BlendFields: Few-Shot Example-Driven Facial Modeling

Kacper Kania[1,2,†] Stephan J. Garbin[3] Andrea Tagliasacchi[4,5,‡] Virginia Estellers[3]
Kwang Moo Yi[2] Julien Valentin[3] Tomasz Trzciński[1,6,7] Marek Kowalski[3]

Warsaw University of Technology[1] University of British Columbia[2] Microsoft[3]
Simon Fraser University[4] Google Brain[5] IDEAS NCBR[6] Tooploox[7]

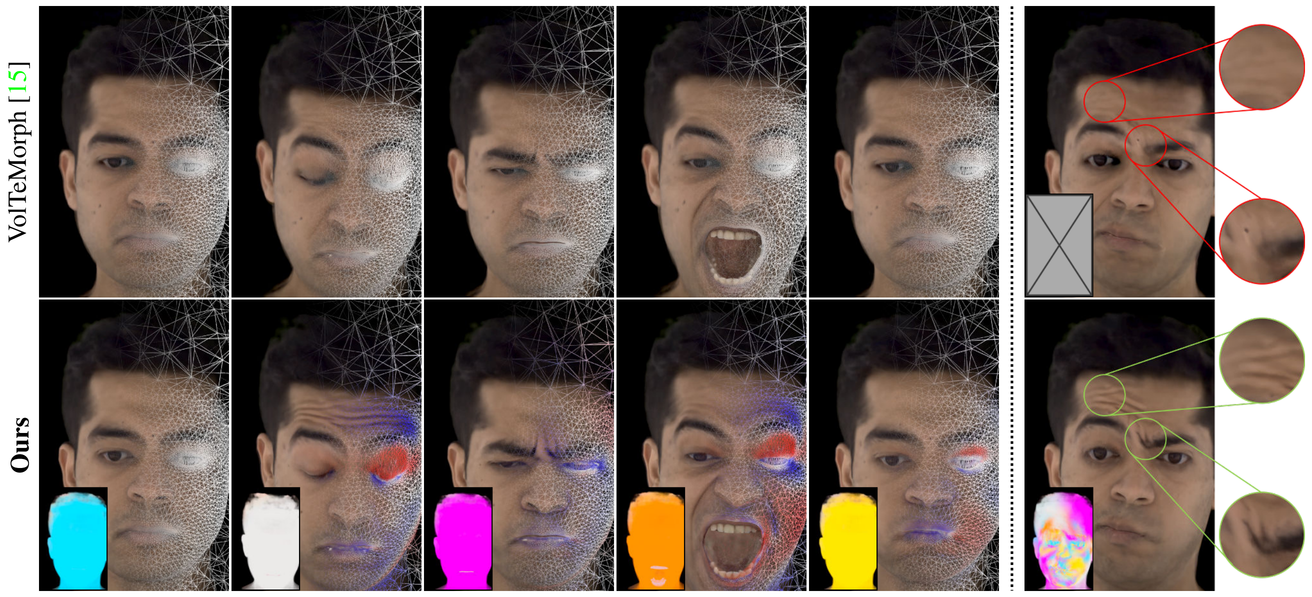


Figure 1. **Teaser** – Given five multi-view frames of different expressions, our approach generates a model capable of capturing the fine-grained details of a novel expression beyond the resolution of the underlying face model [15] (top right corner). This is achieved by *blending* the radiance fields computed for individual expressions, where the blending coefficients are modulated accordingly to *local* volumetric changes. These volumetric changes are measured as the difference in the tetrahedral volume of a mesh that deforms with the expression (🟥 increase, 🟦 decrease, and ⬜ no change in volume). Such an approach allows *BlendFields* to render sharp, expression-dependent details of the face without increasing the resolution of the mesh (bottom right corner).

Abstract

Generating faithful visualizations of human faces requires capturing both coarse and fine-level details of the face geometry and appearance. Existing methods are either data-driven, requiring an extensive corpus of data not publicly accessible to the research community, or fail to capture fine details because they rely on geometric face models that cannot represent fine-grained details in texture with a mesh discretization and linear deformation designed to model only a coarse face geometry. We introduce a method that bridges this gap by drawing inspiration from traditional computer graphics techniques. Unseen expressions are modeled by blending appearance from a sparse set of extreme poses. This blending is performed by measuring local volumetric changes in those expressions and locally reproducing their appearance whenever a similar expression is performed at test time. We show that our method generalizes to unseen expressions, adding fine-grained effects on top of smooth volumetric deformations of a face, and demonstrate how it generalizes beyond faces.

1. Introduction

Recent advances in neural rendering of 3D scenes [53] offer 3D reconstructions of unprecedented quality [36] with an ever-increasing degree of control [23, 29]. Human faces are of particular interest to the research commu-

[†] Work done during an internship at Microsoft Research Cambridge.
[‡] Work done at Simon Fraser University.

	NeRF [36]	NeRFies [42]	HyperNeRF [43]	NeRFace [13]	NHA [17]	AVA [7]	VolTeMorph [15]	**Ours**
Applicability beyond faces	✓	✓	✓	✗	✗	✗	✓	✓
Interpretable control	✗	✗	✗	✓	✓	✗	✓	✓
Data efficiency	✗	✓	✓	✗	✓	✗	✓	✓
Expression-dependent changes	✗	✗	✓	✓	✓	✓	✗	✓
Generalizability to unknown expressions	✗	✗	✗	✓	✓	✗	✓	✓

Table 1. **Comparison** – We compare several methods to our approach. Other methods fall short in data efficiency and applicability. For example, AVA [7] requires 3.1 million training images while VolTeMorph [15] cannot model expression-dependent wrinkles realistically.

nity [1, 13–15] due to their application in creating realistic digital doubles [32, 53, 75, 79].

To render facial expressions not observed during training, current solutions [1, 13–15] rely on *parametric* face models [6]. These allow radiance fields [36] to be controlled by facial parameters estimated by off-the-shelf face trackers [27]. However, parametric models primarily capture smooth deformations and lead to digital doubles that lack realism because fine-grained and expression-dependent phenomena like wrinkles are not faithfully reproduced.

Authentic Volumetric Avatars (AVA) [7] overcomes this issue by learning from a large multi-view dataset of synchronized and calibrated images captured under controlled lighting. Their dataset covers a series of dynamic facial expressions and multiple subjects. However, this dataset remains unavailable to the public and is expensive to reproduce. Additionally, training models from such a large amount of data requires significant compute resources. To democratize digital face avatars, more efficient solutions in terms of hardware, data, and compute are necessary.

We address the efficiency concerns by building on the recent works in Neural Radiance Fields [15, 70, 74]. In particular, we extend VolTeMorph [15] to render facial details learned from images of a sparse set of expressions. To achieve this, we draw inspiration from blend-shape correctives [26], which are often used in computer graphics as a data-driven way to correct potential mismatches between a simplified model and the complex phenomena it aims to represent. In our setting, this mismatch is caused by the low-frequency deformations that the tetrahedral mesh from VolTeMorph [15], designed for real-time performance, can capture, and the high-frequency nature of expression wrinkles.

We train multiple radiance fields, one for each of the K sparse expressions present in the input data, and blend them to correct the low-frequency estimate provided by VolTeMorph [15]; see Fig. 1. We call our method *BlendFields* since it resembles the way blend shapes are employed in 3DMMs [6]. To keep K small (*i.e.*, to maintain a few-shot regime), we perform local blending to exploit the known correlation between wrinkles and changes in local differential properties [21,45]. Using the dynamic geometry of [15], local changes in differential properties can be easily extracted by analyzing the tetrahedral representation underlying the corrective blendfields of our model.

Contributions. We outline the main qualitative differences between our approach and related works in Tab. 1, and our empirical evaluations confirm these advantages. In summary, we:

- extend VolTeMorph [15] to enable modeling of high-frequency information, such as expression wrinkles in a few-shot setting;
- introduce correctives [6] to neural field representations and activate them according to local deformations [45];
- make this topic more accessible with an alternative to techniques that are data and compute-intensive [7];
- show that our model generalizes beyond facial modeling, for example, in the modeling of wrinkles on a deformable object made of rubber.

2. Related Works

Neural Radiance Fields (NeRF) [36] is a method for generating 3D content from images taken with commodity cameras. It has prompted many follow-up works [4, 5, 19, 33, 35, 42, 46, 48, 52, 55, 68, 76] and a major change in the field for its photorealism. The main limitations of NeRF are its rendering speed, being constrained to static scenes, and lack of ways to control the scene. Rendering speed has been successfully addressed by multiple follow-up works [16, 18, 73]. Works solving the limitation to static scenes [1, 2, 22, 40, 59, 61, 67, 78] and adding explicit control [8, 23, 24, 50, 57, 72, 74] have limited resolution or require large amounts of training data because they rely on controllable coarse models of the scene (*e.g.*, 3DMM face model [6]) or a conditioning signal [43]. Methods built on an explicit model are more accessible because they require less training data but are limited by the model's resolution. Our technique finds a sweet spot between these two regimes by using a limited amount of data to learn details missing in the controlled model and combining them together. Our experiments focus on faces because high-quality data and 3DMM face models are publicly available, which are the key component for creating digital humans.

2.1. Radiance Fields

Volumetric representations [56] have grown in popularity because they can represent complex geometries like hair more accurately than mesh-based ones. Neural Radiance Fields (NeRFs) [36] model a radiance volume with a coordinate-based MLP learned from posed images. The MLP predicts density $\sigma(\mathbf{x})$ and color $\mathbf{c}(\mathbf{x}, \mathbf{v})$ for each point $\mathbf{x}$ in the volume and view direction $\mathbf{v}$ of a given camera. To supervise the radiance volume with the input images, each image pixel is associated with a ray $\mathbf{r}(t)$ cast from the camera center to the pixel, and samples along the ray are accumulated to determine the value of the image pixel $C(\mathbf{r})$:

$$C(\mathbf{r}) = \int_{t_n}^{t_f} T(t)\, \sigma(\mathbf{r}(t))\, \mathbf{c}(\mathbf{r}(t), \mathbf{v}) dt, \tag{1}$$

where t_n and t_f are near and far planes, and

$$T(t) = \exp\left(- \int_{t_n}^{t} \sigma(\mathbf{r}(s)) ds \right), \tag{2}$$

is the transmittance function [51]. The weights of the MLP are optimized to minimize the mean squared reconstruction error between the target pixel and the output pixel.Several methods have shown that replacing the implicit functions approximated with an MLP for a function discretized on an explicit voxel grid results in a significant rendering and training speed-up [16, 18, 28, 49, 73].

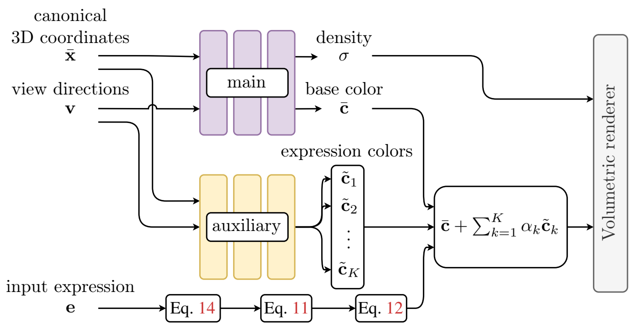
2.2. Animating Radiance Fields

Several works exist to animate the scene represented as a NeRF. D-NeRF uses an implicit deformation model that maps sample positions back to a canonical space [44], but it cannot generalize to unseen deformations. Several works [13, 42, 43, 54] additionally account for changes in the observed scenes with a per-image latent code to model changes in color as well as shape, but it is unclear how to generalize the latents when animating a sequence without input images. Similarly, works focusing on faces [1, 13, 14, 80] use parameters of a face model to condition NeRF's MLP, or learn a latent space of images and geometry [7, 30–32, 34, 58] that does not extrapolate beyond expressions seen during training.

In contrast to these approaches, we focus on using as little temporal training data as possible (*i.e.* five frames) while ensuring generalization. For this reason, we build our method on top of VolTeMorph [15], that uses a parametric model of the face to track the deformation of points in a volume around the face and builds a radiance field controlled by the parameters of a 3DMM. After training, the user can render an image for any expression of the face model. However, the approach cannot generate expression-dependent high-frequency details; see Fig. 1.

Similarly, NeRF-Editing [74] and NeRF Cages [70] propose to use tetrahedral meshes to deform a single-frame NeRF reconstruction. The resolution of the rendered scenes in these methods is limited by the resolution of the tetrahedral cage, which is constrained to a few thousand elements.

We discuss additional concurrent works in `Supplementary`.



canonical 3D coordinates $\bar{\mathbf{x}}$
view directions $\mathbf{v}$
density σ
main
base color $\bar{\mathbf{c}}$
expression colors
$\tilde{\mathbf{c}}_1$
$\tilde{\mathbf{c}}_2$
$\tilde{\mathbf{c}}_K$
auxiliary
$\bar{\mathbf{c}} + \sum_{k=1}^{K} \alpha_k \tilde{\mathbf{c}}_k$
Volumetric renderer
input expression $\mathbf{e}$
Eq. 14
Eq. 11
Eq. 12

Figure 2. **BlendFields** – We implement our approach as a volumetric model, where the *appearance* (i.e. radiance) is the sum of the main appearance corrected by blending a small set of K expression-specific appearances. These appearances are learnt from extreme expressions, and then blended at test-time according to blend weights computed as a function of the input expression $\mathbf{e}$.

2.3. Tetrahedral Cages

To apply parametric mesh models, it is necessary to extend them to the volume to support the volumetric representation of NeRF. Tetrahedral cages are a common choice for their simplicity and ubiquity in computer graphics [15, 70, 72]. For example, VolTeMorph uses dense landmarks [63] to fit a parametric face model whose blendshapes have been extended to a tetrahedral cage with finite elements method [9]. These cages can be quickly deformed and raytraced [37] using parallel computation on GPUs [11] while driving the volume into the target pose and allowing early ray termination for fast rendering. We further leverage the tetrahedral cage and use its differential properties [21], such as a local volume change, to model high-frequency details. For example, a change from one expression to another changes the volume of tetrahedra in regions where wrinkle formation takes place while it remains unchanged in flat areas. We can use this change in volume to select which of the trained NeRF expressions should be used for each tetrahedron to render high-frequency details.

3. Method

We introduce a volumetric model that can be driven by input expressions and visualize it in in Fig. 2. We start this section by explaining our model and how we train and drive it with novel expressions utilizing parametric face models (Sec. 3.1). We then discuss how to compute measures of volume expansion and compression in the tetrahedra to combine volumetric models of different expressions (Sec. 3.2) and how we remove artifacts in out-of-distribution settings (Sec. 3.3). We conclude this section with implementation details (Sec. 3.4).

3.1. Our model

Given a neutral expression $\bar{\mathbf{e}}$, and a collection of posed images $\{C_c\}$ of this expression from multiple views, VolTe-

Morph [15] employs a map $\mathcal{T}$ to fetch the density and radiance[1] for a new expression $\mathbf{e}$ from the *canonical* frame defined by expression $\bar{\mathbf{e}}$:

$$\mathbf{c}(\mathbf{x};\mathbf{e}) = \bar{\mathbf{c}}(\bar{\mathbf{x}}), \quad \bar{\mathbf{x}} = \mathcal{T}(\mathbf{x};\mathbf{e}\to\bar{\mathbf{e}}) \tag{3}$$

$$\sigma(\mathbf{x};\mathbf{e}) = \bar{\sigma}(\bar{\mathbf{x}}), \quad \bar{\mathbf{x}} = \mathcal{T}(\mathbf{x};\mathbf{e}\to\bar{\mathbf{e}}) \tag{4}$$

$$\mathcal{L}_{\text{rgb}} = \mathbb{E}_{C\sim\{C_c\}}\,\mathbb{E}_{\mathbf{r}\sim C}\,\mathcal{L}^{\mathbf{r}}_{\text{rgb}} \tag{5}$$

$$\mathcal{L}^{\mathbf{r}}_{\text{rgb}} = \|C(\mathbf{r};\mathbf{e}) - C(\mathbf{r})\|_2^2, \tag{6}$$

where $C(\mathbf{r};\mathbf{e})$ is a pixel color produced by our model conditioned on the input expression $\mathbf{e}$, $C(\mathbf{r})$ is the ground-truth pixel color, and the mapping $\mathcal{T}$ is computed from smooth deformations of a tetrahedral mesh to render unseen expressions $\mathbf{e}$. We use expression vectors $\mathbf{e}$ from parametric face models, such as FLAME [27, 65]. However, as neither density nor radiance change with $\mathbf{e}$, changes in appearance are limited to the low-frequency deformations that $\mathcal{T}$ can express. For example, this model cannot capture high-frequency dynamic features like expression wrinkles. We overcome this limitation by conditioning radiance on expression. For this purpose, we assume radiance to be the sum of a template radiance (*i.e.* rest pose appearance of a subject) and K residual radiances (*i.e.* details belonging to corresponding facial expressions):

$$\mathbf{c}(\mathbf{x};\mathbf{e}) = \bar{\mathbf{c}}(\mathbf{x}) + \sum_{k=k}^{K} \alpha_k(\mathbf{x};\mathbf{e})\cdot\tilde{\mathbf{c}}_k(\mathbf{x}), \tag{7}$$

We call our model *blend fields*, as it resembles the way in which blending is employed in 3D morphable models [6] or in wrinkle maps [41]. Note that we assume that pose-dependent geometry can be effectively modeled as a convex combination of colors $[\tilde{\mathbf{c}}(\mathbf{x})]_{k=1}^K$, since we employ the same density fields as in (4). In what follows, for convenience, we denote the vector field of blending coefficients as $\boldsymbol{\alpha}(\mathbf{x}){=}[\alpha_k(\mathbf{x})]_{k=1}^K$.

Training the model We train our model by assuming that we have access to a small set of K images $\{C_k\}$ (example in Fig. 3), each corresponding to an "extreme" expression $\{\mathbf{e}_k\}$, and minimize the loss:

$$\mathcal{L}_{\text{rgb}} = \mathbb{E}_k\,\mathbb{E}_{\mathbf{r}}\,\|C_K(\mathbf{r};\mathbf{e}_k) - C_k(\mathbf{r})\|_2^2 \tag{8}$$

$$\text{where} \quad \forall\mathbf{x},\ \boldsymbol{\alpha}(\mathbf{x}) = \mathbb{1}_k, \tag{9}$$

where $\mathbb{1}_k$ is the indicator vector, which has value one at the k-th position and zeroes elsewhere, and C_K represents the output of integrating the radiances in (7) along a ray.

Driving the model To control our model given a novel expression $\mathbf{e}$, we need to map the input expression code to the corresponding blendfield $\boldsymbol{\alpha}(\mathbf{x})$. We parameterize the blend field as a vector field discretized on the vertices $\mathbf{V}(\mathbf{e})$ of our tetrahedral mesh, where the vertices deform according to the given expression. The field is discretized on vertices, but it can be queried within tetrahedra using linear FEM bases [38]. Our core intuition is that when the (local) geometry of the mesh matches the local geometry in one of the input expressions, the corresponding expression blend weight should be locally activated. More formally, let $\mathbf{v}{\in}\mathbf{V}$ be a vertex in the tetrahedra and $\mathcal{G}(\mathbf{v})$ a local measure of volume on the vertex described in Sec. 3.2, then

$$\mathcal{G}(\mathbf{v}(\mathbf{e})){\approx}\mathcal{G}(\mathbf{v}(\mathbf{e}_k)) \implies \boldsymbol{\alpha}(\mathbf{v}(\mathbf{e})) \approx \mathbb{1}_k. \tag{10}$$

To achieve this we first define a *local* similarity measure:

$$[\Delta\mathcal{G}_k(\mathbf{v}(\mathbf{e}))] = [\|\mathcal{G}(\mathbf{v}(\mathbf{e})){-}\mathcal{G}(\mathbf{v}(\mathbf{e}_k))\|_2^2] \in \mathbb{R}^K \tag{11}$$

and then gate it with softmax (with temperature $\tau{=}10^6$) to obtain vertex blend weights:

$$\boldsymbol{\alpha}(\mathbf{v}(\mathbf{e})) = \text{softmax}_\tau\{\Delta\mathcal{G}_k(\mathbf{v}(\mathbf{e}))\} \in [0,1]^K \tag{12}$$

which realizes (10), as well as preserves the typically desirable characteristics of blend weights:

- *partition of unity*: $\forall\mathbf{x}\ \ \boldsymbol{\alpha}(\mathbf{x}) \in [0,1]^K$ and $\|\boldsymbol{\alpha}(\mathbf{x})\|_1{=}1$
- *activations sparsity*: minimizers of $\|\boldsymbol{\alpha}(\mathbf{x})\|_0$

where the former ensures any reconstructed result is a *convex* combination of input data, and the latter prevents destructive interference [20].

3.2. Local geometry descriptor

Let us consider a tetrahedron as the matrix formed by its vertices $\mathbf{T}{=}\{\mathbf{v}_i\} \in \mathbb{R}^{3\times4}$, and its edge matrix as $\mathbf{D} = [\mathbf{v}_3 - \mathbf{v}_0, \mathbf{v}_2 - \mathbf{v}_0, \mathbf{v}_1 - \mathbf{v}_0]$. Let us denote $\bar{\mathbf{D}}$ as the edge matrix in rest pose and $\mathbf{D}$ as one of the deformed tetrahedra (*i.e.*, due to expression). From classical FEM literature, we can then compute the change in volume of the tetrahedra from the determinant of its deformation gradient [21]:

$$\Delta\mathcal{V}(\mathbf{T}) = \det(\mathbf{D}\cdot\bar{\mathbf{D}}^{-1}) \tag{13}$$

[1] We omit view-dependent effects to simplify notation but include them in our implementation.

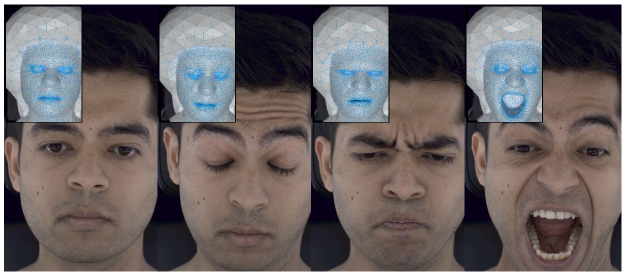


Figure 3. **Data** – We represent the data as a multi-view, multi-expression images. For each of these images, we obtain parameters of a parametric model, such as FLAME [27] to get: an expression vector $\mathbf{e}$ and a tetrahedral mesh described by vertices $\mathbf{V}(\mathbf{e})$. We highlight that our approach works for any object if a rough mesh and its descriptor are already provided.

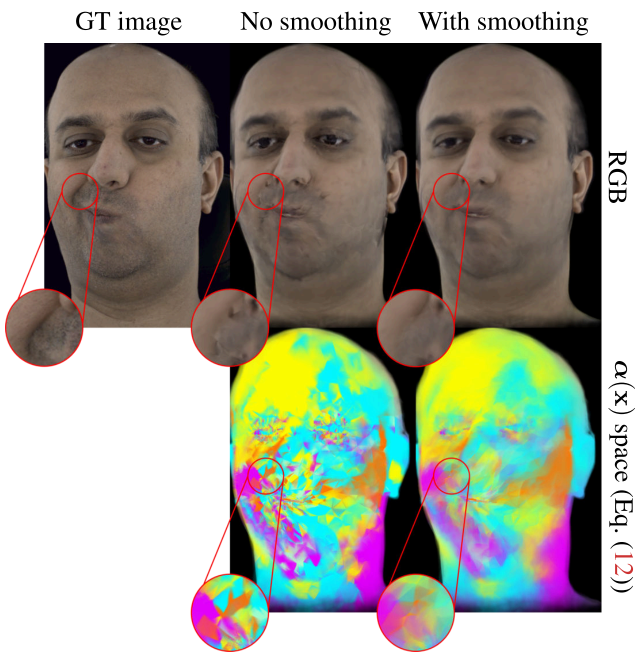


Figure 4. **Laplacian smoothing** – To combat artifacts stemming from calculating weights α across multiple expressions, which may assign different expressions to neighboring tetrahedra, we apply Laplacian smoothing [12]. As seen in the bottom row, smoothing gives a more consistent expression assignment.

We then build a local volumetric descriptor for a specific (deformed) vertex $\mathbf{v}(\mathbf{e})$ by concatenating the changes in volumes of neighboring (deformed) tetrahedra:

$$\mathcal{G}(\mathbf{v}(\mathbf{e})) = \bigoplus_{\mathbf{T}\in\mathcal{N}(\mathbf{v})} \Delta\mathcal{V}(\mathbf{T}(\mathbf{e})), \tag{14}$$

where $\bigoplus$ denotes concatenation and $\mathcal{N}(\mathbf{v})$ topological neighborhood of a vertex $\mathbf{v}$.

3.3. Blend-field smoothness

High-frequency spatial changes in blendfields can cause visual artifacts, see Fig. 4. We overcome this issue by applying a small amount of smoothing to the blendfield. Let us denote with $\mathbf{A}=\{\boldsymbol{\alpha}(\mathbf{v}_v)\}$ the matrix of blend fields defined on all mesh vertices, and with $\mathbf{L}$ the Laplace-Beltrami operator for the tetrahedral mesh induced by linear bases [21]. We exploit the fact that at test-time, the field is discretized on the mesh vertices, execute a diffusion process on the tetrahedral manifold, and, to avoid instability problems, implement it via backward Euler [12]:

$$\mathbf{A}^{\text{diff}} = (\mathbf{I} - \lambda_{\text{diff}}\mathbf{L})^{-1}\mathbf{A}^{n}. \tag{15}$$

3.4. Implementation details

We build on VolTeMorph [15] and use its volumetric 3DMM face model. However, the same methodology can be used with other tetrahedral cages built on top of 3DMM face models. The face model is created by extending the blendshapes of the parametric 3DMM face model [65] to a tetrahedral cage that defines the support in the neural radiance field. It has four bones controlling global rotation, the neck and the eyes with linear blend skinning, 224 expression blendshapes, and 256 identity blendshapes. Our face radiance fields are thus controlled and posed with the identity, expression, and pose parameters of the 3DMM face model [65], can be estimated by a real-time face tracking system like [64], and generalize convincingly to expressions representable by the face model.

Training. During training, we sample rays from a single frame to avoid out-of-memory issues when evaluating the tetrahedral mesh for multiple frames. Each batch contains 1024 rays. We sample $N_{\text{coarse}}=128$ points along a single ray during the coarse sampling and $N_{\text{importance}}=64$ for the importance sampling. We train the network to minimize the loss in Eq. (8) and sparsity losses with standard weights used in VolTeMorph [15, 18]. We train the methods for 5×10^5 steps using Adam [25] optimizer with learning rate 5×10^{-4} decaying exponentially by factor of 0.1 every 5×10^5 steps.

Inference. During inference, we leverage the underlying mesh to sample points around tetrahedra hit by a single ray. Therefore, we perform a single-stage sampling with $N=N_{\text{coarse}}+N_{\text{importance}}$ samples along the ray. When extracting the features (Eq. (14)), we consider $|\mathcal{N}(\mathbf{v})|=20$ neighbors. For the Laplacian smoothing, we set $\lambda_{\text{diff}}=0.1$ and perform a single iteration step. Geometric-related operations impose negligible computational overhead.

4. Experiments

We evaluate all methods on data of four subjects from the publicly available Multiface dataset [66]. We track the face for eight manually-selected ”extreme” expressions. We then select $K=5$ expressions the combinations of which show as many wrinkles as possible. Each subject was captured with ≈38 cameras which gives ≈190 training images per subject[2]. We use Peak Signal To Noise Ratio (PSNR) [3], Structural Similarity Index (SSIM) [60] and Learned Perceptual Image Patch Similarity (LPIPS) [77] to measure the performance of the models. Each of the rendered images has a resolution of 334×512 pixels.

As baselines, we use the following approaches: the original, static NeRF [36], NeRF conditioned on an expression code concatenated with input points $\mathbf{x}$, NeRFies [42], HyperNeRF[3] [43], and VolTeMorph [15]. We replace the learnable code in NeRFies and HyperNeRF with the expression code $\mathbf{e}$ from the parametric model. Since VolTeMorph can be trained on multiple frames, which should

[2]To train BlendFields for a single subject we use ≈ 0.006% of the dataset used by AVA [7].

[3]We use two architectures proposed by Park *et al.* [43].

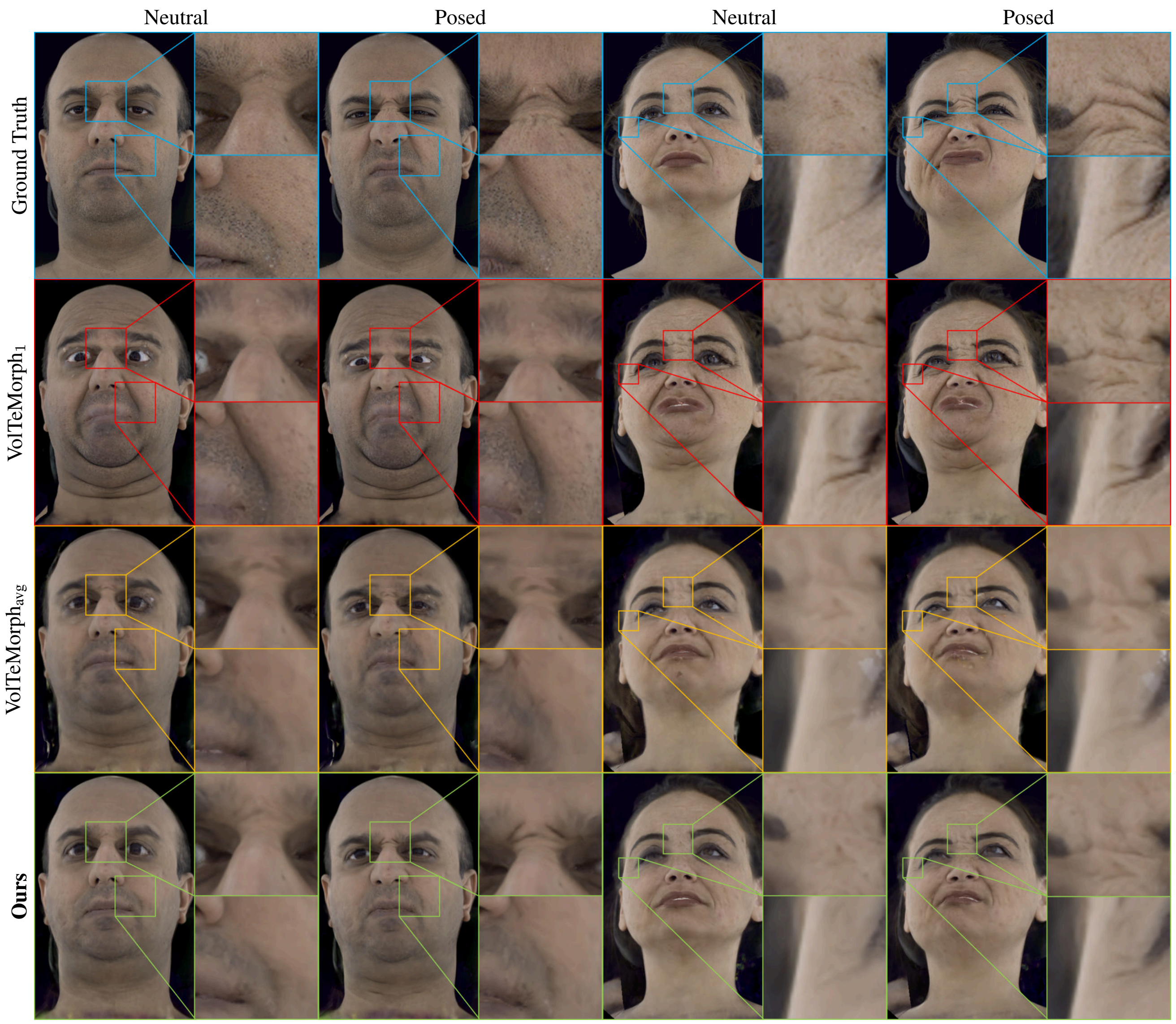


Figure 5. **Novel expression synthesis** – We compare qualitatively BlendFields with selected baselines (vertical) across two selected subjects (horizontal). Firstly, we show a neutral pose of the subject and then any of the available expressions. To our surprise, VolTeMorph$_{\text{avg}}$ trained on multiple frames renders some details but with much lower fidelity. We argue that VolTeMorph$_{\text{arg}}$ considers rendering wrinkles as artifacts that depend on the view direction (see Equation (1)). VolTeMorph$_1$ is limited to producing the wrinkles it was trained for. In contrast to those baselines, **BlendFields** captures the details and generalizes outside of the distribution. Please refer to the `Supplementary` for animated sequences and results for other methods.

lead to averaging of the output colors, we split it into two regimes: one trained on the most extreme expression[4] (VolTeMorph$_1$) and the another trained on all available expressions (VolTeMorph$_{\text{avg}}$)[5]. We use both of these baselines as VolTeMorph was originally designed for a single-frame scenario. By using two versions, we show that it is not trivial to extend it to multiple expressions.

[4]We manually select one frame that has the most visible wrinkles.

[5]We do not compare to NeRFace [13] and NHA [17] as VolTeMorph [15] performs better quantitatively than these methods.

4.1. Realistic Human Captures

Novel expression synthesis. We extract eight multi-view frames from the Multiface dataset [66], each of a different expression. Five of these expressions serve as training data, and the rest are used for evaluation. After training, we can extrapolate from the training expressions by modifying the expression vector $\mathbf{e}$. We use the remaining three expressions: moving mouth left and right, and puffing cheeks, to evaluate the capability of the models to reconstruct other expressions. In Fig. 5 we show that BlendFields is the only

Method	Real Data						Synthetic Data		
	Casual Expressions			Novel Pose Synthesis			Novel Pose Synthesis		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
NeRF [36]	23.6465	0.7384	0.2209	25.6696	0.8127	0.1861	13.7210	0.6868	0.3113
Conditioned NeRF [36]	22.9106	0.7162	0.2029	24.7283	0.7927	0.1682	19.5971	0.8138	0.1545
NeRFies [42]	22.6571	0.7105	0.2271	24.8376	0.7990	0.1884	19.3042	0.8081	0.1591
HyperNeRF-AP [43]	22.6219	0.7087	0.2236	24.7119	0.7931	0.1848	19.3557	0.8132	0.1563
HyperNeRF-DS [43]	22.9299	0.7182	0.2241	24.9909	0.8007	0.1860	19.4637	0.8159	0.1526
VolTeMorph$_1$ [15]	24.9939	0.8358	0.1164	26.7526	0.8749	0.0954	26.7033	0.9500	0.0394
VolTeMorph$_{\text{avg}}$ [15]	26.9209	0.8912	0.1105	28.6866	0.9176	0.0982	30.2107	0.9815	0.0387
BlendFields	27.5977	0.9056	0.0854	29.7372	0.9311	0.0782	32.2900	0.9882	0.0231

Table 2. **Quantitative results** – We compare BlendFields to other related approaches. We split the real data into two settings: one with casual expressions of subjects and the other with novel, static expressions. For the real data, we only compute metrics on the face region, which we separate using an off-the-shelf face segmentation network [62]. Please refer to the `Supplementary` for the results that include the background in the metrics as well. We average results across frames and subjects. VolTeMorph$_{\text{avg}}$ [15] is trained on all frames, while VolTeMorph$_1$ is trained on a single frame. HyperNeRF-AP/-DS follows the design principles from Park *et al.* [43]. The best results are colored in 🟥 and second best results in 🟨. BlendFields performs best in most of the datasets and metrics. Please note that HyperNeRF-AP/DS and NeRFies predict a dense deformation field designed for dense data. However, our input data consists of a few static frames only where the deformation field leads to severe overfitting.

Parameter	Real Data						Synthetic Data		
	Casual Expressions			Novel Pose Synthesis			Novel Pose Synthesis		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
$\lvert\mathcal{N}(\mathbf{v})\rvert = 1$	27.5620	0.9043	0.0893	29.7269	0.9306	0.0815	32.2371	0.9882	0.0234
$\lvert\mathcal{N}(\mathbf{v})\rvert = 5$	27.5880	0.9054	0.0864	29.7548	0.9312	0.0789	32.2900	0.9882	0.0231
$\lvert\mathcal{N}(\mathbf{v})\rvert = 10$	27.5933	0.9054	0.0859	29.7456	0.9312	0.0785	32.3324	0.9882	0.0230
$\lvert\mathcal{N}(\mathbf{v})\rvert = 20$	27.5977	0.9056	0.0854	29.7372	0.9311	0.0782	32.7949	0.9887	0.0221
Without smoothing	27.2535	0.8959	0.0939	29.3726	0.9233	0.0846	32.2452	0.9876	0.0238
With smoothing	27.5977	0.9056	0.0854	29.7372	0.9311	0.0782	32.6323	0.9887	0.0223

Table 3. **Ablation study** – First, we check the effect of the neighborhood size $|\mathcal{N}(\mathbf{v})|$ on the results. Below that, we compare the effect of smoothing. The best results are colored in 🟥 and the second best in 🟨. For the real dataset, changing the neighborhood size gives inconsistent results, while smoothing improves the rendering quality. In the synthetic scenario, setting $|\mathcal{N}(\mathbf{v})|$=20 and the Laplacian smoothing consistently gives the best results. The discrepancy between real and synthetic datasets is caused by inaccurate face tracking for the former. We describe this issue in detail in Section 4.4.

method capable of rendering convincing wrinkles dynamically, depending on the input expression. BlendFields performs favorably compared to the baselines (see Tab. 2).

Casual expressions. The Multiface dataset contains sequences where the subject follows a script of expressions to show during the capture. Each of these captures contains between 1000 and 2000 frames. This experiment tests whether a model can interpolate between the training expressions smoothly and generalize beyond the training data. Quantitative results are shown in Tab. 2. Our approach performs best all the settings. See animations in the `Supplementary` for a comparison of rendered frames across all methods.

4.2. Modeling Objects Beyond Faces

We show that our method can be applied beyond face modeling. We prepare two datasets containing 96 views per frame of bending and twisting cylinders made of a rubber-like material (24 and 72 temporal frames, respectively). When bent or twisted, the cylinders reveal pose-dependent details. The expression vector $\mathbf{e}$ now encodes time: 0 if the cylinder is in the canonical pose, 1 if it is posed, and any values between $[0, 1]$ for the transitioning stage. We select expressions $\{0, 0.5, 1.0\}$ as a training set (for VolTeMorph$_1$ we use 1.0 only). For evaluation, we take every fourth frame from the full sequence using cameras from the bottom and both sides of the object. We take the

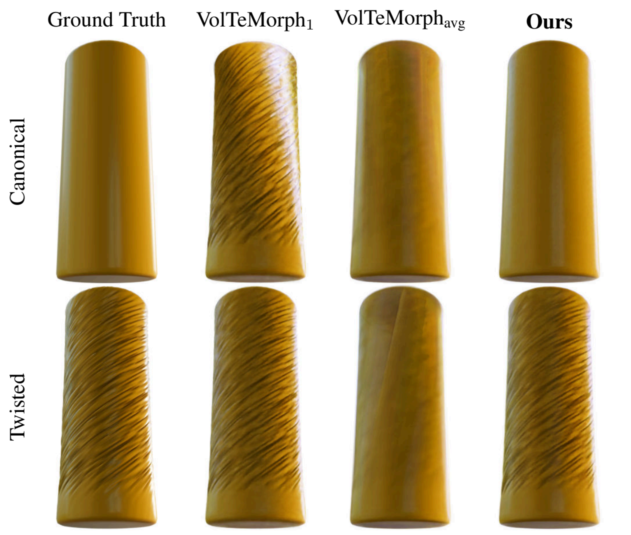


Figure 6. **Qualitative results on synthetic dataset** – For a simple dataset, baselines cannot model high-frequency, pose-dependent details. VolTeMorph$_1$ renders wrinkles for the straight pose as well, as it is trained for the twisted cylinder only, while VolTeMorph$_{avg}$ averages out the texture.

mesh directly from Houdini [69], which we use for wrinkle simulation, and render the images in Blender [10]. We show quantitative results in Tab. 2 for the bending cylinder, and a comparison of the inferred images in Fig. 6 for the twisted one[6]. BlendFields accurately captures the transition from the rest configuration to the deformed state of the cylinder, rendering high-frequency details where required. All other approaches struggle with interpolation between states. VolTeMorph$_1$ (trained on a single extreme pose) renders wrinkles even when the cylinder is not twisted.

4.3. Ablations

We check how the neighborhood size $|\mathcal{N}(\mathbf{v})|$ and the application of the smoothing influence the performance of our method. We show the results in Tab. 3. BlendFields works best in most cases when considering a relatively wide neighborhood for the tetrahedral features[7]. Laplacian smoothing consistently improves the quality across all the datasets (see Fig. 4). We additionally present in the `Supplementary` how the number of expressions used for training affects the results.

4.4. Failure Cases

While BlendFields offers significant advantages for rendering realistic and dynamic high-frequency details, it falls short in some scenarios (see Fig. 7). One of the issues arises

[6]Our motivation is that it is easier to show pose-dependent deformations on twisting as it affects the object globally, while the bending cannot be modeled by all the baselines due to the non-stationary effects.

[7]Larger neighborhood sizes caused out-of-memory errors on our NVIDIA 2080Ti GPU.

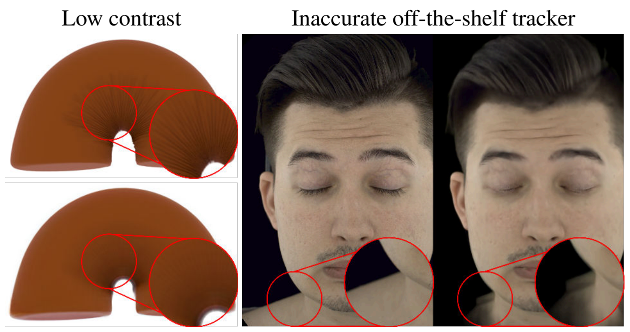


Figure 7. **Failure cases** – We show failure cases for our proposed approach. *Left:* In the presence of wrinkles in low-contrast images, BlendFields takes longer to converge to make wrinkles visible. We show the ground truth on the top, and rendering after training 7×10^5 steps on the bottom. In contrast, we rendered images in Figure 6 after 2×10^5 steps. *Right:* BlendFields inherits issues from VolTeMorph [15], which relies on the initial fit of the face mesh. If the fit is inaccurate, artifacts appear in the final render.

when the contrast between wrinkles and the subject's skin color is low. In those instances, we observe a much longer time to convergence. Moreover, as we build BlendFields on VolTeMorph, we also inherit some of its problems. Namely, the method heavily relies on the initial fit of the parametric model – any inaccuracy leads to ghosting artifacts or details on the face that jump between frames.

5. Conclusions

We present a general approach, BlendFields, for rendering high-frequency expression-dependent details using NeRFs. BlendFields draws inspiration from classical computer graphics by blending expressions from the training data to render expressions unseen during training. We show that BlendFields renders images in a controllable and interpretable manner for novel expressions and can be applied to render human avatars learned from publicly available datasets. We additionally discuss the potential misuse of our work in the `Supplementary`.

6. Acknowledgements

The work was partly supported by the National Sciences and Engineering Research Council of Canada (NSERC), the Digital Research Alliance of Canada, and Microsoft Mesh Labs. This research was funded by Microsoft Research through the EMEA PhD Scholarship Programme. We thank NVIDIA Corporation for granting us access to GPUs through NVIDIA's Academic Hardware Grants Program. This research was partially funded by National Science Centre, Poland (grant no 2020/39/B/ST6/01511 and 2022/45/B/ST6/02817).

References

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

[2] Benjamin Attal, Eliot Laidlaw, Aaron Gokaslan, Changil Kim, Christian Richardt, James Tompkin, and Matthew O'Toole. Törf: Time-of-flight radiance fields for dynamic scene view synthesis. *Advances in Neural Information Processing Systems*, 34:26289–26301, 2021. 2

[3] Ismail Avcibas, Bulent Sankur, and Khalid Sayood. Statistical evaluation of image quality measures. *Journal of Electronic Imaging*, 11(2):206 – 223, 2002. 5

[4] Jonathan Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul Srinivasan. Mip-NeRF: A Multiscale Representation for Anti-aliasing Neural Radiance Fields. In *International Conference on Computer Vision*, pages 5855–5864, 2021. 2

[5] Jonathan Barron, Ben Mildenhall, Dor Verbin, Pratul Srinivasan, and Peter Hedman. Mip-NeRF 360: Unbounded Anti-aliased Neural Radiance Fields. In *Conference on Computer Vision and Pattern Recognition*, pages 5470–5479, 2022. 2

[6] Volker Blanz and Thomas Vetter. A Morphable Model For The Synthesis Of 3D Faces. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 187–194, 1999. 2, 4

[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, et al. Authentic Volumetric Avatars from a Phone Scan. *ACM Transactions on Graphics (TOG)*, 41(4):1–19, 2022. 2, 3, 5

[8] Zezhou Cheng, Menglei Chai, Jian Ren, Hsin-Ying Lee, Kyle Olszewski, Zeng Huang, Subhransu Maji, and Sergey Tulyakov. Cross-Modal 3D Shape Generation and Manipulation. *arXiv preprint arXiv:2207.11795*, 2022. 2

[9] Ray William Clough. The Finite Element Method in Plane Stress Analysis. In *Conference on Electronic Computation*, 1960. 3

[10] Blender Online Community. *Blender - a 3D modeling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2022. 8

[11] Shane Cook. *CUDA Programming: A Developer's Guide to Parallel Computing with GPUs*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1st edition, 2012. 3

[12] Mathieu Desbrun, Mark Meyer, Peter Schröder, and Alan H Barr. Implicit Fairing of Irregular Meshes using Diffusion and Curvature Flow. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 317–324, 1999. 5

[13] Guy Gafni, Justus Thies, Michael Zollhofer, and Matthias Nießner. Dynamic Neural Radiance Fields for Monocular 4d Facial Avatar Reconstruction. In *Conference on Computer Vision and Pattern Recognition*, pages 8649–8658, 2021. 2, 3, 6

[14] Xuan Gao, Chenglai Zhong, Jun Xiang, Yang Hong, Yudong Guo, and Juyong Zhang. Reconstructing Personalized Semantic Facial NeRF Models From Monocular Video. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)*, 41(6), 2022. 2, 3

[15] Stephan J Garbin, Marek Kowalski, Virginia Estellers, Stanislaw Szymanowicz, Shideh Rezaeifar, Jingjing Shen, Matthew Johnson, and Julien Valentin. VolTeMorph: Real-time, Controllable and Generalisable Animation of Volumetric Representations. *arXiv preprint arXiv:2208.00949*, 2022. 1, 2, 3, 4, 5, 6, 7, 8

[16] Stephan J Garbin, Marek Kowalski, Matthew Johnson, Jamie Shotton, and Julien Valentin. FastNeRF: High-Fidelity Neural Rendering at 200FPS. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14346–14355, 2021. 2, 3

[17] Philip-William Grassal, Malte Prinzler, Titus Leistner, Carsten Rother, Matthias Nießner, and Justus Thies. Neural Head Avatars from Monocular RGB Videos. In *Conference on Computer Vision and Pattern Recognition*, pages 18653–18664, 2022. 2, 6

[18] Peter Hedman, Pratul P Srinivasan, Ben Mildenhall, Jonathan T Barron, and Paul Debevec. Baking Neural Radiance Fields for Real-Time View Synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5875–5884, 2021. 2, 3, 5

[19] Xin Huang, Qi Zhang, Ying Feng, Hongdong Li, Xuan Wang, and Qing Wang. HDR-NeRF: High Dynamic Range Neural Radiance Fields. In *Conference on Computer Vision and Pattern Recognition*, pages 18398–18408, 2022. 2

[20] Alexandru Eugen Ichim, Sofien Bouaziz, and Mark Pauly. Dynamic 3D Avatar Creation from Hand-Held Video Input. *ACM Trans. Graph.*, 34(4), jul 2015. 4

[21] Geoffrey Irving, Joseph Teran, and Ronald Fedkiw. Invertible Finite Elements For Robust Simulation of Large Deformation. In *Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 131–140, 2004. 2, 3, 4, 5

[22] Wei Jiang, Kwang Moo Yi, Golnoosh Samei, Oncel Tuzel, and Anurag Ranjan. Neuman: Neural Human Radiance Field from a Single Video. *arXiv preprint arXiv:2203.12575*, 2022. 2

[23] Kacper Kania, Kwang Moo Yi, Marek Kowalski, Tomasz Trzciński, and Andrea Tagliasacchi. CoNeRF: Controllable Neural Radiance Fields. In *Conference on Computer Vision and Pattern Recognition*, 2022. 1, 2

[24] Mira Kim, Jaehoon Ko, Kyusun Cho, Junmyeong Choi, Daewon Choi, and Seungryong Kim. AE-NeRF: Auto-Encoding Neural Radiance Fields for 3D-Aware Object Manipulation. *arXiv preprint arXiv:2204.13426*, 2022. 2

[25] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 5

[26] J. P. Lewis, Ken Anjyo, Taehyun Rhee, Mengjie Zhang, Fred Pighin, and Zhigang Deng. Practice and Theory of Blendshape Facial Models. In Sylvain Lefebvre and Michela Spagnuolo, editors, *Eurographics 2014 - State of the Art Reports*. The Eurographics Association, 2014. 2

[27] Tianye Li, Timo Bolkart, Michael. J. Black, Hao Li, and Javier Romero. Learning a model of facial shape and expression from 4D scans. *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 36(6):194:1–194:17, 2017. 2, 4, 3

[28] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural Sparse Voxel Fields. *Advances in Neural Information Processing Systems*, 33:15651–15663, 2020. 3

[29] Steven Liu, Xiuming Zhang, Zhoutong Zhang, Richard Zhang, Jun-Yan Zhu, and Bryan Russell. Editing Conditional Radiance Fields. In *International Conference on Computer Vision*, pages 5773–5783, 2021. 1

[30] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural Volumes: Learning Dynamic Renderable Volumes from Images. *ACM Trans. Graph.*, 38(4), jul 2019. 3

[31] Stephen Lombardi, Tomas Simon, Gabriel Schwartz, Michael Zollhoefer, Yaser Sheikh, and Jason Saragih. Mixture of Volumetric Primitives for Efficient Neural Rendering. *ACM Transactions on Graphics (ToG)*, 40(4):1–13, 2021. 3

[32] Shugao Ma, Tomas Simon, Jason Saragih, Dawei Wang, Yuecheng Li, Fernando De la Torre, and Yaser Sheikh. Pixel Codec Avatars. In *Conference on Computer Vision and Pattern Recognition*, pages 64–73, June 2021. 2, 3

[33] Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7210–7219, 2021. 2

[34] Marko Mihajlovic, Aayush Bansal, Michael Zollhoefer, Siyu Tang, and Shunsuke Saito. KeypointNeRF: Generalizing Image-based Volumetric Avatars using Relative Spatial Encoding of Keypoints. In *European Conference on Computer Vision*, 2022. 3

[35] Ben Mildenhall, Peter Hedman, Ricardo Martin-Brualla, Pratul Srinivasan, and Jonathan Barron. NeRF in the Dark: High Dynamic Range View Synthesis from Noisy Raw Images. In *Conference on Computer Vision and Pattern Recognition*, pages 16190–16199, 2022. 2

[36] 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. In *ECCV*, 2020. 1, 2, 5, 7, 3

[37] Neil Molino, Robert Bridson, and Ronald Fedkiw. Tetrahedral Mesh Generation for Deformable Bodies. In *ACM Symp. on Comput. Animation*, 2003. 3

[38] Peter Monk. Finite Elements on Tetrahedra. In *Finite Element Methods for Maxwell's Equations*. Oxford University Press, 04 2003. 4

[39] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant Neural Graphics Primitives with a Multiresolution Hash Encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, July 2022. 2

[40] Atsuhiro Noguchi, Umar Iqbal, Jonathan Tremblay, Tatsuya Harada, and Orazio Gallo. Watch It Move: Unsupervised Discovery of 3D Joints for Re-Posing of Articulated Objects. In *Conference on Computer Vision and Pattern Recognition*, pages 3677–3687, 2022. 2

[41] Christopher Oat. Animated Wrinkle Maps. In *ACM SIGGRAPH*, SIGGRAPH '07, page 33–37, New York, NY, USA, 2007. Association for Computing Machinery. 4

[42] 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, 3, 5, 7

[43] 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), dec 2021. 2, 3, 5, 7

[44] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-NeRF: Neural Radiance Fields for Dynamic Scenes. In *Conference on Computer Vision and Pattern Recognition*, pages 10318–10327, 2021. 3

[45] Chirag Raman, Charlie Hewitt, Erroll Wood, and Tadas Baltrusaitis. Mesh-Tension Driven Expression-Based Wrinkles for Synthetic Faces. In *WACV Workshop on Applications of Computer Vision*, 2023. 2

[46] Konstantinos Rematas, Andrew Liu, Pratul Srinivasan, Jonathan Barron, Andrea Tagliasacchi, Thomas Funkhouser, and Vittorio Ferrari. Urban Radiance Fields. In *Conference on Computer Vision and Pattern Recognition*, pages 12932–12942, 2022. 2

[47] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-Resolution Image Synthesis with Latent Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022. 2

[48] Mohammed Suhail, Carlos Esteves, Leonid Sigal, and Ameesh Makadia. Light Field Neural Rendering. In *Conference on Computer Vision and Pattern Recognition*, pages 8269–8279, 2022. 2

[49] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct Voxel Grid Optimization: Super-fast Convergence for Radiance Fields Reconstruction. In *Conference on Computer Vision and Pattern Recognition*, pages 5459–5469, 2022. 3

[50] Jingxiang Sun, Xuan Wang, Yong Zhang, Xiaoyu Li, Qi Zhang, Yebin Liu, and Jue Wang. FENeRF: Face Editing in Neural Radiance Fields. In *Conference on Computer Vision and Pattern Recognition*, pages 7672–7682, 2022. 2

[51] Andrea Tagliasacchi and Ben Mildenhall. Volume Rendering Digest (for NeRF). *arXiv preprint arXiv:2209.02417*, 2022. 3

[52] Matthew Tancik, Vincent Casser, Xinchen Yan, Sabeek Pradhan, Ben Mildenhall, Pratul Srinivasan, Jonathan Barron, and Henrik Kretzschmar. Block-NeRF: Scalable Large Scene Neural View Synthesis. In *Conference on Computer Vision and Pattern Recognition*, pages 8248–8258, 2022. 2

[53] Ayush Tewari, Justus Thies, Ben Mildenhall, Pratul Srinivasan, Edgar Tretschk, W Yifan, Christoph Lassner, Vincent Sitzmann, Ricardo Martin-Brualla, Stephen Lombardi, et al. Advances in Neural Rendering. In *Computer Graphics Forum*, 2022. 1, 2

[54] Edgar Tretschk, Ayush Tewari, Vladislav Golyanik, Michael Zollhöfer, Christoph Lassner, and Christian Theobalt. Non-Rigid Neural Radiance Fields: Reconstruction and Novel View Synthesis of a Dynamic Scene From Monocular Video. In *International Conference on Computer Vision*, pages 12959–12970. IEEE, 2021. 3

[55] Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan Barron, and Pratul Srinivasan. Ref-NeRF: Structured View-dependent Appearance for Neural Radiance Fields. In *Conference on Computer Vision and Pattern Recognition*, pages 5481–5490, 2022. 2

[56] Delio Vicini, Wenzel Jakob, and Anton Kaplanyan. Non-exponential transmittance model for volumetric scene representations. *ACM Trans. Graph.*, 40(4), jul 2021. 2

[57] Can Wang, Menglei Chai, Mingming He, Dongdong Chen, and Jing Liao. CLIP-NeRF: Text-and-Image Driven Manipulation of Neural Radiance Fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3835–3844, 2022. 2

[58] Daoye Wang, Prashanth Chandran, Gaspard Zoss, Derek Bradley, and Paulo Gotardo. MoRF: Morphable Radiance Fields for Multiview Neural Head Modeling. In *ACM SIGGRAPH 2022 Conference Proceedings*, SIGGRAPH '22, New York, NY, USA, 2022. Association for Computing Machinery. 3

[59] Liao Wang, Jiakai Zhang, Xinhang Liu, Fuqiang Zhao, Yanshun Zhang, Yingliang Zhang, Minye Wu, Jingyi Yu, and Lan Xu. Fourier PlenOctrees for Dynamic Radiance Field Rendering in Real-time. In *Conference on Computer Vision and Pattern Recognition*, pages 13524–13534, 2022. 2

[60] Zhou Wang, Eero P Simoncelli, and Alan C Bovik. Multi-scale Structural Similarity for Image Quality Assessment. In *Conference on Signals, Systems & Computers*, 2003. 5

[61] Chung-Yi Weng, Brian Curless, Pratul Srinivasan, Jonathan Barron, and Ira Kemelmacher-Shlizerman. HumanNeRF: Free-viewpoint Rendering of Moving People from Monocular Video. In *Conference on Computer Vision and Pattern Recognition*, pages 16210–16220, 2022. 2

[62] Erroll Wood, Tadas Baltrušaitis, Charlie Hewitt, Sebastian Dziadzio, Thomas J Cashman, and Jamie Shotton. Fake It Till You Make It: Face analysis in the wild using synthetic data alone. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3681–3691, 2021. 7

[63] Erroll Wood, Tadas Baltrusaitis, Charlie Hewitt, Matthew Johnson, Jingjing Shen, Nikola Milosavljevic, Daniel Wilde, Stephan Garbin, Chirag Raman, Jamie Shotton, Toby Sharp, Ivan Stojiljkovic, Tom Cashman, and Julien Valentin. 3D Face Reconstruction with Dense Landmarks, 2022. 3

[64] Erroll Wood, Tadas Baltrusaitis, Charlie Hewitt, Matthew Johnson, Jingjing Shen, Nikola Milosavljevic, Daniel Wilde, Stephan Garbin, Toby Sharp, Ivan Stojiljkovic, Tom Cashman, and Julien Valentin. 3d face reconstruction with dense landmarks, 2022. 5

[65] Erroll Wood, Tadas Baltrušaitis, Charlie Hewitt, Sebastian Dziadzio, Thomas J Cashman, and Jamie Shotton. Fake It Till You Make It: face analysis in the wild using synthetic data alone. In *International Conference on Computer Vision*, pages 3681–3691, 2021. 4, 5

[66] Cheng-hsin Wuu, Ningyuan Zheng, Scott Ardisson, Rohan Bali, Danielle Belko, Eric Brockmeyer, Lucas Evans, Timothy Godisart, Hyowon Ha, 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. 5, 6, 2, 4

[67] Wenqi Xian, Jia-Bin Huang, Johannes Kopf, and Changil Kim. Space-time Neural Irradiance Fields for Free-viewpoint Video. In *Conference on Computer Vision and Pattern Recognition*, pages 9421–9431, 2021. 2

[68] Yuanbo Xiangli, Linning Xu, Xingang Pan, Nanxuan Zhao, Anyi Rao, Christian Theobalt, Bo Dai, and Dahua Lin. CityNeRF: Building NeRF at City Scale. *arXiv preprint arXiv:2112.05504*, 2021. 2

[69] Ken Xu and Damian Campeanuy. Houdini engine: Evolution towards a procedural pipeline. In *Proceedings of the Fourth Symposium on Digital Production*, pages 13–18, 2014. 8

[70] Tianhan Xu and Tatsuya Harada. Deforming Radiance Fields with Cages. In *European Conference on Computer Vision*, 2022. 2, 3

[71] Yuelang Xu, Lizhen Wang, Xiaochen Zhao, Hongwen Zhang, and Yebin Liu. ManVatar: Fast 3D Head Avatar Reconstruction Using Motion-Aware Neural Voxels. *arXiv preprint arXiv:2211.13206*, 2022. 2

[72] Bangbang Yang, Chong Bao, Junyi Zeng, Hujun Bao, Yinda Zhang, Zhaopeng Cui, and Guofeng Zhang. NeuMesh: Learning Disentangled Neural Mesh-based Implicit Field for Geometry and Texture Editing. In *European Conference on Computer Vision*, pages 597–614. Springer, 2022. 2, 3

[73] Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. PlenOctrees for Real-time Rendering of Neural Radiance Fields. In *International Conference on Computer Vision*, pages 5752–5761, 2021. 2, 3

[74] 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 *Conference on Computer Vision and Pattern Recognition*, pages 18353–18364, 2022. 2, 3

[75] Jianfeng Zhang, Zihang Jiang, Dingdong Yang, Hongyi Xu, Yichun Shi, Guoxian Song, Zhongcong Xu, Xinchao Wang, and Jiashi Feng. AvatarGen: A 3D Generative Model for Animatable Human Avatars. *arXiv preprint arXiv:2208.00561*, 2022. 2

[76] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. NeRF++: Analyzing and Improving Neural Radiance Fields. *arXiv:2010.07492*, 2020. 2

[77] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. In *Conference on Computer Vision and Pattern Recognition*, 2018. 5

[78] Fuqiang Zhao, Wei Yang, Jiakai Zhang, Pei Lin, Yingliang Zhang, Jingyi Yu, and Lan Xu. HumanNeRF: Efficiently Generated Human Radiance Field from Sparse Inputs. In

Conference on Computer Vision and Pattern Recognition, pages 7743–7753, 2022. 2

[79] Yihao Zhi, Shenhan Qian, Xinhao Yan, and Shenghua Gao. Dual-Space NeRF: Learning Animatable Avatars and Scene Lighting in Separate Spaces. *arXiv preprint arXiv:2208.14851*, 2022. 2

[80] Yiyu Zhuang, Hao Zhu, Xusen Sun, and Xun Cao. MoFaNeRF: Morphable Facial Neural Radiance Field. *arXiv preprint arXiv:2112.02308*, 2021. 3

[81] Wojciech Zielonka, Timo Bolkart, and Justus Thies. Instant Volumetric Head Avatars, 2022. 2