HeadGaS: Real-Time Animatable Head Avatars via 3D Gaussian Splatting

Helisa Dhamo, Yinyu Nie, Arthur Moreau, Jifei Song, Richard Shaw, Yiren Zhou, and Eduardo Pérez-Pellitero

Huawei Noah's Ark Lab
e.perez.pellitero@huawei.com

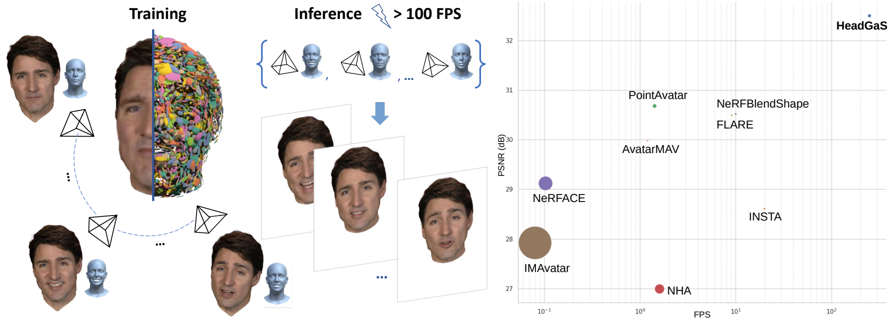

Fig. 1: Overview of HeadGaS. We reconstruct a 3D head based on an expression-aware 3D Gaussian cloud representation, which results in real-time rendering and high image quality. **Left:** The model is trained with a monocular video of a moving head. At inference, we query the model with a novel sequence of poses and expression parameters to render a real-time video. **Right:** Rendering speed (fps in logarithmic scale) vs PSNR plot comparing different methods. The circle radius indicates training time.

Abstract. 3D head animation has seen major quality and runtime improvements over the last few years, particularly empowered by the advances in differentiable rendering and neural radiance fields. Real-time rendering is a highly desirable goal for real-world applications. We propose HeadGaS, a model that uses 3D Gaussian Splats (3DGS) for 3D head reconstruction and animation. In this paper we introduce a hybrid model that extends the explicit 3DGS representation with a base of learnable latent features, which can be linearly blended with low-dimensional parameters from parametric head models to obtain expression-dependent color and opacity values. We demonstrate that HeadGaS delivers state-of-the-art results in real-time inference frame rates, surpassing baselines by up to 2 dB, while accelerating rendering speed by over ×10.

Keywords: animatable head avatars · gaussian splatting · radiance fields

1 Introduction

Reconstructing photorealistic 3D heads which are in turn controllable and naturally expressive is essential for building digital avatars that look and behave like real humans. This has a wide range of applications including AR/VR, teleconferencing, and gaming. Designing head models that accomplish high fidelity in their appearance, are easy to capture and enable expressive control has been an active research field in recent years, specially due to the fast development of neural and differentiable rendering approaches.

Animatable 3D head reconstruction consists in driving a captured head avatar, based on a target sequence of facial expressions and head poses. In the last decades, various parametric 3D morphable models (3DMM) have emerged [5, 6, 24], which can be fitted to sequences of a moving head and later on enable pose and expression control. Though these models make it possible to drive a captured avatar via a set of low-dimensional parameters, generally their generated images lack realism. Other works utilize the fitting of low-dimensional parameters from such 3DMM models for initial estimates and build on other mechanisms to obtain more realistic imagery with animation capabilities [13, 16].

In particular, with the recent success of differentiable rendering, various 3D-aware animatable head models emerged that can reconstruct and render 3D heads, while providing the functionality to drive them based on expression parameters from 3DMM models. These representations can be explicit (mesh, point clouds) [16, 61] or implicit (neural) [14]. Thereby, the explicit models impose stronger constraints on the head surface, which allows for better expression and pose generalization, while making it more difficult to preserve photo realism, as they inherit the limitations and artifacts of the underlying representation (mesh, point cloud) as observed in Gao *et al.* [14] and also seen in our experiments (Fig. 4). With the recent success of neural radiance fields (NeRFs) [31], typically implicit models are based on a NeRF representation [13]. Some of these models [14, 62] prioritize time constraints and therefore rely on very fast volumetric NeRF variants (*e.g.* InstantNGP [33]) to enable fast training and rendering.

Despite impressive efforts to improve NeRFs to be more accurate [3] and fast [33], there is a trade-off between these two aspects that is hard to satisfy simultaneously [3]. Moreover, even fast and efficient NeRF models like InstantNGP typically enable interactive inference frame rates at best (10-15 fps) [20]. Very recently, 3D Gaussian Splatting (3DGS) [20] emerged as a competitive alternative to NeRF, which leads to reasonable photo-realism while bringing the rendering speed to real-time rates. This is thanks to its representation as a set of 3D Gaussian primitives, with a more efficient space coverage compared to point clouds, combined with efficient tile-based rasterization. However, in the light of 3D head animation, in its original form, 3DGS does not constitute an intuitive surface or point set that can be directly deformed based on 3DMM deformation, unlike other well-known representations, *e.g.* surface or pointcloud based.

To circumvent this limitation, we propose HeadGaS, a model that enhances 3D Gaussians with head animation capabilities (see Figure 1). At test time, our model receives a sequence of head poses and expression parameters, and gen-

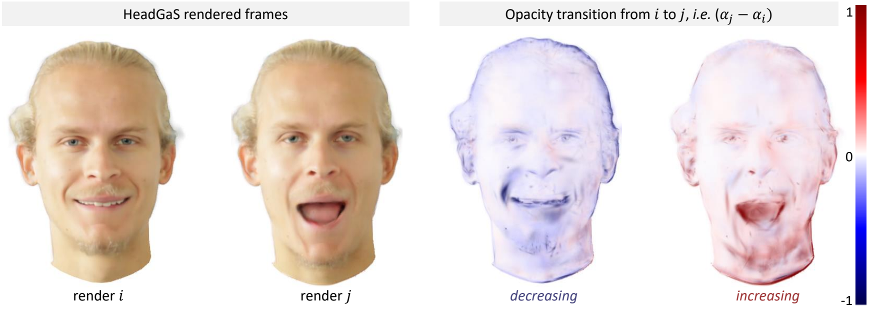

Fig. 2: Motion modelling via opacity change. Left: Two example frames i and j rendered by HeadGaS. **Right**: Rendering of opacity difference $\alpha_j - \alpha_i$ (blue: Gaussians with an opacity decrease; red: Gaussians with an opacity increase; *colors close to white*: minor change, static regions). We observe a strong opacity increase in dynamic areas, *e.g.* lower chin Gaussians turn opaque as the jaw fully opens.

erates a photo-realistic video of the reconstructed avatar. The core idea behind HeadGaS is to represent motion by allowing Gaussians to alter their opacity and color over time. As a consequence, HeadGaS will result in duplicates of the dynamic face areas, *i.e.* achieving dynamics via over-representation. Thus, there will be multiple Gaussians representing the same face region, and these duplicates will become active (apparent) one at a time, to support the face geometry at a certain state of expression. Figure 2 illustrates the opacity change as a result on an expression transition. Note how Gaussians representing lower lip and chin areas in frame i turn transparent to allow for mouth opening in frame j, while another set of chin Gaussians emerge at a lower location to accommodate the new jaw position in frame j. To allow for such varying appearance, guided by an expression vector, we introduce a basis of latent features inside each Gaussian. This learned basis is multiplied with an input expression vector, and its sum is fed to a multi-layer perceptron (MLP) to yield the final color and opacity. This idea is inspired by traditional blendshape 3DMMs [6], and it can be interpreted as a latent-feature shape basis that is blended in the feature domain rather than directly in PCA space [5] or meshes [6]. Our model is simple and effective, and it can work with various 3DMM representations, as it does not explicitly model deformations with respect to a particular mesh topology. Practically, in our experiments we show that HeadGaS can be controlled with expression parameters from two different 3DMMs, namely FLAME [24] and FaceWarehouse [6]. The rendering is done in real-time framerates, at over 100 fps (about 250 fps for 512^2 resolution). We show experimentally that our visibility-varying Gaussians outperform the evident alternative of moving the Gaussians, which we attribute to the fact that adding 3D motion makes the optimization even more complex.

We evaluate our model on publicly available monocular video datasets, commonly used in related works [14, 61, 62]. Thereby, we demonstrate that the proposed model yields superior results, while increasing the rendering speed by at least a ×10 factor compared to interactive NeRF-based baselines [14, 62]. We show the applications of HeadGaS in novel same-person expression transfer, cross-subject expression transfer, as well as novel view synthesis.

To summarize, our contributions include:

1. We formulate a novel framework that can render photo-realistic 3D-aware animatable heads in real-time, adapting an efficient set of 3D Gaussian primitives. This framework handles face dynamics by allowing opacity and color to change over time, *i.e.* leveraging over-representation.
2. We extend 3DGS [20] with a per-Gaussian basis of latent features, which can be blended with expression weights to enable expression control.
3. We extensively evaluate our proposed method, and compare it against state-of-the-art approaches, obtaining up to 2dB improvement and ×10 speed-ups.

2 Related Work

2.1 Towards Fast and Dynamic Radiance Fields

NeRF [31] represent the scene as an implicit neural radiance field, that queries 3D space and predicts density and view-dependent color via a MLP. In the following years, many follow-up works have focused on improving different aspects of it such as anti-aliasing [1–3], regularization for sparse views [11, 34, 47] and speed [8, 33, 45]; or enhancing results post-rendering [7]. DVGO [45] replace the MLP of NeRF with a density and learned feature voxel grid to considerably speed up convergence. TensoRF [8] factorize the 4D feature voxel grid of a scene into a set of low-rank 2D and 3D tensors which improves efficiency. InstantNGP [33] employ a hash grid and an occupancy grid to accelerate computation, followed by a small MLP that infers density and color. NeRFs have also been used to represent dynamic scenes including human bodies [9,38,50], human heads [14,62], and generic time-varying scenes [12,25,36,37,39,46]. Typically these models rely on a canonical space, where all observations are mapped for time consistent reconstruction. Methods aiming at fast rendering speeds [14, 62] build on an InstantNGP hash grid and achieve interactive frame rates (10-15 fps).

3DGS [20] represent a scene as a set of explicit 3D Gaussians with the motivation to minimize computation in empty spaces. Their efficient representation combined with tile-based rasterization algorithm allows for accelerated training and real-time rendering (over 100 fps). A line of works extends the 3D Gaussian representation to model dynamic scenes [29,43,51,56]. Luiten *et al.* [29] propose simultaneous dynamic scene reconstruction and 6-DoF tracking by allowing the Gaussians to change position and rotation over time while enforcing the same color, size and opacity. Yang *et al.* [56] learn a MLP based deformation that maps 3D Gaussians to a canonical space. 4D-GS [51] propose an efficient deformation field by querying features in shared multi-resolution voxel planes.

Very recent concurrent works model human head [10,52,54] and body [23,32] avatars with 3DGS, by deforming a canonical head via a 3DMM-conditioned MLP [52, 54], relying on a tri-plane [49], or binding 3D Gaussians in a FLAME mesh [40]. Our method is quite different from these works, in that we utilize a per-Gaussian feature basis and opacity induced dynamics.

2.2 Head Reconstruction and Animation

Head reconstruction from a set of image observations has been a very active field in the recent years, including models that generalize across subjects [17, 30, 48], or rely on multi-view head captures [18, 22, 27, 28], which can have a static [18, 48] or dynamic form [17, 22]. Most related to our proposed method, are works that learn a dynamic, animatable 3D head model from a monocular video sequence that observes the head in various poses and facial expressions, and is capable to generate a novel expression or pose at test time. A line of works rely on explicit scene representations, such as meshes or point clouds [15, 16, 21, 61]. Neural Head Avatars [16] models the geometry and texture explicitly via a hybrid representation consisting of a coarse morphable model, followed by a neural based refinement that predict voxel offsets. PointAvatar [61] propose a deformable point-based representation, where all points have the same radius. Our 3D Gaussian set shares some similarity with a point cloud, yet it is more flexible. Each Gaussian can have its own radius, orientation and different axis lengths. Recently, FLARE [4] was proposed, a model that updates a traditional graphics pipeline with a few neural components. FLARE can optimize a 3D mesh via differentiable rendering, enabling avatars that are animatable and relightable.

Another line of works extend implicit neural radiance representations. NerFACE [13] use a dynamic NeRF to combine scene information with a morphable head model to enable pose and expression control. IMAvatar [60] utilizes neural implicit surfaces [35] and learns an implicit deformation field from canonical space to observation based on expression parameters and pose. With the goal of fast training, and interactive rendering, more recent works extend Instant-NGP [33] with head aware models. INSTA [62] use a tracked FLAME mesh as a geometrical prior to deform the points into canonical space, followed by Instant-NGP [33]. NeRFBlendShape [14] follow a different approach, that in contrast to most previous works does not rely on deformation. Instead, inspired by classic blendshape models for heads [6], they utilize a base of multi-level hash grid fields [33], where the model can be driven via a linear blending of such hash grid base with the expression vector. Similarly, AvatarMAV [55] use expression weights to blend a set of motion voxel grids. Our proposed approach resembles these blending-based ideas, but instead we use the expression vector to blend latent per-Gaussian features to predict expression-specific color and opacity.

Different from all works discussed in this section, we adopt a set of 3D Gaussians as neural radiance representation, to take advantage of the fast rendering benefits, combined with competitive photorealism.

3 Method

Given a monocular video of a moving head, our goal is to learn a 3D head model and render novel images of this avatar based on a facial expression vector and camera pose. As a pre-processing step, similar to other works [14, 16, 61, 62], we require head poses and a vector of expression weights associated with each frame, and adopt a head tracking pipeline to achieve this. Note that HeadGaS does not

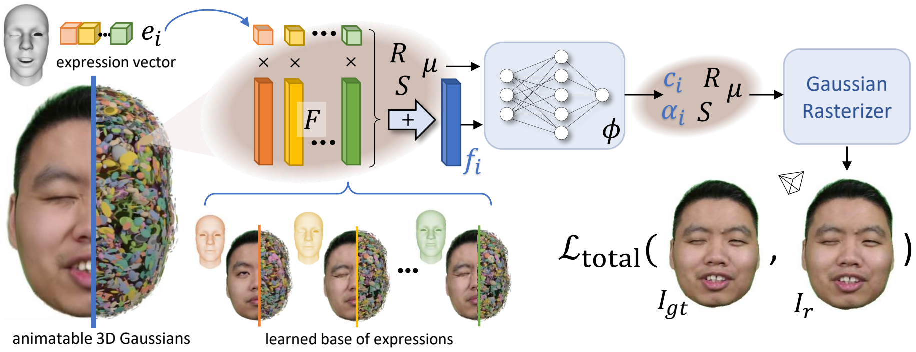

Fig. 3: HeadGaS pipeline. We represent 3D space as a set of feature-enhanced 3D Gaussians. Every Gaussian contains a feature basis $\boldsymbol{F}$ that can be blended via the expression vector to obtain a frame specific feature $\boldsymbol{f}_i$. The frame specific feature is fed to an MLP $\phi(\cdot)$ alongside position $\boldsymbol{\mu}$ to obtain expression-dependent color $\boldsymbol{c}_i$ and opacity $\boldsymbol{\alpha}_i$. Finally, $\boldsymbol{c}_i$ and $\boldsymbol{\alpha}_i$ are fed to the rasterizer alongside other Gaussian parameters like rotation R, scale S and position $\boldsymbol{\mu}$ to render the image.

explicitly build on a certain parametric model, and it can therefore work with different head models. We have performed experiments with two models [6, 24] based on the head tracking frameworks of prior work [14, 60, 62]. The resulting rigid head poses are converted to camera poses, to map all observations to the canonical head pose. In addition, we perform video matting [26] followed by face parsing [58] to discard background areas and clothing to focus on the head region only. In the following sections, we will first provide a background on the original 3DGS (Sec. 3.1) and further describe the proposed strategy for animatable 3DGS (Sec. 3.2), the rendering (Sec. 3.3) and its optimization (Sec. 3.4).

3.1 Original 3DGS Representation

Given a set of images of a static scene and the corresponding camera poses, 3DGS [20] learn a 3D scene as a set of 3D Gaussians, and can render a novel image from a given viewpoint. For initialization, 3DGS utilizes a sparse point cloud, typically originating from the COLMAP [42] framework by which they also obtain the camera poses. Thereby, a 3D Gaussian is represented as a tuple of 3D covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{3\times3}$, Gaussian center $\boldsymbol{\mu} \in \mathbb{R}^3$, color $\boldsymbol{c} \in \mathbb{R}^{3(k+1)^2}$ and opacity $\alpha \in \mathbb{R}$, *i.e.* $\mathcal{G} = (\boldsymbol{\Sigma}, \boldsymbol{\mu}, \boldsymbol{c}, \alpha)$, where k is the degree of the spherical harmonics. The Gaussians are defined in world space, centered at the mean point

$$G(x) = e^{-\frac{1}{2}(x)^T \boldsymbol{\Sigma}^{-1}(x)}. \tag{1}$$

To make optimization stable, *i.e.* guarantee that $\boldsymbol{\Sigma}$ is positive semi-definite, the covariance matrix is further decomposed into rotation R and scaling S:

$$\boldsymbol{\Sigma} = RSS^TR^T. \tag{2}$$

Color is given as spherical harmonics (SH) of degree k and is thus view-dependent.

The Gaussian parameters are optimized via a differentiable rasterizer, that projects the current 3D Gaussians to the image space and compares against the ground truth images. This rasterizer relies on an efficient algorithm for sorting the Gaussians and tiling the image space, which leads to very fast training and rendering. Alongside the optimizations, 3DGS employs an adaptive mechanism for pruning and densification, to make sure that the set of gaussians represents the space effectively. For more details, we refer the reader to the 3DGS paper [20].

3.2 Feature Blending Formulation

Here we describe how we extend the 3DGS representation with animation capabilities. The vanilla 3DGS model does not inherently allow for this, as it learns a static set of parameters, which is the same for all frames. Inspired by 3DMMs, the goal of our model is to explore a blending mechanism for the 3D Gaussian components, using the pre-computed expression parameters as blending weights. Namely, we want each Gaussian to change color and opacity based on the current expression i. This leads to 3D Gaussians with dynamic appearance which occasionally appear and vanish depending on the current expression, and additionally allow color changes for non-rigid appearance effects such as wrinkles. For instance, referring to Figure 2, Gaussians of closed lips visible in frame i will turn transparent in frame j, as the jaw opens, while another set of Gaussians at a *different* location will become visible to render open lips. The model will thus learn multiple Gaussians corresponding to the same region in the face, such that these can become opaque as needed.

With the goal of enabling such dynamic appearance, we extend *every* 3D Gaussian with a basis of latent features $\boldsymbol{F} \in \mathbb{R}^{B \times f_{dim}}$, (see Figure 3). Our animatable 3D Gaussian representation then becomes $\mathcal{G}_a = (\boldsymbol{\Sigma}, \boldsymbol{\mu}, \boldsymbol{F})$. The latent base is optimized together with the other parameters of the 3D Gaussian. At each iteration, we leverage the respective expression weights $\boldsymbol{e}_i \in \mathbb{R}^B$ of the current frame i, to blend the feature basis $\boldsymbol{F}$ into a 1D vector $\boldsymbol{f}_i \in \mathbb{R}^{f_{dim}}$

$$\boldsymbol{f}_i = \boldsymbol{F}^T \boldsymbol{e}_i + \boldsymbol{f}_0 \tag{3}$$

where $\boldsymbol{f}_0$ is a bias term. We index with i all variables that are specific to a particular frame i. This frame specific feature $\boldsymbol{f}_i$ is then fed into a small MLP $\phi(\cdot)$, to compute the color $\boldsymbol{c}_i$ as well as the opacity α_i

$$\boldsymbol{c}_i, \alpha_i = \phi(\boldsymbol{f}_i, \psi(\boldsymbol{\mu})) \tag{4}$$

where ψ denotes sinusoidal positional encoding, the learned color is a 1D vector $\boldsymbol{c}_i \in \mathbb{R}^{3(k+1)^2}$, and the learned opacity is a scalar $\alpha_i \in \mathbb{R}$. As most of the dynamic effects are already captured by the per-Gaussian feature bases $\boldsymbol{F}$, we are able to use a very small MLP that does not compromise the rendering speed. Our MLP is composed of only two linear layers, each followed by LeakyReLU activation [53], where the hidden layer has 64 channels. The last layer consists of two branches, *i.e.* for color and opacity prediction. We use a sigmoid activation function at the end of the opacity branch to constrain it to be in its appropriate range $[0, 1]$. As color and opacity are learned via the MLP, we omit them from the explicit optimizable Gaussian parameters.

An alternative to blending in the latent space, would be to directly define a basis of explicit Gaussian parameters and similarly blend them based on the expression weights. However, as these values have an explicit meaning (*i.e.* color, position), a multiplication with expression weights that are not even learnable, makes this formulation limiting and prone to artifacts, as we also show in our ablation (Sec. 4.2, *Ours w/o MLP*). Interestingly, even though changing the centers and rotations of the Gaussian splats (instead of modifying opacity and colour) would be an intuitive mechanism when it comes to 3D Gaussian animation, our proposed approach results in much better performance (see Sec. 4.2, *Ours w/ $\Delta(\mu, R)$*). We additionally show that the proposed feature blending strategy is superior to the straight-forward approach of using the expression vector as a condition to the MLP (*Ours w/o blending*).

3.3 Rendering

To render frame i, we employ the respective expression $\boldsymbol{e}_i$ to populate each Gaussian with expression-dependent color and opacity. Then, the Gaussians are rendered using the camera view $\boldsymbol{W}_i$, and similarly to Kerbl *et al.* [20] we perform the splatting technique on the primitives [57]. Given a viewing transform $\boldsymbol{W}$ as well as the Jacobian of the affine approximation of the projective transformation $\boldsymbol{J}$, the covariance matrix $\boldsymbol{\Sigma}'$ in camera coordinates can be obtained from

$$\boldsymbol{\Sigma}' = \boldsymbol{J}\boldsymbol{W}\boldsymbol{\Sigma}\boldsymbol{W}^T\boldsymbol{J}^T. \tag{5}$$

The Gaussian splats are then rendered via a tile-based differentiable rasterizer [20] that pre-sorts all primitives of an image at once.

3.4 Optimization

We initialize the 3D Gaussians centers with 2500 points. Whenever available, these points are a subset of vertices from the tracked 3DMM meshes (*e.g.* FLAME based data released by prior works [61, 62]). As there is no mesh available for the data from Gao *et al.* [14], we sample random points within the given near and far bounds. Empirically we found that initializing the latent features $\boldsymbol{F}$ with zeros led to the most stable solution. The model is optimized by rendering the learned Gaussians and comparing the resulting image I_r against the ground truth I_gt. We minimize the following loss objective

$$\mathcal{L}_\mathrm{total} = \lambda_1\mathcal{L}_1(I_\mathrm{r}, I_\mathrm{gt}) + \lambda_s\mathcal{L}_\mathrm{SSIM}(I_\mathrm{r}, I_\mathrm{gt}) + \lambda_p\mathcal{L}_p(I_\mathrm{r}, I_\mathrm{gt}) \tag{6}$$

where the λs are weighting factors and $\mathcal{L}_p$ denotes the perceptual loss [19]. We optimize using Stochastic Gradient Descent [41] with a standard exponential decay scheduling for the Gaussian position centers $\boldsymbol{\mu}$ as well as the MLP.

Adaptive densification and pruning Following 3DGS [20], we combine our optimization with periodic steps of adaptive densification and pruning. First, this mechanism prunes Gaussians that are almost transparent, *i.e.* $\alpha < \tau_\alpha$ smaller than a threshold. Second, the densification targets areas that need to be populated with more Gaussians, represented with Gaussians that are too large, or regions that are too sparse and lack detail. Based on the observation that in both cases the position gradients have high values [20], the Gaussians that should be densified are identified utilizing the average gradient magnitude being above a threshold τ_{pos}. In the case of Gaussians that are too small, the objective is to increase volume and therefore the identified Gaussians are simply cloned, preserving their size. On the other hand, for Gaussians that are too large, the goal is to preserve the overall volume and therefore their scales are decreased by a factor of 1.6 after cloning, obtained empirically by [20].

3.5 Implementation Details

The learning rates for the MLP $\phi(\cdot)$, positions μ, latent features $\boldsymbol{F}$, scale S and rotation R are namely $1.6{\cdot}10^{-4}$, $1.6{\cdot}10^{-4}$, 0.0025, 0.005 and 0.001. We set the latent feature dimensionality to f_{dim}=32. For the FLAME tracking [62] we only use the first 52 expression weights, *i.e.* B=52. All Gaussians are fed as a single batch into the MLP. The $\mathcal{L}_p$ loss is based on a VGG network [44] and has a weight of λ_p=0.1, while λ_1=0.8 and λ_s=0.2. We activate the $\mathcal{L}_p$ loss after 10k iterations such that it does not conflict with photometric loss at the early stage of learning. To save computation, we apply $\mathcal{L}_p$ on the image region defined by the head bounding box. The densification starts after 500 iterations and stops with 15k iterations. In our experiments we use an SH degree of k=3. We train our models on one Tesla V100 GPU for 50k iterations taking about 1 hour.

4 Experiments

In this section we describe the evaluation protocol followed by quantitative and qualitative results in three different scenarios, *e.g.* same-subject novel expression and novel view rendering, as well as cross-subject expression driving.

We evaluate our model on three datasets, made publicly available by recent works such as NeRFBlendShape (we dub this NBS data), INSTA, and PointAvatar. The NBS data [14] contains a set of monocular videos from 8 subjects, where the last 500 frames in each subject constitute the test set. The INSTA dataset [62] contains 10 subjects, with the last 350 frames of each sequence being the test set. We additionally evaluate on the 3 subjects made available by PointAvatar [61], where the test sets contain between $880-1800$ frames per subject. For fairness of comparison, when training our method in all three datasets, we use the same splits and utilize the tracked data (head poses and expression weights) provided originally by the authors, such that tracking quality does not affect the comparison. To train our model we use a subset of about $1500-2500$ frames from the training set of each subject. To assess the quality of the synthesized images

Table 1: Results on the dataset provided by INSTA [62], NeRFBlendShape [14] (NBS) and PointAvatar [61]. We report PSNR, SSIM and LPIPS, together with time measures in seconds of 1 frame rendering (for 512^2 resolution).

Method	dataset	L2 ↓	PSNR ↑	SSIM ↑	LPIPS ↓	Time (s) ↓
NHA [16]	INSTA	0.0024	26.99	0.942	0.043	0.63
IMAvatar [60]		0.0021	27.92	0.943	0.061	12.34
NeRFACE [13]		0.0016	29.12	0.951	0.070	9.68
AvatarMAV [55]		0.0012	29.98	0.948	0.079	0.85
FLARE [4]		0.0010	30.49	0.942	0.050	0.11
INSTA [62]		0.0017	28.61	0.944	0.047	0.05
PointAvatar [61]		0.0009	30.68	0.952	0.058	0.1 - 1.5
NeRFBlendShape [14]		0.0011	30.52	0.955	0.056	0.10
HeadGaS (Ours)		**0.0008**	**32.50**	**0.971**	**0.033**	**0.004**
NeRFBlendShape [14]	NBS	0.0005	34.34	0.970	0.0311	0.10
HeadGaS (Ours)		**0.0003**	**36.66**	**0.976**	**0.0261**	**0.004**
PointAvatar [61]	PointAvatar	**0.0027**	**26.04**	0.885	0.147	0.1 - 1.5
HeadGaS (Ours)		0.0029	25.99	**0.897**	**0.108**	**0.004**

we report common metrics such as Peak Signal-to-Noise Ratio (PSNR), Structural Similarity (SSIM), Learned Perceptual Image Patch Similarity (LPIPS) [59] and the Mean Squared Error (L2). All metrics are computed using white background for non-face regions. Further, we report rendering times in seconds. We compare against common baselines such as NeRFBlendShape [14], INSTA [62], PointAvatar [61], NHA [16], IMAvatar [60], AvatarMAV [55], FLARE [4] and NeRFACE [13]. The evaluation is carried out using their official code repositories, as well as their official checkpoints whenever available.

4.1 Same-Subject Novel Expression Driving

Table 1 reports the results of the metric comparison against baselines on the respective test sets. The three different blocks show namely results on the data released by INSTA, NBS and PointAvatar. Figure 4 illustrates the qualitative comparison with the most recent baselines in all three datasets. We observe that the proposed method outperforms all baselines on the INSTA and NBS datasets in all metrics, with a PSNR gap of about 2 dB. Referring to Figure 4, we observe that HeadGaS leads to higher fidelity to ground truth, less artifacts, and identity preservation for all subjects. Interestingly, as INSTA relies on mesh deformations, it exhibits artifacts such as noticeable triangles on the skin surface (Figure 4a). Moreover, our model preserves better details, such as facial expressions, wrinkles, eyebrows, teeth and glass reflections, while other baselines [14, 61, 62] often struggle in such aspects. The comparison against the PointAvatar baseline shows that we are superior on the INSTA data by about 2 dB. On the 3 subjects of the PointAvatar dataset, our method surpasses the baseline in terms of LPIPS and SSIM, while having very similar PSNR. Overall, looking at both datasets, the performance of HeadGaS is superior to that of PointAvatar. Also qualitatively, we can see that PointAvatar results have distortions of some parts that undergo significant transformation, including inaccurate teeth deformation (Figure 4c). We believe that, as an explicit method, PointAvatar generalizes well for under-observed expressions (resulting in comparable PSNR). However, as these cases struggle with deformation realism (teeth wrongly deform in the same way as the mouth), the structural metrics are worse. Finally, as the table shows, we improve the rendering time of all baselines for 512 resolution by at least a factor of 10. We refer the reader to the supplement for more qualitative results and videos.

4.2 Ablations

In this section we ablate the HeadGaS components. First, we train a model that does not use learned features for blending, but rather a base of colors and opacities, and uses the expression weights to obtain the final color and opacity as a weighted average. We refer to this model as *Ours w/o MLP*. In addition, since an intuitive alternative for dynamic Gaussians would be to deform the points (rather than adapting color and opacity) we introduce *Ours w/* $\Delta(\mu, R)$, which uses the learned feature basis (and a similar MLP) to rather shift the positions μ and rotations R. Additionally, we train a model that predicts color, opacity, as well as a μ and R shift (*Ours change all*). Further, we run a model without the perceptual loss, *i.e. Ours w/o* $\mathcal{L}_p$, to ablate its effectiveness. Additionally, to validate the contribution of using the expression parameters as a weight for blending Gaussian features, rather than a simple condition to the MLP, we ablate a variant named *Ours w/o blending*. For fairness, we increase the capacity of this MLP until it plateaus. More details can be found in the supplement.

Table 2 reports the quantitative evaluation of our model components. We observe that using the expression parameters as a simple condition (*Ours w/o blending*) leads to noticeably inferior performance. In contrast to our model - which learns per-Gaussian dynamics via a feature basis - the per-subject MLP has to learn the face dynamics for all Gaussians at once, leading to poorer expression generalization, as can be also seen on Figure 5. Also, applying a transformation to the positions and rotations leads to worse results (*Ours w/* $\Delta(\mu, R)$). We hypothesize this is because, in the context of 3DGS - which is relying on several heuristics - adding another dimension (in the form of spatial 3D motion) further complicates the already difficult optimization, resulting in geometrically inconsistent transformation (*e.g.* failure to preserve relative distances of points in the skin), especially for large motion. Figure 5 confirms these results and reveals floater artifacts and less accurate expressions. We also observe that allowing all parameters to change (*Ours change all*) increases the solution space and makes the heuristic-based 3DGS optimization more challenging, leading to blurrier results. Further, blending the explicit parameters directly (*Ours w/o MLP*) leads to a worse performance than our neural variant. Despite a tighter PSNR gap, we notice a drastic visual effect on the highly dynamic areas, relevant to the blending, as illustrated in Figure 5. Finally, we see that adding a perceptual loss term $\mathcal{L}_p$ leads to an improvement in most metrics. We refer the reader to the supplement for video comparisons, as well as ablation on additional aspects such as number of Gaussians and speed.

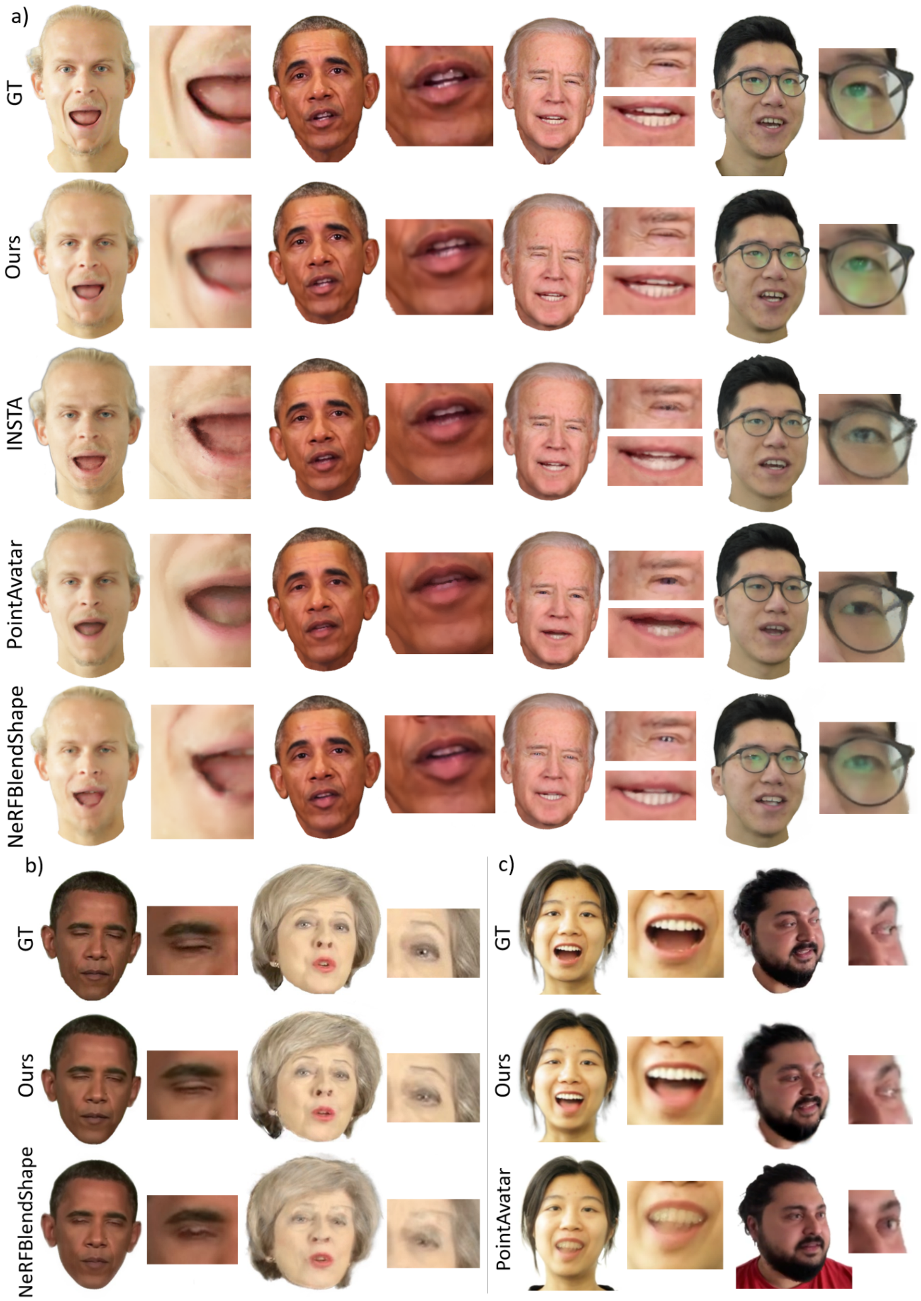

Fig. 4: Qualitative evaluation comparing the proposed model against INSTA [62], PointAvatar [61] and NeRFBlendShape [14] baselines, namely on the **a)** INSTA data, **b)** NBS data and **c)** PointAvatar data. The close-ups on the right of each example highlight our method's ability to capture details like teeth, wrinkles and reflections.

Table 2: Ablation of HeadGaS components on the INSTA dataset.

Method	L2 ↓	PSNR ↑	SSIM ↑	LPIPS ↓
Ours w/o blending	0.0012	30.28	0.955	0.041
Ours w/ $\Delta(\mu, R)$	0.0014	29.83	0.953	0.045
Ours change all	0.0014	29.65	0.951	0.041
Ours w/o MLP	0.0009	32.08	0.968	**0.033**
Ours w/o $\mathcal{L}_p$	0.0008	32.11	0.969	0.046
Ours	**0.0008**	**32.50**	**0.971**	**0.033**

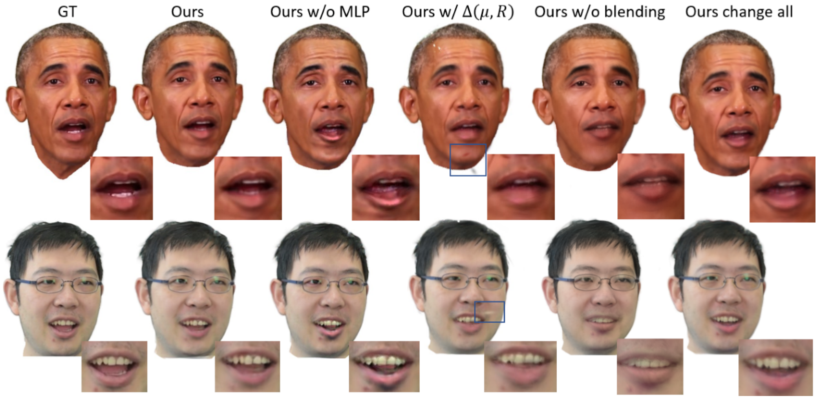

Fig. 5: Qualitative ablation on the INSTA dataset

4.3 Novel View Synthesis

In Figure 6 we render the avatars from multiple views, including the original test set camera (left) and two additional viewpoints (right). Thereby we render the same facial expression. We observe that our model can deliver expressions that are consistent across different views. Videos can be found in the supplement.

4.4 Cross-Subject Expression Driving

Figure 7 reports our cross-subject driving results, *i.e.* use the facial expression from another (ground truth) source subject to drive a target subject. Here we retain the original head pose of the target subject. As can be seen in the figure, our model is capable of transferring various expressions, *e.g.* talking, wink, surprise, across different subjects at a reasonable quality.

5 Limitations and Ethical Consideration

HeadGaS is affected by failures of the head tracker, which can take the form of inaccurate facial expressions, or blurriness in case of pose inaccuracy in the training data (Figure 8). Further, as a data driven method, HeadGaS requires a

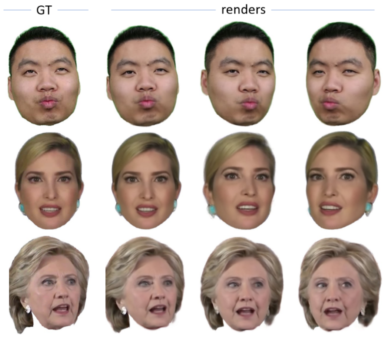

Fig. 6: Rendering from various camera views for the same expressions

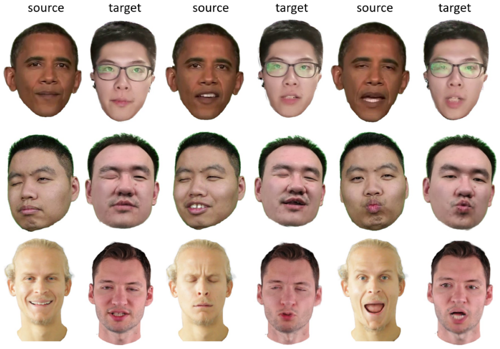

Fig. 7: Cross-subject driving: rendering target subject with expression from the source subject

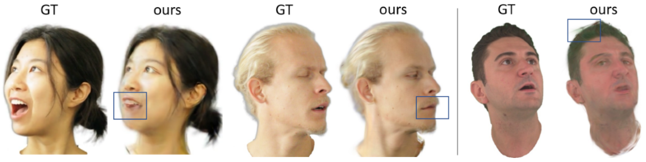

Fig. 8: Limitations. *Left:* subjects only observed in neutral expression in non-frontal view. *Right:* Camera view that is far from the training observations.

reasonable coverage of expressions across different views. For instance, if a head changes pose only in neutral expression, and diverse expressions are observed only in frontal view, it would be difficult to capture a non-neutral expression from a side view (Figure 8, left). Finally, a downside of HeadGaS is the memory consumption originating from the feature bases ($B \times f_{dim}$ floats per Gaussian).

The use of personal data should be handled carefully, and follow local regulations. We note that face reanimation can potentially be used to generate fake content, and convey misinformation. We do not condone such practises, and believe that the community should work accordingly towards mitigating the risks.

6 Conclusion

We presented HeadGaS, a model for animatable head reconstruction and rendering from a monocular video that renders on real-time. Our extensive evaluation showed that the proposed model results in state-of-the-art performance, clearly surpassing the baselines, while rendering on real-time frame rates (about 250 fps for a 512^2 resolution). We justified our design choices via a set of ablations, where we demonstrated that linearly blending implicit features leads to less artifacts than the alternative of blending explicit parameters directly. Moreover, we have shown that changing colors and opacity is more effective than the intuitive alternative of transforming the Gaussian mean positions. Future work can be dedicated to improving the memory efficiency of the HeadGaS feature bases.

References

1. Barron, J.T., Mildenhall, B., Tancik, M., Hedman, P., Martin-Brualla, R., Srinivasan, P.P.: Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. ICCV (2021)
2. Barron, J.T., Mildenhall, B., Verbin, D., Srinivasan, P.P., Hedman, P.: Mip-nerf 360: Unbounded anti-aliased neural radiance fields. CVPR (2022)
3. Barron, J.T., Mildenhall, B., Verbin, D., Srinivasan, P.P., Hedman, P.: Zip-nerf: Anti-aliased grid-based neural radiance fields. ICCV (2023)
4. Bharadwaj, S., Zheng, Y., Hilliges, O., Black, M.J., Abrevaya, V.F.: FLARE: Fast learning of animatable and relightable mesh avatars. ACM TOG (2023)
5. Blanz, V., Vetter, T.: A morphable model for the synthesis of 3d faces. Conference on Computer Graphics and Interactive Techniques, SIGGRAPH (1999)
6. Cao, C., Weng, Y., Zhou, S., Tong, Y., Zhou, K.: Facewarehouse: A 3d facial expression database for visual computing. IEEE Transactions on Visualization and Computer Graphics (2014)
7. Catley-Chandar, S., Shaw, R., Slabaugh, G., Pérez-Pellitero, E.: Roguenerf: A robust geometry-consistent universal enhancer for nerf. ECCV (2024)
8. Chen, A., Xu, Z., Geiger, A., Yu, J., Su, H.: Tensorf: Tensorial radiance fields. ECCV (2022)
9. Chen, J., Zhang, Y., Kang, D., Zhe, X., Bao, L., Jia, X., Lu, H.: Animatable neural radiance fields from monocular rgb videos. ArXiv **abs/2106.13629** (2021)
10. Chen, Y., Wang, L., Li, Q., Xiao, H., Zhang, S., Yao, H., Liu, Y.: Monogaussianavatar: Monocular gaussian point-based head avatar. ACM SIGGRAPH Conference Proceedings (2024)
11. Deng, K., Liu, A., Zhu, J.Y., Ramanan, D.: Depth-supervised NeRF: Fewer views and faster training for free. CVPR (2022)
12. Du, Y., Zhang, Y., Yu, H.X., Tenenbaum, J.B., Wu, J.: Neural radiance flow for 4d view synthesis and video processing. ICCV (2021)
13. Gafni, G., Thies, J., Zollhöfer, M., Nießner, M.: Dynamic neural radiance fields for monocular 4d facial avatar reconstruction. CVPR (2021)
14. Gao, X., Zhong, C., Xiang, J., Hong, Y., Guo, Y., Zhang, J.: Reconstructing personalized semantic facial nerf models from monocular video. ACM TOG (Proceedings of SIGGRAPH Asia) (2022)
15. Garrido, P., Valgaerts, L., Rehmsen, O., Thormählen, T., Pérez, P., Theobalt, C.: Automatic face reenactment. CVPR (2014)
16. Grassal, P.W., Prinzler, M., Leistner, T., Rother, C., Nießner, M., Thies, J.: Neural head avatars from monocular rgb videos. CVPR (2022)
17. Hong, Y., Peng, B., Xiao, H., Liu, L., Zhang, J.: Headnerf: A real-time nerf-based parametric head model. CVPR (2022)
18. Jang, Y., Zheng, J., Song, J., Dhamo, H., Pérez-Pellitero, E., Tanay, T., Maggioni, M., Shaw, R., Catley-Chandar, S., Zhou, Y., Deng, J., Zhu, R., Chang, J., Song, Z., Yu, J., Zhang, T., Nguyen, K.B., Yang, J.S., Dogaru, A., Egger, B., Yu, H., Gupta, A., Julin, J., Jeni, L.A., Kim, H., Cho, J., Hwang, D., Lee, D., Kim, D., Seo, D., Jeon, S., Choi, Y., Kang, J.S., Seker, A.C., Ahn, S.C., Leonardis, A., Zafeiriou, S.: Vschh 2023: A benchmark for the view synthesis challenge of human heads. In: Proceedings of the IEEE/CVF ICCV Workshops (2023)
19. Johnson, J., Alahi, A., Fei-Fei, L.: Perceptual losses for real-time style transfer and super-resolution. ECCV (2016)

20. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. ACM TOG (2023)
21. Kim, H., Garrido, P., Tewari, A., Xu, W., Thies, J., Nießner, M., Pérez, P., Richardt, C., Zollöfer, M., Theobalt, C.: Deep video portraits. ACM TOG (2018)
22. Kirschstein, T., Qian, S., Giebenhain, S., Walter, T., Nießner, M.: Nersemble: Multi-view radiance field reconstruction of human heads. ACM TOG (2023)
23. Kocabas, M., Chang, R., Gabriel, J., Tuzel, O., Ranjan, A.: Hugs: Human gaussian splats. CVPR (2024)
24. Li, T., Bolkart, T., Black, M.J., Li, H., Romero, J.: Learning a model of facial shape and expression from 4D scans. ACM TOG, (Proc. SIGGRAPH Asia) (2017)
25. Li, Z., Niklaus, S., Snavely, N., Wang, O.: Neural scene flow fields for space-time view synthesis of dynamic scenes. CVPR (2021)
26. Lin, S., Yang, L., Saleemi, I., Sengupta, S.: Robust high-resolution video matting with temporal guidance. WACV (2022)
27. Lombardi, S., Saragih, J., Simon, T., Sheikh, Y.: Deep appearance models for face rendering. ACM TOG (2018)
28. Lombardi, S., Simon, T., Schwartz, G., Zollhoefer, M., Sheikh, Y., Saragih, J.: Mixture of volumetric primitives for efficient neural rendering. ACM TOG (2021)
29. Luiten, J., Kopanas, G., Leibe, B., Ramanan, D.: Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. 3DV (2024)
30. Mihajlovic, M., Bansal, A., Zollhoefer, M., Tang, S., Saito, S.: KeypointNeRF: Generalizing image-based volumetric avatars using relative spatial encoding of keypoints. ECCV (2022)
31. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. ECCV (2020)
32. Moreau, A., Song, J., Dhamo, H., Shaw, R., Zhou, Y., Pérez-Pellitero, E.: Human gaussian splatting: Real-time rendering of animatable avatars. CVPR (2024)
33. Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. ACM Trans. Graph. (2022)
34. Niemeyer, M., Barron, J.T., Mildenhall, B., Sajjadi, M.S.M., Geiger, A., Radwan, N.: Regnerf: Regularizing neural radiance fields for view synthesis from sparse inputs. CVPR (2022)
35. Park, J.J., Florence, P., Straub, J., Newcombe, R., Lovegrove, S.: Deepsdf: Learning continuous signed distance functions for shape representation. CVPR (2019)
36. Park, K., Sinha, U., Barron, J.T., Bouaziz, S., Goldman, D.B., Seitz, S.M., Martin-Brualla, R.: Nerfies: Deformable neural radiance fields. ICCV (2021)
37. Park, K., Sinha, U., Hedman, P., Barron, J.T., Bouaziz, S., Goldman, D.B., Martin-Brualla, R., Seitz, S.M.: Hypernerf: A higher-dimensional representation for topologically varying neural radiance fields. ACM TOG (2021)
38. Peng, S., Dong, J., Wang, Q., Zhang, S., Shuai, Q., Zhou, X., Bao, H.: Animatable neural radiance fields for modeling dynamic human bodies. ICCV (2021)
39. Pumarola, A., Corona, E., Pons-Moll, G., Moreno-Noguer, F.: D-nerf: Neural radiance fields for dynamic scenes. CVPR (2020)
40. Qian, S., Kirschstein, T., Schoneveld, L., Davoli, D., Giebenhain, S., Nießner, M.: Gaussianavatars: Photorealistic head avatars with rigged 3d gaussians. CVPR (2024)
41. Ruder, S.: An overview of gradient descent optimization algorithms. arXiv preprint arXiv:1609.04747 (2016)
42. Schönberger, J.L., Frahm, J.M.: Structure-from-motion revisited. CVPR (2016)

43. Shaw, R., Song, J., Moreau, A., Nazarczuk, M., Catley-Chandar, S., Dhamo, H., Pérez-Pellitero, E.: Swings: Sliding windows for dynamic 3d gaussian splatting. ECCV (2024)
44. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. ICLR (2015)
45. Sun, C., Sun, M., Chen, H.: Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. CVPR (2022)
46. Tretschk, E., Tewari, A., Golyanik, V., Zollhöfer, M., Lassner, C., Theobalt, C.: Non-rigid neural radiance fields: Reconstruction and novel view synthesis of a dynamic scene from monocular video. ICCV (2021)
47. Truong, P., Rakotosaona, M.J., Manhardt, F., Tombari, F.: Sparf: Neural radiance fields from sparse and noisy poses. CVPR (2023)
48. Wang, D., Chandran, P., Zoss, G., Bradley, D., Gotardo, P.F.U.: Morf: Morphable radiance fields for multiview neural head modeling. In: ACM SIGGRAPH 2022 Conference Proceedings (2022)
49. Wang, J., Xie, J.C., Li, X., Xu, F., Pun, C.M., Gao, H.: Gaussianhead: High-fidelity head avatars with learnable gaussian derivation. ArXiv:2312.01632 (2024)
50. Weng, C.Y., Curless, B., Srinivasan, P.P., Barron, J.T., Kemelmacher-Shlizerman, I.: HumanNeRF: Free-viewpoint rendering of moving people from monocular video. CVPR (2022)
51. Wu, G., Yi, T., Fang, J., Xie, L., Zhang, X., Wei, W., Liu, W., Tian, Q., Wang, X.: 4d gaussian splatting for real-time dynamic scene rendering. CVPR (2024)
52. Xiang, J., Gao, X., Guo, Y., Zhang, J.: Flashavatar: High-fidelity head avatar with efficient gaussian embedding. CVPR (2024)
53. Xu, B., Wang, N., Chen, T., Li, M.: Empirical evaluation of rectified activations in convolutional network (2015)
54. Xu, Y., Chen, B., Li, Z., Zhang, H., Wang, L., Zheng, Z., Liu, Y.: Gaussian head avatar: Ultra high-fidelity head avatar via dynamic gaussians. CVPR (2024)
55. Xu, Y., Wang, L., Zhao, X., Zhang, H., Liu, Y.: Avatarmav: Fast 3d head avatar reconstruction using motion-aware neural voxels. ACM SIGGRAPH (2023)
56. Yang, Z., Gao, X., Zhou, W., Jiao, S., Zhang, Y., Jin, X.: Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. CVPR (2024)
57. Yifan, W., Serena, F., Wu, S., Öztireli, C., Sorkine-Hornung, O.: Differentiable surface splatting for point-based geometry processing. ACM TOG (Proceedings of ACM SIGGRAPH ASIA) (2019)
58. Yu, C., Gao, C., Wang, J., Yu, G., Shen, C., Sang, N.: Bisenet v2: Bilateral network with guided aggregation for real-time semantic segmentation. IJCV (2021)
59. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. CVPR (2018)
60. Zheng, Y., Abrevaya, V.F., Bühler, M.C., Chen, X., Black, M.J., Hilliges, O.: I M Avatar: Implicit morphable head avatars from videos. CVPR (2022)
61. Zheng, Y., Yifan, W., Wetzstein, G., Black, M.J., Hilliges, O.: Pointavatar: Deformable point-based head avatars from videos. CVPR (2023)
62. Zielonka, W., Bolkart, T., Thies, J.: Instant volumetric head avatars. CVPR (2023)