Neural Scene Flow Fields for Space-Time View Synthesis of Dynamic Scenes

Zhengqi Li[1] Simon Niklaus[2] Noah Snavely[1] Oliver Wang[2]

[1] Cornell Tech [2] Adobe Research

Figure 1: Our method can synthesize novel views in both space and time from a single monocular video of a dynamic scene. Here we show **video** results with various configurations of fixing and interpolating view and time (left), as well as a visualization of the recovered scene geometry (right). Please view with Adobe Acrobat or KDE Okular to see animations.

Abstract

We present a method to perform novel view and time synthesis of dynamic scenes, requiring only a monocular video with known camera poses as input. To do this, we introduce Neural Scene Flow Fields, a new representation that models the dynamic scene as a time-variant continuous function of appearance, geometry, and 3D scene motion. Our representation is optimized through a neural network to fit the observed input views. We show that our representation can be used for varieties of in-the-wild scenes, including thin structures, view-dependent effects, and complex degrees of motion. We conduct a number of experiments that demonstrate our approach significantly outperforms recent monocular view synthesis methods, and show qualitative results of space-time view synthesis on a variety of real-world videos.

1. Introduction

The topic of novel view synthesis has recently seen impressive progress due to the use of neural networks to *learn* representations that are well suited for view synthesis tasks. Most prior approaches in this domain make the assumption that the scene is *static*, or that it is observed from multiple synchronized input views. However, these restrictions are violated by most videos shared on the Internet today, which frequently feature scenes with diverse dynamic content (e.g., humans, animals, vehicles), recorded by a single camera.

We present a new approach for novel view and time synthesis of dynamic scenes from monocular video input with known (or derivable) camera poses. This problem is highly ill-posed since there can be multiple scene configurations that lead to the same observed image sequences. In addition, using multi-view constraints for moving objects is challenging, as doing so requires knowing the dense 3D motion of all scene points (i.e., the "scene flow").

In this work, we propose to represent a dynamic scene as a continuous function of both space and time, where its output consists of not only reflectance and density, but also *3D scene motion*. Similar to prior work, we parameterize this function with a deep neural network (a multi-layer perceptron, MLP), and perform rendering using volume tracing [46]. We optimize the weights of this MLP using a scene flow fields warping loss that enforces that our scene representation is temporally consistent with the input views. Crucially, as we model dense scene flow fields in 3D, our function can represent the sharp motion discontinuities that arise when projecting the scene into image space, even with simple low-level 3D smoothness priors. Further, dense scene flow fields also enable us to interpolate along changes in both space *and* time. To the best our knowledge, our approach is the first to achieve novel view and time synthesis of dynamic scenes captured from a monocular camera.

As the problem is very challenging, we introduce different components that improve rendering quality over a baseline solution. Specifically, we analyze scene flow ambiguity at motion disocclusions and propose a solution to it. We also show how to use data-driven priors to avoid local minima

during optimization, and describe how to effectively combine a static scene representation with a dynamic one which lets us render views with higher quality by leveraging multi-view constraints in static regions.

In summary, our key contributions include: (1) a neural representation for space-time view synthesis of dynamic scenes that we call *Neural Scene Flow Fields*, that has the capacity to model 3D scene dynamics, and (2) a method for optimizing Neural Scene Flow Fields on monocular video by leveraging multiview constraints in both rigid and non-rigid regions, allowing us to synthesize and interpolate both view and time simultaneously.

2. Related Work

Our approach is motivated by a large body of work in the areas of novel view synthesis, dynamic scene reconstruction, and video understanding.

Novel view synthesis. Many methods propose first building an explicit 3D scene geometry such as point clouds or meshes, and rendering this geometry from novel views [11, 13, 17, 24, 33, 68]. Light field rendering methods on the other hand, synthesize novel views by using implicit soft geometry estimates derived from densely sampled images [12, 22, 35]. Numerous other works improve the rendering quality of light fields by exploiting their special structure [16, 57, 63, 75]. Yet another promising 3D representation is multiplane images (MPIs), that have been shown to model complex scene appearance [9, 10, 15, 20, 45, 69].

Recently, deep learning methods have shown promising results by *learning* a representation that is suited for novel view synthesis. Such methods have learned additional deep features that exist on top of reconstructed meshes [25, 60, 73] or dense depth maps [21, 79]. Alternately, pure voxel-based implicit scene representations have become popular due to their simplicity and CNN-friendly structure [14, 19, 38, 66, 67, 73]. Our method is based on a recent variation of these approaches to represent a scene as neural radiance field (NeRF) [46], which model the appearance and geometry of a scene implicitly by a continuous function, represented with an MLP. While the above methods have shown impressive view synthesis results, they all assume a static scene with fixed appearance over time, and hence cannot model temporal changes or dynamic scenes.

Another class of methods synthesize novel views from a *single* RGB image. These methods typically work by predicting depth maps [36, 54], sometimes with additional learned features [78], or a layered scene representation [64, 74] to fill in the content in disocclusions. While such methods, if trained on appropriate data, can be used on dynamic scenes, this is only possible on a per-frame (instantaneous) basis, and they cannot leverage repeated observations across multiple views, or be used to synthesize novel times.

Novel time synthesis. Most approaches for interpolating between video frames work in 2D image space, by directly predicting kernels that blend two images [51, 52, 53], or by modeling optical flow and warping frames/features [2, 31, 49, 50]. More recently, Lu *et al*. [39] show re-timing effect of people by using a layered representation. These approaches generate high-quality frame interpolation results, but operate in 2D and cannot be used to synthesize novel views in space.

Space-time view synthesis. There are two main reasons that scenes change appearance across time. The first is due to illumination changes; prior approaches have proposed to render novel views of single object with plausible relighting [4, 5, 6], or model time-varying appearance from internet photo collections [37, 42, 44]. However, these methods operate on static scenes and treat moving objects as outliers.

Second, appearance change can happen due to 3D scene motion. Most prior work in this domain [1, 3, 70, 85] require multi-view, time synchronized videos as input, and has limited ability to model complicated scene geometry. Most closely related ours, Yoon *et al*. [81] propose to combine single-view depth and depth from multi-view stereo to render novel views by performing explicit depth based 3D warping. However, this method has several drawbacks: it relies on human annotated foreground masks, requires cumbersome preprocessing and pretraining and tends to produce artifacts in disocclusions. Instead, we show that our model can be trained end-to-end and produces much more realistic results, and is able to represent complicated scene structure and view dependent effects along with natural degrees of motion.

Dynamic scene reconstruction. Most successful non-rigid reconstruction systems either require RGBD data as input [7, 18, 29, 48, 80, 86], or can only reconstruct sparse geometry [56, 65, 77, 83]. A few prior monocular methods proposed using strong hand-crafted priors to decompose dynamic scenes into piece-wise rigid parts [34, 59, 61]. Recent work of Luo *et al*. [40] estimates temporally consistent depth maps of scenes with small object motion by optimizing the weights of a single image depth prediction network, but we show that this approach fails to model large and complex 3D motions. Additional work has aimed to predict per-pixel scene flows of dynamic scenes from either monocular or RGBD sequences [8, 26, 30, 41, 47, 71].

3. Approach

We build upon prior work for static scenes [46], to which we add the notion of time, and estimate 3D motion by explicitly modeling forward and backward scene flow as dense 3D vector fields. In this section, we first describe this time-variant (dynamic) scene representation (Sec. 3.1) and the method for effectively optimizing this representation (Sec. 3.2) on the input views. We then discuss how to improve the rendering quality by adding an additional explicit

time-invariant (static) scene representation, optimized jointly with the dynamic one by combining both during rendering (Sec. 3.3). Finally, we describe how to achieve space-time interpolation of dynamic scenes through our trained representation (Sec. 3.4).

Background: static scene rendering. Neural Radiance Fields (NeRFs) [46] represent a static scene as a radiance field defined over a bounded 3D volume. This radiance field, denoted F_Θ, is defined by a set of parameters Θ that are optimized to reconstruct the input views. In NeRF, F_Θ is a multi-layer perceptron (MLP) that takes as input a position ($\mathbf{x}$) and viewing direction ($\mathbf{d}$), and produces as output a volumetric density (σ) and RGB color ($\mathbf{c}$):

$$(\mathbf{c}, \sigma) = F_\Theta(\mathbf{x}, \mathbf{d}) \tag{1}$$

To render the color of an image pixel, NeRF approximates a volume rendering integral. Let $\mathbf{r}$ be the camera ray emitted from the center of projection through a pixel on the image plane. The expected color $\hat{\mathbf{C}}$ of that pixel is then given by:

$$\hat{\mathbf{C}}(\mathbf{r}) = \int_{t_n}^{t_f} T(t)\, \sigma(\mathbf{r}(t))\, \mathbf{c}(\mathbf{r}(t), \mathbf{d})\, dt$$
$$\text{where } T(t) = \exp\left(- \int_{t_n}^{t} \sigma(\mathbf{r}(s))\, ds \right). \tag{2}$$

Intuitively, $T(t)$ corresponds to the accumulated transparency along that ray. The loss is then the difference between the reconstructed color $\hat{\mathbf{C}}$, and the ground truth color $\mathbf{C}$ corresponding to the pixel that ray originated from $\mathbf{r}$:

$$\mathcal{L}_{\text{static}} = \sum_{\mathbf{r}} ||\hat{\mathbf{C}}(\mathbf{r}) - \mathbf{C}(\mathbf{r})||_2^2. \tag{3}$$

3.1. Neural scene flow fields for dynamic scenes

To capture scene dynamics, we extend the static scenario described in Eq. 1 by including time in the domain and explicitly modeling 3D motion as dense scene flow fields. For a given 3D point $\mathbf{x}$ and time i, the model predicts not just reflectance and opacity, but also forward and backward 3D scene flow $\mathcal{F}_i = (\mathbf{f}_{i\to i+1}, \mathbf{f}_{i\to i-1})$, which denote 3D offset vectors that point to the position of $\mathbf{x}$ at times $i+1$ and $i-1$ respectively. Note that we make the simplifying assumption that movement that occurs between observed time instances is linear. To handle motion disocclusions in 3D space, we also predict disocclusion weights $\mathcal{W}_i = (w_{i\to i+1}, w_{i\to i-1})$ (described in Sec. 3.2). Our dynamic model is thus defined as:

$$(\mathbf{c}_i, \sigma_i, \mathcal{F}_i, \mathcal{W}_i) = F^{\text{dy}}_\Theta(\mathbf{x}, \mathbf{d}, i). \tag{4}$$

Note that for convenience, we use the subscript i to indicate a value at a specific time i.

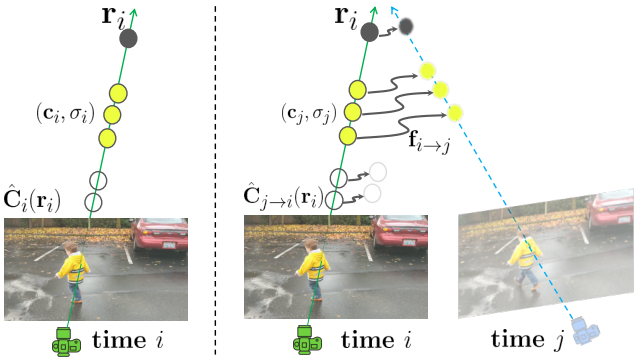


Figure 2: **Scene flow fields warping.** To render a frame at time i, we perform volume rendering along ray $\mathbf{r}_i$ with RGBσ at time i, giving us the pixel color $\hat{\mathbf{C}}_i(\mathbf{r}_i)$ (left). To warp the scene from time j to i, we offset each step along $\mathbf{r}_i$ using scene flow $\mathbf{f}_{i\to j}$ and volume render with the associated color and opacity $(\mathbf{c}_j, \sigma_j)$ (right).

3.2. Optimization

Temporal photometric consistency. The key new loss we introduce enforces that the scene at time i should be consistent with the scene at neighboring times $j \in \mathcal{N}(i)$, *when accounting for motion that occurs due to 3D scene flow*. To do this, we volume render the scene at time i from 1) the perspective of the camera at time i and 2) with the scene warped from j to i, so as to undo any motion that occurred from i to j. As shown in Fig. 2 (right), during volume rendering, we achieve this by warping each 3D sampled point location $\mathbf{x}_i$ along a ray $\mathbf{r}_i$ using the predicted scene flows fields $\mathcal{F}_i$ to look up the RGB color $\mathbf{c}_j$ and opacity σ_j at neighboring time j. This yields a rendered image, denoted $\hat{\mathbf{C}}_{j\to i}$, of the scene at time j with both camera and scene motion warped to time i:

$$\hat{\mathbf{C}}_{j\to i}(\mathbf{r}_i) = \int_{t_n}^{t_f} T_j(t)\, \sigma_j(\mathbf{r}_{i\to j}(t))\, \mathbf{c}_j(\mathbf{r}_{i\to j}(t), \mathbf{d}_i) dt$$
$$\text{where } \mathbf{r}_{i\to j}(t) = \mathbf{r}_i(t) + \mathbf{f}_{i\to j}(\mathbf{r}_i(t)). \tag{5}$$

We minimize the mean squared error (MSE) between each warped rendered view and the ground truth view:

$$\mathcal{L}_{\text{pho}} = \sum_{\mathbf{r}_i} \sum_{j\in\mathcal{N}(i)} ||\hat{\mathbf{C}}_{j\to i}(\mathbf{r}_i) - \mathbf{C}_i(\mathbf{r}_i)||_2^2 \tag{6}$$

An important caveat is that this loss is ambiguous at 3D disocculusion regions caused by motion. Analogous to 2D dense optical flow [43], scene flow is ambiguous when a 3D location becomes occluded or disoccluded between frames. These regions are especially important as they occur at the boundaries of moving objects (see Fig. 3 for an illustration). To mitigate errors due to this ambiguity, we predict extra continuous disocclusion weight fields $w_{i\to i+1}$ and $w_{i\to i-1} \in [0, 1]$, corresponding to $\mathbf{f}_{i\to i+1}$ and $\mathbf{f}_{i\to i-1}$

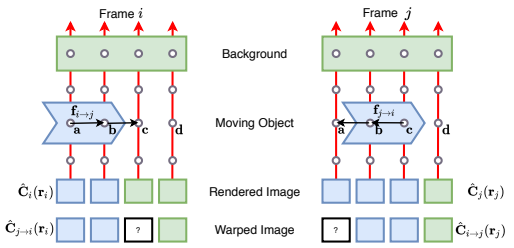


Figure 3: **Scene flow disocclusion ambiguity.** In this 2D orthographic example, a single blue object translates to the right by one unit from frame i to frame j. Here, the correct scene flow at the point labeled $\mathbf{a}$, e.g., $\mathbf{f}_{i\to j}(\mathbf{a})$, points one unit to the right, however, for the scene flow $\mathbf{f}_{i\to j}(\mathbf{c})$ (and similarly $\mathbf{f}_{j\to i}(\mathbf{a})$), there can be multiple answers. If $\mathbf{f}_{i\to j}(\mathbf{c}) = 0$, then the scene flow would incorrectly point to the foreground in the next frame, and if $\mathbf{f}_{i\to j}(\mathbf{c}) = 1$, the scene flow would point to the free-space location d at j.

respectively. These weights serve as an unsupervised confidence of where and how much strength the temporal photo-consistency loss should be applied. We apply these weights by volume rendering the weight along the ray $\mathbf{r}_i$ with opacity from time j, and multiplying the accumulated weight at each *2D pixel*:

$$\hat{W}_{j\to i}(\mathbf{r}_i) = \int_{t_n}^{t_f} T_j(t)\,\sigma_j(\mathbf{r}_{i\to j}(t))\,w_{i\to j}(\mathbf{r}_i(t))\;dt \quad (7)$$

We avoid the trivial solution where all predicted weights are zero by adding ℓ_1 regularization to encourage predicted weights to be close to one, giving us a new weighted loss:

$$\mathcal{L}_{\text{pho}} = \sum_{\mathbf{r}_i}\sum_{j\in\mathcal{N}(i)} \hat{W}_{j\to i}(\mathbf{r}_i)||\hat{\mathbf{C}}_{j\to i}(\mathbf{r}_i) - \mathbf{C}_i(\mathbf{r}_i)||_2^2 + \beta_w \sum_{\mathbf{x}_i} ||w_{i\to j}(\mathbf{x}_i) - 1||_1, \quad (8)$$

where β_w is a regularization weight which we set to 0.1 in all our experiments. We use $\mathcal{N}(i) = \{i, i\pm1, i\pm2\}$, and chain scene flow and disocclusion weights for the $i\pm2$ cases. Note that when $j = i$, there is no scene flow warping or disocclusion weights involved ($\mathbf{f}_{i\to j} = 0, \hat{W}_{j\to i}(\mathbf{r}_i) = 1$), meaning that $\hat{\mathbf{C}}_{i\to i}(\mathbf{r}_i) = \hat{\mathbf{C}}_i(\mathbf{r}_i)$, as in Fig. 2(left). Comparing Fig. 4(e) and Fig. 4(d), we can see that adding this disocclusion weight improves rendering quality near motion boundaries.

Scene flow priors. To regularize the predicted scene flow fields, we add a 3D scene flow cycle consistency term to encourage that at all sampled 3D points $\mathbf{x}_i$, the predicted forward scene flow $\mathbf{f}_{i\to j}$ is consistent with the backward scene flow $\mathbf{f}_{j\to i}$ at the corresponding location at time j (i.e. at position $\mathbf{x}_{i\to j} = \mathbf{x}_i + \mathbf{f}_{i\to j}$). Note that this cycle consistency can also be ambiguous near motion disocclusion regions in 3D, so we use the same predicted disocclusion weights to modulate this term, giving us:

$$\mathcal{L}_{\text{cyc}} = \sum_{\mathbf{x}_i}\sum_{j\in i\pm1} w_{i\to j}||\mathbf{f}_{i\to j}(\mathbf{x}_i) + \mathbf{f}_{j\to i}(\mathbf{x}_{i\to j})||_1 \quad (9)$$

(a) Our rendered views (b) w/o $\mathcal{L}_{\text{data}}$ (c) w/o $\mathcal{L}_{\text{cyc}}$ (d) w/o $\mathcal{W}_i$ (e) Full

Figure 4: **Qualitative ablations.** Results of our full method with different loss components removed. The odd rows show zoom-in rendered color and the even rows show corresponding pseudo depth. Each component reduces the overall quality in different ways.

We additionally add low-level regularizations $\mathcal{L}_{\text{reg}}$ on the predicted scene flow fields. First, following prior work [48, 77], we enforce scene flow spatial-temporal smoothness by applying ℓ_1 regularization to nearby sampled 3D points along the ray and encouraging 3D point trajectories to be piecewise linear. Second, we encourage scene flow to be small in most places [76] by applying an ℓ_1 regularization term. Please see the supplementary material for complete descriptions.

Data-driven priors. Since monocular reconstruction of complex dynamic scenes is highly ill-posed, the above losses can on occasion converge to sub-optimal local minima when randomly initialized. Therefore, we introduce two data-driven losses, a geometric consistency term [29, 28] and a single-view depth term: $\mathcal{L}_{\text{data}} = \mathcal{L}_{\text{geo}} + \beta_z\mathcal{L}_z$. We set $\beta_z = 2$ in all our experiments.

The geometric consistency helps model build accurate correspondence association between adjacent frames. In particular, it minimizes the reprojection error of scene flow displaced 3D points w.r.t. the derived 2D optical flow which we compute using pretrained networks [27, 72].

Suppose $\mathbf{p}_i$ is a 2D pixel position at time i. The corresponding 2D pixel location in the neighboring frame at time j displaced through 2D optical flow $\mathbf{u}_{i\to j}$ can be computed as $\mathbf{p}_{i\to j} = \mathbf{p}_i + \mathbf{u}_{i\to j}$. To estimate the expected 2D point location $\hat{\mathbf{p}}_{i\to j}$ at time j displaced by predicted scene flow fields, we first compute the expected scene flow $\hat{\mathbf{F}}_{i\to j}(\mathbf{r}_i)$ and the expected 3D point location $\hat{\mathbf{X}}_i(\mathbf{r}_i)$ of the ray $\mathbf{r}_i$ through volume rendering. $\hat{\mathbf{p}}_{i\to j}$ is then computed by performing perspective projection of the expected 3D point location displaced by the scene flow (i.e. $\hat{\mathbf{X}}_i(\mathbf{r}_i) + \hat{\mathbf{F}}_{i\to j}(\mathbf{r}_i)$) into the

Combined render | Static only | Dynamic only

Figure 5: **Dynamic and static components.** Our method learns static and dynamic components in the combined representation. Note person is almost still in the bottom example.

viewpoint corresponding to the frame at time j. The geometric consistency is computed as the ℓ_1 difference between $\hat{\mathbf{p}}_{i\to j}$ and $\mathbf{p}_{i\to j}$,

$$\mathcal{L}_{\text{geo}} = \sum_{\mathbf{r}_i} \sum_{j\in\{i\pm1\}} ||\hat{\mathbf{p}}_{i\to j}(\mathbf{r}_i) - \mathbf{p}_{i\to j}(\mathbf{r}_i))||_1. \quad (10)$$

We also add a single view depth prior that encourages the expected termination depth $\hat{Z}_i$ computed along each ray to be close to the depth Z_i predicted from a pre-trained single-view network [58]. As single-view depth predictions are defined up to an unknown scale and shift, we utilize a robust scale-shift invariant loss [58]:

$$\mathcal{L}_z = \sum_{\mathbf{r}_i} ||\hat{Z}_i^*(\mathbf{r}_i) - Z_i^*(\mathbf{r}_i)||_1 \quad (11)$$

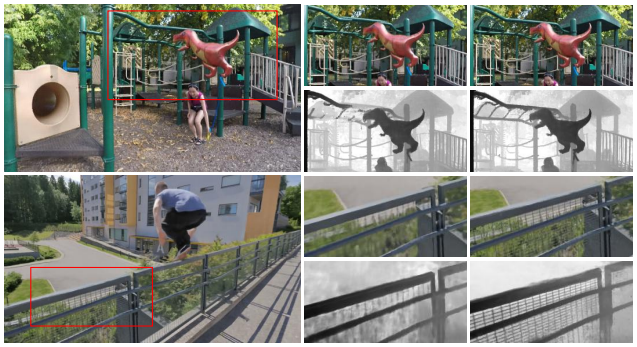
where * is a whitening operation that normalizes the depth to have zero mean and unit scale.

From Fig 4(b), we see that adding data-driven priors help the model learn correct scene geometry especially for dynamic regions. However, as both of these data-driven priors are noisy (rely on inaccurate or incorrect predictions), we use these for *initialization only*, and linearly decay the weight of $\mathcal{L}_{\text{data}}$ to zero during training.

3.3. Integrating a static scene representation

The method described so far already outperforms the state of the art, as shown in Tab. 1. However, unlike NeRF, our warping-based temporal loss can only be used in a *local* temporal neighborhood $\mathcal{N}(i)$, as dynamic components typically undergo too much deformation to reliably infer correspondence over larger temporal gaps. Static regions, however, should be consistent and should leverage observations from *all* frames. Therefore, we propose to combine our dynamic (time-dependent) scene representation with a static (time-independent) one, and require that when combined, the resulting volume-rendered images match the input. We model each representation with its own MLP, where the dynamic scene component is represented with Eq. 4, and the static one is represented as a variant of Eq. 1:

$$(\mathbf{c}, \sigma, v) = F_\Theta^{\text{st}}(\mathbf{x}, \mathbf{d}) \quad (12)$$



(a) Our rendered views w/ static | (b) w/o static | (c) w/ static

Figure 6: **Static scene representation ablation.** Adding a static scene representation yields higher fidelity renderings, especially in static regions (a,c) when compared to the pure dynamic model (b).

where v is an unsupervised 3D blending weight field, that linearly blends the RGBσ from static and dynamic scene representations along the ray. Intuitively, v should assign a low weight to the dynamic representation at static regions with sufficient observations, as these can be rendered in higher fidelity by the static representation, while assigning a lower weight to the static representation in regions that are moving, as these can be better modeled by the dynamic representation. We found adding the extra v leads to less artifacts and more stable training than the configuration without v. The combined rendering equation is then written as:

$$\hat{\mathbf{C}}_i^{\text{cb}}(\mathbf{r}_i) = \int_{t_n}^{t_f} T_i^{\text{cb}}(t)\, \sigma_i^{\text{cb}}(t)\, \mathbf{c}_i^{\text{cb}}(t) dt, \quad (13)$$

where $\sigma_i^{\text{cb}}(t)\, \mathbf{c}_i^{\text{cb}}(t)$ is a linear combination of static and dynamic scene components, weighted by $v(t)$:

$$\sigma_i^{\text{cb}}(t)\, \mathbf{c}_i^{\text{cb}}(t) = v(t)\, \mathbf{c}(t)\, \sigma(t) + (1\text{-}v(t))\, \mathbf{c}_i(t)\, \sigma_i(t) \quad (14)$$

For clarity, we omit $\mathbf{r}_i$ in each prediction. We then train the combined scene representation by minimizing MSE between $\hat{\mathbf{C}}_i^{\text{cb}}$ with the corresponding input view:

$$\mathcal{L}_{\text{cb}} = \sum_{\mathbf{r}_i} ||\hat{\mathbf{C}}_i^{\text{cb}}(\mathbf{r}_i) - \mathbf{C}_i(\mathbf{r}_i)||_2^2. \quad (15)$$

This loss is added to the previously defined losses on the dynamic representation, giving us the final combined loss:

$$\mathcal{L} = \mathcal{L}_{\text{cb}} + \mathcal{L}_{\text{pho}} + \beta_{\text{cyc}}\mathcal{L}_{\text{cyc}} + \beta_{\text{data}}\mathcal{L}_{\text{data}} + \beta_{\text{reg}}\mathcal{L}_{\text{reg}} \quad (16)$$

where the β coefficients weight each term. Fig. 5 shows separately rendered static and dynamic scene components, and Fig. 6 visually compares renderings with and without integrating a static scene representation.

3.4. Space-time view synthesis

To render novel views at a given time, we simply volume render each pixel using Eq. 5 (dynamic) or Eq. 13

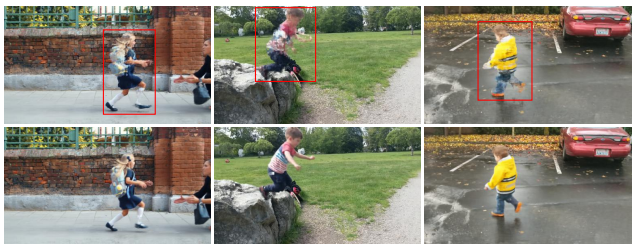


Figure 7: **Novel time synthesis.** Rendering images by interpolating the time index (top) yields blending artifacts compared to our scene flow based rendering (bottom).

(static+dynamic). However, we observe that while this approach produces good results *at times corresponding to input views*, the representation does not allow us to interpolate time-variant geometry at in-between times, leading instead to rendered results that look like linearly blended combinations of existing frames (Fig. 7).

Instead, we render intermediate times by warping the scene based on the predicted scene flow. For efficient rendering, we propose a splatting-based plane-sweep volume rendering approach. To render an image at intermediate time $i + \delta_i, \delta_i \in (0, 1)$ at a specified target viewpoint, we sweep over every step emitted from the novel viewpoint from front to back. At each sampled step t along the ray, we query point information through our models at both times i and $i + 1$, and displace the 3D points at time i by the scaled scene flow $\mathbf{x}_i + \delta_i \mathbf{f}_{i \to i+1}(\mathbf{x}_i)$, and similarity for time $i+1$. We then splat the 3D displaced points onto a $(\mathbf{c}, \alpha)$ accumulation buffer at the novel viewpoint, and blend splats from time i and $i + 1$ with linear weights $1 - \delta_i, \delta_i$. The final rendered view is obtained by volume rendering the accumulation buffer (see supplementary material for a diagram).

4. Experiments

Implementation details. We use COLMAP [62] to estimate camera intrinsics and extrinsics, and consider these fixed during optimization. As COLMAP assumes a static scene, we mask out features from regions associated with common classes of dynamic objects using off-the-shelf instance segmentation [23]. During training and testing, we sample 128 points along each ray and normalize the time indices $i \in [0, 1]$. As with NeRF [46], we use positional encoding to transform the inputs, and parameterize scenes using normalized device coordinates. A separate model is trained for a each scene using the Adam optimizer [32] with a learning rate of 0.0005. While integrating the static scene representation, we optimize two networks simultaneously. Training a full model takes around two days per scene using two NVIDIA V100 GPUs and rendering takes roughly 6 seconds for each 512×288 frame. We refer readers to the supplemental material for our network architectures, hyper-parameter settings, and other implementation details.

Methods	MV	Dynamic Only			Full		
		SSIM (↑)	PSNR (↑)	LPIPS (↓)	SSIM (↑)	PSNR (↑)	LPIPS (↓)
SinSyn [78]	No	0.371	14.61	0.341	0.488	16.21	0.295
MPIs [74]	No	0.494	16.44	0.383	0.629	19.46	0.367
3D Ken Burn [54]	No	0.462	16.33	0.224	0.630	19.25	0.185
3D Photo [64]	No	0.486	16.73	0.217	0.614	19.29	0.215
NeRF [46]	Yes	0.532	16.98	0.314	0.893	24.90	0.098
Luo *et al.* [40]	Yes	0.530	16.97	0.207	0.746	21.37	0.141
Yoon *et al.* [81]	Yes	0.547	17.34	0.199	0.761	21.78	0.127
Ours (wo/ static)	Yes	**0.760**	21.88	0.108	0.906	26.95	0.071
Ours (w/ static)	Yes	0.758	**21.91**	**0.097**	**0.928**	**28.19**	**0.045**

Table 1: **Quantitative evaluation of novel view synthesis on the Dynamic Scenes dataset.** MV indicates whether the approach makes use multi-view information or not.

Methods	Dynamic Only			Full		
	SSIM (↑)	PSNR (↑)	LPIPS (↓)	SSIM (↑)	PSNR (↑)	LPIPS (↓)
NeRF [46]	0.522	16.74	0.328	0.862	24.29	0.113
[64] + [52]	0.490	16.97	0.216	0.616	19.43	0.217
[81] + [52]	0.498	16.85	0.201	0.748	21.55	0.134
Ours (w/o static)	0.720	21.51	0.149	0.875	26.35	0.090
Ours (w/ static)	**0.724**	**21.58**	**0.143**	**0.892**	**27.38**	**0.066**

Table 2: **Quantitative evaluation of novel view and time synthesis.** See Sec. 4.2 for a description of the baselines.

4.1. Baselines and error metrics

We compare our approach to state-of-the-art single-view and multi-view novel view synthesis algorithms. For single-view methods, we compare to MPIs [74] and SinSyn [78], trained on indoor real estate videos [84]; 3D Photos [64] and 3D Ken Burns [54] were trained mainly on images in the wild. Since these methods can only compute depth up to an unknown scale and shift, we align the predicted depths with the SfM sparse point clouds before rendering. For multi-view, we compare to a recent dynamic view synthesis method [81]. Since the authors do not provide source code, we reimplemented their approach based on the paper description. We also compare to a video depth prediction method [40] and perform novel view synthesis by rendering the point cloud into novel views while filling in disoccluded regions. Finally, we train a standard NeRF [46], with and without the added time domain, on each dynamic scene.

We report the rendering quality of each approach with three standard error metrics: structural similarity index measure (SSIM), peak signal-to-noise ratio (PSNR), and perceptual similarity through LPIPS [82], on both the entire scene (Full) and in dynamic regions only (Dynamic Only).

4.2. Quantitative evaluation

We evaluate on the Nvidia Dynamic Scenes Dataset [81], which consists of 8 scenes with human and non-human motion recorded by 12 synchronized cameras. As in the original work [81], we simulate a moving monocular camera by extracting images sampled from each camera viewpoint at different time instances, and evaluate the result of view

3D Photo [64] NeRF [46] Yoon *et al*. [81] Ours GT

Figure 8: **Qualitative comparisons on the Dynamic Scenes dataset.** Compared with prior methods, our rendered images more closely match the ground truth, and include fewer artifacts, as shown in the highlighted regions.

synthesis with respect to known held-out viewpoints and frames. For each scene, we extract 24 frames from the original videos for training and use the remaining 11 held-out images per time instance for evaluation.

Novel view synthesis. We first evaluate our approach and other baselines on the task of novel view synthesis (at the same time instances as the training sequences). The quantitative results are shown in Table 1. Our approach without the static scene representation (Ours w/o static) already significantly outperforms other single-view and multi-view baselines in both dynamic regions and on the entire scene. NeRF has the second best performance on the entire scene, but cannot model scene dynamics. Moreover, adding the static scene representation improves overall rendering quality by more than 30%, demonstrating the benefits of leveraging global multi-view information from rigid regions where possible.

Novel view and time synthesis. We also evaluate the task of novel view and time synthesis by extracting every other frame from the original Dynamic Scenes dataset videos for training, and evaluating on the held-out intermediate time instances at held-out camera viewpoints. Since we are not aware of prior monocular space-time view interpolation methods, we use two state-of-the-art view synthesis baselines [64, 81] to synthesize images at the testing camera viewpoints followed by 2D frame interpolation [52] to render intermediate times, as well as NeRF evaluated directly at the novel space-time views. Table 2 shows that our method significantly outperforms all baselines in both dynamic regions and the entire scene.

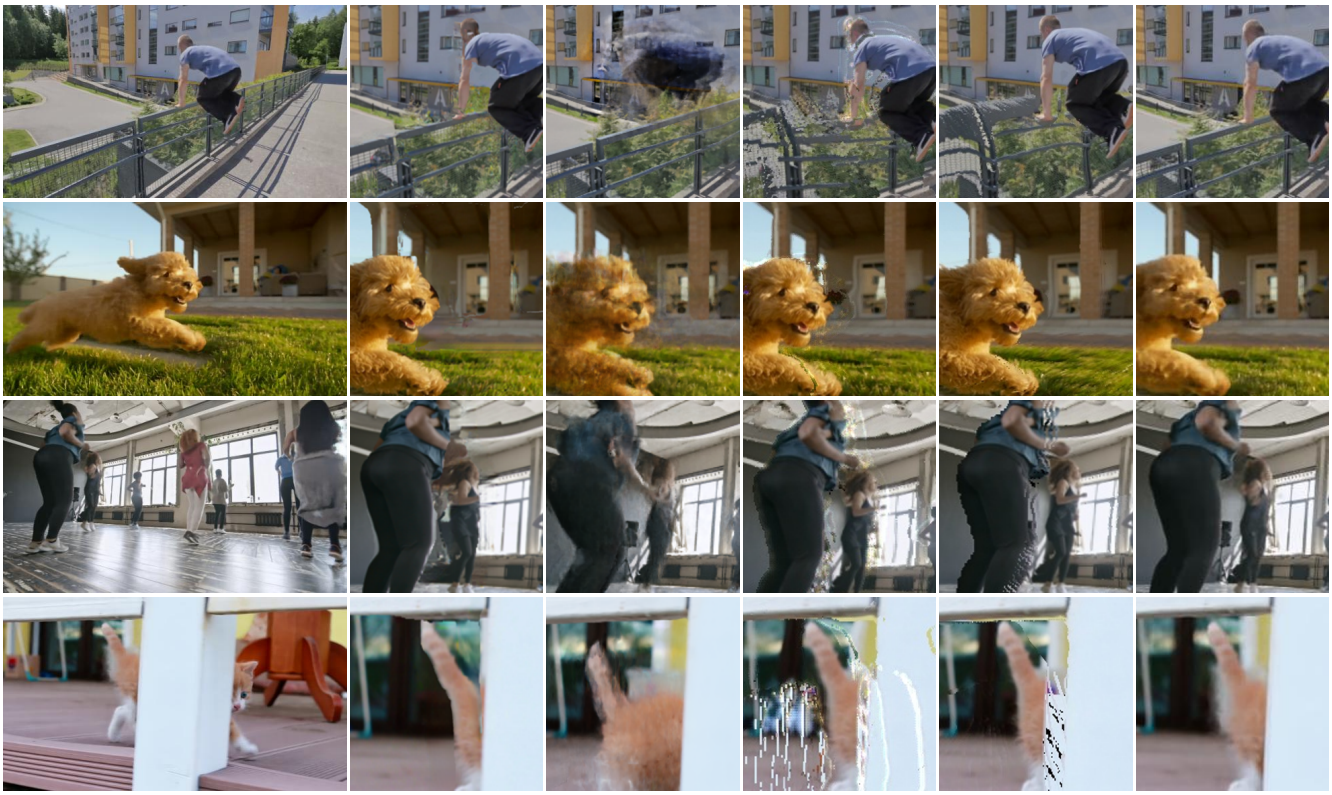
Methods	Dynamic Only			Full		
	SSIM (↑)	PSNR (↑)	LPIPS (↓)	SSIM (↑)	PSNR (↑)	LPIPS (↓)
NeRF (w/ time)	0.630	18.89	0.159	0.875	24.33	0.081
w/o $\mathcal{L}_z$	0.710	19.66	0.132	0.882	25.16	0.078
w/o $\mathcal{L}_{\text{geo}}$	0.713	19.74	0.139	0.885	25.19	0.079
w/o $\mathcal{L}_{\text{cyc}}$	0.731	20.52	0.115	0.890	26.15	0.072
w/o $\mathcal{L}_{\text{reg}}$	0.751	21.22	0.110	0.895	26.67	0.074
w/o $\mathcal{W}_i$	0.754	21.31	0.112	0.894	26.23	0.074
w/o static	**0.760**	21.88	0.108	0.906	26.95	0.071
Full (w/ static)	0.758	**21.91**	**0.097**	**0.928**	**28.19**	**0.045**

Table 3: **Ablation study on the Dynamic Scenes dataset.** See Sec. 4.2 for detailed descriptions of each of the ablations.

Ablation study. We analyze the effect of each proposed system component in the task of novel view synthesis by removing (1) all added losses, which gives us NeRF extended to the temporal domain (NeRF (w/ time)); (2) the single view depth prior (w/o $\mathcal{L}_z$); (3) the geometry consistency prior (w/o $\mathcal{L}_{\text{geo}}$); (4) the scene flow cycle consistency term (w/o $\mathcal{L}_{\text{cyc}}$); (5) the scene flow regularization term (w/o $\mathcal{L}_{\text{reg}}$); (6) the disocclusion weight fields (w/o $\mathcal{W}_i$); (7) the static representation (w/o static). The results, shown in Table 3, demonstrate the relative importance of each component, with the full system performing the best.

4.3. Qualitative evaluation

We provide qualitative comparisons on the Dynamic Scenes dataset (Fig. 8) and on monocular video clips collected in-the-wild from the internet featuring complex object motions such as jumping, running, or dancing with various occlusions (Fig. 9). NeRF [46] correctly reconstructs most static regions, but produces ghosting in dynamic regions



Our rendered views | 3D Photo [64] | NeRF [46] | Yoon *et al*. [81] | Luo *et al*. [40] | Ours

Figure 9: **Qualitative comparisons on monocular video clips.** When compared to baselines, our approach more correctly synthesizes hidden content in disocclusions (shown in the last three rows), and locations with complex scene structure such as the fence in the first row.

since it treats all the moving objects as view-dependent effects, leading to incorrect interpolation results. The state-of-the-art single-view method [64] tends to synthesize incorrect content at disocclusions, such as the bins and speaker in the last three rows of Fig. 9. In contrast, methods based on reconstructing explicit depth maps [40, 81] have difficulty modeling complex scene appearance and geometry such as the thin structures in the third row of Fig. 8 and the first row of Fig. 9.

5. Discussion

Limitations. Monocular space-time view synthesis of dynamic scenes is challenging, and we have only scratched the surface with our proposed method. In particular, there are several limitations to our approach. Similar to NeRF, training and rendering times are high, even at limited resolutions. Additionally, each scene has to be reconstructed from scratch and our representation is unable to extrapolate content unseen in the training views (See Fig. 10(a)). Furthermore, we found that rendering quality degrades when either the length of the sequence is increased given default number of model parameters (most of our sequences were trained for 1$\sim$2 seconds), or when the amount of motion is extreme (See Fig. 10(b-c), where we train a model on a low frame rate video). Our method can end up in the incorrect local minima if object and camera motions are close to a degenerate case, e.g., colinear, as described in Park *et al*. [55].

(a) Non-seen disocclusion | (b) GT for (c) | (c) Missing details

Figure 10: **Limitations.** Our method is unable to extrapolate content unseen in the training views (a), and has difficulty recovering high frequency details if a video involves extreme object motions (b,c).

Conclusion. We presented an approach for monocular novel view and time synthesis of complex dynamic scenes by Neural Scene Flow Fields, a new representation that implicitly models scene time-variant reflectance, geometry and 3D motion. We have shown that our method can generate compelling space-time view synthesis results for scenes with natural in-the-wild scene motion. In the future, we hope that such methods can enable high-resolution views of dynamic scenes with larger scale and larger viewpoint changes.

References

[1] Aayush Bansal, Minh Vo, Yaser Sheikh, Deva Ramanan, and Srinivasa Narasimhan. 4d visualization of dynamic events from unconstrained multi-view videos. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, pages 5366–5375, 2020.

[2] Wenbo Bao, Wei-Sheng Lai, Chao Ma, Xiaoyun Zhang, Zhiyong Gao, and Ming-Hsuan Yang. Depth-aware video frame interpolation. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[3] Mojtaba Bemana, Karol Myszkowski, Hans-Peter Seidel, and Tobias Ritschel. X-fields: Implicit neural view-, light-and time-image interpolation. *ACM Trans. Graphics*, 39(6), 2020.

[4] S. Bi, Zexiang Xu, Pratul P. Srinivasan, Ben Mildenhall, Kalyan Sunkavalli, Milovs Havsan, Yannick Hold-Geoffroy, D. Kriegman, and R. Ramamoorthi. Neural reflectance fields for appearance acquisition. *ArXiv*, abs/2008.03824, 2020.

[5] Sai Bi, Zexiang Xu, Kalyan Sunkavalli, Miloš Hašan, Yannick Hold-Geoffroy, David Kriegman, and Ravi Ramamoorthi. Deep reflectance volumes: Relightable reconstructions from multi-view photometric images. *Proc. European Conf. on Computer Vision (ECCV)*, 2020.

[6] Sai Bi, Zexiang Xu, Kalyan Sunkavalli, David Kriegman, and Ravi Ramamoorthi. Deep 3d capture: Geometry and reflectance from sparse multi-view images. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, pages 5960–5969, 2020.

[7] Aljaz Bozic, Michael Zollhofer, Christian Theobalt, and Matthias Niessner. Deepdeform: Learning non-rigid rgb-d reconstruction with semi-supervised data. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[8] Fabian Brickwedde, Steffen Abraham, and Rudolf Mester. Mono-sf: Multi-view geometry meets single-view depth for monocular scene flow estimation of dynamic traffic scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, pages 2780–2790, 2019.

[9] Michael Broxton, Jay Busch, Jason Dourgarian, Matthew DuVall, Daniel Erickson, Dan Evangelakos, John Flynn, Ryan Overbeck, Matt Whalen, and Paul Debevec. A low cost multi-camera array for panoramic light field video capture. In *SIGGRAPH Asia 2019 Posters*, SA '19, New York, NY, USA, 2019. Association for Computing Machinery.

[10] Michael Broxton, John Flynn, Ryan Overbeck, Daniel Erickson, Peter Hedman, Matthew Duvall, Jason Dourgarian, Jay Busch, Matt Whalen, and Paul Debevec. Immersive light field video with a layered mesh representation. *ACM Trans. Graph.*, 39(4), July 2020.

[11] Chris Buehler, Michael Bosse, Leonard McMillan, Steven Gortler, and Michael Cohen. Unstructured lumigraph rendering. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 425–432, 2001.

[12] Jin-Xiang Chai, Xin Tong, Shing-Chow Chan, and Heung-Yeung Shum. Plenoptic sampling. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, page 307–318, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[13] Gaurav Chaurasia, Sylvain Duchene, Olga Sorkine-Hornung, and George Drettakis. Depth synthesis and local warps for plausible image-based navigation. *ACM Trans. Graphics*, 32(3):1–12, 2013.

[14] Zhang Chen, Anpei Chen, Guli Zhang, Chengyuan Wang, Yu Ji, Kiriakos N Kutulakos, and Jingyi Yu. A neural rendering framework for free-viewpoint relighting. *arXiv preprint arXiv:1911.11530*, 2019.

[15] Inchang Choi, Orazio Gallo, Alejandro Troccoli, Min H Kim, and Jan Kautz. Extreme view synthesis. In *Proc. Int. Conf. on Computer Vision (ICCV)*, pages 7781–7790, 2019.

[16] Abe Davis, Marc Levoy, and Frédo Durand. Unstructured light fields. *Comput. Graph. Forum*, 31:305–314, 2012.

[17] Paul E Debevec, Camillo J Taylor, and Jitendra Malik. Modeling and rendering architecture from photographs: A hybrid geometry-and image-based approach. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 11–20, 1996.

[18] Mingsong Dou, Sameh Khamis, Yury Degtyarev, Philip L. Davidson, Sean Ryan Fanello, Adarsh Kowdle, Sergio Orts, Christoph Rhemann, David Kim, Jonathan Taylor, Pushmeet Kohli, Vladimir Tankovich, and Shahram Izadi. Fusion4D: real-time performance capture of challenging scenes. *ACM Trans. Graphics*, 35:114:1–114:13, 2016.

[19] SM Ali Eslami, Danilo Jimenez Rezende, Frederic Besse, Fabio Viola, Ari S Morcos, Marta Garnelo, Avraham Ruderman, Andrei A Rusu, Ivo Danihelka, Karol Gregor, et al. Neural scene representation and rendering. *Science*, 360(6394):1204–1210, 2018.

[20] John Flynn, Michael Broxton, Paul Debevec, Matthew DuVall, Graham Fyffe, Ryan Overbeck, Noah Snavely, and Richard Tucker. DeepView: View synthesis with learned gradient descent. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, pages 2367–2376, 2019.

[21] John Flynn, Ivan Neulander, James Philbin, and Noah Snavely. Deepstereo: Learning to predict new views from the world's imagery. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, pages 5515–5524, 2016.

[22] Steven J Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F Cohen. The lumigraph. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 43–54, 1996.

[23] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proc. Int. Conf. on Computer Vision (ICCV)*, pages 2961–2969, 2017.

[24] Peter Hedman, Suhib Alsisan, Richard Szeliski, and Johannes Kopf. Casual 3d photography. *ACM Trans. Graphics*, 36:234:1–234:15, 2017.

[25] Peter Hedman, Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel Brostow. Deep blending for free-viewpoint image-based rendering. *ACM Trans. Graphics*, 37(6):1–15, 2018.

[26] Junhwa Hur and Stefan Roth. Self-supervised monocular scene flow estimation. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, pages 7396–7405, 2020.

[27] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *Proc.*

Computer Vision and Pattern Recognition (CVPR), pages 2462–2470, 2017.

[28] M. Innmann, Kihwan Kim, Jinwei Gu, M. Nießner, Charles T. Loop, M. Stamminger, and J. Kautz. Nrmvs: Non-rigid multi-view stereo. *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 2743–2752, 2020.

[29] Matthias Innmann, Michael Zollhöfer, Matthias Niessner, Christian Theobalt, and Marc Stamminger. VolumeDeform: Real-time volumetric non-rigid reconstruction. In *Proc. European Conf. on Computer Vision (ECCV)*, 2016.

[30] Huaizu Jiang, Deqing Sun, Varun Jampani, Zhaoyang Lv, Erik Learned-Miller, and Jan Kautz. Sense: A shared encoder network for scene-flow estimation. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, pages 3195–3204, 2019.

[31] Huaizu Jiang, Deqing Sun, Varun Jampani, Ming-Hsuan Yang, Erik G. Learned-Miller, and Jan Kautz. Super slomo: High quality estimation of multiple intermediate frames for video interpolation. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, pages 9000–9008, 2018.

[32] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[33] Felix Klose, Oliver Wang, Jean-Charles Bazin, Marcus Magnor, and Alexander Sorkine-Hornung. Sampling based scene-space video processing. *ACM Transactions on Graphics (TOG)*, 34(4):67, 2015.

[34] Suryansh Kumar, Yuchao Dai, and Hongdong Li. Monocular dense 3d reconstruction of a complex dynamic scene from two perspective frames. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 4659–4667, 2017.

[35] 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.

[36] Zhengqi Li, Tali Dekel, Forrester Cole, Richard Tucker, Noah Snavely, Ce Liu, and William T Freeman. Learning the depths of moving people by watching frozen people. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, pages 4521–4530, 2019.

[37] Zhengqi Li, Wenqi Xian, Abe Davis, and Noah Snavely. Crowdsampling the plenoptic function. In *Proc. European Conf. on Computer Vision (ECCV)*, 2020.

[38] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *ACM Trans. Graphics*, 38(4):65, 2019.

[39] Erika Lu, F. Cole, Tali Dekel, Weidi Xie, Andrew Zisserman, D. Salesin, W. Freeman, and M. Rubinstein. Layered neural rendering for retiming people in video. *ACM Trans. Graphics (SIGGRAPH Asia)*, 2020.

[40] Xuan Luo, Jia-Bin Huang, Richard Szeliski, Kevin Matzen, and Johannes Kopf. Consistent video depth estimation. *ACM Trans. Graph.*, 39(4), July 2020.

[41] Zhaoyang Lv, Kihwan Kim, Alejandro Troccoli, Deqing Sun, James M Rehg, and Jan Kautz. Learning rigidity in dynamic scenes with a moving camera for 3d motion field estimation. In *Proc. European Conf. on Computer Vision (ECCV)*, pages 468–484, 2018.

[42] Ricardo Martin-Brualla, N. Radwan, Mehdi S. M. Sajjadi, J. Barron, A. Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. *ArXiv*, abs/2008.02268, 2020.

[43] Simon Meister, Junhwa Hur, and Stefan Roth. Unflow: Unsupervised learning of optical flow with a bidirectional census loss. *arXiv preprint arXiv:1711.07837*, 2017.

[44] Moustafa Meshry, Dan B. Goldman, Sameh Khamis, Hugues Hoppe, Rohit Pandey, Noah Snavely, and Ricardo Martin-Brualla. Neural rerendering in the wild. *Proc. Computer Vision and Pattern Recognition (CVPR)*, pages 6871–6880, 2019.

[45] Ben Mildenhall, Pratul P Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Trans. Graphics*, 38(4):1–14, 2019.

[46] 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. *Proc. European Conf. on Computer Vision (ECCV)*, 2020.

[47] Himangi Mittal, Brian Okorn, and David Held. Just go with the flow: Self-supervised scene flow estimation. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, pages 11177–11185, 2020.

[48] Richard A Newcombe, Dieter Fox, and Steven M Seitz. DynamicFusion: Reconstruction and tracking of non-rigid scenes in real-time. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, 2015.

[49] Simon Niklaus and Feng Liu. Context-aware synthesis for video frame interpolation. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, pages 1701–1710, 2018.

[50] Simon Niklaus and Feng Liu. Softmax splatting for video frame interpolation. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, pages 5436–5445, 2020.

[51] Simon Niklaus, Long Mai, and Feng Liu. Video frame interpolation via adaptive convolution. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, pages 2270–2279, 2017.

[52] Simon Niklaus, Long Mai, and Feng Liu. Video frame interpolation via adaptive separable convolution. In *Proc. Int. Conf. on Computer Vision (ICCV)*, pages 261–270, 2017.

[53] Simon Niklaus, Long Mai, and Oliver Wang. Revisiting adaptive convolutions for video frame interpolation. *arXiv preprint arXiv:2011.01280*, 2020.

[54] Simon Niklaus, Long Mai, Jimei Yang, and F. Liu. 3d ken burns effect from a single image. *ACM Trans. Graphics*, 38:1 – 15, 2019.

[55] Hyun Soo Park, Takaaki Shiratori, Iain Matthews, and Yaser Sheikh. 3d reconstruction of a moving point from a series of 2d projections. In *Proc. European Conf. on Computer Vision (ECCV)*, pages 158–171. Springer, 2010.

[56] Hyun Soo Park, Takaaki Shiratori, Iain A. Matthews, and Yaser Sheikh. 3D Reconstruction of a Moving Point from a Series of 2D Projections. In *Proc. European Conf. on Computer Vision (ECCV)*, 2010.

[57] Eric Penner and Li Zhang. Soft 3d reconstruction for view synthesis. *ACM Trans. Graphics*, 36(6):1–11, 2017.

[58] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. 2020.

[59] Rene Ranftl, Vibhav Vineet, Qifeng Chen, and Vladlen Koltun. Dense monocular depth estimation in complex dynamic scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, 2016.

[60] Gernot Riegler and V. Koltun. Free view synthesis. *Proc. European Conf. on Computer Vision (ECCV)*, 2020.

[61] Chris Russell, Rui Yu, and Lourdes Agapito. Video pop-up: Monocular 3d reconstruction of dynamic scenes. In *Proc. European Conf. on Computer Vision (ECCV)*, pages 583–598. Springer, 2014.

[62] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, pages 4104–4113, 2016.

[63] Lixin Shi, Haitham Hassanieh, Abe Davis, Dina Katabi, and Frédo Durand. Light field reconstruction using sparsity in the continuous fourier domain. *ACM Trans. Graphics*, 34:12:1–12:13, 2014.

[64] Meng-Li Shih, Shih-Yang Su, Johannes Kopf, and Jia-Bin Huang. 3d photography using context-aware layered depth inpainting. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, pages 8028–8038, 2020.

[65] Tomas Simon, Jack Valmadre, Iain A. Matthews, and Yaser Sheikh. Kronecker-Markov Prior for Dynamic 3D Reconstruction. *Trans. Pattern Analysis and Machine Intelligence*, 39:2201–2214, 2017.

[66] Vincent Sitzmann, Justus Thies, Felix Heide, Matthias Nießner, Gordon Wetzstein, and Michael Zollhofer. Deepvoxels: Learning persistent 3d feature embeddings. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, pages 2437–2446, 2019.

[67] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. In *Neural Information Processing Systems*, pages 1119–1130, 2019.

[68] Noah Snavely, Steven M. Seitz, and Richard Szeliski. Photo tourism: Exploring photo collections in 3D. In *ACM Trans. Graphics (SIGGRAPH)*, 2006.

[69] 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 *Proc. Computer Vision and Pattern Recognition (CVPR)*, pages 175–184, 2019.

[70] 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.

[71] Deqing Sun, Erik B Sudderth, and Hanspeter Pfister. Layered rgbd scene flow estimation. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, pages 548–556, 2015.

[72] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *Proc. European Conf. on Computer Vision (ECCV)*, pages 402–419. Springer, 2020.

[73] Justus Thies, Michael Zollhöfer, and Matthias Nießner. Deferred neural rendering: Image synthesis using neural textures. *ACM Trans. Graphics*, 38(4):1–12, 2019.

[74] Richard Tucker and Noah Snavely. Single-view view synthesis with multiplane images. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[75] Suren Vagharshakyan, Robert Bregovic, and Atanas P. Gotchev. Light field reconstruction using shearlet transform. *Trans. Pattern Analysis and Machine Intelligence*, 40:133–147, 2015.

[76] Jack Valmadre and S. Lucey. General trajectory prior for non-rigid reconstruction. pages 1394–1401, 2012.

[77] Minh Vo, S. Narasimhan, and Yaser Sheikh. Spatiotemporal bundle adjustment for dynamic 3d reconstruction. *Proc. Computer Vision and Pattern Recognition (CVPR)*, pages 1710–1718, 2016.

[78] Olivia Wiles, Georgia Gkioxari, R. Szeliski, and J. Johnson. Synsin: End-to-end view synthesis from a single image. *Proc. Computer Vision and Pattern Recognition (CVPR)*, pages 7465–7475, 2020.

[79] Zexiang Xu, Sai Bi, Kalyan Sunkavalli, Sunil Hadap, Hao Su, and Ravi Ramamoorthi. Deep view synthesis from sparse photometric images. *ACM Trans. Graphics*, 38(4), 2019.

[80] Mao Ye and Ruigang Yang. Real-time simultaneous pose and shape estimation for articulated objects using a single depth camera. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, 2014.

[81] 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 *Proc. Computer Vision and Pattern Recognition (CVPR)*, pages 5336–5345, 2020.

[82] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, pages 586–595, 2018.

[83] Enliang Zheng, Dinghuang Ji, Enrique Dunn, and Jan-Michael Frahm. Sparse Dynamic 3D Reconstruction from Unsynchronized Videos. *Proc. Int. Conf. on Computer Vision (ICCV)*, pages 4435–4443, 2015.

[84] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: learning view synthesis using multiplane images. *ACM Trans. Graphics*, 37:1 – 12, 2018.

[85] C Lawrence Zitnick, Sing Bing Kang, Matthew Uyttendaele, Simon Winder, and Richard Szeliski. High-quality video view interpolation using a layered representation. *ACM Trans. Graphics*, 23(3):600–608, 2004.

[86] Michael Zollhöfer, Matthias Niessner, Shahram Izadi, Christoph Rehmann, Christopher Zach, Matthew Fisher, Chenglei Wu, Andrew Fitzgibbon, Charles Loop, Christian Theobalt, et al. Real-time non-rigid reconstruction using an RGB-D camera. *ACM Trans. Graphics*, 33(4):156, 2014.