Assignment 1

Xiyao LI

January 30, 2023

1 Introduction

In this coursework, the questions below have been treated:

- Implemented point-to-point ICP baseline and run it to align *bunny_045* to *bunny_00*
- Analysed how the rotation influences the performance of the algorithm and tested on a range of rotation angles varying from 0 to 360°
- Measured how the noise impacts ICP and determined its performance based on the alignment error
- Tested how the sub-sampling rate changes the alignment error on a range of rates
- Found a solution to align multiple models together
- Optimised ICP with point-to-plane distance and tested it on a noisy model

2 Basic ICP Algorithm

Assume that mesh M1 is the target mesh and mesh M2 is the source mesh to be transformed. The algorithm can be summarised as below:

For 50 iterations:

1. Sample randomly 500 vertices $\mathbf{p_i}$ from M2
2. For each sample vertex, match it to the closet vertex $\mathbf{q_i}$ from M1 using a Kd-Tree structure
3. Reject 10% matched pairs by distance
4. Compute the rotation matrix $\mathbf{R}$ and translate matrix $\mathbf{t}$ with the least square minimisation
5. Apply rigid transformations to M2

The alignment error is defined as the sum of distance between vertex pairs:

$$E(\mathbf{R}, \mathbf{t}) = \sum_i ||(\mathbf{R}\mathbf{p_i} + \mathbf{t} - \mathbf{q_i})||^2 \tag{1}$$

The alignment error is saved for each iterations. The error and alignment result from basic ICP routine are shown below.

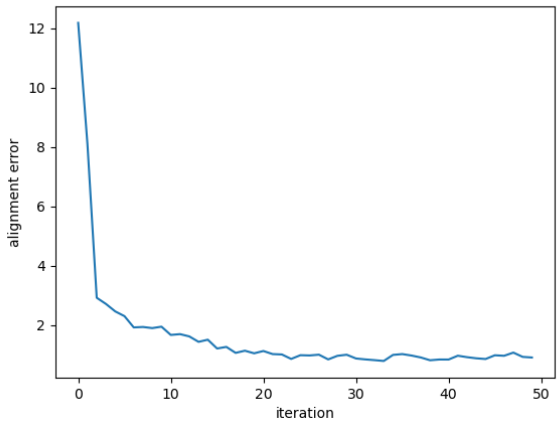


Figure 1: Alignment error for basic ICP routine

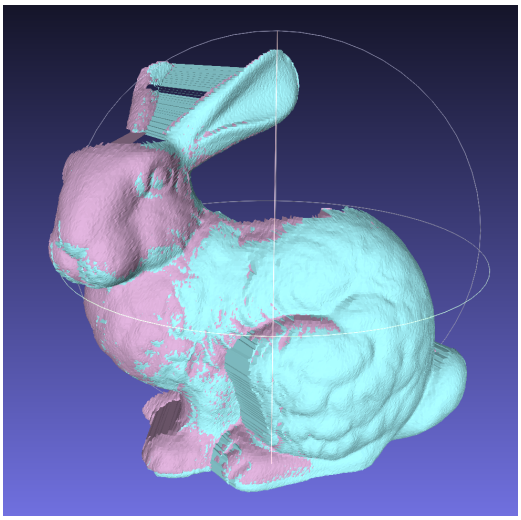


Figure 2: Alignment from bunny_045 to bunny_00

If we optimise equation (1) with an associated weight wi, then the error can be written as:

$$E(\mathbf{R},\mathbf{t}) = \sum_i \omega_i ||(\mathbf{R}\mathbf{p_i} + \mathbf{t} - \mathbf{q_i})||^2 \tag{2}$$

The minimisation of error function is equivalent to:

$$\underset{\beta}{\operatorname{argmin}} \sum_i \omega_i ||(\sum_j p_{ij}\beta_j - \mathbf{q_i})||^2 \implies \underset{A}{\operatorname{argmin}} \underbrace{||\mathbf{W}^{\frac{1}{2}}(\mathbf{PA} - \mathbf{Q})||^2}_{E(\mathbf{A})}$$

with W the diagonal matrix of i-th weight in i-th row.

$$\begin{aligned} E(\mathbf{A}) &= (\mathbf{PA} - \mathbf{Q})^T (\mathbf{W}^{\frac{1}{2}})^T \mathbf{W}^{\frac{1}{2}} (\mathbf{PA} - \mathbf{Q}) \\ &= (\mathbf{A}^T\mathbf{P}^T - \mathbf{Q}^T)\mathbf{W}(\mathbf{PA} - \mathbf{Q}) \\ &= \mathbf{A}^T\mathbf{P}^T\mathbf{WPA} - \mathbf{A}^T\mathbf{P}^T\mathbf{WQ} - \mathbf{Q}^T\mathbf{WPA} + \mathbf{Q}^T\mathbf{WQ} \end{aligned} \tag{3}$$

Derive equation (3):

$$\frac{\partial E(\mathbf{A})}{\partial \mathbf{A}} = \mathbf{P}^T\mathbf{WPA} - \mathbf{P}^T\mathbf{WQ}$$

The expression of $\mathbf{A}$ for $E(\mathbf{A})$ to reach its minimum is then:

$$\mathbf{A} = (\mathbf{P}^T\mathbf{WP})^{-1}\mathbf{P}^T\mathbf{WQ}$$

3 Rotated Models

To measure how the rotation impacts ICP performance, we pick a range of rotation angles from 0 to 360° with a step size of 45° and measure the alignment error after 100 iterations. From the following figure, we conclude that a rotation angle less than 45° leads to good ICP performance, and a rotation angle more than 45° disables the algorithm. It is also why we need to assume models are nearly aligned as a given condition.

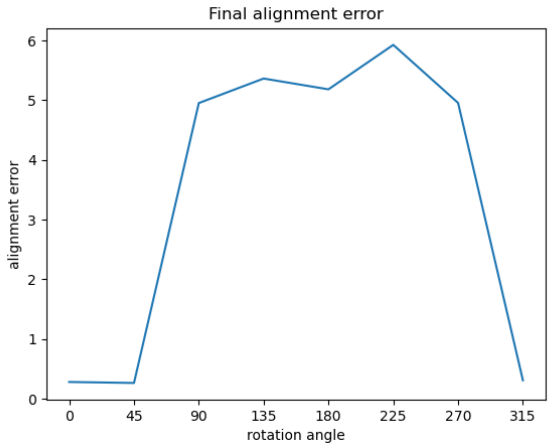


Figure 3: Final alignment error for corresponding rotation angle

Here are examples of ICP alignment from a rotation angle $= 5, 15, 45$ and 90°. The algorithm always converges after around 60 iterations, so the rotation angle does not impact the converging time.

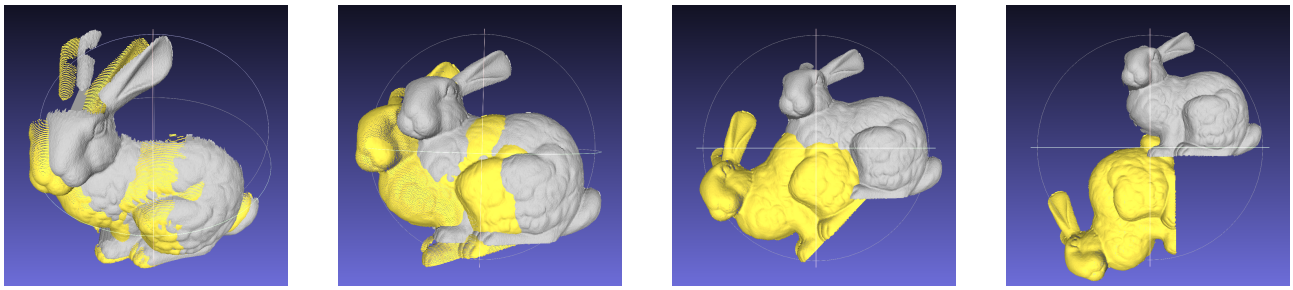


Figure 4: Meshes rotated along z-axis (5°, 15°, 45°, 90° from left to right)

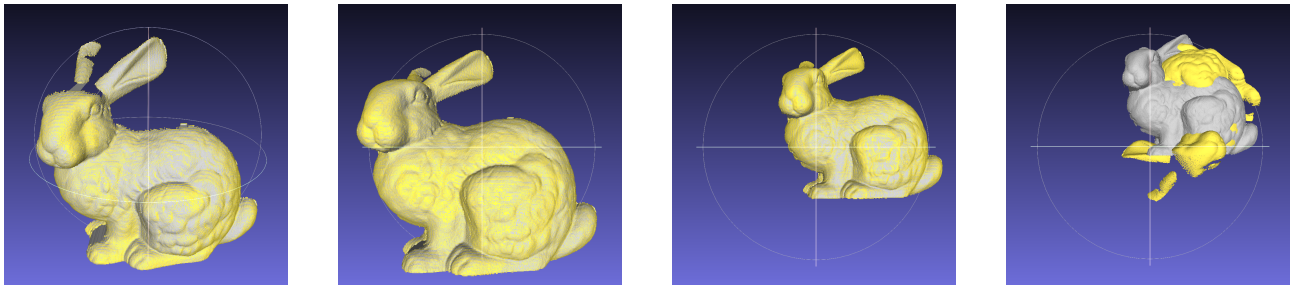


Figure 5: Alignment results

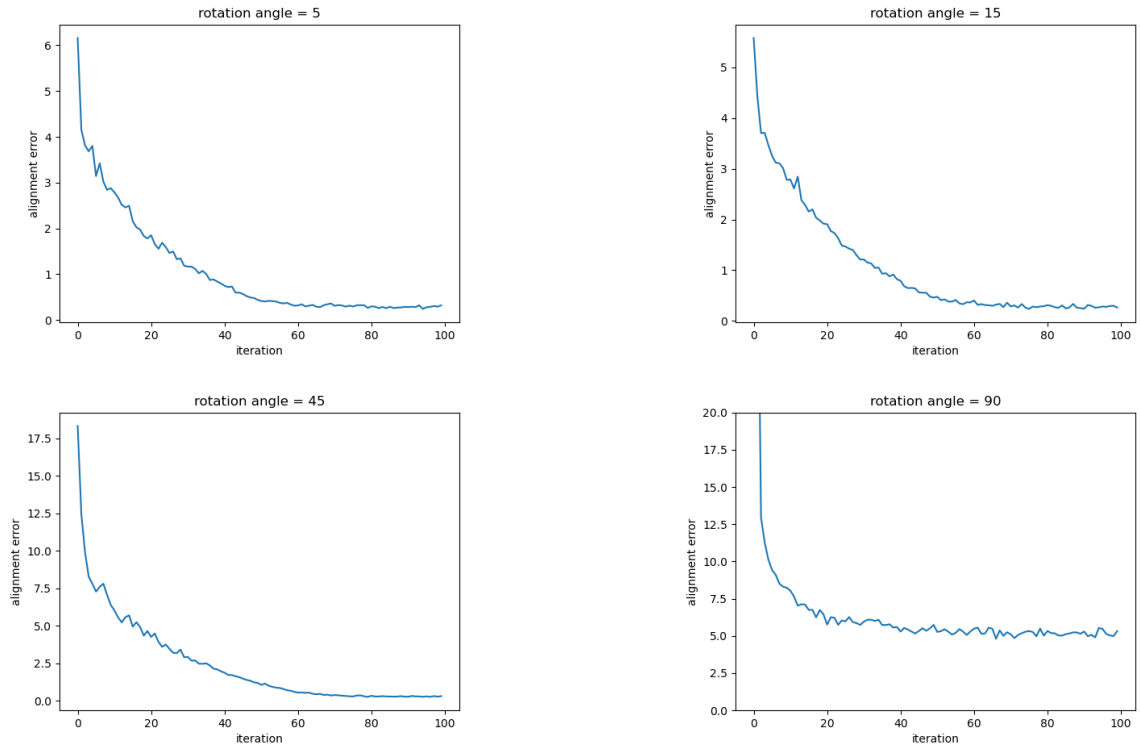


Figure 6: Alignment errors

4 Noisy Models

To determine how noise influences ICP performance, we add Gaussian noise with zero-mean and standard deviation $= 0.0001, 0.001, 0.01$ and 0.1 to mesh 2. Here we choose *bunny_045* as mesh 2 and align it to mesh 1, *bunny_00*. We run the algorithm with 50 iterations and save its final error. From the following figure, we conclude that increasing noise leads to increasing alignment error.

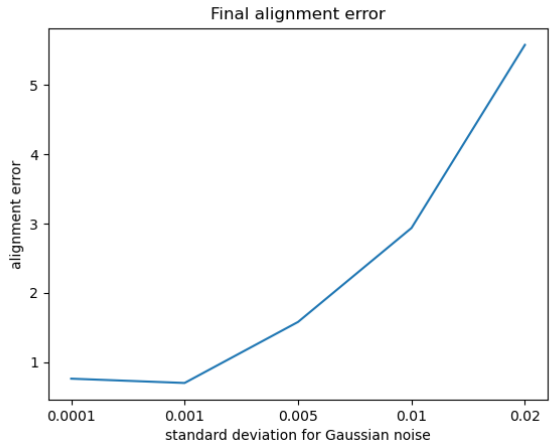


Figure 7: Final alignment error for noisy meshes

Here are examples of ICP alignment applied to noisy meshes with standard deviation $= 0.0001, 0.001$ and 0.01. For standard deviation $= 0.0001$ and 0.001, the algorithm always

converges around 30 iterations, while for standard deviation = 0.01, the algorithm converges after 10 iterations. Due to the high noise level, the algorithm may not find a lower alignment error.

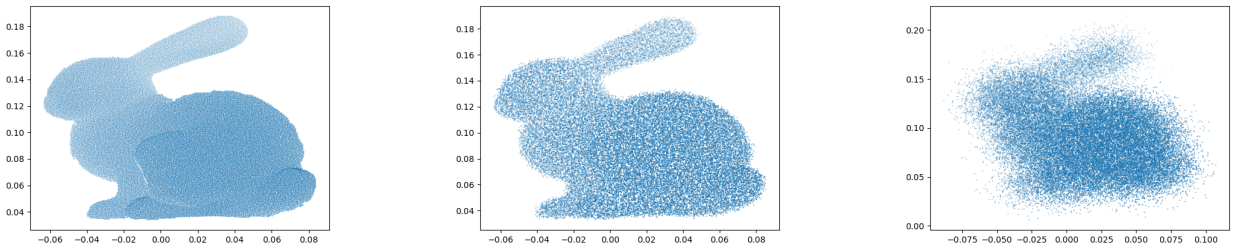


Figure 8: Point cloud of noisy meshes with standard deviation = $0.0001, 0.001$ and 0.01

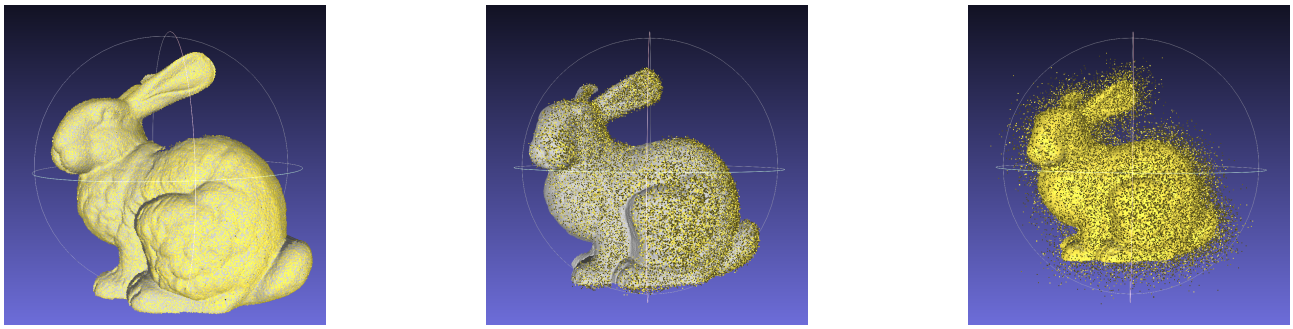


Figure 9: Alignment results (noisy models represented in point cloud)

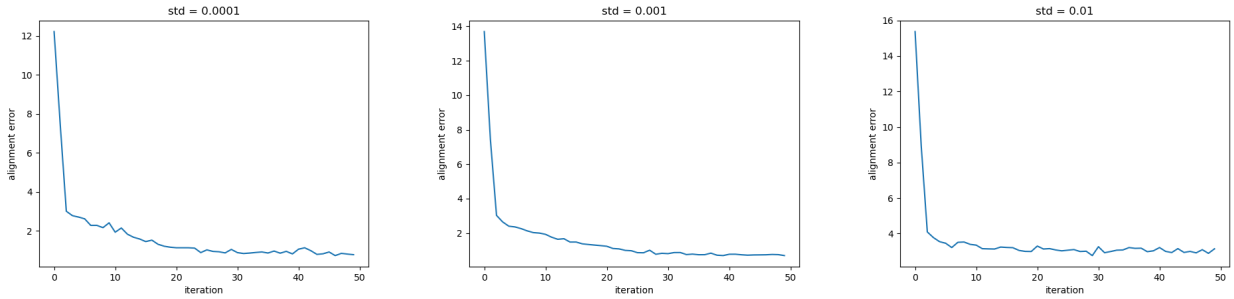


Figure 10: Alignment errors

5 Sub-sampling Rates

To measure the influence of sampling rate on ICP alignment, we use the mean distance between each pair instead of the sum we used in previous sections. We choose sampling rates $= 0.001, 0.005, 0.01, 0.05, 0.1$ and 1 to measure the alignment error and execution time for 50 iterations.

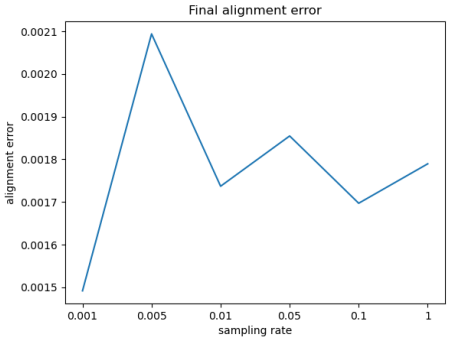


Figure 11: Mean alignment error

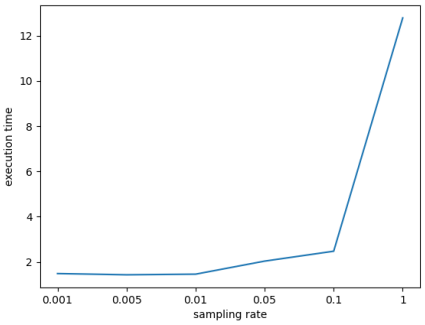


Figure 12: Execution time

We observe that the mean error is relatively similar and not seems to depend on the sampling rate. However, the execution time increases with the sampling rate. In this case, the models have around 40 000 vertices. For sampling rate $= 0.1$, the run time is 2 seconds. And if we use all vertices, the run time increases to 14 seconds. We also notice whatever the rate, the algorithm always converges after 30 iterations. In conclusion, the sampling rate does not influence the converging time of alignment error. In the case of a model with 40 000 vertices, a sampling rate of 0.001 is enough to get good performance.

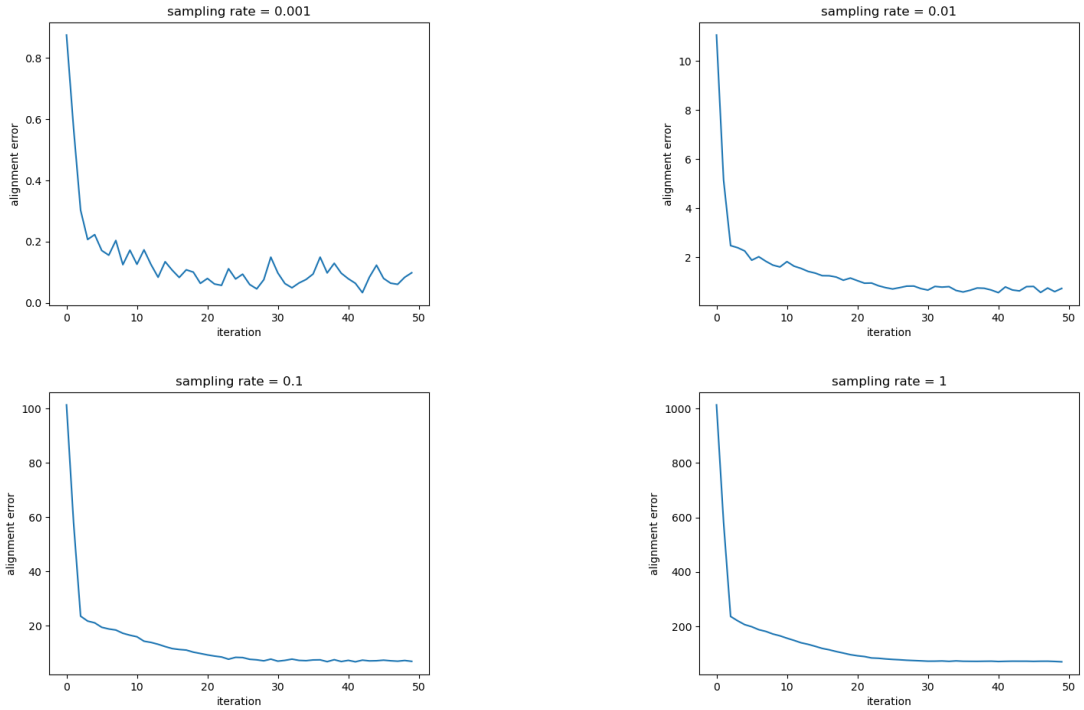


Figure 13: Alignment errors for corresponding sampling rates

6 Multiple Models Alignment

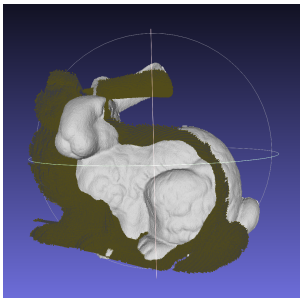
In Section 3, we proved that the algorithm gets satisfying results only when the rotation angle is under 45°. Therefore, the main idea of aligning several models is to privilege models with smaller rotation angle differences and rotate one model no more than 45° at each time.
For example, we choose *bunny_00*, *bunny_45*, *bunny_90*, *bunny_270* and *bunny_315*, because we can always find an rotation angle of 45° between two models from this choice. We align all to *bunny_00*, some model requires only one rotation while some needs two. The alignment process is shown in Figure 15.

- Pros:

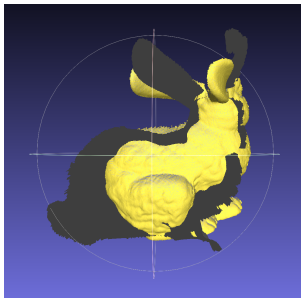
- As *bunny_00* is the closest to the average of all models, using it as a benchmark, the solution uses a minimum overall number of alignments.

- Cons:

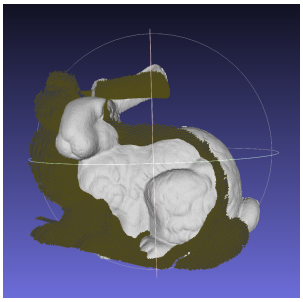
- The rotation angle is needed beforehand to decide which models to align together.
- The algorithm only works for small rotation angles ($\leq 45°$), or we may need to rotate the model before applying ICP.
- The algorithm only works for two similar models. A counter-example is aligning *bunny_180* to *bunny_00*. *bunny_180* is the back part of the bunny, while *bunny_00* is the front part of the bunny. In figure 14, it can be seen that the algorithm is disabled because of the significant difference between models and their rotation angles.



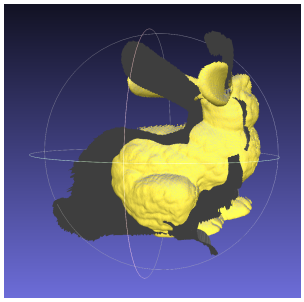
View from the front



View from the back

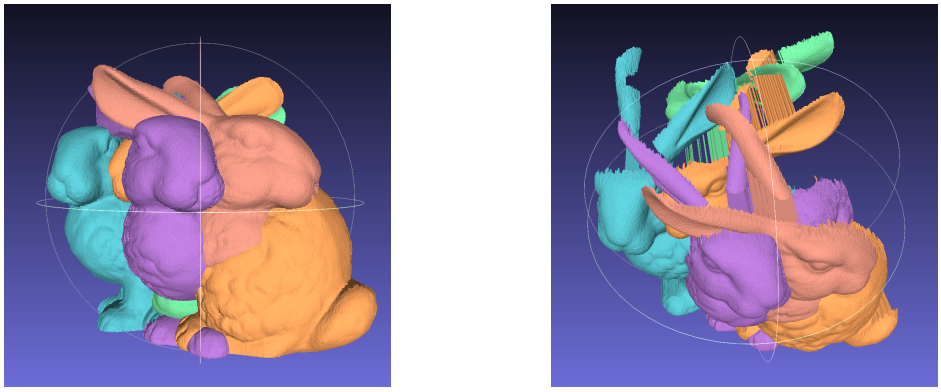


After ICP from the front

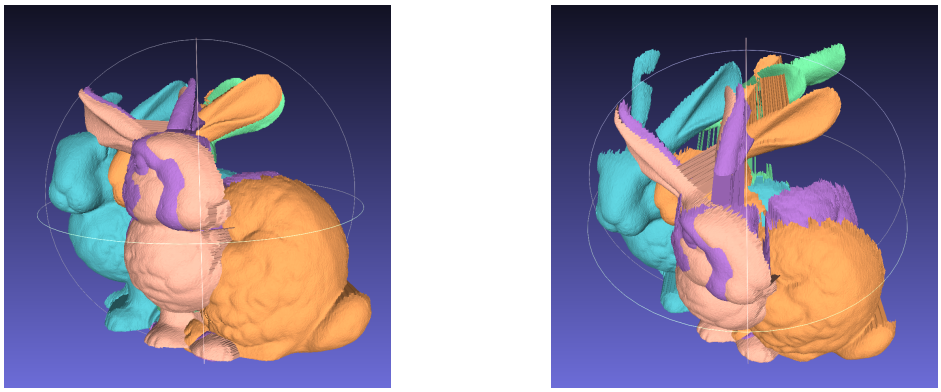


After ICP from the back

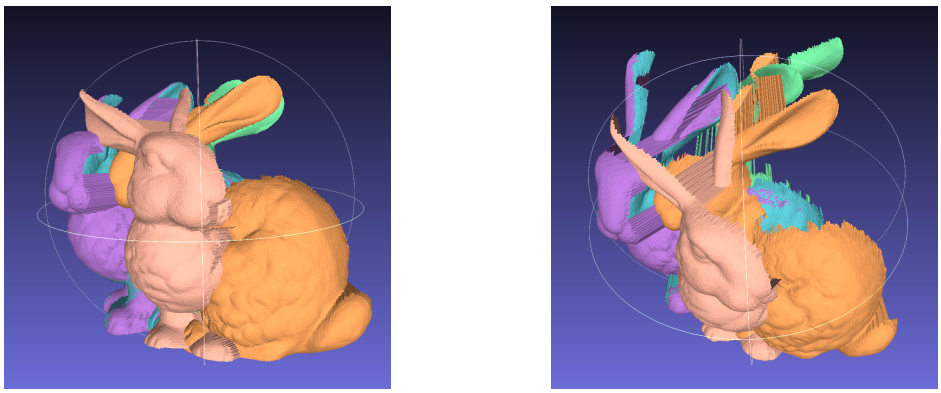
Figure 14: Align *bunny_180* to *bunny_00*



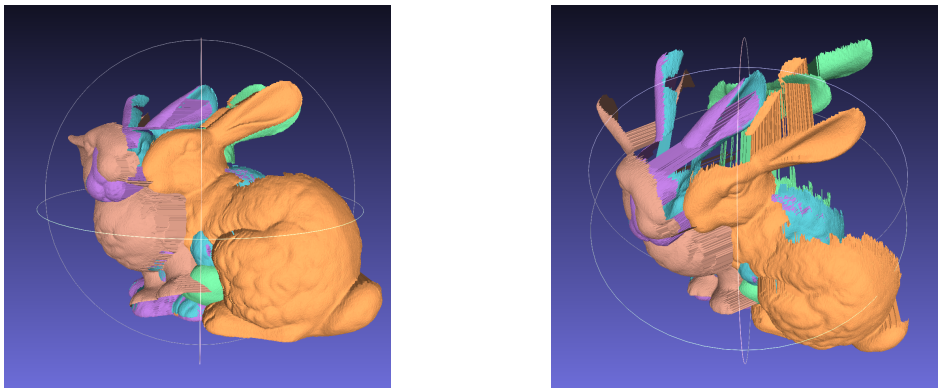
Original positions



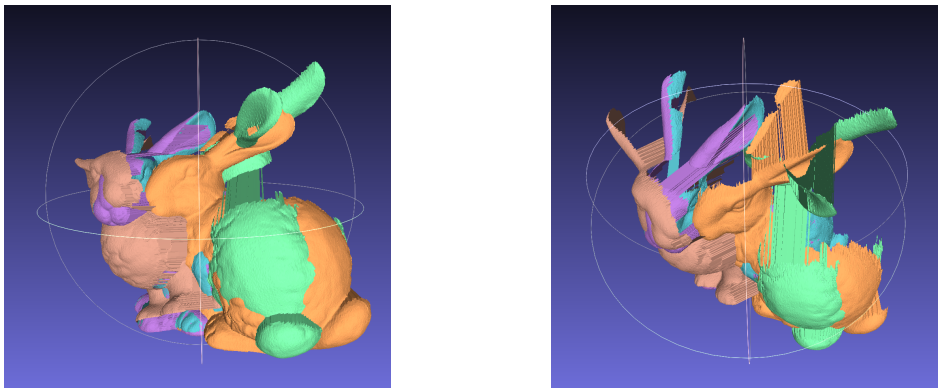
Step1: Align *bunny_270* (pink) to *bunny_315* (purple)



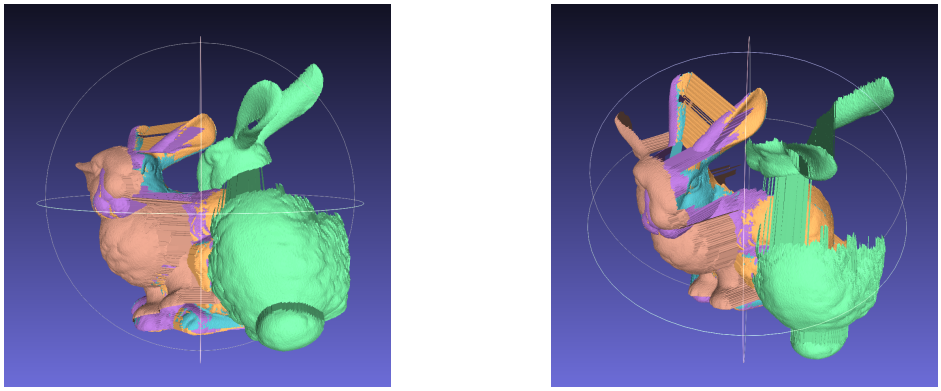
Step2: Align *bunny_315* (purple) to *bunny_00* (blue)



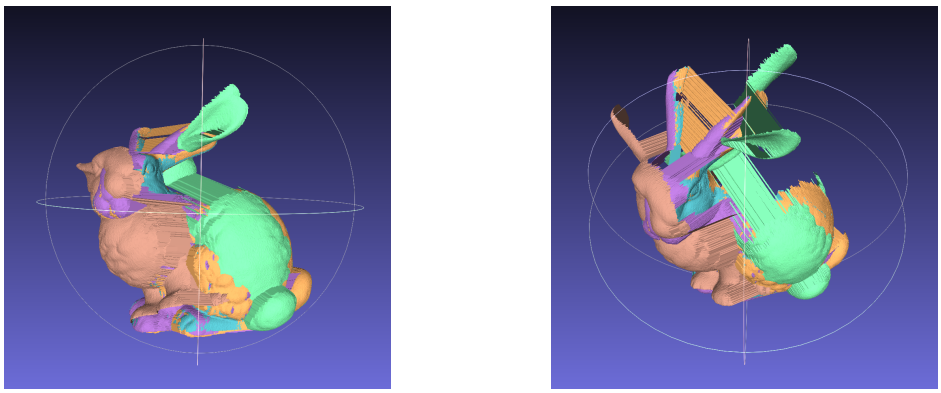
Step3: Align *bunny_270* (pink) to *bunny_315* (purple)



Step4: Align *bunny_90* (green) to *bunny_45* (orange)



Step5: Align *bunny_45* (orange) to *bunny_00* (blue)



Step6: Align *bunny_90* (green) to *bunny_45* (orange)

Figure 15: Alignment process

7 Optimised Point-to-plane ICP

Compared to the baseline described in section 2, the optimisation measures the point-to-plane distance instead of point-to-point. More precisely, we compute firstly the normal of vertices for each sampled point using least square plane fitting. For a point $\mathbf{p_i}$ from M2, we determine a plane from its normal. Then, we measure the distance between its nearest neighbour $\mathbf{q_i}$ from M2 and the plane generated by its normal. And the alignment error equation (1) changes to:

$$E(\mathbf{R}, \mathbf{t}) = \sum_i ||(\mathbf{R}\mathbf{p_i} + \mathbf{t} - \mathbf{q_i}) \cdot \mathbf{n_i^q}||^2 \tag{4}$$

The minimisation of equation (4) is implemented based on Kok-Lim Low's paper[1]. We build matrix $\mathbf{A}$ and vector $\mathbf{b}$ using pointsets from M1, M2 and normals of M1:

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & a_{13} & n_{1x} & n_{1y} & n_{1z} \\ a_{21} & a_{22} & a_{23} & n_{2x} & n_{2y} & n_{2z} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{N1} & a_{N2} & a_{N3} & n_{Nx} & n_{Ny} & n_{Nz} \end{pmatrix}$$

with

$$\begin{aligned} a_{i1} &= n_{iz}p_{iy} - n_{iy}p_{iz} \\ a_{i2} &= n_{ix}p_{iz} - n_{iz}p_{ix} \\ a_{i3} &= n_{iy}p_{ix} - n_{ix}p_{iy} \end{aligned}$$

and

$$\mathbf{b} = \begin{pmatrix} n_{1x}q_{1x} + n_{1y}q_{1y} + n_{1z}q_{1z} - n_{1x}p_{1x} - n_{1y}p_{1y} - n_{1z}p_{1z} \\ n_{2x}q_{2x} + n_{2y}q_{2y} + n_{2z}q_{2z} - n_{2x}p_{2x} - n_{2y}p_{2y} - n_{2z}p_{2z} \\ \vdots \\ n_{Nx}q_{Nx} + n_{Ny}q_{Ny} + n_{Nz}q_{Nz} - n_{Nx}p_{Nx} - n_{Ny}p_{Ny} - n_{Nz}p_{Nz} \end{pmatrix}$$

By solving $\mathbf{Ax} = \mathbf{b}$, we obtain $\mathbf{x} = (\alpha, \beta, \gamma, t_x, t_y, t_z)$ and the transform matrix is formed as:

$$\mathbf{T} = \begin{pmatrix} 1 & \alpha\beta - \gamma & \alpha\gamma + \beta & t_x \\ \gamma & \alpha\beta\gamma + 1 & \beta\gamma - \alpha & t_y \\ -\beta & \alpha & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

The point-to-plane ICP allows the algorithm to handle cases where the surfaces are non-planar or have significant curvature. This results in a more accurate alignment. The alignment error curve shown in Figure 16 is smoother, and the error value is lower than that using basic ICP. Additionally, point-to-plane ICP is less sensitive to outliers, making it more robust to noise in the data. Figure 19 is an example of adding a Gaussian noisy $\sim \mathcal{N}(0, 0.01)$ to model M2 (Figure 17) and align it to M1, the converging error value using point-to-plane ICP is smaller than that using point-to-point ICP.

References

[1] Kok-Lim Low. Linear least-squares optimization for point-to-plane icp surface registration. 01 2004.

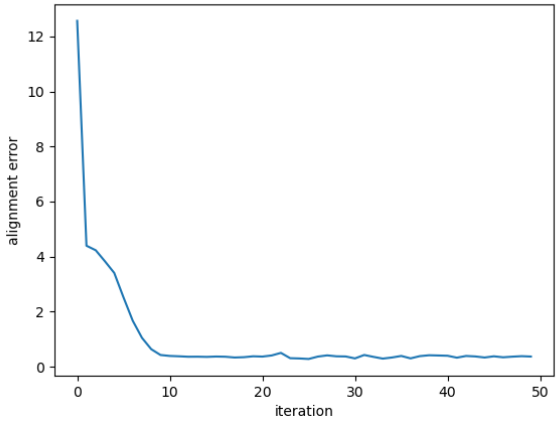


Figure 16: Alignment error using point-to-plane ICP

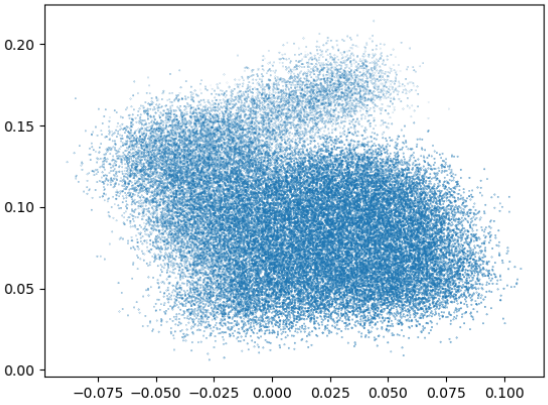


Figure 17: Point cloud of noisy meshes with standard deviation = 0.01

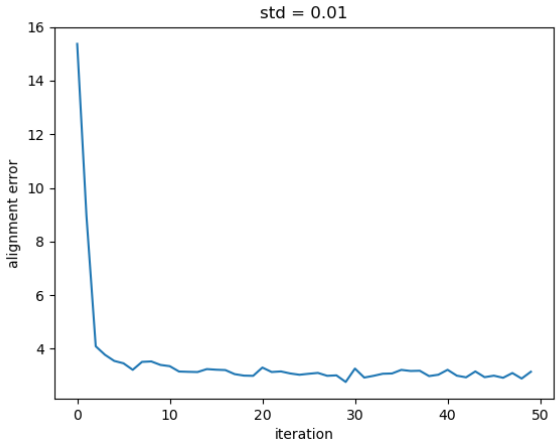


Figure 18: Point-to-point ICP

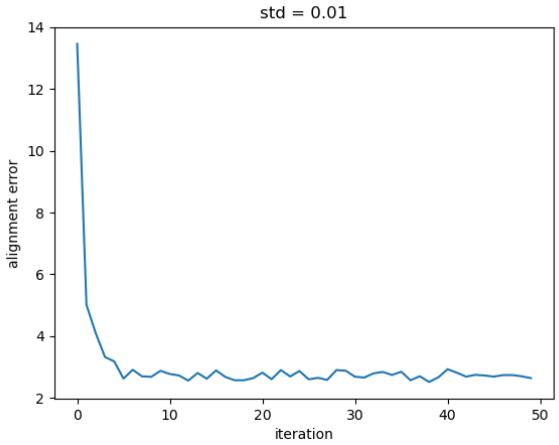


Figure 19: Point-to-plane ICP