

# NeRF: Neural Radiance Field in 3D Vision, Introduction and Review

Kyle (Yilin) Gao, *Graduate Student Member, IEEE*, Yina Gao, Hongjie He, Dening Lu, Linlin Xu, *Member, IEEE*, Jonathan Li, *Fellow, IEEE*

**Abstract**—Neural Radiance Field (NeRF) has recently become a significant development in the field of Computer Vision, allowing for implicit, neural network-based scene representation and novel view synthesis. NeRF models have found diverse applications in robotics, urban mapping, autonomous navigation, virtual reality/augmented reality, and more. Due to the growing popularity of NeRF and its expanding research area, we present a comprehensive survey of NeRF papers from the past two years. Our survey is organized into architecture and application-based taxonomies and provides an introduction to the theory of NeRF and its training via differentiable volume rendering. We also present a benchmark comparison of the performance and speed of key NeRF models. By creating this survey, we hope to introduce new researchers to NeRF, provide a helpful reference for influential works in this field, as well as motivate future research directions with our discussion section.

**Index Terms**—Neural Radiance Field, NeRF, Computer Vision Survey, Novel View Synthesis, Neural Rendering, Volume Rendering, 3D Reconstruction

## 1 INTRODUCTION

NEURAL Radiance Field (NeRF) models are novel view synthesis methods which use volume rendering with (typically) implicit neural scene representation via Multi Layer Perceptrons (MLPs) to learn the geometry and lighting of a 3D scene. Mildenhall et al. first introduced NeRF at ECCV 2020 [1], and since then, it has achieved state-of-the-art visual quality, produced impressive demonstrations, and inspired many subsequent works. Recently, NeRF models have found applications in photo-editing, 3D surface extraction, human avatar modelling, and large/city-scale 3D representation and view synthesis.

NeRF models have important advantages over other methods of novel view synthesis and scene representation.

- NeRF models are self supervised. They can be trained using only multi-view images of a scene. Unlike many other neural representations of 3D scenes, NeRF models require only images and poses to learn a scene, and do not require 3D/depth supervision. The poses can also be estimated using Structure from Motion (SfM) packages such as COLMAP [2], as was done in certain scenes in the original NeRF paper.
- NeRF models are photo-realistic. Compared to classical techniques such as [3] [4], as well as earlier novel view synthesis methods such as [5][6][7], neural 3D

representation methods [8][9][10], the original NeRF model converged to better results in terms of visual quality, with more recent models performing even better.

NeRF models have attracted much attention in the computer vision community in the recent past, with hundreds of papers and preprints appearing on popular code aggregation website<sup>1</sup> with many eventually appearing in top-tier computer vision conference. In 2022, the impact of NeRF is large and ever increasing, with the original NeRF paper by Mildenhall et al. receiving more than 2000 citations (as of May 2023), and growing interest year-over-year. Given current interest, we believe it necessary to organize a survey paper to help computer vision practitioners with this new topic. We also introduce some of the more recent literature missed out by previous surveys.

The rest of this manuscript is organized as follows.

- Section 2 introduces existing NeRF surveys preprints (2.1), explains the theory behind NeRF volume rendering (2.2), introduces the commonly used datasets (2.3) and quality assessment metrics (2.4).
- Section 3 forms the main body of the paper, and introduces the influential NeRF publications, and contains the taxonomy we created to organize these works. Its subsections detail the different families of NeRF innovations proposed in the past two years, as well as recent applications of NeRF models to various computer vision tasks.
- Sections 5 and 6 discuss potential future research directions and applications, and summarize the survey.

- Corresponding authors: Jonathan Li, Linlin Xu.
- This research was partially funded by the Natural Science and Engineering Research Council of Canada under Grant RGPIN-2022-03741 and the Mitacs Accelerate Program under Project IT32340.
- Kyle Gao, Dening Lu, Linlin Xu, and Jonathan Li are with the Department of Systems Design Engineering, University of Waterloo, Waterloo, Ontario N2L 3G1, Canada (e-mail: y56gao, d62lu, l44xu, junli@uwaterloo.ca).
- Hongjie He and Jonathan Li are with the Department of Geography and Environmental Management, University of Waterloo, Waterloo, Ontario N2L 3G1, Canada (e-mail: h69he@uwaterloo.ca, junli@uwaterloo.ca).
- Yina Gao is with the Faculty of Engineering, University of Toronto (e-mail: Yina.gao@mail.utoronto.ca).

1. <https://paperswithcode.com/method/nerf>

## 2 BACKGROUND

### 2.1 Existing NeRF Surveys

In December 2020, Dellart and Yen-Chen published a preprint NeRF survey [11] including approximately 50 NeRF publications/preprints, many of which were eventually published in top tier computer vision conferences. We took inspiration from this preprint survey and used it as a starting point for our own survey. However, the work is only five pages, and does not include detailed descriptions. Moreover, the work only included NeRF papers from 2020 and early 2021 preprints, and did not include several influential NeRF papers published in the latter half of 2021 and beyond.

In November 2021, Xie et al. [12] published a preprint survey titled Neural Fields in Visual Computing and Beyond, which was later revised and published as a state-of-the-art report in Eurographics 2022, as well as presented as a tutorial in CVPR 2022. The survey is broad and includes a detailed technical introduction to the use of Neural Fields in Computer Vision, as well as applications in visual computing, with a large focus on Neural Radiance Fields. Compared to this work, we restrict our review to NeRF papers only, and are able to include more recent work. We also present more detail on a paper-to-paper basis.

In December 2021, Zhang et al. [13] published a preprint survey for Multimodal image synthesis and editing in which they dedicated a paragraph for NeRF. They mostly focused on multi-modal NeRFs such as [14], [15], only citing these two and the original NeRF paper [1] in their survey, as well as four more papers [16] [17][18][19] in their supplementary materials.

In May 2022, Tewari et al. [20] published a state-of-the-art report on advances in Neural Rendering with a focus on NeRF models. It is to date the most comprehensive Neural Rendering survey style report, including many influential NeRF papers, as well as many other Neural Rendering papers. Our survey differs from this report in that our scope is completely focused on NeRF papers, giving detailed paper-by-paper summary of selected works. We also present a NeRF innovation technique taxonomy tree, and a NeRF application classification tree. We are able to include the more recent works from late 2022 and early 2023, as well as introduce in detail common datasets and evaluation metrics used by NeRF practitioners.

### 2.2 Neural Radiance Field (NeRF) Theory

Neural Radiance Fields were first proposed by Mildenhall et al. [1] in 2020 for novel view synthesis. NeRFs achieved highly photo-realistic view synthesis of complex scenes and attracted much attention in the field. In its basic form, a NeRF model represents three-dimensional scenes as a radiance field approximated by a neural network. The radiance field describes color and volume density for every point and for every viewing direction in the scene. This is written as:

$$F(\mathbf{x}, \theta, \phi) \rightarrow (\mathbf{c}, \sigma), \quad (1)$$

where  $\mathbf{x} = (x, y, z)$  is the in-scene coordinate,  $(\theta, \phi)$  represent the azimuthal and polar viewing angles,  $\mathbf{c} = (r, g, b)$  represents color, and  $\sigma$  represents the volume density. This

5D function is approximated by one or more Multi-Layer Preceptron (MLP) sometimes denoted as  $F_\Theta$ . The two viewing angles  $(\theta, \phi)$  are often represented by  $\mathbf{d} = (d_x, d_y, d_z)$ , a 3D Cartesian unit vector. This neural network representation is constrained to be multi-view consistent by restricting the prediction of  $\sigma$ , the volume density (i.e., the content of the scene) to be independent of viewing direction, whereas the color  $\mathbf{c}$  is allowed to depend on both viewing direction and in-scene coordinate. In the baseline NeRF model, this is implemented by designing the MLP to be in two-stages. The first stage takes as input  $\mathbf{x}$  and outputs  $\sigma$  and a high-dimensional feature vector (256 in the original paper). In the second stage, the feature vector is then concatenated with the viewing direction  $\mathbf{d}$ , and passed to an additional MLP, which outputs  $\mathbf{c}$ . We note that Mildenhall et al. [1] consider the  $\sigma$  MLP and the  $\mathbf{c}$  MLP to be two branches of the same neural network, but many subsequent authors consider them to be two separate MLP networks, a convention which we follow from this point on.

One should also note that in existing literature, three types of scene representation emerged; implicit, hybrid and explicit. In the baseline NeRF, the density and color fields are fully represented by MLPs; this is considered an implicit scene representation. Methods with hybrid and explicit scene representation are introduced in sections 3.2 and 3.7.1, respectively.

Broadly speaking, novel view synthesis using a trained NeRF model is as follows.

- For each pixel in the image being synthesized, send camera rays through the scene and generate a set of sampling points (see (a) in Fig. 1).
- For each sampling point, use the viewing direction and sampling location to compute local color and density using the NeRF MLP(s) (as shown in (b) in Fig. 1).
- Use volume rendering to produce the image from these colors and densities (see (c) in Fig. 1).

Given the volume density and color functions of the scene being rendered, volume rendering [21] is used to obtain the color  $C(\mathbf{r})$  of any camera ray  $\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$ , with camera position  $\mathbf{o}$  and viewing direction  $\mathbf{d}$  using

$$C(\mathbf{r}) = \int_{t_1}^{t_2} T(t) \cdot \sigma(\mathbf{r}(t)) \cdot \mathbf{c}(\mathbf{r}(t), \mathbf{d}) \cdot dt, \quad (2)$$

where  $\sigma(\mathbf{r}(t))$  and  $\mathbf{c}(\mathbf{r}(t), \mathbf{d})$  represent the volume density and color at point  $\mathbf{r}(t)$  along the camera ray with viewing direction  $\mathbf{d}$ , and  $dt$  represents the differential distance traveled by the ray at each integration step.

$T(t)$  is the accumulated transmittance, representing the probability that the ray travels from  $t_1$  to  $t$  without being intercepted, given by

$$T(t) = \exp\left(-\int_{t_1}^t \sigma(\mathbf{r}(u)) \cdot du\right). \quad (3)$$

Novel views are rendered by tracing the camera rays  $C(\mathbf{r})$  through each pixel of the to-be-synthesized image. This integral can be computed numerically. The original implementation [1] and most subsequent methods used a non-deterministic stratified sampling approach, where the

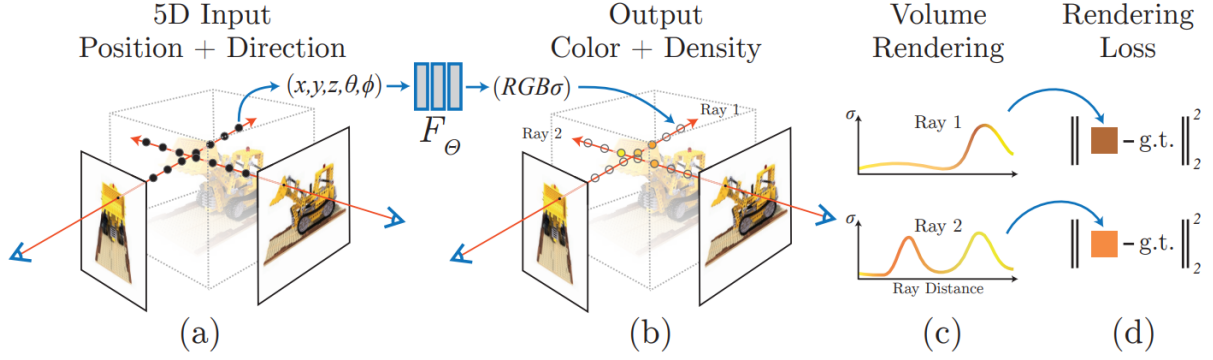


Fig. 1. The NeRF volume rendering and training process. Image sourced from [1]. (a) illustrates the selection of sampling points for individual pixels in a to-be-synthesized image. (b) illustrates the generation of densities and colors at the sampling points using NeRF MLP(s). (c) and (d) illustrate the generation of individual pixel color(s) using in-scene colors and densities along the associated camera ray(s) via volume rendering, and the comparison to ground truth pixel color(s), respectively.

ray was divided into  $N$  equally spaced bins, and a sample was uniformly drawn from each bin. Then, equation (2) can be approximated as

$$\hat{C}(\mathbf{r}) = \sum_{i=1}^N \alpha_i T_i \mathbf{c}_i, \text{ where } T_i = \exp\left(-\sum_{j=1}^{i-1} \sigma_j \delta_j\right). \quad (4)$$

$\delta_i$  is the distance from sample  $i$  to sample  $i + 1$ .  $(\sigma_i, \mathbf{c}_i)$  are the density and color evaluated along the sample point  $i$  given the ray, as computed by the NeRF MLP(s).  $\alpha_i$  the transparency/opacity from alpha compositing at sample point  $i$ , is given by

$$\alpha_i = 1 - \exp(-\sigma_i \delta_i). \quad (5)$$

An expected depth can be calculated for the ray using the accumulated transmittance as

$$d(\mathbf{r}) = \int_{t_1}^{t_2} T(t) \cdot \sigma(\mathbf{r}(t)) \cdot t \cdot dt. \quad (6)$$

This can be approximated analogously to equation (4) approximating equation (2) and (3)

$$\hat{D}(\mathbf{r}) = \sum_{i=1}^N \alpha_i t_i T_i. \quad (7)$$

Certain depth regularization [22] [23] [24] [25] methods use the expected depth to restrict densities to delta-like functions at scene surfaces, or to enforce depth smoothness.

For each pixel, a square error photometric loss is used to optimize the MLP parameters. Over the entire image, this is given by

$$L = \sum_{\mathbf{r} \in R} \|\hat{C}(\mathbf{r}) - C_{gt}(\mathbf{r})\|_2^2 \quad (8)$$

where  $C_{gt}(\mathbf{r})$  is the ground truth color of the training image's pixel associated to  $\mathbf{r}$ , and  $R$  is the batch of rays associated to the to-be-synthesized image.

NeRF models often employ positional encoding, which was shown by Mildenhall et al. [1] to greatly improve fine detail reconstruction in the rendered views. This was also shown in more detail, with corroborating theory using

Neural Tangent Kernels in [26]. (We note a recent work [27] instead tests novel activation functions to address this issue). In the original implementation, the following positional encoding  $\gamma$  was applied to each component of the scene coordinate  $\mathbf{x}$  (normalized to  $[-1,1]$ ) and viewing direction unit vector  $\mathbf{d}$

$$\gamma(v) = (\sin(2^0 \pi v), \cos(2^0 \pi v), \sin(2^1 \pi v), \cos(2^1 \pi v), \dots, \sin(2^{N-1} \pi v), \cos(2^{N-1} \pi v)), \quad (9)$$

where  $N$  is a user determined encoding dimensionality parameter, set to  $N = 10$  for  $\mathbf{x}$  and  $N = 4$  for  $\mathbf{d}$  in the original paper. However, modern researches have experimented and achieved great results with alternate forms of positional encoding including trainable parametric, integral, and hierarchical variants (see section 3).

### 2.3 Datasets

NeRF models are typically trained per-scene and require relatively dense images with relatively varied poses. While some NeRF models have been designed to be trained from sparse input views or unposed images, camera poses can often be extracted using existing Structure-from-Motion libraries such as COLMAP [2].

The original NeRF paper [1] presented a synthetic dataset created from Blender (referred to as **Realistic Synthetic 360°** in [1]). The virtual cameras have the same focal length and are placed at the same distance from the object. The dataset is composed of eight scenes with eight different objects. For six of these, viewpoints are sampled from the upper hemisphere, for the two others, viewpoints are sampled from the entire sphere. These objects are "hot-dog", "materials", "ficus", "lego", "mic", "drums", "chair", "ship". The images are rendered at  $800 \times 800$  pixels, with 100 views for training and 200 views for testing. This is often the first dataset considered by NeRF researchers, as the scenes are well bounded, focused on single object, and the dataset is well benchmarked against known models.

The **LLFF dataset** [5] consists of 24 real-life scenes captured from handheld cellphone cameras. The views are forward facing towards the central object. Each scene consists of 20-30 images. The COLMAP [2] package was used to

compute the poses of the images. The usage of this dataset is comparable to that of the Realistic Synthetic dataset from [1]; the scenes are not too challenging for any particular NeRF model, and the dataset is well benchmarked, offering readily available comparisons to known methods.

The **DTU dataset** [28] is a multi-view stereo dataset captured using a 6-axis industrial robot mounted with both a camera and a structured light scanner. The robot provided precise camera positioning. Both the camera intrinsics and poses are carefully calibrated using the MATLAB calibration toolbox [29]. The light scanner provides reference dense point clouds which provide reference 3D geometry. Nonetheless, due to self-occlusion, the scans of certain areas in certain scenes are not complete. The original paper’s dataset consists of 80 scenes each containing 49 views sampled on a sphere of radius 50 cm around the central object. For 21 of these scenes, an additional 15 camera positions are sampled at a radius of 65 cm, for a total of 64 views. The entire dataset consists of 44 additional scenes that have been rotated and scanned four times at 90 degree interval. The illumination of scenes is varied using 16 LEDs, with seven different lighting conditions. The image resolution is  $1600 \times 1200$ . This dataset differs the previous ones by its higher resolution and carefully calibrated camera motion and poses.

The **ScanNet dataset** [30] is a large-scale real-life RGB-D multi-modal dataset containing more than 2.5 million views of indoor scenes, with annotated camera poses, reference 3D surfaces, semantic labels, and CAD models. The depth frames are captured at  $640 \times 480$  pixels, and the RGB images are captured at  $1296 \times 968$  pixels. The scans were performed using RGB-D sensors attached to handheld devices such as iPhone/iPad. The poses were estimated from BundleFusion [31] and geometric alignment of resulting mesh. This dataset’s rich semantic labels are useful for models that make use of semantic information, such as for scene editing, scene segmentation, and semantic view synthesis.

The **ShapeNet dataset** [32] is a simplistic large scale synthetic 3D dataset, consisting of 3D CAD model classified into 3135 classes. The most used are the 12 common object categories sub-dataset. This dataset is sometimes used when object-based semantic labels are an important part of a particular NeRF model. From ShapeNet CAD models, software such as Blender are often used to render training views with known poses.

### 2.3.1 Building-scale Dataset

The **Tanks and Temples dataset** [33] is a from-video 3D reconstruction dataset. It consists of 14 scenes, including individual objects such as “Tank” and “Train”, and large scale in-door scenes such as “Auditorium” and “Museum”. Ground truth 3D data was captured using high quality industrial laser scanner. The ground truth point cloud was used to estimate camera poses using least squares optimization of correspondence points. This dataset contains large scale scenes, some of which outdoors, and poses a challenge for certain NeRF model. The outdoors scenes are suited for models wishing to challenge unbounded background. It’s ground truth point cloud can also be used for certain data fusion methods, or to test depth reconstruction.

The **Matterport-3D dataset** [34] is a real life dataset consisting 10800 panoramic views from 194400 RGB-D globally registered images of 90 building scale scenes. Depth, semantic and instance annotations are available. Color and depth images are provided at  $1280 \times 1024$  resolution for 18 view points per panoramic pictures. Each of the 90 building consists of an average of  $2437m^2$  of surface area. A total of 50811 object instance labels were provided which were mapped into 40 object categories.

The **Replica dataset**[35] is an real indoor consisting of 18 scenes and 35 indoor rooms captured using a custom built RGB-D rig with IR projector. Certain 3D features were manually fixed (fine-scale mesh details such as small holes), and reflective surfaces were manually assigned. Semantic annotations (88 classes) was performed in two step, once in 2D, and once in 3D. Both class-based and instance-based semantic labels are available.

### 2.3.2 Large-Scale Urban Datasets

Popular autonomous driving benchmark datasets have multiple data modalities such as image, depth map, LiDAR point cloud, poses, and semantic maps, which are potentially suitable for certain NeRF models wishing to target urban scenes. Recently, NeRF models such as those proposed in [49] and [50] have made effective use of these datasets.

**KITTI**[51][52][53][54] is a well-known city-scale 2D-3D computer vision dataset suite, created for the training and benchmarking of vision algorithms for autonomous driving. The suite contains labeled datasets for stereo 3D semantic + 2D semantic segmentation, flow, odometry, 2D-3D object detection, tracking, lane detection, and depth prediction/completion. These were created from raw LiDAR and video data captured in Karlsruhe, Germany, using a car based setup with GPS and inertial measurement unit data, captured with a Velodyne LiDAR scanner and multiple cameras. The depth prediction/completion dataset is by far the largest of these, containing over 93 thousand depth maps with corresponding RGB images and raw LiDAR scans. This dataset however poses a challenge to NeRF training due to its camera coverage that is relatively sparse compared to that of NeRF-specific datasets, requiring sparse-view considerations when designing the model. The recent Kitti-360 [55] extension to the dataset suite even include a novel view synthesis benchmark which tabulates many NeRF models.

The **Waymo Open Dataset** [56] is a recently published alternative to KITTI. Covering  $72km^2$ , the dataset is created from point cloud and video data captured from five LiDAR sensors and five high-resolution pinhole cameras in a car based setup, captured in the San Francisco Bay, Mountain View, and Phoenix in the United States. In addition to matched point cloud and video data, the dataset also contains annotated labels for 2D and 3D object detection and tracking. The dataset contains 1150 separate scenes, (as opposed to KITTI’s 22 scenes) and has higher LiDAR and camera resolution. Its object annotations are also more extensive by two orders of magnitude (80K vs 12M).

### 2.3.3 Human Avatar/Face Dataset

The **Nerfies** [57] and **HyperNerf** [58] datasets are single camera datasets focused on the human faces, with motion

generated by moving two cameras attached to a pole, relative to the subject. The former contains five human subjects staying still as well as four more scenes with moving human subjects, as well as a dog, and two moving objects. The latter focuses on topological changes, and includes scenes such as a human subject opening and closing their eyes and mouth, peeling a banana, 3D printing a chicken toy, and a broom deforming

The **ZJU-MOCap LightStage dataset** [59] is a multi-view (20+ cameras) motion capture dataset consisting of 9 dynamic human sequences consisting of exercise-like motions. The videos were captured using 21 synchronized cameras and have sequence length between 60 to 300 frames.

The **NeuMan dataset** [60] consists of 6 videos each 10 to 20 seconds long captured by mobile phone camera following a walking human subject performing additional simple actions such as twirling or waving.

The **CMU Panoptic dataset** [61] is a large multi-view multi-subject dataset consisting groups of people engaged in social interaction. The dataset consists of 65 sequences with 1.5 millions labelled skeletons. The sensor system consists of 480 VGA views (640x480), 30+HD (1920x1080) views, and 10 RGB-D sensors. Scenes are labelled with labels of individual subjects and social groups semantics, 3D body poses, 3D facial landmarks, and transcripts + speaker ID.

## 2.4 Quality Assessment Metrics

Novel view synthesis via NeRF in the standard setting use visual quality assessment metrics for benchmarks. These metrics attempt to assess the quality of individual images either with (full-reference) or without (no-reference) ground truth images. Peak Signal to Noise Ratio (PSNR), Structural Similarity Index Measure (SSIM) [62], Learned Perceptual Image Patch Similarity (LPIPS) [63] are by far the most commonly used in NeRF literature.

PSNR $\uparrow$  is a no-reference quality assessment metric, given by

$$PSNR(I) = 10 \cdot \log_{10} \left( \frac{MAX(I)^2}{MSE(I)} \right) \quad (10)$$

where  $MAX(I)$  is the maximum possible pixel value in the image (255 for 8bit integer), and  $MSE(I)$  is the pixel-wise mean squared error calculated over all color channels. PSNR is also commonly used in other fields of signal processing and is well understood.

SSIM $\uparrow$  [62] is a full-reference quality assessment metric. For a single patch, this is given by

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \quad (11)$$

where  $C_i = (K_i L)^2$ ,  $L$  is the dynamic range of the pixels (255 for 8bit integer), and  $K_1 = 0.01, K_2 = 0.03$  are constants chosen by the original authors. We note that there is a more general form of SSIM given by (12) in the original paper. The local statistics  $\mu$ 's,  $\sigma$ 's are calculated within a  $11 \times 11$  circular symmetric Gaussian weighted window, with weights  $w_i$  having a standard deviation of 1.5 and normalized to 1. These are given by, without loss of generalization,

$$\mu_x = \sum_i w_i x_i \quad (12)$$

$$\sigma_x = \left( \sum_i w_i (x_i - \mu_x)^2 \right)^{1/2} \quad (13)$$

$$\sigma_{xy} = \sum_i w_i (x_i - \mu_x)(y_i - \mu_y) \quad (14)$$

where  $x_i, y_i$  are pixels sampled from the reference and assessed images respectively. The patch-wise SSIM scores are averaged over the entire image in practice.

LPIPS $\downarrow$  [63] is a full reference quality assessment metric which uses learned convolutional features. The score is

TABLE 1  
Comparison of select NeRF models on the synthetic NeRF dataset [1]

Method	Representation	PSNR $\uparrow$ (dB)	SSIM $\uparrow$	LPIPS $\downarrow$	Training Iteration	Training Time <sup>1</sup>	Inference Speed <sup>2</sup>
Baseline NeRF (2020)[1]	Implicit	31.01	0.947	0.081	100-300k	>12h	1
Speed Improvement							
JaxNeRF (2020)[36]	Implicit	31.65	0.952	0.051	250k	>12h	~1.3
NSVF (2020) [37]	Hybrid (Learned)	31.74	0.953	0.047	100-150k	-	~10
SNeRG (2021) [38]	Explicit (Baked)	30.38	0.950	0.050	250k	>12h	~9000
PlenOctree (2021) [39]	Explicit (Baked)	31.71	0.958	0.053	2000k	>12h	~3000
FastNeRF (2021) [40]	Explicit (Baked)	29.97	0.941	0.053	300k	>12h	~4000
KiloNeRF (2021) [41]	Explicit (Baked)	31.00	0.95	0.03	600k+150k+1000k	>12h	~2000
Instant-NGP (2022) [42]	Hybrid (Learned)	33.18	-	-	256k	~5m	"orders of magnitude"
Plenoxels (2021) [43]	Explicit (Learned)	31.71	0.958	0.049	128k	~12m	-
DVGO (2021) [43]	Explicit (Learned)	31.95-32.80	0.957-0.961	0.053-0.27	128k	~20m	45x
TensorRF (2022) [44]	Explicit/Hybrid (Learned)	31.56-33.14	0.949-0.963	-	15K-30K	8-25m	~100
Quality Improvement							
mip-NeRF (2021)[45]	Implicit	33.09	0.961	0.043	1000k	~3h	~1
ref-NeRF (2021)[46]	Implicit	35.96	0.967	0.058	250k	-	~1
Sparse View / Few Shots							
MVSNeRF (2021)[47]	Hybrid + Pretrained	27.07	0.931	0.163	10k (*3 views)	~15m*	~1
DietNeRF (2021)[48]	Implicit + Pretrained	23.15	0.866	0.109	200k (*8 views)	-	~1
DS-NeRF (2021)[23]	Implicit	24.9	0.72	0.34	150-200k (*10 views)	-	~1

Key speed, sparse-view and quality improvement models focused on small scale scenes trained on the Synthetic NeRF dataset were selected for comparison.

<sup>1</sup>Training speed are given as in the respective original papers. These are to be taken with "a grain of salt", as hardware differences and hyper-parameters such as image/voxel resolution greatly affect training time.

<sup>2</sup>Inference speeds are given as speedup factor over the baseline NeRF.

given by a weighted pixel-wise MSE of feature maps over multiple layers.

$$LPIPS(x, y) = \sum_l \frac{1}{H_l W_l} \sum_{h,w}^{H_l, W_l} \|w_l \odot (x_{hw}^l - y_{hw}^l)\|_2^2 \quad (15)$$

where  $x_{hw}^l, y_{hw}^l$  are the reference and assessed images' feature at pixel width  $w$ , pixel height  $h$ , and layer  $l$ .  $H_l$  and  $W_l$  are the feature maps height and width at the corresponding layer. The original LPIPS paper used SqueezeNet [64], VGG [65] and AlexNet [66] as feature extraction backbone. Five layers were used in the original paper. The original authors offered fine-tuned and from-scratch configurations, but in practice, the pretrained networks are used as is.

### 3 NEURAL RADIANCE FIELD (NeRF)

In this section, we present select NeRF papers organized in a method-based taxonomy, with a separate section reserved for application-based classification. The (arXiv) preprint first draft dates are used to sequence the publications, while conference/journal publication dates can be found in the bibliography. We have categorized select models into innovation (Fig. 3) and application (Fig. 8) taxonomy trees, and included a NeRF synthetic dataset [1] benchmark table (Table 1).

#### 3.1 Improvements in the Quality of Synthesized Views and Learned Geometry

Image quality assessment is a crucial metric for NeRF view synthesis, and as such, many subsequent models have focused on improving view synthesis quality. In this section, we highlight some important models that have aimed to enhance the photometric and geometric aspects of NeRF view synthesis and 3D scene representation, resulting in improved image quality in synthesized views.

##### 3.1.1 Better View Synthesis

**Mip-NeRF** [45] (March 2021) used cone tracing instead of ray tracing of standard NeRF [1] (March 2020) volume rendering. They achieved this by introducing the Integrated Positional Encoding (IPE) (Fig. 2). To generate an individual pixel, a cone was cast from the camera's center along the viewing direction, through the center of the pixel. This cone was approximated by a multivariate Gaussian, whose mean vector and variance matrix were derived to have the appropriate geometry (see Appendix A in [45]), resulting in the Integrated Positional Encoding. This is given by

$$\begin{aligned} \gamma(\boldsymbol{\mu}, \boldsymbol{\Sigma}) &= \mathbb{E}_{\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})}[\gamma(\mathbf{x})] \\ &= \begin{bmatrix} \sin(\boldsymbol{\mu}_\gamma) \odot \exp(-(1/2)\text{diag}(\boldsymbol{\Sigma}_\gamma)) \\ \cos(\boldsymbol{\mu}_\gamma) \odot \exp(-(1/2)\text{diag}(\boldsymbol{\Sigma}_\gamma)) \end{bmatrix} \end{aligned} \quad (16)$$

where  $\boldsymbol{\mu}_\gamma, \boldsymbol{\Sigma}_\gamma$  are the means and variances of the multivariate Gaussian lifted onto the positional encoding basis with  $N$  levels. This process is given by

$$\boldsymbol{\mu}_\gamma = \mathbf{P} \boldsymbol{\mu}, \quad \boldsymbol{\Sigma}_\gamma = \mathbf{P} \boldsymbol{\Sigma} \mathbf{P}^T \quad (17)$$

where

$$\mathbf{P} = \begin{bmatrix} 1 & 0 & 0 & 2 & 0 & 0 & \dots & 2^{N-1} & 0 & 0 \\ 0 & 1 & 0 & 0 & 2 & 0 & \dots & 0 & 2^{N-1} & 0 \\ 0 & 0 & 1 & 0 & 0 & 2 & \dots & 0 & 0 & 2^{N-1} \end{bmatrix}. \quad (18)$$

The diagonal elements of the variance matrix can be directly calculated in practice (see (16) in [45]). The resulting mip-NeRF model was multi-scale in nature and performed anti-aliasing automatically. The model outperformed the baseline NeRF [1], significantly so at lower resolutions. **Mip-NeRF 360** [99] is a highly influential work directly extending Mip-NeRF to unbounded scenes. The key technical improvements were firstly a proposal MLP, supervised using NeRF MLP and not the images. This proposal MLP only predicts volumetric density (not color) which is used to find appropriate sampling intervals. Secondly, a novel scene parametrization was specifically constructed for the Gaussians in Mip-NeRF. Thirdly, a new regularization method was introduced which prevents floater geometric artifacts and background collapse.

**Ref-NeRF** [46] (December 2021) was built on mip-NeRF and was designed to better model reflective surfaces. Ref-NeRF parameterized NeRF radiance based on the reflection of the viewing direction about the local normal vector. They modified the density MLP into a directionless MLP which not only outputs density and the input feature vector of the directional MLP but also diffuse color, specular color, roughness, and surface normal. The diffuse color and specular color were multiplied together and added to the specular color (output of directional MLP), which gave the final color. Additionally, they parameterized the directional vector using the spherical harmonics of vectors sampled from a spherical Gaussian distribution parameterized by the roughness. Ref-NeRF outperformed benchmarked methods, including mip-NeRF [45], NSVF [37], baseline NeRF [1], and non-NeRF models, on the Shiny Blender dataset (created by the authors), the original NeRF dataset [1], and Real Captured Scenes from [100]. Ref-NeRF performed particularly well on reflective surfaces and is capable of accurately modeling specular reflections and highlights (Fig. 4).

**RegNeRF** [22] (December 2021) aimed to solve the problem of NeRF training with sparse input views. Unlike most other methods that approached this task by using image features from pretrained networks as a prior for conditioning NeRF volume rendering, RegNeRF employed additional depth and color regularization. The model was tested on DTU [28] and LLFF [5] datasets, outperformed models such as PixelNeRF [74], SRf [101], MVSNeRF [47]. RegNeRF which did not require pretraining, achieved comparable performance to these models which were pre-trained on DTU and fine-tuned per scene. It outperformed Mip-NeRF and DietNeRF [48].

**Ray Prior NeRF (RapNeRF)** (May 2022) [102] explored a NeRF model better suited for view extrapolation, whereas standard NeRF models were better suited for interpolation. RapNeRF performed Random Ray Casting (RRC) whereby, given a training ray hitting a surface point  $\mathbf{v} = \mathbf{o} + t_z \mathbf{d}$ , a backward ray was cast starting from  $\mathbf{v}$  towards a new origin  $\mathbf{o}'$  using uniformly sampled perturbations in angles. RapNeRF also made use of a Ray Atlas (RA) by first extracting a rough 3D mesh from a pretrained NeRF, and mapping training ray directions onto the 3D vertices. During training, a baseline NeRF was first trained to recover the rough 3D mesh, then RRC and RA are used to augment the training rays, with a predetermined probability. The authors evaluated their methods on the Synthetic NeRF



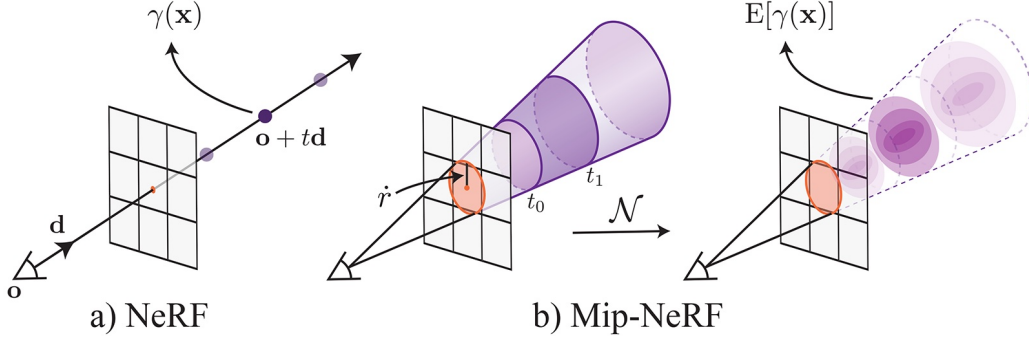


Fig. 2. Diagram of the Integrated Positional Encoding (IPE) of mip-NeRF (Figure 1 in [45]). a) Standard ray-based point sampled of NeRF; b) Cone-sampling of mip-NeRF using IPE, approximating conic frustums with multivariate Gaussian distributions.

dataset [1], and their own MobileObject dataset, showing that their RRC and RA augmentations can be adapted to other NeRF framework, and that it resulted in better view synthesis quality.

### 3.1.2 Depth Supervision and Point Cloud

By using supervision of expected depth (6) with point clouds acquired from LiDAR or SfM, these models converge faster, converge to higher final quality, and require fewer training views than the baseline NeRF model. Many of these models were also built for or as few-shot/sparse view NeRF. At the end of the section, other methods for geometry improvements are introduced.

Deng et al. [23] (July 2021) used **depth supervision from point clouds** with a method named **Depth-Supervised NeRF (DS-NeRF)**. In addition to color supervision via volume rendering and photometric loss, **DS-NeRF also performs depth supervision using sparse point clouds extracted from the training images using COLMAP [2]**. Depth is modeled as a normal distribution around the depth recorded by the sparse point cloud. A **KL divergence term is added to minimize the divergence of the ray's distribution and this noisy depth distribution** (see [23] for details). DS-NeRF was extensively tested on the DTU dataset [28], NeRF dataset [1], and the RedWood-3dscan dataset [103], outperforming benchmark methods such as baseline NeRF [1], PixelNeRF [74], and MVSNeRF [47].

Concurrent to DS-NeRF is a work by **Roessle et al. [68]** (April 2021). In this work, the **authors used COLMAP to extract a sparse point cloud, which was processed by a Depth Completion Network [104] to produce depth and uncertainty maps**. In addition to the standard volumetric loss, the **authors introduced a depth loss based on predicted depth and uncertainty**. The model was trained on RGB-D data from ScanNet [30] and Matterport3D [34] by introducing Gaussian noise to depth. The model outperformed DS-NeRF[23] marginally, and significantly outperformed baseline NeRF [1], and NerfingMVS [69].

**NerfingMVS [69]** (September 2021) **used multi-view images in their NeRF model focused on depth reconstruction**. In NerfingMVS, COLMAP was used to extract sparse depth priors in the form of a point cloud. This was then fed into a pretrained (fine-tuned on the scene) monocular depth network [105] to extract a depth map prior. This depth map prior was used to supervise volume sampling by only

allowing sampling points at the appropriate depth. During **volume rendering, the ray was divided into  $N$  equal bins, with the ray bounds clamped using the depth priors**. The depth value  $D$  of a pixel was approximated by a modified version of (4). NerfingMVS outperformed previous methods on the ScanNet [30] dataset for depth estimation.

**PointNeRF [71]** (January 2022) **used feature point clouds as an intermediate step to volume rendering**. A Pretrained **3D CNN [106] was used to generate depth and surface probability  $\gamma$**  from a cost volume created from training views, and produced a dense point cloud. A pretrained **2D CNN [65] was used to extract image features from training views**. These were used to populate the point cloud features with image features, and probability  $\gamma_i$  of point  $p_i$  lying of a surface. Given the input position and view-direction, a PointNet[107]-like network was used to regress local density and color, which was then used for volume rendering. Using **point cloud features also allowed the model to skip empty spaces**, resulting in a speed-up of a factor of 3 over baseline NeRF. PointNeRF outperformed methods such as PixelNeRF [74], MVSNeRF [47], IBRNet [108] after per-scene optimization in the form of point growing and point pruning on the DTU dataset [28]. Point clouds acquired by other methods such as COLMAP can also be used in place of the 3D depth network-based point cloud, whereby per-scene optimization could be used to improve point cloud quality.

### 3.1.3 Other Geometry Improvement

**SNeS [109]** (June 2022) **improved geometry by learning probable symmetries for partly symmetric and partly hidden in-scene objects through soft symmetry constraints on geometry and material properties**.

**S<sup>3</sup>-NeRF [110]** (October 2022) **used shadow and shading to infer scene geometry and achieved single image NeRF training, focusing on geometry recovery**. S<sup>3</sup>-NeRF used an UNISURF-based occupancy field 3D representation (as opposed to density), a modified physics-based rendering equation, and an occupancy-based shadow calculation as key implementation differences. The method **achieved excellent depth map and surface normal reconstruction from single images from both synthetic and real-life datasets**.

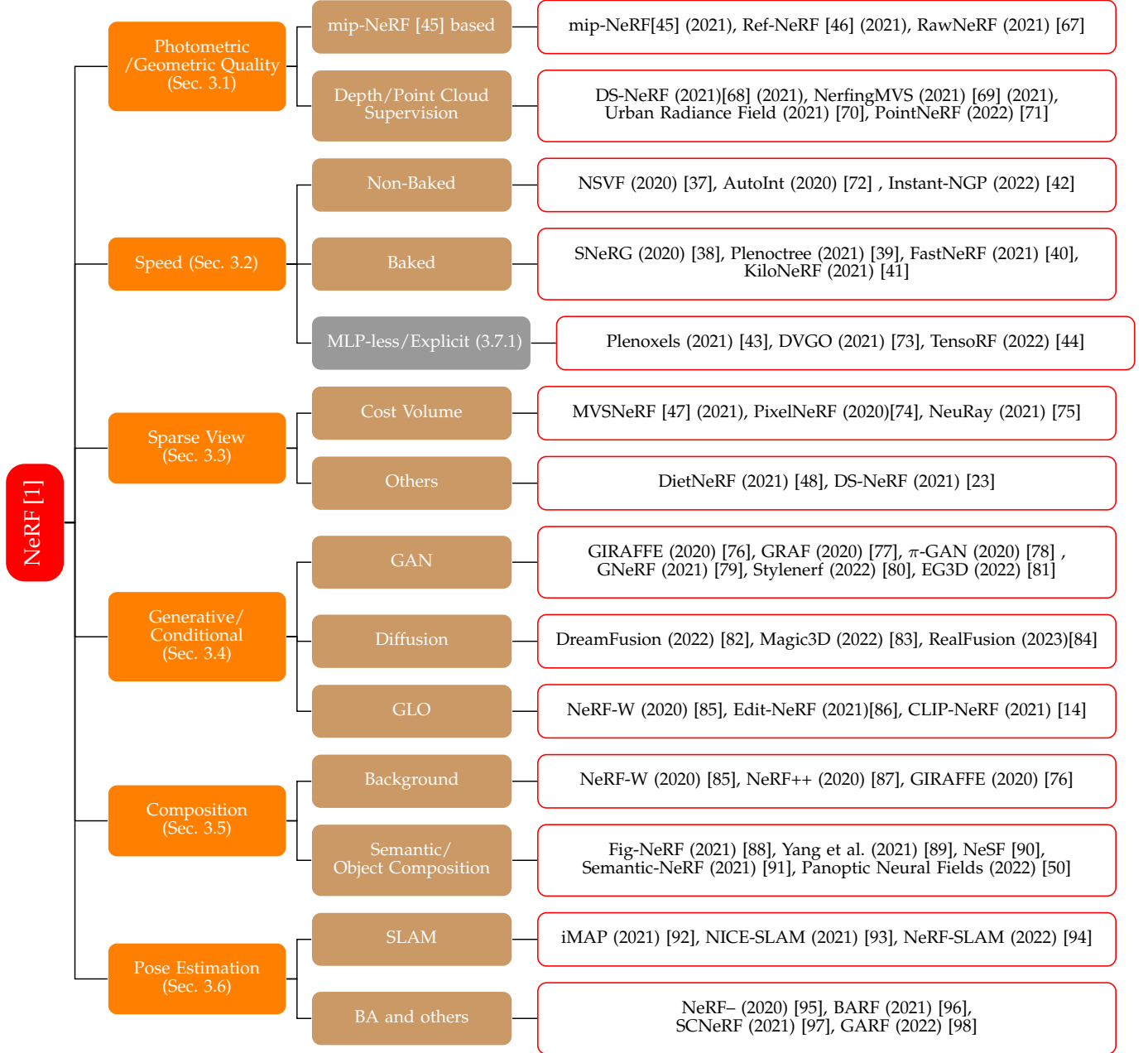


Fig. 3. Taxonomy of selected key NeRF innovation papers. The papers are selected using a combination of citations and GitHub star rating. We note that the **MLP-less speed-based models are not strictly speaking NeRF models**. Nonetheless, we decided to include them in this taxonomy tree due to their recent popularity and their similarity to speed based NeRF models.

### 3.2 Improvements to Training and Inference Speed

In the original implementation by Mildenhall et al. [1], to improve computation efficiency, a hierarchical rendering was used. A naive rendering would require densely evaluating MLPs at all query points along each camera ray during the numerical integration (2). In their proposed method, they used two networks to represent the scene, one coarse and one fine. The output of the coarse network was used to pick sampling points for the fine network, which prevented dense sampling at a fine scale. In subsequent works, most attempts to improve NeRF training and inference speed can be broadly classified into the two following categories.

- 1) Models in the first category train, precompute, and

store NeRF MLP evaluation results into more easily accessible data structures. This only improves inference speed, albeit by a large factor. We refer to these models as baked models.

- 2) The second category is comprised of non-baked models, which include multiple types of innovations. These models commonly (but not always) attempt to learn separate scene features from the learned MLP parameters. This allows for smaller MLPs (for example, by learning and storing features in a voxel grid, which are then fed into MLPs that produce color and density), which can improve both training and inference speed at the cost of memory. These models have hybrid scene representations.



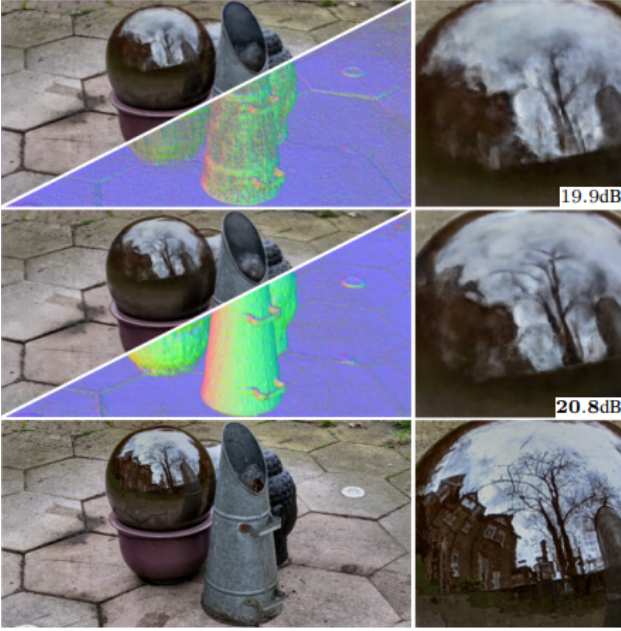


Fig. 4. Ref-NeRF results in the “garden spheres” scene of [38], showing its high performance on reflective scenes and its capability to recover accurate normal vectors of reflective surfaces. (Figure 8 in [46]). Top: mip-NeRF [45]; middle: ref-NeRF [46]; bottom: ground truth.

It’s worth noting that by taking this idea to the limit, certain methods (see Sec. 3.7.1) forgo neural networks entirely and have purely explicit scene representations. In this sense, they are neural rendering models but not NeRF models. However, we include them in this survey due to their relevance and their similarity to NeRF.

Other techniques such as ray termination (prevent further sampling points when accumulated transmittance approaches zero), empty space skipping, and/or hierarchical sampling (coarse+fine MLPs used in the original NeRF paper). These are also often used to further improve training and inference speed in conjunction to per-paper innovations.

A popular early re-implementation of the original NeRF in JAX [111], called **JaxNeRF** [36] (December 2020), was often used as benchmark comparison. This model was slightly faster and more suited for distributed computing than the original TensorFlow implementation.

In addition, a recent trend (2022) introduced multiple NeRF adjacent methods which are based on category 2), using learned voxel/tree features. However, these methods skip over entirely the MLPs and performed volume rendering directly on the learned features. These are introduced in a later section (3.7.1) since by not using neural networks, they are in some sense not neural radiance fields.

### 3.2.1 Baked

A model by Hedman et al. [38] (July 2021) stored a pre-computed NeRF on a sparse voxel grid. The method, called **Sparse Neural Voxel Grid (SNeRG)**, stored precomputed diffuse color, density, and feature vectors on a sparse voxel grid in a process sometimes referred to as “baking”. During evaluation time, an MLP was used to produce specular

color, which combined with the specular color’s alpha composited along the ray, produced the final pixel color. The method was 3000 times faster than the original implementation, with speed comparable to PlenOctree.

Concurrently, the **PlenOctree** [39] approach by Yu et al. (March 2021) achieved an inference time that was 3000 times faster than the original implementation. The authors trained a spherical harmonic NeRF (NeRF-SH), which predicted the spherical harmonic coefficients of the color function instead of directly predicting the color function. They built an octree of precomputed spherical harmonic coefficients of the MLP’s colors. During the building of the octree, the scene was first voxelized, with low transmissivity voxels eliminated. This procedure could also be applied to standard NeRF (non-NeRF-SH models) by performing Monte Carlo estimations of the spherical harmonic components of the NeRF. PlenOctrees could be further optimized using the initial training images. This fine-tuning procedure was fast relative to the NeRF training.

In **FastNeRF** [40] (March 2021), Garbin et al. factorized the color function  $c$  into the inner product of the output of the direction position-dependent MLP (which also produces the density  $\sigma$ ) and the output of a direction-dependent MLP. This allowed Fast-NeRF to easily cache color and density evaluations in a dense grid of the scene, greatly improving inference time by a factor of 3000+. They also included hardware-accelerated ray tracing [112], which skipped empty spaces and stopped when the ray’s transmittance was saturated.

Reiser et al. [41] (May 2021) improved on the baseline NeRF by introducing **KiloNeRF**, which separated the scene into thousands of cells and trained independent MLPs for color and density predictions on each cell. These thousands of small MLPs were trained using knowledge distillation from a large pre-trained teacher MLP, which is closely related to “baking”. They also employed early ray termination and empty space skipping. These two methods alone improved the baseline NeRF’s render time by a factor of 71. Separating the baseline NeRF’s MLP into thousands of smaller MLPs further improved render time by a factor of 36, resulting in a total 2000-fold speedup in render time.

The **Fourier PlenOctree** [113] approach was proposed by Wang et al. in February 2022. It was built for human silhouette rendering since it used the domain specific technique of Shape-From-Silhouette. The approach also takes inspiration from generalizable image conditioned NeRFs such as [47] and [74]. It first constructed a coarse visual hull using sparse views predicted from a generalization NeRF and Shape-From-Silhouette. Then colors and densities were densely sampled inside this hull and stored on a coarse PlenOctree. Dense views were sampled from the PlenOctree, with transmissivity thresholding used to eliminate most empty points. For the remaining points, new leaf densities and SH color coefficients were generated and the PlenOctree was updated. Then a Fourier Transform MLP was used to extract Fourier coefficients the density and SH color coefficients, which were fed into an inverse discrete Fourier transform to restore SH coefficients and density. According to the authors, using the frequency domain helped the model encode time-dependent information for dynamic scenes, such as the moving silhouettes modelled in the paper. The

SH coefficients were then used to restore color. The Fourier Plenocree can be fine-tuned on a per scene basis using the standard photometric loss (8).

A recent preprint (June 2022) proposed a lightweight method, **MobileNeRF** [114]. During training, MobileNeRF trains a NeRF-like models based on a polygonal mesh with color, feature, and opacity MLPs attached to each mesh point. Alpha values were then discretized, and features were super-sampled for anti-aliasing. During rendering, the mesh with associated features and opacities are rasterized based on viewing position, and a small MLP is used to shade each pixel. The method was shown to be around 10 times faster than SNeRG [38].

**EfficientNeRF** [115] (July 2022) was based on PlenOcree [39], choosing to use spherical harmonics and to cache the trained scene in a tree. However, it made several improvements. Most importantly, EfficientNeRF improved the training speed by using momentum density voxel grid to store predicted density using exponential weighted average update. During the coarse sampling stage, the grid was used to discard sampling points with zero density. During the fine sampling stage, a pivot system was also used to speed up volume rendering. Pivot points were defined as points  $\mathbf{x}_i$  for which  $T_i \alpha_i > \epsilon$  where  $\epsilon$  is a predefined threshold, and  $T_i$  and  $\alpha_i$  are the transmittance and alpha values as defined in (4) and (5). During fine sample, only points near the pivot points are considered. These two improvements speed up the training time by a factor of 8 over the baseline NeRF [1]. The authors then cached the trained scene into a NeRF tree. This resulted in rendering speed comparable to FastNeRF [40], and exceeding that of baseline NeRF by thousands fold.

**NeLF** (Mar 2022) instead chose to distill pretrained neural radiance fields into surface light fields using much deeper residual MLPs, achieving both better view synthesis quality (+1.40 PNSR on the NeRF Synthetic Dataset) and rendering speed (30x).

### 3.2.2 Non-Baked

In **Neural Sparse Voxel Fields (NSVF)** (July 2020), Liu et al. [37] developed a voxel-based NeRF model that models the scene as a set of radiance fields bounded by voxels. Feature representations were obtained by interpolating learnable features stored at voxel vertices, which were then processed by a shared MLP that computed  $\sigma$  and  $\mathbf{c}$ . NSVF used a sparse voxel intersection-based point sampling for rays, which was much more efficient than dense sampling or the hierarchical two-step approach of Mildenhall et al. [1]. However, this approach was more memory-intensive due to storing feature vectors on a potentially dense voxel grid.

**AutoInt** (Dec 2020) [72] approximates the volume rendering step. By separating the discrete volume rendering equation (4) piecewise, they developed AutoInt, which trains the MLP  $\Phi_\theta$  by training its gradient (grad) networks  $\Psi_\theta^i$ , which share internal parameters with and are used to reassemble the integral network  $\Phi_\theta$ . This allows for the rendering step to use much fewer samples, resulting in a ten times speed-up over the baseline NeRF with a slight decrease in quality.

**Deterministic Integration for Volume Rendering (DIVER)** [116] (November 2021) took inspiration from NSVF

[37], also jointly optimizing a feature voxel grid and a decoder MLP while performing sparsity regularization and voxel culling. However, they innovated on the volume rendering, using a technique unlike NeRF methods. DIVER performed deterministic ray sampling on the voxel grid which produced an integrated feature for each ray interval (defined by the intersection of the ray with a particular voxel), which was decoded by an MLP to produce density and color of the ray interval. This essentially reversed the usual order between volume sampling and MLP evaluation. The method was evaluated on the NeRF Synthetic [1], BlendedMVS [117] and Tanks and Temple datasets [33], outperforming methods such as PlenOcree [39], FastNeRF [40] and KiloNeRF [41] in terms of quality, at comparable a rendering speed.

A recent innovation by Muller et al., dubbed **Instant-Neural Graphics Primitives (Instant-NGP)** [42] (January 2022) greatly improved NeRF model training and inference speed. The authors proposed a learned parametric multi-resolution hash encoding that was trained simultaneously with the NeRF model MLPs. They also employed advanced ray marching techniques including exponential stepping, empty space skipping, sample compaction. This new positional encoding and associated optimized implementation of MLP greatly improved training and inference speed, as well as scene reconstruction accuracy of the resulting NeRF model. Within seconds of training, they achieved similar results to hours of training in previous NeRF models.

### 3.3 Few Shot/Sparse Training View NeRF

The baseline NeRF requires dense multi-view images with known camera poses for each scene. A common failure case for the baseline NeRF is if the training views are not varied enough, whether enough when training samples are too few, or the samples are not varied enough in pose. This leads to overfitting to individual views, and non-sensible scene geometry. However, a family of NeRF methods leverage pretrained image feature extraction networks, usually a pretrained Convolutional Neural Network (CNN) such as [66][118], to greatly lower the number of training sample required for successful NeRF training. Some authors call this process "image feature conditioning". Certain methods [71] also used depth/3D geometry supervision to this effect (Sec. 3.1.2). These models often have lower training time compared to baseline NeRF models. Certain previously mentioned methods which regularizes or improve geometry also give reduce the training view requirement. Methods such as [23], [68] and [110] from section 3.1.2 and section 3.1.3 approach the sparse view problem by using point clouds for depth supervision.

In **pixelNeRF** [74] (December 2020), Yu et al. used the pretrained layers of a Convolutional Neural Networks (and bilinear interpolation) to extract image features. Camera rays used in NeRF were then projected onto the image plane and the image features were extracted for each query points. The features, view direction, and query points were then passed onto the NeRF network which produced density and color. General Radiance Field (GRF) [119] (Oct 2020) by Trevithick et al. took a similar approach, with the key difference being that GRF operated in canonical space as opposed to view-space for pixelNeRF.

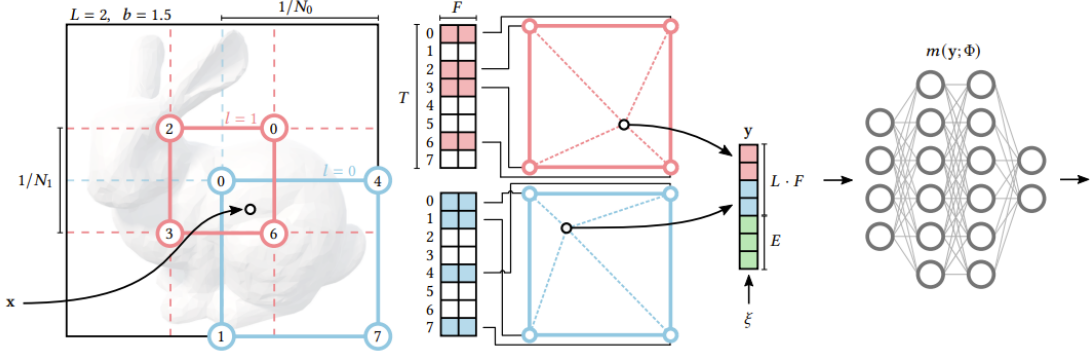


Fig. 5. Example of a hybrid representation scene representation method, the Instant-NGP [42]. During training, the scene information is stored in both the hashed hierarchical voxel grid (of number of levels  $L$ , feature dimension  $F$ , hash table size  $T$ , and resolution  $N_l$  at level  $l$ ) and the decoding MLP  $m(y, \Phi)$ . (See figure 3 in [42] for full details.)

**MVSNerF** [47] (March 2021) used a slightly different approach. They also extracted 2D image features using a pretrained CNN. These 2D features were then mapped to a 3D voxelized cost volume using plane sweeping and a variance based cost. A pretrained 3D CNN was used to extract a 3D neural encoding volume which was used to generate per-point latent codes using interpolation. When performing point sampling for volume rendering, the NeRF MLP then generated point density and color using as input these latent features, point coordinate and viewing direction. The training procedure involves the joint optimization of the 3D feature volume and the NeRF MLP. When evaluating on the DTU dataset, within 15 minutes of training, MVSNerF could achieve similar results to hours of baseline NeRF training.

**DietNeRF** [48] (June 2021) introduced the semantic consistency loss  $L_{sc}$  based on image features extracted from Clip-ViT [120], in addition to the standard photometric loss.

$$L_{sc} = \frac{\lambda}{2} \|\phi(I) - \phi(\hat{I})\|_2^2 \quad (19)$$

where  $\phi$  performs the Clip-ViT feature extraction on training image  $I$  and rendered image  $\hat{I}$ . This reduced to a cosine similarity loss for normalized feature vectors (eq. 5 in [48]). DietNeRF was benchmarked on a subsampled NeRF synthetic dataset [1], and DTU dataset [28]. The best performing method for single-view novel synthesis was a pixelNeRF [74] model fine-tuned using the semantic consistency loss of DietNeRF.

The **Neural Rays (NeuRay)** approach, by Liu et al. [75] (July 2021) also used a cost volume. From all input views, the authors estimated cost volumes (or depth maps) using multi-view stereo algorithms. From these, a CNN is used to create feature maps  $G$ . During volume rendering, from these features, both visibility and local features are extracted and processed using MLPs to extract color and alpha. The visibility is computed as a cumulative density function written as a weighted sum sigmoid functions  $\Phi$

$$v(z) = 1 - t(z), \text{ where } t(z) = \sum_{i=1}^N w_i \Phi((z - \mu_i)/\sigma_i) \quad (20)$$

where  $w_i, \mu_i, \sigma_i$  are decoded from  $G$  using an MLP. NeuRay also used an alpha based sampling strategy, by computing a

hitting probability, and only sampling around points with a high hitting probability (see Sec. 3.6 in [75] for details). Like other NeRF models conditioned on extracted image features from pre-trained neural networks, NeuRay generalizes well to new scenes, and can be further fine-tuned to exceed the performance of the baseline NeRF model. NeuRay outperformed MVSNerF on the NeRF synthetic dataset after fine-tuning both models for 30 minutes.

**GeoNeRF**[121] (November 2021) extracted 2D image features from every view using a pretrained Feature Pyramid Network. This method then constructed a cascading 3D cost-volume using plane sweeping. From these two feature representations, for each of the  $N$  query point along a ray, one view independent, and multiple view dependent feature tokens were extracted. These were refined using a Transformer [122]. Then, the  $N$  view-independent tokens are refined through an AutoEncoder, which returned the  $N$  densities along the ray. The  $N$  sets of view dependent tokens were each fed into an MLP which extracted color. These networks can all be pretrained and generalized well to new scenes as shown by the authors. Moreover, they can be fine tuned per-scene achieving great results on the DTU [28], NeRF synthetic [1], and LLF Forward Facing [5] datasets, outperforming methods such as pixelNeRF [74] and MVSNerF [47].

Concurrent to GeoNeRF is **LOLNeRF** (November 2021) [123], which is capable of single-shot view synthesis of human faces, and is built similarly to  $\pi$ -GAN [78], but uses Generative Latent Optimization [124] instead of adversarial training [125].

**NeRFusion** [126] (March 2022) also extracted a 3D cost volume from 2D image features extracted from CNN, which was then processed by a sparse 3D CNN into a local feature volume. However, this method performs this step for each frame, and then used a GRU [127] to fuse these local feature volumes into a global feature volume, which were used to condition density and color MLPs. NeRFusion outperformed IBRNet, baseline NeRF [1], NeRFingMVS[69], MVSNerF [47] on ScanNet [30], DTU [28], and NeRF Synthetic [1] datasets.

**AutoRF** [128] (April 2022) focused on novel view synthesis of objects without background. Given 2D multi-view images, a 3D object detection algorithm was used with

panoptic segmentation to extract 3D bounding boxes and object masks. The bounding boxes were used to define Normalized Object Coordinate Spaces, which were used for per-object volume rendering. An encoder CNN was used to extract appearance and shape codes which were used in the same way as in GRAF [77]. In addition to the standard photometric loss (8), an additional occupancy loss was defined as

$$L_{occ} = \frac{1}{|W_{occ}|} \sum_{u \in W_{occ}} \log(Y_u(1/2 - \alpha) + 1/2) \quad (21)$$

where  $Y$  is the object mask, and  $W_{occ}$  is either the set of foreground pixels, or background pixels. During test-time, the shape codes, appearance code, and bounding boxes were further refined using the same loss function. The method outperformed pixelNeRF [74] and CodeNeRF [129] for object-based novel view synthesis on the nuScene [130], SRN-Cars [6] and Kitti [52] datasets.

**SinNeRF** [25] (April 2022) attempted NeRF scene reconstruction from single images by integrating multiple techniques. They used image warping and known camera intrinsic and poses to create reference depth for depth-supervision of unseen views. They used adversarial training with a CNN discriminator to provide patch-wise texture supervision. They also used a pretrained ViT to extract global image features from the reference patch and unseen patch, comparing them with an L2 loss term and a global structure prior. SinNeRF outperformed DS-NeRF [23], PixelNeRF [74], and DietNeRF [48] on the NeRF Synthetic dataset [1], the DTU dataset [28], and the LLFF Forward Facing dataset [5].

As an alternate approach, **GeoAug** [131] (Oct 2022) sought to solve the few-shot problem using data augmentation by rendering (with warping) new training images with new noisy camera poses using DSNeRF [68] as baseline.

### 3.4 Generative and Conditional Models

Inspired by the development in generative 2D computer vision, the generative NeRF models generate 3D geometry conditioned on texts, images, or latent codes. This conditioning also allow for a degree of scene editing. These models can be broadly classified into Generative Adversarial Network-based methods and Diffusion methods. These Generative NeRF models typically make use of 2D generative models to create the images of the ‘scene’ which are then used to train the NeRF model. A key challenge include the control of pose in conjunction with keeping the subject invariant. Additionally, methods conditioned on latent codes for scene editing using generative latent optimization can also be found in this section, however, models which are conditioned on latent codes but which do not focus on scene editing are not classified into this section.

In **NeRF-VAE** [132] (January 2021), Kosiorek et al. proposed a generative NeRF model which generalized well to out-of-distribution scenes, and removed the need to train on each scene from scratch. The NeRF renderer in NeRF-VAE was conditioned on latent code which was trained using Iterative Amortized Inference [133][134] and a ResNet [118] based encoder. The authors also introduced an attention-based scene function (as opposed to the typical MLP). NeRF-VAE consistently outperformed the baseline NeRF with low

number (5-20) of scene views, but due to lower scene expressivity, was outperformed by baseline NeRF when large number of views were available (100+).

#### 3.4.1 Generative Adversarial Network-based methods

Adversarial training is often used for generative and/or latent conditioned NeRF models. First developed in 2014, Generative Adversarial Networks (GANs) [125] are generative models which employ a generator  $G$  which synthesizes images from “latent code/noise”, and a discriminator  $D$  which classifies images as “synthesized” or “real”. The generator seeks to “trick” the discriminator, and make its images indistinguishable from “real” training images. The discriminator seeks to maximize its classification accuracy. These two networks are trained adversarially, which is the optimization of the following minimax loss/value function,

$$\min_G \max_D \mathbb{E}_{x \sim data} [\log D(x)] + \mathbb{E}_{z \sim p(z)} [\log(1 - D(G(z)))] \quad (22)$$

where the generator generates images based on latent code  $z$  sampled from some distribution  $p(z)$ , which the discriminator compares to training image  $x$ . In GAN-based NeRF model, the generator  $G$  encompasses all novel-view synthesis steps, and is transitionally thought of as the NeRF model. The generator in this case also requires an input pose, in addition to a latent code. The discriminator  $D$  is usually an image classification CNN. **GRAF** [77] (July 2020) was the first NeRF model trained adversarially. It paved way for many later works. A NeRF based generator was conditioned on latent appearance code  $\mathbf{z}_a$  and shape code  $\mathbf{z}_s$ , and is given by

$$G(\gamma(\mathbf{x}), \gamma(\mathbf{d}), \mathbf{z}_s, \mathbf{z}_a) \rightarrow (\sigma, \mathbf{c}). \quad (23)$$

In practice, the shape code, conditioning scene density, was concatenated with the embedded position, which was input to the direction independent MLP. The appearance code, conditioning scene radiance, was concatenated with the embedded viewing direction, which was input to the direction dependent MLP. As per baseline NeRF, images were generated via volume sampling. These were then compared using a discriminator CNN for adversarial training.

Within three months of GRAF, Chan et al. developed  **$\pi$ -GAN** [78] (December 2020) which also used a GAN approach to train a conditional NeRF model. The generator was a SIREN-based [135] NeRF volumetric renderer, with sinusoidal activation replacing the standard ReLU activation in the density and color MLPs.  $\pi$ -GAN outperformed GRAF[77] on standard GAN datasets such as Celeb-A[136], CARLA [137] and Cats [138].

**EG3D** [81] (December 2021) uses a novel hybrid tri-plane representation with features stored on three axis aligned planes and a small decoder MLP for neural rendering in a GAN framework. The GAN framework is comprised of a pose-conditioned StyleGAN2 feature map generator for the tri-plane, a NeRF rendering module converting tri-plane features into low-resolution images, and a super-resolution module. The super-resolved image is then fed-into a StyleGAN2 discriminator. The model achieved state-of-the-art results on the FFHQ dataset producing realistic images and 3D geometry of human faces.



**StyleNeRF** [80] (January 2022) is a highly influential work which focuses on 2D image synthesis, by using NeRF to bring 3D awareness to the StyleGAN [139][140] image synthesis framework. StyleNeRF uses a style code conditioned NeRF with upsampling module as Generator, and StyleGAN2 [140] Discriminator, and introduces a new path regularization term to the StyleGAN optimization objective.

**Pix2NeRF** [141] (February 2022) was proposed as an adversarial trained NeRF model which could generate NeRF rendered images given randomly sampled latent codes and poses. Built from  $\pi$ -GAN [78], It is composed of a NeRF based generator  $G : d, z \rightarrow I$ , a CNN discriminator  $D : I \rightarrow d, l$  and an Encoder  $E : I \rightarrow d, z$ .  $z$  is a latent code sampled from a multi-variate distribution,  $d$  is a pose,  $I$  is a generated image, and  $l$  is a binary label for real vs. synthesized image. In addition to the  $\pi$ -GAN loss, from which the adversarial architecture is based, the pix2NeRF loss function also include the following: 1) a reconstruction loss comparing  $z_{predicted}$  and  $z_{sampled}$  to ensure consistency of latent space, 2) a reconstruction loss ensuring image reconstruction quality, between  $I_{real}$  and  $I_{reconstructed}$ , where  $I_{reconstructed}$  is created by the Generator from a  $z_{pred}, d_{pred}$  pair produced by the Encoder 3) a conditional adversarial objective which prevents mode collapse towards trivial poses (see original paper for the exact expressions). The model achieved good results on the CelebA dataset [136], the CARLA dataset [137], and the ShapeNet subclasses from SRN [6], but is outperformed by its backbone  $\pi$ -GAN for conditional image synthesis.

### 3.4.2 Jointly Optimized Latent Models

These model use latent codes as a key aspect of view-synthesis, but jointly optimize the latent code with the scene model. The models listed in this section are not generative, but instead use latent codes account for various changeable aspects of the scene. One should note that many NeRF models use latent codes to condition for certain aspects of the scene such as the appearance and transient embeddings in NeRF-W. These models are typically optimized using Generative Latent Optimization (GLO) [124]. We do not list them in this section unless the latent code is used explicitly for scene editing as the key idea in the paper. In GLO, a set of randomly sampled latent codes  $\{z_1, \dots, z_n\}$ , usually normally distributed, is paired to a set of images  $\{I_1, \dots, I_n\}$ . These latent codes are input to a generator  $G$  whose parameters are jointly optimized with the latent code using some reconstruction loss  $L$  such as  $L_2$ . I.e., the optimization is formulated as

$$\min_{G, z_1, \dots, z_n} \sum_{i=1}^n L(G(z_i, \mathbf{u}_i), I_i) \quad (24)$$

where  $\mathbf{u}_i$  represent the other inputs not optimized over (needed in NeRF but not necessarily for other models). According to the GLO authors, this method can be thought of as a Discriminator-less GAN.

**Edit-NeRF** [86] (June 2021) allowed for scene editing using image conditioning from user input. Edit-NeRF’s shape representation was composed of a category specific shared shape network  $F_{share}$ , and a instance specific shape network  $F_{inst}$ .  $F_{inst}$  was conditioned on  $\mathbf{z}_s$  whereas  $F_{shared}$  was

not. In theory, the  $F_{shared}$  behaved as a deformation field, not unlike [57]. The NeRF editing was formulated as a joint optimization problem of both the NeRF network parameters and the latent codes  $\mathbf{z}_s, \mathbf{z}_a$ , using GLO. They then conducted NeRF photometric loss optimization on latent codes, then on the MLP weights, and finally optimized both latent codes and weights jointly.

Innovating on Edit-NeRF, **CLIP-NeRF**’s [14] (December 2021) neural radiance field was based on the standard latent conditioned NeRF, i.e., NeRF models conditioned on shape and appearance latent codes. However, by using Contrastive Language Image Pre-training (CLIP), CLIP-NeRF could extract from user input text or images the induced latent space displacements by using shape and appearance mapper networks. These displacements could then be used to modify the scene’s NeRF representation based on these input text or images. This step allowed for skipping the per-edit latent code optimization used in Edit-NeRF, resulting in a speed-up of a factor of  $\sim 8$ -60, depending on the task. They also used a deformation network similar to deformable NeRFs [57] (called *instance-specific-network*) in Edit-NeRF [86] to help with modifying the scene based on latent space displacement. The model was trained in two stages. In the first, CLIP-NeRF’s conditional latent NeRF models was trained adversarially. In the second, the CLIP shape and appearance mappers were trained using self-supervision. When applying CLIP-NeRF to images with unknown pose and latent codes, the authors used an Expectation-Maximization algorithm which then allowed them to use CLIP-NeRF latent code editing. CLIP-NeRF outperformed EDIT-NeRF in terms of inference speed, and post-edit metrics, especially for global edits, which was weakness of Edit-NeRF.

### 3.4.3 Diffusion NeRF Models

Diffusion models are a family of generative methods that have recently attracted widespread attention. The forward diffusion process adds Gaussian noise to some input image/feature map in some T steps [142]. The reverse process is generative and can be used to create images from Gaussian noise [143][144]. Diffusion models offer a high degree of control over the generation by allowing for text and image-based prompts by using domain-specific encoders. Recent research has produced diffusion NeRF models by training NeRF models from scratch on images generated from diffusion methods, which can generate views of the same object under different poses with appropriate prompting.

**DreamFusion** [82] (September 2022) was proposed as a text-to-3D diffusion NeRF model. The NeRF model in DreamFusion was trained using images from 2D diffusion models. For each object or scene to-be-generated, text prompts are input into the diffusion model Imagen [143], and a mip-NeRF 360 [99] based NeRF model was trained from scratch. The text prompt allowed for control over the view of the subject at the diffusion image generation stage (with some prompting using keywords such as “overhead view”, “front view”, “back view”). A key modification to the NeRF training was that surface color was parameterized by an MLP instead of radiance. Despite impressive results, Imagen images were generated at a 64x64 resolution. As such, the resulting NeRF model lacked certain finer details.



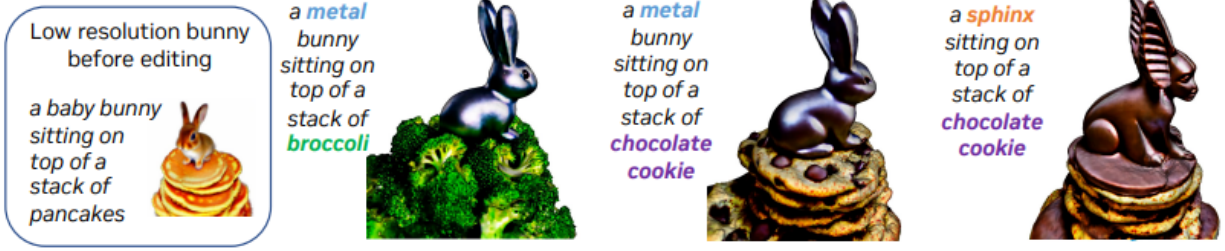


Fig. 6. Magic3D: Example two-stage text-to-3D generation (Figure 1 in [83]). Left: low resolution text-to-3D using NeRF; middle and right: higher resolution text-to-3D with text prompt editing of low resolution mesh.

In **Latent-NeRF** (November 2022) [145] the NeRF model was trained to output  $64 \times 64 \times 4$  latent features that Stable Diffusion [144] operated over, which then resulted in  $512 \times 512 \times 3$  RGB images after a Decoder step. The method allowed for text guidance, and shape guidance (both for further shape refinement and as a strict shape constraint).

Building upon DreamFusion, **Magic3D** [83] (November 2022) targeted the issues caused by the low resolution diffusion images. The authors used a two stage coarse-fine approach. In the coarse stage, the Magic3D used Instant-NGP [42] as the NeRF model trained from images generated from text prompts using image diffusion model eDiff-I [146]. The coarse geometry extracted from Instant-NGP was then placed on a mesh which was optimized over in the fine stage using images generated with a latent diffusion model [144]. The authors noted that their method allowed for prompt based scene editing, personalized text-to-3D generation via conditioning on image of a subject, and style-guided text-to-3D generation. Their experiments over 397 prompts generated objects each rated by 3 users also showed a preference towards Magic3D over DreamFusion.

**RealFusion** [84] (February 2023) also used some of the same ideas, but focused on single-shot scene learning. The underlying diffusion model is Stable Diffusion [144] and the underlying NeRF model is Instant-NGP [42]. The authors used single image textual inversion as substitute for alternate views, by augmented the input 2D image, and associated it with a new vocabulary token to optimize the diffusion loss which insures the radiance field represents the object in the single view photography. The 3D scene was then trained in a coarse-to-fine manner using the NeRF photometric loss, which reconstructs the scene.

In addition to these generative diffusion models, recent preprints have used diffusion models for single-view scene learning via image conditioning (**NeuralLift-360** (Nov 2022) [147], **NeRFDi** (December 2022) [148], **NerfDiff** (February 2023) [149]), as well as for geometry regularization (**DiffusioNeRF** (February 2023) [150]).

### 3.5 Unbounded Scene and Scene Composition

With attempts at applying NeRF models to outdoor scenes came a need to separate foreground and background. These outdoor scenes also posed additional challenges in image-by-image variation in lighting and appearance. The models introduced in this section approached this problem using various methods, with many models adapting the latent conditioning via image-by-image appearance codes. Certain models in this research area also perform semantic

or instance segmentation find applications in 3D semantic labelling.

In **NeRF in the Wild (NeRF-W)** [85] (August 2020), Martin-Brualla et al. addressed two key issues of baseline NeRF models. Real-life photographs of the same scene can contain per-image appearance variations due to lighting conditions, as well as transient objects which are different in each image. The density MLP was kept fixed for all images in a scene. However, NeRF-W conditioned their color MLP on a per-image appearance embedding. Moreover, another MLP conditioned on per-image transient embedding predicted the color and density functions of transient objects. These latent embeddings were constructed using Generative Latent Optimization. NeRF-W did not improve on NeRF in terms of rendering speed, but achieved much higher results in the crowded Phototourism dataset [151].

Zhang et al. developed the **NeRF++** [87] (October 2020) model, which was adapted to generate novel views for unbound scenes, by separating the scene using a sphere. The inside of the sphere contained all foreground object and all fictitious cameras views, whereas the background was outside the sphere. The outside of the sphere was then reparametrized using as an inverted sphere space. Two separate NeRF models were trained, one for the inside the sphere and one for the outside. The camera ray integral was also evaluated in two parts. Using this approach, they outperformed the baseline NeRF on Tanks and Temples [33] scenes as well as scenes from Yucer et al. [152]. Zhang et al. also offered justifications as to how NeRF models resolve certain shape-radiance ambiguity. According to them, the wrong density configuration tends to result in color configurations with high frequency components with respect to viewing angles. However by its construction (the use of lower frequency component and the introduction of view angles at the later stages to the MLP) NeRF models tend to produce smoother color configurations.

**GIRAFFE** [76] (November 2020) was built with a similar approach to NeRF-W, using generative latent codes and separating background and foreground MLPs for scene composition. GIRAFFE was based on GRAF, a previous model used for generative scene modeling. It assigned to each object in the scene an MLP, which produced a scalar density and a deep feature vector (replacing color). These MLPs, with shared architecture and weights, took as input shape and appearance latent vectors, as well as an input pose. The background was treated as all other objects, except with its own MLP and weights. The scene was then composed using a density-weighted sum of features.

A small 2D feature map was then created from this 3D volume feature field using volume rendering, which was fed into an upsampling CNN to produce an image. GIRAFFE performed adversarial training using this synthesized image and a 2D CNN discriminator. The resulting model had a disentangled latent space, allowing for fine control over scene generation.

**Fig-NeRF** [88] (April 2021) also took on scene composition, but focused on object interpolation and amodal segmentation. They used two separate NeRF models, one for the foreground, one for the background. Their foreground model was the deformable Nerfies model [57]. Their background model was an appearance latent code conditioned NeRF. They used two photometric losses, one for the foreground, one for the background. Fig-NeRF achieved good results for amodal segmentation and object interpolation, on datasets such as ShapeNet [32] Gelato [153], and Objectron [154].

**Yang et al.** [89] (September 2021) created composition model which can edit objects within the scene. They used a voxel-based approach [37], creating a voxel grid of features which is jointly optimized with MLP parameters. They used two different NeRFs, one for objects, one for the scene, both of which were conditioned on interpolated voxel features. The object NeRF was further conditioned on a set of object activation latent codes. Their method was trained and evaluated on ScanNet[30] as well as an inhouse ToyDesk dataset with instance segmentation labels. They incorporated segmentation labels with a mask loss term given by

$$w(\mathbf{r})_k \|\hat{O}(\mathbf{r})_k - M(\mathbf{r})\|_2^2 \quad (25)$$

where  $\hat{O}(\mathbf{r})_k = \sum_{i=1}^N T_i \alpha_i$  is the 2D object opacity,  $M(\mathbf{r})$  is the  $k$ th object mask of 0s and 1s, and  $w(\mathbf{r})_k$  is the balance weight between 0s and 1s in the mask. The authors edited objects within the scene by first obtaining the background colors and densities from the scene NeRF branch, pruning sample points at the object’s location. Then the object’s colors and densities are obtained from the object NeRF, and transformed according to user defined manipulations. Finally, all colors and densities are aggregated according to the discrete volume rendering equation (4). The authors’s method outperformed baseline NeRF [1] as well as NSVF [37] on both their inhouse dataset and ScanNet.

**NeRFReN** [24] (November 2021) addressed the problem of reflective surfaces in NeRF view synthesis. The authors separated the radiance field into two components, transmitted ( $\sigma^t, \mathbf{c}^t$ ) and reflected ( $\sigma^r, \mathbf{c}^r$ ), with the final pixel value given by

$$I = I_t + \beta I_r \quad (26)$$

where  $\beta$  is the reflection fraction given by the geometry of the transmitted radiance field as

$$\beta = \sum_i T_{\sigma_i^t} (1 - \exp(-\sigma_i^t \delta_i)) \alpha_i. \quad (27)$$

$T_{\sigma_i^t}$  is given by (3), and  $\alpha_i$  by (5). In addition to the standard photometric loss, the authors used a depth smoothness  $L_d$  (eq. 8 in [24]) loss to encourage the transmitted radiance field to have the correct geometry, and likewise, a bidirectional depth consistency loss  $L_{bdc}$  (eq. 10 in [24]) for the reflected radiance field. NeRFReN was able to render

reflective surface on the authors’ RFFR dataset, outperforming benchmark methods such as baseline NeRF [1], and NerfingMVS [69], as well as ablation models. The method was shown to support scene editing via reflection removal and reflection substitution.

### 3.6 Pose Estimation

NeRF models require both input images and camera poses to train. In the original 2020 paper, unknown poses were acquired by the COLMAP library [2], which was also often used in many subsequent NeRF models when camera poses were not provided. Typically, building models which perform both pose estimation and implicit scene representation with NeRF is formulated as an offline structure from motion (SfM) problem. In these cases, Bundle Adjustment (BA) is often used to jointly optimize the poses and the model. However, some methods also formulate this as an online simultaneous localization and mapping (SLAM) problem.

**iNeRF** [155] (December 2020) formulated pose reconstruction as an inverse problem. Given a pre-trained NeRF, using the photo-metric loss 8, the Yen-Chen et al. optimized the pose instead of the network parameters. The authors used an interest-point detector, and performed interest region-based sampling. The authors also performed semi-supervision experiments, where they used iNeRF pose estimation on unposed training images to augment the NeRF training set, and further train the forward NeRF. This semi-supervision was shown by the author to reduce the requirement of posed photos from the forward NeRF by 25 %.

**NeRF-** [95] (February 2021) however jointly estimated NeRF model parameters and camera parameters. This allowed for the model to construct radiance fields and synthesize novel views only images, in an end-to-end manner. NeRF- overall achieved comparable results to using COLMAP with the 2020 NeRF model in terms of both view synthesis. However, due to limitations with pose initialization, NeRF- was most suited for front facing scenes,

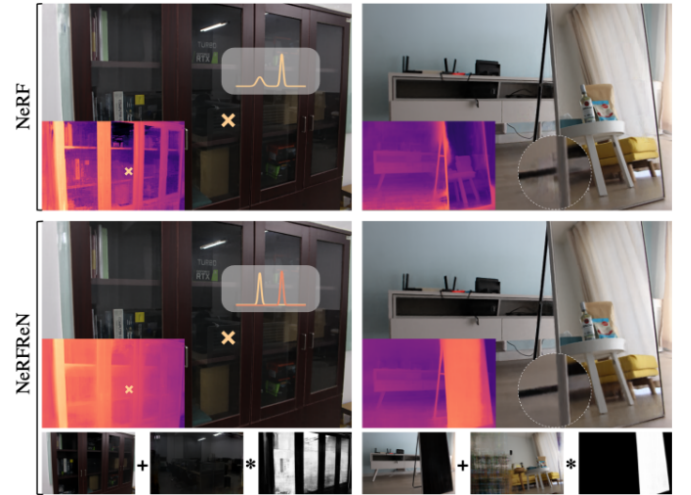


Fig. 7. NeRFReN [24] is capable of accurately reconstructing the depth (and geometry) of reflective surfaces such as glasses, a common failure cases for standard approaches (Figure 1 in [24]).

and struggled with rotational motion and object tracking movements.

Concurrent to NeRF- was the **Bundle-Adjusted Neural Radiance Field (BARF)** [96] (April 2021), which also jointly estimated poses alongside the training of the neural radiance field. BARF also used a coarse-to-fine registration by adaptively masking the positional encoding, similar to the technique used in Nerfies [57]. Overall, BARF results exceeded those of NeRF- on the LLFF forward facing scenes dataset with unknown camera poses by 1.49 PNSR averaged over the eight scenes, and outperformed COLMAP registered baseline NeRF by 0.45 PNSR. Both BARF and NeRF- used naive dense ray sampling for simplicity.

Jeong et al. introduced a self-calibrating joint optimization model for NeRF (**SCNeRF**) [97] (August 2021). Their camera calibration model can not only optimize unknown poses, but also camera intrinsic for non-linear camera models such as fish-eye lens models. By using curriculum learning, they gradually introduce the nonlinear camera/noise parameters to the joint optimization. This camera optimization model was also modular and could be easily used with different NeRF models. The method outperformed BARF [96] on LLFF scenes [5].

**GNeRF**, a different type of approach by Meng et al [79] (March 2021), used pose as generative latent code. GNeRF first obtains coarse camera poses and radiance field with adversarial training. This is done by using a generator which takes a randomly sampled pose, and synthesized a view using NeRF-style rendering. Then a discriminator compared the rendered view with the training image. An inversion network then took the generated image, and output a pose, which was compared to the sampled pose. This resulted in a coarse image-pose pairing. The images and poses were then jointly refined via a photometric loss in a hybrid optimization scheme. GNeRF was slightly outperformed by COLMAP based NeRF on the Synthetic-NeRF dataset, and outperformed COLMAP based NeRF on the DTU dataset.

**GARF** [98] (April 2022) used Gaussian activations as an effective alternative to positional encoding in NeRF, in conjunction with bundle adjustment for pose estimation. The authors showed that GARF can successfully recover scene representations from unknown camera poses, even in challenging scenes with low-textured regions, making it suitable for real-world applications.

### 3.6.1 NeRF and SLAM

Building on iMAP, **NICE-SLAM** [93] (December 2021) improved on various aspects such as keyframe selection and NeRF architecture. Specifically, they used a hierarchical grid based representation of the scene geometry, which was able to fill in gaps iMAP reconstruction in large scale unobserved scene features (walls, floors etc.) for certain scenes. NICE-SLAM achieved lower pose estimation errors and better scene reconstruction results than iMAP. The NICE-SLAM also only used  $\sim 1/4$  of the FLOPs of iMAP,  $\sim 1/3$  the tracking time and  $\sim 1/2$  the mapping time.

Sucar et al. introduced the first NeRF-based dense online SLAM model named **iMAP** [92] (March 2021). The model jointly optimizes camera pose and the implicit scene representation in the form of a NeRF model, making use of continual online learning. They used an iterative two-step

approach of tracking (pose optimization with respect to NeRF) and mapping (bundle-adjustment joint optimization of pose and NeRF model parameters). iMAP achieved a pose tracking speed close to camera framerate by running the much faster tracking step in parallel to the mapping process. iMAP also used keyframe selection by optimizing the scene on a sparse and incrementally selected set of images.

**NeRF-SLAM** [94] (October 2022) improved on existing combined NeRF-based SLAM approach by using Instant-NGP [42] as NeRF model, in conjunction with a state-of-the-art SLAM pipeline, greatly exceeding previous benchmarks on the Replica dataset [35].

## 3.7 Adjacent Methods for Neural Rendering

### 3.7.1 Explicit Methods: Fast MLP-less Volume Rendering

**Plenoxel** [43] (December 2021) followed in Plenocree’s footsteps in voxelizing the scene and storing a scalar for density and spherical harmonics coefficients direction dependent color. However, surprisingly, Plenoxel skipped the MLP training entirely, and instead fit these features directly on the voxel grid. They also obtained comparable results to NeRF ++ and JaxNeRF, with faster training times by a factor of a few hundreds. These results showed that the primary contribution of NeRF models is the volumetric rendering of new view given densities and colors, and not the density and color MLPs themselves. **HDR-Plenoxels** [156] (August 2022) adapted this idea to HDR images by learning 3D High Dynamic Range radiance fields, scene geometry, and various camera settings from Low Dynamic Range images.

A concurrent paper by Sun et al. [73] (November 2021) also explored this topic. The authors also directly optimized a voxel grid of scalars for density. However, instead of using spherical harmonic coefficient, the authors use 12 and 24 dimensional features, and a small shallow decoding MLP. The authors used a sampling strategy analogous to the coarse-fine sampling of the original NeRF paper by training a coarse voxel grid first, and then a fine voxel grid based on the geometry of the coarse grid. The model was named **Direct Voxel Grid Optimization (DVGO)**, which outperformed the baseline NeRF (1-2 days) of training with only 15 minutes of training on the Synthetic-NeRF dataset. The authors obtained a PSNR of 31.95 at voxel resolution  $160^3$  after 11 minutes of training, and a PSNR of 32.80 at voxel resolution of  $256^3$  after 22 minutes of training on a 2080Ti. They outperformed Plenoxel’s  $512^3$  resolution model’s 31.71 PSNR after 11 minutes of training on an RTX Titan.

**TensoRF** [44] (March 2022) stored a scalar density and a vector feature (can work with SH coefficients, or features to be decoded via MLP) in a 3D voxel grid, which were then represented as a rank 3 tensor  $T_\sigma \in R^{H \times W \times D}$  and a rank 4 tensor  $T_c \in R^{H \times W \times D \times C}$ , where  $H, W, D$  are the height, width and depth resolution of the voxel grid, and  $C$  is channel dimension. Then the authors used two factorization schemes: CANDECOMP-PARAFAC (CP) which factorized the tensors as pure vector outer products and Vector Matrix (VM) which factorized the tensors as vector/matrix outer products. These factorizations decreased the memory requirement from Plenoxels by a factor of 200 when using CP. Their VM factorization performed better in terms of



visual quality, albeit at a memory tradeoff. The training speed was comparable to Pleoxels and much faster than the benchmarked NeRF models.

**Streaming Radiance Fields** [157] (October 2022), is an explicit representation method that specifically targeted NeRF training from video, and improved on standard explicit methods. The authors employed model difference based compression to reduce the memory requirement of the explicit representation. The method also uses a narrow band tuning method and a various training acceleration strategies. This method achieved a training time approximately 90 times faster than Plenoxels [43], with 100-300 times less memory requirement.

### 3.7.2 Others Neural Volume Rendering

IBRNet [108] (February 2021) was published in 2021 as a NeRF adjacent method for view synthesis that is widely used in benchmarks. For a target view, IBRNet selected  $N$  views from the training set whose viewing directions are most similar. A CNN was used to extract features from these images. For a single query point, for each of the  $i$  input view, the known camera matrix was used to project onto the corresponding image to extract color  $c_i$  and feature  $f_i$ . An MLP was then used to refine these features  $f_i$  to be multi-view aware and produce pooling weights  $w_i$ . For density prediction these features were summed using the weights. This is done for each query point, and the results (of all query points along the ray) were concatenated together and fed into a ray Transformer [122] which predicted the density. For color prediction, the  $f_i$ 's and  $w_i$ 's were fed into an MLP alongside relative viewing direction  $\Delta d_i$  to produce a color blending weight. The final color was simply the per-image color  $c_i$  summed using the blending weights. IBRNet in general outperformed baseline NeRF [1], and used 1/6 the number of FLOPs in the experimental setup of the paper.

Compared to NeRF models, **Scene Rendering Transformer (SRT)** [158] (November 2021) took a different approach to volume rendering. They used a CNN to extract feature patches from scene images which were then fed into an Encoder-Decoder Transformer [122], which along with camera ray and viewpoint coordinates  $\{o, d\}$ , which then produced the output color. The entire ray was queried at once, unlike with NeRF models. The SRT is geometry-free, and did not produce the scene's density function, nor relied on geometric inductive biases. The **NeRFormer** [159] (September 2021) is a comparable concurrent model which also uses Transformers as part of the volume rendering process. The paper also introduced the Common Objects in 3D dataset, which could gain popularity in the near future.

## 4 APPLICATIONS

This subsection details select works whose innovations focused on specific applications of NeRF, culminating in an organizational classification tree (Fig. 8). The classification tree also includes certain models previously introduced in Sec. 3 with a strong focus on applications.

Many of the methods in section 3 have obvious applications. The **SLAM-based methods** in section 3.6 have applications in navigation. The generative methods in 3.4

have potential applications in 3D graphics and digital entertainment. Additionally, NeRF models with a strong focus on high-quality geometry reconstruction can be used for photogrammetry. This section introduces methods which focus on specific types of scenes (urban, human faces, avatars, and articulated bodies), 3D reconstruction, or specific image processing tasks.

A work by Adamkiewicz et al. [160] (October 2021) focused on the localization and navigation aspect, and demonstrated a real-life application of a pretrained NeRF, in assisting the navigation of a robot through a church. The authors represented the environment with a pretrained NeRF model, with the robot itself approximated by a finite collection of point for collision checking. Since the NeRF model is pretrained, this method cannot be classified as a pose-estimation model, but instead demonstrated an interesting real-life use of NeRF.

**Dex-NeRF** [161] (October 2021) use NeRF's learned density to help robots grasp objects, specifically focusing on transparent objects, a failure case for depth maps produced by certain rgb-d cameras such as RealSense. The paper also presents three novel datasets focused on transparent objects, one synthetic, and two real-world. Dex-NeRF improves upon baseline NeRF with respect to computed depths of transparent objects by using a fixed empirical threshold for density along rays. Their NeRF model is then used to produce a depth map used by Dex-Net [162] for grasp planning. **Evo-NeRF** [163] (November 2022) improves upon Dex-NeRF by reusing weights in sequential grasping, early termination, and an improved Radiance-Adjusted-Grasp Network capable of grasp planning with unreliable geometry.

In the following subsections, we classify applications of NeRF methods into urban reconstruction, human and articulated body reconstruction, surface reconstruction, and low-level image processing, with the understanding that navigation and generative methods are covered in the previous subsection.

### 4.1 Urban

Given current interests in autonomous driving and urban photogrammetry, there has been a recent surge the development of urban NeRF models, both from street-level and remote-sensing views. The training of a urban NeRF model pose some unique challenges. First, outdoor environments are unbounded; second, the camera poses typically lack variety; third, large-scale scenes are desired. We detail in this section how these models overcome some or all of these challenges.

**Urban Radiance Fields** [70] (November 2021) aimed at applying NeRF based view synthesis and 3D reconstruction for urban environment using sparse multi-view images supplemented by LiDAR data. In addition to the standard photometric loss, they also use a LiDAR based depth loss  $L_{depth}$  and sight loss  $L_{sight}$ , as well as a skybox based segmentation loss  $L_{seg}$ . These are given by

$$L_{depth} = \mathbb{E}[(z - \hat{z})^2], \quad (28)$$

$$L_{sight} = \mathbb{E}\left[\int_{t_1}^{t_2} (w(t) - \delta(z))^2 dt\right]. \quad (29)$$

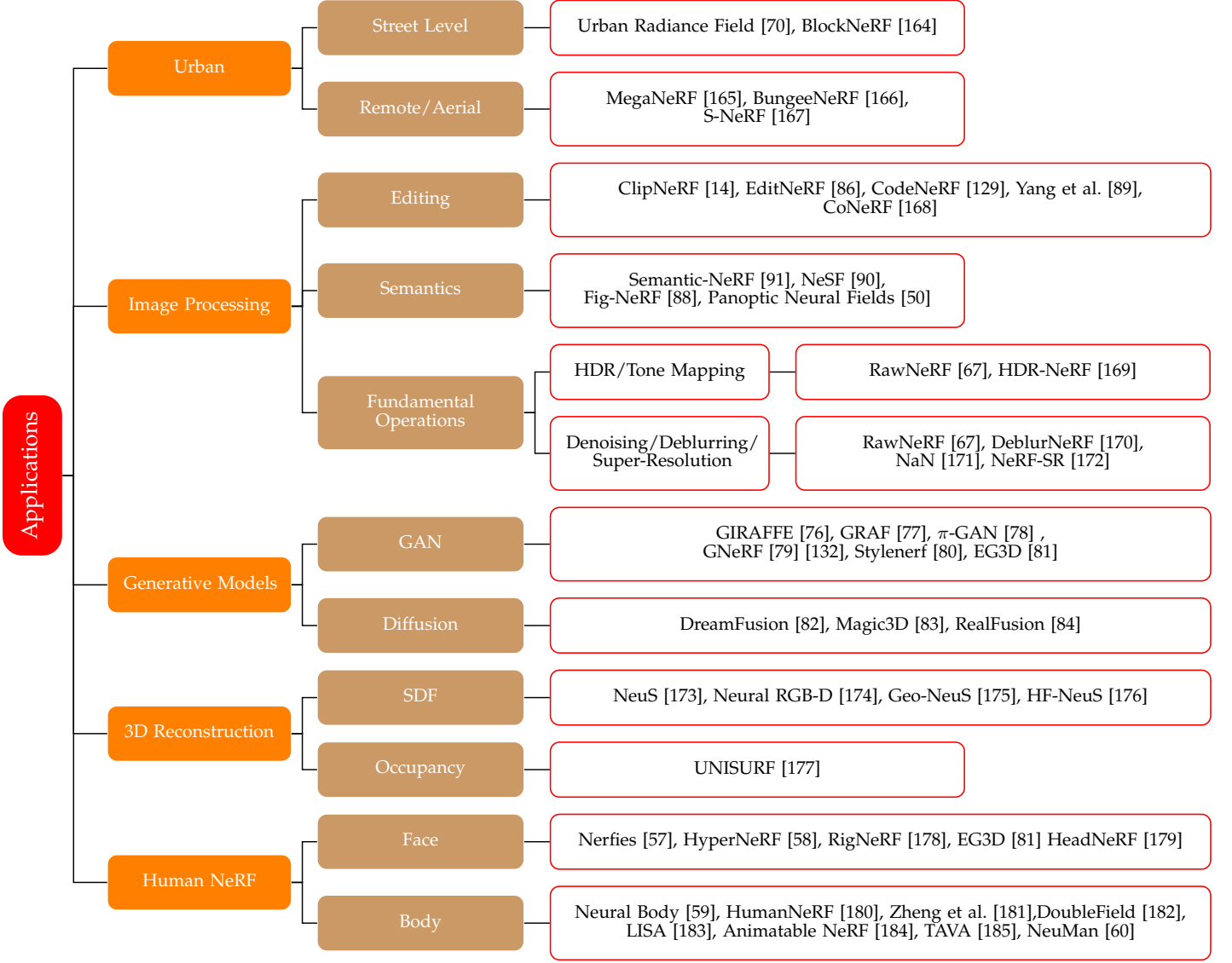


Fig. 8. Application of NeRF models. Papers are selected based on application as well as citation numbers and GitHub star rating.

$$L_{seg} = \mathbb{E}[S_i(\mathbf{r} \int_{t_1}^{t_2} (w(t) - \delta(z))^2 dt)]. \quad (30)$$

$w(t)$  is defined as  $T(t)\sigma(t)$  as defined in eq(3).  $z$  and  $\hat{z}$  are the LiDAR measure depth and estimated depth (6), respectively.  $\delta(z)$  is the Dirac delta function.  $S_i(\mathbf{r}) = 1$  if the ray goes through a sky pixel in the  $i$ th image, where sky pixels are segmented through a pretrained model [186], 0 otherwise. The depth loss forces the estimated depth  $\hat{z}$  to match the LiDAR acquired depth. The sight loss forces the radiance to be concentrated at the surface of the measured depth. The segmentation loss forces point samples along rays through to the sky pixels to have zero density. 3D reconstruction was performed by extracting point clouds from the NeRF model during volumetric rendering. A ray was cast for each pixel in the virtual camera. Then, the estimated depth was used to place the point cloud in the 3D scene. Poisson Surface Reconstruction was used to construct 3D mesh from this generated point cloud. The authors used

Google Street View data on which the Urban Radiance Field outperformed NeRF [1], NeRF-W [85], mip-NeRF [45], and DS-NeRF[23] for both view synthesis and 3D reconstruction.

**Mega-NeRF** [165] (December 2021) performed large scale urban reconstruction from aerial drone images. Mega-NeRF used a NeRF++[87] inverse sphere parameterization to separate foreground from background. However, the authors extended the method by using an ellipsoid which better fit the aerial point of view. They incorporated the per-image appearance embedding code of NeRF-W [85] into their model as well. They partitioned the large urban scenes into cells each one represented by its own NeRF module, and train each module on only the images with potentially relevant pixels. For rendering, the method also cached a coarse rendering of densities and colors into an octree. During rendering for dynamic fly-through, a coarse initial view was quickly produced, and dynamically refined via additional rounds of model sampling. The model outperformed benchmarked baseline such as NeRF++[87],



COLMAP based MVS reconstruction [2], and produced impressive fly-through videos.

**Block-NeRFs** [164] (February 2022) performed city-scale NeRF based reconstruction from 2.8 million street-level images. Such large scale outdoor dataset posed problems such as transient appearance and objects. Each Individual Block-NeRF was built on mip-NeRF [45] by using its IPE and NeRF-W[85] by using its appearance latent code optimization. Moreover, the authors used semantic segmentation to mask out transient objects such as pedestrians and cars during NeRF training. A visibility MLP was trained in parallel, supervised using the transitivity function (3) and the density value generated by the NeRF MLP. These were used to discard low visibility Block-NeRFs. Neighbourhoods were divided into blocks, on which a Block-NeRF was trained. These blocks were assigned with overlap, and images were sampled from overlapping Block-NeRFs and composited using inverse distance weighting after an appearance code matching optimization.

Other methods published in first class conferences such as **S-NeRF** [167] (April 2021), **BungeeNeRF**[166] (December 2021), also deal with urban 3D reconstruction and view synthesis, albeit from remote sensing images.

## 4.2 Human Faces and Avatars, and Articulated Objects

A key application of NeRF models is the reconstruction of human avatars, finding applications in virtual/augmented reality, digital entertainment, and communication. Two families of NeRF models target these applications: those that reconstruct human (or animal) faces and those that reconstruct human/articulated bodies. The reconstruction of human faces requires the NeRF model to be robust under changes of facial expression, which may manifest themselves as topological changes. Models typically parameterize the deformation field as additional MLP(s), potentially conditioned by latent code(s), allowing for controlled deformation from a baseline human face (see subsection 4.2). It is worth noting that many of the GAN-based NeRF models or NeRF models in GAN frameworks (subsection 3.4.1) were trained and tuned on datasets of human faces such as CelebA [136] or FFHQ [139], and can arguably be placed in this section. Human body poses pose a different set of challenges. The NeRF model must be robust under pose changes for articulated bodies, which are often modeled as a deformation field with a template human body model.

Park et al. introduced **Nerfies** [57] (November 2020), a NeRF model build using a deformation field which strongly improved the performance of their model in presence of non-rigid transformations in the scene (e.g., a dynamic scene). By introducing an additional MLP which mapped input observation frame coordinates to deformed canonical coordinates and by adding elastic regularization, background regularization, and coarse-to-fine deformation regularization by adaptive masking the positional encoding, they were able to accurately reconstruct certain non-static scenes which the baseline NeRF completely failed to do. An interesting application the authors found was the creation of multi-view "selfies"<sup>2</sup>. Concurrent to Nerfies

was **NerFace** [187] (December 2020), which also used per-frame learned latent codes, and added facial expression as a 76-dimensional coefficient of a morphable model of constructed from Face2Face [188].

Park et al. introduced **HyperNeRF** [58] (June 2021), which built on Nerfies by extending the canonical space to a higher dimension, and adding an additional slicing MLP which describes how to return to the 3D representation using ambient space coordinates. The canonical coordinate and ambient space coordinate were then used to condition the usual density and color MLPs of baseline NeRF models. HyperNeRF achieved great results in synthesizing views in scenes with topological changes with examples such as a human opening their mouth, or a banana being peeled.

**CoNeRF** [168] (December 2021) was built on HyperNeRF, but allowed for easily controllable photo editing via sliders, whose values are provided to a per-attribute Hypermap deformation field, parameterized by an MLP. This is done via sparse supervised annotation of slider values, and image patch masks, with a  $L_2$  loss term for slider attribute value, and a cross entropy loss for mask supervision. CoNeRF achieved good results, using sliders to adjust facial expressions in their example dataset, which could have broad commercial applications for virtual human avatars. **RigNeRF** also innovated on this topic by using deformation fields MLP guided by a morphable 3D face model, creating fully 3D face portrait with controllable pose and expression.

**Neural Body** (Dec 2020) applied NeRF volume rendering to rendering humans with moving poses from videos. The authors first used the input video to anchor a vertex based deformable human body model (SMPL [189]). Onto each vertex, the authors attached a 16-dimensional latent code  $\mathbf{Z}$ . Human pose parameters  $\mathbf{S}$  (initially estimated from video during training, can be input during inference) were then used to deform the human body model. This model was voxelized in a grid and then processed by a 3D CNN, which extracted 128-dimensional latent code (feature) at each occupied voxel. Any 3D point  $\mathbf{x}$  was first transformed to SMPL coordinate system, then a 128-dimensional latent code/feature  $\psi$  was extract via interpolation. This was passed to the density MLP. The color MLP took in addition the positional encoding of 3D coordinate  $\gamma_x(\mathbf{x})$  and viewing direction  $\gamma_d(\mathbf{d})$  and appearance latent code  $\mathbf{l}_t$  (accounting for per frame difference in the video).

$$\sigma(\mathbf{x}) = M_\sigma(\psi(\mathbf{x}|\mathbf{Z}, \mathbf{S})). \quad (31)$$

$$c(\mathbf{x}) = M_c(\psi(\mathbf{x}|\mathbf{Z}, \mathbf{S}), \gamma_x(\mathbf{x}), \gamma_d(\mathbf{d}), \mathbf{l}_t). \quad (32)$$

Standard NeRF approaches struggled with the moving bodies, whereas the mesh deformation approach of Neural Body was able to interpolate between frames and between poses. State of the art models from top tier conferences in 2021/2022 such as **A-NeRF**[190] (Feb 2021), **Animatable NeRF** [184] (May 2021), **DoubleField** [182] (Jun 2021), **HumanNeRF** [180] (Jan 2022), **Zheng et al.** [181](March 2022), **NeuMan** [60] (March 2022), **TAVA** [185] (June 2022) also innovated on this topic. A popular paradigm for animating articulated body is to use a baseline skeleton, and equip on top of it a MLP-based deformation field.

**PREF** [191] (September 2022) in particular focused on dynamics and motion in image sequences by regularizing esti-

2. Popular self-portraits in social media

mated motion conditioned on latent embedding. Although PREF was trained and tested on image sequences of human avatars, it should be applicable to other domains. Many of the aforementioned papers such as NeuMan and TAVA also focus on animating the subject under novel (human subject) poses and motions.

Unlike the aforementioned models and preprints, LISA (April 2022) specifically targeted the modelling of hands by approximating human hands with a collection of rigid parts. The query points were input into MLPs which were used to predict geometry (via SDFs) and color.

### 4.3 Image Processing

PREF [191] (September 2022) in particular focused on dynamics and motion in image sequences by regularizing estimated motion conditioned on latent embedding. Although PREF was trained and tested on image sequences of human avatars, it should be applicable to other domains. Many of the aforementioned papers such as NeuMan and TAVA also focus on animating the subject under novel (human subject) poses and motions.

Unlike the aforementioned models and preprints, LISA (April 2022) specifically targeted the modelling of hands by approximating human hands with a collection of rigid parts. The query points were input into MLPs which were used to predict geometry (via SDFs) and color.

Mildenhall et al. created RawNeRF [67] (Nov 2021), adapting Mip-NeRF [45], to High Dynamic Range (HDR) image view synthesis and denoising. RawNeRF renders in a linear color space using raw linear images as training data. This allows for varying exposure and tone-mapping curves, essentially applying the post-processing after NeRF rendering instead of directly using post-processed images as training data. It is trained using a relative MSE loss for noisy HDR path tracing from Noise2Noise [192], given by

$$L = \sum_{i=1}^N \left( \frac{\hat{y}_i - y_i}{sg(\hat{y}_i) + \epsilon} \right)^2 \quad (33)$$

where  $sg(\cdot)$  indicates a gradient-stop (treating its argument as a constant with zero gradient). RawNeRF is supervised with variable exposure images, with the NeRF models' "exposure" scaled by the training image's shutter speed, as well as a per-channel learned scaling factor. It achieved impressive results in night-time and low light scene rendering and denoising. RawNeRF is particularly suited for scenes with low lighting. On the topic of NeRF based denoising, NaN [171] (April 10) also explored this emerging research area.

Concurrent to RawNeRF, HDR-NeRF [169] (Nov 2021) from Xin et al. also worked on HDR view synthesis. However, HDR-NeRF approached HDR view synthesis by using Low Dynamic Range training images with variable exposure time as opposed to the raw linear images in RawNeRF. RawNeRF modelled a HDR radiance  $e(\mathbf{r}) \in [0, \infty)$  which replaced the standard  $c(\mathbf{r})$  in (1). This radiance was mapped to a color  $c$  using three MLP camera response functions (one for each color channel)  $f$ . These represent the typical camera manufacturer dependent linear and non-linear post-processing. HDR-NeRF strongly outperformed baseline NeRF and NeRF-W [85] on Low Dynamic Range

(LDR) reconstruction, and achieved high visual assessment scores on HDR reconstruction.

Other methods such as DeblurNeRF [170] (November 2021), NeRF-SR [172] (December 2021), NaN (April 2022) [171]. Focus on fundamental image processing tasks such as denoising, deblurring and super-resolution, allowing for high quality view synthesis from low quality training images.

#### 4.3.1 Semantic NeRF Models

Semantic and instance segmentation are two of the most important and most studied tasks in computer vision. The training of NeRF model with semantic understand or capable of semantic view synthesis is a key area of development for NeRF research.

Semantic-NeRF [91] (March 2021) was a NeRF model capable of synthesizing semantic labels for novel views. This was accomplished using an additional direction-independent MLP (branch) that took position and density MLP features as input and produced a point-wise semantic label  $s$ . The semantic labels were also generated via volume rendering by

$$S(\mathbf{r}) = \sum_i^N T_i \alpha_i s_i. \quad (34)$$

The semantic labels were supervised using categorical cross-entropy loss. The method was able to train with sparse semantic label data (10% labelled), as well as recover semantic labels from pixel-wise noise and region/instance-wise noise. The method also achieved good label super-resolution results and label propagation from sparse point-wise annotation. It can also be used for multi-view semantic fusion, outperforming non-deep learning methods. The previously introduced Fig-NeRF [88] also employed a similar approach.

Panoptic NeRF [193] (Mar 2022) specialized in urban environments, focusing on 3D-to-2D label propagation, a key task in extending urban autonomous driving datasets. The method used two semantic fields, one learned by a semantic head, another, rigid, determined by 3D bounding boxes. According to the authors, the rigid bounding box based semantics forced the model to learn the correct geometry, whereas the learned semantic head improved semantic understanding. Their method was evaluated on the KITTI-360 [55], outperforming previous methods of semantic label transfer.

Panoptic Neural Fields [50] (May 2022) first separated the "stuff" (as named by the authors), considered to be the background static objects, from the "things", considered to be the moving objects in the scene. The "stuff" is represented by a single (two in large scenes, one for foreground, one for background) radiance field MLP which outputs color, density and semantic logits, whereas each dynamic "thing" is represented by its own radiance field inside a dynamic bounding box. The total loss function is the sum of the photometric loss function and per-pixel cross entropy function. The model was trained and tested on KITTI [51] and KITTI 360 [55]. In addition to novel-view synthesis and depth prediction synthesis, the model is also capable of semantic segmentation synthesis, instance segmentation

synthesis, and scene editing via manipulating object-specific MLPs.

**Kobayashi et al.** [194] (May 2022) published a preprint in which they distilled the knowledge of off-the-shelf 2D feature extractor into 3D feature fields which they optimized in conjunction with in-scene radiance fields to produce a NeRF model with semantic understanding which allowed for scene editing. The distillation from CLIP-based feature extractor allow for zero-shot segmentation from an open set of text labels or queries.

**SS-NeRF** [195] (Jun 2022) employed a encoding function two position decoding functions, one direction dependent, one direction independent, all represented by multi-layer perceptrons. The network was trained to produce a variety of scene properties, tested on the Replica dataset [35]: color, semantic labels, surface normal, shading, keypoints, and edges using a combination of losses including MSE for color and normals; MAE for shading, keypoints and edges; and cross entropy for semantic labels. This work showed that scene property synthesis is easily achievable via volume rendering and simple NeRF training without making use of advanced neural architectures.

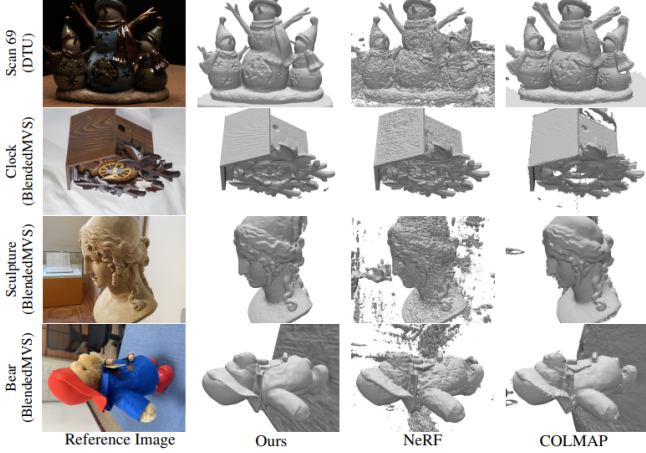


Fig. 9. NeuS [173] surface reconstruction results on the BlendedMVS [117] and DTU dataset [28]. (Figure 5. in [173])

#### 4.4 Surface Reconstruction

The scene geometry of NeRF model is implicit and hidden inside the neural networks. However, for certain applications, more explicit representations, such as 3D mesh are desired. For the baseline NeRF, it is possible to extract a rough geometry by evaluating and thresholding the density MLP. The methods introduced in this subsection used innovative scene representation strategies that change the fundamental behavior of the density MLP.

**UNISURF** [177] (April 2021) reconstructed scene surfaces by replacing the alpha value  $a_i$  at the  $i$ -th sample point used in the discretized volume rendering equation, given by (5), with a discrete occupancy function  $o(\mathbf{x}) = 1$  in occupied space, and  $o(\mathbf{x}) = 0$  in free space. This occupancy function was also computed by an MLP, and essentially replaced the volume density. Surfaces were then retrieved via root-finding along rays. UNISURF outperformed benchmark methods including using density threshold in baseline

NeRF models, as well as IDR [196]. The occupancy MLP can be used to define an explicit surface geometry for the scene. A recent workshop by Tesla [197] showed that autonomous driving module’s 3D understanding is driven by one such NeRF-like occupancy network.

Signed distance functions (SDF) give the signed distance of a 3D point to the surface(s) they define (i.e., negative distance if inside an object, positive distance if outside). They are often used in computer graphics to define surfaces which are the zero set of the function. SDF can be used for surface reconstruction via root-finding, and can be used to define entire scene geometries.

The **Neural Surface (NeuS)** [173] (June 2021) model performed volume rendering like the baseline NeRF model. However it used signed distance functions to define scene geometries. It replaces the density outputting portion of the MLP with an MLP which outputs the signed distance function value. The density  $\rho(t)$  which replaced  $\sigma(t)$  in the volume rendering equation (2) was then constructed as

$$\rho(t) = \max\left(\frac{-\frac{d\Phi}{dt}(f(\mathbf{r}(t)))}{\Phi(f(\mathbf{r}(t)))}, 0\right) \quad (35)$$

where  $\Phi(\cdot)$  is the sigmoid function, and its derivative  $\frac{d\Phi}{dt}$  is the logistic density distribution. The authors have shown that their model outperforms the baseline NeRF model and have provided both theoretical and experimental justifications for their method and its implementation of SDF-based scene density. This method was concurrent to UNISURF and outperformed it on the DTU dataset [28]. Like with UNISURF, performing root finding on the SDF can be used to define an explicit surface geometry for the scene. **HF-NeuS** (June 2022) [176] improved on NeuS by separating low frequency details into a based SDF and high frequency details into a displacement function, greatly increasing reconstruction quality. Concurrently, **Geo-NeuS** (May 2022) [175] introduced a new multi-view constraints in the form of a multi-view geometry constraint for the SDF supervised by sparse point cloud, and a multi-view photometric consistency constraint. **SparseNeuS** (June 2022) [198], also concurrent, improved on NeuS by focusing on sparse-view SDF reconstruction using a geometry encoding volume with learnable image features as a hybrid representation method.

A concurrent work by **Azinovic et al.** [174] (April 2021) also replaced the density MLP with a truncated SDF MLP. They instead computed their pixel color as weighted sum of sampled colors

$$C(\mathbf{r}) = \frac{\sum_{i=1}^N w_i \mathbf{c}_i}{\sum_{i=1}^N w_i} \quad (36)$$

$w_i$  is given by a product of sigmoid function given by

$$w_i = \Phi\left(\frac{D_i}{tr}\right) \cdot \Phi\left(-\frac{D_i}{tr}\right) \quad (37)$$

where  $tr$  is the truncation distance, which cuts off any SDF value too far from individual surfaces. To account for possible multiple ray-surface intersection, subsequent truncation regions weighted to zero, and do not contribute to the pixel color. The authors also use a per-frame appearance latent code from NeRF-W [85] to account for to white-balance and exposure changes. The authors reconstructed the triangular



mesh of the scene by using Marching Cubes [199] on their truncated SDF MLP, and achieved clean reconstruction results on ScanNet[30] and a private synthetic dataset.

## 5 DISCUSSION

In the past two years, NeRF models have made tremendous progress and addressed the disadvantages of the baseline NeRF of Mildenhall et al. [1]. In this section, we summarize advancements which addressed these issues, as well as suggest potential future research directions.

### 5.1 Concerning Speed

#### 5.2 Improving Speed in NeRF Models

The baseline NeRF models are known for their slow training and inference speed. Researchers have developed several methods to speed up NeRF-based models. These methods fall into three main categories:

- Baked models: Baked models cache the results of an already trained NeRF model, but do not improve training time.
- Hybrid scene representation: Hybrid models separate learned scene features from the color and density MLPs and use additional learned voxel/spatial-tree features. While this approach can speed up inference, it requires additional memory.
- Explicit scene representation: Explicit models perform volume rendering directly on learned spatial features such as voxels without use of MLPs. This approach can also speed up inference, but requires additional memory. To further improve speed-based methods, researchers can use advanced volume rendering techniques such as empty space skipping and early ray termination. They can also lower the number of views and ray samples required for training and rendering.

Currently, Instant-NGP [42] shows promise for its use of a multi-resolution hashed positional encoding as additional learned features. This allows the model to represent scenes accurately with tiny and efficient MLPs, leading to extremely fast training and inference. This approach also has applications in image compression and neural SDF scene representation. TensoRF [44] also shows promise by addressing memory consumption, a key issue of hybrid and explicit methods..

In the future, we believe researchers can focus on improving the data structure and design of additional learned scene features to account for the memory trade-off in hybrid and explicit scene representation methods. In addition, improving sparse-view capabilities can also greatly reduce the training time required for NeRF convergence, and should be considered when training speed is of concern.

### 5.3 Concerning Quality

When it comes to quality improvement, recent NeRF research has explored various methods to enhance the appearance and content of generated images. One approach that has proved influential is the use of per-image transient latent codes and appearance codes, as implemented in NeRF-W

[85] and GRAF [77]. These codes allow the NeRF model to control per-image lighting and coloration changes, as well as small variations in scene content.

In terms of NeRF fundamentals, mip-NeRF [45] has been particularly impactful, as its cone tracing IPE offers a novel way to sample and process the scene geometry. Building on mip-NeRF, Ref-NeRF has further improved view-dependent appearance and is now a widely used modern baseline for quality-based NeRF research. For scenes with 360 degree camera coverage, mip-NeRF 360 [99] has been shown to further enhance image quality.

In addition, recent innovations in image processing have also been incorporated into NeRF models to address challenges such as noise and blur. RawNeRF [67] and DeblurNeRF [170] are two examples of such approaches that can be combined with more powerful NeRF models as baseline to build extremely high-quality denoising and deblurring NeRF models.

### 5.4 Concerning Pose Estimation and Sparse View

Given the current state of NeRF research, we believe non-SLAM pose estimation is a solved problem. The SfM using the COLMAP[2] package is used by most NeRF dataset to provide approximate poses, which is sufficient for most NeRF research. BA can also be used to jointly optimize NeRF models and poses during training. NeRF based SLAM is a relatively under-explored area of research. iMAP [92] and Nice-SLAM [93] offer excellent NeRF based SLAM frameworks which could integrate faster and better quality NeRF models. Navigation inside a NeRF environment is especially under explored with [160] opening the door to future applications of NeRF as vision systems for navigation.

Sparse view and few-shot NeRF models typically use 2D/3D feature extraction techniques that rely on pretrained CNNs. Some models also incorporate point cloud data from SfM for additional supervision. We believe many of these models already achieved the goal of few shot (2-10 views). Further small improvements can be achieved by using more advanced feature extraction backbones. We believe a key area of research would be combining sparse views methods and fast methods to achieve real-time NeRF model training and inference deployable to mobile devices. Another interesting recent trend is single-shot NeRF models [84][147][149] by using image/latent diffusion methods and fine tuning via subject image-to-string identifier [200]. We expect this area of few-shot research to be highly productive in conjunction with the text based generative capabilities of diffusion models.

### 5.5 Concerning Applications

We believe that NeRF has immediate applications in novel view synthesis and 3D reconstruction of urban environments and human avatars. To further improve these applications, future research could focus on facilitating the extraction of 3D mesh, point cloud or SDF from density MLPs and integrating faster NeRF models. For urban environments, dividing the environment into separate small scenes and representing each with a small scene-specific

NeRF model is an interesting research direction, particularly at city scale where memory requirements must be considered. Additionally, recent research in human avatar rendering and view synthesis under novel poses and motion is highly relevant to digital entertainment and is currently a popular topic. Combining this research with diffusion-based NeRF models for generating novel human avatars with text/image prompts is a potential future area of research.

Aside from these practical applications, NeRF is also finding applications in fundamental image processing tasks such as denoising, deblurring, upsampling, compression, and image editing. We expect more innovations in these areas in the near future as more computer vision and computer graphics practitioners adopt NeRF models.

### 5.5.1 Generative Models

Generative adversarial networks and, more recently, diffusion models have played a significant role in inspiring and enabling a large body of NeRF models. In particular, the recent text-to-image diffusion-powered NeRF models show the most potential for practical applications. 2D diffusion has recently enabled the development of text-to-3D NeRF models with scene and object editing capabilities, which allow for the generation of 3D mesh and texture from text prompts. This has tremendous potential for application in 3D graphics.

A related research direction for diffusion-based models is few-shot/single-shot NeRF training, which is a highly active research area in the NeRF community. We believe that diffusion-powered NeRF is still in its infancy, and we expect to see significant progress in this area in the near future. This is especially noteworthy given the recent impact of image-based generative stable diffusion models on 2D digital art, as well as the broad impact of recent generative large language models in 2022. We believe that generative 3D models based on NeRF or neural rendering with explicit/hybrid scene representation will have a similar impact on the 3D graphics industry.

### 5.5.2 Semantic Models

Semantic and object segmentation are two of the most important and extensively studied tasks in computer vision. NeRF models with semantic understanding and novel semantic view synthesis represent a key area of NeRF research. One immediate application of semantic view synthesis is 3D to 2D label transfer, an important tool in building certain large-scale autonomous driving datasets [55]. Furthermore, label propagation is possible for certain semantic models [90], enabling cheaper and faster dataset creation. Depending on the implementation, semantic understanding also allows for object-based scene editing [50], [88]. We believe that semantic models will be useful for dataset creation, as semantic labels are expensive both in 2D and 3D. If vision and navigation systems were to be built around NeRF models, combining SLAM-based NeRF models with semantic understanding NeRF models could represent a productive and straightforward extension of current research.

## 6 CONCLUSION

Since the original paper by Mildenhall et al., NeRF models have made tremendous progress in terms of speed, quality, and training view requirements, addressing the weaknesses of the original model. NeRF models have found numerous applications in areas such as urban mapping, photogrammetry, image editing, labelling, processing, and 3D reconstruction and view synthesis of human avatars and urban environments. Both the technical improvements and applications were discussed in detail in this survey. During its completion, we noticed a growing interest in NeRF models, with an increasing number of preprints and publications.

NeRF is an exciting paradigm for novel view synthesis, 3D reconstruction, and neural rendering. By providing this survey, we aim to introduce more computer vision practitioners to this field, provide a helpful reference of existing NeRF models, and motivate future research with our discussions. We are excited to witness future technical innovations and applications of Neural Radiance Fields.

## REFERENCES

- [1] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," in *European conference on computer vision*. Springer, 2020, pp. 405–421.
- [2] J. L. Schonberger and J.-M. Frahm, "Structure-from-motion revisited," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4104–4113.
- [3] M. Levoy and P. Hanrahan, "Light field rendering," in *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, 1996, pp. 31–42.
- [4] S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen, "The lumigraph," in *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, 1996, pp. 43–54.
- [5] B. Mildenhall, P. P. Srinivasan, R. Ortiz-Cayon, N. K. Kalantari, R. Ramamoorthi, R. Ng, and A. Kar, "Local light field fusion: Practical view synthesis with prescriptive sampling guidelines," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 4, pp. 1–14, 2019.
- [6] V. Sitzmann, M. Zollhöfer, and G. Wetzstein, "Scene representation networks: Continuous 3d-structure-aware neural scene representations," *Advances in Neural Information Processing Systems*, vol. 32, 2019.
- [7] S. Lombardi, T. Simon, J. Saragih, G. Schwartz, A. Lehrmann, and Y. Sheikh, "Neural volumes: learning dynamic renderable volumes from images," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 4, pp. 1–14, 2019.
- [8] M. Niemeyer, L. Mescheder, M. Oechsle, and A. Geiger, "Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 3504–3515.
- [9] K. Genova, F. Cole, A. Sud, A. Sarna, and T. Funkhouser, "Local deep implicit functions for 3d shape," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 4857–4866.
- [10] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, "Deepsdf: Learning continuous signed distance functions for shape representation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 165–174.
- [11] F. Dellaert and L. Yen-Chen, "Neural volume rendering: Nerf and beyond," *arXiv preprint arXiv:2101.05204*, 2020.
- [12] Y. Xie, T. Takikawa, S. Saito, O. Litany, S. Yan, N. Khan, F. Tombari, J. Tompkin, V. Sitzmann, and S. Sridhar, "Neural fields in visual computing and beyond," in *Computer Graphics Forum*, vol. 41, no. 2. Wiley Online Library, 2022, pp. 641–676.
- [13] F. Zhan, Y. Yu, R. Wu, J. Zhang, and S. Lu, "Multimodal image synthesis and editing: A survey," *arXiv preprint arXiv:2112.13592*, 2021.



- [14] C. Wang, M. Chai, M. He, D. Chen, and J. Liao, "Clip-nerf: Text-and-image driven manipulation of neural radiance fields," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 3835–3844.
- [15] Y. Guo, K. Chen, S. Liang, Y.-J. Liu, H. Bao, and J. Zhang, "Ad-nerf: Audio driven neural radiance fields for talking head synthesis," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5784–5794.
- [16] A. Jain, B. Mildenhall, J. T. Barron, P. Abbeel, and B. Poole, "Zero-shot text-guided object generation with dream fields," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 867–876.
- [17] K. Jo, G. Shim, S. Jung, S. Yang, and J. Choo, "Cg-nerf: Conditional generative neural radiance fields," *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2023.
- [18] J. Sun, X. Wang, Y. Shi, L. Wang, J. Wang, and Y. Liu, "Ide-3d: Interactive disentangled editing for high-resolution 3d-aware portrait synthesis," *arXiv preprint arXiv:2205.15517*, 2022.
- [19] Y. Chen, Q. Wu, C. Zheng, T.-J. Cham, and J. Cai, "Sem2nerf: Converting single-view semantic masks to neural radiance fields," *European conference on computer vision*, 2022.
- [20] A. Tewari, J. Thies, B. Mildenhall, P. Srinivasan, E. Tretschk, W. Yifan, C. Lassner, V. Sitzmann, R. Martin-Brualla, S. Lombardi *et al.*, "Advances in neural rendering," in *Computer Graphics Forum*, vol. 41, no. 2. Wiley Online Library, 2022, pp. 703–735.
- [21] J. T. Kajiya and B. P. Von Herzen, "Ray tracing volume densities," *ACM SIGGRAPH computer graphics*, vol. 18, no. 3, pp. 165–174, 1984.
- [22] M. Niemeyer, J. T. Barron, B. Mildenhall, M. S. Sajjadi, A. Geiger, and N. Radwan, "Regnerf: Regularizing neural radiance fields for view synthesis from sparse inputs," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 5480–5490.
- [23] K. Deng, A. Liu, J.-Y. Zhu, and D. Ramanan, "Depth-supervised nerf: Fewer views and faster training for free," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 12 882–12 891.
- [24] Y.-C. Guo, D. Kang, L. Bao, Y. He, and S.-H. Zhang, "Nerfren: Neural radiance fields with reflections," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 18 409–18 418.
- [25] D. Xu, Y. Jiang, P. Wang, Z. Fan, H. Shi, and Z. Wang, "Sinnerf: Training neural radiance fields on complex scenes from a single image," 2022.
- [26] M. Tancik, P. Srinivasan, B. Mildenhall, S. Fridovich-Keil, N. Raghavan, U. Singhal, R. Ramamoorthi, J. Barron, and R. Ng, "Fourier features let networks learn high frequency functions in low dimensional domains," *Advances in Neural Information Processing Systems*, vol. 33, pp. 7537–7547, 2020.
- [27] S. Ramasinghe and S. Lucey, "Beyond periodicity: towards a unifying framework for activations in coordinate-mlps," in *Computer Vision—ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXIII*. Springer, 2022, pp. 142–158.
- [28] R. Jensen, A. Dahl, G. Vogiatzis, E. Tola, and H. Aanaes, "Large scale multi-view stereopsis evaluation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 406–413.
- [29] J.-Y. Bouguet, *Camera Calibration Toolbox for Matlab*. CaltechDATA, May 2022. [Online]. Available: <https://data.caltech.edu/records/20164>
- [30] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "ScanNet: Richly-annotated 3d reconstructions of indoor scenes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 5828–5839.
- [31] A. Dai, M. Nießner, M. Zollöfer, S. Izadi, and C. Theobalt, "Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface re-integration," *ACM Transactions on Graphics (TOG)*, 2017.
- [32] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su *et al.*, "Shapenet: An information-rich 3d model repository," *arXiv preprint arXiv:1512.03012*, 2015.
- [33] A. Knapitsch, J. Park, Q.-Y. Zhou, and V. Koltun, "Tanks and temples: Benchmarking large-scale scene reconstruction," *ACM Transactions on Graphics (ToG)*, vol. 36, no. 4, pp. 1–13, 2017.
- [34] A. Chang, A. Dai, T. Funkhouser, M. Halber, M. Niessner, M. Savva, S. Song, A. Zeng, and Y. Zhang, "Matterport3d: Learning from rgb-d data in indoor environments," *arXiv preprint arXiv:1709.06158*, 2017.
- [35] J. Straub, T. Whelan, L. Ma, Y. Chen, E. Wijmans, S. Green, J. J. Engel, R. Mur-Artal, C. Ren, S. Verma *et al.*, "The replica dataset: A digital replica of indoor spaces," *arXiv preprint arXiv:1906.05797*, 2019.
- [36] B. Deng, J. T. Barron, and P. P. Srinivasan, "JaxNeRF: an efficient JAX implementation of NeRF," 2020. [Online]. Available: <https://github.com/google-research/tree/master/jaxnerf>
- [37] L. Liu, J. Gu, K. Z. Lin, T.-S. Chua, and C. Theobalt, "Neural sparse voxel fields," *NeurIPS*, 2020.
- [38] P. Hedman, P. P. Srinivasan, B. Mildenhall, J. T. Barron, and P. Debevec, "Baking neural radiance fields for real-time view synthesis," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5875–5884.
- [39] A. Yu, R. Li, M. Tancik, H. Li, R. Ng, and A. Kanazawa, "PlenOc-trees for real-time rendering of neural radiance fields," in *ICCV*, 2021.
- [40] S. J. Garbin, M. Kowalski, M. Johnson, J. Shotton, and J. Valentin, "Fastnerf: High-fidelity neural rendering at 200fps," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 14 346–14 355.
- [41] C. Reiser, S. Peng, Y. Liao, and A. Geiger, "Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 14 335–14 345.
- [42] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *ACM Trans. Graph.*, vol. 41, no. 4, pp. 102:1–102:15, Jul. 2022. [Online]. Available: <https://doi.org/10.1145/3528223.3530127>
- [43] A. Yu, S. Fridovich-Keil, M. Tancik, Q. Chen, B. Recht, and A. Kanazawa, "Plenoxels: Radiance fields without neural networks," *arXiv preprint arXiv:2112.05131*, 2021.
- [44] A. Chen, Z. Xu, A. Geiger, J. Yu, and H. Su, "Tensorf: Tensorial radiance fields," in *Computer Vision—ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXII*. Springer, 2022, pp. 333–350.
- [45] J. T. Barron, B. Mildenhall, M. Tancik, P. Hedman, R. Martin-Brualla, and P. P. Srinivasan, "Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5855–5864.
- [46] D. Verbin, P. Hedman, B. Mildenhall, T. Zickler, J. T. Barron, and P. P. Srinivasan, "Ref-nerf: Structured view-dependent appearance for neural radiance fields," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 5491–5500.
- [47] A. Chen, Z. Xu, F. Zhao, X. Zhang, F. Xiang, J. Yu, and H. Su, "Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 14 124–14 133.
- [48] A. Jain, M. Tancik, and P. Abbeel, "Putting nerf on a diet: Semantically consistent few-shot view synthesis," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5885–5894.
- [49] J. Li, Z. Feng, Q. She, H. Ding, C. Wang, and G. H. Lee, "Mine: Towards continuous depth mpi with nerf for novel view synthesis," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 12 578–12 588.
- [50] A. Kundu, K. Genova, X. Yin, A. Fathi, C. Pantofaru, L. J. Guibas, A. Tagliasacchi, F. Dellaert, and T. Funkhouser, "Panoptic neural fields: A semantic object-aware neural scene representation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 12 871–12 881.
- [51] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.
- [52] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.
- [53] J. Fritsch, T. Kuehnl, and A. Geiger, "A new performance measure and evaluation benchmark for road detection algorithms," in *International Conference on Intelligent Transportation Systems (ITSC)*, 2013.

- [54] M. Menze and A. Geiger, "Object scene flow for autonomous vehicles," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.
- [55] Y. Liao, J. Xie, and A. Geiger, "Kitti-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.
- [56] P. Sun, H. Kretschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine *et al.*, "Scalability in perception for autonomous driving: Waymo open dataset," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 2446–2454.
- [57] K. Park, U. Sinha, J. T. Barron, S. Bouaziz, D. B. Goldman, S. M. Seitz, and R. Martin-Brualla, "Nerfies: Deformable neural radiance fields," *ICCV*, 2021.
- [58] K. Park, U. Sinha, P. Hedman, J. T. Barron, S. Bouaziz, D. B. Goldman, R. Martin-Brualla, and S. M. Seitz, "Hypernerf: A higher-dimensional representation for topologically varying neural radiance fields," *ACM Trans. Graph.*, vol. 40, no. 6, dec 2021.
- [59] S. Peng, Y. Zhang, Y. Xu, Q. Wang, Q. Shuai, H. Bao, and X. Zhou, "Neural body: Implicit neural representations with structured latent codes for novel view synthesis of dynamic humans," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 9054–9063.
- [60] W. Jiang, K. M. Yi, G. Samei, O. Tuzel, and A. Ranjan, "Neuman: Neural human radiance field from a single video," in *Computer Vision—ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXII*. Springer, 2022, pp. 402–418.
- [61] H. Joo, H. Liu, L. Tan, L. Gui, B. Nabbe, I. Matthews, T. Kanade, S. Nobuhara, and Y. Sheikh, "Panoptic studio: A massively multi-view system for social motion capture," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 3334–3342.
- [62] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.
- [63] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 586–595.
- [64] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and 0.5 mb model size," *arXiv preprint arXiv:1602.07360*, 2016.
- [65] K. Simonyan and A. Zisserman, "Very deep convolutional neural networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.
- [66] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.
- [67] B. Mildenhall, P. Hedman, R. Martin-Brualla, P. P. Srinivasan, and J. T. Barron, "Nerf in the dark: High dynamic range view synthesis from noisy raw images," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 16190–16199.
- [68] B. Roessle, J. T. Barron, B. Mildenhall, P. P. Srinivasan, and M. Nießner, "Dense depth priors for neural radiance fields from sparse input views," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 12892–12901.
- [69] Y. Wei, S. Liu, Y. Rao, W. Zhao, J. Lu, and J. Zhou, "Nerfing-mvs: Guided optimization of neural radiance fields for indoor multi-view stereo," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5610–5619.
- [70] K. Rematas, A. Liu, P. P. Srinivasan, J. T. Barron, A. Tagliasacchi, T. Funkhouser, and V. Ferrari, "Urban radiance fields," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 12932–12942.
- [71] Q. Xu, Z. Xu, J. Philip, S. Bi, Z. Shu, K. Sunkavalli, and U. Neumann, "Point-nerf: Point-based neural radiance fields," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 5438–5448.
- [72] D. B. Lindell, J. N. Martel, and G. Wetzstein, "Autoint: Automatic integration for fast neural volume rendering," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 14556–14565.
- [73] C. Sun, M. Sun, and H.-T. Chen, "Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 5459–5469.
- [74] A. Yu, V. Ye, M. Tancik, and A. Kanazawa, "pixelNeRF: Neural radiance fields from one or few images," in *CVPR*, 2021.
- [75] Y. Liu, S. Peng, L. Liu, Q. Wang, P. Wang, C. Theobalt, X. Zhou, and W. Wang, "Neural rays for occlusion-aware image-based rendering," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 7824–7833.
- [76] M. Niemeyer and A. Geiger, "Giraffe: Representing scenes as compositional generative neural feature fields," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 11453–11464.
- [77] K. Schwarz, Y. Liao, M. Niemeyer, and A. Geiger, "Graf: Generative radiance fields for 3d-aware image synthesis," *Advances in Neural Information Processing Systems*, vol. 33, pp. 20154–20166, 2020.
- [78] E. R. Chan, M. Monteiro, P. Kellnhofer, J. Wu, and G. Wetzstein, "pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 5799–5809.
- [79] Q. Meng, A. Chen, H. Luo, M. Wu, H. Su, L. Xu, X. He, and J. Yu, "Gnerf: Gan-based neural radiance field without posed camera," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 6351–6361.
- [80] J. Gu, L. Liu, P. Wang, and C. Theobalt, "Stylenet: A style-based 3d aware generator for high-resolution image synthesis," in *Tenth International Conference on Learning Representations*, 2022, pp. 1–25.
- [81] E. R. Chan, C. Z. Lin, M. A. Chan, K. Nagano, B. Pan, S. De Mello, O. Gallo, L. J. Guibas, J. Tremblay, S. Khamis *et al.*, "Efficient geometry-aware 3d generative adversarial networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 16123–16133.
- [82] B. Poole, A. Jain, J. T. Barron, and B. Mildenhall, "Dreamfusion: Text-to-3d using 2d diffusion," *arXiv preprint arXiv:2209.14988*, 2022.
- [83] C.-H. Lin, J. Gao, L. Tang, T. Takikawa, X. Zeng, X. Huang, K. Kreis, S. Fidler, M.-Y. Liu, and T.-Y. Lin, "Magic3d: High-resolution text-to-3d content creation," *arXiv preprint arXiv:2211.10440*, 2022.
- [84] L. Melas-Kyriazi, C. Rupprecht, I. Laina, and A. Vedaldi, "Real-fusion: 360° reconstruction of any object from a single image," *arXiv e-prints*, pp. arXiv–2302, 2023.
- [85] R. Martin-Brualla, N. Radwan, M. S. M. Sajjadi, J. T. Barron, A. Dosovitskiy, and D. Duckworth, "NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections," in *CVPR*, 2021.
- [86] S. Liu, X. Zhang, Z. Zhang, R. Zhang, J.-Y. Zhu, and B. Russell, "Editing conditional radiance fields," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5773–5783.
- [87] K. Zhang, G. Riegler, N. Snavely, and V. Koltun, "Nerf++: Analyzing and improving neural radiance fields," *arXiv:2010.07492*, 2020.
- [88] C. Xie, K. Park, R. Martin-Brualla, and M. Brown, "Fig-nerf: Figure-ground neural radiance fields for 3d object category modelling," in *2021 International Conference on 3D Vision (3DV)*. IEEE, 2021, pp. 962–971.
- [89] B. Yang, Y. Zhang, Y. Xu, Y. Li, H. Zhou, H. Bao, G. Zhang, and Z. Cui, "Learning object-compositional neural radiance field for editable scene rendering," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 13779–13788.
- [90] S. Vora\*, N. Radwan\*, K. Greff, H. Meyer, K. Genova, M. S. M. Sajjadi, E. Pot, A. Tagliasacchi, and D. Duckworth, "Neural semantic fields for generalizable semantic segmentation of 3d scenes," *Transactions on Machine Learning Research*, 2022, <https://openreview.net/forum?id=ggPhsYCsm9>.
- [91] S. Zhi, T. Laidlow, S. Leutenegger, and A. J. Davison, "In-place scene labelling and understanding with implicit scene representation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 15838–15847.
- [92] E. Sucar, S. Liu, J. Ortiz, and A. J. Davison, "imap: Implicit mapping and positioning in real-time," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 6229–6238.
- [93] Z. Zhu, S. Peng, V. Larsson, W. Xu, H. Bao, Z. Cui, M. R. Oswald, and M. Pollefeys, "Nice-slam: Neural implicit scalable encoding for slam," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 12786–12796.

- [94] A. Rosinol, J. J. Leonard, and L. Carlone, "Nerf-slam: Real-time dense monocular slam with neural radiance fields," *arXiv preprint arXiv:2210.13641*, 2022.
- [95] Z. Wang, S. Wu, W. Xie, M. Chen, and V. A. Prisacariu, "NeRF--: Neural radiance fields without known camera parameters," *arXiv preprint arXiv:2102.07064*, 2021.
- [96] C.-H. Lin, W.-C. Ma, A. Torralba, and S. Lucey, "Barf: Bundle-adjusting neural radiance fields," in *IEEE International Conference on Computer Vision (ICCV)*, 2021.
- [97] Y. Jeong, S. Ahn, C. Choy, A. Anandkumar, M. Cho, and J. Park, "Self-calibrating neural radiance fields," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5846–5854.
- [98] S.-F. Chng, S. Ramasinghe, J. Sherrah, and S. Lucey, "Gaussian activated neural radiance fields for high fidelity reconstruction and pose estimation," in *Computer Vision—ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXIII*. Springer, 2022, pp. 264–280.
- [99] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman, "Mip-nerf 360: Unbounded anti-aliased neural radiance fields," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 5470–5479.
- [100] P. Hedman, P. P. Srinivasan, B. Mildenhall, J. T. Barron, and P. Debevec, "Baking neural radiance fields for real-time view synthesis," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5875–5884.
- [101] J. Chibane, A. Bansal, V. Lazova, and G. Pons-Moll, "Stereo radiance fields (srf): Learning view synthesis for sparse views of novel scenes," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 7911–7920.
- [102] J. Zhang, Y. Zhang, H. Fu, X. Zhou, B. Cai, J. Huang, R. Jia, B. Zhao, and X. Tang, "Ray priors through reprojection: Improving neural radiance fields for novel view extrapolation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 18 376–18 386.
- [103] S. Choi, Q.-Y. Zhou, S. Miller, and V. Koltun, "A large dataset of object scans," *arXiv preprint arXiv:1602.02481*, 2016.
- [104] X. Cheng, P. Wang, and R. Yang, "Learning depth with convolutional spatial propagation network," *IEEE transactions on pattern analysis and machine intelligence*, vol. 42, no. 10, pp. 2361–2379, 2019.
- [105] Z. Li, T. Dekel, F. Cole, R. Tucker, N. Snavely, C. Liu, and W. T. Freeman, "Learning the depths of moving people by watching frozen people," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 4521–4530.
- [106] Y. Yao, Z. Luo, S. Li, T. Fang, and L. Quan, "Mvsnet: Depth inference for unstructured multi-view stereo," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 767–783.
- [107] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 652–660.
- [108] Q. Wang, Z. Wang, K. Genova, P. P. Srinivasan, H. Zhou, J. T. Barron, R. Martin-Brualla, N. Snavely, and T. Funkhouser, "Ibrnet: Learning multi-view image-based rendering," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 4690–4699.
- [109] E. Insafutdinov, D. Campbell, J. F. Henriques, and A. Vedaldi, "Snes: Learning probably symmetric neural surfaces from incomplete data," in *Computer Vision—ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXII*. Springer, 2022, pp. 367–383.
- [110] W. Yang, G. Chen, C. Chen, Z. Chen, and K.-Y. K. Wong, "S<sup>3</sup>-nerf: Neural reflectance field from shading and shadow under a single viewpoint," in *Advances in Neural Information Processing Systems*, 2022.
- [111] J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, and Q. Zhang, "JAX: composable transformations of Python+NumPy programs," 2018. [Online]. Available: <http://github.com/google/jax>
- [112] S. G. Parker, J. Bigler, A. Dietrich, H. Friedrich, J. Hoberock, D. Luebke, D. McAllister, M. McGuire, K. Morley, A. Robison *et al.*, "Optix: a general purpose ray tracing engine," *Acm transactions on graphics (tog)*, vol. 29, no. 4, pp. 1–13, 2010.
- [113] L. Wang, J. Zhang, X. Liu, F. Zhao, Y. Zhang, Y. Zhang, M. Wu, J. Yu, and L. Xu, "Fourier plenotrees for dynamic radiance field rendering in real-time," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 13 524–13 534.
- [114] Z. Chen, T. Funkhouser, P. Hedman, and A. Tagliasacchi, "Mobilenerf: Exploiting the polygon rasterization pipeline for efficient neural field rendering on mobile architectures," *arXiv preprint arXiv:2208.00277*, 2022.
- [115] T. Hu, S. Liu, Y. Chen, T. Shen, and J. Jia, "Efficientnerf efficient neural radiance fields," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 12 902–12 911.
- [116] L. Wu, J. Y. Lee, A. Bhattad, Y.-X. Wang, and D. Forsyth, "Diver: Real-time and accurate neural radiance fields with deterministic integration for volume rendering," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 16 200–16 209.
- [117] Y. Yao, Z. Luo, S. Li, J. Zhang, Y. Ren, L. Zhou, T. Fang, and L. Quan, "Blendedmvs: A large-scale dataset for generalized multi-view stereo networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 1790–1799.
- [118] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
- [119] A. Trevisi and B. Yang, "Grf: Learning a general radiance field for 3d representation and rendering," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 15 182–15 192.
- [120] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision," in *International Conference on Machine Learning*. PMLR, 2021, pp. 8748–8763.
- [121] M. M. Johari, Y. Lepoittevin, and F. Fleuret, "Geonrf: Generalizing nerf with geometry priors," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 18 365–18 375.
- [122] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.
- [123] D. Rebain, M. Matthews, K. M. Yi, D. Lagun, and A. Tagliasacchi, "Lolnerf: Learn from one look," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 1558–1567.
- [124] P. Bojanowski, A. Joulin, D. Lopez-Paz, and A. Szlam, "Optimizing the latent space of generative networks," *arXiv preprint arXiv:1707.05776*, 2017.
- [125] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.
- [126] X. Zhang, S. Bi, K. Sunkavalli, H. Su, and Z. Xu, "Nerfusion: Fusing radiance fields for large-scale scene reconstruction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 5449–5458.
- [127] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.
- [128] N. Müller, A. Simonelli, L. Porzi, S. R. Bulò, M. Nießner, and P. Kotschieder, "Atrorf: Learning 3d object radiance fields from single view observations," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 3971–3980.
- [129] W. Jang and L. Agapito, "Codenerf: Disentangled neural radiance fields for object categories," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 12 949–12 958.
- [130] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 621–11 631.
- [131] D. Chen, Y. Liu, L. Huang, B. Wang, and P. Pan, "Geoaug: Data augmentation for few-shot nerf with geometry constraints," in *Computer Vision—ECCV 2022: 17th European Conference, Tel Aviv,*

- Israel, October 23–27, 2022, *Proceedings, Part XVII*. Springer, 2022, pp. 322–337.
- [132] A. R. Kosiorek, H. Strathmann, D. Zoran, P. Moreno, R. Schneider, S. Mokrá, and D. J. Rezende, “Nerf-vae: A geometry aware 3d scene generative model,” in *International Conference on Machine Learning*. PMLR, 2021, pp. 5742–5752.
- [133] Y. Kim, S. Wiseman, A. Miller, D. Sontag, and A. Rush, “Semi-amortized variational autoencoders,” in *International Conference on Machine Learning*. PMLR, 2018, pp. 2678–2687.
- [134] J. Marino, Y. Yue, and S. Mandt, “Iterative amortized inference,” in *International Conference on Machine Learning*. PMLR, 2018, pp. 3403–3412.
- [135] V. Sitzmann, J. Martel, A. Bergman, D. Lindell, and G. Wetzstein, “Implicit neural representations with periodic activation functions,” *Advances in Neural Information Processing Systems*, vol. 33, pp. 7462–7473, 2020.
- [136] Y. Guo, L. Zhang, Y. Hu, X. He, and J. Gao, “Ms-celeb-1m: A dataset and benchmark for large-scale face recognition,” in *European conference on computer vision*. Springer, 2016, pp. 87–102.
- [137] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, “Carla: An open urban driving simulator,” in *Conference on robot learning*. PMLR, 2017, pp. 1–16.
- [138] W. Zhang, J. Sun, and X. Tang, “Cat head detection-how to effectively exploit shape and texture features,” in *European conference on computer vision*. Springer, 2008, pp. 802–816.
- [139] T. Karras, S. Laine, and T. Aila, “A style-based generator architecture for generative adversarial networks,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 4401–4410.
- [140] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, “Analyzing and improving the image quality of stylegan,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 8110–8119.
- [141] S. Cai, A. Obukhov, D. Dai, and L. Van Gool, “Pix2nerf: Unsupervised conditional p-gan for single image to neural radiance fields translation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 3981–3990.
- [142] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli, “Deep unsupervised learning using nonequilibrium thermodynamics,” in *International Conference on Machine Learning*. PMLR, 2015, pp. 2256–2265.
- [143] C. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. Denton, S. K. S. Ghasemipour, B. K. Ayan, S. S. Mahdavi, R. G. Lopes *et al.*, “Photorealistic text-to-image diffusion models with deep language understanding,” *arXiv preprint arXiv:2205.11487*, 2022.
- [144] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, “High-resolution image synthesis with latent diffusion models,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 10 684–10 695.
- [145] G. Metzer, E. Richardson, O. Patashnik, R. Giryes, and D. Cohen-Or, “Latent-nerf for shape-guided generation of 3d shapes and textures,” *arXiv preprint arXiv:2211.07600*, 2022.
- [146] Y. Balaji, S. Nah, X. Huang, A. Vahdat, J. Song, K. Kreis, M. Aittala, T. Aila, S. Laine, B. Catanzaro *et al.*, “ediffi: Text-to-image diffusion models with an ensemble of expert denoisers,” *arXiv preprint arXiv:2211.01324*, 2022.
- [147] D. Xu, Y. Jiang, P. Wang, Z. Fan, Y. Wang, and Z. Wang, “Neural-lift-360: Lifting an in-the-wild 2d photo to a 3d object with 360° views,” *arXiv e-prints*, pp. arXiv–2211, 2022.
- [148] C. Deng, C. Jiang, C. R. Qi, X. Yan, Y. Zhou, L. Guibas, D. Anguelov *et al.*, “Nerdi: Single-view nerf synthesis with language-guided diffusion as general image priors,” *arXiv preprint arXiv:2212.03267*, 2022.
- [149] J. Gu, A. Trevithick, K.-E. Lin, J. Susskind, C. Theobalt, L. Liu, and R. Ramamoorthi, “Nerfdiff: Single-image view synthesis with nerf-guided distillation from 3d-aware diffusion,” *arXiv preprint arXiv:2302.10109*, 2023.
- [150] J. Wynn and D. Turmukhambetov, “Diffusionerf: Regularizing neural radiance fields with denoising diffusion models,” *arXiv preprint arXiv:2302.12231*, 2023.
- [151] Y. Jin, D. Mishkin, A. Mishchuk, J. Matas, P. Fua, K. M. Yi, and E. Trulls, “Image matching across wide baselines: From paper to practice,” *International Journal of Computer Vision*, 2020.
- [152] K. Yücer, A. Sorkine-Hornung, O. Wang, and O. Sorkine-Hornung, “Efficient 3d object segmentation from densely sampled light fields with applications to 3d reconstruction,” *ACM Transactions on Graphics (TOG)*, vol. 35, no. 3, pp. 1–15, 2016.
- [153] R. Martin-Brualla, R. Pandey, S. Bouaziz, M. Brown, and D. B. Goldman, “Gelato: Generative latent textured objects,” in *European Conference on Computer Vision*. Springer, 2020, pp. 242–258.
- [154] A. Ahmadyan, L. Zhang, A. Ablavatski, J. Wei, and M. Grundmann, “Objectron: A large scale dataset of object-centric videos in the wild with pose annotations,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 7822–7831.
- [155] L. Yen-Chen, P. Florence, J. T. Barron, A. Rodriguez, P. Isola, and T.-Y. Lin, “inrf: Inverting neural radiance fields for pose estimation,” in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 1323–1330.
- [156] K. Jun-Seong, K. Yu-Ji, M. Ye-Bin, and T.-H. Oh, “Hdr-ploxels: Self-calibrating high dynamic range radiance fields,” in *Computer Vision—ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXII*. Springer, 2022, pp. 384–401.
- [157] L. Li, Z. Shen, L. Shen, P. Tan *et al.*, “Streaming radiance fields for 3d video synthesis,” in *Advances in Neural Information Processing Systems*, 2022.
- [158] M. S. Sajjadi, H. Meyer, E. Pot, U. Bergmann, K. Greff, N. Radwan, S. Vora, M. Lučić, D. Duckworth, A. Dosovitskiy *et al.*, “Scene representation transformer: Geometry-free novel view synthesis through set-latent scene representations,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 6229–6238.
- [159] J. Reizenstein, R. Shapovalov, P. Henzler, L. Sbordone, P. Labatut, and D. Novotny, “Common objects in 3d: Large-scale learning and evaluation of real-life 3d category reconstruction,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 10901–10911.
- [160] M. Adamkiewicz, T. Chen, A. Caccavale, R. Gardner, P. Culbertson, J. Bohg, and M. Schwager, “Vision-only robot navigation in a neural radiance world,” *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4606–4613, 2022.
- [161] J. Ichnowski, Y. Avigal, J. Kerr, and K. Goldberg, “Dex-nerf: Using a neural radiance field to grasp transparent objects,” in *Conference on Robot Learning*. PMLR, 2022, pp. 526–536.
- [162] J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. A. Ojea, and K. Goldberg, “Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics,” *arXiv preprint arXiv:1703.09312*, 2017.
- [163] J. Kerr, L. Fu, H. Huang, Y. Avigal, M. Tancik, J. Ichnowski, A. Kanazawa, and K. Goldberg, “Evo-nerf: Evolving nerf for sequential robot grasping of transparent objects,” in *Conference on Robot Learning*. PMLR, 2022.
- [164] M. Tancik, V. Casser, X. Yan, S. Pradhan, B. Mildenhall, P. P. Srinivasan, J. T. Barron, and H. Kretschmar, “Block-nerf: Scalable large scene neural view synthesis,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 8248–8258.
- [165] H. Turki, D. Ramanan, and M. Satyanarayanan, “Mega-nerf: Scalable construction of large-scale nerfs for virtual fly-throughs,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 12 922–12 931.
- [166] Y. Xiangli, L. Xu, X. Pan, N. Zhao, A. Rao, C. Theobalt, B. Dai, and D. Lin, “Bungeenerf: Progressive neural radiance field for extreme multi-scale scene rendering,” in *The European Conference on Computer Vision (ECCV)*, 2022.
- [167] D. Derksen and D. Izzo, “Shadow neural radiance fields for multi-view satellite photogrammetry,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 1152–1161.
- [168] K. Kania, K. M. Yi, M. Kowalski, T. Trzciński, and A. Tagliasacchi, “Conerf: Controllable neural radiance fields,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 18 623–18 632.
- [169] X. Huang, Q. Zhang, Y. Feng, H. Li, X. Wang, and Q. Wang, “Hdr-nerf: High dynamic range neural radiance fields,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 18 398–18 408.
- [170] L. Ma, X. Li, J. Liao, Q. Zhang, X. Wang, J. Wang, and P. V. Sander, “Deblur-nerf: Neural radiance fields from blurry images,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 12 861–12 870.

- [171] N. Pearl, T. Treibitz, and S. Korman, "Nan: Noise-aware nerfs for burst-denoising," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 12 672–12 681.
- [172] C. Wang, X. Wu, Y.-C. Guo, S.-H. Zhang, Y.-W. Tai, and S.-M. Hu, "Nerf-sr: High quality neural radiance fields using super-sampling," in *Proceedings of the 30th ACM International Conference on Multimedia*, 2022, pp. 6445–6454.
- [173] P. Wang, L. Liu, Y. Liu, C. Theobalt, T. Komura, and W. Wang, "Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction," *Advances in Neural Information Processing Systems*, vol. 34, pp. 27 171–27 183, 2021.
- [174] D. Azinović, R. Martin-Brualla, D. B. Goldman, M. Nießner, and J. Thies, "Neural rgb-d surface reconstruction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 6290–6301.
- [175] Q. Fu, Q. Xu, Y.-S. Ong, and W. Tao, "Geo-neus: geometry-consistent neural implicit surfaces learning for multi-view reconstruction," in *Advances in Neural Information Processing Systems*, 2022.
- [176] Y. Wang, I. Skorokhodov, and P. Wonka, "Hf-neus: Improved surface reconstruction using high-frequency details," in *Advances in Neural Information Processing Systems*, 2022.
- [177] M. Oechsle, S. Peng, and A. Geiger, "Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5589–5599.
- [178] S. Athar, Z. Xu, K. Sunkavalli, E. Shechtman, and Z. Shu, "Rign-erf: Fully controllable neural 3d portraits," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 20 364–20 373.
- [179] Y. Hong, B. Peng, H. Xiao, L. Liu, and J. Zhang, "Headnerf: A real-time nerf-based parametric head model," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 20 374–20 384.
- [180] F. Zhao, W. Yang, J. Zhang, P. Lin, Y. Zhang, J. Yu, and L. Xu, "Humanerf: Efficiently generated human radiance field from sparse inputs," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 7743–7753.
- [181] Z. Zheng, H. Huang, T. Yu, H. Zhang, Y. Guo, and Y. Liu, "Structured local radiance fields for human avatar modeling," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 15 893–15 903.
- [182] R. Shao, H. Zhang, H. Zhang, M. Chen, Y.-P. Cao, T. Yu, and Y. Liu, "Doublefield: Bridging the neural surface and radiance fields for high-fidelity human reconstruction and rendering," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022, pp. 15 872–15 882.
- [183] E. Corona, T. Hodan, M. Vo, F. Moreno-Noguer, C. Sweeney, R. Newcombe, and L. Ma, "Lisa: Learning implicit shape and appearance of hands," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 20 533–20 543.
- [184] S. Peng, J. Dong, Q. Wang, S. Zhang, Q. Shuai, X. Zhou, and H. Bao, "Animatable neural radiance fields for modeling dynamic human bodies," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021, pp. 14 314–14 323.
- [185] R. Li, J. Tanke, M. Vo, M. Zollhöfer, J. Gall, A. Kanazawa, and C. Lassner, "Tava: Template-free animatable volumetric actors," in *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXII*. Springer, 2022, pp. 419–436.
- [186] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 801–818.
- [187] G. Gafni, J. Thies, M. Zollhofer, and M. Nießner, "Dynamic neural radiance fields for monocular 4d facial avatar reconstruction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 8649–8658.
- [188] J. Thies, M. Zollhofer, M. Stamminger, C. Theobalt, and M. Nießner, "Face2face: Real-time face capture and reenactment of rgb videos," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2387–2395.
- [189] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black, "SMPL: A skinned multi-person linear model," *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, vol. 34, no. 6, pp. 248:1–248:16, Oct. 2015.
- [190] S.-Y. Su, F. Yu, M. Zollhöfer, and H. Rhodin, "A-nerf: Articulated neural radiance fields for learning human shape, appearance, and pose," *Advances in Neural Information Processing Systems*, vol. 34, pp. 12 278–12 291, 2021.
- [191] L. Song, X. Gong, B. Planche, M. Zheng, D. Doermann, J. Yuan, T. Chen, and Z. Wu, "Pref: Predictability regularized neural motion fields," in *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXII*. Springer, 2022, pp. 664–681.
- [192] J. Lehtinen, J. Munkberg, J. Hasselgren, S. Laine, T. Karras, M. Aittala, and T. Aila, "Noise2noise: Learning image restoration without clean data," *arXiv preprint arXiv:1803.04189*, 2018.
- [193] X. Fu, S. Zhang, T. Chen, Y. Lu, L. Zhu, X. Zhou, A. Geiger, and Y. Liao, "Panoptic nerf: 3d-to-2d label transfer for panoptic urban scene segmentation," *arXiv preprint arXiv:2203.15224*, 2022.
- [194] S. Kobayashi, E. Matsumoto, and V. Sitzmann, "Decomposing nerf for editing via feature field distillation," *arXiv preprint arXiv:2205.15585*, 2022.
- [195] M. Zhang, S. Zheng, Z. Bao, M. Hebert, and Y.-X. Wang, "Beyond rgb: Scene-property synthesis with neural radiance fields," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2023, pp. 795–805.
- [196] L. Yariv, Y. Kasten, D. Moran, M. Galun, M. Atzmon, B. Ronen, and Y. Lipman, "Multiview neural surface reconstruction by disentangling geometry and appearance," *Advances in Neural Information Processing Systems*, vol. 33, pp. 2492–2502, 2020.
- [197] A. Elluswamy. Tesla, workshop on autonomous driving. CVPR 2022. [Online]. Available: <https://www.youtube.com/watch?v=jPCV4GKX9Dw>
- [198] X. Long, C. Lin, P. Wang, T. Komura, and W. Wang, "Sparseneus: Fast generalizable neural surface reconstruction from sparse views," in *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXII*. Springer, 2022, pp. 210–227.
- [199] W. E. Lorenson and H. E. Cline, "Marching cubes: A high resolution 3d surface construction algorithm," *ACM siggraph computer graphics*, vol. 21, no. 4, pp. 163–169, 1987.
- [200] N. Ruiz, Y. Li, V. Jampani, Y. Pritch, M. Rubinstein, and K. Aberman, "Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation," *arXiv preprint arXiv:2208.12242*, 2022.