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 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 dealing 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 RGBA images, and can be improved by predicting multiple MPFs 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 MPFs 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-
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 rendered 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 \( I = \{ \{ I_{cwh} \}_{w=1}^{W} \}_{h=1}^{H} \}_{v=1}^{3} \). To simplify notations, we omit the dimensions \( c, h, w \) and refer to an individual image as \( I_v \). The camera parameters consist of an intrinsic tensor \( K \) 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 \( R \) of size \( V \times 3 \times 3 \) and a translation tensor \( t \) of size \( V \times 3 \times 1 \). A reference view \( v \) is defined a priori, and the positions of all the cameras are assumed to be expressed relatively to it. The intrinsic matrix \( K_i \) 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 \( n = (0, 0, 1)^T \), 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 \( H \) of size \( D \times V \times 3 \times 3 \).

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 \( H_{dv} \). The sampling rate of the warping operator \( \mathcal{V} \) is a hyperparameter that can be controlled using an up-scaling factor \( s \). Each transformation can thus be written as \( X_{dv} = \mathcal{V}(I_v, H_{dv}, 8) \) and the PSV tensor \( X \) 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 \( Y = MPINet(X) \) where \( Y \) 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 MPFs The MPF is then backward-warped to a set of \( R \) novel views, defined by an homography tensor \( G \) 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 \( Z_{rd} = \mathcal{W}(Y_d, G_{rd}^{-1}, 1/s) \) obtaining a tensor of projected MPFs \( Z \) of size \( R \times D \times 4 \times H \times W \).

Rendered views The projected MPFs are finally collapsed into single RGB images by applying the overcompositing operator \( \mathcal{O} \). This operator splits each RGB image \( Z_{rd} \) into a colour component \( C_{rd} \) and an alpha component \( A_{rd} \) and computes \( \tilde{J}_r = \sum_{d=1}^{D} \left( C_{rd} A_{rd} \prod_{k=d+1}^{D} (1 - A_{rk}) \right) \) obtaining the rendered views \( \tilde{J} \) 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 \( \mathcal{L}(\tilde{J}, J) \) between the rendered views \( \tilde{J} \) and the corresponding ground-truth images \( J \). In practice, \( \mathcal{L} \) is often an \( \ell_1 \) 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 hyper-constrained 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 \(X\) 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 \(X\) across views and depths before feeding it to a network predicting the full MPI in one shot: \(Y = \text{MPINet}(\{(X_{dv})_{v=1}^{V} \}_{d=1}^{D})\).

**Option 2** The second solution is to concatenate \(X\) across views, and process each depth separately—effectively running the MPINet block \(D\) times in parallel:

\[
Y = \{\text{MPINet}(\{X_{dv}\}_{v=1}^{V})\}_{d=1}^{D}.
\]

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: \(X_{dv} = \mathcal{W}((\text{Conv}(I_{v}, H_{dv}, s))\). The PSV tensor \(X\) 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:

\[
Y = \{\text{Encoder}(\{X_{dv}\}_{v=1}^{V})\}_{d=1}^{D}.
\]

The multiplane representation \(Y\) 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 \(G\), but this is now followed by a convolution collapsing the depth and channel dimensions into a single dimension:

\[
Z_r = \text{Conv}(\{\mathcal{W}(Y_d, G_d^{-1}, 1/s))\}_{d=1}^{D}.
\]

The tensor \(Z\) is of size \(R \times C_{3} \times sH \times sW\), 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:

\[
\tilde{J} = \{\text{Renderer}(Z_r)\}_{r=1}^{R}.
\]

The rendered views \(\tilde{J}\) are of size \(R \times 3 \times sH \times sW\).

**Training** The pipeline is still trained end-to-end by minimizing a loss \(\mathcal{L}(\tilde{J}, J)\) between the rendered views \(\tilde{J}\) and the corresponding ground-truth images \(J\), but the renderer and the warping operators are now learnable. When the input views \(I\) are noisy versions of ground-truth images \(I^*\), the pipeline can be turned into a multi-frame denoising method by using the input homographies \(H\) during backward-warping, obtaining denoised outputs \(\tilde{I}\), and by minimizing the loss \(\mathcal{L}(\tilde{I}, I^*)\). There is then a one-to-one mapping between the input views and the rendered views,
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 inputs views and 3 depth planes. A MindSpore [15] implementation of our method is available¹.

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 480×800.

**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 32×32 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 depth 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 ($R$), which are PSV-projected current estimates of the target views.

¹[https://github.com/mindspore-lab/mindediting](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 $I_{vchw} \sim \mathcal{N}(I^*_{vchw}, \sigma^2_{r} + \sigma^2_{s}I^*_{vchw})$, 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 scenarios 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-Recurent) 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-Recurent 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 that 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 × 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 L1 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 L1 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.

![Comparative evaluation on sequences with real noise](Image)

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.

<table>
<thead>
<tr>
<th>Noisy input</th>
<th>Plane 1</th>
<th>Plane 3</th>
<th>Plane 5</th>
<th>Plane 7</th>
<th>Plane 9</th>
<th>Plane 11</th>
<th>Plane 13</th>
<th>Plane 15</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSNR↑</td>
<td>31.93</td>
<td>30.29</td>
<td>30.84</td>
<td>30.57</td>
<td>31.00</td>
<td>24.93</td>
<td>n/a</td>
<td></td>
</tr>
<tr>
<td>SSIM↑</td>
<td>0.940</td>
<td>0.925</td>
<td>0.930</td>
<td>0.931</td>
<td>0.952</td>
<td>0.865</td>
<td>n/a</td>
<td></td>
</tr>
<tr>
<td>LPIPS↓</td>
<td>0.052</td>
<td>0.064</td>
<td>0.060</td>
<td>0.054</td>
<td>0.024</td>
<td>0.085</td>
<td>450</td>
<td></td>
</tr>
<tr>
<td>GFlops@500×800</td>
<td>35.80</td>
<td>33.70</td>
<td>33.99</td>
<td>32.89</td>
<td>26.83</td>
<td>7890</td>
<td>7890</td>
<td></td>
</tr>
</tbody>
</table>

Table 1. 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 stared methods correspond to Table 3. UNet-BR-OF 36.41 0.956 0.048 34.27 0.912 0.070 28.61 0.843 0.115 27.71 0.819 0.138

Table 2. 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.

<table>
<thead>
<tr>
<th>Noisy input</th>
<th>Plane 1</th>
<th>Plane 3</th>
<th>Plane 5</th>
<th>Plane 7</th>
<th>Plane 9</th>
<th>Plane 11</th>
<th>Plane 13</th>
<th>Plane 15</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSNR↑</td>
<td>24.52</td>
<td>24.18</td>
<td>23.50</td>
<td>23.40</td>
<td>23.86</td>
<td>25.65</td>
<td>n/a</td>
<td></td>
</tr>
<tr>
<td>SSIM↑</td>
<td>0.798</td>
<td>0.765</td>
<td>0.796</td>
<td>0.791</td>
<td>0.763</td>
<td>0.730</td>
<td>n/a</td>
<td></td>
</tr>
<tr>
<td>LPIPS↓</td>
<td>0.158</td>
<td>0.170</td>
<td>0.168</td>
<td>0.164</td>
<td>0.148</td>
<td>0.110</td>
<td>810</td>
<td></td>
</tr>
<tr>
<td>GFlops@500×800</td>
<td>39.94</td>
<td>36.41</td>
<td>33.50</td>
<td>33.12</td>
<td>30.85</td>
<td>32.00</td>
<td>3230</td>
<td></td>
</tr>
</tbody>
</table>

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 stared methods correspond to Figure 9 in [31], and were provided by the authors.
References


