Stereo Magnification with Multi-Layer Images

Supplementary Materials

https://samsunglabs.github.io/StereoLayers/

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 S1. Architecture of the 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 S2. Architecture of the coloring network F_c for the 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.

S1. Network architectures

Geometry network F_g. The architecture of our depth estimator resembles the network from SynSin [7]. 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 [4] to the convolution kernel weights. Other details are given in Tab. S1.

Coloring network F_c. The architecture of the coloring network is inspired by the one described in StereoMag paper [8]. 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. S2.

S2. 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. S3 presents the results of the trained network, applied to higher resolution in a 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 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. The results of this experiment are reported in Tab. S4.

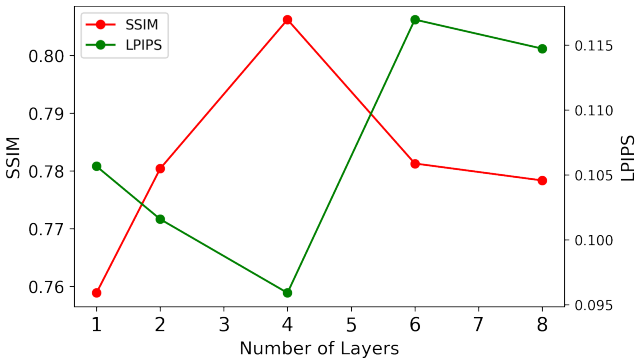

Figure S1. Performance of our system as a function of the number of layers. The plot confirms the ability of our approach to represent complex scenes with just a few layers.

Model	PSNR ↑	SSIM ↑	LPIPS ↓	ƎLIP ↓
IBRNet	**27.4**	0.67	0.219	0.27
StereoMag (256→512)	23.3	0.65	0.178	**0.19**
Ours (256→512)	24.2	**0.69**	**0.155**	**0.19**

Table S3. 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.

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 S4. 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.

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 the default for StereoMag) performs significantly worse than with RSBg: LPIPS of 0.111 vs 0.096. To demonstrate that the texturing scheme is not the most crucial part of our pipeline, we retrained StereoMag-32 model with RSBg scheme. In particular, 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. S2 provides additional examples of the estimated geometry for different scenes.

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 [3, 5]. 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. S1 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 increases. We suggest that the model cannot handle the redundant geometry properly. It is worth noting that the authors of the Worldsheet paper reported a similar effect in the single-image case [2].

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 S5. 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 per group in GC scheme. In addition to our main *bounds interpolation* (BI) scheme of depth parameterization, we study the properties of the *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. S5 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 semitransparent proxy geometry, in which the alpha channel takes the main responsibility for the object structure, and the adaptive layered proxy geometry, which can bend itself under objects to depend less on the number of planes.

S3. Failure cases

To demonstrate the limitations of our approach, we show typical artifacts of the method in Fig. S3. Note that most

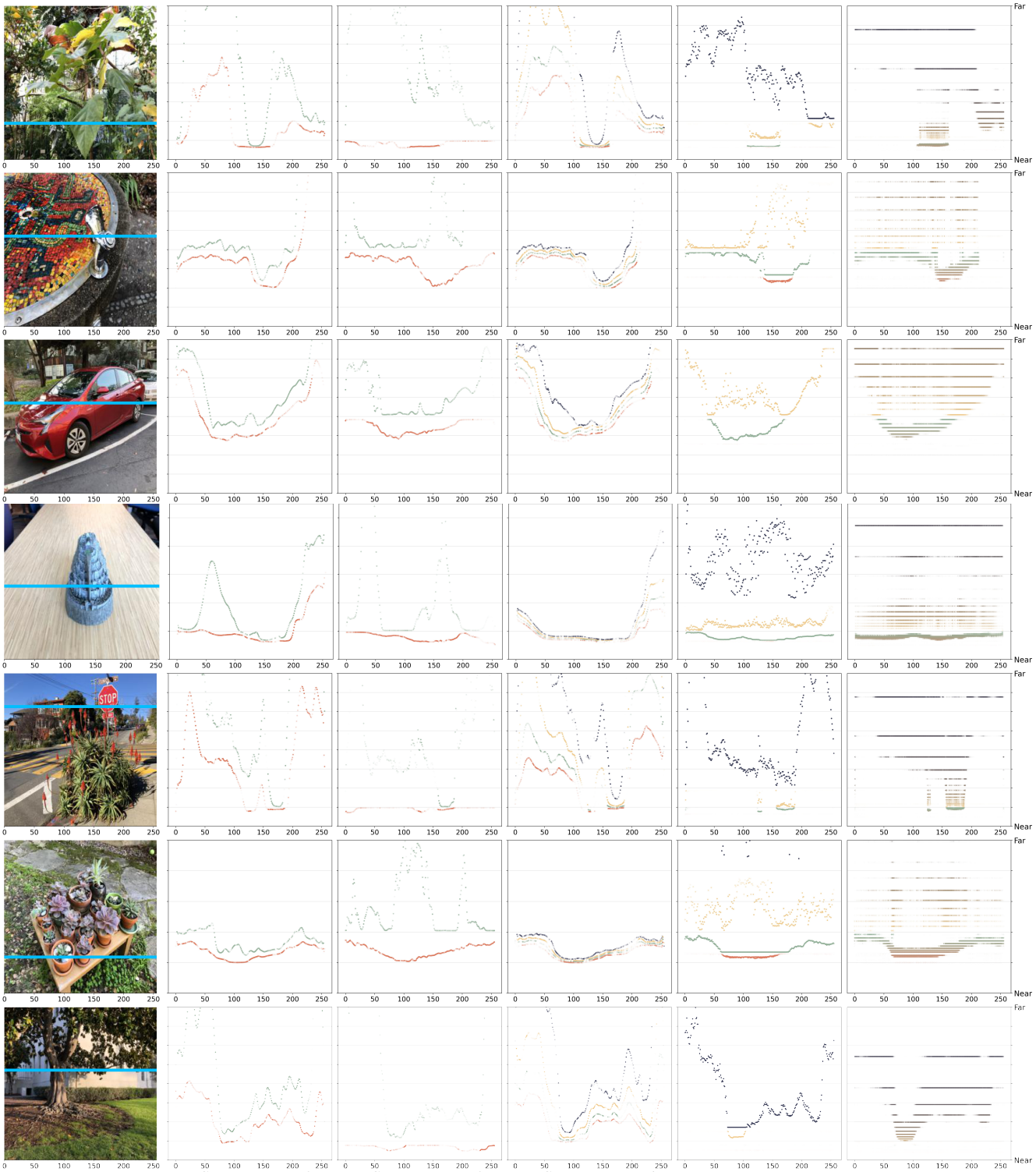

Figure S2. Additional horizontal slices (along the blue line) on scenes from LLFF dataset. Mesh vertices are shown as dots with the predicted opacity. Colors encode the layer number. The horizontal axis corresponds to the pixel coordinate, while the vertical axis stands for the vertex depth w.r.t. the 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.

Figure S3. 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.

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 that this type of artifact is caused by the mesh structure of our geometry. The "ghost" semitransparent textures is another common issue of the synthesized views. One of the problems could also be attributed to inconsistent depth prediction when some pixels have minor errors in depth values, which leads to small ghostings.

S4. MPI postprocessing

In this section we briefly describe the postprocessing procedure that aims to merge the predicted rigid planes of StereoMag-32 [8] to the fewer 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 the one described in [1]. 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,

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

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 $w(\mathbf{r})$ is equal to the Gaussian density value at q. 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 texel t does.

S5. Occlusion masks

In this section, we describe the heuristic to create masks of occluded regions. Examples of such masks are provided in Fig. S4.

S5.1. Cycle consistency of optical flows

Consider two images A and B, without loss of generality, they are assumed to be grayscale. For the coordinates of the pixel p we denote the color of this pixel in the image

A as $A[p]$. The coordinate grid G is such a "image" (two-dimensional matrix) that $\forall p\, G[p] = p$. We define the backward flow matrix $\overleftarrow{F}_{AB}$ of images A and B and the backward warping `backward` operation as follows

$$B = \texttt{backward}\left(A, \overleftarrow{F}_{AB}\right) \iff \forall q\, B[q] = A\left[\overleftarrow{F}_{AB}[q]\right]. \quad \text{(S1)}$$

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

$$B = \texttt{forward}\left(A, \overrightarrow{F}_{AB}\right) \iff \forall p\, A[p] = B\left[\overrightarrow{F}_{AB}[p]\right]. \quad \text{(S2)}$$

Lemma S5.1. *For two optical flows of the same kind F_{AB} and F_{BA} the following cycle-consistency property holds*

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

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:

$$B[q] = A[p], \quad \text{(S3)}$$

$$\overleftarrow{F}_{AB}[q] \overset{(S1)}{=} p, \quad \text{(S4)}$$

$$\overrightarrow{F}_{AB}[p] \overset{(S2)}{=} q. \quad \text{(S5)}$$

By a symmetry argument, we also obtain

$$\overleftarrow{F}_{BA}[p] \overset{(S4)}{=} q, \quad \text{(S6)}$$

$$\overrightarrow{F}_{BA}[q] \overset{(S5)}{=} p. \quad \text{(S7)}$$

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

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

From the definition,

$$X[q] \overset{(S1)}{=} \overleftarrow{F}_{BA}\left[\overleftarrow{F}_{AB}[q]\right] \overset{(S4)}{=} \overleftarrow{F}_{BA}[p] \overset{(S6)}{=} q,$$

therefore, $X = G$.

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

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

The value in the pixel p gives us the following

$$Y[p] \overset{(S1)}{=} \overrightarrow{F}_{BA}\left[\overrightarrow{F}_{AB}[p]\right] \overset{(S5)}{=} \overrightarrow{F}_{BA}[q] \overset{(S7)}{=} p,$$

which leads to $Y = G$. □

Figure S4. 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.

S5.2. Estimation of occlusion masks

We employ the pretrained optical flow estimator [6] 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 S5.1, these flows should be cycle-consistent. However, the views do not completely correspond to each other because of the presence of occluded regions. Therefore, the result $\hat{G}$ of warping of one flow with another

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

does not result in the "ideal" coordinate grid.

Based on this, we treat a pixel p that $|\hat{G}[p] - p| < \epsilon$ as *non-occluded* because the optical flow estimator can find the corresponding pixel in another image. Otherwise, we include the pixel in 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.

References

[1] 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. S4

[2] Ronghang Hu, Nikhila Ravi, Alexander C. Berg, and Deepak Pathak. Worldsheet: 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. S2

[3] 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. S2

[4] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral Normalization for Generative Adversarial Networks. In *ICLR*, 2018. S1

[5] 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. S2

[6] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *Proc. ECCV*, 2020. S5

[7] O. Wiles, G. Gkioxari, R. Szeliski, and J. Johnson. Synsin: End-to-end view synthesis from a single image. In *Proc. CVPR*, 2020. S1

[8] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. In *Proc. ACM SIGGRAPH*, 2018. S1, S4