PROJECT REPORT ON

Image Reconstruction from the Point Cloud Generated by LiDAR

Submitted by

Abhilash Bhardwaj

Computer Science

Towards Partial Fulfilment of the award of certificate of

Orientation Course for Engineering and Science Graduates (OCES 2019)

Under The Guidance of

Mr. Supratim Majumdar

CnID/E&IG

Human Resource Development Division

Bhabha Atomic Research Centre

Mumbai 400085. INDIA

July 2020

Certificate

This is to certify that **Abhilash Bhardwaj, Computer Science**, OCES-2018, BARC Training School, Mumbai has completed his/her Project Work on **Image Reconstruction from the Point Cloud Generated by LiDAR** under my guidance.

Signature ______________________________

Name & Designation_____________________

Division/Unit__________________________________

ACKNOWLEDGEMENT

The elation and gratification of this project will be incomplete without mentioning all the people who helped me to make it possible, whose gratitude and encouragement were invaluable to me. I would like to thank God, almighty, our supreme guide, for bestowing is blessings upon me in my entire endeavor. I express my sincere gratitude to Mr. Supratim Majumdar Sir, for his guidance and support and batch mates for their support and suggestions.

Abhilash Bhardwaj

Table of Contents:

ABSTRACT

LiDAR (Light Detection and Ranging), is a remote sensing method like RADAR, with lot of added advantages. Unlike RADAR, LiDAR incorporates light in form of pulsed laser for its purpose. This method can be extended to wide variety of applications related to surveillance, area survey, object detection, automatic navigation etc. LiDAR provides precise 3D measurement of data over short to long ranges. Here in this project our idea is to explore the different aspects and representation of results provided by a LiDAR. This work presents different approaches to understand the point clouds and visualize them. This work also includes development of a GUI visualization tool which supports most common formats of LiDAR. The tool takes input and constructs a 3D representation of point cloud with some customizable parameters.

Key Terms: LiDAR, Point Cloud, Remote Sensing

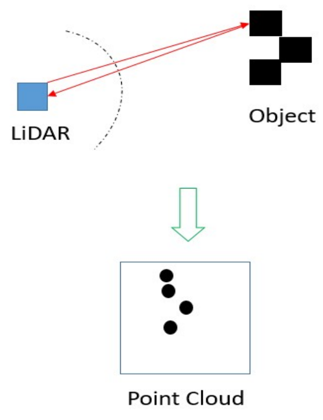
Introduction

<u>LiDAR</u>

Detection and ranging methods are quite popular in variety of applications. These methods allow us to estimate various properties of object and environment efficiently. Imagine a RADAR which detects presence of aircrafts or SONAR which helps a lot in maritime applications. These employ some kind of remote sensing methods to sense objects around them. Imagine how difficult and ineffective it will be for us to detect aircraft manually.

Unlike RADAR and SONAR, LiDAR uses visible, UV or near IR range EM waves to carry out its operation. But this is not the single difference between LiDAR and RADAR. LiDAR is particularly useful in detecting exact shape and size parameter, while RADAR mostly focused on detecting the object its shape and size are not of much of importance. So while surveying any terrain or detecting properties of object LiDAR suits best over RADAR. Also as LiDAR uses shorter wavelength so small object detection is also possible.

<u>Basic Working Principle of LiDAR-</u>



Some basic terms related to LiDAR-

1. *Point Cloud*: In layman terms if we say it is just cloud of points generated by LiDAR as output. Point can be represented as 3D vector where its 3 values represent corresponding distance along X, Y and Z axis. So nearly all LiDAR output file formats have fields for storing coordinate values of all above three coordinates.
2. *Laser*: LASER stands for Light amplification by stimulated emission of Radiation, this is the source of wave used for detection and ranging.
3. *Photodiode*: Semiconductor diode which generates current when exposed to light. This is used as detector in our LiDAR.

The LiDAR instrument fires rapid pulses of laser light at a surface. A detector on the instrument measures the amount of time it takes for each pulse to bounce back. The reflected wave is detected by Photodiode. One simple distance estimation can be done using ToF (Time of flight) information.

Distance = (Speed of Light * ToF) / 2

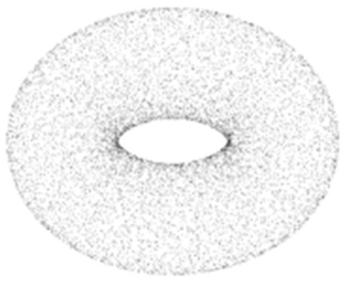
LASER, detector, optics and rotating mechanism, timing electronics to measure difference between transmitted and received pulse are basic components of LiDAR. Here time sensitivity is of much importance as even a difference of 1ns can lead to significant difference in depth estimation because of speed of light.

Applications of LiDAR-

1. LiDAR is particularly used to make high resolution maps with applications in geography, archeology, forestry, military etc.
2. LiDAR point cloud information is also used to generate realistic graphics.
3. It also finds its application in object detection and surveillance.
4. Self-driving cars where LiDAR will be used in real time to detect obstacles and navigation
5. In agriculture LiDAR can be used to make 3D elevation map of a particular land.

Point clouds and their representations

Point cloud is a collection of data points in a given coordinate system. There are many formats associated with representing point cloud data. These file formats not only store the point information but also store metadata associated with the system.



A Sample of point cloud

In this work we have gone through three different representation of LiDAR output. Each file format has its own attributes and properties. The three formats are-

1. las Format (.las files)
2. pcd Format (.pcd files)
3. csv Format (.csv files)

<u>LAS Format</u>

The LAS format is a file format designed for the interchange and archiving of LiDAR point cloud data. It is *an open, binary format* specified by the American Society for Photogrammetry and Remote Sensing. The format is widely used and regarded as an industry standard for LiDAR data.

The LAS file contains a number of fields for each point that can be useful for analysis and for display:

- x, y coordinates, most often in the UTM projection or an equivalent
- z, the elevation
- Classification.
- Intensity of the returned pulse. This looks like a grayscale image, with the caveat that includes a single frequency of light, which may most often be NIR. There is no standard for the range or scaling of these values, and they may not be corrected for range; despite these caveats, the intensity display can be very informative.
- an RGB value, merged from a camera flown with the laser scanner
- The return number, and the total number of returns from the pulse. The LAS format allows multiple returns per transmitted pulse, but typically the vast majority will be the first and only return. When there are multiple returns from a pulse, only the last could be on the ground, but even the last return could be in the vegetation or features like power lines. The older LAS 1.2 and 1.3 could handle 5 returns per pulse, while the 1.4 format can handle up to 15.
- The scan angle, which indicates how far from nadir the scanner was pointed; this is typically up to about 20-25 degrees, positive and negative depending on which side of nadir. Away from nadir, smooth water surfaces will act like mirrors and there will be no returned energy.
- Overlap points, where two flight lines covered the same area.
- The LAS format allows distribution of the full waveform data, but that option is rarely used in practice. The resulting file sizes greatly increase the data storage, and few ultimate end users want or need that data.
- Two fields allow entry of additional information, Point ID and User Data

The LAS file has a header with metadata about the file.

- It is supposed to have the datum/projection information embedded, but it may be missing, which complicates using the data.
- It has the elevation range, but usually has elevations that are both excessively high and excessively low. When used to color the elevations, this leads to a very compressed color scale. There is a sense in the LiDAR community that you should never delete returns, but just code them as noise (the older LAS versions had a classification code for low noise, and 1.4 added high noise), but this does not help in determining the true elevation range. I think the never delete a point may make sense for the data producers, but not the end users.

PCD Format

PCD (Point Cloud Data) format is basically used with PCL (Point Cloud Library). It header contains following fields-

VERSION – Version of PCD file format

FIELDS – Attributes available in file, dimension of data

SIZE – Size of each dimension.

TYPE – data type of each dimension

COUNT - specifies how many elements does each dimension have

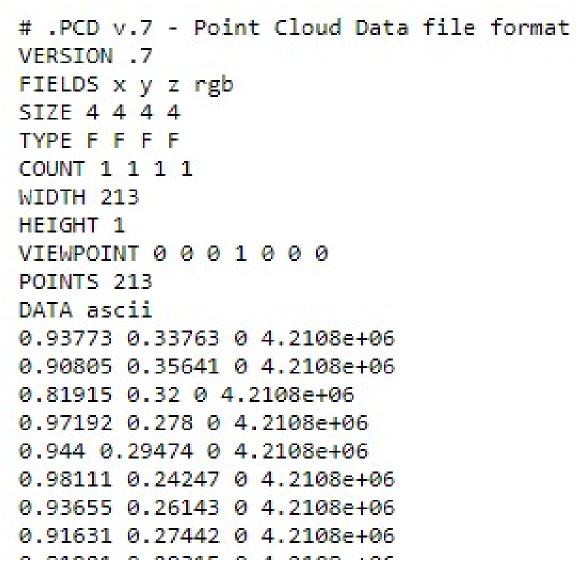
WIDTH - specifies the width of the point cloud dataset in the number of points

HEIGHT - specifies the height of the point cloud dataset in the number of points. HEIGHT has two meanings. It can specify the height (total number of rows) of an organized point cloud dataset. It is set to 1 for unorganized datasets (thus used to check whether a dataset is organized or not).

VIEWPOINT - specifies an acquisition viewpoint for the points in the dataset

POINTS - specifies the total number of points in the cloud

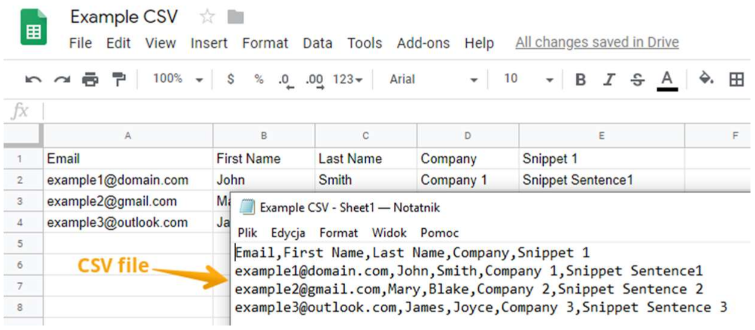
DATA - specifies the data type that the point cloud data is stored in like ASCII or BINARY



```
# .PCD v.7 - Point Cloud Data file format
VERSION .7
FIELDS x y z rgb
SIZE 4 4 4 4
TYPE F F F F
COUNT 1 1 1 1
WIDTH 213
HEIGHT 1
VIEWPOINT 0 0 0 1 0 0 0
POINTS 213
DATA ascii
0.93773 0.33763 0 4.2108e+06
0.90805 0.35641 0 4.2108e+06
0.81915 0.32 0 4.2108e+06
0.97192 0.278 0 4.2108e+06
0.944 0.29474 0 4.2108e+06
0.98111 0.24247 0 4.2108e+06
0.93655 0.26143 0 4.2108e+06
0.91631 0.27442 0 4.2108e+06
```

CSV Format

CSV stands for Comma Separated Values, is a simple storage format. Here every data is separated using comma.


Example CSV
File Edit View Insert Format Data Tools Add-ons Help All changes saved in Drive
Example CSV - Sheet1 — Notatnik
Plik Edycja Format Widok Pomoc
CSV file

	A	B	C	D	E	F
1	Email	First Name	Last Name	Company	Snippet 1	
2	example1@domain.com	John	Smith	Company 1	Snippet Sentence1	
3	example2@gmail.com	Ma				
4	example3@outlook.com	Ja				

```
Email,First Name,Last Name,Company,Snippet 1
example1@domain.com,John,Smith,Company 1,Snippet Sentence1
example2@gmail.com,Mary,Blake,Company 2,Snippet Sentence 2
example3@outlook.com,James,Joyce,Company 3,Snippet Sentence 3
```

Point Cloud Visualization

After acquisition of data using LiDAR in any of the file formats our next goal is to visualize the point cloud dataset. We need to reconstruct the 3D view using point cloud dataset. For visualization purpose we relied on many open source libraries. Language used for reading the acquired dataset as well as plotting the data is Python 3.5.

Packages used for visualization

Matplotlib – Matplotlib library is used for plotting the data in python.

pptk – Another very good library specifically for visualizing the point cloud data. It supports a lot features to set point size, look at point also point cloud coloring feature. [11]

Pandas - pandas is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool, built on top of the Python programming language

Numpy - NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.

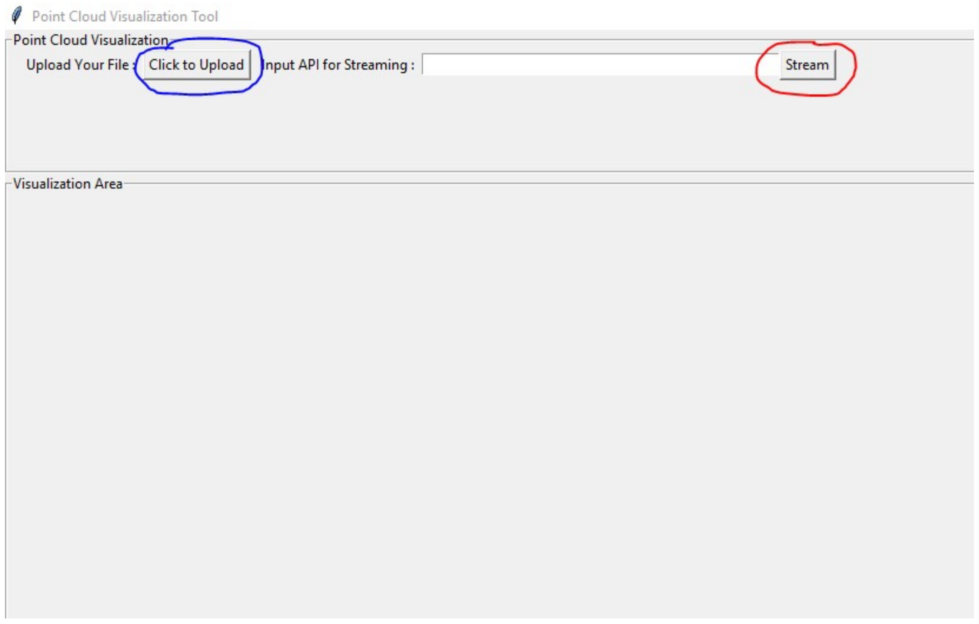
Laspy - Laspy is a pythonic library for reading, modifying and writing LAS files. Support for LAZ is limited to reading LAS version 1.0-1.3 files. Laspy is compatible with Python 2.6+ and 3.5+. [8]

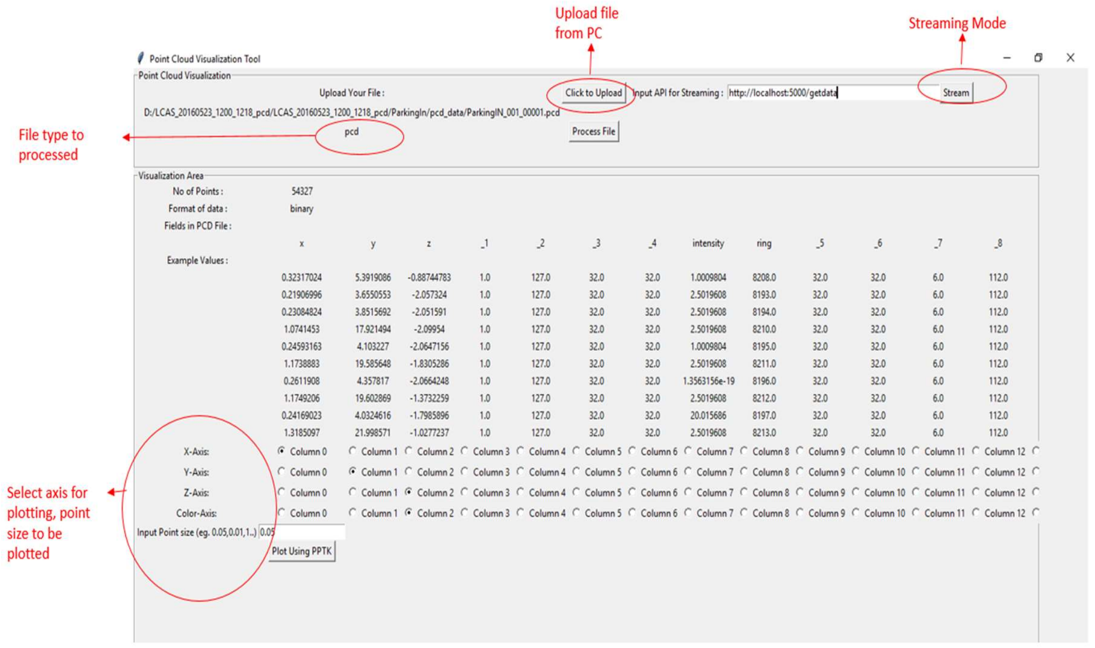
Pypcd - A Python module to read and write point clouds stored in the PCD file format, used by the Point Cloud Library (PCL).[9]

Tkinter – A python library for developing GUI applications.[10]

GUI Point cloud Visualization Tool-

In this work, we have developed a small GUI visualization tool, which facilitates our purpose.


Point Cloud Visualization Tool
Point Cloud Visualization
Upload Your File :
Click to Upload
Input API for Streaming :
Stream
Visualization Area



Input API for Streaming : http://localhost:5000/getdata

D:/LCAS_20160523_1200_1218_pcd/LCAS_20160523_1200_1218_pcd/ParkingIn/pcd_data/ParkingIN_001_00001.pcd

pcd

No of Points : 54327

Format of data : binary

Fields in PCD File :

Example Values :

x	y	z	_1	_2	_3	_4	intensity	ring	_5	_6	_7	_8
0.32317024	5.3919086	-0.88744783	1.0	127.0	32.0	32.0	1.0009804	8208.0	32.0	32.0	6.0	112.0
0.21906996	3.6550553	-2.057324	1.0	127.0	32.0	32.0	2.5019608	8193.0	32.0	32.0	6.0	112.0
0.23084824	3.8515692	-2.051591	1.0	127.0	32.0	32.0	2.5019608	8194.0	32.0	32.0	6.0	112.0
1.0741453	17.921494	-2.09954	1.0	127.0	32.0	32.0	2.5019608	8210.0	32.0	32.0	6.0	112.0
0.24593163	4.103227	-2.0647156	1.0	127.0	32.0	32.0	1.0009804	8195.0	32.0	32.0	6.0	112.0
1.1738883	19.585648	-1.8305286	1.0	127.0	32.0	32.0	2.5019608	8211.0	32.0	32.0	6.0	112.0
0.2611908	4.357817	-2.0664248	1.0	127.0	32.0	32.0	1.3563156e-19	8196.0	32.0	32.0	6.0	112.0
1.1749206	19.602869	-1.3732259	1.0	127.0	32.0	32.0	2.5019608	8212.0	32.0	32.0	6.0	112.0
0.24169023	4.0324616	-1.7985896	1.0	127.0	32.0	32.0	20.015686	8197.0	32.0	32.0	6.0	112.0
1.3185097	21.998571	-1.0277237	1.0	127.0	32.0	32.0	2.5019608	8213.0	32.0	32.0	6.0	112.0

X-Axis: [✓ Column 0] Column 1 Column 2 Column 3 Column 4 Column 5 Column 6 Column 7 Column 8 Column 9 Column 10 Column 11 Column 12

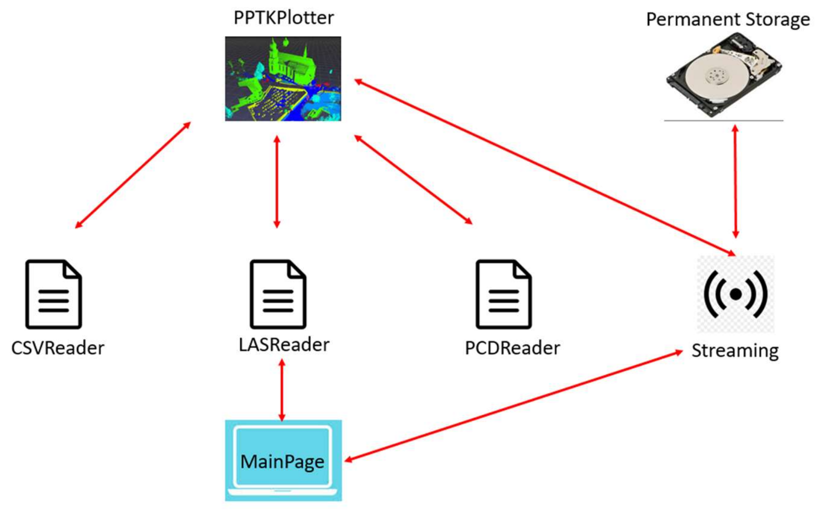
Y-Axis: Column 0 [✓ Column 1] Column 2 Column 3 Column 4 Column 5 Column 6 Column 7 Column 8 Column 9 Column 10 Column 11 Column 12

Z-Axis: Column 0 Column 1 [✓ Column 2] Column 3 Column 4 Column 5 Column 6 Column 7 Column 8 Column 9 Column 10 Column 11 Column 12

Color-Axis: Column 0 Column 1 [✓ Column 2] Column 3 Column 4 Column 5 Column 6 Column 7 Column 8 Column 9 Column 10 Column 11 Column 12

Input Point size (eg. 0.05,0.01,1..) 0.05

Application Architecture



This tool supports two modes of visualization,

1. File Mode
2. Streaming Mode

File Mode – In file mode, we have option of uploading file of supported formats. This tool supports three file formats .las, .pcd and .csv file format. After uploading file one can process the file to get the metadata.

Procedure of visualization using File Mode –

1. Upload the required file from PC (either LAS, PCD or CSV)
2. Process the file to get metadata and few values from the file. If process file fails to load any metadata then file format is not supported by the application.
3. Now we can configure our coordinate values for plotting, also we can choose a color axis which will be used to color the point cloud. Other than color configuration we can configure point size for plotting.

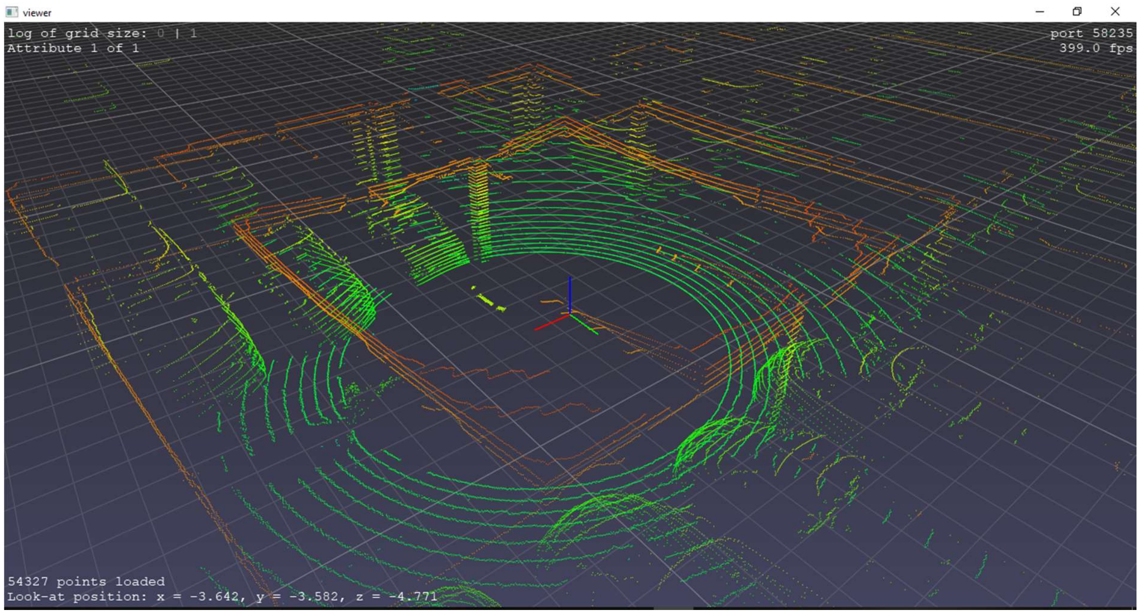
4. After all configuration we can see the plot, the plot will be created using pptk llibrary and it will be shown to user.
5. In viewer window of pptk we can perform multiple functionalities like zoom, rotation etc

Streaming Mode

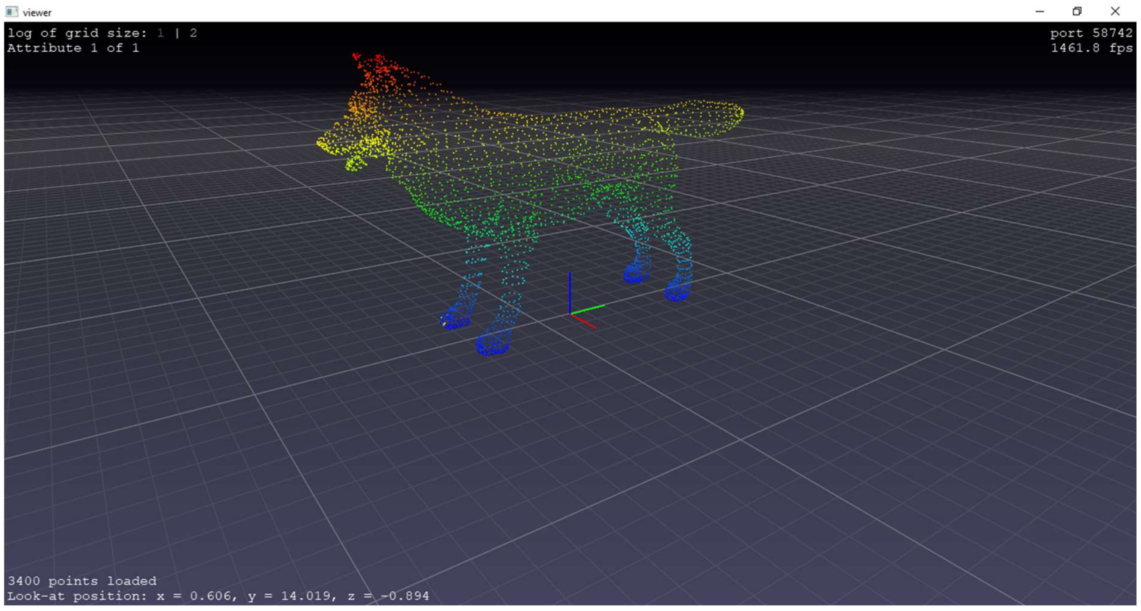
Streaming mode takes a web api as input. The web api should be active. In current implementation we have created a RESTful Web API. On http get request, it responds with a 2D array of (N x 3) dimension. In streaming mode the tool automatically queries API in loop after a fixed unit of time. Upon successfully receiving response from the API, tool loads data to the pptk viewer for visualization. The tool also saves that incoming data to file for further processing. So streaming mode can directly be used to visualize LiDAR point cloud data and save the real time data any point cloud storage format.

Visualization Results: 3D reconstruction of Point Cloud Data

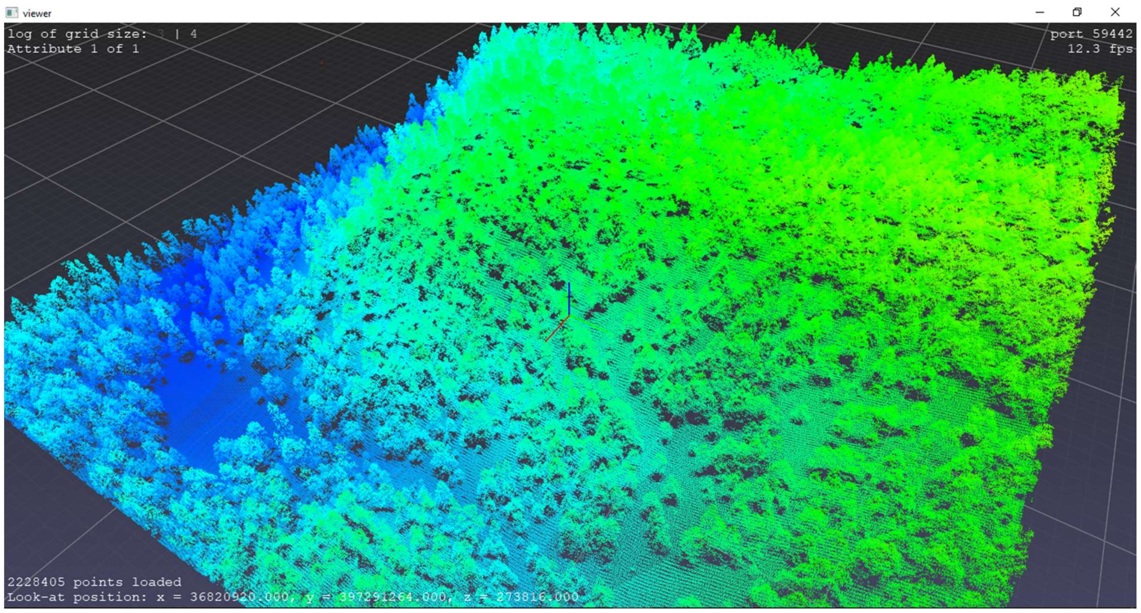
Point cloud reconstruction of In-Parking [5]



Point cloud reconstruction of Wolf [4]



3D reconstruction of Terrain [5]

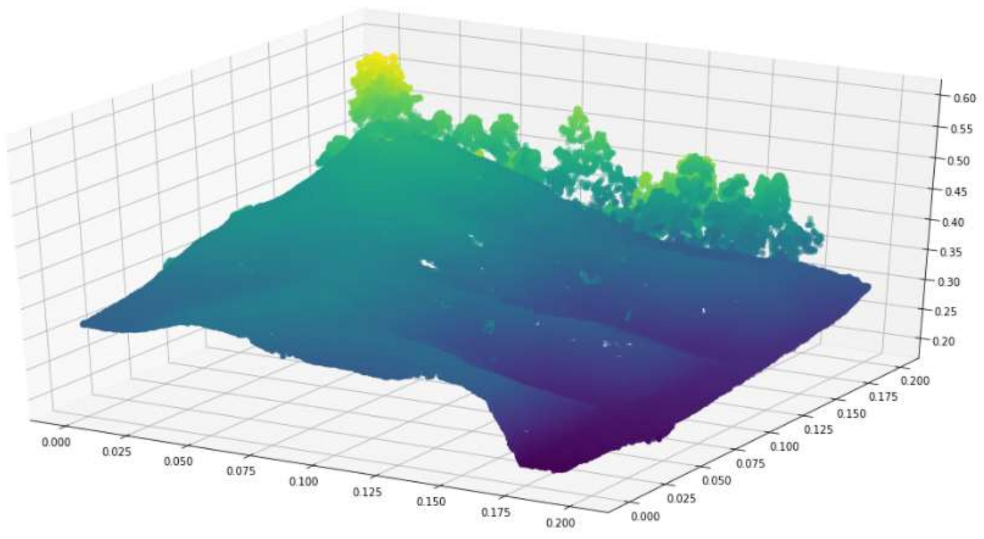


```
log of grid size: 3 | 4
Attribute 1 of 1
port 59442
12.3 fps
2228405 points loaded
Look-at position: x = 36820920.000, y = 397291264.000, z = 273816.000
```

2D Image Experiments: Terrain Point Cloud

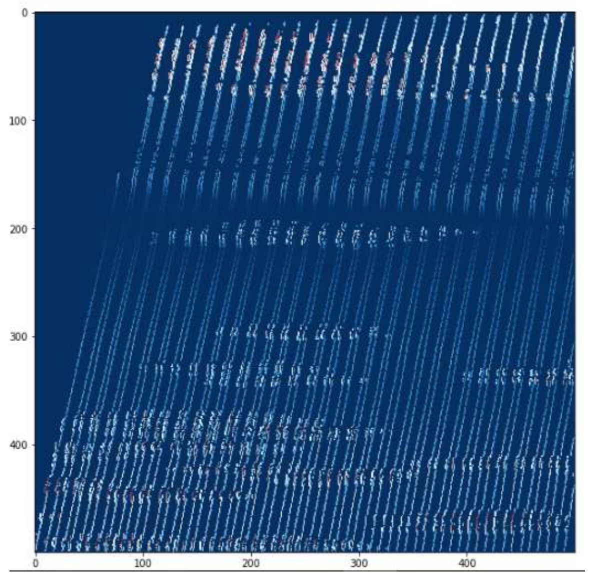
In this work we have also tried to convert 3D data to 2D images (so that we can treat like images). These images can be further used to know elevation or intensity related details. If we can create 2D images then they can be used with CNN for various object detection purpose. In this work we did small work to put 3D data into 2D images. We have used Z-axis information to generate color for a pixel coordinated given by X-Y axis values, hence flattening the 3D data to 2D image.

Following is the 3D plot of terrain using matplotlib-

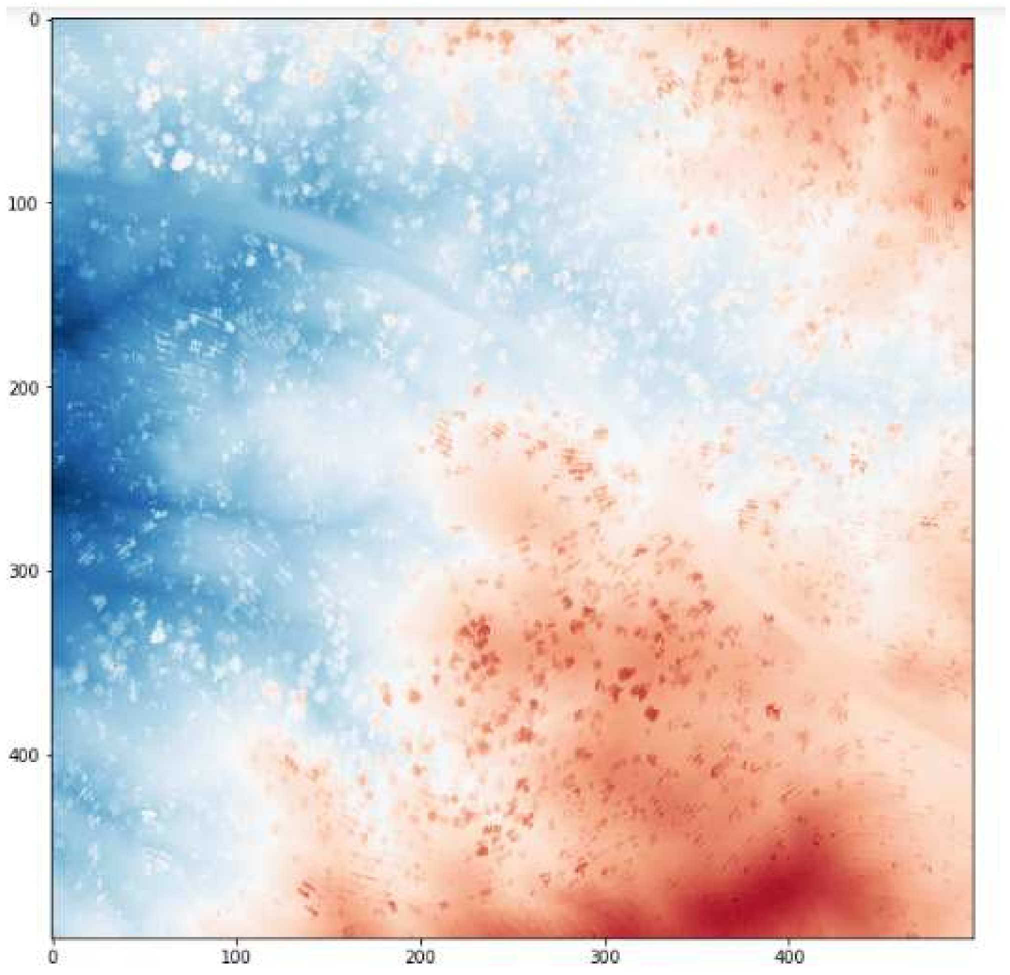


The following image is created using naïve flattening approach, where a pixel is determined by

X-Y coordinate and its intensity is determined by the Z-coordinate value of that pixel.



Above 2D representation was vague and not at all informative. So in next experiments we tried to color all 8-neighbour pixels. This approach we have tried because pixels are always integer values but our X-Y-Z are real values. Due to quantization and taking floor value lots of values are lost.



This image contains more information than previous image. Here RED areas are on more elevation while blue are at depth. Small spots in image may represent vegetation which is at more height than near surrounding. In this image we can easily categorize elevated areas. This approach can be extended to generate DEM (digital elevation models).

Conclusion

In this work, we have explored different formats of point cloud generated by LiDAR and how to use them further for visualization. Starting from the basic principle of LiDAR, we have studied the popular point cloud representations, their attributes and how these attributes can help us in 3D reconstruction. For different formats we have used different open source libraries to extract data. After extracting we tried 3D plotting tools to plot them. To ease the process of 3D visualization we have also created a small GUI visualization tool.

Considering real-time visualization of data we have also included a streaming feature of point cloud. This feature can periodically ask for data to the given endpoint and update the plot. By using endpoint based data acquisition we have made our tool independent of system used at LiDAR side.

Future Work

This project is first step towards understanding point cloud formats and 3D reconstruction. This work can be extended in different ways. One step further we can go for solid 3D construction. Here we can use point cloud to generate 3D surface. Other than this we can use this point cloud data to create Digital Elevation Model (DEM).

Another way will be to integrate and test point cloud library (PCL) for various tasks like segmentation, surface construction, registration, object detection etc. The Point Cloud Library (PCL) is a standalone, large scale, open project for 2D/3D image and point cloud processing. PCL is released under the terms of the BSD license, and thus free for commercial and research use.

Another extension of this project is object recognition and detection. We can integrate a trained machine learning model to our real-time data acquisition and visualization system for real-time detection of objects for surveillance purpose. For machine learning we can either use direct point cloud data in some form to train our model or we can go for 2D images derived using point cloud data for training.

References

[1] Yan, Z., Duckett, T. & Bellotto, N. Online learning for 3D LiDAR-based human detection: experimental analysis of point cloud clustering and classification methods. Auton Robot 44, 147–164 (2020). https://doi.org/10.1007/s10514-019-09883-y

[2] Fernandez-Diaz, Juan & Singhania, A. & Caceres, Jhon & Slatton, K & Starek, Michael & Kumar, R & Slatton, Prof. (2008). An overview of lidar point cloud processing software.

[3] http://www.semantic3d.net/view_dbase.php?chl=1

[4] http://www.acfr.usyd.edu.au/papers/SydneyUrbanObjectsDataset.shtml

[5] http://data.ign.fr/benchmarks/UrbanAnalysis/

[6] https://cs.nyu.edu/~silberman/datasets/nyu_depth_v2.html

[7] http://www.cs.cmu.edu/~vmr/datasets/oakland_3d/cvpr09/doc/

[8] https://laspy.readthedocs.io/en/latest/

[9] https://blog.pollithy.com/python/numpy/pointcloud/tutorial-pypcd

[10] https://docs.python.org/3/library/tkinter.html

[11] https://heremaps.github.io/pptk/