LegacyAvatars: Volumetric Face Avatars For Traditional Graphics Pipelines

Safa C. Medin[1,2] Gengyan Li[1,3] Ziqian Bai[1] Ruofei Du[1] Leonhard Helminger[1]
Yinda Zhang[1] Stephan J. Garbin[1] Philip L. Davidson[1] Gregory W. Wornell[2]
Thabo Beeler[1] Abhimitra Meka[1]

[1]Google [2]MIT [3]ETH Zurich

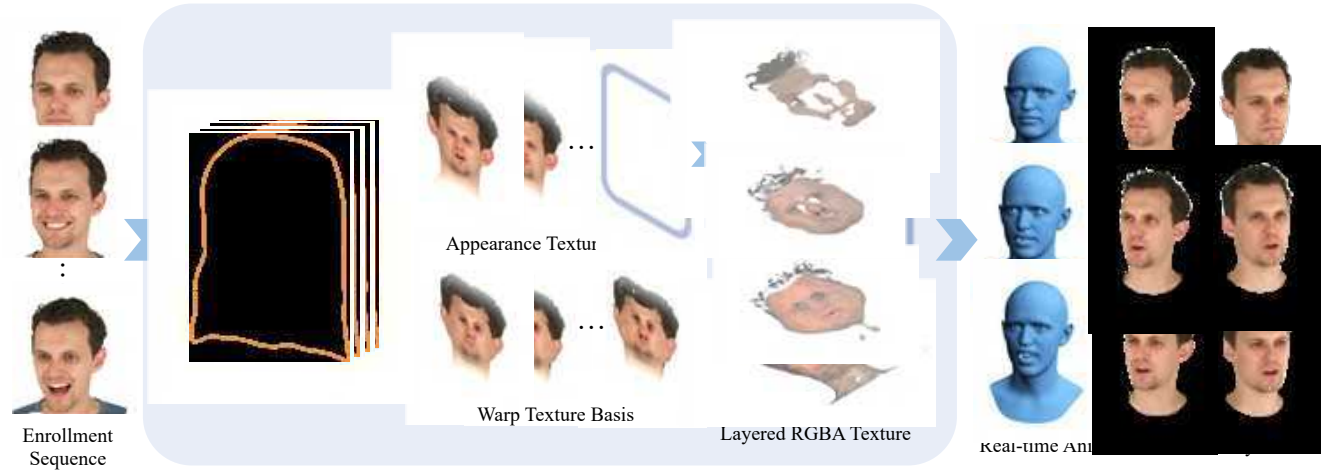

Figure 1. We present a novel representation for rendering animatable volumetric 3D face avatars using meshes and textures. From an enrollment sequence of a subject, we learn a layered mesh and blend-textures that model the geometry, appearance and deformations, and a simple linear transformation that maps tracked face model parameters to blend weights. Our representation is readily compatible with existing streaming infrastructure and can be deployed in traditional graphics pipelines in a device- and platform-agnostic way.

Abstract

We introduce a novel representation for efficient classical rendering of photorealistic 3D face avatars. Leveraging recent advances in radiance fields anchored to parametric face models, our approach achieves controllable volumetric rendering of complex facial features, including hair, skin, and eyes. At enrollment time, we learn a set of radiance manifolds in 3D space to extract an explicit layered mesh, along with appearance and warp textures. During deployment, this allows us to control and animate the face through simple linear blending and alpha compositing of textures over a static mesh. This explicit representation also enables the generated avatar to be efficiently streamed online and then rendered using classical mesh and shader-based rendering on legacy graphics platforms, eliminating the need for any custom engineering or integration.

https://syntec-research.github.io/LegacyAvatars/

1. Introduction

Practical realization of real-time 3D face avatars [7, 37] demands progress on several fronts—capture (or enrollment), streaming, animation, and 3D rendering (or view synthesis). Over the last decade, modeling and rendering of these avatars have seen significant progress by enhancing surface-based geometry representation of parametric face models [12, 29] with volumetric components such as point clouds [62], neural radiance fields [1, 2, 18, 64], or 3D Gaussians [8, 32, 41, 58]. Such volumetric techniques have improved the overall visual quality of the avatars and enabled data-driven modeling of the rigid and non-rigid dynamics of the human face [17, 26, 28, 51].

Current volumetric avatars still face several practical drawbacks that limit their large-scale, real-world applications. *First*, these representations are not natively compatible with the widely available rendering infrastructure. Most avatar applications on real devices are built using

legacy platforms such as game engines (like Unity or Unreal) and design and rendering software (like Blender or Maya), which have been optimized over decades to efficiently use meshes and textures. *Second*, they are not immediately compatible with the existing streaming infrastructure for real-time applications, unlike 2D image and video streaming, which are widely used across devices and applications due to dedicated hardware, driver software, and compressed data formats. Modern volumetric representations typically parameterize the scene density and radiance with a neural network [34], hash grids [35], or 3D Gaussians [22], which have been demonstrated with custom-built proprietary back-end infrastructure, but cannot easily operate across graphics platforms, limiting their practical adoption [9, 50]. Recently, game engine plugins have emerged that support static and dynamic volumetric scenes using 3D and 4D Gaussians [31, 40], but streaming *animatable* or *controllable* volumetric content like face avatars into such legacy graphics platforms remains an unsolved problem.

Motivated by these limitations, we develop a representation that allows *animatable* volumetric rendering of face avatars using only legacy primitives like meshes and textures without relying on ML inference, facilitating native compatibility with standard graphics platforms. To achieve this, our key insight is to discretize and quantize all continuous scene components of a face avatar—*geometry*, *appearance*, and *deformation*—into classical primitives and modulate them using an animation control signal. We draw inspiration from *radiance manifolds (RM)* [10] that represent the volumetric radiance domain using a set of continuous non-intersecting surfaces. Previous works have leveraged RM for 3D playback of pre-recorded dynamic sequences [33] by discretizing the surfaces and exporting them as a layered mesh and transparent RGBA texture that can be rendered through the standard graphics pipeline with additional alpha compositing. A similar paradigm was used for creating animatable avatars [11] by learning deformable RM using a 3D morphable face model, which relies on neural networks to generate the animated appearance, leading to a complex rendering pipeline that disallows native compatibility and easy interoperability between graphics platforms. While these methods utilize RM for discretizing scene geometry and appearance, they do not provide any means to discretize or explicitly control the scene *deformation*.

We build on these ideas by tackling the problem of discretizing and controlling the *deformation* and *animation*. We uniquely show that animation control does not require deforming the layered mesh vertices, but simply offsetting the UV coordinates that are used for sampling the RGB and alpha textures. We also show that these UV-offsets for a given expression can be modeled using a simple linear transform of the tracked face model parameters. Our resulting assets consist of a single layered mesh and a basis of appearance and UV-warp texture maps that need to be streamed *only once* at the beginning, using standard mesh and image compression. We achieve temporal animation of the avatar by streaming only the tracked face model parameters ($\sim$2KBs per frame), which are then linearly transformed to linear-blend coefficients for the texture bases to achieve the final layered appearance texture. Our assets can be rendered on any device using a single pass of a simple programmable shader for linear compositing followed by standard rasterization. In summary,

- We present a novel representation for volumetric 3D face avatars that models geometry, appearance, *and* deformation of the scene using only legacy graphics primitives of a layered triangle mesh and a set of textures, *without* relying on ML inference for rendering. This allows native compatibility with widely available graphics platforms and online streaming infrastructure.
- Through qualitative and quantitative comparisons on a publicly available dataset [57], we demonstrate that our representation achieves efficient rendering on a traditional graphics platform like WebGL on a consumer laptop, while maintaining volumetric visual quality.

2. Related Work

Early methods. The first real-time 3D avatar systems were realized using differentiable 3D morphable face models (3DMMs) [5, 12], enabling efficient optimization frameworks for canonical performance capture and playback. But these models typically suffer from insufficient representational capacity since they are inherently low-dimensional and cannot model complex, high-frequency effects in geometry and appearance, limiting the output visual quality.

Volumetric methods. Complex geometry and appearance of faces, including hair and eyes, have recently been modeled using volumetric methods. Park et al. [38, 39] show that radiance fields can be combined with a learned deformation field to reconstruct a high-quality 3D canonical face model, including minute details such as hair strands. Such ideas have been extended to real-time avatars by enhancing 3DMMs with volumetric radiance fields [1, 3, 7, 14, 18, 27, 30]. More recently, point clouds [62] and 3D Gaussians [8, 28, 32, 41, 46, 53, 58] have been used to achieve best-in-class real-time avatars using similar strategies, but they suffer from other challenges that limit their applications. For example, Gaussian splatting (GS) implementations are typically fine-tuned for specific compute architectures like GPUs, hence cannot be easily deployed in a hardware-agnostic setting. GS also suffers from the famous issue of *popping* phenomenon due to the primitive ordering during view-dependent rendering, causing distracting artifacts in the scene. While there exist methods that introduce sorting-free GS [19, 42], they are tailored for general,

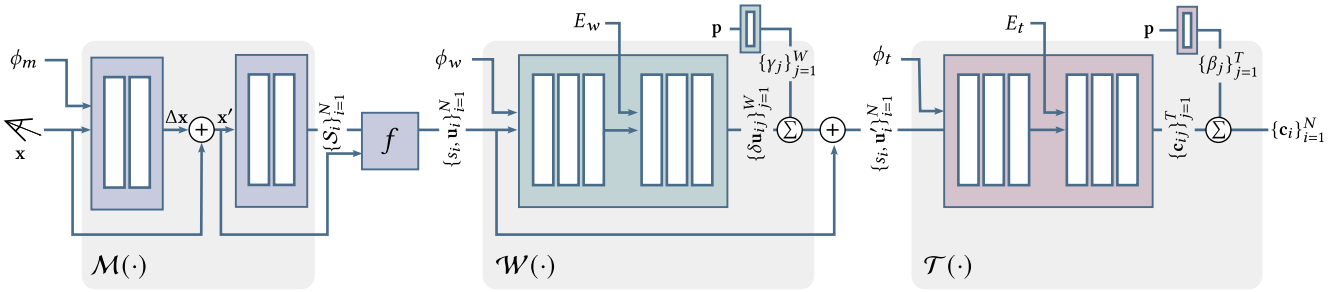

Figure 2. **Training pipeline for enrollment phase.** Our model consists of three separate modules: a manifold predictor $\mathcal{M}$, a warp predictor $\mathcal{W}$, and a texture predictor $\mathcal{T}$. Here, $\mathcal{M}$ is a scalar field that defines layered implicit surfaces. The intersections with these surfaces are spherically mapped to the UV-space via a learnable function f. Then, the output subsequently queries $\mathcal{W}$ to obtain a basis of UV-offsets. These offsets are then linearly blended as a function of expression parameters of a face model and added to the original values. Finally, the new coordinates are fed through $\mathcal{T}$, which predicts a basis of RGBA appearances that are also linearly blended as a function of expression parameters. Each module takes in learned latent codes ϕ_m, ϕ_w, ϕ_t for multi-subject training, while $\mathcal{W}$ and $\mathcal{T}$ take in learnable embedding matrices E_w and E_t to output bases of warps and textures.

static scenes. Finally, Gaussian primitives are not natively compatible with the existing streaming infrastructure, hence streaming them requires specialized engineering [49].

Several avatar representations have focused on efficient engineering by discretizing the 3D volume to achieve real-time results. Tri-plane representations learn features in orthogonal planes and project them to 3D [4, 44, 45, 48]. Hash grids [64] and hash ensembles [3, 25] voxelize the 3D space while promoting quick spatial queries, leading to real-time rendering of radiance fields. Tetrahedral fields have been used to directly deform a volumetric representation to realize animatable avatars [16, 20, 60]. But similar to Gaussian-based approaches, these methods also require custom engineering for deployment to real-world scenarios.

Another class of methods extends classical mesh-based blendshapes to volumetric representations by introducing blendable bases using neural fields [15] or Gaussian primitives [32]. Although our method follows a similar principle—as we also combine a fixed set of assets to synthesize novel expressions—these methods rely on explicit 3D deformations to represent changes in the face geometry. Moreover, like their conventional neural field or 3D Gaussian-based counterparts, they are not immediately suitable for practical telepresence systems.

Implicit surface-based methods. We draw inspiration from a class of methods that learn a collection of implicit surfaces [10, 13] to discretize the 3D volume [11, 33, 56, 59]. These methods were first used in a GAN-based pipeline to generate 3D renderings of novel human identities [10] and then extended to control their generated pose or expression [56, 59]. FaceFolds [33] uses radiance manifolds to model pre-recorded videos of dynamic face performances of real people and exports them as a static layered mesh textured with an RGBA video, demonstrating compatibility with traditional 3D rendering and streaming systems. But it does not explicitly discretize or control face deformations and merely captures them in appearance textures for playback, providing no means to control or animate the faces. We build on these ideas to achieve animation by 1) introducing a warping mechanism to represent most geometric changes in UV-space deformations instead of offloading all variations to alpha composition of appearance textures, 2) representing warp and appearance as linear combinations of a fixed basis of assets to achieve ML-free rendering, and 3) leveraging synthetic data to enable generalization in the low-dimensional expression parameter space without realizing the full 3DMM mesh.

An avatar system most similar to our representation is BakedAvatar [11], which similarly extracts layered mesh and textures from learned implicit surfaces. But differently, they perform face animation by explicitly deforming their layered meshes using per-vertex FLAME deformation weights [29]. In the fragment shader, they rely on an MLP to dynamically compute blend weights, which linearly combine multiple pre-baked textures to produce the final pixel color. Since these non-linear deformations and neural network inference demands custom engineering during deployment, their representation is not natively compatible with legacy renderers. In addition, their per-pixel MLP queries also make scaling to higher resolutions increasingly more expensive. Whereas in our method, we 1) model deformations discretely and linearly in the UV space of a static mesh without having to deform it, and 2) do not rely on any neural networks at deployment time, which helps us achieve real-time performance at high resolutions on consumer devices, as well as backward compatibility and streamability trivially. We perform a thorough evaluation against BakedAvatar [11] to adequately demonstrate, both qualitatively and quantitatively, the improvements our

method offers—in terms of both visual quality *and* compute efficiency and while, most importantly, being readily compatible with legacy graphics pipelines.

3. Methodology

A 3D avatar system generally consists of two phases, an *enrollment* phase in which the avatar is created using data captured from a subject, and a *deployment* phase in which the avatar is streamed, animated, and rendered on the client device from a desired viewpoint. Our goal is to design a volumetric avatar representation that is compatible with legacy rendering platforms during the deployment phase. We achieve this by exporting our enrolled volumetric avatar to classical graphics primitives like meshes and textures that can be rendered efficiently using simple programmable shaders on any graphics platform without additional custom engineering, agnostic of the underlying device hardware or software. We also aim to make our representation conducive to online streaming, which includes the ability to trade off quality and data bandwidth via data compression, similar to today's online video streaming systems.

3.1. Avatar as Layered Mesh

Given the calibration video of a subject, our objective is to learn a single layered mesh representation of the subject that can be dynamically textured using expression coefficients of a parametric face model in real-time. Similar to previous methods, we use radiance manifolds [10] to model the geometry as a set of static 2D implicit surfaces [33, 59], and the appearance as a UV-mapped dynamic radiance controlled by the 3DMM coefficients [11].

Our geometry is modeled by learning a set of N implicit surfaces $\{\mathcal{S}_i\}_{i=1}^{N}$ defined by a single manifold predictor $\mathcal{M} : \mathbb{R}^3 \rightarrow \mathbb{R}$, which takes in a 3D point $\mathbf{x} \in \mathbb{R}^3$ and outputs a scalar value $s \in \mathbb{R}$. We first map all points on these surfaces via a *learnable* function $f : \mathcal{S}_i \rightarrow [-1, 1]^2$ to obtain UV-space coordinates. In the UV-space, we learn a set of W warp fields and T RGBA texture fields for each surface by parameterizing the following functions as MLPs:

$$\mathcal{W}_{ij} : \mathbf{u} \mapsto \delta\mathbf{u} \tag{1}$$

$$\mathcal{T}_{ik} : \mathbf{u} \mapsto \mathbf{c}\,, \tag{2}$$

where $\mathbf{u} \in [-1, 1]^2$ is a UV-space coordinate, $\delta\mathbf{u} \in \mathbb{R}^2$ is a UV-space offset, $\mathbf{c} \in \mathbb{R}^d$ includes a scalar alpha value and view-dependent color parameterized using spherical harmonics [43], and $i \in \{1, 2, \ldots, N\}$, $j \in \{1, 2, \ldots, W\}$, $k \in \{1, 2, \ldots, T\}$. Given a set of face model parameters $\mathbf{p} \in \mathbb{R}^p$ of a single frame, a layered warp field $\{\mathcal{W}_i\}_{i=1}^{N}$ and a layered texture field $\{\mathcal{T}_i\}_{i=1}^{N}$ are obtained by

$$\mathcal{W}_i = \sum_{j=1}^{W} \gamma_j \mathcal{W}_{ij} \quad \text{and} \quad \mathcal{T}_i = \sum_{k=1}^{T} \beta_k \mathcal{T}_{ik} \tag{3}$$

where $\{\gamma_j\}_{j=1}^{W}$ and $\{\beta_k\}_{k=1}^{T}$ are a set of weights obtained as a learnable *linear* function of $\mathbf{p}$.

Given a 3D point $\mathbf{x} \in \mathcal{S}_i$ and its UV-coordinate $\mathbf{u}$, we compute the warped UV-values $\mathbf{u}' = \mathbf{u} + \mathcal{W}_i(\mathbf{u})$, which are used to query the blended texture field to obtain color and transparency as $\mathbf{c} = \mathcal{T}_i(\mathbf{u}')$. We design our pipeline in a way that the implicit surfaces and the warp and texture bases can be efficiently discretized and exported into a layered mesh and UV-space maps in pixel-space, allowing them to be immediately deployed to any graphics platform without relying on custom rendering algorithms or ML tools.

3.2. Dataset

To train our model, we use multiview videos from the NeRSemble dataset [25], which consists of a set of subjects with various facial expressions and talking sequences. We fit a parametric face model to the video sequences of each subject to obtain per-frame expression coefficients as well as per-pixel UV-values to explicitly supervise correspondences between different frames. As a preprocessing step, we transform the camera extrinsics to align faces across frames to a canonical 3D face in order to account for strong head rotations during the capture. We downsample the original images to 802×550 resolution.

Expression generalization. Learning an expressive geometry and appearance model that also generalizes to novel expressions outside of the calibration sequence is a challenging and ill-posed problem. Furthermore, sampling across 2D discrete surfaces instead of the entire 3D volume introduces more instability in training, which can cause shelling artifacts in extreme poses [33]. To aid generalization to novel expressions and to mitigate stability issues arising in joint learning of geometry and appearance, we introduce a multi-subject training paradigm. Uniquely, we make use of a publicly available synthetic multi-view multi-expression image dataset [6]. In our experiments, we combine each subject in the NeRSemble dataset [25] with a number of synthetic subjects that have similar face shapes to the real subject. We show that such joint training helps avoid overfitting to the expressions from the calibration sequence of the target subject and prevents artifacts. Please see the supplementary results for more details.

3.3. Model Architecture and Training

Our architecture consists of three modules: a manifold predictor $\mathcal{M}$, a UV-space warp predictor $\mathcal{W}$, and a texture predictor $\mathcal{T}$, which we illustrate in Fig. 2.

Manifold predictor. We implement our manifold predictor as a *subject-specific* scalar field that takes in 3D coordinates in xyz-space and a learned latent code ϕ_m for each subject. This module defines a set of N implicit surfaces, which are *static* for the entire sequence of a given subject. To achieve

this, an input 3D point is first deformed using a subject-specific deformation field, and subsequently used to predict a scalar value that determines the manifold geometry. Given a camera ray, we first find xyz-space intersections of this ray with each manifold [36] and compute an initial set of UV-coordinates using spherical mapping [33]. But since we are interested in learning dense UV-space correspondences across frames consistent with the ground truth UV values, this spherical transformation is done with respect to a scene center set as a 3D learnable parameter instead of a fixed one.

UV-space warp predictor. Uniquely in our animation model, we learn a basis of UV-space offsets for each of the surfaces implemented as an MLP, which receives a learned subject code ϕ_w and a set of W learned latent codes represented as an embedding matrix E_w. This defines a set of warp fields that can be linearly combined into a single warp field, which is used to add offsets to the input UV-coordinates before querying the appearance model. In particular, we first *linearly* map the per-frame expression coefficients to a set of weights that *linearly* blend the predicted UV warps across different surfaces. We note that the blending weights are the same for all surfaces, facilitating a lightweight compute at rendering time.

UV-space texture predictor. To account for the geometry and appearance changes that cannot be fully modeled by the UV-offsets (such as the inner mouth, eyelid motions, or complex specularities on the skin), we learn a set of UV-space explicit blend-textures that can also be linearly combined into a single layered texture using the 3DMM expression parameters. Our texture predictor receives a learned subject code ϕ_t and a set of T learned latent codes represented as an embedding matrix E_t, and it outputs 2-degree spherical harmonics coefficients for radiance [43] and a scalar alpha value. Finally, given a UV-space coordinate $\mathbf{u}'$ and a view direction in xyz-space, the output of our entire pipeline is an RGB radiance and an alpha value, which are composited across rays to produce the final color:

$$\mathbf{w}_i \triangleq \alpha_i \prod_{j=1}^{i-1}(1-\alpha_j) \quad \mathbf{c} = \sum_{i=1}^{N} \mathbf{w}_i \mathbf{c_i} \tag{4}$$

At inference time, we decompose the radiance into diffuse and specular components, which provides further flexibility on the size of the assets that are exported from the model. In Fig. 3, we illustrate our renders for each component.

Loss functions. We train our pipeline in an end-to-end fashion by adopting the following loss function:

$$\mathcal{L} = \mathcal{L}_{\text{rec}} + \lambda_{\text{uv}}\mathcal{L}_{\text{uv}} + \lambda_{\text{silh}}\mathcal{L}_{\text{silh}} + \lambda_{\text{reg}}\mathcal{L}_{\text{reg}} \tag{5}$$

where $\mathcal{L}_{\text{rec}}$ is an ℓ_1 reconstruction loss between predicted and ground truth pixel values, $\mathcal{L}_{\text{uv}}$ is an ℓ_1 loss between the

Figure 3. **Radiance decomposition.** During training, the radiance is modeled using spherical harmonics coefficients, which can be decomposed into diffuse (view-independent) and specular (view-dependent) components. The appearance can be exported as just diffuse or both diffuse and specular texture images.

Figure 4. **UV-space predictions.** Given a ground truth per-pixel UV-coordinates $\mathbf{u}_{\text{gt}}$, our model is supervised to match the *expectation* of warped coordinates $\bar{\mathbf{u}}'$ to the ground truth. We visualize the expectations of the spherically mapped coordinates $\bar{\mathbf{u}}$ and the warps $\delta\bar{\mathbf{u}}$ for reference.

per-pixel *expected* UV-coordinates and ground truth UV-values computed only on skin regions of the face, $\mathcal{L}_{\text{silh}}$ is a silhouette loss that guides the geometry of each layer using per-image foreground masks [11], and $\mathcal{L}_{\text{reg}}$ is the regularization term that penalizes predicted warps, specular radiances, and manifold predictor weights to promote training stability [33]. Here, the expected UV for each pixel is obtained by a weighted combination of the predicted warped UVs across rays $\{\mathbf{u}'_i\}_{i=1}^N$ as $\bar{\mathbf{u}}' \triangleq \sum_{i=1}^N \mathbf{w}_i\mathbf{u}'_i$, which is supervised to match the ground truth UV-values at that pixel. The UV supervision is a key term to improve the model capacity by registering the facial features consistent with the UV topology of the 3DMM, so that the texture basis focuses on appearance changes that cannot be explained by UV-space warps. We illustrate a representative test frame along with its per-pixel UV values and warps in Fig. 4.

Training details. The manifold predictor is implemented as two cascaded 4-layer MLPs of widths 128, which respectively learn a subject-specific 3D warp field and a scalar field that determines the manifold geometry. The warp predictor is a 6-layer MLP of widths 128, which takes in W learnable embeddings of dimension 128 after the third layer. The texture predictor is a 6-layer MLP of widths 256, which receives T learnable embeddings of dimension 128 after the third layer. The subject embeddings are 128-dimensional vectors, which condition each module individually. We optimize our entire model using the Adam optimizer [24] with

initial learning rates of 0.0007, 0.0005, 0.0008 and exponential decay rates of 0.20 per 200 000 iterations for $\mathcal{M}$, $\mathcal{W}$, and $\mathcal{T}$, respectively. With a batch size of 32 768 rays sampled across all subjects, frames, and views, we train our pipeline for 500 000 iterations, which takes approximately 36 hours over 8 NVIDIA H100 GPUs.

3.4. Exporting 3D Assets and Model Deployment

Similar to previous works [33], we discretize our continuous manifolds by shooting rays from a hemisphere (centered at the learned scene center) uniformly in azimuth and elevation angles, gathering all intersections and topologizing them into a triangle mesh. We use the same set of points to query our warp and texture predictors to obtain a basis of UV-space offsets and appearance maps. Uniquely, we export the linear functions that map expression coefficients to blend weights as individual matrices, allowing us to control animation without deforming the mesh but through simple blending of our warp textures. Depending on the application, the mesh can be decimated to reduce the number of primitives, while UV-offset and appearance maps can be downsampled to lower resolutions. Our final assets merely comprise a single layered mesh with a fixed topology as well as warp and appearance maps, and they can easily be deployed to any graphics platform for rendering.

3.5. Rendering

The linear blending and alpha compositing is performed by a programmable shader that receives the exported 3D assets, as well as face model parameters $\mathbf{p}$ and the camera viewpoint, as shown in Fig. 5. Here, blend and warp phases are simple linear operations, while rasterization is a projection operation handled by the standard graphics pipeline. By learning a canonical geometry and modeling deformations in the 2D UV-space instead of the 3D space, we can disentangle their parameterization into a single mesh and a set of warp maps. In our results, we show that it is possible to model the blend weights for the warp and appearance bases as a linear transformation of the expression parameters. This makes animation a very simple linear operation without having to explicitly account for complex non-rigid dynamics in 3D. This is in contrast to mesh-based avatar methods such as BakedAvatar [11], which handles animations by explicitly deforming meshes.

A note on rendering efficiency in comparison with modern methods. Rasterizing triangles is inherently significantly faster than the standard implementations of volumetric rendering techniques such as Gaussian splatting and neural fields on a per-primitive and per-pixel basis. Gaussian splatting requires an expensive sorting operation, while neural fields rely on tracing rays, sampling multiple points along the ray that need to be contiguously integrated. Rasterizing our ordered mesh representation does not suffer from such challenges since sorting is handled using the depth buffer. Finally, we *do* note that there are several implementations of 3D Gaussian splatting and neural fields, including hash grids like InstantNGP [35] and hierarchical embeddings [23] that offer faster results, but they are often engineered for particular hardware such as a GPU or require custom implementations for wide-scale deployment.

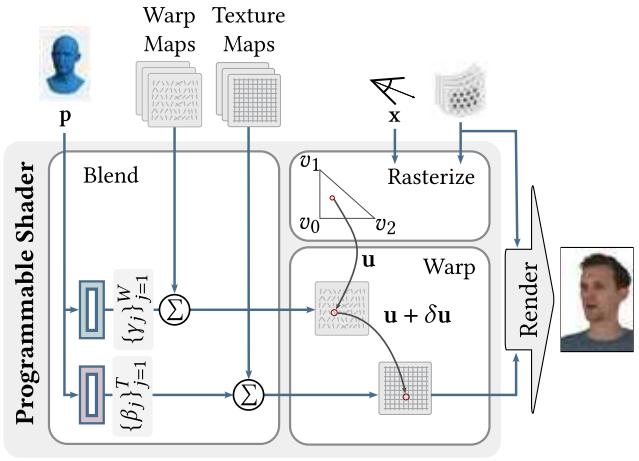

Figure 5. **Programmable shader.** At the deployment phase, our 3D assets (a single static layered mesh and bases of warp and texture maps) can easily be used to render dynamic and volumetric faces via a programmable shader on any graphics platform.

3.6. Streaming

Our method simplifies data transmission by initially sending the static mesh, blend textures, and linear transformations *only once*. Subsequently, only per-frame face model parameters $\mathbf{p}$ are streamed, which are fed directly to the programmable shader in Fig. 5. Our technique also offers a unique advantage in *multi-avatar interaction* scenarios. In a 1-on-1 interaction, complete client-side rendering is ideal, as it minimizes the amount of data that must be transmitted. But in a multi-avatar scenario, offloading compute to the server side is more efficient as it avoids duplication of compute effort across multiple clients. Our pipeline allows for expression-related computations such as warping and blending to be performed on the server side, so that only the view-dependent rendering is left to the client side. Most importantly, since the output of our warping and blending operations is a set of texture maps, they can be conveniently transmitted as a compressed video stream. This allows for a healthy trade-off between compute and transmission bandwidth. Such a trade-off is not trivially available with other volumetric techniques, including other layed-mesh based techniques like BakedAvatar [11].

4. Experiments and Results

In our experiments, we set $N = T = W = 12$. Following FaceFolds [33], we select a number of manifolds that al-

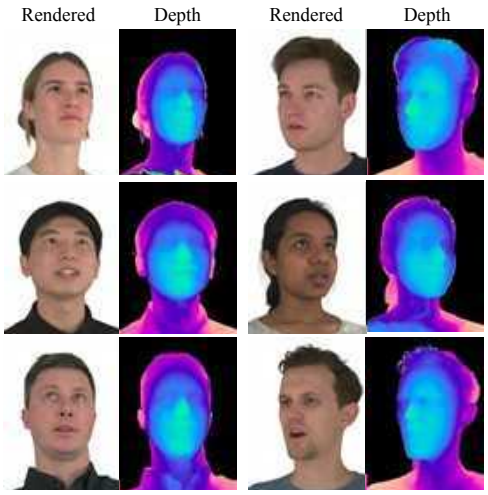

Figure 6. **Novel view synthesis results.** Our model achieves photorealistic volumetric rendering of 3D face avatars. Please see the supplementary material for video demonstrations.

lows for volumetric effects while maintaining rendering efficiency. Similarly, basis sizes are tuned to balance computational efficiency with the model's expressivity. We use the first 9 talking sequences of each subject from the NeRSemble dataset [25] for training and use the last one for testing. We hold out 2 of the 16 cameras in training data to evaluate our model's performance quantitatively. To aid the training stability and generalization, we gather 25 subjects from the synthetic data corpus with the closest head shapes to the real subjects, as discussed in Sec. 3.2. We measure this proximity using a variance-weighted Euclidean distance between the PCA identity vectors.

We compare against five state-of-the-art efficient volumetric avatar techniques, BakedAvatar [11], Gaussian Head Avatar (GHA) [58], GaussianAvatars [41], MonoAvatar++ [3], and PointAvatar [62]. We chose these methods as representative techniques that achieve fast rendering and employ layered meshes, Gaussian splatting, or neural radiance fields as the underlying volumetric representation. **Note:** The goal of our method is to achieve volumetric effects using legacy primitives and no ML inference. We do *not* claim that we outperform continuous volumetric avatar techniques that are based on Gaussians or NeRFs on overall visual quality. We only show comparable results with a sample of such methods in order to visually place our technique in the overall context of the state-of-the-art.

4.1. Qualitative Results

Novel view synthesis. We illustrate our novel view synthesis results in Fig. 6. Our method generalizes over different subjects with varying face geometries and appearances. Please refer to the supplementary material for video demonstrations and comparisons with other methods.

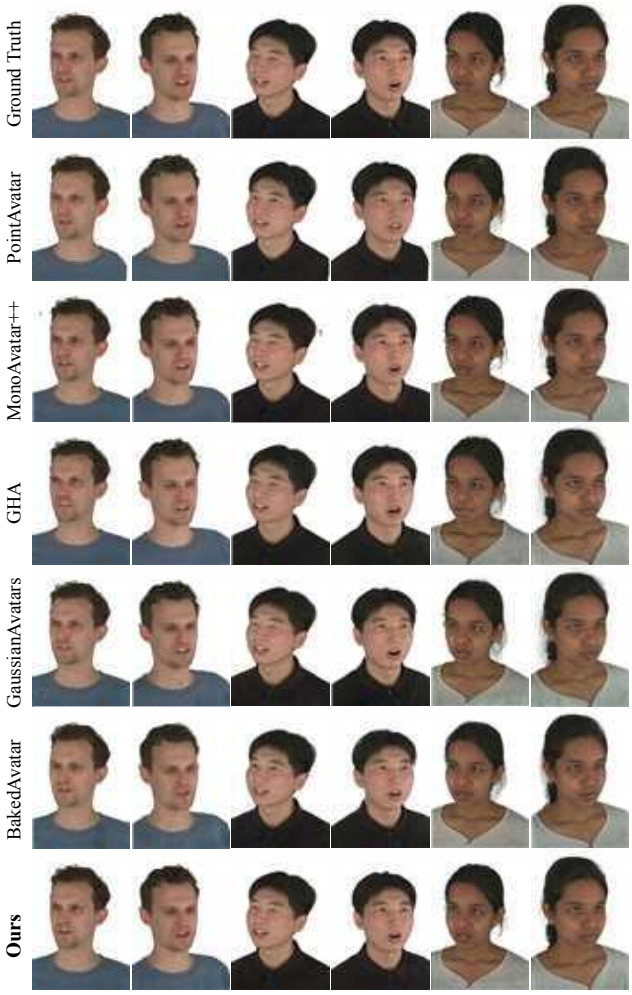

Figure 7. **Qualitative comparisons.** Our technique achieves comparable visual quality to modern neural rendering techniques while facilitating 3D animations in a platform-agnostic way.

Table 1. **Quantitative comparisons.** Our model attains similar performance compared to the state-of-the-art methods.

Method	PSNR ↑	SSIM ↑	LPIPS ↓
PointAvatar	23.80±1.28	0.872±0.016	0.137±0.018
MonoAvatar++	27.45±2.43	0.936±0.011	0.098±0.009
GHA	24.29±2.16	0.863±0.039	0.102±0.024
GaussianAvatars	27.54±1.69	0.931±0.017	0.066±0.015
BakedAvatar	24.38±0.78	0.888±0.013	0.117±0.018
Ours	26.97±1.23	0.929±0.007	0.117±0.006

Self- and cross-reenactment. We show renders of test expressions from the held-out views and compare them with baseline methods in Fig. 7. Our model achieves comparable visual quality to the previous methods that rely on sophisticated primitives or MLP queries at rendering time. Please refer to the supp. video for cross-reenactment results.

4.2. Quantitative Results

For three subjects (with IDs `055`, `264`, and `306`), we perform quantitative evaluations on two held-out views across the entire test sequences, consisting of over 800 images. We report average image quality metrics in PSNR, SSIM [54], and LPIPS [61] for our method and other methods in Tab. 1, where we observe comparable average performance, with differences falling within the margin of variance.

4.3. Ablation Studies

Mesh and texture resolution. Similar to FaceFolds [33], our representation has the flexibility to efficiently trade off image quality with rendering efficiency by reducing the number of primitives of the exported mesh and downsampling the layered textures. Given a layered mesh at 512×512 vertex resolution (per layer) with canonical texture coordinates as vertex attributes, we first gather the vertices from each layer as an oriented point cloud and perform Poisson surface reconstruction [21] to obtain watertight surfaces [33]. Then, we use an off-the-shelf mesh decimation algorithm to reduce the number of vertices in the mesh to a given target. Since our model is trained to represent a variety of expressions with a single set of static surfaces, the exported meshes roughly correspond to the coarse face geometry of the subjects. Therefore, we can reduce the total number of primitives significantly without sacrificing the visual quality, see Fig. 8 and the supplementary video.

If there is any need to decrease the resolution of the streamed avatar (such as reduced data bandwidths), we can dynamically downsample our video textures using the existing infrastructure. We report view synthesis and animation results for a variety of texture resolutions, illustrated in Fig. 8 and the supplementary video. For more ablations, please refer to the supplementary material.

4.4. Real-time Rendering on Web Browsers

Our representation is natively deployable on graphics platforms and enables real-time rendering of volumetric face avatars using a simple programmable shader. Using WebGL on a consumer laptop, we achieve the frame rates shown in Fig. 9 for varying mesh resolutions and rendering resolutions, while keeping the memory usage less than 2 GB at 512^2 mesh resolution and 2K rendering resolution. We emphasize that our approach discretizes all scene components, placing it on the memory-intensive side of the inherent memory–compute trade-off. Nevertheless, by employing a moderate number of layers and basis sizes, our assets remain sufficiently lightweight to maintain a memory footprint compatible with commodity hardware. We also observe that the performance of BakedAvatar [11] suffers significantly at higher rendering resolutions owing to per-pixel MLP queries, while our representation naturally scales well to higher resolutions due to simple texture queries.

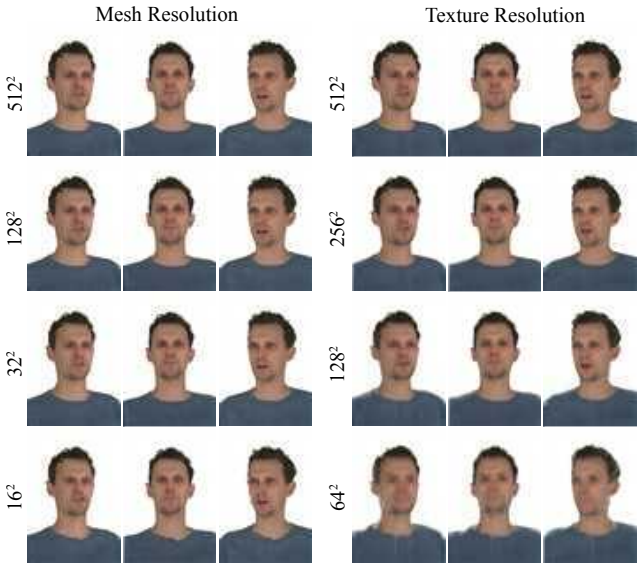

Figure 8. **Ablation on mesh and texture resolution.** At the deployment phase, our asset size can be trivially reduced with standard operations. Due to our smooth surface geometry, the visual quality is maintained down to 32×32 mesh resolution, with a total number of primitives of $<12\,000$, providing a very lightweight volumetric representation for renders at 0.5K resolution. Similarly, the texture resolution can also be adjusted for varying needs of an application by downsampling layered textures.

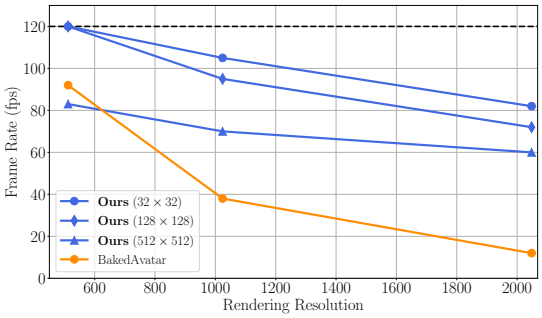

Figure 9. **Frame rates on WebGL.** Our assets can be deployed to traditional graphics engines on web browsers using WebGL, achieving cross-platform compatibility. These numbers are profiled on Chrome 133.0 on a MacBook Pro with M1 Pro chip. Frame rates above the refresh rate of 120 Hz are indicated as 120.

5. Conclusion

We present an efficient and natively-deployable representation for animatable volumetric face avatars for traditional graphics pipelines. We achieve this by modeling the canonical face geometry with a static layered mesh, and appearance and deformation as textures. Our model enables efficient control of facial expressions through simple linear blending of our texture assets based on the tracked face parameters. Thorough experimentation and analysis against modern techniques demonstrate the efficacy of our method.

References

[1] ShahRukh Athar, Zexiang Xu, Kalyan Sunkavalli, Eli Shechtman, and Zhixin Shu. RigNeRF: Fully Controllable Neural 3D Portraits. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20364–20373, 2022. 1, 2

[2] Ziqian Bai, Feitong Tan, Zeng Huang, Kripasindhu Sarkar, Danhang Tang, Di Qiu, Abhimitra Meka, Ruofei Du, Mingsong Dou, Sergio Orts-Escolano, et al. Learning Personalized High Quality Volumetric Head Avatars from Monocular RGB Videos. *arXiv preprint arXiv:2304.01436*, 2023. 1

[3] Ziqian Bai, Feitong Tan, Sean Fanello, Rohit Pandey, Mingsong Dou, Shichen Liu, Ping Tan, and Yinda Zhang. Efficient 3D Implicit Head Avatar with Mesh-anchored Hash Table Blendshapes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1975–1984, 2024. 2, 3, 7, 13

[4] Alexander Bergman, Petr Kellnhofer, Wang Yifan, Eric Chan, David Lindell, and Gordon Wetzstein. Generative Neural Articulated Radiance Fields. *Advances in Neural Information Processing Systems*, 35:19900–19916, 2022. 3

[5] Volker Blanz and Thomas Vetter. A Morphable Model for the Synthesis of 3D Faces. In *SIGGRAPH*, 1999. 2

[6] Marcel C. Buehler, Gengyan Li, Erroll Wood, Leonhard Helminger, Xu Chen, Tanmay Shah, Daoye Wang, Stephan Garbin, Sergio Orts-Escolano, Otmar Hilliges, Dmitry Lagun, Jérémy Riviere, Paulo Gotardo, Thabo Beeler, Abhimitra Meka, and Kripasindhu Sarkar. Cafca: High-quality Novel View Synthesis of Expressive Faces from Casual Few-shot Captures. In *ACM SIGGRAPH Asia 2024 Conference Paper*. ACM, 2024. 4, 12

[7] Chen Cao, Tomas Simon, Jin Kyu Kim, Gabe Schwartz, Michael Zollhoefer, Shun-Suke Saito, Stephen Lombardi, Shih-En Wei, Danielle Belko, Shoou-I Yu, Yaser Sheikh, and Jason Saragih. Authentic Volumetric Avatars From a Phone Scan. *ACM Transactions on Graphics*, 2022. 1, 2

[8] Yufan Chen, Lizhen Wang, Qijing Li, Hongjiang Xiao, Shengping Zhang, Hongxun Yao, and Yebin Liu. MonoGaussianAvatar: Monocular Gaussian Point-based Head Avatar. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–9, 2024. 1, 2

[9] Zhiqin Chen, Thomas Funkhouser, Peter Hedman, and Andrea Tagliasacchi. MobileNeRF: Exploiting the Polygon Rasterization Pipeline for Efficient Neural Field Rendering on Mobile Architectures. In *The Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 2

[10] Yu Deng, Jiaolong Yang, Jianfeng Xiang, and Xin Tong. GRAM: Generative Radiance Manifolds for 3D-Aware Image Generation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 2, 3, 4

[11] Hao-Bin Duan, Miao Wang, Jin-Chuan Shi, Xu-Chuan Chen, and Yan-Pei Cao. BakedAvatar: Baking Neural Fields for Real-Time Head Avatar Synthesis. *ACM Trans. Graph.*, 42 (6), 2023. 2, 3, 4, 5, 6, 7, 8

[12] Bernhard Egger, William AP Smith, Ayush Tewari, Stefanie Wuhrer, Michael Zollhoefer, Thabo Beeler, Florian Bernard, Timo Bolkart, Adam Kortylewski, Sami Romdhani, et al. 3D Morphable Face Models—Past, Present, and Future. *Transactions on Graphics (ToG)*, 39(5):1–38, 2020. 1, 2

[13] Stefano Esposito, Anpei Chen, Christian Reiser, Samuel Rota Bulò, Lorenzo Porzi, Katja Schwarz, Christian Richardt, Michael Zollhoefer, Peter Kontschieder, and Andreas Geiger. Volumetric Surfaces: Representing Fuzzy Geometries with Layered Meshes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025. 3

[14] Guy Gafni, Justus Thies, Michael Zollhöfer, and Matthias Nießner. Dynamic Neural Radiance Fields for Monocular 4D Facial Avatar Reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8649–8658, 2021. 2

[15] Xuan Gao, Chenglai Zhong, Jun Xiang, Yang Hong, Yudong Guo, and Juyong Zhang. Reconstructing Personalized Semantic Facial NeRF Models from Monocular Video. *ACM Transactions on Graphics (TOG)*, 41(6):1–12, 2022. 3

[16] Stephan J Garbin, Marek Kowalski, Virginia Estellers, Stanislaw Szymanowicz, Shideh Rezaeifar, Jingjing Shen, Matthew Johnson, and Julien Valentin. VolTeMorph: Realtime, Controllable and Generalisable Animation of Volumetric Representations. *arXiv preprint arXiv:2208.00949*, 2022. 3

[17] Simon Giebenhain, Tobias Kirschstein, Martin Rünz, Lourdes Agapito, and Matthias Nießner. NPGA: Neural Parametric Gaussian Avatars. In *SIGGRAPH Asia 2024 Conference Papers*, pages 1–11, 2024. 1

[18] Yang Hong, Bo Peng, Haiyao Xiao, Ligang Liu, and Juyong Zhang. HeadNeRF: A Real-Time NeRF-Based Parametric Head Model. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1, 2

[19] Qiqi Hou, Randall Rauwendaal, Zifeng Li, Hoang Le, Farzad Farhadzadeh, Fatih Porikli, Alexei Bourd, and Amir Said. Sort-free Gaussian Splatting via Weighted Sum Rendering. *arXiv preprint arXiv:2410.18931*, 2024. 2

[20] Kacper Kania, Stephan J. Garbin, Andrea Tagliasacchi, Virginia Estellers, Kwang Moo Yi, Julien Valentin, Tomasz Trzciński, and Marek Kowalski. BlendFields: Few-Shot Example-Driven Facial Modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2023. 3

[21] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson Surface Reconstruction. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*, page 0, 2006. 8

[22] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM Transactions on Graphics*, 42(4), 2023. 2

[23] Bernhard Kerbl, Andreas Meuleman, Georgios Kopanas, Michael Wimmer, Alexandre Lanvin, and George Drettakis. A Hierarchical 3D Gaussian Representation for Real-Time Rendering of Very Large Datasets. *ACM Transactions on Graphics (TOG)*, 43(4):1–15, 2024. 6

[24] Diederik P Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *ArXiv*, 2014. 5

[25] Tobias Kirschstein, Shenhan Qian, Simon Giebenhain, Tim Walter, and Matthias Nießner. NeRSemble: Multi-View Radiance Field Reconstruction of Human Heads. *ACM Trans. Graph*, 42(4), 2023. 3, 4, 7

[26] Tobias Kirschstein, Javier Romero, Artem Sevastopolsky, Matthias Nießner, and Shunsuke Saito. Avat3r: Large Animatable Gaussian Reconstruction Model for High-fidelity 3D Head Avatars. *arXiv preprint arXiv:2502.20220*, 2025. 1

[27] G. Li, K. Sarkar, A. Meka, M. Buehler, F. Mueller, P. Gotardo, O. Hilliges, and T. Beeler. ShellNeRF: Learning a Controllable High-resolution Model of the Eye and Periocular Region. *Computer Graphics Forum*, 43(2):e15041, 2024. 2

[28] Gengyan Li, Paulo Gotardo, Timo Bolkart, Stephan Garbin, Kripasindhu Sarkar, Abhimitra Meka, Alexandros Lattas, and Thabo Beeler. TeGA: Texture Space Gaussian Avatars for High-Resolution Dynamic Head Modeling. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*, New York, NY, USA, 2025. Association for Computing Machinery. 1, 2

[29] Tianye Li, Timo Bolkart, Michael J Black, Hao Li, and Javier Romero. Learning a Model of Facial Shape and Expression From 4D Scans. *ACM Trans. Graph*, 2017. 1, 3

[30] Lingjie Liu, Marc Xu, Jiayuan Tan, Yuheng Zhou, Christian Theobalt, and Gerard Pons-Moll. Neural Actor: Neural Free-view Synthesis of Human Actors with Pose Control. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10086–10095, 2021. 2

[31] Luma AI. Luma Unreal Engine Plugin (v0.41). https://lumalabs.ai/ue, 2025. Official docs; Unreal Engine 5 plugin for Neural Radiance Fields and 3D Gaussian Splatting. Accessed 2025-08-15. 2

[32] Shengjie Ma, Yanlin Weng, Tianjia Shao, and Kun Zhou. 3D Gaussian Blendshapes for Head Avatar Animation. In *ACM SIGGRAPH Conference Proceedings, Denver, CO, United States, July 28 - August 1, 2024*, 2024. 1, 2, 3

[33] Safa C. Medin, Gengyan Li, Ruofei Du, Stephan Garbin, Philip Davidson, Gregory W. Wornell, Thabo Beeler, and Abhimitra Meka. FaceFolds: Meshed Radiance Manifolds for Efficient Volumetric Rendering of Dynamic Faces. *Proceedings of the ACM in Computer Graphics and Interactive Techniques*, 2024. 2, 3, 4, 5, 6, 8

[34] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing Scenes As Neural Radiance Fields for View Synthesis. *Communications of the ACM*, 2021. 2

[35] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant Neural Graphics Primitives With a Multiresolution Hash Encoding. *ACM Transactions on Graphics*, 41(102), 2022. 2, 6

[36] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable Volumetric Rendering: Learning Implicit 3D Representations Without 3D Supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3504–3515, 2020. 5

[37] Sergio Orts-Escolano, Christoph Rhemann, Sean Fanello, Wayne Chang, Adarsh Kowdle, Yury Degtyarev, David Kim, Philip Davidson, Sameh Khamis, Mingsong Dou, Vladimir Tankovich, Charles Loop, Qin Cai, Philip Chou, Sarah Mennicken, Julien Valentin, Vivek Pradeep, Shenlong Wang, Sing Bing Kang, Pushmeet Kohli, Yuliya Lutchyn, Cem Keskin, and Shahram Izadi. Holoportation: Virtual 3D Teleportation in Real-Time. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*. ACM, 2016. 1

[38] Keunhong Park, Utkarsh Sinha, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Steven M. Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable Neural Radiance Fields. *ICCV*, 2021. 2

[39] Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M. Seitz. HyperNeRF: A Higher-Dimensional Representation for Topologically Varying Neural Radiance Fields. *ACM Trans. Graph*, 40(6), 2021. 2

[40] Aras Pranckevičius. Unity Gaussian Splatting. https://github.com/aras-p/UnityGaussianSplatting, 2025. GitHub repository. Accessed 2025-08-15. 2

[41] Shenhan Qian, Tobias Kirschstein, Liam Schoneveld, Davide Davoli, Simon Giebenhain, and Matthias Nießner. GaussianAvatars: Photorealistic Head Avatars with Rigged 3D Gaussians. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20299–20309, 2024. 1, 2, 7, 13

[42] Lukas Radl, Michael Steiner, Mathias Parger, Alexander Weinrauch, Bernhard Kerbl, and Markus Steinberger. StopThePop: Sorted Gaussian Splatting for View-Consistent Real-time Rendering. *ACM Transactions on Graphics*, 4 (43), 2024. 2, 13

[43] Ravi Ramamoorthi and Pat Hanrahan. An Efficient Representation for Irradiance Environment Maps. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, pages 497–500, 2001. 4, 5

[44] Pramod Rao, Gereon Fox, Abhimitra Meka, Mallikarjun B R, Fangneng Zhan, Tim Weyrich, Bernd Bickel, Hanspeter Pfister, Wojciech Matusik, Mohamed Elgharib, and Christian Theobalt. Lite2Relight: 3D-aware Single Image Portrait Relighting. In *ACM SIGGRAPH 2024 Conference Papers*, New York, NY, USA, 2024. Association for Computing Machinery. 3

[45] Pramod Rao, Abhimitra Meka, Xilong Zhou, Gereon Fox, Mallikarjun B R, Fangneng Zhan, Tim Weyrich, Bernd Bickel, Hanspeter Pfister, Wojciech Matusik, Thabo Beeler, Mohamed Elgharib, Marc Habermann, and Christian Theobalt. 3DPR: Single Image 3D Portrait Relighting with Generative Priors. In *Proceedings of the SIGGRAPH Asia 2025 Conference Papers*, New York, NY, USA, 2025. Association for Computing Machinery. 3

[46] Shunsuke Saito, Gabriel Schwartz, Tomas Simon, Junxuan Li, and Giljoo Nam. Relightable Gaussian Codec Avatars. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 130–141, 2024. 2

[47] Pratul P Srinivasan, Richard Tucker, Jonathan T Barron, Ravi Ramamoorthi, Ren Ng, and Noah Snavely. Pushing the Boundaries of View Extrapolation with Multiplane Images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 175–184, 2019. 14

[48] Jingxiang Sun, Xuan Wang, Lizhen Wang, Xiaoyu Li, Yong Zhang, Hongwen Zhang, and Yebin Liu. Next3D: Generative Neural Texture Rasterization for 3D-Aware Head Avatars. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 3

[49] Jiakai Sun, Han Jiao, Guangyuan Li, Zhanjie Zhang, Lei Zhao, and Wei Xing. 3DGStream: On-the-Fly Training of 3D Gaussians for Efficient Streaming of Photo-Realistic Free-Viewpoint Videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20675–20685, 2024. 3

[50] Jiaxiang Tang, Hang Zhou, Xiaokang Chen, Tianshu Hu, Errui Ding, Jingdong Wang, and Gang Zeng. Delicate Textured Mesh Recovery from NeRF via Adaptive Surface Refinement. *arXiv preprint arXiv:2303.02091*, 2022. 2

[51] Kartik Teotia, Hyeongwoo Kim, Pablo Garrido, Marc Habermann, Mohamed Elgharib, and Christian Theobalt. GaussianHeads: End-to-End Learning of Drivable Gaussian Head Avatars from Coarse-to-fine Representations. *ACM Trans. Graph.*, 43(6), 2024. 1

[52] Alex Trevithick, Matthew Chan, Michael Stengel, Eric Chan, Chao Liu, Zhiding Yu, Sameh Khamis, Manmohan Chandraker, Ravi Ramamoorthi, and Koki Nagano. Real-Time Radiance Fields for Single-Image Portrait View Synthesis. *ACM Transactions on Graphics (TOG)*, 2023. 14

[53] Cong Wang, Di Kang, He-Yi Sun, Shen-Han Qian, Zi-Xuan Wang, Linchao Bao, and Song-Hai Zhang. MeGA: Hybrid Mesh-Gaussian Head Avatar for High-Fidelity Rendering and Head Editing. *arXiv preprint arXiv:2404.19026*, 2024. 2

[54] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Transactions on Image Processing*, 2004. 8

[55] Erroll Wood, Tadas Baltrušaitis, Charlie Hewitt, Matthew Johnson, Jingjing Shen, Nikola Milosavljević, Daniel Wilde, Stephan Garbin, Toby Sharp, Ivan Stojiljković, et al. 3D Face Reconstruction with Dense Landmarks. In *European Conference on Computer Vision*, pages 160–177. Springer, 2022. 12

[56] Yue Wu, Yu Deng, Jiaolong Yang, Fangyun Wei, Chen Qifeng, and Xin Tong. AniFaceGAN: Animatable 3D-Aware Face Image Generation for Video Avatars. In *Advances in Neural Information Processing Systems*, 2022. 3

[57] Cheng-hsin Wuu, Ningyuan Zheng, Scott Ardisson, Rohan Bali, Danielle Belko, Eric Brockmeyer, Lucas Evans, Timothy Godisart, Hyowon Ha, Xuhua Huang, Alexander Hypes, Taylor Koska, Steven Krenn, Stephen Lombardi, Xiaomin Luo, Kevyn McPhail, Laura Millerschoen, Michal Perdoch, Mark Pitts, Alexander Richard, Jason Saragih, Junko Saragih, Takaaki Shiratori, Tomas Simon, Matt Stewart, Autumn Trimble, Xinshuo Weng, David Whitewolf, Chenglei Wu, Shoou-I Yu, and Yaser Sheikh. Multiface: A Dataset for Neural Face Rendering. In *ArXiv*, 2022. 2

[58] Yuelang Xu, Benwang Chen, Zhe Li, Hongwen Zhang, Lizhen Wang, Zerong Zheng, and Yebin Liu. Gaussian Head Avatar: Ultra High-Fidelity Head Avatar via Dynamic Gaussians. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1931–1941, 2024. 1, 2, 7

[59] Yinghao Xu, Wang Yifan, Alexander W. Bergman, Menglei Chai, Bolei Zhou, and Gordon Wetzstein. Efficient 3D Articulated Human Generation with Layered Surface Volumes. In *3DV*, 2024. 3, 4

[60] Yu-Jie Yuan, Yang-Tian Sun, Yu-Kun Lai, Yuewen Ma, Rongfei Jia, and Lin Gao. NeRF-Editing: Geometry Editing of Neural Radiance Fields. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 18353–18364, 2022. 3

[61] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The Unreasonable Effectiveness of Deep Features As a Perceptual Metric. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018. 8

[62] Yufeng Zheng, Wang Yifan, Gordon Wetzstein, Michael J. Black, and Otmar Hilliges. PointAvatar: Deformable Point-Based Head Avatars From Videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 1, 2, 7

[63] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo Magnification: Learning View Synthesis using Multiplane Images. *arXiv preprint arXiv:1805.09817*, 2018. 13

[64] Wojciech Zielonka, Timo Bolkart, and Justus Thies. Instant Volumetric Head Avatars. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 1, 3

LegacyAvatars: Volumetric Face Avatars For Traditional Graphics Pipelines

Supplementary Material

6. Implementation Details

Architecture details. The warp and texture predictors are implemented as 6-layer MLPs, where the subject-specific embeddings ϕ_w, ϕ_t are concatenated to the input coordinates and each of the learned latent codes in embedding matrices E_w, E_t are concatenated to the features after the third layer and fed through the model in parallel to produce a basis of warps and textures during each forward pass. In this formulation, variations across the elements of the bases are offloaded to single matrix, while the weights of the network are shared. This provides sufficient variability within the bases that can generate deformations and appearances with high expressivity, while also maintaining computational efficiency at training time.

3DMM and fitting. Our 3DMM includes linear bases of identity and expression. For each frame, we fit the 3DMM by estimating 599 probabilistic landmarks in 2D and optimizing identity, expression, rotation, and translation parameters of the 3DMM using a loss function that encourages consistency between per-vertex landmarks of the 3DMM and the observed 2D landmarks [55]. The parameter size of our expression model is $p = 63$.

Synthetic data. We use the synthetic face dataset introduced in [6], where we augment a subset of the subjects in this dataset to the real data. During training, we construct our batches by gathering 50% of the rays from the real subject and the other 50% from synthetic subjects. To ensure broad coverage of facial dynamics, synthetic expressions are drawn uniformly across the entire dataset. We illustrate the effectiveness of our joint real–synthetic training in Fig. 10, where we observe that in the absence of synthetic data, our model is prone to geometric instabilities and may fail to generalize to novel expressions.

Exported assets. The blend weights for our warp and texture bases can be efficiently computed at rendering time by linearly mapping $p = 63$ dimensional expression coefficients to $W = T = 12$ coefficients. Including the learned constant offset in this mapping, this results in two weight matrices of size 12×64.

Our canonical UV values, warp basis, and texture basis are all exported in the UV space, where the resolution and the precision can be modified for different application needs. Please refer to Fig. 11 for visualizations of our assets. For renders at 0.5K resolution, we found that mesh, warp map, and texture map resolutions of 512×512 are sufficient to preserve the overall visual quality. Here, a 32-bit precision is maintained for UV values, while the appearance is exported as 8-bit RGBA maps, where the view-dependent radiances are discarded.

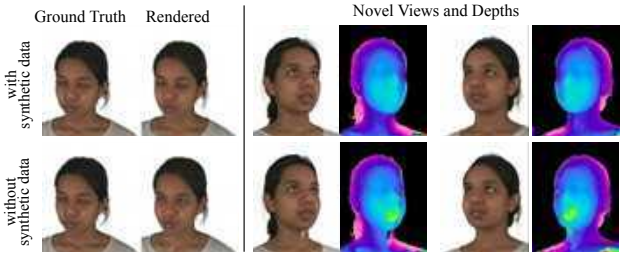

Figure 10. **Ablation on synthetic data.** The stability and overfitting challenges can be mitigated by introducing synthetic face data jointly trained with the real subject. This helps with generalization to novel expressions in addition to regularization of the learned face geometry.

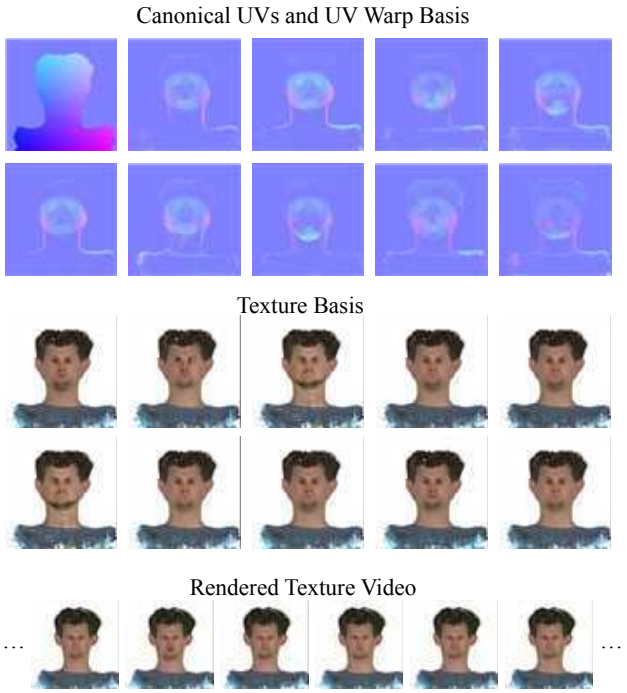

Figure 11. **Visualizations of the assets.** Illustrating a subset of the learned warps and appearances. With tracked expression coefficients of a 3DMM, these assets can be used to render a texture video shown at the bottom. All images are alpha-composited for visualization purposes.

Figure 12. **Comparisons on novel view synthesis.** Our model can synthesize novel views at a comparable visual quality to MonoAvatar++ [3] and GaussianAvatars [41], while being less prone to floater artifacts in NeRFs, and inherently preventing primitive ordering artifacts in 3DGS-based methods.

Table 2. **Ablation study on sizes of warp and texture bases.** Texture and warp basis sizes improve rendering quality and generalization. These metrics are obtained on cropped images that include the face region only.

	PSNR ↑	SSIM ↑	LPIPS ↓
$W = T = 16$	29.67 ± 1.36	0.897 ± 0.012	0.258 ± 0.017
$W = T = 8$	29.47 ± 1.39	0.894 ± 0.012	0.259 ± 0.017
$W = T = 4$	29.38 ± 1.41	0.892 ± 0.012	0.263 ± 0.016
$W = T = 2$	28.46 ± 1.21	0.882 ± 0.012	0.276 ± 0.019

7. Additional Results and Comparisons

Novel view synthesis comparisons. We provide comparisons on novel view synthesis with the state of the art methods, see Fig. 12. NeRF-based methods like MonoAvatar++ [3] can manifest floating artifacts and 3DGS-based methods may result in popping-like artifacts due to explicit sorting of primitives [42]. Our method is not prone to such artifacts by design, and the exported textured meshes can be edited by an artist, providing additional flexibility to remove visual seams as a post-processing step. Please see the supplementary video for better visualizations.

Warp and texture basis size ablations. To provide more insights into our model, we evaluate its expressiveness by modifying its warp and texture basis sizes. We illustrate our results in Fig. 13 and evaluation metrics in Tab. 2, where we observe that the overall rendering quality suffers and the renders manifest artifacts as we reduce the basis sizes. Furthermore, the model does not generalize to novel facial expressions and eye gazes as we reduce its capacity.

Figure 13. **Ablation on sizes of warp and texture bases.** Sufficient number of blendable warps and textures is crucial to achieve good rendering quality and generalization to novel expressions.

Figure 14. **Limitations**. Layered mesh representations may suffer from *shell* artifacts at extreme angles. While our approach outperforms existing baselines on visual quality on novel views at moderate poses *(left)*, at more extreme out-of-training profile views *(right)* it also suffers from *shell* artifacts similar to the baseline.

8. Limitations and Future Work

The basic premise of our representation is the discretization of scene components like geometry, appearance and deformations. While this enables traditional rendering, it also inherently limits the representational capacity compared to continuous volumes such as radiance fields and 3D Gaussians. Furthermore, since we project expression parameters onto low-dimensional blend weights, our model may exhibit blurring artifacts for extreme expressions, particularly those involving strong deformations around the mouth. We also note that our model does not explicitly account for the head pose, and hence the neck and torso regions may show slight instabilities in cases where the training sequences involve strong head pose variations.

Our layered mesh with transparency can be seen as a generalization of multiplane imaging (MPI) [63], where we instead learn a set of surfaces that follow a coarse face geometry and represent dynamic scenes. Due to such coarse

geometry, there exists a fundamental limit to the viewing angle range for artifact-free rendering [47]. At extreme angles, our representation can manifest shell artifacts, please see Fig. 14. We have also not tested our representation for controlling/animating large deformations such as head/neck rotations. These may also require additionally including and optimizing for a root joint UV-deformation of the layered mesh in the neck region. Optimizing and regularizing the topology of the layered mesh and the texture basis to improve the overall representational capacity could be an interesting future line of research.

The enrollment phase in our pipeline relies on ML training and inference, future work could explore simplifying this process. Recently, generative models have been used to learn a strong prior of face geometry and appearance [52], allowing direct regression of the face volume from even single images. Such quick and efficient techniques can further help reduce the compute and memory cost of the enrollment phase by learning a data-driven generative model. This can particularly help extend our representation to consumer use-cases such as monocular enrollment and tracking.

We believe that our work lays the important groundwork of building a novel representation that is capable of representing, animating and synthesizing volumetric effects in traditional graphics pipelines, and future work can build on and expand it towards even more practical settings.