

Stereo Magnification with Multi-Layer Images

T. Khakhulin^{1,2} D. Korzhenkov¹ P. Solovev¹ G. Sterkin¹ T.Ardelean^{1,2} V. Lempitsky^{2*}

¹Samsung AI Center, Moscow

²Skolkovo Institute of Science and Technology, Moscow

Abstract

Representing scenes with multiple semi-transparent colored layers has been a popular and successful choice for real-time novel view synthesis. Existing approaches infer colors and transparency values over regularly-spaced layers of planar or spherical shape. In this work, we introduce a new view synthesis approach based on multiple semi-transparent layers with scene-adapted geometry. Our approach infers such representations from stereo pairs in two stages. The first stage infers the geometry of a small number of data-adaptive layers from a given pair of views. The second stage infers the color and the transparency values for these layers producing the final representation for novel view synthesis. Importantly, both stages are connected through a differentiable renderer, and are trained in an end-to-end manner. In the experiments, we demonstrate the advantage of the proposed approach over the use of regularly-spaced layers with no adaptation to scene geometry. Despite being orders of magnitude faster during rendering, our approach also outperforms a recently proposed IBRNet system based on implicit geometry representation. See results at samsunglabs.github.io/StereoLayers.

1. Introduction

Recent years have seen a rapid progress in the image-based rendering and the novel view synthesis, with a multitude of diverse methods based on neural rendering approaches [33]. Among this diversity, the approaches that are based on semi-transparent multi-layered representations [21, 30, 31, 35, 41] stand out due to their combination of fast rendering time, compatibility with traditional graphics engines, and good quality of re-rendering in the vicinity of the input frames.

Existing approaches [4, 17, 21, 30, 31, 35, 41] build multi-layer representations over the grids of regularly spaced surfaces such as planes or spheres with uniformly changing

inverse depth. As the number of layers is necessarily limited by resource constraints and risk of overfitting, this number is usually taken to be relatively small (*e.g.* 32). The resulting semi-transparent representation may therefore only coarsely approximate the true geometry of the scene, which limits the generalization to novel views and introduces artefacts. The most recent works [4, 17] use excessive number of spheres (up to 128) and then merge the resulting geometry using non-learned post-processing merging step. While the merge step creates scene-adapted and compact geometric representation, it is not incorporated into the learning process of the main matching network, and degrades the quality of novel view synthesis [4].

The coarseness of layered geometry used by multi-layer approaches is in contrast to more traditional image-based rendering methods that start by estimating the *non-discretized* scene geometry in the form of mesh [26, 34], view-dependent meshes [11], a single-layer depth map [24, 28, 38]. The geometry estimates may come from multi-view dense stereo matching or from monocular depth. All these approaches obtain a finer approximation to scene geometry, though most of them have to use a relatively slow neural rendering step to compensate for the errors in the geometry estimation.

Our approach called *StereoLayers* (Fig. 1) combines scene geometry adaptation with multi-layer representation. It thus starts by building a geometric proxy that is customized to a particular scene. The proxy is formed by a small number of mesh layers with *continuous* depth coordinate values. In the second stage, similarly to other multi-layer approaches, we estimate the transparency and the color textures for each layer resulting in the final representation of the scene. When processing a new scene, both stages take the same pair of images of that scene as an input. Two deep neural networks pretrained on a dataset of similar scenes are used to implement these two stages. Crucially, we train both neural networks together in the end-to-end fashion using the differentiable rendering framework [16].

We compare our approach to the previously proposed methods that use regularly spaced layers on the popular

*Most of the work was done while Victor Lempitsky was at Samsung AI Center

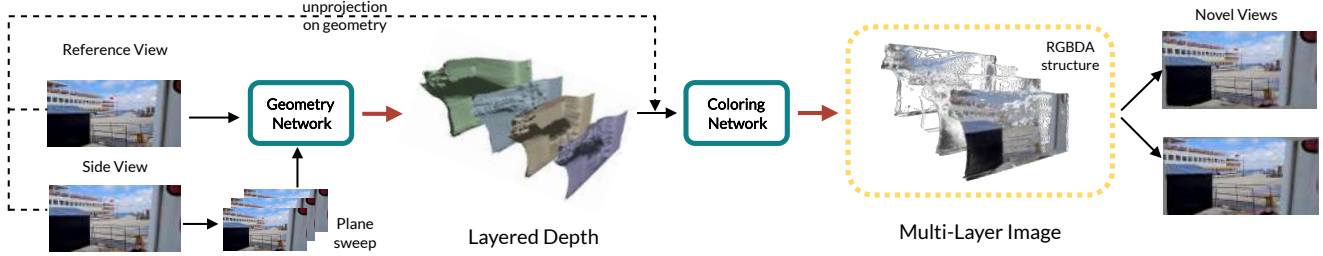


Figure 1. The proposed StereoLayers pipeline estimates scene-adjusted multi-layer geometry from the plane-sweep volume using a pre-trained geometry network, and after that estimates the color and transparency values using a pre-trained coloring network. The layered geometry represents the scene as an ordered set of mesh layers. The geometry and the coloring networks are trained together end-to-end.

RealEstate10k [41] and LLFF [21] datasets. Besides, we propose a more challenging new dataset for novel view synthesis benchmarking. In both cases, we observe that scene-adaptive geometry in our approach results in better novel view synthesis quality than the use of non-adaptive geometry. To put our work into a broader context, we also compare our system’s performance with the IBRNet system [37], and observe the advantage of our approach, despite considerably faster rendering time. In general, our approach produces very compact scene representations amenable for real-time rendering even on low-end devices.

To sum up, our contributions are as follows. First, we propose a new method for geometric scene reconstruction from stereo pairs. The method represents scenes using a small number of semi-transparent layers with scene-adapted geometry. Unlike other related methods, ours uses two jointly (end-to-end) trained deep networks, the first of which estimates the geometry of the layers, while the second estimates the transparency and the color textures of the layers. Finally, we evaluate our approach on a previously proposed dataset and introduce a new challenging dataset for the novel view synthesis methods training and evaluation.

2. Related works

Representations for novel view synthesis. Over the years, different kind of representations have been proposed for novel view synthesis. Almost without exception, when such representations are acquired from multiple images, those are registered using structure-and-motion algorithm or come from a pre-calibrated stereo-rig. Alternatively, some of the recent works investigate the creation of such representations from a single image [36, 38]. The proposed representations fall into several classes, including volumetric representations that rely on volumetric rendering [10, 19, 22, 29], mesh-based representations [7, 11, 12, 26, 34, 42], point-based representations [1, 15]. Most of the representations of these types require extensive computations in order to render a novel view, such as running a raw image through a deep convolutional rendering network [33] or numerous

evaluation of a scene network that has perceptron architecture [18, 22].

An important class of representations is based on depth maps. Such depth maps can be naturally obtained using stereo matching [5] or from monocular depth estimation [28, 38]. In this class, the 3D layered inpainting approach [28] is most related to our work, since after starting from a monocular depth map, it performs its segmentation into multiple layers, and then apply the inpainting procedure to each layer to extend its support behind the more frontal layers. Our work has several important differences, as it uses two (rather than one) images as an input and predicts transparency of the layers. Most importantly, the estimation of the multiple layer geometry and the estimation of their colors and transparency are both implemented using deep architectures, which are trained in an end-to-end fashion.

Multi-layer semi-transparent representations. In 1999, [31] proposed to represent scenes with multiple fronto-parallel semi-transparent layers and to acquire such representations through stereo-matching of a pair of input views. Twenty years later, a number of approaches [8, 21, 30] starting from [41] exploited advances in deep learning to build deep networks that directly map *plane sweep volumes* (i.e. tensors obtained by the “unprojection” operation) to final representations of the same kind. The rendering of semi-transparent layers is well-supported by modern graphics engines, thus the resulting representation is in general more suitable to interactive applications than most other representations that lead to similar level of realism.

The multi-layer representations have been extended to wider fields of view in [3, 4, 17] by replacing planes with spheres. Two approaches [4, 17] suggested to “coalesce” (merge) the groups of nearby layers into layers with scene-adapted geometry. In both cases, the grouping of layers is predefined and the merge process is non-learnable and uses simple accumulation heuristics. Consequently, [4] reported the loss of rendering quality as a result of such merge, which is still justified in their case by increased rendering and storage efficiency.

Our work is highly related to previous works on multi-



Figure 2. View extrapolations obtained by our method. The two input images are shown in the middle. The proposed method (StereoLayers) generate plausible renderings even when the baseline is magnified by a factor of 5x (as in this case).

layer semi-transparent representations. Unlike most works in this group, our pipeline starts with scene-adapted (non-planar, non-spherical) layer estimation and only then estimates the colors and the transparencies of the layers. While [4, 17] also end up with scene-adapted semi-transparent layers as a representation, our approach performs the reconstruction in the opposite order (the geometry is estimated first). More importantly, unlike [4, 17] we estimate the geometry of layers using a neural network, which is trained jointly with the color and transparency estimation network. In the experiments, we show that such approach results in better view synthesis.

Single-layer new view synthesis with differentiable rendering. SynSin [38] and, more recently, Worldsheet [13] systems predict single-layered geometry from a single image, and use differentiable rendering to learn the predicting neural network in a way that is similar to our method. Our approach considers the case of two input images and focuses on multi-layered geometry. While a variant of Worldsheet considers two-layer extension, it is based on a different architecture and a different layer aggregation strategy, and, most importantly, does not outperform a single-layer representation in their experiments.

3. Multi-layer representation from stereo

We consider the task of generating a novel view \hat{I}_n of the scene, based on two input views (images): a reference view I_r and a side view I_s . We assume that the relative camera poses π_s and π_n of the side and novel views to the reference view and the camera intrinsics K_r , K_s and K_n are given. To solve this task, our approach builds the scene representation that depends only on side and reference views. Afterwards, such a representation can be rendered on any novel camera with standard graphic engines (without reestimating scene representation). We now describe our approach in detail. We first explain the rendering procedure of a trained model and then discuss the training process.

3.1. Geometry estimation

Given a trained model and a new stereo pair, the multi-layer representation is inferred in two stages. First, the structure of the scene, such as the geometry of mesh layers, is predicted. Then, in the second stage, the layers' opacity (alpha) and color (textures) are inferred. It is worth noting that we treat the pair of input views asymmetrically as we build the scene representation in the frustum of the reference camera.

We start by computing the plane sweep volume (PSV) [6] by placing P fronto-parallel planes in the reference camera frustum and unprojecting the side view onto these planes. The planes are spaced uniformly in the inverse depth space at depths $\{d_1, \dots, d_P\}$. We sample the planes at $H \times W$ resolution and concatenate the reference view as an additional set of three channels, resulting in $H \times W \times (3P + 3)$ -sized tensor, which is similar to the one used in other multi-layer approaches, e.g. [41], as an input.

The input tensor is then processed by the *geometry network* F_g . While we consider several variants of the architectures discussed below, all these architectures predict L depth maps of size $h \times w$, that correspond to the depths along with the $h \times w$ pencil of rays uniformly spaced in the image coordinate space of the reference view. In our experiments we set the resolution on layers equal to the size of the reference view, $w = W$, though sampling in different resolution is also possible. The backbone of F_g is similar to the depth prediction module of SynSin [38], *i.e.* is a UNet-like 2D-convolutional net with spectral normalization. The only difference is that we increased the number of input and output featuremaps to address the multi-layer nature of our model. More detailed description of the backbone is provided in Supplementary. We consider the following three schemes for encoding the layers.

Group compositing (GC) scheme. In this scheme, F_g returns the tensor of shape $h \times w \times P$ with values in the range between 0 and 1. The P channels and corresponding

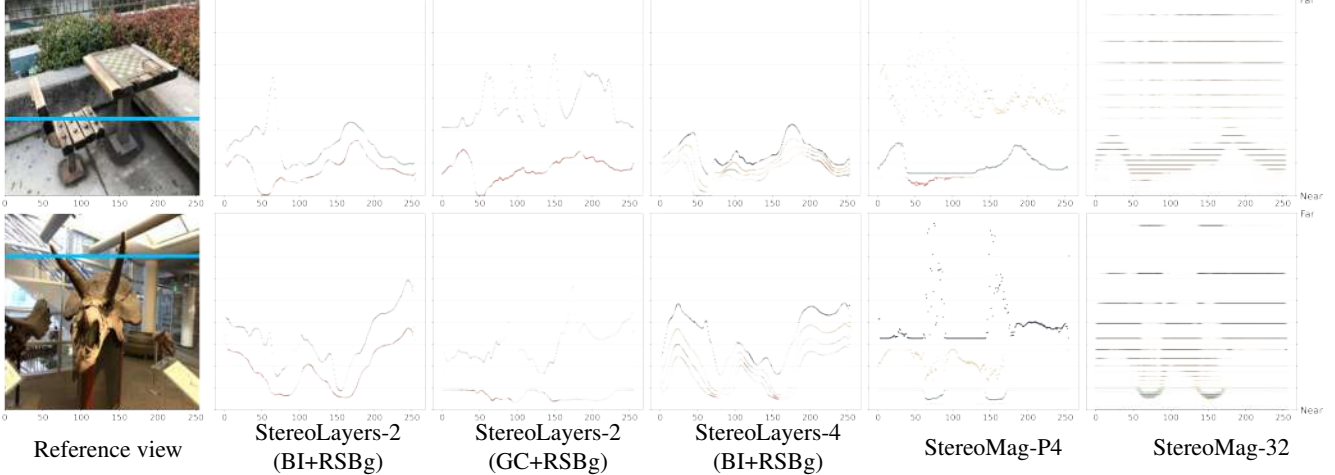


Figure 3. For the two stereopairs (only reference views are shown), we visualize horizontal slices along the blue line. Mesh vertices are shown as dots with the predicted opacity. Colors encode the layer number. The horizontal axis corresponds to the pixel coordinate, while the vertical axis stands for the vertex depth w.r.t. the reference camera (only the most illustrative depth range is shown). StereoLayers method variants generate scene-adaptive geometry in a more efficient way than StereoMag resulting in more frugal geometry representation, while also obtaining better rendering quality.

P planes of PSV are divided into L groups of equal size. Then L deformable layers are obtained as follows: within each group j , $1 \leq j \leq L$, the depth value \hat{d}_j is computed by over-composing [25] the planes’ depths $d_1 < \dots < d_P$ with ‘opacities’ $\{\beta_k\}_{k=1}^P$ predicted by F_g network.

$$\hat{d}_j = \sum_{k=I_j^-}^{I_j^+} d_k \beta_k \prod_{i=I_j^-}^{k-1} (1 - \beta_i), \quad j = 1, \dots, L, \quad (1)$$

where I_j^- and I_j^+ are the indices of limiting planes for each group: $I_j^- = 1 + (j - 1)P/L$, and $I_j^+ = jP/L$. For simplicity of notation, layers and planes in Eq. (1) are enumerated in the front-to-back order. The ‘opacities’ $\beta_{I_j^+}$ (corresponding to farthest planes of each group) are manually set to 1 for $1 \leq j \leq L$. As a result, the depth of j -th layer is bounded by design: $d_{I_j^-} \leq \hat{d}_j \leq d_{I_j^+}$. The compositing Eq. (1) is evaluated for each of $h \times w \times L$ positions independently.

The group compositing scheme is inspired by the merge procedure from [4], but moves this procedure inside the learnable scene representation and before texture and transparency estimation. The main benefit of GC procedure is a guarantee that the L layers do not intersect and have explicit ordering.

Soft-aggregation (SA) scheme. The main drawback of the GC depth aggregation is non-adaptive partition of the depth interval into L layers. Such non-adaptive partition tends to waste representation capacity for parts that do not contain scene surfaces and to underfit parts where multiple layers are beneficial for scene representation. To overcome this, we make F_g to predict the tensor of size $h \times w \times L \cdot P$, that is further reshaped to $h \times w \times L \times P$. After that, soft-max is applied along the last dimension, and the obtained

values are used as weights for the planes’ depths $\{d_k\}_{k=1}^P$ (where the P depths span the whole depth range). These depths are averaged with the predicted weights, and the resulting tensor of shape $h \times w \times L$ is obtained, containing the depths of the layers. It is worth noting, that unlike GC approach this scheme neither provides any ordering of layers, nor guarantees the absence of intersections. Therefore, a special loss promoting the non-intersecting should be applied during training.

Bounds interpolation (BI) scheme. We also consider a simplified version of SA scheme that predicts only weights to blend the minimum depth value d_1 and the maximum depth value d_P (effectively predicting depths by direct regression). In this scheme, F_g network returns the tensor of shape $h \times w \times L$ with values in the range between 0 and 1. The depth \hat{d}_j of j -th layer is computed as $\hat{d}_j = \beta_j d_1 + (1 - \beta_j) d_P$, where β_j is the output of the geometry network. In our experiments this scheme achieves the best results, thus we select BI method as our default one.

Meshing. Irrespective of the layer depth prediction scheme, we treat each predicted layer as a mesh. We use the simplest mesh connectivity pattern, connecting each vertex with the nearby six nodes with edges so that each quad defined by four adjacent vertices is meshed with two triangles. Hereinafter the whole set of resulting L meshes is referred to as the *layered mesh*. The examples of the estimated geometry are showcased in Fig. 3. Now we explain the explored approaches to depth prediction.

3.2. Mesh texturing.

The second stage of the inference process completes the scene reconstruction by inferring the color and the opac-

	SWORD				RealEstate10K				LLFF			
	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	∇ LIP \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	∇ LIP \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	∇ LIP \downarrow
StereoMag-32	24.45	0.76	0.107	0.17	31.40	0.93	0.031	0.10	20.67	0.65	0.132	0.24
StereoMag-8	23.00	0.69	0.126	0.21	27.76	0.90	0.044	0.17	19.13	0.55	0.152	0.29
StereoMag-P8	22.31	0.66	0.209	0.22	22.00	0.69	0.160	0.24	20.11	0.63	0.156	0.27
StereoMag-P4	23.69	0.74	0.137	0.20	28.06	0.89	0.066	0.15	20.29	0.64	0.150	0.26
IBRNet	23.82	0.71	0.188	0.17	30.26	0.90	0.058	0.10	21.19	0.67	0.207	0.22
StereoLayers-8	25.54	0.79	0.113	0.14	31.52	0.92	0.027	0.10	21.58	0.69	0.149	0.21
StereoLayers-4	25.95	0.81	0.096	0.14	32.61	0.94	0.026	0.08	22.19	0.73	0.125	0.20
StereoLayers-2	25.28	0.78	0.102	0.14	31.29	0.92	0.025	0.09	20.78	0.66	0.141	0.22

Table 1. Results of evaluation on SWORD, RealEstate10K [41], and LLFF datasets [21]. For the latter dataset models were trained on SWORD. All metrics are computed on central crops of synthesized novel views. Our approach outperforms all baselines on these datasets despite containing less layers in the scene proxy. Notably, StereoLayers method surpasses IBRNet despite the latter was trained on 80% of LLFF scenes in multi-view setting. The number after the model type denotes the number of layers in the estimated geometry. Suffix *P* stands for the model after the applied postprocessing.

ity textures for the layered mesh. The process is similar to [41] and follow-up works with some important modifications. Most importantly, we consider non-planar/non-spherical layers predicted by the previous stage. We thus ‘unproject’ the side view onto each of the L layers, and then sample each of those reprojections at the $H \times W$ resolution. We employ the `nvdiffrast` differentiable renderer [16] to make the rasterization process differentiable w.r.t. the layered mesh geometry. The reference view sampled at the same resolution is concatenated, resulting in a $H \times W \times (3L + 3)$ -sized tensor.

This tensor is then processed by the *coloring network* F_c that aims to infer the color and the opacity values for the mesh layers. Ultimately, our goal is predict the RGB and alpha values for each pixel in each layer. Previously, the authors of StereoMag [41] observed that predicting the RGB color indirectly produces better results. Namely, they predicted a single “background” RGB image of the size $H \times W \times 3$ and the layer-specific mixture weight tensor $H \times W \times L \times 2$ that provides the blending weights for the linear combination of the reference view with the background. We have confirmed their finding. We have further observed that in our case even better results can be obtained by predicting an additional mixture weight tensor of size $H \times W \times L$ that contain blending weights for the side view unprojected to each layer.

Within our texture prediction scheme, the network F_c thus produces a tensor of size $H \times W \times 3L + 3$ with the last three channels corresponding to the background image, and the remaining channels contain the mixture weights for the **Reference** view, the unprojected **Side** view, and the **Background** image. We refer to this scheme as *RSBg*, and it is default in our experiments. In the ablation study, we further compare it to the scheme from StereoMag, when only the reference and the background images are blended into the texture (denoted as *RBg*), and to the scheme that predicts the RGB colors directly (denote as *RAW*).

In all cases, alongside with RGB values, the network F_c

also predicts a tensor of shape $H \times W \times L$ containing the opacity (alpha) values for each layer. Note that the texturing scheme is able to eliminate the redundant layers by setting their opacity values to zero (Fig. 3 provides such examples, where some layers were made transparent by the texturing network).

The architecture of F_c is borrowed in [41] (except different shapes of the output tensors). It is thus a 2D-convolutional UNet-like net with dilated convolutions in the bottleneck. For completeness, we detail this architecture in the supplementary material.

Rendering. To render a novel view, we project the mesh layers according to the desired camera pose while compositing them using the *compose-over* operator [25]. Fig. 2 demonstrates novel views synthesized with the proposed pipeline.

3.3. Learning

We learn the parameters of the geometry network F_g and the coloring network F_c from datasets of short videos of static scenes, for which camera pose sequences have been estimated using structure-and-motion [27]. Overall, the training is performed by minimizing the weighted combination of losses discussed below.

Image-based losses. Similarly to previous works, *e.g.* [41], the main training loss comes from the image supervision. For example, at each training step, we sample the image triplet (I_s, I_r, I_n) containing the side view I_s , the reference view I_r and the novel (hold-out) view I_n from a training video. Given the current network parameters, we estimate the scene geometry and the textures from (I_s, I_r) and then project the resulting representation onto the I_n resulting in the predicted image \hat{I}_n . We then compute the perceptual [14] and the L_1 losses between I_n and \hat{I}_n and back-propagate them through both networks F_g and F_c .

Ordering and smoothness regularization losses. As was explained above, the BI and the SA schemes of depth prediction cannot guarantee the ordering of layers. Therefore,

Depth estimation	Texturing scheme	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	∇ LIP \downarrow
BI	RSBg	25.95	0.81	0.096	0.14
BI	RBg	24.96	0.77	0.111	0.15
BI	RAW	24.90	0.77	0.099	0.15
SA	RSBg	24.66	0.76	0.121	0.16
SA	RAW	24.30	0.75	0.099	0.15
GC	RSBg	25.24	0.77	0.115	0.15
GC	RAW	24.90	0.77	0.107	0.15

Table 2. Evaluation of StereoLayers configs. Top row represents a model chosen as a default one. Details of the predicting schemes are discussed in Sec. 3. All the configurations were trained with $P = 32$ planes and $L = 4$ layers.

Dataset	Baseline	Our score, %	p -value
SWORD	StereoMag-32	61	< 0.001
	IBRNet	81	< 0.001
LLFF	StereoMag-32	54	< 0.001
	IBRNet	70	< 0.001

Table 3. User study results. The 3rd column contains the ratio of users who selected the output of our model (StereoLayers-4) as more realistic in side-by-side comparison.

we apply a simple hinge loss with zero margin to layers with neighbor indices to ensure that they are predicted in front-to-back order: $L_{ord} = \sum_{j=0}^{L-1} \max(0, \hat{d}_j - \hat{d}_{j+1})$. Additionally, we regularize the geometry of the layers by imposing the total variation (TV) loss on the depths of each layer (the total variation is computed for each of the $h \times w$ maps encoding the depths).

Adversarial loss. While image-based and geometric losses suffice to obtain the plausible quality of novel view generation for RSBg and RBg coloring schemes (see Sec. 4 for metrics), we did not manage to obtain satisfactory results with RAW scheme without adversarial learning. Namely, we impose adversarial losses [9] only for RAW scheme on the predicted images \hat{I}_n . The main goal of adversarial losses is to reduce unnatural artefacts such as ghosting and duplications. To regularize the discriminator, R_1 penalty [20] is applied. We stress that the adversarial loss is only needed for RAW prediction and is not used in our default configuration.

4. Experiments

4.1. Datasets

We consider the RealEstate10k dataset and the Local Lightfield Fusion (LLFF) dataset introduced in previous works, while also proposing a new dataset. The details of the three datasets are provided below.

RealEstate10k dataset. Following prior works [28, 38, 41], we evaluate our approach on the subset of *RealEstate10k* [41] dataset containing consecutive frames from real estate videos with camera parameters. The subset

Transfer	Model	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	∇ LIP \downarrow
(R) \rightarrow (S)	StereoMag-32	24.45	0.76	0.107	0.17
	StereoLayers-4	25.47	0.79	0.098	0.14
(S) \rightarrow (R)	StereoMag-32	31.40	0.93	0.031	0.10
	StereoLayers-4	31.91	0.93	0.029	0.09
(R) \rightarrow (L)	StereoMag-32	20.31	0.62	0.129	0.23
	StereoLayers-4	21.52	0.69	0.133	0.21
(S) \rightarrow (L)	StereoMag-32	20.67	0.65	0.132	0.24
	StereoLayers-4	22.19	0.73	0.125	0.20

Table 4. Cross-dataset generalization. We evaluate models on RealEstate10k (R), SWORD (S) and LLFF (L) datasets. Notation (X) \rightarrow (Y) denotes model, trained on X and being evaluated on Y. Generally, our approach is on par or more robust to the dataset shift, while having a more compact representation. Evaluation on hold-out LLFF dataset also shows the benefit of training on the proposed SWORD dataset (compared to RealEstate10k).

used in our experiments consists of 10,000 scenes for training and 7,700 scenes for test purposes. The RealEstate10k dataset serves as the most popular benchmark for novel view synthesis pipelines. Despite the relatively large size, the diversity of scenes in the dataset is limited. The dataset does not contain enough scenes with central objects, and is predominantly indoor. Consequently, models trained on RealEstate10k generalize poorly to outdoor scenes or scenes with large closeby objects [28, 41].

SWORD dataset. To evaluate our (and prior) methods on more diverse data, we have collected a new dataset, which we call *Scenes With Occluded Regions Dataset* (SWORD). The new dataset contains around 1,500 train video and 290 test videos, with 50 frames per video on average. The dataset was obtained after processing the manually captured video sequences of static real-life urban scenes. The processing pipeline was the same as described in [41].

The main property of the dataset is the abundance of close objects and, consequently, larger prevalence of occlusions. To prove this quantitatively, we calculate occlusion areas, i.e. areas of those regions of the novel frames that are occluded in the reference frames. To get the masks for such regions, the off-the-shelf optical flow estimator [32] is employed. The full procedure of getting the occlusion masks and the examples of those masks are provided in Supplementary. According to this heuristic, the mean area of occluded image parts for SWORD is approximately five times larger than for RealEstate10k data (14% vs 3% respectively). This rationalizes the collection and usage of SWORD and explains that SWORD allows training more powerful models despite being of smaller size.

LLFF dataset. LLFF dataset is another popular dataset with central objects that was released by the authors of Local Lightfield Fusion paper [21]. It is too small to train on it (40 scenes), and we use this data for evaluation goals only to test the models trained on other two datasets.

4.2. Evaluation details

Compared approaches. We use the StereoMag system [41] as our main baseline. By default, the StereoMag system uses 32 regularly spaced fronto-parallel planes (with uniformly spaced inverse depth), for which color and transparency textures are estimated by a deep network operating on a plane sweep volume. The obtained representation is usually referred to as multi-plane images (MPI). The original StereoMag system uses such plane-based geometry for final renderings. In the comparisons, we refer to this baseline as *StereoMag-32* or simply *StereoMag*.

Additionally, we have evaluated variants of the StereoMag (denoted as *StereoMag-P8* and *-P4*) that coalesce the 32 planes into 8 and 4, respectively, non-planar meshes. The coalescence procedure is detailed in the Supplementary material and is very similar to the one proposed in [4]. Finally, for completeness, we trained a variant of StereoMag with eight planes (*StereoMag-8*). We stress that while StereoMag system was proposed some time ago, based on the comparison in the recent work [28, Appendix A], it remains state-of-the-art for two image inputs.

We also consider the more recent IBRNet [37] system trained to model the radiance field of the scene by blending features of the source images. Unlike StereoMag, this approach has no restrictions on the number of input frames, though requires a very significant amount of computations to generate each view. Moreover, as the authors have shown, IBRNet shows its best quality after fine-tuning on the new scene under consideration. For evaluation, we used the implementation and checkpoints of the network, provided by the authors, who used 80% of LLFF dataset for training among other data. Despite this, we compared with this method on all data (including LLFF). We have also tried to retrain IBRNet on the SWORD dataset (with two image input setting). This however lead to considerably worse performance, so we stick with the authors’ provided variant. We also note that test-time fine-tuning of IBRNet is not possible with two view inputs.

We trained different variations of our model with $L \in \{2, 4, 8\}$ layers obtained from $P = 32$ planes of PSV unless another number is specified. All models were trained for 500,000 iterations with batch size 4 on a single NVIDIA P40 GPU. For our approach, we set the following weights for the losses, described above: 1 for L_1 loss, 10 for perceptual loss, 5 for TV regularization and 2 for ordering loss. The RAW scheme for RGB prediction required a careful tuning of parameters, and we report its results for the configuration with adversarial and feature matching losses with weights set to 5, while the gradient of the discriminator was penalized every 16-th step with the weight of R_1 penalty equal to 0.0001. Most experiments were conducted at the resolution of 256 on the smallest side.

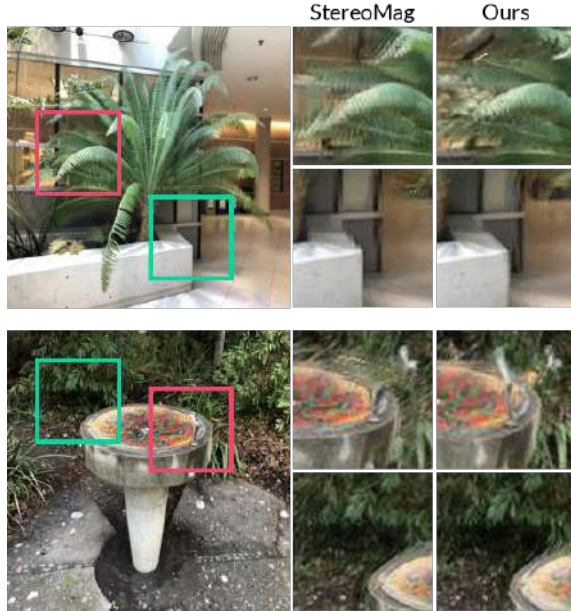


Figure 4. Comparison on challenging scenes from the LLFF [21]. The leftmost column shows the ground truth, two other columns demonstrate patches of a novel view obtained with StereoMag-32 and our system respectively. In the cutout StereoMag results, small translations of a camera from the reference pose reveal discontinuities in the approximated scene geometry leading to ghosting artefacts. In our case, thanks to the scene-adapted geometry, ghosting is not so apparent.

Metrics. We follow the standard evaluation process for novel view task and measure how similar the synthesized view is to the ground-true image. Therefore, we compute peak signal-to-noise ratio (PSNR), structural (SSIM) and perceptual (LPIPS [40]) similarity, as well as the recently introduced FLIP [2] metric between the obtained rendering and the ground truth. Both our approach and StereoMag produce strong artefacts near the image boundaries, though the form of artefacts are different. We therefore exclude the near-boundary areas from consideration by computing metrics over the central crops.

Finally, to measure the plausibility of rendered images, we perform the study of human preference on a crowdsourcing platform. The evaluation protocol was as follows: the assessors were shown two short videos with the virtual camera moving along the predefined trajectory in the same scene from SWORD (validation subset) or LLFF: one video was obtained using the baseline model, and another one was produced with our approach. We asked the users which of the two videos looked more realistic to them. In total, we generated 280 pairs of videos (120 from LLFF and 160 from SWORD scenes), and twenty different workers assessed each pair.

4.3. Main results

Ablation results. In Tab. 2 we present the relative performance for several schemes of depth estimation (denoted as GC, SA and BI) and mesh texturing (RSBg, RBg, RAW). For this ablation, all systems were trained and evaluated on the SWORD dataset. As the results show, best metrics are obtained with a combination of BI+RSBg methods. Therefore, we choose this model as our default one and refer to it as just StereoLayers model. This pipeline is used in further experiments unless another configuration is explicitly specified.

Comparison with prior art. The main results are reported in Tab. 1. Here, due to the relatively small size of both validation part of SWORD and LLFF dataset, we sampled multiple triplets (reference, side and novel cameras) for each scene in order to get a more unbiased estimation of the score. We selected the model with $L = 4$ layers as our main variant because it performs better on the hold-out LLFF data. It consistently outperforms the baseline StereoMag model of 32 planes according to the considered metrics, while containing significantly less layers.

Post-hoc coalescing StereoMag representations into non-planar layers worsened results (this finding is consistent with what is reported in [4]). Finally, StereoMag with eight planes also performed consistently worse than StereoMag with 32 planes.

At the same time, the results of our model with four layers is even better than when using more (eight) layers. Furthermore, a configuration with just two layers remain competitive (better than eight layer configuration in some metrics and better than StereoMag-32 in most metrics).

Notably, there is a rather big gap between our configuration with four layers on one hand and the StereoMag with geometry merged into four layers (StereoMag-P4) on the other. This emphasizes the benefit of the end-to-end training used by our method. As showcased in Fig. 3, the novel approach approximates the scene geometry in a reasonable way even with just two layers and, vice versa, is able to ‘zero out’ the redundant layers. We attribute the superiority of our method over StereoMag-P to the proposed end-to-end training procedure. Also, we conducted a separate study of model’s sensitivity to the group size when estimating the scene geometry (i.e., the ratio of the number of planes in PSV P to the number of layers L). Shortly, the quality does not change dramatically under the variations in the group size; see Supplementary for numerical details.

In the supplementary video, we show results on a wide range of photos from different datasets. The video contains a comparison with StereoMag and IBRNet, that demonstrates that our approach produces less blurry details, while has a similar quality of estimated geometry. We encourage the reader to watch the supplementary video.

We show the percentage of times users prefer each method in Tab. 3. The user study proves the superiority of the proposed approach. One of our qualitative improvements is illustrated in Fig. 4: the deformable layers successfully overcome the “ghost” edges artifacts, occasionally observed in case of rigid planes.

Cross-dataset evaluation. As mentioned above, SWORD contains mostly outdoor scenes with a central object, which is similar in nature to the LLFF dataset. It is the main reason why we observe a pretty good quality of the model trained on the SWORD and evaluated on LLFF (the rightmost part in Tab. 1). We have also investigated a more challenging setting: the performance of methods in the cross-dataset setting is reported in Tab. 4: we cross-evaluate our and baseline models on RealEstate10k and SWORD datasets that are rather different. This experiment suggests that our model is more robust to the out-of-domain data.

Timings. The representations produced by our method are well-suited for rendering within mobile photography applications. Thus on Samsung Galaxy S20 (Mali-G77 GPU), rendering our representations at 512×256 resolution runs at about 180 frames per second. Furthermore, our representations can be quickly created from new stereopairs (our current unoptimized inference takes 0.19 seconds on an NVidia P40 GPU).

5. Summary and discussion

In this work, we proposed an end-to-end pipeline that recovers the scene geometry from an input stereopair using a fixed number of semi-transparent layers. Despite using fewer layers (4 against 32 for the baseline StereoMag model), our approach demonstrated superior quality in terms of commonly used metrics for the novel view synthesis problem as well as human evaluation. Unlike the StereoMag system, the quality of which heavily depends on the number of planes, our method has reached better scores for novel views synthesis problem while being robust to reducing the number of layers. We have verified that the proposed method can be trained on multiple datasets, generalizes well to unseen data, and can be applied at a higher resolution at test time. The resulting mesh geometry can be effectively rendered using standard graphics engines, making the approach attractive for mobile 3D photography. At the same time, while being efficient, our scene representation has only limited ability to represent view-dependent effects, and an extension similar to NeX [39] would be required to improve this ability.

Additionally, we presented a new challenging SWORD dataset, which contains cluttered scenes with heavily occluded regions. Even though SWORD consists of fewer scenes than the popular RealEstate10K dataset, systems

trained on SWORD are likely to generalize better to other datasets, *e.g.* the LLFF dataset.

References

- [1] Kara-Ali Aliev, Artem Sevastopolsky, Maria Kolos, Dmitry Ulyanov, and Victor Lempitsky. Neural point-based graphics. In *Proc. ECCV*, 2020. 2
- [2] Pontus Andersson, Jim Nilsson, Tomas Akenine-Möller, Magnus Oskarsson, Kalle Åström, and Mark D. Fairchild. FLIP: A Difference Evaluator for Alternating Images. In *Proc. ACM SIGGRAPH*, 2020. 7
- [3] Benjamin Attal, Selena Ling, Aaron Gokaslan, Christian Richardt, and James Tompkin. Matryodshka: Real-time 6DoF video view synthesis using multi-sphere images. In *Proc. ECCV*, 2020. 2
- [4] Michael Broxton, John Flynn, Ryan Overbeck, Daniel Erickson, Peter Hedman, Matthew Duvall, Jason Dourgarian, Jay Busch, Matt Whalen, and Paul Debevec. Immersive light field video with a layered mesh representation. In *Proc. ACM SIGGRAPH*, 2020. 1, 2, 3, 4, 7, 8, 11
- [5] Shenchang Eric Chen and Lance Williams. View interpolation for image synthesis. In *Proc. ACM SIGGRAPH*, 1993. 2
- [6] R. T. Collins. A space-sweep approach to true multi-image matching. In *Proc. CVPR*, 1996. 3
- [7] Paul E. Debevec, Camillo J. Taylor, and Jitendra Malik. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. In *Proc. ACM SIGGRAPH*, 1996. 2
- [8] John Flynn, Michael Broxton, Paul Debevec, Matthew Duvall, Graham Fyffe, Ryan Styles Overbeck, Noah Snavely, and Richard Tucker. Deepview: High-quality view synthesis by learned gradient descent. In *Proc. CVPR*, 2019. 2
- [9] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Proc. NeurIPS*, 2014. 6
- [10] Steven J Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F Cohen. The lumigraph. In *Proc. ACM SIGGRAPH*, 1996. 2
- [11] Peter Hedman, Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel Brostow. Deep blending for free-viewpoint image-based rendering. In *Proc. ACM SIGGRAPH*, 2018. 1, 2
- [12] Peter Hedman, Tobias Ritschel, George Drettakis, and Gabriel J. Brostow. Scalable inside-out image-based rendering. *ACM Trans. Graph.*, 2016. 2
- [13] Ronghang Hu, Nikhila Ravi, Alexander C. Berg, and Deepak Pathak. Worldsheets: Wrapping the world in a 3d sheet for view synthesis from a single image. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 12528–12537, October 2021. 3, 14
- [14] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *Proc. ECCV*, 2016. 5
- [15] Leif Kobbelt and Mario Botsch. A survey of point-based techniques in computer graphics. *Computers & Graphics*, 2004. 2
- [16] Samuli Laine, Janne Hellsten, Tero Karras, Yeongho Seol, Jaakko Lehtinen, and Timo Aila. Modular primitives for high-performance differentiable rendering. *ACM Transactions on Graphics*, 39(6), 2020. 1, 5
- [17] Kai-En Lin, Zexiang Xu, Ben Mildenhall, Pratul P. Srinivasan, Yannick Hold-Geoffroy, Stephen DiVerdi, Qi Sun, Kalyan Sunkavalli, and Ravi Ramamoorthi. Deep multi depth panoramas for view synthesis. In *Proc. ECCV*, 2020. 1, 2, 3
- [18] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. In *NeurIPS*, 2020. 2
- [19] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. In *Proc. ACM SIGGRAPH*, 2019. 2
- [20] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which Training Methods for GANs do actually Converge? In *Proc. ICML*, 2018. 6
- [21] Ben Mildenhall, Pratul P. Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. In *SIGGRAPH*, 2019. 1, 2, 5, 6, 7, 14
- [22] 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 *Proc. ECCV*, 2020. 2
- [23] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral Normalization for Generative Adversarial Networks. In *ICLR*, 2018. 11
- [24] Simon Niklaus, Long Mai, Jimei Yang, and Feng Liu. 3d ken burns effect from a single image. 2019. 1
- [25] Thomas Porter and Tom Duff. Compositing digital images. In *Proc. ACM SIGGRAPH*, 1984. 4, 5
- [26] Gernot Riegler and Vladlen Koltun. Free view synthesis. In *Proc. ECCV*, 2020. 1, 2
- [27] J. L. Schönberger and J. Frahm. Structure-from-motion revisited. In *Proc. CVPR*, 2016. 5
- [28] M. L. Shih, S. Y. Su, J. Kopf, and J. B. Huang. 3d photography using context-aware layered depth inpainting. In *Proc. CVPR*, 2020. 1, 2, 6, 7
- [29] Vincent Sitzmann, Justus Thies, Felix Heide, Matthias Nießner, Gordon Wetzstein, and Michael Zollhöfer. Deepvoxels: Learning persistent 3d feature embeddings. In *Proc. CVPR*, 2019. 2
- [30] P. P. Srinivasan, R. Tucker, J. T. Barron, R. Ramamoorthi, R. Ng, and N. Snavely. Pushing the boundaries of view extrapolation with multiplane images. In *Proc. CVPR*, 2019. 1, 2, 14
- [31] Richard Szeliski and Polina Golland. Stereo matching with transparency and matting. 1999. 1, 2
- [32] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *Proc. ECCV*, 2020. 6, 12

- [33] Ayush Tewari, Ohad Fried, Justus Thies, Vincent Sitzmann, Stephen Lombardi, Kalyan Sunkavalli, Ricardo Martin-Brualla, Tomas Simon, Jason Saragih, Matthias Nießner, et al. State of the art on neural rendering. In *Computer Graphics Forum*, 2020. 1, 2
- [34] Justus Thies, Michael Zollhöfer, and Matthias Nießner. Deferred neural rendering: Image synthesis using neural textures. In *Proc. ACM SIGGRAPH*, 2019. 1, 2
- [35] R. Tucker and N. Snavely. Single-view view synthesis with multiplane images. In *Proc. CVPR*, 2020. 1
- [36] Shubham Tulsiani, Richard Tucker, and Noah Snavely. Layer-structured 3d scene inference via view synthesis. In *Proc. ECCV*, 2018. 2
- [37] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul P. Srinivasan, Howard Zhou, Jonathan T. Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. Ibrnet: Learning multi-view image-based rendering. In *CVPR*, pages 4690–4699, June 2021. 2, 7
- [38] O. Wiles, G. Gkioxari, R. Szeliski, and J. Johnson. Synsin: End-to-end view synthesis from a single image. In *Proc. CVPR*, 2020. 1, 2, 3, 6, 11
- [39] Suttisak Wizadwongsa, Pakkapon Phongthawee, Jiraphon Yenphraphai, and Supasorn Suwajanakorn. Nex: Real-time view synthesis with neural basis expansion. In *CVPR*, 2021. 8
- [40] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proc. CVPR*, 2018. 7
- [41] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. In *Proc. ACM SIGGRAPH*, 2018. 1, 2, 3, 5, 6, 7, 11
- [42] C. Lawrence Zitnick, Sing Bing Kang, Matthew Uyttendaele, Simon Winder, and Richard Szeliski. High-quality video view interpolation using a layered representation. In *Proc. ACM SIGGRAPH*, 2004. 2

Appendices

A. MPI postprocessing

In this section we briefly describe the postprocessing procedure, that aims to merge the predicted rigid planes of StereoMag-32 [41] to the less number of deformable layers. In our experiments the final number of such layers equals 8, that coincides with the basic configuration of our approach.

The pipeline partially follows to the one described in [4]. Firstly, we divide 32 planes into eight groups and compose over the depth within each group on top of the furthest plane in the group. This operation results in 8 deformable layers. To infer the textures of those layers, we perform the second step, averaging the color \mathbf{c} and transmittance $\bar{\alpha}$ of RGBA planes within each group over the set $V(t)$ of rays passing through the texel t . Namely, we run the Monte Carlo ray tracing defined by the equations below,

$$\log(\bar{\alpha}_t) = \lambda^{-1} \int_{V(t)} w(\mathbf{r}) [\log(\bar{\alpha}_{\mathbf{r}})]^2 d\mathbf{r},$$

$$\mathbf{c}_t = \lambda^{-1} \int_{V(t)} w(\mathbf{r}) \mathbf{c}_{\mathbf{r}} \log(\bar{\alpha}_{\mathbf{r}}) d\mathbf{r},$$

where λ is a normalizing constant

$$\lambda = \int_{V(t)} w(\mathbf{r}) \log(\bar{\alpha}_{\mathbf{r}}) d\mathbf{r}.$$

The distribution of rays $\mathbf{r} \in V(t)$ is constructed as follows: the line passing through the pinhole camera and texel t intersects the reference image plane at the pixel coordinate p . The coordinate q is normally distributed around p , and the ray \mathbf{r} passes from q through t . The weighing function $w(\mathbf{r})$ equals the Gaussian density value at q . The color $\mathbf{c}_{\mathbf{r}}$ and transmittance $\bar{\alpha}_{\mathbf{r}}$ values are computed with the compose-over operation along the ray \mathbf{r} over the planes that belong to the same group as the texel t does.

B. Network architectures

Geometry network F_g . The architecture of our depth estimator resembles the network from SynSin [38]. It takes the plane-sweep volume (PSV) as its input and returns ‘opacities’ for each of the P regular planes, that are used to construct deformable layers. Each block sequentially applies a convolution, layer normalization and LeakyReLU to the input tensor. We apply spectral normalization [23] to the convolution kernel weights. Other details are given in Tab. 5.

Coloring network F_c . The architecture of coloring network is inspired by the one described StereoMag paper [41]. Each block consists of a convolution, layer normalization and ReLU unit (except for the final block). Detailed parameters for RSBg scheme are provided in Tab. 6.

Block	K	S	D	P	C	Input
Conv1.1	4	2	1	1	32	PSV
Conv1.2	4	2	1	1	64	Conv1.1
Conv1.3	4	2	1	1	128	Conv1.2
Conv2.1	4	2	1	1	256	Conv1.3
Conv2.2	4	2	1	1	256	Conv2.1
Conv2.3	4	2	1	1	256	Conv2.2
Conv2.4	4	2	1	1	256	Conv2.3
Conv2.5	4	2	1	1	256	Conv2.4
Conv3.1	3	1	1	1	256	Conv2.5↑
Conv3.2	3	1	1	1	256	concat[Conv3.1, Conv2.4]↑
Conv3.3	3	1	1	1	256	concat[Conv3.2, Conv2.3]↑
Conv3.4	3	1	1	1	256	concat[Conv3.3, Conv2.2]↑
Conv4.1	3	1	1	1	128	concat[Conv3.4, Conv2.1]↑
Conv4.2	3	1	1	1	64	concat[Conv4.1, Conv1.3]↑
Conv4.3	3	1	1	1	32	concat[Conv4.2, Conv1.2]↑
Conv4.4	3	1	1	1	P	concat[Conv4.3, Conv1.1]↑

Table 5. Architecture of geometry network F_g for BI parameterization. K is the kernel size, S – stride, D – dilation, P – padding, C – the number of output channels for each layer, and *input* denotes the input source of each layer. Up-arrow ↑ denotes the 2x bilinear upscaling operation.

Block	K	S	D	P	C	Input
Conv1.1	3	1	1	1	64	deformed PSV
Conv1.2	3	2	1	1	128	Conv1.1
Conv2.1	3	1	1	1	128	Conv1.2
Conv2.2	3	2	1	1	256	Conv2.1
Conv3.1	3	1	1	1	256	Conv2.2
Conv3.2	3	1	1	1	512	Conv3.1
Conv3.3	3	2	1	1	512	Conv3.2
Conv4.1	3	1	2	2	512	Conv3.3
Conv4.2	3	1	2	2	512	Conv4.1
Conv4.3	3	1	2	2	512	Conv4.2
TransConv5.1	4	2	1	1	256	concat[Conv4.3, Conv3.3]
TransConv5.2	3	1	1	1	256	TransConv5.1
TransConv5.3	3	1	1	1	256	TransConv5.2
TransConv6.1	4	2	1	1	128	concat[TransConv5.3, Conv2.2]
TransConv6.2	3	1	1	1	128	TransConv6.1
TransConv7.1	4	2	1	1	64	concat[TransConv6.2, Conv1.2]
TransConv7.2	3	1	1	1	64	TransConv7.1
Conv7.3	1	1	1	0	$4L+3$	TransConv7.2

Table 6. Architecture of coloring network F_c for RSBg parameterization. K is the kernel size, S – stride, D – dilation, P – padding, C – the number of output channels for each layer, and *input* denotes the input source of each layer.

C. Occlusion masks

In this section we describe the heuristic for creating masks of occluded regions. The examples of such masks are provided in Fig. 5.

C.1. Cycle consistency of optical flows

Consider two images A and B , without the loss of generality they are assumed to be grayscale. For the pixel’s coordinates p we denote the color of this pixel in the image A as $A[p]$. The coordinate grid G is such an “image”

(two-dimensional matrix) that $\forall p G[p] = p$. We define the backward flow \overleftarrow{F}_{AB} of images A and B and the backward warping operation as follows

$$B = \text{backward}\left(A, \overleftarrow{F}_{AB}\right) \iff \forall q B[q] = A\left[\overleftarrow{F}_{AB}[q]\right]. \quad (2)$$

Similarly, the forward flow \overrightarrow{F}_{AB} and forward warping are defined as

$$B = \text{forward}\left(A, \overrightarrow{F}_{AB}\right) \iff \forall p A[p] = B\left[\overrightarrow{F}_{AB}[p]\right]. \quad (3)$$

Lemma C.1. For two optical flows of the same kind F_{AB} and F_{BA} the following cycle-consistency property holds

$$\text{backward}(F_{BA}, F_{AB}) = G.$$

Proof. We assume that the pixel p of the image A corresponds to the pixel q of the image B under the warping operation. This implies the following equations:

$$B[q] = A[p], \quad (4)$$

$$\overleftarrow{F}_{AB}[q] \stackrel{(2)}{=} p, \quad (5)$$

$$\overrightarrow{F}_{AB}[p] \stackrel{(3)}{=} q. \quad (6)$$

By a symmetry argument, we obtain also

$$\overleftarrow{F}_{BA}[p] \stackrel{(5)}{=} q, \quad (7)$$

$$\overrightarrow{F}_{BA}[q] \stackrel{(6)}{=} p. \quad (8)$$

Let X be the result of warping one backward flow with another,

$$X = \text{backward}\left(\overleftarrow{F}_{BA}, \overleftarrow{F}_{AB}\right).$$

From the definition,

$$X[q] \stackrel{(2)}{=} \overleftarrow{F}_{BA}\left[\overleftarrow{F}_{AB}[q]\right] \stackrel{(5)}{=} \overleftarrow{F}_{BA}[p] \stackrel{(7)}{=} q,$$

therefore, $X = G$.

The case of forward flows may be considered in the same way. Denote the result of warping with Y ,

$$Y = \text{backward}\left(\overrightarrow{F}_{BA}, \overrightarrow{F}_{AB}\right).$$

The value at pixel p gives us the following

$$Y[p] \stackrel{(2)}{=} \overrightarrow{F}_{BA}\left[\overrightarrow{F}_{AB}[p]\right] \stackrel{(6)}{=} \overrightarrow{F}_{BA}[q] \stackrel{(8)}{=} p,$$

which leads to $Y = G$. \square



Figure 5. Occlusion masks, obtained with a pretrained optical flow estimator and our heuristic. Left: reference images; middle: generated novel views; right: magenta masks indicate the parts of novel views that were occluded from the reference point of view. The area of such regions in SWORD is much greater than for RealEstate10k, justifying its usage.

C.2. Estimation of occlusion masks

We employ the pretrained optical flow estimator [32] and compute optical flows \hat{F}_{rn} and \hat{F}_{nr} between the reference view I_r and ground-true novel view I_n . According to the lemma C.1, these flows should be cycle-consistent. However, the views do not correspond to each other completely because of the presence of occluded regions. Therefore, the result \hat{G} of warping of one flow with another

$$\hat{G} = \text{backward}\left(\hat{F}_{rn}, \hat{F}_{nr}\right)$$

does not result in the “ideal” coordinate grid.

Based on this, we treat such pixel p that $|\hat{G}[p] - p| < \epsilon$ as *non-occluded* because the optical flow estimator is able to find the corresponding pixel in another image. Otherwise, we include the pixel to the occlusion mask. The threshold ϵ is set to the size of one pixel. As a downside, the flow estimator is very sensitive to the image borders. To overcome this issue, we use central crops that finally contain reasonable masks.

D. Failure cases

To demonstrate the limitations of our approach, we show the typical artefacts of the method in Fig. 6. Note that most of the drawbacks are visible only when the camera moves around the scene and are not distinguishable in randomly selected frames without temporal context.

When the baseline is magnified by a great factor, one can observe “stretching” faces of our layered mesh near the depth discontinuities. We believe this type of artefact is caused by the mesh structure of our geometry. The “ghost”



Figure 6. Examples of most common failures of StereoLayers outputs. In most cases they can be attributed to a combination of photometric scene complexity, and an unfortunate choice of the input pair.

semi-transparent textures is another common issue of the synthesized views. One of the problems could also be attributed to the inconsistent in depth prediction when some pixels have minor errors with depth values, which leads to small ghostings.

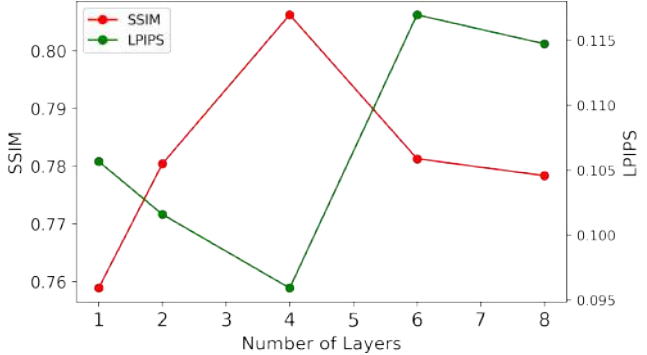


Figure 7. Performance of our system as a function of the number of layers. The plot confirms the ability of our approach to represent complex scenes with just a few layers.

Model	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	∇ LIP \downarrow
IBRNet	27.4	0.67	0.219	0.27
StereoMag (256 \rightarrow 512)	23.3	0.65	0.178	0.19
Ours (256 \rightarrow 512)	24.2	0.69	0.155	0.19

Table 7. Scaling to higher resolution on SWORD dataset. We examine our model and StereoMag in a fully-convolutional regime: both were trained at resolution of 256×512 and applied for 512×1024 . As in previous experiments, we used the checkpoint of IBRNet provided by the authors of the corresponding paper.

E. Additional results

Scaling to hi-res. To investigate the scaling properties of our StereoLayers model, we additionally compared it with the baselines on high-resolution versions of datasets, described in the main text. Tab. 7 presents the results of the trained network, applied to higher resolution in fully-convolutional manner. It outperforms StereoMag operating in the same regime by a significant margin. Additionally we compare the quality with the original IBRNet. This model achieves the best PSNR value and simultaneously the worst LPIPS. This is caused by the inconsistency in the generated frames. Please see examples of such behaviour in the supplementary video.

Besides that, we conducted a user study on 80 scenes from SWORD (with resolution of 512×1024), 60 scenes from RealEstate10k (576×1024) and 80 scenes (40 unique) from LLFF data (512×512). All scenes and input views are randomly sampled from the test sets. Results of this experiment are reported in Tab. 8.

StereoMag with RSBg scheme. As was shown in the Tab. 2 of the main text, our model trained with the RBg texturing scheme (which is default for StereoMag) performs significantly worse than with RSBg: LPIPS of 0.111 vs 0.096. To demonstrate that texturing scheme is not the most crucial part of our pipeline, we re-trained StereoMag-32

Dataset	Baseline	Our score, %	p -value
SWORD	StereoMag-32	55.62	< 0.001
	IBRNet	75.69	< 0.001
LLFF	StereoMag-32	54.42	< 0.001
	IBRNet	50.27	< 0.001
RealEstate10k	StereoMag-32	63.91	< 0.001
	IBRNet	60.74	< 0.001

Table 8. Additional user study on high-resolution images. The 3rd column contains the ratio of users who selected the output of our model as more realistic under the two-alternative forced choice.

model with RSBg scheme. Notably, this modification did not improve the quality of the baseline on SWORD: SSIM of 0.77 vs 0.76, LPIPS of 0.107 vs 0.107.

Scene slices. Fig. 8 provides additional examples of estimated geometry for different scenes.

Group size P/L	Number of planes P	Number of layers L	LPIPS↓	SSIM↑
4	16	4	0.129	0.67
4	24	6	0.120	0.70
4	32	8	0.119	0.70
4	40	10	0.124	0.70
16	64	4	0.122	0.72
32	64	2	0.121	0.71
15	120	8	0.119	0.70
20	120	6	0.122	0.70
30	120	4	0.120	0.70
60	120	2	0.119	0.70

Table 9. Performance dependence on the number of layers and the size of the plane group for group compositing (GC) configuration. The quality in terms of SSIM and LPIPS is slightly dependent on the size of the group and the number of layers for 256×256 images.

Number of layers in BI scheme. For MPI-based approaches the number of planes was shown to be critical for constructing a plausible representation of the scene [21, 30]. To demonstrate the properties of our deformable layers, we consider the influence of the number of layers in the estimated geometry on common quality metrics. Fig. 7 shows that the resulting performance falls as the number of layers decreases to one, proving that multi-layer structure is crucial. Perhaps surprisingly, the measured quality does not always grow as this number is increasing. We suggest that the model cannot handle the redundant geometry properly. It is worth noting that the authors of Worldsheet paper reported about a similar effect in the single-image case [13].

Number of layer per group in GC scheme. In addition

to our main *bounds interpolation* (BI) scheme of depth parameterization, we study the properties of *group compositing* (GC) model. Namely, we investigate the performance of this system as a function of the number of planes in plane-sweep volume during the geometry estimation step. As Tab. 9 shows, the resulting quality of the model does not depend on the size of the group. However, we see that if both the number of layers and the size of the group are reduced simultaneously, the quality deteriorates. And with an increase in the size of the group, there is no increase in metrics. In general, robustness to these parameters is provided by two points: the nature of the multi-transparent proxy geometry, in which alpha takes the main responsibility for object structure, and the adaptive layered proxy geometry, which can bend itself under objects to depend less on the number of planes.



Figure 8. Additional horizontal slices (along the blue line) on scenes from LLFF dataset. Mesh vertices are shown as dots with the predicted opacity. Colors encode the layer number. The horizontal axis corresponds to the pixel coordinate, while the vertical axis stands for the vertex depth w.r.t. the reference camera (only the most illustrative depth range is shown). Configurations of StereoLayers method generate scene-adaptive geometry in a more efficient way than StereoMag, resulting in more frugal geometry representation, while also obtaining better rendering quality.