A Methodology for automatic retrieval of similarly shaped machinable components – Research Update

Mark Ascher - Dept of ECE

Chandan Pitta – Dept of ECE
Motivation

- Retrieval of similarly shaped components can:
  - Add functionality to existing CAD databases
  - allow for the reuse of process plans which can both speed up and reduce the cost of development.

Challenges

- Retrieval of similarly shaped components has many challenges:
  - Multiple interpretations
  - Interacting features
  - Topological differences do not guarantee component dissimilarity
  - Graph matching solution is computationally intensive
System Overview

Query In B-Rep

Feature Extractor

Spatial Logic

Spatial Logic

Geometric Modeler

Query Relaxation

Block Handler

Component Database

Query Evaluator

Query Answer

User Interface
Related Work

- **Shape Based Similarity Retrieval** (Eakins)
  - Two dimensional parts
  - retrieved complete components

- **Volumetric Reasoning** (Lee et al) and **Planar Reasoning** (Cohn et al)
  - Groundwork for symbolic volumetric reasoning
  - Does not address part matching

- **Content Retrieval From Images Based on Knowledge of Shape** (Hsu et al)
  - Worked with medical images
  - Presented the Type Abstraction Hierarchy Concept
Related Work

- **3D Model Shape Based Similarity Retrieval** (Osada et al, Regli et al)
  - Uses D2 Distance measures
  - Works well for simple models

- **Feature Based Model Retrieval** (Regli et al)
  - Retrives complete models
  - Feature interaction representation too simplistic
  - No method for indexing

- **Group Technology**
  - Goal is to group components by similar machining processes for improved factory flow
  - Similar machining processes does not guarantee shape similarity
Component Representation In the Database

- Data Represented in Three Layers
  - Raw B-Rep Component (Representation Layer)
  - Graph of Maximal Features and Interactions (Knowledge Layer)
  - Histograms of Features and Abstracted Interactions (Semantic Layer)
- Interactions Encoded at Multiple Abstraction Levels
  - Interaction Matrix
  - Set of Face Pairs
  - Abstracted Interaction
ACIS BREP Extraction

ACIS is a robust geometric modeler whose output contains both Geometric and Topological data.

- Too much information
- Complexity of data interaction
  - Example a face:

**Plane-Surface:**
- normal
- surface data

**Face:**
- Next Loop
- Surface Direction sidedness

**Loop:**
- next loop
- co-edge face

**Co-edge:**
- next previous
- next on edge direction
- loop edge

**Edge:**
- end1
- end2
- coedge
curve direction

**Vertex:**
- edge point

**Point:**
- edge point
- position

**Straight-Curve:**
- position
direction

**Point:**
- position
ACIS BREP Extraction

- Program written to extract data needed
  - Read in ACIS File
  - Parse Into Objects
  - Create BREP files in Following Format:

```
face 4
vertex 5 10 -20
vertex 5 20 -20
vertex 5 20 -5
vertex 5 10 -5

face 6
vertex -5 20 -20
vertex -5 10 -20
vertex -5 10 -5
vertex -5 20 -5

face 10
vertex -5 10 20
vertex -5 20 20
vertex -5 20 5
vertex -5 10 5

face 15
vertex 5 20 20
vertex 5 10 20
vertex 5 10 5
vertex 5 20 5

face 23
vertex 5 20 20
vertex 5 10 20
vertex 5 10 20
vertex 5 20 20
vertex -20 20 20
vertex 20 -20 20
vertex 20 20 20
```
Demonstration of MakeBREP
Feature Representation
Overview

- Implementation is Based on “Constraint Based Feature Recognition” by Ming-Hsuan Yang and Dr. Michael Marefat
- Features are represented using a set of Predicates
- Basic Features: Pocket, Hole, Slot, Step, Blind-slot, Blind-step
# Pre-defined Predicates

## Table 1: Predefined predicates

<table>
<thead>
<tr>
<th>Predicate</th>
<th>Semantics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Opp($f_1$, $f_2$)</td>
<td>Two faces with opposite normals</td>
</tr>
<tr>
<td>Perp($f_1$, $f_2$)</td>
<td>Two faces with perpendicular normals</td>
</tr>
<tr>
<td>Adj($f_1$, $f_2$)</td>
<td>Two adjacent faces</td>
</tr>
<tr>
<td>Type($f_1$, R), Type($f_2$, B)</td>
<td>Type of Faces: Real, Border</td>
</tr>
<tr>
<td>Concave($f_1$, $f_2$)</td>
<td>Two adjacent faces that form a concave angle</td>
</tr>
<tr>
<td>Convex($f_1$, $f_2$)</td>
<td>Two adjacent faces that form a convex angle</td>
</tr>
<tr>
<td>Unifiable($f_1$, $f_2$)</td>
<td>Two faces that are unifiable</td>
</tr>
<tr>
<td>Num-Of-Opp-Face($f_1$)</td>
<td>Number of Opp faces $f_1$ has</td>
</tr>
<tr>
<td>Num-Of-Real-Face($C_1$)</td>
<td>Number of real faces of feature template $C_1$</td>
</tr>
</tbody>
</table>

![Blind Slot](image)

**Figure 1: A blind slot**
A feature $C$ is a set of conjunctions of predicates within the geometric constraint language.

$$C \leftarrow P_r \land P_b$$

$$P_r = \text{Opp}(f_1, f_3) \land \text{Perp}(f_1, f_2) \land \text{Perp}(f_2, f_4) \land \text{Perp}(f_1, f_5) \land \text{Perp}(f_2, f_6) \land \text{Perp}(f_3, f_5) \land \text{Type}(f_1, R) \land \text{Type}(f_2, R) \land \text{Type}(f_3, R) \land \text{Type}(f_4, R) \land \text{Adj}(f_2, f_3) \land \text{Adj}(f_1, f_3) \land \text{Adj}(f_1, f_6) \land \text{Adj}(f_2, f_6) \land \text{Adj}(f_3, f_5) \land \text{Concave}(f_1, f_2) \land \text{Concave}(f_3, f_5) \land \text{Concave}(f_1, f_6) \land \text{Concave}(f_2, f_3) \land \text{Concave}(f_2, f_6) \land \text{Concave}(f_3, f_6)$$

$$P_b = \text{Opp}(f_2, f_4) \land \text{Opp}(f_5, f_6) \land \text{Type}(f_4, B) \land \text{Type}(f_5, B) \land \text{Adj}(f_1, f_4) \land \text{Adj}(f_3, f_4) \land \text{Adj}(f_4, f_5) \land \text{Adj}(f_2, f_5) \land \text{Adj}(f_3, f_5) \land \text{Adj}(f_4, f_5) \land \text{Adj}(f_4, f_6)$$

**Figure 1: A blind slot**
Feature Recognition Overview

• Geometric Constraint Satisfaction Feature Recognition
  – Predicates for Canonical Features Known
  – Predicates for Input Component are calculated
  – Matching is performed by comparing Calculated Predicates against Canonical Feature Predicates
  – Constraints are matches between feature predicates and component predicates
Feature Recognition Overview

Continued…

- Constraints can be relaxed by ignoring it
  - Opposite face constraints cannot be relaxed
  - Border and Real face type constraints cannot be relaxed
  - Number of relaxations input by the user
Why do we need Relaxations
Practical Implementation

- MakePredicates.c
  - Needs Object file (BREP) and Delta file (BREP)
  - Identifies Border and Real faces
  - Generates normals for faces in Delta Volume
  - Defines predicates using normals and writes all these details to a file with extension .predicates
Predicates File Format

f b X 1 5 P 0 3
f r A 1 5
f r P 1 5
f r X 1 4
f b A 1 4
f b P 1 4
f b X 1 3
O 3 5
X 3 5
A 3 5
P 3 5
X 3 4
A 3 4
P 3 4
X 2 5
A 2 5
P 2 5
X 2 4
A 2 4
P 2 4
O 2 3
P 0 3
X 0 2
A 0 2
P 0 2
O 0 1
Demonstration of makePredicates.c
Practical Implementation of Feature Recognition

- featureRecognition.c
  - Requires
    - BREP files of Object and Delta Volume
    - Predicates of Component and canonical features
  - Identifies possible existences of Pocket, hole, blind slot, slot, blind step and step by
    - Making Nodes Consistent
    - Sorting on Cardinality
    - Making Arcs Consistent to find features and relaxing if necessary
    - Eliminating in consistent Features
  - Removing Duplicate features
Constraint Graph

- Node Consistency
- Cardinality
- Arc Consistency
Demonstration of featureRecognition.c
Feature Graph

- Generate graph of Features and Interactions
  - Nodes Represent Features
    - Maximum vertex
    - Minimum vertex
    - Face Information (Accessibility, Real or Border)
    - ID
    - Type
  - Arcs Represent Interactions
    - Starting Feature
    - Ending Feature
    - Matrix
    - Face Pairs
    - Type
    - ID
Interactions

- Interactions Represented as an nxm matrix where:
  - n is the number of faces in feature 1 (f1)
  - m is the number of faces in feature 2 (f2)
  - Entries are in the set \{ +, -, s, i \}
    - + indicates that f1 lies in the positive half space of f2
    - - indicates that f1 lies in the negative half space of f2
    - s indicates they lie on the same plane
    - I indicates that the features interact

\[ R^{5,2} \]
Interactions Continued

- If only orthonormal features are considered
  - Results in 6x6 interaction Matrices
  - The near diagonal data points contain the pertinent data
  - The number of unique columns is reduced to 5 ordered types that are physically valid
    - The following notation indicates (same, opposite) face
      - (-,+): No Interaction
      - (-,s): Indicates a shared face with no internally shared points
      - (-,-): Indicates a face that is interior to both faces of the other feature
      - (s,-): Indicates a shared face with internally shared points
      - (+,-): Indicates a face that is not interior to one face of the other feature
  - The following Combinations are invalid:
    - (+,+) can not be outside both faces parallel to the face of interest
    - (s,s) can not be the same as two faces which bound a feature
    - (+,s) and (s,+) can not be same as one face and outside the other
Interactions Continued

Note: Arrows Point to interior of second feature

(-,+)
(-,s)
(-,-)
(s, -)
(+,-)

-1  0  1  2  3
Interactions Continued

Pairs of parallel Faces are compared Resulting in 7 Feasible Unordered face-pair Combinations:

<table>
<thead>
<tr>
<th>Type</th>
<th>Face 1</th>
<th>Face 2</th>
<th>Relation of faces to other Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>(s,-)</td>
<td>(s,-)</td>
<td>Both Faces Shared</td>
</tr>
<tr>
<td>B</td>
<td>(s,-)</td>
<td>(-,-)</td>
<td>One Face Shared Other Inside</td>
</tr>
<tr>
<td>E</td>
<td>(s,-)</td>
<td>(+,-)</td>
<td>One face shared, Other Outside Opp</td>
</tr>
<tr>
<td></td>
<td>(s,-)</td>
<td>(-,s)</td>
<td>Not Possible</td>
</tr>
<tr>
<td>A</td>
<td>(-,-)</td>
<td>(-,-)</td>
<td>Both Faces Inside</td>
</tr>
<tr>
<td>C</td>
<td>(-,-)</td>
<td>(+,-)</td>
<td>One Face Inside, Other Outside</td>
</tr>
<tr>
<td></td>
<td>(-,-)</td>
<td>(-,s)</td>
<td>Not Possible</td>
</tr>
<tr>
<td>F</td>
<td>(+,-)</td>
<td>(+,-)</td>
<td>Both Outside</td>
</tr>
<tr>
<td>G</td>
<td>(+,-)</td>
<td>(-,s)</td>
<td>One Face Shared, Other Outside</td>
</tr>
<tr>
<td></td>
<td>(-,s)</td>
<td>(-,s)</td>
<td>Not Possible</td>
</tr>
</tbody>
</table>
Interactions Continued

Inverse Relations Exist for some Face-pair Types.

<table>
<thead>
<tr>
<th>Type</th>
<th>Face 1</th>
<th>Face 2</th>
<th>Inverse Interaction</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>(s,-)</td>
<td>(s,-)</td>
<td>D</td>
</tr>
<tr>
<td>B</td>
<td>(s,-)</td>
<td>(-,-)</td>
<td>E</td>
</tr>
<tr>
<td>E</td>
<td>(s,-)</td>
<td>(+,-)</td>
<td>B</td>
</tr>
<tr>
<td>A</td>
<td>(-,-)</td>
<td>(-,-)</td>
<td>F</td>
</tr>
<tr>
<td>C</td>
<td>(-,-)</td>
<td>(+,-)</td>
<td>C</td>
</tr>
<tr>
<td>F</td>
<td>(+,-)</td>
<td>(+,-)</td>
<td>A</td>
</tr>
<tr>
<td>G</td>
<td>(+,-)</td>
<td>(-,s)</td>
<td>G</td>
</tr>
</tbody>
</table>
Several Face-pair types indicate a family of interactions.

<table>
<thead>
<tr>
<th>Type</th>
<th>Face 1</th>
<th>Face 2</th>
<th>Interaction Family</th>
</tr>
</thead>
<tbody>
<tr>
<td>E</td>
<td>-s</td>
<td>-+</td>
<td>Intrusion</td>
</tr>
<tr>
<td>C</td>
<td>--</td>
<td>-+</td>
<td>Improper Interaction</td>
</tr>
<tr>
<td>F</td>
<td>-+</td>
<td>-+</td>
<td>Pass Through</td>
</tr>
<tr>
<td>G</td>
<td>-+</td>
<td>+s</td>
<td>Face</td>
</tr>
</tbody>
</table>
Show All Interactions Here

All the above interactions support inverse except for PT-3, PT-2, PP-2b, & 2-2 which are asymmetric relations.
Interactions Continued

Each Interaction can be represented by three Face-pairs

<table>
<thead>
<tr>
<th></th>
<th>Normal</th>
<th>Inverse</th>
</tr>
</thead>
<tbody>
<tr>
<td>pi-0</td>
<td>A</td>
<td>E</td>
</tr>
<tr>
<td>pp-0</td>
<td>A</td>
<td>F</td>
</tr>
<tr>
<td>pi-1</td>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>pp-1</td>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>l-1</td>
<td>A</td>
<td>C</td>
</tr>
<tr>
<td>ii-1</td>
<td>A</td>
<td>C</td>
</tr>
<tr>
<td>f-1</td>
<td>A</td>
<td>C</td>
</tr>
<tr>
<td>ip</td>
<td>A</td>
<td>C</td>
</tr>
<tr>
<td>pi-2b</td>
<td>A</td>
<td>D</td>
</tr>
<tr>
<td>pp-2b</td>
<td>A</td>
<td>D</td>
</tr>
<tr>
<td>fi-1</td>
<td>A</td>
<td>E</td>
</tr>
<tr>
<td>fp</td>
<td>A</td>
<td>F</td>
</tr>
<tr>
<td>pi-2a</td>
<td>B</td>
<td>B</td>
</tr>
<tr>
<td>pp-2a</td>
<td>B</td>
<td>B</td>
</tr>
<tr>
<td>ib-1</td>
<td>B</td>
<td>C</td>
</tr>
<tr>
<td>fb-1</td>
<td>B</td>
<td>C</td>
</tr>
<tr>
<td>fi-2</td>
<td>B</td>
<td>E</td>
</tr>
<tr>
<td>l-2</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>ib-2</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>f-2</td>
<td>C</td>
<td>C</td>
</tr>
</tbody>
</table>
Feature Graph Creation Algorithm

- For feature in component:
  - Create Node

- For Node1 in component
  - For Node2 with ID Greater than Node1
    - If Node1 interacts with Node2
      - Compare Faces and Generate Matrix and Face Pairs
      - Determine Face Type
Maximal Feature Sub-Graph Extraction

- For Node in Component Graph
  - If Node is subsumed (no 3s) or
  - If Node Ends inside another feature
    - Delete Node
- For Arc in Component Graph
  - If arc associated with Node
    - Delete Arc
Demonstrate msfgExtract
Future Work

- Block Extractor
- Generate Database of Parts
  - MySQL to be used
- Query Relaxation Unit
  - Two methods being considered
- Query Evaluation Unit
  - Perform graph matching on the candidate matches
- Implement Spatial Logic Unit
  - Complexity Dependent on Query Relaxation Technique