Point-DynRF: Point-based Dynamic Radiance Fields from a Monocular Video

Byeongjun Park Changick Kim
Korea Advanced Institute of Science and Technology (KAIST)
{pbj3810, changick}@kaist.ac.kr

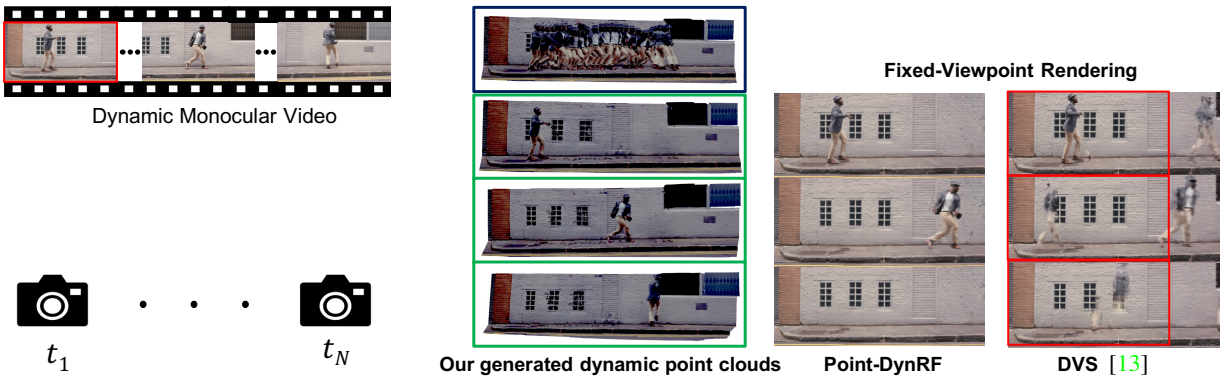


Figure 1. **Point-based Dynamic Radiance Fields for Long-Term Novel View Synthesis.** Point-DynRF takes a monocular video following dynamic objects, as shown in (a), and uses neural 3D points generated from the input video to efficiently represent dynamic radiance fields. (b) We design a novel framework that samples a subset point cloud (green boxes) at each time step from the entire point cloud (a blue box) and regresses dynamic radiance fields only on the scene surface where the subset point cloud are located. Especially with a wide-range camera trajectory, Point-DynRF addresses the duplicating problem of the state-of-the-art method (red boxes).

Abstract

*Dynamic radiance fields have emerged as a promising approach for generating novel views from a monocular video. However, previous methods enforce the geometric consistency to dynamic radiance fields only between adjacent input frames, making it difficult to represent the global scene geometry and degenerates at the viewpoint that is spatio-temporally distant from the input camera trajectory. To solve this problem, we introduce point-based dynamic radiance fields (**Point-DynRF**), a novel framework where the global geometric information and the volume rendering process are trained by neural point clouds and dynamic radiance fields, respectively. Specifically, we reconstruct neural point clouds directly from geometric proxies and optimize both radiance fields and the geometric proxies using our proposed losses, allowing them to complement each other. We validate the effectiveness of our method with experiments on the NVIDIA Dynamic Scenes Dataset and several causally captured monocular video clips.*

1. Introduction

Consider a monocular video recording of dynamic objects. While it is challenging to distinguish between static and dynamic areas in a single frame, analyzing the entire video sequence enables us to differentiate the background from the moving objects. Moreover, we can also predict the background outside a captured frame by assuming that the background scene remains constant over time. This scene reasoning ability enables us to identify the moving objects and integrate partially available scene information, which is crucial for understanding in-the-wild videos and scaling the free-viewpoint rendering.

Existing novel view synthesis methods for monocular videos often use separate modules for static and dynamic regions, where view-dependent radiance fields are designed for static regions and time-dependent radiance fields for dynamic regions [13, 22, 23, 25, 31, 39, 44, 47]. In this regard, recent deformable NeRFs [12, 31, 32, 44] learn sufficient view dependencies from small camera trajectories to represent the background, while representing the remaining regions using time-dependent radiance fields. However, in

the real world, there are many cases where the camera does not follow a narrow trajectory, and deformable NeRFs fail to distinguish between the background and dynamic objects due to the lack of learning view dependencies.

On the other hand, flow-based methods [13, 22, 23, 25] use additional supervisions from pre-trained depth [33], optical flow [38] and semantic segmentation [16] estimation networks to constrain the radiance field since identifying moving objects and estimating their motion in monocular videos are challenging. By imposing geometric constraints on the radiance field, flow-based methods can design dynamic radiance fields for large scenes. Despite its scalability, we observe that flow-based methods quickly degenerate for viewpoints in spatio-temporally distant from the input camera trajectory, and the generated image is blurry and sometimes contain duplicated objects. This is because time-dependent radiance fields are trained by the optical flow supervision to satisfy geometric consistency between adjacent frames, which fails to incorporate global geometric information of entire scene from wide-range camera trajectories. Figure 1-(b) shows the problem of a state-of-the-art dynamic view synthesis method [13] where a person is duplicated outside of the input frame and the background is not preserved after the person walks by because of the duplicated person.

Motivated by our observations, we introduce point-based dynamic radiance fields (**Point-DynRF**) to represent the entire scene geometry and produce more realistic long-term novel view synthesis results. Point-DynRF is built upon the Point-NeRF [46] representation, which reconstructs 3D neural point clouds and encodes the localized scene representation from neighboring neural points. While Point-NeRF aims at static scenes, we extend it to consider the time domain where different subsets of neural point clouds are sampled at each time step to represent time-varying radiance fields. Specifically, we utilize a pre-trained depth estimation network [33] and pre-defined foreground masks [13] to initialize pixel-wise depth and rigidness of our neural point clouds, respectively. Moreover, we propose a dynamic ray marching, where we march a ray over a subset of the entire point cloud consisting of all background points and the dynamic points corresponding to the rendering time. As each subset of neural point clouds represents the actual scene surface of the corresponding rendering time, our Point-DynRF can regress dynamic radiance fields only on the scene surface at that rendering time and alleviate to generate of duplicated dynamic objects.

To train Point-DynRF, we simply modify the training objective of DVS [13] and jointly optimize the neural point clouds and dynamic radiance fields, rather than solely supervising the radiance fields using initialized depth and foreground masks. Specifically, we train Point-DynRF to align the initialized learnable depth and foreground masks with the volume rendered depth and dynamicsness maps. Through the joint optimization scheme, the global scene geometry and dynamic radiance fields are further refined and complement each other, addressing the degeneration problems of previous methods in long-term dynamic view synthesis. Extensive experiments on the NVIDIA Dynamic Scenes [47] and several monocular video clips show the efficiency and effectiveness of our method.

2. Related Works

Neural representations for novel view synthesis. Novel view synthesis aims to generate new views of a scene given multiple posed images. To consider the arbitrary viewpoints in three-dimension, multiple-view geometry is often utilized and combined with image-based rendering methods to synthesize realistic novel views [9, 10, 20, 34, 50]. Moreover, deep neural networks have been explored to improve the visual quality of novel views by using explicit geometric proxies, such as multi-plane image [36, 42, 49], point cloud [1, 40, 43], and voxel [12, 35].

Recently, coordinate-based neural representations [8, 26, 27, 29] have achieved outstanding results in modeling the scene as implicit scene representations. In the context of novel view synthesis, Neural Radiance Fields (NeRF) [27] has been proposed to model the scene as a continuous volumetric field with neural networks. The success of NeRF is attributed to the extension of neural representation design, which facilitates free-viewpoint rendering with various applications, such as relighting [4], appearance editing [24, 48], reflections [14], and generative models [5, 7, 28]. Despite its remarkable scalability, several methods [19,46] focus on the fact that NeRF samples a large number of unnecessary points for each ray. Specifically, Point-NeRF [46] models a volumetric radiance field with 3D neural point clouds, avoiding ray sampling in the empty space and encoding localized scene representations. Our work extends Point-NeRF, encoding different scene representations for static and dynamic regions by leveraging its capability to encode localized scene representations.

Dynamic view synthesis for videos. Dynamic view synthesis focuses on generating novel views with dynamically moving objects at arbitrary viewpoints and time stamps. Several works have been proposed to model time-varying scenes on multiple time-synchronized videos [3, 21, 37, 50], sparse camera views [11, 17], stereo camera [2], and specific domain [6, 15, 41]. However, modeling neural scene representation from a monocular video is more challenging since it contains a single viewpoint for each time stamp. This causes ambiguities that radiance can be changed in either a view-dependent or a time-varying or both. To solve this ambiguity, Yoon *et al*. [47] combines an explicit depth estimation module to leverage geometric transformations

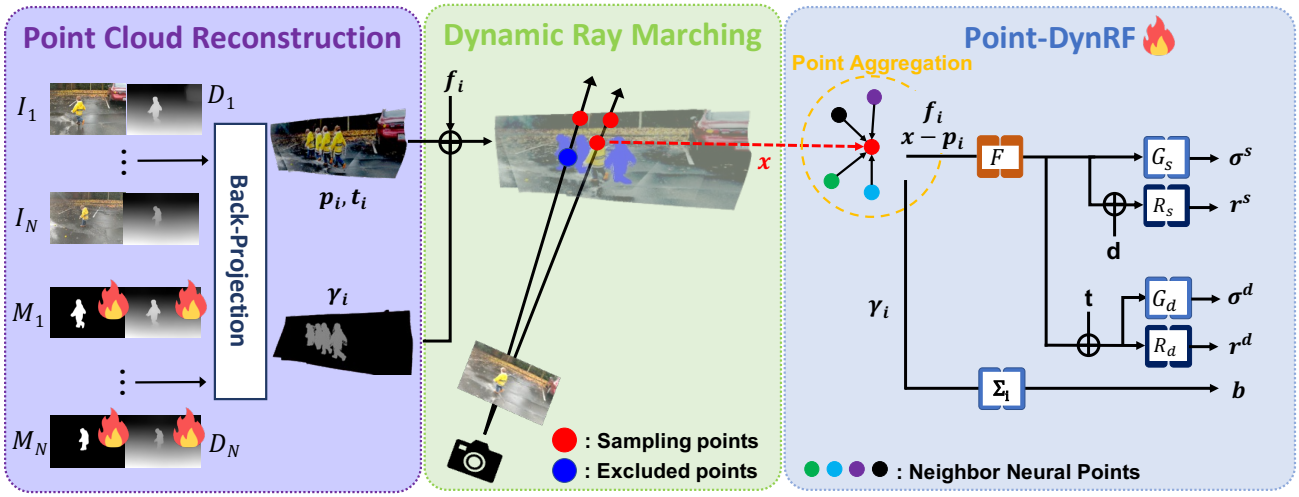


Figure 2. **An overview of network architecture.** Our framework consists of three components. First, we initialize per-frame depth maps D_n and foreground masks M_n for a given N frames. Then, we back-project each pixel of N frames to reconstruct our neural 3D point clouds. Each neural point i contains its spatio-temporal locations $(\mathbf{p}_i, t_i)$, a point-wise rigidness γ_i, a randomly initialized neural feature vector f_i to represent the local scene representation. Then, we select a subset point cloud at a rendering time step t and assign sampling points where the ray meets the neural points as they march. Finally, we regress a volume density and a radiance on both view-dependent and time-dependent radiance fields. The volume density and radiance for each sampling point in the ray are integrated via volume rendering to output an RGB color.

(i.e., warping) and to blend strategies for synthesizing novel views of a dynamic scene, but it requires a time-consuming preprocessing to generate manually annotated foreground masks. Recently, flow-based methods [13, 22, 25, 45] directly regress 4D space-time radiance fields by using additional geometric proxies, such as depth [33] and optical flow [38] estimation networks. Geometric proxies are used as additional supervision to learn their deformation module and constrain temporal changes of a dynamic scene. Several methods [12, 30–32, 39, 44] propose deformable neural radiance fields by modeling a canonical template radiance field and a deformation field for each frame. Our work also uses geometric proxies for point cloud initialization, but we optimize the dynamic radiance fields and geometric proxies together based on the volume rendering process. Moreover, point-based dynamic radiance fields allow us to incorporate the entire scene geometry and regress the radiance fields from the actual scene surface for each rendering time.

3. Method

Given a monocular video $V = \{I_1, I_2, \dots, I_N\}$ consisting of N frames, our goal is to synthesize novel views at arbitrary viewpoints and time steps. To achieve this, we design point-based dynamic radiance fields as shown in Fig. 2. Our model is built on the Point-NeRF [46] representation and extends it to consider time-varying radiance fields. We briefly describe the volume rendering formulation in 3.1 and then explain how to extend Point-NeRF to consider the time domain in 3.2. Finally, we illustrate the optimization scheme of Point-DynRF in 3.3.

3.1. Volume rendering

We construct continuous volumetric fields for modeling dynamic scenes, following the formulation in NeRF [27]. Given the camera center $\mathbf{o} \in \mathbb{R}^3$ and viewing direction $\mathbf{d} \in \mathbb{R}^2$, each pixel's RGB color $\mathbf{C} \in \mathbb{R}^3$ is computed by marching a ray $\mathbf{r}(s) = \mathbf{o} + s\mathbf{d}$ through the pixel and approximate the integration over radiance and its volume density $\{(r_j, \sigma_j) \in \mathbb{R}^3 \times \mathbb{R} \mid j = 1, \dots, M\}$ for M sampling points in the ray as:

$$\mathbf{C}(\mathbf{r}) = \sum_{j=1}^{M} T_j(\alpha(\sigma_j \delta_j)) r_j, \tag{1}$$

$$T_j = \exp(-\sum_{k=1}^{j-1} \sigma_k \delta_k), \tag{2}$$

where $\alpha(x) = 1 - \exp(-x)$ outputs the opacity at each sampling point, δ_j is the distance between two adjacent sampling points and T_j represents a volume transmittance.

3.2. Point-DynRF Representation

Point-NeRF [46] is pre-trained on a multi-view stereo dataset [18] or uses only points located on the actual surface with high confidence from COLMAP. In dynamic scenes, however, it fails to accurately regress the scene geometry since dynamic objects disrupt to estimate point-to-point correspondences. To solve this ambiguity, we propose Point-DynRF with associated neural point clouds, which are initialized by imprecise depth maps and pre-defined fore-

ground masks, and jointly optimize scene geometry and dynamic radiance fields.

Neural Point Clouds Reconstruction. Our neural point clouds are reconstructed by depth maps $\{D_1, ..., D_N\}$ and foreground masks $\{M_1, ..., M_N\}$. We first initialize per-frame depths by using disparity maps $disp_n$ obtained from DPT [33] and convert it to depth maps with per-frame scale s_n and shift b_n values as:

$$D_n(p) = s_n/(disp_n(p) + b_n). \tag{3}$$

Note that we design a more stable network by optimizing scale, shift, and disparity together rather than optimizing pixel-wise depth values individually. Per-frame foreground masks are obtained as same as DVS [13], and we directly parameterize our point-wise rigidness γ with 1 for the background and 0 for moving objects. Thus, we reconstruct neural point clouds as $\mathbb{P} = \{(\mathbf{p}_i, t_i, \mathbf{f}_i, \gamma_i) \mid i = 1, ..., L\}$, where each point i is located at $\mathbf{p}_i$ and captured at time steps t_i with a point-wise rigidness γ_i. We also use a neural feature vector $\mathbf{f}_i$, which are randomly initialized and parameterized to encode local scene representations. Since each neural point is a one-to-one match to each pixel of input frames, training the $\mathbf{p}_i$ and γ_i of each neural point optimizes the depth and foreground masks.

Dynamic Ray Marching. Our dynamic radiance fields are regressed from a different subset of the entire point cloud set $\mathbb{P}$ at each time step based on the sampling time and the point-wise rigidness. Specifically, we select neural points where their point-wise rigidness is higher than the threshold $\lambda = 0.5$, or its temporal location is the same as the sampling time as:

$$\mathbb{P}_t = \{(\mathbf{p}_i, t_i, \mathbf{f}_i, \gamma_i) \in \mathbb{P} \mid t_i = t \text{ or } \gamma_i > \lambda\}, \tag{4}$$

where neural points with γ_i is higher than λ to be background points to represent the static region whether the position of the dynamic object changes with each subset. Moreover, dynamic neural points do not represent the scene surface from different viewpoints, resulting in avoiding unnecessary ray sampling and not duplicating objects.

Neural Point Aggregation. After we select the subset of the neural point cloud, Point-DynRF aggregates neural points to output the density and radiance for each shading point. Specifically, we follow the Point-NeRF [46] to query $K = 8$ neighbor neural points for ray sampling, and we encode per-point local scene features with an MLP layer F for each shading point $\mathbf{x}$ as:

$$f_{i,\mathbf{x}} = F(\mathbf{f}_i, \mathbf{x} - \mathbf{p}_i). \tag{5}$$

Volume Density Regression. We use density regression MLP layers G_s and G_d for static and dynamic regions, respectively. We first encode per-point time-invariant volume density σ^s and time-varying volume density σ^d as:

$$\sigma^s_{i,\mathbf{x}} = G_s(f_{i,\mathbf{x}}), \tag{6}$$

$$\sigma^d_{i,\mathbf{x}} = G_d(f_{i,\mathbf{x}}, t). \tag{7}$$

Then, the time-invariant volume density $\sigma^s_{\mathbf{x}}$ and time-variant volume density $\sigma^d_{\mathbf{x}}$ at the sampling point $\mathbf{x}$ is regressed as:

$$\sigma^s_{\mathbf{x}} = \sum_i \sigma^s_{i,\mathbf{x}} \frac{w_i}{\sum w_i}, \tag{8}$$

$$\sigma^d_{\mathbf{x}} = \sum_i \sigma^d_{i,\mathbf{x}} \frac{w_i}{\sum w_i}, \tag{9}$$

where $w_i = \frac{1}{|p_i - x|}$ is for a distance-based weighted sum that gives higher weight to neural points closer to $\mathbf{x}$.

Radiance Regression. We regress a view-dependent radiance $r^s_{\mathbf{x}}$ and a time-dependent radiance $r^d_{\mathbf{x}}$ by using MLP layers R_s and R_d, respectively, as:

$$r^s_{\mathbf{x}} = R_s(\sum_i \frac{w_i}{\sum w_i} f_{i,\mathbf{x}}, d), \tag{10}$$

$$r^d_{\mathbf{x}} = R_d(\sum_i \frac{w_i}{\sum w_i} f_{i,\mathbf{x}}, t), \tag{11}$$

where d and t is the viewing direction and sampling time, respectively.

Blending Weight Regression. We directly regress blending weights from the point-wise rigidness γ_i of neighboring points as:

$$b_{\mathbf{x}} = \mathbb{1}[\sum_i (\frac{w_i}{\sum w_i}(1 - \gamma_i)) > \lambda], \tag{12}$$

where $\mathbb{1}$ equals to one if the condition is true. We define the blending weight as 0 or 1 so that either static or dynamic radiance fields dominate at each shading point. To optimize γ_i, we use the gradient clamping used in Point-NeRF to $\sum_i (\frac{w_i}{\sum w_i}(1 - \gamma_i))$ if MAX$(\sigma^s_{\mathbf{x}}, \sigma^d_{\mathbf{x}})$ is larger than a threshold 0.7 and there exists at least one dynamic point.

3.3. Training Objectiveness

In this section, we briefly demonstrate how we jointly optimize dynamic radiance fields and neural 3D point clouds. Specifically, we introduce reconstruction losses to learn combined NeRF, static NeRF, and dynamic NeRF in Sec. 3.3.1, scene geometry losses to reconstruct accurate neural points in Sec. 3.3.2 and joint optimization of Point-DynRF and neural 3D points in Sec. 3.3.3.

3.3.1 Reconstruction Loss

Combined NeRF We apply a reconstruction loss to dynamic radiance fields, which are a blend of view-dependent and time-dependent radiance fields. To this end, we combine two radiance fields with blending weights as:

$$\mathbf{C}(\mathbf{r}, t, \mathbb{P}_t) = \sum_{j=1}^{M} T_j \left(\alpha(\sigma_j^s \delta_j)(1-b_j) r_j^s + \alpha(\sigma_j^d \delta_j) b_j r_j^d\right), \tag{13}$$

$$T_j = \exp\left(-\sum_{k=1}^{j-1}((\sigma_k^s(1-b_k) + \sigma_k^d b_k)\delta_k)\right), \tag{14}$$

where $\mathbf{C}(\mathbf{r}, t, \mathbb{P}_t)$ is a volume rendered RGB value from a ray $\mathbf{r}(s) = \mathbf{o} + s\mathbf{d}$, rendering time t, and a subset point cloud $\mathbb{P}_t$. To ensure that the dynamic radiance fields accurately reconstruct the input video sequence, we jointly train view-/time-dependent radiance fields by applying a reconstruction loss L_{rec}^{full} as:

$$L_{rec}^{full} = \sum_{i=1}^{N} \sum_{uv} \|\mathbf{C}(\mathbf{r}_{uv}^i, i, \mathbb{P}_i) - I_{u,v}^i\|_2^2, \tag{15}$$

where $\mathbf{r}_{uv}^i$ is a ray for pixel coordinates (u, v) in i-th frame and $I_{u,v}^i$ is a ground-truth RGB value for pixel coordinates (u, v) in i-th frame.

Static and Dynamic NeRF We leverage point-based neural scene representations to learn time-invariant radiance fields (Static NeRF) and time-variant radiance fields (Dynamic NeRF), respectively. If we sample a subset point cloud $\mathbb{P}_{t,s}$ consisting of only background points as:

$$\mathbb{P}_{t,s} = \{(\mathbf{p}_i, t_i, \mathbf{f}_i, \gamma_i) \in \mathbb{P} \mid \gamma_i > \lambda\}, \tag{16}$$

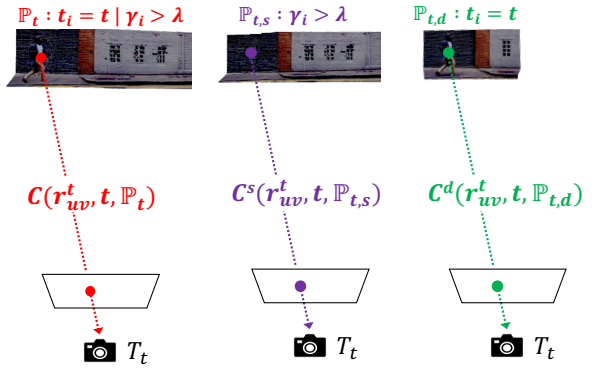
a volume rendered image contain only the background with no dynamic objects. Likewise, if we sample a subset point cloud $\mathbb{P}_{t,d}$ captured at a specific time t as:

$$\mathbb{P}_{t,d} = \{(\mathbf{p}_i, t_i, \mathbf{f}_i, \gamma_i) \in \mathbb{P} \mid t_i = t\}, \tag{17}$$

Point-DynRF can render an image restricted to only the neural points at that moment. Figure 3 shows which subset point clouds are used by combined NeRF, Static NeRF, and Dynamic NeRF. Thus, each radiance field is regressed by using Eq. 1 as:

$$\mathbf{C}^s(\mathbf{r}, t, \mathbb{P}_{t,s}) = \sum_{j=1}^{M} T_j^s(\alpha(\sigma_j^s \delta_j)) r_j^s, \tag{18}$$

$$\mathbf{C}^d(\mathbf{r}, t, \mathbb{P}_{t,d}) = \sum_{j=1}^{M} T_j^d(\alpha(\sigma_j^d \delta_j)) r_j^d. \tag{19}$$



(a) Combined NeRF (b) Static NeRF (c) Dynamic NeRF

Figure 3. **An overview of point cloud subsets for each NeRF.**

Then, we apply reconstruction losses to each radiance field as:

$$L_{rec}^s = \sum_{i,u,v} \|\mathbf{C}^s(\mathbf{r}_{uv}^i, t, \mathbb{P}_{t,s}) - I_{u,v}^i\|_2^2 * \mathbb{1}[M_{u,v}^i > \lambda], \tag{20}$$

$$L_{rec}^d = \sum_{i,u,v} \|\mathbf{C}^s(\mathbf{r}_{uv}^i, t, \mathbb{P}_{t,d}) - I_{u,v}^i\|_2^2, \tag{21}$$

where we only apply L_{rec}^s to background regions by using a foreground mask, and $\mathbb{1}[M_{u,v}^i > \lambda]$ indicates whether a rigidness value of pixel coordinates (u, v) in i-th frame is higher than the threshold λ. Finally, our reconstruction loss is formulated as:

$$L_{rec} = \lambda_{rec}^{full} L_{rec}^{full} + \lambda_{rec}^s L_{rec}^s + \lambda_{rec}^d L_{rec}^d. \tag{22}$$

3.3.2 Scene Geometry Loss

Initialized depth maps well represent the scene geometry but contain scale ambiguities with other frames. Therefore, we use optical flow maps f_{gt} from RAFT [38] to supervise scale s_t and shift b_t by applying a flow loss L_{flow} only for background pixels as:

$$[u', v', z']^T = T_{t'}^{-1} T_t D_t [u, v, 1]^T, \tag{23}$$

$$L_{flow} = \sum_{uv} \|(\frac{u'}{z'} - u, \frac{v'}{z'} - v) - f_{gt}\| * \mathbb{1}[M_{u,v}^i > \lambda], \tag{24}$$

where t' indicates a time step for adjacent frames and T_t is known camera parameters at t. Note that we detach the gradient from back-propagating to the disparity so that only the scale and shift values can be trained from the flow loss.

Moreover, we observe two cases that point-based ray marching can miss the ray as shown in Fig. 4. While learning the scene geometry, some pixels may have large depth values to satisfy the geometric consistency. As a result,

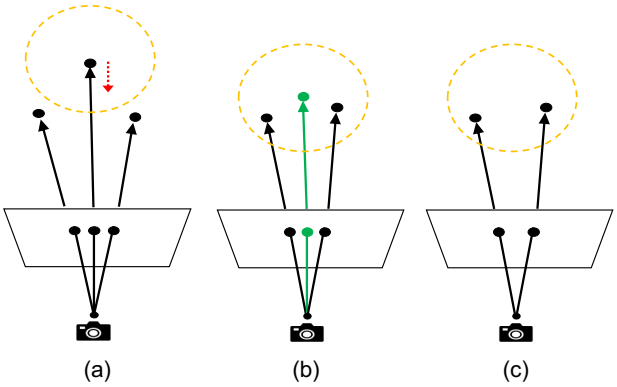


Figure 4. **Missing rays.** Assuming a ray is marched when it has three neighbor points for a shading point. (a) If the depth value is too large, the distance between neighboring pixels in 3D world coordinates is larger than the querying radius and fails to march the ray. Moreover, the subset point cloud is changed as the render time varies, causing rays can be marched (b) or sometimes not (c).

neighboring pixels in the image plane are also outside the query boundary, resulting in the ray can not be marched. Therefore, we introduce L^s_{miss}, which is an ℓ_2-loss to minimize the depth value corresponding to the pixel for which the ray is not marched. Also, rays may not be marched for different render times in the fixed-viewpoint due the dynamic ray sampling. To solve this problem, we introduce L^d_{miss}, which is also an ℓ_2-loss to maximize the rigidness of a green point in Fig. 4-(b) to be one. Note that L^s_{miss} and L^s_{miss} are introduced to deal with outlier cases, since missing rays are rarely present in the entire training process. Consequently, a scene geometry loss L_{geo} is formulated as:

$$L_{geo} = \lambda_{flow} L_{flow} + \lambda^s_{miss} L^s_{miss} + \lambda^d_{miss} L^d_{miss}. \quad (25)$$

3.3.3 Joint Optimization

We further introduce loss functions that optimize the dynamic radiance field and neural points together. Our joint optimization losses are formulated in the same manner as DVS [13]. However, we make a modification by introducing learnable per-frame depth and foreground masks, in contrast to the supervised learning approach of matching the volume-rendered depth $\tilde{\mathbf{D}}(\mathbf{r}, t, \mathbb{P}_t)$ and dynamicsness map $\tilde{\mathbf{M}}(\mathbf{r}, t, \mathbb{P}_t)$ to the initialized depth and foreground mask.

Depth Adjust Loss We apply a depth adjust loss L_{depth} to train the depth map of i-th frame D_i to match the expected depth $\tilde{\mathbf{D}}(\mathbf{r}, t, \mathbb{P}_t)$ as:

$$L_{depth} = \sum_{i=1}^{N} \sum_{uv} \|\tilde{\mathbf{D}}(\mathbf{r}^i_{uv}, i, \mathbb{P}_i) - D^i_{u,v}\|^2_2, \quad (26)$$

where $D^i_{u,v}$ is a depth value of pixel coordinates (u, v) in i-th frame.

Mask Adjust Loss Similar to expected depth maps, we use volume rendering for the blending weight to get the dynamicness map $\tilde{\mathbf{M}}(\mathbf{r}^i_{uv}, i, \mathbb{P}_i)$ and propose a mask adjust loss L_{mask} to match the per-frame foreground mask.

$$L_{mask} = \sum_{i=1}^{N} \sum_{uv} \|\tilde{\mathbf{M}}(\mathbf{r}^i_{uv}, i, \mathbb{P}_i) - M^i_{u,v}\|^2_2. \quad (27)$$

4. Experiments

4.1. Experimental Settings

Dataset. We evaluate our method on the Dynamic Scene Dataset [47]. We also use the same evaluation protocol in DVS [13], which evaluate the quality of the synthesized novel views through PSNR, SSIM and LPIPS metrics with ground truth images. Note that we exclude the Umbrella sequences since COLMAP estimates inaccurate camera poses, failing to regress the scene geometry accurately. Instead, we evaluate our method on several causally captured monocular video clips, which are more realistic videos and have a wide range of camera trajectories. Causally captured videos provide various scene contexts that can be happened in real-world scenarios, and COLMAP accurately estimates camera poses.

4.2. Comparison to Baselines

We now compare our method with the state-of-the-art methods on the NVIDIA Dynamic Scene dataset [47]. Table 1 shows quantitative results, and Point-DynRF demonstrates competitive performance with previous methods across most scenes. Specifically, Point-DynRF outperforms all previous methods on the SSIM metric for all scenes. However, due to the inaccurate camera pose estimated by COLMAP, the construction of neural points in Point-DynRF is not optimal. As a result, the rendered position of the dynamic object by Point-DynRF may differ from the ground truth. In the Playground scene depicted in Fig. 5, Point-DynRF generates a visually pleasing view but the position of the object is slightly shifted behind compared to the ground-truth. In the Skating scene, Point-DynRF generate realistic images, while flow-based methods like NSFF [22], DVS [13], and RoDynRF [25] produce blurry images, and deformable NeRFs such as HyperNeRF [31] and TiNeuVox [12] struggle to represent the scene.

4.3. Long-Term View Synthesis

We evaluate our method and flow-based dynamic view synthesis methods DVS [13], RoDynRF [25] and DynIBaR [23] on real-world scenarios with a wide-range camera

Table 1. **Quantitative results on NVIDIA Dynamic Scene dataset [47].** Image quality is measured by PSNR and LPIPS. Furthermore, we show the average performance over all view changes at the end. Best results in each metric are in **bold**, and second best are underlined.

Methods	PSNR(↑) / SSIM(↑) / LPIPS(↓)						
	Jumping	Skating	Truck	Balloon1	Balloon2	Playground	Avg
NeRF [27] + time	16.6 / 0.42 / 0.48	19.1 / 0.46 / 0.54	17.1 / 0.39 / 0.40	17.5 / 0.40 / 0.29	19.8 / 0.54 / 0.22	13.7 / 0.18 / 0.44	17.3 / 0.40 / 0.40
D-NeRF [32]	21.0 / 0.68 / 0.21	20.8 / 0.62 / 0.35	22.9 / 0.71 / 0.15	18.0 / 0.44 / 0.28	19.8 / 0.52 / 0.30	19.4 / 0.65 / 0.17	20.4 / 0.59 / 0.24
NR-NeRF [39]	19.4 / 0.61 / 0.29	23.2 / 0.72 / 0.23	18.8 / 0.44 / 0.45	17.0 / 0.34 / 0.35	22.0 / 0.70 / 0.21	14.3 / 0.19 / 0.33	19.2 / 0.50 / 0.33
HyperNeRF [31]	17.1 / 0.45 / 0.32	20.6 / 0.58 / 0.19	19.4 / 0.43 / 0.21	12.8 / 0.13 / 0.56	15.4 / 0.20 / 0.44	12.3 / 0.11 / 0.52	16.3 / 0.32 / 0.37
TiNeuVox [12]	19.7 / 0.60 / 0.26	21.9 / 0.68 / 0.16	22.9 / 0.63 / 0.19	16.2 / 0.34 / 0.37	18.1 / 0.41 / 0.29	12.6 / 0.14 / 0.46	18.6 / 0.47 / 0.29
NSFF [22]	23.9 / 0.80 / 0.15	28.8 / 0.88 / 0.13	25.4 / 0.76 / 0.17	21.5 / 0.69 / 0.22	23.8 / 0.73 / 0.23	20.8 / 0.70 / 0.22	24.1 / 0.76 / 0.18
DVS [13]	23.4 / 0.83 / 0.10	**31.9** / 0.94 / **0.04**	27.9 / 0.86 / 0.09	21.6 / 0.75 / **0.11**	**26.6** / 0.85 / **0.05**	23.7 / 0.85 / 0.08	**25.9** / 0.85 / 0.08
RoDynRF [25]	**24.3** / 0.84 / **0.08**	27.5 / 0.93 / 0.06	28.3 / 0.89 / **0.07**	21.4 / 0.76 / **0.11**	25.6 / 0.84 / 0.06	**24.3** / 0.89 / **0.05**	25.2 / 0.86 / **0.07**
Point-DynRF (Ours)	23.6 / **0.90** / 0.14	29.6 / **0.96** / **0.04**	**28.5** / **0.94** / 0.08	**21.7** / **0.88** / 0.12	26.2 / **0.92** / 0.06	22.2 / **0.91** / 0.09	25.3 / **0.92** / 0.08

Figure 5. **Comparison to baselines on NVIDIA Dynamic Scene Dataset [47].**

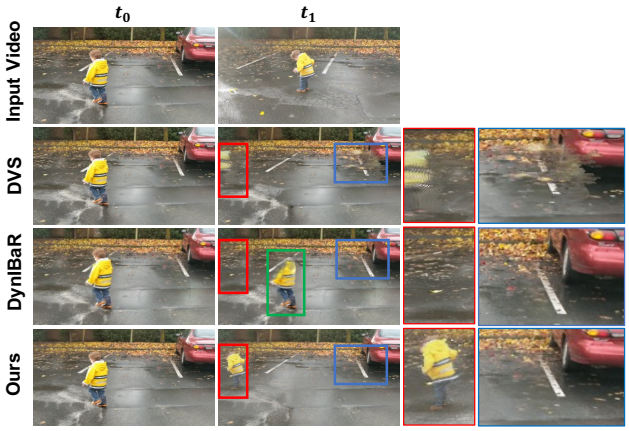


Figure 6. **Long-Term Novel View Generation.** For a fixed camera viewpoint at time t_0, the first column shows the novel view at time t_0 and the second column shows the novel view at time t_1. DynIBaR is over-fitted on the input camera trajectory (green box)

trajectory. Point-DynRF can generate realistic novel views for viewpoints far from the input camera trajectory in both space and time because it leverages global scene geometry (i.e., neural 3D points). Figure 6 shows the long-term view synthesis results where DVS [13] has quickly degenerated for unseen viewing directions and produces artifacts in the background regions. Moreover, DynIBaR [23] fails to generate a dynamic object since it is highly over-fitted on the input trajectory. On the other hand, our Point-DynRF effectively captures both the moving object and the background.

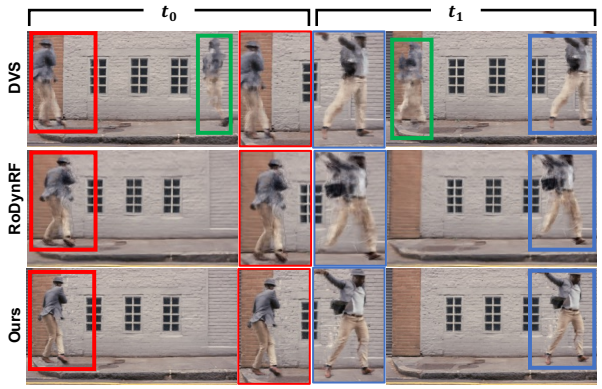


Figure 7. **Extremely Wide-Range Camera Trajectory.** DVS produces duplicated dynamic objects in distant spatio-temporal locations from the input camera trajectory (green boxes). Moreover, previous methods are quickly degenerated at the spatio-temporally distant viewpoints.

We also observe that DVS [13] infinitely duplicates the moving object and RoDynRF [25] is also degenerated when a camera moves in an extremely single direction, as shown in Fig. 7. This is due to geometric constraints focused on the input camera trajectory, which fails to represent the global scene geometry. Notably, our Point-DynRF generates more detailed dynamic regions as well as static background regions. These results confirm the superiority of our dynamic ray sampling and joint optimization scheme, which incorporates the entire scene geometry.

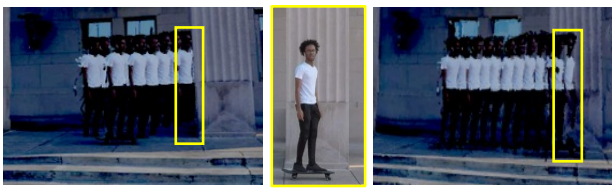
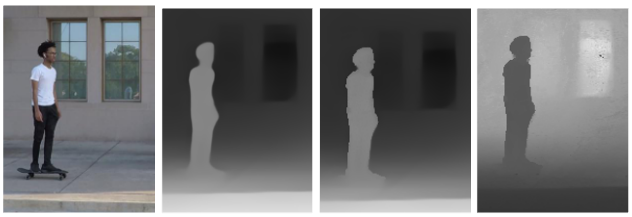


Figure 8. **Refinement of scale ambiguity.** The initial point cloud may not capture the complete scene geometry, but after optimization, the refined point cloud is free from scale ambiguity.



(a) (b) (c) (d)

Figure 9. **Refinement of depth map.** For a input frame (a) and initialized disparity map (b), optimized disparity map (c) well represent the boundary of a dynamic object. Also, expected depth map (d) is well aligned with the disparity map.

4.4. Effect of Accurate Scene Geometry

To verify the effectiveness of our proposed losses for training the scene geometry, we visualize initialized and refined point clouds as well as a depth map on the Skating scene as shown in Fig. 8-9. The results demonstrate that our joint optimization effectively regresses the scene geometry and address the scale ambiguity problem in monocular videos, resulting in a dynamic radiance field that accurately reflects this geometry.

4.5. Training and Rendering Time

Table 2 shows the training and rendering time on NVIDIA Dynamic Scene Dataset [47] for recent dynamic view synthesis methods. Since Point-DynRF avoids the unnecessary ray marching for empty space, the training process converges faster, leading to a reduction in overall training time. In the rendering process, however, searching neighbor neural points for each shading point requires additional computational costs, and the rendering time is slower than DVS [13] and RoDynRF [25].

Table 2. **Comparison of Training and Rendering Time.** Methods denoted by † refer to reported performance in the paper.

Method	Training (GPU hours)	Rendering (s/img)
HyperNeRF [31]	32	15
DVS [13]	36	8
RoDynRF† [25]	28	8
DynIBaR† [23]	48	20
Ours	20	11

Table 3. **Ablation Study of our proposed losses.** We report the PSNR, SSIM and LPIPS on the average of NVIDIA Dynamic Scene Dataset [47].

	PSNR (↑)	SSIM (↑)	LPIPS (↓)
Ours w/o $\mathbb{P}_t$	23.62 **(-1.68)**	0.755 **(-0.161)**	0.148 **(+0.067)**
Ours w/o L^s_{rec}	24.38 **(-0.92)**	0.843 **(-0.073)**	0.121 **(+0.040)**
Ours w/o L^d_{rec}	25.08 **(-0.22)**	0.901 **(-0.015)**	0.097 **(+0.016)**
Ours w/o L_{flow}	24.13 **(-1.07)**	0.872 **(-0.042)**	0.099 **(+0.018)**
Ours w/o L_{depth}	24.45 **(-0.85)**	0.856 **(-0.070)**	0.100 **(+0.019)**
Ours w/o L_{mask}	24.66 **(-0.64)**	0.884 **(-0.032)**	0.111 **(+0.030)**
Ours	**25.30**	**0.916**	**0.081**

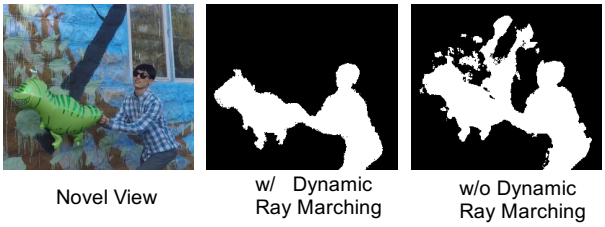


Figure 10. **Qualitative Ablation of Dynamic Ray Marching.** Without the dynamic ray marching, dynamic points at other times interfere with dynamic radiance fields at the rendering time.

4.6. Ablation Study for Point-DynRF Design

We conduct an ablation study for each component of Point-DynRF as shown in Table 3. The results show the quantitative results, and we verify all components contribute to the design of our Point-DynRF. Especially from the results on L_{flow} and L_{depth}, we confirm that the accuracy of the neural points has a direct impact on the performance of dynamic radiance fields. Also, the results on $\mathbb{P}_t$ confirm that our dynamic ray marching scheme significantly improves the performance. Dynamic ray marching ensures that the dynamicsness map for the novel view is well matched to the actual scene, as shown in Fig. 10.

5. Conclusion

We propose a novel framework called point-based dynamic radiance fields for long-term dynamic view synthesis from monocular videos. In our approach, we employ neural point clouds to encode geometric information and dynamic radiance fields to handle the volume rendering process. Our framework, Point-DynRF, optimizes the neural point clouds and dynamic radiance fields jointly, leveraging direct regression from neural 3D points. This allows us to effectively utilize the global scene geometry, which sets our method apart from previous approaches relying on correspondences between neighboring frames, limiting their ability to incorporate the overall scene geometry. We believe that our work contributes significantly to the field of dynamic view synthesis, enabling realistic rendering in various real-world scenarios.

A. Overview

In this supplementary material, we further demonstrate our experimental setup and provide additional results that the scene geometry is well regressed. First, we explain the total loss formulation in our training process in Sec. B. Then, we describe implementation details with image near-far bound determination by neural points in Sec. C and provide additional results for dynamicsness map of novel views in Sec. D. Finally, we demonstrate failure cases in Sec. E.

B. Losses

Our optimization process involves utilizing the loss functions L_{rec}, L_{geo}, L_{depth}, and L_{mask}. These loss functions are either modifications of those used in DVS [13] or newly introduced in this paper. To train Point-DynRF more stable, we also incorporate with a depth order loss L_{order} introduced in DVS [13] and a sparsity loss L_{sparse} introduced in Point-NeRF [46].

Depth Order Loss While the depth adjust loss helps optimize the overall scene geometry, there are inherent challenges in accurately determining the distance between dynamic objects and the background. Therefore, we use depth order loss L_{order} to allow the dynamic radiance fields to be regularized via a frame-by-frame depth map. Since regularizing the dynamic radiance fields with per-frame depth maps has scale and shift ambiguities as mentioned earlier, we leverage the volume rendering process of Dynamic NeRF to propose L_{order} as:

$$L_{order} = \sum_{i=1}^{N} \sum_{uv} \|\tilde{\mathbf{D}}(\mathbf{r}_{uv}^{i}, i, \mathbb{P}_i) - \tilde{\mathbf{D}}^{d}(\mathbf{r}_{uv}^{i}, i, \mathbb{P}_{i,d})\|_2^2. \quad (28)$$

Sparsity Loss Following the point-based representation, we apply a sparsity loss L_{sparse} on the point-wise rigidness to enforce it to be close to zero or one as:

$$L_{sparse} = \sum_{i} (\log(\gamma_i) + \log(1 - \gamma_i)). \quad (29)$$

Total Training Loss Formulation We formulate a reconstruction loss L_{rec}, a scene geometry loss L_{geo}, a depth adjust loss L_{depth}, a depth order loss L_{order}, a mask adjust loss L_{mask} and a sparsity loss L_{sparse}, to train our Point-DynRF and neural points. Specifically, we define $\lambda_{rec}^{full} = 3$, $\lambda_{rec}^{s} = 1$, $\lambda_{rec}^{d} = 1$ for the reconstruction loss. For the scene geometry loss, we define $\lambda_{flow} = 0.1$, $\lambda_{miss}^{s} = 1$, $\lambda_{miss}^{d} = 1$. Finally, we define $\lambda_{depth} = 0.1$, $\lambda_{order} = 0.1$, $\lambda_{mask} = 0.1$, and $\lambda_{sparse} = 0.0002$ to formulate the final loss as:

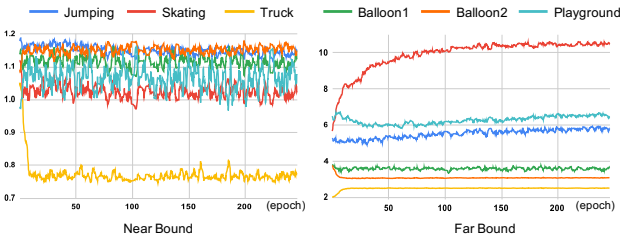


Figure 11. **Image Near-Far Bound Determination.**

$$L_{total} = L_{rec} + L_{geo} + \lambda_{depth} L_{depth} + \lambda_{order} L_{order} + \lambda_{mask} L_{mask} + \lambda_{sparse} L_{order}.$$

C. Implementation Details.

We randomly sampled 1024 rays in a batch, and each ray was assigned up to 32 sampling points. We used COLMAP to estimate the camera poses and resized all images into a resolution of 480×272. Also, we initialized our scale and shift parameters by using near and far bounds from COLMAP. We trained Point-DynRF for $250k$ iterations, and training takes about 20 hours on a single NVIDIA Geforce RTX 3090 GPU.

Near-Far Boundary Determination As our Point-DynRF is built on Point-NeRF [46] representation, dynamic radiance fields are regressed in 3D world coordinates, not in NDC space used by previous methods. Moreover, we need to render the far background as well, so we set the image near-far bound dynamically associated with the neural points. Specifically, we set the image near boundary to be the depth for the nearest neural point multiplied by 0.9, and the image far boundary to be the depth for the farthest neural point multiplied by 1.1. Figure 11 shows the convergence of the image near-far boundary of the scenes in the Dynamic Scene Dataset [47] during training. This result confirms that the scene geometry is stably trained and refined the initialized scene geometry well.

D. Additional Results

Additional Qualitative Results. We further provide additional qualitative results on Dynamic Scene Dataset [47] . Point-DynRF generates more realistic images compared to previous methods, and the human face in the third row of Fig. 12 confirms that Point-DynRF produces much sharper images, while other methods either fail to synthesize or produce blurry images. We also provide a video result of a causally captured monocular video that our Point-DynRF generates realistic images while the state-of-the-art method DVS [13] suffers from duplicated dynamic objects when rendering from a fixed viewpoint.

NeRF + time [27] NR-NeRF [39] NSFF [22] DVS [13] Ours Ground Truth

Figure 12. **Comparison to baselines on NVIDIA Dynamic Scene Dataset [47].**

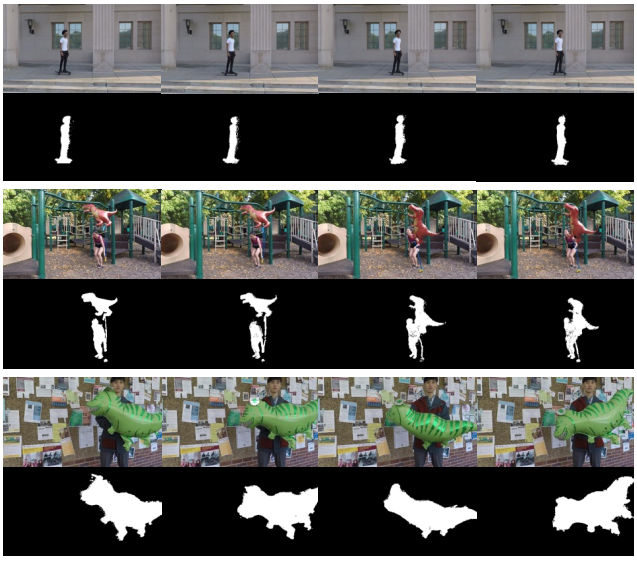


Figure 13. **Dynamicsness Maps for novel views.**

Our foreground masks $(M_1, \ldots, M_N)$ are also optimized during the training, so we provide dynamicsness maps for novel views, as shown in Fig 13. For each novel view, our Point-DynRF can render blending weights by using the volume rendering process. These dynamicsness maps for novel views confirm that our Point-DynRF well represents dynamic regions in the scene, and we can see that the static representation in the center of the person in the Playground Sequence is due to the fact that all the sequences in the input video for that region are learned as dynamic regions and represented as background by the miss ray marching scheme.

E. Failure Cases

While Point-DynRF optimizes well the ambiguous initial geometry and foreground masks, it fails to represent the scene if the neural point clouds are unnaturally initialized.

Figure 14. **Failure Case.**

A combination of inaccurate camera pose, depth map, and foreground masks sometimes unnaturally initialize neural point clouds where background points are closer to the camera than dynamic points as shown in Fig. 14. In this failure case, Point-DynRF falls short of distinguishing background points in front of the dynamic objects even addressing the scale ambiguity, and novel views also contain artifacts on these background points.

References

[1] Kara-Ali Aliev, Artem Sevastopolsky, Maria Kolos, Dmitry Ulyanov, and Victor Lempitsky. Neural point-based graphics. In *European Conference on Computer Vision*, pages 696–712. Springer, 2020. 2

[2] Benjamin Attal, Selena Ling, Aaron Gokaslan, Christian Richardt, and James Tompkin. Matryodshka: Real-time 6dof video view synthesis using multi-sphere images. In *European Conference on Computer Vision*, pages 441–459. Springer, 2020. 2

[3] 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/CVF Conference on Computer Vision and Pattern Recognition*, pages 5366–5375, 2020. 2

[4] Mark Boss, Raphael Braun, Varun Jampani, Jonathan T Barron, Ce Liu, and Hendrik Lensch. Nerd: Neural reflectance decomposition from image collections. In *Proceedings of*

the IEEE/CVF International Conference on Computer Vision, pages 12684–12694, 2021. 2

[5] Shengqu Cai, Anton Obukhov, Dengxin Dai, and Luc Van Gool. Pix2nerf: Unsupervised conditional p-gan for single image to neural radiance fields translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3981–3990, 2022. 2

[6] Joel Carranza, Christian Theobalt, Marcus A Magnor, and Hans-Peter Seidel. Free-viewpoint video of human actors. *ACM transactions on graphics (TOG)*, 22(3):569–577, 2003. 2

[7] Eric R Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5799–5809, 2021. 2

[8] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5939–5948, 2019. 2

[9] Paul Debevec, Yizhou Yu, and George Borshukov. Efficient view-dependent image-based rendering with projective texture-mapping. In *Eurographics Workshop on Rendering Techniques*, pages 105–116. Springer, 1998. 2

[10] 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*, pages 11–20, 1996. 2

[11] Yilun Du, Yinan Zhang, Hong-Xing Yu, Joshua B Tenenbaum, and Jiajun Wu. Neural radiance flow for 4d view synthesis and video processing. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14304–14314. IEEE Computer Society, 2021. 2

[12] Jiemin Fang, Taoran Yi, Xinggang Wang, Lingxi Xie, Xiaopeng Zhang, Wenyu Liu, Matthias Nießner, and Qi Tian. Fast dynamic radiance fields with time-aware neural voxels. In *SIGGRAPH Asia 2022 Conference Papers*, 2022. 1, 2, 3, 6, 7

[13] Chen Gao, Ayush Saraf, Johannes Kopf, and Jia-Bin Huang. Dynamic view synthesis from dynamic monocular video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5712–5721, 2021. 1, 2, 3, 4, 6, 7, 8, 9, 10

[14] Yuan-Chen Guo, Di Kang, Linchao Bao, Yu He, and Song-Hai Zhang. Nerfren: Neural radiance fields with reflections. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18409–18418, 2022. 2

[15] Marc Habermann, Weipeng Xu, Michael Zollhoefer, Gerard Pons-Moll, and Christian Theobalt. Livecap: Real-time human performance capture from monocular video. *ACM Transactions On Graphics (TOG)*, 38(2):1–17, 2019. 2

[16] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 2

[17] 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)*, September 2018. 2

[18] Rasmus Jensen, Anders Dahl, George Vogiatzis, Engin Tola, and Henrik Aanæs. Large scale multi-view stereopsis evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 406–413, 2014. 3

[19] Andreas Kurz, Thomas Neff, Zhaoyang Lv, Michael Zollhöfer, and Markus Steinberger. Adanerf: Adaptive sampling for real-time rendering of neural radiance fields. In *European Conference on Computer Vision*, pages 254–270. Springer, 2022. 2

[20] Marc Levoy and Pat Hanrahan. Light field rendering. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 31–42, 1996. 2

[21] Tianye Li, Mira Slavcheva, Michael Zollhoefer, Simon Green, Christoph Lassner, Changil Kim, Tanner Schmidt, Steven Lovegrove, Michael Goesele, Richard Newcombe, et al. Neural 3d video synthesis from multi-view video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5521–5531, 2022. 2

[22] 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/CVF Conference on Computer Vision and Pattern Recognition*, pages 6498–6508, 2021. 1, 2, 3, 6, 7, 10

[23] Zhengqi Li, Qianqian Wang, Forrester Cole, Richard Tucker, and Noah Snavely. Dynibar: Neural dynamic image-based rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4273–4284, 2023. 1, 2, 6, 7, 8

[24] Steven Liu, Xiuming Zhang, Zhoutong Zhang, Richard Zhang, Jun-Yan Zhu, and Bryan Russell. Editing conditional radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5773–5783, 2021. 2

[25] Yu-Lun Liu, Chen Gao, Andreas Meuleman, Hung-Yu Tseng, Ayush Saraf, Changil Kim, Yung-Yu Chuang, Johannes Kopf, and Jia-Bin Huang. Robust dynamic radiance fields. *arXiv preprint arXiv:2301.02239*, 2023. 1, 2, 3, 6, 7, 8

[26] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4460–4470, 2019. 2

[27] 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. *Communications of the ACM*, 65(1):99–106, 2021. 2, 3, 7, 10

[28] Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11453–11464, 2021. 2

[29] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning con-

tinuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 165–174, 2019. 2

[30] 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/CVF International Conference on Computer Vision*, pages 5865–5874, 2021. 3

[31] 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. *ACM Transactions on Graphics (TOG)*, 40(6):1–12, 2021. 1, 3, 6, 7, 8

[32] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10318–10327, 2021. 1, 3, 7

[33] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12179–12188, 2021. 2, 3, 4

[34] Steven M Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *2006 IEEE computer society conference on computer vision and pattern recognition (CVPR'06)*, volume 1, pages 519–528. IEEE, 2006. 2

[35] Vincent Sitzmann, Justus Thies, Felix Heide, Matthias Nießner, Gordon Wetzstein, and Michael Zollhofer. Deepvoxels: Learning persistent 3d feature embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2437–2446, 2019. 2

[36] 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/CVF Conference on Computer Vision and Pattern Recognition*, pages 175–184, 2019. 2

[37] Timo Stich, Christian Linz, Georgia Albuquerque, and Marcus Magnor. View and time interpolation in image space. In *Computer Graphics Forum*, volume 27, pages 1781–1787. Wiley Online Library, 2008. 2

[38] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *European conference on computer vision*, pages 402–419. Springer, 2020. 2, 3, 5

[39] Edgar Tretschk, Ayush Tewari, Vladislav Golyanik, Michael Zollhöfer, 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/CVF International Conference on Computer Vision*, pages 12959–12970, 2021. 1, 3, 7, 10

[40] Cen Wang, Minye Wu, Ziyu Wang, Liao Wang, Hao Sheng, and Jingyi Yu. Neural opacity point cloud. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(7):1570–1581, 2020. 2

[41] Chung-Yi Weng, Brian Curless, Pratul P Srinivasan, Jonathan T Barron, and Ira Kemelmacher-Shlizerman. Humannerf: Free-viewpoint rendering of moving people from monocular video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16210–16220, 2022. 2

[42] Suttisak Wizadwongsa, Pakkapon Phongthawee, Jiraphon Yenphraphai, and Supasorn Suwajanakorn. Nex: Real-time view synthesis with neural basis expansion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8534–8543, 2021. 2

[43] Minye Wu, Yuehao Wang, Qiang Hu, and Jingyi Yu. Multi-view neural human rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1682–1691, 2020. 2

[44] Tianhao Wu, Fangcheng Zhong, Andrea Tagliasacchi, Forrester Cole, and Cengiz Oztireli. D2nerf: Self-supervised decoupling of dynamic and static objects from a monocular video. *arXiv preprint arXiv:2205.15838*, 2022. 1, 3

[45] Wenqi Xian, Jia-Bin Huang, Johannes Kopf, and Changil Kim. Space-time neural irradiance fields for free-viewpoint video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9421–9431, 2021. 3

[46] Qiangeng Xu, Zexiang Xu, Julien Philip, Sai Bi, Zhixin Shu, Kalyan Sunkavalli, and Ulrich Neumann. Point-nerf: Point-based neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5438–5448, 2022. 2, 3, 4, 9

[47] 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/CVF Conference on Computer Vision and Pattern Recognition*, pages 5336–5345, 2020. 1, 2, 6, 7, 8, 9, 10

[48] Yu-Jie Yuan, Yang-Tian Sun, Yu-Kun Lai, Yuewen Ma, Rongfei Jia, and Lin Gao. Nerf-editing: geometry editing of neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18353–18364, 2022. 2

[49] 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

[50] 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. 2