19

A Feature Recognition Algorithm for NC-Machining

N Sormaz, D E R Clark**, J R Corney*, J R Tuttle**
**Department of Mechanical & Chemical Engineering,*
***Department of Mathematics*
Heriot-Watt University, Edinburgh, SCOTLAND, EH14 4AS
Tel: +44 (131) 449 5111 Fax: +44 (131) 451 3129
email: b.tuttle@hw.ac.uk; mecjrc@hw.ac.uk: derc@ma.hw.ac.uk

Abstract

Previous work of the authors (Corney, 1993) has been extended to allow the recognition of closed depression and protrusion features on components containing cylindrical faces. The aim is to automate the creation of volumes which need to be cleared from a stock body in order to manufacture a component. The original algorithm was inefficient because geometric interrogations on complex components were repeated and a large amount of backtracking was required. In the new approach the interrogations are performed prior to the search algorithm. The recognition algorithm will eventually be extended to detect open features and to handle multi-sided components. The theory in this paper pertains only to single-sided, 2½-dimensional components.

Keywords

CAD/CAM, Feature Recognition, Vertex-Edge Graph, Face-Edge Graph, Aspect Direction, On/Off Edge, Cycles, Volumes, Manufacturing

1 INTRODUCTION

The theory of Corney and Clark (1993) is based on the identification of cycles of faces which are vertical with respect to the machining direction. Because of this, the algorithm is suitable only for components for which the outward normals of all planar faces are either perpendicular or parallel to the machining direction, and the axis directions of cylindrical faces are oriented in the machining direction. Their theory further assumes that the model is single-sided (i.e. may be machined completely from one direction). In the implementation of the algorithms these conditions may be relaxed somewhat since any faces in the model which do not meet the above criteria are simply ignored. Thus, for example, a pocket composed of vertical faces will still be detected even if it emanates from a free-form surface.

Much of the work in feature recognition has been based on the ideas of Kyprianou (1980). His theory is that a loop made up of convex edges indicates the presence of a depression feature, and a loop of concave edges indicates the presence of a protrusion.

M. J. Pratt et al. (eds.), *Product Modeling for Computer Integrated Design and Manufacture*

The work of Vandenbrande and Requicha (1993) tackles feature recognition specifically from the machining point of view. The stock volume from which the component is to be manufactured, and the delta-volume (stock minus volume to be removed) play a role in the generation of feature "hints" as well as in the validation of the feature volumes. The hint of the profile of a closed pocket is provided by either a cylindrical face, or by a planar face which is possibly accessible as a profile (it has at least one convex edge) and there is some stock to be removed along that face (there is an adjacent face in the delta-volume). For more complex components the result is a large number of hints which the algorithm seeks to discard at the earliest stage possible. A profile axis is determined as the search progresses and faces contributing to the profile must be vertical with respect to that direction. The condition is also imposed that two adjacent faces must not meet at a concave angle as, in that case, a circular cutter cannot access it from the profile axis direction.

2 GRAPHICAL REPRESENTATION

Several different graphs representing topology of solid models have been proposed. Chuang and Henderson (1989) utilised a *vertex-edge graph* in which the nodes of the graph represent the vertices of the component and the arcs represent the edges connecting the vertices. Vertices were classified according to the edge vexity and underlying geometry of the adjacent edges and patterns in the graph sought representing specific features.

A graphical face-based representation, known as a *face-edge graph,* was developed by Ansaldi et al (1985). Here the nodes of the graph represent faces in the model, and the arcs represent common edges of the faces. DeFloriani (1989) observed that a blind hole (or depression) may be detected by the presence of a cut-vertex in the face-edge graph, whose deletion would, by definition, separate the graph into two subgraphs. Similarly, a through hole on the component may be detected by the presence of a separation-pair, whose deletion would split the graph into two disjoint subgraphs. However, despite the novel use of graph theory, the only features that may be identified using the method are those emanating from the inner loops of faces.

Joshi and Chang (1988) considered an *attributed adjacency graph* (AAG) by assigning attributes to the arcs denoting whether the underlying edges are convex or concave. A hierarchy of machinable features (not protrusions) is defined for which a subgraph, containing only concave edges, displays particular properties. To search the AAG for feature subgraphs is computationally exhaustive, so a heuristic is employed based on the observation that "a face that is adjacent to all its neighbouring faces with a convex angle does not form part of a feature". While feature interactions are considered, in certain cases a change in the solid model may result in an existing feature subgraph losing its adjacency relationships.

The approach of Corney and Clark (1993) was to consider feature recognition from the specific viewpoint of NC machining. Here the features of a model became associated with the machining direction, or *aspect direction,* **d**. Their theory 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** x **d** (where **n** is the outward normal of the face). This is implemented by creating a vector, **v**, from the start position of the edge connecting the face to the previous face to the start position of the edge connecting the face to the next face in the cycle (See Figure 1). If **v**.(**n** x **d**) is positive then this traversal is valid. As a result of this principle, depression cycles are found with a clockwise orientation and protrusion cycles anti-clockwise with respect to **d**

Using the Corney-Clark algorithm it is found that when stepping off a complex face approximately half of the possible routes considered lead to invalid cycles. This is one source of inefficiency in the algorithm. The interrogations as to the relative positions of the edges on a face become more time consuming on cylindrical faces and the repetition of these within the algorithm becomes undesirable. 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. 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 theory is proposed here, whereby the coedges of a face are divided into those which may only be used to step *onto* the face and those which may only be used to step *off* it (see section 4 and figure 2). This results in both an ordered reduction in repetition and less interrogation of invalid cycles.

3 ASPECT CLASSIFICATION

The algorithm does not presume that the negative z-direction is the approach direction as seems to be the case in many machining packages. Instead, it is intended that the user should be able to choose a direction and then the algorithm should return volumes which are accessible from that direction. However, the recognition algorithm described here is only pertinent to single-sided components and so the software user is allowed to select any of the directions offered but is only permitted to proceed if the choice is a single-sided, 2½-dimensional aspect direction.

A component is said to be *2½-dimensional with respect to the direction, **d***, if the normals, **n**, of all planar faces satisfy **d**.**n** = -1, 0 or +1 and the axis directions, **a**, of all cylindrical faces are such **a**.**n** = +1 or -1.

A component is said to be *single-sided from the direction, **d***, if there is exactly one planar face whose outward normal is equal to **d** and the outward normal, **n**, of every other face is such that **d**.**n** >=0. For cylindrical faces, the normal at any position on the face should satisfy this condition.

All faces of the component are classified into one of the following four categories:

(1) p-face (parallel face). A planar face whose outward normal is equal to **d**.
(2) ap-face (anti-parallel face). A planar face whose outward normal is equal to **-d**.
(3) v-face (vertical face). A planar face whose outward normal, **n**, is such that **n.d** = 0, or a cylindrical face whose axis direction is equal to either **d** or **-d**.
(4) o-face (other face). A face not in any of the above categories.

Any single-sided, 2½-dimensional component has one p-face, any number of v-faces and ap-faces and no o-faces. Future research will incorporate another category, b-faces (or blended faces), these should emerge from the o-faces, representing faces which are the result of a blending operation in which an edge is chamfered or filleted. These faces will play no role in the search for feature profiles but will, of necessity, be taken into account in the volume creation stage of the algorithm. For the purposes of this paper it is assumed that all faces fall into one of the first three categories.

4 PATH GENERATION

Rather than performing the geometric interrogation within the actual search routine, the current algorithm achieves increased efficiency by collecting all possible paths beforehand. A further improvement is made by omitting traversals such as that between edges e1 and e4 shown in Figure 2, on the grounds that there is no correctly oriented path *from* e4 across any face other than the present one. The directions of the vertical coedges of the face are indicated in Figure 2, since it is these directions which determine whether the underlying edge may be used to step onto or step off a face. 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**. A path may only be created from an on-edge to an off-edge. Any other path, such as one from an on-edge to an on-edge, cannot occur in a cycle.

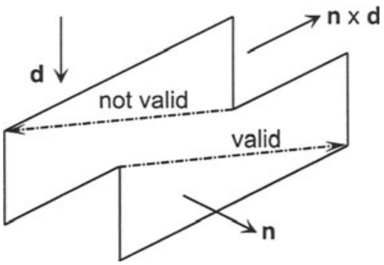

Figure 1 Corney and Clark Theory.

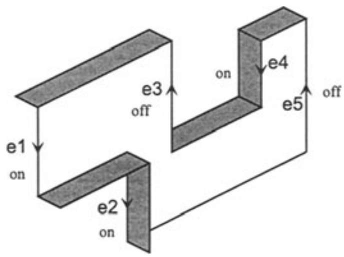

Figure 2 Present Path Selection.

On a plane face a path is said to be *correctly oriented* if it is in the direction **n** x **d**, **n** being the outward normal of the face. The problem of the orientation of a path across a *cylindrical* face is more complex as the normal direction varies around the face, so this condition cannot be applied. Instead, the rule is that if the body is internal (resp. external) to the cylinder then a path should travel anti-clockwise (resp. clockwise) when viewed from the approach direction.

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-edges and off-edges.

An example of a cylindrical face is shown in Figure 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. The path from el to e7 is correctly oriented since the angle of the anticlockwise coedge, e2, situated between them in the loop, is greater than the sum of the angles of the clockwise coedges e4 and e6. However, between coedges e5 and e7 there is just one clockwise coedge, e6, so a path from e5 to e7 would not be correctly oriented.

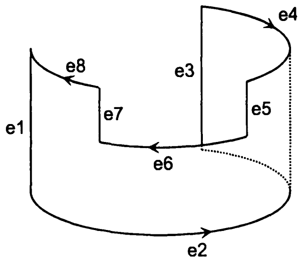

Figure 3 Cylindrical Face

The algorithm for generating the paths 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 are dealt with in different ways:

A periodic cylindrical face: This is most easily detected by the fact that the face is bounded by two loops (since the model is single-sided and the face is vertical there can be no internal 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. A closed path is also created which circumnavigates the face between two null edges. This will form a cycle of its own.

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.

A planar face containing more than one on-edge and 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** x **d** relative to the on-edge.

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

5 CYCLE FINDING AND CLASSIFICATION

The result of the path generation phase is a directed graph. An example is shown in Figure 4. The arcs show valid paths between the vertical edges of the model, each arc having the label of the face traversed by that path. Note that in the graph there is no path across face f0 between the on-edge e7 and the off-edge e4 since that path is not correctly oriented.

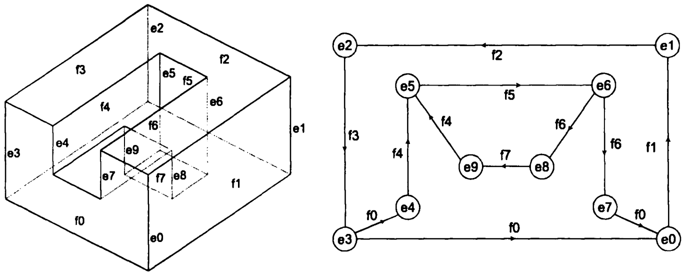

Figure 4 Component and its Directed Graph

The recognition algorithm proceeds by taking a list of all the paths that have been created and searching 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 process is continued until one of two possible situations occur:

[i] More than one possible succeeding path exists: 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.

[ii] The goal of completing a closed cycle is achieved: The closing edge, or 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 the buffer is not empty, then one of the unfinished cycles is set to be the present cycle and the search continues on that cycle. If 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 Figure 4 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 c1 = {e1-f2-e2, e2-f3-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:
c1 = {e1-f2-e2, e2-f3-e3, e3-f0-e0}
c2 = {e1-f2-e2, e2-f3-e3, e3-f0-e4}
The present cycle, c1, is then completed with the succeeding path, e0-f1-e1 and the new present cycle is now c2,
c1 = {e1-f2-e2, e2-f3-e3, e3-f0-e0, e0-f1-e1}
c2 = {e1-f2-e2, e2-f3-e3, e3-f0-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.
c1 = {e1-f2-e2, e2-f3-e3, e3-f0-e0, e0-f1-e1}
c2 = {e1-f2-e2, e2-f3-e3, e3-f0-e4, e4-f4-e5, e5-f5-e6, e6-f6-e8}
c3 = {e1-f2-e2, e2-f3-e3, e3-f0-e4, e4-f4-e5, e5-f5-e6, e6-f6-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
c1 = {e1-f2-e2, e2-f3-e3, e3-f0-e0, e0-f1-e1}
c2 = {e5-f5-e6, e6-f6-e8, e8-f7-e9, e9-f4-e5}
c3 = {e1-f2-e2, e2-f3-e3, e3-f0-e4, e4-f4-e5, e5-f5-e6, e6-f6-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 is complete having found
c1 = {e1-f2-e2, e2-f3-e3, e3-f0-e0, e0-f1-e1}
c2 = {e5-f5-e6, e6-f6-e8, e8-f7-e9, e9-f4-e5}
c3 = {e1-f2-e2, e2-f3-e3, e3-f0-e4, e4-f4-e5, e5-f5-e6, e6-f6-e7, e7-f0-e0, e0-f1-e1}
A flow chart for the algorithm appears in figure 5.

A disadvantage of this algorithm is 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 those which cross the subsequent face. This problem is heavily outweighed however by the absence of any geometric interrogations during the search, and in any case, may be overcome by simply adding a list of next paths as an attribute for each edge at the path generation stage.

At this stage a list of closed cycles has been collected and it is now required to classify these as either depressions or protrusions. First however duplicate cycles and self-intersecting cycles must be eliminated. Duplicate cycles are found quite frequently when the initial path is not actually part of the cycle and the cycle is reached by different 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. Each path contains the attribute of an edge, termed a *p-edge*, which is situated in the *profile-plane*. The *profile-plane* is located at the top of the stock, if a stock is given, or the top of the component otherwise, and has the machining direction, **d**, as its normal. The p-edge connects the projections of the on-edge and off-edge onto the profile-plane (being vertical, they project onto points) and takes into account the underlying geometry of the face which is traversed by the path.

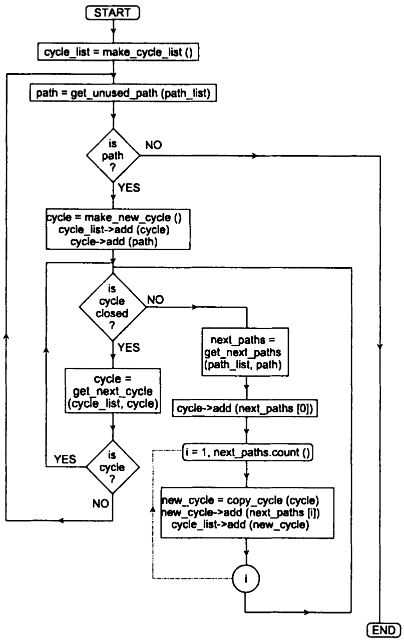

Figure 5 Flow Chart of Cycle Finding

A cycle is said to be *self-intersecting* if any two of its p-edges intersect at positions other than at their vertices. On a single-sided component, a self-intersection may only occur between the p-edges of paths which traverse the same face. For a cycle containing *n* paths, there are $n^2/2$ pairs of paths that must be checked, however this is not computationally expensive if the intersections of the p-edges are computed only for those paths which traverse the same face. The p-edges of all of the paths of a cycle form a closed profile in the profile-plane and are used to determine whether the cycle represents a protrusion or a depression. A semi-infinite line is created from an arbitrary position on one of the p-edges of a cycle in the direction of the outward normal of the relevant face at that point. This line is directed *into* the profile if the cycle is a depression cycle and *away from* the profile if it is a protrusion cycle.

Therefore, if the number of intersections between this line and the other p-edges in the cycle is odd then the cycle bounds a depression feature and if it is even then the cycle bounds a protrusion feature. It is worth emphasising that protrusion features indicate the presence of material on a component, but individually they do not aid the process of generating volumes for NC-machining and hence are discarded. For more complex components the number of cycles found may be very large so a heuristic is employed to decrease the number of protrusion cycles. A closed depression feature cannot traverse the outer profile of the component and so it is unnecessary to generate paths on the faces of the outer profile. These faces can be identified by the fact that they are adjacent to the only p-face of the component via the external loop of that face (faces adjacent to the p-face via an internal loop form a through hole). Any vertical faces which are adjacent to a concave internal loop of an ap-face may only be traversed by protrusion cycles and therefore need not be considered.

6 VOLUME CREATION

It is desirable at this stage that the algorithm should allow the user some influence in the creation of the volumes. The volumes required are dependent on the sequence in which the profiles are to be machined, and engineers may select different methods of manufacture. A default set of volumes is created which the user is given the opportunity to modify. The volumes offered by the default solution do not intersect each other as this would lead to the machining of empty space. In order to ensure that the default volumes do not intersect, 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 *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 interface.

The depth range of a cycle is the intersection of the depth ranges of all of the paths making up that cycle. The depth range of a path, by definition, is the range of distances below the profile-plane between which the path may traverse the face without leaving the face. An initial range is obtained for a path by intersecting its on-edges 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 range of the path. 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 Grayer's (1975) technique of slicing the body at various depths. 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. In general, there will be cycles which have a zero depth range which the user may also find useful but would not be detected by Grayer's slicing method. These are offered as profiles from which the user has the option of creating volumes manually.

7 CONCLUSIONS

An algorithm for the recognition of closed depression features has been described, This algorithm encompasses cylindrical faces and has been implemented using a commercially available solid modeller. The algorithm is of comparable computational complexity to that presented by Corney and Clark (1993). The resulting software is being used by an industrial partner in conjunction with their CAM software. There are plans to incorporate the feature recognition algorithms in the CAM software in the near future.

It is intended that further work will be carried out on the volume creation described in the previous section. An extension of the technique is under development to detect open features. In certain instances, the algorithm is susceptible to combinatorial explosion. This phenomena was commented upon by Peters (1992) and methods proposed which could limit the problem. A number of heuristics are being investigated to restrict the search to features of manufacturing significance.

8 REFERENCES

Ansaldi S., De Floriani L. and Falcidieno B. (1985) Geometric modeling of solid objects by using a face adjacency graph representation. Computer Graphics **16** (3), 131-139, 1985.

Chuang S.H. and Henderson M.R. (1989) Three-dimensional shape pattern recognition using vertex classification and the vertex edge graph. Computer-Aided Design **22** (6), 377-387, 1989.

Corney J.R. and Clark D.E.R. (1993) Efficient face-based feature recognition. Proceedings Second Symposium on Solid Modeling and Applications 1993. 313-322, 1993.

De Floriani L. (1989) Feature extraction from boundary models of three dimensional objects. IEEE Transactions on Pattern Analysis and Machine Intelligence, **11** (8), 785-798, 1989.

Grayer A.R. (1975) A computer link between design and manufacture. PhD thesis, University of Cambridge, 1975.

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

Kyprianou L.K. (1980) Shape classification in computer aided design. PhD thesis, University of Cambridge.

Peters Thomas J. (1992) Combinatorial Analysis of Feature Recognition. Proc ASME 4th International Conference on Design Theory and Methodology, Scottsdale, AZ. September 1992. **Vol DE42**, pp.91-97.

Vandenbrande J.H. and Requicha A.A.G. (1993) Spatial reasoning for the automatic recognition of machinable features in solid models. IEEE Transactions on Pattern Analysis and Machine Intelligance, **15** (12), 1269-1285, 1993.

9 BIOGRAPHY

Nik Sormaz. Received B.Sc. in Mathematics from Newcastle University in 1991 and Ph.D. from the Hull University in 1994. Research Associate at Heriot-Watt University 1995-1996. Research interests include Computational Geometry, Solid Modelling Applications, Computer Programming and Information Technology.

Jonathan R. Corney. Received B.Sc. in Mechanical Engineering from Heriot-Watt University in 1983 and Ph.D. from the same University in 1993. Industrial experience includes work in the fields of Mechanical Engineering & 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 B.Sc. in Mathematics from London University in 1967 and Ph.D. 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.

J.R. (Bob) Tuttle. Received B.Eng. in Mechanical & Manufacturing Engineering from Sheffield City Polytechnic in 1991 and awarded the I.Mech.E. (Institution of Mechanical Engineers) Prize for outstanding final year Mechanical Engineering dissertation. A Chartered Engineer (C.Eng.) with fifteen years industrial experience in Manufacturing Engineering Technology. Research Associate at Heriot-Watt University since 1995. Research interests include CAD/CAM, Information Technology, Design for Manufacture, Computer Simulations and Computer Aided Process Planning.