# Deep 3D Mask Volume for View Synthesis of Dynamic Scenes

Kai-En Lin[1] Lei Xiao[2] Feng Liu[2] Guowei Yang[1] Ravi Ramamoorthi[1]
[1] University of California, San Diego [2] Facebook Reality Labs

## Abstract

*Image view synthesis has seen great success in reconstructing photorealistic visuals, thanks to deep learning and various novel representations. The next key step in immersive virtual experiences is view synthesis of dynamic scenes. However, several challenges exist due to the lack of high-quality training datasets, and the additional time dimension for videos of dynamic scenes. To address this issue, we introduce a multi-view video dataset, captured with a custom 10-camera rig in 120FPS. The dataset contains 96 high-quality scenes showing various visual effects and human interactions in outdoor scenes. We develop a new algorithm, Deep 3D Mask Volume, which enables temporally-stable view extrapolation from binocular videos of dynamic scenes, captured by static cameras. Our algorithm addresses the temporal inconsistency of disocclusions by identifying the error-prone areas with a 3D mask volume, and replaces them with static background observed throughout the video. Our method enables manipulation in 3D space as opposed to simple 2D masks, We demonstrate better temporal stability than frame-by-frame static view synthesis methods, or those that use 2D masks. The resulting view synthesis videos show minimal flickering artifacts and allow for larger translational movements.*

## 1. Introduction

Recent advances in view synthesis have shown promising results in creating immersive virtual experiences from images. Nonetheless, in order to reconstruct compelling and intimate interaction with the virtual scene, the ability to incorporate temporal information is much needed. In this paper, we study a specific setup where the input videos are from static, binocular cameras and novel views are mostly extrapolated from the input videos, similar to the case in StereoMag[48]. We believe that this case is useful as dual- and multi-camera smartphones are gaining traction and it could also prove to be interesting for 3D teleconferencing, surveillance or playback on virtual reality headsets. Moreover, we can acquire the dataset from a static camera rig as shown in Fig.1. Although we can apply state-of-the-art image view synthesis algorithms [48, 41, 28, 37] on each individual video frame, the results lack temporal consistency and often show flickering artifacts. The issues mostly come from the unseen occluded regions as the algorithm predicts them on a per-frame basis. The resulting estimations are not consistent across the time dimension and it causes some regions to become unstable when shown in a video.

In this paper, we address the temporal inconsistency when extrapolating views by exploiting the static background information across time. To this end, we employ a 3D mask volume, which allows manipulation in 3D space as opposed to a 2D mask, to reason about moving objects in the scene and reuse static background observations across the video. As shown in Fig.4, we first promote the instantaneous and background inputs into two sets of multiplane images (MPI)[48] via an MPI network. Then, we warp the same set of input images to create a temporal plane sweep volume, providing information about the 3D structure of the scene. The mask network converts this volume to a 3D mask volume which allows us to blend between the two sets of MPIs. Finally, the blended MPI volume can render novel views with minimal flickering artifacts.

To train this network, we also introduce a new multi-view video dataset to address the lack of publicly available data. We build a custom camera rig comprised of 10 action cameras and capture high-quality 120FPS videos with the static rig (see Fig.1). Our dataset contains 96 dynamic scenes of various outdoor environments and human interactions. We show that the proposed method generates temporally stable results against previous state-of-the-art methods, while only using two input views.

Our contributions can be summarized as:

- a multi-view video dataset composed of 96 dynamic scenes (Sec. 3);
- a novel 3D volumetric mask able to segment dynamic objects from static background in 3D, producing higher-quality and temporally stable results than state-of-the-art methods (Sec. 4.2).

## 2. Related Work

Our goal is to achieve temporally stable view synthesis on dynamic scenes. We are inspired by several previous methods in view synthesis and space-time synthesis.

### 2.1. View synthesis

View synthesis is a complicated problem which has become a popular field of research in computer vision and graphics. Earlier lines of work utilize dense sampling from


Figure 1. Our custom camera rig. Top left figure shows the configuration we use for evaluation in Sec. 5. We show the input stereo image sequences from camera 4 and camera 5 in the middle. Rightmost column shows the crops of rendered novel view at camera 0. Artifacts appear when the novel view is translated by a larger distance. We use conventional MPI method [28] as our baseline algorithm. Note how the area on top of the person's head is distorted and shows "stack of cards" artifacts. This type of artifact flickers in a dynamic video as the network hallucinates the disocclusion per-frame.

the scene to create light fields [14, 20]. Image-based rendering techniques [7, 10] exploit proxy geometry of the scene to produce novel view renderings. Later extensions on this topic introduce better modeling of the scene structure [36] and hand-crafted heuristics [9, 33]. As deep learning became dominant, learning-based methods [17, 12, 19, 45, 30] have shown promising results. Recently, a class of research works focuses on combining novel representations [48, 29, 28, 41, 11, 18, 25, 39, 31, 23] with a differentiable rendering pipeline to produce high-quality results. Another exciting advance is neural radiance fields (NeRF) [29], which encodes the 3D scene structure in a compact continuous 5D volumetric function. Although NeRF has shown promising view synthesis results, it has to overfit to the given scene with enough samples (10 or more), requiring time-consuming per-scene training. Rendering time could take up to 30s for one image, whereas our pipeline allows inference and rendering in less than 2s without dedicated optimization, using only binocular input views.

Instead, in this paper we focus on a specific layered representation, multiplane images (MPI) [42, 48, 41, 28, 6], as it provides good generalizability across various scenes and efficiency capable of real-time rendering. Our proposed method directly tackles the temporal instability introduced in MPIs when the disoccluded areas lead to different estimations across time.

### 2.2. Space-time synthesis

Space-time synthesis is a more complicated problem since it not only involves movement of the novel viewpoint in space, but also incorporates differences of time. A body of work covers appearance changes such as relighting while changing views [46, 45, 4, 3, 27]. However, these methods focus on the lighting change with respect to a static scene, treating dynamic objects in the scene as outliers. On the other hand, some methods directly target dynamic scenes [6, 1, 2, 47, 49]. While our method utilizes MPIs similar to Broxton et al.[6], they employ dense sampling of 46 cameras to reconstruct light fields of the viewing volume, essentially interpolating between cameras. Our method focuses on the stereo case similar to StereoMag[48], targeting extrapolation from stereo inputs like dual-camera smartphones. In addition, unlike depth-based methods [1, 47], we do not require any explicit depth maps to render novel viewpoints. As depth-based methods often yield flickering and require hole-filling, we instead use a representation more suitable for rendering. Another issue that these methods do not address is the lack of generalizability. Bansal et al. [1] is trained on limited data which could make the learned network overfit to a small number of scenes. Moreover, while Yoon et al. [47] uses a pretrained network to ensure generalizability on unseen scenes, it still requires human-generated masks for foreground and background separation. We capture various dynamic scenes with human interactions to train our network and ensure that it is generalizable across different unseen scenes. Also, our network utilizes the background information extracted from video and uses it to directly segment the foreground and background in 3D space without any human input.

## 3. Dataset

High-quality video datasets are crucial for learning-based novel-view video synthesis algorithms. The ideal datasets would contain a diversity of scenes, captured at multiple synchronized views. In this work we introduce a novel multi-view video datasets. We discuss the limitations of existing datasets compared to our dataset in Sec. 3.1. We describe our data capture and generation process in Sec. 3.2. Finally, we discuss the statistics and advanced properties of our dataset in Sec. 3.3.

### 3.1. Multi-view video dataset

As shown in Table 1, we evaluate several properties which are important to train a generalized view synthesis

| Dataset | Scene count | Rigid rig | Large disparity | Views | Dynamic | Public | Remarks |
|---|---|---|---|---|---|---|---|
| Real Forward-Facing [28] | 65 | ✗ | ✓ | 25 | ✗ | ✗ | Loosely gridlike formation |
| Spaces [11] | 100 | ✓ | ✓ | 16 | ✗ | ✓ | Strictly gridlike formation |
| Immersive LF Video [6] | 130 | ✓ | ✓ | 46 | ✓ | ✗ | Spherical formation |
| Dynamic Scene [47] | 8 | ✓ | ✓ | 12 | ✓ | ✓ | Few temporal frames |
| Single Image LF [21] | ∼2000 | ✓ | ✗ | 196 | ✗ | ✓ | Small baseline light fields |
| RealEstate10K [48] | ∼10000 | ✗ | ✓ | 1 | ✗ | ✓ | Static scenes |
| Open4D [1] | 6 | ✗ | ✓ | 15 | ✓ | ✓ | Free-viewpoint capture |
| MannequinChallenge [22] | ∼2000 | ✗ | ✓ | 1 | ✗ | ✓ | Mostly static scenes |
| X-Fields [2] | 8 | ✓ | ✓ | 5 | ✓ | ✓ | Few temporal frames |
| KITTI [26] | 400 | ✓ | ✓ | 2 | ✓ | ✓ | Binocular setup on cars |
| **Ours** | 96 | ✓ | ✓ | 10 | ✓ | ✓ | Publicly released |

Table 1. Comparison of different multi-view datasets.


Figure 2. Digital clock and the randomly moving QR code pattern used to perform synchronization. We have two ways to do synchronization: (1) matching the timestamp; (2) aligning the QR code location in all views. We use these methods to ensure the synchronization is accurate enough.

network. Specifically, a rigid camera rig is preferred as it can provide good pose priors and ensure the accuracy of the estimated camera poses. On the contrary, unstructured captures like Real Forward-Facing [28] and Open4D [1] do not use pose priors and utilize structure from motion, which could produce varying accuracy depending on scene geometry and the texture presented. In addition, rigid camera rigs allow for capture of dynamic scenes with multiple simultaneous camera views. On account of the above reasons, our dataset is captured with a custom camera rig that is rigid and robust enough to offer good pose priors.

Number of views is also an important factor for a multi-view dataset since different combinations of input and target camera pairs provide diversity in baselines and camera motions. X-Fields [2] and KITTI [26] provide limited views and camera motions and thus are not as useful for video view synthesis tasks. Our dataset offers 10 different camera views in a gridlike formation (see Fig. 1). For our binocular view synthesis task, we choose 2 views out of 10 and 1 from the rest to construct a training pair.

The most important feature is to have enough temporal frames and dynamic movements for training. Most datasets fail at this part as they target the image view synthesis task instead of a video one. Although Dynamic Scene Dataset presented by Yoon et al.[47] targets dynamic scenes, it uses frame skips to keep salient movements. Thus, the movements shown in the dataset are not smooth and fail to provide enough training samples. To address this issue, our dataset is captured in 120 FPS and synchronized as a post-process (see Sec. 3.2), making it easy to perform and evaluate view synthesis at different framerates.

One dataset that targets the purpose of video view synthesis is the Immersive Light Field Video dataset proposed by Broxton et al.[6], which contains 46 camera views and 130 different dynamic scenes. However, the full dataset is not publicly available to the community. Our full dataset can be found at http://cseweb.ucsd.edu/~viscomp/projects/ICCV21Deep/

| **Occlusion Types** | **(a)** | **(b)** | **(c)** | **(d)** | **Total videos** |
|---|---|---|---|---|---|
| Count | 90 | 96 | 42 | 19 | 96 |

Table 2. Number of videos that contain each occlusion type as described in Sec. 3.3. Note that most scenes typically contain multiple types of occlusion.

### 3.2. Dataset generation

Our video dataset is captured with a custom camera rig that consists of 10 GoPro Hero 7 Black action cameras as shown in Fig. 1. The horizontal baseline between neighbor cameras is approximately 10 cm and the vertical distance between rows is around 14 cm. We captured 96 outdoor videos in 120 FPS, with the camera rig being static for each video. As GoPros only allow fisheye mode for high FPS captures, we calibrate the cameras with a 17x14 checkerboard pattern (squares have side lengths of 40mm) and undistort the videos using a pinhole camera model [13] implemented in OpenCV [5]. For camera extrinsics, we choose the first frame from all views as inputs to COLMAP [35, 34], which then does feature extraction, feature matching, and sparse reconstruction. The reconstructed camera poses are assumed to be fixed for the duration of each video. In addition, to achieve synchronization, we display a digital clock with randomly appearing QR code patterns (see Fig. 2) on a high refresh rate screen that can be seen by all cameras at the same time. Then, we manually edit and align the multi-view videos according to the digital clock and QR code pattern.

### 3.3. Dataset statistics

Our videos are mostly around 1 to 2 minutes long and all videos are shot in 120 FPS. We cover different scenes to ensure that the surface reflectance variety is high enough. For example, in Fig.3 we show that in our dataset we cover different buildings, furniture, foliage and specularity effects.


Figure 3. A selection of still frames from our dataset. We captured various dynamic scenes with human motions, including walking, running, jumping and sitting down. Note that cameras remain static for the whole duration of the capture.

Another important aspect of our dataset is the inclusion of different human motions, including slower motions like walking, sitting down and faster motions, such as running, jumping and arms waving. We now discuss four possible types of occlusion interactions and show the numbers of their occurrences in Table 2.

**(a) Static occluder and static background.** Most view synthesis methods target this case as this is one of the most common cases. We desribe it as a static occluder in the scene blocking the line-of-sight from the cameras to the background scene. For instance, the table in the sitting scene shown in Fig. 3 occludes the areas behind. Background information can only be acquired from the views with direct line-of-sight. As such, it is difficult to recover the unseen regions without prior knowledge of the scene. However, temporal consistency in these areas is easily achievable because inputs remain relatively unchanged throughout the video. Hallucination of the disoccluded areas can also remain the same for this case.

**(b) Dynamic occluder and static background.** Another type of event happens when a dynamic object is moving across the scene. For example, when a person is walking through the scene, the camera has line-of-sight on the background behind the person at some point in the video. In this case, it is relatively easy to acquire static background information as the occluder does not block the line-of-sight in all video frames. Combining information from multiple frames throughout the video provides an accurate rendering of what is behind the dynamic occluder. Temporal consistency in this case can also be maintained by substituting the static background for the dynamically-occluded regions. In other words, we can perform hole-filling based on the observations from other video frames. Our proposed method takes advantage of this prior knowledge to generate temporally-stable view synthesis results, as opposed to previous methods.

**(c) Static occluder and dynamic background.** This case happens when an object moves behind a static occluder and thus the camera does not have full visuals on it. For instance, a person walks behind a traffic sign or a wall. In the traffic sign case, as it is only a short-term occlusion, the person's appearance can be interpolated between different frames. However, in the case of a larger wall, this becomes difficult to solve as extrapolating the movement is complicated and the ambiguity could lead to different outcomes. In general, it is difficult to accurately predict the trajectory of the occluded object without assuming it is moving at constant velocity. For temporal consistency, the movement of dynamic objects can lead to instability of the novel view prediction. Our method learns to detect the dynamic movements and treat the static part of the scene as (a) such that flickering artifacts are kept at a minimal level.

**(d) Dynamic occluder and dynamic background.** The last case happens when the occluder and the background object are both moving or the background appearance is changing. For instance, this can happen when two people are walking in the opposite direction parallel to the camera's image plane. Similar to (c), how the occluded object is moving remains ambiguous and hard to resolve deterministically. Although we do not have a clear idea of the occluded parts, we can still ensure it is temporally stable when shown. We can reduce this case to (b) with the ambiguity that the occluded object can move anywhere. And as a result, the occluded regions look more or less similar to the static background.

Our dataset contains diverse occlusion interactions and we show results in Sec.5.2 and provide an analysis in Fig.6.

## 4. Deep 3D Mask Volume

Our goal is to synthesize temporally consistent novel view videos given stereo video inputs. Consequently, we build our algorithm upon prior work on multiplane images [48, 28] and propose a novel mask volume structure to fully utilize the temporal background information and the layered representation. In this section, we start with a brief review of the multiplane images in Sec. 4.1. Then we describe our 3D mask volume in Sec. 4.2. Finally we discuss our loss function design in Sec. 4.3. Please refer to Fig.4 for an overview of our algorithm pipeline.

### 4.1. Multiplane images

Our approach takes inspiration from recent advancements in multiplane image representation [43, 48]. Multiplane images (MPI) are a layered representation of the 3D scene. They consist of $D$ layers of RGB$\alpha$ images, representing the viewing frustum from the perspective of a virtual reference camera. The planes partition the viewing frustum according to equally-spaced disparity (inverse depth) values $d_0, d_1, ..., d_{D-1}$. Each layer of the MPI encodes color $C$ and transparency information $\alpha$ at a specified plane depth $d$. We denote the MPI layer at disparity $d$ as a tuple of $(C_d, \alpha_d)$. To construct such a volume, we warp input views to the reference camera position to construct a plane sweep volume (PSV). The PSV is then used as the input to a 3D CNN similar to the one used by Mildenhall et al.[28] and it generates the corresponding MPI volume. To render a novel viewpoint $j$ from camera $i$, the MPI layers are warped using planar homography as follows:

$$\mathcal{W}^d_{i\to j}(C_d, \alpha_d), \tag{1}$$

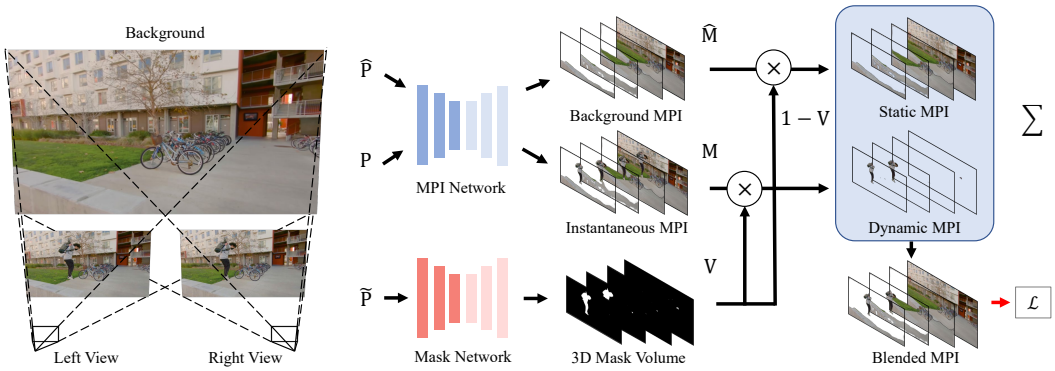


Figure 4. Overview of our pipeline. Given binocular input videos, our MPI network promotes the 2D multiview images to two 3D MPI representations; one encodes the instantaneous information and the other encodes the background information. The mask network produces a 3D mask volume $\mathbf{V}$ to modulate the MPIs and blend them together, producing the final output. Please see Sec. 4.2 for more details.

where $\mathcal{W}$ is the warping operator. The warped MPIs are then composited with the over operation. To be more specific, we calculate the per-pixel transmittance $t$ from the alpha value at location $(x, y)$ on plane $d$ by

$$t(x, y, d) = \alpha(x, y, d) \prod_{d' > d} [1 - \alpha(x, y, d')]. \quad (2)$$

The final rendering at each pixel $C_{final}$ is computed as

$$C_{final}(x, y) = \sum_{d} C(x, y, d)\alpha(x, y, d) \prod_{d' > d} [1 - \alpha(x, y, d')]. \quad (3)$$

These computations are parallelizable and their efficiency during rendering makes the MPI a good representation for fast view synthesis.

One observation of MPIs is that the unseen parts in the volume are often merely repeated texture of the foreground objects [41]. This happens when the input camera baseline is not large enough and the resulting PSV cannot provide further information about the background. In addition, these areas typically present different estimations between frames. Therefore, the unseen areas produce visible artifacts, especially in video view synthesis (see Fig. 1). On the other hand, visible parts usually provide temporally stable results as can be seen in Broxton et al. [6]

### 4.2. 3D mask volume generation

From Sec. 4.1, we observe that most artifacts are introduced by the disocclusion of moving objects. In order to address this issue, we seek to find a 3D mask volume that identifies the dynamic components and removes the flickering artifacts behind them accordingly. To be more specific, given a pair of stereo image sequences of length $n$, $\{\mathbf{I}_0^L, \mathbf{I}_1^L, ..., \mathbf{I}_{n-1}^L\}$ and $\{\mathbf{I}_0^R, \mathbf{I}_1^R, ..., \mathbf{I}_{n-1}^R\}$, we wish to derive a 3D mask $\mathbf{V}(x, y, d)$, such that

$$\mathbf{V}(x, y, d) = \begin{cases} 1, & \text{if } \mathbf{I}(x, y) \neq \hat{\mathbf{I}}(x, y), d > \mathbf{D}(x, y) \\ 0, & \text{otherwise} \end{cases}, \quad (4)$$

where $\mathbf{I}$ is the instantaneous frame, $\hat{\mathbf{I}}$ denotes the background image, and $\mathbf{D}$ is the scene disparity observed by the camera. We drop the frame subscript as a shorthand for instantaneous frame in the following discussion. In addition, we represent the instantaneous MPI of the scene as $\mathbf{M}(x, y, d)$, and the background MPI as $\hat{\mathbf{M}}(x, y, d)$.

The main purpose of the 3D mask volume $\mathbf{V}(x, y, d)$ is to partition the scene $\mathbf{M}(x, y, d)$ into two parts: static and dynamic. The static portion of the MPI does not change for the whole video duration, and thus $\mathbf{M}(x, y, d) = \hat{\mathbf{M}}(x, y, d)$, when $\mathbf{V}(x, y, d) = 0$. The synthesized novel view of these parts is temporally stable and requires no further modification to the algorithm. On the contrary, the dynamic objects ($\mathbf{V}(x, y, d) = 1$) could move in different directions. The disoccluded areas, given mathematically by $\mathbf{M}(x, y, d)$ if $\mathbf{I}(x, y) \neq \hat{\mathbf{I}}(x, y), d > \mathbf{D}(x, y)$, often change with them, producing "stack of card" artifacts and flickering when viewed from another angle (see Fig. 1). However these areas in fact usually resemble the background $\hat{\mathbf{I}}$. With this knowledge, a clear separation between the static and dynamic scene components allows us to identify the disocclusion and minimize the temporal inconsistency by

$$\mathbf{M}(x, y, d) \leftarrow \hat{\mathbf{M}}(x, y, d) \text{ if } \mathbf{I}(x, y) \neq \hat{\mathbf{I}}(x, y), d < \mathbf{D}(x, y). \quad (5)$$

Essentially, we are using the temporally-stable static background to replace the unknown disoccluded areas. An illustration of the mask is given in Fig. 4.

In order to perform the operation in Eq.5, our network is composed of two networks: **MPI network** generates 2 layered representations of the 3D scene, namely $\mathbf{M}(x, y, d)$ and $\hat{\mathbf{M}}(x, y, d)$; **Mask network** produces the 3D mask volume $\mathbf{V}(x, y, d)$ satisfying Eq.4. We show each network in Fig.4 and discuss them in details as follows:

**MPI network.** It is necessary to acquire 3D information from both the instantaneous frame and throughout the whole video, so we can then obtain the needed information behind the dynamic occluder. To this end, we first apply a

median filter $\mathcal{A}$ on the image sequences

$$\hat{\mathbf{I}} = \mathcal{A}(\{\mathbf{I}_0, \mathbf{I}_1, ..., \mathbf{I}_{n-1}\}). \quad (6)$$

It is applied to both views to generate the corresponding background images.

Then, we can inversely warp $\mathbf{I}^R$ and $\hat{\mathbf{I}}^R$ to the left camera and construct a PSV. The PSV from the instantaneous frame is generated as

$$\mathbf{P} = \{\mathbf{I}^L, \mathcal{W}^{d_0}_{R\to L}(\mathbf{I}^R), \mathcal{W}^{d_1}_{R\to L}(\mathbf{I}^R), ..., \mathcal{W}^{d_{D-1}}_{R\to L}(\mathbf{I}^R)\}. \quad (7)$$

It is then used as an input to a 3D CNN $\mathcal{F}_\theta$ to produce the instantaneous MPI, $\mathbf{M} = \mathcal{F}_\theta(\mathbf{P})$. Similarly, we construct the background MPI, $\hat{\mathbf{M}}$, using another PSV, $\hat{\mathbf{P}}$, generated from $\hat{\mathbf{I}}^L$ and $\hat{\mathbf{I}}^R$. The two MPIs, $\mathbf{M}$ and $\hat{\mathbf{M}}$, now contain the information of the dynamic occluder and the static background.

**Mask network.** We utilize another 3D CNN $\mathcal{G}_\theta$ to reason about the relationship between the MPIs and generate a mask volume $\mathbf{V}$ to satisfy Eq.4. Inspired by background matting [24] on 2D images, our mask network takes a similar approach but in 3D space. From Eq.6, we define a temporal plane sweep volume (TPSV) as follows

$$\tilde{\mathbf{P}} = \{\mathbf{I}^L, \mathcal{W}^{d_0}_{R\to L}(\mathbf{I}^R), ..., \mathcal{W}^{d_{D-1}}_{R\to L}(\mathbf{I}^R), \hat{\mathbf{I}}^L, \mathcal{W}^{d_0}_{R\to L}(\hat{\mathbf{I}}^R), ..., \mathcal{W}^{d_{D-1}}_{R\to L}(\hat{\mathbf{I}}^R)\}. \quad (8)$$

The TPSV helps the network to distinguish the dynamically-occluded parts in the 3D scene. Then, we acquire the 3D mask volume by $\mathbf{V} = \mathcal{G}_\theta(\tilde{\mathbf{P}})$.

Finally, we can calculate the final MPI $\mathbf{M}_o$ by:

$$\mathbf{M}_o(x, y, d) = \mathbf{M}(x, y, d)\mathbf{V}(x, y, d) + \hat{\mathbf{M}}(x, y, d)(1 - \mathbf{V}(x, y, d)), \quad (9)$$

for all $(x, y, d)$. We define a shorthand version as

$$\mathbf{M}_o = \mathbf{M} \odot \mathbf{V} + \hat{\mathbf{M}} \odot (1 - \mathbf{V}), \quad (10)$$

where $\odot$ means element-wise multiplication. $\mathbf{M}_o$ achieves Eq.5 as our learnable mask volume $\mathbf{V}$ satisfies Eq.4 and we can then render the output image $\mathbf{I}_o$ using planar homography and the over composite operation described in Sec. 4.1. Please refer to Fig.4 for illustrations.

One major difference between using a 3D mask volume $\mathbf{V}(x, y, d)$ and a 2D mask $\mathbf{V}'(x, y)$ is that the former is able to segment out the dynamic objects in the 3D space, namely Eq.4 and subsequently do Eq.5. In Fig.4, notice that the mask volume only contains the dynamic object (jumping person in this case). In contrast, a 2D mask $\mathbf{V}'(x, y)$ does not vary with respect to the disparity $d$, making it impossible to manipulate the areas behind dynamic objects.

### 4.3. Loss function

We implement our loss function as a rendering loss, similar to previous work on MPIs [48, 41, 28]. For the rendering loss, we use view synthesis as the supervision task and let the algorithm render a held-out view from the final MPI $\mathbf{M}_o$ (see Fig.4). The rendering loss is as follows:

$$\mathcal{L} = \frac{||\mathcal{F}_{VGG}(\mathbf{I}_o) - \mathcal{F}_{VGG}(\mathbf{I}_{gt})||_1}{N}, \quad (11)$$

where $\mathcal{F}_{VGG}$ is the VGG-19 network [38], $N$ is the number of elements in the image $\mathbf{I}_o$, and $\mathbf{I}_{gt}$ is the held-out ground truth view. This perceptual loss is similar to the implementation of Chen et al. [8]. We also considered a mask supervision loss $\mathcal{L}_m$ and a mask sparsity constraint $\mathcal{L}_s$. However, we did not find them to be useful for temporal consistency. Ablation studies on these two losses can be found later in Table 4, and details are in the supplementary materials.

## 5. Results

In this section, we discuss implementation details for our network in Sec. 5.1. Then we show comparisons to other methods on our dataset in Sec. 5.2. Finally we discuss limitations of our current setup and method in Sec. 5.3. Result videos can be found in the supplementary materials.

### 5.1. Implementation details

Due to GPU memory constraints, we choose a two-step training scheme to train our network. We first train the MPI network on RealEstate10K dataset [48], and then train only the mask network on our own video dataset. This training scheme can keep the memory usage within a reasonable range and the speed fast enough.

The MPI generation network is trained by predicting a held-out novel view and applying the rendering loss $\mathcal{L}$ as supervision. This stage is trained for 800K steps. After the previous pretraining stage, we freeze the weights of the MPI network and train only the mask network using the loss $\mathcal{L}$. The network takes 2 random views from the 10 views as input and we randomly choose a target camera position from the rest of the views at each step. We select 86 out of the 96 scenes as our training dataset and images are rescaled to 640×360. This second stage is trained for 100K steps. The learning rate is set to $5e-4$ for both stages. Our training pipeline is implemented in PyTorch[32] and training takes around 5 days on a single RTX 2080Ti GPU. With resolution in 640×360, inferencing $\mathbf{M}_o$ using our full pipeline takes around 1.75s, while rendering takes another 0.28s. Note that the rendering pipeline is implemented in PyTorch without further optimization. In practice, it could be significantly faster with OpenGL or other rasterizer.

### 5.2. Comparisons

For comparison, we choose 7 unseen videos from the dataset and subdivide them into 14 clips, focusing on salient movements in the scene. We ran all methods on the clips with camera 4 and 5 as input and others as the target output (see Fig.1). Error metrics are calculated between the output and the ground truth images. We compare with three baseline approaches: (1) MPI/LLFF is our adaptation of Mildenhall et al.[28] to work with only two input views and different camera intrinsics. It processes the stereo input videos


Figure 5. We show visual results on 4 different scenes. These scenes include both fast and slow movements, such as waving, jumping and walking. The novel viewpoint is an extrapolation from the input camera views. Our proposed method produces results with fewer artifacts and more temporal stability.

| Methods | Mask | STRRED↓ | PSNR↑ | SSIM↑ |
|---|---|---|---|---|
| MPI/LLFF [28] | No Mask | 0.2917 | 25.52 | 0.8227 |
| 2D Mask | 2D | 0.2892 | 25.50 | 0.8242 |
| IBRNet (2-view) [44] | No Mask | 2.2606 | 21.49 | 0.6713 |
| **Ours** | 3D | **0.1683** | **26.22** | **0.8390** |

Table 3. Comparison on our evaluation dataset. We compare with different baseline methods and the results show that our 3D mask offers much better temporal stability. 2D mask does not improve much because it fails to resolve the ambiguity in disoccluded areas.

and renders the novel view frames on a per-frame basis. (2) 2D mask is our naive baseline method, which is similar to our pipeline, except that it uses a foreground mask $\mathbf{V}'(x, y)$ generated by the background matting method[24] with $\mathbf{I}$ and $\hat{\mathbf{I}}$ as inputs. The blended MPI $\mathbf{M}'_o$ for (2) is obtained by

$$\mathbf{M}'_o = \mathbf{V}'(x, y) \odot \mathbf{M} + (1 - \mathbf{V}'(x, y)) \odot \hat{\mathbf{M}},$$

where the 2D mask has been expanded into 3D by repeating its values along the depth dimension. (3) IBRNet uses the official implementation and takes 2 views as input on a per-frame basis. Please refer to our supplementary materials for the video results.

From Table 3, we see that our method is able to achieve temporally-coherent rendering, while offering better visual quality and fewer distortions. Specifically, we employ the STRRED metric [40] to evaluate stability across time. Our method significantly reduces the temporal artifacts across most scenes while also keeping PSNR and SSIM better than the baseline methods. For MPI/LLFF, since it does not utilize the information across the whole video, it yields more flickering and distorted areas as can be seen in Fig. 5. For example, in the top scene, there is a ghosting artifact around the person's head and it changes frame-by-frame, resulting in flickering video. The 2D mask method is a binary mask that naively selects the dynamic parts in $\mathbf{M}$ and the background in $\hat{\mathbf{M}}$ to produce the final MPI. As a result, it amplifies the stack of cards artifacts (see Fig. 5) and also slightly worsens the visual quality as shown in Table 3. IBRNet [44], does not work well with 2-view input and it produces poor results compared to ours. In the second row of Fig. 5, MPI/LLFF is able to hallucinate reasonable disocclusion similar to the background, whereas 2D mask repeats the texture of the person for those areas. This is because 2D mask incorrectly blends the dynamic and static parts of the scene.

To further analyze how temporal consistency is affected,

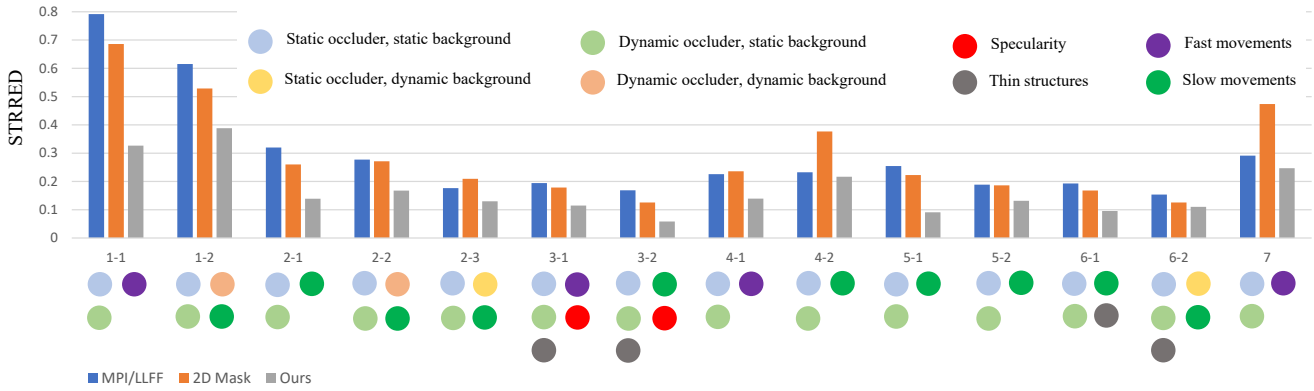


Figure 6. STRRED comparison on our dataset with baseline methods. We select 14 clips from 7 different scenes. 1-1, 1-2 denotes clip 1 and clip 2 from scene 1.

| Methods | STRRED↓ | PSNR↑ | SSIM↑ |
|---|---|---|---|
| Ours | **0.1683** | **26.22** | 0.8390 |
| Ours w/ $\mathcal{L}_s$ | 0.1745 | 26.18 | **0.8393** |
| Ours w/ $\mathcal{L}_s, \mathcal{L}_m$ | 0.1900 | 26.09 | 0.8374 |

Table 4. Effect of different loss functions. Our rendering loss offers better temporal consistency and slightly better visual quality.

we characterize the clips with different properties including different types of occlusion discussed in Sec. 3.3 and show the results in Fig. 6. As stated earlier, several clips are selected from the 7 scenes to show salient motions. From the results, we observe that faster movements could often result in worse temporal consistency, like the differences between clip 1-1 and 1-2. There is an interesting failure in 4-2 for the 2D mask method. 4-1 is the jumping scene in Fig.5, and 4-2 shows a person walking in the same scene. Although the movement is slower, the person walks past several areas with large appearance changes in 4-2. As a result, the artifacts in the 2D mask are much more obvious, and the video flickers more than other methods, leading to a worse STRRED score.

As shown in Table 4, our rendering loss still offers the most temporally-stable results, whereas the other two losses (mask supervision $\mathcal{L}_m$ and sparsity $\mathcal{L}_s$) trade temporal consistency for better interpretability. We provide more details and mask visualizations in the supplementary materials.

### 5.3. Limitations

The proposed dataset and algorithm have a few limitations: First, we limit our camera to stay static when capturing. This is mainly due to the limitations of synchronization and pose estimation. Although we can achieve good synchronization with software-based methods, there are still a few milliseconds of error. This error could be magnified when the camera rig is in motion and lead to bad estimates of the camera poses. The camera poses across time would also require more calculations, possibly leading to accumulating errors in the system. These issues could be solved by calibrating the camera trajectory of one of the cameras and utilizing the rigid assumption to infer the trajectories of other cameras. Another limitation is that we require an estimate of the static background. This is easily achievable by applying a median filter. While it works for most of the scenes, this method is sometimes not reliable. There are more advanced approaches[16, 15] that can be used in the future.

## 6. Conclusions and Future Work

In this paper, we discuss view synthesis of dynamic scenes with stereo input videos. The main challenge is that rendered results are prone to temporal artifacts like flickering in the disoccluded regions. To tackle this issue, we introduce a novel 3D mask volume extension to carefully replace the disoccluded areas with background information acquired from the temporal frames. Additionally, we introduce a high-quality multi-view video dataset, which contains 96 scenes of various human interactions and outdoor environments shot in 120FPS.

In future work, we would like to extend our dataset and method to consider dynamic camera motions, and to operate on even larger baselines. In summary, we believe video view synthesis for dynamic scenes is the next frontier for immersive applications, and this paper has taken a key step in that direction.

## 7. Acknowledgement

This work was supported in part by ONR grant N000142012529, ONR grant N000141912293, NSF grant 1730158, all awarded to the UCSD researchers, a Facebook Distinguished Faculty Award, the Ronald L. Graham Chair, and the UC San Diego Center for Visual Computing. Part of the work was done when KEL was an intern at Facebook. We also acknowledge a Qualcomm FMA Fellowship and an Amazon Research Award. Lastly, we thank Yuzhe Qin, Dominique Meyer, Eric Lo, Thomas DeFanti, Jürgen Schulze and Michael Broxton for comments on hardware setup.

## References


[1] Aayush Bansal, Minh Vo, Yaser Sheikh, Deva Ramanan, and Srinivasa Narasimhan. 4d visualization of dynamic events from unconstrained multi-view videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5366–5375, 2020. 2, 3

[2] Mojtaba Bemana, Karol Myszkowski, Hans-Peter Seidel, and Tobias Ritschel. X-fields: Implicit neural view-, light- and time-image interpolation. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia 2020)*, 39(6), 2020. 2, 3

[3] Sai Bi, Zexiang Xu, Pratul Srinivasan, Ben Mildenhall, Kalyan Sunkavalli, Miloš Hašan, Yannick Hold-Geoffroy, David Kriegman, and Ravi Ramamoorthi. Neural reflectance fields for appearance acquisition. *arXiv preprint arXiv:2008.03824*, 2020. 2

[4] 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. *arXiv preprint arXiv:2007.09892*, 2020. 2

[5] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000. 3

[6] 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. 39(4):86:1–86:15, 2020. 2, 3, 5

[7] 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. 2

[8] Qifeng Chen and Vladlen Koltun. Photographic image synthesis with cascaded refinement networks. In *Proceedings of the IEEE international conference on computer vision*, pages 1511–1520, 2017. 6

[9] Abe Davis, Marc Levoy, and Fredo Durand. Unstructured light fields. In *Computer Graphics Forum*, volume 31, pages 305–314. Wiley Online Library, 2012. 2

[10] Paul E Debevec, Camillo J Taylor, and Jitendra Malik. Modeling and rendering architecture from photographs: A hybrid geometry-and image-based approach. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 11–20, 1996. 2

[11] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2367–2376, 2019. 2, 3

[12] John Flynn, Ivan Neulander, James Philbin, and Noah Snavely. Deepstereo: Learning to predict new views from the world's imagery. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5515–5524, 2016. 2

[13] David A. Forsyth and Jean Ponce. *Computer Vision: A Modern Approach*. Prentice Hall Professional Technical Reference, 2002. 3

[14] 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. 2

[15] Soren Hauberg, Aasa Feragen, and Michael J Black. Grassmann averages for scalable robust pca. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3810–3817, 2014. 8

[16] Jun He, Laura Balzano, and Arthur Szlam. Incremental gradient on the grassmannian for online foreground and background separation in subsampled video. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1568–1575. IEEE, 2012. 8

[17] Peter Hedman, Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel Brostow. Deep blending for free-viewpoint image-based rendering. *ACM Transactions on Graphics (TOG)*, 37(6):1–15, 2018. 2

[18] Chiyu Jiang, Avneesh Sud, Ameesh Makadia, Jingwei Huang, Matthias Nießner, Thomas Funkhouser, et al. Local implicit grid representations for 3d scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6001–6010, 2020. 2

[19] Nima Khademi Kalantari, Ting-Chun Wang, and Ravi Ramamoorthi. Learning-based view synthesis for light field cameras. *ACM Transactions on Graphics (TOG)*, 35(6):1–10, 2016. 2

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

[21] Qinbo Li and Nima Khademi Kalantari. Synthesizing light field from a single image with variable mpi and two network fusion. *ACM Transactions on Graphics*, 39(6), 12 2020. 3

[22] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4521–4530, 2019. 3

[23] Kai-En Lin, Zexiang Xu, Ben Mildenhall, Pratul P Srinivasan, Yannick Hold-Geoffroy, Stephen DiVerdi, Qi Sun, Kalyan Sunkavalli, and Ravi Ramamoorthi. Deep multi depth panoramas for view synthesis. In *ECCV*, 2020. 2

[24] Shanchuan Lin, Andrey Ryabtsev, Soumyadip Sengupta, Brian Curless, Steve Seitz, and Ira Kemelmacher-Shlizerman. Real-time high-resolution background matting. *arXiv*, pages arXiv–2012, 2020. 6, 7

[25] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *arXiv preprint arXiv:1906.07751*, 2019. 2

[26] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 3

[27] Moustafa Meshry, Dan B Goldman, Sameh Khamis, Hugues Hoppe, Rohit Pandey, Noah Snavely, and Ricardo Martin-Brualla. Neural rerendering in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6878–6887, 2019. 2

[28] Ben Mildenhall, Pratul P. Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Transactions on Graphics (TOG)*, 2019. 1, 2, 3, 4, 6, 7

[29] 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 *European Conference on Computer Vision*, pages 405–421. Springer, 2020. 2

[30] Simon Niklaus, Long Mai, Jimei Yang, and Feng Liu. 3d ken burns effect from a single image. *ACM Transactions on Graphics (TOG)*, 38(6):1–15, 2019. 2

[31] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 165–174, 2019. 2

[32] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 6

[33] Eric Penner and Li Zhang. Soft 3d reconstruction for view synthesis. *ACM Transactions on Graphics (TOG)*, 36(6):1–11, 2017. 2

[34] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 3

[35] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, 2016. 3

[36] Jonathan Shade, Steven Gortler, Li-wei He, and Richard Szeliski. Layered depth images. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 231–242, 1998. 2

[37] Meng-Li Shih, Shih-Yang Su, Johannes Kopf, and Jia-Bin Huang. 3d photography using context-aware layered depth inpainting. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 1

[38] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 6

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

[40] Rajiv Soundararajan and Alan C Bovik. Video quality assessment by reduced reference spatio-temporal entropic differencing. *IEEE Transactions on Circuits and Systems for Video Technology*, 23(4):684–694, 2012. 7

[41] Pratul P. Srinivasan, Richard Tucker, Jonathan T. Barron, Ravi Ramamoorthi, Ren Ng, and Noah Snavely. Pushing the boundaries of view extrapolation with multiplane images. *CVPR*, 2019. 1, 2, 5, 6

[42] Richard Szeliski and Polina Golland. Stereo matching with transparency and matting. In *Sixth International Conference on Computer Vision (IEEE Cat. No. 98CH36271)*, pages 517–524. IEEE, 1998. 2

[43] Richard Szeliski and Polina Golland. Stereo matching with transparency and matting. *International Journal of Computer Vision*, 32(1):45–61, 1999. 4

[44] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul Srinivasan, Howard Zhou, Jonathan T. Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. Ibrnet: Learning multi-view image-based rendering. In *CVPR*, 2021. 7

[45] Zexiang Xu, Sai Bi, Kalyan Sunkavalli, Sunil Hadap, Hao Su, and Ravi Ramamoorthi. Deep view synthesis from sparse photometric images. *ACM Trans. Graph.*, 38(4), July 2019. 2

[46] Zexiang Xu, Kalyan Sunkavalli, Sunil Hadap, and Ravi Ramamoorthi. Deep image-based relighting from optimal sparse samples. *ACM Trans. Graph.*, 37(4), July 2018. 2

[47] Jae Shin Yoon, Kihwan Kim, Orazio Gallo, Hyun Soo Park, and Jan Kautz. Novel view synthesis of dynamic scenes with globally coherent depths from a monocular camera. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5336–5345, 2020. 2, 3

[48] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. In *SIGGRAPH*, 2018. 1, 2, 3, 4, 6

[49] C Lawrence Zitnick, Sing Bing Kang, Matthew Uyttendaele, Simon Winder, and Richard Szeliski. High-quality video view interpolation using a layered representation. *ACM transactions on graphics (TOG)*, 23(3):600–608, 2004. 2