Supplemental Material of Z-Order Transformer for Feed-Forward Gaussian Splatting

Can Wang[1] Lei Liu[1] Wei Jiang[2] Dong Xu[1†]
[1]The University of Hong Kong [2]Futurewei Technologies Inc.

[1]

1. Z-order Based Maximum Coverage Viewpoint Selection Algorithm

In the main paper, we provide a high-level introduction to the proposed Z-order based Maximum Coverage Viewpoint Selection algorithm. Here, we include the detailed description of this algorithm, as shown in Algorithm 1. The proposed algorithm aims to select at most M viewpoints from a dense candidate set $\mathcal{V} = \{V_1, \ldots, V_N\}$ such that their union maximally covers the underlying 3D scene while avoiding redundant views. For each candidate viewpoint V_i, we first obtain its associated point cloud P_i and discretize the 3D space into a regular grid with a cell size δ. Each point $p \in P_i$ is mapped to its grid cell and then serialized into a one-dimensional key using a Z-order encoding $\mathcal{Z}(\phi(p/\delta))$, which preserves spatial locality, where ϕ denotes the quantization from continuous 3D coordinates to integer grid indices, implemented as component-wise floor $\phi(p/\delta) = \lfloor p/\delta \rfloor$. The resulting set of encoded grid cells for viewpoint V_i is denoted as S_i, and its cardinality $|S_i|$ reflects the raw coverage of that viewpoint. We then initialize a max-heap $\mathcal{H}$ with tuples $(|S_i|, i)$ for all viewpoints, so that the viewpoint with the largest coverage can be efficiently retrieved. The algorithm proceeds in a greedy fashion for at most M iterations. At each iteration, we repeatedly pop the current best candidate (c, j) from $\mathcal{H}$ and compute its marginal contribution $\Delta = S_j \setminus \mathcal{C}$ with respect to the already covered set $\mathcal{C}$. If $|\Delta| > 0$, viewpoint V_j is accepted: we update the global coverage $\mathcal{C} \leftarrow \mathcal{C} \cup \Delta$ and append j to the selected viewpoint index set $\mathcal{S}$. If the heap is exhausted without finding a viewpoint that adds new coverage, the selection process terminates early, indicating that remaining viewpoints are completely redundant. After each successful selection, we rebuild the heap by recomputing, for every remaining candidate viewpoint V_j, its updated marginal coverage $\Delta_j = S_j \setminus \mathcal{C}$ and pushing only those with $|\Delta_j| > 0$ back into a new heap $\mathcal{H}_{\text{new}}$. Overall, this Z-order based greedy scheme incrementally selects the most informative viewpoints under a discretized spatial coverage criterion, effectively eliminating redundant views and improving inference efficiency while avoiding significant performance degradation.

Inp. Views	Selection	PSNR↑	SSIM↑	LPIPS↓	Time(s)↓
64	NA.	**29.44**	**0.911**	**0.098**	1.891
	RS. 16	28.50	0.897	0.118	**0.421**
	ZS. 16	29.13	0.908	0.106	0.498
24	NA.	**28.91**	**0.906**	**0.102**	0.622
	RS. 16	27.97	0.891	0.113	**0.417**
	ZS. 16	28.73	0.903	0.108	0.448
16	NA.	**28.67**	**0.901**	**0.110**	0.417
	RS. 8	27.06	0.876	0.125	**0.255**
	ZS. 8	28.07	0.898	0.116	0.272

Table S1. **Ablation Study with Different Selection Strategies.** NA. indicates that all views will be used during inference, RS. refers to random selection, while ZS. denotes our Z-order-based view selection method. The input resolution is 360×640.

In Tab. 6 of the main paper, we investigate the impact of different view selection strategies. Here, we further include a denser input setting to extend Tab. 6 and evaluate performance, as shown in Tab. S1-top. For the denser 64-view input setting, our method still performs well, exhibiting only a minor performance drop while removing redundant views. The efficiency improvement is also more pronounced in this case, as we discard half of the input views.

2. More Comparisons with Related Works

While our comparison baseline, AnySplat [4], has already been shown to outperform NoPoSplat [14] and FLARE [15], and DepthSplat [12] has been shown to surpass MVSplat [2] and PixelSplat [1], we have also included direct comparisons with these methods, as well as MonoSplat [6], to further validate the effectiveness of our approach. All methods are trained and evaluated on the RealEstate10K dataset [16] using the same training and testing split at a resolution of 256×256 with 2 input views.

[1†] Dong Xu is the corresponding author.

Algorithm 1 Z-order Based Maximum Coverage Viewpoint Selection with Max-Heap

Require: Viewpoint set $\mathcal{V} = \{V_1, V_2, \ldots, V_N\}$, Number of viewpoints to select M, Grid size δ
Ensure: Selected viewpoint indices $\mathcal{S}$, final coverage C

$\mathcal{C} \leftarrow \emptyset$ ▷ Set of covered grid cells
$\mathcal{S} \leftarrow \emptyset$ ▷ Set of selected viewpoints
$\mathcal{H} \leftarrow \emptyset$ ▷ Max-heap (coverage, viewpoint index)
for $i = 1$ **to** N **do**
  $P_i \leftarrow$ Get point cloud for viewpoint V_i
  $S_i \leftarrow \{\mathcal{Z}(\phi(p/\delta)) \mid p \in P_i\}$ ▷ Z-order serialization with grid size δ
  push $(|S_i|, i)$ into $\mathcal{H}$ ▷ Push initial coverage into heap
end for
for $k = 1$ **to** M **do**
  if $\mathcal{H} = \emptyset$ **then**
    break
  end if
  found $\leftarrow$ False
  while $\mathcal{H} \neq \emptyset$ **and not** found **do**
    $(c, j) \leftarrow$ pop from $\mathcal{H}$ ▷ Get max coverage candidate
    $\Delta \leftarrow S_j \setminus \mathcal{C}$ ▷ Find new covered grid cells for viewpoint j
    if $|\Delta| > 0$ **then**
      $\mathcal{C} \leftarrow \mathcal{C} \cup \Delta$ ▷ Add new covered grid cells
      $\mathcal{S} \leftarrow \mathcal{S} \cup \{j\}$ ▷ Add viewpoint j to selected viewpoints
      found $\leftarrow$ True
      break
    end if
  end while
  if not found **then**
    break ▷ No more coverage can be added
  end if
  $\mathcal{H}_{\text{new}} \leftarrow \emptyset$
  while $\mathcal{H} \neq \emptyset$ **do**
    $(c, j) \leftarrow$ pop from $\mathcal{H}$
    $\Delta_j \leftarrow S_j \setminus \mathcal{C}$ ▷ Recalculate new coverage for viewpoint j
    if $|\Delta_j| > 0$ **then**
      push $(|\Delta_j|, j)$ into $\mathcal{H}_{\text{new}}$
    end if
  end while
  $\mathcal{H} \leftarrow \mathcal{H}_{\text{new}}$
end for
return $\mathcal{S}, |\mathcal{C}|$

As summarized in Table S2, our method consistently delivers better reconstruction quality (higher PSNR and SSIM, lower LPIPS), demonstrating that our design provides clear advantages over these related approaches.

Method	PSNR↑	SSIM↑	LPIPS↓
NoPoSplat [14]	27.41	0.884	0.116
FLARE [15]	23.78	0.801	0.191
MonoSplat [6]	26.68	0.875	0.123
MVSplat [2]	26.39	0.869	0.128
pixelSplat [1]	25.89	0.858	0.142
Ours	**27.89**	**0.892**	**0.110**

Table S2. **More Comparisons.** We present additional comparison results on the RealEstate10K dataset at a resolution of 256×256 with 2 input views.

Method	PSNR↑	SSIM↑	LPIPS↓
Ours w/o sel	26.79	0.847	0.174
Ours w/o SA	26.79	0.847	0.174
Ours	**28.56**	**0.901**	**0.110**

Table S3. **Ablation Studies of Sparse Attention.** We conduct ablation experiments on different components of our sparse attention module. Experiments are conducted on the RealEstate10K dataset at a resolution of 360×640 with 12 input views. Note that we do not perform an ablation of the group attention removal, as the selection attention depends on it.

3. Ablation Studies of Sparse Attention

To quantify the contribution of each component in our sparse attention module, we perform ablation experiments on the RealEstate10K dataset with 12 input views under the same training and evaluation settings as in the main paper. As reported in Table S3, disabling the selection mechanism ("Ours w/o sel") degrades performance, demonstrating that adaptively selecting informative tokens is important for effective sparse attention. When we remove the entire sparse attention module ("Ours w/o SA"), the performance degrades the most, confirming that group attention and selective attention jointly contribute to the overall gain. The complete model achieves the best reconstruction quality, with the highest PSNR/SSIM and the lowest LPIPS, demonstrating the effectiveness of our sparse attention design. Note that we do not perform an ablation of the group attention removal, as the selection attention depends on it.

4. Analysis of Using the VGGT Backbone

We also considered using VGGT [10] as the backbone for feature extraction instead of the depth-anything-v2-small [13]. Since VGGT also utilizes DINOv2 [8] and DPT head [9] structures, we fused the global and geometric features extracted from VGGT. The experiment was

Method	PSNR↑	SSIM↑	LPIPS↓	Time(s)↓
with VGGT [10]	**28.81**	**0.907**	**0.108**	0.815
Ours	28.56	0.901	0.110	**0.337**

Table S4. **Comparisons with and without the VGGT Backbone.** Using a pre-trained VGGT as the feature and depth extractor slightly improves performance but requires more inference time. Experiments are conducted on the RealEstate10K dataset at a resolution of 360×640 with 12 input views.

also conducted on the RealEstate10K dataset with 12 input views. As shown in Table S4, using a pre-trained VGGT backbone slightly improves reconstruction quality (higher PSNR/SSIM and lower LPIPS), likely due to VGGT being trained on a larger dataset. However, the inference time is significantly longer when using VGGT, as it has many more model parameters (1B) compared to depth-anything-v2-small (24.8M). As a result, we continue to adopt the depth-anything-v2-small backbone. This experiment also demonstrates that our framework is not limited to a specific feature extractor.

5. Supplementary Video

We provide a supplementary video showcasing additional visual results rendered from multiple viewpoints. We highly recommend watching it to better appreciate the view consistency and high fidelity achieved by our method.

6. Limitations and Future Work

Although our proposed method demonstrates improvements in feed-forward 3D Gaussian Splatting for novel view synthesis, there are still certain limitations that could be addressed in future work. One key limitation of our model is that, although it is efficient, it may still face difficulties when processing very high-resolution datasets (e.g., those exceeding 1K), where fine details are not always accurately captured due to the inherent trade-off between model complexity and memory constraints. Moreover, our current Z-order transformer employs one or two Z-order blocks to aggregate Gaussian primitives. Using more such blocks can further reduce the number of Gaussian points; however, it leads to a noticeable degradation in performance, as illustrated in Fig. 7 of the main paper. Therefore, achieving higher compression without significant performance loss remains a challenging task.

In future work, several possible directions could be explored to further improve the proposed approach. One potential direction is to investigate hierarchical or multi-scale feature representations, which might help preserve fine-grained details when processing very high-resolution datasets while maintaining computational efficiency. It is also worth exploring alternative designs of the Z-order transformer, such as varying block configurations or introducing learnable aggregation depths to achieve a better balance between Gaussian primitive reduction and performance. Finally, combining the proposed framework with hybrid neural rendering methods [3, 5, 7, 11] might help enhance robustness and generalization across diverse scenarios.

References

[1] 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. 1, 2

[2] Yuedong Chen, Haofei 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. 1, 2

[3] Shuangkang Fang, I Shen, Takeo Igarashi, Yufeng Wang, ZeSheng Wang, Yi Yang, Wenrui Ding, Shuchang Zhou, et al. Nerf is a valuable assistant for 3d gaussian splatting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 26230–26240, 2025. 3

[4] Lihan Jiang, Yucheng Mao, Linning Xu, Tao Lu, Kerui Ren, Yichen Jin, Xudong Xu, Mulin Yu, Jiangmiao Pang, Feng Zhao, et al. Anysplat: Feed-forward 3d gaussian splatting from unconstrained views. *arXiv preprint arXiv:2505.23716*, 2025. 1

[5] Jingyu Lin, Jiaqi Gu, Lubin Fan, Bojian Wu, Yujing Lou, Renjie Chen, Ligang Liu, and Jieping Ye. Hybridgs: Decoupling transients and statics with 2d and 3d gaussian splatting. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 788–797, 2025. 3

[6] Yifan Liu, Keyu Fan, Weihao Yu, Chenxin Li, Hao Lu, and Yixuan Yuan. Monosplat: Generalizable 3d gaussian splatting from monocular depth foundation models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 21570–21579, 2025. 1, 2

[7] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM transactions on graphics (TOG)*, 41(4):1–15, 2022. 3

[8] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel HAZIZA, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2025. 2

[9] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 12179–12188, 2021. 2

[10] Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *Proceedings of the*

Computer Vision and Pattern Recognition Conference, pages 5294–5306, 2025. 2, 3

[11] Zipeng Wang and Dan Xu. Hyrf: Hybrid radiance fields for memory-efficient and high-quality novel view synthesis. *arXiv preprint arXiv:2509.17083*, 2025. 3

[12] Haofei Xu, Songyou Peng, Fangjinhua Wang, Hermann Blum, Daniel Barath, Andreas Geiger, and Marc Pollefeys. Depthsplat: Connecting gaussian splatting and depth. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 16453–16463, 2025. 1

[13] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. *Advances in Neural Information Processing Systems*, 37:21875–21911, 2024. 2

[14] Botao Ye, Sifei Liu, Haofei Xu, Xueting Li, Marc Pollefeys, Ming-Hsuan Yang, and Songyou Peng. No pose, no problem: Surprisingly simple 3d gaussian splats from sparse unposed images. In *The Thirteenth International Conference on Learning Representations*, 2025. 1, 2

[15] Shangzhan Zhang, Jianyuan Wang, Yinghao Xu, Nan Xue, Christian Rupprecht, Xiaowei Zhou, Yujun Shen, and Gordon Wetzstein. Flare: Feed-forward geometry, appearance and camera estimation from uncalibrated sparse views. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 21936–21947, 2025. 1, 2

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