

# VolSplat: Rethinking Feed-Forward 3D Gaussian Splatting with Voxel-Aligned Prediction

Weijie Wang<sup>1,2\*</sup>, Yeqing Chen<sup>3\*</sup>, Zeyu Zhang<sup>2</sup>, Hengyu Liu<sup>2,4</sup>, Haoxiao Wang<sup>1</sup>, Zhiyuan Feng<sup>5</sup>, Wenkang Qin<sup>2</sup>, Zheng Zhu<sup>2†</sup>, Donny Y. Chen<sup>6</sup>, Bohan Zhuang<sup>1†</sup>

<sup>1</sup>Zhejiang University <sup>2</sup>GigaAI <sup>3</sup>University of Electronic Science and Technology of China  
<sup>4</sup>The Chinese University of Hong Kong <sup>5</sup>Tsinghua University <sup>6</sup>Monash University

**Abstract**—Feed-forward 3D Gaussian Splatting (3DGS) has emerged as a highly effective solution for novel view synthesis. Existing methods predominantly rely on a *pixel-aligned* Gaussian prediction paradigm, where each 2D pixel is mapped to a 3D Gaussian. We rethink this widely adopted formulation and identify several inherent limitations: it renders the reconstructed 3D models heavily dependent on the number of input views, leads to view-biased density distributions, and introduces alignment errors, particularly when source views contain occlusions or low texture. To address these challenges, we introduce VolSplat, a new multi-view feed-forward paradigm that replaces pixel alignment with voxel-aligned Gaussians. By directly predicting Gaussians from a predicted 3D voxel grid, it overcomes pixel alignment’s reliance on error-prone 2D feature matching, ensuring robust multi-view consistency. Furthermore, it enables adaptive control over Gaussian density based on 3D scene complexity, yielding more faithful Gaussian point clouds, improved geometric consistency, and enhanced novel-view rendering quality. Experiments on widely used benchmarks including RealEstate10K and ScanNet demonstrate that VolSplat achieves state-of-the-art performance while producing more plausible and view-consistent Gaussian reconstructions. In addition to superior results, our approach establishes a more scalable framework for feed-forward 3D reconstruction with denser and more robust representations, paving the way for further research in wider communities. The video results, code and trained models are available on our project page: <https://1hmd.top/volsplat>.

## I. INTRODUCTION

3D reconstruction is a cornerstone of modern robotics, empowering autonomous systems with the critical ability to perceive, map, and comprehend their physical environment, which is fundamental for advanced navigation, object manipulation, and intelligent interaction. Traditional optimization based approaches, including Neural Radiance Fields (NeRF) [1] and 3D Gaussian Splatting (3DGS) [2], obtain high fidelity results by iteratively enforcing photometric or geometric consistency. These methods achieve excellent accuracy but are computationally intensive and slow to run at inference time. By contrast, feed-forward approaches [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13] trade per instance optimization for fast learned inference. A single forward pass predicts scene geometry or a 3D representation directly from input images. This speed and simplicity make feed-forward systems attractive for real time applications,

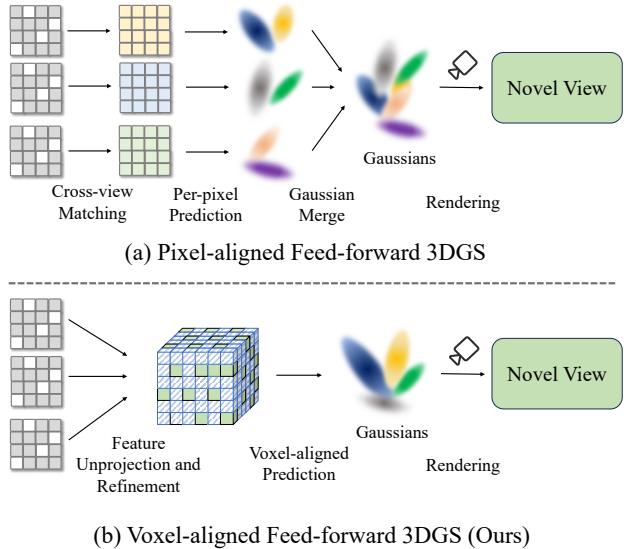


Fig. 1: **Comparison between the pixel-aligned feed-forward method and our approach.** Pixel-aligned feed-forward 3DGS methods suffer from two primary limitations: 1) 2D feature matching struggles to effectively resolve the multi-view alignment problem, and 2) the Gaussian density is constrained and cannot be adaptively controlled according to scene complexity. We propose VolSplat, a framework that directly regresses Gaussians from 3D features based on a voxel-aligned prediction strategy. This approach achieves adaptive control over scene complexity and resolves the multi-view alignment challenge.

large scale datasets, and downstream tasks that require many reconstructions.

Prior feed-forward 3DGS methods [6], [8], [9], [11], [13] commonly rely on pixel alignment as their fundamental mechanism for associating image features with pixel aligned Gaussians. In this design, per-pixel features from precomputed image feature maps are unprojected to define the corresponding Gaussians. The prevailing consensus has been to perform fusion directly within the 2D feature representation. However, pixel aligned designs inherit several intrinsic limitations. Sampling at discrete pixel locations is sensitive to camera calibration and discretization error, produces inconsistent sampling patterns across views, and

\* Equal contribution. † Corresponding authors.

struggles in regions affected by occlusion, motion parallax, or low texture. Because the association is mediated entirely by two dimensional image coordinates, the resulting 3D predictions can suffer from depth ambiguities and unstable geometric reasoning, particularly when input viewpoints are sparse or when depth cues are weak.

In this work we shift the alignment paradigm from pixels to voxels, illustrated in Fig. 1. Instead of sampling features at projected pixel coordinates, we align and aggregate image features directly into a 3D voxel grid that shares a canonical volumetric coordinate frame with the predicted geometry. This voxel aligned formulation changes where and how information is associated. Features are deposited into and retrieved from consistent 3D locations, enabling the model to reason volumetrically and to exploit inductive biases of 3D U-Net [14] and sparse volumetric operators. Voxel alignment removes the need for per query 2D prediction patterns and results in more stable multi-view fusion, cleaner occlusion handling, and better support for joint geometry and appearance inference.

There are practical and conceptual advantages to voxel alignment. Volumetric aggregation reduces floaters and view dependent inconsistency because information from multiple views is fused into a shared 3D container before Gaussian prediction. Operating in a 3D grid enables the use of well studied 3D decoder and regularization strategies, which naturally encode locality and geometrical context. Voxel alignment simplifies the integration of auxiliary 3D signals such as depth maps [9] and point clouds [15]. These signals can be injected into or extracted from the same voxel frame without ad hoc reprojection heuristics. Finally, voxel centric representations are amenable to modern acceleration strategies such as sparse data structures, making the approach practical at the resolutions required for high quality reconstruction.

In this paper we present a feed-forward three dimensional reconstruction framework built around voxel alignment. As shown in Fig. 2, we first construct 3D feature grids using the extracted 2D image features, then refine the 3D features and use them to predict voxel-aligned Gaussians. We analyze the alignment errors that arise in pixel aligned pipelines and show how voxel alignment reduces these errors both conceptually and empirically. Through systematic experiments on synthetic and real world benchmarks, we demonstrate that voxel aligned feed-forward models achieve more accurate and robust reconstructions than comparable pixel aligned baselines on large-scale benchmarks such as ScanNet [16] and RealEstate10K [17]. Our contributions are as follows:

- We introduce voxel alignment as a principled alternative to pixel alignment for feed-forward 3DGS and present a practical end-to-end framework.
- We provide an analysis of alignment induced errors in pixel aligned systems and show how volumetric aggregation mitigates these failure modes.
- Experimental results suggest VolSplat achieves state-of-the-art performance on both public benchmarks ScanNet [16] and RealEstate10K [17].

## II. RELATED WORK

### A. Novel View Synthesis

Traditional approaches to Novel View Synthesis (NVS) primarily rely on geometry-based rendering methods that reconstruct explicit 3D scene geometry from images [18], image-based rendering techniques that interpolate between captured views without full 3D reconstruction [19], and light field rendering that samples and reprojects densely captured rays in space [20]. These methods required either accurate geometric proxies, densely sampled viewpoints, or both to produce convincing visual results, limiting their applicability in real-world scenarios. The emergence of NeRF [1] marked a paradigm shift, significantly improving both rendering quality and robustness over prior methods, which learns a continuous, implicit scene representation by utilizing a MLP to map position and viewing direction to a corresponding color and volume density. While NeRF-based methods [21], [22] require a long training time due to the per-ray rendering, 3DGS [2] and its variants [23], [24], [25] have been introduced to represent the 3D scene using a set of anisotropic 3D Gaussians.

### B. 3D Voxelization

Voxelization, which discretizes 3D space into regular voxel grids, has been a foundational representation in 3D reconstruction and modeling [26]. Prior methods used dense grids for their simplicity, but suffered from high memory costs and poor scalability [27]. To address this, sparse structures like octrees were introduced for more efficient storage and computation [28]. In modern applications, voxels are widely used as input to 3D Convolutional Neural Network (CNN) for tasks such as object detection [29] and semantic segmentation [30]. More recently, voxels are often used as sparse scaffolding rather than as the final representation, supporting more advanced rendering techniques. Representative methods include Plenoxels [31] and K-Planes [32], which optimize voxel-based radiance fields for fast, high-quality rendering, as well as structured strategies such as Scaffold-GS [33] and Octree-GS [34], which leverage voxel grids to organize and accelerate 3DGS.

### C. Feed-Forward 3D Gaussian Splatting

Recent advances in feed-forward 3DGS [6], [8], [9], [11], [13], [35], [36] offer a compelling alternative that directly predicts 3D Gaussians from input images in a single forward pass: pixelSplat [6] proposes a two-view feed-forward pipeline that combines epipolar transformers and depth prediction to generate Gaussians. MVsplat [8] introduces a cost-volume-based fusion strategy to enhance multi-view consistency. DepthSplat [9] leverages monocular depth features to improve fine 3D structure reconstruction from sparse views. Follow-up work extends feed-forward 3DGS to more complex scenarios, including pose-free inputs [37], [38], online stream inputs [11], and more dense inputs [10]. While these works adopt a pixel-aligned strategy to predict Gaussian primitives, the pixel-wise formulation struggles to handle multiple input views due to redundancy

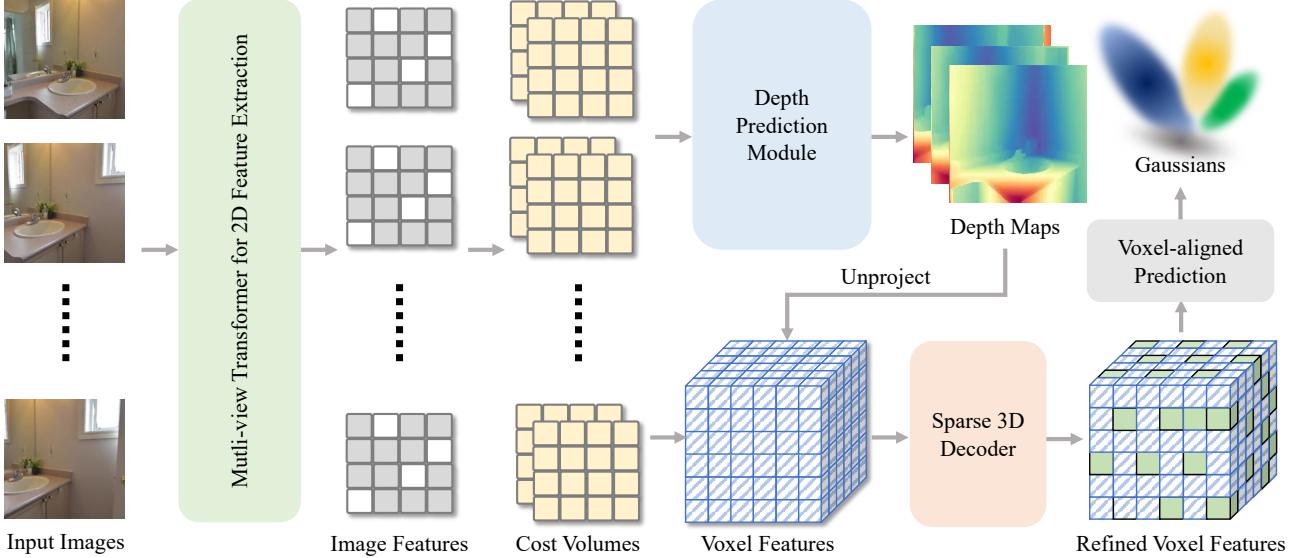


Fig. 2: **Overview of VolSplat.** Given multi-view images as input, we first extract 2D features for each image using a Transformer-based network and construct per-view cost volumes with plane sweeping. Depth Prediction Module then estimates a depth map for each view, which is used to unproject the 2D features into 3D space to form a voxel feature grid. Subsequently, we employ a sparse 3D decoder (details in Sec. III-C.1) to refine these features in 3D space and predict the parameters of a 3D Gaussian for each occupied voxel. Finally, novel views are rendered from the predicted 3D Gaussians.

and inconsistency across pixels. Existing methods attempt to improve the per-pixel strategy by pruning the number of Gaussians [35], token merging [39] and voxel-based fusion [36]. However, these approaches do not fundamentally address the limitations inherent in per-pixel processing. EVoLSplat [40] has explored voxel features in autonomous driving scenarios, but it has not been generalized to general scenarios and requires explicit 3D point clouds as intermediate representations. In contrast, our method introduces a voxel-aligned method, which eliminates the need for per-query 2D prediction patterns. This alignment enables more stable multi-view fusion, cleaner occlusion handling, and more coherent joint inference of geometry and appearance.

### III. METHOD

#### A. Preliminary and Observation

Feed-forward 3D reconstruction aims to learn a mapping from  $N$  input images  $\mathcal{I} = \{\mathbf{I}_i\}_{i=1}^N$  where  $\mathbf{I}_i \in \mathbb{R}^{H \times W \times 3}$  and their corresponding camera poses  $\mathcal{P} = \{\mathbf{P}_i\}_{i=1}^N$ , to a 3D scene representation. In the context of pixel-aligned 3DGS, features are extracted from images and refined by cross-view interaction:

$$\mathcal{F} = \{\mathbf{F}_i\}_{i=1}^N = h(\Phi_{\text{image}}(\mathcal{I}, \mathcal{P})), \quad \mathbf{F}_i \in \mathbb{R}^{\frac{H}{p} \times \frac{W}{p} \times C} \quad (1)$$

where  $\Phi_{\text{image}}$  is a pretrained image encoder. The function  $h$  is responsible for processing these features from different viewpoints, with its core purpose being to perform cross-view feature matching and fusion. For pixel-aligned Gaussian prediction, the features must be upsampled to the same

resolution as the input image:

$$\mathcal{F}_{\text{full}} = U(\mathcal{F}), \quad \mathbf{F}_{\text{full}} \in \mathbb{R}^{H \times W \times C} \quad (2)$$

where  $U$  is a feature upsampler such as CNN-based network (in MVSplat [8]) and deconvolution-based network [41] (in DepthSplat [9]) and per-pixel Gaussian predictions are then performed using the upsampled features:

$$\mathcal{G} = \{(\mu_i, \Sigma_i, \alpha_i, \mathbf{c}_i)\}_{i=1}^{H \times W \times N} = \Psi_{\text{pred}}(\mathcal{F}_{\text{full}}, \mathcal{P}) \quad (3)$$

Where the position of the Gaussians are determined by the predicted depth and pixel location.

While straightforward, this pixel-aligned formulation introduces two critical limitations. First, the geometric accuracy of the reconstruction is critically dependent on the quality of the predicted depth map. After depth unprojecting features into 3D space, the lack of interaction with neighboring points within the 3D space significantly contributes to the generation of floaters. Second, the structure of the 3D representation is rigidly tied to the 2D image grid. The total number of Gaussians is fixed at  $|S| = H \times W \times N$ , which is often suboptimal and cause an over-densification of Gaussians on simple, texture-less surfaces and an insufficient number for representing complex geometry not captured at the pixel level. These observations reveal a fundamental bottleneck and motivate our proposed voxel-aligned framework, designed to decouple the 3D representation from the 2D pixel grid.

#### B. 3D Feature Construction

1) *Feature Extraction and Feature Matching:* For  $N$  input images, we first apply a weight-sharing ResNet [42]

backbone to each RGB image to obtain  $s \times$  downsampled feature maps. These features are then refined with cross-view attention that exchanges information with the two nearest neighboring views. For efficiency, this cross-attention is implemented using the Swin Transformer’s [43] local window attention. After this stage we obtain cross-view-aware Transformer features  $\{F_i\}_{i=1}^N$  ( $F^i \in R^{\frac{H}{p} \times \frac{W}{p} \times C}$ ), where  $C$  denotes the feature dimension.

Next, we build per-view cost volumes  $\{C_i\}_{i=1}^N$  using a plane-sweep strategy [44]. For each view  $i$ , we sample  $D$  candidate depths  $\{d_m\}_{m=1}^D$ , warp the feature from neighboring views to the reference view at each hypothesized depth, and compute pairwise feature similarities [8]. These similarities are aggregated by dot-product matching and stacked along the depth axis to form  $\{C_i\}_{i=1}^N$  ( $C_i \in R^{\frac{H}{s} \times \frac{W}{s} \times D}$ )

To produce robust, multi-view consistent depth estimates, a depth module fuses the monocular features  $\{F_{mono}^i\}_{i=1}^N$  ( $F_{mono}^i \in R^{\frac{H}{s} \times \frac{W}{s} \times C}$ ) with the cost volume  $C^i$  and regresses a dense per-pixel depth map  $D_i \in R^{H \times W}$ , which serves as a geometric prior for lifting image features into 3D space. These per-view features  $F^i$  and depths  $D_i$  are used in the next stage to construct 3D point clouds and voxel-based features for volumetric reasoning.

**2) Lifting to 3D Feature:** Given the predicted depth maps  $D_i$  and camera parameters, we conveniently aggregate different depth map views by transforming the point clouds into a global coordinate system. First each pixel  $(u, v)$  in image space is unprojected to a 3D point in the camera coordinate frame using the camera intrinsics. Then the 3D point is transformed into the world coordinate system via the corresponding extrinsic parameters, including the rotation matrix  $R_i$  and translation  $T_i$  vector.

$$\begin{aligned} P_{\text{world}} &= R_i P_{\text{cam}} + T_i \\ &= R_i \left( D_i(u, v) K^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \right) + T_i \quad (4) \end{aligned}$$

By repeating this process across all views, we obtain a dense  $|S| = H \times W \times N$  point cloud in world space, where each 3D point is associated with its corresponding image feature.

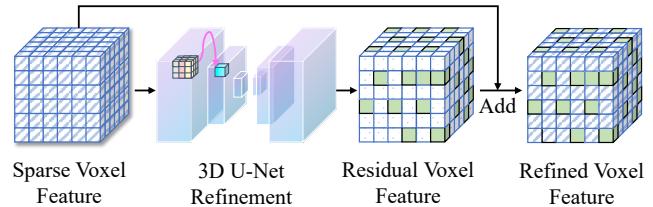
To convert the unstructured dense point cloud  $P$  into a structured volumetric representation, we voxelize the points [45]. For each 3D point  $p = (x_p, y_p, z_p)$  we compute integer voxel index  $(i, j, k)$  by dividing by the voxel size  $v_s$  and rounding.

$$i = \text{rnd} \left( \frac{x_p}{v_s} \right), \quad j = \text{rnd} \left( \frac{y_p}{v_s} \right), \quad k = \text{rnd} \left( \frac{z_p}{v_s} \right) \quad (5)$$

where  $\text{rnd}(\cdot)$  denotes rounding to the nearest integer.

Let  $S_{i,j,k}$  be the set of all points falling into voxel  $(i, j, k)$  and  $f_p$  be the image feature corresponding to each point  $p \in S_{i,j,k}$ . The features within this voxel are aggregated via average pooling along the channel dimension, resulting in the voxel feature, resulting in the voxel feature  $V_{i,j,k}$ :

$$V_{i,j,k} = \frac{1}{|S_{i,j,k}|} \sum_{p \in S_{i,j,k}} f_p \quad (6)$$



**Fig. 3: Architecture of Sparse 3D Decoder.** Sparse 3D features are fed into a 3D U-Net for processing, which predicts residual features for each voxel. These residual features are then added to the original 3D voxel features to obtain the refined features.

### C. Feature Refinement and Gaussian Prediction

**1) Feature Refinement:** To improve the spatial consistency and structural fidelity of the voxel representation, we apply an explicit voxel feature refinement stage as shown in Fig. 3. Given an input voxel grid  $V$  (with per-voxel feature vectors), a sparse convolutional 3D U-Net [14]  $\mathcal{R}$  predicts a residual voxel field  $R$ :

$$R = \mathcal{R}(V), \quad R_i \in \mathbb{R}^{\mathcal{V} \times C} \quad (7)$$

where  $\mathcal{V}$  denotes the set of occupied voxels and the refined voxel features are obtained by a residual update:

$$V' = V + R, \quad V'_i \in \mathbb{R}^{\mathcal{V} \times C} \quad (8)$$

The refinement network is implemented with hierarchical sparse 3D convolutional blocks, symmetric encoder-decoder stages, and upsampling layers connected by skip connections. This architecture enables multi-scale fusion of local and global geometric context while keeping computation efficient through sparsity. The residual formulation encourages the network to learn correction terms (fine geometric detail and consistency cues) rather than relearning the entire feature content, which empirically stabilizes training and preserves the coarse voxel information supplied by the lifting stage.

**2) Gaussian Prediction:** The output of our network for each voxel  $v$  is a set of learnable Gaussian parameters  $\{[\bar{\mu}_j, \bar{\alpha}_j, \Sigma_j, c_j] \in R^{18}\}$ . These include the offset of the Gaussian center  $\bar{\mu}_j$ , opacity  $\bar{\alpha}_j$ , covariance  $\Sigma_j$ , and spherical harmonic color representation  $c_j$ . To obtain the final rendering parameters, we apply the following transformations:

$$\begin{aligned} \mu_j &= r \cdot \sigma(\bar{\mu}_j) + \text{Center}_j, \\ \alpha_j &= \sigma(\bar{\alpha}_j) \end{aligned} \quad (9)$$

where  $\mu_j$  denotes the 3D center of the Gaussian,  $\text{Center}_j$  is the centroid of voxel  $v$  and  $r$  is a hyperparameter controlling the spatial extent of the offset (typically set to three times the voxel size),  $\sigma(\cdot)$  denotes the sigmoid function.

### D. Optimization

Our network predicts a collection of 3D Gaussians  $(\mu_v, \alpha_v, \Sigma_v, c_v)\}_{v \in \mathcal{V}}$ . These per-voxel Gaussians are subsequently used to synthesize images at novel camera poses.

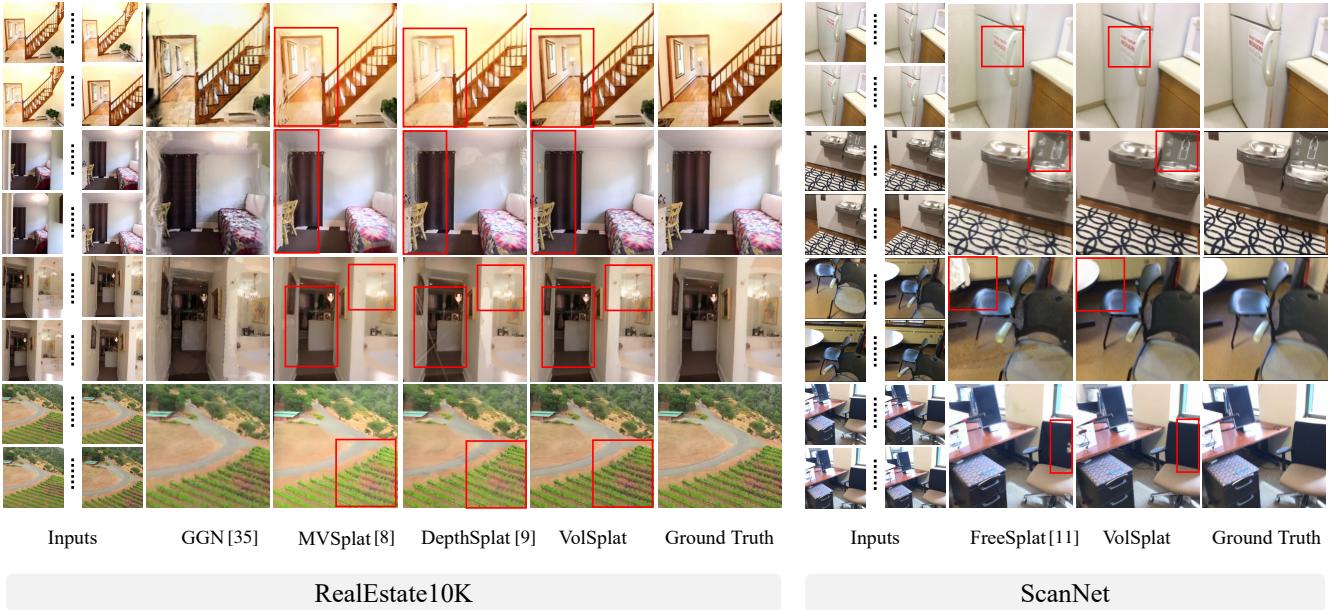


Fig. 4: **Qualitative comparison under multi-view input conditions.** The results on the left are from RealEstate10K [17], and the results on the right are from ScanNet [16]. Models are trained on each indicated dataset and tested on the same dataset. VolSplat achieved state-of-the-art results on both of them.

The network is trained end-to-end using ground-truth RGB images as supervision. For a forward pass that renders  $M$  novel views, we optimize a combined photometric and perceptual loss:

$$\mathcal{L} = \sum_{m=1}^M \left( \mathcal{L}_{\text{MSE}}(I_{\text{render}}^{(m)}, I_{\text{gt}}^{(m)}) + \lambda \mathcal{L}_{\text{LPIPS}}(I_{\text{render}}^{(m)}, I_{\text{gt}}^{(m)}) \right) \quad (10)$$

where  $M$  is the number of novel views to render in a single forward pass. The  $\mathcal{L}_{\text{LPIPS}}$  [46] loss weight  $\lambda$  is set to 0.05.

#### IV. EXPERIMENTS

##### A. Experimental Setup

**Datasets.** We train our method using two expansive datasets, RealEstate10K [17] and ScanNet [16], and evaluate its performance on the held-out test splits of both. For RealEstate10K, we adopt the conventional partition of 67,477 training scenes and 7,289 test scenes. As for ScanNet, which consists of 1,513 videos of indoor scenes, we follow past work [47], [48], [11] in using roughly 100 scenes for training and 8 scenes for evaluation. These datasets span a wide variety of environments, including indoor and outdoor real-estate walkthroughs (RealEstate10K), and real-world videos of numerous scenes suitable for indoor robot applications (ScanNet). We resize training and test images to  $256 \times 256$ .

**Baselines and metrics.** We benchmark VolSplat against several recent feed-forward methods for sparse-view novel view synthesis, including both pixel-aligned and enhanced pixel-aligned Gaussian splatting approaches. Pixel-aligned methods predict Gaussian parameters on a per-pixel basis in image space before unprojecting to 3D. These include

pixelSplat [6], MVsplat [8], FreeSplat [11], TranSplat [49] and DepthSplat [9]. In contrast to the enhanced pixel-aligned approach, Gaussian Graph Network (GGN) [35] refines the pixel-aligned approach by modeling the relationships between groups of predicted Gaussians across different views while building upon it.

In contrast to both pixel-aligned and enhanced pixel-aligned methods, VolSplat employs a voxel-aligned approach, predicting Gaussian primitives within a 3D voxel grid. This method aggregates multi-view evidence in 3D space, aligning Gaussian predictions to a voxel structure, which facilitates better geometric consistency and efficient redundancy reduction.

For quantitative evaluation, we adopt standard image quality metrics commonly used in NVS, including pixel-level Peak Signal-to-Noise Ratio (PSNR), patch-level Structural Similarity Index Measure (SSIM), and feature-level Learned Perceptual Image Patch Similarity (LPIPS).

**Implementation details.** We implement VolSplat using PyTorch [50] and optimize the model with the AdamW [51] optimizer and a cosine learning rate schedule. The monocular Vision Transformer backbone is implemented using the xFormers [52] library. For the pre-trained Depth Anything V2 [53] backbone, we use a lower learning rate of  $2 \times 10^{-6}$ , while other layers are trained with a learning rate of  $2 \times 10^{-4}$  following DepthSplat [9].

For experiments on the RealEstate10K [17] and ScanNet [16] dataset, we train the model for 150,000 iterations using  $4 \times$  A100 GPUs with a total batch size of 4. Following the setting of the baseline, we use  $256 \times 256$  as input resolution. In the training stage, the number of input views

TABLE I: **Quantitative comparisons on RealEstate10K [17].** The first five methods are all pixel-aligned methods, and GGN [35] performs post-processing on pixel-aligned Gaussians.  $\dagger$  Model uses more input views to get more Gaussians for pruning. "PGS" stands for "average number of per-view Gaussians".

Method	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	PGS
pixelSplat [6]	26.09	0.863	0.136	196608
MVSplat [8]	26.39	0.869	0.128	65536
TranSplat [49]	26.69	0.875	0.125	65536
DepthSplat [9]	27.47	0.889	0.114	65536
GGN $\dagger$ [35]	26.18	0.825	0.154	9375
<b>VolSplat</b>	<b>31.30</b>	<b>0.941</b>	<b>0.075</b>	65529

TABLE II: **Quantitative comparisons on ScanNet [16].** FreeSplat [11] and FreeSplat++ [54] performs Gaussian fusion after pixel-aligned Gaussian prediction.

Method	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	PGS
FreeSplat [11]	27.45	0.829	0.222	63668
FreeSplat++ [54]	27.45	0.829	0.223	69569
<b>VolSplat</b>	<b>28.41</b>	<b>0.906</b>	<b>0.127</b>	65406

is set to 6, and we evaluate the model’s performance with same numbers of input views.

### B. Experimental Results and Analysis

**Comparisons with SoTA models.** As shown in Tab. I and Tab. II, we report VolSplat’s performance compared to current mainstream pixel-aligned models [6], [8], [49], [11], [9] and their variant [35]. On both the RealEstate10K [17] and ScanNet [16] datasets, VolSplat achieves state-of-the-art results. Our experiments reveal a critical distinction between pixel-aligned and voxel-aligned paradigms. A key observation is that under sparse multi-view settings, all pixel-aligned models exhibit a significant degradation in performance. In contrast, VolSplat demonstrates promising performance to these challenging conditions. As illustrated in Fig. 4, images rendered by our method are largely free of the common floaters and artifacts that plague competing methods at object boundaries. This visual improvement stems directly from our model’s ability to resolve multi-view alignment issues within its 3D feature representation, resulting in cleaner edges and a more coherent 3D scene reconstruction.

**Cross-dataset generalization.** Following MVSplat [8], we assess the generalization capabilities of our model on unseen outdoor datasets to verify its broad reliability. To this end, we conducted a cross-dataset generalization experiment by taking our model pre-trained on the RealEstate10K [17] dataset and evaluating it directly on the ACID [55] dataset without any fine-tuning.

As demonstrated in Tab. III, VolSplat maintains significantly higher performance in this zero-shot transfer setting. We attribute this superior generalization to the inherent robustness of our voxel-aligned framework. Pixel-aligned models exhibit a much higher sensitivity to the variations in

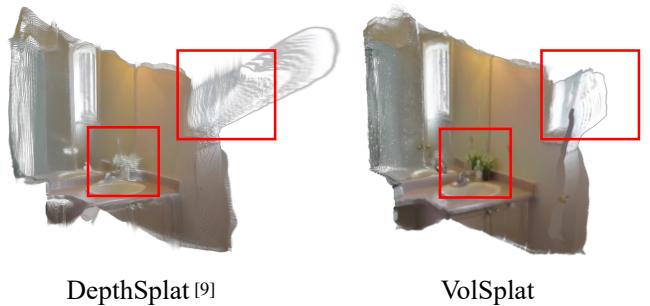


Fig. 5: **Visualization on Gaussians of DepthSplat [9] and VolSplat.** The Gaussian distribution in the pixel-aligned method is limited by the pixel distribution of the input view. Since the density must remain uniform, it cannot learn the complex geometry of the corners of the washbasin in the center and forces the Gaussians to be distributed to unnecessary edge areas. VolSplat is more realistic and reasonable.

TABLE III: **Cross-dataset generalization results on ACID [55].** All models were trained on RealEstate10K [17] and evaluated on the test set without any fine-tuning.

Method	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	PGS
pixelSplat [6]	27.64	0.830	0.160	196608
MVSplat [8]	28.15	0.841	0.147	65536
TranSplat [49]	28.17	0.842	0.146	65536
DepthSplat [9]	28.37	0.847	0.141	65536
<b>VolSplat</b>	<b>32.65</b>	<b>0.932</b>	<b>0.092</b>	65527

data complexity and distribution between different datasets. In contrast, VolSplat is less susceptible to these domain shifts.

**Analysis of Gaussian Density.** A fundamental principle of efficient 3D reconstruction is that the complexity of the representation should adapt to the complexity of the scene. Real-world environments contain a mix of simple, planar surfaces and intricate, high-frequency geometric details. An ideal model should allocate its descriptive capacity accordingly. However, pixel-aligned methods are inherently limited in this regard. Their paradigm of predicting one Gaussian per pixel results in a fixed number of primitives, predetermined by the input image resolution (e.g.,  $H \times W$  Gaussians from a reference view), regardless of whether the scene is a simple room or a complex outdoor environment. In stark contrast, our voxel-aligned framework enables adaptive control over the density of the 3D Gaussians. By predicting primitives based on the occupancy of 3D voxel features, VolSplat naturally allocates a higher concentration of Gaussians to regions of high geometric detail while using a sparser representation for simple or empty spaces.

This adaptive capability is quantitatively validated by the results we reported in Tab. I, Tab. II and Tab. III. Here, we analyze these findings in greater detail. The data shows that pixel-aligned methods consistently generate a large and constant number of Gaussians, irrespective of the scene content.

TABLE IV: **Analysis of voxel size.** The default voxel size is more suitable for real-world data than other options.

Voxel Size (cm)	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	PGS
0.05	31.03	0.939	0.077	65535
0.1 (default)	<b>31.30</b>	<b>0.941</b>	<b>0.075</b>	65529
0.5	28.73	0.916	0.132	64455
1	21.22	0.782	0.315	57806

TABLE V: **Ablation of sparse 3D decoder.** "w/ 3D CNN" means replacing the 3D U-Net with a sparse 3D CNN, and "w/o decoder" means removing the refinement stage.

Components	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	Memory(GB)
default	<b>31.30</b>	<b>0.941</b>	<b>0.075</b>	8.06
w/ 3D CNN	29.84	0.926	0.096	8.04
w/o decoder	29.34	0.921	0.092	<b>8.00</b>

This leads to significant redundancy and computational inefficiency, especially in simpler scenes. Conversely, Gaussians for VolSplat demonstrate significant variance across different scenes, confirming its ability to tailor the representation’s complexity, as shown in Fig. 5. Notably, VolSplat often achieves superior rendering quality with a more efficient and, in many cases, more compact set of Gaussians compared to the brute-force density of pixel-aligned approaches.

### C. Ablations Study

In this section, we study the properties of our key components on the RealEstate10K [17] dataset.

**Ablation of Voxel Size.** The voxel size is a critical hyperparameter in our framework, as it dictates the resolution of the 3D feature grid. This choice involves a fundamental trade-off between the fidelity of the geometric representation and computational resource consumption. In Tab. IV, we analyze this trade-off by comparing our default setting against configurations with different voxels.

Using a small voxel size increases the granularity of the 3D grid, allowing the model to capture finer geometric details. It comes at a significant cost, substantially increasing memory usage and processing time due to the cubic growth of the voxel volume. Conversely, employing a large voxel size reduces the computational footprint but results in a coarser quantization of the 3D space. Our default configuration strikes an effective balance, achieving state-of-the-art performance while maintaining manageable computational requirements.

**Ablation of Model Architecture.** Directly predicting Gaussians from the initial unprojected 3D features is less effective, particularly for challenging scenes with complex geometry or sparse viewpoints. To address this, we incorporate a 3D U-Net architecture to refine and enhance this raw feature volume. To validate the necessity and efficacy of this design, we conduct an ablation study with two variants: 1) removing the refinement module entirely, and 2) replacing the 3D U-Net with a standard 3D CNN.

The results, presented in Tab. V, confirm our architectural choices. Removing the refinement stage altogether leads

to a significant drop in performance, demonstrating that processing the initial voxel features is critical for producing a coherent 3D representation. While substituting our module with a sparse 3D CNN yields better results than no refinement, it still falls short of our full model’s performance. The multi-scale feature fusion inherent in the U-Net structure is crucial for capturing both fine-grained local details and broader spatial context.

## V. CONCLUSION

In this paper, we have addressed the fundamental limitations inherent in the prevailing pixel-aligned paradigm for feed-forward 3D Gaussian Splatting. We have identified that existing methods suffer from a rigid coupling of Gaussian density to input image resolution and a high sensitivity to multi-view alignment errors. To overcome these challenges, we have introduced VolSplat, a novel framework that fundamentally shifts the reconstruction process from 2D pixels to a 3D voxel-aligned space. By constructing 3D voxel feature and predicting Gaussians directly from this unified representation, our method effectively decouples the 3D scene from the constraints of the input views. This voxel-centric design enables adaptive control over Gaussian density according to scene complexity and inherently resolves alignment ambiguities, leading to more geometrically consistent and faithful reconstructions for downstream tasks.

## REFERENCES

- [1] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, “Nerf: Representing scenes as neural radiance fields for view synthesis,” *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021.
- [2] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, “3d gaussian splatting for real-time radiance field rendering.” *ACM Trans. Graph.*, vol. 42, no. 4, pp. 139–1, 2023.
- [3] A. Yu, V. Ye, M. Tancik, and A. Kanazawa, “pixelnerf: Neural radiance fields from one or few images,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 4578–4587.
- [4] Q. Wang, Z. Wang, K. Genova, P. P. Srinivasan, H. Zhou, J. T. Barron, R. Martin-Brualla, N. Snavely, and T. Funkhouser, “Ibrnet: Learning multi-view image-based rendering,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 4690–4699.
- [5] A. Chen, Z. Xu, F. Zhao, X. Zhang, F. Xiang, J. Yu, and H. Su, “Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo,” in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 14 124–14 133.
- [6] D. Charatan, S. L. Li, A. Tagliasacchi, and V. 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*, 2024, pp. 19 457–19 467.
- [7] K. Zhang, S. Bi, H. Tan, Y. Xiangli, N. Zhao, K. Sunkavalli, and Z. Xu, “Gs-lrm: Large reconstruction model for 3d gaussian splatting,” in *European Conference on Computer Vision*. Springer, 2024, pp. 1–19.
- [8] Y. Chen, H. Xu, C. Zheng, B. Zhuang, M. Pollefeys, A. Geiger, T.-J. Cham, and J. Cai, “Mvsplat: Efficient 3d gaussian splatting from sparse multi-view images,” in *European Conference on Computer Vision*. Springer, 2024, pp. 370–386.
- [9] H. Xu, S. Peng, F. Wang, H. Blum, D. Barath, A. Geiger, and M. Pollefeys, “Depthsplat: Connecting gaussian splatting and depth,” in *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025, pp. 16 453–16 463.
- [10] W. Wang, D. Y. Chen, Z. Zhang, D. Shi, A. Liu, and B. Zhuang, “Zpressor: Bottleneck-aware compression for scalable feed-forward 3dgs,” *arXiv preprint arXiv:2505.23734*, 2025.

- [11] Y. Wang, T. Huang, H. Chen, and G. H. Lee, “Freesplat: Generalizable 3d gaussian splatting towards free view synthesis of indoor scenes,” *Advances in Neural Information Processing Systems*, vol. 37, pp. 107 326–107 349, 2024.
- [12] Y. Chen, C. Zheng, H. Xu, B. Zhuang, A. Vedaldi, T.-J. Cham, and J. Cai, “Mvsplat360: Feed-forward 360 scene synthesis from sparse views,” *Advances in Neural Information Processing Systems*, vol. 37, pp. 107 064–107 086, 2024.
- [13] G. Kang, S. Nam, X. Sun, S. Khamis, A. Mohamed, and E. Park, “ilrm: An iterative large 3d reconstruction model,” *arXiv preprint arXiv:2507.23277*, 2025.
- [14] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, “3d u-net: learning dense volumetric segmentation from sparse annotation,” in *International conference on medical image computing and computer-assisted intervention*. Springer, 2016, pp. 424–432.
- [15] D. Shi, W. Wang, D. Y. Chen, Z. Zhang, J. Bian, B. Zhuang, and C. Shen, “Revisiting depth representations for feed-forward 3d gaussian splatting,” *arXiv preprint arXiv:2506.05327*, 2025.
- [16] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, “Scannet: Richly-annotated 3d reconstructions of indoor scenes,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 5828–5839.
- [17] T. Zhou, R. Tucker, J. Flynn, G. Fyffe, and N. Snavely, “Stereo magnification: Learning view synthesis using multiplane images,” *arXiv preprint arXiv:1805.09817*, 2018.
- [18] P. E. Debevec, C. J. Taylor, and J. Malik, “Modeling and rendering architecture from photographs: A hybrid geometry-and image-based approach,” in *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*, 2023, pp. 465–474.
- [19] D. Ji, J. Kwon, M. McFarland, and S. Savarese, “Deep view morphing,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2155–2163.
- [20] M. Levoy and P. Hanrahan, “Light field rendering,” in *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*, 2023, pp. 441–452.
- [21] J. T. Barron, B. Mildenhall, M. Tancik, P. Hedman, R. Martin-Brualla, and P. P. Srinivasan, “Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields,” in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 5855–5864.
- [22] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman, “Zip-nerf: Anti-aliased grid-based neural radiance fields,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 19 697–19 705.
- [23] Z. Fan, K. Wang, K. Wen, Z. Zhu, D. Xu, Z. Wang, et al., “Light-gaussian: Unbounded 3d gaussian compression with 15x reduction and 200+ fps,” *Advances in neural information processing systems*, vol. 37, pp. 140 138–140 158, 2024.
- [24] Z. Zhu, Z. Fan, Y. Jiang, and Z. Wang, “Fsgs: Real-time few-shot view synthesis using gaussian splatting,” in *European conference on computer vision*. Springer, 2024, pp. 145–163.
- [25] H. Liu, Y. Wang, C. Li, R. Cai, K. Wang, W. Li, P. Molchanov, P. Wang, and Z. Wang, “Flexgs: Train once, deploy everywhere with many-in-one flexible 3d gaussian splatting,” in *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025, pp. 16 336–16 345.
- [26] D. Meagher, “Geometric modeling using octree encoding,” *Computer graphics and image processing*, vol. 19, no. 2, pp. 129–147, 1982.
- [27] A. E. Kaufman and K. Mueller, “Overview of volume rendering.” *The visualization handbook*, vol. 7, pp. 127–174, 2005.
- [28] C. H. Koneputugodage, Y. Ben-Shabat, and S. Gould, “Octree guided unoriented surface reconstruction,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 16 717–16 726.
- [29] Y. Zhou and O. Tuzel, “Voxelnet: End-to-end learning for point cloud based 3d object detection,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4490–4499.
- [30] G. Riegler, A. Osman Ulusoy, and A. Geiger, “Octnet: Learning deep 3d representations at high resolutions,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 3577–3586.
- [31] S. Fridovich-Keil, A. Yu, M. Tancik, Q. Chen, B. Recht, and A. Kanazawa, “Plenoxels: Radiance fields without neural networks,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 5501–5510.
- [32] S. Fridovich-Keil, G. Meanti, F. R. Warburg, B. Recht, and A. Kanazawa, “K-planes: Explicit radiance fields in space, time, and appearance,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 12 479–12 488.
- [33] T. Lu, M. Yu, L. Xu, Y. Xiangli, L. Wang, D. Lin, and B. Dai, “Scaffold-gs: Structured 3d gaussians for view-adaptive rendering,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 20 654–20 664.
- [34] K. Ren, L. Jiang, T. Lu, M. Yu, L. Xu, Z. Ni, and B. Dai, “Octreegs: Towards consistent real-time rendering with lod-structured 3d gaussians,” *arXiv preprint arXiv:2403.17898*, 2024.
- [35] S. Zhang, X. Fei, F. Liu, H. Song, and Y. Duan, “Gaussian graph network: Learning efficient and generalizable gaussian representations from multi-view images,” *Advances in Neural Information Processing Systems*, vol. 37, pp. 50 361–50 380, 2024.
- [36] L. Jiang, Y. Mao, L. Xu, T. Lu, K. Ren, Y. Jin, X. Xu, M. Yu, J. Pang, F. Zhao, et al., “Anysplat: Feed-forward 3d gaussian splatting from unconstrained views,” *arXiv preprint arXiv:2505.23716*, 2025.
- [37] B. Ye, S. Liu, H. Xu, X. Li, M. Pollefeys, M.-H. Yang, and S. Peng, “No pose, no problem: Surprisingly simple 3d gaussian splats from sparse unposed images,” *arXiv preprint arXiv:2410.24207*, 2024.
- [38] G. Kang, J. Yoo, J. Park, S. Nam, H. Im, S. Shin, S. Kim, and E. Park, “Selfsplat: Pose-free and 3d prior-free generalizable 3d gaussian splatting,” in *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025, pp. 22 012–22 022.
- [39] C. Ziwen, H. Tan, K. Zhang, S. Bi, F. Luan, Y. Hong, L. Fuxin, and Z. Xu, “Long-lrm: Long-sequence large reconstruction model for wide-coverage gaussian splats,” *arXiv preprint arXiv:2410.12781*, 2024.
- [40] S. Miao, J. Huang, D. Bai, X. Yan, H. Zhou, Y. Wang, B. Liu, A. Geiger, and Y. Liao, “Evolsplat: Efficient volume-based gaussian splatting for urban view synthesis,” in *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025, pp. 11 286–11 296.
- [41] A. Odena, V. Dumoulin, and C. Olah, “Deconvolution and checkerboard artifacts,” *Distill*, 2016. [Online]. Available: <http://distill.pub/2016/deconv-checkerboard>
- [42] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
- [43] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, “Swin transformer: Hierarchical vision transformer using shifted windows,” in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 10 012–10 022.
- [44] H. Xu, J. Zhang, J. Cai, H. Rezatofighi, F. Yu, D. Tao, and A. Geiger, “Unifying flow, stereo and depth estimation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 11, pp. 13 941–13 958, 2023.
- [45] T. Wang, X. Mao, C. Zhu, R. Xu, R. Lyu, P. Li, X. Chen, W. Zhang, K. Chen, T. Xue, et al., “Embodiedscan: A holistic multi-modal 3d perception suite towards embodied ai,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 19 757–19 767.
- [46] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, “The unreasonable effectiveness of deep features as a perceptual metric,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 586–595.
- [47] X. Zhang, S. Bi, K. Sunkavalli, H. Su, and Z. Xu, “Nerfusion: Fusing radiance fields for large-scale scene reconstruction,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 5449–5458.
- [48] Y. Gao, Y.-P. Cao, and Y. Shan, “Surfelnerf: Neural surfel radiance fields for online photorealistic reconstruction of indoor scenes,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2023, pp. 108–118.
- [49] J. Kim, J. Noh, D.-G. Lee, and A. Kim, “Transplat: Surface embedding-guided 3d gaussian splatting for transparent object manipulation,” *arXiv preprint arXiv:2502.07840*, 2025.
- [50] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al., “Pytorch: An imperative style, high-performance deep learning library,” *Advances in neural information processing systems*, vol. 32, 2019.
- [51] I. Loshchilov and F. Hutter, “Decoupled weight decay regularization,” *arXiv preprint arXiv:1711.05101*, 2017.
- [52] B. Lefauveaux, F. Massa, D. Liskovich, W. Xiong, V. Caggiano, S. Naren, M. Xu, J. Hu, M. Tintore, S. Zhang, P. Labatut, D. Haziza,

- L. Wehrstedt, J. Reizenstein, and G. Sizov, “xformers: A modular and hackable transformer modelling library,” <https://github.com/facebookresearch/xformers>, 2022.
- [53] L. Yang, B. Kang, Z. Huang, Z. Zhao, X. Xu, J. Feng, and H. Zhao, “Depth anything v2,” *arXiv:2406.09414*, 2024.
- [54] Y. Wang, T. Huang, H. Chen, and G. H. Lee, “Freesplat++: Generalizable 3d gaussian splatting for efficient indoor scene reconstruction,” *arXiv preprint arXiv:2503.22986*, 2025.
- [55] A. Liu, R. Tucker, V. Jampani, A. Makadia, N. Snavely, and A. Kanazawa, “Infinite nature: Perpetual view generation of natural scenes from a single image,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 14458–14467.