Gaussian Opacity Fields: Efficient Adaptive Surface Reconstruction in Unbounded Scenes

Reviewer
Leonardo Mendonça
IMPA
leonardo.mendonca@impa.br

Archeologist
Mateus Barbosa
IMPA
mateus.barbosa@impa.br

Hacker
Institution3
secondauthor@i3.org

PhD Student
Diana Aldana
IMPA
diana.aldana@impa.br

1. Reviewer

1.1. Summary

GOF proposes improvements on the method of 3D Gaussian Splatting for surface reconstruction and novel-view synthesis: its central contribution is the development of the *Opacity field*, derived from the same images used for training the model. Additional improvements are also made to the 3DGS training process, by incorporating the depth distortion and normal consistency regularization terms from 2DGS[5]. Not only do these modifications improve performance, but they are also modular, which allows for their incorporation in different Gaussian Splatting pipelines.

GOF's main objective is surface reconstruction from multiple views of a static scene, in the form of a mesh that accurately describes the surface of the objects in the scene. There are also two secondary objectives: keeping a limited mesh size, and extracting accurate background meshes in unbounded scenes, a challenge which existing methods such as Neuralangelo [11] and 2DGS [4] still struggle with.

1.2. Methodology

Gaussian primitives. Similar to 3DGS [6], GOF models the scene through a set of K 3D Gaussian primitives $\mathcal{G}_1, \ldots, \mathcal{G}_K$. For a given point $\mathbf{x}$ in world space, a Gaussian $\mathcal{G}_k$ is defined as follows:

$$\mathcal{G}_{\mathbf{k}}(\mathbf{x}) = e^{-\frac{1}{2}(\mathbf{x}-\mathbf{p_k})^T \cdot \mathbf{\Sigma_k}^{-1} \cdot (\mathbf{x}-\mathbf{p_k})} \tag{1}$$

where $\mathbf{p_k}$ is the Gaussian's center and $\mathbf{\Sigma_k} \coloneqq \mathbf{R_k} \cdot \mathbf{S_k} \cdot \mathbf{S_k}^T \cdot \mathbf{R_k}^T$ is the variance matrix.

For rendering and optimization of these Gaussians, each $\mathcal{G}_\mathbf{k}$ is evaluated on a set of rays departing from each camera position. To find the intersection between a ray and a 3D Gaussian, one writes the ray with camera origin $\mathbf{o}$ and direction $\mathbf{r}$ as

$$\mathbf{x}(t) = \mathbf{o} + t\,\mathbf{r}; \tag{2}$$

We then express the ray in the Gaussian $\mathcal{G}_k$'s coordinate system[1] :

$$\mathbf{o_g} = \mathbf{S_k}^{-1} \cdot \mathbf{R_k}^T \cdot (\mathbf{o} - \mathbf{p_k}) \tag{3}$$
$$\mathbf{r_g} = \mathbf{S_k}^{-1} \cdot \mathbf{R_k}^T \cdot \mathbf{r} \tag{4}$$
$$\mathbf{x_g} = \mathbf{o_g} + t\,\mathbf{r_g} \tag{5}$$

Using these definitions, one can compute the depth t^* at which the Gaussian intensity $\mathcal{G}_k$ is maximized as $t^* = -\frac{\mathbf{o_g}^T \cdot \mathbf{r_g}}{\mathbf{r_g}^T \cdot \mathbf{r_g}}$.

Opacity field. For each Gaussian and viewing ray, one can define the *opacity along the ray*:

$$O_k(\mathcal{G}_k, \mathbf{o}, \mathbf{r}, t) \coloneqq \begin{cases} \mathcal{G}_k^{1D}(t) & \text{, if } t \leq t^* \\ \mathcal{G}_k^{1D}(t^*) & \text{, if } t > t^* \end{cases} \tag{6}$$

This definition extends naturally to multiple Gaussians through alpha-blending:

$$O'(\mathbf{o}, \mathbf{r}, t) \coloneqq \sum_{k=1}^{K} \alpha_k O_k(\mathcal{G}_k, \mathbf{o}, \mathbf{r}, t) \prod_{j=1}^{k-1} \Big(1 - \alpha_j O_j(\mathcal{G}_j, \mathbf{o}, \mathbf{r}, t)\Big); \tag{7}$$

[1] It should be noted that, in the paper, the definitions of $\mathbf{o_g}$ and $\mathbf{r_g}$ have a minor mistake: the authors write $\mathbf{R_k}$ instead of $\mathbf{R_k}^T$.

Finally, the *Gaussian Opacity Field* $O : \mathbb{R}^3 \mapsto [0,1]$ is defined as the minimum opacity of a point $\mathbf{x}$ along all viewing rays[2]:

$$O(\mathbf{x}) \coloneqq \min_{(\mathbf{o},\mathbf{r})} \left(\left\{ O'(\mathbf{o},\mathbf{r},t) \colon \mathbf{x}(t) = \mathbf{o} + t\,\mathbf{r} \right\} \cup \{1\} \right); \tag{8}$$

The intuition behind this field is that it measures how visible a given point is in relation to the training camera views: a point of opacity 1 is understood to be on the interior an object or outside the bounds of the scene, while a point of opacity 0 is plainly visible to some camera. Defined in this way, opacity is continuous and bounded throughout $\mathbb{R}^3$, which facilitates mesh extraction through Marching Tetrahedra.

For mesh extraction, the object's surface is defined as a level set of opacity $\{\mathbf{x} \colon O(\mathbf{x}) = 0.5\}$. Some points of interest are created on a bounding box of size 3σ around each Gaussian, from which a sparse mesh of tetrahedra is created through Delaunay triangulation. A Marching Tetrahedra algorithm [10] modified with a binary search is then used to extract a triangular mesh corresponding to this surface. Creating the tetrahedra only in the Gaussian's bounding boxes limits the computation time and the complexity of the final triangular mesh.

Regularizations. The paper incorporates the depth distortion and normal consistency regularizations from 2DGS[4]. However, the normal consistency relies on the definition of Gaussian normals, which is not quite clear in the text. The paper defines the normal as being $\mathbf{n_g} = -\mathbf{r_g}$ in the Gaussian coordinate system. Then, to get the normal in world coordinates, one "reverses" normalization and scaling:

$$\mathbf{n} = -\mathbf{R_k}^T \cdot \mathbf{S_k}^{-1} \cdot \mathbf{r_g} \tag{9}$$

Following our corrected definition of $\mathbf{r_g}$, the normal direction becomes

$$\mathbf{n_g} = -\mathbf{R_k}^T \cdot \mathbf{S_k}^{-2} \cdot \mathbf{R_k}^T \cdot \mathbf{r}; \tag{10}$$

Despite what is stated in the text, this is *not* equivalent to normalizing and denormalizing. Even though the ablation section shows good quantitative results for this approach, a better mathematical explanation is required.

1.3. Strengths

The paper's main contributions to the state-of-the-art are:

1. Applying the depth distortion and normal consistency regularizations from 2DGS [4] to 3D Gaussian optimization;
2. Defining the *Gaussian Opacity Field* for delimiting the surface of an object or scene;
3. Proposing an efficient method for extracting a mesh from a 3D Gaussian cloud, by adapting the Marching Tetrahedra [10] algorithm.

These contributions can be implemented together, as proposed in the paper, or separately, as part of a different Gaussian Splatting pipeline. As an example, the authors apply the opacity field and the mesh extraction method to the output of Mip-Splatting in order to extract a surface mesh, obtaining positive results.

Additionally, GOF achieves the best surface reconstruction among state-of-the-art explicit methods, with computational cost significantly smaller than implicit methods, as demonstrated through extensive experimentation on well-established datasets, and shows unprecedented capacity for background mesh reconstruction in unbounded scenes. We also argue that the definition of the Opacity field elegantly associates surface reconstruction with available view information, the result of which is a high quality mesh extraction.

1.4. Weaknesses

Some equations in the paper are not written in a rigorous manner or have minor mistakes, as pointed out throughout this review. More concerningly, the definition given for the Gaussian normal used in the normal consistency regularization is unclear, and the theoretical justification for it is very weak. Furthermore, the advantageous properties of the Gaussian Opacity Field that allow it to be used to extract a level surface are not mathematically proved.

From an empirical standpoint, we believe the paper lacks a quantitative experiment backing its claim that the meshes generated by GOF are less memory-intensive than the ones provided by existing methods. We also found the source code provided difficult to run in order to replicate the authors' results: even though installation instructions were provided along with the code, following them directly resulted in installation errors.

1.5. Evaluation

The mathematically imprecise definitions of the Gaussian normals and the Opacity field are the paper's main flaws. However, if these definitions are rewritten more clearly, we obtain a sound and innovative methodology that produces good results, as shown by the experiments. Therefore, we give the paper a score of 4 out of 5, and suggest that the authors further develop the mathematical aspects of

[2]One problem with the paper is that it writes $O(\mathbf{x}) \coloneqq \min_{(\mathbf{o},\mathbf{r})} O'(\mathbf{o},\mathbf{r},t)$, which is not rigorous.

the Gaussian normal and the Opacity field, in order to make these definitions more sound and prove that they work as intended.

2. Archeologist

GOF[15] is situated within the literature on surface reconstruction with Gaussians. Previous approaches in this domain have demonstrated limited success, while relying on different methods.

For example, SuGaR[3] employed Poisson reconstruction, while 2DGS[4] utilized 2D disks combined with TSDF fusion techniques.

GOF differentiates itself from these surface extraction methods by constructing opacity fields through a ray-tracing approach. This approach hinges on a unique method of calculating the intersection between rays and Gaussians.

The authors attribute their ray-Gaussian intersection technique to Gao et al. (2023)[2] and Keselman and Hebert (2022)[7]. Gao et al. (2023)[2] acknowledged that their method was inspired by Keselman and Hebert (2023)[8], who, in turn, clarified that their 2023 paper[8] built upon their earlier 2022 approach[7], making it compatible with 3D Gaussians.

Keselman and Hebert (2022)[7] proposed approximating the intersection of a ray with a 3D Gaussian as the point where the Gaussian's contribution peaks.

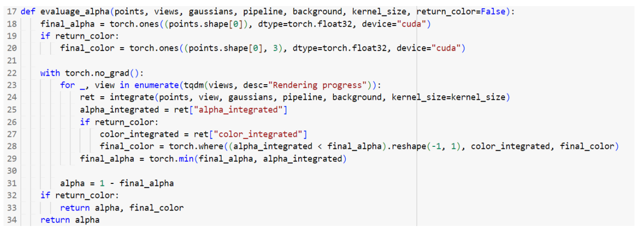
Currently, GOF[15] has accrued 34 citations. Of these, 10 papers actually incorporate elements of the method introduced in GOF, with 6 adopting only the improved densification method proposed in the paper.

One notable example is 3Diffusion[16], which aims to create realistic avatars with high-fidelity geometry and texture using 3D Gaussians. While 3Diffusion relies on GOF to generate depth maps, it employs volumetric TSDF fusion to extract the final meshes.

3. Code and experiments

First we will review the overall structure of the code and dive into certain details of it.

- It begins by initializing points in 3 dimensions with previous runs of COLMAP
- It then trains RGB and depth model using 3DGS [6] and 2DGS [4] loss functions and model including hyper . To do this it employs densification as seen in 3DGS and 2DGS. Particularly it does so every 100 iterations between iterations 500 and 15000. The hyperparameters for depth distortion and normal consistency are hard coded as 100 and 0.05 respectively.



```python
def evaluage_alpha(points, views, gaussians, pipeline, background, kernel_size, return_color=False):
    final_alpha = torch.ones((points.shape[0]), dtype=torch.float32, device="cuda")
    if return_color:
        final_color = torch.ones((points.shape[0], 3), dtype=torch.float32, device="cuda")

    with torch.no_grad():
        for _, view in enumerate(tqdm(views, desc="Rendering progress")):
            ret = integrate(points, view, gaussians, pipeline, background, kernel_size=kernel_size)
            alpha_integrated = ret["alpha_integrated"]
            if return_color:
                color_integrated = ret["color_integrated"]
                final_color = torch.where((alpha_integrated < final_alpha).reshape(-1, 1), color_integrated, final_color)
            final_alpha = torch.min(final_alpha, alpha_integrated)

        alpha = 1 - final_alpha
    if return_color:
        return alpha, final_color
    return alpha
```

Figure 1. Code to construct GOF

- To make sure it explores the scene appropriately the cameras are sampled randomly but eliminated until every camera has been explored. Also it has hard coded within its specifications to use 30% high-resolution camera which are identified in the pre-processing of data.
- To make sure they avoid numerical failures they reset opacity every 3000 iterations. In addition to this they use spherical harmonics to give a view dependent color of a single point in space. To make sure this Spherical Harmonics converge they begin with no view dependency and make sure to increase the complexity of the S.H. every 1000 iterations.
- Finally the code extracts the mesh from the depth calculations (obtained from the minimum opacity values from different view points mapped onto object space) using a binary search over the tetrahedra vertices and renders the scene. The specific code is provided in figure 1.

Then we proceeded to do a small experiment over the code. Particularly we turned off the hyper parameters of depth distortion and normal consistency and compared them to the original hyper parameters. Also we tried to compare the result of the render when ending the training at 7000 iterations comparing it to 30000 iterations. Unfortunately the results of the experiments where unsuccesful because the results in all 4 cases where all the same as shown in Figures 2 through 5.

4. PhD Project

As stated previously, one of the major advantages of this paper is their mesh reconstruction on unbounded scenes. We aim to leverage this property to segment the 3D objects of the scene and manipulate them separately. Specifically, our main idea would be to cluster the scene gaussians such that each group represents a different object. Then, these gaussians would become a discrete representation of an element

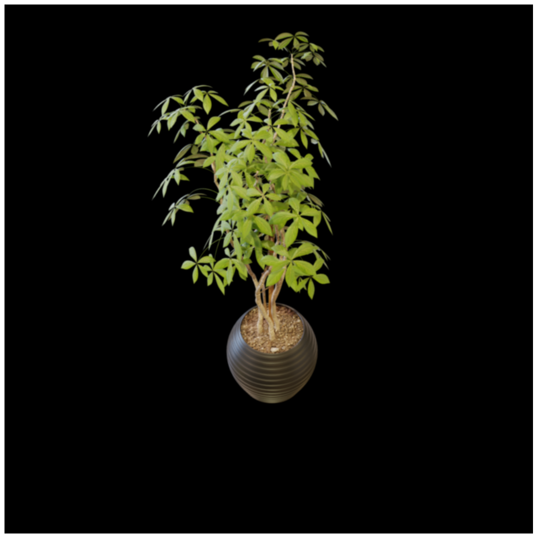


Figure 2. Original hyperparameters 7000 iterations

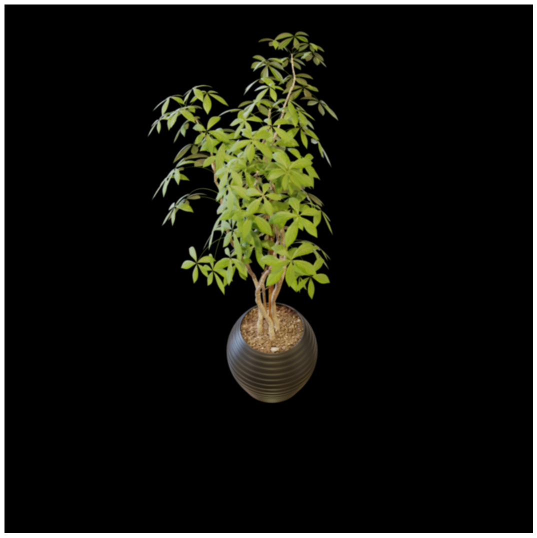


Figure 3. Hyperparameters set to 0 and 7000 iterations

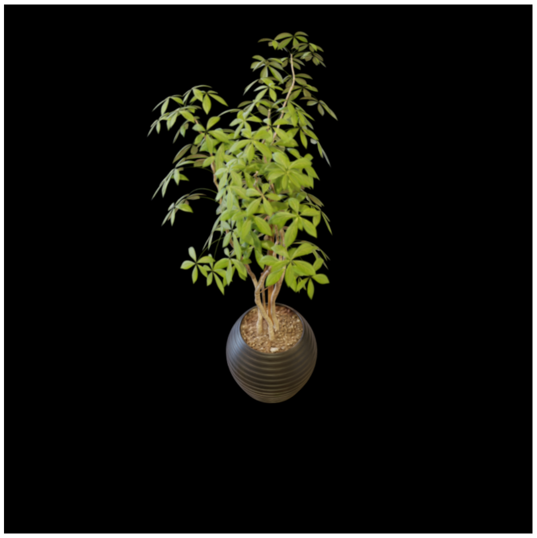


Figure 4. Original hyperparameters and 30000 iterations

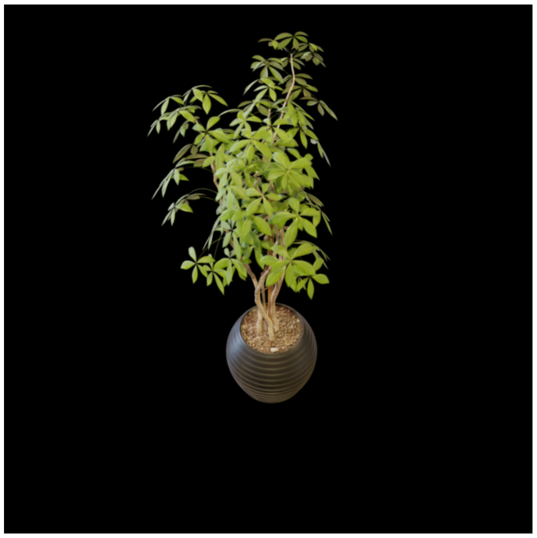


Figure 5. Hyperparameters set to 0 and 30000 iterations

of the scene, and can be manipulated to introduce transformations to the object.

Similar ideas of scene edition using gaussian splatting representations have been explored before. Zhou et al. [17] uses contrastive learning to categorize the gaussians and apply scene editing. However all experiments are done on closed scenes. Other works use gaussians to modify objects on the scene such as erasing/translating them [1, 14]. Zhu et al. [18] use physical constraints to move dynamically the gaussians that compose an object. Their results show realistic movement of objects, however their method focus only on traffic scenes.

Given the trained gaussians representing a 3D scene and the set of views used to train them, we want to learn the clusters of the scene. First, we obtain the segmentation maps and image correspondences using the algorithms of Depth Anything [9] and COLMAP [12, 13]. We use them to select the initial categorization of the gaussians in the different classes in the following fashion:

- **Single view**: We classify the gaussians closest to the camera by their class in the segmentation mask.
- **Multi-view**: We use the correspondences between pixels of different images to find the 3D point that is projected to the pixels. Then, we select one of such points as the centroid of that class' cluster. Experiments are necessary to observe which criterion is appropiated to select the cen-

troid.

- **All gausians classification**: For those gaussians occluded from all angles, we may use an algorithm such as *kkn* to classify them to the nearest cluster center.

After initializing the cluster classification, we may train the model to learn the classification using contrastive learning, and may even consider learning the centroids of the clusters to suppress the initialization dependence. Since the gaussians have already learned the scene, pruning and densification may appear unnecessary. However, we could take advantage of such stages to better define the boundaries between classes. For example, a big gaussian in the boundary of two classes may be split into two, each belonging to one of the classes.

Ideally, the result of such training may guarantee than a cluster of gaussians represents a single object detected by the segmentation mask. This cluster of gaussians can be manipulated to modify the object (e.g., rotation, removal, translation, flow) in the scene. Finally, observe that since each object is represented by a cluster of gaussians, we can reconstruct a mesh using the thethraedra algorithm in the paper applied to only this class, reducing the time and cost of mesh extraction.

5. Conclusion

GOF shows a novel scheme for extracting a mesh from a 3D scene with multiple training views. In doing so, it achieves a good balance between mesh fidelity and computation time, which is clearly demonstrated by the experiments done by the authors. However, it still shows room for improvement compared to implicit methods such as NeurAngelo [11]. Furthermore, there is still room for improvement when it comes to the mathematical definitions utilized in the paper: a possibility of improvement would be an effort to demonstrate why, *theoretically*, the GOF pipeline outperforms its competitors, notably in background meshes.

References

[1] Bin Dou, Tianyu Zhang, Yongjia Ma, Zhaohui Wang, and Zejian Yuan. Cosseggaussians: Compact and swift scene segmenting 3d gaussians with dual feature fusion. *arXiv preprint arXiv:2401.05925*, 2024. 4

[2] J. Gao, C. Gu, Y. Lin, H. Zhu, X. Cao, L. Zhang, and Y. Yao. Relightable 3d gaussian: Real-time point cloud relighting with brdf decomposition and ray tracing. *arXiv preprint arXiv:2311.16043*, 2023. 3

[3] Antoine Guédon and Vincent Lepetit. Sugar: Surface-aligned gaussian splatting for efficient 3d mesh reconstruction and high-quality mesh rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. 3

[4] Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2d gaussian splatting for geometrically accurate radiance fields. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–11, 2024. 1, 2, 3

[5] Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2d gaussian splatting for geometrically accurate radiance fields. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–11, 2024. 1

[6] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023. 1, 3

[7] Leonid Keselman and Martial Hebert. Approximate differentiable rendering with algebraic surfaces. In *Proceedings of the European Conference on Computer Vision (ECCV).*, pages 596–614, 2022. 3

[8] Leonid Keselman and Martial Hebert. Flexible techniques for differentiable rendering with 3d gaussians. *arXiv preprint arXiv:2308.14737*, 2023. 3

[9] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4015–4026, 2023. 4

[10] Jonas Kulhanek and Torsten Sattler. Tetra-nerf: Representing neural radiance fields using tetrahedra. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 18458–18469, 2023. 2

[11] Zhaoshuo Li, Thomas Müller, Alex Evans, Russell H Taylor, Mathias Unberath, Ming-Yu Liu, and Chen-Hsuan Lin. Neuralangelo: High-fidelity neural surface reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8456–8465, 2023. 1, 5

[12] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-Motion Revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 4

[13] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise View Selection for Unstructured Multi-View Stereo. In *European Conference on Computer Vision (ECCV)*, 2016. 4

[14] Mingqiao Ye, Martin Danelljan, Fisher Yu, and Lei Ke. Gaussian grouping: Segment and edit anything in 3d scenes. In *European Conference on Computer Vision*, pages 162–179. Springer, 2025. 4

[15] Zehao Yu, Torsten Sattler, and Andreas Geiger. Gaussian opacity fields: Efficient adaptive surface reconstruction in unbounded scenes. *ACM Transactions on Graphics*, 2024. 3

[16] Riccardo Marin Gerard Pons-Moll Yuxuan Xue, Xianghui Xie. Human 3diffusion: Realistic avatar creation via explicit 3d consistent diffusion models. *arXiv preprint arXiv:2406.08475*, 2024. 3

[17] Hongyu Zhou, Jiahao Shao, Lu Xu, Dongfeng Bai, Weichao Qiu, Bingbing Liu, Yue Wang, Andreas Geiger, and Yiyi Liao. Hugs: Holistic urban 3d scene understanding via gaussian splatting. In *Proceedings of the IEEE/CVF Conference*

on Computer Vision and Pattern Recognition, pages 21336–21345, 2024. 4

[18] Runsong Zhu, Shi Qiu, Zhengzhe Liu, Ka-Hei Hui, Qianyi Wu, Pheng-Ann Heng, and Chi-Wing Fu. Object-aware lifting for 3d scene segmentation in gaussian splatting. 4