A Survey of Automated Feature Recognition Techniques

by W.C. Regli

TECHNICAL RESEARCH REPORT



Supported by the National Science Foundation Engineering Research Center Program (NSFD CD 8803012), Industry and the University

A Survey of Automated Feature Recognition Techniques*

William C. Regli

The Department of Computer Science and Systems Research Center The University of Maryland College Park, MD 20742

February 12, 1992

Abstract

Automated feature recognition has been attempted through many methodologies for a wide range of application domains. This survey focuses on its use for geometric reasoning problems in mechanical engineering. Many of these methods are greatly limited in scope. Often they perform on a restricted class of objects with confining feature definitions. Furthermore, problems with interactions between features can render objects unrecognizable. This survey presents an overview of many of the works in this area. Included are descriptions of the approaches and an analysis of their abilities to provide a definition for and solution to the general problem of recognizing features from a solid model. It is hoped that previous research will provide the guidance for the development of a feature recognition system that is complete over a mathematically definable set of objects.

^{*}Funding provided by the National Science Foundation through their support of the Systems Research Center

1 Introduction

Automated feature recognition is the identification of objects of manufacturing interest in a computer representation of an mechanical part. This survey covers various techiques for feature recognition in the domain of mechanical engineering and specifically with the design and manufacture of mechanical parts in a concurrent engineering [Eng91] environment. Concurrent engineering is a philosophy that has evolved from the desire to improve the efficiency of the design process. The decisions made by a designer developing a new product have major reprocussions on quality, cost, and time. A concurrent engineering environment provides active feedback about the manufacturability of the design during the creative process and promotes robust design [Eng91]. A key component in such a system is feature recognition.

An automated feature recognition program extracts instances of manufacturing features from an engineering design, such as in figure 1, done with a computer aided design (CAD) system or solid modeler¹. The feature description then can be used for a variety of purposes including part classi-

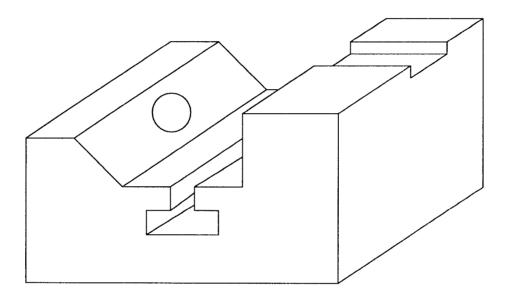


Figure 1: A design of a solid object

fication and generating process plans for manufacturing. The definition of a manufacturing feature varies widely depending on the approach and application domain [SSR+88]. For most cases, a manufacturing feature is a geometric, topological, or volumetric entity with manufacturing significance. For example, features are usually defined to be manufacturing entities. As in figure 2, a "hole" can be viewed as a cylindrical surface with certain properties. Factors such as the location of the hole or its angle can have a significant effect not only on the ability to recognize a particular hole, but also on the number of features needed to describe holes in general. When feature definitions attempt to create rigid links between volumetric or geometric shape and a manufacturing operation, quite often there will be many ways to manufacture the same shape. When two or more features interact, there may exist multiple feature interpretations for the object. In many cases the approaches operate in restrictive domains and their capabilities are inadequate for handling anything that could be called a realistic object.

A computer system for the generation and manipulation of 3-dimensional objects. See also [Hof89, Man88, Mor85].

```
Radius: R Axis Direction: \bar{\mathbf{A}} = (a_x \ a_y \ a_z)

Axis Origin: \bar{\mathbf{P}} = (p_x \ p_y \ p_z)

Local Z-Axis: \bar{\mathbf{Z}} = \bar{\mathbf{A}}_{\underline{\mathbf{P}}}

Local Y-Axis: \bar{\mathbf{Y}} = \frac{\bar{\mathbf{P}}}{\|\bar{\mathbf{P}}\|}

Local X-Axis: \bar{\mathbf{X}} = \bar{\mathbf{Y}} \times \bar{\mathbf{Z}}

Local & Semi-Infinite Accessibility: -\bar{\mathbf{A}} or +\bar{\mathbf{A}} or both
```

- 1. A Hole begins with an entrance face.
- 2. All subsequent faces of the hole share a common axis.
- 3. All faces of the hole are sequentially adjacent.
- 4. The hole terminates with a valid hole bottom.

Figure 2: Definitions for a cylindrical hole from [Van90] and [Hen84].

This survey of feature recognition research includes descriptions of the approaches and an analysis of their abilities to provide a definition for and solution to the general problem of recognizing features from a solid model. It is hoped that previous research will provide the guidance for the development of a feature recognition system that is complete over a mathematically definable set of objects.

2 Survey

Creating a survey of the field of automated feature recognition presents difficulty because attempts span a wide variety of applications. Research goals, application domain, and technique vary greatly over the works in the field and leave few bases for comparisons between methods. Motivations for feature recognition include classification of parts for group technology, generation of paths for numerically controlled machining, and creation of process plans for part manufacture.

These research projects were groundbreaking attempts to use computer technology in the mechanical engineering domain with much of their focus on computer aided manufacturing applications. Our review of these works focuses what each contributes to the development of a general paradigm for the feature recognition problem. In most of these cases, the research goals were dictated by the specific project and did not require the development of universal mathematical formalisms. This research is interested what issues are commonly found in existing work and what they reveal about development of a mathematically formal approach to a general problem of feature recognition. This analysis of previous work is biased in to that end.

We classify these approaches based on the computer science techniques they employ. This section presents an overview, by no means complete, of many existing approaches. For another summary of recent work [Kar90].

2.1 Syntactic Pattern Recognition

Extracting features from a 3-dimensional computer aided design model can be viewed as a problem in pattern recognition. Syntactic pattern recognition [Fu82] uses structural information to create a description or a classification of the subject we want to reason about. This approach to pattern recognition has widespread use in vision applications (picture recognition, scene analysis, classification of pictorial patterns), speech recognition, natural language processing, and recognition of written characters.

A variation or extension of a context free grammar [HU79, LP81] forms the core of a syntactic pattern recognition algorithm. In such a grammar, the terminal symbols usually represent a primitive element of the application domain. For automated feature recognition, a primitive can be an edge or face—these being among the fundamental building blocks of every feature. The grammar

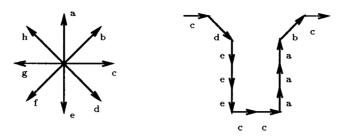


Figure 3: The pattern primitives and parse of a chamfered hole from [AHS83].

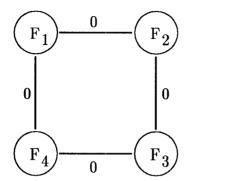
also contains rules that determine how these fundamental primitives may be combined. There may be a rule $S \longrightarrow \alpha$ that generates a specific type of feature or a rule $T \longrightarrow \gamma$ that generates a class of similar edge contours. The scene is then parsed—much like a compiler parses a computer language [ASU86]. During parsing the structural information, such as the existence of depression in the object, embodied in the rules can be exploited. Utilizing these techniques, algorithms can recognize features and classify the shape of a solid.

Luc Kypraniou, in his doctoral dissertation [Kyp80], used syntactic pattern recognition to code parts for group technology classification. He developed an algorithm that identified basic types of protrusions and depressions in a computer aided design model and from them inferred a classification for the global shape of the object. The algorithm used a grammar based on adjacency relationships of faces, convexity and concavity of edges, and existence of edge loops to parse the object. Specific feature structures could be looked for during the parse and, based on the types of features found, a classification for the part could be generated.

Kypraniou's objectives did not require that set of classifiable objects be clearly defined and it is unclear what happens when presented with an unclassifiable object. The method, while restricted by grammar, has difficulty parsing objects where the necessary structural information is lost due to interactions among feature types. Kypraniou's thesis was one of the first in the area of automated feature recognition and it has served as a fundamental reference in later works.

Ryszard Jakubowski [Jak82, Jak85], using edges as pattern primitives, generated group technology part codes for mechanical parts based on their 2-dimensional cross-sections. This approach is concerned only with mechanical part classification and is described for a restricted set of $2\frac{1}{2}$ -dimensional parts. Information is obtained through parsing the edge primitives in the silhouette of the part. This information and basic manufacturing data are used to categorize the part. In Jakubowski's work there is much grammatical formality and little mention of features. Amoung the practical limitations, for example, the approach addresses only $2\frac{1}{2}$ -dimensional parts. Also, the primitives used in this study limit the complexity of a silhouette because there are only a finite number of 2-dimensional edges and rules in the grammar. These limitations and that the method is presented as a formalism without demonstrated computational viability make this approach impractical for classification of complex parts and automated feature recognition.

In [SLF85], an approach similar to Jakubowski's is investigated. The goals of the work are shape classification for group technology process selection and representation of the volume that must be machined to create the object. Grammars for generating shape families are presented. These grammars, like those in [Jak82, Jak85], describe the outline of the part. This work built on Jakubowski's, but many of the same limitations still exist: the class of shapes is limited and it does not allow for the reasoning about geometry and topology necessary for automated feature recognition.



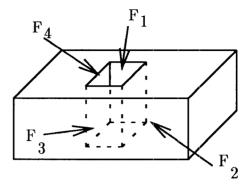


Figure 4: The attributed adjacency graph representation for a rectangular hole from [JC88].

[AHS83] use a grammar to classify holes based on how they are manufactured. A 2-dimensional cross-section is obtained from a 3-dimensional computer aided design database. The grammar is then used to extract the holes from the cross-section, as illustrated in figure 3. The grammar classifies the hole based on several hole families. The limitations in this approach are representative of the limits of all feature recognition systems based on syntactic pattern recognition techniques. The eight pattern primitives in the grammar are capable of representing small subset of the angles that could exist in a hole. In addition, there could be ambiguity in the 2-dimensional cross-section caused by features other than holes (this problem could be eliminated by assuming that a hole is the only type of feature). Ambiguity could also arise from interactions between holes—what if two perpendicular holes intersect? What if holes are not perpendicular to the surface of the object? These limitations do not diminish the significance of the work, which is its contribution to the ability to communicate with a geometric database on a feature level. However, this communication consists of a small set of very basic questions and about an extremely restricted set of objects.

2.2 Graph-based Methods

Graph-based feature recognition systems represent the geometric and topological information about an object using graph structure. Recognition of features can be defined as a search or parse of the graphical structure. An immediate advantage of graph-based systems is the theoretical foundation of graph theory [Har69]. This affords the opportunity to exploit the language of graph theory to define the problem of feature recognition and draw on the multifarious computer science algorithms that deal with searching, traversing, parsing, or matching graphs [AHU83, BB88, PB85].

2.2.1 Searching

Joshi and Chang [JC88] present a graphical structure called an attributed adjacency graph. Their goal is to improve machine understanding of design to facilitate the automation of process planning. The attributed adjacency graph is built from the information contained in the boundary representation of the solid model. Nodes in the graph represent faces of the object and arcs in the graph denote edges of the object. Each arc has an attribute indicating the concavity or convexity of the edge it represents. Figure 4 provides an example of the attributed adjacency graph representation of a rectangular hole.

An elegant aspect of this approach is that it facilitates clear definitions of the types of 3dimensional objects and features it can handle. It also presents the feature recognition algorithm as a subgraph matching problem. However, while stating the algorithm and the classes of objects and features, this approach's limits become evident. There are only six types of features in this approach and the object class is restricted to polyhedral objects. The feature recognition algorithm involves heuristic improvements to the NP-hard subgraph isomorphism problem. Attempts are made to handle feature interactions using rules for two specific types of interaction. These two types of interactions cover many, but not all, of the interactions possible for the six types of recognizable features. These drawbacks emphasize the difficulty in developing an elegant and practical approach to automated feature recognition.

Leila De Floriani [De 89] introduces a graphical structure called a generalized edge-face adjacency graph for recognition of features such as protrusions, depressions, through holes, and handles (see figure 5). In this scheme, the features and the object are represented using the generalized face-edge adjacency graph in such a way that the features form biconnected and triconnected graphs. Hence, the problem of feature extraction becomes searching a graph for the biconnected and triconnected components and classifying them as one of the feature types.

This approach uses graph theory and established, polynomial-time, algorithms to produce a formal feature recognition system. The recognition algorithm avoids the NP-hard pitfalls of the Joshi and Chang approach because the features are all defined as biconnected and triconnected components. Unfortunately, not all desirable features can be described as a biconnected or triconnected component in the generalized face-edge adjacency graph. The approach ignores most geometric information about faces, placing little emphasis on shape and location of faces, not to mention how this approach would fare on shapes with nonplanar faces, nonperpendicular edges, and complex feature interactions. The algorithm for recognizing the features defines the type of features that will be recognizable—those that form unique components in the generalized face-edge adjacency graph. These restrictions are too limiting for this method to be applied to more complex features and shapes. Other work from the same research group [FG89] uses an approach similar to this to generate a hierarchical feature structure to represent objects at higher levels of abstraction.

Sakurai and Gossard [SG90] developed a system for recognition of user-defined features based on a similar graph search strategy. An important contribution of this work is that it emphasized the separation of the feature definitions and the feature recognition algorithm. Because features are often domain specific, Sakurai and Gossard advocate a procedure for the user to specify those features that are important to the application. The features and the object are then represented with some variation of an attributed adjacency graph. The process of feature recognition is redefined to a graph search problem. This method is very similar to Joshi and Chang's [JC88] and shares all its drawbacks. However, unlike Joshi and Chang, no attempt is made to characterize and deal with feature interactions. In fact, the approach characterizes features as collections of surfaces and then "fills in" the volumes when they are recognized. This act of "filling in" the feature volumes can make what remains of a valid object unrecognizable within the available feature set. Since filling in one feature will inadvertently fill in part of any other feature that shares its volume. A feature might require the existence of that shared volume—hence "filling in" that mutually required volume would make other feature unrecognizable.

2.2.2 Graph Grammars

Research performed at the Engineering Design Research Center and the Robotics Institute at Carnegie Mellon University [PFP89, SF90] has been directed toward using a graph grammar to parse a graphical representation of an object. They define objects to be elements in a language generated by augmented topology graph grammar. The advantage of their graphical structure is that it contains both geometric and topological information (unlike [De 89, FG89, JC88, SG90]). This graphical structure, as in figure 6, can also be used to define general features (a point stressed in [SG90]). A grammar can be defined to describe the class of objects generatable with a specific set of features. To recognize the features, the grammar is used to parse the object. This approach

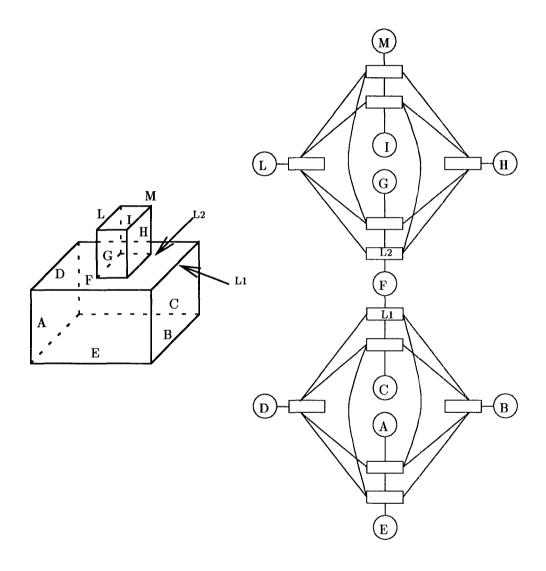


Figure 5: The generalized edge-face graph representation for an object from [De 89].

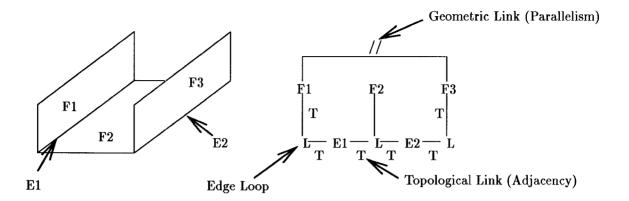


Figure 6: The representation of a slot using the augmented topology grammar from [PFP89, SF90].

is more efficient than those of [JC88, SG90] because it relys on parsing a graph grammar instead of searching a graph structure. The grammar is used to produce the object in much the same way that a computer language compiler uses a grammar to determine if the program it has been given is in the computer language [ASU86]. As the object is parsed, information about the features can be produced.

The method has many elegant points, among them the ability to define the needed features and the use of a formal graph grammar for describing a language of shapes. However, the use of a graph grammar to parse an object creates its own computational complexity because of the pattern searching required for parsing. Here too, feature interactions are ignored. This not only adds to the computational burden of additional feature instances in the object, but also might render objects unrecognizable. As for the types of objects that are recognizable, the current scope of the work has been limited to injection molding.

2.3 Expert Systems

Expert systems are used to automate deductive reasoning tasks requiring expert knowledge. The idea is to encode the knowledge and experience of an expert into rules. Given a set of facts, the rules can be used to reason about the domain. A mechanical engineer analyzes a solid object to determine how to manufacture it; therefore, feature recognition can be viewed as creation of an expert system to reason about the manufacturability of a solid object.

Mark Henderson, in his 1984 thesis [Hen84], created an expert system in Prolog to perform feature recognition. In this system, rules are used to define feature instances. For example, a hole might be defined as "a cylindrical surface with an open top and open bottom." The feature rules become a part of an expert system that interacts with the computer aided design database. The process of feature recognition is performed by applying the rules to the database and letting Prolog's theorem prover determine feature instances. This approach was the first to employ an expert system to perform feature recognition.

An important aspect of Henderson's system is the attempt to create general rules. The idea was that the basic concept of "hole" remains unchanged despite the many, possibly infinite, number of specific instances. The system contained definitions for classes of $2\frac{1}{2}$ -dimensional features including holes, slots, and pockets.

However, as the approach is detailed in the thesis, no attempt is made to deal with the feature

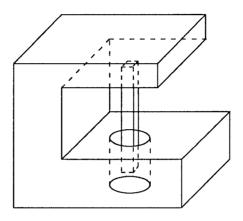


Figure 7: A particularly nasty interaction

interactions and the system is restricted to only $2\frac{1}{2}$ -dimensional parts. The Prolog rules defining the features rigidly specify each class of feature—hence each new feature type requires adding rules. Expert systems have proven effective tools for creating systems with large feature vocabularies. As these vocabularies grow, so does the time needed for Prolog to perform the searching. In order for expert system based methods to deal with feature interactions, rules need to be given to specify all types of interactions—including interactions with feature classes and interactions between specific instances (as in figure 7). Expert systems perform well with limited sets of features and possible interactions but require large amounts of special-case information to be feasible for more realistic applications.

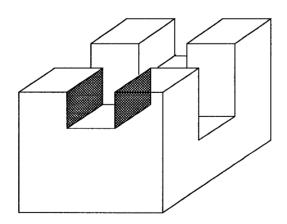


Figure 8: A hint for a retangular slot from [Van90].

2.4 Knowledge Based

Research performed by Xin Dong [Don88] pursued the development of feature recognition by frame-based reasoning to alow computer aided process planning from a computer aided design system. To address this problem, he presented a prototype frame-based knowledge representation scheme for features and parts, along with a rudimentary feature description language. With the representations, algorithms were developed to translate the solid model into the frame representation and recognize a set of eleven feature types from four feature classes. These eleven feature types are explicit manufacturing features that can be employed directly in the process planning system. The significance of this work lies not as much in its contributions as a feature recognition system, rather in its recognizing of features adequate for interpretation with a process planner. The frame-based approach is novel, however it remains unclear how feature interactions would affect the systems ability to find adequate feature information to facilitate process planning.

Jan Vandenbrande's thesis [Van90] advocates a sophisticated approach to feature recognition involving a variety of artificial intelligence techniques. The goal of the work is to generate a feature description of an object that satisfies rigid machinability requirements. The primitive features can be combined into composite features that have manufacturing significance. Features are built by searching the object for hints. A hint for a type of feature can be thought of as a byproduct of the feature being in the object. For instance, a cylindrical surface can be a hint for the existence of a hole; a slot hint may be two parallel surfaces, as shown in figure 8. If the recognition algorithm finds a cylindrical surface, it may indicate the existence of a hole. The recognizer collects all the hints that exist in the object, categorizes them based on how promising they are, and uses a rule-based approach to build a set of features compatible with the hints. This approach also allows the recognizer to deal with feature interactions in a more general manner in that hints deal with the features on a more abstract level; thus interactions can be dealt with based on the hints. The approach also produces alternate feature interpretations.

This method benefits from dealing with the features at a higher level of abstraction via hints. However, the hints are actually special cases themselves. To add more features you need more hints and more rules for combining the hints. The approach is also limited to a small set of $2\frac{1}{2}$ -dimensional features—no provision is made for other alternatives.

3 Conclusions

Feature recognition in mechanical engineering emerges as a problem of how to reason about shape. In these previous attempts there do not exist clear definitions for the terms that describe feature recognition. Without a precise statement of the problem it is impossible to judge the effectiveness of any algorithm.

One may argue that for engineers "producing a manufacturing feature description of an object for machining" serves as an adequate definition. This cannot be so, for there is no clear definition of a "manufacturing feature" [SSR+88]. Theoretically, there are infinitely many—just design a new tool with a previously unheard of shape. The existing techniques attempt to solve specific instances of the feature recognition problem. Those with rigid feature sets have difficulties dealing with the large numbers of heuristics and rules needed to define each feature and deal with their interactions. This tends to create ad hoc solutions with algorithms that are collections of special cases to deal with each feature and interaction. To add a new feature, all the appropriate special cases must be added to the algorithm to handle recognition under all possible interactions. In figure 9, for example, a system may have definitions for holes, recesses, and rectangular holes but may fail because of the way they interact. Those approaches that have attempted to allow for general or user-defined features have been more successful; but they cannot handle recognition for complex objects. The inability to recognize "real" objects is a drawback to all of the approaches; it especially applies to those with general and user-defined features.

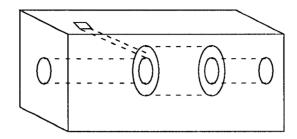


Figure 9: A object with a non-trivial interaction—possibly requiring additional feature definitions.

Computational complexity is a problem for all of the approaches. As with many other problems in computational intelligence, the computer time grows tremendously when dealing with realistic situations. The expense required to maintain the rules or perform the needed searches can be difficult or impossible to manage. This problem will hinder the ability to handle complex or multiple objects. Because these will eventually be desirable to represent, it is important to formalize matters of complexity. Existing methods make mention of it but rarely attempt to classify it. Feature recognition, even in an abstract form, is computationally expensive. A formalization of the feature recognition problem will provide a means to explore how computationally expensive the problem is and where the expense occurs. Reducing the expense is desirable but the nature of the problem suggests that even the best cases will be costly.

References

- [AHS83] D. C. Anderson, M. R. Henderson, and S. M. Staley. Using syntactic pattern recognition to extract feature information from a solid geometric data base. Computers in Mechanical Engineering, September 1983.
- [AHU83] Alfred V. Aho, John E. Hopcroft, and Jeffery D. Ullman. Data Structures and Algorithms. Addison-Wesley Publishing Company, MA, 1983.
- [ASU86] Alfred Aho, Ravi Sethi, and Jeffery Ullman. Compilers: Principles, Techniques, and Tools. Addison-Wesley Publishing Company, MA, 1986.
- [BB88] Gilles Brassard and Paul Bratley. Algorithms: Theory and Practice. Prentice Hall Incorporated, NJ, 1988.
- [De 89] Leila De Floriani. Feature extraction from boundry models of three-dimensional objects. IEEE Transactions on Pattern Analysis and Machine Intelligence, 11(8), August 1989.
- [Don88] Xin Dong. Geometric Feature Extraction for Computer-Aided Process Planning. PhD thesis, Rensselaer Polytechnic Institute, Troy, NY, USA, 1988.
- [Eng91] Concurrent Engineering. Special report. IEEE Spectrum, 28(7), July 1991.
- [FG89] Bianca Falcidieno and Franca Giannini. Automatic recognition and representation of shape-based features in a geometric modeling system. Computer Vision, Graphics, and Image Processing, 48:93-123, 1989.

- [Fu82] K. S. Fu. Syntactic Pattern Recognition and Applications. Prentice Hall Incorporated, NJ, 1982.
- [Har69] Frank Harary. Graph Theory. Addison-Wesley, Reading, MA, 1969.
- [Hen84] Mark R. Henderson. Extraction of Feature Information from Three-Dimensional CAD Data. PhD thesis, Purdue University, West Lafayette, IN, USA, 1984.
- [Hof89] Christoph M. Hoffman. Geometric and Solid Modeling: An Introduction. Morgan Kaufmann Publishers Incorporated, CA, 1989.
- [HU79] John E. Hopcroft and Jeffery D. Ullman. Intoduction to Automata Theory, Languages, and Computation. Addison-Wesley Publishing Company, MA, 1979.
- [Jak82] Ryszard Jakubowski. Syntactic characterization of machine parts shapes. Cybernetics and Systems: An International Journal, 13(1):1-24, 1982.
- [Jak85] Ryszard Jakubowski. Extraction of shape features for syntactic recognition of mechanical parts. *IEEE Transactions on Systems, Man and Cybernetics*, SMC-15(5), September/October 1985.
- [JC88] S. Joshi and T. C. Chang. Graph-based heurisites for recognition of machined features from a 3D solid model. *Computer-Aided Design*, 20(2):58-66, March 1988.
- [Kar90] Raghu R. Karinthi. An Algebraic Approach to Feature Interactions. PhD thesis, The University of Maryland, College Park, MD, USA, 1990.
- [Kyp80] Lycourgos K. Kyprianou. Shape Classification in Computer Aided Design. PhD thesis, Christ College, University of Cambridge, Cambridge, United Kingdom, 1980.
- [LP81] Harry R. Lewis and Christos H. Papadimitriou. Elements of the Theory of Computation. Prentice Hall Incorporated, NJ, 1981.
- [Man88] M. Mantyla. An Introduction to Solid Modeling. Computer Science Press, College Park, MD, 1988.
- [Mor85] Michael E. Mortenson. Geometric Modeling. John Wiley & Sons, New York, NY, 1985.
- [PB85] Paul W. Purdom and Cynthia A. Brown. The Analysis of Algorithms. Holt, Rinehart, and Winston, New York, NY, 1985.
- [PFP89] J. Miguel Pinilla, Susan Finger, and Friedrich B. Prinz. Shape feature description using an augmented topology graph grammar. In Proceedings NSF Engineering Design Research Conference, pages 285-300. National Science Foundation, June 1989.
- [SF90] Scott A. Safier and Susan Finger. Parsing features in solid geometric models. In Submitted to: European Conference on Artificial Intelligence, 1990.
- [SG90] Hiroshi Sakurai and David C. Gossard. Recognizing shape features in solid models. *IEEE Computer Graphics & Applications*, September 1990.
- [SLF85] R. Srinivasan, R. C. Liu, and K. S. Fu. Extraction of manufacturing details from geometric models. Computers & Industrial Engineering, 9(2):125-133, 1985.
- [SSR+88] Jami Shah, Palat Sreevalsan, Mary Rogers, Rick Billo, and Abraham Mathew. Current status of features technology. Technical Report R-88-GM-04.1, Computer Aided Manufacturing-International, Arlington, TX, USA, November 1988.
- [Van90] Jan H. Vandenbrande. Automatic Recognition of Machinable Features in Solid Models. PhD thesis, University of Rochester, Rochester, NY, USA, 1990.

		·	
·			