TECHNICAL RESEARCH REPORT

Recognition of Volumetric Features from CAD Models: Problem Formalization and Algorithms

by W.C. Regli, D.S. Nau

T.R. 93-41

Sponsored by
the National Science Foundation
Engineering Research Center Program,
the University of Maryland,
Harvard University,
and Industry

Recognition of Volumetric Features from CAD Models: Problem Formalization and Algorithms *

William C. Regli† and Dana S. Nau‡

Department of Computer Science and
Institute for Systems Research
University of Maryland
College Park, MD 20742 USA

Abstract

Automated recognition of features from CAD models has been attempted for a wide range of application domains in mechanical engineering. However, the absence of a clear mathematical formalism for the problem has made it difficult to develop a general approach—and thus most of these methods are limited in scope.

In this paper, we develop a formalization of the problem of recognizing a class of machinable features expressed as MRSEVs (a PDES/STEP library of machining features) [19], and an algorithm for solving this problem. Some of the characteristics of this approach are:

- the algorithm handles a large variety of hole and pocket features features along with elementary accessibility constraints and blends for those features;
- it is provably complete, even if the features intersect with each other in arbitrarily complex ways;
- it has $O(n^4)$ worst-case time complexity.

1 Introduction

Although many approaches have been developed for recognizing machinable features in solid models of mechanical parts, the absence of a clear mathematical formalism for the problem has made it difficult to develop a general approach—and thus most of these methods are limited in scope. It is often unclear what specific classes of objects, features, and feature interactions can be handled by various approaches, making it difficult to evaluate their overall utility.

As a first step toward addressing this difficulty, we have developed a formalization of the problem of recognizing a subset of the set of all machinable features expressible as MRSEVs (Material Removal Shape Element Volumes) [19]. MRSEVs are volumetric features corresponding to machining operations on 3-axis milling machines. MRSEVs can be defined using EXPRESS (the official PDES information modeling language) and PDES form features. Kramer [19] has already done this for a subset of the MRSEV library, and has defined the rest using an EXPRESS-like language.

Based on this formalization, we have developed a algorithm for solving the problem of recognizing every solid that can be described as the difference between an arbitrary piece of stock and

*This work was supported in part by NSF Grants NSFD CDR-88003012 and DDM-9201779.

†Email: regli@cs.umd.edu.

‡Email: nau@cs.umd.edu.

an arbitrary set of machinable features. The features in our class include a large variety of hole and pocket features, along with some elementary accessibility constraints for those features. The algorithm is provably complete over the set of all solids in our class, even if the features intersect with each other in arbitrarily complex ways. For example, our algorithm can handle each of the objects in Figs. 3 and 5. In addition, the recognition algorithm has a worst-case time complexity of $O(n^4)$ where n is the size of the solid model.

This paper describes our formalism and feature recognition algorithm, with an analysis of its complexity. Section 2 describes how this work relates to other research. Section 3 defines the class of MRSEVs that we are interested in, how they are used to generate descriptions of mechanical parts, and the feature recognition problem for this domain. Section 4 presents procedures for recognizing individual features in this domain. Section 5 derives theorems guaranteeing recognizability; builds an algorithm for finding feature models of parts; and proves its completeness and its ability to handle arbitrarily complex feature interactions. Section 6 presents complexity results on the algorithm, and Section 7 provides examples of how the algorithm would operate on various parts. Section 8 gives conclusions and future directions for work in this area. The proofs and details of procedures are in appendix A.

2 Related Work

The graph algorithm approaches of [6, 14] provide an excellent level of computational formality. However, while they have known algorithmic properties, they appear difficult to extend to realistic manufacturing problems. Additionally, graph-based methods and the graph grammars of [26, 32] are prone to combinatorial difficulties [25]. The recent work in [9] describes recognition techniques that attempt to combat the combinatorial problems by abstracting an approximation of the geometric and topological information in a solid model and finding features in the approximation.

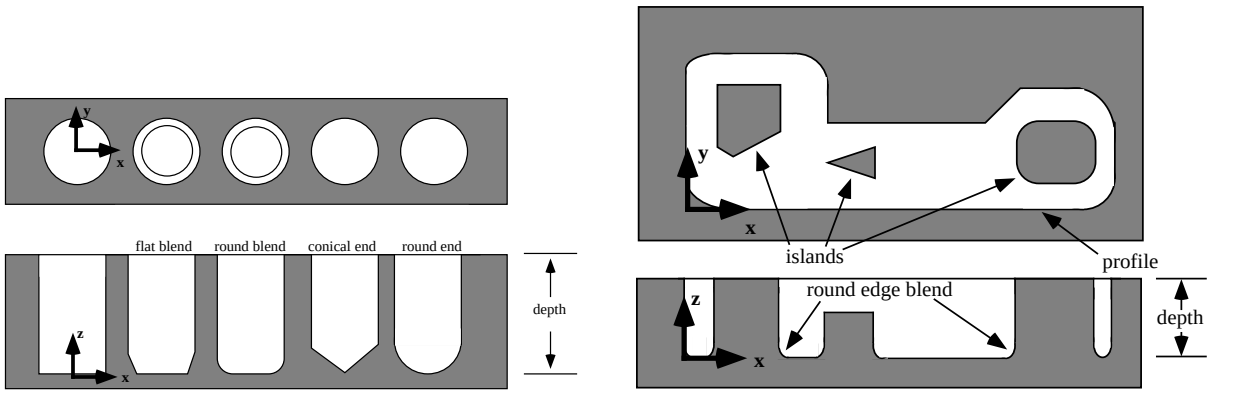
The feature interaction problem has been the focus of numerous research efforts, notably the heuristic approaches of [14, 34]. In [15, 16], an algebra of features is developed for the computation of alternate feature interpretations for parts. The work of [7] included the formalization of a feature description language and employed frame-based reasoning algorithms to extract machining features for computer aided process planning. [33] illustrates the need for extracting user defined features types that may arise in specific applications. Each of these goals would benefit ¿from a general feature recognition formalism.

The work of Henderson [4, 10] was seminal in employing expert systems on the feature recognition problem. Large expert systems, such as [3] for part coding, have practical applicability but do not present a framework for their feature recognition domain. In this case, there may be no formal means of specifying the capabilities of the system due to the subjective nature of the part coding problem. Kyprianou [20] presents an early effort to use grammars to parse solid models of parts for group coding.

Perhaps the most comprehensive and formal approach to date has been attempted by Vandenbrande [34]. This method provides a computationally rigorous way of recognizing a class of realistic manufacturing features via artificial intelligence techniques in combination with queries to a solid modeler.[1] The work stopped short of proving the completeness of the approach and, while providing techniques for handling interacting features, does not formalize the complete class of interactions within its capabilities; arbitrarily complex feature interactions may pose problems.

An aggressive approach to handle feature interactions and intersections was done by Marefat [21]. The work built on the representation scheme of [14] and used a novel combination of expert system and hypothesis testing techniques to extract surface features from polyhedral ob-

[1] A more detailed outline of this method can be found in [31].



Attribute of hole	Type
location	point
orientation	vector
depth	positive real number
radius	positive real number
end	end condition

Attribute of pocket	Type
location	point
orientation	vector
depth	positive real number
profile	edge loop
bottom blend	edge blend
islands	set of one or more islands
Attribute of island	**Type**
profile	edge loop
bottom blend	optional edge blend
height	positive real number

(a) (b)

Figure 1: Definitions for MRSEV Holes and MRSEV Pockets with islands.

jects. While providing insights into the feature interaction problem, the method has the same combinatorial difficulties of [14] and the added costs of an expert system to handle interactions. Additionally, although the method is complete for its domain, it is difficult to envision how it would extend to more complex objects and feature classes (such as those involving curved surfaces) and how completeness would be still be possible.

Other recent work includes feature recognition from 2D engineering drawings [22], via neural network techniques [27], and using convex volume decompositions [17].

The bibliography for complexity results specific to solid modeling operations is not extensive. Existing work contains empirical and some theoretical analysis of the time and space costs for various data structures for representing solids. Woo [36] analyzes several types of boundary representation data structures and compares their time costs for a set of primitive operations and space requirements. Weiler [35] presents data structures for curved surfaces and with their time and storage complexities. Ala [1, 2] builds on this work and introduces variations on the boundary representation data structure with advantages for certain application. Extensions to face-based representations are introduced in [8] and algorithms for their manipulation are analyzed. An excellent source of worst-case complexity analysis for boolean operations on boundary data structures is Hoffmann [11]. Peters [25] illustrates some of the combinatorial difficulties inherent in many graph-theoretic approaches to the feature recognition problem. In the feature recognition literature, [6] presents an analysis of the complexity of her methodology. Other attempts to measure performance include timing results, most notably in [3, 20]. Results of this type are highly dependent on the hardware, software implementation, the domain of interest, and the particular test cases chosen for the timing tests. Hence they represent a weak basis for comparisons between feature recognition methodologies.

3 Definitions

A *solid object* is a manifold r-set [30] with planar, cylindrical, toroidal, conical and spherical bounding surfaces. These are the only surfaces present in MRSEVs defined below, hence this set contains any object that they describe.

The set of features that we will consider in this paper is based on the library of material removal shape element volumes (MRSEVs), which was developed by Kramer [19] as a means of categorizing the shapes of volumes to be removed by machining operations on a 3-axis machine tool. MRSEVs are volumetric features, some of the benefits of which have been explained in [28]. The MRSEV hierarchy provides a framework for describing a large class of volumes of interest to machining.

Kramer's primary MRSEV types include linear swept features, edge-cut features, ramps, and rotational pockets. For the purpose of this paper we confine our domain to the linear swept features, i.e., holes, pockets, and pockets with islands. Kramer defines linear swept feature as a shape resulting from sweeping a closed profile of edges along a straight line perpendicular to the plane of the profile.[2] Figs. 1 (a) and 1 (b) present our illustrations of pocket and hole MRSEVs.

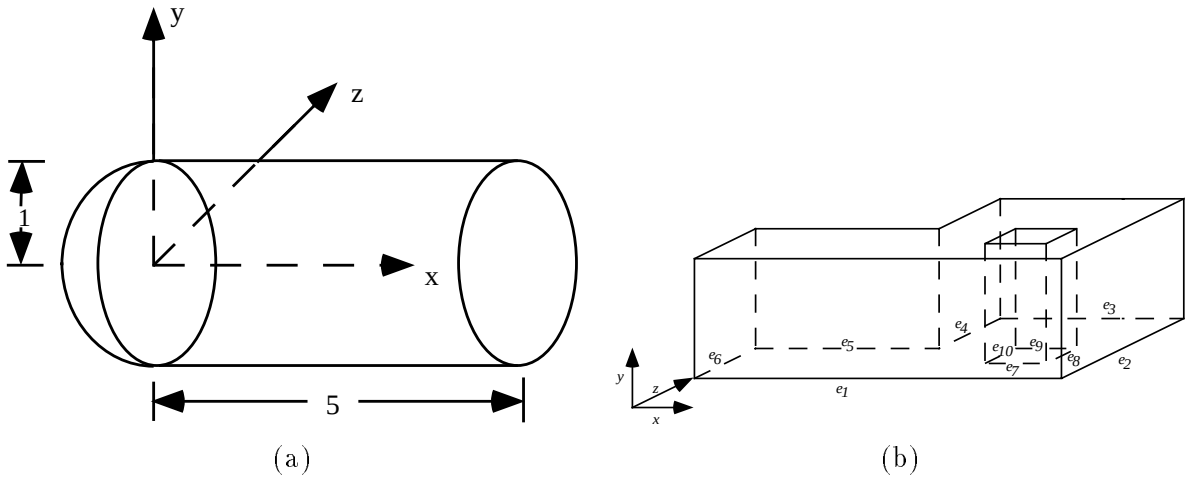


Figure 2: Instances of a hole and a pocket.

The MRSEVs are parameterized solids; a *feature* is a specific instance of one of these MRSEVs, resulting from a specific choice of attribute values. For example, suppose we choose the following attribute values:

location	=	$(-1, 0, 0)$;
orientation	=	$(1, 0, 0)$;
depth	=	5;
radius	=	1;
end	=	round end.

This would define the round-bottomed hole illustrated in Fig. 2 (a). Geometrically, this hole consists of the point set $(S_1 \cap^* S_2 \cap^* S_3) \cup S_4$, where

$$\begin{aligned} S_1 &= \{(x, y, z) : x \leq 5\}; \\ S_2 &= \{(x, y, z) : y^2 + z^2 \leq 1\}; \\ S_3 &= \{(x, y, z) : x \geq 0\}; \end{aligned}$$

[2] In the case of a pocket with islands, an island is considered to be a subfeature defined by its own closed profile.

$$S_4 = \{(x, y, z) : x^2 + y^2 + z^2 \leq 1\}.$$

Similarly, consider the following values for the attributes of a MRSEV pocket:

location	=	$(0, 0, 1)$;
orientation	=	$(0, 1, 0)$;
depth	=	2;
profile	=	$\{e_1, e_2, e_3, e_4, e_5, e_6\}$;
bottom blend	=	$\emptyset$;
islands	=	$\{I_1\}$;
profile I_1	=	$\{e_7, e_8, e_9, e_{10}\}$;
bottom blend I_1	=	$\emptyset$;
height I_1	=	2.

This would define a pocket with a single island as pictured in Fig. 2 (b). Geometrically, this pocket consists of the point set $(S_5 \cup S_6) -^* S_7$, where:

$$\begin{aligned} S_5 &= \{(x, y, z) : 1 \leq x \leq 4, 0 \leq y \leq 1, 1 \leq z \leq 3\}; \\ S_6 &= \{(x, y, z) : 4 \leq x \leq 7, 0 \leq y \leq 1, 1 \leq z \leq 5\}; \\ S_7 &= \{(x, y, z) : 5 \leq x \leq 6, 0 \leq y \leq 1, 3 \leq z \leq 4\}. \end{aligned}$$

The initial workpiece, $\mathbf{WP}_0$, is a solid object of raw stock material to be acted upon by a set of machining operations generating MRSEVs. The machined part (or just *part*) is a solid object **part** produced as a result of subtracting a finite set F of MRSEVs from an initial workpiece (hence $\mathbf{part} \subseteq \mathbf{WP}_0$). The *delta volume* is the regularized[3] difference of the initial workpiece and the part: $\Delta = \mathbf{WP}_0 -^* \mathbf{part}$.

Given a **part** and an initial workpiece $\mathbf{WP}_0$, we assume that the solid objects are bounded. Let **min** be the a value (or dimension) such that given any point in $\mathbf{WP}_0$ and a vector, a line of length **min** centered at the point and oriented on the trajectory of the vector would extend beyond the minimum enclosing sphere of $\mathbf{WP}_0$ in both directions. For example, given a hole of depth **min** with its center located within $\mathbf{WP}_0$, neither end-face of the hole is contained in $\mathbf{WP}_0$.[4]

Each feature has conditions that must be met in order for it to be *accessible*.[5] A hole is *accessible* if the circular cross-section of the hole can be swept in at least one of the two directions perpendicular to it a distance **min** without intersecting the **part**. The condition for a pocket involves a sweep of the **profile** and is similar.

A *feature model* of an initial workpiece $\mathbf{WP}_0$ and a **part** is a set of accessible features $FM = \{M_1, M_2, M_3, \ldots, M_n\}$ such that

$$\begin{aligned} &i. \quad \forall M_i \in FM, M_i \cap^* \mathbf{part} = \emptyset \\ &ii. \quad \mathbf{WP}_0 -^* \mathbf{part} \subseteq \bigcup FM \end{aligned}$$

We say FM is a feature model of **part** and $\mathbf{WP}_0$. There may be many feature models of **part** and $\mathbf{WP}_0$. FM is an element of this class of feature models of **part** and $\mathbf{WP}_0$—it *describes* Δ as a set of features.

[3] See [11] for the definitions of the regularized boolean operations.

[4] The value of **min** tries to capture the idea that each element in our class of objects is bounded. Boundedness implies that there exists an uncountably infinite number of upper bounds on the size of a given object. Because defining a least upper bound may present many degenerative special cases, for the purposes of this paper, **min** represents a reasonable upper bound that is not minimal.

[5] The MRSEV definitions from [19] do not include associated accessibility volumes. The potential benefits of such a development is mentioned in [24].

A feature M is *recognizable* in Δ if it is part of a feature model that describes Δ, and there exists a computable method of recognizing M from Δ.

The feature recognition problem is defined as follows:

Definition: Feature Recognition
INPUT: **part**, $\mathbf{WP}_0$
OUTPUT: return a feature model FM of $\mathbf{WP}_0$ and **part** if one exists;
$\emptyset$ otherwise.

We define a feature recognition algorithm to be *complete* if it returns a feature model of **part** and $\mathbf{WP}_0$ whenever they are describable by a feature model.

In this paper, we will only consider parts that satisfy the following restrictions:

- for any hole in F, a subface of its cylindrical face or its complete ending surfaces are present in Δ;
- for any pocket in F, either a subface of its bottom face is present in Δ, or else it is a through pocket with a corner radius or at least two of its non-parallel planar side faces present in Δ;

These restrictions provide the minimum conditions for recognizability. For example, consider the restrictions placed on the MRSEV pockets. Assume we have a **part** and a $\mathbf{WP}_0$ for which there exists a feature model containing an instance of a pocket. If this MRSEV pocket instance is a through pocket whose only remnant is a single planar side face there would be no tractable means of determining its orientation and location—with no other face from which the obtain a hint, there would be an uncountably infinite number of possible orientations. It is granted that there may exist other kinds of feature hints such as faces of the part blocking some of the possibilities. However no amount of information is going to remove all of possibilities and, in the worst case, there will still be a huge number of possibilities that must be explored by often ad hoc heuristics.

Additionally, we must assume the existence of constraints on the size of certain features; in particular, threshold values on the dimensions of surfaces that are identified as blends or holes. A *bottom blend* is basically a transition surface between a pocket's sides and its bottom. Though the geometric definitions may allow us to classify a portion of a large hole as a blend, it would not be realistic to do so. Let **max_blend** be the maximum radius or, for flat blends, width that a blend may have. For example, any cylindrical surface with a radius greater than **max_blend** must be described as an instance of a MRSEV hole.

4 Recognition of MRSEV Instances

The conditions for recognizability of each MRSEV feature type are formulated in the following lemmas. Note that in presenting these lemmas, we outline the cases for a general feature recognition algorithm for this class of MRSEVs. The proofs of these lemmas can be found in appendix A.

For any given feature instance in Δ, these lemmas show that in our domain as defined in section 3, a method exists to determine the attributes of a feature that subsumes it.

Lemma 4.1 *Let M be a MRSEV hole having a subface of one of its faces as a face in Δ, then there exists a recognizable MRSEV hole M' such that $M \subseteq M'$.*

As stated in the definitions of section 3, pockets are of two types: through pockets and non-through pockets. The subsequent lemmas prove that for any feature instance in Δ a feature instance subsuming it can be extracted.

Lemma 4.2 *Let M be a MRSEV through pocket with subfaces of two or more non-parallel planar side faces or a subface of a corner radius as subfaces of the boundary of Δ, then there exists a set of recognizable MRSEV pockets $\mathcal{M}$ such that $M \subseteq \bigcup \mathcal{M}$.*

Lemma 4.3 *Let M be a MRSEV pocket having a subface of its bottom face as a subset of the boundary of Δ, then there exists a set of recognizable MRSEV pockets $\mathcal{M}$ such that $M \subseteq \bigcup \mathcal{M}$.*

The lemmas provide an outline for building procedures to recognize individual feature instances in Δ through queries to the solid modeling system. For a complete elaboration of the details of these procedures and the primitive procedures needed to construct them please see appendix A.

These procedures share a common parameter list of **part**, $\mathbf{WP}_0$, and a surface f in Δ. If, for example, the surface f belongs to an instance of a hole, the procedure FIND MRSEV HOLE INSTANCE will return an instance of a maximal hole that creates f. If there is no such hole then the empty set is returned. One situation in which this could happen is the case of an inaccessible cylindrical surface—no hole instance will be found when all possible holes have a non-empty intersection with the **part**.

Procedure 4.1 FIND MRSEV HOLE INSTANCE
INPUT: **part**, $\mathbf{WP}_0$, *a face f in* $\mathbf{WP}_0 -^*$ **part** *formed from an instance of a MRSEV hole.*
OUTPUT: *M, a MRSEV hole found from f.*

The class of pockets dealt with in lemma 4.3 may include bottom blends. The next procedure finds the set of bottom blend surfaces associated with the profile of a MRSEV pocket instance. To keep the features consistent with the definitions of [19], FIND BOTTOM BLENDS returns a set of profiles and blend surfaces. These profiles will be contain the original profiles and enclose any bottom blend that may be found.

Procedure 4.2 FIND BOTTOM BLENDS
INPUT: **part**, *a set of edge profiles P, and a vector v indicating the direction of the top of the pocket.*
OUTPUT: *a set $\mathcal{B} = \{\langle P', B\rangle\}$ where P' is a set of profiles, and B is a set $B = \{s\}$ where s is a blend surface for the pocket whose profile and islands are elements of P'.*

Finding a set of possible bottom blends is necessary because there may be situations where overlapping pockets share part of the same bottom face. Hence, the profiles of the maximal pocket instances may be incident to a set of different blend surfaces. FIND BOTTOM BLENDS reconciles the different blend surfaces by returning a set of maximal pocket profiles for each type and size of blend surface found. This allows us to maintain the condition of the maximality of recognized features while ensuring that each pocket has exactly one bottom blend.

Employing the routine for recognizing bottom blends, we give the procedure outlined by lemmas 4.2 and 4.3 for the recognition of maximal pocket instances. If the surface f is not a subface of any pocket side or pocket bottom then the procedure returns the empty set.

Procedure 4.3 FIND MRSEV POCKET INSTANCE
INPUT: **part**, $\mathbf{WP}_0$, *a face f in* $\mathbf{WP}_0 -^*$ **part** *formed from an instance of a MRSEV pocket.*
OUTPUT: *a set $\mathcal{M} = \{M'\}$ of instances of a MRSEV pockets found from f.*

5 Finding a Feature Model

The procedures of the previous section recognize instances of individual MRSEV features from a face created by them. The issue now becomes how the procedures can be used to built a feature model. If a feature model FM exists for a particular **part** and $\mathbf{WP}_0$ will the procedures built from the lemmas find an equivalent feature model? While there is no guarantee that the procedures will find the exactly the same FM, what the following theorem asserts is that *if* any feature model exists for **part** and $\mathbf{WP}_0$ an equivalent feature model will be found. As with the lemmas, the proofs for the theorem and its corollaries are located in appendix A.

Theorem 5.1 *Given any feature model of a* **part** *and* $\mathbf{WP}_0, FM$, *for all MRSEV features* $M \in FM$, *there exists a set of features* $\mathcal{M} = \{M'\}$ *where each* M' *is recognizable in* Δ *and* $M \subseteq \bigcup M$.

The theorem proves that if individual feature instances are recognizable then a feature model can be constructed. Hence an algorithm can be built that determines whether or not a feature model exists for a specific **part** and $\mathbf{WP}_0$. If a feature model does exist, then the algorithm returns an equivalent model. If no model exists, the algorithm will return the empty set.[6]

Algorithm 5.1 MRSEV Feature Recognition
INPUT: **part** *and* $\mathbf{WP}_0$ *for which a feature model exists*
OUTPUT: *a set* F, *a feature model of* **part** *and* $\mathbf{WP}_0$

$F = \emptyset$
  While $\Delta -^* \bigcup F \neq \emptyset$ and $\exists$ unmarked faces of $\mathbf{boundary}(\Delta -^* \bigcup F) \cap^* \mathbf{boundary}(\Delta)$
   do
    Pick an unmarked face f from the $\mathbf{boundary}(\Delta -^* \bigcup F) \cap^* \mathbf{boundary}(\Delta)$
    Mark face f
    if f is planar:
      $F = F \cup$ (Find MRSEV Pocket Instance($\mathbf{part}, \mathbf{WP}_0, f$))
    else
      if f is cylindrical:
      $F = F \cup \{$Find MRSEV Hole Instance($\mathbf{part}, \mathbf{WP}_0, f$)$\}$
      $F = F \cup$ (Find MRSEV Pocket Instance($\mathbf{part}, \mathbf{WP}_0, f$))
      else
        $F = F \cup \{$Find MRSEV Hole Instance($\mathbf{part}, \mathbf{WP}_0, f$)$\}$
      endif
    Remove redundant features: $N \in F \ni N \subseteq \bigcup(F - \{N\})$
  EndWhile

Our claim is that this algorithm is complete; that is it returns a feature model for every **part** and $\mathbf{WP}_0$ for which one exists. The proof of this is a consequence of theorem 5.1.

Corollary 5.1 *(*Completeness*) Suppose* **part** *and* $\mathbf{WP}_0$ *can be described by a feature model, then the algorithm* MRSEV Feature Recognition *returns a feature model of* **part** *and* $\mathbf{WP}_0$.

The corollary that the algorithm can find a feature model for any $\mathbf{WP}_0$ and **part** with arbitrarily complex feature intersections is similar. Note that the algorithm deals with features interactions in a general manner—that is to say there is not a set of feature-by-feature interaction cases and techniques employed by the algorithm.

[6] The fact that the algorithm halts is proven in section 6.

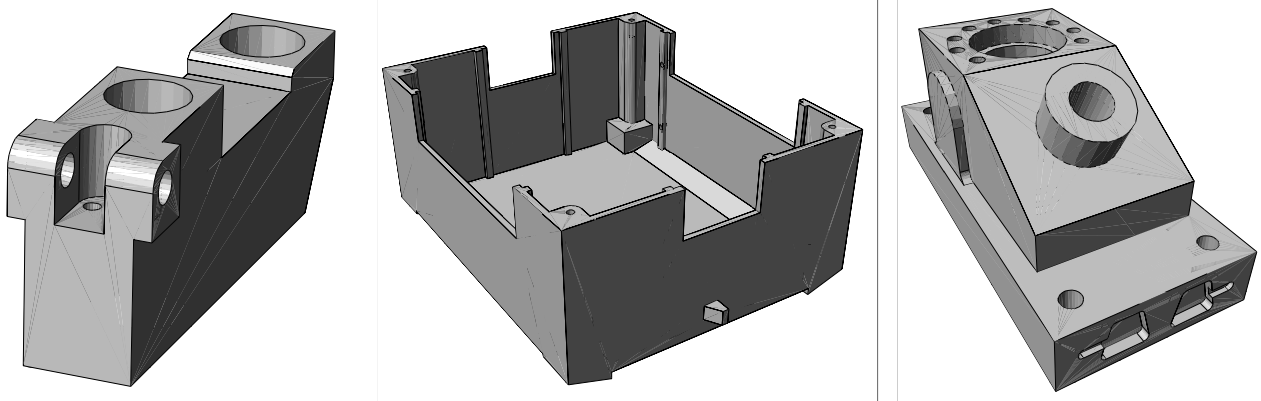


Figure 3: Examples parts from [12, 13] and a variant on the ANC101 part recognizable with this algorithm.

Corollary 5.2 *(*Feature Intersection Independence*) Suppose* **part** *and* $\mathbf{WP}_0$ *can be described by a feature model and for every such model the features intersect. The algorithm* MRSEV Feature Recognition *returns a feature model of* **part** *and* $\mathbf{WP}_0$.

Corollaries 5.1 and 5.2 show that algorithm 5.1 will extract a feature model for a given **part** and $\mathbf{WP}_0$ for which one exists regardless of how the features intersect. Hence algorithm 5.1 is complete over class of objects describable by MRSEV holes and pockets, independent of how the feature instances intersect.

6 Computational Complexity

In this section we derive several results on the computational complexity of the general MRSEV feature recognition algorithm. As mentioned in section 2, the bibliography of work on theoretical complexity results for solid modeling systems consists of a handful of references [1, 2, 8, 11, 35, 36].

Calculating the computational complexity of a feature recognition system can be approached several ways. If the feature recognition algorithm operates on a data structure abstracted from a solid model of an object of interest, such as in [6, 14], its complexity can be calculated by counting the number of operations on the data structure. Alternatively, many systems employ extensive queries to the solid modeling system in a search for hints from which to build up a feature model—most notably [34]. In such systems, a precise calculation of complexity will depend on the cost of the queries. Factors such as the cost of primitive solid modeling operations such as "union", "intersect", or "sweep" are dependent on the particular implementation of solid modeler. Some commercially available solid modeling packages contain additional layers of data abstraction and overhead which may also contribute to the computational cost. Because algorithm 5.1 uses extensive queries to the solid modeling system to access the information necessary to extract feature instances, we will choose the latter approach.

For the purposes of this analysis, we will not consider the overhead costs of commercial solid modelers nor the consequences of the various different representations they may employ. What appears to be a reasonable goal is to isolate the subset of the fundamental functions needed to implement the procedures of section 4 and 5 and to determine the complexity of those procedures with respect to that subset.

Assume a solid S is represented by some boundary data structure, in general, the size n_S of this data structure will be $n_S = O(E_S)$ where E_S is the number of edges of the solid.[7] There are

[7]For the worst case of these data structures, we can say the size is $O(n_S)$ where $n_S = E_S + V_S + F_S$ and $E_S, V_S,$

two fundamental types of primitive solid modeling operations required by our procedures:

- Operations that query information from a solid model and its constituent parts. For example, given a solid S, generate a list of the edges of S; given an edge e, determine the endpoints and parameters[8] of the edge.

  Similar to analyses presented for boundary data structures in [2, 11, 36], we assume that queries of this type can be made in time linear in the number of primitive elements queried. For instance, given a solid model S, a query to generate a list of all the faces in the solid model takes $O(n_S)$ time—the time required to traverse a structure, such as a boundary representation, and identify all the faces. A query to find all the edges of a particular face would also take time $O(n_S)$. If we have a given edge e, query to determine the parameters of e takes $O(1)$ time.

- Operations that change a solid model either by changing its characteristics or by its interaction with other solid models. For example, creation of solids, Euler operations, and boolean set operations.

  Procedures 4.1 through 4.3 and algorithm 5.1 require routines for creation of solid models; the regularized boolean operations of union, subtraction, and intersection; translation; rotation; and sweeping. The boolean operations are discussed extensively in [11] and their complexities are quadratic in n_S. For example, if A and B are two solid models and n_A and n_B are their sizes then the operation $A -^* B$ can be performed in $O((n_A + n_B)^2)$ time. Translation and rotation of solid can be done in linear time by applying the appropriate transformation to each of the elements in the boundary representation. Sweeping also requires the consideration of all entities of a boundary representation of an entity S and, depending on the particular implementation, the creation and union of additional solids to get the swept volume—hence it is also $O(n_S^2)$. These operations are common model construction techniques [23].

Given these assumptions, the complexity of procedure 4.1 is $O(n^2_{\mathbf{part}})$; procedure 4.2 is $O(n^2_{\mathbf{part}})$; and procedure 4.3 is $O(n^3_{\mathbf{part}})$. The detailed analysis of the complexity for each procedure in section 4 can be found in appendix A.

In order to make a meaningful statement on the complexity of algorithm 5.1, it is necessary to prove that the problem as defined is computable. To prove computability, we must show that algorithm 5.1 will halt for any input from its domain of objects. This implies we consider an arbitrary solid from the domain of objects stated in section 3 and show that the algorithm will halt—returning a feature model for the object or reporting that none exists.

Let S be a solid object and let $\mathbf{WP}_0$ be the stock we will attempt to describe it from using our class of MRSEVs. We will show that the algorithm MRSEV Feature Recognition will only need to consider each face of Δ once, where $\Delta = \mathbf{WP}_0 -^* S$. The proof of theorem 6.1 can be found in appendix A.

Theorem 6.1 *(*Termination of MRSEV Feature Recognition*) Let S be a solid object and $\mathbf{WP}_0$ be an initial workpiece. The algorithm* MRSEV Feature Recognition *will consider each face of Δ at most once.*

Analysis of algorithm 5.1: Theorem 6.1 proves that the While loop is executed only a finite number of times. Note that at each iteration, the algorithm considers a face of Δ that remains in

and F_S are the number of edges, vertices, and faces of S respectively. By Euler's equation $2 = V - E + F$, we can simplify this to be $n_S = 2 + 2E$ or $n_S = O(E_S)$.

[8]For instance, the parameters of a circular edge would be the axis and radius; of an elliptical edge would be the major and minor axes and foci.

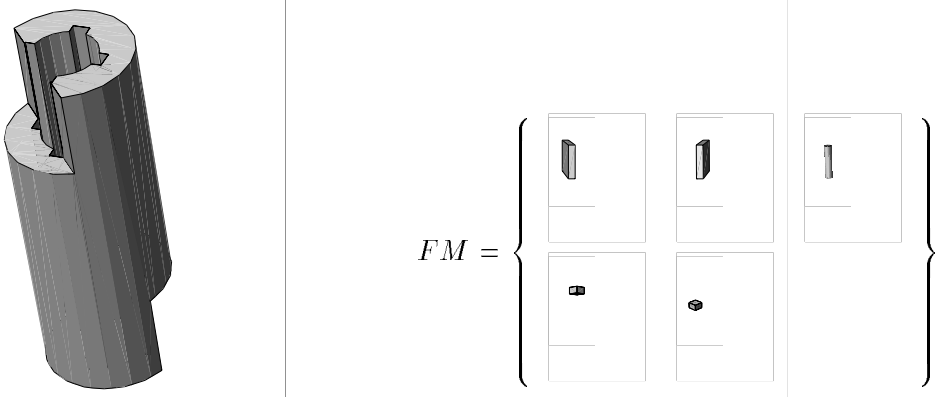


Figure 4: An example part with a variety of feature intersections and a feature model describing it.

$\Delta -^* \bigcup F$ and that each face of Δ is picked at most once. Hence, the While loop will be executed at most $O(n_\Delta)$ times, where n_Δ is the size of Δ. Because $O(n_\Delta) = O(n_{\mathbf{part}})$, we will say that the While loop is executed $O(n_{\mathbf{part}})$ times.

The interior of the While loop consists of identifying the type of face that was picked. Depending on the geometry of the face, the algorithm calls one or more of our procedures from section 4 to find the feature or set of features that may associated with that face. There is at most a single call to the subroutines for holes and pockets, the most costly of which is FIND MRSEV POCKET INSTANCE. Hence the complexity of the algorithm MRSEV FEATURE RECOGNITION is $O(n^4_{\mathbf{part}})$.□

7 Examples

To illustrate that this algorithm can function in realistic machining situations, Figure 3 provides some examples from the domain described by these MRSEVs. These figures appeared previously in [5, 12, 13].

An example of the general algorithm's feature intersection independence can be found in Fig. 4.[9] This figure depicts a cylindrical part containing a hole with two intersecting keyways and shoulders. The interaction of the criss-crossing keyways within a cylindrical hole could be problematic for many feature recognition methodologies. If delta-volume reduction technique were used, the possibility exists that it may recognize the hole first—thus making the keyways difficult, if not impossible, to recognize. Any methodology attempting to find edge loops to determine the cross-section of the hole would find there is no such planar edge loop in the part. The shoulders, while easily described as instances of MRSEV pockets, may confuse a system that cannot deduce the existence of faces not in the final part. In some graph-based recognition schemes, each keyway doubles the number of graph elements needed to describe the hole. While this may not preclude the recognition of the hole, it will require additional computational time to recognize the hole—a task that, in the worst case for some systems, is exponential.

Algorithm 5.1 handles this example without any special-case reasoning. If the stock material is the complete cylinder, then the two shoulders both have part of their bottom faces in Δ—hence, by lemma 4.3, the algorithm will find MRSEV pockets subsuming all other MRSEV pockets that describe the shoulders. For each of the keyways, it is evident that at least two of their non-planar side faces are present in Δ—hence, by lemma 4.2, they will be recognized as instances of MRSEV

[9] The four MRSEV pocket instances and single MRSEV hole instance in FM are not rendered in scale with respect to the part.

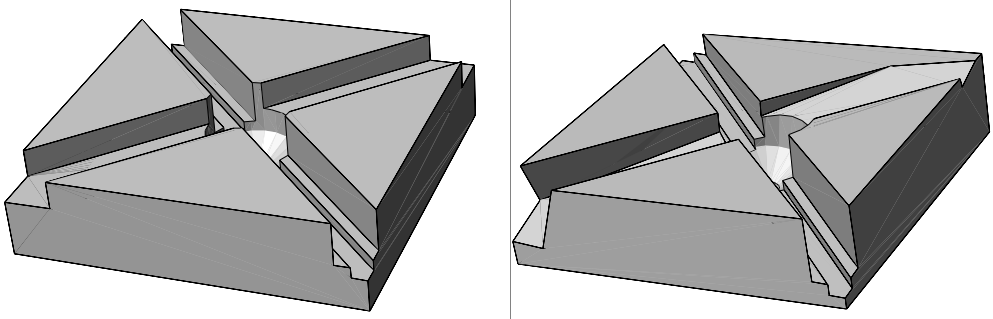


Figure 5: An example from [34] with interacting slots and a hole and a variation on it.

pockets. Recognizability of the hole is given by lemma 4.1.

Figure 5 presents an example from [34] with interacting slots and a variation on it. In the original example, algorithm 5.1 would be capable of finding a feature model Because each of the slots can be identified from the part of its bottom face in Δ and the center hole can be found from the part of its cylindrical face in Δ. The variation contains the same features but complicates their instances and interactions. The hole and slots occur at different angles relative to the part—no longer are the opposite faces of the slots necessarily similar. However, the information essential to recognizing a feature model, the same information as for the original example, still exists in Δ.

It is important to note that the theorems, while strong enough to guarantee the recognizability of a large class of parts, do not make any statements regarding the MRSEVs instances that will be found. As described, the algorithm is non-deterministic and, depending on its implementation, the MRSEV feature model described above for Fig. 4 is one of many alternate valid models.

8 Conclusions

In this paper, we have presented a new approach for recognition of machinable features from solid models.

Most previous work on this topic has focused on introducing new techniques for getting a solution to the problem, without fully exploring the capabilities and limitations of these techniques. As a result, it has often been unclear what specific classes of objects, features, and feature interactions can be handled by previous approaches. In contrast, the primary goal of our work has been to develop a general definition of the problem domain, an algorithm capable of handling all problems in that domain, and formal analyses of the algorithm's completeness and complexity.

Our approach has the following characteristics:

1. Our problem definition is based on a standard class of machining features [18] that describe a wide variety of shapes manufacturable on 3-axis milling machines. The primary limitation of our approach as presented here is that it is designed only to handle linearly swept features (i.e., holes and pockets). However, our definitions of holes and pockets are more general than the definitions used in a number of feature recognition systems; for example, the pockets may be complicated swept contours that include corner radii, islands, blends and other characteristics, in order to realistically describe a non-trivial set of mechanical parts.

2. Our approach, like that of [34], uses queries to the solid modeler to search for the information from which to identify feature instances. It differs from the iterative Δ-volume subtraction techniques such as developed in [4, 10] in that we do not reduce Δ to $\emptyset$. It is not based on expert-system rules for deducing feature instances and resolving interaction as in [3, 4, 10, 21].

Further, because our algorithm and features are not graph-based, we have been able to base them on a realistic feature class and avoid the tractability problems of [14, 21]. In particular, our algorithm is guaranteed to find feature models for all parts describable with our features, regardless of how complicated the interactions are among the features.

3. The algorithm's complexity of $O(n^4)$ represents an improvement over other approaches such as those based on subgraph isomorphism matching and expert systems, some of which require exponential time in the worst case. In addition, $O(n^4)$ is a worst-case bound on the algorithm and would only occur in the most geometrically pathological examples. The typical-case run-time would likely be lower.

Near-term goals for future work include incorporating a more sophisticated definition of accessibility and implementing our algorithm. Medium-term directions include extending our results and procedures to include other MRSEVs; generalizing these results to encompass a wider variety of feature recognition domains; and exploring techniques for the simplifying the model in order to achieve a reduction in complexity (as done in [9]).

As a long-term goal, we hope to develop a general computational paradigm for recognition of machinable features, and mathematical results presented in this paper can be viewed as a first step toward that goal. More powerful results of similar nature will be required to build a satisfactory and useful formalism for a wider class of feature recognition problems. Such a formalism would provide a framework within which to compare and contrast the results of feature recognition research in any application area that can be represented in this class. This would allow conclusions about complexity, features recognized, feature interactions, and completeness of an approach to have significance outside individual application areas.

Acknowledgments

The authors would like to thank Tom Kramer of NIST for the information he gave us about MRSEVs, and Peter F. Brown and the Engineering Design Laboratory for many of the insights gained while at NIST. Also, Aristides Requicha and Jan Vandenbrande for their helpful input on a direction for our formalism. We also thank Satyandra K. Gupta, Raghu R. Karinthi and Thomas J. Peters for their feedback on the structure and content of the paper. Finally we also thank Ithaca Software for the use of their HOOPS© graphical rendering system.

A Appendix

A.1 Primitive Procedures

The following procedures will help us in recognizing instances of the MRSEVs we described earlier. These procedural primitives will be used to build the recognition procedure for the MRSEVs defined in section 3. We give pseudo-code for these procedures, because the specific details will depend on what technique one uses to represent solid models. Instead, for each procedure we give an outline illustrating that it is within the abilities of solid modeling systems. The details showing how these procedures could be implemented can also be found in [29].

Procedure A.1 MAXIMUM ENCLOSING CYLINDER
INPUT: **part**, *s, a subset of a cylindrical surface.*
OUTPUT: *an accessible cylinder* C_s *such that* C_s *is the largest cylinder of height less than or equal to* **min** *having s as a subface and* $C_s \cap^*$ **part** $= \emptyset$.

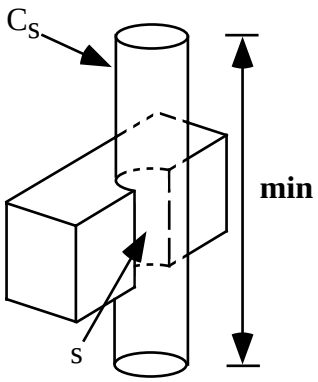


Figure 6: Maximum Enclosing Cylinder.

Consider the cylinder C_m of height **min** with the same axis and radius as the surface s and centered along the axis such that the surface s contains one or more points in the cylindrical side face of C_m equidistant from C_m's planar side faces. Recall, that because C_m is of height **min** and has its center somewhere within the stock we are guaranteed that the both end faces of C_m lie outside the stock. We define C_s to be the largest cylinder with the same radius and axis as s, centered along s, such that $C_s \subseteq C_m$ and $C_s \cap^*$ **part** $= \emptyset$ and C_s is accessible. This can be determined by examining the maximum and minimum points on the edges of **part** $\cap^* C_m$ with respect to a plane perpendicular to the axis of s. □

In this feature class, there are only two ways to make a cylindrical surface: as part of a hole or as a corner radius in the profile of a pocket. The procedure Maximum Enclosing Cylinder returns the cylinder that is a the body of the maximal MRSEV hole capable of creating surface s. Note, by the definition of accessibility, the cylinder returned must be accessible via at least one of its two ends. If both ends of C_s are blocked by the **part** then there would be no way of making such a hole.

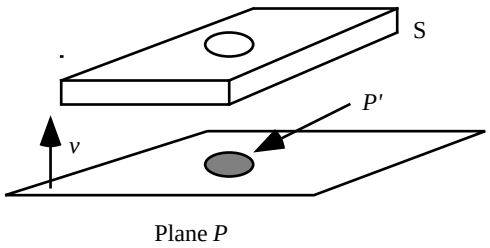


Figure 7: Project Object Onto Plane.

Procedure A.2 Project Object Onto Plane
INPUT: *a plane P, an object S, and a normal vector to P, v.*
OUTPUT: *subplane P' of P such that P' is the projection of S in the direction of v onto P. Note: P' may be zero or more disjoint faces.*

The plane P and vector v define a half-space; the intersection of this half-space with S yields S', the portion of S "above" P in the direction of v. A transformation of the entities of S' computes the projection onto P, leaving a set $\overline{P}$, of two dimensional faces. Therefore, $P' = P -^* \overline{P}$, is a set of disjoint faces such that for each face f, is a potential bottom face of a pocket. □

The vital defining attribute of a MRSEV pocket is the **profile**. PROJECT OBJECT ONTO PLANE can be used to get the profile of any MRSEV pocket. There are two possibilities for a pocket instance: a through pocket and non-through pocket. For the through pocket, if we know its orientation, the profiles of all through pockets in that orientation can be calculated by projecting the **part** onto any plane perpendicular to that orientation. Each of the edge loops in the projection are profiles of through pockets on that orientation.

In the case of a non-through pocket, we know there exists a subface of its bottom surface in Δ. PROJECT OBJECT ONTO PLANE can then be used to find an initial **profile** for the pocket. This **profile** can be used to find blends and islands. The final pocket's **profile** will be altered to include the additional area of any blends.

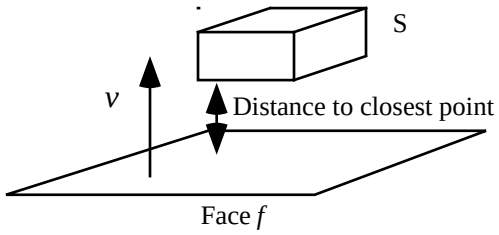


Figure 8: DISTANCE TO CLOSEST POINT.

Procedure A.3 DISTANCE TO CLOSEST POINT
INPUT: *a planar face f, a vector normal to f, v, and an object S.*
OUTPUT: *the distance from f on vector v to a point p on the surface of S such that, for all other points p' on S on vector v from f, the distance from f to p' is greater than or equal to the distance to p.*

Let S' be the intersection of the swept solid with bottom face f and height equal to **min** with the **part**. The solid resulting from the intersection, if one exists, can be transformed into a coordinate system having v as an axis. The point on each face of S' closest to f on vector v can be calculated by geometric based on the surface type (i.e. calculating the closest point when the face of S' is planar is a different formula than the case when it is cylindrical). The smallest of these distances is the distance from f to S on vector v. □

DISTANCE TO CLOSEST POINT is used in the case of a MRSEV hole instance to obtain the distance to the bottom of a hole.

A.2 Recognition Procedures

Procedure 4.2 FIND MRSEV HOLE INSTANCE
INPUT: **part**, $\mathbf{WP}_0$, *a face f in* $\mathbf{WP}_0 -^*$ **part** *formed from an instance of a MRSEV hole.*
OUTPUT: *M, a MRSEV hole found from f.*

$\Delta = \mathbf{WP}_0 -^* \mathbf{part}$
if f is cylindrical and $>$ **max_blend**:
  orientation = query(f,*axis direction*)
  radius = query(f,*radius*)
  $f_{\max}$ =MAXIMUM ENCLOSING CYLINDER(f)
  location = query($f_{\max}$,*location*)
  depth = query($f_{\max}$,*height*)
  end = *flat*
  create instance M
else
  if f is conical, toroidal, or spherical and $>$ **max_blend**:
    end = query(f,*surface type*)
    location = query(f,*location*)
    orientation = query(f,*axis*)
    depth = **min**
    /* in the case where f is conical, query for the
          radius of the base of the f */
    radius = query(f,*radius*)
    create instance M
  else
    if f is planar:
      noop
    endif
return(M)

Procedure 4.3 FIND BOTTOM BLENDS
INPUT: **part**, *a set of edge profiles P, and a vector v indicating the direction of the top of the pocket.*
OUTPUT: *a set $\mathcal{B} = \{\langle P', B\rangle\}$ where P' is a set of profiles, and B is a set $B = \{s\}$ where s is a blend surface for the pocket whose profile and islands are in P'.*

$B = \emptyset$
V = volumes of height **min** with the profiles of P as bottom faces
/* these volume can be created by, among other ways, sweeping
    the faces outlined by P in direction v distance **min** */
S = query(**part**,*surfaces incident with elements of V* and of size $\leq$ **max_blend**)
P' =PROJECT OBJECT ONTO PLANE(S,*plane of P*)
for each $p \in P$ that shares an edge with $p' \in P'$ do:
  $P' = P' - \{p'\}$
  $P' = P' \cup \{p \cup p'\}$
  $B = B \cup \{s\}$
  /* s is the surface that was projected onto
      the plane of P to get p' */
return(P', B)

Procedure 4.4 FIND MRSEV POCKET INSTANCE
INPUT: **part**, $\mathbf{WP}_0$, *a face f in* $\mathbf{WP}_0 -^* \mathbf{part}$ *formed from an instance of a MRSEV pocket.*
OUTPUT: *a set $\mathcal{M} = \{M'\}$ of instances of a MRSEV pockets found from f.*

$\Delta = \mathbf{WP}_0 -^* \mathbf{part}$

$\mathcal{M} = \emptyset$
if f is planar:
  /* Find the set of maximal pockets for which f is
       coplanar with the bottom face */
  f_P = PROJECT OBJECT ONTO PLANE(**part**, f, **norm**(f))
  P =query(f_P, *outer edge loops*)
  for each $p \in P$ do
    orientation = **norm**(f)
    depth = **min**
    location = query(p, *vertices*)
    I = query(f_P, *inner edge loops inside* p)
    islands = I
    $\mathcal{B} = \emptyset$
    $\mathcal{B}$ =FIND BOTTOM BLENDS(**part**, $\{p\} \cup I$, **norm**(f))
    for each $\langle P', B \rangle \in \mathcal{B}$ do
      profile =query(P',*edge loop containing* p)
      bottom blend =query($s \in B$,*radius or width*)
        /*depends on type of surface of s*/
      for each $l \in I$ do
        profile I_l =query(P',*edge loop containing* l)
        orientation I_l = **norm**(f)
        bottom blend $I_l = B$
        height I_l = **min**
      create instance M
      $\mathcal{M} = \mathcal{M} \cup \{M\}$
  /* Find the set of maximal through pockets for which
       f contains part of a side face */
  F =query(Δ,*planar faces not parallel to* f)
  for each $f' \in F$ do
    calculate a plane p perpendicular to f and f'
    p_{up} =PROJECT OBJECT ONTO PLANE(**part**, p, **norm**(p))
    p_{down} =PROJECT OBJECT ONTO PLANE(**part**, p, $-$**norm**(p))
    $p' = p_{\text{up}} \cap^* p_{\text{down}}$
    P =query(p',*edge loops*)
    for each **profile** $\in P$ do
    /* each edge loop in P is a profile of a through pocket */
      bottom blend = *none*
      orientation = **norm**(p)
      depth = **min**
      location =query(p',*vertices*)
      islands = $\emptyset$
      create instance M
      $\mathcal{M} = \mathcal{M} \cup \{M\}$
endif
if f is cylindrical:
  /* Find the set of maximal through pockets for which f is
       a corner radius */
  orientation =query(f, *axis*)
  calculate any plane p normal to **orientation**

p_{up} =PROJECT OBJECT ONTO PLANE(**part**, p, **norm**(p))
p_{down} =PROJECT OBJECT ONTO PLANE(**part**, p, $-$**norm**(p))
$p' = p_{\text{up}} \cap^* p_{\text{down}}$
P =query(p',*edge loops*)
for each **profile** $\in P$ do
/* each edge loop in P is a profile of a through pocket */
bottom blend = *none*
orientation = **norm**(p)
depth = **min**
location =query(p',*vertices*)
islands = $\emptyset$
create instance M
$\mathcal{M} = \mathcal{M} \cup \{M\}$
endif
return($\mathcal{M}$)

A.3 Lemmas, Theorems, and Corollaries

Lemma 4.1 *Let M be a MRSEV hole having a subface of one of its faces as a face in Δ, then there exists a recognizable MRSEV hole M' such that $M \subseteq M'$.*

Proof: Let f be a face of M in Δ, to find an instance of a MRSEV hole requires **location**, **orientation**, **radius**, **depth**, and **hole end**. We will show how suitable values for these attribute can be found from Δ and the **part**.

Case 1: f is conical, spherical, or toroidal

If f is one of these surface types then it must be the end surface for the hole—hence providing a value for **hole end** and **location**. Also, each of these surfaces provides an **orientation** for the hole. The **depth** is determined from the hole bottom of which f is a subface; it must extend past the initial workpiece $\mathbf{WP}_0$. Hence the **depth** may be arbitrarily set to **min**. We know that f is the complete ending surface for some hole. Hence the **radius** of the hole is determined by the radius if it is a spherical surface, the major and minor radii if it is a toroidal surface, or the radius of the base if it is a conical surface.

Case 2: f is cylindrical

Let C_s be the largest cylinder containing f as a subface as computed by MAXIMUM ENCLOSING CYLINDER. C_s gives us values for the **radius** and **orientation**. If the cylinder C_m centered in C_s of height **min** with the same radius and orientation as C_s does not intersect the **part** then C_s is creatable by a through hole. In this case the **hole end** and **location** attributes can be arbitrarily set to any values instantiating such a through hole.

If C_m does intersect the **part** then it does so in one direction of the **axis**.[10] In the direction of C_m's intersection with the **part** will be a planar **hole end**—any other kind of hole end would have been found in case 1.

[10] If C_m intersected the **part** in both directions of the **axis**, the hole would be inaccessible.

Using the circular cross section of C_s as a face, determine the DISTANCE TO CLOSEST POINT of the **part** in the direction of the hole bottom. The provides the **depth** and **location** of the deepest hole that may have created face f. Therefore we have found an instance of a flat bottomed hole.

Case 3: f is planar

We need not consider planar surfaces created by MRSEV holes. The only such planar surface must be a hole bottom and, if not recognized already by case 2, it must also be recognizable as the bottom of a MRSEV pocket.

Hence in each case we have determined attributes for an MRSEV hole M' recognizable from Δ and at least as large as the hole M that contained f as a subsurface. □

Lemma 4.2 *Let M be a MRSEV through pocket with subfaces of two or more non-parallel planar side faces or a subface of a corner radius as subfaces of the boundary of Δ, then there exists a set of recognizable MRSEV pockets $\mathcal{M}$ such that $M \subseteq \bigcup \mathcal{M}$.*

Proof:

Case 1: There exists non-parallel planar faces f_1 and f_2 in Δ

Let f_1 and f_2 be the faces of Δ containing subfaces of two non-parallel planar side faces of M. Again, we wish to find values for the **location**, **orientation**, **depth**, **profile** and **islands** attributes of a MRSEV pocket M'.

Because f_1 and f_2 are non-parallel, it is known they intersect at a line, l. Pick a point p on l and consider the plane P passing through p and perpendicular to l.

Compute P_1' and P_2' with PROJECT OBJECT ONTO PLANE. P_1' is the projection of the **part** onto P in one direction on l, P_2' is the projection from the other direction. The set of planar faces given by $P_1' \cap^* P_2' \cap^* \Delta$ are the cross-sections of the through pockets parallel to line l. Consider the face f_p in this set that has edges from the projection of f_1 and f_2 onto P.

Line l provides an **orientation**; because it is a through pocket the **depth** may be set to **min** and the **location** can be set to an arbitrary place outside Δ. Face f_p provides a **profile** and, again because it is a through pocket, there can be no **islands**.

Case 2: There exists a cylindrical face f in Δ

In this case, f forms part of a corner radius of the pocket. Computing the axis of the surface yields us the **orientation**. Again, because it is a through pocket the **depth** may be set to **min** and the **location** can be fixed to a point outside Δ.

As above, compute P_1' and P_2' with PROJECT OBJECT ONTO PLANE where P_1' is the projection of the **part** onto P in one direction on l, P_2' is the projection ¿from the other direction. The set of planar faces given by $P_1' \cap^* P_2' \cap^* \Delta$ are the cross-sections of the through pockets parallel to line l. Consider the face f_p in this set that has edge from the projection f onto P. Face f_p provides a **profile** and, because it is a through pocket, there can be no **islands** or **bottom_ blends**.□

Lemma 4.3 *Let M be a MRSEV pocket having a subface of its bottom face as a subset of the boundary of Δ, then there exists a set of recognizable MRSEV pockets $\mathcal{M}$ such that $M \subseteq \bigcup \mathcal{M}$.*

Proof: Let f be the face of Δ containing a subface of the bottom of M. An instance of a MRSEV pocket requires finding values for the **location**, **orientation**, **depth**, **profile**, **bottom_ blends**, and **islands** attributes.

Let v be the normal vector[11] to the surface of f, P be the plane containing f, and S be the solid object that is the intersection of the **part** with the half-space above P in the direction of f. Compute the set of subfaces of plane P, P' using PROJECT OBJECT ONTO PLANE. Consider the subface f' of $P' \cap^* \Delta$ containing f.

The face f' determines a **location** for the MRSEV pocket bottom and vector v provides an **orientation**. The **depth** can be set to **min** because the result of the projection implies that there is no subset of the **part** above f' in the direction v. The outside edge loop of f' gives an initial value for the **profile** of the MRSEV pocket and, lastly, any interior edge loops define the locations for any **islands**.

Bottom blends are found by considering the surfaces on the **part** with edges incident on the solid formed by sweeping face f' to height **min** in direction v. Of the many incident edges, the possible blends will be those tangent to f' or extendable to being tangent to f'. In addition, blend surfaces must conform to a size restriction, as stated in our assumption from section 3. There are two things that must be done:

- Ensure that the potential blend is accessible by considering a planar cross-section in the plane of the pocket bottom. If this planar cross section is not blocked in the direction given by v (this can be determined with PROJECT OBJECT ONTO PLANE) then the surface is a blend.
- If multiple blend types or radii exist for a particular pocket bottom, then the removal volume will be described as a set $\mathcal{M} = \{M'\}$ of overlapping MRSEV pockets each with different blend conditions. Satisfying the condition $M \subseteq \mathcal{M}$ requires that we make each pocket instance as large as possible. This can be done by letting each have f' as its bottom face unless the blends interfere with one another. In which case, the dimensions of the interference can be calculated and the bottom **profile** for that particular MRSEV can be altered to insure that it is largest MRSEV instance where the blends do not interfere.

Hence, for any MRSEV pocket M having a subface of its bottom face as a subface of Δ, there exists a set of recognizable MRSEV pockets $\mathcal{M} = \{M'\}$ where $M \subseteq \bigcup \mathcal{M}$.□

Theorem 5.1 *Given any feature model of a* **part** *and* $\mathbf{WP}_0, FM$, *for all MRSEV features* $M \in FM$, *there exists a set of features* $\mathcal{M} = \{M'\}$ *where each* M' *is recognizable in* Δ *and* $M \subseteq \bigcup M$.

Proof: Let FM be a feature model for a **part** and $\mathbf{WP}_0$ and let $M \in FM$. M is either a MRSEV hole or pocket feature.

Case 1: M is a hole

By our previous assumptions, it is known that a subface of one of the faces of M is a subface of a face in Δ. Therefore, by Lemma 4.1 there exists a recognizable feature M' such that $M \subseteq M'$.

Case 2: M is a pocket

[11] The vector v points away from the interior of the **part**

M has a portion of its bottom face in Δ or is a through pocket with at least two of its non-parallel planar side faces or a portion of a corner radius present in Δ. The former case has been proven in Lemma 4.3 and the latter in Lemma 4.2.

Hence, if **part** and $\mathbf{WP}_0$ have a feature model FM we can recognize a set of features forming another feature model of **part** and $\mathbf{WP}_0$, FM', such that $\forall M \in FM, \exists M' \in FM'$ such that $M \subseteq M'$.□

Corollary 5.1 *(*Completeness*) Suppose* **part** *and* $\mathbf{WP}_0$ *can be described by a feature model, then the procedure* MRSEV FEATURE RECOGNITION *returns a feature model of* **part** *and* $\mathbf{WP}_0$*.*

Proof: The procedure, using the results and subroutines from section 4, finds features M' such that there does not exist a feature M'' where $M' \subseteq M''$. We must show that the set F returned by MRSEV FEATURE RECOGNITION is a feature model. Therefore we must show:

$$i. \quad \forall M_i \in F, M_i \cap^* \mathbf{part} = \emptyset$$
$$ii. \quad \mathbf{WP}_0 -^* \mathbf{part} \subseteq \bigcup F$$

The way recognizable features are instantiated by the procedure satisfies *(i)*. To show *(ii)*, recall that there exists a feature model FM of $\mathbf{WP}_0$ and **part**. By theorem 5.1, it is known that for every feature $M \in FM$ there exists a feature M' in F such that $M \subseteq M'$. Therefore $\mathbf{WP}_0 -^* \mathbf{part} \subseteq \bigcup FM \subseteq \bigcup F$

Hence the procedure returns an feature model for any $\mathbf{WP}_0$ and **part** that have one. □

Corollary 5.2 *(*Feature Intersection Independence*) Suppose* **part** *and* $\mathbf{WP}_0$ *can be described by a feature model and for every such model the features intersect. The procedure* MRSEV FEATURE RECOGNITION *returns a feature model of* **part** *and* $\mathbf{WP}_0$*.*

Proof: Again, we know a feature model exists: call it FM. The fact that the features intersect in an inconceivably pathological manner does not alter the fact that the procedure, using the results and subroutines from section 4, finds features M' such that there does not exist a feature M'' where $M' \subseteq M''$. We must show that the set F returned by MRSEV FEATURE RECOGNITION is a feature model.

As before, we get *(i)* for free. To show *(ii)*, recall by theorem 5.1, it is known that for every feature $M \in FM$ there exists a feature M' in F such that $M \subseteq M'$. Therefore $\mathbf{WP}_0 -^* \mathbf{part} \subseteq \bigcup FM \subseteq \bigcup F$

Hence the procedure returns an feature model for any $\mathbf{WP}_0$ and **part** regardless of how the features describing $\mathbf{WP}_0$ and **part** interact. □

Theorem 6.2 *(*Termination of MRSEV FEATURE RECOGNITION*) Let S be a solid object and* $\mathbf{WP}_0$ *be an initial workpiece. The algorithm* MRSEV FEATURE RECOGNITION *will consider each face of* Δ *at most once.*

Proof:

Case 1: There exists a feature model describing Δ

Because a feature model exists for Δ, there must exist a set of features $FM = \{M_1, M_2, M_3, \ldots, M_n\}$ such that $\Delta \subseteq \bigcup FM$. Hence, each face of Δ must belong to some feature. Further, because $\forall M_i \in FM, M_i \cap^* \mathbf{part} = \emptyset$, each face of Δ that is also a face of **part** must be a subface of a face of one or more $M_i \in FM$.

Let f be a face in Δ and assume f is "picked" while MRSEV FEATURE RECOGNITION is computing a feature model. If f is cylindrical and created by a MRSEV hole, the hole instance can be computed—f need not be considered again. If f is cylindrical and not created by a hole (for example, f may be the corner radius of a pocket) the MRSEV instance describing f will be built from other faces in Δ. If f is planar then it may: 1) contain part of the bottom faces of one or more MRSEV pockets; 2) contain part of the side faces of one or more bottomless MRSEV pockets; 3) contain portions of the side faces of MRSEV pockets which have parts of their bottom faces elsewhere in Δ. In case (3), these portion of f face make no contribution to building a feature model and can be discarded. For cases (1) and (2), all such features can be recognized ¿from f and f will not have to be picked again. When f is toroidal, spherical, or conical it may be formed as an end condition for a hole or as part of a pocket. When f is an end condition for a hole, it can be used to determine the attributes of the instance of the hole. Otherwise, when f is part of a pocket, the pocket will be recognized by other planar faces that are either its bottom or sides.

Hence f needs to be considered at most once and the algorithm MRSEV FEATURE RECOGNITION will then halt having found a feature model.

Case 2: There does not exist a feature model describing Δ

If there is no feature model describing S and $\mathbf{WP}_0$ then the algorithm will terminate when all faces will become marked. Upon termination, we will wish to determine whether or not the set returned, F, is a feature model. This can easily be done by checking to see that F satisfies the two criteria of a feature model:

$$\begin{array}{ll} i. & \forall M_i \in F, M_i \cap^* \mathbf{part} = \emptyset \\ ii. & \mathbf{WP}_0 -^* \mathbf{part} \subseteq \bigcup F \end{array}$$

As a byproduct of the way the algorithm builds MRSEV instances, we know F will satisfy *(i)*. However, Because no feature model exists, we know that for every set of MRSEV instances MI, one of the following must be true otherwise there would exist a feature model:

$$\begin{array}{ll} a. & \forall M_i \in MI, M_i \cap^* \mathbf{part} \neq \emptyset \\ b. & \mathbf{WP}_0 -^* \mathbf{part} \not\subseteq \bigcup MI \end{array}$$

Because F satisfies *(i)*, *(a)* cannot be true; to verify that a feature model was found requires that we check *(b)*—this can easily be done with the routines available to us through the solid modeler.

Therefore, if no feature model exists, MRSEV FEATURE RECOGNITION will halt and return a set F that will be an invalid feature model. □

A.4 Additional Complexity

Analysis of procedure A.1: Suppose we have part of a cylindrical face, s. We can, in constant time, determine its axis v and radius r. Further, we can also obtain in constant time a point x on surface s and the point y on the axis of s that is perpendicular to it. The cylinder C_m centered at y, with axis v and radius r, having its base a distance $\frac{\mathbf{min}}{\mathbf{2}}$ below y can be instantiated and the intersection with **part** computed in $O(n^2_{\mathbf{part}})$. Finding the maximum cylinder of radius r and axis v that fits inside this intersection volume requires examining all the edges of the volume for extrema points. The extrema points can be found in constant time with known formula hence this takes

$O(n_{\mathbf{part}})$.[12] Therefore the MAXIMUM ENCLOSING CYLINDER can be found in $O(n^2_{\mathbf{part}} + n_{\mathbf{part}})$ or just $O(n^2_{\mathbf{part}})$ time.□

Analysis of procedure A.2: Given a plane P, a vector v normal to P, and a solid model S, we want to determine P', the complement of the projection of the part of S above plane P as defined by v onto P. Let S' be the portion of S that lies in the half plane above P in the direction given by v; S' can be computed via an intersection operation in $O(n_S^2)$ time. P' can be created by taking the solid S' and sweeping it distance **min** in the direction opposite v, through P, to create a solid S''. Cost of the sweep is $O(n_S^2)$. Hence P' now can be computed as: $P' = P -^* S''$ at cost $O(n_S^2)$. Hence the cost of procedure PROJECT OBJECT ONTO PLANE is $O(n_S^2)$.□

Analysis of procedure A.3: Given f, v, and S, create the solid S_f of height **min** sweeping f in direction v distance **min**. This operation takes $O(n_S^2)$ time. Form the intersection $S_f \cap S$ and call it S'—this also takes $O(n_S^2)$. Now we need to determine the minimum point on the top surface or surfaces of S' relative to f and v. This can be done by examining all the top surfaces and their bounding edges and employing min/max formulas to find minimum points ($O(n_S)$). The distance from the least of all of these points to f will yield us the DISTANCE TO CLOSEST POINT and can be found in $O(n_S^2)$.□

Analysis of procedure 4.1: This procedure contains several queries to the solid modeler and a single call to MAXIMUM ENCLOSING CYLINDER. In the worst case, the execution of the queries costs $O(n_{\mathbf{part}} + n_{\mathbf{WP}_0})$ and the call to MAXIMUM ENCLOSING CYLINDER costs $O(n^2_{\mathbf{part}})$. Hence FIND MRSEV HOLE INSTANCE can be executed in $O(n^2_{\mathbf{part}} + n_{\mathbf{part}} + n_{\mathbf{WP}_0})$ or just $O(n^2_{\mathbf{part}})$.□

Analysis of procedure 4.2: The cost in this procedure is incured during the call to PROJECT OBJECT ONTO PLANE. All the other queries and the for each loop are $O(\mathbf{part})$. Hence the procedure FIND BOTTOM BLENDS is $O(n^2_{\mathbf{part}})$.□

Analysis of procedure 4.3: The worst case for this algorithm occurs when the face f is planar and it must enter a for each loop iterating through all profiles of pockets sharing that bottom. This loop has a worst case time of $O(n_{\mathbf{part}})$ because there cannot be more profiles in f_P than elements the structure of **part**.[13] The loop contains a call to FIND BOTTOM BLEND at a cost of $O(n^2_{\mathbf{part}})$. Hence the overall complexity of the procedure FIND MRSEV POCKET INSTANCE is $O(n^3_{\mathbf{part}})$.□

References

[1] Seshagiri Rao Ala. Design methodology of boundary data structures. In Jaroslaw Rossignac and Joshua Turner, editors, *Symposium on Solid Modeling Foundations and CAD/CAM Applications*, pages 13–23, New York, NY 10036, USA, Austin, TX, June 1991. ACM SIGGRAPH, ACM Press.

[12] C_m is always a cylinder, hence the cost of querying it is constant with respect the input to this procedure.

[13] A more precise calculation of these quantities may be possible but will not be attempted here for we are only interested in worst case estimates.

[2] Seshagiri Rao Ala. Performance anomalies in boundary data structures. *IEEE Computer Graphics and Applications*, 12(2):49–58, March 1992.

[3] Arlo L. Ames. Production ready feature recognition based automatic group technology part coding. In Jaroslaw Rossignac and Joshua Turner, editors, *Symposium on Solid Modeling Foundations and CAD/CAM Applications*, pages 161–169, New York, NY 10036, USA, Austin, TX, June 1991. ACM SIGGRAPH, ACM Press.

[4] David C. Anderson and Mark R. Henderson. Computer recognition and extraction of form features: A CAD/CAM link. *Computers in Industry*, 5:329–339, 1984.

[5] Computer Aided Manufacturing–International, Incorporated. *CAM-I Features Symposium*, Arlington, TX, USA, August 9-10, Boston, MA 1990. P-90-PM-02.

[6] Leila De Floriani. Feature extraction from boundary models of three-dimensional objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(8), August 1989.

[7] Xin Dong. *Geometric Feature Extraction for Computer-Aided Process Planning.* PhD thesis, Rensselaer Polytechnic Institute, Troy, NY, USA, 1988.

[8] Leila De Floriani and Bienca Falcidieno. A hierarchical boundary model for solid object representation. *ACM Transactions on Graphics*, 7(1):42–60, January 1988.

[9] R. Gadh and F. B. Prinz. Recognition of geometric forms using the differential depth filter. *Computer Aided Design*, 24(11):583–598, November 1992.

[10] Mark R. Henderson. *Extraction of Feature Information from Three-Dimensional CAD Data.* PhD thesis, Purdue University, West Lafayette, IN, USA, 1984.

[11] Christoph M. Hoffman. *Geometric and Solid Modeling: An Introduction.* Morgan Kaufmann Publishers Incorporated, CA, 1989.

[12] K. E. Hummel. The role of features in computer-aided process planning. In *CAM-I Features Symposium*, pages 285–320, Arlington, TX, USA, August 9-10, Boston, MA 1990. Computer Aided Manufacturing–International, Incorporated. P-90-PM-02.

[13] K. E. Hummel and C. W. Brown. The role of features in the implementation of concurrent product and process design. In *ASME Winter Annual Meeting*, pages 1–8, New York, NY 10017, 1989. ASME. DE-Vol 21, PED-Vol 36.

[14] S. Joshi and T. C. Chang. Graph-based heurisitcs for recognition of machined features from a 3D solid model. *Computer-Aided Design*, 20(2):58–66, March 1988.

[15] R. Karinthi and D. Nau. An algebraic approach to feature interactions. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 14(4):469–484, April 1992.

[16] Raghu R. Karinthi. *An Algebraic Approach to Feature Interactions.* PhD thesis, The University of Maryland, College Park, MD, USA, 1990.

[17] Y. S. Kim. Recognition of form features using convex decomposition. *Computer Aided Design*, 24(9):461–476, September 1992.

[18] Thomas R. Kramer. A parser that converts a boundary representation into a features representation. *International Journal of Computer Integrated Manufacturing*, 2(3):154–163, 1989.

[19] Thomas R. Kramer. A library of material removal shape element volumes (mrsevs). Technical Report NISTIR 4809, The National Institute of Standards and Technology, Gaithersburg, MD 20899, March 1992.

[20] Lycourgos K. Kyprianou. *Shape Classification in Computer Aided Design*. PhD thesis, Christ College, University of Cambridge, Cambridge, United Kingdom, 1980.

[21] M. Marefat and R. L. Kashyap. Geometric reasoning for recognition of three-dimensional object features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(10):949–965, October 1990.

[22] S. Meeran and M. J. Pratt. Automated feature recognition from 2d drawings. *Computer Aided Design*, 25(1):7–17, January 1993.

[23] James R. Miller. Architectural issues in solid modelers. *IEEE Computer Graphics and Applications*, 9(5):72–87, September 1989.

[24] Dana S. Nau, Satyandra K. Gupta, Thomas R. Kramer, William C. Regli, and Guangming Zhang. Development of machining alternatives based on mrsevs. In *International Conference on Computer Integrated Engineering*. ASME, August 1993.

[25] Thomas J. Peters. Combinatorial anaylsis for feature recognition. In *Fourth International Conference on Design Theory and Methodology*, Scottsdale, AZ, September 14, 91-97 1992.

[26] 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.

[27] S. Prabhakar and M. R. Henderson. Automatic form-feature recognition using neural-network-based techniques on boundary representations of solid models. *Computer Aided Design*, 24(7):381–393, July 1992.

[28] Micheal J. Pratt. Tutorial paper on advanced topics in solid modeling: Form features and their application in solid modeling. In *SIGGRAPH 1987*. The Association for Computing Machinery, ACM Press, 1987.

[29] William C. Regli and Dana S. Nau. Building a general approach to feature recognition of material removal shape element volumes (mrsevs). In Jaroslaw Rossignac and Joshua Turner, editors, *Second Symposium on Solid Modeling Foundations and CAD/CAM Applications*, New York, NY 10036, USA, May 19-21, Montreal, Canada 1993. ACM SIGGRAPH, ACM Press.

[30] Aristides A. G. Requicha. Representation for rigid solids: Theory, methods, and systems. *Computing Surveys*, 12(4):437–464, December 1980.

[31] Aristides A. G. Requicha and Jarek R. Rossignac. Solid modeling and beyond. *IEEE Computer Graphics and Applications*, 12(5):31–44, September 1992.

[32] Scott A. Safier and Susan Finger. Parsing features in solid geometric models. In *European Conference on Artificial Intelligence*, 1990.

[33] Hiroshi Sakurai and David C. Gossard. Recognizing shape features in solid models. *IEEE Computer Graphics & Applications*, September 1990.

[34] Jan H. Vandenbrande. *Automatic Recognition of Machinable Features in Solid Models*. PhD thesis, University of Rochester, Rochester, NY, USA, 1990.

[35] Kevin Weiler. Edge-based data structures for solid modeling in curved-surface environments. *IEEE Computer Graphics and Applications*, 5(1):21–40, January 1985.

[36] Tony C. Woo. A combinatorial analysis of boundary data structure schemata. *IEEE Computer Graphics and Applications*, pages 19–27, March 1985.