Point Clouds with ZED camera.ppt

Reference: lec3-11-7-PointCloud-frustum-pointnets-2018-12-20.odp

CTI One Corporation

Version: x0.1
Date: Jan 4, 2019
Project Lead: Harry Li, Ph.D.

Team members: Minh Duc Ong

Point Clouds

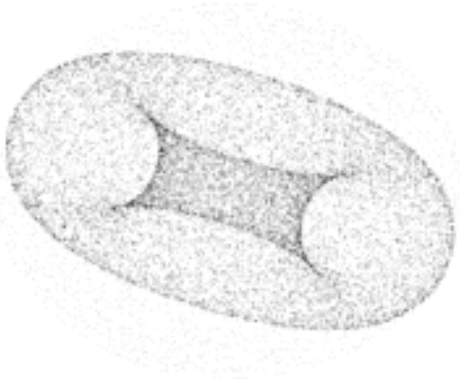
https://en.wikipedia.org/wiki/Point_cloud

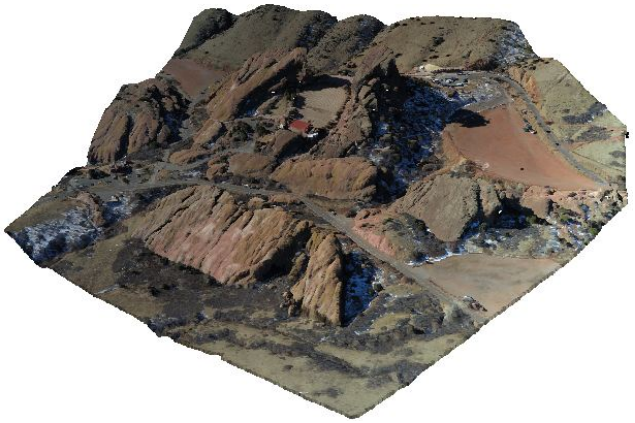
What is Point Clouds?

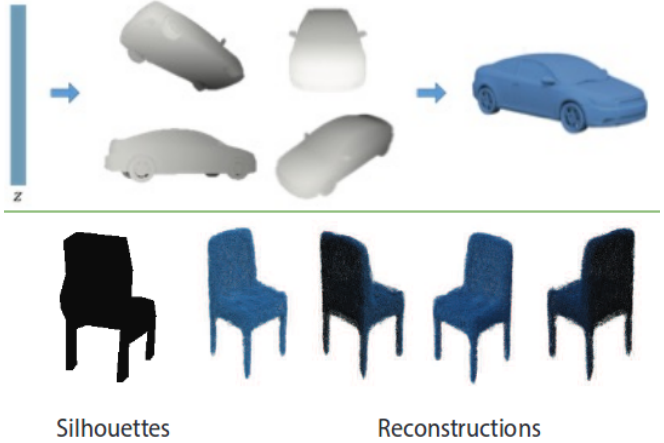
- Set of data Points in space.

- Point Clouds are generally produced by 3D scanners(Eg: ZED camera,..), which measure a large number of points on the external surfaces of objects around them.

- Point clouds are used for multiple purposes: 3D CAD model for manufactured parts, visualization, animation,..







Functions Point Cloud data

We use ZED API to get Point Cloud data:

```
sl::Mat point_cloud;
zed.retrieveMeasure(point_cloud, MEASURE_XYZRGBA);
float4 point3D;
// Get the 3D point cloud values for pixel (i,j)
point_cloud.getValue(i,j,&point3D);
float x = point3D.x;
float y = point3D.y;
float z = point3D.z;
float color = point3D.w;
```

Table 1. ZED API Point-Cloud data format

Float4 point3D;
Channel 1: float x = point3D.x; Channel 2: float y Channel 3: float z Channel 4: float color //RGB values

The point cloud saves its data on 4 channels using 32-bit float for each channel. The last float (color) is used to store color information, R, G, B values are concaternated into a single 32 bit float.

To save the pointcloud to .pcd format:

```
bool saved = savePointCloudAs(zed, “PCD”, filename.c_str(), true);
```

The last argument = *true* indicate that RGB values are saved to .pcd file also.

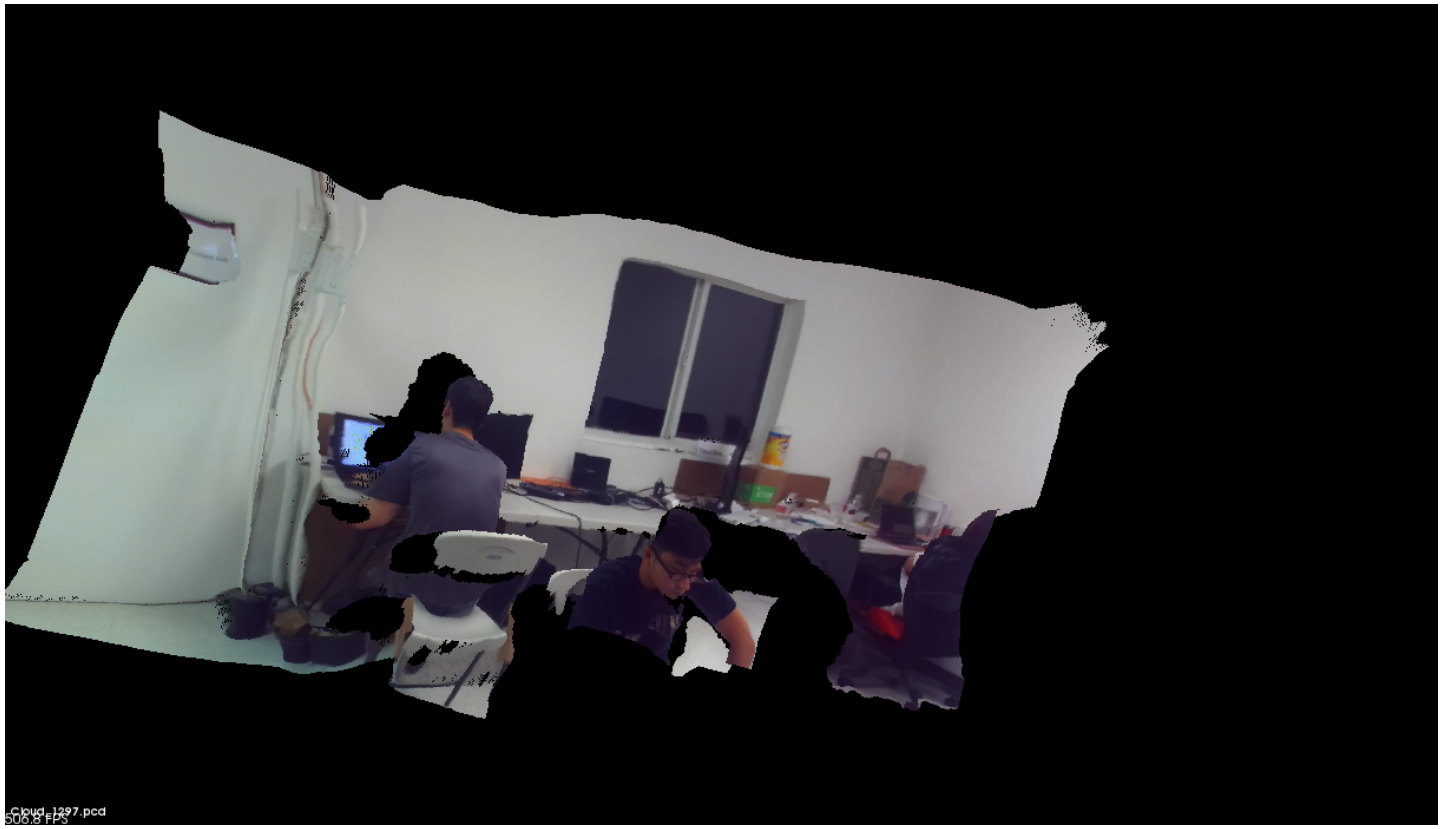
PCL library for Point Cloud Data

https://larrylisky.com/2016/11/03/point-cloud-library-on-ubuntu-16-04-lts/

Check the reference wiki: url above

Installation guide from the wiki: (1) how to build PCL on Ubunut 14.04 LTS. And (2) on 16.04 LTS and PCL to version 1.8

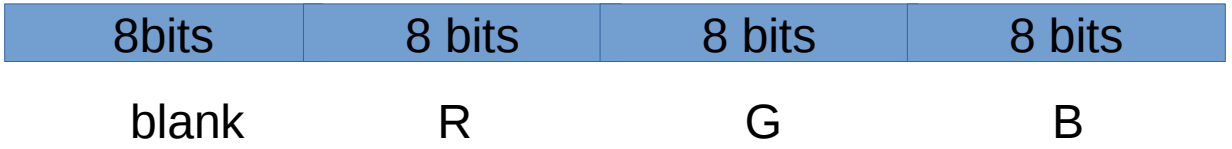
Run the program:
use pcl_viewer to visualize the .pcd file, open Terminal and run this command:
$pclviewer <file_name>.pcd



Note loading the file for the program to run takes more than 10 seconds

Read the X Y Z RGB from .pcd

We can read X Y Z RGB values from .pcd file by using pcl library; because R, G, B values are concaternated into 32 bit, so we need to use masks to do “bitwise and” to get R, G, B values




```
for (size_t i = 0; i < cloud->points.size (); ++i){
      uint32_t r = ((cloud->points[i].rgba) &
0x00ff0000)>>16;
      uint32_t g = ((cloud->points[i].rgba) &
0x0000ff00)>>8;
      uint32_t b = ((cloud->points[i].rgba) &
0x000000ff);
      std::cout << "    " << cloud->points[i].x
            << " "    << cloud->points[i].y
            << " "    << cloud->points[i].z
            << " "    << r
            << " "    << g
            << " "    << b
            << std::endl;
 }
```

X	Y	Z	R	G	B
0.626969	0.350749	1.02499	120	72	73
0.627736	0.350758	1.02501	121	73	70
0.628504	0.350767	1.02504	122	74	74
0.629275	0.350779	1.02508	122	74	74
0.630048	0.350792	1.02511	121	74	70
0.630821	0.350804	1.02515	121	75	71
0.631594	0.350817	1.02519	120	75	75
0.632393	0.350844	1.02527	120	76	76

Fig1. Result printed on Terminal