TP 6 : Modelling

NPM3D - February 27th, 2020

mva.npm3d@gmail.com

Objectives

- Implement RANSAC algorithm and analyse its behaviour.
- Implement Region Growing algorithm and analyse its behaviour.

The report should be a pdf containing the answers to the **Questions** and named "TP6_LASTNAME_Firstname.pdf". Send your code along with the report in a zip file following the same naming rule.
Send your code along with the report to the email address above. The object of the mail must be "[MVA_NPM3D] TP6 LASTNAME Firstname". If you do the report as pair, the object of the mail must be "[MVA_NPM3D] TPX LASTNAME1 Firstname1 LASTNAME2 Firstname2".

A. RANdom SAmple Consensus

There are several ways to define a plane, here we will use a point and a normal for convenience.

1) In RANSAC.py write the function compute_plane that computes the plane passing through the three points.

2) Implement the function in_plane that takes as input a list of points, a plane, and a threshold value and returns the indices of the points whose distance to the plane are smaller than *threshold_in*.

The RANSAC algorithm follows a very simple concept of trial and errors. In our case, we want to use it to find the prominent plane in a point cloud. Each iteration consists of two simple steps:

- Randomly sample 3 points from the cloud. Compute the plane they define.
- Count how many points from the cloud are in range of this plane as votes.

The plane that has the most votes is kept as the prominent plane.

3) Write the function RANSAC returning the prominent plane in a point cloud, given a number of tries and a threshold distance.

Question 1 (1 point) : What does the prominent plane represent in the cloud? How many points does it count?

Question 2 (1 point) : How many tries do you need to get 99% chance of finding this plane.

We may want to extract more than one plane from a cloud. This can be achieved quite easily by applying RANSAC multiple times on the cloud. The points from the found planes just need to be removed between each new RANSAC call.

4) Write a function multi_RANSAC that apply RANSAC m times on a point cloud. Try with m=5 on indoor_scan.ply,

Question 3 (2 points) : Show a screenshot of the extracted planes. Are you satisfied with the extraction? Explain what produces this behaviour.

B. Region Growing

An alternative to RANSAC is to use a region growing algorithm. As a reminder, the steps of Region Growing algorithm are:

- Choose a seed that we put in a queue Q
- Instantiate a region R, containing the seed
- While the queue Q is not empty:
    - extract one point q of the queue Q,
    - find all its neighbors,
    - For each neighbor p:
        - If p, q and their normals verify some criterion:
            - add p to the region R,
            - If p verify some criterion (for exemple on its curvature):
                - add p to Q.

The crucial parts of this algorithm are the two criterions. In the following we will choose define these criterions with local descriptors : the normals and planarities of the points.

5) In RegionGrowing.py write a function compute_planarities_and_normals that computes for each point of the planarity and the normal with the radius *r*. You can use your code from TP4.

6) Write a function region_criterion(p1, p2, n1, n2) that returns *True* if two conditions are met:
   - distance from the point *p2* to the plane *(p1, n1)* is smaller than *threshold1.*
   - normals *n1* and *n2* form an angle smaller than *threshold2*.

   Tip: You can adapt the definition of verticality (TP4) for the second condition

7) Write a function queue_criterion that returns True if planarity *p* is bigger than a certain threshold.

8) Implement the function RegionGrowing that applies this algorithm to find a plane in a point cloud. Apply this function to indoor_scan.ply. Test different values of *r* and different thresholds in functions region_criterion and queue_criterion.

<u>Question 4 (2 points) :</u> How does the three thresholds affect the plane segmentation? What about the growing radius?

<u>Question 5 (1 points) :</u> Do you have any ideas to find a seed which increases the chances of finding a plane?

9) Change RegionGrowing so that the chosen seed has the best chance of defining a plane.

10) Write a function multi_RegionGrowing that apply the Region Growing algorithm *m* times on a point cloud, like multi_RANSAC.

<u>Question 6 (3 points) :</u> Show a screenshot of multiple planes extracted with this method. What are the advantages and drawbacks of region growing compared to RANSAC?