ELSEVIER

0010-4485(95)00075-5

Computer-Aided Design, Vol. 28, No. 8, pp. 603–616, 1996

0010-4485/96 $15.00 + 0.00

Analysis on machined feature recognition techniques based on B-rep

M C Wu and C R Liu*

Solving the machine feature recognition problem has been widely recognized as a cornerstone for developing an automated process planning system directly linked to a CAD system. Various recognition techniques have been developed; however, they are in general deficient in robustness. That is, valid machined features may not be recognized and features which are recognized may not be valid in practice. This paper is intended to analyse the existing machined feature recognition techniques, which are based on the B-rep solid modelling scheme, in order to give the reasons why the robustness problem would occur. The pros and cons for recognizing machined features are also analysed. Finally, a cutter selection methodology, known as process requirement modelling, is introduced; this methodology seems to provide a promising way to solve the machined feature recognition problem.

Keywords: machined feature recognition, cutter selection, computer-aided process planning

INTRODUCTION

Developing methods for recognizing machined features have been widely regarded as a significant research topic in the area of CAD/CAM. The main purpose of recognizing machined features is to provide a direct link between CAD and CAM. That is, the output of CAD systems can be directly taken as the input to CAM systems so that the integrated productivity of CAD/CAM systems can be enhanced.

One essential part of a typical CAD system is a solid modeller, which is intended to model the shape of a workpiece in computational form. Based on the solid modelling schemes, algorithms can be developed for some applications such as the graphic display of workpieces, finite element analysis, group technology (GT) coding and process planning. The solid modelling file of a workpiece is generally considered as the output of a CAD system and subsequently as the input of a typical machined features recognizer.

Currently, the two most commonly used solid modelling schemes are CSG (constructive solid geometry) and B-rep (boundary representation). In the CSG scheme, a solid is modelled by the regularized Boolean operation of several primitive solids such as blocks and cylinders[1]. In the B-rep scheme, a solid is modelled by its boundary faces which are bounded regions on planes or surfaces. Without concise mathematical formula for its explicit representation, such a boundary face is indirectly described by its bounding loops, edges and vertices.

The shape of a typical workpiece can be precisely described by both of these two solid modelling schemes. However, a solid represented in CSG is implicitly defined; that is, its shape is not known unless the associated Boolean operations have been evaluated. A solid represented in B-rep is explicitly defined; yet its modelling primitives (i.e. faces, loops, edges and vertices) have been widely interpreted as low-level information and cannot be directly used in developing some downstream applications such as process planning.

In relevant literature, machined feature are generally regarded as high-level information suitable for developing an automated process planning system which can be directly linked to a CAD system[2]. A machined feature such as a slot, a step, or a pocket in practice can be easily associated with a predefined operation plan. Interpreting a workpiece as a set of machined features therefore would facilitate the development of an automated process planning system. A machined feature is named as high-level information because it is a 'macro' description of faces; that is, it generally involvesa set of connected faces rather than a single one.

The problem of recognizing features from a CAD model can be traced back to the pioneer work of Kyprianou, reported in 1980, at the University of Cambridge[3]. His work was intended to develop methods for recognizing some depression and protrusion features from a B-rep model in order to give the GT code description of a part. The intended application of the GT codes was for the development of an automated process planning system.

Since then, much research work (at least over 30 journal papers) on solving the feature recognition problem has been published. These studies in general can be classified into three categories according to the CAD models they used. That is, their investigated

Department of Industrial Engineering and Management, National Chiao Tung University, Hsin Chu, Taiwan, ROC
*School of Industrial Engineering, Purdue University, West Lafayette, IN 47907, USA
Paper received: 1 May 1995. Revised: 4 October 1995

problems were defined as recognizing features either from B-rep, from CSG, or from an enhanced model based on B-rep or CSG. Of these studies, most are essentially based on B-rep.

Significant milestones have been set up by these studies. However, the proposed techniques are generally domain limited or not satisfactorily robust in recognizing features, especially in dealing with interacting features. An interacting feature is the result caused by the intersection of multiple features, an example of such a feature is the intersection result of two simple slot features (*Figure 1a*).

Note that it would be possible to find an existing technique to recognize a particular interacting feature such as the one in *Figure 1a*. However, in general cases, almost all the feature recognition techniques are not robust enough in recognizing the interacting features. That is, a novel interacting feature can always be proposed so that an existing feature recognition technique would find its lakc of robustness.

Recognizing interacting features from CAD models seemed to be a formidable problem. Another way of developing an automated process planning system from CAD is by adopting the concept of design-by-features, which is intended to bypass the burden of recognizing features. The concept of design-by-features advocates that machined features should be taken as the description primitives in designing a part; the resulting workpiece representation would therefore facilitate the tasks of process planning. A design-by-feature, also known as a feature-based, CAD system would freatly enhance the productivity of designers if the primitive features are design features. However, if the available features are limited only to machined features, this amounts to a request to designers to perform part of the process planning tasks and would increase the burden on designers.

Moreover, a design-by-feature approach cannot completely release the burden of feature recognition if protrusion features are to be included and interaction of features are allowed. For example, adding a protrusion feature (e.g. a boss) on a depression feature (e.g. a slot) would result in a significant change on the depression feature. That is, the depression feature or slot now cannot be produced by a typical operation plan for slots due to the boss obstacle. In addition, interaction of features can be so widely varied that a user-defined depression feature might subsequently be encapsulated by a user-defined protrusion feature. Such an interaction would finally cause the function and shape of the depression feature to totallyh disappear from the part. Therefore, in worst-case analysis, a feature-based CAD system would essentially amount ot a CSG model. This denotes that the traditional feature recognition problem canot be totally bypassed in a feature-based CAD system.

This paper is intended to analyse the methodology of existing techniques developed for solving the feature recognition problems based on B-rep. Although two surveyt papers on feature recognition techniques have been published[4,5], we focus on analysing the reasons why machined features cannot be robustly recognized. Solving the machined feature recognition problem has been widely regarded as an essential cornerstone for developing an automated process planning system. The pros and cons of this verdict is also discussed. Finally, a novel cutter selection methodology, known as process

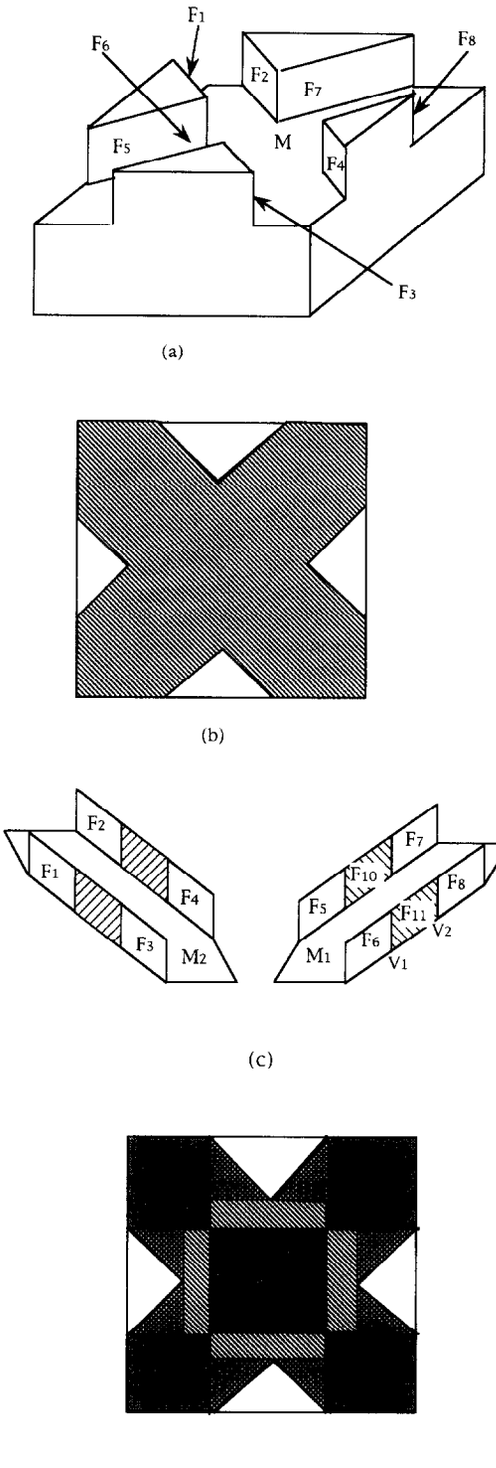


Figure 1 An interacting feature consists of two simple slots; (a) pictorial view, (b) 2D projection view, (c) missing information to be recovered, (d) the decomposition result of the interacting feature by Wang's algorithm

requirement modelling[6], is introduced. This methodology seems to provide a promising way to the robust recognition of machined features.

DEFINITIONS OF FEATURES AND THEIR IMPLICATIONS

In relevant literature, numerous ways of defining features have been proposed. These definitions in general can

be classified into two main categories: conceptual and operational definitions. A conceptural definition gives the general description of all concerned feature; such a description is too rough to be directly implemented. Alternatively, an operational definition is intended to give a precise description on a particular type of features so that it can be coded in a computational form and operated on a computer. These two types of definitions together with their implications are discussed below.

Conceptual definitions of features

Among the previous conceptual definitions of features, Pratt and Wilson in 1985[7] gave a relatively broader one; 'A feature is a region of interest on the surface of a part'. Such a region of interest (known as form features in their paper) can be used in various applications such as machining, assembly, fixturing, finite element analysis, etc. Form features therefore can further be classified into machined features, assembly features, fixturing features and analysis features, depending upon their intended applications[8]. In this paper, we are concerned only with the machined feature; that is, features relevant to the machining of workpieces.

Conceptual definitions of machined features

Many conceptual definitions for machined features have been proposed, which might be named with different terms. Choi *et al.*[9] defined a machined feature (known as machined surface in their paper) as follows: 'a portion of workpiece generated by a certain mode of metal cutting'. Joshi and Chang[2] gave a similar definition: 'regions of a part having some manufacturing significance in the context of machining'.

Implication: defining a mapping between feature space and machining process space

The above two conceptual definitions of machined features, together with some other similar ones[10–13], are intended to describe a mapping between two distinct spaces: the feature space and the machining process space (*Figure 2*). An element in the feature space is aregion or a portion on a workpiece, which may involve a set of connected faces or a volume. An an element in the machining process space is a machining mode, which may denote a type combination of machine tools and cutters.

The proposal of such a mapping can be understood by the observation of the three slot features in *Figure 3*. These slot features, identical in their shape and varying in dimensions, can both be generated by a certain machining mode, say a 3-axis milling machine and a flat-end milling cutter.

An analogy between the feature space and the machining process space can be built up from the above analysis. For the three slot features, the characterization of their likeness in shape would correspond to the requirement of a common machining mode. And the distinction in their dimensions would correspond to the required variation in the detailed specifications of machine tools and cutters.

Based on such an analogy, the recognition of features would correspond to the process planning of manufacturing a workpiece. Feature recognition therefore has been widely regarded as an indispensable research issue in developing an automated process planning system. That is, many researchers interpret the problem of selecting machining modes as a problem of recognizing machined features.

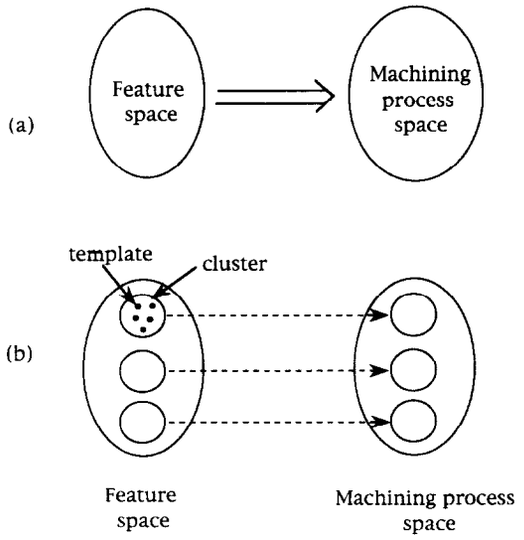


Figure 2 (a) A conceptual definition of machined features provides a mapping between the feature space and the machining process space. (b) An operational definition of machined features is to describe a cluster of features by choosing some templates as samples, then characterizing their common characteristics as the boundary constraints of the cluster

The proposal of a conceptual definition for machined features implies the creation of a mapping between the feature space and the machining process space. However, as stated above, such a description is too rough to be directly implemented on a computer; its realization or operational definitions should be developed for dealing with various types of features and machining modes.

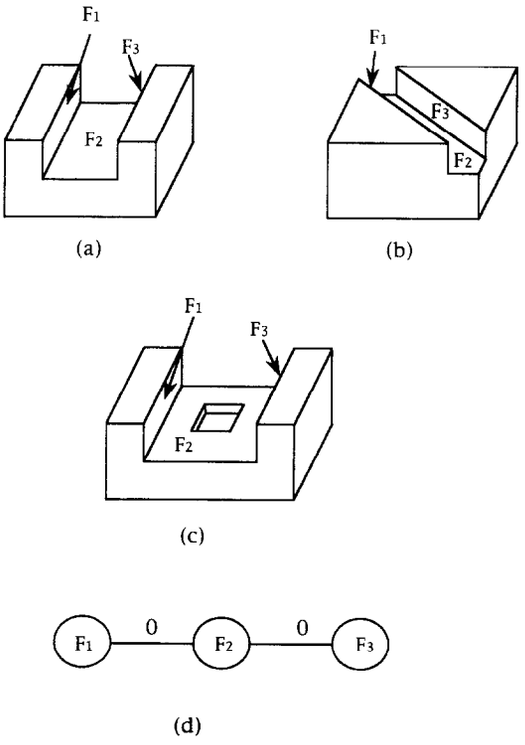


Figure 3 Three simple slots and their graph representation proposed by Joshi and Chang

Operational definitions of machined features

In the literature, an operational definition for features is generally proposed for describing a class of machined features, which are similar in shape. A typical example for defining a class of features, given by Joshi and Chang[2], is quoted below.

- A slot feature is composed of three faces F_1, F_2 and F_3.
- 'F_1 is adjacent to F_2'.
- 'F_2 is adjacent to F_3'.
- 'F_1 forms a concave angle with F_2'.
- 'F_3 forms a concave angle with F_2'.

According to this definition, each of the three face sets ($\{F_1, F_2, F_3\}$) in *Figure 3* can be regarded as a valid slot, even though they are varied in dimensions, orientations, and adjacency relationships with the other faces.

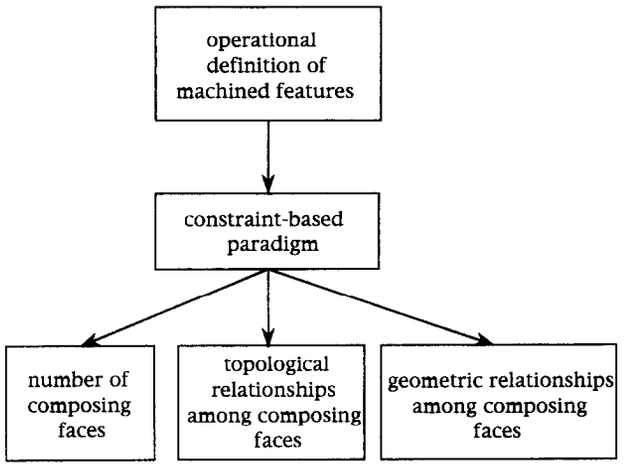


Figure 4 A typical procedure for giving the operational definition for a class of features

First implication: constraint-based description

In the above example, an operational definition is described by a set of 'constraints'. That is, a valid slot feature should satisfy the above five constraints. The first one constrains the number of composing faces. The next four constrain the relationships among these composing faces, which usually are further classified into topological (e.g. constraints 2 and 3) and geometric (e.g. constraints 4 and 5) relationships.

In such an approach, the underlying definition for a class of features is based on a 'three-categories-constraints' paradigm; that is an operational definition involves three categories of constraints: number of composing faces, topological relationships and geometric relationships among composing faces (*Figure 4*). Most relevant studies adopt such a paradigm in giving their operational definitions for typical machined features, such as holes, slots, steps, pockets, etc. However, for a particular class of features, the constraints proposed by different researchers might be various.

Second implication: clustering of features

Another implication for giving operational definitions of features is for 'clustering' purposes. That is, in the universal feature space, the set of any region on any workpiece in the world, similar regions (or features) should be clustered into a class. Each member in a particular feature class should be reproducible by similar or common metal cutting modes. The 'three-categories-constraints' paradigm then implies a way to clarify the boundary of a cluster. That is, a region on a workpiece satisfying the required constraints of a cluster would be attributed as a feature member; otherwise, it would be rejected.

Third implication: sampling of templates

One interesting question may now be raised: how to set up a set of constraints for defining a particular class of machined features. The number of all members in a particular class of features might be infinite.

In previous relevant studies, typical ways of defining the boundary of a cluster are based on a 'sampling' or 'templates-representing-all' methodology. That is, some typical features/regions, belonging to this cluster and finite in their number, are manually chosen as templates for representing the whole cluster. These templates are analysed to derive their common characteristics and then proposed as the constraints for forming the particular class of features. For example, adopting the three slots in *Figure 3* as templates, one may derive the five constraints proposed by Joshi and Chang[2] or another constraints set.

In summary, an operational definition for features is a set of constraints, which is intended to define the boundary for forming a cluster of similar regions/features. The idea for proposing such a constraint set is based on a 'sampling-and-characterization' process. That is, typical features are first sampled as templates and subsequently analysed in order to derive their common characteristics, which are then used to create the constraints set.

REPRESENTATIONS OF FEATURES

To describe the operational definitions of features, many representation schemes have been proposed in previous literature. These representation schemes in general can be classified into three types: rule-based; graph-based; and syntactic-pattern-based approaches.

Rule-based representation scheme

Typical examples of the rule-based approach can be illustrated by Henderson's study[10]. In this approach, a set of heuristic rules are used to describe the operational definition for a class of features. For example, by the observation of the three templates shown in *Figure 3*, one might propose the following set of heuristic rules to describe a slot feature, which are a little different from that proposed by Joshi and Chang[2].

- A slot is composed of three faces $\{F_1, F_2, F_3\}$.
- Face F_1 is adjacent to face F_2.
- Face F_2 is adjacent to face F_3.
- Face F_1 and F_3 are parallel.
- Face F_1 forms a 90° angle with F_2.
- Face F_3 forms a 90° angle with F_2.

Such heuristic rules for describing features are generally coded in Prolog, Lisp or some other languages

which are known as the building tools for developing expert systems. Rule-based approaches therefore are also regarded as an expert system approach. Previous studies adopting the rule-based representation scheme involves References 14 and 15.

Graph-based representation scheme

In the graph-based approach, a class of features is first modelled by a graph structure, which depicts the required topological and geometric constraints of for identifying a feature. Then the graph structure may be coded in various computational forms.

The description of a slot feature, introduced earlier in this paper[2], is in fact a typical example of the graph-based approach, even though it was presented in a form of five rules. Referring to *Figure 3*, one can model the three templates by a graph structure consisting of three nodes. In this graph, a node denotes a face and an arc denotes an adjacency relationship between two nodes/faces; an attribute value 0 or 1 can be assigned on each arc to denote that the intersection angle of the two adjacent faces is concave or convex.

Such a graph structure, when used in describing some other types of features such as steps, holes or pockets, can also be converted into a set of rules. Following the 'three-categories-constraint' paradigm, these rules representing a graph involve three types. The first type describes the required number of faces for comprising a concerned feature (i.e. the number of nodes in the graph structure). The second type gives the topological relationships among composing faces (i.e. the arcs in the graph structure). The third type presents the geometric relationship among faces (i.e. the attribute value on each arc in the graph structure).

Many graph-based representation schemes have been proposed in the literature. Examples of these studies involve References 16–23; among these, the work by Ansaldi *et al.*[16] might be the first one appearing in the literature. The variations of these graph-based representation schemes can be understood by modifying the illustrated graph structure in the following ways:

- labelling nodes to give geometric constraints on nodes[23],
- giving more attributes (i.e. geometric constraints) on arcs[17],
- interpreting nodes as surfaces (instead of faces) and surface intersection curves as arcs[22],
- interpreting nodes as vertices and edges as arcs[19].

By representing a machined feature as a graph, to recognize a feature from a B-rep, which is also a graph, then becomes a matching problem. That is, a subgraph (feature) is to be identified from a larger graph (workpiece in B-rep). A graph-matching problem has been widely known as an NP-complete problem[24]; its computing time in worst case tends to grow exponentially.

Various techniques have been proposed for reducing the computing time in solving the graph-matching problem. Srinivasan[17] simplifies a graph by a tree, and performs tree-matching; Joshi and Chang[2] use some heuristic rules to decompose a graph into several sub-graphs; Corney and Clark[18] use the orientation of faces as a guide to simplify a graph. These techniques are essentially based on an 'information-deletion' process. That is, some heuristics are proposed to systematically delete some information from the graph to make it simplified. This deletion would improve the computation time, but might also cause the problem of robustness.

Graph-based and rule-based representations are mutually convertible

From the above analysis, we already observe that the graph-based representation scheme proposed by Joshi and Chang[2] can be converted into the rule-based representation scheme. One might wonder if any other graph-based representation scheme can also be converted into a rule-based one. This verdict is true because a graph, in whatever way it has been modified, involves only arcs, attributes/labels and nodes, which, respectively, give the constraints of topological relationships, geometric relationships and number of concerned entities (i.e. faces or vertices). Each arc, arc attribute, node label and the number of total nodes can individually be converted into a single rule.

Conversely, any rule-based representation scheme, if its templates for defining features are regarded as a set of connected faces, can also be converted into a graph-based one. This verdict can be understood by interpreting a rule as a constraint to be imposed on a single face or on the relationships of any two faces, which may be neighbouring or non-neighbouring. Taking faces as nodes and edges as arcs, constraints about a single face can be seen as node attributes; constraints about neighbouring faces are arc attributes. Note that a constraint about a pair of non-neighbouring faces can also be interpreted as an arc attribute, if a virtual arc connecting the two non-neighbouring faces in constructed. Therefore, a set of rules for defining features can be represented by a graph structure.

Note that the above verdict theoretically can be extended to other types of rules, which impose constraints on surfaces, loops, edges and vertices, the other four types of geometric entities in a B-rep. The worst case for representing such a rule set by a graph structure is by taking each surface, loop, curve, vertex as a node; constraint on a single node as the node attribute; constraint on any two nodes as arc attributes. Therefore, the rule-based representation scheme and the graph-based representation scheme in fact are mutually convertible.

Syntactic-pattern representation scheme

The syntactic-pattern approach for representing a class of features[9,11,15] can be illustrated by an example of defining slot features, adopted from the work of Choi *et al.*[9]. In their approach, a slot is defined as:

SLOT ::= {SES} SSS {SES}

where SSS is a slot-starting-face, defined by some geometric constraints and SES is a slot-element face, defined by some geometric constraints.

The above expression denotes that a slot is a pattern of

faces which is composed of one SSS face and some SES faces on both sides of the SSS; the expression {SES} emphasizes that the SES faces on each side may be multiple. Such a slot pattern is very good in modelling the interacting slot features as shown in *Figure 5a*. Simple slots can also be dealt with because they are special cases of the interacting ones.

From the above example, we can see that a syntactic pattern for modelling features also follows the 'three-categories-constraints' paradigm. That is, the first two categories of constraints (i.e. the required number of composing faces and the topological constraints among faces) are described in the syntactic rule (e.g. SLOT ::= {SES} SSS {SES}). And the third category (i.e. geometric constraints) are embedded in the definition of the rule primitives (e.g. {SES}, SSS). The syntactic-pattern representation scheme therefore can be interpreted as a rule-based approach.

For example, the above 'syntactic rule' can be described by a set of 'common rules' as below.

Rule 1: A slot is composed of three set of faces $\{F_{1i}, i = 1$ to $n\}$, $\{F_2\}$, and $\{F_{3j}, j = 1$ to $m\}$.
Rule 2: Face F_2 is adjacent to each member of $\{F_{1i}, i = 1$ to $n\}$.
Rule 3: Face F_2 is adjacent to each member of $\{F_{3j}, j = 1$ to $m\}$.
Rule 4 to M: Describing the geometric constraints about faces F_{1i}, F_2 and F_{3j}.

As we can see from *Figure 5b*, such a slot pattern can also be represented by a graph, with the SSS face as the centre node and two sets of SES faces as its neighbouring nodes. As stated, the geometric constraints of SSS and SES faces can be converted into the attributes of the graph arcs and nodes. Therefore, a syntactic-pattern representation scheme for machined features can also be converted into a graph-based one.

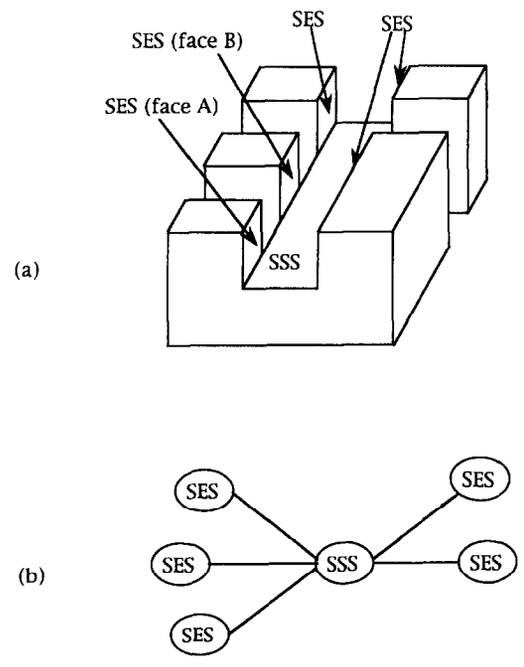


Figure 5 (a) A slot involving more than three faces can be represented by a syntactic pattern. (b) The graph representation of the slot

Configurations of feature recognizers

In summary, the configuration of a typical feature recognizer would involve three modules: feature definition, feature representation, and feature recognition mechanism (*Figure 6*).

The feature definition module is intended to decompose the feature space into a number of clusters. One cluster denotes a class of features which can all be produced by a common machining process mode. As stated, the boundaries for clarifying a cluster are generally described by a set of constraints, by sampling some templates and characterizing them following the 'three-categories-constraints' paradigm.

In the feature representation module, the set of boundary constraints are subsequently modelled by one of the three representation schemes (rule-based, graph-based, and syntactic-pattern-based). These three representation schemes, even though different in their appearance, can be mutually convertible.

In the feature recognition mechanisms, typical pattern recognition techniques (e.g. expert system approach, graph-matching approach, and syntactic-pattern recognition approach) would be correspondingly chosen or developed so as to be compatible with the feature representation scheme. A typical graph-matching problem is NP-complete; therefore some heuristic procedures might be performed in order to simplify the representation of a graph and reduce the computation time.

ANALYSIS OF ROBUSTNESS PROBLEMS

The existing techniques for machined feature recognition are generally deficient in their robustness. That is, for a particular technique, a feature valid in machining practice may not be recognizable (known as type I error) and a recognized feature may not be a valid one in machining practice (known as type II error). The causes of robustness problems will be analysed in the following.

Robustness problems born with feature definitions

The robustness problems are essentially generated from the process of defining machined features. As stated, an operational definition of features is intended to form a cluster in the feature space, where each feature member

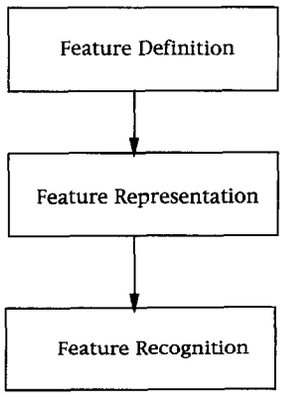


Figure 6 Three major modules of a machined feature recognizer

can be produced by a common machining mode. The boundary of a cluster, which are expressed by a set of constraints, should be appropriately defined; otherwise, it may result in the robustness problems. That is, too small a cluster would cause type I error, and too large a cluster would cause type II error. In the ideal case, we would expect that the set of features producible by a particular machining mode should be equal to the defined feature cluster.

However, such an ideal case seems to be very difficult to be achieved. As stated, the process of giving an operational definition of features involves three major stages (*Figure 7*). First, a sample of feature templates should be chosen to represent the world of the concerned feature class. In this stage, one cause of the robustness problems may occur. That is, the sampled templates might not be enough in number so that some features in the feature class would find no representative templates. Therefore, these features subsequently would not be recognizable by the proposed recognizer.

In the second stage, the sampled templates have to be analysed to derive their commonality, which would then be expressed as the boundary constraints of the feature cluster. The existing techniques for characterizing the sampled templates in general are heuristic. That is, the boundary constraints of a feature cluster are derived by only observing the templates' shape, which are not validated by a mathematical proof. Therefore, the heuristic procedure of characterizing commonality would be another cause fo robustness problems.

In the third stage, the boundary constraints for a feature cluster would be expressed by one of the three representation schemes (rule-based, graph-based, and syntactic-pattern-based approaches). The three representation schemes are mutually convertible; this implies that they are equally capable in the sense of modelling a feature cluster. That is, the robustness problems cannot be cured by only choosing different representation schemes.

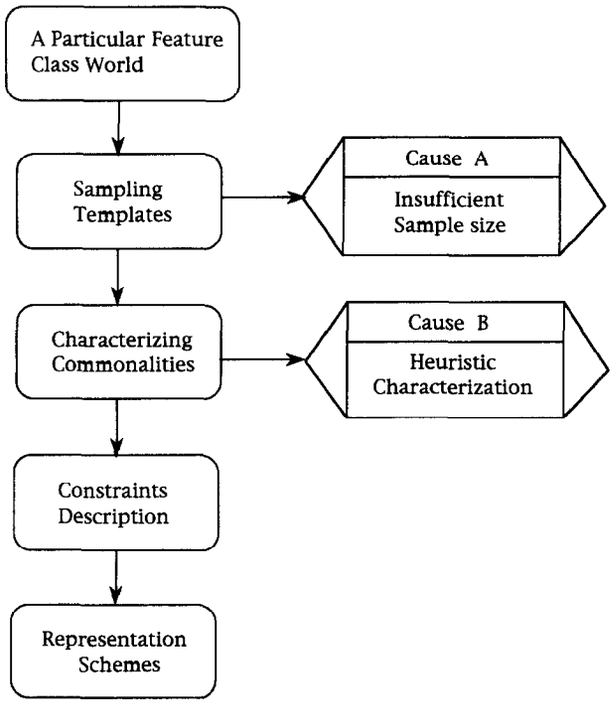


Figure 7 A flow chart for analysing the robustness problem that occurs in the existing feature recognition techniques

From the above analysis, it is known that the robustness problem is due to insufficient and inappropriate modelling of feature information, which occurs in the stage of defining features. Suppose the required information for modelling a class of features is sufficient and appropriate (even though this may not be obtainable), one might question whether any of the three representation schemes be comprehensive enough to model the information. This verdict should be positive, because the B-rep scheme for modelling a workpiece is essentially a graph, and a feature is part of a workpiece. The feature information extracted from a B-rep therefore should be able to be represented by a graph, and can be modelled by any of the three representation schemes because they are mutually convertible.

In summary, the robustness problems are essentially caused by the definition of features. There are two major causes. One is due to the insufficient sample size of templates, and the other is due to the heuristic characterization of their commonalities. It seems that these two major causes might be lessened by applying some techniques but are very difficult to be removed completely. This verdict will be illustrated below.

Cause A: insufficient sample size of templates

To model the world of a particular feature class, e.g.slots, the representative templates are very difficult to be enumerated exhaustively, especially when the interacting features are to be included in the feature world. For example, the face-adjacency-graph for modelling a slot (*Figure 3*) is created through characterizing three templates of simple slots such as those shown in this figure. Such a slot definition would cause vagueness when used in recognizing the interacting slot shown in *Figure 5*. That is, the two SES faces (faces A and B) and the SSS face, as a whole, would be recognized as a slot, yet such a recognition is inappropriate in practice.

Including the feature in *Figure 5a* as a slot template, the slot definition might be represented as a syntactic pattern shown in *Figure 5b*. Such an enhanced definition still might cause vagueness in recognizing some other interacting features. For example, each of the two interacting slots shown in *Figure 1a* would not be recognized as a slot, because an SSS face in its original definition involves only four edges[9]. The candidate SSS face (face M) in *Figure 1b* involved 16 edges and cannot be a valid SSS face. To include the two interacting slots in *Figure 1a*, the constraints for clarifying slot definitions should further be relaxed.

However, such a relaxation might cause another side-effect. The relaxation of a slot definition would enlarge the domain of recognizable slots; however, the enlarged domain might intersect with the domains of some other types of features. This would cause another type of vagueness.

In summary, features theoretically can be mutually interacted in a combinatorial manner. This is, the number of patterns for modelling interacting features is infinite and cannot be exhaustively enumerated. Even though the representative templates are sufficiently sampled, the characterized domain might unexpectedly intersect with other feature domains—this is undesirable according to the criterion of robustness recognition.

Cause B: heuristic characterization of commonalities

Characterizing commonalities from the sampled templates is generally a heuristic process and might cause the robustness problem. Examples for characterizing the commonalities of slot features are illustrated below.

By observing the three slots in *Figure 3*, one might propose the following heuristic rule set (Rule Set 1) for defining the slot features.

Rule Set 1:
- *Rule 1*: A slot is composed of three faces $\{F_1, F_2, F_3\}$.
- *Rule 2*: Face F_1 is adjacent to face F_2.
- *Rule 3*: Face F_3 is adjacent to face F_2.
- *Rule 4*: Face F_1 is perpendicular to face F_2.
- *Rule 5*: Face F_3 is perpendicular to face F_2.

It seems quite reasonable to define slot features by the above heuristic rule set. However, a counter example (*Figure 8a*) denotes that the rule set should be enhanced; otherwise a blind step would be recognized as a slot. Suppose a new heuristic rule as stated below is additionally included in order to form Rule Set 2, which would exclude the counter example (*Figure 8a*) from the domain of slot features.

Rule Set 2:
- Rules 1 to 5 in Rule Set 1 are included.
- *Rule 6*: Face F_1 is parallelto face F_3.

However, the enhanced rule set is still insufficient by observing a counter example in *Figure 8b*, which is not a slot in practice yet satisfies all the constraints in Rule Set 2. One might propose to further enhance the rule set by emphasizing that the intersection angles in Rules 4 and 5 should be concave.

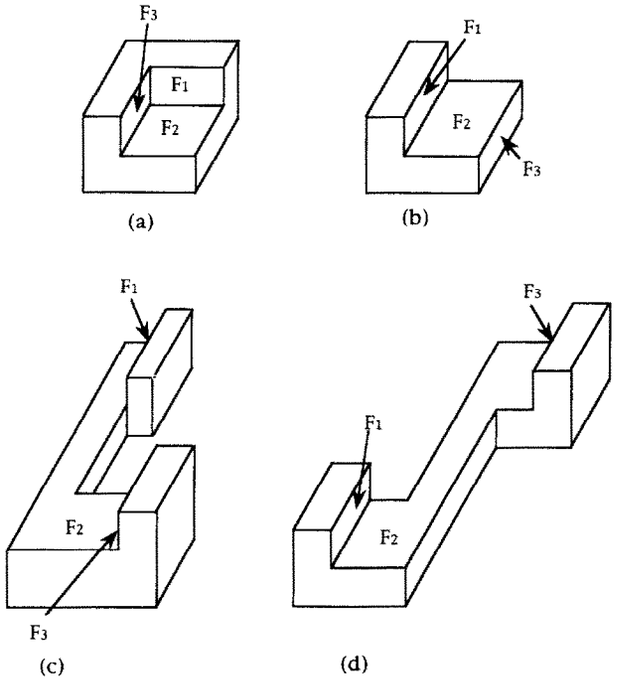


Figure 8 Counter examples for depicting the robustness problems occurred in the process of heuristically characterizing representative templates

Rule Set 3:
- Rules 1, 2, 3 and 6 in Rule Set 2 are included.
- *Rule 4*: Face F_1 forms a 90° angle with face F_2.
- *Rule 5*: Face F_3 forms a 90° angle with face F_2.

After two stages of enhancement, Rule Set 3 is still insufficient by observing a counter example in *Figure 8c*, which satisfies all the constraints in Rule Set 3 yet is not a slot in practice. The rule set might further be enhanced as follows in order to exclude the counter example.

Rule Set 4:
- Rules 1 to 6 in Rule Set 3 are included.
- *Rule 7*: The outward normals of faces F_1 and F_3 are opposite.

Now the rule set seems quite comprehensive. However, one can still propose a counter example to highlight its insufficiency. For example, the three faces $\{F_1, F_2, F_3\}$ in *Figure 8d* meet all the constraints in Rule Set 4; however, in practice they are better regarded as two steps because faces F_1 and F_3 are too far away. One might therefore enhance the rule set as follows.

Rule Set 5:
- Rules 1 to 7 in Rule Set 4 are included.
- *Rule 8*: The distance between faces F_1 and F_3 should not be too 'far'.

By observing the above rule set, we can see that the description of Rule 8 is vague because of the requirement of justifying a distance as too 'far' or not, where the semantics of 'far' is vague. On the other hand, it also seems inappropriate by giving a definite upper bound for clarifying the vagueness.

Note that the above rule sets are essentially created based on simple slots (i.e. a slot involves one SSS face and two or more SES faces). Some other problems may arise when these rule sets are to be applied on interacting slot features. For example, the two interacting features in *Figure 1a* can be recognized in involving 8 slot features (i.e. $\{F_1, M, F_2\}$, $\{F_1, M, F_4\}$, $\{F_2, M, F_3\}$, $\{F_3, M, F_4\}$, $\{F_4, M, F_6\}$, $\{F_5, M, F_8\}$, $\{F_7, M, F_6\}$, $\{F_7, M, F_8\}$).

However, each of the 8 slot features in practice cannot be individually regarded as a slot feature. They have to be merged into two slots as shown in *Figure 1c*. Note that the face M now has to be decomposed into two overlapping faces M_1 and M_2 (i.e. $M_i \cup M_2 = M$). In addition, faces F_5 and F_7, together with a newly created face F_{10}, have to be merged into one face, likewise for faces F_6, F_{11} and F_8, in order to satisfy the first rule in the above rule set. In dealing with interacting features, various techniques for decomposing faces and creating new faces have been proposed previously. However, they are in general heuristic procedures and therefore deficient in their robustness.

From the above illustration, we can see that heuristic characterization of sample templates, even carefully performed, might still unexpectedly cause the robustness problem.

TECHNIQUES FOR DEALING WITH INTERACTING FEATURES

As discussed above, the recognition of interacting

features is a crucial research issue in developing a feature recognizer. Various techniques for solving this issue have been proposed. These techniques in general can be classified into the following five categories:

- Generic representation schemes for features.
- Recognizer with learning capability.
- Missing information recovery.
- Feature-based system mixed with B-rep.
- Other works.

Generic representation schemes for features

The first approach for dealing with interacting features is by developing a generic representation scheme which would be able to model both interacting features and simple features. That is, simple features are taken as special cases of interacting features. And such a feature recognizer would be able to recognize both interacting and simple features by the same procedure.

A typical example of this approach is the work done by Choi *et al.*[9] Their proposed syntactic pattern representation scheme is good at modelling some interacting features as shown in *Figure 5*. However, as discussed above, such a representation scheme would cause vagueness in dealing with the interacting features as shown in *Figure 1*.

Recognizers with learning capability

The second approach for dealing with interacting features is by enhancing the recognition mechanism through a learning process. That is, if an interacting feature cannot be well recognized by the existing recognizer, this feature would be taken as a training template for enhancing the capability of the recognizer. Through this learning mechanism, the capability of the recognizer would be gradually improved.

One example of this approach is the work proposed by Chan and Case[26]. The idea of this approach is quite novel. However, some side-effects might have to be considered. In order to deal with an unrecognizable feature, a new set of inference rules have to be included in the recognizer. Many new rule sets then would be gradually included in the system. It is important to assure that the semantics of these rule sets be mutually exclusive. Otherwise, a feature might be classified into more than one type and vagueness would occur.

The side-effect can be understood in another way. Suppose the unrecognizable features have been identified before the initial creation of the recognizer and are regarded as the sampling templates. Such a heuristic characterization of templates would tend to cause the robustness problem. Therefore, the robustness issue does exist whether the unrecognizable features are simultaneously proposed before the creation of the recognizer, or identified one by one after the creation of the recognizer.

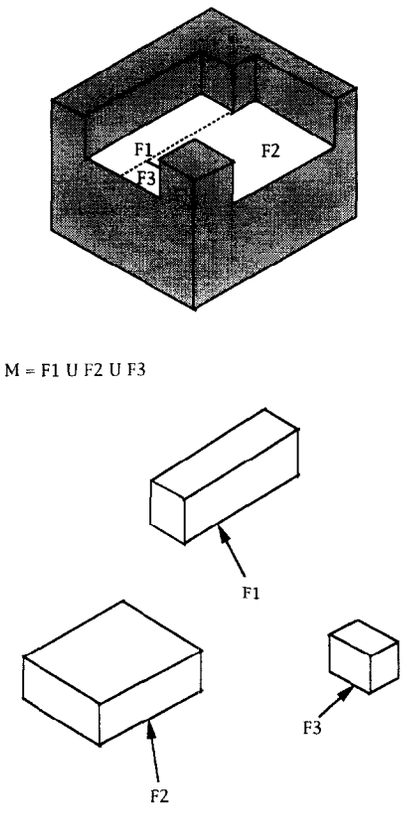


Figure 9 The decomposition result of an interacting feature proposed by Wang's algorithm

Missing information recovery

Referring to *Figure 1c*, in the creation of interacting features, some face information of simple features might become missing. In an attempt to decompose an interacting feature into multiple simple features, the information of these missing faces requires recovery. Therefore, another approach for dealing with interacting features aims to develop techniques for recovering the missing information. The missing geometric entities may involve edges or faces, and their purposes of recovery may be fo r the separation of individual simple features (e.g. edges $\overline{V_1V_2}$ in *Figure 1c*) or for the information completeness of simple features (e.g. faces F_{10}, F_{11}, F_{12}, F_{13} in *Figure 1c*).

One typical example of this approach is the work done by Joshi and Chang[2]. They propose some heuristic rules which would effectively recover the missing information for two types of interacting features. This study sets up a good milestone; however, interacting features in practice should be more than two types. For robust extension of this approach, it is essential that the proposed heuristic rules be rigourously validated.

Another noted example of this approach in the work by Wang[27]. This study proposes a polynomial time alforithm to solve the machined feature recognition problem which before then had been widely interpreted as a graph-matching problem (an NP-complete problem). Also, interacting features can be well solved by this algorithm in a certain workpiece domain.

In this study, the workpiece should first be transformed into a rectilinear form. That is, the workpiece should be modified so that each resulting face should be parallel to one axis of the world coordinate system. A rectilinear but non-rectangular face (e.g. face M in *Figure 9*), which implies the existence of an interacting feature, should be decomposed into a set of rectangles. Then, for each rectangle, a block-volume-growing

procedure would be performed to recover the missing information for identifying simple features. Each rectangle implies the existence of a simple feature.

This technique is quite creative yet still leaves some space for further investigation. First, the techniques seems only applicable to 'near-rectilinear' workpieces. That is, most of the faces on a workpiece should be rectilinear; otherwise the recognition result would be quite unusual. For example, the two interacting slots in *Figure 1a* would be decomposed or recognized as involving 17 simple features (9 blocks and 8 prisms in the sense of considering their volume to be removed, see *Figure 1d*). Such a decomposition is quite from the usual practice. Moreover, the 17 simple features cannot be merged back to obtain two separated simple slots.

Second this study assumes that all the simple features are of rectangular shape. A depression region as shown in *Figure 9* would be decomposed into three simple rectangular features. Suppose a recognized simple feature is intended to correspond to a machining mode. Such a process planning for machining the depression region seems quite unutual. It seems that a procedure for merging the rectangular simple features into a single feature is needed, where the resulting feature corresponds to a machining mode.

Feature-based system mixed with B-rep

The fourth approach for dealing with interacting features is by using a hybrid solid modeller, which is feature-based design system enhanced with B-rep. Based on such a solid modeller, the process of separating interacting features into simple features will be freatly simplified because the information of the built-in features provide clues for the separation.

One typical example of this approach is the work done by Laako and Mäntylä[22], others involve References 21 and 28. In average case analysis, such approaches would greatly reduce the complexity of recognizing interacting features. However, in worst case analysis, a feature-based design system as stated is just like a typical CSG system; such a hybrid representation in this sense would not help much in dealing with interacting features.

Other works

In dealing with interacting features, some other approaches, which combine or modify some of the above four techniques, have been proposed. Most of them are essentially based on heuristic algorithms, and unavoidably would lack robustness. Examples of these studies are discussed below.

Gavankar and Henderson propose a heuristic algorithm by which nested depression and nested protrusions (both are interacting features) can be separated[29]. Chamberlain *et al.* adopt Wang's back-propagation algorithm[28] and propose an approach to decompose a protrusion feature into several depression features, where such a protrusion feature is generally relevant to the occurrence of an interacting machined feature[30].

Prabhakar and Henderson develop a neural network approach which can recognize intersecting slots[31]. Marefat and Kashyap propose a hypothesis-generation-elimination approach to identify primitive features from interacting features in which some missing information (known as virtual links) are heuristically recovered[23]. Corney and Clark propose an algorithm which can identify multiply connected depressions or protrusions and provide multiple interpretations of a 2.5D component's shape[32]. Donaldson and Corney define several classes of interacting features and develop a rule-based system to identifying interacting features[13].

Some other relevant works which study the impact of interacting features on process planning have also been published[33–35]. In Young and Bells' study, the impact of interacting features on a design-by-feature CAD system is extensively explored[36]. There are numerous ways to decompose an interacting feature into several simple features. These relevant studies therefore provide a reference to justify the usefulness of interacting feature decomposition techniques.

PROS AND CONS FOR RECOGNIZING FEATURES

As stated, the original purpose for recognizing machined features is to develop a CAPP system which can directly interface CAD and CAM. A process plan in general involves a series of operation sheets, and each operation sheet gives the required machine, cutter and fixture. The recognition of machined features in fact is an intermediate step, which subsequently has to be converted into operation plans. One might question the possibility of developing a procedure which can directly generate the operation plans from a CAD system, rather than travelling through an intermediate state.

In the literature, such approaches have been proposed. Typical examples involve the work done by Grayer[37] and that by Armstrong[38]. In Grayer's work, a part is sliced with a number of parallel planes so that it can be decomposed into a set of disjoint laminae. For each lamina, a cutter can be manually chosen in order to generate the NC code for machining the lamina.

In Armstrong's work, a 3D space enclosing a part is decomposed into a set of spatially ordered cells, so that each cell can be classified as either a stock cell, a part cell, or an on-boundary cell. The stock cells would be used to generate the NC code of roughing cuts. The on-boundary cells would be used for generating the NC codes of finishing cuts. And the diameter of cutter in both finishing and roughing cuts is determined by the cell size.

In these two studies, a process plan can be automatically generated. However, their major deficiency is that the cutter size is predetermined and the proposed process plan may not be optimum. For example in finishing a slot, an appropriate selection of cutter would require only one stroke to generate the slot. Conversely, if the cutter size is not appropriately selected, it might require multiple strokes to generate the slot and would increase the machining time.

This example highlights the significance of recognizing machined features. That is, in the sense of proposing an optimum process plan, the recognition of machined features would provide a space for selecting a more suitable cutter. From this point, the recognition of machined features can be interpreted as an optimum cutter selection problem.

PROCESS REQUIREMENT MODELLING APPROACH

As discussed above, one ultimate goal of recognizing machined features is to select appropriate machining processes so that multiple faces can be simultaneously machined by a single cutter in order to reduce the machining time. Based on this verdict, techniques developed for automatically selecting cutters from CAD systems are relevant to the feature recognition problem and should be reviewed.

In the following, a cutter selection technique, known as process requirement modelling, is introduced. This technique proposed by Wu and Liu[6] can provide an optimum selection of cutters for machining a part, based on the production criteria and the available manufacturing facilities. In their study, the feature recognition problem was bypassed or it was interpreted as a cutter selection problem. However, if needed, their methodology can also be used in recognizing machined features.

In their study, a machining process, which involves a machine, a cutter and a fixture, is characterized by only two elements. That is, a machine tool together with a set-up of the workpiece is represented by a cutter approach direction. A machining process therefore can be represented by a combination of a cutter approach direction and a cutter. Furthermore, only six predetermined cutter approach directions are concerned in this study; therefore the machining process planning problem becomes a cutter selection problem.

Single face modelling

In most previous studies, a machined feature is generally defined as a set of connected faces which can be simultaneously generated by a common machining process. Typical approaches to describe a machined feature are in an attempt to model its composing faces and their topological and geometric relationships. However, the number of machined features which share a common machining process is almost infinite. And such features could be so widely varied in their shapedue to the occurrence of interacting features. Therefore, developing a feature representation scheme for modelling a machining process would become a formidable problem, because it is essentially an infinite-to-one mapping problem.

In Wu and Liu's study, they avoid the track of modelling multiple faces and focus on modelling a single face by its process requirement. In their definition, the process requirement of a face explicitly describes the specifications of acceptable cutters in each cutter approach direction. Such specifications give the lower bound or upper bound of cutter parameters. The process requirement of a face would therefore correspond to a set of cutters, if the available cutters in the production system has been explicitly stated.

Based on process requirement modelling, a set of connected faces would be recognized as a machined feature if these faces have a common acceptable cutter and can be simultaneously machined by the cutter. In such an approach, the machined feature recognition problem would become a problem of finding the intersection of cutters sets rather than finding graph isomorphism and turns out to be relatively easy.

Representation of process requirements

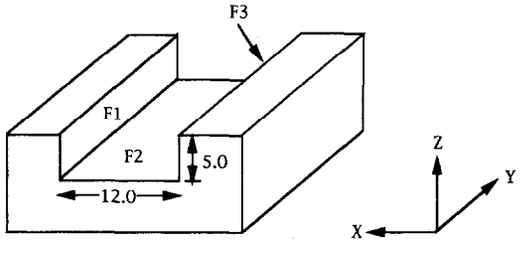
The process requirement of a face (e.g. faces F_1, F_2 and F_3 in *Figure 10*) can be represented as shown in *Figure 10b*. In this table, only the $+Z$ cutter approach directions and three types of cutters (flat-end, torous-end, ball-end milling cutters) are concerned. And four cutter parameters as stated below are used to describe the acceptable range of cutters:

- minimum tool length;
- maximum peripheral radius;
- maximum nose radius;
- minimum nose of radius;

An algorithm for deriving the process requirement of a face from its workpiece B-rep has been proposed[6]. In a sense, the process requirement essentially describes the constraints on selecting cutters for machining a face. These constraints are imposed by the geometric properties of the face, its neighbourign and its non-neighbouring faces. As shown in *Figure 10b*, the minimum tool length of face F_1 is 5 units, which implicitly states the depth of the face viewed from the $+Z$ direction. The maximum peripheral radius of face F_1 is 6 units which denotes the constraints imposed by face F_3. That is, increasing the value of this parameter would cause undesirable overcutting on face F_3. The lat two parameters are for defining the ball-end and torus-end milling cutters, in which no value is given because only flat-end milling cutters are acceptable if the workpieces in *Figure 10* are to be machined from the $+Z$ approach direction.

Feature recognition and process requirement modelling

Currently, a study applying the process requirement modelling methodology to solve the machined feature



(a)

Cutter Parameters	F1	F2	F3
D1	5.0	5.0	5.0
D2	6.0	6.0	6.0
D3	*	*	*
D4	*	*	*

(b)

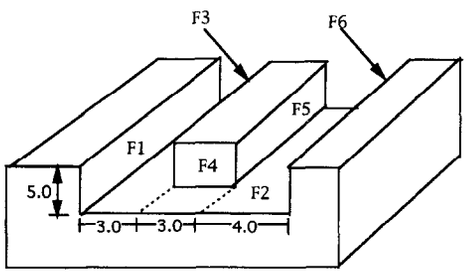
Figure 10 Process requirement modelling for a simple slot which involves three faces

recognition problem is being investigated by authors. Some of the observations and ideas are discussed below.

In *Figure 10a*, the three faces F_1, F_2, and F_3 constitute a slot feature. Such a simple slot feature can be recognized by reviewing their process requirements as shown in *Figure 10b*. That is, the intersection set of the three process requirements would provide a space for finding an optimum cutter which can simultaneously machine these three faces in a single cutter path. Suppose such a cutter can be found, these three faces can be recognized as a slot feature. Intuitively, the optimum cutter could be found to be with 5 units on the first parameter, and 6 units on the second one and the last two parameters are null because only flat-end milling is acceptable.

In addition, the process requirement modelling approach seems quite promising in solving the recognition of interacting features. As shown in *Figure 11a*, there are 6 faces $(F_1, F_2, \ldots, F_6)$ on a workpiece; these six faces can be interpreted as interacting features involving two slot features and one step feature. By considering faces F_1, F_2 and F_3 only, we would find their common acceptable cutter set $\{D_1 = 5.0, D_2 = 1.5\}$ (*Figure 11b*). The cutter with $D_1 = 5.0$ and $D_2 = 1.5$ could be used to machine these three faces like a slot. Likewise, by considering faces F_5, F_2 and F_6 only, the cutter with $D_1 = 5.0$ and $D_2 = 1.5$ could be used to machine these three faces like a slot. In a similar manner, suppose only faces F_2 and F_4 are concerned, a step feature would be recognized.

Alternatively, by reviewing the process requirements of these faces, we would find that these 6 faces can be simultaneously machined by a cutter with $D_1 = 5.0$ units and $D_2 = 1.5$ units. Producing these 6 faces as a whole can be seen as machining a protrusion region. In this sense, these 6 faces form a single machined feature. That is, in the process requirement modelling approach, a set of connected faces can be recognized either as a single feature or an interacting feature. This characteristic is good in the sense of providing more flexibility in process planning.



(a)

cutter parameters	F1	F2	F3	F4	F5	F6
D1	5.0	5.0	5.0	5.0	5.0	5.0
D2	1.5	1.5	1.5	1.5	2.0	2.0
D3	*	*	*	*	*	*
D4	*	*	*	*	*	*

(b)

Figure 11 Process requirement modelling for an interacting features which involves six faces

CONCLUDING REMARKS

This paper analyses the methodology of existing techniques for recognizing machined features based on B-rep, which in general are not robust in recognizing interacting features. Three popular feature representation schemes (rule-based, graph-based and syntactic-pattern-based approaches) have been illustrated to be mutually convertible. Therefore, the robustness problem cannot be solved by only choosing different feature representation schemes.

However, the above verdict does not imply there is an inherent restriction on the modelling capability of the three representation schemes. Alternatively, it means that if the extracted information for defining a class of features is insufficient or inappropriate, one cannot overcome this deficiency by solely using different representation schemes. This verdict hopefully would encourage researchers to focus on developing a new way to define features (e.g. process requirement modelling approach[6]) rather than on modifying the existing feature representation schemes. From our analysis, the modelling capability of the graph-based and the other two representation schemes would be sufficient in describing a class of features if their boundary of constraints can be explicitly and precisely characterized, yet the demanded explicitness and preciseness seems unobtainable.

Robustness problems have been explained to be born with the definition of machined features. In the process of defining machined features, representative templates are first sampled and subsequently characterized to define a class of features. In this process, there are two points which might cause robustness problems. First, the number of representative templates may not be enough in number. Second, the processes of characterizing the templates are almost all heuristic and may not be valid.

Existing techniques for recognizing interacting features in general can be classified into five approaches: (a) developing generic representation schemes for features; (2) developing recognizers with learning capability; (3) recovering missing information; (4) adopting a feature-based CAD system mixed with B-rep and (5) other works: the combination and modification of the above four approaches. These approaches, in worst case analysis, cannot solve well the interacting feature recognition problems.

The ultimate purpose for recognizing machined features is for selecting machines, cutters and fixtures. Suppose that machine and fixtures have been determined, a machined feature recognition problem in fact is a cutter selection problem. That is, appropriate cutters are expected to be found so that multiple faces on a workpiece can be simultaneously machined in order to reduce machining time.

A cutter selection technique known as process requirement modelling is introduced. This tchnique emphases modelling the manufacturability of a single face, and uses the derived manufacturability to select a common cutter for machining multiple faces. These multiple faces which share a common cutter are then interpreted as a machined feature. This methodology seems quite promising in developing a technique for

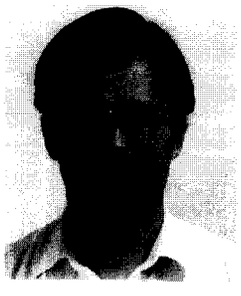
solving the recognition problem of interacting features, which are currently being investigated by the authors.

ACKNOWLEDGEMENTS

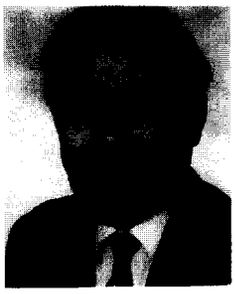
This research was partially supported by the NSF/ Purdue University Engineering Research Center on Intelligent Manufacturing Systems and partially supported by the contract of NSC84-0415-E-009-001, National Science Council, Taiwan, ROC.

REFERENCES

1 Requicha, A A G 'Representations for rigid solids: theory, methods, and systems' *ACM Comput. Surveys* Vol 12 No 4 (1980) pp 437–464
2 Joshi, S and Chang, T C 'Graph-based heuristics for recognition of machined features from a 3D solid model' *Comput.-Aided Des.* Vol 20 No 2 (1988) pp 58–66
3 Kyprianou, L K 'Shape classification in computer aided design' *PhD Thesis* University of Cambridge, UK (1980)
4 Case, K and Gao, J. 'Feature technology: an overview' *Int. J. Comput. Integrated Manfng* Vol 6 No 1/2 (1993) pp 2–12
5 Shah J J 'Assessment of features technology' *Comput.-Aided Des.* Vol 23 No 5 (1991) pp 331–343
6 Wu, M C and Liu, C R 'Flexible process planning for finish machining based on process requirement modeling' *Int. J. Comput. Integrated Manfng* Vol 4 No 2 (1991) pp 121–132
7 Pratt, M J and Wilson, P R 'Requirements for support of form features in a solid modelling systems' *R-85-ASPP-01* CAM-I Inc, Arlington, Texas, USA (1985)
8 Pratt, M J 'Applications of feature recognition in the product life cycle' *Int. J. Comput. Integrated Manfng* Vol 6 No 1/2 (1993) pp 13–19
9 Choi, B K, Barash, M M and Anderson, D c'Automatic recognition of machined surfaces from a 3D solid model' *Comput.-Aided Des.* Vol 16 No 2 (1984) pp 81–86
10 Henderson, M R 'Extraction of feature information from three dimensional CAD data' *PhD thesis* Purdue University, USA (1984)
11 Liu, C R and Srinivasan, R 'Generative process planning—a syntactic pattern recognition approach' *J. Comput. Mech. Engng* Vol 2 No 5 (1984) pp 63–66
12 Herbert, P J, Hinde, C J, Bray, A D, Launders, V A, Round, D and Temple, D M 'Feature recognition within a truth maintained processing system' *Int. J. Comput. Integrated Manfng* Vol 3 No 2 (1990) pp 121–132
13 Donaldson, I A and Corney, J R 'Rules-based feature recognition for 2.5D machined components' *Int. J. Comput. Integrated Manfng* Vol 6 No 1/2 (1993) pp 51–64
14 Kung, H 'An investigation into development of process plans from solid geometric modeling representation' *PhD Thesis* Oklahoma State University, OK, USA (1984)
15 Bartholomew, O N and Liu, H C 'Feature reasoning for automatic robotic assembly and machining in polyhedral representation' *Int. J. Prod. Res.* Vol 28 No 3 (1990) pp 517–540
16 Ansaldi, S, De Floriani, L and Falcidieno, B 'Geometric modelling of solid objects by using a face adjacency graphy representation' *ACM Comput. Graph.* Vol 16 No 3 (1985) pp 131–139
17 Srinivasan, R 'A generalized analytical methodology for generative process planning' *PhD Thesis* Purdue University, USA (1986)
18 Corney, J and Clark, D E 'Method for defining holes and pockets that connect multiple fces in 2 1/2D objects' *Comput.-Aided Des.* Vol 23 No 10 (1991) pp 658–669
19 Chuang, S H and Henderson, M R 'Three-dimensional shape pattern recognition using vertex classification and vertex-edge graphs' *Comput,-Aided Des.* Vol 22 No 6 (1990) pp 377–387
20 Falcidieno, B and Biannini, F 'Automatic recongition and representation of shape-based features in a geometric modeling system' *Comput. Vision, Graph. & Image Process.* Vol 48 (1989) pp 93–123
21 Fu, Z, de Pennington, A and Saia, A 'A graph grammer approach to feature representation and transformation' *Int. J. Comput. Integrated Manfng* Vol 6 No 1/2 (1993) pp 137–151
22 Laakko, T and Mäntylä, M 'Feature modelling by incremental feature recognition' *Comput.-Aided Des.* Vol 25 No 8 (1993) pp 479–492
23 Marefat, M and Kashyap, R L 'Geometric reasoning for recognition of three-dimensional object features' *IEEE Trans. Pattern Analysis & Machine Intell.* Vol 12 No 10 (1990) pp 949–965
24 Garey, M R and Johnson, D S *Computers and Intractability—A Guide to the Theory of NP-Completeness* W. H. Freeman (1979)
25 Srinivasan, R, Liu, C R and Fu, K S 'Extraction of manufacturing details from geometric models' *Int. J. Comput. & Indust. Engng* Vol 9 No 2 (1985) pp 125–133
26 Chan, A K W and Case, K 'Process planning by recognizing and learning machining features' *Int. J. Comput. Integrated Manfng* Vol 7 No 2 (1994) pp 125–133
27 Wang, M T 'A geometric reasoning methodology for manufacturing feature extraction from a 3-D CAD model' *PhD Thesis* Purdue University (1990)
28 Kanumury, K and Chang, T C 'Process planning in automated manufacturing environment' *J. Manfng Syst.* Vol 10 No 1 (1991) pp 67–78
29 Gavankar, P and Henderson, M R 'Graph-based extraction of protrusions and depressions from boundary representations' *Comput.-Aided Des.* Vol 22 No 7 (1989) pp 442–450
30 Chamberlain, M A, Joneja, A and Chang, T C 'Protrusion-features handling in design and manufacturing planning' *Comput.-Aided Des.* Vol 25 No 1 (1993) pp 19–28
31 Prabhakar, S and Henderson, M R 'Automatic form-feature recognition using neural-network-based techniques on boundary representations of solid models' *Comput.-Aided Des.* Vol 24 No 7 (1992) pp 381–393
32 Corney, J and Clark, D E R 'Face-based feature recognition: generalizing special cases' *Int. J. Comput. Integrated Manfng* Vol 6 No 1/2 (1993) pp 39–50
33 Mill, F G, Salmon, J C and Pedley, A G 'Representation problems in feature-based approaches to design and process planning' *Int. J. Comput. Integrated Manfng* Vol 6 No 1/2 (1993) pp 27–33
34 Chen, C L P and LeClair, S R 'Integration of design and manufacturing: solving setup generation and feature sequencing using an unsupervised-learning approach' *Comput.-Aided Des.* Vol 26 No 1 (1994) pp 59–75
35 V'ancza, J and M'arkus, A 'Features and the principle of locality in process planning' *Int. J. Comput. Integrated Manfng* Vol 6 No 1/2 (1993) pp 126–136
36 Young, R I M and Bell, R 'Design by features: advantages and limitations in machining process planning integration' *Int. J. Comput. Integrated Manfng* Vol 6 No 1/2 (1993) pp 105–112
37 Grayer, A R 'The automatic production of machined components starting from a stored geometric description' *in* McPherson, D (Ed.) *Advances in Computer Aided Manufacture* North Holland (1977) pp 137–150
38 Armstrong, G T, Carey, G C and de Pennington, A 'Numerical code generation from a geometric modelling system' *in* Pickett, M S and Boyse, J W (Eds.) *Solid Modelling by Computers: from Theory to Applications* Plenum Press, USA (1984) pp 139–154



Muh-Cherng Wu is currently a Professor in the Department of Industrial Engineering and Management, National Chiao Tung University, Taiwan, ROC. He received the BS degree in electronical engineering from National Chaio Tung University in 1977, the MBA degree from National Chen Chi University, Taiwan, ROC in 1979, the MS and PhD degrees in industrial engineering from Purdue University in 1985 and 1988, respectively. His current research interests encompass CAD/CAM, computer aided process planning and production planning.



C Richard Liu is a Fellow of the American Society of Mechanical Engineers, and has served as reviewer for numerous R&D programs and research journals. His early research has resulted in a major improvement of a product now that has an annual sale of $US 1 billion. He has prioneered the technologies of real time error compensation of machine tools and single-step superfinish hard machining. he has also carried out extensive research on CAD/CAM/CIM. With about 20 PhD students, he has published over 1120 research papers and edited four books. He has received, among others, the ASME Blackall Award for his contribution to the basic understanding of the integrity of machined surfaces: the IR100 Award for his joint effort in pioneering real time error compensation of machine tools; and certificates of appreciation from ASME and the State of Indiana, USA. Dr Liu is now Professor of Manufacturing Systems, School of Industrial Engineering, Purdue University, USA.