Semantic Segmentation

Aadhya Puttur

I. Introduction

The goal of this project is to build a map from raw LIDAR point cloud and then transfer the predicted semantic labels from the camera image onto the LIDAR point cloud. We will be using the KITTI-360 dataset as their dataset holds LIDAR scans and camera images.

II. Building the Map

The first step is building a LIDAR point cloud map. For this task we will be using the classical Point-to-Point ICP. Iterative closest point (ICP) is an algorithm employed to minimize the difference between two clouds of points, and widely used in localization of robot. In an example below, I have analyzed ICP working on a dataset that involved a robot scanning a room. The reason we use ICP is because when taking raw LIDAR data from different poses is very noisy because it is hard to assume the objects measured by the LIDAR will stay static. It collects drift quickly because of it, although we can fix it through a process called scan matching. The process is through taking a LiDAR scan and find the transformation that best aligns the new scan with either previous scans or some sort of abstracted map. Using the Iterative Closest Point (ICP) algorithm we will align the newest LIDAR scan with the previous scan.

The ICP algorithm involves 3 steps: association, transformation, and error evaluation. These are repeated until the scans are aligned. In this code for example, we were trying to align the two scans from the robot below 1. The cyan is the target scan or previous scan and the magenta is the the new scan. Our goal is to find the best transformation to align the source with the target.

We know the problem is we don't know point correspondence. Intuitively we would know what point corresponds to what. In code we will perform euclidean distance to all the nearest neighbor points and pick the closest one. We basically align the centroids and take the closest point in Euclidean. In robot odometry we need to measure the differences between the points to get R and T. Where there is some function f that is rigid alignment or SE(3) 2. We mean center and compute the rigid alignment between two point sets. Then we take the point sets and find the correlation using dot products which gives us rotation from taking the SVD.

A. Puttur is with the Department of Computer Science, Worcester Polytechnic Institute, Worcester, MA 01609, USA (e-mail: aputtur@wpi.edu)

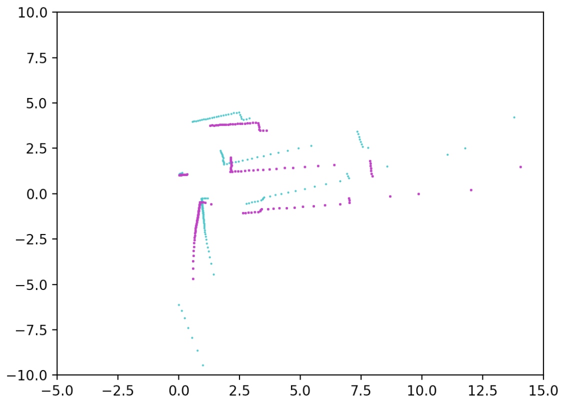


Fig. 1. Two scans that will be aligned with ICP

$H = \Sigma p_i \times q_i^T$

We check if the determinant is valid then we get translation $T = q - Rp$. We increment this process over and over again. We combine the previous rotation and translation with the new.

$$argmin_{[R,T]} \Sigma (Rp_i + T - q_i)^2$$

Fig. 2. For every single point P with a Rotation and translation Value

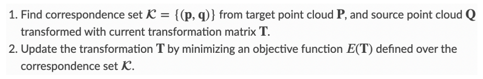

1. Find correspondence set $\mathcal{K} = \{(\mathbf{p}, \mathbf{q})\}$ from target point cloud $\mathbf{P}$, and source point cloud $\mathbf{Q}$ transformed with current transformation matrix $\mathbf{T}$.
2. Update the transformation $\mathbf{T}$ by minimizing an objective function $E(\mathbf{T})$ defined over the correspondence set $\mathcal{K}$.

Fig. 3. ICP Algorithm

With the algorithm 3, we align the scans 5.

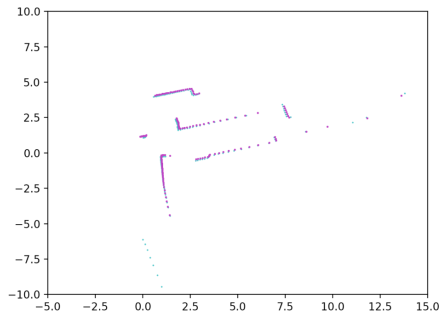


Fig. 4. Scans that are aligned using icp

Of course, I've realized there are better methods such as Doppler iterative closest point algorithm when the environment gets more complicated.

Now that the map is built we need to paint it.

III. Semantic Point Painting the Map

A point cloud is a set of 3D points $p_i \in R$, which can be created by different kinds of sensors, such as a lidar scannerA point cloud can have a RGB value for each point, which gives us a colored point cloud. To distinguish objects in a point cloud, a common method is semantic segmentationWe will be using RGB images to obtain semantic information. Specifically, we will be using semantic segmentation neural network to predict the semantic labels on every image frame. Then we will transfer this information on a map. We will do this by using the extrinsic between the sensors to transfer the labels from RGB image to LIDAR point cloud. We separate the segmentation from building the map so we can learn to implement both separately and be able to segment a full 3D LiDAR map. In point painting we are fusing semantic segmentation results based on RGB images and add class scores to the raw LiDAR point-cloud. In the first step we pass the images through a semantic segmentation network where we receive pixelwise segmentation scores. Then those scores with the corresponding points are projected back on the on the LiDAR cloud data. For the image based semantic segmentation we use mmsegmentation [1]. In

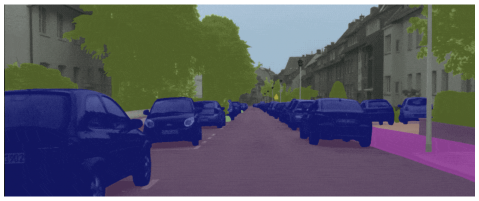


Fig. 5. Segmentation of one image frame using mmsegmentation

semantic segmentation of images, an image is divided into regions that are classified into one of the pre-defined classes. mmSegmentation uses convolution neural networks to pixel-wise label images. The Laplacian Pyramid Reconstruction and Refinement (LRR) network defines an architecture of a Laplacian reconstruction pyramid to fuse predictions from high resolution layers with low resolution layers [2]. First are the objects categorized in three large macro classes, foreground and parts. To recognize the different macro classes, different parts of high-level layers are used.The final segmentation result by fusing all the segmentation from the different levels, which finally outputs scores for each class for each pixel in the image.

Now that we can segment images frame by frame, we just need to be able to project them on the LiDAR point cloud data. We know that the pinhole camera is defined as $C = K[R,T]$ where R is the rotation matrix, T is the translation, and K is the intrinsic camera matrix. We design an operation to propagate external image labels to point clouds called "Search based Superpixel Labeling" [3]. We first use Mean Shift to segment individual images into "superpixels", and then propagate their labels onto the visually similar superpixels in the reference images of point clouds by using Exemplar SVM. We first directly find the k nearest neighbors in S. S is the external superpixel labeling pool consisting of superpixels with ground truth labels 6. S_q represents superpixels to be labeled in the reference images.

With this equation we can find the top k superpixels with the least euclidean distance D from S_q. For every superpixel

$$k\text{NN}(S_q) = \arg \min_{S_i \in \mathcal{S}}^{k} D(S_i, S_q)$$

Fig. 6. Find the most similar superpixels in given label

extracted from the labeling pool $S_i \in S$, we train a linear SVM to identify its visually similar super pixels **??**. To further guarantee the matching robustness, every superpixel S_i is translated and rotated to expand to more positive examples for training. To label the superpixel Sq in the reference images, wefind the superpixels with the k strongest responses as its k nearest neighbors in S_i 7.

$$k\text{NN}(S_q) = \arg \max_{S_i \in \mathcal{S}}^{k} F_i(\mathbf{w}_i^T \mathbf{x}(S_q) + b_i).$$

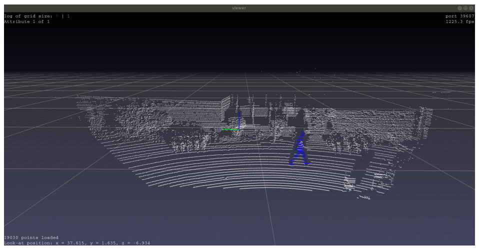
Fig. 7. label super pixels to their reference images

First the point cloud is segmented, producing a 3D segment set Di. Second, with the transform matrices Mi from local 2D coordinates in reference images $I_{i}{}_{i=1}^{R}$ to the global 3D coordinates, segments in Di are matched to the reference images, each resulting in a 2D region SMj (Di). If this projected region shares enough portion with some superpixel Si from this reference image, we connect an edge between Si and SMj (Di) 8.

```
Algorithm 2: Search based Label Propagation.
1 Input: Point cloud $\mathcal{P}$ with reference images $\mathcal{I}_R$, and
    superpixel propagation pool $\mathcal{S}$
2 Do over-segmentation on both $\mathcal{P}$ and $\mathcal{I}_R$
3 for Superpixel $S_q \in \mathcal{I}_R$ do
4     Use Equation 3 to retrieve $k$NN from $\mathcal{S}$
5 end
6 Use the over-segmentation structure of $\mathcal{P}$ and $\mathcal{I}_R$, and
    the $k$NN of $S_q$ to construct the graphical model as
    introduced in Section 3
7 Solve the graphical model by minimizing Equation 6
    with Loopy Belief Propagation
8 Output: 3D-segment-wise semantic labels of $\mathcal{P}$
```

Fig. 8. superpixel labeling pool

1) https://github.com/open-mmlab/mmsegmentation
2) https://liu.diva-portal.org/smash/get/diva2:1091059/FULLTEXT01.pd
3) $https : //www.ee.columbia.edu/ln/dvmm/publications/13/cvp$
4) $https : //github.com/vobecant/pix2pixHD_{rnd_{insertion.git}}$



```
log of grid size: 0 | 1
Attribute 1 of 1
port 39607
1225.3 fps
19030 points loaded
Look-at position: x = 37.615, y = 1.635, z = -6.934
```

Fig. 9. label 3D point cloud

5) $https://github.com/dtczhl/dtc-KITTI-For-Beginners$
6) $http://andrewjkramer.net/tag/slam/$

References