FaceFolds: Meshed Radiance Manifolds for Efficient Volumetric Rendering of Dynamic Faces

SAFA C. MEDIN, MIT and Google, USA
GENGYAN LI, ETH Zurich and Google, Switzerland
RUOFEI DU, Google, USA
STEPHAN GARBIN, Google, United Kingdom
PHILIP DAVIDSON, Google, USA
GREGORY W. WORNELL, MIT, USA
THABO BEELER, Google, Switzerland
ABHIMITRA MEKA, Google, USA

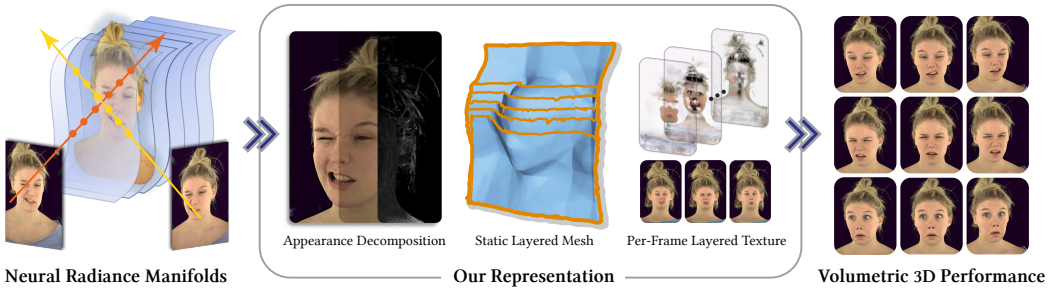

Fig. 1. We model a dynamic face sequence as a set of radiance manifolds, which are exported as a static layered mesh and an animated texture. This allows for smoothly controlling the quality vs. memory and compute footprints, while achieving efficient and photorealistic rendering of volumetric scenes using established graphics pipelines without any neural network integration. https://syntec-research.github.io/FaceFolds/

3D rendering of dynamic face captures is a challenging problem, and it demands improvements on several fronts—photorealism, efficiency, compatibility, and configurability. We present a novel representation that enables high-quality volumetric rendering of an actor's dynamic facial performances with minimal compute and memory footprint. It runs natively on commodity graphics soft- and hardware, and allows for a graceful trade-off between quality and efficiency. Our method utilizes recent advances in neural rendering, particularly learning discrete radiance manifolds to sparsely sample the scene to model volumetric effects. We achieve efficient modeling by learning a single set of manifolds for the entire dynamic sequence, while implicitly modeling appearance changes as temporal canonical texture. We export a single layered mesh and view-independent RGBA texture video that is compatible with legacy graphics renderers without additional ML integration. We demonstrate our method by rendering dynamic face captures of real actors in a game engine, at comparable photorealism to state-of-the-art neural rendering techniques at previously unseen frame rates.

CCS Concepts: • **Computing methodologies** → **Rendering**; **Machine learning**; **Graphics systems and interfaces**; **Mixed / augmented reality**.

Additional Key Words and Phrases: Volumetric Rendering, Face Modeling, Novel View Synthesis, Neural Radiance Fields, Performance Capture

Authors' addresses: Safa C. Medin, medin@mit.edu, MIT and Google, USA; Gengyan Li, gengyan.li@inf.ethz.ch, ETH Zurich and Google, Switzerland; Ruofei Du, me@duruofei.com, Google, USA; Stephan Garbin, stephangarbin@google.com, Google, United Kingdom; Philip Davidson, pdavidson@google.com, Google, USA; Gregory W. Wornell, gww@mit.edu, MIT, USA; Thabo Beeler, tbeeler@google.com, Google, Switzerland; Abhimitra Meka, abhim@google.com, Google, USA.

1 INTRODUCTION

Facial expressions are our primary means of communication—video streaming of our faces is a frequent part of our daily digital lives. Perceptually lossless and efficient video compression algorithms have enabled this application at consumer scale by achieving compute and memory efficiency. Other key enablers of this ubiquity of video streaming are 1) compatibility of compressed video playback with existing platform infrastructure such as operating systems and web browsers, and 2) easy trade-off of quality vs. memory through variable image resolution, allowing for seamless streaming across dynamically varying data bandwidth. But the same cannot be said for streaming and playback of personalized 3D face animation, which is a formidable task with major algorithmic challenges such as 3D reconstruction, animation, and transmission. While photorealism and compute/memory efficiency are obvious underlying challenges to each of these steps, other hurdles include compatibility with legacy software infrastructure and ability to smoothly trade-off quality vs. bandwidth. Solving these challenges promises widespread adoption of immersive experiences in 3D media content [22, 25], immersive AR/VR communication and 3D telepresence [10, 17, 18, 48].

Traditional graphics-based acquisition techniques for facial performances reconstructs 3D meshes and texture maps [13, 28] for each individual frame, which can be rendered very efficiently on commodity hardware. However, such traditional mesh-based representations encounter significant challenges in accurately modeling and rendering the fine-scale detailed geometry and complex appearance of hair and skin [17, 28], resulting in limited photorealism. On the other hand, recent advances in implicit volumetric representations such as neural radiance fields [44] and Gaussian splatting [32] have enabled high-quality acquisition and photorealistic rendering of dynamic human faces [41, 49, 51], including hair and skin at unprecedented quality [9, 53], without requiring explicit geometry reconstruction. But such techniques require deep ML integration for inference and are not natively compatible with existing graphics rendering platforms such as game engines. They also do not always provide means to trade-off quality with compute or memory efficiency.

We present a layered-mesh based volumetric representation for 3D view-synthesis of dynamic face performances that works efficiently on legacy renderers. At training time, our method takes inspiration from radiance manifolds [16] and models the scene density for the entire sequence using a set of static spatial manifolds of alpha values, and the temporal appearance changes as a time-conditioned UV-mapped radiance over these manifolds. The alpha-manifolds and corresponding temporal UV appearance maps are parameterized by dense neural networks, and the appearance is further decomposed into view-conditioned components. From this trained model, we export the radiance manifolds as a single layered mesh for the entire sequence, and the corresponding view-independent UV-space appearance as RGBA texture maps, encoded as a video. This exported representation is then rendered efficiently through simple alpha-blending of the textured mesh layers in any renderer. Unlike previous methods that cannot change the resolution or quality once trained, our layered mesh representation also allows for trading off the image quality for efficiency through standard operations like mesh decimation and subsampling of texture resolution. Our view-independent texture maps allow for easy rendering through texture look-up without the need for evaluating complex view-dependent reflectance or radiance transfer.

We demonstrate the efficacy of our representation using the Multiface dataset [61] consisting of multi-view real world captures of face performances of several actors. We qualitatively and quantitatively compare our method with state-of-the-art neural rendering techniques on both image quality and efficiency, and show previously unseen high frame-rate rendering of these sequences on the Unity game engine on a consumer device.

2 RELATED WORK

Impressive results have been achieved for editing and animation of face images and videos using purely 2D or hybrid approaches [8, 34, 43, 59, 64], but such methods do not guarantee consistent rendering when changing perspectives, which is crucial for gaming or XR applications.

Traditional 3D performance capture techniques [13, 22, 25, 28] have relied on textured meshes both due to ease of use for playback and editing, and to rely on rasterization for fast rendering. Parametric face models such as 3D morphable face models (3DMMs) [6, 21, 38] compress the dimensionality of mesh representations and make them differentiable, enabling efficient optimization frameworks that achieve the difficult task of canonical performance capture and playback. However, they suffer from low representational capacity and cannot model high-frequency effects in appearance and geometry. Some recent ML-based approaches build on such surface-based representations, *e.g.*, Neural Head Avatars employs a surface mesh with a dynamic texture [26], while IMAvatar [66] opts for an implicit surface. While efficient, these methods need to generate expression-specific appearance via feed-forward neural networks and have issues representing semi-transparent effects such as hair and beards. Adding deferred rendering networks, such as in the single-shot model [33] struggles to mitigate these effects and jeopardizes multi-view consistency.

Volumetric representations such as NeRFs parameterize a compressed emission-absorption volume via a multi-layer perceptron (MLP) trained with frequency encoding [45, 56]. Since the MLP has to be invoked for decompression at every sample location, inference performance stands out as one of the main limitations of the original work. Subsequently, hybrid representations have replaced the need for an MLP at inference time with (sparse) volumetric grids [11, 20, 24, 29, 40, 52, 54], explicit geometry [12, 16, 37], and hash grids [46]. Many recent works have exploited such volumetric representations to track or animate head models, but typically with computational requirements far exceeding our target application. Nerfies [49] and HyperNeRF [50] propose a continuous deformation field conditioned on a frame-specific latent code, which enables replay but requires an additional deformation MLP.

A popular class of methods rely on combining the low-dimensional tracking capabilities of 3DMMs with high representational power of volumetric radiance. GNARF [5] and Next3D [55] use a tri-plane representation with a form of mesh-based deformation but still require 2D super-resolution modules to generate image at the desired resolution. INSTA [69] increases the efficiency of this type of approach by using hash grids [69], while NerSemble replaces the explicit head model with a hash ensemble for increased generality [36]. Other methods [23, 30, 63] employ tetrahedral fields to directly deform a volumetric representation. MonoAvatar [2] uses a full volumetric NeRF model, where deformation is a function of K nearest points on the driving surface mesh. Ray tracing and the use of MLPs means that these methods are not nearly as fast as rasterization-based techniques. MVP [42] and its generative extension [10] offer a more efficient rendering pipeline by employing volumetric primitives attached to a guide mesh. More recent works have used other alternatives to NeRF-like volumes, such as point based representations [67] and Gaussian Splatting [51, 53] driven by combinations of explicit models like FLAME [38] and neural functions. While these models offer improved performance compared to volumetric methods, they still require custom rasterization components and are thus not trivial to deploy in existing software. For all these methods, use of 3DMMs makes these methods challenging to deploy without additional software components including iterative optimizers.

An alternate way to mitigate issues of predefined mesh topology and increase the flexibility of the representation is to use one [39, 57] or a collection of (implicit) surfaces [16]. In GRAM [16], the authors propose a discretized volumetric representation using a set of learned non-intersecting implicit surfaces which can be efficiently used for sampling radiance. GRAMInverter [15] and

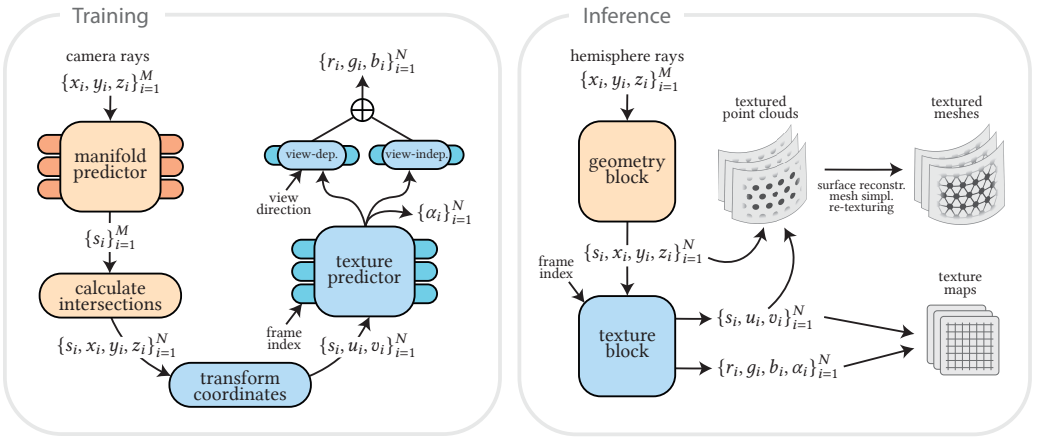

Fig. 2. **Training and inference pipelines.** Given a set of rays from the training cameras, we determine the intersection of these rays with a set of implicit manifolds predicted by a single manifold predictor. After transforming these intersections to UV-space coordinates, a texture predictor outputs RGBA texture maps conditioned on the video frame index. At inference time, we shoot rays from the surface of a designated hemisphere around the scene towards its center, obtaining a single geometry and a video texture. The view-dependent branch is bypassed to ensure that the appearance is fully diffuse.

GRAM-HD [62] propose high-resolution variants of GRAM, albeit at increased computational cost and model complexity not suitable for real-time applications. BakedAvatar [19] is a concurrently developed avatar animation technique that also uses radiance manifolds, and like previous methods, employs deformation fields conditioned on tracked 3DMM coefficients. However, their model still relies on an ML model to estimate appearance at inference time. During training, our method builds on layered implicit surfaces used in GRAM [16], but is not generative, focusing on high-quality, per-person specific models instead. We show that a static set of manifolds can model an entire performance sequence, and we offer improved performance by decomposing the radiance into view-dependent and -independent components, which allows us to export our results to a single explicit triangle mesh with video texture. This enables real-time playback in commonly available software packages at low computational cost.

3 METHOD

In this section, we first formulate the objective problem and present an overview of our pipeline. After describing how we process our datasets, we elaborate on how we leverage radiance manifolds to learn efficient 3D representations from multi-view videos. Finally, we describe how we export our representation to a single set of textured meshes that can be rendered natively on traditional graphics software while maintaining the rendering quality.

3.1 Definitions and Overview

Our objective is to learn a volumetric 3D representation of a subject that can be played back on game engines without any special neural network integration. Given a multi-view video of a subject with K frames, we learn a static geometry and a dynamic appearance model in an end-to-end fashion. We take inspiration from recent advances in implicit geometry representations [39, 57] that significantly outperform explicit 3D reconstruction techniques that rely on mesh or point cloud representations. In our pipeline, similar to GRAM [16], the geometry is modeled by a set of

2D manifolds, embodied as a set of implicit surfaces. But unlike GRAM, the appearance is learned as a UV-mapped dynamic radiance over these manifolds, instead of the 3D xyz-space. We learn N distinct manifolds defined implicitly by a single manifold predictor:

$$\mathcal{G} : (x, y, z) \in \mathbb{R}^3 \rightarrow s \in \mathbb{R} \tag{1}$$

Given a set of fixed scalars $\{s_i \in \mathbb{R} \mid i \in \mathcal{I} \triangleq \{1, 2, \ldots, N)\}$, which we refer to as *s-values*, the manifold predictor defines a set of N isosurfaces that represent our static geometry:

$$\mathcal{S}_i = \{(x, y, z) \mid \mathcal{G}(x, y, z) = s_i\}\,. \tag{2}$$

In our appearance model, we first transform points on each manifold to UV-space coordinates via a fixed function $f : (x, y, z) \in \mathcal{S}_i \rightarrow (u, v) \in [-1, 1] \times [-1, 1]$. For each manifold $i \in \mathcal{I}$ and each frame $j \in \mathcal{J} \triangleq \{1, 2, \ldots, K\}$, a texture predictor $\mathcal{T}_{ij}$ defines RGB and transparency fields:

$$\mathcal{T}_{ij} : (u, v) \in [-1, 1] \times [-1, 1] \rightarrow (r, g, b, \alpha) \in \mathbb{R}^4, \;\; i \in \mathcal{I} \;\&\; j \in \mathcal{J}\,. \tag{3}$$

Note that we do not estimate volume density as is the case for traditional volume rendering, but instead model 3D point transparency with an alpha value. This makes the radiance accumulation independent of the ray path, which is crucial for enabling the next step of exporting the learned manifolds as textured mesh layers. Our approach can be treated as a generalization of the multi-plane image representation [68], where we optimize arbitrary 2D surfaces instead of planes.

Once the training is completed, we collect samples across each manifold at a specific resolution and export these collections of 3D points, UV-coordinates, and RGBA values as a single set of topologized triangle meshes with UV-textures that can be efficiently rendered on legacy graphics renderers. We illustrate our training and inference pipelines in Figure 2.

3.2 Dataset

We use the publicly available dataset Multiface [61], from which we gather multi-view videos of 3 subjects from V1 of the dataset (subject IDs `002914589`, `002643814`, `5372021`), and 2 subjects from V2 (subject IDs `002421669`, `002645310`). We pick $K = 60$ consecutive frames from each video sequence where subjects perform facial expressions freely. For all subjects, we scale and transform the scene parameters such that the subjects are centered at the origin and oriented along the positive x-axis with up-vector aligned with the positive z-axis, and that the camera centers are distributed roughly 1 unit away from the subjects. We discard the cameras with elevation angles of more than 45° and azimuth angles of more than 90°, which yields a set of cameras in the $x > 0$ half-space. For each subject, we also hold out 2 cameras to perform quantitative evaluations. This yields us 23 training cameras for V1 subjects and 50 training cameras for V2. Finally, we downsample all images to 768×500 resolution while adjusting the camera parameters accordingly. We do not perform any background masking as our method is able to separate the foreground significantly either by restricting the scene volume or by placing the background into the view-dependent component of radiance, which is discarded at inference time.

3.3 Model Architecture and Training

Given K frames from a multi-view video with corresponding camera parameters, we sample M points uniformly along each camera ray and compute the intersections between these rays and the manifolds using the differentiable ray-manifold intersection algorithm [47] adopted in GRAM [16]. In our method, we sample $M = 256$ points along each ray, set the number of manifolds to $N = 12$, and train our model using videos consisting of $K = 60$ frames for each subject.

Previous techniques that model dynamic scenes with radiance manifolds [19, 60] have used explicit learned deformation of the manifolds to model scene animation. On the contrary, we

model all frames of a dynamic sequence with a single set of static manifolds, which poses a non-trivial challenge. To achieve this, our technique uses a unique sequence of steps, where we 1) transform intersection points to UV-space, 2) separate RGB predictions into view-independent and view-dependent components, and 3) estimate the transparency of each intersection directly without computing volume densities. Given an intersection $\mathbf{p} = (x, y, z) \in \mathbb{R}^3$ and a fixed center $\mathbf{c} \in \mathbb{R}^3$ of a unit sphere, we first project the intersection to the surface of the sphere and obtain $\mathbf{p}' \triangleq (x', y', z') = (\mathbf{p} - \mathbf{c})/\|\mathbf{p} - \mathbf{c}\|$. We then calculate the UV-space coordinates as $u = \frac{2}{\pi} \sin^{-1}(z')$ and $v = \frac{2}{\pi} \tan^{-1}(y'/x')$. The texture predictor receives UV-coordinates and the s-values of the intersections, as well as the frame index that is mapped to a learned latent code that conditions the predictions. The texture predictor is then branched into two layers that predict single-channel view-dependent compontent and three-channel view-independent component, former of which is conditioned on the view direction. The outputs of these branches are added together to produce the final RGB prediction. Such architecture allows us to discount view directional effects at inference time and achieve view-consistent rendering of exported meshes. Furthermore, it also helps us to separate most of the background from foreground by attributing the background to view-dependent component, particularly if the background is primarily grayscale, thus eliminating the need for explicit background removal. Finally, the alpha values are predicted as the raw output of our texture predictor, which can be directly used to alpha-composite our N-layered representation.

3.3.1 Training details. To promote training stability, we adopt the manifold initialization technique [1] used in GRAM [16] and begin training with sphere-like manifolds centered at $\mathbf{c}$. We optimize our model in an end-to-end fashion by adopting ℓ_1 loss between the predicted and ground truth pixel values. To ensure that the appearance is mostly explained by the view-independent component, we penalize the output of the view-dependent branch with ℓ_2 penalty. To promote more stability, we apply ℓ_2 regularization to all manifold predictor layers except for the final one. The manifold and texture predictors are optimized jointly by minimizing the loss function

$$\mathcal{L} = \mathcal{L}_{\text{rec}} + \lambda_{\text{vd}}\mathcal{L}_{\text{vd}} + \lambda_{\text{reg}}\mathcal{L}_{\text{reg}} \tag{4}$$

where $\mathcal{L}_{\text{rec}}$ is the ℓ_1 reconstruction loss, $\mathcal{L}_{\text{vd}}$ is the view-dependent branch penalty, and $\mathcal{L}_{\text{reg}}$ is the manifold regularization with $\lambda_{\text{vd}} = 1.0$ and $\lambda_{\text{reg}} = 0.0001$. The manifold and texture predictors are jointly optimized using the Adam optimizer [35] with initial learning rates of 0.0007 and 0.0010, and exponential decay rates of 0.05 and 0.20 per 200 000 iterations, respectively. Using a batch size of 32 768 rays sampled across all training frames and views, we perform training for 500 000 iterations for each subject.

3.3.2 Architecture Details. The manifold predictor is implemented as an MLP with 3 hidden layers of widths 128 and a final layer, where we choose the set of fixed scalars $\{s_i\}_{i=1}^{N}$ so that the initial concentric and sphere-like surfaces roughly falls within ±0.03 units of the surface of the face. These scalars are tuned slightly for each subject according to the size of the faces inferred by the tracked meshes provided in the Multiface dataset [61]. The texture predictor is implemented as an MLP with 8 hidden layers of widths 256 and 2 final layers corresponding to view-independent and view-dependent branches. Both input points and view directions undergo positional encodings and the frame indices are mapped to 32-dimensional learned latent codes through an embedding layer. Encoded input points and frame indices are fed into the MLP at its first layer whereas the encoded view directions are concatenated to the input to the view-dependent branch.

3.4 Exporting Layered Meshes and Textures

At test time, we gather points across the unit *hemisphere* by collecting azimuth and elevation angles in $[-\pi/2, \pi/2]$ uniformly at resolution $R \times R$ and shoot rays towards the sphere center $\mathbf{c}$. We set this

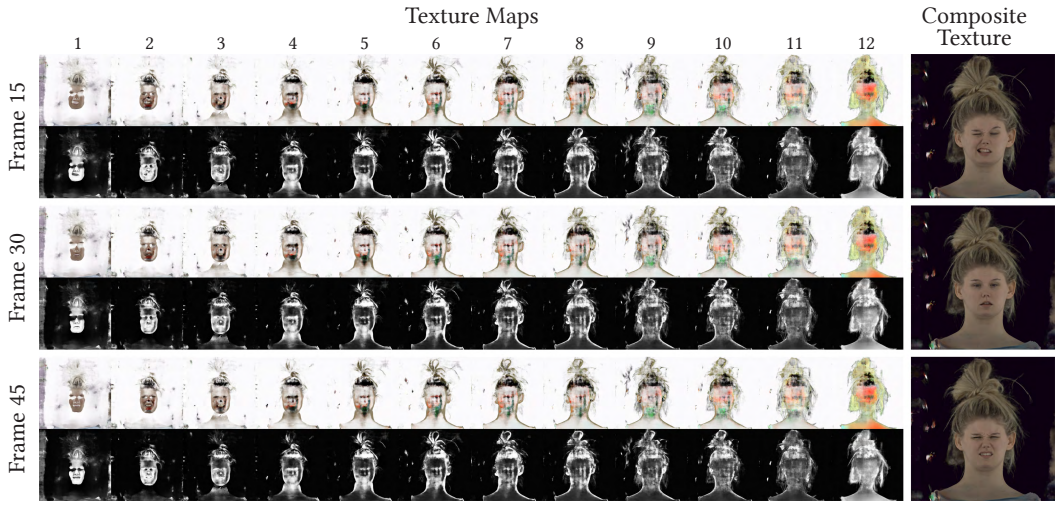

Fig. 3. **Video texture visualization.** We illustrate 3 frames (15^{th}, 30^{th}, and 45^{th} frames) from the learned texture video of subject `002914589`. For each frame, we show full RGBA and alpha-only UV-space texture maps in the top and bottom rows, respectively.

center 0.25 units away from the scene center in the direction of negative x-axis to ensure that the entire scene is encompassed by the hemisphere. This gives us $R \times R$ samples across each manifold with UV-coordinates that are distributed uniformly in $[-1, 1] \times [-1, 1]$. For each of the K frames, these points are used to query the texture predictor to yield N RGBA texture maps at resolution $R \times R$, where the view-dependent branch is bypassed to ensure the texture maps are fully diffuse. Our simple spherical projection yields reasonable texture mapping despite slight distortions near the edges of the maps [17]. We emphasize that we export the alpha channel as 8-bit images and hence store them very efficiently without sacrificing the visual quality. Finally, while the texture maps vary according to their respective frame indices, the geometry is constant across all frames.

To export our manifold-based representation to explicit surfaces, we reconstruct meshes from each of the N point clouds of size $R \times R$ via Poisson surface reconstruction [31], where the normals for each point are computed with respect to their neighboring points. We then simplify these meshes using a mesh decimation algorithm to reduce the number of vertices to a target mesh resolution $R^{\text{m}} \times R^{\text{m}}$. Finally, for each vertex in the simplified mesh, we determine the nearest point in the original point cloud and assign its corresponding UV-coordinate. The texture maps, on the other hand, can be downsampled to a specific target resolution $R^{\text{t}} \times R^{\text{t}}$ to meet the memory requirements of the renderer. To summarize, our final assets are: 1) a single set of N triangle meshes, each with number of vertices less than the target resolution $R^{\text{m}} \times R^{\text{m}}$ and 2) K sets of N RGBA texture maps at resolution $R^{\text{t}} \times R^{\text{t}}$ that form a UV texture video. We illustrate 3 frames from an example texture video along with composited texture maps in Figure 3.

3.5 Rendering on Game Engines

Our rendering pipeline in Unity using the exported layered mesh and texture sequences runs in real-time. We leverage two-pass deferred shading [14] on the GPU. When given a camera pose at runtime, we generate N G-buffers by shading each mesh layer and its opacity in a single render pass. Modern game engines use multiple render targets (MRT) for this purpose and we used culling masks to achieve this in Unity.

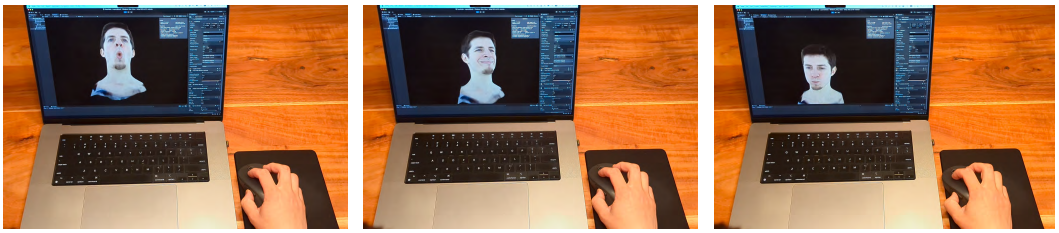

Fig. 4. **Free-viewpoint rendering on Unity.** Our representation allows for free-viewpoint rendering of dynamic 3D volumes on consumer hardware. Please refer to the supplementary material for the videos.

For a small number of layers (*e.g.*, $N < 16$, which is the maximum texture sampler count supported in Unity), we composite G-buffers on the GPU by tracing a ray through all layers in one pass, similar to accumulating luminance in the traditional volume rendering pipeline. For more than 16 layers, we suggest using a prefix sum algorithm [7] on the GPU for efficient layer compositing.

In our experiments with $N = 12$, we achieved real-time performance on a 2019 Macbook Pro with an M1 Max chip and Unity 2021.3. This was consistent across five datasets and over 1,000 frames. The average rendering time per frame was under 17 ms (above 60 FPS) at a rendering resolution of 2560×1440, even for our largest reconstructed mesh of 6.3M triangles.

4 EXPERIMENTS AND RESULTS

We evaluate the performance of our method on several subjects from the Multiface dataset [61], where we provide qualitative and quantitative comparisons against state-of-the-art neural rendering methods. We then perform more analysis of the configurability of our representation by assessing its performance with respect to varying number of manifolds, mesh resolution, and texture resolution.

4.1 Qualitative Results

We train our pipeline individually on 5 multi-view video sequences from [61], and illustrate our novel view synthesis results in Figure 5. Here, we provide comparisons against 4 state-of-the-art neural rendering methods—MonoAvatar [2], MVP [42], HyperNeRF [50], and Nerfies [49]. Despite discretizing the 3D volume into only $N = 12$ manifolds and hence sampling much fewer points across the scene during both training and evaluation, our approach manifests a comparable performance against other techniques. Furthermore, our technique does not require any MLP query or a custom pipeline during rendering and thus can be exported into a game engine, where we can perform free-viewpoint rendering of a dynamic 3D scene. We import our layered meshes and UV-textures into Unity and achieve the results demonstrated in Figure 4. We encourage the reader to refer to the supplementary material for video visualizations.

By interpolating between the learned latent codes of different frames at inference time, we can render our representation at higher frame rates to enhance the overall visual quality. We depict our frame interpolation results and provide comparisons in Figure 6, where we observe comparable performance against other methods. Please refer to the supplementary material for the videos.

4.2 Quantitative Results

For each subject, we perform quantitative evaluations on 2 held-out cameras across all $K = 60$ frames, totaling 120 images. In Table 1, we report average image quality metrics for our method and other methods in PSNR, SSIM [58], and LPIPS [65], where we consistently observe comparable performance across all methods. We also report VRAM usage, required disk storage, and frame

Fig. 5. **Qualitative comparisons.** Our method achieves comparable visual quality to state-of-the-art neural rendering techniques while facilitating very efficient rendering of dynamic sequences on legacy graphics software without any custom integration of ML pipelines.

Fig. 6. **Frame interpolation results.** Interpolating between the learned latent codes of frame indices allows us to achieve high-quality temporal interpolation between training frames with comparable performance to other approaches. Original frames are highligted in red.

rates for each of the methods, where we compress individual texture maps into a video and apply mesh compression to individual meshes using Draco[1] with no quantization of vertex positions and texture coordinates, and using the lowest compression amount. From our results, we observe that

[1]https://google.github.io/draco/

Table 1. **Quantitative comparisons.** Our method attains comparable visual quality across various metrics while utilizing significantly less VRAM and enabling much higher frame rates. The image quality metrics are averaged over a total of 600 test images of all 5 subjects.

Method	PSNR ↑	SSIM ↑	LPIPS ↓	VRAM ↓	Disk ↓	FPS ↑
Ours	25.49 ± 3.16	0.788 ± 0.069	0.356 ± 0.038	602 MiB	118 MiB	> 60
MonoAvatar [2]	24.59 ± 1.71	0.786 ± 0.042	0.341 ± 0.018	2746 MiB	296 MiB	0.33
MVP [42]	26.20 ± 2.34	0.738 ± 0.54	0.313 ± 0.025	1412 MiB	356 MiB	12.7
HyperNeRF [50]	26.91 ± 3.14	0.845 ± 0.0394	0.305 ± 0.041	2861 MiB	15 MiB	0.02
Nerfies [49]	26.11 ± 3.15	0.815 ± 0.042	0.332 ± 0.044	4205 MiB	15 MiB	0.03

Table 2. **Ablation on number of manifolds.** While significant gains in memory efficiency can be achieved by reducing the number of manifolds, it has a noteable effect on the visual quality. Numbers are averaged over a total of 240 test images of two subjects with IDs `002914589` and `002421669`. We report total disk storage required by the meshes and the video texture as well as the total number of triangles in all meshes.

Num. Manifolds	PSNR ↑	SSIM ↑	LPIPS ↓	Num. Tri. ↓	Disk ↓
12	25.54	0.760	0.341	6264412	125.1 MB
4	24.47	0.728	0.372	2088497	42.9 MB
1	22.51	0.704	0.395	522242	11.0 MB

our method is able to achieve higher frame rates despite utilizing a comparable amount of storage against other methods. Note here that other methods can be run with much lower VRAM usage by reducing the batch size down to single ray per batch.

All ML training, including ours and the state-of-the-art methods, was done on a workstation with NVIDIA V100 GPU. Since the state-of-the-art methods require a Linux workstation with NVIDIA GPU also for inference, they were evaluated and profiled on this same workstation. Our method does not require such special ML integration, and we perform our evaluation on the Unity game engine on a 2019 Macbook Pro laptop.

4.3 Ablation Studies

Since radiance manifolds [16] constrain 3D volumes to a number of implicit surfaces, the rendering quality is strongly influenced by the number of manifolds chosen before training. We provide qualitative and quantitative comparisons across different numbers of manifolds in Figure 7 and Table 2. Here, we observe that not only the rendering quality suffers with decreasing number of manifolds, capturing volumetric effects also requires a sufficient number of samples.

Our exported representation allows for trading off image quality with memory efficiency by performing standard operations such as mesh simplification and texture downsampling. After reconstructing a single set of meshes via Poisson surface reconstruction [31], we decimate each of these meshes to meet a target number of vertices. We illustrate qualitative and quantitative evaluations for varying mesh resolutions in Figure 7 and Table 3. We observe that the image quality does not undergo a significant drop until 16×16 resolution per surface. This is because our layered mesh representation does not manifest high-frequency changes while still allowing for state-of-the-art rendering quality via learned alpha-manifolds. This provides us with an extremely lightweight geometry representation without sacrificing any visual quality or volumetric effects.

The texture resolution, on the other hand, naturally plays a vital role in rendering quality. To compare, we individually subsample each of the texture maps across all manifolds and frames

Fig. 7. **Ablation results.** *Number of manifolds.* Using a sufficient number of manifolds is essential to attain photorealism and volumetric effects. *Mesh resolution.* We can decimate the exported meshes to much lower resolutions without sacrificing significant visual quality. *Texture resolution.* We can modify texture resolution arbitrarily at inference time to trade off image quality with rendering efficiency as desired.

Table 3. **Ablation on mesh resolution.** Despite reducing the memory footprint of the geometry, visual quality is maintained for resolutions as low as 32×32. For all mesh resolutions, the texture resolution is set to 1024×1024 and requires 5.9 MB storage after video compression. Numbers are averaged over 120 test images of a single subject with ID `002914589`.

Mesh resolution	PSNR ↑	SSIM ↑	LPIPS ↓	Num. Tri. ↓	Disk (mesh) ↓
512×512	30.10	0.858	0.284	6261920	118.1 MB
256×256	29.58	0.838	0.293	1377119	22.2 MB
128×128	29.57	0.839	0.290	250740	4.1 MB
64×64	29.49	0.837	0.289	43224	721 KB
32×32	29.12	0.831	0.291	9962	170 KB
16×16	27.26	0.795	0.306	2574	38 KB
8×8	23.81	0.705	0.364	728	11 KB

bilinearly, and render each frame at original training resolution 768×500. We illustrate our results in Figure 7 and Table 4 where we observe a notable reduction in quality at 128×128 resolution.

Table 4. **Ablation on texture resolution.** We can reduce the memory footprint of the video texture by simply subsampling each frame at inference time. For renders of resolution 768×500, a noteable drop in quality occurs at 256×256 texture resolution. For all texture resolutions, the mesh resolution is set to 512×512, hence the number of triangles and disk storage for meshes are constant and are 6261920 and 118.1 MB, respectively. Numbers are averaged over 120 test images of a single subject with ID `002914589`. Note that the texture video size will increase with the number of frames in the input video.

Texture resolution	PSNR ↑	SSIM ↑	LPIPS ↓	Disk (texture) ↓
1024×1024	30.10	0.858	0.284	5.9 MB
512×512	30.29	0.859	0.303	2.1 MB
256×256	29.51	0.836	0.341	667 KB
128×128	27.32	0.791	0.404	183 KB

5 LIMITATIONS AND FUTURE WORK

Since we export view-independent texture maps, our results do not exhibit view directional effects, as shown in Figure 8(a). Our pipeline could be extended to estimate specular or roughness maps from the view-dependent component to enable plausible specular relighting in graphics pipelines. While we demonstrate that the radiance manifolds can be leveraged to export lightweight geometry and appearance models, joint learning of these models poses challenges in training stability and requires careful tuning of relative learning rates of the two models. Besides these challenges, we observe that sampling across discrete manifolds instead of the entire 3D volume causes shell artifacts in extreme poses, as illustrated in Figure 8(b). Our experiments suggest that these artifacts can be mitigated by initializing the manifolds according to the size of the scene and keeping the distance between the consecutive manifolds sufficiently small, while also ensuring that these distances are large enough to allow for volumetric effects. In addition to these heuristics and the regularization of the manifold predictor weights, more sophisticated regularization techniques [27] can be utilized to promote training stability and improve geometry predictions.

We should note that our method involves sampling across the 3D volume by casting a single ray per pixel and query MLPs for each point across these rays, which is prone to producing aliasing artifacts [3] and lead to slow training [46]. Using more recent neural rendering frameworks that, for example, combine anti-aliasing techniques and fast grid-based representations [4] would be a possible next step towards improving the overall performance and visual quality of our method, although incorporating radiance manifolds into such pipelines can present nontrivial challenges.

While our spherical mapping to UV-space coordinates works well since our learned radiance manifolds are smooth and mostly convex, such mapping could be problematic for non-convex regions such as the nose. Also, our spherical sampling technique at inference time could be modified to have denser number of samples around more detailed regions such as the eyes and the hair. A non-linear UV mapping technique for the face in the context of radiance manifolds would be an exciting future research exploration.

6 CONCLUSION

In this work, we introduce FaceFolds, a novel representation for high-quality and memory-efficient volumetric rendering of dynamic facial performances in legacy renderers. We achieve this by leveraging radiance manifolds to model the animated performance. Our novel contribution includes a unique sequence of operations and design choices required to make the radiance manifold framework view-*independent* to enable exporting of the layered mesh and video textures. Once these assets are obtained, our representation does not require any ML-based operations or complex computations, and hence can be easily rendered in standard graphics software on consumer

Fig. 8. **Limitations.** While our method achieves good image quality in general, it suffers from some drawbacks. (a) Since we export only view-independent radiance to the texture, we cannot render specularities such as the ones on the nose, teeth and cheeks. (b) At extreme out-of-training-distribution viewpoints, our method sometimes exhibits shell artifacts due to transparency of the layered mesh from grazing views. Please refer to the supplementary material for video visualizations.

hardware at high frame rates. Our results demonstrate that we still achieve state-of-the-art rendering quality despite securing notable gains in memory and compute footprint.

REFERENCES

[1] Matan Atzmon and Yaron Lipman. 2020. SAL: Sign Agnostic Learning of Shapes From Raw Data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2565–2574. https://doi.org/10.1109/CVPR42600.2020.00264

[2] Ziqian Bai, Feitong Tan, Zeng Huang, Kripasindhu Sarkar, Danhang Tang, Di Qiu, Abhimitra Meka, Ruofei Du, Mingsong Dou, Sergio Orts-Escolano, Rohit Pandey, Ping Tan, Thabo Beeler, Sean Fanello, and Yinda Zhang. 2023. Learning Personalized High Quality Volumetric Head Avatars From Monocular RGB Videos. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. https://doi.org/10.1109/CVPR52729.2023.01620

[3] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. 2022. Mip-NeRF 360: Unbounded Anti-Aliased Neural Radiance Fields. *CVPR* (2022).

[4] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. 2023. Zip-NeRF: Anti-Aliased Grid-Based Neural Radiance Fields. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*. https://doi.org/10.1109/ICCV51070.2023.01804

[5] Alexander W. Bergman, Petr Kellnhofer, Wang Yifan, Eric R. Chan, David B. Lindell, and Gordon Wetzstein. 2023. Generative Neural Articulated Radiance Fields. arXiv:2206.14314 [cs.CV]

[6] Volker Blanz and Thomas Vetter. 1999. A Morphable Model for the Synthesis of 3D Faces. In *SIGGRAPH*. https://doi.org/10.1145/311535.311556

[7] Guy E Blelloch. 1990. Prefix Sums and Their Applications. (1990). https://doi.org/10.1109/ISTCS.1995.377028

[8] Marcel C. Buehler, Abhimitra Meka, Gengyan Li, Thabo Beeler, and Otmar Hilliges. 2021. VariTex: Variational Neural Face Textures. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. https://doi.org/10.1109/ICCV48922.2021.01363

[9] Marcel C Bühler, Kripasindhu Sarkar, Tanmay Shah, Gengyan Li, Daoye Wang, Leonhard Helminger, Sergio Orts-Escolano, Dmitry Lagun, Otmar Hilliges, Thabo Beeler, et al. 2023. Preface: A Data-Driven Volumetric Prior for Few-Shot Ultra High-Resolution Face Synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 3402–3413. https://doi.org/10.1109/ICCV51070.2023.00315

[10] 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. 2022. Authentic Volumetric Avatars From a Phone Scan. *ACM Transactions on Graphics* (2022). https://doi.org/10.1145/3528223.3530143

[11] Eric R. Chan, Connor Z. Lin, Matthew A. Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas Guibas, Jonathan Tremblay, Sameh Khamis, Tero Karras, and Gordon Wetzstein. 2021. Efficient Geometry-Aware 3D Generative Adversarial Networks. In *ArXiv*. https://doi.org/10.1109/CVPR52688.2022.01565

[12] Zhiqin Chen, Thomas Funkhouser, Peter Hedman, and Andrea Tagliasacchi. 2023. MobileNeRF: Exploiting the Polygon Rasterization Pipeline for Efficient Neural Field Rendering on Mobile Architectures. In *The Conference on Computer Vision and Pattern Recognition (CVPR)*. https://doi.org/10.1109/CVPR52729.2023.01590

[13] Alvaro Collet, Ming Chuang, Pat Sweeney, Don Gillett, Dennis Evseev, David Calabrese, Hugues Hoppe, Adam Kirk, and Steve Sullivan. 2015. High-Quality Streamable Free-Viewpoint Video. *ACM Trans. Graph* 34, 4, Article 69 (jul 2015), 13 pages. https://doi.org/10.1145/2766945

[14] Michael Deering, Stephanie Winner, Bic Schediwy, Chris Duffy, and Neil Hunt. 1988. The Triangle Processor and Normal Vector Shader: A VLSI System for High Performance Graphics. *ACM SIGGRAPH Computer Graphics* 22, 4 (1988), 21–30. https://doi.org/10.1145/378456.378468

[15] Yu Deng, Baoyuan Wang, and Heung-Yeung Shum. 2023. Learning Detailed Radiance Manifolds for High-Fidelity and 3D-Consistent Portrait Synthesis From Monocular Image. https://doi.org/10.1109/CVPR52729.2023.02017 arXiv:2211.13901 [cs.CV]

[16] Yu Deng, Jiaolong Yang, Jianfeng Xiang, and Xin Tong. 2022. GRAM: Generative Radiance Manifolds for 3D-Aware Image Generation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. https://doi.org/10.1145/3592460

[17] Ruofei Du, Ming Chuang, Wayne Chang, Hugues Hoppe, and Amitabh Varshney. 2018. Montage4D: Interactive Seamless Fusion of Multiview Video Textures. In *Proceedings of ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games (I3D)*. ACM, 124–133. https://doi.org/10.1145/3190834.3190843

[18] Ruofei Du, Ming Chuang, Wayne Chang, Hugues Hoppe, and Amitabh Varshney. 2019. Montage4D: Real-time Seamless Fusion and Stylization of Multiview Video Textures. *Journal of Computer Graphics Techniques* 8, 1 (17 Jan. 2019), 1–34. https://www.jcgt.org/published/0008/01/01/

[19] Hao-Bin Duan, Miao Wang, Jin-Chuan Shi, Xu-Chuan Chen, and Yan-Pei Cao. 2023. BakedAvatar: Baking Neural Fields for Real-Time Head Avatar Synthesis. *ACM Trans. Graph.* 42, 6, Article 225 (sep 2023), 14 pages. https://doi.org/10.1145/3618399

[20] Daniel Duckworth, Peter Hedman, Christian Reiser, Peter Zhizhin, Jean-François Thibert, Mario Lučić, Richard Szeliski, and Jonathan T. Barron. 2023. SMERF: Streamable Memory Efficient Radiance Fields for Real-Time Large-Scene Exploration. arXiv:2312.07541 [cs.CV]

[21] Bernhard Egger, William AP Smith, Ayush Tewari, Stefanie Wuhrer, Michael Zollhoefer, Thabo Beeler, Florian Bernard, Timo Bolkart, Adam Kortylewski, Sami Romdhani, et al. 2020. 3D Morphable Face Models—Past, Present, and Future. *Transactions on Graphics (ToG)* 39, 5 (2020), 1–38. https://doi.org/10.1145/3395208

[22] Graham Fyffe, Tim Hawkins, Chris Watts, Wan-Chun Ma, and Paul Debevec. 2011. Comprehensive Facial Performance Capture. *Computer Graphics Forum* 30, 2 (2011), 425–434. https://doi.org/10.1111/j.1467-8659.2011.01888.x

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

[24] Stephan J. Garbin, Marek Kowalski, Matthew Johnson, Jamie Shotton, and Julien P. C. Valentin. 2021. FastNeRF: High-Fidelity Neural Rendering at 200FPS. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. 14326–14335. https://doi.org/10.1109/ICCV48922.2021.01408

[25] Paulo Gotardo, Jérémy Riviere, Derek Bradley, Abhijeet Ghosh, and Thabo Beeler. 2018. Practical Dynamic Facial Appearance Modeling and Acquisition. *ACM Trans. Graph* 37, 6, Article 232 (dec 2018), 13 pages. https://doi.org/10.1145/3272127.3275073

[26] Philip-William Grassal, Malte Prinzler, Titus Leistner, Carsten Rother, Matthias Nießner, and Justus Thies. 2022. Neural Head Avatars From Monocular RGB Videos. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. https://doi.org/10.1109/CVPR52688.2022.01810

[27] Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. 2020. Implicit Geometric Regularization for Learning Shapes. *ArXiv Preprint ArXiv:2002.10099* (2020). https://arxiv.org/pdf/2002.10099

[28] Kaiwen Guo, Peter Lincoln, Philip Davidson, Jay Busch, Xueming Yu, Matt Whalen, Geoff Harvey, Sergio Orts-Escolano, Rohit Pandey, Jason Dourgarian, Danhang Tang, Anastasia Tkach, Adarsh Kowdle, Emily Cooper, Mingsong Dou, Sean Fanello, Graham Fyffe, Christoph Rhemann, Jonathan Taylor, Paul Debevec, and Shahram Izadi. 2019. The Relightables: Volumetric Performance Capture of Humans With Realistic Relighting. (2019). https://doi.org/10.1145/3355089.3356571

[29] Peter Hedman, Pratul P. Srinivasan, Ben Mildenhall, Jonathan T. Barron, and Paul Debevec. 2021. Baking Neural Radiance Fields for Real-Time View Synthesis. In *10.1109/ICCV48922.2021.00582*. 5855–5864. https://doi.org/10.1109/ICCV48922.2021.00582

[30] Kacper Kania, Stephan J. Garbin, Andrea Tagliasacchi, Virginia Estellers, Kwang Moo Yi, Julien Valentin, Tomasz Trzciński, and Marek Kowalski. 2023. BlendFields: Few-Shot Example-Driven Facial Modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. https://doi.org/10.1109/CVPR52729.2023.00047

[31] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. 2006. Poisson Surface Reconstruction. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*, Vol. 7. 0. https://doi.org/10.1145/2487228.2487237x26amp

[32] 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://doi.org/10.1145/3592433

[33] Taras Khakhulin, Vanessa Sklyarova, Victor Lempitsky, and Egor Zakharov. 2022. Realistic One-Shot Mesh-Based Head Avatars. In *European Conference of Computer Vision (ECCV)*. https://doi.org/10.1007/978-3-031-20086-1_20

[34] Hyeongwoo Kim, Pablo Garrido, Ayush Tewari, Weipeng Xu, Justus Thies, Matthias Nießner, Patrick Pérez, Christian Richardt, Michael Zollöfer, and Christian Theobalt. 2018. Deep Video Portraits. *ACM Transactions on Graphics (TOG)*

37, 4 (2018), 163. https://doi.org/10.1145/3197517.3201283

[35] Diederik P Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *ArXiv* (2014). https://doi.org/10.48550/arXiv.1412.6980

[36] Tobias Kirschstein, Shenhan Qian, Simon Giebenhain, Tim Walter, and Matthias Nießner. 2023. NeRSemble: Multi-View Radiance Field Reconstruction of Human Heads. *ACM Trans. Graph* 42, 4, Article 161 (jul 2023), 14 pages. https://doi.org/10.1145/3592455

[37] Gengyan Li, Abhimitra Meka, Franziska Mueller, Marcel C. Buehler, Otmar Hilliges, and Thabo Beeler. 2022. EyeNeRF: a hybrid representation for photorealistic synthesis, animation and relighting of human eyes. *ACM Trans. Graph.* 41, 4, Article 166 (jul 2022), 16 pages. https://doi.org/10.1145/3528223.3530130

[38] 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* (2017). https://doi.org/10.1145/3130800.3130813

[39] Zhaoshuo Li, Thomas Müller, Alex Evans, Russell H Taylor, Mathias Unberath, Ming-Yu Liu, and Chen-Hsuan Lin. 2023. Neuralangelo: High-Fidelity Neural Surface Reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 8456–8465. https://doi.org/10.1109/CVPR52729.2023.00817

[40] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. 2020. Neural Sparse Voxel Fields. In *Proceedings of the 34th International Conference on Neural Information Processing Systems* (Vancouver, BC, Canada) *(NIPS'20)*. Curran Associates Inc., Red Hook, NY, USA, Article 1313, 13 pages. https://doi.org/10.5555/3495724.3497037

[41] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. 2019. Neural Volumes: Learning Dynamic Renderable Volumes From Images. *ACM Trans. Graph* 38, 4, Article 65 (2019), 14 pages. https://doi.org/10.1145/3306346.3323020

[42] Stephen Lombardi, Tomas Simon, Gabriel Schwartz, Michael Zollhoefer, Yaser Sheikh, and Jason Saragih. 2021. Mixture of Volumetric Primitives for Efficient Neural Rendering. *ACM Trans. Graph* 40, 4, Article 59 (jul 2021), 13 pages. https://doi.org/10.1145/3450626.3459863

[43] Safa C. Medin, Bernhard Egger, Anoop Cherian, Ye Wang, Joshua B. Tenenbaum, Xiaoming Liu, and Tim K. Marks. 2022. MOST-GAN: 3D Morphable StyleGAN for Disentangled Face Image Manipulation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 1962–1971. https://doi.org/10.1609/aaai.v36i2.20091

[44] 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 *ECCV*. https://doi.org/10.1145/3503250

[45] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. 2021. Nerf: Representing Scenes As Neural Radiance Fields for View Synthesis. *Commun. ACM* (2021). https://doi.org/10.1145/3503250

[46] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. 2022. Instant Neural Graphics Primitives With a Multiresolution Hash Encoding. *ACM Transactions on Graphics* 41, 102 (2022). Issue 4. https://doi.org/10.1145/3528223.3530127

[47] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. 2020. Differentiable Volumetric Rendering: Learning Implicit 3d Representations Without 3d Supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 3504–3515. https://doi.org/10.1109/CVPR42600.2020.00356

[48] 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. 2016. Holoportation: Virtual 3D Teleportation in Real-Time. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*. ACM. https://doi.org/10.1145/2984511.2984517

[49] Keunhong Park, Utkarsh Sinha, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Steven M. Seitz, and Ricardo Martin-Brualla. 2021. Nerfies: Deformable Neural Radiance Fields. *ICCV* (2021). https://doi.org/10.1109/ICCV48922.2021.00581

[50] Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M. Seitz. 2021. HyperNeRF: A Higher-Dimensional Representation for Topologically Varying Neural Radiance Fields. *ACM Trans. Graph* 40, 6, Article 238 (dec 2021). https://doi.org/10.1145/3478513.3480487

[51] Shenhan Qian, Tobias Kirschstein, Liam Schoneveld, Davide Davoli, Simon Giebenhain, and Matthias Nießner. 2023. GaussianAvatars: Photorealistic Head Avatars With Rigged 3D Gaussians. *ArXiv Preprint ArXiv:2312.02069* (2023). https://doi.org/10.48550/arXiv.2312.02069

[52] Christian Reiser, Richard Szeliski, Dor Verbin, Pratul P. Srinivasan, Ben Mildenhall, Andreas Geiger, Jonathan T. Barron, and Peter Hedman. 2023. MERF: Memory-Efficient Radiance Fields for Real-Time View Synthesis in Unbounded Scenes. *ACM Transactions on Graphics* 42, 4 (2023). https://doi.org/10.1145/3592426

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

[54] Sara Fridovich-Keil and Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. 2022. Plenoxels: Radiance Fields Without Neural Networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

https://doi.org/10.1109/CVPR52688.2022.00542

[55] Jingxiang Sun, Xuan Wang, Lizhen Wang, Xiaoyu Li, Yong Zhang, Hongwen Zhang, and Yebin Liu. 2023. Next3D: Generative Neural Texture Rasterization for 3D-Aware Head Avatars. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. https://doi.org/10.1109/CVPR52729.2023.02011

[56] Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng. 2020. Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains. *NeurIPS* (2020). https://doi.org/10.5555/3495724.3496356

[57] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. 2021. NeuS: Learning Neural Implicit Surfaces by Volume Rendering for Multi-view Reconstruction. *arXiv preprint arXiv:2106.10689* (2021). https://doi.org/10.48550/arXiv.2106.10689

[58] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. 2004. Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Transactions on Image Processing* (2004). https://doi.org/10.1109/tip.2003.819861

[59] Xin Wen, Miao Wang, Christian Richardt, Ze-Yin Chen, and Shi-Min Hu. 2020. Photorealistic Audio-driven Video Portraits. *IEEE Transactions on Visualization and Computer Graphics* 26, 12 (2020), 3457–3466. https://doi.org/10.1109/TVCG.2020.3023573

[60] Yue Wu, Yu Deng, Jiaolong Yang, Fangyun Wei, Qifeng Chen, and Xin Tong. 2022. AniFaceGAN: Animatable 3D-Aware Face Image Generation for Video Avatars. *NeurIPS* (2022). https://doi.org/10.48550/arXiv.2210.06465

[61] 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. 2022. Multiface: A Dataset for Neural Face Rendering. In *ArXiv*. https://doi.org/10.48550/ARXIV.2207.11243

[62] Jianfeng Xiang, Jiaolong Yang, Yu Deng, and Xin Tong. 2023. GRAM-HD: 3D-Consistent Image Generation at High Resolution With Generative Radiance Manifolds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2195–2205. https://doi.org/10.1109/ICCV51070.2023.00209

[63] Yu-Jie Yuan, Yang-Tian Sun, Yu-Kun Lai, Yuewen Ma, Rongfei Jia, and Lin Gao. 2022. NeRF-Editing: Geometry Editing of Neural Radiance Fields. arXiv:2205.04978 [cs.GR]

[64] Egor Zakharov, Aleksei Ivakhnenko, Aliaksandra Shysheya, and Victor S. Lempitsky. 2020. Fast Bi-Layer Neural Synthesis of One-Shot Realistic Head Avatars. In *European Conference on Computer Vision*. https://doi.org/10.1007/978-3-030-58610-_31

[65] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. 2018. The Unreasonable Effectiveness of Deep Features As a Perceptual Metric. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. https://doi.org/10.1109/CVPR.2018.00068

[66] Yufeng Zheng, Victoria Fernández Abrevaya, Marcel C. Bühler, Xu Chen, Michael J. Black, and Otmar Hilliges. 2022. I M Avatar: Implicit Morphable Head Avatars From Videos. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. https://doi.org/10.1109/CVPR52688.2022.01318

[67] 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 (CVPR)*. https://doi.org/10.1109/CVPR52729.2023.02017

[68] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. 2018. Stereo Magnification: Learning view synthesis using multiplane images. *ACM Trans. Graph. (Proc. SIGGRAPH)* 37 (2018). https://arxiv.org/abs/1805.09817

[69] Wojciech Zielonka, Timo Bolkart, and Justus Thies. 2023. Instant Volumetric Head Avatars. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. https://doi.org/10.1109/CVPR52729.2023.00444