Design Intent of Reverse Engineered Geometric Models

F. C. Langbein

17th July 2002

Department of Computer Science
Cardiff University

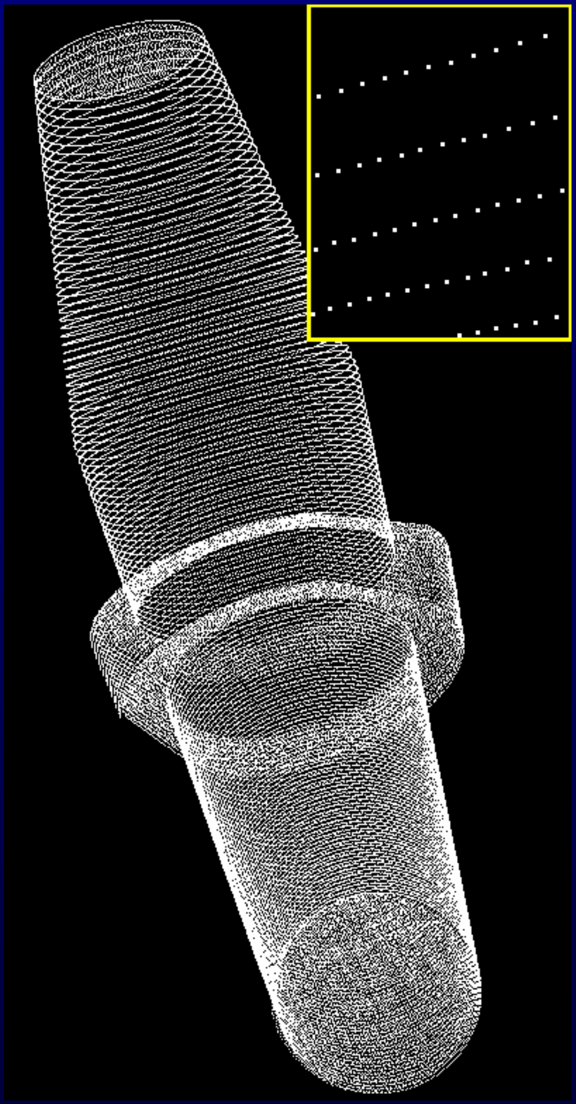
Design Intent

- Engineering converts a concept into an artifact
- Reverse engineering converts an artifact into a concept
- In both applications design intent is a central factor

Goal: Automatically detect and represent the design intent of geometric models for intelligent CAD applications

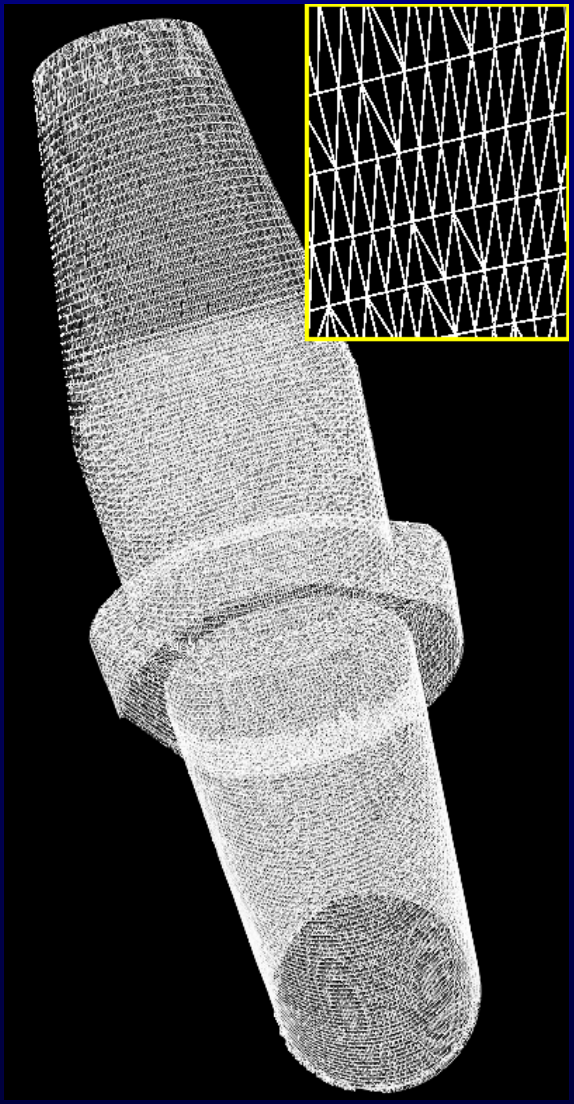
- ⋆ Simpler, high-level design interfaces for engineers
- ⋆ Support for non-expert users

Reverse Engineering



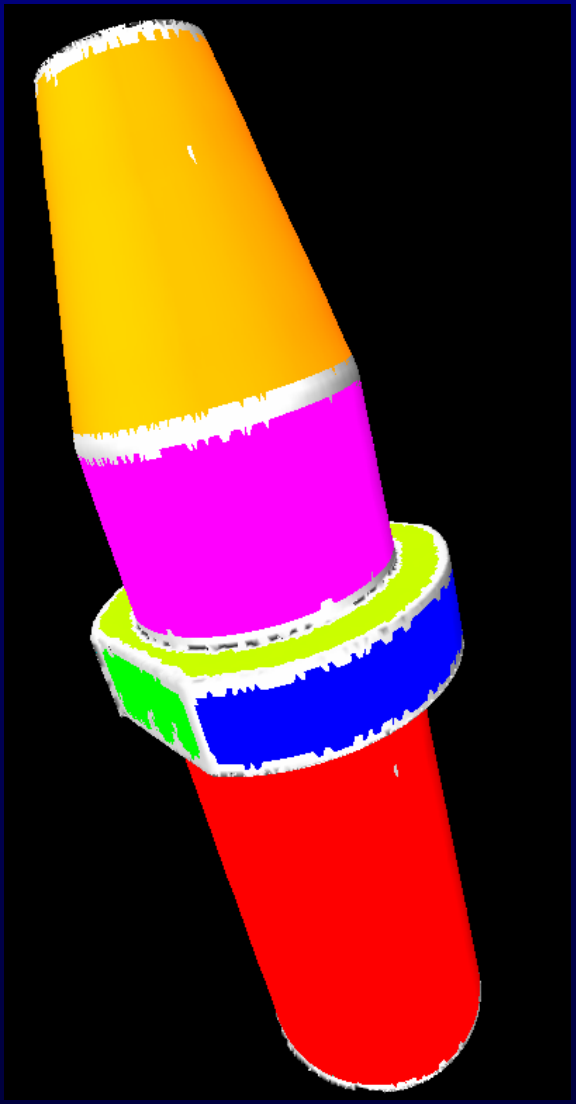
- **Data Acquisition**
  - ⋆ Obtain multiple views from a 3D laser scanner
  - ⋆ Register views to a single 3D point set

Reverse Engineering



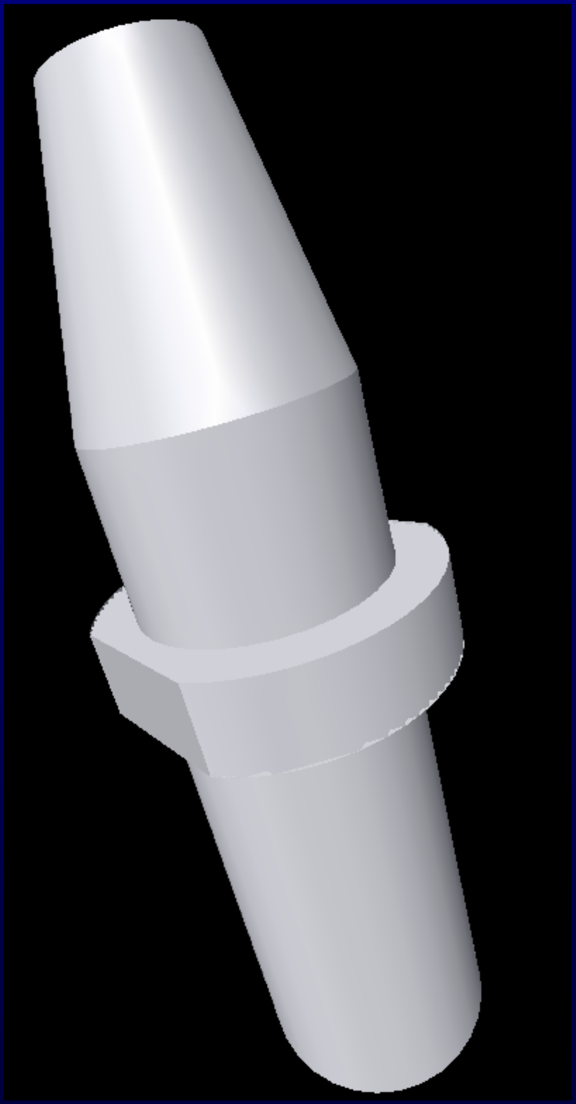
- **Data Acquisition**
- **Triangulation**
  - ⋆ Create a triangular mesh for the point set

Reverse Engineering



- **Data Acquisition**
- **Triangulation**
- **Segmentation, Surface Fitting**
  - ⋆ Split the point set into subsets representing natural surfaces
  - ⋆ Find the surface type (plane, sphere, cylinder, cone, torus) and fit a surface of this type for each subset

Reverse Engineering



- **Data Acquisition**
- **Triangulation**
- **Segmentation, Surface Fitting**
- **Model Creation**
  - ⋆ Create an initial solid model by stitching surfaces

Beautification

- **Problem:** Reverse engineered models suffer from inaccuracies caused by
  - ⋆ sensing errors during data acquisition
  - ⋆ approximation/numerical errors during reconstruction
  - ⋆ possible wear of the artifact
  - ⋆ manufacturing method used to make the artifact
- **Goal:** Reconstruct an *ideal* model of a physical object with intended geometric regularities
- Design intent has to be considered at some stage of the process
- **Our approach:** **Beautification**, improve the model in a post-processing step

Our Beautification Strategy

Analyser

Detect potential geometric regularities which are approximately present in the initial model

→

Hypothesizer

Selection

Based on priorities and inconsistencies select a set of likely regularities

↓ ↑

Solvability Test

Test if selected regularities are mutually consistent and indicate inconsistencies

←

Reconstruction

Reconstruct an improved model, align model with coordinate axes, etc.

Key Aspects of Beautification

- Detection of suitable approximate regularities

- Selection of regularities to improve the model

  Selection criteria:

  - ⋆ *intended, consistent design*
  - ⋆ *solvability of the model*

- Representation of design intent in improved model

Approximate Geometric Regularities

- Expressed as similarities and special arrangements
- Similarities detected by *hierarchical clustering* of face, edge and vertex properties
- Clustering hierarchy simplified by detection of
  - ⋆ distinct tolerance levels based on regularity cond.
  - ⋆ large tolerance jumps
- For instance: cluster positions at tolerance levels t_l such that the distance between positions
  - ⋆ in the same cluster is significantly smaller than t_l
  - ⋆ in different clusters is significantly larger than t_l

Regular Arrangements

- Previous steps create clusters at distinct tol. levels
- Look for regular arrangements of the clusters

  (e.g. vertex clusters, clusters of directions as points on a sphere)

  - ⋆ Approximate symmetries
  - ⋆ Almost congruent subsets
  - ⋆ Approximate partial symmetries

    (minimal number of subsets with maximal symmetry)
- Define and detect regularities such that non-arbitrary tolerance levels can be determined automatically

Strategy for Selection of Regularities

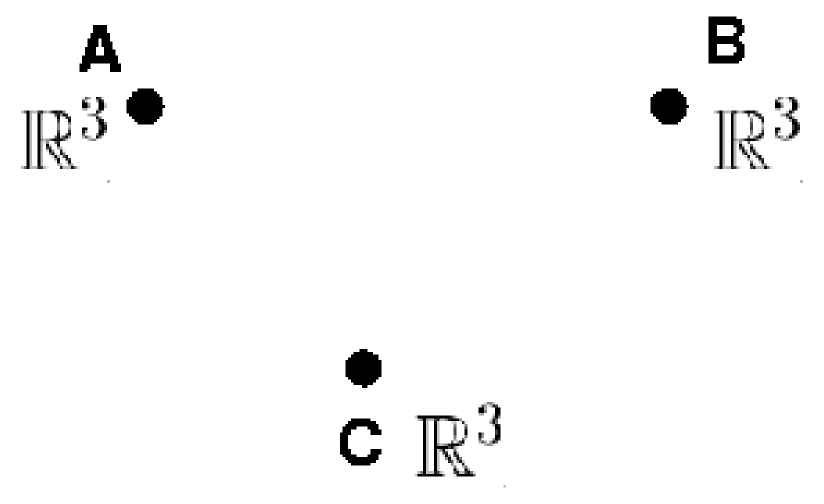
I. Prioritize the regularities based on
  - how well they are satisfied in the initial model
  - how common and desirable the regularity is

II. Select initial subset $\mathbf{S}$ from all detected regularities
  - Resolve simple inconsistencies using selection rules
  - Favour regularities with high priorities

III. In order of highest to lowest priority add regularities $\mathbf{c}$ from $\mathbf{S}$ to a constraint system $\mathbf{C}$
  - If $\mathbf{C}$ with the new regularity $\mathbf{c}$ is solvable add $\mathbf{c}$
  - Otherwise adjust $\mathbf{S}$ using the selection rules from II as $\mathbf{c}$ is not used

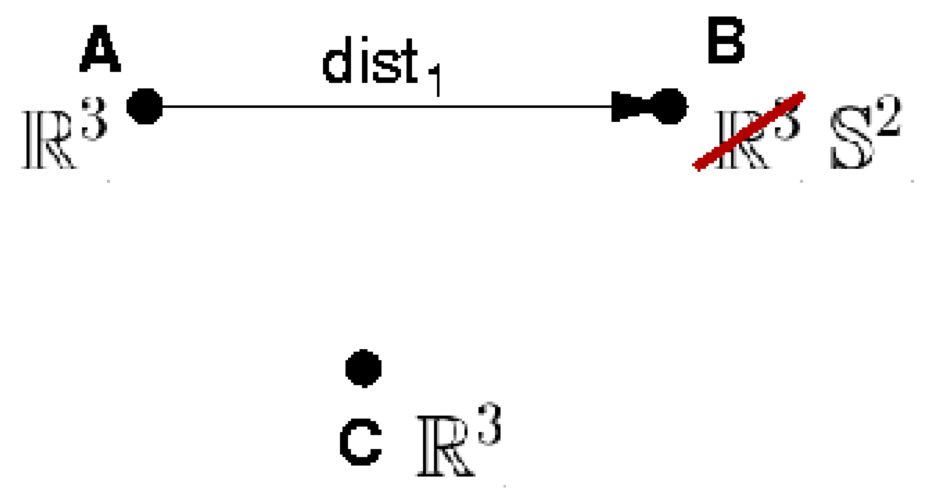
Solvability Test

- Geometric constraints can be represented as edges between geometric objects in a graph
- Analysing the graph provides information about generic solvability of the constraint system
- Solving the constraint system is not required
- Simple example of distances between points:



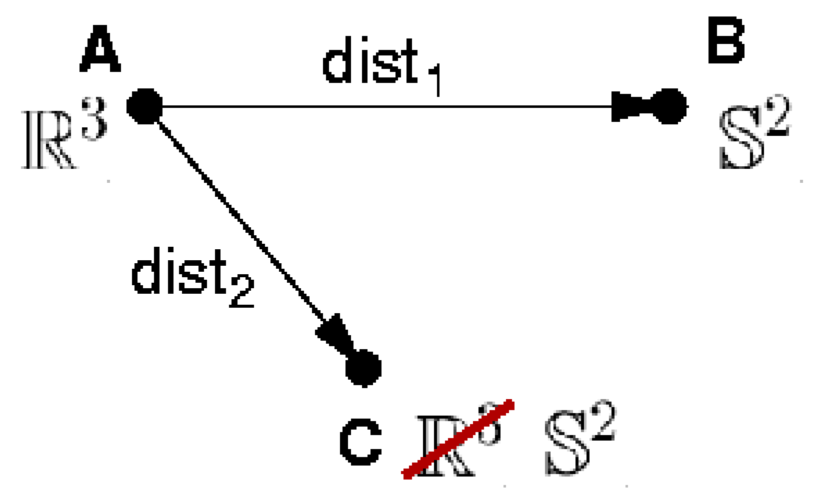
Solvability Test

- Geometric constraints can be represented as edges between geometric objects in a graph
- Analysing the graph provides information about generic solvability of the constraint system
- Solving the constraint system is not required
- Simple example of distances between points:



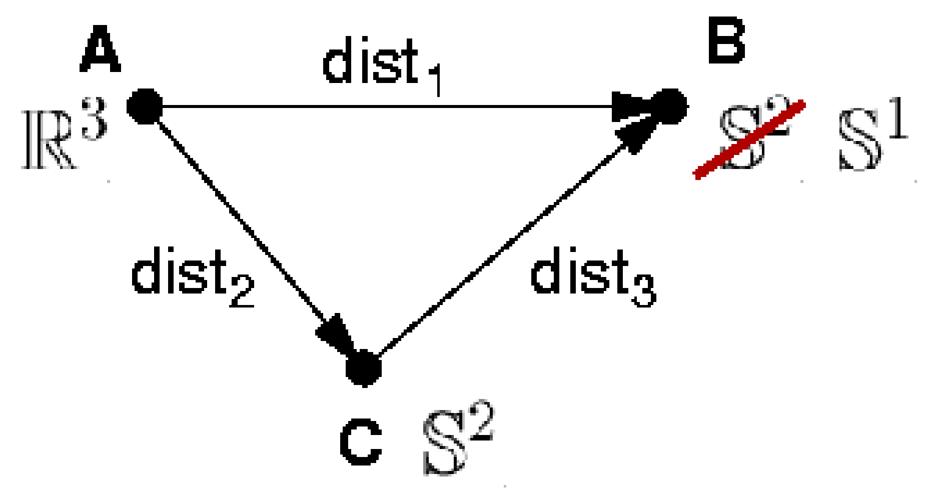
Solvability Test

- Geometric constraints can be represented as edges between geometric objects in a graph
- Analysing the graph provides information about generic solvability of the constraint system
- Solving the constraint system is not required
- Simple example of distances between points:



Solvability Test

- Geometric constraints can be represented as edges between geometric objects in a graph
- Analysing the graph provides information about generic solvability of the constraint system
- Solving the constraint system is not required
- Simple example of distances between points:



Degree of Freedom Analysis

- Degree of freedom analysis detects generic solvability
- Non-generic cases are not detected at this stage

  (e.g. if the distances between the 3 points force the points to be on a line, we do not detect this)
- After a constraint has been successfully added to the graph we simplify the graph such that
  - ⋆ sufficient information for the solvability test remains
  - ⋆ solvable sub-systems (rigid sub-parts) are identified

Current State of Development

- Suitable approximate regularities can be detected
- Major intended regularities can be identified by prioritizing the regularities
- Efficient solvability test works reliably for generic cases

⇒ We can improve small to medium sized models within a few minutes

Future Work on Intelligent Selection

- Ambiguities between approximate regularities cause inconsistent selection of regularities with respect to design intent

  (e.g. do we have a cube with edge lengths 2 or a rectangular prism with edge lengths 2 and 2.1, or $\ldots$?)

- Requires to make decisions in the context of the whole model, not locally with respect to inconsistencies

- Develop intelligent selection process employing
  - ⋆ general geometric reasoning
  - ⋆ specific design knowledge

Future Work on Constraints

- Expand theoretical foundation of geometric constraints
  - ⋆ Handle non-generic cases
  - ⋆ Handle inequality constraints
  - ⋆ Improve efficiency and reliability of solvability results
- Investigate relations between geometric constraints and representation of geometric models
  - ⋆ Encode design intent in the representation
  - ⋆ Develop representations more robust to approximation errors and numerical ambiguities

Conclusion

- Design intent is crucial for intelligent CAD applications
- Inaccurate models can be improved by detecting regularities and enforcing them using geometric constraints

Open Problems

- Intelligent regularity selection methods
- Inequality constraints and non-generic cases
- Robust detection and representation of design intent