7. Supplementary Material

7.1. Filtering Gaussians that are not Surface Aligned

Another issue is that there are some Gaussians that, even with high visual contributions, are not positioned close to surfaces of the scene. The surface of the scene in a particular image can be predicted by calculating the depth of each pixel. A depth map of a Gaussian scene can be rendered as:

$$D = \sum_{j \in P(k)} \sum_{i \in \mathcal{G}_t} T_{ij} \cdot \alpha_i \cdot d_i, \tag{9}$$

where D is the depth map and d is the z-depth of that Gaussian in image space. During the colour rendering process, we calculate the distance between each contributing Gaussian and the calculated depth of that pixel, which quantifies how close each contributing Gaussian is to an object surface. The difference between the predicted surface and Gaussian depth in image space is calculated as:

$$s_{ijk} = \sum_{k \in \text{Cameras}} \sum_{j \in P(k)} \sum_{i \in \mathcal{G}_t} |D_{jk} - d_i|, \tag{10}$$

where s is the surface distance. These values will vary between rendered images, since not all cameras render Gaussians from perspectives where all Gaussians in the pixel are accurately projected onto their closest surface. Hence, we take the minimum calculated surface distance for each Gaussian over all pixels. The final Gaussian surface distance is calculated as:

$$S_i = S_{j^*} \quad where \quad j^* = \arg\min_j s_{ij}, \tag{11}$$

where i is index of the current Gaussian and j is the index of the current pixel. Since we have calculated a minimum surface distance of each Gaussian, we can remove outlier Gaussians that are not deemed close enough to the scene surfaces. Hence, Gaussians with comparatively large surface distance values are culled, with $\sigma = 2$ producing adequate results for large scenes. This process is illustrated in Figure 7.

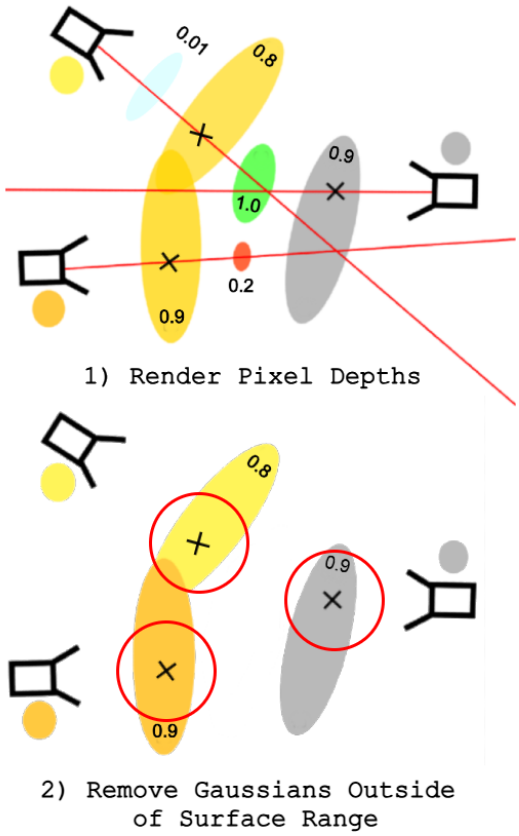

Figure 7. Overview of our approach for filtering Gaussians that are not aligned with the surfaces in the scene. Firstly, the depth of each pixel is calculated during the colour rendering process, as represented by each black cross. Then all Gaussians that have a minimum distance to each pixel depth that is less than a predefined threshold are removed, with these distance ranges being represented by each red ring.

7.2. Ensuring Gaussians are Positive Semi-definite

To correctly sample points from Gaussians, their covariance matrix Σ must be positive semi-definite. This is not a strict requirement for rendering, and as such, 3DGS scenes often contain Gaussians that break this property. We enforce this by regularising each matrix, by adding a small scalar value $\epsilon = 10^{-6}$ to the diagonal elements:

$$\Sigma' = \Sigma + \epsilon I, \tag{12}$$

where I is the identity matrix. Furthermore, we perform eigenvalue decomposition $\Sigma' = Q\Lambda Q^{-1}$, and clamp the eigenvalues in Λ such that $\lambda_i \text{,} max(\lambda_i, \epsilon)$, with $\epsilon = 10^{-7}$. The final covariance matrix is reconstructed as:

$$\Sigma'' = Q\Lambda' Q^{-1}. \tag{13}$$

Any Gaussian with a covariance matrix that is still not positive semi-definite is considered erroneous and removed from point sampling.

7.3. Gaussian Magnitude Scaling Behaviour

Using $\sqrt{SA}$ in equation (5) means that the magnitude scales linearly when the entire Gaussian is uniformly enlarged or shrunk (i.e., all axes are scaled by the same factor). In the Gaussian equation used by 3DGS [19]

$$G(x) = e^{-\frac{1}{2}(x)^T \Sigma^{-1}(x)},$$

it can be shown that uniformly scaling by a factor of s is equivalent to scaling our covariance by s^2:

$$G(\frac{x}{s}) = e^{-\frac{1}{2}(x)^T(s^2\Sigma)^{-1}(x)}, s \in \mathbb{R}^+.$$

It follows then that the eigenvalues are scaled by s^2. Hence our new new semi-axis lengths are scaled by s: $a', b', c' = s\sqrt{\lambda}_1, s\sqrt{\lambda}_2, s\sqrt{\lambda}_3 = sa, sb, sc$. Using our new semi-axes in Equation 6 reveals that our new surface area, SA', is scaled by s^2:

$$SA' = 4\pi \sqrt[p]{\frac{a'^p b'^p + a'^p c'^p + b'^p c'^p}{3}} \quad (14)$$

$$= 4\pi \sqrt[p]{\frac{(s^2)^p(a^p b^p + a^p c^p + b^p c^p)}{3}} \quad (15)$$

$$= s^2 SA. \quad (16)$$

So, $\sqrt{SA'} = s\sqrt{SA}$, which is a more desirable trait for fine detail preserving allotment of points. For completeness, we note that the Gaussian volume would be scaled by a factor of s^3.

7.4. Converting 3DGS-to-PC Point Clouds into Meshes

While 3DGS-to-PC focuses on converting 3DGS scenes into point clouds, our framework also offers capabilities for producing subsequent meshes via predicted surface point generation.

7.4.1. Methodology

As discussed in Section 7.1, our method can determine Gaussians in close proximity to surfaces by calculating the minimum distance of each Gaussian to the predicted pixel depth. Hence, we can determine a small subset of Gaussians that effectively map on predicted surfaces of the scene. We found that culling Gaussians outside a range of $\sigma = 0.5$ produced the best results to retain Gaussians on predicted surfaces.

In order to effectively mesh each Gaussian, the normal vector of each Gaussian needs to be calculated, which determines the orientation of the surface. We chose to assign the normal of each Gaussian to its smallest scale. This approach has merit, since we observe that Gaussians often elongate along surfaces that they represent.

We then leverage the process described in Section 3.3, which produces a point cloud that is generated specifically from a subset of predicted surface Gaussians, with each point having the same normal vector as its associated Gaussian. We then employ Open3D's [43] Poisson surface reconstruction algorithm to convert this surface point cloud into a coloured mesh. To further enhance the mesh quality, we remove the 10th percentile of points with the lowest densities. In this context, densities refer to the number of connected vertices that each vertex has. Additionally, Laplacian smoothing [38] can be optionally applied to reduce surface noise and improve mesh topology. This process is highlighted in Figure 7.

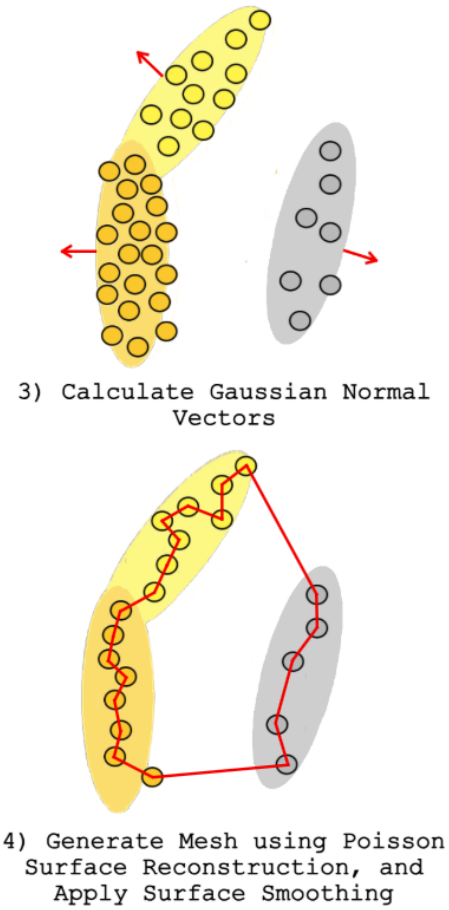

Figure 8. Overview of our approach for meshing 3DGS scenes. Firstly, points are generated using the standard point generation process on predicted surface Gaussians. The normals for each of these points are determined as the smallest side of its associated Gaussians. Next, a mesh is created by using Poisson surface reconstruction and then improved using Laplacian smoothing.

7.4.2. Discussion

Due to the efficiency of our rendering pipeline and point generation approach, extending our method to support mesh reconstruction introduces minimal computational overhead. As a result, mesh generation is considerably fast compared to other state-of-the-art models. However, the output quality is poorer than other dedicated 3DGS meshing methods, such as PGSR, in terms of detail and geometric fidelity. This is highlighted in Figure 9, which presents a visual comparison between meshes produced by our approach and PGSR. Since our method does not optimise Gaussians to conform to surface geometry during training, we rely on a Gaussian

Figure 9. Comparison between PGSR and 3DGS-to-PC for generating meshes on the DTU and TnT datasets.

cleaning step to discard noisy Gaussians, while retaining those that capture scene surface structures. However, this step is sensitive to the input 3DGS scene. On the DTU dataset, this filtering inadvertently removes Gaussians that were correctly aligned with object surfaces, leading to incomplete mesh topology. In contrast, on the TnT dataset, the filtering process struggled to cull Gaussians not situated on object surfaces, resulting in noisy mesh generation.

In summary, while our meshing approach is computationally efficient, it currently lacks the robustness and precision compared to other dedicated 3DGS meshing models. Hence, further refinement is necessary to achieve competitive performance.

7.5. Full DTU Results

Model	Metric	24	37	40	55	63	65
PGSR	PC → GT	0.73	1.10	0.65	0.43	1.56	0.49
	GT → PC	0.37	0.41	0.38	0.36	0.34	0.52
Ours	PC → GT	1.15	1.21	1.17	1.2	1.66	1.59
	GT → PC	0.43	0.42	0.58	0.41	0.35	0.48

69	83	97	105	106	110	114	118	122
0.49	1.17	0.94	0.78	0.40	0.53	0.42	0.48	0.41
0.39	0.41	0.40	0.38	0.43	0.32	0.28	0.38	0.33
1.38	1.47	1.42	1.07	1.17	1.43	1.18	0.94	1.18
0.43	0.8	0.48	0.45	0.56	0.48	0.33	0.43	0.33

Table 3. All point cloud to ground truth and ground truth to point cloud comparison results for each individual scene in the DTU dataset, for PGSR and 3DGS-to-PC methods.