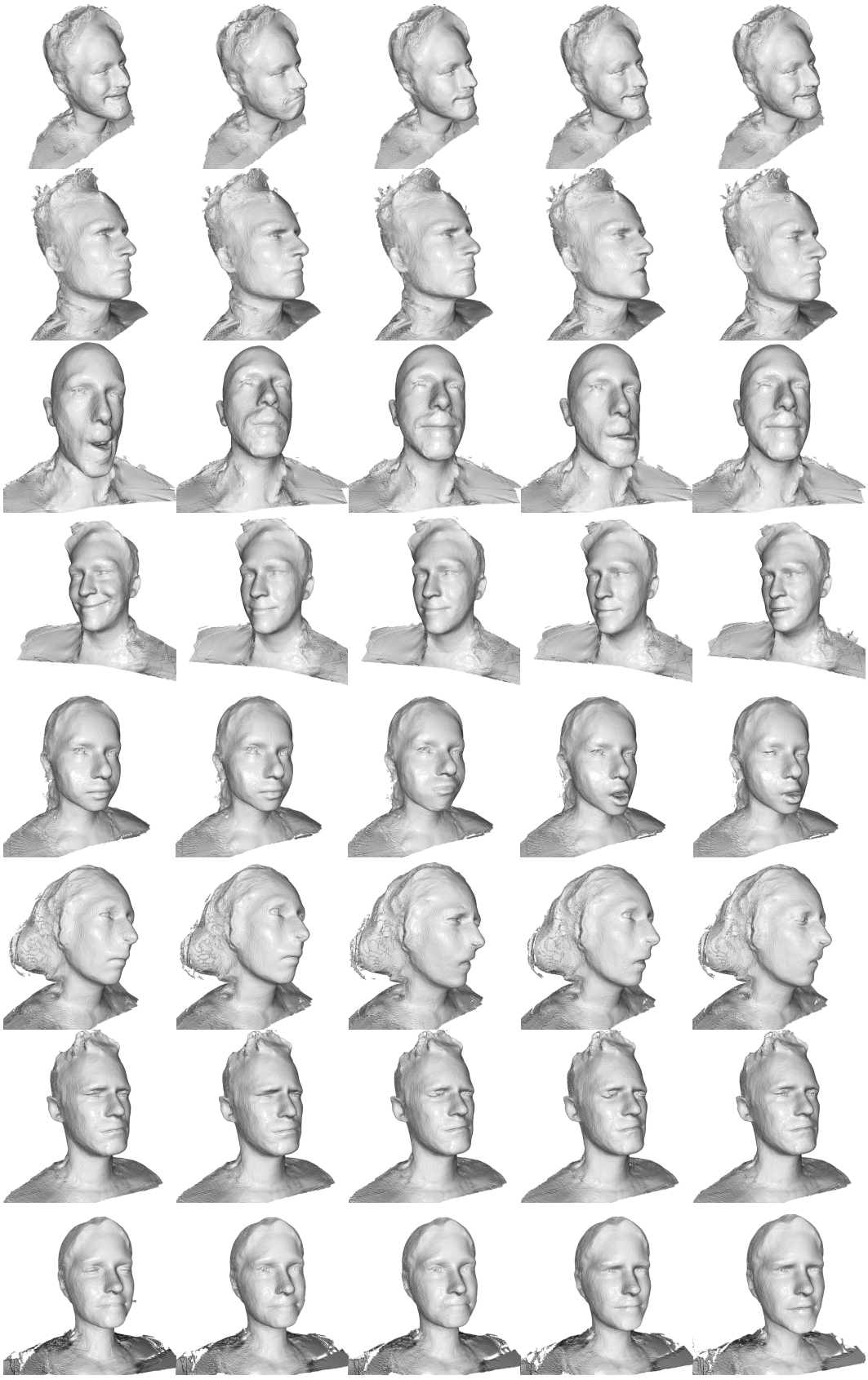

Figure 4. Visualization results of mesh w/o texture.

in front of its parent 2D Gaussian projection, and the alpha blending of the child 3D Gaussian will be computed first. During the backward pass, we update the gradients of all parameters for the 3D Gaussians, as well as the gradients for the color parameters of the 2D Gaussians.

Training Details As shown in Alg.1 of the main text, with $n = 5000$ and $k = 800$, we perform 2D Gaussian training in the first 30,000 iterations, mixed 2D-3D Gaussian modeling between 30,000 and 35,000 iterations, and 2D-3D Gaussian Training between 35,000 and 40,000 iterations. Finally, our MixedGaussianAvatar consists of an average of approximately 40,000 2D Gaussians and 20,000 3D Gaussians in the NeRSemble dataset, which is comparable to the number of Gaussians in GaussianAvatars [20]. All our experiments were conducted on NVIDIA RTX V100s.

Dataset Details For the NeRSemble dataset [13], we conducted experiments using the videos from it. The dataset includes footage captured from 9 subjects across 16 different viewpoints, including frontal and side views. For each subject, the dataset selected 11 video sequences and adjusted the image resolution to 802×550. In the first 10 sequences, participants were instructed to display specific facial expressions or emotions, while the 11th sequence allowed them to express themselves freely. We used VHAP [19] to preprocess the videos in this dataset, obtaining the FLAME parameters and camera parameters for each frame.

For the INSTA dataset [34], it comprises performances by various actors, utilizing video clips obtained from YouTube, ultimately featuring data from twelve different actors. Custom recordings within the dataset capture around 2-3 minutes of monocular RGB full HD video. This footage is subsequently cropped, downsampled to 25 fps, and resized to a resolution of 512×512 pixels. Additionally, the dataset incorporates background-foreground segmentation techniques through robust matting and employs a pre-existing facial parsing framework for both image segmentation and clothing removal.

D. Limitaions and Prospects

Although MixedGaussianAvatar can reconstruct high-quality 3D head avatars, there are still some limitations. First, our method still relies on face tracking tools that extract FLAME parameters from RGB images, often resulting in a cumbersome and time-consuming data preprocessing process. Second, our method uses 2DGS to reconstruct the geometric surface, but 2DGS essentially uses 2D color as supervision rather than the geometric structure. This can lead to a lack of precision in the geometric details such as hair strands and eyelids. Third, our training process is divided into progressive stages, resulting in a lengthy training process. We will address these limitations in future work.

Overall, we have currently applied the mixed 2D-3D Gaussian Splatting method only to 3D head avatars. However, in theory, this method can be extended to other types of scenes, enabling the reconstruction of high-precision surface meshes while rendering high-quality textures or RGB images. Additionally, it demonstrates excellent performance in dynamic scenes. Therefore, this work has very promising prospects and is worth further exploration.

E. Ethical Consideration

Our method has the potential to generate high-quality 3D head avatars, which could be used to spread misinformation, influence public perception, and undermine trust in media sources. This could have significant negative impacts on society. Therefore, it is crucial to consider methods that can reliably distinguish between authentic and fabricated content. We strongly condemn unauthorized and malicious use of this technology and emphasize the need to consider ethical issues when utilizing our method.

F. Demo Video

We provide demo videos (please see the anonymous website https://mixedgaussianavatar.github.io/), which contains the overall framework of our proposed MixedGaussianAvatar, the visualization of the mesh w/o texture and the visualization of the mesh w/ texture.