Numerical Methods for Beautification of Reverse Engineered Geometric Models

R. R. Martin
F. C. Langbein A. D. Marshall

11th July 2002

Department of Computer Science
Cardiff University
Reverse Engineering

- Engineering converts a concept into an artifact
- Reverse engineering converts an artifact into a concept
- Desired result is a representation of the design intent, not a simple copy
- **Problem:** Reverse engineered models suffer from inaccuracies caused by
  - sensing errors during data acquisition
  - approximation and numerical errors from reconstruction algorithms
  - possible wear of the artifact
  - manufacturing method used to make the artifact
Beautification

- **Goal:** Reconstruct an *ideal* model of a physical object with intended geometric regularities
  - Only for engineering objects with planar, spherical, cylindrical, conical and toroidal surfaces (and blends)
- Previous approaches:
  - Augment the surface fitting step by constraint solving methods [Fisher, Benkő]
  - Manually identify features like slots and pockets and use them to drive the segmentation and surface fitting [Thompson]
- **Our approach:** Beautification, improve the model in a post-processing step
Beautification Strategy

**Analyser**
Detect potential regularities which are approximately present in the initial model

**Reconstruction**
Reconstruct an improved model, fix topological problems, align model with coordinate axes, etc.

**Hypothesizer**
Solve a constraint system derived from the regularities which describes a complete, improved model with likely regularities (only a subset of the constraints will be mutually consistent)
Beautification Strategy

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

Reconstruction
Reconstruct an improved model, fix topological problems, align model with coordinate axes, etc.

Hypothesizer
Constraint Selection
Based on priorities and inconsistencies select a set of likely constraints

Constraint Solver
Try to solve constraint system and indicate inconsistencies (solvability test)
## Geometric Regularities

### Directions
- Parallel directions
- Directions with same angle relative to a special direction
- Symmetrical arrangements of directions

![Parallel directions diagram](image)

### Axes
- Axis intersections
- Aligned axes
- Parallel axes arranged equally spaced along lines, grids or on cylinders

![Axes diagram](image)

### Positions
- Equal positions
- Positions equal under projection
- Positions arranged regularly on lines and grids

### Parameter
- Equal lengths
- Equal angles
- Integer relations
- Special values:
  - ★ integers
  - ★ simple fractions

Numerical Methods for Beautification
Core of beautification problem:

constraint selection and solving

- Regularities become sets of geometric constraints
- Model topology is described by geometric constraints

Algorithm has to select desirable constraints such that the constraint system has a solution

Finding a solution is a secondary task
Geometric Constraints and Objects

• Geometric constraints used to express regularities:
  \[ d_1^t d_2 = \cos(\alpha) \quad \text{const./var. angle between directions} \]
  \[ d^t d = 1 \quad \text{Normalize direction} \]
  \[ \| p_1 - p_2 \| = \lambda \quad \text{const./var. distance between positions} \]
  \[ p = \frac{1}{n} \sum_{k=1}^{n} p_k \quad \text{Average position} \]
  \[ \sum_k \alpha_k s_k = \sigma \quad \text{Linear relation between scalars} \]
  \[ p \in O \quad \text{Vertex on geometric object} \]

• Geometric object types:
  ★ plane, sphere, cone, cylinder, torus, circular ellipsoid
  ★ straight line, circle, ellipse
  ★ vertex
  ★ direction, angle, length
Selection & Solving Strategy

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

II. Select initial subset $S$ 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 $c$ from $S$ to a constraint system $C$
   – Try to solve constraint system $C$ with new regularity $c$
   – Add $c$ to $C$ if extended system is solvable
   – Otherwise adjust $S$ using the selection rules from II as $c$ is not used
Prioritizing Regularities

- Priorities determine which regularity to choose in case of an inconsistency.

- Computed as weighted average with values in [0,1]:

\[
wc(r)(ce we(r) + cq wq(r) + cb wb(r))
\]

☆ with merit functions in [0,1]

\(we(r)\) numerical accuracy of regularity

\(wc(r)\) constant indicating how common the regularity type is

\(wq(r)\) specific quality/desirability depending on type and involved arrangements and constants

\(wb(r)\) constant minimum desirability

☆ and weighting constants \(ce, cq, cb\)
Quality Factors

- Specific quality \( w_q(r) \) of a regularity is determined by factors like
  - Special values involved (fractions with small integers preferred)
  - Number of involved B-rep elements with common boundary
  - Number of occupied positions for symmetrical arrangements on grids, circles, cylinders, ...
  - Number of B-rep elements of same geometric type
- \( w_q(r) \) is weighted average of merit functions for these factors
Selection Rules

• Selection rules to resolve simple inconsistencies between constraints:
  ★ Constraints between the same objects with different constants
  ★ Different constraints between objects which are identified by coincidence constraints

• Generic selection rule:
  A rule \((n_1, R_1, n_2, R_2)\) is violated if at least \(n_k + 1\) constraints in the set \(R_k\) are selected for each \(R_k \neq \emptyset\)

• One interpretation:
  If at least \(n_1 + 1\) elements of \(R_1\) are selected, then at most \(n_2\) elements of \(R_2\) should be selected
• Initially all constraints are selected
• Consecutively enforce each rule such that a maximal set of constraints with largest priorities are selected
• Basic concept for rule enforcement algorithm:
  ★ If a rule is violated deselect sufficient constraints with lowest priority
  ★ Check if these deselections allow the selection of previously deselected constraints by checking previously enforced rules
Regularities $A, B, C, D$ with $w(A) < w(B) < w(C) < w(D)$

A B C D 3 rules enforced in sequence:

Initially all regularities selected

•: regularity selected, □: regularity deselected
Rule Enforcement Example

Regularities $A, B, C, D$ with $w(A) < w(B) < w(C) < w(D)$

$A \ B \ C \ D$  3 rules enforced in sequence:

1: $R_1 = \emptyset, n_1 = 0, R_2 = \{A, B\}, n_2 = 1$

[• • • •]

At most either $A$ or $B$, not both, can be selected

→ Deselect $A$ due to lower priority

•: regularity selected, □: regularity deselected
Regularities A, B, C, D with $w(A) < w(B) < w(C) < w(D)$

A B C D  

3 rules enforced in sequence:

1: $R_1 = \emptyset$, $n_1 = 0$, $R_2 = \{A, B\}$, $n_2 = 1$

2: $R_1 = \{B\}$, $n_1 = 0$, $R_2 = \{C, D\}$, $n_2 = 1$

If B is selected, then at most either C or D, not both, can be selected

→ Deselect B due to lower priority

•: regularity selected, □: regularity deselected
Regularities $A, B, C, D$ with $w(A) < w(B) < w(C) < w(D)$

3 rules enforced in sequence:

1: $R_1 = \emptyset$, $n_1 = 0$, $R_2 = \{A, B\}$, $n_2 = 1$

2: $R_1 = \{B\}$, $n_1 = 0$, $R_2 = \{C, D\}$, $n_2 = 1$

$\leftarrow$ $B$ modified, so recheck rule 1

$\rightarrow$ Can reselect $A$

$\bullet$: regularity selected, $\square$: regularity deselected
Regularities A, B, C, D with $w(A) < w(B) < w(C) < w(D)$

3 rules enforced in sequence:

1: $R_1 = \emptyset$, $n_1 = 0$, $R_2 = \{A, B\}$, $n_2 = 1$

2: $R_1 = \{B\}$, $n_1 = 0$, $R_2 = \{C, D\}$, $n_2 = 1$

← B modified, so recheck rule 1

3: $R_1 = \{B, D\}$, $n_1 = 0$, $R_2 = \{A, C\}$, $n_2 = 1$

If B or D are selected, then at most either A or C, not both, can be selected

→ Deselect A due to lower priority

●: regularity selected, □: regularity deselected
Rule Enforcement Example

Regularities $A, B, C, D$ with $w(A) < w(B) < w(C) < w(D)$

A B C D 3 rules enforced in sequence:

1: $R_1 = \emptyset$, $n_1 = 0$, $R_2 = \{A, B\}$, $n_2 = 1$

2: $R_1 = \{B\}$, $n_1 = 0$, $R_2 = \{C, D\}$, $n_2 = 1$

← $B$ modified, so recheck rule 1

3: $R_1 = \{B, D\}$, $n_1 = 0$, $R_2 = \{A, C\}$, $n_2 = 1$

← $A$ modified, so recheck rule 1

→ Cannot reselect $B$ due to rule 2

→ Final selection

●: regularity selected, □: regularity deselected
Numerical Solvability Test

- After initial selection each regularity is consecutively added to a constraint system in order of priority
- Each time the constraint system has to be checked for solvability:
  - System is solved as numerical minimization problem (numerically stable BFGS, BFGS/Gauss-Newton hybrid, ...)
  - System is solvable if least-squares error is $\sim 0$
  - Guarantees solvability up to numerical tolerance
- If system is solvable, the regularity is accepted
- Otherwise (conditional) selection rules involving the deselected regularity are used to check if alternative regularities can be selected
### Examples

<table>
<thead>
<tr>
<th>Detected Regularities</th>
<th>89</th>
<th>382</th>
<th>216</th>
</tr>
</thead>
<tbody>
<tr>
<td>After initial selection</td>
<td>34</td>
<td>263</td>
<td>156</td>
</tr>
<tr>
<td>Finally selected</td>
<td>23</td>
<td>117</td>
<td>93</td>
</tr>
<tr>
<td>Time</td>
<td>$\sim 2h$</td>
<td>$\sim 25h$</td>
<td>$\sim 23h$</td>
</tr>
</tbody>
</table>
Results

- Major regularities can be enforced on improved model
  - Symmetrically arranged face groups
  - Orthogonal/parallel directions, …

- Some problems due to ambiguities remain
  - Choice of special values for lengths and angles not always consistent
  - There is a choice between regularities of high quality and regularities with small errors

- Slow computation
  - Most of the time spent on solving constraint systems (one constraint system solved for each regularity)
  - Only proof of concept (see next slide)
Solvability Test Improvements

- To speed up algorithm: faster solvability test
- Detect structural inconsistencies without solving the constraint system
- Use degree-of-freedom analysis to build up a consistent constraint graph
- Solve the constraint system numerically only once
- Currently under investigation:
  - Speeds up solvability test from hours to minutes
  - Further work required to ensure selection of consistent system
Selection Improvements

- To improve quality of models: **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 block with edge lengths 2 and 2.1, or ... ?
  - Local decisions can favour the block arrangements
- Make decisions in the context of the whole model, not locally with respect to inconsistencies
- Reduce/replace user-defined constants for priorities by simpler methods based on multiple-choice questions and learning
Conclusion

- We have given a general approach to improve inaccurately reverse engineered geometric models
- The current numerical method improves models, but
  - Running time is too long
  - Regularity selection not consistent with respect to design intent
- Current work is addressing these problems