

A More Experimental Analysis

Cross dataset generalization. Following MVsplat [1], we conducted experiments using a pretrained model on the RealEstate10K (RE10K) dataset [2] (as detailed in Tab. 2) and tested its performance on the ACID dataset [3] to evaluate the generalization capabilities of our proposed ZPressor across diverse datasets. As demonstrated in Table A, MVsplat with ZPressor exhibits remarkable efficacy in cross-dataset generalization. Notably, this performance advantage becomes progressively more pronounced with an increasing number of input views.

Table A: **Quantitative comparison on ACID [3] with trained model on RE10K.** Trained on indoor scenes (RE10K), MVsplat [1] and pixelSplat [4] with ZPressor perform much better as evaluated on the ACID dataset.

Views	Methods	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow
36 views	pixelSplat	OOM	OOM	OOM
	pixelSplat + Ours	27.78	0.823	0.238
	MVsplat	24.89	0.812	0.179
	MVsplat + Ours	28.16 ^{+3.27}	0.853 ^{+0.041}	0.145 ^{-0.034}
24 views	pixelSplat	OOM	OOM	OOM
	pixelSplat + Ours	27.91	0.825	0.235
	MVsplat	25.46	0.829	0.167
	MVsplat + Ours	28.33 ^{+2.87}	0.856 ^{+0.027}	0.142 ^{-0.025}
16 views	pixelSplat	OOM	OOM	OOM
	pixelSplat + Ours	27.97	0.826	0.234
	MVsplat	26.08	0.844	0.156
	MVsplat + Ours	28.42 ^{+2.34}	0.858 ^{+0.014}	0.141 ^{-0.015}
8 views	pixelSplat	26.69	0.807	0.260
	pixelSplat + Ours	28.05 ^{+1.36}	0.828 ^{+0.021}	0.234 ^{-0.026}
	MVsplat	27.89	0.864	0.140
	MVsplat + Ours	28.60 ^{+0.71}	0.860 ^{-0.004}	0.140 ^{-0.000}

B More Implementation Details

Network architectures. In Algorithm 1, we provide a detailed description of how ZPressor is integrated into existing feed-forward 3D Gaussian Splatting (3DGS) frameworks [1, 4, 5]. Initially, we select anchor views and their corresponding support views following Algorithm 2 and Eq. (5). The features associated with these views are then processed by an attention-based network. This network is composed of 6 structurally identical blocks, wherein each block encompasses a cross-attention layer, a self-attention layer, and an MLP layer. The cross-attention mechanism operates by employing the anchor features as query, while the support features provide the key and value. Subsequent to this fusion, the resulting features are further refined by the self-attention and MLP layers.

To ensure training stability, deviating from traditional Transformer architectures, we employ Pre-Layer Normalization [6] (Pre-LN), which enhances the robustness of the model. Furthermore, system-level advancements have been incorporated to accelerate computation. For example, we employ FlashAttention [7, 8], which uses highly optimized GPU kernels and leverages hardware topology to compute attention in a time- and memory-efficient manner.

More training details. We use the first model version of DepthSplat [5] from its October 2024 release. Experimental results obtained with this specific version may exhibit slight variations when compared to the current version. ZPressor was incorporated subsequent to the monocular feature extraction performed by CNN.

Adhering to its original configuration, experiments were conducted at a resolution of 256×448 . The model was initially trained on the RE10K [2] for 100,000 steps and subsequently fine-tuned on the DL3DV [9] for an additional 100,000 steps. We employed the AdamW optimizer [10] with a learning

Algorithm 1 Overview of Feed-Forward 3DGS framework with ZPressor

Input: K input views $\mathcal{V} = \{V_i\}_{i=1}^K$, camera poses $\mathcal{P} = \{P_i\}_{i=1}^K$, the number of anchor views N , the number of network blocks h .

Output: Gaussian parameters $\mathcal{Y} = \{(\mu, \Sigma, \alpha, c)\}$.

$\mathcal{X} \leftarrow \Phi_{image}(\mathcal{V}, \mathcal{P})$

$\mathcal{X}_{\text{anchor}}, \mathcal{X}_{\text{support}} \leftarrow \mathcal{X}$, with Anchor view selection.

Assign support views to anchor cluster $\mathcal{C} \leftarrow \mathcal{X}_{\text{support}}$

Initialize state $\mathcal{Z} \leftarrow \mathcal{X}_{\text{anchor}}$

for $i \leftarrow 1$ to h **do**

$\mathcal{Z} \leftarrow \text{Cross-Attn}(Q, K, V)$, where $Q \leftarrow \mathcal{Z}$ $K, V \leftarrow \mathcal{X}_{\text{support}}$

$\mathcal{Z} \leftarrow \text{Self-Attn}(Q, K, V)$, where $Q, K, V \leftarrow \mathcal{Z}$

$\mathcal{Z} \leftarrow \text{MLP}(\mathcal{Z})$

end for

$\{(\mu_i, \Sigma_i, \alpha_i, \mathbf{c}_i)\} \leftarrow \Psi_{\text{pred}}(\mathcal{Z}, \mathcal{P})$

return $\mathcal{Y} \leftarrow \{(\mu_i, \Sigma_i, \alpha_i, \mathbf{c}_i)\}$

Algorithm 2 Farthest Point Sampling for Anchor View Selection

Input: Set of view camera positions $\mathcal{T} = \{\mathbf{T}_1, \mathbf{T}_2, \dots, \mathbf{T}_K\}$, Number of anchor views N

Output: Indices of the selected anchor views $\mathcal{S} = \{\mathbf{T}_{a_1}, \mathbf{T}_{a_2}, \dots, \mathbf{T}_{a_n}\}$

Initialize the set of anchor view indices $\mathcal{S} \leftarrow \emptyset$

Randomly select a random anchor view $\mathbf{T}_{a_1} \in \mathcal{T}$, where $\mathbf{T}_{a_1} \sim \text{Uniform}(\mathcal{T})$

Add \mathbf{T}_{a_1} to \mathcal{S} : $\mathcal{S} \leftarrow \{\mathbf{T}_{a_1}\}$

for $j \leftarrow 2$ to N **do**

 Initialize a dictionary to store minimum distances $D \leftarrow \{\}$

for $k \leftarrow 1$ to K **do**

if $k \notin \mathcal{S}$ **then**

 Calculate the minimum distance $d_k \leftarrow \min_{i \in \mathcal{S}} \|\mathbf{T}_k - \mathbf{T}_i\|_2$

 Store the distance: $D[k] \leftarrow d_k$

end if

end for

 Find the view position T_{a_j} with the maximum minimum distance: $T_{a_j} \leftarrow \arg \max_{k \notin \mathcal{S}} D[k]$

 Add a_j to \mathcal{S} : $\mathcal{S} \leftarrow \mathcal{S} \cup \{T_{a_j}\}$

end for

return \mathcal{S}

29 rate of 2×10^{-4} . The total training duration was approximately two days, and the integration of
30 ZPressor did not significantly alter the original training time of DepthSplat.

31 Similarly, for MVSplat [1] and pixelSplat [4], ZPressor was integrated after the monocular feature
32 extraction stage. MVSplat utilizes a CNN for feature extraction, whereas pixelSplat employs
33 DINO [11, 12]; this architectural choice in pixelSplat contributes to a marginally higher VRAM
34 consumption compared to the other two baselines. We maintained the model parameter settings as
35 published in their respective original works, training models on the RE10K [2] at a resolution of
36 256×256 . The learning rate was set to 2×10^{-4} for MVSplat and 1.5×10^{-4} for pixelSplat, where
37 both of which were trained for 100,000 steps. Notably, due to memory constraints, we trained the
38 pixelSplat model incorporating ZPressor using 4 anchor views, in contrast to the 6 anchor views
39 configured for DepthSplat and MVSplat. The training times for MVSplat and pixelSplat, when
40 augmented with ZPressor, remained comparable to their original durations.

41 We will **open-source** the complete codebase for ZPressor, our ZPressor-integrated versions of
42 DepthSplat, MVSplat, and pixelSplat, and all associated model checkpoints.

43 C Limitation and Societal Impacts

44 **Limitation analysis.** As discussed in Sec. 5, ZPressor exhibits limitations when processing scenarios
45 with an extremely high density of input views. Specifically, its efficacy in compressing the information

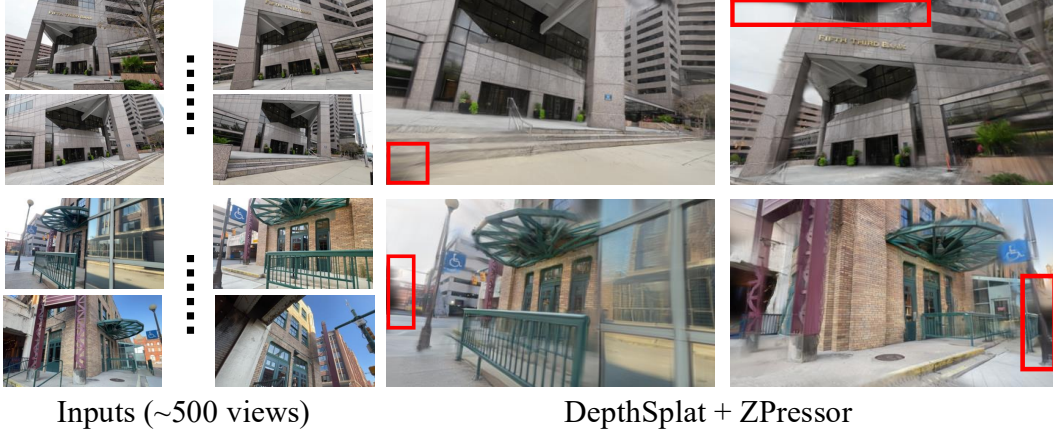


Figure A: **Limitations.** Visual results from extremely dense input views show slightly poor presentation effect.

from such dense views through a limited set of anchor views is diminished. To illustrate this, we conducted an experiment on DepthSplat [5] integrated with ZPressor, using approximately 500 images as input. As depicted in Fig. A, the quality of the rendered novel views was perceptibly affected, which can be attributed to an insufficient number of Gaussian primitives to adequately represent the scene under these dense input conditions.

Potential and negative societal impacts. ZPressor can significantly reduce the training costs associated with feed-forward 3DGS networks. It enables the processing of a larger number of input views within the same VRAM budget and training duration, delivering high-fidelity rendering results and thereby decreasing energy consumption during the model training process. While the capability to render higher-quality novel views from more densely sampled perspectives positions ZPressor as a valuable tool for augmented reality applications, it is important to acknowledge that the fidelity of the rendering can be compromised by the emergence of artifacts, particularly when processing input views of extremely high density. Consequently, in safety-critical applications, such as the training of autonomous driving models, the deployment of ZPressor would necessitate the implementation of additional precautionary measures to mitigate potential risks arising from such limitations.

D More Visual Comparisons

This section provides additional qualitative comparison results. We present further visualizations for DepthSplat [5] on the DL3DV [9] and MVSplat [1] on the RE10K [2] in Fig. B and Fig. F, with our ZPressor.

Furthermore, to illustrate how ZPressor performs with dense input views, we showcase comparative results. For DepthSplat [5], comparisons between the original framework and DepthSplat augmented with ZPressor are presented for scenarios with 24, 16, and 12 input views in Fig. C, Fig. D, and Fig. E. Similarly, for MVSplat [1], visual comparisons between the original framework and MVSplat integrated with ZPressor are displayed for inputs of 24, 16, and 8 views in Fig. G, Fig. H, and Fig. I. The corresponding quantitative results for these multi-view experiments can be found in Tab. 2.

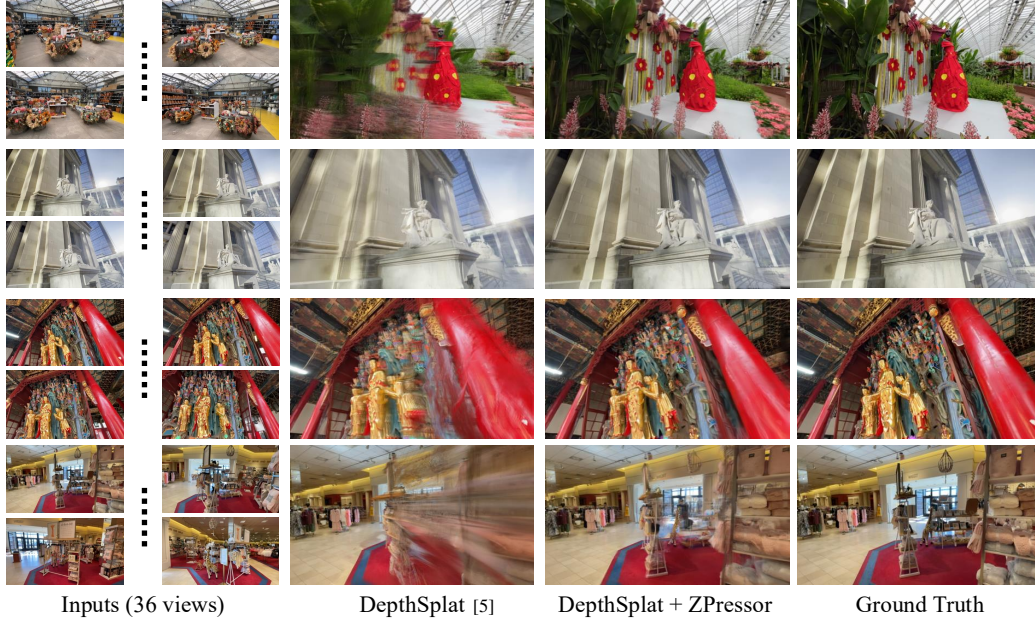


Figure B: **More qualitative comparisons on DL3DV [9] with DepthSplat [5] under 36 input views.** Models with ZPressor performs the best in all cases.

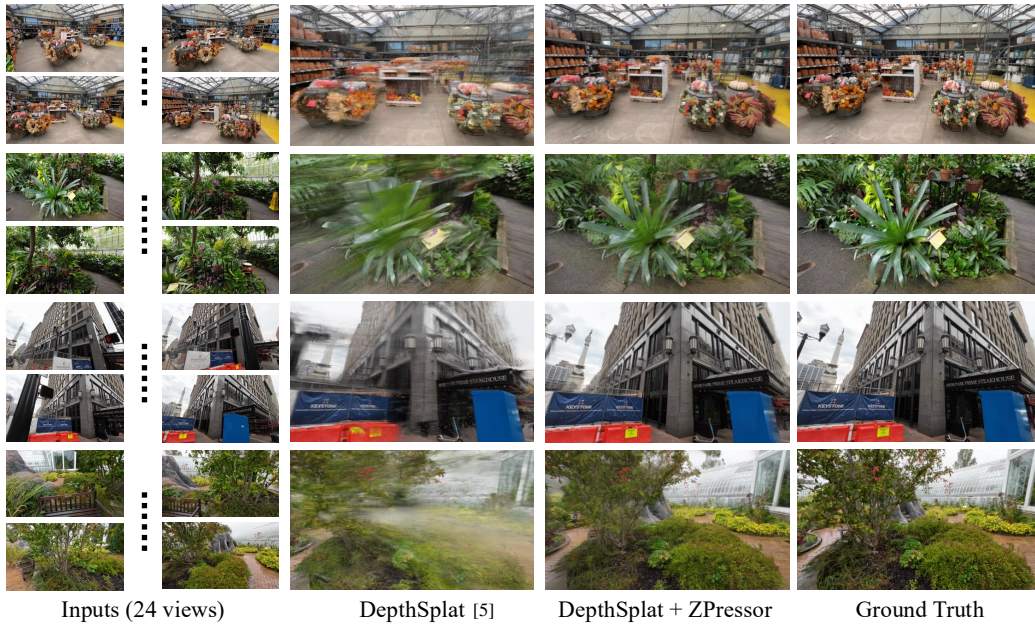


Figure C: **More qualitative comparisons on DL3DV [9] with DepthSplat [5] under 24 input views.**



Figure D: More qualitative comparisons on DL3DV [9] with DepthSplat [5] under 16 input views.

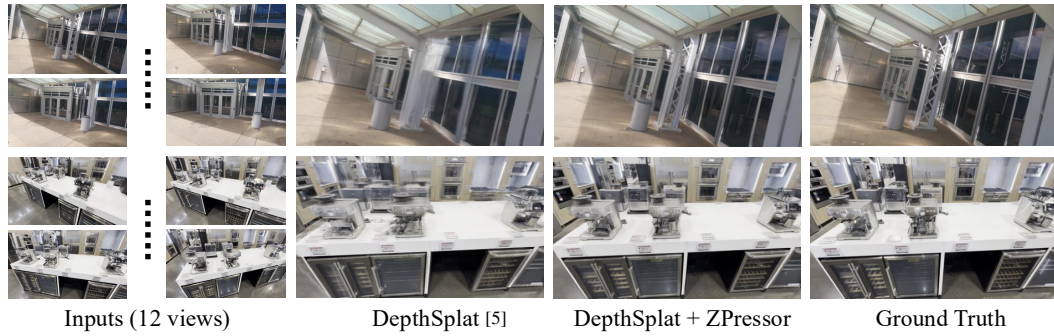


Figure E: More qualitative comparisons on DL3DV [9] with DepthSplat [5] under 12 input views.

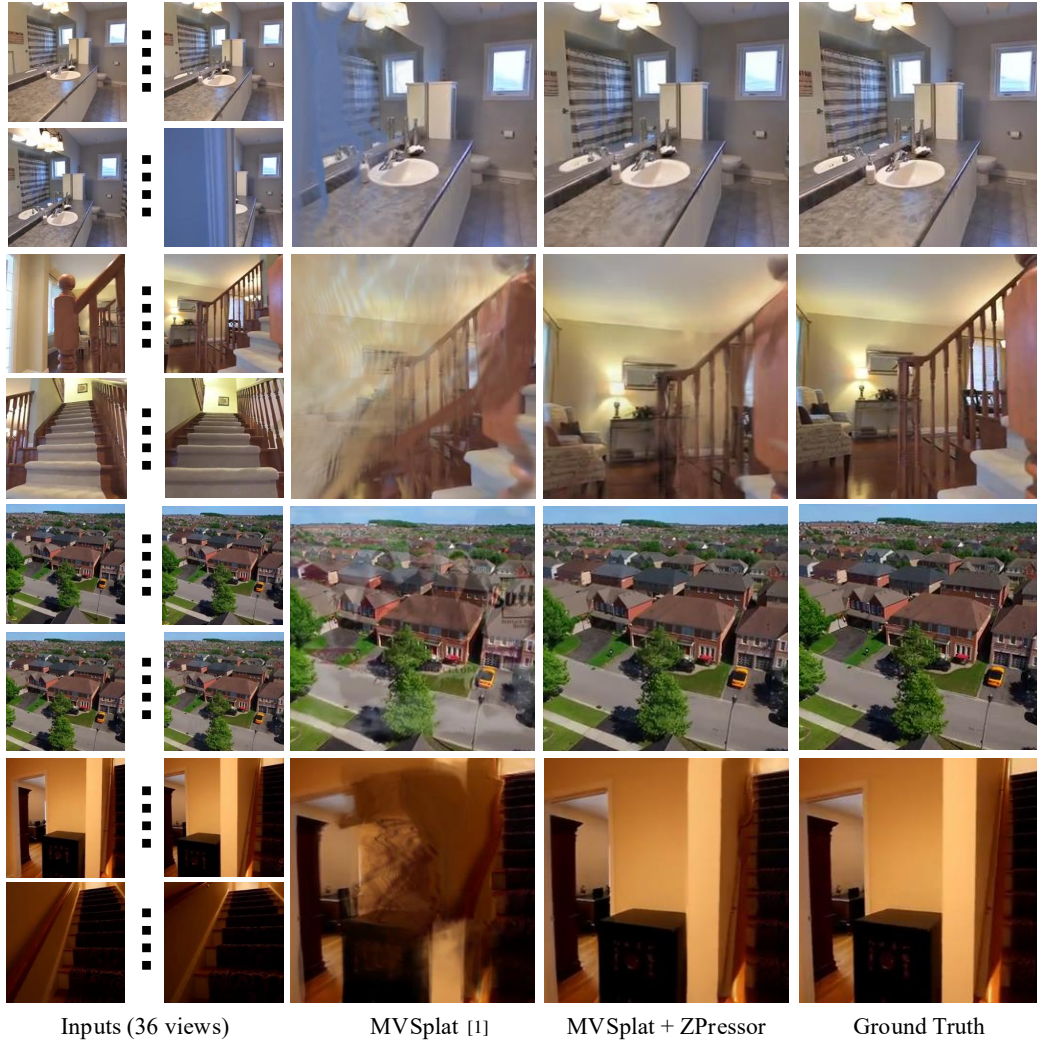


Figure F: **More qualitative comparisons on RE10K [2] with MVSplat [1] under 36 input views.** Models with ZPressor performs the best in all cases.

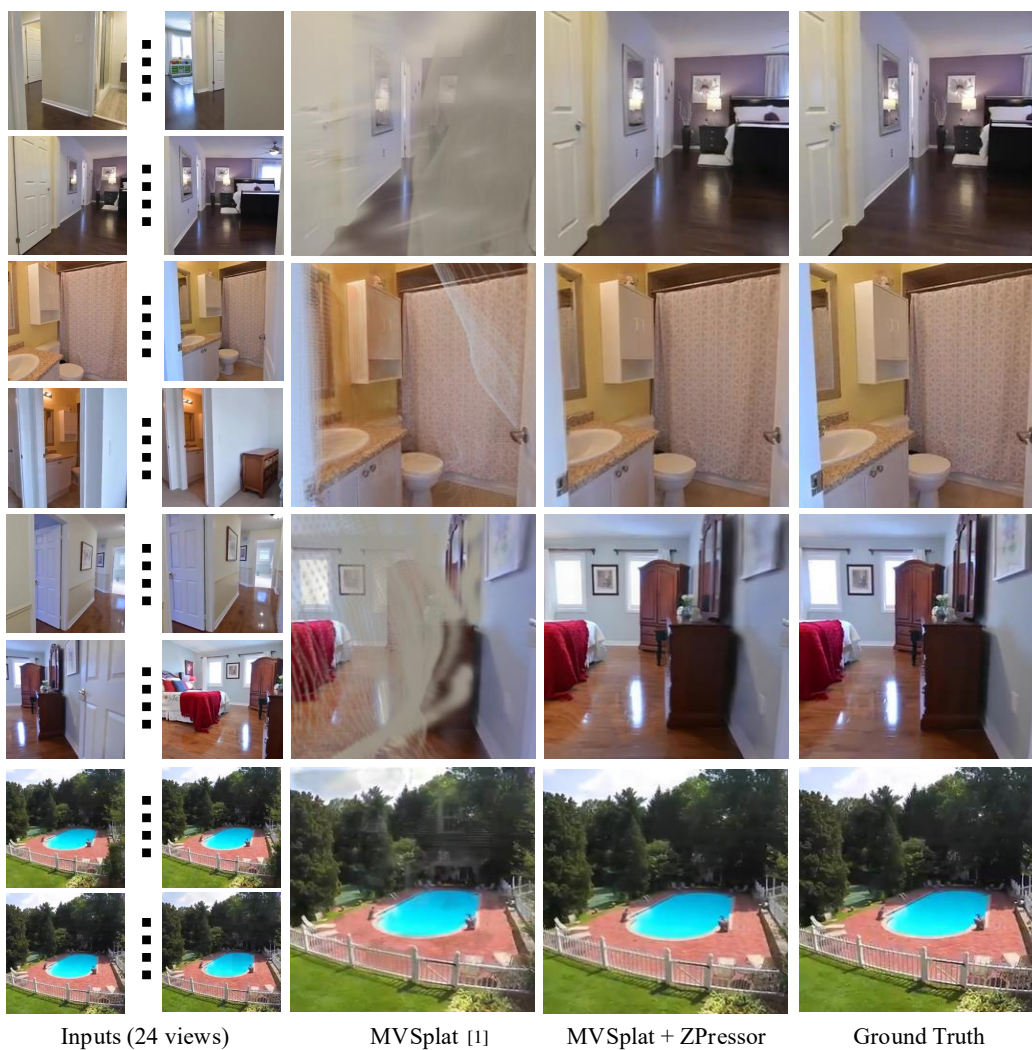


Figure G: More qualitative comparisons on RE10K [2] with MVSplat [1] under 24 input views.



Figure H: More qualitative comparisons on RE10K [2] with MV Splat [1] under 16 input views.



Figure I: More qualitative comparisons on RE10K [2] with MV Splat [1] under 8 input views.

References

- [1] Yuedong Chen, Haoifei Xu, Chuanxia Zheng, Bohan Zhuang, Marc Pollefeys, Andreas Geiger, Tat-Jen Cham, and Jianfei Cai. Mvsplat: Efficient 3d gaussian splatting from sparse multi-view images. In *European Conference on Computer Vision*, pages 370–386. Springer, 2024.
- [2] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. *arXiv preprint arXiv:1805.09817*, 2018.
- [3] Andrew Liu, Richard Tucker, Varun Jampani, Ameesh Makadia, Noah Snavely, and Angjoo Kanazawa. Infinite nature: Perpetual view generation of natural scenes from a single image. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14458–14467, 2021.
- [4] David Charatan, Sizhe Lester Li, Andrea Tagliasacchi, and Vincent Sitzmann. pixelsplat: 3d gaussian splats from image pairs for scalable generalizable 3d reconstruction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 19457–19467, 2024.
- [5] Haoifei Xu, Songyou Peng, Fangjinhua Wang, Hermann Blum, Daniel Barath, Andreas Geiger, and Marc Pollefeys. Depthsplat: Connecting gaussian splatting and depth. *arXiv preprint arXiv:2410.13862*, 2024.
- [6] Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. On layer normalization in the transformer architecture. In *International conference on machine learning*, pages 10524–10533. PMLR, 2020.
- [7] Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in neural information processing systems*, 35:16344–16359, 2022.
- [8] Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*, 2023.
- [9] Lu Ling, Yichen Sheng, Zhi Tu, Wentian Zhao, Cheng Xin, Kun Wan, Lantao Yu, Qianyu Guo, Zixun Yu, Yawen Lu, et al. D13dv-10k: A large-scale scene dataset for deep learning-based 3d vision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22160–22169, 2024.
- [10] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.
- [11] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021.
- [12] Maxime Oquab, Timothée Darcet, Theo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Russell Howes, Po-Yao Huang, Hu Xu, Vasu Sharma, Shang-Wen Li, Wojciech Galuba, Mike Rabbat, Mido Assran, Nicolas Ballas, Gabriel Synnaeve, Ishan Misra, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2023.