151228619/151248619: OBJECT ORIENTED PROGRAMMING I
2022-2023 FALL SEMESTERS
PROJECTS
Due: January 5, 2023, Thursday, 17:00

Remarks:

- Source code should be cleverly commented
- For the project you should write a report explaining design and implementation issues as well as anything a person may need to know while testing your program
- You are expected to make a good design and exploit object oriented programming principles (abstraction, inheritance, templates, exception handling, etc) as often as possible.
- For projects, you need to send the project workspace including your .cpp and .h files. In addition, you must submit your document. You must organize all this material in a single .zip or .rar archive, name the file as "yourID_Name_ProjectID.zip/rar", and load ESOGU UZEM.
- Grades will be given according to
    - 50% Class design and availability of object-oriented programming principles,
    - 15% code quality (commented and well formatted),
    - 15% documentation,
    - 20% functionality (implemented and correctly running functionalities).
- The application must be console-based.

PROJECT: Segmentation of a point cloud data

In this project, you will implement a class hierarchy to segment a point cloud data. In this project you will use Point Cloud Library (PCL) [1] to perform manipulations on a point cloud data. You have to implement RAndom SAmple Consensus (RANSAC) [2] and Region Growing [3] methods to segment point cloud data.

In the class design, you must have four classes: CommonProcesses (base class), Segmentation, RANSAC and RegionGrowing.

CommonProcesses class will be used to store raw point cloud data, viewer, and appropriate data members to read point cloud from a file, to write the point cloud to a file, to show the point cloud data on the viewer, to sample, scale, and rotation operations for point cloud.

Segmentation class is another base class for RANSAC and RegionGrowing classes. In this class, implement labeling of the segmented parts, colorized of the segmented parts, save the segmented parts individually. Besides, you can add more functionality to get extra credit.

RANSAC class inherits from Segmentation class and includes only functionalities for that specific segmentation method. Implement get and set functions for data members. For example, RANSAC must have a member function to return model parameter of plane.

RegionGrowing class inherits from Segmentation class and includes only functionalities for that specific segmentation method. Implement get and set functions for data members. For example, Region Growing must have a member function to return cluster size.

The segmentation results of four point clouds are given in Figure 1 for RANSAC and Region Growing methods. Segmented parts of these methods are colorized with different colors and a label is assigned to each color. For example, red represents to label 0, yellow represents to label 1, and etc.

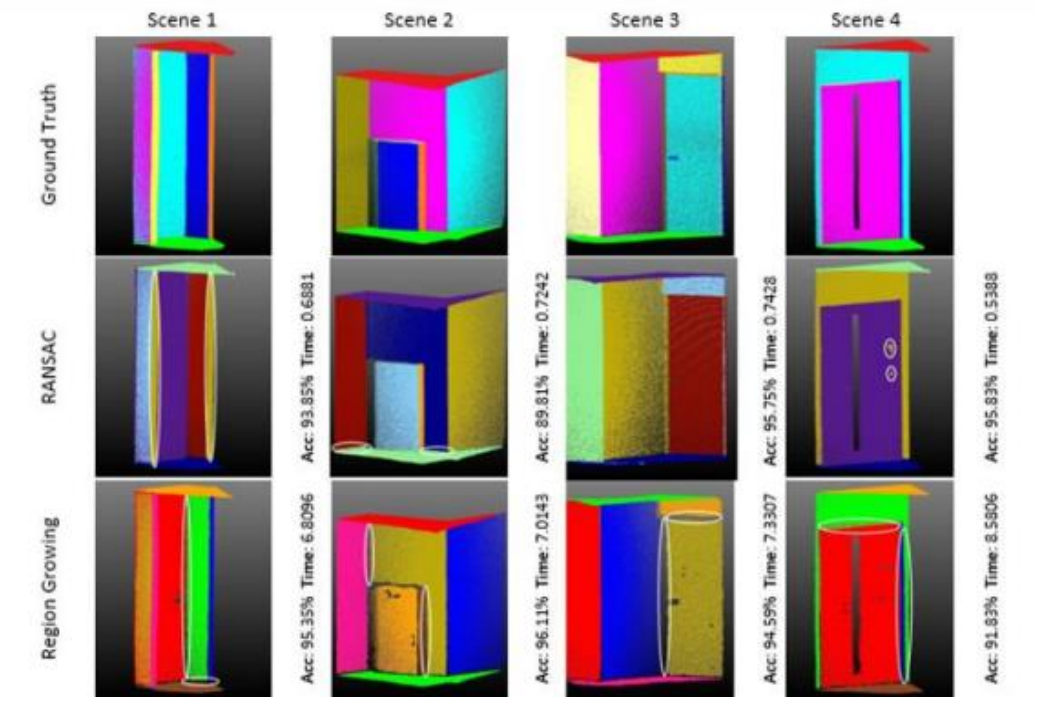

Fig 1. Example of segmented point clouds from our previous work [4].

Your design must meet the following requirements:

i. Data members of all classes must be private.
ii. Each class must contain constructor and destructor. Also, use member initializer and base class member initializer in appropriate situations.
iii. Validate data members if necessary. Use exception handling statements.
iv. Implement and use get and set functions for data members.
v. Implement appropriate access (read file and print functions) and utility functions.
vi. Each class must contain appropriate print functions or overloaded stream insertion operator (<<). Also, in appropriate situation you can implement operator overloading. This will get you extra credits.
vii. Use const qualifier, static storage class, and reference parameters in appropriate situations.
viii. Use dynamic memory allocation or use at least one STL container and one STL algorithm.
ix. Use composition and/or inheritance in appropriate situations.
x. The project is a console-based work but if you can provide graphical results such as Figure 1, you will get extra credits (20 pts).
xi. Draw UML diagram of your design and use doxygen for documentation of classes.

REFERENCES

[1] - https://pointclouds.org/

[2] - "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography", Martin A. Fischler and Robert C. Bolles, Comm. Of the ACM 24: 381–395, June 1981.

[3] - "Segmentation of point clouds using smoothness constraint", T. Rabbani, F. A. van den Heuvel, G. Vosselman. Environmental Science, Computer Science, 2006.

[4] - "A Comparative Study for Indoor Planar Surface Segmentation via 3D Laser Point Cloud Data". Eruyar, E. E., Yılmaz, M., Yılmaz, B., Akbulut, O., Turgut, K. & Kaleci, B. Black Sea Journal of Engineering and Science, 3(4), Pages 128-137 (2020).