Tiled Multiplane Images for Practical 3D Photography

Numair Khan  Douglas Lanman  Lei Xiao
Reality Labs Research, Meta

Abstract

The task of synthesizing novel views from a single image has useful applications in virtual reality and mobile computing, and a number of approaches to the problem have been proposed in recent years. A Multiplane Image (MPI) estimates the scene as a stack of RGBA layers, and can model complex appearance effects, anti-alias depth errors and synthesize soft edges better than methods that use textured meshes or layered depth images. And unlike neural radiance fields, an MPI can be efficiently rendered on graphics hardware. However, MPIs are highly redundant and require a large number of depth layers to achieve plausible results. Based on the observation that the depth complexity in local image regions is lower than that over the entire image, we split an MPI into many small, tiled regions, each with only a few depth planes. We call this representation a Tiled Multiplane Image (TMPI). We propose a method for generating a TMPI with adaptive depth planes for single-view 3D photography in the wild. Our synthesized results are comparable to state-of-the-art single-view MPI methods while having lower computational overhead.

1. Introduction

The novel view synthesis (NVS) problem involves using a set of input images to generate views from new and unseen camera positions, allowing three dimensional interaction with photos. This is a long-studied problem, with early work relying on interpolation within dense structured image sets [20, 11, 8]. The specialized rigs commonly required to capture the large number of images restricted these methods to lab settings [46]. However, the potential applications offered by novel view synthesis on modern mobile and VR devices has kindled wide interest in the problem, and encouraged researchers to seek methods that make the technology more accessible. The term 3D photography refers to the use of novel view synthesis in everyday capture settings, often from a single image.

Over the past few years a number of proposed scene representations have leveraged the great strides being made in learning-based techniques to achieve more accurate synthesis with fewer constraints. The most recent of these are neural radiance fields (NeRFs) [26, 50] which represent the scene as multi-layer perceptrons. Their results define the high bar of novel view synthesis. However, this high quality has a significant data and computational cost.
An alternate representation, a multiplane image (MPI), defines the scene as a stack of fronto-parallel RGBA planes that can be warped and rendered into novel viewpoints [53, 26, 9]. An MPI offers the advantage of rendering speed, and suffers from less aliasing than mesh or point-based methods [38, 47, 29]. The latter characteristic is important for applications that require temporal stability. However, an MPI is a highly redundant scene representation: the number of RGBA planes required to capture and reconstruct all the depth variation in a scene can be quite high. Since most scenes have a larger amount of free space than occupied, most of the planes in an MPI are very sparse. This makes them inefficient as a representation [2], and expensive to generate, transmit, and store.

In this paper we propose to address these shortcomings of multiplane images and develop a lightweight solution to the 3D photography problem that can be practically implemented on mobile and VR devices. Some examples of the applications we envision are 3D video conferencing, tele-presence, and VR passthrough [49]. We show how subdividing the image plane into many small MPIs with only a few planes in each, provides a more efficient representation from a computational and memory perspective. However, the naive approach of directly using existing MPI methods with tiles creates boundary artifacts in the novel views. This happens because the commonly used fixed spacing of MPI planes fails to capture the full depth range of a tile when the number of planes is small. Furthermore, it is sensitive to outliers in small regions. We propose a clustering-based approach using learnt confidence weights to predict per-tile MPI planes that better represent local depth features. Our method is lightweight and generates results comparable to the state-of-the-art in MPI-based 3D photography.

In summary, the main contribution of this work are,

1. The demonstration of tiled multiplane images as a practical representation for view synthesis tasks.
2. A learning framework for generating tiled multiplane images from a single RGB input for 3D photography.
3. A novel approach to adaptive MPI plane positioning.

2. Related Work

The progenitors of current 3D photography were the early works on image-based rendering (IBR). These methods usually relied on interpolation within the convex hull of a large set of images to generate novel views. Levoy et al. [20] and Gortler et al. [11] proposed the canonical two-plane parameterization of light fields that renders novel views by quadrilinear interpolation. Gortler’s method also provided an early demonstration of the use of geometric proxies to improve rendering quality. Davis et al.’s [8] work extended interpolation-based view synthesis to unstructured images. The excellent analysis of plenoptic sampling done by Chai et al. [5] proved, however, that for large distances the number of images required for view synthesis by interpolation was impractically high. Consequently, the large majority of recent view synthesis methods have relied on learnt priors to overcome the high sampling requirements.

One of the corollaries of Chai et al.’s analysis was that the sampling requirements for view-interpolation are inversely related to the geometric information of the scene. Thus, many subsequent methods have relied on coarse geometric proxies to improve view synthesis quality [34, 35, 13]. In Mildenhall et al.’s [26] method this proxy takes the form of a multiplane image (MPI), which they use to achieve interpolation-based view synthesis that overcomes Chai et al.’s sampling limits. MPIs were first proposed by Zhou et al. [53] who used them for extrapolating novel views outside the convex hull of the input stereo cameras (Tucker and Snavely [42] observe that an MPI can be considered as an instance of Szeliski and Golland’s [40] earlier “stack of acetates” volumetric model). Mildenhall et al. [26] and Srinivasan et al. [39], respectively, provide a theoretical analysis of the limits of view interpolation and extrapolation using MPIs. Flynn et al. [9] use learnt gradient descent to generate MPIs from multiple views. Attil et al. [1] and Broxton et al. [3] extend the MPI concept to concentric RGBA spheres that can be used for view synthesis in 360 degrees. Wizadwongsa et al. [48] and Li et al. [22] replace the discrete RGBA planes of MPIs with continuous neural surfaces to achieve higher quality results. Tucker and Snavely [42] use a scale-invariant method that allows them to learn strong data priors that can generate MPIs from a single view. Their approach is in a line of recent work dubbed 3D photography that aims for novel views from in-the-wild, single-view images. Li et al. [23], Han et al. [12], and Luvizon et al.’s [25] methods are MPI-based examples of this approach. An MPI, however, is an over-parameterized scene representation. Recent methods [24, 10, 25, 12] have sought to overcome this shortcoming to some extent through a more judicious placement of depth planes. Nonetheless, their high level of redundancy dilates their memory and computational footprint, and limits their wider adoption in mobile and AR/VR applications.

For such use cases, a more efficient approach to novel view synthesis involves depth-based warping [6, 49], often followed by inpainting [18, 19]. Shih et al. [38] propose to guide inpainting in disoccluded regions of the warped view using a layered depth representation. Li et al. [21] apply this general approach to 360-degree input. Wiles et al. [47] use a depth map to generate a point cloud of neural features which can be projected and rendered in novel views using a generative network. Choi et al. [7] estimate a probability volume instead of a single depth map to handle uncertainty in difficult regions. Niklaus et al. [29] use seg-
3. Method

A traditional MPI represents the scene as a set of $N$ fronto-parallel planes in the camera frustum of a reference view $I$. Each plane is associated with an RGB image. While it is possible to place the planes at any depth, they are usually arranged linearly in disparity (inverse depth) [53, 9, 26, 42]. A novel view $I_t$ is rendered by warping the planes into the target camera’s image space via a homography, and compositing them front-to-back using the over [31] operator:

$$I_t = \sum_{i=1}^{N} (\alpha_i c_i \prod_{j=i+1}^{N} (1 - \alpha_j))$$  \hspace{1cm} (1)

where $\alpha_i$ and $c_i$ are the warped alpha and color channels, respectively, of the $i^{th}$ plane.

Both warping and compositing can be done very efficiently on graphics hardware, allowing real-time rendering of novel views [26]. Moreover, the alpha channel at each plane allows MPIs to represent soft edges and anti-alias any errors in the scene reconstruction, leading to fewer perceptually objectionable artifacts than depth-based warping methods. However, the number of planes $N$ required to capture all the depth variation in a scene is usually large, even though most planes are very sparse. We propose to overcome this shortcoming by representing the reference image $I$ as a tiled grid of many small MPIs (Figure 3). Given $I$ and its depth map from a monocular depth estimator, our method predicts the placement of $n \ll N$ depth planes within each tiled region and uses this prediction to generate the RGBA images of the MPI in a single forward pass.

3.1. Tiled Multiplane Image Representation

Our scene representation is based on a set of $m$ tiles, each representing a square sliding block of size $h$ at 2D pixel locations $x_1, x_2, ..., x_m$ in the source image $I$. The locations $x_i$ lie on a regular grid with spacing determined by some stride $r$. Each tile consists of $n$ front-parallel RGBA planes, the depth placement of which is not fixed but varies across tiles. We let $\alpha_j^i$, $c_j^i$ and $d_j^i$ denote, respectively, the alpha channel, the color channel, and depth of the $i^{th}$ plane in the $j^{th}$ tile. Then the tiled multiplane image (TMPI) representation $\Gamma(I)$ of the image is defined as:

$$\Gamma(I) = \{(\alpha_j^i, c_j^i, d_j^i, x_j)\}$$  \hspace{1cm} (2)

for all $i = 1, 2, ..., n$ and $j = 1, 2, ..., m$. Further, we define an ordering on this set of 4-tuplets as,

$$(\Gamma(I), \preceq) = \{(\alpha, c, d, x)_{k=1,...,mn} \mid d_k \leq d_{k+1}\}$$  \hspace{1cm} (3)
As with traditional MPIs, novel views are rendered in a differentiable manner by warping all planes into the target camera’s image space. However, as the depth of the planes varies across the tiles and, hence, across the pixels of $I_t$, a planar inverse warp on the target image plane cannot be directly computed via a homography. Instead, each plane must be warped in tile space via a homography computed using a shifted intrinsic matrix, and all the warped planes composited sequentially in the target view at their respective tile locations (Algorithm 1). While this makes the rendering of tiled multiplane images less efficient and somewhat less elegant than MPIs (Equation 1), this is only true during training when differentiability is required. At inference, the tiles can be rendered as textured quads using hardware-accelerated rasterization making this stage of the pipeline as efficient as traditional MPIs and with lower texture memory requirements. Thus, their compact form makes TMPIs well-suited for rendering over networks, or on mobile devices and VR headsets. The blending of many MPIs locally is related to Mildenhall et al.’s [26] light field fusion. However, their method renders each MPI separately before blending the results with scalar weights. Our method operates at a much finer scale, and composites the planes of all MPIs together to synthesize a novel view.

Algorithm 1: Differentiable view synthesis using TMPIs. Angled brackets denote pixel indexing.

\[ \text{RenderTMPI} \left( \left( \Gamma(I), \leq \right), R, t, K \right) \]

**Input**: $I$: ordered TMPI planes
R: relative rotation of novel view
t: relative translation
K: camera intrinsics

**Output**: Novel view $I_t \in \mathbb{R}^{3 \times H \times W}$
$I_t \leftarrow 0^{3 \times H \times W}$; \( T_t \leftarrow 1^{H \times W} \);

**foreach** \((\alpha, c, d, x) \in \left( \Gamma(I), \leq \right)\) do
\[
\hat{K} \leftarrow K - \left[ \begin{array}{c} 1 \\ x \end{array} \right] ;
\]

**foreach** \(u \in [1, \ldots, s] \times [1, \ldots, s]\) do
\[
\begin{align*}
\hat{K} & \leftarrow K - \left[ \begin{array}{c} 1 \\ x \end{array} \right] ; \\
\mathbf{u} & \leftarrow \hat{K} (R - t n^T / d) \hat{K}^{-1} \left[ \begin{array}{c} u \\ 1 \end{array} \right] ; \\
\mathbf{w} & \leftarrow \alpha \mathbf{u}; \\
I_t(u + x) & \leftarrow I_t(u + x) + w c(u); \\
T_t(u + x) & \leftarrow T_t(u + x) (1 - \alpha(u)); \\
end \end{align*}
\]

**return** $I_t$

3.2. Single-View 3D Photography

We now describe our approach to generating tiled multiplane images from a single RGB input (Figure 2). Broadly, our method splits the image plane into a regular tiled grid of learnt confidence and outlier-corrected depth. For each tile, the placement of \(n\) fronto-parallel depth planes is determined by clustering pixel depth values weighted by the predicted confidence estimates. This latter step is motivated by the fact that with a small depth plane budget \(n\), the commonly used equal spacing in disparity [53, 9, 26, 42] is wasteful. Thus, the goal is to predict the planes that optimally represent all depth variation within a tile. Using the predicted planes, a fully convolutional network generates the \(n\) RGBA images that constitute a per-tile MPI. Unlike Han et al.’s [12] adaptive plane method, we generate the RGBA images in a single forward pass. The resulting TMPI is rendered as a set of textured quads using the rasterization pipeline of graphics hardware.

In more detail, given the source image $I$, we first obtain a depth map $\mathcal{D}$ for it using a monocular depth estimator. A two-headed U-Net $\Theta(\cdot)$ then predicts a confidence map $\mathcal{C}$ along with denoised depth $\mathcal{D}$. The goal is to learn a representation that ameliorates the sensitivity of the subsequent $k$-means clustering step to outliers. The joint prediction of confidence and depth is similar to the depth-routing of Weder et al. [45] and the aleatoric uncertainty estimation of [15]. The predicted depth and confidence, and the original color image are then unfolded into a set of $m$ square sliding blocks of size $h$ and stride $r$: \(\{(D^i, C^i), \forall i \in 1, \ldots, m\}\).

Running $\Theta(\cdot)$ on $I$ and $\mathcal{D}$ rather than individual tiles allows it to consider non-local features and avoid undesirable tiling artifacts. Additionally, setting $r < h$ in the unfolding step allows neighboring tiles to overlap. This prevents gaps along tile boundaries and also regularizes per-tile operations across neighbors. However, it also increases the total number of tiles and, thus, the computational requirements. We empirically determine a stride value of $r = h - h/8$ for a good balance between quality and computational efficiency.

Figure 3: (a) A traditional MPI uses five planes per pixel to represent this toy scene, even though no region has more than two overlapping objects. (b) A TMPI exploits the low local depth complexity and has only two planes per pixel.
**Per-Tile Planes Prediction:** Next, we predict the $n$ depth planes $\{z^i_j\}_{j=1,...,n}$ that optimally represent the features of the $i$-th tile. The common approach of spacing the planes linearly in disparity grows inaccurate as $n$ becomes small. Lu and Zhou's [25] approach places the planes at depth discontinuities identified via the histogram of depth values. However, their approach is sensitive to parameter settings and fails for smooth surfaces which have no discontinuities. A learning-based approach is adopted by Han et al. [12] and Li et al. [23]. The former use multi-headed self-attention to adjust a linear placement. While capable of modeling inter-plane interactions, their method is computationally expensive (Table 2). The latter uses a CNN to directly predict the planes. However, we found that without strong regularization a direct approach lacks topological order and has a strong bias towards a fixed placement. An adversarial loss helps improve this but makes the training more unstable.

We observe that as depth is known, plane positioning can be posed as a simple clustering problem. Thus, we predict $\{z^i_j\}$ using $k$-means clustering on the depth in each tile $D^i$.

Standard $k$-means is sensitive to outliers and can generate significantly different plane predictions across neighboring tiles causing artifacts in novel views. Advantageously, along with the $n$ depth planes, clustering also assigns a label to each input pixel, thereby generating a label map that represents discretized depth. Furthermore, the cluster centers of $k$-means are differentiable with respect to the input samples. Thus, we address the outlier problem by training $\Theta(\cdot)$ to filter the input through a self-supervised reconstruction loss on the discretized depth map generated by a weighted $k$-means. In weighted $k$-means, the cluster centers are updated each iteration using the confidence-weighted mean of the constituent samples. Since we do not directly supervise the depth output of $\Theta(\cdot)$, the network can go beyond outlier-filtering to learn any modifications that improves the discrete reconstruction, and consequently optimizes the placement of the $n$ depth planes within each tile.

**Per-Tile MPI Generation:** Given $\{z^i_j\}$ and the discrete depth map, we estimate a preliminary MPI per tile by peeling RGB layers from $I^i$ using the discrete labels as an alpha mask. The masked RGB regions of each tile are inpainted by upsampling valid values from a Gaussian pyramid. A second network $\Psi(\cdot)$ then refines these estimates to generate the final $n$ RGBA images of the MPI for each tile. Following Zhou et al. [53] and Tucker and Snavely [42], we represent the RGBA output as a pixel-wise blend of the input image $I^i$ and a learnt background. However, unlike these works we predict a background image $B^i_j$ per plane:

$$W_j^i = \prod_{k>j}(1 - \alpha_k^i), \quad (4)$$

$$I_j^i = W_j^i I^i + (1 - W_j^i) B_j^i \quad \text{(5)}$$

Where $\alpha_k^i$ is the predicted alpha value for each plane.

---

**Table 1:** Quantitative evaluation of view synthesis results on the *Spaces* and *Tanks and Temples* multi-view datasets. SVMPI, MINE and AdaMPI use 32 MPI depth planes; VMPI uses 8; our approach uses 4 planes per image tile.

<table>
<thead>
<tr>
<th>Method</th>
<th>Spaces Dataset</th>
<th>Tanks &amp; Temples Dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PSNR ↑</td>
<td>SSIM ↑</td>
</tr>
<tr>
<td>SVMPI [42]</td>
<td>25.42</td>
<td>0.748</td>
</tr>
<tr>
<td>VMPI [23]</td>
<td>22.37</td>
<td>0.636</td>
</tr>
<tr>
<td>MINE [22]</td>
<td>24.02</td>
<td>0.702</td>
</tr>
<tr>
<td>AdaMPI [12]</td>
<td>26.17</td>
<td>0.703</td>
</tr>
<tr>
<td>Ours</td>
<td>24.93</td>
<td>0.750</td>
</tr>
</tbody>
</table>

---

4. **Training Procedure**

We train $\Theta(\cdot)$ in a self-supervised manner by minimizing the L1 loss between the input depth $Z$ and the folded discrete depth maps produced per tile by weighted $k$-means clustering. Then, we freeze $\Theta(\cdot)$ and train the MPI generation network $\Psi(\cdot)$ on a novel view synthesis task by following Han et al.'s [12] warp-back strategy to generate pseudo ground truth multi-view training data. This involves warping single-view images into a target camera using monocular depth, and inpainting disocclusion holes with a specially trained network. The view-synthesis training objective is a combination of VGG, structural similarity [44], and L1 losses on the synthesized color image $I_{\tau}$, weighed as 0.1, 0.25, and 1.0 respectively. For both networks, we use the 111K images of the COCO dataset [4] and the monocular depth method of Ranftl et al. [32].

5. **Experiments**

5.1. **Implementation**

Our method is implemented in PyTorch and trained on eight Nvidia Tesla V100 GPUs. We use 256×384 images and accumulate gradients across eight mini-batches of 16 samples each. For both networks $\Theta(\cdot)$ and $\Psi(\cdot)$, we use the Adam optimizer with a learning rate of $1 \times 10^{-3}$ and a cosine annealing schedule with restarts every 200 epochs. A vectorized implementation of the differentiable TMPI renderer (Algorithm 1) runs at ~100ms / mini-batch allowing efficient parallel computation.
5.2. Baselines

We compare our approach to four state-of-the-art single-view 3D photography methods based on multiplane images: Tucker et al. [42] (SVMPI), Li et al. [23] (VMPI), Li et al. [22] (MINE) and Han et al. [12] (AdaMPI). Like us, VMPI and AdaMPI use a monocular estimator to recover depth as the first step of their pipeline. The input to all methods is the same, however — a single unconstrained RGB image — and so we evaluate them as end-to-end 3D photography approaches. Nonetheless, we do note the additional depth estimation step when evaluating computational and memory performance (Table 2). We use $N = 32$ planes for all baselines except VMPI, which is designed for $N = 8$.

We do not compare to single-view methods based on neural radiance fields [51] or the recent work of Nicklaus et al. [29] and Shih et al. [38] as these are too computationally intensive for our intended use cases on mobile and VR devices. While recent work on NeRFs has demonstrated impressive rendering speeds, training remains expensive, and further, requires a large number of input views and a static scene. Jampani et al.’s [14] method, though not MPI-based, is related. But the authors have not released their code.

5.3. Testing Datasets

We test all methods on the Spaces [9], and Tanks and Temples [17] datasets. Spaces consists of 100 indoor and outdoor scenes captured using a purpose-built, 16-camera rig. For Tanks and Temples we use the Intermediate split which has uniformly sampled frames from eight high-resolution videos of more challenging outdoor environments. We compute camera poses and depth maps for all scenes using COLMAP [37, 36]. The depth maps are required to resolve the scale ambiguity of monocular depth estimation using Ranftl et al.’s [32] DPT. Our approach has much fewer planes per tile can be attributed to the adaptive placement of depth planes in each tile which allows it to effectively cover a larger depth range than the monolithic MPIs of the baselines, Memory and computational performance is evaluated in Table 2. VMPI, AdaMPI and our method have the additional overhead of monocular depth estimation using Ranftl et al.’s [32] DPT. Our approach has lower runtime, peak memory and space requirements than AdaMPI while achieving similar quality results.

5.4. Evaluation Metrics

We evaluate the rendered views quantitatively on four metrics: Peak-Signal-to-Noise Ratio (PSNR), Structural Similarity [44] (SSIM), Learned Perceptual Image Patch Similarity [52] (LPIPS) with a VGG-16 backbone, and the mean absolute error (L1).

5.5. Results

Table 1 presents quantitative evaluation of all methods on the two test sets. Qualitative results are shown in Figure 4 for Tanks and Temples and Figure 9 for Spaces. Our approach uses $n = 4$ depth planes with a tile size of $h = 64$. The competitive performance of our method despite having much fewer planes per tile can be attributed to the adaptive placement of depth planes in each tile which allows it to effectively cover a larger depth range than the monolithic MPIs of the baselines. Memory and computational performance is evaluated in Table 2. VMPI, AdaMPI and our method have the additional overhead of monocular depth estimation using Ranftl et al.’s [32] DPT. Our approach has lower runtime, peak memory and space requirements than AdaMPI while achieving similar quality results.

We evaluate our plane placement strategy in Table 3. We measure the reconstruction quality of the discretized depth
Figure 4: Comparing the novel view synthesis results of the baseline methods and our approach on the Tanks and Temples dataset. Our results are better than SVMPI, VMPI and MINE, and comparable to AdaMPI while using far fewer depth planes.

Figure 5: View synthesis results of our method on the HD (1080 × 1920) Davis [30] dataset. Original views are inset.
Figure 6: Top to bottom: Input image (Davis dataset); inverse monocular depth from DPT [32]; naive discretization by spacing four planes linearly in the inverse-depth range of each tile; our weighted clustering-based discretization with four planes. Our representation shows finer variation on receding surfaces and suffers fewer tiling artifacts.

map defined by the planes in each tile on a monocular depth input that is perturbed by a small amount of Gaussian noise ($\lambda = 0, \sigma^2 = 1 \times 10^{-3}$). We compare our approach to naive linear spacing of depth planes in inverse disparity space, and to vanilla unweighted $k$-means. Our method is robust to outliers and yields a much better reconstruction. Figure 6 provides qualitative comparison with linear spacing on samples from the Davis dataset [30].

Figure 7 evaluates the effect of tile size and number of depth planes on view synthesis quality. In general, the results uphold the intuition that small tiles and a large number of planes improve quality. The trend, however, is less clear when the number of planes $> 8$, indicating that the model may be unable to handle redundant geometry effectively.

6. Limitations

As previously observed, a tiled multiplane image cannot exploit the elegant warping and compositing equations of a traditional MPI for differentiable rendering during training. Furthermore, in some cases our method fails to reconstruct thin features consistently across tiles (Figure 8).

7. Conclusion

We present a method for estimating tiled multiplane images from a single RGB input for 3D photography. This includes a novel approach to adaptively spacing a small number of depth planes within an MPI tile to better represent local features. Our method is lightweight, and points a path to realizing novel view synthesis on mobile and VR devices.

References


Figure 9: Comparing the novel view synthesis results of the baseline methods and our approach on the Spaces dataset.


[28] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a mult-