Laplace-Beltrami Operator for Gaussian Splatting

Hongyu Zhou[1,2]   Zorah Lähner[1,2]
[1]University of Bonn   [2]Lamarr Institute for Machine Learning and Artificial Intelligence
{hzhou, laehner}@uni-bonn.de   lamarr-institute.org

Abstract

With the rising popularity of 3D Gaussian splatting and the expanse of applications from rendering to 3D reconstruction, the need for geometry processing methods tailored directly to this representation becomes increasingly apparent. While existing approaches convert the centers of Gaussians to a point cloud or mesh to use them in existing algorithms, this conversion might discard valuable information present in the Gaussian parameters or introduce unnecessary computational overhead. Additionally, Gaussian splatting tends to contain a large number of outliers that, while not affecting the rendering quality, need to be handled correctly to not produce noisy results in geometry processing applications. In this work, we present a novel framework that operates directly on Gaussian splatting representations for geometry processing tasks. Our work introduces a graph-based outlier removal designed for Gaussian distributions as well as a formulation to compute the Laplace-Beltrami operator, a widely used tool in geometry processing, directly on Gaussian splatting. Both use the Mahalanobis distance to account for the anisotropic nature of Gaussians. Our experiments show superior performance to the point cloud Laplacian operator and competitive performance to the traditional Laplacian operator computed on a mesh, while avoiding the need for intermediate representation conversion.

Website*: zero-4869.github.io/LBO4GS*

1. Introduction

3D Gaussian Splatting (3DGS) [20] has recently revolutionized 3D scene representations. By representing complex scenes as a set of 3D Gaussians, it achieves photorealistic results for novel view synthesis while allowing much more efficient training and real-time rendering compared to NeRF-based approaches [2]. Even though it was originally proposed for rendering applications, it is now also popular as a general-purpose 3D representation [14, 16]. Despite its success in rendering, most 3DGS methods face fundamental challenges when applied to geometry processing tasks. 3DGS optimization is usually about visual fidelity but fails to produce geometrically accurate surfaces, leading to recent work focusing on surface reconstruction instead [11, 16]. There are two main strategies in this direction: methods that pull the 3D Gaussians to the suspected surface and then reconstruct a mesh via Poisson reconstruction [15, 19] or TSDF [9] for regularization, and dimensionality-constraint approaches that restrict the Gaussians to be 2-dimensional which captures the 2-manifold structure of 3D objects [16] .

As the geometric reconstruction accuracy improves, the need for traditional geometry processing tools that operate natively on Gaussian splatting arises. Recent work has demonstrated the potential direct manipulation through deformation energies [17] and semantic segmentation [8] on Gaussian splats. In this paper, we will expand this direction with a traditional operator from geometry processing, particularly the Laplace-Beltrami operator (LBO), which is often called the "Swiss Army knife" of geometry processing. The LBO serves as an important part in applications ranging from mesh smoothing and deformation to spectral analysis and shape matching [25, 32, 36].

One core information in the LBO is the connectivity information between points, which is by design missing in Gaussian splatting. A possible solution is to convert the 3DGS representation into a mesh, but this increases vertex count and computational overhead while often producing both over-smoothed details and noisy artifacts. Alternatively, treating Gaussians as points clouds by only using their centers discards valuable geometric information encoded in the covariance matrices and suffers from poor performance due to the high density of outliers. The outliers stem from the image-based optimization that naturally does not depict the inside of objects, and low-opacity Gaussians that contribute minimally to the rendering quality but add noise to the point cloud. As a consequence, point cloud methods fail to capture the geometric structure implicitly encoded in the Gaussian splatting representation.

We propose a novel framework that enables direct geometry processing on Gaussian splats without conversion into an intermediate representation, making the process lighter and more flexible. The two key aspects are: (1) a graph-based outlier removal strategy specifically designed for Gaussian

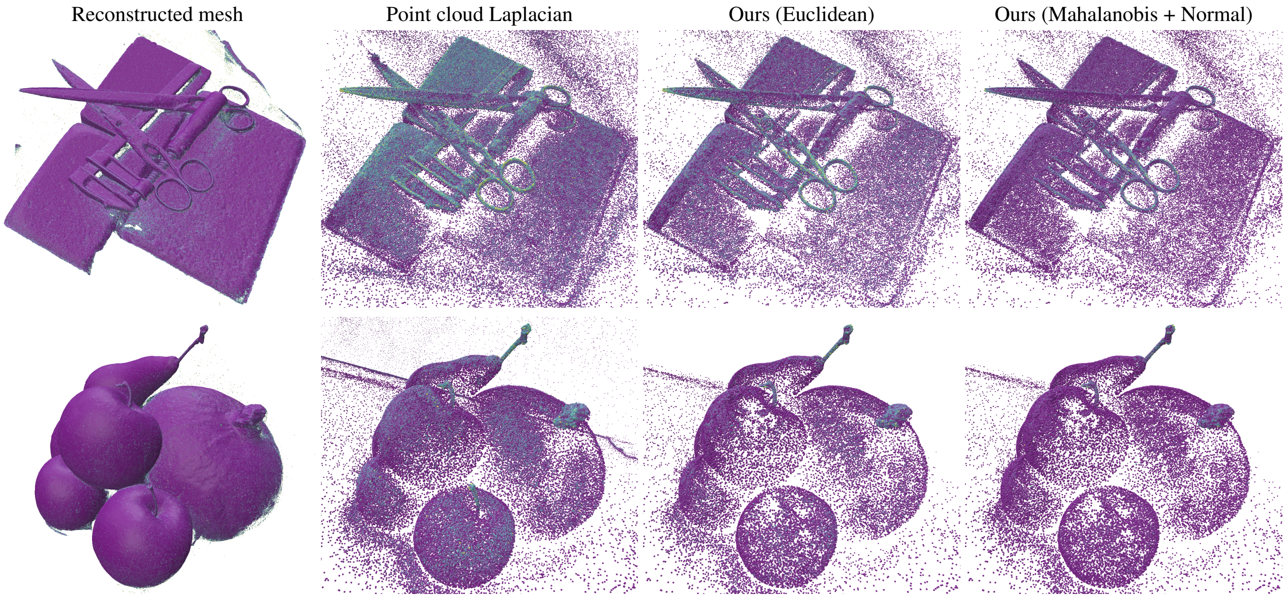

Figure 1. Curvature computation using the Laplace-Beltrami operator of the mesh extracted from the 3DGS, the raw Gaussian center point cloud, our filtered result with Euclidean distance and our proposed method and filtering with Mahalanobis distance and normal direction on the DTU dataset (best view zoomed in). Purple indicates low curvature and turquoise to yellow higher curvature. (Top row) Our method does not overestimate curvature in areas with many outliers as the point cloud Laplacian or the pure Euclidean neighborhood does. (Bottom row) Even a clean-looking extracted mesh will add high errors close to mesh boundaries which our method does not suffer from. The resulting curvature function of our approach also represents the smooth surface better.

distributions, and (2) an LBO formulation that leverages the Mahalanobis distance to incorporate covariance information as surface cues. This approach is computationally efficient while fully utilizing the implicit geometric information encoded in 3D Gaussian splatting. Fig. 1 shows the advantages on the application of curvature computation.

Contributions. Our core contributions are as follows:

1. A definition of the Laplace-Beltrami operator that is computed directly on the Gaussian splatting representation and can fully leverage the geometric information encoded by the covariance.
2. An effective graph-based outlier removal method that is tailored for Gaussian distributions and preserves geometric structure while reducing noisy artifacts.
3. We experimentally demonstrate the effectiveness of this LBO formulation on a variety of geometry processing applications, including geodesic distance computation and shape matching, showing clear improvement over mesh-based and point cloud-based baselines.

2. Related Work

In this section, we show connections to the most relevant related work. A broader introduction into Gaussian splatting can be found in the survey of [10].

2.1. Gaussian Splatting

3D Gaussian splatting was introduced in [20] as an efficient framework for novel-view synthesis. It optimizes over a set of Gaussian distributions in 3D space with color and opacity values which can be rendered from new view points. While this works exceptionally well and has been applied to many applications [30, 37], the pipeline is focused on clean-looking rendering results but not clean geometry. To overcome this, several methods introduced additional constraints that focus on the geometric accuracy in the optimization. For example, the method of [16] restricts the variance to a 2D plane (the dimension of the surface). Another approach, SuGaR [15], extracts a mesh and realigns the Gaussian splats with the surface of this mesh to obtain a cleaner geometry. Gaussian opacity fields [39] proposes an approximation of the surface normals and incorporates normal regularization for better geometry. GSDF [38] and 3DGSR [22] combine Gaussians with a signed distance function to further regularize the results with geometric losses. PGSR [9] proposes a better estimation of the depth map and normal map, which allows single-view and multi-view regularization. While these methods lead to better geometric accuracy, the extracted mesh is often noisy and/or overly smooth, and typically contains a large number of vertices.

2.2. Laplace-Beltrami Operator

Laplacian Operator on Meshes. The Laplace-Beltrami operator (LBO) is the generalization of the second derivative on general manifolds and a popular tool in many geometry processing applications. Its discretization, especially on triangular meshes, has been studied extensively and it has been shown that not all properties of the continuous Laplacian can be fulfilled in the discrete case at the same time [35]. While it is possible to use the graph Laplacian [33] on a mesh by discarding the face information, this fails to take into account all information about the local geometry. More advanced mesh LBOs, like the cotan discretization [23, 28] or intrinsic Delaunay discretization [5], provide a more accurate approximation of the continuous behaviour. These can also be extended to more complex domains, like n-dimensional data [12], general polygonal meshes [1, 7], or non-manifold meshes [31]. However, all of these depend on explicitly given connectivity information which can guarantee certain properties but does not work for less structured shape representations, like point clouds or Gaussian splats.

Laplacian Operator on Point Clouds. As the discrete Laplacian operator relies on the definition of the neighborhood that is not explicitly given in the point cloud, it is essential to estimate the neighborhood function in a good way so that it approximate the intrinsic connectivity. A straightforward solution would be to triangulate the point cloud, however, this is expensive and often leads to errors on sparse or noisy point clouds. Instead, a common solution is to *locally* approximate the surface via its tangent plane and projection of surrounding points onto it, often by a nearest neighbor search [4]. This works well on smooth or flat regions, but struggles around very sharp features. The resulting inaccuracies can be diminished by building an operator that is robust to incorrectly found and non-manifold surface connections [31], or by employing improvements in the surface estimation for these cases, for example through anisotropic Voronoi diagrams [29], or physic dynamics [27]. The recent work of [26] avoids direct estimation of the surface by learning the behavior of the LBO on different examples and then generalizing the behavior directly to new point clouds. While the centers of Gaussian splats do form a point cloud on which the previous methods can be applied, the directional variance at each point provides valuable additional information about the surface. In this work we propose a more accurate way to extract the LBO on Gaussian splats which include the variances in the surface estimation.

3. Background

3.1. Gaussian Splatting

3D Gaussian splatting [20] represents a scene as a collection of 3D Gaussian distributions, where each Gaussian is parametrized by $\{(\mu_i, \Sigma_i, \alpha_i, c_i)\}_i$ with a mean $\mu_i \in \mathbb{R}^3$, covariance matrix $\Sigma_i \in \mathbb{R}^{3\times3}$, opacity α_i, and color function c_i encoded in spherical harmonics. Each Gaussian represents a local region of the scene with the mean defining its spatial location, the covariance matrix encoding its 3D shape and orientation, and the opacity and color function contributing to the final rendering. This collection can be easily projected to 2D and rendered by accumulating density along a ray. The optimization process aims to match the render of these parameters to a collection of images from different view points which leads to bias towards photometric consistency instead of geometric accuracy, leading to Gaussians that may not be localized exactly on the object surface but still produce visually compelling results.

The rendering-focused optimization creates several challenges for geometry processing applications. First, many Gaussians may be positioned in free space or within object interiors if they contribute to visual quality. Second, low-opacity Gaussians that minimally affect rendering quality are retained in the representation, creating outliers for geometric analysis. Recent work has addressed these limitations through surface regularization using signed distance functions [15], depth and normal supervision [39], and constraining the Gaussian covariance to be 2-dimensional and aligned to a surface [16].

3.2. (Discrete) Laplace-Beltrami Operator

The Laplace-Beltrami operator (LBO) $L = \text{div} \cdot \nabla$ generalizes the second derivative to general closed compact manifolds. This operator, along with its eigenfunctions ϕ_i and eigenvalues λ_i which are non-trivial solutions to $L\phi_i = \lambda_i\phi_i$, are popular tools in geometry processing, enabling applications like spectral analysis and functional maps (see Sec. 2.2).

When discretizing the underlying manifold, for example as a mesh or point cloud, the continuous LBO must be approximated as well which leads to discretization artifacts [35]. For the triangular meshes, the cotan discretization [28] is the most widely used version. It defines the stiffness matrix as

$$W_{ij} = \begin{cases} \frac{1}{2}(\text{cot}\alpha_{ij} + \text{cot}\beta_{ij}), & \text{if } (i,j) \in E \\ -\sum_{k\in\mathcal{N}(i)} w_{ik}, & \text{if } j = i \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

where α_{ij}, β_{ij} are the opposing angles to the edge between vertices v_i, v_j (see Fig. 2) and $\mathcal{N}(i)$ the set of vertices in the 1-neighborhood of vertex i. The discrete LBO can the be computed as $L = M^{-1}W$, where M is a diagonal mass matrix, describing the local weight at each vertex M_{ii} as the area of the voronoi cell around vertex i. This discretization of the Laplacian does not necessarily fulfill the maximum principle [35], but will if applied the intrinsic Delaunay triangulation of the mesh [5]. However, this can only be computed clean, pure triangular meshes without non-manifoldness (e.g., edges with three triangles attached). For meshes of arbitrary topology, [31] suggested to use the tufted cover, which generates an implicit manifold overlay of a given connectivity, in combination with the intrinsic triangulation. Due to its flexibility with respect to the connectivity, the tufted Laplacian is well-suited to be used on point clouds by approximating a local neighborhood and connectivity for each point without the need to reconstruct a full, consistent triangle mesh, which is expensive and prone to be noisy. We will use this property but extend the neighborhood definition to be explicitly designed for Gaussian distributions.

4. Method

We propose a discretization of the Laplace-Beltrami operator (LBO) that operates directly on 3D Gaussian splatting representations, preserving encoded geometric information without requiring a conversion to an intermediate format. Our approach addresses two fundamental challenges: constructing an appropriate local neighborhood in the absence of explicit connectivity information, and handling noisy outliers that arise from rendering-focused optimization. First, we introduce a graph-based filtering procedure that leverages symmetric Mahalanobis neighborhoods to remove geometric outliers while preserving surface structure, see Sec. 4.1. Second, we formulate an Laplace-Beltrami operator that is designed to use the full covariance information of the Gaussians to capture local geometric relationships, see Sec. 4.2.

4.1. Graph-based Outlier Removal

3D Gaussian splatting optimization often generates outliers that only contribute minimally to the rendering quality but significantly degrade geometric accuracy. While the original 3DGS [20] already proposed to remove near-zero opacity splats during training, many low-information Gaussians persist inside the objects and close the surface [39]. Due to their low opacity or position inside of closed objects, these splats can take high variance forms in all direction without changing the rendering result. We propose to use a symmetric Mahalanobis neighborhood construction that can detect such outliers.

Mahalanobis Distance. The Mahalanobis distance is a measure of the distance between a point $p \in \mathbb{R}^d$ and a distribution $\mathcal{G}$ with mean $\mu \in \mathbb{R}^d$ and covariance $\Sigma \in \mathbb{R}^{d \times d}$. We only consider $d = 3$ in this paper. It is defined as

$$d_M(p, \mathcal{G}) = \sqrt{(p-\mu)^T \Sigma^{-1} (p-\mu)}.$$

See Fig. 2 for a visualization and relation to Euclidean distance. The metric captures the anisotropic structure of Gaussians, where surface-aligned splats typically exhibit high variance along tangent directions and low variance in normal direction. Thus, a neighborhood based on the Mahalanobis distance clusters together points that form a coherent surface while assigning outliers a higher distance.

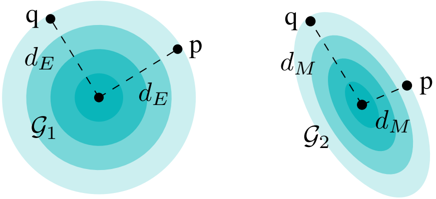

Figure 2. Difference between the Euclidean distance (left) and Mahalanobis distance (right). p and q both have the same distance to distribution $\mathcal{G}_i$. The Euclidean distance is a special case of the Mahalanobis distance if the covariance is equal in all directions while the general Mahalanobis distance weights directions differently based on the directional variance.

However, we can only compute the distance between a *point* and a distribution. What we actually want to do is compute the distance between two Gaussian distributions. Computing the distance between two distributions is possible, for example, with the earth mover's distance, but computationally very expensive. In practice, we approximate by taking a symmetric approach, computing the Mahalanobis distance between center points and surrounding Gaussian distributions, and building neighborhoods between splats where the center points are both each other's Mahalanobis neighbors.

Graph Construction We compute the following neighborhood graph for each Gaussian center: (p_i, p_j) is an edge if and only if p_i is among the k-nearest Mahalanobis neighbors of p_j **and** p_j is among the k-nearest Mahalanobis neighbors of p_i. This bidirectional constraint leads to a robust surface connectivity while naturally isolating outliers. The resulting graph typically contains multiple connected components due to interior outliers and noise. We choose only the largest connected component and prune all Gaussians belonging to others. This filtering leads to a sparser and cleaner representation of the geometry, see Fig. 3 for a visualization of the effect. In addition, Sec. 5.4 shows experimentally that this efficient processing during training leads to a very compact and clean representation. For larger scenes, that are expected to have multiple disconnected but relevant components, it is possible to only prune components below a threshold size.

4.2. Gaussian Laplacian

The LBO encodes local connectivity and geometry. Gaussian splats do not encode connectivity explicitly but the covariance implies the surface geometry. The common point cloud Laplacian [31] approximates the surface by using Eu-

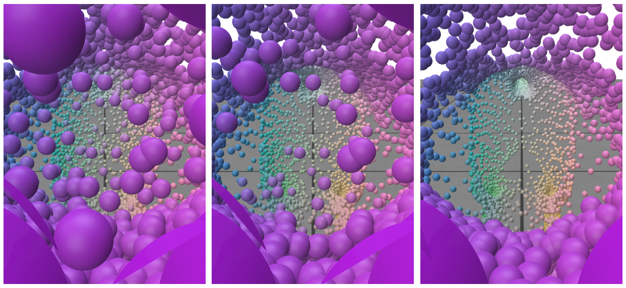

Figure 3. Example of the inside of the point cloud before (left), after (middle) cleaning using the training from GOF [39] and using adaptive training (right).

clidean neighborhoods of each point. We extend this idea to leverage the full geometric information encoded in the Gaussian covariance matrices. Instead of Euclidean distance, we construct k-nearest neighbor relationships using the Mahalanobis distance, enabling the neighborhood structure to respect the anisotropic geometry encoded in each Gaussian. This approach naturally weights connections based on both spatial proximity and geometric compatibility.

We adapt the robust point cloud Laplacian formulation to operate on our cleaned Gaussian centers with Mahalanobis-based connectivity. The resulting operator inherits the mathematical properties of the non-manifold and point cloud Laplacian [31] while incorporating the geometric information of Gaussian covariance matrices. Sec. 5 experimentally validates these changes.

Normal Estimation. The non-manifold Laplacian [31] contains a normal estimation step from the point neighborhood. This estimation is strongly effected by outliers and noise and for the Gaussian Laplacian we replace it by estimating the normal of each point as the eigenvector corresponding to the smallest eigenvalue of the covariance matrix of the corresponding Gaussian distribution. This change is applicable under the assumption that the Gaussian splats are mostly flat and well aligned to the surface, as is the case in methods such as SuGaR [15], GOF [39] and PGSR [9].

Adaptive Training of 3DGS We integrate our outlier removal directly into the 3DGS training process to improve geometric quality. Specifically, we filter the Gaussian splats every $N = 3000$ iterations by removing all except the $k = 1$ biggest components which prunes all outliers. k can be increased for larger scenes. Note that finding all connected components can be achieved by breadth-first search in complexity $O(N)$, hence the extra cost of computation is not significant during training.

5. Experiments

We show the effectiveness of our method through comprehensive evaluation against established baselines. Our Gaussian Laplacian is compared against the robust point cloud Laplacian [31] as a baseline and the cotan Laplacian [28] on meshes, either on the ground-truth mesh if it exists or on meshes extracted from 3DGS results if it does not exist. Since common 3DGS datasets lack ground-truth meshes due to their focus on rendering, they are not suitable for quantitative evaluation. Instead, we turn to the datasets that are commonly used in geometry processing for quantitative evaluation, see Sec. 5.1. Our experiments demonstrate superior performance when computing heat diffusion and geodesics in heat (Sec. 5.2), competitive results in shape correspondence (Sec. 5.3), and significant improvements in the compactness and cleanliness of 3DGS representations when our outlier detection was integrated in the training process (Sec. 5.4). Gaussian splatting representations are generated using either [39] or [9] for all experiments, depending which works better on each dataset. All experiments were done with an NVIDIA RTX 4090 with 24 GB RAM.

5.1. Evaluations and Comparisons

Dataset. We generated a dataset for evaluation from TOSCA [6], which provides 8 classes of non-rigidly deformed meshes with ground-truth correspondence and at least 3 different poses in each class. Since common 3DGS datasets contain neither ground-truth meshes nor correspondences, we use this to provide reliable quantitative evaluations. We used Blender to render an image dataset of TOSCA suitable for 3DGS optimization, generating 100 training images, 50 validation images, and 30 test images per shape across 5 poses per class (or maximum available). Objects are scaled by $\frac{1}{20}$ to fit into a default camera frame. We also experiment on Shiny Blender [34], an object-centric dataset that contains test views for the evaluation of novel view rendering with reflective surfaces, to show the effectiveness of the adaptive training. Qualitative results on data commonly used for 3DGS (DTU [18] and TnT [21]) can be found in Fig. 1 and the supplementary.

Evaluation Metrics. Since the LBO matrices across different 3D representations cannot be compared directly due to varying discretization, e.g. ordering and amount of vertices, we evaluate derived properties of the operators to compare performance. Derived point-wise functions, like curvature, can not be compared directly under varying discretizations, either. In that case, we project each point to its Euclidean nearest neighbor on the ground-truth mesh and compare with this value. The evaluation is done with the eigenvalues, a curvature loss, the geodesic distances and shape matching accuracy.

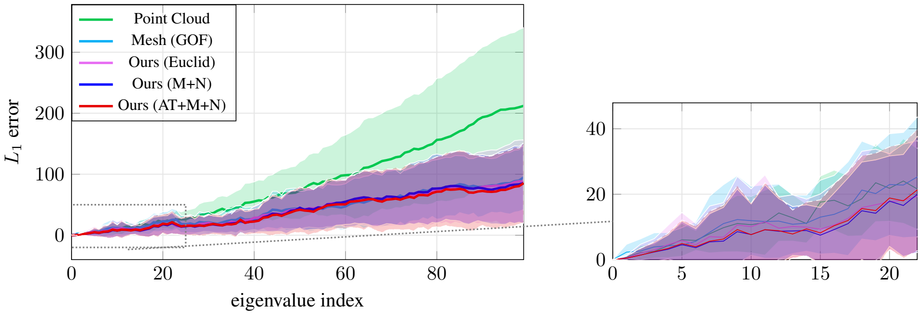

Figure 4. Statistical analysis over all the objects on the loss of eigenvalues. The index represents the order of the eigenvalue by magnitude. The line represent the average of the loss and one std is used for the confidence interval. Our method outperforms the Mesh (GOF).

Category	Mesh (GT)	Point Cloud	Mesh (GOF)	Ours (Euclid)	Ours (M+N)	Ours (AT+M+N)
Cat	0.017	0.045	0.028	0.034	0.027	**0.026**
Centaur	0.018	0.036	0.024	0.023	0.023	**0.022**
David	0.036	0.050	0.058	0.038	**0.035**	**0.035**
Dog	0.016	0.050	0.037	0.036	**0.031**	0.032
Horse	0.016	0.046	0.029	**0.022**	**0.022**	**0.022**
Michael	0.038	0.051	0.065	**0.046**	0.047	**0.046**
Victoria	0.035	0.117	**0.078**	0.113	0.095	0.082
Wolf	0.012	0.036	**0.010**	0.017	0.018	0.018
Total	0.025	0.056	0.045	0.045	0.040	**0.038**

Table 1. Average error in geodesic computation $\mathcal{E}_{geo}$ by Geodesics in Heat [13] in comparison to the exact distance on the ground truth mesh. The bold represents the best score and the gray represents the second best. Note that Mesh (GT) represents the approximation error introduced by Geodesics in Heat on the ground-truth mesh as a reference and is not a competitor as the ground truth is not known in general.

Competing Methods. We compare our method with the following baselines since no other direct approaches that define the LBO for Gaussian splatting representations exist:

- **Point Cloud (PC):** the robust point cloud Laplacian computed on centers of the 3D Gaussian splats [31],
- **Mesh (GOF):** the tufted Laplacian [31] on a mesh reconstructed from the state of the art 3DGS approach Gaussian opacity fields [39],
- **Ours (Euclid):** our proposed Gaussian Laplacian using the k-nearest Euclidean neighbors to compute the normal direction (as in [31]), applied on the outlier-filtered 3DGS (very similar to the robust point cloud Laplacian on the outlier-filtered point cloud),
- **Ours (M+N):** our proposed Gaussian Laplacian using the k-nearest Mahalanobis neighbors and lowest magnitude variance direction as the normal, applied on the outlier-filtered 3DGS (Sec. 4.1),
- **Ours (AT+M+N):** using the k-nearest Mahalanobis neighbors and lowest magnitude variance direction as the normal, applied on the 3DGS optimized by our proposed adaptive training (Sec. 4.2).

If given, we compute the Laplace-Beltrami operator on the ground-truth meshes as the optimal solution, and we use the state-of-the-art tufted Laplacian [31] to compute the Laplacian operator on both point clouds, meshes and our adaption.

5.2. Experimental Results

We present a series of experiments showing the performance of our proposed Gaussian Laplacian in comparison to other baselines approximating the LBO on 3DGS.

Eigenvalues. Following the convergence analysis of Belkin and Niyogi [3] for the spectrum of the LBO, we compare the eigenvalues of the normalized Laplacian matrix by solving $Lx = \lambda Mx$ and computing the first K eigenvalues (the lower the more stable). Since the eigenvalue scales inversely proportional to surface area, we calculate the normalized difference $S|\lambda_i - \lambda_i^{gt}|$ for $i = 1, \ldots, 100$ for comparison.

We take the first $K = 100$ eigenvalues of each Laplacian operator and compute the difference to the operator on the ground-truth mesh. A shown in Fig. 4, our method and Mesh (GOF) perform on-par, and both outperform the point cloud-based Laplacian. For the first 20 eigenvalues, which should be the most stable, our method is more accurate than even the mesh-based Laplacian and, compared to using Euclidean distance, the Mahalanobis distance makes the spectrum of the Laplacian operator closer to the ground truth.

Curvature As an intrinsic property, mean curvature can be derived from the Laplace-Beltrami operator via the equation

Dataset	Mesh (GT)	3DGS	Mesh (reconstructed)	Ours (Outlier Removal)	Ours (Adaptive Training)
TOSCA	30416	19394	197364	17763	14523
Shiny Blender	/	169877	3546714	/	99453

Table 2. Comparison of the average size of different representations. For meshes we report the number of vertices. For 3DGS we report the number of splats. We use GOF to obtain the 3DGS and reconstructed mesh on TOSCA, and use PGSR on Shiny Blender. The number of vertices in the reconstructed mesh is ten times more than that of 3D Gaussian splatting while still performing worse in several quantitative evaluations, see Sec. 5.

Method	Car	Coffee	Helmet	Teapot	Toaster	Mean
3DGS (PGSR)	26.43	30.86	25.72	37.08	20.24	28.07
Ours (Adaptive Training)	26.21	30.82	25.67	36.96	20.04	27.94

Table 3. Comparison of the rendering quality with and without adaptive training on the Shiny Blender dataset [34] measured by PSNR. In the end, adaptive training does not notably degrade the rendering quality but is able to remove a significant amount of outliers.

$\Delta p = -2Hn$, where p denotes the point coordinates, H the mean curvature and n the unit normal vector. Tab. 4 demonstrates that our Gaussian Laplacian with adaptive training estimates curvature much closer to the ground truth (based on the ground-truth mesh) than all other alternatives. Surprisingly, the curvature error is largest for the extracted mesh. This is due to artifacts produced in the extraction process to which curvature computation is highly sensitive. Behavior like this indicates that avoiding unnecessary representation conversion is not always a good choice and using representation-native operations has great merit.

Type	Mesh (GOF)	PC	Ours (M+N)	Ours (AT+M+N)
Avg.	43.049	17.746	16.317	**16.023**
Min.	12.720	7.954	6.332	**6.202**
Max.	129.369	33.621	29.867	**29.306**

Table 4. L_1-error of the mean curvature computed from the Laplacian operator on an extracted mesh, point cloud, ours with Mahalanobis distance and ours with adaptive training.

Geodesic Distance The geodesic distance describes the length of the shortest path between two points on a surface. It can be efficiently approximated using Geodesics in Heat [13] which is based on heat diffusion, a function describing how heat diffuses over time that can be computed using the Laplacian operator. As ground truth, we use an exact algorithm [24] for geodesic distances on the ground-truth meshes and compare the approximation quality of each method to this function.

Since computing exact geodesic distances is expensive, we uniformly sample 100 source points on the ground-truth mesh and use only the distance functions of these points. We compute the error normalized by surface area:

$$\mathcal{E}_{geo}(\Delta) = \frac{1}{100}\frac{1}{\sqrt{S}}\sum_{i=1}^{100}\sum_{j=1}^{N}|d^j - d^j_\Delta|$$

where S is the surface area, N is the number of points of the ground-truth mesh, d^j is the exact geodesic distance [24] from the point j to the source point and d^j_Δ is the computed geodesic distance from j to the source point using the LBO Δ. Tab. 1 shows the comparison of geodesic distance errors averaged over all 100 distance functions. The Mesh (GT) column is given as a reference to see the error induced by Geodesics in Heat algorithm to the exact geodesics on the ground-truth mesh, not as a direct competitor. Our method significantly outperforms the point cloud Laplacian in all cases and outperforms the extracted mesh-based computation in almost all cases. In cases where the adaptive training does not improve the results, it is still on-par with the best results. Fig. 6 shows qualitative results.

5.3. Shape Matching

A fundamental problem in geometry processing is finding correspondences between pairs of isometrically deformed 3D shapes. We evaluate correspondence quality in the functional maps framework [25], which uses the Laplace-Beltrami eigenfunctions for dimensionality reduction of the problem, by testing preservation of the ground-truth correspondence after the representation change.

Given the ground-truth correspondence as a permutation P, the functional map representation is computed as

$$C = \Phi_t^\top M_t P \Phi_s \tag{2}$$

where Φ_s, Φ_t are the metric of the first k (ordered by frequency) stacked eigenvectors of the source and target shape, respectively, and M_t the mass matrix on the target shape. The point-wise correspondence Q can be extracted again via nearest neighbors in the aligned spectral space between the row-wise entries of Φ_t and $C\Phi_s$, see [25] for detailed explanation. Since we do not want to evaluate a specific matching algorithm, this projection of the ground truth, which still leads to errors due to the dimension reduction, gives us

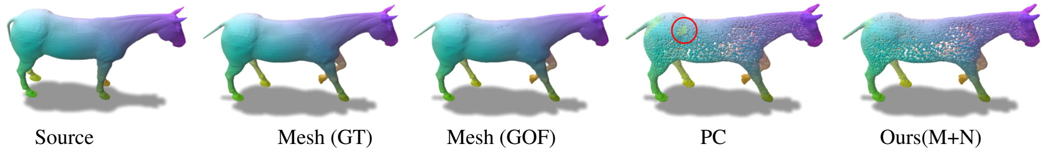

Figure 5. Color map for shape matching in the horse category. The source map (center) is colored by coordinate and the colors transferred with the correspondence computed by each method to the surrounding shapes. Inconsistencies in color indicate problems with the correspondence.

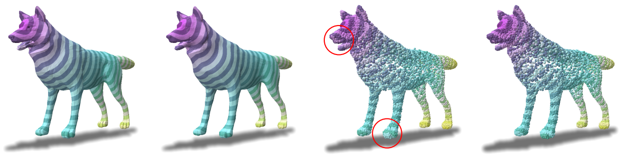

Figure 6. Geodesics in Heat [13] computed on (from left to right) the ground-truth mesh, the reconstructed mesh (GOF), the center point cloud (uncleaned) and the cleaned Gaussian splats (ours). The cleaning significantly improves the results (see red circles).

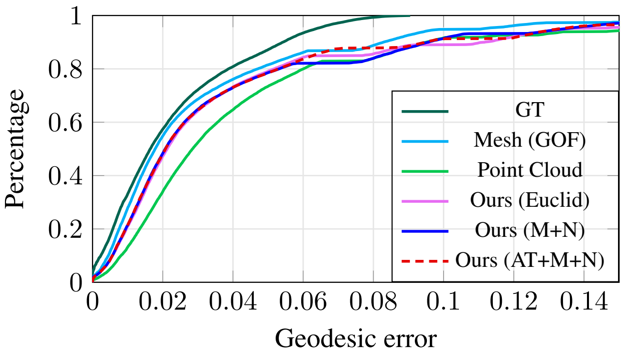

Figure 7. Cumulative geodesic error curve after representing the ground-truth correspondence in a low-dimensional functional map matrix, averaged over all the categories. The x-axis shows the percentage of shape diameter and y-axis percentage of matches with lower error. Dashed only for visibility.

a measure of quality for the eigenbasis and, in turn, our Laplace-Beltrami operator.

Using 100 eigenfunctions on 1000 uniformly sampled vertices, we compute the geodesic error of each Laplacian on the TOSCA dataset, which provides ground-truth correspondences. Again, points are projected onto their nearest neighbor when discretizations are not compatible:

$$\mathcal{E}_{corr}^{(s,t)}(p) = \frac{1}{\sqrt{S^t}} \text{dist}_{geo}(P(p), Q(p)) \tag{3}$$

where p is a vertex on the mesh of the source object, $P(p)$ is the ground-truth correspondence of p (projected onto the target), and $Q(p)$ is the predicted correspondence of p on the target. Fig. 7 shows the cumulative error curve computed from $\mathcal{E}_{corr}^{(s,t)}$ on all intra-class pairs. In this experiment, our Laplacian does outperform the point cloud Laplacian again, but the extracted meshes from GOF perform slightly better. This could be due to smaller effects of artifacts in the mesh from the dimensionality reduction in functional maps. Qualitative results can be found in Fig. 5.

5.4. Adaptive training

Remarkably, this training scheme leads to clean geometry with nearly no outlier splats in the geometry (see Fig. 3) while maintaining the shape of the object. In addition to having the highest quality geometry in the sense of the spectrum of Laplacian operator, the adaptive training produces a significantly more compact representation (in terms of number of points) compared to the reconstructed meshes, the Gaussian splats from GOF, the post-processed 3D Gaussian splats, and even the original meshes, see Tab. 2. Moreover, the rendering quality is not notably degraded using adaptive training, see Tab. 3.

6. Conclusion

We have introduced a new Laplace-Beltrami operator that operates directly on 3D Gaussian splatting representations without conversion into intermediate formats. Our formulation is based on a Mahalanobis distance neighborhood computation which preserves the connectivity information implied by the anisotropic covariance matrices of each splat. Our method addresses a fundamental gap in geometry processing for Gaussian representations. Experimental validation demonstrates superior performance compared to point cloud approaches and competitive results with mesh-based methods across eigenvalue analysis, curvature estimation, geodesic computation, and shape correspondence. The direct computation eliminates mesh reconstruction overhead while our adaptive training produces remarkably compact and clean representations. We hope this will lead to a broader adaption of 3DGS beyond rendering in computer vision and graphics applications.

Limitations. The quality of our LBO depends on the assumption that the splats are mostly flat and aligned to the surface. This is only the case for some 3DGS implementations but due to the wider adaption of geometry priors not an unreasonable assumption.

Acknowledgements. This research has been funded by the DFG Sachbeihilfe grant LA 5191/2-1.

References

[1] Marc Alexa and Max Wardetzky. Discrete laplacians on general polygonal meshes. *ACM Trans. Graph.*, 30(4), 2011. 3

[2] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5470–5479, 2022. 1

[3] Mikhail Belkin and Partha Niyogi. Convergence of laplacian eigenmaps. *Advances in neural information processing systems*, 19, 2006. 6

[4] Mikhail Belkin, Jian Sun, and Yusu Wang. Constructing laplace operator from point clouds in rd. In *ACM-SIAM Symposium on Discrete Algorithms*, 2009. 3

[5] Alexander I. Bobenko and Boris A. Springborn. A discrete laplace-beltrami operator for simplicial surfaces. *Disc. & Comp. Geom.*, 38(4), 2007. 3

[6] Alex M. Bronstein, Michael M. Bronstein, and Ron Kimmel. *Numerical Geometry of Non-Rigid Shapes*. Springer, 2008. 5

[7] Astrid Bunge, Philipp Herholz, Misha Kazhdan, and Mario Botsch. Polygon laplacian made simple. *Computer Graphics Forum*, 39(2), 2020. 3

[8] Jiazhong Cen, Jiemin Fang, Chen Yang, Lingxi Xie, Xiaopeng Zhang, Wei Shen, and Qi Tian. Segment any 3d gaussians. *arXiv preprint arXiv:2312.00860*, 2023. 1

[9] Danpeng Chen, Hai Li, Weicai Ye, Yifan Wang, Weijian Xie, Shangjin Zhai, Nan Wang, Haomin Liu, Hujun Bao, and Guofeng Zhang. Pgsr: Planar-based gaussian splatting for efficient and high-fidelity surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 2024. 1, 2, 5

[10] Guikun Chen and Wenguan Wang. A survey on 3d gaussian splatting, 2024. 2

[11] Hanlin Chen, Chen Li, and Gim Hee Lee. Neusg: Neural implicit surface reconstruction with 3d gaussian splatting guidance, 2023. 1

[12] Keenan Crane. The n-dimensional cotangent formula. Technical report, Carnegie Mellon University, 2019. 3

[13] Keenan Crane, Clarisse Weischedel, and Max Wardetzky. The heat method for distance computation. *Communications of the ACM*, 60(11):90–99, 2017. 6, 7, 8

[14] Lin Gao, Jie Yang, Botao Zhang, Jiamu Sun, Yujie Yuan, Hongbo Fu, and Yu-Kun Lai. Real-time large-scale deformation of gaussian splatting. *ACM Transactions on Graphics (SIGGRAPH Asia 2024)*, 2024. 1

[15] Antoine Guédon and Vincent Lepetit. Sugar: Surface-aligned gaussian splatting for efficient 3d mesh reconstruction and high-quality mesh rendering. *CVPR*, 2024. 1, 2, 3, 5

[16] Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2d gaussian splatting for geometrically accurate radiance fields. In *SIGGRAPH 2024 Conference Papers*. Association for Computing Machinery, 2024. 1, 2, 3

[17] Yi-Hua Huang, Yang-Tian Sun, Ziyi Yang, Xiaoyang Lyu, Yan-Pei Cao, and Xiaojuan Qi. Sc-gs: Sparse-controlled gaussian splatting for editable dynamic scenes. *CVPR*, 2024. 1

[18] Rasmus Jensen, Anders Dahl, George Vogiatzis, Engin Tola, and Henrik Aanæs. Large scale multi-view stereopsis evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 406–413, 2014. 5

[19] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*. Eurographics Association, 2006. 1

[20] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), 2023. 1, 2, 3, 4

[21] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)*, 36(4): 1–13, 2017. 5

[22] Xiaoyang Lyu, Yang-Tian Sun, Yi-Hua Huang, Xiuzhe Wu, Ziyi Yang, Yilun Chen, Jiangmiao Pang, and Xiaojuan Qi. 3dgsr: Implicit surface reconstruction with 3d gaussian splatting. *ACM Transactions on Graphics (TOG)*, 43(6):1–12, 2024. 2

[23] Mark Meyer, Mathieu Desbrun, Peter Schröder, and Alan H. Barr. Discrete differential-geometry operators for triangulated 2-manifolds. In *Visualization and Mathematics III*, 2003. 3

[24] Joseph SB Mitchell, David M Mount, and Christos H Papadimitriou. The discrete geodesic problem. *SIAM Journal on Computing*, 16(4):647–668, 1987. 7

[25] Maks Ovsjanikov, Mirela Ben-Chen, Justin Solomon, Adrian Butscher, and Leonidas Guibas. Functional maps: a flexible representation of maps between shapes. *ACM Transactions on Graphics (ToG)*, 31(4):1–11, 2012. 1, 7

[26] Bo Pang, Zhongtian Zheng, Yilong Li, Guoping Wang, and Peng-Shuai Wang. Neural laplacian operator for 3d point clouds, 2024. 3

[27] F. Petronetto, A. Paiva, E. S. Helou, D. E. Stewart, and L. G. Nonato. Mesh-free discrete laplace–beltrami operator. *Computer Graphics Forum*, 32(6), 2013. 3

[28] Ulrich Pinkall and Konrad Porthier. Computing discrete minimal surfaces and their conjugates. *Experim. Math.*, 1993. 3, 5

[29] Hongxing Qin, Yi Chen, Yunhai Wang, Xiaoyang Hong, Kangkang Yin, and Hui Huang. Laplace–beltrami operator on point clouds based on anisotropic voronoi diagram. *Computer Graphics Forum*, 37(6), 2018. 3

[30] Zhijing Shao, Zhaolong Wang, Zhuang Li, Duotun Wang, Xiangru Lin, Yu Zhang, Mingming Fan, and Zeyu Wang. SplattingAvatar: Realistic Real-Time Human Avatars with Mesh-Embedded Gaussian Splatting. In *Computer Vision and Pattern Recognition (CVPR)*, 2024. 2

[31] Nicholas Sharp and Keenan Crane. A laplacian for nonmanifold triangle meshes. *Computer Graphics Forum (CGF)*, 2020. 3, 4, 5, 6

[32] Olga Sorkine. Laplacian Mesh Processing. In *Eurographics*, 2005. 1

[33] Gabriel Taubin. A signal processing approach to fair surface design. In *SIGGRAPH*, 1995. 3

[34] Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T Barron, and Pratul P Srinivasan. Ref-nerf: Structured view-dependent appearance for neural radiance fields. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5481–5490. IEEE, 2022. 5, 7

[35] Max Wardetzky, Saurabh Mathur, Felix Kälberer, and Eitan Grinspun. Discrete laplace operators: No free lunch. In *Proc. of Eurographics Symposium on Geometry Processing (SGP)*, 2007. 3

[36] Simon Weber, Thomas Dages, Maolin Gao, and Daniel Cremers. Finsler-laplace-beltrami operators with application to shape analysis. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3131–3140, 2024. 1

[37] Yunzhi Yan, Haotong Lin, Chenxu Zhou, Weijie Wang, Haiyang Sun, Kun Zhan, Xianpeng Lang, Xiaowei Zhou, and Sida Peng. Street gaussians for modeling dynamic urban scenes. In *ECCV*, 2024. 2

[38] Mulin Yu, Tao Lu, Linning Xu, Lihan Jiang, Yuanbo Xiangli, and Bo Dai. Gsdf: 3dgs meets sdf for improved rendering and reconstruction. *arXiv preprint arXiv:2403.16964*, 2024. 2

[39] Zehao Yu, Torsten Sattler, and Andreas Geiger. Gaussian opacity fields: Efficient and compact surface reconstruction in unbounded scenes. *arXiv preprint arXiv:2404.10772*, 2024. 2, 3, 4, 5, 6