COMP0119: Acquisition and Processing of 3D Geometry

Coursework 1: Iterative Closest Point

This report will demonstrate the algorithm implemented for the Iterative Closest Point.

Task 1: Point-to-Point Alignment

The basic ICP algorithm can be a breakdown of following steps:

1. Select source points
2. Match points in the other mesh
3. Reject bad pairs
4. Compute rigid transform
5. Detect error and check if stop iteration

The first task requires us to align M2 to M1 using such an algorithm. After loading two OFF meshes which are converted from PLY using MeshLab, we perform the KNN search for M2 to find nearest neighbour points on M1 using all available vertex (subsample will be utilised later in task 4) and then reject distant vertex. After this step we obtain a refined version of M2 which contains matched vertex that are 1-to-1 mapping to the M1. Once we have the mapping, we start calculating the rigid transform which contains the rotation R and translation T.

By applying rigid transform to M2, the position and rotation of it is updated. Then we repeat the steps above until we reach the reasonably accurate alignment.

The results are provided below:

A picture containing yellow, indoor, orange, sky

Description automatically generated A picture containing yellow, artichoke, sky, orange

Description automatically generated

*Result at 10th and 50th iteration*

A picture containing sky

Description automatically generated A picture containing yellow, sky

Description automatically generated

*Result at 100th iteration with non-overlapping area marked out and at 150th iteration*

The results are…

Task 2: Rotation Matching

This task requires to produce a rotated version of M1, which can be simply done with the GUI interaction. The GUI provides three text fields for x, y, z axis rotation in degree, and our goal is to check how well the algorithm can handle different degrees of misalignments. For ease of demonstration, the model used for this task is “bun000.off” and the initial rotation degrees for x, y, z axis are 0, 0, 0 respectively. The result is provided below:

From the result, it can be noticed that the iterations required for a perfect alignment increases almost at the same as the rotation degrees increases either clockwise or anticlockwise.

Task 3: Adding Noise

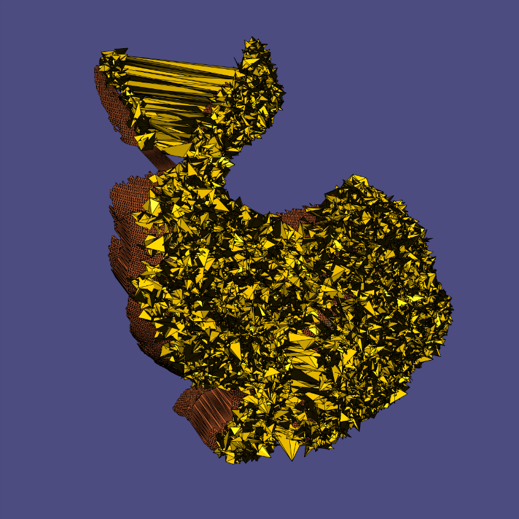
This task requires us to add zero-mean Gaussian noise to M2 and use the noised M2’ to match M1 using ICP algorithm. The standard deviation is used to obtain the overall noise level and it will be scaled and distributed to each axis based on the bounding box dimension.

A picture containing sky, holding, indoor

Description automatically generated A close up of a yellow wall

Description automatically generated with low confidence

*Noised M2 and matched M2 (sd = 0.5)*



*Matched M2 (sd = 3.0)*

|  |  |  |
| --- | --- | --- |
| Standard Deviation | Aligned with M1 | Iterations Taken |
| 0.1 | Yes | 92 |
| 0.25 | Yes | 93 |
| 0.5 | Yes | 88 |
| 1.0 | Yes | 86 |
| 2.0 | Yes | 85 |
| 3.0 | Yes |  |
| 4.0 | Yes |  |

The result is

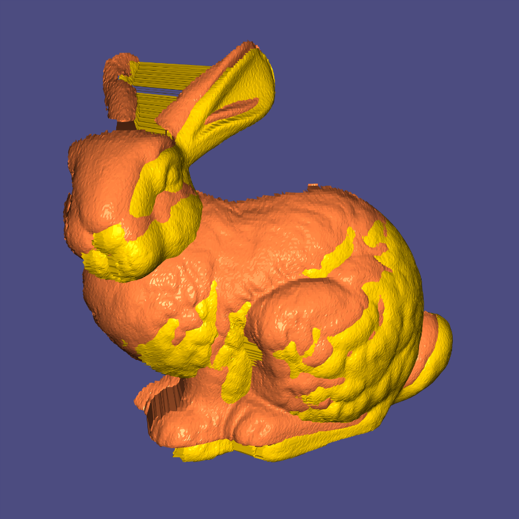
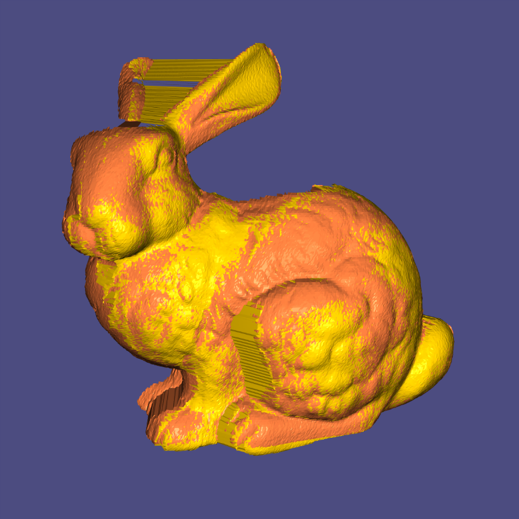
Task 4: Subsampling

bla

Task 5: Multiple Meshes Alignment

bla

Task 6: Point-to-Plane Alignment

Result at 5th an 10th iteration

Reference

Gelfand. N. et al. (2003). Geometrically Stable Sampling for the ICP Algorithm, *Forth International Conference on 3-D Digital Imaging and Modelling*, 260-267. doi: 10.1109/IM.2003.1240258

Low, K. (2004). Linear Least-Squares Optimization for Point-to-Plane ICP Surface Registration. Retrieved from <https://www.comp.nus.edu.sg/~lowkl/publications/lowk_point-to-plane_icp_techrep.pdf>

Planck Institut Informatik. (n.d.). Pairwise, Rigid Registration: The ICP Algorithm and Its Variants. Max Retrieved from <http://resources.mpi-inf.mpg.de/deformableShapeMatching/EG2011_Tutorial/slides/2.1%20Rigid%20ICP.pdf>