From Blobs to Spokes: High-Fidelity Surface Reconstruction via Oriented Gaussians

Diego Gomez[1⋆], Antoine Guédon[1⋆], Nissim Maruani[1,2], Bingchen Gong[1], and Maks Ovsjanikov[1]

[1] LIX, École Polytechnique, France
[2] Inria, Côte d'Azur, France

Fig. 1: High-fidelity surface reconstruction from 3D Gaussian Splatting. Our method, Gaussian Wrapping, reconstructs watertight and textured surface meshes of full 3D scenes given multiview RGB images, by interpreting 3D Gaussians as stochastic oriented surface elements. This figure illustrates the resulting surface meshes, with and without the RGB texture. Our meshes represent the full scene—including background geometry and extremely thin structures such as bicycle spokes where existing methods fail—with a *significantly* more compact representation than concurrent works [10,57].

Abstract. 3D Gaussian Splatting (3DGS) has revolutionized fast novel view synthesis, yet its opacity-based formulation makes *surface extraction* fundamentally difficult. Unlike implicit methods built on Signed Distance Fields or occupancy, 3DGS lacks a global geometric field, forcing existing approaches to resort to heuristics such as TSDF fusion of blended depth maps. Inspired by the *Objects as Volumes* framework [39], we derive a principled occupancy field for Gaussian Splatting and show how it can be used to extract highly accurate watertight meshes of complex scenes. Our key contribution is to introduce a learnable oriented normal at each Gaussian element and to define an adapted attenuation formulation, which leads to closed-form expressions for both the normal and occupancy fields at arbitrary locations in space. We further introduce a novel consistency loss and a dedicated densification strategy to enforce Gaussians to *wrap* the entire surface by closing geometric holes,

⋆ Both authors contributed equally to the paper.

ensuring a complete shell of oriented primitives. We modify the differentiable rasterizer to output depth as an isosurface of our continuous model, and introduce **Primal Adaptive Meshing** for Region-of-Interest meshing at arbitrary resolution. We additionally expose fundamental biases in standard surface evaluation protocols and propose two more rigorous alternatives. Overall, our method **Gaussian Wrapping** sets a new state-of-the-art on DTU and Tanks and Temples, producing complete, watertight meshes at a fraction of the size of concurrent work—recovering thin structures such as the notoriously elusive bicycle spokes. Our project page is available here.

Keywords: 3D Gaussian Splatting · Surface Reconstruction · Meshes

1 Introduction

Reconstructing high-quality 3D surfaces from a set of 2D images is a key problem in computer vision. While recent advances in neural rendering, such as NeRF [38] and 3DGS [25], have shown impressive results on the novel view synthesis problem, extracting accurate and explicit geometry continues to be challenging. To bridge efficient volumetric rendering and surface reconstruction requires addressing their inherent differences: neural rendering is particularly suited to soft, semi-transparent representations, whereas surface reconstruction demands hard, well-defined boundaries to extract watertight geometry.

Current approaches to this inverse problem generally fall into two categories. Implicit representations, such as Signed Distance Functions (SDFs) and occupancy fields, excel at representing continuous, topologically consistent surfaces. However, they are often computationally expensive to train and tend to produce over-smoothed results, as their global solvers struggle to capture high-frequency details [29, 50]. On the other hand, explicit particle-based methods (i.e, 3DGS) exhibit real-time rendering and high-fidelity visual quality by representing scenes as discrete primitives. While they can capture fine details, there is no straightforward way to recover an ordered structure, i.e, a mesh, from such representations. Recent hybrid approaches attempt to close the gap by regularizing 3DGS [18, 19, 42, 44, 55, 57], but often resort to ad-hoc density thresholds or compromise the rendering speed and quality that make 3DGS attractive.

A core limitation of extracting surfaces from 3DGS is the interpretation of the primitive itself. Standard approaches treat Gaussians as symmetric "blobs" of mass or density. This assumption conflicts with the nature of an orientable surface, which is an asymmetric boundary separating empty and occupied space. By modeling surface points with symmetric primitives, existing methods bias reconstruction, and make surface extraction significantly more challenging.

In this work, we address this limitation by introducing **Gaussian Wrapping**, a novel framework that reinterprets the role of the Gaussian primitive inspired by Objects as Volumes [39]. Rather than treating Gaussians as symmetric clouds, we view them as *oriented probabilistic surface elements* that represent the underlying geometry only within an oriented half-space. In this view, the

(a) Mesh w/wo texture **(b)** Close-up view of Mesh w/wo texture

Fig. 2: Examples of textured meshes reconstructed with our method. While being watertight and lighter than meshes reconstructed by state-of-the-art methods, our approach is able to reconstruct extremely fine details such as a few blades of grass—as can be seen in the close-up view of the stump scene [5].

Gaussian models the density decay on the outward-facing side of the boundary, while the inward-facing side is considered as fully occupied. This formulation allows us to derive a robust "wrapping" strategy: we identify geometric gaps where our oriented surface assumption is violated and densify the representation to obtain a watertight shell of oriented primitives. Our framework allows the derivation of closed-form geometric quantities—namely occupancy, its complement (*vacancy*), and the normal field—enabling reconstruction of thin structures without artifacts or missing geometry. Links to extracted meshes, code and more are available here. Our key contributions are as follows.

- We introduce **Oriented-Gaussians** and their associated training strategy in the multi-view setting.
- We derive a theoretical connection between 3DGS and implicit surface reconstruction by formulating Gaussians as oriented surface elements, inspired by *Objects as Volumes* [39]. Importantly, this leads to closed-form expressions for both normal and occupancy fields at arbitrary locations without any additional learnable parameters.
- We propose **Primal Adaptive Meshing**, a mesh extraction procedure that leverages the derived Gaussian fields to produce high-quality, watertight meshes at controllable resolution, enabling recovery of extremely thin structures such as bicycle spokes (Fig. 1).

2 Related work

Novel View Synthesis. NeRF [38] models scenes as continuous radiance fields whose image formation relies on ray-marching through an attenuation field (also called *density*) parameterized by an MLP. Subsequent work improves anti-aliasing [4,5] and training efficiency via hash encodings [40] or explicit volumetric structures [9,15]. 3D Gaussian Splatting [25] redefined this landscape by replacing ray-marching with rasterization of learned Gaussian primitives, achieving

real-time, high-quality rendering. Further extensions address aliasing [54], scalability [26, 35], and rendering fidelity. Despite impressive visual quality, none of these methods target explicit surface mesh extraction.

Surface Reconstruction from Images. Implicit approaches coupling volume rendering with signed distance functions (NeuS [50], VolSDF [51], Neuralangelo [29]) convert a learnable SDF into a view-dependent attenuation field and yield smooth, topologically consistent surfaces but require very long optimization times. Mesh Baking [43, 52] approaches employ a two-stage pipeline that first trains a volumetric representation, then extracts a mesh to bake rendering onto it; this line of work however focuses on rendering and does not measure the geometric loyalty of the produced meshes. Discrete mesh methods [41, 45] produce strong results on isolated objects but do not scale to entire scenes.

In this regard, the efficiency of Gaussian Splatting motivates a growing family of surface-extraction approaches. Several jointly optimize Gaussians with an auxiliary neural implicit field [11, 28, 31, 53, 59], but the dual optimization limits scalability and the implicit branch remains bounded by MLP capacity. Others regularize Gaussians directly: 2DGS [21] replaces 3D primitives with lower-dimensional ones, RaDe-GS [56], PGSR [10], Quadratic Gaussian Splatting [60], and VCR-GauS [12] explore alternative rendering strategies or multi-view consistency constraints, while SuGaR [19] and GOF [55] propose specialized extraction procedures. A shared limitation is the reliance on heuristic depth definitions—typically TSDF fusion of blended depth maps—without a principled link between the Gaussian representation and a well-defined geometric field.

Objects as Volumes [39] establishes a closed-form relationship between vacancy and view-dependent attenuation under reciprocal exponential transport, grounding the theory behind Neural SDF methods [50, 51]. Since the 3DGS image formation model is not attenuation-based, this result does not apply directly—though Celarek *et al.* [8] show 3DGS approximates volumetric rendering with view-dependent attenuation. Concurrent work GGGS [57] builds upon [39] and exploits a continuous transmittance derived from Gaussians for improved depth estimation. However, their formulation suffers from surface erosion and fails to capture fine geometric details. Our method addresses this limitation by introducing oriented Gaussians that yield a reciprocal attenuation, from which closed-form vacancy and normal fields naturally follow—enabling principled, high-fidelity mesh extraction and limiting erosion.

Predicting Normals from Radiance Fields. Estimating normals from radiance fields is established practice in inverse rendering, where accurate normal maps help disentangle material properties [16, 24, 30, 34, 49]. In contrast, we predict normals not for appearance decomposition but to construct an oriented surface—directly enabling robust surface reconstruction.

Voronoi & Delaunay-based Methods. Voronoi diagrams and their dual Delaunay triangulations were first utilized as backbones for surface reconstruction with provable guarantees [1–3, 13], and later integrated into learning-based pipelines [6, 36, 37, 46–48, 58]. Within neural rendering, GOF [55] and MILo [18]

build Delaunay triangulations over Gaussian-derived vertices, while other methods [17,20,33] integrate them into ray-tracing pipelines for view synthesis rather than reconstruction. In contrast, our Primal Adaptive Meshing incorporates ideas from modern reconstruction pipelines [58] to generate Delaunay-based watertight meshes at any desired resolution.

3 Background and Motivation

We focus on *fully opaque objects*, where the scene is binary: each point is either inside or outside the object. Our goal is to derive a *principled occupancy field* from 3D Gaussian Splatting, *i.e.*, a view-independent scalar field encoding scene geometry, from which high-quality surface meshes can be extracted (Fig. 2).

Objects as Volumes. Reconstructing well-defined surfaces requires a properly defined geometric field such as an occupancy function $\mathcal{O} : \mathbb{R}^3 \to [0, 1]$, yet volumetric rendering—the backbone of both NeRF [38] and 3DGS [25]—operates on continuous, semi-transparent volumes rather than on sharp boundaries. *Objects as Volumes* (OaV) [39] bridges this gap by interpreting the scene as a stochastic surface: the occupied region is modeled as a random set and the occupancy $\mathcal{O}$ as the probability of a point to be occupied. Standard volumetric rendering quantities then emerge as expectations over this random geometry.

Specifically, under the assumption of exponential light transport, this framework grounds the theoretical justification of Neural SDFs [50, 51], and links the *vacancy* $v(\mathbf{x}) = 1 - \mathcal{O}(\mathbf{x})$ (the probability that $\mathbf{x} \in \mathbb{R}^3$ is unoccupied) to the attenuation (or density) $\sigma : \mathbb{R}^3 \times \mathcal{S}^2 \to \mathbb{R}_+$ of standard volumetric ray-marching:

$$\forall (\mathbf{x}, \mathbf{w}) \in \mathbb{R}^3 \times \mathcal{S}^2, \quad \sigma(\mathbf{x}, \mathbf{w}) = |\mathbf{w} \cdot \nabla \log v(\mathbf{x})| \,, \tag{1}$$

where attenuation is assumed to be *reciprocal*, *i.e.* $\sigma(\mathbf{x}, \mathbf{w}) = \sigma(\mathbf{x}, -\mathbf{w})$. This equation directly connects a geometric field, the vacancy, to the attenuation driving volumetric rendering. It is the cornerstone of our approach: if Gaussian Splatting can be cast as an attenuation-based model satisfying reciprocity, then Eq. 1 yields an explicit occupancy field and, in turn, a principled surface.

Gaussian Splatting. 3D Gaussian Splatting [25] represents a scene as N Gaussian primitives, each defined by $G_i(\mathbf{x}) = \alpha_i \exp(-\frac{1}{2}(\mathbf{x} - \mu_i)^T \Sigma_i^{-1}(\mathbf{x} - \mu_i))$, with opacity $\alpha_i \in [0, 1]$, mean $\mu_i \in \mathbb{R}^3$, and precision matrix $\Sigma_i^{-1} \in \mathbb{S}^3_{++}$. Unlike NeRF [38], which ray-marches through a continuous attenuation field σ, 3DGS relies on *opacity*: it renders the pixel p by projecting each primitive onto the image plane and alpha-compositing contributions in depth order:

$$C(p) = \sum_i c_i \, \alpha_i G_i^*(p) \prod_{j<i} \big(1 - \alpha_j G_j^*(p)\big) \,, \tag{2}$$

where G_i^* is either a projected 2D Gaussian as in 3DGS [25], or the evaluation of the 3D Gaussian at the point of maximum contribution along the ray [55–57].

A key consequence is that each Gaussian's contribution depends only on its projected value at the pixel—not on the ray path length through the primitive. Gaussian opacities are therefore bounded in $[0, 1]$ and cannot be identified with the potentially unbounded attenuation coefficient σ. As a result, the OaV framework cannot be applied to 3DGS directly.

Finding an attenuation model equivalent to 3DGS. Recent work [8] shows that the 3DGS image formation model can nonetheless be reinterpreted as *"a volumetric rendering model with view-dependent extinction"*, assuming primitives do not overlap in 3D. To be precise, we show in the appendix that, *assuming statistical independence and non-overlapping Gaussian primitives*, the image formation model of a Gaussian Splatting representation—relying on alpha-blending with opacity—is equivalent to a ray-marching volumetric rendering model with the following continuous attenuation:

$$\sigma(\mathbf{x}, \mathbf{w}) = \sum_{i=1}^{N} \max\left(0, -\mathbf{w} \cdot \nabla \log(1 - G_i(\mathbf{x}))\right) ,$$

for any ray origin $\mathbf{o} \in \mathbb{R}^3$ in empty space, $\mathbf{w} \in \mathcal{S}^2$, and $t \in [0, +\infty)$. Contrary to previous work [57], this attenuation factor enforces geometric properties necessary for accurate surface extraction; please refer to the appendix for more details. However, this expression is not reciprocal in general, *i.e.* $\sigma(\mathbf{x}, \mathbf{w}) \neq \sigma(\mathbf{x}, -\mathbf{w})$. Reciprocity is a necessary condition for applying results from OaV, thus preventing us from deriving a closed-form vacancy expression.

A reciprocal attenuation model. To restore reciprocity, we propose to *orient* Gaussians: we equip each primitive with a learnable, oriented normal vector. This results in an explicit attenuation-based formulation of 3DGS that satisfies the requirements of OaV, yielding both more accurate depth rendering and an explicit geometric field for surface extraction. This attenuation-based formulation also induces a volumetric rendering model equivalent to the image formation model of a Gaussian Splatting representation—*under the additional assumption that Gaussians properly "wrap" the scene*. Specifically, we define the following oriented attenuation:

Definition 1 (Oriented Gaussian). *We assign an oriented normal vector parameter $n_i \in \mathcal{S}^2$ to each Gaussian G_i, and define its oriented attenuation coefficient as*

$$\bar{\sigma}_i(\boldsymbol{x}, \boldsymbol{w}) = \mathbb{1}_{n_i^T(\boldsymbol{x}-\mu_i)\geq 0} |\boldsymbol{w} \cdot \nabla \log(1 - G_i(\boldsymbol{x}))| . \tag{3}$$

We detail in the appendix—both theoretically and qualitatively—how this formulation not only satisfies the geometric properties required for applying OaV, but is also a valid approximation of the Gaussian Splatting image formation

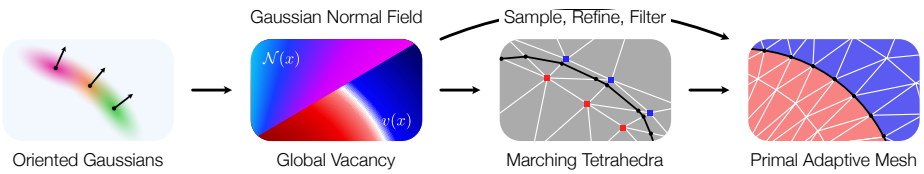


Fig. 3: Wrapping the surface with Oriented Gaussians. Our Gaussian Wrapping setting allows to evaluate the normal and vacancy fields of the scene. We first leverage the vacancy on our pivot-based marching tetrahedra meshing approach. We then leverage both the vacancy and normal fields in our Primal Adaptive Meshing.

model under specific assumptions. This attenuation-based model constitutes the basis of our approach, and allows us to derive the method below.

While Geometry Field Splatting [23] leverages OaV to define a geometric field out of particles, it applies its analysis to Gaussian surfels only and relies on TSDF for mesh extraction, limiting its capability to recover details and scale to full scenes with background geometry. Geometry-Grounded Gaussian Splatting [57] adopts a setting similar to our method, applying Equation 1 to derive a continuous transmittance and improve depth estimation. However, it does not guarantee the multi-view consistency of the depth, which is essential for a well-defined occupancy field as illustrated in Section 5. In contrast, we restore reciprocity while enforcing accurate and multiview-consistent geometry by *orienting* each Gaussian with a normal vector. We provide in the appendix an illustration of how the non-oriented attenuation from [57] fails to produce multi-view consistent depth maps and complete surfaces.

In the following section, we expose our theoretical results and describe how our formulation can be used for improving surface extraction from Gaussian Splatting representations. **Please refer to the appendix for extensive proofs and derivations motivating our results.**

4 Method

Method Overview: Our method operates as follows: first, as mentioned above, we endow each Gaussian with a learnable oriented normal parameter (note that these are *the only* additional learnable parameters of our framework), and introduce the corresponding attenuation coefficient, as in Definition 1. Based on this construction and thanks to the Objects as Volumes formalism [39], we derive closed-form expressions for both the normal and occupancy fields for arbitrary locations in space, without any additional learnable parameters. Second, we propose a training strategy, which learns the oriented normals through an image-based consistency loss, using an adapted rasterizer and coupled with a new normal-aware densification strategy. Lastly, we exploit our normal field and occupancy estimates in a novel dedicated mesh-extraction approach, which both respects the underlying geometry, while being adaptive to details. For a pipeline figure presenting our method, please see Fig. 3.

4.1 Gaussian Wrapping

Our core intuition is simple: for accurate surface reconstruction, Gaussians must form a sealed boundary around the object. We achieve this by endowing each Gaussian with a learnable oriented normal n_i and enforcing a *wrapping* behavior: visible Gaussians near the surface should point outward, away from the occupied region and toward the camera. This wrapping creates a continuous shell of oriented primitives even around extremely thin structures—such as bicycle spokes—for which, standard methods only recover the *appearance* with a few unoriented primitives, but cannot recover the geometry.

More fundamentally, we show in the appendix that when Gaussians properly wrap the scene, the oriented attenuation of Definition 1 becomes a valid approximation of the Gaussian Splatting image formation model, and the *Objects as Volumes* framework [39] directly yields an explicit vacancy field v.

We promote wrapping in the scene through two complementary mechanisms.

Normal alignment loss. Rendering depth and normals n_i with the Splatting rasterizer [55, 56] yields, respectively, an estimate of the surface depth and the expected orientation of Gaussians near the surface [39]. As a result, we enforce the alignment between the normals n_i and the surface during training via:

$$\mathcal{L}_{\mathrm{N}} = \sum_{p} 1 - N(p) \cdot \nabla D(p) \,, \tag{4}$$

where $N(p)$ is the rendered oriented normal at pixel p and $\nabla D(p)$ the image-space gradient of the rendered depth.

Specifically, we modify the differentiable CUDA rasterizer introduced in [57] to match our framework, and render depth as the exact 0.5-isosurface of our geometric field.

Densification. Gaps appear in the wrapping shell where Gaussians fail to cover all sides of a surface, causing $\mathcal{L}_{\mathrm{N}}(p)$ to rise locally. This provides a natural densification criterion: every K iterations, we compute errors $\mathcal{L}_{\mathrm{N}}(p)$ across all training views and propagate them to individual Gaussians via their blending weights. High-error Gaussians are cloned with flipped normals, closing holes and reinforcing the shell. We illustrate this process in the appendix.

Loss Details. During optimization, we combine our novel normal alignment objective $\mathcal{L}_{\mathrm{N}}$ in conjunction with the following set of losses: The standard photometric [25] loss $\mathcal{L}_{\mathrm{RGB}}$, depth-normal consistency $\mathcal{L}_{\mathrm{DN}}$ [55,56] and the multi-view loss $\mathcal{L}_{\mathrm{MV}} = \lambda_{\mathrm{pc}}\mathcal{L}_{\mathrm{pc}} + \lambda_{\mathrm{gc}}\mathcal{L}_{\mathrm{gc}}$, where the terms control the strength of the photometric and geometric consistency as defined in [10, 57]. These have the following weights $\lambda_{\mathrm{DN}} = 0.05, \lambda_{\mathrm{N}} = 0.05, \lambda_{\mathrm{pc}} = 0.6, \lambda_{\mathrm{gc}} = 0.02$. This results in the total loss

$$\mathcal{L} = \mathcal{L}_{\mathrm{RGB}} + \lambda_{\mathrm{DN}}\mathcal{L}_{\mathrm{DN}} + \lambda_{\mathrm{N}}\mathcal{L}_{\mathrm{N}} + \mathcal{L}_{\mathrm{MV}}$$

4.2 Gaussian Vector and Normal Fields

Assuming Gaussians properly wrap the scene, applying Eq. 1 from *Objects as Volumes* [39] to our reciprocal attenuation (Eq. 3) yields our main result.

Proposition 1 (Gaussian Vector and Normal Fields). *We define the Gaussian vector field* $V\colon \mathbb{R}^3 \to \mathbb{R}^3$ *of* N *oriented Gaussians and its normalization* $\mathcal{N}\colon \mathbb{R}^3 \to \mathcal{S}^2$ *as:*

$$V(x) := \nabla \log v(x) = \sum_{i=1}^{N} \mathbb{1}_{n_i^T(x-\mu_i)\geq 0} \nabla \log(1 - G_i(x)) \tag{5}$$

$$\mathcal{N}(x) := \frac{V(x)}{\|V(x)\|} \tag{6}$$

$\mathcal{N}(x)$ *is well-defined and coincides with the true normal field of the expected stochastic surface of* M *in a neighborhood of the surface.*

These closed-form expressions link Gaussian parameters directly to surface geometry and form the basis of our mesh extraction pipeline (Sec. 4.3). Note that we can query these quantities explicitly by querying a neighborhood of Gaussians around the point x. Proofs are deferred to the appendix.

4.3 Mesh Extraction

We now leverage the vacancy v as an implicit function for mesh extraction. While direct integration of V (Eq. 5) along any camera ray is possible, Gaussians that do not contribute to rendering can remain hidden inside the geometry after optimization, violating the wrapping assumption and producing artifacts if intersected along a ray. In practice, we compute the vacancy by iterating over the set of all training camera rays $\mathcal{T}_c$ with

$$v(\mathbf{x}) = \max_{(\mathbf{o},\mathbf{w})\in\mathcal{T}_c} \left\{ \prod_{i=1}^{N} \left(1 - \bar{G}^{(i)}_{\mathbf{o},\mathbf{w}}(t)\right) : \mathbf{x} = \mathbf{o} + t\mathbf{w}, t > 0 \right\}, \tag{7}$$

where $\bar{G}^{(i)}_{\mathbf{o},\mathbf{w}}(t) = G_i(\mathbf{o} + \min(t, t^{G_i}_{\mathbf{o},\mathbf{w}})\mathbf{w})$; and $t^{G_i}_{\mathbf{o},\mathbf{w}} := \arg\max_{t\geq 0} G_i(\mathbf{o} + t\mathbf{w})$. Intuitively, this computation corresponds to finding the "most" unobstructed ray reaching point x. This result *formalizes the opacity-field meshing heuristics* empirically adopted in recent works [42,55,57]. Strictly speaking, this expression is a lower bound on the true vacancy (see appendix for proof) and is inherently robust: since the products can only decrease along a ray, floating Gaussians hidden inside the geometry cannot inflate the vacancy estimate.

Pivot-Based Marching Tetrahedra. Under our oriented Gaussian assumption, the surface intersects each Gaussian between its center and the low-density side, parallel to the oriented plane. We therefore spawn two *Delaunay pivots* per Gaussian: the center μ_i and $\mu_i + 3s_i n_i$, where $s_i = \|S_i R_i^T n_i\|_2$ is the ellipsoid scaling

along n_i. Occupancy values at each pivot are obtained using the vacancy lower bound (Eq. 7); we then apply Marching Tetrahedra on the resulting Delaunay triangulation and refine vertices to the 0.5-isosurface via binary search. This approach is conceptually similar to GOF [55] and GGGS [57], yet contrasts with the 9 pivots per Gaussian required by prior works [18, 55–57], yielding substantially lighter, fully watertight meshes without sacrificing surface fidelity.

Primal Adaptive Meshing. Although simple and efficient, the pivot-based formulation instantiates vertices directly from Gaussian primitives rather than from the underlying scene geometry. To leverage our global vacancy field v and further improve mesh generation, we introduce an adaptive meshing framework inspired by primal Delaunay-based reconstruction methods [1, 14, 37, 46–48, 58]. The pipeline proceeds in four stages:

1. **Vertices Initialization.** We initialize our vertices by sampling the marching tetrahedra mesh faces, weighting each face inversely proportional to its distance from the nearest training camera.
2. **Isosurface Refinement.** Vertices are projected onto the 0.5-isosurface of v via an iterative Newton update:

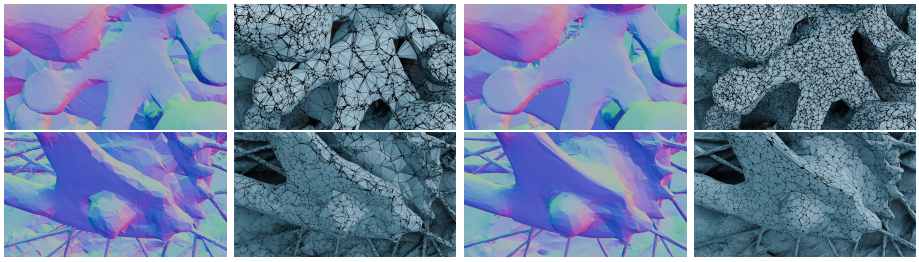
$$x_{i+1} = x_i + \frac{1}{2}(0.5 - v(x_i))\mathcal{N}(x_i), \tag{8}$$

3. **Filtering.** Vertices x with $|0.5 - v(x)| > \epsilon$ are considered outliers and removed. Stages 1–3 iterate until no points are removed.
4. **Delaunay.** We compute the Delaunay tetrahedralization of the remaining vertices and classify each tetrahedron as inside or outside, following [58], based on the vacancy value of randomly sampled interior points.
5. **Mesh extraction.** The final surface mesh is obtained by extracting triangle faces that separate inside and outside tetrahedra (see Fig. 3).

This approach effectively decouples mesh resolution from Gaussian distribution, enabling extremely fine meshing of restricted segments of the scene (Fig. 4). Although global distance-based metrics remain largely unchanged —being inherently insensitive to high-frequency detail— the resulting meshes exhibit substantially improved smoothness and reduced discretization artifacts. Details for the isosurface refinement stage can be found in the appendix.

5 Experiments

In this section, we evaluate and ablate Gaussian Wrapping on standard benchmarks. We expose a fundamental bias in the standard evaluation protocol for Gaussian-based surface reconstruction, and propose two complementary evaluations in Sec. 5.2: *Uniform Sampling* and *Virtual Scanning*. We refer the reader to the appendix for all of our implementation details.



(a) Pivot-based Marching Tetrahedra **(b)** Primal Adaptive Mesh

Fig. 4: Primal Adaptive Meshing on restricted scene segments. Close-up comparisons between our standard Pivot-based MTet extraction (a) and the Primal Adaptive Mesh (b) applied to the same segment on the bicycle and bonsai scenes of the MipNeRF 360 dataset. The Primal Adaptive Mesh produces topologically clean surfaces that faithfully reflect the underlying Gaussian isosurface.

5.1 Experimental Setup

Datasets & Evaluations. We perform quantitative and qualitative evaluations on the widely used **DTU** [22] and **Tanks and Temples (T&T)** [27] datasets. Furthermore, we leverage the Mesh-Based Rendering (MBR) evaluation introduced in MILo [18] to evaluate the mesh completeness.

Baselines. We compare our method against state-of-the-art explicit surface reconstruction techniques. We separate these into two categories: Methods capable of full scene reconstruction (GOF [55], SOF [42], MILo [18], RaDe-GS [56], GGGS [57]), and methods focusing on foreground only (PGSR [10], 2DGS [21]).

Metrics. We report Chamfer Distance (CD) on the DTU dataset; F1-score on the T&T dataset; and image metrics for NVS on the Mip-NeRF 360 dataset. Moreover, we present image metrics for the MBR evaluation on T&T and Mip-NeRF 360 datasets. For *Uniform Sampling*, *Virtual Scanning*, and *Mesh-Based Rendering*, we run all baseline methods ourselves under the exact same protocol.

5.2 Revisiting Surface Evaluation for Gaussian Splatting

The standard evaluation practice on T&T and structured scan datasets is fundamentally flawed for two reasons: (1) the conventional approach builds the predicted surface point cloud from the vertices and face centers of the extracted mesh, biasing metrics toward denser tessellations;

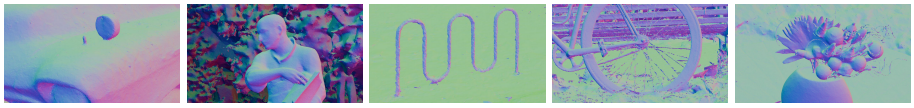
and (2) ground data (GT) acquisition, relying on laser scan, unfairly penalizes methods that correctly reconstruct occluded geometry and surfaces visible at a grazing angle. To resolve this, we introduce two robust evaluation strategies (Tab. 1).

Uniform Sampling Evaluation. To eliminate vertex-density bias, we uniformly sample a fixed number of points from the reconstructed mesh surface within the GT crop volume, ensuring a strictly fair geometric comparison.

Ground Truth Images

(b) PGSR

(a) GGGS



(c) MILo

(d) Ours

Fig. 5: Qualitative comparison on T&T and MipNerf360. Our method recovers significantly finer surface detail and thin structures compared to baselines. Close-up insets highlight regions where competing methods erode geometry or fail to close holes.

Virtual Scanning Evaluation. Uniform sampling alone does not take into account incomplete meshes that mimic GT bias. We thus propose to simulate the original data acquisition process by rendering depth maps of the reconstructed meshes from the input camera poses and back-projecting them into the GT crop volume.

Legacy Evaluation. For backward compatibility, we report the standard vertex-based metric. However, this metric is highly sensitive to tessellation density (e.g., extracting our meshes with 9 pivots instead of 2 artificially inflates scores). Please see the appendix for a detailed discussion of these flaws.

5.3 Main Results

Geometry on DTU. We report Chamfer Distance across 15 scenes of the DTU dataset in the appendix. Our method yields highly competitive results, outperforming most prior methods and achieving scores comparable to GGGS [57]. Importantly, our method does not exhibit the surface erosion artifacts observed

Table 1: Quantitative comparison on the Tanks & Temples Dataset [27]. F1 scores under two complementary evaluation protocols. *Uniform Sampling*: points are uniformly sampled from the mesh surface, eliminating vertex-count bias. *Virtual Scanning*: depth maps are back-projected from the original camera viewpoints, mirroring the ground truth acquisition process. We report average training time.

	Foreground only			Full scene extraction					
	2DGS	PGSR	**Ours (PAM)**	GOF	SOF	RaDe-GS	MILo	GGGS	**Ours (2p)**
Uniform Sampling Evaluation									
Barn	0.48	0.65	0.63	0.40	0.45	0.42	0.40	0.57	0.64
Caterpillar	0.24	0.45	0.38	0.23	0.20	0.22	0.24	0.41	0.41
Courthouse	0.13	0.24	0.07	0.10	0.07	0.12	0.11	0.10	0.19
Ignatius	0.51	0.81	0.76	0.55	0.68	0.66	0.72	0.79	0.77
Meetingroom	0.18	0.33	0.28	0.19	0.15	0.22	0.14	0.36	0.37
Truck	0.46	0.63	0.46	0.41	0.39	0.45	0.43	0.50	0.50
Mean	0.33	0.52	0.43	0.31	0.32	0.35	0.34	0.45	0.48
Virtual Scanning Evaluation									
Barn	0.51	0.65	0.67	0.49	0.49	0.47	0.52	0.62	0.67
Caterpillar	0.31	0.50	0.50	0.29	0.31	0.30	0.33	0.52	0.50
Courthouse	0.10	0.18	0.16	0.11	0.06	0.11	0.12	0.16	0.18
Ignatius	0.61	0.82	0.78	0.63	0.63	0.66	0.74	0.80	0.78
Meetingroom	0.19	0.35	0.41	0.24	0.23	0.25	0.23	0.42	0.41
Truck	0.54	0.66	0.63	0.50	0.48	0.57	0.56	0.65	0.63
Mean	0.38	0.53	0.53	0.38	0.37	0.39	0.42	0.53	0.53
Times	12 m	45 m	27 m	69 m	17 m	12 m	150 m	32 m	27 m

in GGGS, as illustrated in Fig. 5. We note that we outperform foreground only methods such as 2DGS and PGSR on this dataset.

Geometry on T&T. We evaluate on the T&T dataset under three complementary protocols, all reported in Tab. 1.

- *Uniform Sampling Evaluation.*
  Our method sets a new state of the art under this unbiased protocol among the full scene extraction methods. PGSR [10] remains an apparent outlier: its use of TSDF fusion and depth filtering yields non-watertight meshes whose holes coincide with those of the ground truth scan, an artifact of the acquisition process rather than genuine geometric quality. We document this in the appendix. We note that Ours (PAM)'s performance takes a toll on this metric. Uniformly sampling points to reconstruct scenes such as Courthouse or Meetingroom is far from optimal. We highlight that the advantage of this meshing procedure is the meshing of regions of interest at arbitrary resolution (Fig. 4).
- *Virtual Scanning Evaluation.* Simulating the acquisition bias of the ground truth via our virtual scanning procedure, we set a new state of the art alongside GGGS, and see that PGSR has no longer an advantage when this bias is taken into account.

Mesh-Based Rendering on T&T and Mip-NeRF 360. To complement our geometric evaluation, we report Mesh-Based Rendering (MBR) metrics [18], which assesses background reconstruction quality and serve as a proxy for mesh completeness. Results are shown in Tab. 2. Despite lacking the dedicated rendering prior of MILo [18], our method performs competitively across all datasets

and sets a new state of the art on Mip-NeRF 360. This evaluation also exposes a critical weakness of PGSR: its depth-filtering strategy leaves substantial holes in background geometry, resulting in severely degraded rendering quality—confirming that its strong geometric scores are an artifact of the evaluation protocol rather than true reconstruction quality.

Novel View Synthesis on Mip-NeRF 360. Although NVS is not the primary objective of our method, we achieve competitive performance on Mip-NeRF 360 [5] and set a new state of the art for outdoor scenes, reflecting the effectiveness of our representation in challenging, unbounded settings. Results are reported in the appendix.

Table 2: Mesh-Based Novel View synthesis metrics on real-world datasets. We report PSNR, SSIM, and LPIPS metrics, as well as the total number of Gaussians and vertices (in millions). Our method demonstrates superior performance on these challenging unbounded scenes.

	MipNeRF 360					Tanks & Temples				
	PSNR ↑	SSIM ↑	LPIPS ↓	#Gaussians ↓	#Verts ↓	PSNR ↑	SSIM ↑	LPIPS ↓	#Gaussians ↓	#Verts ↓
Foreground only										
2DGS	15.36	0.4987	0.4749	1.88	4.31	14.23	0.5697	0.4854	0.98	16.39
PGSR	15.63	0.5834	0.4509	3.16	22.78	14.55	0.5956	0.4819	1.47	11.79
Full scene extraction										
GOF	24.25	0.7017	0.3454	2.99	32.80	20.10	0.6475	0.4073	1.25	11.63
RaDe-GS	24.84	0.7291	0.3128	2.91	30.85	20.70	0.6767	0.3876	1.18	10.06
MILo	25.075	0.7339	0.3096	0.46	6.73	21.198	0.6908	0.3782	0.28	4.36
GGGS	24.55	0.7386	0.2986	4.08	43.46	20.70	0.6912	0.3734	1.73	21.81
Ours	25.10	0.752	0.289	4.0	11.79	20.98	0.7004	0.3674	2.07	5.80

Qualitative Comparisons. Fig. 5 presents visual comparisons against all baselines. Gaussian Wrapping consistently produces surfaces with sharper fine details, fewer topological artifacts, and better coverage of thin structures such as bicycle spokes, corroborating the quantitative findings.

Conclusion. Our results demonstrate that Gaussian Wrapping achieves robust state-of-the-art performance across all evaluation settings. While methods such as PGSR appear competitive under specific protocols by exploiting biases in GT data, the TSDF extraction limits their quality. Our method performs consistently across both legacy and newly proposed metrics, and is further validated by complementary rendering-based and qualitative evidence. This consistency confirms our improvements reflect genuine advances in surface reconstruction.

5.4 Ablation Studies

All ablation studies are conducted on T&T, reporting F1 scores and MBR metrics (SSIM, PSNR, LPIPS). More details are provided in appendix.

Normal Alignment Losses. Tab. 3 reports results with and without our normal alignment losses. F1 scores remain stagnant—our rasterizer already places Gaussians near the isosurface—but MBR metrics improve substantially: our losses align Gaussians with the surface, enabling clean mesh extraction with only 2

Table 3: Ablation studies on Tanks & Temples. *Top*: Mean scores across all scenes for our component ablation, reporting F1-score under both evaluation protocols and Mesh-Based Rendering (MBR) metrics. *Bottom*: Per-scene Virtual Scan F1-score demonstrating that our contributions generalize as a plug-in regularizer to RaDe-GS [56]. The normal alignment loss and densification procedure constitute the Gaussian Wrapping approach.

Component Ablation (Mean across scenes)

	F1-Score ↑		MBR Metrics		
Configuration	**Virt. Scan**	**Mesh Qual.**	**PSNR ↑**	**SSIM ↑**	**LPIPS ↓**
Baseline	0.53	0.48	20.74	0.6958	0.3718
+ Normal Alignment Loss	0.52	0.48	20.78	0.6986	0.3683
+ Normal Alignment Loss + Densification	0.53	0.48	20.98	0.7004	0.3674

Gaussian Wrapping Generalization to RaDe-GS (Virtual Scan F1-Score)

Configuration	Barn	Caterpillar	Courthouse	Ignatius	Meetingroom	Truck	**Mean**
RaDe-GS	0.47	0.30	0.11	0.66	0.25	0.57	0.39
RaDe-GS + Gaussian Wrapping	0.63	0.46	0.13	0.71	0.34	0.60	0.48

pivots. Without them, fine details such as thin structures and sharp edges are lost. This also serves as an illustration that the T&T dataset and ground truth is best to measure coarse geometry alignment, and fails to measure detail loyalty.

Generalizability of Gaussian Wrapping. To confirm that the Gaussian Wrapping approach is architecture-agnostic, we plug it into RaDe-GS [56]. The consistent improvement across all T&T scenes (Tab. 3; and figure in the appendix) demonstrates that our method operates as an effective drop-in regularizer well beyond our specific pipeline. Note that the difference between the RaDe-GS + Gaussian Wrapping combination and Ours in previous tables is the use of our depth rasterization, which precisely queries the 0.5-isosurface via binary search.

6 Conclusion

We presented Gaussian Wrapping, a principled method grounded on the OaV framework. Our method reconstructs complete scenes including extremely thin structures where prior methods fail, without surface erosion artifacts, and at a fraction of the mesh weight of competitors. We further expose fundamental biases in standard evaluation protocols and propose two alternatives. Finally, Primal Adaptive Meshing enables extraction of meshes with better topology and controllable resolution.

Limitations and Future Work. Our Primal Adaptive Meshing currently relies on uniform sampling of an initial MTet mesh, which can be suboptimal for highly detailed scenes. Developing more principled sampling strategies—for instance, guided by the Gaussian Vector Field or local surface curvature—would extend the approach to finer and more intricate geometry. Moreover, our attenuation-based formulation is not specific to 3DGS: extending it to more accurate volumetric rendering models such as EVER [32] is a natural and promising direction.

7 Acknowledgements

We are grateful to George Drettakis for insightful discussions. Parts of this work were supported by the ERC Consolidator Grant "VEGA" (No. 101087347), the ERC Advanced Grant "explorer" (No. 101097259), as well as gifts from Ansys Inc., and Adobe Research.

References

1. Amenta, N., Bern, M., Kamvysselis, M.: A new Voronoi-based surface reconstruction algorithm. In: Proceedings of the 25th annual conference on Computer graphics and interactive techniques - SIGGRAPH '98. pp. 415–421. ACM Press (1998)
2. Amenta, N., Choi, S., Kolluri, R.K.: The power crust. In: Proceedings of the sixth ACM symposium on Solid modeling and applications. pp. 249–266. SMA '01, Association for Computing Machinery (2001)
3. Aurenhammer, F.: Voronoi diagrams—a survey of a fundamental geometric data structure. ACM computing surveys (CSUR) **23**(3), 345–405 (1991)
4. Barron, J.T., Mildenhall, B., Tancik, M., Hedman, P., Martin Brualla, R., Srinivasan, P.P.: Mip-NeRF: A Multiscale Representation for Anti-Aliasing Neural Radiance Fields. In: Int. Conf. Comput. Vis. pp. 5855–5864 (2021)
5. Barron, J.T., Mildenhall, B., Verbin, D., Srinivasan, P.P., Hedman, P.: Mip-NeRF 360: Unbounded Anti-Aliased Neural Radiance Fields. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 5460–5469. IEEE (2022)
6. Binninger, A., Wiersma, R., Herholz, P., Sorkine-Hornung, O.: TetWeave: Isosurface Extraction using On-The-Fly Delaunay Tetrahedral Grids for Gradient-Based Mesh Optimization. ACM Trans. Graph. **44**(4), 1–19 (2025)
7. Blanc, H., Deschaud, J.E., Paljic, A.: Raygauss: Volumetric Gaussian-based Ray Casting for Photorealistic Novel View Synthesis. In: 2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV). pp. 1808–1817. IEEE (2025)
8. Celarek, A., Kopanas, G., Drettakis, G., Wimmer, M., Kerbl, B.: Does 3d gaussian splatting need accurate volumetric rendering? In: Computer Graphics Forum. vol. 44, p. e70032. Wiley Online Library (2025)
9. Chen, A., Xu, Z., Geiger, A., Yu, J., Su, H.: TensoRF: Tensorial Radiance Fields. In: Eur. Conf. Comput. Vis. pp. 333–350 (2022)
10. Chen, D., Li, H., Ye, W., Wang, Y., Xie, W., Zhai, S., Wang, N., Liu, H., Bao, H., Zhang, G.: PGSR: Planar-based Gaussian Splatting for Efficient and High-Fidelity Surface Reconstruction. IEEE Transactions on Visualization and Computer Graphics **31**(9), 6100–6111 (2024)
11. Chen, H., Li, C., Lee, G.H.: NeuSG: Neural Implicit Surface Reconstruction with 3D Gaussian Splatting Guidance. arXiv preprint arXiv:2312.00846 (2023)
12. Chen, H., Wei, F., Li, C., Huang, T., Wang, Y., Lee, G.H.: VCR-GauS: View Consistent Depth-Normal Regularizer for Gaussian Surface Reconstruction. In: Adv. Neural Inform. Process. Syst. (2024)
13. Dey, T.K., Goswami, S.: Tight cocone: a water-tight surface reconstructor. In: Proceedings of the eighth ACM symposium on Solid modeling and applications. pp. 127–134. ACM (2003)
14. Dey, T.K., Levine, J.A.: Delaunay meshing of isosurfaces. The Visual Computer **24**(6), 411–422 (2008)
15. Fridovich-Keil, S., Yu, A., Tancik, M., Chen, Q., Recht, B., Kanazawa, A.: Plenoxels: Radiance Fields without Neural Networks. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 5501–5510 (2022)
16. Gomez, D., Philip, J., Kaiser, A., Michel, É.: RRM: Relightable assets using Radiance guided Material extraction. In: Computer Graphics International Conference. pp. 17–41. Springer (2024)
17. Govindarajan, S., Rebain, D., Yi, K.M., Tagliasacchi, A.: Radiant Foam: Real-Time Differentiable Ray Tracing. In: Int. Conf. Comput. Vis. (2025)

18. Guédon, A., Gomez, D., Maruani, N., Gong, B., Drettakis, G., Ovsjanikov, M.: MILo: Mesh-In-the-Loop Gaussian Splatting for Detailed and Efficient Surface Reconstruction. ACM Trans. Graph. **44**(6), 1–15 (2025)
19. Guédon, A., Lepetit, V.: SuGaR: Surface-Aligned Gaussian Splatting for Efficient 3D Mesh Reconstruction and High-Quality Mesh Rendering. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 5354–5363 (2024)
20. Held, J., Son, S., Vandeghen, R., Rebain, D., Gadelha, M., Zhou, Y., Cioppa, A., Lin, M.C., Van Droogenbroeck, M., Tagliasacchi, A.: MeshSplatting: Differentiable Rendering with Opaque Meshes. arXiv preprint arXiv:2512.06818 (2025)
21. Huang, B., Yu, Z., Chen, A., Geiger, A., Gao, S.: 2D Gaussian Splatting for Geometrically Accurate Radiance Fields. In: ACM SIGGRAPH 2024 conference papers. pp. 1–11 (2024)
22. Jensen, R., Dahl, A., Vogiatzis, G., Tola, E., Aanæs, H.: Large scale multi-view stereopsis evaluation. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 406–413 (2014)
23. Jiang, K., Sivaram, V., Peng, C., Ramamoorthi, R.: Geometry Field Splatting with Gaussian Surfels. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 5752–5762 (2025)
24. Jin, H., Liu, I., Xu, P., Zhang, X., Han, S., Bi, S., Zhou, X., Xu, Z., Su, H.: TensoIR: Tensorial Inverse Rendering. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 165–174 (2023)
25. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3D Gaussian Splatting for Real-Time Radiance Field Rendering. ACM Trans. Graph. **42**(4) (July 2023)
26. Kheradmand, S., Rebain, D., Sharma, G., Sun, W., Tseng, Y.C., Isack, H., Kar, A., Tagliasacchi, A., Yi, K.M.: 3D gaussian splatting as markov chain monte carlo. Adv. Neural Inform. Process. Syst. **37**, 80965–80986 (2024)
27. Knapitsch, A., Park, J., Zhou, Q.Y., Koltun, V.: Tanks and temples: Benchmarking large-scale scene reconstruction. ACM Transactions on Graphics **36**(4) (2017)
28. Li, S., Liu, Y.S., Han, Z.: Gaussianudf: Inferring unsigned distance functions through 3d gaussian splatting. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 27113–27123 (2025)
29. Li, Z., Müller, T., Evans, A., Taylor, R.H., Unberath, M., Liu, M.Y., Lin, C.H.: Neuralangelo: High-fidelity neural surface reconstruction. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 8456–8465 (2023)
30. Liang, Z., Zhang, Q., Feng, Y., Shan, Y., Jia, K.: GS-IR: 3D Gaussian Splatting for Inverse Rendering. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 21644–21653 (2024)
31. Lyu, X., Sun, Y.T., Huang, Y.H., Wu, X., Yang, Z., Chen, Y., Pang, J., Qi, X.: 3DGSR: Implicit Surface Reconstruction with 3D Gaussian Splatting. ACM Trans. Graph. **43**(6), 1–12 (2024)
32. Mai, A., Hedman, P., Kopanas, G., Verbin, D., Futschik, D., Xu, Q., Kuester, F., Barron, J.T., Zhang, Y.: Ever: Exact volumetric ellipsoid rendering for real-time view synthesis. In: Int. Conf. Comput. Vis. pp. 4930–4939 (2025)
33. Mai, A., Hedstrom, T., Kopanas, G., Kontkanen, J., Kuester, F., Barron, J.T.: Radiance Meshes for Volumetric Reconstruction (2025)
34. Mai, A., Verbin, D., Kuester, F., Fridovich-Keil, S.: Neural Microfacet Fields for Inverse Rendering (2023)
35. Mallick, S.S., Goel, R., Kerbl, B., Steinberger, M., Carrasco, F.V., De La Torre, F.: Taming 3DGS: High-quality Radiance Fields with Limited Resources. In: SIGGRAPH Asia Conference Proceedings. pp. 1–11 (2024)

36. Maruani, N., Klokov, R., Ovsjanikov, M., Alliez, P., Desbrun, M.: VoroMesh: Learning Watertight Surface Meshes with Voronoi Diagrams. In: Int. Conf. Comput. Vis. pp. 14565–14574 (2023)
37. Maruani, N., Ovsjanikov, M., Alliez, P., Desbrun, M.: PoNQ: A Neural QEM-Based Mesh Representation. In: 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 3647–3657 (Jun 2024)
38. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In: Eur. Conf. Comput. Vis. pp. 99–106 (2020)
39. Miller, B., Chen, H., Lai, A., Gkioulekas, I.: Objects as Volumes: A Stochastic Geometry View of Opaque Solids. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 87–97 (June 2024)
40. Müller, T., Evans, A., Schied, C., Keller, A.: Instant Neural Graphics Primitives with a Multiresolution Hash Encoding. ACM Trans. Graph. **41**(4), 102:1–102:15 (Jul 2022)
41. Munkberg, J., Hasselgren, J., Shen, T., Gao, J., Chen, W., Evans, A., Müller, T., Fidler, S.: Extracting Triangular 3D Models, Materials, and Lighting From Images. In: IEEE Conf. Comput. Vis. Pattern Recog. (2022)
42. Radl, L., Windisch, F., Deixelberger, T., Hladky, J., Steiner, M., Schmalstieg, D., Steinberger, M.: SOF: Sorted Opacity Fields for Fast Unbounded Surface Rconstruction. In: SIGGRAPH Asia Conference Proceedings (2025)
43. Reiser, C., Garbin, S., Srinivasan, P.P., Verbin, D., Szeliski, R., Mildenhall, B., Barron, J.T., Hedman, P., Geiger, A.: Binary Opacity Grids: Capturing Fine Geometric Detail for Mesh-Based View Synthesis. SIGGRAPH Conference Proceedings (2024)
44. Ren, H., Yan, Q., Lu, M., Lu, R., Zhu, Z.: 2DGS-R: Revisiting the Normal Consistency Regularization in 2D Gaussian Splatting. arXiv preprint arXiv:2510.16837 (2025)
45. Shen, T., Munkberg, J., Hasselgren, J., Yin, K., Wang, Z., Chen, W., Gojcic, Z., Fidler, S., Sharp, N., Gao, J.: Flexible Isosurface Extraction for Gradient-Based Mesh Optimization. ACM Trans. Graph. **42**(4) (2023)
46. Son, S., Gadelha, M., Zhou, Y., Fisher, M., Xu, Z., Qiao, Y.L., Lin, M.C., Zhou, Y.: DMesh++: An Efficient Differentiable Mesh for Complex Shapes. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 26590–26599 (2025)
47. Son, S., Gadelha, M., Zhou, Y., Xu, Z., Lin, M., Zhou, Y.: DMesh: A Differentiable Mesh Representation. Advances in Neural Information Processing Systems **37**, 12035–12077 (2024)
48. Sulzer, R., Landrieu, L., Marlet, R., Vallet, B.: Scalable Surface Reconstruction with Delaunay-Graph Neural Networks. In: Computer Graphics Forum. vol. 40, pp. 157–167. Wiley Online Library (2021)
49. Verbin, D., Hedman, P., Mildenhall, B., Zickler, T., Barron, J.T., Srinivasan, P.P.: Ref-NeRF: Structured View-Dependent Appearance for Neural Radiance Fields. IEEE Conf. Comput. Vis. Pattern Recog. (2022)
50. Wang, P., Liu, L., Liu, Y., Theobalt, C., Komura, T., Wang, W.: NeuS: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. In: Adv. Neural Inform. Process. Syst. (2021)
51. Yariv, L., Gu, J., Kasten, Y., Lipman, Y.: Volume rendering of neural implicit surfaces. In: Adv. Neural Inform. Process. Syst. (2021)

52. Yariv, L., Hedman, P., Reiser, C., Verbin, D., Srinivasan, P.P., Szeliski, R., Barron, J.T., Mildenhall, B.: BakedSDF: Meshing Neural SDFs for Real-Time View Synthesis. In: ACM SIGGRAPH 2023 conference proceedings. pp. 1–9 (2023)
53. Yu, M., Lu, T., Xu, L., Jiang, L., Xiangli, Y., Dai, B.: GSDF: 3DGS Meets SDF for Improved Rendering and Reconstruction. In: Adv. Neural Inform. Process. Syst. (2024)
54. Yu, Z., Chen, A., Huang, B., Sattler, T., Geiger, A.: Mip-Splatting: Alias-free 3D Gaussian Splatting. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 19447–19456 (2024)
55. Yu, Z., Sattler, T., Geiger, A.: Gaussian opacity fields: Efficient adaptive surface reconstruction in unbounded scenes. ACM Transactions on Graphics (ToG) **43**(6), 1–13 (2024)
56. Zhang, B., Fang, C., Shrestha, R., Liang, Y., Long, X., Tan, P.: RaDe-GS: Rasterizing Depth in Gaussian Splatting. ACM Trans. Graph. (2024)
57. Zhang, B., Jiang, C., Li, H., Shen, S., Tan, P.: Geometry-Grounded Gaussian Splatting. arXiv preprint arXiv:2601.17835 (2026)
58. Zhang, C., Wang, W., Li, X., Liao, X., Su, W., Tao, W.: High-Fidelity Lightweight Mesh Reconstruction from Point Clouds. In: 2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 11739–11748 (Jun 2025), iSSN: 2575-7075
59. Zhang, W., Liu, Y.S., Han, Z.: Neural Signed Distance Function Inference through Splatting 3D Gaussians Pulled on Zero-Level Set. In: Adv. Neural Inform. Process. Syst. (2024)
60. Zhang, Z., Huang, B., Jiang, H., Zhou, L., Xiang, X., Shen, S.: Quadratic Gaussian Splatting for Efficient and Detailed Surface Reconstruction. In: Int. Conf. Comput. Vis. (2025)

A Overview of the Appendix

This appendix serves three purposes: **(i)** provide proofs for the theoretical claims made in the main paper; **(ii)** provide the implementation details required to reproduce our results; and **(iii)** showcase additional results, including quantitative tables, qualitative figures, and an illustration of the evaluation bias we expose. Throughout the theory sections, key results are highlighted in blue boxes and their corresponding proofs in orange boxes.

Section B provides details on the theoretical results presented in the paper. We give further proof and motivation for our choice of attenuation and normal field. We also illustrate how the transmittance introduced in GGGS [57] leads to surface erosion.

Section C presents the derivation of the lower-bound for estimating vacancy (Sec. C), and the Newton projection step used in our Primal Adaptive Meshing procedure (Proposition 5).

Section D provides additional quantitative results (DTU, legacy evaluation on T&T, Novel View Synthesis on Mip-NeRF 360), examples of the evaluation bias discussed in the main paper with a concrete code snippet from the RaDe-GS repository (Fig. 13).

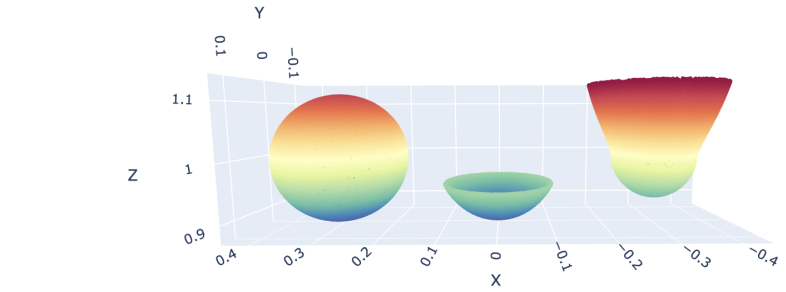
B Deriving an Equivalent Ray-Marching Formulation

Objects as Volumes (OaV) [39] establishes a strong theoretical framework exploring the relationship between surface extraction and volumetric rendering. Specifically, the results introduced in OaV only hold for volumetric rendering models relying on ray-marching through an attenuation field σ (also called *density* field), such as Neural Radiance Fields [38].

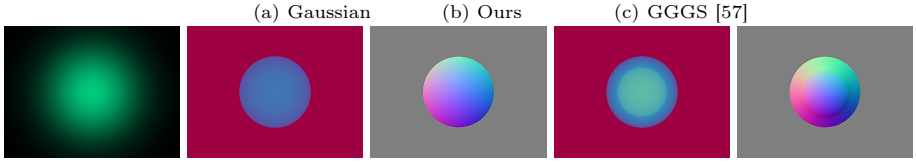
Even though 3D Gaussian Splatting relies on volumetric primitives, its image formation model is based on rasterization: volumetric particles are first projected onto the image plane as flat 2D Gaussians, then blended together to obtain pixel colors. This formulation is not strictly equivalent to volumetric ray-marching, and theoretical results from OaV cannot be directly applied to Gaussian Splatting.

While rasterization allows for incredibly fast rendering, it is less accurate than methods relying on ray-marching through attenuation. Still, in practice, it can be shown [8] that given enough Gaussians, the lack of accuracy has minor impact on the novel view synthesis quality of 3DGS representations.

The goal of this section is to propose, motivate and justify a volumetric ray-marching formulation equivalent to the image formation model of 3D Gaussian Splatting under the right set of assumptions. Building on this equivalence, we then apply the results of OaV to 3D Gaussian Splatting, enhancing surface extraction from Gaussians.



(a) Gaussian (b) Ours (c) GGGS [57]



(d) RGB render (e) Depth (Ours) (f) Normal (Ours) (g) Depth [57] (h) Normal [57]

Fig. 6: Comparison between our continuous attenuation model, and the one proposed in GGGS [57]. Top figure: The sphere illustrated on the left (a) represents the 0.5-isosurface of a single isotropic Gaussian. Given a camera located below the Gaussian and looking up, we represent the corresponding 0.5-isosurface of the transmittance for both our continuous attenuation model (b) and GGGS [57] (c). Our method produces unbiased, spherical isosurfaces aligned with the visible part of the actual Gaussian isosurface. On the contrary, GGGS produces non-multi-view consistent surfaces misaligned with the Gaussian isosurface, leading to erosion. **Bottom row** illustrates the actual RGB rendering, depth maps and normal maps obtained for both methods.

B.1 Deriving Attenuation

Let us start by deriving the attenuation in this formulation.

Lemma 1 (Transmittance of a single Gaussian.). *The image formation model of a Gaussian Splatting representation composed of a single Gaussian, which relies on alpha-blending with opacity, is equivalent to a ray-marching volumetric rendering model with the following continuous transmittance function: for any ray origin $\boldsymbol{o} \in \mathbb{R}^3$ in empty space, direction $\boldsymbol{w} \in \mathcal{S}^2$, and flight distance $t \in [0, +\infty)$:*

$$T_{\boldsymbol{o},\boldsymbol{w}}(t) = 1 - G(\boldsymbol{o} + \min(t, t^G_{\boldsymbol{o},\boldsymbol{w}})\boldsymbol{w}) \,, \tag{9}$$

where $t^G_{\boldsymbol{o},\boldsymbol{w}} := \arg\max_{t \geq 0}\{G(\boldsymbol{o} + t\boldsymbol{w})\}$. The corresponding attenuation coefficient is view-dependent and is given by:

$$\sigma(\boldsymbol{x}, \boldsymbol{w}) = \max\left(0, -\boldsymbol{w} \cdot \nabla \log(1 - G(\boldsymbol{x}))\right) \,. \tag{10}$$

Sketch of proof. Our model is uniquely determined by enforcing three conditions on the attenuation coefficient:

1. First, applying ray-marching (as in NeRF [38]) with the attenuation-based model should produce the same rendered image as Gaussian Splatting.
2. Second, the transmittance function should reach its maximum value at the point of maximum contribution along the ray $\mathbf{x} = \mathbf{o} + t^G_{\mathbf{o},\mathbf{w}}\mathbf{w}$.
3. Third, following the definition of transmittance from Objects as Volumes [39], we interpret transmittance as visibility probability along a ray. Therefore, we assume an isosurface of the transmittance function for a single Gaussian should be a portion of an ellipsoid centered at the Gaussian center.

We note that, transmittance having ellipsoidal isosurfaces is not a simplification, and uniquely determines the form of the attenuation coefficient in Eq. 18. Removing this assumption, another natural and simpler possible attenuation can be proposed: $\sigma(\mathbf{x}, \mathbf{w}) = \frac{1}{2} |\mathbf{w} \cdot \nabla \log(1 - G(\mathbf{x}))|$. This attenuation factor corresponds to the continuous transmittance introduced in the concurrent work [57]. It does not enforce the aforementioned geometric constraint on isosurfaces. Fig. 6 illustrates the difference between our transmittance and [57]: For a single Gaussian, we rendered the depth as the 0.5-isosurface of the transmittance, as proposed in [57]. The simpler attenuation from [57] produces non-spherical surfaces for a single Gaussian. This results in (a) non-multi-view consistent depth maps that are largely misaligned with the actual Gaussian; and (b) eroded surfaces. The qualitative comparison presented in the main paper and in Fig. 6 illustrates how this misalignment erodes the extracted surfaces.

Proof. We first derive a continuous transmittance expression for a single Gaussian by imposing consistency with alpha blending while following the reasoning of [39]. We then verify equivalence in the rendering result between alpha blending and the proposed density-based model.
Following [55, 56], the alpha-blending formula models the contribution of a single Gaussian to the transmittance as being constant after the point of maximum Gaussian contribution along the ray. Hence, for a scene composed of a single Gaussian G:

$$\forall t \geq t^G_{\mathbf{o},\mathbf{w}} \ , \ \mathrm{T}_{\mathbf{o},\mathbf{w}}(t) = \mathrm{T}_{\mathbf{o},\mathbf{w}}(t^G_{\mathbf{o},\mathbf{w}}) = 1 - G(\mathbf{o} + t^G_{\mathbf{o},\mathbf{w}}\mathbf{w}) \ . \tag{11}$$

We now determine how $\mathrm{T}_{\mathbf{o},\mathbf{w}}$ interpolates from its initial value to $\mathrm{T}_{\mathbf{o},\mathbf{w}}(t^G_{\mathbf{o},\mathbf{w}})$ before reaching the maximum-contribution point.
Following [39], we interpret transmittance as the visibility probability along a ray, *i.e.*, the probability of a point not being occluded by the scene along the ray. Therefore, for a fixed ray origin $\mathbf{o}$ and scalar $\lambda > 0$, the λ-isosurface of $(\mathbf{w}, t) \mapsto \mathrm{T}_{\mathbf{o},\mathbf{w}}(t)$ corresponds to the visible part of a candidate scene surface at decision level λ.
To simplify the analysis, we temporarily assume that the Gaussian G is isotropic. In this case, the scene surface is assumed to be spherical around μ, such that each isosurface of the transmittance function $(\mathbf{w}, t) \mapsto \mathrm{T}_{\mathbf{o},\mathbf{w}}(t)$ contains the visible part of a sphere centered at μ.

For any ray direction $\mathbf{w}$ and flight distance $t < t^G_{\mathbf{o},\mathbf{w}}$, the point $\mathbf{o} + t\mathbf{w}$ is located on the λ-isosurface of $(\mathbf{w}, s) \mapsto \mathrm{T}_{\mathbf{o},\mathbf{w}}(s)$ with $\lambda = \mathrm{T}_{\mathbf{o},\mathbf{w}}(t)$. There exists at least one direction $\mathbf{w}'$ and flight distance t' such that $\mathbf{o} + t'\mathbf{w}'$ is located on the same λ-isosurface and $\mathbf{o} + t'\mathbf{w}'$ is the point with maximum contribution along the ray with direction $\mathbf{w}'$.
It follows that $\mathrm{T}_{\mathbf{o},\mathbf{w}}(t) = T_{\mathbf{o},\mathbf{w}'}(t') = 1 - G(\mathbf{o} + t'\mathbf{w}')$. Since the Gaussian is isotropic and the spherical λ-isosurface is centered at μ, we have $||\mathbf{o} + t\mathbf{w} - \mu|| = ||\mathbf{o} + t'\mathbf{w}' - \mu||$, such that $G(\mathbf{o} + t\mathbf{w}) = G(\mathbf{o} + t'\mathbf{w}')$.
We conclude that, for any ray direction $\mathbf{w}$ and flight distance t, the transmittance satisfies

$$\mathrm{T}_{\mathbf{o},\mathbf{w}}(t) = \begin{cases} 1 - G(\mathbf{o} + t\mathbf{w})\,, & t < t^G_{\mathbf{o},\mathbf{w}} \\ 1 - G(\mathbf{o} + t^G_{\mathbf{o},\mathbf{w}}\mathbf{w})\,, & t \geq t^G_{\mathbf{o},\mathbf{w}} \end{cases}\,, \tag{12}$$

such that:

$$\mathrm{T}_{\mathbf{o},\mathbf{w}}(t) = 1 - G(\mathbf{o} + \min(t, t^G_{\mathbf{o},\mathbf{w}})\mathbf{w})\,. \tag{13}$$

Differentiating yields

$$-\frac{\mathrm{d}}{\mathrm{d}t}\log \mathrm{T}_{\mathbf{o},\mathbf{w}}(t) = \sigma(\mathbf{o} + t\mathbf{w}, \mathbf{w})\,, \tag{14}$$

with

$$\sigma(\mathbf{x}, \mathbf{w}) = \max\left(0, -\mathbf{w} \cdot \nabla \log(1 - G(\mathbf{x}))\right)\,. \tag{15}$$

The same derivation extends to anisotropic Gaussians by replacing spherical level sets with ellipsoidal ones centered at μ.
We now verify equivalence between alpha blending and the proposed density-based model by showing that both formulations produce the same rendered color. Let the Gaussian G be equipped with a directional color field $\mathbf{c} : \mathcal{S}^2 \to [0,1]^3$. The rendered color $C_{\mathbf{o},\mathbf{w}}$ along a ray from an empty point $\mathbf{o}$ in direction $\mathbf{w}$ under the density-based model is

$$\begin{aligned} C_{\mathbf{o},\mathbf{w}} &= \int_0^{+\infty} \sigma(\mathbf{o} + s\mathbf{w}, \mathbf{w})\mathrm{T}_{\mathbf{o},\mathbf{w}}(s)\mathbf{c}(\mathbf{w})\mathrm{d}s \\ &= \mathbf{c}(\mathbf{w}) \int_0^{+\infty} -\frac{\mathrm{d}}{\mathrm{d}t}\mathrm{T}_{\mathbf{o},\mathbf{w}}(s)\mathrm{d}s \\ &= \mathbf{c}(\mathbf{w})\left[\mathrm{T}_{\mathbf{o},\mathbf{w}}(0) - \mathrm{T}_{\mathbf{o},\mathbf{w}}(+\infty)\right] \\ &= \mathbf{c}(\mathbf{w})\left[G(\mathbf{o} + t^G_{\mathbf{o},\mathbf{w}}\mathbf{w}) - G(\mathbf{o})\right] \\ &= \mathbf{c}(\mathbf{w})G(\mathbf{o} + t^G_{\mathbf{o},\mathbf{w}}\mathbf{w})\,, \end{aligned} \tag{16}$$

where $G(\mathbf{o}) \simeq 0$ since $\mathbf{o}$ lies in empty space. The right-hand side matches the alpha-blending expression for a single Gaussian [55, 56], which proves equivalence.

□

(a) Gaussian Splatting rendering [56] **(b)** Ray-marching with attenuation **(c)** Ray-marching with oriented attenuation

Fig. 7: Equivalence between Gaussian Splatting rendering and our attenuation-based formulations. (a) Standard Gaussian Splatting rasterization. (b) Ray-marching our derived attenuation field from Equation 18, with corresponding error map. (c) Ray-marching the oriented attenuation after optimizing Gaussians with our oriented normal regularization 4. Both attenuation-based models produce visually similar results, confirming the formal equivalence established in Sec. B.1 and the approximation proposed in Sec. B.2. The discrepancies occur precisely when the non-overlapping conditions aren't met, such as the poorly supervised background.

We now extend the lemma to a set of N Gaussians $\{G_i\}_{i=1}^N$ under two assumptions: (a) statistical independence, following RayGauss [7], and (b) non-overlapping Gaussian primitives [8]. Although the second assumption is approximate, alpha blending makes a similar approximation by sorting Gaussians front-to-back and compositing them as if overlap were negligible in 3D. In practice, this approximation is effective.

Proposition 2 (Transmittance of Gaussian Splatting). *Assuming statistical independence and non-overlapping Gaussian primitives, the image formation model of a Gaussian Splatting representation—relying on alpha-blending with opacity—is equivalent to a ray-marching volumetric rendering model with the following continuous transmittance function, for any ray origin $\boldsymbol{o} \in \mathbb{R}^3$ in empty space, $\boldsymbol{w} \in \mathcal{S}^2$, and $t \in [0, +\infty)$:*

$$T_{\boldsymbol{o},\boldsymbol{w}}(t) = \prod_{i=1}^{N} \left(1 - \bar{G}^{(i)}_{\boldsymbol{o},\boldsymbol{w}}(t)\right) , \tag{17}$$

where $\bar{G}^{(i)}_{\boldsymbol{o},\boldsymbol{w}}(t) = G_i(\boldsymbol{o} + \min(t, t^{G_i}_{\boldsymbol{o},\boldsymbol{w}})\boldsymbol{w})$. The corresponding attenuation coefficient is view-dependent and given by:

$$\sigma(\boldsymbol{x}, \boldsymbol{w}) = \sum_{i=1}^{N} \max\left(0, -\boldsymbol{w} \cdot \nabla \log(1 - G_i(\boldsymbol{x}))\right) . \tag{18}$$

Sketch of proof. Extending the previous proposition to multiple Gaussians is straightforward, as statistical independence between Gaussians implies that the

total transmittance is the product of the individual transmittances. Finally, assuming Gaussians do not overlap in the 3D domain allows for proving the equivalence between the two image formation models. Figure 7 compares images rendered by ray-marching our attenuation field and by the standard Gaussian Splatting rasterizer, confirming visual consistency.

Proof. We first derive the transmittance expression from statistical independence. We then verify equivalence between alpha blending and the proposed density-based model.
We assume the primitives to contribute independently to attenuation, such that the total coefficient is the sum of the individual coefficients: $\sigma(\mathbf{x}, \mathbf{w}) = \sum_i \sigma_i(\mathbf{x}, \mathbf{w})$. Equivalently, under exponential transport, the total transmittance is the product of the individual transmittances:

$$\mathrm{T}_{\mathbf{o},\mathbf{w}}(t) = \exp\left(-\int_0^t \sum_i \sigma_i(\mathbf{o}+s\mathbf{w},\mathbf{w})\mathrm{d}s\right) \tag{19}$$

$$= \prod_i \exp\left(-\int_0^t \sigma_i(\mathbf{o}+s\mathbf{w},\mathbf{w})\mathrm{d}s\right) \tag{20}$$

$$= \prod_i \mathrm{T}^{(i)}_{\mathbf{o},\mathbf{w}}(t)\,, \tag{21}$$

where $\mathrm{T}^{(i)}_{\mathbf{o},\mathbf{w}}(t)$ denotes the transmittance of the i-th Gaussian. Using Lemma 1, we obtain

$$\mathrm{T}_{\mathbf{o},\mathbf{w}}(t) = \prod_i \left(1 - \bar{G}^{(i)}_{\mathbf{o},\mathbf{w}}(t)\right)\,. \tag{22}$$

We now verify equivalence between alpha blending and the proposed density-based model by comparing rendered color values.
Consider an empty ray origin $\mathbf{o}$ and direction $\mathbf{w}$. Under the non-overlap assumption, Gaussians can easily be ordered along the ray, e.g., by projecting their centers onto the ray. We retain the M Gaussians with positive projection and order them by increasing projected distance from $\mathbf{o}$. As Gaussians do not overlap, there exist $t_0 < \cdots < t_M$ such that

$$t_0 = 0 \tag{23}$$

$$t_M = +\infty \tag{24}$$

$$\forall t \in [t_{i-1}, t_i]\,,\ \sigma(\mathbf{o}+t\mathbf{w},\mathbf{w}) \simeq \sigma_i(\mathbf{o}+t\mathbf{w},\mathbf{w}) \tag{25}$$

$$\forall i \in [1, M]\,,\ \bar{G}^{(i)}_{\mathbf{o},\mathbf{w}}(t_{i-1}) \simeq 0 \tag{26}$$

$$\forall i \in [1, M]\,,\ \bar{G}^{(i)}_{\mathbf{o},\mathbf{w}}(t_i) = \bar{G}^{(i)}_{\mathbf{o},\mathbf{w}}(t^{G_i}_{\mathbf{o},\mathbf{w}})\quad. \tag{27}$$

Similarly, we equip each Gaussian G_i with a directional color field $\mathbf{c}_i : \mathcal{S}^2 \to [0,1]^3$ and assume the total color field $\mathbf{c} : \mathbb{R}^3 \times \mathcal{S}^2 \to [0,1]^3$ to verify

$\mathbf{c}(\mathbf{o}+t\mathbf{w},\mathbf{w}) \simeq \mathbf{c}_i(\mathbf{w})$ for all $t \in [t_{i-1}, t_i]$. As a result, for any $i \leq M$ and $t \in [t_{i-1}, t_i]$, we have:

$$\bar{G}^{(j)}_{\mathbf{o},\mathbf{w}}(t) = \begin{cases} G_j(\mathbf{o} + t^{G_j}_{\mathbf{o},\mathbf{w}}\mathbf{w}) & \text{if } j < i \\ G_i(\mathbf{o}+t\mathbf{w}) & \text{if } j = i \\ 0 & \text{if } j > i \end{cases}, \tag{28}$$

such that for all $t \in [t_{i-1}, t_i]$:

$$\mathrm{T}^{(j)}_{\mathbf{o},\mathbf{w}}(t) = \begin{cases} 1 - G_j(\mathbf{o} + t^{G_j}_{\mathbf{o},\mathbf{w}}\mathbf{w}) & \text{if } j < i \\ 1 - G_i(\mathbf{o}+t\mathbf{w}) & \text{if } j = i \\ 1 & \text{if } j > i \end{cases}, \tag{29}$$

and, still for all $t \in [t_{i-1}, t_i]$:

$$\begin{aligned} \mathrm{T}_{\mathbf{o},\mathbf{w}}(t) &= \prod_{j=1}^{M} \mathrm{T}^{(j)}_{\mathbf{o},\mathbf{w}}(t) \\ &= \mathrm{T}^{(i)}_{\mathbf{o},\mathbf{w}}(t) \prod_{j=1}^{i-1} \mathrm{T}^{(j)}_{\mathbf{o},\mathbf{w}}(t^{G_j}_{\mathbf{o},\mathbf{w}}) \quad . \end{aligned} \tag{30}$$

The rendered color value $C_{\mathbf{o},\mathbf{w}}$ along the ray is then given by:

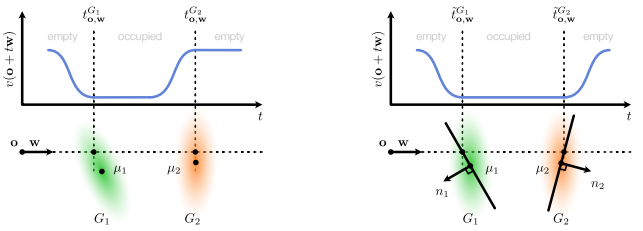
$$\begin{aligned} C_{\mathbf{o},\mathbf{w}} &= \int_0^{+\infty} \sigma(\mathbf{o}+s\mathbf{w},\mathbf{w})\mathrm{T}_{\mathbf{o},\mathbf{w}}(s)\mathbf{c}(\mathbf{o}+s\mathbf{w},\mathbf{w})\mathrm{d}s \\ &= \sum_{i=1}^{M} \int_{t_{i-1}}^{t_i} \sigma(\mathbf{o}+s\mathbf{w},\mathbf{w})\mathrm{T}_{\mathbf{o},\mathbf{w}}(s)\mathbf{c}(\mathbf{o}+s\mathbf{w},\mathbf{w})\mathrm{d}s \\ &= \sum_{i=1}^{M} \int_{t_{i-1}}^{t_i} \sigma_i(\mathbf{o}+s\mathbf{w},\mathbf{w}) \left(\mathrm{T}^{(i)}_{\mathbf{o},\mathbf{w}}(s) \prod_{j=1}^{i-1} \mathrm{T}^{(j)}_{\mathbf{o},\mathbf{w}}(t^{G_j}_{\mathbf{o},\mathbf{w}}) \right) \mathbf{c}_i(\mathbf{w})\mathrm{d}s \\ &= \sum_{i=1}^{M} \mathbf{c}_i(\mathbf{w}) \left(\int_{t_{i-1}}^{t_i} \sigma_i(\mathbf{o}+s\mathbf{w},\mathbf{w})\mathrm{T}^{(i)}_{\mathbf{o},\mathbf{w}}(s)\mathrm{d}s \right) \prod_{j=1}^{i-1} \mathrm{T}^{(j)}_{\mathbf{o},\mathbf{w}}(t^{G_j}_{\mathbf{o},\mathbf{w}}) \\ &= \sum_{i=1}^{M} \mathbf{c}_i(\mathbf{w}) \left[\mathrm{T}^{(i)}_{\mathbf{o},\mathbf{w}}(t_i) - \mathrm{T}^{(i)}_{\mathbf{o},\mathbf{w}}(t_{i-1}) \right] \prod_{j=1}^{i-1} \mathrm{T}^{(j)}_{\mathbf{o},\mathbf{w}}(t^{G_j}_{\mathbf{o},\mathbf{w}}) \\ &= \sum_{i=1}^{M} \mathbf{c}_i(\mathbf{w}) \bar{G}^{(i)}_{\mathbf{o},\mathbf{w}}(t^{G_i}_{\mathbf{o},\mathbf{w}}) \prod_{j=1}^{i-1} \left(1 - \bar{G}^{(j)}_{\mathbf{o},\mathbf{w}}(t^{G_j}_{\mathbf{o},\mathbf{w}}) \right) \quad . \end{aligned} \tag{31}$$

The right-hand side is exactly the alpha-blending color for multiple Gaussians [55, 56], proving equivalence. □

B.2 Oriented Gaussians

The attenuation factor from the previous section is not reciprocal: $\sigma(\mathbf{x}, \mathbf{w}) \neq \sigma(\mathbf{x}, -\mathbf{w})$ in general. Each Gaussian G_i contributes to attenuation only before the ray reaches its point of maximum contribution $\mathbf{o} + t^{G_i}_{\mathbf{o},\mathbf{w}}\mathbf{w}$, so reversing the ray changes where transmittance varies, breaking reciprocity. This asymmetry primarily manifests when a ray *exits* an object (see Fig. 8).

Reciprocity is a necessary condition for applying the OaV results (see main paper), which links attenuation to the direction-independent $\nabla \log v$. Without it, back-facing Gaussians introduce a systematic bias that prevents deriving a closed-form vacancy expression.



(a) Initial attenuation. **(b)** Oriented attenuation.

Fig. 8: Evolution of the vacancy along a ray. In contrast to the initial attenuation we derived for Gaussian Splatting (a), our oriented Gaussian formulation (b) allows for a reciprocal formulation of the vacancy. This figure shows how our formulation corrects the bias present in (a). Indeed, in (a) vacancy starts changing just before each Gaussian. This poses a problem for G_2, since vacancy starts evolving from occupied to empty before crossing $t^{G_2}_{o,w}$. You can see in that in our formulation (b) this change happens after crossing this critical point. This renders the evaluation of vacancy reciprocal

To restore reciprocity, we *orient* each Gaussian with a normal vector and modify the attenuation accordingly, assuming the scene geometry is *wrapped* by oriented Gaussians each covering a local surface patch.

Definition 2 (Oriented Gaussian). *Consider a parameter $n_i \in \mathcal{S}^2$ associated to each G_i — we call it the normal vector of the Gaussian. We define the oriented attenuation coefficient of the Gaussian G_i as*

$$\bar{\sigma}_i(\boldsymbol{x}, \boldsymbol{w}) = \mathbb{1}_{n_i^T(\boldsymbol{x}-\mu_i)\geq 0} |\boldsymbol{w} \cdot \nabla \log(1 - G_i(\boldsymbol{x}))| \,. \tag{32}$$

By introducing n_i, we replace the direction-dependent split at $t^{G_i}_{\mathbf{o},\mathbf{w}}$ with a fixed half-space: $\bar{\sigma}_i$ is non-zero where $n_i^T(\mathbf{x} - \mu_i) \geq 0$ and zero otherwise. Since this half-space depends only on the Gaussian's position and normal—not the ray direction—the resulting attenuation is reciprocal by construction, enabling the application of OaV's results, and thus the derivation of a closed-form equation for vacancy (see Eq. (34)).

This introduces a consistency requirement: the region where transmittance is constant must lie inside the object (see Fig. 8), which holds when Gaussians properly *wrap* the surface. Concretely, both formulations agree for non-occluded, front-facing Gaussians when the point of maximum contribution $\mathbf{o}+t^{G_1}_{\mathbf{o},\mathbf{w}}\mathbf{w}$ aligns with the plane intersection $\mathbf{o}+\tilde{t}^{G_1}_{\mathbf{o},\mathbf{w}}\mathbf{w}$.

Normal alignment loss. We promote Gaussians to "wrap" the scene through the following normal alignment loss.

$$\mathcal{L}_{\mathrm{N}} = \sum_{p} 1 - N(p) \cdot \nabla D(p) \,, \tag{33}$$

where $N(p)$ is the rendered oriented normal at pixel p and $\nabla D(p)$ is the image-space gradient of the rendered depth. Following [39], rendering depth and normals n_i with the Splatting rasterizer [55, 56] yields respectively the expected depth of the point of maximum contribution $\mathbf{o}+t^{G_1}_{\mathbf{o},\mathbf{w}}\mathbf{w}$, and the expected orientation of the Gaussians along the ray. As a result, $\mathcal{L}_{\mathrm{N}}$ enforces $\mathbf{o}+t^{G_1}_{\mathbf{o},\mathbf{w}}\mathbf{w}$ and the plane intersections $\mathbf{o}+\tilde{t}^{G_1}_{\mathbf{o},\mathbf{w}}\mathbf{w}$ to be similar on average, encouraging the two formulations to agree.

B.3 Deriving Surface Normals

The previous section established that a Gaussian Splatting representation is approximately equivalent to an attenuation-based model. When Gaussians are aligned on a surface and properly wrap the scene geometry, orienting each Gaussian with a normal vector yields a reciprocal attenuation coefficient σ. Gaussian Splatting then satisfies the reciprocal exponential transport assumption of OaV [39], and their result provides an explicit expression of $\nabla \log v$.

This section leverages this gradient field to derive an explicit normal field on the stochastic surface. This normal field serves two purposes: supervising oriented Gaussians during training to enforce proper surface wrapping, and extracting a mesh from the trained representation.

Definition 3 (Gaussian vector field). *Consider a Gaussian Splatting representation of N oriented Gaussians. The Gaussian vector field V is defined as the gradient of the log-vacancy. Following OaV:*

$$V(x) = \nabla \log v(x) = \sum_{i=1}^{N} \mathbb{1}_{n_i^T(\boldsymbol{x}-\mu_i)\geq 0} \nabla \log(1 - G_i(x)) \,. \tag{34}$$

Proof. Combining the result of OaV and Eq. (32), for any $\mathbf{x} \in \mathbb{R}^3$ and $\mathbf{w} \in \mathcal{S}^2$, yields:

$$|\mathbf{w} \cdot \nabla \log v(\mathbf{x})| = \sum_{i} \mathbb{1}_{n_i^T(\mathbf{x}-\mu_i)\geq 0} |\mathbf{w} \cdot \nabla \log(1 - G_i(\mathbf{x}))| \,. \tag{35}$$

Under the wrapping assumption, each gradient term $\nabla \log(1 - G_i(\mathbf{x}))$ points from occupied space ($v = 0$) toward empty space ($v = 1$) when $\mathbb{1}_{n_i^T(\mathbf{x}-\mu_i)\geq 0} > 0$. As a result, we assume that, locally, each term $\mathbb{1}_{n_i^T(\mathbf{x}-\mu_i)\geq 0}\mathbf{w} \cdot \nabla \log(1 - G_i(\mathbf{x}))$ either has the same sign as $\mathbf{w} \cdot \nabla \log v(\mathbf{x})$ or is equal to zero. Hence, we can disambiguate the absolute values within the sum and write:

$$\mathbf{w} \cdot \nabla \log v(\mathbf{x}) = \sum_i \mathbf{w} \cdot \mathbb{1}_{n_i^T(\mathbf{x}-\mu_i)\geq 0} \nabla \log(1 - G_i(\mathbf{x})) \,. \tag{36}$$

Since Eq. 36 holds for arbitrary direction $\mathbf{w} \in \mathcal{S}^2$, Eq. 34 follows. □

Proposition 3 (Gaussian normal field). *Consider a Gaussian Splatting representation of N oriented Gaussians. The Gaussian normal field $\mathcal{N} : \mathbb{R}^3 \to \mathcal{S}^2$ is defined as the normalized Gaussian vector field $V : \mathbb{R}^3 \to \mathbb{R}^3$:*

$$\mathcal{N}(x) = \begin{cases} \frac{V(x)}{\|V(x)\|} & \text{if } \|V(x)\| > 0 \\ 0 & \text{otherwise} \end{cases} \,. \tag{37}$$

In a neighborhood of the surface, $\|V\| > 0$ and $\mathcal{N}$ coincides with the true normal field of the expected stochastic surface of M.

Proof. The normal field of a scene $E \subset \mathbb{R}^3$ with surface $\partial E \subset E$ is defined as the gradient of a signed distance function (SDF) $f : \mathbb{R}^3 \to \mathbb{R}$ satisfying

$$f(x) \leq 0 \quad \text{if } x \in E, \tag{38}$$
$$f(x) > 0 \quad \text{if } x \notin E, \tag{39}$$
$$f(x) = 0 \quad \text{if } x \in \partial E, \tag{40}$$
$$\|\nabla f(x)\| = 1 \quad \text{for all } x \in \mathbb{R}^3 \quad \text{(Eikonal equation)}. \tag{41}$$

Following *Objects as Volumes* [39], we assume the SDF f representing the expected stochastic surface ∂M is locally a scalar function of the vacancy v. Specifically, we assume there exists an increasing sigmoidal function $\Psi : \mathbb{R} \to (0, 1)$ and a scalar $s > 0$ such that, for all x in a neighborhood of the surface:

$$f(x) = \frac{1}{s}\Psi^{-1}(v(x)) \,. \tag{42}$$

Since Ψ^{-1} and log are strictly increasing, ∇f and $\nabla \log v$ are collinear with a positive scalar factor in a neighborhood of the surface. The Eikonal constraint $\|\nabla f\| = 1$ then implies $\nabla f = \nabla \log v \,/\, \|\nabla \log v\|$, so $\mathcal{N}$ coincides with the true surface normal field near the surface. □

In summary, under our oriented Gaussian assumption, the normal field of the stochastic surface represented by Gaussian Splatting admits an explicit closed-

form expression through the Gaussian vector field V (Eq. 34). This expression enables both surface-aware regularization during training and mesh extraction at inference time. We can see in Fig. 9, that the normalized vector field (i.e, the normal field) queried at the median depth (i.e, 0.5-isosurface our transmittance) has been regularized to match the rendered normals only through rendering losses. We show that this is the case in theory too in Sec. B.3.

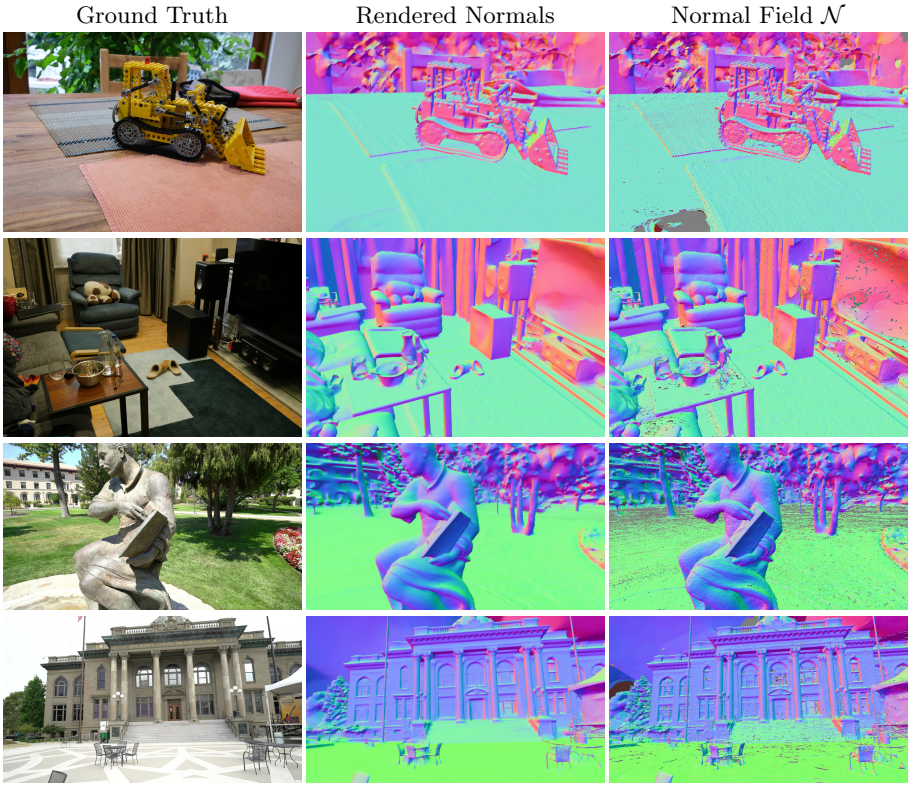


Fig. 9: Gaussian vector field aligned with surface normals. Visualization of the Gaussian normal field $\mathcal{N}$, *i.e.* the normalized vector field $V = \nabla \log v$, across four scenes (Kitchen, Room, Ignatius, Courthouse). Each row shows, from left to right: the ground truth RGB image, the normals rendered by alpha-blending the oriented Gaussian normals n_i, and the normal field $\mathcal{N}(x)$ queried at points x lying on the 0.5-isosurface of the transmittance (median depth). In practice, $\mathcal{N}(x)$ is evaluated by querying the K-nearest Gaussians to x ($K = 32$). The close agreement between rendered normals and $\mathcal{N}$ confirms that the Gaussian vector field is well-aligned with the surface. We note that artifacts in the normal field $\mathcal{N}$ correspond to under-reconstructed parts of the scene (e.g specular spot on Kitchen table, TV in room, background in Ignatius scene, and sky in Courthouse scene).

Lemma 2 (Supervision of Normal Field). *We supervise the normal field $\mathcal{N}$ by enforcing consistency between:*

1. *The rendered normal maps created by rasterizing the 3D Gaussian Splatting representation with normals n_i;*
2. *The image-derivative of the depth.*

Doing so promotes the value of our normal field at the 0.5-isosurface of the transmittance to match the true normal of the surface.

Proof. First, let us start by showing why supervising the rendered normal maps—produced by splatting Gaussians with their learnable normals n_i—also supervises the normal field $\mathcal{N}$.
For a given ray $r_{\mathbf{o},\mathbf{w}}$, the expectancy of the normal field value at the observed surface is equal to

$$\mathbb{E}_M(\mathcal{N}(\mathbf{o}+t^*_{\mathbf{o},\mathbf{w}}\mathbf{w})) = \int_0^{+\infty} p^{\text{ff}}_{\mathbf{o},\mathbf{w}}(s)N(\mathbf{o}+s\mathbf{w})\mathrm{d}s \tag{43}$$

where $t^*_{\mathbf{o},\mathbf{w}}$ is the free-flight distance and $p^{\text{ff}}_{\mathbf{o},\mathbf{w}}$ is the pdf of $t^*_{\mathbf{o},\mathbf{w}}$. Following OaV, $p^{\text{ff}}_{\mathbf{o},\mathbf{w}}(t) = \sigma(\mathbf{o}+t\mathbf{w},\mathbf{w})\mathrm{T}_{\mathbf{o},\mathbf{w}}(t)$ and the expectancy is equal to

$$\mathbb{E}_M(\mathcal{N}(\mathbf{o}+t^*_{\mathbf{o},\mathbf{w}}\mathbf{w})) = \int_0^{+\infty} \sigma(\mathbf{o}+s\mathbf{w},\mathbf{w})\mathrm{T}_{\mathbf{o},\mathbf{w}}(s)N(\mathbf{o}+s\mathbf{w})\mathrm{d}s\,, \tag{44}$$

which is precisely equal to the rendering equation of radiance fields.
In the context of Gaussian splatting, this integral is discretized and approximated with alpha-blending. As such, for rendering the normal field, we need to attribute to every Gaussian a single value $\mathcal{N}_i$ per Gaussian to approximate the above integral. Let's start with the vector field V. We need to select a single value V_i for rendering. A natural choice is to use the average value of the contribution of the Gaussian to V, *i.e.*, the integral of the i-th term of the sum multiplied by the normalized density of the Gaussian:

$$V_i \propto \int_{\mathbb{R}^3} G_i(x)\cdot \mathbb{1}_{n_i^T(\mathbf{x}-\mu_i)\geq 0}\nabla\log(1-G_i(x))\mathrm{d}x \tag{45}$$

This integral has a symmetry around the vector n_i (this can be easily seen with a change of variable to reduce and center the Gaussian; the resulting integral is over a hemisphere). As such, the result $V_i \propto n_i$, and similarly the value of the normalized vector field $\mathcal{N}_i \sim n_i$ as both have norm 1.
As a consequence, *one can simply render the plane normals n_i with Gaussian splatting to estimate the average value of the Normal field $\mathcal{N}$ on the observed surface*, and supervise $\mathcal{N}$ through a rendering loss.
The question then arises of how to supervise the rendered normal maps. We propose to compare this value to the derivative of the rendered depth.

Indeed, according to [39], rendering the depth directly estimates the expectancy of the free-flight distance $t^*_{\mathbf{o},\mathbf{w}}$ for each pixel, *i.e.*, the average position of the visible surface point. By taking the derivative of the depth (*i.e.*, the normal of the expected visible surface), we can then compute an estimate of the surface normals that does not depend on our *oriented Gaussians* assumption and normals n_i. We can finally use this estimate to supervise our plane normals.
More precisely, we choose as the depth to use, the median value of $t^*_{\mathbf{o},\mathbf{w}}$ (as opposed to the expectancy). Indeed, the median provides an estimate of $t^*_{\mathbf{o},\mathbf{w}}$ which is more robust to statistical outliers than the expectancy. The median of $t^*_{\mathbf{o},\mathbf{w}}$ can be computed as

$$\begin{aligned}
\text{Median}_M(t^*_{\mathbf{o},\mathbf{w}}) &= \inf\left\{t \in [0,+\infty) : \int_0^t p^{\text{ff}}_{\mathbf{o},\mathbf{w}}(s)\mathrm{d}s \geq 0.5\right\} \\
&= \inf\left\{t \in [0,+\infty) : \int_0^t -\frac{\mathrm{d}}{\mathrm{d}t}\mathrm{T}_{\mathbf{o},\mathbf{w}}(s)\mathrm{d}s \geq 0.5\right\} \\
&= \inf\left\{t \in [0,+\infty) : \mathrm{T}_{\mathbf{o},\mathbf{w}}(0) - \mathrm{T}_{\mathbf{o},\mathbf{w}}(t) \geq 0.5\right\} \\
&= \inf\left\{t \in [0,+\infty) : 1 - \mathrm{T}_{\mathbf{o},\mathbf{w}}(t) \geq 0.5\right\} \\
&= \inf\left\{t \in [0,+\infty) : \mathrm{T}_{\mathbf{o},\mathbf{w}}(t) \leq 0.5\right\}\ ,
\end{aligned} \tag{46}$$

such that in the context of point-based rendering, the median of $t^*_{\mathbf{o},\mathbf{w}}$ can be estimated as the depth $t^G_{\mathbf{o},\mathbf{w}}$ of the first Gaussian along the ray for which the transmittance reaches the 0.5 threshold. This definition is consistent with the median depth as implemented in previous works [42, 55, 56] □

C Mesh Extraction Details

We now proceed to leverage the closed-form equation for the vacancy for mesh extraction. In this section, we provide additional details on the procedures introduced in the main paper.

C.1 Estimating Vacancy

As stated, we leverage the ability to compute the vacancy v—the probability of a point to be outside the opaque volume—for arbitrary points of space. In this section, we provide further motivation and proof for the formula introduced in the main document.

The explicit value of the log-vacancy $\mathbf{x}$ can be computed by integrating the vector field along any camera ray $r_{\mathbf{o},\mathbf{w}}$ such that $\mathbf{x} = o + t\mathbf{w}$ for some $t \in [0,+\infty)$:

$$\begin{aligned}\int_0^t V(\mathbf{o}+s\mathbf{w})\cdot\mathbf{w}\mathrm{d}s &= \int_0^t \nabla \log v(\mathbf{o}+s\mathbf{w})\cdot\mathbf{w}\mathrm{d}s \\ &= \log v(\mathbf{o}+t\mathbf{w}) - \log v(\mathbf{o}) \\ &= \log v(\mathbf{x})\end{aligned} \quad , \tag{47}$$

since $\log v(\mathbf{o}) = \log 1 = 0$ as the ray origin $\mathbf{o}$ is located in empty space.

However, directly integrating the vector field might produce noisy results as it assumes all Gaussians follow our *oriented Gaussian* assumption. In practice, some Gaussians not participating in the rendering might be trapped inside the volume and perturb the computation of this integral. Moreover, even though our optimization and densification greatly help to enforce our assumption and wrap the geometry with oriented Gaussians, it is possible for *holes* to remain. Consequently, we propose another computation which proves to be more robust to such scenarios.

Proposition 4. *Let $v : \mathbb{R}^3 \to [0,1]$ be the vacancy function, and let $\mathcal{T}_c$ denote the set of all training camera rays. For any point $\boldsymbol{x} \in \mathbb{R}^3$, the vacancy admits the following lower bound:*

$$v(\boldsymbol{x}) \geq \max_{(\boldsymbol{o},\boldsymbol{w})\in\mathcal{T}_c} \left\{ \prod_{i=1}^{N} \left(1 - \bar{G}^{(i)}_{\boldsymbol{o},\boldsymbol{w}}(t)\right) : \boldsymbol{x} = \boldsymbol{o} + t\boldsymbol{w}, t > 0 \right\} , \tag{48}$$

where $\bar{G}^{(i)}_{\boldsymbol{o},\boldsymbol{w}}(t) = G_i(\boldsymbol{o} + \min(t, t^{G_i}_{\boldsymbol{o},\boldsymbol{w}})\boldsymbol{w})$; and $t^{G_i}_{\boldsymbol{o},\boldsymbol{w}} := \arg\max_{t\geq 0} G_i(\boldsymbol{o}+t\boldsymbol{w})$.

Proof. Following OaV, we establish the following relation between the log-vacancy and the log-transmittance for any $\mathbf{o}, \mathbf{w}, t$:

$$|w \cdot \nabla \log v(\mathbf{o}+t\mathbf{w})| = -\frac{\mathrm{d}}{\mathrm{d}t} \log \mathrm{T}_{\mathbf{o},\mathbf{w}}(t) , \tag{49}$$

which implies, as $\frac{\mathrm{d}}{\mathrm{d}t} \log \mathrm{T}_{\mathbf{o},\mathbf{w}}(t)$ is always non-positive:

$$w \cdot \nabla \log v(\mathbf{o}+t\mathbf{w}) \geq \frac{\mathrm{d}}{\mathrm{d}t} \log \mathrm{T}_{\mathbf{o},\mathbf{w}}(t) . \tag{50}$$

By integrating the formula from 0 to t along the ray $r_{\mathbf{o},\mathbf{w}}$,

$$\log v(\mathbf{o}+t\mathbf{w}) - \log v(\mathbf{o}) \geq \log \mathrm{T}_{\mathbf{o},\mathbf{w}}(t) - \log \mathrm{T}_{\mathbf{o},\mathbf{w}}(0) , \tag{51}$$

with $\log v(\mathbf{o}) = \log \mathrm{T}_{\mathbf{o},\mathbf{w}}(0) = \log 1 = 0$, resulting in:

$$\begin{aligned}\log v(\mathbf{o}+t\mathbf{w}) &\geq \log \mathrm{T}_{\mathbf{o},\mathbf{w}}(t) \\ v(\mathbf{o}+t\mathbf{w}) &\geq \mathrm{T}_{\mathbf{o},\mathbf{w}}(t) \ .\end{aligned} \tag{52}$$

Consequently, for any point $\mathbf{x}$, we have:

$$v(\mathbf{x}) \geq \max\left\{ \mathrm{T}_{\mathbf{o},\mathbf{w}}(t) : (\mathbf{o},\mathbf{w}) \in \mathbb{R}^3 \times \mathcal{S}^2, t > 0, \text{ s.t. } \mathbf{x} = \mathbf{o} + t\mathbf{w} \right\} . \tag{53}$$

Equation 53 shows that we can compute a lower bound of the vacancy at $\mathbf{x}$ as the maximum of transmittance values along camera rays passing through $\mathbf{x}$. This particularly holds for the set $\mathcal{T}_c$ of all training camera rays. □

In practice, for any 3D point $\mathbf{x}$, we iterate over training cameras and compute the transmittance along the corresponding ray for each camera. **We finally select the maximum transmittance over the camera rays as an estimate of the vacancy.**

The computation of the lower bound is more robust to floaters and Gaussians hidden inside the geometry, as transmittance is non-decreasing and mainly depends on the visible Gaussians participating in the rendering. Additionally, we use the transmittance from the non-oriented scenario in Eq. (53). This choice of transmittance is more robust to outlier Gaussians not "properly" wrapping some parts of the scene. Moreover, our losses enforce un-oriented and oriented formulations of the transmittance formula to be consistent for visible Gaussians.

C.2 Newton Projection Step

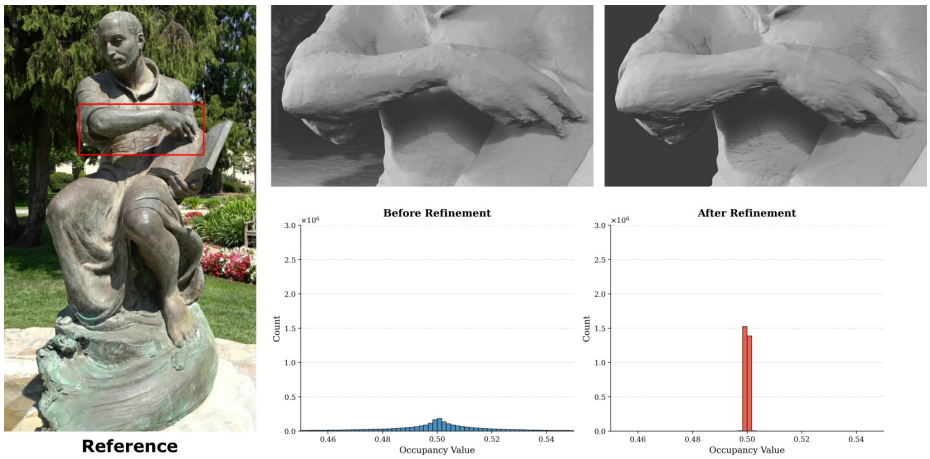


Fig. 10: Effect of the Newton projection step on vacancy values at sampled points. *Left:* Reference image with a mark on zoomed area. *Top:* mesh renders of the Baseline and our proposed method on the Truck scene of T&T. *Bottom:* distribution of vacancy values $v(x)$ evaluated at the points to refine for PAM (zoomed in on the $[0.45, 0.55]$ range around the target isosurface $v = 0.5$). This plot was obtained performing PAM on the Ignatius scene of T&T, sampling 3 Million points and performing 10 refinement steps. *Bottom Left (blue):* Sampled points on mesh without Newton refinement; *Bottom Right (red):* after applying our Newton projection step (Proposition 5), points collapse onto the isosurface $v = 0.5$, validating the efficiency of our refinement. Qualitatively we can see that disentangling the vertex distribution from the Gaussians distributions allows to retrieve extremely fine-grained details.

Here, we motivate the choice of the Newton step we use for refining the point cloud during *Primal Adaptive Meshing*. Fig. 10 illustrates the effectiveness of this procedure. The Newton step aims to collapse points to the 0.5-isosurface of the vacancy.

Proposition 5. *Let $v\colon \mathbb{R}^3 \to [0,1]$ be the vacancy function and note that the normal field $\mathcal{N}(x) = \nabla v(x)/\|\nabla v(x)\|^2$. The first-order Newton step projecting x_i onto the isosurface $v^{-1}(0.5)$ and restricting updates to the gradient direction is:*

$$x_{i+1} = x_i + \frac{0.5 - v(x_i)}{2}\mathcal{N}(x_i). \tag{54}$$

Proof. Introduce the auxiliary function $f(x) = (0.5 - v(x))^2$, whose roots coincide with the isosurface. Linearising f around x_i and setting to zero gives the Newton condition:

$$f(x_i) + \nabla f(x_i)^\top (x_{i+1} - x_i) = 0. \tag{55}$$

We restrict to gradient-direction updates by writing $x_{i+1} = x_i + t\,\nabla v(x_i)$ for a scalar step t. Substituting $\nabla f(x) = -2\big(0.5 - v(x)\big)\nabla v(x)$ and solving for t:

$$t = \frac{-f(x_i)}{\nabla f(x_i)^\top \nabla v(x_i)} = \frac{-(0.5 - v(x_i))^2}{-2(0.5 - v(x_i))\|\nabla v(x_i)\|^2} = \frac{0.5 - v(x_i)}{2\|\nabla v(x_i)\|^2}. \tag{56}$$

Therefore:

$$x_{i+1} = x_i + t\,\nabla v(x_i) = x_i + \frac{0.5 - v(x_i)}{2\|\nabla v(x_i)\|^2}\,\nabla v(x_i) = x_i + \frac{0.5 - v(x_i)}{2}\mathcal{N}(x_i). \tag{57}$$

□

D Experiments Details

Loss Details. At training time, we use our normal alignment loss $\mathcal{L}_{\mathrm{N}}$ in conjunction with the following set of losses: The standard photometric [25] loss $\mathcal{L}_{\mathrm{RGB}}$, depth-normal consistency $\mathcal{L}_{\mathrm{DN}}$ [55, 56] and the multi-view loss $\mathcal{L}_{\mathrm{MV}} = \lambda_{\mathrm{pc}}\mathcal{L}_{\mathrm{pc}} + \lambda_{\mathrm{gc}}\mathcal{L}_{\mathrm{gc}}$, where the terms control the strength of the photometric and geometric consistency as defined in [10, 57]. These have the following weights $\lambda_{\mathrm{DN}} = 0.05, \lambda_{\mathrm{N}} = 0.05, \lambda_{\mathrm{pc}} = 0.6, \lambda_{\mathrm{gc}} = 0.02$. This results in the total loss

$$\mathcal{L} = \mathcal{L}_{\mathrm{RGB}} + \lambda_{\mathrm{DN}}\mathcal{L}_{\mathrm{DN}} + \lambda_{\mathrm{N}}\mathcal{L}_{\mathrm{N}} + \mathcal{L}_{\mathrm{MV}}$$

Learnable Normals Parameterization. We observed that parameterizing the learnable normals n_i as simple, normalized vectors of 3 coordinates leads to unstable

optimization. Indeed, flipping the direction of the normal under this naive parameterization requires the vector to perform a rotation of 180 degrees, leading to convergence toward inaccurate local minima and poor efficiency. Instead, we parameterize the normal n_i of the Gaussian i with 4 parameters, including a sign parameter $s_i \in \mathbb{R}$ and a direction $d_i \in \mathbb{R}^3$. The normal is computed as:

$$n_i = \tanh(s_i) \cdot \frac{d_i}{\|d_i\|}$$

This decomposition into direction and sign enforces the norm of the normal to be lower than 1, while allowing for "flipping" the direction simply by changing the sign parameter s_i. Overall, it stabilizes gradients and prevents optimization from getting stuck in local minima.

CUDA implementation of forward and backward pass. As stated in the main paper, we provide a modified version of the efficient rasterizer proposed by the work Geometry Grounded Gaussian Splatting [57]. This modification consists of using our definition of transmittance (Proposition 2), instead of the original one that leads to the issues presented in Fig. 6. In practice, the differences are simple and involve removing the square root of the original formulation. We will release the code upon acceptance.

T&T Eval Bias. In the main paper we mention that there is a bias towards dense meshes in the current evaluation used by the rest of the literature on the T&T dataset. We give as simple support this snippet of code available in the RaDe-GS [56] repository (see Fig. 13). The code creates the point cloud to be compared to the ground truth by simply concatenating the available vertices of the mesh and the center of each face—as opposed to uniformly sampling the mesh surface.

D.1 DTU Qualitative Examples

In this section we briefly present some renders of the meshes extracted from DTU with our method. They can be found in Fig. 11.

D.2 T&T Shortcomings Visualizations.

We show in Fig. 12 that the T&T ground-truth point clouds contain holes due to occlusion and unfavorable acquisition angles. These gaps are not a property of the reconstructed mesh but of the acquisition process itself. Methods such as PGSR tend to produce meshes with holes that align with these GT gaps—due to their use of depth filtering [10] that relies on removing points viewed at grazing angles—which inflates their scores under the standard F1 metric. Our method reconstructs a more complete surface shell, which can be penalized in these under-observed regions.

Fig. 11: Qualitative mesh renders on DTU. Mesh renders of our method on three DTU scenes.

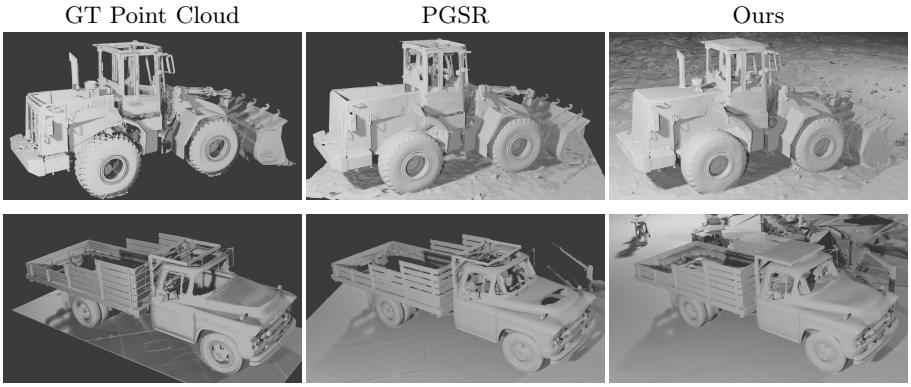


Fig. 12: T&T evaluation shortcomings on the Caterpillar scene. The standard Tanks & Temples ground-truth point cloud (left) contains holes due to occlusion and unfavorable acquisition angles—these are limitations of the LiDAR acquisition, not of the reconstruction. PGSR (center) produces a mesh with holes that align with GT gaps, which inflates its F1 score. Our method (right) reconstructs a more complete surface shell, which can be penalized by the evaluation metric in these under-observed regions. Note that there is no GT mesh, instead for visualization purposes we use the Points to Mesh Blender functionality to render the GT point cloud.

D.3 Additional Quantitative Results.

In this section we showcase some of the mentioned and discussed experiments and results in the main paper, such as the Legacy Table for the T&T dataset (Tab. 4); the DTU evaluation (Tab. 5); and the Mip NeRF360 Novel View Synthesis Evaluation (Tab. 6).

```python
import trimesh
mesh = trimesh.load_mesh(ply_path)
# add center points
sampled_vertices = mesh.vertices[mesh.faces].mean(axis=1)
# add 4 points based on the face vertices
# face_vertices = mesh.vertices[mesh.faces]# .mean(axis=1)
# weights = np.array([[3, 3, 3],
#                     [4, 4, 1],
#                     [4, 1, 4],
#                     [1, 4, 4]],dtype=np.float32) / 9.0
# sampled_vertices = np.sum(face_vertices.reshape(-1, 1, 3, 3) * weights.reshape(1, 4, 3, 1), axis=2)

vertices = np.concatenate([mesh.vertices, sampled_vertices], axis=0)
pcd = o3d.geometry.PointCloud()
pcd.points = o3d.utility.Vector3dVector(vertices)
### end add center points
```

Fig. 13: Evaluation Code of T&T used by the literature [10, 23, 55, 56] The traditional evaluation script uses the vertices and face center points from the mesh and uses them to build a point cloud. This way of constructing a prediction point cloud biases the metric towards dense meshes.

Table 4: Legacy quantitative comparison on the Tanks & Temples dataset [27] using the standard vertex-based metric. We report F1-score and average optimization time. All results besides ours are taken from the MILo paper. When a method fails due to out-of-memory errors, we report the mean over successful scenes, denoted by a superscript asterisk (x^*). Despite the known bias of this metric (see Fig. 13)—vertex-based sampling non-uniformly queries the isosurface in proportion to mesh density—our method achieves the best F1 score among all representations. The sensitivity of this metric to mesh density is further evidenced by the significant score increase obtained simply by adding more pivot points.

	Implicit						Explicit									
	Gaussian UDF	GS-Pull	GSDF	NeuS	Geo-Neus	Neuralangelo	SuGaR	3DGS	2DGS	GOF	RaDe-GS	QGS	VCR-GauS	MILo (GOF base)	**Ours (2p)**	**Ours (9p)**
Barn	0.27	0.60	0.16	0.29	0.33	0.70	0.14	0.13	0.36	0.51	0.43	0.43	0.62	0.59	0.58	0.65
Caterpillar	0.17	0.37	0.11	0.29	0.26	0.36	0.16	0.08	0.23	0.41	0.32	0.31	0.26	0.39	0.48	0.50
Courthouse	0.03	0.16	OOM	0.17	0.12	0.28	0.08	0.09	0.13	0.28	0.21	0.26	0.19	0.29	0.30	0.35
Ignatius	0.38	0.71	0.34	0.83	0.72	0.89	0.33	0.04	0.44	0.68	0.69	0.79	0.61	0.78	0.70	0.75
Meetingroom	0.17	0.22	0.01	0.24	0.20	0.32	0.15	0.01	0.16	0.28	0.25	0.25	0.19	0.26	0.33	0.39
Truck	0.30	0.52	0.25	0.45	0.45	0.48	0.26	0.19	0.26	0.59	0.51	0.60	0.52	0.62	0.58	0.61
Mean	0.22	0.43	0.18*	0.38	0.35	0.50	0.19	0.09	0.30	0.46	0.40	0.44	0.40	0.49	0.50	0.54
Time	90m	37.6m	70m	>24h	>24h	>24h	73 m	7.9 m	12.3 m	69m	11.5 m	120 m	53 m	150 m	27 m	27m

Table 5: Surface reconstruction metrics on the DTU dataset. We report the Chamfer Distance across 15 scenes. Our method achieves competitive performance on object-centric scenes, while being able to recover very fine details in unbounded scenes.

		24	37	40	55	63	65	69	83	97	105	106	110	114	118	122	Mean
implicit	NeRF [38]	1.90	1.60	1.85	0.58	2.28	1.27	1.47	1.67	2.05	1.07	0.88	2.53	1.06	1.15	0.96	1.49
	VolSDF [51]	1.14	1.26	0.81	0.49	1.25	0.70	0.72	1.29	1.18	0.70	0.66	1.08	0.42	0.61	0.55	0.86
	NeuS [50]	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	0.84
	Neuralangelo [29]	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	0.61
explicit	3D GS [25]	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	1.96
	SuGaR [19]	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	1.33
	2D GS [21]	0.48	0.91	0.39	0.39	1.01	0.83	0.81	1.36	1.27	0.76	0.70	1.40	0.40	0.76	0.52	0.80
	GOF [55]	0.50	0.82	0.37	0.37	1.12	0.74	0.73	1.18	1.29	0.68	0.77	0.90	0.42	0.66	0.49	0.74
	RaDe-GS [56]	0.46	0.73	0.33	0.38	0.79	0.75	0.76	1.19	1.22	0.62	0.70	0.78	0.36	0.68	0.47	0.68
	MILo [18]	0.43	0.74	0.34	0.37	0.80	0.74	0.70	1.21	1.22	0.66	0.62	0.80	0.37	0.76	0.48	0.68
	PGSR [10]	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	0.52
	GGGS (20k iter)	0.38	0.50	0.27	0.31	0.80	0.43	0.42	1.04	0.64	0.52	0.31	0.56	0.30	0.31	0.33	0.47
	GGGS (30k iter)	0.37	0.51	0.27	0.31	0.81	0.43	0.42	1.05	0.64	0.52	0.32	0.58	0.30	0.31	0.33	0.48
	Ours (20k iter)	0.39	0.50	0.26	0.32	0.80	0.46	0.44	1.05	0.65	0.52	0.34	0.58	0.32	0.33	0.34	0.49
	Ours (30k iter)	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	0.49

Table 6: Quantitative results for novel view synthesis on MipNeRF 360 [5]. We report PSNR, SSIM, and LPIPS. Our method maintains competitive rendering quality while significantly improving surface reconstruction.

	Indoor Scenes			Outdoor Scenes		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
3DGS [25]	30.41	0.920	0.189	24.64	0.731	0.234
Mip-Splatting [54]	30.90	0.921	0.194	24.65	0.729	0.245
BakedSDF [52]	27.06	0.836	0.258	22.47	0.585	0.349
SuGaR [19]	29.43	0.906	0.225	22.93	0.629	0.356
2DGS [44]	30.40	0.916	0.195	24.34	0.717	0.246
GOF [55]	30.79	0.924	0.184	24.82	0.750	0.202
RaDe-GS [56]	30.74	0.928	0.165	25.17	0.764	0.199
MILo (dense) [18]	30.76	0.934	0.155	24.81	0.744	0.229
Ours	30.43	0.932	0.162	26.49	0.815	0.174

D.4 Additional Qualitative Results

In this section we present some additional qualitative results. In Fig. 14, we show qualitative results of our ablation study. In Fig. 15, we show qualitative results of adding Gaussian Wrapping to RaDe-GS.

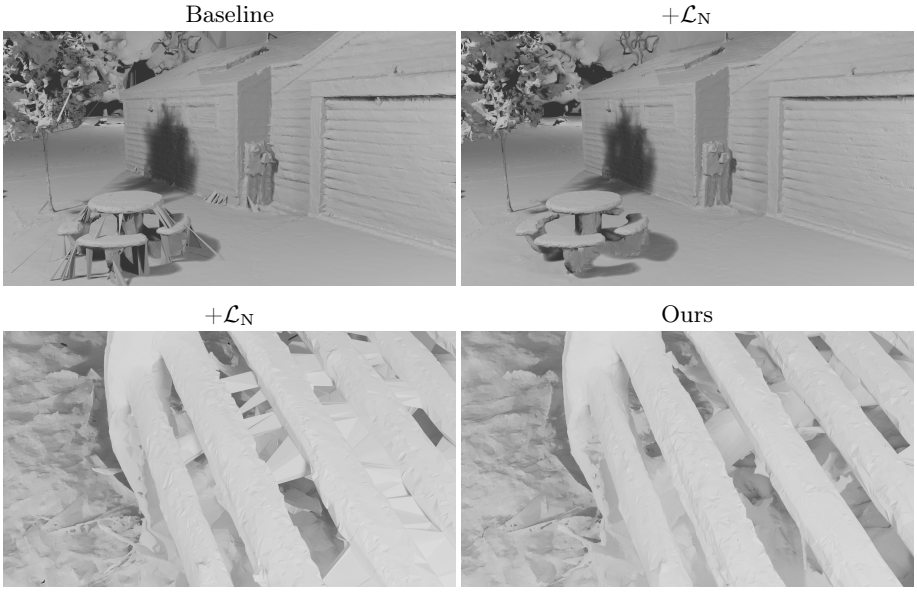


Fig. 14: Ablation of the normal field loss $\mathcal{L}_{\mathrm{N}}$ and densification on the Barn and Truck scenes. On the Barn scene, we see that by extracting a mesh with 2 pivots without our alignment loss $\mathcal{L}_N$, several artifacts are present. These come in the form of large triangles that stem from a bad orientation of the Gaussians which result in bad placement for the pivots. Then below, we ablate the need for our densification (Ours = Baseline + $\mathcal{L}_N$ + Densification), we can see that it allows to solve for complex and poorly supervised geometry such as in between the bench slats in the bicyle scene.

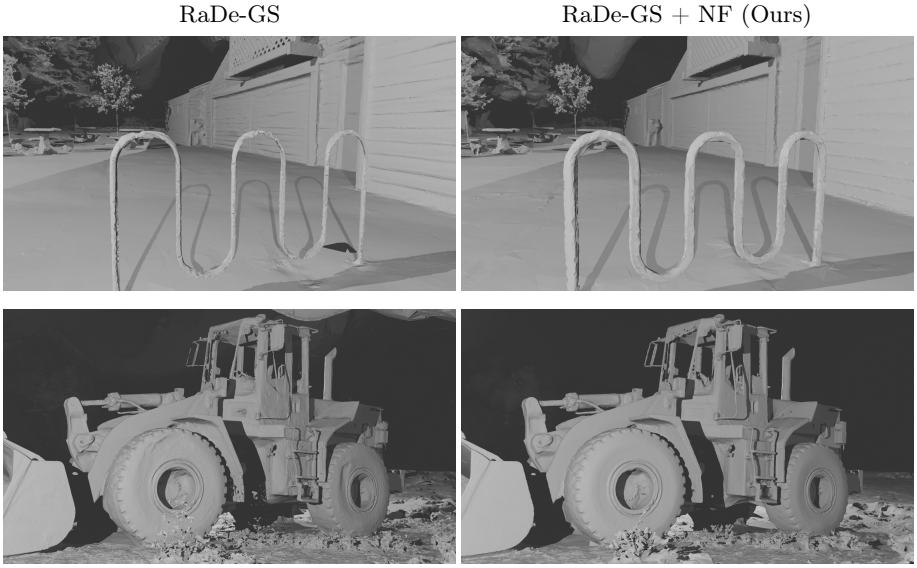


Fig. 15: Qualitative effect of Normal Field (NF) supervision on RaDe-GS. Adding our NF losses and densification to RaDe-GS yields more faithful geometry. For instance, we mititage the erosion of the bike lock in the Barn scene, and avoid holes in the wheel of the Caterpillar scene. This shows our approach can be used to improved other existing methods.