Final Presentation

Object Segmentation and Recognition
27 June 2016

Nitin Deshpande
Saad Abdullah
Radiah Rivu

Agenda (1/2)

- What are Point Clouds? [ND]
- Where do they come from? [ND]
- Why do we:
  - Need Point Clouds?
  - Need Point Cloud Segmentation? [ND]
- Segmentation and Recognition:
  - Region Growing
  - Group Correspondences [ND]

Agenda (2/2)

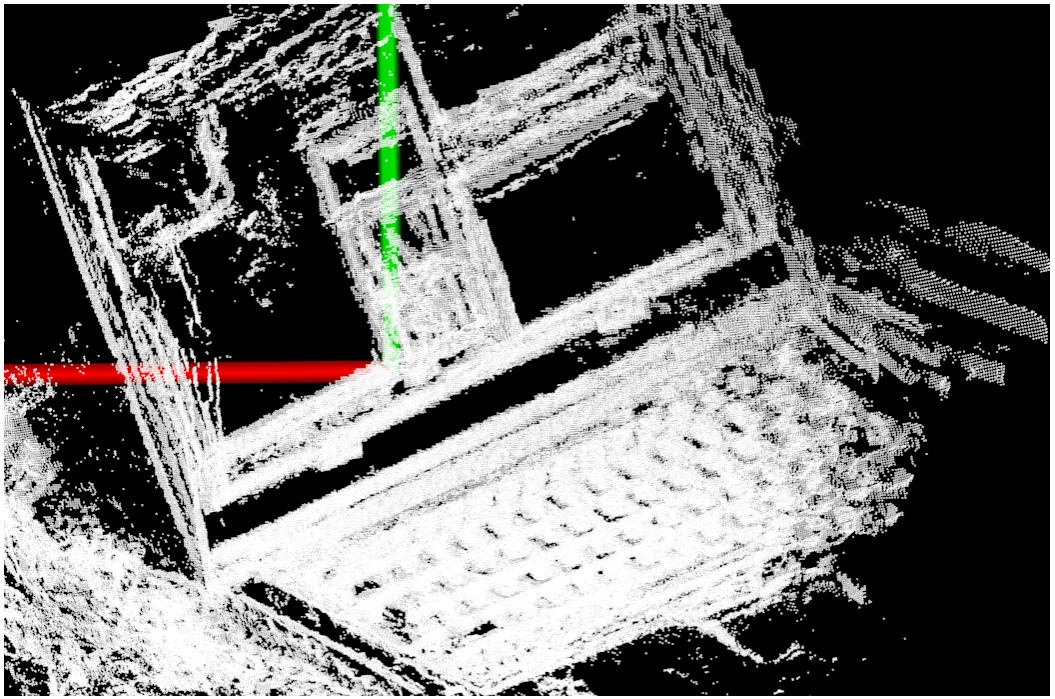
- Visualizing:
  - Point Clouds
  - Segmented Point Clouds [SA]
- Visualizing Point Cloud Data (PCD) format [SA]
- Visualizing Stanford Triangle Format or Polygon file format (PLY) [RR]
- Demonstration of work so far [RR]

What are Point Clouds? [1/3]

- Point Clouds are collection of *n-dimensional* points. In our case $n = 3$.
  - $p_i = \{x_i, y_i, z_i\}$; $P = \{p_1, p_2, p_3, \ldots, p_i\}$
  - every point p_i can have additional information on color and depth values » $p_i = \{x_i, y_i, z_i, r_i, g_i, b_i, dist_i\}$
- Point Clouds represent information about the surrounding world.

What are Point Clouds? [2/3]

- In C++ → p_i = std::vector<float> point;
  - P = std::vector<Point> points;



What are Point Clouds? [3/3]

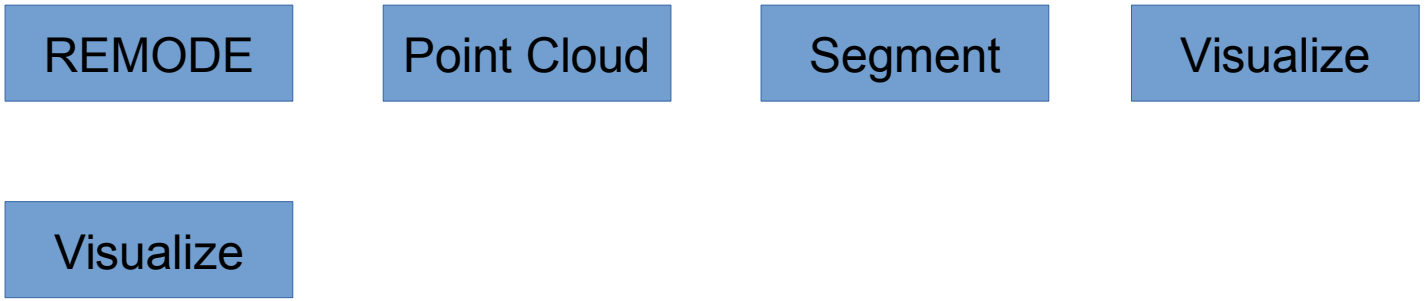
- Point Clouds have information about the world
- Different point cloud formats:
  - Point Cloud Data, Stanford Triangle Format [SA]
  - $p_i = \{x_i, y_i, z_i\}$; $P = \{p_1, p_2, p_3, \ldots, p_i\}$
  - p_i can have additional information »
  - $p_i = \{x_i, y_i, z_i, r_i, g_i, b_i, dist_i\}$
-

Where do they come from? [1/2]

- LASER scans from Velodyne LiDAR
- Kinect Camera
- ASUS Xtion
- Time of flight cameras
- Reconstruction (our interest)
- Simulation
- Many more sources …

Where do they come from? [2/2]

- In our case : the reconstructed point cloud obtained from REMODE.
- Monochromatic Point Cloud → **no RGB information!**

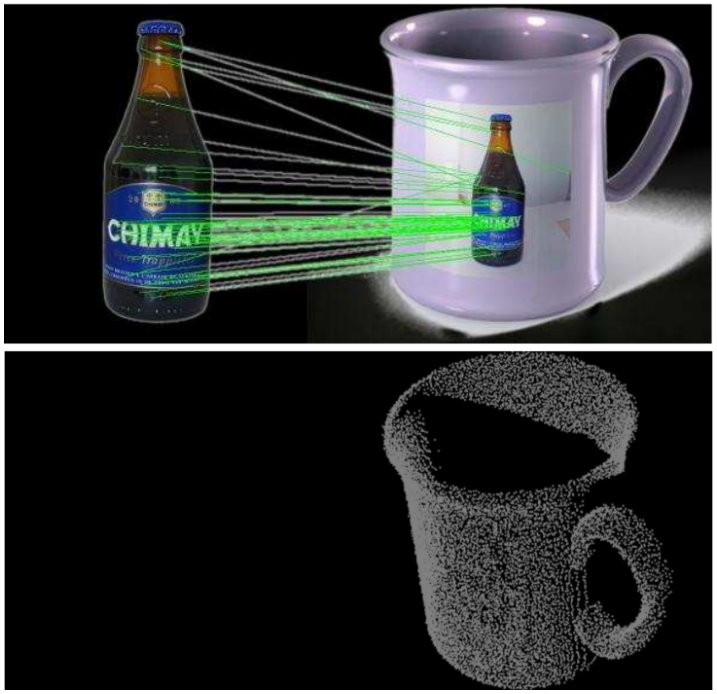


Why do we need Point Clouds? [1/2]

- Point Clouds have spatial information about the world
- It can be employed in different applications:
  - Navigation*
  - Obstacle detection and avoidance methods*
  - Object Recognition*
  - Robot interaction with the surroundings – picking & grasping – robot pouring some water into a glass.

Why do we need Point Clouds? [2/2]

- Point Clouds can solve ambiguities that otherwise may arise in images. Ex: Grabbing a bottle by the neck/body.



Why do we need Point Cloud Segmentation?

- Essential step in obstacle detection – to improve the performance.
- Improve the classification performance of the vehicle or a robot.
- Scenario : Complex scenes needs deeper understanding. Solution: Write a classifier. Role segmentation plays – Improves the performance of the classifier.

Segmentation [1/2]

- Region Growing: Intuition - Based on smoothness constraint of the points.
- Sorting the points in the point cloud by their curvature value.
- Pick a seed point with the least curvature and add to a list of points – region growing starts.
- Check the angle of normals between the seed point and current point – if yes, then add to current region.

Segmentation [2/2]

- Check for the distance between the seed and the neighboring point. Include it in the list if distance is less than the threshold.
- Exit criteria: Until there are unlabeled points in the cloud with the minimum curvature.
- Output : A set of clusters that are considered as the part of the same smooth surface.
- Region growing is simple and can be used for separating ground from the objects.

Recognition [1/3]

- Intuition: Given a set of correspondences group them together in geometrically consistent clusters → Correspondence grouping
- Use local features in the recognition pipelines – wealth of methods to find out features.
- Removes false positives, while retaining true positives

Recognition [2/3]

- Considering geometrical consistencies – build correspondences incrementally.
- Two schemes used in our approach:
  - Hough based voting scheme
  - geometric consistency based scheme.
- Geometric clustering is the simplest of scheme –
  $\|p_{i,m} - p_{j,m}\|_2 - \|p_{i,s} - p_{j,s}\|_2 < \varepsilon$, ε is the noise value or inaccuracy in key detection.

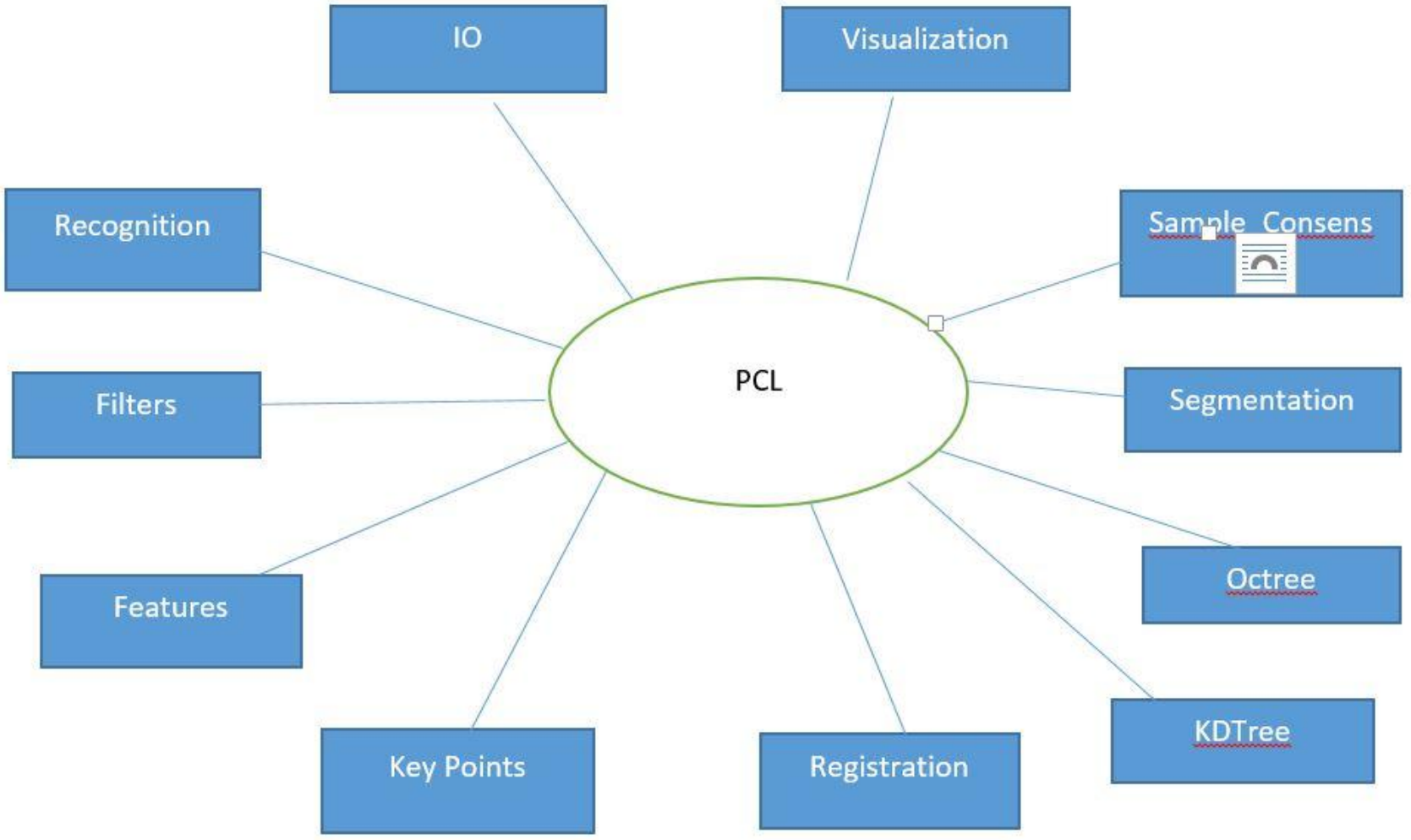
Recognition [3/3]

- Hough voting scheme: Intuition: Correspondence voting scheme based on 3D space. How?
- It is achieved by using a point's RF to reduce the voting space from 6D to 3D
- Handles multiple instances of the same object – by computing local maxima in the Hough Space.

Segmentation and Visualization

Library Used: Point Cloud Library(PCL)

- The PCL framework contains numerous state-of-the art algorithms including filtering, feature estimation, surface reconstruction, registration, model fitting and segmentation.
- For simplicity PCL provides a series of modular libraries.


IO
Visualization
Recognition
Sample_Consens
PCL
Filters
Segmentation
Features
Octree
KDTree
Key Points
Registration

Our focus

- We have selected the pcl_visualization library.
- This library helps to visualize the results of algorithms operating on 3D point cloud data.
- One advantage of visualization library is that it allows us to set visual properties for any n-D point cloud datasets in pcl.

Requirements

- Common- The **pcl_common** library contains the common data structures and methods used by the majority of PCL libraries
- VTK- The Visualization Toolkit (VTK) is an open-source, freely available software system for 3D computer graphics, image processing, and visualization
- IO- The **pcl_io** library contains classes and functions for reading and writing point cloud data (PCD) files, as well as capturing point clouds from a variety of sensing devices

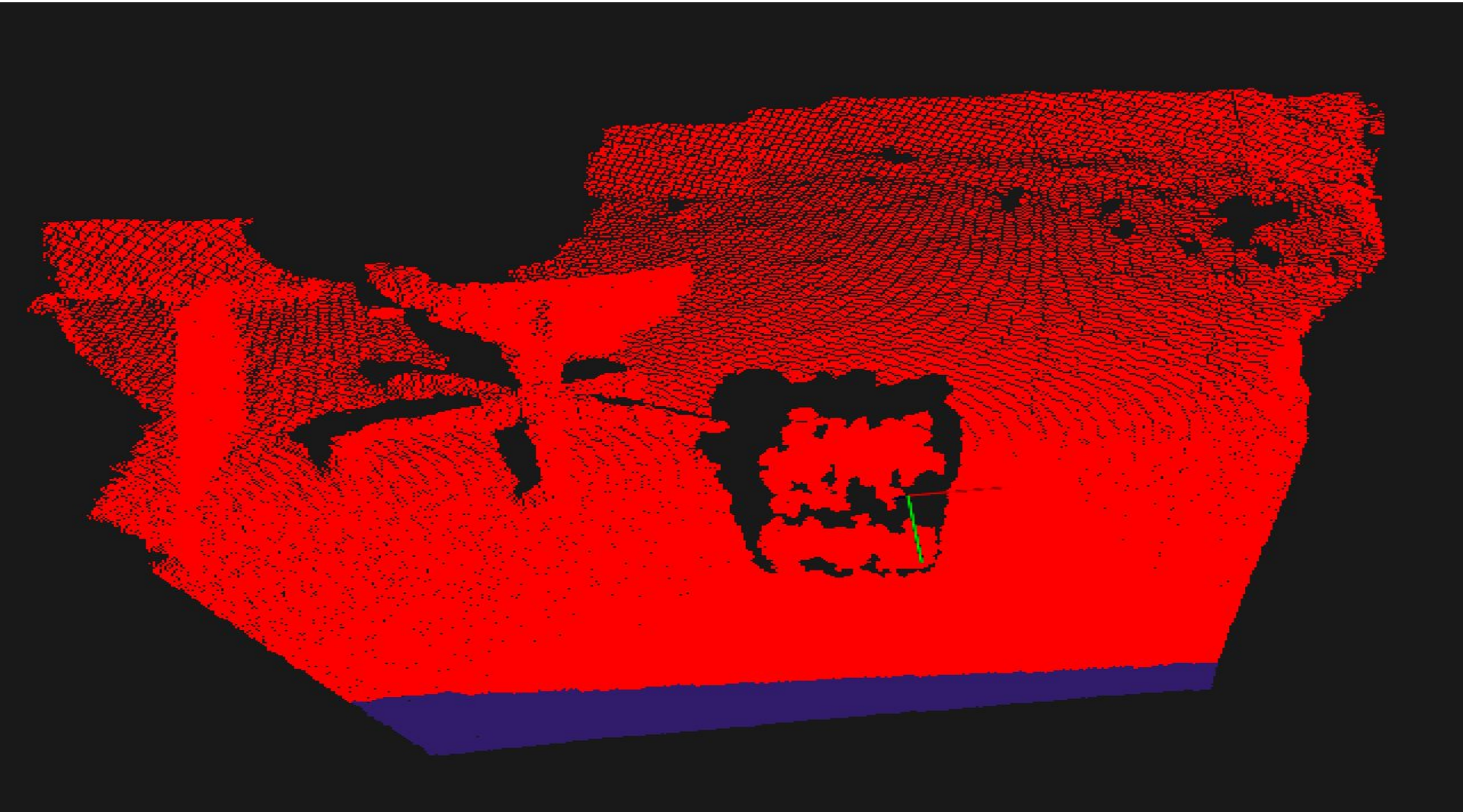
Requirements continued...

- Kdtree- The **pcl_kdtree** library provides the kd-tree data-structure. A means for fast indexing the points in the data cloud.
- RangeImage- RangeImage is derived from pcl/PointCloud and provides functionalities with focus on situations where a 3D scene was captured from a specific view point.

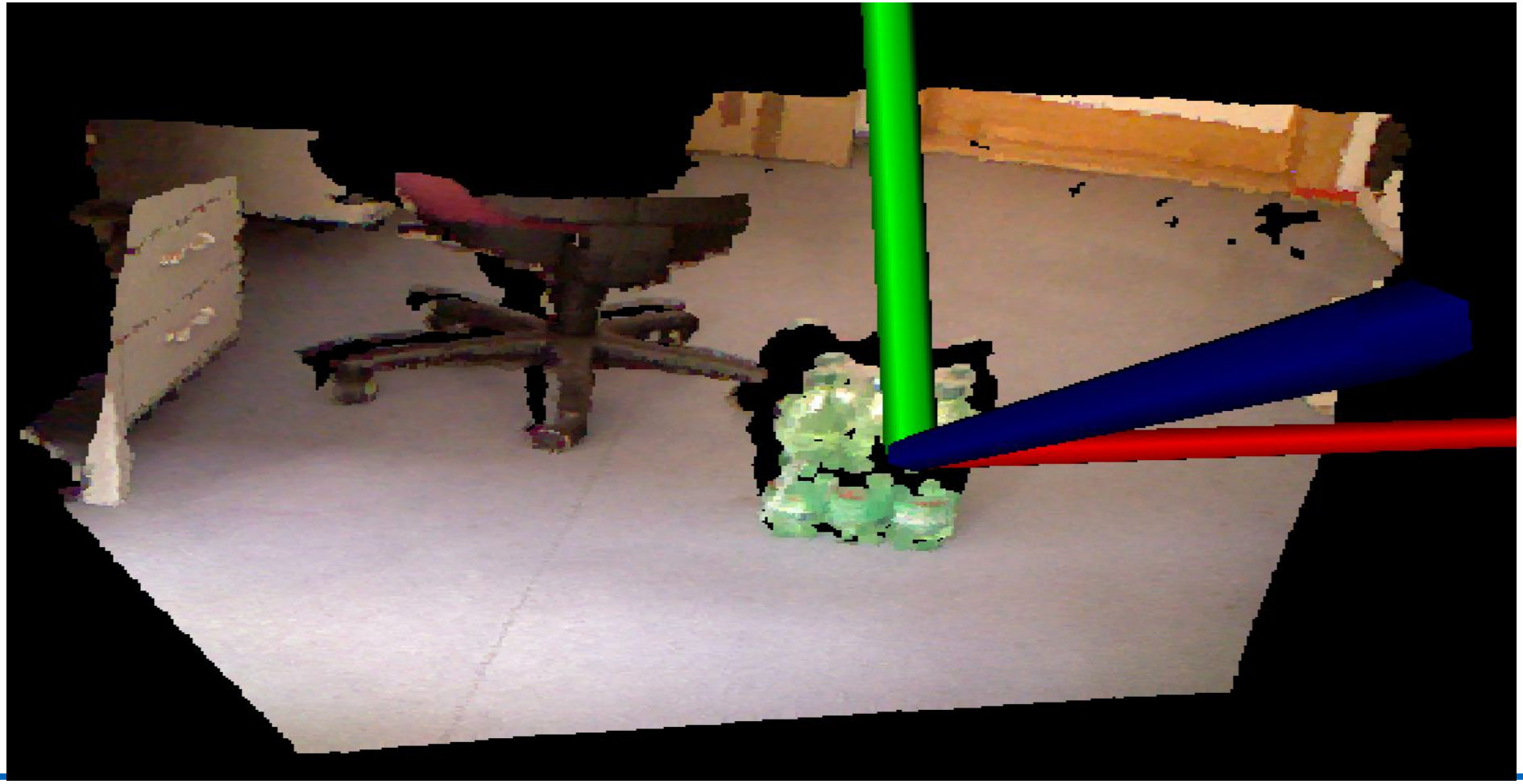
Which class to choose

- Visualization library contains numerous classes to suit our different needs.
- Among the classes available, we have primarily used the PCLVisualizer class for Visualization.
- While more complex to use than the CloudViewer, it is also more powerful, offering features such as displaying normals, drawing shapes and multiple viewports.

Visualization: Using CloudViewer



Visualization: Using PCLVisualizer



File format

- What is PCD?
- A **point cloud** is a set of **data points** in some coordinate system.

Some other file formats:

Some other file formats include:

PLY-

STL-

OBJ-

X3D-

And many others

What is PLY?

PLY is a computer file format known as the **Polygon File Format** or the **Stanford Triangle Format.**

What we do with PLY files

- PCD is not the first file type to support 3D point cloud data.
- So along with reading PCD files, we have also ensured that our program reads ply file formats.
- PCL supports various types of file formats to make it flexible and efficient to use.

How to implement

- To be able to read ply files and visualize them, we needed to use
- PLY Writer class provided in the IO module of PCL.

Polygon File Format (PLY) file reader

- The PLY data format is organized in the following way
- ply
- format [ascii|binary_little_endian|binary_big_endian] 1.0
- element vertex COUNT
- property float x
- property float y
- [property float z]
- [property float normal_x]
- [property float normal_y]
- [property float normal_z]

Polygon File Format (PLY) file reader cont...

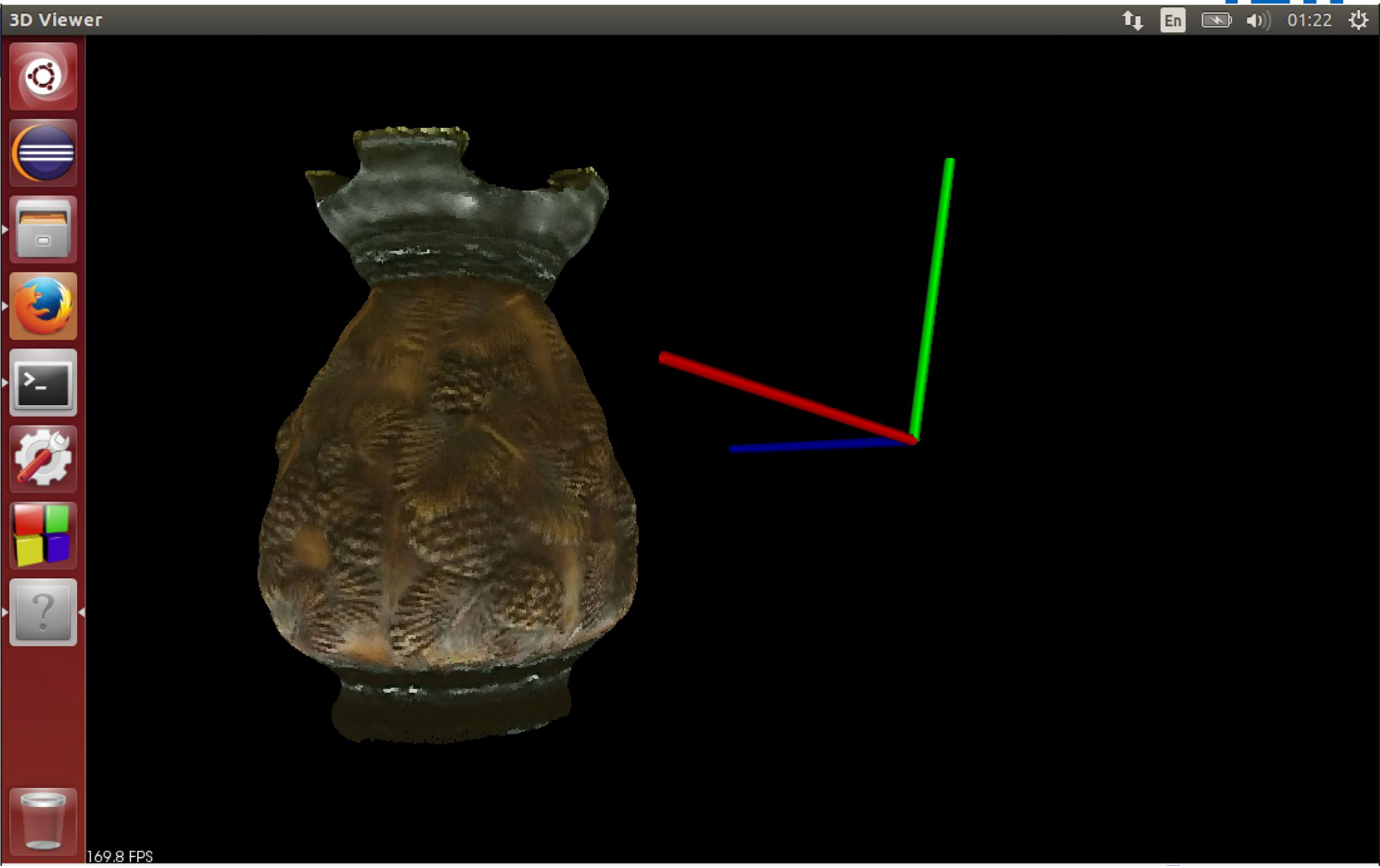
- [property uchar red]
- [property uchar green]
- [property uchar blue] ...
- ascii/binary point coordinates
- [element camera 1]
- [property float view_px] ...
- [element range_grid COUNT]
- [property list uchar int vertex_indices]
- end header

Implementation

- We have tested our code using various ply files taken from http://www.kscan3d.com/gallery/

- And we have used alternate functions compatible with ply data for visualization as used for pcd files.

Example:


3D Viewer
En
01:22
169.8 FPS

How we do it

- What we basically do is use PCLVisualizer but instead of the default file format which is in pcd, we also allow our program to read ply file to make it more compatible and efficient for heterogeneous files

Difficulties faced:

- The problem we faced was with our reader class as mentioned, the class provided was buggy and failed to take input at times
- But what really was time consuming was the lack of resources.
- Very few papers and documentations were available for us to get a good grip on the library modules available.

Difficulties faced:

•We also faced problems while implementing the ply reader class because we had to change all the functions for the algorithms to be compatible with ply file format. Lack of resources made it difficult for us to understand things conceptually.

References:

- http://www.kscan3d.com/gallery/
- http://docs.pointclouds.org/trunk/classpcl_1_1visualization_1_1_p_c_l_visualizer.html
- http://www.pointclouds.org/documentation/tutorials/pcl_visualizer.php