RBE/CS549: Homework 2
LIDAR Semantics

Mihir Kulkarni
MS, Robotics
Worcester Polytechnic Institute
Email: mmkulkarni@wpi.edu

***Abstract*—This project is an implementation of ICP to build and stitch point cloud map. We also perform fusion of RGB data from the cameras and the generated Point cloud map in order to have a better understanding of the scene. Fusing the data from lidar and camera sensors can be useful for a variety of tasks, including localization, mapping, and object recognition. In the second part of the project, we perform semantic segmentation of RGB images using deep learning and project the segmented colors onto the Point Cloud map.**

I. Building the map using ICP

We use the KITTI 360 dataset for this project which contains raw LIDAR scans, RGB image information, and camera intrinsics as well as extrinsics. The LIDAR scans are in the format .bin. We have to first convert them into .pcd format in order to make use of Open3d functions for calculating transformations between point clouds and combining them. Once we get the point clouds in the .pcd format, we perform Point-to-Point Iterative closest point algorithm (ICP) to stitch them. ICP is an algorithm for aligning two point clouds by minimizing the sum of squared distances between the corresponding points in the two clouds.

In Point-to-Point ICP, the distance between the two points defined by a set of corresponding points is minimized.

To find the optimal transformation, the ICP algorithm iteratively performs the following steps:

Initialize the transformation between the two point clouds. This can be done randomly, or by using some prior information about the relative orientation of the two point clouds. For each point in the first point cloud, find the closest point in the second point cloud. Use the correspondences between the points in the two point clouds to compute a new transformation that aligns the point clouds. Iterate until the transformation converges or a maximum number of iterations is reached.

Repeat the above steps until the transformation converges (i.e., the sum of squared distances between the points in the two clouds stops decreasing).

I have used the Open3d point cloud library to perform the Point-to-Point ICP of 10 selective images from the KITTI 360 vision dataset. The input .pcd files are the converted ones from the .bin files as given in the dataset.

The output of the ICP is as follows:

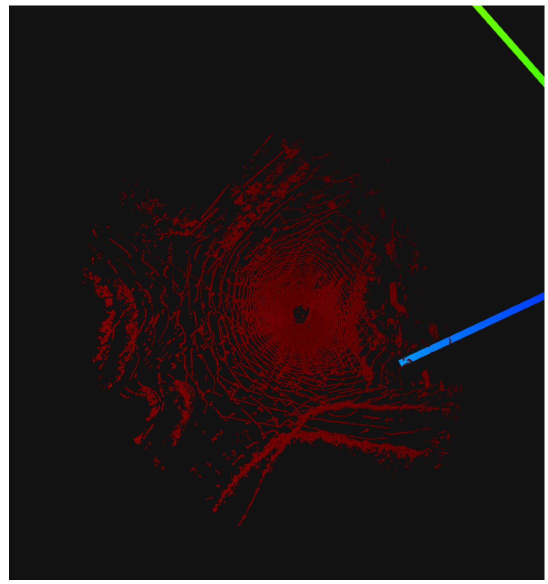

Fig. 1: Point-to-Point ICP

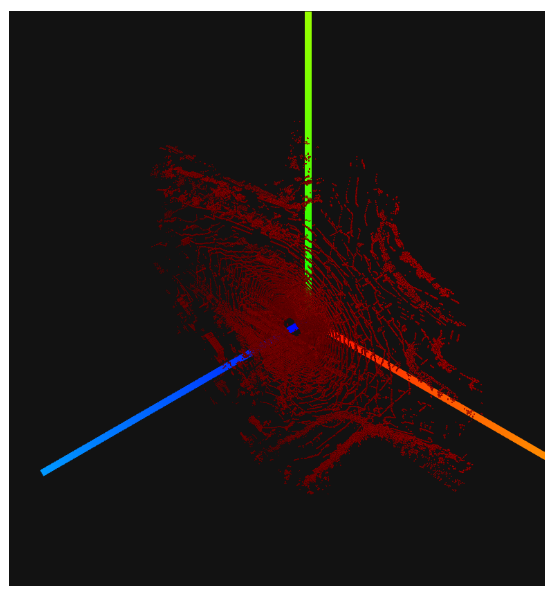

Fig. 2: Point-to-Point ICP

II. Semantic Segmentation of RGB images

Semantic segmentation involves assigning a semantic label to each pixel in an image. In the case of RGB (red-green-blue) images, the semantic label indicates the class or category to which the pixel belongs.

A. Network Architecture - BiSeNetv2

To perform Semantic segmentation on the same 10 images taken from the dataset, I have used the popular BiSeNetv2 model that works on the encoder-decoder network architecture. The input images to the network are RGB images of size (1242, 375).

The archietcture of BiSeNetv2 is shown in the figure below:

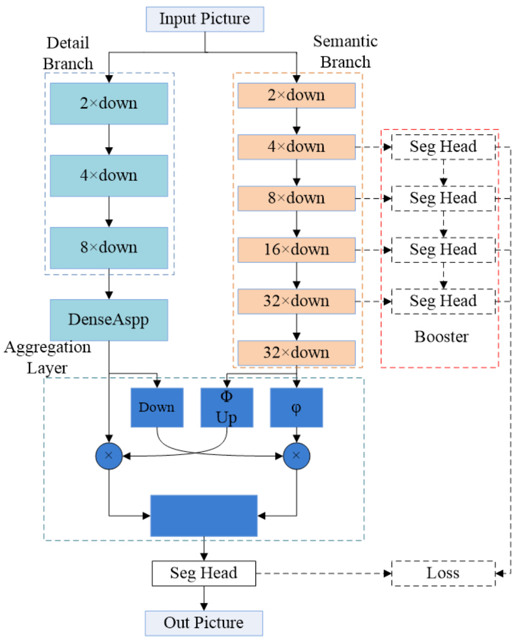

Fig. 3: BiSeNetv2 architecture

B. Projecting semantic labels on the PointCloud

The last part is to project these detected semantic labels on the PointCloud of the 10 images combined. This is also called as Point Painting. We first use the extrinsic parameters (i.e., the rotation and translation) to transform the 3D points computed from the ICP map into the coordinate frame of the RGB sensor. Then, we can use the intrinsic parameters of each sensor to project the transformed 3D points onto the image planes of the RGB and IR sensors.

It's worth noting that this process will only work if the point cloud and the RGB image have sufficient overlap and if the extrinsic and intrinsic parameters of the camera are accurately known. If there are any discrepancies in these parameters, the resulting point cloud may not align correctly with the image. For each pixel in the semantic segmented image, find the corresponding 3D point in the point cloud using the intrinsic parameters of the RGB camera. The intrinsic parameters describe the internal geometry of the camera, such as the focal length and principal point.

The final output after projecting the semantic labels on the point cloud map, we get the following:

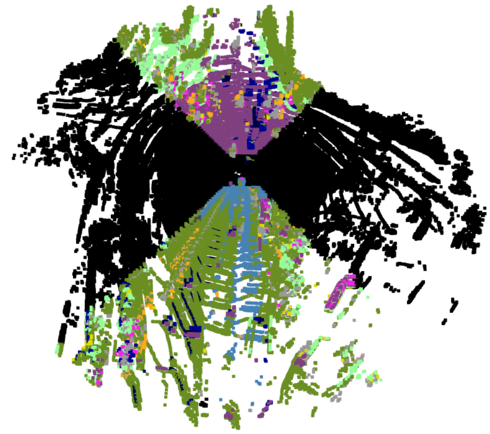

Fig. 4: Semantic Segmented Point Cloud

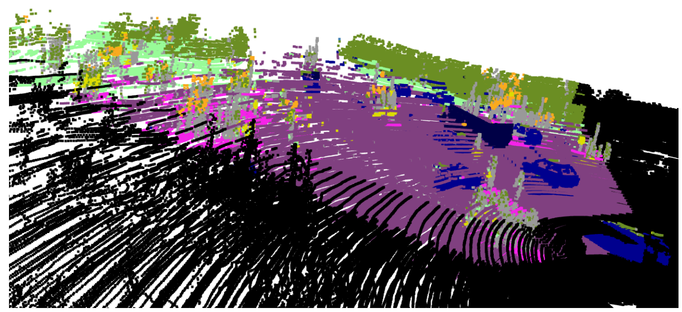

Fig. 5: Semantic Segmented Point Cloud

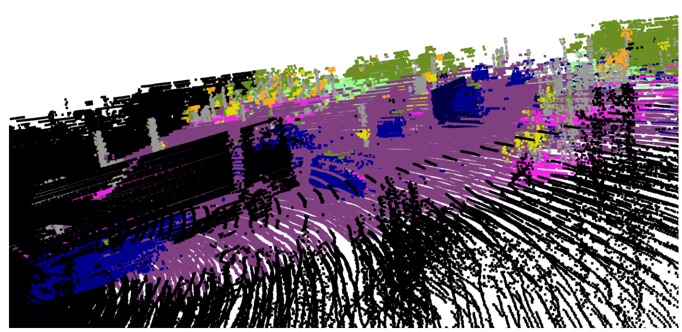

Fig. 6: Semantic Segmented Point Cloud

References

[1] https://github.com/AmrElsersy/PointPainting
[2] https://github.com/naitri/PointPainting

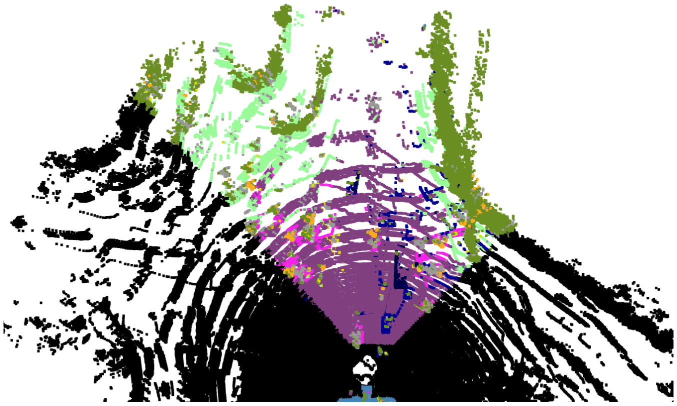

Fig. 7: Semantic Segmented Point Cloud

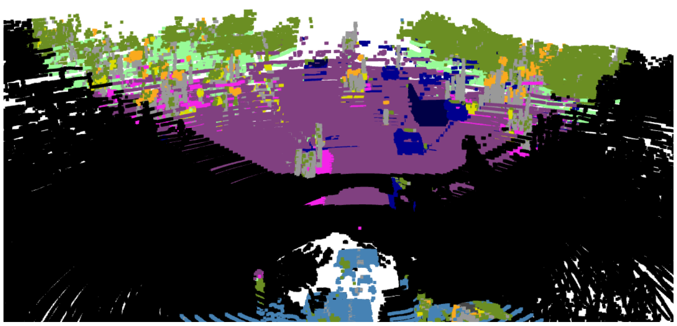

Fig. 8: Semantic Segmented Point Cloud

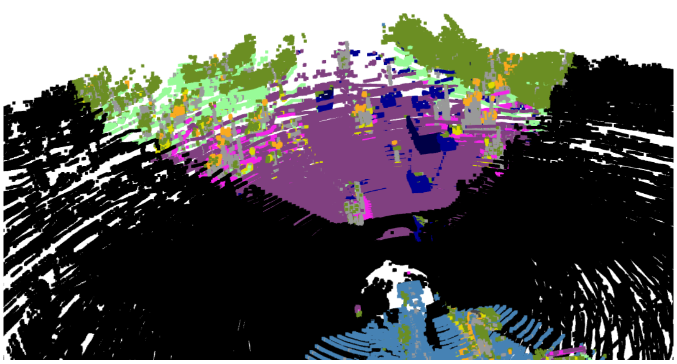

Fig. 9: Semantic Segmented Point Cloud

[3] https://www.cvlibs.net/datasets/kitti/eval$_{o}bject.php?obj_{b}enchmark = 3d$

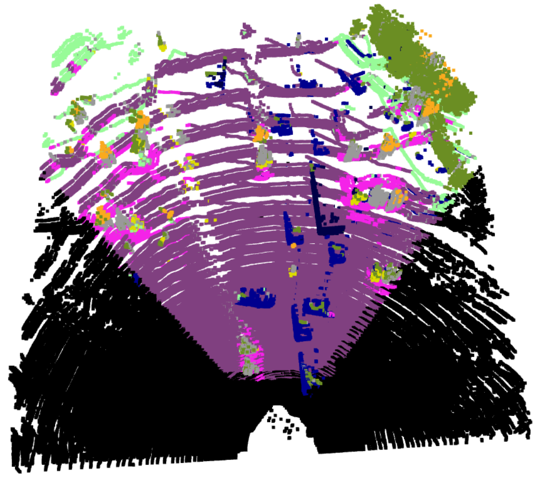

Fig. 10: Semantic Segmented Point Cloud

[4] https://chat.openai.com/chat