3D Data Processing: Point Cloud Registration

Pooya Nasiri *

June 1, 2024

1 Introduction

The objective of this assignment is to implement the Iterative Closest Point (ICP) algorithm to find the fine alignment transformation between a source and a target point cloud. The assignment involved modifying the provided C++ software to complete the ICP main loop, closest point matching, and transformation matrix estimation.

2 Implementation Details

The implementation focused on completing the following methods in the `registration.cpp` file:

2.1 find_closest_point(...)

This method finds the closest points in the target cloud for each point in the source cloud. The KD-Tree data structure from the Point Cloud Library (PCL) is used for efficient nearest neighbor search.

- **KD-Tree Construction:** The target point cloud is used to construct the KD-Tree.
- **Nearest Neighbor Search:** For each point in the source cloud, the nearest neighbor in the target cloud is found using the KD-Tree.
- **Output:** The method returns the indices of the closest points in the target cloud.

2.2 get_svd_icp_registration(...)

This method computes the transformation matrix using Singular Value Decomposition (SVD).

*Email: pooya.nasiri@studenti.unipd.it — Matricola 2071437

- **Centroid Calculation:** The centroids of both the source and target point clouds are calculated.
- **Covariance Matrix:** A covariance matrix is constructed using the centered coordinates of the point clouds.
- **SVD Computation:** SVD is performed on the covariance matrix to obtain the rotation matrix.
- **Translation Vector:** The translation vector is computed using the centroids and the rotation matrix.
- **Transformation Matrix:** The final transformation matrix is assembled from the rotation matrix and translation vector.

2.3 get_lm_icp_registration(...)

This method uses the Levenberg-Marquardt (LM) algorithm, implemented in the Ceres Solver, to optimize the transformation parameters.

- **Cost Function:** A custom cost function is defined to minimize the Euclidean distance between the transformed source points and the target points.
- **Parameter Initialization:** The initial parameters (rotation and translation) are set.
- **Optimization:** The LM optimizer is run to refine the transformation parameters.
- **Output:** The optimized transformation matrix is returned.

2.4 execute_icp_registration(...)

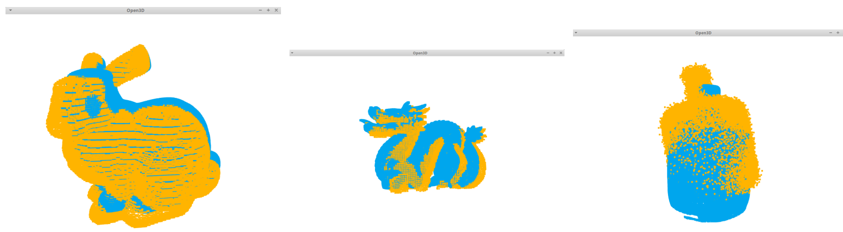
This is the main ICP loop that iteratively calls the closest point matching and transformation estimation methods.

3 Results

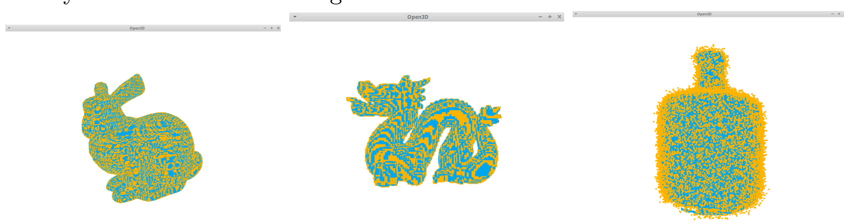
The ICP algorithm was tested on the provided datasets, and the following results were obtained:

3.1 Qualitative Results

Figures 1 show the initial and aligned point clouds for the two datasets. The source cloud is colored in orange, and the target cloud is colored in blue.



(a) Initial point clouds for Bunny. (b) Initial point clouds for Dragon. (c) Initial point clouds for Vase.



(d) Aligned point clouds for Bunny. (e) Aligned point clouds for Dragon. (f) Aligned point clouds for Vase.

Figure 1: Initial and aligned point clouds for Bunny, Dragon, and Vase datasets.

3.2 Quantitative Results

The RMSE values for the aligned point clouds were computed to quantify the accuracy of the registration:

Dataset	Initial RMSE	Final RMSE	Iterations
Bunny	0.045392	0.00341366	20
Vase	0.0714058	0.0162217	28
Dragon	0.0276689	0.00564134	18

Table 1: RMSE reduction and iterations for different datasets

4 Conclusion

The ICP algorithm was successfully implemented, achieving fine alignment between the source and target point clouds. The qualitative results visually confirm the alignment, and the quantitative RMSE values provide a measure of the accuracy.