Tool selection and path planning in 3-axis rough machining

by

Mahadevan Balasubramaniam

B. Tech., Mechanical Engineering (1997)
Indian Institute of Technology, Madras.

Submitted to the Department of Mechanical Engineering
in partial fulfillment of the requirements for the degree of

Master of Science in Mechanical Engineering

at the

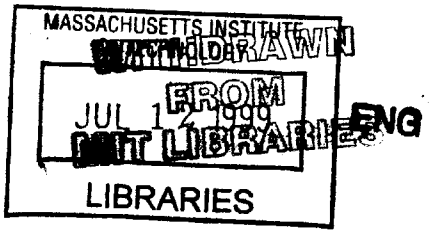
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 1999

Author .....
Department of Mechanical Engineering
May 8, 1999

Certified by .....
Sanjay E. Sarma
Assistant Professor of Mechanical Engineering
Thesis Supervisor

Accepted by .....
Prof. Ain A. Sonin
Chairman, Department Committee on Graduate Students



Tool selection and path planning in 3-axis rough machining

by

Mahadevan Balasubramaniam

Submitted to the Department of Mechanical Engineering on May 8, 1999, in partial fulfillment of the requirements for the degree of Master of Science in Mechanical Engineering

Abstract

We present a novel approach to CAD/CAM integration for 3-axis machining. Instead of redefining the workpiece in terms of machining features, we generate tool paths directly by using the accessibility of the surface of the part as the central and the over-arching constraint. This eliminates the problem of feature extraction. We envision this as the core strategy of a new direct and seamless CAD/CAM system. We perform our analysis in two steps: *global roughing* and *face-based finishing*. In this thesis, we present algorithms to select optimal roughing tools and to generate roughing tool paths directly from the shape of the model using geometric volume filling algorithms for a given setup. Assuming that the part is already oriented in a given setup direction, our approach to global roughing is based on slicing, clipping and filling. We slice the component into a number of layers, clip lower layers to the vertical shadows created for upper layers (because shadowed regions cannot be reached by a non-overhung tool), and generate tool paths to fill each slab. In this way, we can generate interference free tool paths directly from the boundary representation. The clipping approach permits us to account for the shapes of tool holders and spindle while computing the accessibility of a tool-assembly. Our approach to tool selection is based on a comprehensive computation of the area the tool can access at each slab. In the process of selecting an optimal tool sequence, we develop methods to rank the material removal rates achievable by using metal cutting theory. The search space for optimal tool sequence selection, is drastically reduced by using cluster-ordering techniques and then pruning the ineffective tools from the sequence using a greedy algorithm. Once the tools and sequence have been selected, a contour-shrinking tool path is generated using a Voronoi Diagram. The output of this system is a numerical control(NC) tool-path code. This information is generated directly from the CAD representation with no operator assistance.

Thesis Supervisor: Prof. Sanjay E. Sarma
Title: Assistant Professor of Mechanical Engineering

Acknowledgments

The thesis is the outcome of two of my best years of learning and research. It started with me joining as a research assistant under Prof. Sanjay Sarma. Sanjay had been very supportive throughout the duration of my research and also helped me to grasp the fundamentals of Computational geometry, CAD/CAM and Computer graphics in a matter of months. Sanjay was very freely accessible for discussion about new ideas in research: I cannot forget the numerous trips to Toscannini and Indian restaurants, where we would have brain-storming sessions to study the feasibility of new ideas.

I could not have asked for anyone better than Laxmiprasad to do research with. Laxmiprasad laid the framework for this technology and played an important role in the implementation of the technology as well. I am fortunate to interact with my lab-mates Elmer, Stephen, Taejung, Seung-Kil, Niranjan, Yogesh, Marty, Samir and Paula.

I could not have gone through all this without the help of *LORD ALMIGHTY*, my parents and sister's family. Their constant support, love and encouragement was instrumental in successful completion of the thesis. The note would not be complete without mentioning my friends Harish, Venkatesan, Sreeram, Suvranu and Mahesh for their constant support during my stay in the U.S.

I thank everyone mentioned above and others whom I may have missed here, for making my stay at M.I.T. pleasant and productive.

Table of Contents

List of Figures

List of Tables

Chapter 1: Introduction

Machining is the most widely used process today for producing functional mechanical prototypes. Machined parts can be obtained in a variety of materials with good finish and accuracy. However, machining is not usually considered a "rapid" prototyping process because it requires considerable effort and expertise, both intellectual and manual, to plan and operate machine tools like milling machines and lathes. In recent years this has lead to many attempts to automate machining and integrate it with computer aided design. This is commonly referred to as computer aided manufacturing (CAM) and CAD/CAM integration.

Over the last decade, the CAD/CAM research community has developed the concept of *machining features* to assist in the conversion of design information into machining instructions. Machining features are 2-1/2 D shape primitives defined in terms of access directions, and mapped to pre-determined, parametrized cutting paths. Typical CAM systems today require input in the form of these features; in turn, they generate low-level cutting instructions by "fleshing out" the details from the parametrized input. Machining features have proved to be convenient because they characterize the capabilities of machining processes such as 3-axis milling and turning fairly well. For example, the important classes of 3-axis cutting operations are end-milling, face-milling and drilling. The machining features that correspond to these operations are pockets, faces and holes respectively. There is little doubt that the concept of features has been a major step forward in the automation of machining, and remains an important avenue of research.

Yet, the feature based approach is not without its disadvantages. Firstly, any feature based system is limited by the extent of its vocabulary. Features are essentially 2 1/2 D entities that work well in prismatic parts. But if the workpiece has a complicated spline surface, then representing it with a set of features is a tough, in some cases impossible, task. Secondly, machining features are not directly available from CAD representations. They must be extracted by a process that is referred to as *feature extraction.* Although there has been some promising research in feature extraction in recent years, no commercially viable solution has yet emerged. Commercial CAM systems and featured based design systems circumvent the recognition problem by requiring the designer to *recreate* the shape in terms of the primitives defined in the system. Since this strategy places the onus of feature extraction on the manufacturing engineer, it is time-consuming, and to an extent, defeats the purpose of generating an initial CAD representation.

Therefore, despite recent strides in feature-based techniques, CAD/CAM integration remains a time-consuming and expensive step in machining. The operation of commercial CAM systems involves considerable operator skill, which is often difficult to come by. It has been argued that for parts of medium complexity, CAD/CAM may be responsible for up to 20% of cycle time and a considerably greater fraction of the actual cost. In order to make machining technology more accessible in today's demanding industrial environment, it is necessary to explore other paradigms which may, in the future, overcome the limitations of existing approaches.

1.1 Background

There has been a large body of work in CAD/CAM integration. Below we summarize this previous research.

Feature based machining: The concept of machining features has been an important step in the understanding and development of manufacturing planning. Machining features have the following advantages: 1) features are a convenient decomposition of a CAD model into handleable units for high level planning; 2) tool-path generating algorithms can be developed and implemented up-front; 3) since features fit the object-oriented model well, tool selection and cutting parameter selection can be linked cleanly to knowledge bases; 4) machining features implicitly define access directions and accessibility volumes. The first mention of features is probably by Krypianou [Krypianou 80]. The concept of manufacturing features first appears in [Arbab 82]. Arbab points out the similarity between the boolean difference operation in constructive solid geometry and the material removal in machining. This lead to the idea of destructive solid geometry (DSG), a design input methodology later refined in a series of papers: [Hummel 86, Kramer 88, Turner 88, Cutkosky 88, Shah 88 and Gindy 89]. In DSG, the user defines a "stock" and then *subtracts* primitives (features) to define the part. The development of process planning systems for machining has closely followed the development of features technology. Beginning with early work by Nau [Nau 86], Hayes [Hayes 89], Anderson [Anderson 90] and Cutkosky [Cutkosky 90], to more recent papers by [Yut 95, Gupta 95, and Sarma 96], the use of features has become better understood and more widespread.

Meanwhile, there has been interesting research in feature extraction in recent years. Seminal work on feature recognition was done by Woo [Woo 94]. Later, Joshi [Joshi 88]

used graph-based heuristics to extract features from adjacency graphs. [Dong 88, Sakurai 90, Finger 90 and Vandenbrande 90] made important contributions to the field. Kim extended Woo's work on convex decomposition [Kim 90]. Gadh introduced the concept of depth filters for feature recognition [Gadh 92]. Nau *et al* introduced the idea of generating alternative, optimal machining volumes in [Nau 92]. Recently, Regli has reported a promising new approach to feature extraction in his Ph. D. Dissertation [Regli 95]. His approach is based on the extrapolation of "maximum cover features" for 3-axis machining from the faces of a boundary representation. In general, most feature-based approaches have been limited to three-axis machining.

Surface machining: The field of surface machining has been a similarly intense area of research in the last few years. Since Faux' widely used book [Faux 81] a number of systems have been developed over the years for surface machining with special emphasis on die-mold applications: [Oetjens 87, Loney 87, Kuragano 88, Choi 89]. Most early systems, however, were either 3-axis based, or were relatively limited in their applicability because of problems of gouging and surface finish. Recognizing this problem, a few researchers in recent years have looked into the simulation of multi-axis cutting: [Oliver 86, Jerrard 89, Jerrard 91]. The issue of global interference is discussed in [Choi 89, Tseng 91 and Elber 94], and most recently by Lee *et al* [Lee 92, Lee 95, Lee 96].

Accessibility based approaches: The problem of tool access has been approached from both, a solids, and a surface perspective. Seminal work in the area of visibility and visibility maps was performed Chen & Woo [Chen 92, Woo 94]. They introduced the concept of visibility cones for points on a workpiece, which can be mapped on to the unit sphere to create a Spherical Map. The same authors also show how the Gaussian projection can be

extended with a central projection to manipulate access information and minimize setups. The idea of Spherical maps has been adopted by Wuerger and Gadh [Wuerger 95] to evaluate the separability of dies. The concept of access is also handled in a feature-based approach in [Sarma 96]. The ideas of a visibility cone have influenced surface machining as well. Lee [Lee 95] uses a convex hull based approach to approximate local visibility. An innovative approach to surface accessibility is presented in [Elber 94], in which convex surfaces are mapped to a space in which they become planar. Obstacles to the surface are also mapped into this space, and tool-path generation is carried out in a 3D world.

Commercial CAD/CAM systems: The commercial CAM systems are feature based and the user of the system must perform additional tasks of selecting a tool, selecting a cutting strategy, and selecting a cutting order. As a result, 3-axis machining is still very much an acquired skill today. Recent awareness of these problems has lead to interest in a new technology called **Generative NC**. SDRC has recently offered an early version of its Generative NC package.

1.2 Our Approach:

In Figure 1.1 shows two possible paths of generating tool-paths from the CAD model. One approach is to extract the features from the CAD model and then generate the tool-paths to machine the features. Eventhough the path for generating tool-paths from machining features is well-proven, the bottleneck operation is the extraction of the machinable features of the CAD model. The second approach is to generate 3-axis tool-paths directly from the boundary representation of the geometric object and we refer to this as Art-to-Part Machining.

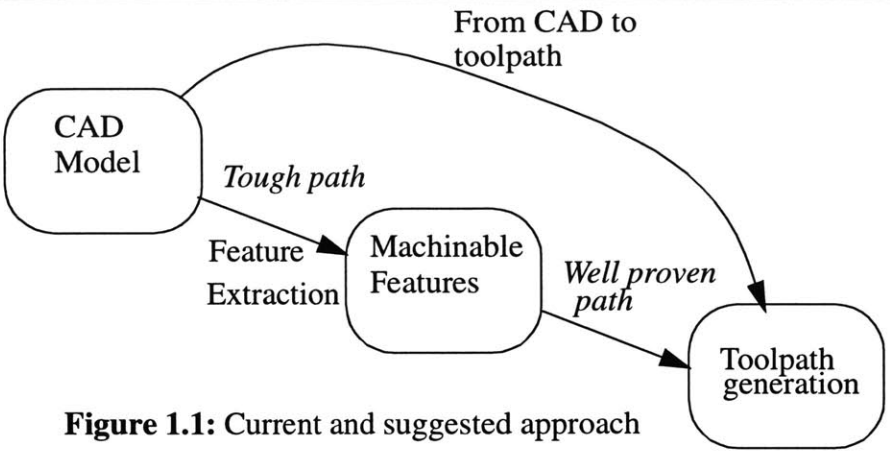


Figure 1.1: Current and suggested approach

The key idea in Art-to-Part Machining is simple: we will generate free-form cutting paths to remove all the excess material from the stock using accessibility as the central and over arching constraint. Little effort is devoted to the organization of tool-paths into formal primitives like features. The analysis is performed in two steps: *global roughing* and *face-based finishing*.

We present algorithms to select optimal roughing tools and to generate roughing tool paths for machining a part oriented in a given setup direction. Assuming that the part is already oriented in a given setup direction, our approach to global roughing is based on

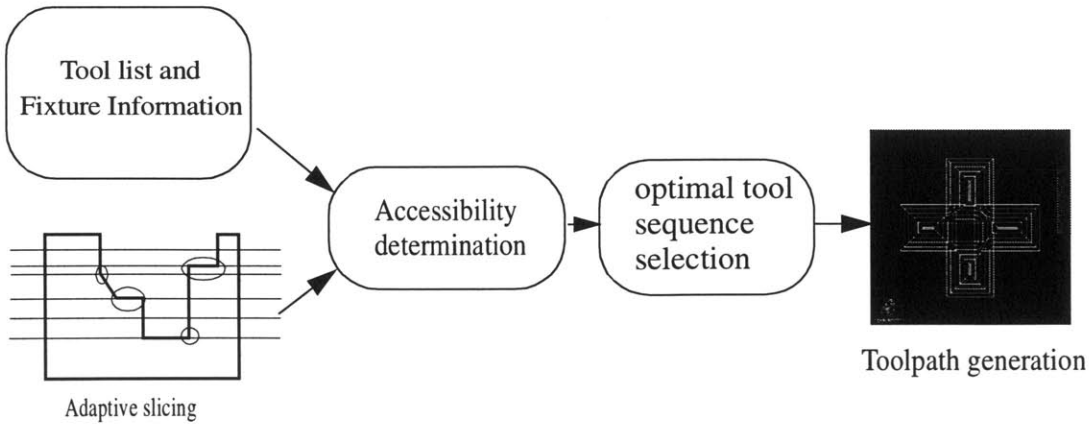


Figure 1.2: Our approach to CAD/CAM

slicing, clipping and filling. We slice the component into a number of layers, clip lower layers to the vertical shadows created for upper layers because shadowed regions cannot be reached by a vertical tool. The accessible area of the tool is computed at every slice of the object. Unlike many previous systems, the clipping approach permits us to account for the shapes of tool holders and complex tool assemblies during tool selection. Once the optimal tools and tool sequence have been determined [Veeramani 97], we generate interference free 3-axis tool paths using techniques of Voronoi Diagram. The graphical description of the various stages of the algorithm is shown in Figure 1.2.

Chapter 2: Layering of the object

Assuming the stock has been fixtured for a particular setup direction, the steps involved in layering the object are selecting slice plane positions, obtaining the slices of the object at those slice planes and then merging slices to form $2\frac{1}{2}D$ machinable slabs. The various steps are discussed in detail below.

2.1 Generation of 2D object-sliceplane intersection contour

The first step in layering the object is to determine the 2D sections of the workpiece at the sliceplanes. The sections below describe the data format for representing the object and an algorithm to determine the section contours corresponding to the sliceplane.

2.1.1 Tessellated representation

The first step of slice generation is in tessellating the 3D object. In this representation the surface of the object is filled with triangles and stereolithography(STL) format is an example of such a representation. Figure 2.1 shows the 3D model of an object along with its tessellated representation. The tessellated representation enables implementation of

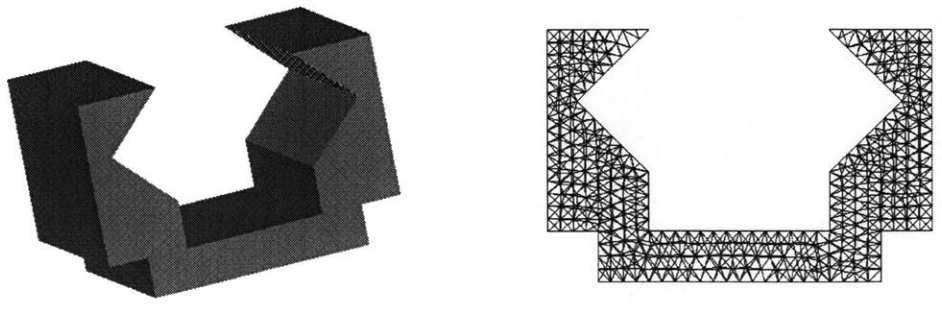


Figure 2.1: An object and its tessellated representation

very rapid slicing algorithms because the problem is reduced to a triangle plane intersection, followed by stitching edges to from closed contours.

2.1.1.1 Problems with the STL representation

The data format for a STL representation of a 3D object consists of defining the triangles on the surface of the object. A sample STL representation of a triangle is shown below:

```
facet normal 0.000000e+000 0.000000e+000 1.000000e+000
  outer loop
    vertex  4.330127e-001 -2.500000e-001 2.500000e-001
    vertex  4.330127e-001 0.000000e+000 2.500000e-001
    vertex  1.801342e-016 -3.685678e-016 2.500000e-001
  endloop
endfacet
```

Each triangle is represented by 9 floating point numbers corresponding to the x, y and z coordinates of its three vertices and by its outward normal direction. The inherent flaw in this representation is that even shared edges between adjacent triangles is represented twice in floating point values of its vertices. This leads to geometrical flaws, such as tears in the surface of the mesh, that will lead to the tripping of the slice generation algorithm due to inaccuracies in floating point computation.

2.1.1.2 Indexed Mesh representation

This problem was avoided by converting the STL mesh into an Indexed mesh format where the vertices of the mesh are represented once and the edges of the triangles have pointers to its vertices. Thus this representation has minimal storage of floating point numbers and the geometric information like equality of edges, equality of the intersecting

point and adjacency can be accurately determined. An Indexed representation of a two triangles is shown below:

Vertex coordinate file

<Vertex Number> <x-cood> <y-cood> <z-cood>

1 4.330127e-001 -2.500000e-001 2.500000e-001
2 4.330127e-001 0.000000e+000 2.500000e-001
3 1.801342e-016 -3.685678e-016 2.500000e-001
4 1.801342e-016 -2.500000e-001 2.500000e-001

Triangle file

<Triangle Number> <Vertex 1> <Vertex 2> <Vertex 3>

1 1 2 3
2 3 4 2

If the tessellated representation is generated from the original BRep file, each triangle is given additional attributes — a description of the surface from which the triangle was derived, its actual normal direction and its actual radius of curvature. These attributes provide a level of geometric and topological detail that most triangulated formats miss. We have used ACIS™ geometric modeler to generate the 3D models and the tessellated mesh was generated with the necessary parameters. The parameters that could be defined by the user are the surface deviation and the aspect ration of the triangle.

2.1.2 Intersection of object with slice plane

The tessellated model is intersected with a sliceplane and the triangles that intersect the plane are determined rapidly by checking the position of its vertices with respect to the sliceplane. An edge can be obtained from a triangle plane intersection and a list of edges can be obtained by intersecting all the triangles of the object with the sliceplane. The edges are stitched together to form closed contours by following the topological connections among the triangles that intersect the sliceplace [Jara-Almonte 92]. This

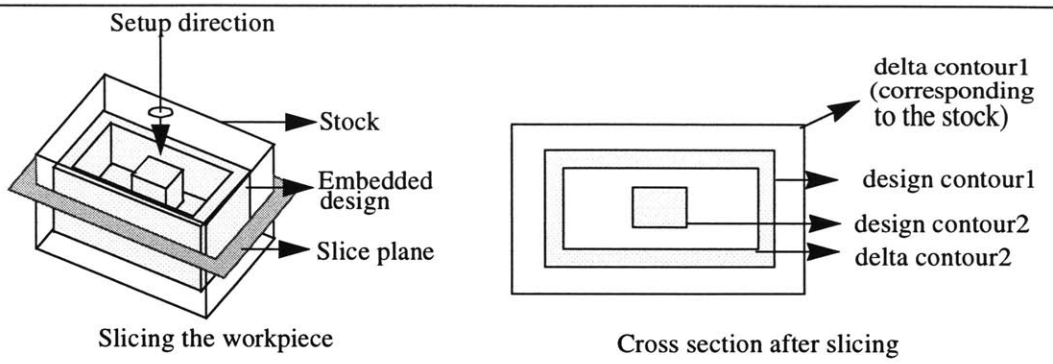


Figure 2.2: Part intersected by sliceplane

could result in a creation of several closed contours, where the contours correspond to either the embedded design or the delta volume. Since any part needs to be machined from a stock, a tessellated representation of the stock is used along with the model of the object to obtain the limiting contour. Figure 2.2 shows an example of a part intersected by a slice plane and the contours generated, where the outermost contour corresponds to the stock - plane and the other inner contours arise out of the embedded design - plane intersection.

The Slice Generation algorithm is given below:

Algorithm 1: 2D Slice Generation

Input:

Sliceplane Equation, P(x,y,z)=d
Tessellated Mesh, T

Output

List of Closed contours, ContourList

Algorithm

EdgeList = φ;
ContourList = φ;
For i ← to No of Triangles do
    Triangle_i ← get_ith_triangle

```
        isIntersect ← is_triangle_plane_intersecting(Triangle_i)
        If (isIntersect)
            IntersectingEdge ← get_Intersecting_Edge(Triangle_i, P)
            EdgeList ← EdgeList ∪ IntersectingEdge
    endFor
    addContour = φ;
    While EdgeList != φ do
        addEdge ← get_Next_addable_edge(addContour, EdgeList)
        insert addEdge to addContour
        if(addContour closed)
            ContourList ← ContourList ∪ addContour
            addContour = φ;
        EdgeList ← EdgeList — addEdge
    endWhile
    return ContourList
```

2.1.3 Representation of Contours

The contours that are obtained in the above slice generation algorithm need to be stored in the a data structure that is useful for further operations. The information about the area enclosed by a contour is required for clipping operation and the toolpath generation.

A 2D contour can be made to represent an area by assigning either a clockwise or a counterclockwise direction as viewed from the setup direction. The convention is that the region lying to the left of the edges of the contour is area that is enclosed by the contour.

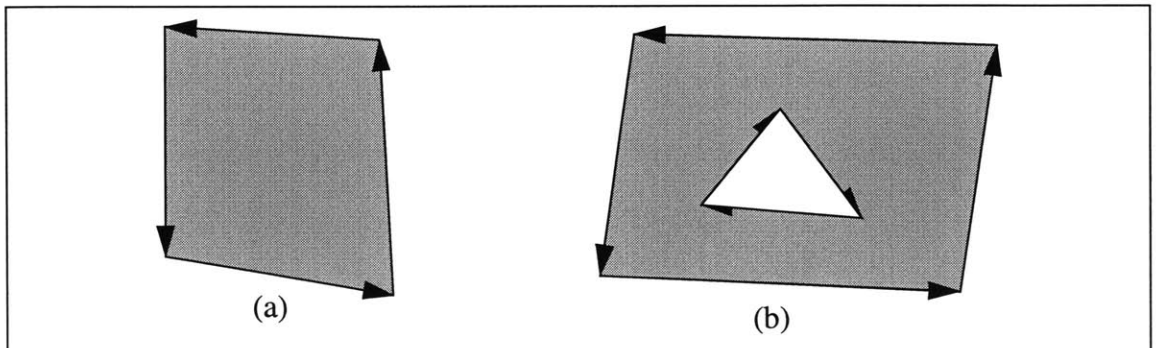


Figure 2.3: Directional contour and area enclosed

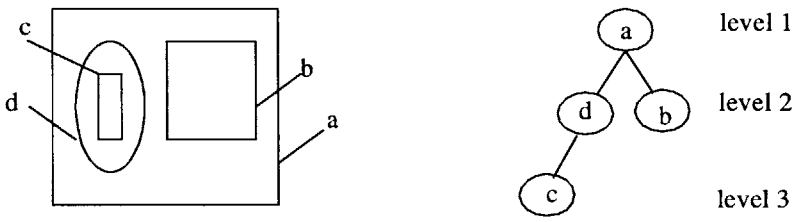


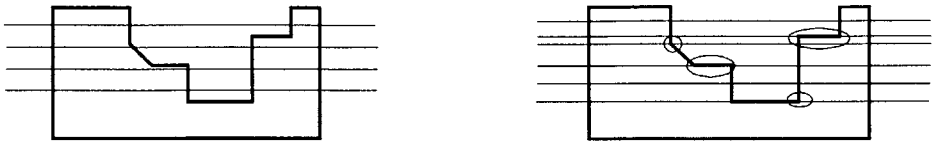
Figure 2.4: Ordered Tree

Figure 2.3a shows a counter-clockwise contour and the area enclosed by it. The idea can be extended to indicate voids in the area by assigning clockwise directions to inner contours. Figure 2.3b shows two contours with different direction and the area enclosed by it. Therefore, by assigning directions to the contours obtained from intersection we can represent the removal area corresponding to the slice plane.

The direction assignment for contours can be implemented by assembling the contours in an Ordered Tree based on the *insideness* of one contour with respect to the other contours. The parent-child relation is defined by the *immediate insideness* of one contour with respect to another. The tree can be formed on the basis of this parent-child relationship. The removal area is represented by assigning a counterclockwise direction to an odd-level contour and a clockwise direction to an even-level contour. By this direction assignment the removal area in an Ordered tree contour is always between a odd-level and a even-level contours. An example showing a contour and its Ordered tree representation is shown in Figure 2.4.

2.2 Selecting slice planes - Intelligent slice positioning

The naive way to generate sliceplanes would be to locate them at evenly spaced intervals. Obviously, this would be inefficient, as shown in Figure 2.5a, as it would be



a) Randomly placed planes approximate the feature poorly and may require more finishing

b) Intelligently placed slice planes can make roughing passes more effective.

Figure 2.5: Intelligent slicing

possible to miss important features such as horizontal faces and inter-surface edges [Dolenc 93]. In our approach, the locations of the slice planes are obtained by classifying the triangles in the tessellated representation as vertical, horizontal or inclined with respect to the setup direction. *Vertical* and *horizontal* faces are important to us because they can be machined most accurately during 3-axis roughing. Inclined planes must be approximated with stair-stepping passes. We identify possible positions of the slice plane as areas where a vertical or horizontal region begins or ends as shown in Figure 2.5b. This approach permits us to generate "tighter" roughing tool paths and hence achieve better cutting performance during finishing.

2.3 Forming 2-1/2D machining slabs - Intralayer clipping

There is always a depth of cut associated with machining and every layer machined by a tool will have a thickness. We refer to the $2\frac{1}{2}D$ layer machined by the tool as a *slab*. A slab is bounded by two contours: an upper slice contour and a lower slice contour. The use of vertical tool in a 3-axis machining makes accessibility at a lower layer analogous to a vertical shadow of the upper layer because we cannot reach under the upper layer. It is therefore Therefore, to form a slab it is necessary to fit a $2\frac{1}{2}D$ cylinder that defines the

common area that could be accessed without crossing the boundary of the lower and the upper slice contour. If the sides of the machinable slab are sloping then the cross-section of the cylinder will differ from the upper and the lower slice contours. The cross-section of this cylinder is the defined mathematically as the boolean intersection of the areas represented by the ordered tree contours at the upper and lower slices. We refer to the clipping between the layers of the slab as *Intralayer clipping*. The algorithm and the preliminary C-language code for performing boolean operations on polygons has been developed by Leonov [Leonov 95]. The C-language code was made robust by implementing GRID based Integer computation [Appendix A]. In brief the algorithm proceeds as follows.

Input:

- Ordered Tree Contour A
- Ordered Tree Contour B
- {*Operation*} - UNION, INTERSECTION, SUBTRACTION AND XOR

Output:

- Ordered Tree Contour C = A {*Operation*} B

Steps of Algorithm:

1. Intersect the two polygons represented by A and B
2. The intersection points with the other polygon are added as new vertices. Every added vertex has the information of the edges from which it was derived.
3. Label all the (new) edge from both polygons into Inside, Shared, or Outside with respect to the other polygon.

4. A set of start rules are specified for every operation

Operation	Start Rule
A INTERSECTION B	edge is (Inside) or (Shared and the directions of shared edges are the same)
A UNION B	edge is (Outside) or (Shared and the directions of shared edges are the same)
A SUBTRACTION B (edge in A)	edge is ((Outside) or (Shared and the directions of shared edges are the different))
A SUBTRACTION B (edge in B)	edge is (Inside)

Table 2.1: Start rules for Boolean Operations on polygons

5. A set of jump rules at intersections are specified for every operation

Operation	Start Rule
A INTERSECTION B	edge.link is (Inside)
A UNION B	edge.link is (Outside) and edge.next is not (Outside)
A SUBTRACTION B (edge in A)	edge.link is (Inside) and the current direction will be changed to backward
A SUBTRACTION B (edge in B)	edge.link is (Outside or Shared) and the current direction will be changed to forward

Table 2.2: Jump rules for Boolean Operations on polygons

6. The closed contours are collected by starting at an edge based on the start rule and then applying appropriate jump rules at intersections. The collected contours are added to the Ordered Tree Contour-{C}.

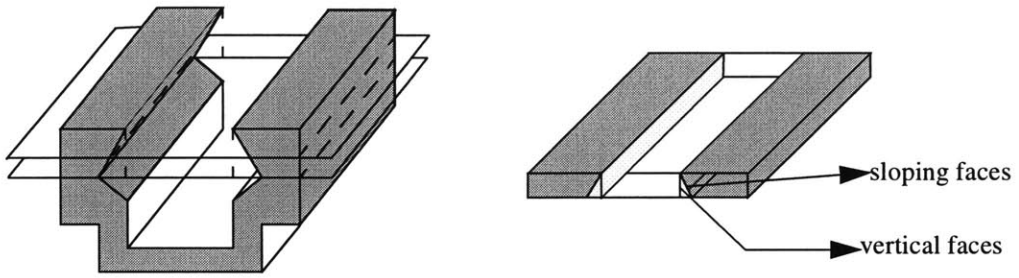


Figure 2.6: 2-1/2D machinable slabs

In Figure 2.6, an object with the 2 sliceplanes is shown along with the slab they create. Since the actual object has sloping faces, the contours corresponding to the upper and the lower slices are different. However, the sides of the machinable volume are vertical because tools must be vertical in 3-axis machining.

Chapter 3: Accessibility of Tools

A point in the machinable delta volume is accessible by a tool if the tool can reach the point without interfering with the embedded design volume. The importance of determining the accessibility of a tool is explained in Section 3.1. The accessibility will depend upon the geometry of the tool that is being used to machine the workpiece. In the subsequent sections, we develop the idea of determining accessibility of simple tools. The idea is extended to obtain the accessibility of *real tools* at any machinable slab of the object.

3.1 Necessity for accessibility computation

In order to machine an area, we have to decide which tool needs to be used so as to minimize machining time. A common approach in the literature has been to use the *largest* possible tool that will fit the 2 1/2 D slice [Lee 94, Lee 96]. However, this is not necessarily the optimal tool because it may not in fact be able to sweep much of the area of the slice as shown in Figure 3.1. In Figure 3.1, a pocket with an island is considered for machining using a large and small diameter tool and the shaded area indicates the sweep areas of the tools. We can conclude that the trade-offs are as follows: large tools, though capable of higher material removal rates, have less reach than small tools; hence they

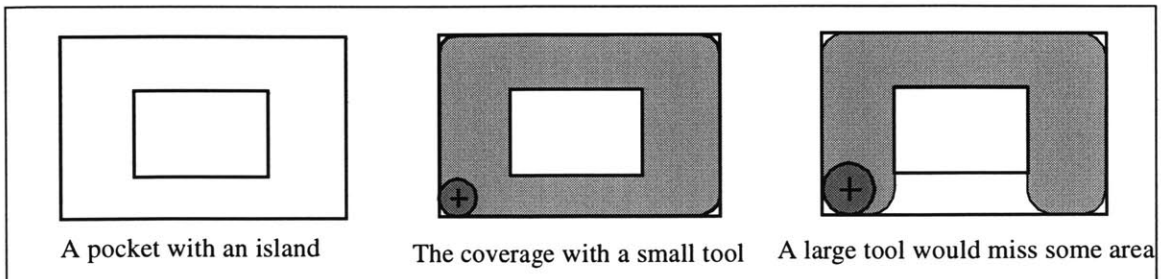


Figure 3.1: Sweep areas and tool diameter

incur the penalty of a tool change, which negates the advantages of higher material removal rates. The optimal tool sequence must therefore be selected based how much area each tool can access in all the slabs. This accessibility computation is dependent not only on the diameter of the tool, but also on the overall geometry of the tool assembly including the tool holder and the spindle. This consideration is one of the distinguishing features of our approach, and is described in greater detail in the following sections.

3.2 Accessibility determination of simple tools

Geometrically, a simple tool is modeled as a semi-infinite cylinder of the required diameter. The analysis for the simple tools forms the building block for determining the accessibility of complex *real tools*.

In order to machine a delta volume, the centerline path of the tool must be determined. A limiting centerline tool-path is defined as the path in which the tool grazes the embedded design without gouging it. Since the tool has a finite diameter, the limiting centerline toolpath cannot be the boundary of the area being machined. The limiting centerline tool-path for machining a contour can be obtained by offsetting the contour inwards by the radius of the tool. The tool can freely trace the region interior to the limiting offset contour. The algorithm to offset an Ordered tree contour was developed by Laxmiprasad [LaxmiPrasad 98]. Geometrically, the region accessed by the tool when it sweeps a region bounded by the limiting offset contour can be obtained by *re-offsetting* the limiting offset contour *outwards* by the radius of the tool. Since, the logic for re-offsetting is similar that of offsetting except for the edges being shifted in the opposite direction: the algorithm for re-offsetting a contour can be extended from the offsetting algorithm [LaxmiPrasad 98].

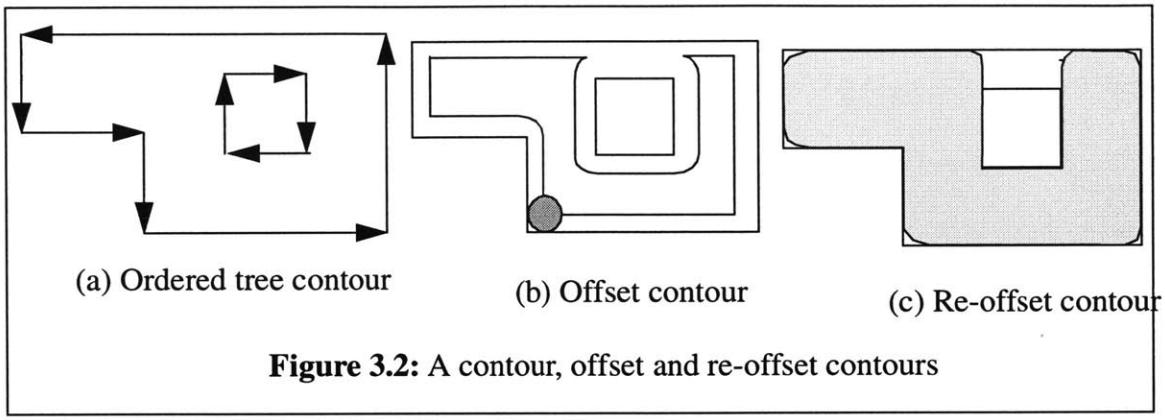


Figure 3.2: A contour, offset and re-offset contours

This seemingly regressive step provides us with important information: it tells us how much cutting a tool can actually perform; it is therefore an important metric of the efficacy of the tool.

Figure 3.2a shows an example of an Ordered tree contour being machined by a simple tool. Figure 3.2b shows the offset contour corresponding to the tool formed by offsetting the Ordered tree contour. Figure 3.2c shows the area the tool can reach in the Ordered tree contour.

3.2.1 Accessibility determination for multiple slabs - Interlayer clipping

The following observation can be made about accessibility:

> If a point p, in the delta volume of the part, is accessible by a non-overhung tool from a direction O, then it should be visible along direction O. In other words, for simple tools, accessibility implies visibility and to arrive at the accessibility from the visibility information a check for interference needs to be performed. Therefore, as viewed from the setup direction if we ensure visibility of a point at a lower slab and absence of interference between any part of the tool and the embedded design then the point is accessible by the tool. The visibility of a point at a lower slice can be

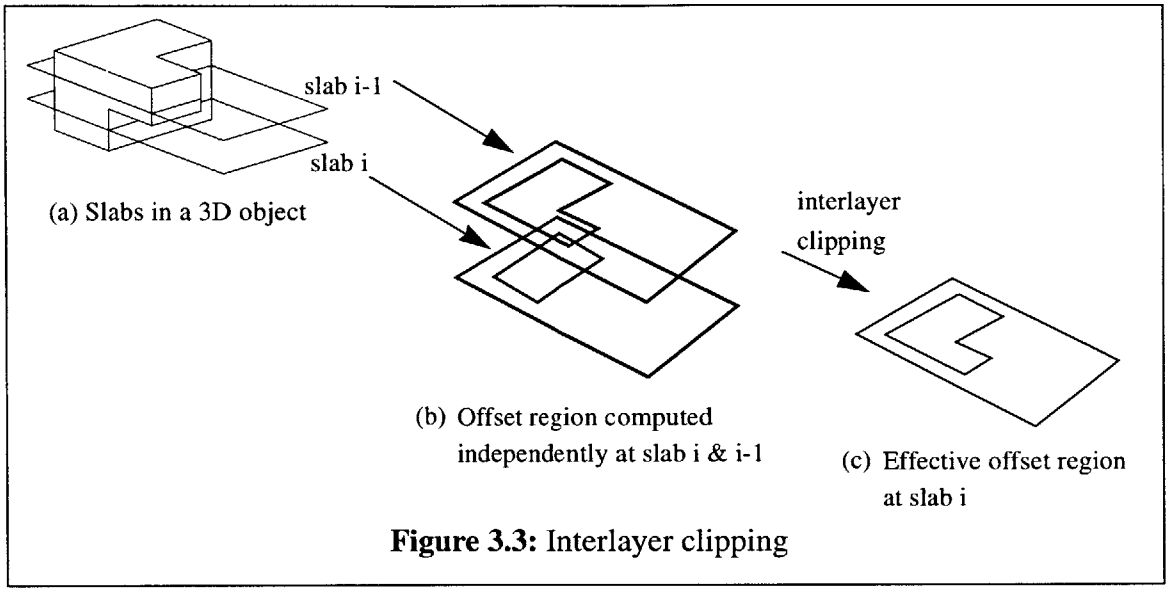


Figure 3.3: Interlayer clipping

ensured by the shadowing operation and the accessibility can be ensured by offsetting and re-offsetting operations. Therefore accessibility can be geometrically obtained by performing sequence of offsetting, re-offsetting and boolean operations on polygonal areas.

Consider a simple tool that is machining a part made up of n slabs. From the observation made in the previous paragraph, the center-line path of the tool at slab i, will not only be decided by the slab i but also by the previous slab i-1. The accessibility of a tool at a lower slab is dependent on all the upper slabs because in 3-axis machining, a tool can trace only the common region between the offset paths of the current slab and that of the previous slabs. In other words, in 3-axis machining, as we go deeper into the work piece the offset region decreases monotonically. Accessibility is therefore similar to a vertical shadow and can be geometrically realized by performing an polygon intersection operation. This is the second example of clipping in our approach, and we refer to this as *interlayer clipping*. A 3D object with i and i-1 slabs is shown in Figure 3.3(a). The offset

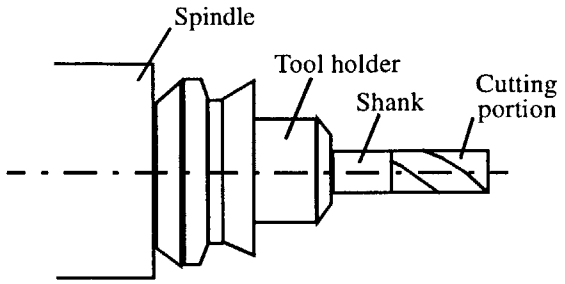


Figure 3.4: Milling tool

region corresponding to the delta area that is to be removed independently at slabs *i-1 and i* is shown in Figures 3.3(b). The feasible offset region shown in Figure 3.3(c) is obtained by intersecting the offset regions of i^{th} and $i\text{-}1^{th}$ slab. Applying the above mentioned technique in a top-down manner i.e. from the first slab to n^{th} slab we can obtain the feasible offset region of the tool at all the slabs. The feasible accessible area can be obtained at the slabs by re-offsetting the feasible offset regions of the corresponding slabs by the radius of the tool. Section 3.4 discusses the algorithm for using the feasible offset region of simple tools to determine the accessibility of tool assemblies.

3.3 Modeling "Real tools"

Figure 3.4 shows an actual milling cutter that will be used in milling machines to hog out the material. A real cutter has several parts – the cutting portion of the tool, the shank of the tool, the tool holder and the spindle. It is very common for portions of the tool assembly other than the cutting portion to "enter" the workpiece, especially when the length of the tool is small or the component that is being machined is large. A feasible tool path is that in which no part of the tool enters the boundary of the final desired part while removing material from the stock.

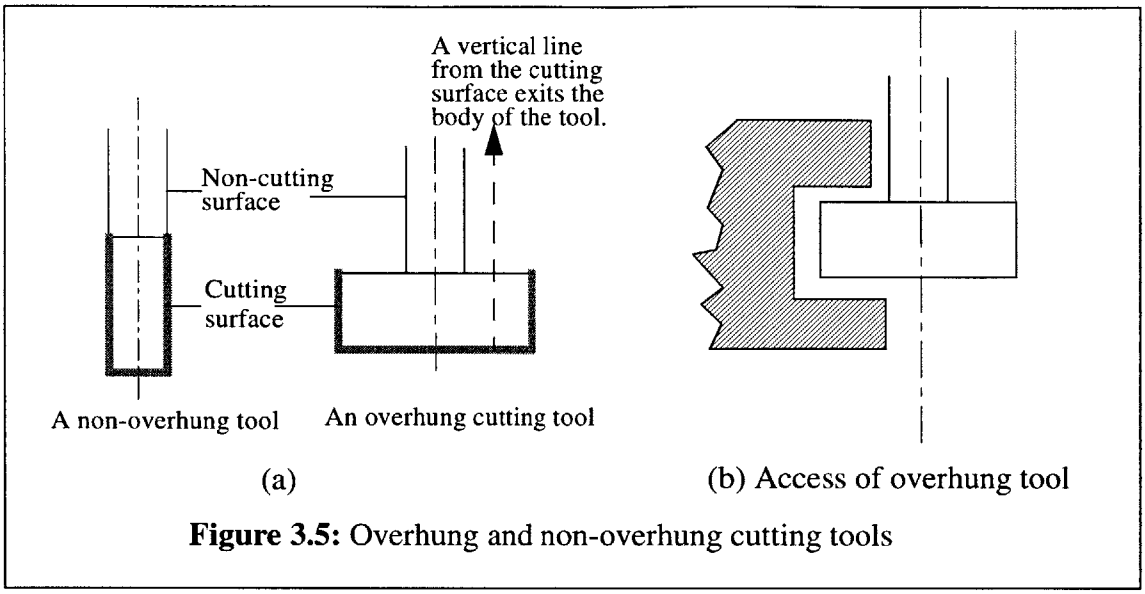


Figure 3.5: Overhung and non-overhung cutting tools

3.3.1 Types of cutting tools

The surface of a cutting tool in a standard, vertical, milling center can be divided into two regions: the cutting portion and the non-cutting portion. We define a non-overhung cutting tool as one in which any ray upwards along the axis of the tool from a point on the cutting surface does not exit from the body of the tool. A tool in which this condition is not satisfied will be referred to as an overhung cutting tool.

Most standard cutting tools like face-mills, end-mills and drills can be adequately approximated as non-overhung tools. Overhung tools are those which have either an upward cutting region, like T-slot cutters, or narrower shanks in the upper part of the tool. Note that the latter kind of overhung tools can be approximated as non-overhung tools. Of course, the approximation of an overhung tool as an non-overhung tool reduces our ability to reason about access in "undercut features". Figure 3.5 shows examples of overhung and non-overhung tools and the ability of a overhung tool to reach undercut features. In this thesis, we will limit the discussion to non-overhung tools.

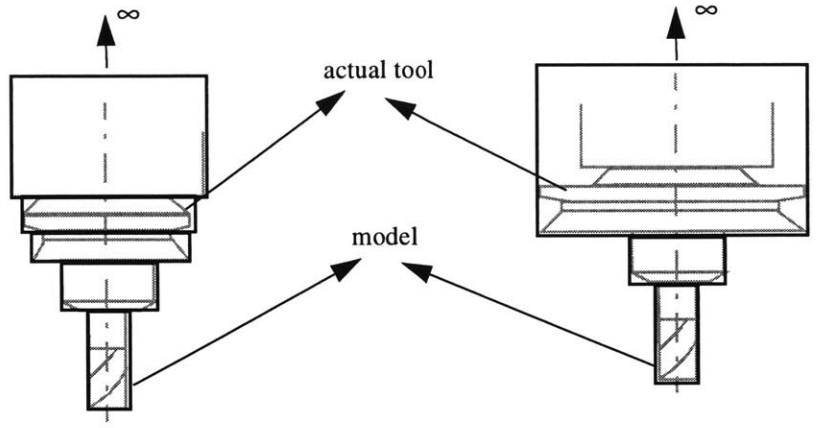


Figure 3.6: Actual tool and its non-overhung model

3.3.2 Modeling "Real tools" as *non-overhung* tools

A tool can be approximated as a non-overhung tool by modeling it as a union of simple semi-infinite cylinders of increasing diameter with their bounding planes separated by an "appropriate" overhang distance. We refer to this *non-overhung* approximation of a tool as the tool-assembly. Inorder to be conservative in our approximation as a *non-overhung* tool, the cylinders are positioned such that the actual tool is contained within the approximated model. Figure 3.6 shows the actual tools in dimmed lines along with their approximation.

3.4 .Accessibility determination for tool-assemblies

Consider a tool-assembly entering to machine the slabs of a delta-volume. The accessibility of the tool-assembly at slices will be the same as cutting portion as long as the other portions of the tool-assembly does not enter the delta volume. The effect of the tool holder will be to reduce the accessible areas in slices deeper than the first overhang in the tool assembly because we must consider the collision of the tool holder with the work

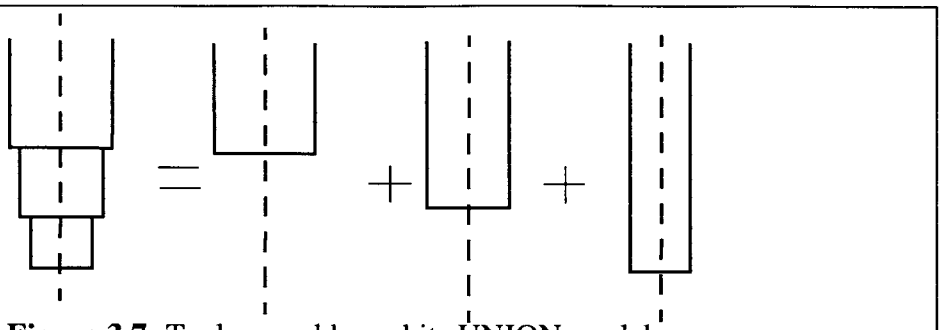


Figure 3.7: Tool-assembly and its UNION model

piece. Below we describe in detail how a tool assembly could be modeled as a non-overhung tool and methodical way of determining its accessibility.

3.4.1 Accessibility of tool-assembly at a slab

As discussed in Section 3.3.2, an actual tool can be approximated as a non-overhung tool-assembly and its constituent cylinders can be identified. Geometrically, the tool-assembly can be modeled as an *union* of several simple tools (semi-infinite cylinders) separated by the overhang. Figure 3.7 shows a tool-assembly being modeled as an *union* of simple tools.

An outline of the approach would be to generate the offset region of all the constituent cylinders and then merge them based on the geometry of the tool and the object. Below we list the steps of the algorithm to compute the feasible offset region of the tool-assembly:

Step 1: Generating the *offset regions* at each slab for every constituent semi-infinite cylinder

The constituent cylinders of the tool assembly are identified and as described in Section 3.2.1, the feasible offset region at every slab of the delta volume is determined for all the constituent cylinders.

Step 2: Generating the *feasible offset region* and tool sweep region for a *tool assembly.*

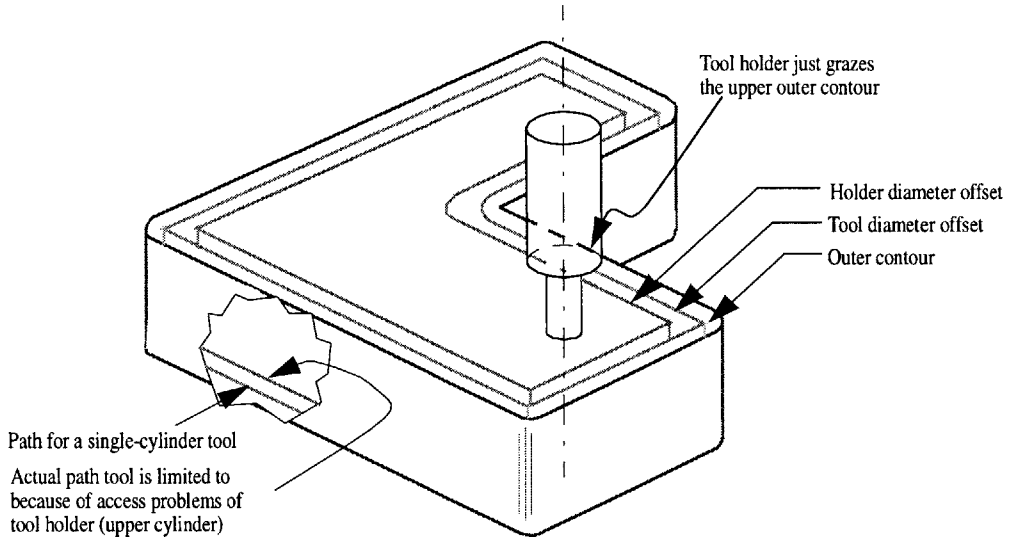


Figure 3.8: Offset profiles of two cylinder model

The tool assembly will pose a problem only when the upper portions which are larger, collide with the boundaries of the delta volume. It must be realized that the upper portions having a larger diameter will be tracing the same centerline path of the tool-tip at the lower slice. Thus for the tool-tip (centerline) to reach a point on the slab *i*, the point must lie in the region common to the feasible offset regions of the tool tip at the slab and the tool holder/assembly at their bounding slabs. The common offset region is obtained by the intersection of offset regions of the simple tool cylinders at slabs corresponding to the location of its bounding plane.

This is explained in Figure 3.8, where a 2 cylinder tool assembly is machining the pocket. The middle contour in the upper and lower slice is the offset contour for the smaller diameter, while the inner contour in the upper slab is the offset contour for the larger diameter. It is easy to see that the centerline of the tool should be contained within the inner contour to prevent the collision between the tool assembly and the embedded design. Geometrically, the feasible region for the tool assembly tracing

the i^{th} slab is given by the intersection of the offset regions of smaller cylinder at the lower slab and the larger cylinder at the upper slab. This approach can be easily extended for a multi-cylinder model, where we have to take the intersection of offset regions of all the constitutive cylinders at the slab they are tracing. The area reached by the tool-assembly at a slab can be got by re-offsetting the computed feasible offset region (at that slab) by the radius of the tool-tip.

3.4.2 Accessibility, when tool assembly is part of sequence

In reality, we will have a sequence of tools to machine a part. Therefore, it is necessary to determine the area a tool assembly can access when it is machining as a part of the sequence. The solution to this problem is pretty much the logical answer, where we have to keep track of the material already removed by the previous tool assemblies. The remaining stock information is used in conjunction with the accessible area to determine the net area machinable by a tool-assembly. Below we list the step to compute the accessibility of the tool assembly as a part of the sequence.

Step 1: Determining the area cut by a *particular tool assembly within a sequence* of tool assemblies selected to machine the delta volume of a part:

An update table contains the information about the total area that has been removed at each slab by the tool assemblies used in the sequence so far. When the i^{th} tool assembly in the sequence enters to cut the slice, the area that could be cut by the tool assembly alone for that slice is obtained from the data generated in Section 3.4.1. The update table has the information about the area that has already been removed at that slab by the previous i-1 tool assemblies. A Boolean subtraction of the two sweep areas [Leonov 95] is performed to determine the additional area that could be

removed by the new tool assembly. If the Boolean subtraction results in a positive area then the tool assembly will cut that area; otherwise the tool assembly will proceed to the next slab. If any area is removed then the update table is modified to the latest area removed from that slab.

The number of passes made by a tool-assembly to machine the area in a slab will depend on the thickness of the slab and the maximum depth of cut of the tool-assembly. For example, if the thickness of the slice was twice the depth of cut then the tool assembly will have to take two passes to machine the area at its allowable depth of cut. Therefore the area that the tool needs to sweep to machine the delta volume of the slab will accessible area at the slab times the number of passes. Thus the total area cut by each tool assembly in the component can be obtained by summing the areas machined by it at the various slabs of the object.

Chapter 4: Optimal Tool sequence selection and Path generation

In the previous chapter, given a tool sequence, a method of the determining accessibility area of a tool assembly as a part of the sequence of tools was explained. An optimal tool sequence is the sequence which best satisfies the machining objectives and Section 4.1 discusses several machining objectives. The tools available in the machine shop determine the input sequence for the algorithm. However, the computational complexity in evaluating all the possible tool-sequences is daunting. Inorder to reduce the computational burden, the initial sequence is derived based on heuristics.

Section 4.2.1 discusses the methods of rating tools, to determine the *utility*, based on the constraints from cutting mechanics and material strength. Section 4.2 addresses the complexity associated with evaluation of all the possible tool sequences and suggests some ordering heuristics to reduce the search space of the problem and a greedy algorithm to prune the ineffective tool-assemblies in the tool-sequence.

4.1 Machining objective

Any optimization problem will require an objective function to be specified. Below we suggest some common objective functions that are used in the industry.

Minimize total machining time with maximum tool life: The bottleneck operation in a large scale manufacturing enterprise will be the machining operation. Inorder to decrease the cycle time and minimize the number of interruptions due to a tool failure: the total safe machining time at a station needs to be reduced. Section 3.4.2 discussed in detail the algo-

rithm to compute the exact area machined by each tool-assembly as a part of the sequence in each slab. The total area cut by a particular tool-assembly j of the sequence A_j^{total} is the arithmetic sum of the areas it machines in different slabs. If m represents the total number of tool-assemblies used in the sequence the total time for machining can be computed as

$$TotalTime = \sum_{j=1}^{m} Kf(A_j^{total}, D_j) + (m-1)T_{toolchange}$$

Where f is some correlating function between a tool-assembly, the area machined at its depth of cut and cutting time. $T_{toolchange}$ is the time required to change to next tool-assembly in the sequence. Based on the theory of cutting mechanics and [Green 96], the feed per tooth is strongly dependent on the diameter of the cutter and weakly dependent on the depth of cut. While any correlation, including a lookup table can be used, we have found the following relationship derived from the cutting mechanics to be sufficiently accurate: $f(A_j^{total}, D_j) \propto \frac{A_j^{total}}{D_j}$

Minimize number of tool changes: In a rapid prototyping industry where the number of produced parts is very less, the interest would be to minimize the number of tools used to machine the part. The reason is that the *hidden cost* incurred due to a tool failure is not as much as that of a large scale manufacturing industry.

4.2 Complexity in initial sequence determination and heuristics

The initial sequence of tool-assemblies need to be formed from the list of available tools in the machineshop. If there are n tools, then theoretically there are 2^n permutations of tool sequences. The computational resource to evaluate all the alternatives would be enormous. In the effort to expedite the results, we must prune the search space by decom-

posing the problem in the interests of computational efforts. Below we present some heuristics to rate a tool based on its utility and reduce the search space by using cluster ordering techniques.

4.2.1 Rating a tool based on its utility

For the reasons described above, it is imperative to classify the tools based on the utility. A tool-assembly can be rated based on the Material Removal Rate (MRR), accessibility and cutting quality achievable by it. Tool-assemblies can be characterized by the diameters and overhangs of its constituent cylinders. Inorder to rate a tool, the following parameters need to be considered:

1. Diameter of the cutter portion of the tool-assemblies [discussed in Section 4.2.1.1]
2. Overhang of the constituent cylinders of the tool-assemblies [discussed in Section 4.2.1.2]
3. Diameter of the constituent cylinders of the tool-assemblies [discussed in Section 4.2.1.2]

4.2.1.1 Diameter of the cutting portion Vs. MRR and accessibility

The Material removal rate (MRR) for end-milling is given by $MRR = aDtf_t n_t N$ where

a is the fraction of the diameter used in milling

t is the depth of cut

f_t is the feed per tooth

n_t is the number of tooth present in the mill

N represents the spindle speed

From Taylor's tool life equation, restrictions can be applied on cutting velocity for maximizing the life of the tool. It can be shown that the optimal MRR is proportional to the depth of cut (*t)* and the feed per tooth (f_t). As discussed in Section 4.1, the f_t is strongly dependent on the diameter of the cutter and weakly dependent on the depth of cut. For a maximum tool life and higher MRR, it is preferable to use a larger diameter tool; however, the trade-off is the reduced horizontal reach of a large tool within a slice. On the other hand, with a small tool, it is necessary to operate at a lower MRR. With a small tool, the horizontal reach is greater but the vertical reach is lower because smaller tools tend to be shorter, and there is greater chance of tool-holder intersection. Thus we can conclude that access permitting, a larger diameter tool is better than a smaller diameter tool. If that is not the case, then the algorithms presented in Section 4.3 will prune the *ineffective* tools in the sequence.

4.2.1.2 Cylinder overhang and diameter Vs. MRR and cutting quality

The geometry of the composite tool can be treated as a cantilevered beam subjected to bending under the action of the cutting force *F.* While any measure depending on the user, could be used to classify the tool-assemblies, we have used the bending energy induced in the tool assembly as a good measure indicative of both the stress and the strain experienced by the tool-assembly.

A two cylinder tool-assembly is considered in Figure 4.1 to study the influence of overhang and diameter of the cylinders on the bending energy (our measure for rating a tool). In the Figure 4.1: D_1 and D_2 represents the diameter of the tool-tip and the tool-holder, l_1 and l_2 represents the overhang of the tool and the tool-holder. By applying a

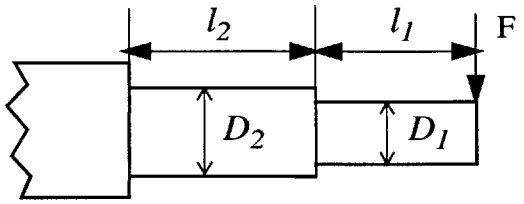


Figure 4.1: A cantilevered tool

force F at the tip of tool the bending energy can be computed as a function of the defined parameters. The parameters can be varied and their influence on the bending energy can be determined. Below, we summarize the observations made:

Bending Energy increases:

1. If l_1 increases and (l_1+l_2), D_1 and D_2 are maintained a constant.

Bending energy decreases:

1. If D_2 increases and l_1, l_2 and D_1 are maintained a constant.
2. If D_1 increases and l_1, l_2 and D_2 are maintained a constant.
3. If l_1 decreases and D_2 increases and l_2 and D_1 are maintained a constant.

We can conclude the following

1. A shorter tool overhang, results in a lower bending energy, thereby facilitates faster traversal rates (higher MRR) and better cutting quality (less chatter)
2. A larger diameter of the constituent cylinder, results in a lower bending energy. The effect of the diameter on bending energy is more than that of the overhang because of the higher exponent for the diameter in the analytical expression.

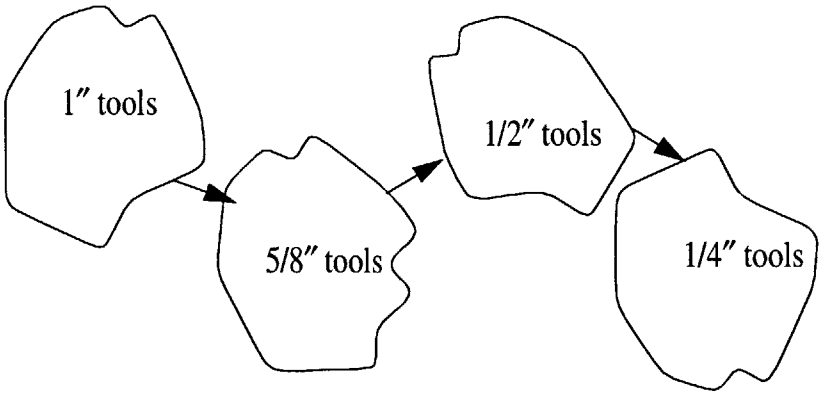


Figure 4.2: Cluster Ordering

4.2.2 Cluster ordering

As discussed in Section 4.2.1, access permitting a larger diameter tool will always be used before a smaller diameter tool. Therefore, it is possible to order all the tools in a shop in a series of clusters such that larger diameter tools come first. Figure 4.2 shows an example of the tools being organized into four clusters based on the diameter of the cutting portion of the tool-assembly. The combinations of the sequence are selected from the clusters, with each sequence picking up exactly one tool from each cluster. Figure 4.3 shows an

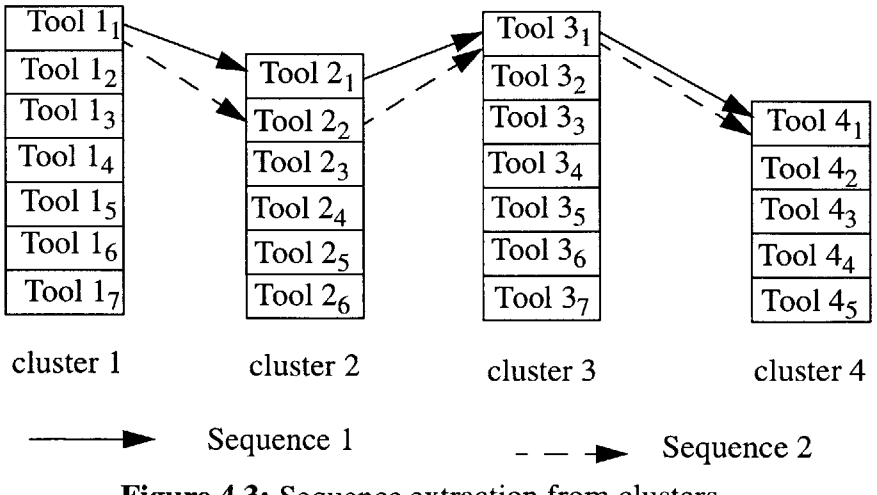


Figure 4.3: Sequence extraction from clusters

example where two sequences have been selected from among the four clusters. The cluster ordering reduces the number of combinations to be checked to $\prod_{k=1}^{K} l_k$ where l_k is the number of tools in each cluster and K is the total number of clusters. As discussed in Section 4.2.1, the tool-assemblies within a cluster can be ordered based on their overhang and the diameter of the constituent cylinders.

4.3 Determining optimal tool sequence

The previous section discusses heuristics for determining the initial tool sequence. Inorder to determine the optimal tool sequence from among the several initial tool sequences, the following steps need to be followed:

1. The objective function should be evaluated for all the initial sequences.
2. The objective functions for an initial sequence should be maximized by pruning the ineffective tools in the sequence. An algorithm for pruning the sequence rapidly has been suggested in Section 4.3.1.
3. Pick the tool sequence which maximizes the objective function

4.3.1 Tool sequence pruning

The initial sequence will give an ordering with exactly one representative from each cluster, therefore all tools in the initial sequence need not be efficient. Pruning is a process by which ineffective tools are removed from the sequence. A tool is ineffective if its presence in sequence decreases the objective function.

The steps for pruning the sequence for the objective function: *minimize total machining time and maximize tool life* is given below

Step 1: As explained in Section 3.4.2, the area machinable by each tool-assembly in the sequence is determined. The tool-assemblies that remove *zero* area, are removed from the sequence. This occurs when the tool-assemblies prior to this tool-assembly had machined all the area that this could have reached in the component. The tool-assembly sequence now has tool-assemblies that perform some machining only.

Step 2: Ineffective tools can be identified following a *greedy approach.* Since the initial ordering of the tool-sequence is based on the diameter, we ensure that later tools in the sequence have a lower Material removal rate. Therefore a trade-off needs to be made between the extra accessibility of the following tool-assembly with respect to the difference in the Material removal rate of these tools. The trade-off can be achieved by following this *greedy step:*

> If the additional time required by the $i+1^{th}$ tool-assembly to machine the accessible area of the i^{th} tool, is lower than the tool change time, then we conclude that the i^{th} tool is only adding to the cutting time.

We can use the above criterion to prune the ineffective tools by proceeding from the last tool-assembly to the first tool-assembly in the sequence. The pruning of the sequence ensures the removal of ineffective tools and the objective function can be evaluated for the final pruned sequence, which will be used to select the optimal sequence from among the several initial sequences.

4.4 : Tool Path generation

In rough machining, the aim is to machine the delta-volume rapidly and leave very little material for the finish machining. In the previous section, we discussed the methods for

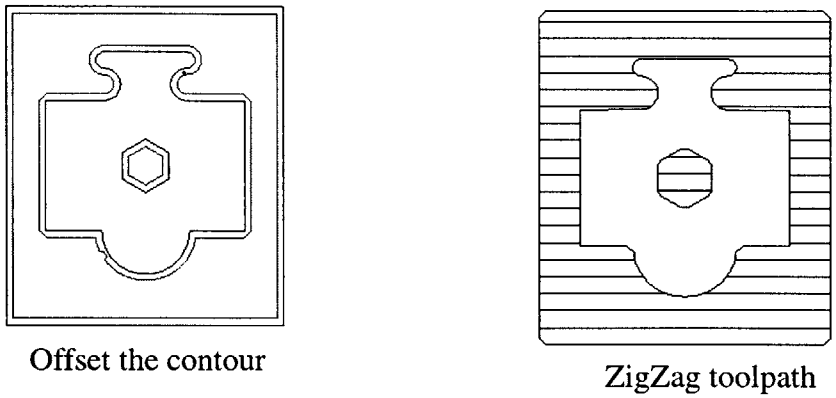


Figure 4.4: Zig-zag toolpath generation

determining the optimal tool sequence and the areas reached at every slab by the tool-assemblies of the sequence. Below we present the methods to fill the machinable area represented as an Ordered tree contour (Section 2.1.3) with tool-paths.

4.4.1 Tool path generation methods

The machinable area can be filled using either zig-zag toolpaths (Figure 4.4) or contour parallel toolpaths (Figure 4.5). The input for the tool-path generation problem are the Ordered Tree Contour representing the machinable area and the radius of the cutting portion of the tool assembly.

Zig-Zag toolpaths: The boundary of the center-line toolpath is determined by offsetting the machinable Ordered tree contour *inwards* by the radius of the tool. The boundary is scan-converted using a modified raster scanning algorithm [Foley 95]. The algorithm that identifies spans of lines that lie inside the contour is described briefly below:

1. The edges of the contour are sorted based on the coordinate perpendicular to the raster-scanning direction and an edge-table is created. The data-structure contains the information about the next edge, slope of this edge, maximum and minimum coordi-

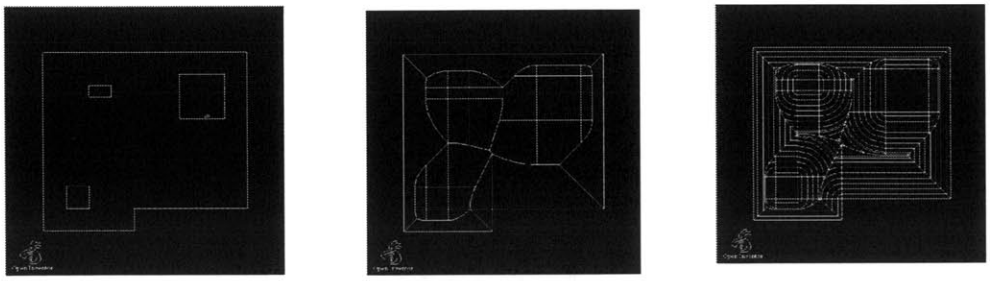
nates of a edge in a direction perpendicular to the raster scanning direction.

2. The contour is scanned in a top-down manner and an active edge list is created for the current scan. The active-edge list has information about the spans that lie within the contour.

3. When the active-edge list has several spans then the tool will have to be retracted at the end point of a segment and re-fed at the starting point of the next segment. Inorder to minimize the number of retraction and plunging of the tool, continuous sets of path segments are identified from among the spans by turning the tool to the next span line when it encounters an obstruction.

The output of the above algorithm will be continuous spans of the toolpath to machine the Ordered tree contour. The zig-zag toolpath is inefficient from the view point of dynamics of the machine tool. The machine has to accelerate rapidly in the machining sections and decelerate when it nears the end of the segment and make a turn and then accelerate again.

Contour Parallel toolpaths: The center-line tool-path with the least number of turns will be the one that is parallel to the contour and can be geometrically obtained by offsetting the contour. In this technique each contour element is individually offset by a pre-



Ordered tree contour voronoi diagram contour parallel toolpaths

Figure 4.5: Contour parallel toolpath generation

determined distance [Gurbuz 95, Held 1991, Tiller 84] to obtain the centerline tool-path. One method of obtaining the offset contour is to use Voronoi diagrams, which provide the locus of self-intersection points of all offsets [Held 93a, Held 93b, Yap 87]. The Voronoi diagram for a closed contour divides the contour into proximity areas. The contour area is divided such that the proximity areas have a one-to-one correspondence with the contour elements. Any point in the proximity area is closer to the contour element it is associated with than any other contour element. Once the Voronoi diagrams are generated, each contour element is offset, and then trimmed where it leaves its proximity area. This technique is efficient since the voronoi diagram is computed only once and then several offset contours are determined rapidly from the voronoi diagram. Figure 4.5 shows an example of a contour, its voronoi diagram and the contour parallel toolpaths.

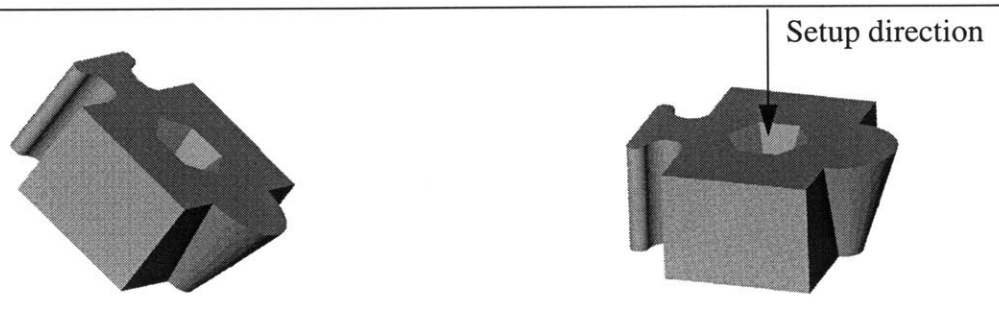
Chapter 5: Examples and Illustrations

The algorithms described in the previous chapters have been implemented in C++ language on SGI-IRIX platform. The results of the program are shown below for the simple cases to illustrate the proof of concept and it is perfectly scalable to handle complex cases also.

5.1 Accessible area determination for a simple tool

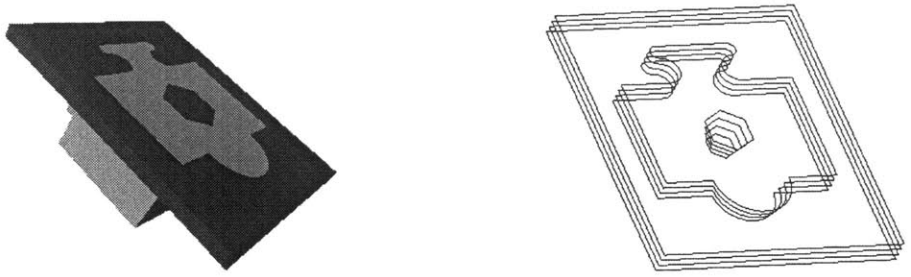
We illustrate below the various steps in determining the area accessible by a simple tool (semi-infinite cylinder) in the object. Figure 5.1a shows the 3D model of the object that is being machined. The part has the following features that need to be noted:

1. A half-frustum on the right side to illustrate the principle of shadowing in determining the accessibility
2. An oddly shaped extrusion on the left side to illustrate that our approach does not recognize the various shapes in the object
3. A tapering hexagonal pocket in the middle of the work-piece



a) 3D Object b) Object and its setup direction

Figure 5.1: 3D object and its setup direction



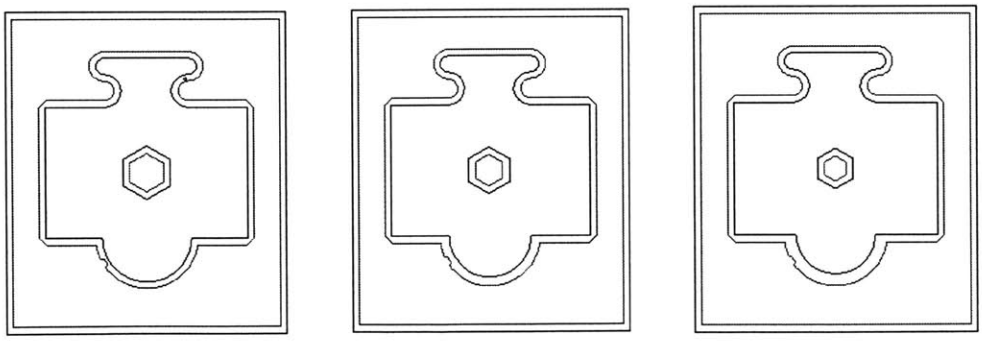
a) Object and its slab b) approximation contours

Figure 5.2: Object with slab and approximation contours

The setup direction of the object is determined [LaxmiPrasad 98] and the setup direction is showed in Figure 5.1b.

Using the concept of adaptive slicing the slice planes are positioned in the object and the machinable slabs are formed. Figure 5.2a shows the object along with the uppermost slab. The $2\frac{1}{2}D$ approximation contours corresponding to the various slabs of the object is shown in Figure 5.2b.

The tool that is being considered is a simple *1/2*" uniform diameter milling cutter and the offset region of this tool is determined and the results for the first three slabs have been



a) Slab 1 b) Slab 2 c) Slab 3

Figure 5.3: Offset regions at the slabs

Slab Number	Delta Area (sq units)	Accessible Area (sq units)
1	54.928	54.928
2	54.56	53.8236
3	54.3447	52.9497

Table 5.1: Accessibility area of single tool

shown in Figure 5.3. The distance between the intersection contour and the feasible offset region contour at portions corresponding to the half-frustum is increasing because a vertical tool in a 3-axis milling environment cannot *reach under* the tapering frustum. The area that could be reached by the tool in the various slabs is determined by re-offsetting the feasible offset region *outwards* by the radius of the tool. The numerical value of the *delta area* that needs to be removed from the slabs is shown in the first column of the Table 5.1. The *accessible area* of the tool is shown in the second column of Table 5.1. It is clearly observed that the accessible areas are monotonically decreasing, thus illustrating the vertical shadow-principle in 3-axis machining. The toolpath is generated for the accessible areas of the slab and Figure 5.4 shows the Zig-zag toolpath of the tool at the first slab.

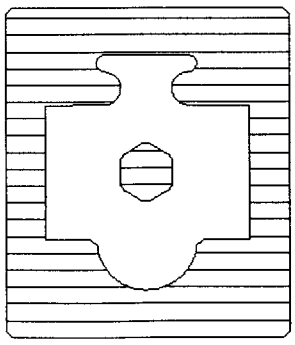
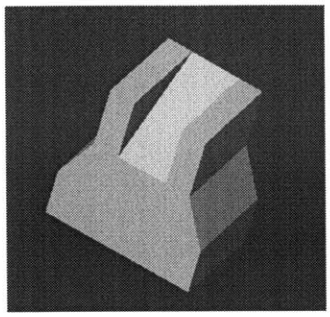
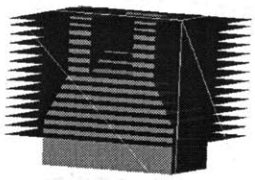


Figure 5.4: Toolpath at first slab





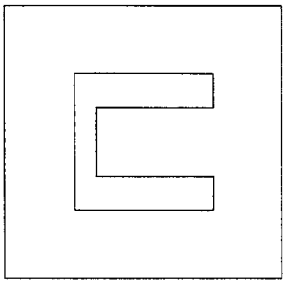
a) 3D object b) tool-assembly model c) object with sliceplanes

Figure 5.5: Model of object, tool-assembly and Layered object

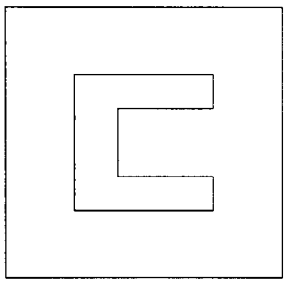
5.2 Accessible area determination for a tool-assembly

We extend the previous example to determine the accessibility of a tool-assembly in an object. Figure 5.4a shows the model of the 3D object that is to be machined using the tool-assembly shown in Figure 5.5b. The object is stratified and the object along with its sliceplanes is shown in Figure 5.5c. The object is sliced at the locations of the sliceplanes and using $2\frac{1}{2}D$ machinable slabs are formed. The 2D contours corresponding to the slabs are shown in Figures 5.6a-h.

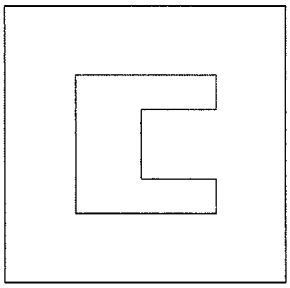
The tool-assembly has two constituent cylinders. The feasible offset regions of cylinder-1 and cylinder-2 along with the contour corresponding to the slab are shown in Figures 5.7a-h and Figures 5.8a-h. The feasible offset region of the tool-assembly is obtained by merging the results of the two cylinders and the results are shown in Figures 5.9a-h. It is observed that the effect of the larger cylinder is felt from the 3^{rd} *slab* because the larger cylinder enters the embedded design when the tool tip is tracing the 3^{rd} *slab*. Figures 5.10a-h show the center-line toolpaths of the tool-assembly at the slabs to machine the part.



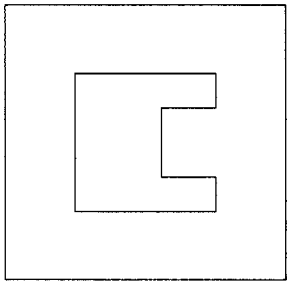
a) slab 1



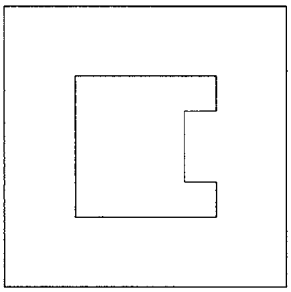
b) slab 2



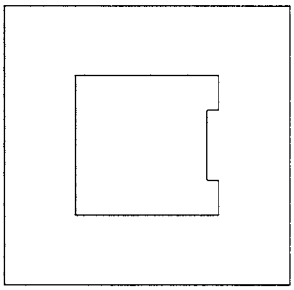
c) slab 3



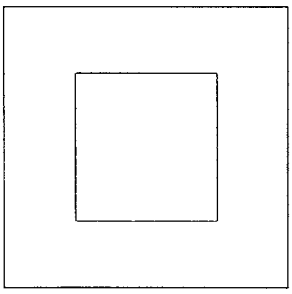
d) slab 4



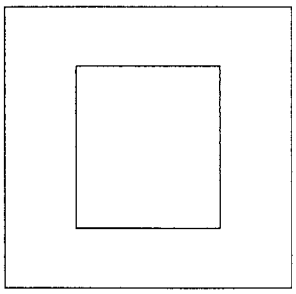
e) slab 5



f) slab 6

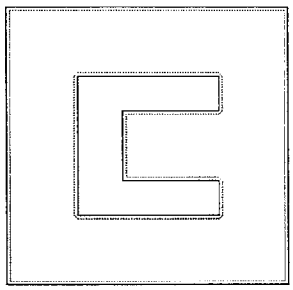


g) slab 7

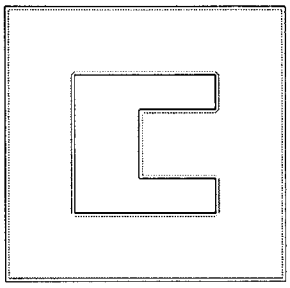


h) slab 8

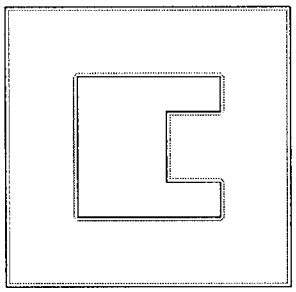
Figure 5.6: 2D approximation contours at slabs



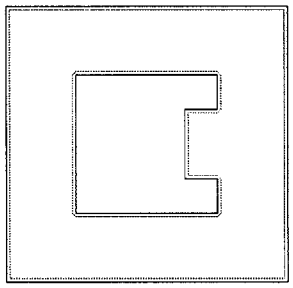
a) slab 1



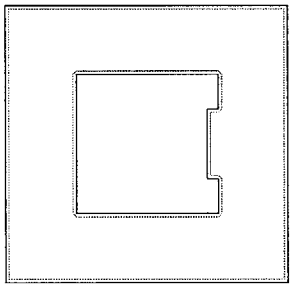
b) slab 2



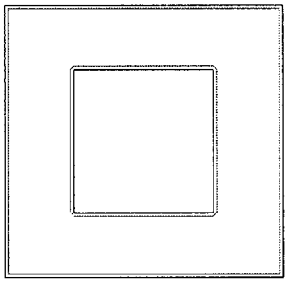
c) slab 3



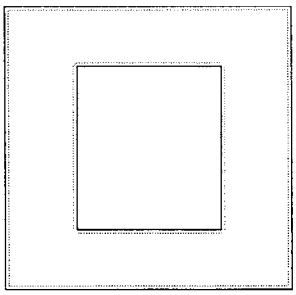
d) slab 4



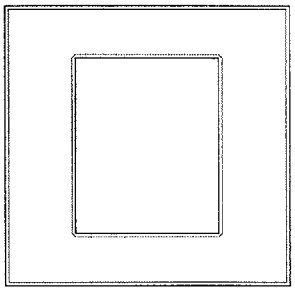
e) slab 5



f) slab 6

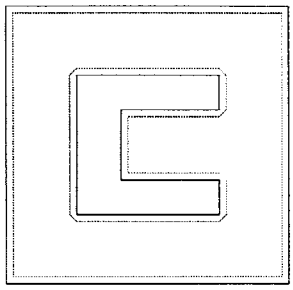


g) slab 7

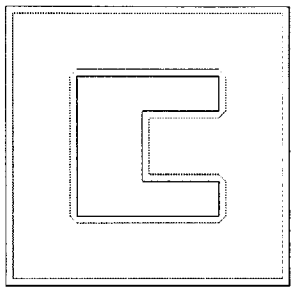


h) slab 8

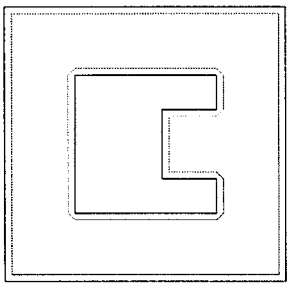
Figure 5.7: Offset regions cylinder-1 at different slabs



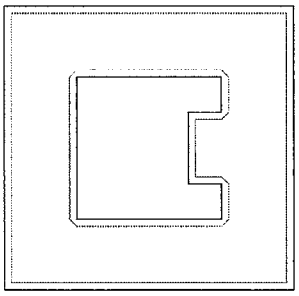
a) slab 1



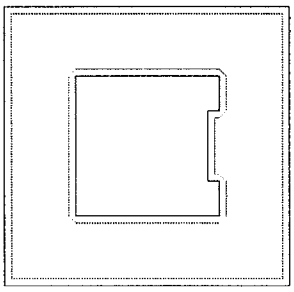
b) slab 2



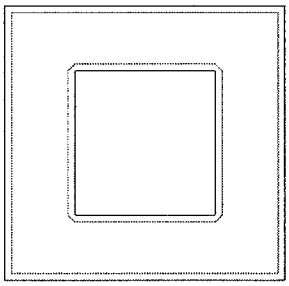
c) slab 3



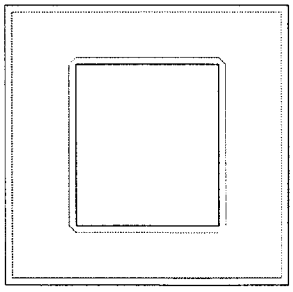
d) slab 4



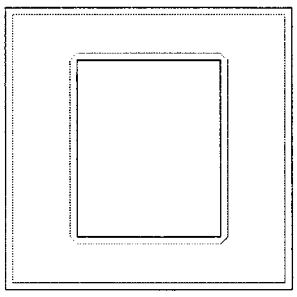
e) slab 5



f) slab 6

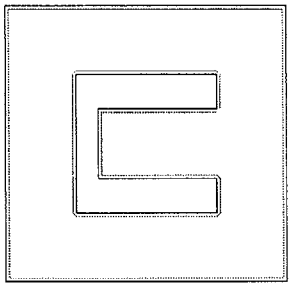


g) slab 7

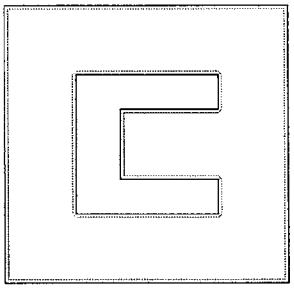


h) slab 8

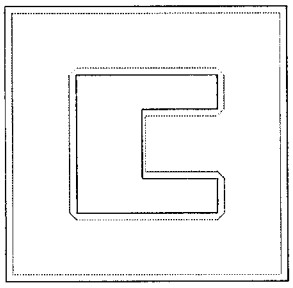
Figure 5.8: Offset regions cylinder-2 at different slabs



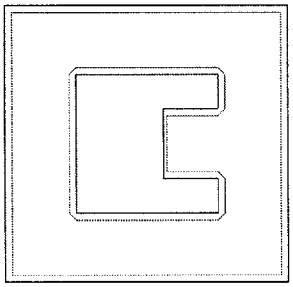
a) slab 1



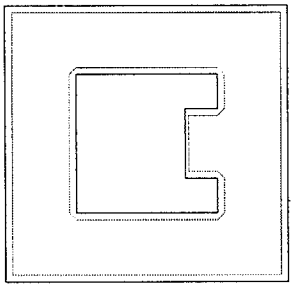
b) slab 2



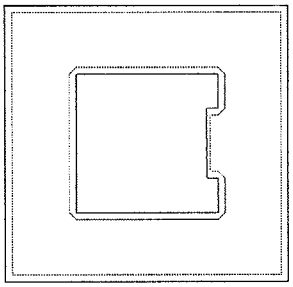
c) slab 3



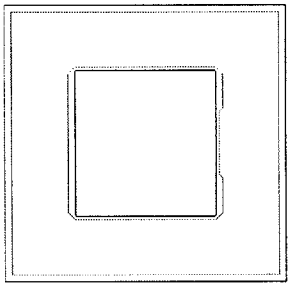
d) slab 4



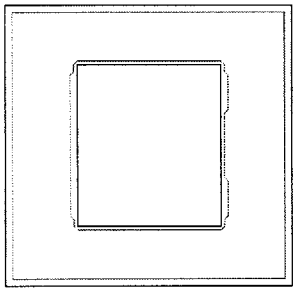
e) slab 5



f) slab 6

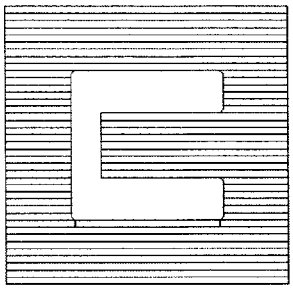


g) slab 7

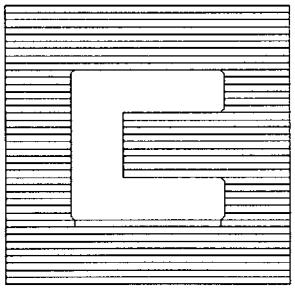


h) slab 8

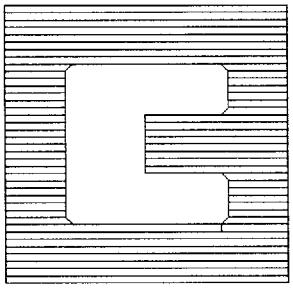
Figure 5.9: Feasible Offset regions of tool-assembly



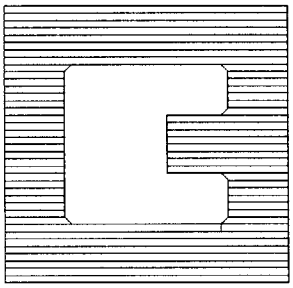
a) slab 1



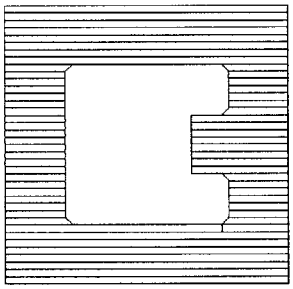
b) slab 2



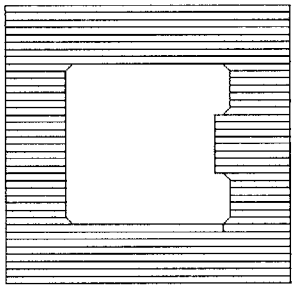
c) slab 3



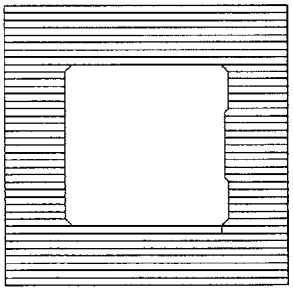
d) slab 4



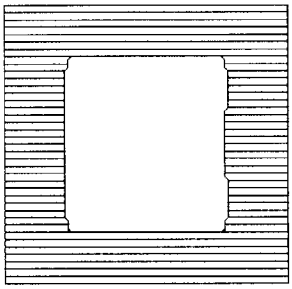
e) slab 5



f) slab 6

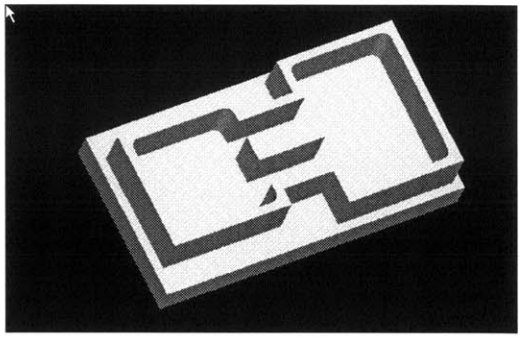


g) slab 7

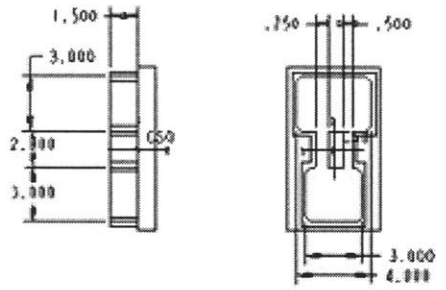


h) slab 8

Figure 5.10: Center-line toolpaths of tool-assembly



a) 3D object



b) Part drawing

Figure 5.11: 3D object and part drawing

5.3 Optimal tool selection

To illustrate the optimal tool selection algorithm, consider a part and its drawing shown in Figure 5.11. The objective function is to minimize the total machining time and ensure the maximum possible tool life. To illustrate the concept behind the optimal tool selection, The object is stratified into three slabs and a sequence of simple tool-assemblies is used to machine the object. The tool-assemblies that are considered in the sequence are

1. toolA - *1"* uniform diameter cutter.
2. toolB - *3/4"* uniform diameter cutter.
3. toolC - *1/2"* uniform diameter cutter.

Slab Number	ToolA (sq units)	Tool B (sq units)	Tool C (sq units)
1	21.0	22.5	23.5
2	21.0	22.5	23.5
3	21.0	22.5	23.5

Table 5.2: Individual accessible area of tools of a sequence

The accessibility areas of all the tool-assemblies at the slabs are determined and are given in Table 5.2. The numerical values indicate the amount of area each tool-assembly can reach independently in every slab. Since the tools are of uniform diameter, it is obvious that the smallest diameter tool would be able to reach the maximum area in the slabs.

The pruning criterion in the optimal tool selection is validated by selecting two sequences, using the techniques described in Section 4.2. The sequences that are to be considered are:

1. Sequence 1: $toolA \rightarrow toolB \rightarrow toolC$

2. Sequence 2: $toolA \rightarrow toolC$, where *toolB* has been removed from the sequence

Analytical expressions for machining times are computed for the tool-sequences. Inorder to select a sequence that has the smallest machining time, a condition relating the delta-area, MRR and tool-change time will be obtained. We will finally conclude that the condition for selecting a sequence is same as the pruning criterion explained in Section 4.3.

Let us consider the sequence-1 where the tools are ordered as $toolA \rightarrow toolB \rightarrow toolC$ The numerical values of the actual area machined by each tool as a part of the sequence is given in Table 5.3. Since the geometry of the object and the tools are very simple, the

Slab Number	ToolA (sq units)	Tool B (sq units)	Tool C (sq units)
1	21.0	1.5	1.0
2	21.0	1.5	1.0
3	21.0	1.5	1.0

Sequence-1

Slab Number	ToolA (sq units)	Tool C (sq units)
1	21.0	2.5
2	21.0	2.5
3	21.0	2.5

Sequence-2

Table 5.3: Areas machined by tools as a part of sequence

accessible areas cut in the lower slices matches the numerical difference of the areas in the respective slabs. In order to compute the time for machining the pocket, let us denote the MRR of a 1" tool as R and the tool change time as T. The total machining time of the sequence $toolA \rightarrow toolB \rightarrow toolC$ is $(63/R + (T + 4.5/(0.75R)) + (T + 3/(0.5R)))$ units *i.e.* $2T + 75/R$ units. Similarly for sequence-2, where the tools are ordered as $toolA \rightarrow toolC$ and the numerical values of the actual area machined by each tool as a part of the sequence is given in Table 5.3. The total machining time of the sequence $toolA \rightarrow toolC$ is $(63/R + (T + (7.5)/(0.5R)))$ units *i.e.* $T + 78/R$ units.

It can be observed that the choice of selecting a tool sequence depends on value of the tool-change time with respect to the MRR. Therefore we can conclude that, if $T > 3/R$ then *sequence-2* is preferred else *sequence-1* is preferred. The above result validates the criterion for pruning, which is based on the magnitude of tool change time (T) and the time taken to machine the excess *delta area* by the following tool $(4.5)/(0.5R) - (4.5)/(0.75R)$ units *i.e.* $3/R$ units.

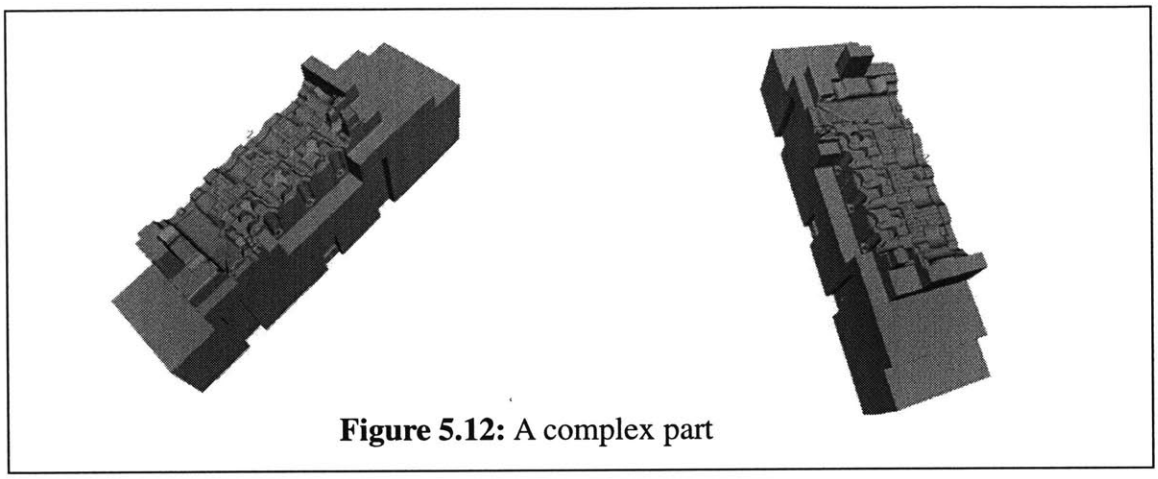


Figure 5.12: A complex part

5.4 Robustness of the implementation

A robustness of the layering, offsetting and the polygon-boolean routines were considerably improved by the grid based approach [Appendix A]. Figure 5.12 shows a very complex part given by Daimler-Benz, Germany to M.I.T. and the part had around 250,000 triangles to represent its very fine geometric details. The results of the layering have been shown in Figure 5.13.

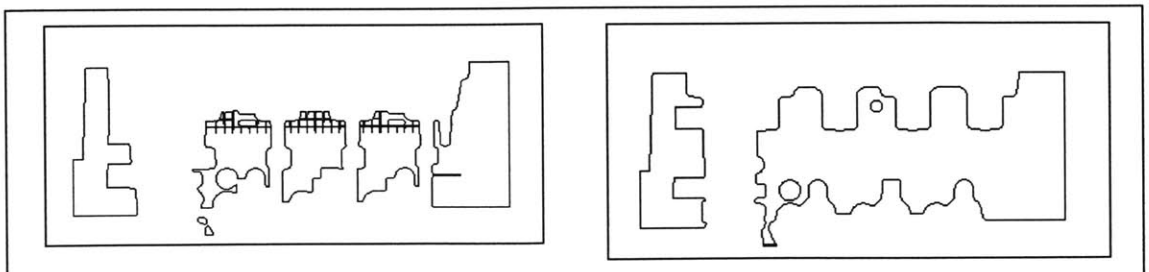


Figure 5.13: Selected slices of the complex object

Chapter 6: Conclusions and Future Work

The work presented in this thesis is a part of a larger effort to make machining a rapid prototyping process. In summary, we present a new technology, "Art-to-Part manufacturing", for CAD/CAM integration. The key idea here is the use of accessibility considerations as the driving constraint in path generation. This is fundamentally different from the feature-based approach: Instead of identifying and generating tool paths for each specialized feature, we developed techniques to generate tool paths for an arbitrary 2D contour. The technique is applied repeatedly to machine the entire workpiece. This approach is an extension of concepts developed by numerous other researchers in the areas of CAM, surface machining and robotics.

We have presented a set of algorithms to select a tool sequence and generate tool paths for 3-axis roughing. These are demanding problems in the automotive and aerospace industries because as quality demands rise, hog-milling is slowly regaining prominence as a near-net shape manufacturing process. Improvements in cutting productivity can have significant economic impact in such situations. Although the feature-based approach remains intellectually appealing, we see the Art-to-Part approach as the one that is potentially more comprehensive, and one which can overcome some of the limitations of feature-based machining.

6.1 Model updation for different setup

All the discussions in the thesis were with respect to one setup direction only. A machining operation could have several setup directions for the part, so it is necessary to

transform the information about the machined delta volume to the new setup. The solid model of the machined delta-volume can be made up by performing a UNION operation of the $2\frac{1}{2}D$ accessible regions of all the slabs by all the tool-assemblies used in that setup. A solid model corresponding to the remaining stock after the machining operation in this setup can be obtained by performing a SUBTRACTION operation of the machined delta-volume from the initial stock model. The updated stock model will be the initial stock for the next setup and both the embedded design and the new stock will be rotated to the new setup orientation. The sliceplanes will be positioned w.r.t. the embedded design in the new orientation as described in Section 2.2. The Ordered tree contour for the $2\frac{1}{2}D$ slab can be obtained by a SUBTRACTION of the Ordered Tree contours corresponding to the embedded design from that of the new stock. Once the $2\frac{1}{2}D$ slabs have been obtained, the procedure to determine the accessibility and optimal tool sequence is same as that described in Chapters 3 and 4.

6.2 Improvements in tool-path generation

The region to be machined in a slab is bounded by closed contours. The toolpath is generated such that the tool does not overshoot the contour boundary and this would result in undesirable corners in the machinable region. The undesirable corners could be avoided by recognizing the contour boundaries that could be overshot without colliding with the embedded design. Section 6.2.1 describes a method of reclassifying boundaries of the contour to generate *better* toolpaths.

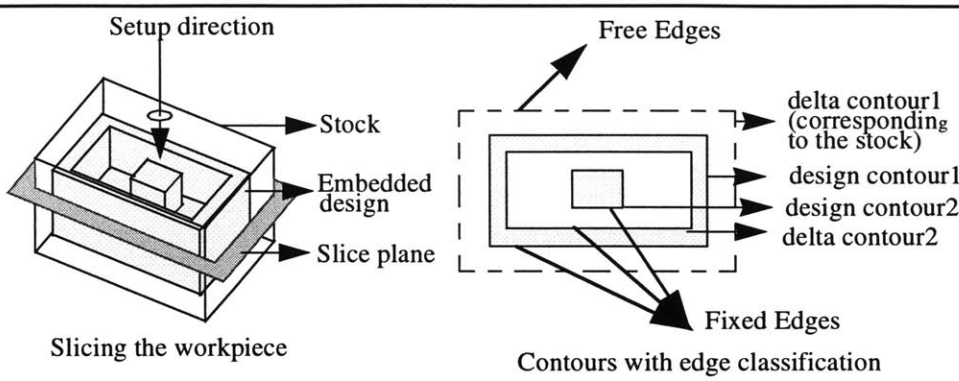
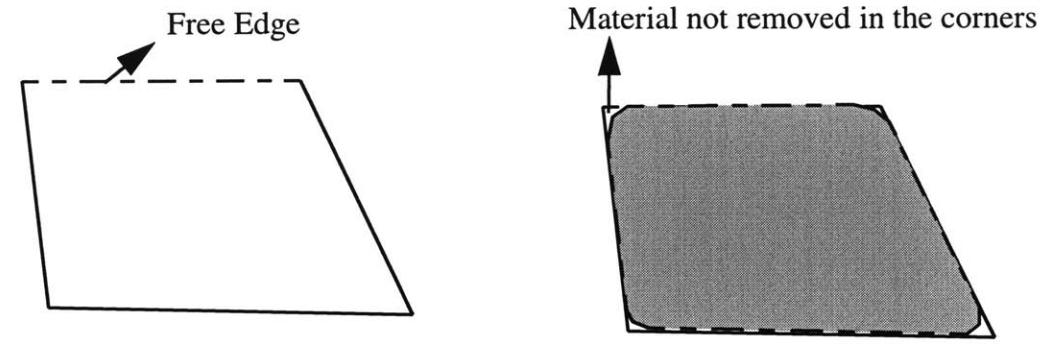


Figure 6.1: Contours and edge classification

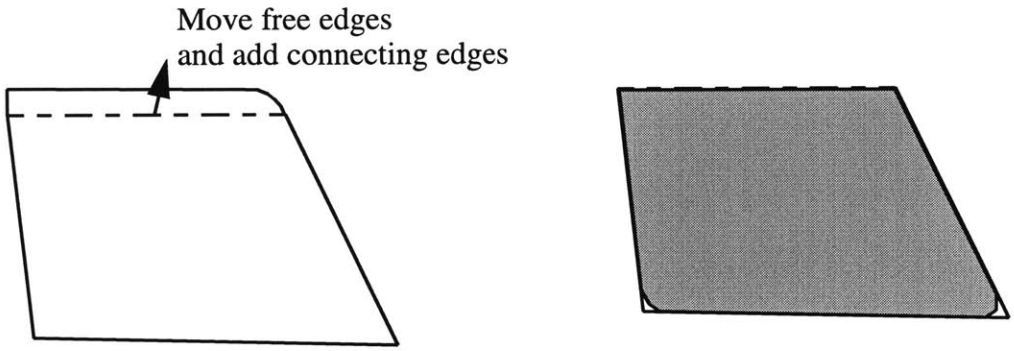
6.2.1 Reclassification of contour boundaries

The toolpath generation given the Ordered tree contour should take into account the nature of the boundaries of the contour also. The boundaries need to be classified into categories based on whether a tool can overshoot it without colliding with the embedded design. We can therefore classify the boundary edges as either *fixed* or *free*, where the fixed edges cannot be overshot by the tool, while the *free* edges could be overshot by the tool. Figure 6.1 shows an example of a intersection contour whose boundary edges have been classified as either *fixed* or *free* edges.

If all edges were treated the same then the tool will leave undesirable corners, which could be avoided by the suggested approach. This is of far more importance when the object is machined in several setups. Since the tool can overshoot a *free* boundary, undesirable corners can be avoided by shifting the *free* edges *outwards* by the radius of the cutter. The contour needs to be closed by adding new edges and the type of edge that is added will depend on the edge-edge combination that should be connected (i.e. *free - free* edge or *fixed - free* edge) and the nature of the common vertex (i.e. concave vertex or convex vertex).



(a) contour and area removed by a tool



(b) Modified contour and area removed by a tool

Figure 6.2: Advantages of Edge offsetting

Figure 6.2a shows a contour with one *free* edge and the area machined by the tool in the contour. Figure 6.2b shows the same contour with the *free* edge offseted outwards by the radius of the tool and the area machined by the tool in the contour. It is clearly seen that the *artificial round edges* created in the machined contour can be eliminated as much as possible by this approach.

6.3 Finishing - issues and suggested approach

After the component has been reduced to near net shape using global roughing, it has to be finished to achieve the required surface and form accuracy. Finishing is usually per-

formed with small, accurate tools, and at a light cutting rate. A face-based finishing strategy is proposed in this research. The central idea is to target each face for finishing independently without necessarily grouping them into features. This strategy is being pursued for two reasons: firstly, it bypasses the problem of feature recognition, and secondly, it enables the system to handle shapes that can not be expressed in terms of classical features. The subsequent sections discuss the major milling modes and some of the aspects in finish machining plane and cylindrical faces.

6.3.1 Machining modes in milling

Milling tools can be used in four machining modes: surface milling, peripheral milling, face milling and shape milling. In surface milling, typically performed with ball-nosed or bull-nosed end mill, the profile of the work piece is generated entirely by the path of the cutting tool. In other words, the shape of the tool is not used to impart shape to the component. In peripheral milling, however, the side of the tool is used to impart flatness to the workpiece. Peripheral milling is ordinarily used to machine either flat or cylindrical. In face milling, the bottom of the tool is used to impart flatness to the surface. This technique is used only for flat surfaces. Finally, shape milling is the most specialized form of milling, and is used to impart shapes like chamfers, tapers and fillets being machined. Because of this specialized nature of milling, different types of surfaces must be treated appropriately to maximize performance.

6.3.2 Finishing Plane faces

When surface finish requirements are reasonably stringent, plane faces must typically be machined by face or end milling. Since smooth plane faces must only be side milled or

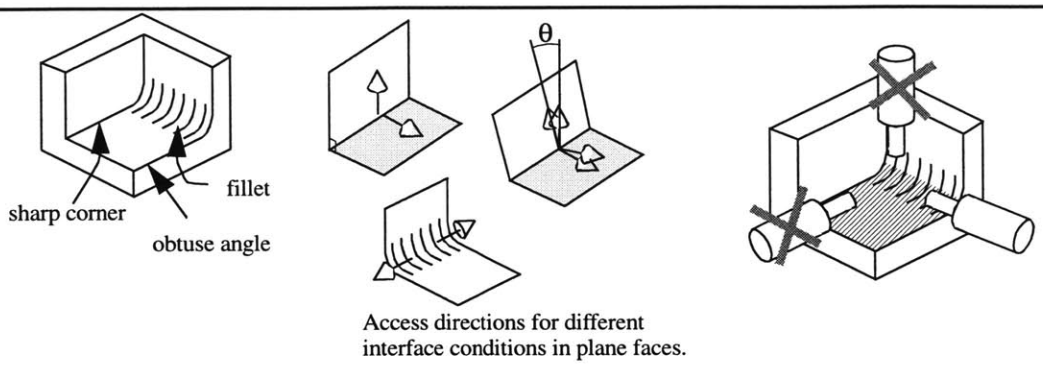


Figure 6.3: Edge conditions in Finishing

face milled, the access-direction for machining needs to be either parallel or perpendicular to the face. Furthermore, the entire face must be machined from the same direction, in the same setup since discontinuous tool paths leave dwell marks and seam lines. Together, these criteria restrict the ways in which flat faces can be finished.

If the surface finish requirements on a nominally plane face are *not* stringent, then it can be generated by profile milling with a ball-nosed end mill. Profile milling is advantageous because it offers a larger range of accessibility. This freedom can be exploited to reduce the number of setups in machining. Rough surfaces can therefore be treated as curved surfaces for the purposes of tool path generation — however, we do not discuss surface machining in this paper. For more information, the reader may refer to [Lee 95].

Using edge conditions: The first criterion that needs to be considered in determining which direction a face can be accessed from is the condition at the edge between the face and its neighbors. This edge condition can be an acute angle, an obtuse angle or a fillet, and can limit the possible perpendicular/parallel access directions as shown in Figure 6.3. The least restrictive edge conditions are obtuse angles. For example, the top surface of an exposed boss can be accessed from every parallel and perpendicular direction. Sharp

edges, however, dictate that the tool approach direction be perpendicular to the edge. Fillets can only be accessed along the edge. Together, these conditions restrict the number of access directions in to machine the component.

Picking a tool: The face to be machined will nearly always neighbor a portion of the delta volume that was roughed. A tool diameter would already have been picked while generating the path for roughing. By querying neighboring voxels to the face as to which tool they were roughed with, it is possible to pick the size of the tool that can be used for finishing.

Harnessing commonalities: One of the motivations for the feature based approach is that by bunching groups of faces into features, it is possible, for example, to pick a single tool to perform the entire cut. Since the face based approach is fundamentally more "atomic", it is necessary to explicitly identify and exploit commonalities in the cutting plan. Schemes to perform this optimization are currently being developed in this research. One scheme is to consider neighboring faces with the same access direction, and to attempt to pick the same tool as the neighbor. This strategy can also be used in shape features like rounded edges and fillets. If a choice exists then an attempt will be made to pick a face finishing tool that can also impart the appropriate fillet or corner radius at the edge. This grouping is intended merely to optimize the process and reduce tool changes; explicit recognition of features will not be performed.

6.3.3 Finishing cylindrical faces and holes

Faces in which the curvature vanishes everywhere along exactly one direction can be machined by peripheral milling. Such faces are referred to as extruded surfaces, and we will refer to the zero-curvature direction as the principal direction. These shapes can only

be accessed for peripheral milling from either side of the principal direction. The principal direction can usually be ascertained from the BRep file.

The special case of drilled holes: Drilled holes, unlike other ruled surfaces like milled holes and profiles, are very special entities; their entire shape, including the adjacent bottom face, is imparted by the shape of the drill. The drill performs most of the roughing and finishing, although an additional reaming operation may also be required. For this reason, drilled holes require special consideration similar to traditional feature based analysis. For every cylindrical surface, therefore, we will first investigate whether: 1) a characteristic shape like a conical bottom or counterbore can be located along the principal axis; 2) the diameter of the cylinder corresponds to standard ream or drill size; 3) depth of the feature corresponds to an available tool. If these criteria are met, then that cylindrical face and all associated entities will be marked for drilling. Furthermore, the convex hull of the drilling operation will be appropriately tagged or filled in the model so that no attempt is made to rough it. Other shape elements, like taps, will also be associated with the hole and appropriate tools will be selected. In this aspect, our approach resembles the technique developed by Regli [Regli 95].

6.4 Conclusions

This summarizes our technique for selecting the optimal tool sequence and generating 3-axis roughing toolpaths to machine a object fixtured in a particular setup direction. The concepts described here have been tested experimentally, as illustrated by the screen dumps in the figures. The construction of the entire system for a comprehensive Art-to-Part machining is underway.

Appendix A Topology preserving snap-fits using GRID based Integer computation

Floating point operations and the instabilities associated with them have been dealt with by a number of researchers over the past decade. The problem with floating point computations leads to instability of application, since the algorithms are valid and stable for perfect geometric objects only. The error accumulation due to successive floating-point operation introduces C^0 discontinuity in the contours, by opening up the contour. The problems associated with this are two-fold: Firstly the concept of area enclosed by a contour is valid only for a perfectly closed contour. Secondly, additional computation on these distorted edges results in improper intersection computation and classification of the edges [Leonov 95] causing very small length edges to be introduced in the contour leading to instability in further computations. Below we suggest some methods to avoid the errors and perform very stable geometric operations.

A.1 Grid based snap Approximation

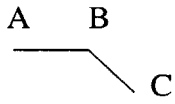


Figure A.1: Problem using TINY

The work done by previous researchers [Patrikalakis 96, Maekawa 98] have discredited the use of a TINY parameter when comparing two floating point numbers because it can introduce discrepancy in the geometry as illustrated below in Figure A.1. Figure A.1 shows 3 points for which the TINY based comparison is used, A and B are close within the accuracy of TINY and similarly B and C are also close but this does not necessarily imply A and C are close. This is one of the major flaws of floating point computation to a very basic equality checks itself. Inorder to bypass the floating point method of checking for equality of vertices, the geometric entity in $\{R\}$ is mapped to

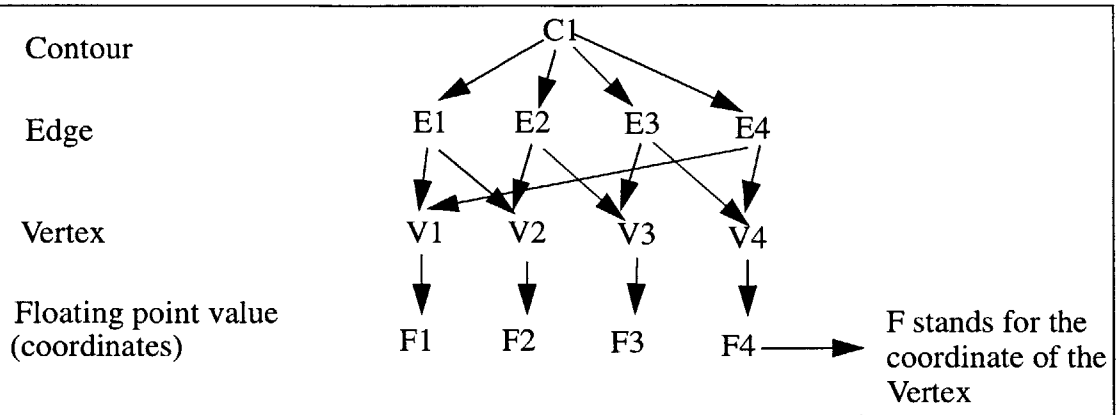
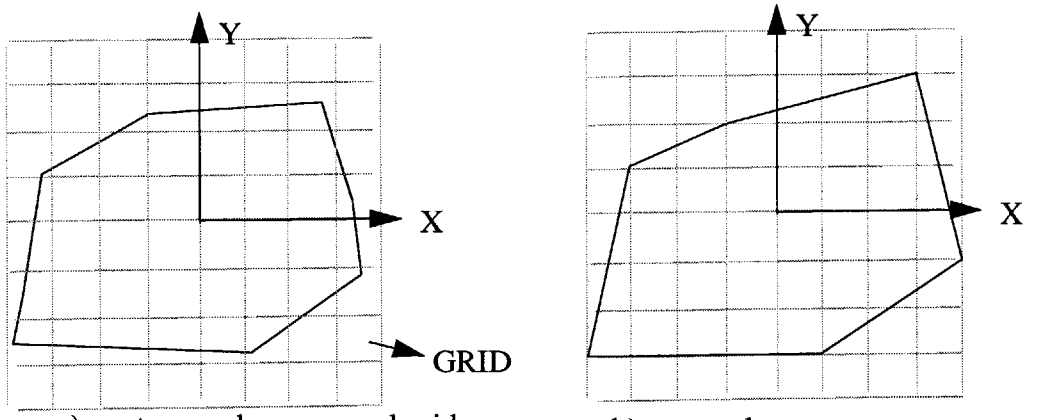


Figure A.2: Internal data-structure representation

a very fine grid of defined accuracy i.e. $\{G\}$. In the mapping the points are snapped to the nearest grid point obtained by rounding the vertex. Every grid point has an effect on a square region centered on it and spanning ± 0.5 times the grid spacing in both the dimensions. The snapping has a two-fold advantage: Firstly, the contour is always closed and Secondly, the minimum length of an edge in the contour is atleast equal to the grid spacing.

Any geometrical entity, i.e. a contour, would be stored in a data structure shown in Figure A.2, which would enable easy consistency checks and manipulation of the contour. The advantage of this representation is that geometry definition and position definition have been separated and it is therefore easier to redefine positions and then check for consistency of the tree as explained below. The vertices which form the leaf nodes in this geometric representation can be snapped to the nearest grid-point by altering its floating point values and then a consistency check for self-intersection can be performed and the contour data-structure can be altered accordingly. Normally, operation involving the snapping of the vertices would not cause very extensive self-intersection problems since it a very localized operation, which can however be eliminated using loop-elimination algorithms.



a) contour and superposed grid b) snapped contour

Figure A.3: A superposed grid and snapped contour

Figure A.3 shows an example of a contour, which has been super-posed on a very coarse grid for illustration purposes and its snapped contour. Once the contour has been snapped, the data-structure needs to be modified to take into account the following:

1. if there were any zero length edges that were removed
2. if several small edges lying in the same straight line were merged to from longer edges.

The contour obtained after snapping will have the advantages of a integer representation since the vertex can be referred to by the x and y coordinate of the grid to which it is mapped. The equality checks for the vertex and the parallelism checks for the edge can now be grid-based. The robustness of the layering, offsetting and polygon-boolean operations was considerably enhanced when the grid based computation was implemented.

References

Anderson 90

Anderson, D. C. and Chang, T. C. "Geometric Reasoning in feature-based design and manufacturing," Computers and Graphics, 14(2) 225-235. 1990.

Arbab 82

Arbab, F. "Requirements and Architecture of CAM-Oriented CAD Systems for Design and Manufacture of Mechanical Parts," Ph.D. Thesis, University of California, Los Angeles. 1982.

Chen 92

Chen. L-L and Woo, T. C. "Computational geometry on the sphere with application to automated machining" *Transactions of ASME*, Vol. 114: 288-295.

Choi 89

Choi, B. K. and Jun, C. S. "Ball end-cutter interference avoidance in NC machining of sculptured surfaces.

Chou 89

Chou, J. J. *Numerical control toolpath generation for for regions bounded by freeform curves and surfaces*, Ph. D Thesis, University of Utah. 1989.

Cutkosky 88

Cutkosky, M., Tenenbaum, M. and Miller, D. "Features in Process-based Design", *Proceedings of the ASME Computers in Engineering Conference*, San Francisco. 1988.

Cutkosky 90

Cutkoskt, M. R. and Tenenbaum, J. M. "A methodology and computational framework for concurrent product and process design" *Mech. Mach. Theory* **25**(3) pp. 365-381. 1990.

Dolenc 93

Dolenc, A and Makela, I.: "Slicing Procedures for layered manufacturing techniques", *Computer Aided Design*, Vol. 26, No. 2, 1994, pp. 119-126.

Dong 88

Dong, X. and Wozny, M. "FRAFES: A frame based feature extraction system" *Proceedings of the Int. Conf. on Computer Integrated Manufacturing,* RPI, USA. 1988.

Elber 94

Elber, G. and Cohen, E. "Toolpath generation for freeform surface models" *Computer Aided Design*, **26**(6): 490-496. 1994.

Faux 81

Faux, I. D. and Pratt, M. J., *Computational Geometry for Design and Manufacture*, Ellis Horwood, Chichester, England. 1978.

Finger 90

Finger, S. and Safier, S. "Representing and recognizing features in Mechanical Designs" *Proc. Second International Conference on Design Theory and Methodology*, DTM '90. 1990.

Foley 95

Foley, J. et al, *Computer Graphics: Principals and Practice,* Addison-Wesley, 1995.

Gadh 92

Gadh, R. and Prinz, F. "Recognition of Geometric Features Using Differential Depth Filters," *Computer Aided Design*, **24**(11): 583-598. 1992.

Gindy 89

Gindy, N. N. Z. "A hierarchical structure for form features" *Int J. of Prod. Research.*, **27**(12):2089-2103, 1989.

Green 96

Green R. E, Oberg E, Jones F. D, Horton H.L and Ryhhel H.H: "Machinery's Handbook - twenty-fifth edition", Industrial press Inc

Gupta 95

Gupta, S. K. Automated Manufacturability Analysis of Machined Parts, Ph. D. Thesis 95-3, Institute for Systems research, University of Maryland. 1994.

Gurbuz 95

Gurbuz, A and Zeid, I 'Offsetting operations via closed ball approximation,' *Comput.-Aided Des.*Vol 27 No 11 pp 805-810, 1995.

Hayes 89

Hayes, C. C. "Automating Process Planning; Using Feature Interactions to Guide Search," Journal of Manufacturing Systems, **8**:1-15. 1989.

Held 93a

Held, M 'A fast incremental algorithm for computing the voronoi diagram of a planar shape,' *Computing with virtual worlds* (CGI'93) Springer-Verlag, pp 318-329, 1993.

Held 93b

Held, M *et al* 'Pocket machining based on contour-parallel tool paths generated by means of proximity maps,' *Comput.-Aided Des.* Vol 26 No 3, pp 189-203. 1994.

Held 1991

Held, M. *On the computational geometry of pocket milling.* Lexture Notes in Computer Science, Springer-Verlag, London. 1991.

Henderson 96

Henderson, M. R., Chell, A. R. and Hubele, N. F. "Feature based manufacturing evaluation using statistical process control" *Proceedings of the 1996 NSF Design And Manufacturing Grantees Conference.* 1996.

Hoschek 85

Hoschek, J 'Offset curves in the plane,' *Comput.-Aided Des.* Vol 17 No 2 pp 77-82, 1985.

Hummel 86

Hummel, K. E. and Brooks, S. L. "Symbolic Representation of Manufacturing Features for an Automated Process Planning System" ASME WAM. 1987.

Jara-Almonte 92

Kirschman,C. F., and Jara-Almonte, C. C., "A Parallel Slicing Algorithm for Solid Freeform Fabrication Processes", Proceedings of the 1992 Solid Freeform Fabrication Symposium, Austin, TX, August 3-5, 1992.

Jerrard 89

Jerrard, R. B., Hussaini, S. Z., Drysdale, R. L. and Schaudt, B. "Approximate methods for simulation and verification of numerically controlled machining programs" *Visual Computer*, **5**(6). 1989.

Jerrard 91

Jerrard, R. B., Angleton, J. M. and Drysdale, R. L "Sculptured surface toolpath generation with global interference checking" Pres 1991 Des. Productivity Int. Conf., Honolulu, Hawaii. 1991.

Joshi 88

Joshi, S. and Chang, T. C. "Graph-based heuristics for recognizing machining features from a 3D solid model", *Computer Aided-Design*, **20**(2). 1988.

Kim 90

Kim, Y. S. *Convex decomposition and solid geometric modeling*, Ph. D. Dissertation, Stanford University. 1990.

Kramer 88

Kramer, T. R. "Process Planning for Milling Machines from Feature Based Design" *Proceedings of Manufacturing International* '88, pp178-189, ASME. 1988.

Krypianou 80

Krypianou, NL. K. Shape Classification in Computer Aided Design, Ph. D. Thesis, Computer Laboratory, University of Cambridge, U. K. 1980.

Kuragano 88

Kuragano, T., Sasaki, N. and Kikuchi, A. "the FRESDAM System for designing and manufacturing freeform objects" *USA-Japan Cross Bridge. Flexible Automation* edited by R. Martin II, 931-938. 1988.

Laxmiprasad 97a

Laxmiprasad, P. and Sarma, S. "5-axis pockets: definition and tool-path algorithms" Working paper, 1997.

LaxmiPrasad 98

"Automatic toolpath generation for multi-axis machining" M. S. in Mechanical Engineering, Massachesetts Institute of Technology, 1998.

Lee 92

Lee, Y. S., Choi, B. K. and Chang, T. C. "Cut distribution and cutter selection for sculptured surface cavity machining" Int. J. of Prod. Res. 30(6):1447-1470.

Lee 94

Lee, K., Kim, T. J., Hong, S. E. "Generation of toolpath with selection of proper tool for rough cutting process," *Computer Aided Design*, **26**(11), pp 822-831. 1994.

Lee 95

Lee, Y.S., and Chang, T.C., "Two-Phase Approach to Global Tool Interference Avoidance in 5-axis Machining", *Computer Aided Design*, Vol. 27, No. 10, 1995, pp. 715-729.

Lee 96

Lee, Y.S., and Chang, T.C., "Automatic Cutter Selection For 5-axis Sculptured Surface Machining," *International Journal of Production Research*, Vol. 34, No. 4, 1996, pp. 977-998.

Leonov 95

Web page regerence: "http://woland.afti.nsu.ru/~leonov/clipdoc.html"

Loney 87

Loney, G and Ozsoy, T. "NC machining of freeform surfaces" *Computer Aided Design* **19**(2):85-90. 1987.

Lozano-Perez 83

Lozano-Perez, T. "Spatial planning: A configuratin space approach" IEEE Transactions of Computers, C-**32**(2): 108-120. 1983.

Maekawa 98

S.L. Abrams, W.Cho, C.-Y. Hu, T. Maekawa, N.M. Patrikalakis, E. C. Sherbrooke and X. Ye "Efficient and reliable methods for rounded interval arithmetic" *Computer Aided Design*, 1998 to appear.

Nau 86

Nau, D. S. and Gray, M. "SIPS: An Approach of Heirarchical Knowledge Clustering in Process Planning" ASME WAM. 1986.

Nau 92

Nau, D. S., Zhang, G. and Gupta, S. K. "Generation and evaluation of alternative operation sequences" In A. R. Thangaraj, A. Bagchi, A. Ajanappa and D. K. Anand, editors, *Quality Assurance through the Integration of Manufacturing Processes and Systems*, PED-vol. 56, pp 93-108. 1992.

Oetjens 87

Oetjens, T. J. "Automotive CAD/CAM die: the sculptured surface" Proceeings of Autofact '87.

Oliver 86

Oliver, J. H. "Graphic verification of NC milling programs for sculptured surface parts" Ph. D. Thesis, Michigan State University. 1986.

Patrikalakis 96

C.-Y. Hu, T.Maekawa, E. C. Sherbrooke and N. M. Patrikalakis. "Robust interval algorithm for surface intersections" *Computer Aided Design*, 22(10): 645-654

Regli 95

Regli, W. *Geometric algorithms for the recognition of features from solid models*, Ph. D. Dissertation, University of Maryland.

Sakurai 90

Sakurai, J. and Gossard, D. "Recognizing shape features in solid models" *IEEE*

Computer Graphics and Applications, September, 1990.

Sarma 96

Sarma, S. E., Schofield, S., Stori, J, MacFarlane, J. and Wright, P. K. "Rapid Part Realization from Detail Design," *Computer Aided-Design* **28**(5):383-392. 1996.

Shah 88

Shah, J. J., and Rogers, M. "Functional requirement and conceptual design of the feature based modeling system" *Computer Aided Engineering Journal*, **7**(2):9-15. 1988.

Shah 94

Shah, J. J., Mantyla, M. and Nau, D. S. *Advances in feature based manufacturing,* Elsevier. 1994.

Spyridi 90

Spyridi, A. J. and Requicha, A. A. G. "Accessibility analysis for automatic inspection of parts" in R. A. Volz, ed., *Proc. IEEE International Conf. on Robotics and Automation*, pp 1284-1289, Cincinnati, Ohio. 1990.

Suh 90

Suh, Y S and Lee, K 'NC milling tool path generation for arbitrary pockets defined by sculptured surfaces,' *Comput.-Aided Des.* Vol 22 No 5 pp 273-284. 1990.

Tangelder 96

Tangelder, J., Vergeest, J. and Overmars, M. "Computation of voxel maps containing tool access directions for machining free-form shapes" *Proceedings of the ASME DETC/DFM*. 1996.

Tiller 84

Tiller, W and Hanson, E G 'Offsets of two-dimensional profiles,' *IEEE Computer graphics and applications* Vol 4 No 9, pp 36-46, 1994.

Tseng 91

Tseng, Y. J. and Joshi, S. "Determining feasible approach directions for machining Bezier curves and surfaces" Computer Aided Design, **23**(5):367-379. 1991.

Turner 88

Turner, G. P. and Anderson, D. C. "An object oriented approach to interactive feature based design for quick turn around manufacturing," *Proceedings of the ASME Computers in Engineering Conference*, San Francisco. 1988.

Vandenbrande 90

Vandenbrande, J. *Automatic recognition of machinable features in solid models*, Ph. D. Dissertation, University of Rochester, 1990.

Veeramani 97

Dharmaraj Veeramani and Yuh-Shying Gau, "Selection of an optimal set of cutting-tool sizes for 2-1/2D pocket machining", *Computer Aided Design,* **29**(12), pp 869-877, 1997.

Woo 82

Woo, T. C. "Feature Extraction by Volume Decomposition" *Proceedings of the Conf.*

on CAD/CAM Tech. in Mechanical Engineering, Cambridge. 1982.

Woo 94

Woo, T. "Visibility maps and spherical algorithms" *Computer Aided Design* **26**(1). 1994.

Wuerger 95

Wuerger, D. and Gadh, R. "Virtual prototyping of die design æ algorithmic, computational and practical considerations" proceedings of the *Computer Aided Concurrent Design Symposium*, ASME Design Engineering Technical Conferences, Boston. 1995.

Yap 87

Yap, C K 'An O(nlogn) algorithm for the voronoi diagram of a set of simple curve segments,' *Discrete Comput. Geom.* Vol 2 No 4 pp 365-393. 1987.

Yut 95

Yut, G. and Chang, T. C. " A Heuristic Grouping Algorithm for Fixture and Tool Setups" Engineering Design and Automation, 1(1):21-31. 1995.