Ced-NeRF: A Compact and Efficient Method for Dynamic Neural Radiance Fields

Youtian Lin

Nanjing University
Harbin Institute of Technology
linyoutian.loyot@gmail.com

Abstract

Rendering photorealistic dynamic scenes has been a focus of recent research, with applications in virtual and augmented reality. While the Neural Radiance Field (NeRF) has shown remarkable rendering quality for static scenes, achieving real-time rendering of dynamic scenes remains challenging due to expansive computation for the time dimension. The incorporation of explicit-based methods, specifically voxel grids, has been proposed to accelerate the training and rendering of neural radiance fields with hybrid representation. However, employing a hybrid representation for dynamic scenes results in overfitting due to fast convergence, which can result in artifacts (e.g., floaters, noisy geometric) on novel views. To address this, we propose a compact and efficient method for dynamic neural radiance fields, namely Ced-NeRF which only requires a small number of additional parameters to construct a hybrid representation of dynamic NeRF. Evaluation of dynamic scene datasets shows that our Ced-NeRF achieves fast rendering speeds while maintaining high-quality rendering results. Our method outperforms the current state-of-the-art methods in terms of quality, training and rendering speed.

1 Introduction

Rendering photorealistic dynamic scenes presents a significant challenge, yet offers a broad range of applications, including virtual and augmented reality. Recently, there have been significant advances in novel view synthesis using Neural Radiance Fields (NeRF), leading to remarkable rendering quality for static scenes (Mildenhall et al. 2020; Zhang et al. 2020; Liu et al. 2020; Barron et al. 2021; Neff et al. 2021). There are numerous attempts to explore the feasibility of NeRF rendering for dynamic scenes and devise alternative ways of modeling the time dimension (Park et al. 2021a; Pumarola et al. 2021; Li et al. 2021; Park et al. 2021b). For example, D-NeRF (Pumarola et al. 2021) uses a canonical space to model the motion in the time dimension. However, achieving real-time rendering of dynamic scenes remains an elusive goal. This challenge stems not only from the inherent slowness of the original NeRF but also from the increased computational time required to model the time dimension of dynamic view synthesis, resulting in slow training and rendering speeds.

Figure 1: Our approach leverages a compact auxiliary network to accurately predict scene deformations, enabling the grid-based neural radiance field to synthesize novel views of dynamic scenes. Our model achieves impressive state-of-the-art performance, while also ensuring a rapid training speed. We demonstrate our method's rendering results on three datasets: D-NeRF (Pumarola et al. 2021), HyperNeRF (Park et al. 2021b), and DyNeRF (Li et al. 2022). ArXiv version with supplementary materials is available at https://github.com/Linyou/Ced-NeRF

Previous studies have demonstrated that incorporating explicit-based methods, particularly using voxel grids to model the local features of a scene, can significantly accelerate the training and rendering of NeRF (Sara Fridovich-Keil and Alex Yu et al. 2022; Sun, Sun, and Chen 2022a; Müller et al. 2022; Chen et al. 2022). These methods also referred to as hybrid representations (Xie et al. 2022), combine both explicit and implicit representations, resulting in faster convergence and rendering speeds for NeRF. Therefore, it is reasonable to extend the use of hybrid representation to combine with the previous dynamic NeRF methods (Pumarola

et al. 2021) to accelerate dynamic scene synthesis. However, as been discovered in DeVRF (Liu et al. 2022), the combination of hybrid representation and canonical space tends to yield overfitting results, which will produce artifacts (e.g., floaters, noisy geometric) on novel views.

Furthermore, the overfitting issue is rooted in the locality of the hybrid representation, which leads to rapid convergence in the early stages of training before obtaining an accurate motion prediction from the deformation network. This inaccuracy, in turn, introduces noise to the hybrid representation, making it difficult to correct in later stages of training. Moreover, employing a high-resolution hybrid (grid) representation, such as Multi-resolution hash tables (Müller et al. 2022), exacerbates this overfitting issue. Another approach is to use a low-res grid as the hybrid representation (Fang et al. 2022) or employ additional networks to model the dynamic and static parts of the scene (Song et al. 2022) to prevent the overfitting issue. However, these approaches lead to relatively slower convergence speed compared to high-res grid.

We present a series of strategies to mitigate overfitting in the hybrid representation, while simultaneously ensuring fast rendering speeds. To achieve this goal, we employ a grid-based hybrid representation to construct Ced-NeRF, a compact and efficient dynamic NeRF that can render scenes quickly, with few minutes of training time. Specifically, we introduce a neural latent regularization during training, which effectively prevents overfitting by regularizing the hybrid representation feature, without compromising rendering speed. Additionally, we propose a hierarchical motion prediction method to help the network learn accurate motion for the hybrid representation. Recent studies (Fang et al. 2022; Song et al. 2022) have demonstrated that incorporating a time feature as an additional input to the radiance field can significantly improve the modeling of illumination changes. However, we observed that this additional input also leads to overfitting. Hence, we propose a novel time-integration method that provides temporal information to the radiance field, improving its ability to model illumination changes while reducing motion overfitting from time-variants. In summary, our contributions are as follows:

- The introduction of a dynamic NeRF that enables fast training and rendering speeds and high-quality rendering with a rather simple and efficient framework.
- A neural regularization loss to regularize the hybrid representation during training, effectively preventing overfitting.
- A hierarchical motion prediction that assists the network in learning a more accurate motion for the hybrid representation.
- A novel time-integration method that provides the network with temporal information to better model changing illumination.

We evaluate our approach on dynamic scenes captured with a monocular camera as well as in a multi-camera setting. Our results demonstrate that our approach achieves fast training while maintaining high-quality rendering results compared to state-of-the-art methods.

2 Related Works

2.1 Dynamic Neural Radiance Field

The challenge of rendering dynamic scenes using NeRF has garnered significant attention in recent years. One intuitive solution is incorporating time as an additional input to NeRF for scene reconstruction (Li et al. 2021; Xian et al. 2021; Gao et al. 2021; Du et al. 2021; Lombardi et al. 2019). Nonetheless, due to the absence of scene structure prior knowledge, reconstruction results are often unsatisfactory. To mitigate this issue, some methods (Pumarola et al. 2021; Park et al. 2021a) first reconstruct a canonical space before establishing point motion from the canonical template. HyperNeRF (Park et al. 2021b) inputs higher-dimensional coordinates and enables deformations to capture object topology changes. DyNeRF (Li et al. 2022) utilizes time-conditioned neural radiance fields for dynamic scenes representation. However, these methods are based on the original NeRF pipeline, which lacks efficiency in training and rendering. Our proposed method addresses this by developing a dynamic NeRF pipeline on a hybrid representation for expedited training and rendering.

2.2 Hybrid Representation for Dynamic Scene

Various methods have been proposed to accelerate NeRF training and rendering through more efficient strategies (Liu et al. 2020; Fang et al. 2021; Lindell, Martel, and Wetzstein 2021; Piala and Clark 2021; Garbin et al. 2021; Hedman et al. 2021; Wu et al. 2022; Kurz et al. 2022). Several approaches also suggest combining explicit structures with neural implicit functions to create a hybrid representation (Müller et al. 2022; Sun, Sun, and Chen 2022b,a; Chen et al. 2022). These methods serve as robust baselines for accelerating dynamic NeRF while reducing training and inference times. A PlenOctrees extension (Yu et al. 2021) has been proposed to accelerate dynamic NeRF (Wang et al. 2022) by employing Fourier transform for dynamic scene representation and PlenOctrees for rendering acceleration. Tineuvox (Fang et al. 2022) employs the sparse voxel grid (Sun, Sun, and Chen 2022a) for dynamic scene representation, achieving a favorable quality-speed trade-off through proper computation allocation between explicit and implicit representations.

To further improve dynamic NeRF training and rendering speed while reducing memory overhead, several methods have been proposed. One such approach, akin to TensorRF (Chen et al. 2022), uses multiple planes as explicit representations for direct dynamic scene modeling. Methods such as K-Planes (Fridovich-Keil et al. 2023), Tensor4D (Shao et al. 2022), and HexPlane (Cao and Johnson 2023) adopt this strategy. Another approach (Song et al. 2022) develops a general streaming representation for grid-based (Müller et al. 2022) and plane-based methods, utilizing distinct models to separate static and dynamic scene components. However, this results in slower rendering times. HyperReel (Attal et al. 2023) proposes a flexible sampling network combined with two planes for dynamic scene representation.

In contrast, we avoid using low-resolution planes (Cao

and Johnson 2023) or sparse voxel grids (Fang et al. 2022) for dynamic scene reconstruction as they lack the fast convergence of high-resolution grids (Müller et al. 2022). We also eschew separated fields or models for capturing dynamic and static scene components due to the increased training difficulty and slower rendering times. Instead, we focus on employing a high-resolution grid (Müller et al. 2022) with canonical space modeling for rapid convergence (in 4 minutes)

3 Preliminary

Neural Radiance Field The problem of synthesizing 3D scenes from 2D images can be formulated as a function approximation task, where the goal is to learn a radiance function Φ_r that maps a 3D spatial location $\mathbf{x} = (x, y, z)$ and viewing direction $\mathbf{d} = (\theta, \phi)$ to density value σ and color value $\mathbf{c}$ at that location and direction:

$$\mathbf{c}, \sigma = \Phi_r(x, y, z, \theta, \phi). \tag{1}$$

To predict the radiance from a novel viewpoint, the neural radiance field is evaluated by integrating it along a ray passing through the scene. Specifically, given a camera ray $\mathbf{r} = \mathbf{o} + s\mathbf{d}$, a series points $\mathbf{r}_i = \mathbf{o} + s_i\mathbf{d}$ can be sampled along the ray, and the radiance function is evaluated at each point. The final predicted radiance value $C(\mathbf{r})$ is then computed by performing the classical volume rendering (Kajiya and Von Herzen 1984):

$$\bar{C}(\mathbf{r}) = \sum_{n=1}^{N} T_i(1 - e^{-\sigma_i\delta_i})\mathbf{c}_i, \quad T_i = \prod_{j=1}^{i-1} e^{-\sigma_j\delta_j}, \tag{2}$$

where δ_i indicates the distance between s_i and s_{i+1}, and N is the number of samples along the ray. To predict the color and density of the sample points, a neural network can be used as Φ_r, which can be trained by minimize the following reconstruction loss:

$$\mathcal{L}_{rec} = \sum_{\mathbf{r}\in\mathcal{R}} \left\|\bar{C}(\mathbf{r}) - C(\mathbf{r})\right\|_2^2, \tag{3}$$

where $\mathcal{R}$ is the set of all camera rays in the training set, and $C(\mathbf{r})$ is the groundtruth color of the ray pixel.

Position Encoding Utilizing the position $\mathbf{x}$ and viewing direction $\mathbf{d}$ as input to Φ_r can result in blurred reconstruction due to its inherent low frequency (Mildenhall et al. 2020). To counteract this, a positional encoding technique is employed to capture position information in a higher frequency representation, as follows:

$$\begin{aligned}\gamma(p) =& (\sin(2^0 p), \cos(2^0 p), \\ & \ldots, \sin(2^{L-1} p), \cos(2^{L-1} p)),\end{aligned} \tag{4}$$

where the hyperparameter L denotes the highest frequency that can be utilized in the encoding process and p is the input scalar of the encoding.

In contrast, grid-based NeRFs (Yu et al. 2021; Sara Fridovich-Keil and Alex Yu et al. 2022; Müller et al. 2022) employ an explicit approach to encode coordinates. As illustrated in Figure. 2, grid-based representations store local features in a grid cell and evaluate a point's feature by first determining the closest grid cell and then interpolating between features from adjacent cells. This process is formulated as:

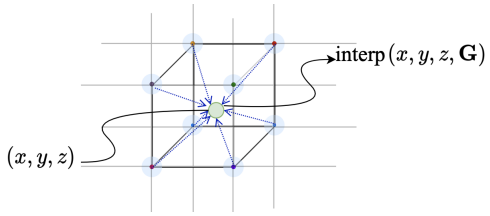


Figure 2: A grid representation performing interpolation on a given coordinate. The grid cells are represented by the color of the center of the voxel vertexes.

$$\gamma_{grid}(\mathbf{x}) = \text{interp}(x, y, z, \mathbf{G}), \tag{5}$$

where $\mathbf{G}$ is the set of grid cells closest to the coordinate (x, y, z). For instance, Instant-NGP (Müller et al. 2022) selects eight grid cells to perform linear interpolation.

In our method, we utilize both frequency encoding, as in Eq. (4), and grid encoding, as in Eq. (5). Frequency encoding is used to encode position and view direction, while grid encoding is used to encode position alone.

Modeling Time in Canonical Space As stated in the introduction, the NeRF method is typically used to model static scenes. However, it is possible to extend NeRF to model dynamic scenes by using a deformation network Φ_d to map a 3-dimensional coordinate $\mathbf{x}$ and a timestamp t to a new 3-dimensional coordinate $\mathbf{x}'$, as done in D-NeRF (Pumarola et al. 2021).

To achieve this, we use the encoding function $\gamma(\cdot)$ described in Eq. (4) to encode both the spatial coordinates and the timestamp. The deformation network Φ_d then maps the encoded coordinates and timestamp to a new coordinate $\mathbf{x}'$, which is used as input to the NeRF model. The formulation for this process is as follows:

$$\mathbf{x}' = \Phi_d(\gamma(\mathbf{x}), \gamma(t)) + \mathbf{x}. \tag{6}$$

We adopt this approach to directly model the motion of the scene, which enables us to construct a compact and efficient NeRF model for dynamic scenes.

4 Method

4.1 Hierarchical Motion Prediction

As depicted in Figure. 3, the conventional approach to motion modeling (Pumarola et al. 2021) involves using a deformation network to predict the canonical coordinate of the current point, as expressed in Eq. (6). The resulting coordinate is then employed to query grid features as encoding to NeRF. However, due to the possible inaccuracies of the deformation network, motion prediction may be imprecise, leading to the production of inaccurate grid features.

To mitigate this issue, we propose a hierarchical motion prediction approach that enables the deformation network to better decouple coarse and fine-grained motion. As shown

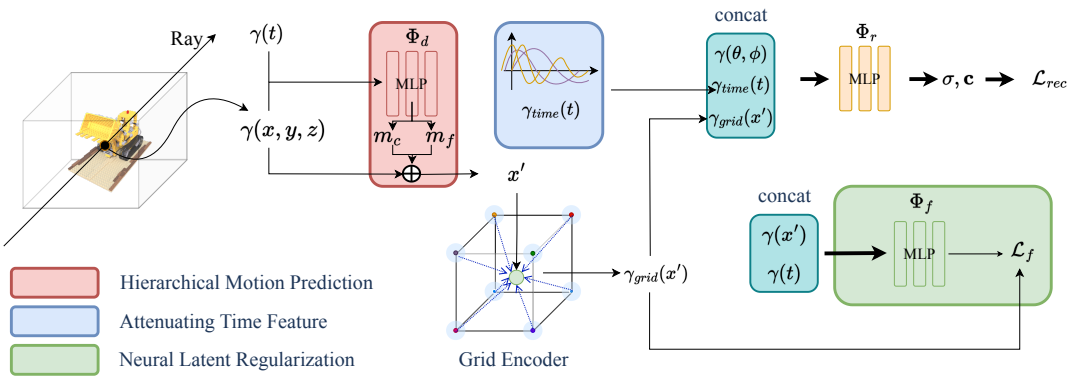


Figure 3: The overall framework of our method. We additionally use two small MLP upon the radiance field, the deformation network Φ_d for predicting the deformation coordinate, and the latent regulator Φ_f for regularizing the grid representation. The Φ_f only use in the training stage and will not affect the rendering speed of the radiance field.

in Figure. 4, the motion prediction process is divided into two stages. The first stage predicts a coarse motion m_c that corresponds to the motion between grid cells. The second stage predicts a fine-grained motion m_f that only operates within the grid cells. This design allows the new canonical space modeling to be formulated as:

$$\begin{aligned} \Delta\mathbf{x} &= \alpha \cdot \tanh(\mathbf{m}_f) + \alpha\mathbf{m}_c, \\ \mathbf{x}' &= \Delta\mathbf{x} + \mathbf{x}, \end{aligned} \tag{7}$$

where

$$\mathbf{m}_c, \mathbf{m}_f = \Phi_d(\gamma(\mathbf{x}), \gamma(t)). \tag{8}$$

To ensure that the deformation network predicts the expected motion for both coarse and fine-grained features, we employ an activation function $\tanh(\cdot)$ that maps the input to the range [-1, 1], and a step size α to control the magnitude of motion.

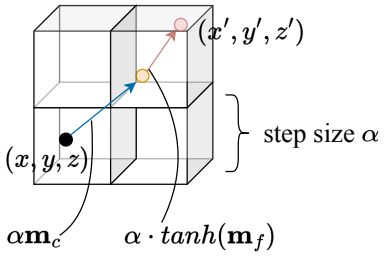
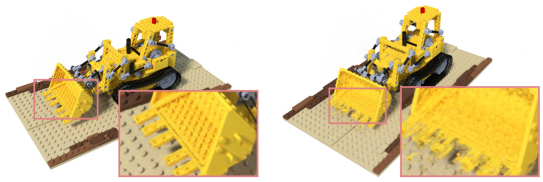


Figure 4: Illustration of hierarchical motion prediction in four grid cells. The step size α is set to be close to the size of a single grid cell. The term $\alpha\mathbf{m}_c$ allows the point to move between cells, while the term $\alpha \cdot tanh(\mathbf{m}_f)$ restricts the point's movement to within each cell.

4.2 Neural Latent Regularization

After obtaining the deformation coordinates $\mathbf{x}'$, we apply Eq. (5) to retrieve the corresponding encoding of the coordinates from the grid representation, which serves as input to the radiance field. However, the deformation network is trained end-to-end with NeRF, and the early predictions of the deformation network may be inaccurate, leading to the generation of imprecise grid features and potentially querying the wrong grid cell. Furthermore, the local features of the grid representation are challenging to modify in the later stages of training, as the parameters have already converged to a sub-optimal state. As shown in Figure. 7b, inaccurate or noisy grid features can cause overfitting of the radiance field, resulting in the failure to generate novel views.



(a) training view (b) test view

Figure 5: The overfitting of the radiance field. The training view image has fine detail, while the test view image generates noisy geometric.

To address this issue, we propose a novel neural regularization loss that only regulates the grid feature during training to prevent overfitting, without introducing additional computation during inference or compromising rendering speed. As depicted in Figure. 3, we accomplish this by first using a small neural network Φ_f to reconstruct the grid feature using the canonical coordinate $\mathbf{x}'$ and timestamp t as inputs, at every training step. Next, we compute the loss between the feature predicted by Φ_f and the encoding feature of the grid representation $\gamma_{grid}(\mathbf{x}')$, and optimize the network parameters to minimize the loss $\mathcal{L}_f$, which is formulated as:

$$\mathcal{L}_f = \left\| \Phi_f(\gamma(\mathbf{x}'), \gamma(t)) - \gamma_{grid}(\mathbf{x}') \right\|, \tag{9}$$

Importantly, the grid representation parameters that generate the grid feature are also optimized for this loss. This optimization encourages Φ_f to approximate the grid representation in both the spatial and time dimension (i.e., coordi-

nate and timestamp) and across all training steps, effectively approximating a "mean" representation within the training process. Additionally, it encourages the grid representation to lean towards the approximate feature that Φ_f predicts, which penalizes the grid feature to change too fast. Consequently, Φ_f serves as a global regularizer that regulates the grid representation throughout the entire training process. It is worth noting that since the grid representation only optimizes a few local grid cells for each training step, the overall scene representation will not be overwhelmed by this optimization.

4.3 Attenuating Time Feature

To enhance the reconstruction of dynamic scenes, we propose incorporating the time feature along with the grid feature as input to the radiance field, as depicted in Figure. 3. Recent studies (Fang et al. 2022; Song et al. 2022) have shown that the time feature can effectively explain illumination changes of the scene for both static and deforming parts.

However, in some settings such as monocular cameras, the use of the time feature can result in noisy reconstructions of the deforming part of the scene. This is because in certain cases, as shown in Figure. 5, the frequency of the time feature can be sufficient to reconstruct the deforming part of the scene in the training set. This leads the radiance field to rely solely on the time feature instead of using the coordinate. Consequently, the deformation network fails to predict the accurate deformation coordinate, resulting in noisy reconstructions in novel views (test set). This issue is similar to the overfitting problem investigated in NeRF++ (Zhang et al. 2020).

To address this issue, we introduce a time feature attenuation method that reduces the frequency of the time feature for the deforming part of the scene. We achieve this by using the norm of the deformation distance to attenuate the time feature, as shown in Eq. (10), where $\gamma_{time}(t)$ is the attenuated time feature, $\gamma(t)$ is the original time feature, Δx is the norm of the deformation distance, and λ is a hyper-parameter that controls the attenuation strength.

$$\gamma_{time}(t) = \gamma(t) e^{-\lambda 2^{\mathbf{L}(\gamma(t))} \Delta x}. \tag{10}$$

When the norm of the deformation distance is large, the frequency of the time feature is attenuated, and when the deformation is close to zero, the full frequency of the time feature is used. This method ensures that the radiance field guides the deformation network to learn the correct deformation rather than relying solely on the time feature to explain the scene, resulting in improved reconstructions.

4.4 Optimization

Gradient. During optimization, we permit the loss gradient $\mathcal{L}_f$ from Sec.4.2 to propagate back to the deformation network. This promotes the network to estimate more precise coordinates in the initial training phases. In contrast, we halt the gradient flow from the deformation distance norm in Eq.(10) to the deformation network, preventing the network from maximizing the norm to increase time feature frequency.

Objective. We follow the standard NeRF pipeline, as detailed in Sec. 3, for the rendering process and reconstruction loss, incorporating an additional grid feature regularization loss. The comprehensive training objective loss of our method is expressed as:

$$\mathcal{L} = \mathcal{L}_{rec} + \xi \mathcal{L}_f, \tag{11}$$

where ξ is a hyper-parameter that balances the reconstruction loss and the grid feature regularization loss.

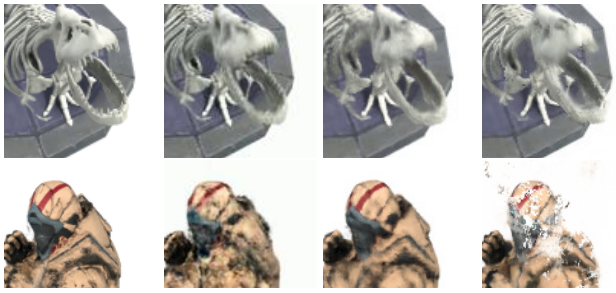
5 Experiments

5.1 Implementation Details

Our framework is built on top of Nerfacc (Li, Tancik, and Kanazawa 2022), and is implemented in PyTorch (Paszke et al. 2019). We adopt Instant-NGP (Müller et al. 2022) as our backbone, which uses a multi-resolution hash table as the grid representation. We use the same radiance field setting as Instant-NGP. As described in Sec. 3, our method only requires an additional deformation network to predict the deformation field, which is implemented with a 3-layer MLP with a width of 64. We employ only a 1-layer MLP with a width of 64 for our neural latent regularization, as described in Sec. 4.2. We train our model using the Adam (Kingma and Ba 2015) optimizer with a learning rate of 0.01, and decay the learning rate to 0.0003 every 1k steps. We train the model for 20K steps in synthetic scenes and 40K steps in real-world scenes. For better performance, we set the step size of the deformation, α, to 0.0001, the attenuation strength, λ, to 60, and the loss hyper-parameter, ξ, to 0.001 for synthetic scenes and 1 for real-world scenes. For a fair comparison, we conduct all experiments on a single RTX 3090 GPU.

5.2 Comparison with State-of-the-Art Methods

In this section, we present a comprehensive evaluation of our proposed method by presenting both quantitative and qualitative comparisons against state-of-the-art approaches, which demonstrate the performance and efficiency of our method. The comparison is conducted on two monocular datasets and one multi-view dataset, which included a wide range of challenges for non-rigid reconstruction.



Ours(20K) K-Planes TiNeuVox D-NeRF

Figure 6: Qualitative results on D-NeRF dataset. The visual comparison between our method, K-Planes (Fridovich-Keil et al. 2023), TiNeuVox (Fang et al. 2022) and D-NeRF (Pumarola et al. 2021).

Method	Train Time↓	Render Time↓ (s/img)	Broom PSNR↑	3D Printer PSNR↑	Chicken PSNR↑	Peel Banana PSNR↑	Mean PSNR↑	Mean SSIM↑
NeRF	∼ hours	∼ 75	19.9	20.7	19.9	20.0	20.1	0.745
Nerfies	∼ hours	∼ 90	19.2	20.6	26.7	22.4	22.2	0.803
HyperNeRF	32 hours	∼ 90	19.3	20.0	26.9	23.3	22.4	0.814
NeRFPlayer	∼ hours	4.80	21.7	22.9	26.3	24.0	23.7	0.803
TiNeuVox	30 min	-	21.5	22.8	28.3	24.4	24.3	0.837
Ours (20K)	9 min	0.16	21.6	23.1	28.4	24.5	24.4	0.823

Table 1: Per-scene quantitative comparisons on HyperNeRF dataset.

Method	Train Time	PSNR↑	SSIM ↑	LPIPS↓
DNeRF	20 hours	30.43	0.95	0.070
TiNeuVox	30 min	32.67	0.97	0.041
K-Planes	52 min	30.84	0.96	-
TIDNeRF	8 min	29.84	0.96	0.062
Ours (20K)	4 min	34.21	0.99	0.037

Table 2: Quantitative results on D-NeRF dataset. TIDNeRF (Park et al. 2023) is a recently proposed method that uses a similar framework as our method.

Synthetic Scenes. We first evaluate our method on the D-NeRF dataset (Pumarola et al. 2021), which comprises synthetic monocular scenes of 360-degree rotation for objects. The dataset consists of nine synthesis scenes, each containing 50-200 frames in the training set and 20 frames in the test set. We report the PSNR, SSIM (Wang et al. 2004), and LPIPS (Zhang et al. 2018) scores in Table. 2. To establish a performance baseline, we compare our approach with D-NeRF, an implicit NeRF method, as well as recently proposed hybrid NeRF methods including K-Planes (Fridovich-Keil et al. 2023), TiNeuVox (Fang et al. 2022), and TIDNeRF (Park et al. 2023). Our results indicate that our method achieves superior performance in terms of PSNR, SSIM, and LPIPS metrics while requiring a training time that is more than two times shorter than previous methods. Specifically, our method achieves a PSNR of 34.21, an SSIM of 0.987, and an LPIPS of 0.037, outperforming all other methods in the comparison. Moreover, as illustrated in Figure. 6, our method exhibits robustness to scene deformations, producing clear and detailed results. This performance advantage is particularly evident when comparing our method to TIDNeRF (Park et al. 2023), a recent dynamic reconstruction method, which, similar to our method, also uses Instant-NGP as its backbone. Our method not only surpasses TIDNeRF in terms of reconstruction quality, as evidenced by the PSNR, SSIM, and LPIPS scores, but it also demonstrates greater efficiency with a training time that is half as long.

Real Monocular Scenes. We then present a comprehensive analysis comparing our method with state-of-the-art techniques on the HyperNeRF dataset (Park et al. 2021b), which is captured using a monocular camera and includes real non-rigidly deforming scenes. This dataset poses challenges due to large deformations, complex lighting conditions, and thin object structures. To ensure a fair comparison, we downsampled images to 540 $\times$ 960 in our experiments and followed the training and validation camera split provided by (Park et al. 2021b). We conducted experiments on four "vrig" scenes, as shown in Table. 1. Our method outperforms previous approaches such as HyperNeRF and Nerfies in terms of both PSNR and SSIM metrics. Moreover, our method demonstrates competitive performance with recent state-of-the-art methods, namely NeRFPlayer and TiNeuVox, on all four scenes. In terms of the average PSNR and SSIM, our method achieves 24.4 and 0.823, respectively, and outperforms TiNeuVox slightly, our method shows a considerable advantage in terms of computational efficiency. Notably, our method is more than 3$\times$ faster than all the other methods in terms of training speed, requiring only 9 minutes of training time. Additionally, our method exhibits impressive rendering performance, achieving a render time of just 0.16 seconds per image, which is significantly faster than other state-of-the-art methods, including NeRFPlayer, which takes 4.80 seconds per image.

Method	Train Time	PSNR ↑	SSIM ↑
LLFF	-	23.2	-
DyNeRF	1344 hours	29.6	0.961
NeRFPlayer	5.5 hours	30.7	-
K-Planes	1.8 hours	31.6	0.964
Ours (20K)	21 min	30.6	0.919

Table 3: Quantitative comparison on Plenoptic Video dataset (Li et al. 2022).

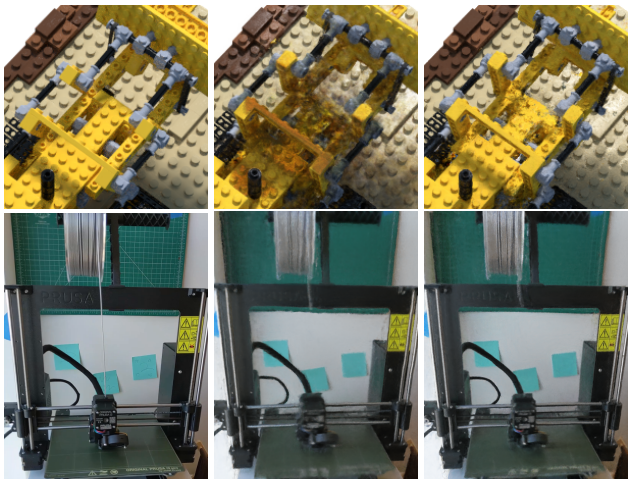
Multi-view Scenes. We further compare our method with state-of-the-art methods on the Plenoptic Video dataset (Li et al. 2022), which was captured using 21 cameras at a resolution of 2704 $\times$ 2028, with each camera recording a 10-second video. Six scenes from this dataset are publicly available. For a fair comparison, we downsampled the images to 1352 $\times$ 1014 in our experiments. As shown in Table. 3, our method outperforms previous methods such as LLFF and DyNeRF in terms of PSNR and achieves competitive performance with recent state-of-the-art methods, namely NeRFPlayer and K-Planes. Specifically, our method attains

a PSNR of 30.6 and an SSIM of 0.919, which are comparable to the performance metrics of NeRFPlayer and K-Planes. However, our method offers a significant improvement in training speed, with a time that is 5× shorter than that of K-Planes and 15× shorter than NeRFPlayer. While our method's SSIM is slightly lower than that of DyNeRF and K-Planes, it is essential to consider the balance between reconstruction quality and computational efficiency. Our method provides competitive reconstruction quality while requiring considerably less training time, making it a suitable option for applications with limited computational resources.

5.3 Ablation Study

To further validate the proposed components of our method, we conduct ablation studies on both the D-NeRF dataset and the HyperNeRF dataset.

Latent Regularization. We first evaluate the effectiveness of our proposed Neural Latent Regularization. As shown in Table. 4, our regularization method improves the model's performance by only 0.25 dB compared to the base model (An Instant-NGP with only a deformation network). However, as demonstrated in Figure. 7, the regularization significantly improves the quality of the results. Specifically, our regularization effectively prevents overfitting in the top row of Figure. 7b, as well as enhances the clarity of the 3D Printer (bottom row) scene.



(a) GT (b) w/o Latent-R (c) with Latent-R

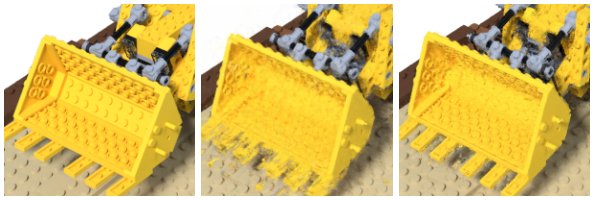
Figure 7: Visual results of Neural Latent Regularization (Latent-R). We show the result of with and without Neural Latent Regularization on both the Lego (top) and 3D Printer (bottom) scene.

Hierarchical Motion Prediction. Secondly, we evaluate the effectiveness of our proposed Hierarchical Motion Prediction. As shown in Table. 4, we observe that the Hierarchical Motion Prediction significantly improves the model's performance compared to other proposed components.

Latent Regula.	Hierarchical Pred.	Time Attenu.	PSNR	SSIM
✓	✓	✓	**34.2**	**0.987**
✓	✓	✗	34.0	0.987
✓	✗	✗	33.3	0.985
✗	✗	✗	33.0	0.984

Table 4: Ablation results on the test set of the DNeRF dataset. We show the PSNR and SSIM scores of our Neural Latent Regularization (Latent Regula.), Hierarchical Motion Prediction(Hierarchical Pred.) and Time Feature Attenuation(Time Attenu.).

Time Feature Attenuation. Finally, we evaluate the effectiveness of our proposed Time Feature Attenuation. As shown in Table. 4, similar to the Neural Latent Regularization, we observe that the Time Feature Attenuation only marginally improves the model's performance. Nevertheless, we show that it effectively prevents the overfitting issue as demonstrated in Figure. 8. When employing an additional time feature to the radiance field can enhance the illumination change in the dynamic scene. However, the model produces the overfitting result which tends to generate a large amount of noise in geometry which is the deformation of the scene. And as shown in Figure. 8c, the Time Feature Attenuation effectively mitigates the overfitting issue.



(a) GT (b) w/o Time-A (c) with Time-A

Figure 8: Visual results of Time Feature Attenuation (Time-A). We show the result of with and without Time Feature Attenuation on the Lego scene.

6 Conclusion

In this paper, we have presented Ced-NeRF, an efficient and compact framework for dynamic NeRF. Our contributions include a neural latent regularization function and a time feature attenuation strategy, which effectively address overfitting issues commonly encountered in high-resolution grid-based representations for dynamic NeRF. Additionally, we have proposed a hierarchical motion prediction approach that significantly improves the accuracy of scene deformation in grid representations. Our experimental results demonstrate that our approach achieves rapid convergence and efficiently renders dynamic NeRF. Ablation analyses further validate the effectiveness of the proposed method. Overall, the Ced-NeRF framework shows great potential for advancing the field of dynamic NeRF by enabling faster and more accurate rendering of dynamic scenes.

References

Attal, B.; Huang, J.-B.; Richardt, C.; Zollhoefer, M.; Kopf, J.; O'Toole, M.; and Kim, C. 2023. HyperReel: High-Fidelity 6-DoF Video with Ray-Conditioned Sampling. *arXiv preprint arXiv:2301.02238*.

Barron, J. T.; Mildenhall, B.; Tancik, M.; Hedman, P.; Martin-Brualla, R.; and Srinivasan, P. P. 2021. Mip-NeRF: A Multiscale Representation for Anti-Aliasing Neural Radiance Fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 5855–5864.

Cao, A.; and Johnson, J. 2023. HexPlane: a fast representation for dynamic scenes. *arXiv preprint arXiv:2301.09632*.

Chen, A.; Xu, Z.; Geiger, A.; Yu, J.; and Su, H. 2022. TensoRF: Tensorial Radiance Fields. In *Proceedings of the European Conference on Computer Vision*.

Du, Y.; Zhang, Y.; Yu, H.-X.; Tenenbaum, J. B.; and Wu, J. 2021. Neural radiance flow for 4d view synthesis and video processing. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 14304–14314. IEEE Computer Society.

Fang, J.; Xie, L.; Wang, X.; Zhang, X.; Liu, W.; and Tian, Q. 2021. NeuSample: Neural Sample Field for Efficient View Synthesis. *arXiv:2111.15552*.

Fang, J.; Yi, T.; Wang, X.; Xie, L.; Zhang, X.; Liu, W.; Nießner, M.; and Tian, Q. 2022. Fast Dynamic Radiance Fields with Time-Aware Neural Voxels. *arXiv preprint arXiv:2205.15285*.

Fridovich-Keil, S.; Meanti, G.; Warburg, F.; Recht, B.; and Kanazawa, A. 2023. K-planes: Explicit radiance fields in space, time, and appearance. *arXiv preprint arXiv:2301.10241*.

Gao, C.; Saraf, A.; Kopf, J.; and Huang, J.-B. 2021. Dynamic view synthesis from dynamic monocular video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 5712–5721.

Garbin, S. J.; Kowalski, M.; Johnson, M.; Shotton, J.; and Valentin, J. 2021. FastNeRF: High-Fidelity Neural Rendering at 200FPS. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 14346–14355.

Hedman, P.; Srinivasan, P. P.; Mildenhall, B.; Barron, J. T.; and Debevec, P. 2021. Baking Neural Radiance Fields for Real-Time View Synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 5875–5884.

Kajiya, J. T.; and Von Herzen, B. P. 1984. Ray tracing volume densities. *ACM SIGGRAPH computer graphics*, 18(3): 165–174.

Kingma, D. P.; and Ba, J. 2015. Adam: A Method for Stochastic Optimization. In Bengio, Y.; and LeCun, Y., eds., *International Conference on Learning Representations*.

Kurz, A.; Neff, T.; Lv, Z.; Zollhofer, M.; and Steinberger, M. 2022. AdaNeRF: Adaptive Sampling for Real-time Rendering of Neural Radiance Fields. In *European Conference on Computer Vision*.

Li, R.; Tancik, M.; and Kanazawa, A. 2022. NerfAcc: A General NeRF Acceleration Toolbox. *arXiv preprint arXiv:2210.04847*.

Li, T.; Slavcheva, M.; Zollhöfer, M.; Green, S.; Lassner, C.; Kim, C.; Schmidt, T.; Lovegrove, S.; Goesele, M.; Newcombe, R.; and Lv, Z. 2022. Neural 3D Video Synthesis From Multi-View Video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 5521–5531.

Li, Z.; Niklaus, S.; Snavely, N.; and Wang, O. 2021. 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*.

Lindell, D. B.; Martel, J. N. P.; and Wetzstein, G. 2021. AutoInt: Automatic Integration for Fast Neural Volume Rendering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*.

Liu, J.-W.; Cao, Y.-P.; Mao, W.; Zhang, W.; Zhang, D. J.; Keppo, J.; Shan, Y.; Qie, X.; and Shou, M. Z. 2022. DeVRF: Fast Deformable Voxel Radiance Fields for Dynamic Scenes. *arXiv preprint arXiv:2205.15723*.

Liu, L.; Gu, J.; Lin, K. Z.; Chua, T.; and Theobalt, C. 2020. Neural Sparse Voxel Fields. In *Advances in Neural Information Processing Systems*.

Lombardi, S.; Simon, T.; Saragih, J.; Schwartz, G.; Lehrmann, A.; and Sheikh, Y. 2019. Neural volumes: learning dynamic renderable volumes from images. *ACM Transactions on Graphics*, 38(4): 1–14.

Mildenhall, B.; Srinivasan, P. P.; Tancik, M.; Barron, J. T.; Ramamoorthi, R.; and Ng, R. 2020. NeRF: Representing scenes as neural radiance fields for view synthesis. In *European Conference on Computer Vision*, 405–421. Springer.

Müller, T.; Evans, A.; Schied, C.; and Keller, A. 2022. Instant Neural Graphics Primitives with a Multiresolution Hash Encoding. *ACM Trans. Graph.*, 41(4): 102:1–102:15.

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

Park, K.; Sinha, U.; Barron, J. T.; Bouaziz, S.; Goldman, D. B.; Seitz, S. M.; and Martin-Brualla, R. 2021a. Nerfies: Deformable Neural Radiance Fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 5865–5874.

Park, K.; Sinha, U.; Hedman, P.; Barron, J. T.; Bouaziz, S.; Goldman, D. B.; Martin-Brualla, R.; and Seitz, S. M. 2021b. HyperNeRF: A Higher-Dimensional Representation for Topologically Varying Neural Radiance Fields. *ACM Trans. Graph.*, 40(6).

Park, S.; Son, M.; Jang, S.; Ahn, Y. C.; Kim, J.-Y.; and Kang, N. 2023. Temporal Interpolation Is All You Need for Dynamic Neural Radiance Fields. *arXiv preprint arXiv:2302.09311*.

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

Piala, M.; and Clark, R. 2021. TermiNeRF: Ray Termination Prediction for Efficient Neural Rendering. *International Conference on 3D Vision*, 1106–1114.

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

Sara Fridovich-Keil and Alex Yu; Tancik, M.; Chen, Q.; Recht, B.; and Kanazawa, A. 2022. Plenoxels: Radiance Fields without Neural Networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Shao, R.; Zheng, Z.; Tu, H.; Liu, B.; Zhang, H.; and Liu, Y. 2022. Tensor4D: Efficient Neural 4D Decomposition for High-fidelity Dynamic Reconstruction and Rendering. *arXiv preprint arXiv:2211.11610*.

Song, L.; Chen, A.; Li, Z.; Chen, Z.; Chen, L.; Yuan, J.; Xu, Y.; and Geiger, A. 2022. NeRFPlayer: A Streamable Dynamic Scene Representation with Decomposed Neural Radiance Fields. *arXiv preprint arXiv:2210.15947*.

Sun, C.; Sun, M.; and Chen, H.-T. 2022a. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5459–5469.

Sun, C.; Sun, M.; and Chen, H.-T. 2022b. Improved Direct Voxel Grid Optimization for Radiance Fields Reconstruction. *arXiv preprint arXiv:2206.05085*.

Wang, L.; Zhang, J.; Liu, X.; Zhao, F.; Zhang, Y.; Zhang, Y.; Wu, M.; Yu, J.; and Xu, L. 2022. Fourier PlenOctrees for Dynamic Radiance Field Rendering in Real-time. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 13524–13534.

Wang, Z.; Bovik, A. C.; Sheikh, H. R.; and Simoncelli, E. P. 2004. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4): 600–612.

Wu, L.; Lee, J. Y.; Bhattad, A.; Wang, Y.-X.; and Forsyth, D. 2022. 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*, 16200–16209.

Xian, W.; Huang, J.-B.; Kopf, J.; and Kim, C. 2021. Spacetime Neural Irradiance Fields for Free-Viewpoint Video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 9421–9431.

Xie, Y.; Takikawa, T.; Saito, S.; Litany, O.; Yan, S.; Khan, N.; Tombari, F.; Tompkin, J.; Sitzmann, V.; and Sridhar, S. 2022. Neural fields in visual computing and beyond. In *Computer Graphics Forum*, volume 41, 641–676. Wiley Online Library.

Yu, A.; Li, R.; Tancik, M.; Li, H.; Ng, R.; and Kanazawa, A. 2021. PlenOctrees for Real-Time Rendering of Neural Radiance Fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 5752–5761.

Zhang, K.; Riegler, G.; Snavely, N.; and Koltun, V. 2020. NeRF++: Analyzing and Improving Neural Radiance Fields. *arXiv:2010.07492*.

Zhang, R.; Isola, P.; Efros, A. A.; Shechtman, E.; and Wang, O. 2018. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 586–595.