# Laplace-Beltrami Operator for Gaussian Splatting

## Supplementary Material

## A. Data Generation

We generate a new data set that has multi-view images together with corresponding camera poses from 3D shapes in TOSCA [6] using Blender. This has the advantage that we can provide clean ground-truth meshes for the scene instead of a noisy reconstruction. The images are split into train, validation and test dataset for each shape with a ratio of 10 : 5 : 3. Thus, we have 100 images for training, 50 for validation and 30 for testing. The camera poses rotate around the Z-axis, and at the same time going from positive value to negative value along the Z-axis in order to cover the whole object.

## B. Implementation Details

**Implementation** We implement the 3DGS model based on Yu et al.'s work [39][1] and Chen et al.'s work [9][2], and the implementation on point cloud/mesh Laplacian is based on Sharp et al.'s work [31][3]. The heat method is implemented based on Crane et al. [13][4]. We add some extensions to adjust the code to Gaussian splatting. We thank the authors for providing these amazing tools.

**Hyper-parameters** To generate the graph, we take the 10 nearest neighbors with respect to the Mahalanobis distance for each Gaussian. To compute the Laplacian operator, we take 30 nearest neighbors to compute the normals and do Delaunay triangulation as it is the default value in the original paper [31].

**Running time** The training time of 3DGS is the same as in GOF [39] or in PGSR [9]. The computation of the Laplacian operator takes 5s for a 3DGS of size 140K, and the runtime is almost proportional to the number of Gaussian splats.

## C. Additional Gaussian cleaning results

We provide additional qualitative results to show how Gaussian cleaning and adaptive learning perform on other shapes in Fig. A.1. It turns out that Gaussian cleaning can remove most of the 'noisy' Gaussian splatting inside the object for various shapes while keeping the Gaussian splatting at the surface, and applying adaptive training can further clean the Gaussian splatting.

[1]https://github.com/autonomousvision/gaussian-opacity-fields.git
[2]https://github.com/zju3dv/PGSR
[3]https://github.com/nmwsharp/robust-laplacians-py.git
[4]https://github.com/nmwsharp/geometry-central.git

## D. Curvature

We compute the curvature in TOSCA (Fig. A.2), DTU (Fig. A.3), and Tanks and Temples(TnT) (Fig. A.4). See Sec. 5.2 for details. Since there are ground truth meshes in TOSCA, we compare the result of our methods with that of the reconstructed mesh and that of the ground truth mesh. For DTU, we compare the result between the reconstructed mesh, our(Euclid) and ours(Mahalanobis + Normal). For TnT, we only show the result of ours(Mahalanobis + Normal) in 4 different views. The results clearly show that the reconstructed mesh from GOF introduces both high frequency noise as well as smoothed out areas which should contain more details which leads to a a noisy curvature computation. Our results are more faithful to the ground-truth.

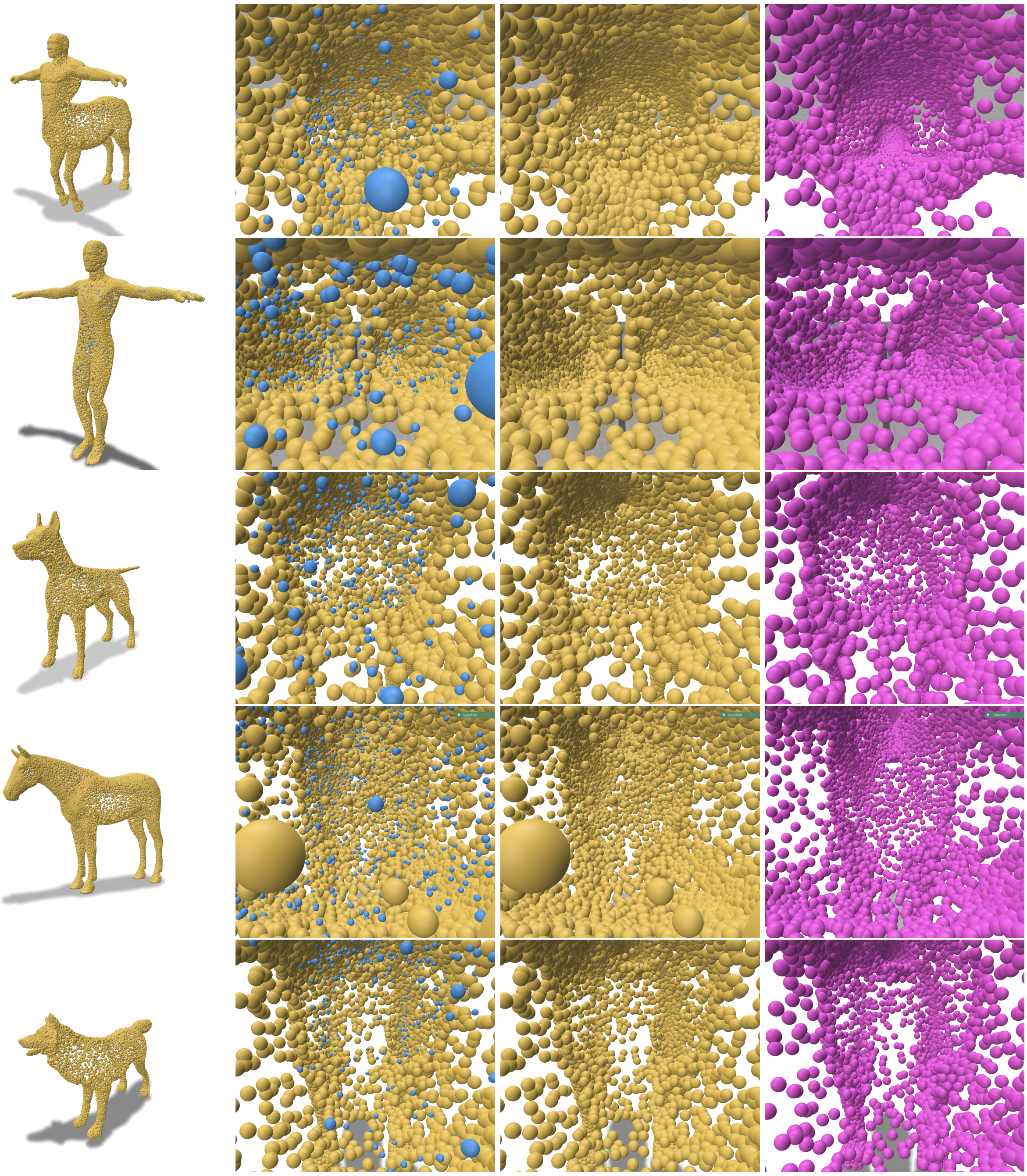

Figure A.1. Gaussian cleaning and adaptive training on other shapes. From left to right: Overview of the object, overlap of 3DGS (GOF)+cleaning, 3DGS after cleaning, and adaptive training. Blue circle denotes center of 3DGS removed. Our method removes almost all outliers on the inside and provides a clear surface structure.

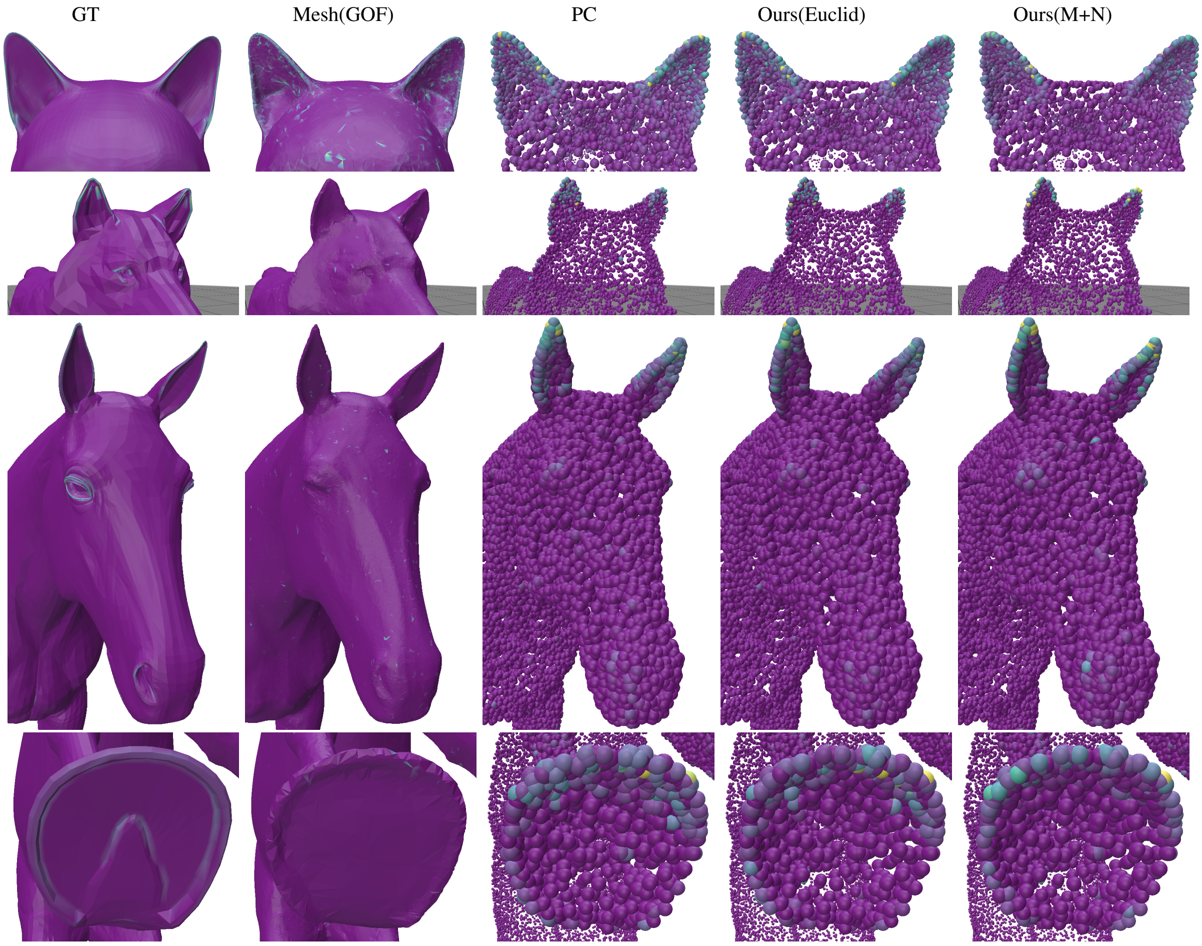


Figure A.2. Curvature computed from Laplace-Beltrami operator. The reconstructed mesh (GOF) has many noise at the surface. Our method using Mahalanobis distance estimates the curvature accurately on high curvature parts (edges) while having less noise for points with low curvature(smooth surface). The figure is visualized using the viridis colormap.

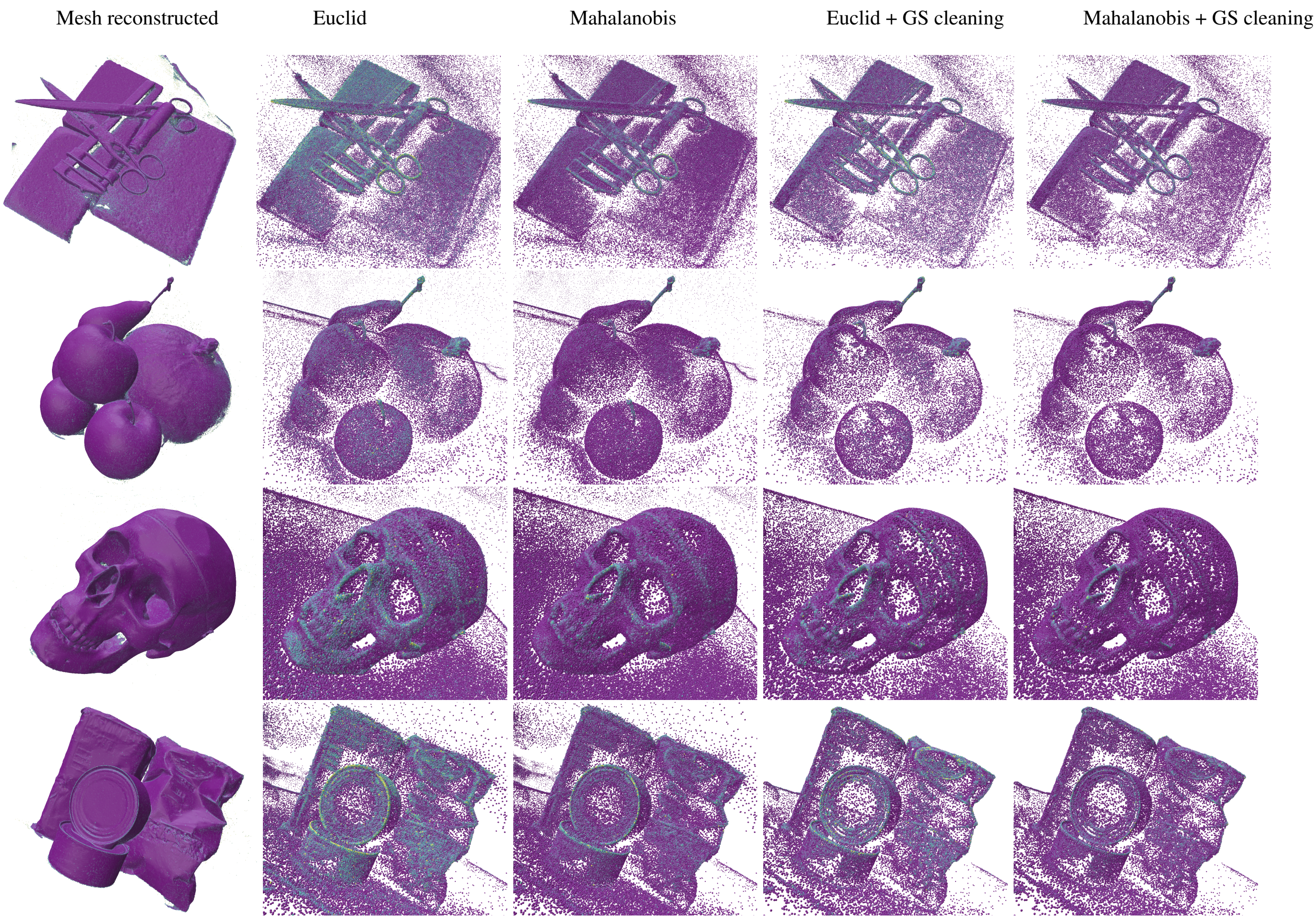


Figure A.3. Curvature computed by Mesh (reconstructed), ours (Euclid) and ours (M+N) on the DTU dataset. Our method using Mahalanobis distance estimates the curvature accurately on high curvature parts (turquoise) while having less noise for points with low curvature (purple). It is especially interesting that the reconstructed mesh introduces very high values at the mesh boundary (2nd row) which is technically incorrect while the Gaussian representation is not affected.

Figure A.4. Curvature computed by ours (M+N) on the Tanks and Temples dataset. The result is visualized in different views. Our method can well estimate the curvature for both the high curvature points lying on the edges and low curvature points lying on the smooth surface. The figure is visualized using the viridis colormap.