School of Computer Science
Cardiff University

Design Intent of Geometric Models

Frank C. Langbein

EPSRC GR/M78267 GR/S69085/01

The Nuffield Foundation NUF-NAL 00638/G

Design Intent

- Engineering converts a concept into an artifact
- Reverse engineering converts an artifact into a concept
- Design intent is a detailed representation of the concept
- Explicit representation of design intent required for high-level CAD applications
  - Description of intended properties of the object's shape (geometric regularities)
  - Different abstraction levels
    (e.g. "a cube", "6 parallel/orthogonal planes",
    "$n_1{}^t x - d_1 = 0$, $n_2{}^t x - d_2 = 0$, ...")
  - Additionally, represent functional properties, etc. (not considered here)

Design Intent in CAD Applications

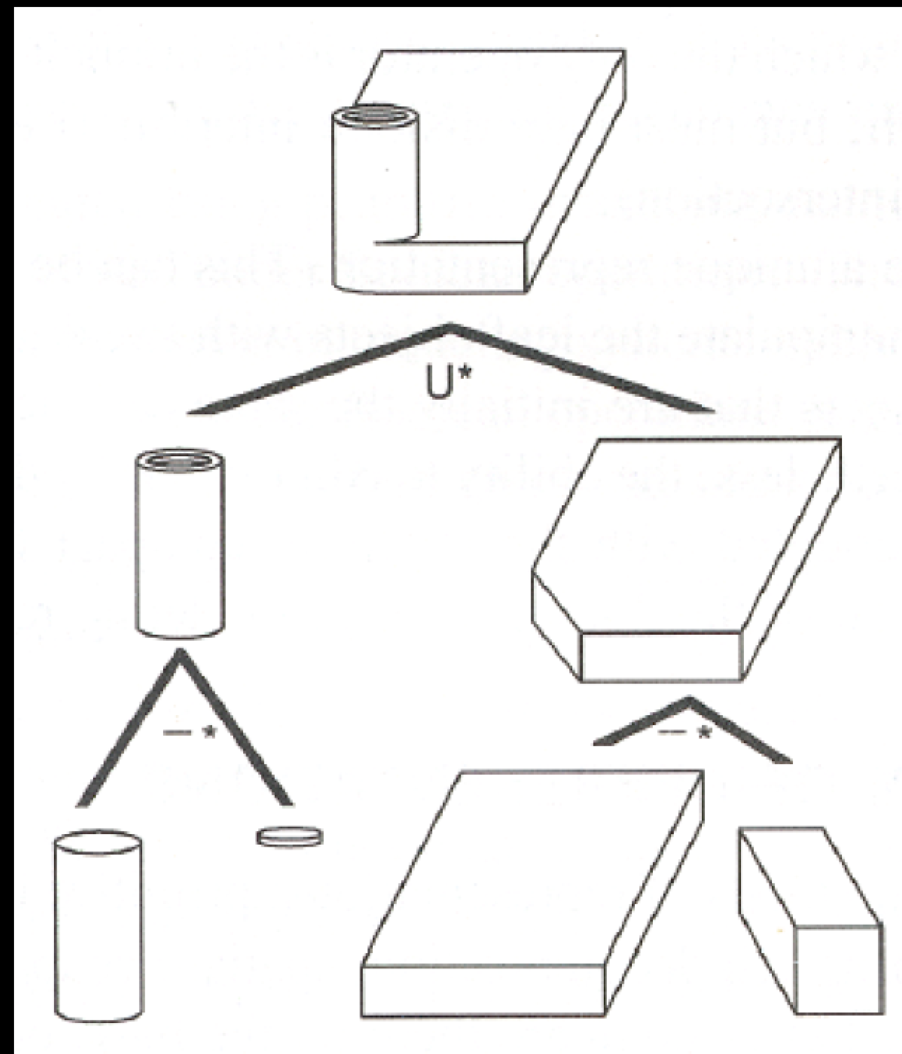
- High-level representation of design intent in CAD applications
  - Allow modifications and adjustments without destroying important properties unintentionally
  - Improve robustness of modelling operations
  - Enable data exchange between different applications without creating broken models or loosing important properties (Healing)
  - Analyse the model's properties

Approaches towards Design Intent

- Standard CAD model data structures do not explicitly represent design intent
  - Constructive Solid Geometry (CSG):
    union, intersection, etc. of primitive shapes
  - Boundary representation:
    faces, edges and vertices with geometry and topology (boundary relations)
- Extensions of above data structures for design intent:
  - Feature-based modelling
  - History-based modelling
  - Constraint-based modelling

Feature-based Modelling

- Describe model by machining, design, . . . features (holes, slots, pockets, . . .)
  - Common method for creating models
  - Hard to detect features (many alternative interpretations possible)
  - Features add semantics to CSG-type data structures

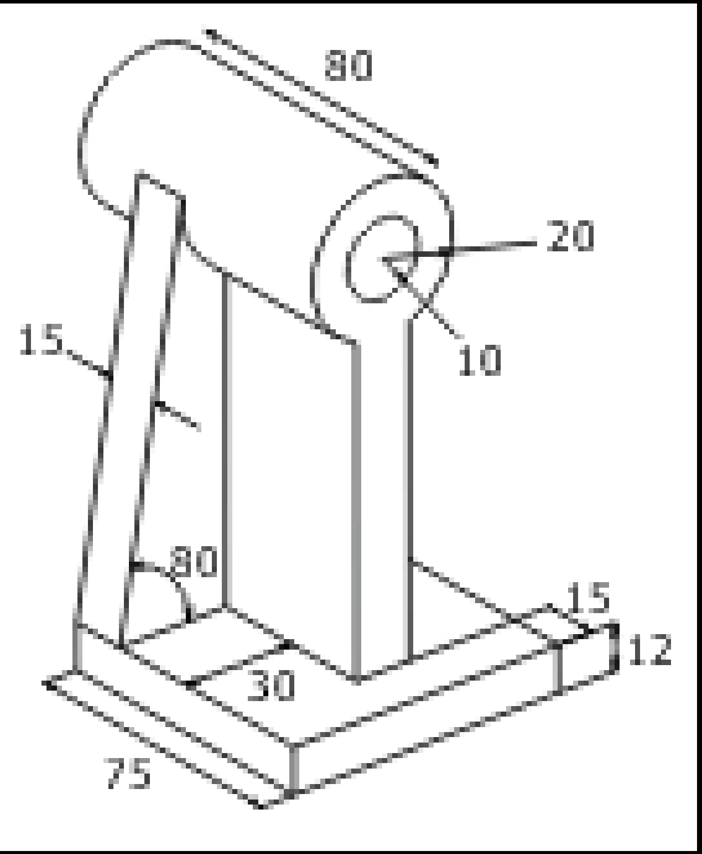


History-based Modelling

- Idea: Store the complete history of the model building operations
  - Edit object by changing the history and “replaying” it (or relevant part of it)
  - Edit operations are simpler and more robust
  - Proposed extension to STEP standard
- But complete history often contains irrelevant information
- Operations used to make object may contain hints for design intent

Constraint-based Modelling

- Specify desired relations between geometric objects by geometric constraints
  - One huge polynomial equation system describes the whole object
- Design intent specified exactly, but
  - Hard to find a solution
  - Under- and over-constrained cases are hard to determine by the user
  - Constraints only describe low-level relations



Forward and Reverse Problem

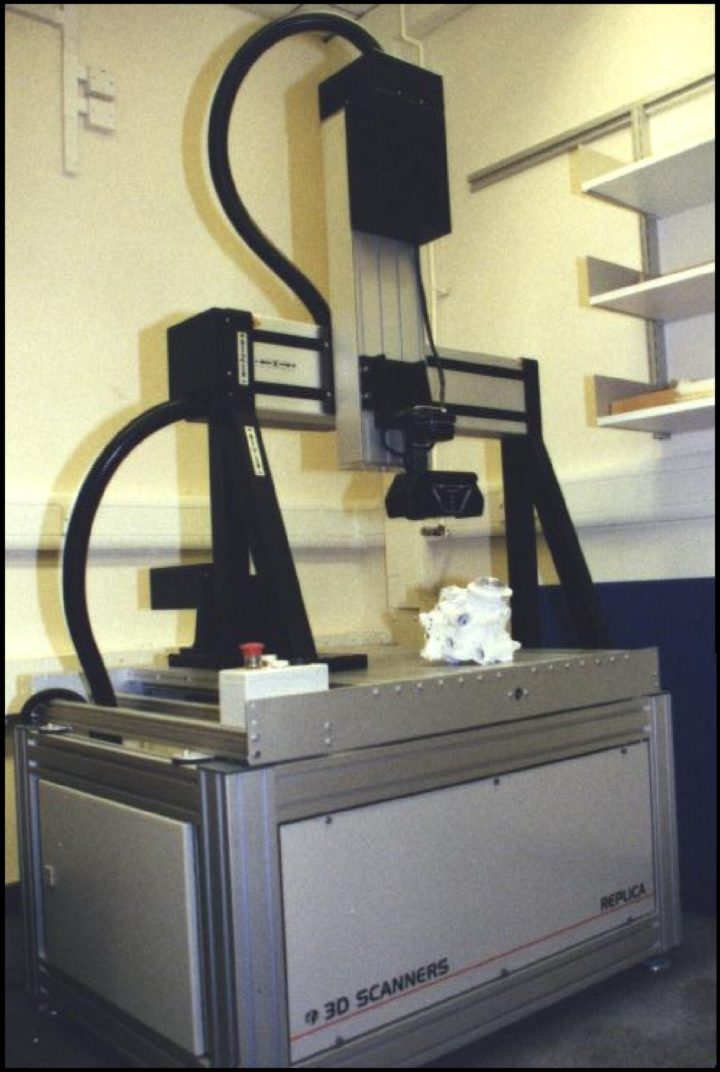
- Need an appropriate representation of high-level design intent
- *Forward* problem:
  - Record design intent during model creation
- *Reverse* problem:
  - Determine the design intent of a given model

Reverse Engineering

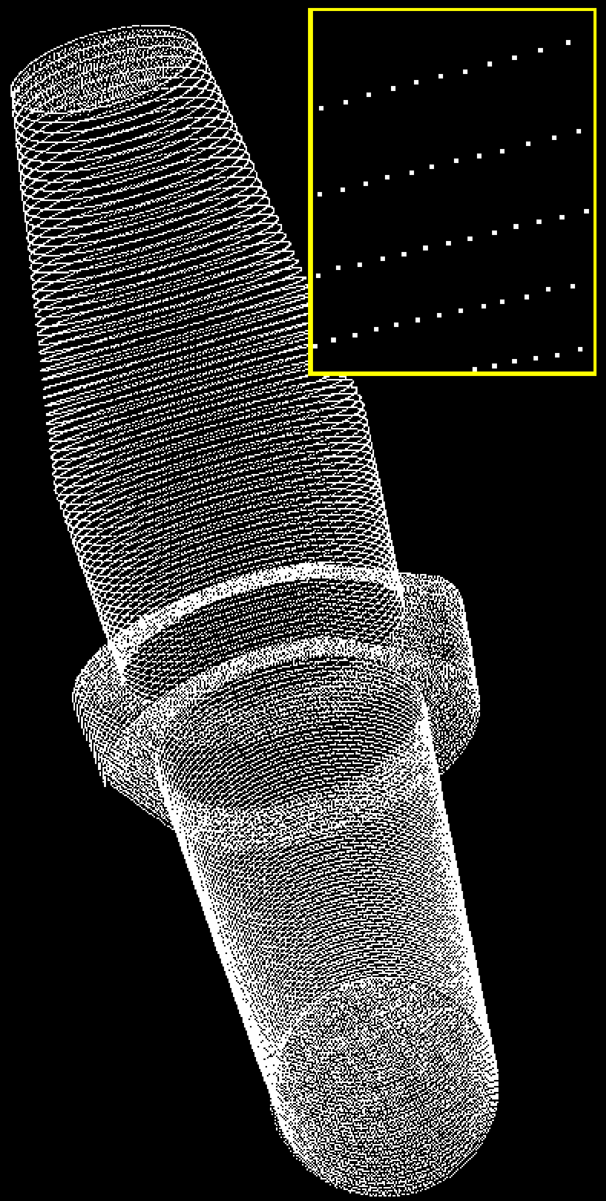
- Extract sufficient information from physical object for particular purpose
- For *reproduction* applications:
  - Exact information about shape of physical object is sufficient for one-to-one copy
- For *quality control* applications:
  - Exact shape information has to be compared with an original model
- For *redesign* applications:
  - Reconstructed model should exhibit exactly the same geometric properties as the original model

Reverse Engineering Process

- *Data Capture*



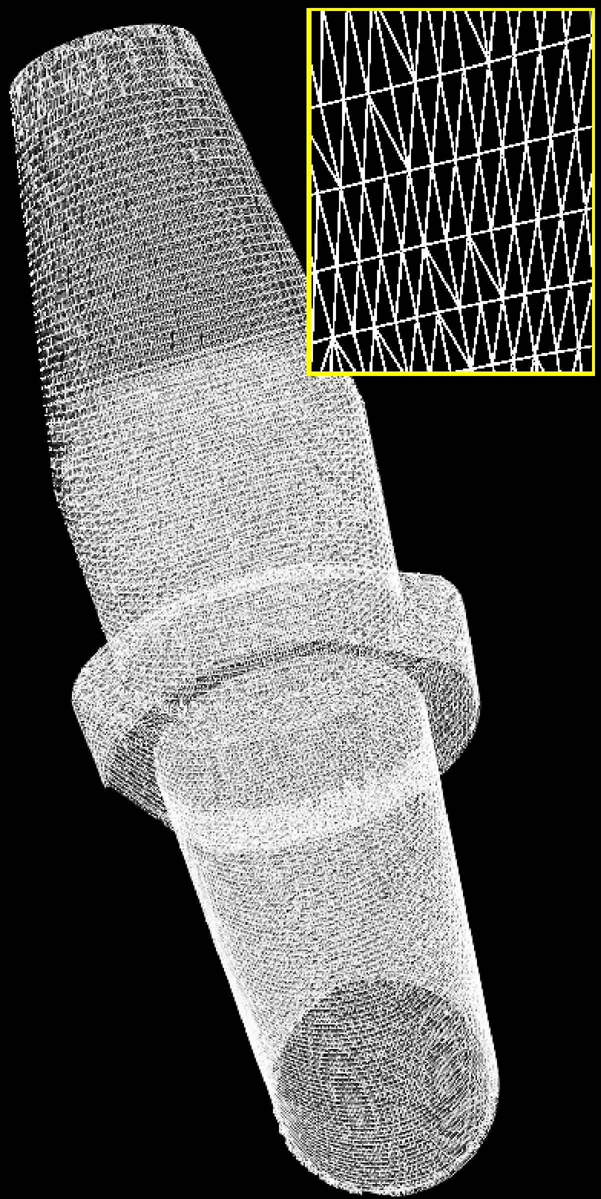
Reverse Engineering Process



➤ *Data Capture*

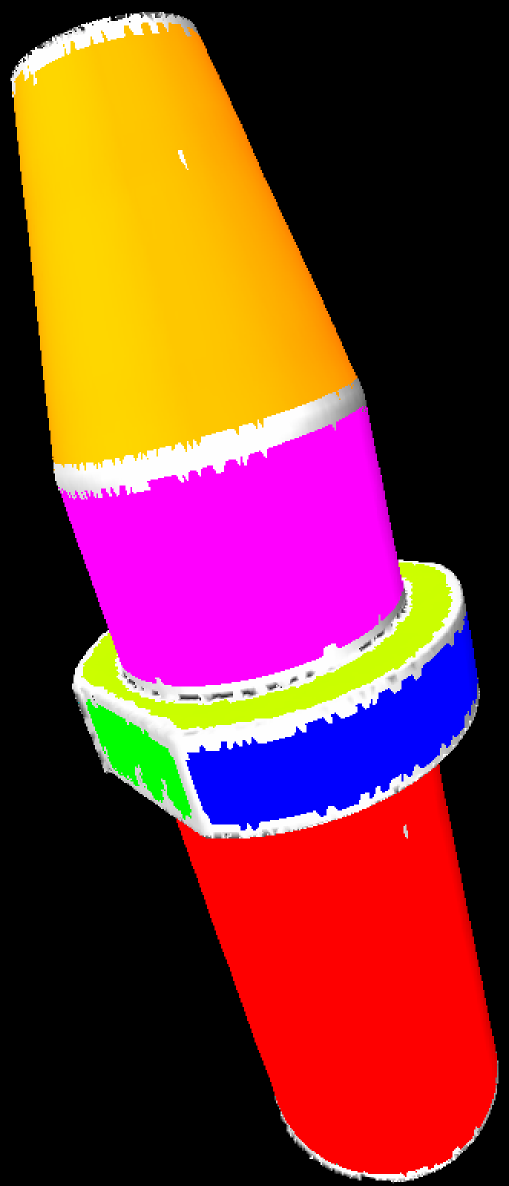
- Obtain multiple views from a 3D laser scanner
- Register views to a single 3D point set

Reverse Engineering Process



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

Reverse Engineering Process



- *Data Capture*
- *Triangulation*
- *Segmentation & Surface Fitting*
  - Split the point set into subsets representing natural surfaces
  - Find the surface type and fit a surface of this type for each subset

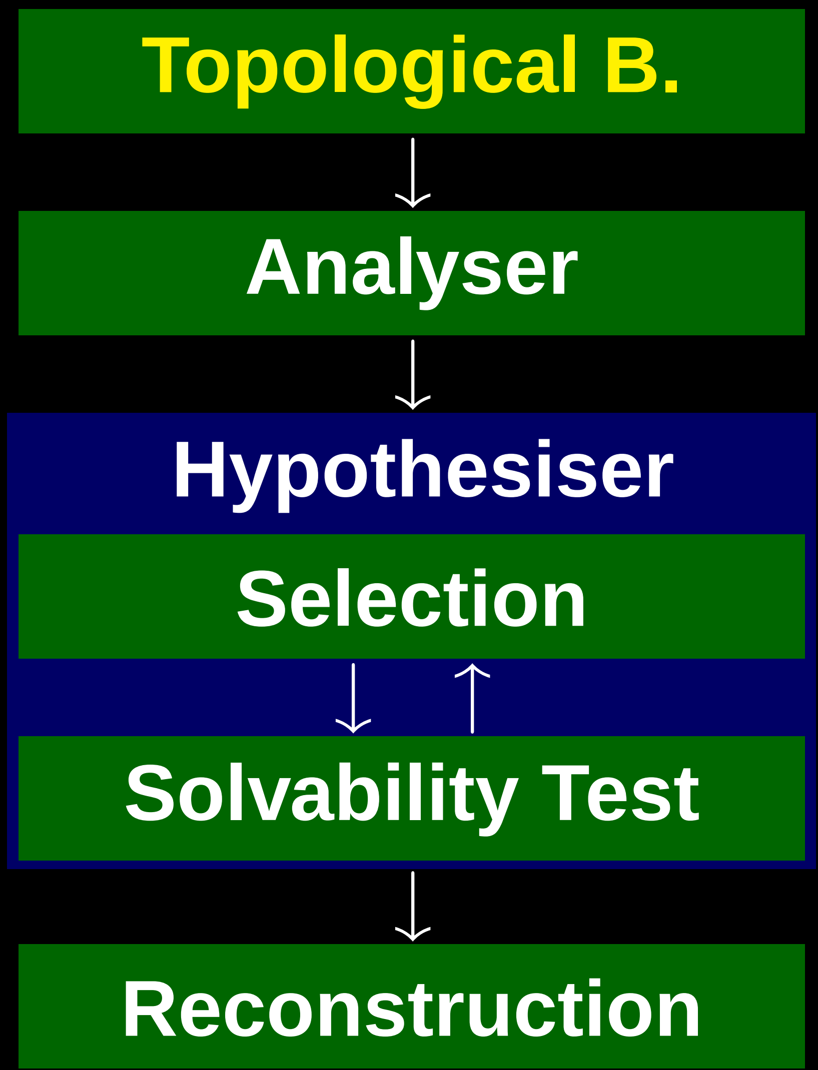
Reverse Engineering Process

- *Data Capture*
- *Triangulation*
- *Segmentation & Surface Fitting*
- *CAD Model Creation*
  - Create an initial solid model by stitching surfaces

Beautification

- *Problem:* Reverse engineered models suffer from inaccuracies caused by
  - sensing errors (data capture)
  - approximation and numerical errors (reconstruction)
  - possible wear of the object
  - manufacturing method used to make the object
- *Goal:* Reconstruct an ideal model of a physical object with intended geometric regularities
  - Design intent has to be considered at some stage
- *Beautification* aims to improve the reconstructed model in a post-processing step

Beautification Process



- *Local Topological Beautification*
  - Detect top. defects (gaps, pinched faces, small faces, sliver faces, short edges,. . .)
  - Repair defects by replacing faces with edges, edges with vertices, extending faces, . . .
- Defects are typically localised
  - Interaction between defects is limited to local faces
  - Gives well-defined sequence for repairing

Beautification Process

Topological B.

↓

Analyser

↓

Hypothesiser

Selection

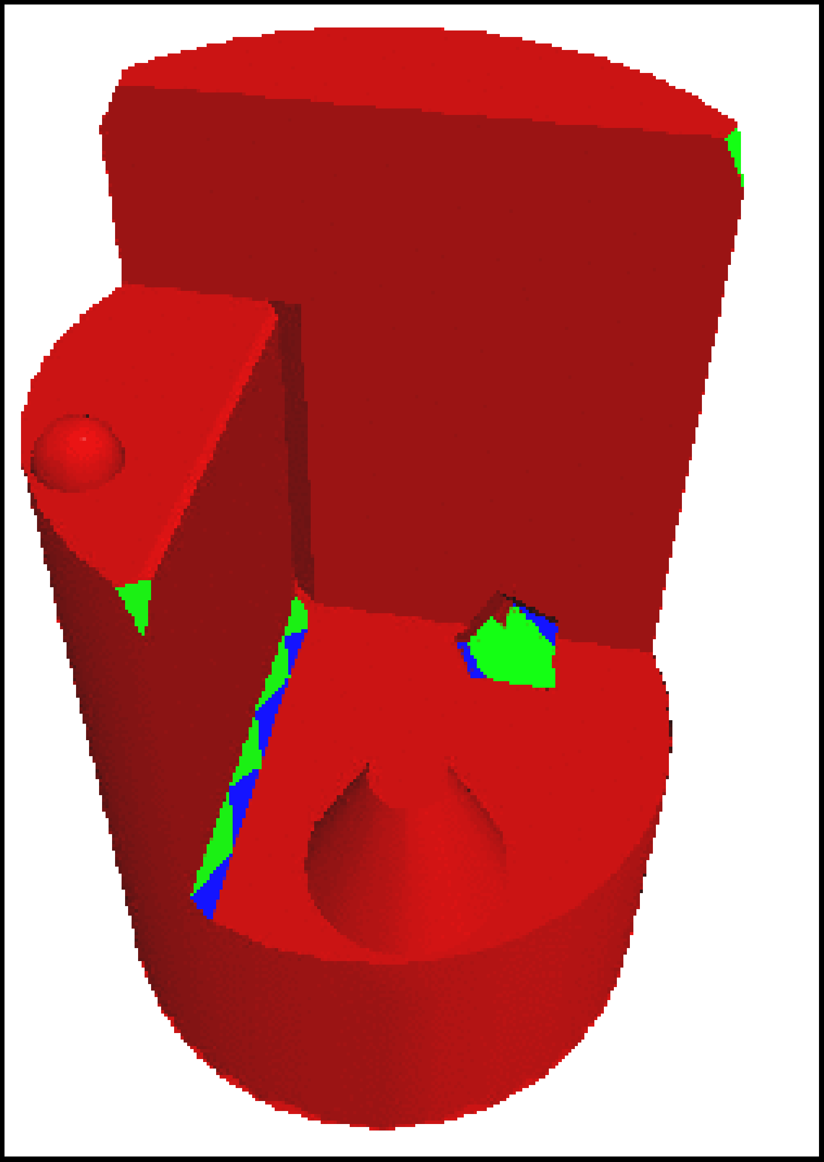
↓ ↑

Solvability Test

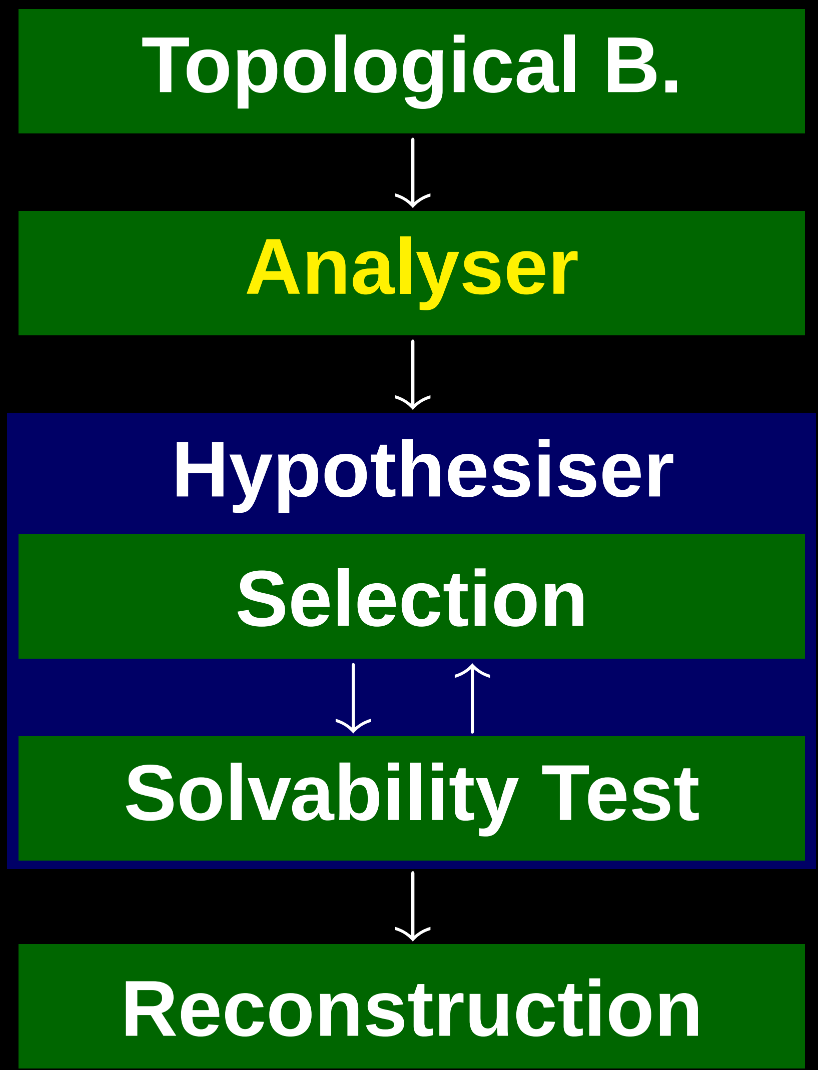
↓

Reconstruction

- *Local Topological Beautification*

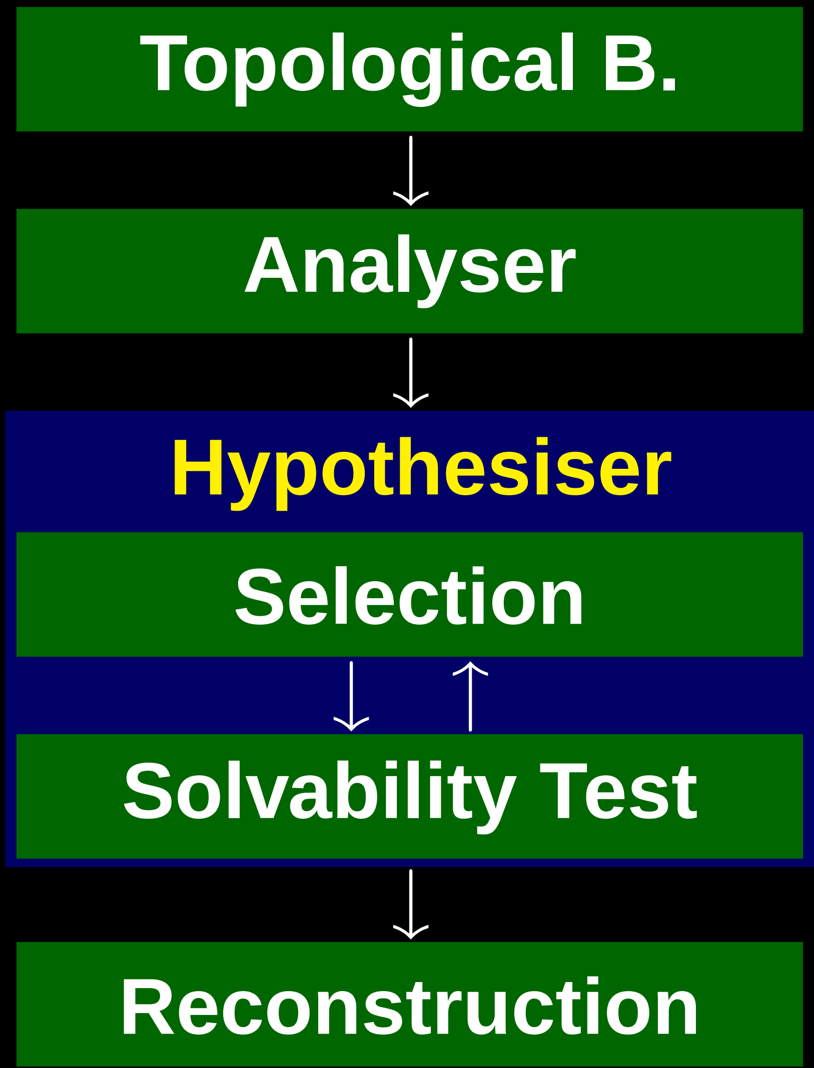


Beautification Process



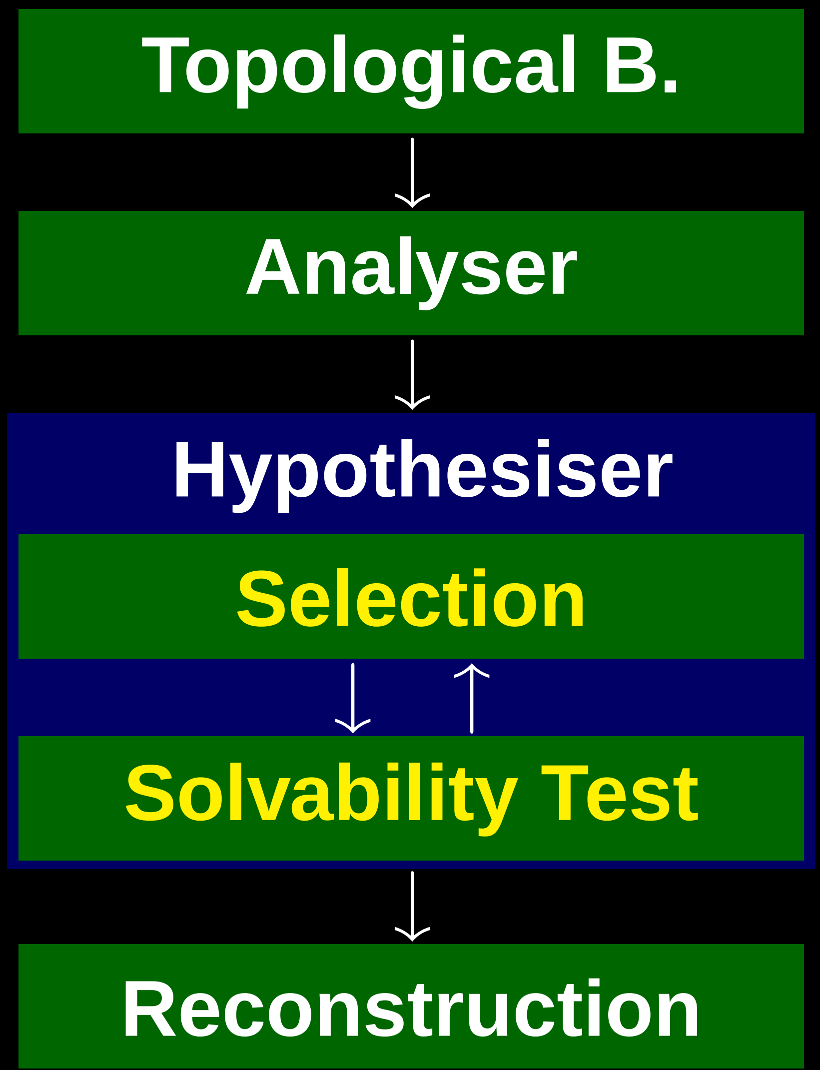
- Detect approximate geometric regularities
  - Approximate symmetric arrangement of faces, vertices, directions, etc.
  - Large number of potential regularities
  - Regularities may or may not be intended
- Exact conditions for approximate regularities are used rather than arbitrary tolerances

Beautification Process



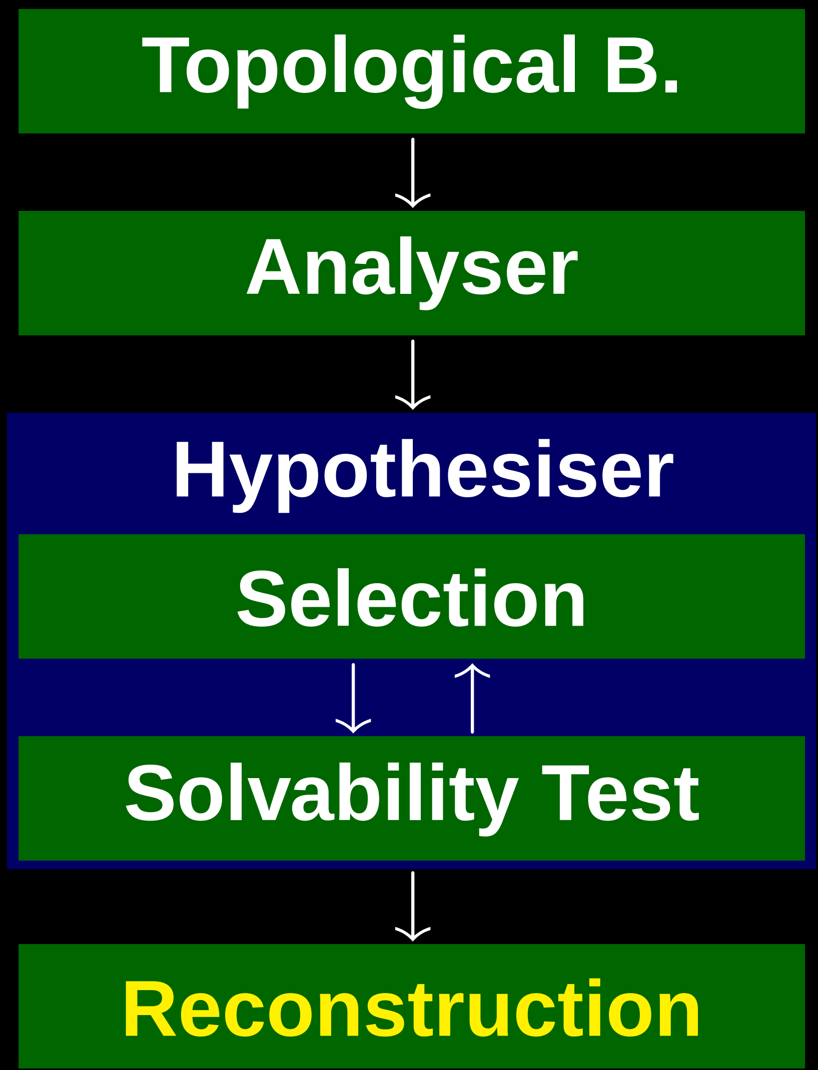
- Detected regularities are unlikely to be mutually consistent
- Have to select regularities consistent with respect to
  - design intent
  - simultaneous realisability (solvability)

Beautification Process



- Use geometric constraints to describe regularities
- Add regularities in order of a priority to a constraint system
- Only accept regularity if constraint system remains solvable
- Priority is based on
  - how common the regularity is
  - “desirability” of regularity
  - error of regularity in original model

Beautification Process



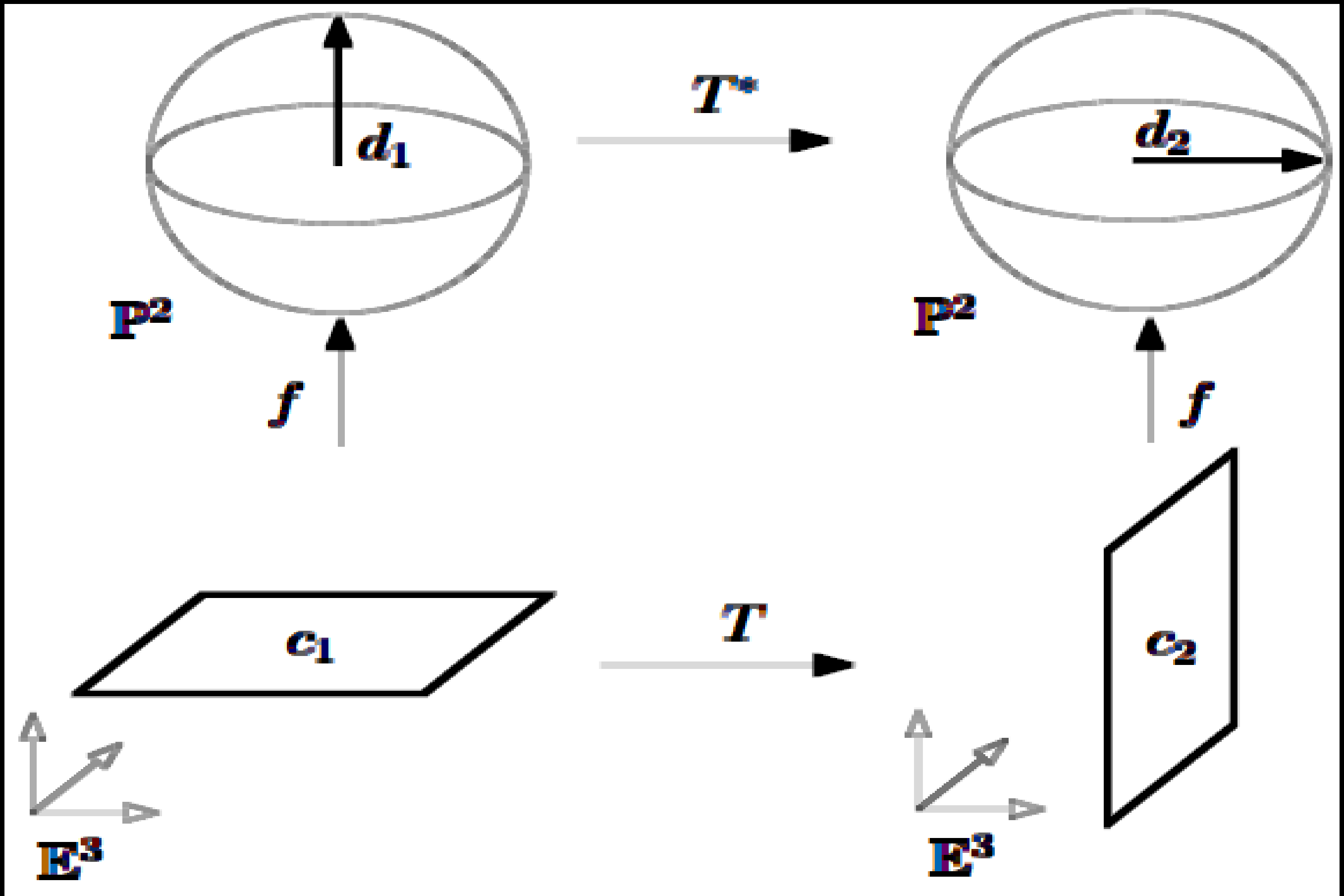
- Compute solution of constraint system
  - Numerical optimiser
  - Decomposition/recombination solver
- Rebuild model from solution and topology of original model
- Align model with coordinate axes, fix potential topological defects, . . .

Approximate Regularities

- Regularities are described as *symmetries* of *shape features*
  - Shape features describe properties of B-rep elements
  - Shape features are points in a feature space
- For beautification detect *approximate* symmetries of feature point sets
  - Point set symmetries are distance-preserving permutations
  - No pre-set tolerance
  - Seek tolerance levels where a local match implies a global match to ensure unambiguous regularities

Shape Features

- Features are properties of B-rep elements (faces, edges, vertices, sets of these elements)
  - Features change in a similar way to the element itself under isometric transformations



Regularity Types

Features	Regularity	Symmetries
Direction	Parallel directions	Identity
	Symmetries of directions	Isometries
	Rotational symmetries of directions like in regular prisms and pyramids	Rotations
Axis	Aligned axes	Identity
	Parallel axes arranged equi-spaced along lines and grids	Translations
	Parallel axes arranged symmetrically on cylinders	Rotations
	Axes intersecting in a point	Identity

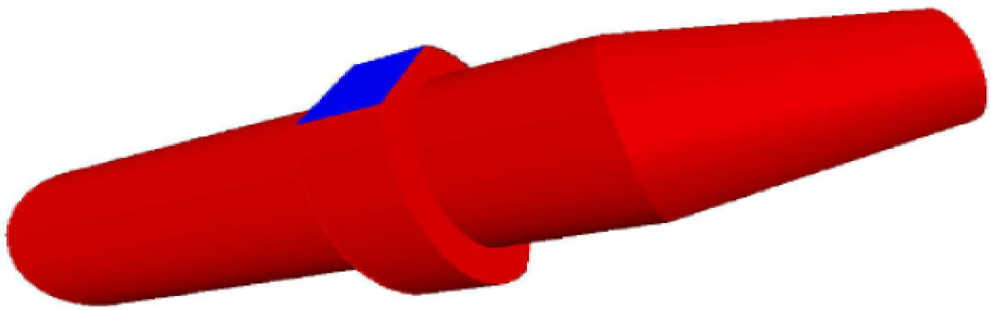
Regularity Types

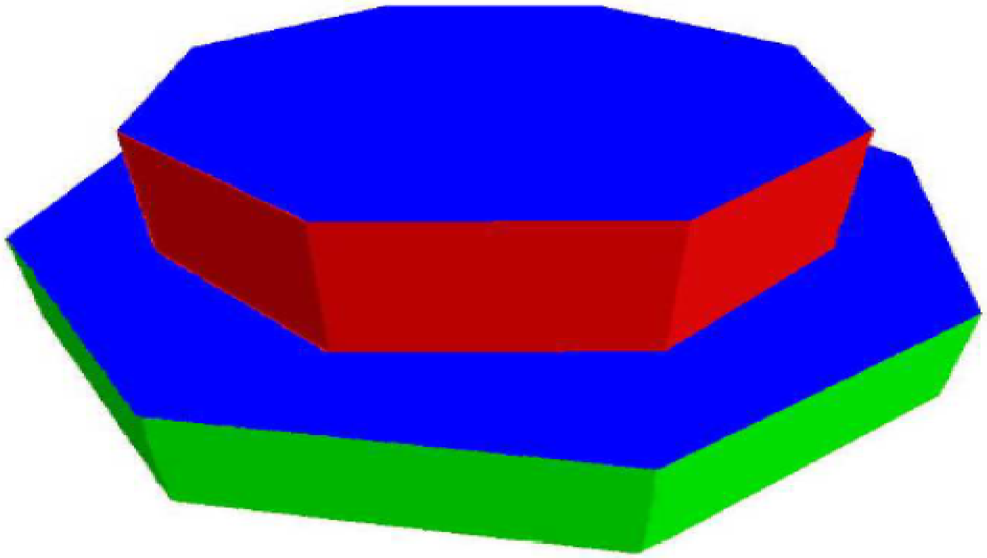
Features	Regularity	Symmetries
Position	Equal positions	Identity
	Point set symmetries	Isometries
	Equi-spaced positions arranged on a line or a grid	Translations
	Positions arranged symmetrically on a circle	Rotations
	Equal positions when projected on a special line or plane	Identity
Length / Angle	Equal scalar parameters	Identity
	Special scalar parameter values	(special value)
	Simple integer relations	(special value)

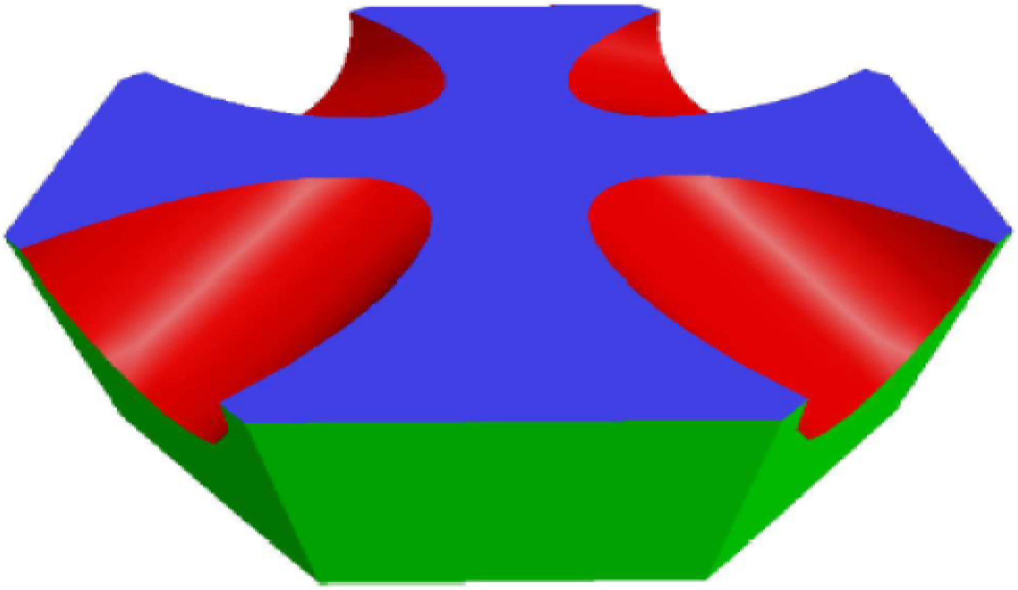
Regularity Detection

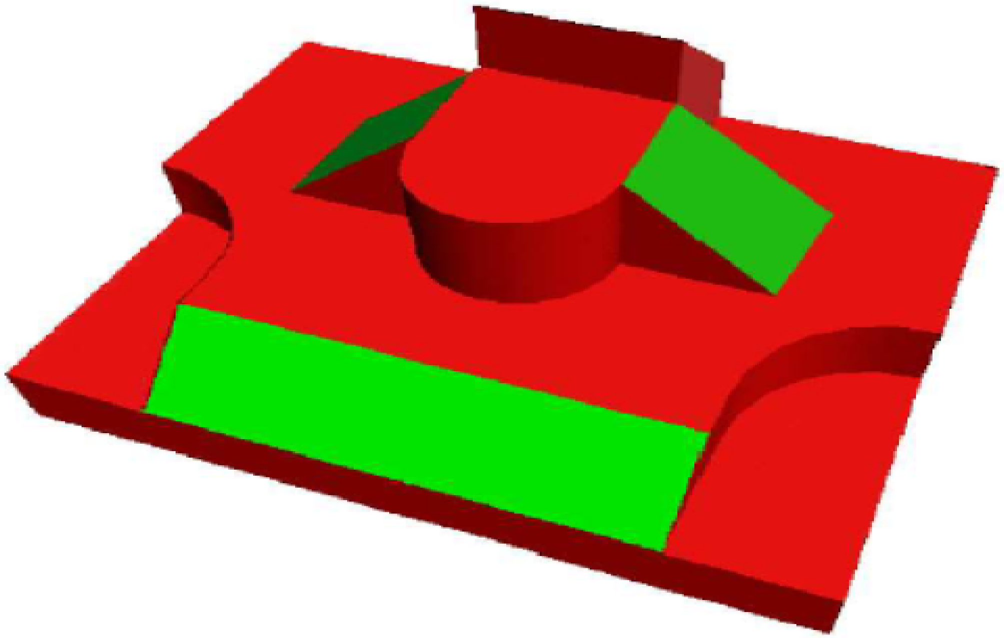
- Principle approach to detect approximate regularities
  I. Cluster shape features hierarchically
     – Transitive clusters: distance between features in same cluster is smaller than distance between features in different clusters
     – Ensures that local match gives a global match
  II. For each tolerance level in the cluster hierarchy: Determine approximately distance-preserving permutations
- Exact algorithm depends on symmetry type (global symmetries of point sets, partial symmetries of directions, incomplete symmetries. . .)

Beautification Examples









Problems of Current Approach

- Current system can improve simple to medium complexity models
  - Independent, major regularities relating to most of the faces (global symmetries, orthogonal systems)
  - Desirable regularities with high accuracy
- Problems in selecting regularities:
  - Individual regularities rather than combinations
  - Many dependent, ambiguous regularities for complex models
- For complex models selected regularities are consistent w/r to solvability, but not w/r to design intent

Hierarchical Decomposition

- Regularity detection for complex models
  - Many ambiguous regularities
  - Topological structure not considered (only regular arrangements of shape features)
- Often complex models can be partitioned into interesting sub-parts (feature-based modelling)
  - Beautification in one step has to deal with many ambiguous regularities
  - Handling sub-parts separately may reduce number of regularities

Hierarchical Decomposition

- Approach to *hierarchical beautification*
  - *Partition* model hierarchically into suitable sub-parts
    - Requires rules for partitioning
    - E.g. determine symmetry breaks and take model apart such that sub-parts are more symmetric
  - Beautify sub-parts separately
  - *Re-combine* sub-parts
    - Requires suitable relations between sub-parts
    - E.g. use relative relations between sub-parts (rotations, translations, etc. to specify relative positions and symmetries)

Conclusion

- Design intent is essential for handling geometric models on a high abstraction level
- Beautification provides useful concepts for design intent in general
- Symmetry allows to describe and detect many types of geometric regularities
  - Creating is symmetry breaking
  - Recovering is symmetry building
- Main problem is still to include design intent directly in the model representation, modelling operations, etc.