SLIDE: Single Image 3D Photography with Soft Layering and Depth-aware Inpainting

Varun Jampani*, Huiwen Chang*, Kyle Sargent, Abhishek Kar, Richard Tucker, Michael Krainin,
Dominik Kaeser, William T. Freeman, David Salesin, Brian Curless, Ce Liu

Google Research

Figure 1: **Appearance Details with SLIDE.** View synthesis results show better preservation of hair structures in SLIDE (Ours) compared to that of 3D-Photo [31]. We also show the novel view (Ours-No-BG) where the background (BG) layer is greyed-out to showcase our soft layering. See the supplementary video for a better illustration of view synthesis results.

Abstract

Single image 3D photography enables viewers to view a still image from novel viewpoints. Recent approaches combine monocular depth networks with inpainting networks to achieve compelling results. A drawback of these techniques is the use of hard depth layering, making them unable to model intricate appearance details such as thin hair-like structures. We present SLIDE, a modular and unified system for single image 3D photography that uses a simple yet effective soft layering strategy to better preserve appearance details in novel views. In addition, we propose a novel depth-aware training strategy for our inpainting module, better suited for the 3D photography task. The resulting SLIDE approach is modular, enabling the use of other components such as segmentation and matting for improved layering. At the same time, SLIDE uses an efficient layered depth formulation that only requires a single forward pass through the component networks to produce high quality 3D photos. Extensive experimental analysis on three view-synthesis datasets, in combination with user studies on in-the-wild image collections, demonstrate superior performance of our technique in comparison to existing strong baselines while being conceptually much simpler. Project page: https://varunjampani.github.io/slide

1. Introduction

Still images remain a popular choice for capturing, storing, and sharing visual memories despite the advances in richer capturing technologies such as depth and video sensing. Recent advances [34, 39, 26, 31, 16, 17] show how such 2D images can be "brought to life" just by interactively changing the camera viewpoint, even without scene movement, thereby creating a more engaging 3D viewing experience. Following recent works, we use the term 'Single image 3D photography' to describe the process of converting a 2D image into a 3D viewing experience. Single image 3D photography is quite challenging, as it requires estimating scene geometry from a single image along with inferring the dissoccluded scene content when moving the camera around. Recent state-of-the-art techniques for this problem can be broadly classified into two approaches - *modular systems* [26, 31] and *monolithic networks* [34, 39].

Modular systems [31, 26, 16, 17] leverage state-of-the-art 2D networks such as single-image depth estimation, 2D inpainting, and instance segmentation. Given recent advances in monocular depth estimation [29, 21, 20] and inpainting [40, 41] fueled by deep learning on large-scale 2D datasets, these modular approaches have been shown to work remarkably well on in-the-wild images. A key component of these modular approaches is decomposing the scene into a set of layers based on depth discontinuities. The scene is usually decomposed into a set of layers with *hard* discontinuities and thus can not model soft appearance effects such

as matting. See Figure 1 (right) for an example novel view synthesis result from 3D-Photo [31], which is a state-of-the-art single image 3D photography system.

In contrast, monolithic approaches [34, 39] attempt to learn end-to-end trainable networks using view synthesis losses on multi-view image datasets. These networks usually take a single image as input and produce a 3D representation of a scene, such as point clouds [39] or multi-plane images [34], from which one could interactively render the scene from different camera viewpoints. Since these networks usually decompose the scene into a set of soft 3D layers [34] or directly generate 3D structures [39], they can model appearance effects such as matting. Despite being elegant, these networks usually perform poorly while inferring disoccluded content and have difficulty generalizing to scenes out of the training distribution, a considerable limitation given the difficulty in obtaining multi-view datasets on a wide range of scene types.

In this work, we propose a new 3D photography approach that uses soft layering and depth-aware inpainting. We refer to our approach as 'SLIDE' (Soft-Layering and Inpainting that is Depth-aware). Our key technique is a simple yet effective soft layering scheme that can incorporate intricate appearance effects. See Figure 1 for an example view synthesis result of SLIDE (Ours), where thin hair structures are preserved in novel views. In addition, we propose an RGBD inpainting network that is trained in a novel depth-aware fashion resulting in higher quality view synthesis. The resulting SLIDE framework is modular, and allows easy incorporation of state-of-the-art components such as depth and segmentation networks. SLIDE uses a simple two-layer decomposition of the scene and requires only a single forward pass through different components. This is in contrast to the state-of-the-art approaches [31, 26], which are modular and require several passes through some components networks. Moreover, all of the components in the SLIDE framework are differentiable and can be implemented using standard GPU layers in a deep learning toolbox, resulting in a unified system. This also brings our SLIDE framework closer to single network approaches. We make the following contributions in this work:

- We propose a simple yet effective soft layering formulation that enables synthesizing intricate appearance details such as thin hair-like structures in novel views.

- We propose a novel depth-aware technique for training an inpainting network for the 3D photography task.

- The resulting SLIDE framework is both modular and unified with favorable properties such as only requiring a single forward computation with favorable runtime.

- Extensive experiments on four different datasets demonstrate the superior performance of SLIDE in terms of both quantitative metrics as well as from user studies.

2. Related Work

Classical optimization methods have been applied to the view synthesis task [13, 18, 27], but the most recent approaches are learning-based. Some works [8, 15] have predicted novel views independently, but to achieve consistency between output views, it is preferable to predict a scene representation from which many output views can be generated. Such representations include point clouds [39, 24], meshes [31], layered representations such as layered depth images [30, 35] and multi-plane images (MPIs) [44, 7, 32], and implicit representations such as NeRF [25, 23, 5]. Much research in view synthesis has centered on the task of interpolation between multiple images, but most relevant to our work are methods focusing on the very challenging task of extrapolation from a single image.

Single Network Approaches. For narrow baselines, Srinivasan *et al*. predict a 4D lightfield directly [33], while Li and Kalantari [19] represent a lightfield as a blend of two Variable-depth MPIs. For larger baselines, Single-view MPI [34] applies the MPI representation to the single image case, and SynSin [39] uses point clouds and applies a neural rendering stage, which enables it to generate content outside the original camera frustum. These learning-based methods are trained end-to-end, with held-out views from novel viewpoints being used for supervision via a reconstruction loss. Training data can be obtained from light-field cameras [19, 33] or multi-camera rigs [7], or derived from photo collections [24] or videos of static scenes [44]. A key drawback of these approaches is their poor generalization to in-the-wild images.

Depth-based 3D Photography. Another approach to single image 3D photography is to build a system combining depth prediction and inpainting modules with a 3D renderer. For depth estimation from a single image, a variety of learning-based methods [6, 21, 10] exist. The MiDaS system [29] achieves excellent results by training on frames from 3D movies [29] as well as other depth datasets. Non-learning-based approaches for inpainting apply patch matching and blending [2, 14] or diffusion [4, 38], but since arbitrary amounts of training data can be generated simply by obscuring random sections of input images, this task is a natural target for learning-based approaches. Recent methods propose augmenting convolutional networks with contextual attention mechanisms such as DeepFill's gated convolutions [41, 42] or a coherent semantic attention layer [22], and applying patch-based discriminators for GAN-based training as well as reconstruction losses. Another recent work, HiFill [40] takes a residual-based approach to inpaint even very high resolution images.

In the context of 3D Photography, inpainting will typically operate on a more complex representation than a simple image, and systems may need to inpaint depth as well as texture. The method of Shih *et al*. [31] introduces
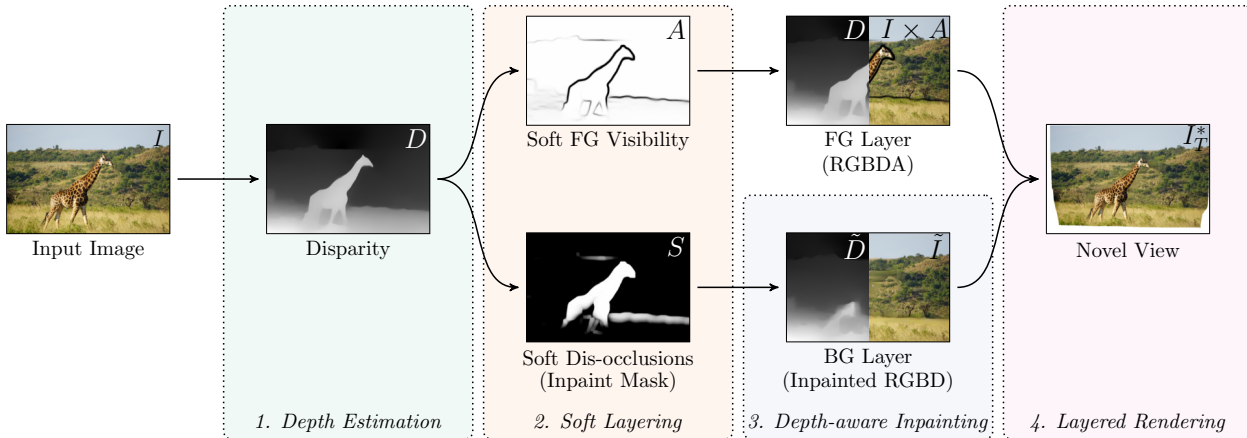
Figure 2: **SLIDE Overview.** SLIDE is a modular and unified framework for 3D photography and consists of the four main components of depth estimation, soft layering, depth-aware RGBD inpainting and layered rendering. In addition, one can optionally use foreground alpha mattes (not shown in this figure) to improve the layering.

an extension of the layered depth image format. The system performs multiple inpainting steps in which edges and depth, as well as images, are inpainted within different image patches. The system of Niklaus *et al.* [26] performs inpainting on rendered novel images and projects the inpainted content back into a point cloud to augment its representation. This latter system also adds an additional network to refine the estimated depth, and incorporates instance segmentation to ensure that people and other important objects in the scene do not straddle depth boundaries. These systems are somewhat complex, requiring multiple passes of the same network (e.g., inpainting). As mentioned earlier, another key drawback of these approaches is that the layering is hard and can not incorporate intricate appearance effects in layers. Our approach follows the depth-plus-inpainting paradigm but operates on a simple yet effective two-layer representation that enables the incorporation of soft appearance effects. In addition, due to our simple two-layer soft formulation, we only need a single forward pass through different component networks; thus, it can be considered as a single unified network while being modular.

3. Methods

SLIDE Overview. As illustrated in Figure 2, our 3D photography approach, SLIDE, has four main components: 1. monocular depth estimation, 2. soft layering, 3. depth-aware RGBD inpainting, and 4. layered rendering. From a given still image $I \in \mathbb{R}^{n \times 3}$ with n pixels, we first estimate depth $D \in \mathbb{R}^n$. We then decompose the scene into two layers via our soft-layering formulation where we estimate foreground (FG) pixel visibility $A \in \mathbb{R}^n$ and inpainting mask $S \in \mathbb{R}^n$ in a soft manner. Using these, we construct the foreground RGBDA layer with the input image I, its corresponding disparity D and the pixel visibility map A; and the background RGBD layer with the inpainted

RGB image $\tilde{I}$ and the inpainted disparity map $\tilde{D}$. We then construct triangle meshes from the two disparity maps, textured with I and A for the foreground and $\tilde{I}$ for the background, render each into a target viewpoint, and composite the foreground rendering over the background rendering. In addition, we can optionally use foreground alpha mattes to improve layering as our soft layering enables easy incorporation of alpha-mattes.

3.1. Monocular Depth Estimation

Given an RGB image $I \in \mathbb{R}^{n \times 3}$ with n pixels, we first estimate a disparity (inverse depth) map $D \in \mathbb{R}^{n \times 1}$ using a CNN. We use the publicly released MiDaS v2 [29] network Φ_D for monocular inverse depth prediction. Specifically, the MiDaS model is trained on a large and diverse set of datasets to achieve zero-shot cross dataset transfer. It proposes a principled dataset mixing strategy and a robust scale and shift invariant loss function that results in predicted disparity maps up to an unknown scale and shift factor. The final output of Φ_D is a normalized disparity map $D \in [0,1]^n$, which is then used in the subsequent parts of the SLIDE pipeline. To reduce missing foreground pixels and noise in layering (Section 3.2), we do slight Gaussian blur and maxpool the disparity map. One could use any other monocular depth network in our framework. We choose MiDaS for its good generalization across different types of images.

3.2. Soft Layering

A key technical contribution in SLIDE is estimating layers in a soft fashion so that we can model partial visibility effects across layers. As illustrated in Figure 2, layering also connects the depth and inpainting networks making it a crucial component of SLIDE. Our soft layering has two main components: 1. estimating soft pixel visibility of the foreground layer, and 2. estimating a soft disocclusion map that is used for background RGBD inpainting.

Without FG Visibility   *With* FG Visibility

Figure 3: **Foreground Pixel Visibility.** Rendering an RGBD layer without pixel visibility (left) leads to stretchy triangles, whereas rendering with pixel visibilities (right) enables seeing through to the background (represented with black pixels here).

Image with Disparity   Soft Disocclusions   Soft Occlusions

Figure 4: **Soft Disocclusions and Occlusions.** At each point (x, y) in an image, we compare disparity differences across horizontal and vertical scan lines to compute soft disocclusion and occlusion maps.

Soft FG Pixel Visibility. We estimate visibility at each image pixel, which enables us to see through to the background layer when rendering novel-view images. Figure 3 (left) shows a single RGBD layer, represented by the given RGB image and the corresponding estimated disparity, rendered as a textured triangle mesh into a new viewpoint. Stretching artifacts appear at depth discontinuities. To address these artifacts, we construct a visibility map A that has lower values (higher transparency) at depth discontinuities – lower visibility in proportion to changes in disparity – later allowing us to see through these discontinuities to the (inpainted) background layer. More formally, given the estimated disparity map D for a given image I, we compute the FG pixel visibility map $A \in [0, 1]^n$ as:

$$A = e^{-\beta||\nabla D||^2}, \qquad (1)$$

where ∇ is the Sobel gradient operator and $\beta \in \mathbb{R}$ is a scalar parameter. Thus the pixel visibility varies inversely with disparity gradient magnitude. Low FG visibility ($A \approx 0$) corresponds to high FG pixel transparency. Figure 3 (right) shows a novel-view rendering with the pixel visibility map A multiplied against the original rendering; black regions indicate areas that are now transparent in the foreground layer. Note that modelling this FG visibility in a soft manner allows SLIDE to easily incorporate segmentation based soft alpha mattes into layering, as we discuss in Section 3.3.

Soft Disocclusions. In addition to foreground visiblity, we need to construct a mask to guide inpainting in the background layer. Intuitively, we need to inpaint the pixels that

have potential to be dissoccluded when the camera moves around. The relationship between (dis-)occlusion and disparity is well known [3, 37, 36], and we make use of this relationship to compute soft disocclusions from the estimated disparities. The disparity-occlusion relationship in these prior works is derived in the stereo image setting, where we have metric disparity, and the occlusions are defined with respect to a second camera. In our case, we only have relative depth (disparity), but we can still assume some maximum camera movement and introduce a scalar parameter that can scale the disparities accordingly. Consider the background region at pixel location (x, y) behind the giraffe's head in Figure 4 (left). This background region has the potential to be dissoccluded by the foreground if there exists a neighborhood pixel (x_i, y_j) whose disparity difference with respect to the foreground pixel at (x, y) is greater than the distance between those pixels' locations. More formally, the background pixel at (x, y) will be dissoccluded if

$$\exists_{(x_i, y_j)} \left(D_{(x,y)} - D_{(x_i, y_j)} > \rho K_{(x_i, y_j)} \right), \qquad (2)$$

where ρ is a scalar parameter that scales the disparity difference. $K_{(x_i, y_j)} = \sqrt{(x_i - x)^2 + (y_j - y)^2}$ is the distance between (x_i, y_j) to the center pixel location (x, y). In simpler terms, a background pixel is more likely to be dissoccluded if the foreground disparity at that point is higher compared to that of surrounding regions. For our soft-layering formulation, we convert the above step function into a soft version resulting in a soft disocclusion map $S \in [0, 1]^n$:

$$S_{(x,y)} = \tanh\Big(\gamma\max_{(x_i, y_j)} \big(D(x, y) - D(x_i, y_j)$$
$$-\rho K_{(x_i, y_j)} \big) \Big), \qquad (3)$$

where γ is another scalar parameter that controls the steepness of the tanh activation. In addition, we apply a ReLU activation on top of tanh to make S positive. Computing S with the above equation requires computing pairwise disparity differences between all the pixel pairs in an image. Since this is computationally expensive, we constrain the disparity difference computation to a fixed neighborhood (m pixels) around each pixel ($(x_i, y_j) \in \mathcal{N}_{(x,y)}$, where $\mathcal{N}$ is the m pixel neighborhood of (x, y)). This is still computationally expensive for reasonable values of m (> 30). So, we constrain our disparity difference computation along horizontal and vertical scan lines as illustrated with red lines in Figure 4 (left). We implement pairwise disparity differences and also pixel distances along horizontal and vertical neighborhoods with efficient convolution operations on disparity and pixel coordinate maps. This results in an effective feed-forward computation of dissocclusion maps using standard deep learning layers. For efficiency, we can also compute the disocclusion map on a downsampled disparity map and then upsample the resulting map to the desired resolution. Figure 4 (middle) shows the soft dissocclusion map
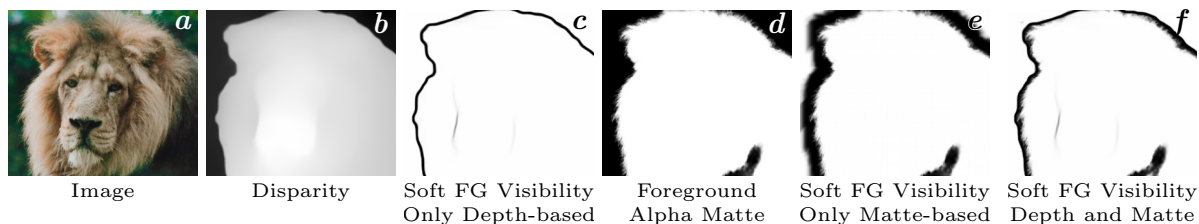
Figure 5: **Layering With Alpha Mattes.** Depth-based FG visibility (c) can not capture hair-like structures. Computing FG visibility (e) based on FG alpha matte (d) and then incorporating it into visibility can capture fine details (f).

that is estimated using this technique. In a similar fashion, we obtain a soft *occlusion* map $\hat{S} \in [0,1]^n$ as shown in Figure 4 (right) by replacing '>' with '<' in Eqn. 2 and modifying Eqn. 3 accordingly. We make use of the occlusion and disocclusion maps in inpainting training (Section 3.4).

3.3. Improved Layering with Segmentation

Consider the input image shown in Figure 5(a), which has thin hair structures. The soft FG visibilities (Eqn. 1) purely based on depth discontinuities will not preserve such thin structures in novel views. For this image, we can see that these fine structures are not captured in the disparity map and are therefore also missed in the visibility map (Figure 5 (b) and (c), resp.). To address this shortcoming, we incorporate FG alpha mattes – computed using state-of-the-art segmentation and matting techniques – into our layering. Our soft layering can naturally incorporate soft mattes.

We first compute FG segmentation using the U2Net saliency network [28], which we then pass to a matting network [9] to obtain FG alpha matte $M \in [0,1]^n$ as shown in Figure 5 (d). Note that we can not directly use these alpha mattes as visibility maps, as we want visibility to be low (close to zero) *only* around the FG object boundaries. So, we dilate (max-pool) the alpha matte (denoted as $\bar{M}$) and then subtract the original alpha matte from it. Figure 5 (e) shows $1 - (\bar{M} - M)$. The resulting matte-based FG visibility map will only have low visibility around FG alpha matte. We then compute the depth-matte based FG visibility map $A' \in [0,1]^n$ as: $A' = A * (1 - (\bar{M} - M)(1 - \hat{S}))$, where A denotes the depth-based visibility map (Eqn. 1, Figure 5 (c)) and $\hat{S}$ denotes the occlusion map, with example shown in Figure 4 (right). $(1 - \hat{S})$ term reduces the leakage of matte-based visibility map onto too much background and the multiplication with depth-based visibility A ensures that the final visibility map also accounts for depth discontinuities. Figure 5 (f) shows the depth-matte based FG visibility map A', which captures fine hair-like structures while also respecting the depth discontinuities.

3.4. Depth-Aware RGBD Inpainting

To avoid exposing black background pixels when the camera moves, as shown in Figure 3, we inpaint the disoccluded regions S and incorporate the result into our background layer. Inpainting such disocclusions is different from traditional inpainting problems because the model

needs to learn to neglect the regions in front of each to-be-inpainted pixel. In Figure 6, we show sample results of two state-of-the-art image inpainting methods [42, 40] to inpaint the disoccluded regions. While they synthesize good textures and are even capable of completing the basketball and the dog's head, this is actually undesirable in our pipeline, as we want to inpaint the BG, not the FG. In addition, we perform RGBD inpainting which is in contrast to the existing RGB inpainting networks.

One of the key challenges in training our inpainting network using disocclusion masks (see Figure 4 (middle) for an example mask) is that we do not have ground-truth background RGB or depth for these regions in single image datasets. To overcome this, we instead make use of occlusion masks (see Figure 4 (right)) that surround the objects as inpaint masks during training. Since we have GT background RGB along with estimated background depth within occlusion masks, we can directly use these masks along with the original image as GT for training.

The intuition behind inpainting the occlusion mask is to pretend that the FG is larger than its actual size along its silhouette. We find training on these masks helps the model learn to borrow from the regions with larger depth values. In other words, such training with occlusion masks makes inpainting depth-aware. Training with this type of mask only, however, is insufficient as the model has not learned to inpaint thin objects or perform regular inpainting. We address this by randomly adding traditional stroke-shape inpainting masks used in standard inpainting training following Deepfillv2 [42], which enables the model to learn to inpaint thin or small objects. So our dataset consists of two types of inpainting masks: occlusion masks and random strokes. In this way, any single image dataset can be adapted to be used in training our inpainting model without requiring any annotations. We show examples from our custom training dataset in the supplement. See Figure 6 for a sample inpainting result with more in the supplementary material.

We employ a patch-based discriminator D to discriminate between real and generated results, and apply an adversarial loss to the inpainting network, as in Deepfillv2 [42]. So the objective loss for the inpainting network is a weighted sum of the reconstruction loss – the L_1 distance between inpainted results and ground truth – and the hinge adversarial loss. More details about network training and

| Input | Disparity | Inpaint Mask | HiFill | DeepFillv2 | Ours |

Figure 6: **Depth-aware Inpainting.** Inpainting techniques (HiFill [40] and DeepFill [42]) borrow information from both FG and BG. Our depth-aware inpainting mostly borrows information from the BG making it more suitable for 3D photography.

architectures are discussed in the supplementary material.

It is worth contrasting our inpainting approach with that of 3D-Photo inpainting [31], which is also depth-aware and its dataset is annotation-free. One big difference is that our inpainting is global while 3D-Photo inpaints on local patches based on depth edges. Due to this, our inference only requires a single-pass and is relatively fast, while 3D photos requires multi-stage processing and iterative flood-fill like algorithms to generate the inpainting masks per patch, which is relatively time-consuming.

3.5. Layered Rendering

Given the foreground and background images and disparity maps, we can now render each into a novel view and composite them together. The soft layering stage produces a foreground layer comprised of the input image I, visibility map A, and disparity D. We back-project the disparity map in the standard way to recover a 3D point per pixel and connect points that neighbor each other on the 2D pixel grid to construct a triangle mesh. We then texture this mesh with I and A and render it into the novel viewpoint; A is resampled, but not used for compositing during rendering at this stage. The novel viewpoint is given by a rigid transformation T from the canonical viewpoint, and the result of this rendering step is a new foreground RGB image I_T and visibility A_T. The output of the inpainting stage is background image $\tilde{I}$ and disparity $\tilde{D}$. We similarly construct a triangle mesh from $\tilde{D}$, texture it with $\tilde{I}$, and project into the novel view to generate new background image $\tilde{I}_T$. Finally, we composite foreground over background to obtain the final novel-view image I_T^*:

$$I_T^* = A_T I_T + (1 - A_T)\tilde{I}_T. \tag{4}$$

We use a TensorFlow differentiable renderer [12] to generate I_T and $\tilde{I}_T$ to enable a unified framework.

4. Experiments

We quantitatively evaluate SLIDE on three multi-view datasets: RealEstate10k [44] (RE10K), Dual-Pixels [11] and Mannequin-Challenge (MC) [20] that provide videos or multi-view images of a static scene. In addition, we perform user studies on the photographs from Unsplash [1].

Baselines and Metrics. We quantitatively compared with three recent state-of-the-art techniques, for which code is publicly available: SynSin [39], Single-image MPI [34] (SMPI) and 3D-Photo [31]. SynSin and SMPI are end-to-end trained networks that take a single image as input and generate novel-view images. 3D-Photo, on the other hand, is a modular approach that is not end-to-end trainable. Like SLIDE, 3D-Photo uses a disparity network coupled with a specialized inpainting network to generate novel-view images. For fair comparison, both 3D-Photo and SLIDE techniques use MiDaSv2 [29] disparities. But, unlike SLIDE, 3D-Photo does not model fine structures, like fur and hair, on foreground silhouettes. We refer to our model that does not use alpha mattes as 'SLIDE' and the one that does as 'SLIDE with Matte'.

Following SMPI [34], we quantitatively measured the accuracy of the predicted target views with respect to the ground-truth images using three different metrics of LPIPS [43], PSNR and SSIM. Since SLIDE and several baselines do not perform explicit out-painting (in-filling the newly exposed border regions), we ignore a 20% border region when computing the metrics.

Results on RealEstate10K. RealEstate10K (RE10K) [44] is a video clips dataset with around 10K YouTube videos of static scenes. We use 1K randomly sampled video clips from the test set for evaluations. We follow [34] and use structure-from-motion and SLAM (Simultaneous Localization and Mapping) algorithms to recover camera intrinsics and extrinsics along with the sparse point clouds. As in [34], we also compute the point-visibility from each frame. For the evaluation, we randomly sample a source view from each test clip and consider the following 5th ($t = 5$) and 10th ($t = 10$) frames as target views. We compute evaluation metrics with respect to these target views. Table 1 shows that SLIDE performs better or on-par with current state-of-the-art techniques with respect to all the evaluation metrics. The improvement is especially considerable in the LPIPS perceptual metric, indicating that SLIDE view synthesis preserves the overall scene content better than existing techniques. Figure 7 shows sample visual results. SMPI [34] generate blurrier novel views; and SLIDE usually preserves the structures better around the occlusion
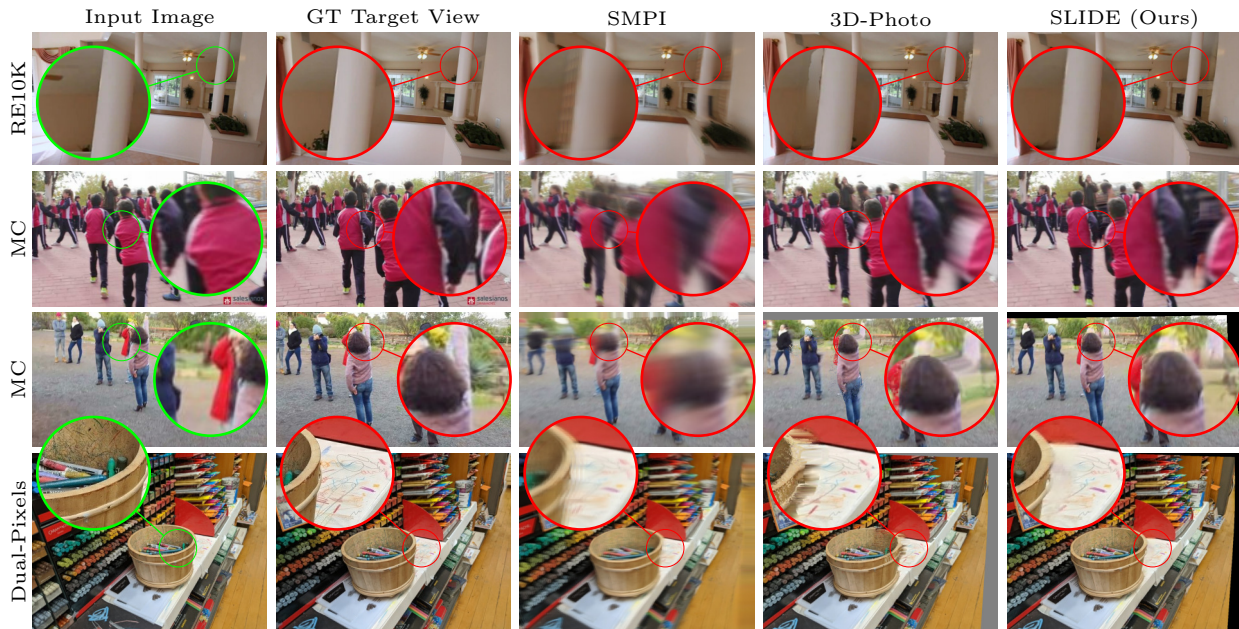
Figure 7: **Sample Visual Results on Benchmarks.** Novel view synthesis results of different techniques: Single-image MPI [34], 3D-Photo [31] and SLIDE (Ours) on sample images from RE10K [44], MC [20] and Dual-Pixels [11] datasets.
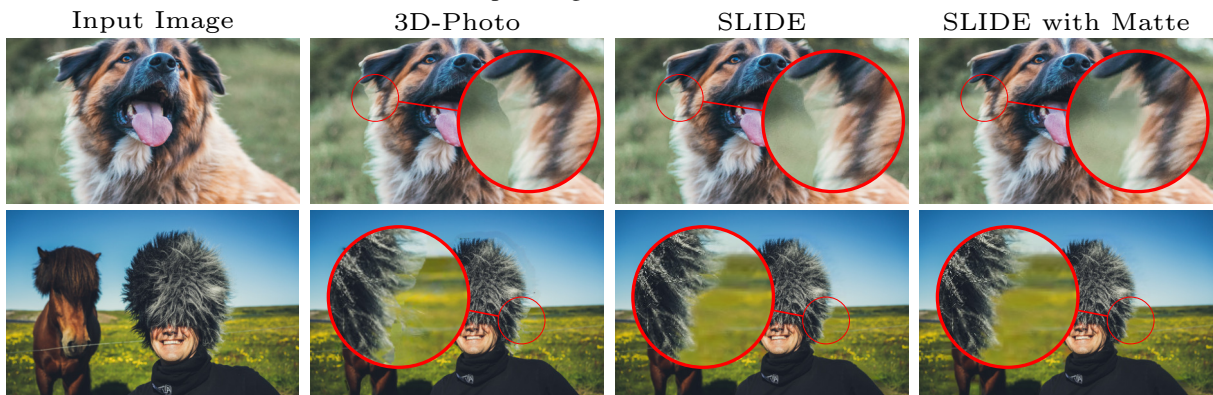

Figure 8: **Visual Results on in-the-wild Images.** View synthesis results on sample Unsplash dataset [1] images that we use in user studies. SLIDE and SLIDE with Matte approaches can more faithfully represent thin hair-like structures compared to 3D-Photo [31]. See the supplementary video for better illustration of results.

boundaries where the scene elements on images move with respect to each other when camera moves. On all the 3 benchmark datasets, we do not see further improvements by incorporating FG matting into SLIDE (SLIDE with Matte), as these dataset images do not have predominant foreground objects with thin hair-like structures.

Results on Dual-Pixels. Dual-Pixels [11] is a multi-view dataset taken with a custom-made, hand-held capture rig consisting of 5 mobile phones. That is, each scene is captured simultaneously with 5 cameras that are separated by a moderate baseline. We evaluate SLIDE and other baselines on the 684 publicly available test scenes. For each scene, we consider one of the side views as input and consider the remaining 4 views as target views. Compared to RE10K, dual-pixels data consists of more challenging scenes cap-

tured in diverse settings. Table 2 shows the quantitative results on the dual-pixels test dataset. Results show that SLIDE outperforms all three baselines. Figure 7 shows some qualitative results.

Results on Mannequin Challenge (MC). MC is a video dataset collected and processed in a similar fashion as RE10K dataset. It contains videos of people performing the "mannequin challenge," in which subjects attempt to hold still as a camera moves through the scene. This (near) static setup enables the use of standard structure-from-motion pipelines, like in RE10K, to obtain ground-truth camera poses and rough 3D point clouds of the scenes. MC provides a good benchmark for view synthesis in scenes with people. We randomly sampled 190 5-frame sequences from the publicly available test set for our evaluation purposes.

Method	LPIPS ↓		PSNR ↑		SSIM ↑	
	$t = 5$	$t = 10$	$t = 5$	$t = 10$	$t = 5$	$t = 10$
SynSin [39]	0.31	0.34	22.7	20.6	0.72	0.67
SMPI [34]	0.14	0.19	26.7	24.1	0.86	0.80
3D-Photo [31]	0.09	0.12	26.9	23.7	**0.87**	**0.80**
SLIDE (Ours)	**0.06**	**0.10**	**27.1**	23.7	**0.87**	**0.80**

Table 1: **Results on RE10K.** LPIPS [43], PSNR and SSIM scores of different techniques computed w.r.t. target views at two time steps $t = 5$ and $t = 10$.

	LPIPS ↓	PSNR ↑	SSIM ↑
SynSin [39]	0.75	12.9	0.31
SMPI [34]	0.49	16.3	0.42
3D-Photo [31]	0.27	16.3	0.42
SLIDE (Ours)	**0.23**	**16.8**	**0.44**

Table 2: **Results on Dual Pixels.** LPIPS [43], PSNR and SSIM scores of different techniques computed w.r.t. four target views in the 684 test data scenes.

	LPIPS ↓	PSNR ↑	SSIM ↑
SynSin [39]	0.33	16.9	0.53
SMPI [34]	0.31	17.8	0.59
3D-Photo [31]	0.22	18.8	0.60
SLIDE (Ours)	**0.18**	**20.0**	**0.66**

Table 3: **Results on Mannequin Challenge dataset.** LPIPS [43], PSNR and SSIM scores of different techniques computed w.r.t. four target views.

We treat the first frame as input and the remaining four as target views. Table 3 shows the average metrics for the different single-image 3D photography techniques. Again we observe that SLIDE outperforms the other techniques. Figure 7 shows some qualitative results.

User Studies on in-the-wild images. In order to evaluate different techniques on "in-the-wild" photographs, we assembled two image sets from Unsplash [1], a database of free-licensed, high-quality photographs taken by professional photographers and hobbyists. As we had no GT for these images, we performed user studies to compare view-synthesis results of different techniques. For the first set (Set-1), we collected 99 images with the elements we usually see in photo albums: people, animals, vehicles and landscapes; sometimes with multiple instances of these elements. For the second set (Set-2), to demonstrate the use of matting, we collected 50 images with close-ups of people and animals with thin hair structures. For each image in both sets, we created short videos with views synthesized from circular camera paths (Please refer to the supplementary video to see sample generated videos). We then showed side-by-side results of our (SLIDE) method and a baseline method (in randomized pair order) to Amazon Mechanical Turkers and asked the user to choose the better looking video. At least 15 users rated each video pair, and we took the majority vote to compute the percentage of time users
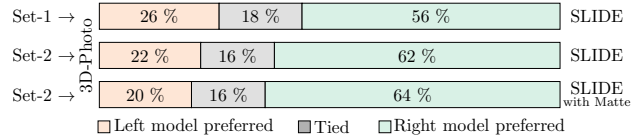

Figure 9: **User Study.** Percentage of results users preferred between 3D-Photo and SLIDE results. SLIDE results are consistently preferred on both sets of in-the-wild images.

preferred one result over another. User studies show SLIDE was preferred over SynSin [39] or SMPI [34] over 99% of time. This result is not surprising, as these approaches usually produce blurrier results than depth-based rendering approaches like SLIDE and 3D-Photo [31]. Figure 9 shows the result of comparing SLIDE and 'SLIDE with Matte' to 3D-Photos. On both image sets, users prefer SLIDE considerably more often than 3D-Photo. SLIDE is preferred still more on Set-2 images with thin hair-like structures, which are not well-handled by 3D-Photo, with additional gains when adding matting to the SLIDE pipeline ('SLIDE with Matte'). Figure 8 shows some sample view synthesis results with more in the supplementary material.

Runtime Analysis. The SLIDE framework requires only a single forward pass over all its components to generate the two layer representation, which can then be rendered in realtime on modern graphics chips to synthesize new views. All of the SLIDE components can be realized using standard GPU layers in modern deep learning toolboxes resulting in a unified and efficient system. We implemented most of the SLIDE components in Tensorflow except the depth estimation, foreground saliency and alpha matting networks, for which we used the original PyTorch networks. The following are the runtimes of different SLIDE components when processing a 672×1008 image on an Nvidia Titan P100 GPU: Depth estimation [29] (0.023s), Soft-layering (0.013s), Depth-aware inpainting (0.037s). And optionally for matting: foreground segmentation [28] (0.069s) and FBA-matting [9] (0.203s), for a total runtime of 0.07s or 0.35s (w/o or w/ matting, resp.). By comparison, the closest competitor in quality, 3D-Photo, takes several seconds to process a single image.

5. Conclusion

SLIDE is a modular yet unified approach for 3D photography that has several favorable properties: Soft layering that can model intricate appearance details; depth-aware inpainting; modular and unified system with efficient runtime; and state-of-the-art results. We observe artifacts when a component such as depth estimation or alpha matting fails. We believe that the SLIDE 3D photographs will get better as these components become even more mature and robust.

References

[1] Unsplash Dataset. https://unsplash.com/data, 2020. [Online; accessed March-2020]. 6, 7, 8

[2] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. PatchMatch: A randomized correspondence algorithm for structural image editing. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 28(3), Aug. 2009. 2

[3] Peter N Belhumeur. A Bayesian approach to binocular steropsis. *International Journal of Computer Vision*, 19(3):237–260, 1996. 4

[4] Marcelo Bertalmio, Guillermo Sapiro, Vincent Caselles, and Coloma Ballester. Image inpainting. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 417–424, 2000. 2

[5] Mark Boss, Raphael Braun, Varun Jampani, Jonathan T Barron, Ce Liu, and Hendrik Lensch. NeRD: Neural reflectance decomposition from image collections. In *The IEEE International Conference on Computer Vision (ICCV)*, 2021. 2

[6] Weifeng Chen, Zhao Fu, Dawei Yang, and Jia Deng. Single-image depth perception in the wild. In *Advances in Neural Information Processing Systems (NIPS)*, pages 730–738. 2016. 2

[7] John Flynn, Michael Broxton, Paul Debevec, Matthew Du-Vall, Graham Fyffe, Ryan Overbeck, Noah Snavely, and Richard Tucker. DeepView: View synthesis with learned gradient descent. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[8] John Flynn, Ivan Neulander, James Philbin, and Noah Snavely. DeepStereo: Learning to predict new views from the world's imagery. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2

[9] Marco Forte and François Pitié. F, B, alpha matting. *arXiv preprint arXiv:2003.07711*, 2020. 5, 8

[10] Ravi Garg, Vijay Kumar BG, Gustavo Carneiro, and Ian Reid. Unsupervised CNN for single view depth estimation: Geometry to the rescue. In *European Conference on Computer Vision (ECCV)*, 2016. 2

[11] Rahul Garg, Neal Wadhwa, Sameer Ansari, and Jonathan T Barron. Learning single camera depth estimation using dual-pixels. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 6, 7

[12] Kyle Genova, Forrester Cole, Aaron Maschinot, Aaron Sarna, Daniel Vlasic, and William T Freeman. Unsupervised training for 3d morphable model regression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8377–8386, 2018. 6

[13] Steven J. Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F. Cohen. The lumigraph. In *Proceedings of SIGGRAPH 96*, Annual Conference Series, 1996. 2

[14] Jia-Bin Huang, Sing Bing Kang, Narendra Ahuja, and Johannes Kopf. Image completion using planar structure guidance. *ACM Trans. Graph.*, 33(4), July 2014. 2

[15] Nima Khademi Kalantari, Ting-Chun Wang, and Ravi Ramamoorthi. Learning-based view synthesis for light field cameras. *ACM Trans. Graph.*, 35(6):193:1–193:10, 2016. 2

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

[17] Johannes Kopf, Kevin Matzen, Suhib Alsisan, Ocean Quigley, Francis Ge, Yangming Chong, Josh Patterson, Jan-Michael Frahm, Shu Wu, Matthew Yu, et al. One shot 3D photography. *ACM Transactions on Graphics (TOG)*, 39(4):76–1, 2020. 1

[18] Marc Levoy and Pat Hanrahan. Light field rendering. In *Proceedings of SIGGRAPH 96*, Annual Conference Series, 1996. 2

[19] Qinbo Li and Nima Khademi Kalantari. Synthesizing light field from a single image with variable MPI and two network fusion. *ACM Transactions on Graphics*, 39(6), 12 2020. 2

[20] Zhengqi Li, Tali Dekel, Forrester Cole, Richard Tucker, Noah Snavely, Ce Liu, and William T. Freeman. Learning the depths of moving people by watching frozen people. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 1, 6, 7

[21] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 1, 2

[22] Hongyu Liu, Bin Jiang, Yi Xiao, and Chao Yang. Coherent semantic attention for image inpainting. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4170–4179, 2019. 2

[23] Ricardo Martin-Brualla, Noha Radwan, Mehdi S. M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, and Daniel Duckworth. NeRF in the wild: Neural radiance fields for unconstrained photo collections. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[24] Moustafa Mahmoud Meshry, Dan B Goldman, Sameh Khamis, Hugues Hoppe, Rohit Kumar Pandey, Noah Snavely, and Ricardo Martin Brualla. Neural rerendering in the wild. In *Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[25] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *European Conference on Computer Vision (ECCV)*, 2020. 2

[26] Simon Niklaus, Long Mai, Jimei Yang, and Feng Liu. 3D Ken Burns effect from a single image. *ACM Transactions on Graphics (TOG)*, 2019. 1, 2, 3

[27] Eric Penner and Li Zhang. Soft 3D reconstruction for view synthesis. *ACM Trans. Graph.*, 36(6):235:1–235:11, 2017. 2

[28] Xuebin Qin, Zichen Zhang, Chenyang Huang, Masood Dehghan, Osmar R Zaiane, and Martin Jagersand. U2-net: Going deeper with nested u-structure for salient object detection. *Pattern Recognition*, 106:107404, 2020. 5, 8

[29] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2020. 1, 2, 3, 6, 8

[30] Jonathan Shade, Steven Gortler, Li-wei He, and Richard Szeliski. Layered depth images. In *Proceedings of SIG-GRAPH 98*, 1998. 2

[31] Meng-Li Shih, Shih-Yang Su, Johannes Kopf, and Jia-Bin Huang. 3d photography using context-aware layered depth inpainting. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 1, 2, 6, 7, 8

[32] 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 *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 2

[33] Pratul P. Srinivasan, Tongzhou Wang, Ashwin Sreelal, Ravi Ramamoorthi, and Ren Ng. Learning to synthesize a 4D RGBD light field from a single image. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct. 2017. 2

[34] Richard Tucker and Noah Snavely. Single-view view synthesis with multiplane images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 1, 2, 6, 7, 8

[35] Shubham Tulsiani, Richard Tucker, and Noah Snavely. Layer-structured 3D scene inference via view synthesis. In *The European Conference on Computer Vision (ECCV)*, Sept. 2018. 2

[36] Jialiang Wang, Varun Jampani, Deqing Sun, Charles Loop, Stan Birchfield, and Jan Kautz. Improving deep stereo network generalization with geometric priors. *arXiv preprint arXiv:2008.11098*, 2020. 4

[37] Jialiang Wang and Todd Zickler. Local detection of stereo occlusion boundaries. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3818–3827, 2019. 4

[38] Joachim Weickert. Coherence-enhancing diffusion filtering. *International journal of computer vision*, 31(2-3):111–127, 1999. 2

[39] Olivia Wiles, Georgia Gkioxari, Richard Szeliski, and Justin Johnson. SynSin: End-to-end view synthesis from a single image. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 1, 2, 6, 8

[40] Zili Yi, Qiang Tang, Shekoofeh Azizi, Daesik Jang, and Zhan Xu. Contextual residual aggregation for ultra high-resolution image inpainting. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 1, 2, 5, 6

[41] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Generative image inpainting with contextual attention. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1, 2

[42] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S. Huang. Free-form image inpainting with gated convolution. In *IEEE International Conference on Computer Vision (ICCV)*, Oct. 2019. 2, 5, 6

[43] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 6, 8

[44] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: learning view synthesis using multiplane images. *ACM Transactions on Graphics (TOG)*, 37(4):1–12, 2018. 2, 6, 7