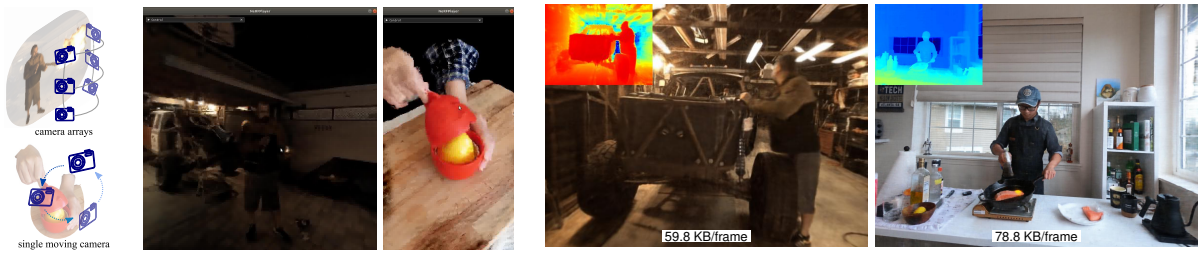
NeRFPlayer: A Streamable Dynamic Scene Representation with Decomposed Neural Radiance Fields

Liangchen Song, Anpei Chen, Zhong Li, Zhang Chen, Lele Chen, Junsong Yuan, Yi Xu, and Andreas Geiger



(a) Inputs (b) Interactive rendering (c) Low bitrate streaming

Fig. 1: (a) Our framework takes as input the RGB images captured from a camera array or a single moving camera. (b) After offline optimization, our framework can render a novel view and perform temporal interpolation interactively. (c) Our framework is highly configurable. Adopting TensoRF-CP [9] voxel representation in our framework results in low bitrate streaming of high-quality rendering.

Abstract—Visually exploring in a real-world 4D spatiotemporal space freely in VR has been a long-term quest. The task is especially appealing when only a few or even single RGB cameras are used for capturing the dynamic scene. To this end, we present an efficient framework capable of fast reconstruction, compact modeling, and streamable rendering. First, we propose to decompose the 4D spatiotemporal space according to temporal characteristics. Points in the 4D space are associated with probabilities of belonging to three categories: static, deforming, and new areas. Each area is represented and regularized by a separate neural field. Second, we propose a hybrid representations based feature streaming scheme for efficiently modeling the neural fields. Our approach, coined NeRFPlayer, is evaluated on dynamic scenes captured by single hand-held cameras and multi-camera arrays, achieving comparable or superior rendering performance in terms of quality and speed comparable to recent state-of-the-art methods, achieving reconstruction in 10 seconds per frame and interactive rendering. Project website: https://bit.ly/nerfplayer.

Index Terms—Neural rendering, free-viewpoint video, immersive video, NeRF

1 Introduction

Representing scenes as Neural Radiance Fields (NeRF) has brought a series of breakthroughs in immersive 3D experiences [45, 77]. High-fidelity real-time rendering of real-world scenes now can be obtained after a few seconds of training [46, 58]. The rendering system only requires a few real-world RGB images [43], but can well model scenes as small as a cell [37] and as large as a city [65].

Despite NeRF's success in static scenes, extending it to handle dynamic scenes remains challenging. Introducing an extra time dimension t to NeRF's 5D representation (3D location x, y, z and 2D viewing direction θ, ϕ) is non-trivial for the following two reasons. First, the supervisory signal for a spatiotemporal point (x, y, z, t) is sparser than a static point (x, y, z). Multi-view images of static scenes are easy to access as we can move the camera around, but an extra view in dynamic scenes requires an extra recording camera, leading to sparse input views. Second, the appearance and geometry frequency of the scene are different along the spatial axis and temporal axis. The content usually changes a lot when moving from one location to another location, but the background scene is unlikely to completely change from one timestamp to another. An inappropriate frequency modeling for the time t dimension results in poor temporal interpolation performance.

A lot of progress has been made in addressing the aforementioned two challenges. Existing solutions include adopting motion models for matching the points (*e.g.*, [15, 35, 51, 52, 55, 66]) and leveraging data-driven priors like depth and optical flow (*e.g.*, [13, 32, 69, 76]). Different from existing works, we are motivated by the observation that in dynamic scenes different spatial areas have different temporal characteristics. We assume that there are three kinds of temporal patterns in a dynamic scene (Figs. 2 and 3): static, deforming, and new areas. We thus propose to decompose the dynamic scene into these categories, which is achieved by a decomposition field that predicts the point-wise probabilities of being static, deforming, and new. The decomposition field is self-supervised and regularized by a manually assigned global parsimony regularization (*e.g.*, suppressing the global probabilities of being new).

The proposed decomposition can address both of the aforementioned challenges. First, different temporal regularizations are introduced for each decomposed area, thus alleviating the ambiguity in reconstruction from sparse observations. For instance, the static area decomposition simplifies the dynamic modeling to a static scene modeling problem. The deforming areas enforce the foreground object to be consistent in the dynamic scene. Second, the scene is split into different areas according to their temporal characteristics, thus resulting in consistent

- *Liangchen Song and Junsong Yuan are with SUNY Buffalo. Work done while Liangchen was an intern at OPPO US Research Center, InnoPeak Technology. E-mail: lsong8@buffalo.edu*
- *Anpei Chen is with ETH Zürich and University of Tübingen.*
- *Zhong Li, Zhang Chen, Lele Chen and Yi Xu are with OPPO US Research Center, InnoPeak Technology.*
- *Andreas Geiger is with University of Tübingen.*
- *Corresponding author: Zhong Li. E-mail: zhonglee323@gmail.com*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

frequency in the time dimension in each of the areas.

In response to the discrepancy between spatial and temporal frequency, we further decouple spatial and temporal dimensions based on the recent developed hybrid representations [9, 46, 58, 62]. Hybrid representations maintain a grid of (x, y, z) feature volumes for fast rendering. Instead of designing a grid of (x, y, z, t) feature volumes, we treat the channels of (x, y, z) feature volumes as temporally dependent. To support streamable dynamic scene representation, we propose a sliding window scheme on the feature channels to introduce t into the representation (Fig. 4). Sliding window not only supports streaming of the feature volumes, but also implicitly encourages the representation to be compact by leveraging the overlapped channels in adjacent frames. For validation, we conduct experiments on datasets captured under both single-camera and multi-camera settings. To sum up, our contributions are as follows:

- We propose to decompose the dynamic scene according to their temporal characteristics. The decomposition is achieved by a decomposition field that takes as input each (x, y, z, t) point and outputs probabilities belonging to three categories: static, deforming, and new.
- We design a self-supervised scheme for optimizing the decomposition field and regularizing the decomposition field with a global parsimony loss.
- We design a sliding window scheme on recently developed hybrid representations for efficiently modeling spatiotemporal fields.
- We present extensive experiments and interactive rendering demos on both single-camera and multi-camera datasets. Our ablation studies validate the implied regularization behind the proposed three temporal patterns.

2 Related Work

2.1 Neural Fields

Neural fields are neural networks that take in the coordinates and output the properties of that point [77]. 3D representations based on neural fields have made tremendous advancements in recent years. The pioneering work Occupancy Networks [42] represents the geometry of 3D objects with a continuous decision boundary modeled by a neural network. Occupancy Networks are further improved to model dynamic objects [48]. Concurrently, DeepSDF [50] represents the geometry with signed distance function with a network. Chibane *et al*. [10] predicts the unsigned distance field for 3D shapes from point clouds. NeRF [45], a milestone work, proposes to represent the scene with a 5D function modeled by MLP. NeRF significantly improves the performance of novel view synthesis (*i.e*., image-based rendering). The scene representation in NeRF inspired a number of works focusing on 3D modeling, such as relighting [4, 5, 61], 3D content generation [8, 21, 27] and AR/VR systems [12].

Hybrid Representation Scenes are implicitly represented by MLPs in vanilla NeRF and forwarding with the MLPs is time-consuming. Some methods like DONeRF [47] accelerate the sampling step [14, 28, 33, 34, 54] and HyperReel [1] has adapted this idea in dynamic scenes. Another set of insightful methods [18, 20, 36, 57, 73, 74, 80] are designed by adopting explicit data structures to efficiently query from the fields. Further, hybrid representations are developed by leveraging both explicit and implicit representations to improve the differentiability of the framework. DVGO [62] uses two feature voxels to represent occupancy and appearance. The feature vectors queried from the voxels are decoded by small MLPs. Plenoxels [58] prune empty spaces and save the sphere harmonic coefficients. Instant-NGP [46] proposes a hash encoding of the saved feature grids and solves hashing collision by multi-scale encoding and small MLP decoding. TensoRF [9] leverages tensor decomposition to reduce the model size of the voxels. Hybrid representations are further leveraged for efficient dynamic scene modeling. Recent concurrent works [15, 16, 35, 70] propose to model canonical spaces with voxels and motion with deformation fields. Li *et al*. [30] propose to stream the difference of voxels in dynamic scenes. Different from the above methods, our method decomposes the scene into different areas and models them separately. A straightforward InstantNGP based dynamic representation is adding extra input dimension of time, but such a baseline requires the full representation of a dynamic sequence to be completely loaded into the GPU memory before rendering. For TensoRF based dynamic modeling, D-TensoRF [23] uses a 5D tensor to represent a 4D spatiotemporal grid. Our method can be widely applicable, as long as the representation adopts feature vectors for modeling points in the space.

Scene Decomposition Neural fields have been adopted for decomposing scenes. Yang *et al*. [78] and Zhang *et al*. [82] decompose the scene by objects for editing. DeRF [56] spatially decomposes the scene and uses small networks for each area for efficiency. Kobayashi *et al*. [25] and Tschernezki *et al*. [67] semantically decompose the scene with pre-trained models. Ost *et al*. [49] decompose scenes into semantic scene graphs. Objects are decomposed by motion in NeuralDiff [68] and STaR [81]. More recently, a decomposition between static and dynamic areas is studied in D^{2}NeRF [75] and Sharma *et al*. [60]. Our decomposition is different from existing works, since we decompose areas according to the temporal changing patterns.

2.2 4D Modeling of Dynamic Scenes

Free viewpoint rendering from video captures has been widely studied over the decades. The idea of viewing an event from multiple perspectives dates back to Multiple Perspective Interactive Video [22], in which 3D environments are generated with dynamic motion models. Virtualized Reality [24] design 3D dome and recovered 3D structure based on multi-camera stereo methods. Inspired by image-based rendering [19, 29], some video-based rendering methods are developed [7, 59, 79], which requires dense capturing of the scene. Zitnick *et al*. [84] propose a layered depth image representation for the high-quality video-based rendering of dynamic scenes. More recently, a milestone work developed by Collet *et al*. [11] utilizes tracked textured meshes for free-viewpoint video streaming. With RGB, infrared (IR), and silhouette information as the input, their system can output accurate geometric, detailed texture, and efficient streaming. Another impressive system developed by Broxton *et al*. [6] proposes multi-sphere image based Layered Meshes. The capturing setting is a low-cost hemispherical array with 46 synchronized cameras, and Layered Meshes are validated to be efficient and can well-handle non-Lambertian surfaces with view-dependent or semi-transparent effects. Bansal *et al*. [2] use convolutional neural nets to compose static and dynamic parts of the event and then adopt U-Net to render images from intermediate results composited from depth-based re-projected images. Neural Volumes (NV) [38] leverages differentiable volume rendering for optimizing a 3D volume representation, which can be transformed from 2D input RGB images using an encoder-decoder network. NV is further strengthened in [39] with volumetric primitives. X-Fields [3] consider input images from different view, time or illumination conditions in structured captures. DyNeRF [31] assign observation frames with a set of compact latent codes and then use time-conditioned neural radiance fields to represent dynamic scenes. Fourier PlenOctrees [71] extend the real-time rendering framework PlenOctrees [80] to dynamic scenes. DeVRF [35] proposes a voxel-based representation that first reconstructs a canonical object from multi-view dense supervisions and then reconstructs deformation from few-view observations.

Another thread of research aims at modeling dynamic scenes without the requirements of multiple synchronized cameras. Multi-view information is collected by moving the camera around in the dynamic space. The setting of single-camera input is much more challenging than the multi-camera setting mentioned above. Data-driven solutions like video depth estimation [26, 40] are developed. Based on the priors and motivated by the success of NeRF, motions are modeled by neural fields. Some methods first define a canonical space that is modeled by a NeRF, then align the following frames from the canonical space. Representative methods include D-NeRF [55], Nerfies [51] and NR-NeRF [66]. The trajectory of points is modeled by a neural field in DCT-NeRF [69]. Directly modeling the 4D field by introducing an extra time dimension into the original radiance field is adopted in NSFF [32], VideoNeRF [76], and NeRFlow [13]. HyperNeRF [52]

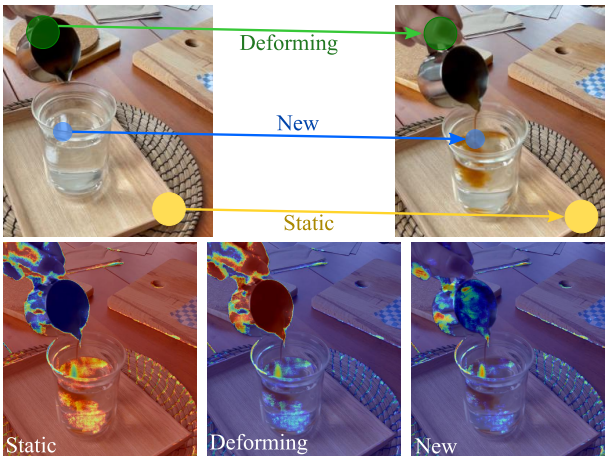


Fig. 2: First row: We categorize the areas in a dynamic scene into three groups: deforming, new and static areas. Second row: Visualization of the self-supervised decomposition obtained from our framework. Red and blue areas indicate estimated high and low probabilities of a category.

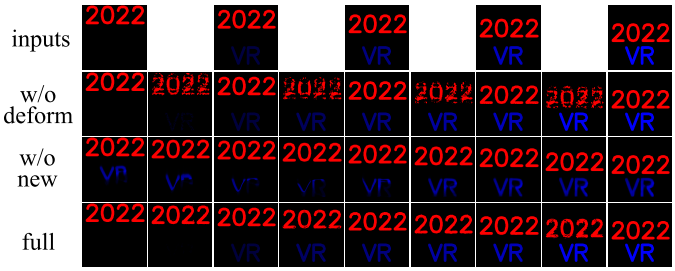


Fig. 3: A toy example of 2D dynamic sequence interpolation. The first row shows the 2D input sequence with missing frames. Without modeling deformation $d(\cdot)$, the second row fails to interpolate the rigid motion of '2022'. Without modeling newness $n(\cdot)$, the third row fails to interpolate the gradually appearing effect. Full decomposition handles both phenomena well.

points out the issue of motion inconsistency in topologically varying scenes and proposes a hyperspace representation, which is inspired by the level-set methods, for optimizing motion in a more smooth solution space. Gao *et al*. [17] demonstrate the discrepancy between the casual monocular video and the above existing monocular testing videos.

The above methods are able to generate impressive results under various settings. However, rendering with both single- and multi-camera inputs can be further studied, such as the effectiveness of motion modeling with multi-camera inputs. Moreover, a tradeoff still exists among model size, training and rendering speed, and rendering quality. Our method studies both single- and multi-camera inputs and focuses on efficient and high-quality free-viewpoint video rendering.

3 Preliminaries

Our method leverages the rendering scheme proposed by NeRF [45] and hybrid representation for static scenes [9, 36, 46, 58, 62–64, 74, 83]. We first briefly review the rendering framework in NeRF, then we introduce the recently developed hybrid representation for efficient neural fields.

For each point $\boldsymbol{p} = (x, y, z)$ in NeRF, we denote its volume density as $\sigma(\boldsymbol{p})$ and its color as $\boldsymbol{c}(\boldsymbol{p}, \boldsymbol{d})$, where $\boldsymbol{d} = (\theta, \phi)$ is the viewing direction. The pixel color $\boldsymbol{C}$ of a camera ray $\boldsymbol{r}$ is computed by accumulating a set of samples on the ray with volume rendering. Let the optical origin and direction of the camera be $\boldsymbol{o}$ and $\boldsymbol{d}$, then a set of points are sampled by

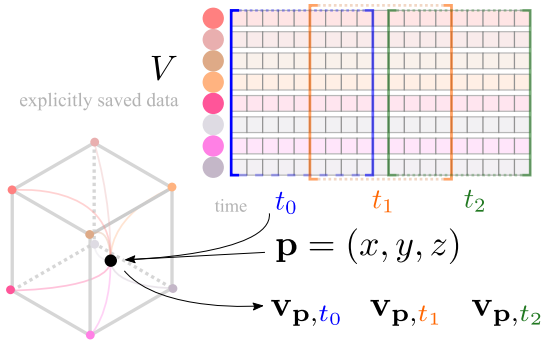


Fig. 4: The proposed streamable hybrid representation. A time-dependent sliding window is adopted for streaming the feature channels.

$\boldsymbol{p}_i = \boldsymbol{o} + i\boldsymbol{d}$ and the expected color $\boldsymbol{C}(\boldsymbol{r})$ is computed by

$$\boldsymbol{C}(\boldsymbol{r}) = \int_{i_n}^{i_f} e^{-\int_{i_n}^{i} \sigma(\boldsymbol{p}_j)\, dj}\, \sigma(\boldsymbol{p}_i)\, \boldsymbol{c}(\boldsymbol{p}_i, \boldsymbol{d})\, di, \tag{1}$$

where i_n, i_f are near and far bounds. Numerical approximation by summing up a set of sample points on the ray is used for computing the integration in Eq. (1). In vanilla NeRF, the radiance field is implicitly represented by an MLP that takes in the point $\boldsymbol{p}$ as input and outputs its density and color. The MLP is then trained with a reconstruction loss between the reconstructed color and ground-truth color $\boldsymbol{C}_{\text{gt}}(\boldsymbol{r})$, *i.e.*,

$$L_{\text{rec}} = \sum_{\boldsymbol{r} \in \mathscr{R}} \| \boldsymbol{C}(\boldsymbol{r}) - \boldsymbol{C}_{\text{gt}}(\boldsymbol{r}) \|_2^2, \tag{2}$$

where $\mathscr{R}$ is a batch of ray samples.

The implicit representation in NeRF is highly compact but computationally expensive, resulting in slow training and rendering speed. Hybrid representations, in which both explicit and implicit representations can be adopted, are developed for efficiently reconstructing and rendering with a radiance field. Though these methods have their unique standouts, all these hybrid representations follow a common framework. First, we have some explicitly stored features V, which can be in the form of a voxel grid [36, 58, 62, 64, 74, 83], a hash table [46] or a set of basis vectors/matrices [9]. For any point $\boldsymbol{p}$ in the 3D space, a feature vector $\boldsymbol{v}_{\boldsymbol{p}} = V(\boldsymbol{p})$ can be efficiently obtained with cheap operations (*e.g.*, tri-linear interpolation for a voxel). Next, a decoder D is adopted to get properties like the density σ and color $\boldsymbol{c}$ of the point from $\boldsymbol{v}_{\boldsymbol{p}}$. The decoder D can be an MLP [9, 46, 62] or spherical harmonics [58].

4 Our Method

Our method is built on the assumption that different areas in a dynamic scene can have different temporal changing patterns. Modeling different areas with different temporal regularizations not only helps keep temporal consistency, but also saves computation. For example, some objects in the background may have a static geometry in the dynamic sequence, which allows us to reduce the capacity and complexity of their representation. We begin our method with a decomposed spatiotemporal representation, which aims to first categorize and then model different dynamic areas using different representations based on their categories.

4.1 Decomposed Spatiotemporal Representation

As illustrated in Fig. 2, we assume three kinds of areas in a dynamic scene and model these areas with separate fields:

- **Static** areas have a constant geometry and location in the dynamic scene, such as the table. Besides, we assume the appearance of the static areas will not change frequently over time, *i.e.*, is temporally low-frequency. This is based on the observation that the appearance change is mainly caused by lighting conditions and the albedo is time-invariant. Hence, a stationary field $s(\cdot)$ is used for representing static points.

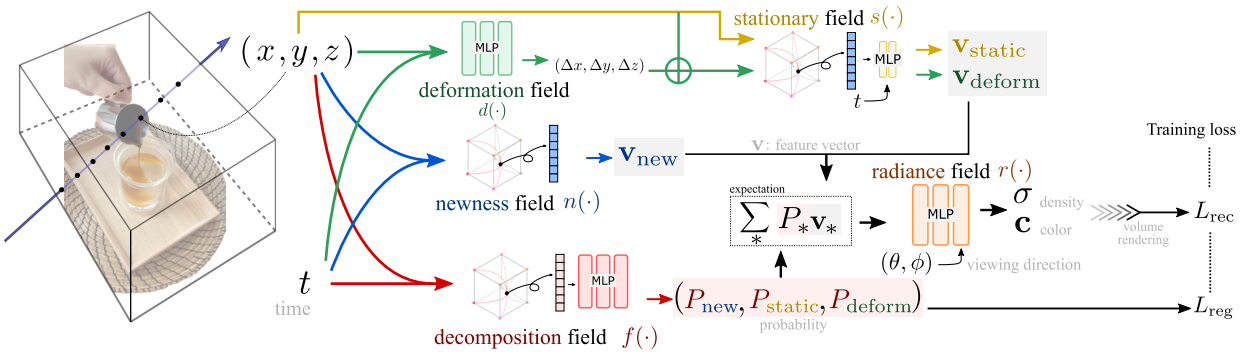


Fig. 5: An overview of our framework. The newness field and decomposition field are implemented with the channel streaming technique proposed in Fig. 4. A small MLP is adopted in the decomposition field for predicting the probabilities. The stationary field consists of a static feature volume for modeling time-invariant areas and a tiny MLP with time t input for modeling low-frequency time-varying appearance. The deformation field and radiance field are two small MLPs.

- **Deforming** areas model objects with deforming surfaces, such as the hand and the cup in Fig. 2. Deforming areas may comprise rigid or non-rigid motion, but they are always presented in the sequence of interest. Deforming points are represented by a deformation field $d(\cdot) : (\boldsymbol{p}, t) \mapsto (\Delta \boldsymbol{p})$. Then the deformed point $\boldsymbol{p} + \Delta \boldsymbol{p}$ is sent as the query point into a predefined canonical space (*e.g.*, the static field s).
- **New** areas model new content emerging at some point in the sequence, such as the new fluid after pouring espresso into water. A newness feature field $n(\cdot)$ with inputs $(\boldsymbol{p}, t)$ is adopted for representing new areas.

To decompose the scene, we design a decomposition field $f(\cdot) : (\boldsymbol{p}, t) \mapsto (P_{\text{static}}, P_{\text{deform}}, P_{\text{new}})$, where $P_{\text{static}}, P_{\text{deform}}, P_{\text{new}}$ denotes the probability of being static, deforming and new. Next, we consider the output of the fields mentioned above (s, n) to be feature vectors rather than properties like the density of the point and denote the output feature vector as $\boldsymbol{v}_{\text{static}}, \boldsymbol{v}_{\text{deform}}, \boldsymbol{v}_{\text{new}}$, respectively. Finally, given a query point $\boldsymbol{p}$, we first collect the outputs from the above fields and then compute the expected feature vector $\boldsymbol{v}$ of this point by $\boldsymbol{v} = \sum_* P_* \boldsymbol{v}_*$, where $* \in \{\text{static}, \text{deform}, \text{new}\}$. Then $\boldsymbol{v}$ is sent to a lightweight view-conditioned network for density and color prediction.

In Fig. 3, we demonstrate our approach using a simple 2D toy example. The task studied in the figure is a temporal interpolation from the given 2D images. The fields mentioned above take (x, y) locations as the input. The string '2022' undergoes rigid motion, while the string 'VR' gradually appears. The different interpolation performance demonstrates the necessity of modeling dynamic scenes with both deforming and new fields. Note that we do not manually annotate the probability when performing decomposition. Instead, the decomposition field f is only supervised by the reconstruction loss and generic parsimony priors, which penalize objects being modeled as new. More details about training will be introduced in Sec. 4.4.

Hybrid representations, which enable fast training and real-time rendering, are adopted for implementing the above neural fields. However, most of existing static scene targeted hybrid representations implement the mapping $\boldsymbol{p} \mapsto (\sigma, \boldsymbol{c})$. Adapting to inputs with an extra dimension time t (*i.e.*, dynamic scenes) is not straightforward, since modeling 4D inputs with the explicit representation V may significantly increase the model size. A streamable hybrid representation for efficient spatiotemporal mapping is introduced in the next section.

4.2 Streamable Hybrid Representation

We observe that the explicit representation V commonly consists of array entries with a predefined feature dimension. For example, each entry in the hash table in InstantNGP [46] and each basis vector/matrix in TensoRF [9] both have a fixed feature dimension. Thus, we propose to stream the *feature channels* so that V can be a mapping from a spatiotemporal point $(\boldsymbol{p}, t)$ to the fixed-length feature vector $\boldsymbol{v}_{\boldsymbol{p},t}$.

We propose to select feature channels with a sliding window along with the time dimension t, as demonstrated in Fig. 4. Assume that for each frame the feature vector $\boldsymbol{v}_{\boldsymbol{p},t}$ is of dimension F and k channels are newly needed for a new frame, then for a T frame sequence the array entry v in V is of dimension $F + k(T-1)$. For a single frame t, the channels $[kt, kt+F]$ in V will be used for computing $\boldsymbol{v}_{\boldsymbol{p},t}$, such as trilinear interpolation in InstantNGP or tensor multiplication in TensoRF.

To ensure $\boldsymbol{v}_{\boldsymbol{p},t}$ smoothly translates along with t, a rearrangement of feature channels is conducted to match the shared channels. For example, let $t = 0, k = 2$, and $F = 4$, then channels $[0, 1, 2, 3]$ of V are used for $\boldsymbol{v}_{\boldsymbol{p},0}$. Next, we use channels $[4, 5, 2, 3]$ for $t = 1$ and channels $[4, 5, 6, 7]$ for $t = 2$. The principle behind the rearrangement is that shared feature channels are always aligned to be with the same index in the vector. Otherwise, a smooth translation between frames is not guaranteed.

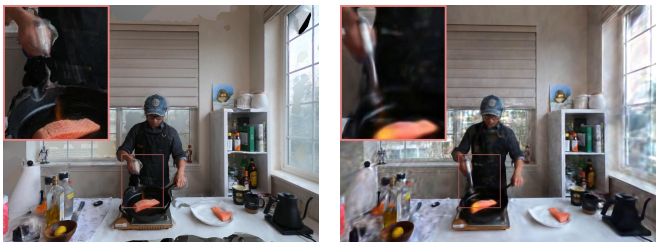
The streaming channels readily enable us to temporally interpolate a frame t between two observed frames t_s and t_{s+1} by linearly interpolating the feature vectors: $\boldsymbol{v}_{\boldsymbol{p},t} = \frac{t - t_s}{t_{s+1} - t_s} \boldsymbol{v}_{\boldsymbol{p},t_{s+1}} + \frac{t_{s+1} - t}{t_{s+1} - t_s} \boldsymbol{v}_{\boldsymbol{p},t_s}$. Note that our proposed method can be applied to any hybrid representation V that contains entries of feature vectors. The implementation of V employed will be referred to as *backbone* in the following text. The sliding window scheme brings two benefits: First, overlapping feature channels are forced to be shared in adjacent frames, thus reducing the model size; Second, after rendering one frame, only new feature channels need to be loaded when moving to the subsequent frames, thus being streaming friendly.

4.3 Overall Framework.

Now we introduce the details of implementing the decomposed spatiotemporal representation (Sec. 4.1) with the streamable hybrid representation (Sec. 4.2). An illustration is presented in Fig. 5. The decomposition field f consists of explicitly cached features (denoted by V_f) and a small MLP decoder D_f. The deformation field d is an MLP since the deformation is sparse and of low-frequency, where a small MLP is enough. The stationary field consists of explicitly cached features (denoted by V_s) and a tiny MLP decoder. Time t and feature obtained from V_s will be the input to the tiny MLP. The reason for using a tiny MLP is for modeling time-dependent appearance changes caused by time-varying illumination, which is assumed to be of low-frequency. The newness field n is explicitly saved features V_n. Note that in the above explicit representations, both V_f and V_n take in a 4D input $(\boldsymbol{p}, t)$, hence streaming channels are used here. The final expected feature vector $\boldsymbol{v}$ is then decoded by a radiance field r. Viewing direction (θ, ϕ) is also sent to r as in NeRF.

4.4 Optimization

Training. Our training process follows the practice of NeRF. A batch of camera rays $\mathscr{R}$ is first randomly sampled from the observed



Multi-View Stereo NV [38]

LLFF [44] DyNeRF [31]

Ours-InstantNGP Ours-TensoRF-VM

Fig. 6: Qualitative comparisons on the Plenoptic Video (multi-camera setting) dataset.

data and then points on those rays, denoted by $\mathscr{R}_{\boldsymbol{p}}$, are sampled for training.

In practical reconstruction and rendering tasks, a precise supervisory signal to the decomposition field is inaccessible. Instead, we supervise the output probabilities with a global parsimony regularization. Therefore, besides the reconstruction loss defined in Eq. (2), a regularization loss L_{reg} is introduced in our method. We use the average probability of all points in the batch for this loss, denoted as $\overline{P_*} = \frac{1}{|\mathscr{R}_{\boldsymbol{p}}|}\sum_{\boldsymbol{p}\in\mathscr{R}_{\boldsymbol{p}}} P_*(\boldsymbol{p})$, where $|\mathscr{R}_{\boldsymbol{p}}|$ is the number of points. In our implementation, assuming the existence of static background, we propose to minimize the probability of not being a static point, thus the regularization loss is chosen as

$$L_{\text{reg}} = \alpha\overline{P_{\text{deform}}} + \overline{P_{\text{new}}}, \tag{3}$$

where α is a tunable parameter for weighting the ratio of being deforming and new. Minimizing the probability of being new points in Eq. (3) relies on the assumption that most of the points in the dynamic scene are either static or deforming. Overall, our training loss is

$$L = L_{\text{rec}} + \lambda L_{\text{reg}}, \tag{4}$$

where λ is a balancing hyperparameter.

Rendering. When rendering an image with a given camera pose, we first forward the sampled points using the decomposition field. After knowing the probabilities, we can skip the forwarding process of some fields for efficiency. With a predefined threshold τ, if $P_* < \tau$ then we directly set $\boldsymbol{v}_* = \mathbf{0}$ and skip the field. We set τ as 0.001 in our implementation.

5 Experiments

We first quantitatively and qualitatively compare our method with prior works, then extensive ablation studies are presented to validate our proposed components. We urge the reader to watch our video to better appreciate the efficiency and rendering quality of our system.

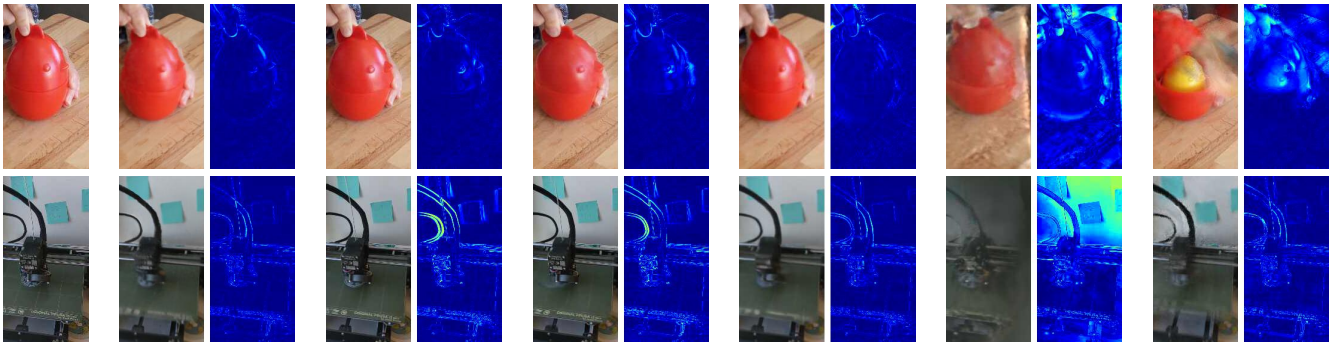
DyNeRF [31] Ours-TensoRF-VM

Fig. 7: Comparisons of rendering performance on fast moving objects.

Datasets. Our method requires only RGB observations of the dynamic scene for reconstruction. Unlike most existing methods, our framework does not require special capturing settings or prior knowledge, and detailed comparisons of the framework's requirements against competitive methods are attached in the supplementary. Two multi-camera datasets and one single-camera dataset are used:

- **Immersive Video** [6] includes synchronized videos from 46 4K fisheye cameras. For the raw video data provided by the authors, each camera has different imaging parameters like exposure and white balance. We select 7 dynamic scenes with relatively similar imaging parameters. We downsample the images to 1280×960 in our experiments. The camera with ID 0 (the central camera) is used for validation, and the other cameras are used for training.
- **Plenoptic Video** [31] is captured with 21 cameras at a resolution of 2704×2028. Different from Immersive Video, which mostly focuses on outdoor scenes, Plenoptic Video consists of indoor activities in various lighting conditions. We downsample images to 1352×1014 in our experiments. We follow the training and validation camera split provided by [31]. Six scenes are publically available.
- **HyperNeRF** [51, 52] provides only one view for each timestamp in a dynamic scene. The dataset is challenging due to the single-camera setting. We adopt the same training and validation settings as in [52]: images of 960×540 are used for quantitative evaluation and images of 1920×1080 are used for qualitative comparisons. There are two capturing settings in HyperNeRF: "*vrig*" captures the scene with stereo cameras and training with one camera and validating with the other; "*interp*" is a monocular video from a moving camera capturing dynamic scenes.

Implementation details. Our framework, as demonstrated by Fig. 5, is implemented with PyTorch [53]. As highlighted in Sec. 4.3, our framework is general and any hybrid representation adopting explicit features can be used. We implement our framework with two backbones: InstantNGP [46] and TensoRF [9]. In both of the implementations, the deformation network is a 4-layer MLP with a width of 256. The stationary field s uses a 2-layer MLP with a width of 64. The radiance field r is a 4-layer MLP with a width of 64 and has the same structure as the decoder in the backbone. For InstantNGP based model, the number of levels is 8 and the number of feature dimensions per entry is 4. TensoRF based model follows the same setting as in their experiments on the real forward-facing datasets (*i.e.*, LLFF [44]). For both of the two backbones, we set the number of channels for streaming k to be 1, and loss hyperparameters $\lambda = 0.1$, $\alpha = 0.01$. We follow the default optimization schedule and settings as in the static-scene targeted backbone methods. We validate the effectiveness of deformation decomposition (Figs. 9 and 10) and further found that our framework remains effective and becomes more efficient if we skip deformation modeling on multi-camera datasets with dense temporal sampling. Due to the limitation of model sizes, we split a long video into 90-frame clips and trained on these clips separately. PSNR and SSIM [72] are



GT Ours-InstantNGP HyperNeRF [52] Nerfies [51] NSFF [32] NV [38] NeRF [45]

Fig. 8: Qualitative comparisons on the HyperNeRF-*vrig* dataset. An error map is demonstrated beside each rendered image.

Table 1: Quantitative comparisons on Plenoptic Video [31] for multi-camera dynamic scenes.

Method	PSNR↑	Training Time (GPU Hours)	Rendering Time (s/img)
Neural Volumes [38]	22.797	-	-
LLFF [44]	23.238	-	-
NeRF-T [31]	28.448	-	90
DyNeRF [31]	29.580	1344	90
Ours-InstantNGP	**30.293**	**5.5**	**10.8**
Ours-TensoRF-VM	**30.692**	**6**	**22.1**

reported for evaluating the rendering performance. We report training time with RTX A6000 and rendering time with RTX 2080Ti.

5.1 Comparison with State-of-The-Art Methods

DyNeRF [31] and HyperNeRF [52] are considered for multi- and single-camera settings. Besides the two methods, we also quote the results of other baseline methods reported in their paper.

5.1.1 On multi-camera dataset

In Tab. 1, we report our results with both InstantNGP and TensoRF backbones. Training time and rendering time of DyNeRF are quoted from their paper. Our method reaches a higher PSNR while significantly reducing the training and rendering time. We further compare the rendered images in Fig. 6. Images of comparison methods are again quoted from the result images in DyNeRF's paper. Our method with InstantNGP renders images with 12% of the time required by DyNeRF while being comparable. Besides, our method with TensoRF-VM achieves better performance on fast-moving objects. As demonstrated in Fig. 7, we compare our rendered results with extracted frames from DyNeRF's result video. Since the code and rendering parameters of DyNeRF are not publically available, we manually select similar camera poses and timestamp for comparison. We can observe that the knife in DyNeRF's results is blurry, while our method yields clearer results.

On Immersive Video, unfortunately, we are not able to directly our method against [6] as their evaluation data are not publically available. While [6] may have better performance and work in real-time on a laptop, our paper offers several unique benefits such as accepting single-camera inputs, enabling temporal interpolation, and significantly faster training speed.

5.1.2 On single-camera dataset

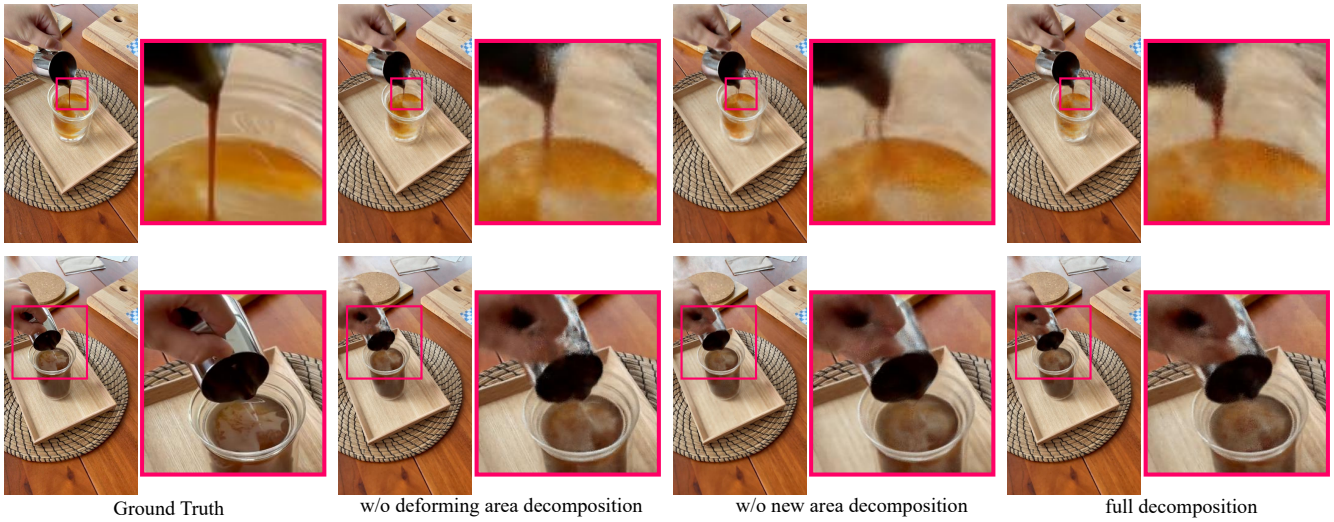
A challenging and practical appealing setting is reconstructing and rendering without per-frame multi-view observations, *i.e.*, capturing with a single camera. Our method with TensoRF backbone is not reported on this dataset, since we find that the GPU memory required for training is too large with the default model setting. We compare our method with SoTA single-camera reconstruction methods in Tab. 2. Note that NSFF requires data-driven depth and optical flow priors. We can observe that our method outperforms HyperNeRF in terms of PSNR, but is slightly worse than HyperNeRF on SSIM. We presume the reason is that our method generates more accurate, but less sharp images compared to HyperNeRF. Visual comparisons can be found at Fig. 8. We can observe that HyperNeRF sometimes has misalignment between the rendered and real images regarding moving objects, such as the wire in the second row. We attribute the misalignment problem to not correctly modeling the deformation. The incorrect modeling is partially caused by treating all pointing as deforming in their representation. Lacking decomposing static and dynamic areas also leads to a flickering background (demonstrated in our video). In our method, by decomposing static and dynamic areas, the deformation field is regularized to only model dynamic areas.

5.2 Ablation Studies

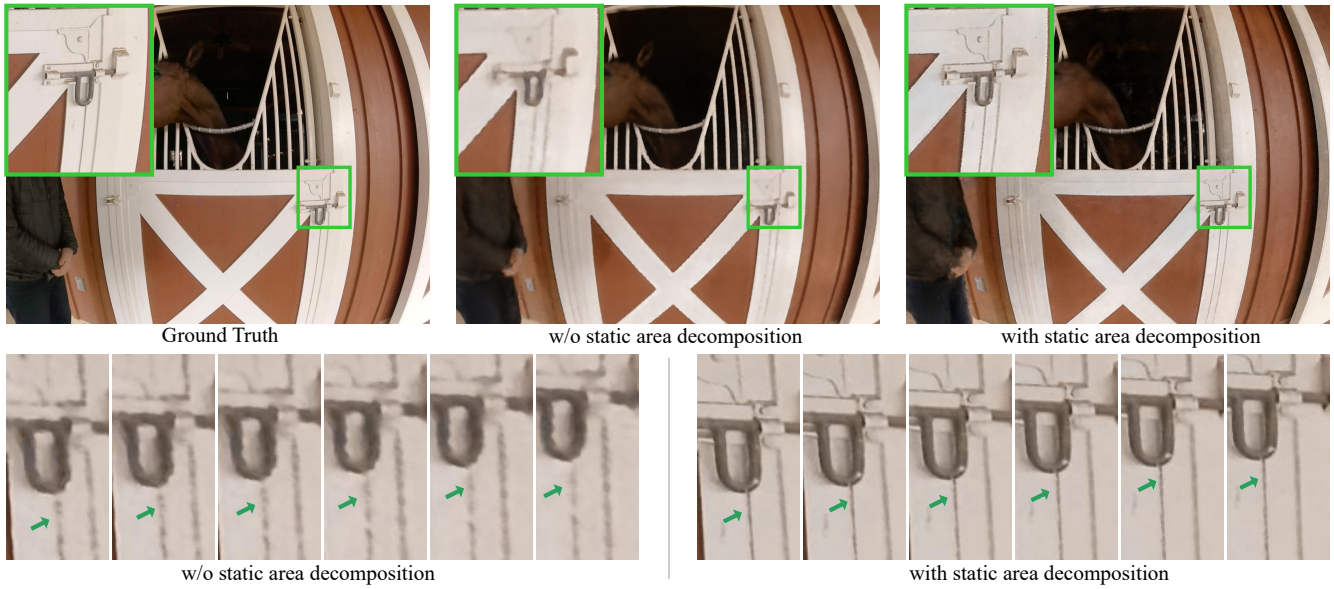
5.2.1 Impact of Decomposition

We first study the necessity of the proposed three categories for decomposition. Visual comparisons of different decomposition variants are demonstrated in Fig. 9 and quantitative results are reported in Tab. 3. First, we study the impact of decomposing deforming and new areas on a single-camera dataset. We can observe from Fig. 9(a) that removing new area decomposition leads to failure of modeling the newly poured out espresso, and removing deforming area decomposition leads to a blurred hand and cup. Second, we study the impact of our decomposition on the multi-camera dataset. Fig. 9(b) demonstrates that the static area becomes sharper after decomposing static areas. Besides, without static area decomposition, we observe that the background is flickering as we render images with novel time and view.

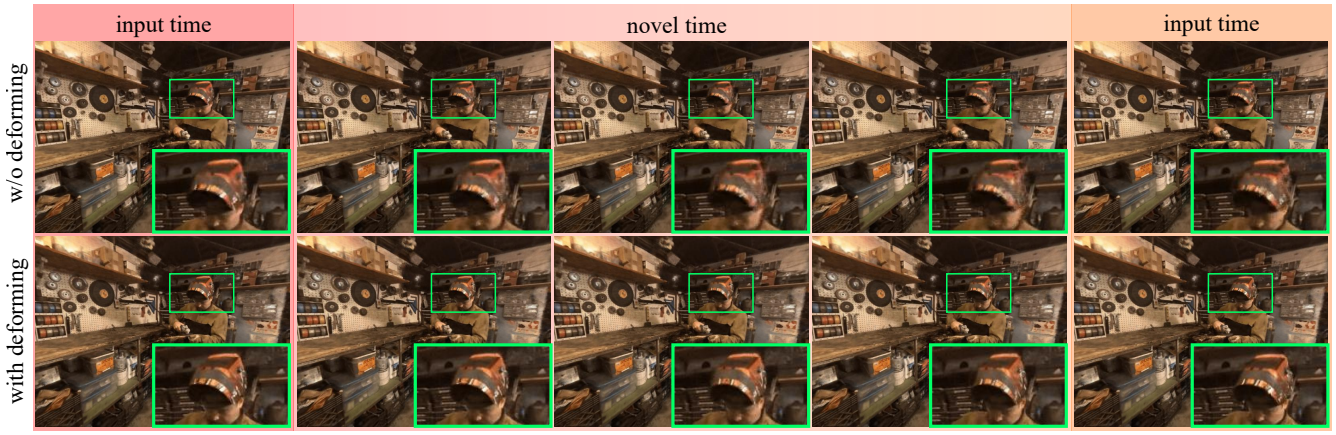
Finally, we study the impact of deforming area decomposition on the multi-camera dataset. This ablation study is motivated by the observation that rendering with and without deformation modeling leads to little difference (PSNR difference less than 0.1). We presume that this is because the motion of objects between frames is small from cameras with a high FPS recording rate. Therefore, modeling all dynamic areas with a newness field can still produce a smooth interpolation. We manually downsample the frame sampling rate for training in Fig. 9(c) to enlarge the motion between frames. We can observe that without deformation modeling, the moving helmet becomes first disappeared and then reappeared when interpolating between two training timestamps. As a comparison, the content of the helmet is well-preserved if the deformation is modeled. To further validate the effectiveness of the deformation decomposition, we study the impact of the size of deformation network (*i.e.*, num of layers and hidden dimensions) w.r.t. to different temporal sampling rate in Fig. 10. We can observe that the deformation decomposition becomes important if we consider a sparser temporal sampling rate. The results support our claimed contribution that deformation module is for regularizing temporal behaviors. Therefore, if the temporal resolution is high in the input videos, we can safely skip the deformation modeling, which validates the flexibility of our framework.



(a) Results after removing deforming and new area decomposition on the single-camera dataset (HyperNeRF).



(b) Ablation of static area decomposition on the multi-camera dataset (Immersive Video). Second row: novel time and view rendering.



(c) Ablation of deforming area decomposition on the multi-camera dataset (Immersive Video). Every 8 frames are used for training.

Fig. 9: Ablation of scene decompositions with InstantNGP as the backbone. (a) For the single-camera dataset (HyperNeRF), the full decomposition can well reconstruct the newly generated fluid and moving cup. (b) For multi-camera dataset (Immersive Video), static area decomposition leads to a clearer background and suppress flickering background. (c) When the recording frame rate is low (∼4 FPS) and objects move faster, deformation decomposition can help generate smoother temporal interpolation results.

Table 2: Per-scene quantitative comparisons on HyperNeRF-*vrig* [52] for single-camera dynamic scenes.

Method	Rendering Time (s/img)	Broom		3D Printer		Chicken		Peel Banana		**Mean**	
		PSNR↑	SSIM↑	PSNR↑	SSIM↑	PSNR↑	SSIM↑	PSNR↑	SSIM↑	PSNR↑	SSIM↑
NeRF [45]	∼75	19.9	0.653	20.7	0.780	19.9	0.777	20.0	0.769	20.1	0.745
NV [38]	<0.03	17.7	0.623	16.2	0.665	17.6	0.615	15.9	0.380	16.9	0.571
NSFF [32]*	∼90	26.1	0.871	27.7	0.947	26.9	0.944	24.6	0.902	26.3	0.916
Nerfies [51]	∼90	19.2	0.567	20.6	0.830	26.7	0.943	22.4	0.872	22.2	0.803
HyperNeRF [52]	∼90	19.3	0.591	20.0	0.821	26.9	0.948	23.3	0.896	22.4	**0.814**
Ours-InstantNGP	4.8	21.7	0.635	22.9	0.810	26.3	0.905	24.0	0.863	**23.7**	0.803

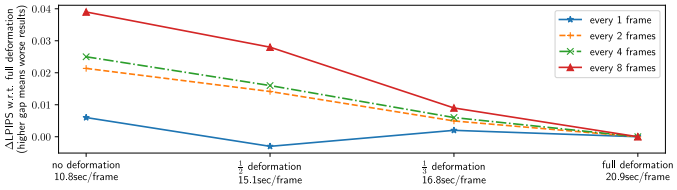


Fig. 10: Ablation of the deformation decomposition on Immersive Video. We test the relative performance under different temporal resolution settings. The results demonstrate the effectiveness of the deformation modeling with a sparse temporal sampling rate for multi-camera inputs.

Table 3: Quantitative results of different decomposition variants with InstantNGP as the backbone.

Method	PSNR↑	SSIM↑	$\text{LPIPS}_{\text{VGG}}\downarrow$	$\text{LPIPS}_{\text{Alex}}\downarrow$
on HyperNeRF-interp				
w/o deforming	28.2	0.820	0.358	0.220
w/o new	28.2	0.837	0.323	0.188
full	29.2	0.858	0.294	0.163
on Immersive Video ("Horse")				
w/o static	27.0	0.860	0.423	0.255
w/ static	27.4	0.871	0.425	0.295
on Immersive Video ("Welder" with every 8 frames)				
w/o deforming	26.1	0.826	0.366	0.195
w/ deforming	25.2	0.800	0.327	0.168

The ablation studies validate our proposed method in three aspects: 1) the necessity of modeling all of the three areas on single-camera datasets, 2) the necessity of decomposing static areas on multi-camera datasets, and 3) the necessity of deforming decomposition on inputs with large frame-wise motion even for multi-camera datasets.

5.2.2 Scene Decomposition Regularizing

In our method, we use α to balance the ratio of being deforming and new in Eq. (3). A larger α encourages the scene to contain fewer deforming areas. As introduced in the previous section, single-camera datasets are more sensitive to the deformation field, thus we study the impact of α in Fig. 12 on a scene from HyperNeRF. We can observe

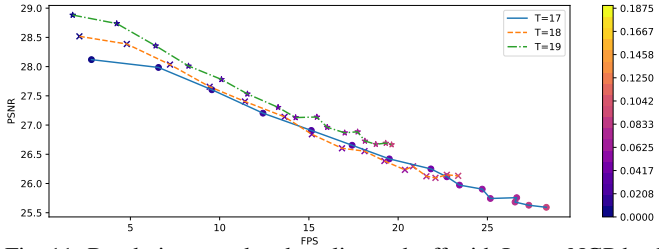


Fig. 11: Rendering speed and quality tradeoff with InstantNGP backbone. The color of the marker demonstrates the value of exponential stepping during ray marching.

Table 4: Quantitative results of rendering with different bitrate budgets for streaming.

Method	PSNR↑	SSIM↑	$\text{LPIPS}_{\text{VGG}}\downarrow$	Bitrates (MB/frame)
Immersive [6]	-	-	-	∼0.5
Ours-TensoRF-CP				
k=0.50	25.200	0.754	0.284	0.041
k=1.00	25.798	0.846	0.264	0.058
k=4.00	25.870	0.835	0.266	0.114
k=16.00	25.885	0.857	0.244	0.333
Ours-TensoRF-VM				
k=0.05	26.093	0.866	0.184	3.423
k=0.25	26.032	0.865	0.188	6.357
k=0.50	26.187	0.872	0.192	9.942
k=1.00	26.203	0.878	0.173	17.112

that over-suppressing deforming areas ($\alpha = 1$) lead to blurry moving objects, and under-suppressing deforming areas ($\alpha = 0.005$) leads to a noisy scene. The reason behind the blur from large α is the same as the second row in Fig. 3 and Fig. 9(c): falsely modeling a moving object as first-disappear-then-reappear. A good practice is that we can start with a relatively large α to penalize deforming areas and then gradually allow areas to deform by decreasing α. Note that in practice, we can reduce the number of frames when tuning α since our representation is streamable. The fewer number of frames, the smaller model size. Thus tuning α can be efficient in our framework.

5.2.3 Streaming Bitrates

An important metric for a streaming service is the bitrate. To render a new frame, the user is usually sensitive to the new data needed to download. We can easily tune the bitrate requirements in our method by setting the value of k. In Tab. 4, we report the bitrate for streaming a new frame with different k values. The testing data is a sequence from Immersive Video with 90 frames. Note that k denotes new channels needed for rendering a new frame and rendering the first frame still follows the channels required for static scenes (96 for TensoRF-CP and 4 for TensoRF-VM). For fair comparisons, bitrate is computed by the total model size over the number of frames.

The TensoRF-CP based model achieves low bitrate and reasonable performance, while the cost of TensoRF-VM is higher, but the performance gain is obvious. We further present rendering results in Fig. 13. We can observe clearer details of the background (*i.e.*, car) and the moving objects (*i.e.*, person) with increased bitrate budgets. The above results validate the extensibility of our framework.

5.2.4 Rendering Speed and Quality

The performance of our framework is highly correlated with the chosen backbones. Thus, in our method, there exists a tradeoff between rendering speed and quality, mainly affected by predefined model size and rendering hyperparameters. In Fig. 11, we present the rendering FPS and PSNR with different hyperparameter settings. We consider two parameters: T for the hash table size and the stepping value during ray marching. Scenes from the Immersive Video dataset are considered.

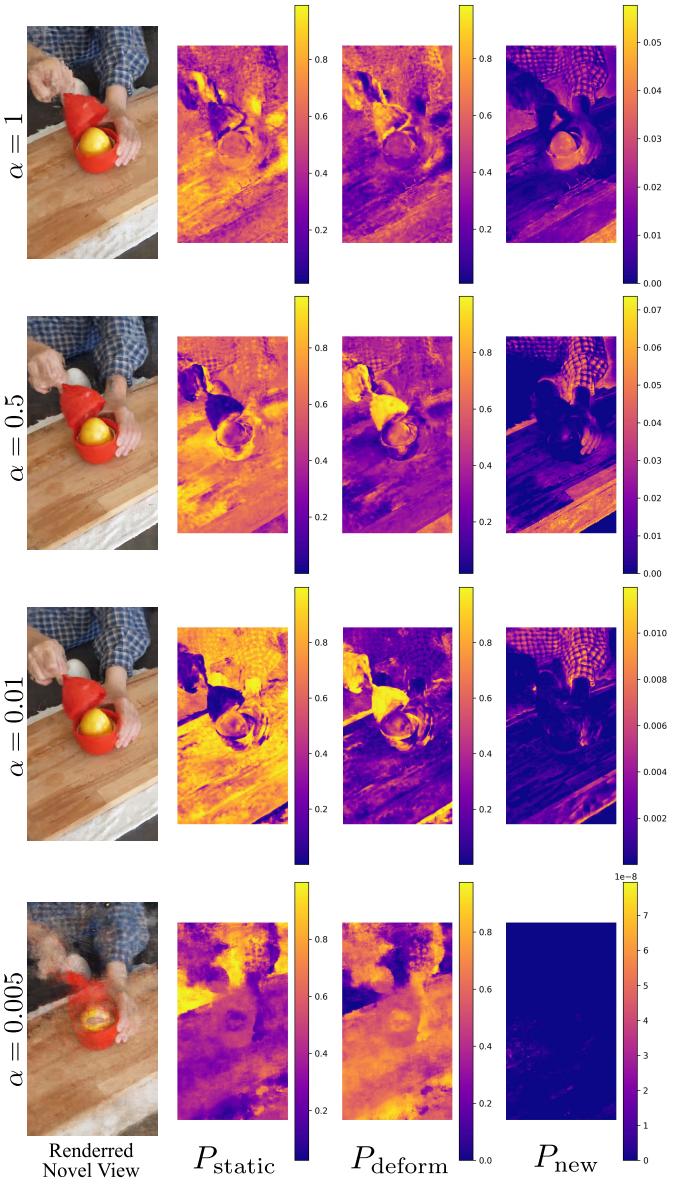


Fig. 12: Visualization of the decomposition and rendering results on HyperNeRF under different scene regularization settings.

Fig. 13: Comparison of results with different bitrate budgets for streaming. On each image, we report the model name followed by the value of k and the bitrate respectively.

Fig. 14: Our method fails when inputs are with different imaging configurations (*e.g.*, exposure).

We can observe that our method inherits the flexibility of the backbone, and we can easily tune the parameters to obtain the desired speed and quality.

6 Limitation and Failure Cases

Our method models each frame in the scene with local feature channels, which enables streaming but limits the representation of long-range repeated activities. For example, the activity of pouring espresso in Fig. 2 may repeat several times in a scene. Further modeling the repeating activities can reduce redundancy and improve the reconstruction quality by leveraging all the views of the same object. Moreover, our method assumes input multi-view images are with the same camera imaging configuration (*e.g.*, exposure). A failure example from Immersive Video is demonstrated in Fig. 14. Though view dependency can still be modeled in the framework, the model tends to generate floating points to overfit the training views. Recent progress that considers the photography process [41, 43] may help solve the issue. Our method is still computational expensive for real-time VR applications. Further baking and caching the decomposition results may help make the framework more efficient.

7 Conclusion

We present a framework for representing dynamic scenes from both multi- and single-camera captured images. The key components of our framework are the decomposition module and the feature streaming module. The decomposition module decomposes the scene into static, deforming, and new areas. A sliding window based hybrid representation is then designed for efficiently modeling the decomposed neural fields. Experiments on multi- and single-camera datasets validate our method's efficiency and effectiveness. Extensive ablation studies further provide insight into the model design, such as the necessity of modeling deformation in large-motion scenes captured by camera arrays.

References

[1] B. Attal, J.-B. Huang, C. Richardt, M. Zollhoefer, J. Kopf, M. O'Toole, and C. Kim. HyperReel: High-fidelity 6-DoF video with ray-conditioned sampling. *arXiv preprint arXiv:2301.02238*, 2023. 2

[2] A. Bansal, M. Vo, Y. Sheikh, D. Ramanan, and S. Narasimhan. 4d visualization of dynamic events from unconstrained multi-view videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5366–5375, 2020. 2

[3] M. Bemana, K. Myszkowski, H.-P. Seidel, and T. Ritschel. X-fields: Implicit neural view-, light-and time-image interpolation. *ACM Transactions on Graphics (TOG)*, 39(6):1–15, 2020. 2

[4] M. Boss, R. Braun, V. Jampani, J. T. Barron, C. Liu, and H. Lensch. Nerd: Neural reflectance decomposition from image collections. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 12684–12694, 2021. 2

[5] M. Boss, V. Jampani, R. Braun, C. Liu, J. T. Barron, and H. P. Lensch. Neural-pil: Neural pre-integrated lighting for reflectance decomposition. In *Advances in Neural Information Processing Systems*, 2021. 2

[6] M. Broxton, J. Flynn, R. Overbeck, D. Erickson, P. Hedman, M. Duvall, J. Dourgarian, J. Busch, M. Whalen, and P. Debevec. Immersive light field video with a layered mesh representation. *ACM Transactions on Graphics (TOG)*, 39(4):86–1, 2020. 2, 5, 6, 8

[7] J. Carranza, C. Theobalt, M. A. Magnor, and H.-P. Seidel. Free-viewpoint video of human actors. *ACM transactions on graphics (TOG)*, 22(3):569–577, 2003. 2

[8] E. R. Chan, C. Z. Lin, M. A. Chan, K. Nagano, B. Pan, S. D. Mello, O. Gallo, L. Guibas, J. Tremblay, S. Khamis, T. Karras, and G. Wetzstein. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16123–16133, 2022. 2

[9] A. Chen, Z. Xu, A. Geiger, J. Yu, and H. Su. Tensorf: Tensorial radiance fields. In *Proceedings of the European Conference on Computer Vision*, 2022. 1, 2, 3, 4, 5

[10] J. Chibane, G. Pons-Moll, et al. Neural unsigned distance fields for implicit function learning. *Advances in Neural Information Processing Systems*, 33, 2020. 2

[11] A. Collet, M. Chuang, P. Sweeney, D. Gillett, D. Evseev, D. Calabrese, H. Hoppe, A. Kirk, and S. Sullivan. High-quality streamable free-viewpoint video. *ACM Transactions on Graphics (ToG)*, 34(4):1–13, 2015. 2

[12] N. Deng, Z. He, J. Ye, B. Duinkharjav, P. Chakravarthula, X. Yang, and Q. Sun. Fov-nerf: Foveated neural radiance fields for virtual reality. *IEEE Transactions on Visualization and Computer Graphics*, 28:3854–3864, 2021. 2

[13] Y. Du, Y. Zhang, H.-X. Yu, J. B. Tenenbaum, and J. Wu. Neural radiance flow for 4d view synthesis and video processing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021. 1, 2

[14] J. Fang, L. Xie, X. Wang, X. Zhang, W. Liu, and Q. Tian. Neusample: Neural sample field for efficient view synthesis. *arXiv:2111.15552*, 2021. 2

[15] J. Fang, T. Yi, X. Wang, L. Xie, X. Zhang, W. Liu, M. Nießner, and Q. Tian. Fast dynamic radiance fields with time-aware neural voxels. *arXiv preprint arXiv:2205.15285*, 2022. 1, 2

[16] W. Gan, H. Xu, Y. Huang, S. Chen, and N. Yokoya. V4d: Voxel for 4d novel view synthesis. *arXiv preprint arXiv:2205.14332*, 2022. 2

[17] H. Gao, R. Li, S. Tulsiani, B. Russell, and A. Kanazawa. Monocular dynamic view synthesis: A reality check. In *Neural Information Processing Systems (Neurips)*, 2022. 3

[18] 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*, pp. 14346–14355, October 2021. 2

[19] S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen. The lumigraph. In J. Fujii, ed., *SIGGRAPH*, pp. 43–54. ACM, 1996. 2

[20] 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*, pp. 5875–5884, October 2021. 2

[21] 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*, pp. 867–876, 2022. 2

[22] R. Jain and K. Wakimoto. Multiple perspective interactive video. In *Proceedings of the international conference on multimedia computing and systems*, pp. 202–211. IEEE, 1995. 2

[23] H. Jang and D. Kim. D-tensorf: Tensorial radiance fields for dynamic scenes. *ArXiv*, abs/2212.02375, 2022. 2

[24] T. Kanade, P. Rander, and P. Narayanan. Virtualized reality: Constructing virtual worlds from real scenes. *IEEE MultiMedia*, 4(1):34–47, 1997. 2

[25] S. Kobayashi, E. Matsumoto, and V. Sitzmann. Decomposing nerf for editing via feature field distillation. In *arXiv*, 2022. 2

[26] J. Kopf, X. Rong, and J.-B. Huang. Robust consistent video depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1611–1621, 2021. 2

[27] 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*, pp. 5742–5752. PMLR, 2021. 2

[28] A. Kurz, T. Neff, Z. Lv, M. Zollhofer, and M. Steinberger. Adanerf: Adaptive sampling for real-time rendering of neural radiance fields. In *European Conference on Computer Vision*, 2022. 2

[29] M. Levoy and P. Hanrahan. Light field rendering. In J. Fujii, ed., *SIGGRAPH*, pp. 31–42. ACM, 1996. 2

[30] L. Li, Z. Shen, Z. Wang, L. Shen, and P. Tan. Streaming radiance fields for 3d video synthesis. In *Neural Information Processing Systems (Neurips)*, 2022. 2

[31] T. Li, M. Slavcheva, M. Zollhöfer, S. Green, C. Lassner, C. Kim, T. Schmidt, S. Lovegrove, M. Goesele, R. Newcombe, and Z. Lv. Neural 3d video synthesis from multi-view video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5521–5531, June 2022. 2, 5, 6

[32] Z. Li, S. Niklaus, N. Snavely, and O. Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021. 1, 2, 6, 8

[33] Z. Li, L. Song, C. Liu, J. Yuan, and Y. Xu. Neulf: Efficient novel view synthesis with neural 4d light field. In *Eurographics Symposium on Rendering*, 2022. 2

[34] D. B. Lindell, J. N. P. Martel, and G. Wetzstein. Autoint: Automatic integration for fast neural volume rendering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021. 2

[35] J.-W. Liu, Y.-P. Cao, W. Mao, W. Zhang, D. J. Zhang, J. Keppo, Y. Shan, X. Qie, and M. Z. Shou. Devrf: Fast deformable voxel radiance fields for dynamic scenes. *arXiv preprint arXiv:2205.15723*, 2022. 1, 2

[36] L. Liu, J. Gu, K. Z. Lin, T. Chua, and C. Theobalt. Neural sparse voxel fields. In *Advances in Neural Information Processing Systems*, 2020. 2, 3

[37] R. Liu, Y. Sun, J. Zhu, L. Tian, and U. S. Kamilov. Recovery of continuous 3d refractive index maps from discrete intensity-only measurements using neural fields. *Nature Machine Intelligence*, 4(9):781–791, 2022. 1

[38] 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*, 38(4):1–14, 2019. 2, 5, 6, 8

[39] S. Lombardi, T. Simon, G. Schwartz, M. Zollhöfer, Y. Sheikh, and J. M. Saragih. Mixture of volumetric primitives for efficient neural rendering. *ACM Trans. Graph.*, 40(4):59:1–59:13, 2021. 2

[40] X. Luo, J.-B. Huang, R. Szeliski, K. Matzen, and J. Kopf. Consistent video depth estimation. *ACM Transactions on Graphics (ToG)*, 39(4):71–1, 2020. 2

[41] R. Martin-Brualla, N. Radwan, M. S. Sajjadi, J. T. Barron, A. Dosovitskiy, and D. Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7210–7219, 2021. 9

[42] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4460–4470, 2019. 2

[43] 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*, pp. 16190–16199, 2022. 1, 9

[44] 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 (Proceedings of SIGGRAPH)*, 38(4), 2019. 5, 6

[45] 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*, pp. 405–421. Springer, 2020. 1, 2, 3, 6, 8

[46] T. Müller, A. Evans, C. Schied, and A. Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, July 2022. doi: 10.1145/3528223.3530127 1, 2, 3, 4, 5

[47] T. Neff, P. Stadlbauer, M. Parger, A. Kurz, J. H. Mueller, C. R. A. Chaitanya, A. S. Kaplanyan, and M. Steinberger. DONeRF: Towards Real-Time Rendering of Compact Neural Radiance Fields using Depth Oracle Networks. *Computer Graphics Forum*, 40(4), 2021. doi: 10.1111/cgf.14340 2

[48] M. Niemeyer, L. Mescheder, M. Oechsle, and A. Geiger. Occupancy flow: 4d reconstruction by learning particle dynamics. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 5379–5389, 2019. 2

[49] J. Ost, F. Mannan, N. Thuerey, J. Knodt, and F. Heide. Neural scene graphs for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2856–2865, 2021. 2

[50] 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*, pp. 165–174, 2019. 2

[51] K. Park, U. Sinha, J. T. Barron, S. Bouaziz, D. B. Goldman, S. M. Seitz, and R. Martin-Brualla. Nerfies: Deformable neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5865–5874, October 2021. 1, 2, 5, 6, 8

[52] 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.*, 40(6), dec 2021. 1, 2, 5, 6, 8

[53] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32:8026–8037, 2019. 5

[54] M. Piala and R. Clark. Terminerf: Ray termination prediction for efficient neural rendering. *International Conference on 3D Vision*, pp. 1106–1114, 2021. 2

[55] A. Pumarola, E. Corona, G. Pons-Moll, and F. Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10318–10327, 2021. 1, 2

[56] D. Rebain, W. Jiang, S. Yazdani, K. Li, K. M. Yi, and A. Tagliasacchi. Derf: Decomposed radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14153–14161, 2021. 2

[57] 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*, pp. 14335–14345, October 2021. 2

[58] Sara Fridovich-Keil and Alex Yu, M. Tancik, Q. Chen, B. Recht, and A. Kanazawa. Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 1, 2, 3

[59] H. Schirmacher, L. Ming, and H.-P. Seidel. On-the-fly processing of generalized lumigraphs. In *Computer Graphics Forum*, vol. 20, pp. 165–174. Wiley Online Library, 2001. 2

[60] P. Sharma, A. Tewari, Y. Du, S. Zakharov, R. Ambrus, A. Gaidon, W. T. Freeman, F. Durand, J. B. Tenenbaum, and V. Sitzmann. Seeing 3d objects in a single image via self-supervised static-dynamic disentanglement. *arXiv preprint arXiv:2207.11232*, 2022. 2

[61] P. P. Srinivasan, B. Deng, X. Zhang, M. Tancik, B. Mildenhall, and J. T. Barron. Nerv: Neural reflectance and visibility fields for relighting and view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7495–7504, 2021. 2

[62] 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*, pp. 5459–5469, 2022. 2, 3

[63] T. Takikawa, A. Evans, J. Tremblay, T. Müller, M. McGuire, A. Jacobson, and S. Fidler. Variable bitrate neural fields. In *ACM SIGGRAPH Conference Proceedings*, pp. 1–9, 2022. 3

[64] T. Takikawa, J. Litalien, K. Yin, K. Kreis, C. Loop, D. Nowrouzezahrai, A. Jacobson, M. McGuire, and S. Fidler. Neural geometric level of detail: Real-time rendering with implicit 3d shapes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11358–11367, 2021. 3

[65] M. Tancik, V. Casser, X. Yan, S. Pradhan, B. Mildenhall, P. P. Srinivasan, J. T. Barron, and H. Kretzschmar. Block-nerf: Scalable large scene neural view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8248–8258, 2022. 1

[66] E. Tretschk, A. Tewari, V. Golyanik, M. Zollhöfer, C. Lassner, and C. Theobalt. Non-rigid neural radiance fields: Reconstruction and novel view synthesis of a dynamic scene from monocular video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 12959–12970, October 2021. 1, 2

[67] V. Tschernezki, I. Laina, D. Larlus, and A. Vedaldi. Neural Feature Fusion Fields: 3D distillation of self-supervised 2D image representations. In *Proceedings of the International Conference on 3D Vision (3DV)*, 2022. 2

[68] V. Tschernezki, D. Larlus, and A. Vedaldi. Neuraldiff: Segmenting 3d objects that move in egocentric videos. In *International Conference on 3D Vision (3DV)*, pp. 910–919. IEEE, 2021. 2

[69] C. Wang, B. Eckart, S. Lucey, and O. Gallo. Neural trajectory fields for dynamic novel view synthesis. *arXiv preprint arXiv:2105.05994*, 2021. 1, 2

[70] F. Wang, S. Tan, X. Li, Z. Tian, and H. Liu. Mixed neural voxels for fast multi-view video synthesis. *ArXiv*, abs/2212.00190, 2022. 2

[71] L. Wang, J. Zhang, X. Liu, F. Zhao, Y. Zhang, Y. Zhang, M. Wu, J. Yu, and L. Xu. Fourier plenoctrees for dynamic radiance field rendering in real-time. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13524–13534, 2022. 2

[72] 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*, 13(4):600–612, 2004. 5

[73] S. Wizadwongsa, P. Phongthawee, J. Yenphraphai, and S. Suwajanakorn. Nex: Real-time view synthesis with neural basis expansion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8534–8543, 2021. 2

[74] 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*, pp. 16200–16209, 2022. 2, 3

[75] T. Wu, F. Zhong, A. Tagliasacchi, F. Cole, and C. Oztireli. D^2nerf: Self-supervised decoupling of dynamic and static objects from a monocular video. *arXiv preprint arXiv:2205.15838*, 2022. 2

[76] W. Xian, J.-B. Huang, J. Kopf, and C. Kim. Space-time neural irradiance fields for free-viewpoint video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9421–9431, 2021. 1, 2

[77] 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, pp. 641–676. Wiley Online Library, 2022. 1, 2

[78] 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*, pp. 13779–13788, 2021. 2

[79] J. C. Yang, M. Everett, C. Buehler, and L. McMillan. A real-time distributed light field camera. *Rendering Techniques*, 2002:77–86, 2002. 2

[80] A. Yu, R. Li, M. Tancik, H. Li, R. Ng, and A. Kanazawa. Plenoctrees for real-time rendering of neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5752–5761, October 2021. 2

[81] W. Yuan, Z. Lv, T. Schmidt, and S. Lovegrove. Star: Self-supervised tracking and reconstruction of rigid objects in motion with neural rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13144–13152, 2021. 2

[82] J. Zhang, X. Liu, X. Ye, F. Zhao, Y. Zhang, M. Wu, Y. Zhang, L. Xu, and J. Yu. Editable free-viewpoint video using a layered neural representation. *ACM Trans. Graph.*, 40(4):149:1–149:18, 2021. 2

[83] 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*, pp. 5449–5458, 2022. 3

[84] C. L. Zitnick, S. B. Kang, M. Uyttendaele, S. Winder, and R. Szeliski. High-quality video view interpolation using a layered representation. *ACM transactions on graphics (TOG)*, 23(3):600–608, 2004. 2