INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.
NOTE TO USERS

This reproduction is the best copy available.

UMI
A HYBRID APPROACH TO AUTOMATIC FEATURE RECOGNITION FOR MULTIPLE APPLICATIONS IN CONCURRENT ENGINEERING

DISSERTATION

Presented in Partial Fulfillment of the Requirements for
the Degree Doctor of Philosophy in the Graduate
School of The Ohio State University

By

Heng Zhu, M.S.

The Ohio State University
2000

Dissertation Committee:
Professor Chia-Hsiang Menq, Adviser
Professor Gary L. Kinzel
Professor Henry R. Busby
Professor Blaine W. Lilly

Approved by

Adviser

Department of Mechanical Engineering
ABSTRACT

Industry's demand of new mechanical components and the increased complexity of them continue to challenge the Computer-Aided Design (CAD) community as well as Computer-Aided Manufacturing (CAM) community. Particularly, the rising interest in life-cycle design optimization and concurrent engineering results in a major change of the functional role of a geometric modeler from being a pure geometric modeling tool to being a common information model that supports life-cycle design and various downstream applications in concurrent engineering environment. Feature, a high-level semantic entity with engineering meanings, has been widely regarded as the key to building the desired information model supporting CAD/CAM integration. Two major approaches to feature-based model generation exist in current feature research: feature-based design and automatic feature recognition. Feature-based design allows designers to create a feature-based model from scratch by using a set of predefined features and it has matured to the point that results have been incorporated into commercial CAD systems. For legacy CAD data without feature information, automatic feature recognition is needed to build a feature-based model. Although automatic feature recognition has been studied for almost two decades and some results have come out, some open technical issues have not yet been resolved satisfactorily. The goal of this research work is to
develop an information framework that can bridge the gap between the geometric data of a design work and the higher level geometric abstraction that supports complex reasoning incurred in concurrent engineering applications. In order to accomplish this goal, two specific objectives of this research are identified. The first one is to investigate the underlying invariant properties of boundary entities of CAD models and to characterize as well as group them into surface features. Although surface feature recognition can deal with various shapes by general curvature analysis, it can not handle feature interactions very well due to its boundary-based characteristic. Thus feature volume extraction is introduced to facilitate the achievement of robust feature interaction resolving and the generation of multiple feature interpretations. The second objective is to establish an algorithmic foundation so as to construct and manipulate the volumetric features based on the results from the first objective. The resulting volumetric features can yield parameterized feature shapes and lead to a feature-based model. Based on the approaches developed for the two objectives, a hybrid feature recognition method is proposed to support geometric reasoning needed in various downstream manufacturing applications and to facilitate concurrent design optimization. The proposed hybrid feature recognition method leads to a feature reasoning kernel which consists of four functional modules: pre-recognition processor, surface feature recognizer, volumetric feature recognizer and post-recognition processor. The pre-recognition processor is used to simplify the input B-Rep model by automatic fillet/round suppressing for efficient feature recognition. The surface feature recognizer is designed to demonstrate the implementation of the proposed methods that transform the discrete boundary entities of a CAD model into meaningful groupings, called surface features. The volumetric feature recognizer extracts feature
volumes based on surface features and resolves feature interactions incurred in real-world engineering designs. The post-recognition processor manipulates the feature volumes according to the desired applications and process requirements and generates multiple interpretations of the parameterized feature shapes. Based on the proposed hybrid feature recognition method, a feature recognition prototype system, named Feature Explorer, has been built on Windows NT to verify the proposed approaches. As demonstrated by the results, the feature reasoning kernel can be practically implemented and integrated with modern geometric modeling kernel. Therefore, it is believed that the proposed hybrid feature recognition method can facilitate the development of a robust feature recognition system that supports multiple applications in concurrent engineering.
Dedicated to my parents, wife and sisters
ACKNOWLEDGMENTS

I would like to express my sincere appreciation to my adviser Prof. Chia-Hsiang Menq for his patient guidance and instructive suggestions throughout my Ph.D. study. Without his continuous support and encouragement, this work would not be possible. Thanks also go to the other members of my Ph.D. advisory committee, Prof. Gary L. Kinzel, Prof. Henry R. Busby and Prof. Blaine W. Lilly, for their suggestions and comments. My appreciation and thanks are extended to my colleagues in Dr. Menq's research group for their friendships.

I want to share this joyful achievement with my parents and three sisters in China. Without their constant moral, spiritual and financial support, I will not able to get to this point. A special thanks to my sister-in-law, Ling Guo for her care and support since my journey to the States for higher education and training. And finally, my deep gratitude goes to my beloved wife, Qi Guo for her love, support, and understanding during my long period of studying. No matter what the geographic location (Maryland, Virginia, Ohio, New York and Massachusetts), her love makes all things possible.
VITA

November 7, 1971 ...................................... Born — Wuhan, Hubei, China

1993 ................................................... B.S. Mechanical Engineering
Huazhong University of Science & Technology
Wuhan, Hubei, China

1996 .................................................... M.S. Mechanical Engineering
Huazhong University of Science & Technology
Wuhan, Hubei, China

1996 – 1997 ........................................... Graduate Administrative Associate
Department of Mechanical Engineering
The Ohio State University, Columbus

1997-present ........................................... Graduate Research Associate
Department of Mechanical Engineering
The Ohio State University, Columbus

PUBLICATIONS

Digitization and Agile Metrology of High Precision Engineered Mechanical
Grantees Conference, CD-ROM, Long Beach, CA, Jan 5-8, 1999

Developing Environment”, Journal of China Mechanical Engineering, Vol. 8, No. 3,
pp81-84, 1997


**FIELDS OF STUDY**

**Major Field:** Mechanical Engineering  
**Specialization:** CAD/CAM
TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dedication .................................................................</td>
</tr>
<tr>
<td>Acknowledgments ......................................................</td>
</tr>
<tr>
<td>Vita .................................................................</td>
</tr>
<tr>
<td>List of Tables .........................................................</td>
</tr>
<tr>
<td>List of Figures .......................................................</td>
</tr>
</tbody>
</table>

Chapters:

1 Introduction ........................................................................................................ 1
   1.1 Background and Motivation ........................................................................ 1
   1.2 Research Scope and Problem Statement .................................................. 3
       1.2.1 Research Objectives .......................................................................... 3
       1.2.2 Research Scope .................................................................................. 5
   1.3 Organization of the Dissertation ................................................................... 5

2 Survey of Related Research Work ...................................................................... 7
   2.1 Features and Feature Creation Methods .................................................... 7
       2.1.1 Surface Feature and Volumetric Feature ........................................... 8
       2.1.2 Explicit Feature and Implicit Feature ................................................ 11
       2.1.3 Feature Taxonomy ............................................................................. 12
   2.2 Automatic Feature Recognition ................................................................... 16
       2.2.1 Boundary-based Feature Recognition ................................................ 18
       2.2.2 Volume-based Feature Recognition ..................................................... 29
       2.2.3 Hybrid Feature Recognition ................................................................ 37
   2.3 Research Issues for Automatic Feature Recognition .................................... 40
       2.3.1 Application Independent Feature ......................................................... 40
       2.3.2 Feature Interaction ............................................................................ 41
       2.3.3 Multiple Interpretations ..................................................................... 42
5.4.1 Methodology ............................................................................................ 104
5.4.2 Seed Entity ............................................................................................... 106
5.4.3 Characterization and Identification of Feature Loops ............................... 107
5.4.4 Recognition of Basic Surface Features ..................................................... 108
5.5 Example ............................................................................................................ 108
5.6 Summary ........................................................................................................... 112

5 Feature Recognition Phase II: Volumetric Feature Extraction ......................... 113
6.1 Introduction ....................................................................................................... 113
6.2 Mathematical & Theoretical Foundation ........................................................... 114
   6.2.1 Manifold Topology ............................................................................. 115
   6.2.2 Part Volume, Stock Volume and Delta Volume ........................................ 116
   6.2.3 Positive and Negative Volumetric Features.............................................. 124
   6.2.4 Machining Features and Design Features ................................................. 125
   6.2.5 Properties of Volumetric Features ........................................................ 127
6.3 Volumetric Feature Extraction ........................................................................... 131
   6.3.1 Negative Volumetric Feature Extraction ................................................... 131
      6.3.1.1 Feature Accessibility........................................................................ 132
      6.3.1.2 Spatial Relationship ......................................................................... 134
      6.3.1.3 Halfspace Partition........................................................................... 135
      6.3.1.4 Processing of Cylindrical End Faces of Slots................................... 143
      6.3.1.5 Algorithm for Negative Volumetric Feature Extraction ................... 146
   6.3.2 Positive Volumetric Feature Extraction .................................................... 146
   6.3.3 Updating of Type III Surface Feature ....................................................... 149
6.4 Examples ........................................................................................................... 150
6.5 Summary ........................................................................................................... 151

7 Post-recognition Processing .................................................................................. 152
7.1 Introduction....................................................................................................... 152
7.2 Feature Selection for Various Applications....................................................... 153
   7.2.1 Design Intent Recovery ........................................................................ 153
   7.2.2 Machining .............................................................................................. 156
7.3 Multiple Interpretations of Machining Features ................................................. 156
7.4 Feature Semantics Classification ....................................................................... 160
7.5 Parameterization of Feature Shapes ................................................................... 164
7.6 Other Applications of AFR ............................................................................. 169
7.7 Summary ........................................................................................................... 170

8 Implementation: A Proof-of-Concept Prototype System Feature Explorer ............. 171
8.1 System Architecture .......................................................................................... 171
8.2 Software Foundations Utilized ......................................................................... 173
LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1</td>
<td>Curvature distribution of the six features shown in Figure 5.2</td>
<td>94</td>
</tr>
<tr>
<td>5.2</td>
<td>Classification of faces in bi-attribute</td>
<td>96</td>
</tr>
<tr>
<td>5.3</td>
<td>Computation of the second attribute of sharp edges</td>
<td>100</td>
</tr>
<tr>
<td>5.4</td>
<td>Classification of smooth edges in bi-attribute</td>
<td>101</td>
</tr>
<tr>
<td>5.5</td>
<td>Classification of vertices in bi-attribute</td>
<td>103</td>
</tr>
<tr>
<td>5.6</td>
<td>Classification of seed entities for entity propagation</td>
<td>106</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Surface and volumetric representation of a pocket feature</td>
<td>9</td>
</tr>
<tr>
<td>2.2</td>
<td>Four external access directions (EADs) of a step feature</td>
<td>13</td>
</tr>
<tr>
<td>2.3</td>
<td>Classification of form features in STEP Part 48</td>
<td>15</td>
</tr>
<tr>
<td>2.4</td>
<td>The hierarchical structure of the proposed feature taxonomy</td>
<td>17</td>
</tr>
<tr>
<td>2.5</td>
<td>Graph pattern matching of a slot feature</td>
<td>20</td>
</tr>
<tr>
<td>2.6</td>
<td>An unidentified step feature by AAG due to feature interaction</td>
<td>21</td>
</tr>
<tr>
<td>2.7</td>
<td>Ambiguity of graph pattern matching</td>
<td>21</td>
</tr>
<tr>
<td>2.8</td>
<td>An unmanufacturable slot</td>
<td>23</td>
</tr>
<tr>
<td>2.9</td>
<td>ASV decomposition of an example part P</td>
<td>31</td>
</tr>
<tr>
<td>2.10</td>
<td>ASVP decomposition of a simple part</td>
<td>33</td>
</tr>
<tr>
<td>2.11</td>
<td>Intersection of a plane and a cylindrical face</td>
<td>35</td>
</tr>
<tr>
<td>2.12</td>
<td>Unrecognizable feature in hint-based method</td>
<td>39</td>
</tr>
<tr>
<td>2.13</td>
<td>Different feature volumes for a hole feature considering multiple interpretation</td>
<td>43</td>
</tr>
<tr>
<td>2.14</td>
<td>Geometric reasoning kernel vs. feature reasoning kernel</td>
<td>46</td>
</tr>
<tr>
<td>3.1</td>
<td>The feature reasoning kernel architecture of the proposed hybrid feature</td>
<td>49</td>
</tr>
<tr>
<td>4.1</td>
<td>Some variations of fillets and rounds of a notch feature and its canonical</td>
<td>54</td>
</tr>
<tr>
<td></td>
<td>boundary after fillet/round suppressing</td>
<td></td>
</tr>
<tr>
<td>4.2</td>
<td>The framework of the B-Rep model simplification by fillet/round suppressing</td>
<td>57</td>
</tr>
</tbody>
</table>
4.3 Illustration of the formation and suppressing of round and fillet ......................... 58
4.4 Various trace faces formed by the blending on edges and vertices ....................... 60
4.5 Illustration of the spanning angle of a blending cylindrical face in section view 61
4.6 An example part with two slots having cylindrical faces .................................. 62
4.7 Closed and open trace face chains and their homeomorphic equivalence in $E^2$ .. 63
4.8 A ring-type chain in a cylinder and the H.E. of its enclosure faces ....................... 64
4.9 A disc-type chain in a block and the H.E. of its enclosure faces .......................... 65
4.10 The H.E. of the enclosure faces of the example part in Figure 4.7 after chain removing ........................................................................................................................ 66
4.11 Illustration of some terms for the proposed incremental knitting process ............. 67
4.12 The detailed flow chart of the incremental knitting process ............................... 71
4.13 Deduction of multiple rings into a single ring for a ring-type chain .................... 72
4.14 The knitting process for the example part shown in Figure 4.10 ....................... 73
4.15 Five example parts (Note: gray-filled faces are enclosure faces of chains) ...... 76
4.16 Illustration of a KR operation on the example part shown in Figure 4.15a .......... 77
4.17 Branching→Regular conversion of $R$ and role switch between $F_e$ and $F_i$ for the example part shown in Figure 4.15b ................................................................. 78
4.18 Illustration of a KR operation on the example part shown in Figure 4.15c ....... 78
4.19 Illustration of a KR operation on the example part shown in Figure 4.15d ....... 79
4.20 Illustration of edge-face adjacency ...................................................................... 82
4.21 Illustration of a KD operation on the example part shown in Figure 4.15e .......... 83
4.22 Test part 1: NEW_DEMO_US ................................................................. 85
4.23 Test part 2: the example part shown in Figure 4.4 ........................................... 86
5.1 Cylindrical protrusion & depression on a block ................................................. 89
5.2 Three groups of two different features with same total curvature ................... 93
<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.2</td>
<td>Preliminary outer shape in positive design features of CAM-I ANC101</td>
<td>154</td>
</tr>
<tr>
<td>7.3</td>
<td>Design feature tree of CAM-I ANC101</td>
<td>155</td>
</tr>
<tr>
<td>7.4</td>
<td>The complete hierarchy of the design features of CAM-I ANC101</td>
<td>157</td>
</tr>
<tr>
<td>7.5</td>
<td>A possible machining sequence for the part NEW_DEMO_US</td>
<td>159</td>
</tr>
<tr>
<td>7.6</td>
<td>Configurations of the existent faces of a slot feature</td>
<td>161</td>
</tr>
<tr>
<td>7.7</td>
<td>Flow chart of semantics classification based on decision tree approach</td>
<td>163</td>
</tr>
<tr>
<td>7.8</td>
<td>Modifier features suppressed by pre-processing in CAM-I ANC101</td>
<td>165</td>
</tr>
<tr>
<td>7.9</td>
<td>Some extracted rotational generics in CAM-I ANC101</td>
<td>166</td>
</tr>
<tr>
<td>7.10</td>
<td>Some extracted prismatic generics in CAM-I ANC101</td>
<td>166</td>
</tr>
<tr>
<td>7.11</td>
<td>Parameterization of prismatic generics and rotational generics</td>
<td>167</td>
</tr>
<tr>
<td>7.12</td>
<td>Irregular feature shape of a blind slot and its canonical volume</td>
<td>168</td>
</tr>
<tr>
<td>7.13</td>
<td>Part localization for inspection planning</td>
<td>169</td>
</tr>
<tr>
<td>8.1</td>
<td>System architecture of Feature Explorer- the proof-of-concept prototype system</td>
<td>172</td>
</tr>
</tbody>
</table>
CHAPTER 1

INTRODUCTION

1.1 Background and Motivation

Design and manufacturing-preparation activities can be regarded as information processing activities that, starting from the original rough design idea of the product and with the use of product models, generate detailed manufacturing control information and the framework which describes all the necessary information about the product and its processing. Increasingly high competition is demanding the manufacturing industry to bring competitively priced, well-designed and well-manufactured products to market in a timely fashion. Manufacturing systems have become more product-oriented, aiming at decreased lead times, minimal work-in-process, the just-in-time flow of material, and high efficiency and the flexibility of manufacturing capacity utilization. To achieve this end, increasing research attention has been given to integrating Computer-Aided Design (CAD) and Computer-Aided Manufacturing (CAM). To fully implement CAD/CAM integration, product design and production engineering functions must become closely integrated with manufacturing, and all bottlenecks in the flow of production must be eliminated. The seamless integration between design, planning, and manufacturing functions requires that sufficiently complete and accurate information of all aspects of products, production processes, and operations is available. Hence future design and planning systems will be closely aligned with manufacturing technology, and future-manufacturing systems will need more complete and accurate product information than
what is currently available. Ideally this integration should lead to concurrent life-cycle engineering, where all design, manufacturing, maintenance, and eventual disposal and reuse issues of the product can be considered simultaneously during its design.

Although traditional solid modeling can provide some necessary product design information, it has been proved deficient for modern product design as well as for many essential manufacturing applications such as process planning, group technology classification, coordinate measuring machine (CMM) inspection planning, and assembly planning. Several reasons account for the inefficiency of geometry-based CAD system:

- The geometry construction methods provided are primitive for real-world engineering designs;
- Design modification is time-consuming because the resulting low-level model does not preserve the design intents of the designer;
- The geometry-based models generated by traditional CAD systems do not contain all the information needed for downstream applications, such as process planning, assembly planning, and inspection planning.

As a result, feature, a high-level semantic entity which is not available in a geometric model, has been proposed to bridge the gap between engineering design and manufacturing. Features can enhance the product model by enabling clustering of geometric and topological entities and their attributes, which can be associated with engineering knowledge used in various applications. The outcome from the intensive research on feature-based CAD/CAM has already demonstrated the clear potential of features in creating attractive design environments and in facilitating geometric reasoning related to design function, performance evaluation, manufacturing process planning, NC programming, and other engineering tasks [Shah and Mantyla, 1995; Mantyla, Nau and Shah, 1996]. Therefore, features are widely regarded as a key enabling technology for the implementation of concurrent engineering and CAD/CAM integration.
1.2 Research Scope and Problem Statement

It is obvious that the geometric and topological entities such as faces, edges and vertices of a solid model only provide a low-level definition of the characteristics of a part, which are localized and unconnected to the part's overall function and behavior. The concept of feature, generally defined as an information cluster of low-level geometric and topological elements with a functional meaning in a given application context, allows an association between shape and functionality. Clearly, a major advantage of features is that they can augment geometric models with engineering meanings to make them useful for design and to integrate design with downstream manufacturing applications. Because of the higher semantic level of features, they can provide a basis for recording a more complete product definition as required for various applications in concurrent life-cycle engineering.

1.2.1 Research Objectives

In order to achieve a higher level of automation and integration for design and manufacturing processes, it is essential to analyze features, which can integrate the shape and functional information of products, as transferred and manipulated among related design and manufacturing activities. It is also important to prepare the framework to capture and manage this semantic information systematically. The representation and manipulation of features are fundamental but challenging problems for both information processing and manufacturing engineers.

In current feature research two major approaches for creating feature-based representations exist: feature-based design (FBD) and automatic feature recognition (AFR). Feature-based design allows designers to create a feature-based product model from scratch by using a set of predefined features and it has matured to the point that results have been incorporated into commercial CAD systems. For legacy CAD data without feature information, automatic feature recognition is needed to build a feature-based model. Although automatic feature recognition has been studied for almost two decades and some results have come out, some open technical issues have not yet been
resolved satisfactorily. One issue is that the domain of the features considered in current AFRs is very limited. AFRs need to be improved to scale up to more realistic mechanical parts with various complex shapes. The other bottleneck of current AFRs is their inability to deal with complex feature interactions, thus their performance is not reliable when processing real-world mechanical parts. Most of current AFRs focus on machining applications, thus only machining features are generated. When considering the information needed for various applications in concurrent engineering, the output from AFRs needs be abundant enough to provide multiple feature interpretations for different applications and processes. Since AFR plays an important role in recovering features from pure geometric legacy CAD models, it has been challenging researchers to find a more general and robust method.

The goal of this research work is to develop an information framework leading to a robust automatic feature recognition system that can scale up to more realistic mechanical parts with various complex shapes and can provide the information supporting complex reasoning incurred in concurrent engineering applications. In order to accomplish this goal, two specific objectives of the proposed research are identified. The first one is to investigate the underlying invariant properties of boundary entities of CAD models and characterize as well as group them into surface features. Although surface feature recognition can deal with various shapes by general curvature analysis, it can not handle feature interactions very well due to its boundary-based characteristic. Thus feature volume extraction is introduced to facilitate the achievement of robust feature interaction resolving and the generation of multiple feature interpretations. The second objective is to establish an algorithmic foundation so as to construct and manipulate the volumetric features based on the results from the first objective. The resulting volumetric features can yield parameterized feature shapes and lead to a feature-based model. Based on the approaches developed for the two objectives, a hybrid feature recognition method is proposed to support geometric reasoning needed in various downstream manufacturing applications and to facilitate concurrent design optimization.
1.2.2 Research Scope

The proposed hybrid feature recognition method leads to a feature reasoning kernel which consists of four functional modules: pre-recognition processor, surface feature recognizer, volumetric feature recognizer and post-recognition processor. The pre-recognition processor is used to simplify the input B-Rep model by automatic fillet/round suppressing for efficient feature recognition. The surface feature recognizer is designed to demonstrate the implementation of the proposed methods that transform the discrete boundary entities of a CAD model into meaningful groupings, called surface features. The volumetric feature recognizer extracts maximal feature volumes based on surface features and resolves feature interactions incurred in real-world engineering designs. The post-recognition processor manipulates the maximal feature volumes according to the desired applications and process requirements and generates multiple interpretations of the parameterized feature shapes. An explicit feature representation can be built based on the integration of surface features and volumetric features. The more abstract implicit feature representation such as feature parameters is extracted from the explicit feature model to achieve a complete feature-based model. Based on the proposed hybrid feature recognition method, a feature recognition prototype system, named Feature Explorer, has been built on Windows NT to verify the proposed approaches. As demonstrated by the results, the feature reasoning kernel can be practically implemented and integrated with modern geometric modeling kernel. Therefore, it is believed that the proposed hybrid feature recognition method can facilitate the development of a robust feature recognition system that supports multiple applications in concurrent engineering.

1.3 Organization of the Dissertation

This dissertation comprises 9 chapters. The contents of each are summarized as follows: Chapter 2 presents a summary of previous research work in the areas related to this dissertation with emphasizes on feature categorization, taxonomy and automatic feature recognition. The research issues in automatic feature recognition are summarized as well. In addition, architectural issues are discussed by introducing the concept of feature reasoning kernel. Chapter 3 introduces the framework of a feature reasoning kernel that
can encapsulate all the information processing modules and data structures of the proposed hybrid feature recognition method. Chapter 4 presents a methodology of B-Rep model simplification by automatic fillet/round suppressing to facilitate automatic feature recognition. Chapter 5 develops a bi-attribute based method to recognize basic surface features from the simplified B-Rep model. Chapter 6 presents an algorithmic foundation to extract feature volumes based on basic surface features. Chapter 7 discusses the post-recognition process - in particular, the multiple feature interpretations, feature semantics classification and feature parameterization. Chapter 8 presents the implementation of a prototype feature recognition system, which is named Feature Explorer and built on top of the proposed feature reasoning kernel. Two typical parts are tested to show the capabilities of the developed prototype system. Finally, Chapter 9 gives a summary of this research and offers recommendations for future work.
CHAPTER 2

SURVEY OF RELATED RESEARCH WORK

This chapter presents a summary of previous research work in the areas related to this dissertation. It briefly discusses three feature creation methods, two major categorizations of feature representation and feature taxonomy. Since automatic feature recognition is the focus of this research work, the related work is discussed in details and some of the open technical issues in this field are summarized. Lastly based on the study of the architecture of geometric modeling kernel, the concept of feature reasoning kernel is presented to encapsulate the data structures and all the functional modules of automatic feature recognition.

2.1 Features and Feature Creation Methods

Features have been proposed as a means of providing high-level semantic information for interfacing CAD to manufacturing applications in product life-cycle [Pratt, 1993; Mantyla et al., 1996]. Features are generally classified into the following categories: form features, precision features, material features, assembly features, and technological features. Among these feature categories, form features have been studied much more intensively than the other features [Shah, 1991]. There have been three major approaches to create features:

• Feature-based design: This approach normally consists of several modules including feature modeling, feature library, validity check [Chang, Anderson and Mitchell, 1988;
Shah et al., 1990]. Parts are modeled by using pre-defined features from the feature library. The drawback is that the feature model is not interchangeable among different applications.

- Interactive or human-assisted feature recognition: In this approach, a geometric model of a part is created first. Then, features are defined by the users through picking topological entities, associated with each feature, from a CAD model. The limitation of this approach is that it is time-consuming and the recognized features could be user dependent.

- Automatic feature recognition (AFR): In this approach, form features of interest to an application are automatically extracted from the geometric model of a part. Thus designers are no longer restricted to the limited modeling method within predefined feature elements. Automatic feature recognition has fundamental significance from the integration perspective of product development cycle, and it will be reviewed and discussed in detail in section 2.2.

2.1.1 Surface Feature and Volumetric Feature

Features are representations of shape aspects of a physical product that are mappable to generic shapes and are functionally significant. From the geometric point of view, feature representation can be surface-based or volume-based. Surface-based features, *surface features* in short, are collections or groupings of faces with functional meanings on the part boundary. Volume-based features, *volumetric features* in short, are three-dimensional point sets of a part or its complement, i.e., closed solids. Both include positive features and negative features that represent protrusions and depressions. Figure 2.1 illustrates the surface and volumetric representations of a pocket feature.

The major difference between the two representations is that the volumetric representation requires additional faces to define a closed volume from the feature faces that lie on the part boundary. Different methods have been specified to define feature volume [Dong, 1991]. Typically, a selected set of closing faces is added to a surface feature to produce a shell that unambiguously defines a volume. This is often referred as
the entity growing problem [Shah, 1991]. However, there is no unique mapping from surface feature to volumetric feature. In order to solve the problem, we proposed to create maximal volumetric features based on surface features to achieve one-to-one mapping, which will be discussed in Chapter 6.

![Figure 2.1: Surface and volumetric representation of a pocket feature](image)

The disadvantage to the use of volumetric features rather than surface features is increased complexity due to additional topological entities. The volumetric approach typically leads to representing the geometry of a part as a Boolean combination of feature volumes of the features comprising the part. This poses some additional complexities, for example, in defining feature volumes. It is necessary to check which entities lie on the exterior of the part and which do not. However, many applications such as process planning heavily rely on volumetric features because feature volumes correspond to the
volumes removed by machining operations and, thus for instance, the verification of machining operations is more straightforward using volumetric features rather than surface features.

Although the surface-based approach avoids the difficulties of defining feature volumes, there exist some other complexities. The regeneration of the part geometry after some changes in feature attributes, for example, is more difficult because it requires local geometry modifications in terms of boundary elements, and consequently feature interactions obviously cause additional challenges and, hence, require more validity conditions.

[Pratt, 1988] compared volumetric and surface representations for features. He concluded that a volumetric representation is superior since the interaction between features is easier to deal with. In addition, a volumetric representation offers a more user-friendly specification method and reduces the possibility of object invalidity. A volumetric feature representation in a B-Rep context gives a unique representation of a feature and enables the specification of attributes to faces. A volumetric feature representation in a CSG context, however, does not provide a unique representation and does not offer a facility for attaching properties to different faces of a primitive solid.

In feature recognition, not only do we need to get the feature semantics, but also obtain the feature parameters. Surface features consist of feature faces lying on the part boundary. Thus they provide a good basis to perform feature semantics classification. However, they can only provide a partial boundary of the feature, which is not enough for feature parameterization. Volumetric features provide the closed boundary. Thus they can be directly used to extract feature parameters. Therefore, for feature recognition both surface features and volumetric features are indispensable to achieve a complete feature representation including feature semantics and parameters.
2.1.2 Explicit Feature and Implicit Feature

From the viewpoint of information abstraction, geometric representation of a feature can be explicit or implicit [Wilson and Pratt, 1988]. The explicit representation fully defines the geometric details of a feature by employing boundary entities or volumes, which correspond to surface feature and volumetric feature respectively. The implicit representation employs a small number of parameters to model the shape aspect of interest, such as profile, center-axis and path. Therefore it is more compact than the explicit representation. These high-level data supplies sufficient information to define the feature but the full geometric details have to be evaluated when required. Thus explicit representation is an evaluated one while implicit representation is an unevaluated one. For example, a blind hole might be explicitly represented in a B-Rep by listing the faces that represent the hole; or it might be implicitly represented by diameter, center-axis, entry point and height.

Several reasons account for the importance of implicit feature representation. The major one comes from the feature-based design. Considering the friendly and easy-to-use aspects, feature-based design must provide a very convenient way to construct a part model. Therefore, the desired method is to provide the user several parameters such as the width, height and radius plus positioning and orientation information to create feature instances from templates. The users don't need to take care of the complex geometric computation to get the explicit details such as the entities forming the final surface boundary or volume of the features, which is left to the computer program. Another reason is that the parameters of the features are very important information for tolerancing and dimensioning, a key procedure towards so-called product modeling [Gu and Chan, 1995]. It is advantageous that a large class of rotational and prismatic features can be represented in terms of the volume by sweeping a 2D profile along a 3D path. Both the profile and its path can be represented in terms of a comparatively small number of parameters.
Feature recognition requires the identification and grouping of the explicit topological entities from a solid model. Thus explicit representation is used to organize the groupings of low-level topological entities. In addition to recognized explicit representation, feature parameter is an indispensable part of feature definition required by various applications. Therefore, feature recognition is also involved with deriving an implicit feature representation from the recognized explicit description. As mentioned above, both surface features and volumetric features are indispensable to achieve a complete feature representation including feature semantics and parameters. Thus explicit feature representation should be the integration of surface feature and volumetric feature. Based on this requirement, a feature recognition system should not only be able to construct explicit representation by recognition in the forms of surface features and volumetric features, but also be capable of deriving implicit representation of features.

2.1.3 Feature Taxonomy

Both automatic feature recognition and feature-based design need a precise and systematic definition of the features that allow each instance of a feature to be identified. The feature definition should contain the minimal set of necessary conditions that classify a feature uniquely [Joshi and Chang, 1988]. To achieve this aim, the unique feature definition should be based on a general classification scheme for features that facilitates their automatic recognition, and an inference procedure capable of recognition in a complete and consistent manner [Hirschtick and Gossard, 1986]. Thus in feature-based modeling, feature taxonomy is introduced to provide the needed classification scheme that specifies the feature domains for feature-based design and automatic feature recognition. A major advantage of feature taxonomy is the hierarchical way in which it can be used to classify features systematically. Thus it can reduce execution time for feature semantics classification in feature recognition. Another advantage arises from the notion of inheritance. Since the instance of a particular subclass is also an instance of its super-class in a taxonomy, the properties of the super-class can be inherited by the subclass without explicitly being repeated. Thus a more compact representation of knowledge is achieved [Joshi and Chang, 1988].
Several efforts have been made to propose feature taxonomies. But none of them is universally accepted or widely used at the present time. In the following, several design-oriented and machining-oriented feature taxonomies that have been reported in literature are examined.

[Figure 2.2: Four external access directions (EADs) of a step feature]

[Gindy, 1989] presented a model for describing form feature geometry. The model treats form features as volumes enveloped by entry/exit and depth boundaries. The geometric characteristics of a feature are decided by the degree of accessibility to its volume, its boundary type, its exit boundary status and its form variation with respect to its depth axis. He also presented a hierarchical structure for form feature classification, in which form features are divided into three categories: protrusions, depressions, and surfaces. Feature geometry is described by deciding on its number (1–5) of external access directions (EADs) (see Figure 2.2), its boundary type (open, closed) and its exit boundary status (through, not through). The result of grouping features according to these characteristics is a list of form features classes or primary features, such as boss, pocket, hole, slot, notch, and real and imaginary surfaces. A feature subclass is decided by attaching perimeter geometry to the feature class. It is at this stage that the feature becomes a fully recognizable entity such as a square pocket, round hole and rectangular slot. As the author pointed out, an object-oriented approach to the implementation of the
hierarchical structure is advantageous. Since features are instances of a subclass, a class and a category, the characteristics of encapsulation and inheritance can be utilized to facilitate the coding process.

[Wilson and Pratt, 1988] presented a taxonomy of features based on the overall shape of features and the assumption that features will be incorporated in solid modeling systems. Features are defined as regions of interest in a part model and are classified into four main classes: through holes, protrusions, depressions and areas. In terms of their cross-sectional shapes, the first three classes are further classified as rotational type and prismatic type. [Hummel and Brooks, 1986] applied a similar taxonomy to their expert process planning system (XCUT), using an object-oriented programming language for the implementation.

[Butterfield et al., 1985] developed a feature classification scheme for CAM-I (Computer Aided Manufacturing International). In this scheme form features are classified into three main categories: rotational features, non-rotational (prismatic) features and sheet features. Sheet features are further classified as flat or formed. Non-rotational features are classified as depressions, protrusions and surfaces. Further classification is performed for depressions and protrusions based on the properties such as symmetric/non-symmetric, internal/external and through/non-through. Rotational features are classified as concentric and non-concentric. Since this scheme is intended to be the standard for all the application programs that carried out in the CAM-I project, it is broad and general.

International Standard Organization (ISO) also put efforts in developing feature data exchange standard. STEP (STandard for Exchange of Product data) Part 48 [Dunn, 1988; Dunn, 1992] provides a general-purpose form feature data model in a relatively abstract level. Part 48 defines a form feature to be a shape aspect that conforms to some preconceived pattern or stereotype. A shape aspect is defined as a shape or a distinguishable portion of a shape. Part 48 schema supports three types of features: volume features, transition features and feature patterns, each of which has several
subtypes, as shown in Figure 2.3. A volume feature is an increment or decrement to the volume of a shape, such as a hole or a boss. Attributes of a volume feature are an associated volume and cross-section type (optional). A transition feature separates or blends surfaces, such as fillets or chamfers. A feature pattern is a set of similar features in some regular geometric arrangement. [Ovtcharova et al., 1992] proposed a form feature classification based on STEP Part 48. Same as STEP Part 48, no further detail classification is performed for the abstract features such as protrusions and depressions.

Figure 2.3: Classification of form features in STEP Part 48

As we can see from the above discussions, current feature classifications are either too abstract to fully define the geometric and topological characteristics of features, or too rigid to handle the variations of features. The feature research community has not reached agreement on a canonical set of features that can be used to evaluate the available algorithms in feature recognition. A well-defined feature classification is essential for efficient identification of features and should be able to cover a large domain of features with each one fully defined. Based on CAM-I feature classification [Butterfield et al., 1985] and the concept of external access direction (EAD) [Gindy, 1989], we propose a feature classification scheme (see Figure 2.4) considering the requirements for automatic
recognition of a large domain of features from B-Rep models of real-world mechanical parts. Please note that feature pattern is not considered in this classification scheme. In the proposed feature taxonomy, features are classified into primary features and secondary features based on their shape appearance. Primary features form the major shape of a CAD model and are further classified into positive features and negative features based on whether the major shape is a protrusion or a depression. Secondary features are small detailed features used to modify the primary features such as fillets, rounds and chamfers. From the concept of EAD, EAD is only valid for negative features and can be 1, 2, 3, 4, or 5 as illustrated in Figure 2.2. For positive features EAD is meaningless so that it is designated as zero. Positive features are classified into cylindrical boss, symmetric boss and non-symmetric boss based on circular/non-circular cross section and symmetric/non-symmetric axis. As shown in Figure 2.4, negative features are classified into more detailed feature types based on EAD, through/non-through, circular/non-circular, symmetric/non-symmetric conditions. When EAD is 5, the feature is defined as face feature, which is used to describe portions of negative features. Appendix A illustrates all the final feature types in the proposed feature taxonomy. This proposed feature taxonomy forms the feature domain of this research work.

2.2 Automatic Feature Recognition

The automatic feature recognition work was initiated by Kyprianou at Cambridge University in 1980 [Kyprianou, 1980]. Since then automatic feature recognition was performed by discovering topological and geometric patterns in boundary CAD models and comparing them to the characterized features needed to be identified. The key idea was topological entity classification based on geometry. Almost all the subsequent methods for automatic feature recognition have used this idea in some form. At the same time, [Woo, 1982] developed a convex hull decomposition algorithm to find volumes to be removed by machining. Since then, a significant amount of research has been conducted on automatic feature recognition, which can be classified into three major categories: boundary-based approach, volume-based approach and hybrid approach.
Figure 2.4: The hierarchical structure of the proposed feature taxonomy
2.2.1 Boundary-based Feature Recognition

Various boundary-based feature recognition approaches have been proposed since the seminal work by Kyprianou. They can be categorized into these four major types: *syntactic pattern recognition* [Jakubowski, 1982; Staley et al., 1983; Choi et al., 1984; Jakubowski, 1985; Li, 1988; Li et al., 1989; Chuang and Henderson, 1990; Nnaji et al., 1991], *graph-based approach* [Kyprianou, 1980; Henderson and Anderson, 1984; DeFloriani, 1987; Joshi and Chang, 1988; DeFloriani, 1989; Falcidieno and Giannini, 1989; Marefat and Kashyap, 1990; Gavankar and Henderson, 1990; Corney and Clark, 1991; Trika and Kashyap, 1994; Ji and Marefat, 1995; Ji, Marefat and Lever, 1995], *rule-based approach* [Henderson, 1984; Kung, 1984; Dong and Wozny, 1988] and *artificial neural network-based approach* [Prabhakar and Henderson, 1992]. Except graph-based approach, all the other three approaches have become extinct. Graph-based approach still attracts attention today and is reviewed as follows.

Graph-based approach was initiated by Joshi and Chang [Joshi and Chang, 1988] and has become one of the current most prevalent approaches in the field of feature recognition. The reason this approach is so popular is that it benefits from the theoretical foundation of graph theory and topology and utilizes many developed concepts from these theories to build various algorithms to deal with searching, traversing, parsing and matching. In this approach, the B-Rep of a part is translated into a graph representing its topology. Subgraphs are used to create the templates of primitive features. Usually, the graph representation consists of nodes and links corresponding to the geometric entities (faces, edges and vertices) of the part. Additional information may be incorporated into the graph to represent the properties of geometric entities such as concavity and face orientation. The graph representation is then searched, using subgraph isomorphism, for certain properties to identify the features embedded in the part that match the templates of the primitive features. The identified subgraphs are subsequently extracted as the features embedded in the part. Graph-based approach is performed very well for some simple
experimental parts with isolated features. When exposed to real mechanical parts in industry, it shows its inability to deal with the complicated feature interactions that exist commonly in the real-world mechanical parts.

Joshi and Chang [Joshi and Chang, 1988] used face and edge information from B-Rep models to devise the attributed adjacency graph (AAG) method for a part. An AAG is defined as a triple $G = (N, A, T)$, where $N$ is the set of nodes, $A$ is the set of arcs/links, and $T$ is a set of attribute values for arcs in $A$. The arc values (0 or 1) in $T$ represent the concavity of the corresponding geometric entities. Figure 2.5a and 2.5b show a part and its AAG. The nodes $F_1$–$F_{10}$ represent the faces and the arcs represent the edges. An arc is labeled 0 if the corresponding edge is concave and 1 if the edge is convex. A graph of a slot feature template is shown in Figure 2.5c. By graph pattern matching, a slot feature ($F_3, F_4, F_5$) can be identified from the part.

Instead of directly applying subgraph isomorphism to the AAG of the part, which is computationally intensive, a heuristic method is proposed to divide the part graph into components that could contain features. The heuristic is based on the following observation: a face that is adjacent to all its neighboring faces through a convex edge is not likely to belong to a depression feature. The removal of such convex nodes will separate the AAG into several connected components, and if there are no interactions between features, each of these connected components (a subgraph) will represent an isolated feature or a depression in the original part. The recognition method, which is based on subgraph isomorphism and graph-based heuristics, is then applied to each component to identify the kind of feature represented by the component. To handle interacting features, Joshi implemented a heuristic rule that splits arcs and nodes to form complete feature subgraphs. However, this heuristic splitting can handle only some interactions between features. The difficulty arises when there are interactions that can possibly destroy an adjacency between two faces. This in turn means that the pattern of
the feature within the AAG is lost, and hence unrecognizable. Therefore, more complex interactions in which the graph patterns of the features involved are modified are beyond the capability of this method, which requires more information than simple heuristics.

Figure 2.5: Graph pattern matching of a slot feature

The significance of this research is in being among the first to propose the graph-based approach for extracting machining features. The advantage of Joshi’s method is its use of both forward chaining and a heuristic method to reduce the computational effort. Figure 2.6 shows a simple part whose constituting primitive features are not correctly identified by the previous approach. The heuristic eliminates face $F_1$ and thus the step feature
formed by faces $F_1$ and $F_2$ will not be recognized. Another limitation of this approach is that the feature graph pattern may be ambiguous. As shown in Figure 2.7, the same graph pattern may describe two different features.

Figure 2.6: An unidentified step feature by AAG due to feature interaction

Figure 2.7: Ambiguity of graph pattern matching
Marefat and Kashyap [Marefat and Kashyap, 1990] extend Joshi's approach to more fully address mutually-modifying feature interactions. They observed that the arcs between (a feature's) face nodes in the part graph may be missing when features intersect, hence proposed to restore the missing arcs into the part graph. They collected all possible candidates for missing arcs, and ranked the candidates based on part geometry information using the Dempster-Shafer theory. The arcs with "notably different (higher)" ranks were restored. The problem with their method is that we cannot identify the exact set of missing arcs. When we add fewer arcs than necessary, there are unrecognized features. When we add extraneous arcs, we may introduce bogus features. Ji and Marefat [Ji and Marefat, 1995; Ji, Marefat and Lever, 1995] proposed a similar approach based on Bayesian networks, but could not overcome the problem. In addition, the graph pattern matching approach does not ensure the machinability of a recognized feature as long as a feature is defined as a collection of faces. As shown in Figure 2.8, even though the three faces in bold match the face template of a slot, the recognized feature is not manufacturable as a slot because it is not accessible from the side. The non-volumetric notion of feature may cause fatal problems in machining applications. Along the line of Marefat's work, [Trika and Kashyap, 1994] devised algorithms that can compute the exact set of missing arcs. However, their algorithms place very strong restrictions on input parts and feature intersections: the part must be polyhedral (only with planar faces) and iso-oriented (with no inclined faces). As a consequence, all recognized features are cuboidal, and therefore may have to be combined with one another in order to generate features meaningful for manufacturing. Their work is pure pattern matching with little consideration of manufacturing. However, they raised the important issue of completeness. The input for feature recognizers is typically a solid model for the desired part, plus a solid model of the stock, or raw material. The material to be removed by machining, called the delta volume, is computed by subtracting the part from the stock. Trika and Kashyap called a feature recognizer complete if, for every part, the delta volume is contained in the union of all feature volumes generated by the feature recognizer. If a feature recognizer is not complete, there may exist unrecognized regions of the delta volume and therefore the specified part may not be obtained even after all
feature removal operations are performed. Trika and Kashyap proved that their algorithm is complete. The completeness issue will also be discussed in Chapter 6 when the properties of volumetric features are summarized.

![Figure 2.8: An unmanufacturable slot](image)

DeFloriani [DeFloriani, 1987; DeFloriani, 1989] introduced a similar graph-based approach to identifying the topological features of an object. The graphs used in this technique are called edge-face graphs (EFG). In an edge-face graph, a cut node corresponds to a face in the object that splits the graph into two or more connected parts described as bi-connected components. Similarly, separation pairs correspond to pairs of faces on the object which, when removed, may split the object into connected pairs. Thus the feature extraction algorithm decomposes the EFG into bi-connected and tri-connected components. The components are subsequently organized into what is called an object decomposition graph (ODG). Based on arcs incident on a component in the ODG, an entity is classified as a DP-feature or an H-feature. DP-features are protrusions or
depressions on a set of faces of the object and H-features are through-holes, handles, or bridges. The method provides a global understanding of the object shape. Construction of the ODG provides not only a list of features but also their relationships in the global shape of the part. Unlike local extraction based on geometric information, this method can identify "compound" features formed by a combination of through-holes and protrusions or depressions. However, the identification does not provide a sufficiently detailed classification for manufacturing and engineering purposes. In order to be appropriate for automation and planning functions, additional mechanisms to identify manufacturing-related primitives, such as pockets, slots, and holes that comprise the DP- and H-features, would be necessary.

[Falcidieno and Giannini, 1989] developed a feature recognition system based on a face-adjacency hypergraph (FAH). In the FAH model nodes represent the faces of the part model. The arcs and hyper-arcs represent the edges and vertices of the part model, respectively. The system consists of three modules: (i) feature recognizer, which uses FAH of the part model and Kyprianou's [Kyprianou, 1980] approach; (ii) feature extractor, which separates the form features from the part model; and (iii) feature organizer, which arranges the extracted features into a hierarchical feature graph, called Structured Face Adjacency Graph. Based on this work, [Ansaldi and Falcidieno, 1990] used a depth-first search on a FAH representation of the part model to determine its connectivity properties. Cut nodes (tri-connected and bi-connected) are used as a basis to recognize protrusion and depression features originating from one face. Cut nodes are also used for recognizing features originating from holes and handles which define inner loops on more than one face of the part model. Features that are not identified by applying the cut-node concept are recognized by using the property of concavity/convexity of edges belonging to the loops. The system cannot identify features such as bevel and chamfer because these features coincide with the bounding loop of the face.
[Gavankar and Henderson, 1990] proposed a graph-based technique capable of isolating protrusion and depression features by considering only the topological information of the part model. They used an important property that protrusion and depression features constitute bi-connected components in the face adjacency graph (FAG) of the boundary model of the part. This feature recognition methodology decomposes a part model into its bi-connected components by finding the cut nodes in the FAG. A cut node is a node in the graph such that its removal results in an increased number of connected components in the graph. The authors also proposed a heuristic for limiting the search space to identify the cut nodes. The proposed system is capable of extracting all the features that have a unique entrance face (used in identifying the cut nodes). Features such as blind holes and pockets that open up into more than one face cannot be recognized since their cut node cannot be identified. Further, through holes cannot be identified since they cannot be separated into bi-connected components from the FAG of the part model. The limitation of the cut-vertex method is that there is no proof that a cut vertex node always represents an entrance face that links a feature to the body of the part. This aspect is further complicated by situations in which a feature may have more than one entrance face in the part.

[Sakurai and Gossard, 1990] proposed a different approach to graph-based feature definition and recognition. A feature can be a cavity, a protrusion, or neither. In this technique, a feature is represented by a feature graph. A feature graph is a B-Rep of the feature’s faces augmented with user fact nodes and parameter nodes. Both user fact nodes and parameter nodes can refer to the same entity in the B-Rep graph. Although parameter nodes specify an attribute of a feature such as width, length, and the like, user fact nodes may supply additional geometric information such as parallelism, coaxiality, and perpendicularity among faces. A template feature is the generic definition of a feature, and a recognized feature is an instance of a template feature. The template feature is defined by interactively selecting a set of faces in a graphical display of the solid model and coupling them with user facts and parameters. Feature recognition is accomplished through graph matching by searching the entire solid model to find instances for each
template feature. The graph matching is implemented by comparing each face of the template feature with every face of the solid model to determine whether the two faces have the same geometric type, a matching number of loops, and a matching number of edges. To recognize interacting features, the algorithm first removes the recognized features from the original object. When the recognized feature is a cavity, the feature removal generates a volume that fills the cavity and it is added to the solid model. On the other hand, when the recognized feature is a protrusion, the procedure generates the volume of the protrusion and subtracts it from the solid model. The purpose of such removal is to simplify the shape of the solid model so that other yet unrecognized features can be identified. The advantage of this method is that each feature can be defined easily and interactively without requiring a sophisticated language. However, there are interacting situations in which the features are not correctly identified, as when intersecting features have coincident faces when features have volumes that split each other completely (i.e., the volumetric difference between the two features is two strictly disjoint regions of space). A general problem with such face-based pattern recognition methods is that it is difficult to create a feature volume from a set of faces that do not enclose a volume. The removal of a volumetric feature also tends to complicate further recognition in cases of arbitrary interactions between features because it tends to generate extraneous elements or alterations that may complicate recognition of the involved features.

[Corney and Clark, 1991] developed a graph-based algorithm for identifying holes and pockets that emanate from multiple faces in a 2.5-D part model. Extraction of pockets and holes emerging from a single face is simple because the edges of these features form an inner loop comprising convex edges. The input format required for extracting holes and pockets is an attribute face-edge graph of a part model that is similar to the graph devised by [Joshi, 1988]. The attribute '1' and '0' on the edges connecting the faces in the face-edge graph indicate whether the edge is convex or concave, respectively. To start with, an aspect vector along the direction from which the part model is viewed is chosen. Subsequently, all the nodes in the face-edge graph whose normal vector is parallel or
perpendicular to the aspect vector are deleted. The resulting subgraph is called the aspect face-edge graph (FEG-A). Next, simple cycles (face-to-face traversal beginning and ending at the same face) in the FEG-A are identified and labeled. After this step a pseudo-edge (p-edge) polygon in 3-D space and a projected p-edge polygon are generated. Based on some further analysis on the p-edge polygon and using the face adjacency relationship information (convex or concave) the pocket or hole features are identified. One of the major advantages of this feature extraction system is that it determines the perimeter of the projected area enclosed by each feature in a specific direction. This information is of vital use in the process planning activity. However, one of the disadvantages of this system is that all the walls of the pockets and holes identified should be parallel to the direction of the aspect vector. The proposed system is not good at detecting features that have undercuts such as a T-slot. In recent related work, [Clark and Corney, 1994] described similar steps for recognizing general depressions and protrusions of a part, which can then be mapped into domain-specific features.

[Fields and Anderson, 1994] introduced an oriented face adjacency graph (OFAG), similar to the face adjacency hypergraph (FAH), to address the problem of feature recognition. As with the FAH, the OFAG for a part has nodes that correspond to the faces of the part and arcs that correspond to the edges of the part. However, between any two nodes (faces) there are two directed arcs, each augmented with concavity/convexity information as well as information about where each face intersects with another. One face can intersect the exterior or interior of another face. Using this new graph, Fields presents a linear-time algorithm for matching templates within it for the recognition of features. The sacrifice involved in a time-efficient algorithm is that this approach classifies sets of faces in the part into categories of features. Although such a categorization is not necessarily a drawback, it does mean the features extracted are further removed from domain-specific features than in the approaches previously described.
The intrinsic of graph-based approaches is to extract features from the constructed face-edge graph based on predefined face patterns. Most graph-based approaches consider only a limited set of patterns for the definitions of features. Searching for more general definitions becomes complicated due to the large number of patterns. Thus face pattern based feature recognitions are limited by the available face patterns defined by the systems. They are not able to identify the features whose face patterns are beyond all the existing definitions. Thus some features may not be recognized due to the face pattern constraints even though they are not intersected with other features. In order to recognize features covered in the real-world parts as many as possible, graph-based methods have to consider all the possible face patterns, which is not practical. Our proposed bi-attribute based method (see Chapter 5) can prevent this rigidity by investigating the intrinsic and invariant properties underlying all the boundary entities and clustering the related entities for feature classification. If any entity clustering satisfies a defined pattern, a feature is identified. The unidentified entity clusterings indicate that new feature type or new patterns for the existing features are needed. The recognition system can automatically ask the users to update the system by adding new patterns or new feature definitions. Thus the maximal flexibility can be achieved in identifying features.

In addition, an important inherent problem in all graph-based techniques is computational complexity. Graph construction for the primitives and parts and subgraph isomorphism both can be computationally expensive. As discussed in [Sedgewick, 1984], the problem of subgraph isomorphism corresponds to determining whether two graphs can be made identical by renaming the vertices. The general problem remains computationally intractable, although efficient algorithms for special types of graphs are known. To ease this problem, research is now being directed towards reducing the required subgraph matching time by applying heuristics. [Peters, 1993] presents an example for the domain of sheet metal parts to show that engineering knowledge can be used to reduce a theoretical combinatorial explosion to tractable bounds. Researchers have also succeeded in dividing the graph representation of a part into feature and body components so that pattern matching only applies to feature components, thus reducing the search space.
[Joshi and Chang, 1988; Henderson, 1984; Henderson et al., 1990; Marefat and Kashyap, 1990; Marefat et al., 1990]. However, simply reducing the search space does not necessarily expand the domain of parts that graph-based methods can handle. To expand the scope of this work, mathematicians have developed efficient algorithms [Wang, 1992] for identifying the graph nodes that separate feature subgraphs from body subgraphs.

Computational complexity is not the only drawback of graph-based approaches. Graph-based approaches are typically weak at recognizing features that intersect. The feature graph representations are built from the topology of the part. However, in feature interactions the face adjacencies in the topology that are changed as a result of interactions make this information non-unique. Therefore, when exact matches are required in graph-based approaches, there is often difficulty in correctly identifying features within interactions.

Another potential limitation of graph-based approaches is verbosity in terms of the number of graphs required to represent the features. As illustrated in [Gadh and Prinz, 1992], for each primitive and its variants, a new set of graph representations must be created. Therefore, mechanisms must be developed to accommodate the variations in feature shapes.

2.2.2 Volume-based Feature Recognition

Sakurai, 1994; Tseng and Joshi, 1994; Sakurai, 1995; Dave and Sakurai, 1995; Sakurai and Dave, 1996] are the two major methods proposed in volume-based feature recognition.

**Convex-Hull Decomposition**

This technique is based on the idea of finding the materials that must be removed from a raw stock to produce a part. Instead of relying on pattern matching like the graph-based techniques previously mentioned, this approach exploits convex hull algorithms and Boolean operations for feature analysis. The convex hull is the smallest convex set that contains the polyhedral object. The recognition is attained by decomposing the object in stages as the (regularized) set difference from its convex hull.

To reflect the nature of this decomposition method, Woo [Woo, 1982] called the convex decompositions alternating sum of volumes (ASV). ASV decomposition represents an object by a series of convex volumes with alternating signs (for volume addition and subtraction). ASV decomposition works by first constructing a convex hull for the given object and then finding the regularized set difference between the object and its convex hull. If this set difference is empty, the ASV returns the object as a single convex object and terminates; otherwise, the object is partitioned into connected components and each connected component represents its deficiency. ASV decomposition is then reapplied recursively to each deficiency until the resulting set differences for all deficiencies are empty. Figure 2.9 shows the ASV decomposition of an example part. Here \( P \) stands for the object, \( CH(P) \) is the convex hull of \( P \), and \( CHD^*(P) \) represents the regularized convex-hull difference (deficiency).
Using the ASV technique, a non-convex polyhedron can be represented by a hierarchical tree (similar to a CSG tree) whose root node represents the object, intermediate nodes represent the union and set difference operators, and leaf nodes represent the primitive convex polyhedral sets or an empty set. There are certain major issues in convex-hull based techniques. The first is non-convergence: ASV decomposition does not always terminate, which limits the domain of geometric objects that ASV decomposition can handle. The ASV decomposition does not terminate when the convex hull of one volume in the decomposition is equal to the convex hull of that same volume's deficiency. Intuitively, such a situation creates a cycle in the ASV decomposition. That is, the same volume occurs over and over again within the ASV tree. Another issue is conversion from alternating sum-of-volumes to features. Although the components in the hierarchical ASV tree represent convex hulls and set-differences, these convex sets may
not represent features in general. Conversion into features is therefore required. The research by Woo et al., discussed in the following, represents the major work in this field.

Woo's [1983] work is the earliest effort in this category. In this approach, the ASV of an object $V$ can be formally defined as $\Omega = \sum (-1)^{i} H_i$ where $H_i$ represents each convex hull at different stages of decomposition.

No knowledge concerning the types of the volumes or their classification is produced. Human interaction is therefore required to guide the manufacturing of each convex hull component. The difficulties with this earliest effort include its limitation to convex parts because of lack of guaranteed convergence for general non-convex parts, or of conversion of the generated volume components into features for subsequent machining. Tang and Woo [Tang and Woo, 1991a; Tang and Woo, 1991b] also proposed an approach to reducing the issue of computational complexity involved in the ASV decomposition and describe an algorithm that checks if a given object will have a convergent ASV decomposition. For the computational complexity issue, they describe a linear space and an $O(n\log N)$-time algorithm and data structure for computing the difference between an object and its convex hull. An $O(N^2\log N)$ algorithm that determines if a part has a convergent ASV decomposition is also developed. This algorithm utilizes the algorithm for finding differences between objects and their convex hulls to check whether any given stage of the ASV decomposition is non-convergent. It then must check each such stage.

Kim and his colleagues [Kim, 1990; Kim, 1992; Kim, 1993; Kim and Roe, 1992; Waco and Kim, 1994a; Waco and Kim, 1994b] put great efforts in convex hull techniques for feature recognition: they solve the non-convergence and component volume conversion problems and therefore expand the application scope to concave parts. To remedy the non-convergence problem, they propose a modified version of the convex decomposition algorithm called alternating sum of volumes with partitioning (ASVP). The ASVP
decomposition begins with the steps for ASV decomposition until non-convergence is detected. Once non-convergence is detected, remedial partitioning is applied. Then, the ASV decomposition steps are reapplied to each partitioned component. Figure 2.10 illustrates the ASVP algorithm for a simple object.

Figure 2.10: ASVP decomposition of a simple part

Using the ASVP decomposition, they proposed an approach to identifying and extracting volumetric features from polyhedral objects: (i) ASVP decomposition is applied to the boundary representation of the given object to obtain the alternating sum of volumes; (ii) the ASVP volumes are then converted into feature entities by various combinations of
these volumes; (iii) recognized features are then classified into generic groups based on volume contribution and local accessibility information using the normal vectors of the original faces of each component. The various combination operations used to convert ASVP components into features include combination of volumes with opposite volume contributions and combining components with positive volume contributions. Since the resulting features may have both positive and negative components for machining applications, positive features are subsequently converted to negative features (which correspond to volumes to be removed from the part). This task is achieved by rewriting the Boolean expression of the positive components.

In summary, the convex-hull approach can handle most parts with interacting features by finding the alternating sum of volume decomposition. However, when features interact, certain common volumes may be shared by more than one feature, but the decomposition algorithm allocates the common volume to only one of the involved features. This problem may be addressed either by using feature-growing techniques or by recognizing features through hints. Also, it should be noted that it is difficult to construct convex hulls for curved objects. Although [Wang and Kim, 1997] enlarges the geometric domain of solids that ASVP can process by incorporating cylindrical features, it will get into trouble when numerous cylinder-cylinder interactions exist. ASVP extracts the cylindrical surfaces forming hole features and decompose those surfaces into a set of volumes representing whole or partial cylindrical cross-sections enclosed by the cylindrical surfaces. These volumes are combined with the part to fill the holes. The decomposition method is devised by combining cylindrical hull, set difference and cutting operations, which limit ASVP to the cylindrical features that interact with polyhedral features along principle directions, as shown in Figure 2.11a. Thus ASVP can not deal with the non-orthogonal intersection of planes and cylindrical faces, as shown in Figure 2.11b.
Cell-based Decomposition

Cell-based techniques are similar to convex hull techniques in that both decompose volume. However, unlike convex hull techniques which are concerned with decomposing the part volume, cell-based techniques focus on decomposing the delta volume of the part into smaller volumes by extending its bounding surfaces, then to reconstruct the sub-volumes into feature volumes which correspond to specific machining operations. The sub-volumes are often called cells and are used as the fundamental and intermediate connections among local topologies. Methods of this approach differ only in their decomposition and reconstruction criteria.

[Tseng, 1993; Tseng and Joshi, 1994] give an example of some recent work with cell-based feature recognition. First the volume to be machined is identified and decomposed into basic removable blocks (cells) by extending all the bounding faces and considering all possible intersections. This intuitively "cuts" the volume into pieces according to its own halfspaces. The next step involves the reconstruction of these cells into feature
volumes. Through 1D, 2D, and 3D cell connections a maximal connected feature block is created. Next, in order to classify the feature block, the nonsolid virtual faces are identified and deleted from the attributed adjacency graph (AAG) of the feature block. The classification of the feature block is then performed via matching of the resultant AAG through a simple rule-based algorithm. One advantage of this approach is that by recombining the cells in different orders it is possible to generate all the possible alternative interpretations of a part. The most prevalent drawback is that the faces of the delta volume (i.e., the faces used in the decomposition into cells) must sufficiently divide the delta volume itself. Therefore, the delta volume must be polyhedral, and if the delta volume of a part is convex, the cell decomposition will simply be one large cell.

Several researchers [Sakurai and Chin, 1993; Sakurai, 1994; Sakurai, 1995; Dave and Sakurai, 1995] also utilized cell-based techniques for feature recognition, extending the work of Tseng and addressing some of its weaknesses. Their work also begins by decomposing the delta volume of a part. However, they allow for curved faces to be recognized by matching cylindrical faces to portions of the curved faces. After the decomposition, these methods must also recombine the cells into features. The approach taken is first to combine the cells into maximal cells, which are sets of the cells that obey some proximity and adjacency specifications. These maximal cells are then used to find the features of the part: they are subtracted from each other to produce features. It should be noted that these methods are also well suited to generating alternate interpretations of a part by subtracting the maximal cells in different orders.

Cell-based techniques reveal serious problems in both the decomposition and composition steps. The main problem in the cell decomposition is the global effect of local geometry. A feature usually leaves its traces (faces) in a localized area of the part. However, the cell decomposition step extends the surfaces or halfspaces associated with the faces globally (within the delta volume) and quite often generates a huge number of cells that will result in the combinatorial explosion for cell composition. As early as in 1981, Tilove [Tilove, 1981] pointed out that nearly all the mechanical parts exhibit a high
degree of locality, and that failure to exploit this results in unacceptably slow algorithms. Thus, it is of practical importance to improve and apply locality techniques for use with important applications, and of theoretical interest to seek to place the techniques in a common, unifying framework. This is one of the major issues considered in this research work. We exploit the locality by recognizing the basic surface features (Chapter 5), and use the basic surface feature to efficiently construct the corresponding volumetric feature (Chapter 6) by halfspace partition. Thus the problem of combinatorial explosion can be avoided so as to achieve better computational efficiency.

2.2.3 Hybrid Feature Recognition

From the above observations, we can conclude that graph-based method, a typical boundary-based method, can identify the surface feature based on subgraph matching, but suffer from lack of generality in dealing with interacting features; while volume decomposition is suitable to deal with feature interactions, but is computationally expensive due to the arbitrary subdivision of the part in the elementary volumes. Thus it is very natural to consider building a hybrid method that can utilize the advantages both from the boundary-based and volume-based techniques to achieve better recognition performance. The hint-based methods [Vandenbrande and Requicha, 1993; Han and Requicha, 1995; Regli and Rau, 1995] represent the efforts towards hybrid methods.

Hint-based method was first proposed by [Vandenbrande and Requicha, 1993]. A feature hint is defined as a feature’s geometric entities left in the nominal geometry of a part after a feature is removed. For example, a pair of parallel opposing faces is a hint that a slot may be present. To recognize arbitrarily interacting features, they first extract all the feature hints by searching the B-Reps of the part, then test them for validity through geometric completion procedures, that attempt to construct the largest volumetric feature based on a variety of AI and computational geometry techniques. Finally, they represent a feature’s interaction with others by segmenting the feature into optional and required volumes. This approach deals with feature interactions in a general manner. However, since the hint generation and completion algorithms in this approach are not general, but
depend on the specific feature type, new algorithms need to be developed and added to
the system in order to recognize new, user-defined feature classes. Usually, many hints
need to be explored for almost every feature class, which may become cumbersome for
some feature classes like a dovetail slot. Han and Requicha [Han and Requicha, 1995;
Han, 1996] extended Vandenbrande’s work. A reasoning method for rating and
prioritizing the hints was added to further reduce the possibilities to be explored and the
order in which to explore them.

The hint-based feature recognition method has also been investigated by Regli and Rau
[Regli, Gupta and Rau, 1994; Regli, 1995]. Their approach addresses a class of
machinable features expressible as MRSEV (material removal shape element volumes).
The problem is that all the prismatic cavities are classified as pockets in this method,
which makes it difficult to machine some types of features such as T-slots with methods
particularly efficient for them. As with Vandenbrande’s work, there are also many hints
needed.

The hints proposed by Vandenbrande, Han and Requicha are limited to the regular
features (slot, hole and pocket) that are machined by a cutter along a fixed direction. Thus
the hints are related to the parallel and perpendicular relations between planar faces and
the selected machining direction. This severely limits the domain of the recognizable
features. In real-world mechanical parts, the geometry and topology of the features can be
various due to different manufacturing processes and machining methods. As pointed by
Han’s dissertation [Han, 1995], in order to handle other types of features, the feature hints
have to be extended. Another drawback of the hint-based system is that it can not
recognize the pocket features with edge blendings at the bottom face (or floor face) (see
Figure 2.12) since it assumes only flat-end milling process is used. As we see, the hint
generation based on limited machining process poses the same problem that rule-based
system has – rigidity. In fact, hints are rules defined based on certain geometric and
topological constraints. Thus this can be limited by the ad hoc nature of rules. A rule
written for a particular configuration of a feature cannot be generalized to cover a similar
feature in a different configuration (e.g., due to feature interaction). In addition, the exhaustive searches needed to match rules against the solid model data representation make the method quite slow. Thus the key to a successful hybrid method is to provide an efficient way for hint generation that is free from rules. We believe it is essential to study the underlying invariant properties of the boundary entities of the part model and come up with a uniform way to recognize various surface features with different topology and geometry that form the hints for volumetric feature extraction.

![Figure 2.12: Unrecognizable feature in hint-based method](image)

Another problem exists in current hybrid approaches is that they only focus on the machining features or negative volumetric features. Thus the output can only be used in process planning. This can not meet the information requirements of multiple applications in concurrent engineering. Considering design intent recovery, positive volumetric features should be recognized.
2.3 Research Issues for Automatic Feature Recognition

Although various approaches have been proposed to solve automatic feature recognition, there are still some open technical issues that affect the development of a general, reliable and robust method, such as application independent features, feature interactions, and multiple feature interpretations.

2.3.1 Application Independent Feature

The primary objective of having feature information is to intelligently reuse CAD data in design scheme optimization and various manufacturing applications such as process planning, inspection planning, and finite element analysis (FEA). Current methods of automatic feature recognition often focus on the applications in machining such as process planning. The specific aim of these methods is to identify material removal volumes, or machining features. Thus it is not really capable of addressing the issues in capturing the design intents. General-purpose automatic feature recognition should be able to extract any subset of geometric and topological entities from CAD models and needs to be flexible enough to support design intent recovery and various manufacturing applications. Since the application domains such as design, process planning, inspection planning and FEA take different views of the features [Kraker, Dohmen and Bronsvoort, 1995], most researchers are convinced that no single set of features can satisfy the requirements of every possible design and manufacturing domain. Thus the feature recognized should not be immediately associated with applications. It would be highly desirable to have a means to capture the higher-level neutral feature information that is application-independent and convert the neutral features into application-specific features that can be directly used by the manufacturing applications. The separation of general geometric reasoning about application-independent features from specific reasoning about application-dependent features can provide a novel way to construct an information framework that tightly and efficiently integrates the automatic feature recognition into the concurrent engineering environment.
Hence there is a need for an application-independent feature recognition that can automatically identify features from part models. Application-independent features (AIFs), a.k.a. neutral features, are the generic shapes that are independent from specific applications such as depressions (negative volumes) and protrusions (positive volumes). AIFs are different from the machining features in the field of process planning, NC code generation and cost estimation, which are related to material removal volumes or pure negative volumes with functional meanings. Such application-independent features can then be mapped or converted to application-specific features depending on the applications' requirements. Thus AIFs can provide the maximum flexibility to support various downstream applications. In this research, a bi-attribute based method is proposed in Chapter 5 to recognize surface-based AIFs.

2.3.2 Feature Interaction

Feature recognition generates a feature-based model suitable for specific applications by recognizing specific features in its solid model. It has been studied for two decades but still there has been no satisfactory feature recognizer. The major difficulty in feature recognition has been the recognition of interacting features. A feature can interact with any number of features. Due to interactions, some of the faces that correspond to a feature may be entirely absent, partially missing, or fragmented into several regions. Faces may be shared without clear demarcations between features. Edges may be missing or unexpected edges may appear. This interaction can distort the individual feature instances and information that is used for recognizing a feature can be lost. Feature interactions pose a major challenge to the development of robust and reliable feature recognition systems. The ability to handle interacting features has become an informal benchmark for feature recognition systems [Regli, 1995] and has been the focus of numerous research efforts.

Boundary-based feature recognizer typically examines the topological and geometric characteristics of the boundary entities of a B-Rep solid model. Graph formalism is commonly used to express the topological and geometric characteristics of features, and
graph matching, which matches the feature graphs with the B-Rep graph of the solid model, is used to recognize features in a solid model. Inherent in these approaches that check topological and geometric characteristics of the boundary entities of a solid model is the inability of recognizing interacting features. When features interact, their topology changes and thus the approaches do not work except some limited patterns of interactions. The same problem will happen to our proposed bi-attribute based surface feature recognition since intrinsically it is a boundary-based approach.

In volume-based approaches, cell-based methods can solve feature interactions to some extents, but most of them rely on simple case-by-case or implementation-specific heuristics. In addition, the computational complexity due to exhaustive volume decomposition and selected composition became the major obstacle for practical adoption of volume decomposition methods. Although hint-based methods put some efforts to come up with a hybrid way to deal with feature interactions, the rule-based hint generation is not general and reliable.

The better way to handle the feature interaction is to utilize the advantages from boundary-based methods and volume-based methods. In this research, we devise a bi-attribute based method to recognize discrete basic surface features, which provide general and reliable hints for volumetric feature extraction (see Chapter 5). Maximal feature volumes are constructed by halfspace partition for each hint during the feature recognition with the feature interaction resolved on the fly (see Chapter 6). By this way we can solve feature interactions without losing generality and computational efficiency.

2.3.3 Multiple Interpretations

Usually a mechanical part can be machined in different ways, i.e. different process plans. Each process plan may have different time and cost performance under different machining conditions such as tool setup, fixture configurations. A process plan may be
optimal under a set of machining conditions, but may not be best under another set of machining conditions. Thus it is important to consider all the feasible feature interpretations.

Figure 2.13: Different feature volumes for a hole feature considering multiple interpretation

In order to ease the difficulty in dealing with feature interactions, some researchers [Sakurai and Gossard, 1990; Zhang et al, 1998] partitioned the feature recognition into two steps. The isolated features are identified first from the part model and suppressed by uniting the isolated volumetric features with the part model. Then the interacted features are recognized by certain methods either already existing or devised from scratch. One of the obvious disadvantages of this partition method is that the so-called isolated features may not be absolutely separated from other features. As we can see from Figure 2.13, the partition method will recognize the blind hole feature and unite its feature volume (shown in Figure 2.13b) with the part model. The problem is that if the hole is machined before the step, the actual feature volume of the hole will extend to the boundary of the stock volume as shown in Figure 2.13c. Thus its feature parameter will be different due to
larger height. Since the part can be manufactured in different processing plans, multiple feature interpretations are desired. In order to obtain multiple interpretations, all the features should be viewed having the same priority before reasoning a complete feature interpretation. This requires all the features should be recognized at the same time whether it is an isolated feature or a feature having interactions with other features. Based on this consideration, our proposed approach can identify all the surface features from the part model and then construct the maximal feature volume for each surface feature in a consistent way. Since no order exists in recognizing each feature, multiple feature interpretations are well supported.

In machining application, multiple interpretations of a part in terms of machining features correspond to different ways to machine a part and therefore provide a process planner with added flexibility. From the viewpoint of concurrent engineering, the multiple interpretations of the feature model of a part are not limited to machining domain. They are appropriate combinations of so-called positive and negative features for different applications such as design, inspection planning, and assembly. This is one of the most convincing reasons to recognize positive and negative features at the same time in this research work. If the machining application is considered, only the negative feature volumes are selected and combined as a feature interpretation, called a machining feature model. If we are interested in recovering the design intents and generating a design scheme for the part, positive feature volumes and all the negative feature volumes that are not adjacent to any positive feature volume are selected to form a suitable interpretation, called a design feature model. Chapter 7 explores some issues and techniques in multiple feature interpretations.

2.4 Geometric Modeling Kernel vs. Feature Reasoning Kernel

Geometric modeling kernel put together all the knowledge on the construction of solid models into comprehensive and efficient geometric and topological data architecture and an abundant set of operations. It has matured to be adopted in many commercial CAD
systems and forms an indispensable building foundation for the development of the state-of-the-art feature-based design packages. The two most successful commercial geometric modeling kernels are ACIS (from Spatial) and Parasolid (from Unigraphics). They share the similar data architecture, which includes a hierarchical structure of geometric entities such as surface, curve, and point and topological entities such as body, lump, shell, face, edge and vertex.

Feature reasoning research emerged not long after geometric modeling research appeared [Braid, 1993; Voelcker and Requicha, 1993]. However, a general, reliable and robust feature reasoning system has not been delivered. One of the major reasons is that the available feature recognition methods are built on a limited feature domain and can not meet the requirements from the real-world mechanical parts that have various complex features caused by feature interactions. Another reason is that no efforts have been spent to systematically modularized the research results in software engineering. In order to advance the progress towards a general, reliable and robust feature recognition system, a kernel architecture is highly desirable for feature reasoning. In order to ensure the extendibility and ease-of-use, the feature reasoning kernel needs to provide a modularized and open architecture, which is the most successful point of modern geometric modeling kernels such as ACIS and Parasolid.

The processing target of feature reasoning kernel is a B-Rep geometric model, which is constructed from geometric modeling kernel. The feature-based model generated by feature reasoning kernel should contain abundant information supporting various concurrent engineering applications such as design optimization, process planning and inspection planning. It is evident that the feature reasoning in automatic feature recognition is involved with intensive geometric computation and topological interrogation, which is also supplied by geometric modeling kernel. Thus feature reasoning kernel is closely interrelated with geometric modeling kernel as illustrated in Figure 2.14.
Our research made the first effort to put together the developed feature recognition work into a kernel architecture, as discussed in Chapter 3. A prototype system named Feature Explorer is built based on the developed feature reasoning kernel with tight integration with a modern geometric modeling kernel, as discussed in Chapter 8.

2.5 Summary

While automatic feature recognition has become a promising technology in re-utilizing legacy CAD data and advancing information automation in CAD/CAM integration, many approaches that have been developed employ techniques that are inherently limiting, either representationally or in terms of the computational complexity of the reasoning and recognition algorithms. Few approaches have demonstrated the ability to scale to real world mechanical parts and even no one has made the effort to kernelize the feature recognition work for concurrent engineering.
CHAPTER 3

OVERVIEW OF THE HYBRID FEATURE RECOGNITION FOR MULTIPLE APPLICATIONS

This chapter gives an overview of our proposed hybrid feature recognition for multiple applications, which is implemented as a feature reasoning kernel that consists of four functional modules: pre-recognition processing, basic surface feature recognition, volumetric feature extraction and post-recognition processing.

3.1 Introduction

Geometric modeling kernels provide a foundation for solid modeling of the mechanical artifacts. They deal with the manipulation and management of low-level geometric and topological entities. In order to efficiently identify the lost high-level features from the pure geometric models and provide a consistent representation of features to support feature modeling – the indispensable high-level product modeling, a feature reasoning kernel built on top of the geometric modeling kernel is in great demand. The desired feature reasoning kernel needs to be able to extract not only the explicit surface boundary and volumetric information of the features, but also the implicit abstract parameters. The integration of explicit and implicit feature representation can achieve a complete feature representation.
3.2 Architecture

As shown in Figure 3.1, the input to the developed feature reasoning kernel is a pure geometric B-Rep model, i.e. a B-Rep model without built-in feature information. It can be a design work from traditional solid modeling or a CAD model generated from reverse engineering or other resources. The B-Rep model is adopted in this research since it has the advantage of providing explicit geometric and topologic information and it has been widely chosen as the final geometric representation of parts by nearly all the CAD packages.

In order to facilitate the geometric reasoning for feature-based model generation from pure geometric B-Rep models, the feature reasoning kernel consists of four main functional modules: pre-recognition processing, basic surface feature recognition, volumetric feature extraction and post-recognition processing. Pre-processing module is incorporated to simplify the input B-Rep models by automatic fillet/round suppressing for efficient feature recognition. Basic surface feature recognition module and volumetric feature extraction module form the core of automatic feature recognition. Basic surface feature recognition investigates the underlying invariant properties of boundary entities and characterizes all the topological entities in proposed bi-attribute. Based on these attributes, primitive forms and shapes of an object can be extracted based on the clustering of surface entities and represented by basic surface features. Volumetric feature extraction constructs maximal feature volumes by halfspace partition based on the basic surface feature and solves feature interaction. Post-processing is employed to ensure the consistency and completeness of derived feature information. Several functions are performed in post-recognition processing. One is the restore of suppressed fillet/round to recover the original B-Rep model from its simplified counterpart. Another function is to manipulate the maximal feature volumes through selection and sequencing to suit the requirements of applications and processes. After selection and sequencing the feature volumes are re-computed and their semantics are updated. An explicit feature model will be built based on the integration of surface features and volumetric features.
Figure 3.1: The feature reasoning kernel architecture of the proposed hybrid feature recognition for multiple applications
The more abstract implicit feature representation such as feature parameters is extracted from the explicit feature model to provide the necessary information for other applications such as feature-based tolerancing and dimensioning, an indispensable functional module (not within the scope of this research) to build a complete feature-oriented product model for various concurrent engineering applications such as design optimization, process planning and inspection planning.

3.3 Advantages

The proposed feature reasoning kernel has the following advantages over the current available approaches:

(1) The systematic and uniform two-phase automatic feature recognition can deal with a large class of features and their interactions without loss of computational efficiency;
(2) It can extract application and process independent features such as positive features and negative features, which can be converted and transformed to the appropriate feature representations based on the requirements of applications such as design optimization and process planning;
(3) It functions as a software component that can be easily incorporated into current commercial CAD software packages.

A software component is a functionally specialized unit of software – a collection of software items (functions, classes, etc.) grouped together to serve some distinct purpose. Software component technology provides an efficient way to compose and maintain a software package. The proposed feature reasoning kernel is programmed as a software component with several functional modules in object-oriented structure. While all the functional modules are interrelated, each functional module can be extended without affecting the other ones. Chapter 8 presents the implementation details of the feature reasoning kernel and the incorporation of it into a prototype system named Feature Explorer.
3.4 Summary

An overview of the proposed feature reasoning kernel is given. The major advantage of the proposed feature reasoning kernel is to provide a uniform framework to deal with a large variety of features encountered in the real-world mechanical parts. From the viewpoint of software engineering, it can function as a software component that is ready for direct incorporation into current CAD packages to enhance their feature reasoning capability.
CHAPTER 4

PRE-RECOGNITION PROCESSING

The CAD models of real-world mechanical parts usually have many fillets and rounds that are essentially important to ensure the manufacturability and assemblability. In feature-based modeling, fillets and rounds are often referred as secondary features that are used to modify the local details of the primary features such as holes, slots and pockets. Although the major shape of the primary features may not be affected, fillets and rounds can greatly change the geometric and topological patterns of the primary features. The geometric and topological variations can result in inefficient feature semantics classification in feature recognition. When feature interactions occur, it may become even worse to identify the regular patterns of the primary features. In addition, the fillets and rounds consist of no-linear surfaces such as cylindrical surfaces, spherical surfaces and toroidal surfaces, which bring the difficulties in volumetric feature extraction by half-space partition. In order to facilitate volumetric feature extraction and feature semantics classification, we pre-process the input B-Rep models by suppressing fillets and rounds before the feature recognition. Thus the input B-Rep models can be simplified without altering the major shapes of primary features, the targets of the feature recognition. The B-Rep simplification can be viewed as the reverse process of the edge blending in feature-based design. In this chapter, several issues on fillet/round suppressing are discussed and a relatively general and robust approach is proposed to suppress blendings in B-Rep models of mechanical parts before the surface feature recognition and volumetric feature extraction.
4.1 Introduction

Fillets and rounds are common features in the real-world mechanical parts. They are used in mechanical part design to provide a transition between different surfaces of a solid. This results in smoothing of sharp corners and edges in a part and thus improves safety of handling, enhance strength by reducing stress concentrations, and provide aesthetic appearance [Holmstrom & Laakko, 1988]. From the machining point of view, fillets are generated in two major ways: one is the by-product by using the cylindrical cutting tool or ball end cutting tool; the other is specially machined by using the filleting cutting tool [Regli, 1995]. From the design point of view, fillets and rounds are the secondary features formed by applying blending operations on the sharp edges and vertices of the primary features such as slots, holes and pockets.

Fillets and rounds are formed by applying blending operations on sharp edges of solid models [Laakko and Mantyla, 1993b]. Since the blending operations cause geometric and topological variations, they bring difficulties for the classification of surface feature semantics and the construction of volumetric features in automatic feature recognition. Among the published works in automatic feature recognition, no efficient ways exist to deal with the blending features. Many of those published works only focused on unblended B-Rep models and avoided the discussion of the handling of blending features. Thus we tried to solve this problem by totally suppressing the blending features in B-Rep models to get a simplified B-Rep model without blending features. Figure 4.1 illustrates some possible variations of fillets and rounds in a notch feature. As we can see from this figure, if fillets and rounds are removed from the B-Rep model, a canonical feature boundary can be achieved for this notch feature, which will obviously support the efficient feature type classification. Thus it is desirable to suppress the blendings in B-Rep models to facilitate feature recognition. Since blendings are secondary features used to modify the primary features, the suppressing will not affect the recognition of primary features.
In this chapter three issues involving the B-Rep model simplification are discussed in Section 4.2. Section 4.3 presents the methodology of the B-Rep model simplification by automatic fillet/round suppressing, in which the framework is introduced first, and then two parts are discussed in details, including fillet/round identification by trace faces and suppressing by an incremental knitting process. Section 4.4 gives two typical examples to verify the proposed approach. At last conclusions are discussed in Section 4.5.
4.2 Issues in B-Rep Model Simplification

The suppressing of the blendings will change the geometry and topology of the original B-Rep model. Thus, in order to prevent information missing and to obtain a complete correct manifold B-Rep model after the fillet/round suppressing, we need to pay attention to at least three issues including geometric and topological consistency and reversibility.

4.2.1 Topological Consistency of Simplified B-Rep Model

After removing the faces that form the fillets and rounds, gaps will generate between the faces adjacent to the removed faces, which will result in a non-manifold [Weiler, 1986] object that has an open boundary. Real-world 3D objects cannot have any non-manifold forms such as open boundaries and dangling or isolated faces, edges and vertices. Thus the simplified B-Rep model must be a manifold (two-manifold more precisely) solid to be a valid object. No gaps or non-manifold entities are allowed in the simplified model. This requires new topological entities such as edges and vertices to be added to fill the gaps after suppressing the fillets and rounds. Considering the structure requirements of B-Rep models, no redundant edges and vertices are allowed in the final model after filling gaps.

4.2.2 Geometric Consistency of Simplified B-Rep Model

From the viewpoint of feature-based design, the fillets and rounds are formed by applying blending operations on sharp edges. The suppressing of fillets and rounds can be viewed as a reverse process of blending operations. Therefore, sharp edges corresponding to the removed faces are recovered and added to ensure the topological consistency. Every topological entity must have its underlying geometry to form a complete representation in a B-Rep model. The underlying geometry of the newly created sharp edge must satisfy the constraint that the new edge should be on the two neighbor faces of the corresponding removed face. This geometric consistency can be achieved by performing the extended face intersection between the two neighbor faces. Similarly, the geometric consistency for newly created vertices can be achieved by performing the extended edge intersection.
4.2.3 Reversibility

The purpose of B-Rep model simplification by fillet/round suppressing is to facilitate feature recognition, especially in the volumetric feature extraction and feature semantics classification. In order to be able to recover the original B-Rep model, the information of the suppressed fillets and rounds should be kept in the simplified B-Rep model, which can be achieved by attaching certain attributes to the newly created edges. The attached attribute will record the radius of the suppressed fillet or round and other data such as the property distinguishing fillets from rounds. The original B-Rep model can be computed by reapplying the blending operations on those attributed edges of the simplified B-Rep model. Thus, reversibility can be guaranteed by this attribute management. The feature recognized from the simplified B-Rep model can also recapture its actual boundary and volume by applying the blending operations on the attributed edges in its boundary.

4.3 Methodology

4.3.1 The Framework of the B-Rep Model Simplification by Fillet/Round Suppressing

In order to obtain a complete simplified B-Rep model by fillet/round suppressing, a systematic approach, as shown in Figure 4.2, is developed to guide the information processing. First, we characterize the convexity of all the topological entities (faces, edges and vertices) in terms of bi-attribute (see Chapter 5) which will be used to facilitate the identification of trace faces. Then based on trace faces the identification of fillets and rounds in the original B-Rep model is performed. After these two steps, the suppressing is performed by cleaning all the related topological entities of the fillets and rounds and filling the resulting gaps due to the suppressing by an incremental knitting process to ensure the geometric and topological consistency of the simplified B-Rep model. In the incremental knitting process, the blending attribute is attached to the newly generated
edges corresponding to the suppressed trace faces to ensure the reversibility. At the end of knitting process, all the attached bi-attributes are removed from the topological entities so that the input to the feature recognition is a clean B-Rep model.

Figure 4.2: The framework of the B-Rep model simplification by fillet/round suppressing

4.3.2 Fillet/Round Identification by Trace Faces

In order to suppress fillets and rounds in the B-Rep model of a mechanical part, we need to identify them first. Thus the formation of the fillets and rounds is investigated first to provide a hint for the desired identification.

4.3.2.1 Formation of Fillets and Rounds

From the viewpoint of feature-based design, fillets and rounds are formed by the blending operations on sharp edges. Blending may either add material to or remove material from a model, depending on the model convexity local to the blending. A blending on a convex edge removes material from the model, as shown in Figure 4.3a. This rounds out an external corner, removing excess material. This type of blending is referred to as a round. A blending on a concave edge adds material to the model, as shown in Figure 4.3b. This type of blending is referred to as a fillet. Thus rounds are blendings on convex
Figure 4.3: Illustration of the formation and suppressing of round and fillet
(a) round; (b) fillet
sharp edges that remove material from the blank body, while fillets are blendings on concave sharp edges that adds material to the body being blended. As illustrated in Figure 4.3, the suppressing is a reverse operation of the blending. For rounds, material is added to form the original convex sharp edge; for fillet, material is removed to form the original concave sharp edge.

4.3.2.2 Classification of Trace Faces

Since the fillets and rounds are formed by applying the blending operations on sharp edges, new faces will generate to replace the to-be-blended sharp edges. The blending operations will also cause the generation of smooth edges between the new faces and their adjacent faces. Thus the newly generated faces will have certain typical geometric and topological characteristics, and these characteristic faces are defined as trace faces when considering the identification of fillets and rounds. Considering automatic identification in this research, we only focus on constant-radius fillets and rounds, which are the most common edge blending features in real-world mechanical parts.

By carefully studying the geometry and topology of the trace faces formed due to the edge blending operations, we can classify the trace faces into the following classes that satisfy certain geometric and topological constraints:

**Toroidal faces:** Toroidal faces are the torus-type faces formed due to blending operation on circular edges and character vertices (see Chapter 5). As shown in Figure 4.4, faces $F_1$, $F_2$ and $F_3$ are toroidal faces from the blending on circular edges $E_1$, $E_2$ and $E_3$, and faces $F_4$ and $F_5$ are toroidal faces from the blending on character vertices $V_3$ and $V_4$.

**Cylindrical faces:** Cylindrical faces are the cylinder-type faces formed due to blending operation on straight edges. The faces $F_8 \sim F_{15}$ in Figure 4.4 are cylindrical faces from the blending on straight edges $E_4 \sim E_{11}$. We limit the spanning angle of the cylindrical faces to be less than 180° based on the following justification:
Figure 4.4: Various trace faces formed by the blending on edges and vertices
(left) unblended B-Rep model; (right) blended B-Rep model

Figure 4.5a shows a section view of a sharp corner blended by a cylindrical face. Angle $\alpha_1$ is defined as the angle of the sharp corner. When the edge along the sharp corner is convex, $\alpha_1$ is in the material side as shown in Figure 4.5b. When the edge along the sharp corner is concave, $\alpha_1$ is in the non-material side as shown in Figure 4.5c. For regularized manifold solid no cusp and embedded edge are allowed, i.e. $\alpha_1$ cannot be $0^\circ$ and $180^\circ$. Thus we always have $0^\circ < \alpha_1 < 180^\circ$. Angle $\alpha_2$ is defined as the spanning angle of the blending cylindrical face. From Figure 4.5a, we can easily have the following constraints:

$$\alpha_1 + \alpha_2 = 180^\circ$$

Therefore,

$$0^\circ < \alpha_2 < 180^\circ$$

i.e. the spanning angle of the blending cylindrical face is less than $180^\circ$. Based on this
constraint, we can see from Figure 4.6 that the cylindrical faces $F_1$ and $F_2$ are trace faces while $F_3$ is not a trace face. In fact, face $F_3$ forms the indispensable cylindrical end face of the blind slot [Vandenbrande, 1990] and should not be suppressed.

Figure 4.5: Illustration of the spanning angle of a blending cylindrical face in section view

(a) convex
(b) concave
Spherical faces: Spherical faces are the sphere-type faces formed due to blending operation on convex or concave vertices incident to straight edges only. Any face in this type is surrounded by three smooth circular boundary edges. The faces $F_6$ and $F_7$ in Figure 4.4 are spherical faces from the blending on concave vertex $V_1$ and convex vertex $V_2$ respectively.

All the faces that satisfy the above geometric and topological characteristics are identified as traces faces of fillets and rounds.

4.3.2.3 Definition of Trace Face Chain and its Homeomorphic Equivalence

Usually several trace faces are connected together to form a chain. If the trace face chain forms a closed loop, it is called a closed trace face chain; otherwise it is called an open
trace face chain. Figure 4.7 illustrates two closed trace face chains and two open trace face chains. Among them one closed trace chain and one open trace chain both consist of only one face. They are the special cases of trace face chains.

Based on the concept of homeomorphism [Firby and Gardiner, 1991] (see detailed discussion in Section 5.2.1), any surface in $E^3$ space can be deformed elastically to become a disc in $E^2$ space. If we deform a closed face chain in $E^3$ space to $E^2$ space, a ring can be formed as shown in Figure 4.7. An open face chain will become a disc under the deformation since no loop exists. Thus the homeomorphic equivalence in $E^2$ (H.E. in

Figure 4.7: Closed and open trace face chains and their homeomorphic equivalence in $E^2$
short) of a closed face chain in $E^3$ is a ring, and the closed face chain is defined as a ring-type chain. The H.E. of an open face chain in $E^3$ is a disc, and the open face chain is defined as a disc-type chain.

The intrinsic of fillet and round suppressing is to remove the trace faces from the B-Rep model. After removing trace faces, the resulting gaps need to be filled to form a complete B-Rep model. It is observed that only the immediately adjacent faces of the trace faces would be affected after the face removing. The immediately adjacent faces are all the faces around the to-be-removed chain, and they are defined as enclosure faces. The ring-type chain separates its enclosure faces into outside enclosure faces and inside enclosure faces. All the enclosure faces of the disc-type chain are outside of it. Therefore, the H.E. of the enclosure faces of the ring-type chain and the H.E. of the enclosure faces of the disc-type chain will be different after the chain removing.

![Figure 4.8: A ring-type chain in a cylinder and the H.E. of its enclosure faces](image)

Figure 4.8: A ring-type chain in a cylinder and the H.E. of its enclosure faces
Considering a simple cylinder with a ring-type chain as shown in Figure 4.8, the face set \( \{ F_2 \} \) forms the ring-type chain and \( \{ F_1, F_3 \} \) are enclosure faces. The H.E. of the face set \( \{ F_1, F_2, F_3 \} \) is a solid (non-empty) disc \( D_1 \), and the H.E. of the ring-type chain \( \{ F_2 \} \) is a solid (non-empty) ring \( R_1 \). The removing of the ring-type chain equals the subtraction of \( R_1 \) from \( D_1 \). Thus an empty ring and a solid disc will generate as shown in Figure 4.8. In fact, the subtraction result is the H.E. of the enclosure faces \( \{ F_1, F_3 \} \) after chain removing. We can also have the following observations from the subtraction results: the outer solid ring is the H.E. of \( \{ F_3 \} \) — the outside enclosure faces of the ring-type chain; the middle empty ring is the H.E. of \( \{ F_2 \} \) — the ring-type chain that is to be removed; the inner solid disc is the H.E. of \( \{ F_1 \} \) — the inside enclosure faces of the ring-type chain. Thus the H.E. of the enclosure faces of a ring-type chain is a solid ring with an embedded solid disc, and the H.E. of the removed ring-type chain is an empty ring.

![Figure 4.9: A disc-type chain in a block and the H.E. of its enclosure faces](image-url)
The H.E. of the enclosure faces of a disc-type chain is different from that of a ring-type chain. A disc-type chain in a block is shown in Figure 4.9. \( \{F_3\} \) forms the disc-type chain and \( \{F_1, F_2, F_3, F_4\} \) are the enclosure faces of the disc-type chain. It is seen that the H.E. of the enclosure faces of the disc-type chain is a solid ring and the H.E. of the removed disc-type chain is an empty disc.

Based on the above H.E. analysis, we know that the H.E. of the four chains in the example part in Figure 4.7 includes two solid rings and two solid discs. From the viewpoint of H.E., the chain removing equals the subtraction of the two solid rings and two solid discs from a larger disc that represent the face set including the enclosure faces and the four chains. Thus the H.E. of the enclosure faces after chain removing is a solid disc with two embedded empty rings and two embedded empty discs as shown in Figure 4.10. The generated embedded solid discs represent the inside enclosure faces of the two ring-type chains.

![Figure 4.10: The H.E. of the enclosure faces of the example part in Figure 4.7 after chain removing](image_url)
4.3.3 Fillet/Round Suppressing by an Incremental Knitting Process

From the above discussion, we know that the trace faces of fillets and rounds exist in the part boundary in chain form, either ring-type chain or disc-type chain. The chain removing results in the generation of empty rings and discs in the H.E. of the enclosure faces of all the chains. The empty rings and discs can be viewed as the gaps left in part boundary after chain removing. Thus new topological entities such as edges and vertices are needed to fill the gaps to build a complete B-Rep model. An incremental knitting process is proposed to deal with the gap filling.

![Original B-Rep Model](image1)

![Simplified B-Rep Model](image2)

Figure 4.11: Illustration of some terms for the proposed incremental knitting process

4.3.3.1 Topological Terms

Before we further present the incremental knitting process, some topological terms are given first besides the ones mentioned above such as ring-type chain, disc-type chain,
enclosure faces, outside enclosure faces and inside enclosure faces. In order to facilitate the term definitions, an example part is illustrated first in Figure 4.11, where \( \{F_1, F_2, F_3, F_4\} \) forms a disc-type ring, \( \{F_5, F_6, F_7, F_8, F_9, F_{10}\} \) forms the enclosure faces.

**Enclosure Edge:** An enclosure edge is an edge that is between two enclosure faces and immediately adjacent to a trace face chain, such as \( E_1-E_6 \) in Figure 4.11. We observed that enclosure edges are those edges whose topology will be changed in the original B-Rep model after the chain removing. As shown in Figure 4.11, the enclosure edge \( E_1 \) has two adjacent enclosure faces \( F_5 \) and \( F_8 \) and is connected to the trace face chain through end vertex \( V_1 \), which is replaced by \( V_1' \) in the simplified B-Rep model.

**Forward Enclosure Coedge & Backward Enclosure Coedge:** In the topological structure of a B-Rep model, an edge has two coedges associated with it, each one joining the coedge loop of the edge’s adjacent face. Thus an enclosure edge will have two coedges. The coedge pointing toward the trace face chain is defined as a forward enclosure coedge, such as \( e_r \) in Figure 4.11, and the coedge pointing away from the trace face chain is defined as a backward enclosure coedge, such as \( e_b \) in Figure 4.11. Since the topology of the enclosure edge will change after chain removing, the topology of the associated forward enclosure coedge and backward enclosure coedge will change as well.

**Demarcation Edge:** Demarcation edges are the edges that separate the trace faces from the enclosure faces. The demarcation edges are always connected and form an edge loop. For a disc-type chain, one demarcation edge loop exits and it corresponds to the boundary of the empty disc. For a ring-type chain, two demarcation edge loops exist and they correspond to the two boundaries (inner and outer) of the empty ring. As shown in Figure 4.11, all the highlighted edges in the original B-Rep model are demarcation edges and form an edge loop enclosing all the trace faces.

**Construction Edge & Construction Vertex:** A construction edge is a constructed new edge in the simplified B-Rep model to replace the trace faces, such as \( E_1', E_2' \) and \( E_3' \) in
Figure 4.11. A construction vertex is a constructed new vertex that forms the end vertex of a construction edge and also the new end vertex of an enclosure edge, such as $V_1'$, $V_2'$, $V_3'$, $V_4'$ in Figure 4.11.

**Floating Edge:** A floating edge is an intermediate edge whose topology and underlying geometry are not fully defined. Floating edges are introduced to facilitate the incremental knitting process proposed in the following.

**Floating Vertex:** A floating vertex is a vertex on an enclosure face satisfying these two conditions: (1) adjacent to the trace face chain; (2) not adjacent to any enclosure edge. From this definition we can know that vertices $V_2$, $V_3$, $V_4$ in Figure 4.11 are the floating vertices, respectively on face $F_5$, $F_6$, $F_7$. During the fillet/round suppressing, a floating vertex will be converted into a construction vertex by a three-face intersection discussed in the following algorithm. In Figure 4.11, $V_4'$ is the construction vertex in the simplified B-Rep model that replaces $V_2$, $V_3$, $V_4$.

### 4.3.3.2 Algorithm of the Incremental Knitting Process

The incremental knitting process is a process that fills the gaps represented by empty rings and discs one at a time till a solid disc is formed. We define the gap filling as a knitting operation applied on empty rings or discs. Since ring-type chains and disc-type chains have different topology, the knitting operations will be different. For convenience, KR is defined as a knitting operation on a ring-type chain, and KD is defined as a knitting operation on a disc-type chain. Thus an incremental knitting process consists of a series of KR and KD operations. The detailed flow chart of the incremental knitting process is shown in Figure 4.12.

As shown in Figure 4.12, the ring-type chains are processed before the disc-type chains. One major reason is that the KR operation is simpler and more efficient than the KD operation, which can be seen from their algorithms discussed in the following. Another reason is that KR operation devised later will only consider the trace faces that
immediately enclose (or immediately adjacent to) the inside enclosure face for algorithmic efficiency. If the ring-type chain includes some trace faces that are not immediately adjacent to the inside enclosure face, they will not be processed during the KR operation. These unprocessed trace faces will be converted into disc-type chains that are processed by KD operations. As illustrated in Figure 4.10 and Figure 4.14, \{F_{24}\sim F_{35}\} forms a ring-type chain R_i, in which the faces \{F_{28}\sim F_{35}\} immediately enclose the inside enclosure face F_{15}, and the faces \{F_{24}\sim F_{27}\} are not immediately adjacent to the inside enclosure face F_{15}. Thus KR(R_i) will only process \{F_{28}\sim F_{35}\} and \{F_{24}\sim F_{27}\} are converted into four disc-type chains D_3\sim D_6. It is noticed that those unprocessed trace faces in the ring-type chain are like branches attached to the ring, thus this kind of ring-type chain is also called branching ring-type chain. The ring-type chain without branching trace faces is also called regular ring-type chain.

In the processing of ring-type chains, each time only one ring-type chain is identified and knitted by a KR operation. After each KR operation, the trace faces are updated and a new ring-type chain is identified and knitted till no ring-type chain exists in the updated trace faces. The branching ring-type chain is one reason for choosing (KR\rightarrow Update\rightarrow\ldots\rightarrow KR\rightarrow Update\rightarrow\ldots\rightarrow No Rings) process as explained above. Another reason is that there may exist multiple-ring chains in the original B-Rep model. As shown in Figure 4.13a, a rectangular boss on a base block is fully blended, i.e. all the edges of the boss are blended. From the H.E. analysis we can find that there are 6 rings (R_1, R_2, R_3, R_4, R_5, R_6) connected together. This multiple-ring chain can not be directly knitted by applying a KR operation. Thus we can choose one ring R_1 and all the other faces can be viewed as non-trace faces. After applying a KR operation on the ring-type chain R_1 and then updating the trace faces, only ring R_4 is left, i.e. the multiple-ring chain is converted into a single-ring chain. As we can see that the single-ring chain R_4 is a branching ring-type chain, it can be knitted by applying a KR operation first and then several KD operations. This multiple-ring chain can also be reduced into a single-ring chain by
Figure 4.12: The detailed flow chart of the incremental knitting process
Figure 4.13: Deduction of multiple rings into a single ring for a ring-type chain
Figure 4.14: The knitting process for the example part shown in Figure 4.10
removing $R_3$ first. As shown in Figure 4.13b, after removing $R_3$, only $R_6$ is left and it also forms a branching ring-type chain that can be knitted by applying a KR operation first and then several KD operations.

After finishing ring-type chain processing, the remaining trace faces must form disc-type chains. By checking the connection relationship among the unprocessed trace faces, all disc-type chains can be identified. Since each disc-type chain is isolated from one another, the KD operation can be applied without performing the trace face updating and the order of the series of KD operations for all the disc-type chains can be arbitrary. Once all the disc-type chains are knitted a complete simplified B-Rep model can be derived for automatic feature recognition. Figure 4.14 illustrates the knitting process for the example part shown in Figure 4.10. In this knitting process, the ring-type chain processing consists of 2 KR operations and the disc-type chain processing consists of 6 KD operations. After these operations a simplified B-Rep without fillets and rounds can be obtained from the original B-Rep model.

4.3.3.3 Algorithms of KR and KD Operations

KR and KD operations are applied to create new topological entities such as edges and vertices to knit the gap caused by chain removing. As mentioned before, we need to guarantee the geometric and topological consistency and reversibility of the simplified B-Rep model after fillet/round suppressing. The devised KR and KD operations can meet these requirements authentically, and they are described as follows. In order to facilitate the description of the algorithms for KR and KD operations, we will use five example parts pictured in Figure 4.15, each sub-figure of which represents a different topology of a trace face chain and its surrounding enclosure faces as follows:

- Figure 4.15a: a part with a branching ring-type chain surrounded by one inside and multiple outside enclosure faces;
- Figure 4.15b: a part with a branching ring-type chain surrounded by one outside and multiple inside enclosure faces;
- Figure 4.15c: a part with a regular ring-type chain surrounded by one inside and one outside enclosure faces;
- Figure 4.15d: a part with a regular ring-type chain surrounded by multiple inside and multiple outside enclosure faces;
- Figure 4.15e: a part with a disc-type chain surrounded by multiple enclosure faces.

It is observed that the above five topologies cover all the possible topological situations of trace faces in B-Rep models if multiple-ring chains have already been reduced into single-ring chains.

Algorithm: KR Operation

1. Input a ring-type chain $R$.
2. Find the outside enclosure faces $F_o$ and inside enclosure faces $F_i$ of $R$.
3. If $R$ is a branching ring-type chain, remove all the branching trace faces $F_b$ from $R$ to form a regular ring-type chain: $R - F_b \rightarrow R$; then update the enclosure faces by joining the removed branching trace faces as follows:
   (a) If the removed branching trace faces are outside the regular ring-type chain (Figure 4.15a), join them with outside enclosure faces: $F_o \cup F_b \rightarrow F_o$ (Figure 4.16a).
   (b) If the removed branching trace faces are inside the regular ring-type chain (Figure 4.15b), join them with inside enclosure faces: $F_i \cup F_b \rightarrow F_i$ (Figure 4.17a).
4. Calculate the face numbers Num of $F_o$ and $F_i$.
   (a) If Num($F_o$) > 1 and Num($F_i$) = 1 (Figure 4.16a), go to step 5;
   (b) If Num($F_o$) = 1 and Num($F_i$) > 1 (Figure 4.17a), switch the roles of $F_o$ and $F_i$ (Figure 4.17b), then go to step 5;
   (c) If Num($F_o$) > 1 and Num($F_i$) > 1 (Figure 4.15d), go to step 5;
   (d) If Num($F_o$) = 1 and Num($F_i$) = 1 (Figure 4.15c), go to step 7.
Figure 4.15: Five example parts (Note: gray-filled faces are enclosure faces of chains)
Figure 4.16: Illustration of a KR operation on the example part shown in Figure 4.15a
Figure 4.17: Branching→Regular conversion of $R$ and role switch between $F_o$ and $F_i$ for the example part shown in Figure 4.15b

Figure 4.18: Illustration of a KR operation on the example part shown in Figure 4.15c
Figure 4.19: Illustration of a KR operation on the example part shown in Figure 4.15d
5. Perform edge-face intersection to obtain construction vertices $V_c$:
   - For cases 4a and 4b:
     Advance each forward enclosure coedge on the outside enclosure faces to intersect the enclosure edge with the inside enclosure face and get the fixed end vertex of the enclosure edges, as shown in Figure 4.16b. These fixed end vertices form the construction vertices $V_c$.
   - For case 4c:
     Advance each forward enclosure coedge on the outside enclosure faces to intersect the enclosure edge with the inside enclosure faces and choose the nearest intersection point as the fixed end vertex of the enclosure edge; also advance each forward enclosure coedge on the inside enclosure faces to intersect the enclosure edge with the outside enclosure faces and choose the nearest intersection point as the fixed end vertex of the enclosure edge, as shown in Figure 4.19a. These fixed end vertices form the construction vertices $V_c$.

6. Perform face-face intersection to obtain construction edges $E_c$:
   (1) According to the topology of demarcation edge loops between the ring-type chain and the enclosure faces, connect the construction vertices to form a floating edge loop, as shown in Figure 4.16c or 4.19b. Each floating edge has two fixed end vertices, but the underlying geometry is not defined so far.
   (2) Intersect the outside enclosure faces with the inside enclosure faces to get the underlying geometry of the corresponding floating edge, as shown in Figure 4.16d or 4.19c. Thus the floating edges are converted into construction edges $E_c$.
   (3) Go to step 8.

7. Perform face-face intersection to obtain a construction edge and a construction vertex:
   (1) Intersect the outside enclosure face with the inside enclosure face to get a construction edge $E_c$ which forms a closed edge loop by itself, as shown in Figure 18a;
(2) Artificially add a construction vertex $V_c$ on the construction edge so that the closed edge has two coincident end vertices, which can meet the topological structure of a closed edge in B-Rep model, as shown in Figure 4.18b.

8. Clean all the faces, edges and vertices forming the ring-type chain from the B-Rep model and update the topology of the outside enclosure faces and inside enclosure faces by adding the construction vertices and construction edges, as shown in Figure 4.16e, 4.18c or 4.19d.

9. Extract the radius of the removed trace faces and attach a blending attribute to the corresponding construction edges, as shown in Figure 4.16f, 4.18d or 4.19e.

(a) For cylindrical faces, the radius is the one defining its base circle;

(b) For toroidal faces, the radius is the minor radius defining the circular cross-section since a torus is defined by a circular spine and a circular cross-section at each point on the spine.

(c) For spherical faces, the radius is the one defining the size of the sphere.

10. Return the updated B-Rep model after removing the ring-type chain $R$.

Algorithm: KD Operation

1. Input a disc-type chain $D$.

2. Find the enclosure faces $F_e$ of $D$.

3. Perform edge-face intersection to obtain construction vertices $V_{ct}$:

   (1) Advance each forward enclosure coedge on the enclosure faces to intersect the enclosure edge with non-adjacent enclosure faces and choose the nearest intersection point as the fixed end vertex of the enclosure edge, as shown in Figure 4.21a. These fixed end vertices form the construction vertices $V_{ct}$. Please note that the edge-face adjacency defined here is different from that defined in B-Rep model. Figure 4.20 illustrates a local topology of an edge $E_1$. Based on the definition used in B-Rep model, $F_1$, $F_2$, $F_3$, $F_4$ and $F_5$ are the adjacent faces of $E_1$. Our definition used in this step will take $F_1$, $F_2$, $F_3$, $F_4$ and $F_5$ as adjacent faces of $E_1$. 

81
(2) Check the duplication of construction vertices.

Merge the coincident construction vertices and update the fixed end vertex of each associated enclosure edge. Figure 4.20 Illustration of edge-face adjacency

4. Perform three-face intersection for all the floating vertices to obtain construction vertices $V_{c2}$:
   (1) Check if any floating vertex exists on an enclosure face. For each floating vertex, search its three neighbor enclosure faces and perform three-face intersection to get a construction vertex, as shown in Figure 4.21b.
   (2) Check the duplication of the construction vertices and merge the coincident construction vertices.

5. Combine $V_{c1}$ and $V_{c2}$ to form the set of all the construction vertices $V_c$.

6. Perform face-face intersection to obtain construction edges $E_c$:
   (1) According to the topology of demarcation edge loops between the ring-type chain
Figure 4.21: Illustration of a KD operation on the example part shown in Figure 4.15e
and the enclosure faces, connect the construction vertices to form a floating edge loop, as shown in Figure 4.21c. Each floating edge has two fixed end vertices, but underlying geometry is not defined so far.

(2) Check the duplication of floating edges.

If two floating edges have the same end vertices and adjacent faces, they are merged.

(3) Intersect the two adjacent enclosure faces of each floating edge to obtain the underlying geometry for the floating edge, as shown in Figure 4.21d. Thus the floating edges are converted into construction edges $E_c$.

7. Clean all the faces, edges and vertices forming the disc-type chain from the B-Rep model and update the topology of the enclosure faces by adding the construction vertices and construction edges, as shown in Figure 4.21e.

8. Extract the radius $r$ of the removed trace faces and attach a blending attribute to the corresponding construction edges, as shown in Figure 4.21f.

(a) For cylindrical faces, the radius is the one defining its base circle;

(b) For toroidal faces, the radius is the minor radius defining the circular cross-section since a torus is defined by a circular spine and a circular cross-section at each point on the spine.

(c) For spherical faces, the radius is the one defining the size of the sphere.

9. Return the updated B-Rep model after removing the disc-type chain $D$.

4.4 Examples

Two test parts with various fillets and rounds are chosen here to verify the proposed approach.

Test Part 1: NEW_DEMO_US

NEW_DEMO_US is obtained from NIST (National Institute of Standards and Technology) part library (http://www.nist.gov/parts). It is a very popular test part in the feature recognition community, and it is a typical part machined by a 3-axis flat-head
milling machine. Thus all the fillets are generated around edge corners discretely. 19 cylindrical faces are identified as trace faces, and each one forms a disc-type chain, as shown in Figure 4.22a. Therefore, 19 KD operations are applied in the incremental knitting process to suppress the fillets for B-Rep model simplification, as shown in Figure 4.22b.

![Figure 4.22: Test part 1: NEW_DEMO_US](image)

**Test Part 2: The Example Part Shown in Figure 4.4**

The example part shown in Figure 4.4 has blendings on straight edges, circular edges and vertices. Thus the identified trace faces include cylindrical, toroidal and spherical faces, which form 1 regular ring-type chain $R_1$, 1 branching ring-type chain $R_2$ and 2 disc-type chains $D_1$ and $D_2$, as shown in Figure 4.23a. A KR operation is applied on $R_1$. Another
KR operation is applied on $R_2$. During the KR operation on $R_2$, $R_2$ is first converted into a regular ring-type chain and then knitted. The conversion will generate 4 disc-type chains. Thus 6 KD operations are applied on the 4 generated disc-type chains plus $D_1$ and $D_2$. Totally 2 KR operations and 6 KD operations are employed in the incremental knitting process to suppress fillets and rounds for B-Rep model simplification, as shown in Figure 4.23b.

![Diagram](image)

**Figure 4.23**: Test part 2: the example part shown in Figure 4.4

### 4.5 Summary

In this paper, we have presented an approach to simplifying B-Rep models by automatic fillet/round suppressing. The proposed approach utilizes an incremental knitting process...
to handle various topological structures of fillets and rounds such as ring-type chain and disc-type chain so as to achieve generality and robustness without loss of algorithmic efficiency.

As mentioned before, the geometric domain of the fillets and rounds that we handle in this research work covers constant-radius fillets and rounds. Such fillets and rounds meet the requirements of most blendings encountered in mechanical part design. In some mechanical parts, variable radius blendings are introduced to meet the special manufacturing and assembly requirements. The variable radius blending starts at one end of the edge being blended with a radius and tapers or expends to a different radius at the other end. Therefore, the domain of the trace faces for fillet/round identification needs to be extended when variable radius blendings are introduced. As long as trace faces can be identified, the knitting process can be applied to simplify the B-Rep model.
CHAPTER 5

FEATURE RECOGNITION PHASE I:

BASIC SURFACE FEATURE RECOGNITION

The first step of feature recognition is to identify the surface boundary of the form features, also called surface features, which provides the hints for volumetric feature extraction. In order to identify the surface boundary of features, a general basic surface feature recognition module for automatic feature recognition is proposed in this chapter. The necessary theoretical foundation and related issues for the module development are presented.

5.1 Introduction

[Lee and Menq, 1995] was the first to propose the curvature computation method for primitive feature recognition. Later [Sonthi et al, 1997] presented a very similar method, called curvature region. The topological entity classification in both methods is incomplete and problematic. For instance, both can not identify the entities shared by multiple features due to feature interactions. Thus the recognized surface boundary of the feature may be insufficient. In addition, the sub-graph isomorphism used by [Sonthi et al, 1997] to find the feature definition obviously can not handle the topological variants caused by the edge blends. Based on the groundwork of [Lee and Menq, 1995], this research introduces the definition of bi-attribute and a comprehensive classification
of bi-attribute for all the boundary entities. Basic surface feature is recognized by entity propagation starting from seed entity in terms of bi-attribute. Shared faces are identified from the bi-attribute pattern of the outer edge loop of the face.

5.2 Mathematical & Theoretical Foundation

Part models with similar shapes may contain different geometric and topologic entities. For example, a cylindrical protrusion and a cylindrical depression are created on a rectangular block as shown in Figure 5.1a. If all the sharp edges of the protrusion and depression are replaced by fillets and rounds as shown in Figure 5.1b, the major global shapes of the protrusion and depression do not change but the geometric and topological entities are different due to the local edge blending. In order to recognize the major forms and local shapes from a part model in a consistent and general way, underlying invariant properties need be investigated to facilitate the characterization of various geometric and topological entities.

![Figure 5.1: Cylindrical protrusion & depression on a block](image)

(a) without edge blendings; (b) with edge blendings
In this section several invariant surface properties derived from differential geometry and discrete topology as well as their applications to the geometric reasoning for basic surface feature recognition are investigated. The concepts of homeomorphism and global Gauss-Bonnet theorem [Carmo, 1976] are introduced and their applications to the derivation of bi-attribute, a proposed common attribute which can characterize all the basic topological entities such as faces, edges and vertices, are developed as follows.

5.2.1 Homeomorphism

In topology, a topological space \((X, T)\) is defined as a set \(X\) for which a topology \(T\) has been specified. The concept of homeomorphism is defined as follows.

Definition: Let \((X, T), (Y, S)\) be two topological spaces and let \(h: X \rightarrow Y\) be bijective. Then \(h\) is a homeomorphism iff \(h\) is continuous and \(h^{-1}\) is continuous. If such a map exists, \((X, T)\) and \((Y, S)\) are called homeomorphic.

In other words, a homeomorphism, also known as topological invariance, is a continuous, one-one, and onto mapping between two topological spaces that has an inverse, and which is also continuous. Intuitively homeomorphism can be thought of as elastic deformation that preserves adjacency properties. For solid objects such as baseballs, donut, cups, etc., their shapes can be represented by the surfaces enclosing the objects. Ordinarily, such surfaces are compact, i.e. closed and bounded. Two surfaces are considered to be homeomorphic if one of the surfaces can be continuously distorted to look like the other. Continuous distortion can be bending, stretching, and squashing without tearing or gluing points together. According to these criteria, the surfaces of baseballs, chalk, and bolts are homeomorphic. Similarly, the surfaces of nuts, the teacups with one handle, and the torus are homeomorphic also.
5.2.2 Global Gauss-Bonnet Theorem

For any compact orientable surface, a triangulation can be performed to divide the surface into a number of triangular patches. Let \( F, E, V \) denote the number of triangles, edges, and vertices, respectively on the surface after triangulation, the number
\[
\chi = V - E + F = 2 - 2g \quad (5.1)
\]
is called the Euler characteristic of the triangulation [Carmo, 1976], also known as the Euler-Poincare characteristic. In equation (5.1), \( g \) stands for genus, representing the number of handles added to a sphere. This shows that \( \chi \) is a topological invariant of compact orientable surfaces in \( \mathbb{E}^3 \).

Given a triangulation of a compact orientable surface \( R \), and let \( C_1, \ldots, C_n \) be the closed, simple, piecewise regular curves which form the boundary of each triangle and \( \theta_1, \ldots, \theta_p \) be the set of external angles of the curves \( C_1, \ldots, C_n \), the global Gauss-Bonnet theorem can be expressed as
\[
\int_{\partial R} Kds + \sum_{i=1}^{n} \int_{C_i} K_g(s)ds + \sum_{i=1}^{p} \theta_i = 2\pi\chi(R) \quad (5.2)
\]
where \( s \) denotes the arc length of \( C_i \), \( K_g \) is the geodesic curvature of \( C_i \), \( K \) is the Gaussian curvature of \( R \), \( n \) is the number of piecewise regular curves, and \( p \) is the number of vertices after triangulation [Carmo, 1976]. The global Gauss-Bonnet theorem is important since it provides a relationship between the Euler characteristic that is defined in terms of topology and the total curvature that is defined in terms of differential geometry.

5.2.3 Curvature Distribution

From the equation (5.1), we can know that a sphere and a rectangular block have the same genus, i.e. \( g = 0 \) because both of them are solids without handles. Therefore they have the identical Euler characteristic, i.e. \( \chi = 2 - 2g = 2 \), and the total curvatures are both \( 4\pi \)'s since they are homeomorphic to each other. However, the distributions of the two curvature functions are very different. The Gaussian curvature of a sphere is a constant
and is equal to the inverse square of the radius. On the other hand, the total curvature in the planar faces and straight edges of a rectangular block is zero and it is concentrated at the vertices.

If a protrusion is made on the top face of a rectangular block (as the cylindrical protrusion shown in Figure 5.1a), the total curvature remains invariant since this deformation does not affect the Euler characteristic. It can be observed that even though the protrusion has positive total curvature, the intersecting edge between the protrusion and the top face of the block has concentrated negative total curvature that offsets the positive total curvature generated by the protrusion. If the sharp edge between the protrusion and the face is replaced by a smooth fillet as shown in Figure 5.1b, the total curvature still remains invariant. The negative total curvature to offset the positive total curvature of the protrusion is distributed on the fillet and can be calculated by $\int_{r}K\sigma$. This shows that once a feature having positive total curvature is created on an object, there must exist a form that could be faces, edges, or vertices and has an opposite sign of total curvature to offset it.

From the above discussion, we can infer that adding one form feature to an existing object will lead to variations of curvature distributions. Curvature distributions can be characterized as feature properties and can be applied to surface feature recognition. A further investigation in the classification of basic topological entities of B-Rep models for feature recognition is performed in Section 5.3.

5.2.4 Bi-attribute

Attribute is defined by [Subrahmanyan, DeVries and Pratt, 1995] as a characteristic quality or property that associates meaning to an entity, significant to a particular stage in the life cycle of a product. Attribute emphasizes on the entity itself. Here entity has more general scope. It includes the low-level topological entities such as faces, edges and vertices, and the high-level features. Attributes can be geometric, topological, functional
and technological. Bi-attribute defined in this section is a typical geometric and topological attribute that plays an important role in the automatic recognition of basic surface features.

Figure 5.2: Three groups of two different features with same total curvature

Figure 5.2 shows three groups of two different features on the top face of a rectangular block. Applying the global Gauss-Bonnet theorem, we can find that all these features have positive total curvature and the boundaries intersecting with the planar face have negative total curvatures. The curvature distribution among the three terms in equation (5.2) for all the six features is listed in Table 5.1. From this figure, we can know that there is only one non-zero term in equation (5.2) for each feature in Figure 5.2. In other words, the non-zero total curvature can be obtained by faces such as features shown in Figure 5.2a, edges such as features in Figure 5.2b, or vertices such as features in Figure 5.2c. However, features 1, 3, and 5, added on the base surface, and the other features 2, 4, and 6, subtracted from the base surface, cannot be distinguished based on the sign of the total curvature. Therefore, additional criteria are needed to identify the differences between these two classes of features.
<table>
<thead>
<tr>
<th>Feature</th>
<th>$\int K d\sigma$</th>
<th>$\sum_{i=1}^{n} \int_{s_i} \kappa_k (s) ds$</th>
<th>$\sum_{i=1}^{n} \theta_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Total face curvature term</td>
<td>total edge curvature term</td>
<td>total vertex curvature term</td>
</tr>
<tr>
<td>1</td>
<td>&gt; 0</td>
<td>= 0</td>
<td>= 0</td>
</tr>
<tr>
<td>2</td>
<td>&gt; 0</td>
<td>= 0</td>
<td>= 0</td>
</tr>
<tr>
<td>3</td>
<td>= 0</td>
<td>&gt; 0</td>
<td>= 0</td>
</tr>
<tr>
<td>4</td>
<td>= 0</td>
<td>&gt; 0</td>
<td>= 0</td>
</tr>
<tr>
<td>5</td>
<td>= 0</td>
<td>= 0</td>
<td>&gt; 0</td>
</tr>
<tr>
<td>6</td>
<td>= 0</td>
<td>= 0</td>
<td>&gt; 0</td>
</tr>
</tbody>
</table>

Table 5.1: Curvature distribution of the six features shown in Figure 5.2

For those cases where the total curvature can be evaluated by surface curvatures, i.e., $\int K d\sigma$, the properties of Gaussian curvature $K$ can be applied to classify these surfaces. We know that Gaussian curvature is the multiplication of the two principal curvature $k_1$ and $k_2$, i.e., $K = k_1 k_2$. In addition, it is known that faces with positive Gaussian curvature ($K > 0$) can be classified into either concave or convex faces. Thus the sign of the two principal curvatures will determine the convexity of the face with $K > 0$. The face that has two negative principal curvatures is a convex face while the face that has two positive principal curvatures is a concave face. Faces with negative Gaussian curvature ($K < 0$) must contain mixed concave and convex properties since the two principal curvatures have different signs, i.e., one is positive and the other negative. These classifications result in three basic categories of form features: the positive feature referring to convex faces, the negative feature referring to concave faces, and the transition feature referring to mixed concave and convex faces, usually called transitive faces. We can find that transition features that consist of transitive faces always correspond to the fillets and rounds due to the local edge blending.
According to the above criteria, feature 1 in Figure 5.2 represents a positive feature because of convex face property while feature 2 represents a negative feature because of concave face property. The features 3, 4, 5, and 6 in Figure 5.2 still cannot be evaluated by Gaussian curvature since the Gaussian curvature is zero and total curvature is concentrated at face discontinuities such as edges and vertices. Nevertheless, the concept of concave and convex properties can be extended to characterize edges and vertices. Thus we employ the concept of concave and convex properties in describing a continuous curve to describe a curve containing curvature discontinuities. The “character edge” (defined in Section 5.3.2) is proposed to describe convexity discontinuities in a feature. The use of two attributes is similar to that of the two principal curvatures that are used to characterize a point on a surface. These two attributes will describe the two extreme curvature properties by using properties of concave, convex, and straight instead of numerical values like principal curvatures. Similar to the character edge, a vertex defined as the intersection of character edges can also have two attributes. The attributes of a vertex can be determined similar to a point on a character edge. Thereby, we introduce the definition of bi-attribute, BAtt in short, to represent the two typical attributes of various topological entities.

Definition: Let E be a topological entity, then the bi-attribute of E is a two-tuple representation \( BAtt(E) = (b_1, b_2) \), where \( b_1 \) and \( b_2 \) are the two typical attributes of topological entities.

For example, the two typical attributes of faces can be derived from the sign of their two principal curvatures. The detailed classification of faces, edges and vertices in bi-attributes is explained in the next section.
5.3 Classification of Topological Entities of B-Rep Models

In this section, the topological entities including faces, edges, and vertices, are considered since they are the basic constructive entities that form the B-Rep model of a part. Bi-attribute is used to classify these three topological entities as follows.

5.3.1 Classification of Faces in Bi-attribute

As mentioned in Section 5.2.4, faces can be characterized by using signs of the two principal curvature $k_1$ and $k_2$. Thus, the proposed bi-attributes are specified by the signs of the two principal curvatures to characterize face $F$ property as follows.

$$BAtt(F) = (-\text{sgn}(k_1), -\text{sgn}(k_2))$$

Thus the value set of the two components of the face’s bi-attribute is $\{-1,0,1\}$.

<table>
<thead>
<tr>
<th>case</th>
<th>$k_1$</th>
<th>$k_2$</th>
<th>$K$</th>
<th>$M$</th>
<th>Bi-attribute</th>
<th>Convexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$&lt;0$</td>
<td>$&lt;0$</td>
<td>$&gt;0$</td>
<td>$&lt;0$</td>
<td>$BAtt=(1,1)$</td>
<td>Convex</td>
</tr>
<tr>
<td>2</td>
<td>$=0$</td>
<td>$&lt;0$</td>
<td>$=0$</td>
<td>$&gt;0$</td>
<td>$BAtt=(0,1)$</td>
<td>Concave</td>
</tr>
<tr>
<td></td>
<td>or</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$&lt;0$</td>
<td>$=0$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>$&gt;0$</td>
<td>$&gt;0$</td>
<td>$&gt;0$</td>
<td>$&gt;0$</td>
<td>$BAtt=(-1,-1)$</td>
<td>Transitive</td>
</tr>
<tr>
<td>4</td>
<td>$&gt;0$</td>
<td>$=0$</td>
<td>$=0$</td>
<td>$&gt;0$</td>
<td>$BAtt=(-1,0)$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>or</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$=0$</td>
<td>$&gt;0$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>$&gt;0$</td>
<td>$&lt;0$</td>
<td>$&lt;0$</td>
<td>Any</td>
<td>$BAtt=(1,-1)$</td>
<td>Planar</td>
</tr>
<tr>
<td></td>
<td>or</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$&lt;0$</td>
<td>$&gt;0$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>$=0$</td>
<td>$=0$</td>
<td>$=0$</td>
<td>$=0$</td>
<td>$BAtt=(0,0)$</td>
<td></td>
</tr>
</tbody>
</table>

(1: convex attribute; 0: straight attribute; -1: concave attribute)

Table 5.2: Classification of faces in bi-attribute
If the principal curvature is less than zero, the attribute is 1 and defined as convex. If the principal curvature is greater than zero, the attribute is -1 and defined as concave. Otherwise, the attribute is 0 and defined as straight when the principal curvature is zero.

Mean curvature \( m \) is another important property of surfaces, and it is the average of the two principal curvatures. It can also be used to classify faces by coupling with the Gaussian curvature \( K \). Hence by either using signs of the two principal curvatures \( k_1 \) and \( k_2 \) or using the signs of Gaussian curvature \( K \) and mean curvature \( m \), six cases of faces can be categorized into four types including convex faces, concave faces, transitive faces and planar faces as shown in Table 5.2. The planar faces do not belong to concave or convex categories. But it will be included in either positive or negative features as explained in Section 5.4.

5.3.2 Classification of Edges in Bi-attribute

Edges with \( C^0 \) or \( C^1 \) continuities have curvature discontinuities and thus the total curvature cannot be evaluated by Gaussian curvature. However, from the global Gauss-Bonnet theorem in equation (5.2), the total curvature can still be evaluated by geodesic curvature along the edge. By applying the concept of homeomorphism, the geometric form of an edge can be altered to a surface such as a fillet and maintains the total curvature. This implies that we may also define bi-attribute to characterize the total curvature for an edge similar to the bi-attribute describing curvature properties of a point on a continuous surface. Thus, the definition of the bi-attribute of edges can be derived as follows.

From equation (5.2), the sign of total curvature from a curve can be determined by the sign of geodesic curvature \( K_g \). Since the geodesic curvature can be calculated from the curve and surface normal at the selected point, the sign of total curvature can then be determined. Similar to the classification of a face, the bi-attribute for an edge with negative total curvature will be concave and convex. For an edge with positive total curvature, the bi-attribute will be concave and concave or convex and convex which need to be determined by additional criteria. As mentioned earlier, positive total curvature at a
selected point will have all concave or convex curves passing the point. Thus any one curve can be used to identify the type of the bi-attribute for the selected point on an edge.

**Computation of the First Attribute of an Edge**

Given an edge E and its two neighbor faces F₁ and F₂, and let F₁ be the edge’s owner face and F₂ be the edge’s partner face [Spatial, 1998a], the retrieval operations of the edge’s owner face and partner face can be performed in ACIS as follows.

```c
EDGE *E;
FACE *F1, *F2;
F1 = ((LOOP *)E->coedge()->owner())->face();
F2 = ((LOOP *)E->coedge()->partner()->owner())->face();
```

Let \( n₁ \) and \( n₂ \) be the normals of the tangent planes \( P₁ \) and \( P₂ \) of the neighbor faces \( F₁ \) and \( F₂ \) at a point \( P \) (usually take the parametrically middle point) on the edge, and let \( v \) be the tangent vector of the edge at the point along the edge direction (Figure 5.3), then the first attribute \( b₁ \) of the edge’s bi-attribute \( BAtt(E) \) is computed as follows.

\[
b₁ = \begin{cases} 
0 & \text{(smooth)} \\
1 & \text{(convex)} \\
-1 & \text{(concave)}
\end{cases} \quad \text{if} \quad \begin{cases} 
(n₁ \times n₂) \cdot v = 0 \\
(n₁ \times n₂) \cdot v > 0 \\
(n₁ \times n₂) \cdot v < 0
\end{cases}
\]

![Figure 5.3: Computation of the first attribute of edges](image-url)
From the morphological point of view, edges can be classified into smooth edges and sharp edges. According to the above first attribute computation, the definition of smooth edges and sharps is introduced as follows.

Definition: A smooth edge is an edge with zero first attribute or an edge when the tangent planes of the edge's two neighbor faces at any point on the edge are coplanar, i.e. having common tangent planes.

Definition: A sharp edge is an edge with non-zero first attribute or an edge when the tangent planes of the edge's two neighbor faces are not coplanar.

\begin{figure}
\centering
\includegraphics[width=0.5\textwidth]{sharp_edges_diagram.png}
\caption{Illustration of all the cases of bi-attribute for sharp edges}
\end{figure}

Note: The bi-attributes of all the unmarked edges are (1, 0).

Computation of the Second Attribute of a Sharp Edge

As discussed above, the sign of geodesic curvature $K_g$ of a curve determines the sign of the total curvature of the curve. Thus for sharp edges it is straightforward to compute the
second attribute according to the sign of the geodesic curvature and the known non-zero first attribute as follows.

\[ b_2 = \text{sgn}(K) \times b_1 \]

The computation for all the cases of sharp edges is shown in Table 5.3. An example part is shown in Figure 5.4 to illustrate all the cases of bi-attribute for sharp edges.

<table>
<thead>
<tr>
<th>( K_\parallel )</th>
<th>( \text{sgn}(K_\parallel) )</th>
<th>( b_1 )</th>
<th>( b_2 )</th>
<th>Bi-attribute</th>
<th>Convexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>&gt; 0</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>( \text{BiAtt}=(-1,-1) )</td>
<td>Concave</td>
</tr>
<tr>
<td>&lt; 0</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
<td>( \text{BiAtt}=(-1,1) )</td>
<td>Character</td>
</tr>
<tr>
<td>= 0</td>
<td>0</td>
<td>-1</td>
<td>0</td>
<td>( \text{BiAtt}=(-1,0) )</td>
<td>Concave</td>
</tr>
</tbody>
</table>

Table 5.3: Computation of the second attribute of sharp edges

**Computation of the Second Attribute of a Smooth Edge**

Smooth edges are very common in the models with fillets and rounds, which are created in the modeling considering the requirements from machining and assembly as discussed in Chapter 4. Smooth edges usually constitute part of the boundaries of fillets and rounds so that they play an important role in identifying the faces forming these auxiliary features. From the underlying geometry, smooth edges can be linear or non-linear.

The first attribute of all the smooth edges are set to 0 from the above computation. The computation of the second attribute of a smooth edge is different from that of a sharp edge because the first attribute of a smooth edge is zero and we can not decide the value of the second attribute just by using the geodesic curvature. By adopting the similar
classification of smooth straight edges proposed by [Kyprianou, 1980], the second attribute of smooth linear and non-linear edges can be decided by the convexity of their two neighbor faces since the objects we consider are two-manifold objects. The classification detail is shown in Table 5.4.

Two example parts are shown in Figure 5.5 to illustrate all the above cases of bi-attribute for smooth edges except case L6. The case L6 usually does not occur in the B-Rep model of real-world parts because if a smooth linear edge has two planar neighbor faces, it will be merged into either of the neighbor faces and the two coplanar neighbor faces will be merged into one planar face to form a regular two-manifold object.

<table>
<thead>
<tr>
<th>Case</th>
<th>Convexity of F₁</th>
<th>Convexity of F₂</th>
<th>Bi-attribute BAtt</th>
<th>Convexity</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Smooth Linear Edge</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L₁</td>
<td>Planar</td>
<td>Convex</td>
<td>BAtt=(0,1)</td>
<td>(Smooth) Convex</td>
</tr>
<tr>
<td>L₂</td>
<td>Convex</td>
<td>Convex</td>
<td>BAtt=(0,1)</td>
<td>(Smooth) Convex</td>
</tr>
<tr>
<td>L₃</td>
<td>Planar</td>
<td>Concave</td>
<td>BAtt=(0,-1)</td>
<td>(Smooth) Concave</td>
</tr>
<tr>
<td>L₄</td>
<td>Concave</td>
<td>Concave</td>
<td>BAtt=(0,-1)</td>
<td>(Smooth) Concave</td>
</tr>
<tr>
<td>L₅</td>
<td>Convex</td>
<td>Concave</td>
<td>BAtt=(0,-1)</td>
<td>(Smooth) Concave</td>
</tr>
<tr>
<td>L₆</td>
<td>Planar</td>
<td>Planar</td>
<td>BAtt=(0,0)</td>
<td>(Smooth) Straight</td>
</tr>
<tr>
<td><strong>Smooth Non-linear Edge</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NL₁</td>
<td>Planar</td>
<td>Convex</td>
<td>BAtt=(0,1)</td>
<td>(Smooth) Convex</td>
</tr>
<tr>
<td>NL₂</td>
<td>Convex</td>
<td>Convex</td>
<td>BAtt=(0,1)</td>
<td>(Smooth) Convex</td>
</tr>
<tr>
<td>NL₃</td>
<td>Planar</td>
<td>Concave</td>
<td>BAtt=(0,-1)</td>
<td>(Smooth) Concave</td>
</tr>
<tr>
<td>NL₄</td>
<td>Concave</td>
<td>Concave</td>
<td>BAtt=(0,-1)</td>
<td>(Smooth) Concave</td>
</tr>
<tr>
<td>NL₅</td>
<td>Convex</td>
<td>Transitive</td>
<td>BAtt=(0,1)</td>
<td>(Smooth) Convex</td>
</tr>
<tr>
<td>NL₆</td>
<td>Planar</td>
<td>Transitive</td>
<td>BAtt=(0,0)</td>
<td>(Smooth) Transitive</td>
</tr>
<tr>
<td>NL₇</td>
<td>Concave</td>
<td>Transitive</td>
<td>BAtt=(0,-1)</td>
<td>(Smooth) Concave</td>
</tr>
</tbody>
</table>

Note: (1) F₁ and F₂ are the two neighbor faces of the edge;  
(2) L₅ can be (0,-1) or (0,1). (0,-1) is chose here;  
(3) L₆ will not exist after the B-Rep regularization;  
(4) NL₆ is set to (0,0) for simplicity.

Table 5.4: Classification of smooth edges in bi-attribute
Classification of Edges in Bi-attribute

Whether the edges are smooth edges or sharp edges, generally they can be classified into three types: convex edge, concave edge and character edge, and their definitions are as follows.

Definition: A convex edge is an edge with the bi-attribute of which two components are both \( \geq 0 \) and at least one is 1.

Definition: A concave edge is an edge with the bi-attribute of which two components are both \( \leq 0 \) and at least one is -1.

Definition: A character edge is an edge with the bi-attribute \((-1, 1)\) or \((1, -1)\).
Character edges usually represent the convexity discontinuities in a B-Rep model. Character edges with bi-attribute (-1, 1) or (1, -1) indicate the sharp discontinuities.

<table>
<thead>
<tr>
<th>Case</th>
<th>Number of Incident edges</th>
<th>Convexity of incident edges</th>
<th>Bi-attribute</th>
<th>Convexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1(closed) or 2</td>
<td>Smooth</td>
<td>(0,0)</td>
<td>Smooth</td>
</tr>
<tr>
<td>2</td>
<td>≥3</td>
<td>Convex</td>
<td>(1,1)</td>
<td>Convex</td>
</tr>
<tr>
<td>3</td>
<td>≥3</td>
<td>Concave</td>
<td>(-1,-1)</td>
<td>Concave</td>
</tr>
<tr>
<td>4</td>
<td>≥3</td>
<td>Concave + Convex or ≥1 Character</td>
<td>(-1,1)</td>
<td>Character</td>
</tr>
</tbody>
</table>

Table 5.5: Classification of vertices in bi-attribute

5.3.3 Classification of Vertices in Bi-attribute

Similar to the edge, a vertex, which is defined as the intersection of edges, can also be classified by bi-attribute by applying the concept of homeomorphism. Ideally, bi-attributes of a vertex should be determined based on all the curves passing at the vertex on the adjacent surfaces. However, there is an easier way to determine the bi-attribute by using all the edges incident at the vertex in a B-Rep model. The bi-attribute of a vertex can be determined from the bi-attributes of all the incident edges based on the following principles: (1) if all the incident edges at the vertex are concave ones, the two components of the bi-attribute of the vertex will both be concave as (-1, -1) and the vertex is defined as a concave vertex; (2) similarly if all the incident edges at the vertex are convex ones, the two components of the bi-attribute of the vertex will both be convex as (1,1) and the vertex is defined as a convex vertex; (3) if the incident edges at the vertex have both concave and convex ones, the vertex will have a convex and a concave
attribute as (-1,1) and is defined as character vertex. As long as there is at least one character edge existing among all the incident edges at the vertex, the bi-attribute of the vertex is defined as (-1,1) because the character edges represent the convexity discontinuities. In conclusion, the vertex with bi-attribute (-1,1) is defined as a character vertex to represent the convexity discontinuity. Table 5.5 lists the all the cases in the classification of vertices in bi-attribute.

5.4 Basic Surface Feature Recognition

The proposed bi-attribute can be used to characterize basic topological entities such as faces, edges and vertices. Thus the topological entities with similar bi-attribute can be grouped to form a higher-level geometric form, which is called basic surface feature in this research. According to the bi-attribute classification, basic surface features such as positive features and negative features can be extracted from the topological entities characterized by the bi-attribute and they can be applied further to extract volumetric features. Since the recognition of basic surface features only focuses on the clustering of boundary topological entities locally, the feature interaction problem, which has to be solved globally, is left to the work on the volumetric feature extraction (Chapter 6).

5.4.1 Methodology

The positive and negative forms in the basic surface features provide not only the essential information for the forms and shape of a part model but also the links to feature extraction for various manufacturing processes.

In order to recognize all the positive and negative features from the B-Rep model of any real-world mechanical part, a systematic and general methodology is proposed. The major aim of the methodology is to establish a comprehensive way to cluster topological entities (faces, edges and vertices) based on entity propagation in bi-attribute. As discussed earlier, the convexity of faces, edges and vertices can be categorized based on the bi-attribute computation. Thus positive forms can be identified by clustering convex
topological entities, and negative forms can be identified by grouping concave topological entities. Transitive faces, character edges and character vertices represent convexity discontinuities that separate positive and negative clusterings.

Figure 5.6: The functional diagram of the module of basic surface feature recognition

The framework of the proposed basic surface feature recognition consists of five functional modules (Figure 5.6) including geometric and topological preparation, classification of topological entities (faces, edges and vertices) in bi-attribute, classification of seed entities, characterization & identification of feature loops, recognition of basic surface features. The geometric and topological preparation module creates the adjacent entity lists for all the primitive topological elements including faces, edges and vertices. The generated entity adjacency information is used in the feature loop identification module to reduce the effort of retrieving adjacent entities.
5.4.2 Seed Entity

The concept of seed entity is introduced to facilitate the entity propagation. Seed entity is a topological entity from which a depth-first searching algorithm starts for the entity clustering. Based on the categorized convexity of faces, edges and vertices, three types of seed entities can be classified as shown in Table 5.6. Each type of seed entity will lead to the corresponding type of surface features. Type I seed entities have the highest priority in the entity propagation. Therefore, the entity propagation will first identify type I surface features, followed by type II surface features and then type III surface features.

<table>
<thead>
<tr>
<th>Type I Seed Entity</th>
<th>Convex</th>
<th>Concave</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1,1) or (1,0) face</td>
<td>(-1,-1) or (-1,0) face</td>
<td></td>
</tr>
<tr>
<td>(1,1) edge</td>
<td>(-1,-1) edge</td>
<td></td>
</tr>
<tr>
<td>(1,1) vertex</td>
<td>(-1,-1) vertex</td>
<td></td>
</tr>
<tr>
<td>Type II Seed Entity</td>
<td>(1,0) edge with two (-1,1) end vertices</td>
<td>(-1,0) edge with two (-1,1) end vertices</td>
</tr>
<tr>
<td>Type III Seed Entity</td>
<td>(0,0) face with all convex edges</td>
<td></td>
</tr>
</tbody>
</table>

Table 5.6: Classification of seed entities for entity propagation

Type I and type II seed entities are further classified into convex ones and concave ones based on the convexity property. Type II concave seed entity is used for the identification of step-shaped features and type II convex seed entity I is used for the identification of rib-shaped features. Type III seed entity is a special seed entity used to identify limbo faces [Marefat and Kashyap, 1990]. Limbo feature face is defined as a face in a depression without concave edges. In graph-based methods [Joshi and Chang, 1988], limbo faces are removed from the part graph so as to create separated feature graphs. In
fact, limbo faces indicate the existence of negative feature and positive feature at the same location. Thus limbo faces should result in the recognition of both positive and negative features. Limbo faces are defined as type III seed entity during the negative surface feature recognition.

5.4.3 Characterization and Identification of Feature Loops

The feature loop is defined as the connected topological entities with common attributes. Despite its name, a feature loop is not restricted to comprising only edges and vertices. It may comprise adjacent entities of vertices, edges, or faces. A single face can be defined as a feature loop, for example a simple through hole consists of only one cylindrical face. A depth-first searching technique is applied to identify forms by searching the adjacent topological elements. Five categories of feature loops can be identified as follows.

- **Type I Positive Feature Loop (PFL-I):** Starting with any type I convex seed entity, connect the adjacent face, edge, or vertex with bi-attributes including (1,1), (1,0), (0,1), or (0,0).
- **Type I Negative Feature Loop (NFL-I):** Starting with any type I concave seed entity, connect the adjacent face, edge, or vertex with bi-attributes including (-1,-1), (-1,0), (0,-1), or (0,0).
- **Type II Positive Feature Loop (PFL-II):** Starting with any type II convex seed entity, connect its two adjacent faces;
- **Type II Negative Feature Loop (NFL-II):** Starting with any type II concave seed entity, connect its two adjacent faces;
- **Type III Negative Feature Loop (NFL-III):** Starting with any type III concave seed entity,

The adjacent relationship is defined based on the connectivity among basic topological elements in B-Rep models. By applying the depth-first searching technique, the positive and negative feature loops can be identified. Since the bi-attribute of a planar face is
(0,0), it can join either positive feature loop or negative feature loop depending on the configurations of the associated feature loops of all the edges in its outer edge loop. The planar face will join all the feature loops that the edges in its outer edge loop are associated with. Therefore, the planar face can be shared by multiple surface features.

5.4.4 Recognition of Basic Surface Features

By studying the formation of features existing in the mechanical parts, we concluded the following presence rules for surface features: a negative feature must leave one or several negative entity clusterings in the part boundary after material removing; a positive feature must exist in the part boundary in a positive entity clustering. Thus the identified feature loops indicate the existence of basic surface features. The identified type I positive feature loop, type I negative feature loop and type III negative feature loop are converted into basic surface features without further processing. If two type II negative feature loops share a face and the non-shared faces are parallel, they can be merged to form a slot-shaped basic negative surface feature. If two type II positive feature loops share a face, they can be merged to form a rib-shaped basic positive surface feature. Therefore, the basic surface features can be recognized from the identified feature loops.

5.5 Example

Many real-world parts have been tested and the results show that it can robustly extract discrete basic surface features. An example is given below to demonstrate the generality of the proposed approach.

Test Part: NEW_DEMO_US

As described in Chapter 4, NEW_DEMO_US is a typical part machined by a 3-axis milling machine. The surface feature recognition is based on the simplified B-Rep model, as shown in Figure 4.22b, after pre-processing. There are total 28 negative basic surface features and 3 positive basic surface features recognized by applying the above proposed bi-attribute based method. Among the 28 negative basic surface features, there are 12
Figure 5.7: All the identified basic surface feature for the part shown in Figure 4.22b
Figure 5.7 continued
hole features (Figures 5.7a–5.7c), 4 blind slots with a cylindrical end face (Figure 5.7d), 4 pockets (Figure 5.7e–Figure 5.7h), 2 blind slots (Figure 5.7i) and 6 steps (Figure 5.7j–5.7l). Figure 5.7m–5.7o illustrates 3 recognized positive basic surface features. We can see from Figure 5.7g and 5.7h that the two identified negative surface features (pockets) share two planar faces. The face sharing also happens to the 6 steps.
5.6 Summary

In this chapter, we have presented a bi-attribute based approach to deal with basic surface feature recognition – Phase I of the proposed hybrid feature recognition. This approach investigates the underlying invariant properties of boundary entities and characterizes all the topological entities in proposed bi-attribute. Based on these attributes, primitive forms and shapes of an object can be extracted based on the clustering of surface entities and represented by basic surface features.
The surface feature recognition based on bi-attribute can very well handle the recognition of discrete features with various topology and geometry. When features interact, the topological pattern of the original features will be altered and the boundary entities of the features may be split, merged, or even removed. Thus it is not an easy job to recover the feature interactions only based on the analysis of boundary entities locally. Volumetric method is believed to be a good solution for dealing with the feature interactions. Volumetric feature extraction discussed in this chapter involves finding the maximum volumes for surface features. An incremental trimming procedure is devised to create the maximum feature volumes by exploiting the accessibility characteristic of the feature face set and utilizing the halfspace partition. Feature interaction is resolved by subsumption checking for all the extracted maximum feature volumes. The created maximum feature volumes always remain in contact with a portion of the original surface features to ensure that the boundary of the volumetric feature includes the corresponding surface features.

6.1 Introduction

Several methods have been proposed to construct feature volumes by adding closure faces to the faces of a surface feature to form a closed solid object. The techniques such
as half-space closure [Sakurai and Gossard, 1988], edge extension [Falcidieno and Giannini, 1989], and face extension [Dong and Wozny, 1991] are employed in those methods. In general they all use the neighboring faces of a feature to form the closure faces of a feature volume. Thus only local information is utilized to get the feature volume. None of them explored the accessibility analysis of the surface features to capture the global characteristics of feature volumes. Therefore, the problem of feature interaction can not be resolved in those methods.

Some researchers proposed the incremental volumetric features creation by removing each feature volume once it has been recognized. It may improve the efficiency of volume feature construction, but the major limitation of this approach is that it can not facilitate the generation of multiple interpretations, which is a very important consideration on the manufacturing optimization in process planning.

In order to solve feature interaction and support multiple feature interpretation, we propose to extract the maximum feature volume for each surface feature by exploiting accessibility characteristic and utilizing halfspace partition. Feature interaction is solved by subsumption checking for maximum feature volumes and reorganization of surface features. The extracted maximum feature volumes are independent from applications and processes and can be manipulated to generate multiple feature interpretations under constraints from applications and processes.

6.2 Mathematical & Theoretical Foundation

It is essential to have a clear mathematical formalism for the development of a general problem-solving approach in geometric reasoning. In this section some necessary mathematical and theoretical concepts are explored before proceeding with the presentation of our volumetric feature extraction methodology.
From the manufacturing viewpoint, a part is obtained by removing materials from its original stock. Various feature volumes are subtracted one by one till the formation of the final part. From the design viewpoint, a part can be formed by combining different feature volumes. The combining can be adding or subtracting. In geometric modeling the part, stock and feature volumes are represented by manifold solids bounded by a closed boundary. In this mathematical and theoretical foundation, manifold topology is introduced and the relationship among part, stock and volumetric features are investigated in terms of manifold solid and its boundary.

6.2.1 Manifold Topology

Topology usually is concerned with the adjacency relationships between topological entities such as faces, edges and vertices. Current solid modeling technology is based on what is known as manifold topology. A solid model has manifold (or two-manifold) topology if every point on its boundary is homeomorphic to an open disk. This means that the neighborhood of every point on the solid boundary (i.e. on the surfaces of the solid) is represented by a two-dimensional disk (therefore the term two-manifold). In other words, a face is topologically 'flat' when it is examined closely in a small enough area around any given point even though the face exists in $E^3$ space. The implication of manifold topology in practice is that all manifold solid models are closed, i.e. the solid boundary (vertices, edges and faces) are in direct contact with its interior. In other words, the boundary forms a tight skin covering the solid interior, or the solid interior is geometrically closed by its boundary.

In terms of the above definition of manifold topology, a solid model is a point set $X$ that divides the $E^3$ space into two regions: interior and exterior separated by the solid boundary. Let $\beta$ and $\gamma$ denote the boundary and interior set operators [Requicha, 1980] respectively, the solid model $X$ can be represented by:

$$S = \beta(S) \cup \gamma(S)$$  \hspace{1cm} (6.1)

Usually the solid can be interpreted as the geometric closure of the point set $S$, i.e.:
Thus from (6.1) and (6.2), the geometric closure \( k(S) \) can be represented by:
\[
k(S) = \beta(S) \cup \gamma(S) \quad (6.3)
\]
The implication of the geometric closure suggests that valid solid models are bounded, closed, regular, and semi-analytic subsets of \( E^3 \). These subsets are known as r-sets (regularized sets) [Requicha, 1980]. The point set \( S \) that defines a manifold solid model and is given by (6.1) is always an r-set. Intuitively, an r-set means that the set is closed and has no dangling entities such as vertices, edges and faces, has no more than two faces meeting at an edge, and has no edges emanating from a point on a face. The boundary of a two-manifold object, when divided into faces, edges, and vertices, has the following characteristics:
- every face is surrounded by one or more than one loop of edges;
- every edge is shared by two faces;
- every edge is incident at exactly two vertices, although these two vertices may be identical;
- every vertex is surrounded by only one disk of faces.

Boolean set operations \( \cup, \cap, \) and \( - \) are normally used to build solid models by combining 3D solid primitives which are r-sets. The results of these operations are not closed in general under manifold representations. A modified version of these operators, known as regularized set operators \( \cup^*, \cap^* \) and \( -^* \), was developed [Requicha, 1980] to generate only r-sets as results. In this research we extend the regularized set operators from 3D entity to 2D entity, i.e. the result of the regularized set operation between two 2D entities will also be a uniformly 2D entity.

6.2.2 Part Volume, Stock Volume and Delta Volume

A part is produced by removing material from the original stock material in manufacturing view. The union of all the removed materials is called delta volume \( \Delta \) in geometric modeling and reasoning. The space that the stock material occupies is called
stock volume $S$, while the space occupied by the part is called part volume $T$. In geometric modeling the delta volume equals the regularized Boolean subtraction [Hoffman, 1989] of part volume from stock volume as follows:

$$\Delta = S - T$$  \hspace{1cm} (6.4)

Part volume, stock volume and delta volume exist as solid models, thus they should conform with manifold topology and each should consists of two parts: interior and boundary as follows:

$$T = \beta(T) \cup \gamma(T)$$  \hspace{1cm} (6.5)

$$S = \beta(S) \cup \gamma(S)$$  \hspace{1cm} (6.6)

$$\Delta = \beta(\Delta) \cup \gamma(\Delta)$$  \hspace{1cm} (6.7)

We need to notice that the removed materials may not be connected so that the delta volume may comprise several separated solid volumes as shown in Figure 6.1c and 6.1d, i.e.

$$\Delta = \{\delta_i, i = 1..n\}$$  \hspace{1cm} (6.8)

Since each sub-volumes $\delta_i$ in the delta volumes is a solid, we have

$$\delta_i = \beta(\delta_i) \cup \gamma(\delta_i)$$  \hspace{1cm} (6.9)

For the delta volume consisting of more than one solid, we define its interior and boundary to be the union of the interior and boundary of its comprising sub-volumes as follows:

$$\beta(\Delta) = \bigcup_i \beta(\delta_i)$$  \hspace{1cm} (6.10)

$$\gamma(\Delta) = \bigcup_i \gamma(\delta_i)$$  \hspace{1cm} (6.11)

Both the boundary of the part volume $\beta(T)$ and the boundary of the delta volume $\beta(\Delta)$ can be partitioned into two parts: the face set that has overlap with the boundary of the stock volume $\beta(S)$ and the face set that has no overlap with $\beta(S)$. The former one is called stock face set and the latter one is called non-stock face set. Let $\beta^s$ and $\beta^u$ denote the stock face set operator and non-stock face set operator respectively, we can obtain the following:
\begin{align}
\beta(T) &= \beta^s(T) \cup^* \beta^m(T), \quad \beta^s(T) = \beta(T) \cap^* \beta(S) \quad \text{and} \quad \beta^m(T) \cap^* \beta(S) = \emptyset \quad (6.12) \\
\beta(\Delta) &= \beta^s(\Delta) \cup^* \beta^m(\Delta), \quad \beta^s(\Delta) = \beta(\Delta) \cap^* \beta(S) \quad \text{and} \quad \beta^m(\Delta) \cap^* \beta(S) = \emptyset \quad (6.13)
\end{align}

From (6.4) we can have

\[ S = T \cup^* \Delta \quad (6.14) \]

i.e. the regularized union of the part volume and delta volume equals the stock volume. Therefore, it is obvious that we can have

\[ \beta(S) = \beta^s(T) \cup^* \beta^s(\Delta) \quad (6.15) \]

i.e. the boundary of the stock volume can be divided into two parts: the stock face set of the boundary of the part volume and the stock face set of the boundary of the delta volume. \( \beta^s(\Delta) \) represents all the removed faces in \( \beta(S) \) while \( \beta^s(T) \) represents all the existent faces in \( \beta(S) \) after removing \( \Delta \) from \( S \). Removing \( \Delta \) will result in the generation of new faces, which forms the non-stock faces. The non-stock face set forms the demarcation between delta volume \( \Delta \) and part volume \( T \) and is same for \( \Delta \) and \( T \), i.e.

\[ \beta^m(T) = \beta^m(\Delta) \quad (6.16) \]

The computation of stock face set and non-stock face set is decided by the stock volume. Once the stock volume is available, the stock face set and non-stock face set for part volume and delta volume can be computed based on (6.12), (6.13) and (6.14).

For simplicity the stock volume \( S \) can be computed as the bounding box for prismatic parts (Figure 6.1b) or the enclosing cylinder for rotational parts (Figure 6.2b). It can be enlarged to meet certain manufacturing requirements. For rectangular stock volume, it can be enlarged along the three major axes (Figure 6.1e). For cylindrical stock, it can be increased along the axial and radial directions (Figure 6.2d). The stock volume \( S \) is not necessary to be a rectangular block or cylinder. In order to save material and cut the machining cost, some mechanical parts may be manufactured by using multiple manufacturing processes. Primary processes such as casting realize the primary shape of the part, while secondary processes such as machining generate more detailed shape of
the part. Thus the stock volume for those parts will be the model of the casting parts which may have very near shape with respect to the model of the final parts as shown in Figure 6.3b. We will not discuss the automatic generation of the stock volumes for the model of the final parts manufactured whether by a single process or multiple processes since it is outside of our research focus. In this research work the stock volume, which can be bounding block/cylinder or near part shape, is assumed to be available before the volumetric feature extraction. If the stock volume is not available in the input, we will use the minimum enclosing box or cylinder as the default stock volume.

Different stock volumes may result in different stock face set and non-stock face set. Figure 6.4a shows a simple part $T$ (a through hole in a base block). A stock volume $S$ can be computed in bounding box as shown in Figure 6.4b. The corresponding delta volume $\Delta$ (Figure 6.4c) can be computed based on equation (6.4). The stock volume can be enlarged evenly along the positive and negative directions of the three major axes as shown in Figure 6.4d. Based on the enlarged stock volume $S_e$ and equation (6.4), an enlarged delta volume $\Delta_e$ can be computed as shown in Figure 6.4e. The stock face set and non-stock face set of part volume and delta volume can be computed for the stock volume $S$ and the enlarged stock volume $S_e$ based on equations (6.12) and (6.13), and the results are as follows:

For $S$:

\[
\begin{align*}
\beta^s(T) &= \{F_1, F_2, F_3, F_4, F_5, F_6\} \\
\beta^m(T) &= \{F_7\} \\
\beta^s(\Delta) &= \{F_8, F_9\} \\
\beta^m(\Delta) &= \{F_7\}
\end{align*}
\]

For $S_e$:

\[
\begin{align*}
\beta^s(T) &= \{\emptyset\} \\
\beta^m(T) &= \{F_1, F_2, F_3, F_4, F_5, F_6, F_7\} \\
\beta^s(\Delta_e) &= \{F_8, F_9, F_{10}, F_{11}, F_{12}, F_{13}\} \\
\beta^m(\Delta_e) &= \{F_1, F_2, F_3, F_4, F_5, F_6, F_7\}
\end{align*}
\]

119
Figure 6.1: Illustration of the part volume, stock volume, delta volume and enlarged stock volume of NEW_DEMO_US, a test part from NIST Part Library
Figure 6.2: Illustration of the part volume, stock volume, delta volume and enlarged stock volume of a socket, an example part from [Regli, 1995]

Figure 6.3: Near-shape stock volume of an example part
We can see that when the stock volume is enlarged to subsume the part volume, all the faces in the boundary of the part volume will become non-stock faces, and the stock face set of the part volume will become an empty set. According to equation (6.16) the non-stock face set of the enlarged delta volume is same as the non-stock face set of the part volume. From equation (6.15), we can have \( \beta(S_x) = \beta^*(T) \cup^* \beta^*(A_e) \). Since \( \beta^*(T) = \emptyset \), we can get \( \beta(S_x) = \beta^*(A_e) \), i.e. all the faces in the boundary of the enlarged stock volume form the stock face set of the enlarged delta volume, which can be verified from the above results and Figure 6.4d and 6.4e. Therefore, we can conclude that the stock volume decides the distribution of stock faces and non-stock faces in the boundary of part volume and delta volume. Figure 6.5 illustrates the distribution of stock faces and non-stock faces of \( \beta(T) \), \( \beta(S) \) and \( \beta(A) \) for the example part NEW_DEMO_US (Figure 6.1a) and its stock volume shown in Figure 6.1b.

Figure 6.4: Stock face set and non-stock face set for different stock volumes
Figure 6.5: Distribution of stock faces and non-stock faces of $\beta(T)$, $\beta(S)$ and $\beta(\Delta)$ for NEW DEMO US
6.2.3 Positive and Negative Volumetric Features

Geometric reasoning applications naturally differentiate between features that consist of the existent material of the part and those that consist of material removal volumes with respect to the part's original stock. Positive and negative volumetric feature classes provide for this type of composition based distinction.

Let \( P' \) and \( N' \) denote positive volumetric feature and negative volumetric feature respectively. Each volumetric feature can be viewed as an independent manifold solid. The interior of any positive volumetric feature \( P' \) is a subset of the interior of the part volume \( T \) and the intersection of the boundary of \( P' \) and the boundary of \( T \) is not empty, i.e.

\[
\gamma(P') \subseteq \gamma(T) \quad \text{and} \quad \beta(P') \cap \beta(T) \neq \emptyset \quad (6.17)
\]

The interior of any negative volumetric feature \( N' \) is a subset of the interior of the delta volume \( \Delta \) and the intersection of the boundary of \( N' \) and the boundary of \( T \) is not empty, i.e.:

\[
\gamma(N') \subseteq \gamma(\Delta) \quad \text{and} \quad \beta(N') \cap \beta(T) \neq \emptyset \quad (6.18)
\]

From \( \gamma(N') \subseteq \gamma(\Delta) \) and \( \Delta = S - T \), we can get:

\[
\gamma(N') \cap \gamma(T) = \emptyset \quad (6.19)
\]

i.e., the interiors of the negative volumetric feature \( N' \) and the part volume \( T \) are disjoint.

For volumetric feature, the closed boundary can be divided into two portions: real boundary and virtual boundary. Real boundary is the portion that is overlapped with the boundary of the part volume, and virtual boundary is the portion that is not overlapped with the boundary of the part volume. Let \( \beta' \) and \( \beta'' \) denote the real boundary set operator and virtual boundary set operator respectively, then we can have the following equations:
\[ \text{for } P^*: \begin{align*}
\beta^*(P^*) &= \beta(P^*) \cap \beta(T) \\
\beta^*(P^*) &= \beta(P^*) - \beta^*(P^*)
\end{align*} \quad (a) \quad (6.20) \\
\text{for } N^*: \begin{align*}
\beta^*(N^*) &= \beta(N^*) \cap \beta(T) \\
\beta^*(N^*) &= \beta(N^*) - \beta^*(N^*)
\end{align*} \quad (b) \quad (6.21) \]

Let \( P^* \) and \( N^* \) denote basic positive surface feature and basic negative surface feature respectively. Positive volumetric features and negative volumetric features can be constructed based on the identified \( P^* \) and \( N^* \) that are the output of surface feature recognizer (Chapter 5). However, when feature interactions exist, the mapping from surface feature to volumetric feature will be non-unique, i.e. multiple volumetric feature sets can be obtained from one surface feature set. In real-world mechanical parts, feature interactions usually are not simply pair-wise. More than two features may be involved in the interaction so that the mapping can be more complicated. In order to solve this problem, we choose to construct the maximum feature volume for all the recognized surface features to achieve one-to-one mapping. Figure 6.6 illustrates the mapping from surface feature to volumetric feature. Two negative surface features \( P^*_1 \) and \( P^*_2 \) can be recognized by the proposed surface feature recognizer. From \( M_1 \) mapping, we can see that multiple sets of volumetric features can be derived, and the removing of each volumetric feature set can generate the corresponding surface features and the final part volume. If only maximum feature volume is allowed in the mapping from the negative surface feature set, only one volumetric feature set can be obtained as shown from the \( M_2 \) mapping in Figure 6.6. If not indicated specially, in the following discussion volumetric features means the maximum volumetric features.

**6.2.4 Machining Features and Design Features**

The extracted volumetric features form a hybrid feature set including positive volumetric features and negative volumetric features. The hybrid feature set is application independent and process independent. In order to utilize the hybrid feature set, application-related features need to be derived to support various concurrent engineering, we will focus on the generation of machining feature and design feature in this research.
Figure 6.6: Illustration of 1:n and 1:1 mapping from negative surface feature to negative volumetric feature

Machining features represent the actual material volumes removed from the stock after a process plan is decided. Thus their actual shapes are depended on the practical machining conditions. Based on the machining conditions, negative volumetric features can be sequenced and the actual volume of the machining features can be computed. In design application, positive features are more preferable since they can be used to quickly layout the preliminary shape of the part model. Negative features then can be applied in this coarse shape. Thus all the positive volumetric features and certain negative volumetric
features can be selected from the hybrid feature set to form a design feature set. The derivation of machining feature and design feature can be performed in the post-recognition processing, which will be discussed in detail in Chapter 7.

6.2.5 Properties of Volumetric Features

Based on the above discussion, we explored the intrinsic characteristics of volumetric features including negative volumetric feature, positive volumetric feature, machining feature and design feature, and summarized them into six properties including completeness, inclusion, non-intrusion, existence, uniqueness and exclusion. Each property is discussed as follows:

(1) Completeness

The union of all the negative volumetric features should be equal to the delta volume, i.e.

\[ \bigcup_i N_i^* = \Delta \text{, where } \Delta = S^* T \]  \hspace{1cm} (6.22)

Thus the removing of all the negative volumetric features from the stock volume will generate the final part volume as the following expression,

\[ S^* (\bigcup_i N_i^*) = T \]  \hspace{1cm} (6.23)

Considering the volumetric overlapping among the negative volumetric features and efficient manufacturing for less time and cost, the negative volumetric features are sequenced in certain order and their volume are updated and transformed into machining features \( M_i \). Since only the overlapping is reduced by such a process, the union of all the negative volumetric features should be same as the union of the machining features as follows,

\[ \bigcup_i N_i^* = \bigcup_i M_i \]  \hspace{1cm} (6.24)

From (6.22), (6.23) and (6.24), we can have

\[ \bigcup_i M_i = \Delta \]  \hspace{1cm} (6.25)

\[ S^* (\bigcup_i M_i) = T \]  \hspace{1cm} (6.26)
Equation (6.25) shows that completeness also applies for machining features. Equation (6.26) shows that the final part can be obtained by the removing of a set of machining features, which is derived from the set of negative volumetric features, from the stock volume. Thus $M$ can be directly used in the process planning. Based on certain heuristics and different machining setups and resources, different sets of machining features can be derived from the set of negative volumetric features. This is so-called multiple feature interpretation in machining. Thus we can get multiple process plans to machine a part. An optimal set of machining features can be achieved by comparing the multiple process plans. Further discussion in multiple feature interpretation will be presented in Chapter 7.

From the machining viewpoint, completeness plays an important role in ensuring that the removing of all the machining features can result in the final part volume. For design applications, completeness has a different interpretation than that in machining applications. Completeness in the design domain requires that all the design features be combined to be able to generate the final part shape. Let $\{D_i, i = 1..n\}$ be a set of design features, among which $\{D_i, i = 1..k\}$ be the positive design features and $\{D_j, j = (k+1)..n\}$ be the negative design features. The design-based completeness can be expressed as follows,

$$\left( \bigcup_{i=1}^{k} D_i \right)^* - \left( \bigcup_{j=k+1}^{n} D_j \right)^* = T$$  \hspace{1cm} (6.27)

i.e. the final part volume is created by the subtraction of the union of all the negative design features from the union of all the positive design features. If that is not the case, it shows that the set of recognized design features are incomplete.

Since the union of the negative volumetric features forms a delta volume, we can view the delta volume as a decomposition into many negative volumetric features.
In this research the negative volumetric features are extracted based on basic surface features. Thus it is a functional decomposition, more than just spatial decomposition which is employed in cellular decomposition as discussed in Chapter 2.

(2) Inclusion
Since machining features are derived from negative volumetric features, every derived machining feature must be a subset of its corresponding negative volumetric feature as follows,

\[ M_i \subseteq N_i \]  \hspace{1cm} (6.28)

Notice that if we have

\[ N_i \cap \bigcap_{j=1}^{n} N_j = \emptyset, j \neq i \]  \hspace{1cm} (6.29)

i.e. the negative volumetric feature \( N_i \) is isolated and not overlap with any other negative volumetric feature \( N_j \), then the derived machining feature \( M_i \) will be same as \( N_i \), i.e.

\[ M_i = N_i \]  \hspace{1cm} (6.30)

Thus the negative volumetric feature can be directly converted into a machining feature.

(3) Non-intrusion
The machining operation for a negative volumetric feature should not intrude the part, i.e.

\[ N_i \cap T = \emptyset \]  \hspace{1cm} (6.31)

From property (2), we know \( M_i \subseteq N_i \). Thus, we can have

\[ M_i \cap T = \emptyset \]  \hspace{1cm} (6.32)

(4) Existence
As discussed in Section 6.2.3, both the boundary of \( N' \) and the boundary of \( P' \) have overlap with the boundary of the part volume \( T \). In other words, at least a portion of the boundary of the volumetric features must come from the part boundary to show
their existence; otherwise they are invalid volumetric features. We can also interpret that the removal of the negative volumetric features must leave a trace in part boundary and the positive volumetric features must have their boundary appearance in the final part.

(5) Uniqueness

Each negative volumetric feature should not be subsumed by another negative volumetric feature, i.e. each negative volumetric feature should not be a subset of another negative volumetric feature as follows,

$$N_i \nsubseteq N_j, j \neq i \text{ and } j = 1..n$$ (6.33)

If $N_i$ is subsumed by $N_j$, it shows that the machining of $N_j$ will automatically remove $N_i$. Thus there is no need to keep $N_i$ in the set of recognized negative volumetric features. If $\beta^e(N_j)$, the real boundary of $N_i$ is not subsumed by $\beta^e(N_j)$, the real boundary of $N_j$, $\beta^e(N_i)$ should be merged with $\beta^e(N_j)$ to generate a larger real boundary for $N_j$ since the removing of $N_j$ will not only create $\beta^e(N_j)$ in the part boundary, but also generate $\beta^e(N_i)$ at the same time. It can be expressed as follows.

$$\text{if } N_i \subseteq N_j \text{ and } \beta^e(N_i) \nsubseteq \beta^e(N_j), \text{ then } \beta^e(N_i) \cup \beta^e(N_j) \Rightarrow \beta^e(N_j)$$ (6.34)

Similarly, the uniqueness property is also applied to the positive volumetric features. Each positive volumetric feature should not be a subset of another positive volumetric feature, i.e.

$$P_i \nsubseteq P_j, j \neq i \text{ and } j = 1..n$$ (6.38)

(6) Exclusion

Positive features are viewed differently in design and machining. From the machining viewpoint, the positive features are generated due to the removing of the surrounding negative features. From the design viewpoint, a positive feature is added to another parent positive feature by adding some material. As mentioned above, in order to
recover a reasonable design scheme, it is more natural to choose positive features as many as possible based on the observation that positive features are preferred in design activity so as to quickly layout the coarse outer shape of the desired part, then negative features are added to further modify the coarse shape to get the final part. Thus, in this research we choose all the positive volumetric features as design features and the negative volumetric features that are not surrounding the positive volumetric features. Here the surrounding means the negative volumetric feature has overlapping with the positive volumetric feature in real boundary. Based on the nomenclature defined in property of completeness, we can have the following exclusiveness constraint for the design features in terms of real boundary:

$$\beta^*(D_i) \cap ^* \beta^*(D_j) = \emptyset, i \in [1..k], j \in [(k+1)..n]$$  \hspace{1cm} (6.39)

### 6.3 Volumetric Feature Extraction

Volumetric feature extraction involves the finding of the maximum feature volume $V_{\text{max}}$ of the surface features. Maximum feature volumes are constructed to achieve one-to-one mapping between surface feature and volumetric feature. Considering the different formation of positive features and negative features, different strategies are employed to extract volumetric features from negative surface feature and positive surface feature.

#### 6.3.1 Negative Volumetric Feature Extraction

The negative volumetric feature extraction consists of two phases: construction and reorganization. In the construction phase, the maximum feature volume is created for each surface feature by a trimming procedure based on feature accessibility and halfspace partition. In the reorganization phase, the subsumption relationship among the feature volumes is checked and the surface features are merged if their corresponding feature volumes have subsumption relationship, which results in the resolving of feature interaction problem.
6.3.1.1 Feature Accessibility

Features are recognizable by the characteristic traces they leave in the nominal geometry of a part. These characteristic traces, also called surface features, provide reliable clues or hints for the potential existence of volumetric features, even when feature interactions occur. In order to construct volumetric features from identified surface features, feature accessibility needs to be exploited so as to guide the global volume construction based on the local surface feature. In this research the accessibility of surface feature is quantified by the possible accessible directions, denoted by $D_{\text{acc}}$. Figure 6.7 illustrates several typical surface features and their $D_{\text{acc}}$.

\[ N_1 = \{F_1, F_2\}, \quad \text{Num}(D_{\text{acc}}) = 1 \]
\[ N_2 = \{F_3, F_4, F_5\}, \quad \text{Num}(D_{\text{acc}}) = 3 \]
\[ N_3 = \{F_6, F_7\}, \quad \text{Num}(D_{\text{acc}}) = 4 \]
\[ N_4 = \{F_8, F_9\}, \quad \text{Num}(D_{\text{acc}}) = 3 \]
\[ N_5 = \{F_{10}\}, \quad \text{Num}(D_{\text{acc}}) = 5 \]

Figure 6.7: Illustration of feature accessibility of various surface features
We observed that two types of accessible directions exist in $D_{acc}$. One type is the accessible direction that is constrained by the comprising surface of the surface feature and has the same direction with the face normal vector. The other type is the accessible direction that is not constrained by any comprising surface and thus its opposite direction is also an accessible one. The first type is referred to type I $D_{acc}$, and the second type is referred to type II $D_{acc}$. As shown in Figure 6.7, all the three accessible directions of $N^*_d$ are type I $D_{acc}$. For $N^*_f$, two accessible directions are type I $D_{acc}$ and the other two in opposite directions are type II $D_{acc}$. For planar faces in surface feature, the face normal forms a type I $D_{acc}$. Since the input B-Rep model for feature recognition is already simplified, the existent cylindrical faces will be the traces left by machining operations such as hole drilling and slot milling. It is obvious that the possible accessible directions for closed cylindrical faces are along the axial direction of the cylindrical face and no $D_{acc}$ is allowed along any radial direction. Thus no type I $D_{acc}$ exist for closed cylindrical faces. Type II $D_{acc}$ can be computed based on type I $D_{acc}$ and the planar face configuration of the surface feature. Therefore, type II $D_{acc}$ is derived after the computation of type I $D_{acc}$. For a closed cylindrical face, there exists two possible type II $D_{acc}$ along its axial direction. However, if a planar face is grouped with the cylindrical face to form a blind hole (such as $N^*_1$ in Figure 6.7), one type II $D_{acc}$ will be constrained. The only one allowable type II $D_{acc}$ is in the same direction with that of the type I $D_{acc}$ of the planar face. Thus for a blind hole, only the type I $D_{acc}$ is computed from its comprising planar face; for a through hole, two type II $D_{acc}$ are computed in opposite direction along the axial direction of its comprising cylindrical face. The $D_{acc}$ of half-open cylindrical end face of a slot is similar to that of closed cylindrical faces except that it has one more type I $D_{acc}$ along the normal of its open side face.
According to the 3rd property of volumetric features, the maximum feature volume of the basic negative surface feature should not intrude the part. This implies that along the accessible direction the nearest part face will become the constraint face to limit the scope of the maximum feature volume. If no part face exists along the accessible direction, the boundary face of the stock volume will become the constraint face. It is obvious that extensive spatial relationship checking will be involved for the efficient constraint face searching. Thus before our discussion of the proposed strategies for constraint face searching, some useful spatial relationships are introduced.

6.3.1.2 Spatial Relationship

Spatial relationships between topological entities play a vital role in geometric reasoning. In our volumetric feature extraction face-face, face-volume and volume-volume spatial relationships are mainly utilized to facilitate the searching of constraint face for the generation of the maximum feature volume.

Three specific types are considered in face-face spatial relationship: parallel/counter-parallel, angular/counter-angular and overlap/non-overlap. Given two non-adjacent and non-coplanar faces F1 and F2 and their normal vectors n1 and n2, let $\angle(n_1, n_2)$ denote the angle between n1 and n2, the parallel(∥), counter-parallel(⊥), angular(∠) and counter-angular(∡) relationships can be described as follows:

- Parallel: $\angle(n_1, n_2) = 0^\circ \Rightarrow F_1 \parallel F_2$
- Counter-parallel: $\angle(n_1, n_2) = 180^\circ \Rightarrow F_1 \perp F_2$
- Angular: $\angle(n_1, n_2) \in (0^\circ, 90^\circ) \Rightarrow F_1 \angle F_2$
- Counter-angular: $\angle(n_1, n_2) \in [90^\circ, 180^\circ) \Rightarrow F_1 \angle F_2$

Since F1 and F2 are non-coplanar, we can project face F1 onto face F2 along the reverse direction of n2 and then check the overlap relationship between F2 and projected F1. Let $\rho$ denote the projection operator and $\rho(F_1 \rightarrow F_2)$ denote the projection of F1 onto F2, overlap(∩) and non-overlap(⊈) relationships can be described as follows:
• Overlap: \[ \rho(F_1 \rightarrow F_2) \cap^* F_2 \neq \emptyset \Rightarrow F_1 \vee F_2 \]
• Non-overlap: \[ \rho(F_1 \rightarrow F_2) \cap^* F_2 = \emptyset \Rightarrow F_1 \vee F_2 \]

The face-volume spatial relationship considered in this section is inclusion/exclusion. The inclusion/exclusion can be checked by the regularized Boolean intersection of a face F and a volume V, i.e. \( F \cap V \). If \( F \cap V \neq \emptyset \), then it shows that at least a portion of F is included by V, and we define it as inclusion (\( \prec \)). If \( F \cap V = \emptyset \), then it shows that F is totally outside of V, and we define it as exclusion (\( \succ \)). Thus inclusion/exclusion can be expressed as follows:
• Inclusion: \( F \cap V \neq \emptyset \Rightarrow F \prec V \)
• Exclusion: \( F \cap V = \emptyset \Rightarrow F \succ V \)

The volume-volume spatial relationship defined here is subsumption. Given two volumes \( V_1 \) and \( V_2 \), if and only if \( V_1 \) is fully subsumed by \( V_2 \), we can claim that there exist a subsumption relationship (\( \subseteq \)) between \( V_1 \) and \( V_2 \). The subsumption relationship can be checked by a combination of regularized Boolean union and subtraction and expressed as follows:
• Subsumption: \( (V_1 \cup^* V_2) \setminus^* V_2 = \emptyset \Rightarrow V_1 \subseteq V_2 \)

One extreme case is that \( V_1 \) and \( V_2 \) are same. It is obvious that the equivalence relationship is a special case of subsumption relationship.

### 6.3.1.3 Halfspace Partition

Maximum feature volume construction can be achieved by the halfspace partition of stock volume based on the feature faces and constraint faces. Before the discussion of halfspace partition, the concept of halfspace is briefly presented at first.
**Halfspace**

It is intuitive that the infinite extension of a planar face can separate a $E^3$ space into two parts, each called a halfspace $HS$. Thus a halfspace in $E^3$ is the set of points on or on one side of a planar face $F$. In order to differentiate the two halfspaces, the face normal vector $n$ is utilized. The halfspace where the face normal points toward is referred to positive halfspace $HS^+$, and the halfspace where the face normal points away is referred to negative halfspace $HS^-$, as shown in Figure 6.8. Let $q$ be a point on the planar face; then the negative halfspace $HS^-$ and the positive halfspace $HS^+$ determined by the planar face $F$ are defined as follows:

$$HS^- (F) = \{ p | n^T (p - q) \leq 0 \}$$

$$HS^+ (F) = \{ p | n^T (p - q) \geq 0 \}$$

Figure 6.8: Two halfspaces of a planar face
The concept of halfspace can be extended to other face types such as cylindrical face, conical face and spherical face, as shown in Figure 6.9, since their infinite extension can also separate $E^3$ space into two parts. We know that for any face in the boundary of a manifold solid, the face normal is defined to be pointing outward, i.e. pointing away from the solid material. Thus for the face in the boundary of part model, $HS^-$ is the material (solid) side of the face, and $HS^+$ is the non-material (air) side.

**Halfspace Partition**

Halfspace partition is a very popular technique in the cell-based volumetric decomposition for feature recognition. As discussed in Chapter 2, the major problem of cell-based volumetric decomposition is the global effect of local geometry, i.e., it uses the
halfspaces of all the faces in the delta volume to brutally decompose the whole delta volume into quite a huge number of small cells that will result in the combinatorial explosion in cell composition for the construction of reasonable machining feature volumes. [Tilove, 1981] pointed out that nearly all the mechanical parts exhibit a high degree of locality, and that failure to exploit this results in unacceptably slow algorithms. Thus it is of practical importance to apply locality techniques for use with important geometric reasoning applications. In this research we also adopt halfspace partition method to construct feature volumes. However, we exploit the locality by performing the recognition of basic surface feature (Chapter 5) and the identification of constraint faces based on feature accessibility of the underlying basic surface feature. The constraint faces limit the scope of the feature volume so as to ensure the non-intrusion property of constructed feature volume and at the same time guarantee that the constructed volume is maximum based on the feature accessibility. The maximum feature volume is computed by intersecting the stock volume with the positive halfspace (non-material side) of all the feature faces and the identified constraint faces. It is apparent that no volume composition is needed in the construction of feature volumes. Therefore, the problem of combinatorial explosion can be avoided in our halfspace partition method. Figure 6.10 illustrates the halfspace partition for the maximum feature volume construction of the surface feature $N^*_4$ shown in Figure 6.7. \{$F_7,F_8,F_9$\} forms the feature face set and \{$F_{11},F_{12}$\} forms the constraint face set that are computed based on the feature accessibility of $N^*_4$. After the identification of constraint faces for each surface feature, the halfspace partition method can be applied to generate its maximum feature volume. Figure 6.11 shows all the constructed maximum feature volumes for the recognized basic negative surface features under the condition that the stock volume is the bounding box of the part volume.

**Constraint Face**

Constraint faces are identified to limit the scope of the maximum feature volume to ensure the non-intrusion property. Since the removing of negative features will only
generate the non-stock faces in the part boundary, the constraint faces must come from non-stock faces of the part boundary. The basic idea for constraint face searching is: along the accessible directions $D_{acc}$ of the basic surface feature, identify the non-stock faces that can ensure the non-intrusion property.

Figure 6.10: Halfspace partition for the maximum feature volume construction of $N_4^*$ shown in Figure 6.7
Along $D_{ac}$ non-stock faces can have various spatial relationships with respect to the faces of the surface feature such as the face-face relationships described in 6.3.1.2. In addition, in a complex part model with many features, a large amount of non-stock faces can exist. In order to quickly identify the suitable constraint faces, we propose an incremental trimming procedure to achieve good computational efficiency. First the positive halfspaces of the feature faces are intersected with the stock volume to generate an intermediate volume. It is obvious that only those non-stock faces that have inclusion relationship with the intermediate volume will be valid candidates for constraint faces. The constraint face identification is prioritized in five steps based on two criteria and three derived rules discussed in the following. Once a subset of constraint faces is found
in each step, the positive halfspaces of the identified constraint faces are intersected with the current intermediate volume to generate a new trimmed intermediate volume. Each time after the creation of a new intermediate volume, the non-stock faces are updated by discarding those that are not included by the new intermediate volume. As we can see that the proposed trimming procedure can significantly reduce the searching domain for the identification of constraint faces.

Based on the in-depth observation, two criteria are proposed for the constraint face identification: (1) type I $D_{\text{acc}}$ has higher priority than type II $D_{\text{acc}}$ since it is more reliable to search constraint faces based on the existent faces of the surface feature; (2) regular features are more preferable than irregular features. Similar to the presence rules proposed by [Vandenbrande, 1990], we proposed the following rules for the identification of constraint faces:

(1) Only non-stock faces that are facing opposed to the current $D_{\text{acc}}$ will be valid candidates of constraint faces since they can ensure the non-intrusion property;

(2) The non-stock faces that have overlap relationship with the feature face are more preferable than those without overlap relationship.

(3) The non-stock faces that have parallel relationship with the feature face are more preferable than those without parallel relationship, i.e. with angular relationship.

Based on the proposed criteria and rules, the identification of constraint faces are prioritized in the following five steps:

Step 1: For type I $D_{\text{acc}}$, find the nearest non-stock faces $F^{m1}$ that have overlap ($\vee$) and counter-parallel ($\hat{i}$) relationships with respect to the feature faces;

Step 2: For type I $D_{\text{acc}}$, find the nearest non-stock faces $F^{m2}$ that have overlap ($\vee$) and counter-angular ($\hat{Z}$) relationships with respect to the feature faces;

Step 3: For type II $D_{\text{acc}}$, find the nearest non-stock faces $F^{m3}$ that have overlap relationship ($\vee$) with respect to the feature faces and are counter-parallel ($\hat{i}$) to $D_{\text{acc}}$;
Step 4: For type II $D_{occ}$, find the non-stock faces $F^{ms}$ that have overlap relationship ($\lor$) with respect to the feature faces and are counter-angular ($\angle$) to $D_{occ}$.

Step 5: Find the unprocessed non-stock faces $F^{ms}$ that are counter-angular to one of the feature faces.

**Incremental Trimming Procedure $T_{ef}(N_{i}^{s})$**

Based on the above five steps, an incremental trimming procedure $T_{ef}(N_{i}^{s})$ is devised to construct the maximum feature volume more efficiently for each recognized basic negative surface feature.

1. Trim stock volume $S$ by the positive halfspaces of all the feature faces $\{F_{i}^{sf}\}$ of the surface feature $N_{i}^{s}$, which can be expressed by the following regularized Boolean intersections:

$$V_{max}^{1} = S \cap H^{+}(F_{i}^{sf}) \cap H^{+}(F_{2}^{sf}) \cap \ldots \cap H^{+}(F_{n}^{sf})$$

where $V_{max}^{1}$ is the resulting intermediate volume after trimming $S$.

2. Trim $V_{max}^{1}$ by the positive halfspaces of all the faces $F^{ms1}$ included by $V_{max}^{1}$, i.e., $\{F_{i}^{ms1} | F_{i}^{ms1} \in F^{ms1}, F_{i}^{ms1} \neq V_{max}^{1}\}$. This can be expressed by the following regularized Boolean intersections:

$$V_{max}^{2} = V_{max}^{1} \cap H^{+}(F_{1}^{ms1}) \cap H^{+}(F_{2}^{ms1}) \cap \ldots \cap H^{+}(F_{n}^{ms1})$$

where $V_{max}^{2}$ is the resulting intermediate volume after trimming $V_{max}^{1}$.

3. Trim $V_{max}^{2}$ by the positive halfspaces of all the faces $F^{ms2}$ included by $V_{max}^{2}$, i.e., $\{F_{i}^{ms2} | F_{i}^{ms2} \in F^{ms2}, F_{i}^{ms2} \neq V_{max}^{2}\}$. This can be expressed by the following regularized Boolean intersections:

$$V_{max}^{3} = V_{max}^{2} \cap H^{+}(F_{1}^{ms2}) \cap H^{+}(F_{2}^{ms2}) \cap \ldots \cap H^{+}(F_{n}^{ms2})$$

where $V_{max}^{3}$ is the resulting intermediate volume after trimming $V_{max}^{2}$.
4. Trim $V_{\text{max}}^3$ by the positive halfspaces of all the faces $F_{\text{ms}}$ included by $V_{\text{max}}^3$, i.e.,
\[ \{ F_{i\text{ms}} \mid F_{i\text{ms}} \in F_{\text{ms}}, F_{i\text{ms}} \prec V_{\text{max}}^3 \} . \]
This can be expressed by the following regularized Boolean intersections:
\[ V_{\text{max}}^4 = V_{\text{max}}^3 \cap^* HS^+(F_{i\text{ms}}) \cap^* HS^+(F_{2\text{ms}}) \cap^* \cdots \cap^* HS^+(F_{n\text{ms}}) \]
where $V_{\text{max}}^4$ is the resulting intermediate volume after trimming $V_{\text{max}}^3$.

5. Trim $V_{\text{max}}^4$ by the positive halfspaces of all the faces $F_{\text{ms}}$ included by $V_{\text{max}}^4$, i.e.,
\[ \{ F_{i\text{ms}} \mid F_{i\text{ms}} \in F_{\text{ms}}, F_{i\text{ms}} \prec V_{\text{max}}^4 \} . \]
This can be expressed by the following regularized Boolean intersections:
\[ V_{\text{max}}^5 = V_{\text{max}}^4 \cap^* HS^+(F_{i\text{ms}}) \cap^* HS^+(F_{2\text{ms}}) \cap^* \cdots \cap^* HS^+(F_{n\text{ms}}) \]
where $V_{\text{max}}^5$ is the resulting intermediate volume after trimming $V_{\text{max}}^4$.

6. Trim $V_{\text{max}}^5$ by the positive halfspaces of all the faces $F_{\text{ms}}$ included by $V_{\text{max}}^5$, i.e.,
\[ \{ F_{i\text{ms}} \mid F_{i\text{ms}} \in F_{\text{ms}}, F_{i\text{ms}} \prec V_{\text{max}}^5 \} . \]
This can be expressed by the following regularized Boolean intersections:
\[ V_{\text{max}} = V_{\text{max}}^5 \cap^* HS^+(F_{i\text{ms}}) \cap^* HS^+(F_{2\text{ms}}) \cap^* \cdots \cap^* HS^+(F_{n\text{ms}}) \]
where $V_{\text{max}}$ is the final maximum feature volume after trimming $V_{\text{max}}^5$.

After the above six trimming steps, a maximum feature volume $V_{\text{max}}$ can be created for each basic negative surface feature. From the viewpoint of accessibility, $T_{\text{nf}}(N_i)$ can be viewed as a procedure that grow the feature volumetrically along the reverse accessible directions.

6.3.1.4 Processing of Cylindrical End Faces of Slots

Halfspace partition works well for the surface feature comprising a group of planar faces such as pockets, or a single cylindrical face such as holes. When the surface feature consists of both planar faces and cylindrical faces such as slots with cylindrical end faces, we can not simply use all the faces in the surface feature to create the feature volumes.
[Vandenbrande, 1990] divided a slot into two parts: the main body of the slot and the slot ends. The main body of the slot, also called primitive solid slot, corresponds to the volume swept by the cross section of the cutter along a path, while the slot ends partially correspond to the volume occupied by a cutter placed at the extremities of the sweep trajectory. We take the similar idea when we utilize halfspace partition to deal with the volume construction for slot features with cylindrical end faces. The procedure consists of the following steps:

1. Extract all the planar faces \( \{ F^p \} \) from the surface feature \( N^s \) and call the trimming procedure \( T_{of}(\{ F^p \}) \) to construct a maximum feature volume \( V_{max}^p \);

2. Intersect \( V_{max}^p \) with \( HS^+(F^{ce}) \), the positive halfspace of the cylindrical end face \( F^{ce} \) to get a trimmed cylinder \( V^{ce} \), i.e. \( V^{ce} = V_{max}^p \cap HS^+(F^{ce}) \). \( V^{ce} \) corresponds to the volume of the slot end;

3. Along a smooth edge \( E^s \) of the cylindrical end face, create an imaginary face \( F^{im} \) that is perpendicular to the planar face \( F^{op} \) adjacent to the smooth edge. The normal vector \( n^{im} \) of \( F^{im} \) is pointing away from the cylindrical end face and can be decided by \( n^{op} \times n^{ce} \), where \( n^{op} \) is the normal vector of \( F^{op} \) and \( n^{ce} \) is the directional vector of the smooth edge’s coedge on \( F^{op} \);

4. Intersect \( V_{max}^p \) with \( HS^+(F^{im}) \), the positive halfspace of the imaginary face \( F^{im} \) to get a trimmed volume \( V^{sb} \), i.e. \( V^{sb} = V_{max}^p \cap HS^+(F^{im}) \). \( V^{sb} \) corresponds to the volume of the slot’s main body;

5. Unite \( V^{ce} \) and \( V^{sb} \) to get the final maximum feature volume \( N^r \), i.e. \( N^r = V^{ce} \cup V^{sb} \).

Please note that if the slot has two cylindrical end faces, the volumes for the two ends can be constructed by step 2 and then unite with the volume of the main body to form the final maximum feature volume. Figure 6.12 illustrates the above five steps.
Figure 6.12: Maximum feature volume creation for a slot with a cylindrical end face
6.3.1.5 Algorithm for Negative Volumetric Feature Extraction

After the construction of maximum feature volume for all the recognized basic negative surface features, subsumption relationship is checked to resolve the feature interaction problem. The complete algorithm for negative volumetric feature extraction is as follows.

**Input:** a part model \( T \), its stock volume \( S \), and the set of recognized basic negative surface features \( N^\prime \)

**Output:** the set of negative volumetric features \( N^\prime \) with feature interaction resolved.

1. Compute \( \Delta \) based on equation (6.4);
2. Compute the stock face set \( \beta^S(T) \) and non-stock face set \( \beta^\prime(T) \) based on equation (6.12) and (6.13);
3. (construction phase) For each basic negative surface feature \( N^\prime_i \) call the trimming procedure \( T_{nf}(N^\prime_i) \) to construct the maximum feature volume \( N^\prime_i \) and add it to \( N^\prime \);
4. (reorganization phase) Check the subsumption relationship between each \( N^\prime_i \) and all the other \( N^\prime_j, j \neq i \);
   
   If \( N^\prime_i \) is subsumed by a \( N^\prime_j \), i.e. \( N^\prime_i \subseteq N^\prime_j \), update \( N^\prime_j \) by merging \( N^\prime_i \) with \( N^\prime_j \) and mark \( N^\prime_i \) as a subsumed negative volumetric feature.
5. Remove all the subsumed negative volumetric features from \( N^\prime \);
6. Return the feature set \( N^\prime \).

6.3.2 Positive Volumetric Feature Extraction

As mentioned in Section 6.2.5, from the design viewpoint, a positive feature is added to another parent positive feature by adding some material. That means a base face is needed to attach a child positive feature to a parent positive feature. In this scenario, we proposed the concept of supporting face to facilitate the volume construction for positive surface features.
Let $F_i', i = 1 \cdots n$ represent a feature face of a positive surface feature, and a supporting face $F_j', j = 1 \cdots m$ is a face satisfying the following two conditions:

- $F_j'$ is an adjacent face along the outer edge loop of a feature face $F_i'$, and $F_j' \not\in \{F_i'\}$;
- All the non-shared feature faces are totally within the positive halfspace of $F_j'$.

The volume of a positive surface feature is the volume enclosed by all the feature faces and the supporting faces. The computation of the volume construction can be performed by the regularized Boolean intersection among the negative halfspaces of all the feature faces and the positive halfspaces of all the supporting faces as follows:

$$P^v = HS^-(F_1') \cap^* \cdots \cap^* HS^-(F_n') \cap^* HS^+(F_1') \cap^* \cdots \cap^* HS^+(F_m')$$

If no supporting face is found, such as the positive feature $P_1$ as shown in Figure 6.13, the computation of the volume construction for positive surface features can be simplified by the regularized Boolean intersections only among the negative halfspaces of all the feature faces as follows,

$$P^v = HS^-(F_1') \cap^* \cdots \cap^* HS^-(F_n')$$

As described in Section 6.3.1, an incremental trimming procedure is devised to construct volumes for negative surface features. For positive surface features, the volume is created in one time. Considering the completeness, the union of the constructed volumes of all the positive surface features may not cover the whole part. Thus it is necessary to do a further check. A pruning operation is proposed here to perform the needed verification. Once a positive volumetric feature is constructed, it is pruned from the part volume. Ideally after the volume construction of all the positive surface features and their corresponding pruning operations, the resulting part volume should be void (or an empty set), as shown in Figure 6.14. If any solid volume remains, it shows that the volume construction for all the positive surface feature is not complete. The remedy is to merge the remaining volumes with the constructed neighbor positive feature volumes.
Figure 6.13: Constructed positive volumetric features of the example part ANC-101
6.3.3 Updating of Type III Surface Feature

As discussed in 6.2.2, the enlarged stock volume will affect the distribution of stock faces and non-stock faces of the part volume. If the stock volume is larger than the bounding box of the part, some stock faces in the part boundary will be converted into non-stock faces after the re-computation of the delta volume. Those altered stock faces will form type III negative surface features. Therefore, the set of type III surface features will be updated by adding the new features from the converted non-stock faces. Once the new type III negative surface features are recognized, their corresponding maximum feature volumes can be extracted by applying the proposed trimming procedure $T_{nf}$. As illustrated in Figure 6.15, if the stock volume (bounding box) of the example part shown in Figure 6.7 is enlarged evenly by a small size along the two opposite directions of each
major axis, the six stock faces of the part volume will be converted into non-stock faces and each converted non-stock face will be identified as a type III negative surface feature. In addition, the corresponding maximum feature volume for each new type III surface feature is extracted by the trimming procedure.

![Figure 6.15: Updated type III surface feature and its maximum feature volume](image)

6.4 Examples

We have tested various parts including some popular parts in feature recognition community, such as NEW_DEMO_US and CAM-I ANC101. Volumetric feature extraction is performed after fillet/round suppressing and basic surface feature recognition. Appendix B and Appendix C give the extraction results (screen snapshot) of NEW_DEMO_US and CAM-I ANC101 respectively. In the volumetric feature extraction of these two example parts, the stock volume used is the bounding box of the part volume. The test results show that the proposed volumetric feature extraction can efficiently construct maximum feature volumes for various surface features and resolve
the feature interaction problem without the consideration of the number of features involved.

6.5 Summary

This chapter presents a methodology to extract feature volumes for the surface features recognized from the algorithms in Chapter 5. A theoretical foundation is provided to support the proposed strategies and devised algorithms.
CHAPTER 7

POST-RECOGNITION PROCESSING

The identified features from the previous chapters are independent of applications and manufacturing processes. Different applications and manufacturing processes may pose a variety of constraints on the feature types and feature sizes. In order to utilize the application-independent (or neutral) features, post-processing is needed to manipulate the neutral features to generate application-specific and process-specific parameterized feature shapes. This chapter explores some issues and techniques in post-processing neutral features for various applications such as design and machining.

7.1 Introduction

The extracted volumetric features by using the methods in Chapter 6 form a hybrid feature set including positive volumetric features and negative volumetric features, which is independent of applications and manufacturing processes. In order to utilize the hybrid feature set, application-related features need to be derived to support various applications. In this research, we will focus on the generation of machining feature and design feature. Machining features represent the actual material volumes removed from the stock after a process plan is decided. Thus their actual shapes are depended on the practical machining conditions. Based on the machining conditions, negative volumetric features can be sequenced and the actual volume of the machining features can be computed. In design application, positive features are more preferable since they can be used to quickly layout...
the preliminary shape of the part model. Negative features then can be applied in this coarse shape. Thus all the positive volumetric features and certain negative volumetric features can be selected from the hybrid feature set to form a design feature set. The derivation of machining feature and design feature will be discussed in the following.

7.2 Feature Selection for Various Applications

One basic intent of feature recognition is to provide a complete and correct feature interpretation that may be tailed to different applications and processes. This section presents our preliminary ideas in feature selection for various applications including design intent recovery and machining. More efforts need be spent in the future research to devise a systematic methodology for utilizing the neutral features in multiple applications.

7.2.1 Design Intent Recovery

By extensive observation, we realized that the features that are most natural for use during the design phase are not machining features. A designer working through a conceptual design and attempting to create a detailed geometry does not think in terms of machining operations. Rather, design features are often better defined in terms of function, shape, and form. Especially, additive shapes and forms are chosen first to layout the preliminary outer shape of the product. This preliminary shape is further modified by applying subtractive shapes and forms. For example, a designer is more preferable to add a rib $D_2$ onto a based block $D_1$ than to subtract two steps $N_1$ and $N_2$ from a stock volume $S$ as shown in Figure 7.1. Figure 7.2b illustrates that combining four positive design features $D_1$~$D_4$ can quickly layout the preliminary outer shape of CAM-I ANC101 (Figure 7.2a), which is obviously more intuitive and efficient than subtracting many negative features from a stock volume. Negative design features can be applied onto this preliminary shape to create a final part model.
Figure 7.1: A part with a rib or two steps

Figure 7.2: Preliminary outer shape in positive design features of CAM-I ANC101
The recovery of the design features of a part model is very important for the design optimization. Once design feature recovery is completed, we can easily fine-tune the design by editing the features such as modifying the dimensions or even remove the whole feature and add new features as desired. This requires that the design features recovered be fully editable, associative, and parametric. Figure 7.3 illustrates the recovered design feature tree of CAM-I ANC101 and a highlighted feature with its parameters. Therefore, the design intent recovery can enhance the value of legacy data and enable easier, more productive sharing of 3D models between different CAD systems.

Figure 7.3: Design feature tree of CAM-I ANC101
During the design intent recovery feature hierarchy must be created since it is an essential part of the representation of design features. It plays an important role in feature editing. When a feature is deleted, all its immediately associated child features and their further successors should be removed. Parent-child relationship among design features is the foundation for the building of feature hierarchy. As shown in Figure 7.2, design feature D₂ is created on D₁. Thus D₁ is the parent of D₂ and D₂ is a child of D₁. Similarly, D₃ and D₄ are the children of D₂. A partial feature hierarchy can be created for D₁~D₄ as shown in the right half of Figure 7.2. After the identification of the negative design features, a complete feature hierarchy can be built for the example part. Figure 7.4 illustrates the complete hierarchy of the design features of CAM-I ANC101.

7.2.2 Machining

The intrinsic of machining is to remove materials from a stock or initial workpiece to obtain the desired part. From property (1) of volumetric features discussed in Section 6.2.5, we know that the union of all the negative volumetric features can generate the delta volume, which is the set of all the removed materials from the stock. Obviously all the extracted negative volumetric features should be selected for the derivation of machining features. Since the negative volumetric features represent maximum feature volumes extracted based on surface features, a large amount of volumetric overlap can occur when many features are involved in feature interactions. The major post-processing of the negative volumetric features involves the derivation of multiple interpretations of machining features discussed in the following.

7.3 Multiple Interpretations of Machining Features

Since there can be more than one way to manufacture a design it follows that there may be many different alternative sets of features that transform an initial workpiece or stock into a final part. Early work on feature recognition focused on finding a single best
Figure 7.4: The complete hierarchy of the design features of CAM-I ANC101
feature decomposition for a given part. However, as the problem of feature alternatives has become better understood [Regli, 1995] there is general agreement that consideration of alternative feature interpretation is critical.

[Tseng and Joshi, 1994] argued that the multiple interpretations generated are dependent on the order in which the features are recognized. In fact, feature recognition is performed before the reasoning of the multiple feature interpretations. The multiple interpretations should be generated based on the manufacturing requirements such as the tool and set-up selection. Different manufacturing constraints can result in different sequences of the feature volumes, thus multiple interpretations of machining features. In cell-based volume decomposition method, [Sakurai, 1995] proposed to utilize heuristics to sequence the maximum convex cells obtained by cell composition and then generate different sets of machining features. Some heuristics applied in his method include the following: (1) select the set-up direction in which the largest number of features can be created; (2) in a set-up, select the largest cutter from the inventory that can machine a feature; (3) machine the features with the selected cutter in the descending order of their projected areas in the cutter axis direction. We believe that heuristic-based method is a feasible way to sequence the extracted negative volumetric features for the generation of multiple interpretations of machining features. Since the output negative volumetric features from our feature recognition system can provide the information such as parameters and accessible directions, the above-mentioned three heuristic rules can be applied to the set of maximal volumes of negative features to generate a sequence of machining features. Figure 7.5 illustrates a possible machining sequence for the part NEW_DEMO_US based on those three rules. Various heuristic rules based on different machining conditions can be applied to generate different machining sequences.

One of the key problems in multiple feature interpretation is how to maintain the consistent correspondence between the geometrical description of a volumetric feature and its related functional meaning (semantics) during the manipulation process.
Uncontrolled feature shape modification due to updating among interacted features can affect the correct semantics of volumetric features. Thus feature semantics classification and updating need to be studied.

Figure 7.5: A possible machining sequence for NEW_DEMO_US
7.4 Feature Semantics Classification

The feature semantics classification relies mainly on the geometric and topological characteristics of the comprising faces of a feature. When interactions occur and the face configuration and adjacency information is not present exactly in the same manner as in the template feature, the classification may be incorrect. Thus, the semantics classification method should be flexible enough to accommodate possible feature interactions. Therefore we proposed the concept of existent face to deal with the variations of features due to feature interactions and identify feature types with more tolerance of the geometry and topology.

As discussed in Section 6.2.3, the closed boundary of volumetric features can be divided into two portions: real boundary and virtual boundary. Real boundary is the portion that is overlapped with the boundary of the part volume, and virtual boundary is the portion that is not overlapped with the boundary of the part volume. Thus it is obvious that any face of a volumetric feature can consist of two portions: real and virtual. Let $\sigma'^r$ and $\sigma'^v$ denote the set operators for the real and virtual portions of a face $F$ respectively, then

$$F = \sigma'^r(F) \cup \sigma'^v(F)$$

Therefore, the real boundary of a volumetric feature is the union of the real portions of all the faces, i.e. $\beta'^r(N^r) = \bigcup \sigma'^r(F_i)$; the virtual boundary of a volumetric feature is the union of the virtual portions of all the faces, i.e. $\beta'^v(N^v) = \bigcup \sigma'^v(F_i)$. If the real portion of a face $F_i$ is not empty, i.e. $\sigma'^r(F_i) \neq \emptyset$, $F_i$ is defined as an existent face. As shown in Figure 7.6, faces $F_1$~$F_5$ are existent faces for the extracted volumetric slot feature. Since $\sigma'^r(F_6) = \emptyset$, feature face $F_6$ is not an existent face.

With the introduction of existent face, we can perform the semantics classification based on the geometry and adjacency information of existent faces of a volumetric feature. In this research, we utilize the Existent Face Adjacency Graph (EFAG) to represent features. Similar to the attributed adjacency graph (AAG) used in the popular graph-based method.
for feature recognition [Joshi and Chang, 1988], an EFAG is a graph whose nodes are represented by existent faces, and arcs are represented by the edges connecting existent faces. Certain attributes can be attached to nodes and arcs such as geometry and convexity. With the EFAG representation, graph isomorphism is employed to identify feature types. Two graphs $G_a = (V_a, E_a)$ and $G_b = (V_b, E_b)$ are isomorphic if there is a one-to-one mapping $f$, such that $f: V_a \rightarrow V_b : \forall (v_i, v_j) \in E_a, \exists (f(v_i), f(v_j)) \in E_b$. In other words, there is a correspondence between the nodes of $G_a$ and $G_b$ that preserves the arc relationships.
Let $G = (V, E)$ be a graph with $V = \{v_1, v_2, \ldots, v_n\}$. Then $G$ can also be represented by its adjacency matrix $M = (m_{ij}), i = 1, \ldots, n$. The adjacency matrix $M$ is an $n \times n$ matrix in which the entry $m_{ij} = 1$ if there is an edge from vertex $i$ to vertex $j$ and is 0 if there is no edge from vertex $i$ to vertex $j$. Clearly, the matrix $M$ is not unique for a graph $G$. If $M$ represents $G$, then any permutation of $M$ is also a valid representation of $G$.

Based on the study of the current available graph matching methods, the decision tree approach [Messmer, 1996] is chosen as our solution for graph matching. The conventional graph matching method can only be applied to two graphs at a time. Thus for applications that are dealing with a database of model graphs, a time complexity that is linearly dependent on the size of the database results. The decision tree approach is devised to deal with a set of model graphs. In the compact representation provided by a decision tree, subgraphs that are common to more than one model graph are represented only once. Consequently, at run time, the computational effort for finding these subgraphs in the input graph must be done only once. The time complexity of the decision tree method is completely independent of the number of the model graphs. However, the size of the decision tree is exponential in the number of vertices of the model graphs. Because the decision tree must be present in the main memory at run time, the application of this method is limited to small graphs. Therefore the decision tree approach is particular interest to applications where the underlying graphs are rather small, but where there is a potentially large number of model graphs and almost real time behavior is required.

The approach is based on the following idea. We assume that there is a set of model graphs that are known a prior and an input graph that becomes accessible at run time only. For each model graph we compute all possible permutations of its adjacency matrix and transform these adjacency matrices into a decision tree. The decision tree is built from the model graphs in an off-line preprocessing step. At run time, the matrix of the
input graph is then used to find those adjacency matrices in the decision tree that are identical to it. The permutation matrices that correspond to these adjacency matrices represent the graph isomorphisms that we are looking for.

One of the most obvious benefits of the decision tree approach is the extendibility. Once a new feature type is identified, its correspondent graph can be easily incorporated into the existent database of feature graphs. The extension can be achieved by recompiling the decision tree, as illustrated in Figure 7.7. The compilation of a decision tree for a model graph $G$ consists in generating all permutations of the adjacency matrix of $G$ and incorporating a classification path in the decision tree for each of these matrices. The actual compilation and traversal algorithm can be referred to [Messmer, 1996].

![Figure 7.7: Flow chart of semantics classification based on decision tree approach](image-url)
7.5 Parameterization of Feature Shapes

As we mentioned in Section 2.1.2, an implicit feature representation is desired to achieve a complete feature representation. This requires the feature shapes to be parameterized. A feature parameter is a dimensional variable whose value establishes the size and relationships of various geometric elements comprising a pre-defined set of geometric elements. For example, the elements of the hole include (1) the curved surface (negative cylindrical surface), (2) the edge, and (3) the included cone angle at the bottom of the hole. The parameters for the above include such things as hole diameter, hole depth, cone point angle, and edge definition. Edge definition parameters may include angle and depth of a chamfer, radius of a fillet, etc., in addition to the above parameters. The angle of the axis of the hole with respect to the surface of the workpiece and the positional coordinates are other important parameters. The hole may be normal to the surface, or at an angle to the surface. Thus, in order to perform feature parameterization, the sufficient parameter definition of each feature type must be identified. With the fully-defined parameters available, the parameterization can be performed.

Two types of implicit features were proposed by [Pratt and Wilson, 1985] for use with B-Rep modelers. The first is the modifier feature, which represents such local modifications as the filleting or rounding of an edge by the simple attachment of a label to that edge. Since all the fillets/rounds have been suppressed by the pre-processing as discussed in Chapter 4, the modifier features can be recovered by identifying all the edges with suppressed fillets/rounds. As shown in Figure 7.8, the highlighted edges are those with suppressed fillets/rounds for the example part CAM-I ANC101. The second type of implicit feature is the generic feature, which allows a large class of rotational and prismatic features to be represented in terms of the volume swept out by the motion of a two-dimensional profile. Both the profile and its motion can be represented in terms of a comparatively small number of parameters. Generic features can be further classified into prismatic generics and rotational generics depending on whether the sweep of a 2D profile is translational or rotational. A prismatic generic is formed by sweeping a 2D
profile along a line, while a rotational generic is formed by sweeping a 2D profile around an axis. Generic features include many of the standard features such as holes, slots, pockets and bosses that can be described in terms of a small number of dimensional parameters. Figure 7.9 and Figure 7.10 show some extracted rotational generics and prismatic generics in volume representation for the example part CAM-I ANC101.

Figure 7.8: Modifier features suppressed by pre-processing in CAM-I ANC101
Figure 7.9: Some extracted rotational generics in CAM-I ANC101

Figure 7.10: Some extracted prismatic generics in CAM-I ANC101
Every generic feature is defined with a datum point to aid in its correct positioning with respect to the part; this point lies in the part surface before the feature can be 'fixed'. As illustrated in Figure 7.11a, the parameterization of a prismatic generic is accomplished by the following: (1) parameterization of its 2D profile (closed); (2) the sweep distance D; (3) the datum point (the profile starting point). The parameterization of the polygonal 2D profile is achieved by the following information: (1) a point (the starting point $O$ of the profile); (2) a direction (the direction vector $V_1$ of the first profile edge); (3) a sequence of lengths $L_i$ (of the sides of a polygon); (4) a sequence of angles $\theta_i$ (of the corners of the polygon). The edges are assumed to be directed; the positive sense of the first edge is in the specified direction away from the starting point, and the remainders are directed in the same sense around the polygon. The profile starting point $O$ is used as the datum point for the feature.

![Parameterization of prismatic generics and rotational generics](Image)

Figure 7.11: Parameterization of prismatic generics and rotational generics
The parameterization of a rotational generic is accomplished by the following: (1) parameterization of its 2D profile (not closed); (2) the rotation axis; (3) the datum point. As illustrated in Figure 7.11b, the parameterization of the 2D profile for a rotational generic is achieved by the following information: (1) a starting point; (2) a sequence of numbers representing either lengths (of profile edges parallel to the axis) or angles (of profile edges not parallel to the axis); (3) a sequence of diameters (associated with the ends of the profile edges).

We observed that the extracted feature volumes might not be regular due to the constraint of stock volume. As shown in Figure 7.12, the extracted feature volume of the blind slot is not a regular volume. Thus the parameterization will be a problem. One possible solution is to construct the canonical volume for the irregular feature volume. For the blind slot shown in Figure 7.12, the canonical feature volume can be computed in the maximal bounding box along the symmetric axis of the slot. Once the canonical volume can be computed for the irregular feature volume, the parameterization can be performed based on the parameter definition of the feature type.

![Blind slot, Irregular feature volume, Canonical feature volume](image)

Figure 7.12: Irregular feature shape of a blind slot and its canonical volume

168
7.6 Other Applications of AFR

As mentioned in Chapter 3, AFR is also useful for inspection planning besides design and machining as discussed above. One obvious application is for part localization for inspection planning. As shown in Figure 7.13a, 3D data points of an example part can be measured by a computer vision system. These 3D data points can be further processed by the information aggregation module [Shen et al, 1999] to obtain certain geometric features such as circles and polygons as shown in Figure 7.13b. The extracted form features from the CAD model by our AFR (shown in Figure 7.13c) can be matched with...
those geometric features based on the shape and parameter information. Once the match
is achieved, the transformation between part coordinate system and machine coordinate
system can be obtained automatically. As shown in Figure 7.13d, the part coordinate
system is successfully aligned with the CMM machine coordinate system after part
localization. For dimensional inspection planning using CMM, the Automated Inspection
Planning System (AIPS) [Menq et al., 1992] can be applied to generate a collision free
inspection path as shown in Figure 7.13d.

7.7 Summary

This chapter explores some issues and techniques in post-processing neutral features for
various applications such as design and machining. Feature selection, semantics
classification and parameterization are discussed in greater details.
CHAPTER 8

IMPLEMENTATION: A PROOF-OF-CONCEPT PROTOTYPE SYSTEM FEATURE EXPLORER

This chapter gives an overview of the implementation of a proof-of-concept prototype system Feature Explorer developed with the methodology described in Chapter 4 through Chapter 7. Various parts have been tested and the recognition results of several popular industry parts are given.

8.1 System Architecture

In this research, a feature recognition prototype system, named Feature Explorer, has been built (see Figure 8.1) based on the proposed feature reasoning kernel. ACIS, a current leading geometric modeling engine in object-oriented structure [Spatial, 1998a], is used to provide necessary solid modeling operations and attribute management mechanism, and OpenGL visualization toolkit [Woo et al., 1997] is utilized to supply fast 3D wireframe and shading display for feature information investigation. The input of the system is a B-Rep model and the output feature model will consist of explicit and implicit representations so as to provide as much information as possible for downstream applications. A windows-based graphical user interface (GUI), as shown in the top part of Figure 8.1, has been designed for the prototype system to allow the users to visualize the feature recognition process, interrogate the information of the recognized features such as
Figure 8.1: System architecture of Feature Explorer - the proof-of-concept prototype system
feature boundary, volume and parameters, and investigate the bi-attributes of all the
topological entities including faces, edges and vertices that the features consist of. With
such information in mind, the user can activate the feature browser from the GUI to list
all the identified features and expand the subtree under the interested feature to check its
comprising faces, edges and vertices and their bi-attributes and its parameters. The
prototype system can be viewed as a wrapper of the integration of ACIS geometric solid
modeling kernel and our feature reasoning kernel by incorporating the functionality of
user interface and visualization.

The feature descriptions of the output feature model from the prototype system will
contain the feature parameter set, local coordinate frame and hierarchical graph
references, as well as a feature identification such as feature name, feature number. Some
constraints may also be applied to parameters, their values to ensure the semantic
correctness of parameters during feature modification.

8.2 Software Foundations Utilized

All the source codes of the prototype system are written in C++ [Stroustrup, 1997], an
object-oriented programming language. Object-oriented programming can provide the
functionalities such encapsulation, inheritance and polymorphism, which are essential for
the provision of reusability and extensibility of a software project. During the
implementation of the prototype system, we utilized a number of software tools, such as
ACIS geometric modeling kernel, MFC library and OpenGL visualization kit.

**ACIS Geometric Modeling Kernel**

ACIS is an object-oriented 3D geometric modeling engine from Spatial Technology Inc.
It is designed for use as the geometry foundation within virtually any end user 3D
modeling application. ACIS is written in C++ and consists of a set of C++ classes
(including data and member functions, or methods) and API (Application Procedural
Interface) functions. A developer can use these classes and functions to create an end user 3D application. ACIS integrates wireframe, surface and solid modeling by allowing these alternative representations to coexist naturally in a unified data structure, which is implemented in a hierarchy of C++ classes.

**MFC Class Library**

MFC is a rich class library from Microsoft for efficient graphical user interface development on Windows systems [Blaszczak, 1997]. It can significantly reduce the effort to write an application for Windows by providing true Windows API functions and easy-to-use yet powerful abstractions of complicated features such as ActiveX, database support, printing, toolbars, and status bars. The AppWizard tool provides a good starting point to develop an application that has the “look and feel” expected by experienced users of the Windows environment. By extending the default behavior in derived classes, developer-specific functions can be added efficiently to enhance GUI.

**OpenGL Visualization Kit**

OpenGL visualization kit is a powerful software interface used to produce high-quality computer generated images and interactive applications using 2D and 3D objects and color bitmaps and images. It covers many basic computer graphics techniques such as building and rendering 3D models; interactively viewing objects from different perspective points; and using shading, lighting, and texturing effects for greater realism. In addition, it provides in-depth coverage of advanced techniques, including texture mapping, antialiasing, fog and atmospheric effects, NURBS, image processing, and more. The developed prototype system in this research fully utilized the wireframe and shading functionalities to visualize the recognized surface features, volumetric features and their constructive entities.
8.3 Feature Data Structure

It has been well recognized that the explicit feature representation consists of the surface and volume information. The feature type is decided by the surface features, i.e., the configurations of the real faces of the feature volume. The feature volume is used to facilitate the feature interaction resolving and feature parameterization. Thus the surface and volume information become the indispensable part of the feature data structure. In current research the surface and volume information of features are incorporated into the feature data structure to form the explicit feature representation. After the feature parameterization the implicit feature representation is also added into the feature data structure to provide a complete feature representation.

8.4 Examples

In order to verify the prototype system based on the proposed feature reasoning kernel, various parts from NIST part library are tested. The tested parts need to be complex enough to have various feature types and many feature interactions. Several popular industry parts are chosen such as ANC101 and NEW_DEMO_US and their results are displayed in Appendix B and Appendix C.
CHAPTER 9

CONCLUSIONS

9.1 Contributions

In this research we developed a general, reliable and robust automatic feature recognition method for multiple applications in concurrent engineering. The hybrid automatic feature recognition method consists of four functional modules: pre-processing, surface feature recognition, volumetric feature extraction and post-processing. The contributions of this research work can be identified as follows.

- More efficient and reliable feature recognition methodology by integrating surface feature recognition and volumetric feature extraction. In feature recognition, hint-based reasoning has been acknowledged as an effective approach for recognizing intersecting features [Han, 1996]. The problem is that the hints may be unreliable and inflexible when dealing with complex features. It is essential to identify the invariant properties of the B-Rep entities and features and use them to come up with reliable and flexible hint-based reasoning. We identified convexity and accessibility as the two major invariants that are playing an important role in the feature recognition with the capability of feature interaction resolving. Convexity is utilized to devise a bi-attribute based method for surface feature recognition from boundary entities. Since the bi-attribute based method is graph-free, it is more tolerant and flexible when dealing with topological variations. Accessibility is employed to build a half-space
partition based method for volumetric feature extraction. Maximum feature volumes are extracted for each recognized surface feature with the feature interaction resolved on the fly.

- Feature reasoning for multiple applications. Unlike most of the current research works that are limited to recognition of machining features, or negative features, or material removal volumes more generally. The proposed hybrid feature recognition methodology can recognize both positive and negative features in a part, which are rich enough to provide the information for various design and manufacturing applications in concurrent engineering.

- The proposed hybrid feature recognition techniques are integrated into a feature reasoning kernel. It is believed that the kernel architecture has great potential payoff in building a general, reliable and robust feature recognition system which can be used as a component in the next-generation CAD/CAM systems.

9.2 Recommendations for Future Work

To build a scalable and practical feature recognition system for real-world mechanical parts and integrate engineering design with downstream activities, some extensions to this research will be needed. In particular, here are the recommendations for future research:

- Efficient Underlying Modeling Space Based on Non-manifold Topology

Two different strategies have been traditionally proposed to model solid objects. In the first strategy, one aims to represent a solid through an explicit description of its boundary. These are the so-called boundary representations (B-Rep), which are based on data structures that describe the adjacency relationships of the topological entities such as vertices, edges, and faces of the solid. The other possibility is the CSG approach, which
consists of representing a solid as a result of a sequence of set operations performed on simple primitive solids. Traditional B-Rep and CSG techniques apply when one just needs to look at an object as including a three-part space decomposition: its interior, its exterior and its boundary. For feature-based modeling, not only the surface boundary of the features on the final part need to be computed, but also it is very important to get the volumetric occupation and the spatial relations of the features. In this situation, one would like to have, not only a representation for each feature of the part, but also a description about the way these features are connected to each other. In general, traditional modeling systems have a hard time modeling contact relationships between solids.

Cellular modeling seems to be an efficient underlying modeling scheme in recording and managing both volume and boundary information of multiple adjacent solids. Compared with classical B-Rep and CSG, a cellular approach offers several advantages such as locality of manipulation, efficient access to both geometric and topological information and the possibility of attaching tolerances to the various object parts. Unlike CSG, the primitive components in a cellular model are represented in a boundary data structure, and can be arbitrary manifold objects. Also unlike traditional B-Rep in which only one solid is considered, cellular model aims to deal with the representation and manipulation of multiple adjacent solids. Thus cellular models can provide a coherent integration between shapes of a feature and cells in the cellular model. Every feature shape has an explicit volumetric representation in terms of cells. Specific subsets of its boundary are also distinguished in terms of cell faces, edges and vertices. Therefore, the cellular modeling scheme has more flexibility and can carry more information than previous modeling methods such as B-Rep and CSG. We believe cellular model has the potential to contain all the information required for maintaining high-level feature semantics.

Although cellular models were used in the feature-based design recently [Bidarra, Kraker and Bronsvoort, 1998], no sufficient mathematical and theoretical basis has been proposed to systematically support the appropriate data structure and its operations.
Manifold topology, which is the topological foundation of traditional B-Rep, can not provide enough topological support for cellular model which involves the representation and manipulation of multiple adjacent solids. Non-manifold geometric modeling has been paid attention recently on the volume representation, maintenance, and editing of form-features. Several efforts have been spent in establishing the mathematical & theoretical foundation of non-manifold topology [Weiler, 1986; Gursoz et al., 1990; Masuda et al., 1990; Yamaguchi and Kimura, 1995]. From the published literature, the discussions on non-manifold geometric modeling focused on the modeling of entities with inhomogeneous dimensions, such as objects with mixed 1D, 2D and 3D entities. Volumetric feature representation requires a mutually exclusive, completely exhaustive, irregular cellular structure to model multiple adjacent 3D solid components, the selective collection of which forms the functional features. Thus non-manifold topology covers more general scope than that of volumetric feature representation. We believe generalized non-manifold topology can be refined to provide a topological structure more suitable for cellular model, and the generalized Euler-Poincare formula can be adjusted to support the necessary operations on the data structure.

- Reusable CAD Models by Feature-based Indexing

CAD databases are at the core of the modern digital engineering enterprise. The emerging huge web-based digital part libraries store all the part information over their lifecycle. Given the solid model of a new mechanical part which may be obtained from reverse engineering or other sources, an engineer might need to design plans for the manufacturing and inspection of this new part. There might be other parts similar to this new part in structure and behavior stored in the CAD repository. If a similar part can be found from the existent CAD databases, the engineer will be able to reuse the manufacturing information of the similar part, therefore engineering efficiency can be greatly improved. To achieve this goal, techniques for the intelligent storage and retrieval of solid models are needed. Since features can provide structural, hierarchical and behavioral information of the solid model, feature recognition can be used to generate
feature-based indexes that can be utilized to classify components and allow efficient retrieval of CAD models. Based on the hierarchical representation of part features, an effective similarity metric needs to be developed to measure the similarity between the feature-based CAD models of two mechanical parts so that the most similar case can be retrieved rapidly and correctly.

• **Real-world Mechanical Parts with Sculptured Surfaces**

Unlike the benchmark parts tested in academic research, the real-world mechanical parts can be very diverse and complicated due to the existence of sculptured surfaces. In order to push the academic feature recognition systems to the industrial level, certain approaches are required to extend the recognition scope into realistic mechanical parts. Since our proposed bi-attribute based surface feature recognition is established from the fundamental curvature computation of geometric entities, it can be smoothly applied on the sculptured surfaces. However, our devised volumetric feature extraction, which is based on halfspace partition, can not handle sculptured surfaces since halfspace does not exist for the free-form surface patches in solid models due to their non-divisibility of the 3D space into distinct subsets [Mantyla, 1988]. This means a new method has to be devised to handle parts with sculptured surfaces or a whole new method is invented to deal with the volume construction for all surface types including regular and sculptured ones. I believe the breakthrough in techniques for the conversion from B-Rep to CSG will help the extension of feature recognition to real-world parts with sculptured surfaces.

• **Integrated Feature-based Modeling**

While much work has been done both in feature-based design and automatic feature recognition from traditional CAD models, few attempts to integrate the two approaches have been made. Due to various feature interactions among so many features, the feature recognition result can not be guaranteed to be 100% correct. There might be some misinterpretations in the recognition result or some irregular portions of the part
boundary that cannot be recognized. Thus the interactive capability from feature-based
design can be utilized to re-interpret the misunderstandings or identify the non-
recognizable portions by users' observation from the computer display and interactive
operations with the highlighted portions that may have problems. On the other hand,
feature-based modeling alone may not be sufficient. In concurrent engineering
environment, multiple applications may share a common geometric database so that a
geometric change from one application can be propagated to other applications. Then, it
is necessary to support multiple views of the geometric database in terms of feature by
individual applications. With the current feature-based modeling technology, a geometric
database cannot be shared by multiple application domains because each feature-based
modeling in an application will create a distinct geometric model. On the other hand,
multiple views of a single geometric database can be supported by feature
mapping/conversion from automatic feature recognition. What's more, almost all the
current available feature-based design systems don't have automatic feature semantics
updating capability. The feature semantics are either too abstract due to its solid modeling
operations or are altered due to feature interactions. In order to ensure the correct
semantics of output feature-based models, automatic feature recognition can be applied to
further detect and update the semantics of those features affected by feature
modifications. Therefore, it is suggested that feature-based design and automatic feature
recognition be combined to achieve an integrated feature-based modeling environment in
concurrent engineering.

With integrated feature-based modeling it will be more easily to handle the CAD model
with a large feature set of hundreds or even thousands. One of the obvious advantages is
to allow rapid local modification of any large feature-based CAD model. The typical way
to deal with the feature modifications in current available commercial feature-based
design systems is to re-evaluate all the procedural Boolean operations from the
modification point or just from the very beginning of the design. Such a processing
method works well for CAD models without lots of features. But for the CAD models
with a large feature set, the re-evaluation will be very time-consuming and severely affect
the efficiency of the feature-based design systems. By integrating feature recognition capability into the feature-based design systems, the evaluation after each modification is performed locally, i.e., updating the modified feature and all the features that interact with it. Therefore only a very small portion of the feature set is involved in the re-evaluation, thus design efficiency can be greatly improved.
BIBLIOGRAPHY


Pratt, M.J. and Wilson, P.R., “Requirements for Support of Form Features in Solid Modeling System”, CAM-I Report R85-ASPP-01, 1985


**APPENDIX A**

All the final feature types in the proposed feature taxonomy based on EAD concept and CAM-I classification are illustrated as follows:

<table>
<thead>
<tr>
<th>Feature Type</th>
<th>Illustration</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cylindrical Boss (EAD = 0)</td>
<td><img src="image" alt="Cylindrical Boss Illustration" /></td>
<td>Circular cross-section</td>
</tr>
<tr>
<td>Symmetric Boss (EAD = 0)</td>
<td><img src="image" alt="Symmetric Boss Illustration" /></td>
<td>Symmetric cross-section: Square, rectangular, partial obround, complete obround, diamond-shaped, triangle-shaped, other symmetric such as multigon-shaped (pentagon, hexagon, octagon and etc.)</td>
</tr>
<tr>
<td>Non-symmetric Boss (EAD = 0)</td>
<td><img src="image" alt="Non-symmetric Boss Illustration" /></td>
<td>Non-symmetric cross-section</td>
</tr>
<tr>
<td>Description</td>
<td>Image</td>
<td>Details</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>--------------------------------------------</td>
<td>----------------------------------------------</td>
</tr>
<tr>
<td>Symmetric Pocket (EAD = 1)</td>
<td><img src="image" alt="Symmetric Pocket" /></td>
<td>Symmetric cross-section except obround</td>
</tr>
<tr>
<td>Non-symmetric Pocket (EAD = 1)</td>
<td><img src="image" alt="Non-symmetric Pocket" /></td>
<td>Non-symmetric cross-section</td>
</tr>
<tr>
<td>Round Blind Hole (EAD = 1)</td>
<td><img src="image" alt="Round Blind Hole" /></td>
<td>Circular cross-section</td>
</tr>
<tr>
<td>Internal Blind Slot (EAD = 1)</td>
<td><img src="image" alt="Internal Blind Slot" /></td>
<td>Obround cross-section</td>
</tr>
<tr>
<td>Round Through Hole (EAD = 2na)</td>
<td><img src="image" alt="Round Through Hole" /></td>
<td>Circular cross-section</td>
</tr>
<tr>
<td>Symmetric Through Hole (EAD = 2na)</td>
<td><img src="image" alt="Symmetric Through Hole" /></td>
<td>Symmetric cross-section except obround</td>
</tr>
</tbody>
</table>

"na" means the two entry faces are non-adjacent.
<table>
<thead>
<tr>
<th>Internal Through Slot (EAD = 2na)</th>
<th>Obround cross-section</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non-symmetric Through Hole (EAD = 2na)</td>
<td>Non-symmetric cross-section</td>
</tr>
<tr>
<td>Symmetric Blind Slot (EAD = 2a)</td>
<td>“a” means the two entry faces are adjacent</td>
</tr>
<tr>
<td>Non-symmetric Blind Slot (EAD = 2a)</td>
<td></td>
</tr>
<tr>
<td>Through Slot (EAD = 3)</td>
<td></td>
</tr>
<tr>
<td>Notch (EAD = 3)</td>
<td></td>
</tr>
<tr>
<td>Step</td>
<td><img src="image1" alt="Diagram" /></td>
</tr>
<tr>
<td>--------</td>
<td>--------------------</td>
</tr>
<tr>
<td>(EAD = 4)</td>
<td>Step and Step removed</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Face</th>
<th><img src="image2" alt="Diagram" /></th>
</tr>
</thead>
<tbody>
<tr>
<td>(EAD = 5)</td>
<td>The top block in dotted line represents a removed slab. The highlighted face represents a face feature corresponding to the removed slab.</td>
</tr>
</tbody>
</table>

| Fillets & Round | ![Diagram](image3) |

| Chamfer | ![Diagram](image4) |
APPENDIX B

This appendix gives the AFR results for the test part NEW_DEMO_US. All the related data in AFR for NEW_DEMO_US is shown as follows:

<table>
<thead>
<tr>
<th>Simplified B-Rep Model</th>
<th>Stock</th>
<th>Delta Volume (2 views)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non-stock Faces</td>
<td>Stock Faces</td>
<td>Maximum Feature Volume Set</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Surface Feature Set</th>
<th>Features with Suppressed Edge Blendings</th>
</tr>
</thead>
</table>

199
All the extracted negative features are shown as follows:

<table>
<thead>
<tr>
<th>Basic Surface Feature</th>
<th>Maximum Feature Volume</th>
<th>Maximum Feature Volume (isolated view)</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>10 Holes (8 Blind, 2 Through)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 Blind Hole</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 Blind Hole</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2 Internal Slots (Two Cylindrical End Faces)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 Pocket (Regular)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 Pocket (Regular)</td>
</tr>
<tr>
<td>1 Pocket (Regular)</td>
<td>1 Pocket (Regular)</td>
<td>1 Pocket (Regular)</td>
<td>1 Pocket (Regular)</td>
</tr>
<tr>
<td>-------------------</td>
<td>-------------------</td>
<td>-------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td><img src="image1" alt="Diagram" /></td>
<td><img src="image2" alt="Diagram" /></td>
<td><img src="image3" alt="Diagram" /></td>
<td><img src="image4" alt="Diagram" /></td>
</tr>
<tr>
<td>2 Blind Slots</td>
<td>2 Blind Slots</td>
<td>2 Blind Slots</td>
<td>2 Blind Slots</td>
</tr>
<tr>
<td><img src="image5" alt="Diagram" /></td>
<td><img src="image6" alt="Diagram" /></td>
<td><img src="image7" alt="Diagram" /></td>
<td><img src="image8" alt="Diagram" /></td>
</tr>
<tr>
<td>2 Steps</td>
<td>2 Steps</td>
<td>2 Steps</td>
<td>2 Steps</td>
</tr>
<tr>
<td><img src="image9" alt="Diagram" /></td>
<td><img src="image10" alt="Diagram" /></td>
<td><img src="image11" alt="Diagram" /></td>
<td><img src="image12" alt="Diagram" /></td>
</tr>
</tbody>
</table>
APPENDIX C

This appendix gives the AFR results for the test part CAM-I ANC101. All the related data in AFR for CAM-I ANC101 is shown as follows:

<table>
<thead>
<tr>
<th>Original B-Rep Model</th>
<th>Simplified B-Rep Model</th>
<th>Stock</th>
<th>Delta Volume</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non-stock Faces</td>
<td>Stock Faces</td>
<td>Surface Feature Set</td>
<td>Maximum Feature Volume Set</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Features with Suppressed Edge Blending</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
All the extracted negative features are shown as follows:

<table>
<thead>
<tr>
<th>Basic Surface Feature</th>
<th>Maximum Feature Volume</th>
<th>Maximum Feature Volume (isolated view)</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>19 Holes (5 Through, 14 Blind)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3 Blind Slots with Cylindrical End Face</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4 Symmetric Pockets</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2 Notches</td>
</tr>
</tbody>
</table>

19 Holes (5 Through, 14 Blind)
3 Blind Slots with Cylindrical End Face
4 Symmetric Pockets
2 Notches
All the extracted positive features are shown as follows:

```
<table>
<thead>
<tr>
<th>2 Steps</th>
<th>2 Steps</th>
<th>1 Surface Feature</th>
<th>1 Surface Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Image" /></td>
<td><img src="image2.png" alt="Image" /></td>
<td><img src="image3.png" alt="Image" /></td>
<td><img src="image4.png" alt="Image" /></td>
</tr>
<tr>
<td><img src="image5.png" alt="Image" /></td>
<td><img src="image6.png" alt="Image" /></td>
<td><img src="image7.png" alt="Image" /></td>
<td><img src="image8.png" alt="Image" /></td>
</tr>
<tr>
<td><img src="image9.png" alt="Image" /></td>
<td><img src="image10.png" alt="Image" /></td>
<td><img src="image11.png" alt="Image" /></td>
<td><img src="image12.png" alt="Image" /></td>
</tr>
<tr>
<td><img src="image13.png" alt="Image" /></td>
<td><img src="image14.png" alt="Image" /></td>
<td><img src="image15.png" alt="Image" /></td>
<td><img src="image16.png" alt="Image" /></td>
</tr>
</tbody>
</table>
```

204