3D Gaussian Blendshapes for Head Avatar Animation

Shengjie Ma
State Key Lab of CAD&CG, Zhejiang University
Hangzhou, China
qtdysjj@gmail.com

Yanlin Weng
State Key Lab of CAD&CG, Zhejiang University
Hangzhou, China
weng@cad.zju.edu.cn

Tianjia Shao
State Key Lab of CAD&CG, Zhejiang University
Hangzhou, China
tjshao@zju.edu.cn

Kun Zhou*
State Key Lab of CAD&CG, Zhejiang University
Hangzhou, China
kunzhou@acm.org

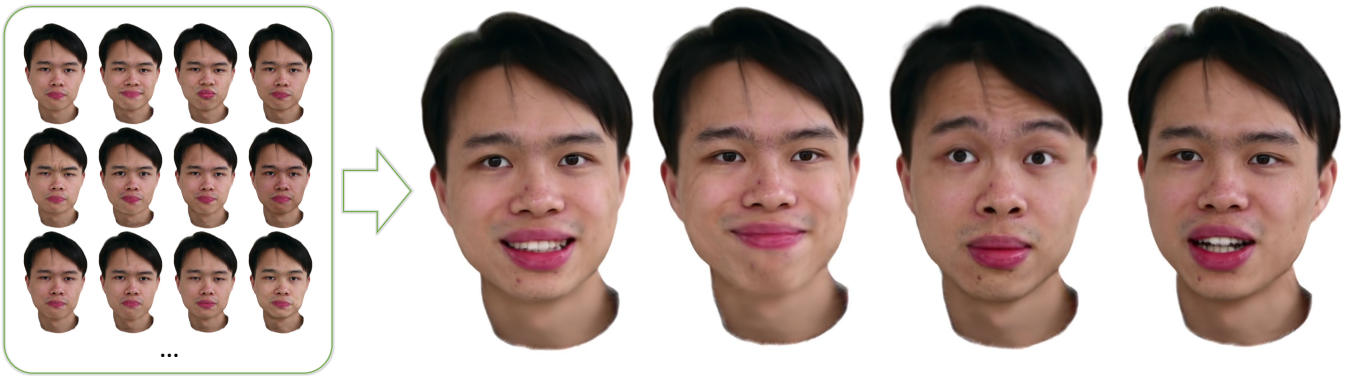


Figure 1: Our 3D Gaussian blendshapes are analogous to mesh blendshapes in classical parametric face models, which can be linearly blended with expressions coefficients to synthesize photo-realistic avatar animations in real time (370fps).

ABSTRACT

We introduce 3D Gaussian blendshapes for modeling photorealistic head avatars. Taking a monocular video as input, we learn a base head model of neutral expression, along with a group of expression blendshapes, each of which corresponds to a basis expression in classical parametric face models. Both the neutral model and expression blendshapes are represented as 3D Gaussians, which contain a few properties to depict the avatar appearance. The avatar model of an arbitrary expression can be effectively generated by combining the neutral model and expression blendshapes through linear blending of Gaussians with the expression coefficients. High-fidelity head avatar animations can be synthesized in real time using Gaussian splatting. Compared to state-of-the-art methods, our Gaussian blendshape representation better captures high-frequency details exhibited in input video, and achieves superior rendering performance.

CCS CONCEPTS

• **Computing methodologies** → **Reconstruction**; **Point-based models**.

KEYWORDS

Parametric face models, facial animation, facial tracking, facial reenactment

ACM Reference Format:
Shengjie Ma, Yanlin Weng, Tianjia Shao, and Kun Zhou. 2024. 3D Gaussian Blendshapes for Head Avatar Animation. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers '24 (SIGGRAPH Conference Papers '24), July 27–August 01, 2024, Denver, CO, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3641519.3657462

*Corresponding author

SIGGRAPH Conference Papers '24, July 27–August 01, 2024, Denver, CO, USA

ACM ISBN 979-8-4007-0525-0/24/07
https://doi.org/10.1145/3641519.3657462

1 INTRODUCTION

Reconstructing and animating 3D human heads has been a long studied problem in computer graphics and computer vision, which is the key technology in a variety of applications such as telepresence, VR/AR and movies. Most recently, head avatars based on neural radiance fields (NeRF) [Mildenhall et al. 2020] demonstrate great potential in synthesizing photorealistic images. These techniques achieve dynamic avatar control typically by conditioning

NeRFs on a parametric head model [Zielonka et al. 2023] or expression codes [Gafni et al. 2021]. Gao et al. [2022] and Zheng et al. [2022] instead propose to construct a set of NeRF blendshapes and linearly blend them to animate the avatar.

The blendshape model is a classic representation for avatar animation. It consists of a group of 3D meshes, each of which corresponds to a basis expression. The face shape of an arbitrary expression can be efficiently computed by linearly blending the basis meshes with corresponding expression coefficients. The advantages of easy-to-control and high efficiency make blendshape models the most popular representation in professional animation production [Lewis et al. 2014] as well as consumer avatar applications (e.g., iPhone Memoji) [Weng et al. 2014].

In this paper, we introduce a 3D Gaussian blendshape representation for constructing and animating head avatars. We build the representation upon 3D Gaussian splatting (3DGS) [Kerbl et al. 2023], which represents the radiance field of a static scene as 3D Gaussians and provides compelling quality and speed in novel view synthesis. Our representation consists of a base model of neutral expression and a group of expression blendshapes, all represented as 3D Gaussians. Each Gaussian contains a few properties (e.g., position, rotation and colors) as in 3DGS and depicts the appearance of the head avatar. Each Gaussian blendshape corresponds to a mesh blendshape of traditional parametric face models [Cao et al. 2014b; Li et al. 2017] and has the same semantic meaning. A Gaussian head model of an arbitrary expression can be generated by blending the Gaussian blendshapes with the expression coefficients, which is rendered to high-fidelity images in real time using Gaussian splatting. The motion parameters tracked by previous face tracking algorithms (e.g., [Cao et al. 2014a; Zielonka et al. 2022]) can be used to drive the Gaussian blendshapes to produce head avatar animations.

We propose to learn the Gaussian blendshape representation from a monocular video. We use previous methods to construct the mesh blendshapes from the input video, and distribute a number of Gaussians on the mesh surfaces as an initialization. We then jointly optimize all Gaussian properties. As Gaussian blendshapes are driven by the same expression coefficients for mesh blendshapes, each Gaussian blendshape must be semantically consistent with its corresponding mesh blendshape, i.e., the differences between the Gaussian blendshape and neutral model should be consistent with the differences between the corresponding mesh blendshape and neutral mesh. Directly optimizing Gaussian properties without considering blendshape consistency causes overfitting and artifacts for novel expressions unseen in training. To this end, we present an effective strategy to guide the Gaussian optimization to follow the consistency requirement. Specifically, we introduce an intermediate variable to formulate the Gaussian difference as terms proportional to the mesh difference. By optimizing this intermediate variable directly during training, we produce Gaussian blendshapes differing from the neutral model in a consistent way that mesh blendshapes differ from the neutral mesh.

Extensive experiments demonstrate that our Gaussian blendshape method outperforms state-of-the-art methods [Gao et al. 2022; Zheng et al. 2023; Zielonka et al. 2023] in synthesizing high-fidelity head avatar animations that best capture high-frequency details observed in input video, and achieving significantly faster speeds in avatar animation and rendering (see Fig. 1).

2 RELATED WORK

Researchers have proposed various representations for head avatars. Early works employ explicit 3D mesh representation to reconstruct the 3D shape and appearance from images. The seminal work [Blanz and Vetter 1999] proposes the 3D Morphable Model (3DMM) to model the face shape and texture on a low-dimensional linear subspace. There are many follow-up works along this direction such as full-head models [Ploumpis et al. 2021], and deep non-linear models [Tran and Liu 2018]. The 3D mesh representation is also used to build riggable heads for head animation [Bai et al. 2021; Chaudhuri et al. 2020; Hu et al. 2017]. To generate detailed animations, researchers further propose image-based dynamic avatars controlling the full head with hair and headwear [Cao et al. 2016], or additionally reconstruct fine-level correctives [Feng et al. 2021; Garrido et al. 2016; Ichim et al. 2015; Yang et al. 2020].

In order to achieve high realism rendering, recent approaches utilize neural radiance fields (NeRF) [Mildenhall et al. 2020] to implicitly represent head avatars and have achieved impressive results [Gafni et al. 2021; Grassal et al. 2022; Jiang et al. 2022; Lombardi et al. 2021; Xu et al. 2022, 2023b,c; Zheng et al. 2022]. For instance, i3DMM [Yenamandra et al. 2021] presents the first neural implicit function based on the 3D morphable model of full heads. HeadNerf [Hong et al. 2022] introduces a NeRF-based parametric head model that integrates the neural radiance field to the parametric representation of the head. The state-of-the-art work INSTA [Zielonka et al. 2023] models a dynamic neural radiance field based on Instant-NGP [Müller et al. 2022] embedded around a parametric face model. It is able to reconstruct a head avatar in less than 10 minutes. PointAvatar [Zheng et al. 2023] presents a point-based representation and learns a deformation field based on FLAME's expression vectors to drive the points. NeRFBlendshape [Gao et al. 2022] constructs NeRF-based blendshape models for semantic animation control and photorealistic rendering by combining multi-level voxel fields with expression coefficients.

Many concurrent works have been proposed to apply the 3D Gaussian representation introduced by [Kerbl et al. 2023] to construct head avatars (e.g., [Chen et al. 2023; Dhamo et al. 2023; Qian et al. 2023; Saito et al. 2023; Wang et al. 2023; Xiang et al. 2023; Xu et al. 2023a]). Most of them use the 3D Gaussian representation together with neural networks. For example, GaussianHead [Wang et al. 2023] uses Multi-layer Perceptrons (MLPs) to decode the dynamic geometry and radiance parameters of Gaussians. FlashAvatar [Xiang et al. 2023] attaches Gaussians on a mesh with learnable offsets, which are represented as MLPs. Saito et al. [2023] construct relightable head avatars by using networks to decode the parameters of 3D Gaussians and learnable radiance transfer functions. To our knowledge, none of concurrent works introduce the idea of Gaussian blendshapes as in our paper. A unique advantage of our method is that it only requires linear blending of Gaussian blendshapes to construct a head avatar of arbitrary expressions, which brings significant benefits in both training and runtime performance. The method closest to our work in terms of performance is FlashAvatar [Xiang et al. 2023], which achieves 300fps for 10k

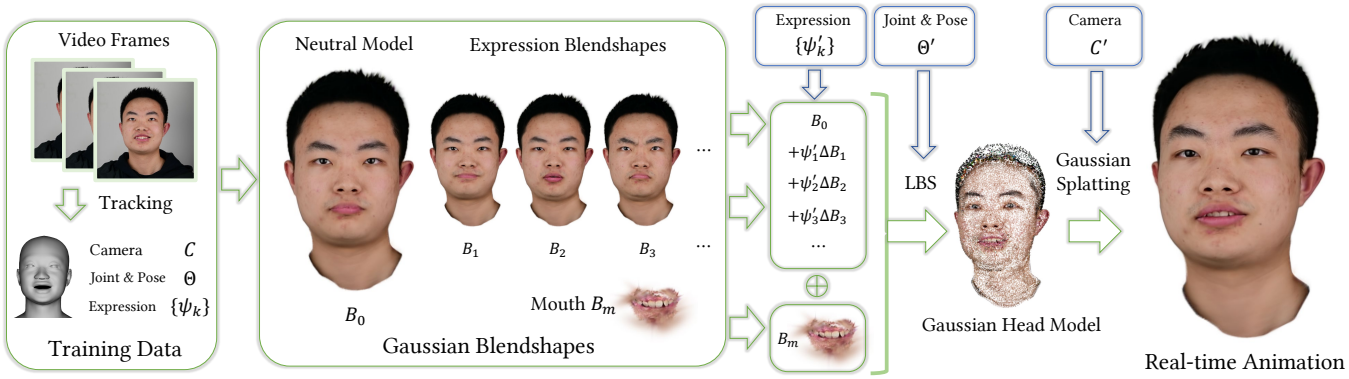


Figure 2: Overview of our method. Taking a monocular video as input, our method learns a Gaussian blendshape representation of a head avatar, which consists of a neutral model B_0, a group of expression blendshapes $\{B_1, B_2, ..., B_K\}$, and the mouth interior model B_m, all represented as 3D Gaussians. Avatar models of arbitrary expressions and poses can be generated by linear blending with expression coefficients $\{\psi'_k\}$ and linear blend skinning with joint and pose parameters Θ', from which we render high-fidelity images in real time using Gaussian splatting.

Gaussians and degrades to ~100fps for 50k Gaussians, while we achieve 370fps for 70k Gaussians.

3 METHOD

3.1 3D Gaussian BlendShapes

Our Gaussian blendshape representation consists of a neutral base model B_0 and a group of expression blendshapes $\{B_1, B_2, ..., B_K\}$, all represented as a set of 3D Gaussians, each of which has a few basic properties (i.e., position $\mathbf{x}$, opacity α, rotation $\mathbf{q}$, scale $\mathbf{s}$ and spherical harmonics coefficients SH) as in 3DGS [Kerbl et al. 2023]. Each Gaussian of B_0 also has a set of blend weights $\mathbf{w}$ for joint and pose control. There is a one-to-one correspondence between the Gaussians of B_0 and each blendshape B_k. The deviation of B_k from B_0 can be defined as the difference between their Gaussian properties, $\Delta B_k = B_k - B_0$. The head avatar model of an arbitrary expression is computed as:

$$B^{\psi} = B_0 + \sum_{k=1}^{K} \psi_k \Delta B_k, \tag{1}$$

where $\{\psi_k\}$ are the expression coefficients.

Currently we use the Principal Component Analysis (PCA) based blendshape model FLAME [Li et al. 2017], although other muscle-inspired blendshapes such as the Facial Action Coding System (FACS) based model FaceWarehouse [Cao et al. 2014b] can be also employed. Besides facial expression control, FLAME also provides joint and pose parameters, Θ, for controlling the motions of head, jaw, eyeballs and eyelids, which are used with linear blend skinning (LBS) to transform the head avatar model (i.e., its Gaussians): $B^{\psi *} = LBS(B^{\psi}, \Theta)$, where the blend weights associated with Gaussians of B_0 are used.

Mouth Interior Gaussians. The motions of mouth interior and hair are usually not affected by facial expressions, and thus not covered in the FLAME mesh, neither the blendshape model described above. Hair can move with the head rigidly, while the motion of teeth is controlled by the jaw joint in FLAME. We find that in practice the blendshape Gaussians generated in our training are able to model hair well, but the mouth interior results are not good enough. We thus define a separate set of Gaussians for mouth interior, B_m, which move with the jaw joint in FLAME. The properties of these mouth Gaussians do not change with expressions, but are only transformed with the jaw joint, i.e., $B^*_m = LBS(B_m, \Theta)$.

Finally, the transformed Gaussian model $(B^{\psi *}, B^*_m)$ can be rendered to high-fidelity images I_r in real time using Gaussian splatting. Fig. 2 shows the overview of our method.

3.2 Training

Data Preparation. Following [Zielonka et al. 2023], we use the face tracker of [Zielonka et al. 2022] to compute the FLAME meshes of neutral expression and $K = 50$ basis expressions, as well as the camera parameters, joint and pose parameters, and expression coefficients, for each video frame. We also extract the foreground head mask for each input frame.

Initialization. We first initialize the neutral model B_0, expression blendshapes $\{B_k\}$, as well as the mouth interior Gaussians B_m. For B_0, we distribute a number of points on the neutral FLAME mesh M_0 using Poisson disk sampling [Bowers et al. 2010], and use them as the initialization of Gaussian positions. Other Gaussian properties are initialized as in 3DGS [Kerbl et al. 2023]. For each Gaussian, we also find its closest triangle on M_0, and compute its LBS blend weights as the linear interpolation of blend weights of the triangle vertices. To initialize the mouth interior Gaussians B_m, we use two pre-defined billboards to represent the upper and lower teeth, which are sampled to Gaussians using Poisson disk sampling. The upper teeth Gaussians are rigidly bound to the back of the head, while lower teeth Gaussians are bound to the vertex having the largest skinning weight for the jaw joint.

To initialize the expression blendshape B_k, we transform each Gaussian of B_0 using the deformation gradients [Sumner and Popović 2004] from M_0 to the expression FLAME mesh M_k. Specifically, for each neutral Gaussian G^i_0, we compute the affine transformation

from its closest triangle on M_0 to the corresponding triangle on M_k, and extract the rotation component [Shoemake and Duff 1992], which is applied to the position, rotation and spherical harmonics (SH) coefficients of G_0^i to yield the corresponding Gaussian G_k^i of expression blendshape B_k. Note that we omit the scale component as we find the transformation is very close to rigid. The scale and opacity properties of G_k^i are kept the same as those of G_0^i. In this way, we can construct each expression blendshape B_k from B_0, as well as their difference $\Delta B_k = B_k - B_0$.

Optimization. After initialization, we jointly optimize B_0, $\{\Delta B_k\}$, and B_m. For each video frame, we reconstruct the Gaussian head model B_ψ by linearly blending B_0 and $\{\Delta B_k\}$ with the tracked expression coefficients according to Eq. (1), and then transform B_ψ and B_m using LBS with the tracked joint and pose parameters: $B^{\psi*} = LBS(B^\psi, \Theta)$, $B_m^* = LBS(B_m, \Theta)$. Finally, we get the rendered image from $B^{\psi*}$ and B_m^* using Gaussian splatting. The optimization process is similar to 3DGS [Kerbl et al. 2023], which also involves adaptive density control steps of adding and removing Gaussians.

During optimization, a crucial thing to avoid overfitting is to preserve the semantic consistency between each Gaussian blendshape B_k and its corresponding mesh blendshape M_k. As aforementioned, the Gaussian blendshapes are blended using the same tracked expression coefficients based on the parametric mesh model of FLAME, in both training and runtime computations. To ensure the semantic validity of such blending calculation, the difference between B_k and B_0 (i.e., ΔB_k) must be consistent with the difference between M_k and M_0 (i.e., ΔM_k), which means in head regions having large vertex position differences between M_k and M_0, the Gaussian differences between B_k and B_0 should also be large, and small otherwise. Directly optimizing $\{\Delta B_k\}$ without such consistency consideration will lead to overfitting, where apparent artifacts easily occur on novel expression coefficients unseen in the training images (see Fig. 4 for examples).

However, unlike ΔM_k only containing vertex position displacements, ΔB_k contains different kinds of properties, such as position, rotation, and color. It is thus difficult to design a loss function term to explicitly enforce consistency between ΔB_k and ΔM_k, while not sacrificing the image loss. Instead, we propose a simple yet effective strategy to guide the Gaussian optimization to implicitly follow the consistency requirement. Specifically, for each Gaussian G_i, let $\Delta G_{i,k}$ be the difference between its properties in B_k and B_0. We introduce an intermediate variable, $\Delta\widehat{G}_{i,k}$, to formulate $\Delta G_{i,k}$ as:

$$\Delta G_{i,k} = \Delta G_{i,k}^{init} + max(f(d_{i,k}), 0)\Delta\widehat{G}_{i,k}, \tag{2}$$

where $\Delta G_{i,k}^{init}$ is the initial value of $\Delta G_{i,k}$ calculated in the aforementioned initialization stage and regarded as a constant during optimization, and $d_{i,k}$ is the magnitude of position displacement of the surface point closest to G_i, from M_0 to M_k. The linear function $f(x) = (x - \epsilon)/(\tilde{d} - \epsilon)$ scales the maximum magnitude of position difference $\tilde{d}$ between M_k and M_0 to 1, and a threshold magnitude $\epsilon = 0.00001$ to 0. The *max* function is necessary to avoid negative scaling values for positional displacement magnitudes below ϵ.

Eq. (2) essentially represents the actual Gaussian difference $\Delta G_{i,k}$ as the sum of its initial value and the scaled value of $\Delta\widehat{G}_{i,k}$ according to its corresponding positional displacement in mesh blendshapes,

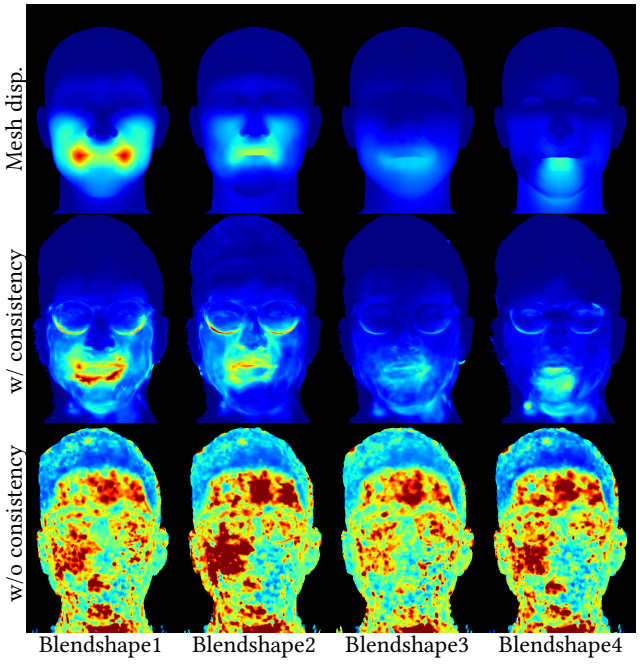


Figure 3: The impact of the blendshape consistency on the optimization of expression blendshapes. The first row shows the displacement magnitude between M_k and M_0. The second and the third rows show the magnitude of optimized ΔB_k with or without blendshape consistency.

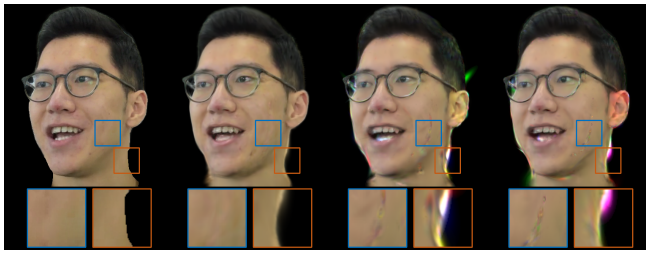


Figure 4: Ablation study on blendshape consistency. The optimization without blendshape consistency leads to apparent artifacts like dirty color and glitch in both interior and boundary areas. Enforcing blendshape consistency only on Gaussian positions also leads to poor results.

which effectively correlates Gaussian differences with position displacements. The initial Gaussian difference $\Delta G_{i,k}^{init}$ is proportional to the position displacement of mesh blendshapes, as it is computed using the deformation gradients from M_0 to M_k. Scaling $\Delta\widehat{G}_{i,k}$ according to positional displacement ensures that the Gaussian difference $\Delta G_{i,k}$ is updated at a rate proportional to the position displacement. Please note for Gaussians with positional displacement magnitudes below ϵ, the second term in Eq. (2) vanishes to

Table 1: Quantitative comparisons between INSTA [Zielonka et al. 2023], PointAvatar [Zheng et al. 2023], and our method.

Datasets		INSTA dataset								Our dataset			
		bala	biden	justin	malte_1	marcel	nf_01	nf_03	wojtek_1	subject1	subject2	subject3	subject4
PSNR ↑	INSTA	28.66	28.38	29.74	26.27	23.75	25.89	26.10	29.84	28.88	28.16	28.60	30.83
	PointAvatar	29.60	31.72	32.31	27.46	24.60	28.34	29.82	31.94	31.43	32.57	30.95	32.57
	Ours	33.34	32.48	32.49	28.56	26.61	27.92	28.62	32.39	32.87	32.55	31.54	34.03
SSIM ↑	INSTA	0.9130	0.9484	0.9530	0.9262	0.9133	0.9246	0.9129	0.9457	0.9195	0.9443	0.9078	0.9445
	PointAvatar	0.9099	0.9565	0.9595	0.9225	0.9121	0.9278	0.9208	0.9502	0.9219	0.9463	0.9062	0.9433
	Ours	0.9490	0.9672	0.9696	0.9461	0.9348	0.9448	0.9381	0.9645	0.9384	0.9577	0.9268	0.9646
LPIPS ↓	INSTA	0.0817	0.0545	0.0614	0.0751	0.1540	0.1285	0.1137	0.0588	0.1536	0.1208	0.1733	0.1144
	PointAvatar	0.0821	0.0535	0.0649	0.0718	0.1574	0.1350	0.1221	0.0661	0.1568	0.1190	0.1715	0.1285
	Ours	0.0772	0.0522	0.0631	0.0703	0.1391	0.1174	0.0965	0.0595	0.1520	0.1197	0.1689	0.1075

0 and $\Delta G_{i,k}$ is always equal to $\Delta G_{i,k}^{init}$, which is inherently proportional to the positional displacement.

Instead of optimizing $\Delta G_{i,k}$, we directly optimize $\Delta \widehat{G}_{i,k}$ using the loss functions described in the following section. Specifically, $\Delta \widehat{G}_{i,k}$ is initialized to 0. Each time $\Delta \widehat{G}_{i,k}$ is updated, we calculate $\Delta G_{i,k}$ according to Eq. (2), from which the avatar model is constructed. The avatar model is then rendered to an image through Gaussian splatting, which is used in loss function computation.

In this way, we effectively guide the Gaussian differences to change consistently with positional displacements, leading to optimized Gaussian blendshapes with strong semantic consistency with mesh blendshapes (see Fig. 3).

3.3 Loss Functions

The optimization goal is to minimize the image loss between the rendering and input, under some regularization constraints. The first loss is the image loss as in 3DGS [Kerbl et al. 2023], consisting of the L_1 differences between the rendered images and the video frames and a D-SSIM term:

$$L_{rgb} = (1-\lambda)L_1 + \lambda L_{D-SSIM} \tag{3}$$

with $\lambda = 0.2$.

We also design an alpha loss to constrain the Gaussians to stay within the head region. We perform Gaussian splatting to get the accumulated opacity image I_α, and compare it with the foreground head mask $Mask_h$. The alpha loss is defined as:

$$L_\alpha = \frac{1}{F}\sum_{i=1}^{F}(||(I_\alpha^i - Mask_h^i)||_2), \tag{4}$$

where F is the frame number.

We further introduce a regularization loss to constrain the mouth interior Gaussians to stay within a pre-defined volume of the mouth. Specifically, we compute the signed distance for each Gaussian to the volume boundary and apply an L_2 loss to retract it when it goes out of the volume. The regularization loss is defined as:

$$L_{reg} = \frac{1}{N}\sum_{i=1}^{N}(||max(SDF(\mathbf{x}_i, V), 0)||_2^2), \tag{5}$$

where V is the pre-defined cylindrical volume, $\mathbf{x}_i$ is the Gaussian position and N is the number of mouth interior Gaussians. The overall loss function is defined as:

$$L = \lambda_1 L_{rgb} + \lambda_2 L_\alpha + \lambda_3 L_{reg}. \tag{6}$$

We set $\lambda_1 = 1, \lambda_2 = 10, \lambda_3 = 100$ by default.

Table 2: Quantitative comparisons between NeRFBlendShape [Gao et al. 2022] and our method.

Datasets		NeRFBlendShape dataset					
		id1	id2	id3	id4	id5	id6
PSNR ↑	NeRFBlendShape	32.12	32.25	37.25	36.86	34.26	35.74
	Ours(w/o LPIPS)	33.13	33.23	39.83	38.34	35.58	36.59
	Ours(w/ LPIPS)	33.09	33.15	39.64	38.17	35.46	36.46
SSIM ↑	NeRFBlendShape	0.9412	0.9369	0.9750	0.9791	0.9541	0.9786
	Ours(w/o LPIPS)	0.9532	0.9473	0.9836	0.9846	0.9635	0.9814
	Ours(w/ LPIPS)	0.9522	0.9457	0.9828	0.9837	0.9625	0.9806
LPIPS ↓	NeRFBlendShape	0.0715	0.0756	0.0436	0.0460	0.0448	0.0366
	Ours(w/o LPIPS)	0.0862	0.0937	0.0486	0.0495	0.0538	0.0414
	Ours(w/ LPIPS)	0.0550	0.0620	0.0359	0.0334	0.0381	0.0303

3.4 Implementation Details

We implement our method using Pytorch. The Adam solver [Kingma and Ba 2015] is employed for parameter optimization. The learning rates are 3.2×10^{-7}, 5×10^{-5}, 5×10^{-4}, 1×10^{-4}, 1.25×10^{-3} respectively for the Gaussian properties $\{\mathbf{x}_k, \alpha_k, \mathbf{s}_k, \mathbf{q}_k, SH_k\}$. The initially sampled Gaussian number is 50k for the neutral model, and 14k for the mouth interior Gaussians.

The training is conducted on an A800 GPU and testing is conducted on an RTX 4090 GPU. We also build a C++/CUDA interactive viewer following 3DGS [Kerbl et al. 2023] and use it to measure our runtime frame rates.

As Gaussian positions frequently change during optimization, we need to efficiently update their LBS blend weights $\mathbf{w}$ and the positional displacements of nearest points $\{d_{i,k}\}$. We precompute and store these values on a 3D grid of $256 \times 256 \times 256$ surrounding the neutral mesh M_0. The values of an arbitrary Gaussian can be effectively computed as the linear blending of the values of eight grid points nearest to the Gaussian center.

4 RESULTS

4.1 Baselines and Datasets

We compare our method with state-of-the-art methods, NeRF-based INSTA [Zielonka et al. 2023] and point-based PointAvatar [Zheng

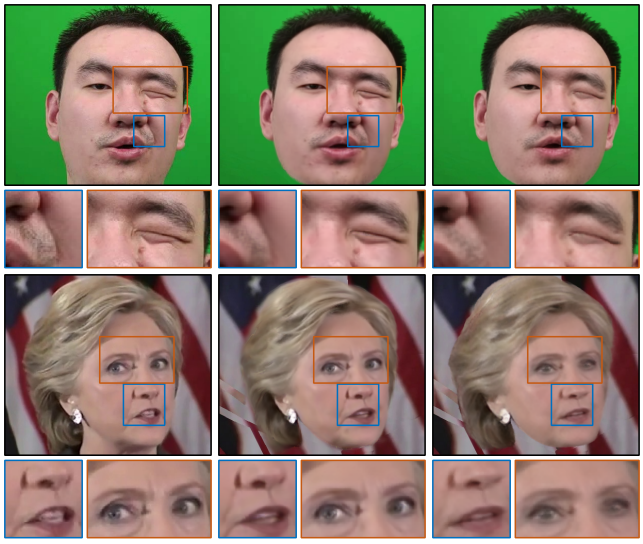


Figure 5: Qualitative comparisons with NeRFBlendShape [Gao et al. 2022]. Our method more faithfully captures fine facial details (e.g., wrinkles around the eyes and nose), and better recovers the eyeball movement. YouTube video ID is -yHgE9W699w for Hillary Clinton.

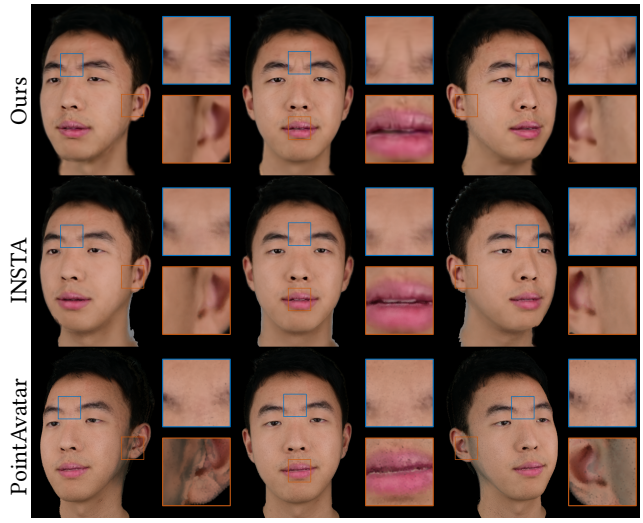


Figure 6: Qualitative comparisons for novel view extrapolation. Our method produces better results with fine details under novel views.

Figure 7: Results of cross-identity reenactment. YouTube video ID is mKHgXHKbJUE for Justin Trudeau.

Figure 8: Demonstration of mouth interior Gaussians.

Table 3: Performance comparisons. We perform training for all methods on an A800 GPU. Testing is done on a RTX 4090 GPU for INSTA, NeRFBlendshape, and our method, but on an A800 GPU for PointAvatar due to out-of-memory errors on the RTX 4090 GPU. The rendering resolution is 512×512. Note our running time includes both animation (i.e., linear blending and LBS transformation) and rendering, and our performance is insensitive to the rendering resolution. We also report the peak memory consumption during training and runtime computation.

	Training	Runtime	Mem. (train)	Mem. (runtime)
INSTA	10min	70fps	16G	4G
PointAvatar	3.5h	5fps*	40G	32G*
NeRFBlendShape	20min	26fps	7G	2G
Ours	25min	370fps	14G	2G

et al. 2023], on the INSTA dataset and our own dataset. We hold the last 350 frames of each video as the test set for self-reenactment similar to INSTA. The training data preparation time is about 12 hours for 4500 frames as in INSTA. We also compare our method with NeRFBlendShape [Gao et al. 2022] on their public dataset consisting of eight videos. The last 500 frames of each video are reserved for test. We reduce the alpha weight λ_2 to 1 for the entire dataset due to the relatively inaccurate binary foreground mask provided, which we find slightly sharpens the hair around the contour.

Our own dataset consists of four subjects, captured in an indoor environment using a Nikon D850 camera. For each subject, we collected a 3 minute video in 1080p, which was then cropped and resized to 1024×1024 resolution.

Figure 9: Ablation study on blendshape optimization. The first row shows the results with the initial values of $\{\Delta B_k\}$ kept unchanged during optimization. The second row shows our results with joint optimization of B_0, $\{\Delta B_k\}$, and B_m, which better capture intricate details of facial animations.

4.2 Comparisons

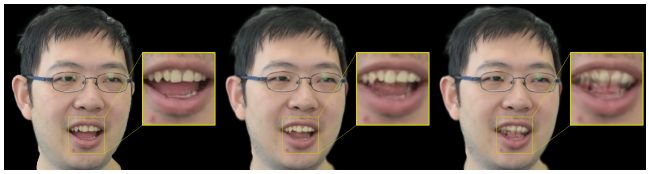
We evaluate the results using standard metrics including PSNR, SSIM and LPIPS [Zhang et al. 2018]. As show in the quantitative results Table 1, in most cases, our method outperforms INSTA and pointAvatar in terms of PSNR and LPIPS, while the SSIM of our method is consistently better. As shown in Table 2, our method also surpasses NeRFBlendShape in terms of PSNR and SSIM. Note that NeRFBlendShape utilizes the LPIPS loss during training, leading to better LPIPS. When we add the LPIPS loss with a weight of 0.05 in training, our method also performs better.

The qualitative comparisons are shown in Fig. 13 and Fig. 5. Compared with INSTA and PointAvatar, our method is better at capturing high-frequency details observed in the training video, such as wrinkles, teeth, and specular highlights of glasses and noses. Compared with NeRFBlendShape, our method also synthesizes images of higher quality with sharper details. Moreover, our method better recovers the eyeball movement than NeRFBlendShape, thanks to the eyeball motion control provided in FLAME.

Our method also performs better in novel view extrapolation (Fig. 6), while PointAvatar [Zheng et al. 2023] suffers from artifacts around the ear region, and both INSTA [Zielonka et al. 2023] and PointAvatar tend to lose high-frequency details.

We show qualitative results on cross-identity reenactment in Fig. 7. Our method faithfully transfers expressions while maintaining the personal attributes of the target subject.

The training and runtime performance comparison is shown in Table 3. Our method is able to synthesize facial animations at 370fps, over 5× faster than INSTA and about 14× faster than NeRFBlendshape. Our training time is comparable to NeRFBlendshape.



Ground truth Ours w/o mouth Gaussians

Figure 10: Ablation study on mouth interior Gaussians. We find that without the mouth interior Gaussians, the teeth may not be well modeled, leading to blurry or ghosting artifacts.

Ours FlashAvatar NeRFBlendShape

Figure 11: Failure cases of side view rendering.

4.3 Gaussian Blendshape Visualization

Fig. 12 demonstrates eight Gaussian blendshapes of a subject and their corresponding mesh blendshapes. Please refer to the supplementary video for live demonstration. The effect of our mouth Gaussians is show in Fig. 8.

4.4 Ablation Studies

Blendshape consistency. Fig. 3 visualizes the magnitudes of ΔM_k and ΔB_k. As you can see, imposing blendshape consistency during optimization does produce Gaussian blenshapes $\{B_k\}$ differing from the base model B_0 in a consistent way that mesh blendshapes $\{M_k\}$ differ from the base mesh M_0. Fig. 4 demonstrates the importance of blendshape consistency between Gaussian and mesh blendshapes. The optimization without considering blendshape consistency on all Gaussian properties results in apparent artifacts on the face and head boundary under novel expressions. Note that the magnitude of ΔB_k visualized in Fig. 3 only represents the difference between each individual blendshape B_k and the base model B_0, and thus does not necessarily correspond to errors in rendered images of the avatar model, which is the linear blending of all blendshapes.

Optimization of $\{\Delta B_k\}$. The initialization stage of our training constructs Gaussian blendshapes $\{B_k\}$ by transforming Gaussians of B_0 using the deformation gradients from M_0 to $\{M_k\}$, resulting in Gaussian differences $\{\Delta B_k\}$ consistent with mesh differences $\{\Delta M_k\}$. Keeping the initial values of $\{\Delta B_k\}$ unchanged during optimization and only optimizing B_0 and B_m can produce reasonable results, but fail to capture the fine details of facial animations, as shown in Fig. 9.

Mouth interior Gaussians. We evaluate the effect of mouth interior Gaussians by comparing our full result with the one using only the neutral model and expression blendshapes to represent the

whole head (Fig. 10). We can see that apparent artifacts and blurring occur around the mouth region, demonstrating the necessity of mouth interior Gaussians.

5 CONCLUSION

We present a novel 3D Gaussian blendshape representation for animating photorealistic head avatars. We also introduce an optimization process to learn the Gaussian blendshapes from a monocular video, which are semantically consistent with their corresponding mesh blendshapes. Our method outperforms state-of-the-art NeRF and point based methods in producing avatar animations of superior quality at significantly faster speeds, while the training and memory cost is moderate.

Limitation and Discussion. Our constructed avatar models can exhibit apparent artifacts in side-view rendering if the training data does not contain side views. As shown in Fig. 11, this is also a limitation in previous NeRF-based methods and concurrent Gaussian-based methods. Improving the generalization capability to handle dramatically novel views is an open problem for further research. The extrapolation capabilities of our model are also restricted by its linear blending nature of the model, leading to potential failures when processing exaggerated expressions unseen in the training set. Another limitation is that the model cannot represent deformable hair, which is an interesting direction for future investigation. It is worth noting that there is a risk of misuse of our method (e.g., the so-called DeepFakes). We strongly oppose applying our work to produce fake images or videos of individuals with the intention of spreading false information or damaging their reputations.

ACKNOWLEDGMENTS

This work is supported by the National Key Research and Development Program of China (No. 2022YFF0902302), NSF China (No. 62172357 & 62322209), and the XPLORER PRIZE. The source code is available at https://gapszju.github.io/GaussianBlendshape.

REFERENCES

Ziqian Bai, Zhaopeng Cui, Xiaoming Liu, and Ping Tan. 2021. Riggable 3D Face Reconstruction via In-Network Optimization. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*. Computer Vision Foundation / IEEE, 6216–6225. https://doi.org/10.1109/CVPR46437.2021.00615

Volker Blanz and Thomas Vetter. 1999. A Morphable Model for the Synthesis of 3D Faces. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1999, Los Angeles, CA, USA, August 8-13, 1999*, Warren N. Waggenspack (Ed.). ACM, 187–194. https://dl.acm.org/citation.cfm?id=311556

John C. Bowers, Rui Wang, Li-Yi Wei, and David Maletz. 2010. Parallel Poisson disk sampling with spectrum analysis on surfaces. *ACM Trans. Graph.* 29, 6 (2010), 166. https://doi.org/10.1145/1882261.1866188

Chen Cao, Qiming Hou, and Kun Zhou. 2014a. Displaced dynamic expression regression for real-time facial tracking and animation. *ACM Trans. Graph.* 33, 4, Article 43 (jul 2014), 10 pages. https://doi.org/10.1145/2601097.2601204

Chen Cao, Yanlin Weng, Shun Zhou, Yiying Tong, and Kun Zhou. 2014b. FaceWarehouse: A 3D Facial Expression Database for Visual Computing. *IEEE Trans. Vis. Comput. Graph.* 20, 3 (2014), 413–425. https://doi.org/10.1109/TVCG.2013.249

Chen Cao, Hongzhi Wu, Yanlin Weng, Tianjia Shao, and Kun Zhou. 2016. Real-time facial animation with image-based dynamic avatars. *ACM Trans. Graph.* 35, 4 (2016), 126:1–126:12. https://doi.org/10.1145/2897824.2925873

Bindita Chaudhuri, Noranart Vesdapunt, Linda G. Shapiro, and Baoyuan Wang. 2020. Personalized Face Modeling for Improved Face Reconstruction and Motion Retargeting. In *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part V (Lecture Notes in Computer Science, Vol. 12350)*, Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm (Eds.). Springer, 142–160. https://doi.org/10.1007/978-3-030-58558-7_9

Yufan Chen, Lizhen Wang, Qijing Li, Hongjiang Xiao, Shengping Zhang, Hongxun Yao, and Yebin Liu. 2023. MonoGaussianAvatar: Monocular Gaussian Point-based Head Avatar. arXiv:2312.04558 [cs.CV]

Helisa Dhamo, Yinyu Nie, Arthur Moreau, Jifei Song, Richard Shaw, Yiren Zhou, and Eduardo Pérez-Pellitero. 2023. HeadGaS: Real-Time Animatable Head Avatars via 3D Gaussian Splatting. arXiv:2312.02902 [cs.CV]

Yao Feng, Haiwen Feng, Michael J. Black, and Timo Bolkart. 2021. Learning an animatable detailed 3D face model from in-the-wild images. *ACM Trans. Graph.* 40, 4 (2021), 88:1–88:13. https://doi.org/10.1145/3450626.3459936

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

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

Pablo Garrido, Michael Zollhöfer, Dan Casas, Levi Valgaerts, Kiran Varanasi, Patrick Pérez, and Christian Theobalt. 2016. Reconstruction of Personalized 3D Face Rigs from Monocular Video. *ACM Trans. Graph.* 35, 3 (2016), 28:1–28:15. https://doi.org/10.1145/2890493

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

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

Liwen Hu, Shunsuke Saito, Lingyu Wei, Koki Nagano, Jaewoo Seo, Jens Fursund, Iman Sadeghi, Carrie Sun, Yen-Chun Chen, and Hao Li. 2017. Avatar digitization from a single image for real-time rendering. *ACM Trans. Graph.* 36, 6 (2017), 195:1–195:14. https://doi.org/10.1145/3130800.31310887

Alexandru Eugen Ichim, Sofien Bouaziz, and Mark Pauly. 2015. Dynamic 3D avatar creation from hand-held video input. *ACM Trans. Graph.* 34, 4 (2015), 45:1–45:14. https://doi.org/10.1145/2766974

Boyi Jiang, Yang Hong, Hujun Bao, and Juyong Zhang. 2022. SelfRecon: Self Reconstruction Your Digital Avatar from Monocular Video. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*. IEEE, 5595–5605. https://doi.org/10.1109/CVPR52688.2022.00552

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 2023. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM Transactions on Graphics* 42, 4 (July 2023). https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.). http://arxiv.org/abs/1412.6980

J. P. Lewis, Ken Anjyo, Taehyun Rhee, Mengjie Zhang, Fred Pighin, and Zhigang Deng. 2014. Practice and Theory of Blendshape Facial Models. In *Eurographics 2014 - State of the Art Reports*, Sylvain Lefebvre and Michela Spagnuolo (Eds.). The Eurographics Association. https://doi.org/10.2312/egst.20141042

Tianye Li, Timo Bolkart, Michael J. Black, Hao Li, and Javier Romero. 2017. Learning a model of facial shape and expression from 4D scans. *ACM Trans. Graph.* 36, 6 (2017), 194:1–194:17. https://doi.org/10.1145/3130800.3130813

Stephen Lombardi, Tomas Simon, Gabriel Schwartz, Michael Zollhoefer, Yaser Sheikh, and Jason Saragih. 2021. Mixture of volumetric primitives for efficient neural rendering. *ACM Transactions on Graphics (ToG)* 40, 4 (2021), 1–13.

Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. 2020. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part I (Lecture Notes in Computer Science, Vol. 12346)*, Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm (Eds.). Springer, 405–421. https://doi.org/10.1007/978-3-030-58452-8_24

Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. 2022. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (ToG)* 41, 4 (2022), 1–15.

Stylianos Ploumpis, Evangelos Ververas, Eimear O' Sullivan, Stylianos Moschoglou, Haoyang Wang, Nick E. Pears, William A. P. Smith, Baris Gecer, and Stefanos Zafeiriou. 2021. Towards a Complete 3D Morphable Model of the Human Head. *IEEE Trans. Pattern Anal. Mach. Intell.* 43, 11 (2021), 4142–4160. https://doi.org/10.1109/TPAMI.2020.2991150

Shenhan Qian, Tobias Kirschstein, Liam Schoneveld, Davide Davoli, Simon Giebenhain, and Matthias Nießner. 2023. GaussianAvatars: Photorealistic Head Avatars with Rigged 3D Gaussians. arXiv:2312.02069 [cs.CV]

Shunsuke Saito, Gabriel Schwartz, Tomas Simon, Junxuan Li, and Giljoo Nam. 2023. Relightable Gaussian Codec Avatars. arXiv:2312.03704 [cs.GR]

Ken Shoemake and Tom Duff. 1992. Matrix animation and polar decomposition. In *Proceedings of the conference on Graphics interface*, Vol. 92. 258–264.

Robert W Sumner and Jovan Popović. 2004. Deformation transfer for triangle meshes. *ACM Transactions on graphics (TOG)* 23, 3 (2004), 399–405.

Luan Tran and Xiaoming Liu. 2018. Nonlinear 3D Face Morphable Model. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*. Computer Vision Foundation / IEEE Computer Society, 7346–7355. https://doi.org/10.1109/CVPR.2018.00767

Jie Wang, Jiu-Cheng Xie, Xianyan Li, Feng Xu, Chi-Man Pun, and Hao Gao. 2023. GaussianHead: High-fidelity Head Avatars with Learnable Gaussian Derivation. arXiv:arXiv:2312.01632 [cs.CV]

Yanlin Weng, Chen Cao, Qiming Hou, and Kun Zhou. 2014. Real-time facial animation on mobile devices. *Graphical Models* 76, 3 (2014), 172–179. https://doi.org/10.1016/j.gmod.2013.10.002 Computational Visual Media Conference 2013.

Jun Xiang, Xuan Gao, Yudong Guo, and Juyong Zhang. 2023. FlashAvatar: High-Fidelity Digital Avatar Rendering at 300FPS. arXiv:2312.02214 [cs.CV]

Tianhan Xu, Yasuhiro Fujita, and Eiichi Matsumoto. 2022. Surface-aligned neural radiance fields for controllable 3d human synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 15883–15892.

Yuelang Xu, Benwang Chen, Zhe Li, Hongwen Zhang, Lizhen Wang, Zerong Zheng, and Yebin Liu. 2023a. Gaussian Head Avatar: Ultra High-fidelity Head Avatar via Dynamic Gaussians. arXiv:2312.03029 [cs.CV]

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

Yuelang Xu, Hongwen Zhang, Lizhen Wang, Xiaochen Zhao, Huang Han, Qi Guojun, and Yebin Liu. 2023c. LatentAvatar: Learning Latent Expression Code for Expressive Neural Head Avatar. In *ACM SIGGRAPH 2023 Conference Proceedings*.

Haotian Yang, Hao Zhu, Yanru Wang, Mingkai Huang, Qiu Shen, Ruigang Yang, and Xun Cao. 2020. FaceScape: A Large-Scale High Quality 3D Face Dataset and Detailed Riggable 3D Face Prediction. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*. Computer Vision Foundation / IEEE, 598–607. https://doi.org/10.1109/CVPR42600.2020.00068

Tarun Yenamandra, Ayush Tewari, Florian Bernard, Hans-Peter Seidel, Mohamed Elgharib, Daniel Cremers, and Christian Theobalt. 2021. i3DMM: Deep Implicit 3D Morphable Model of Human Heads. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*. Computer Vision Foundation / IEEE, 12803–12813. https://doi.org/10.1109/CVPR46437.2021.01261

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. 2018. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 586–595.

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

Yufeng Zheng, Wang Yifan, Gordon Wetzstein, Michael J Black, and Otmar Hilliges. 2023. Pointavatar: Deformable point-based head avatars from videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 21057–21067.

Wojciech Zielonka, Timo Bolkart, and Justus Thies. 2022. Towards Metrical Reconstruction of Human Faces. In *European Conference on Computer Vision*. https://api.semanticscholar.org/CorpusID:248177832

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

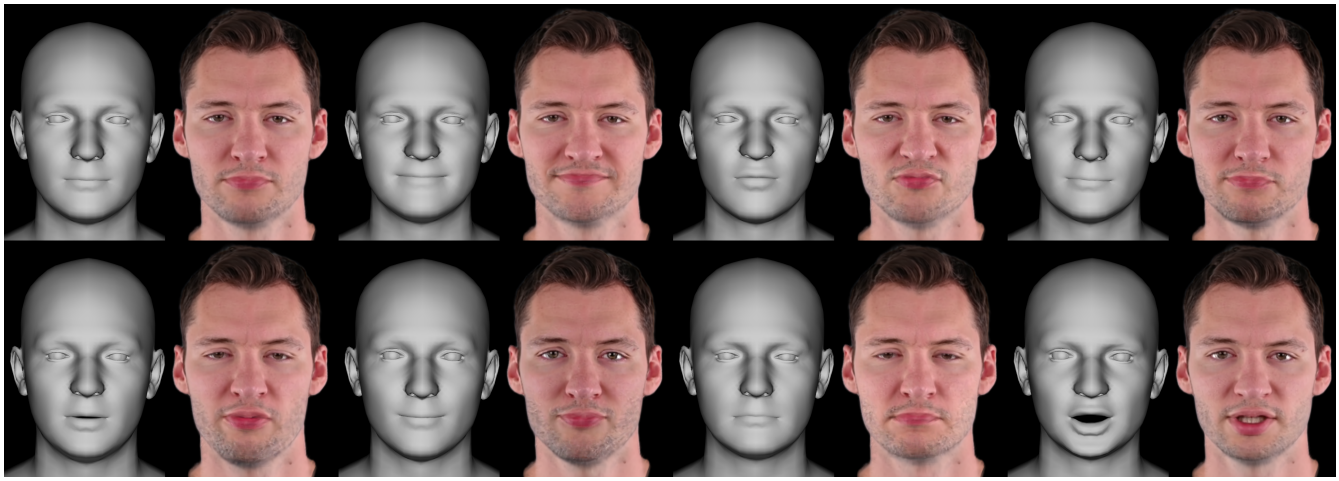


Figure 12: Visualization of our Gaussian blendshapes. Each Gaussian blendshape resembles its corresponding FLAME mesh blendshape, and captures photo-realistic appearance.

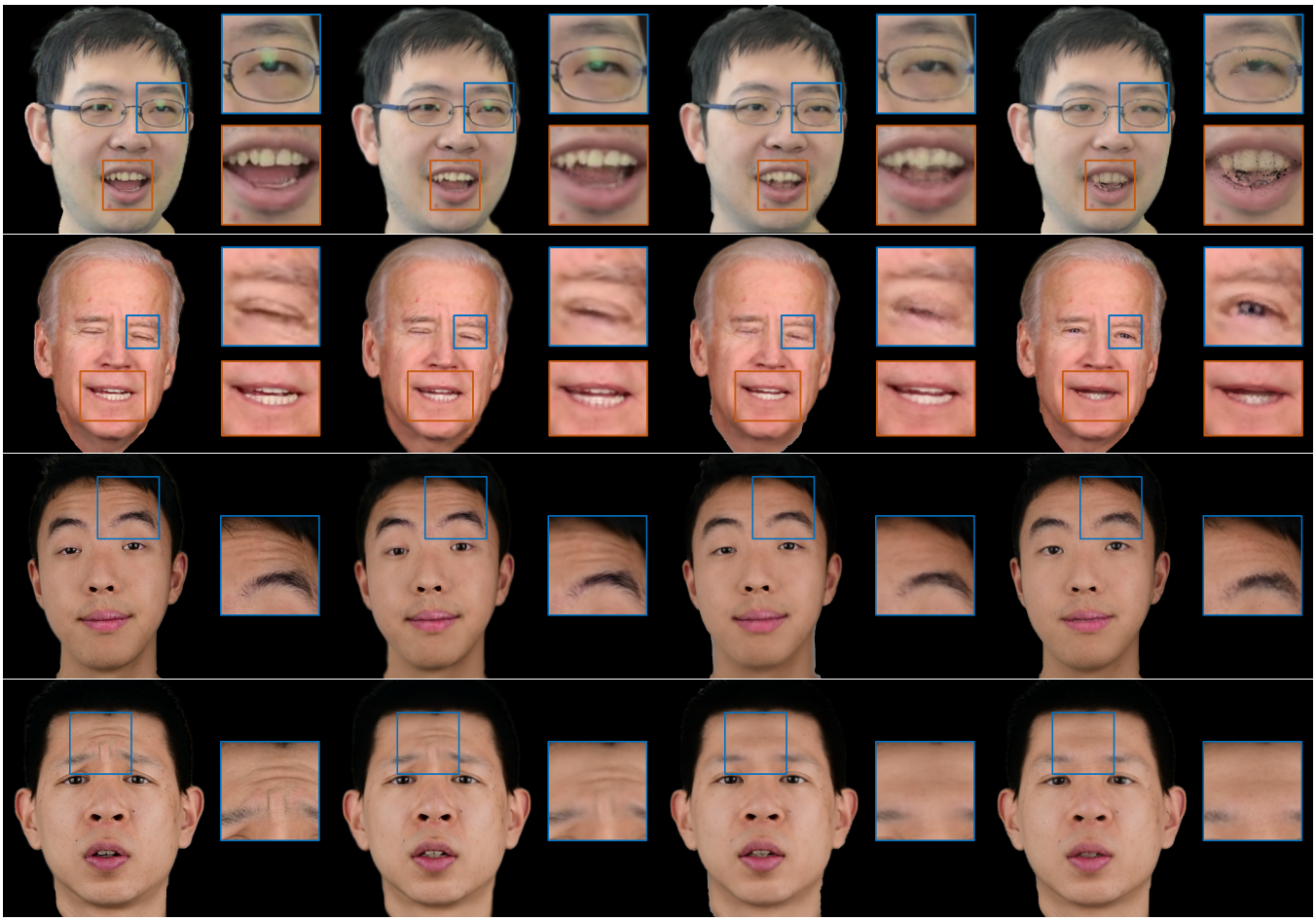


Figure 13: Qualitative comparisons between INSTA [Zielonka et al. 2023], PointAvatar [Zheng et al. 2023], and our method. Our method better captures high-frequency details and specular highlights. YouTube video ID is smghyezLW5o for Joe Biden.