Fast Dynamic Radiance Fields with Time-Aware Neural Voxels

Jiemin Fang
Institute of AI & School of EIC, HUST
Wuhan, China
jaminfong@hust.edu.cn

Taoran Yi
School of EIC, HUST
Wuhan, China
taoranyi@hust.edu.cn

Xinggang Wang†
School of EIC, HUST
Wuhan, China
xgwang@hust.edu.cn

Lingxi Xie
Huawei Inc.
Beijing, China
198808xc@gmail.com

Xiaopeng Zhang
Huawei Inc.
Shanghai, China
zxphistory@gmail.com

Wenyu Liu
School of EIC, HUST
Wuhan, China
liuwy@hust.edu.cn

Matthias Nießner
Technical University of Munich
Munich, Germany
niessner@tum.de

Qi Tian
Huawei Inc.
Shenzhen, China
tian.qi1@huawei.com

Figure 1: We propose a radiance field framework equipped with time-aware neural voxels, which can learn dynamic scenes with an extremely fast convergence speed. Comparisons with D-NeRF [Pumarola et al. 2021] are shown. Sparse time-view images are taken and novel time and view images can be synthesized with our method.

ABSTRACT

Neural radiance fields (NeRF) have shown great success in modeling 3D scenes and synthesizing novel-view images. However, most previous NeRF methods take much time to optimize one single scene. Explicit data structures, e.g., voxel features, show great potential to accelerate the training process. However, voxel features face two big challenges to be applied to dynamic scenes, i.e., modeling temporal information and capturing different scales of point motions. We propose a radiance field framework by representing scenes with time-aware voxel features, named as TiNeuVox. A tiny coordinate deformation network is introduced to model coarse motion trajectories and temporal information is further enhanced in the radiance network. A multi-distance interpolation method is proposed and applied on voxel features to model both small and large motions. Our framework significantly accelerates the optimization of dynamic radiance fields while maintaining high rendering quality. Empirical evaluation is performed on both synthetic and real scenes. Our TiNeuVox completes training with only 8 minutes and
1 INTRODUCTION
Rendering plays a critically important role in various applications, e.g., virtual reality, interactive gaming, and movie production etc. High-quality and fast rendering techniques bring users realistic experience and make more applications possible. Recent neural rendering methods, represented by NeRF (neural radiance fields) [Mildenhall et al. 2020], have shown great power for modeling 3D scenes with compact implicit representations and synthesizing high-quality novel-view images. However, conventional NeRF methods [Barron et al. 2021; Mildenhall et al. 2020] mainly focus on static scenes, while real-life scenarios usually involve object motions or topological changes. A series of subsequent NeRF works [Li et al. 2021a; Park et al. 2021a,b; Pumarola et al. 2021; Tretschk et al. 2021a] improve radiance field construction towards dynamic scenes.

Besides, fast training and rendering speed of NeRF is needed in real-life applications. Conventional NeRF methods bear large time and computation cost to optimize the field networks, i.e. dozens of hours in general. Especially, most existing methods model dynamic scenes by introducing an additional deformation network with a similar scale of the radiance network, which maps point coordinates into a canonical space. This manner means much more cost for training and inferring dynamic fields. The cumbersome time cost impedes wide applications in real-life scenarios.

Representing scenes with explicit data structures shows great success in dramatically accelerating NeRF training and rendering [Hedman et al. 2021; Müller et al. 2022; Sun et al. 2022; Yu et al. 2022]. However, it is challenging to represent dynamic scenes with explicit structures from two main aspects. On the one hand, these scenes involve complicated point motions where encoding temporal information is required. One direct and simple solution is to expand the voxel grids with an additional time dimension. However, this manner will undoubtedly increase memory cost significantly. Changing voxel grids from 4D to 5D, i.e. $(C, N_x, N_y, N_z) \rightarrow (C, N_x, N_y, N_z, N_t)$, will multiply the storage cost by $N_t$. On the other hand, there usually exist motions of different scales. Voxels with high resolutions locate in small grids, which fail to model large motions; voxels in large grids fail to capture details with small motions.

To tackle the above challenges, we propose a new dynamic radiance field method, named as TiNeuVox, by representing scenes with time-aware voxel features. To encode temporal information, we first build a highly compressed deformation network which maps 3D point coordinates into a coarse canonical space. Voxel features are queried with the transformed coordinates. We further enhance the temporal information by feeding time and coordinate embeddings into the latter radiance network. Thus deviation introduced by point mapping can be automatically suppressed by the neural network. Moreover, we propose a multi-distance interpolation method, where features are obtained from voxels with multiple distances. In this way, both small and large motions can be modeled even though only one single-resolution voxel features are constructed.

We summarize our contributions as follows.

- We are the first to represent dynamic scenes with optimizable explicit data structures, which shows extremely high training efficiency.
- We encode coarse point motions with a tiny deformation network and enhance temporal information in the radiance network. A multi-distance interpolation technique is proposed to model both small and large motions with one single resolution of voxel features.
- We evaluate our method on both synthetic and real scenes, where our TiNeuVox achieves better or similar rendering quality with 8-MB storage by taking only 8 minutes, 150x faster than D-NeRF [Pumarola et al. 2021] and 192x faster than Hyper-NeRF [Park et al. 2021b].
2021] and producing sharper results by setting up key frames [Li et al. 2021b] etc. A series of articulated NeRF methods [Noguchi et al. 2021; Su et al. 2021; Weng et al. 2022; Xu et al. 2021] are also proposed to represent human body motions. Most of current dynamic NeRF methods still bear cumbersome training cost. We dramatically accelerate the training speed by introducing optimized explicit voxel features, while a compressed deformation network along with temporal information enhancement is designed. The overall framework achieves a good quality-speed trade-off via proper computation allocation on explicit and implicit representations.

2.2 Neural Rendering Acceleration

Rendering Acceleration. Though conventional NeRF methods show high rendering quality, it bears high latency as a series of points along each ray need to be sampled and inferred for volume rendering. Some works propose to reduce inference times for acceleration by improving sampling strategies [Arandjelović and Zisserman 2021; Fang et al. 2021; Lindell et al. 2021; Neff et al. 2021; Pa la and Clark 2021] or introducing efficient rendering techniques [Sitzmann et al. 2021]. Garbin et al. [2021]; Hedman et al. [2021]; Liu et al. [2020]; Reiser et al. [2021]; Sitzmann et al. [2019]; Wizadwongsa et al. [2021]; Yu et al. [2021a] store properties like densities produced by pre-trained radiance fields into explicit data structures, e.g. voxel grids or MIPs (multiple image planes). Only a few points need to be inferred by a small network for view-dependent color predictions. Though these methods have achieved real-time rendering performance, they still bear immense pre-training cost and cumbersome additional storage cost.

Convergence Acceleration. Some methods explore to reduce training cost from the generalization perspective. Chen et al. [2021]; Liu et al. [2022]; Wang et al. [2022, 2021a]; Yu et al. [2021b] substantially pre-train NeRF on various scenes to obtain generalizable properties or features. [Deng et al. 2021] achieves faster training speed with external depth information. Some works [Sun et al. 2022; Yu et al. 2022] propose to represent scenes with explicit voxel-grid features/properties and directly optimize these voxels for extremely fast convergence speed, reducing training time from hours to minutes. However, storage cost is significantly increased for storing voxel features. Recent works effectively reduce the storage cost via voxel hashing [Müller et al. 2022; Nießner et al. 2013], tensor decomposition [Chen et al. 2022; Kolda and Bader 2009] and bitrate dictionary lookup [Takikawa et al. 2022] while still maintaining surprisingly high training speed. These voxel-optimized methods yet only focus on static scenes, where voxel features are direct to construct for only spatial information. We introduce optimizing explicit voxel features into dynamic scenes. Temporal information is encoded to obtain time-aware neural voxel features. Our TiNeuVox achieves similar or better rendering performance than previous dynamic NeRF methods with training time reduced from days to 8 minutes.

3 METHOD

In this section, we first review methodologies of the original NeRF [Mildenhall et al. 2020] in Sec 3.1. Second, we describe how we represent dynamic scenes with explicit voxel features in Sec. 3.2. Then we propose to encode temporal information along with voxel features in Sec. 3.3. Finally, the overall framework and optimization procedures are presented in Sec. 3.4.

3.1 Preliminaries

Neural radiance fields are first proposed in [Mildenhall et al. 2020], which models 3D scenes by mapping the coordinate \((x, y, z)\) and view direction \((\theta, \phi)\) of each point in the space into its color \(c\) and density \(\sigma\). The mapping function is usually instantiated as a neural network \(\Phi_r\). This process can be formulated as

\[
c, \sigma = \Phi_r(x, y, z, \theta, \phi).
\] (1)

To get the expected color \(C(r)\) of the pixel in the image captured by the camera, a ray \(r(t) = o + td\) marching from the center of the camera to the pixel is involved, where \(o\) and \(d\) are the origin and direction of the ray respectively. \(t\) denotes the distance from one point to the camera, which ranges from a pre-defined near bound \(t_0\) to far bound \(t_f\). The pixel color is rendered by sampling a series of points along the ray and performing the classical volume rendering [Kajiya and Von Herzen 1984]:

\[
\hat{C}(r) = \sum_{i=1}^{N} T_i(1 - \exp(-\sigma_i \delta_i)) c_i,
\] (2)

where \(\delta_i\) is the distance between the \(i_{th}\) and \((i+1)_{th}\) sample point, \(N\) denotes the number of sampled points. Eq. 2 connects real 3D points with image pixels by accumulating colors \(c_i\) and densities \(\sigma_i\) of sample points along the ray. Finally, radiance fields are optimized via gradient descent by minimizing the following loss:

\[
\mathcal{L} = \|\hat{C}(r) - C(r)\|^2_2,
\] (3)

where \(C(r)\) denotes the groundtruth color of the pixel.

Besides, NeRF [Mildenhall et al. 2020] finds that details cannot be depicted by merely inputting \(x, y, z\) coordinates and \(\theta, \phi\) view directions. A positional encoding is introduced to map the input into a periodic formualation.

\[
y(x) = (\sin(2^0 x), \cos(2^0 x), ..., \sin(2^{L-1} x), \cos(2^{L-1} x)),
\] (4)

where \(L\) is a hyperparameter that controls the highest frequency of the input.

3.2 Multi-Distance Interpolation with Neural Voxels

Conventional NeRF methods [Barron et al. 2021; Mildenhall et al. 2020; Park et al. 2021a,b; Pumarola et al. 2021] build radiance fields with pure implicit representations, i.e. neural networks. Though this manner achieves promising rendering quality with high storage efficiency, it usually takes non-negligible time cost to optimize the fields, e.g. dozens of hours or even several days. To accelerate the convergence of radiance fields, we propose to represent scenes with explicit data structures except for implicit ones. To this end, neural voxels are introduced, which are a set of features organized as morphology of voxel grids. These features are designed to be further queried and inferred by neural networks to obtain certain properties, such as radiance and transmittance. As shown in Fig. 2, a scene is represented with grids of neural voxels \(V \in \mathbb{R}^{C_x \times N_x \times N_y \times N_z}\).
Figure 2: Overall framework of TiNeuVox. First, a deformation network $\Phi_d$ takes both point coordinates $\gamma(x, y, z)$ and encoded time embeddings $t_i = \Phi_t(\gamma(t_i))$ as input to obtain the shifted coordinates $(x', y', z')$. Then voxel features in grids with different sampling strides are queried and interpolated according to deformed coordinates. To enhance temporal information, coordinates $\gamma(x, y, z)$ and time embeddings $t_i$ are further concatenated with interpolated voxel features, $\gamma(v_s), \gamma(v_m)$, and $\gamma(v_l)$, which are finally fed into the radiance network to produce the density $\sigma$ and color $c$.

Figure 3: Illustration of multi-distance interpolation.

Figure 3: Illustration of multi-distance interpolation. To predict the property of one 3D point, neural voxels stored in eight vertices of the grid this point lies in are queried and interpolated trilinearly. The interpolated feature is then inferred by neural networks to predict the expected properties. Considering points in dynamic scenes may move with a dramatic motion trajectory, small grids of neural voxels have limited capacity to model these point movements. We propose a multi-distance interpolation method to model point motions with various scales. As shown in Fig. 3, besides the smallest grid, voxel features are also interpolated from vertices of larger grids. This means the final features for inference not only come from nearest voxels, but also from sub- and subsub-nearest voxels. In this way, small motions can be modeled via near voxels while motions in a large region are perceived with farther voxels.

We implement the above process as follows. When performing interpolation, we sample neural voxels from the pre-built grids $V$ with different strides $s_1, s_2, s_3, \ldots$. Then features are trilinearly interpolated with several sampled voxel grids respectively. Finally, these features are concatenated and fed into neural networks. This process can be formulated as

$$v = v_1 \oplus \cdots \oplus v_m \oplus \cdots \oplus v_M,$$

where $M$ denotes the total number of defined sampling strides, $x, y, z$ denote the coordinates of the 3D point.

Moreover, before being fed into the neural network for inference, these interpolated voxel features are first positional-encoded as in Eq. 4. This manner is of critical importance for compressing neural voxels into small sizes while maintaining strong performance for modeling details, which is evaluated in experiments (Sec. 4.3).

3.3 Temporal Information Encoding

Dynamic scenes involve complicated point motions in the space. We propose to encode temporal information from two perspectives as follows.

Coarse Coordinate Deformation. Like most previous implicit NeRF methods for dynamic scenes [Park et al. 2021a,b; Pumarola et al. 2021], we introduce a deformation network to shift coordinates of points which simulates the movement of points, but compress the network into a very small one. Denoting coordinates of one point as $(x, y, z)$ and the deformation network as $\Phi_d$, the coordinates are mapped into new ones according to encoded time embeddings $t_i = \Phi_t(y(t_i))$:

$$x', y', z' = \Phi_d(x, y, z, t_i).$$
We use only 3-layer MLPs (multilayer perceptrons) as the deformation network, which is much smaller than ones adopted in previous dynamic-scene methods. As this deformation network is applied on every sample point which accounts for a large potion of computation cost, we compress this network from both widths and depths for accelerating optimization and rendering processes.

Temporal Information Enhancement. As the aforementioned deformation network is severely compressed in our method, it may introduces unavoidable deviation for coordinate shifting due to its limited capacity. Besides, this deviation will be aggravated as neural voxels are queried according to point coordinates. Thus, final error not only comes from interpolation weights but also from the queried vertices. As shown in Fig. 2, to alleviate this deviation/mismatch, we propose to further enhance the temporal information by concatenating interpolated features in Eq. 5 with positional-encoded coordinates and neural-encoded temporal embeddings. All the concatenated features and embeddings are further fed into neural networks, where the above deviation will be automatically suppressed.

3.4 Overall Framework and Optimization

Our overall framework is illustrated in Fig. 2. First, the time stamp is encoded by a two-layer MLPs Φt, and then fed into a compressed deformation network Φd along with coordinates of the sampled point (x, y, z) to obtain shifted coordinates (x’, y’, z’) as in Eq. 6. The shifted coordinates are used for querying and interpolating neural voxels with a multi-distance manner as in Eq. 5. Then, the concatenated neural voxel σ as in Eq. 5, encoded time embeddings t = Φt(γ(t)) and original coordinates (x, y, z) are all concatenated to be fed into a narrow and shallow radiance network Φr to obtain the final density σ and color \(c, \sigma = \Phi_r(\gamma(a), t, \gamma(x, y, z), \gamma(d))\),

\[e, \sigma = \Phi_r(\gamma(a), t, \gamma(x, y, z), \gamma(d)),\]  

where γ denotes the positional encoding as in Eq. 4 and \(d = (\theta, \phi)\) represents the ray direction which is fed into Φr in the latter stage.

For each ray, we sample points evenly from the near bound to the far bound. By performing the above computation on each sampled point along a ray, the final predicted color can be obtained via volume rendering as Eq. 2. Parameters of all the neural voxels and networks can be optimized by minimizing the distance between predicted colors and groundtruth colors of image pixels as Eq. 3. Besides, following [Sun et al. 2022] we adopt two additional loss functions for regularization. One supervises predicted colors of all the sampled points along the ray with the groundtruth image pixel color for stabilization. The other one builds a cross-entropy loss on \(T_{N+1}\) to distinguish fore- and back-ground, where \(T_{N+1}\) denotes the accumulated transmittance for an additional point as computed in Eq. 2.

Besides, once we obtain the predicted density values with the radiance network, we filter points for the rest part of neural network inference with a pre-defined density threshold. This manner can effectively reduce cost of view-dependent color prediction but rarely affect the rendering quality.

4 EXPERIMENTS

In this section, we first provide our implementation details in Sec. 4.1. Then we show evaluation results and compare TiNeuVox with other related methods in Sec. 4.2. We further perform a series of ablation studies on the key components of TiNeuVox and provide detailed results and analysis in Sec. 4.3.

4.1 Implementation Details

We implement our framework as Fig. 2 mainly with PyTorch [Paszke et al. 2019] and provide two versions, i.e. TiNeuVox-S (small) and TiNeuVox-B (base). For TiNeuVox-S, neural voxels are constructed with a resolution of 100× and a channel number of 4; neural voxels in TiNeuVox-B are at 160×××. All neural voxels are initialized with zero-values. For acceleration, we set the initial resolutions of voxel grids as \(2^5\) of the given ones, which are doubled after 2k, 4k, and 6k iterations during training. This manner reduces the training time by 12% but achieves a similar PSNR. The channel dimension \(C_{t}\) of hidden layers is set as 64 for TiNeuVox-S and 256 for TiNeuVox-B. The dimension \(C_{t}\) of time embeddings is set the same as each positional-encoded voxel feature, i.e. \(20\) for TiNeuVox-S and \(30\) for TiNeuVox-B. The frequency number \(L\) of positional encoding (Eq. 4) is set as \(10\) for coordinates \((x, y, z)\), \(4\) for view direction \(d\), \(8\) for the time stamp \(t\), and \(2\) for neural voxels. Points are sampled with a step of half the voxel size along each ray for volume rendering.

For optimization, an Adam [Kingma and Ba 2015] optimizer is used with \((0.9, 0.99)\) \(\beta\) values. In each iteration, 4096 rays are randomly sampled from the whole dataset to form a batch. The initial learning rate is set as \(8 \times 10^{-2}\) for all voxels features, \(6 \times 10^{-4}\) for parameters of the deformation network \(\Phi_d\), and \(8 \times 10^{-4}\) for parameters of the other MLPs, which finally decays by \(0.1\) with an exponential schedule. The color regularization loss and background cross-entropy loss are weighted by \(10^{-2}\) and \(10^{-3}\) respectively. To further compress the neural voxel storage, we convert them into the half-precision floating-point format for the last 1k iterations. It takes 20k iterations in total on one single GeForce RTX 3090 GPU for every scene evaluated in this paper unless specified.

4.2 Evaluation

In this section, we evaluate our method on both synthetic and real dynamic scenes for novel view synthesis. Experimental results are compared with other state-of-the-art (SOTA) methods both quantitatively and qualitatively.

360° Synthetic Scenes. We adopt the dataset provided by D-NeRF [Pumarola et al. 2021] for synthetic-scene evaluation, containing 8 scenes with dynamic objects under large motions and realistic non-Lambertian materials. Each scene contains 50 – 200 images for training and 20 images for testing. To fairly compare with D-NeRF [Pumarola et al. 2021], each image is trained and rendered at 400 × 400 pixels.

As shown in Tab. 1, we provide two versions of our TiNeuVox as described in Sec. 4.1. Three metrics are used for evaluation, i.e. peak signal-to-noise ratio (PSNR), structural similarity (SSIM) [Wang et al. 2004] and learned perceptual image patch similarity (LPIPS) [Zhang et al. 2018]. Conventional NeRF [Mildenhall et al. 2020] and fast-convergence methods, including DirectVoxGO
Table 1: Comparisons about training/memory cost and rendering quality on synthetic scenes.

| Method         | w/ Time Enc. | w/ Explicit Rep. | Time | Storage | PSNR ↑ | SSIM ↑ | LPIPS ↓ |
|----------------|--------------|------------------|------|---------|--------|--------|---------|
| NeRF [Mildenhall et al. 2020] | ✗            | ✗                | ~ hours | 5 MB    | 19.00  | 0.87   | 0.18    |
| DirectVoxGO [Sun et al. 2022] | ✗            | ✓                | 5 mins  | 205 MB  | 18.61  | 0.85   | 0.17    |
| Plenoxels [Yu et al. 2022]     | ✗            | ✓                | 6 mins  | 717 MB  | 20.24  | 0.87   | 0.16    |
| T-NeRF [Pumarola et al. 2021]  | ✓            | ✗                | ~ hours | –       | 29.51  | 0.95   | 0.08    |
| D-NeRF [Pumarola et al. 2021]  | ✓            | ✗                | ~ hours | –       | 30.50  | 0.95   | 0.07    |
| TiNeuVox-S (ours)              | ✓            | ✓                | 8 mins  | 8 MB    | 30.75  | 0.96   | 0.07    |
| TiNeuVox-B (ours)              | ✓            | ✓                | 28 mins | 48 MB   | 32.67  | 0.97   | 0.04    |

Figure 4: Qualitative comparisons between D-NeRF [Pumarola et al. 2021] and our TiNeuVox on synthetic scenes.

Real Scenes. We further evaluate our method on real non-rigidly deforming scenes provided by HyperNeRF [Park et al. 2021b]. To obtain images from these scenes, a multi-view data capture rig is built with 2 Pixel 3 phones rigidly attached roughly 16cm apart. More details can be referred to in Park et al. [2021a,b]. We perform experiments on four scenes released by Park et al. [2021b], i.e. Broom, 3D Printer, Chicken, and Peel Banana. Following Park et al. [2021b], PSNR and MS-SSIM [Wang et al. 2003] are used as evaluation metrics. Each image is trained and rendered at half of 1080p resolutions, i.e. 960 × 540 pixels, for quantitative evaluation. To fairly compare with HyperNeRF [Park et al. 2021b]3, qualitative results are obtained at full-HD with roughly 1920 × 1080 pixels, taking 40k iterations while HyperNeRF takes 1M iterations.

Table 2: Quantitative comparisons on real dynamic scenes.

| Method         | Time | PSNR↑ | MS-SSIM↑ |
|----------------|------|-------|----------|
| NeRF [Mildenhall et al. 2020] | ~ hours | 20.1  | 0.745    |
| NV [Lombardi et al. 2019]     | ~ hours | 16.9  | 0.571    |
| NSFF [Li et al. 2021a]        | ~ hours | 26.3  | 0.916    |
| Nerfies [Park et al. 2021a]   | ~ hours | 22.2  | 0.803    |
| HyperNeRF [Park et al. 2021b] | 32 hours | 22.4  | 0.814    |
| TiNeuVox-S (ours)              | 10 mins | 23.4  | 0.813    |
| TiNeuVox-B (ours)              | 30 mins | 24.3  | 0.837    |

3We find LPIPS [Zhang et al. 2018] values reported in [Park et al. 2021b] hard to reproduce so we omit this evaluation metric.

3Time cost of HyperNeRF [Park et al. 2021b] is estimated according to descriptions in their paper but on TPU.

1We test this time cost by reproducing D-NeRF [Pumarola et al. 2021] on one single RTX 3090 GPU for fair comparison.
Figure 5: Qualitative comparisons between TiNeuVox and other methods on real dynamic scenes.

Table 3: Ablation study about components of encoding time information with TiNeuVox-B, i.e. deforming coordinates, enhancing temporal information, and encoding neural time embeddings.

| Deform Coord. | Enhance Temp. Info. | Enc. Neural Time Embeds. | PSNR | SSIM | LPIPS |
|---------------|---------------------|--------------------------|------|------|-------|
| ✓             | ✓                   | ✓                        | 32.668 | 0.971 | 0.041 |
| ✓             | ✓                   | ✗                        | 29.684 | 0.956 | 0.065 |
| ✓             | ✗                   | ✓                        | 31.473 | 0.968 | 0.045 |
| ✓             | ✓                   | ✗                        | 32.384 | 0.971 | 0.044 |

However, our rendered images are slightly more blurred than HyperNeRF. We deduce that far more iterations ($1M$) and additional regularization losses for real dynamic scenes in HyperNeRF matter. We would like to further explore these regularization techniques in future, which are compatible with our frameworks.

4.3 Ablation Study

In this section, we perform a series of experiments to study key components and factors involved in our method to better understand the mechanism and demonstrate the effectiveness. Following experiments are performed on all the synthetic dynamic scenes [Pumarola et al. 2021] and averaged metric values are reported.

Temporal Information Encoding. We study three components for encoding time information, i.e. coordinate deformation $\Phi_d$, temporal information enhancement as in Fig. 2, and encoding time embeddings with the neural network $\Phi_t$. As shown in Tab. 3, all of the three components is of critical importance to the final rendering performance.

Multi-distance Interpolation Effectiveness. We study effectiveness of the proposed multi-distance interpolation (MDI) and show results in Tab. 4. Experiments with a single $1$ sampling stride equal to interpolation without the multi-distance manner. For the small resolution setting with $100^3$, MDI brings a $0.467$ PSNR promotion. It can be observed the larger resolutions are, the bigger advantages MDI bring, i.e. $0.874$ for $160^3$ and $1.593$ for $256^3$. This is because larger the resolution is, a smaller region each grid can represent. Noting that for $256^3$, each grid is too small to capture complete motions without MDI.

To clearly demonstrate the MDI mechanism, we visualize gradient magnitudes on neural voxels interpolated with different distances, where red colors denote distant voxel samples have larger gradients and yellow colors denote near samples have larger gradients. Similar to ideas in Grad-CAM [Selvaraju et al. 2017], gradient magnitudes represent the activation strength or importance of voxel samples. As shown in Fig. 6, distant voxels samples (red) show higher importance for large-motion points (e.g. arms); small-motion points (e.g. head) prefer near voxel samples (yellow). We clarify the MDI mechanism as follows. The coordinate deformation network is highly compressed so it can only model a coarse motion. This deviation becomes more dominant for larger motions and may lead to wrong voxel queries. This exacerbates even more at higher-resolution voxel grids. MDI enables each voxel to perceive multi-distance points; hence even when wrong voxels are queried, voxel features can still provide correct information.

5 DISCUSSION AND CONCLUSION

Limitations & Future Works. We perform preliminary experiments on the NSFF [Li et al. 2021a] scene "truck", which contains a...
We thank Prof. Angela Dai for narration in the video, and Liangchen Song and Yingqing Rao for their valuable discussions and assistance. This work was supported by National Natural Science Foundation of China (NSFC No. 61733007 and No. 61876212).

REFERENCES
Rojta Arandjelovic and Andrew Zisserman. 2021. NeRF in detail: Learning to sample for view synthesis. arXiv:2106.05264 (2021).

Benjamin Attal, Eloi Laidlaw, Aaron Gokasan, Changil Kim, Christian Richardt, James Tompkin, and Matthew O’Toole. 2021. Törf: Time-of-flight radiance fields for dynamic scene view synthesis. NeurIPS (2021).

Jonathan T. Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P. Srinivasan. 2021. Mip-NeRF: A Multiresolution Hash Encoding for Anti-Aliasing Neural Radiance Fields. In ICCV.

Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. 2021. MVSNerf: Fast Generalizable Radiance Field Reconstruction From Multi-View Stereo. In ICCV.

Kangle Deng, Andrew Liu, Jun-Yan Zhu, and Deva Ramanan. 2021. Depth-supervised nerf: Fewer views and faster training for free. arXiv:2107.02791 (2021).

Yilun Du, Yinan Zhang, Hong-Xing Yu, Joshua B Tenenbaum, and Jaajun Wu. 2021. Neural radiance flow for 4d view synthesis and video processing. In ECCV.

Jiemin Fang, Lingxi Xie, Xinggang Wang, Xiaogang Zhang, Wenyu Liu, and Qi Tian. 2021. NeuSample: Neural Sample Field for Efficient View Synthesis. arXiv:2111.15552 (2021).

Chen Gao, Ayush Saraf, Johannes Kopf, and Jia-Bin Huang. 2021. Dynamic view synthesis from dynamic monocular video. In ICCV.

Stephan J. Garbin, Marek Kowalski, Matthew Johnson, Jamie Shotton, and Julien Valentin. 2021. FastNeRF: High-Fidelity Neural Rendering at 200FPS. In ICCV.

Peter Hedman, Pratul P. Srinivasan, Ben Mildenhall, Jonathan T. Barron, and Paul Debevec. 2021. Baking Neural Radiance Fields for Real-Time View Synthesis. In ICCV.

James T Kajiya and Brian P Von Herzen. 1984. Ray tracing volume densities. ACM SIGGRAPH computer graphics (1984).

Diederik P Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In ICLR.

Tahar Kolda and Brett W. Bader. 2009. Tensor decompositions and applications. SIAM review (2009).

Tianye Li, Mira Slavcheva, Michael Zollhoefer, Simon Green, Christoph Lassner, Changil Kim, Tanner Schmidt, Steven Lovegrove, Michael Goesele, and Zhaoyang Lv. 2021b. Neural 3d video synthesis. arXiv:2103.02597 (2021).

Zhengyi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. 2021a. Neural scene flow fields for space-time view synthesis of dynamic scenes. In CVPR.

Chen-Hsuan Lin, Wei-Chiu Ma, Antonios Torralba, and Simon Lucey. 2021. Barf: Bundle-adjusting neural radiance fields. In ICCV.

David B Lindell, Julien NP Martel, and Gordon Wetzstein. 2021. Autoint: Automatic integration for fast neural volume rendering. In CVPR.

Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. 2020. Neural Sparse Voxel Fields. In NeurIPS.

Yuan Liu, Sida Peng, Lingjie Liu, Quentin Wang, Peng Wang, Tobiacht Christian, Xiaowei Zhou, and Wenzing Wang. 2022. Neural Rays for Occlusion-aware Image-based Rendering. In CVPR.

Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. 2019. Neural volumes: learning dynamic renderable volumes from images. ACM Transactions on Graphics (2019).

Ricardo Martin-Brualla, Noha Radwan, Mehrdad S. M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, and Daniel Duckworth. 2021. NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections. In CVPR.

Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. 2020. NeRF: Representing scenes as neural radiance fields for view synthesis. In ICCV.

Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. 2022. Instant Neural Graphics Primitives with a Multiresolution Hash Encoding. arXiv:2201.05989 (2022).

Thomas Neff, Pascal Staal, Mathias Parger, Andreas Kurz, Joerg H. Mueller, Chakravarthy R. Alla Chaitanya, Anton S. Kaplanyan, and Markus Steinberger. 2021. DONeRF: Towards Real-Time Rendering of Compact Neural Radiance Fields using Depth Oracle Networks. Computer Graphics Forum (2021).

Matthias Nießner, Michael Zollhoefer, Shahrab Izadi, and Marc Stamminger. 2013. Real-time 3D reconstruction at scale using voxel hashing. ACM Transactions on Graphics (ToG) (2013).

Atsuhito Noguchi, Xiao Sun, Stephen Lin, and Tatsuya Harada. 2021. Neural articulated radiance field. In ICCV.

Keunhong Park, Utkarsh Sinha, Jonathan T Barron, Sofian Bouaziz, Dan B Goldman, Steven M. Seitz, and Ricardo Martin-Brualla. 2021a. Nerfies: Deformable Neural Radiance Fields. ICCV (2021).

Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T. Barron, Sofian Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M. Seitz. 2021b. HyperNeRF: A Higher-Dimensional Representation for Topologically Varying Neural Radiance Fields. ACM Trans. Graph. (2021).

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. PyTorch: An imperative style, high-performance deep learning library. NeurIPS (2019).

Martin Piaia and Ronald Clark. 2021. TermiNeRF: Ray Termination Prediction for Efficient Neural Rendering. In 3DV.

Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. 2021. D-nerf: Neural radiance fields for dynamic scenes. In CVPR.

Christian Reizer, Songyou Peng, Vivi Liao, and Andreas Geiger. 2021. KiloNeRF: Speeding Up Neural Radiance Fields With Thousands of Tiny MLPs. In ICCV.

Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. 2017. Grad-cam: Visual explanations from deep networks via gradient-based localization. In ICCV.
Vincent Sitzmann, Semon Rezchikov, William T. Freeman, Joshua B. Tenenbaum, and Fredo Durand. 2021. Light Field Networks: Neural Scene Representations with Single-Evaluation Rendering. In NeurIPS.

Vincent Sitzmann, Justin Thies, Felix Heide, Matthias Nießner, Gordon Wetzstein, and Michael Zollhofer. 2019. Deepvoxels: Learning persistent 3d feature embeddings. In CVPR.

Shih-Yang Su, Frank Yu, Michael Zollhofer, and Helge Rhodin. 2021. A-neRF: Articulated neural radiance fields for learning human shape, appearance, and pose. NeurIPS (2021).

Cheng Sun, Min Sun, and Hwann-Tzong Chen. 2022. Direct Voxel Grid Optimization: Super-fast Convergence for Radiance Fields Reconstruction. In CVPR.

Towaki Takikawa, Alex Evans, Jonathan Tremblay, Thomas Müller, Morgan McGuire, Alec Jacobson, and Sanja Fidler. 2022. Variable Bitrate Neural Fields. In SIGGRAPH.

Edgar Tretschk, Ayush Tewari, Vladislav Golyanik, Michael Zollhofer, Christoph Lassner, and Christian Theobalt. 2021a. Non-Rigid Neural Radiance Fields: Reconstruction and Novel View Synthesis of a Dynamic Scene From Monocular Video. In ICCV.

Edgar Tretschk, Ayush Tewari, Vladislav Golyanik, Michael Zollhofer, Christoph Lassner, and Christian Theobalt. 2021b. Non-rigid neural radiance fields: Reconstruction and novel view synthesis of a dynamic scene from monocular video. In ICCV.

Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T. Barron, and Pratul P. Srinivasan. 2022. Ref-NeRF: Structured View-Dependent Appearance for Neural Radiance Fields. CVPR (2022).

Liao Wang, Jiaxian Zhang, Xinhang Liu, Fuqiang Zhao, Yanshun Zhang, Yingliang Zhang, Minye Wu, Lan Xu, and Jingyi Yu. 2022. Fourier PlenOctrees for Dynamic Radiance Field Rendering in Real-time. arXiv:2202.08614 (2022).

Quanquan Wang, Zhicheng Wang, Kyle Genova, Pratul P Srinivasan, Howard Zhou, Jonathan T Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. 2021a. Ibrnet: Learning multi-view image-based rendering. In CVPR.

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. 2004. Image quality assessment: from error visibility to structural similarity. IEEE TIP (2004).

Zhou Wang, Eero P Simoncelli, and Alan C Bovik. 2003. Multiscale structural similarity for image quality assessment. In The Thirty-Seventh Asilomar Conference on Signals, Systems & Computers.

Zirui Wang, Shangze Wu, Weidi Xie, Min Chen, and Victor Adrian Prisacariu. 2021b. NeRF++: Neural radiance fields without known camera parameters. arXiv:2102.07864 (2021).

Chung-Yi Weng, Brian Curless, Pratul P Srinivasan, Jonathan T Barron, and Ira Kemelmacher-Shlizerman. 2022. Humanner: Free-viewpoint rendering of moving people from monocular video. In CVPR.

Suttisak Wizadwongsa, Pakkapon Phongthawee, Jiraphon Yenphraphai, and Supasorn Suwajanakorn. 2021. Nex: Real-time view synthesis with neural basis expansion. In CVPR.

Wenqi Xian, Jia-Bin Huang, Johannes Kopf, and Changil Kim. 2021. Space-time Neural Irradiance Fields for Free-Viewpoint Video. In CVPR.

Hongyi Xu, Thiemo Alldieck, and Cristian Sminchisescu. 2021. H-nerf: Neural radiance fields for rendering and temporal reconstruction of humans in motion. NeurIPS (2021).

Alex Yu, Sara Fridovich-Keil, Matthew Tancik, Qinlong Chen, Benjamin Recht, and Angjoo Kanazawa. 2022. Plenoxels: Radiance Fields without Neural Networks. In CVPR.

Alex Yu, Rui Long Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. 2021a. PlenOctrees for Real-Time Rendering of Neural Radiance Fields. In ICCV.

Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. 2021b. pixelNeRF: Neural radiance fields from one or few images. In CVPR.

Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. 2020. NeRF++: Analyzing and Improving Neural Radiance Fields. arXiv:2010.07492 (2020).

Richard Zhang, Philip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. 2018. The unreasonable effectiveness of deep features as a perceptual metric. In CVPR.