Information-Guided 3D Gaussian Splatting

Master Thesis

of the

Graduate School of Natural Sciences,
Utrecht University

by

Ying-Kai Dang

Utrecht University Supervisor:	Dr. ir. Peter Vangorp
Utrecht University Daily Supervisor:	Victor Stenvers
Utrecht University Second Examiner:	Prof. dr. ir. Frank van der Stappen
NLR Supervisor:	Chihab Amghane
Project Duration:	May 2024 - January 2025
Company:	Royal Netherlands Aerospace Centre (NLR)

Dedicated to innovation in aerospace

Abstract

This thesis presents an intrinsically informed segmentation method using clustering for 3D Gaussian Splatting (3DGS) scenes to reconstruct solely the object of interest (OOI). We developed a pipeline that leverages the underlying distribution and density of Gaussians initialized from Structure from Motion (SfM) to segment the OOI. While the results do not show comprehensive segmentation of the OOI, the results are promising with room for improvement in future work. The pipeline is evaluated on a subset of the MiP-NeRF 360 [2] dataset, for which the ground truth segmentation masks have been manually created. This work contributes to creating 3D Gaussian Splats of solely an object at the center of the scene and is the first to the authors known method that allows 3DGS reconstruction on a specific object without foundational models. Furure research directions include incorporating the clustering pipeline into the training loop of 3DGS to enable more detailed segmentation of the OOI.

Contents

1 Introduction **4**
1.1 Problem Statement . . . 4
1.2 Motivation and Background . . . 5
1.2.1 Project Objective . . . 5
1.3 Research questions . . . 5

2 Related Work **7**
2.1 3D Reconstruction . . . 7
2.2 3D Gaussian Splatting . . . 8
2.2.1 Segmentation . . . 11
2.3 Point Cloud Algorithms . . . 12
2.4 Datasets . . . 13
2.5 Evaluation Metrics . . . 13
2.6 Information-Guided Gaussian Splatting . . . 14
2.6.1 Prior-Informed Segmentation . . . 14
2.6.2 Intrinsically-Informed Segmentation . . . 14

3 Methods **15**
3.1 Tools and Libraries . . . 15
3.2 Proof of Concept . . . 15
3.3 Prior-Informed Segmentation . . . 16
3.3.1 Reference Gaussian Selection . . . 16
3.3.2 Integration into 3DGS Pipeline . . . 17
3.4 Intrinsically-Informed Segmentation . . . 17
3.4.1 Density Clustering . . . 18
3.4.2 Floor Detection and Separation . . . 21
3.4.3 Noise Removal . . . 22
3.4.4 Meshing . . . 23
3.4.5 Masking . . . 24
3.4.6 3D Gaussian Splatting . . . 25

4 Results **27**
4.1 Dataset . . . 27
4.1.1 Ground Truth Generation . . . 27
4.2 Prior-Informed Segmentation . . . 29
4.3 Intrinsically-Informed Segmentation . . . 30
4.3.1 Clustering . . . 30
4.3.2 Meshing . . . 34
4.3.3 Masking . . . 34
4.3.4 3D Gaussian Splating . . . 35

5 Conclusion **39**

6 Future Work **40**
6.1 Initial Point Cloud . . . 40
6.2 Floor Detection . . . 40
6.3 Noise Removal . . . 41

Nomenclature

3DGS	3D Gaussian Splatting
BPA	Ball-Pivot Algorithm
DBSCAN	Density-based spatial clustering of applications with noise
IoU	Intersection over Union
KDE	Kernel Density Estimation
LPIPS	Learned Perceptual Image Patch Similarity
MC	Monte Carlo
NeRF	Neural Radiance Fields
OOI	Object of Interest
PSNR	Peak Signal-to-Noise Ratio
PSR	Poisson Surface Reconstruction
RANSAC	Random Sample Consensus
RG	Reference Gaussian
SfM	Structure from Motion
SSIM	Structural Similarity Index Measure

Chapter 1

Introduction

This thesis is conducted at the Royal Netherlands Aerospace Centre (NLR), a research institute committed to innovation in the field of aerospace. Among its projects, NLR dedicates efforts to cutting-edge research in 3D scene reconstruction from images. These initiatives make use of emerging techniques, including Neural Radiance Fields (NeRF) [33]and 3D Gaussian Splatting (3DGS) [18], to improve spatial data analysis within aerospace applications.

Current 3D reconstruction methods that aim to reconstruct solely an object within a scene rely on a fully-reconstructed scene, on which the object extraction is performed. This means that computational power and time needed to reconstruct anything but the object (e.g. the background) is wasted. This thesis proposes a new method that requires no prior knowledge of the scene to create a reconstruction of the object. The requirement for this is that the images taken revolve mostly around the to-be-extracted object.

This section introduces the problem statement and motivation, the research questions and a brief overview of the proposed method. Chapter 2 describes the related work on which the method builds up on. Following that, Chapter 3 explains the method. Then, Chapter 4 presents the evaluation and results of the method. Chapter 5 discusses the results wraps up with a conclusion and finally, potential future work is discussed in Chapter 6.

1.1 Problem Statement

Currently, exterior aircraft inspections are generally performed with a manual visual inspection. Depending on the type of maintenance, an inspection of the entire hull of the aircraft might need to be performed. A lightning strike calls for such an inspection and involves two to three aircraft engineers operating heavy machinery like cherry picker lifts to reach spots that are difficult to access, such as the top of an aircraft. These manual inspections can take between eight and twelve hours, causing the aircraft to be required to be grounded for that time before further actions are decided, which is very costly for the airlines [30]. Thus, it is important to leverage technological advances to create more time-efficient processes and operations to offset the lack of workforce and minimize aircraft downtime.

Companies such as Donecle[1] or Mainblades[2] are already providing solutions to speed up aircraft inspections. Their solutions involve using a semi-autonomous drone that flies around the aircraft and takes high-quality pictures of every part of the aircraft, which are then processed into a report. While this method can reduce inspection time by up to 75% [31], there are some limitations:

- **Lack of Depth Information**: Scratches or punctures in the hull have different severity levels, depending on the depth of the damage. Single images do not provide any depth information, making it difficult to assess the extent of the damage.
- **Single Viewpoint Constraint**: The inspection is limited to the angle from which the image was taken, potentially missing issues visible from other perspectives. Minor damages such as paint chips and scratches could be missed due to reflections of the aircraft.
- **Human Validation**: Since each image corresponds to just one part of the aircraft, it is challenging for a human inspector to assess the aircraft as a whole. Stitching these images together often

[1] https://www.donecle.com/

[2] https://www.mainblades.com/

lacks coherence and fails to provide a seamless experience. This is because the 2D images, when combined, do not always align perfectly and can distort the overall view. This fragmented approach makes it difficult to get a complete picture of the aircraft's condition.

By utilizing 3D reconstructions, the inherent limitations of 2D imaging, such as the loss of spatial information and restricted viewpoints, can be overcome. These reconstructions allow for the visualization of locations from multiple perspectives, providing a more comprehensive understanding of the scene. However, existing 3D reconstruction methods are impractical for aircraft inspections due to their prolonged processing times, which can extend to several hours. A significant portion of this processing time is spent reconstructing irrelevant parts of the scene, such as the background or elements that are not part of the aircraft.

Focusing the reconstruction process exclusively on the object of interest (OOI) solves this inefficiency. By omitting irrelevant parts of the scene, the computational burden is reduced, significantly accelerating the reconstruction. Additionally, isolating the OOI allows the resulting 3D model to be further integrated into specialized tools for subsequent processing and analysis.

1.2 Motivation and Background

Project BrightSky has been initiated to maintain the Netherlands' aviation expertise and strengthen its competitive international position. BrightSky is a collaboration between various companies and research centers in the aviation sector within the Netherlands, including the Royal Netherlands Aerospace Center (NLR). The project encompasses multiple objectives covering social innovation, digitization, and sustainability. One of such objectives aims to enhance aircraft fleet maintenance efficiency by creating digital tools to enable high availability, reliability, and cost-efficiency in fleet operations.

Aircraft fleet maintenance inspections ensure airworthiness, yet they remain a highly time-consuming and personnel-intensive process [30, 31]. The challenge is compounded when no qualified personnel is on-site, necessitating experts to fly in or rely on remote video assessments. The bottleneck created by the need for experts, along with the prolonged grounding time for aircraft inspections, highlights the necessity for more efficient solutions.

Within the BrightSky project, the NLR aims to develop autonomous drone swarms to acquire image data for 3D reconstructions. BrightSky aims to create a solution for aircraft maintenance inspections that enhances the quality of inspections while reducing the time and manual labor required. The primary objective is to generate a high-fidelity digital twin, allowing inspectors to interact with a comprehensive, integrated model. This approach reduces the number of personnel required for inspections, thereby enhancing cost-effectiveness. By enabling remote and digital maintenance inspections, there is no need for personnel to inspect the aircraft, which minimizes aircraft downtime physically.

1.2.1 Project Objective

To aid in achieving BrightSky's goal, the objective of this thesis is to develop a method with which a 3D reconstruction of a specific object within a scene can be created without reconstructing the entire scene first. Specifically, segmentation masks are used to reconstruct only the desired object instead of the entire scene. This allows for more efficient processing, as computational resources are focused solely on the object of interest. Additionally, this targeted approach enhances the level of detail and accuracy in the reconstructed object, as the system is not burdened with unnecessary background data.

Currently, 3DGS does not achieve a level of fidelity sufficient to replace the high-quality images captured by drones [29]. Despite this limitation, 3DGS remains a rapidly evolving research area, with significant advancements and new publications emerging on a near-weekly basis. Given this dynamic progress, the focus of this thesis is to investigate the current state of 3DGS and assess its potential capabilities. In the scope of BrightSky, this thesis explores how 3DGS could be effectively utilized in practical applications once its fidelity improves to meet the necessary fidelity.

1.3 Research questions

Based on the discussed project relevance, the following research question emerges:

How can information-guided Gaussian splatting be developed to selectively enhance the reconstruction quality of targeted objects within a 3D Gaussian Splat scene?

Current segmentation methods predominantly focus on 2D segmentation rather than 3D segmentation. This is largely due to the limited availability of 3D data and the high costs associated with annotating 3D datasets [14]. Additionally, within the domain of 3DGS, segmentation research typically focuses on post-training segmentation rather than utilizing segmentation to direct the training process toward an OOI. Existing 3DGS methods aim to reconstruct the entire scene, lacking the capability to concentrate on specific objects during reconstruction selectively. While reconstructing the entire scene is beneficial for comprehensive representation, it results in the unnecessary reconstruction of irrelevant parts, thereby reducing the overall reconstruction quality of the OOI due to the limited allocation of Gaussians for its representation.

A naive approach to address this issue is to create masks for the OOI in the input images and set the loss for all regions outside the OOI to zero. This method focuses the reconstruction solely on the OOI. However, this approach necessitates accurately segmented input images, which is typically a labor-intensive process.

The general idea is to implement information-guided Gaussian Splatting, where the information dictates what should be reconstructed and where Gaussians should be allocated. In the naive approach, the information is the mask of the OOI. However, the guiding information could be derived from other sources, such as semantic or spatial information. This brings up a sub-question:

What types of information can be most effectively utilized to guide the Gaussian Splatting process for selective detail enhancement?

Segmentation can be performed in various ways, such as by a point prompt or semantic meaning. Similarly, different types of information can be employed to select the OOI. One approach could be using the object's location within the scene, as the captured images for an OOI typically focus on the OOI itself.

Chapter 2

Related Work

This section provides an overview of the methods and techniques that are fundamental to the proposed method. To give the appropriate background of the assumptions on which the proposed method is based on, Structure from Motion (SfM) is explained. Afterward, 3D reconstruction with 3D Gaussian Splatting is discussed. Following that, an overview of the segmentation methods commonly used in 3D reconstruction is given. Then, the used point cloud algorithms and related methods are explained. This is concluded with a description of the used dataset, evaluation metrics and a brief overview of information-guided Gaussian Splatting.

2.1 3D Reconstruction

Structure from Motion

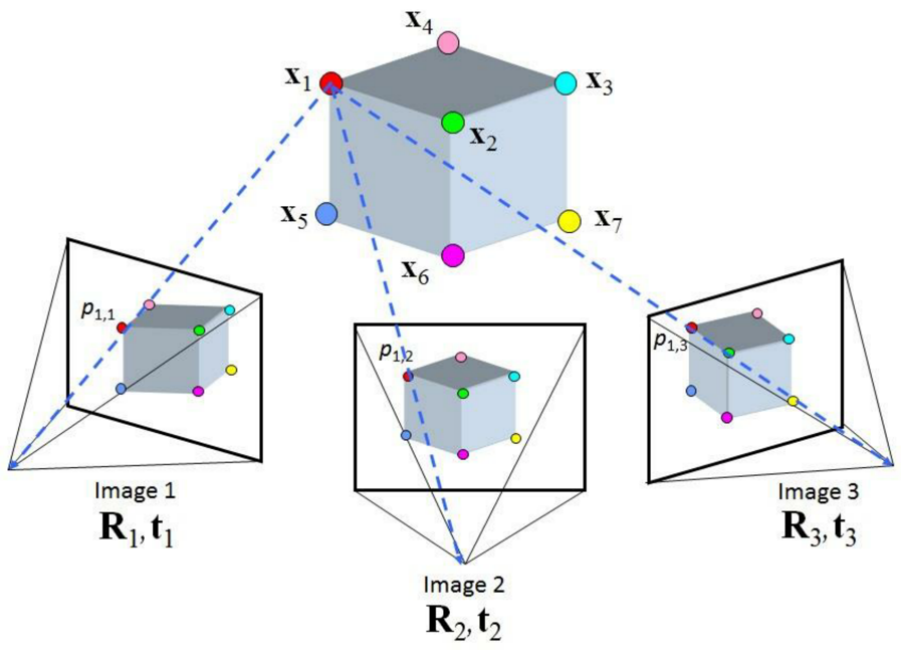


Figure 2.1: Structure from Motion process [11]. Each feature $p_{i,j}$ is matched across all images I_i to a point x_k in 3D space. R_i, t_i are the rotation matrix and translation vector of each image I_i

Structure from Motion (SfM) is a photogrammetric technique to create 3D reconstructions from a set of images. This process estimates 3D structures from 2D image sequences. SfM is based on the principle of motion parallax, where objects in a scene appear to move differently based on their distance from the observer. The technique typically involves three main stages: feature detection and extraction, feature matching, and structure and motion reconstruction.

COLMAP [37], a SfM and Multi-View Stereo (MVS) pipeline, is a widely used tool in the SfM domain and in COLMAP follows the three main stages. It begins with feature detection, where distinctive points in each image are identified. This is followed by feature matching, where corresponding features are matched across multiple images. Geometric verification then ensures that the matched features are consistent with geometric constraints. Finally, the incremental reconstruction stage gradually estimates and refines camera poses and 3D points.

Feature Detection and Extraction: COLMAP uses algorithms like SIFT (Scale-Invariant Feature Transform) [28] for feature detection and extraction. The goal is to identify distinctive keypoints in images that are invariant to scale, rotation, and partial occlusion. For each keypoint $\mathbf{x}_i = (x_i, y_i)$, a descriptor $\mathbf{d}_i \in \mathbb{R}^{128}$ is computed to represent the local image structure around the keypoint. This enables robust comparisons between keypoints in different images, even under varying lighting conditions or perspectives.

Feature Matching: Feature matching involves pairing keypoints across images. Given two sets of descriptors $\{\mathbf{d}_i\}$ and $\{\mathbf{d}'_j\}$, a nearest-neighbor search is performed to find matches. Robust estimation techniques, like RANSAC (Random Sample Consensus), refine the matches by estimating a transformation, such as the fundamental matrix $\mathbf{F}$, that satisfies $\mathbf{x}_j^\top \mathbf{F}\mathbf{x}_i = 0$. Matches violating this constraint are treated as outliers and discarded, leaving only consistent correspondences for reconstruction.

Structure and Motion Reconstruction: Once matches are established, COLMAP performs Structure-from-Motion (SfM) to jointly estimate camera poses and 3D points. The process starts with an initial image pair, solving for their relative pose using the essential matrix $\mathbf{E} = \mathbf{K}^\top \mathbf{F}\mathbf{K}'$. The reconstructed 3D points $\mathbf{X}_k$ satisfy the triangulation constraint:

$$\lambda_i \mathbf{x}_i = \mathbf{P}_i \mathbf{X}_k, \quad \lambda_j \mathbf{x}_j = \mathbf{P}_j \mathbf{X}_k,$$

where $\mathbf{P}_i$ and $\mathbf{P}_j$ are the projection matrices of the cameras. Bundle adjustment optimizes the camera parameters and 3D structure by minimizing the reprojection error:

$$\min \sum_{i,k} \|\mathbf{x}_i - \pi(\mathbf{P}_i, \mathbf{X}_k)\|^2,$$

where $\pi(\mathbf{P}_i, \mathbf{X}_k)$ projects 3D points into the image plane. This iterative refinement produces a globally consistent model.

New images are incrementally added to the reconstruction. For an image to be incorporated, it must have matching feature pairs with at least one image already included in the scene. These matches can either update existing points in X or expand the point cloud by introducing new triangulated features.

As the reconstruction progresses, COLMAP refines camera poses (position, rotation, and scale) for each image. This process often relies on Random Sample Consensus (RANSAC). By selecting a subset of triangulated points, RANSAC estimates the camera pose that best projects these points back onto the image. The algorithm validates this pose by testing the remaining points, discarding outliers that do not project correctly, and improving the robustness of the reconstruction.

The SfM process in COLMAP produces a dense 3D point cloud X, comprising all successfully triangulated features, and the corresponding camera poses for each image. This output forms the foundation for further processing, such as dense reconstruction and visualization.

2.2 3D Gaussian Splatting

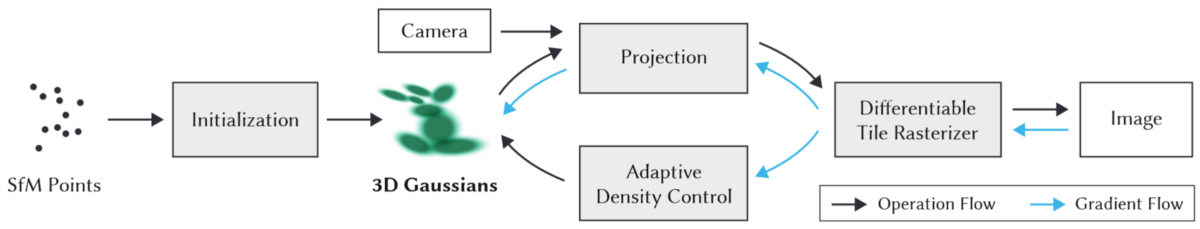


Figure 2.2: 3D Gaussian Splatting Pipeline[18]

Addressing the computational performance drawback that NeRF has, Kerbl et al. proposed 3D Gaussian Splatting (3DGS) [18], which contributes threefold: Introduction of 3D Gaussians as a high-quality unstructured representation of radiance fields, optimization method of 3D Gaussian properties with adaptive density control, and a fast differentiable rendering approach for the GPU. The overview of the method is shown in Figure 2.2.

Gaussian primitive. The basic idea of 3DGS is that the scene is represented as a set of Gaussian primitives, which are geometrically 3D Gaussian kernels, as opposed to an implicit representation using a function in NeRF. This approach is more consistent with traditional computer vision methodologies, making it easier to integrate and apply established techniques from the field. Those Gaussians are rasterized and "splat" via α-blending into the desired view. 3DGS uses COLMAP for Structure from Motion (SfM) [37] to obtain the camera poses that are needed as input. A nice by-product of SfM is that it generates a sparse point cloud. Those points can be taken as initial Gaussians to be placed in the scene.

A Gaussian $g_k := (\mathbf{x}_k, \Sigma_k, \alpha_k, c_k)$ is defined by a full 3D covariance matrix Σ centered at a point in the world space:

$$G(x) = e^{-\frac{1}{2}(x)^T \Sigma^{-1}(x)} \tag{2.1}$$

where a Gaussian has a position $\mathbf{x} \in \mathbb{R}^3$, 3×3 covariance matrix Σ which describes the shape and size of the Gaussian, as well as an opacity $\alpha \in \mathbb{R}$. The feature vector c stores the spherical harmonics coefficients up to the 3rd order, which represents the color.

Optimization. This 3D Gaussian needs to be projected to 2D for rendering. Since the covariance matrix Σ is analogous to the configuration of an ellipsoid, Σ can be described with a scaling matrix S and a rotation matrix R.

$$\Sigma = RSS^T R^T \tag{2.2}$$

Since optimization of both the scaling and rotation matrix is desired, they are stored separately as a 3D vector $s \in \mathbb{R}^3$ and a quaternion $q \in \mathbb{R}^4$, respectively. This representation of anisotropic covariance enables the optimization of the 3D Gaussians to change itself to the geometry of the scenes.

The optimization step optimizes the position $\mathbf{x}$, opacity α, covariance Σ, and SH coefficients representing the color c. This step is based on iteratively rendering the scene and comparing the resulting image to the training view. Stochastic Gradient Descent is used as an optimizer and the loss function is a photometric loss function, which combines the L1-loss with the Structural Similarity Index Metric (SSIM):

$$\mathcal{L} = (1 - \lambda)\mathcal{L}_1 + \lambda\mathcal{L}_{\mathrm{D-SSHM}}. \tag{2.3}$$

Density control. Alongside the optimization step, the Adaptive Density Control (ADC) is used to control the density of the Gaussians by adding or removing Gaussians. The proposed (heuristical) method is applied to the initial sparse point cloud from SfM to adaptively control the total amount of Gaussians in the scene as well as the density of each Gaussian. Gaussians that cover a large area of the scene (over-reconstruction) have to be split into two separate Gaussians, while Gaussians that miss geometric features (under-reconstruction) are cloned to capture more detail. The ADC essentially determines which regions of the scene need to shrink or expand in terms of scene representation budget, assigning the right amount of Gaussians to represent the region correctly. As the opacity is updated during the optimization step, some Gaussians can become nearly transparent. To combat this, Gaussians with an opacity below a set threshold are simply removed. This ensures that all present Gaussians affect the rendered image. Additionally, in order to remove floaters in the scene, the opacity is regularly reset to nearly zero, which is then corrected again during the optimization step for Gaussians that should not be transparent.

Rasterization. Kerbl et al. developed a tile-based rasterizer for Gaussian splats that pre-sorts the Gaussians for an entire image at a time. This allows for fast α-blending, which was expensive in previous works due to the usage of per-pixel sorting [25]. Additionally, the entire rasterization pipeline is fully differentiable. The rasterizer divides the screen into 16×16 tiles and performs culling of 3D Gaussians against the view frustum and each individual tile. Additionally, a guard band is utilized to remove Gaussians that are very close to the near plane or significantly outside the view frustum. After this, the Gaussians are sorted by their depth in view space and processed in a front-to-back order to accumulate their color and α values, a process known as α-blending. This accumulation is done for each pixel to determine the contribution of each overlapping Gaussian to that pixel. During this step, the only stopping criterion is the saturation of the α value, meaning the blending of Gaussians continues until the α value reaches a predefined threshold.

Many parameters in the optimization and densification process, such as performing densification every 100 iterations, are determined experimentally. Additionally, the heuristics for cloning, splitting, and pruning Gaussians depend significantly on having a high-quality initial point cloud for optimal performance.

3D Gaussian Splatting as Markov Chain Monte Carlo

Kheradmand et al. [20] propose a new approach where 3DGS is reimagined as Markov Chain Monte Carlo (MCMC) samples drawn from an underlying probabilistic distribution that is proportional to how faithfully the Gaussians reconstruct the scene. Stochastic Gradient Langevin Dynamics (SGLD) is used to explore the scene and update the Gaussians in the densification process. This method renders the heuristics involved in the densification process (densifying, pruning, and resetting opacity) unnecessary. Gaussians serve as samples for MCMC, with their locations explored through SGLD updates. Changes to the set of Gaussians can be reformulated as deterministic state transitions.

Optimization. Instead of defining a loss function and moving towards a local minimum, they propose a distribution G that assigns high probability to sets of Gaussians that accurately reconstruct the training images. Using this distribution, the MCMC framework can draw samples in a mathematically robust way, even with discrete changes in the parameter space. This allows for discrete operations similar to the original splitting and pruning heuristics of Gaussian Splatting without disrupting the continuity of gradient-based optimization. The SGLD method is used to update/optimize the Gaussians' parameters. It resembles the widely used Stochastic Gradient Descent update rule but introduces stochastic noise. The update of a Gaussian can be formulated as:

$$\mathbf{g} \leftarrow \mathbf{g} - \lambda_{\mathbf{lr}} \cdot \nabla_{\mathbf{g}} \mathbb{E}_{\mathbf{I} \sim \mathcal{I}} \left[\mathcal{L}_{\text{total}} \left(\mathbf{g}; \mathbf{I}\right)\right] + \lambda_{\text{noise}} \cdot \boldsymbol{\epsilon}, \tag{2.4}$$

where λ_{noise} and λ_{lr} are hyperparameters to control the learning rate and amount of exploration, $\mathbf{I}$ is a sampled image from the training set, $\boldsymbol{\epsilon}$ is the noise term, and $\nabla_{\mathbf{g}} \mathbb{E}_{\mathbf{I} \sim \mathcal{I}} \left[\mathcal{L}_{\text{total}} \left(\mathbf{g}; \mathbf{I}\right)\right]$ represents the raw gradients.

In the original 3DGS, the placement and optimization of Gaussians rely heavily on heuristics like cloning, splitting, and pruning. With the introduction of the noise term in SGLD, this method allows Gaussians to explore the full scene extent by being perturbed with anisotropic noise that matches the profile of the Gaussian. The noise is applied only on the center locations of the Gaussians. This means that the exploration of a Gaussian while being dependent on its covariances and opacities only changes the location of the Gaussian in the 3D space.

Heuristics as state transitions. The heuristics of moving, splitting, cloning, pruning, and adding Gaussians in the original 3DGS can be replaced with a state transition from $\mathbf{g}^{old}$ to $\mathbf{g}^{new}$. In order to integrate this mechanic into MCMC frameworks, the MCMC sampling may not be disturbed. This means that the probability of the sample state before and after the move should be preserved, $\mathcal{P}\left(\mathbf{g}^{ncw}\right) \approx \mathcal{P}\left(\mathbf{g}^{\text{old}}\right)$. This is achieved by moving Gaussians whose opacity is below a certain threshold, called 'dead' Gaussians, to the location of 'live' Gaussians. The covariance matrix of the moved Gaussian is then updated in a way that it tries to preserve $\mathcal{P}\left(\mathbf{g}^{\text{old}}\right)$ as much as possible:

$$\boldsymbol{\Sigma}_{1,\dots,N}^{new} = \left(o_N^{old}\right)^2 \left(\sum_{i=1}^{N} \sum_{k=0}^{i-1} \left(\binom{i-1}{k} \frac{(-1)^k \left(o_N^{new}\right)^{k+1}}{\sqrt{k+1}}\right)\right)^{-2} \boldsymbol{\Sigma}_N^{old} \tag{2.5}$$

As discussed in Section 2.2, the original 3DGS method depends on having a high-quality initial point cloud. Kheradmand et al.'s improvements eliminate this dependency on a good initial point cloud, as the assumed underlying distribution is modeled by the Gaussians themselves.

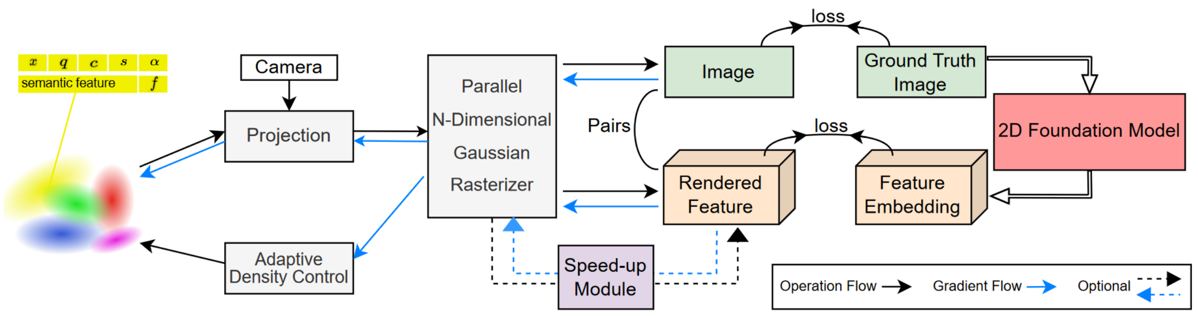


Figure 2.3: Feature 3DGS Pipeline[44]

2.2.1 Segmentation

Promptable 2D segmentation has seen significant advances, especially with the development of the Segment Anything Model (SAM) [22]. Promptable 3D segmentation has seen advances recently thanks to applying SAM to radiance fields such as NeRF [6, 21, 27]. Similar to NeRF, 3D Gaussians in 3DGS do not model object instances or semantic features and have to be extracted separately. While distilling 2D features into the realm of 3D Gaussians has been explored [5, 15, 16], the majority of those methods perform the segmentation on a trained set of Gaussians as opposed to attributing semantic meaning to them during the training, as their task is to allow interactive tasks such as editing the geometry or deleting objects from the scene.

Feature 3DGS

Zhou et al. [44] introduce a framework based on 3DGS for distilling semantic features at each 3D Gaussian point, allowing modifying selected Gaussians. This method learns semantic features of the Gaussians during training, allowing for more flexible use than learning the features after training. The general pipeline can be seen in Figure 2.3

Architecture. While NeRF-based feature distillation [24] can be achieved using two branches of multilayer perceptrons (MLPs) to output color and feature, it is more complex than simply rasterizing RGB images and feature maps independently for 3DGS. The proposed method in Feature 3DGS utilizes a parallel N-dimensional Gaussian rasterizer, effectively addressing the challenges of rendering arbitrary dimensional semantic feature maps. It employs a student-teacher framework for feature distillation, where a large 2D foundation model (teacher) extracts high-dimensional semantic features from 2D images, and the original 3DGS framework (student) is enhanced to learn and represent these features.

Rasterization. Based on the original 3DGS framework, Feature 3DGS incorporates the semantic feature $\mathbf{f} \in \mathbb{R}^N$, where N is an arbitrary number that represents the latent dimension of the feature, for example 512 for LSeg[26] encoding and 256 for SAM [22] encoding. The parallel N-dimensional Gaussian rasterizer is based on the original 3DGS rasterizer but features a point-based α-blending technique to rasterize the feature map. This means that for the k-th Gaussian, the attributes that can be optimized are $g_k := (\mathbf{x}_k, \Sigma_k, \alpha_k, c_k, \mathbf{f}_k)$. Similar to α-blending, projecting the color C of a pixel and feature value F_s order is done in front-to-back depth order [25]:

$$C = \sum_{i \in N} c_i \alpha_i T_i, \quad F_s = \sum_{i \in N} f_i \alpha_i T_i, \tag{2.6}$$

where $T_i = \prod_{j=1}^{i-1}(1 - \alpha_j)$ is the transmittance, $\mathcal{N}$ is the set of sorted Gaussians that overlap the given pixel, and s represents student, meaning the feature F_s is per-pixel supervised by the teacher feature F_t, which is the latent embedding encoded from the ground truth image using the 2D foundation model.

For rasterization, both the image and feature map use the same tile-based rasterization method as the original 3DGS rasterizer, using 16×16 tiles. The feature map and RGB image are rasterized to the same resolution but in different dimensions corresponding to the dimensions of c_k and f_k initialized in the Gaussians. This ensures that the rendered feature map has the same fidelity as the RGB image, which preserves the per-pixel accuracy.

Optimization. The loss function extends Equation 2.3 with a feature loss:

$$\mathcal{L}_{new} = \mathcal{L}_{rgb} + \gamma \mathcal{L}_f, \tag{2.7}$$

where $\mathcal{L}_{rgb}$ is Equation 2.3 and $\mathcal{L}_f = \|F_t(I) - F_s(\hat{I})\|_1$. I is the ground truth image and $\hat{I}$ is the rendered image.

The optimization of the semantic feature $f \in \mathbb{R}^N$ is done by minimizing the difference between the rendered student feature map $F_s(\hat{I}) \in \mathbb{R}^{H \times W \times N}$ and the teacher feature map $F_t(I) \in \mathbb{R}^{H \times W \times M}$, ideally with $N = M$. In practice, M is generally a large number due to the high latent dimensions of the output of the 2D foundation models, causing the rendering of the feature maps to be time-consuming. For this reason, feature 3DGS introduces a speed-up module consisting of a convolutional decoder to upsample the rendered feature map with a kernel size 1×1, allowing it to achieve $N = M$ without compromising the performance.

Prompting Gaussians. Through the student-teacher distillation process, the distilled feature fields enable the extension of all 2D functionalities of the 2D foundation models into the 3D realm. For a Gaussian in a set of ordered Gaussians overlapping a given pixel, the activation score for a prompt τ is calculated using the cosine similarity between the query $q(\tau)$ in feature space and the semantic feature $f(x)$ of the Gaussian, followed by a softmax operation:

$$s = \frac{f(x) \cdot q(\tau)}{\|f(x)\| \|q(\tau)\|} \tag{2.8}$$

Given a set $\mathcal{T}$ of possible labels, such as a text label set for semantic segmentation or a point set for all possible pixels in a point-prompt, the probability of a prompt τ for a 3D Gaussian can be determined using a softmax function:

$$\mathbf{p}(\tau | x) = \text{softmax}(s) = \frac{\exp(s)}{\sum_{s_j \in T} \exp(s_j)} \tag{2.9}$$

These computed probabilities are used to filter out Gaussians with low probability scores. This method of selecting Gaussians enables various operations, such as extraction, deletion, or appearance modification, by updating the color $c(x)$ and opacity $\alpha(x)$ values as needed. With point-based α-blending, these edited Gaussians can be rendered again from any view.

Feature 3DGS uses semantic feature segmentation for editing the Gaussian splatting scene after the training is done. For this thesis, the semantic features are being used *during* training.

2.3 Point Cloud Algorithms

Density-Based Spatial Clustering of Applications with Noise (DBSCAN)[9, 38] is a clustering algorithm that can discover clusters of arbitrary shapes and sizes in datasets containing noise and outliers. The algorithm operates on the principle of density-based clustering, where clusters are defined as dense regions of points separated by areas of lower point density. DBSCAN requires two main parameters: epsilon ϵ, which specifies the radius of the neighborhood around a point, and *minPts*, the minimum number of points required to form a dense region. The algorithm works by classifying points into three categories: core points, which have at least *minPts* points within their ϵ-neighborhood; border points, which are within the ϵ-neighborhood of a core point but don't have enough neighbors to be core points themselves; and noise points, which are neither core nor border points. DBSCAN starts with an arbitrary unvisited point, examines its ϵ-neighborhood, and if it contains sufficient points to satisfy the *minPts* criterion, a new cluster is formed. The algorithm then recursively expands this cluster by examining the ϵ-neighborhoods of its points, adding new points to the cluster if they are density-reachable. This process continues until no new points can be added to the cluster, at which point DBSCAN moves on to process the next unvisited point in the dataset.

Kernel Density Estimation (KDE)[8, 34] is a non-parametric method for estimating the probability density function of a random variable based on a finite data sample. The method works by placing a kernel function (typically a Gaussian distribution) at each data point and then summing these functions to create a smooth, continuous estimate of the underlying density. The key parameter in KDE is the bandwidth, which controls the width of the kernel functions and, therefore, the smoothness of the resulting estimate. A smaller bandwidth results in a more detailed but potentially noisier estimate, while a larger bandwidth produces a smoother but potentially oversimplified density estimate. In the context of clustering, KDE can be used to identify regions of high density in the data space, which often correspond to cluster centers.

Random Sample Consensus (RANSAC)[12] is a robust iterative method for estimating parameters of a mathematical model from a set of observed data that contains outliers. The method operates on the principle that data consists of "inliers," which can be explained by a model with a particular set of parameter values, and "outliers," which do not fit the model. RANSAC works by repeatedly selecting a random subset of the original data, hypothesizing a model from this subset, and then testing the model against the entire dataset. The algorithm determines how many points from the set are consistent with the model, the consensus set. This process is repeated for a fixed number of iterations or until a model with a sufficiently large consensus set is found. RANSAC can be used to find a plane of a point cloud when three points are chosen for the random subset.

Poisson Surface Reconstruction (PSR)[17] method for generating watertight 3D surfaces from point cloud data. PSR approaches the surface reconstruction problem by formulating it as a Poisson equation. The algorithm works by first computing a vector field that approximates the normals of the input points. It then finds a scalar function whose gradient best matches this vector field, with the reconstructed surface defined as an appropriate isosurface of this function. PSR's key strengths lie in its ability to handle noisy data and produce smooth, closed surfaces. The method employs an adaptive octree data structure, which can efficiently handle large datasets and capture fine details where the input point density is high. PSR is effective at filling holes and completing missing data, which makes it robust to incomplete scans or sparse point clouds. However, this hole-filling property can sometimes lead to oversmoothing of sharp features or the creation of spurious surfaces in areas with sparse data.

2.4 Datasets

The dataset utilized for evaluation in this thesis is Mip-NeRF 360 [2]. This dataset was chosen due to its compliance with a key requirement of the method under consideration: it must capture a complete 360-degree rotation around the OOI. While other datasets, such as Tanks and Temples [23], also provide 360-degree scenes, they are primarily designed for larger-scale environments. The authors of Mip-NeRF 360 critique the Tanks and Temples dataset for inconsistencies in photometric properties during data capture. These variations lead to changes in the appearance of objects across images, complicating the evaluation of metrics like PSNR, LPIPS, and SSIM, and reducing their reliability for meaningful comparisons.

Despite its suitability for 360-degree scenes, the Mip-NeRF 360 dataset is not without limitations. A significant issue identified is the presence of blurry images within the dataset. These blurs are attributed to motion during image capture or unintentional camera movement, which can introduce errors in the reconstruction process. Such inaccuracies propagate into the evaluation metrics, potentially skewing conclusions. Ideally, these blurry images should be excluded from the dataset to ensure data quality and the reliability of derived metrics.

However, a review of publications utilizing the Mip-NeRF 360 dataset for similar evaluations revealed that none explicitly mention the removal of blurry images. This omission suggests a lack of standardized preprocessing or quality control in the use of this dataset, which could influence the reproducibility and consistency of results across studies. Addressing these dataset shortcomings could improve the robustness of evaluations and the validity of comparative analyses.

2.5 Evaluation Metrics

For evaluating 3D Gaussian Splatting scenes, typically three metrics are used: Structural Similarity Index (SSIM), Learned Perceptual Image Patch Similarity (LPIPS), and Peak Signal-to-Noise Ratio (PSNR). SSIM[40] is a metric used to measure the similarity between two images. Unlike traditional methods like mean squared error, SSIM is designed to model the human visual system's perception of image quality. It compares local patterns of pixel intensities across three components: luminance, contrast, and structure. SSIM values range from -1 to 1, with 1 indicating perfect similarity. The metric is especially effective at capturing structural changes in images.

LPIPS[42] is a neural network-based metric for assessing image similarity. LPIPS uses deep learning models pre-trained on large-scale image classification tasks and fine-tuned on human perceptual judgments. The metric computes distances in deep feature space, capturing perceptual similarities that often align more closely with human visual assessment compared to traditional pixel-based metrics. LPIPS captures both low-level features and high-level semantic information.

PSNR is a widely used metric for assessing the quality of reconstructed or compressed images and videos. It is defined as the ratio between the maximum possible signal power and the power of distorting noise. PSNR is typically expressed in decibels (dB), with higher values indicating better quality. The metric is computed using the mean squared error between the original and the processed image. While PSNR is simple to calculate and provides a quick assessment of image fidelity, it doesn't always correlate well with human perception of image quality, especially for highly textured or structurally complex images, which is why it is typically used in conjunction with LPIPS and SSIM for 3D reconstruction tasks[7, 10, 41].

Chamfer distance[3] is a metric commonly used to measure the similarity between two point sets. For each point in one set, it finds the nearest neighbor in the other set and computes the squared distance. The Chamfer distance is then calculated as the average of these distances, summed over both directions (from set A to B and from B to A) to ensure symmetry. This bidirectional nature makes it robust to differences in point density and distribution.

The Hausdorff distance[13] is a metric used to measure the distance between two subsets of a metric space. The Hausdorff distance is defined as the maximum distance of a set to the nearest point in the other set. Formally, it's the greater of two values: the maximum minimum distance from points in one set to the other, and vice versa. This metric is particularly sensitive to outliers, making it useful for detecting significant shape differences or misalignments.

2.6 Information-Guided Gaussian Splatting

To answer the main research question "How can information-guided Gaussian Splatting be developed to selectively enhance the reconstruction quality of targeted objects within a 3D Gaussian Splat scene?", the sub-question of which *type* of information can be used to guide the Gaussian Splatting. What was identified as a requirement for any information is that a segmentation mask of the OOI is required since the calculated loss affects the Gaussian-optimization during the backpropagation. The loss for 3DGS is L1 photometric loss and SSIM, which essentially compare the pixels of the generated image to the input/ground truth.

For this reason, the research was focused on segmentation. Segmentation for 3D Gaussian Splatting scenes in this work is split into two categories: (a) Prior information and (b) intrinsic information. Prior information relies on data that must be processed or collected before training a 3D Gaussian Splat, whereas intrinsic information exclusively utilizes the data inherently available within the 3D Gaussian Splat itself.

2.6.1 Prior-Informed Segmentation

Prior-informed segmentation relies on input images that have been segmented already. Some works [5, 21, 44] rely on segmenting the input images and using the generated masks as a base for semantic segmentation. Semantic features are assigned to the Gaussians either during training or on a pre-trained 3DGS model. The use case is that segmentation using point or text prompts on a 3DGS model can be done.

2.6.2 Intrinsically-Informed Segmentation

Intrinsically-informed segmentation uses only information that is already included in a 3DGS model. As discussed by Kheradmand et al. [20], the Gaussian distribution of a 3DGS scene already describes the physical representation of the scene. This means the locations of the Gaussians can be taken to inform the segmentation of an OOI, as a high fidelity representation of an object in a 3DGS model also means a high number of Gaussians for that object.

Chapter 3

Methods

This section outlines the methods explored to address the research questions presented in Section 1.3. The groundwork for these methods, particularly that involving feature-based information, is drawn from the related work discussed in Section 2. The methods described pertain to the prior-informed and intrinsically-informed segmentation categories described in Section 2.6

3.1 Tools and Libraries

Framework. The main framework for this thesis uses the 3DGS MCMC[20] implementation for 3D Gaussian Splatting reconstruction. The development environment uses Docker for ease of reproducibility and portability.

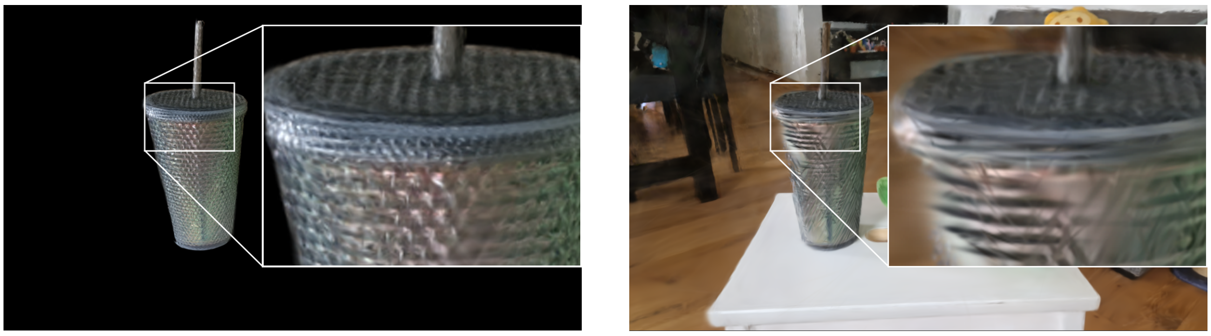
Libraries. The main libraries used for the implementation are Open3D[43] for point cloud manipulation and PyTorch[35].

Hardware. The NLR provides compute servers with two NVIDIA A100 80GB and one NVIDIA A100 40GB. For local development, an NVIDIA RTX 3080 10GB is available.

3.2 Proof of Concept

To verify that the 3DGS framework works correctly, the truck and garden scenes from the Mip-NeRF 360 dataset have been run on the 3DGS MCMC implementation. The yielded results align with the reported PSNR scores in the original paper. The time it took for each scene to complete 30,000 iterations was approximately 42 minutes on one NVIDIA A100 80GB GPU.

A naive approach to create a 3DGS reconstruction of just the OOI is to create masks for the OOI in the input images and set the loss for all regions outside the OOI to zero. This method focuses the reconstruction solely on the OOI. Figure 3.1a shows a rendered view of such an approach. This approach was tested on a custom dataset to serve as a proof of concept, trained for 30,000 iterations. Figure 3.1 illustrates the difference in detail between the reconstruction with segmentation masks, where the background loss is set to zero, and the reconstruction without any modifications. Although no formal evaluation was conducted, it is evident that the detail on the reconstructed cup is significantly higher using the naive segmentation approach compared to the original method without modifications.



(a) Background removed (b) Full scene reconstructed

Figure 3.1: 3D reconstruction of a custom scene with a cup[39]. Figure 3.1a shows the detail on the cup when the background is not reconstructed. Figure 3.1b shows the detail on the cup when the full scene is reconstructed.

3.3 Prior-Informed Segmentation

3.3.1 Reference Gaussian Selection

The concept of a Reference Gaussian (RG) involves selecting a single Gaussian that represents the OOI within a 3D Gaussian Scene. The RG is defined as the Gaussian with the highest semantic similarity to a text-encoded vector that represents the OOI. The RG acts as a reference point for identifying all Gaussians that belong to the OOI. This method allows segmentation of the OOI using user inputs such as text descriptions or point prompts (similar to approaches like SAM).

This approach to RG selection standardizes the selection process and ensures consistency across scenes. Even if the similarity between the text vector and the most similar Gaussian is low (e.g., 60%), normalizing all comparisons relative to the RG allows the use of a fixed threshold for selecting OOI-related Gaussians. This normalization step ensures the method's robustness regardless of the scene's variability or training iteration.

To start with RG selection, a naive K-Means clustering approach was chosen to test the method's feasibility in practice. After confirmation that the method works as intended, the RG selection was done via semantic similarity matching between the desired OOI and the semantic features of all Gaussians.

The semantic feature vectors of all Gaussians were clustered into ten groups, so $k = 10$. This k was arbitrarily chosen based on a qualitative estimation of how many objects are identifiable in the SfM point cloud. Each cluster was iteratively analyzed to identify clusters associated with the OOI. By trial-and-error, a cluster and Gaussian within that cluster was manually selected to represent the OOI.

After confirmation that using an RG works to segment the OOI from the 3DGS model, a more dynamic method was used to select the RG by using semantic similarity between the Gaussians and the text vector of the desired OOI.

Text Encoding: At first, the text description of the OOI is encoded into a feature vector using CLIP [36], a model designed to connect visual and textual features by embedding both images and text into a shared semantic space. This encoding translates the text description into a high-dimensional vector that captures its semantic meaning.

Cosine Similarity: Since the Gaussians' semantic features are also based on CLIP, it is possible to compare the Gaussian's features to the text feature. The cosine similarity between the Gaussian and the text feature is calculated for each Gaussian. The Gaussian with the highest cosine similarity to the text feature is chosen as the RG. This approach ensures that the RG is the Gaussian most semantically aligned with the OOI.

Gaussian Assignment: To identify the Gaussians corresponding to the OOI, the same cosine similarity as in the step above is done. However, the Gaussians' features are being compared to the RG instead of the text feature this time. This allows using a fixed threshold to identify all Gaussians belonging to the OOI, as cosine similarity is not transitive, and all comparisons are normalized relative to the RG.

3.3.2 Integration into 3DGS Pipeline

Once the RG is identified, the labeled Gaussians segment the OOI. The original 3DGS code's Gaussian pruning function is adjusted to remove not only low-opacity Gaussians but also those not labeled as part of the OOI. After a pruning step, the 3DGS model has only Gaussians of the OOI.

As discussed in Section 2.6, the loss calculation needs to take the removal of the non-OOI Gaussians into account, otherwise, the model will try to reconstruct those parts again. While the segmentation process qualitatively preserves the OOI during reconstruction, the loss is still calculated between the OOI and the original ground-truth image. This results in massively expanding Gaussians near the OOI's edges. To mitigate this, the pixel values from the rendered image are copied into the ground-truth image where the OOI is absent, essentially using the rendered image as a mask for the ground-truth image. With this, the loss for background regions is zero, preventing them from influencing backpropagation.

However, this approach of masking comes with its limitations. Each Gaussian's influence extends beyond its median position. If a Gaussian's median is within the OOI (meaning it is part of the OOI) but its extent overlaps the background, the resulting mask becomes "dirty," introducing artifacts along the OOI edges. This issue is resolved by relying on segmentation masks derived directly from input images rather than using rendered images. However, this approach ties the quality of the reconstruction directly to the accuracy of the input segmentation masks.

3.4 Intrinsically-Informed Segmentation

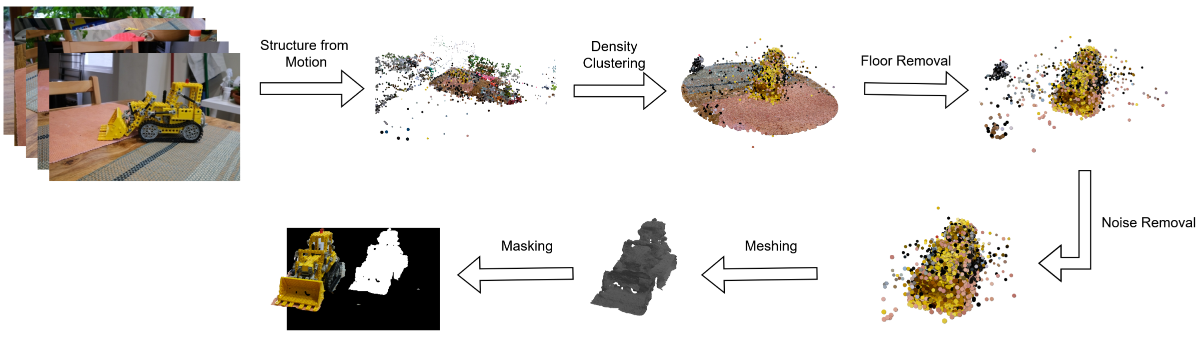


Figure 3.2: Pipeline for intrinsically-informed segmentation

An alternative method to isolate the OOI in 3D scene reconstruction involves clustering techniques. A 3D scene from SfM is represented as a point cloud, where higher detail corresponds to denser point regions. By using this property, clusters of high density can be identified, with the OOI expected to reside in one of these clusters. This proposed method serves as a preprocessing method for both the input images and initial point cloud that is fed into any 3D Gaussian Splatting method. It also offers advantages over feature-based 3DGS approaches[5, 44], as it avoids reliance on complex foundational models. Instead, it utilizes the natural point-cloud density generated by SfM, which inherently reflects the OOI's visibility across multiple dataset images. The density of points tends to be higher around the OOI due to its consistent appearance in the image set. Since the underlying probability distribution is proportional to the faithfulness of the Gaussians reconstructing the scene and does not need to be modeled explicitly[20], this proposed method can also work on 3DGS methods that do not rely on the initial SfM point cloud initialization. More details are discussed in Chapter 5.

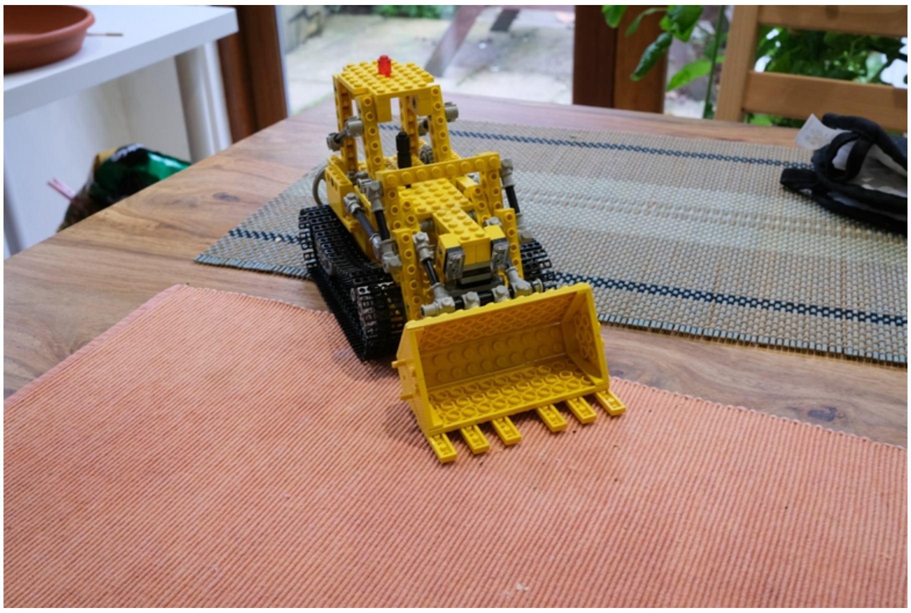
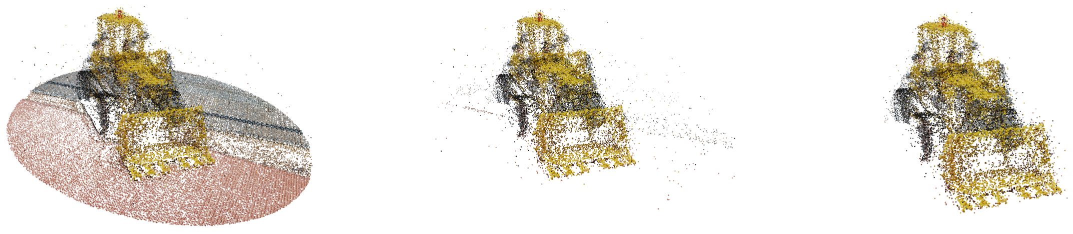


Figure 3.3: Reference image. The Object of Interest in this scene is the LEGO bulldozer.

This work builds on the assumption that datasets focus on single objects, similar to MiP-NeRF 360. The pipeline involves the stages (1) Density Clustering, (2) Floor Detection and Separation, (3) Noise Removal, (4) Meshing, and (5) Masking. To illustrate the effects of each stage, the kitchen scene from the MiP-NeRF 360 dataset is used as an example. A reference image from the kitchen scene is shown in Figure 3.3.

The first three stages, results depicted in Figure 3.4, are responsible for extracting only the points belonging to the OOI. They were designed to gradually decrease the size of the point cloud, first removing larger chunks that are irrelevant to the OOI and, lastly, going to the finer points around the OOI, which are noisy points due to inaccuracies from the SfM process. The meshing stage creates a surface reconstruction of the resulting OOI point cloud, essentially a 3D model of the OOI. This stage allows projecting the mesh onto the input images to create masks of the OOI.

The resulting masks and point cloud can subsequently be used as input images for any 3DGS method, following a similar approach to the proof of concept outlined in Section 3.2.

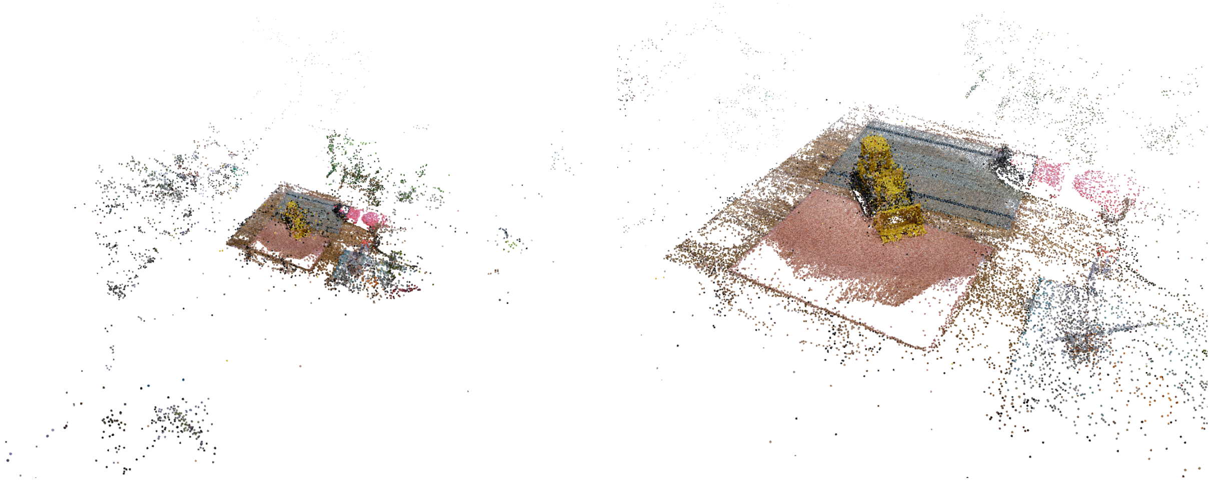


(a) Density clustering (b) Floor removal (c) Noise removal

Figure 3.4: Resulting point clouds of the OOI after each stage.

3.4.1 Density Clustering

The process of isolating the OOI within a point cloud relies on the underlying distribution of Gaussians in 3DGS. This distribution inherently reflects the density of points, with higher densities corresponding to regions of greater detail, particularly around the OOI. SfM-generated point clouds, such as the one in Figure 3.5, naturally exhibit this property: Firstly, SfM algorithms reconstruct 3D points by triangulating corresponding features across multiple images. Objects of interest tend to appear in a larger number of images and from various angles. This increased coverage results in more feature matches and, consequently, a higher density of reconstructed 3D points around the OOI. Secondly, objects of interest are typically captured from closer distances, leading to higher image resolution and more detectable features in these areas. The closer proximity also allows for better feature matching across images, further increasing point density. Additionally, objects of interest often have more distinct textures and features, which are easier for SfM algorithms to detect and match consistently across images. In contrast, background elements or less important areas may be captured less frequently, from greater distances, or with less consistent angles, resulting in sparser point reconstruction.



(a) Point cloud generated from SfM. (b) Zoomed in point cloud generated from SfM.

Figure 3.5: Point cloud generated from SfM is used as the input for the pipeline.

Building on the inherent density-based distribution of points in SfM-generated point clouds, this initial stage of the pipeline aims to drastically downsample the scene, effectively removing a large proportion of points that are not associated with the OOI. The primary objective of this stage is to maximize point removal while ensuring that all points belonging to the OOI are preserved, therefore maintaining the object's representation in the point cloud.

A Monte Carlo Kernel Density Estimation (MC KDE) approach is employed to quantify point density and enable OOI extraction. Instead of applying density estimation to the entire point cloud, which can be computationally expensive for large datasets, the method uses random subsampling. Multiple subsets of the point cloud are sampled, and a Kernel Density Estimation (KDE) model is fitted to each subset using a Gaussian kernel. The densities for all points in the original dataset are then estimated for each sampled subset, and the final density is computed as the average of these estimates. Experimental evaluations determined that a bandwidth of $h = 1$, sample size of $n = 1000$, and $k = 10$ iterations yielded consistent results across all scenes.

Subsequently, the estimated density values are filtered to remove low-density regions, allowing the peak-finding algorithm to isolate the OOI correctly. The lowest 30% of density values are discarded, a threshold determined through experimental evaluation. As background objects, although having low-density values, often comprise a large number of points due to their extensive presence in the background of the scene, filtering them out provides a solid base for the peak finding algorithm.

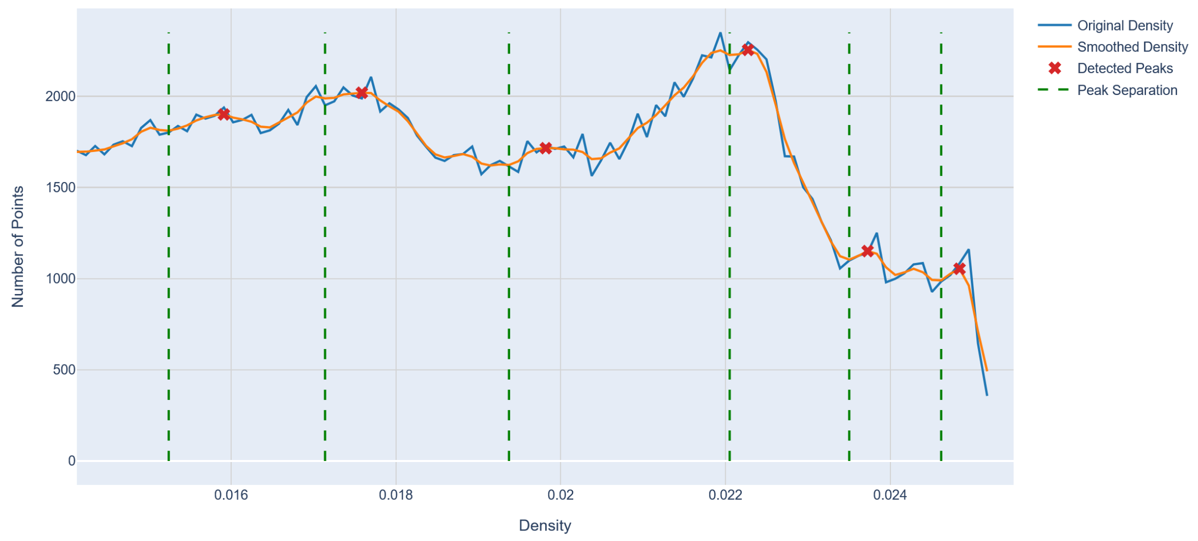


Figure 3.6: Density plot with smoothed peak detection.

The density distribution is analyzed to identify peaks using a histogram-based method. The density values are binned and sorted, and the upper portion of the distribution is smoothed with a Gaussian filter. A Gaussian filter with a standard deviation of $\sigma = 3$, determined experimentally after evaluating values in the range of 1 to 5, is applied to the density plot. Peaks are detected using a straightforward local maxima detection method provided by `scipy.signal.find_peaks`. The function finds all local maxima by a simple comparison of neighboring values. The density plot with smoothed peak detection is shown in Figure 3.6

Through experimentation, it was found that filtering out the points before the first peak leaves the OOI intact and filters out the non-OOI points that are further away from the high-density region, the OOI. The remaining points belong to the OOI and its immediate surroundings. Since the images are captured 360 degrees around the OOI, the density also decreases evenly the further out you go from the OOI, which results in the circular shape seen in Figure 3.7.

The following two stages focus on fine-tuning the point cloud. The objective is to narrow it down to represent the OOI exclusively.

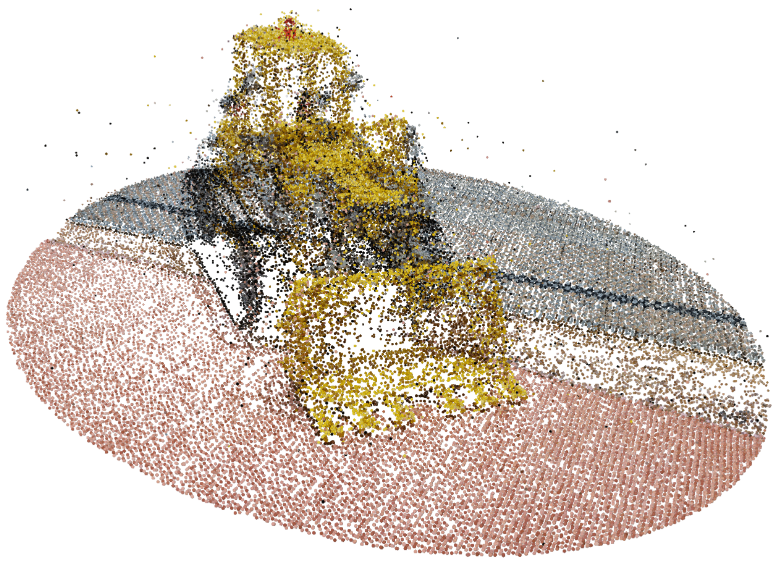


Figure 3.7: Extracted points from OOI based on density, where low-density regions have been omitted.

3.4.2 Floor Detection and Separation

The previous step does not completely remove the area around the OOI, leaving parts of the surrounding surface, such as the floor or a table, still present. These surfaces are not essential for reconstructing the OOI, but removing them can be challenging. If not done carefully, parts of the OOI close to the surface might also be removed, affecting the reconstruction's completeness. To address this challenge, the process is divided into three substages: detecting the floor, analyzing the model's orientation, and removing the floor, which allows for finer control of separating the OOI from the floor.

The process of floor detection employs a Random Sample Consensus (RANSAC)-based algorithm to fit a plane model to the point cloud. Initially, a random subset of three points is selected to approximate the plane of the floor. The number of iterations determining how many times the algorithm tries to fit the model to the subset of points is set to `num_iterations=1000`.

The distance threshold, which defines the maximum allowable distance a point can be from the model to be considered an inlier (part of the floor), is determined dynamically. The algorithm chooses the ideal distance threshold by trying 50 different thresholds between 0.005 and 0.1 and choosing the optimal threshold by the minimum value of the second derivative of the ratio between inliers and the total points. The second derivative identifies the point where the rate of change in the ratio of inliers to total points decreases most sharply. The minimum value of the second derivative corresponds to the "elbow" of the floor ratio curve, which is marked as the red vertical line in Figure 3.8. This threshold represents the transition where adding points provides minimal new information about the floor while minimizing the inclusion of extraneous points.

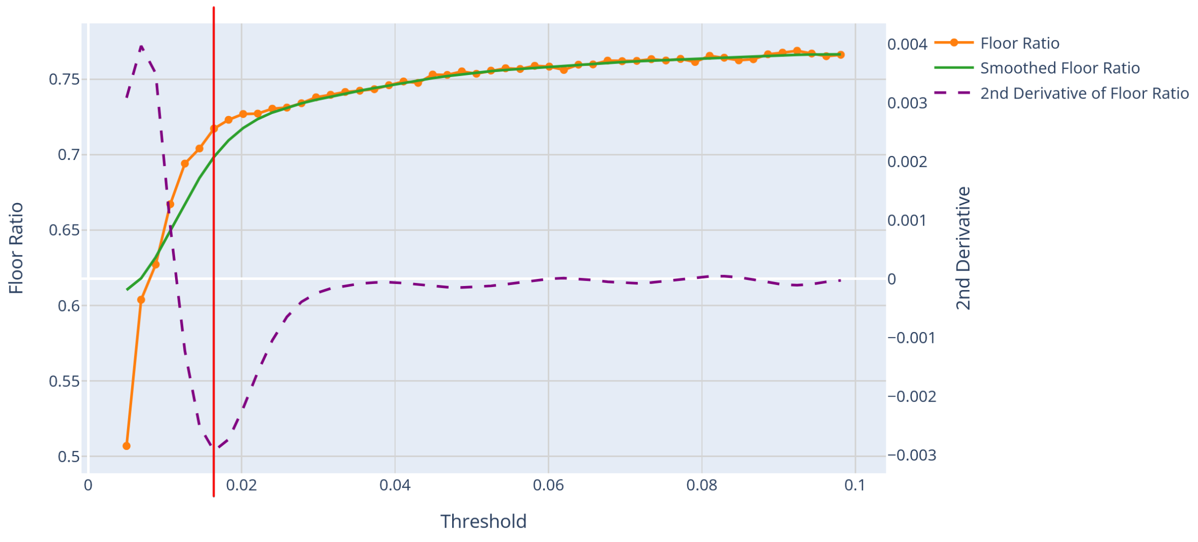


Figure 3.8: RANSAC distance threshold search evolution. The optimal distance threshold was found to be 0.0164, as indicated by the vertical red line, which also marks the "elbow" point of the floor ratio curve.

Intuitively, this can be explained as follows: The distance threshold determines how "thick" the floor is perceived. For a thicker floor, a higher distance threshold is required. As the threshold increases and not all floor points are yet identified, each increment significantly increases the number of inliers. However, once the entire floor is captured, further increases in the distance threshold result in diminishing returns, as the additional points captured belong to areas above the floor, like the surrounding air. Since the floor in the point cloud extends well beyond the area covered by the OOI, this extension only marginally increases the inlier count because the OOI occupies a relatively small portion of the floor.

This step ensures accurate identification of the floor, minimizing overlap with OOI points. Figure 3.9 highlights the detected floor in green.

Orientation analysis of the detected floor plane is needed to ensure proper separation. The normal vector of the plane found in the previous step is computed to identify its orientation in the 3D space. Points above and below the plane are then segregated. Given that no part of the OOI should exist beneath the floor, the region with the fewest points is discarded. This ensures that only the points corresponding to the OOI remain, free of any extraneous points from the floor or the surrounding area.

Figure 3.10 shows a side view from the OOI where the floor has been removed. Note that due to a non-optimal floor thickness, part of the OOI was removed as well.

The resulting point cloud consists of the OOI, with the majority of the floor points removed.

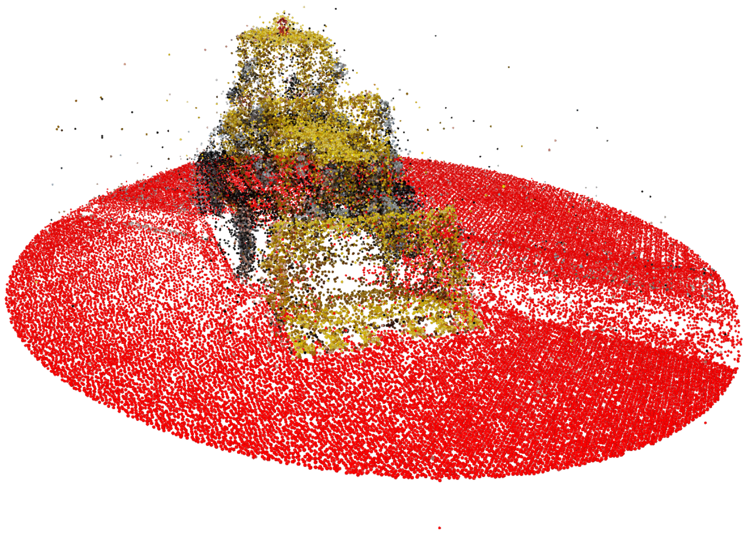


Figure 3.9: Point cloud of OOI with floor, where the detected floor is colored red.

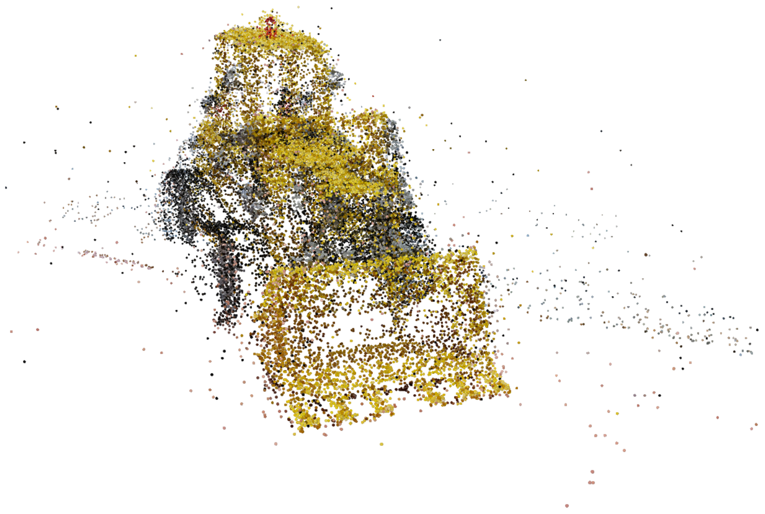


Figure 3.10: Point cloud of OOI with floor removed. Note that the bottom of the tracks on the bulldozer has also been mostly removed.

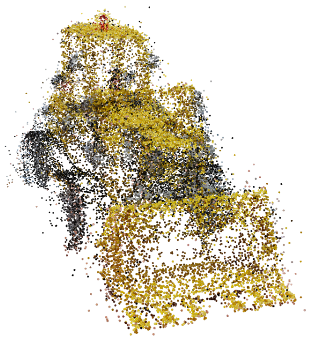
3.4.3 Noise Removal

Feature extraction and matching in SfM pipelines are inherently imperfect and often introduce noise into the generated point cloud. This noise arises from inaccuracies in feature matching, camera alignment, or reconstruction processes. When extracting the OOI, noise can propagate into the filtered point cloud because the extraction typically relies on point density criteria. As a result, noisy points in proximity to the OOI may remain in the dataset, complicating surface reconstruction on the point cloud and potentially degrading the quality of the reconstructed model.

To address this issue, the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm is employed to refine the filtered point cloud. DBSCAN groups points into clusters based on their spatial proximity, guided by two key parameters: the neighborhood distance threshold ϵ and the minimum number of points required to form a cluster. These parameters are critical to the algorithm's performance and are influenced by the scene's characteristics. For example, scenes with larger spatial extents or lower point density typically require a larger ϵ to account for greater inter-point distances.

DBSCAN also identifies outliers as noise, effectively removing isolated points that do not belong to any cluster. This dual capability makes it particularly suitable for noisy datasets. Following clustering, the resulting clusters are evaluated to determine the one most representative of the OOI. Typically, the largest cluster defined by the highest number of points is assumed to correspond to the OOI, as it reflects the region of greatest spatial coherence within the filtered point cloud. The denoised OOI using DBSCAN is shown in Figure 3.11a.

Another noise removal method, statistical outlier removal, was also tested for this purpose. The results are seen in Figure 3.11b. While the noisy points around the OOI are mostly removed, similar to DBSCAN, some smaller clusters not part of the OOI still remain.


(a) Denoising of OOI using DBSCAN after floor separation.

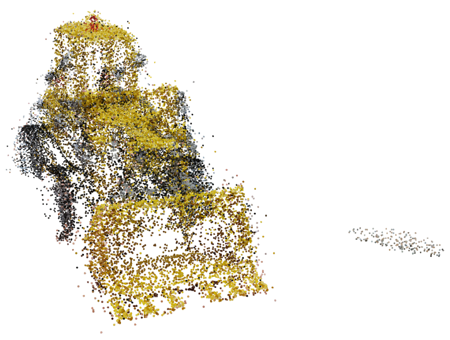

(b) Denoising of OOI using statistical outlier removal after floor separation.

Figure 3.11: Noice removal using DBSCAN and statistical outlier removal.

The resulting point cloud can be used for initializing the Gaussians in place of the SfM point cloud.

3.4.4 Meshing

To generate segmentation masks, the point cloud of the OOI must first undergo surface reconstruction or meshing to create a continuous representation of the surface based on the discrete points. Among surface reconstruction methods explored (ball-pivot algorithm (BPA), Delaunay triangulation, and Poisson Surface Reconstruction (PSR)), PSR emerged as the most suitable for this application due to its robustness in handling noisy data. By integrating smoothing into the reconstruction process, PSR effectively mitigates noise, although this comes at the cost of potentially losing finer geometric details. Both BPA, seen in Figure 3.12b, and Delaunay triangulation are sensitive to noise and require dense or highly uniform points [4, 19], making them unsuitable for this task.

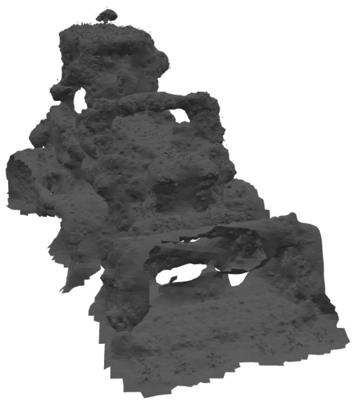
However, a key limitation of PSR is its reliance on point normals. SfM point clouds often lack precomputed normals; an additional preprocessing step to estimate them is necessary, which, especially for sparse point clouds, can yield incorrect normals. The normals are estimated using Open3D's `estimate_normals` and `KDTreeSearchParamHybrid` functions. The function uses a k-d tree, which is a data structure optimized for efficient neighbor searches, to identify the set of neighboring points for each point in the point cloud. The KDTreeSearchParamHybrid parameter defines how this neighborhood is determined: it considers all points within a given radius and ensures that at most `max_nn` neighbors are selected.

For each point, the algorithm computes a best-fit plane through its selected neighbors using Principal Component Analysis (PCA)[1]. PCA identifies the direction of minimal variance among the neighboring points, which corresponds to the normal vector perpendicular to the best-fit plane. This computed normal is then assigned to the current point and stored in the normals attribute of the point cloud. A smaller radius captures fine local details but can be more sensitive to noise, while a larger radius smooths the normals over a broader area, which is beneficial for noisy or sparse data. Through experimentation, it was found that the parameter choice did not affect the resulting normals for the scenes evaluated in this thesis. This is likely due to the OOI being dense enough. Therefore, the radius parameter was set to $r = 2$ and the maximum nearest neighbors to $n = 100$.

Furthermore, the complexity of the OOI, particularly when it features indentations or hollow structures, raises challenges in capturing its concavity. In such cases, a concave hull would theoretically offer a more accurate representation of the OOI. However, the sparsity of SfM point clouds often undermines their ability to accurately define concave regions, making concave hull generation unreliable for preserving the true geometry. Instead, a simple convex hull could be used to ensure no significant details are excluded, although at the expense of vastly over-simplifying concave features.

In the context of PSR, the depth parameter controls the depth of the octree used in the algorithm, which in turn dictates the level of detail in the final mesh. A higher depth allows for more detailed surfaces, but it also increases noise and computational demands. To balance detail preservation and noise suppression, a default depth value of 9 has been experimentally selected for all scenes. This depth provides sufficient detail to capture the geometry of the OOI while maintaining a degree of smoothing to address noise commonly present in SfM-generated point clouds.

As illustrated in Figure 3.12a, the mesh generated by the Poisson surface reconstruction method on the OOI dataset shows its robustness against noise. Specifically, the reconstructed surface effectively disregards the residual noisy points.


(a) PSR of the OOI.

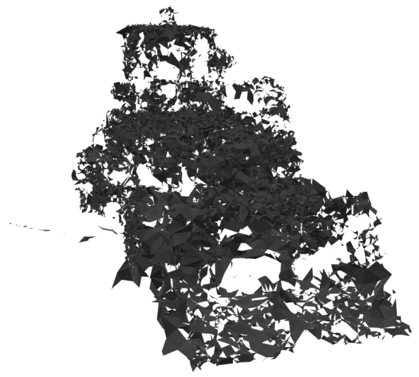

(b) BPA reconstruction of the OOI.

Figure 3.12: Resulting meshes of the denoised OOI point cloud with PSR and BPA.

3.4.5 Masking

By applying meshing, a 3D segmentation of the OOI is effectively achieved. The meshed representation of the point cloud ensures consistency across all viewpoints, creating a unified surface model of the OOI. This mesh can then be used to generate segmentation masks for the input images. Specifically, the meshed OOI is rendered from the same viewpoints as the original camera positions used during data acquisition, allowing for accurate mask generation for each input image.

To achieve this, the positions and orientations of the original cameras must be replicated. Using the camera extrinsics (position and orientation) and intrinsics (lens parameters) derived from the SfM process, virtual cameras are placed in 3D space to precisely emulate the locations and orientations of the real-world cameras that captured the dataset. The meshed OOI is then rendered from the perspective of each emulated virtual camera, producing binary masks that delineate the OOI's silhouette as viewed from those angles.

More specifically, the projection begins by extracting the vertices of the input mesh and converting them to homogeneous coordinates. Homogeneous coordinates include an additional dimension, allowing the vertices to undergo transformations using the provided camera extrinsics. These extrinsics, represented as a 4×4 matrix, encode both the rotation and translation of the camera, mapping points from the world coordinate system to the camera's coordinate system.

Once transformed into the camera coordinate system, the vertices are filtered to retain only those with positive Z-coordinates, as points with non-positive Z-values lie behind the camera and cannot be projected onto the image plane. This step not only ensures accurate projections but also enhances computational efficiency by reducing the number of vertices processed further. The triangular faces of the mesh are updated accordingly by remapping the indices of valid vertices and keeping only the triangles whose vertices are all visible.

The filtered 3D vertices are then projected onto the 2D image plane using the camera's intrinsic parameters. These parameters, which include focal lengths (f_x, f_y) and the optical center (c_x, c_y), are used to construct the camera calibration matrix K. This matrix transforms the 3D points into normalized 2D-pixel coordinates. The projection is completed by dividing the resulting coordinates by their Z-component, converting them from homogeneous to Euclidean space. This results in the 2D-pixel coordinates of the projected vertices, a mask indicating which vertices were retained after filtering, and the updated triangular faces of the mesh. The output of this projection process is utilized to generate a binary mask.

During the meshing process, small gaps may arise in the reconstructed surface due to the inherent characteristics of PSR, such as smoothing or insufficient point density in certain areas. These gaps can result in incomplete coverage of the OOI in the generated masks. To address this, morphological image processing operations, including opening, closing, and dilation, are applied to the binary masks. These operations help to achieve more coverage of the OOI by filling in small gaps, smoothing boundaries, and enhancing the integrity of the masks.

The result of the masking process can be seen in Figure 3.13

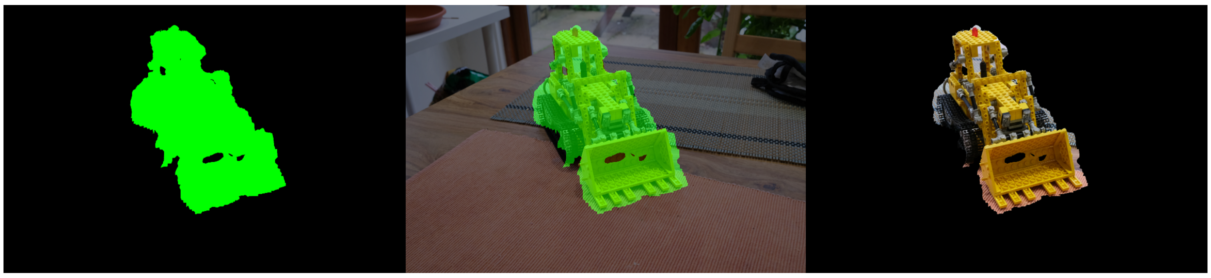


Figure 3.13: Projected mask on the input viewpoint to create a segmentation mask of the OOI.

The generated masks can subsequently be used as input images for 3DGS.

3.4.6 3D Gaussian Splatting

The base 3DGS code utilized for this thesis is derived from 3DGS MCMC [20]. Unlike the original 3DGS implementation [18], which dynamically adjusts the number of Gaussians in each scene, 3DGS MCMC provides the ability to set a maximum number of Gaussians explicitly. This makes it easier to control and compare the proposed method of this thesis with existing approaches, as the number of Gaussians used for reconstructing the OOI can be precisely defined. To use the generated masked images from Section 3.4.5 as input, the loss function needs to be modified to take the masks into account, otherwise the black background will also be reconstructed.

For the L1 loss, the modification involves multiplying the pixel-wise difference between the rendered image and the ground truth by the binary mask before computing the absolute values and taking the mean. This makes sure that only the regions indicated by the mask contribute to the loss. Pixels outside the mask (where the mask value is zero) are effectively ignored in the loss calculation. For the SSIM, the modification involves applying the binary mask to the computed SSIM map. The SSIM map represents the structural similarity between two images across spatial dimensions. By multiplying this map with the binary mask, the contribution of regions outside the mask is eliminated. These modifications ensure that the background does not influence the optimization process.

Additionally, the point clouds generated in Section 3.4.3 can be directly used for the Gaussian initialization and replaces the SfM point cloud. This brings the advantage that the initialized Gaussians are already only on the OOI and not in the background.

Chapter 4

Results

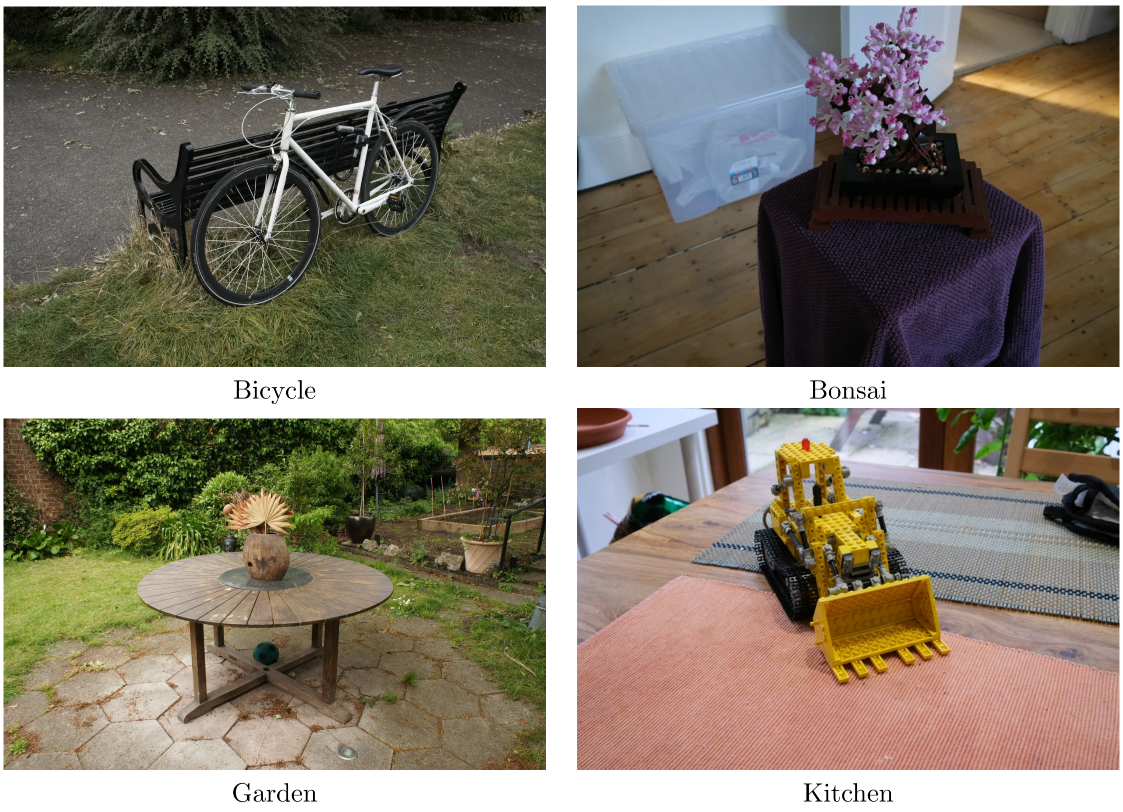


Figure 4.1: Dataset

4.1 Dataset

For this thesis, four scenes from the Mip-NeRF 360 dataset - bicycle, bonsai, garden, and kitchen, see Figure 4.1 - have been selected. Although the other scenes also depict inward-facing 360-degree captures, they lack a singular, well-defined object at the center that allows for precise evaluation. While the stump scene does feature a central object (a tree stump), its boundaries are highly subjective, as the stump blends seamlessly into the surrounding earth and vegetation.

The method used for prior-informed segmentation was evaluated on a custom dataset.

4.1.1 Ground Truth Generation

To evaluate both OOI extraction from the point cloud and mask generation, ground truth data for the point cloud and segmentation masks are essential. While Liu et al. [27] evaluated their method using Mip-NeRF 360 by randomly selecting approximately 10 views and refining the generated masks with

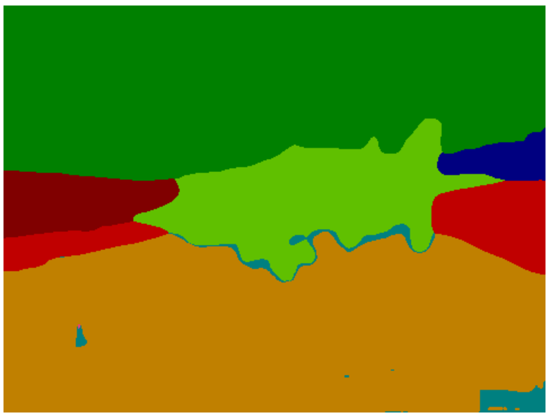
MMLAB and CascadePSP, this approach is insufficient for comprehensive evaluation. The generated masks, derived from semi-supervised algorithms, do not constitute true ground truth data, as they are still algorithmically created and may lack the accuracy needed for reliable benchmarking.

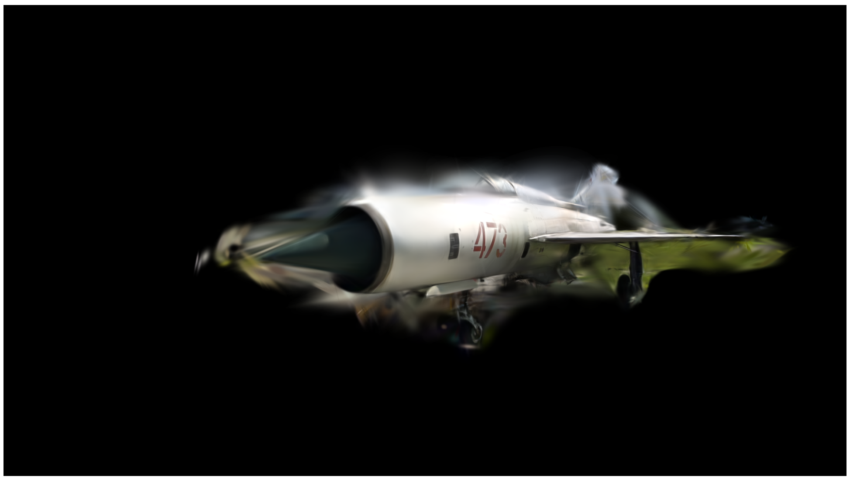
To address this limitation, a more rigorous approach has been applied to the Mip-NeRF 360 dataset. Specifically, the kitchen scene has been fully manually masked for all 279 images to provide high-quality ground truth segmentation. For other scenes, 20 representative views have been selected and similarly manually masked. This ensures that the evaluation of mask generation is based on true ground truth data, significantly enhancing the reliability of the assessment.

The creation of ground truth point clouds follows a manual process as well. The SfM-derived point clouds are manually cleaned and refined to include only the points belonging to the OOI. This manual cleaning ensures that the point cloud ground truth accurately represents the OOI without extraneous or noisy points, providing a robust baseline for evaluating OOI extraction and point cloud quality.

4.2 Prior-Informed Segmentation

(a) Input image


(b) Segmentation mask of the input image


(c) Rendered view of the 3DGS scene with only selecting Gaussians belonging to the OOI via feature maps

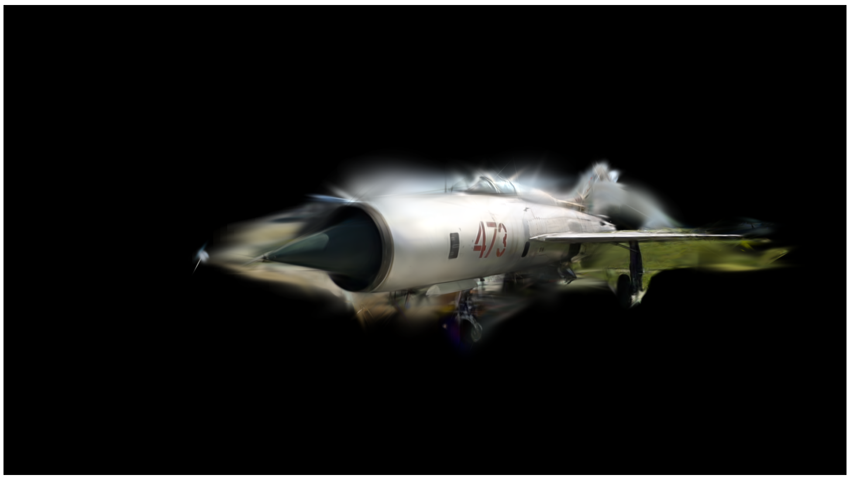

(d) Rendered view of the 3DGS scene using the segmentation mask directly, without feature maps

Figure 4.2: Prior-Informed segmentation using feature maps lead to no improvement over using the masks directly.

Testing showed quantitatively that relying on segmentation masks in conjunction with feature maps does not improve reconstruction quality. In fact, it introduces inefficiencies. While the use of feature maps aimed to enhance segmentation, the process involved additional computational complexity without yielding benefits over using segmentation masks alone. Additionally, the quality of the reconstruction is ultimately constrained by the accuracy of the segmentation masks. Incorporating (threshold-based) clustering of the feature maps introduces even more uncertainty and room for error. Since using plain segmentation masks is already resulting in good results, incorporating feature maps only adds to the complexity and does not improve on the results achieved using segmentation masks only. Figure 4.2c shows the result of using feature maps as input, where only the Gaussians belonging to the OOI are selected and the rest are discarded. Figure 4.2d shows the same but by only using the segmentation mask of the airplane (light green in Figure 4.2b). The reported PSNR, LPIPS and SSIM scores for both methods are virtually the same, as shown in Table 4.1. However, the time it took for the feature map to finish its 30k iterations was significantly longer, almost 3 times as slow as using only the segmentation mask.

	PSNR ↑	LPIPS ↓	SSIM ↑	Time (hh:mm) ↓
Feature Map	23.05	0.102	0.75	1:42
Segmentation Mask	23.12	0.091	0.81	0:35

Table 4.1: Comparison of PSNR, LPIPS, SSIM, and runtime metrics for Feature Map and Segmentation Mask.

4.3 Intrinsically-Informed Segmentation

The results of the experiments are split up into three sections. The first section compares the extracted point cloud of the OOI using the proposed method to the ground truth point cloud. The second section compares the resulting segmentation masks to the ground truth masks. The last section analyses the produced 3D Gaussian Splats.

As ground truth data for the meshed point clouds is unavailable, the meshing stage cannot be directly evaluated. Instead, its performance can be assessed indirectly through the results of the masking stage. If the masks align well with the ground truth, it implies that the mesh accurately represents the geometry and structure of the OOI.

4.3.1 Clustering

The intrinsically-informed segmentation approach outlined in Section 3.4 consists of five stages. The metrics presented in Table 4.2 evaluate the resulting point cloud against the ground truth point cloud for the density clustering, floor removal, and noise removal stages across all scenes.

Chamfer Distance[3] and Hausdorff Distance[13] are metrics to measure the similarity between two point sets. The Chamfer Distance computes the average of the shortest distances from each point in one set to the other, ensuring that all points are well-aligned and capturing overall similarity. It is robust to outliers, as it averages distances rather than focusing on the most extreme discrepancies. In contrast, the Hausdorff Distance considers the maximum distance between the nearest neighbors of the two point sets. This metric identifies the worst-case discrepancy between the sets, making it sensitive to outliers. These distances provide complementary perspectives: Chamfer Distance emphasizes overall shape similarity, while Hausdorff Distance highlights the maximum deviation between the sets. Both are crucial for evaluating point cloud alignments and 3D reconstruction accuracy. The ratio of the Hausdorff distance to the bounding box diagonal provides a normalized measure of the largest discrepancy between the two point sets. For instance, if the bounding box diagonal is 1 meter, this ratio indicates the proportion of the scene's extent corresponding to the largest mismatch between the point clouds. This helps contextualize the Hausdorff metric relative to the scale of the scene.

The reported metrics also show how many points from the filtered point clouds are exact matches, missing, or extra points to the ground truth point cloud. Lastly, the recall, precision, and F1 Score are reported.

		Chamfer ↓	Hausdorff ↓	HD/BB ↓	Matches ↑	Missing ↓	Extra ↓	Recall↑ / Precision↑ / F1↑
Kitchen	Density	0.28121	1.306	0.472	37,630	8	97,984	0.999 / 0.277 / 0.434
	Floor	0.00920	1.283	0.463	35,423	2,215	1,042	0.941 / 0.971 / 0.956
	Noise	0.00038	0.445	0.161	35,423	2,215	943	0.941 / 0.974 / 0.957
Bicycle	Density	0.43860	2.329	0.612	6,334	0	10,643	1.000 / 0.373 / 0.543
	Floor	0.01661	2.328	0.611	6,272	62	475	0.990 / 0.930 / 0.959
	Noise	0.00264	0.478	0.125	6,272	62	443	0.990 / 0.934 / 0.961
Bonsai	Density	0.22608	1.906	0.641	25,220	0	101,654	1.000 / 0.199 / 0.332
	Floor	0.00127	1.351	0.455	24,710	510	216	0.980 / 0.991 / 0.986
	Noise	0.00031	0.636	0.214	24,710	510	181	0.980 / 0.993 / 0.986
Garden	Density	0.99917	2.349	0.479	14,055	0	37,227	1.000 / 0.274 / 0.430
	Floor	0.64211	2.349	0.479	13,175	880	3,452	0.937 / 0.792 / 0.859
	Noise	0.00101	0.697	0.142	13,171	884	142	0.937 / 0.989 / 0.963
Average	Density	0.48627	1.972	0.551	-	-	-	0.999 / 0.280 / 0.434
	Floor	0.16730	1.827	0.502	-	-	-	0.962 / 0.921 / 0.940
	Noise	0.00109	0.564	0.161	-	-	-	0.962 / 0.973 / 0.967

Table 4.2: Combined metrics for Kitchen, Bicycle, Bonsai, and Garden point clouds, categorized by processing steps (density clustering, floor removal, noise removal), with average metrics at the bottom.

The results in Table 4.2 illustrate the effectiveness of the pipeline in isolating and refining the OOI across various scenes. For each scene, the metrics show a significant improvement in accuracy and a reduction in error as the pipeline progresses through its stages.

In the initial density clustering stage, the focus is on broadly removing regions that are not dense enough to belong to the OOI, such as sparse outliers or points far from the object. At this stage, the Chamfer and Hausdorff distances are relatively high, as the point cloud still includes large portions of irrelevant structures like the floor or background elements. These higher values are expected because the floor and other non-OOI points remain close to the OOI.

The floor removal stage narrows down the point cloud by identifying and removing points that belong to the floor, which comprises a significant portion of the non-OOI data. This stage results in a significant reduction in the Chamfer distance, reflecting the removal of large irrelevant regions. However, the Hausdorff distance remains relatively unchanged, suggesting that the most extreme outliers may not belong to the floor but instead originate from noise. Alternatively, this could indicate that the floor removal step was not entirely effective, leaving some distant floor points that remain further away from the OOI. The recall decreases due to the risk of accidentally removing some OOI-adjacent points near the floor.

The noise removal stage sees its most significant improvements in the Chamfer and Hausdorff distance and precision. The missing points do not increase except in Garden, where a minimal increase in missing points is observed.

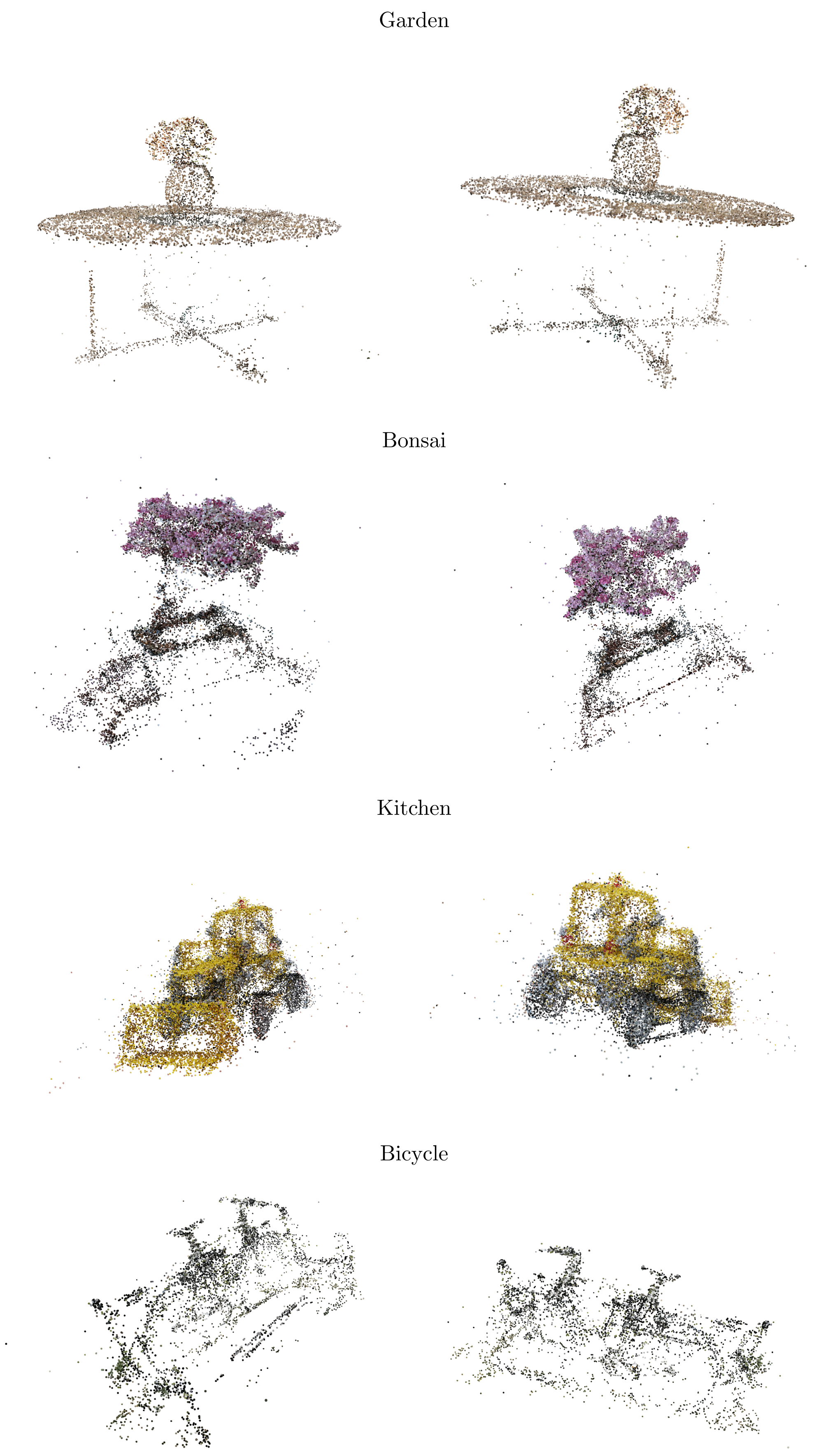


Figure 4.3: Point cloud results of each scene

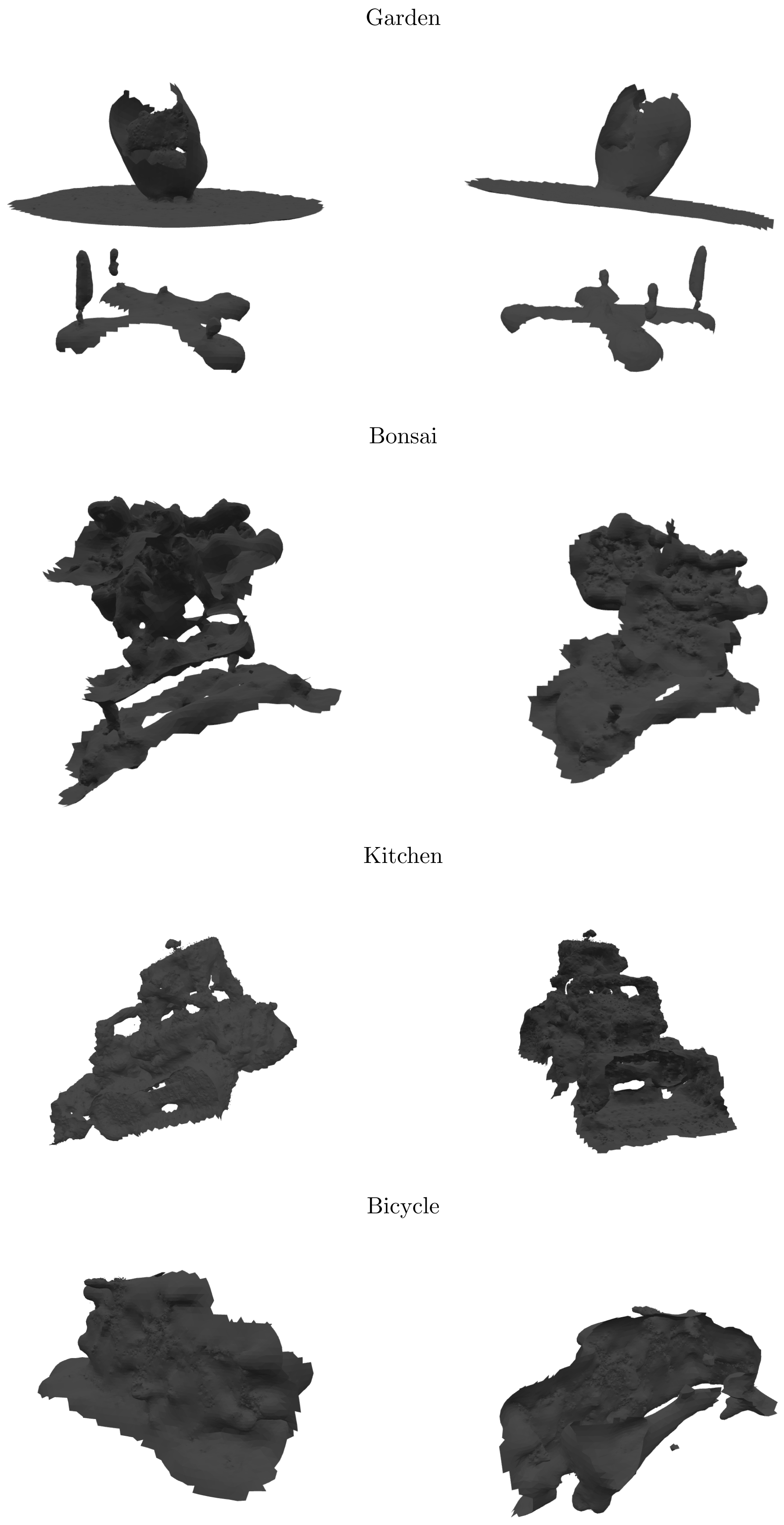


Figure 4.4: Mesh results of each scene. The bicycle scene is difficult to recognize: The left image shows the scene from the side of the bicycle, while the right image shows the scene from the side of the bench.

4.3.2 Meshing

The generated meshes from the point clouds show that PSR is not an ideal surface reconstruction method, even for sparse point clouds. For example, the table legs in the garden scene in Figure 4.4 are barely present, despite being generally visible in the point cloud seen in Figure 4.3. PSR's robustness to noise is therefore also a drawback when it comes to sparse point clouds.

4.3.3 Masking

IoU (Intersection over Union) measures the average IoU across all test images, where IoU represents the overlap between predicted and ground truth masks relative to their union. It shows the overall accuracy measure of the segmentation. The Dice coefficient, equivalent to the F1 Score for binary segmentation, assesses the balance between precision (the proportion of correctly predicted positive pixels) and recall (the proportion of ground truth pixels correctly identified).

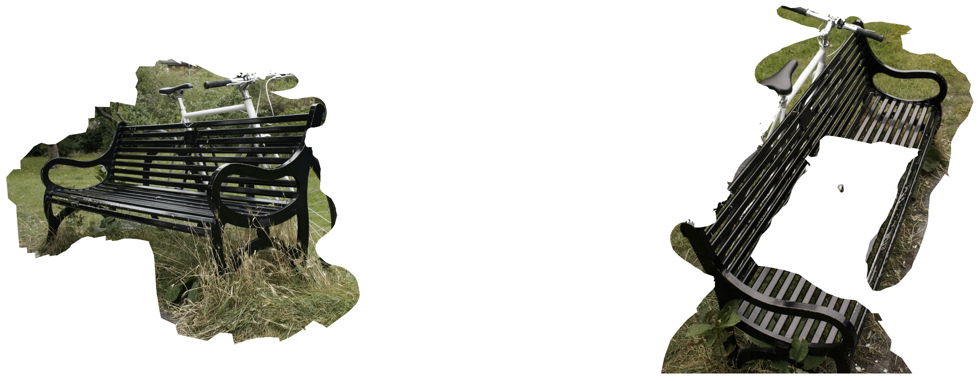
Scene	IoU ↑	Precision ↑	Recall ↑	Dice Coeff ↑
Bicycle (20 images)	0.5549	0.5696	0.9592	0.7123
Bonsai (20 images)	0.6932	0.7205	0.9484	0.8175
Garden (20 images)	0.7706	0.7865	0.9728	0.8694
Kitchen (all images)	0.8646	0.9059	0.9501	0.9273
Average	0.72082	0.74562	0.95762	0.83162

Table 4.3: Binary segmentation mask metrics for Bicycle, Bonsai, Garden, and Kitchen scenes. IoU: Intersection over Union.

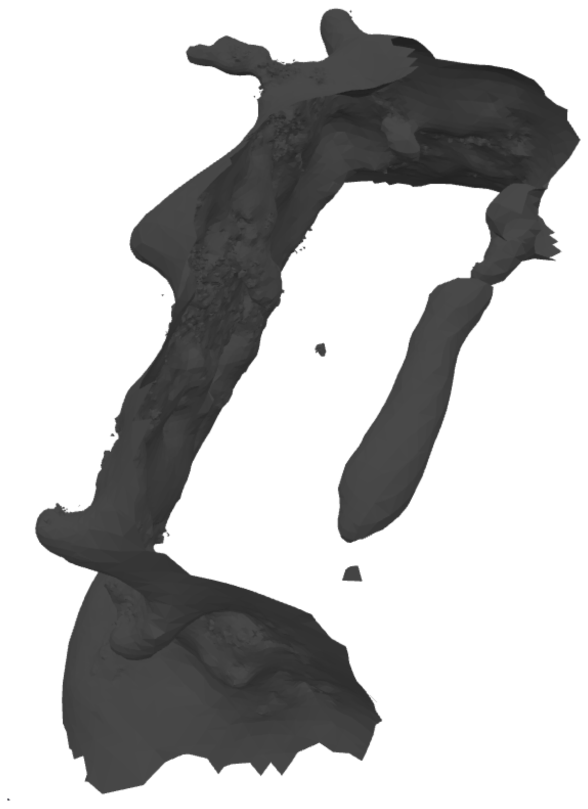
The binary segmentation mask metrics in Table 4.3 report the performance of the segmentation pipeline across four different scenes.

The bicycle and bonsai scenes achieved low IoU and precision scores. This is as expected, as the bicycle scene contains a lot of small vegetation and grass in the immediate area around the OOI. The bonsai OOI contains many details and has a complex structure, which cannot be captured accurately with a relatively sparse point cloud. On the other hand, the kitchen scene performs the best. Since the kitchen OOI has a more straightforward outline than the bonsai OOI, fewer points are needed to represent the OOI's structure accurately. Garden loses out mostly in precision since the legs of the table in the SfM point cloud are only represented by sparse points.

As discussed in Section 1.1, the transition from 3D space to 2D inherently results in information loss. Figure 4.5 illustrates how the resulting masks can vary significantly in coverage due to the projection process of the mesh. For example, while the seating area of the bench is not part of the mesh, the mask generated when viewing the bicycle from the side of the bench includes the seating area because the bicycle, located behind the bench, is meshed. This creates inconsistencies in the masked images across different viewpoints, which are reflected in the final 3DGS result, as further elaborated in Section 4.3.4.



(a) Mask completely covers OOI from this viewpoint (b) Mask misses part of the OOI from this viewpoint



(c) Mesh of OOI

Figure 4.5: Inconsistent masks of the OOI depending on the viewpoint. This is due to the mesh not covering every part of the OOI in 3D, but when projected to 2D, the OOI is covered.

4.3.4 3D Gaussian Splatting

Three standard metrics are used: Learned Perceptual Image Patch Similarity (LPIPS) [40], Structural Similarity Index Measure (SSIM) [42], and Peak Signal-to-Noise Ratio (PSNR). LPIPS quantifies perceptual differences between rendered images and ground truth using deep neural networks trained on human visual similarity. A lower LPIPS value indicates that the rendered image is visually closer to the ground truth, which captures fine-grained perceptual quality. SSIM assesses the structural consistency between the images, focusing on luminance, contrast, and texture. Higher SSIM values indicate better preservation of the structural details and overall appearance of the original image. Finally, PSNR measures the pixel-wise fidelity of the rendered image compared to the ground truth. Higher PSNR values suggest a lower level of noise or distortion in the reconstructed image.

Table 4.4 shows the quantitative results. ”MCMC” refers to the 3DGS MCMC [20] method and serves as a reference, whose metrics are taken directly from the paper. ”Masked” uses the same MCMC method but evaluates the scene by masking the OOI in the ground truth images, thereby focusing on the reconstruction quality of just the OOI with the MCMC method. Lastly, "Ours" refers to the method where the 3DGS of the OOI is being reconstructed directly using the masks generated from this thesis' proposed method. To keep the results compatible with [20], the images were downsampled by a factor of two.

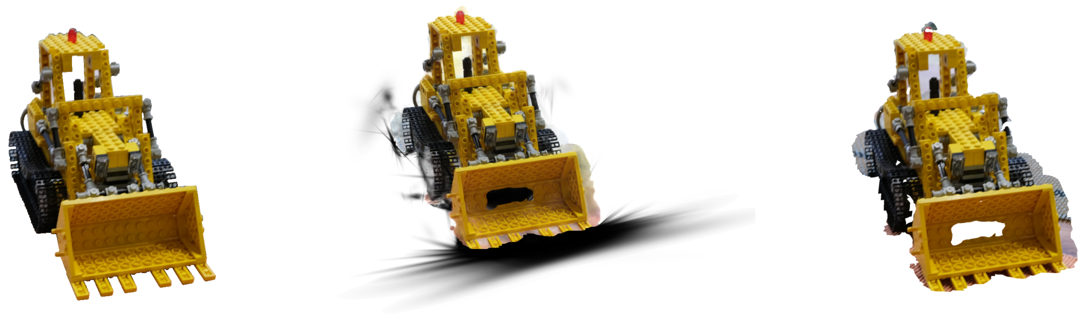
The parameters used for the "Masked" method are identical to those in 3DGS MCMC [20], including a scale regularizer of $\lambda_\Sigma = 0.01$, an opacity regularizer of $\lambda_o = 0.01$, and the maximum number of Gaussians set to match the values used in the original 3DGS method [18] ($n_{kitchen} = 1800000$, $n_{bonsai} = 1300000$, $n_{bicycle} = 5900000$, $n_{garden} = 5200000$), with all other parameters left at their default settings. In the "Ours" method, the parameters remain consistent except for the maximum number of Gaussians, which is reduced. To determine this reduction, the 3DGS OOI generated using the "Masked" method was filtered from the full scene to estimate the number of Gaussians contributing to the OOI. These values were then used to set the maximum number of Gaussians: $n_{kitchen} = 500000$, $n_{bonsai} = 120000$, $n_{bicycle} = 500000$, and $n_{garden} = 450000$.

For both "Masked" and "Ours," the noise learning rate was adjusted from $\lambda_{noise} = 50000$ (used in 3DGS MCMC) to $\lambda_{noise} = 2000$. The higher noise learning rate in 3DGS MCMC was designed to balance scene-wide exploration, ensuring the entire scene could be reconstructed, while preventing stray Gaussians from shooting into regions outside the view frustum. However, in this work, where Gaussians are initialized from the SfM point cloud with higher point density around the OOI, the goal is to achieve high-fidelity reconstruction of the OOI while discarding the rest of the scene. By using a lower noise learning rate, the Gaussians are better able to finely explore the tight space on and around the OOI, without the need to reconstruct the background, which lies further away from the OOI.

	Method	SSIM ↑	PSNR ↑	LPIPS ↓
Bicycle	MCMC	0.810	26.150	0.180
	Masked	0.770	29.377	0.037
	Ours	0.645	26.186	0.061
Kitchen	MCMC	0.940	32.270	0.140
	Masked	0.911	36.101	0.014
	Ours	0.880	31.403	0.021
Bonsai	MCMC	0.950	32.880	0.220
	Masked	0.903	36.058	0.016
	Ours	0.822	32.803	0.025
Garden	MCMC	0.890	28.160	0.100
	Masked	0.904	34.956	0.033
	Ours	0.870	30.270	0.042

Table 4.4: 3DGS metrics (SSIM, PSNR, LPIPS) for Bicycle, Kitchen, Bonsai, and Garden scenes, comparing MCMC[20], masked MCMC, and this thesis' methods.

A direct comparison between the "Masked" and "Ours" methods and "MCMC" is not entirely meaningful, as "Masked" and "Ours" focus on the reconstruction quality of the OOI, whereas "MCMC" evaluates the quality of the entire scene. Nevertheless, "MCMC" provides a useful point of reference. This distinction also explains why the reported PSNR and LPIPS scores tend to improve, while SSIM generally decreases when transitioning from "MCMC" to "Masked." PSNR, which measures pixel-wise fidelity, benefits from the removal of irrelevant or noisy parts of the scene through masking, as the evaluation becomes concentrated on the OOI, where reconstruction is more precise. Similarly, LPIPS, a metric for perceptual similarity, improves due to the exclusion of less accurate or noisy background regions, allowing the metric to focus on the high-quality reconstruction of the OOI. The reason for the generally lower SSIM in "Masked" compared to "MCMC" is less clear, but one possibility is that masking introduces abrupt transitions or less consistent structures in the remaining region, which SSIM penalizes. Even when the OOI itself is reconstructed well, such inconsistencies can negatively impact the SSIM score. While these metric shifts make sense mathematically, they make clear that masked and full-scene evaluations are fundamentally different and not directly comparable.



(a) Ground truth masked image (b) Rendered image (c) Generated masked image

Figure 4.6: Ground truth masked image and generated masked image from the proposed method.

The results show a significant loss in reconstruction quality across the board. This is to be expected, since the generated masks are not perfect, as seen in Table 4.3. Figure 4.6 shows clearly that despite 3D segmentation done via meshing, described in Section 3.4.4, undersegmentation leads to masks that are not consistent across all viewpoints, since the segments that do not belong to the OOI represent a different part of the background depending on the viewpoint. This flaw is showcased in Figure 4.7.

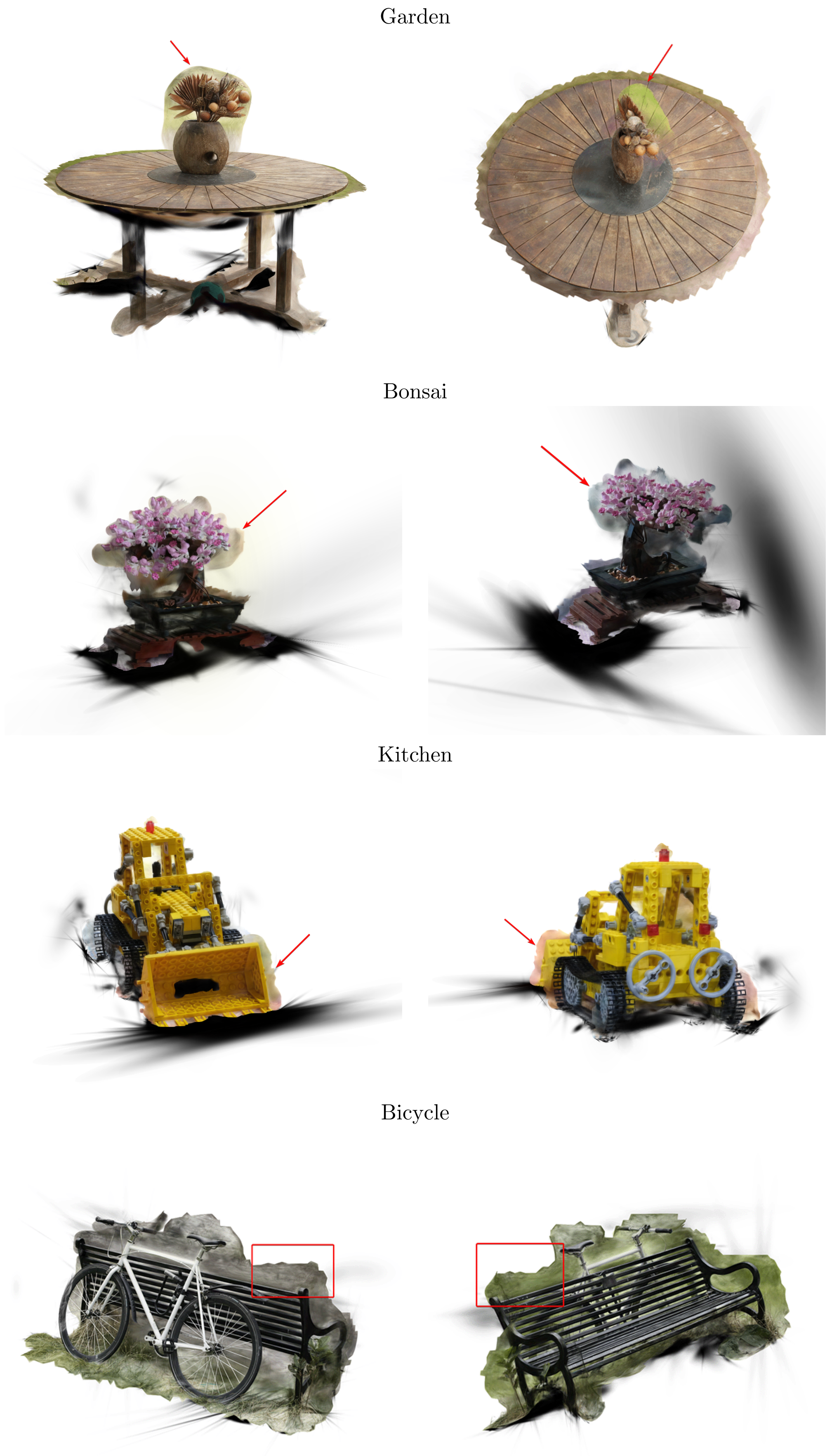


Figure 4.7: View inconsistencies in the 3DGS reconstruction due to undersegmentation in the meshing process. The black Gaussians and the areas marked with red arrows/rectangles showcase the view inconsistencies.

Chapter 5

Conclusion

This thesis presented an approach to segmentation within 3D Gaussian Splatting, focusing on an intrinsically-informed method to perform object-specific reconstructions without reliance on pre-segmented data. While the results demonstrate promising advances in isolating objects of interest using density-based clustering, challenges remain in achieving full reconstruction coverage of the OOI.

The prior-informed approach using feature maps significantly increased computation time, proving to be a slower proxy for directly using segmentation masks. In hindsight, the feature map approach, being another representation of the segmentation mask, was unlikely to outperform it; an oversight during the development process.

For this reason, the intrinsically-informed method was pursued. Despite challenges arising from the sparse nature of the initial SfM point cloud, the results are promising: the OOI was effectively separated from the rest of the scene and used to create a 3DGS reconstruction.

The primary research question - *How can information-guided Gaussian splatting be developed to selectively enhance the reconstruction quality of targeted objects within a 3D Gaussian Splat scene?* - was not fully answered due to not being able to enhance reconstruction quality itself, further refinement of this approach might achieve the initial goal.

The subquestion - *What types of information can be most effectively utilized to guide the Gaussian Splatting process for selective detail enhancement?* - was addressed. It is evident that feature maps, being no different from segmentation masks, are an unsuitable type of information to guide the Gaussian Splatting process. However, using the Gaussian distribution of the scene itself proves to be a valuable type of information, as it is already intrinsically included in any Gaussian Splat. The downside of this approach is its reliance on the OOI being the single prominent object in the majority of the input images. Datasets that focus on an OOI but do not capture it from all angles, such as forward-facing scenes in [32], are likely unsuitable for this method.

Chapter 6

Future Work

6.1 Initial Point Cloud

This thesis primarily explored the use of sparse point clouds generated from SfM, which introduced challenges due to their inherent sparseness. Certain details of the OOI were not captured, as they were absent in the initial SfM point cloud. For instance, as shown in Figure 3.12b, the surface reconstruction using the BPA method results in a mesh with significant gaps. A denser point cloud would allow for a more complete surface reconstruction of the OOI while avoiding over-construction.

To address this limitation, future work could focus on incorporating the proposed segmentation and reconstruction methods directly into the Gaussian Splatting training loop. For example, a warm-up phase could be introduced, during which additional Gaussians are iteratively added to the scene, resulting in a denser point cloud. This adaptive approach could enhance the detail and fidelity of the reconstruction, ensuring that finer features of the OOI are adequately represented.

6.2 Floor Detection

The current floor detection approach, based on RANSAC, may fail to align the detected plane with the actual floor. This misalignment can cause parts of the OOI to be incorrectly classified as floor inliers or sections of the floor to be treated as outliers. Figure 6.1 demonstrates this issue, where the red points represent the floor, the green points correspond to the OOI, and the blue points mark the detected plane. Due to the detected plane not being parallel to the floor, the inlier region extends into the OOI, leading to inaccuracies in segmentation and floor removal.

A more robust floor detection method could mitigate these challenges. Techniques such as multi-plane RANSAC, adaptive thresholding, or context-aware segmentation could improve the alignment and accuracy of the detected plane. Ensuring proper floor separation would significantly reduce the risk of erroneously removing parts of the OOI near the floor.

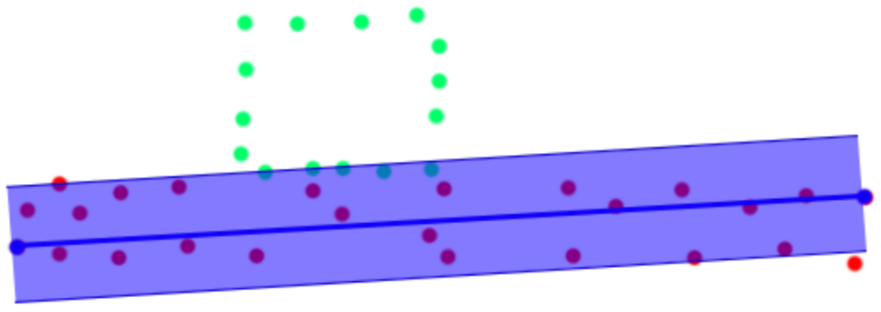


Figure 6.1: Visualization of a potential issue in the floor detection stage. Red points are the floor plane, green the OOI, blue the points RANSAC used to model the floor. The blue area includes the inlier points detected by RANSAC. Note that some of the green points are erroneously counted as inliers.

6.3 Noise Removal

The current implementation of noise removal using DBSCAN is applied only once, which can leave residual noise that obstructs subsequent stages, such as meshing. Future improvements could involve adopting an iterative noise removal process with a dynamic stopping criterion or statistical outlier removal. By refining the noise removal step through successive iterations, the process could adapt to the dataset's characteristics, gradually eliminating more noise without compromising the integrity of the OOI. This iterative approach would likely result in cleaner point clouds, improving the quality of the final surface reconstruction and overall segmentation performance.

Bibliography

[1] Hervé Abdi and Lynne J. Williams. "Principal Component Analysis". In: *WIREs Computational Statistics* 2.4 (2010), pp. 433–459. ISSN: 1939-0068. DOI: 10.1002/wics.101. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/wics.101 (visited on 01/25/2025).

[2] Jonathan T. Barron et al. "Mip-NeRF 360: Unbounded Anti-Aliased Neural Radiance Fields". In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022, pp. 5470–5479. URL: https://openaccess.thecvf.com/content/CVPR2022/html/Barron_Mip-NeRF_360_Unbounded_Anti-Aliased_Neural_Radiance_Fields_CVPR_2022_paper.html (visited on 05/24/2024).

[3] Harry G Barrow et al. "Parametric Correspondence and Chamfer Matching: Two New Techniques for Image Matching". In: *Proceedings: Image Understanding Workshop*. Science Applications, Inc, 1977, pp. 21–27.

[4] Matthew Berger et al. "A Survey of Surface Reconstruction from Point Clouds". In: *Computer Graphics Forum* 36.1 (Jan. 1, 2017), pp. 301–329. ISSN: 1467-8659. DOI: 10.1111/cgf.12802. URL: https://onlinelibrary-wiley-com.utrechtuniversity.idm.oclc.org/doi/10.1111/cgf.12802 (visited on 01/25/2025).

[5] Jiazhong Cen et al. *Segment Any 3D Gaussians*. Dec. 1, 2023. DOI: 10.48550/arXiv.2312.00860. arXiv: 2312.00860 [cs]. URL: http://arxiv.org/abs/2312.00860 (visited on 05/24/2024). Pre-published.

[6] Jiazhong Cen et al. "Segment Anything in 3D with NeRFs". In: *Advances in Neural Information Processing Systems* 36 (Dec. 15, 2023), pp. 25971–25990. URL: https://proceedings.neurips.cc/paper_files/paper/2023/hash/525d24400247f884c3419b0b7b1c4829-Abstract-Conference.html (visited on 05/14/2024).

[7] Guikun Chen and Wenguan Wang. *A Survey on 3D Gaussian Splatting*. Apr. 14, 2024. DOI: 10.48550/arXiv.2401.03890. arXiv: 2401.03890 [cs]. URL: http://arxiv.org/abs/2401.03890 (visited on 06/26/2024). Pre-published.

[8] Richard A. Davis, Keh-Shin Lii, and Dimitris N. Politis. "Remarks on Some Nonparametric Estimates of a Density Function". In: *Selected Works of Murray Rosenblatt*. Springer, New York, NY, 2011, pp. 95–100. ISBN: 978-1-4419-8339-8. DOI: 10.1007/978-1-4419-8339-8_13. URL: https://link.springer.com/chapter/10.1007/978-1-4419-8339-8_13 (visited on 01/26/2025).

[9] Martin Ester et al. "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise". In: *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*. KDD'96. Portland, Oregon: AAAI Press, Aug. 2, 1996, pp. 226–231.

[10] Ben Fei et al. "3D Gaussian Splatting as New Era: A Survey". In: *IEEE Transactions on Visualization and Computer Graphics* (2024), pp. 1–20. ISSN: 1941-0506. DOI: 10.1109/TVCG.2024.3397828. URL: https://ieeexplore.ieee.org/abstract/document/10521791 (visited on 06/20/2024).

[11] *Figure 2. Structure from Motion (SfM) Process Is Illustrated. The...* ResearchGate. URL: https://www.researchgate.net/figure/Structure-from-Motion-SfM-process-is-illustrated-The-structure-in-the_fig2_269327935 (visited on 01/08/2025).

[12] Martin A. Fischler and Robert C. Bolles. "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography". In: *Commun. ACM* 24.6 (June 1, 1981), pp. 381–395. ISSN: 0001-0782. DOI: 10.1145/358669.358692. URL: https://dl.acm.org/doi/10.1145/358669.358692 (visited on 01/26/2025).

[13] Felix Hausdorff. *Grundzüge Der Mengenlehre*. Vol. 7. von Veit, 1914.

[14] Yong He et al. *Deep Learning Based 3D Segmentation: A Survey.* July 26, 2023. DOI: 10.48550/arXiv.2103.05423. arXiv: 2103.05423 [cs]. URL: http://arxiv.org/abs/2103.05423 (visited on 07/21/2024). Pre-published.

[15] Xu Hu et al. *Segment Anything in 3D Gaussians.* Feb. 1, 2024. DOI: 10.48550/arXiv.2401.17857. arXiv: 2401.17857 [cs]. URL: http://arxiv.org/abs/2401.17857 (visited on 05/14/2024). Pre-published.

[16] Jiajun Huang and Hongchuan Yu. *Point'n Move: Interactive Scene Object Manipulation on Gaussian Splatting Radiance Fields.* Nov. 28, 2023. DOI: 10.48550/arXiv.2311.16737. arXiv: 2311.16737 [cs]. URL: http://arxiv.org/abs/2311.16737 (visited on 05/24/2024). Pre-published.

[17] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. “Poisson Surface Reconstruction”. In: *Proceedings of the Fourth Eurographics Symposium on Geometry Processing.* Vol. 7. 4. 2006.

[18] Bernhard Kerbl et al. “3D Gaussian Splatting for Real-Time Radiance Field Rendering”. In: *ACM Transactions on Graphics* 42.4 (July 26, 2023), 139:1–139:14. ISSN: 0730-0301. DOI: 10.1145/3592433. URL: https://dl.acm.org/doi/10.1145/3592433 (visited on 05/01/2024).

[19] Alireza Khatamian and Hamid R. Arabnia. “Survey on 3D Surface Reconstruction”. In: *Journal of Information Processing Systems* 12.3 (Sept. 30, 2016), pp. 338–357. DOI: 10.3745/JIPS.01.0010. URL: https://doi.org/10.3745/JIPS.01.0010 (visited on 01/25/2025).

[20] Shakiba Kheradmand et al. *3D Gaussian Splatting as Markov Chain Monte Carlo.* June 16, 2024. DOI: 10.48550/arXiv.2404.09591. arXiv: 2404.09591 [cs]. URL: http://arxiv.org/abs/2404.09591 (visited on 07/17/2024). Pre-published.

[21] Chung Min Kim et al. *GARField: Group Anything with Radiance Fields.* Jan. 17, 2024. DOI: 10.48550/arXiv.2401.09419. arXiv: 2401.09419 [cs]. URL: http://arxiv.org/abs/2401.09419 (visited on 05/14/2024). Pre-published.

[22] Alexander Kirillov et al. *Segment Anything.* Apr. 5, 2023. DOI: 10.48550/arXiv.2304.02643. arXiv: 2304.02643 [cs]. URL: http://arxiv.org/abs/2304.02643 (visited on 05/22/2024). Pre-published.

[23] Arno Knapitsch et al. “Tanks and Temples: Benchmarking Large-Scale Scene Reconstruction”. In: *ACM Trans. Graph.* 36.4 (July 20, 2017), 78:1–78:13. ISSN: 0730-0301. DOI: 10.1145/3072959.3073599. URL: https://doi.org/10.1145/3072959.3073599 (visited on 07/21/2024).

[24] Sosuke Kobayashi, Eiichi Matsumoto, and Vincent Sitzmann. “Decomposing NeRF for Editing via Feature Field Distillation”. In: *Advances in Neural Information Processing Systems* 35 (Dec. 6, 2022), pp. 23311–23330. URL: https://proceedings.neurips.cc/paper_files/paper/2022/hash/93f250215e4889119807b6fac3a57aec-Abstract-Conference.html (visited on 07/19/2024).

[25] Georgios Kopanas et al. “Point-Based Neural Rendering with Per-View Optimization”. In: *Computer Graphics Forum* 40.4 (2021), pp. 29–43. ISSN: 1467-8659. DOI: 10.1111/cgf.14339. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.14339 (visited on 07/21/2024).

[26] Boyi Li et al. *Language-Driven Semantic Segmentation.* Apr. 2, 2022. DOI: 10.48550/arXiv.2201.03546. arXiv: 2201.03546 [cs]. URL: http://arxiv.org/abs/2201.03546 (visited on 07/19/2024). Pre-published.

[27] Yichen Liu et al. *SANeRF-HQ: Segment Anything for NeRF in High Quality.* Apr. 6, 2024. DOI: 10.48550/arXiv.2312.01531. arXiv: 2312.01531 [cs]. URL: http://arxiv.org/abs/2312.01531 (visited on 05/14/2024). Pre-published.

[28] David G. Lowe. “Distinctive Image Features from Scale-Invariant Keypoints”. In: *International Journal of Computer Vision* 60.2 (Nov. 1, 2004), pp. 91–110. ISSN: 1573-1405. DOI: 10.1023/B:VISI.0000029664.99615.94. URL: https://doi.org/10.1023/B:VISI.0000029664.99615.94 (visited on 01/08/2025).

[29] *Mainblades Interview Notes.* May 29, 2024.

[30] *Mainblades Lightning Strike Inspections.* URL: https://www.mainblades.com/case-studies/lightning-strike-inspections (visited on 05/22/2024).

[31] *Mainblades Wing Inspection.* URL: https://www.mainblades.com/case-studies/wing-inspection (visited on 05/22/2024).

[32] Ben Mildenhall et al. *Local Light Field Fusion: Practical View Synthesis with Prescriptive Sampling Guidelines*. May 2, 2019. DOI: 10.48550/arXiv.1905.00889. arXiv: 1905.00889 [cs]. URL: http://arxiv.org/abs/1905.00889 (visited on 01/26/2025). Pre-published.

[33] Ben Mildenhall et al. *NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis*. Aug. 3, 2020. DOI: 10.48550/arXiv.2003.08934. arXiv: 2003.08934 [cs]. URL: http://arxiv.org/abs/2003.08934 (visited on 05/22/2024). Pre-published.

[34] Emanuel Parzen. "On Estimation of a Probability Density Function and Mode". In: *The Annals of Mathematical Statistics* 33.3 (1962), pp. 1065–1076. ISSN: 0003-4851. JSTOR: 2237880. URL: https://www.jstor.org/stable/2237880 (visited on 01/26/2025).

[35] Adam Paszke et al. *PyTorch: An Imperative Style, High-Performance Deep Learning Library*. Dec. 3, 2019. DOI: 10.48550/arXiv.1912.01703. arXiv: 1912.01703 [cs]. URL: http://arxiv.org/abs/1912.01703 (visited on 01/26/2025). Pre-published.

[36] Alec Radford et al. "Learning Transferable Visual Models From Natural Language Supervision". In: *Proceedings of the 38th International Conference on Machine Learning*. International Conference on Machine Learning. PMLR, July 1, 2021, pp. 8748–8763. URL: https://proceedings.mlr.press/v139/radford21a.html (visited on 07/21/2024).

[37] Johannes L. Schonberger and Jan-Michael Frahm. "Structure-From-Motion Revisited". In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016, pp. 4104–4113. URL: https://www.cv-foundation.org/openaccess/content_cvpr_2016/html/Schonberger_Structure-From-Motion_Revisited_CVPR_2016_paper.html (visited on 07/21/2024).

[38] Erich Schubert et al. "DBSCAN Revisited, Revisited: Why and How You Should (Still) Use DBSCAN". In: *ACM Trans. Database Syst.* 42.3 (July 31, 2017), 19:1–19:21. ISSN: 0362-5915. DOI: 10.1145/3068335. URL: https://dl.acm.org/doi/10.1145/3068335 (visited on 10/18/2024).

[39] Renate Stuurman. *Sparkly Cup*. 2022.

[40] Zhou Wang et al. "Image Quality Assessment: From Error Visibility to Structural Similarity". In: *IEEE Transactions on Image Processing* 13.4 (Apr. 2004), pp. 600–612. ISSN: 1941-0042. DOI: 10.1109/TIP.2003.819861. URL: https://ieeexplore.ieee.org/abstract/document/1284395 (visited on 01/26/2025).

[41] Tong Wu et al. *Recent Advances in 3D Gaussian Splatting*. Apr. 13, 2024. DOI: 10.48550/arXiv.2403.11134. arXiv: 2403.11134 [cs]. URL: http://arxiv.org/abs/2403.11134 (visited on 06/20/2024). Pre-published.

[42] Richard Zhang et al. "The Unreasonable Effectiveness of Deep Features as a Perceptual Metric". In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018, pp. 586–595. URL: https://openaccess.thecvf.com/content_cvpr_2018/html/Zhang_The_Unreasonable_Effectiveness_CVPR_2018_paper.html (visited on 01/26/2025).

[43] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. *Open3D: A Modern Library for 3D Data Processing*. Jan. 30, 2018. DOI: 10.48550/arXiv.1801.09847. arXiv: 1801.09847 [cs]. URL: http://arxiv.org/abs/1801.09847 (visited on 01/26/2025). Pre-published.

[44] Shijie Zhou et al. *Feature 3DGS: Supercharging 3D Gaussian Splatting to Enable Distilled Feature Fields*. Apr. 8, 2024. DOI: 10.48550/arXiv.2312.03203. arXiv: 2312.03203 [cs]. URL: http://arxiv.org/abs/2312.03203 (visited on 07/17/2024). Pre-published.