Flash3D: Feed-Forward Generalisable 3D Scene Reconstruction from a Single Image
Supplementary Material

1. Dataset details

RealEstate10k. We download the videos from provided links, resulting in above 65,000 videos, as well as the provided camera pose trajectories. Using the provided cameras, we run sparse point cloud reconstruction with COLMAP [11]. We use the test split provided by MINE, and following prior work we evaluate PSNR on novel frames which are 5 and 10 frames ahead of the source frame. In addition, we evaluate on a random frame sampled from an interval of ± 30 frames. We use the same frames as [8] did for their evaluation. As a result, we evaluate on 3205 frames. We reproduced the results from [8] using their released checkpoint with the common protocol of cropping 5% of the image around the border, achieving scores similar to those presented in the original paper. We confirmed with authors of BTS [17] that this is the commonly used protocol. We do our training and testing at 256×384 resolution.

NYUv2. We form a benchmark that is similar in nature to RealEstate10k in that it shows indoor scenes, but is visually radically different. We download 80 raw sequences of NYUv2 [12] and run COLMAP [11] on them to recover camera pose trajectories. On each video we sample 3 random souce frames and use a random frame uniformly sampled within ± 30 frames from the source frame, mirroring the protocol of RealEstate10k. We undistort images, and rescale to 256×384 resolution.

KITTI We evaluate on the Tulsiani test split [15] of the KITTI [2] dataset. The cameras in the KITTI dataset are in metric scale, our network works directly with the provided cameras and scenes without additional preprocessing. For evaluation, following prior work [8, 17] we crop the outer 5% of the images.

2. Baselines and competing methods

2.1. Depth unprojection

A crucial baseline in our experiments is measuring performance of monocular depth prediction for monocular Novel View Synthesis. In this baseline, we place isotropic 3D Gaussians with fixed opacity without view-dependent effects (i.e. a point cloud with soft point boundaries) at the depths predicted by the monocular depth predictor. We set the Gaussian colours to be a scaled copy from the input view so that $c_G = \alpha c_{RGB}$ and we initialise $\alpha = 1.0$. We initialise Gaussian opacity to be $\sigma = \text{sigmoid}(\sigma_0)$, with $\sigma_0 = 4.0$, i.e., almost opaque. We test two variants of setting the scale of Gaussians: (1) one where Gaussians have a fixed scale $s = \exp s_0$ with $s_0 = -4.5$, and (2) one where the radius is proportional to depth from camera, allowing the Gaussians to fit inside the ray cast from the pixel: $s = \exp s_0 d/d_0$, where d is metric depth output from UniDepth, $d_0 = 10.0$ and $s_0 = -4.5$. Next, while we determined $\alpha = 1.0$, $s_0 = -4.5$ and $\sigma_0 = 4.0$ to be reasonable initialisations, they might not correspond to the highest quality of Novel View Synthesis. Thus, we run gradient-based optimisation of the parameters of this baseline, optimising α, s_0, σ_0 to minimise the photometric loss in the source view and 3 novel views (identical to our final model) on the training set. We train these models for $5,000$ iterations and choose the one with the best performance on validation split. Finally, we evaluate the model with the best α, s_0, σ_0 on the test split and report the metrics.

2.2. Splatter Image

We implemented the Splatter Image baseline using the same U-Net convolutional neural network with a ResNet-50 backbone as our own method for a fair comparison. We trained it on two NVIDIA A6000 GPUs for a total of $350,000$ steps, an order of magnitude more than our proposed Flash3D. Training took 6GPU days, same as reported in [13].

2.3. MINE

MINE [8] only provided model weights but no inference and evaluation code on RealEstate10K dataset, hence we re-run the inference and evaluation for reproducibility. The results match those reported in [8]. We use the $N = 64$ model since that is the best one made available by the authors. For evaluation on NYU we use the model trained on Re10k, identically to our method.

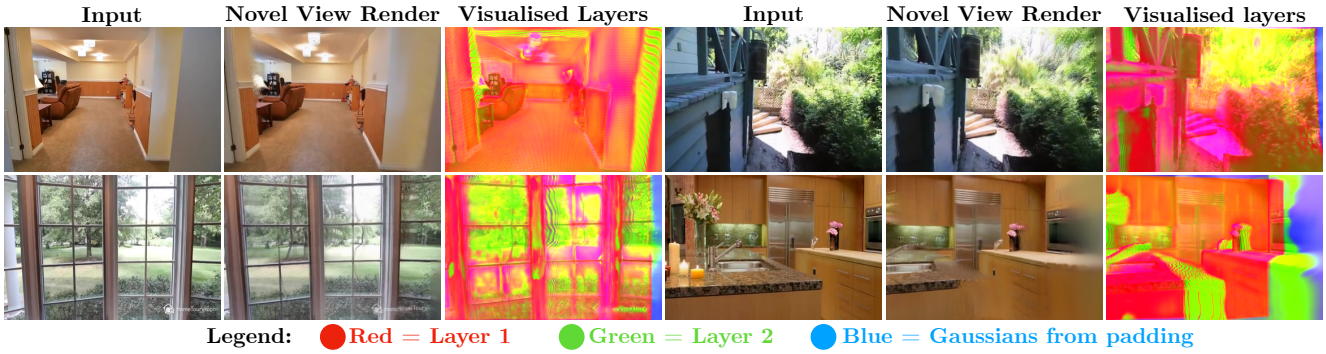

Figure 1. **Analysis of Gaussian allocation.** Gaussians from the first layer (red) are allocated in visible parts, from the second layer (green) in occluded regions (top row, bottom right) and on windows (bottom left) and Gaussians from the padding region (blue) are revealed when camera reveals regions that were not present in the frustrum of the input camera.

Table 1. **Depth Unprojection Baseline**. We fit hyperparameters of the depth unprojection model via gradient-based optimisation. We try two variants: one with fixed-size Gaussians and one where the Gaussian scale is increased proportionally to depth. Top two rows are before correcting depth-wise unprojection to be from pixel centers instead of pixel corners. All measured with croppint.

		5 frames			10 frames			random frame		
Model	Backbone	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
Fixed size	ConvNeXT-L	26.47	0.864	0.120	24.08	0.808	0.173	22.60	0.774	0.211
Fixed size	ViT-L	26.62	**0.867**	**0.120**	24.25	**0.814**	**0.172**	22.78	**0.781**	0.209
Depth-dependent	ConvNeXT-L	26.49	0.861	0.124	24.10	0.806	0.175	22.61	0.774	0.209
Depth-dependent	ViT-L	**26.65**	0.864	0.123	**24.29**	0.812	0.173	**22.80**	**0.781**	**0.207**

2.4. Two-view methods

When comparing to two-view methods, we ought to choose one of them as our source view. For any method, the most indicative factor of performance on a target frame is the baseline to the source frame. We run this comparison on 256×256 without border-cropping for being comparable.

2.5. Probability distribution of Gaussian

An alternative approach to the multiple Gaussians is to predict depth probabilities as in pixelSplat [1]. However, without the estimated depth from the pre-trained depth predictor, the coverage speed is very slow, and the performance is worse in our monocular setting. For a fair comparison, we ablate only on other Gaussian layers, *i.e.* $K > 1$ of Gaussians. The results are reported in Tab. 2. The continuous depth offset outperforms the depth probabilities design in pixelSplat.

2.6. Off-the-Shelf Depth Models

We also assess the effect of different monocular depth estimation methods. We first evaluate the recent DepthAnthing V2 [18, 19] model, which provides better details for depth prediction. However, since their metric depth is either trained only for indoor scenes (Hypersim) or outdoor scenes (KITTI), we used the indoor checkpoints as the metric depth. As shown in the Tab. 3, our framework also achieves comparable results using depths from DepthAnthing V2, without adjusting any hyper-parameters. Secondly, we evaluated our method using another recent Metric3D V2 monocular depth estimation model [6]. Similarly, the results are comparable to our main model reaffirming the choice of Unidepth [9] as the backbone in our method.

3. Implementation details

3.1. Architecture

We base our convolutional network on a ResNet-50 [5] backbone and implement a U-Net [10] encoder-decoder as in [4]. Specifically, a single ResNet encoder is shared by a multiple decoders, one for each layer of appearance parameters as well as depth offset decoders, barring the offset decoder for the first layer as we obtain depth values directly from a pre-trained model.

3.2. Optimisation

We define the photometric loss following [3] as a weighted sum of L_1 and SSIM [16] terms:

$$\mathcal{L} = \|\hat{J} - J\| + \alpha\,\mathrm{SSIM}(\hat{J}, J) \tag{1}$$

Unlike previous works [8, 14], we do not use sparse depth supervision.

Table 2. **Ablation Study for Depth Decoder Architectures.** Here, we ablate the probabilistic depth as in pixelSplat [1], but only for the $K > 1$ of Gaussians. $-K$ means K Gaussians per-pixel. Here, cross-domain (CD) denotes that the method was not trained on the dataset being evaluated.

	KITTI				NYU			
Method	CD	PSNR ↑	SSIM ↑	LPIPS ↓	CD	PSNR ↑	SSIM ↑	LPIPS ↓
Flash3D (Discrete)-2	✓	21.35	0.805	0.153	✓	24.52	0.763	0.200
Flash3D (Discrete)-3	✓	21.50	0.814	0.136	✓	24.84	0.772	0.189
Flash3D (Ours)-2	✓	**21.96**	**0.826**	**0.132**	✓	**25.09**	**0.775**	**0.182**

Table 3. **Ablations on different depth models**. We fit hyperparameters of the depth unprojection model via gradient-based optimisation. We try two variants: one with fixed-size Gaussians and one where the Gaussian scale is increased proportionally to depth. Top two rows are before correcting depth-wise unprojection to be from pixel centers instead of pixel corners. All measured with croppint.

	5 frames			10 frames			random frame		
Model	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
Unidepth V1	**28.46**	**0.899**	**0.100**	**25.94**	**0.857**	**0.133**	**24.93**	**0.833**	**0.160**
DepthAnything V2	28.31	0.895	0.101	25.79	0.849	0.136	24.49	0.823	0.165
Metric3D V2	28.00	0.893	0.107	25.62	0.852	0.140	24.55	0.826	0.167

where J is a target image, $\hat{J}$ is a rendering, and $\alpha = 0.85$. We optimise the network with Adam [7] with batch size 16 and a learning rate of 0.0001 for a total of $40,000$ training steps.

3.3. Scale alignment

Camera poses are typically estimated with COLMAP. These camera poses are in an arbitrary scale in each scene. Following prior work, we align the scale of the COLMAP cameras to those estimated by our network using the scale factor computation from [14]. However, if there are outliers in depth estimation (both in our method and baselines), they will impact the scale estimation. As a result, there might be mismatch between the scene reconstruction scale and the scale of camera poses from which novel views are rendered. In consequence, the rendered novel views can be shifted compared to ground truth, which does not significantly impact LPIPS but it does affect PSNR. Thus, at test-time we run scale alignment with RANSAC. We do the same for MINE when evaluating it on the transfer dataset, NYU, since the accuracy of its depth prediction deteriorates in this unseen dataset. When estimating scale we thus use the RANSAC scheme with sample size of 5, $1,000$ iterations and threshold 0.1.

4. Limitations

A primary limitation of the proposed approach is due to it being a deterministic, regressive model. This incentivises it to generate blurry renderings in presence of ambiguity, such as when baselines are very large, in occluded regions or when camera moves backward.

Another limitation is that not all occluded surfaces are captured by the reconstructor: the reconstructed 3D models still have some holes. While many of these regions are filled in, some are missed, even when multiple Gaussians are predicted.

Finally, failures in the pre-trained depth estimator are likely to lead to failures in our scene reconstructions, especially if the estimated depth is over-estimated. This is due to the non-negativity of our depth offsets, which therefore cannot recover scene structure closer to the camera than the surface estimated by the pre-trained depth estimator. This makes the model dependent on the quality of a third-party model within the domain of use at inference time.

5. Broader impacts

This work, on monocular scene reconstruction, has potential positive and negative social impacts. On the positive side, the approach significantly reduces the compute and time resources needed to acquire 3D assets in-the-wild, opening the door to consumer applications with positive impacts. For example, the ability to quickly reconstruct one's house to facilitate its sale; the ability to digitally preserve artefacts and sites of cultural heritage; and uses in safe autonomous driving.

On the negative side, this technology has the potential to be used for malicious purposes, such as illegal or unethical tracking and surveillance, or be invasive of someone's privacy, for example by reconstructing their body without their consent. In addition, incorrect predictions may cause harm if used in applications like autonomous driving and robotics, where mis-estimated 3D structures could lead to crashes or

suboptimal performance.

References

[1] David Charatan, Sizhe Li, Andrea Tagliasacchi, and Vincent Sitzmann. pixelSplat: 3D Gaussian splats from image pairs for scalable generalizable 3d reconstruction. In *arXiv.cs*, 2023. 2, 3

[2] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The KITTI dataset. *International Journal of Robotics Research (IJRR)*, 2013. 1

[3] Clément Godard, Oisin Mac Aodha, and Gabriel J. Brostow. Unsupervised monocular depth estimation with left-right consistency. *arXiv.cs*, abs/1609.03677, 2016. 2

[4] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J. Brostow. Digging into self-supervised monocular depth prediction. In *Proc. ICCV*, 2019. 2

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. CVPR*, 2016. 2

[6] Mu Hu, Wei Yin, Chi Zhang, Zhipeng Cai, Xiaoxiao Long, Hao Chen, Kaixuan Wang, Gang Yu, Chunhua Shen, and Shaojie Shen. Metric3d v2: A versatile monocular geometric foundation model for zero-shot metric depth and surface normal estimation. *arXiv:2404.15506*, 2024. 2

[7] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *Proc. ICLR*, 2015. 3

[8] Jiaxin Li, Zijian Feng, Qi She, Henghui Ding, Changhu Wang, and Gim Hee Lee. MINE: towards continuous depth MPI with nerf for novel view synthesis. In *Proc. ICCV*, 2021. 1, 2

[9] Luigi Piccinelli, Yung-Hsu Yang, Christos Sakaridis, Mattia Segu, Siyuan Li, Luc Van Gool, and Fisher Yu. Unidepth: Universal monocular metric depth estimation. In *Proc. CVPR*, 2024. 2

[10] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Proc. MICCAI*, 2015. 2

[11] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proc. CVPR*, 2016. 1

[12] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from RGBD images. In *Proc. ECCV*, 2012. 1

[13] Stanislaw Szymanowicz, Christian Rupprecht, and Andrea Vedaldi. Splatter Image: Ultra-fast single-view 3D reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 1

[14] Richard Tucker and Noah Snavely. Single-view view synthesis with multiplane images. In *Proc. CVPR*, 2020. 2, 3

[15] Shubham Tulsiani, Richard Tucker, and Noah Snavely. Layer-structured 3d scene inference via view synthesis. In *Proc. ECCV*, 2018. 1

[16] Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Trans. on Image Processing*, 13 (4), 2004. 2

[17] Felix Wimbauer, Nan Yang, Christian Rupprecht, and Daniel Cremers. Behind the scenes: Density fields for single view reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 1

[18] Lihe Yang, Bingyi Kang, Zilong Huang, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything: Unleashing the power of large-scale unlabeled data. In *Proc. CVPR*, 2024. 2

[19] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. *arXiv:2406.09414*, 2024. 2