MonoNeRF: Learning a Generalizable Dynamic Radiance Field from Monocular Videos

Fengrui Tian[1] Shaoyi Du[1] Yueqi Duan[2†]
[1]National Key Laboratory of Human-Machine Hybrid Augmented Intelligence,
National Engineering Research Center for Visual Information and Applications,
and Institute of Artificial Intelligence and Robotics, Xi'an Jiaotong University
[2]Department of Electronic Engineering, Tsinghua University
tianfr@stu.xjtu.edu.cn, dushaoyi@gmail.com, duanyueqi@tsinghua.edu.cn

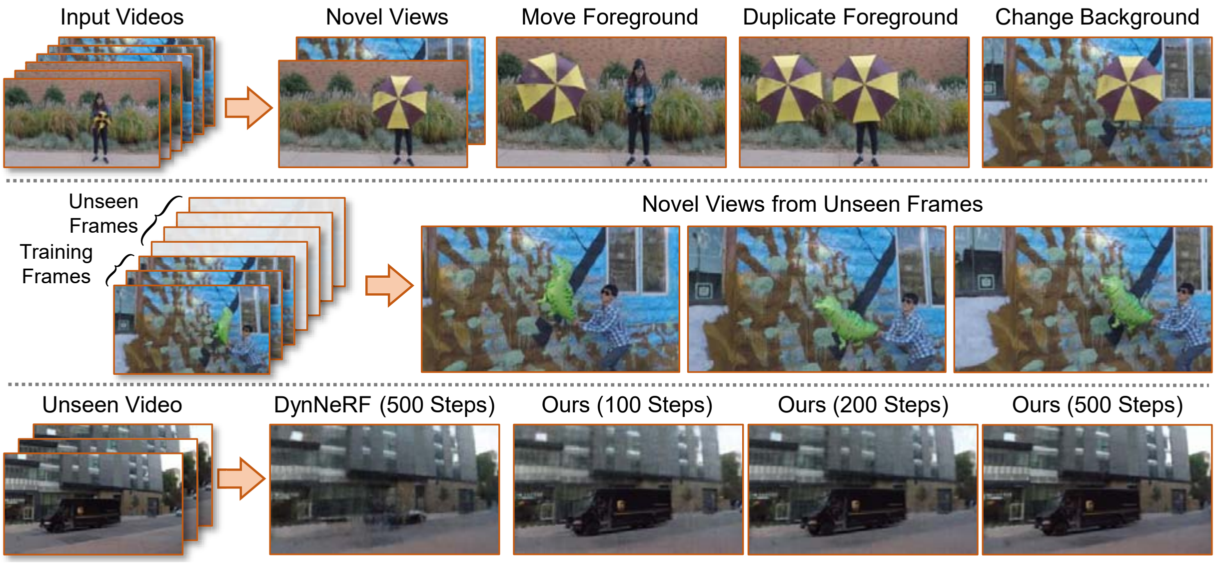

Figure 1: MonoNeRF learns a generalizable dynamic radiance field from multiple monocular videos. While video-based NeRF learns the field from positional encoding, we propose to learn from the extracted video features, which are generalizable to scenes. In this way, MonoNeRF supports scene editing applications (**Top**) and unseen frame synthesis (**Middle**). Compared with DynNeRF [13], MonoNeRF could adapt to novel scenes (**Bottom**) with hundreds of fine-tuning steps (about 10 minutes).

Abstract

In this paper, we target at the problem of learning a generalizable dynamic radiance field from monocular videos. Different from most existing NeRF methods that are based on multiple views, monocular videos only contain one view at each timestamp, thereby suffering from ambiguity along the view direction in estimating point features and scene flows. Previous studies such as DynNeRF disambiguate point features by positional encoding, which is not transferable and severely limits the generalization ability. As a result, these methods have to train one independent model for each scene and suffer from heavy computational costs when applying to increasing monocular videos in real-world applications. To address this, We propose MonoNeRF to simultaneously learn point features and scene flows with point trajectory and feature correspondence constraints across frames. More specifically, we learn an implicit velocity field to estimate point trajectory from temporal features with Neural ODE, which is followed by a flow-based feature aggregation module to obtain spatial features along the point trajectory. We jointly optimize temporal and spatial features in an end-to-end manner. Experiments show that our MonoNeRF is

†: Corresponding author.

able to learn from multiple scenes and support new applications such as scene editing, unseen frame synthesis, and fast novel scene adaptation. Codes are available at https://github.com/tianfr/MonoNeRF.

1. Introduction

Novel view synthesis [6] is a highly challenging problem. It facilitates many important applications in movie production, sports event, and virtual reality. The long standing problem recently has witnessed impressive progress due to the neural rendering technology [28, 26, 9]. Neural Radiance Field (NeRF) [28, 54, 22, 27, 46, 51, 52] shows that photo-realistic scenes can be represented by an implicit neural network. Concretely, taken as a query the position and viewing direction of the posed image, the network outputs the color of each pixel by volume rendering method. Among these approaches, it is supposed that the scene is static and can be observed from multiple views at the same time. Such assumptions are violated by numerous videos uploaded to the Internet, which usually contain dynamic foregrounds, recorded by the monocular camera.

More recently, some studies aim to explore how to learn dynamic radiance field from monocular videos [10, 13, 21, 49, 50, 40]. Novel view synthesis from monocular videos is a challenging task. As foreground usually dynamically changes in a video, there is ambiguity in the view direction to estimate precise point features and dense object motions (*i.e.,* scene flow [44]) from single views. In other words, we can only extract the projected 2D intra-frame local features and inter-frame optical flows, but fail to obtain precise 3D estimations. Previous works address this challenge by representing points as 4D position features (coordinate and time), so that the learned positional encoding provides specific information for each 3D point in the space [10, 13, 21, 49, 50, 40]. Based on positional encoding, these methods make efforts on exploiting scene priors [12, 48, 49] or adding spatio-temporal regularization [10, 13, 21, 50] to learn a more accurate dynamic radiance field.

However, while positional encoding successfully disambiguates 3D points from monocular 2D projections, it severely overfits to the training video clip and is not transferable. Therefore, existing positional encoding based methods have to optimize one independent model for each dynamic scene. With the fast increase of monocular videos in reality, they suffer from heavy computational costs and lengthy training time to learn from multiple dynamic scenes. Also, the lack of generalization ability limits further applications of scene editing which requires interaction among different scenes. A natural question is raised: can we learn a generalizable dynamic radiance field from monocular videos?

In this paper, we provide a positive answer to this question. The key challenge of this task is to learn to extract generalizable point features in the 3D space from monocular videos. While independently using 2D local features and optical flows suffers from ambiguity along the ray direction, they provide complementary constraints to jointly learn 3D point features and scene flows. On the one hand, for the sampled points on each ray, optical flow provides generalizable constraints that limit the relations of their point trajectories. On the other hand, for the flowing points on each estimated point trajectory, we consider that they should share the same point features. We estimate each point feature by aggregating their projected 2D local features, and design feature correspondence constraints to correct unreasonable trajectories.

To achieve this, we propose MonoNeRF to build a generalizable dynamic radiance field for multiple dynamic scenes. We hypothesize that a point moving along its trajectory over time keeps the consistent point feature. Our method concurrently predicts 3D point features and scene flows with point trajectory and feature correspondence constraints in monocular video frames. More specifically, we first propose to learn an implicit velocity field that encodes the speed from the temporal feature of each point. We supervise the velocity field with optical flow and integrate continuous point trajectories on the field with Neural ODE [5]. Then, we propose a flow-based feature aggregation module to sample spatial features of each point along the point trajectory. We incorporate the spatial and temporal features as the point feature to query the color and density for image rendering and jointly optimize point features and trajectories in an end-to-end manner. As shown in Figure 1, experiments demonstrate that our MonoNeRF is able to render novel views from multiple dynamic videos and support new applications such as scene editing, unseen frame synthesis, and fast novel scene adaption. Also, in the widely-used setting of novel view synthesis on training frames from single videos, our MonoNeRF still achieves better performance than existing methods despite that cross-scene generalization ability is not required in this setting.

2. Related Work

Novel view synthesis for static scenes. The long standing problem of novel view synthesis aims to construct new views of a scene from multiple posed images. Early works needed dense views captured from the scene [19, 14]. Recent studies have shown great progress by explicitly representing 3D scenes as neural representations [26, 28, 9, 7, 51, 25, 18, 38]. However, these methods train a separate model for each scene and need various training time for optimization. PixelNeRF [54] and MVSNeRF [4] proposed feature-based methods that directly render new scenes from the encoded features. Additionally, many researchers studied the generalization and decomposition abilities of novel

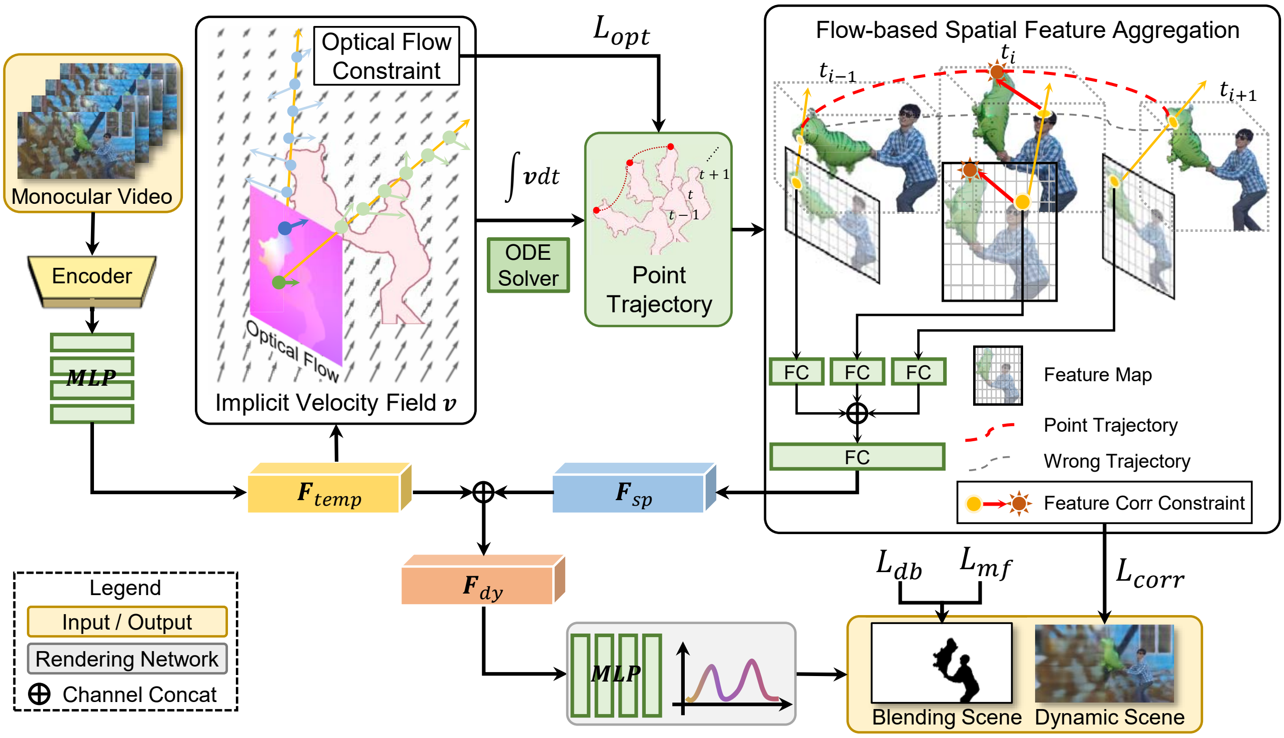

Figure 2: The overview of our generalizable dynamic field. We first exploit an implicit velocity field from the extracted temporal feature $\boldsymbol{F}_{temp}$. Then, we calculate point trajectory on the velocity field, and exploit the spatial feature $\boldsymbol{F}_{sp}$ with the proposed flow-based spatial feature aggregation module. We incorporate $\boldsymbol{F}_{temp}$ and $\boldsymbol{F}_{sp}$ as the point feature $\boldsymbol{F}_{dy}$ for rendering dynamic scene and design L_{opt} and L_{corr} to jointly optimize point features and trajectories.

view synthesis model [3, 37, 29, 15, 1, 18, 41]. Compared with these methods, our method studies the synthesis and generalization ability of dynamic scenes.

Space-time view synthesis from monocular videos. With the development of neural radiance field in static scenes, recent studies started to address the dynamic scene reconstruction problem [21, 50, 10, 20, 31, 32]. In monocular videos, a key challenge is that there exist multiple scene constructions implying the same observed image sequences. Previous approaches addressed this challenge by using 3D coordinates with time as point features and adding scene priors or spatio-temporal regularization. Scene priors such as shadow modeling [49], human shape prior [48, 23], and facial appearance [12] are object-specific and closer to special contents. Spatio-temporal regularization such as depth and flow regularization [10, 13, 21, 50] and object deformations [40] are more object-agnostic, but weaker in applying consistency restriction to dynamic foregrounds. In this paper, we study the challenge by constraining point features.

Scene flow for reconstruction. Scene flow firstly proposed by [44] depicts the motion of each point in the dynamic scene. Its 2D projection *i.e.,* optical flow contributes to many downstream applications such as video super-resolution [45], video tracking [42], video segmentation [43], and video recognition [2]. Several methods studied the scene flow estimation problem based on point cloud sequences [30, 24]. However, estimating scene flow from an observed video is an ill-posed problem. In dynamic scene reconstruction, previous works [21, 13] established discrete scene flow for pixel consistency in observation time. In this way, the flow consistency in observed frames can only be constrained, leading to ambiguity in non-observed time. Du *et al.* [10] proposed to build the continuous flow field in a dynamic scene. Compared to these methods, we study the generalization ability of the flow field in different scenes.

3. Approach

In this section, we introduce the proposed model that we term as MonoNeRF. We first present the overview of our model and then detail the approach of each part. Lastly, we introduce several strategies for optimizing the model.

3.1. MonoNeRF

MonoNeRF aims to model generalizable radiance field of dynamic scenes. Our method takes multiple monocular videos as input and uses optical flows, depth maps, and binary masks of foreground objects as supervision signals. Those signals could be generated by pretrained models automatically. We build the generalizable static and dynamic fields for backgrounds and foregrounds separately. For generalizable dynamic field, we suppose that spatio-temporal points flow along their trajectories and hold consistent features over time. We first extract video features by a CNN encoder and generate the temporal feature vectors based on the extracted features. Then, we build an implicit velocity field from the temporal feature vectors to estimate point trajectories. Next, we exploit the estimated point trajectories as indexes to find the local image patches on each frame and extract the spatial features from the patches with the proposed flow-based feature aggregation module. Different from NeRF [28] that uses scene-agnostic point positions to render image color, we aggregate the scene-dependent spatial and temporal features as the final point features to render foreground scenes with volume rendering. Finally, we design the optical flow and feature correspondence constraints for jointly learning point features and trajectories. For generalizable static field, we suppose backgrounds are static and each video frame is considered as a different view of the background. Since some background parts may be occluded by the foreground, we design an effective strategy to sample background point features. In the end, we combine the generalizable static and dynamic fields for rendering free-point videos.

3.2. Generalizable Dynamic Field

In this section, we introduce our generalizable dynamic field that renders novel views of dynamic foregrounds. We denote a monocular video as $\boldsymbol{V} = \{\boldsymbol{V}^i, i = 1, 2..., K\}$ consisting of K frames . For each frame $\boldsymbol{V}^i$, t_i is the observed timestamp and Π^i is the projection transform from the world coordinate to the 2D frame pixel coordinate. As Figure 2 shows, given a video $\boldsymbol{V}$, we exploit a video encoder E_{dy} to extract the video feature vector represented as $E_{dy}(\boldsymbol{V})$ and generate the temporal feature vector $\boldsymbol{F}_{temp}$ with a multiple layer perceptron (MLP) W_{temp}: $E_{dy}(\boldsymbol{V}) \to \boldsymbol{F}_{temp}$. We build the implicit velocity field based on $\boldsymbol{F}_{temp}$.

Implicit velocity field. We suppose points are moving in the scene and represent a spatio-temporal point as $\boldsymbol{p} = (\boldsymbol{x}_p, t_p)$ including the 3D point position $\boldsymbol{x}_p$ and time t_p. We define $\boldsymbol{\Phi}$ as the continuous point trajectory and $\boldsymbol{\Phi}(\boldsymbol{p}, t)$ denotes the position of point $\boldsymbol{p}$ at timestamp t. We further define the velocity field $\boldsymbol{v}$ which includes the 3D velocity of each point. The relationship between point velocity and trajectory is specified as follows,

$$\frac{\partial \boldsymbol{\Phi}(\boldsymbol{p}, t)}{\partial t} = \boldsymbol{v}\Big(\boldsymbol{\Phi}(\boldsymbol{p}, t), t\Big), \quad \text{s.t. } \boldsymbol{\Phi}(\boldsymbol{p}, t_p) = \boldsymbol{x}_p. \tag{1}$$

$\boldsymbol{v}$ is conditioned on $\boldsymbol{F}_{temp}$ and implemented by an MLP W_{vel}: $(\boldsymbol{F}_{temp}(\boldsymbol{V}), \boldsymbol{\Phi}, t) \to \boldsymbol{v}$. We calculate the point trajectory at an observed timestamp t_i with Neural ODE [5],

$$\boldsymbol{\Phi}(\boldsymbol{p}, t_i) = \boldsymbol{x}_p + \int_{t_p}^{t_i} \boldsymbol{v}\big(\boldsymbol{\Phi}(\boldsymbol{p}, t), t\big) dt. \tag{2}$$

We take $\boldsymbol{\Phi}(\boldsymbol{p}, t_i)$ as an index to query the spatial feature.

Flow-based spatial feature aggregation. We employ Π^i to project $\boldsymbol{\Phi}(\boldsymbol{p}, t_i)$ on $\boldsymbol{V}^i$ and find the local image patch indexed by the projected position $\Pi^i(\boldsymbol{\Phi}(\boldsymbol{p}, t_i))$. We extract the feature vector $\boldsymbol{F}^i_{sp}$ from the patch with the encoder E_{dy} and a fully connected layer fc_1,

$$\boldsymbol{F}^i_{sp}(\boldsymbol{V}; \boldsymbol{p}) = fc_1\Big(E_{dy}\big(\boldsymbol{V}^i; \Pi^i(\boldsymbol{\Phi}(\boldsymbol{p}, t_i))\big)\Big). \tag{3}$$

In practice, $E_{dy}\big(\boldsymbol{V}^i; \Pi^i(\boldsymbol{\Phi}(\boldsymbol{p}, t_i))\big)$ is implemented by using E_{dy} to extract the frame-wise feature map of $\boldsymbol{V}^i$ and sampling the feature vector at $\Pi^i(\boldsymbol{\Phi}(\boldsymbol{p}, t_i))$ with bilinear interpolation. The spatial feature vector $\boldsymbol{F}_{sp}$ is then calculated by incorporating $\{\boldsymbol{F}^1_{sp}, \boldsymbol{F}^2_{sp}, ..., \boldsymbol{F}^K_{sp}\}$ with a fully connected layer fc_2,

$$\boldsymbol{F}_{sp}(\boldsymbol{V}; \boldsymbol{p}) = fc_2\big(\boldsymbol{F}^1_{sp}(\boldsymbol{V}; \boldsymbol{p}), \boldsymbol{F}^2_{sp}(\boldsymbol{V}; \boldsymbol{p}), ..., \boldsymbol{F}^K_{sp}(\boldsymbol{V}; \boldsymbol{p})\big). \tag{4}$$

Finally, we incorporate $\boldsymbol{F}_{temp}$ and $\boldsymbol{F}_{sp}$ as the point feature vector $\boldsymbol{F}_{dy}$,

$$\boldsymbol{F}_{dy}(\boldsymbol{V}; \boldsymbol{p}) = concat\{\boldsymbol{F}_{temp}(\boldsymbol{V}), \boldsymbol{F}_{sp}(\boldsymbol{V}; \boldsymbol{p})\}. \tag{5}$$

We build the generalizable dynamic field based on $\boldsymbol{F}_{dy}$.

Dynamic foreground rendering. Taking a point $\boldsymbol{p}$ and its feature vector $\boldsymbol{F}_{dy}(\boldsymbol{V}; \boldsymbol{p})$ as input, our generalizable dynamic radiance field W_{dy}: $(\boldsymbol{p}, \boldsymbol{F}_{dy}(\boldsymbol{V}; \boldsymbol{p})) \to (\boldsymbol{c}_{dy}, \sigma_{dy}, b)$ predicts the volume density σ_{dy}, color $\boldsymbol{c}_{dy}$ and blending weight b of the point. We follow DynNeRF [13] and utilize b to judge whether a point belongs to static background or dynamic foreground. We exploit volume rendering to approximate each pixel color of an image. Concretely, given a ray $\boldsymbol{r}(u) = \boldsymbol{o} + u\boldsymbol{d}$ starting from the camera center $\boldsymbol{o}$ along the direction $\boldsymbol{d}$ through a pixel, its color is integrated by

$$\boldsymbol{C}_{dy}(\boldsymbol{r}) = \int_{u_n}^{u_f} T_{dy}(u)\sigma_{dy}(u)\boldsymbol{c}_{dy}(u)du, \tag{6}$$

where u_n, u_f are the bounds of the volume rendering depth range and $T_{dy}(u) = exp(-\int_{u_n}^{u} \sigma_{dy}(\boldsymbol{r}(s))ds$ is the accumulated transparency. We simplify $\boldsymbol{c}(u) = \boldsymbol{c}(\boldsymbol{r}(u), \boldsymbol{d})$ and $\sigma(u) = \sigma(\boldsymbol{r}(u))$ here and in the following sections. Next we present the optical flow and feature correspondence constraints that jointly supervise our model.

Optical flow constraint. We supervise $\boldsymbol{v}$ with the optical flow $\boldsymbol{f}^{gt}$. In practice, it can only approximate the backward flow $\boldsymbol{f}^{gt}_{bw}$ and forward flow $\boldsymbol{f}^{gt}_{fw}$ between two consecutive video frames [39]. We hence estimate the point backward and forward trajectory variations during the period that the point passes between two frames and calculate optical flows by integrating the trajectory variations of each point along camera rays. Formally, given a ray $\boldsymbol{r}(u)$ through a pixel on the frame $\boldsymbol{V}^i$ at time t_i, for each point on the ray *i.e.,* $\boldsymbol{p}(u) = (\boldsymbol{r}(u), t_i)$ the trajectory variations $\Delta\boldsymbol{\Phi}_{bw}$ back to t_{i-1} and $\Delta\boldsymbol{\Phi}_{fw}$ forward to t_{i+1} are obtained by the following equation,

$$\Delta\boldsymbol{\Phi}_{\{bw,fw\}}(\boldsymbol{p}(u)) = \int_{t_i}^{t_{\{i-1,i+1\}}} \boldsymbol{v}\Big(\boldsymbol{\Phi}(\boldsymbol{p}(u),t),t\Big)dt. \tag{7}$$

We follow previous works [21, 13] and exploit the volume rendering to integrate pseudo optical flows $\boldsymbol{f}_{bw}$ and $\boldsymbol{f}_{fw}$ by utilizing the estimated volume density σ_{dy} of each point,

$$\boldsymbol{f}_{\{bw,fw\}}(\boldsymbol{r}) = \int_{u_n}^{u_f} T_{dy}(u)\sigma_{dy}(u)\Delta\boldsymbol{\Phi}^i_{\{bw,fw\}}(\boldsymbol{p}(u))du, \tag{8}$$

where $\Delta\boldsymbol{\Phi}^i_{\{bw,fw\}} = \Pi^i(\Delta\boldsymbol{\Phi}_{\{bw,fw\}})$ denotes we use Π^i to project 3D trajectory variations on the image plane of $\boldsymbol{V}^i$. We supervise the pseudo flows by the ground truth flows,

$$L_{opt} = \sum_{\boldsymbol{r}} \Big(\boldsymbol{f}_{\{bw,fw\}}(\boldsymbol{r}) - \boldsymbol{f}^{gt}_{\{bw,fw\}}(\boldsymbol{r})\Big). \tag{9}$$

In this way, the relation of point trajectories along a ray is limited by the optical flow supervision.

Feature correspondence constraint. According to Section 3.1, a point moving along the trajectory holds the same feature and represents the consistent color and density. For each ray $\boldsymbol{r}_{curr}$ at the current time t_i through a pixel of $\boldsymbol{V}^i$, we warp the ray from the neighboring observed timestamps t_{i-1} and t_{i+1} as $\boldsymbol{r}_{bw}$ and $\boldsymbol{r}_{fw}$ separately by using (7),

$$\boldsymbol{r}_{\{bw,fw\}}(u) = \boldsymbol{r}_{curr}(u) + \Delta\boldsymbol{\Phi}_{\{bw,fw\}}(\boldsymbol{p}_{curr}(u)), \tag{10}$$

where $\boldsymbol{p}_{curr}(u) = (\boldsymbol{r}_{curr}(u), t_i)$. We render the pixel color from the point features not only along the ray $\boldsymbol{r}_{curr}$ at the time t_i, but also along the wrapped rays $\boldsymbol{r}_{bw}$ at t_{i-1} and $\boldsymbol{r}_{fw}$ at t_{i+1}. The predicted pixel color $\boldsymbol{C}_{dy}$ is rendered by using (6) and supervised by the ground truth colors $\boldsymbol{C}^{gt}_{dy}$,

$$L_{\{bw,curr,fw\}} = \sum_{\boldsymbol{r}} ||\boldsymbol{C}_{dy}(\boldsymbol{r}_{\{bw,curr,fw\}}) - \boldsymbol{C}^{gt}_{dy}(\boldsymbol{r})||_2. \tag{11}$$

The feature correspondence constraint L_{corr} is defined as

$$L_{corr} = L_{bw} + L_{curr} + L_{fw}. \tag{12}$$

L_{corr} supervises the predicted color and point features $\boldsymbol{F}_{dy}$.

3.3. Generalizable Static Field

As mentioned in Section 3.1, for some background parts occluded by the changing foreground in the current view, their features implying the foreground cannot infer the correct background information. However, the occluded parts could be seen in non-occluded views with correct background features. To this end, given a point position $\boldsymbol{x}$, the background point feature vector $\boldsymbol{F}_{st}$ is produced by

$$\boldsymbol{F}_{st}(\boldsymbol{V};\boldsymbol{x}) = fc_3\Big(E_{st}(\boldsymbol{V}^*;\Pi^*(\boldsymbol{x}))\Big), \tag{13}$$

where fc_3 is a fully connected layer and E_{st} denotes the image encoder. $\boldsymbol{V}^*$ and Π^* are the non-occluded frame and corresponding projection transform respectively. $E_{st}(\boldsymbol{V}^*;\Pi^*(\boldsymbol{x}))$ denotes that we find the local image patch at $\Pi^*(\boldsymbol{x})$ and extract the feature vectors with an image encoder E_{st} similar to $E_{dy}\big(\boldsymbol{V}^i;\Pi^i(\boldsymbol{\Phi}(\boldsymbol{p},t_i))\big)$. Since there is no prior in which frames they can be exposed, we apply a straightforward yet effective strategy by randomly sampling one frame from the other frames in the video. We represent the static scene as a radiance field to infer the color $\boldsymbol{c}_{st}$ and density σ_{st} by using an MLP W_{st}: $(\boldsymbol{x},\boldsymbol{d},\boldsymbol{F}_{st}(\boldsymbol{V};\boldsymbol{x})) \rightarrow (\boldsymbol{c}_{st},\sigma_{st})$. The expected background color $\boldsymbol{C}_{st}$ is given by

$$\boldsymbol{C}_{st}(\boldsymbol{r}) = \int_{u_n}^{u_f} T_{st}(u)\sigma_{st}(u)\boldsymbol{c}_{st}(u)du, \tag{14}$$

where $T_{st}(t) = exp(-\int_{u_n}^{u}\sigma_{st}(\boldsymbol{r}(s))ds$. We employ foreground masks M to optimize the static field by supervising the pixel color in each video frame in the background regions (where $M(\boldsymbol{r}) = 0$),

$$L_{st} = \sum_{\boldsymbol{r}} ||(\boldsymbol{C}_{st}(\boldsymbol{r}) - \boldsymbol{C}^{gt}_{st}(\boldsymbol{r}))(1 - M(\boldsymbol{r}))||_2, \tag{15}$$

where $\boldsymbol{C}^{gt}_{st}$ represents the ground truth color.

3.4. Optimization

The final dynamic scene color combines the colors from the generalizable dynamic and static fields,

$$\boldsymbol{C}_{full}(\boldsymbol{r}) = \int_{u_n}^{u_f} T_{full}(u)\sigma_{full}(u)\boldsymbol{c}_{full}(u)du, \tag{16}$$

where

$$\sigma_{full}(u)\boldsymbol{c}_{full}(u) = (1-b)\sigma_{st}(u)\boldsymbol{c}_{st}(u) + b\sigma_{dy}(u)\boldsymbol{c}_{dy}(u), \tag{17}$$

and applies the reconstruction loss,

$$L_{full} = \sum_{\boldsymbol{r}} ||\boldsymbol{C}_{full}(\boldsymbol{r}) - \boldsymbol{C}^{gt}_{full}(\boldsymbol{r})||_2. \tag{18}$$

Next we design several strategies to optimize our model.

Point trajectory discretization. While point trajectory can be numerically estimated by the continuous integral (2) with Neural ODE solvers [5], it needs plenty of time for querying each point trajectory. To accelerate the process,

Table 1: Novel view synthesis on training frames from single videos. While this setting does not require cross-scene generalization ability, our MonoNeRF still achieves better performance.

PSNR ↑ / LPIPS ↓	Jumping	Skating	Truck	Umbrella	Balloon1	Balloon2	Playground	Average
NeRF [28]	20.58 / 0.305	23.05 / 0.316	22.61 / 0.225	21.08 / 0.441	19.07 / 0.214	24.08 / 0.098	20.86 / 0.164	21.62 / 0.252
NeRF [28] + time	16.72 / 0.489	19.23 / 0.542	17.17 / 0.403	17.17 / 0.752	17.33 / 0.304	19.67 / 0.236	13.80 / 0.444	17.30 / 0.453
Yoon *et al.* [53]	20.16 / 0.148	21.75 / 0.135	23.93 / 0.109	20.35 / 0.179	18.76 / 0.178	19.89 / 0.138	15.09 / 0.183	19.99 / 0.153
Tretschk *et al.* [40]	19.38 / 0.295	23.29 / 0.234	19.02 / 0.453	19.26 / 0.427	16.98 / 0.353	22.23 / 0.212	14.24 / 0.336	19.20 / 0.330
Li *et al.* [21]	24.12 / 0.156	28.91 / 0.135	25.94 / 0.171	22.58 / 0.302	21.40 / 0.225	24.09 / 0.228	20.91 / 0.220	23.99 / 0.205
NeuPhysics [33]	20.16 / 0.205	25.13 / 0.166	22.62 / 0.212	21.02 / 0.426	16.68 / 0.238	22.54 / 0.265	15.10 / 0.367	20.48 / 0.282
DynNeRF [13]	24.23 / 0.144	28.90 / 0.124	25.78 / 0.134	23.15 / **0.146**	21.47 / **0.125**	25.97 / 0.059	**23.65 / 0.093**	24.74 / 0.118
MonoNeRF	**24.26 / 0.091**	**32.06 / 0.044**	**27.56 / 0.115**	**23.62** / 0.180	**21.89** / 0.129	**27.36 / 0.052**	22.61 / 0.130	**25.62 / 0.106**

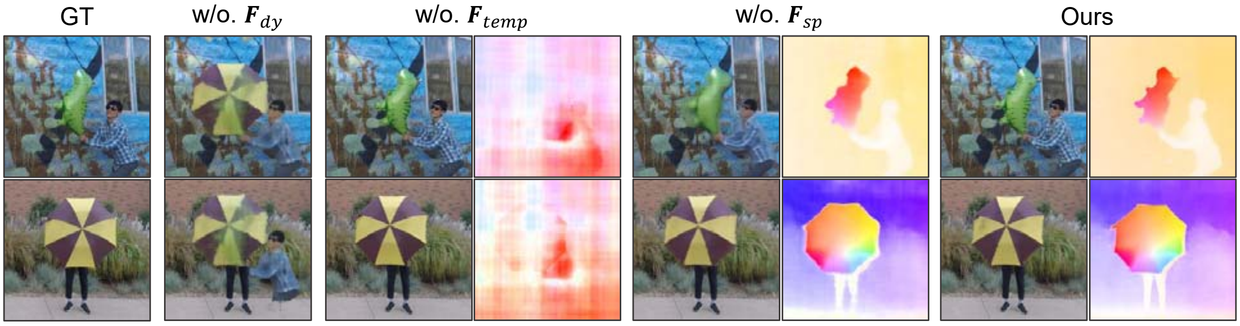

Figure 3: Qualitative results of jointly optimizing Balloon2 and Umbrella scenes. The foregrounds of two scenes are mixed without $\boldsymbol{F}_{dy}$. Our method renders more accurate novel views and predicts plausible scene flows (listed beside the RGB images) by incorporating $\boldsymbol{F}_{temp}$ and $\boldsymbol{F}_{sp}$.

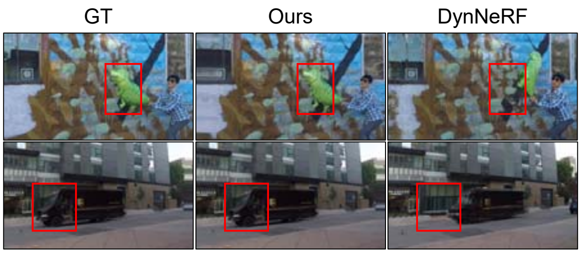

Figure 4: Novel view synthesis on unseen frames. Compared to the ground truths, our model could transfer to new motions, whereas DynNeRF [13] only interpolates in the training frames.

we propose to partition $[t_p, t_i]$ into N evenly-spaced bins and suppose the point velocity in each bin is a constant. The point trajectory can be estimated by the following equation,

$$\boldsymbol{\Phi}(\boldsymbol{p}, t_i) = \boldsymbol{x}_p + \sum_{n=1}^{N} \boldsymbol{v}\big(\boldsymbol{\Phi}(\boldsymbol{p}, t+\Delta t), t+\Delta t\big)\Delta t, \quad (19)$$

where $\Delta t = \frac{n}{N}(t_i - t_p)$.

Depth-blending restriction. To calculate the blending weights within a foreground ray in the generalizable dynamic field, only the points in close proximity to the esti-

Table 2: Quantitative results of novel view synthesis on unseen frames. We used the first four frames for training and tested the performance on the rest eight frames.

Balloon2 / Truck	PSNR ↑	SSIM ↑	LPIPS ↓
NeRF [28]	20.33 / 20.26	0.662 / 0.669	0.224 / 0.256
NeRF [28] + time	20.22 / 20.26	0.661 / 0.639	0.218 / 0.256
NeuPhysics [33]	19.45 / 20.24	0.478 / 0.517	0.343 / 0.285
DynNeRF [13]	19.99 / 20.33	0.641 / 0.621	0.291 / 0.273
MonoNeRF	**21.30 / 23.74**	**0.669 / 0.702**	**0.204 / 0.174**

mated ray depth are regarded as foreground points, whereas points beyond this range are excluded. We penalize the blending weights of non-foreground points in foreground rays. Specifically, given a ray $\boldsymbol{r}(u)$ through a pixel and the pixel depth u_d, We penalize the blending weights of the points on the ray out of the interval $(u_d - \epsilon, u_d + \epsilon)$,

$$L_{db} = \sum_{u \in (u_n, u_d - \epsilon) \cup (u_d + \epsilon, u_f)} ||b(\boldsymbol{r}(u))||_2, \quad (20)$$

where $b(\boldsymbol{r}(u))$ is the blending weight at the position $\boldsymbol{r}(u)$. ϵ controls the surface thickness of the dynamic foreground.

Mask flow loss. We further constrain the consistency of point features by minimizing blending weight variations

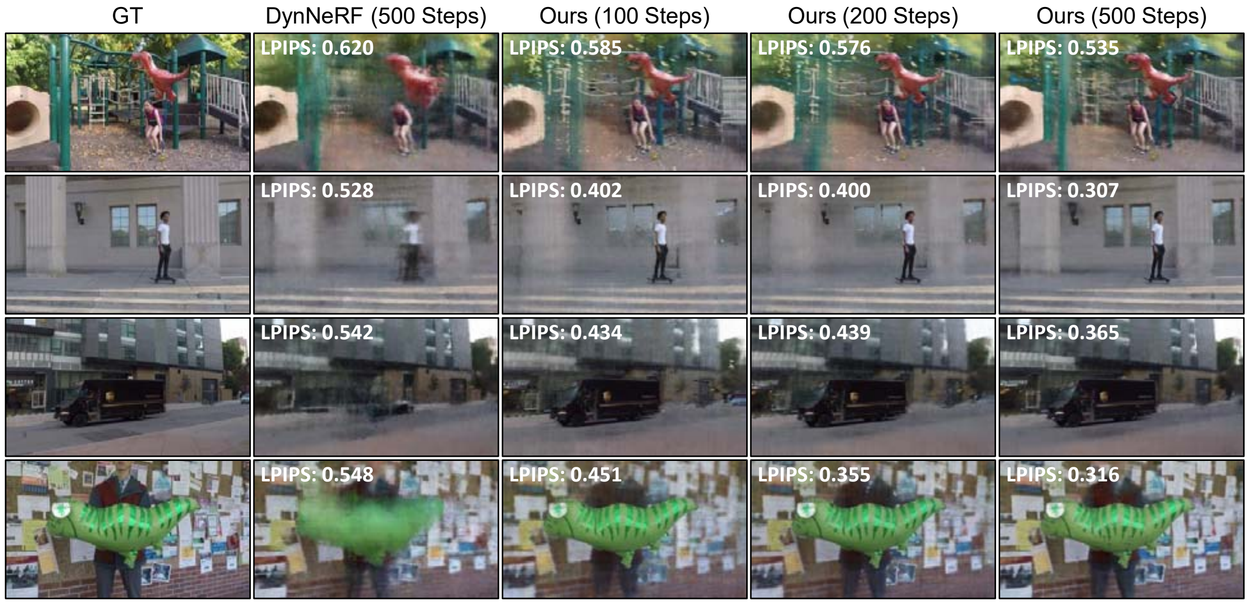

Figure 5: Fast novel scene adaption. We used the pretrained model on Balloon2 scene to fine-tune the novel views of unseen dynamic scenes. We employed LPIPS [55] (the lower is better) to evaluate the image similarity, which provides more correlation with human judgment than other indexes.

along trajectories,

$$L_{mf} = \sum_{i,j \in \{1,2,..,K\}} ||b(\mathbf{\Phi}(\boldsymbol{p}, t_i)) - b(\mathbf{\Phi}(\boldsymbol{p}, t_j))||_2, \quad (21)$$

where $b(\mathbf{\Phi}(\boldsymbol{p}, t))$ denotes the blending weight at $\mathbf{\Phi}(\boldsymbol{p}, t)$.

Other regularization. We follow previous works [21, 13, 10] to use the depth constraint, sparsity constraint, and motion regularization and smoothness to train the model.

Table 3: Ablation studies on $\boldsymbol{F}_{dy}$, $\boldsymbol{F}_{temp}$, and $\boldsymbol{F}_{sp}$ by jointly optimizing Balloon2 and Umbrella scenes.

PSNR ↑ / LPIPS ↓	Umbrella	Balloon2	Average
w/o. $\boldsymbol{F}_{dy}$	20.59 / 0.256	22.79 / 0.159	21.57 / 0.230
w/o. $\boldsymbol{F}_{temp}$	22.75 / 0.175	24.85 / 0.098	23.80 / 0.137
w/o. $\boldsymbol{F}_{sp}$	22.81 / 0.181	25.09 / 0.143	23.95 / 0.162
Ours	**23.44 / 0.169**	**25.44 / 0.093**	**24.44 / 0.131**

4. Experiments

In this section, we conducted experiments on the Dynamic Scene dataset [53]. We first tested the performance of synthesizing novel views from single videos, and then we tested the generalization ability from multiple videos. In the end, we carried out ablation studies on our model.

4.1. Experimental Setup

Dataset. We used the Dynamic Scene dataset [53] to evaluate the proposed method. Concretely, it contains 9 video sequences that are captured with 12 cameras by using a camera rig. We followed previous works [21, 13] and derived each frame of the video from different cameras to simulate the camera motion. All the cameras capture images at 12 different timestamps $\{t_i, i = 1, 2..., 12\}$. Each training video contains twelve frames sampled from the i^{th} camera at time t_i followed DynNeRF [13]. We used COLMAP [35, 36] to approximate the camera poses. It is assumed the intrinsic parameters of all the cameras are the same. DynamicFace sequences were excluded because COLMAP fails to estimate camera poses. All video frames were resized to 480×270 resolution. We generated the depth, mask, and optical flow signals from the depth estimation model [34], Mask R-CNN [16], and RAFT [39].

Implementation details. We followed pixelNeRF [54] and used ResNet-based MLPs as our implicit velocity field W_{vel} and rendering networks W_{dy} and W_{st}. For generalizable dynamic field, we utilized Slowonly50 [11] pretrained on Kinetics400 [2] dataset as the encoder E_{dy} with the frozen weights. We removed the final fully connected layer in the backbone and incorporated the first, second, and third feature layers for querying $\boldsymbol{F}_{sp}$. We simplified (4) that $\boldsymbol{F}_{sp}$ at time t_i is sampled from $\{\boldsymbol{V}^{i-1}, \boldsymbol{V}^{i}, \boldsymbol{V}^{i+1}\}$. For generalizable static field, we used ResNet18 [17] pretrained on ImageNet [8] as the encoder E_{st}. We extracted a feature pyramid from video frames for querying $\boldsymbol{F}_{st}$. The vector sizes of $\boldsymbol{F}_{temp}$, $\boldsymbol{F}_{sp}$, $\boldsymbol{F}_{dy}$, and $\boldsymbol{F}_{st}$ are 256. Please

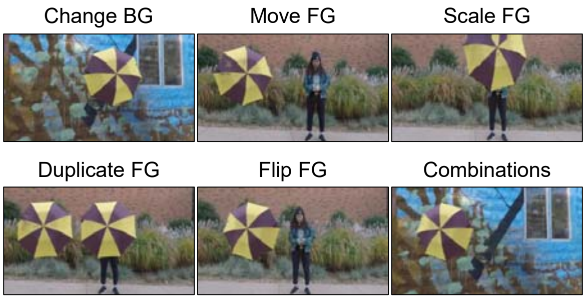

Figure 6: Our model supports many scene editing applications such as background changing, foreground moving, scaling, duplicating, flipping, and arbitrary combinations.

Table 4: Numeric comparisons on $L_{db}, L_{mf}, \boldsymbol{F}_{st}$, and random sampling strategy.

PSNR ↑ / LPIPS ↓	Training frames	Unseen frames	Unseen videos
w/o. L_{db}	22.76 / 0.148	20.69 / 0.348	21.68 / 0.345
w/o. L_{mf}	22.95 / 0.135	20.95 / 0.354	21.67 / 0.342
w/o. $\boldsymbol{F}_{st}$	21.75 / 0.230	19.79 / 0.371	21.81 / 0.337
w/o. *random*	17.30 / 0.435	16.93 / 0.512	20.28 / 0.353
Ours	**23.02 / 0.130**	**21.30 / 0.304**	**22.63 / 0.277**

Table 5: Comparisons of solving the continuous trajectory by using ODE solver [5] and our discretization method.

Method	$N = 1$	$N = 2$	ODE solver [5]
PSNR ↑ / LPIPS ↓	22.90 / 0.136	22.97 / 0.134	**23.75 / 0.129**

refer to the *supplementary material* for more details.

4.2. Novel View Synthesis from Single Video

In this section, we trained the models from single monocular videos, where existing methods are applicable to this setting. Specifically, we first tested the performance on training frames, which is the widely-used setting to evaluate video-based NeRFs. Then, we tested the generalization ability on unseen frames in the video, where other existing methods are not able to transfer well to the unseen motions.

Novel view synthesis on training frames. To evaluate the synthesized novel views, we followed DynNeRF [13] and tested the performance by fixing the view to the first camera and changing time. We reported the PSNR and LPIPS [55] in Table 1. We evaluated the performance of Li *et al.* [21], Tretschk *et al.* [40], NeuPhysics [33], and Chen *et al.* [13] from the official implementations. Even without the need of generalization ability, our method achieves better results.

Novel view synthesis on unseen frames. We split the video frames into two groups: four front frames were used for training and the rest of eight unseen frames were utilized to render novel views. Figure 4 shows that our model successfully renders new motions in the unseen frames, while DynNeRF [13] only interpolates novel views in the training frames. The reported PSNR, SSIM [47], and LPIPS scores in Table 2 quantitatively verify the superiority of our model.

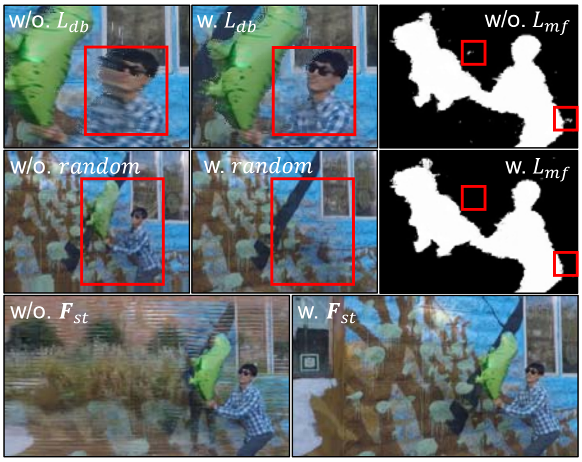

Figure 7: Ablation studies on L_{db}, L_{mf}, random sampling, and $\boldsymbol{F}_{st}$.

4.3. Novel View Synthesis from Multiple Videos

In the section, we tested the novel view synthesis performance on multiple dynamic scenes. It is worth noting that as existing methods can only learn from single monocular videos, they are not applicable to the settings that need to train on multiple videos. Then, we evaluated the novel scene adaption ability on several novel monocular videos. Lastly, we conducted a series of scene editing experiments.

Novel view synthesis on training videos. We selected Balloon2 and Umbrella scenes to train our model. As shown in Figure 3 and Table 3, our model could distinguish foregrounds from two scenes and perform well with $\boldsymbol{F}_{dy}$. Specifically, it predicts generalizable scene flows with $\boldsymbol{F}_{temp}$ and renders more accurate details with $\boldsymbol{F}_{sp}$.

Novel view synthesis on unseen videos. We explored the generalization ability of our model by pretraining on the Balloon2 scene and fine-tuning the pretrained model on other scenes. Figure 5 presents the results of four unseen videos: Playground, Skating, Truck, and Balloon1. We also pretrained DynNeRF [13] on the Balloon2 scene for a fair comparison. While DynNeRF only learns to render new scenes from scratch, our model transfers to scenes with correct dynamic motions. By further training with 500 steps, our model achieves better image rendering quality and higher LPIPS scores, which takes about 10 minutes.

Scene editing. As Figure 6 shows, our model further supports many scene editing applications by directly processing point features without extra training. Changing background was implemented by exchanging the static scene features between two scenes. Moving, scaling, duplicating and flipping foreground were directly applied by operating video features in dynamic radiance field. The above applications could be combined arbitrarily.

4.4. Ablation Study

We conducted a series of experiments to examine the proposed L_{db}, L_{mf}, $\boldsymbol{F}_{st}$, random sampling strategy, and trajectory discretization method. We show in Figure 7 that L_{db} deblurs the margin of the synthesized foreground in novel views, L_{mf} keeps the blending consistency in different views for delineating the foreground more accurately, The random sampling strategy successfully renders static scene without dynamic foreground, and backgrounds of two scenes are mixed without $\boldsymbol{F}_{st}$. We also show the numeric comparisons of $L_{db}, L_{mf}, \boldsymbol{F}_{st}$, and random sampling strategy in Table 4. In Table 5, our discretization method reaches comparable results with ODE solvers [5] and achieves higher performance with the increase of N.

5. Conclusion

In this paper, we study the generalization ability of novel view synthesis from monocular videos. The challenge is the ambiguity of 3D point features and scene flows in the viewing directions. We find that video frame features and optical flows are a pair of complementary constraints for learning 3D point features and scene flows. To achieve this, we propose a generalizable dynamic radiance field called MonoNeRF. We estimate point features and trajectories from the video features extracted by a video encoder and render dynamic scenes from points features. Experiments show that our model could learn a generalizable radiance field for dynamic scenes, and support many scene editing applications.

Acknowledgements. This work was supported by the National Key Research and Development Program of China under Grant No. 2020AAA0108100, the National Natural Science Foundation of China under Grant No. 62206147 and 61971343. The authors would like to thank the Shanghai Artificial Intelligence Laboratory for its generous support of computational resources.

References

[1] Shengqu Cai, Anton Obukhov, Dengxin Dai, and Luc Van Gool. Pix2NeRF: Unsupervised conditional π-GAN for single image to neural radiance fields translation. In *CVPR*, pages 3971–3980, 2022. 3

[2] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*, pages 6299–6308, 2017. 3, 7

[3] Eric Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. pi-GAN: Periodic implicit generative adversarial networks for 3D-aware image synthesis. In *CVPR*, 2021. 3

[4] Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. MVSNeRF: Fast generalizable radiance field reconstruction from multi-view stereo. In *ICCV*, pages 14124–14133, 2021. 2

[5] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *NeurIPS*, volume 31, 2018. 2, 4, 5, 8, 9

[6] Shenchang Eric Chen and Lance Williams. View interpolation for image synthesis. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pages 279–288, 1993. 2

[7] Xingyu Chen, Qi Zhang, Xiaoyu Li, Yue Chen, Ying Feng, Xuan Wang, and Jue Wang. Hallucinated neural radiance fields in the wild. In *CVPR*, pages 12943–12952, 2022. 2

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *CVPR*, pages 248–255, 2009. 7

[9] Kangle Deng, Andrew Liu, Jun-Yan Zhu, and Deva Ramanan. Depth-supervised NeRF: Fewer views and faster training for free. In *CVPR*, June 2022. 2

[10] Yilun Du, Yinan Zhang, Hong-Xing Yu, Joshua B. Tenenbaum, and Jiajun Wu. Neural radiance flow for 4d view synthesis and video processing. In *ICCV*, 2021. 2, 3, 7

[11] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. SlowFast networks for video recognition. In *ICCV*, pages 6201–6210, 2019. 7

[12] Guy Gafni, Justus Thies, Michael Zollhöfer, and Matthias Nießner. Dynamic neural radiance fields for monocular 4d facial avatar reconstruction. In *CVPR*, pages 8649–8658, June 2021. 2, 3

[13] Chen Gao, Ayush Saraf, Johannes Kopf, and Jia-Bin Huang. Dynamic view synthesis from dynamic monocular video. In *ICCV*, 2021. 1, 2, 3, 4, 5, 6, 7, 8

[14] Steven J Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F Cohen. The lumigraph. In *ACM SIGGRAPH*, pages 43–54, 1996. 2

[15] Jiatao Gu, Lingjie Liu, Peng Wang, and Christian Theobalt. StyleNeRF: A style-based 3D aware generator for high-resolution image synthesis. In *ICLR*, 2022. 3

[16] Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. Mask r-cnn. In *ICCV*, Oct 2017. 7

[17] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CVPR*, pages 770–778, 2016. 7

[18] Wonbong Jang and Lourdes Agapito. CodeNeRF: Disentangled neural radiance fields for object categories. In *ICCV*, pages 12949–12958, October 2021. 2, 3

[19] Marc Levoy and Pat Hanrahan. Light field rendering. In *ACM SIGGRAPH*, 1996. 2

[20] Tianye Li, Mira Slavcheva, Michael Zollhöfer, 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 *CVPR*, pages 5521–5531, June 2022. 3

[21] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. In *CVPR*, 2021. 2, 3, 5, 6, 7, 8

[22] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. In *NeurIPS*, volume 33, 2020. 2

[23] Lingjie Liu, Marc Habermann, Viktor Rudnev, Kripasindhu Sarkar, Jiatao Gu, and Christian Theobalt. Neural actor: Neural free-view synthesis of human actors with pose control. *ACM Trans. Graph.(ACM SIGGRAPH Asia)*, 2021. 3

[24] Xingyu Liu, Charles R Qi, and Leonidas J Guibas. Flownet3D: Learning scene flow in 3D point clouds. In *CVPR*, pages 529–537, 2019. 3

[25] Yuan Liu, Sida Peng, Lingjie Liu, Qianqian Wang, Peng Wang, Theobalt Christian, Xiaowei Zhou, and Wenping Wang. Neural rays for occlusion-aware image-based rendering. In *CVPR*, 2022. 2

[26] 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 *CVPR*, 2021. 2

[27] Ben Mildenhall, Peter Hedman, Ricardo Martin-Brualla, Pratul P. Srinivasan, and Jonathan T. Barron. NeRF in the dark: High dynamic range view synthesis from noisy raw images. In *CVPR*, pages 16169–16178, 2022. 2

[28] 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 *ECCV*, 2020. 2, 4, 6

[29] Michael Niemeyer and Andreas Geiger. GIRAFFE: Representing scenes as compositional generative neural feature fields. In *CVPR*, pages 11453–11464, June 2021. 3

[30] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Occupancy Flow: 4D reconstruction by learning particle dynamics. In *ICCV*, 2019. 3

[31] Julian Ost, Fahim Mannan, Nils Thuerey, Julian Knodt, and Felix Heide. Neural scene graphs for dynamic scenes. In *CVPR*, pages 2856–2865, June 2021. 3

[32] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-NeRF: Neural Radiance Fields for Dynamic Scenes. In *CVPR*, 2020. 3

[33] Yi-Ling Qiao et al. Neuphysics: Editable neural geometry and physics from monocular videos. In *NeurIPS*, 2022. 6, 8

[34] 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 TPAMI*, 44(3):1623–1637, 2022. 7

[35] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-Motion Revisited. In *CVPR*, 2016. 7

[36] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise View Selection for Unstructured Multi-View Stereo. In *ECCV*, 2016. 7

[37] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. GRAF: Generative radiance fields for 3D-aware image synthesis. In *NeurIPS*, 2020. 3

[38] Pratul P. Srinivasan, Boyang Deng, Xiuming Zhang, Matthew Tancik, Ben Mildenhall, and Jonathan T. Barron. NeRV: Neural reflectance and visibility fields for relighting and view synthesis. In *CVPR*, 2021. 2

[39] Zachary Teed and Jia Deng. RAFT: Recurrent All-Pairs field transforms for optical flow. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *ECCV*, pages 402–419, 2020. 4, 7

[40] 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 *ICCV*, pages 12939–12950, 2021. 2, 3, 6, 8

[41] Alex Trevithick and Bo Yang. GRF: Learning a general radiance field for 3D scene representation and rendering. In *ICCV*, 2021. 3

[42] Emanuele Trucco and Konstantinos Plakas. Video tracking: a concise survey. *IEEE Journal of oceanic engineering*, 31(2):520–529, 2006. 3

[43] Yi-Hsuan Tsai, Ming-Hsuan Yang, and Michael J Black. Video segmentation via object flow. In *CVPR*, pages 3899–3908, 2016. 3

[44] S. Vedula, S. Baker, P. Rander, R. Collins, and T. Kanade. Three-dimensional scene flow. In *ICCV*, volume 2, pages 722–729 vol.2, 1999. 2, 3

[45] Longguang Wang, Yulan Guo, Li Liu, Zaiping Lin, Xinpu Deng, and Wei An. Deep video super-resolution using HR optical flow estimation. *IEEE TIP*, 29:4323–4336, 2020. 3

[46] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. NeuS: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. In *NeurIPS*, 2021. 2

[47] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE TIP*, 13(4):600–612, 2004. 8

[48] 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 *CVPR*, pages 16210–16220, June 2022. 2, 3

[49] Tianhao Wu, Fangcheng Zhong, Andrea Tagliasacchi, Forrester Cole, and Cengiz Öztireli. D2NeRF: Self-supervised decoupling of dynamic and static objects from a monocular video. In *NeurIPS*, 2022. 2, 3

[50] Wenqi Xian, Jia-Bin Huang, Johannes Kopf, and Changil Kim. Space-time neural irradiance fields for free-viewpoint video. In *CVPR*, pages 9421–9431, 2021. 2, 3

[51] Yuanbo Xiangli, Linning 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 *ECCV*, 2022. 2

[52] Qiangeng Xu, Zexiang Xu, Julien Philip, Sai Bi, Zhixin Shu, Kalyan Sunkavalli, and Ulrich Neumann. Point-NeRF: Point-based neural radiance fields. *CVPR*, 2022. 2

[53] 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 *CVPR*, June 2020. 6, 7

[54] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelNeRF: Neural radiance fields from one or few images. In *CVPR*, 2021. 2, 7

[55] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, June 2018. 7, 8