PointSplat: Compact Gaussian Splatting via Human-Centric Prediction

Yujie Guo[1], Yudong Jin[1], Lingteng Qiu[3], Zehong Shen[1], Zhen Xu[1], Jing Zhang[2], Xianchao Shen[2], Hujun Bao[1], Sida Peng[1], and Xiaowei Zhou[1]✉

[1] State Key Lab of CAD&CG, Zhejiang University, Hangzhou, China
[2] ByteDance, Beijing, China
[3] The Chinese University of Hong Kong, Shenzhen, China

Abstract. Producing 3D human representations from input views on the fly is essential for immersive live streaming systems, where representation compactness is as critical as high fidelity given limited computational power and transmission bandwidth. Although recent feed-forward reconstruction methods achieve impressive quality through the view-centric prediction of 3D representations, they repeatedly encode the same subject content across multiple views, leading to significant inter-view redundancy. Our key insight is to perform predictions directly in 3D space, enabling the network to learn and produce a highly compact representation. To this end, we propose PointSplat, a novel human-centric approach that directly infers Gaussian primitives from an input point set. The proposed method first estimates a coarse geometric proxy and performs ray casting to prune redundant points and establish explicit 2D–3D correspondences. Subsequently, it employs a Point-Image Transformer to fuse appearance and geometry features, predicting Gaussian attributes in a single forward pass. This design restricts predictions to foreground regions of interest, substantially reducing the total number of Gaussians while improving novel-view rendering quality. Extensive experiments demonstrate that PointSplat achieves higher efficiency and quality while exhibiting strong robustness to variations in view count and image resolution across multiple datasets.

Keywords: 3D Representation · Compactness · Feed–Forward Model

1 Introduction

This paper addresses the problem of producing compact 3D human representations from input views, which is critical for immersive live streaming systems, holographic communication, and related applications. These real-time scenarios demand high-fidelity and high-speed reconstruction, while also requiring low hardware cost and efficient transmission to deliver an immersive experience to users [9,19,20,26,27]. Traditional light-field-based approaches [9,20,25,28], which employ dense camera arrays for view interpolation, can achieve high-quality

✉ Corresponding author.

Fig. 1: Our method synthesizes high-fidelity novel views from sparse views by predicting a compact 3D Gaussian [17] representation. The bottom row shows representative results of our method on diverse human performances.

rendering with low latency. However, their reliance on costly capture hardware severely limits scalability and deployment in practical settings.

Recently, neural reconstruction methods [2,4,17,34,43,48,51,55] have demonstrated impressive view synthesis quality from sparse views, significantly reducing the deployment cost of capture hardware. These approaches typically adopt a *view-centric* representation, where pixel-aligned 3D features or Gaussian maps are predicted for each input view. While this design enables fast reconstruction and real-time rendering from limited inputs, it inherently introduces inter-view redundancy, where the same object is repeatedly represented across different views. Such redundancy leads to rapidly increasing computation and transmission overhead as the number of views or rendering resolution grows, limiting scalability under high-resolution and arbitrary-viewpoint settings.

To overcome the redundancy inherent in view-centric representations, we propose an *human-centric* feed-forward approach that directly infers a compact Gaussian representation in 3D space, as illustrated in Figure 2. This design not only reduces the number of Gaussians required but also improves the quality of novel view synthesis from sparse inputs. Our pipeline first extracts appearance features from input images together with Plücker ray embeddings. A coarse geometric proxy is then estimated via space carving [18] and voxelized to aggregate point-level features. Finally, a Point-Image Transformer predicts the Gaussian parameters for each point in 3D space. Benefiting from the geometric proxy, our method focuses prediction on 3D regions of interest, fully exploiting the advantages of human-centric prediction, which leads to substantially higher efficiency and improved robustness to variations in input view numbers and resolutions.

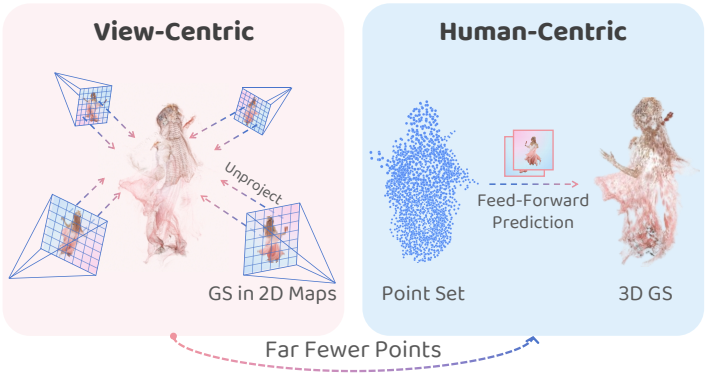

Fig. 2: Comparison between view-centric and human-centric prediction. View-centric methods first predict pixel-aligned Gaussian maps for input views and then unproject them into 3D space. In contrast, human-centric methods directly infer Gaussian primitives in 3D space by fusing multi-view observations.

However, a remaining challenge is that the geometric hull often contains redundant interior points and lacks fine details, which degrades both prediction efficiency and quality. To address this issue, we design a ray-casting mechanism to identify surface points within the geometric proxy. This mechanism effectively removes internal points that do not contribute to rendering, thus improving prediction efficiency. Moreover, by establishing explicit 2D–3D correspondences, it enriches point embeddings and enables more effective cross-modal interaction between image and point features, ultimately enhancing the quality of Gaussian prediction.

We conduct extensive experiments on various real-world (DNA-Rendering [5], ActorsHQ [13], and PKU-DyMVHumans [56]) and synthetic (THuman2.0 [49], RenderPeople [32]) benchmarks, demonstrating that our method outperforms existing view-centric approaches in both efficiency and quality. Notably, our approach achieves superior novel-view synthesis quality while using only about 33% of the number of Gaussians required by view-centric methods. Moreover, it exhibits strong robustness across varying input resolutions and view counts, highlighting its potential for practical applications. In summary, our contributions are as follows:

1. We introduce PointSplat, a novel human-centric feed-forward approach that directly infers a compact Gaussian representation in 3D space, eliminating inter-view redundancy in view-centric methods.
2. We propose a Point-Image Transformer that incorporates a ray-casting mechanism to remove redundant points and build explicit 2D–3D correspondences, unifying geometry and appearance modeling for efficient and accurate Gaussian prediction.
3. We demonstrate that our approach consistently outperforms prior methods in both efficiency and reconstruction quality, while exhibiting strong robustness to variations in input view number and resolution.

2 Related Work

Feed-forward 3D Reconstruction Feed-forward 3D reconstruction methods can be broadly categorized into implicit and explicit 3D representation learning approaches. On one hand, implicit methods [23,37,47] leverage continuous functions that map input observations to target views. Recently, LVSM [15] learns to render novel views with minimal 3D bias, but faces challenges in rendering efficiency, which is required for real-time applications. On the other hand, explicit methods represent scenes with discrete geometric structures (e.g., point clouds), allowing direct geometric supervision and efficient rendering. DUSt3R [38] unifies monocular and stereo reconstruction through pairwise pointmap regression, offering flexibility but accumulating errors across multiple views. [22,36,39] extend this paradigm with unified transformer architectures that infer complete 3D attributes in a feed-forward manner. However, their inherently view-centric design leads to inter-view redundancy and inconsistency in the final 3D representation.

View-centric Representations View-centric methods typically reconstruct explicit geometry by predicting pixel-aligned representations for each input image. Binocular methods [55] unproject Gaussian maps from adjacent views into 3D Gaussians, while MVS methods (e.g., Pixelsplat [2], MVSplat [4], and DepthSplat [43]) use intermediate constraints and then unproject Gaussian maps from multiple views into a 3D representation. LRM family [34, 48, 51] predicts per-view pixel-aligned 3D Gaussians from sparse images via transformers, achieving fast feed-forward reconstruction in an end-to-end manner. However, a common issue with these methods is that overlapping per-view predictions often cover the same regions multiple times, leading to redundancy in the final representation. Post-processing strategies like Gaussian pruning [54, 57] or feed-forward compression [3] are proposed to reduce redundancy but introduce ambiguities in predicting dominant or low-opacity results on specific views. Similarly, generative approaches [8,33] generate Gaussian maps or multi-view images before lifting them into 3D, producing diverse results but still suffering from redundancy and ambiguities due to the view-centric design.

Native 3D Representations Several distinct approaches have been developed for 3D reconstruction, each with unique strengths and limitations. Radiance field-based methods [44] build on the implicit representation of 3D geometry using dense input point clouds. However, they are computationally expensive and depend heavily on dense, structured data. KPlanes-based methods [11, 21] use multi-plane features for efficient 3D reconstruction, yet learning consistent view-to-plane mappings remains challenging. For human-centric applications, approaches like LHM [30] leverage a parametric model as a shape prior, enabling the prediction of 3D Gaussians. While this offers high precision for human models, its reliance on SMPL restricts its applicability to complicated scenes.

Voxel-based optimization methods [7, 24, 31] further explore explicit 3D representations by discretizing scenes into voxel grids. Recently, AnySplat [14] employs differentiable voxelization to aggregate 3D Gaussians in a feed-forward

manner. Although effective in reducing redundancy, such designs rely on additional geometric supervision, which may limit their adaptability to diverse scenarios.

Beyond the reconstruction task, recent advances in 3D generation [35, 41, 50, 52] have explored learning object-centric representations directly from data. These methods typically employ VAEs to encode point clouds into a latent space for generating Signed Distance Functions (SDFs). While such generative frameworks excel at producing coherent 3D structures, they are not designed for multi-view reconstruction and struggle to ensure consistency with 2D observations. Nonetheless, they highlight the potential of compact and unified 3D representations—a goal our approach aims to achieve in a reconstruction context.

3 Method

Given a set of calibrated images and corresponding object masks, our method aims to reconstruct a compact 3D Gaussian Splatting (3DGS) representation in a feed-forward manner. The overview of the proposed method is illustrated in Figure 3. It begins with the extraction of appearance features from 2D images and camera rays (Section 3.1). Next, a coarse geometric proxy is estimated to ensure visibility consistency. To reduce redundancy and establish 2D–3D correspondence, we employ ray casting with ray embeddings, which unify 2D and 3D representations (Section 3.2). Subsequently, a Point-Image Transformer fuses appearance and geometry features and decodes them into Gaussian parameters (Section 3.3). The entire network is trained end-to-end, requiring only RGB supervision (Section 3.4).

3.1 Appearance Feature Extraction

We extract 3D-aware appearance features from 2D images by embedding Plücker rays into the appearance representation.

Plücker Ray Embedding Each pixel is represented by its Plücker ray, encoding the ray direction and position. The Plücker coordinates are defined as:

$$\mathbf{f}_{\text{ray}} = [\mathbf{d}, \mathbf{o} \times \mathbf{d}] \in \mathbb{R}^6, \tag{1}$$

where $\mathbf{o}$ and $\mathbf{d}$ are the camera origin and ray direction, respectively. These are concatenated with the pixel color $\mathbf{c}$ to form:

$$\mathbf{f}_{\text{pixel}} = [\mathbf{c}, \mathbf{f}_{\text{ray}}] \in \mathbb{R}^9. \tag{2}$$

Images are patchified into a token sequence:

$$\mathbf{t}_{\text{images}} = \text{Patchify}(\mathbf{f}_{\text{pixel}}) \in \mathbb{R}^{N \times C}. \tag{3}$$

Masked Token Sampling We apply mask-guided top-n sampling to select informative tokens. The score for each token is computed by summing the mask

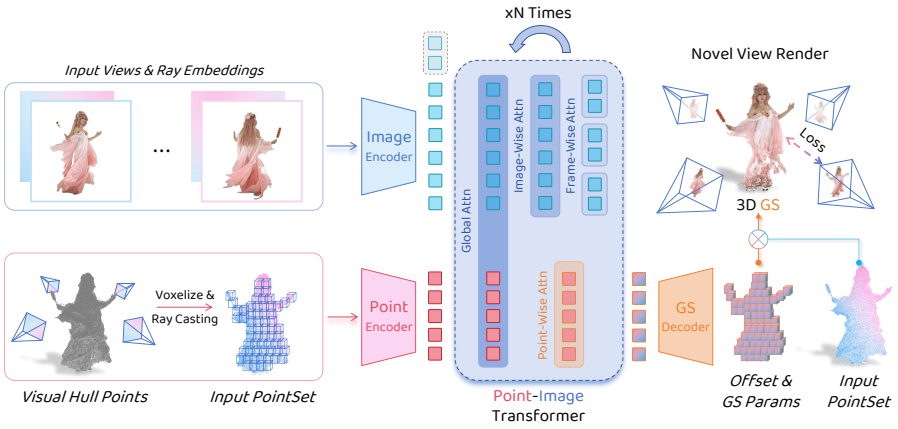

Fig. 3: Pipeline Overview. Given calibrated multi-view images and masks, PointSplat reconstructs a compact 3D Gaussian (3DGS) representation in a feed-forward step. We first extract appearance features from input images together with Plücker ray embeddings. A coarse geometric proxy is then estimated via space carving, followed by voxelization for structured representation. For each anchor view, we perform ray casting to select surface points and build explicit 2D–3D correspondences. A Point-Image Transformer then integrates appearance and geometry features and predicts point offsets and the remaining Gaussian parameters, constructing the final 3DGS representation. The entire framework is trained end-to-end under RGB supervision only, enabling efficient reconstruction.

values within its patch:

$$s_i = \sum_{j \in \text{Patchify}(i)} m_j. \tag{4}$$

The top-n tokens are then selected based on these scores:

$$\mathbf{t}_{\text{appearance}} = \text{SelectTop-}n(\{t_i\}, \{s_i\}). \tag{5}$$

Appearance tokens are then projected into a hidden dimension C using a linear layer:

$$\mathbf{T}_{\text{appearance}} = \text{Linear}(\mathbf{t}_{\text{appearance}}) \in \mathbb{R}^{n \times C}. \tag{6}$$

3.2 Geometry Feature Extraction

We aim to obtain reliable 3D geometry that can serve as a spatial reference for appearance features. Unlike methods that rely solely on depth estimation, which often suffer from noise and inconsistency, our approach combines space carving and ray casting to build accurate 2D–3D correspondences.

Geometry Proxy We first construct a visibility-consistent proxy of the object using a space carving algorithm [18]. Given the calibrated masks from multiple views, we iteratively remove points that are not visible in all masks, producing a

dense visual hull that approximates the object surface. The resulting point cloud is denoted as $\mathbf{p}$ and serves as the geometric basis for subsequent voxelization.

Voxelization and Ray Casting To remove redundant interior points and establish explicit 2D–3D correspondences, we perform ray casting from each anchor view. A voxel grid is first constructed to enclose the input point cloud $\mathbf{p}$, forming a structured spatial partition of the scene. We denote the resulting voxelized points as $\mathbf{p}_v$. For each pixel ray, parameterized as $\mathbf{r}(t) = \mathbf{c} + t\mathbf{d}$, with $\mathbf{c}$ as the camera center and $\mathbf{d}$ as the ray direction, we apply a Digital Differential Analyzer (DDA) to efficiently traverse the grid. During traversal, for each voxel intersected by the ray, we identify the point within that voxel that lies closest to the ray by minimizing the point-to-ray distance:

$$d(\mathbf{p}_v, \mathbf{r}) = \frac{\|(\mathbf{p}_v - \mathbf{c}) \times \mathbf{d}\|}{\|\mathbf{d}\|}, \quad \mathbf{p}_v^* = \arg\min_{\mathbf{p}_v \in v} d(\mathbf{p}_v, \mathbf{r}). \tag{7}$$

Once the ray intersects a voxel, we select a fixed number of points s from the voxel. Voxels not intersected by any ray are discarded. We denote the resulting set of points as $\mathbf{p}_s^*$. This facilitates explicit image-to-point correspondences while removing redundant internal geometry.

Point Encoding Each voxel, which contains a subset of 3D points, is first hashed to identify the occupied regions in the voxel grid. For every occupied voxel, we apply a sinusoidal positional encoding to the points it contains. The encoded point features $\gamma(\mathbf{p}_s^*)$ are then concatenated with the corresponding ray feature $\mathbf{f}_{\text{ray}} \in \mathbb{R}^6$ cast from the 2D anchor view. This representation is further projected through a linear layer followed by layer normalization [1] to produce the geometry token:

$$\mathbf{T}_{\text{geometry}} = \text{LN}\left(\text{Linear}\left([\gamma(\mathbf{p}_s^*); \mathbf{f}_{\text{ray}}]\right)\right) \in \mathbb{R}^{M \times C}. \tag{8}$$

Here, $\gamma : \mathbb{R}^3 \to \mathbb{R}^{3L}$ applies an L-frequency sinusoidal encoding to spatial coordinates, and M denotes the number of occupied voxels. Through this process, we obtain a unified embedding that jointly represents appearance and geometry, facilitating effective cross-modal modeling.

3.3 3DGS Prediction

Point-Image Transformer The fusion of 2D and 3D information is achieved through a transformer tailored for point-image interaction. It takes as input the appearance tokens $\mathbf{T}_{\text{appearance}}$ and the geometric tokens $\mathbf{T}_{\text{geometry}}$, and outputs the fused Gaussian Splatting tokens $\mathbf{T}_{\text{Gaussians}}$:

$$\mathbf{T}_{\text{Gaussians}} = \text{Network}(\mathbf{T}_{\text{appearance}}, \mathbf{T}_{\text{geometry}}), \tag{9}$$

where the network employs alternating attention [36] to integrate global, point-wise, and image-wise information.

- **Global Attention** Self-attention is performed to capture holistic context by enabling interactions among all tokens (appearance and geometry).
- **Point-wise Attention** We then use full self-attention to enhance local geometric coherence by focusing on neighboring point sets.
- **Image-wise Attention** Self-attention is further used to model intra-view relationships (frame-wise attention) and inter-view dependencies (cross-view attention) for multi-view aggregation.

The resulting $\mathbf{T}_{\text{Gaussians}}$ serves as the input for the 3DGS parameter prediction module.

3DGS Parameter Prediction The fused token features are decoded into Gaussian splatting parameters, including position offset, scale, rotation, spherical harmonics (SH) coefficients, and opacity.

The decoding process is formulated as:

$$\mathbf{T}_{\text{Gaussians}} : \{g_i\}_{i=1}^{M} \;\rightarrow\; \left\{ \left\{ \left(o_i^k,\, c_i^k,\, s_i^k,\, \alpha_i^k,\, r_i^k\right) \right\}_{k=1}^{K} \right\}_{i=1}^{M}, \tag{10}$$

where each g_i is decoded into K Gaussians with position offsets o, colors c, scales s, opacities α, and rotations r.

Each parameter is predicted using a separate linear layer with different activation functions to ensure valid ranges. For each property, we use a variable $v \in \{o, c, s, \alpha, r\}$. The prediction can be written as:

$$v = f_v\big(\text{Linear}_v(\mathbf{T}_{\text{Gaussians}})\big) \in \mathbb{R}^{K \times D_v} \tag{11}$$

where f_v is the activation function for property v, and D_v is the dimension of property v. The predicted offsets are added to the input point set positions to obtain the final Gaussian centers.

$$\mathbf{p}_{\text{final}} = \mathbf{p}_s^* + o \tag{12}$$

3.4 Training

We train the entire framework end-to-end using only RGB supervision, enabling joint refinement of geometry and appearance without additional priors. The overall objective combines pixel-wise and perceptual losses:

$$\mathcal{L} = \lambda_{\text{L1}}\mathcal{L}_{\text{L1}} + \lambda_{\text{LPIPS}}\mathcal{L}_{\text{LPIPS}}, \tag{13}$$

where $\mathcal{L}_{\text{L1}}$ enforces photometric consistency and $\mathcal{L}_{\text{LPIPS}}$ promotes perceptual similarity. λ_{L1} and λ_{LPIPS} balance the two terms.

4 Experiments

4.1 Implementation Details

We normalize the scene to a $2 \times 2 \times 2$ cube. The appearance branch operates on 4×4 image patches, while the geometry branch uses voxels of size 0.005. We

use a Point-Image Transformer with 4 blocks and a hidden size of 1024. During decoding, we set the number of per-token Gaussians K to 16. QK-Norm [10] is applied to every attention layer for stability, following prior work [15, 51]. The initial learning rate is 2×10^{-4} with a linear warmup over the first 10% of iterations. λ_{L1} and λ_{LPIPS} are set to 1.0 and 1.0, respectively. Training efficiency is improved using FlashAttention [6], gradient checkpointing, and mixed precision. We train for 300k iterations with a total batch size of 32 on NVIDIA H20 GPUs. All inference timing is measured on an NVIDIA A6000 (48GB).

4.2 Datasets and Metrics

We train our model on DNA-Rendering [5], using approximately 1,000 sequences for training and 16 sequences for evaluation, following prior work [16], unless otherwise specified. To assess performance in multi-human scenarios, we evaluate the proposed method using the PKU-DyMVHumans [56] dataset. Furthermore, we investigate the zero-shot generalization capability of the model on the ActorsHQ [13] dataset, which consists of 12 dynamic human sequences. Following prior work on human novel-view synthesis [23,29,55], we report PSNR, SSIM [40], and LPIPS [53] computed on foreground regions defined by the subject's bounding box. For efficiency, we also report the number of Gaussians and the per-frame inference time.

4.3 Comparison with State-of-the-art Methods

Baselines. We compare the proposed approach against state-of-the-art feed-forward NVS methods. These include view-centric methods, namely GS-LRM [51], DepthSplat [43], LVSM [15], and GPS-Gaussian [55], the pose-free baseline AnySplat [14], as well as human-specific approaches, such as RoGSplat [42] and LHM [30]. For DNA-Rendering comparisons, all trainable baselines are trained or fine-tuned on DNA-Rendering with a matched optimization budget. For Table 2, we train a separate model on THuman2.0 with the same training-set size, since GPS-Gaussian and RoGSplat are trained on THuman2.0 and require ground-truth depth supervision. Furthermore, we compare our approach with optimization-based techniques, specifically 4DGS [45, 46] and GauHuman [12], along with the state-of-the-art generative frameworks LGM [34] and Diffuman4D [16].

Results on different datasets. We report results on the DNA-Rendering and ActorsHQ datasets in Table 1 and Fig. 4. All methods are evaluated at 512×512 resolution using 8 input views, with pure PyTorch implementations (without custom CUDA kernels for acceleration). As shown by both quantitative metrics and qualitative comparisons, our method achieves the best overall performance across both datasets. Our approach directly infers compact 3D Gaussians, enabling consistent novel view synthesis without redundant structures or unmodeled regions. In contrast, GS-LRM, GPS-Gaussian, and DepthSplat exhibit redundancy across views, leading to floating artifacts and noisy points when rendering novel views. LVSM can model continuous novel views but tends

Table 1: Quantitative comparison on the DNA-Rendering and ActorsHQ datasets. $\times N$ denotes that the metric scales linearly with N target views. "GS-Num" represents the number of Gaussians, and "Time" refers to inference time (in seconds). All methods are evaluated with eight 512×512 input views. The best results are highlighted in bold.

Method	**DNA-Rendering** [5]			**ActorsHQ** [13]			**GS-Num**	**Time**
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓		
GS-LRM [51]	18.25	0.582	0.364	16.96	0.675	0.299	204k	**0.32**
AnySplat [14]	20.97	0.790	0.143	19.11	0.699	0.238	86k	0.53
GPS-Gaussian [55]	22.35	0.797	0.172	23.81	0.887	0.084	51k × N	0.07 × N
DepthSplat [43]	23.98	0.821	0.131	22.39	0.797	0.157	212k	0.45
LVSM [15]	23.24	0.802	0.135	24.00	0.804	0.128	-	1.0 × N
Ours	**27.18**	**0.891**	**0.071**	**27.62**	**0.887**	**0.084**	**71k**	0.40

Table 2: Quantitative comparison on synthetic datasets. The evaluation metrics are computed across the entire images, following the protocol established in [42].

Method	**View**	**THuman2.0** [49]			**RenderPeople** [32]			**GS-Num**
		PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	
RoGSplat [42]	4	28.94	0.961	0.043	25.12	0.938	0.066	135k
GPS-Gaussian	6	27.79	0.960	0.039	25.11	0.932	0.068	67k × N
Ours	4	**30.68**	**0.969**	**0.021**	**26.70**	**0.954**	**0.051**	**89k**
DepthSplat	8	26.32	0.951	0.048	26.90	0.958	0.051	270k
GPS-Gaussian	8	31.08	0.970	0.032	29.97	0.970	0.035	67k × N
Ours	8	**34.30**	**0.980**	**0.015**	**32.36**	**0.974**	**0.026**	**75k**

to produce over-smoothed, mosaic-like results and suffers from slow rendering. AnySplat operates without pose priors, which partly explains its lower fidelity on this human-centric benchmark. We further demonstrate our method's effectiveness on multi-human-centric dynamic scenes from the PKU-DyMVHumans dataset in Fig. 5.

Results on high resolution. We further evaluate at 1024×1024 on the DNA-Rendering test set in a zero-shot setting, without additional fine-tuning. For the 4DGS baseline [46], we adopt the reconstruction and rendering framework described in [45] and synthesize novel views for evaluation. As shown in Table 3, our method maintains clear superiority over other feed-forward methods at higher resolution, demonstrating robust zero-shot scaling to detailed human performances. Our method achieves comparable PSNR, SSIM, and LPIPS while being much faster compared to Diffuman4D. Moreover, even when compared with state-of-the-art dense-view (e.g., 44 views) optimization-based 4DGS methods,

Fig. 4: Qualitative comparison across datasets. Our method produces compact and complete 3D reconstructions that yield clean and consistent novel-view renderings. View-centric baselines (e.g., GPS-Gaussian, DepthSplat) show redundant or missing geometry, resulting in floating artifacts, while LVSM exhibits over-smoothed and mosaic-like appearances.

it delivers similar visual fidelity with a fraction of the computational cost. Note that LVSM cannot handle this resolution due to memory constraints.

Results on different view numbers. We further evaluate the zero-shot generalization ability by varying the number of input views. As shown in Table 4, our method consistently outperforms all baselines under different view counts, demonstrating strong robustness to view number variation, which can be attributed to its object-centric design.

4.4 Ablation Study

Effects of Modules. We conduct ablation studies to investigate the contribution of each component in our approach, as reported in Table 5 and Table 6.

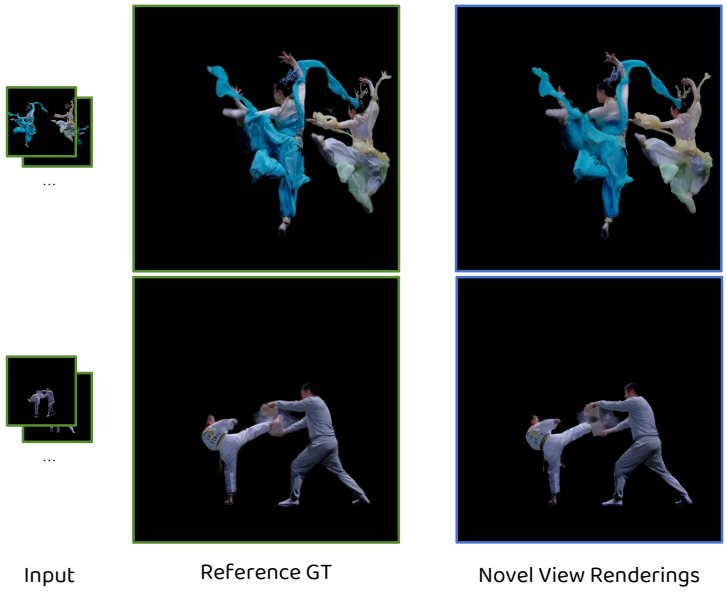

Fig. 5: Qualitative examples of novel view synthesis on the DyMVHumans dataset. Our method generates high-quality novel views for multi-human scenes with diverse poses and appearances using eight input views.

Table 3: Quantitative comparison at 1024 resolution on the DNA-Rendering test set. "opt" indicates that the method is optimization-based, whereas "prior" denotes the utilization of templates such as SMPL.

Method	Type	View	PSNR↑	SSIM↑	LPIPS↓
LongVolCap(4DGS) [45]	opt	8	24.21	0.840	0.221
GauHuman [12]	prior, opt	8	17.60	0.714	0.270
Diffuman4D [16]	prior, opt	8	26.32	0.881	0.150
LHM [30]	prior, feed-forward	1	19.73	0.751	0.242
LGM [34]	feed-forward	4	17.18	0.698	0.311
GPS-GS [55]	feed-forward	8	20.63	0.779	0.225
DepthSplat [43]	feed-forward	8	19.82	0.704	0.328
Ours	feed-forward	8	**26.54**	**0.882**	**0.123**

Table 4: Quantitative comparison with different numbers of input cameras. We evaluate the zero-shot generalization of our method to different numbers of input views. Our method remains robust when the number of input views changes.

	4 cameras			16 cameras		
Method	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
GPS-GS [55]	13.71	0.535	0.380	24.75	0.845	0.129
DepthSplat [43]	19.55	0.715	0.197	25.35	0.852	0.117
LVSM [15]	20.58	0.744	0.195	24.05	0.820	0.120
Ours	**23.16**	**0.811**	**0.112**	**27.31**	**0.900**	**0.072**

Each variant is obtained by selectively removing or altering a specific module to assess its individual effect. As illustrated in Fig. 6, removing the point-encoding module leads to noticeable artifacts and inconsistencies in the rendered views, as the model struggles to learn a unified 3D representation from multiple views. Omitting the ray-casting mechanism results in degradation of high-frequency details and introduces internal redundancy. Finally, as shown in Table 6, the proposed Point-Image Transformer attains better quality with full self-attention transformers [15, 51], yet with markedly lower computational overhead.

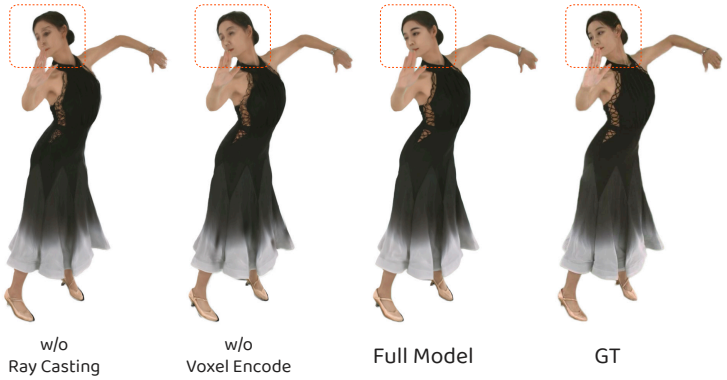

Fig. 6: Ablation study of modules. Removing our proposed components leads to visible artifacts in the rendered results, especially in challenging regions such as the face.

Table 5: Quantitative ablation study of different modules. We evaluate the impact of different modules on performance.

Model	PSNR↑	SSIM↑	LPIPS↓	GS-Num	Time
Full Model	**27.18**	**0.891**	**0.071**	**71k**	**0.4**
w/o Voxel Encode	26.60	0.876	0.091	160k	0.7
w/o Ray Casting	26.32	0.856	0.093	224k	0.7

Geometry Proxy Type We investigate the impact of different geometry proxy types on the performance of our method. The results are summarized in Table 7, where we compare the performance using various geometry proxy types. VGGT Depth indicates using depth maps predicted by a pre-trained VGGT [36] model as the geometry proxy. Bounding Box indicates using the foreground bounding box as the geometry proxy. Frustum Sample indicates sampling 3D points along camera rays within a predefined frustum as the geometry proxy. Our findings indicate that the consistent geometry proxy (e.g., Visual Hull) enhances the rendering quality and robustness.

Voxel Size We explore the influence of voxel size on the performance of our method. The results are presented in Table 8, where we evaluate different voxel sizes and their corresponding effects on rendering quality. In our experiments, we choose a voxel size of 0.0050, which achieves the best trade-off between performance and efficiency.

Table 6: Quantitative ablation study of different network architectures. We evaluate the impact of model architectures on performance. "AA" indicates the designed Alternating Attention.

Model	PSNR↑	SSIM↑	LPIPS↓	Time
AA(Ours)	**27.18**	**0.891**	**0.071**	**0.4**
Full	27.09	0.887	0.076	1.1

Table 7: Quantitative ablation study of geometry proxies. We evaluate the impact of different geometry proxies on performance.

Model	PSNR↑	SSIM↑	LPIPS↓
Visual Hull(Ours)	**27.18**	**0.891**	**0.071**
VGGT Depth	25.95	0.871	0.077
Bounding Box	23.89	0.775	0.153
Frustum Sample	22.92	0.749	0.184

Table 8: Quantitative ablation study on voxel size. We evaluate how voxel size affects quality–efficiency trade-offs.

Voxel Size	PSNR↑	SSIM↑	LPIPS↓	GS-Num	Time
0.0025	27.18	0.894	0.074	238k	1.4
0.0030	27.52	0.897	0.072	189k	0.9
0.0050	27.18	0.891	0.071	71k	0.4
0.0100	25.51	0.853	0.102	30k	0.3

5 Discussion

Conclusion. We presented PointSplat, an object-centric, feed-forward framework that reconstructs compact 3D Gaussian representations directly from sparse RGB inputs. By establishing appearance and geometry correspondences and introducing a Point-Image Transformer with modality-aware encodings, our method jointly reasons about geometry and appearance in 3D space, avoiding the inter-view redundancy of view-centric pipelines. This design scales efficiently with the number of views and image resolution, and supports real-time rendering from arbitrary viewpoints with lightweight user-side computation.

Limitations. Although our feed-forward method demonstrates significant advantages in efficiency and quality for novel view synthesis, it may struggle with unbounded scenes due to memory constraints. Additionally, extending it to construct temporally consistent and compact 4D representations remains an open challenge. We believe that more memory-efficient architectures and 3D/4D representations could help address these limitations in future work.

Acknowledgements

This work was partially supported by National Key R&D Program of China (No. 2024YFB2809105), Zhejiang Provincial Natural Science Foundation of China (No. LR25F020003), and Information Technology Center and State Key Lab of CAD&CG, Zhejiang University.

References

1. Ba, J.L., Kiros, J.R., Hinton, G.E.: Layer normalization (2016), arXiv:1607.06450
2. Charatan, D., Li, S., Tagliasacchi, A., Sitzmann, V.: 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 (CVPR) (2024)
3. Chen, Y., Wu, Q., Li, M., Lin, W., Harandi, M., Cai, J.: Fast feedforward 3d gaussian splatting compression. In: Proceedings of the The International Conference on Learning Representations (ICLR) (2025)
4. Chen, Y., Xu, H., Zheng, C., Zhuang, B., Pollefeys, M., Geiger, A., Cham, T.J., Cai, J.: Mvsplat: Efficient 3d gaussian splatting from sparse multi-view images. arXiv preprint arXiv:2403.14627 (2024)
5. Cheng, W., Chen, R., Yin, W., Fan, S., Chen, K., He, H., Luo, H., Cai, Z., Wang, J., Gao, Y., Yu, Z., Lin, Z., Ren, D., Yang, L., Liu, Z., Loy, C.C., Qian, C., Wu, W., Lin, D., Dai, B., Lin, K.Y.: Dna-rendering: A diverse neural actor repository for high-fidelity human-centric rendering. arXiv preprint **arXiv:2307.10173** (2023)
6. Dao, T.: FlashAttention-2: Faster attention with better parallelism and work partitioning. In: Proceedings of the The International Conference on Learning Representations (ICLR) (2024)
7. Fridovich-Keil, S., Yu, A., Tancik, M., Chen, Q., Recht, B., Kanazawa, A.: Plenoxels: Radiance fields without neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2022)
8. Gao, R., Holynski, A., Henzler, P., Brussee, A., Martin-Brualla, R., Srinivasan, P.P., Barron, J.T., Poole, B.: Cat3d: Create anything in 3d with multi-view diffusion models. Proceedings of the NeurIPS (NeurIPS) (2024)
9. Gortler, S.J., Grzeszczuk, R., Szeliski, R., Cohen, M.F.: The lumigraph. In: Proceedings of the Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH). pp. 43–54. SIGGRAPH '96 (1996). https://doi.org/10.1145/237170.237200
10. Henry, A., Dachapally, P.R., Pawar, S., Chen, Y.: Query-key normalization for transformers (2020), arXiv:2010.04245
11. Hong, Y., Zhang, K., Gu, J., Bi, S., Zhou, Y., Liu, D., Liu, F., Sunkavalli, K., Bui, T., Tan, H.: Lrm: Large reconstruction model for single image to 3d. arXiv preprint arXiv:2311.04400 (2023)
12. Hu, S., Liu, Z.: Gauhuman: Articulated gaussian splatting for real-time 3d human rendering. arXiv preprint (2023)
13. Işık, M., Rünz, M., Georgopoulos, M., Khakhulin, T., Starck, J., Agapito, L., Nießner, M.: Humanrf: High-fidelity neural radiance fields for humans in motion. ACM Transactions on Graphics (TOG) **42**(4), 1–12 (2023). https://doi.org/10.1145/3592415

14. Jiang, L., Mao, Y., Xu, L., Lu, T., Ren, K., Jin, Y., Xu, X., Yu, M., Pang, J., Zhao, F., et al.: Anysplat: Feed-forward 3d gaussian splatting from unconstrained views. arXiv preprint arXiv:2505.23716 (2025)
15. Jin, H., Jiang, H., Tan, H., Zhang, K., Bi, S., Zhang, T., Luan, F., Snavely, N., Xu, Z.: Lvsm: A large view synthesis model with minimal 3d inductive bias. In: Proceedings of the The International Conference on Learning Representations (ICLR) (2025)
16. Jin, Y., Peng, S., Wang, X., Xie, T., Xu, Z., Yang, Y., Shen, Y., Bao, H., Zhou, X.: Diffuman4d: 4d consistent human view synthesis from sparse-view videos with spatio-temporal diffusion models. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) (2025)
17. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. ACM Transactions on Graphics (TOG) **42**(4) (2023)
18. Kutulakos, K., Seitz, S.: A theory of shape by space carving. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV). vol. 1, pp. 307–314 vol.1 (1999). https://doi.org/10.1109/ICCV.1999.791235
19. Lee, C.C., Tabatabai, A., Tashiro, K.: Free viewpoint video (fvv) survey and future research direction. APSIPA Transactions on Signal and Information Processing **4**, e15 (2015). https://doi.org/10.1017/ATSIP.2015.18
20. Levoy, M., Hanrahan, P.: Light field rendering. In: Proceedings of the Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH). pp. 31–42. SIGGRAPH '96 (1996). https://doi.org/10.1145/237170.237199
21. Li, J., Tan, H., Zhang, K., Xu, Z., Luan, F., Xu, Y., Hong, Y., Sunkavalli, K., Shakhnarovich, G., Bi, S.: Instant3d: Fast text-to-3d with sparse-view generation and large reconstruction model. arXiv preprint arXiv:2311.06214 (2023)
22. Lin, H., Chen, S., Liew, J.H., Chen, D.Y., Li, Z., Shi, G., Feng, J., Kang, B.: Depth anything 3: recovering the visual space from any views. arXiv preprint arXiv:2511.10647 (2025)
23. Lin, H., Peng, S., Xu, Z., Yan, Y., Shuai, Q., Bao, H., Zhou, X.: Efficient neural radiance fields for interactive free-viewpoint video. In: Proceedings of the ACM SIGGRAPH Conference and Exhibition on Computer Graphics and Interactive Techniques in Asia (SIGGRAPH Asia) (2022)
24. Lu, T., Yu, M., Xu, L., Xiangli, Y., Wang, L., Lin, D., Dai, B.: Scaffold-gs: Structured 3d gaussians for view-adaptive rendering. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 20654–20664 (2024)
25. Mildenhall, B., Srinivasan, P.P., Ortiz-Cayon, R., Kalantari, N.K., Ramamoorthi, R., Ng, R., Kar, A.: Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. ACM Transactions on Graphics (TOG) **38**(4), 1–14 (2019)
26. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: representing scenes as neural radiance fields for view synthesis. Commun. ACM **65**(1), 99–106 (2021). https://doi.org/10.1145/3503250
27. Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. ACM Transactions on Graphics (TOG) **41**(4) (2022). https://doi.org/10.1145/3528223.3530127
28. Ng, R., Levoy, M., Brédif, M., Duval, G., Horowitz, M., Hanrahan, P.: Light field photography with a hand-held plenoptic camera. Ph.D. thesis, Stanford university (2005)

29. Peng, S., Zhang, Y., Xu, Y., Wang, Q., Shuai, Q., Bao, H., Zhou, X.: Neural body: Implicit neural representations with structured latent codes for novel view synthesis of dynamic humans. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2021)
30. Qiu, L., Gu, X., Li, P., Zuo, Q., Shen, W., Zhang, J., Qiu, K., Yuan, W., Chen, G., Dong, Z., Bo, L.: Lhm: Large animatable human reconstruction model from a single image in seconds. arXiv preprint arXiv:2503.10625 (2025)
31. Ren, K., Jiang, L., Lu, T., Yu, M., Xu, L., Ni, Z., Dai, B.: Octree-gs: Towards consistent real-time rendering with lod-structured 3d gaussians. arXiv preprint arXiv:2403.17898 (2024)
32. Renderpeople: Renderpeople. `https://renderpeople.com/3d-people` (2018)
33. Szymanowicz, S., Zhang, J.Y., Srinivasan, P., Gao, R., Brussee, A., Holynski, A., Martin-Brualla, R., Barron, J.T., Henzler, P.: Bolt3D: Generating 3D Scenes in Seconds. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) (2025)
34. Tang, J., Chen, Z., Chen, X., Wang, T., Zeng, G., Liu, Z.: Lgm: Large multi-view gaussian model for high-resolution 3d content creation. arXiv preprint arXiv:2402.05054 (2024)
35. Team, T.H.: Hunyuan3d 2.0: Scaling diffusion models for high resolution textured 3d assets generation (2025), arXiv:2501.12202
36. Wang, J., Chen, M., Karaev, N., Vedaldi, A., Rupprecht, C., Novotny, D.: Vggt: Visual geometry grounded transformer. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2025)
37. Wang, Q., Wang, Z., Genova, K., Srinivasan, P., Zhou, H., Barron, J.T., Martin-Brualla, R., Snavely, N., Funkhouser, T.: Ibrnet: Learning multi-view image-based rendering. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2021)
38. Wang, S., Leroy, V., Cabon, Y., Chidlovskii, B., Revaud, J.: Dust3r: Geometric 3d vision made easy (2023), arXiv:2312.14132
39. Wang, Y., Zhou, J., Zhu, H., Chang, W., Zhou, Y., Li, Z., Chen, J., Pang, J., Shen, C., He, T.: π^3: Scalable permutation-equivariant visual geometry learning (2025), arXiv:2507.13347
40. Wang, Z., Bovik, A., Sheikh, H., Simoncelli, E.: Image quality assessment: from error visibility to structural similarity. IEEE Transactions on Image Processing (TIP) **13**(4), 600–612 (2004). `https://doi.org/10.1109/TIP.2003.819861`
41. Wu, S., Lin, Y., Zhang, F., Zeng, Y., Xu, J., Torr, P., Cao, X., Yao, Y.: Direct3d: Scalable image-to-3d generation via 3d latent diffusion transformer. arXiv preprint arXiv:2405.14832 (2024)
42. Xiao, J., Zhang, Q., Nie, Y., Zhu, L., Zheng, W.S.: RoGSplat: Learning robust generalizable human gaussian splatting from sparse multi-view images. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2025)
43. Xu, H., Peng, S., Wang, F., Blum, H., Barath, D., Geiger, A., Pollefeys, M.: Depthsplat: Connecting gaussian splatting and depth. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2025)
44. Xu, Q., Xu, Z., Philip, J., Bi, S., Shu, Z., Sunkavalli, K., Neumann, U.: Point-nerf: Point-based neural radiance fields. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5438–5448 (2022)
45. Xu, Z., Xu, Y., Yu, Z., Peng, S., Sun, J., Bao, H., Zhou, X.: Representing long volumetric video with temporal gaussian hierarchy. ACM Transactions on Graphics (TOG) **43**(6) (2024)

46. Yang, Z., Yang, H., Pan, Z., Zhang, L.: Real-time photorealistic dynamic scene representation and rendering with 4d gaussian splatting. In: Proceedings of the The International Conference on Learning Representations (ICLR) (2024)
47. Yao, Y., Luo, Z., Li, S., Fang, T., Quan, L.: Mvsnet: Depth inference for unstructured multi-view stereo. In: Proceedings of the European Conference on Computer Vision (ECCV) (2018)
48. Yinghao, X., Zifan, S., Wang, Y., Hansheng, C., Ceyuan, Y., Sida, P., Yujun, S., Gordon, W.: Grm: Large gaussian reconstruction model for efficient 3d reconstruction and generation (2024), arXiv:2403.14621
49. Yu, T., Zheng, Z., Guo, K., Liu, P., Dai, Q., Liu, Y.: Function4d: Real-time human volumetric capture from very sparse consumer rgbd sensors. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2021)
50. Zhang, B., Tang, J., Nießner, M., Wonka, P.: 3dshape2vecset: A 3d shape representation for neural fields and generative diffusion models. ACM Transactions on Graphics (TOG) **42**(4) (2023). https://doi.org/10.1145/3592442
51. Zhang, K., Bi, S., Tan, H., Xiangli, Y., Zhao, N., Sunkavalli, K., Xu, Z.: Gs-lrm: Large reconstruction model for 3d gaussian splatting. In: Proceedings of the European Conference on Computer Vision (ECCV) (2024)
52. Zhang, L., Wang, Z., Zhang, Q., Qiu, Q., Pang, A., Jiang, H., Yang, W., Xu, L., Yu, J.: Clay: A controllable large-scale generative model for creating high-quality 3d assets. ACM Transactions on Graphics (TOG) **43**(4) (2024). https://doi.org/10.1145/3658146
53. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2018)
54. Zhang, S., Fei, X., Liu, F., Song, H., Duan, Y.: Gaussian graph network: Learning efficient and generalizable gaussian representations from multi-view images. Proceedings of the NeurIPS (NeurIPS) (2024)
55. Zheng, S., Zhou, B., Shao, R., Liu, B., Zhang, S., Nie, L., Liu, Y.: Gps-gaussian: Generalizable pixel-wise 3d gaussian splatting for real-time human novel view synthesis. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2024)
56. Zheng, X., Liao, L., Li, X., Jiao, J., Wang, R., Gao, F., Wang, S., Wang, R.: Pku-dymvhumans: A multi-view video benchmark for high-fidelity dynamic human modeling. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2024)
57. Ziwen, C., Tan, H., Zhang, K., Bi, S., Luan, F., Hong, Y., Fuxin, L., Xu, Z.: Long-lrm: Long-sequence large reconstruction model for wide-coverage gaussian splats. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) (2025)