Layered Multiplane Videos for Novel View Synthesis

Anonymous ICCV submission

Paper ID 6575

Abstract

The multiplane images (MPI) has shown great promise as a representation for efficient novel view synthesis. In this work, we present a new MPI-based approach for real-time novel view synthesis of monocular videos. We first formulate a new representation, referred to as layered MPI (LMPI), to reduce the number of parameters in MPI and make it suitable for videos. Then we propose a pipeline that generates sequence of temporally consistent LMPI using a single monocular video as input. The pipeline exploits the information from multiple frames, does not require any camera pose information, and can generate compelling layered multiplane video results. Experiments validate that our framework achieves better visual quality than several baselines and is capable of interactive novel view synthesis during video playback.

1. Introduction

Novel View Syntesis (NVS) addresses the problem of generating novel views of a scene from a given image(s) or video(s). It provides a compelling way of interacting with images or video recordings and thus has lots of exciting applications in content creation and rendering. Existing works have shown remarkable performance in generating novel views using images from multiple or even single image. However, there are few attempts to generate new views from monocular videos of dynamic scenes.

Existing NVS methods [46, 20] on monocular videos focus on videos that are carefully captured so that: 1. the camera has enough translation, and 2. the dynamic objects do not dominate the field of view. More specifically, their methods rely on structure from motion (SfM) to obtain geometry of the static part of the scene as a prior before inferring the dynamic structure. Therefore, they are not robust to examples where SfM cannot capture the shape accurately. Moreover, they are unable to generate new views in real-time, making it more restricted for some practical use scenarios.

To circumvent this limitation posed by the SfM results, we propose a different approach that extends the single image method to videos. Particularly, we focus on the MPI representation for its rendering efficiency. However, there are several challenges when trying to apply the single image methods to videos:

Fully Exploit Cross-frame Information: Single frame methods achieve plausible results by exploiting the spatial information inside one frame while the temporal information is ignored. How to aggregate cross-frame information remains an open problem, especially when the SfM fails.

Temporal Consistency: Single frame methods usually suffer from flickering artifacts if applied in a frame-by-frame manner. This can potentially be solved using post-processing methods [4, 16, 18]. In the task of NVS, however, both geometry and texture consistency need to be preserved and the heterogeneous representation makes it challenging to directly apply existing post-processing methods.

Rendering Efficiency: Several methods have achieved interactive NVS for single image [41, 14]. But to the best of our knowledge, there are no attempts for monocular videos.

In this work we attempt to tackle these challenges. We first introduce a novel representation, layered multiplane image, or LMPI for short, that can be rendered in real-time as an MPI but is more compact in terms of parameter size. We then propose a pipeline that utilizes the cross-frame information and generates temporally consistent LMPIs. We consider two properties of videos that can *always* be exploited: motion boundaries for predicting geometry, and disocclusions for background textures. Moreover, disocclusions are also key to producing temporal consistency in novel views, since the same background will potentially be seen in different timestamps.

The proposed pipeline formulates the motion boundary guidance as the motion field guiding the upsampling process of the LMPI. Disocclusions are aggregated as context for generating the background image in the representation. To summarize, our main contributions include:

- To our knowledge, the first framework that achieves real-time NVS of monocular videos without any camera pose information.

- A new representation that is more compact than MPI and achieves better visual quality.
- A new method to exploit motion boundaries for generating temporally consistent geometry.
- A new algorithm to aggregate background context from multiple frames.

2. Related work

Novel View Synthesis Interpolating or extrapolating views given multiple input views is a well-studied problem [6, 29, 11, 7, 49]. On the contrary, NVS on a single image is a highly ill-posed problem, since both structure and occluded texture need to be recovered from a single image. Several methods have been proposed to synthesize novel views from a single image [28, 35, 14, 41, 44]. These methods are difficult to generalize to videos due to the aforementioned challenges.

Most NVS methods usually first generate an intermediate representation like layered depth [34, 50] and neural radiance field [27, 25, 36, 3]. In particular, we focus on works that use multiplane image (MPI) as proxy. MPI has achieved great success in generating photo-realistic images [49, 10, 37, 26, 41] because of its ability on modeling non-Lambertian shading and soft edges. Moreover, the rendering process is efficient and differentiable, so the pipeline can be trained end-to-end. Some variants have been explored such as multi sphere image (MSI) [1], layered mesh [5], deep MPI [21] and NeX [45]. Most existing methods generate MPIs from multiple views. We instead focus on a single monocular video input.

Novel view synthesis for dynamic scenes Many works have explored the possibility of generating novel views in dynamic scenes. However most approaches require multi-view input in each timestamp [50, 23, 2, 38, 5, 1].

Recent works have taken a step forward for NVS for dynamic scene using monocular video. [46] manually mask out dynamic objects and use SfM to obtain an incomplete structure. This structure is then used for correcting the depth predicted from a single image. [20, 40] try to fit the dynamic scene using a neural radiance field by training a fully connected neural network during test time. However, these methods require camera poses and fail to produce any results when SfM does not work well, e.g. a scene with homogeneous textures, where the camera-motion is negligible, or when dynamic objects occupy too much image space. To best of our knowledge, there are no methods that focus on NVS in dynamic scenes without poses as additional input.

Structure from monocular video We also review methods that only predict depth from videos in a dynamic scene. [31] use motion segmentation and occluder-occludee relationships to infer relative depth. [19] compute an incomplete depth map using Plane-Plus-Parallax representation and use it as a prior to generate a complete one. [22] use probability volumes among different frames to refine the depth from a single view. Recent attempts achieve globally consistent results by applying test-time learning [24, 15]. These methods again need camera poses and thus are not applicable in our task.

3. Approach

3.1. System Overview

Given an input image sequence $\mathbb{V} = \{\mathbf{I}_t | t = 0, 1, ...\}$, our pipeline operates on a local time window $\{\mathbf{I}_k \in \mathcal{N}(t)\}$ and predicts a LMPI $\mathbf{R}_t$ for each frame. $\mathbf{R}_t$ consists of three components $\{\mathbf{P}_t, \mathbf{I}_t, \mathbf{B}_t\}$, where $\mathbf{P}_t$ is the geometry representation, namely parameter map, that defines a density function over the depth for each pixel; and $\mathbf{B}_t$ is the predicted background image. While rendering the novel view, we first convert $\mathbf{R}_t$ to MPI representation and then follow the standard MPI rendering pipeline [49].

Our pipeline can be partitioned into two modules. The first module estimates the parameter map $\mathbf{P}_t$ and the other predicts the background image $\mathbf{B}_t$. We elaborate our new representation $\mathbf{R}_t$ in Section 3.2 and the above two modules in Section 3.3 and 3.4. Then we describe the data for training in Section 3.5 and finally the losses in Section 3.6.

3.2. Layered Representation of Multiplane Image

The MPI represents the scene geometry using D fronto-parallel alpha planes in the frustum of a reference camera [49] with each plane arranged at fixed depths. Typically D varies from dozens to hundreds, which can easily become a bottleneck when processing videos. Another prevalent representation for NVS is layered depth image (LDI), which models the geometry as only two or more layers of depth map. However, the LDI cannot model soft edges and is inefficient to render for videos. We take the advantages of both representations by parameterizing the D alpha planes to layers and converting back to MPI during rendering. Thus we call our representation *layered* MPI.

As illustrated in Figure 2, for each pixel, the LDI models the density over disparity x (inverse depth) as several pulse functions, while MPI fits the geometry by a discrete density function. In contrast, our parameter map $\mathbf{P}_t$ defines a continuous density function using two sets of parameters $\{d_{fg}, t_{fg}\}$[1] and $\{d_{bg}, t_{bg}\}$ that represent the foreground and background layers, respectively. Formally:

$$\sigma(x) = \sigma_0 \sum_{n=\{fg,bg\}} \mathbf{1}(d_n - t_n < x < d_n), \qquad (1)$$

[1] For ease of notation, we omit the subscript for pixel index and time index and use a lower case letter per pixel parameter.

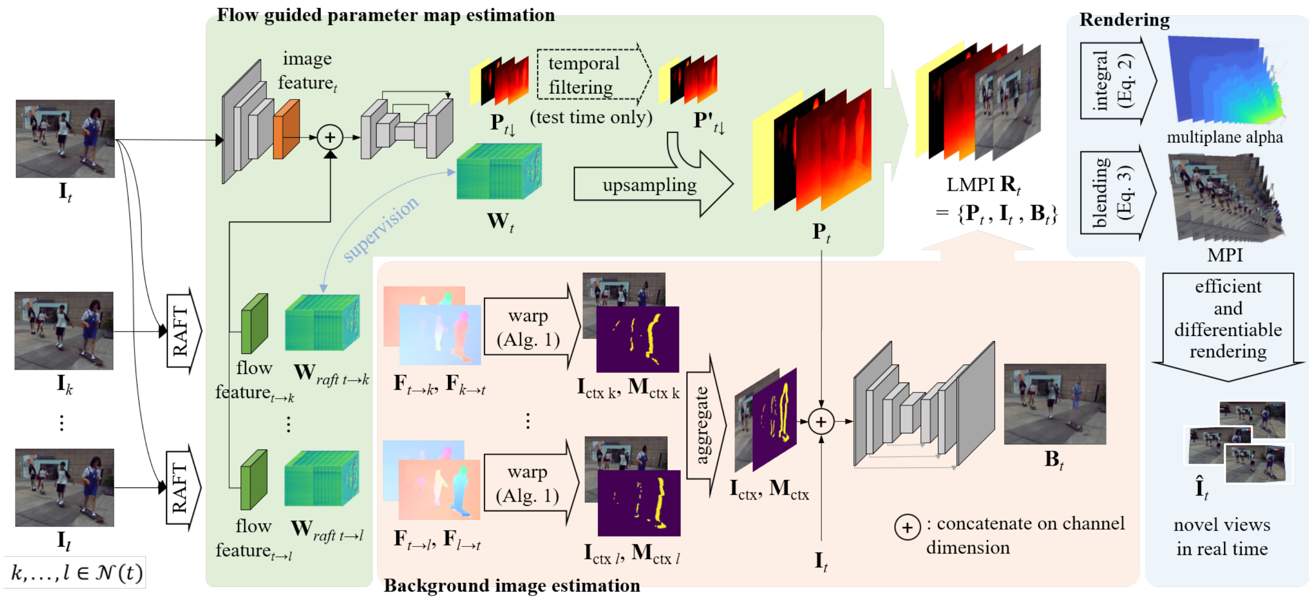


Figure 1: The proposed framework. For each frame $\mathbf{I}_t$ we operate on a local time window $\mathcal{N}(t)$ and generate LMPI $\mathbf{R}_t$ for rendering. The pipeline consists of two modules that generate parameter map $\mathbf{P}_t$ and background image $\mathbf{B}_t$ respectively. Parameter map estimation module (green) first predicts a coarse parameter map $\mathbf{P}_t\downarrow$ and an upsampling weight $\mathbf{W}_t$. $\mathbf{P}_t\downarrow$ is then upsampled to get final $\mathbf{P}_t$. Background estimation module (red) first use Algorithm 1 to aggregate context image $\mathbf{I}_{ctx}$ and context mask $\mathbf{M}_{ctx}$ from neighbor frames, then use U-Net to generate the final background image $\mathbf{B}_t$

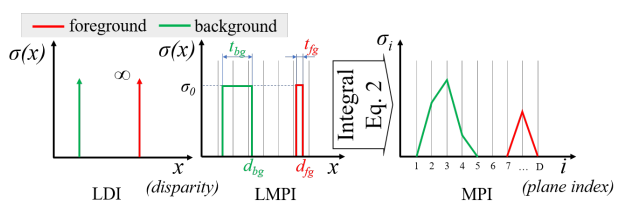


Figure 2: An illustration of defined density function $\sigma(x)$ in LDI, LMPI and MPI representation.

where $\mathbf{1}(\cdot)$ is an indicator function and σ_0 is a constant. By defining $\sigma(x)$, we assume that each layer is positioned at disparity d_n and has thickness t_n. We also assume each layer has constant density σ_0. In early experiments we found that optimizing over σ_0 leads to a half-transparent object even for solid materials, causing blur artifacts. We can additionally predict more than two layers, but we find that not necessary in practice since two layers are already expressive enough to fit in most structures that are inferred from single views.

During rendering, we first convert $\mathbf{P}$ to multi-plane alpha α_i following classical volume rendering [12]:

$$\alpha_i = 1 - \exp\left(-\int_{x_i}^{x_{i+1}} \sigma(x)dx\right), \tag{2}$$

where i indicates the plane index and x_i the disparity of i-th plane. The color of each plane c_i is a linear combination of $\mathbf{I}$ and $\mathbf{B}$:

$$c_i = w_i c_{fg} + (1 - w_i) c_{bg}, \tag{3}$$

where c_{fg} and c_{bg} are the RGB values from $\mathbf{I}$ and $\mathbf{B}$, respectively, and the blending weight w_i is determined by:

$$w_i = \begin{cases} 1 & x_i > (d_{fg} + d_{bg})/2 \\ 1 - \alpha_{fg} & \text{otherwise, and} \end{cases} \tag{4}$$

$$\alpha_{fg} = 1 - \exp\left(-\sigma_0 t_{fg}\right), \tag{5}$$

where α_{fg} is the transparency of the foreground layer. The blending weight is inspired by the observation that invisible regions should use the background image [41]. We also synthesize a pseudo disparity map for depth supervision by:

$$\hat{d} = \alpha_{fg} d_{fg} + (1 - \alpha_{fg}) d_{bg}. \tag{6}$$

After converting to the MPI representation, new views can be synthesized using the standard MPI rendering pipeline.

3.3. Flow Guided Parameter Map Estimation

We propose to predict the geometry representation $\mathbf{P}$ with the pipeline shown in the green box in Figure 1. As is discussed in the introduction, motion boundaries provide guidance for predicting depth and we formulate the guidance as the flow field guiding the up-sampling process of a

ICCV #6575

coarse parameter map. The pipeline predicts a coarse parameter map $\mathbf{P}_{\downarrow}$ and an up-sampling weight $\mathbf{W}$ at $1/8$ resolution by fusing the image feature and the flow feature predicted by RAFT [39]. $\mathbf{W}$ is supervised by the up-sampling weight $\mathbf{W}_{raft}$ that is used for upsampling the flow field, so the edges of the parameter map are encouraged to align with the flow edges.

One potential problem of fusing the flow features is that for static scenes, where the flow features implicitly encode epipolar geometry, the network may learn to infer structure from the flow even if we do not input any camera pose. This should not happen for a dynamic scene since the epipolar constraint no longer holds. So we remove the flow feature to break the flow-depth relationship when we are training the dataset with static scenes. In the inference stage, we apply occlusion-aware temporal filtering at coarse parameter map:

$$\mathbf{P}'_{t\downarrow} = \sum_{k\in\mathcal{N}(t)} \bar{\mathbf{O}}_{t\rightarrow k\downarrow} * \mathcal{W}(\mathbf{F}_{t\rightarrow k\downarrow}, \mathbf{P}_{k\downarrow}), \tag{7}$$

where $\mathbf{F}_{t\rightarrow k\downarrow}$ is the optical flow from frame t to k at $1/8$ resolution predicted also by RAFT, $\mathcal{W}(\mathbf{F}, \mathbf{C})$ is the backward warping function that bilinearly samples the content $\mathbf{C}$ with flow $\mathbf{F}$, and $\bar{\mathbf{O}}_{t\rightarrow k\downarrow}$ is a normalized soft occlusion mask:

$$\bar{\mathbf{O}}_{t\rightarrow k\downarrow} = g_k \mathbf{O}_{t\rightarrow k\downarrow} / \sum_{i\in\mathcal{N}(t)} g_i \mathbf{O}_{t\rightarrow i\downarrow}, \tag{8}$$

where $\cdot/\cdot$ is the pixel-wise division, g_i is the Gaussian kernel and $\mathbf{O}_{t\rightarrow k\downarrow}$ is a soft occlusion mask:

$$\mathbf{O}_{t\rightarrow k\downarrow} = \exp\left(-\alpha_0 |\mathbf{F}_{t\rightarrow k\downarrow} + \mathcal{W}(\mathbf{F}_{t\rightarrow k\downarrow}, \mathbf{F}_{k\rightarrow t\downarrow})|_1\right). \tag{9}$$

$|\cdot|_1$ is the pixel-wise L^1 norm and α_0 is a constant which we set to 0.2. The fine parameter map $\mathbf{P}$ is then generated by up-sampling $\mathbf{P}'_{t\downarrow}$ using $\mathbf{W}$. The up-sampling follows the same process as [39].

3.4. Background Image estimation

Next, we describe the pipeline to generate the background image (refer to orange box in Figure 1). We first aggregate context information from neighboring frames. Specifically, for each frame $\mathbf{I}_t$ and its temporal neighbor $\mathbf{I}_k$, we try to grab all the disoccluded pixels from $\mathbf{I}_k$ and align in $\mathbf{I}_t$. This is challenging because the disoccluded pixels in $\mathbf{I}_t$ are covered by foreground and optical flow is valid only in the visible region. One solution is to forward warp (splat) the disoccluded pixels using $\mathbf{F}_{k\rightarrow t}$, but splatting generally suffers from holes and blurriness. Therefore, we propose an algorithm to generate the background flow $\mathbf{F}_{bg}$ so that all contexts can be aligned using backward warp. As illustrated in Algorithm 1, we generate an initial background flow by splatting the $-\mathbf{F}_{k\rightarrow t}$ using $\mathbf{F}_{k\rightarrow t}$ itself. The splatted flow is

Algorithm 1: Generate context image and mask of $\mathbf{I_k}$ with respect to time t

Input: frame k: $\mathbf{I}_k$, bidirectional flow between frame t and k: $\mathbf{F}_{t\rightarrow k}$, $\mathbf{F}_{k\rightarrow t}$
Result: $\mathbf{I}_{ctx_k}$, $\mathbf{M}_{ctx_k}$
Note that **One** is a map filled with 1.
$\mathbf{Occ}_k \leftarrow 1 - \mathcal{S}\,(\mathbf{F}_{t\rightarrow k}, \mathbf{One}, \mathbf{One})$;
$\mathbf{F}^0_{bg} \leftarrow \mathcal{S}\,(\mathbf{F}_{k\rightarrow t}, -\mathbf{F}_{k\rightarrow t}, \mathbf{Occ}_k)$;
for $i \leftarrow 0$ **to** 2 **do**
  $\mathbf{F}^{i+1}_{bg} \leftarrow \mathcal{W}\,(\mathbf{F}^i_{bg}, \mathbf{F}_{k\rightarrow t})$;
end
$\mathbf{M}_{ctx_k} \leftarrow \mathcal{S}\,(\mathbf{F}_{k\rightarrow t}, \mathbf{Occ}_k, \mathbf{Occ}_k)$;
$\mathbf{I}_{ctx_k} \leftarrow \mathcal{W}\,(\mathbf{F}^3_{bg}, \mathbf{I}_k)$;
def $\mathcal{S}\,(\mathbf{F}, \mathbf{C}, \mathbf{W})$ **:**
  $\mathbf{C}' \leftarrow \mathbf{0}$;
  forward splat content $\mathbf{C}$ and weight $\mathbf{W}$ using flow $\mathbf{F}$ to 4 nearest neighbors onto $\mathbf{C}'$. For each position in $\mathbf{C}'$ we cache list of splatted content $\mathbb{C} = \{c_i | i \in [1, k]\}$ and weight $\mathbb{W} = \{w_i | i \in [1, k]\}$, as well as the bilinear splatting weight $\mathbb{M} = \{m_i | i \in [1, k]\}$;
  for *each pixel c' in C'* **do**
    $c' = \sum_{i=1}^{k} \frac{c_i w_i m_i}{w_i m_i}$
  end
  return $\mathbf{C}'$
end

weighted by occlusion mask in $\mathbf{I}_k$ so the background flow will not be fused by the foreground flow. To eliminate small errors we iteratively search the $\mathbf{F}_{bg}$ to meet the bidirectional consistency with $\mathbf{F}_{k\rightarrow t}$.

For each frame t, Algorithm 1 is applied to several neighbors to obtain a collection of context images and masks $\{\mathbf{I}_{ctx\,k}, \mathbf{M}_{ctx\,k} | k \in \mathcal{N}(t)\}$. We aggregate them to a unified context image by iteratively overwriting the pixels in $\mathbf{I}_t$ with $\mathbf{I}_{ctx\,k}$ in the pixel positions where $\mathbf{M}_{ctx\,k}$ is larger than a threshold $\epsilon = 0.5$. The order is not of concern because usually the context masks have few overlaps. $\mathbf{M}_{ctx}$ is obtained in a similar manner. The context image and mask are concatenated together with the current image $\mathbf{I}$ and parameter map $\mathbf{P}$ to predict the final background image $\mathbf{B}$. The network learns to inpaint the remaining occluded region where there is no context from neighbor frames.

3.5. Data

Acquiring proper data for training this pipeline is challenging, requiring videos with ground truth geometry as well as ground truth novel views for every timestamp. Next, we describe the two types of sources that we used.

Cameras exploring static scene: Since the scene is static, every timestamp is a ground truth novel view for the current frame. Considering the diversity of data, we use a combination of RealEstate10K (RE10K) [49] and MannequinChallenge (MC) [19] for training. We reconstruct a sparse point cloud model for each scene using COLMAP [32, 33] and apply a customized pipeline to filter out bad data.

Stereo Cameras exploring dynamic scene: One source that perfectly suits our need is stereo video, since it provides semi-dense disparity maps as well as ground-truth novel views. We use WSVD [43] for training and StereoBlur (SB) [48] for evaluation. We also re-process the raw data of WSVD using a customized pipeline since they only provide video links. More detailed descriptions of the data processing pipeline and dataset statistics can be found in the supplementary material.

3.6. Losses

Scale invariant depth supervision: Structures reconstructed from a single view usually suffer from scale ambiguity, and the common practice is to correct the scale [8, 42, 41] before comparing with ground truth. We follow the same equation of $\mathcal{L}_{depth}$ as in [41] to supervise the pseudo disparity map we predict in Equation 6.

Reconstruction loss: Given the LMPI representation, we reconstruct the novel view using the method described in Section 3.1. We then penalize the reconstruction error by per-pixel L^1 loss between ground truth novel view and rendered novel view:

$$\mathcal{L}_{reconstruction} = \|\hat{\mathbf{I}}_{rendered} - \mathbf{I}_{groundtruth}\|_1, \quad (10)$$

We denote the $\|\cdot\|_1$ as the L^1 norm over all pixel positions and channels.

Parameter prior: Training two-layer structure from a single view is not trivial since there is almost no supervision for the background layers. We find it necessary to constraint the behavior of the background layer using a prior loss. One observation is that when there is a sharp edge in the disparity map, the background layer should remain smooth and be properly occluded by the foreground. Thus we formulate the prior loss as:

$$\mathcal{L}_{prior} = |\mathcal{E}(\hat{\mathbf{D}}_\downarrow) - \hat{\mathbf{D}}_{bg\downarrow}|_1 * \mathbf{M_{in}} + |\hat{\mathbf{D}}_\downarrow - \hat{\mathbf{D}}_{bg\downarrow}|_1 * \mathbf{M_{out}}, \quad (11)$$

where $\mathbf{M_{in}}$ and $\mathbf{M_{out}}$ are two masks that softly indicates the two sides of a disparity edge:

$$\begin{aligned} \mathbf{M_{in}} &= |\mathcal{E}(\hat{\mathbf{D}}_\downarrow) - \hat{\mathbf{D}}_\downarrow|_1, \\ \mathbf{M_{out}} &= |\mathcal{D}(\hat{\mathbf{D}}_\downarrow) - \hat{\mathbf{D}}_\downarrow|_1, \end{aligned} \quad (12)$$

where $\mathcal{E}$ and $\mathcal{D}$ is the morphological erosion and dilation. Note that we apply the prior loss only in at the coarse resolution.

Hybrid smoothness: Unlike most methods that apply edge-aware smoothness on the disparity map, we argue that two layers should have different priors. The foreground should align its edges to input image while the background layer should remain smooth. We first compute an edge mask $\mathbf{E}$:

$$\mathbf{E} = max\left(1 - \frac{\mathcal{G}(\mathbf{I})}{e_{min}\max(\mathcal{G}(\mathbf{I}))}, 0\right), \quad (13)$$

where $\mathcal{G}(\mathbf{I})$ is the per-pixel L^1 norm of the gradient of $\mathbf{I}$. The smooth loss is then a combination of edge-aware smoothness of foreground disparity $\hat{\mathbf{D}}_{fg}$ and first order smoothness of background disparity $\hat{\mathbf{D}}_{bg}$:

$$\mathcal{L}_{smooth} = \|\mathcal{G}(\hat{\mathbf{D}}_{fg}) * \mathbf{E} + \lambda_g \mathcal{G}(\hat{\mathbf{D}}_{bg})\|_1. \quad (14)$$

We empirically set $\lambda_g = 0.2$.

Upsampling supervision: As is mentioned in Section 3.3, we supervise the upsampling weight $\mathbf{W}$ by:

$$\mathcal{L}_{upsampling} = \min(\mathcal{G}(\mathbf{F}_\downarrow), 1)\|\mathbf{W} - \mathbf{W}_{raft}\|_1, \quad (15)$$

where $\mathbf{F}_\downarrow$ is the coarse flow that has the same resolution as $\mathbf{P}_\downarrow$. We use the soft mask $\min(\mathcal{G}(\mathbf{F}_\downarrow), 1)$ to decay the weight where the flow has a small gradient since there is no motion boundary in those regions.

Background supervision: We encourage the predicted background $\hat{\mathbf{B}}$ to copy the context from the motion disocclusions by background supervision:

$$\mathcal{L}_{background} = \||\hat{\mathbf{B}} - \mathbf{I}_{ctx}|_1 * \mathbf{M}_{ctx}\|_1. \quad (16)$$

Final loss: The final loss is a weighted sum of all the losses:

$$\begin{aligned} \mathcal{L} =& \lambda_d \mathcal{L}_{depth} + \lambda_r \mathcal{L}_{reconstruct} + \lambda_p \mathcal{L}_{prior} \\ &+ \lambda_s \mathcal{L}_{smooth} + \lambda_u \mathcal{L}_{upsampling} + \lambda_b \mathcal{L}_{background}. \end{aligned}$$

In the experiments we empirically set the loss weight as follows: $\lambda_r = \lambda_u = \lambda_b = 1.0$, $\lambda_p = 0.2$, $\lambda_s = 0.5$, $\lambda_d = 0.2$ for sparse depth supervision and 1.0 for semi-dense depth supervision. Semi-dense depth supervision should have stronger effect since it provides more guidance.

4. Experiments

Due to space limitations, we put the experiment settings and implementation details in the supplementary material. In this section we first describe the metrics and methods that we choose for evaluation in Section 4.1 and 4.2. Then we show the quantitative and qualitative results of NVS and depth in Section 4.3 and 4.4. Finally, we perform ablations to show the need of several components of our pipeline in Section 4.5.

4.1. Metrics

We evaluate our approach on the StereoBlur dataset, which contains calibrated stereo videos as well as corresponding semi-dense depth maps. For each sequence, we extract consecutive 20 frames for evaluation. For each frame, we use the left view as input and generate the right view using scale-invariant rendering [41]. We report *SSIM*, *PSNR* and *LPIPS* [47] for the generated images. We additionally report the average and median optical flow magnitude between the ground truth novel view and generated view, denoted as *FMean* [46] and *FMid*, respectively. We evaluate the depth quality on the same frames using Standard Metrics described in iBims-1 [13].

Since temporal consistency is an important property for video applications, we additionally formulate several metrics for evaluating the temporal consistency:

First, we use *FEPE* for evaluating the temporal consistency of the novel view. It measures the agreement between the ground truth flow and the rendered novel view flow:

$$FEPE = \|\mathbf{F}_{t\rightarrow t+1} - \hat{\mathbf{F}}_{t\rightarrow t+1}\|_1, \tag{17}$$

where $\hat{\mathbf{F}}_{t\rightarrow t+1}$ is the optical flow between $\hat{\mathbf{I}}_t$ and $\hat{\mathbf{I}}_{t+1}$.

We compute the warping error of the estimated disparity map, which measures the first order derivative of the disparity map:

$$DT_1 = \||\mathcal{W}(\mathbf{F}_{t\rightarrow t+1}, \hat{\mathbf{D}}_{t+1}) - \hat{\mathbf{D}}_t| * \mathbf{N}_{t\rightarrow t+1}\|_1, \tag{18}$$

where $\mathbf{N}_{t\rightarrow t+1}$ is the occlusion mask that mask out pixels with bidirectional flow error larger than 1 pixel.

We also compute the second order derivative of the disparity map, which is given by:

$$\begin{aligned} DT_2 =& \||\mathcal{W}(\mathbf{F}_{t\rightarrow t+1}, \hat{\mathbf{D}}_{t+1}) + \mathcal{W}(\mathbf{F}_{t\rightarrow t-1}, \hat{\mathbf{D}}_{t-1}) \\ &- 2\hat{\mathbf{D}}_t|_1 * \mathbf{N}_{t\rightarrow t+1} * \mathbf{N}_{t\rightarrow t-1}\|_1, \end{aligned} \tag{19}$$

4.2. Baselines

Since there are no previous works that focus on the exact same task, we carefully select several baselines for comparison.

The first baseline is the original method of [41], referred to as **svMPI**. For a fair comparison, we retrain the model with our dataset using the losses and settings described in the paper. The second baseline is **svMPI+svreg**, which is the same as **svMPI**, except that we train the model using an additional temporal consistency loss described in [9]. For **svMPI+filter**, we apply the same occlusion-aware temporal filtering as in our method, except that the filtering is operated on multiplane alpha at the original resolution. Furthermore, we try to use the Learned Blind Temporal Consistency (LBTC) [17] to smooth **svMPI**, denoted as **svMPI+lbtc**, which does not require any dense correspondence. We include the details of training the LBTC module on the supplementary material. Finally, we compare to the method that use LDI representation [35], denoted as **svLDI**.

To further evaluate the depth quality, we compare with methods that predict only the depth map from a single image (**MiDaS** [30]) or video (**MC** [19] and **WSVD** [43]).

method	SSIM↑	PSNR↑	LPIPS↓	FMean↓	FMid↓	FEPE↓
svMPI	0.79	20.98	0.21	5.30	3.77	1.70
svMPI +svreg	0.79	21.16	0.34	5.84	4.35	1.37
svMPI +filter	**0.80**	21.13	0.22	5.13	3.62	1.08
svMPI +lbtc	**0.80**	21.15	0.20	5.05	3.43	1.36
svLDI	0.76	19.84	0.16	5.31	3.68	2.15
Ours	**0.80**	**21.32**	**0.15**	**4.60**	**3.06**	**1.06**

Table 1: Evaluation of novel view synthesis and consistency. ↑ means higher is better and ↓ lower better. We highlight the metrics that perform best in **bold**. Our method outperforms other baselines in terms of the perceptional similarity and the flow magnitude. See Section 4.3

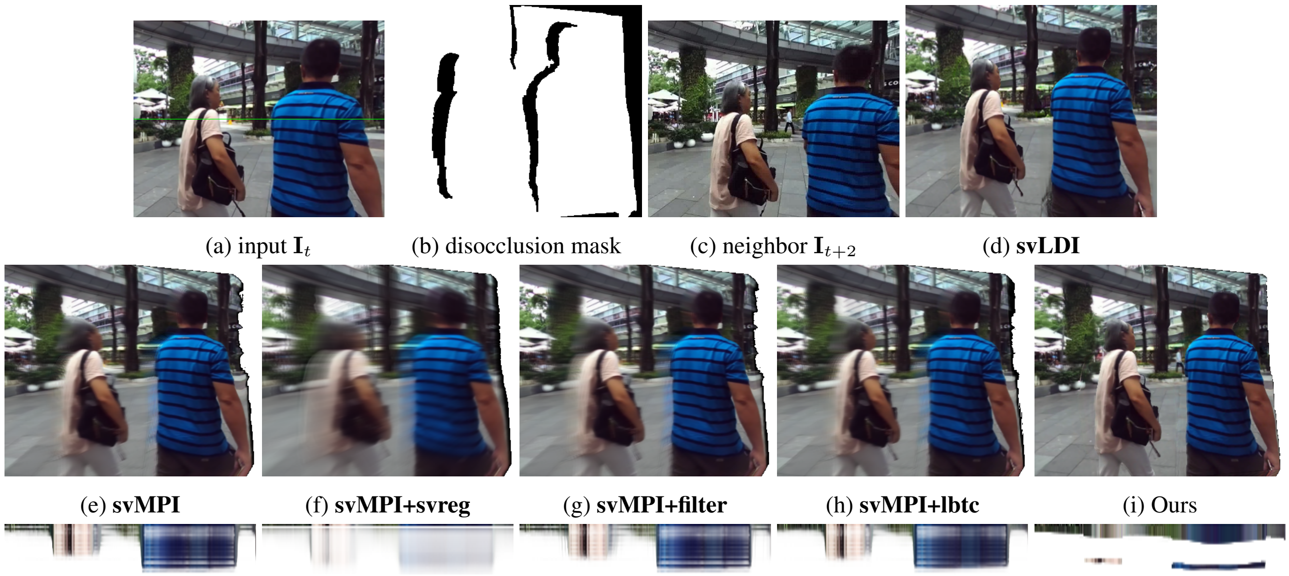
4.3. Evaluation of Novel View Synthesis

The quantitative results of NVS are shown in Table 1. It can be seen that our methods does not have a big transcendence over SSIM and PSNR, which we find due to the phenomenon that SSIM and PSNR favor blurry results than misaligned images (refer to the supplementary material for an example). However, there is a clear improvement on the perceptual similarity and the flow magnitude, which we find are more consistent with human perception of visual quality. Applying filtering or LBTC post processing slightly improves the NVS quality, while adding single frame regularization significantly decreases the performance. **svLDI** achieves similar LPIPS as our method. However, it is not good at producing temporally consistent results.

We demonstrate some NVS results as well as MPI visualizations on Figure 3 (more examples can be found in supplementary material). We can see that **svMPI** tries to fill the disoccluded regions using repeated textures, causing obvious blurry artifacts, which become even more serious when we try to apply various temporal consistency methods. We visualize the MPI by slicing through the green line in Figure 3a along the depth (plane index) direction. We can see that all the single view MPI based methods exibit repeated content in the layers behind the foreground, while ours produce an obvious two-layer structure, each with a different texture. This significantly reduces the blurriness in the results. **svLDI** shows plausible results for the novel views, however the generated texture in the disoccluded regions are not temporally consistent. Specifically, notice the small person appearing in the background of frame $\mathbf{I}_{t+2}$. Unlike **svLDI**, our method successfully generates the person by aggregating textures from neighbor frames.

4.4. Evaluations on Depth

We show numerical results of depth estimation in Table 2. Additionally, we plot the **log10 - DT₁** graph of all the methods in Figure 5. **MiDaS** shows the best performance

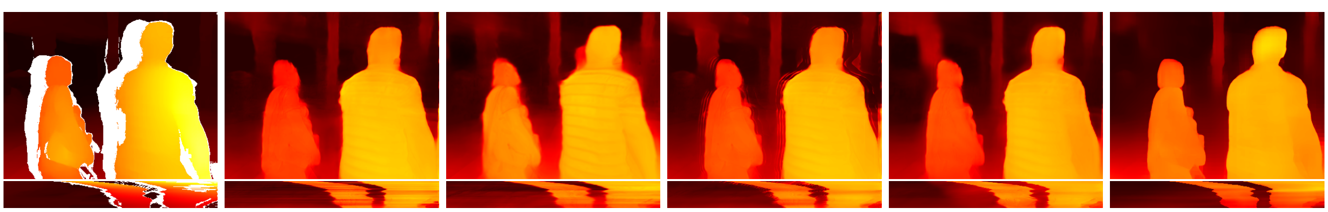


(a) input $\mathbf{I}_t$ (b) disocclusion mask (c) neighbor $\mathbf{I}_{t+2}$ (d) **svLDI**

(e) **svMPI** (f) **svMPI+svreg** (g) **svMPI+filter** (h) **svMPI+lbtc** (i) Ours

(j) **svMPI** (k) **svMPI+svreg** (l) **svMPI+filter** (m) **svMPI+lbtc** (n) Ours

Figure 3: NVS results and MPI visualization of baselines and ours. Figures 2a to 2c: the input frame, disocclusion mask in novel views and neighbor frame. Figures 2d to 2i: the novel views synthesized by corresponding methods. Figures 2j to 2n: MPI visualization. The visualization is done by slicing the MPI along the green line in Figure 3a, and the vertical axis indicates plane index. From the results we can see that **svMPI** based methods generate blurry results, while **svLDI** fails to generate the background that is consistent with frame $\mathbf{I}_{t+2}$ (notice the background in disoccluded region).



(a) ground truth (b) **svMPI** (c) **svMPI+svreg** (d) **svMPI+filter** (e) **svMPI+lbtc** (f) Ours

Figure 4: First row: the predicted disparity maps. Second row: visualizations of temporal consistency. The vertical axis indicates timestamps while the horizontal axis indicates the spatial location. Our method produce disparity maps with more temporal consistency and less spatial artifacts

in terms of depth accuracy. It is trained on 10 datasets[30]. Our method achieves slightly worse accuracy but has the best trade-off between accuracy and temporal consistency, as shown on Figure 5.

We visualize several predicted depth maps in Figure 4. We find that although **svMPI+filter** achieves best temporal consistency, it produces unpleasant artifacts along depth boundaries due to the misalignment between depth edges and flow edges. **svreg** and **lbtc** seem improve the temporal consistency, but both produce some spatial artifacts. In contrast, our method achieves both sharp edges and temporal consistency, thus being the closest to the ground truth.

4.5. Ablations

We first ablate on the module that generates the parameter maps. We change several settings based on the **full** model described in Section 3. We first test the necessity of the flow feature by not feeding it during training and testing, denoted as **noflow**. For **noupsu**, we train the pipeline with no upsampling supervision, i.e. $\lambda_u = 0$. Finally, as illustrated in Section 3.3, we remove the flow feature during training for static scenes. We ablate this operation by treating the static scene as a dynamic one during training. This is denoted as **nodrop**.

The numerical results are presented in Table 2 and one example is shown in Figure 6. We find that the flow feature

Methods	Rel↓	log10↓	$\sigma1$↑	$\sigma2$↑	$\sigma3$↑	DT_1↓	DT_2↓
svMPI	0.463	0.159	0.436	0.695	0.833	1.024	1.465
svMPI+svreg	0.467	0.162	0.415	0.676	0.831	0.663	0.866
svMPI+filter	0.456	0.154	0.445	0.701	0.841	**0.313**	**0.188**
svMPI+lbtc	0.467	0.157	0.435	0.700	0.845	0.453	0.496
WSVD	0.423	0.149	0.457	0.729	0.865	0.910	1.109
MC	0.580	0.175	0.397	0.673	0.827	0.726	1.006
MiDaS	**0.277**	**0.142**	**0.594**	**0.826**	**0.909**	0.819	1.100
Ours **full**	0.366	**0.142**	0.473	0.733	0.872	0.362	0.334
noflow	0.452	0.159	0.417	0.683	0.839	0.320	0.303
noupsu	0.409	0.146	0.460	0.736	0.873	0.402	0.388
nodrop	0.527	0.168	0.419	0.681	0.824	0.462	0.384

Table 2: Evaluation of depth accuracy and consistency. We highlight the metrics that perform best in **bold** and second-best in underline. MiDaS shows best regarding depth accuracy, while Ours demonstrates slightly worse accuracy but far more consistency.

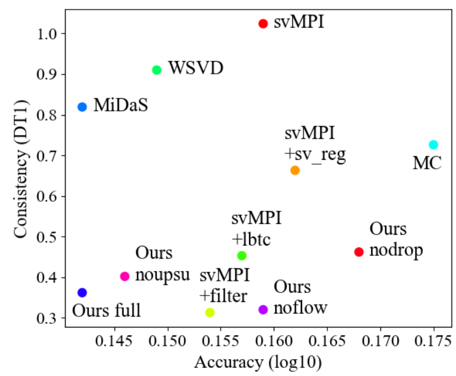


Figure 5: Accuracy-Consistency graph. This graph indicates that our model achieves best trade-off between depth accuracy and temporal consistency

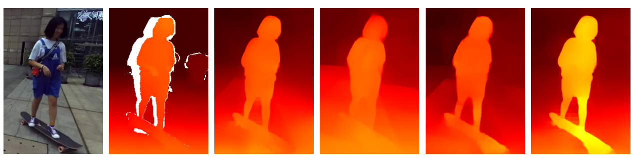


Figure 6: Ablations of depth quality. From left to right: input image, ground truth depth map, our full model (**full**), model without flow feature (**noflow**), model without upsampling supervision (**noupsu**) and model that do not drop out flow features (**nodrop**). See Section 4.5

Figure 7: Ablations of background supervision. From left to right: context image $\mathbf{I}_{ctx}$, context mask $\mathbf{M}_{ctx}$, $\mathbf{B}$ from our full model, $\mathbf{B}$ from our model without $\mathcal{L}_{background}$

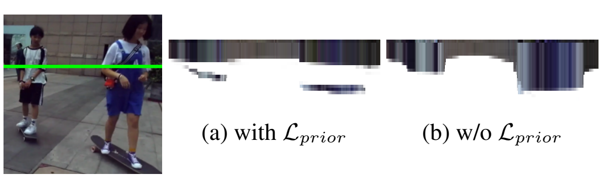


Figure 8: Ablations of prior loss $\mathcal{L}_{prior}$. We visualize the MPI along green line in the left input image. The pipeline fails to predict two-layer structure without $\mathcal{L}_{prior}$.

significantly helps generating accurate, sharp depth maps, while upsampling supervision results in a small ones. Interestingly, if we use the flow feature in the static scene as in **nodrop**, the predicted depth map, though contain sharp edges, attempts to infer incorrect geometry from the motion implicitly encoded in the flow feature.

Next, we explore the necessity of background supervision. As shown in Figure 7, without $\mathcal{L}_{background}$, the background generation is trained in a purely unsupervised manner. The generated background loses high-frequency information. Thus background supervision is necessary for generating a finely detailed novel view.

Finally, we show the results by training with and without $\mathcal{L}_{prior}$ in Figure 8. The pipeline fails to predict two layer structure and exhibits similar pattern as **svMPI** in Figure 3j. Specifically, the pipeline without $\mathcal{L}_{prior}$ predicts parameter map with the thickness of the foreground close to 0, i.e., the foreground are fully transparent.

5. Conclusions and Limitations

In this work we propose an integral framework for novel view synthesis using only monocular video as input. In the process, we propose a new representation, LMPI, that greatly reducing the redundancy of MPI, and a pipeline that effectively generates temporally consistent LMPIs. Results show that our method achieves the best visual quality and the best balance between depth accuracy and temporal consistency compared to existing methods. The framework still has some limitations which are left for future work, for examples, the inpainted textures of regions that are not visible in any of the video frames, such as the background behind a static object, do not have very high quality. This can be solved by manually generating synthetic data and using a more advanced loss, such as a GAN loss.

References

[1] Benjamin Attal, Selena Ling, Aaron Gokaslan, Christian Richardt, and James Tompkin. Matryodshka: Real-time 6dof video view synthesis using multi-sphere images. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 441–459, Cham, 2020. Springer International Publishing. 2

[2] A. Bansal, M. Vo, Y. Sheikh, D. Ramanan, and S. Narasimhan. 4d visualization of dynamic events from unconstrained multi-view videos. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5365–5374, 2020. 2

[3] Mojtaba Bemana, Karol Myszkowski, Hans-Peter Seidel, and Tobias Ritschel. X-fields: Implicit neural view-, light- and time-image interpolation. *ACM Trans. Graph.*, 39(6), Nov. 2020. 2

[4] Nicolas Bonneel, James Tompkin, Kalyan Sunkavalli, Deqing Sun, Sylvain Paris, and Hanspeter Pfister. Blind video temporal consistency. *ACM Trans. Graph.*, 34(6), Oct. 2015. 1

[5] 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. 2

[6] Gaurav Chaurasia, Sylvain Duchene, Olga Sorkine-Hornung, and George Drettakis. Depth synthesis and local warps for plausible image-based navigation. *ACM Trans. Graph.*, 32(3), July 2013. 2

[7] Inchang Choi, Orazio Gallo, Alejandro Troccoli, Min H Kim, and Jan Kautz. Extreme view synthesis. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7781–7790, 2019. 2

[8] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. 5

[9] Gabriel Eilertsen, Rafal K. Mantiuk, and Jonas Unger. Single-frame regularization for temporally stable cnns. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 6

[10] J. Flynn, M. Broxton, P. Debevec, M. DuVall, G. Fyffe, R. Overbeck, N. Snavely, and R. Tucker. Deepview: View synthesis with learned gradient descent. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2362–2371, 2019. 2

[11] Peter Hedman, Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel Brostow. Deep blending for free-viewpoint image-based rendering. *ACM Trans. Graph.*, 37(6), Dec. 2018. 2

[12] James T. Kajiya and Brian P Von Herzen. Ray tracing volume densities. *SIGGRAPH Comput. Graph.*, 18(3):165–174, Jan. 1984. 3

[13] Tobias Koch, Lukas Liebel, Friedrich Fraundorfer, and Marco Körner. Evaluation of cnn-based single-image depth estimation methods. In Laura Leal-Taixé and Stefan Roth, editors, *European Conference on Computer Vision Workshop (ECCV-WS)*, pages 331–348. Springer International Publishing, 2018. 6

[14] Johannes Kopf, Suhib Alsisan, Francis Ge, Yangming Chong, Kevin Matzen, Ocean Quigley, Josh Patterson, Jossie Tirado, Shu Wu, and Michael F Cohen. Practical 3d photography. In *Proceedings of CVPR Workshops*, 2019. 1, 2

[15] Johannes Kopf, Xuejian Rong, and Jia-Bin Huang. Robust consistent video depth estimation. *arXiv preprint arXiv:2012.05901*, 2020. 2

[16] Wei-Sheng Lai, Jia-Bin Huang, Oliver Wang, Eli Shechtman, Ersin Yumer, and Ming-Hsuan Yang. Learning blind video temporal consistency. In *European Conference on Computer Vision*, 2018. 1

[17] Wei-Sheng Lai, Jia-Bin Huang, Oliver Wang, Eli Shechtman, Ersin Yumer, and Ming-Hsuan Yang. Learning blind video temporal consistency. In *European Conference on Computer Vision*, 2018. 6

[18] Chenyang Lei, Yazhou Xing, and Qifeng Chen. Blind video temporal consistency via deep video prior. In *Advances in Neural Information Processing Systems*, 2020. 1

[19] 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)*, 2019. 2, 5, 6

[20] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. *arXiv preprint arXiv:2011.13084*, 2020. 1, 2

[21] Zhengqi Li, Wenqi Xian, Abe Davis, and Noah Snavely. Crowdsampling the plenoptic function. In *Proc. European Conference on Computer Vision (ECCV)*, 2020. 2

[22] Chao Liu, Jinwei Gu, Kihwan Kim, Srinivasa G Narasimhan, and Jan Kautz. Neural rgb (r) d sensing: Depth and uncertainty from a video camera. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10986–10995, 2019. 2

[23] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *ACM Trans. Graph.*, 38(4), July 2019. 2

[24] Xuan Luo, Jia-Bin Huang, Richard Szeliski, Kevin Matzen, and Johannes Kopf. Consistent video depth estimation. *ACM Trans. Graph.*, 39(4), July 2020. 2

[25] Ricardo Martin-Brualla, Noha Radwan, Mehdi S. M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, and Daniel Duckworth. NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections. In *arXiv*, 2020. 2

[26] 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. 2

[27] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf:

Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 2

[28] Simon Niklaus, Long Mai, Jimei Yang, and Feng Liu. 3d ken burns effect from a single image. *ACM Trans. Graph.*, 38(6), Nov. 2019. 2

[29] Eric Penner and Li Zhang. Soft 3d reconstruction for view synthesis. *ACM Trans. Graph.*, 36(6), Nov. 2017. 2

[30] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2020. 6, 7

[31] R. Ranftl, V. Vineet, Q. Chen, and V. Koltun. Dense monocular depth estimation in complex dynamic scenes. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4058–4066, 2016. 2

[32] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 5

[33] 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. 5

[34] 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*, SIGGRAPH '98, page 231–242, New York, NY, USA, 1998. Association for Computing Machinery. 2

[35] 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. 2, 6

[36] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. In *Advances in Neural Information Processing Systems*, 2019. 2

[37] P. P. Srinivasan, R. Tucker, J. T. Barron, R. Ramamoorthi, R. Ng, and N. Snavely. Pushing the boundaries of view extrapolation with multiplane images. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 175–184, 2019. 2

[38] Timo Stich, Christian Linz, Georgia Albuquerque, and Marcus Magnor. View and time interpolation in image space. *Computer Graphics Forum*, 27(7):1781–1787, 2008. 2

[39] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *European Conference on Computer Vision*, pages 402–419. Springer, 2020. 4

[40] Edgar Tretschk, Ayush Tewari, Vladislav Golyanik, Michael Zollhöfer, Christoph Lassner, and Christian Theobalt. Non-rigid neural radiance fields: Reconstruction and novel view synthesis of a dynamic scene from monocular video, 2020. 2

[41] Richard Tucker and Noah Snavely. Single-view view synthesis with multiplane images. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 1, 2, 3, 5, 6

[42] B. Ummenhofer, H. Zhou, J. Uhrig, N. Mayer, E. Ilg, A. Dosovitskiy, and T. Brox. Demon: Depth and motion network for learning monocular stereo. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 5

[43] Chaoyang Wang, Simon Lucey, Federico Perazzi, and Oliver Wang. Web stereo video supervision for depth prediction from dynamic scenes, 2019. 5, 6

[44] Olivia Wiles, Georgia Gkioxari, Richard Szeliski, and Justin Johnson. Synsin: End-to-end view synthesis from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2

[45] Suttisak Wizadwongsa, Pakkapon Phongthawee, Jiraphon Yenphraphai, and Supasorn Suwajanakorn. Nex: Real-time view synthesis with neural basis expansion. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[46] Jae Shin Yoon, Kihwan Kim, Orazio Gallo, Hyun Soo Park, and Jan Kautz. Novel view synthesis of dynamic scenes with globally coherent depths from a monocular camera. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5336–5345, 2020. 1, 2, 6

[47] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 6

[48] Shangchen Zhou, Jiawei Zhang, Wangmeng Zuo, Haozhe Xie, Jinshan Pan, and Jimmy Ren. Davanet: Stereo deblurring with view aggregation. In *CVPR*, 2019. 5

[49] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. *ACM Trans. Graph.*, 37(4), July 2018. 2, 5

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