SMJE 4383 ADVANCED PROGRAMMING ASSIGNMENT

CloudCompare / CloudComPy

Twu Le Sen A18MJ0158

Ng Yih Kai A18MJ0100

INTRODUCTION

CloudCompare is a 3D point cloud processing software (such as those obtained with a laser scanner). It can also handle triangular meshes and calibrated images.

Originally created during a collaboration between Telecom ParisTech and the R&D division of EDF, the CloudCompare project began in 2003 with the PhD of Daniel Girardeau-Montaut on Change detection on 3D geometric data.[2] At that time, its main purpose was to quickly detect changes in 3D high density point clouds acquired with laser scanners in industrial facilities (such as power plants) or building sites.[3] Afterwards it evolved towards a more general and advanced 3D data processing software. It is now an independent open source project and a free software.

CloudCompare provides a set of basic tools for manually editing and rendering 3D points clouds and triangular meshes. It also offers various advanced processing algorithms, among which methods for performing:

- projections (axis-based, cylinder or a cone unrolling, ...)
- registration (ICP, ...)
- distance computation (cloud-cloud or cloud-mesh the nearest neighbor distance, ...)
- statistics computation (spatial Chi-squared test, ...)
- segmentation (connected components labeling, front propagation based, ...)
- geometric features estimation (density, curvature, roughness, geological plane orientation, ...)

CloudCompare can handle unlimited scalar fields per point cloud on which various dedicated algorithms can be applied (smoothing, gradient evaluation, statistics, etc.). A dynamic color rendering system helps the user to visualize per-point scalar fields in an efficient way. Therefore, CloudCompare can also be used to visualize N-D data.

The user can interactively segment 3D entities (with a 2D polyline drawn on screen), interactively rotate/translate one or more entities relatively to the others, interactively pick single points or couples of points (to get the corresponding segment length) or triplets of points (to get the corresponding angle and plane normal). The latest version also supports the creation of 2D labels attached to points or rectangular areas annotations.

CloudCompare is available on Windows, Linux and Mac OS X platforms, for both 32 and 64 bits architectures. It is developed in C++ with Qt.

METHODOLOGY

First of all, we need to install anaconda in ubuntu which can refer to website (https://www.addictivetips.com/ubuntu-linux-tips/how-to-install-anaconda-on-ubuntu/)

Anaconda is a Python-based data science platform. It comes in various editions, is open source, and installable on most Linux operating systems.

To start the installation of Anaconda dependencies on Ubuntu PC, open up a terminal window. Once the terminal window is opened, use the apt install command below to get all of the dependencies set up on the system.

1. Run sudo apt install libgl1-mesa-glx libegl1-mesa libxrandr2 libxrandr2 libxss1 libxcursor1 libxcomposite1 libasound2 libxi6 libxtst6
2.

After getting all the dependencies taken care of, it is time to download the installation script. The script is hosted on Anaconda's repo site. Using the wget downloader command below, grab the install script.

2. Wget https://repo.anaconda.com/archive/Anaconda3-2020.11-Linux-x86_64.sh -O ~/Downloads/Anaconda3-2020.11-Linux-x86_64.sh

When the download process is complete, we need to enter the "Downloads" directory. This directory holds the installation script. To access "Downloads" via terminal, enter the following CD command.

3. cd ~/Downloads

Inside the “Downloads” directory, use the chmod command to update the installation script’s permissions. This script must have permissions changed so that it can execute as a program.

4. sudo chmod +x Anaconda3-2020.11-Linux-x86_64.sh

The installation of Anaconda on Ubuntu can begin as the installation script is downloaded and the permissions are set.

To start the installation of Anaconda on Ubuntu as a user account (non-root), run the installation without the sudo command. We highly recommend installing the app in this way.

5. ./Anaconda3-2020.11-Linux-x86_64.sh

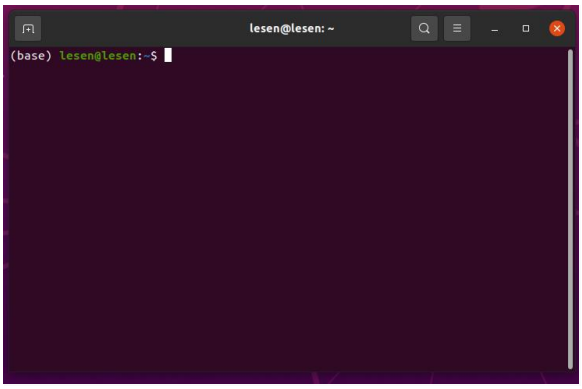
Once the script is started up, we press the Enter key to continue to the EULA.

After pressing the Enter key, we will be required to view the EULA. To get through it, press the Enter key keyboard. Once we have read the agreement, type out “yes” in the prompt.

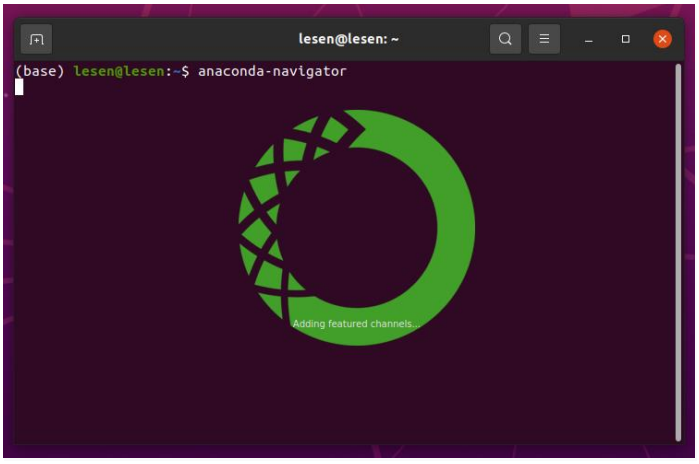
When “yes” is typed into the prompt, the Anaconda installer will default select the home directory to install the software.

Once we press the Enter key, the Anaconda installation script will begin installing the app to our system. It will install both the terminal application (conda) as well as the GUI application (Anaconda Navigator).

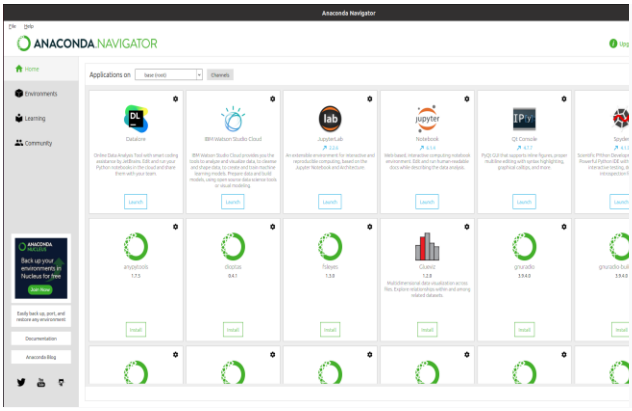
After installation done, it will have a (base) inside the terminal as the picture below



While we type a command (anaconda-navigator) in the terminal, it is not an error and prepare to run the command then this means the anaconda software is successfully installed in ubuntu.



After few time loading, the anaconda navigator will be opened as the picture below

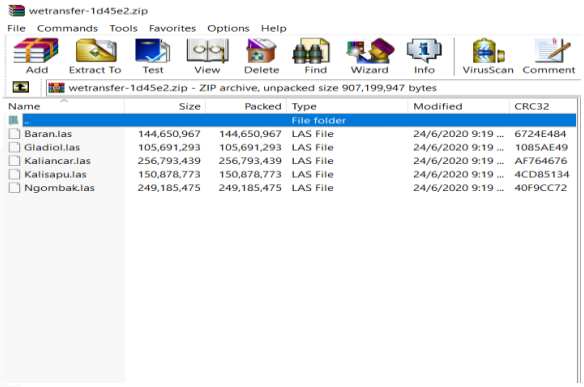


But we will not use an anaconda navigator to do this assignment, so we can just directly close this application.

Secondly, we download the data set files from google drive.

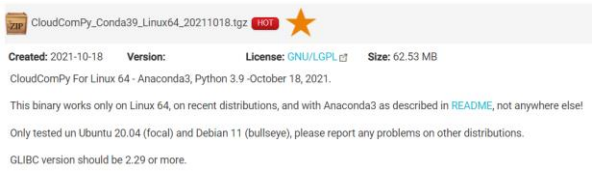
(https://drive.google.com/open?id=1EU5Q7U0j2jOzvSLAAELrd9BH0ZPar-Sl&authuser=zool%40utm.my&usp=drive_fs)

After completely downloading the dataset file, we will get a zip file and it will contain 6 las files inside.



Then the last source needed to install is to download CloudCompy binary from the resources website and select the .tgz file (which can install in ubuntu).

(https://www.simulation.openfields.fr/index.php/download-binaries)



After installation of all needed software, we can start testing a CloudComPy binary on linux with anaconda 3.

Before we run the command to do the testing, we need to check the GLIBC version first and should be 2.29 or above.

1. Run ldd --version

After checking the GLIBC version, we need to create an environment for CloudComPy in Anaconda3, from the terminal

2. Run . ~/anaconda3/etc/profile.d/conda.sh

3. Run conda activate

4. Run conda create --name CloudComPy39 python=3.9

5. Run conda activate CloudComPy39

6. Run conda config --add channels conda-forge

7. Run conda config --set channel_priority strict

8. Run conda install qt numpy psutil boost xerces-c pcl gdal cgal cmake pdal opencv ffmpeg mysql "qhull=2019.1" matplotlib "eigen=3.3.9" tbb openmp

Before using CloudCompare or CloudComPy, we load the environment. From a new prompt (replace by its value):

9. Extract the CloudCloudCompy binary files

10. Copy 4 files which is inside the installCloudcompare file to /anaconda3/envs/CloudComPy39

11. Run . ~/anaconda3/etc/profile.d/conda.sh

12. Run conda activate CloudComPy39

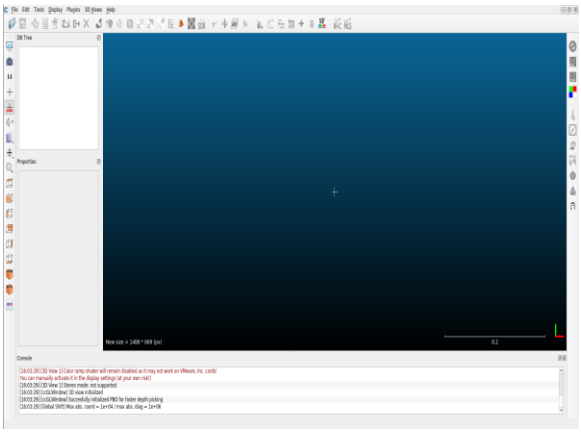
13. Run export LD_LIBRARY_PATH=~/anaconda3/envs/CloudComPy39/lib:${LD_LIBRARY_PATH}

14. Run export LD_LIBRARY_PATH=~/anaconda3/envs/CloudComPy39/lib/cloudcompare:${LD_LIBRARY_PATH}

15. Run export LD_LIBRARY_PATH=~/anaconda3/envs/CloudComPy39/lib/cloudcompare/plugins:${LD_LIBRARY_PATH}

Then we type the command below to call out the CloudComapare software

16. Run ~/anaconda3/envs/CloudComPy39/bin/CloudCompare



Moreover, from the prompt we can test the cloudcompare

17. Run cd ~/anaconda3/envs/ CloudComPy39/doc/PythonAPI_test

Python test

18. Run ctest

To execute all the test

19. Run . envPyCC.sh

To complete the pythonpath d run a script

20. Run python test001.py

Furthermore, we can use cloudcompare to do some animation on the dataset. So we also need to install cloudcompare with some command first.

1. Snap install core
2. Sudo snap install cloudcompare

After we install the cloudcompare, we use some command to open pointcloud viewer and the main software to do the animation.

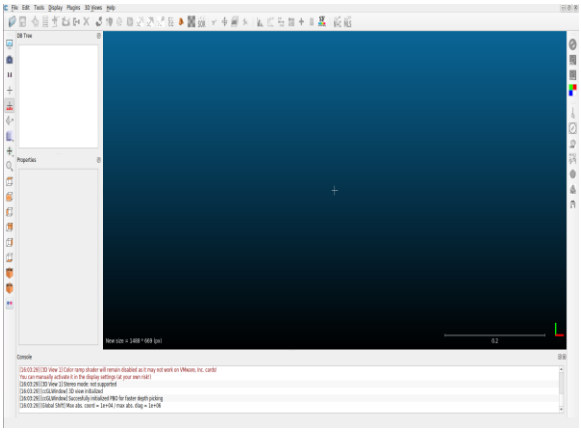
3. Cloudcompare.ccViewer
4. cloudcompare.CloudCompare

Then after few moments the cloudcompare will be open. After that, we can open the dataset las file and click on keyboard with 'ctrl+v' to snap every moment of the dataset to do an animation.

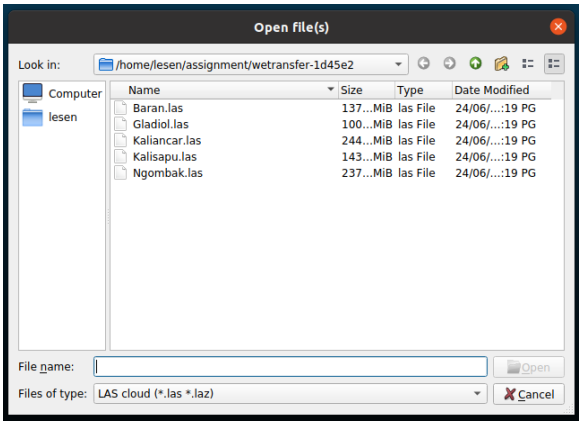
RESULT

CloudCompare

Below is the cloudcompare interface and it will be called out when some command is typed in the terminal.

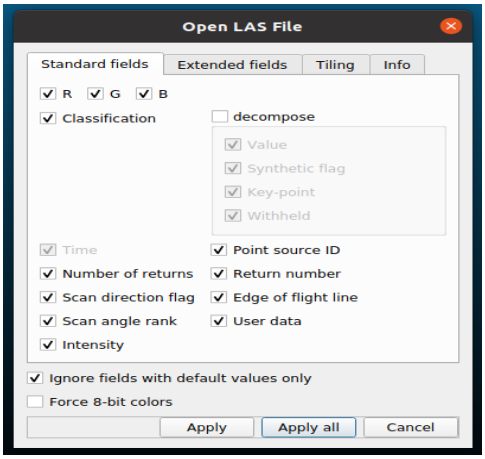


Then we can open the dataset file by using this software. Besides, we need to set the type of file to become "LAS Cloud (*.las, *.laz)" because the dataset we got is in las type.

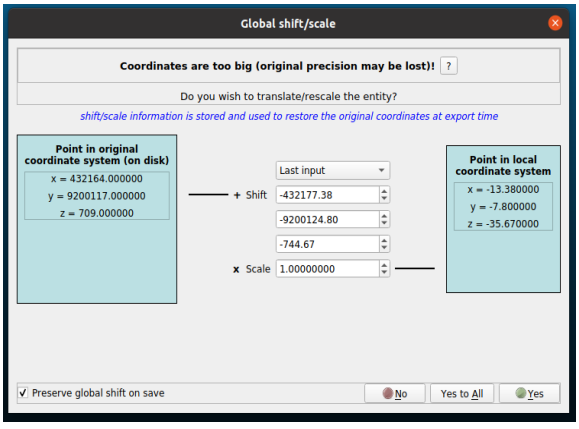


After one of the dataset is selected, some window will pop out to do some setting for the dataset.

The first window is about some standard fields setting, extended fields setting, tiling and info setting. As in the picture below, we can modify the setting by tick in the box.



After applying all in the window above, the other window will pop out also which is a global shift or scale setting. In this window, we can modify the shift and scale of the dataset and click yes to all to proceed to the next step.



Below are some pictures which are opened by cloudcompare software from the dataset.

ABaran.las

Kaliancar.las

Kalisapu.las

Gladiol.las

Ngombak.las

Ctest

Ctest is run to test all the test001.py to test 025.py files.

```
(CloudComPy39) lesen@lesen:~/anaconda3/envs/CloudComPy39/doc/PythonAPI_test$ ctest
Test project /home/lesen/anaconda3/envs/CloudComPy39/doc/PythonAPI_test
      Start  1: PYCC_test001
 1/25 Test  #1: PYCC_test001 .....................   Passed    8.45 sec
      Start  2: PYCC_test002
 2/25 Test  #2: PYCC_test002 .....................   Passed   27.48 sec
      Start  3: PYCC_test003
 3/25 Test  #3: PYCC_test003 .....................   Passed   22.53 sec
      Start  4: PYCC_test004
 4/25 Test  #4: PYCC_test004 .....................   Passed    8.93 sec
      Start  5: PYCC_test005
 5/25 Test  #5: PYCC_test005 .....................   Passed    9.75 sec
      Start  6: PYCC_test006
 6/25 Test  #6: PYCC_test006 .....................   Passed   23.37 sec
      Start  7: PYCC_test007
 7/25 Test  #7: PYCC_test007 .....................   Passed   13.60 sec
      Start  8: PYCC_test008
 8/25 Test  #8: PYCC_test008 .....................   Passed    8.46 sec
      Start  9: PYCC_test009
 9/25 Test  #9: PYCC_test009 .....................   Passed   13.97 sec
      Start 10: PYCC_test010
10/25 Test #10: PYCC_test010 .....................   Passed   35.28 sec
      Start 11: PYCC_test011
11/25 Test #11: PYCC_test011 .....................   Passed   11.70 sec
      Start 12: PYCC_test012
12/25 Test #12: PYCC_test012 .....................   Passed    2.75 sec
      Start 13: PYCC_test013
13/25 Test #13: PYCC_test013 .....................   Passed    7.74 sec
      Start 14: PYCC_test014
14/25 Test #14: PYCC_test014 .....................   Passed   39.83 sec
      Start 15: PYCC_test015
15/25 Test #15: PYCC_test015 .....................   Passed    3.47 sec
      Start 16: PYCC_test016
16/25 Test #16: PYCC_test016 .....................   Passed    8.81 sec
      Start 17: PYCC_test017
17/25 Test #17: PYCC_test017 .....................   Passed   12.94 sec
      Start 18: PYCC_test018
18/25 Test #18: PYCC_test018 .....................   Passed    7.90 sec
      Start 19: PYCC_test019
19/25 Test #19: PYCC_test019 .....................***Failed   21.18 sec
      Start 20: PYCC_test020
20/25 Test #20: PYCC_test020 .....................   Passed   15.14 sec
```

But if the process time is too long, the test will be failed

```
      Start 21: PYCC_test021
21/25 Test #21: PYCC_test021 .....................   Passed   81.14 sec
      Start 22: PYCC_test022
22/25 Test #22: PYCC_test022 .....................   Passed   24.50 sec
      Start 23: PYCC_test023
23/25 Test #23: PYCC_test023 .....................   Passed   14.98 sec
      Start 24: PYCC_test024
24/25 Test #24: PYCC_test024 .....................***Timeout 100.04 sec
      Start 25: PYCC_test025
25/25 Test #25: PYCC_test025 .....................   Passed   11.90 sec

92% tests passed, 2 tests failed out of 25

Total Test time (real) = 535.87 sec

The following tests FAILED:
        19 - PYCC_test019 (Failed)
        24 - PYCC_test024 (Timeout)
```

If the files are unable to successfully run it, we need to find the directory **/anaconda3/envs/CloudComPy39/doc/PythonAPI_test**, open the DartConfiguration.tcl file and extend the timeout value to 100 seconds or more.

```
87 # Testing options
88 # TimeOut is the amount of time in seconds to wait for processes
89 # to complete during testing.  After TimeOut seconds, the
90 # process will be summarily terminated.
91 # Currently set to 40 seconds per test
92 TimeOut: 40
```

```
87 # Testing options
88 # TimeOut is the amount of time in seconds to wait for processes
89 # to complete during testing.  After TimeOut seconds, the
90 # process will be summarily terminated.
91 # Currently set to 40 seconds per test
92 TimeOut: 100
```

```
87 # Testing options
88 # TimeOut is the amount of time in seconds to wait for processes
89 # to complete during testing.  After TimeOut seconds, the
90 # process will be summarily terminated.
91 # Currently set to 40 seconds per test
92 TimeOut: 150
```

After that run the ctest—rerun-failed to run all the test0xx.py until all python scripts are successfully executed.

```
(CloudComPy39) lesen@lesen:~/anaconda3/envs/CloudComPy39/doc/PythonAPI_test$ ctest --rerun-failed
Test project /home/lesen/anaconda3/envs/CloudComPy39/doc/PythonAPI_test
    Start 19: PYCC_test019
1/2 Test #19: PYCC_test019 .....................   Passed   23.13 sec
    Start 24: PYCC_test024
2/2 Test #24: PYCC_test024 .....................   Passed  148.98 sec

100% tests passed, 0 tests failed out of 2
```

Animation link

https://drive.google.com/file/d/1zn2h93f5u5yl54o9MykN1XiR0vUs2umw/view?usp=sharing