

Detachable Novel Views Synthesis of Dynamic Scenes Using Distribution-Driven Neural Radiance Fields

Boyu Zhang^{1,2}, Wenbo Xu², Zheng Zhu², Guan Huang²

¹Southeast University, ²PhiGent Robotics

{wenbo.xu, guan.huang}@phigent.ai, byz@seu.edu.cn, zhengzhu@ieee.org

Abstract

Representing and synthesizing novel views in real-world dynamic scenes from casual monocular videos is a long-standing problem. Existing solutions typically approach dynamic scenes by applying geometry techniques or utilizing temporal information between several adjacent frames without considering the underlying background distribution in the entire scene or the transmittance over the ray dimension, limiting their performance on static and occlusion areas. Our approach **Distribution-Driven** neural radiance fields offers high-quality view synthesis and a 3D solution to **Detach** the background from the entire **Dynamic** scene, which is called **D⁴NeRF**. Specifically, it employs a neural representation to capture the scene distribution in the static background and a 6D-input NeRF to represent dynamic objects, respectively. Each ray sample is given an additional occlusion weight to indicate the transmittance lying in the static and dynamic components. We evaluate D⁴NeRF on public dynamic scenes and our urban driving scenes acquired from an autonomous-driving dataset. Extensive experiments demonstrate that our approach outperforms previous methods in rendering texture details and motion areas while also producing a clean static background. Our code will be released¹.

1. Introduction

Novel view synthesis [2,3,15] from dynamic scenes aims to generate photorealistic views at arbitrary viewpoints and any time step with given images from one or more cameras as input. It provides the possibility to generate free-view rendering using finite-view input and brings a life-like representation. This can have vast and varied applications, e.g. switching views in cinematic/games special effects [70], rendering visuals in AR/VR world [9, 39, 53], assisting camera imaging [46], and enabling interactive exploration in robot/autonomous-driving perception and navigation [35, 67].

¹<https://github.com/Luciferbobo/D4NeRF>

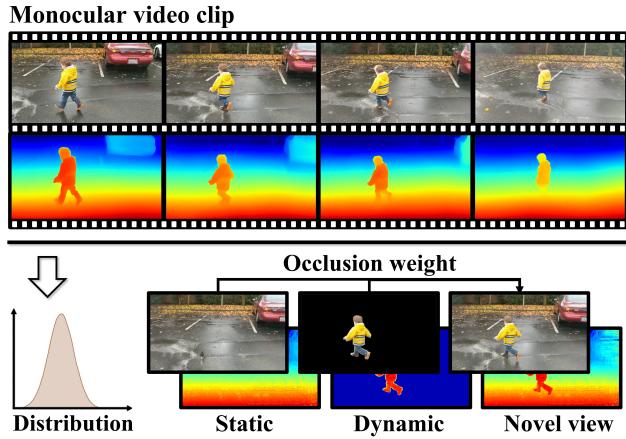


Figure 1. Our method takes a casual monocular video clip captured from real-world dynamic scenes as input and generates a static representation driven by the underlying background and a dynamic representation. The target novel view rendering can be obtained by blending them with occlusion weight.

In essence, novel view synthesis of dynamic scenes is an extension of representing static 3D scenes from discrete 2D images, which is highly ill-posed [29] since there are infinite solutions that can render the input video appropriately while only a single 2D image observation is available at each view. Compared to static scenes [21, 27, 32, 49, 65, 86], dynamic novel view synthesis is more challenging since approaches for dynamic scenes need to capture spatial-temporal information using a 6D plenoptic representation ($\mathbf{I}(t) = \Phi(\mathbf{x}, \mathbf{d}, t; \boldsymbol{\theta}) : \mathbb{R}^3 \times \mathbb{R}^2 \times \mathbb{R} \rightarrow \mathbb{R}^3$), and the occlusion between different objects might also be tricky.

Traditionally, novel view synthesis in real-world dynamic scenes [7, 14, 16, 36, 42, 51, 52, 82] requires images from multiple camera views to estimate the geometry and occlusion of the entire scene. However, capturing such multi-view inputs is laborious and expensive. Neural Radiance Fields (NeRF) [48] offers a new circumvention to this problem when just given monocular views. Rather than explicit shape representation, NeRF encodes

3D location and viewing angle to color and density using a neural network and generates arbitrary novel views through volume rendering. Recent NeRF-based methods [1, 20, 23, 30, 35, 39, 40, 55, 58, 67, 71, 76, 79] show promising results. Using one or more NeRFs, these methods can implicitly represent dynamic scenes for novel view synthesis and time interpolation.

Several methods [30, 35, 39, 67] use supervision networks (*e.g.* semantic segmentation) to produce dynamic view synthesis and can achieve editable results, but they are limited to specific domains. Other methods [20, 23, 40, 55, 71] focus on adjacent temporal information in a finite number of frames to capture time-dependent components, such as optical flow and depth, while overlooking the underlying distribution existing in the entire video clip. For scenes containing a single object, such as Lego and Fern [48], the 2D scene projection on the camera plane will alter drastically due to the switch of the view angle. Unlike these scenes, real-world locomotor scenes often contain a dynamic foreground and a static background. These static backgrounds, such as blocks, buildings, and other rigid areas, are far away from the camera lens. When photographing these scenes from various perspectives, the 2D projection on the camera plane will be only minimally disturbed by the change in viewpoint. It implies that the bulk of similar projection pixels in a real-world dynamic scene are shared by the static background across multiple frames. Assuming that identical regions are the observations for each frame and obey the scene distribution, then the camera projections from different angles can be obtained with tiny shifts based on this representation of distribution. Moreover, all previous approaches determine occlusion only from RGB space by merging rendering color and neglect transmittance weight existing on each ray sample, resulting in a sub-optimal performance in occlusion areas between the static backdrop and dynamic objects.

To tackle the aforementioned distribution representation problem and provide occlusion weight over the ray dimension, we introduce D⁴NeRF, a novel method that *captures the underlying distribution in the entire scene and adds transmittance to each ray*. It can generate high-quality novel views at arbitrary viewpoints and any interpolation time steps for view synthesis of real-world dynamic scenes, and applied to general domains. Specifically, we extend NeRF to a parallel structure, as shown in Fig. 1. A background pipeline presenting the underlying distribution and a 6D-input NeRF generating time-varying fields are used to express static and dynamic components, respectively. Then we weight transmittance matrices on their rendering corresponding to each ray to learn the occlusion relationship. To optimize the pipeline effectively, we use multiple regularization losses to drive training in different modules. We evaluate D⁴NeRF on NVIDIA dynamic scenes [82] and our

urban driving scenes obtained from an autonomous-driving dataset, Argoverse [78]. Our contributions can be summarized as follows:

- To the best of our knowledge, we are the first to exploit the underlying scene distribution and produce transmittance across each ray sample in order to depict dynamic real-world scenes.
- We present a novel attention-based structure and an occlusion weight neural representation that offers a 3D pattern for decoupling the clean static background from the entire scene.
- Substantial quantitative and qualitative experiments suggest that D⁴NeRF operates better than existing methods. Additionally, an urban driving scenes dataset is built for dynamic novel view synthesis. We will release this dataset for research purpose.

2. Related Work

Novel view synthesis. The fundamental idea behind novel view synthesis is to generate the geometry of scenes from a set of limited views. Intuitively, we need to create an explicit 3D representation of scene geometry, such as point clouds or meshes [8, 11, 17, 25, 64], and render novel views by transporting pixels among views via this representation. There are many works that concentrate on using various graphic techniques, such as light field [10, 24, 37, 47, 57, 63] and multi-plane based methods [13, 61, 66, 73, 87], to obtain 3D representation. In addition, learning-based methods [12, 44, 45, 54, 72] attempt to utilize neural networks to learn view interpolation to obtain implicit representation. Meanwhile, neural networks can provide multiple priors to improve producing 3D representation, such as extracting meshes features [26, 61] or estimating sparse/dense depth maps as prior [22, 50, 81]. More recently, Neural Radiance Fields (NeRF) [48] have attained photorealistic synthesis performance by rendering novel views through MLP-based radiance and opacity fields. NeRF brings new enlightenment to novel view synthesis, and many methods try applying it to various scenes. NeRFw [43] is suitable for novel view synthesis in outdoor scenes. Mip-NeRF [5] enhances NeRF for unbounded scenes. And many methods [60, 74, 80] extend NeRF to large-scale scenes. Nevertheless, these methods are only applicable to static scenes, as a 6D-input of dynamic scenes will lead to radiance ambiguity [84].

Video view synthesis for dynamic scenes. Since the promising performance of neural radiance field, many works [20, 23, 40, 55, 58, 71, 83] consider extending NeRF as a spatial-temporal representation to address view synthesis in dynamic scenes. Some of the previous meth-

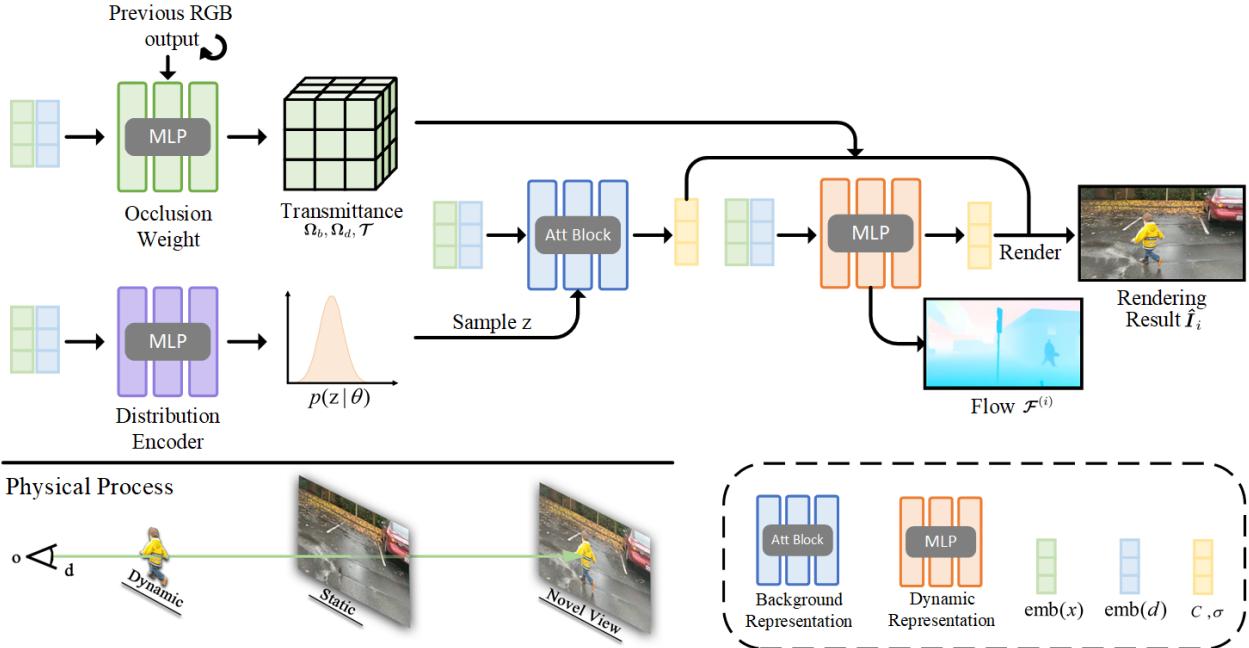


Figure 2. **Network pipeline of D^4 NeRF.** Our approach trains two neural representations jointly: a distribution-driven **background representation** and a flow-predicting **dynamic representation**, producing RGB color C and volume density σ . A **distribution encoder** is constructed to learn the implicit distribution from the background. And an **occlusion weight module** is created to generate transmittance. At i -th time instance, the final rendering result \hat{I}_i is made by mixing occlusion weight Ω and \mathcal{T} with the outputs of two networks, as described in the physical process (bottom).

ods [31, 38, 88] for dynamic view synthesis capture geometry and textures explicitly. Other previous learning-based models [4, 6, 28, 82] attempt to produce novel views using neural networks, while their performance is limited by the count of input views. Unlike these approaches, NeRF-based methods perform view synthesis for dynamic scenes with fewer input views and could enable time interpolation. D-NeRF [58] considers time as an additional input to handle single dynamic objects indoors. [20, 40] model the dynamic scene as a time-dependent continuous function with optical flow. Gao *et al.* [23] introduce regularization losses to encourage plausible reconstruction. Xian *et al.* [79] employ video depth estimation to supervise a space-time radiance field. Using a ray-blending network, Tretschk *et al.* [71] model the occlusion parts in dynamic scenes. TöRF [1] utilizes prior information from the time-of-flight camera to enhance reconstruction quality. Park *et al.* [55] optimize an additional continuous volumetric deformation field to capture dynamic humans. Li *et al.* [39] aim to represent scenes in multi-view videos effectively. These NeRF-based approaches take into account the use of temporal information between several successive frames, such as optical flow and depth, but disregard the global distribution over all frames. More recently, NeRF-based methods [30, 35] have provided a more accurate representation of dynamic scenes and achieved editable effects, but they are limited to specific domains.

3. Method

In this section, we introduce the proposed ***Distribution-Driven Neural Radiance Fields for Detachable Novel Views Synthesis of Dynamic Scenes***, D^4 NeRF. As shown in Fig. 2, D^4 NeRF consists of a background distribution driving module and a 6D-input NeRF. An occlusion weight module is designed to decouple the dynamic and static components. We first describe the background pipeline and the temporal-input NeRF in Sec. 3.2 and Sec. 3.3, respectively. Then the occlusion weight module will be discussed in Sec. 3.4. Lastly, a group of loss functions for optimizing the proposed network will be explained in Sec. 3.5.

3.1. Preliminaries

NeRF represents the radiance $\mathbf{c} = (r, g, b)$ and differential volume density σ at a 3D location $\mathbf{x} = (x, y, z)$ of a scene observed from a viewing direction $\mathbf{d} = (\theta, \phi)$ as a continuous multi-variate function using a neural network, $\Phi : (\mathbf{x}, \mathbf{d}) \rightarrow (\mathbf{c}, \sigma)$. To capture high-frequency signals [68] in inputs, NeRF uses a set of sine and cosine functions for increasing frequencies, $\gamma(\mathbf{x}) = (\mathbf{x}, \dots, \sin(2^k \pi \mathbf{x}), \cos(2^k \pi \mathbf{x}), \dots), k \in \{0, \dots, m-1\}$, where m is a hyper-parameter that controls the total number of frequency bands. The pixel color $\hat{\mathbf{C}}$ can be rendered by integrating the radiance modulated by the volume density along the camera ray $\mathbf{r}(s) = \mathbf{o} + s\mathbf{d}$, with the origin

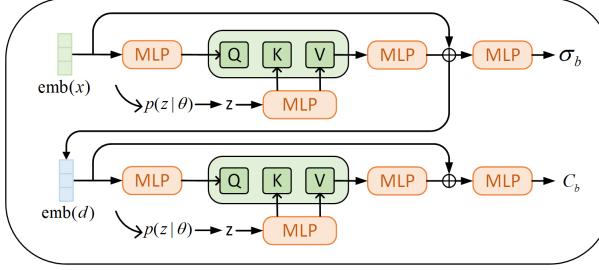


Figure 3. **Attention block.** To compute attention, we use a latent variable sampled from the background distribution as **Key** and **Value**, and embedded 3D position and direction as **Query**. \oplus denotes element-wise addition.

o and direction **d** defined by the specified camera pose and intrinsic parameters:

$$\hat{\mathbf{C}}(\mathbf{r}) = \int_{s_n}^{s_f} T(s) \sigma(\mathbf{r}(s)) \mathbf{c}(\mathbf{r}(s), \mathbf{d}) ds,$$

where $T(s) = \exp\left(-\int_{s_n}^s \sigma(\mathbf{r}(p)) dp\right).$ (1)

s_n and s_f denote the bounds of the volume depth range, $T(s)$ corresponds to the accumulated transparency along that ray. The loss is then the difference between the reconstructed color $\hat{\mathbf{C}}$ and the ground truth color \mathbf{C} corresponding to the pixel that ray \mathbf{r} originated from:

$$\mathcal{L} = \sum_{\mathbf{r}} \|\hat{\mathbf{C}}(\mathbf{r}) - \mathbf{C}(\mathbf{r})\|_2^2. \quad (2)$$

3.2. Distribution Representation

We intend to represent the static background in terms of distribution such that camera projections from different views can be queried by small shifts based on this distribution representation. Assuming that the distribution obeys $P(\mathbf{z} \in \mathbb{R}^D | \Theta)$, it can be formulated to:

$$\Phi_b : (\mathbf{x}, \mathbf{d}, \mathbf{z}) \rightarrow (\sigma_b, \mathbf{c}_b). \quad (3)$$

To obtain the implicit distribution from the static background, we construct a distribution encoder, as shown in Fig. 2. Then a latent variable is sampled from the output posterior $\mathbf{z} \sim P_\Theta(\mathbf{z})$. The sample is utilized to drive the rendering of the background component. Due to the resultant performance of attention mechanisms [19, 41, 75], we apply an attention-based structure to exploit the latent variable, as shown in Fig. 9.

Specifically, we apply attention to embedded inputs with querying sample z . By mapping embedded features and the latent distribution sample into a low-dimensional space, attention elements of 3D location \mathbf{Q}_l , \mathbf{K} , and \mathbf{V} can be derived (see supplemental material). \mathbf{F}_\uparrow denotes re-projecting inputs to the original-dimension space and extending elements along the ray dimension with interleaving repetition.

Then the attention feature \mathcal{C}_l of 3D location can be interpreted as:

$$\mathcal{C}_l = \mathbf{F}_\uparrow(\text{Att}(\mathbf{Q}_l)) + emb(\mathbf{x}),$$

$$\text{where } \text{Att}(\mathbf{Q}_l) = \text{softmax}\left(\frac{\mathbf{Q}_l \mathbf{K}^T}{\sqrt{d}}\right) \mathbf{V}. \quad (4)$$

Attention feature of view direction \mathcal{C}_d can be calculated in the same pattern as Eq. (15), with \mathcal{C}_l concatenated to the input. Then the volumetric density and RGB color will be generated by subsequent networks $\sigma_b = \text{MLP}(\mathcal{C}_l)$ and $\mathbf{c}_b = \text{MLP}(\mathcal{C}_d)$. Same to NeRF, the density part is solely determined by the position \mathbf{x} . The rendered RGB color $\hat{\mathbf{C}}_b(\mathbf{r})$ is denoted as:

$$\hat{\mathbf{C}}_b(\mathbf{r}) = \int_{s_n}^{s_f} T(s) \sigma_b(\mathbf{r}(s), \mathbf{z}) \mathbf{c}_b(\mathbf{r}(s), \mathbf{d}, \mathbf{z}) ds$$

where $T(s) = \exp\left(-\int_{s_n}^s \sigma_b(\mathbf{r}(p), \mathbf{z}) dp\right).$ (5)

3.3. Motion Representation

We employ a dynamic NeRF to describe the time-varying motion in real-world dynamic scenes. At i -th time instance, we have:

$$\Phi_d : (\mathbf{x}, \mathbf{d}, i) \rightarrow (\sigma_d, \mathbf{c}_d, \mathcal{F}^{(i)}) \quad (6)$$

The rendered color $\hat{\mathbf{C}}^i(\mathbf{r})$ of the pixel corresponding to i -th time step is an integral over the radiance weighted by accumulated opacity $T(s) = \exp\left(-\int_{s_n}^s \sigma_d(\mathbf{r}(p), i) dp\right)$:

$$\hat{\mathbf{C}}_d^{(i)}(\mathbf{r}) = \int_{s_n}^{s_f} T(s) \sigma_d(\mathbf{r}(s), i) \mathbf{c}_d(\mathbf{r}(s), \mathbf{d}, i) ds \quad (7)$$

Dynamic NeRF also predicts scene flow in neighboring time instance to the current time step, $\mathcal{F}^{(i)} = (\mathbf{f}_{\text{fw}}^{(i)}, \mathbf{f}_{\text{bw}}^{(i)})$. Instead of rendering, this flow prediction is used to create region masks and geometric constraints, as explained in Sec. 3.5.

At each time step, only one observation view is available. Consequently, generating the implicit representation of dynamic objects from a single video is an ill-posed problem. To effectively optimize our dynamic NeRF and maximize the usage of temporal information, we introduce multiple regularization losses described in Sec. 3.5.

3.4. Occlusion Weight

Our background pipeline models the time-invariant component, and our dynamic NeRF generates a per-time-step deformation field. To combine them at pixel-level (ray-level), here we introduce our occlusion weight module, which targets rendering transmittance between static and dynamic components.

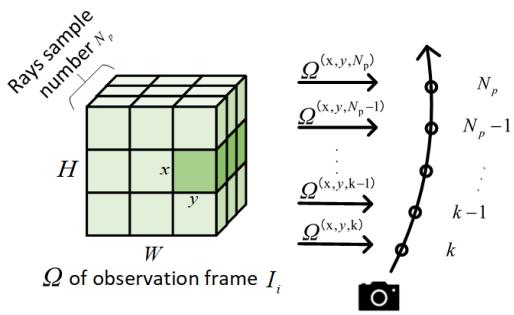


Figure 4. **Transmittance on each ray.** Each ray sample for a ray at observation time step i at location x, y is weighted by transmittance Ω over the ray dimension.

As shown in Fig. 4, at i -th frame, we calculate the transmittance $\Omega_b, \Omega_d \in \mathbb{R}^{H \times W \times N_p}$ corresponding to each sample for each ray and the transmittance $\mathcal{T} \in \mathbb{R}^{H \times W}$ ² corresponding to each ray using the warped previous rendering result I_{i-1} concatenated as input:

$$\mathbf{F}_w : (\mathbf{x}, \mathbf{d}, I_{i-1}) \rightarrow (\Omega_b, \Omega_d, \mathcal{T}). \quad (8)$$

The discrete integral formulation of rendered color $\hat{\mathbf{C}}_b(\mathbf{r})$ and $\hat{\mathbf{C}}_d^{(i)}(\mathbf{r})$ are weighted along the ray dimension (see supplemental material), giving us the final rendered pixel at time step i for location x, y :

$$\begin{aligned} \hat{\mathbf{C}}^{(i)}(\mathbf{r}) &= \mathcal{T}^{(x,y)} \Omega_b^{(x,y)} \hat{\mathbf{C}}_b(\mathbf{r}) \\ &\quad + (1 - \mathcal{T}^{(x,y)}) \Omega_d^{(x,y)} \hat{\mathbf{C}}_d^{(i)}(\mathbf{r}) \end{aligned} \quad (9)$$

Given the entire rays set $\mathcal{R} \in \mathbb{R}^{H \times W}$, the final rendering image can be calculated from $\hat{\mathbf{I}}(x, y, i) = \{\hat{\mathbf{C}}^{(i)}(\mathbf{r}) | \mathbf{r} \in \mathcal{R}\}$. Our experiments show that the occlusion weight helps decouple static and dynamic scene components.

3.5. Loss

The lack of input views makes modeling time-dependent scenes ill-posed and difficult to optimize. To this end, several losses are designed to control pipeline training.

Reconstruction loss. To learn the implicit representation from a N -frames input video clip $\mathcal{D} = \{(\mathbf{x}^i, \mathbf{d}^i, I^i) | i = 1, \dots, N\}$, we apply a reconstruction loss penalizing the difference between the target video frame $\mathbf{C}^{(i)}(\mathbf{r})$ and our yielding volume-rendered image $\hat{\mathbf{C}}^{(i)}(\mathbf{r})$:

$$\mathcal{L}_{\text{rec}} = \sum_{i=1}^N \sum_{\mathbf{r} \in \mathcal{R}} \left\| \hat{\mathbf{C}}^{(i)}(\mathbf{r}) - \mathbf{C}^{(i)}(\mathbf{r}) \right\|_2^2 \quad (10)$$

²In practice, limited by memory space, NeRF-based methods randomly choose N_{rand} rays rather than rendering all $H \times W$ rays for an image at each training iteration.

Distribution loss. Following Variational Auto-Encoder (VAE) [34], we impose scene distribution $P(\mathbf{z} \in \mathbb{R}^D | \Theta)$ to a specified constant distribution. For Gaussian form, we take the mean component μ in VAE as our latent variable sample $P(\mathbf{z} | \Theta) \sim \mathcal{N}(\mu, \sigma^2 = 0)$, and use the loss of this component throughout training:

$$\mathcal{L}_{\text{dist}} = \frac{1}{D} \sum_{i=1}^D -\frac{1}{2}(1 - \mu_i^2). \quad (11)$$

Transmittance loss. Our occlusion weight module mitigates the ambiguous problem in occluded components between frames by rendering transmittance, targeting at predicting the overlay relationship for rendering results between the backdrop and dynamic foreground. Ideally, each ray should be rendered from either dynamic or static parts, hence the corresponding transmittance should be regularized to 0 or 1. Ray samples need to exist initially, and this encourages transmittance for each sample to be close to one:

$$\begin{aligned} \mathcal{L}_{\text{w}} &= \sum_{\mathbf{r} \in \mathcal{R}} -\mathcal{T}^{(x,y)} \log(-\mathcal{T}^{(x,y)} + \epsilon) \\ &\quad + \sum_{\mathbf{r} \in \mathcal{R}} \sum_{k=1}^{N_p} \left\| 1 - \omega_b^{(k)} \right\|_1 + \left\| 1 - \omega_d^{(k)} \right\|_1. \end{aligned} \quad (12)$$

Depth and optical flow serve as geometry supervision priors in many NeRF-based methods, aiding resolve motion-appearance ambiguity and accelerating convergence [18, 40, 79]. Here we use depth loss $\mathcal{L}_{\text{depth}}$ and flow consistency losses $\mathcal{L}_{\text{cons}}, \mathcal{L}_{\text{flow}}$ for regularization, which will be discussed detailedly in supplemental material. With using λ coefficients weight each term, the overall loss can be interpreted as:

$$\begin{aligned} \mathcal{L}_{\text{overall}} &= \lambda_{\text{rec}} \mathcal{L}_{\text{rec}} + \lambda_{\text{dist}} \mathcal{L}_{\text{dist}} + \lambda_{\text{depth}} \mathcal{L}_{\text{depth}} \\ &\quad + \lambda_{\text{cons}} \mathcal{L}_{\text{cons}} + \lambda_{\text{flow}} \mathcal{L}_{\text{flow}} + \lambda_{\text{w}} \mathcal{L}_{\text{w}} \end{aligned} \quad (13)$$

4. Experiments

4.1. Datasets and Metrics

Urban driving scenes. We built a dataset of real-world street scenes captured from an autonomous-driving collection Argoverse [78]. All scenes are taken in urban street, using one of seven high-resolution surrounded monocular cameras recorded at 30 Hz. Each gathered video clip depicts an outdoor scene with vehicles or people performing dynamic actions, spans 20-50 frames, and is scaled to a resolution of 436×272 . We use corresponding estimation methods (See Sec. 4.2) to acquire depth, optical flow, and camera poses. Since monocular cameras cannot provide multiple views, we carry out time interpolation for evaluation in each scene. The recording frequency is decreased to 15 Hz for training.

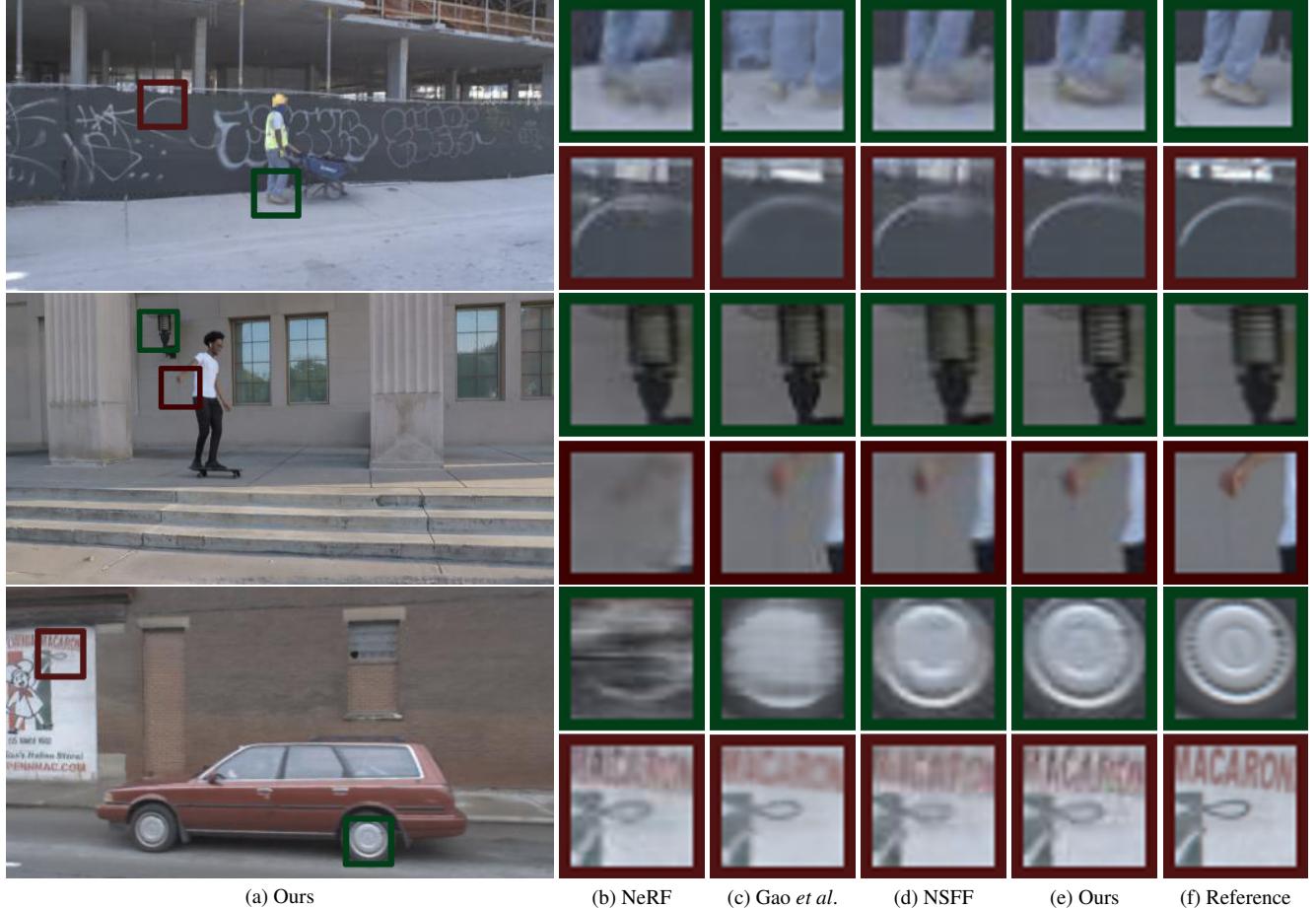


Figure 5. **Comparison of novel view synthesis details.** Our method shows a more effective way of recovering texture in both static and motion regions.

NVIDIA dynamic scenes. We then evaluate the dynamic scenes from NVIDIA [82], which comprise 4 scenes captured by a hand-held monocular moving camera and 8 scenes captured by 12 stationary multi-view cameras evenly distributed and manually synchronized. 7 multi-view scenes are selected as our evaluation target for their correct camera pose estimation. For each scene, 20 to 30 frames are extracted from the source video for training. At each time instance, just one of these cameras is utilized as input, and other 11 camera views per time step are used for evaluation. Selected clips are downsized to a resolution of 510×272 .

Metrics. All comparison approaches are evaluated by PSNR, SSIM [77] and LPIPS [85]. Higher PSNR and SSIM imply superior, while lower LPIPS indicate better.

4.2. Implementation Details

All experiments are carried out on 8 NVIDIA 3090 GPUs in the Pytorch framework. We use 48 threads of Intel(R) Xeon(R) Platinum 8255C CPU @ 2.50GHz to expedite data loading. The weight coefficients for

\mathcal{L}_{rec} , $\mathcal{L}_{\text{dist}}$, $\mathcal{L}_{\text{depth}}$, $\mathcal{L}_{\text{cons}}$, $\mathcal{L}_{\text{flow}}$, and \mathcal{L}_w are 1, 5e-1, 3e-2, 3e-2, 1e-2, and 5e-1, respectively. Adam optimizer [33] with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon=1e-8$ is used with learning rate of 3e-4. We generate $N_{\text{rand}} = 1024$ rays randomly in a batch for each training iteration, and we sample $N_p = 128$ ray points along each ray. We initially warm up our occlusion weight module to get the right flow estimation for 2e5 iterations supervised by ground truth flow estimated from RAFT [69]. Thereafter, all losses are involved to train another 5e5 iterations. To get ground truth depth, we leverage the state-of-the-art monocular image depth estimation method [59]. COLMAP SfM technique [62] is applied to estimate the camera poses, scene bounds, and camera params with the premise that intrinsics and extrinsics are fixed.

4.3. Quantitative Evaluation

To quantitatively validate the performance of our method, we carry out experiments on NVIDIA dynamic scenes and urban driving scenes. We compare D⁴NeRF with four baselines. The first is a classical learning-based

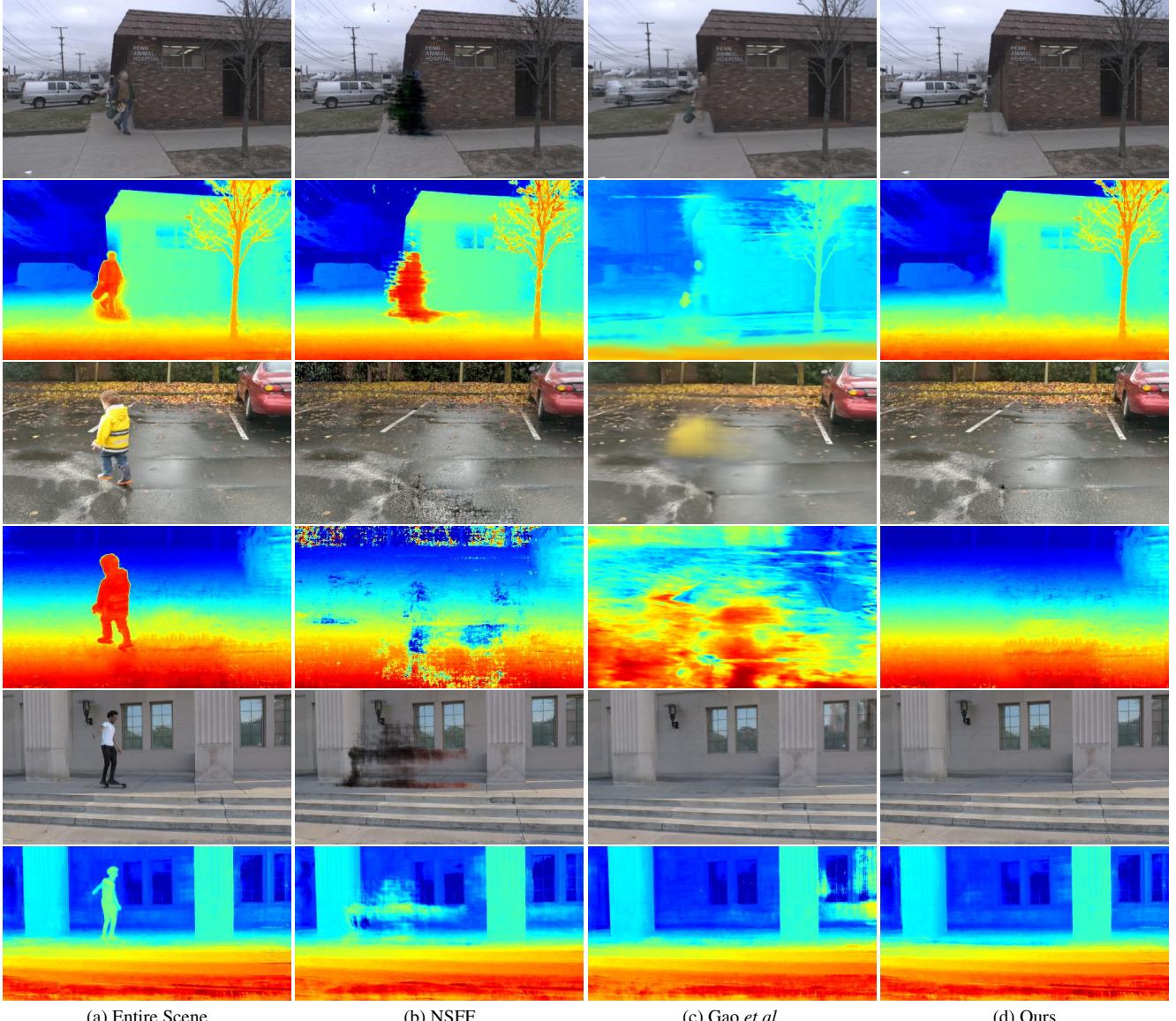


Figure 6. **Comparison on decoupling static background from entire scenes.** (a) shows the entire scene, (b-d) shows the separation results of the static background from NSFF, Gao *et al.*, and ours. D⁴NeRF obtains an uncontaminated background in both RGB and depth space.

method [82]. The second is vanilla NeRF [48]. The third and the fourth are recent state-of-the-art methods [23, 40] that also use multiple NeRFs to perform dynamic novel view synthesis. All baseline methods are implemented using the author’s provided official codebase and default hyper-parameters.

Tab. 1 shows the performance of our method on NVIDIA dynamic scenes compared to other baseline methods. D⁴NeRF achieves the best performance on average and in most scenes. Due to space limitations, we only present the results of PSNR and LPIPS on NVIDIA dynamic scenes, and please refer to our supplemental material for complete comparison results of all three metrics on both datasets.

Ablation study. We verify the effectiveness of different modules in our approach, including the distribution encoder (DT), the ray occlusion weight module (RW), and two regularization losses $\mathcal{L}_{\text{depth}}$ and $\mathcal{L}_{\text{flow}}$. We conduct our ablation study on NVIDIA dynamic scenes shown in Tab. 2. The first row shows the performance of only two naive NeRFs, while the second row proves the effectiveness of DT, which brings $0.79 \uparrow$ and $.0245 \downarrow$ improvement on PSNR and LPIPS. Besides, RW also shows an increase of 5.5% on PSNR and a decrease of 27.1% on LPIPS. The depth loss and the flow loss further show their efficacy in the fourth and fifth rows, respectively. Our ablation study shows that the model performs best when all modules are used.

	Skating	Balloon1	Balloon2	Truck	Umbrella	Jumping	Playground	Ave								
	PSNR	LPIPS														
Yoon <i>et al.</i> [82]	24.33	.1563	19.33	.1625	19.78	.1797	28.78	.0835	20.37	.1696	21.51	.2294	17.43	.2124	21.65	.1705
NeRF [48]	25.40	.0906	21.41	.1412	23.30	.0705	27.93	.0975	21.23	.2341	22.21	.1511	20.75	.1566	23.18	.1345
NSFF [40]	33.42	.0235	23.39	.0762	27.85	.0490	32.09	.0372	23.67	.1153	26.67	.0624	23.77	.0857	27.26	.0642
Gao <i>et al.</i> [23]	33.83	.0225	23.60	.0736	27.90	.0457	31.00	.0321	24.25	.1071	26.72	.0584	23.82	.0823	27.30	.0603
Ours	34.52	.0171	23.87	.0666	<u>27.60</u>	<u>.0411</u>	<u>31.75</u>	<u>.0232</u>	<u>24.20</u>	<u>.1037</u>	<u>27.00</u>	<u>.0463</u>	<u>23.94</u>	<u>.0734</u>	<u>27.55</u>	<u>.0530</u>

Table 1. **Quantitative comparison results on NVIDIA dynamic scenes.** The best result is in bold, and the second-best is underlined in each column.

Method	DT	RW	$\mathcal{L}_{\text{flow}}$	$\mathcal{L}_{\text{depth}}$	PSNR/LPIPS
Base					25.23/.1144
Base+DT	✓				26.02/.0899
Base+DT+RW	✓	✓			27.44/.0655
Ours+ $\mathcal{L}_{\text{flow}}$	✓	✓	✓		27.50/.0601
Ours+ $\mathcal{L}_{\text{flow}}+\mathcal{L}_{\text{depth}}$	✓	✓	✓	✓	27.55/.0530

Table 2. **Ablation study.** We evaluate different modules on NVIDIA dynamic scenes dataset. PSNR and LPIPS are shown on average.

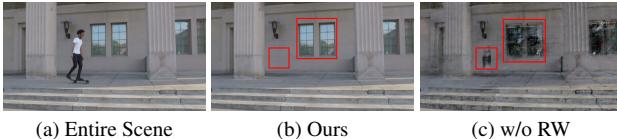


Figure 7. **Visual ablation on the occlusion weight module.** RW ensures proper separating capacity.

4.4. Qualitative Evaluation

While this section has many qualitative evaluations, we strongly advise the reader to watch the supplementary video for a more thorough look.

In Fig. 5, we compare novel view rendering visually. D⁴NeRF surpasses all other methods on dynamic scenes. NeRF is incapable of handling dynamic regions and causes significant ghosting. NSFF recovers dynamic objects properly but produces imprecise details, such as the wall in the first scene and the lamp in the second scene. Gao *et al.* capture finer details but does poorly in recovering exact motion. Our method presents the highest qualitative results in recovering static texture and motion details.

Background decoupling. We further show that our approach could decouple static backgrounds from entire dynamic scenes self-supervised. D⁴NeRF can restore the unoccluded clean background simply by observing the occluded regions in different frames of a video clip. Other NeRF-based methods also tend to infer occlusion relationships, but get subpar performance. As seen in Fig. 6, compared to other multi-NeRF-based methods, D⁴NeRF obtains the most accurate separation results in RGB-D space with fewer artifacts and blurs. In Fig. 7, we also show ablation that without our occlusion weight, the decoupled syn-

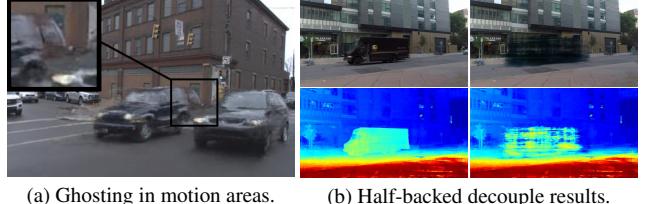


Figure 8. **Limitations.** Our method fails to recover the motion area (a) and is unable to produce a complete decoupling result (b) with incorrect flow or depth estimation.

thesized results are fuzzy and noisy.

Though the separation of static components from dynamic scenes has been addressed in 2D geometric algorithms, the geometry and technique of inpainting lack 3D comprehension, leading to restricted results in arbitrary view synthesis and time interpolation. Our approach gives a 3D pattern for dealing with this problem and allows the reconstruction of a decoupled implicit 3D scene representation. This makes it possible to generate novel views at arbitrary viewpoints and any input time step.

5. Conclusion

We presented D⁴NeRF, a method for modeling dynamic scenes in the real world and performing novel view synthesis from casual monocular videos. D⁴NeRF provides a 3D pattern that could decouple the occlusion area into static and dynamic components and recovers a clean background. We empirically demonstrate superior quantitative and qualitative performance on both urban driving scenes and NVIDIA dynamic scenes. Our work suggests various directions for future research, such as exploring high-quality decomposition and editor models for 3D dynamic scenes. In addition, we notice that some approaches [55, 56] use deformable latent code to control the representation of dynamic regions. The same can be applied in D⁴NeRF to generate delicate deformations for dynamic objects.

Limitations. Same as other NeRF-based methods, our approach relies on accurate estimation, *e.g.* camera poses and optical flow, to enable the proper representation of dynamic and static parts. Inaccurate ones may lead to motion artifacts and incorrect decoupling results (See Fig. 8).

A. Details of Rendering Formulations

A.1. Attention Block

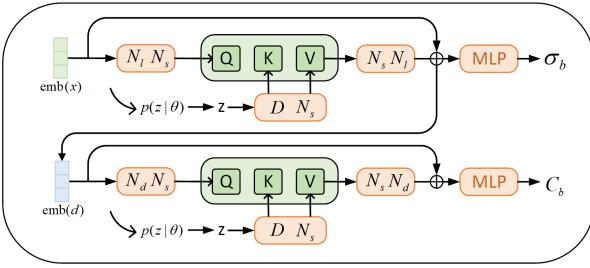


Figure 9. **Detailed Pipeline of Attention block.** The numbers under each network layer represent the input and output channels. \oplus denotes element-wise addition.

Here we describe attention block detailedly. To numerically estimate the continuous rendering integral, we adopt the same sampling technique and embedding function in NeRF [48]. The volume depth range can be partitioned into N_p evenly-spaced bins, and we draw one sample uniformly and randomly from within each bin. Input embedding 3D location and view direction are written as $emb(x) \in \mathbb{R}^{N_l \times N_p}$, $emb(d) \in \mathbb{R}^{N_d \times N_p}$. N_l and N_d are the dimensions extended by embedding. We use fully connected layers $\mathbf{F}_\downarrow : \mathbb{R}^{* \times N_p} \rightarrow \mathbb{R}^{N_s \times N_p}$ to project inputs into a lower-dimension space N_s . In order to reduce calculation overhead, both in terms of time and memory space, we accumulate the latent features along with rays via average pooling over the ray dimension $\mathbf{F}_p(\mathbf{x}) = \frac{1}{N} \sum_{i=1}^N \mathbf{x}[:, i] : \mathbb{R}^{* \times N_p} \rightarrow \mathbb{R}^*$. The attention can be expressed as:

$$\begin{aligned} \mathbf{Q}_l &= \mathbf{F}_p(\mathbf{F}_\downarrow(emb(\mathbf{x}))), \\ \mathbf{Q}_d &= \mathbf{F}_p(\mathbf{F}_\downarrow(emb(\mathbf{d}))), \\ \mathbf{K} &= \mathbf{V} = \mathbf{F}_p(\mathbf{F}_\downarrow(\mathbf{z})). \end{aligned} \quad (14)$$

We use $\mathbf{F}_\uparrow : \mathbb{R}^{N_s} \rightarrow \mathbb{R}^{* \times N_p}$ to re-project inputs into the original-dimension space and extend elements along the ray dimension with interleaving repetition. The attention feature of location is denoted as:

$$\begin{aligned} \mathbf{C}_l &= \mathbf{F}_\uparrow(\text{Att}(\mathbf{Q}_l)) + emb(\mathbf{x}), \\ \text{where } \text{Att}(\mathbf{Q}_l) &= \text{softmax}\left(\frac{\mathbf{Q}_l \mathbf{K}^T}{\sqrt{d}}\right) \mathbf{V}. \end{aligned} \quad (15)$$

A.2. Discrete Integral with Occlusion Weight

The discrete integral formulations of rendered color $\hat{\mathbf{C}}_b(\mathbf{r})$, $\hat{\mathbf{C}}_d^{(i)}(\mathbf{r})$ are weighted by ω_b and ω_d over the ray dimension:

$$\Omega_b^{(x,y)} \hat{\mathbf{C}}_b(\mathbf{r}) = \sum_{k=1}^{N_p} \left(T_k \left(1 - e^{-\sigma^{(k)} \delta^{(k)}} \right) \mathbf{c}^{(k)} \omega_b^{(k)} \right),$$

$$\Omega_d^{(x,y)} \hat{\mathbf{C}}_d^{(i)}(\mathbf{r}) = \sum_{k=1}^{N_p} \left(T_k \left(1 - e^{-\sigma^{(k)} \delta^{(k)}} \right) \mathbf{c}^{(i,k)} \omega_d^{(k)} \right),$$

$$\text{where } T_k = e^{-\sum_{j=1}^{k-1} \sigma^{(j)} \delta^{(j)}}, \delta_i = s_{i+1} - s_i. \quad (16)$$

B. Regularization Loss

Depth loss. Depth serves as a geometry prior in many NeRF-based methods, and helps resolve motion–appearance ambiguity and accelerates convergence [18, 79]. We use depth loss to supervise the geometry learning of the entire scene. For neural rendering, depth is generated as an additional output using $\hat{\mathbf{D}}^{(i)}(\mathbf{r}) = \int_{s_n}^{s_f} T(s) \sigma(\mathbf{r}(s), i) s ds$. Our depth loss aims at minimizing the difference between rendered depth $\hat{\mathbf{D}}^{(i)}(\mathbf{r})$ and ground truth depth $\mathbf{D}^{(i)}(\mathbf{r})$:

$$\mathcal{L}_{\text{depth}} = \sum_{i=1}^N \sum_{\mathbf{r} \in \mathcal{R}} \left\| \hat{\mathbf{D}}^{(i)}(\mathbf{r}) - \mathbf{D}^{(i)}(\mathbf{r}) \right\|_2^2 \quad (17)$$

Optical flow loss. Optical flow presents a general method to describe dynamic motion. At i -th time step, for a pixel located at (x, y, z) , flow is defined as $\mathbf{f} = (f_x, f_y, f_z) = \left(\frac{\partial x}{\partial t}, \frac{\partial y}{\partial t}, \frac{\partial z}{\partial t} \right)$. Different from the semantic segmentation models for a specific single domain [30, 35], flow-based segmentation provides a generalized paradigm for decoupling motion and static. We generate masks of dynamic objects and static backgrounds using optical flow estimation:

$$\hat{\mathcal{M}}^{(i)}(\mathbf{r}) = \begin{cases} 1, & \text{dynamic} \\ 0, & \text{static} \end{cases}$$

Warping manipulates the current frame for distorting position to adjacent frames. With optical flow, the future position of a continuous point $\mathbf{v}^{(i-1)} \in \mathbb{R}^3$ at timestamp $i-1$ is obtained through:

$$\hat{\mathbf{v}}^{(i)} = \text{warp}(\mathbf{v}^{(i-1)}, i) = \mathbf{v}^{(i-1)} + \mathbf{f}_{\text{fw}}^{(i-1)}. \quad (18)$$

We use the operator $\text{warp}(\cdot)$ to denote warping. To maintain temporal photometric consistency in neighboring time steps, we force the rendering results of adjacent frames to be consistent, constraining forward and backward simultaneously:

$$\begin{aligned} \mathcal{L}_{\text{cons}} &= \sum_{i=2}^N \sum_{\mathbf{r} \in \mathcal{R}} \hat{\mathcal{M}}^{(i)}(\mathbf{r}) \left\| \text{warp} \left(\hat{\mathbf{C}}^{(i-1)}(\mathbf{r}), i \right) - \mathbf{C}^{(i)}(\mathbf{r}) \right\|_2^2 \\ &\quad + \sum_{i=1}^{N-2} \sum_{\mathbf{r} \in \mathcal{R}} \hat{\mathcal{M}}^{(i)}(\mathbf{r}) \left\| \text{warp} \left(\hat{\mathbf{C}}^{(i+1)}(\mathbf{r}), i \right) - \mathbf{C}^{(i)}(\mathbf{r}) \right\|_2^2 \end{aligned} \quad (19)$$

	Skating			Balloon1			Balloon2			Truck		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
Yoon <i>et al.</i> [82]	24.33	.8273	.1563	19.33	.7324	.1625	19.78	.7458	.1797	28.78	.9451	.0835
NeRF [48]	25.40	.9419	.0906	21.41	.8630	.1412	23.30	.9198	.0705	27.93	.9289	.0975
NSFF [40]	33.42	.9754	.0235	23.39	<u>.8822</u>	.0762	<u>.27.85</u>	.9355	.0490	32.09	.9618	.0372
Gao <i>et al.</i> [23]	33.83	<u>.9782</u>	<u>.0225</u>	<u>23.60</u>	.8795	<u>.0736</u>	27.90	.9342	<u>.0457</u>	31.00	<u>.9628</u>	<u>.0321</u>
Ours	34.52	.9824	.0171	23.87	.8835	.0666	27.60	<u>.9348</u>	.0411	<u>31.75</u>	.9652	.0232

Table 3. **Quantitative comparison results on Skating, Balloon1, Balloon1 and Truck.** The **best** result is in bold, and the second-best is underlined in each column.

	Umbrella			Jumping			Playground			Ave		
	PSNR	SSIM	LPIPS									
Yoon <i>et al.</i> [82]	20.37	.7344	.1696	21.51	.7928	.2294	17.43	.5479	.2124	21.65	.7608	.1705
NeRF [48]	21.23	.7793	.2341	22.21	.8466	.1511	20.75	.8253	.1566	23.18	.8721	.1345
NSFF [40]	23.67	.8320	.1153	26.67	<u>.9220</u>	.0624	23.77	.8707	.0857	27.26	.9114	.0642
Gao <i>et al.</i> [23]	24.25	.8445	<u>.1071</u>	<u>26.72</u>	.9132	<u>.0584</u>	<u>23.82</u>	.8766	<u>.0823</u>	<u>27.30</u>	<u>.9127</u>	<u>.0602</u>
Ours	24.20	<u>.8323</u>	.1037	27.00	.9262	.0463	23.94	.8795	.0734	27.55	.9148	.0531

Table 4. **Quantitative comparison results on Umbrella, Jumping and playground.** Last column shows the average performance on NVIDIA dynamic scenes. The **best** result is in bold, and the second-best is underlined in each column.

	S1			S2			S3		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
Yoon <i>et al.</i> [82]	35.33	.9542	.0544	25.31	.8955	.1526	28.77	.9014	.1138
NeRF [48]	36.40	.9720	.0349	24.90	.8846	.1384	30.87	.9152	.0871
NSFF [40]	39.56	<u>.9902</u>	<u>.0094</u>	31.41	.9525	.0398	32.19	.9441	.0529
Gao <i>et al.</i> [23]	<u>40.61</u>	.9897	.0112	<u>31.56</u>	.9544	<u>.0393</u>	<u>31.55</u>	.9365	.0636
Ours	40.92	.9921	.0081	31.91	.9594	.0367	31.45	<u>.9372</u>	<u>.0537</u>

Table 5. **Quantitative comparison results on S1, S2 and S3.** The **best** result is in bold, and the second-best is underlined in each column.

	S4			S5			Ave		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
Yoon <i>et al.</i> [82]	25.12	.8829	.1409	31.40	.9651	.0562	29.19	.9198	.1036
NeRF [48]	25.26	.8848	.1323	33.66	<u>.9736</u>	.0366	30.22	.9260	.0859
NSFF [40]	26.20	.9176	.0790	<u>34.67</u>	<u>.9803</u>	<u>.0254</u>	32.81	.9569	.0413
Gao <i>et al.</i> [23]	26.53	.9259	.0608	34.02	.9788	.0327	<u>32.85</u>	<u>.9571</u>	.0415
Ours	<u>26.40</u>	<u>.9225</u>	<u>.0616</u>	35.10	.9818	.0196	33.16	.9586	.0359

Table 6. **Quantitative comparison results on S4, S5.** Last column shows the average performance on urban street scenes. The **best** result is in bold, and the second-best is underlined in each column.

Generating optical flow correctly from scratch is also challenging. Similar to methods [20, 23, 40] regularizing geometric meaning on optical flow, we minimize the difference between forward and backward flow and the penalty term on flow:

$$\begin{aligned} \mathcal{L}_{\text{flow}} = & \sum_{i=1}^{N-1} \sum_{\mathbf{r} \in \mathcal{R}} \left\| \mathbf{f}_{\text{fw}}^{(i)} - \mathbf{f}_{\text{bw}}^{(i+1)} \right\|_2^2 \\ & + \frac{1}{N-1} \sum_{i=2}^{N-1} \sum_{\mathbf{r} \in \mathcal{R}} \left(\left\| \mathbf{f}_{\text{fw}}^{(i)} \right\|_1 + \left\| \mathbf{f}_{\text{bw}}^{(i)} \right\|_1 \right). \end{aligned} \quad (20)$$

C. Additional Comparison Results

Here we offer more comparison results on NVIDIA dynamic scenes dataset and urban street scenes dataset, evaluating with all three comparison metrics PSNR, SSIM, and LPIPS. Comparisons for each scene on NVIDIA dataset are shown in Tab. 3 and Tab. 4. Evaluation results in urban street scenes are presented in Tab. 5 and Tab. 6.

References

- [1] Benjamin Attal, Eliot Laidlaw, Aaron Gokaslan, Changil Kim, and Matthew O’Toole. T’orf: Time-of-flight radiance fields for dynamic scene view synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 2, 3
- [2] Shai Avidan and Amnon Shashua. Novel view synthesis in tensor space. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1997. 1
- [3] Shai Avidan and Amnon Shashua. Novel view synthesis by cascading trilinear tensors. *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, 4(4):293–306, 1998. 1
- [4] Aayush Bansal, Minh Vo, Yaser Sheikh, Deva Ramanan, and Srinivasa Narasimhan. 4d visualization of dynamic events from unconstrained multi-view videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 3
- [5] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2
- [6] Mojtaba Bemana, Karol Myszkowski, Hans-Peter Seidel, and Tobias Ritschel. X-fields: Implicit neural view-, light- and time-image interpolation. *ACM Transactions on Graphics (TOG)*, 39(6):1–15, 2020. 3
- [7] Michael Broxton, John Flynn, Ryan Overbeck, Daniel Erickson, Peter Hedman, Matthew Duvall, Jason Dourgarian, Jay Busch, Matt Whalen, and Paul Debevec. Immersive light field video with a layered mesh representation. *ACM Transactions on Graphics (TOG)*, 39(4):86–1, 2020. 1
- [8] Chris Buehler, Michael Bosse, Leonard McMillan, Steven Gortler, and Michael Cohen. Unstructured lumigraph rendering. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques (CGIT)*, pages 425–432, 2001. 2
- [9] G. Cai, K. Yan, Z. Dong, I. Gkioulekas, and S. Zhao. Physics-based inverse rendering using combined implicit and explicit geometries. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1
- [10] Jin-Xiang Chai, Xin Tong, Shing-Chow Chan, and Heung-Yeung Shum. Plenoptic sampling. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques (CGIT)*, pages 307–318, 2000. 2
- [11] Gaurav Chaurasia, Sylvain Duchene, Olga Sorkine-Hornung, and George Drettakis. Depth synthesis and local warps for plausible image-based navigation. *ACM Transactions on Graphics (TOG)*, 32(3):1–12, 2013. 2
- [12] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2
- [13] Inchang Choi, Orazio Gallo, Alejandro Troccoli, Min H Kim, and Jan Kautz. Extreme view synthesis. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019. 2
- [14] Alvaro Collet, Ming Chuang, Pat Sweeney, Don Gillett, Dennis Evseev, David Calabrese, Hugues Hoppe, Adam Kirk, and Steve Sullivan. High-quality streamable free-viewpoint video. *ACM Transactions on Graphics (TOG)*, 34(4):1–13, 2015. 1
- [15] Antonio Criminisi, Andrew Blake, Carsten Rother, Jamie Shotton, and Philip HS Torr. Efficient dense stereo with occlusions for new view-synthesis by four-state dynamic programming. *International Journal of Computer Vision (IJCV)*, 71(1):89–110, 2007. 1
- [16] B Curless and S. M Seitz. Spacetime stereo: shape recovery for dynamic scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2003. 1
- [17] Paul E Debevec, Camillo J Taylor, and Jitendra Malik. Modeling and rendering architecture from photographs: A hybrid geometry-and image-based approach. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques (CGIT)*, pages 11–20, 1996. 2
- [18] Kangle Deng, Andrew Liu, Jun-Yan Zhu, and Deva Ramanan. Depth-supervised nerf: Fewer views and faster training for free. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 5, 9
- [19] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 4
- [20] Yilun Du, Yinan Zhang, Hong Xing Yu, Joshua B. Tenenbaum, and Jiajun Wu. Neural radiance flow for 4d view synthesis and video processing. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2021. 2, 3, 10
- [21] J. Flynn, M. Broxton, P. Debevec, M. Duvall, G. Fyffe, R. Overbeck, N. Snavely, and R. Tucker. Deepview: View synthesis with learned gradient descent. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1
- [22] John Flynn, Keith Snavely, Ivan Neulander, and James Philbin. Deepstereo: learning to predict new views from real world imagery, Mar. 13 2018. US Patent 9,916,679. 2
- [23] C. Gao, A. Saraf, J. Kopf, and J. B. Huang. Dynamic view synthesis from dynamic monocular video. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2021. 2, 3, 7, 8, 10
- [24] Steven J Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F Cohen. The lumigraph. In *Proceedings of the*

- 23rd annual conference on Computer graphics and interactive techniques (CGIT)*, pages 43–54, 1996. 2
- [25] Peter Hedman, Suhib Alsisan, Richard Szeliski, and Johannes Kopf. Casual 3d photography. *ACM Transactions on Graphics (TOG)*, 36(6):1–15, 2017. 2
- [26] Peter Hedman, Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel Brostow. Deep blending for free-viewpoint image-based rendering. *ACM Transactions on Graphics (TOG)*, 37(6):1–15, 2018. 2
- [27] Philipp Henzler, Volker Rassche, Timo Ropinski, and Tobias Ritschel. Single-image tomography: 3d volumes from 2d x-rays. *Computer Graphics Forum (CGF)*, 37(2), 2017. 1
- [28] Zeng Huang, Tianye Li, Weikai Chen, Yajie Zhao, Jun Xing, Chloe LeGendre, Linjie Luo, Chongyang Ma, and Hao Li. Deep volumetric video from very sparse multi-view performance capture. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. 3
- [29] Valentin K Ivanov, Vladimir V Vasin, and Vitalii P Tanana. *Theory of Linear Ill-Posed Problems and its Applications*, volume 36. Walter de Gruyter, 2013. 1
- [30] W. Jiang, K. M. Yi, G. Samei, O. Tuzel, and A. Ranjan. Neuman: Neural human radiance field from a single video. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022. 2, 3, 9
- [31] Takeo Kanade, Peter Rander, and PJ Narayanan. Virtualized reality: Constructing virtual worlds from real scenes. *IEEE Transactions on Multimedia (MM)*, 4(1):34–47, 1997. 3
- [32] Abhishek Kar, Christian Hne, and Jitendra Malik. Learning a multi-view stereo machine. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. 1
- [33] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6
- [34] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 5
- [35] A. Kundu, K. Genova, X. Yin, A. Fathi, C. Pantofaru, L. Guibas, A. Tagliasacchi, F. Dellaert, and T. Funkhouser. Panoptic neural fields: A semantic object-aware neural scene representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1, 2, 3, 9
- [36] Chenyang Lei, Yazhou Xing, and Qifeng Chen. Blind video temporal consistency via deep video prior. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 1
- [37] Marc Levoy and Pat Hanrahan. Light field rendering. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques (CGIT)*, pages 31–42, 1996. 2
- [38] Hao Li, Linjie Luo, Daniel Vlasic, Pieter Peers, Jovan Popović, Mark Pauly, and Szymon Rusinkiewicz. Temporally coherent completion of dynamic shapes. *ACM Transactions on Graphics (TOG)*, 31(1):1–11, 2012. 3
- [39] Tianye Li, Mira Slavcheva, Michael Zollhoefer, Simon Green, Christoph Lassner, Changil Kim, Tanner Schmidt, Steven Lovegrove, Michael Goesele, Richard Newcombe, and Zhaoyang Lv. Neural 3D video synthesis from multi-view video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1, 2, 3
- [40] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2, 3, 5, 7, 8, 10
- [41] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2021. 4
- [42] Xuan Luo, Jia-Bin Huang, Richard Szeliski, Kevin Matzen, and Johannes Kopf. Consistent video depth estimation. In *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH)*, volume 39. ACM, 2020. 1
- [43] Ricardo Martin-Brualla, Noha Radwan, Mehdi S. M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2
- [44] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2
- [45] Mateusz Michalkiewicz, Jhony K Pontes, Dominic Jack, Mahsa Baktashmotagh, and Anders Eriksson. Deep level sets: Implicit surface representations for 3d shape inference. *arXiv preprint arXiv:1901.06802*, 2019. 2
- [46] B. Mildenhall, P. Hedman, R. Martin-Brualla, P. Srinivasan, and J. T. Barron. Nerf in the dark: High dynamic range view synthesis from noisy raw images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1
- [47] Ben Mildenhall, Pratul P Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Transactions on Graphics (TOG)*, 38(4):1–14, 2019. 2
- [48] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 1, 2, 7, 8, 9, 10
- [49] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1
- [50] Simon Niklaus, Long Mai, Jimei Yang, and Feng Liu. 3d ken burns effect from a single image. *ACM Transactions on Graphics (TOG)*, 38(6):1–15, 2019. 2
- [51] Sergio Orts-Escalano, Christoph Rhemann, Sean Fanello, Wayne Chang, Adarsh Kowdle, Yury Degtyarev, David Kim, Philip L Davidson, Sameh Khamis, Mingsong Dou, et al. Holoportation: Virtual 3d teleportation in real-time. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology (UIST)*, pages 741–754, 2016. 1

- [52] Martin Ralf Oswald, Jan Stühmer, and Daniel Cremers. Generalized connectivity constraints for spatio-temporal 3d reconstruction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2014. 1
- [53] Ryan S. Overbeck, Daniel Erickson, Daniel Evangelakos, and Paul Debevec. The making of welcome to light fields vr. In *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH)*, pages 1–2. ACM, 2018. 1
- [54] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2
- [55] Keunhong Park, Utkarsh Sinha, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Steven M. Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2021. 2, 3, 8
- [56] Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M Seitz. Hypernerf: A higher-dimensional representation for topologically varying neural radiance fields. *arXiv preprint arXiv:2106.13228*, 2021. 8
- [57] Eric Penner and Li Zhang. Soft 3d reconstruction for view synthesis. *ACM Transactions on Graphics (TOG)*, 36(6):1–11, 2017. 2
- [58] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2, 3
- [59] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2020. 6
- [60] K. Rematas, A. Liu, P. P. Srinivasan, J. T. Barron, A. Tagliasacchi, T. Funkhouser, and V. Ferrari. Urban radiance fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2
- [61] Gernot Riegler and Vladlen Koltun. Free view synthesis. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 2
- [62] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 6
- [63] Lixin Shi, Haitham Hassanieh, Abe Davis, Dina Katabi, and Fredo Durand. Light field reconstruction using sparsity in the continuous fourier domain. *ACM Transactions on Graphics (TOG)*, 34(1):1–13, 2014. 2
- [64] Noah Snavely, Steven M Seitz, and Richard Szeliski. Photo tourism: exploring photo collections in 3d. In *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH)*, pages 835–846. ACM, 2006. 2
- [65] Pratul P. Srinivasan, Richard Tucker, Jonathan T. Barron, Ravi Ramamoorthi, Ren Ng, and Noah Snavely. Pushing the boundaries of view extrapolation with multiplane images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1
- [66] Pratul P Srinivasan, Richard Tucker, Jonathan T Barron, Ravi Ramamoorthi, Ren Ng, and Noah Snavely. Pushing the boundaries of view extrapolation with multiplane images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2
- [67] M. Tancik, V. Casser, X. Yan, S. Pradhan, B. Mildenhall, P. P. Srinivasan, J. T. Barron, and H. Kretzschmar. Blocknerf: Scalable large scene neural view synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1, 2
- [68] Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 3
- [69] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 6
- [70] Ayush Tewari, Ohad Fried, Justus Thies, Vincent Sitzmann, Stephen Lombardi, Kalyan Sunkavalli, Ricardo Martin-Brualla, Tomas Simon, Jason Saragih, Matthias Nießner, et al. State of the art on neural rendering. In *Computer Graphics Forum (CGF)*, volume 39, pages 701–727. Wiley Online Library, 2020. 1
- [71] Edgar Tretschk, Ayush Tewari, Vladislav Golyanik, Michael Zollhfer, Christoph Lassner, and Christian Theobalt. Non-rigid neural radiance fields: Reconstruction and novel view synthesis of a dynamic scene from monocular video. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2021. 2, 3
- [72] Alex Trevithick and Bo Yang. Grf: Learning a general radiance field for 3d representation and rendering. *arXiv preprint arXiv:2010.04595*, 2020. 2
- [73] Richard Tucker and Noah Snavely. Single-view view synthesis with multiplane images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2
- [74] H. Turki, D. Ramanan, and M. Satyanarayanan. Meganerf: Scalable construction of large-scale nerfs for virtual fly-throughs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2
- [75] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. 4
- [76] L. Wang, J. Zhang, X. Liu, F. Zhao, Y. Zhang, Y. Zhang, M. Wu, L. Xu, and J. Yu. Fourier plenocubes for dynamic radiance field rendering in real-time. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2
- [77] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing (TIP)*, 13(4):600–612, 2004. 6

- [78] Benjamin Wilson, William Qi, Tanmay Agarwal, John Lambert, Jagjeet Singh, Siddhesh Khandelwal, Bowen Pan, Ratnesh Kumar, Andrew Hartnett, Jhony Kaesemeyer Pontes, Deva Ramanan, Peter Carr, and James Hays. Argoverse 2: Next generation datasets for self-driving perception and forecasting. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks (NeurIPS Datasets and Benchmarks 2021)*, 2021. 2, 5
- [79] Wenqi Xian, Jia Bin Huang, Johannes Kopf, and Changil Kim. Space-time neural irradiance fields for free-viewpoint video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2, 3, 5, 9
- [80] Yuanbo Xiangli, Lining Xu, Xingang Pan, Nanxuan Zhao, Anyi Rao, Christian Theobalt, Bo Dai, and Dahua Lin. Bungeenerf: Progressive neural radiance field for extreme multi-scale scene rendering. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022. 2
- [81] Zexiang Xu, Sai Bi, Kalyan Sunkavalli, Sunil Hadap, Hao Su, and Ravi Ramamoorthi. Deep view synthesis from sparse photometric images. *ACM Transactions on Graphics (TOG)*, 38(4):1–13, 2019. 2
- [82] Jae Shin Yoon, Kihwan Kim, Orazio Gallo, Hyun Soo Park, and Jan Kautz. Novel view synthesis of dynamic scenes with globally coherent depths from a monocular camera. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 1, 2, 3, 6, 7, 8, 10
- [83] Wentao Yuan, Zhaoyang Lv, Tanner Schmidt, and Steven Lovegrove. Star: Self-supervised tracking and reconstruction of rigid objects in motion with neural rendering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2
- [84] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. Nerf++: Analyzing and improving neural radiance fields. *arXiv:2010.07492*, 2021. 2
- [85] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 6
- [86] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: learning view synthesis using multiplane images. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH)*, 4, 2018. 1
- [87] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. *arXiv preprint arXiv:1805.09817*, 2018. 2
- [88] C Lawrence Zitnick, Sing Bing Kang, Matthew Uyttendaele, Simon Winder, and Richard Szeliski. High-quality video view interpolation using a layered representation. *ACM Transactions on Graphics (TOG)*, 23(3):600–608, 2004. 3