Lab2 - Line features with Split & Merge

Probabilistic Robotics

Songyou Peng
psy920710@gmail.com

March 13th , 2016

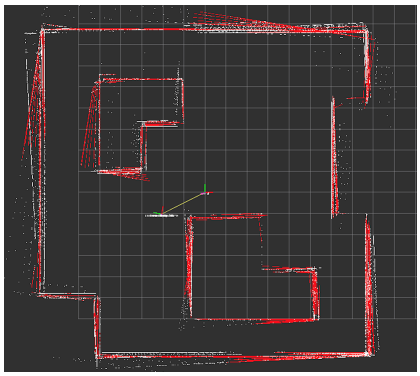
1 Introduction

In this report, we will briefly discuss the implementation of the split and merge algorithm in order to get the line features of the map. Also, some discussions will be focused on the comparisons of results with different parameters.

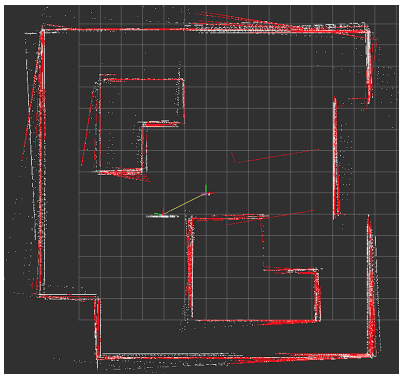
2 Implementation

For the implementation, we subscribe `LaserScan` messages from the topic `scan` and then project the information in the messages to point space. And then we use the point set as the input of the split and merge algorithm. In the algorithm, we acquire the first point (x_1, y_1) and the last point (x_2, y_2), and then we get the line $ax + by + c = 0$ through these two points with the parameters: $a = y_1 - y_2, b = x_2 - x_1, c = x_1 * y_2 - x_2 * y_1$. Then we find the maximal distance between the points in the point and the line. If the distance is larger than a threshold, we split the whole point set to two. If not, we will split the line as well when the distance between two consecutive points is larger than another threshold. This process should be done recursively until the number of points in a point set is small enough. This step mainly aims to distinguish the lines in the corner from normal lines. After the split step, some small line segments are acquired. We want the lines to be as continuous as possible, so the merge step is introduced.

As for merging, we simply calculate the angles of every two consecutive points and then compare the absolute difference of the angles with a threshold. We also check the distance between the last point of the first line and the first point of the second line. If both angle and dis difference values are really small, we merge the two lines. The result is shown in the Fig. 1(a).



(a) $dist\_thres = 0.12$ (default)



(b) $dist\_thres = 0.3$

(c) $split\_thres = 0.001$

Figure 1: Illustrations for the results with different parameters

3 Discussions

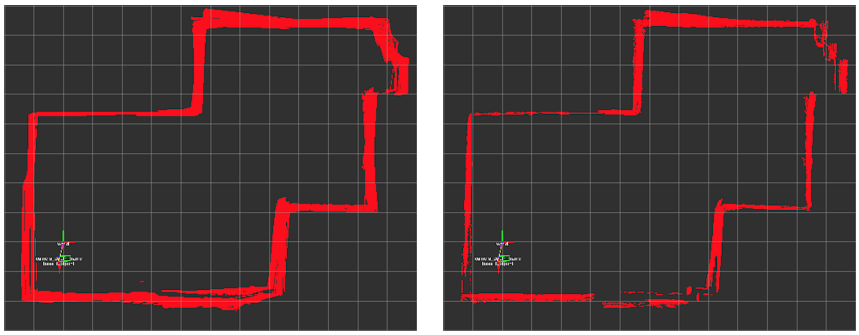
This part we will mainly talk about the influences of two main parameters: $dist\_thres$ and $split\_thres$.

First we analyse the `dataset2.bag`. Different $distance\_thres$ (0.05,0.12,0.3) almost makes no difference on the result. We can easily notice that, there are too many points in `dataset2.bag` and the map is very dense, so the distance between two points is really small. Therefore, in the merge step, the condition that distance between two consecutive points is smaller than $dist\_thres$ can be always satisfied, so the results have almost no differences.

We also try to change the $dist\_thres$ to a larger value for `dataset3.bag`, 0.3 in our case, shown in Fig.1(b). It turns out this change includes more wrong lines in the map. This may be because large $dist\_thres$ allows more lines to merge, which increases the incorrect merging rate. When we reduce this parameter

to 0.05, the result doesn't change at all. We can explain that, just like in the `dataset2.bag`, this parameter depends on the point set distributions. As the points in the map are relatively sparse, there is no more change after a certain threshold.

Besides, we try to tune *split_thres*. The default parameters lead to the good result shown in Fig. 2(a). However, a too small value may lead to too few line information (Fig. 2(b)).



(a) 0.1 (default) (b) 0.001

Figure 2: Illustrations for the results of `dataset2.bag` with various *split_thres*

Similarly, as for the `dataset3.bag`, only a few lines lie in the map if the *split_thres* is too small. Because the map is relatively sparse, more split case may happen, which makes the merging process more difficult. This lead to a problem that many details are discarded (Fig. 1(c)).

Figure 3: Bad case because of wrong angle difference

Also, we faced a problem for a long time. At the beginning, our result for the default parameters is as above. Too many oblique points are included in the

image. After checking the code, we found out that it was because when we calculate the angle difference in the merge step, we forgot to add an absolute sign for the angle difference. Therefore, some line segments' angles are actually really different (e.g. $-40°$) can still satisfy the merging condition. So so many oblique lines are included.

4 Conclusions

In this lab, the split and merge algorithm is implemented and tested in two point set bags. We also discuss two most important parameters for the split and merge step respectively, which proves that we should carefully and wisely tune the parameters for different maps.