PCASPLAT: GAUSSIAN SPLATTING WITH LOCAL PCA REGULARIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Gaussian splatting has emerged as a flexible representation for 3D reconstruction from posed images. However, existing methods are optimized primarily using rasterization-based losses, which supervise a splat only when it contributes to sampled camera rays. Gaussians that are occluded or contribute little to the sampled view therefore receive weak or no geometric gradients and may drift away from the underlying surface, producing undesired floaters. We introduce PCAsplat, a geometry-aware regularization framework for Gaussian splatting based on differentiable local principal component analysis (PCA). Our PCA regularizer acts directly on neighborhoods of Gaussian centers and can therefore update Gaussians that do not contribute to the current training view. We regularize the PCA eigenvalues to encourage Gaussians to move to the underlying surface with isotropic tangent-plane coverage. We also align each Gaussian normal with the PCA-estimated neighborhood normal to enforce consistent orientation. Experiments on DTU, Tanks and Temples, and NeRF Synthetic show that the splats produced by PCAsplat better approximate samples of the reference surface while substantially reducing undesired floaters. These surface-aligned splats enable downstream geometry-processing tasks, including point cloud segmentation, and direct Poisson reconstruction. Additionally, PCAsplat remains competitive under conventional novel view synthesis and mesh extraction tasks. Code will be released.

1 INTRODUCTION

Gaussian splatting (Kerbl et al., 2023) is becoming the standard framework for novel view synthesis (NVS) and 3D reconstruction from images, offering a favorable trade-off between rendering quality, optimization speed, and interactive visualization. It represents a scene as a collection of anisotropic Gaussians with appearance attributes optimized via differentiable rendering. While Gaussian splatting is highly effective for NVS, Gaussians can be located far from the underlying surfaces, forming *floaters* both outside and inside objects (see Fig. 1, top-left). These floaters remain a challenge for using the learned representation directly in geometry-processing tasks such as segmentation and Poisson surface reconstruction.

Methods such as 2DGS (Huang et al., 2024), GOF (Yu et al., 2024b), PGSR (Chen et al., 2025), and RaDe-GS (Zhang et al., 2026) use losses based on rendered depth and normals to improve geometric reconstruction. However, these rendering-based losses supervise Gaussians through their contributions to sampled views. Gaussians that are occluded or contribute little to the view therefore receive weak or no geometric gradients from these losses and may drift away from the underlying surface, producing exterior and interior floaters. The learned representation may also contain oversized, diffuse, or poorly localized Gaussians. Although floaters may have little impact on training views, exterior ones can appear in novel views while interior ones may remain occluded, with both limiting the representation's use in downstream applications. Other approaches couple Gaussians to an auxiliary surface representation. GS-Pull (Zhang et al., 2024) pulls Gaussians onto the zero-level set of a learned neural signed distance function (SDF), while MILo (Guédon et al., 2025) differentiably extracts a mesh from Gaussian parameters during training and enforces consistency between the mesh and the Gaussians. However, these approaches rely on an auxiliary representation that must also be optimized, and undesired floaters remain an issue.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

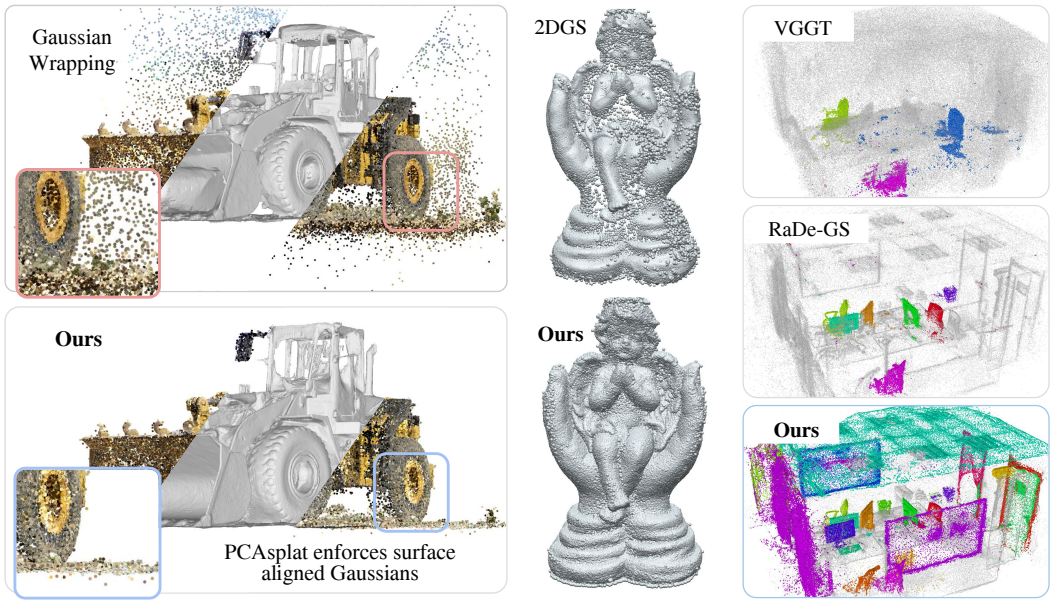

Figure 1: We present **PCAsplat**, a geometric regularization of Gaussian splatting that aligns Gaussians with surfaces through differentiable local PCA. Compared with current methods, PCAsplat substantially reduces floaters (left) and produces cleaner point clouds with better surface coverage (center). This improved point-cloud enables downstream tasks such as instance segmentation (right).

Overall, learning surface-aligned Gaussians suitable for downstream geometry processing remains a challenge. To address this problem, we introduce PCAsplat, a geometry-aware regularization framework for Gaussian splatting based on differentiable local principal component analysis (PCA). Our key idea is to regularize Gaussian neighborhoods directly in scene space through PCA statistics. This provides geometric supervision even to Gaussians that do not contribute to the current training view. For each Gaussian, we estimate a local 3D neighborhood and compute its covariance eigenspectrum to characterize local thickness and tangent-plane anisotropy. We use these statistics to encourage Gaussian centers to move toward the surface by reducing local thickness, while promoting isotropic tangent-plane coverage and aligning Gaussian normals with PCA-derived normals up to sign.

Experiments on DTU, Tanks and Temples, and NeRF Synthetic show that PCAsplat achieves strong surface reconstruction results compared to current methods. It produces surface-aligned splats with fewer exterior and interior floaters (Fig. 1, bottom-left and center), allowing for more accurate downstream geometry processing. PCAsplat yields substantially more accurate instance segmentation and more coherent object masks than current methods using the same segmentation pipeline (Fig. 1, right). PCAsplat also remains competitive under conventional novel view synthesis and TSDF-based mesh extraction.

In summary, our **main contributions** are:

- A differentiable local PCA framework for regularizing Gaussian neighborhoods directly in scene space, including Gaussians that do not contribute to the current training view.
- Geometric losses for learning surface-aligned Gaussians by moving them toward the surface, promoting isotropic coverage, and aligning their normals with PCA-normals.
- An evaluation of splats as surface samples, showing improved geometric accuracy and their use in downstream applications, including segmentation and Poisson reconstruction.

2 Related Work

Surface reconstruction from images is a long-standing problem in vision. Classical methods such as COLMAP (Schonberger & Frahm, 2016) combine structure from motion and multi-view stereo to recover camera poses and point clouds. Recently, feed-forward models such as VGGT (Wang et al., 2025) and VGGT-Ω (Wang et al., 2026) predict cameras, depth maps, and point clouds di-

rectly from unposed images. However, these methods lack a view-dependent appearance model for novel-view synthesis. In contrast, neural implicit methods jointly learn geometry and view-dependent appearance: VolSDF (Yariv et al., 2021), NeuS (Wang et al., 2021), and Neuralangelo (Li et al., 2023) optimize neural SDFs for each scene through differentiable rendering. However, repeated neural-field evaluations along each ray make volume rendering computationally expensive. Gaussian splatting (Kerbl et al., 2023) enabled fast rasterization of explicit primitives but lacked geometry constraints. 2DGS (Huang et al., 2024), Gaussian Surfels (Dai et al., 2024), and Quadratic Gaussian Splatting (Zhang et al., 2025) made splats surface-like through modified primitives or rendering formulations together with rendering losses to regularize geometric properties of the scene. SuGaR (Guédon & Lepetit, 2024), GOF (Yu et al., 2024b), and RaDe-GS (Zhang et al., 2026) improve mesh extraction via surface alignment, opacity fields, or better depth rasterization. And, PGSR (Chen et al., 2025) introduced multi-view consistency losses to further improve geometry. For mesh extraction, methods such as 2DGS, RaDe-GS, and PGSR fuse rendered depth maps using TSDF (Curless & Levoy, 1996), which are based on ray-gaussian contributions. Consequently, accurate meshes do not necessarily imply surface-aligned Gaussian centers.

Rendering-based geometric regularization augments photometric supervision with losses on rendered depth and normals, aiming to reduce undesired **floaters**. Mip-NeRF 360 (Barron et al., 2022) introduces a distortion loss that encourages rendering contributions to concentrate within a narrow interval along each ray, discouraging diffuse density and floaters. 2DGS and GOF adapt this loss to encourage surface alignment by reducing depth differences between contributing Gaussians along viewing rays. These differences are weighted by rendering contributions, which depend on opacity and accumulated transmittance. As such, Gaussians that are occluded or contribute little to a view may receive weak or no geometric gradients from these losses, allowing exterior and interior floaters to persist. PCAsplat complements this supervision with local PCA losses that act directly on 3D Gaussian neighborhoods, so Gaussians that do not contribute to the current training view can still receive geometric gradients.

Hybrid Gaussian–surface representations couple Gaussians with auxiliary surface representations. GS-Pull (Zhang et al., 2024) pulls Gaussian centers onto the zero-level set of a neural SDF, at the additional computational cost of jointly optimizing this field. Its Gaussian-disk supervision addresses sparse, nonuniform support but remains sensitive to extreme sparsity. GSDF (Yu et al., 2024a) couples Gaussians and SDFs through depth and normal consistency, using Gaussian-rendered depth to guide SDF sampling and SDF values to control Gaussian growth and pruning. Gaussian-UDF (Li et al., 2025) jointly learns 2D Gaussians and an unsigned distance field for open surfaces, using Gaussian centers and local planes to supervise distances. MILo (Guédon et al., 2025) and GaussianWrapping (GW) (Gomez et al., 2026) differentiably extracts a mesh from Gaussian-derived pivots and learned signed distances, enforcing Gaussian–mesh rendering consistency.

In contrast, we adapt local PCA, a classical method for estimating tangent planes and normals from point clouds (Hoppe et al., 1992; Pauly et al., 2002), to directly regularize Gaussian neighborhoods. We add losses for neighborhood thickness, tangent-plane coverage, and normal alignment to existing splatting objectives without learning auxiliary SDFs or extracting meshes during training.

3 METHOD

Our goal is to learn surface-aligned Gaussians by regularizing their local geometry directly in scene space, reducing floaters and supporting downstream geometry processing. We formulate local PCA-based regularization as differentiable losses on Gaussian geometry, optimized jointly with the rendering objective. For each Gaussian, we compute the covariance eigenspectrum and principal directions of a local neighborhood of Gaussian centers. These quantities define losses that reduce neighborhood thickness, enforce isotropic tangent-plane coverage, and align Gaussian normals with PCA-derived normals. Fig. 2 gives an overview of our framework.

3.1 PROBLEM SETTING

Given a set of posed images $\{I_i\}_{i=1}^{M}$, our goal is to optimize a Gaussian splatting point cloud to be geometrically aligned with the underlying scene surface. Recall that Gaussian splatting represents a scene as a set of anisotropic Gaussians $\mathcal{G} = \{g_k\}_{k=1}^{N}$, $g_k = (\mu_k, \Sigma_k, \alpha_k, c_k)$, where $\mu_k \in \mathbb{R}^3$ de-

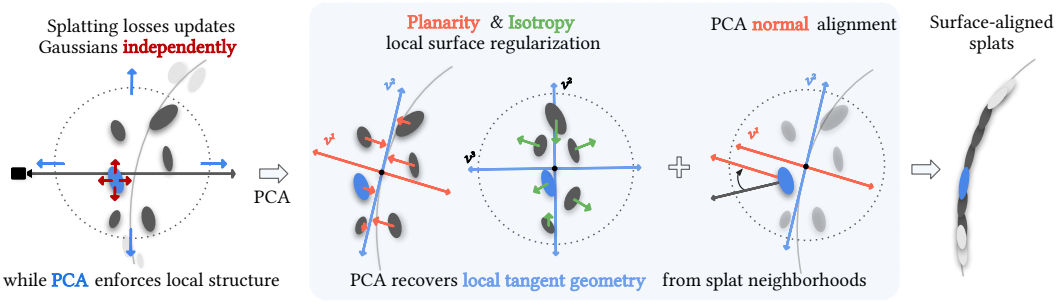

Figure 2: **Overview of PCAsplat.** We regularize Gaussian neighborhoods directly in scene space. For each Gaussian, we estimate a local neighborhood and compute differentiable PCA statistics that capture the underlying tangent structure. We then regularize the local eigenspectrum to reduce neighborhood thickness and promote isotropic tangent-plane coverage. We also align Gaussian normals with PCA-derived normals, encouraging surface-aligned and well-localized splats. Our formulation is lightweight, differentiable, and directly compatible with existing splatting pipelines.

notes the Gaussian center, $\Sigma_k \in \mathbb{R}^{3\times3}$ its positive-definite covariance matrix, $\alpha_k \in [0,1]$ its opacity, and c_k its view-dependent RGB color, represented using spherical harmonics. Each Gaussian defines kernels $g_k(x) = \exp\!\left(-\frac{1}{2}\left\langle x-\mu_k, \Sigma_k^{-1}(x-\mu_k)\right\rangle\right)$. which are projected onto the image plane and rasterized through alpha compositing. For a view ray r, with Gaussians indexed in front-to-back order, the rendered color is given by

$$\hat{I}(r) = \sum_{k=1}^{N} T_k(r)\,\alpha_k(r)\,c_k + T_{N+1}(r)\,c_{\mathrm{bg}}, \qquad T_k(r) = \prod_{j=1}^{k-1}\big(1-\alpha_j(r)\big),$$

where $\alpha_k(r)=\alpha_k g_k^{2\mathrm{D}}(r)$ is the effective opacity, with $g_k^{2\mathrm{D}}(r)$ denoting the projected Gaussian evaluated at the pixel corresponding to r. Here, $T_k(r)$ denotes the transmittance, c_k is evaluated for the current view, and c_{bg} is the background color. $\mathcal{G}$ is optimized using the standard photometric loss

$$\mathcal{L}_{\mathrm{RGB}} = (1-\lambda)\,\|I-\hat{I}\|_1 + \lambda\,\mathcal{L}_{\mathrm{SSIM}}(I,\hat{I}), \tag{1}$$

where I and $\hat{I}$ denote the ground-truth and rendered images, $\lambda \in [0,1]$ balances the two terms, and $\mathcal{L}_{\mathrm{SSIM}}(I,\hat{I}) = 1-\mathrm{SSIM}(I,\hat{I})$ denotes the structural dissimilarity loss. The ℓ_1 term is averaged over pixels and RGB channels. While Gaussian splatting is effective for appearance reconstruction, this photometric objective does not explicitly regularize the local geometry of Gaussian neighborhoods.

We therefore use differentiable local PCA (Section 3.2) to regularize Gaussian neighborhoods directly in scene space. Our losses reduce neighborhood thickness, enforce isotropic tangent-plane coverage, and align Gaussian normals with PCA-derived normals (Section 3.3).

3.2 Local PCA neighborhoods

For each Gaussian g_i, we define its local neighborhood $\mathcal{N}_K(i) = \mathrm{KNN}(\mu_i, K)$ as the indices of the K nearest Gaussian centers to $\mu_i \in \mathbb{R}^3$. As in classical point-cloud normal estimation (Hoppe et al., 1992; Pauly et al., 2002), we compute the covariance matrix of the neighboring centers:

$$\mathbf{C}_i = \frac{1}{K}\sum_{j\in\mathcal{N}_K(i)} (\mu_j-\bar{\mu}_i)(\mu_j-\bar{\mu}_i)^\top, \qquad \bar{\mu}_i = \frac{1}{K}\sum_{j\in\mathcal{N}_K(i)} \mu_j. \tag{2}$$

Here, $\bar{\mu}_i$ is the neighborhood centroid. The eigendecomposition of $\mathbf{C}_i$, $\mathbf{C}_i\mathbf{v}_i^{(m)} = \lambda_i^{(m)}\mathbf{v}_i^{(m)}$, with orthonormal eigenvectors and $0 \le \lambda_i^{(1)} \le \lambda_i^{(2)} \le \lambda_i^{(3)}$, summarizes the local geometric structure. The eigenvector $\mathbf{v}_i^{(1)}$ associated with the smallest eigenvalue estimates the local surface *normal*, while $\mathbf{v}_i^{(2)}$ and $\mathbf{v}_i^{(3)}$ span the estimated *tangent plane*. The plane through $\bar{\mu}_i$ with these tangent directions locally approximates the underlying surface. Since $\lambda_i^{(1)}$ measures the mean squared distance of neighboring Gaussian centers to this plane, we minimize it to pull the centers toward the plane during optimization. Optimizing this objective jointly with the rendering loss requires computing its gradient with respect to the Gaussian centers.

Assuming fixed neighborhood indices and no repeated eigenvalues, the partial derivatives needed for backpropagation are (Greenbaum et al., 2020)

$$\frac{\partial \lambda_i^{(m)}}{\partial \mu_j} = \frac{2}{K} \left\langle \mathbf{v}_i^{(m)}, \mu_j - \bar{\mu}_i \right\rangle \mathbf{v}_i^{(m)}, \qquad j \in \mathcal{N}_K(i).$$

The gradient is proportional to the projection of $\mu_j - \bar{\mu}_i$ onto $\mathbf{v}_i^{(m)}$; for $m = 1$, its negative points toward the estimated tangent plane. We use these derivatives to optimize the eigenspectrum loss introduced next directly with respect to the Gaussian centers.

3.3 PCA-BASED GEOMETRIC LOSSES

PCA eigenspectrum loss. We use local PCA to move the Gaussian centers $\{\mu_i\}_{i=1}^N$ toward the underlying surface. In other words, surface-like Gaussian neighborhoods should be thin along the PCA normal direction and well distributed within the tangent plane of the underlying surface, i.e., $\lambda_i^{(1)} \ll \lambda_i^{(2)} \approx \lambda_i^{(3)}$. We therefore regularize the local PCA eigenspectrum using

$$\mathcal{L}_{\text{PCA}} = \frac{1}{N} \sum_{i=1}^{N} \left[\lambda_i^{(1)} + \gamma \left| \lambda_i^{(3)} - \lambda_i^{(2)} \right| \right]. \tag{3}$$

The first term penalizes variance along the estimated normal, while the second promotes isotropic tangent-plane coverage. We balance these terms with $\gamma = 0.001$.

This loss acts directly on the spatial distribution of Gaussian centers, rather than on rendered depth or normal maps. Minimizing $\lambda_i^{(1)}$ pulls neighboring Gaussian centers toward the estimated tangent plane through $\bar{\mu}_i$, discouraging floaters. The tangent isotropy term discourages elongated neighborhoods by encouraging balanced spread along the two tangent directions, as shown in Fig. 3. Together, these terms encourage thin neighborhoods with balanced tangent-plane support, improving the samples used for downstream mesh extraction and point-cloud processing.

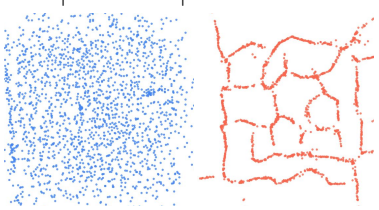

Figure 3: Balanced coverage with isotropy (left); elongated structures without it (right).

PCA normal alignment. To encourage Gaussians to align with the local geometry, we define a normal consistency loss between each Gaussian normal and the local PCA normal. Specifically, each Gaussian g_i has a rotation matrix $\mathbf{R}_i \in \text{SO}(3)$, with $\mathbf{R}_i = (\mathbf{r}_{i,1}, \mathbf{r}_{i,2}, \mathbf{r}_{i,3})$, and a scaling vector $\mathbf{s}_i = (s_{i,1}, s_{i,2}, s_{i,3}) \in \mathbb{R}^3_{>0}$. These parameters define its covariance as $\Sigma_i = \mathbf{R}_i \, \text{diag}(s_{i,1}^2, s_{i,2}^2, s_{i,3}^2) \mathbf{R}_i^\top$. We define the Gaussian normal $\mathbf{n}_i^g$ as the axis associated with the smallest scale:

$$\mathbf{n}_i^g = \mathbf{r}_{i,k^*}, \qquad k^* = \arg\min_{1 \le k \le 3} s_{i,k}.$$

We combine sign-invariant alignment between the PCA normal $\mathbf{v}_i^{(1)}$ and the Gaussian normal $\mathbf{n}_i^g$ with a scale penalty in the normal loss

$$\mathcal{L}_{\text{normal}} = w_{\text{normal}} \frac{1}{N} \sum_{i=1}^{N} \left(1 - \left| \left\langle \mathbf{v}_i^{(1)}, \mathbf{n}_i^g \right\rangle \right| \right) + w_{\text{surfel}} \frac{1}{N} \sum_{i=1}^{N} \min(s_{i,1}, s_{i,2}, s_{i,3}). \tag{4}$$

The second term encourages each Gaussian to behave as a 2D surfel by penalizing its smallest scale, with w_{normal} and w_{surfel} controlling the strength of this regularization.

Final loss function. Together, the PCA eigenspectrum and normal alignment terms encourage Gaussian centers and orientations to form coherent, disk-like local surface structures. Thus, our core training objective is

$$\mathcal{L} = \mathcal{L}_{\text{RGB}} + w_{\text{PCA}} \mathcal{L}_{\text{PCA}} + \mathcal{L}_{\text{normal}}, \tag{5}$$

where $\mathcal{L}_{\text{RGB}}$ is the photometric rendering loss in Eq. 1, $\mathcal{L}_{\text{PCA}}$ regularizes the local eigenspectrum with w_{PCA} being scheduled to start at 1 and grow to 10, and $\mathcal{L}_{\text{normal}}$ aligns Gaussian normals with PCA-derived normals. In our implementation, we also keep the standard PGSR multi-view loss $\mathcal{L}_{\text{pgsr}} = \mathcal{L}_{\text{mvgeo}} + \mathcal{L}_{\text{ncc}}$ from the base pipeline, where $\mathcal{L}_{\text{mvgeo}}$ enforces multi-view geometric consistency and $\mathcal{L}_{\text{ncc}}$ measures patch-wise normal consistency across views.

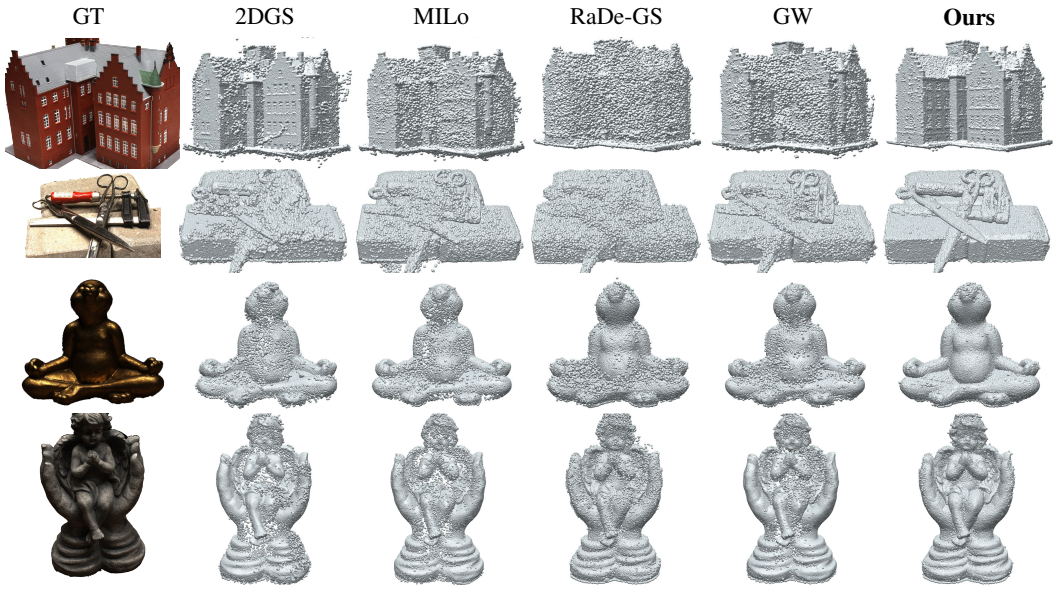

Figure 4: **Qualitative comparison on DTU.** PCAsplat produces surface-aligned Gaussian centers with substantially fewer floaters, sharper edges, and finer geometric detail than the baselines.

4 EXPERIMENTS

We evaluate PCAsplat on the surface alignment of its learned Gaussians and their performance in downstream geometry processing. We compare against Gaussian-based reconstruction methods on point-cloud accuracy, 3D instance segmentation (Schult et al., 2023), and Poisson surface reconstruction (Kazhdan & Hoppe, 2013).

Datasets and metrics. We evaluate surface accuracy on DTU (Jensen et al., 2014) and Tanks and Temples (Knapitsch et al., 2017) using standard Chamfer distance (CD) and F1-score. Both metrics compare reconstructed point clouds (Gaussian centers for splatting methods) with ground-truth point clouds. For 3D semantic instance segmentation, we use ScanNet++ (Yeshwanth et al., 2023) and report Top-1 and Top-3 mean intersection over union (mIoU). We also evaluate F1-score on a visible-surface benchmark constructed from the NeRF Synthetic (Mildenhall et al., 2021) scene assets. Implementation details are provided in Appendix A, and the visible-surface reference meshes we introduce are described in Appendix B.

Baselines. We compare with standard 3DGS and leading Gaussian splatting methods for surface reconstruction, including 2DGS (Huang et al., 2024), PGSR (Chen et al., 2025), GS-Pull (Zhang et al., 2024), MILo (Guédon et al., 2025), RaDe-GS (Zhang et al., 2026), and GaussianWrapping (GW) (Gomez et al., 2026). The latter use depth distortion, geometric consistency, or surface constraints to improve reconstruction. VGGT-Ω (Wang et al., 2026) and COLMAP-D (Schonberger & Frahm, 2016) provide additional point-cloud references; "-D" denotes dense output.

Method	24	37	40	55	63	65	69	83	97	105	106	110	114	118	122	Mean	Time
VGGT-Ω	0.931	1.217	1.502	1.302	1.995	1.468	1.162	5.626	3.363	4.084	2.000	1.263	1.219	2.131	1.754	2.068	63s
COLMAP-D	0.382	0.618	0.336	0.349	0.570	0.837	0.455	1.129	0.650	0.540	0.418	0.468	0.300	0.387	0.348	0.519	26.9m
3DGS	1.056	1.148	1.396	1.035	1.285	1.609	1.354	1.836	1.349	1.005	1.662	1.536	1.037	1.274	1.077	1.311	5.2m
2DGS	0.906	1.879	1.065	0.920	1.261	2.351	1.399	1.377	1.395	0.949	1.231	1.525	1.356	1.308	1.051	1.332	10.9m
GS-Pull	1.243	0.803	1.127	0.719	1.129	1.698	1.554	1.744	1.493	0.954	1.891	1.791	0.806	1.395	1.107	1.297	21.8m
PGSR	*0.621*	0.651	0.817	*0.611*	*0.804*	*0.780*	*0.729*	1.020	0.800	0.647	0.801	0.651	0.510	0.666	0.571	*0.712*	30.5m
MILo	0.758	0.897	0.827	0.732	0.869	1.019	0.924	*1.013*	0.943	*0.577*	0.816	0.886	0.788	0.701	0.569	0.821	25.0m
RaDe-GS	0.776	*0.764*	*0.790*	0.656	0.732	0.648	0.614	0.956	*0.857*	0.505	0.496	*0.696*	*0.558*	0.433	0.445	0.662	14.5m
GW	0.616	0.928	0.507	**0.457**	0.950	0.833	0.732	1.416	0.947	0.785	*0.730*	1.157	0.566	*0.656*	*0.558*	0.789	18.8m
Ours	**0.509**	**0.551**	**0.488**	0.464	**0.656**	**0.563**	**0.511**	**0.910**	**0.607**	**0.436**	**0.425**	**0.503**	**0.412**	**0.371**	**0.382**	**0.519**	22.3m

Table 1: **Quantitative comparison on DTU.** We report Chamfer distance (↓) between reconstructed and ground-truth point clouds. We highlight the **best**, second-best, and *third-best* Gaussian-based results, with PCAsplat achieving the best overall results.

4.1 COMPARISONS

Tab. 1 shows that PCAsplat outperforms all evaluated Gaussian-based methods in mean Chamfer distance on DTU. Our mean CD of 0.519 mm is 22% lower than that of the strongest Gaussian-based baseline, RaDe-GS (0.662 mm). While our method matches COLMAP-D's reported mean CD, we have a lower runtime (22.3 vs. 26.9 minutes) and can perform novel view synthesis. Fig. 4 shows that PCAsplat produces surface-aligned Gaussian centers with sharper building edges, clearer tool boundaries, and finer statue details. The cross-sections in Fig. 5 show thinner surface profiles and substantially fewer **exterior and interior floaters** than RaDe-GS and GW. In the head cross-section, our Gaussian centers concentrate around the surface, as expected, while both baselines leave many centers scattered inside the object.

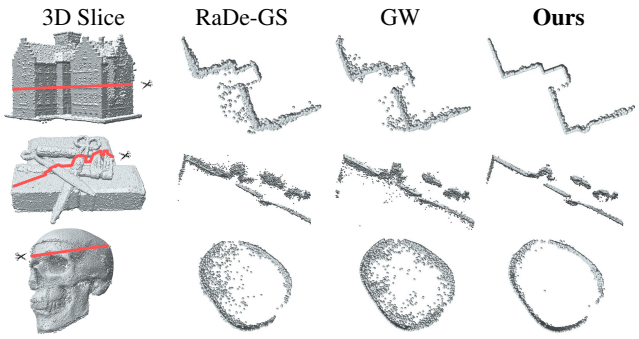

Figure 5: Qualitative comparison of cross-section slices on challenging DTU scenes.

To show that our local PCA regularization benefits other splatting methods, we apply it to three representative baselines (2DGS, PGSR, and MILo) without additional hyperparameter tuning. All three achieve **lower Chamfer distance** (Tab. 2), showing that our regularization improves Gaussian geometry as a **plug-and-play** addition to existing methods.

	Chamfer Dist. (↓)		
Method	**w/o** PCA	→	**w/** PCA
2DGS	1.332	→	**1.211**
PGSR	0.712	→	**0.667**
MILo	0.821	→	**0.754**
RaDe-GS	0.662	→	**0.519**

Table 2: Effect of local PCA regularization across splatting methods on DTU.

On Tanks and Temples, PCAsplat achieves the highest mean F1-score among the evaluated Gaussian-based methods (Tab. 3). The qualitative comparisons in Fig. 6 show fewer floaters and better surface coverage than the Gaussian-based baselines shown. On the new NeRF Synthetic visible mesh benchmark, PCAsplat also achieves the highest mean F1-score among the compared methods and the best or second-best score on every scene. By using reference meshes derived directly from the original synthetic scene assets, this benchmark evaluates alignment to known surfaces and complements DTU and Tanks and Temples, whose captured reference geometry can contain inaccuracies (Guédon et al., 2025; Gomez et al., 2026). Together, these results show that PCAsplat improves Gaussian-to-surface alignment on both captured and synthetic scenes.

Appendix C expands our evaluation. First, our surface-aligned splats are asset-ready, allowing object extraction through spatial cropping without manual floater removal. Second, aerial and below-ground views of Tanks and Temples show Gaussian centers better aligned with the ground and walls, with fewer floaters. Finally, for TSDF-based mesh extraction and novel view synthesis, PCAsplat achieves the lowest mean Chamfer distance on DTU while remaining competitive on Tanks and Temples.

	Tanks and Temples								**NeRF Synthetic Mesh**									
Method	Barn	Cater.	Court.	Igna.	Meet.	Truck	Mean	Time	Chair	Drums	Ficus	Hotdog	Lego	Mats.	Mic	Ship	Mean	Time
VGGT-Ω*	0.041	0.049	0.033	0.058	0.054	0.116	0.059	2.8	0.112	0.121	0.130	0.168	0.199	0.161	0.091	0.164	0.143	1.5
COLMAP-D	0.555	0.494	0.359	0.712	0.332	0.639	0.515	168.7	0.654	0.543	0.632	0.496	0.489	0.240	0.428	0.409	0.486	25.3
3DGS	0.076	0.080	0.047	0.128	0.056	0.133	0.087	7.9	0.336	0.392	0.543	0.113	0.284	0.161	0.303	0.215	0.293	4.8
2DGS	0.074	0.075	0.032	0.136	0.026	0.144	0.081	12.3	0.253	0.243	0.306	0.051	0.206	0.076	0.221	0.159	0.189	5.5
GSPull	0.071	0.057	0.074	0.105	0.012	0.106	0.071	113	0.119	0.103	0.160	0.009	0.036	0.028	0.077	0.006	0.067	26.4
PGSR	*0.165*	*0.145*	0.075	0.207	*0.111*	*0.246*	*0.158*	72.0	*0.673*	0.620	**0.874**	0.313	*0.462*	*0.338*	**0.594**	*0.431*	0.538	26.9
MILo	0.324	0.258	0.172	**0.553**	0.154	0.462	0.321	110	0.433	0.408	0.530	0.146	0.329	0.165	0.363	0.244	0.327	25.0
RaDe-GS	0.120	0.112	0.068	0.156	0.099	0.191	0.124	18.3	0.653	*0.605*	*0.752*	0.409	0.450	0.355	0.544	0.375	0.518	10.7
GW	0.153	0.134	*0.079*	*0.223*	0.088	0.234	0.152	27.0	**0.740**	0.595	0.695	*0.360*	**0.515**	0.284	*0.567*	0.440	*0.524*	13.8
Ours	**0.428**	**0.344**	**0.231**	0.509	**0.293**	**0.518**	**0.387**	53.0	0.727	**0.666**	0.811	**0.441**	0.506	**0.386**	0.593	**0.515**	**0.581**	25.0

* Maximum of 250 images per scene on Tanks and Temples.

Table 3: **Quantitative comparison on Tanks and Temples and NeRF Synthetic**. We report F1-score (↑) between reconstructed and ground-truth point clouds. We highlight the **best**, second-best, and *third-best* Gaussian-based results. Time is in minutes.

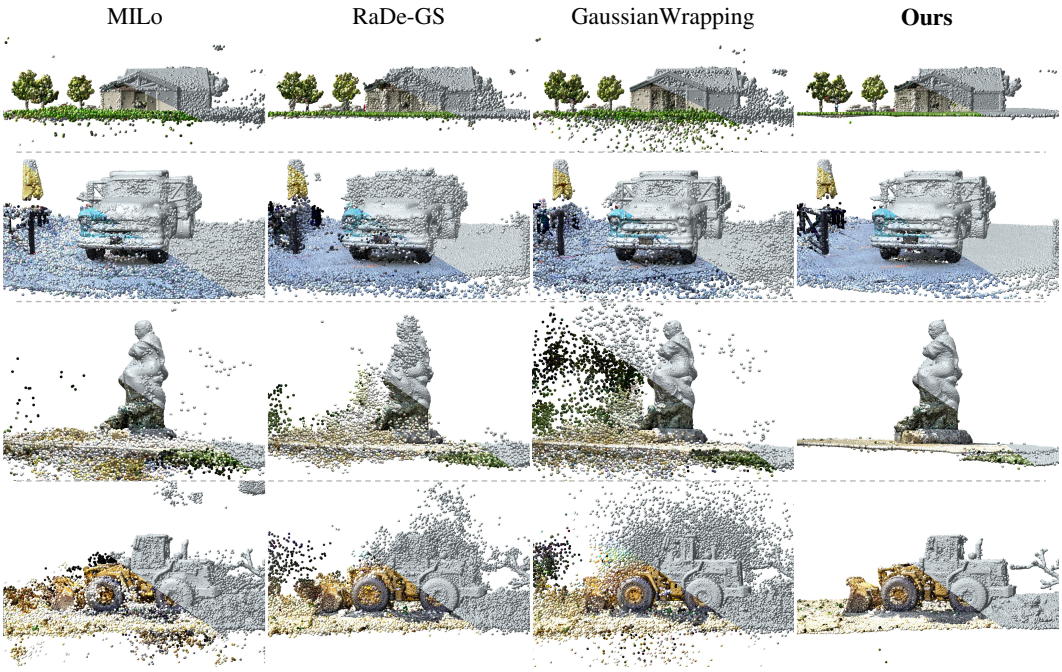

Figure 6: **Qualitative comparison on Tanks and Temples.** Compared with MILo, RaDe-GS, and GaussianWrapping, PCAsplat concentrates Gaussian centers along scene surfaces, with substantially fewer floaters around objects and below ground. See Fig. 12 and Fig. 11 for additional views of the geometry above and below ground.

4.2 Ablation Study

We evaluate the PCA loss $\mathcal{L}_{\text{PCA}}$ and the normal alignment loss $\mathcal{L}_{\text{normal}}$, individually and together. Tab. 4 reports Chamfer distance between Gaussian centers and DTU ground-truth point clouds. The PCA loss, which penalizes neighborhood thickness and promotes isotropic tangent-plane coverage, accounts for most of the improvement, reducing CD from 0.655 to 0.520. Combining it with $\mathcal{L}_{\text{normal}}$ yields the lowest CD of 0.519.

$\mathcal{L}_{\text{PCA}}$	$\mathcal{L}_{\text{normal}}$	CD ↓
		0.655
	✓	0.654
✓		0.520
✓	✓	**0.519**

Table 4: **Ablation study**.

4.3 Applications

We use the learned Gaussian centers directly as surface samples for instance segmentation and Poisson surface reconstruction. These experiments test how improved Gaussian-to-surface alignment benefits downstream geometry processing.

Point-cloud segmentation. We evaluate 3D semantic instance segmentation on ScanNet++ using point clouds extracted from Gaussian splatting, COLMAP, and VGGT-Ω reconstructions. We feed each method's colored point cloud into Mask3D (Schult et al., 2023) and apply the same pipeline to the ground-truth ScanNet++ point cloud to obtain an oracle reference. Tab. 5 shows that PCAsplat achieves the highest mIoU among the evaluated reconstructed point clouds, more than doubling the Top-1 and Top-3 mIoU of the strongest baseline, VGGT-Ω, and narrowing the gap to the oracle. Textureless regions, such as white walls, remain challenging, but PCAsplat produces more coherent masks with clearer object boundaries in the examples shown in Fig. 7.

	COLMAP-S	COLMAP-D	RaDe-GS	VGGT-Ω	**Ours**	Oracle
Top-1 mIoU	0.02%	8.21%	7.77%	10.54%	**24.35%**	30.41%
Top-3 mIoU	0.21%	8.20%	7.81%	10.58%	**24.84%**	30.56%

Table 5: **Semantic instance segmentation on ScanNet++.** We report Top-1 and Top-3 mIoU (↑) using Mask3D on point clouds reconstructed by each method.

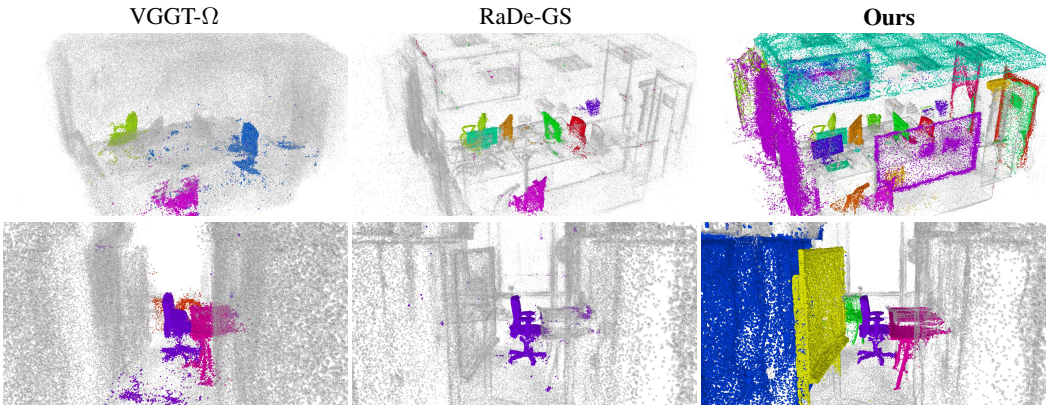

Figure 7: **Qualitative results of 3D semantic instance segmentation.** We apply Mask3D directly to colored point clouds reconstructed by each method on ScanNet++. PCAsplat produces more coherent object masks in the examples shown.

Poisson surface reconstruction. To evaluate the quality of the resulting oriented point clouds, defined by Gaussian centers and normals, we apply Screened Poisson Surface Reconstruction (SPSR) (Kazhdan & Hoppe, 2013). In the Tanks and Temples example in Fig. 8, PCAsplat produces cleaner meshes with finer geometric detail and fewer spurious components than RaDe-GS and GaussianWrapping.

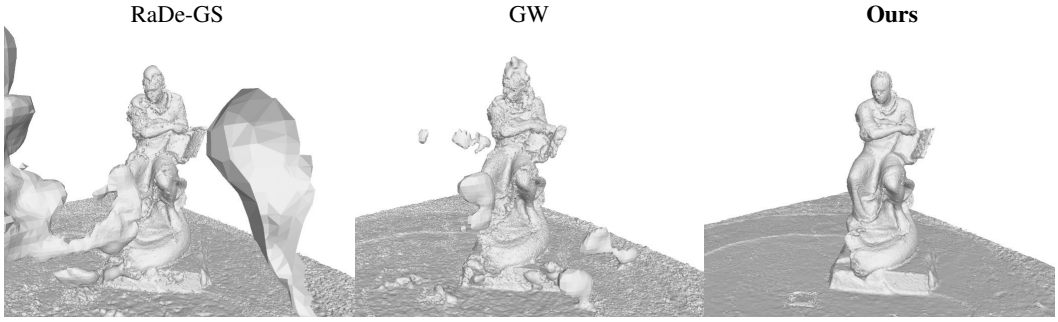

Figure 8: Using Gaussian centers and normals from PCAsplat, SPSR recovers finer statue details and a smoother ground surface, with substantially fewer spurious mesh components than with RaDe-GS or GaussianWrapping.

5 Conclusion

We presented PCAsplat, a geometry-aware Gaussian Splatting framework that aligns Gaussians to the surface by integrating local PCA regularization into training. Our method achieves surface-aligned Gaussians, which enables downstream tasks while still performing competitively in Novel View Synthesis and Mesh Extraction. This refined representation helps bridge classical point-set surface analysis and modern novel view synthesis, enabling downstream geometry-processing tasks such as 3D segmentation and Poisson surface reconstruction to be applied directly to the optimized Gaussian means. By encouraging Gaussians to lie on the underlying surface, PCAsplat also provides a natural basis for estimating local geometric quantities, such as curvature and density, opening promising directions for heat propagation (Zhou & Lähner, 2026) and geometry-aware pruning and densification (see Appendix D material). While PCAsplat introduces modest training overhead and requires a depth-based step for distant floaters and sign disambiguation for normals, it substantially improves local surface fidelity

AI USE STATEMENT

We used generative AI tools to assist with implementing and debugging methods. We also used these tools for manuscript editing and making visualization code. The authors take responsibility for the final text, claims, code, and experimental results.

REFERENCES

Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5460–5469. IEEE, 2022.

Danpeng Chen, Hai Li, Weicai Ye, Yifan Wang, Weijian Xie, Shangjin Zhai, Nan Wang, Haomin Liu, Hujun Bao, and Guofeng Zhang. Pgsr: Planar-based gaussian splatting for efficient and high-fidelity surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 31 (9):6100–6111, 2025.

Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pp. 303–312, 1996.

Pinxuan Dai, Jiamin Xu, Wenxiang Xie, Xinguo Liu, Huamin Wang, and Weiwei Xu. High-quality surface reconstruction using gaussian surfels. In *ACM SIGGRAPH 2024 conference papers*, pp. 1–11, 2024.

Diego Gomez, Antoine Guédon, Nissim Maruani, Bingchen Gong, and Maks Ovsjanikov. From blobs to spokes: High-fidelity surface reconstruction via oriented gaussians. In *European Conference on Computer Vision*, pp. 54–72. Springer, 2026.

Anne Greenbaum, Ren-cang Li, and Michael L Overton. First-order perturbation theory for eigenvalues and eigenvectors. *SIAM review*, 62(2):463–482, 2020.

Antoine Guédon and Vincent Lepetit. Sugar: Surface-aligned gaussian splatting for efficient 3d mesh reconstruction and high-quality mesh rendering. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5354–5363. IEEE, 2024.

Antoine Guédon, Diego Gomez, Nissim Maruani, Bingchen Gong, George Drettakis, and Maks Ovsjanikov. Milo: Mesh-in-the-loop gaussian splatting for detailed and efficient surface reconstruction. *ACM Transactions on Graphics (TOG)*, 44(6):1–15, 2025.

Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. Surface reconstruction from unorganized points. In *Proceedings of the 19th annual conference on computer graphics and interactive techniques*, pp. 71–78, 1992.

Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2d gaussian splatting for geometrically accurate radiance fields. In *ACM SIGGRAPH 2024 conference papers*, pp. 1–11, 2024.

Rasmus Jensen, Anders Dahl, George Vogiatzis, Engin Tola, and Henrik Aanæs. Large scale multi-view stereopsis evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 406–413, 2014.

Michael Kazhdan and Hugues Hoppe. Screened poisson surface reconstruction. *ACM Transactions on Graphics (ToG)*, 32(3):1–13, 2013.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), July 2023. URL https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/.

Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)*, 36(4):1–13, 2017.

Shujuan Li, Yu-Shen Liu, and Zhizhong Han. Gaussianudf: Inferring unsigned distance functions through 3d gaussian splatting. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 27113–27123. IEEE, 2025.

Zhaoshuo Li, Thomas Müller, Alex Evans, Russell H Taylor, Mathias Unberath, Ming-Yu Liu, and Chen-Hsuan Lin. Neuralangelo: High-fidelity neural surface reconstruction. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8456–8465. IEEE, 2023.

Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

Mark Pauly, Markus Gross, and Leif P Kobbelt. Efficient simplification of point-sampled surfaces. In *IEEE Visualization, 2002. VIS 2002.*, pp. 163–170. IEEE, 2002.

Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4104–4113, 2016.

Jonas Schult, Francis Engelmann, Alexander Hermans, Or Litany, Siyu Tang, and Bastian Leibe. Mask3d: Mask transformer for 3d semantic instance segmentation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 8216–8223. IEEE, 2023.

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5294–5306. IEEE, 2025.

Jianyuan Wang, Minghao Chen, Shangzhan Zhang, Nikita Karaev, Johannes Schönberger, Patrick Labatut, Piotr Bojanowski, David Novotny, Andrea Vedaldi, and Christian Rupprecht. VGGT-Ω. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *arXiv preprint arXiv:2106.10689*, 2021.

Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. *Advances in neural information processing systems*, 34:4805–4815, 2021.

Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. Scannet++: A high-fidelity dataset of 3d indoor scenes. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 12–22. IEEE, 2023.

Mulin Yu, Tao Lu, Linning Xu, Lihan Jiang, Yuanbo Xiangli, and Bo Dai. Gsdf: 3dgs meets sdf for improved neural rendering and reconstruction. *Advances in Neural Information Processing Systems*, 37:129507–129530, 2024a.

Zehao Yu, Torsten Sattler, and Andreas Geiger. Gaussian opacity fields: Efficient adaptive surface reconstruction in unbounded scenes. *ACM Transactions on Graphics (ToG)*, 43(6):1–13, 2024b.

Baowen Zhang, Chuan Fang, Rakesh Shrestha, Yixun Liang, Xiao-Xiao Long, and Ping Tan. Rade-gs: Rasterizing depth in gaussian splatting. *ACM Transactions on Graphics*, 45(2):1–14, 2026.

Wenyuan Zhang, Yu-Shen Liu, and Zhizhong Han. Neural signed distance function inference through splatting 3d gaussians pulled on zero-level set. *Advances in Neural Information Processing Systems*, 37:101856–101879, 2024.

Ziyu Zhang, Binbin Huang, Hanqing Jiang, Liyang Zhou, Xiaojun Xiang, and Shuhan Shen. Quadratic gaussian splatting: High quality surface reconstruction with second-order geometric primitives. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 28260–28270. IEEE, 2025.

Hongyu Zhou and Zorah Lähner. Laplace-beltrami operator for gaussian splatting. In *2026 International Conference on 3D Vision (3DV)*, pp. 1107–1116. IEEE, 2026.

APPENDIX

A IMPLEMENTATION DETAILS.

Our method is implemented in `PyTorch` on top of the RaDe-GS (Zhang et al., 2026) codebase. We first train the 3D Gaussians with $\mathcal{L}_{\text{RGB}}$ for 7K iterations, and then enable PCA losses, $\mathcal{L}_{\text{PCA}}$ and $\mathcal{L}_{\text{normal}}$, for the remaining iterations. We train bounded scenes for 20K iterations and unbounded scenes for 30K iterations. To accelerate PCAsplat, we implemented a custom CUDA kernel for batched 3×3 eigendecomposition, since `torch.linalg.eigh` relies on general-purpose routines that are not optimized for this setting. All experiments are conducted on an NVIDIA RTX A5500 with 24GB of VRAM.

Results Reproducibility We ran all methods using their most up-to-date GitHub repositories on all datasets using their specified hyperparameters for each dataset; however, for NeRF Synthetic we used the DTU hyperparameters for all methods, this is due to the original hyperparameters used for this dataset being PSNR-focused and lacking surface regularizations. For VGGT-Ω, we unproject the predicted depth maps through the predicted camera poses to obtain a metric-scaled point cloud. We then estimate a similarity that aligns the predicted cameras to the dataset COLMAP poses and apply the same transform to the point cloud, placing it in the ScanNet++ coordinate frame. COLMAP-S stands for a sparse point output, while COLMAP-D stands for a dense output. Tab. 1 and Tab. 3 have all results generated with our runs of each method with their specified hyperparameters for each dataset.

B SYNTHETIC MESH DATASET

NeRF Synthetic (Mildenhall et al., 2021) provides synthetic Blender scenes with posed views but no official surface mesh for 3D reconstruction metrics. We extract only the visible mesh from each scene's original 3D assets downloaded from Blendswap. First, all components are merged via self-intersection so shared seams become explicit edges, then hidden-surface removal is run for every train, validation, and test camera, where we raycast each view and keep a triangle if and only if it is the first hit for at least one pixel, yielding a watertight surface that matches what a 3D reconstruction method can actually observe. The resulting visible meshes are the exclusive reference for the F-score measured from 10^6 Farthest Point Sampling (FPS) samples. This dataset will be made available. See Fig. 9. Additionally, our pipeline for creating such a dataset is useful for evaluating future and previous works on mesh extraction, and creating newer, more precise datasets.



Figure 9: Visible meshes from the NeRF Synthetic eight scenes.

C ADITIONAL RESULTS

Asset-Ready Splats Unlike standard 3DGS pipelines, PCAsplat directly generate asset-ready splats suitable for downstream pipelines. In conventional workflows, isolating a reconstructed object within interactive editing software (e.g., SuperSplat) requires extensive manual cleanup to clean floaters and artifacts. Crucially, such manual pruning often removes important primitives: for instance, filtering Gaussians beneath ground may degrade the opacity of surfaces, introducing trans-

parency. By enforcing surface alignment across the Gaussian primitives, PCAsplat suppresses unconstrained floating artifacts and as result, target objects can be isolated via simple bounding crops and used out of the box, eliminating manual post-processing while strictly preserving opaque surface integrity. Fig. 10 showcases a comparison between PCAsplat and previous works that have surface regularization.



Figure 10: **Qualitative comparison of object isolation**. Previous methods leave floating primitives and subsurface artifacts (red arrows) upon cropping, needing further manual editing to erase them. In contrast, PCAsplat (Ours) achieves clean, artifact-free boundaries, yielding an asset-ready reconstruction without manual cleanup.

Overworld and underworld Ray-Contribution supervision alone fails to constrain regions with sparse visibility or with low contributing Gaussians, such as subterranean volumes beneath the ground or occluded regions behind walls. In Fig. 11, we show the "underworld" from beneath the scene, prior baselines suffer from floaters and disorganized Gaussian sprawl below the ground. In contrast, PCAsplat regularizes primitives along the floor plane, effectively suppressing subsurface spurious floaters and recovering clean, well-behaved boundary geometry (e.g., the walls in the *Barn* scene). Concurrently, as shown in the "overworld" views in Fig. 12, PCAsplat improves structural contours, accurately recovers thin planar elements, and filters out surface noise across all open world scenes in the Tanks and Temples benchmark.

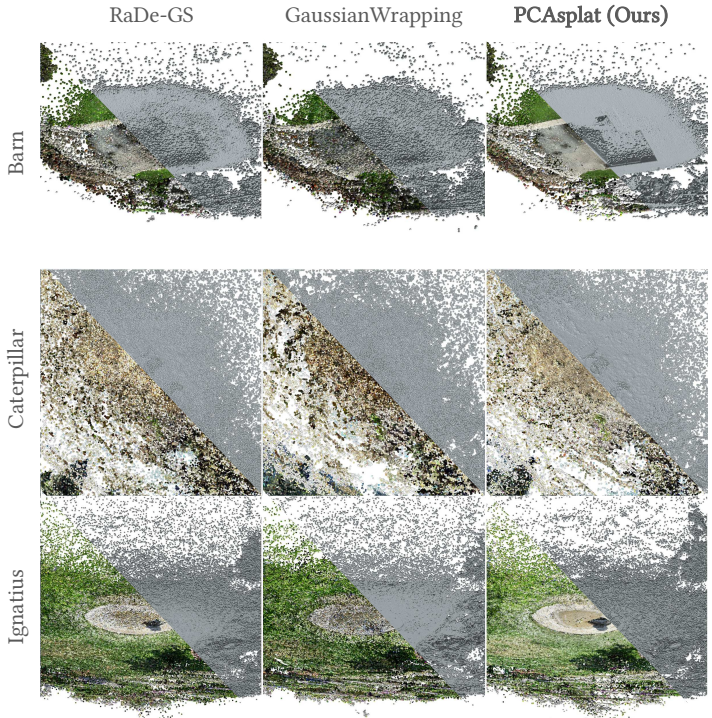

Figure 11: **"Underworld" geometry comparison on Tanks and Temples.** We visualize the Gaussian means of reconstructed scenes from beneath the ground. Baselines fail to control interior floaters in unobserved regions and behind structures. PCAsplat **(Ours)** suppresses all floaters, resulting in regularized, clean ground and wall boundaries.

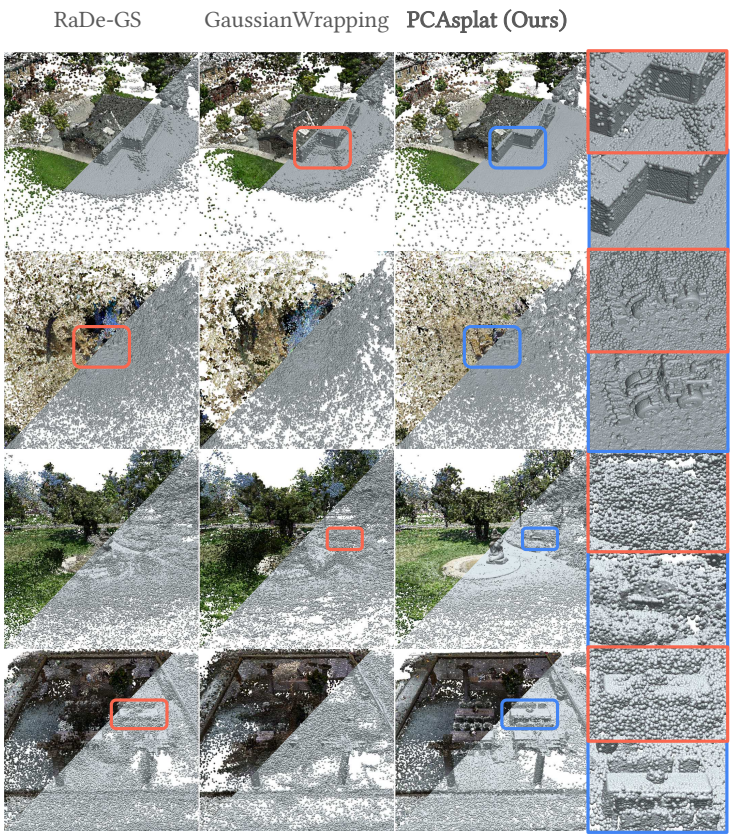

Figure 12: **Qualitative point cloud detail comparison on Tanks and Temples** Across diverse outdoor scenes, baseline methods produce noisy clouds. PCAsplat reconstructs scenes with high geometric fidelity, cleanly resolving intricate structures (e.g., roofs, vehicle contours, and ground details) while removing floating outliers.

Surface Reconstruction

We evaluate PCAsplat on surface reconstruction using standard mesh extraction from Gaussian splatting representations. Since our method directly regularizes Gaussian centers through local PCA geometry, we also evaluate the learned splat cloud itself as a point-set surface. We report quantitative results on DTU and Tanks and Temples, provide qualitative mesh comparisons, and visualize the spatial distribution of the learned Gaussian point clouds.

Table 6: **Quantitative comparison on the DTU dataset**. We report the Chamfer Distance (CD) ↓ across 15 scenes and the average optimization time ↓. Our method achieves the best mean accuracy with a competitive performance.

	Methods	24	37	40	55	**63**	65	69	83	97	105	106	110	114	118	122	Mean	Time
Neural	VolSDF (Yariv et al., 2021)	1.14	1.26	0.81	0.49	1.25	0.7	0.72	1.29	1.18	0.7	0.66	1.08	0.42	0.61	0.55	0.857	12h¿
	Neus (Wang et al., 2021)	1.00	1.37	0.93	0.43	1.10	0.65	0.57	1.48	1.09	0.83	0.52	1.20	0.35	0.49	0.54	0.837	12h¿
	Neuralangelo (Li et al., 2023)	0.37	0.72	0.35	0.35	0.87	0.54	0.53	1.29	0.97	0.73	0.47	0.74	0.32	0.41	0.43	0.610	12h¿
Gaussian Based	3DGS (Kerbl et al., 2023)	2.14	1.53	2.08	1.68	3.49	2.21	1.43	2.07	2.22	1.75	1.79	2.55	1.53	1.52	1.50	1.966	**5.2m**
	SuGaR (Guédon & Lepetit, 2024)	1.47	1.33	1.13	0.61	2.25	1.71	1.15	1.63	1.62	1.07	0.79	2.45	0.98	0.88	0.79	1.330	52m
	2DGS (Huang et al., 2024)	0.50	0.72	0.39	0.39	0.93	0.85	0.78	1.21	1.12	0.67	0.66	1.14	0.43	0.67	0.50	0.731	10.9m
	GS-Pull (Zhang et al., 2024)	0.51	0.56	0.46	0.39	0.82	0.67	0.85	1.37	1.25	0.73	0.54	1.39	0.35	0.88	0.42	0.746	21.8m
	GOF (Yu et al., 2024b)	0.48	0.82	0.39	0.38	1.36	0.86	0.75	1.20	1.31	0.83	0.65	1.30	0.48	0.69	0.53	0.802	55m
	PGSR (Chen et al., 2025)	**0.34**	0.54	0.44	0.37	0.78	0.57	0.49	1.06	**0.63**	0.59	0.47	**0.50**	**0.30**	0.37	0.34	0.519	30.5m
	MILo (Guédon et al., 2025)	0.43	0.74	0.34	0.37	0.80	0.74	0.70	1.21	1.22	0.66	0.62	0.8	0.37	0.76	0.48	0.683	25.0m
	RaDe-GS (Zhang et al., 2026)	0.37	0.50	0.28	0.31	0.78	0.48	0.46	1.05	0.64	0.52	0.33	0.57	0.31	0.33	0.34	0.485	14.5m
	GW (Gomez et al., 2026)	0.40	0.51	0.27	0.32	0.81	0.46	0.45	1.04	0.65	0.52	0.33	0.59	0.32	0.33	0.34	0.489	—
	PCAsplat (ours)	0.36	**0.47**	**0.26**	**0.30**	**0.74**	**0.46**	**0.44**	**0.99**	0.64	**0.51**	**0.32**	0.63	0.31	**0.31**	**0.32**	**0.471**	22.3m

Mesh extraction. As shown in Tab. 6 and Tab. 7, PCAsplat achieves SoTA or competitive performance across both benchmarks. On DTU, our method obtains the lowest mean CD of 0.471 using TSDF extraction, improving over the strongest baseline. On Tanks and Temples, our method achieves a competitive F1-score among Gaussian-based methods, matching the best mean score while maintaining a cleaner underlying point cloud. Qualitatively, Fig. 13 shows that PCAsplat pro-

duces meshes with finer detail. Compared to 2DGS, RaDe-GS, and MILo, our reconstructions better preserve fine structures, such as building windows, scissor tips, and plush texture, while reducing spurious artifacts around specular and high-curvature regions. Another reason, beyond better geometry, is that our model may have additional learning capacity because its parameters focus on what matters most: the surface.

For evaluation, we followed the standard splatting mesh-extraction pipeline: we render depth maps from all views with a maximum depth of 5 and fuse them with TSDF (Curless & Levoy, 1996) using a voxel size of 0.002 using Open3D implemented TSDF.

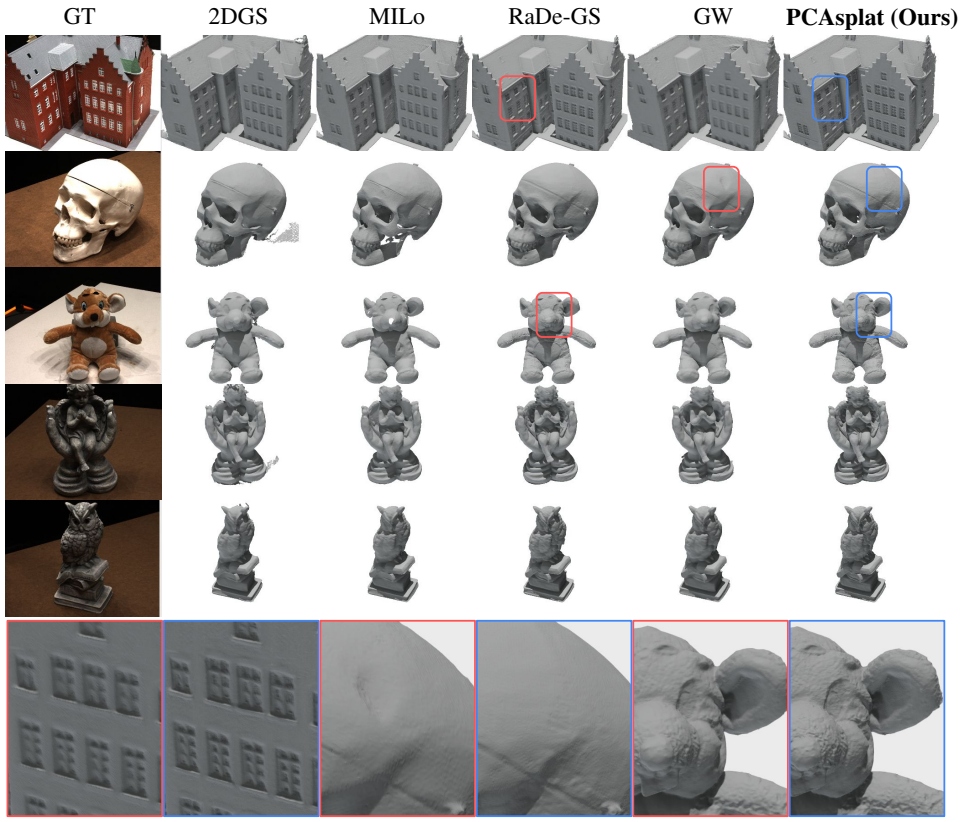

Figure 13: **Qualitative comparison on the DTU Dataset.** Our method produces cleaner meshes with more details (e.g. teddy bear plush or house windows) and reduced artifacts (e.g. scissor tip, fruit specularity or skull forehead).

	Methods	Barn	Caterp.	Court.	Ignat.	Meet.	Truck	Mean	Time
Neural	Neus	0.29	0.29	0.17	0.83	0.24	0.45	0.38	12h+
	Geo-Neus	0.33	0.26	0.12	0.72	0.20	0.45	0.35	12h+
	Neuralangelo	0.70	0.36	0.28	0.89	0.32	0.48	0.50	12h+
Gaussian Based	3DGS	0.13	0.08	0.09	0.04	0.01	0.19	0.09	**7.9m**
	SuGaR	0.14	0.16	0.08	0.44	0.16	0.26	0.21	73m
	2DGS	0.36	0.23	0.13	0.44	0.16	0.26	0.26	12.3m
	GS-Pull	0.60	0.37	0.16	0.71	0.22	0.52	0.43	37.6m
	GOF	0.51	0.38	0.26	0.68	0.24	0.52	0.43	69m
	PGSR	*0.66*	0.41	0.21	**0.80**	0.29	0.60	*0.50*	72m
	MILo	0.64	0.38	0.31	*0.76*	0.28	0.59	0.49	110m
	RaDe-GS*	**0.69***	0.41*	0.18*	**0.80***	**0.40***	0.62*	0.52*	*18.3m**
	GW	0.65	**0.50**	**0.35**	0.75	0.39	*0.61*	**0.54**	27m
	PCAsplat	0.64	0.38	*0.21*	0.73	0.39	**0.63**	*0.50*	53m

Table 7: **Quantitative comparison on the Tanks and Temples dataset**. We report F1-score ↑. We evaluate all results using the scores reported in the papers. PCAsplat achieves competitive results while producing a cleaner underlying point cloud. * The results presented are from RaDe-GS paper. However, due to compute-limiting constraints, we were unable to re-run RaDe-GS TSDF extraction and achieve the same results as they show in their table. As such, we also append here the average results we got from running their code on our end, **0.49**, thus, our 0.50 improves over 0.49, and makes our method competitive with SoTA results.

Novel View Synthesis Our primary focus is improving Gaussian surface alignment; however, to comply with common 3DGS evaluations, we evaluate the novel view synthesis quality of our representation. Tab. 8 presents the MipNeRF-360 dataset with our method maintaining competitive performance across all scenes, which demonstrates that our PCA regularization does not hinder the appearance of the scene.

	Indoor Scenes			Outdoor Scenes		
Method	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
3DGS	30.41	0.920	0.189	24.64	0.731	0.234
Mip-Splatting	**30.90**	0.921	0.194	24.65	0.729	0.245
BakedSDF	27.06	0.836	0.258	22.47	0.585	0.349
SuGaR	29.43	0.906	0.225	22.93	0.629	0.356
2DGS	30.40	0.916	0.195	24.34	0.717	0.246
GOF	<u>30.79</u>	0.924	0.184	<u>24.82</u>	<u>0.750</u>	<u>0.202</u>
RaDe-GS	30.74	*0.928*	<u>0.165</u>	**25.17**	**0.764**	**0.199**
MiLo	*30.76*	**0.934**	**0.155**	*24.81*	*0.744*	*0.229*
Ours	30.46	<u>0.930</u>	*0.170*	24.70	0.730	0.250

Table 8: Quantitative comparison across indoor and outdoor scenes of the MipNeRF 360 dataset (Barron et al., 2022). We highlight the **best**, <u>second-best</u>, and *third-best* PSNR, SSIM, and LPIPS. Our method maintains competitive performance while significantly improving Gaussian localization.

Code integration To facilitate integration with existing frameworks, our surface alignment framework is designed as a modular, plug-and-play extension that remains entirely decoupled from the core 3D Gaussian Splatting rasterization pipeline. By encapsulating the Principal Component Analysis (PCA) regularization, users can seamlessly integrate our geometric constraints—such as rotational, isotropic, and surfel losses—into existing optimization loops. As demonstrated in the code snippet 1, applying our method requires minimal code modification.

```
pca_pipeline = PCAPipeline(pca_params)
for iteration in range(num_iterations):
    # Accumulate our geometric constraints
    pca_pipeline.update(iteration, gaussians, visibility_filter, opt=opt)
    surface_loss = pca_pipeline.get_surface_loss()
    rotational_loss = pca_pipeline.get_rotational_loss()
    isotropic_loss = pca_pipeline.get_isotropic_loss()
    surfel_loss = pca_pipeline.get_surfel_loss()
```

Listing 1: Integration of our modular PCA regularization into a standard training loop. The code is entirely decoupled from the core rasterizer.

D Other Geometrical Statistics

Using our learned PCA eigenspectrum, we leverage well-known formulas for calculating the local density and local curvature of our Gaussian, allowing geometry inspection of trained splats. A shown by (Pauly et al., 2002) the local curvature for a neighborhood of size k can be evaluated via:

$$\sigma_k = \frac{\lambda^{(1)}}{\lambda^{(1)} + \lambda^{(2)} + \lambda^{(3)}}.$$

Additionally, the point cloud density can be easily calculated via the inverse of the root mean squared distance to its K-nearest neighbors

$$\rho = \frac{1}{\sqrt{D}} \quad D = \frac{1}{k} \sum_{\mu_j \in N_K(i)} \|\mu_i - \mu_j\|^2. \tag{6}$$

We visualize both of these equations applied to the splatting context in Fig. 14.

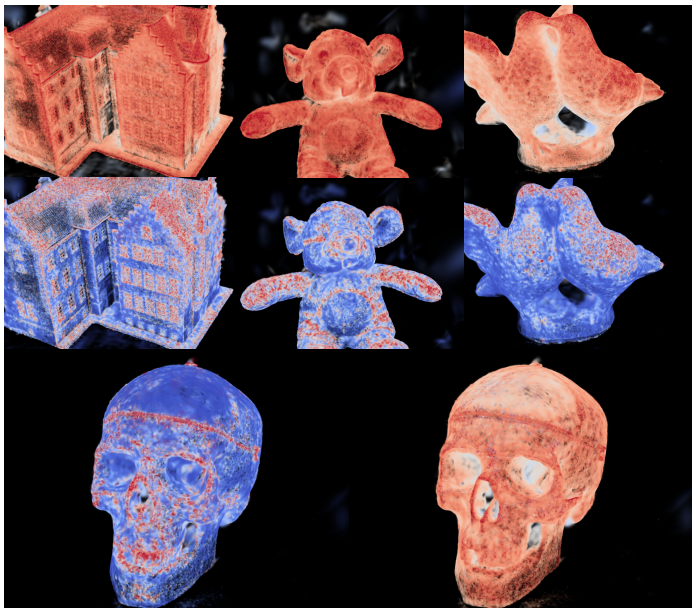

Figure 14: **Curvature (Blue-Red) and Density (Red) splatting.** Our method enables local curvature and local density rendering.