

3-DIMENSIONAL DRONE TOPOGRAPHY (SIMULATION IN ROS)

PROJECT REPORT

EKLAVYA MENTORSHIP PROGRAMME

AT

SOCIETY OF ROBOTICS AND AUTOMATION,
VEERMATA JIJABAI TECHNOLOGICAL INSTITUTE,
MUMBAI

AUGUST-SEPTEMBER 2022

ACKNOWLEDGEMENT

We're extremely grateful to our mentors Jash Shah and Sarrah Bastawala for their support and guidance throughout the duration of the project. We would also like to thank all the members of SRA VJTI for their timely support as well as for organizing Eklavya and giving us a chance to work on this project.

Soham Mulye

<mail-ID>

Unmani Shinde

<mailto:unmani@gmail.com>

TABLE OF CONTENTS

PROJECT OVERVIEW	4
1. INTRODUCTION	5
1.1 DESIGNING A DRONE	5
1.2 NEED FOR SIMULATION	5
1.3 USE OF THE POINT CLOUD LIBRARY	5
1.4 USE OF ROS & GAZEBO	6
2. IN-DEPTH ANALYSIS	7
2.1 OBTAINING POINT CLOUD DATA FROM ROS	7
2.2 SURFACE RECONSTRUCTION.....	7
2.2.1 PRE-PROCESSING	8
2.2.2 FEATURE (NORMAL) ESTIMATION.....	8

PROJECT OVERVIEW

- The traditional problem addressed by surface reconstruction is to recover the digital representation of a physical shape that has been scanned, where the scanned data contains a wide variety of defects. At its core, therefore, surface reconstruction is the process by which a 3D object is inferred, or “reconstructed”, from a collection of discrete points that sample the shape, which is in our case, is obtained from LiDAR Sensors.
- The idea is to have a Drone fly over some terrain in ROS with a GPS and a Depth sensor, then get the point cloud data from the drone and create a 3D map of the topography of the terrain.
- The project was started with the use of ROS to obtain the Point Cloud data of a terrain with the use of LiDAR sensors. The use of Point Cloud Library followed to create 3D Polygonal Meshes and understanding its usage and experimenting with the various algorithms used in PCL for reconstructing meshes from the point clouds.
- Throughout the course of this project, we learnt about several reconstruction techniques that included such as subsampling, up-sampling, estimation of surface normals and algorithm for mesh creation from these calculated normals.
- PCL makes use of many algorithms like Marching Cubes, Grid Projection, Greedy Projection, etc. After experimenting, the project focused on creating the mesh using the Greedy Projection Triangulation algorithm. The data for GPT has been created from the Point Cloud using Moving Least Squares via Maximum Likelihood Estimation.

1. INTRODUCTION


1.1 DESIGNING A DRONE

1.2 NEED FOR SIMULATION

Post the design, testing experimental software on real world hardware consists of several risk factors. Having a good simulation environment is a valuable tool in robotics, as testing on real hardware can often be expensive and time consuming. The actions of some robots can be hazardous, and deploying code (especially in early development) carries risks. Working with real hardware can also introduce issues that impede core algorithm development. For this reason, time spent building an accurate simulation environment is usually well worth it.

1.3 USE OF THE POINT CLOUD LIBRARY

The Point Cloud Library (PCL) is a standalone, large scale, open project for 2D/3D image and point cloud processing. To simplify both usage and development, PCL is split into a series of modular libraries. The set of released modules in PCL that have been heavily utilized in our project have been tabulated as follows:

			
Sr. No.:	Library/Module:	Description:	Mechanism(s) Used:
1.	pcl_io	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.	---
2.	pcl_filters	contains outlier and noise removal mechanisms for 3D point cloud data filtering applications.	Voxel Grid Filter
3.	pcl_features	contains data structures and mechanisms for 3D feature estimation from point cloud data, like estimated curvature and normal at a query point p	Principal Component Analysis, Moving Least Squares

4.	pcl_kdtree	provides the kd-tree data-structure, using FLANN, that allows for fast nearest neighbor searches.	Kd-Tree
5.	pcl_surface	deals with reconstructing the original surfaces from 3D scans, usually a mesh representation or a smoothed/resampled surface with normals.	Greedy Projection, Marching Cubes with RBF

1.4 USE OF ROS & GAZEBO

- Although ROS stands for Robot Operating System, it is not actually an OS, but open-source robotics middleware suite, i.e., a set of common libraries and tools on which to build more complex robotic systems. ROS was designed to be as distributive and modular as possible. ROS packages are easy to make and distribute, which makes it an ideal choice for testing robots.
- Gazebo is a free, open-source robot simulation environment. The project is run by Open Robotics, the same group looking after ROS, however the projects are managed separately and Gazebo is not a “part of” ROS. With Gazebo, we can create a virtual “world”, and load simulated versions of our robots into it. Simulated sensors can detect the environment, and publish the data to the same ROS topics that real sensors would, allowing easy testing of our surface reconstruction algorithms.

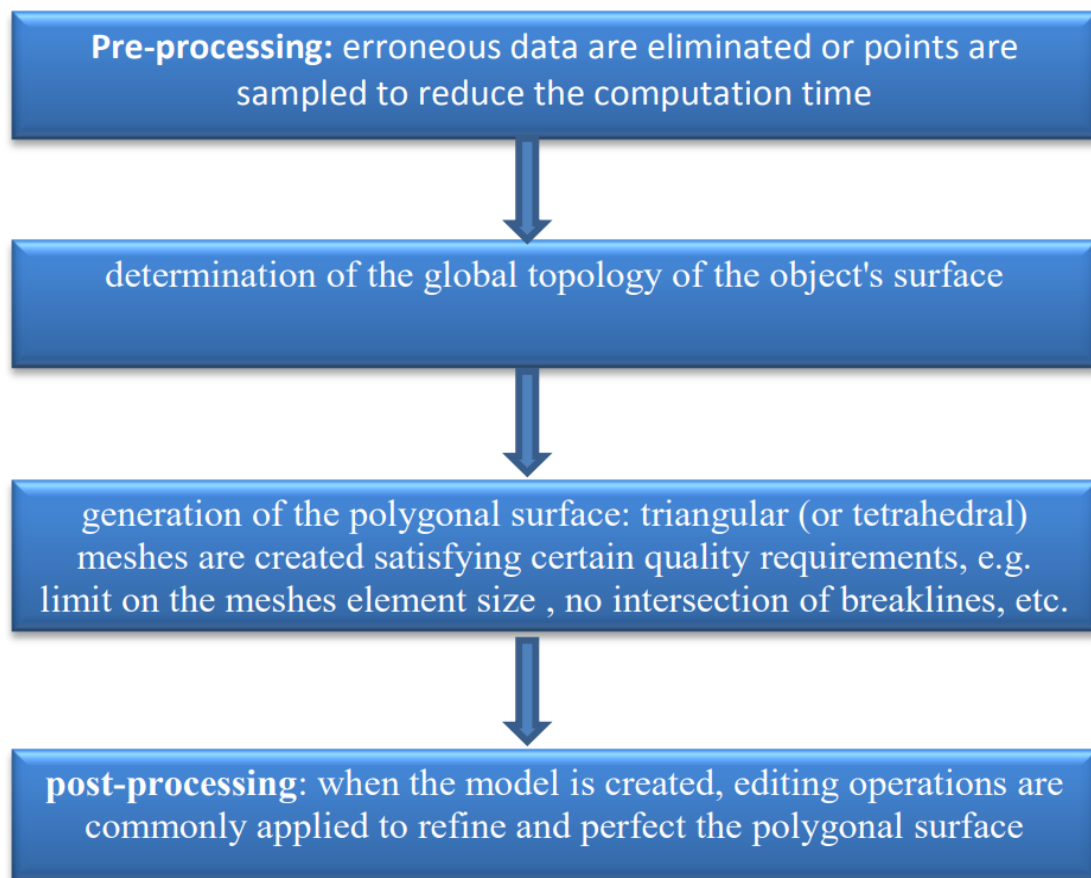
2. IN-DEPTH ANALYSIS

2.1 OBTAINING POINT CLOUD DATA FROM ROS

2.2 SURFACE RECONSTRUCTION

From the depth sensor we obtained an unstructured point cloud representing our terrain, and the final goal is to be able to generate a dense 3D geometry based on this point cloud, i.e., a polygonal (specifically, triangular) mesh which is the commonly accepted graphic representation with the widest support from existing software and hardware.

The general steps in any surface reconstruction algorithm are as follows:



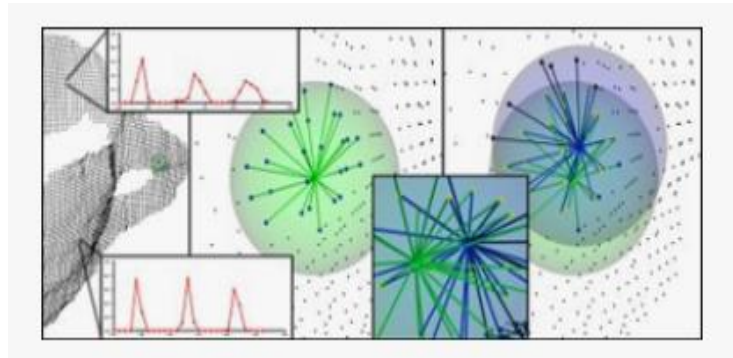
2.2.1 PRE-PROCESSING

The point cloud data obtained might be erroneous due to inaccurate motion of the quadrotor, which have to be eliminated to enhance the accuracy of feature estimation. In addition to inaccurate points, the dataset may also contain some redundant points, which upon being eliminated, significantly reduce the computational time at some minimal expense of accuracy. Such points are eliminated using the **VoxelGrid** Filter Mechanism in PCL.

The `VoxelGrid` class creates a 3D voxel grid (think about a voxel grid as a set of tiny 3D boxes in space) over the input point cloud data. Then, in each voxel, all the points present will be approximated (i.e., downsampled) with their centroid. This approach is a bit slower than approximating them with the center of the voxel, but it represents the underlying surface more accurately.

2.2.2 FEATURE (NORMAL) ESTIMATION

In computer graphics, the surface normals are used for generating the correct shadows based on the lightning. In surface reconstruction, the normals define the direction of the polygon faces. Normal Estimation has been implemented using two



algorithms in this project, which are as described below:

2.2.2.0 KD-TREES & NEAREST NEIGHBOUR SEARCH PROBLEM

2.2.2.1 PRINCIPAL COMPONENT ANALYSIS

- Principal component analysis (PCA) involves a mathematical procedure that transforms a number of correlated variables into a (smaller) number of uncorrelated variables called principal components.
- The first principal component accounts for as much of the variability in the data as possible, and each succeeding component accounts for as much of the remaining variability as possible.
- Normals characterize a point using the information provided by its k closest point neighbors. For determining these neighbors efficiently, the input dataset is usually split into smaller chunks using spatial decomposition techniques (kD-trees in this project, see section 2.2.2.0), and then closest point searches are performed in that space.

- Depending on the application one can opt for either determining a fixed number of k points in the vicinity of p , or all points which are found inside of a sphere of radius r centered at p .
- To estimate the surface normals and curvature changes at a point p is to perform an eigen-decomposition (i.e., compute the eigenvectors and eigenvalues) of the k -neighborhood point surface patch. Thus, the eigenvector corresponding to the smallest eigenvalue will approximate the surface normal n at point p .



2.2.2.2 MOVING LEAST SQUARES WITH MAXIMUM LIKELIHOOD ESTIMATION