MBARI Monterey Bay Aquarium Research Institute

GULP3R: Gaussian Underwater spLatting with semantic Perception for 3D Reconstruction.

Creating Labeled 3D Maps from Camera Images

Alessandro Puglisi, University of Montpellier (RSM-LIRMM)

Mentors: Pushyami Kaveti, Giancarlo Troni

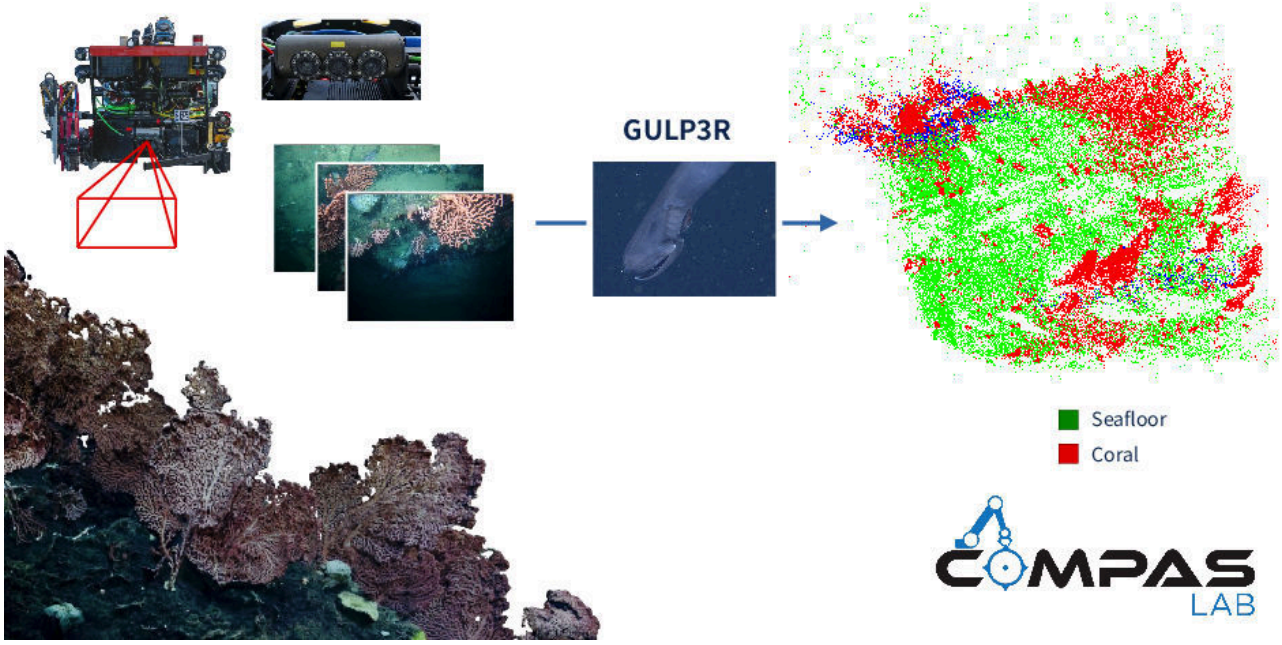
Research Team: CoMPAS Lab

Summer 2026

Keywords: Underwater Semantic Mapping, 3D Reconstruction, Gaussian Splatting

Progress in scientific research is highly dependent on the instrumentation and equipment available to the scientists doing the research.

— *David Packard (1989, Welcoming remarks to The Oceanography Society at its inaugural meeting)*



We propose GULP3R, a pipeline to create labeled 3D models from underwater images.

Abstract

Efforts to map the seafloor date back to the nineteenth century, when depth was sampled at kilometer-scale resolution using manual "lead and line" methods. Today, novel technologies allow us to collect camera images near the seafloor and digitalize them as centimeter-scale 3D models. This work addresses the reconstruction of semantic maps, which are 3D models enhanced with labeled contextual information. Labeled digital reconstructions allow scientists to study geology and ecology of benthic environments more efficiently, as well as support unmanned underwater vehicles' contextual-aware navigation. To achieve our objective, we propose and evaluate a custom pipeline, built with open source algorithms for image enhancement, 3D reconstruction, segmentation, and semantic Gaussian Splatting. We name our framework GULP3R: *Gaussian Underwater spLatting with semantic Perception for 3D Reconstruction*. GULP3R was evaluated on field datasets acquired in Monterey Bay and the Arctic Ocean using Remotely Operated Vehicles (ROVs) from the Monterey Bay Aquarium Research Institute (MBARI) fleet. Validation on our datasets shows quantitatively promising results on 2D novel-view synthesis, and qualitatively cleaner 3D geometric and semantic reconstruction, as well as rendered 2D semantic masks. Quantitative evaluation of 3D metrics and semantics remains future work, due to the lack of ground truth.

The video recording of the symposium presentation is available here:
https://www.youtube.com/watch?v=PzVEjmM61SA

1 Introduction

The use of unmanned underwater robots has made observing and studying the world deep beneath the waves easier. These systems integrate multiple sensors to perceive their surrounding environment, and cameras are ubiquitous due to their affordability, high spatial resolution, and rich visual information compared with many other sensing modalities, such as sonars or LiDARs. Collecting a sequence of images enables the creation of high-resolution three-dimensional reconstructions of the environment, similarly to how humans use multiple views of a scene to build a mental map of the surrounding space to move and interact with it. While ship-mounted acoustic sensors enable kilometer to meter scale reconstruction of the seabed, close-range colored optical images make centimeter-scale reconstructions with appearance information possible. Accurate 3D models are valuable in many scientific applications, from supporting real-time navigation of underwater vehicles to characterizing geological formations, benthic community ecology, ecosystem health, and analyzing them on shore after a scientific expedition. They can also be used for education purposes and to engage with the public (try not to be amazed by some of the models published by the Monterey Bay Aquarium Research Institute, MBARI, on Sketchfab [1]). Classical reconstruction techniques generate geometric models from a sequence of images, without any high-level contextual understanding of the scene [2]. However, full awareness requires knowing what the structures in the map actually are: is it a coral or a rock? What species of coral is it? Is it healthy? Moreover, integrating this information into the reconstruction process can improve the accuracy of the resulting model and support autonomous robotics navigation and exploration with context. The improved accuracy could be obtained by additional reconstruction constraints related to semantic

boundaries. The contextual-aware navigation could be feasible by using the final labeled map, which allow high-level mission planning: ”explore and update the map covering all known bubblegum corals” rather than geometric specifications: ”cover the seafloor with a lawn mower trajectory”. This high-level understanding is known as semantic awareness.

For the rest of the paper the terms semantic/labeled/contextual-aware maps and 3D models are used interchangeable to indicate the output of our framework, while semantic/labeled/contextual-aware reconstruction interchangeably indicate the process itself.

This project addresses the problem of semantic three-dimensional reconstruction of benthic environments from camera images. The scientific relevance and objectives of this research are the following:

- Semantic maps can assist scientists in the study of benthic environments onshore.
- Labeled maps enable the development of algorithms for automatic ecological analysis.
- Contextual-aware benthic maps can be used for smarter robot mission planning, scaling different domains of oceanographic research with new technologies.
- Contextual-aware reconstruction can improve the geometric accuracy of the 3D model.

We investigate semantic mapping by extending the Gaussian Splatting framework, an emerging paradigm in 3D reconstruction and novel view synthesis [3][4]. In this context, novel view synthesis refers to the generation of images from viewpoints that were not captured during dataset acquisition. While foundational algorithms for 3D reconstruction output the structure of the scene as a cloud of 3D points (sparse point cloud), new neural rendering approaches use the sparse point-based map and the captured images to densify the space representation, capturing both 3D geometry and photorealistic appearance. For this reason, we decide to use Gaussian Splatting [3] and one of its semantic extensions [5] to investigate methods to create labeled 3D maps from camera images.

Unlike other works in semantic reconstruction with Gaussian splatting, we focus on the underwater domain, a medium that introduces many unique challenges. With our work, we address two relevant problems of working with underwater images: medium degradation and the scarcity of labeled datasets. The former is related to how the water medium interferes with the light captured by imaging systems, degrading the quality of the data [6][7]. For this reason, we explore and compare different ways of color enhancing underwater images, looking for improvements in the final results of our methodology. The latter depends on the fact that most of the research on semantic understanding is targeted to airborne applications[8][9], where collecting a large amount of data is easier, and labeling doesn't require expertise. Underwater environments instead are less accessible, less studied, highly variable, and require expertise to be annotated. Here, we explore the usage of ready to use foundation models [10], therefore including multi-view inconsistent annotations by pseudo-labeling reference semantics. Multi-view inconsistency is a problem present in any dataset, but it becomes harder when the segmentation model is not fine-tuned for the specific application. Moreover, to the best of our knowledge, there is no prior work focusing on the explicit and queryable semantic labeling

of underwater 3DGS reconstructions, with most existing studies exploiting semantic information only as an additional source of guidance during scene learning [11][12].

Based on the objectives of our research and its challenges, our main technical investigation questions are the following:

- To what extent does underwater imaging degradation affect the final output?
- Is it possible to learn the semantics of the underwater scene using pre-trained deep learning models?
- Can 2D semantic information (labeled images) be projected into 3D sparse models?
- How can semantics be integrated into Gaussian Splatting as a densification strategy for sparse models?
- Can the use of Semantic Gaussian Splatting improve the quality of the labeled 3D map?

To address the points above, we develop and test the GULP3R pipeline, *Gaussian Underwater spLatting with semantic Perception for 3D Reconstruction*. By extensively testing our pipeline on field datasets, the main contributions of our research are:

- Demonstrate the feasibility of using pre-trained foundation models to derive 2D semantic masks on underwater images, used to project semantics onto 3D sparse models.
- Show the results of Semantic Gaussian Splatting in qualitatively improving the semantic reconstruction with respect to a first estimate semantic 3D model. The used Gaussian Splatting algorithm suggests a context-aware removal and redistribution of reconstruction artifacts by learning from input 2D images and semantic masks.
- Introduce new research questions about how underwater image enhancement can improve semantic reconstruction quality with Gaussian Splatting. Our qualitative results with color corrected images don't suggest any improvement in the quality of 3D geometric and semantic reconstruction. However, quantitative evaluation and further model tuning is necessary to derive conclusions.

Extensive research and development of tools like GULP3R will support the scientific understanding of our under-explored and rapidly evolving ocean, as well as other ecosystems. Using these frameworks, a scientist can measure the extent of bubblegum coral, distinguish coral from seafloor, and revisit mapped colonies over time, tracking their growth and health. Additionally, similar pipelines can be extended for scientific applications where scene coverage with an imaging system is possible, such as intertidal surveys, shallow coral reef monitoring, and other habitat studies.

Before introducing the proposed methodology in Section 3, the backbone of our research will be explained in Section 2, introducing the algorithms with a logical order linked to our technical questions and to the working principle of GULP3R. The main algorithms are extensively explained in order to guarantee the understanding of this work for researchers with different backgrounds. The field datasets used for testing our method, the metrics, and the results of GULP3R are presented in Section 4. Additionally, qualitative discussion of the

results is given in Section 5, highlighting the limitations of our pipeline and of the evaluation process, leading to possible future work. Finally, Section 6 concludes this paper with a summary of the research outcome.

2 Background and Related Work

Creating underwater maps with semantics involve different domain of research in the fields of computer vision, computer graphics, and deep learning, and this section will explain the relevant algorithms and the backbone of the proposed pipeline. GULP3R relies on foundational and novel 3D reconstruction techniques, underwater color correction, semantic segmentation, and recent semantic splatting works.

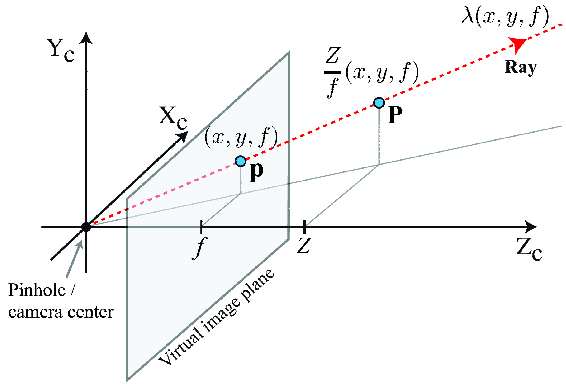
First, the process of making 3D models from 2D images is described in Section 2.1, and the novel paradigms based on Gaussian Splatting (3DGS) are introduced in Section 2.2. The additional complexity of imaging underwater and restoring colors is described in Section 2.3. Finally, the semantic integration process is covered, from the 2D semantic mask estimation in Section 2.4, to the projection of semantics in 3D points and splatting algorithms in Section 2.5.

2.1 Photogrammetry and Structure from Motion

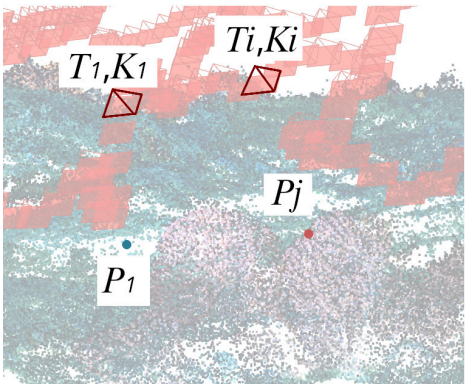
The science of photogrammetry studies how to extract reliable 3D world measurements from 2D photographic images, and Structure from Motion (SfM) is a specific photogrammetric algorithm [13][14][15]. As the name suggests, SfM reconstructs the structure of the scene from the estimation of the imaging camera motion. In the most general case, the SfM algorithm takes as input a sequence of overlapping images $\{I_i\}_{i=1,\dots,N_I}$, meaning that for each image there is at least another one capturing common features, and estimates:

- The cameras' extrinsics $\{T_i\}_{i=1,\dots,N_I}$: Position and orientation (pose) in which each image has been captured, with respect to an arbitrarily chosen world frame.
- The cameras' intrinsics $\{K_i\}_{i=1,\dots,N_I}$: A projection matrix for every different camera configuration, describing how 3D points are mapped into 2D image pixels, based on a chosen camera model [13]. This process is known as intrinsic calibration.
- The structure of the imaged scene $\{P_j\}_{j=1,\dots,N_P}$: A set of N_P points that form what is called the sparse point cloud reconstruction.

See Figure 1b and Figure 1a for a graphical representation of the estimated quantities and the projection process described above.



(a) 3D-to-2D perspective projection [14].

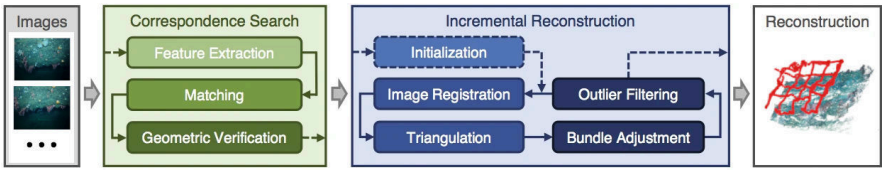


(b) Highlighted camera poses and points [2].

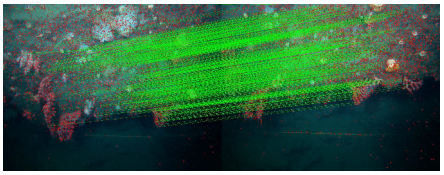
Figure 1: Graphical representation of SfM.

There are different ways in which the SfM problem can be solved, and a broadly used approach is incremental SfM, with an open-source solution proposed in [2]. This incremental pipeline is schematized in Figure 2a with a high level description. The steps are the following:

1. Feature Extraction: identify recognizable points in images' pixels (e.g. clear object edges), stored in explicit descriptors [16][17].
2. Feature Matching: use the descriptors to search for the same features in every couple of images, creating a matching dictionary [18].
3. Initialization: initialize the SfM solution from a first couple of images [13].
4. Incremental Optimization with Filtering: include new images in the solution by minimizing a 3D-to-2D re-projection error, based on the current estimated solution. Outliers rejection strategies are used to avoid completely erroneous solutions.
5. Bundle Adjustment [19]: a computationally intensive process that runs regularly to improve and verify the current solution.



(a) COLMAP's incremental SfM pipeline (adapted from[2]).



(b) Example SIFT[16] matching.

Figure 2: Overview of Incremental-SfM with COLMAP[2].

It is relevant to note how SfM has already been extensively used by scientists to scale the study of geomorphology [20] and coral ecology [21][22] through digital models.

The incremental SfM pipeline emulates how humans mentally reconstruct a surrounding 3D scene by observing it from multiple views and correlating the same features seen in different moments. However, a complete understanding of the sense of the scene goes beyond the geometrical process of SfM, integrating high level

concepts and scene awareness, such as object types and instances. This is why semantic reconstruction is an interesting domain of research.

To conclude, the SfM process described is widely used in many applications, but the sparse reconstruction obtained is not always descriptive enough for scientific research. The next step can be a dense reconstruction of scene surfaces [2][23][24][25]; however, these dense models takes hours to days to be computed and occupy large amounts of disk space (GBs)[2]. Furthermore, this surface reconstruction process usually creates inaccurate 3D models full of noises and artifacts, visible only after the time and computationally intensive process. For these reasons, a paradigm shift is necessary to reconstruct geometrically accurate and denser models in an efficient way. Solving different research questions, recent learning-based rendering methods such as NeRF[26] and 3DGS[3] propose alternative strategies to reconstruct and represent scene appearance from an initial sparse reconstruction. It turns out that these frameworks optimize and densify 3D geometry of the scenes, providing novel state-of-the-art tools for 3D reconstruction.

2.2 Neural Rendering and 3D Gaussian Splatting

An understanding of the concepts of rendering, volumetric rendering, neural rendering, and novel view synthesis is required in order to fully understand 3D Gaussian Splatting (3DGS) and its semantic extension.

Rendering refers to the generation of virtual images from a digital representation of the world, which is what happens in visual effects (VFX) and 3D animated movies. Given a digital description of your scene/world and a pose of a virtual camera, a rendering algorithm produces the corresponding image as it would appear from that viewpoint. Volume rendering is a specific algorithm that uses a volumetric representation of the scene to generate images. It emulate how real imaging works by shooting virtual rays from the digital camera center and considering characteristics like opacity and color of ”floating particles”, encoded in the scene representation. Neural rendering refers to rendering approaches in which the scene representation is learned from data through an optimization process, often involving neural networks or other learnable parametric models. Images are then generated by rendering this learned scene representation. In this context, the goal of novel view synthesis is to generate (synthesize) photorealistic images of a real scene from viewpoints that were not observed during data acquisition. Given a set of photographs captured from multiple viewpoints, a novel view synthesis method predicts how the scene would appear from previously unseen camera poses [27][26].

Neural Radiance Field (NeRF)[26] has been a foundational work on novel view synthesis, relying on a neural network to encode the volumetric information of the scene and render new images. To learn this neural representation from a training set of captured images, a volumetric rendering equation is used to estimate the color $\hat{C}(r)$ of each pixel, by querying the neural network in all the rays $r \in \mathcal{R}$ that influence the virtual camera. The estimated color $\hat{C}(r)$ can then be compared with the real color $C(r)$, as a measure of how close the current model is to the optimal one. Neglecting some of the implementation details, the overall learning

loss $\mathcal{L}$ used to train the network is the following:

$$\mathcal{L} = \sum_{r \in \mathcal{R}} \left\| \hat{C}(r) - C(r) \right\|_2^2 \tag{1}$$

The learning process focuses on estimating the view-dependency of the scene, for photorealistic view synthesis close to the real scene, intrinsically learning the geometric structure. While classical reconstruction technique optimize only geometry and lose any material properties and view-dependency, neural rendering captures both. An overview of the NeRF learning and rendering process is represented in Figure 3. However,

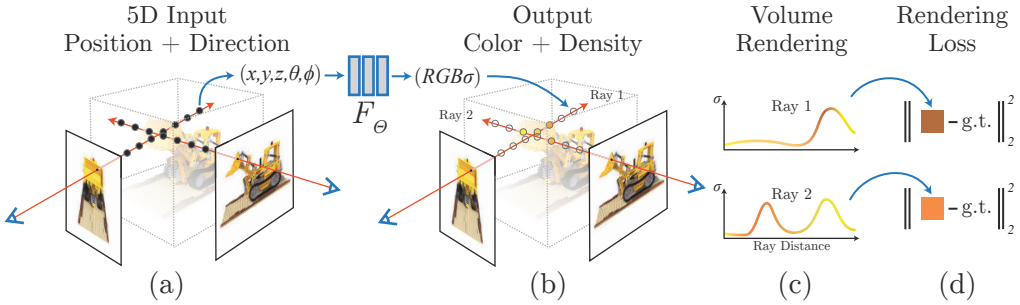


Figure 3: Overview of NeRF pipeline [26].

NeRF comes with multiple problems, such as rendering speed and computational requirements (high GPU memory usage), addressed later by many researchers [28][29][27]. Even after improvements to the baseline method, these neural radiance approaches rely on scene representations often hard to interpret and to edit.

3D Gaussian Splatting (3DGS) [3] has been a paradigm shift in novel view synthesis, introducing an explicit and editable representation of the scene, and a rendering process that exploit a previously proposed volumetric splatting approach[30]. While NeRF used the camera poses as initialization (discarding the SfM sparse model), 3DGS additionally exploits the sparse point cloud in the initialization process, employing a new way of describing the volumetric space: a Gaussian cloud. Gaussian functions are widely used by scientists and engineers to represent the probability distribution of a random variable x, the well-known bell curve, which can be written in normalized form as:

$$G_{1d}(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Where μ represents the mean and σ the standard deviation of the random variable, as represented in Figure 4a. The splatting framework extends this distribution to 3D space, defining Gaussians with a 3D covariance matrix Σ centered at point (mean) μ:

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

This primitive takes the shape of an ellipsoid (see Figure 4b).

With this volumetric representation of the scene, it is possible to render virtual cameras by projecting (splatting) the 3D ellipsoid in the 2D camera plane [30]. The splatting process guarantees a quick rendering approach, in which an explicit representation is directly projected from the scene to the image plane. This is

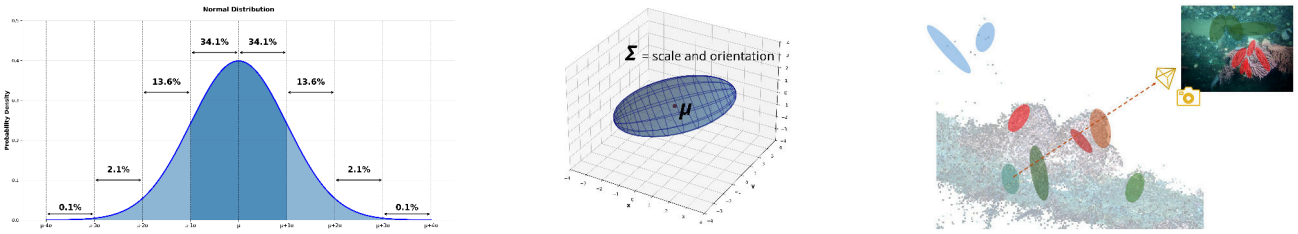
very different from the expensive NeRF volume rendering [26], based on discrete neural querying over the ray. However, the volumetric rendering equation follows the same process, here referred to as α-blending. The view dependency and blending during splatting are made possible by additional embedded parameters. Overall, each Gaussian $G_i(x)$ is represented by:

- mean μ_i: position of the ellipsoid center.
- covariance Σ_i: Combination of scale (S_i) and rotation (R_i) as $\Sigma_i = R_i S_i S_i^T R_i^T$.
- opacity α_i: transparency factor, related to the material properties .
- spherical harmonic (SH) coefficients $c_i(r)$: high-dimensional coefficients as functions of the camera view direction (ray r), encoding color view-dependency.

With the quantities described above, the α-blending (splatting) process renders each pixel color C, corresponding to camera ray r as:

$$C(r) = \sum_{i=1}^{N} \alpha_i c_i(r) T_i, \quad \text{where } T_i = \prod_{j=1}^{i-1} (1 - \alpha_j). \tag{3}$$

T_i is defined as transmittance, depending on the Gaussians' opacity and order. The ordering effect is evident from the index j domain, which considers the Gaussians encountered before index i along the ray. See Figure 4c for a graphical representation of a Gaussian cloud splatting, or Figure 20 for a visualization of a Gaussian cloud reconstructed from real data. 3DGS takes as training input a set of images, the camera



(a) One-dimensional Gaussian. (b) Three-dimensional Gaussian. (c) Gaussian Cloud Splatting [31].

Figure 4: Graphical overview of Gaussian representation and rasterization.

poses, and a sparse point cloud estimation of the scene (computed with an SfM algorithm), and optimizes the Gaussian parameters, number of Gaussians, and distribution in the 3D space. As for NeRF [26], the goal of 3DGS is to obtain a scene representation that renders images as close as possible to the training ones. Again, this process intrinsically optimizes the geometry of the Gaussian cloud, in this case even more evidently due to the explicit geometrical representation of the scene.

The reconstruction process begins by initializing a set of Gaussians whose means are centered at the points of an input point cloud, while their covariance matrices, opacities, and spherical harmonics (SH) color coefficients are randomly initialized. The learning process can then begin without relying on any neural network architecture. Instead, 3DGS employs a non-linear optimization procedure that iteratively adjusts the parameters of each Gaussian to better reproduce the appearance of the training scene. This learning is guided by

minimizing the following loss function $\mathcal{L}$:

$$\mathcal{L} = (1 - \lambda)\mathcal{L}_1 + \lambda\mathcal{L}_{\text{D-SSIM}} \tag{4}$$

Where $\mathcal{L}_1$ corresponds to the photometric difference used in NeRF (see Equation (1)) and $\mathcal{L}_{\text{D-SSIM}}$ is an image similarity index, capturing also the structure of the rendered image with respect to the ground truth training image. The Structural Similarity Index Measure ($SSIM$) will be further described when defining the evaluation metrics (Section 4.2). The scene learning is possible due to the differentiability of Gaussian functions, which makes the overall loss becomes differentiable. Differentiability, in fact, makes it possible to use any gradient descent algorithm to learn the optimal variables of the Gaussian cloud.

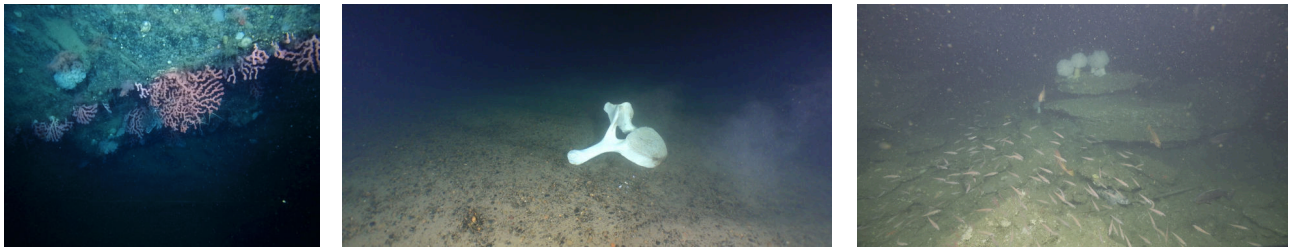
However, the initial number of points in the point cloud and the distribution of those points is a sparse volume representation, and it may not be the optimal one, which is why 3DGS integrates additional densification and pruning strategies over its primitive during training. These strategies remove Gaussians where they are not needed (over-reconstructed scene) and move them where the scene is under-reconstructed, using gradient and opacity information as guidance.

The final 3DGS model allows real-time rendering (120 fps), fast training, and interpretability of the results, which attracts many researchers in improving and adapting the baseline for a plethora of application[4].

Examples such as Scaffold-GS [32] integrate new scene representation and loss regularization strategies to improve convergence speed and photometric performances. Extensions such as SeaSplat [7] include underwater degradation to address the creation of water Gaussian floaters, a limitation of the baseline methodology of 3DGS, which compromise its usage in hazing mediums (such as fog or fluids). To better understand how to work with underwater images, it is necessary to describe the challenges of imaging beneath the waves.

2.3 Underwater Image Enhancement

From the images below (Figure 5), many different forms of degradation and artifacts are visible. The degraded color and hazing effects are created by the water medium, causing multiple forms of disturbances to the imaging system. Additionally, degradations such as blurring of fine details are related to dynamic entities such as sinking particles and fish. Images in Figure 5 are selected to highlight some of the main issues of underwater imaging: selective light absorption (Figure 5a, Figure 5b), backscatter(Figure 5a, Figure 5c), and often unstructured scene full of unexpected dynamic entities(Figure 5c). To better understand these challenges, the main underwater image degradation phenomena are described below, along with their causes and two approaches for their correction.



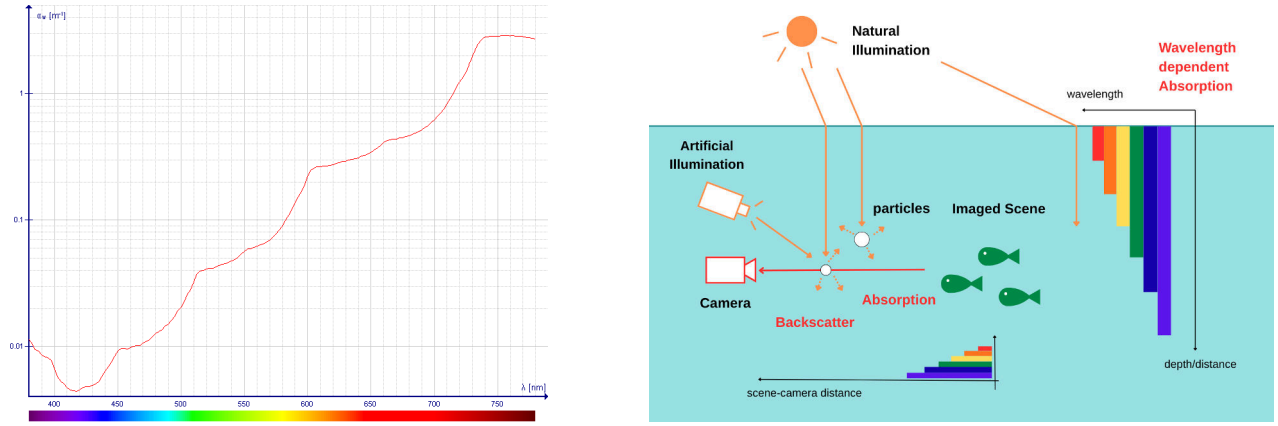
(a) Color Absorption. (b) Darkness and Hazing. (c) Backscatter and Dynamic distractors.

Figure 5: Images captured underwater with minimal or no post-processing.

Light is an electromagnetic wave that is selectively absorbed by the water due to chemical and physical properties of the medium, where selectivity depends on the wavelength. This wavelength-dependent absorption defines the limits of the euphotic (sunlight) zone and necessitates the use of artificial illumination systems for deep-sea imaging. Moreover, the strong attenuation of electromagnetic waves in water renders technologies such as GPS and radio communication ineffective underwater. From the plot in Figure 6a[33] it can be inferred that visible light rays with longer wavelengths are absorbed more, meaning that they travel less distance in the water column before disappearing. Therefore, if an imaging system sinks towards the ocean depth, colors like red (3-5m) and yellow (5-15m) disappear first, then green (80m) and blue (100m) (see Figure 6b). This is evident in Figure 5a, where a normally vibrant red bubblegum coral appears more greenish, and the overall scene seems imaged with some blue-green filter.

When illuminating the darkness, every source of light interact with free floating particles [34] (see Figure 6b), creating a hazy, low-contrast appearance named backscatter (see Figure 5c).

Phenomena like caustics (shadows from sea surface motion) and other shallow water degradation are not considered here.

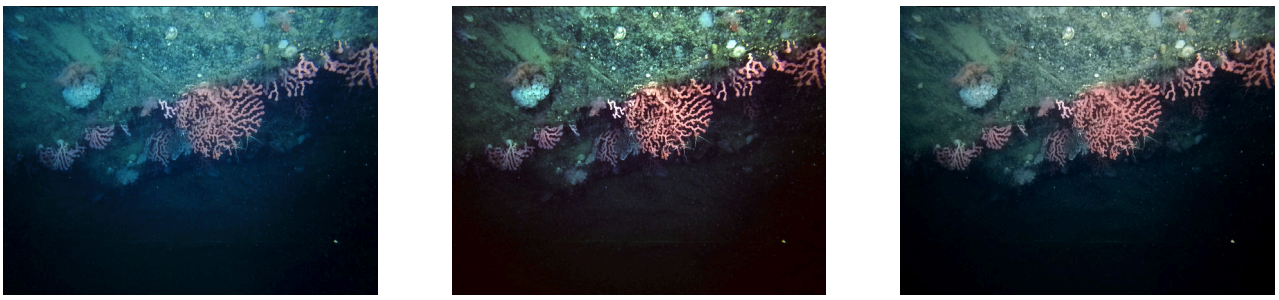


(a) Absorption coefficient over wavelength[33]. (b) Causes of underwater degradation.

Figure 6: Underwater image degradation.

These degradations can compromise algorithms applied to images, either to extract 2D (semantic, photometry) or 3D (geometric) information. For these reasons, researchers come up with different ideas to model/learn the underwater imaging process and correct it [6][35]. The most successful techniques rely either on classical operations applied to image pixels or deep learning architectures trained to estimate the degradation and

correct for it. An example of a classical approach is the Contrast-Limited Adaptive Histogram Equalization (CLAHE) [36], which can be paired with additional digital image processing [37] based on scene assumptions and parameters tuning. CLAHE was proposed for medical imaging, and it properly equalizes the image appearance without amplifying noise. In the learning domain, transformers architectures like Single-Step Latent Diffusion for Underwater Image Restoration (SLURPP) [38] show state-of-the-art performance, without requiring the time consuming fine tuning of classical approaches. An example result of Figure 5a corrected with a custom CLAHE pipeline [37] and SLURPP [38] is reported below (Figure 7)



(a) Underwater Image. (b) CLAHE enhanced. (c) SLURPP enhanced.

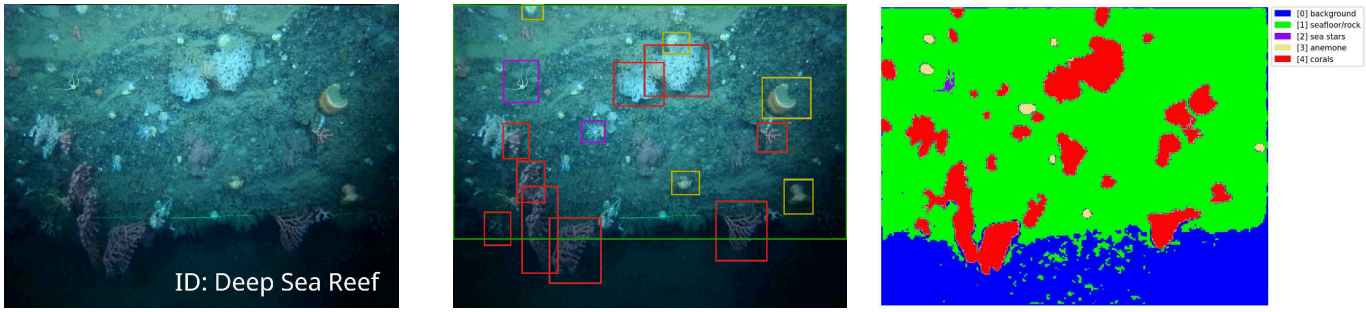
Figure 7: Underwater Image (a) enhanced with CLAHE pipeline (b) and SLURPP (c).

SLURPP [38] is an example of how deep learning models are trained on big datasets to perform operations on two-dimensional data, such as images. The application of learning approaches is fundamental when high level information (semantics) has to be extracted from images. Semantic segmentation and the techniques to extract semantic masks of 2D images are fundamental concepts to embed labels in 3D maps, making semantic reconstruction possible.

2.4 2D Semantic Segmentation

While color enhancement post-processes 2D images to restore real appearance and improve image quality, understanding the content of an image require very different solutions, enabling multiple level of image awareness. In the research domain of high level image analysis, the term ”object” is used for both living beings and non-living objects, while the term ”label” is the text/digit identifier associated. The main techniques to extract different levels of information from 2D images (see Figure 8) are:

- Image Classification: Produce a single label to classify what is present in the image.
- Object Detection: Detect the position of the objects in the scene and associate a class label. A bounding box is centered on the position of each object detected, with a dimension related to the occupied pixels.
- Semantic Segmentation: Label each pixel in the image, achieving a semantic understanding of the scene. In case of multiple instances of the same type (e.g. two corals), they are classified with the same label.
- Instance Segmentation: Label each pixel in the image. Unlike semantic segmentation, here one label for each separate entity is used, even if they are the same type of object.



(a) Image Classification. (b) Object Detection. (c) Semantic Segmentation.

Figure 8: Different levels of image understanding (class labels are indicated by colors).

The techniques described above give a different level of semantic resolution, and image classification algorithms are the lowest level. The single output label from image classification is informative for simple images with a single entity or to get a general idea of the scene, but not to have 3D awareness in a complex scenario like the one in Figure 8a. Object detection, on the other hand, gives a more informative data output, but still doesn't provide high resolution details on object contours and clear boundaries between instances. Semantic and instance segmentation techniques provide a semantic mask of the scene (as in Figure 8c), which is a full classification in the pixel space. Each 2D pixel in the mask has a label corresponding to the object in the original colored image pixel, achieving a higher level of understanding with respect to object detection algorithms.

Researchers have been investigating how to make these levels of understanding possible, first with classical approaches based on image features [16], explicit modeling, and processing steps. The advent of deep learning architectures and Convolutional Neural Networks (CNNs) has completely transformed image understanding research. Semantic understanding is continuously improving due to competitions like ImageNet [8] and architectures like Residual Network (ResNet)[39], You Only Look Once (YOLO)[40], and transformer-based architectures [41][42].

Recently, new powerful models emerged in the domains of semantic segmentation, and deep learning research, referred to as foundation models. Foundation models are large-scale architectures trained on vast and diverse datasets with the goal of generalizing across a wide range of tasks and application domains, in contrast to traditional models designed or fine-tuned for a specific task. Segment Anything Model 3 (SAM3) [10] is an example of this emerging class of models. Trained on massive datasets, including FathomNet [43], SAM3 is capable of producing high-quality segmentations across a variety of environments. FathomNet is a large-scale ocean imagery database, and its inclusion in the training data enables SAM3 to better understand and segment underwater scenes. In particular, SAM3 [10] has a Vision Transformers (ViT) based architecture, merging the visual and textual domain, enabling segmentation from text input. As a state-of-the-art model for semantic segmentation, SAM3 can be used to generate pseudo-labels to train models in domains where labeled data are scarce [44]. The process of pseudo-labeling consists of using data labeled by a high-performance deep learning model to train another model, bypassing the time-consuming hand-labeling step. Pseudo labels can

then be used iteratively to improve the performance of the model. In the process of underwater semantic 3D reconstruction, ground-truth segmented masks would avoid problems such as view-dependent semantic inconsistency, but pseudo-labeled masks are used in practical application to integrate semantics in 3D. Therefore, all the training semantic masks reported in this report are labeled with SAM3 and are not to be considered as ground truths but as pseudo-labels. This concept will be discussed again in Section 3.3.

To conclude, novel technologies allow us to create semantic masks without intensive dataset collection and labeling effort. This opens the possibility of contextually understanding 3D scene and integrating labels in the 3D reconstruction process.

2.5 Semantic 3D Reconstruction

Embedding semantic information in 3D algorithms is an active area of research, with the main approaches relying either on a direct segmentation in 3D space or on projecting 2D masks on the 3D model. Additionally, semantics can be used as guidance to improve the final quality of the reconstruction, or as added value to the model, integrating semantic information and enabling more applications.

Using classical approaches, clustering based on point cloud patterns allows a first rough segmentation, but this would never capture the semantic complexity of the unstructured scenes. Instead, deep-learning architectures try to segment directly in 3D, either with supervised[45] or self-supervised[46] models. In deep learning, supervised models require a labeled training dataset to learn the optimal parameters. Self-supervised models, instead, do not rely on any explicit input-output information for training. Therefore, if segmentation models fine-tuned for the application of interest are not available, and labeled data are scarce, self-supervised solutions are preferred.

Semantic Lifting is the process of projecting semantics in 3D from segmented images. This process requires both a 3D model and the semantic masks of the image dataset used for the reconstruction. As represented in Figure 9, relying on the SfM outputs (see Section 2.1), each 3D point in the sparse point cloud can be re-projected to the corresponding 2D pixel of every camera image where that point is visible. This re-projection can be computed by using the cameras' extrinsics (pose) and intrinsics (projection model). If, then, for each image a semantic mask is available, the class of each pixel is known from multiple views, and a voting strategy can be used to chose the label of that 3D point. An example voting strategy is taking the most common class. This simple lifting process generates a semantic point cloud, where each 3D point embeds an additional label dimension (enhancing the points from 3D to 4D). Further details on voting strategies for labeling 3D points from 2D masks are given in Appendix B.

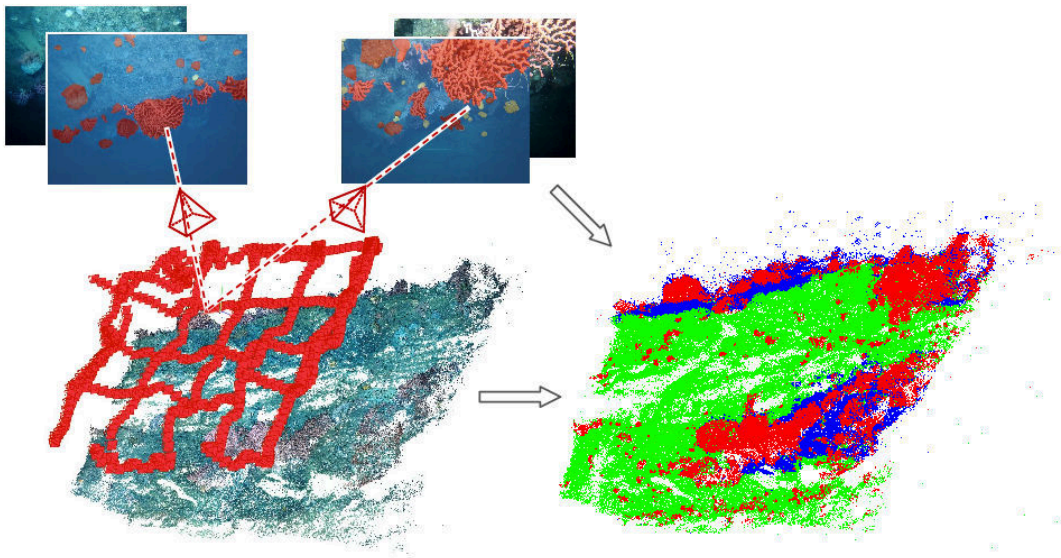


Figure 9: Semantic Lifting on point cloud. Each 3D point is re-projected to the corresponding 2D pixels in the semantic masks where it is visible, then a voting strategy is used to assign a 3D semantic label from multiple (inconsistent) 2D labels.

DeepReefMap[47] uses this lifting approach to generate semantic point clouds of coral reefs, using pre trained semantic segmentation models and learning-based SfM approaches, enabling spatio-temporal monitoring of reef ecosystems. However, it doesn't perform any densification and optimization strategy on the initial sparse point cloud, and the learning-based SfM is prone to reconstruction failures.

The initial effort on developing new solutions for semantic reconstruction relied on classical 3D representations such as point clouds or dense surface models (meshes), obtained from SfM or MVS approaches (see Section 2.1). The advent of neural rendering approaches such as NeRF[26] and 3DGS[3] open new possibilities and semantic representations [48][4].

In the research domain of explicitly labeled models, ObjectGS[5] extends a previous work on 3DGS [32], integrating a semantic variable to the Gaussian parameters. The Gaussian map is initialized using a labeled point cloud, where each Gaussian include an additional semantic class information. The semantic Gaussian cloud can be then learned with a similar approach to 3DGS [3][32], here using an additional semantic loss for a semantically-constrained guidance.

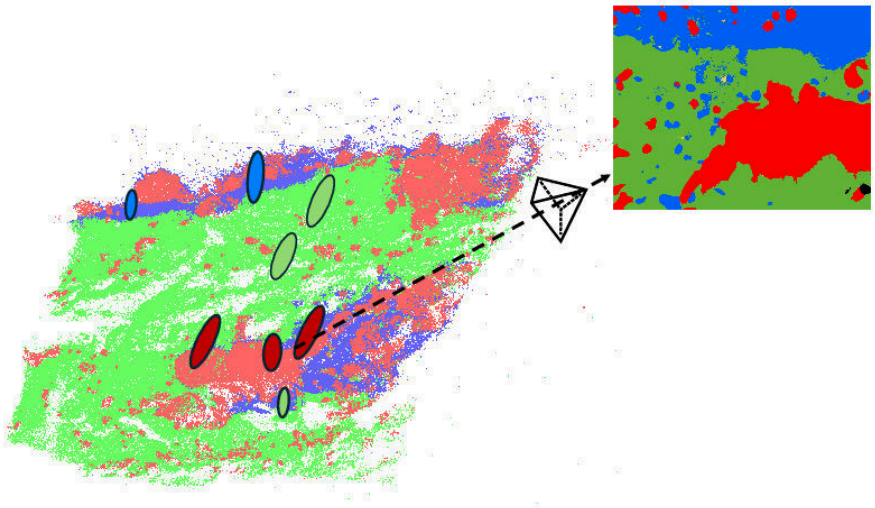


Figure 10: Semantic Splatting for Mask Rendering. Following the process explained by Equations (5) and (6) Gaussians Semantics are alpha blended along camera rays, and the most probable class is associated to each ray (pixel).

This new loss is computed by comparing a predicted semantic mask, rendered from the labeled Gaussian cloud, and the training mask (ground truth),which has been either labeled by hand or from a reliable segmentation model. As illustrated in Figure 10, the semantic mask is rendered with a similar principle of 3DGS α-blending, here projecting semantics instead of view dependent color. Each Gaussian is assigned a one-hot encoded label vector L, whose dimension equals the number of semantic classes in the dataset. In this representation, all elements of the vector are set to zero except for the one corresponding to the predicted class, which is assigned a value of one. Consequently, a Gaussian associated with an object of class i is represented by the label vector:

$$L = [0, 0, \ldots, 1, \ldots, 0]. \quad \text{(with 1 in the } i\text{-th dim)}$$

Then the semantic splatting renderer compute the probability distribution of each class on the corresponding image pixel x, by α-blending over the Gaussians.

$$P(x) = \sum_k \alpha_k \cdot T_k \cdot L_k, \tag{5}$$

Labeling that pixel with the maximum probability class:

$$\mathrm{ID}(x) = \arg\max_i (P_i(x)), \tag{6}$$

Finally, the semantic loss is computed similarly to how semantic segmentation models are trained:

$$\mathcal{L}_{\text{semantic}} = -\sum_x \sum_{i=1}^{n} \mathbb{1}\left(\mathrm{ID}'(x) = i\right) \cdot \log\left(P_i(x)\right), \tag{7}$$

Where $\mathbb{1}$ is the indicator function (1 if the condition is true and 0 otherwise), and $\mathrm{ID}'(x)$ as the ground truth

object ID for pixel x. Other losses are included in the overall training loss, not explained here for simplicity.

2.6 Background Summary

This section explained the algorithms necessary to create a labeled 3D model from an input set of underwater images. SfM is used to initialize a point cloud from the input dataset, and color enhancement can be used to improve the quality of the model and restore colors. 3DGS densifies the 3D reconstruction and improves photorealism. Finally, Semantic Segmentation models extract contextual information from the input images and can be integrated into reconstruction pipelines to embed labels on 3D maps.

Few works on Semantic 3DGS have been implemented or tested in the underwater domain. SWAGSplatting [11] and WaterSplat SLAM[12] exploit semantic masks for reconstruction guidance, without integrating labels on the 3D model. Our framework, GULP3R, is a custom ObjectGS[5] pipeline, chosen for its explicit labeling of the reconstruction model. Since our baseline [5] has been developed for the airborne domain, it is relevant to investigate its usage with underwater images, coupling custom processing and learning models.

3 Methodology

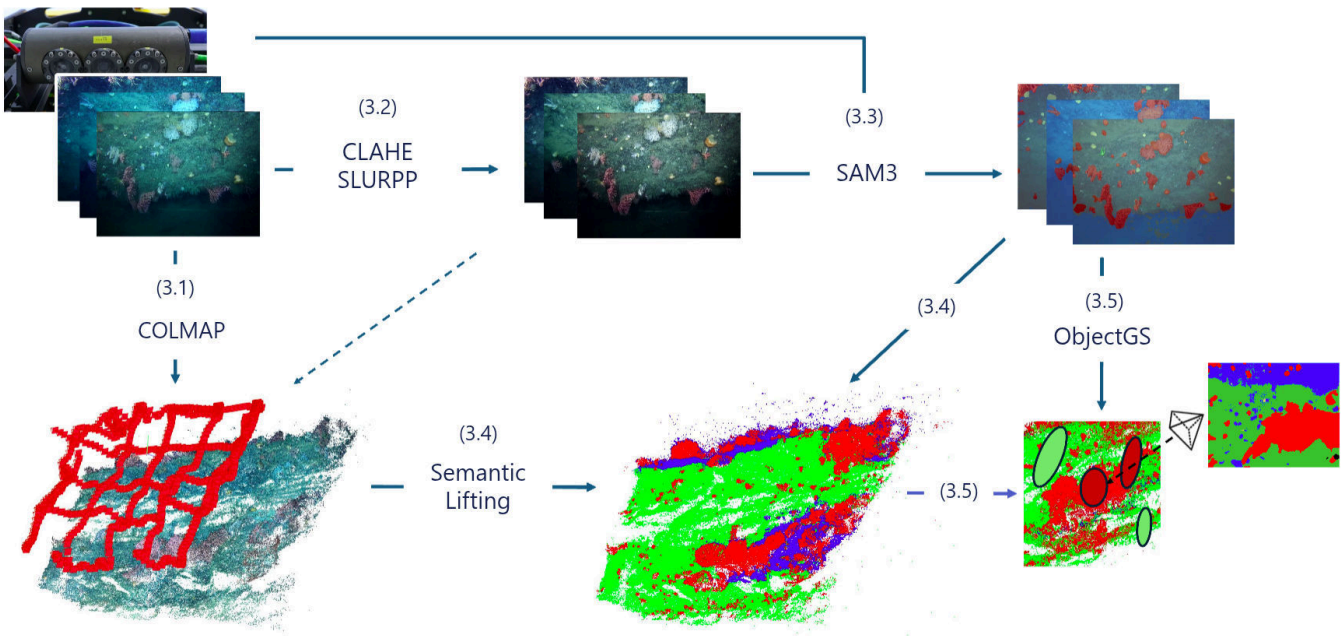


Figure 11: Overview of the GULP3R data flow and processing steps.

The previous chapter describes the processing pipeline we propose with GULP3R (represented in Figure 11). Our method takes as input a collection of photos captured with any imaging system, and performs the following steps:

- (3.1) SfM : Process input images to estimate the scene as point cloud and cameras.
- (3.2) Color Enhance : If necessary, enhance color and re-run SfM.
- (3.3) Semantic Segment: Extract segmentation masks.
- (3.4) Semantic Lift: Project semantics onto the sparse point cloud.

- (3.5) Semantic Splat: Derive a semantic Gaussian cloud using the data computed before.

Next, additional details about each step are given.

3.1 Structure from Motion

Given a collection of N_I photos captured with any monocular (single camera) imaging system, $\mathcal{I} = \{I_i\}_{i=1,...,N_I}$, the first step of GULP3R is the processing of the captured images with any SfM software (e.g. COLMAP[2], Agisoft Metashape[49]). Here we have used the open source SfM software COLMAP[2], configured with SIFT [16] feature extractor and an exhaustive search for correspondences for feature matching, which searches for corresponding descriptors across all pairs of images rather than between sequential frames. Finally, the sparse reconstruction runs and computes the camera poses $\mathcal{T} = \{T_i\}_{i=1,...,N_I}$, the camera matrices $\mathcal{K} = \{K_i\}_{i=1,...,N_I}$, and the sparse point cloud $\mathcal{P} = \{P_j\}_{j=1,...,N_P}$. Note that, if the same camera configuration is used for the whole dataset collection process (no zoom, image resize, or different cameras), a single camera matrix K can be defined. See Section 2.1 and Figures 1 and 2 for a high level overview of the working principle.

Each point in the point cloud $P_j \in \mathbb{R}^3$ is described by 3 coordinates x, y, z and additionally one color with three parameters RGB encoding (r, g, b), representing overall the scene geometry and appearance sparsely, with the colored point cloud $\mathcal{P}$.

3.2 Enhance Underwater Images

The colors and image quality are affected by the underwater medium, and sometimes this degradation can compromise the SfM optimization process or the semantic masks extraction. If the dataset is significantly affected by the medium or the SfM output gets inconsistent results, it is good practice to correct the images with color enhancement techniques and re-run COLMAP[2]. By using a color enhancement module, a dataset of N_I input images $\mathcal{I}$ degraded by the water medium are processed and color corrected as a new outout dataset $\mathcal{I}^* = \{I_i^*\}_{i=1,...,N_I}$.

With GULP3R, we tested both a classical CLAHE based approach[37][36] and the deep learning model SLURPP[38]. While the CLAHE-pipeline applies explicit hand-tuned mathematical operations and image processing techniques in pixel space, SLURPP processes the image through a neural network trained to infer and correct the underwater image degradation.

Figure 7 represents the results of these two techniques applied to a deep-sea reef image.

SLURPP[38] is a ready-to-use transformers model, with network weights provided by the authors. Using this deep learning technique requires some minor processing to rescale the images from the original 512x512 network output.

The CLAHE-based pipeline[36] used here is inspired by an OpenCV blog post on how to correct underwater images[37]. The steps are based on assumptions and knowledge on how the water medium affects the imaging

device:

1. White Balance Correction: Restore the gray-world assumption [50] (a natural image has neutral colors).
2. Red Channel Restoration: Redistribute the red channels' dynamic (red is the most absorbed).
3. CLAHE: Apply local equalization of color with contrast clipping to avoid noise amplification.
4. Dehazing via Dark Channel Prior: Recover scene radiance from bright pixels.
5. Adaptive Unsharp Masking: Enhance edges by subtracting a blurred version of the image.
6. Gamma Correction: Brighten shadows and improve visibility.

The modularity of GULP3R allows to use either SLURPP or CLAHE-based enhancement, or any other technique that works accordingly to the desired performances on the processed dataset.

3.3 Create Semantic Masks

Given the original images $\mathcal{I}$ or the color-enhanced ones $\mathcal{I}^*$, the semantic segmentation module outputs N_I segmented masks $\mathcal{M} = \{M_i\}_{i=1,\dots,N_I}$. See Figure 8c for an example segmentation mask.

To retrieve $\mathcal{M}$, we use a large-scale foundation model, the Segment Anything Model 3 (SAM3) from Meta[10]. As anticipated in Section 2.4, we use SAM3 to generate pseudo-labeled masks, used both to project the semantics in the sparse point cloud (Section 3.4) and to train the semantic splatting model (Section 3.5). This model has been chosen for its open-source nature, its simple usage, and the lack of underwater segmented labeled data for our context. Additionally, SAM3 has been trained using contents from the FathomNet [43] marine database. An example labeled Semantic Mask created with SAM3 can be seen in Figure 12, while additional segmentations are shown in Appendix A.1.

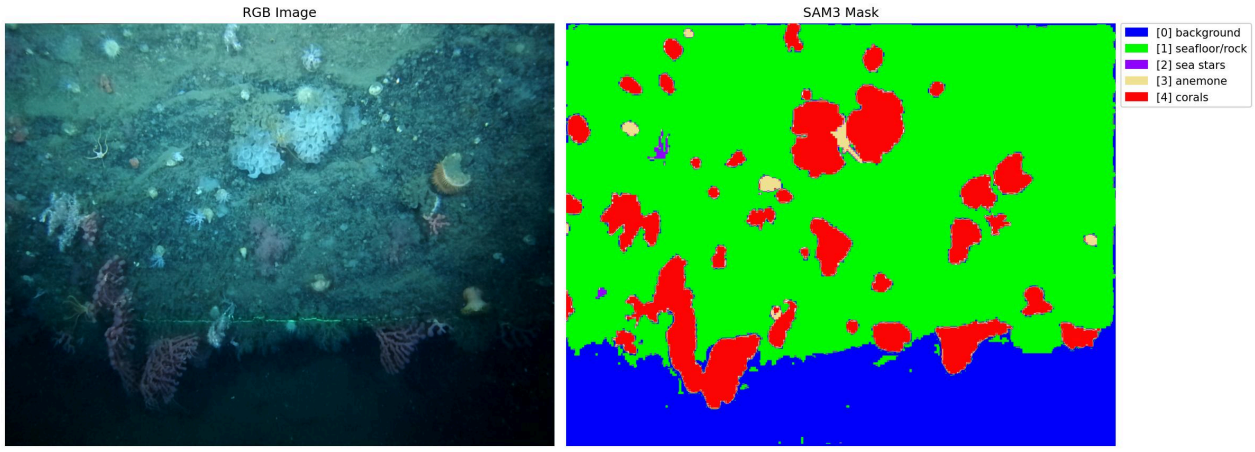


Figure 12: Example RGB image of a deep-sea reef (left) and corresponding semantic segmentation mask estimated with SAM3 pseudo-labeling (right). The mask labels are color encoded following the legend on the top right (blue=background, green=seafloor, purple=sea star, yellow=anemone, red=corals).

SAM3 enables open-vocabulary semantic segmentation[51], allowing objects to be identified from textual descriptions rather than from a fixed set of predefined classes. Unlike closed-vocabulary models[40], whose recognizable classes are determined during training, open-vocabulary models can generalize to previously

unseen categories. In this setting, segmentation is guided by text prompts that specify the classes of interest. These prompts may contain class names or textual descriptions, together with a confidence threshold used to determine whether a prediction is considered a valid detection. Examples of class metadata are reported in Appendix A.2. SAM3 by default works as an instance segmentation model, meaning that even each object of the same class will be separated into a different label. However, the main objective of GULP3R is segmenting by class type, which is why different instances of the same class are grouped with the same label.

GULP3R is modular, meaning that the semantic masks can be created with any model with the required performances. The only requirement is having a labeled semantic mask for every image in the dataset. Additionally, color augmented images (Section 3.2) can be used to create the masks if improved performance can be achieved. This depends on the generalizability of the segmentation model, which may fail when the color balance or object contour adds artifacts to the objects, not learned during training.

3.4 Project Semantic in the Geometric Point Cloud

To initialize the 3D sparse model for an explicit segmentation splatting algorithm, a semantic point cloud is required. The SfM output sparse reconstruction $\mathcal{P}$ should be enhanced to: $\mathcal{P}_L = \{P_{L,j}\}_{j=1,\dots,N_P}$, with $P_{L,j} \in \mathbb{R}^4$ where each point includes both the x, y, z coordinates and the point label l. The projecting strategies used for our pipeline is the same proposed for ObejectGS[5] initialization, and have been described in Section 2.5.

Since the lifted masks are affected by the segmentation model, a strategy to handle multi-view inconsistency (same 3D point labeled differently from different views) has to be defined. The same 3D point $P_j \in \mathcal{P}$ can, in fact, have inconsistent labels projected from different views. The first step is counting the labels' occurrences on the same point, collected in a voting report as a vector storing the number of points labeling occurring for each class. For example, if there are 4 classes identified by numerical labels $\{0, 1, 2, 3\}$, and the same 3D point is seen 8 times from different 2D views and classified twice as 0, twice as 1 and four times as 2, that point have a voting report $\mathcal{L} = \{2, 2, 4, 0\}$. In this situation, the simpler approach is majority voting, which takes the label with the most votes. Therefore, $P_{L,j}$ is extended with a label $l_j = 3$ or one-hot encoded $l_j = [0, 0, 1, 0]$. Other voting strategies such as probability-based and correspondence-based[5] are discussed in Appendix B. In our pipeline, we mainly tested majority-based, because it is faster and satisfactory enough for initializing the semantic sparse model, but correspondence-based often provides more accurate results at the cost of higher computation time.

3.5 Semantic splatting

Given the set of images $\mathcal{I}$ (or the enhanced ones $\mathcal{I}^*$), the semantic masks $\mathcal{M}$, and the semantic point cloud $\mathcal{P}_L$, ObjectGS [5] is used to create a semantic Gaussian cloud. As described in 2.5, a custom semantic renderer and a semantic loss are used to guide the learning process.

The final result of the GULP3R pipeline is a semantic Gaussian cloud $\mathcal{G}_\mathcal{L} = \{G_{L,j}\}_{j=1,\dots,N_G}$, used to synthesize 2D Semantic Masks as explained in Section 2.5. $\mathcal{G}_\mathcal{L}$ can be interpreted as a semantic point cloud $\mathcal{P}_\mathcal{L}$ of

N_G points, each one $P_{L,j}$ with coordinates equal to the corresponding Gaussian $G_{L,j}$ 3D mean $\mu_{L,j}$. Unlike the basic implementation of 3DGS[3], ObjectGS[5] encodes Gaussians' color, opacity, and covariance in neural networks, using a hybrid explicit-implicit representation. The information in the Gaussian cloud and appearance networks allow to render colored and semantic views on the scene. Additionally, if the training is successful, the sematic Gaussian cloud $\mathcal{G}_{\mathcal{L}}$ is a denser and more optimized semantic 3D model with respect to $\mathcal{P}_L$.

4 Results

GULP3R has been evaluated on field data from different sites, collected with multiple robotics platforms and imaging systems. In this section, the datasets will be described, then the evaluation metrics introduced, and finally the results of our pipeline are presented and compared with other state of the art techniques.

4.1 Datasets

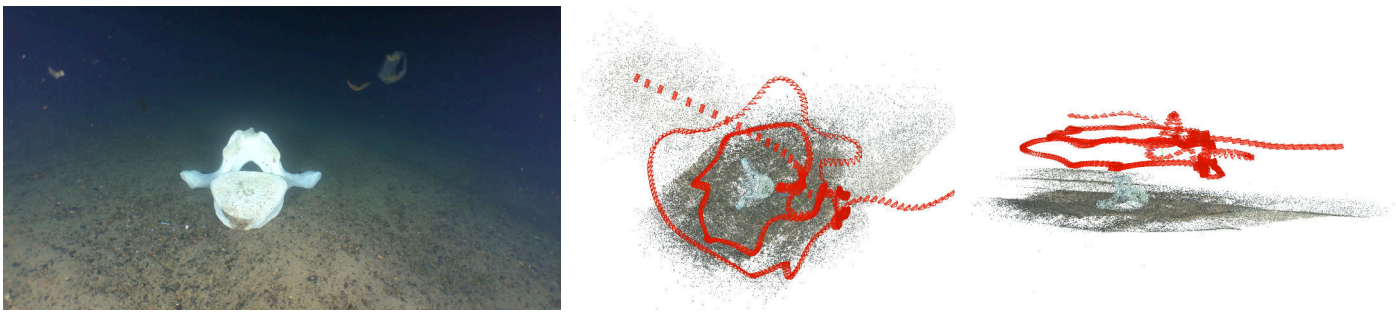


(a) MOLA AUV. (b) MiniROV. (c) ROV Ventana.

Figure 13: MBARI's underwater robots used to collect the datasets.

For the evaluation of our pipeline, we have used three datasets collected by the CoMPAS Lab (MBARI), with different levels of semantic and geometric complexity. The **bone** dataset includes a whale bone on the seafloor, **corals** covers a coral-sponge community on a vertical rock, and **anemone** has images of some white anemone standing on top of a rock. Further details are described below.

bone: Arctic Whale Bone - MOLA AUV

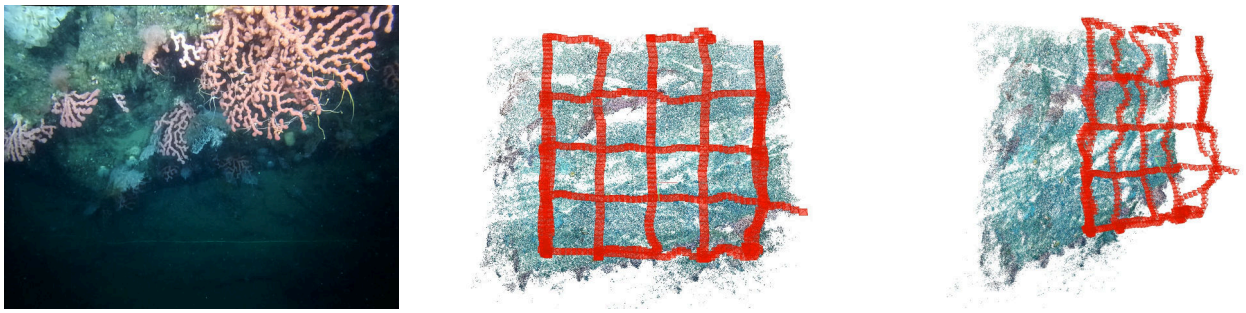


(a) Dataset frame. (b) Trajectory top view. (c) Trajectory side view.

Figure 14: Example RGB image (a) and trajectory (b,c) of the imaging system for the bone dataset. (b) and (c) are two perspective views of COLMAP[2] output, the red shapes represent the estimated camera's poses (trajectory) and the points are the reconstructed 3D scene point cloud.

This dataset was collected in the Arctic Ocean, using the portable MOLA AUV (see Figure 13a). A total of 505 images have been collected, with a resolution of 2258x1116 pixels. An image from the dataset and the trajectory of the imaging system are represented in Figure 14. The 4K forward looking camera mounted on the system has been used to collect the images, and proper lighting is guaranteed by MOLA's strobes. The scene includes a whale bone in an almost flat sandy seafloor, creating an evident contour of the object. Additionally, some jelatinous animals act as dynamic distractors. Overall, this dataset allows testing of GULP3R on two instances (seafloor, bone) with recognizable semantics and simple geometry.

corals: Sur Ridge Corals - MiniROV



(a) Dataset frame. (b) Trajectory front view. (c) Trajectory side view.

Figure 15: Example RGB image (a) and trajectory (b,c) of the imaging system for the corals dataset. (b) and (c) are two perspective views of COLMAP[2] output, the red shapes represent the estimated camera's poses (trajectory) and the points are the reconstructed 3D scene point cloud.

This dataset was collected in Sur Ridge, in the Monterey Bay, using the MiniROV robotic system (see Figure 13b), during the Deep Sea 3D expedition. A total of 729 images have been collected, with a resolution of 1946x1496 pixels. An image from the dataset and the trajectory of the imaging system are represented in Figure 15. The scene include a colony of bubblegum corals and sponges, populated by different species of sea stars in a vertical rock wall. Overall, this dataset allows testing of GULP3R on a complex semantic scenario with structured geometry. The following semantic classes have been used: coral, anemone, sponge, sea star, rock surface.

anemone: Portuguese Ledge Anemone - ROV Ventana



(a) Dataset frame. (b) Trajectory top view. (c) Trajectory side view.

Figure 16: Example RGB image (a) and trajectory (b,c) of the imaging system for the anemone dataset. (b) and (c) are two perspective views of COLMAP[2] output, the red shapes represent the estimated camera's poses (trajectory) and the points are the reconstructed 3D scene point cloud.

This dataset was collected off of Portuguese Ledge, a State Marine Conservation Area in the Monterey Bay, using the ROV Ventana(see Figure 13c). A total of 750 images have been collected, with a resolution of 3840x2160 pixels. An image from the dataset and the trajectory of the imaging system are represented in Figure 16. The images have been extracted from the forward-looking 4K MxD SeaCam. We collected those images during the CoMPAS Lab expedition on the R/V Rachel Carson on the 08/05/2026. The vertical rock with a group of white-plumed anemone (*Metridium farcimen*) was selected as a good case study for GULP3R, containing multiple instances of the same class in a structured space. This scene is a valuable test for the proposed pipeline, as being less challenging than the semantic complexity of Sur Ridge, but more insightful than the single bone instance in the flat Arctic seafloor.

4.2 Metrics and Training

A complete evaluation of an algorithm for semantic 3D reconstruction through Gaussian splatting should consider the following metrics:

- 3D Geometry: accuracy of the point cloud with respect to the real 3D structure.
- 3D Semantics: accuracy of the semantic cloud in labeling each point.
- 2D Synthesis: photometric performance of novel view synthesis via splatting.
- 2D Semantics: semantic segmentation performance via semantic splatting.

The lack of a ground truth (or a high-quality) 3D geometric and semantic model makes the evaluation of the desired 3D metrics challenging. Moreover, as described in Section 3.3, the used 2D semantic masks are prone to errors. Therefore, they cannot be used as a valid ground truth to evaluate the 2D semantic splatting performance.

For the reasons above, we quantitative evaluate our method using the standard photometric indices (2D Synthesis): PSNR, SSIM [52], and LPIPS [53]. Additionally, qualitative comparison of 3D Geometry, 3D Semantics, and 2D semantics is also discussed. The 2D synthesis indices used describe the photometric distance between the image rendered with the splatting process and the corresponding original image. To evaluate how the model performs in generalizing novel-view synthesis, the original dataset is split in a 88/12 split train/test dataset. The train dataset is used to learn the model, while the test one evaluates the generalizability in unseen scenarios. Therefore, the results reported in Table 2 are computed at the end of the training by using the camera poses and corresponding images of the test dataset. Further details on the training configuration and the train/test split is described below, a brief explanation of the qualitative and quantitative metrics.

Qualitative Evaluation: 3D Geometry and Semantics, 2D Semantics

In the absence of ground truth for the 3D geometric and semantic models, as well as the 2D semantic masks, it is necessary to define clear non numeric strategies to evaluate the results. To evaluate 3D Geometric data visually, we refer to ”artifacts” and ”floaters”, which are corrupted 3D points in the reconstructed model due to erroneous estimation processes. The first refers to estimated 3D point far from the original 3D scene surfaces.

The latter are 3D points with coordinates in empty space on the original scene. Both are created by errors in the reconstruction process, introducing inexact geometric knowledge in the final 3D model. Additionally, density of the point or Gaussian cloud is used as first approximation of the degree of details and resolution that the reconstruction has captured about the original 3D scene. Similarly, to evaluate the 3D Semantics, we refer again to ”(semantic) artifacts/floater”, corresponding to erroneous semantic labeling of the 3D points. For example, A 3D point occupying the space of a coral is labeled as background. This is evaluated by manual observation of the semantic point or Gaussian cloud, exploiting human understanding of the semantic structure of the scene. For example, a 3D point occupying the space of a coral but classified as seafloor can be qualitatively identified as a semantic artifact. To evaluate 2D semantics, GULP3R's semantic rendering is compared with respect to the pseudo-labeled masks of SAM3[10], using manual observations and contextual understanding of the original image as ground truth.

Although these qualitative strategies to evaluate and compare the tested algorithms are not rigorous, they can provide a first approximation of the quality of our results in the absence of ground truths.

Quantitative Evaluation: 2D Synthesis

PSNR↑: Peak Signal-to-Noise Ratio

The PSNR (in decibels, dB) metric captures the difference between two images' pixel intensities, in terms of reconstruction quality and noise level. Therefore, it is not representative of high-level visual perception or structural coherence between the two images. This index is also used to compare image compression quality. A higher PSNR means more accurate rendering quality with respect to the ground truth image. In general, satisfactory values are between 25 and 35 dB [3][7], but it depends on the complexity of the dataset.

SSIM↑: Structural Similarity Index Measure

SSIM (adimensional) focuses on local image structure, characterizing differences in edges, texture, and structural integrity. This metric identifies three aspects of visual perception of structural information: luminance, contrast, and structure. Higher SSIM indicates a higher level of similarity between the two images. SSIM is bounded between 0 and 1 and in general, typical satisfactory values are higher than 0.8. However, as for the PSNR, the high-level semantics and the visual realism between the two images is not captured by the SSIM. Two images may have differences in the structure but appear very similar for the human eye.

LPIPS↓: Learned Perceptual Image Patch Similarity

To quantify the difference between two images as if a human eye is evaluating it, LPIPS (adimensional) proposes a perceptual similarity metric[53]. This index measure images distance in VGGNet[54] feature space, where VGGNet is a deep CNN able to extract high-level visual features. Lower LPIPS means a smaller distance between the two images and higher perceptual similarity. In general, satisfactory values are between 0.3 and 0.1, but it depends on the complexity of the dataset.

Training Details

As explained in Section 3, the last step of GULP3R consists of processing the (optionally enhanced) images, the semantic masks, and the semantic point cloud with ObjectGS[5]. For our evaluation, we used the same training configuration proposed by the authors in [5], training for 30k iterations with a train/test split of 88/12. Table 1 summarizes the information about our datasets. Computation time reported includes only ObjectGS training, as we did not keep track of the computation time of the preliminary steps (enhancement, SfM, segmentation, and lifting).

Table 1: Summary of the MBARI field datasets and the corresponding Semantic Gaussian Splatting training settings. For each dataset, the table reports the number of images, image resolution, training and test set sizes, and the training time required by ObjectGS [5].

Dataset	N. Images	Resolution [pixels]	N. Training Images	N. Test Images	Training time [minutes]
bone	505	2258x1116	441	64	48
corals	729	1946x1496	637	92	58
anemone	750	3840x2160	656	94	31

4.3 Comparison

GULP3R 2D Synthesis has been quantitatively compared to 3DGS [3] with the nerfstudio splatfacto implementation[55][56] and SeaSplat [7]. 3D Geometric qualitative results have also been compared with 3DGS and SeaSplat, while 3D Semantic is compared to the initial sparse semantic point cloud derived with the mask lifting approach from Section 2.5 and Section 3.4. Additionally, 2D Semantic is evaluated based on the semantic masks generated by ObjectGS[5] by splatting the labeled Gaussians.

2D Synthesis Quantitative Comparison

Table 2: 2D Rendering comparison on MBARI's Datasets. For each dataset, the 2D novel view synthesis photometric results are reported with respect to the compared models: 3DGS, SeaSplat, GULP3R. Bolded numbers highlight to the best results for each dataset with respect to each metric.

Dataset	Model	PSNR↑	SSIM↑	LPIPS↓
bone	3DGS	**32.12**	**0.9241**	**0.1953**
	SeaSplat	31.69	0.9127	0.2744
	GULP3R(ours)	20.85	0.8753	0.2994
corals	3DGS	27.55	0.8766	**0.2483**
	SeaSplat	27.51	0.8751	0.3021
	GULP3R(ours)	**28.357**	**0.9098**	0.2785
anemone	3DGS	**35.60**	0.9401	0.3306
	SeaSplat	32.25	0.9315	0.3666
	GULP3R(ours)	30.36	**0.9431**	**0.3186**

Table 2 reports the photometric indices on our datasets, with best results bolded. Additionally, example GULP3R and SeaSplat[7] renderings on the test images are reported in Figures 17 to 19. From our experiments, color enhancing the datasets did not improve quantitatively the 3D performances; therefore, here we discuss the results using the collected images, either raw or with minimal post-processing. GULP3R is outperformed by 3DGS on the bone dataset, while it provides competitive results on corals and anemone. SeaSplat instead outperforms our pipeline on the bone dataset, but it is overall less photometrically accurate than GULP3R or 3DGS.

The photometrically more satisfactory performances of 3DGS on the bone dataset may be explained by the presence of many images that are highly occupied by ocean background. In fact, GULP3R semantic guidance ignore densification and learning on the background, creating significant artifacts affecting photometric performance. Additionally, it is worth noting that the Gaussian splatting algorithm [5] used with our method relies on semantic masks to guide the scene learning. This aspect is visible from the improved results on corals, which is a more semantically rich dataset. Another relevant result is how semantic guidance improves the removal of dynamic distractors like fishes and sinking particles from synthesized images.

The results of photometric performance don't show advantages in using GULP3R; however, the main focus of our work is the semantic reconstruction rather than the novel view synthesis. The next section will describe the real potential of GULP3R in semantic reconstruction (Section 4.3). To anticipate the 3D geometric findings, Figure 20 represents the Gaussians learned by 3DGS and SeaSplat. 3DGS does not implement any appropriate regularization to limit densification, and the underwater medium create several artifacts that are visible from the Gaussians' distribution in space. SeaSplat integrates the medium modeling and additional regularization terms in the loss, constraining the propagation of the Gaussians in empty space. GULP3R, on the other hand, exploit semantics information to guide the Gaussians densification, adding an additional degree of constraint which is fundamental in hazing mediums like water (this will be seen next in Figure 21).

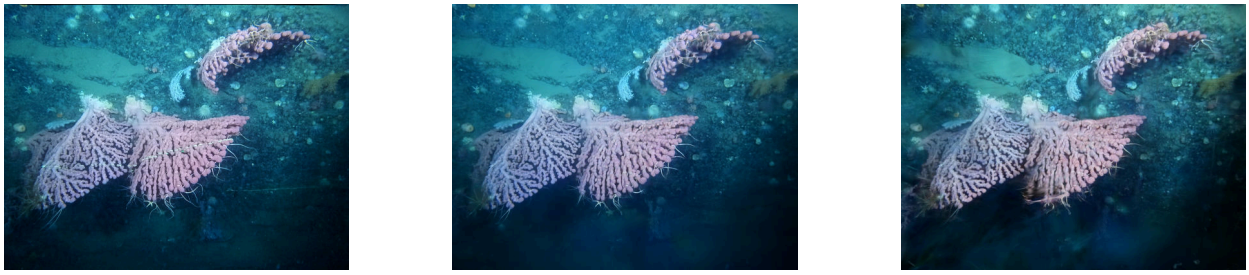


Ground Truth C1. GULP3R(ours) C1. (a) SeaSplat C1.



Ground Truth C2. GULP3R(ours) C2. SeaSplat C2.

Figure 17: Comparison between Ground Truth, GULP3R(ours), and SeaSplat rendering on two cameras (C1, C2) of the bone dataset.



Ground Truth C1. GULP3R(ours) C1. SeaSplat C1.



Ground Truth C2. GULP3R(ours) C2. SeaSplat C2.

Figure 18: Comparison between Ground Truth, GULP3R(ours), and SeaSplat rendering on two cameras (C1, C2) of the corals dataset.

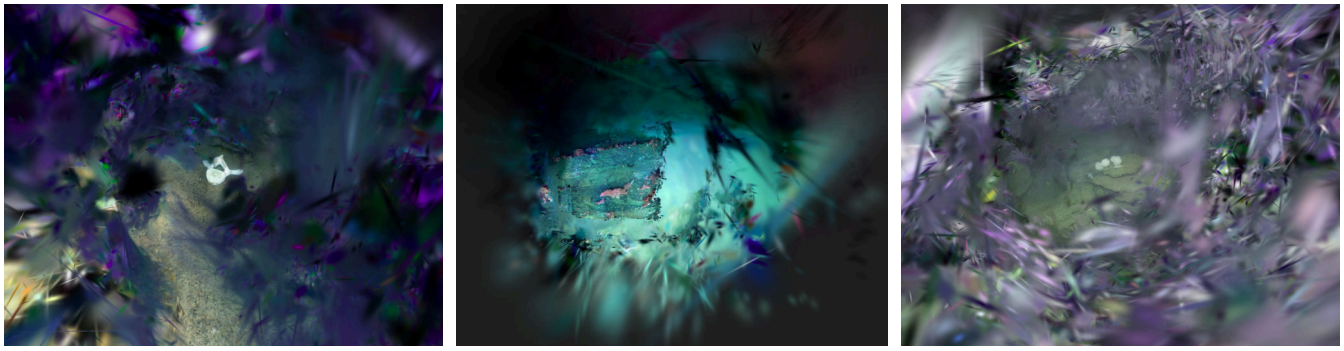


Ground Truth C1. GULP3R(ours) C1. SeaSplat C1.

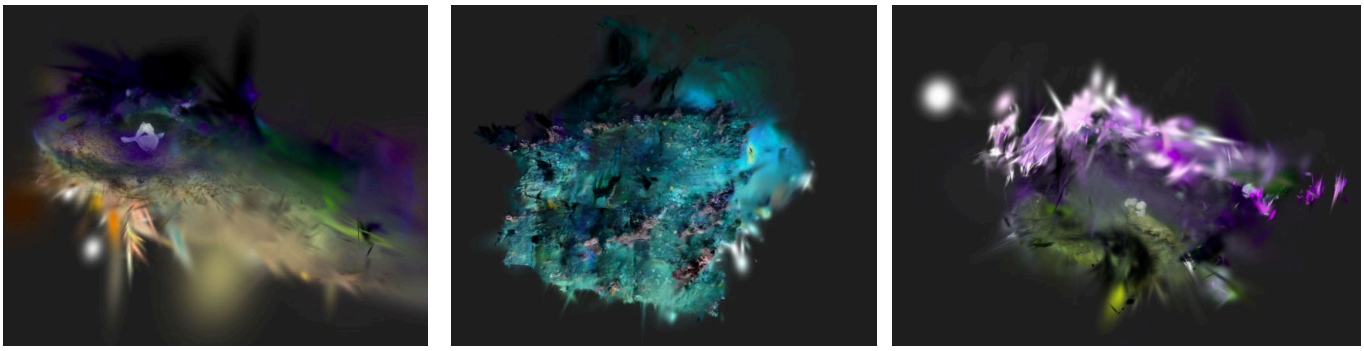


Ground Truth C2. GULP3R(ours) C2. SeaSplat C2.

Figure 19: Comparison between Ground Truth, GULP3R(ours) and SeaSplat rendering on two cameras (C1, C2) of the anemone dataset.



(a) 3DGS bone Gaussians. (b) 3DGS corals Gaussians. (c) 3DGS anemone Gaussians.



(d) Seasplat bone Gaussians. (e) Seasplat corals Gaussians. (f) Seasplat anemone Gaussians.

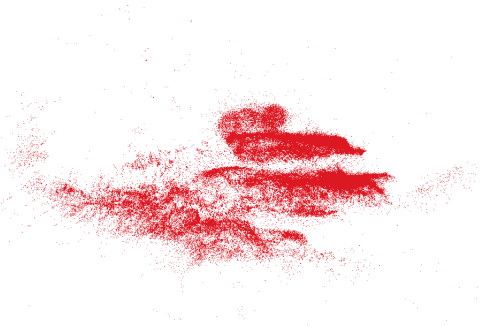
Figure 20: Comparison between 3DGS (first row: a,b,c) and SeaSplat (second row:d,e,f) Gaussian cloud for the bone (a,d), corals (b,e) and anemone (c,f) datasets.

3D Geometry and Semantics Qualitative Comparison

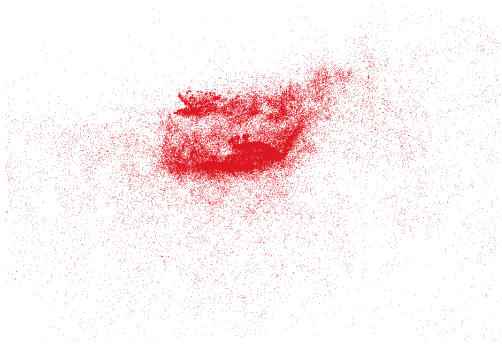
Table 3 reports the number of points in the initial sparse point cloud model (N_P) obtained with COLMAP SfM [2], and the number of Gaussians (N_G) after running the related Gaussian splatting method. Additionally, Figure 21 reports the sparse 3D map for the anemone dataset (in the case of Gaussian clouds, the Gaussians' mean is used as the point cloud). The sparse point clouds for bone and corals are reported in the supplementary Figures 32 and 33. Overall, from the results reported in Table 3 and Figures 21, 32 and 33 it can be seen how 3DGS and SeaSplat reconstructions are rich in artifacts and floaters, while GULP3R constrains the Gaussians to be located in space truly occupied by objects' surfaces and volume in the reconstructed physical 3D scene. These more satisfactory results in densification may be related to the backbone of ObjectGS[5][32] or to the semantic guidance that ignores pixels classified as background in the semantic masks.

Table 3: Number of points in the 3D models. for each dataset it is reported the number of points N_P in the sparse point cloud (computed with COLMAP[2]), and the number of Gaussians N_G in the Gaussian Clouds (computed with the compared methods: 3DGS, SeaSplat, GULP3R.).

Dataset	N_P (SfM)	Model	N_G (GS)
bone	317676	3DGS	179026
		SeaSplat	332862
		GULP3R(ours)	302608
corals	366166	3DGS	704549
		SeaSplat	2164557
		GULP3R(ours)	407841
anemone	180676	3DGS	100586
		SeaSplat	165150
		GULP3R(ours)	164368



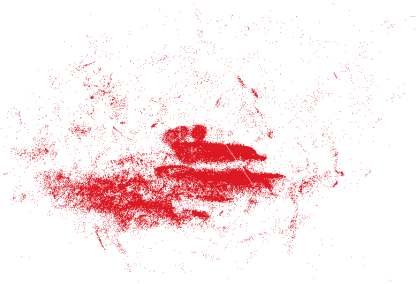
(a) COLMAP.



(b) 3DGS.



(c) SeaSplat.



(d) GULP3R(ours).

Figure 21: Comparison between COLMAP, 3DGS, SeaSplat, GULP3R point clouds on the anemone dataset. Here the RGB points of the reconstructed geometry are reported as red, in order to highlight the spatial coverage and geometric distribution of the compared reconstruction techniques. The red points are either the coordinates P_j of the SfM point cloud output $\mathcal{P}$ (a) or the centers (mean μ_j) of a gaussian G_j in the gaussian cloud $\mathcal{G}$ (b,c,d).

The semantic reconstruction in the anemone dataset is reported in Figure 22, comparing mask lifting (intermediate step of GULP3R Section 3.4) and semantic splatting (GULP3R outputSection 3.5). Semantic reconstruction results for bone and corals are reported in the supplementary Figures 34 and 35. Optimizing the lifted semantic cloud with GULP3R redistributes the semantic labels, removing most of the point cloud and densifying the model in the region of interest. Therefore, it can be assumed that the semantic guidance

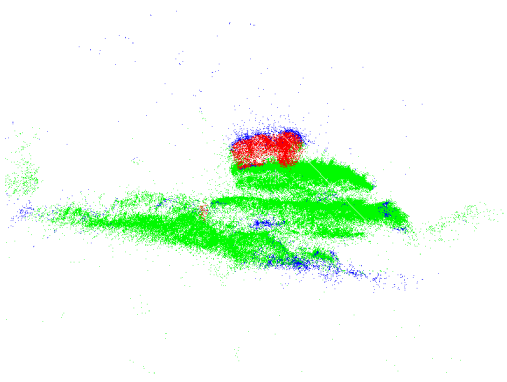
implicitly addresses the multi-view inconsistency problem to some extent. However, some additional floaters are created, which can compromise the rendered semantic mask, as will be discussed next.



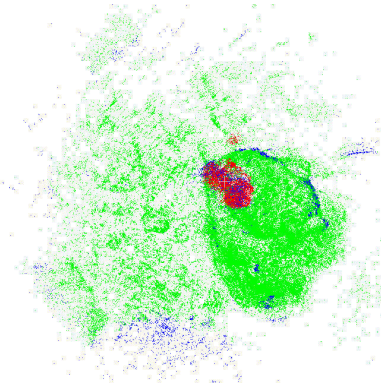
(a) Colored point cloud V1.



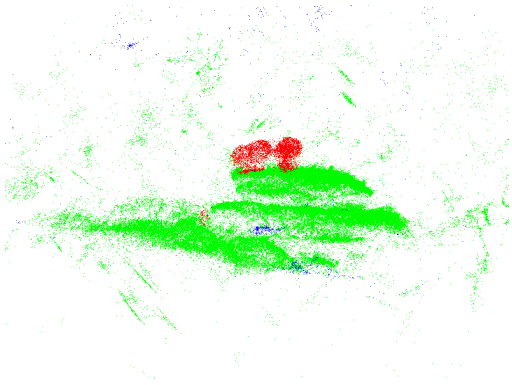
Colored point cloud V2.



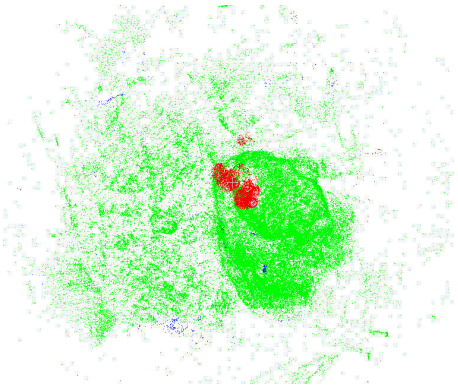
(b) Lifted point cloud (majority voting, Algorithm 1) V1.



(c) Lifted point cloud (majority voting, Algorithm 1) V2.



(d) GULP3R semantic Gaussian cloud V1.



(e) GULP3R semantic Gaussian cloud V2.

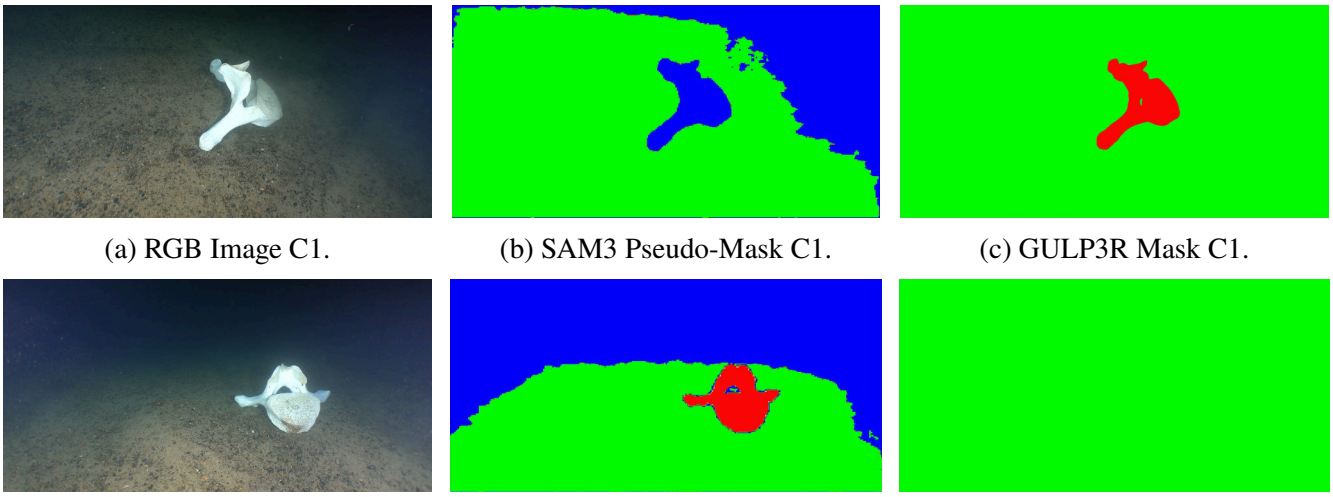
Figure 22: Two perspective views (V1, V2) on the SfM RGB point cloud (a,b), lifted semantic(c,d), and GULP3R semantic Gaussian cloud (e,f) for the anemone dataset. Point labels are color encoded as: blue=background, green=seafloor, red=anemone.

2D Semantic Synthesis Qualitative Comparison

Semantic Splatting from a labeled Gaussian cloud has been explained in Section 2.5 with Equations (5) and (6). In simple terms, instead of colors, semantics are splatted, and the pixel label is chosen based on the maximum α-blending based class probability.

Figures 23 to 25 report some example evaluation semantic rendering with respect to the training mask (notice

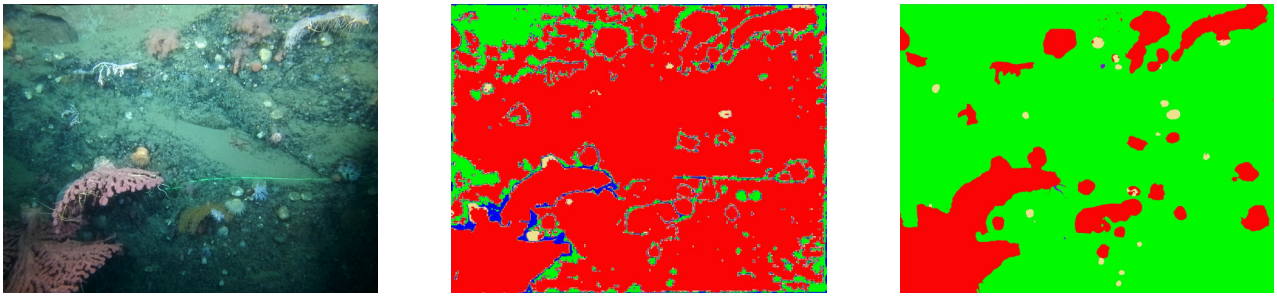
that the training masks are generated with SAM3, so these are not the ground truths but pseudo-labeled masks[44]. In case of successful semantic reconstruction, we expect that it is feasible to improve the training semantic masks with the synthesized masks). However, the results are not always consistent; in some cases some pixels labeled wrongly by SAM3 are improved with GULP3R, while in other cases the synthesized masks are far from the expected results. We hypothesize that this behavior is related to the floater semantic artifacts that get projected and assume a higher weight on the pixel label. In fact, rendered masks closer to the objects are more accurate, corresponding to situations in which less Gaussians in empty space are encountered by the rendering rays. Additionally, GULP3R learns the scene without background labels, this is why no pixel appears blue in the rendered masks.



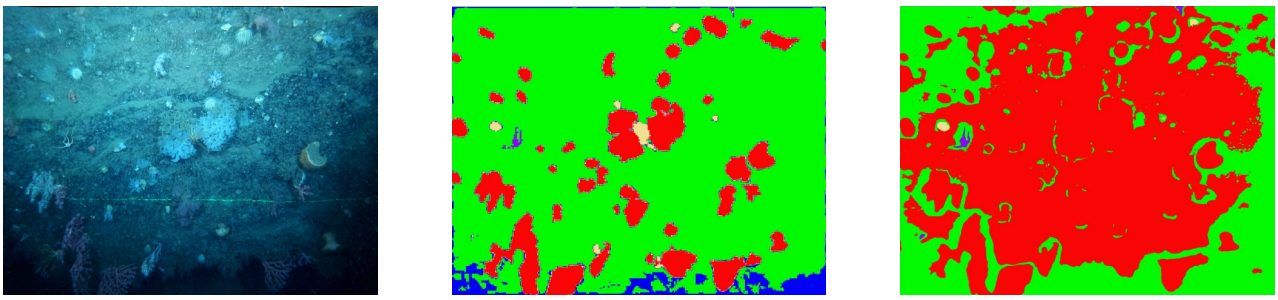
(a) RGB Image C1. (b) SAM3 Pseudo-Mask C1. (c) GULP3R Mask C1.

(d) RGB Image C2. (e) SAM3 Pseudo-Mask C2. (f) GULP3R Mask C2.

Figure 23: Comparison between SAM3 (pseudo-labeled) (b,e) and GULP3R (c,f) semantic mask on two cameras C1,C2 of the bone dataset. Mask pixels are color encoded as: blue=background, green=seafloor, red=bone.

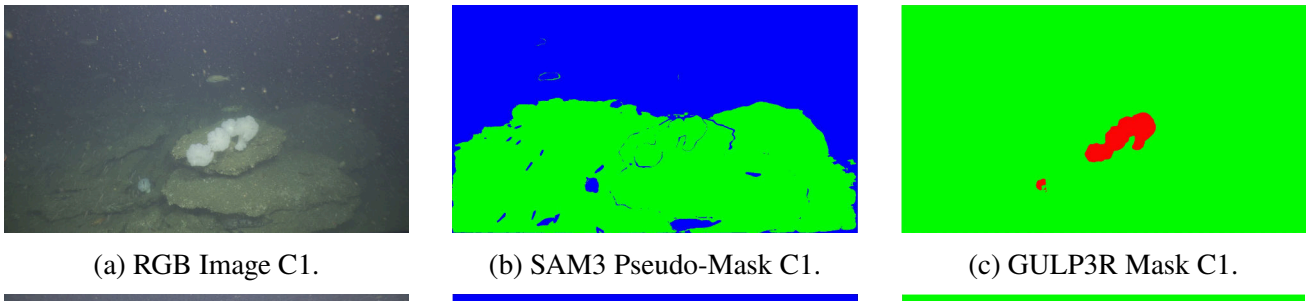


(a) RGB Image C1. (b) SAM3 Pseudo-Mas C1. (c) GULP3R Mask C1.



(d) RGB Image C2. (e) SAM3 Pseudo-Mask C2. (f) GULP3R Mask C2.

Figure 24: Comparison between SAM3 (pseudo-labeled) (b,e) and GULP3R (c,f) semantic mask on two cameras C1,C2 of the corals dataset. Mask pixels are color encoded as: blue=background, green=seafloor, yellow=anemone, purple=sea star, red=coral.



(a) RGB Image C1. (b) SAM3 Pseudo-Mask C1. (c) GULP3R Mask C1.



(d) RGB Image C2. (e) SAM3 Pseudo-Mask C2. (f) GULP3R Mask C2.

Figure 25: Comparison between SAM3 (pseudo-labeled) (b,e) and GULP3R (c,f) semantic mask on two cameras C1,C2 of the anemone dataset. Mask pixels are color encoded as: blue=background, green=seafloor, red=anemone.

5 Discussion

This section provides some additional comments on the results, implementation details, and research findings of our work.

5.1 Implementation

The GULP3R processing pipeline has been implemented using open-source projects, with the objective of being reproducible by any researcher working on similar problems.

In our current pipeline evaluation, the algorithms used are executed sequentially and manually, step by step, but an automatic computational flow with intermediate verifications and parallelization can be implemented with some software effort. For the reason above, GULP3R is not yet open sourced. Additionally, we are considering possible additional improvements and research before public release. The objective is to realize a modular system, ready to use by anyone without any specific background knowledge, but also easy to modify and improve for the community.

The results have been computed on a machine with 32 GB of RAM and an NVIDIA L40S-48C graphic card (48GB of VRAM). However, analysis of the memory consumption to run the pipeline has not been performed yet, therefore, the system used doesn't reflect the possibly less constraining requirements.

5.2 Future Directions

The findings from this research highlight the limitations of GULP3R and open many possible future directions to develop smarter mapping systems for scientists. Both ideas at the engineering and theoretical levels are discussed next. The main directions of research involve: implementing an automatic pipeline, investigating online extensions, performing more extensive quantitative evaluation, fine-tuning the modules, exploring underwater medium integration, considering instance segmentation, and further investigate semantic splatting methods. These aspects are discussed below.

Automatic Pipeline

As introduced before (Section 5.1), a first necessary requirement is the development of an usable pipeline that doesn't involve manual execution of each step. This will help scientists to use this tool on their own data, and researchers in robotic perception to reproduce the results and propose improvements.

Online Extensions

Another direction involving both engineering and theoretical effort is the investigation on an online technique to generate/update semantic maps. GULP3R, in fact, relies on algorithms that run offline on collected datasets, and can take a few hours to create a semantic map. From novel findings on 3DGS-SLAM [12] and explicit semantic embeddings in 3DGS[3][4], there are many alternative directions to propose and implement novel Semantic-3DGS-SLAM algorithms. Underwater systems can rely on these data processing techniques during deployment, updating a pre-existing semantic map or creating a rough labeled map, which can help to better explore the site. This online map can be less accurate at first, then optimized with more computationally demanding reconstruction techniques after deployment.

Quantitative Evaluation

As reported in Section 4.2, only novel view synthesis' photometric quality has been quantitatively evaluated, even if the main objective of GULP3R is the semantic map reconstruction of a benthic environment. In order to further evaluate GULP3R and improve its semantic mapping, it is necessary to collect a synthetic dataset (on simulations) or in a controlled environment. With a ground truth semantic 3D model and semantic masks, also 3D geometry and 3D/2D semantics will be quantitatively evaluable.

Fine-Tune the Modules

It is evident from Section 2 that many module combinations and tuning are possible on the proposed modular pipeline. In fact, the SfM, color enhancement, semantic segmentation, and semantic splatting depends on the selected model and the hyperparameter tuning (values pre defined before training/optimization). During the testing of GULP3R, few combinations and quantitative evaluations on different color enhancements or semantic segmentation models were done. Therefore, the performance can be improved with finer pipeline design choices. In particular, the effect of color enhancement on the rendering and geometry has not been investigated. If a synthetic/controlled dataset is available (known ground truth), geometric performances and semantic reconstruction can highlight new limitations and research direction.

Underwater Medium Model

In the GULP3R pipeline, to consider the underwater image degradation, the dataset can be pre-processed with color enhancement algorithms. However, an alternative approach could be inherited from SeaSplat[7]. Instead of pre processing the data, the underwater imaging modeling can be integrated in the model itself and learned with the scene semantics, structure, and appearance. Investigating how this differs from our proposed approach can provide novel research findings.

Instance Segmentation

While the baseline ObjectGS[5] method relies on instance segmentation masks, with GULP3R we use semantically segmented masks, grouping different object instances with the same label. However, differentiating each object boundary and area in the images can improve the reconstruction quality by better guiding the Gaussian redistribution for each object rather then for a whole object type class. Additionally, an instance segmented map can be more useful than a semantic one for scientists, by isolating each organism of a colony.

Semantic Gaussian Splatting

Finally, the semantic Gaussian cloud is reconstructed and splatted with the method proposed in ObjectGS[5]; however, the domain of semantic splatting is an active area of research in continuous evolution. Further investigation on alternative semantic splatting algorithms and improving the current state-of-the-art could improve GULP3R. For example, our qualitative evaluation of 2D semantic synthesis highlighted how wrongly labeled and positioned Gaussian floaters compromise the synthesized masks, labeled as uniquely seafloor, as visible in Figures 23f and 25f. Therefore, alternative splatting approaches, semantic embedding, or loss

regularization should be considered next for better guiding the geometric and semantic learning of the 3D scene.

6 Conclusion

In this study, we proposed and tested a pipeline for creating semantic maps of underwater scenes from camera images. GULP3R optionally enhances the underwater images affected by the medium, and then processes the data to obtain semantic masks and a sparse 3D reconstruction. Finally, the semantic information is embedded on the initial sparse model, and a Gaussian splatting based approach is used to densify and improve the semantic reconstruction. We quantitatively and qualitatively evaluated our method on underwater datasets collected during field experiments, highlighting the positive and negative results, as well as future research directions. Our pipeline shows preliminary results about how an estimated labeled 3D model can be improved with semantic Gaussian splatting. Additionally, we demonstrate the feasibility of using foundation models to create semantic masks of marine images, although with limited performance in complex scenes. Our investigations into color enhanced images did not suggest qualitative improvement in the 3D geometric reconstruction. However, conclusions about possible advantages of pre-processing the dataset with color enhancement techniques cannot be derived, since extensive tests and quantitative evaluation have not been performed. Future work should include supplementary investigation on the usage of fine-tuned semantic and reconstruction models, the effect of color enhancement on semantic reconstruction, and the ways to fully integrate semantics in the reconstruction process. Finally, continuous exchange of ideas and feedback from scientists is critical to develop useful tools to solve real-world problems.

Acknowledgments

This project would not have been possible without the help of my amazing mentors, Pushyami and Giancarlo. Since before the internship, they gave me the freedom to propose my own ideas on the project, that end up to be shaped around my personal research interests. They give me the guidance, tools, and cheering that I needed to address such a complex robotics problem. I would like to thank all the members of the CoMPAS Lab for the insightful chats and for making me feel part of the team. Special thanks to Onur, who has been a third mentor for me during this summer. Great spotlight to George and Megan for making this opportunity possible and for everything they did to make it an amazing, life-changing experience. I'm very grateful to the ROV Ventana pilots Ethan and Chris, and all the crew of the Carson expedition on 05/08, using the anemone dataset that I have collected thanks to you has been emotional.

I have teary eyes but a joyful smile in expressing my thanks to all the summer interns that have lowkey (high-key?) made Monterey a special place in my "heart". If I learned something new about phytoplankton and respiration quotient, floats and motors, sinking particles and Lagrangian, "abalone", transects and synchronization problems, underwater photography, and especially about American idioms, ice skating, easy hikes, building brick cities, and enjoying sunsets, it is thanks to you (prish cudis for the shared neuron).

And to you, the reader, thank you for reading this paper. Whether you are a roboticist, an oceanographer, or simply a curious mind, I hope you learned something that inspires you to explore, understand, and protect the ocean.

The MBARI Summer Internship Program is generously supported by the David and Lucile Packard Foundation. Additional funding is provided by Dalio Philanthropies and the Maxwell/Hanrahan Foundation.

The oceans deserve our respect and care, but you have to know something before you can care about it.
— Sylvia A. Earle

References

[1] Monterey Bay Aquarium Research Institute. *MBARI 3D Models*. 2024. URL: https://sketchfab.com/mbari (visited on 08/13/2026).

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

[3] Bernhard Kerbl et al. "3D Gaussian Splatting for Real-Time Radiance Field Rendering". In: *ACM Transactions on Graphics* 42.4 (July 2023).

[4] Shuaitao Fan et al. "A review of recent advances in Gaussian splatting". In: *Applied Intelligence* 56.5 (Apr. 2026). ISSN: 0924-669X. DOI: 10.1007/s10489-026-07227-9.

[5] Ruijie Zhu et al. "ObjectGS: Object-Aware Scene Reconstruction and Scene Understanding via Gaussian Splatting". In: *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2025, pp. 8350–8360. DOI: 10.1109/ICCV51701.2025.00782.

[6] Derya Akkaynak and Tali Treibitz. "A Revised Underwater Image Formation Model". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2018.

[7] Daniel Yang, John J. Leonard, and Yogesh Girdhar. "SeaSplat: Representing Underwater Scenes with 3D Gaussian Splatting and a Physically Grounded Image Formation Model". In: *2025 IEEE International Conference on Robotics and Automation (ICRA)*. 2025, pp. 7632–7638. DOI: 10.1109/ICRA55743.2025.11128502.

[8] Olga Russakovsky et al. "ImageNet Large Scale Visual Recognition Challenge". In: 115.3 (Dec. 2015), pp. 211–252. ISSN: 0920-5691. DOI: 10.1007/s11263-015-0816-y.

[9] Jiadong Tang et al. "DroneSplat+: Semantics-Enhanced 3D Gaussian Splatting for Robust 3D Reconstruction From In-the-Wild Drone Imagery". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 48.10 (2026), pp. 12863–12879. DOI: 10.1109/TPAMI.2026.3700887.

[10] Nicolas Carion et al. *SAM 3: Segment Anything with Concepts*. 2025. arXiv: 2511.16719 [cs.CV].

[11] Zhuodong Jiang et al. *Semantic-guided Gaussian Splatting for High-Fidelity Underwater Scene Reconstruction*. 2026. arXiv: 2509.00800 [cs.CV].

[12] Kangxu Wang et al. "WaterSplat-SLAM: Photorealistic Monocular SLAM in Underwater Environment". In: *IEEE Robotics and Automation Letters* 11.5 (2026), pp. 5614–5621. DOI: 10.1109/LRA.2026.3668465.

[13] Richard Hartley, Andrew Zisserman, et al. *Multiple view geometry in computer vision*. Vol. 2. 3. Cambridge university press Cambridge, 2003.

[14] A. Torralba, P. Isola, and W.T. Freeman. *Foundations of Computer Vision*. Adaptive Computation and Machine Learning series. MIT Press, 2024. ISBN: 9780262378666.

[15] Noah Snavely, Steven Seitz, and Richard Szeliski. "Modeling the World from Internet Photo Collections". In: *International Journal of Computer Vision* 80 (Nov. 2008), pp. 189–210. DOI: 10.1007/s11263-007-0107-3.

[16] D.G. Lowe. "Object recognition from local scale-invariant features". In: *Proceedings of the Seventh IEEE International Conference on Computer Vision*. Vol. 2. 1999, 1150–1157 vol.2. DOI: 10.1109/ICCV.1999.790410.

[17] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. " SuperPoint: Self-Supervised Interest Point Detection and Description ". In: *2018 IEEE/CVF Conference on Computer Vision and Pat-*

tern Recognition Workshops (CVPRW). Los Alamitos, CA, USA: IEEE Computer Society, June 2018, pp. 337–33712. DOI: 10.1109/CVPRW.2018.00060.

[18] Paul-Edouard Sarlin et al. " SuperGlue: Learning Feature Matching With Graph Neural Networks ". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, June 2020, pp. 4937–4946. DOI: 10.1109/CVPR42600.2020.00499.

[19] Yu Chen, Yisong Chen, and Guoping Wang. "Bundle Adjustment Revisited". In: *ArXiv* abs/1912.03858 (2019).

[20] M.J. Westoby et al. "'Structure-from-Motion' photogrammetry: A low-cost, effective tool for geoscience applications". In: *Geomorphology* 179 (2012), pp. 300–314. ISSN: 0169-555X. DOI: https://doi.org/10.1016/j.geomorph.2012.08.021.

[21] Caroline Rodriguez. "Measuring the Impact of Thermal Stress on Coral Resilience in Hawai'i Using Large-Area Imagery". Master's Thesis. California State University, Monterey Bay, 2022.

[22] Hannah C. Barkley et al. "Estimating coral reef carbonate budgets using Structure-from-Motion photogrammetry". In: *Coral Reefs* 44.3 (June 2025), pp. 937–951. ISSN: 1432-0975. DOI: 10.1007/s00338-025-02660-7.

[23] Dan Cernea. *OpenMVS: Multi-View Stereo Reconstruction Library*. 2020.

[24] Johannes L. Schönberger et al. "Pixelwise View Selection for Unstructured Multi-View Stereo". In: *Computer Vision – ECCV 2016*. Ed. by Bastian Leibe et al. Cham: Springer International Publishing, 2016, pp. 501–518. ISBN: 978-3-319-46487-9.

[25] Michael Kazhdan and Hugues Hoppe. "Screened poisson surface reconstruction". In: *ACM Trans. Graph.* 32.3 (July 2013). ISSN: 0730-0301. DOI: 10.1145/2487228.2487237.

[26] Ben Mildenhall et al. "NeRF: representing scenes as neural radiance fields for view synthesis". In: *Commun. ACM* 65.1 (Dec. 2021), pp. 99–106. ISSN: 0001-0782. DOI: 10.1145/3503250.

[27] A. Tewari et al. "Advances in neural rendering". In: *ACM SIGGRAPH 2021 Courses*. SIGGRAPH '21. Virtual Event, USA: Association for Computing Machinery, 2021. ISBN: 9781450383615. DOI: 10.1145/3450508.3464573.

[28] Sara Fridovich-Keil et al. "Plenoxels: Radiance Fields Without Neural Networks". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2022, pp. 5501–5510.

[29] Thomas Müller et al. "Instant Neural Graphics Primitives with a Multiresolution Hash Encoding". In: *ACM Trans. Graph.* 41.4 (July 2022), 102:1–102:15. DOI: 10.1145/3528223.3530127.

[30] M. Zwicker et al. "EWA volume splatting". In: *Proceedings Visualization, 2001. VIS '01.* 2001, pp. 29–538. DOI: 10.1109/VISUAL.2001.964490.

[31] Tavus. *Gaussian Splatting from Scratch Through Code*. 2024. URL: https://www.tavus.io/blog/gaussian-splatting-from-scratch-through-code (visited on 08/24/2024).

[32] Tao Lu et al. "Scaffold-GS: Structured 3D Gaussians for View-Adaptive Rendering". In: *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2024, pp. 20654–20664. DOI: 10.1109/CVPR52733.2024.01952.

[33] *UW underwater absorption explanation*. University of Washington. 2024. URL: https://uw.pressbooks.pub/ocean285/chapter/absorption/ (visited on 08/25/2026).

[34] Sasha J Kramer et al. "Sinking particles exporting diatoms and Hacrobia predict the magnitude of oceanic POC flux". In: *The ISME Journal* 19.1 (Jan. 2025), wraf105. DOI: 10.1093/ismejo/wraf105.

eprint: https://academic.oup.com/ismej/article-pdf/19/1/wraf105/63299197/wraf105.pdf.

[35] Derya Akkaynak and Tali Treibitz. "Sea-Thru: A Method for Removing Water From Underwater Images". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2019.

[36] Karel Zuiderveld. "Contrast limited adaptive histogram equalization". In: *Graphics Gems IV*. USA: Academic Press Professional, Inc., 1994, pp. 474–485. ISBN: 0123361559.

[37] Samruddhi. *Guide to Underwater Image Enhancement Using OpenCV*. Dec. 2025. URL: https://opencv.org/underwater-image-enhancement-using-opencv/ (visited on 07/31/2026).

[38] Jiayi Wu et al. "Single-Step Latent Diffusion for Underwater Image Restoration". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2025), pp. 1–11. DOI: 10.1109/TPAMI.2025.3599775.

[39] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 [cs.CV].

[40] Joseph Redmon et al. "You Only Look Once: Unified, Real-Time Object Detection". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Las Vegas, NV, USA: IEEE, June 2016, pp. 779–788. ISBN: 978-1-4673-8851-1. DOI: 10.1109/CVPR.2016.91.

[41] Ashish Vaswani et al. "Attention is All you Need". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017.

[42] Gabriela Csurka, Riccardo Volpi, and Boris Chidlovskii. "Semantic Image Segmentation: Two Decades of Research". In: *Found. Trends Comput. Graph. Vis.* 14 (2023), pp. 1–162.

[43] Kakani Katija et al. "FathomNet: A global image database for enabling artificial intelligence in the ocean". In: *Scientific Reports* 12.1 (Sept. 2022), p. 15914. ISSN: 2045-2322. DOI: 10.1038/s41598-022-19939-2.

[44] Patrick Kage et al. *A Review of Pseudo-Labeling for Computer Vision*. 2025. DOI: https://doi.org/10.1613/jair.1.19656. arXiv: 2408.07221 [cs.CV].

[45] Yong He et al. "Deep Learning based 3D Segmentation: A Survey". In: *CoRR* abs/2103.05423 (2021). arXiv: 2103.05423.

[46] Yunhan Yang et al. *SAM3D: Segment Anything in 3D Scenes*. 2023. arXiv: 2306.03908 [cs.CV].

[47] Jonathan Sauder et al. "Scalable semantic 3D mapping of coral reefs with deep learning". In: *Methods in Ecology and Evolution* 15.5 (2024), pp. 916–934. DOI: https://doi.org/10.1111/2041-210X.14307. eprint: https://besjournals.onlinelibrary.wiley.com/doi/pdf/10.1111/2041-210X.14307.

[48] Z. Zhang et al. "Semantic-Consistent 3D Reconstruction via Gaussian Splatting and SAM-Guided Annotation". In: *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* XLVIII-1/W5-2025 (2025), pp. 185–192. DOI: 10.5194/isprs-archives-XLVIII-1-W5-2025-185-2025.

[49] Agisoft LLC. *Agisoft Metashape Professional*. Version 2.1. 2024.

[50] Gershon Buchsbaum. "A spatial processor model for object colour perception". In: *Journal of The Franklin Institute-engineering and Applied Mathematics* 310 (1980), pp. 1–26.

[51] Kurran Singh, Nick Rypkema, and John Leonard. "Attention-based Self-Supervised Hierarchical Semantic Segmentation for Underwater Imagery". In: *OCEANS 2023 - Limerick*. 2023, pp. 1–6. DOI: 10.1109/OCEANSLimerick52467.2023.10244736.

[52] Z. Wang, E.P. Simoncelli, and A.C. Bovik. “Multiscale structural similarity for image quality assessment”. In: *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*. Vol. 2. 2003, 1398–1402 Vol.2. DOI: 10.1109/ACSSC.2003.1292216.

[53] Richard Zhang et al. “The Unreasonable Effectiveness of Deep Features as a Perceptual Metric”. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2018, pp. 586–595. DOI: 10.1109/CVPR.2018.00068.

[54] Karen Simonyan and Andrew Zisserman. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. 2015. arXiv: 1409.1556 [cs.CV].

[55] Matthew Tancik et al. “Nerfstudio: A Modular Framework for Neural Radiance Field Development”. In: *ACM SIGGRAPH 2023 Conference Proceedings*. SIGGRAPH ’23. 2023.

[56] Vickie Ye et al. *gsplat: An Open-Source Library for Gaussian Splatting*. 2024. arXiv: 2409.06765 [cs.CV].

A Using SAM3 to create Semantic Masks

Notice that all the Masks referred as SAM3 Masks are pseudo-labeled semantic segmentation and not ground-truth labels, as extensively discussed in Section 3.3 and Section 4.3.

A.1 SAM3 Mask Examples

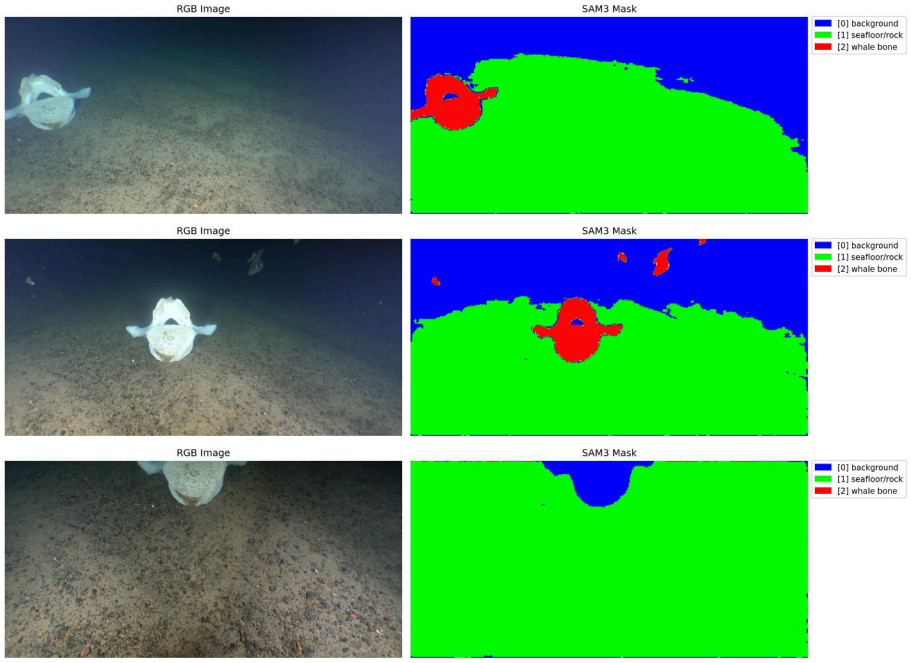


Figure 26: Example SAM3 masks on bone dataset (blue=background, green=seafloor, red=bone).

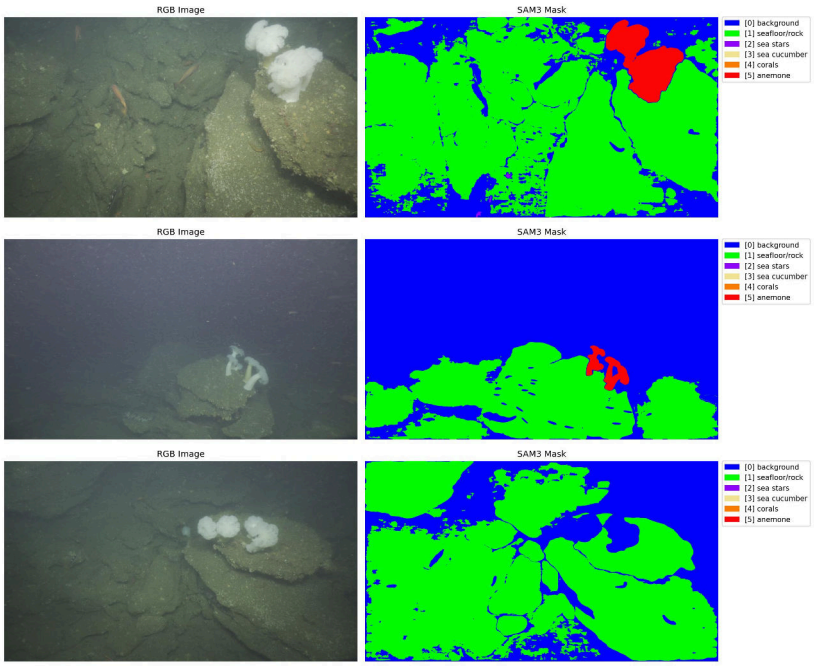


Figure 27: Example SAM3 masks on anemone dataset(blue=background, green=seafloor, red=anemone).

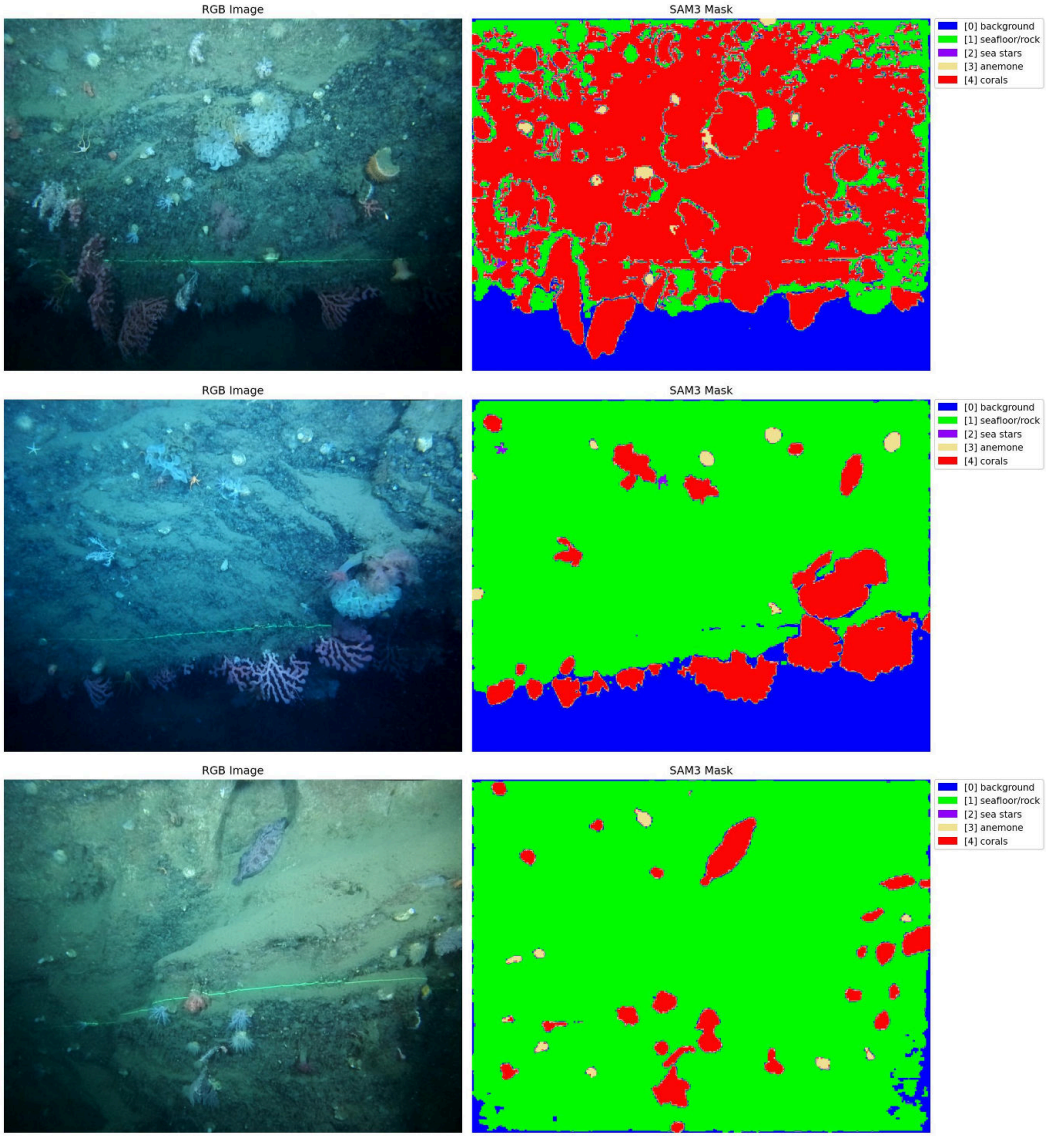


Figure 28: Example SAM3 masks on corals dataset
(blue=background, green=seafloor, red=coral, yellow=anemone, purple=starfish).

A.2 SAM3 Configuration

As explained in Section 3.3 text prompts are used to configure SAM3 instance segmentation model, here the configuration files for the three datasets discussed in Section 4.1 are reported.

```yaml
# Write order: lowest priority first, highest last
priority_order: [0, 1, 2, 3, 4, 5]
# Fixed colors per class ID (hex)
class_colors:
  0: "#0000FF"   # blue    -- background (and fish)
  1: "#00FC02"   # green   -- sea floor
  2: "#9105FC"   # purple  -- sea star
  3: "#F2E394"   # yellow  -- sea cucumber
  4: "#FC7D05"   # orange  -- corals
  5: "#FD0605"   # red     -- anemone

classes:
  - id: 0
    label: "background"
    confidence_threshold: 0.15
    prompts:
      - "open water"
      - "water column"
      - "deep ocean"
      - "fish"
      - "particles"

  - id: 1
    label: "seafloor/rock"
    confidence_threshold: 0.10
    prompts:
      - "rock"
      - "rocks"
      - "sand"
      - "silt"
      - "seafloor rock"
      - "underwater rock"
      - "ocean floor"


  - id: 2
    label: "sea stars"
    confidence_threshold: 0.10
    prompts:
      - "sea star"
      - "sea stars"
      - "brittle star"

  - id: 3
    label: "sea cucumber"
    confidence_threshold: 0.25
    prompts:
      - "sea cucumber"
      - "sea worm"

  - id: 4
    label: "corals"
    confidence_threshold: 0.15
    prompts:
      - "coral"
      - "soft coral"
      - "branching coral"
      - "gorgonian"

  - id: 5
    label: "anemone"
    confidence_threshold: 0.10
    prompts:
      - "sea anemone"
      - "anemone"
```

Figure 29: SAM3 configuration of anemone.

```yaml
# Write order: lowest priority first, highest last
priority_order: [0, 1, 2]
# Fixed colors per class ID (hex)
class_colors:
  0: "#0000FF"   # blue   -- background
  1: "#00FC02"   # green  -- sea floor/rock
  2: "#FD0605"   # red    -- whale bone

classes:
  - id: 0
    label: "background"
    confidence_threshold: 0.10
    prompts:
      - "ocean background"
      - "blue background"
      - "background"
      - "black background"
      - "fish"
      - "fishes"
      - "jellyfish"
      - "squid"


  - id: 1
    label: "seafloor/rock"
    confidence_threshold: 0.10
    prompts:
      - "sand floor"
      - "sea floor"
      - "sea bottom"
      - "sand"
      - "slit"
      - "rocks"

  - id: 2
    label: "whale bone"
    confidence_threshold: 0.07
    prompts:
      - "whale bone"
      - "white bone"
      - "bones"
      - "bone"
      - "skeleton"
      - "white skeleton"
      - "whale skeleton"
```

Figure 30: SAM3 configuration of bone.

```yaml
# Write order: lowest priority first, highest last
priority_order: [0, 1, 2, 3, 4]
# Fixed colors per class ID (hex)
class_colors:
  0: "#0000FF"   # blue    -- background
  1: "#00FC02"   # green   -- sea floor
  2: "#9105FC"   # purple  -- sea star
  3: "#F2E394"   # yellow  -- anemone
  4: "#FD0605"   # red     -- corals

classes:
  - id: 0
    label: "background"
    confidence_threshold: 0.10
    prompts:
      - "open water"
      - "water column"
      - "deep ocean water"
      - "empty water"
      - "underwater background"
      - "dark water"
      - "blue water"

  - id: 1
    label: "seafloor/rock"
    confidence_threshold: 0.10
    prompts:
      - "rock outcrop"
      - "rock wall"
      - "rock face"
      - "rock substrate"
      - "rock"
      - "rocks"
      - "sand"
      - "silt"
      - "seafloor rock"
      - "underwater rock"
      - "ocean floor"
      - "benthic substrate"
      - "hard substrate"
      - "sediment covered rock"

  - id: 2
    label: "sea stars"
    confidence_threshold: 0.10
    prompts:
      - "sea star"
      - "sea stars"
      - "brittle star"
      - "fire star"
      - "starfish"
      - "snake star"
      - "echinoderm"

  - id: 3
    label: "anemone"
    confidence_threshold: 0.15
    prompts:
      - "sea anemone"
      - "anemone"
      - "yellow anemone"
      - "white sea anemone"
      - "tentacled anemone"
      - "colonial anemone"
      - "tube anemone"
      - "small sea anemone"
      - "anemone colony"

  - id: 4
    label: "corals"
    confidence_threshold: 0.20
    prompts:
      - "soft coral"
      - "corals"
      - "coral"
      - "branching soft coral"
      - "gorgonian coral"
      - "sea fan coral"
      - "octocoral"
      - "branching coral"
      - "pink coral"
      - "purple coral"
      - "fan shaped coral"
      - "deep sea coral"
```

Figure 31: SAM3 configuration of corals.

B Semantic Lifting Algorithms

As anticipated in Section 2.5 a simple semantic lifting approach can be used to project semantic masks to a sparse 3D point cloud. A clear problem is the multi-view inconsistency that can be handled via different voting strategies. The algorithms below are taken from ObjectGS[5] paper and adapted to the notation used here. The main differences rely on how the label is selected (voting strategy) and how the 3D-to-2D correspondence is solved. Regarding the labeling, majority (Algorithm 1) and correspondence (Algorithm 3) select the most common label associated with the projected 3D point on the semantic mask, while probability (Algorithm 2) defines a probability distribution based on class frequency and samples from it. In terms of projection, majority and probability use the intrinsic camera matrix estimated by the SfM process, while correspondence relies on the feature matching dictionary that software like COLMAP automatically saves in the output data. The list of correspondences appends to each 3D point the corresponding image in which it is seen and the 2d location.

Algorithm 1 Object ID Majority Voting

Input:
  Point cloud: $\mathcal{P} = \{P_1, P_2, \ldots, P_{N_P}\}$
  Semantic masks: $\mathcal{M} = \{L_1, L_2, \ldots, L_{N_I}\}$
  Camera poses: $\mathcal{T} = \{T_1, T_2, \ldots, T_{N_I}\}$
  Projection function: $x_i = \text{Project}(P_i, T_j)$ based on Intrinsics $\mathcal{K}$
Initialization:
  labels = $\emptyset$
  $\mathcal{P}_L = \emptyset$
for each point $P_i \in \mathcal{P}$ **do**
    for each camera pose $T_j \in \mathcal{T}$ **do**
      $x_i = \text{Project}(P_i, T_j)$
      Append $M_j(x_i)$ to labels$[P_i]$
    end for
end for
for each point $P_i \in \mathcal{P}$ **do**
    if labels$[P_i] \neq \emptyset$ **then**
      frequency(ID) = Counter(labels$[P_i]$)
      ID = arg max frequency(ID)
    end if
    $P_{i,L} = (P_i, \text{object ID})$
    Append $P_{i,L}$ to $\mathcal{P}_L$
end for
Output: Semantic point cloud $\mathcal{P}_L$ with object IDs.

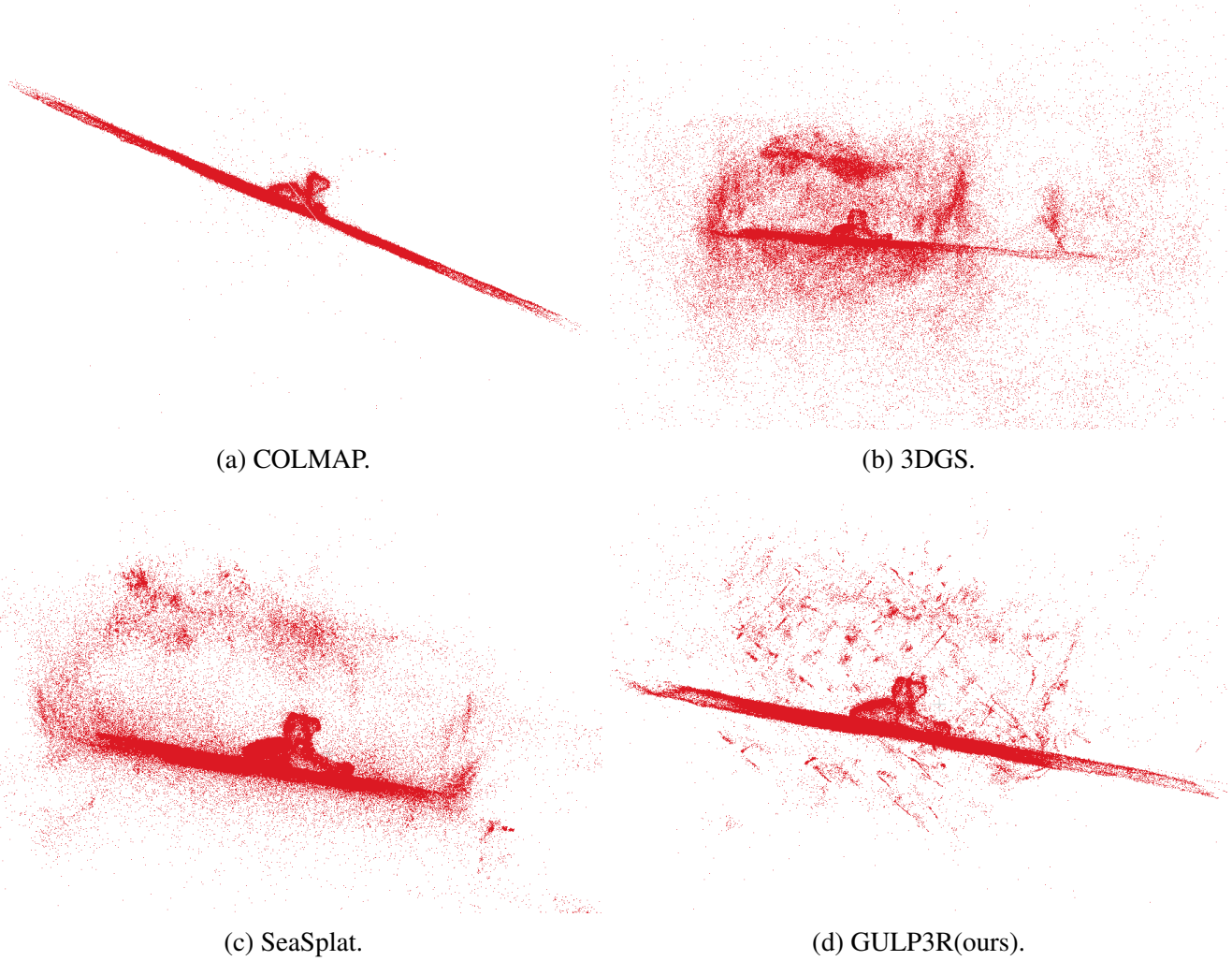
Algorithm 2 Object ID Probability-based Voting

Input:
  Point cloud: $\mathcal{P} = \{P_1, P_2, \ldots, P_{N_P}\}$
  Semantic masks: $\mathcal{M} = \{L_1, L_2, \ldots, L_{N_I}\}$
  Camera poses: $\mathcal{T} = \{T_1, T_2, \ldots, T_{N_I}\}$
  Projection function: $x_i = \text{Project}(P_i, T_j)$ based on Intrinsics $\mathcal{K}$
Initialization:
  labels = $\emptyset$
  $\mathcal{P}_L = \emptyset$
for each point $P_i \in \mathcal{P}$ **do**
  for each camera pose $T_j \in \mathcal{T}$ **do**
    $x_i = \text{Project}(P_i, T_j)$
    Append $L_j(x_i)$ to labels$[P_i]$
  end for
end for
for each point $P_i \in \mathcal{P}$ **do**
  if labels$[P_i] \neq \emptyset$ **then**
    frequency(ID) = Counter(labels$[P_i]$)
    ID = Random(Prob = frequency(ID))
  end if
  $P_{i,L} = (P_i, \text{object ID})$
  Append $P_{i,L}$ to $\mathcal{P}_L$
end for
Output: Semantic point cloud $\mathcal{P}_L$ with object IDs.

Algorithm 3 Object ID Correspondence-based Voting

Input:
  Point cloud: $\mathcal{P} = \{P_1, P_2, \ldots, P_{N_P}\}$
  Semantic masks: $\mathcal{M} = \{L_1, L_2, \ldots, L_{N_I}\}$
  Correspondences: $C = \{C_1, C_2, \ldots, C_N\}$
  Projection function: $x_i = \text{Project}(P_i, C_j)$ based on Correspondences C
Initialization:
  labels = $\emptyset$
for each point $P_i \in \mathcal{P}$ **do**
  for each correspondence $C_j \in C$ **do**
    $x_i = \text{Project}(P_i, C_j)$
    Append $M_j(x_i)$ to labels$[P_i]$
  end for
end for
for each point $P_i \in \mathcal{P}$ **do**
  if labels$[P_i] \neq \emptyset$ **then**
    frequency(ID) = Counter(labels$[P_i]$)
    ID = arg max frequency(ID)
  end if
  $P_{i,L} = (P_i, \text{object ID})$
  Append $P_{i,L}$ to $\mathcal{P}_L$
end for
Output: Semantic point cloud $\mathcal{P}_L$ with object IDs.

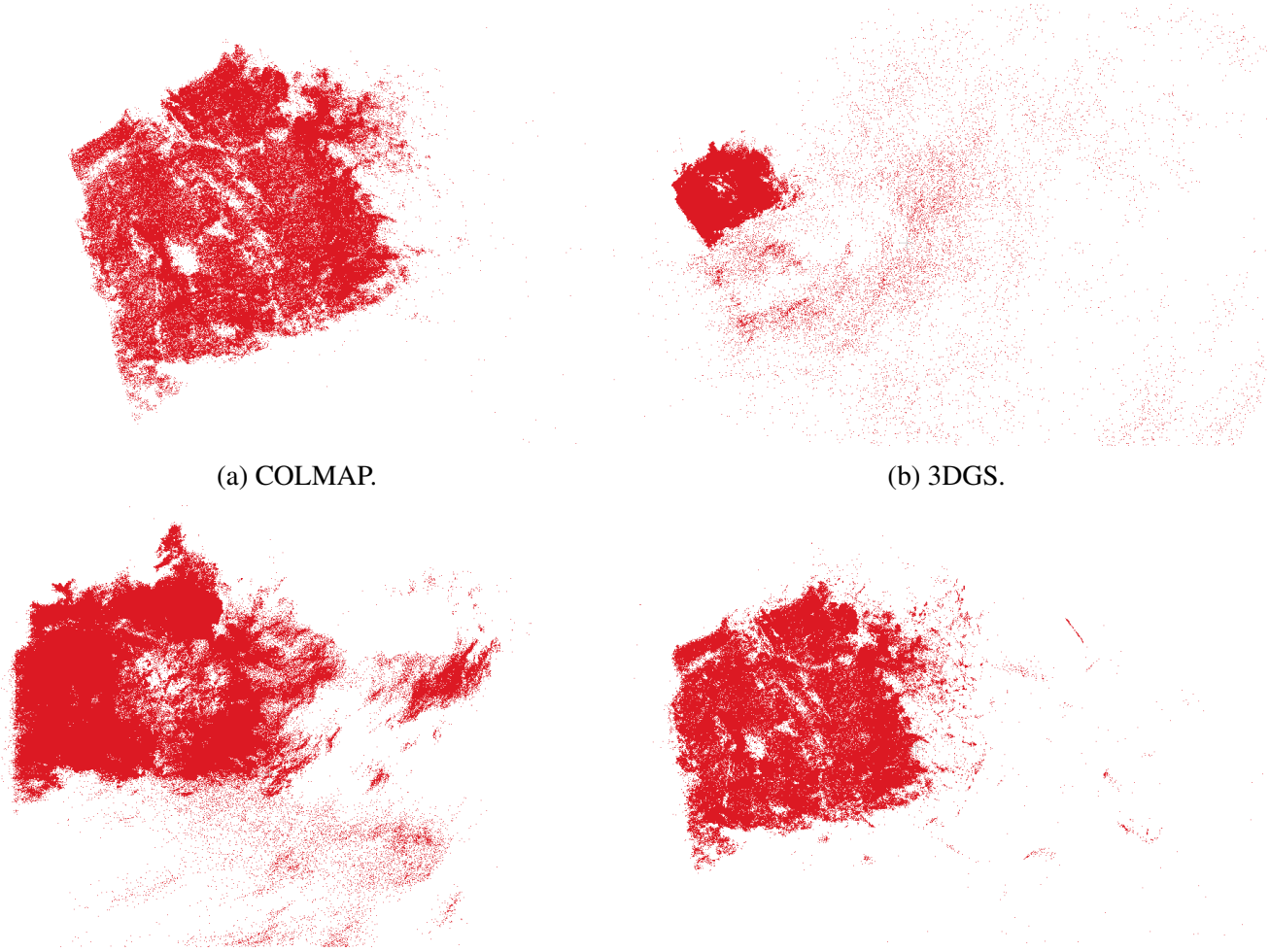
C Supplementary Results Figures



(a) COLMAP. (b) 3DGS.

(c) SeaSplat. (d) GULP3R(ours).

Figure 32: Comparison between COLMAP, 3DGS, SeaSplat, GULP3R point clouds on the bone dataset. Here the RGB points of the reconstructed geometry are reported as red, in order to highlight the spatial coverage and geometric distribution of the compared reconstruction techniques. The red points are either the coordinates P_j of the SfM point cloud output $\mathcal{P}$ (a) or the centers (mean μ_j) of a gaussian G_j in the gaussian cloud $\mathcal{G}$ (b,c,d).



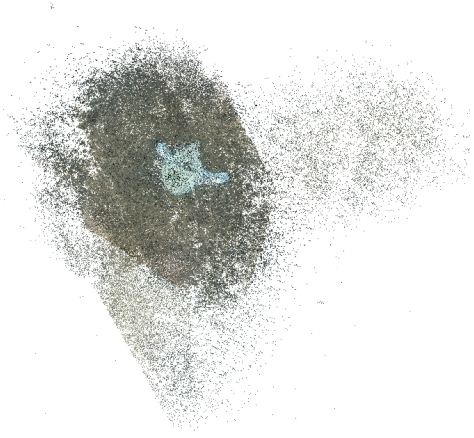
(a) COLMAP.

(b) 3DGS.

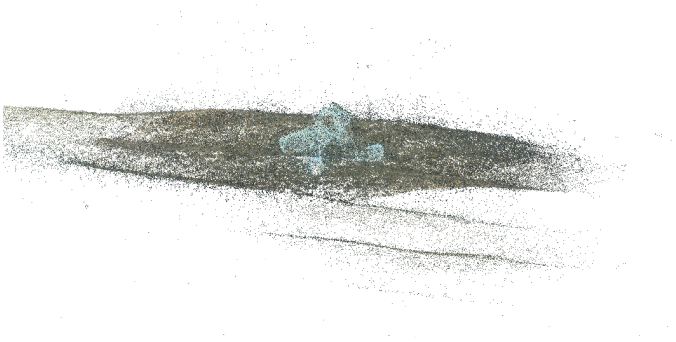
(c) SeaSplat.

(d) GULP3R(ours).

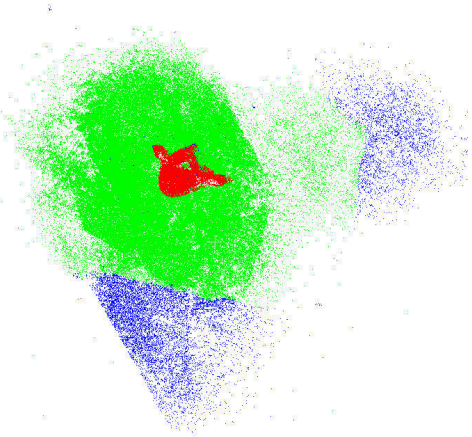
Figure 33: Comparison between COLMAP, 3DGS, SeaSplat, GULP3R point clouds on the corals dataset. Here the RGB points of the reconstructed geometry are reported as red, in order to highlight the spatial coverage and geometric distribution of the compared reconstruction techniques. The red points are either the coordinates P_j of the SfM point cloud output $\mathcal{P}$ (a) or the centers (mean μ_j) of a gaussian G_j in the gaussian cloud $\mathcal{G}$ (b,c,d).



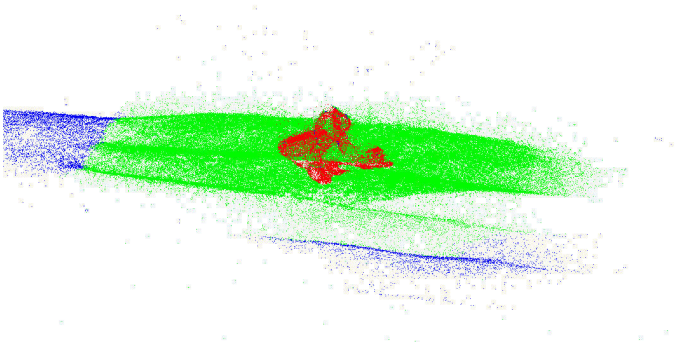
(a) Colored point cloud V1.



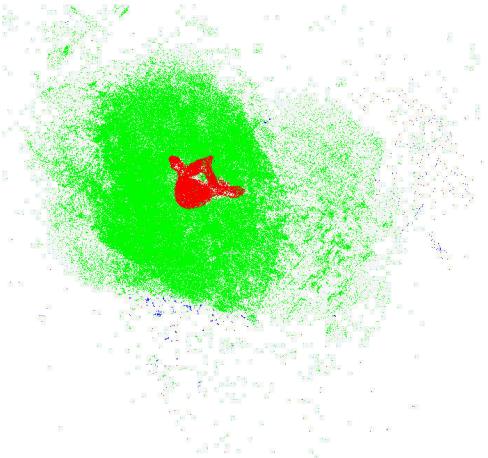
(b) Colored point cloud V2.



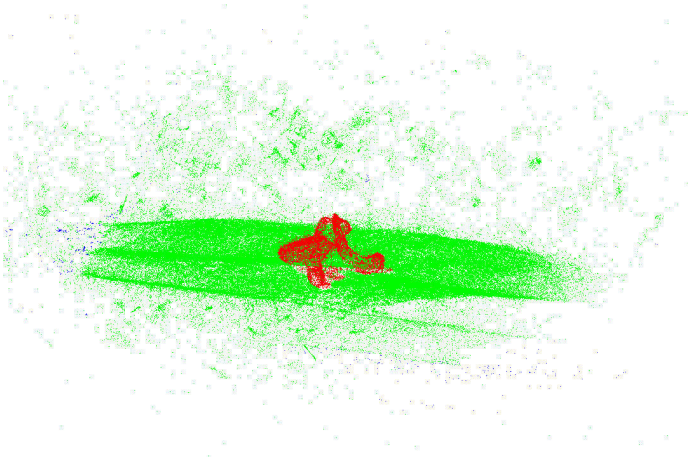
(c) Lifted point cloud
(majority voting, Algorithm 1) V1.



(d) Lifted point cloud
(majority voting, Algorithm 1) V2.

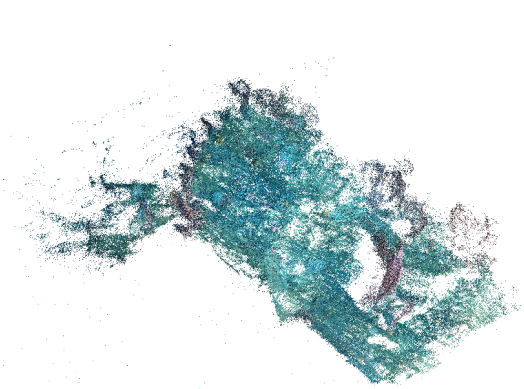


(e) GULP3R semantic Gaussian cloud V1.

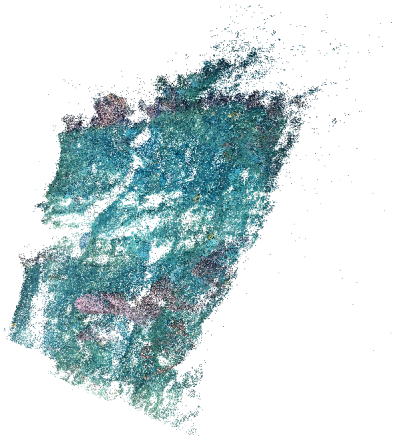


(f) GULP3R semantic Gaussian cloud V2.

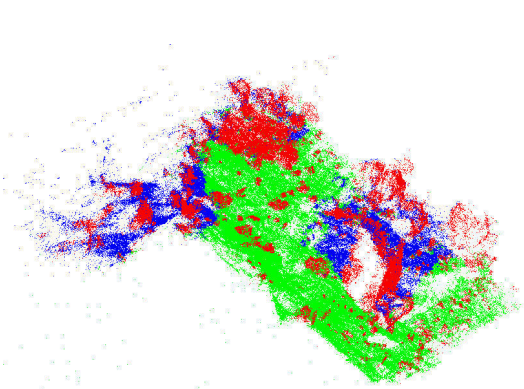
Figure 34: Two perspective views (V1, V2) on the SfM RGB point cloud (a,b), lifted semantic(c,d), and GULP3R semantic Gaussian cloud (e,f) for the bone dataset. Point labels are color encoded as: blue=background, green=seafloor, red=bone.



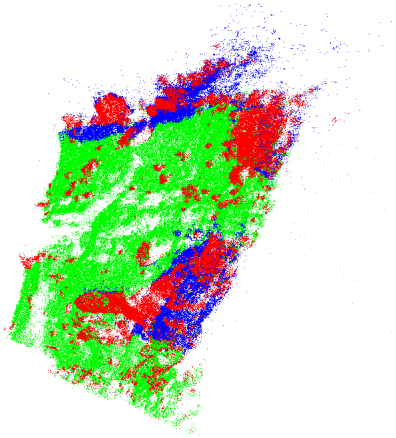
(a) Colored point cloud V1.



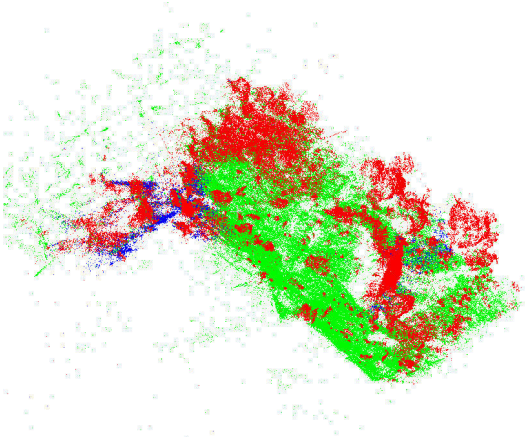
(b) Colored point cloud V2.



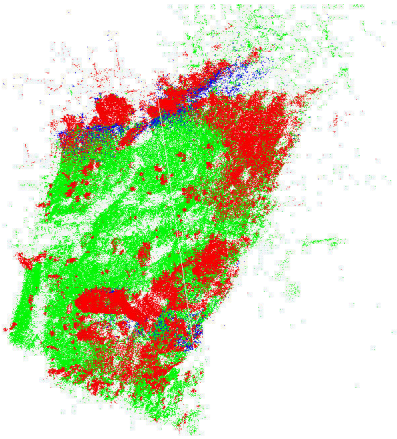
(c) Lifted point cloud (majority voting, Algorithm 1) V1.



(d) Lifted point cloud (majority voting, Algorithm 1) V2.



(e) GULP3R semantic Gaussian cloud V1.



(f) GULP3R semantic Gaussian cloud V2.

Figure 35: Two perspective views (V1, V2) on the SfM RGB point cloud (a,b), lifted semantic(c,d), and GULP3R semantic Gaussian cloud (e,f) for the corals dataset. Point labels are color encoded as: blue=background, green=seafloor, red=corals.