# Semantic Segmentation and Depth Estimation for Real-Time Lunar Surface Mapping Using 3D Gaussian Splatting

**Guillem Casadesus Vila, Adam Dai, and Grace Gao**
**Department of Aeronautics and Astronautics**
**Stanford University**
**Stanford, CA, United States**
**{guillemc,adai,gracegao}@stanford.edu**

***Abstract*— Navigation and mapping on the lunar surface require robust perception under challenging conditions, including low texture, harsh lighting, and limited sensor resources. We explore how fine-tunnoing image segmentation methods trained on Earth environments perform in lunar conditions and their impact on mapping using 3D Gaussian Splatting (3DGS).**

## TABLE OF CONTENTS



## 1. INTRODUCTION

*Motivation*

The renewed focus on robotic exploration of the lunar surface presents distinct challenges for autonomous navigation [**?**]. Long-range traverses are required in environments characterized by poorly textured regolith, high-contrast shadows, and the absence of atmospheric scattering. Furthermore, rovers are often limited to radiation-hardened computing hardware and vision-based sensors, precluding the use of active sensors like LiDAR. These constraints necessitate the development of computationally efficient, vision-based mapping and localization solutions.

Simultaneous Localization and Mapping (SLAM) is a standard approach for building a map of an unknown environment while tracking an agent's position within it. Classical SLAM methods, such as those based on geometric features, can operate in real time but exhibit reduced performance in the texture-deficient and high-contrast lighting conditions common on the Moon. While learning-based SLAM methods can improve robustness by learning feature representations from data, their generalization to novel environments like the lunar surface is not guaranteed. A common limitation of both classical and early learning-based methods is their reliance on discrete map representations (e.g., point clouds, voxels), which can be memory-intensive and may not capture fine surface detail.

*Related Work*

Recent advances in neural scene representations have produced methods capable of high-fidelity 3D reconstruction from images. A recent survey [1] details the rapid evolution of this field for terrestrial applications, covering approaches that use various sensor modalities (e.g., monocular, stereo, depth, IMU) and scene encodings (e.g., Neural Radiance Fields (NeRF), 3D Gaussian Splatting (3DGS), neural point clouds). These systems often rely on perception models for tasks like feature extraction, depth estimation, or semantic understanding. However, since these perception components are typically trained on terrestrial data, it is not clear whether their performance will generalize to the unique visual characteristics of extraterrestrial environments.

Semantic understanding of planetary environments is an active area of research. A systematic literature review on semantic terrain segmentation for planetary rovers [2] highlights a gap in existing solutions, noting that no current method simultaneously achieves pixel-level accuracy, real-time inference, and compatibility with onboard hardware. Specific models have shown promise for feature identification, such as detecting impact craters from digital elevation models [3]. Despite these advances in perception, the integration of such semantic information into modern neural mapping frameworks for planetary surfaces has not been thoroughly investigated.

Several works have applied neural scene representations to model the lunar surface. Some have focused on surface reconstruction from orbital imagery to generate digital elevation models, often to handle challenging illumination in permanently shadowed regions [4, 5]. More relevant to rover navigation, a few recent studies have explored using NeRFs with surface-level imagery for localization, mapping, and path planning [6–9]. These approaches demonstrate the potential of neural representations for capturing lunar terrain.

However, existing works for surface mapping predominantly rely on NeRF [10], which has several limitations for real-time, incremental rover navigation. First, NeRF's rendering process is computationally expensive due to its reliance on volumetric sampling, making it ill-suited for real-time applications on resource-constrained hardware. Second, the underlying multi-layer perceptron (MLP) represents a fixed volume, which is difficult to extend as the rover explores new areas and is not easily deformable to accommodate loop

979-8-3315-7360-7/26/$31.00 ©2026 IEEE

closures. This necessitates stitching multiple models, which can introduce inconsistencies. Finally, the implicit nature of NeRF makes it less interpretable and difficult to integrate with or query for explicit semantic information. Many of these works also assume diverse camera viewpoints during training, a condition not met by the typically forward-facing trajectory of a rover.

*Contributions*

This work addresses the aforementioned limitations by developing a framework for building semantically-aware, large-scale maps of the lunar surface in real time. Our primary contributions are:

- We propose a real-time mapping framework for lunar surface environments based on 3D Gaussian Splatting, which offers fast rendering and a flexible, explicit scene representation suitable for incremental updates.
- We integrate and benchmark a suite of dense perception models, including semantic segmentation and both stereo and monocular depth estimation networks adapted for lunar conditions. We analyze the direct impact of these perception inputs on the geometric and semantic quality of the final 3D reconstruction.

*Paper Organization*

The rest of the paper is organized as follows.

As traverse speeds increase, real-time perception becomes even more critical for safe autonomous navigation. Accurate dense depth estimation and semantic segmentation are essential for precise distance measurements to hazards, untraversable terrain, and shadowed regions where obstacles may be hidden. These complementary modalities provide the geometric and semantic understanding of the scene necessary for safe and efficient navigation. In this work, we propose a real-time 3D Gaussian splatting framework for mapping that leverages both depth estimation and semantic segmentation models adapted for lunar environments. Our approach focuses on improving the performance of these vision-based perception modules under challenging lunar conditions, enabling robust real-time mapping using neural scene representations.

## 2. Datasets and Models

We used three lunar environments to test depth estimation and scene reconstruction methods, each providing different levels of simulation realism and control.

This work develops robust perception systems for lunar rovers to enable safe and efficient navigation in challenging

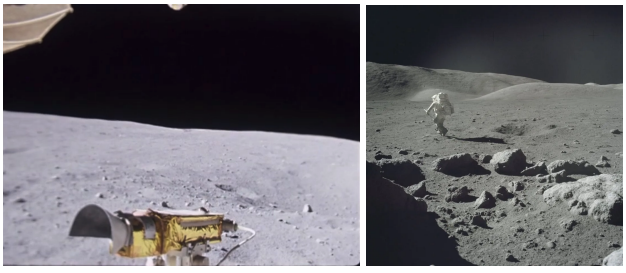


**Figure 1**: **Images from the Apollo missions [11].**

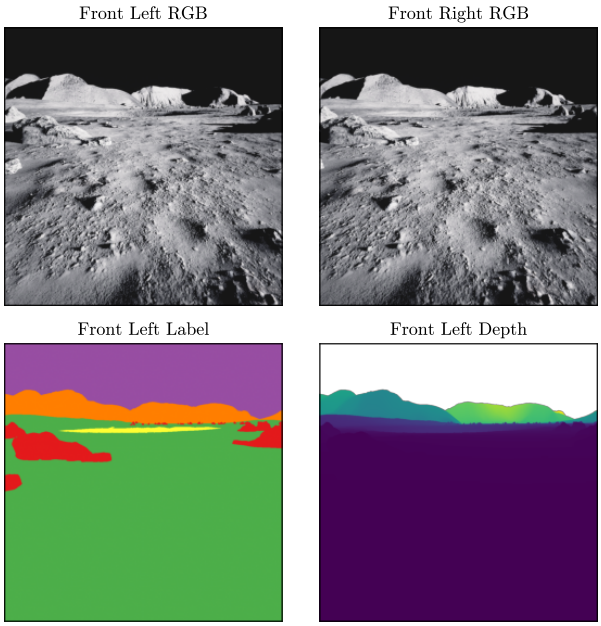



**Figure 2**: **Sample images from the LuSNAR dataset.**

lunar environments. Accurate scene understanding is crucial for hazard avoidance, as lunar surfaces contain numerous obstacles like rocks, craters, and steep slopes that could damage the rover. Additionally, long-term operations require detailed mapping capabilities that combine geometric layout with semantic understanding of terrain types and features for better mission planning and scientific exploration.

To achieve these goals, we focus on three key perception tasks: semantic segmentation to identify and classify scene elements like rocks and craters; depth estimation to understand the 3D structure of the environment; and scene reconstruction to create detailed 3D maps that combine geometric and semantic information. By integrating these components, we aim to create a robust perception system that supports both immediate navigation needs and long-term operational requirements in lunar environments.

While semantic segmentation has shown remarkable success in Earth-based scenarios, its effectiveness in lunar environments remains uncertain. The unique characteristics of lunar terrain - including extreme lighting conditions, lack of atmospheric scattering, and distinctive surface features like regolith and craters - present challenges that differ significantly from terrestrial scenes. Models trained on Earth datasets may struggle to recognize and segment these lunar-specific features, as they are optimized for features commonly found in terrestrial environments like vegetation, buildings, and roads.

To address this uncertainty, we conducted a comprehensive benchmarking study of various semantic segmentation architectures. Our goal was to identify which models best adapt to lunar conditions and understand their strengths and limitations in this challenging environment. We selected a diverse set of architectures ranging from traditional convolutional networks to modern transformer-based approaches, each with different characteristics in terms of feature extraction, attention mechanisms, and multi-scale processing capabilities.

*Datasets*

We use the Lunar Segmentation, Navigation, and Reconstruction (LuSNAR) dataset, a comprehensive lunar benchmark dataset designed for evaluating autonomous perception and navigation systems. It contains high-resolution stereo image pairs, panoramic semantic labels, dense depth maps, LiDAR point clouds, and rover position data. The dataset consists of 9 simulated lunar scenes created using Unreal Engine, each varying in terrain features and object density. These scenes are carefully designed to represent different lunar surface conditions, from flat plains to cratered regions with varying rock distributions. As demonstrated in [12], the dataset enables evaluation of key perception tasks including semantic segmentation, 3D reconstruction, and autonomous navigation, making it a valuable resource for testing and validating lunar surface algorithms. However, the dataset has some limitations: it does not provide full 3D geometry information, lacks closed-loop simulation capabilities, and has limited control over illumination conditions, which are crucial for testing robustness under varying lighting scenarios.

*Stereo Depth Estimation*

- **Block Matching (BM)**: Computes disparity by comparing fixed-size image patches along epipolar lines using a cost function, such as Sum of Absolute Differences (SAD). Its performance is limited in textureless regions and under non-ideal illumination.

- **Semi-Global Matching (SGM)** [13]: Aggregates matching costs along multiple 1D paths across the image to approximate a 2D global smoothness constraint. This approach combines the efficiency of local methods with improved performance in low-texture areas.

- **RAFT-Stereo** [14]: Adapts the RAFT architecture from optical flow by constructing a 3D correlation volume of all disparities at all pixels. A recurrent, GRU-based unit then iteratively updates a high-resolution disparity field from this volume.

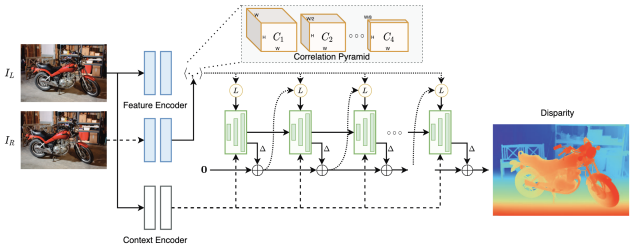



**Figure 3**: **RAFT-Stereo architecture (adapted from [14]).**

- **CREStereo** [15]: A cascaded recurrent network that operates in a coarse-to-fine manner. It employs recurrent refinement units at each stage and uses an adaptive group correlation layer to handle large displacements between stereo images.

*Monocular Depth Estimation*

- **Depth Anything V2** [**?**]: A transformer-based model trained on a dataset of over 62 million synthetic and pseudo-labeled real images. It is released in variants from 25M to 1.3B parameters for zero-shot relative depth estimation.

- **GLPN** [16]: An architecture that uses a transformer-based encoder to capture global context and a lightweight decoder. A selective feature fusion module combines features from different stages of the encoder.

- **DPT** [17]: Employs a Vision Transformer (ViT) as its backbone for dense prediction tasks. It processes feature maps at a constant resolution, providing global receptive fields at every stage of the network.

- **Depth Pro** [18]: A model for zero-shot metric depth estimation that uses a multi-scale vision transformer and is designed to operate on high-resolution inputs (e.g., 1536x1536).

*Semantic Segmentation*

- **U-Net** [19]: An encoder-decoder architecture with skip connections that concatenate features from the downsampling path to the upsampling path to retain spatial information.

- **U-Net++** [20]: Modifies the U-Net skip pathways with nested and dense convolutional blocks to reduce the semantic gap between encoder and decoder features.

- **MA-Net** [21]: Augments an encoder-decoder network with a module that combines position-wise and channel-wise attention to capture contextual dependencies.

- **LinkNet** [22]: A lightweight encoder-decoder network for real-time applications that passes encoder features at each level directly to the corresponding decoder level.

- **FPN** [23]: Constructs a multi-scale feature pyramid using a top-down pathway and lateral connections to merge semantic information from deep layers with spatial information from shallow layers.

- **PSPNet** [24]: Introduces a pyramid pooling module that applies pooling operations at multiple scales to aggregate global context information.

- **PAN (Path Aggregation Network)** [25]: Augments the top-down feature pyramid with an additional bottom-up pathway to shorten the information path for low-level features.

- **DeepLabV3** [26]: Uses atrous (dilated) convolutions to control the spatial resolution of feature maps and an Atrous Spatial Pyramid Pooling (ASPP) module to probe features at multiple scales.

- **DeepLabV3+** [27]: Extends DeepLabV3 by adding a decoder module to refine object boundaries and uses depthwise separable convolutions for computational efficiency.

- **UPerNet** [28]: A unified framework that combines a Feature Pyramid Network (FPN) backbone with a pyramid pooling module to parse features at various scales simultaneously.

- **Segformer** [29]: A transformer-based model using a hierarchical transformer encoder to produce multi-scale features without positional encodings, combined with a lightweight multilayer perceptron (MLP) decoder.

- **DPT** [17]: Employs a Vision Transformer (ViT) as its backbone, providing global receptive fields at every stage of the feature extraction process for dense prediction tasks.

*3D Gaussian Splatting*

3D Gaussian Splatting (3DGS) [30] is a rasterization-based method for novel view synthesis that represents a 3D scene with a collection of explicit, optimizable primitives. Unlike implicit representations like NeRF [10], 3DGS uses thousands to millions of 3D Gaussians to explicitly model the scene's geometry and appearance, as illustrated in Fig. 4.

Each Gaussian is defined by a set of learnable parameters:

• **Position (Mean):** $\boldsymbol{\mu} \in \mathbb{R}^3$ determines its location in 3D space.

• **Covariance:** $\boldsymbol{\Sigma} \in \mathbb{R}^{3\times3}$ defines its shape and orientation. To ensure $\boldsymbol{\Sigma}$ is always a valid positive semi-definite matrix and to allow for intuitive optimization, it is parameterized by a scaling vector $\mathbf{s} \in \mathbb{R}^3$ and a rotation quaternion $\mathbf{q} \in \mathbb{R}^4$.

$$\boldsymbol{\Sigma} = \mathbf{R}\mathbf{S}\mathbf{S}^\top\mathbf{R}^\top \tag{1}$$

where $\mathbf{R}$ is the rotation matrix derived from $\mathbf{q}$ and $\mathbf{S}$ is a diagonal scaling matrix derived from $\mathbf{s}$.

• **Opacity:** $\alpha \in [0, 1]$ controls the transparency of the Gaussian.

• **Color:** View-dependent color is modeled using Spherical Harmonics (SH) coefficients.

*Differentiable Rendering*—To synthesize outputs from a novel viewpoint, the 3D Gaussians are projected onto the 2D image plane, forming elliptical "splats." These splats are sorted by depth and composited in a front-to-back order using alpha blending. This differentiable rendering process produces not only the final RGB image ($\hat{I}$), but also a depth map and an accumulation (opacity) map. The rendered image can be directly compared to a ground truth image for gradient-based optimization, while the depth and accumulation maps provide additional supervision or regularization signals as needed.

*Adaptive Densification Strategy*— A key component of the 3DGS training process is the strategy for adaptive densification, which dynamically adjusts the set of Gaussians to efficiently represent the scene. This process is governed by periodic checks that add or remove primitives based on three main heuristics: the magnitude of the positional image-plane gradient, the 3D scale, and the opacity of each Gaussian. The strategy consists of three primary operations: growing, pruning, and opacity reset.

• **Growing (Densification)**: To represent complex regions that are not yet well-reconstructed, new Gaussians are introduced where the image-plane positional gradients exceed a threshold. This indicates that the optimizer is struggling to place the existing Gaussians correctly. Two methods are used:

– Duplication (Cloning): A Gaussian with a high gradient and a small 3D scale is considered to be in an under-reconstructed area that requires more detail. It is duplicated, and the optimizer then positions the new copy to better fill the local region.

– Splitting: A Gaussian with a high gradient and a large 3D scale is likely trying to represent a complex area that is too large for a single primitive. It is split into two smaller Gaussians, which can then more accurately model the underlying geometry.

• **Pruning**: To maintain a compact representation and remove artifacts, unnecessary Gaussians are pruned. A Gaussian is removed if it meets certain criteria, such as:

– Its opacity $\alpha$ falls below a minimum threshold, rendering it effectively invisible.

– Its 3D scale grows excessively large, which can cause blurry or hazy artifacts.

• **Opacity Reset**: Periodically during training, the opacities of all Gaussians are reset to a low value. This acts as a regularizer, forcing the model to re-evaluate the importance of each Gaussian. Primitives that are essential to the reconstruction will quickly regain high opacity, while transient or unnecessary ones will fail to do so and be removed in a subsequent pruning phase.

**Figure 4**: **3D Gaussian Splatting (adapted from [31]).**

*Loss Function*—The total loss $\mathcal{L}_{\text{total}}$ used to train the standard 3DGS model is a weighted sum of a reconstruction loss and a scale regularization term.

The reconstruction loss $\mathcal{L}_{\text{recon}}$ combines the L1 and D-SSIM losses, balanced by a hyperparameter $\lambda_{\text{SSIM}}$:

$$\mathcal{L}_{\text{recon}} = (1 - \lambda_{\text{SSIM}}) \cdot \|I - \hat{I}\|_1 + \lambda_{\text{SSIM}} \cdot \left(1 - \text{SSIM}(I, \hat{I})\right) \tag{2}$$

where $I$ is the ground truth image and $\hat{I}$ is the rendered image.

To prevent the Gaussians from becoming overly stretched, a scale regularization loss $\mathcal{L}_{\text{scale}}$ is applied. For each Gaussian $g$ with a scale vector $\mathbf{s}_g$ out of $N_g$ total Gaussians, this loss penalizes the ratio of its largest to smallest scale component if it exceeds a threshold $\tau_{\text{ratio}}$:

$$\mathcal{L}_{\text{scale}} = \frac{1}{N_g}\sum_{g=1}^{N_g} \max\left(0, \frac{\max(\mathbf{s}_g)}{\min(\mathbf{s}_g)} - \tau_{\text{ratio}}\right) \tag{3}$$

The final loss is the sum of these two components, with a weighting factor $\lambda_{\text{scale}}$ for the regularization term:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{recon}} + \lambda_{\text{scale}}\mathcal{L}_{\text{scale}} \tag{4}$$

## 3. METHODOLOGY

Our primary contribution is a real-time pipeline for the incremental, semantic 3D reconstruction of the lunar surface using a 3D Gaussian Splatting (3DGS) representation. The framework is designed as the mapping component of a SLAM system, processing sensor data to build and continuously optimize a semantic 3DGS map. The architecture is composed of three principal stages: a perception frontend, an incremental mapping backend, and a keyframe-based optimization engine.

*Perception and Pose Frontend*

The frontend processes raw sensor data into a structured format. We assume that rover pose estimates are continuously available from an external tracking system, such as visual-inertial odometry; the development of this tracking component is outside the scope of this work. For each incoming camera frame, the frontend computes a dense depth map using either traditional stereo algorithms or monocular depth estimation networks, demonstrating modular support for various sensor configurations. Concurrently, a semantic segmentation network classifies each pixel into relevant lunar

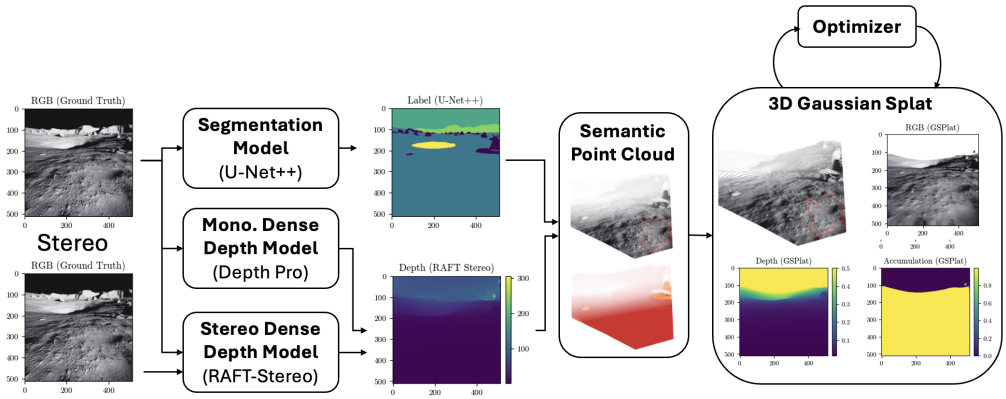



**Figure 5**: **Diagram of the proposed method.**

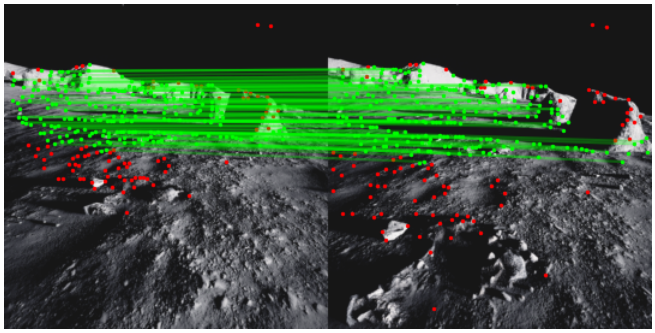


**Figure 6**: **Matches between consecutive frames used for monocular depth estimation scaling.**

categories (e.g., regolith, rock), providing critical context. Pixels classified as sky are explicitly masked to prevent the erroneous creation of 3D points.

For monocular depth estimation models, we employ a scaling strategy using triangulated points from matched features between consecutive frames. We detect and match keypoints using SuperPoint [32] and SuperGlue [33], as shown in Fig. 6, then triangulate these points to obtain metric depth estimates. We use these triangulated depths to scale the output of the monocular models. To ensure robust scaling, we apply depth masking to filter out unreliable matches and use random sample consensus (RANSAC) to remove outliers from the triangulated points. Specifically, we solve for the scale parameter $\theta$ and offset $\gamma$ by minimizing:

$$\min_{\theta,\gamma} \sum_{p \in M_{\text{valid}}} \|\theta \hat{D}(p) + \gamma - D(p)\|_2^2, \tag{5}$$

where $\hat{D}$ represents the estimated depth from the monocular model, $D$ is the ground truth depth from triangulation, and $M_{\text{valid}}$ is the mask of valid depth points after filtering.

*Incremental Mapping*

The backend receives the processed frames from the frontend and is responsible for incrementally building the global 3D map. For each new frame, the estimated pose, dense depth map, and semantic labels are fused to generate a registered, per-frame 3D semantic point cloud. This step transforms the 2D perception outputs into a 3D context where every point is associated with a color and a semantic class. This point cloud is then used to add new primitives to the global 3DGS model. Each new Gaussian's size is set proportional to the average distance to its nearest neighbors, ensuring adaptive coverage based on local point density. To maintain real-time performance and prevent unbounded map growth over long trajectories, a voxel-based filtering strategy is employed. Only 3D points that fall into previously unobserved regions of space are used to initialize new Gaussians in order to maintain real-time performance and prevent unbounded map growth. A key aspect of our system is that each new Gaussian is initialized with the semantic label from its corresponding point.

*Optimization Backend*

The optimization engine runs as a background process, continuously refining all Gaussian parameters and camera poses to improve global consistency.

*Keyframe-Based Asynchronous Optimization*— The system maintains a keyframe buffer that acts as a long-term memory, storing a history of processed frames. This is essential for preventing catastrophic forgetting, as the optimizer can periodically sample batches of past views to enforce global consistency. The optimization schedule is decoupled from a continuous step counter, allowing refinement to be triggered based on elapsed training intervals. This accommodates the asynchronous and intermittent execution of the optimization thread, making practical use of idle compute time during rover operations. While the Gaussian parameters are the primary target of optimization, the framework could be extended to refine camera poses as well; however, this is considered outside the scope of the current work.

*Densification Strategy*—To manage the set of Gaussians, we modify the standard 3DGS densification strategy. The data structure for each Gaussian is augmented to include a semantic class identifier. The periodic, global reset of Gaussian opacities is removed; this is critical for long-term mapping as our optimization prioritizes recent keyframes, and a global reset could cause stable, older portions of the map to be incorrectly pruned, leading to catastrophic forgetting. Furthermore, our implementation omits the standard splitting and cloning operations. Since the backend incrementally adds geometry from dense depth maps with each new keyframe, the large, under-reconstructed regions that typically necessitate splitting are less prevalent. Instead, we manage map complexity by pruning Gaussians that are inconsistent with a known prior DEM model.

*Loss Function*—The optimization process minimizes a total loss function, $\mathcal{L}_{\text{total}}$, which extends the standard 3DGS objective ($\mathcal{L}_{\text{recon}} + \mathcal{L}_{\text{scale}}$). The unique characteristics of lunar environments—namely high-contrast lighting and deep, hard-edged shadows with no atmospheric scattering—make relying on a simple background color insufficient. A deep shadow on the regolith can be photometrically indistinguishable from the black sky, providing ambiguous signals to the optimizer. To address this, we introduce explicit geometric and semantic supervision through additional loss components that use masks derived from sensor data.

These losses are applied selectively using two masks. The surface mask, $M_{\text{surface}}$, identifies pixels corresponding to valid geometry. A pixel is included in this mask only if it is not classified as sky, its estimated depth value is finite, and, very importantly, it is within a pre-specified depth range for optimization. In this sense, a pixel that corresponds to a rock that is far from the current camera pose is not included in the surface mask. The empty space mask, $M_{\text{empty}}$, identifies pixels known to contain no geometry and is primarily composed of pixels identified as sky.

Our additional loss components are:

• **Supervised Depth Loss**: To leverage the dense depth maps from our perception frontend, we add a supervised depth loss, $\mathcal{L}_{\text{depth}}$. This term enforces geometric accuracy by penalizing the L1 difference between the rendered depth, $\hat{D}$, and the input depth map, $D$, only on the $M_{\text{surface}}$ mask:

$$\mathcal{L}_{\text{depth}} = \frac{1}{|M_{\text{surface}}|} \sum_{p \in M_{\text{surface}}} \frac{|D(p) - \hat{D}(p)|}{\max_{q \in M_{\text{surface}}} D(q)} \tag{6}$$

• **Volumetric Regularization Losses**: To regularize the density distribution and remove artifacts, we use two losses based on the rendered alpha accumulation, $\hat{\alpha}$. An *empty loss*, $\mathcal{L}_{\text{empty}}$, penalizes density in regions defined by $M_{\text{empty}}$. A complementary *full loss*, $\mathcal{L}_{\text{surface}}$, encourages surfaces within $M_{\text{surface}}$ to be opaque:

$$\mathcal{L}_{\text{empty}} = \frac{1}{|M_{\text{empty}}|} \sum_{p \in M_{\text{empty}}} \hat{\alpha}(p) \tag{7}$$

$$\mathcal{L}_{\text{surface}} = \frac{1}{|M_{\text{surface}}|} \sum_{p \in M_{\text{surface}}} (1 - \hat{\alpha}(p)) \tag{8}$$

The final objective function is a weighted summation of the baseline and our new components:

$$\begin{aligned} \mathcal{L}_{\text{total}} = \mathcal{L}_{\text{recon}} + \lambda_{\text{scale}}\mathcal{L}_{\text{scale}} + \lambda_{\text{depth}}\mathcal{L}_{\text{depth}} \\ + \lambda_{\text{empty}}\mathcal{L}_{\text{empty}} + \lambda_{\text{surface}}\mathcal{L}_{\text{surface}} \end{aligned} \tag{9}$$

where the $\lambda$ terms are the corresponding weighting coefficients.

## 4. RESULTS

In this section, we conduct experiments to benchmark semantic segmentation models and analyze their impact on 3D reconstruction quality. We quantitatively evaluate segmentation performance and qualitatively investigate how semantic information improves surface reconstruction accuracy and rock detection.

*Evaluation Metrics*

We evaluate the performance of our pipeline across three key tasks. The notation used here is consistent with that of our defined loss functions.

*Depth Estimation*—All depth estimation metrics are computed only over the valid pixels defined by the surface mask, $M_{\text{full}}$. For a rendered depth map $\hat{D}$ and a ground truth depth map $D$:

• **Mean Absolute Error (MAE) [m]** $\downarrow$: The average L1 distance between the rendered and ground truth depth.

$$\text{MAE} = \frac{1}{|M_{\text{full}}|} \sum_{p \in M_{\text{full}}} |\hat{D}(p) - D(p)| \tag{10}$$

• **Absolute Relative Error (AbsRel)** $\downarrow$: The scale-invariant average of the absolute relative difference.

$$\text{AbsRel} = \frac{1}{|M_{\text{full}}|} \sum_{p \in M_{\text{full}}} \frac{|\hat{D}(p) - D(p)|}{D(p)} \tag{11}$$

• **Threshold Accuracy** ($\delta_{25\%}$) $\uparrow$: The percentage of pixels where the ratio between the rendered and ground truth depth is within a factor of 25%.

$$\delta_{25\%} = \frac{1}{|M_{\text{full}}|} \sum_{p \in M_{\text{full}}} \mathbb{1}\left\{ \max\left( \frac{\hat{D}(p)}{D(p)}, \frac{D(p)}{\hat{D}(p)} \right) < 1.25 \right\} \tag{12}$$

• **Frames Per Second (FPS)** $\uparrow$: The processing throughput of the depth estimation model.

*Semantic Segmentation*—We compute the following metrics over all the pixels in the image for the semantic segmentation models:

• **Intersection over Union (mIoU)** $\uparrow$: The overlap between the predicted and ground truth masks for a single class or label.

$$\text{IoU} = \frac{\text{TP}}{\text{TP} + \text{FP} + \text{FN}} \tag{13}$$

• **Accuracy (Acc.)** $\uparrow$: The percentage of all pixels in the image that are correctly classified.

$$\text{Acc.} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{14}$$

*Surface Reconstruction*—To evaluate the final 3D map, we extract a point cloud, $\hat{P}$, by taking the means of the Gaussians in the dense representation and compare it to the ground truth point cloud, $P$. While a more accurate surface reconstruction could be achieved by leveraging the full density field of the Gaussians, this is beyond the scope of our current work. However, since our map is sufficiently dense, using the means of the Gaussians provides a good approximation for evaluation.

• **Accuracy (Chamfer-$L_2$) [cm]** $\downarrow$: The average distance from each point in the reconstructed point cloud to its nearest neighbor in the ground truth point cloud. This measures the correctness of the reconstructed surface.

$$\text{Accuracy} = \frac{1}{|\hat{P}|} \sum_{\hat{p} \in \hat{P}} \min_{p \in P} \|\hat{p} - p\|_2 \tag{15}$$

• **Completeness (Chamfer-$L_2$) [cm] ↓**: The average distance from each point in the ground truth to its nearest neighbor in the reconstruction. This measures how well the reconstruction covers the ground truth surface.

$$\text{Completeness} = \frac{1}{|P|} \sum_{p \in P} \min_{\hat{p} \in \hat{P}} \|p - \hat{p}\|_2 \tag{16}$$

• **Precision ($d$) [%] ↑**: The percentage of reconstructed points within a distance threshold $d$ of the ground truth, measuring correctness.

$$\text{Precision}(d) = \frac{1}{|\hat{P}|} \sum_{\hat{p} \in \hat{P}} \mathbb{1} \left\{ \min_{p \in P} \|p - \hat{p}\|_2 < d \right\} \tag{17}$$

• **Recall ($d$) [%] ↑**: The percentage of ground truth points that have a reconstructed point within a distance threshold $d$, measuring completeness.

$$\text{Recall}(d) = \frac{1}{|P|} \sum_{p \in P} \mathbb{1} \left\{ \min_{\hat{p} \in \hat{P}} \|p - \hat{p}\|_2 < d \right\} \tag{18}$$

• **F1-Score ($d$) [%] ↑**: The harmonic mean of Precision and Recall, providing a single metric that balances correctness and completeness.

$$F_1(d) = 2 \cdot \frac{\text{Precision}(d) \cdot \text{Recall}(d)}{\text{Precision}(d) + \text{Recall}(d)} \tag{19}$$

*Depth Estimation*

We evaluate the depth estimation models described in Section 2 in the LuSNAR dataset. Figure 7 shows the RGB input and the depth errors for different depth estimation models in logarithmic scale to qualitatively characterize the performance of the models. Traditional stereo methods like BM and SGBM perform well under favorable illumination conditions but struggle when the light source is behind the camera or when dealing with bright areas and shadows. Learning-based stereo models, particularly RAFT-Stereo, demonstrate superior performance with high depth completion rates and excellent accuracy at short distances. While monocular models lag behind even after proper scaling, achieving error rates comparable to previous models only at medium distances, they could still be valuable for surface reconstruction when combined with proper uncertainty quantification, as they provide significantly more information than RGB images alone.

Table 1 shows the quantitative results of the depth estimation models. Traditional stereo methods achieve low MAE and AbsRel by producing depth only in regions where they are confident, which is evident from their low $\delta_{25\%}$ values. Learning-based stereo models like RAFT-Stereo predict depth more densely, including at longer ranges, which increases MAE due to larger absolute errors at far distances. However, the AbsRel remains comparable, indicating that relative errors remain low across distances. These models also operate at significantly lower frame rates compared to traditional methods. Monocular models underperform across all metrics, even after scale correction, but remain potentially useful when stereo input is not available. Since they are trained on Earth-based datasets, their performance on this terrain requires further evaluation, particularly with respect to finetuning.

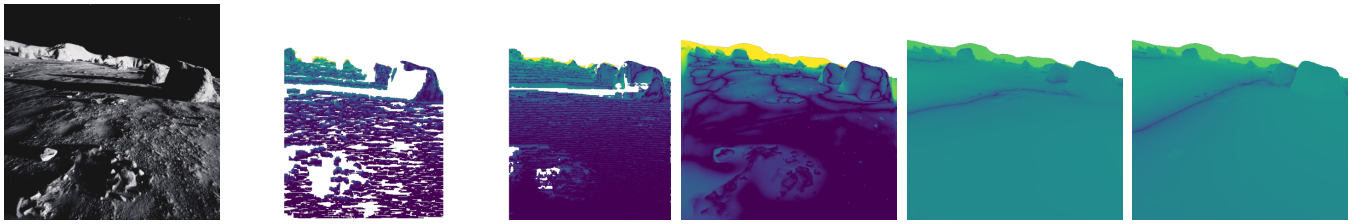
**Table 1**: **Comparison of depth estimation models. Monocular depth is scaled using triangulated points from matched features.**

| Model | Params | MAE↓ | AbsRel↓ | $\delta_{25\%}$↑ | FPS↑ |
|---|---|---|---|---|---|
| **Stereo** | | | | | |
| BM | – | 1.6 | 0.02 | 0.30 | 24.5 |
| SGBM | – | 1.6 | 0.02 | 0.41 | 14.9 |
| RAFT-Stereo | 11M | 7.3 | 0.05 | 0.72 | 4.5 |
| CREStereo | 5M | 3.5 | 0.04 | 0.73 | 1.4 |
| **Monocular** | | | | | |
| Depth Anything V2 | | | | | |
| Small | 25M | 11.1 | 0.21 | 0.53 | 13.2 |
| Base | 97M | 10.8 | 0.19 | 0.56 | 13.2 |
| Large | 335M | 10.7 | 0.18 | 0.59 | 9.8 |
| GLPN | 61M | 15.1 | 1.82 | 0.07 | 9.9 |
| DPT | 343M | 11.2 | 0.19 | 0.55 | 13.6 |
| Depth Pro | 952M | 11.4 | 0.20 | 0.55 | 1.7 |

Fig. 8 shows the performance of RAFT-Stereo. The model shows consistent performance with high accuracy in near regions (below 5 cm error). Precise depth estimates even in challenging lighting conditions and complex geometries.

*Semantic Segmentation*

We trained our semantic segmentation model on all available datasets, including LAC, Open3D, and LuSNAR. The model was trained to predict seven semantic classes: fiducial markers, rocks, lander, regolith, sky, mountains, and craters. Note that not all datasets contain all classes - for instance, the LAC dataset does not include craters or mountains. Here, we present the results specifically for the LuSNAR dataset, which does not contain landers or fiducial markers. The training was conducted on approximately 8,000 images, with



(a) **RGB input.** (b) **BM.** (c) **SGBM.** (d) **RAFT-Stereo.** (e) **Depth Anything.** (f) **DPT.**

**Figure 7**: **Input image and depth estimation errors for a sample of the LuSNAR dataset [12] . The errors are in logarithmic scale to qualitatively characterize the performance of the models.**

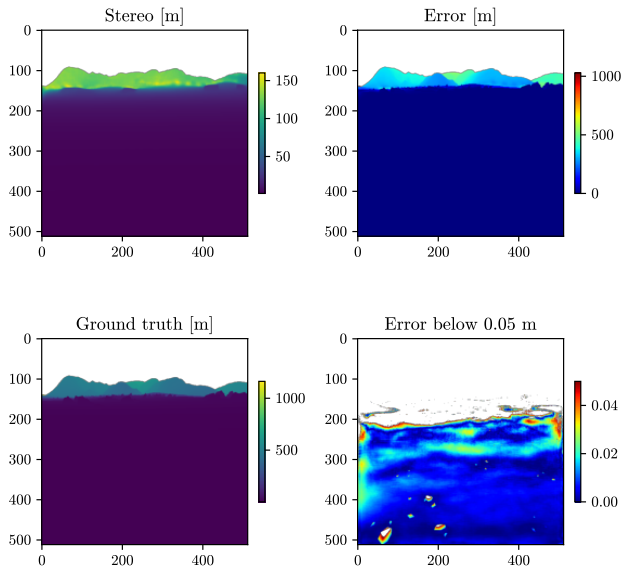



**Figure 8**: **LuSNAR dataset.**

1,000 images used for validation and 2,000 images from different scenes reserved for testing. The model was trained for 10 epochs using the Adam optimizer with an initial learning rate of $1 \times 10^{-3}$. We implemented a learning rate schedule with a reduction factor of 0.75, patience of 10 epochs, and a minimum learning rate of $1 \times 10^{-7}$. The batch size was set to 24 images.

Figure 9a shows the training losses for the different models, showing good convergence. Figure 10 shows the predictions of different models on the test set, which are able to successfully segment the craters, rocks, regolith, and sky. Table 2 shows the quantitative results for the different models on the LuSNAR dataset. We see that FPN performs best in crater detection, U-Net++ in rock detection, and LinkNet shows good overall. PSPNet is outperfoms the rest in terms of computational cost, measured in frames per second (FPS), which is highly desirable for aerospace applications where computational resources are limited. We found the U-Net++ performed well on the Open3D and LAC datasets, and was our choice perform the initial 3D reconstruction experiments.

*Surface Reconstruction*

We implemented our own real-time 3DGS reconstruction pipeline based on Splat-SLAM [34] and a point cloud mapping baseline. Our implementation uses ground truth pose information and prior terrain knowledge (without rocks or craters) to initialize the scene. For each new image, we use ground truth segmentation to remove sky regions and tag Gaussians with semantic information, compute depth using our depth estimation models, add new Gaussians to the scene based on RGB and depth information (computing their scale), and optimize the scene in the background using past images. The optimization objective includes L1 RGB and depth losses, penalties for Gaussian density above the surface, Structural Similarity Index Measure (SSIM) loss to ensure the reconstructed images maintain proper structural relationships and perceptual quality compared to the input images, and additional Gaussian regularization terms. Currently, we do not implement splitting and merging of Gaussians, but we remove Gaussians that are far from the prior terrain. **??** shows the steps of the pipeline using the custom Open3D environment.

**??** shows the 3DGS and point cloud map reconstructions on the Open3D environment, Table 3 shows the surface reconstruction and rock detection results, and **??** shows a top view of the 3DGS map. Gaussian splatting with RAFT-Stereo depth outperforms the point cloud baseline in reconstruction accuracy, likely due to its joint optimization over depth and RGB data across multiple views. This demonstrates the potential of neural scene representations for SLAM applications. When ground truth depth is used, the point cloud baseline performs slightly better. We attribute this to our current approach of reconstructing the mesh from the centers of the Gaussians rather than accounting for their density.

In both cases, most errors occur around rocks, which are often dark and hard to distinguish from the sky, making depth estimation challenging. Rock detection performance is comparable between methods, while the point cloud baseline yields lower height error—-possibly due to differences in how the mesh is extracted from the 3DGS map.

## 5. Conclusion

Our work demonstrates the effectiveness of combining semantic segmentation with geometric reconstruction for improved hazard detection in planetary navigation. Through extensive experimentation, we found that U-Net++ and LinkNet models achieve superior performance on lunar datasets, enabling accurate terrain classification crucial for surface exploration. The integration with 3D Gaussian Splatting (3DGS) not only improves reconstruction accuracy but also enables real-time SLAM and hazard detection capabilities. While our current results show promising improvements in rock detection accuracy, future work should focus on several key areas: incorporating semantic and depth uncertainty estimates to improve reliability, addressing map deformation after pose updates, developing path planning algorithms specifically designed for semantic 3DGS maps, and creating high-fidelity closed-loop simulations for comprehensive system validation. Additionally, evaluating additional test cases and exploring different depth detection models would help validate the generalizability of our findings across diverse terrain types and conditions.

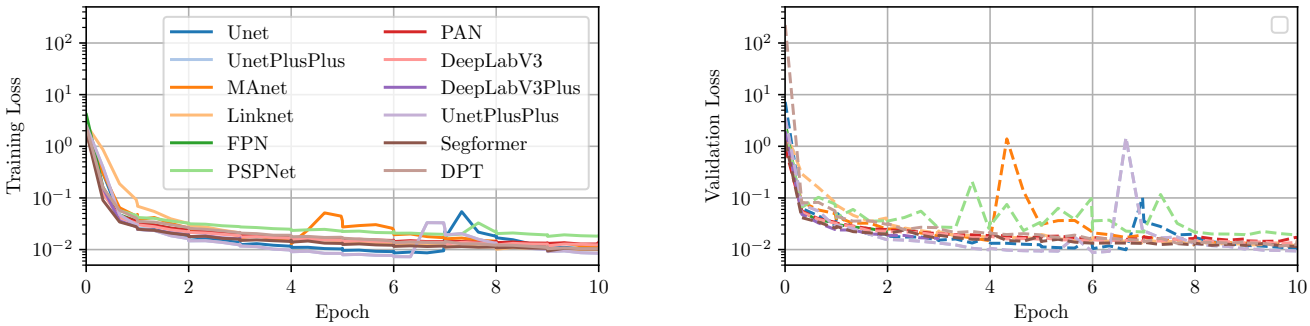
## Acknowledgements

We gratefully acknowledge Blue Origin for funding this project and for valuable discussions that contributed to its development. This manuscript benefited from the use of AI-based assistants, including Claude Sonnet 4.5 and Gemini 2.5 Pro, which were employed for coding assistance and revising. All final content was reviewed and edited by the authors.

## References

[1] F. Tosi, Y. Zhang, Z. Gong, E. Sandström, S. Mattoccia, M. R. Oswald, and M. Poggi, "How NeRFs and 3d gaussian splatting are reshaping SLAM: a survey."

[2] B. Kuang, C. Gu, Z. A. Rana, Y. Zhao, S. Sun, and S. G. Nnabuife, "Semantic terrain segmentation in the




(a) **Training losses.** (b) **Validation losses.**

**Figure 9**: **Training and validation losses.**

**Table 2**: **Semantic segmentation results on the LuSNAR dataset.**

| Model | Size [MB] | Params | FPS ↑ | IoU [%] ↑ | | | | | Mean IoU ↑ | Mean Accuracy ↑ |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Regolith | Crater | Rock | Mountain | Sky | | |
| U-Net | 55.0 | 14.3M | 114.4 | **99.5** | 95.5 | 89.2 | 96.5 | **99.8** | 96.1 | 99.5 |
| U-Net++ | 61.0 | 16.0M | 78.0 | **99.5** | 91.1 | **91.6** | **98.5** | **99.8** | 96.1 | **99.6** |
| MA-Net | 83.0 | 21.7M | 71.7 | 99.2 | 70.9 | 90.1 | 97.5 | **99.8** | 91.5 | 99.3 |
| Linknet | 44.0 | 11.7M | 104.5 | 99.2 | 96.4 | 91.1 | 97.0 | **99.8** | **96.7** | 99.3 |
| FPN | 50.0 | 13.0M | 108.7 | 98.9 | **97.7** | 86.8 | 97.8 | 99.7 | 96.2 | 99.2 |
| PSPNet | **3.0** | **0.9M** | **203.2** | 98.8 | 92.7 | 79.7 | 94.3 | 99.6 | 93.0 | 99.0 |
| PAN | 43.0 | 11.4M | 92.8 | 99.0 | 94.6 | 84.3 | 97.4 | 99.7 | 95.0 | 99.2 |
| DeepLabV3 | 61.0 | 15.9M | 131.5 | 99.0 | 96.7 | 86.6 | 98.3 | 99.7 | 96.1 | 99.2 |
| DeepLabV3Plus | 47.0 | 12.3M | 118.8 | 99.2 | 96.4 | 89.2 | 98.0 | **99.8** | 96.5 | 99.4 |
| Segformer | 45.0 | 11.8M | 140.5 | 99.2 | 95.5 | 88.9 | 97.7 | **99.8** | 96.2 | 99.4 |
| DPT | 159.0 | 41.6M | 79.6 | 99.1 | 94.1 | 87.4 | 95.7 | 99.7 | 95.2 | 99.3 |

**Table 3**: **Surface reconstruction results. TO BE UPDATED**

| Metric | Accuracy [cm] ↓ | Completion [cm] ↓ | Precision [%] ↑ | Recall [%] ↑ | F-Score [%] ↑ | Height Error [cm] ↓ |
|---|---|---|---|---|---|---|
| **3DGS** | | | | | | |
| Rock | 61.3 | 27.0 | 13.7 | 26.7 | 18.1 | 15.4 |
| Regolith | 12.9 | 28.0 | 60.0 | 66.8 | 63.2 | 8.2 |
| Crater | 941.1 | 27.0 | 9.9 | 15.3 | 12.0 | 74.6 |
| All | 17.5 | 25.8 | 53.2 | 60.7 | 56.7 | 10.4 |
| **+ Ground Truth Depth and Segmentation** | | | | | | |
| Rock | 61.3 | 27.0 | 13.7 | 26.7 | 18.1 | 15.4 |
| Regolith | 12.9 | 28.0 | 60.0 | 66.8 | 63.2 | 8.2 |
| Crater | 941.1 | 27.0 | 9.9 | 15.3 | 12.0 | 74.6 |
| All | 17.5 | 25.8 | 53.2 | 60.7 | 56.7 | 10.4 |
| **Point Cloud** | | | | | | |
| Rock | 61.3 | 27.0 | 13.7 | 26.7 | 18.1 | 15.4 |
| Regolith | 12.9 | 28.0 | 60.0 | 66.8 | 63.2 | 8.2 |
| Crater | 941.1 | 27.0 | 9.9 | 15.3 | 12.0 | 74.6 |
| All | 17.5 | 25.8 | 53.2 | 60.7 | 56.7 | 10.4 |
| **+ Ground Truth Depth and Segmentation** | | | | | | |
| Rock | 61.3 | 27.0 | 13.7 | 26.7 | 18.1 | 15.4 |
| Regolith | 12.9 | 28.0 | 60.0 | 66.8 | 63.2 | 8.2 |
| Crater | 941.1 | 27.0 | 9.9 | 15.3 | 12.0 | 74.6 |
| All | 17.5 | 25.8 | 53.2 | 60.7 | 56.7 | 10.4 |

navigation vision of planetary rovers—a systematic literature review," vol. 22, no. 21, p. 8393, publisher:

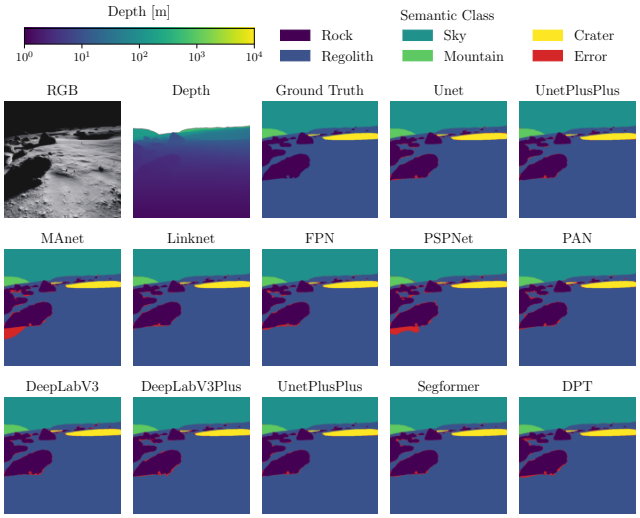



**Figure 10**: **Predictions.**

**Table 4**: **Memory usage for 3DGS.**

| **GPU VRAM** | 1017 MB |
|---|---|
| **3DGS** | 793 MB |
| Means | 170 MB |
| Scales | 170 MB |
| Rotations | 227 MB |
| Opacities | 56 MB |
| **Strategy** | 224 MB |
| Radii | 56 MB |
| Labels | 56 MB |
| Gradients | 56 MB |
| Counts | 56 MB |

| **System RAM** | 1900 MB |
|---|---|
| RGBs | 1200 MB |
| Depths | 400 MB |
| Masks | 200 MB |
| Labels | 100 MB |


Multidisciplinary Digital Publishing Institute.

[3] Y. Jia, L. Liu, and C. Zhang, "Moon impact crater detection using nested attention mechanism based UNet++," vol. 9, pp. 44 107–44 116.

[4] E. Van Kints, A. Hammond, C. Adams, and I. G. Lopez-Francos, "Neural radiance methods for lunar terrain modeling," in *2025 IEEE Aerospace Conference*, pp. 1–17, ISSN: 2996-2358.

[5] C. Adams, I. Lopez-Francos, E. V. Kints, and A. Hammond, "A summary of neural radiance fields for shadow removal and relighting of satellite imagery."

[6] R. Huang, C. Liu, H. Xie, J. Yu, T. Tao, Y. Xu, Z. Ye, and X. Tong, "Monocular visual SLAM with adjusting neural radiance fields for 3-d reconstruction in planetary environments," vol. 63, pp. 1–19.

[7] M. Hansen, C. Adams, T. Fong, and D. Wettergreen, "Analyzing the effectiveness of neural radiance fields for geometric modeling of lunar terrain," in *2024 IEEE Aerospace Conference*, pp. 1–12, ISSN: 1095-323X.

[8] A. Dai, S. Gupta, and G. Gao, "Neural radiance maps for extraterrestrial navigation and path planning," pp. 1606–1620, ISSN: 2331-5954.

[9] X. Zhang, L. Cui, and J. Yin, "Neural radiance fields for unbounded lunar surface scene," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 16 858–16 864.

[10] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "NeRF: representing scenes as neural radiance fields for view synthesis," vol. 65, no. 1, pp. 99–106.

[11] NASA. Apollo lunar surface journal.

[12] J. Liu, Q. Zhang, X. Wan, S. Zhang, Y. Tian, H. Han, Y. Zhao, B. Liu, Z. Zhao, and X. Luo. LuSNAR:a lunar segmentation, navigation and reconstruction dataset based on muti-sensor for autonomous exploration.

[13] H. Hirschmuller, "Stereo processing by semiglobal matching and mutual information," vol. 30, no. 2, pp. 328–341.

[14] L. Lipson, Z. Teed, and J. Deng, "RAFT-stereo: Multilevel recurrent field transforms for stereo matching," in *2021 International Conference on 3D Vision (3DV)*, pp. 218–227, ISSN: 2475-7888.

[15] J. Li, P. Wang, P. Xiong, T. Cai, Z. Yan, L. Yang, J. Liu, H. Fan, and S. Liu, "Practical stereo matching via cascaded recurrent network with adaptive correlation."

[16] D. Kim, W. Ka, P. Ahn, D. Joo, S. Chun, and J. Kim, "Global-local path networks for monocular depth estimation with vertical CutDepth."

[17] R. Ranftl, A. Bochkovskiy, and V. Koltun, "Vision transformers for dense prediction."

[18] A. Bochkovskii, A. Delaunoy, H. Germain, M. Santos, Y. Zhou, S. R. Richter, and V. Koltun, "Depth pro: Sharp monocular metric depth in less than a second."

[19] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation."

[20] Z. Zhou, M. M. R. Siddiquee, N. Tajbakhsh, and J. Liang, "UNet++: A nested u-net architecture for medical image segmentation."

[21] T. Fan, G. Wang, Y. Li, and H. Wang, "MA-net: A multi-scale attention network for liver and tumor segmentation," vol. 8, pp. 179 656–179 665.

[22] A. Chaurasia and E. Culurciello, "LinkNet: Exploiting encoder representations for efficient semantic segmentation," in *2017 IEEE Visual Communications and Image Processing (VCIP)*, pp. 1–4.

[23] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," pp. 2117–2125.

[24] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network."

[25] H. Li, P. Xiong, J. An, and L. Wang, "Pyramid attention network for semantic segmentation."

[26] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation."

[27] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation."

[28] T. Xiao, Y. Liu, B. Zhou, Y. Jiang, and J. Sun, "Unified perceptual parsing for scene understanding."

[29] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "SegFormer: Simple and efficient design for semantic segmentation with transformers."

[30] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis,

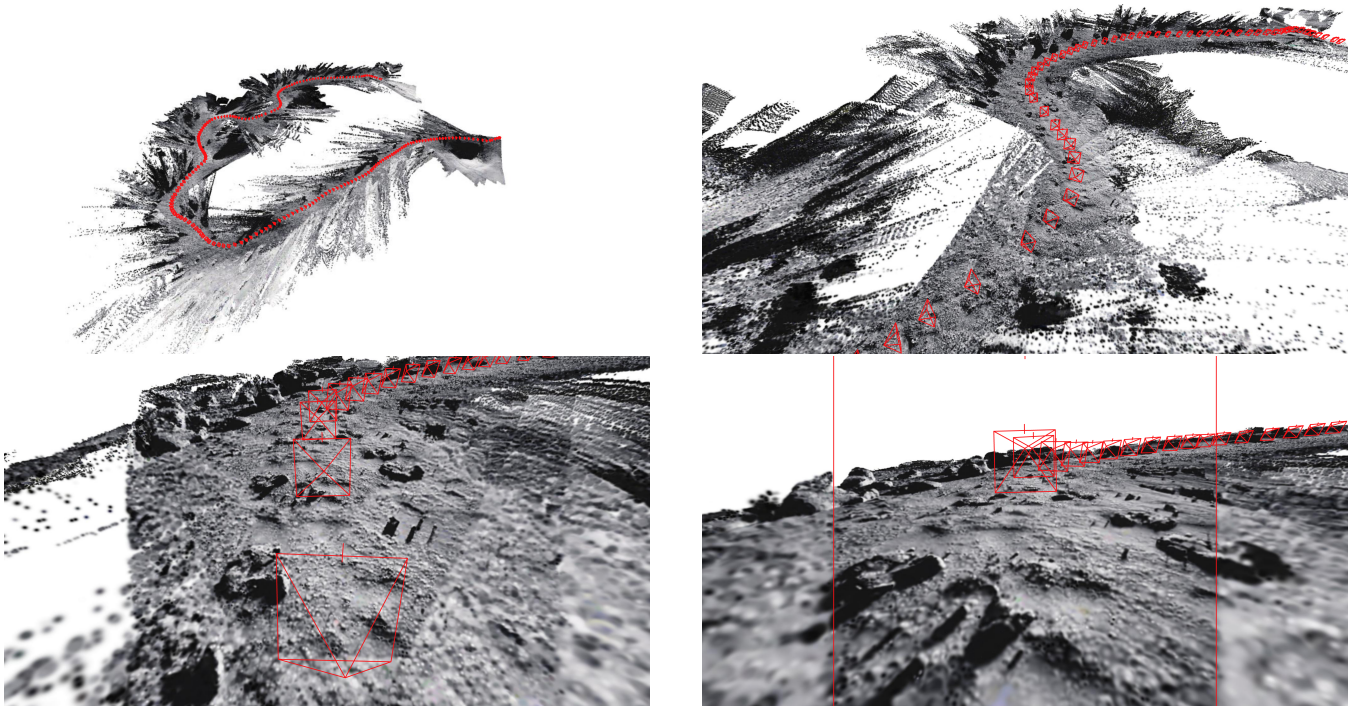


**Figure 11**: **Qualitative results of the real-time 3DGS reconstruction.**


"3d gaussian splatting for real-time radiance field rendering." vol. 42, no. 4, pp. 139–1.

[31] A. Dalal, D. Hagen, K. G. Robbersmyr, and K. M. Knausgård, "Gaussian splatting: 3d reconstruction and novel view synthesis: A review," vol. 12, pp. 96 797–96 820.

[32] D. DeTone, T. Malisiewicz, and A. Rabinovich, "SuperPoint: Self-supervised interest point detection and description."

[33] P.-E. Sarlin, D. DeTone, T. Malisiewicz, and A. Rabinovich, "SuperGlue: Learning feature matching with graph neural networks."

[34] E. Sandström, K. Tateno, M. Oechsle, M. Niemeyer, L. V. Gool, M. R. Oswald, and F. Tombari, "Splat-SLAM: Globally optimized RGB-only SLAM with 3d gaussians."


## BIOGRAPHY

***Guillem Casadesus Vila*** is a Ph.D. candidate in the Department of Aeronautics and Astronautics at Stanford University. He received his B.Sc. degree in Aerospace and Telecommunications Engineering in 2022 from the Universitat Politecnica de Catalunya (UPC) under the CFIS program. His research interests include robotic space exploration and space navigation and communication.

***Adam Dai*** is a Ph.D. candidate in the Department of Electrical Engineering at Stanford University. He received his B.Sc. degree in Electrical Engineering with a minor in Computer Science in 2019 from the California Institute of Technology. His research interests include navigation, mapping, and planning in unstructured 3D environments.

***Grace Gao*** is an assistant professor in the Department of Aeronautics and Astronautics at Stanford University. Before joining Stanford University, she was an assistant professor at University of Illinois at Urbana-Champaign. She obtained her Ph.D. degree at Stanford University. Her research is on robust and secure positioning, navigation, and timing with applications to manned and unmanned aerial vehicles, autonomous driving cars, as well as space robotics.