EE 576 Final Project

- Aras Güngöre
- Arif Yılmaz

Introduction

•Objective 1: Implement algorithms for point cloud segmentation, flat face detection, and object tracking.

•Objective 2: Apply the algorithms to RGB-D data for real-time analysis.

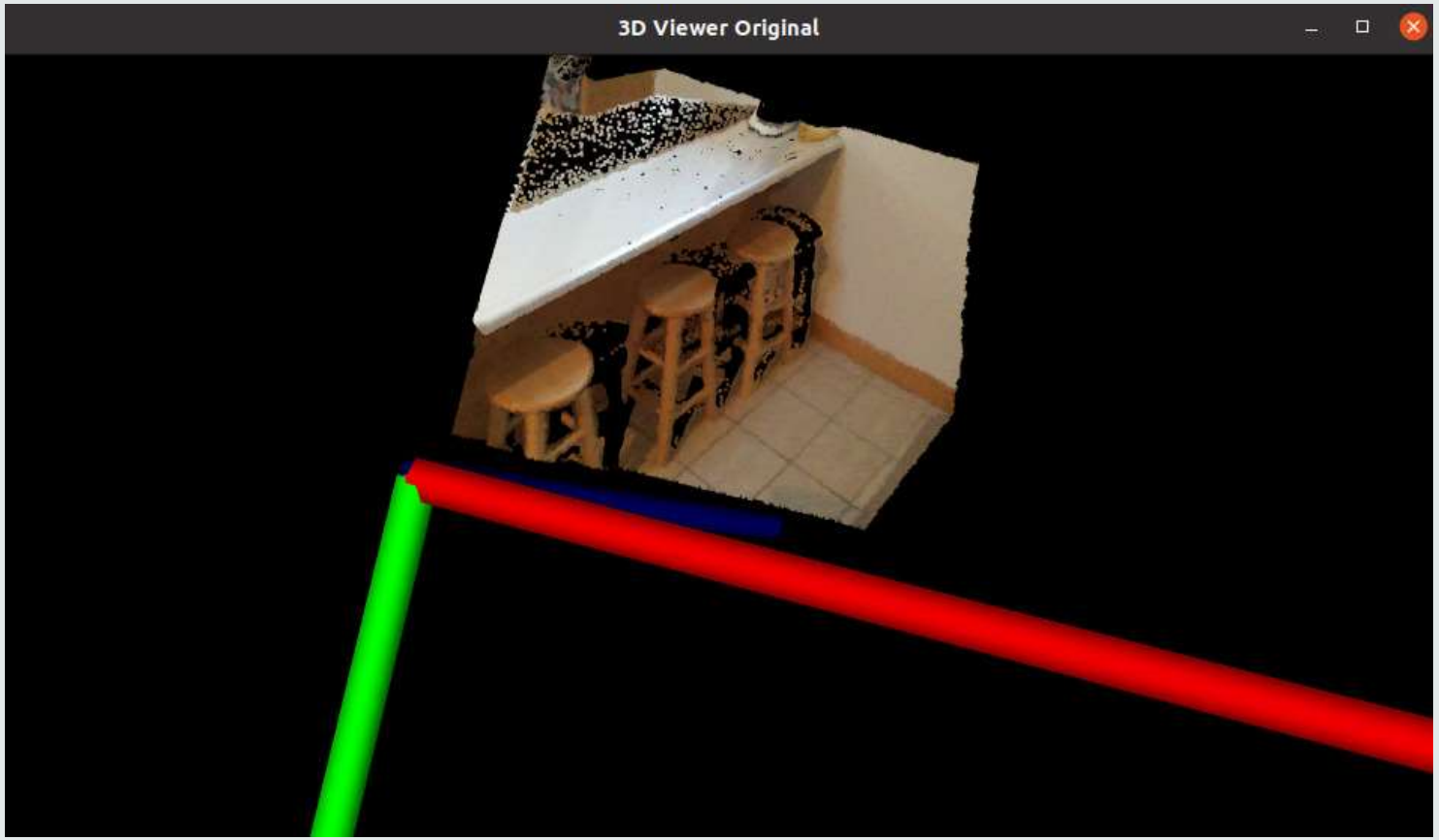
•Objective 3: Develop a visualization module to enhance understanding and analysis.

•Importance: These tasks are crucial for robotics, augmented reality, and autonomous systems.

RGB-D Data Conversion

- Importance: Converting RGB-D data into a point cloud representation.
- 'rgb_depth_to_point_cloud' function:
  - Read and resize RGB and depth images.
  - Inpaint missing depth values for accurate depth information.
  - Convert pixel coordinates to 3D world coordinates using camera intrinsic parameters.

RGB-D Data Conversion



Segmentation using Region Growing

•Importance: Segmenting point clouds aids in understanding object boundaries and relationships.

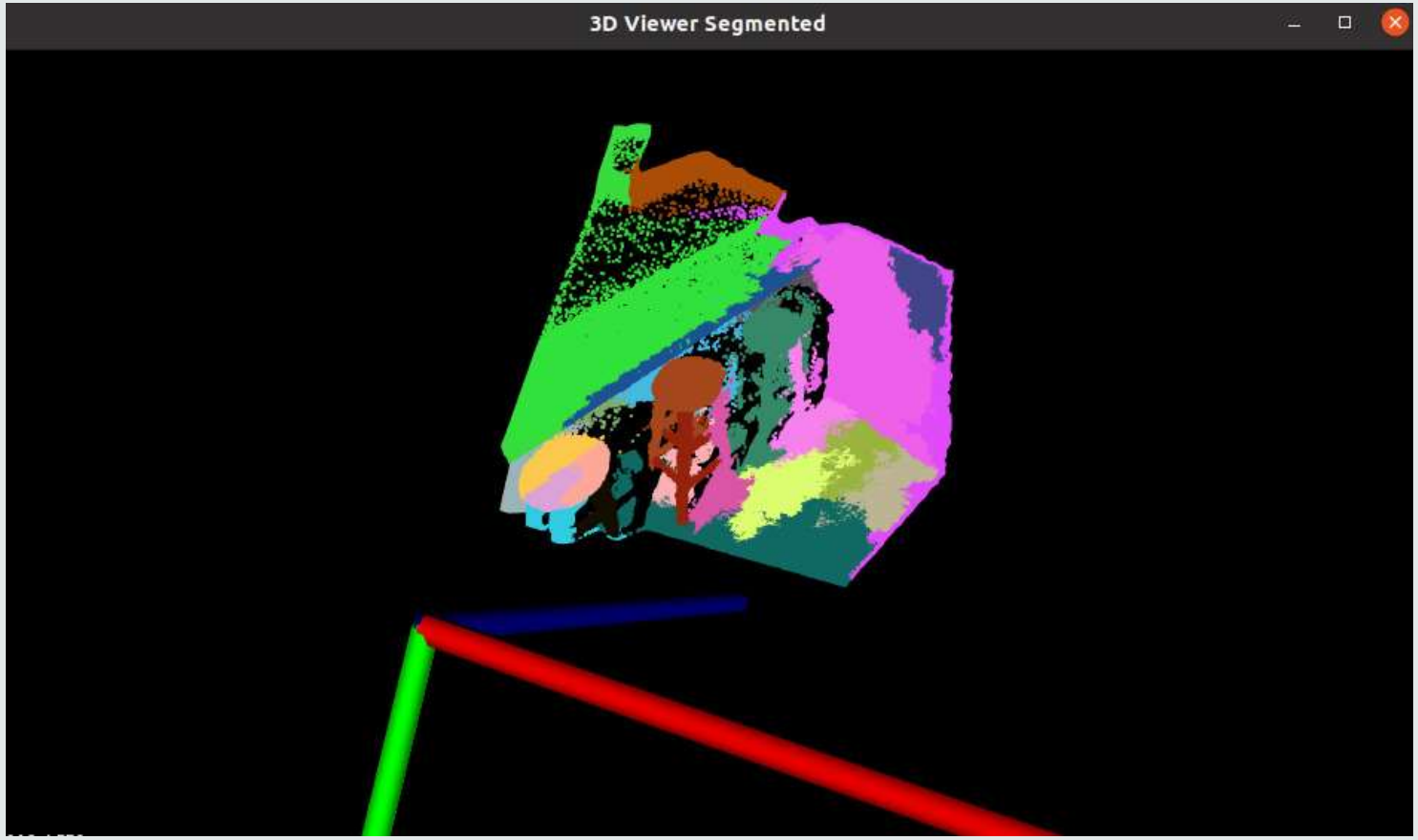
•'rgb_segmentation' function:

- Input: Original point cloud.
- Outputs: Segmented point cloud and cluster indices.

•Utilize the 'pcl::RegionGrowingRGB' class for region growing segmentation:

- Consider RGB color and spatial proximity for segmenting points.
- Adjust parameters (e.g., minimum and maximum cluster size, color threshold) for optimal segmentation.

Segmentation using Region Growing



Finding Horizontally Flat Faces

•Importance: Detecting horizontally flat faces to identify tabletops, shelves, or planar surfaces.

•'find_horizontal_planes' function:

- Inputs: Original point cloud, point cloud of surface normals, segmented clusters.
- Output: Vector of cluster indices with horizontally flat faces.

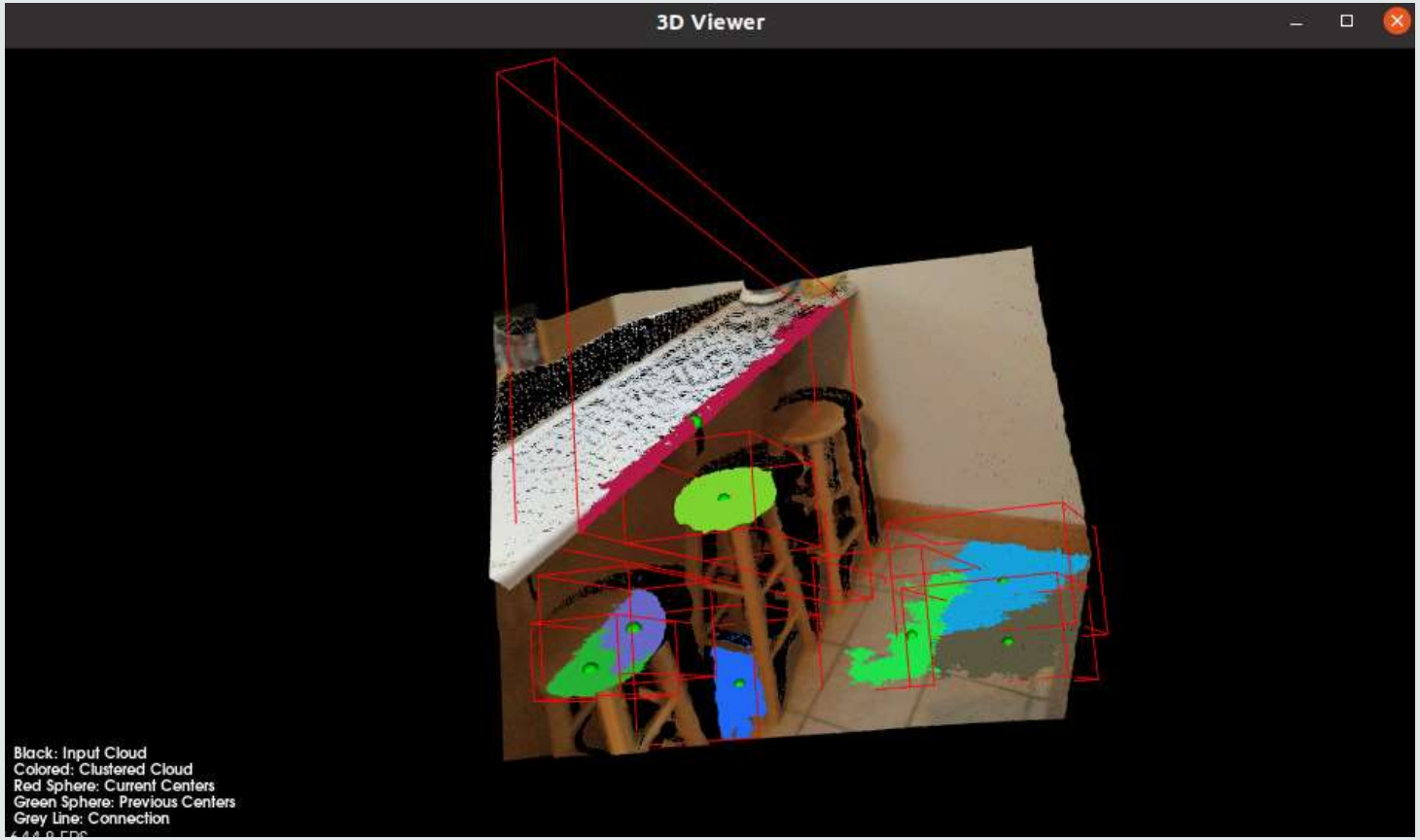
•Calculate average surface normal for each cluster:

- Utilize the 'pcl::NormalEstimation' class to estimate surface normals.
- Determine flat faces using a threshold and fine-tune as necessary.

Visualization of Segmented Clusters

- Importance: Visualizing segmented clusters enhances understanding and analysis.
- 'visualize_clusters' function:
  - Inputs: Original point cloud, segmented clusters, previous and current cluster centers.
- Create a 3D viewer using the PCLVisualizer class:
  - Set up a window and coordinate system.
  - Adjust camera parameters for viewpoint customization.
- Visualization elements:
  - Display segmented clusters with different colors.
  - Show cluster centers as spheres or symbols.
  - Add bounding boxes around clusters for reference.

Visualization of Segmented Clusters



Object Tracking

•Importance: Track object movement across frames.

•Object tracking algorithm in 'visualize_clusters':

- Compare current and previous cluster centers for object displacement.
- Update tracked object centers or add new objects based on distance thresholds.

•Utilize occurrence count for analyzing object frequency and temporal consistency.

Object Tracking