RAS 598: Space Robotics and AI (Spring 2025)
May 7, p1~5
ASU ID: 1234000809

Semantic 3D Point Cloud Map for Lunar Surface

Kaito Namatame (1234000809)

Robotic and Autonomous Systems (Mechanical and Aerospace Engineering), Arizona State University

Abstract

I present a novel approach to Semantic SLAM that combines semantic segmentation with ORB-SLAM3 across a sequence of images. This system works on images streams from a stereo camera on the lunar surface simulation. I first train U-Net with VGG16 as an encoder and segment images frame into 6 categories (Rover, Ground, Rocks, Earth, Lander, and Fiducials). I then track camera pose and generate 3d point cloud using ORB-SLAM3 which is feature-based tracking and mapping. In order to generate semantic 3D point cloud, I combine 3D point cloud with segmented images. This method constructs a semantic 3D point cloud map. Experimental results demonstrate improved scene understanding and localization under the lunar surface.

1. Introduction

Simultaneous Localization and Mapping (SLAM) is a cornerstone of autonomous robotic systems, enabling robots to build maps of unknown environments while simultaneously determining their location. ORB-SLAM3 rely on geometric features to construct sparse or dense maps. However, this method lack semantic information, limiting their ability to interpret the environment beyond geometric structures. Semantic Segmentation is able to fill this gap by classifying each pixel in an image into object categories, providing rich contextual information for tasks such as scene understanding. Despite its strengths, semantic segmentation has notable downside which is that semantic segmentation itself does not provide 3D geometric information.

To overcome these challenges, I propose integrating semantic segmentation with ORB-SLAM3. This approach combines the strengths of both techniques: semantic segmentation enriches the SLAM map with object-level understanding, while ORB-SLAM3 ensures geometric consistency and real-time performance.

2. Methodology

2.1 Camera Model

The camera is modelled as perfect pinhole camera with square pixels and there is no distortion. The camera has the same field of view of 1.22 radians (70 degrees). The front facing stereo camera is located just above the rover chassis. The stereo baseline is 0.162 meters.

2.2 Segmentation with Deep Learning

The ground truth and data is obtained from left stereo camera and semantic camera attached to the rover about 9,000 images.

In this project, the neural network architecture is an encoder-decoder neural network based on the U-net framework that segments the image of lunar terrain into 6 categories. Every pixel has its own label of class. In this U-Net architecture, pre-trained VGG16 model was utilized as an encoder and frozen to utilize ImageNet weights. Using the same weights as ImageNet allows for faster training than initialized weights. An results of segmentation is illustrated in Fig. 1.

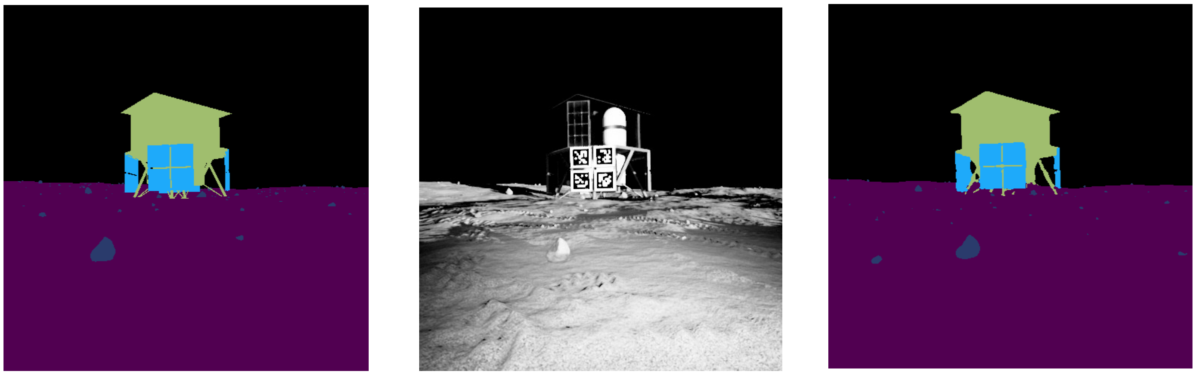

Fig. 1: Segmentation Results. From left, Ground truth, Raw image, and semantic mask

2.3 Main System pipeline

- Stereo images are processed by ORB-SLAM3 to estimate camera pose and generate a 3D point cloud.
- The left stereo image is segmented using U-Net model to produce a semantic mask.
- 3D points are projected onto the semantic mask using coordinate transformations.
- Semantic labels (RGB colors) are assigned to each 3D point based on the corresponding pixel’s class.

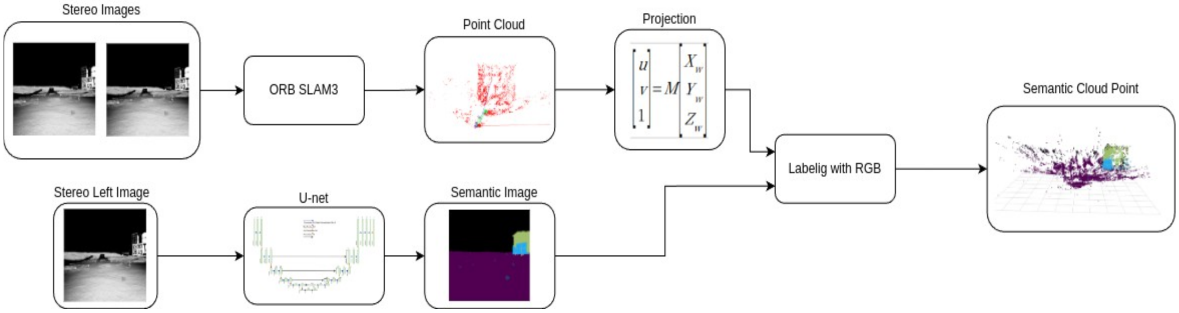

Fig. 2: Main pipeline for integrating orb slam3 and semantic segmentation

• Coordinate Transformation

To integrate semantic labels with 3d point cloud, points are projected onto the semantic images. Coordinate transformation from the world coordinate to the pixel coordinate is necessary. Two transformation is involved; world, camera, and pixel coordinate.

i. World to Camera Coordinates

$$X_c = T_{cw} \cdot X_w$$

- $X_w = \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$: World Coordinate
- $T_{cw} \in SE(3)$: Transformation Matrix
- $X_c = \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix}$: Camera Coordinate

ii. Camera to pixel coordinate

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \frac{1}{Z_c} \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix}$$

- u, v : Pixel Coordinate

- f_x, f_y : Focal Length
- c_x, c_y : Principle Point Offset

- **Label Extraction**

  The projected 2D coordinates are used to access the corresponding pixel in the semantic segmentation mask. If the projected point lies within the image boundaries, the semantic class ID for that pixel is retrieved. This class ID is then used to assign a specific RGB color to the 3D point based on a predefined color.

3. Results

In this project, the semantic point cloud was generated by projecting the point cloud obtained from ORB-SLAM3 onto semantic segmentation images produced by a U-Net architecture with a VGG16 encoder. Each point in the point cloud was labeled based on the corresponding semantic category in the image, resulting a semantically labeled point cloud. The point cloud has XYZ and RGB information, which represents Euclidean coordinate and the color data for each point. However, the semantic images provided by U-net architecture model run at only 4fps which is too slow for real-time performance.

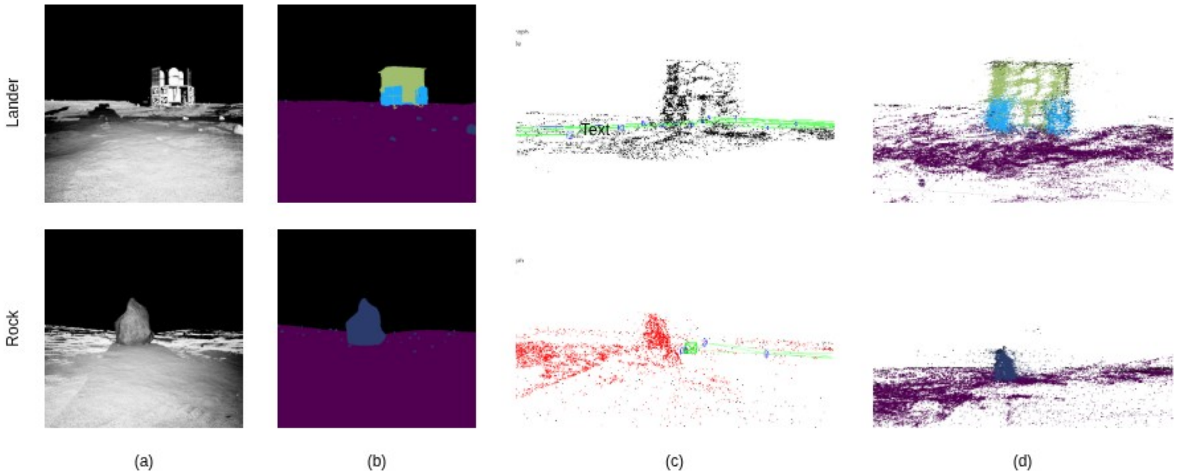

Fig. 3: (a) is a raw image and (b) is a corresponding semantic mask. (c) is a 3d point cloud results of ORB-SLAM3 and (d) is the results of semantic 3d point cloud.

4. Conclusion and Future Plan

In this project, I present a 3D semantic mapping in visual SLAM by combining semantic segmentation and ORB-SLAM3. Semantic 3d point cloud was successfully generated by combining semantic segmentation and ORB SLAM3.

However, there are several challenges.

- The U-Net architecture with VGG16 as the encoder is relatively slow, producing 2D semantic images at only 4 FPS. This issue can be addressed by replacing the encoder with a more efficient backbone, such as EfficientNet or MobileNet.
- The generated point cloud contains some noise. This can be addressed by applying filtering techniques to each ray. For example, sampling points along each ray and removing outliers could result in a cleaner and more accurate 3D point cloud.
- In this approach, I assign class labels to 3D points based on the semantic segmentation of each frame and visualize the resulting semantic point cloud in RViz2. However, this semantic point cloud is not updated when loop closure occurs in ORB-SLAM, and therefore it does not reflect the pose corrections resulting from loop closure.