Warsaw University of Technology

FACULTY OF ELECTRONICS AND INFORMATION TECHNOLOGY

PhD Thesis

in the discipline of Information and Communication Technology

Few-Shot Human Neural Rendering with Partial Information

Kacper Kania, M.Sc.

supervisor
Tomasz Trzciński, Prof. PhD DSc.

assistant supervisor
Marek Kowalski, PhD DSc.

WARSZAWA 2025

Acknowledgements

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Abstract

This thesis is a series of publications that introduce novel methods for human neural rendering using limited information, focusing on Neural Radiance Fields (NeRFs) and 3D Gaussian Splatting (3DGS). It explores how these models construct 3D representations from 2D images and demonstrates ways to condition these representations for generating high-quality human renderings. We propose techniques that use simple, interpretable inputs derived from sparse training data and extends these methods to perform effectively in few-shot learning scenarios.

We begin by examining the field of neural radiance fields, addressing limitations in existing approaches and presenting contributions to controllable radiance fields. By incorporating partial and sparse data during training, it leverages the smoothness of neural networks to produce controllable, high-quality human images.

To tackle the reliance on extensive, high-quality data annotations from multi-view videos, we introduce a new method for training neural radiance fields in few-shot, multi-view settings. This approach learns internal deformation templates, which blend smoothly during inference, significantly improving image quality compared to existing baselines and enabling effective human rendering from limited input images.

The work also addresses the need for adaptable computational efficiency during inference. It proposes a fine-to-coarse learning strategy for 3D Gaussian Splatting, which upscales a latent 2D grid that stores Gaussian representations. This strategy achieves competitive results while allowing deployment on various computational devices with minimal quality loss.

In addition, we develop a novel model for controlling radiance fields through environmental lighting. By incorporating precomputed radiance transfer, this model enables physically plausible scene relighting and provides users with intuitive control over lighting in reconstructed scenes.

This research advances the state of the art in controllable neural radiance fields and expands their application to few-shot learning scenarios. These innovations enhance the possibilities for human rendering from limited information and open new directions for future research in the field.

Keywords: Neural Rendering, Neural Radiance Fields, Few-Shot Learning, Human Rendering, Partial Information, Gaussian Splatting

Streszczenie

To jest streszczenie. To jest trochę za krótkie, jako że powinno zająć całą stronę.

Słowa kluczowe: A, B, C

Lay Summary

ok

Publications in this thesis

Title	Authors	Venue	Status
CoNeRF: Controllable Neural Radiance Fields	**Kacper Kania**, Kwang Moo Yi, Marek Kowalski, Tomasz Trzciński, Andrea Tagliasacchi	CVPR 2022	Accepted
BlendFields: Few-Shot Example-Driven Facial Modeling	**Kacper Kania**, Stephan J. Garbin, Andrea Taglisacchi, Virginia Estellers, Kwang Moo Yi, Julien Valentin, Tomasz Trzciński, Marek Kowalski	CVPR 2023	Accepted
LumiGauss: High-Fidelity Outdoor Relighting with 2D Gaussian Splatting	Joanna Kaleta, **Kacper Kania**, Tomasz Trzciński, Marek Kowalski	WACV 2025	Accepted
CLoG: Leveraging UV Space for Continuous Levels of Detail	**Kacper Kania**, Rawal Khirodkar, Shunsuke Saito, Kwang Moo Yi, Julieta Martinez	CVPR 2025	Under Review

Contents

List of Abbreviations and Symbols

List of Figures

List of Tables

π Stała matematyczna równa stosunkowi obwodu okręgu do jego średnicy

I Natężenie prądu elektrycznego

Chapter 1

Introduction

With the advent of deep learning, research have been exploring varying ways to apply it to computer graphics. One of the most recent and promising approaches is neural rendering. Neural rendering is a field that combines deep learning and computer graphics to generate realistic images of 3D scenes. The neural radiance field (NeRF) is a popular neural rendering technique that represents a 3D scene as a continuous function that maps 3D coordinates to radiance values. NeRF has shown impressive results in generating photorealistic images of 3D scenes. However, NeRF has limitations in terms of memory and computational requirements, which makes it difficult to scale to large scenes.

To alleviate the problem, Kerbl *et al.* [52] proposed a new technique—3D Gaussian Splatting (3DGS). 3DGS is a neural rendering technique that represents a 3D scene as a set of 3D Gaussian that are splatted to an image space using algorithm proposed by Zwicker *et al.* [152]. In contrast to NeRF, 3DGS is more memory efficient and can be used to render large scenes. It can also render scenes with millions of points in real-time on a single GPU.

In this thesis, we focus on those two milestone techniques in neural rendering and address their fundamental problem—lack of controllability.

1.1 Motivation and challenges

NeRF and 3DGS are both impressive techniques that can generate realistic images. However, a single scene representation needs to be trained on a high-end GPU for hours or even days just to render a novel view at the inference time. However, any type of controllability is difficult to achieve with those models. That includes changing the lighting conditions, subject's attributes or even the scene itself. We see imbuing those models with controllability as a an important step towards making them more useful in practice. Our proposed models are designed to address this issue.

One may ask why the controllability is a feat sought after to be researched. We see the inspiration in how human artists work. Imagine an artist working on 3D game where they

need an asset, like a 3D mesh, to be created. Such a mesh takes much effort since it includes modeling, creating a UV map which can then be textured. After the process is finish, the artist's supervisor may task him to change the model to some extent which requires the artist to redo all the effort again. Such a process is not limited to 3D assets as meshes and could be applied to 3DGS or NeRF. However, 3DGS and NeRFs are volumetric in nature. Our exploited and well-established practices no longer apply to them since volumetric representations do not have the underlying surface representation. For that reason, we see a couple of avenues which we explore in this thesis.

Firstly, Park *et al.* [83] proposed NeRFies, a model that creates a volumetric representation of a person from a self-captured sequence with a phone camera. Since the inception of NeRFs [72], it was among the first works the achieved such a high quality of reconstructions from a casual videos. In its primal form, NeRFies were unable to control the avatar in any other way than by a linear interpolation of latent embeddings that embedded the video's time dimension. The follow-up work, HyperNeRF [84] handles this issue by projecting the learnable embeddings with D onto a lower-dimensional space $\mathbb{R}^d$ where $d \ll D$. After the assumption that the $d=2$ is enough to explain the sequence variability, that projected embedding becomes a 2-dimensional space that can be traversed in an interpretable way. However, that space is not intuitive since the projection is a non-linear operation and one cannot predict how values affect the results. To mitigate that issue, we propose to leverage smoothness of Multilayer Perceptrons (MLPs) [83, 106] to constrain the projection via sparse supervision. We realize our approach as a weakly-supervised MLP that out of many images from the sequence (we assume at least 100 frames in our work) only a few are provided with a coarse annotation. Such annotations denote what values a chosen attribute takes and where its effect spans in the image space. We show that our method, which we dubbed CoNeRF [50] and published at the CVPR 2022 conference, imbues NeRFs with a flexible editability feature without the lose of the rendering quality.

Secondly, approaches such as CoNeRF [50], EditNeRF [62] or FigNeRF [128] focus solely on static elements of the scene, hence their controllability is limited to changing colors or textures in general. HyperNeRF [84] arises as a potential solution due to its ability to model object deformations. However, our initial experiments showed that those changes cannot handle motions that affect a subject globally, *e.g.*, jumping jacks performed by a person. To solve the issue, Fang *et al.* [24] proposes to model the deformation via a multi-scale voxel structure which works well in the synthetic setting, such as the one proposed by Pumarola *et al.* [86].

There exists a plethora of works that approach the problem from the another angle—instead of modeling the motion purely from data, they use a template model in the form of a 3D mesh to canonicalize deformed points [150]. Such methods rely on the accuracy of the *registration*, *i.e.*, fitting the template mesh to subject. Since the registration methods [25, 149] are imperfect estimators, they inherently contain registration errors. Those deviations are exacerbated by learnable radiance field models which assume a perfectly calibrated scene. The authors of those approaches usually mitigate the issue with additional latent space [30, 50, 68] that requires

thousands of video frames to learn an avatar of high-fidelity that reacts correctly to deformations such as wrinkles on the forehead. At the same time, performing the registration on the large scale is costly [13]. In this thesis, we seek a remedy for those obstacles. We propose a method that is data-efficient, easy to improve with a minimal user input and can model realistic deformation dependent changes in the subject. Inspired by classical methods in character texturing [80] and motion modeling [56], we propose BlendFields [47], an *homage* to traditional blendshapes [56]. We build on VolTeMorph's [28] approach to point canonicalization to provide a data-efficient way to control the character. We further introduce a physically-based mixture of predefined, learned from data wrinkle templates that represent expression-dependent skin deformations. Our proposed was acknowledged by the reviewers and was accepted to the CVPR 2023 conference.

Thirdly, having the texture and coarse mesh-based controllability, we strive for control scene settings directly. The inverse rendering of 3D scenes is an ill-posed problem where many different lighting settings may explain the same light effects [85]. To facilitate solving the problem, many approaches use datasets of single object's images captured under different lighting conditions [15, 93, 133]. These approaches cannot decouple albedo from the lighting effects [15, 133] or need additional neural networks to predict correct shadows [93] which limits methods' practicality. We propose to use recently proposed 2D Gaussian Splatting [39] which exhibit remarkable quality of the surface reconstruction. Together with our precomputed radiance transfer from classical computer graphics approaches [87, 97], our LumiGauss achieves state-of-the-art reconstruction quality with the ability to render novel lighting conditions with high fidelity. Our work received positive reviews for the WACV 2025 conference.

Finally, volumetric representation are computationally intensive to render, compared to the traditional mesh representation. For NeRFs [72], it takes seven days on V100 NVidia GPU to train for single scene, and more than 60 seconds to render a single image—way beyond any practical applications. Although many approaches have been proposed to speed up the rendering process [29, 36, 76, 89, 135], they usually make a trade-off between memory requirements, quality, and rendering speed. 3D Gaussian Splatting [52] (3DGS) rose as an alternative to NeRF, offering both high-quality rendering at interactive frame rates. However, those frame rates could be achieved with the most advanced GPU units available at that time. As we see the potential in 3DGS to be a viable canonical representation for 3D data, akin to 3D meshes, a need for its adaptability to different computational resources exists. Meshes can be adapted easily with levels of detail (LoD) approaches that remove detail from meshes that do not affect the general object's perception if necessary.

1.2 Research objectives

In this thesis, we explore different avenues of radiance field controllability. With this goal in mind, we aim at answering the following research questions:

(**RQ 1**) Can we imbue a Neural Radiance Field (NeRF) with a controllability by providing sparse annotations to the training dataset? How many annotations suffice to learn smooth interpolation capabilities between controlled values?

(**RQ 2**) Are extreme facial expressions known from the literature sufficient to learn expression-dependent details that extrapolate to expressions unseen at the training time?

(**RQ 3**) Is it possible to learn an underlying radiance transfer function of a scene from images taken in "in-the-wild" setting? Can such a transfer function generalize to unknown environment maps?

(**RQ 4**) How to learn a single 3D Gaussian Splatting (3DGS) representation that can be adapted to different computational regimes at inference time in a feed-forward mode?

Each of these questions is a representative of possible among many others controllability directions for radiance fields. In case of this thesis, we present summarize our methods as controls of: texture [47, 50], shape [47], lightning [46] and use of resources [48]. To answer (**RQ 1**), we describe our CoNeRF [50]. It is one of the pioneering works that uses sparsely annotated frames to continuously control the subject in a post-hoc manner. We leverage the fact that MLP used in NeRFs are smooth functions biased towards low frequency signals [106]. For that reason, NeRF can learn to interpolate smoothly the annotation signal between frames of high similarity. We show that this assumption is sufficient to obtain both novel view synthesis and novel attribute synthesis with a single model.

We further move towards answering (**RQ 2**). We introduce BlendFields [47], achieving two primary goals: ability to generalize unknown expressions via a predefined face mesh template, and a mixture model of training expressions that can produce spatially coherent, expression-dependent wrinkles on the face from as few as three expressions. In our work, we build on VolTeMorph [28] to achieve to former, and focus on the latter contribution. Inspired by texture maps in classical computer graphics pipelines [80], we define a set of learnable radiance fields, each being overfit to a particular extreme expression from the training set. We define an extreme expression as one of the possible expressions involving the most facial muscles. Building on VolTeMorph [28] allows us to use an underlying tetrahedral mesh to compute physical quantities such as the volume change of tetrahedra under a given expression. We use those quantities to linearly interpolate between the pretrained radiance fields. We mix the tetrahedra independently which makes rendering novel expressions possible. For example, BlendFields can render one of the eyebrows raised while maintaining the other in a natural position, which is a difficult expression to make for majority of people.

Our LumiGauss [46] answers (**RQ 3**). In contrary to common approaches [93], we posit that the radiance transfer function known among computer graphics researchers [87] can be learned from unconstrained photo collection under varying lighting conditions. To this end, we use 2DGS [39] which gives us a smooth shape representation, difficult to achieve when

using 3DGS [52]. With the use of contributed priors, we induce learning Gaussian's Spherical Harmonics such that they correctly react to changing environment maps. Not only our approach is fast to train, but renders realistic scenes and object's shadows even under novel lighting conditions.

All the contributions so far are affected by a specific disadvantage—a single model needs to be trained from scratch and it can be deployed only on high-end hardware. We then ask if we can train a single model that is adaptable at inference time to different hardware regimes (**RQ 4**). We propose CLoG [48] as a potential remedy. Our approach uses the fact that one can constrain the number of Gaussians in 3DGS [52] to a specific number, such that it can be formed as a 2D grid by simple reshape operation. With a specific training coarse-to-fine training protocol we contributed, the model learns a representation that converges to a high-quality volumetric structure. In the second training stage, we leverage the fact that Gaussians's can be spatially sorted in such a manner that their descriptors are placed next to each other if they are similar. That forms a low-frequency image which can be modulated with an off-the-shelf continuous upsampling architecture [112]. The architecture outputs a new grid of the given resolution. We show in our work that such an approach achieves remarkable results and can output any number of Gaussians at inference time with high quality.

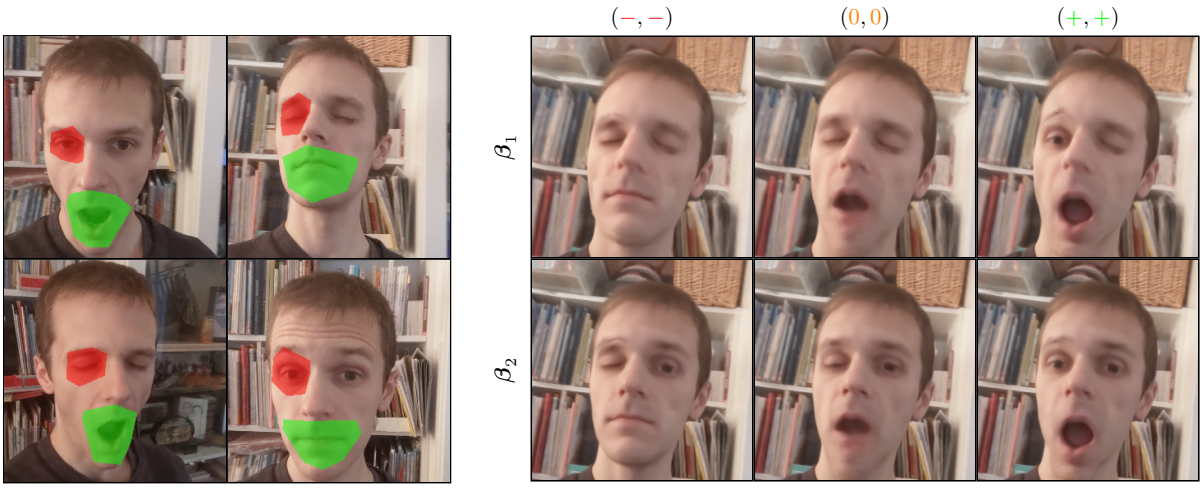
1.3 Contributions

Building on those questions, we structure this thesis in several chapters corresponding to the answers. An answer is in a form of a scientific article where we introduce the following:

- A novel approach for controlling trained radiance fields with the use of sparsely annotated images from a casually captured data.
- A new model capable of blending trained radiance fields from multi-view frames in an interpretable way and extrapolating to novel human expressions for the trained subject.
- The first use of Gaussian Splatting methods that learns a coherent shape representation of a subject and an ability to distill the varying lighting conditions in the data to a radiance transfer function.
- A novel paradigm for learning Gaussian Splatting models as 2D grids to achieve flexibility to adapt the model to different computational regimes at inference time.

1.3.1 Texture from Partial Information

Existing NeRF-based approaches in 2021 were simple models—they were overfit to a single subject, for a new subject the model needed to be retrained from scratch, and the editability



(a) annotated samples **(b)** renderings from the trained model

Figure 1. Example with annotations and controlled attributes – We show an example of how our CoNeRF is capable of controlling attributes selected sparse annotations at the training time. Top row shows possible value combinations (− denoting closed eye, 0 neutral position and + an open eye) and $\{\boldsymbol{\beta}_1, \boldsymbol{\beta}_2\}$ possible values of attributes that are not explicitly learned from the annotations but purely from data (please see Chapter 3 for further explanation).

capabilities were limited [62]. NeRFactor [144] could be considered a more sophisticated model. However, it worked only for simple scene with calibrated scenes and could not handle any motion.

We approach those issues in our CoNeRF [50] published at the CVPR 2021 conference. Inspired by HyperNeRF [84], we propose to revisit the weak supervision in the context of radiance fields. Specifically, under an assumption that one can provide a few of sparse annotations to the dataset, we can leverage the smoothness of the neural networks to propagate the annotation across the dataset. The annotations also consist of what regions in the image space it refers to and hence we can train a semantic segmentation radiance fields that decouples attribute controls. We show an example in Fig. 1 where the left side represents the annotations present in the data and the right one possible manipulations at the inference time. We note that those annotations are easy to make in a matter of a few minutes for a single dataset. However, the complexity of the annotations grows with the number of attributes to control.

1.3.2 Expression from Few-Shot Learning

CoNeRF [50] is capable of rendering complex motions given sufficient amount of data and provided annotations. However, motions as the ones a person performs daily when speaking are infeasible in practice. We propose a solution to target that issue. We introduce BlendFields [47] from the CVPR 2023 proceedings which learns motion-dependent face deformation from data. We build on VolTeMorph [28] which uses existing face template models, such as FLAME [58], to learn a single canonical representation, akin to the canonicalization module in CoNeRF. Internally, BlendFields creates a tetrahedral cage around the face model. For each sample

along the ray in NeRF, it moves the points to a "neutral position", chosen once prior to the training procedure. Such a procedure comes insufficient to model realistic facial features, such as wrinkles. We contribute a novel approach to modeling those deformations. We compute a deformation gradient of each tetrahedra for a given expression which is a physically-based and easy to interpret quantity. The value serves us to smoothly transition face regions textures to appropriate colors. We obtain the colors from NeRFs branches, each overfit to particular expression. In principle, BlendFields predicts face bases which are conditioned on the face expression vector to output the final point color. Our framework works well even when only a few "extreme" expressions are provided such as grinning face with closed eyes and wide open mouth with open eyes.

1.3.3 Light from Unconstrained Images

In both approaches above, we tackle the problem of texture controllability. They assume an ideal case scenario where a capture can be taken in an idealized environment with constant camera exposure and lighting. Moreover, the produced colors are blended together, making the change of light impossible in practice. We then ask the questions if we can decouple an intrinsic color of the subject and change of that color stemming from the environment light. We answer that question with our LumiGauss [46] published at the WACV 2025 conference that, indeed, we can achieve that by learning the radiance transfer function directly from data. For that end, we train a 2DGS [39] model to obtain a smooth and spatially coherent surface of objects. On top of the other attributes known from 3DGS [52], we imbue our Gaussians with additional features corresponding to the radiance transfer, expressed as Spherical Harmonics [32]. We show in our experiments that such a formulation is sufficient to train a model that reacts to changing environment maps in a realistic manner and renders images of higher fidelity than prior approaches.

1.3.4 Levels of Detail in One Model

All the contributions above require considerable computation hardware to be trained on and then run inference in near real-time frame rates. Drawing an inspiration from the gaming industry where Levels of Details (LoD) for meshes is used heavily, we introduce CLoG [48]. Our approach trains a single Gaussian Splatting model such that it can be modulated at inference time and adapted to the target computational requirements with a minimal loss on the rendering quality. That allows us to democratize the use of 3DGS and deploy a single model even on handheld devices. We show later that even under a strict case where only 2,000 Gaussians[1] can be used, the object is still recognizable. The most important contribution of our model is that it is continuous by design while the existing baselines assume prior the training the model how many LoD the model should comprise at inference. To change the size of particular LoD in those

[1] Common 3DGS models can achieve from 10^5 to even 10^6 Gaussians.

baselines requires training the whole model from scratch, imposing a significant computational burden.

1.4 Thesis outline

This thesis is structured as follows. We start by introducing the preliminaries related to the Neural Radiance Fields and Gaussian Splatting in Chapter 2—a common theme in all works that appear later. We then move towards describing CoNeRF [50] in Chapter 3, our approach to control trained neural radiance fields by using sparse annotations. In **??**, we describe BlendFields [47] that can produce realistic expression-dependent texture from just a few multi-view frames of the subjects. **??** introduces the LumiGauss [46], a Gaussian Splatting model that learns a radiance transfer function for novel lighting rendering capabilities. Finally, we bring a general method, CLoG [48], that uses Gaussian Splatting to learn continuous levels of detail while training only a single model. We conclude the thesis in Chapter 5 where we also explore possible future avenues that can be undertaken.

1.5 Publications not included in the thesis

We attach a list of articles that are related to and can be used to in neural rendering approaches:

- **Kania**, **K.**, Zięba, M., and Kajdanowicz, T., "UCSG-NET – Unsupervised Discovering of Constructive Solid Geometry Tree," *NeurIPS*, vol. 33, pp. 8776–8786, 2020,
- **Kania**, **K.**, Kowalski, M., and Trzciński, T., "TrajeVAE: Controllable Human Motion Generation from Trajectories," *arXiv preprint arXiv:2104.00351*, 2021,
- Stypułkowski, M., **Kania**, **K.**, Zamorski, M., Zięba, M., Trzciński, T., and Chorowski, J., "Representing Point Clouds with Generative Conditional Invertible Flow Networks," *Pattern Recognition Letters*, vol. 150, pp. 26–32, 2021,
- Xia, S., Yue, J., **Kania**, **K.**, Fang, L., Tagliasacchi, A., Yi, K. M., and Sun, W., "Densify Your Labels: Unsupervised Clustering with Bipartite Matching for Weakly Supervised Point Cloud Segmentation," *arXiv preprint arXiv:2312.06799*, 2023,
- Esposito, S., Xu, Q., **Kania**, **K.**, Hewitt, C., Mariotti, O., Petikam, L., Valentin, J., Onken, A., and Mac Aodha, O., "GeoGen: Geometry-Aware Generative Modeling via Signed Distance Functions," in *CVPRW*, 2024, pp. 7479–7488,
- Spurek, P., Winczowski, S., Zięba, M., Trzciński, T., **Kania**, **K.**, and Mazur, M., "Modeling 3D Surfaces with a Locally Conditioned Atlas," in *ICCS*, Springer, 2024, pp. 100–115.

Chapter 2

Background

2.1 Neural Rendering

2.2 Neural Radiance Field

2.3 3D Gaussian Splatting

Chapter 3

CoNeRF: Controllable Neural Radiance Fields

3.1 Abstract

We extend neural 3D representations to allow for intuitive and interpretable user control beyond novel view rendering (i.e. camera control). We allow the user to annotate which part of the scene one wishes to control with just a small number of mask annotations in the training images. Our key idea is to treat the attributes as latent variables that are regressed by the neural network given the scene encoding. This leads to a few-shot learning framework, where attributes are discovered automatically by the framework, when annotations are not provided. We apply our method to various scenes with different types of controllable attributes (e.g. expression control on human faces, or state control in movement of inanimate objects). Overall, we demonstrate, to the best of our knowledge, for the first time novel view and novel attribute re-rendering of scenes from a single video.

3.2 Introduction

Neural radiance field (NeRF) [72] methods have recently gained popularity thanks to their ability to render photorealistic novel-view images [68, 82, 84, 139]. In order to widen the scope to other possible applications, such as digital media production, a natural question is whether these methods could be extended to enable *direct* and *intuitive* control by a digital artist, or even a casual user. However, current techniques only allow coarse-grain controls over materials [144], color [43], or object placement [132], or only support changes that they are designed to deal with, such as shape deformations on a learned shape space of chairs [62], or are limited to facial expressions encoded by an explicit face model [26]. By contrast, we are interested in *fine-grained* control without limiting ourselves to a specific class of objects or their properties. For example, given a self-portrait video, we would like to be able to control individual *attributes* (e.g. whether

(a) annotations **(b)** novel views **(c)** novel attributes

Figure 2. Teaser – We train a controllable neural radiance field from multiple views of a dynamic 3D scene, under varying poses and attributes; in this example eye being open/closed and mouth smiling/frowning. Given only six annotations (a), our method provides full control over the scene appearance, allowing us to synthesize (b) novel views and (c) novel attributes, including attribute combinations that were *never seen* in the training data (green box).

the mouth is open or closed); see Fig. 2. We would like to achieve this objective with minimal user intervention, without the need of specialized capture setups [61].

However, it is unclear how fine-grained control can be achieved, as current state-of-the-art models [84] encode the structure of the 3D scene in a *single* and *not interpretable* latent code. For the example of face manipulation, one could attempt to resolve this problem by providing *dense* supervision by matching images to the corresponding Facial Action Coding System (FACS) [22] action units. Unfortunately, this would require either an automatic annotation process or careful and extensive per-frame human annotations, making the process expensive, generally unwieldy, and, most importantly, domain-specific. Automated tools for domain-agnostic latent disentanglement are a very active topic of research in machine learning [14, 37, 38], but no effective plug-and-play solution exists yet.

Conversely, we borrow ideas from 3D morphable models (3DMM) [10], and in particular to recent extensions that achieve local control by *spatial disentanglement* of control attributes [77, 123]. Rather than having a single global code controlling the expression of the *entire* face, we would like to have a set of *local* "attributes", each controlling the corresponding *localized* appearance; more specifically, we assume spatial quasi-conditional independence of attributes [123]. For our example in Fig. 2, we seek an attribute capable to control the appearance of the mouth, another to control the appearance of the eye, etc.

Thus, we introduce a learning framework denoted CoNeRF (i.e. Controllable NeRF) that is capable of achieving this objective with just *few-shot* supervision. As illustrated in Fig. 2, given a single one-minute video, and with as little as two annotations per attribute, CoNeRF allows *fine-grained*, *direct*, and *interpretable* control over attributes. Our core idea is to provide, on top of the ground truth attribute tuple, *sparse* 2D mask annotations that specify which region of the image an attribute controls, in spirit of Interactive Digital Photomontage [1] and Video Sprites [95]. Further, by treating attributes as latent variables within the framework, the mask

annotations can be automatically propagated to the whole input video. Thanks to the quasi-conditional independence of attributes, our technique allows us to synthesize expressions that were *never* seen at training time; e.g. the input video never contained a frame where both eyes were closed and the actor had a smiling expression; see Fig. 2 (green box).

Contributions To summarize, our CoNeRF method[1]:

- provides *direct*, *intuitive*, and *fine-grained* control over 3D neural representations encoded as NeRF;
- achieves this via *few-shot* supervision, *e.g.*, just a handful of annotations in the form of attribute values and corresponding 2D mask are needed for a one minute video;
- while inspired by domain-specific facial animation research [123], it provides a *domain-agnostic* technique.

3.3 Related works

Neural Radiance Fields [72] provide high-quality renderings of scenes from novel views with just a few exemplar images captured by a handheld device. Various extensions have been suggested to date. These include ones that focus on improving the quality of results [68, 82, 84, 139], ones that allow a single model to be used for multiple scenes [96, 111], and some considering controllability of the rendering output at a coarse level [34, 62, 128, 132, 136, 144], as we detail next.

In more detail, existing works enable only compositional control of object location [132, 136], and recent extensions also allow for finer-grain reproduction of global illumination effects [34]. NeRFactor [144] shows one can model albedos and BRDFs, and shadows, which can be used to, *e.g.*, edit material, but the manipulation they support is limited to what is modeled through the rendering equation. CodeNeRF [43] and EditNeRF [62] showed that one can edit NeRF models by modifying the shape and appearance encoding, but they require a curated dataset of objects viewed under different views and colors. HyperNeRF [84], on the other hand can adapt to unseen changes specific to the scene, but learns an arbitrary attribute (ambient) space that cannot be supervised, and, as we show in Sec. 3.5, cannot be easily related to specific local attribute within the scene for controllability.

Explicit supervision One can also condition NeRF representations [26] with face attribute predicted by pre-trained face tracking networks, such as Face2Face [109]. Similarly, for human bodies, A-NeRF [100] and NARF [79] use the SMPL [65] model to generate interpretable pose parameters, and Neural Actor [61] further includes normal and texture maps for more detailed rendering. While these models result in controllable NeRF, they are limited to domain-specific control and the availability of a heavily engineered control model.

[1] Code and dataset are released here.

Controllable neural implicits Controllability of neural 3D *implicit* representations has also been addressed by the research community. Many works have limited focus on learning *human* neural implicit representations while enabling the control via SMPL parameters [65], or linear blend skinning weights [2, 20, 35, 66, 70, 94, 146, 151]. Some initial attempts at learned disentangled of shape and poses have also been made in A-SDF [75], allowing behavior control of the output geometry (*e.g.* doors open vs. closed) while maintaining the general shape. However, the approach is limited to controlling SE(3) articulation of objects, and requires dense 3D supervision.

3.3.1 Neural Radiance Field (NeRF)

For completeness, we briefly discuss NeRF before diving into the details of our method. A Neural Radiance Field captures a volumetric representation of a specific scene within the weights of a neural network. As input, it receives a sample position $\mathbf{x}$ and a view direction $\mathbf{v}$ and outputs the density of the scene σ at position $\mathbf{x}$ as well as the color $\mathbf{c}$ at position $\mathbf{x}$ as seen from view direction $\mathbf{v}$. One then renders image pixels $\mathbf{C}$ via volume rendering [45]. In more detail, $\mathbf{x}$ is defined by observing rays $\mathbf{r}(t)$ as $\mathbf{x} = \mathbf{r}(t)$, where t parameterizes at which point of the ray you are computing for. One then renders the color of each pixel $\mathbf{C}(\mathbf{r})$ by computing

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

where $\mathbf{v}$ is the viewing angle of the ray $\mathbf{r}$, t_n and t_f are the near and far planes of the rendering volume, and

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

is the accumulated transmittance. Integration in Eq. (1) is typically done via numerical integration [72].

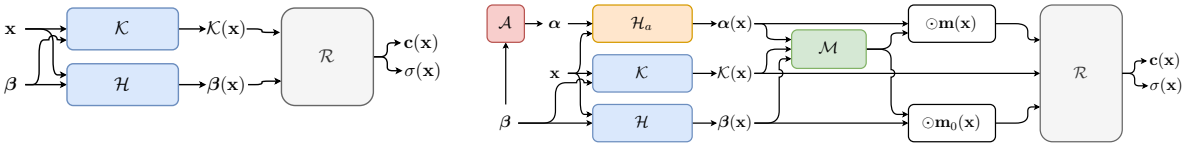
3.3.2 HyperNeRF

Note that in its original formulation Eq. (1) is only able to model *static* scenes. Various recent works [82, 84, 110] have been proposed to explicitly account for possible appearance changes in a scene (for example, temporal changes in a video). To achieve this, they introduce the notion of *canonical hyperspace* – more formally given a 3D query point $\mathbf{x}$ and the collection $\boldsymbol{\theta}$ of all parameters that describe the model, they define:

$$\mathcal{K}(\mathbf{x}) \equiv \mathcal{K}(\mathbf{x}; \boldsymbol{\beta}, \boldsymbol{\theta}), \qquad \text{Canonicalizer} \tag{3}$$

$$\boldsymbol{\beta}(\mathbf{x}) \equiv \mathcal{H}(\mathbf{x}; \boldsymbol{\beta}, \boldsymbol{\theta}), \qquad \text{Hyper Map} \tag{4}$$

$$\mathbf{c}(\mathbf{x}), \sigma(\mathbf{x}) = \mathcal{R}(\mathcal{K}(\mathbf{x}), \boldsymbol{\beta}(\mathbf{x}); \boldsymbol{\theta}). \qquad \text{Hyper NeRF} \tag{5}$$



(a) Neural radiance field with lifting [84] **(b)** Controllable neural radiance field (our method)

Figure 3. Framework – We depict in (a) the HyperNeRF [84] formulation, and (b) our Controllable-NeRF (CoNeRF). In (a), both point coordinates $\mathbf{x}$ and latent representation $\boldsymbol{\beta}$ are respectively processed by a canonicalizer $\mathcal{K}$ and a hyper map $\mathcal{H}$, which are then turned into radiance and density field values by $\mathcal{R}$. In (b), we introduce regressors $\mathcal{A}$ and $\mathcal{M}$ that regress the attribute and the corresponding mask that enable few-shot attribute-based control of the NeRF model. See Sec. 3.4.3 for details.

where the location is canonicalized via a canonicalizer $\mathcal{K}$, and the appearances, represented by $\boldsymbol{\beta}$, are mapped to a hyperspace via $\mathcal{H}$, which are then utilized by another neural network $\mathcal{R}$ to retrieve the color $\mathbf{c}$ and the density σ at the query location. Note throughout this paper we denote $\boldsymbol{\beta}$ to indicates a latent code, while $\boldsymbol{\beta}(\mathbf{x})$ to indicate the corresponding field generated by the hypermap lifting. With this latent lifting, these methods render the scene via Eq. (1). Note that the original NeRF model can be thought of the case where $\mathcal{K}$ and $\mathcal{H}$ are identity mappings.

3.4 Controllable NeRF (CoNeRF)

Given a collection of C color images $\{\mathbf{C}_c\} \in [0,1]^{W\times H\times 3}$, we train our controllable neural radiance field model by an auto-decoding optimization [81] whose losses can be grouped into two main subsets:

$$\underset{\boldsymbol{\theta}=\boldsymbol{\theta},\{\boldsymbol{\beta}_c\}}{\arg\min}\ \underbrace{\ell_{\text{rep}}(\boldsymbol{\theta};\{\mathbf{C}_c\})}_{\text{Sec. 3.4.1}} + \underbrace{\ell_{\text{ctrl}}(\boldsymbol{\theta};\{\mathbf{M}^{\text{gt}}_{c,a}\},\{\alpha^{\text{gt}}_{c,a}\})}_{\text{Sec. 3.4.2}}. \tag{6}$$

The first group consists of the classical HyperNeRF [84] auto-decoder losses, attempting to optimize neural network parameters $\boldsymbol{\theta}$ jointly with latent codes $\{\boldsymbol{\beta}_c\}$ to *reproduce* the corresponding input images $\{\mathbf{C}_c\}$:

$$\ell_{\text{rep}}(\cdot) = \ell_{\text{recon}}(\boldsymbol{\theta},\{\boldsymbol{\beta}_c\};\{\mathbf{C}_c\}) + \ell_{\text{enc}}(\{\boldsymbol{\beta}_c\}). \tag{7}$$

The latter allow us to inject *explicit control* into the representation, and are our core contribution:

$$\ell_{\text{ctrl}}(\cdot) = \ell_{\text{mask}}(\boldsymbol{\theta},\{\boldsymbol{\beta}_c\};\{\mathbf{M}^{\text{gt}}_{c,a}\}) \qquad \text{g.t. masks} \tag{8}$$

$$+\ \ell_{\text{attr}}(\boldsymbol{\theta},\{\boldsymbol{\beta}_c\};\{\alpha^{\text{gt}}_{c,a}\}). \qquad \text{g.t. attributes} \tag{9}$$

As mentioned earlier in Sec. 3.2, we aim for a neural 3D appearance model that is controlled by a collection of attributes $\boldsymbol{\alpha}=\{\alpha_a\}$, and we expect each image to be a manifestation of a different value of attributes, that is, each image $\mathbf{C}_c$, and hence each latent code $\boldsymbol{\beta}_c$, will have a

corresponding attribute $\boldsymbol{\alpha}_c$. The learnable connection between latent codes $\boldsymbol{\beta}$ and the attributes $\boldsymbol{\alpha}$, which we represent via regressors, is detailed in Sec. 3.4.3.

3.4.1 Reconstruction losses

The primary loss guiding the training of the NeRF model is the reconstruction loss, which simply aims to reconstruct observations $\{\mathbf{C}_c\}$. As in other neural radiance field models [68, 72, 82, 84] we simply minimize the L2 photometric reconstruction error with respect to ground truth images:

$$\ell_{\text{recon}}(\cdot) = \sum_c \mathbb{E}_{\mathbf{r}\sim\mathbf{C}_c}\left[\left\|\mathbf{C}(\mathbf{r};\boldsymbol{\beta}_c,\boldsymbol{\theta}) - \mathbf{C}^{\text{gt}}(\mathbf{r})\right\|_2^2\right]. \tag{10}$$

As is typical in auto-decoders, and following [81], we impose a zero-mean Gaussian prior on the latent codes $\{\boldsymbol{\beta}_c\}$:

$$\ell_{\text{enc}}(\cdot) = \sum_c \|\boldsymbol{\beta}_c\|_2^2. \tag{11}$$

3.4.2 Control losses

The user defines a *discrete* set of A number of attributes that they seek to control, that are *sparsely* supervised across frames—we only supervise attributes *when* we have an annotation, and let others be discovered on their own throughout the training process, as guided by Eq. (7). More specifically, for a particular image $\mathbf{C}_c$, and a particular attribute α_a, the user specifies the quantities:

- $\alpha_{c,a} \in [-1, 1]$: specifying the value for the a-th attribute in the c-th image; see the *sliders* in Fig. 2;
- $\mathbf{M}_{c,a} \in [0,1]^{W\times H}$: roughly specifying the image region that is controlled by the a-th attribute in the c-th image; see the *masks* in Fig. 2.

To formalize sparse supervision, we employ an indicator function $\mathbb{1}_{c,a}$, where $\mathbb{1}_{c,a} = 1$ if an annotation for attribute a for image c is provided, otherwise $\mathbb{1}_{c,a} = 0$. We then write the loss for *attribute* supervision as:

$$\ell_{\text{attr}}(\cdot) = \sum_c \sum_a \mathbb{1}_{c,a} |\alpha_{c,a} - \alpha_{c,a}^{\text{gt}}|^2. \tag{12}$$

For the mask few-shot supervision, we employ the volume rendering in Eq. (20) to project the 3D volumetric neural mask field $\mathbf{m}_a(\mathbf{x})$ into image space, and then supervise it as:

$$\ell_{\text{mask}}(\cdot) = \sum_{c,a} \mathbb{1}_{c,a}\, \mathbb{E}_{\mathbf{r}}\left[\text{CE}\left(\mathbf{M}(\mathbf{r};\boldsymbol{\beta}_c,\boldsymbol{\theta}), \mathbf{M}_{c,a}^{\text{gt}}(\mathbf{r})\right)\right], \tag{13}$$

where $\text{CE}(\cdot,\cdot)$ denotes cross entropy, and the field $\sigma(\mathbf{x})$ in Eq. (20) is learned by minimizing Eq. (10). Importantly, as we do not wish for Eq. (13) to interfere with the training of

the underlying 3D representation learned through Eq. (10), we *stop gradients* in Eq. (13) w.r.t. $\sigma(\mathbf{x})$. Furthermore, in practice, because the attribute mask vs. background distribution can be highly imbalanced depending on which attribute the user is trying to control (*e.g.* an eye only covers a very small portion of an image), we employ a *focal loss* [59] in place of the standard cross entropy loss.

3.4.3 Controlling and rendering images

In what follows, we drop the image subscript c to simplify notation without any loss of generality. Given a B-dimensional latent code $\boldsymbol{\beta}$ representing the 3D scene behind an image, we derive a mapping to our A attributes via a neural map $\mathcal{A}$ with learnable parameters $\boldsymbol{\theta}$:

$$\{\alpha_a\} = \mathcal{A}(\boldsymbol{\beta}; \boldsymbol{\theta}), \quad \mathcal{A} : \mathbb{R}^B \rightarrow [0,1]^A, \tag{14}$$

where these correspond to the *sliders* in Fig. 2. In the same spirit of Eq. (4), to allow for complex topological changes that may not be represented by the change in a single scalar value alone, we lift the attributes to a hyperspace. In addition, since each attribute governs different aspects of the scene, we employ *per-attribute* learnable hypermaps $\{\mathcal{H}_a\}$, which we write:

$$\alpha_a(\mathbf{x}) = \mathcal{H}_a(\mathbf{x}, \alpha_a; \boldsymbol{\theta}) \quad \mathcal{H}_a : \mathbb{R}^3 \times \mathbb{R} \rightarrow \mathbb{R}^d. \tag{15}$$

Note that while α_a is a scalar *value*, $\alpha_a(\mathbf{x})$ is a *field* that can be queried at any point $\mathbf{x}$ in space. These fields are concatenated to form $\boldsymbol{\alpha}(\mathbf{x}) = \{\alpha_a(\mathbf{x})\}$.

We then provide all this information to generate an *attribute masking field* via a network $\mathcal{M}(\cdot; \boldsymbol{\theta})$. This field determines which attribute *attends* to which position in space $\mathbf{x}$:

$$\mathbf{m}_0(\mathbf{x}) \oplus \mathbf{m}(\mathbf{x}) = \mathcal{M}(\mathcal{K}(\mathbf{x}), \boldsymbol{\beta}(\mathbf{x}), \boldsymbol{\alpha}(\mathbf{x}); \boldsymbol{\theta}), \tag{16}$$

$$\mathcal{M} : \mathbb{R}^3 \times \mathbb{R}^B \times \mathbb{R}^{A \times d} \rightarrow \mathbb{R}_+^{A+1}, \tag{17}$$

where $\oplus$ is a concatenation operator, $\mathbf{m}(\mathbf{x})$=$\{\mathbf{m}_a(\mathbf{x})\}$, and the additional mask $\mathbf{m}_0(\mathbf{x})$ denotes space that is not affected by *any* attribute. Note that because the mask location should be affected by both the particular attribute of interest (*e.g.*, the selected eye status) and the global appearance of the scene (*e.g.*, head movement), $\mathcal{M}$ takes both $\boldsymbol{\beta}(\mathbf{x})$ and $\boldsymbol{\alpha}(\mathbf{x})$ as input in addition to $\mathcal{K}(\mathbf{x})$. In addition, because the mask is modeling the attention related to attributes, collectively, these masks satisfy the partition of unity property:

$$\mathbf{m}_0(\mathbf{x}) + \Sigma_a[\mathbf{m}_a(\mathbf{x})] = 1 \quad \forall \mathbf{x} \in \mathbb{R}^3. \tag{18}$$

Finally, in a similar spirit to Eq. (5), all of this information is processed by a neural network that produces the desired radiance and density fields used in volume rendering:

$$\left.\begin{matrix}\mathbf{c}(\mathbf{x})\\ \sigma(\mathbf{x})\end{matrix}\right\} = \mathcal{R}(\mathcal{K}(\mathbf{x}), \underbrace{\mathbf{m}(\mathbf{x}) \odot \boldsymbol{\alpha}(\mathbf{x})}_{\text{attribute controls}}, \underbrace{\mathbf{m}_0(\mathbf{x}) \cdot \boldsymbol{\beta}(\mathbf{x})}_{\text{everything else}}; \boldsymbol{\theta}). \tag{19}$$

In particular, note that $\mathbf{m}(\mathbf{x})=0$ implies $\mathbf{m}_0(\mathbf{x})=1$, hence our solution has the capability of reverting to classical HyperNeRF Eq. (5), where all change in the scene is globally encoded in $\boldsymbol{\beta}(\mathbf{x})$. Finally, these fields can be used to render the mask in image space, following a process analogous to volume rendering of radiance:

$$\mathbf{M}(\mathbf{r}; \boldsymbol{\theta}) = \int_{t_n}^{t_f} T(t) \cdot \sigma(\mathbf{r}(t)) \cdot [\mathbf{m}_0(\mathbf{r}(t)) \oplus \mathbf{m}(\mathbf{r}(t))]\, dt. \tag{20}$$

We depict our inference flow in Fig. 3 (b).

3.4.4 Implementation details

We implement our method for NeRF based on the JAX [11] implementation of HyperNeRF [84]. We use both the scheduled windowed positional encoding and weight initialization of [82], as well as the coarse-to-fine training strategy [84].

Besides the newly added networks, we follow the same architecture as HyperNeRF. For the attribute network $\mathcal{A}$ we use a six-layer multi-layer perceptron (MLP) with 32 neurons at each layer, with a skip connection at the fifth layer, following [82, 84]. For the lifting network $\mathcal{H}_a$, we use the same architecture as $\mathcal{H}$, except for the input and output dimension sizes. For the masking network $\mathcal{M}$ we use a four-layer MLP with 128 neurons at each layer, followed by an additional 64 neuron layer with a skip connection. The network $\mathcal{R}$ also shares the same architecture as HyperNeRF, but with a different input dimension size to accommodate for the changes our method introduces.

2D implementation To show that our idea is not limited to neural radiance fields, we also test a 2D version of our framework that can be used to directly represent images, without going through volume rendering. We use the same architecture and training procedure as in the NeRF case, with the exception that we do not predict the density σ, and we also do not have the notion of depth—each ray is directly the pixel. We center crop each video and resize each frame to be 128×128.

Hyperparameters We train all our NeRF models with 480×270 images and with 128 samples per ray. We train for 250k iterations with a batch size of 512 rays. During training, we maintain that 10% of rays are sampled from annotated images. We set $\ell_{\text{attr}} = 10^{-1}$, $\ell_{\text{mask}} = 10^{-2}$ and $\ell_{\text{enc}} = 10^{-4}$. For the number of hyper dimensions we set $d = 8$. For the 2D implementation

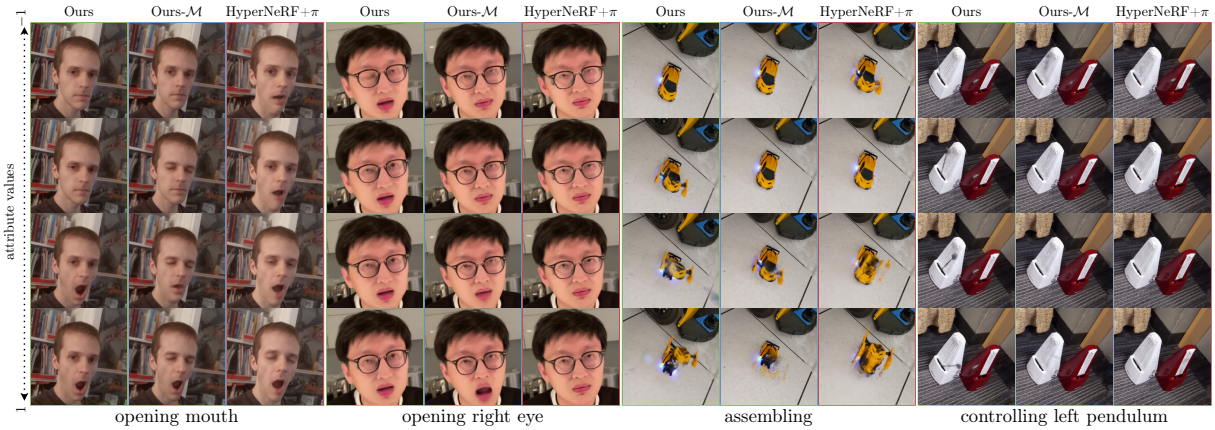


Figure 4. Novel view and novel attribute synthesis on real data – We synthesize scenes from a novel view and with a novel attribute combination, not seen during training. A naive extension of HyperNeRF, HyperNeRF+π fails to disentangle attributes and results in a modification of the scene irrespectively of attribute meaning *e.g.*, opening mouth results in closing eyes at the same time. Ours-$\mathcal{M}$ improves the results, but does not disentangles the attribute space, as successfully done by our complete method. The differences between these methods can even lead to complete failure cases, as shown in the metronome and the toy car case.

experiments, we sample 64 random images from the scene and further subsample 1024 pixels from each of them. For all experiments we use Adam [54] with learning rate 10^{-4} exponentially decaying to 10^{-5} in 250k iterations. We provide additional details in the supplementary material. Training a single model takes around 12 hours on an NVIDIA V100 GPU.

3.5 Results

3.5.1 Datasets and baselines

We evaluate our method on two datasets: real video sequences captured with a smartphone (*real dataset*) and synthetically rendered sequences (*synthetic dataset*). Here we introduce those datasets and the baselines for our approach.

Real dataset Each of the seven real sequences is 1 minute long and was captured either with a Google Pixel 3a or an Apple iPhone 13 Pro. Four of them consists of people performing different facial expressions including smiling, frowning, closing or opening eyes, and opening mouth. For the other three, we captured a toy car changing its shape (*a.k.a.* Transformer), a single metronome, and two metronomes beating with different rates. For one of the four videos depicting people, to use it for the 2D implementation case, we captured it with a static camera that shows a frontal view of the person. All other sequences feature camera motions showing front and sides of the object in the center of the scene. For videos with human subjects, the subjects signed a participant consent form, which was approved by a research ethics board. We informed the participants that their data will be altered with our method.

We extract frames at 15 FPS which gives approximately 900 frames per capture. Because novel attribute synthesis via user control on real scenes does not have a ground truth view—we aim to create scenes with unseen attribute combinations—the benefit of our method is best seen qualitatively. Nonetheless, to quantitatively evaluate the rendering quality, we interpolate between two frames and evaluate its quality. In more detail, to minimize the chance of the dynamic nature of the scene interfering with this assessment, we use every other frame as a test frame for the interpolation task.

For all human videos, we define three attributes—one for the status of each of the two eyes, and one for the mouth. We annotate only six frames per video in this case, specifically the frames that contain the extremes of each attribute (*e.g.*, left eye fully open). For the toy car, we set the shape of the toy car to be an attribute, and annotate two extremes from two different view points—when the toy is in robot-mode and when it is in car-mode from its left and right side. For the metronomes, we consider the state of the pendulum to be the attribute and annotate the two frames with the two extremes for the single metronome case, and seven frames for the two metronome case as the pendulums of the two metronomes are often close to each other and required special annotations for these close-up cases; see Fig. 4.

Synthetic dataset Since the lack of ground-truth data renders measuring the quality of novel attribute synthesis infeasible in practice, we leverage Kubric software [33] to generate synthetic dataset, where we know exactly the state of each object in the scene. We create a simple scene where three 3D objects, the teapot [107], the Stanford bunny [12], and Suzanne [103], are placed within the scene and are rendered with varying surface colors, which are our attributes; see Fig. 6. We generate 900 frames for training and 900 frames for testing. To ensure that the attribute combination during training is not seen in the test scene, we set the attributes to be synchronized for the training split, and desynchronized for the test split. We further render the test split from different camera positions than the training split to account for novel views. We randomly sample 5% of the frames with a given attribute for each object to be set as the ground-truth attribute. During validation, we use attribute values directly to predict the image.

Baselines To evaluate the reconstruction quality of our method, CoNeRF, we compare it with four different baselines: ① standard NeRF [72]; ② NeRF+Latent, a simple extension to NeRF where we concatenate each coordinate $\mathbf{x}$ with a learnable latent code $\boldsymbol{\beta}$ to support appearance changes of the scene; ③ Nerfies [82]; and ④ HyperNeRF[2] [84]. Additionally, as existing methods do not support attribute-based control with a few-shot supervision, we create another baseline ⑤ by extending HyperNeRF with a simple linear regressor π that regresses $\boldsymbol{\beta}_c$ given $\boldsymbol{\alpha}_c$. We call this baseline HyperNeRF+π. To further show the importance of masking, we also compare our approach against a stripped-down version of our method, Ours-$\mathcal{M}$, where we disable the part

[2]We use the version with dynamic plane slicing as it consistently outperforms the axis-aligned strategy; see [84] for more details.

(a) annotation **(b)** unannotated views

Figure 5. Annotation example – We provide only a rough annotation for each attribute, which is enough for the method to discover the mask for each attribute across all views automatically. Bottom row shows masks overlaid on the image.

of our pipeline responsible for masking. All baselines that utilize annotations were trained with the same sparse labels as our method.

3.5.2 Comparison with the baselines

Qualitative highlights We first show qualitative examples of novel attribute and view synthesis on the real dataset in Fig. 4. Our method allows for controlling the selected attribute without changing other aspects of the image—our control is disentangled. This disentanglement allows our method to generate images with attribute combinations that were not seen at training time. On the contrary, as there is no incentive for the learned embeddings of HyperNeRF to be disentangled, the simple regression strategy of HyperNeRF+π results in entangled control, where when one tries to close/open the mouth it ends up affecting the eyes. The same phenomenon happens also for Ours-$\mathcal{M}$. Moreover, due to the complexity of motions in the scene, HyperNeRF+π fails completely to render novel views of the toy car, whereas our method, with only four annotated frames, successfully provides both controllability and high-quality renderings. Please also see `Supplementary` for more qualitative results, including a video demonstration.

Note that in all of these sequences, we provide highly sparse annotations and yet our method learns how each attribute should influence the appearance of the scene. In Fig. 5, we show an example annotation and how the method finds the mask for unannotated views.

Quantitative results on synthetic dataset To complete the qualitative evaluation of our method, we provide results using synthetic dataset with available ground truth. We measure Peak Signal-to-Noise Ratio (PSNR), Multi-scale Structural Similarity (MS-SSIM) [118], and Learned Perceptual Image Patch Similarity (LPIPS) [141] and report them in Tab. 1. With only

Method	PSNR↑	MS-SSIM↑	LPIPS↓
HyperNeRF+π	25.963	0.854	0.158
Ours-$\mathcal{M}$	27.868	0.898	0.155
Ours	**32.394**	**0.972**	**0.139**

Table 1. Novel view and novel attributes results – We report average PSNR, MS-SSIM, and LPIPS values for novel view and novel attribute synthesis on synthetic data. Our method gives the best results.

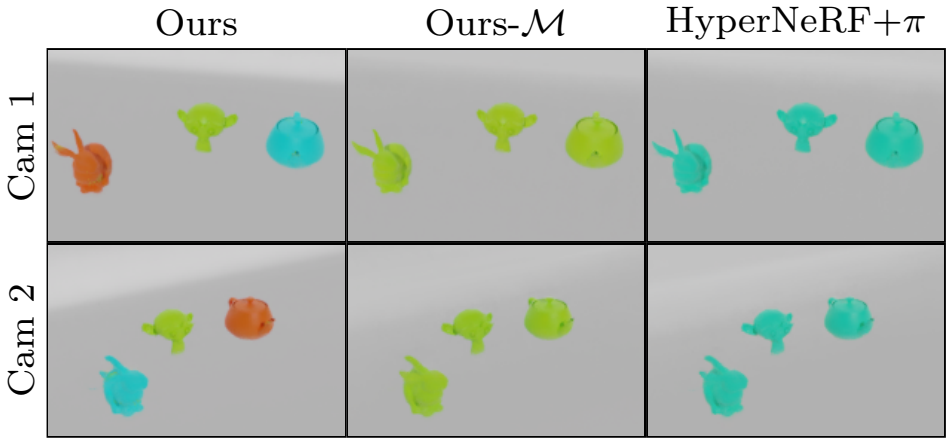


Figure 6. Novel view and novel attribute synthesis on synthetic data – We show examples of novel view and novel attribute synthesis on synthetic data. The scene is composed of three objects, where the color of each object is their attribute. Our method provides control over the color of each object independently, whereas both HyperNeRF+π and Ours-$\mathcal{M}$ fail to deliver controllability and results in all three objects having the same attribute in the rendered scene.

Method	PSNR ↑	MS-SSIM ↑	LPIPS ↓
NeRF	28.795	0.951	0.210
NeRF + Latent	32.653	0.981	0.182
NeRFies	32.274	0.981	0.180
HyperNeRF	32.520	0.981	0.169
Ours-$\mathcal{M}$	32.061	0.979	0.167
Ours	32.342	0.981	0.168

Table 2. Quantitative results (interpolation) – We report results in terms of PSNR, MS-SSIM, and LPIPS for the interpolation task. These results are obtained for interpolated view synthesis only, not for novel attribute rendering. Our method provides similar performance in terms of rendering quality, but with controllability.

5% of the annotations, our method provides the best novel-view and novel-attribute synthesis results, as reconfirmed by the qualitative examples in Tab. 1. As shown, neither HyperNeRF+π nor Ours-$\mathcal{M}$ is able to provide good results in this case, as without disentangled control of each attribute, the novel attribute and view settings of each test frame cannot be synthesized properly.

Interpolation task To further verify that our rendering quality does not degrade with the introduction of controllability, we evaluate our method on a frame interpolation task without any attribute control. Unsurprisingly, as shown in Tab. 2, all methods that support dynamic scenes work similarly, including ours for interpolation. Note that for the interpolation task, we interpolate every other frame, in order to minimize the chance of attributes affecting the evaluation. Here, we are purely interested in the rendering quality from a novel view.

3.5.3 Direct 2D rendering

To verify how our approach generalizes beyond NeRF models and volume rendering, we apply our method to videos taken from a single view point, creating a 2D rendering task. We show in Fig. 7 a proof-of-concept for employing our approach outside of NeRF applications to allow controllable neural generative models.

3.5.4 Ablation study

Loss functions In Tab. 3, we show how each loss term affects the network's performance, contributing to performance improvements. When rendering novel views with novel attributes, the full formulation is a must, as without all loss terms the performance drops significantly—for example, results without ℓ_{mask} is similar to Ours-$\mathcal{M}$ results in Tab. 1 and Fig. 6. In the case of the interpolation task, the additional loss functions for controllability have no significant effect on the rendering quality. In other words, our controllability losses **do not interfere** with the rendering quality, other than imbuing the framework with controllability.

Figure 7. 2D image generation example – Our framework also generalizes to direct generation of 2D images. Here we show novel attribute synthesis for a webcam video of a person making expressions. Each individual part of the scene is correctly controlled according to the attribute values.

	Real (interpolation)			Synthetic (novel view & attr.)		
Model	PSNR ↑	MS-SSIM ↑	LPIPS ↓	PSNR ↑	MS-SSIM ↑	LPIPS ↓
Base (ℓ_{recon})	32.457	0.981	0.168	24.407	0.718	0.173
$+\ell_{\text{enc}}$	32.478	0.982	0.167	27.018	0.871	0.164
$+\ell_{\text{enc}} + \ell_{\text{attr}}$	32.254	0.981	0.167	27.322	0.873	0.147
$+\ell_{\text{enc}} + \ell_{\text{attr}} + \ell_{\text{mask}}$	32.342	0.981	0.168	**32.394**	**0.972**	**0.139**

Table 3. Effect of loss functions – We report the rendering quality of our method as we procedurally introduce the loss terms. For controlled rendering with novel views and attributes (synthetic data), each loss term adds to the rendering quality, with the ℓ_{mask} being critical. For the novel view rendering on real data, addition of loss functions for controllability do not have a significant effect on the rendering quality—they do no harm.

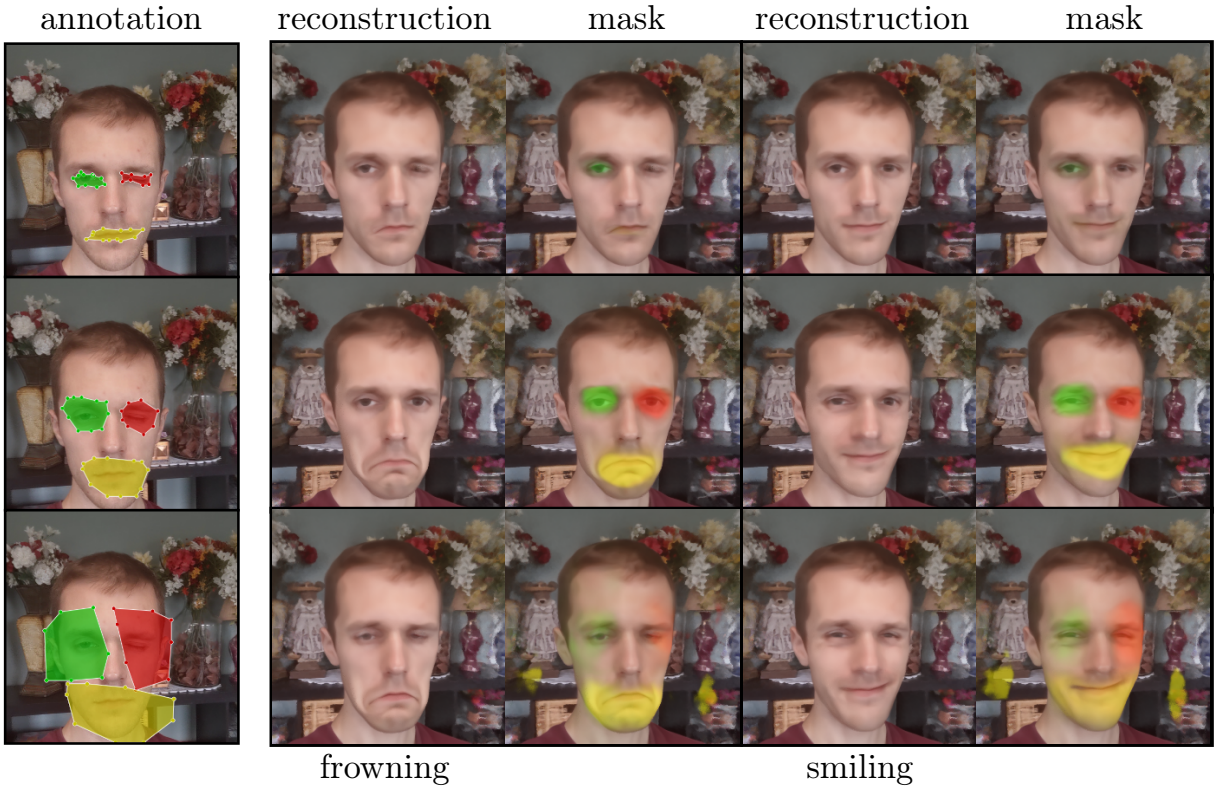
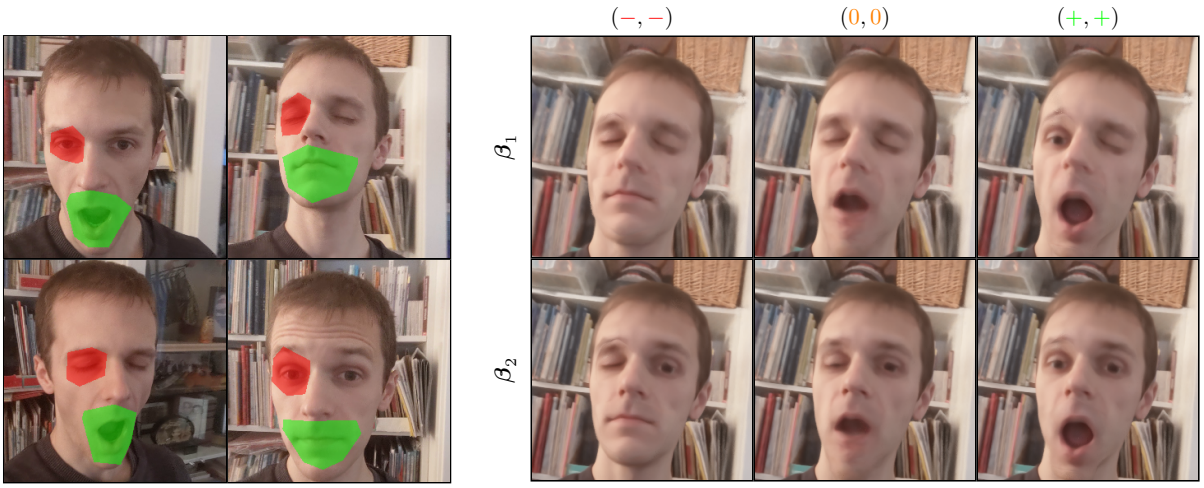


Figure 8. Effect of annotation quality – Our method is moderately robust to the quality of annotations. We visualize the results for two expressions: frowning and smiling, while keeping both eyes in a neutral position. Even with wildly varying annotations as shown, the reconstructions are reasonably controlled, with the exception of the top row, where we show a case where the annotations is too restrictive, resulting in the annotation being ignored for one eye. We show in bottom row also an interesting case, where the mask is large enough to start capturing the correlation among mouth expressions and the eye.



(a) annotated samples **(b)** renderings

Figure 9. Example with unannotated attributes – We show an example of how our method performs when a part of the image changes appearance, but is not annotated. With the annotations in (a), we synthesize the scene with novel view and attributes in (b), where the two rows are with different $\boldsymbol{\beta}$ configurations. We denote the attribute configuration on the top of each column in (b). As shown, the change that is not annotated is simply encoded in the per-image encoding $\boldsymbol{\beta}$.

Quality of few shot supervision We test how sensitive our method is against the quality of annotation supervision. In Fig. 8 we demonstrate how each annotation leads to the final rendering quality. Our framework is robust to a moderate degree to the inaccuracies in the annotations. However, when they are too restrictive, the mask may collapse, as shown on the top row. Too large of a mask could also lead to moderate entanglement of attributes, as shown in the bottom row. Still, in all cases, our method provides a reasonable control over what is annotated.

Unannotated attributes A natural question to ask is then what happens with the unannotated changes that may exist in the scene. In Fig. 9 we show how the method performs when annotating only parts of the appearance change within the scene. The unannotated changes of the scene get encoded as $\boldsymbol{\beta}$, as in the case of HyperNeRF [84].

3.6 Conclusions

We have introduced CoNeRF, an intuitive controllable NeRF model that can be trained with few-shot annotations in the form of attribute masks. The core contribution of our method is that we represent attributes as localized masks, which are then treated as latent variables within the framework. To do so we regress the attribute and their corresponding masks with neural networks. This leads to a few-shot learning setup, where the network learns to regress provided annotations, and if they are not provided for a given image, proper attributes and masked are

discovered throughout training automatically. We have shown that our method allows users to easily annotate what to control and how, within a single video simply by annotating a few frames, which then allows rendering of the scene from novel views and with novel attributes, at high quality.

Limitations While our method delivers controllability to NeRF models, there is room for improvement. First, our disentanglement of attribute strictly relies on the locality assumption—if multiple attributes act on a single pixel, our method is likely to have entangled outcomes when rendering with different attributes. An interesting direction would therefore be to incorporate manifold disentanglement approaches [57, 143] to our method. Second, while very few, we still require sparse annotations. Unsupervised discovery of controllable attributes, for example as in [55], in a scene remains yet to be explored. Lastly, we resort to user intuition on which frames should be annotated—we heuristically choose frames with extreme attributes (*e.g.*, mouth fully open). While this is a valid strategy, an interesting direction for future research would be to employ active learning techniques for this purpose [9, 91]

We further discuss potential societal impact of our work in the `Supplementary`.

3.7 Acknowledgements

We thank Thabo Beeler, JP Lewis, and Mark J. Matthews for their fruitful discussions, and Daniel Rebain for helping with processing the synthetic dataset. The work was partly supported by National Sciences and Engineering Research Council of Canada (NSERC), Compute Canada, and Microsoft Mixed Reality & AI Lab. This research was funded by Foundation for Polish Science (grant no POIR.04.04.00-00-14DE/18-00 carried out within the Team-Net program co-financed by the European Union under the European Regional Development Fund), National Science Centre, Poland (grant no 2020/39/B/ST6/01511), and by Microsoft Research through EMEA PhD Scholarship Programme. The authors have applied a CC BY license to any Author Accepted Manuscript (AAM) version arising from this submission, in accordance with the grants' open access conditions.

CoNeRF: Controllable Neural Radiance Fields

Supplementary Material

3.8 Potential social impact

Our work is originally intended for creative and entertainment purposes, for example to allow users to easily edit their personal photos to have all the members of a group photo to have

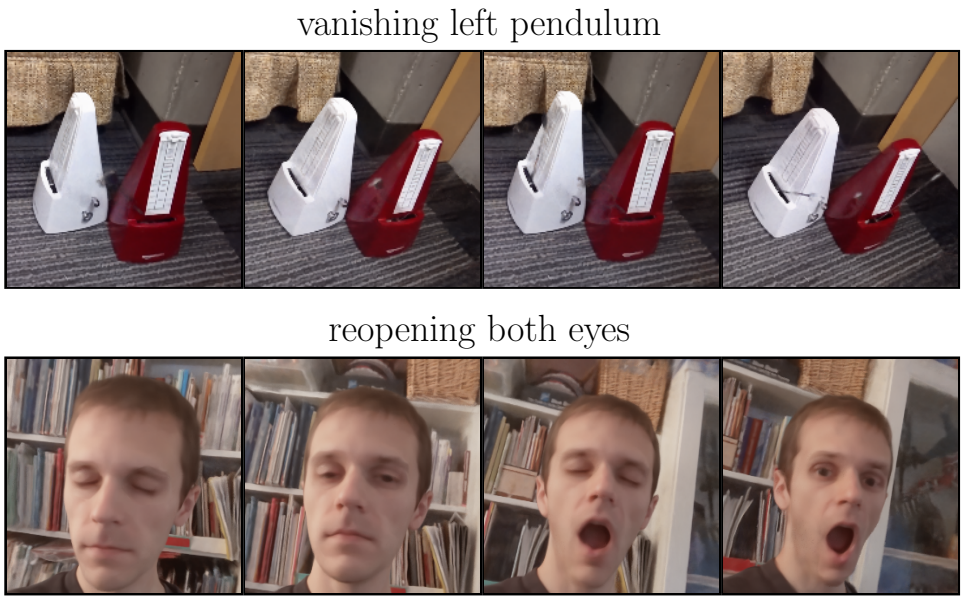


Figure 10. Failure cases – Our model may learn spurious interpolations for controlled elements that occupy little space in the image and with insufficient/careless annotations. For the metronome, due to the fast motion of the pendulum and its specularity, without careful annotation our method may simply learn its motion blur or sometimes even completely ignore the pendulum. In the face example, this may result in the eye blinking multiple times while interpolating between the attribute values of -1 and 1. Both cases are preventable with more careful annotations and by annotating more frames.

their eyes open. However, as with all work that enable editable models, our method has the potential to be misused for malicious purposes such as deep fakes. We strongly advise against such misuse. Recent work [3] has shown that it is possible to detect deep fakes, hinting that it should be possible to detect these deep learning-generated images. One of our future research direction is also along these lines, where we now aim to reliably detect images generated by our method.

3.9 Architecture details

We present architecture of: canonicalizer $\mathcal{K}$ in Fig. 11, attribute map $\mathcal{A}$ in Fig. 12, hypermap $\mathcal{H}$ in Fig. 13, per-attribute hypermap in Fig. 14, mask prediction network in Fig. 15 and the rendering network in Fig. 16. Each network contains only fully connected layers. Hidden layers use ReLU activation function. Colors of figures correspond to colors of blocks in Fig. 3b.

3.10 Failure Cases

We identify two modes of failure cases in our approach and present them in Fig. 10. In some cases with particular mask annotations, our model can struggle with controlling elements that occupy small space in the image. The problem is especially visible for controlling pendulum

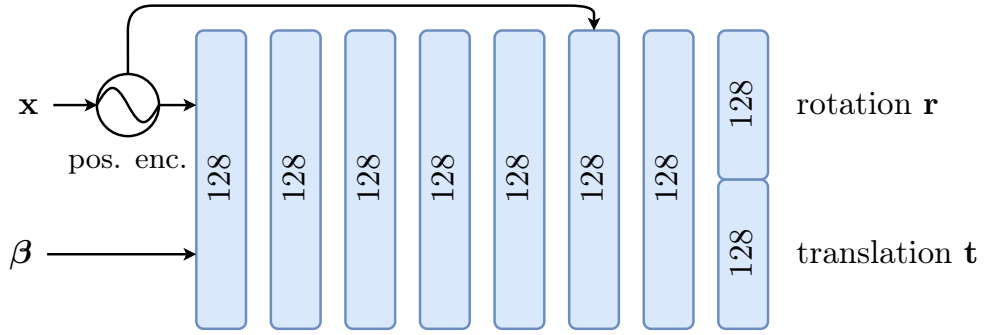


Figure 11. The canonicalization network takes positionally encoded raw coordinates $\mathbf{x}$ and learnable per-image latent code $\boldsymbol{\beta}$ and outputs rotation $\mathbf{r}$ expressed as a quaternion and translation $\mathbf{t}$. We rigidly transform each point $\mathbf{x}$ with an affine transform using both output. We use windowed positional encoding [82] for $\mathbf{x}$ with 8 components, linearly increasing contribution of components throughout 80k steps. We initialize the last layer to small values so the network can learn a base structure of the data.

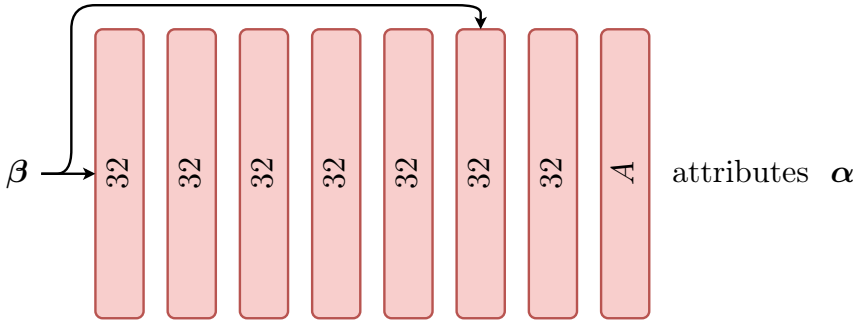


Figure 12. The attribute map $\mathcal{A}$ takes a per-image learnable latent code $\boldsymbol{\beta}$ and outputs A attributes $\boldsymbol{\alpha}$.

movement or opening and closing eyes. In the former, pendulum disappears and reappears in different places. In the latter, the control of eyes is periodic and there are two distant values in $[-1, 1]$ that produce opening eyes. While with careful annotations we noticed that the problem is mostly preventable, this problem may occur in practice.

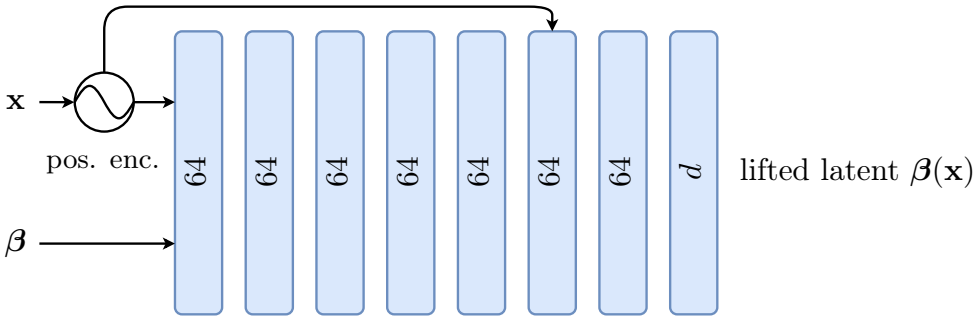


Figure 13. The network predicting lifted latent code $\boldsymbol{\beta}$, takes per-image $\boldsymbol{\beta}$ as an input, positionally encoded raw points $\boldsymbol{\beta}$ and outputs a lifted code of size d. We use only one sine component to encode $\mathbf{x}$.

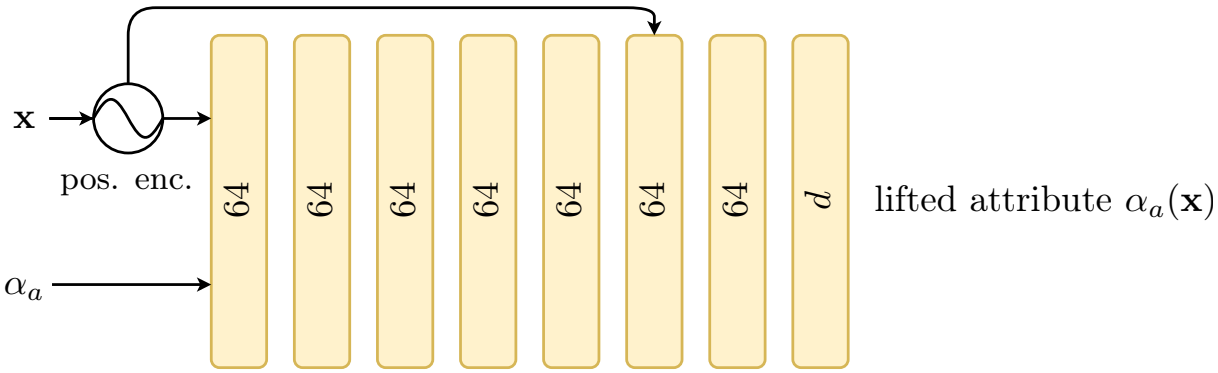


Figure 14. Per-attributes hypermaps take an attribute together with encoded $\mathbf{x}$ coordinates and output lifted $\alpha_a(\mathbf{x})$ ambient code of size d. We encode $\mathbf{x}$ with only single component.

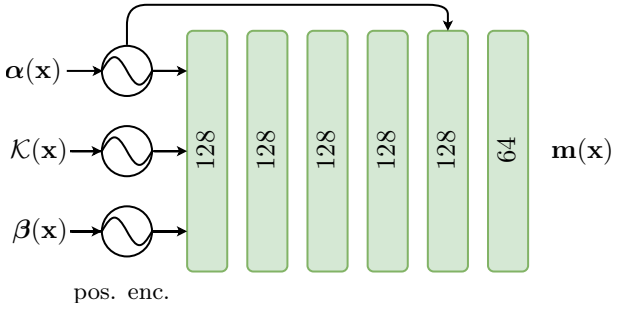


Figure 15. Masking network $\mathcal{M}$ take lifted attributes $\boldsymbol{\alpha}(\mathbf{x})$, lifted latent code $\boldsymbol{\beta}(\mathbf{x})$ and canonicalized points $\mathcal{K}(\mathbf{x})$. We transform $\boldsymbol{\alpha}(\mathbf{x})$ and $\boldsymbol{\beta}(\mathbf{x})$ through a windowed positional encoding where we start at 1k-th step linearly increasing a single sine component for the next 10k steps. Points $\mathcal{K}(\mathbf{x})$ are encoded with 8 components. The output is activated with a sigmoid function. We realize $\mathbf{m}_0(\mathbf{x})$ as $\mathbf{m}(\mathbf{x})_0 = 1 - \sum_{a \in A} \mathbf{m}_a(\mathbf{x})$, and clip the output to ensure the values range to be in $[0, 1]$. Note that while the network shares similarities with the radiance field prediction part $\mathcal{R}$, it is not conditioned on view directions and appearance codes.

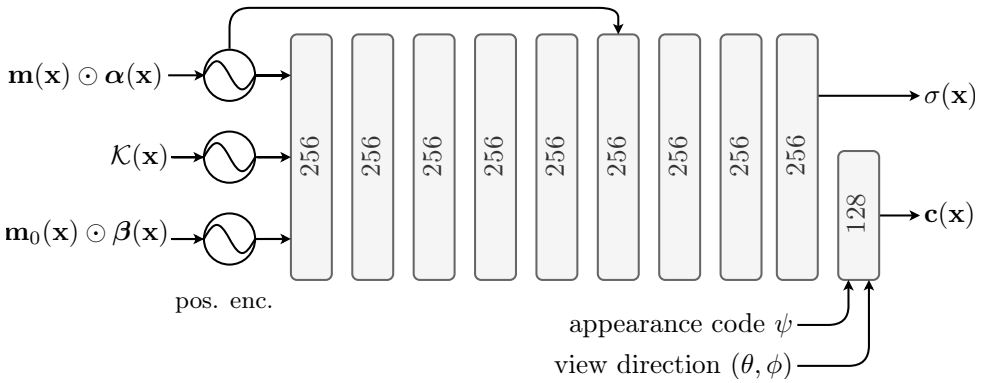


Figure 16. The radiance field prediction network predicts RGB colors $\mathbf{c}(\mathbf{x})$ and density values $\sigma(\mathbf{x})$ from canonicalized points. We encode points $\mathbf{x}$ with 8 sine components and linearly increase contribution of a single component in $\boldsymbol{\alpha}(\mathbf{x})$ and $\boldsymbol{\beta}(\mathbf{x})$ from 1k to 11k step. Per-point predicted predicted attributes $\boldsymbol{\alpha}(\mathbf{x})$ and lifted latent code $\boldsymbol{\beta}(\mathbf{x})$ are masked by a mask predicted from the masking network depicted in Fig. 15. The final linear layer takes additional per-image learnable appearance code ψ to account for any visual variations that cannot be explained by the rest of the framework (*e.g.* changes in lighting). The code can discarded during evaluation. The same layer is additionally conditioned on the positionally encoded view directions. We activate the color output with a sigmoid function.

	NeRF [72]	NeRFies [83]	HyperNeRF [84]	NeRFace [26]	NHA [31]	AVA [13]	VolTeMorph [28]	**Ours**
Applicability beyond faces	✓	✓	✓	✗	✗	✗	✓	✓
Interpretable control	✗	✗	✗	✓	✓	✗	✓	✓
Data efficiency	✗	✓	✓	✗	✓	✗	✓	✓
Expression-dependent changes	✗	✗	✓	✓	✓	✓	✗	✓
Generalizability to unknown expressions	✗	✗	✗	✓	✓	✗	✓	✓

Table 4. Comparison – We compare several methods to our approach. Other methods fall short in data efficiency and applicability. For example, AVA [13] requires 3.1 million training images while VolTeMorph [28] cannot model expression-dependent wrinkles realistically.

Chapter 4

BlendFields: Few-Shot Example-Driven Facial Modeling

4.1 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.

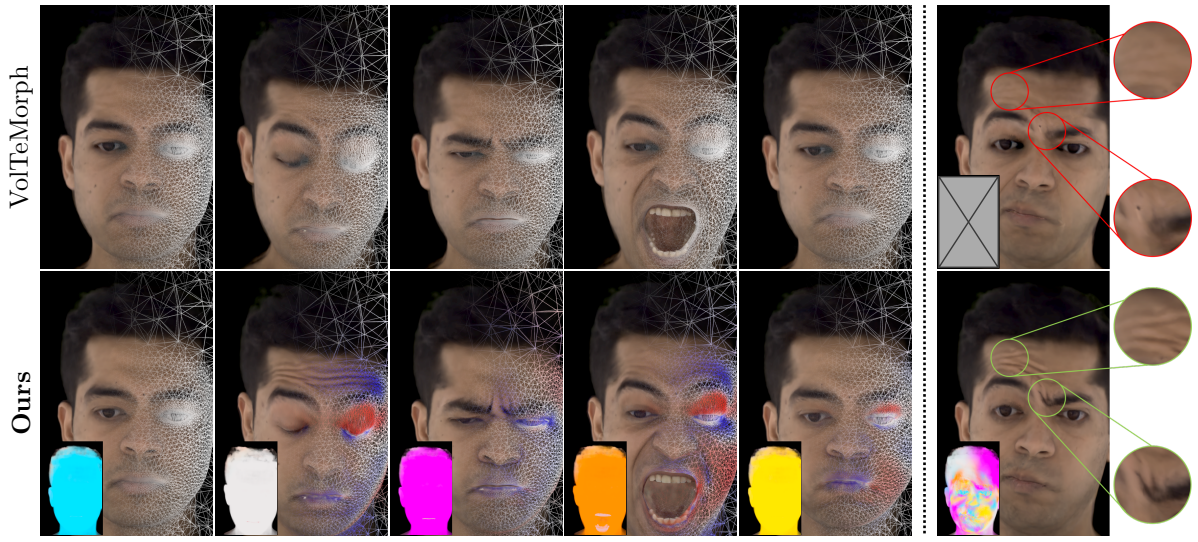


Figure 17. Teaser – Given five multi-view frames of different expressions, our approach generates a model capable of capturing the fine-grained details of a novel expression beyond the resolution of the underlying face model [28] (top right corner). This is achieved by *blending* the radiance fields 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).

4.2 Introduction

Recent advances in neural rendering of 3D scenes [108] offer 3D reconstructions of unprecedented quality [72] with an ever-increasing degree of control [50, 62]. Human faces are of particular interest to the research community [4, 26–28] due to their application in creating realistic digital doubles [67, 108, 138, 147].

To render facial expressions not observed during training, current solutions [4, 26–28] rely on *parametric* face models [10]. These allow radiance fields [72] to be controlled by facial parameters estimated by off-the-shelf face trackers [58]. 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) [13] 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 [**ggarbin2024voltemorphxu2022deforming**, 137]. In particular, we extend VolTeMorph [28] to render facial details learned from images of a sparse set of expressions. To achieve this, we draw inspiration from blend-shape correctives [56], 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 [28], 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 [28]; see **??**. We call our method *BlendFields* since it resembles the way blend shapes are employed in 3DMMs [10]. 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 [42, 88]. Using the dynamic geometry of [28], 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 **??**, and our empirical evaluations confirm these advantages. In summary, we:

- extend VolTeMorph [28] to enable modeling of high-frequency information, such as expression wrinkles in a few-shot setting;

- introduce correctives [10] to neural field representations and activate them according to local deformations [88];
- make this topic more accessible with an alternative to techniques that are data and compute-intensive [13];
- show that our model generalizes beyond facial modeling, for example, in the modeling of wrinkles on a deformable object made of rubber.

4.3 Related Works

Neural Radiance Fields (NeRF) [72] is a method for generating 3D content from images taken with commodity cameras. It has prompted many follow-up works [7, 8, 40, 68, 71, 83, 90, 101, 105, 113, 127, 140] 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 [29, 36, 134]. Works solving the limitation to static scenes[4, 5, 44, 78, 117, 119, 126, 145] and adding explicit control [16, 50, 53, 102, 115, 131, 137] 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 [10]) or a conditioning signal [84]. 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.

4.3.1 Radiance Fields

Volumetric representations [114] have grown in popularity because they can represent complex geometries like hair more accurately than mesh-based ones. Neural Radiance Fields (NeRFs) [72] 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 $C(\mathbf{r})$:

$$C(\mathbf{r}) = \int_{t_n}^{t_f} T(t)\, \sigma(\mathbf{r}(t))\, \mathbf{c}(\mathbf{r}(t), \mathbf{v}) dt, \tag{21}$$

where t_n and t_f are near and far planes, and

$$T(t) = \exp\left(-\int_{t_n}^{t} \sigma(\mathbf{r}(s))ds\right), \tag{22}$$

is the transmittance function [104]. 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 [**directVoxelOptimisation**, 29, 36, 60, 134].

4.3.2 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 [86], but it cannot generalize to unseen deformations. Several works [26, 83, 84, 110] 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 [4, 26, 27, 148] use parameters of a face model to condition NeRF's MLP, or learn a latent space of images and geometry [13, 63, 64, 67, 69, 116] 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 [28], 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. 17.

Similarly, NeRF-Editing [137] and NeRF Cages [130] 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`.

4.3.3 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 [28, 130, 131]. For example, VolTeMorph uses dense landmarks [122] to fit a parametric face model whose blendshapes have been extended to a tetrahedral cage with finite elements method [17]. These cages can be quickly deformed and raytraced [73] using parallel computation on GPUs [19] while driving the volume into the target

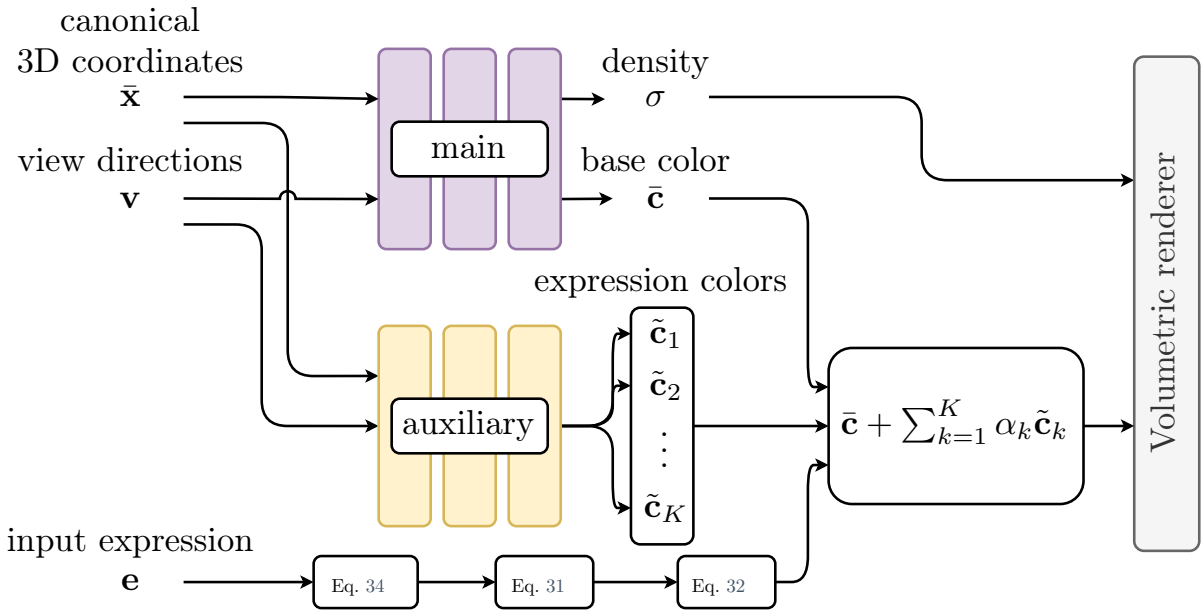


Figure 18. BlendFields – We implement our approach as a volumetric model, where the *appearance* (*i.e.* radiance) is the sum of the main appearance corrected by blending a small set of K expression-specific appearances. These appearances are learnt from extreme expressions, and then blended at test-time according to blend weights computed as a function of the input expression $\mathbf{e}$.

pose and allowing early ray termination for fast rendering. We further leverage the tetrahedral cage and use its differential properties [42], 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.

4.4 Method

We introduce a volumetric model that can be driven by input expressions and visualize it in in Fig. 18. We start this section by explaining our model and how we train and drive it with novel expressions utilizing parametric face models (Sec. 4.4.1). We then discuss how to compute measures of volume expansion and compression in the tetrahedra to combine volumetric models of different expressions (Sec. 4.4.2) and how we remove artifacts in out-of-distribution settings (Sec. 4.4.3). We conclude this section with implementation details (Sec. 4.4.4).

4.4.1 Our model

Given a neutral expression $\bar{\mathbf{e}}$, and a collection of posed images $\{C_c\}$ of this expression from multiple views, VolTeMorph [**garbin2022voltemorph**] 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 $\bar{\mathbf{e}}$:

$$\mathbf{c}(\mathbf{x};\mathbf{e}) = \bar{\mathbf{c}}(\bar{\mathbf{x}}), \quad \bar{\mathbf{x}} = \mathcal{T}(\mathbf{x};\mathbf{e}\rightarrow\bar{\mathbf{e}}) \tag{23}$$

$$\sigma(\mathbf{x};\mathbf{e}) = \bar{\sigma}(\bar{\mathbf{x}}), \quad \bar{\mathbf{x}} = \mathcal{T}(\mathbf{x};\mathbf{e}\rightarrow\bar{\mathbf{e}}) \tag{24}$$

$$\ell_{\text{rgb}} = \mathbb{E}_{C\sim\{C_c\}}\, \mathbb{E}_{\mathbf{r}\sim C}\, \ell^{\mathbf{r}}_{\text{rgb}} \tag{25}$$

$$\ell^{\mathbf{r}}_{\text{rgb}} = \|C(\mathbf{r};\mathbf{e}) - C(\mathbf{r})\|_2^2, \tag{26}$$

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 [58, 120]. 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):

$$\mathbf{c}(\mathbf{x};\mathbf{e}) = \bar{\mathbf{c}}(\mathbf{x}) + \sum_{k=k}^{K} \alpha_k(\mathbf{x};\mathbf{e}) \cdot \tilde{\mathbf{c}}_k(\mathbf{x}), \tag{27}$$

We call our model *blend fields*, as it resembles the way in which blending is employed in 3D morphable models [10] or in wrinkle maps [80]. 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 Eq. (24). 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 $\{\mathbf{C}_k\}$ (example in Fig. 19), each corresponding to an "extreme" expression $\{\mathbf{e}_k\}$, and minimize the loss:

$$\ell_{\text{rgb}} = \mathbb{E}_k\, \mathbb{E}_{\mathbf{r}}\, \|C_K(\mathbf{r};\mathbf{e}_k) - C_k(\mathbf{r})\|_2^2 \tag{28}$$

$$\text{where} \quad \forall\mathbf{x},\ \boldsymbol{\alpha}(\mathbf{x}) = \mathbb{1}_k, \tag{29}$$

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 Eq. (27) 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

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

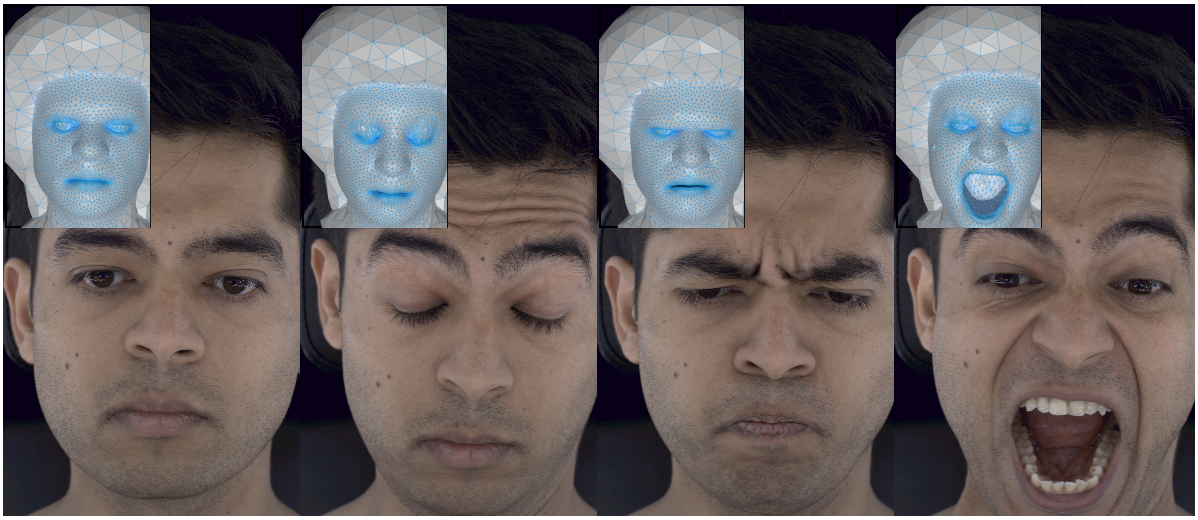


Figure 19. Data – We represent the data as a multi-view, multi-expression images. For each of these images, we obtain parameters of a parametric model, such as FLAME [58] to get: an expression vector $\mathbf{e}$ and a tetrahedral mesh described by vertices $\mathbf{V}(\mathbf{e})$. We highlight that our approach works for any object if a rough mesh and its descriptor are already provided.

tetrahedra using linear FEM bases [74]. 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. 4.4.2, then

$$\mathcal{G}(\mathbf{v}(\mathbf{e}))\approx\mathcal{G}(\mathbf{v}(\mathbf{e}_k)) \implies \boldsymbol{\alpha}(\mathbf{v}(\mathbf{e})) \approx \mathbb{1}_k. \tag{30}$$

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

$$[\Delta\mathcal{G}_k(\mathbf{v}(\mathbf{e}))] = [\|\mathcal{G}(\mathbf{v}(\mathbf{e}))-\mathcal{G}(\mathbf{v}(\mathbf{e}_k))\|_2^2] \in \mathbb{R}^K \tag{31}$$

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

$$\boldsymbol{\alpha}(\mathbf{v}(\mathbf{e})) = \mathrm{softmax}_\tau\{\Delta\mathcal{G}_k(\mathbf{v}(\mathbf{e}))\} \in [0,1]^K \tag{32}$$

which realizes Eq. (30), 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 [41].

4.4.2 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 [42]:

$$\Delta\mathcal{V}(\mathbf{T}) = \det(\mathbf{D} \cdot \bar{\mathbf{D}}^{-1}) \tag{33}$$

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:

$$\mathcal{G}(\mathbf{v}(\mathbf{e})) = \bigoplus_{\mathbf{T}\in\mathcal{N}(\mathbf{v})} \Delta\mathcal{V}(\mathbf{T}(\mathbf{e})), \tag{34}$$

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

4.4.3 Blend-field smoothness

High-frequency spatial changes in blendfields can cause visual artifacts, see Fig. 20. 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 [42]. 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 [21]:

$$\mathbf{A}^{\text{diff}} = (\mathbf{I} - \lambda_{\text{diff}}\mathbf{L})^{-1}\mathbf{A}^{n}. \tag{35}$$

4.4.4 Implementation details

We build on VolTeMorph [28] 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 [120] 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 [120], can be estimated by a real-time face tracking system like [**wft**], 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

Method	Real Data						Synthetic Data		
	Casual Expressions			Novel Pose Synthesis			Novel Pose Synthesis		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
NeRF [72]	23.6465	0.7384	0.2209	25.6696	0.8127	0.1861	13.7210	0.6868	0.3113
Conditioned NeRF [72]	22.9106	0.7162	0.2029	24.7283	0.7927	0.1682	19.5971	0.8138	0.1545
NeRFies [83]	22.6571	0.7105	0.2271	24.8376	0.7990	0.1884	19.3042	0.8081	0.1591
HyperNeRF-AP [84]	22.6219	0.7087	0.2236	24.7119	0.7931	0.1848	19.3557	0.8132	0.1563
HyperNeRF-DS [84]	22.9299	0.7182	0.2241	24.9909	0.8007	0.1860	19.4637	0.8159	0.1526
VolTeMorph$_1$ [28]	24.9939	0.8358	0.1164	26.7526	0.8749	0.0954	26.7033	0.9500	0.0394
VolTeMorph$_{avg}$[28]	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	32.7949	0.9882	0.0221

Table 5. 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 [121]. 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$_{avg}$ [28] 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.* [84]. 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.

sample N_{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. (28) and sparsity losses with standard weights used in VolTeMorph [28, 36]. We train the methods for 5×10^5 steps using Adam [54] 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 (**??**), we consider $|\mathcal{N}(\mathbf{v})|$=20 neighbors. For the Laplacian smoothing, we set λ_{diff}=0.1 and perform a single iteration step. Geometric-related operations impose negligible computational overhead.

4.5 Experiments

We evaluate all methods on data of four subjects from the publicly available Multiface dataset [124]. 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) [6], Structural Similarity Index (SSIM) [118] and Learned Perceptual Image Patch Similarity (LPIPS) [142] to measure the performance of the models. Each of the rendered images has a resolution of 334×512 pixels.

[2] To train BlendFields for a single subject we use ≈ 0.006% of the dataset used by AVA [13].

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

Table 6. 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 Sec. 4.5.4.

As baselines, we use the following approaches: the original, static NeRF [72], NeRF conditioned on an expression code concatenated with input points $\mathbf{x}$, NeRFies [83], HyperNeRF[3] [84], and VolTeMorph [28]. 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 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 ($\text{VolTeMorph}_{\text{avg}}$)[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.

4.5.1 Realistic Human Captures

Novel expression synthesis. We extract eight multi-view frames from the Multiface dataset [124], 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. 21 we show that BlendFields is the only method capable of rendering convincing wrinkles dynamically, depending on the input expression. BlendFields performs favorably compared to the baselines (see Tab. 5).

[3] We use two architectures proposed by Park *et al.* [84].

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

[5] We do not compare to NeRFace [26] and NHA [31] as VolTeMorph [28] performs better quantitatively than these methods.

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. 5. Our approach performs best all the settings. See animations in the `Supplementary` for a comparison of rendered frames across all methods.

4.5.2 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 mesh directly from Houdini [129], which we use for wrinkle simulation, and render the images in Blender [18]. We show quantitative results in Tab. 5 for the bending cylinder, and a comparison of the inferred images in Fig. 22 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.

4.5.3 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 **??**. 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. 20). We additionally present in the `Supplementary` how the number of expressions used for training affects the results.

4.5.4 Failure Cases

While BlendFields offers significant advantages for rendering realistic and dynamic high-frequency details, it falls short in some scenarios (see Fig. 23). One of the issues arises 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

[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.

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.

4.6 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`.

4.7 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).

CoNeRF: Controllable Neural Radiance Fields

Supplementary Material

4.8 Potential social impact

Our motivation for this work was to enable the creation of 3D avatars that could be used as communication devices in the remote working era. As our approach stems from blendshapes [56], these avatars are easily adjustable via texture coloring and may be used for entertainment. We note, however, that the potential misuse of our work includes using it as deep fakes. We highly discourage such usage. One of our future directions includes detecting fake images generated by our method. At the same time, we highlight the importance of BlendFields—in the presence of closed technologies [13, 67], it is crucial to democratize techniques for personalized avatar creation. We achieve that by limiting the required data volume to train a single model. As history shows, when given an open, readily available technology for generative modeling of

# expr.	Casual Expressions			Novel Pose Synthesis		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
K=1	27.5834	0.9028	0.0834	28.7589	0.9147	0.0806
K=2	27.6783	0.9026	0.0856	29.2859	0.9186	0.0803
K=3	27.9137	0.9054	0.0819	29.8551	0.9279	0.0728
K=4	27.8140	0.9055	0.0815	30.1543	0.9336	0.0701
K=5	28.0254	0.9110	0.0778	30.4721	0.9372	0.0688
K=6	28.0517	0.9091	0.0813	–	–	–
K=7	28.2004	0.9115	0.0823	–	–	–
K=8	28.2542	0.9124	0.0830	–	–	–

Table 7. Number of training expressions – We ablate over the number of training expressions. We evaluate the model on the captures from the Multiface dataset [124]. We run the model for each possible expression combination for a given K and average the results. The best results are colored in ■ and the second best in ■. Increasing the number of available training expressions consistently improves the results. However, using K=5 expressions saturates the quality and using K>5 brings diminishing improvements. We do not report "Novel Pose Synthesis" for K>5 as we use validation expressions and poses to train those models (refer to Sec. 4.5.1 for more details).

images [92], users can scrutinize it with unprecedented thoroughness, thus raising the general awareness of potential misuses.

4.9 Concurrent Works

Gao *et al.* [27] and Xu *et al.* [**xu2022manvatar**] also use an interpolation between known expressions to combine multiple neural radiance fields trained for those expressions. However, their approach interpolates between grids of latent vectors [76] globally. The interpolation weights are taken from blendshape coefficients.

Zielonka *et al.* [150] use a parametric head model to canonicalize 3D points similarly to our ends. However, instead of building a tetrahedral cage around the head, they smoothly assign each face triangle to 3D points. Then they canonicalize points using transformations that each of the assigned triangles undergoes for a given expression. They concatenate 3D points with the expression code from FLAME [58] to model expression-dependent effects.

Method	Casual Expressions			Novel Pose Synthesis		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
NeRF [72]	22.0060	0.6556	0.3222	23.8077	0.7448	0.2779
Conditioned NeRF [72]	21.0846	0.6280	0.3042	22.9991	0.7261	0.2362
NeRFies [83]	20.7004	0.6076	0.3579	23.0123	0.7253	0.2840
HyperNeRF-AP [84]	20.8105	0.6214	0.3504	22.8193	0.7185	0.2689
HyperNeRF-DS [84]	20.8847	0.6111	0.3656	23.0075	0.7259	0.2729
VolTeMorph$_1$ [28]	21.3265	0.7091	0.2706	22.3007	0.7795	0.2281
VolTeMorph$_{\text{avg}}$[**garbin2022voltemorph**]	22.0759	0.7755	0.2615	23.8974	0.8458	0.2302
BlendFields	22.8982	0.7954	0.2256	24.4432	0.8477	0.2052

Table 8. Quantitative results without masking – Similarly to Tab. 5, we compare BlendFields to other related approaches. However, we calculate the results over the whole image space, without removing the background. BlendFields and VolTeMorph [28] model the background as a separate NeRF-based [72] network. The points that do not fall into the tetrahedral mesh are assigned to the background. As the network overfits to sparse training views, it poorly extrapolates to novel expressions (as the new head pose or expression may reveal some unknown parts of the background) and views. At the same time, all other baselines do not have any mechanism to disambiguate the background and the foreground.

4.10 Additional results

4.10.1 Ablating number of expressions

We ablate over the number of used expressions during the training. To evaluate the effect of the number of expressions, we add consecutive frames to the training set (starting from a single, neutral one), *i.e.*, the training set has $k<K$ expressions. We train BlendFields for such a set for each subject separately. We then average the results for a given k across subjects. We present the results in **??**. When selecting the training expressions, we aim to choose those that show all wrinkles when combined. We can see from Fig. 25 that if removed, *e.g.*, the expressions with eyebrows raised, then the model cannot render wrinkles on the forehead. In summary, increasing the number of expressions improves the quality results with diminishing returns when $K>5$, while $K=5$ provides a sufficient trade-off between the data capture cost and the quality.

4.10.2 Training frames

We present in Fig. 24 example training frames for one of the subjects. Each frame is a multi-view frame captured with ≈35 cameras (the number of available cameras varied slightly between subjects).

4.10.3 Quantitative results with background

We compare BlendFields and the baselines similarly to **??**. However, in this experiment, we deliberately include the background in metric calculation. We show the results in Tab. 8. In all the cases, BlendFields performs best even though the method was not designed to model the

background accurately. Additionally, as HyperNeRF [84], NeRFies [83], and NeRF [72] do not have any mechanism to disambiguate between the foreground and the background, the metrics are significantly worse when including the latter.

4.10.4 Additional qualitative results

We show in **??** results of baselines that do not rely on parametric models of the face [58]. Compared to BlendFields, they cannot render high-fidelity faces. The issue comes from the assumed data sparsity—those approaches rely on the interpolation in the training data. As we assume access to just a few frames, there is no continuity in the training data that would guide them to interpolate between known expressions. BlendFields presents superior results given novel expressions even with such a sparse dataset. results.

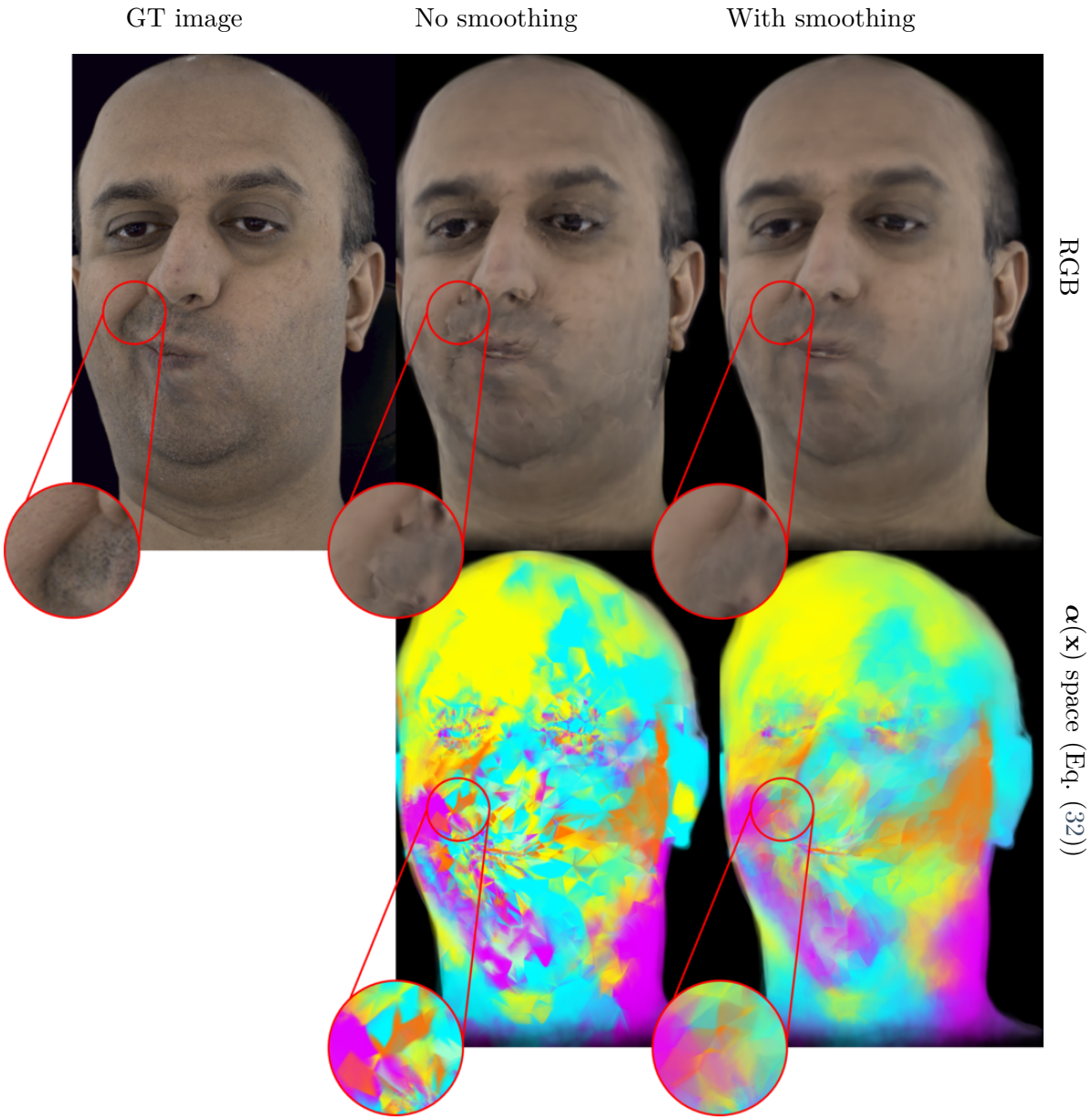


Figure 20. Laplacian smoothing – To combat artifacts stemming from calculating weights α across multiple expressions, which may assign different expressions to neighboring tetrahedra, we apply Laplacian smoothing [21]. As seen in the bottom row, smoothing gives a more consistent expression assignment.

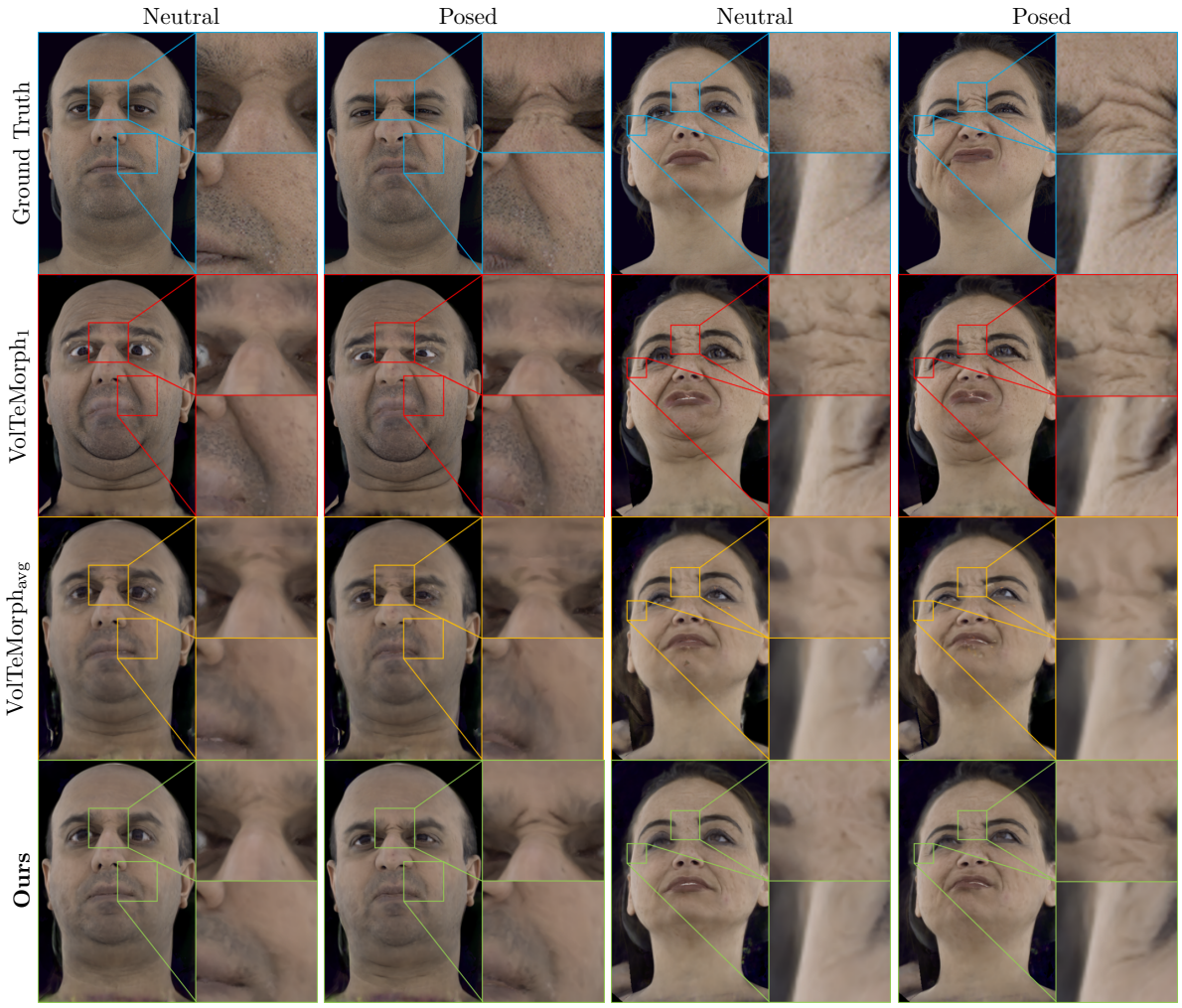


Figure 21. Novel expression synthesis – We compare qualitatively BlendFields with selected baselines (vertical) across two selected subjects (horizontal). Firstly, we show a neutral pose of the subject and then any of the available expressions. To our surprise, VolTeMorph$_{avg}$ trained on multiple frames renders some details but with much lower fidelity. We argue that VolTeMorph$_{arg}$ considers rendering wrinkles as artifacts that depend on the view direction (see **??**). 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.

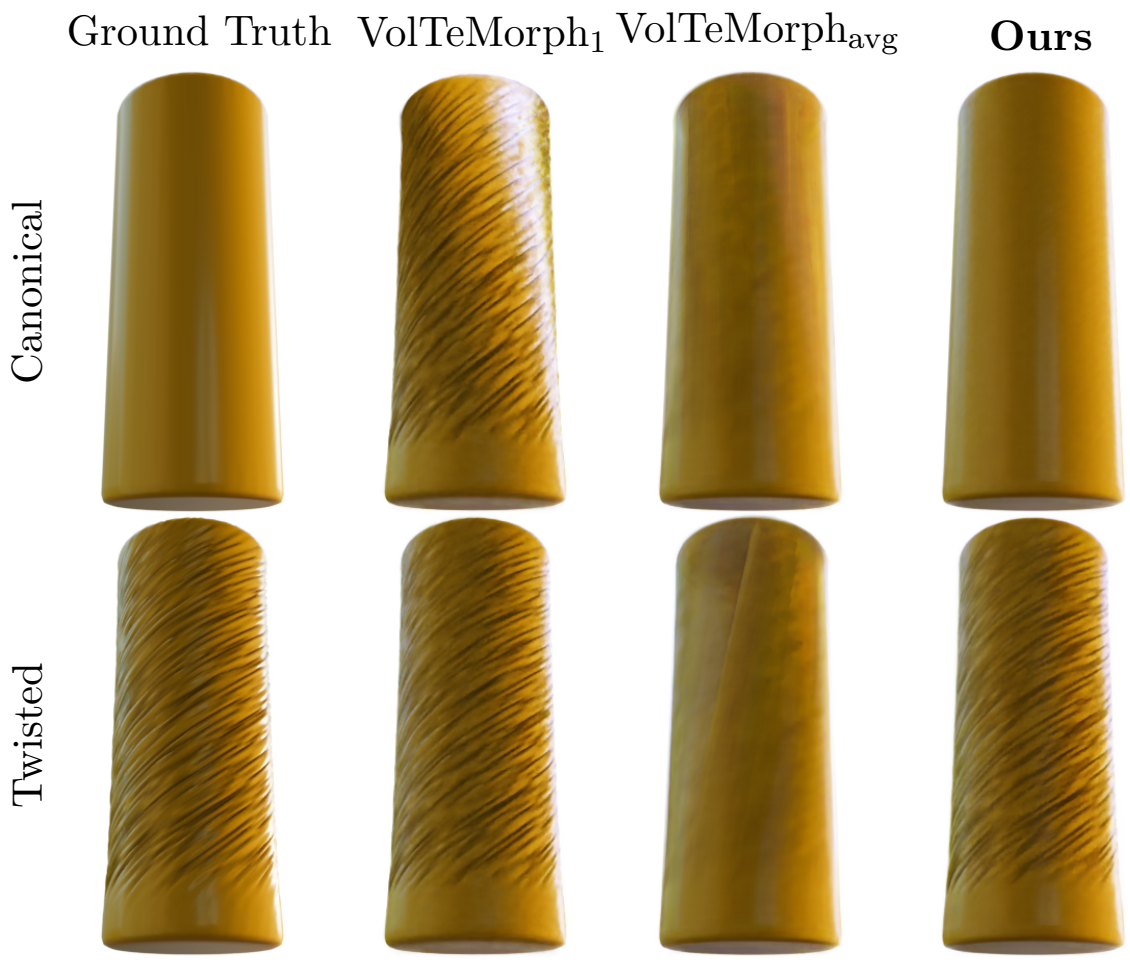


Figure 22. Qualitative results on synthetic dataset – For a simple dataset, baselines cannot model high-frequency, pose-dependent details. VolTeMorph$_1$ renders wrinkles for the straight pose as well, as it is trained for the twisted cylinder only, while VolTeMorph$_{avg}$ averages out the texture.

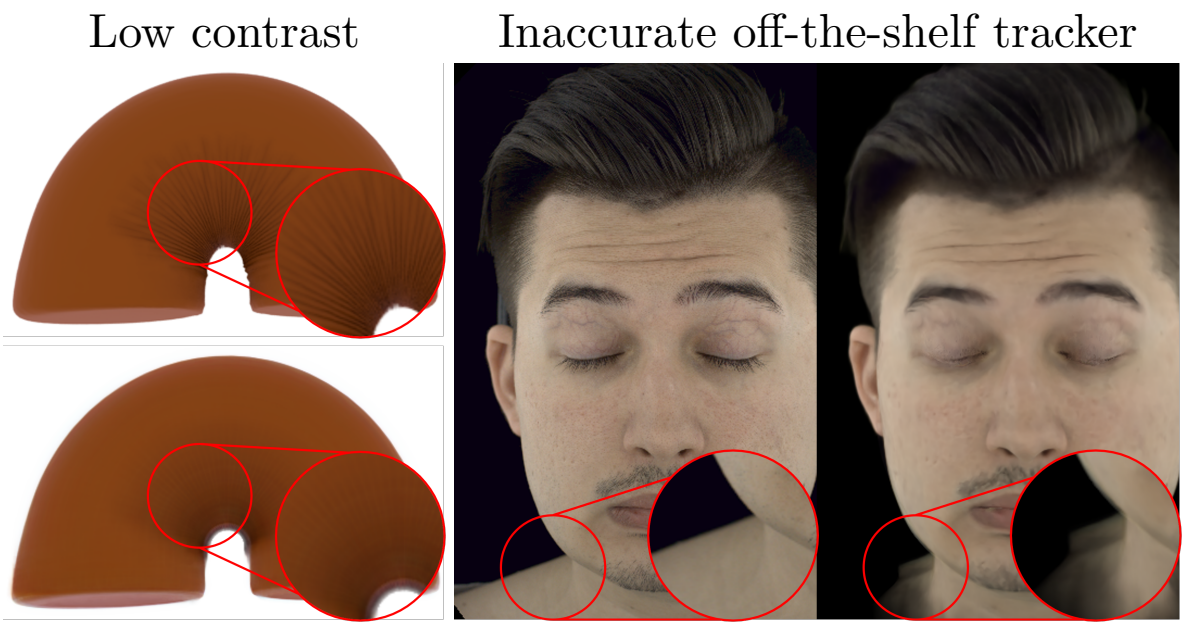


Figure 23. Failure cases – We show failure cases for our proposed approach. *Left:* In the presence of wrinkles in low-contrast images, BlendFields takes longer to converge to make wrinkles visible. We show the ground truth on the top, and rendering after training $7{\times}10^5$ steps on the bottom. In contrast, we rendered images in Fig. 22 after $2{\times}10^5$ steps. *Right:* BlendFields inherits issues from VolTeMorph [28], which relies on the initial fit of the face mesh. If the fit is inaccurate, artifacts appear in the final render.

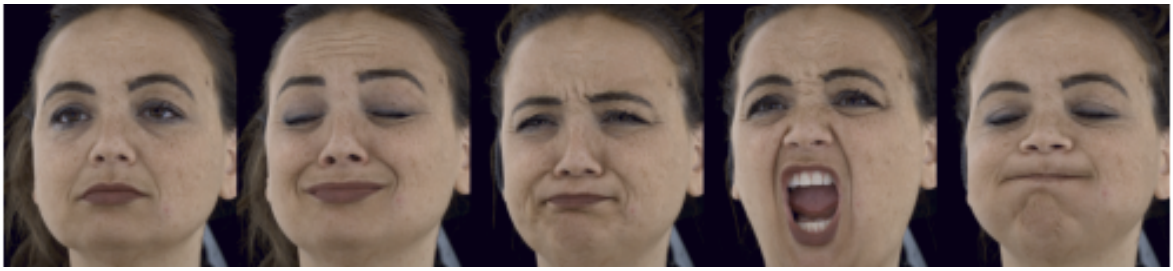


Figure 24. Training frames – In Sec. 4.5, we show results for the BlendFields trained on $K{=}5$ expressions. The images represent these expressions for one of the subjects. For each subject, we selected similar expressions to show all possible wrinkles when combined. Please note that we also include a "neutral" expression (the first from the left)—it is necessary to enable the learning of a face without any wrinkles.

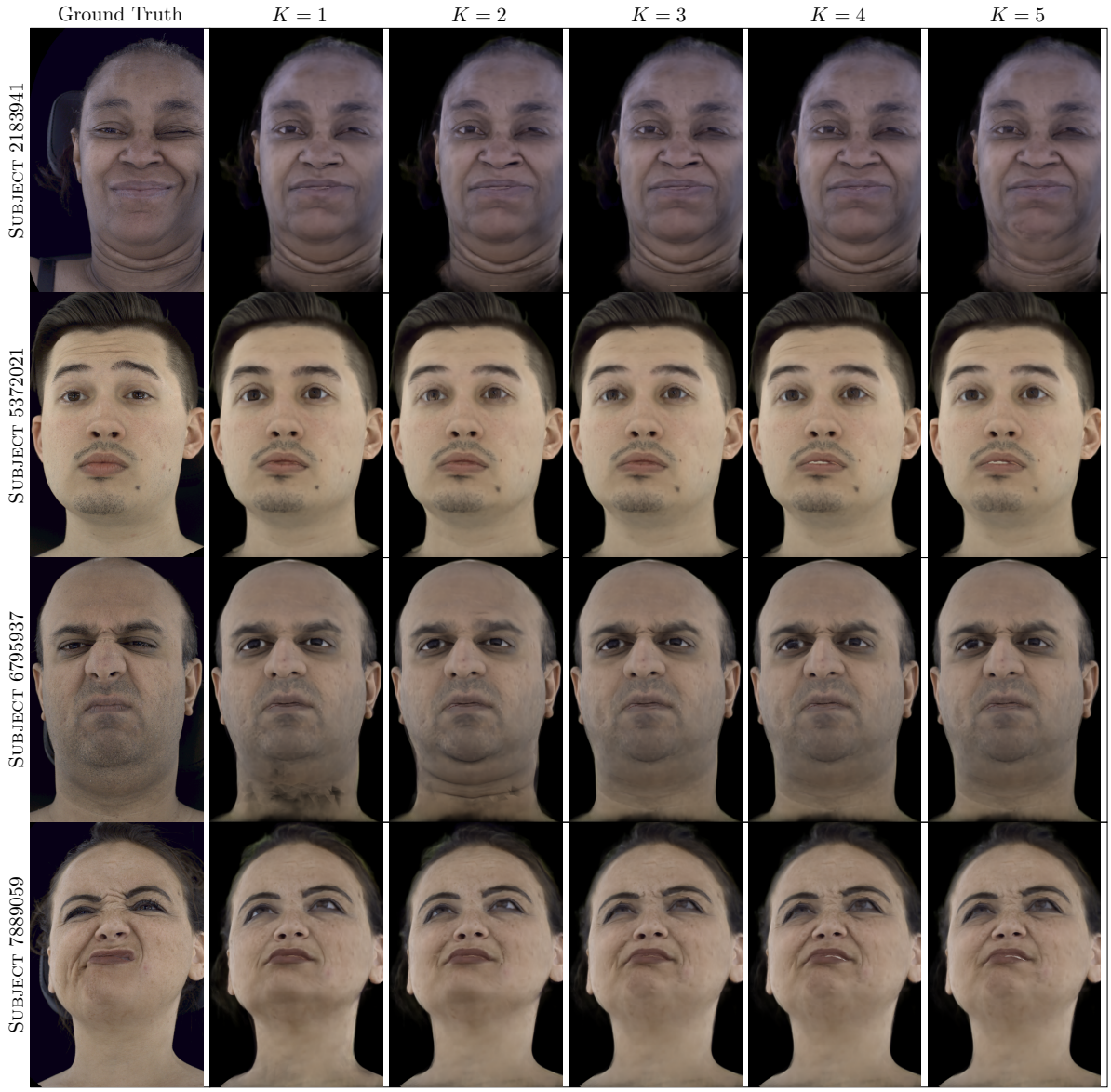


Figure 25. Qualitative ablation over the number of training expressions – We show qualitatively how the number of training expressions K affects the rendering quality. The first row shows the ground truth images. All other consecutive rows show the images rendered with BlendFields while increasing the number of training expressions. The last row, K=5 corresponds to the results presented in the main part of the article. The subject's naming follows the convention introduced in the Multiface repository [124]. Please refer to Tab. 7 for quantitative results.

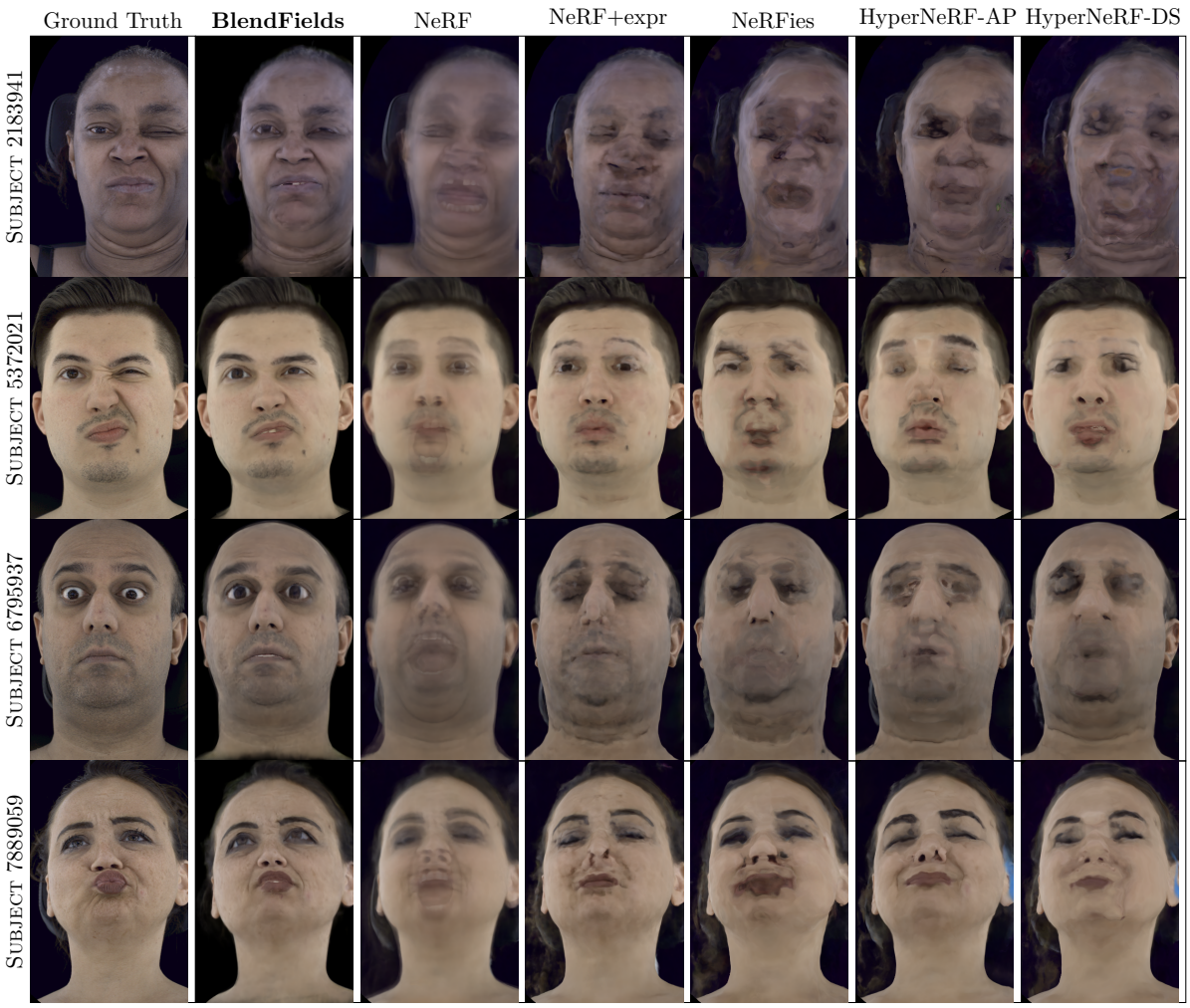


Figure 26. Comparison to strictly data-driven approaches – We compare BlendFields to other baselines that do not rely on mesh-driven rendering: NeRF [72], NeRF conditioned on the expression code (NeRF+expr) [72], NeRFies [83], and HyperNeRF-AP/DS [84]. As a static model, NeRF converges to an average face from available (K=5) expressions. All other baselines exhibit severe artifacts compared to BlendFields. Those baselines rely on the data continuity in the training set (*e.g.*, from a video), and cannot generalize to any other expression.

Chapter 5

Final remarks and discussion

5.1 Conclusions

5.2 Future work

Bibliography

[1] Agarwala, A., Dontcheva, M., Agrawala, M., Drucker, S., Colburn, A., Curless, B., Salesin, D., and Cohen, M., "Interactive Digital Photomontage," in *ACM SIGGRAPH 2004 Papers*, ser. SIGGRAPH '04, Los Angeles, California: Association for Computing Machinery, 2004, pp. 294–302, ISBN: 9781450378239. DOI: 10.1145/1186562.1015718. [Online]. Available: https://doi.org/10.1145/1186562.1015718 (cit. on p. 20).

[2] Alldieck, T., Xu, H., and Sminchisescu, C., "imGHUM: Implicit Generative Models of 3D Human Shape and Articulated Pose," in *CVPR*, 2021 (cit. on p. 22).

[3] Asnani, V., Yin, X., Hassner, T., and Liu, X., "Reverse Engineering of Generative Models: Inferring Model Hyperparameters from Generated Images," 2021 (cit. on p. 36).

[4] Athar, S., Xu, Z., Sunkavalli, K., Shechtman, E., and Shu, Z., "RigNeRF: Fully Controllable Neural 3D Portraits," in *CVPR*, 2022, pp. 20 364–20 373 (cit. on pp. 43–45).

[5] Attal, B., Laidlaw, E., Gokaslan, A., Kim, C., Richardt, C., Tompkin, J., and O'Toole, M., "TöRF: Time-of-Flight Radiance Fields for Dynamic Scene View Synthesis," *NeurIPS*, vol. 34, pp. 26 289–26 301, 2021 (cit. on p. 44).

[6] Avcibas, I., Sankur, B., and Sayood, K., "Statistical evaluation of image quality measures," *Journal of Electronic Imaging*, vol. 11, no. 2, pp. 206–223, 2002 (cit. on p. 50).

[7] Barron, J., Mildenhall, B., Tancik, M., Hedman, P., Martin-Brualla, R., and Srinivasan, P., "Mip-NeRF: A Multiscale Representation for Anti-aliasing Neural Radiance Fields," in *ICCV*, 2021, pp. 5855–5864 (cit. on p. 44).

[8] Barron, J., Mildenhall, B., Verbin, D., Srinivasan, P., and Hedman, P., "Mip-NeRF 360: Unbounded Anti-aliased Neural Radiance Fields," in *CVPR*, 2022, pp. 5470–5479 (cit. on p. 44).

[9] Belharbi, S., Ben Ayed, I., McCaffrey, L., and Granger, E., "Deep Active Learning for Joint Classification & Segmentation with Weak Annotator," 2021 (cit. on p. 35).

[10] Blanz, V. and Vetter, T., "A Morphable Model For The Synthesis Of 3D Faces," in *CGIT*, 1999, pp. 187–194 (cit. on pp. 20, 43, 44, 47).

[11] Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., and Zhang, Q., *JAX: Composable Transformations of Python+NumPy Programs*, http://github.com/google/jax, version 0.2.5, 2018 (cit. on p. 26).

[12] *Bunny 3D model*, https://graphics.stanford.edu/~mdfisher/Data/Meshes/bunny.obj, Accessed: 2021-11-16 (cit. on p. 28).

[13] Cao, C., Simon, T., Kim, J. K., Schwartz, G., Zollhoefer, M., Saito, S.-S., Lombardi, S., Wei, S.-E., Belko, D., Yu, S.-I., *et al.*, "Authentic Volumetric Avatars from a Phone Scan," *ToG*, vol. 41, no. 4, pp. 1–19, 2022 (cit. on pp. 11, 41, 43–45, 50, 53).

[14] Chen, X., Duan, Y., Houthooft, R., Schulman, J., Sutskever, I., and Abbeel, P., "InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets," in *NeurIPS*, 2016 (cit. on p. 20).

[15] Chen, X., Zhang, Q., Li, X., Chen, Y., Feng, Y., Wang, X., and Wang, J., "Hallucinated Neural Radiance Fields in the Wild," in *CVPR*, 2022, pp. 12 943–12 952 (cit. on p. 11).

[16] Cheng, Z., Chai, M., Ren, J., Lee, H.-Y., Olszewski, K., Huang, Z., Maji, S., and Tulyakov, S., "Cross-Modal 3D Shape Generation and Manipulation," in *ECCV*, Springer, 2022, pp. 303–321 (cit. on p. 44).

[17] Clough, R. W., "The Finite Element Method in Plane Stress Analysis," in *Conference on Electronic Computation*, 1960 (cit. on p. 45).

[18] Community, B. O., *Blender - a 3d modeling and rendering package*, Blender Foundation, Stichting Blender Foundation, Amsterdam, 2022. [Online]. Available: http://www.blender.org (cit. on p. 52).

[19] Cook, S., *CUDA Programming: A Developer's Guide to Parallel Computing with GPUs*, 1st. 2012, ISBN: 9780124159334 (cit. on p. 45).

[20] Deng, B., Lewis, J. P., Jeruzalski, T., Pons-Moll, G., Hinton, G., Norouzi, M., and Tagliasacchi, A., "NASA: Neural Articulated Shape Approximation," in *ECCV*, 2020 (cit. on p. 22).

[21] Desbrun, M., Meyer, M., Schröder, P., and Barr, A. H., "Implicit Fairing of Irregular Meshes using Diffusion and Curvature Flow," in *CGIT*, 1999, pp. 317–324 (cit. on pp. 49, 57).

[22] Ekman, P. and Rosenberg, E. L., *What the Face Reveals: Basic and Applied Studies of Spontaneous Expression Using the Facial Action Coding System (FACS)*. Oxford University Press, USA, 1997 (cit. on p. 20).

[23] Esposito, S., Xu, Q., **Kania**, **K.**, Hewitt, C., Mariotti, O., Petikam, L., Valentin, J., Onken, A., and Mac Aodha, O., "GeoGen: Geometry-Aware Generative Modeling via Signed Distance Functions," in *CVPRW*, 2024, pp. 7479–7488 (cit. on p. 16).

[24] Fang, J., Yi, T., Wang, X., Xie, L., Zhang, X., Liu, W., Nießner, M., and Tian, Q., "Fast Dynamic Radiance Fields with Time-Aware Neural Voxels," in *SIGGRAPH Asia 2022 Conference Papers*, 2022, pp. 1–9 (cit. on p. 10).

[25] Feng, Y., Feng, H., Black, M. J., and Bolkart, T., "Learning an Animatable Detailed 3D Face Model from In-The-Wild Images," *ToG*, vol. 40, no. 4, pp. 1–13, 2021 (cit. on p. 10).

[26] Gafni, G., Thies, J., Zollhofer, M., and Nießner, M., "Dynamic Neural Radiance Fields for Monocular 4d Facial Avatar Reconstruction," in *CVPR*, 2021, pp. 8649–8658 (cit. on pp. 19, 21, 41, 43, 45, 51).

[27] Gao, X., Zhong, C., Xiang, J., Hong, Y., Guo, Y., and Zhang, J., "Reconstructing Personalized Semantic Facial NeRF Models From Monocular Video," *SIGGRAPH Asia*, vol. 41, no. 6, 2022 (cit. on pp. 43, 45, 54).

[28] Garbin, S. J., Kowalski, M., Estellers, V., Szymanowicz, S., Rezaeifar, S., Shen, J., Johnson, M. A., and Valentin, J., "VolTeMorph: Real-time, Controllable and Generalizable Animation of Volumetric Representations," in *CGS*, Wiley Online Library, vol. 43, 2024, e15117 (cit. on pp. 11, 12, 14, 41–43, 45, 49–51, 55, 60).

[29] Garbin, S. J., Kowalski, M., Johnson, M., Shotton, J., and Valentin, J., "FastNeRF: High-Fidelity Neural Rendering at 200FPS," in *ICCV*, 2021, pp. 14 346–14 355 (cit. on pp. 11, 44, 45).

[30] Grassal, P.-W., Prinzler, M., Leistner, T., Rother, C., Nießner, M., and Thies, J., "Neural Head Avatars From Monocular RGB Videos," in *CVPR*, 2022, pp. 18 653–18 664 (cit. on p. 10).

[31] Grassal, P.-W., Prinzler, M., Leistner, T., Rother, C., Nießner, M., and Thies, J., "Neural Head Avatars from Monocular RGB Videos," in *CVPR*, 2022, pp. 18 653–18 664 (cit. on pp. 41, 51).

[32] Green, R., "Spherical harmonic lighting: The gritty details," in *Archives of the game developers conference*, vol. 56, 2003, p. 4 (cit. on p. 15).

[33] Greff, K., Tagliasacchi, A., Liu, D., and Laradji, I., *Kubric*, `http://github.com/google-research/kubric`, version 0.1.1, 2021 (cit. on p. 28).

[34] Guo, M., Fathi, A., Wu, J., and Funkhouser, T., "Object-Centric Neural Scene Rendering," 2020 (cit. on p. 21).

[35] He, T., Xu, Y., Saito, S., Soatto, S., and Tung, T., "ARCH++: Animation-Ready Clothed Human Reconstruction Revisited," in *ICCV*, 2021 (cit. on p. 22).

[36] Hedman, P., Srinivasan, P. P., Mildenhall, B., Barron, J. T., and Debevec, P., "Baking Neural Radiance Fields for Real-Time View Synthesis," in *ICCV*, 2021, pp. 5875–5884 (cit. on pp. 11, 44, 45, 50).

[37] Higgins, I., Amos, D., Pfau, D., Racaniere, S., Matthey, L., Rezende, D., and Lerchner, A., "Towards a Definition of Disentangled Representations," 2018 (cit. on p. 20).

[38] Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., and Lerchner, A., "beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework," in *ICLR*, 2017 (cit. on p. 20).

[39] Huang, B., Yu, Z., Chen, A., Geiger, A., and Gao, S., "2D Gaussian Splatting for Geometrically Accurate Radiance Fields," in *ACM SIGGRAPH 2024 Conference Papers*, 2024, pp. 1–11 (cit. on pp. 11, 12, 15).

[40] Huang, X., Zhang, Q., Feng, Y., Li, H., Wang, X., and Wang, Q., "HDR-NeRF: High Dynamic Range Neural Radiance Fields," in *CVPR*, 2022, pp. 18 398–18 408 (cit. on p. 44).

[41] Ichim, A. E., Bouaziz, S., and Pauly, M., "Dynamic 3D Avatar Creation from Hand-Held Video Input," *ACM Trans. Graph.*, vol. 34, no. 4, Jul. 2015, ISSN: 0730-0301 (cit. on p. 48).

[42] Irving, G., Teran, J., and Fedkiw, R., "Invertible Finite Elements For Robust Simulation of Large Deformation," in *SIGGRAPH*, 2004, pp. 131–140 (cit. on pp. 43, 46, 49).

[43] Jang, W. and Agapito, L., "CodeNeRF: Disentangled Neural Radiance Fields for Object Categories," in *ICCV*, 2021 (cit. on pp. 19, 21).

[44] Jiang, W., Yi, K. M., Samei, G., Tuzel, O., and Ranjan, A., "NeuMan: Neural Human Radiance Field from a Single Video," in *ECCV*, Springer, 2022, pp. 402–418 (cit. on p. 44).

[45] Kajiya, J. T. and Von Herzen, B. P., "Ray Tracing Volume Densities," *ACM SIGGRAPH Computer Graphics*, vol. 18, no. 3, pp. 165–174, 1984 (cit. on p. 22).

[46] Kaleta, J., Kania, K., Trzcinski, T., and Kowalski, M., "LumiGauss: High-Fidelity Outdoor Relighting with 2D Gaussian Splatting," 2025 (cit. on pp. 12, 15, 16).

[47] Kania, K., Garbin, S. J., Tagliasacchi, A., Estellers, V., Yi, K. M., Valentin, J., Trzciński, T., and Kowalski, M., "BlendFields: Few-Shot Example-Driven Facial Modeling," in *CVPR*, 2023, pp. 404–415 (cit. on pp. 11, 12, 14, 16).

[48] Kania, K., Khirodkar, R., Saito, S., Yi, K. M., and Martinez, J., *CLoG: Leveraging UV Space for Continuous Levels of Detail*, 2024 (cit. on pp. 12, 13, 15, 16).

[49] **Kania**, **K.**, Kowalski, M., and Trzciński, T., "TrajeVAE: Controllable Human Motion Generation from Trajectories," *arXiv preprint arXiv:2104.00351*, 2021 (cit. on p. 16).

[50] Kania, K., Yi, K. M., Kowalski, M., Trzciński, T., and Tagliasacchi, A., "CoNeRF: Controllable Neural Radiance Fields," in *CVPR*, 2022 (cit. on pp. 10, 12, 14, 16, 43, 44).

[51] **Kania**, **K.**, Zięba, M., and Kajdanowicz, T., "UCSG-NET – Unsupervised Discovering of Constructive Solid Geometry Tree," *NeurIPS*, vol. 33, pp. 8776–8786, 2020 (cit. on p. 16).

[52] Kerbl, B., Kopanas, G., Leimkühler, T., and Drettakis, G., "3D Gaussian Splatting for Real-Time Radiance Field Rendering," *TOG*, vol. 42, no. 4, pp. 139–1, 2023 (cit. on pp. 9, 11, 13, 15).

[53] Kim, M., Ko, J., Cho, K., Choi, J., Choi, D., and Kim, S., "AE-NeRF: Auto-Encoding Neural Radiance Fields for 3D-Aware Object Manipulation," *arXiv preprint arXiv:2204.13426*, 2022 (cit. on p. 44).

[54] Kingma, D. P. and Ba, J., "Adam: A Method for Stochastic Optimization," in *ICLR*, 2015 (cit. on pp. 27, 50).

[55] Kulkarni, T., Gupta, A., Ionescu, C., Borgeaud, S., Reynolds, M., Zisserman, A., and Mnih, V., "Unsupervised Learning of Object Keypoints for Perception and Control," in *NeurIPS*, 2019 (cit. on p. 35).

[56] Lewis, J. P., Anjyo, K., Rhee, T., Zhang, M., Pighin, F., and Deng, Z., "Practice and Theory of Blendshape Facial Models," in *Eurographics 2014 - State of the Art Reports*, Lefebvre, S. and Spagnuolo, M., Eds., The Eurographics Association, 2014 (cit. on pp. 11, 43, 53).

[57] Li, S. Z., Zang, Z., and Wu, L., "Markov-Lipschitz Deep Learning," 2020 (cit. on p. 35).

[58] Li, T., Bolkart, T., Black, M. J., Li, H., and Romero, J., "Learning a model of facial shape and expression from 4D scans," *SIGGRAPH Asia*, vol. 36, no. 6, 194:1–194:17, 2017 (cit. on pp. 14, 43, 47, 48, 54, 56).

[59] Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P., "Focal Loss for Dense Object Detection," in *ICCV*, 2017 (cit. on p. 25).

[60] Liu, L., Gu, J., Zaw Lin, K., Chua, T.-S., and Theobalt, C., "Neural Sparse Voxel Fields," *Advances in Neural Information Processing Systems*, vol. 33, pp. 15 651–15 663, 2020 (cit. on p. 45).

[61] Liu, L., Habermann, M., Rudnev, V., Sarkar, K., Gu, J., and Theobalt, C., "Neural Actor: Neural Free-view Synthesis of Human Actors with Pose Control," 2021 (cit. on pp. 20, 21).

[62] Liu, S., Zhang, X., Zhang, Z., Zhang, R., Zhu, J.-Y., and Russell, B., "Editing Conditional Radiance Fields," in *ICCV*, 2021, pp. 5773–5783 (cit. on pp. 10, 14, 19, 21, 43).

[63] Lombardi, S., Simon, T., Saragih, J., Schwartz, G., Lehrmann, A., and Sheikh, Y., "Neural Volumes: Learning Dynamic Renderable Volumes from Images," *ACM Trans. Graph.*, vol. 38, no. 4, Jul. 2019, ISSN: 0730-0301 (cit. on p. 45).

[64] Lombardi, S., Simon, T., Schwartz, G., Zollhoefer, M., Sheikh, Y., and Saragih, J., "Mixture of Volumetric Primitives for Efficient Neural Rendering," *ACM Transactions on Graphics (ToG)*, vol. 40, no. 4, pp. 1–13, 2021 (cit. on p. 45).

[65] Loper, M., Mahmood, N., Romero, J., Pons-Moll, G., and Black, M., "SMPL: A Skinned Multi-Person Linear Model," *ACM Trans. Graph.*, vol. 34, no. 6, 2015 (cit. on pp. 21, 22).

[66] Ma, Q., Saito, S., Yang, J., Tang, S., and Black, M. J., "SCALE: Modeling Clothed Humans with a Surface Codec of Articulated Local Elements," in *CVPR*, 2021 (cit. on p. 22).

[67] Ma, S., Simon, T., Saragih, J., Wang, D., Li, Y., De la Torre, F., and Sheikh, Y., "Pixel Codec Avatars," in *CVPR*, Jul. 2021, pp. 64–73 (cit. on pp. 43, 45, 53).

[68] Martin-Brualla, R., Radwan, N., Sajjadi, M. S., Barron, J. T., Dosovitskiy, A., and Duckworth, D., "NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections," in *CVPR*, 2021, pp. 7210–7219 (cit. on pp. 10, 19, 21, 24, 44).

[69] Mihajlovic, M., Bansal, A., Zollhoefer, M., Tang, S., and Saito, S., "KeypointNeRF: Generalizing Image-based Volumetric Avatars using Relative Spatial Encoding of Keypoints," in *ECCV*, 2022 (cit. on p. 45).

[70] Mihajlovic, M., Zhang, Y., Black, M. J., and Tang, S., "LEAP: Learning Articulated Occupancy of People," in *CVPR*, 2021 (cit. on p. 22).

[71] Mildenhall, B., Hedman, P., Martin-Brualla, R., Srinivasan, P., and Barron, J., "NeRF in the Dark: High Dynamic Range View Synthesis from Noisy Raw Images," in *CVPR*, 2022, pp. 16 190–16 199 (cit. on p. 44).

[72] Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., and Ng, R., "NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis," *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021 (cit. on pp. 10, 11, 19, 21, 22, 24, 28, 41, 43, 44, 50, 51, 55, 56, 62).

[73] Molino, N., Bridson, R., and Fedkiw, R., "Tetrahedral Mesh Generation for Deformable Bodies," 2003 (cit. on p. 45).

[74] Monk, P., "Finite Elements on Tetrahedra," in *Finite Element Methods for Maxwell's Equations*, Oxford, United Kingdom: Oxford University Press, Apr. 2003, pp. 99–154, ISBN: 9780198508885 (cit. on p. 48).

[75] Mu, J., Qiu, W., Kortylewski, A., Yuille, A., Vasconcelos, N., and Wang, X., "A-SDF: Learning Disentangled Signed Distance Functions for Articulated Shape Representation," in *ICCV*, 2021 (cit. on p. 22).

[76] Müller, T., Evans, A., Schied, C., and Keller, A., "Instant Neural Graphics Primitives with a Multiresolution Hash Encoding," *ToG*, vol. 41, no. 4, 102:1–102:15, Jul. 2022 (cit. on pp. 11, 54).

[77] Neumann, T., Varanasi, K., Wenger, S., Wacker, M., Magnor, M., and Theobalt, C., "Sparse Localized Deformation Components," *ACM TOG*, vol. 32, no. 6, pp. 1–10, 2013 (cit. on p. 20).

[78] Noguchi, A., Iqbal, U., Tremblay, J., Harada, T., and Gallo, O., "Watch It Move: Unsupervised Discovery of 3D Joints for Re-Posing of Articulated Objects," in *CVPR*, 2022, pp. 3677–3687 (cit. on p. 44).

[79] Noguchi, A., Sun, X., Lin, S., and Harada, T., "Neural Articulated Radiance Field," in *ICCV*, 2021 (cit. on p. 21).

[80] Oat, C., "Animated Wrinkle Maps," in *SIGGRAPH*, ser. SIGGRAPH '07, San Diego, California: Association for Computing Machinery, 2007, pp. 33–37, ISBN: 9781450318235 (cit. on pp. 11, 12, 47).

[81] Park, J. J., Florence, P., Straub, J., Newcombe, R., and Lovegrove, S., "DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation," in *CVPR*, 2019 (cit. on pp. 23, 24).

[82] Park, K., Sinha, U., Barron, J. T., Bouaziz, S., Goldman, D. B., Seitz, S. M., and Martin-Brualla, R., "Deformable Neural Radiance Fields," in *ICCV*, 2021 (cit. on pp. 19, 21, 22, 24, 26, 28, 37).

[83] Park, K., Sinha, U., Barron, J. T., Bouaziz, S., Goldman, D. B., Seitz, S. M., and Martin-Brualla, R., "Nerfies: Deformable Neural Radiance Fields," in *ICCV*, 2021, pp. 5865–5874 (cit. on pp. 10, 41, 44, 45, 50, 51, 55, 56, 62).

[84] Park, K., Sinha, U., Hedman, P., Barron, J. T., Bouaziz, S., Goldman, D. B., Martin-Brualla, R., and Seitz, S. M., "HyperNeRF: A Higher-Dimensional Representation for Topologically Varying Neural Radiance Fields," *ToG*, vol. 40, no. 6, 2021 (cit. on pp. 10, 14, 19–24, 26, 28, 34, 41, 44, 45, 50, 51, 55, 56, 62).

[85] Patow, G. and Pueyo, X., "A Survey of Inverse Rendering Problems," in *Computer graphics forum*, Wiley Online Library, vol. 22, 2003, pp. 663–687 (cit. on p. 11).

[86] Pumarola, A., Corona, E., Pons-Moll, G., and Moreno-Noguer, F., "D-NeRF: Neural Radiance Fields for Dynamic Scenes," in *CVPR*, 2021, pp. 10 318–10 327 (cit. on pp. 10, 45).

[87] Ramamoorthi, R. and Hanrahan, P., "An efficient representation for irradiance environment maps," in *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, 2001, pp. 497–500 (cit. on pp. 11, 12).

[88] Raman, C., Hewitt, C., Wood, E., and Baltrusaitis, T., "Mesh-Tension Driven Expression-Based Wrinkles for Synthetic Faces," in *WACV Workshop on Applications of Computer Vision*, 2023 (cit. on pp. 43, 44).

[89] Reiser, C., Peng, S., Liao, Y., and Geiger, A., "KiloNeRF: Speeding Up Neural Radiance Fields With Thousands of Tiny MLPs," in *ICCV*, 2021, pp. 14 335–14 345 (cit. on p. 11).

[90] Rematas, K., Liu, A., Srinivasan, P., Barron, J., Tagliasacchi, A., Funkhouser, T., and Ferrari, V., "Urban Radiance Fields," in *CVPR*, 2022, pp. 12 932–12 942 (cit. on p. 44).

[91] Ren, P., Xiao, Y., Chang, X., Huang, P.-Y., Li, Z., Chen, X., and Wang, X., "A Survey of Deep Active Learning," 2020 (cit. on p. 35).

[92] Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B., "High-Resolution Image Synthesis with Latent Diffusion Models," in *CVPR*, 2022, pp. 10 684–10 695 (cit. on p. 54).

[93] Rudnev, V., Elgharib, M., Smith, W., Liu, L., Golyanik, V., and Theobalt, C., "Neural Radiance Fields for Outdoor Scene Relighting," in *European Conference on Computer Vision*, Springer, 2022, pp. 615–631 (cit. on pp. 11, 12).

[94] Saito, S., Yang, J., Ma, Q., and Black, M. J., "SCANimate: Weakly Supervised Learning of Skinned Clothed Avatar Networks," in *CVPR*, 2021 (cit. on p. 22).

[95] Schödl, A. and Essa, I. A., "Controlled Animation of Video Sprites," in *Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ser. SCA '02, San Antonio, Texas: Association for Computing Machinery, 2002, pp. 121–127, ISBN: 1581135734. DOI: 10.1145/545261.545281. [Online]. Available: https://doi.org/10.1145/545261.545281 (cit. on p. 20).

[96] Schwarz, K., Liao, Y., Niemeyer, M., and Geiger, A., "GRAF: Generative Radiance Fields for 3D-Aware Image Synthesis," in *NeurIPS*, 2020 (cit. on p. 21).

[97] Slomp, M. P. B., Oliveira Neto, M. M. d., and Patrício, D. I., "A gentle introduction to precomputed radiance transfer," *Revista de informática teórica e aplicada. Porto Alegre. Vol. 13, n. 2 (2006), p. 131-160*, 2006 (cit. on p. 11).

[98] Spurek, P., Winczowski, S., Zięba, M., Trzciński, T., **Kania**, **K.**, and Mazur, M., "Modeling 3D Surfaces with a Locally Conditioned Atlas," in *ICCS*, Springer, 2024, pp. 100–115 (cit. on p. 16).

[99] Stypułkowski, M., **Kania**, **K.**, Zamorski, M., Zięba, M., Trzciński, T., and Chorowski, J., "Representing Point Clouds with Generative Conditional Invertible Flow Networks," *Pattern Recognition Letters*, vol. 150, pp. 26–32, 2021 (cit. on p. 16).

[100] Su, S.-Y., Yu, F., Zollhöfer, M., and Rhodin, H., "A-NeRF: A-NeRF: Articulated Neural Radiance Fields for Learning Human Shape, Appearance, and Pose," in *NeurIPS*, 2021 (cit. on p. 21).

[101] Suhail, M., Esteves, C., Sigal, L., and Makadia, A., "Light Field Neural Rendering," in *CVPR*, 2022, pp. 8269–8279 (cit. on p. 44).

[102] Sun, J., Wang, X., Zhang, Y., Li, X., Zhang, Q., Liu, Y., and Wang, J., "FENeRF: Face Editing in Neural Radiance Fields," in *CVPR*, 2022, pp. 7672–7682 (cit. on p. 44).

[103] *Suzanne 3D model*, https://github.com/OpenGLInsights/OpenGLInsightsCode/blob/master/Chapter%2026%20Indexing%20Multiple%20Vertex%20Arrays/article/suzanne.obj, Accessed: 2021-11-16 (cit. on p. 28).

[104] Tagliasacchi, A. and Mildenhall, B., "Volume Rendering Digest (for NeRF)," *arXiv preprint arXiv:2209.02417*, 2022 (cit. on p. 45).

[105] Tancik, M., Casser, V., Yan, X., Pradhan, S., Mildenhall, B., Srinivasan, P., Barron, J., and Kretzschmar, H., "Block-NeRF: Scalable Large Scene Neural View Synthesis," in *CVPR*, 2022, pp. 8248–8258 (cit. on p. 44).

[106] Tancik, M., Srinivasan, P. P., Mildenhall, B., Fridovich-Keil, S., Raghavan, N., Singhal, U., Ramamoorthi, R., Barron, J. T., and Ng, R., "Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains," in *NeurIPS*, 2020 (cit. on pp. 10, 12).

[107] *Teapot 3D model*, https://graphics.stanford.edu/courses/cs148-10-summer/as3/code/as3/teapot.obj, Accessed: 2021-11-16 (cit. on p. 28).

[108] Tewari, A., Thies, J., Mildenhall, B., Srinivasan, P., Tretschk, E., Yifan, W., Lassner, C., Sitzmann, V., Martin-Brualla, R., Lombardi, S., *et al.*, "Advances in Neural Rendering," in *CGF*, 2022 (cit. on p. 43).

[109] Thies, J., Zollhofer, M., Stamminger, M., Theobalt, C., and Nießner, M., "Face2Face: Real-time Face Capture and Reenactment of RGB Videos," in *CVPR*, 2016 (cit. on p. 21).

[110] Tretschk, E., Tewari, A., Golyanik, V., Zollhöfer, M., Lassner, C., and Theobalt, C., "Non-Rigid Neural Radiance Fields: Reconstruction and Novel View Synthesis of a Dynamic Scene From Monocular Video," in *ICCV*, 2021, pp. 12 959–12 970 (cit. on pp. 22, 45).

[111] Trevithick, A. and Yang, B., "GRF: Learning a General Radiance Field for 3D Scene Representation and Rendering," in *ICCV*, 2021 (cit. on p. 21).

[112] Vasconcelos, C. N., Oztireli, C., Matthews, M., Hashemi, M., Swersky, K., and Tagliasacchi, A., "CUF: Continuous Upsampling Filters," in *CVPR*, 2023, pp. 9999–10 008 (cit. on p. 13).

[113] Verbin, D., Hedman, P., Mildenhall, B., Zickler, T., Barron, J., and Srinivasan, P., "Ref-NeRF: Structured View-dependent Appearance for Neural Radiance Fields," in *CVPR*, 2022, pp. 5481–5490 (cit. on p. 44).

[114] Vicini, D., Jakob, W., and Kaplanyan, A., "Non-exponential transmittance model for volumetric scene representations," *ToG*, vol. 40, no. 4, Jul. 2021, ISSN: 0730-0301 (cit. on p. 44).

[115] Wang, C., Chai, M., He, M., Chen, D., and Liao, J., "CLIP-NeRF: Text-and-Image Driven Manipulation of Neural Radiance Fields," in *CVPR*, 2022, pp. 3835–3844 (cit. on p. 44).

[116] Wang, D., Chandran, P., Zoss, G., Bradley, D., and Gotardo, P., "MoRF: Morphable Radiance Fields for Multiview Neural Head Modeling," in *SIGGRAPH*, ser. SIGGRAPH '22, Vancouver, BC, Canada: Association for Computing Machinery, 2022, ISBN: 9781450393379 (cit. on p. 45).

[117] Wang, L., Zhang, J., Liu, X., Zhao, F., Zhang, Y., Zhang, Y., Wu, M., Yu, J., and Xu, L., "Fourier PlenOctrees for Dynamic Radiance Field Rendering in Real-time," in *CVPR*, 2022, pp. 13 524–13 534 (cit. on p. 44).

[118] Wang, Z., Simoncelli, E. P., and Bovik, A. C., "Multiscale Structural Similarity for Image Quality Assessment," in *Conference on Signals, Systems & Computers*, 2003 (cit. on pp. 29, 50).

[119] Weng, C.-Y., Curless, B., Srinivasan, P., Barron, J., and Kemelmacher-Shlizerman, I., "HumanNeRF: Free-viewpoint Rendering of Moving People from Monocular Video," in *CVPR*, 2022, pp. 16 210–16 220 (cit. on p. 44).

[120] Wood, E., Baltrušaitis, T., Hewitt, C., Dziadzio, S., Cashman, T. J., and Shotton, J., "Fake It Till You Make It: Face analysis in the wild using synthetic data alone," in *ICCV*, 2021, pp. 3681–3691 (cit. on pp. 47, 49).

[121] Wood, E., Baltrušaitis, T., Hewitt, C., Dziadzio, S., Cashman, T. J., and Shotton, J., "Fake It Till You Make It: Face analysis in the wild using synthetic data alone," in *CVPR*, 2021, pp. 3681–3691 (cit. on p. 50).

[122] Wood, E., Baltrušaitis, T., Hewitt, C., Johnson, M., Shen, J., Milosavljević, N., Wilde, D., Garbin, S., Sharp, T., Stojiljković, I., *et al.*, "3D Face Reconstruction with Dense Landmarks," in *ECCV*, 2022, pp. 160–177 (cit. on p. 45).

[123] Wu, C., Bradley, D., Gross, M., and Beeler, T., "An Anatomically-Constrained Local Deformation Model for Monocular Face Capture," *ACM TOG*, vol. 35, no. 4, pp. 1–12, 2016 (cit. on pp. 20, 21).

[124] Wuu, C.-h., Zheng, N., Ardisson, S., Bali, R., Belko, D., Brockmeyer, E., Evans, L., Godisart, T., Ha, H., Hypes, A., Koska, T., Krenn, S., Lombardi, S., Luo, X., McPhail, K., Millerschoen, L., Perdoch, M., Pitts, M., Richard, A., Saragih, J., Saragih, J., Shiratori, T., Simon, T., Stewart, M., Trimble, A., Weng, X., Whitewolf, D., Wu, C., Yu, S.-I., and Sheikh, Y., "Multiface: A Dataset for Neural Face Rendering," in *arXiv*, 2022 (cit. on pp. 50, 51, 54, 61).

[125] Xia, S., Yue, J., **Kania**, **K.**, Fang, L., Tagliasacchi, A., Yi, K. M., and Sun, W., "Densify Your Labels: Unsupervised Clustering with Bipartite Matching for Weakly Supervised Point Cloud Segmentation," *arXiv preprint arXiv:2312.06799*, 2023 (cit. on p. 16).

[126] Xian, W., Huang, J.-B., Kopf, J., and Kim, C., "Space-time Neural Irradiance Fields for Free-viewpoint Video," in *CVPR*, 2021, pp. 9421–9431 (cit. on p. 44).

[127] Xiangli, Y., Xu, L., Pan, X., Zhao, N., Rao, A., Theobalt, C., Dai, B., and Lin, D., "BungeeNeRF: Progressive Neural Radiance Field for Extreme Multi-scale Scene Rendering," in *ECCV*, 2022, pp. 106–122 (cit. on p. 44).

[128] Xie, C., Park, K., Martin-Brualla, R., and Brown, M., "FiG-NeRF: Figure-Ground Neural Radiance Fields for 3D Object Category Modelling," 2021 (cit. on pp. 10, 21).

[129] Xu, K. and Campeanuy, D., "Houdini engine: Evolution towards a procedural pipeline," in *Proceedings of the Fourth Symposium on Digital Production*, 2014, pp. 13–18 (cit. on p. 52).

[130] Xu, T. and Harada, T., "Deforming Radiance Fields with Cages," in *ECCV*, 2022 (cit. on p. 45).

[131] Yang, B., Bao, C., Zeng, J., Bao, H., Zhang, Y., Cui, Z., and Zhang, G., "NeuMesh: Learning Disentangled Neural Mesh-based Implicit Field for Geometry and Texture Editing," in *ECCV*, 2022, pp. 597–614 (cit. on pp. 44, 45).

[132] Yang, B., Zhang, Y., Xu, Y., Li, Y., Zhou, H., Bao, H., Zhang, G., and Cui, Z., "Learning Object-Compositional Neural Radiance Field for Editable Scene Rendering," in *ICCV*, 2021 (cit. on pp. 19, 21).

[133] Yang, Y., Zhang, S., Huang, Z., Zhang, Y., and Tan, M., "Cross-Ray Neural Radiance Fields for Novel-view Synthesis from Unconstrained Image Collections," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 15 901–15 911 (cit. on p. 11).

[134] Yu, A., Li, R., Tancik, M., Li, H., Ng, R., and Kanazawa, A., "PlenOctrees for Real-time Rendering of Neural Radiance Fields," in *ICCV*, 2021, pp. 5752–5761 (cit. on pp. 44, 45).

[135] Yu, A., Li, R., Tancik, M., Li, H., Ng, R., and Kanazawa, A., "Plenoctrees for Real-time Rendering of Neural Radiance Fields," in *ICCV*, 2021, pp. 5752–5761 (cit. on p. 11).

[136] Yu, H.-X., Guibas, L. J., and Wu, J., "Unsupervised Discovery of Object Radiance Fields," 2021 (cit. on p. 21).

[137] Yuan, Y.-J., Sun, Y.-T., Lai, Y.-K., Ma, Y., Jia, R., and Gao, L., "NeRF-Editing: Geometry Editing of Neural Radiance Fields," in *CVPR*, 2022, pp. 18 353–18 364 (cit. on pp. 43–45).

[138] Zhang, J., Jiang, Z., Yang, D., Xu, H., Shi, Y., Song, G., Xu, Z., Wang, X., and Feng, J., "AvatarGen: A 3D Generative Model for Animatable Human Avatars," in *ECCV*, 2022, pp. 668–685 (cit. on p. 43).

[139] Zhang, K., Riegler, G., Snavely, N., and Koltun, V., "NeRF++: Analyzing and Improving Neural Radiance Fields," 2020 (cit. on pp. 19, 21).

[140] Zhang, K., Riegler, G., Snavely, N., and Koltun, V., "NeRF++: Analyzing and Improving Neural Radiance Fields," *arXiv:2010.07492*, 2020 (cit. on p. 44).

[141] Zhang, R., Isola, P., Efros, A. A., Shechtman, E., and Wang, O., "The Unreasonable Effectiveness of Deep Features as a Perceptual Metric," in *CVPR*, 2018 (cit. on p. 29).

[142] Zhang, R., Isola, P., Efros, A. A., Shechtman, E., and Wang, O., "The Unreasonable Effectiveness of Deep Features as a Perceptual Metric," in *CVPR*, 2018 (cit. on p. 50).

[143] Zhang, S., Moscovich, A., and Singer, A., "Product Manifold Learning," in *Inter. Conf. on Artif. Intell. and Stat.*, 2021 (cit. on p. 35).

[144] Zhang, X., Srinivasan, P. P., Deng, B., Debevec, P., Freeman, W. T., and Barron, J. T., "NeRFactor: Neural Factorization of Shape and Reflectance Under an Unknown Illumination," 2021 (cit. on pp. 14, 19, 21).

[145] Zhao, F., Yang, W., Zhang, J., Lin, P., Zhang, Y., Yu, J., and Xu, L., "HumanNeRF: Efficiently Generated Human Radiance Field from Sparse Inputs," in *CVPR*, 2022, pp. 7743–7753 (cit. on p. 44).

[146] Zheng, Z., Yu, T., Liu, Y., and Dai, Q., "PaMIR: Parametric Model-Conditioned Implicit Representation for Image-based Human Reconstruction," *IEEE TPAMI*, 2021 (cit. on p. 22).

[147] Zhi, Y., Qian, S., Yan, X., and Gao, S., "Dual-Space NeRF: Learning Animatable Avatars and Scene Lighting in Separate Spaces," in *3DV*, IEEE, 2022, pp. 1–10 (cit. on p. 43).

[148] Zhuang, Y., Zhu, H., Sun, X., and Cao, X., "MoFaNeRF: Morphable Facial Neural Radiance Field," in *ECCV*, 2022, pp. 268–285 (cit. on p. 45).

[149] Zielonka, W., Bolkart, T., and Thies, J., "Towards Metrical Reconstruction of Human Faces," in *ECCV*, Springer, 2022, pp. 250–269 (cit. on p. 10).

[150] Zielonka, W., Bolkart, T., and Thies, J., "Instant Volumetric Head Avatars," in *CVPR*, 2023, pp. 4574–4584 (cit. on pp. 10, 54).

[151] Zins, P., Xu, Y., Boyer, E., Wuhrer, S., and Tung, T., "Data-Driven 3D Reconstruction of Dressed Humans From Sparse Views," 2021 (cit. on p. 22).

[152] Zwicker, M., Pfister, H., Van Baar, J., and Gross, M., "EWA volume splatting," in *Proceedings Visualization, 2001. VIS'01.*, IEEE, 2001, pp. 29–538 (cit. on p. 9).