Feature recognition for NC part programming

R. Tuttle a, *, G. Little a,1, J. Corney a,2, D.E.R. Clark b,3

a Department of Mechanical and Chemical Engineering, Heriot-Watt University, Riccarton Campus, Edinburgh, Scotland EH14 4AS, UK
b Department of Mathematics, Heriot-Watt University, Riccarton Campus, Edinburgh, Scotland EH14 4AS, UK

Accepted 3 October 1997

Abstract

This paper reports on some current research concerning the automated recognition of manufacturing features on solid models to support the generation of NC machine code for process planning and manufacturing. Generic feature recognition algorithms previously developed by the authors for polyhedral objects have been considerably extended to enable the interrogation of more complex geometry and, in particular, the recognition of both ‘closed’ and ‘open’ manufacturing features. Once identified, these feature profiles are extruded to form 3D volumes, the so called Δ-Volumes, which are then passed to the solids machining package SolidCAM. A Graphical User Interface GUI has been designed which allows interpretation of the manufacturing features from the geometry of a component and keeps track of volumes created. The algorithm has also been integrated into SolidCAM and in such mode provides seamless operation within a commercial CAM system. © 1998 Elsevier Science B.V. All rights reserved.

Keywords: CAD/CAM; Feature recognition; Cycles; Volumes; Manufacturing

1. Introduction

Automated recognition of features on a boundary representation of a solid model has been studied for nearly two decades. Much of this evolved from the work of Kyprianou [1] on shape classification in Computer Aided Design (CAD). Although many algorithms have been developed from the original observations of Kyprianou, very few have actually been implemented outside of academic research laboratories. The work of Corney and Clark [2] was based on the identification of cycles of planar faces which were ‘vertical’ with respect to the machining direction, therefore their algorithm was suitable only for components on which the outward normals of all such planar faces were either perpendicular or parallel to the machining direction. The algorithm further assumed that the model was ‘single-sided’—i.e. could be machined entirely from one single direction. Furthermore, the original search method was also seen to be inefficient since during geometric interrogations on complex components a large amount of repetition was required. In the present paper a new approach has been developed in which the interrogations are performed prior to invoking a search algorithm.

* Corresponding author. Tel.: +44-131-449-5111 ext. 4343; fax: +44-131-451-3129; e-mail: b.tuttle@hw.ac.uk
1 Tel.: +44-131-449-5111 ext. 4343; e-mail: g.little@hw.ac.uk
2 Tel.: +44-131-449-5111 ext. 4381; e-mail: j.r.corney@bonaly.hw.ac.uk
3 Tel.: +44-131-449-5111 ext. 3234; e-mail: derc@ma.hw.ac.uk
4 SolidCAM is the copyright of CADCentre, Cambridge, UK

0166-3615/98/$19.00 © 1998 Elsevier Science B.V. All rights reserved.
PII S0166-3615(97)00089-4
The search algorithm itself has also been extended to allow the recognition of depression and protrusion features on components containing cylindrical faces and to allow the detection of open features, such as slots. The methodology has further been adapted to facilitate feature recognition on multi-sided components, creating machinable feature volumes which are accessible from different approach directions. The aim is to recognise manufacturing features on the solid model of a component and so, with knowledge of the original stock size, automate the creation of those volumes which need to be cleared from the stock body in order to manufacture the component. If original stock size is unknown the algorithm will allow the creation of the necessary volumes from a solid model assuming a complete solid within its outer profile. Furthermore, the refined algorithm collects geometrical information on interacting features (e.g. nested depressions). An important aspect of the algorithm is that the default depression volumes created then specifically preclude the formation of any intersecting volumes, these being unnecessary for manufacture.

Two recurring problems are encountered in the practical application of feature technology. Firstly, that of so-called feature interaction. This feature interaction phenomenon was noted by Trika [3] who pointed out that features can interact in an infinite number of ways. The second problem to arise is that of multiple interpretations which was discussed by Tseng and Joshi [4] who found that, for machined parts with interacting features, multiple valid sets of feature interpretation exist. The problem of multiple interpretations is also highlighted in the work of Sakurai and Gossard [5] who have concentrated on identifying all possible feature decompositions of a component generated from a small number of very simple (almost atomic) features (e.g. localised faces and concave edges). Another approach, used to good effect by Sakurai and Chin [6], entails decomposing the solid model of an object into a number of blocks to represent the component. This can lead to a large number of blocks being necessary in order to approximate the solid model successfully. From this it would appear that the solution is not one of generating the variety of possible interpretations, but rather of knowing which ones to use.

A further limiting factor in many feature recognition techniques is that of pattern matching for the identification of features. The work of Kulkarni and Pande [7] incorporates detection of, so called, feature signatures which can then be compared with known geometry. Similarly, the feature recognition module developed for the STEP system discussed by Kjellberg and Bohlin [8] uses comparative techniques from a pre-defined library. Allada and Anand [9] incorporate a rigorous rule-based system into their methodology for manufacturing feature recognition. Karadkar and Pande [10] also incorporate feature recognition into their system for CNC code generation, by the use of a rule based feature data structure. While such systems as those mentioned have many attractive attributes, they all rely upon either some form of feature library or data comparative techniques for feature labelling. This restricts their ability to label features which do not fit known shape or data. In contrast, the system described here requires no feature library as the geometry of a part is analysed: feature recognition by this method is completed by processing purely topological information without the need for pattern matching.

The current research and development being carried out is for a recognition system which avoids many of the previously mentioned classic difficulties by redefining the problem. The feature recognition for manufacturing has been interpreted by the authors in the context of an interactive feature recognition system. The need to analyse manufacturing features relative to original stock, as opposed to final component shape, has also been addressed. The previous research of Vandenbrande and Riequicha [11] also analysed stock, as it was felt to be a primary requirement for manufacturing needs. A description of the modus operandi of the recognition algorithm is given. The user-interface and its link to the Computer Aided Manufacturing (CAM) package SolidCAM is also described.

2. Algorithm description

The graph-theoretical approach of Corney and Clark [2] considered feature recognition from the specific viewpoint of NC machining. In doing this, the features of a model became associated with the machining
direction, or aspect direction, $d$. Their approach was applicable to models containing only planar faces and involved the omission of all non-vertical edges and faces from the face-edge graph leaving, what was termed, an aspect face-edge graph. This graph was then searched for closed cycles whilst checking that the orientation of the traversals of the faces was consistent.

The condition was imposed that any traversal of a face should be in the direction $n \times d$ (where $n$ is the outward normal of the face). This was implemented by creating a vector, $v$, from the start position of the edge connecting the current face to the previous face to the start position of the edge connecting the current face to the next face in the cycle (Fig. 1). If $v \cdot (n \times d)$ is positive then this traversal is valid. As a result of this rule, depression cycles are found with a clockwise orientation and protrusion cycles with an anti-clockwise orientation with respect to $d$.

3. Valid path criteria

Using the Corney/Clark algorithm it was noted that, when stepping off a more complex face, perhaps half of the possible routes considered lead to invalid cycles. This is the source of one of the inefficiencies of their algorithm. The algorithm presented here circumvents this problem. In the solid modeller used, a face is bounded by a loop of coedges which are directed instances of the edges adjacent to that face (Spatial Technology [12]). Viewed from the exterior of the model, the external boundary of a face consists of a list of coedges which form an anti-clockwise loop and any internal boundary of the face takes the form of a clockwise loop. A path search criteria has been adopted whereby the coedges of a face are divided into those which may only be used to step onto a face and those which may only be used to step off it. These coedges have been classified as on-edges and off-edges respectively. An on-edge of a face is defined as one whose coedge on that face is in the direction $d$, and an off-edge is one whose coedge is in the direction $-d$ (Fig. 2). A path may be created from an on-edge to an off-edge. It should be noted that, so called, void paths can also be generated out with the boundaries of the
faces. In Fig. 2 there is a void path from e4 → e3 thus identifying an open feature. Any other path, such as one from an on-edge to an on-edge, is invalid.

The orientation of a path across a cylindrical face is more complex. On a plane face, a path is said to be correctly oriented if it is in the direction \( n \times d \), where \( n \) is the outward normal of the face. But around a cylindrical face the direction of the normal varies so this condition is inapplicable. Instead, here the rule is that if the body is internal to the cylinder then a path should travel anti-clockwise when viewed from the approach direction, and, if the body is external, then the path should travel clockwise. In order to determine whether a path is correctly oriented the angles and sense of rotation need to be computed for all circular coedges which occur in the loop between the on and off-edges. An example of a cylindrical face is shown in Fig. 3.

Here the external loop of the face appears anti-clockwise from outside the cylinder and hence the body is internal to the cylinder. Therefore paths which traverse this face anti-clockwise are correctly oriented. For example, a path from e1 to e7 is correctly oriented since this would be anti-clockwise with respect to \( d \). However, between coedges e5 and e7 there is just one clockwise coedge, e6, so a path from e5 to e7 would be correctly oriented only as a void. Thus, the algorithm for generating the path across a face has to take into account not only the geometry of the underlying surface but also the number of on-edges and off-edges. There are several cases which arise and these must be dealt with in different ways as listed below.

(1) A periodic cylindrical face. This is most easily detected by the fact that the face is bounded by two loops. Since it is possible to travel right around the face, each off-edge is accessible from each on-edge and so paths are generated for all the possible combinations of these edges. A closed path is also created which circumnavigates the face between two null edges. This will form a cycle of its own.

(2) A face containing either one on-edge or one off-edge. If a face contains one on-edge, then all of the off-edges must be correctly oriented relative to it, so a path is created from the on-edge to each of the off-edges. Similarly, if there is one off-edge, then a path is created from each on-edge to the single off-edge.

(3) A planar face containing more than one on-edge as well as more than one off-edge. A path is created for each on/off pair for which the off-edge is situated in the direction \( n \times d \) relative to the on-edge.

(4) A cylindrical face containing more than one on-edge as well as more than one off-edge. If the body is internal (respectively external) to the cylinder then a path is created for each on/off pair for which the off-edge is anticlockwise (respectively clockwise) relative to the on-edge.

4. Cycle searching

The result of the path generation phase of the algorithm is a directed graph. An example is shown in Fig. 4 which is obtained for the component shown in Fig. 5. The arcs on the directed graph show valid paths between
the vertical edges of the model and each arc has the label of the face traversed by that path. Note that in the
graph there is a path across the underlying surface of face f0 between the on-edge e7 and the off-edge e4. The
reason for this is that that particular path is a void path used to identify open features.

At this stage the recognition algorithm proceeds by taking a list of all the paths which have been created and
searching amongst them for all closed cycles in the graph. The cycle data structure contains a list which is
empty at construction but which has paths added to it as the search proceeds. The search is initialised by the
construction of a cycle, referred to here as the present cycle, and the selection of an arbitrary path. The path is
added to the present cycle and a succeeding path is sought. A succeeding path is one which has an on-edge
identical to the off-edge of the last path in the present cycle. This simple process is continued until one of two
possible situations occur.

Firstly, more than one possible succeeding path may exist. In this case copies of the present cycle are made
and the possible succeeding paths are added, one to each cycle. The copies of the present cycle are placed in a
buffer to be completed later and the search is continued using the present cycle.

The second situation which may occur is that the goal of completing a closed cycle is achieved. The closing
edge, i.e. the off-edge of the last path, has already appeared as the on-edge of a previous path in the cycle. The
search may not have commenced with a path belonging to the cycle, in which case those paths prior to the first
appearance of the closing edge are removed from the cycle. If at this stage the buffer is not empty, then one of
the incomplete cycles is set to be the present cycle and the search continues on that cycle. If on the other hand

---

Fig. 4. Directed graph for component shown in Fig. 5.

Fig. 5. Component used for graph generation (Fig. 4).
the buffer is empty then the process is repeated starting with any path which has not yet been followed in the search. Such paths would be part of a disjoint subgraph.

Suppose for the example in Fig. 5 that the path e1→f2→e2 (the path from e1 across f2 to e2) is the one which initialised the search as described above. The search would proceed anti-clockwise with the present cycle being
\[ c_1 = \{ e1\rightarrow f2\rightarrow e2, e2\rightarrow f3\rightarrow e3 \} \]
at which point there are two possible succeeding paths, namely e3→f0→e0 and e3→f0→e4. A copy, c2, is now made of c1 and the two paths are added one to each cycle, so that
\[ c_1 = \{ e1\rightarrow f2\rightarrow e2, e2\rightarrow f3\rightarrow e3, e3\rightarrow f0\rightarrow e0 \} \]
\[ c_2 = \{ e1\rightarrow f2\rightarrow e2, e2\rightarrow f3\rightarrow e3, e3\rightarrow f0\rightarrow e4 \} \]
The present cycle, c1, is then completed with the succeeding path, e0→f1→e1 and the new present cycle is now c2,
\[ c_1 = \{ e1\rightarrow f2\rightarrow e2, e2\rightarrow f3\rightarrow e3, e3\rightarrow f0\rightarrow e0, e0\rightarrow f1\rightarrow e1 \} \]
\[ c_2 = \{ e1\rightarrow f2\rightarrow e2, e2\rightarrow f3\rightarrow e3, e3\rightarrow f0\rightarrow e4 \} \]

Two paths are added to c2 before reaching e6 at which point there are again two possible successors so a copy of c2 is made and the two paths added, one to each cycle.
\[ c_1 = \{ e1\rightarrow f2\rightarrow e2, e2\rightarrow f3\rightarrow e3, e3\rightarrow f0\rightarrow e0, e0\rightarrow f1\rightarrow e1 \} \]
\[ c_2 = \{ e1\rightarrow f2\rightarrow e2, e2\rightarrow f3\rightarrow e3, e3\rightarrow f0\rightarrow e4, e4\rightarrow f4\rightarrow e5, e5\rightarrow f5\rightarrow e6, e6\rightarrow f6\rightarrow e8 \} \]
\[ c_3 = \{ e1\rightarrow f2\rightarrow e2, e2\rightarrow f3\rightarrow e3, e3\rightarrow f0\rightarrow e4, e4\rightarrow f4\rightarrow e5, e5\rightarrow f5\rightarrow e6, e6\rightarrow f6\rightarrow e7 \} \]
The present cycle, c2, is completed by the addition of the paths e8→f7→e9 and e9→f4→e5 and all paths prior to the first occurrence of e5 as an on-edge are deleted, so
\[ c_1 = \{ e1\rightarrow f2\rightarrow e2, e2\rightarrow f3\rightarrow e3, e3\rightarrow f0\rightarrow e0, e0\rightarrow f1\rightarrow e1 \} \]
\[ c_2 = \{ e5\rightarrow f5\rightarrow e6, e6\rightarrow f6\rightarrow e8, e8\rightarrow f7\rightarrow e9, e9\rightarrow f4\rightarrow e5 \} \]
\[ c_3 = \{ e1\rightarrow f2\rightarrow e2, e2\rightarrow f3\rightarrow e3, e3\rightarrow f0\rightarrow e4, e4\rightarrow f4\rightarrow e5, e5\rightarrow f5\rightarrow e6, e6\rightarrow f6\rightarrow e7 \} \]
The addition of e7→f0→e0 and e0→f1→e1 closes the present cycle, c3, and now there are no unfinished cycles and no paths which have not been used so the search for closed cycles is complete.

Paths are further searched to identify any open features by searching for any existing cycles of valid paths connected by void paths. In this case there is only possible search, that of
\[ c_4 = \{ e4\rightarrow f4\rightarrow e5, e5\rightarrow f5\rightarrow e6, e6\rightarrow f6\rightarrow e7, e7\rightarrow f0\rightarrow e4 \} \]

(italics denote the void path)

The cycle search is now completed, having found
\[ c_1 = \{ e1\rightarrow f2\rightarrow e2, e2\rightarrow f3\rightarrow e3, e3\rightarrow f0\rightarrow e0, e0\rightarrow f1\rightarrow e1 \} \]
\[ c_2 = \{ e5\rightarrow f5\rightarrow e6, e6\rightarrow f6\rightarrow e8, e8\rightarrow f7\rightarrow e9, e9\rightarrow f4\rightarrow e5 \} \]
\[ c_3 = \{ e1\rightarrow f2\rightarrow e2, e2\rightarrow f3\rightarrow e3, e3\rightarrow f0\rightarrow e4, e4\rightarrow f4\rightarrow e5, e5\rightarrow f5\rightarrow e6, e6\rightarrow f6\rightarrow e7, e7\rightarrow f0\rightarrow e0, e0\rightarrow f1\rightarrow e1 \} \]
\[ c_4 = \{ e4\rightarrow f4\rightarrow e5, e5\rightarrow f5\rightarrow e6, e6\rightarrow f6\rightarrow e7, e7\rightarrow f0\rightarrow e4 \} \]

5. Remark

At first glance it might appear that by generating the paths prior to the search, in order to find the possible next paths in a cycle, all of the paths created over the entire model have to be checked rather than merely those
which cross the subsequent face. It turns out, however, that this problem is heavily outweighed by the absence of any geometric interrogations during the search. In any case, it is easily overcome simply by adding a list of next paths as an attribute for each edge at the path generation stage.

Next, duplicate cycles and self-intersecting cycles must be eliminated. Duplicate cycles occur frequently when the initial path is not actually part of the cycle and the cycle is reached by distinct routes. Each pair of cycles is checked for equality. Two cycles are equal, by definition, if they contain the same paths in the same sequence.

At this stage a list of closed cycles has been obtained and it is now required to classify these as either depressions or protrusions. The flow-chart for this algorithm appears in Fig. 6.

The closed cycles created by the algorithm must either bound a depression feature or a protrusion feature on the component being interrogated. A cycle with the normal of the face traversed by its first path being directed out from the profile is a protrusion feature, whilst if the normal is directed in from the profile this indicates the cycle bounds a depression feature (i.e. a feature where material needs to be removed from the original stock in order to create the component). The cycles are thus classified as either depression or protrusion cycles. The protrusion features indicate the presence of material on a component, but individually they do not aid the next process, that of generating volumes for NC-machining. Therefore any protrusion cycles, whilst originally created from the path search and cycle finding sections of the algorithm, play no further role.

Fig. 6. Flow chart of cycle creation method.
6. Multi-sided component feature recognition

For feature recognition on multi-sided components a number of additional changes have been made to the algorithm. These include an ‘approachability’ check, though its basic feature recognition procedure remains unaltered. A multi-sided component can be defined as a component whose manufacture requires machining from two or more directions. In previous work by the authors on single-sided components (Corney and Clark [2]) only one aspect (or machining approach) direction was applicable. For multi-sided components, however, a number of aspect directions are required. The actual number of aspects will vary depending upon the profile of the component, but each aspect can be specified by a unit vector anti-parallel to a planar face on the component.

When analysing a multi-sided component, as each aspect direction is selected for feature recognition, all valid paths are originally found without any check on visibility. The visibility, or approachability, of each path is then determined by analysis of its start and end bounding points. If a human observer were to determine the visibility of a point on a component from a specific direction i.e. a machining approach direction then a primary indicator of visibility of the point would be that light can travel directly from the said point to the human eye without obstruction. In a similar manner to a ray of light, a solid modeller ‘ray’ is used to detect path visibility from any given aspect direction. The ray is directed to travel along a specific path (the machining path) in a given direction (opposite to the aspect direction) from a given point (start and end points of the path in question), and a record is kept of any entities it encounters during its journey. If a ray passes the outer profile of the body uninterrupted then it is labelled as approachable. Once all paths have been labelled as approachable or not the feature recognition algorithm continues to the cycle generation stage. Only paths which have been labelled approachable are used for cycle generation, as previously described, and machining volume creation, as described below.

7. Volume creation

Experience has shown that it is desirable at this stage for the algorithm to allow the user some influence in the creation of the volumes. The volumes that are required are dependent on the sequence in which the profiles are to be machined, and two engineers may choose two different ways to manufacture a component. Therefore a default set of volumes is created which the user is given the opportunity to modify. It is also advantageous on the grounds of efficiency that the volumes offered by the default solution do not intersect each other, as this would lead to the machining of ‘fresh air’. In order to ensure that the default volumes are disjoint, it is necessary not only to associate a maximum depth to which the profiles can be machined, but also a minimum depth which specifies the height at which the volume clearance should commence. Therefore, a so-called depth-range is associated with each depression cycle which is computed using the following geometric analysis but may be altered by the user picking positions on the model using the mouse.

The depth-range of a cycle is the intersection of the depth-ranges of all its constituent paths. The depth-range of a path is by definition, the range of distances below the profile-plane between which the path may traverse the face without leaving the face. An initial depth-range is obtained for a path by intersecting the depth-ranges of its on- and off-edges. If the maximum of the initial range is greater than the minimum then a check is made to ensure that there are no off-edges obstructing the path. Off-edges represent steps off the face and so a path may not travel within the depth-range of any off-edge situated between the on- and off-edges of the path and the ranges of all such off-edges are subtracted from the initial depth-range of the path. As an example, Fig. 7 shows the depth-range of a path from edge e1 to edge e2.

Initially the range of the path is set to the intersection of the ranges of the two edges, that is, between the lower and upper of the horizontal lines bounding the shaded area. However we see that off-edges e3 and e4 are situated so as to obstruct the path in part of this range. This strategy prevents the default creation of intersecting
volumes. As a result, volumes are created only for those cycles whose profiles would be detected using the technique Grayer [13] put to good effect of slicing the body at various depths. It should be noted that the method developed by Grayer could only be translated to useful manufacturing information when applied to components which could be fully machined from one a single approach direction (i.e. single sided). If the body were sliced at a depth internal to the range of a cycle then the profile of the cycle would be identified by the slicing operation. It is to be also noted that, in general, there will be cycles which have a zero depth-range which the user may find useful but would not be detected by Grayer’s slicing method. This highlights a further advantage of the present algorithm. The cycles with zero depth-range are offered as profiles from which the user has the option of creating volumes manually, if so desired.

8. The user interface

The FeatureFinder, as the system is known, has been designed to run in a Motif-based test harness. The interface has been developed in order to support all the necessary feature based operations undertaken by the underlying algorithm whilst showing the results graphically on the display screen. The structure and functionality of the user interface is of a similar nature to that discussed by Plamen et al. [14]. Solid models of a component and its associated original stock are loaded into the graphics window where they can be viewed in either wireframe or rendered modes. Fig. 8 shows this interface with a component and its stock loaded.
The system can operate without the inclusion of the original stock, however the ‘remainder’ δ-volume (i.e. stock minus component and created δ-volumes) cannot then be created. When operating the recognition system the orientation of the component relative to the machine tool is selected and the shape of the stock is pre-defined. Knowledge of the machining direction is necessary as this becomes the aspect direction, d, as used in the search algorithm. This allows the system to identify appropriate volumes of material for removal for a given set-up. ‘Appropriate’, in this instance, means accessible and unique (i.e. the volumes do not intersect). The non-intersecting volumes are created by sweeping 2D profiles generated by a search procedure described in section 8. The requirement for non-intersecting feature volumes means the system only offers one interpretation of the component’s decomposition.

By default this interpretation is one based on the relative heights of the faces and edges in the profile. Yet, because the feature volumes returned are solid models, and the system is interactive, the user is free to modify (or ignore) them as appropriate. Thus the system, like a spelling checker, can suggest one solution which the engineer is free to adopt, modify or disregard.

The user interface is operated in three basic stages. Firstly, aspect direction selection, secondly cycle search and, finally, volume creation. These options are accessed via a series of pull-down menus, each of which is now described.

9. Aspect selection

The approach direction, or aspect vector, must be specified before any other operation. On selecting ‘Aspect’ from the menu there is displayed a list of direction vectors from which the user selects. The object of this is to specify a direction or view, the aspect direction d, from which the model will be interpreted. This action is repeated, following cycle search and volume creation, for each necessary machining direction.

10. Cycle search

The find cycles option is selected after choosing the correct aspect direction. Although nothing new appears on the screen at this point, the feature finding algorithm will interrogate the geometry of the component and collect the topographical information needed to identify any depression cycles on the component. With the generic knowledge gained from the analysis of the component it is then possible to create the volumes for removal from the stock during manufacture of the component.

11. Volume creation

There are a variety of volume creation methods available to the user. These can be used either independently or in combination. Default volumes can be created and once this option has been selected a radio-toggle button widget is used to select individual cycles which then displays the appropriate volume on the screen. Should generation of a volume from any particular cycle cause intersecting volumes then it will not be created and only the profile will be displayed. Default volume creation acts only on those cycles which have an associated depth-range. Volumes can be created and/or modified manually. This is achieved by picking the minimum and
maximum heights required for a highlighted cycle. Once the user is satisfied with the result, the volume will be created. If required, single or multiple volumes can also be removed from the volumes list.

The volumes and remainder are saved as a solid model file for importation into the manufacturing package SolidCAM. The SolidCAM software modus operandi for generation of a manufacturing sequence and process plan commences with the creation of a δ-volume which is the product of removal of the component from a pre-defined stock, thereby creating a model of the material to be removed. Having created the single δ-volume, the user then manipulates this to create individual volumes to be removed in a production process plan. It has been shown that, even with relatively simple components, substantial time can be saved in manipulating the δ-volumes in preparation for machining if the geometry has previously been interrogated and volumes created (Tuttle et al. [15]). Fig. 9 shows the Feature Recognition GUI with the (automatically created) volumes, the
original stock, the remainder and the component. Fig. 10 shows SolidCAM with the volumes and remainder, created by feature recognition for a component, loaded and visible in the graphics window.

12. Integrated FeatureFinder

Currently the feature recognition system described has been integrated into the CAM package SolidCAM and is being tested by a number of end-users. When operating the integrated system a number of further enhancements have been incorporated: (1) loading of part and stock: this is unnecessary, the current active component and stock are used. (2) Aspect selection: not needed, as the active co-ordinate system (either global or local) is linked into the FeatureFinder code. (3) Loading volumes from FeatureFinder into SolidCAM: redundant, when the volumes are saved they are automatically passed through to the SolidCAM parts list with δ-volume attributes attached. (4) Machining direction attributes are also automatically attached to the volumes created by FeatureFinder. This is of critical importance when working with multi-sided components requiring more than one machining direction.

Further enhancements and changes may be made in the light of feedback and results of the current field trials. For example, it has already been decided that for integrated use the find cycles stage should be automated and the end-user be offered default volumes directly.

13. Current limitations

The theory and descriptions given here are only applicable to multi-sided components with 2.5 dimensional attributes from each separate tool approach direction. Work is ongoing to increase the range of the algorithm to handle non 2.5-dimensional features such as draft angles, fillets, chamfers and pockets with sloping faces.

Fig. 11. Feature recognition on a ‘multi-sided’ component.
When analysing components with a large number of interacting features the algorithm was susceptible to combinatorial explosion which limited its use. This phenomena was commented upon by Peters [16], and methods proposed there which could contain the problem. Collaboration with other feature recognition groups showed that this was a problem of a common nature (Han et al. [17]). In the present case this has been addressed by the elimination of any paths of zero depth-range from the paths-list during the path-list creation. This has significantly reduced combinatorial explosion to within computationally acceptable bounds for the vast majority of engineering components tested to date. Work is ongoing to further address this problem.

14. Conclusions

A generic feature recognition algorithm has been described which is capable of automatically interrogating the geometry of a model of a multi-sided component and generating $\delta$-volumes for manufacturing.

The examples given previously were, by nature, of simple geometry. This was necessary in order to introduce the algorithmic theory and method to the reader. The algorithm is capable of analysis of more complex topology, as shown in Fig. 11. This figure indicates the results of applying the algorithm to the, so called, mohne component shown in the centre of the diagram. Four approach directions (labelled A1–A4) were, in turn, selected by the user. Volumes 1 to 4 represent currently represent unidentified material in the $\delta$-volume and as such are left as remainder of rectangular stock. Several of the identified volumes should be given particular attention as they highlight strengths of the search algorithm: volume number 7 arises from a complex pocket, note that the volumes associated with the nested cylindrical holes in the base of the feature are identified as separate features with non-intersecting volumes created. Volumes numbered 5, 6 and 8 are examples of volumes arising from open features which the system has identified. Two of these, volumes 5 and 6, emanated from more than one face and demonstrates how the algorithm can find a wide range of ‘slot’ features. The lower part of the open pocket volume number 8 includes an undercut region. Because this is not accessible from any of the approach directions A1–A4 it forms volume number 4 in the remainder.

The program is being used by industrial collaborators in conjunction with commercial CAM software. Preliminary incorporation of the feature recognition algorithms into the commercial CAM package has been undertaken. Current field trials of this integrated version should ensure a final robust and useful feature recognition system.

Open features such as slots are incorporated into the feature recognition system. Small-scale shape features (e.g. fillets and chamfers) are not yet recognised and work is ongoing to incorporate such features, although it is interesting to note that they are currently easily identifiable in the remainder.

Acknowledgements

The authors wish to thank the collaborating companies, MacTaggart Scott (UK) and CADCentre (UK), for their assistance and part funding of the project and Dr. N. Sormaz of Theorem Solutions UK, for his contribution as a previous member of our research team. We would also like to thank the Engineering and Physical Sciences Research Council (UK) for continued financial support of the research under grant no. GK/K48020.
References


Gordon Little. Received B.Eng. in Mechanical Engineering from Heriot-Watt University in 1993. Research Associate at Heriot-Watt University since 1996. Research interests include Computational Geometry, Solid Modelling Applications, Object Oriented Techniques in Computer Programming and Information Technology.
Jonathan R. Corney. Received BSc in Mechanical Engineering from Heriot-Watt University in 1983 and PhD from the same University in 1993. Industrial experience includes work in the fields of Mechanical Engineering and Design Engineering. Lecturer in Mechanical Engineering at Heriot-Watt University since 1988. Research interests include Applications of Solid Modelling to Design and Manufacture, Artificial Intelligence, Rapid Prototyping and Information Technology.

Douglas E.R. Clark. Received BSc in Mathematics from London University in 1967 and PhD from the University of Durham in 1970. A 1970 recipient of a Royal Society European Fellowship to continue research in the Geometry of Manifolds at the Mathematics Institute in the University of Amsterdam, Netherlands. Lecturer in Mathematics at Heriot-Watt University since 1971. Research interests include Differential Geometry, Low Dimensional Topology and Computational Geometry.