Symmetry and Orbit Detection via Lie-Algebra Voting

Project of Nuages de Points et Modélisation 3D

Tong ZHAO

Overview

Introduction

Algorithm

Experiment

Conclusion

Task

Symmetries and orbits are everywhere in our daily life.

Task

Symmetries and orbits are everywhere in our daily life.

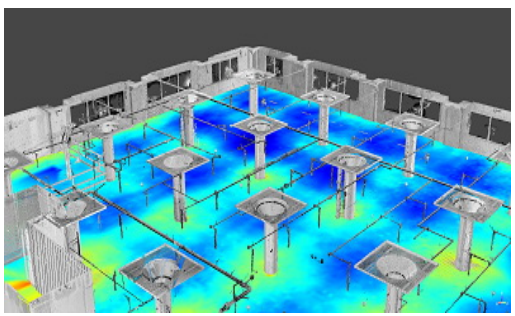
How to find them from a raw point cloud?

Motivation

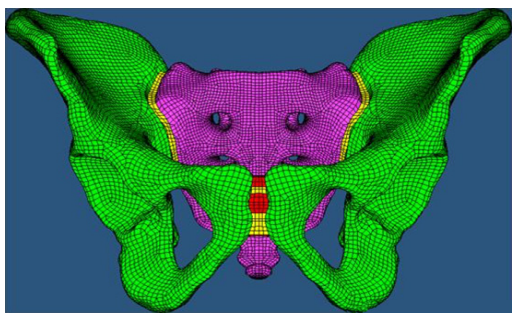
- Simplify the structure of the point cloud
- Eliminate noise
- Make the model physically balanced

Motivation

- Simplify the structure of the point cloud
- Eliminate noise
- Make the model physically balanced



Architecture Design



Biomedical Analysis

Physical Modelling

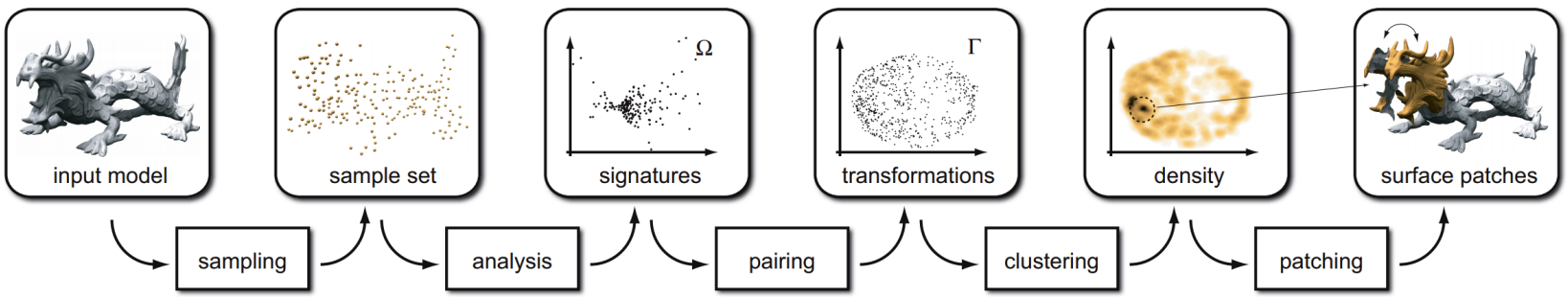
Goal

In this project, we studied the article: Symmetry and Orbit Detection via Lie-Algebra Voting [5].

- 2D Implementation in Python - Good visualization
- Parameter study in 2D cases
- 3D Implementation in C++ - Fast calculation (partially completed)
- Critical judgement

Pipeline

The algorithm is based on a complicated pipeline [4] going from local features to partial and global features.



Detection Pipeline

Step 1: Sampling

We sample uniformly n points (no more than 500) from the raw point cloud P_0. The reduced point set P is then used for calculating local features.

Motivation: Reduce the following computational complexity

Step 1: Sampling

We sample uniformly n points (no more than 500) from the raw point cloud P_0. The reduced point set P is then used for calculating local features.

Motivation: Reduce the following computational complexity

Limit: UNIFORMLY sampling

Step 2: Local Feature Computation

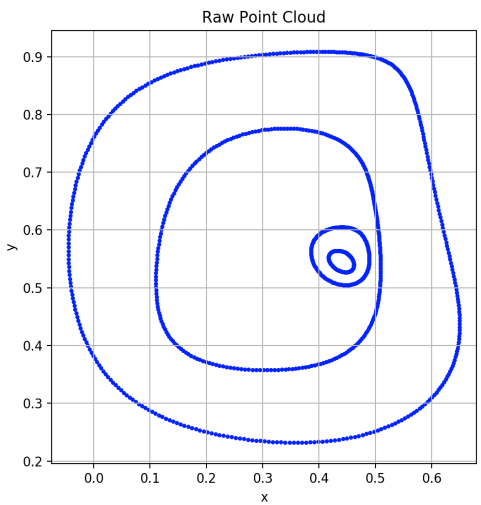
We estimate the oriented normals and the principal curvatures for every point p in P.

- Normal: estimated the normal n_i for each point $p_i \in P$ by local PCA.
- Orientation: oriented by connectivity in 2D point sets, by MST in 3D point sets.
- Curvature: estimated by second order differential form.

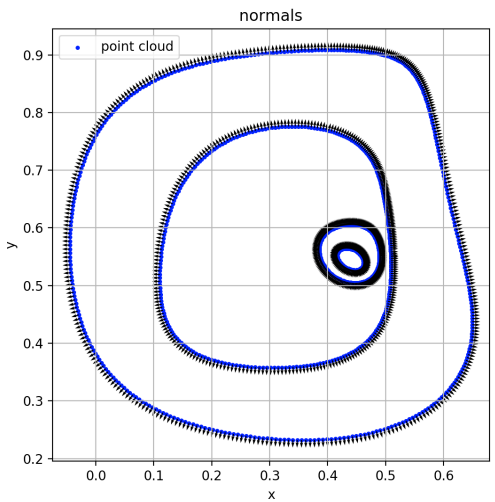
Step 2: Local Feature Computation

We estimate the oriented normals and the principal curvatures for every point p in P.

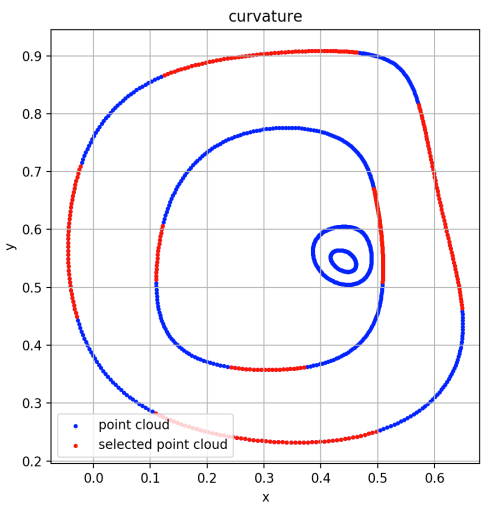
- Normal: estimated the normal n_i for each point $p_i \in P$ by local PCA.
- Orientation: oriented by connectivity in 2D point sets, by MST in 3D point sets.
- Curvature: estimated by second order differential form.



Sampled Point Cloud P



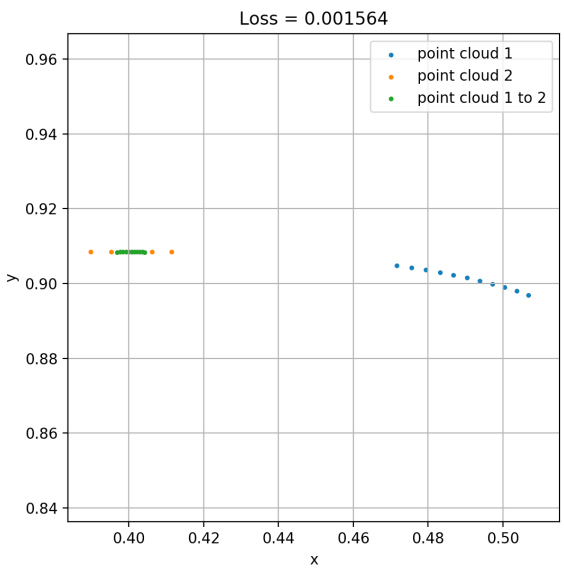
Oriented Normals



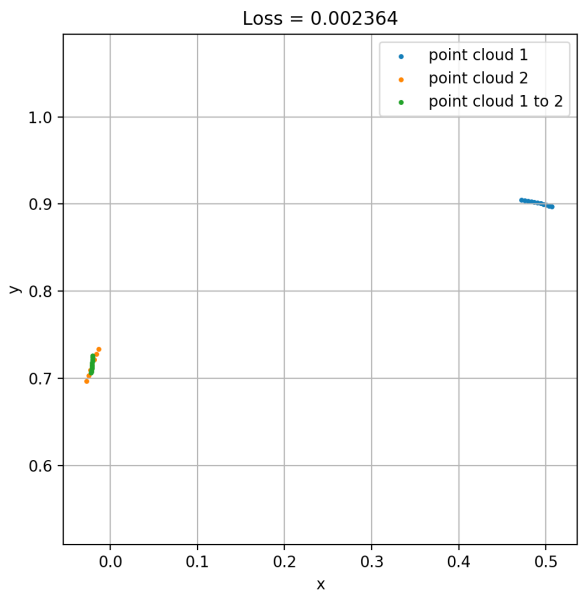
Curvatures > 3

Step 3: Point Pairing

We sample randomly one-fifth points from P, noted $\tilde{P}$. Then for each pair of points (p_i, p_j) where $p_i \in \tilde{P}$, $p_j \in P$, we estimate a transformation T_{ij}. Transformations with large alignment error are rejected, given a threshold s_1.



Accepted Example



Rejected Example

Step 4: Clustering

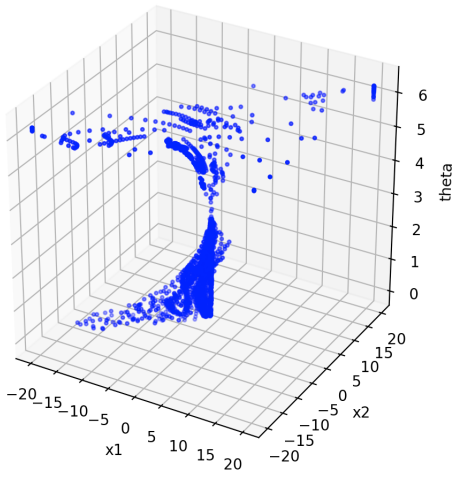
Given T_{ij} in SIM(2) / SIM(3), we embed it in the corresponding Lie algebra space $\mathfrak{sim}(2)$ / $\mathfrak{sim}(3)$ [2].

$$T = \begin{pmatrix} R & t \\ 0 & w^{-1} \end{pmatrix} \rightarrow \log(T) = \begin{pmatrix} \omega & u \\ 0 & -s \end{pmatrix}$$

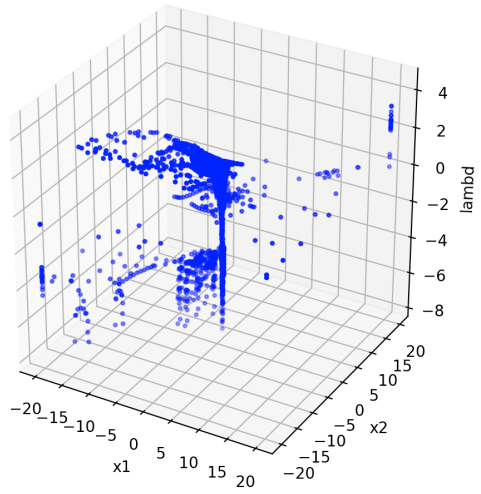
Since ω is a skew-symmetric matrix, the embedding can be represented by a vector:

- 2D cases: $log(T) \in \mathbb{R}^4$
- 3D cases: $log(T) \in \mathbb{R}^7$

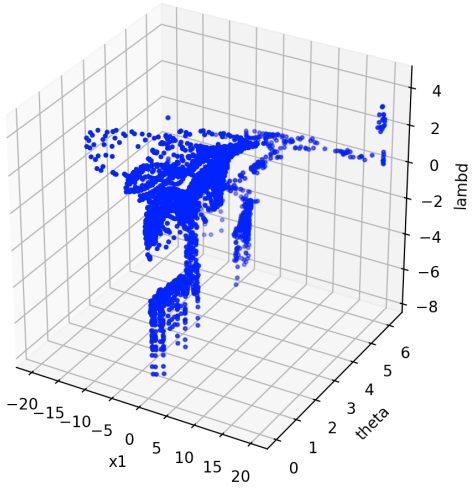
Step 4: Clustering



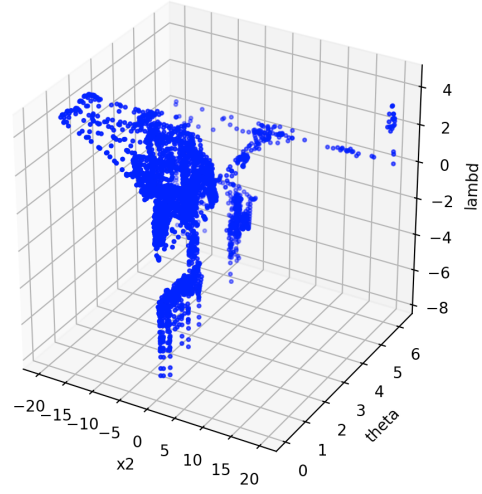
(x_1, x_2, θ)



(x_1, x_2, λ)



(x_1, θ, λ)



(x_2, θ, λ)

Step 4: Distance Metric

Given two transformations F and G, we have $\delta = FG^{-1}$ use a metric as following:

$$d^2(F,G) = log(\delta)^t \begin{pmatrix} \alpha Id & 0 & 0 \\ 0 & \beta Id & 0 \\ 0 & 0 & \gamma \end{pmatrix} log(\delta) = \|log(\delta)\|_E^2 = \|log(F) - log(G)\|_E^2$$

Step 4: Distance Metric

Given two transformations F and G, we have $\delta = FG^{-1}$ use a metric as following:

$$d^2(F,G) = log(\delta)^t \begin{pmatrix} \alpha Id & 0 & 0 \\ 0 & \beta Id & 0 \\ 0 & 0 & \gamma \end{pmatrix} log(\delta) = \|log(\delta)\|_E^2 = \|log(F) - log(G)\|_E^2$$

This distance is not independent from its scale.

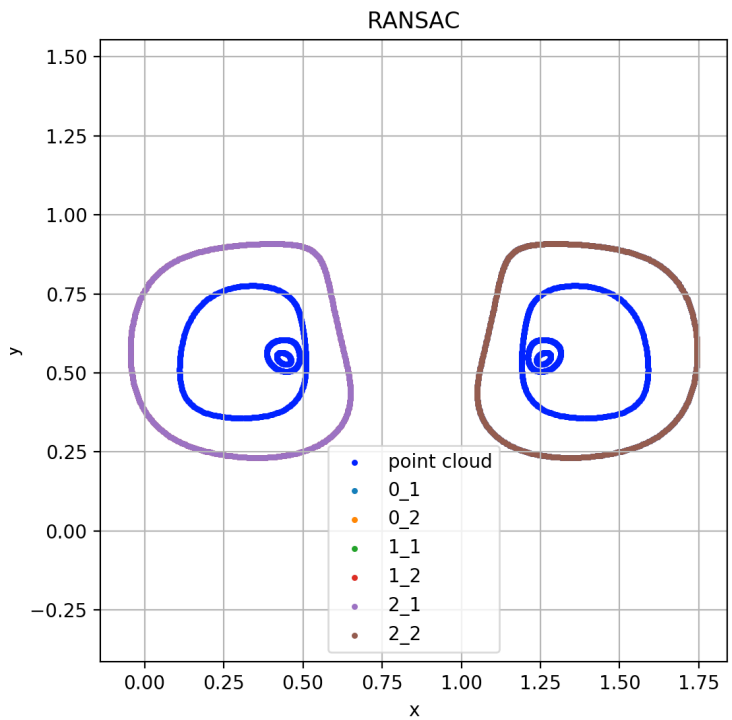
Step 4: Detection

- Symmetry Detection - Mean Shift [3]: The symmetry is represented by the mode of the transformations.
- Orbit Detection - RANSAC: According to Lie Algebra properties, a k-orbit always generates a k-dimensional linear subspace.

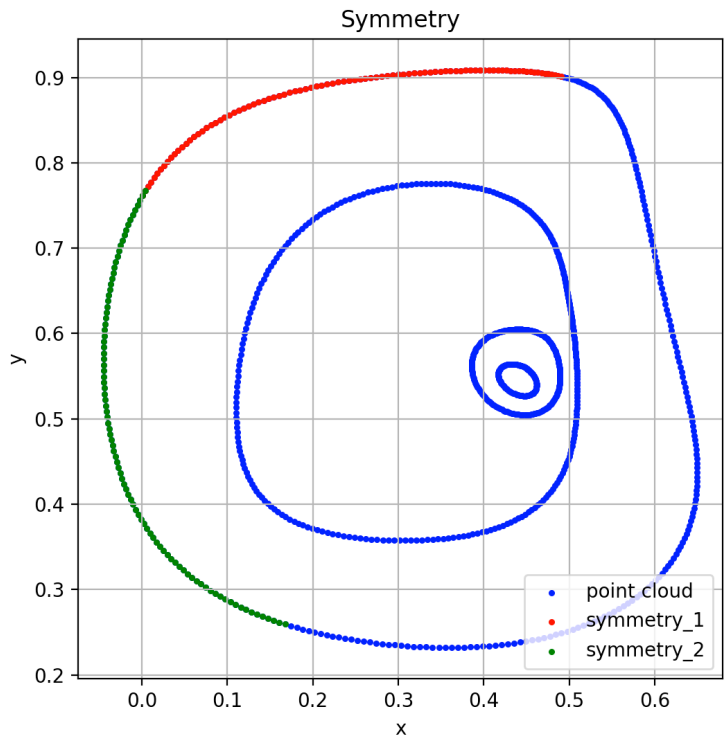
Both algorithms are implemented in Lie Algebra domain using the defined metric.

Step 5: Patching

Given a transformation T_{ij}, we perform a Region Growing algorithm [1] on both p_i and p_j to find the corresponding patches.

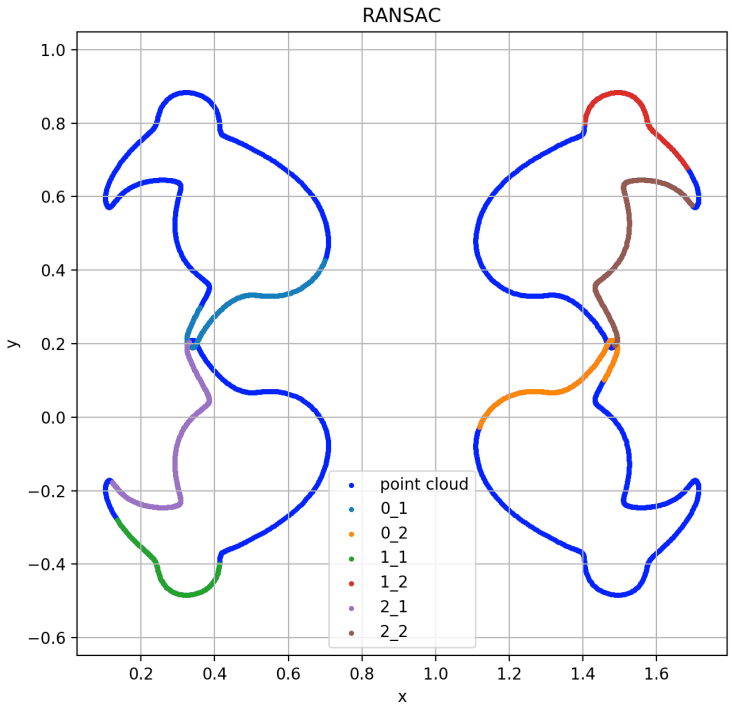


RANSAC

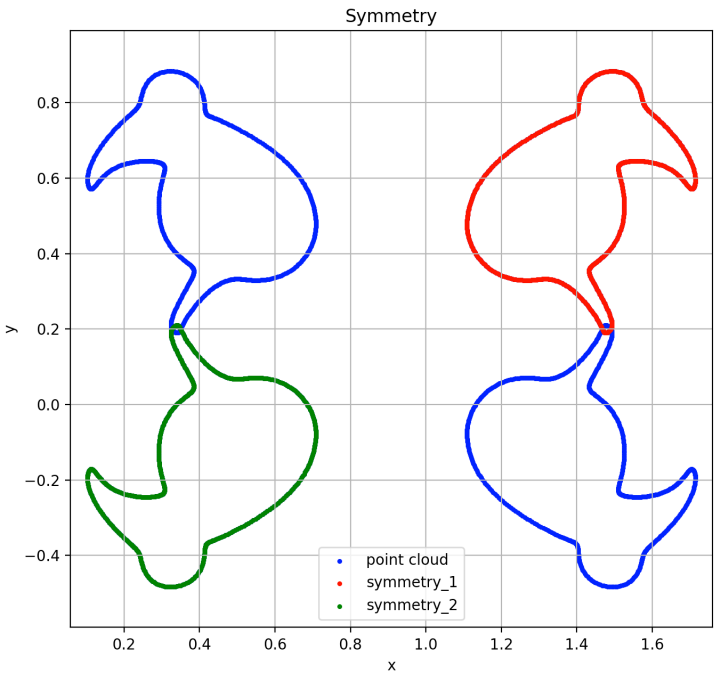


Mean Shift

Example 1: Bird

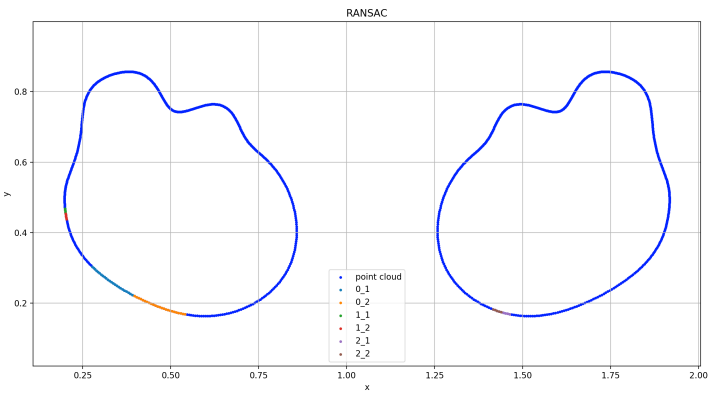


Orbit Detection - Bird

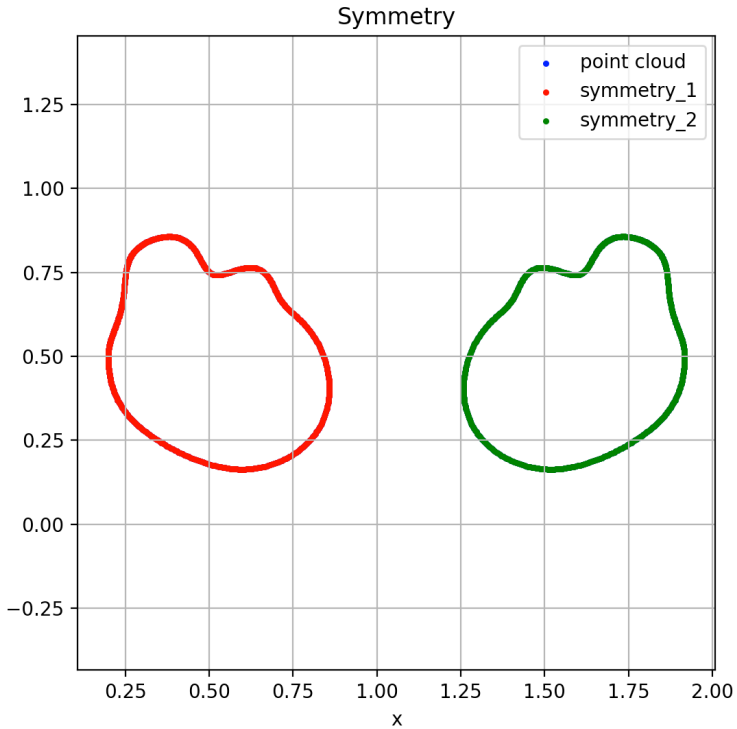


Symmetry Detection - Bird

Example 2: Blob



Orbit Detection - Blob



Symmetry Detection - Blob

Parameter Tuning

Parameter	Role	Choice
Radius r	decides the quality of estimations	can be estimated by average spacing
Number of points n	trade-off between accuracy and complexity	between 100 and 500
Threshold for pairing s_1	controls the quality of the paired transformations	fixed to 0.0005
Threshold for RANSAC s_2	decides the quality of RANSAC	fixed to 0.005
Variance of gaussian kernel σ	decides the smoothness of the gram matrix in Mean-shift	can be estimeted by Monte-Carlo method
Threshold for region growing s_3	controls the quality of the matching patches	difficult to decide

Advantage

1. **An Unified Framework** to detect both local and global symmetries and orbits. It avoids mapping the transformations to humain-designed spaces based on which symmetries are looked for.

2. **A Coordinate Frame Independent Clustering** to reduce manual tuning everytime when we change the point cloud.

Disadvantage

1. **High Computational Cost** due to the huge amount of possible transformations. A small amount of points lead to a bad result since it can not represent perfectly the raw shape.

2. **Parameter Tuning** is important to get a good result but it is not trivial.

3. **Limited Sampling Method** leads to failure in some cases.

4. **Subjective Evaluation Criteria** makes it difficult to evaluate the algorithm quantitatively.

Future works

1. Transformations based on segmentation can help us to reduce the complexity of the algorithm and make it more robust.

2. Deep learning methods may help to capture symmetries and orbits given the transformations.

References

[1] Rolf Adams and Leanne Bischof. Seeded region growing. *IEEE Transactions on pattern analysis and machine intelligence*, 16(6):641–647, 1994.

[2] Jose-Luis Blanco. A tutorial on se (3) transformation parameterizations and on-manifold optimization. *University of Malaga, Tech. Rep*, 3, 2010.

[3] Yizong Cheng. Mean shift, mode seeking, and clustering. *IEEE transactions on pattern analysis and machine intelligence*, 17(8):790–799, 1995.

[4] Niloy J Mitra, Leonidas J Guibas, and Mark Pauly. Partial and approximate symmetry detection for 3d geometry. In *ACM Transactions on Graphics (TOG)*, volume 25, pages 560–568. ACM, 2006.

[5] Zeyun Shi, Pierre Alliez, Mathieu Desbrun, Hujun Bao, and Jin Huang. Symmetry and orbit detection via lie-algebra voting. In *Computer Graphics Forum*, volume 35, pages 217–227. Wiley Online Library, 2016.