Approximate Symmetry Detection For Reverse Engineering

Bruce I. Mills
⟨B.I.Mills@cs.cf.ac.uk⟩

Frank C. Langbein
⟨F.C.Langbein@cs.cf.ac.uk⟩

David Marshall
⟨A.D.Marshall@cs.cf.ac.uk⟩

Ralph R. Martin
⟨R.R.Martin@cs.cf.ac.uk⟩

8th June 2001

Department of Computer Science
Cardiff University

Reverse Engineering

- Engineering converts a concept into an artifact
- Reverse engineering converts an artifact into a concept
- The desired result is a representation of the design intent, not a simple copy

Goal: Reconstruct an *ideal* model of a physical object with intended geometric regularities

Reconstructing Solid Models

- Initially the shape of a physical object is represented by a point cloud, e.g. obtained from a laser scanner
- An initial B-rep model can be generated from this
  - ⋆ Models with only planar, spherical, cylindrical, conical and toroidal surfaces
  - ⋆ There are methods to reconstruct these reliably
  - ⋆ Many engineering parts can be described in this way
- The initial model is disturbed by noise introduced by
  - ⋆ inaccuracies of the physical object
  - ⋆ scanning process
  - ⋆ reconstruction phases

Beautification

- Improve initial model in a post–processing step, called **Beautification**:
  - Analyse the model to find approximate geometric regularities
  - Reconstruct an improved model using geometric constraints
- Approximate symmetries of the initial model can be used to beautify it

Approximate Symmetry

- A geometric symmetry of a solid object is an isometry mapping the object onto itself
- Finite symmetry groups of the object are symmetries of special point sets
- Infinite symmetry groups are not the topic of this talk; they are either spherical or have a single central axis
- There are various ways to define approximate symmetry; none more justified than the other

Previous Work

- Exact symmetry of polyhedra in 3D can be detected in $O(n \log n)$ time [Sugihara]
- Different approaches for approximate symmetry:
  - ⋆ Find exact isometries which approximately preserve a given set of points ($O(n^6)$) [Alt]
  - ⋆ Find point sets close to the original points which are exactly symmetric (NP–complete) [Iwanowski]

Our Approach

- Find symmetries of the model as point set symmetries
- Detect symmetries as distance–preserving permutations; the geometric realization becomes secondary
- Automatically choose natural tolerances reducing local ambiguity instead of finding symmetry for a given tolerance

Point Set Symmetries

- A symmetry of the model is a symmetry of a point set derived from the model (vertices, centres of spheres, tori, apices of cones)
- There is typically a point set with the same symmetries as the model
- The point set could have more, but not less symmetries
- Add a post–processing step to check if the point set symmetries also preserve geometry types and combinatorial information

Permutations

- An approximate isometry of a point set is a permutation preserving the distances between the points approximately
- The permutations are the leaves of a tree of partial injections:
  - ⋆ A partial injection is a list of point pairs where each point appears at most once as first and at most once as second element of the pairs
  - ⋆ The root of the tree is the empty list
  - ⋆ The children of a partial injection are obtained by adding one more point pair to the list

Algorithm Overview

Approximate symmetry detection for point sets:

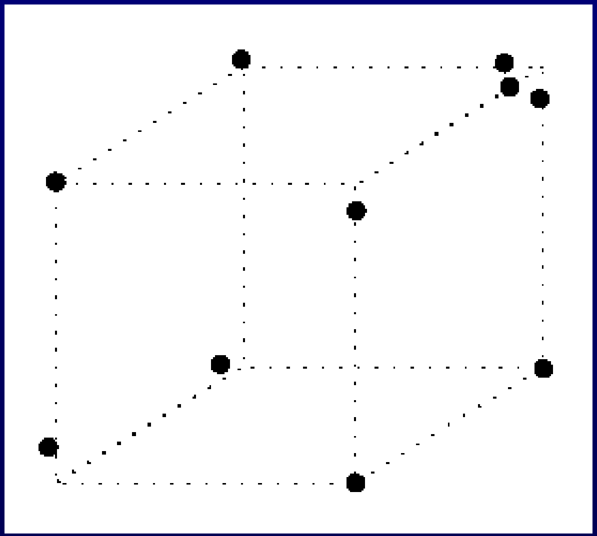
I. Create consistent clusterings of the points at different tolerance levels

II. For each consistent clustering, search the tree of partial injections to find valid isometries

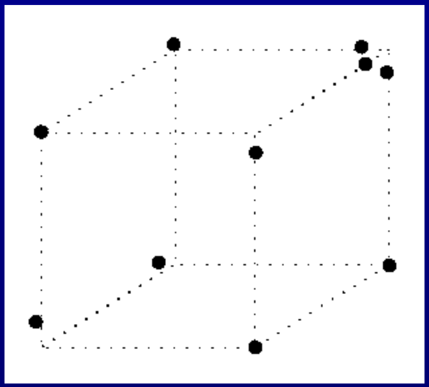
Stage I: Consistent Clusterings

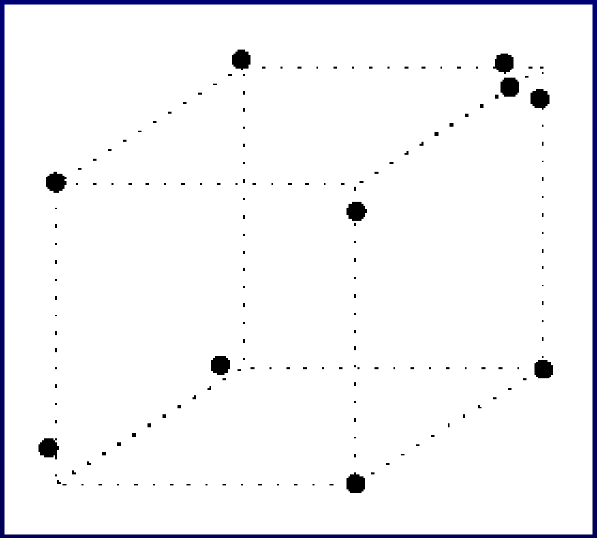
- Select tolerance levels by creating consistent clusterings of the points
- Consistent clusterings:
  - ⋆ Each point belongs to exactly one cluster
  - ⋆ All distances between the points in a cluster are smaller than the tolerance
  - ⋆ Distances between points from different clusters are larger than the tolerance

Example for Consistent Clusterings

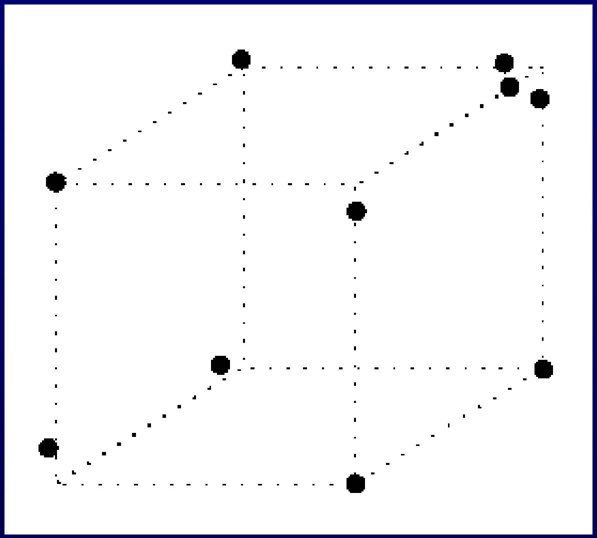


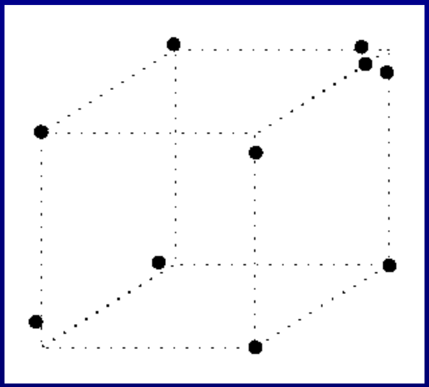
Example for Consistent Clusterings

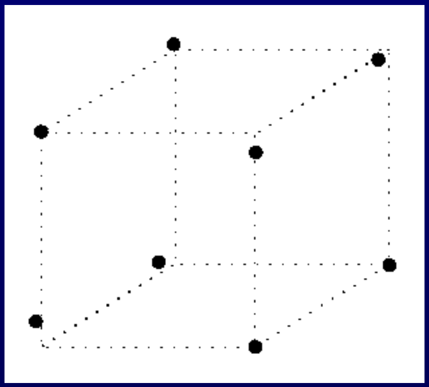




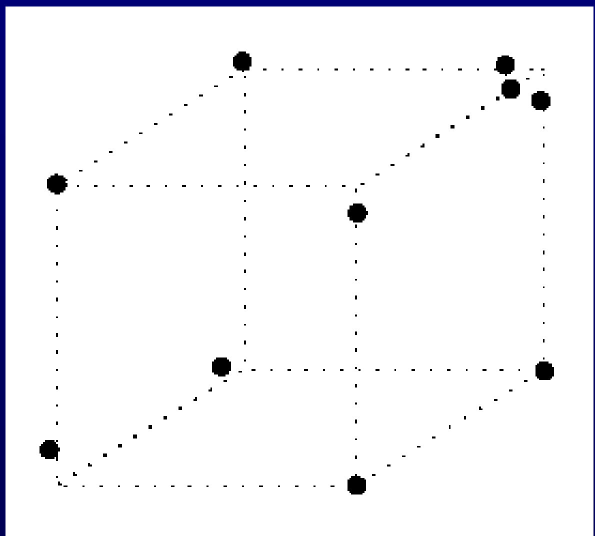
Example for Consistent Clusterings

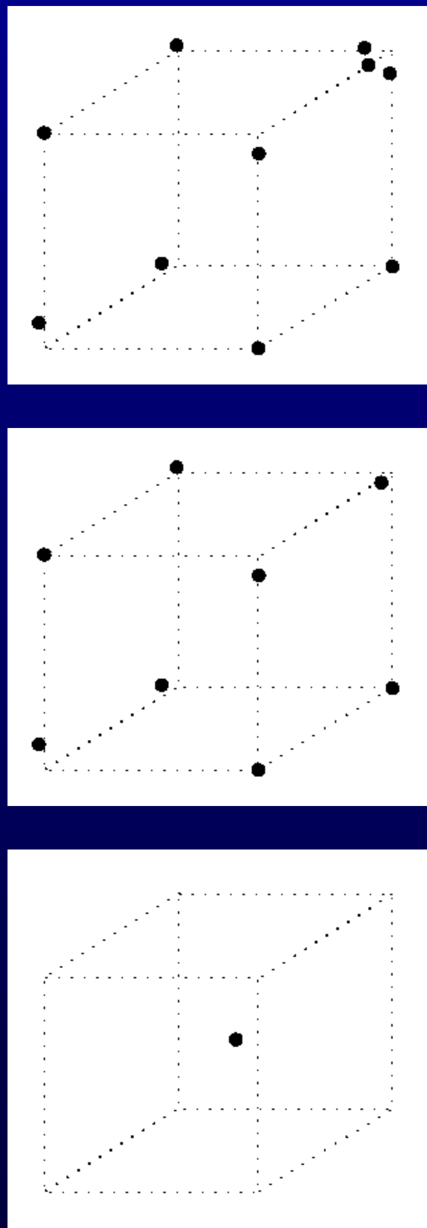






Example for Consistent Clusterings





Stage II: Symmetry Analysis 1

Detect distance–preserving permutations of the clustered point set

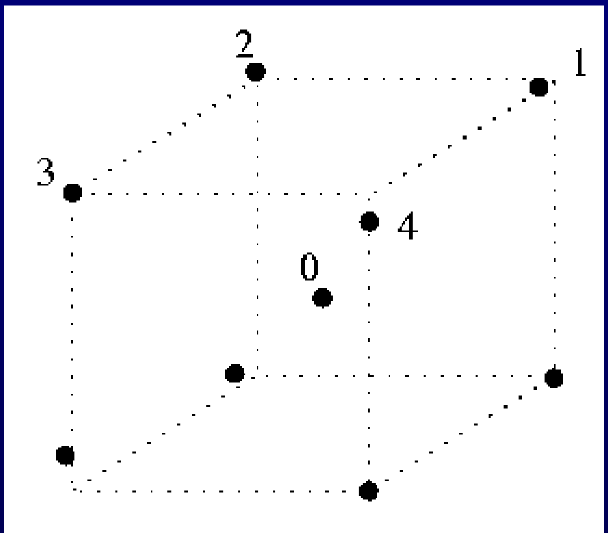
1. Find a large, non–degenerate tetrahedron whose vertices are
   - ⋆ the centroid of the clustered point set
   - ⋆ three points on the convex hull of the clustered point set chosen to be as far apart as possible from each other

Stage II: Symmetry Analysis 2

2. Do a limited depth–first search over the tree of partial injections mapping the points of the tetrahedron:
   - ⋆ The centroid always has to be mapped onto itself
   - ⋆ Backtrack to the parent whenever the newly added point pair induces an isometry which does not approximately preserve the distances between the points
   - ⋆ Once three points are mapped, the fourth point can only be mapped to two possible locations
   - ⋆ All subsequent points are mapped to one location, thus check the remaining distances directly

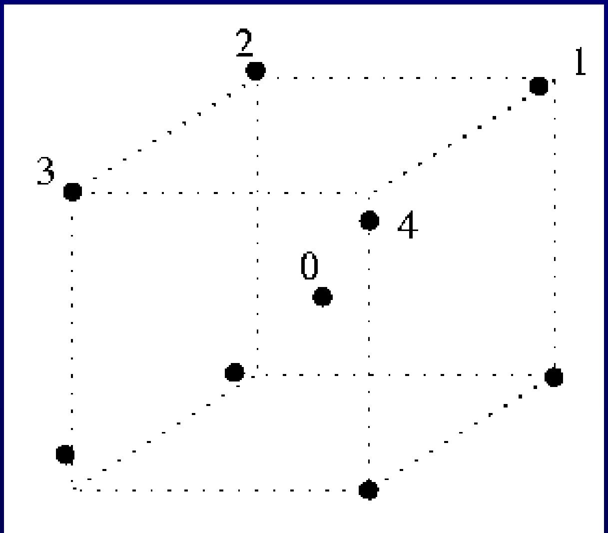
Symmetry Analysis Example

- Select tetrahedron: $0, 1, 2, 3$

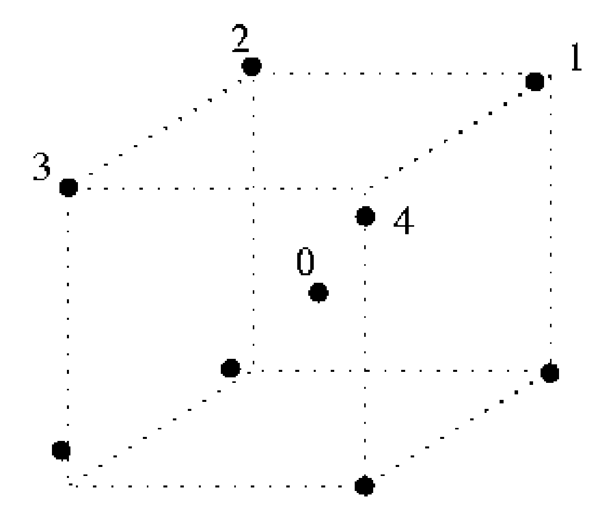


Symmetry Analysis Example

- Select tetrahedron: $0, 1, 2, 3$
- Map the centroid: $0 \longrightarrow 0$

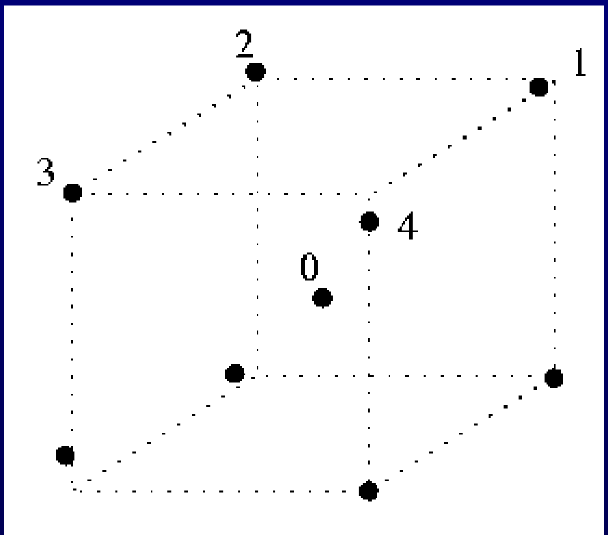


Symmetry Analysis Example



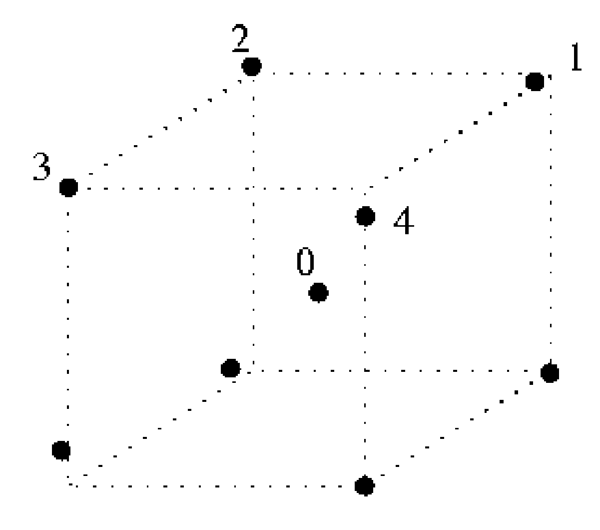
- Select tetrahedron: $0, 1, 2, 3$
- Map the centroid: $0 \longrightarrow 0$
- Map $1 \longrightarrow 2$
  - ⋆ Distance check: $d(0,1) \approx d(0,2)$

Symmetry Analysis Example



- Select tetrahedron: $0, 1, 2, 3$
- Map the centroid: $0 \longrightarrow 0$
- Map $1 \longrightarrow 2$
  - ⋆ Distance check: $d(0,1) \approx d(0,2)$
- Map $2 \longrightarrow 4$
  - ⋆ Distance check: $d(0,2) \approx d(0,4)$
  - ⋆ Distance check: $d(1,2) \not\approx d(2,4)$

Symmetry Analysis Example



- Select tetrahedron: $0, 1, 2, 3$
- Map the centroid: $0 \longrightarrow 0$
- Map $1 \longrightarrow 2$
  - ⋆ Distance check: $d(0,1) \approx d(0,2)$
- Map $2 \longrightarrow 4$
  - ⋆ Distance check: $d(0,2) \approx d(0,4)$
  - ⋆ Distance check: $d(1,2) \not\approx d(2,4)$
- Backtrack and map $2 \longrightarrow 3$
  - ⋆ Distance check: $d(0,2) \approx d(0,3)$
  - ⋆ Distance check: $d(1,2) \approx d(2,3)$

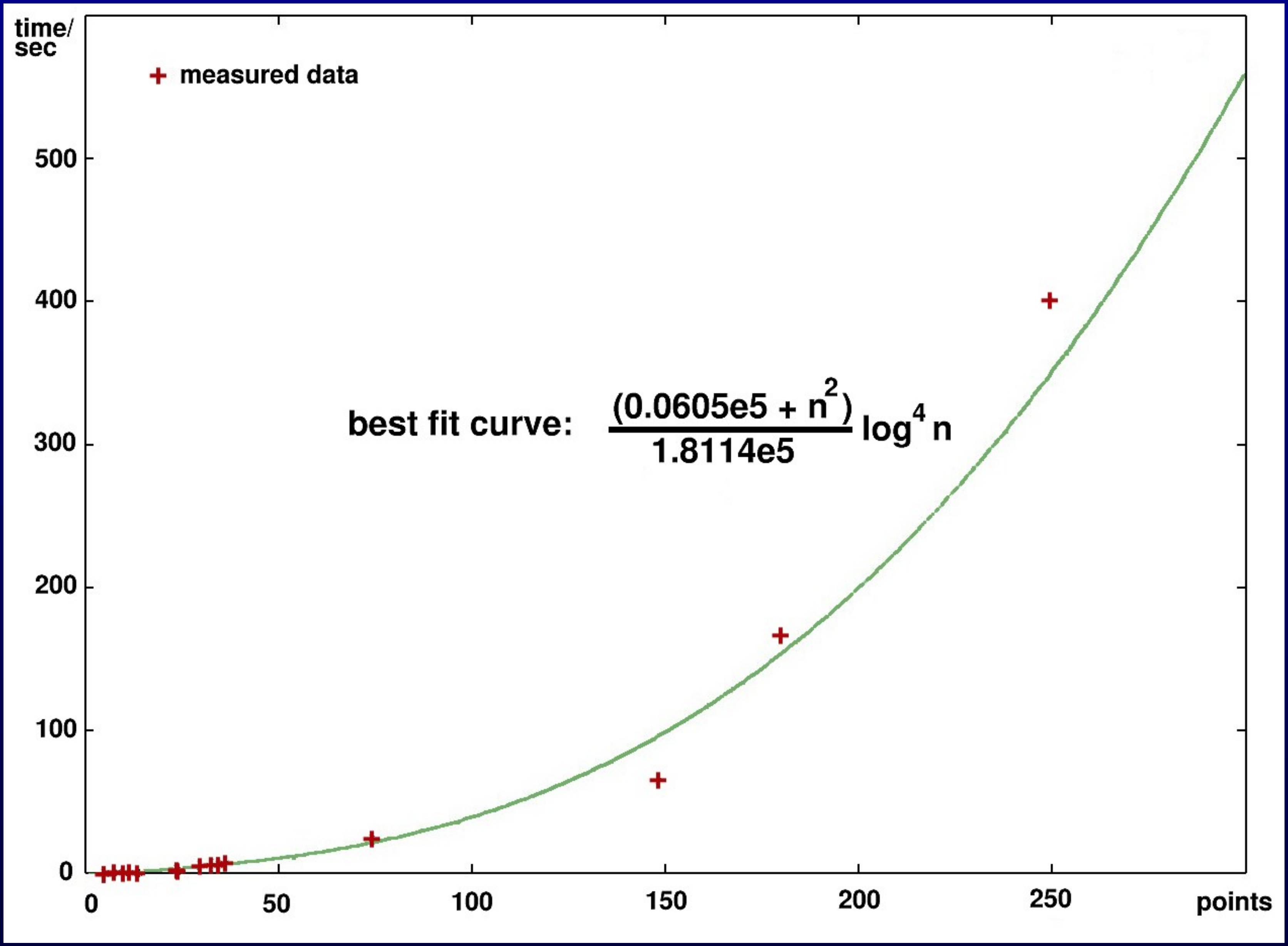
Elongated Objects

- A point set that is several times longer than it is across, can only have prismatic symmetries
- The algorithm sees this as an approximately linear arrangement
- The rotational and mirror symmetries can be detected in a second pass by expanding the points radially from the central axis

Performance Analysis

- Worst case time order: $O(n^{3.5} \log^4 n)$; requires as many consistent clusterings as points
- For usual engineering objects a pragmatic upper bound is $O(n^2 \log^4 n)$
- Tests showed that the algorithm produces correct results for typical engineering objects in about 20 minutes
- For objects with little symmetry, the algorithm is very fast
- More time is needed for very symmetric objects
- Experiments with typical objects supported the pragmatic upper bound

Experiments



Conclusions

- Our concept of approximate symmetry lead to an algorithm with good theoretical and practical performance
- The algorithm has no tuning parameters, no measure of symmetry, just exact answers to the existence of approximate symmetry
- The results are highly immune to small variations

Future Work: Detect partial symmetries