

# Efficient View Synthesis and 3D-based Multi-Frame Denoising with Multiplane Feature Representations

Thomas Tanay    Aleš Leonardis    Matteo Maggioni  
Huawei Noah’s Ark Lab

{thomas.tanay, ales.leonardis, matteo.maggioni}@huawei.com

## Abstract

While current multi-frame restoration methods combine information from multiple input images using 2D alignment techniques, recent advances in novel view synthesis are paving the way for a new paradigm relying on volumetric scene representations. In this work, we introduce the first 3D-based multi-frame denoising method that significantly outperforms its 2D-based counterparts with lower computational requirements. Our method extends the multiplane image (MPI) framework for novel view synthesis by introducing a learnable encoder-renderer pair manipulating multiplane representations in feature space. The encoder fuses information across views and operates in a depth-wise manner while the renderer fuses information across depths and operates in a view-wise manner. The two modules are trained end-to-end and learn to separate depths in an unsupervised way, giving rise to Multiplane Feature (MPF) representations. Experiments on the Spaces and Real Forward-Facing datasets as well as on raw burst data validate our approach for view synthesis, multi-frame denoising, and view synthesis under noisy conditions.

## 1. Introduction

Multi-frame denoising is a classical problem of computer vision where a noise process affecting a set of images must be inverted. The main challenge is to extract consistent information across images effectively and the current state of the art relies on optical flow-based 2D alignment [3, 7, 45]. Novel view synthesis, on the other hand, is a classical problem of computer graphics where a scene is viewed from one or more camera positions and the task is to predict novel views from target camera positions. This problem requires to reason about the 3D structure of the scene and is typically solved using some form of volumetric representation [28, 32, 55]. Although the two problems are traditionally considered distinct, some novel view synthesis approaches have recently been observed to handle noisy

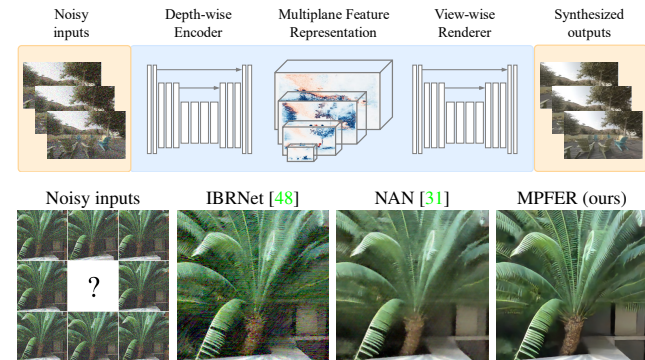


Figure 1. *Top*: Our Multiplane Features Encoder-Renderer (MPFER) reimagines the MPI pipeline by moving the multiplane representation to feature space. *Bottom*: MPFER significantly outperforms existing methods in multiple challenging scenarios, including here, novel view synthesis from 8 highly degraded inputs.

inputs well, and to have a denoising effect in synthesized views by discarding inconsistent information across input views [18, 26]. This observation opens the door to 3D-based multi-frame denoising, by recasting the problem as a special case of novel view synthesis where the input views are noisy and the target views are the clean input views [26, 31].

Recently, novel view synthesis has been approached as an encoding-rendering process where a scene representation is first *encoded* from a set of input images and an arbitrary number of novel views are then *rendered* from this scene representation. In the Neural Radiance Field (NeRF) framework for instance, the scene representation is a radiance field function encoded by training a neural network on the input views. Novel views are then rendered by querying and integrating this radiance field function over light rays originating from a target camera position [2, 24, 28]. In the Multiplane Image (MPI) framework on the other hand, the scene representation is a stack of semi-transparent colored layers arranged at various depths, encoded by feeding the input views to a neural network trained on a large number of scenes. Novel views are then rendered by warping and overcompositing the semi-transparent layers [8, 41, 55].

In the present work, we adopt the MPI framework because it is much lighter than the NeRF framework computationally. The encoding stage only requires one inference pass on a network that generalizes to new scenes instead of training one neural network per-scene, and the rendering stage is essentially free instead of requiring a large number of inference passes. However, the standard MPI pipeline struggles to predict multiplane representations that are self-consistent across depths from multiple viewpoints. This problem can lead to depth-discretization artifacts in synthesized views [40] and has previously been addressed at the encoding stage using computationally expensive mechanisms and a large number of depth planes [8, 11, 27, 40]. Here, we propose to enforce cross-depth consistency at the rendering stage by replacing the fixed overcompositing operator with a learnable renderer. This change of approach has three important implications. First, the encoder module can now process depths independently from each other and focus on fusing information across views. This significantly reduces the computational load of the encoding stage. Second, the scene representation changes from a static MPI to Multiplane Features (MPF) rendered dynamically. This significantly increases the expressive power of the scene encoding. Finally, the framework’s overall performance is greatly improved, making it suitable for novel scenarios including multi-frame denoising where it outperforms standard 2D-based approaches at a fraction of their computational cost. Our main contributions are as follow:

- We solve the cross-depth consistency problem for multiplane representations at the rendering stage, by introducing a learnable renderer.
- We introduce the Multiplane Feature (MPF) representation, a generalization of the multiplane image with higher representational power.
- We re-purpose the multiplane image framework originally developed for novel view synthesis to perform 3D-based multi-frame denoising.
- We validate the approach with experiments on 3 tasks and 3 datasets and significantly outperform existing 2D-based and 3D-based methods for multi-frame denoising.

## 2. Related work

**Multi-frame denoising** Multi-frame restoration methods are frequently divided into two categories, depending on the type of image alignment employed. *Explicit alignment* refers to the direct warping of images using optical flows predicted by a motion compensation module [4, 43, 44, 54]. In contrast, *implicit alignment* refers to local, data-driven deformations implemented using dynamic upsampling filters [16, 17], deformable convolutions [45, 49], kernel prediction networks [25] or their extension, basis prediction networks [53]. Explicit alignment is better at dealing with large motion while implicit alignment is better at deal-

ing with residual motion, and state-of-the-art performance can be achieved by combining both in the form of flow-guided deformable convolutions [6, 7]. Another distinction between multi-frame restoration methods is the type of processing used. A common approach is to concatenate the input frames together along the channel dimension [4, 43, 44, 49, 54] but recurrent processing is more efficient [9, 10, 34, 42], especially when implemented in a bidirectional way [5, 7, 14]. BasicVSR++ achieves state-of-the-art performance by combining flow-guided deformable alignment with bidirectional recurrent processing iterated multiple times [7]. In a different spirit, the recent DeepRep method [3] introduces a deep reparameterization in feature space of the maximum a posteriori formulation of multi-frame restoration. Similarly to the previous methods however, it still uses a form of explicit 2D alignment, and lacks any ability to reason about the 3D structure of the scene.

**View synthesis** The idea to decompose a scene into a set of semi-transparent planes can be traced back to the use of mattes and blue screens in special effects filmmaking [38, 47]. This scene representation was first applied to view interpolation in [41], and recently gained popularity under the name of Multiplane Image (MPI) [55]. It is particularly powerful to generate novel views from a small set of forward facing views [8, 27, 55], and can even be used to generate novel views from a single image [11, 20, 46]. The rendering of view dependent-effects and non-Lambertian surfaces is challenging due to the use of a single set of  $RGB_\alpha$  images, and can be improved by predicting multiple MPIs combined as a weighted average of the distance from the input views [27], or as a set of basis components [51]. The simplicity of this representation is appealing, but it can still be computationally heavy when the number of depth planes grows [8, 51], and the rendered views can suffer from depth discretization artifacts [40]. A number of alternative layered scene representations exist, including the Layered Depth Image (LDI) consisting in one RGB image with an extra depth channel [36], and variants of MPIs and LDIs [13, 19, 21, 39]. So far however, all these methods use a fixed overcompositing operator at the rendering stage. The idea to perform view synthesis by applying 3D operations in the feature space of an encoder-decoder architecture was explored on simple geometries in [52], and used with success on a point-cloud scene representation for view synthesis from a single image in [50]. Recently, Neural Radiance Fields (NeRFs) have become highly popular for their ability to produce high quality renderings of complex scenes from arbitrary viewpoints [2, 24, 28–30]. However, they tend to be very heavy computationally, require a large number of input views, and lack the ability to generalize to novel scenes. IBRNet [48] improves generalizability by learning a generic view interpolation function, but the approach remains computationally heavy. The application of view syn-

thesis approaches to multi-frame restoration has been limited so far, and exclusively based on NeRF. RawNeRF [26] explores novel view synthesis in low-light conditions, and reports strong denoising effects in the synthesized views. Deblur-NeRF [22] augments the ability of NeRFs to deal with blur by including Deformable Sparse Kernels. Noise-aware-NeRFs [31] improves the IBRNet architecture to explicitly deal with noise. However, these different restoration approaches still suffer from the limitations affecting their underlying NeRF representations.

### 3. Method

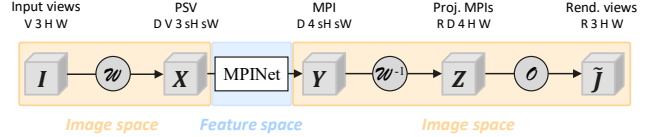
We start by describing the standard MPI processing pipeline, before discussing the cross-depth consistency problem. We then introduce our MPF Encoder-Renderer and its adaptation to multi-frame denoising.

#### 3.1. Standard MPI processing pipeline

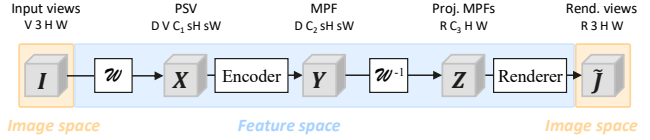
The standard MPI processing pipeline turns a set of input views into an arbitrary set of rendered novel views by applying 4 main transformations: forward-warping, MPI prediction, backward-warping, overcompositing (See Figure 2a). We describe this pipeline in more details below.

**Input views** The inputs of the pipeline are a set of  $V$  views of a scene, consisting of images and camera parameters. The images are of height  $H$  and width  $W$ , with red-green-blue color channels, and can be stacked into a 4D tensor  $\mathbf{I} = \{\{\{\{\mathbf{I}_{vchw}\}_{w=1}^W\}_{h=1}^H\}_{c=1}^3\}_{v=1}^V$ . To simplify notations, we omit the dimensions  $c, h, w$  and refer to an individual image as  $\mathbf{I}_v$ . The camera parameters consist of an intrinsic tensor  $\mathbf{K}$  of size  $V \times 3 \times 3$  containing information about the focal lengths and principal point of the cameras, and an extrinsic tensor containing information about the camera orientations in the world coordinate system, that can be split into a rotation tensor  $\mathbf{R}$  of size  $V \times 3 \times 3$  and a translation tensor  $\mathbf{t}$  of size  $V \times 3 \times 1$ . A reference view  $i$  is defined a priori, and the positions of all the cameras are assumed to be expressed relatively to it. The intrinsic matrix  $\mathbf{K}_i$  of the reference camera is defined such that the corresponding field of view covers the region of space visible to all the input views. Finally, a set of  $D$  depth planes is distributed orthogonally to the reference viewing direction such that their normal is  $\mathbf{n} = (0, 0, 1)^\top$ , and their distances  $\{a_d\}_{d=1}^D$  from the reference camera center are sampled uniformly in disparity. The camera parameters and the depth planes are used to define a set of  $D \times V$  homography projections, represented by a tensor  $\mathbf{H}$  of size  $D \times V \times 3 \times 3$ . Each homography is between one of the input views and the reference view, and is induced by one of the depth planes, such that its matrix is expressed as [12]:

$$\mathbf{H}_{dv} = \mathbf{K}_v \left( \mathbf{R}_v - \frac{\mathbf{t}_v \mathbf{n}^\top}{a_d} \right) \mathbf{K}_i^{-1} \quad (1)$$



(a) Standard MPI processing pipeline.



(b) Our MPF Encoder-Renderer.

Figure 2. In the standard MPI processing pipeline, all the learning and most of the processing happens in the MPINet module. We propose to move the multiplane representation to feature space, by giving some processing power to the warping operators, and replacing the overcompositing operator with a learnable renderer.

**Plane Sweep Volumes** The first transformation in the MPI pipeline is the computation of Plane Sweep Volumes (PSVs), obtained by forward-warping each input image  $D$  times, according to the homography  $\mathbf{H}_{dv}$ . The sampling rate of the warping operator  $\mathcal{W}$  is a hyperparameter that can be controlled using an up-scaling factor  $s$ . Each transformation can thus be written as  $\mathbf{X}_{dv} = \mathcal{W}(\mathbf{I}_v, \mathbf{H}_{dv}, s)$  and the PSV tensor  $\mathbf{X}$  is of size  $D \times V \times 3 \times sH \times sW$ .

**Multiplane Image** The main processing block of the pipeline is a neural network MPINet, turning the set of PSVs into a multiplane image representation of the scene  $\mathbf{Y} = \text{MPINet}(\mathbf{X})$  where  $\mathbf{Y}$  is a set of  $D$  semi-transparent RGB images of size  $D \times 4 \times sH \times sW$ , constrained to the  $[0, 1]$  range by using a sigmoid activation function.

**Projected MPIs** The MPI is then backward-warped to a set of  $R$  novel views, defined by an homography tensor  $\mathbf{G}$  of size  $R \times D \times 3 \times 3$  following Eq. (1) with the depth and view dimensions transposed. The backward-warping operation is defined as  $\mathbf{Z}_{rd} = \mathcal{W}(\mathbf{Y}_d, \mathbf{G}_{rd}^{-1}, 1/s)$  obtaining a tensor of projected MPIs  $\mathbf{Z}$  of size  $R \times D \times 4 \times H \times W$ .

**Rendered views** The projected MPIs are finally collapsed into single RGB images by applying the overcompositing operator  $\mathcal{O}$ . This operator splits each  $\text{RGB}\alpha$  image  $\mathbf{Z}_{rd}$  into a colour component  $\mathbf{C}_{rd}$  and an alpha component  $\mathbf{A}_{rd}$  and computes  $\tilde{\mathbf{J}}_r = \sum_{d=1}^D \left( \mathbf{C}_{rd} \mathbf{A}_{rd} \prod_{k=d+1}^D (1 - \mathbf{A}_{rk}) \right)$  obtaining the rendered views  $\tilde{\mathbf{J}}$  of size  $R \times 3 \times H \times W$ .

**Training** The pipeline is typically trained end-to-end in a supervised way by minimizing a loss  $\mathcal{L}(\tilde{\mathbf{J}}, \mathbf{J})$  between the rendered views  $\tilde{\mathbf{J}}$  and the corresponding ground-truth images  $\mathbf{J}$ . In practice,  $\mathcal{L}$  is often an  $\mathcal{L}_1$  loss applied to low level features of a VGG network [37].

### 3.2. Cross-depth consistency

The main and only learnable module of the standard MPI pipeline is the prediction network MPINet, transforming a 5D PSV tensor into a 4D MPI scene representation. Its task is challenging because multiplane images are hyperconstrained representations: their semi-transparent layers interact with each other in non-trivial and view-dependent ways, and missing or redundant information across layers can result in depth discretization artifacts after applying the overcompositing operator [40]. To perform well, MPINet requires a mechanism enforcing *cross-depth consistency*; and several approaches have been considered before.

In the original case of stereo-magnification [55], there is one reference input and one secondary input which is turned into a single PSV. Both are concatenated along the channel dimension and fed to an MPINet module predicting the full MPI in one shot as a 3D tensor of size  $(D \times 4) \times sH \times sW$ , such that cross-depth consistency is enforced within the convolutional layers of MPINet. A similar solution can be used in the case of single-view view synthesis [46], where there is a single reference input image and no PSV. In the general case with  $V$  inputs, however, the PSV tensor  $\mathbf{X}$  becomes very large and there are two main ways to process it.

**Option 1** The first solution is to generalize the approach of [55] and concatenate  $\mathbf{X}$  across views and depths before feeding it to a network predicting the full MPI in one shot:  $\mathbf{Y} = \text{MPINet}(\{\{\mathbf{X}_{dv}\}_{v=1}^V\}_{d=1}^D)$ .

**Option 2** The second solution is to concatenate  $\mathbf{X}$  across views, and process each depth separately—effectively running the MPINet block  $D$  times in parallel:  $\mathbf{Y} = \{\text{MPINet}(\{\mathbf{X}_{dv}\}_{v=1}^V)\}_{d=1}^D$ .

Option 1 tends to work poorly in practice [8], as it requires to either use very large convolutional layers with intractable memory requirements, or discard most of the information contained in the input PSVs after the first convolutional layer. Option 2 is appealing as it fuses information across views more effectively, but the resulting MPI typically suffers from a lack of cross-depth consistency as each depth is processed separately. Most previous works adopt Option 2 as a starting point, and augment it with various mechanisms allowing some information to flow across depths. For instance, some methods implement MPINet with 3D convolutions [27, 40], such that each depth is treated semi-independently within a local depth neighborhood dependent on the size of the kernel, which is typically 3. By design however, this solution cannot handle interactions between distant depth planes. DeepView [8] proposes to solve the recursive cross-depth constraint by iteratively refining the prediction of the MPINet block, effectively performing a form of learned gradient descent. However, this solution requires to run the MPI network multiple times, which is computationally heavy. Finally, the method of [11] uses a feature

masking strategy to deal with inter-plane interactions explicitly. This solution is both complex (multiple networks are required to predict the masks) and rigid (the masking operations are fixed and still work on a per-depth basis).

### 3.3. Our MPF Encoder-Renderer

Here, we propose to solve the cross-depth consistency problem in a novel way, by addressing it at the rendering stage. Specifically, we replace the fixed overcompositing operator with a learnable renderer enforcing consistency directly at the output level on a per-view basis. This design change greatly simplifies the task of the multiplane representation encoder, which can now focus on fusing information across views in a depth-independent way. It also promotes the multiplane representation to feature space, by relaxing existing constraints on the number of channels and scaling in the  $[0,1]$  range. The four transformations of the pipeline are modified as follows (See Figure 2b).

**Plane Sweep Volumes** To decrease the amount of information loss through image warping and promote the PSVs to feature space, we now apply a convolution to the images before warping them:  $\mathbf{X}_{dv} = \mathcal{W}(\text{Conv}(\mathbf{I}_v), \mathbf{H}_{dv}, s)$ . The PSV tensor  $\mathbf{X}$  is now of size  $D \times V \times C_1 \times sH \times sW$  where the number of channels  $C_1$  is a hyperparameter.

**Multiplane Features** We then replace the MPINet module with an encoder applied to each depth independently:  $\mathbf{Y} = \{\text{Encoder}(\{\mathbf{X}_{dv}\}_{v=1}^V)\}_{d=1}^D$ . The multiplane representation  $\mathbf{Y}$  is now in feature space: its range is not constrained by a sigmoid function anymore and its size is  $D \times C_2 \times sH \times sW$  where the number of channels  $C_2$  is a second hyperparameter.

**Projected MPFs** The MPF is still backward-warped to a set of  $R$  novel views defined by an homography tensor  $\mathbf{G}$ , but this is now followed by a convolution collapsing the depth and channel dimensions into a single dimension:  $\mathbf{Z}_r = \text{Conv}(\{\mathcal{W}(\mathbf{Y}_d, \mathbf{G}_{dr}^{-1}, 1/s)\}_{d=1}^D)$ . The tensor  $\mathbf{Z}$  is of size  $R \times C_3 \times H \times W$  where  $C_3$  is a third hyperparameter.

**Rendered views** Finally, the projected MPFs are turned into the final rendered views by a simple CNN renderer operating on each view separately:  $\tilde{\mathbf{J}} = \{\text{Renderer}(\mathbf{Z}_r)\}_{r=1}^R$ . The rendered views  $\tilde{\mathbf{J}}$  are of size  $R \times 3 \times H \times W$ .

**Training** The pipeline is still trained end-to-end by minimizing a loss  $\mathcal{L}(\tilde{\mathbf{J}}, \mathbf{J})$  between the rendered views  $\tilde{\mathbf{J}}$  and the corresponding ground-truth images  $\mathbf{J}$ , but the renderer and the warping operators are now learnable. When the input views  $\mathbf{I}$  are noisy versions of ground-truth images  $\mathbf{I}^*$ , the pipeline can be turned into a multi-frame denoising method by using the input homographies  $\mathbf{H}$  during backward-warping, obtaining denoised outputs  $\tilde{\mathbf{I}}$ , and by minimizing the loss  $\mathcal{L}(\tilde{\mathbf{I}}, \mathbf{I}^*)$ . There is then a one-to-one mapping between the input views and the rendered views,

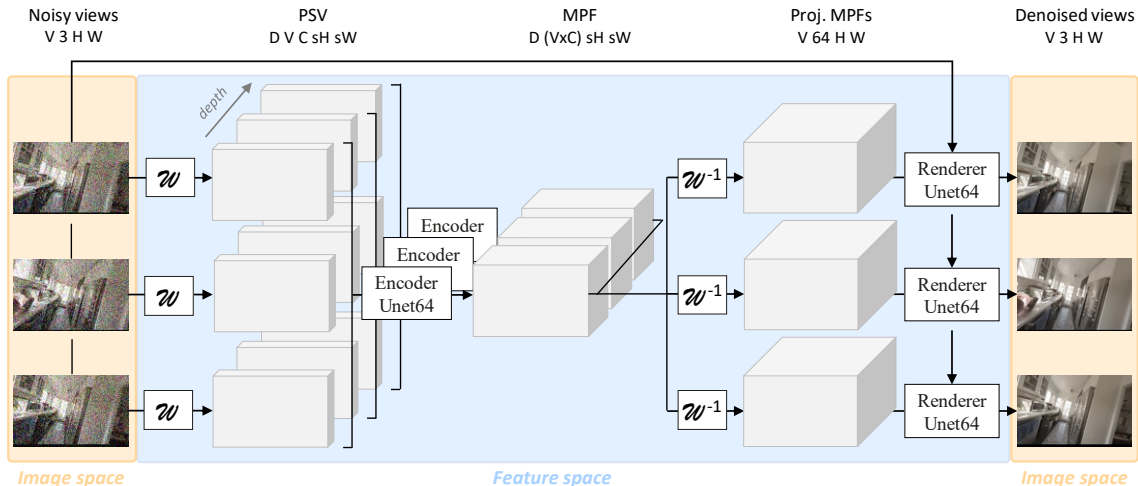


Figure 3. Our Multiplane Features Encoder-Renderer (MPFER). Input views are forward-warped into plane sweep volumes (PSVs) which are processed depthwise by the Encoder Unet64. The resulting multiplane feature representation (MPF) can then be back-projected to an arbitrary number of novel views, or to the same views as the inputs—allowing the integration of a skip connection (illustrated here). The Renderer Unet64 processes the projected MPFs on a per-view basis, producing the final synthesised or denoised outputs.

and it is possible to integrate a skip connection feeding the noisy inputs directly to the renderer to guide its final predictions. In all our experiments, we use Unets [33] with a base of 64 channels to implement both the encoder and the renderer. We set  $C_1 = C$ ,  $C_2 = V \times C$  and  $C_3 = 64$  such that there is a single hyperparameter  $C$  to vary. Our Multiplane Features Encoder-Renderer (MPFER) is illustrated in Figure 3 for 3 input views and 3 depth planes. A MindSpore [15] implementation of our method is available<sup>1</sup>.

## 4. Experiments

We first consider the Spaces dataset [8] and validate our approach on novel view synthesis. We then focus on a denoising setup and perform extensive comparisons to state-of-the-art 2D-based methods. Finally, we compare our approach to the 3D-based multi-frame denoising method of [31] by replicating their experimental setup on the Real Forward-Facing dataset. In all cases, our method outperforms competitors at a fraction of the computational cost.

### 4.1. Spaces

The Spaces dataset [8] consists of 100 indoor and outdoor scenes, captured 5 to 10 times each using a 16-camera rig placed at slightly different locations. 90 scenes are used for training and 10 scenes are held-out for evaluation. The resolution of the images is  $480 \times 800$ .

**Novel view synthesis** We start by replicating the novel view synthesis setup of DeepView [8] with four scenarios: one with 12 input views and three with 4 input views. Similarly to DeepView, we use a VGG loss and train our models

for 100k steps using the Adam optimizer, with a learning rate of  $1.5e-3$ . We reduce the learning rate to  $1.5e-4$  after 80k steps, and use a batch size of 4. Memory usage was reported to be a major challenge in DeepView, and all our models are kept at a significantly smaller size to avoid this issue. While DeepView uses a sophisticated strategy to only generate enough of the MPI to render a  $32 \times 32$  crop in the target image, we use a large patch size of 192 and only apply the loss on the region of the patch that contains more than 80% of the depths planes after backward warping. We compute all metrics by averaging over the validation scenes and target views of each setup, and after cropping a boundary of 16 pixels on all images as done in [8]. We compare to DeepView [8] and Soft3D [32] by using the synthesised images provided with the Spaces dataset. We also consider three variants of the standard MPI pipeline using the same Unet backbone as our MPFER method and trained in the same conditions, but processing the input PSV in different ways. MPINet implements *Option 1* from Sec. 3.2. The views and depths dimensions of the PSV tensor are stacked along the channel dimension and fed to the Unet backbone to predict the output MPI in one shot. MPINet-dw implements *Option 2* from Sec. 3.2. The Unet backbone runs depthwise on slices of the PSV to predict each depth plane of the MPI separately, without communication mechanism across depths. Finally, MPINet-dw-it implements a one-step version of the learned gradient descent algorithm of DeepView. A first estimate of the MPI is predicted by a Unet backbone running depthwise, and this estimate is fed to a second Unet backbone also running depthwise, along with the input PSV and gradient components ( $\mathbb{R}$ ), which are PSV-projected current estimates of the target views.

<sup>1</sup><https://github.com/mindspore-lab/mindediting>

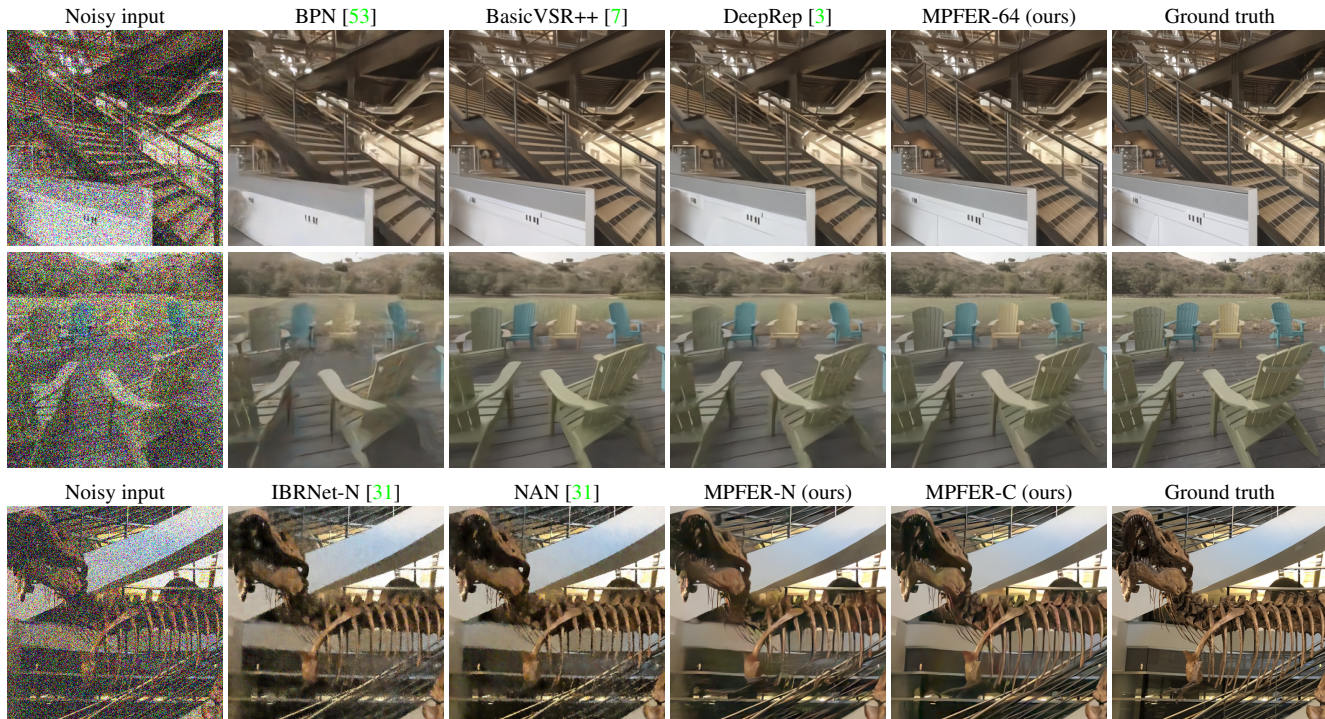


Figure 4. Qualitative evaluation for multi-frame denoising with Gain 20 (best viewed zoomed in). We compare MPFER to 2D-based methods on Spaces (top) and to 3D-based methods on the Real Forward-Facing dataset (bottom).

For our MPFER method and the MPINet ablations, we use a number of depth planes  $D = 64$  distributed between 100 and 0.5 meters away from the reference camera, placed at the average position of the input views. We use a number of channels  $C = 8$  and a PSV/MPF upscaling factor  $s = 1.5$ . Since the Unet backbone is not agnostic to the number of input views, we train one version of each model for the setup with 12 input views and one version for the three setups with 4 input views. The results are presented in Table 1. We observe a clear progression between the performances of MPINet, MPINet-dw and MPINet-dw-it, illustrating the benefit of each design change. Our MPFER method outperforms MPINet-dw-it by up to 4dBs in PSNR at a similar computational complexity, and outperforms DeepView by up to 1.8dB at a fraction of the complexity, clearly motivating the use of a learnt renderer for efficient depth fusion.

**Multi-frame denoising** We now consider a different setup where the inputs are 16 views from one rig position with images degraded with noise, and the targets are the same 16 views denoised. Similarly to previous works [3, 25, 31, 53], we apply synthetic noise with a signal dependent Gaussian distribution  $I_{vchw} \sim \mathcal{N}(I_{vchw}^*, \sigma_r^2 + \sigma_s I_{vchw}^*)$  where  $I$  is the tensor of noisy inputs,  $I^*$  is the ground truth signal, and  $\sigma_r$  and  $\sigma_s$  are noise parameters that are fixed for each sequence. We focus in particular on challenging sce-

narios with moderate to high gain levels [4, 8, 16, 20], corresponding to the  $(\log(\sigma_r), \log(\sigma_s))$  values  $[(-1.44, -1.84), (-1.08, -1.48), (-0.72, -1.12), (-0.6, -1.0)]$  respectively.

We consider two patch-based approaches: VBM4D [23] and VNLB [1], as well as four state-of-the-art learning-based methods: BPN [53], BasicVSR [5] and its extension BasicVSR++ [7], and DeepRep [3]. To evaluate the influence of the model size and in particular the number of depth planes, we train three MPFER models: MPFER-16 with  $(D, C, s) = (16, 8, 1)$ , MPFER-32 with  $(D, C, s) = (32, 16, 1.25)$ , and MPFER-64 with  $(D, C, s) = (64, 8, 1.25)$ . MPFER-16 has the particularity of using the same number of depth planes as there are input images, meaning that the number of Unet passes per frame to denoise the sequence is  $(D + V)/V = 2$ . This observation motivates us to perform a comparison with three other architectures, using a strict computational budget of 2 Unet passes per frame. Unet-SF (for Single-Frame) is constituted of two Unet blocks without temporal connection, therefore processing the sequence as a disjoint set of single frames. Unet-BR (for Bidirectional-Recurrent) is constituted of two Unet blocks with bidirectional recurrent connections: the lower Unet processes the sequence in a backward way, and the higher Unet processes the sequence in a feedforward way. Finally, Unet-BR-OF (for Bidirectional-Recurrent with Optical-Flow alignment) is constituted of two Unet

blocks with bidirectional recurrent connections, and the recurrent hidden-state is aligned using a SpyNet module, as done in basicVSR [5]. We train all the models in the same conditions as for the novel view synthesis setup, except for the patch size which we increase to 256, and the loss which we replace with a simple L1 loss. During training, we vary the gain level randomly and concatenate an estimate of the standard deviation of the noise to the input, as in [25, 31, 53]. We evaluate on the first rig position of the 10 validation scenes of the Spaces dataset for the 4 gain levels without boundary-cropping, and present the results in Table 2. Each model receives 16 noisy images as input and produces 16 restored images as output, except for BPN and DeepRep which are burst processing methods and only produce one output. For these methods, we choose the view number 6 at the center of the camera rig as the target output, and compare the performances of all methods on this frame. Our MPFER method clearly outperforms all the other methods at a fraction of the computational cost. It performs particularly strongly at high noise levels, with improvements over other methods of more than 2dBs in PSNR. MPFER-16 also performs remarkably well, despite using only 16 depth planes. This suggests that the high representational power of multiplane features allows to significantly reduce the number of depth planes—and therefore the computational cost—compared to standard MPI approaches, which typically use a very high number of planes (80 in the case of DeepView [8], up to 192 in the case of NeX [51]). A qualitative evaluation is available in Figure 4 (top), and we observe that MPFER is able to reconstruct scenes with much better details. We also present a visualization of multiplane features in Figure 6, illustrating how the model learns to separate depths in an unsupervised way.

## 4.2. LLFF-N

The LLFF-N dataset [31] is a variant of the Real Forward-Facing dataset [27] where images are linearized by applying inverse gamma correction and random inverse white balancing, and synthetic noise is applied following the same signal dependent Gaussian distribution as used in the previous section with the six gain levels [1, 2, 4, 8, 16, 20]. The dataset contains 35 scenes for training and 8 scenes for testing, and the resolution of the images is  $756 \times 1008$ .

**Denoising** In this setup, the model receives 8 frames in input: the target frame plus its 7 nearest neighbors in terms of camera distances. We train one MPFER model with  $(D, C, s) = (64, 8, 1.25)$ , using an  $\mathcal{L}_1$  loss applied to the target frame. We evaluate on the 43 bursts used in [31] (every 8th frame of the test set) and present the results in the first half of Table 3. A qualitative evaluation is also available in Figure 4 (bottom). To assess the robustness of our method to noisy camera positions, we evaluate it using camera positions computed on clean images (MPFER-C) and

on noisy images (MPFER-N) using COLMAP [35]. Our method outperforms IBRNet-N and NAN in both scenarios by large margins, but the evaluation using clean camera poses performs significantly better at high noise levels.

**Synthesis under noisy conditions** In this setup, the model receives as input the 8 nearest neighbors to a held-out target. Again, we train one MPFER model with  $(D, C, s) = (64, 8, 1.25)$ , using an  $\mathcal{L}_1$  loss applied to the target frame. We evaluate on the same 43 bursts as before and report the results in the second half of Table 3. Our method performs on par with IBRNet and NAN at very low noise levels (close to a pure synthesis problem), and significantly outperforms the other methods at larger noise levels. MPFER only requires  $D$  Unet passes to produce an MPF, and 1 Unet pass to render a novel view, which is significantly lighter than IBRNet and NAN. At inference time, the Unet pass requires 0.6 Mflops per pixel, compared to 45 Mflops for IBRNet [48]. A qualitative evaluation is available in Figure 1 for Gain 20.

**Low-Light Scenes** Finally, we qualitatively evaluate our denoising model trained on LLFF-N on sequences with real noise captured with a Google Pixel 4 under low-light conditions. We use the sequences from [31] and estimate camera poses using COLMAP [35] on noisy images. We compare our results to those of [31]—downloaded from their project page where more baselines can be found—in Figure 5.

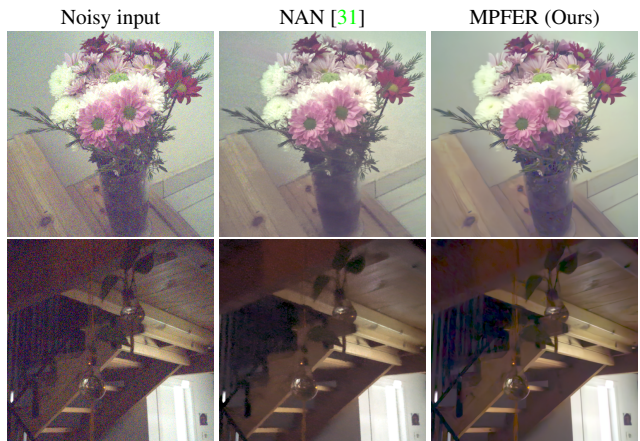


Figure 5. Qualitative evaluation on sequences with real noise from [31], captured with a Google Pixel 4.

## 5. Conclusion

We proposed to approach multi-frame denoising as a view synthesis problem and argued in favor of using multiplane representations for their low computational cost and generalizability. We introduced a powerful generalization of multiplane images to feature space, and demonstrated its effectiveness in multiple challenging scenarios.

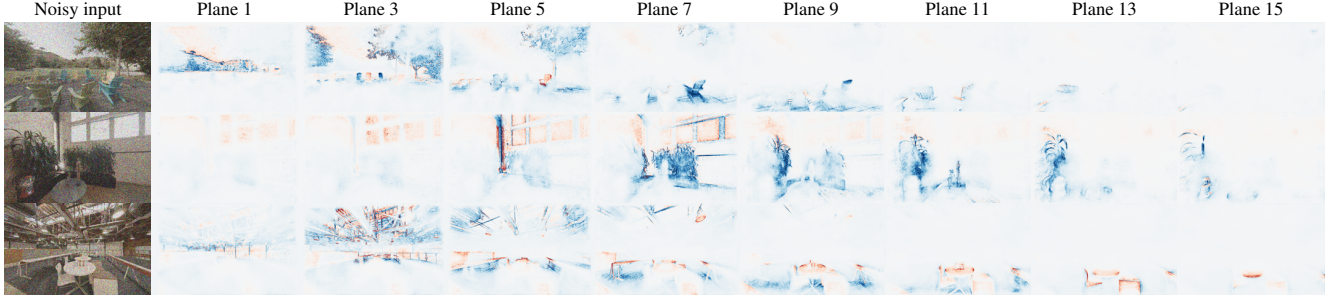


Figure 6. Visualization of Multiplane Features for 3 scenes. We plot the first 3 channels of planes 1, 3, 5, 7, 9, 11, 13, 15 from MPFER-16. Our method learns to separate depths in an unsupervised way, in this case from a pure denoising problem.

	12 input views (dense)			4 input views (small)			4 input views (medium)			4 input views (large)			GFlops@ 500×800
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	
Soft3D*	31.93	0.940	0.052	30.29	0.925	0.064	30.84	0.930	0.060	30.57	0.931	0.054	n/a
DeepView*	<u>34.23</u>	0.965	<u>0.015</u>	<u>31.42</u>	0.954	0.026	<u>32.38</u>	<u>0.957</u>	<u>0.021</u>	<u>31.00</u>	<u>0.952</u>	<u>0.024</u>	45800
MPINet	27.43	0.914	0.035	27.00	0.906	0.054	26.16	0.896	0.062	24.93	0.865	0.085	450
MPINet-dw	30.70	0.963	0.021	29.39	0.951	0.027	28.47	0.948	0.030	26.83	0.937	0.040	7890
MPINet-dw-it	30.85	<u>0.966</u>	0.017	30.22	<u>0.955</u>	<u>0.024</u>	29.37	0.953	0.026	28.00	0.943	0.034	14800
MPFER-64	<b>35.73</b>	<b>0.972</b>	<b>0.012</b>	<b>33.20</b>	<b>0.959</b>	<b>0.018</b>	<b>33.47</b>	<b>0.959</b>	<b>0.018</b>	<b>32.38</b>	<b>0.953</b>	<b>0.021</b>	8490

Table 1. Novel view synthesis on Spaces. All metrics were computed on predicted images with a 16-pixel boundary cropped, as done in [8]. Stared methods were evaluated using the predicted images released with the Spaces dataset.

	Gain 4			Gain 8			Gain 16			Gain 20			GFlops@ 500×800
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	
VBM4D	32.00	0.900	0.108	29.94	0.850	0.172	27.48	0.769	0.280	26.55	0.730	0.331	n/a
VNLB	33.41	0.918	0.089	30.30	0.871	0.144	25.74	0.793	0.283	23.51	0.743	0.366	n/a
BPN	34.52	0.934	0.048	32.10	0.900	0.082	29.45	0.846	0.144	28.56	0.824	0.168	810
BasicVSR	36.87	0.959	0.027	34.52	0.937	0.049	31.73	0.898	0.095	30.68	0.879	0.119	2090
BasicVSR++	36.98	0.959	0.026	34.66	0.938	0.045	31.97	0.902	0.083	30.92	0.883	0.102	4300
DeepRep	37.37	0.963	0.024	35.13	0.943	0.043	32.37	0.906	0.085	31.32	0.888	0.107	3230
UNet-SF	35.10	0.942	0.043	32.62	0.909	0.075	29.81	0.857	0.134	28.81	0.834	0.161	440
UNet-BR	35.19	0.943	0.040	32.67	0.912	0.070	29.90	0.861	0.124	28.94	0.840	0.148	470
UNet-BR-OF	36.41	0.956	0.029	34.27	0.936	0.048	31.77	0.899	0.089	30.85	0.882	0.110	710
MPFER-16	37.56	<u>0.968</u>	0.020	35.80	0.955	0.030	33.70	<u>0.933</u>	<u>0.051</u>	32.89	0.921	0.063	<b>470</b>
MPFER-32	<u>37.94</u>	<b>0.970</b>	<u>0.019</u>	<u>36.17</u>	<u>0.958</u>	<u>0.028</u>	<u>33.99</u>	<b>0.936</b>	<b>0.047</b>	<u>33.14</u>	<u>0.924</u>	<u>0.058</u>	1210
MPFER-64	<b>38.00</b>	<b>0.970</b>	<b>0.018</b>	<b>36.25</b>	<b>0.959</b>	<b>0.027</b>	<b>34.08</b>	<b>0.936</b>	<b>0.047</b>	<b>33.23</b>	<b>0.925</b>	<b>0.057</b>	1810

Table 2. Denoising on Spaces. All metrics were computed on frame 6, as BPN and DeepRep are burst-denoising methods producing only one output. For the other methods, average performances over the entire sequence are provided in Supplementary Material.

	Gain 1			Gain 2			Gain 4			Gain 8			Gain 16			Gain 20		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
<i>Denoising of Synthetic Noise</i>																		
IBRNet-N*	33.50	0.915	0.039	31.29	0.877	0.070	29.01	0.822	0.123	26.57	0.741	0.210	24.19	0.634	0.331	23.40	0.591	0.380
NAN*	35.84	0.955	0.018	33.67	0.930	0.034	31.26	0.892	0.068	28.64	0.834	0.132	25.95	0.749	0.231	25.07	0.715	0.271
MPFER-N	<u>37.90</u>	<u>0.969</u>	<u>0.013</u>	<u>35.61</u>	<u>0.951</u>	<u>0.025</u>	<u>33.02</u>	<u>0.921</u>	<u>0.048</u>	<u>30.21</u>	<u>0.872</u>	<u>0.091</u>	<u>27.24</u>	<u>0.797</u>	<u>0.164</u>	<u>26.23</u>	<u>0.765</u>	<u>0.198</u>
MPFER-C	<b>38.06</b>	<b>0.971</b>	<b>0.011</b>	<b>35.95</b>	<b>0.956</b>	<b>0.020</b>	<b>33.65</b>	<b>0.934</b>	<b>0.036</b>	<b>31.21</b>	<b>0.898</b>	<b>0.065</b>	<b>28.61</b>	<b>0.843</b>	<b>0.115</b>	<b>27.71</b>	<b>0.819</b>	<b>0.138</b>
<i>Novel View Synthesis Under Noisy Conditions</i>																		
IBRNet*	<b>24.53</b>	0.774	<u>0.135</u>	24.20	0.730	0.159	23.44	0.653	0.217	22.02	0.536	0.327	19.76	0.377	0.492	18.80	0.319	0.553
IBRNet-N*	23.86	0.763	0.170	23.73	0.744	0.178	23.38	0.703	0.208	22.68	0.638	0.275	21.67	0.549	0.377	21.29	0.514	0.418
NAN*	<u>24.52</u>	<b>0.799</b>	<b>0.132</b>	<u>24.41</u>	<u>0.787</u>	<b>0.145</b>	<u>24.18</u>	<u>0.765</u>	<u>0.171</u>	<u>23.70</u>	<u>0.726</u>	<u>0.221</u>	<u>22.79</u>	<u>0.666</u>	<u>0.305</u>	<u>22.37</u>	<u>0.641</u>	<u>0.342</u>
MPFER	<u>24.52</u>	<u>0.798</u>	0.157	<b>24.51</b>	<b>0.796</b>	<u>0.158</u>	<b>24.47</b>	<b>0.789</b>	<b>0.164</b>	<b>24.33</b>	<b>0.775</b>	<b>0.178</b>	<b>23.94</b>	<b>0.746</b>	<b>0.212</b>	<b>23.72</b>	<b>0.731</b>	<b>0.230</b>

Table 3. LLFF-N. We consider the two scenarios introduced in [31]: Denoising of Synthetic Noise, where the noisy target is accessible, and Novel View Synthesis Under Noisy Conditions, where the noisy target is held-out. The numbers for the starred methods correspond to Figure 9 in [31], and were provided by the authors.



## References

- [1] Pablo Arias and Jean-Michel Morel. Video denoising via empirical bayesian estimation of space-time patches. *Journal of Mathematical Imaging and Vision*, 60(1):70–93, 2018. 6
- [2] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *ICCV*, pages 5855–5864, 2021. 1, 2
- [3] Goutam Bhat, Martin Danelljan, Fisher Yu, Luc Van Gool, and Radu Timofte. Deep reparametrization of multi-frame super-resolution and denoising. In *ICCV*, pages 2460–2470, 2021. 1, 2, 6
- [4] Jose Caballero, Christian Ledig, Andrew Aitken, Alejandro Acosta, Johannes Totz, Zehan Wang, and Wenzhe Shi. Real-time video super-resolution with spatio-temporal networks and motion compensation. In *CVPR*, pages 4778–4787, 2017. 2
- [5] Kelvin CK Chan, Xintao Wang, Ke Yu, Chao Dong, and Chen Change Loy. Basicvsr: The search for essential components in video super-resolution and beyond. In *CVPR*, pages 4947–4956, 2021. 2, 6, 7
- [6] Kelvin CK Chan, Xintao Wang, Ke Yu, Chao Dong, and Chen Change Loy. Understanding deformable alignment in video super-resolution. In *AAAI*, pages 973–981, 2021. 2
- [7] Kelvin CK Chan, Shangchen Zhou, Xiangyu Xu, and Chen Change Loy. Basicvsr++: Improving video super-resolution with enhanced propagation and alignment. In *CVPR*, pages 5972–5981, 2022. 1, 2, 6
- [8] 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 *CVPR*, pages 2367–2376, 2019. 1, 2, 4, 5, 7, 8
- [9] Dario Fuoli, Shuhang Gu, and Radu Timofte. Efficient video super-resolution through recurrent latent space propagation. In *ICCVW*, pages 3476–3485. IEEE, 2019. 2
- [10] Clément Godard, Kevin Matzen, and Matt Uyttendaele. Deep burst denoising. In *ECCV*, pages 538–554, 2018. 2
- [11] Yuxuan Han, Ruicheng Wang, and Jiaolong Yang. Single-view view synthesis in the wild with learned adaptive multi-plane images. In *ACM SIGGRAPH*, 2022. 2, 4
- [12] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003. 3
- [13] Ronghang Hu, Nikhila Ravi, Alexander C Berg, and Deepak Pathak. Worldsheets: Wrapping the world in a 3d sheet for view synthesis from a single image. In *ICCV*, pages 12528–12537, 2021. 2
- [14] Yan Huang, Wei Wang, and Liang Wang. Bidirectional recurrent convolutional networks for multi-frame super-resolution. In *NeurIPS*, 2015. 2
- [15] Huawei. Mindspore. <https://www.mindspore.cn/en>, 2020. 5
- [16] Xu Jia, Bert De Brabandere, Tinne Tuytelaars, and Luc Van Gool. Dynamic filter networks. In *NeurIPS*, 2016. 2
- [17] Younghyun Jo, Seoung Wug Oh, Jaeyeon Kang, and Seon Joo Kim. Deep video super-resolution network using dynamic upsampling filters without explicit motion compensation. In *CVPR*, pages 3224–3232, 2018. 2
- [18] Nima Khademi Kalantari, Ting-Chun Wang, and Ravi Ramamoorthi. Learning-based view synthesis for light field cameras. *ACM TOG*, 35(6):1–10, 2016. 1
- [19] Taras Khakhulin, Denis Korzhenkov, Pavel Solovev, Gleb Sterkin, Andrei-Timotei Ardelean, and Victor Lempitsky. Stereo magnification with multi-layer images. In *CVPR*, pages 8687–8696, 2022. 2
- [20] Jiaxin Li, Zijian Feng, Qi She, Henghui Ding, Changhu Wang, and Gim Hee Lee. Mine: Towards continuous depth mpi with nerf for novel view synthesis. In *ICCV*, pages 12578–12588, 2021. 2
- [21] 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*, pages 328–344. Springer, 2020. 2
- [22] Li Ma, Xiaoyu Li, Jing Liao, Qi Zhang, Xuan Wang, Jue Wang, and Pedro V Sander. Deblur-nerf: Neural radiance fields from blurry images. In *CVPR*, pages 12861–12870, 2022. 3
- [23] Matteo Maggioni, Giacomo Boracchi, Alessandro Foi, and Karen Egiazarian. Video denoising, deblocking, and enhancement through separable 4-d nonlocal spatiotemporal transforms. *IEEE TIP*, 21(9):3952–3966, 2012. 6
- [24] Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *CVPR*, pages 7210–7219, 2021. 1, 2
- [25] Ben Mildenhall, Jonathan T Barron, Jiawen Chen, Dillon Sharlet, Ren Ng, and Robert Carroll. Burst denoising with kernel prediction networks. In *CVPR*, pages 2502–2510, 2018. 2, 6, 7
- [26] Ben Mildenhall, Peter Hedman, Ricardo Martin-Brualla, Pratul P Srinivasan, and Jonathan T Barron. Nerf in the dark: High dynamic range view synthesis from noisy raw images. In *CVPR*, pages 16190–16199, 2022. 1, 3
- [27] 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 TOG*, 38(4):1–14, 2019. 2, 4, 7
- [28] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 1, 2
- [29] Michael Niemeyer, Jonathan T Barron, Ben Mildenhall, Mehdi SM Sajjadi, Andreas Geiger, and Noha Radwan. Regnerf: Regularizing neural radiance fields for view synthesis from sparse inputs. In *CVPR*, pages 5480–5490, 2022. 2
- [30] Keunhong Park, Utkarsh Sinha, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Steven M Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *ICCV*, pages 5865–5874, 2021. 2

- [31] Naama Pearl, Tali Treibitz, and Simon Korman. Nan: Noise-aware nerfs for burst-denoising. In *CVPR*, pages 12672–12681, 2022. [1](#), [3](#), [5](#), [6](#), [7](#), [8](#)
- [32] Eric Penner and Li Zhang. Soft 3d reconstruction for view synthesis. *ACM TOG*, 36(6):1–11, 2017. [1](#), [5](#)
- [33] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. [5](#)
- [34] Mehdi SM Sajjadi, Raviteja Vemulapalli, and Matthew Brown. Frame-recurrent video super-resolution. In *CVPR*, pages 6626–6634, 2018. [2](#)
- [35] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, pages 4104–4113, 2016. [7](#)
- [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] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. [3](#)
- [38] Alvy Ray Smith and James F Blinn. Blue screen matting. In *ACM SIGGRAPH*, pages 259–268, 1996. [2](#)
- [39] Pavel Solovev, Taras Khakhulin, and Denis Korzhenkov. Self-improving multiplane-to-layer images for novel view synthesis. In *WACV*, pages 4309–4318, 2023. [2](#)
- [40] Pratul P Srinivasan, Richard Tucker, Jonathan T Barron, Ravi Ramamoorthi, Ren Ng, and Noah Snavely. Pushing the boundaries of view extrapolation with multiplane images. In *CVPR*, pages 175–184, 2019. [2](#), [4](#)
- [41] Richard Szeliski and Polina Golland. Stereo matching with transparency and matting. In *ICCV*, pages 517–524, 1998. [1](#), [2](#)
- [42] Thomas Tanay, Aivar Sootla, Matteo Maggioni, Puneet K Dokania, Philip HS Torr, Ales Leonardis, and Greg Slabaugh. Diagnosing and preventing instabilities in recurrent video processing. *IEEE TPAMI*, 2022. [2](#)
- [43] Xin Tao, Hongyun Gao, Renjie Liao, Jue Wang, and Jiaya Jia. Detail-revealing deep video super-resolution. In *ICCV*, pages 4472–4480, 2017. [2](#)
- [44] Matias Tassano, Julie Delon, and Thomas Veit. Dvdnet: A fast network for deep video denoising. In *ICIP*, pages 1805–1809, 2019. [2](#)
- [45] Yapeng Tian, Yulun Zhang, Yun Fu, and Chenliang Xu. Tdan: Temporally-deformable alignment network for video super-resolution. In *CVPR*, pages 3360–3369, 2020. [1](#), [2](#)
- [46] Richard Tucker and Noah Snavely. Single-view view synthesis with multiplane images. In *CVPR*, pages 551–560, 2020. [2](#), [4](#)
- [47] Petro Vlahos and B Taylor. Traveling matte composite photography. *American Cinematographer Manual*, pages 430–445, 1993. [2](#)
- [48] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul P Srinivasan, Howard Zhou, Jonathan T Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. Ibrnet: Learning multi-view image-based rendering. In *CVPR*, pages 4690–4699, 2021. [1](#), [2](#), [7](#)
- [49] Xintao Wang, Kelvin CK Chan, Ke Yu, Chao Dong, and Chen Change Loy. Edvr: Video restoration with enhanced deformable convolutional networks. In *CVPRW*, 2019. [2](#)
- [50] Olivia Wiles, Georgia Gkioxari, Richard Szeliski, and Justin Johnson. Synsin: End-to-end view synthesis from a single image. In *CVPR*, pages 7467–7477, 2020. [2](#)
- [51] Suttisak Wizadwongsa, Pakkapon Phongthawee, Jiraphon Yenphraphai, and Supasorn Suwajanakorn. Nex: Real-time view synthesis with neural basis expansion. In *CVPR*, pages 8534–8543, 2021. [2](#), [7](#)
- [52] Daniel E Worrall, Stephan J Garbin, Daniyar Turmukhambetov, and Gabriel J Brostow. Interpretable transformations with encoder-decoder networks. In *ICCV*, pages 5726–5735, 2017. [2](#)
- [53] Zhihao Xia, Federico Perazzi, Michaël Gharbi, Kalyan Sunkavalli, and Ayan Chakrabarti. Basis prediction networks for effective burst denoising with large kernels. In *CVPR*, pages 11844–11853, 2020. [2](#), [6](#), [7](#)
- [54] Tianfan Xue, Baian Chen, Jiajun Wu, Donglai Wei, and William T Freeman. Video enhancement with task-oriented flow. *IJCV*, 127(8):1106–1125, 2019. [2](#)
- [55] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: learning view synthesis using multiplane images. *ACM TOG*, 37(4):1–12, 2018. [1](#), [2](#), [4](#)