Point-Based Mathematics For Computer-Aided Manufacture

by

Matthew Fisher

A thesis submitted to the
Faculty of Engineering of
The University of Birmingham
for the Degree of
DOCTOR OF PHILOSOPHY

School of Manufacturing and
Mechanical Engineering
The Univeristy of Birmingham
Edgbaston
Birmingham
B15 2TT
United Kingdom

October 2000

Synopsis

This thesis demonstrates the feasibility of machining high quality sculptured surfaces directly from a point-based definition. The work is founded on the strategy of using a sparse set of points to characterise shape although it is assumed that an appropriately dense definition can be generated by the use of some unspecified high quality interpolation algorithm. This is in contrast to the conventional CAD/CAM approach where explicit parametric expressions are used to describe the part.

The research is founded on the Inverse Offset Method (IOM) proposed by Kishinami; the algorithm is chosen because it possesses a number of desirable properties, most notably its versatility and robustness. The first fundamental contribution is an error analysis of the IOM that has not been published before, the analysis is dependent on the surface and cutter path point spacing, the tool radius and the local surface curvature. The accuracy of the error analysis is corroborated by the machining and measuring of a physical part. Furthermore it is established that the quality of the finished part produced by the IOM compares favourably with that produced by a commercial package for similar tolerances.

The principal research achievement is the optimisation of the IOM to exploit the coherence of data ordered into sections. This results in the IOM generating cutter paths in a time period comparable to that of the commercial package without a reduction in the quality of the finished part. The last contribution made in this thesis is a report on the issues concerning the machining of point definitions derived from multi-surfaces.

The work presented in this thesis offers an alternative strategy to the design and manufacture of free-form surfaces. The main benefits of adopting this strategy are gained because it removes the need to generate a parametric surface definition.

Acknowledgements

I would like to thank everyone at the Geometric Modelling Group for making my Postgraduate days highly enjoyable ones. In particular I would like to thank Professor A. A. Ball for his most excellent supervision. Thanks also go to Delcam and the Engineering and Physical Sciences Research Council for their financial support.

I would also like to thank my wife, Soose, for her endless love. Finally I would like to thank God for blessing me every day of my life.

Table of Contents

Chapter 1

Introduction

1.1 CAD/CAM

Manufactured products designed with free-form surfaces are becoming increasingly common in today's society for aesthetic as well as functional [Li *et al.* 1994]. From Bart Simpson styled alarm clocks to fashionably curved kettles, the eye pleasing shape of a product can be key to its success in the market [Kuragno 1992]. In any industry where performance is dependent on shape, e.g. the automobile industry [Jerard *et al.* 1989], free-form surfaces of good quality are essential. Another functional use of the free-form surface is in the ergonomic design of a product, e.g. the comfortable hand shaped grip of a razor handle [Baxter 1995]. Hence there is an obvious need to efficiently manufacture such surfaces. Two key stages in the manufacturing process are the defining of the part and the machining of the respective tooling. These stages come under the more general heading of Computer Aided Design and Manufacturing (CAD/CAM) which is a large subject area and covers a large number of disciplines. For a good general description see [Bedworth *et al.* 1991].

1.1.1 Defining a Part

In general a part is defined by a number of parametric patches that are joined with tangent plane continuity where appropriate. A parametric patch is a mapping of a rectangular region from 2-dimensional parametric space to 3-dimensional space [Faux and Pratt 1987], an example is given in Figure 1.1. The geometric control of these patches is important and hence a variety of different basis functions have arisen (the means by which parametric curves and surfaces are controlled). An interesting

selection of papers documenting the historical development of these methods written by the pioneers involved can be found in [Piegl 1993].

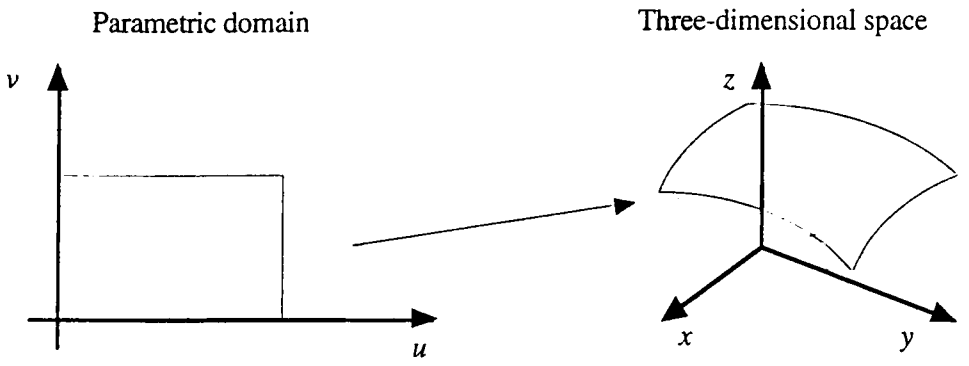


Figure 1.1: A parametric patch

A surface model consists of an assembly of patches, the patch boundaries of the definition of a shoe last are shown in Figure 1.2. General practice is to regard this assembly as the master model and all further operations are performed on it.

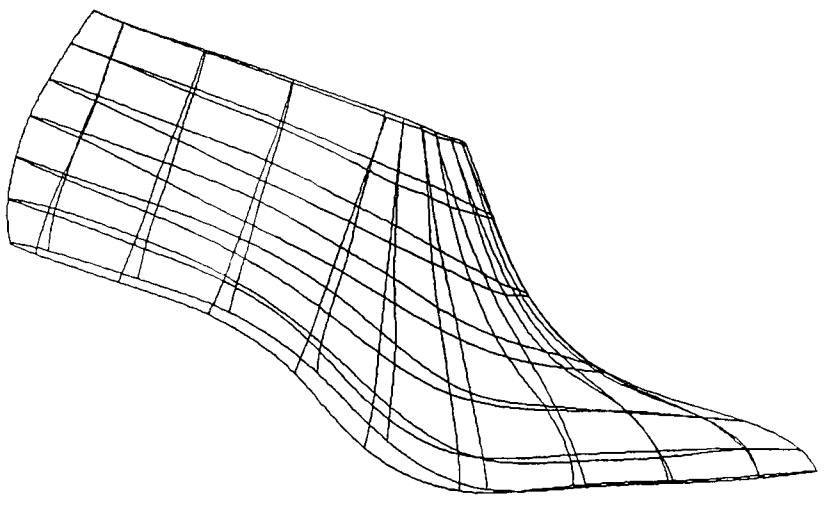


Figure 1.2: An assembly of patches representing the shoe last

1.1.2 Machining a Part

A numerical control (NC) milling machine is used to create a physically machined part from a block of raw stock. The NC machine reads numerical input that instructs the tool to make a number of movements [Marciniak 1991]. A collection of these movements is called a cutter path or a cut; as the tool traces out a cutter path it removes material from a block of raw stock to leave the desired shape. 3-axis machines allow movement in the x, y and z directions as indicated in Figure 1.3. To achieve this a combination of movement of both the tool and the stock may be

employed. Some milling machines also allow rotation about one or more of these axes, e.g. those offered by a 5-axis machine are also indicated in Figure 1.3. In this thesis we only consider generating cutter paths for a 3-axis milling machine.

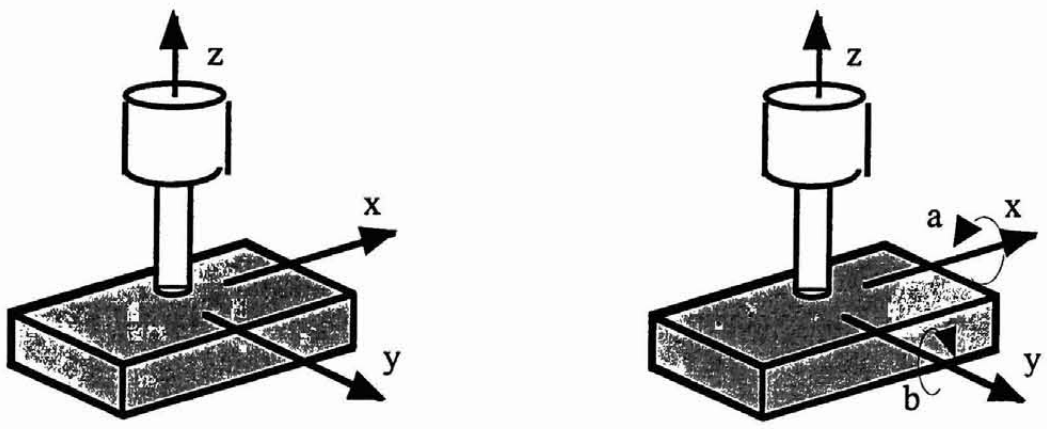


Figure 1.3: 3-axis and 5-axis machining

There are two main stages to machining, the rough and the finish cut. The main purpose of the rough cut is to efficiently remove large volumes of material and leave only a small proportion of raw stock to be removed by the finishing cut. A number of finishing cuts can be performed to leave a good surface finish. However, there is a balance to be considered between the amount of hand finishing required and the time taken to complete all the finishing cuts. In this thesis we will only be concerned with the generation of finish cuts as rough cutting has a contrasting set of issues to be considered [Dong and Vickers 1994]. For work on generating rough paths see [Kuragno 1992], [Lee *et al.* 1994] or [Loney and Ozsoy 1987].

The cutting tool used by the machine has a number of possible shapes and sizes. A diagram of the general definition of an APT-like tool as given by [Marciniak 1991] is shown in Figure 1.4. APT stands for Automatically Programmed Tool and is a NC language dating back to the fifties [Bedworth *et al.* 1991]. The most common types of cutter are the ball-nosed ($r=d/2$), the flat-end ($\alpha=0$, $\beta=0$, $r=0$) and the filleted-end cutter ($\alpha=0$, $\beta=0$, $0<r<d/2$). The work presented in this thesis is applicable to all tool shapes, however most of the error analysis is performed for a ball-nosed cutter and needs further work to generalise to all cutters.

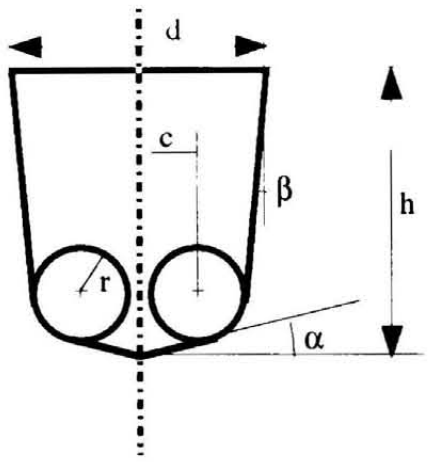


Figure 1.4: General description of an APT-like tool

The contact point between the surface and the cutter is called the cutter contact (CC) point and we define CC data as consisting of the CC point and its unit normal, **n**, as shown in Figure 1.5. Cutter centre location (CL) data also consists of a point and a vector, the CL point represents a datum for the tool and the vector is of unit length and lies in the direction of the tool axis, **t**.

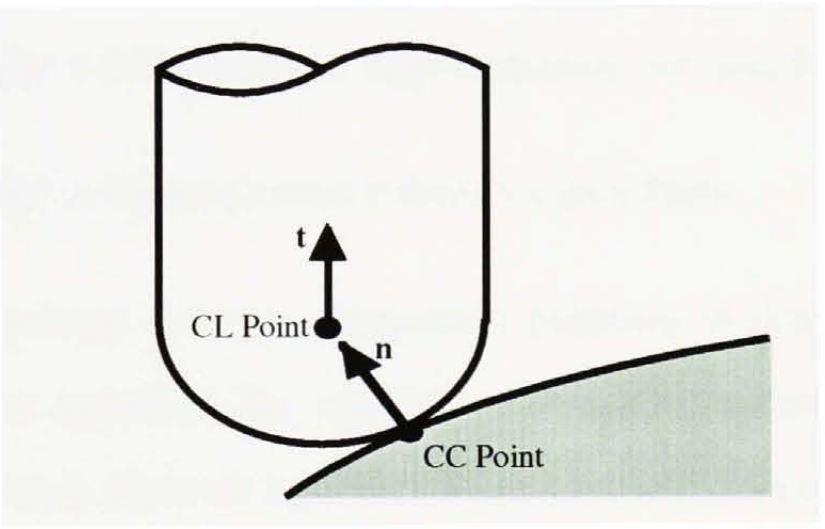


Figure 1.5: CC and CL data

A cutter path consists of a list of CL points. A subset of CL points from the cutter path leading to a single traversal of the region being machined will be called a pass. Figure 1.6 shows a number of passes on an example surface. The distance between two adjacent passes is called the pass interval. The pass interval controls the height of the cusps left between adjacent passes. Finally, the distance between consecutive CL points on a pass is called the step forward.

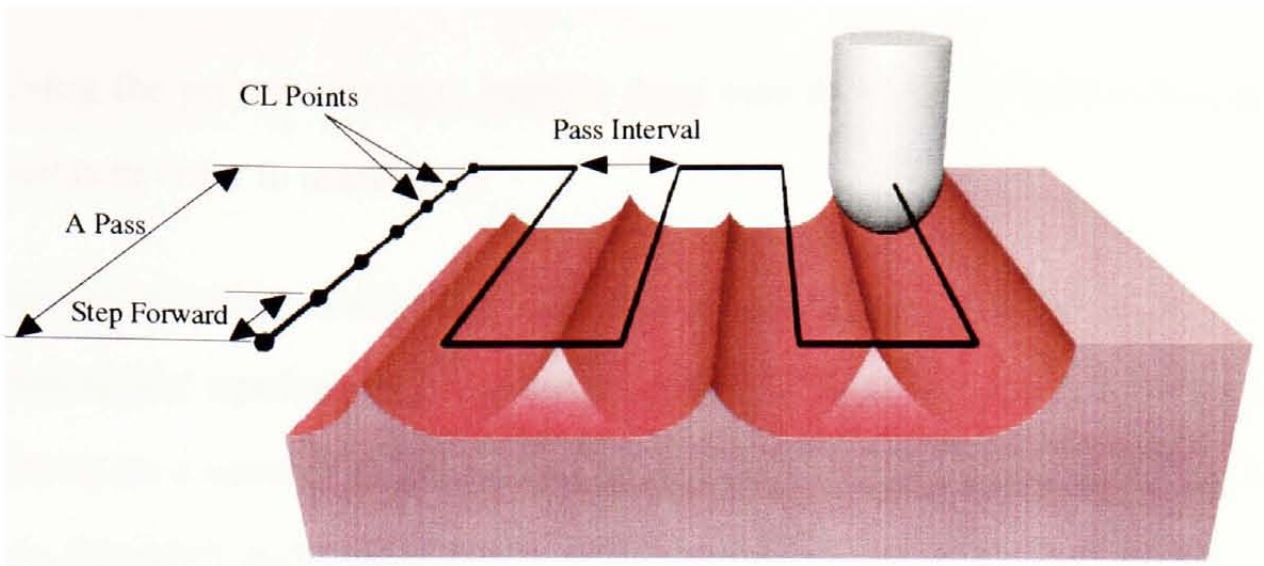


Figure 1.6: A cutter path consisting of a number of passes

[Lin and Liu 1998] give a geometry-based classification of the main approaches to generating cutter paths from a surface model, they are:

- Surface Model -> Polyhedral Approximation of Surface -> Cutter Paths
- Surface Model -> Offset Surface Approximation -> Cutter Paths
- Surface Model -> Cutter Contact Points -> Cutter Paths

The software package used for comparison purposes in this research, Powermill, employs the first approach, the surface is triangulated before the cutter paths are generated. However, there are a number of inherent problems with the current overall strategy of requiring a surface model be defined and then regarding it as the master model.

Firstly, a small discrepancy may exist at the joint between two patches. Suppose we wish to manufacture a panel on a car that runs across a number of surface joins, then we need to machine a press tool. Although the discrepancy lies within tolerance, it will lead to difficulties in defining the press tool and hence in the machining of it.

Reverse engineering is the process of defining a part from a physical model. This usually involves the sampling of numerous points from the physical model using a probe on a Co-ordinate Measuring Machine (CMM) or a non-contact Laser system.

Using the present strategy, patches must then be fitted by skilled designers to those points in order to machine it.

Special attention is required when the part to be machined contains a patch of non-rectangular topology, e.g. the red patch as highlighted on the car front in Figure 1.7. There are a number of alternative strategies for coping with non-rectangular regions, see [Hoschek and Lasser 1993], however each needs evaluating for its suitability for the given problem.

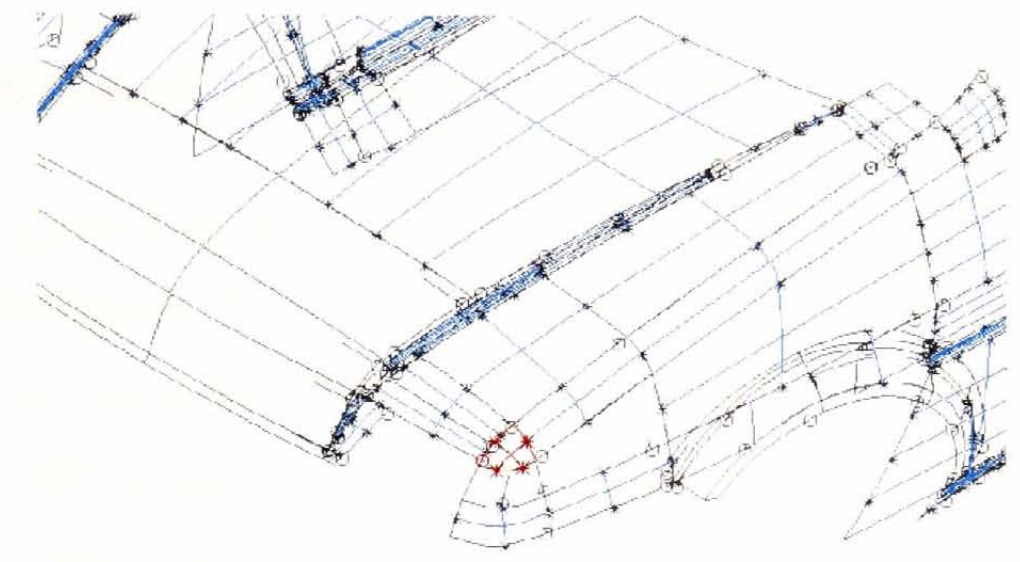


Figure 1.7: A surface patch that does not have a rectangular boundary

The final problem identified here is the proliferation of surfaces that results from transferring a surface model from one CAD/CAM software package to another [Ball 1994]. If two packages do not support the same types of parametric patches then an approximation of the part has to be made by the receiving software. Tight tolerances are generally required and this leads to the generation of numerous small patches to satisfy those tolerances.

1.2 Point-Based Modelling

We now describe an alternative approach to CAD/CAM proposed by [Ball 1997] upon which the research in this thesis is based. The main thrust of the argument is the questionable practice of considering the surface model to be the master model. Since CAD systems in general cannot match exactly shapes defined by other CAD systems,

manufactured products or an engineer's conceptual design, why should the surface model it produces be considered exact? Hence a new strategy, geometric in nature, is suggested where the surface is characterised by a set of points. This shift in philosophy results in CAD/CAM tolerances that are consistent with the design process. This approach to problem solving is alluded to by [Hartquist *et al.* 1999], that is to focus on the mathematical requirements that are inherent to a particular application.

1.2.1 Characterisation of Shape by Points

Assuming a part is curvature continuous, we can characterise that part by a set of sparse surface points. These points characterise an equivalence class of surfaces that are within an acceptable tolerance of the part using some unspecified interpolation technique. We assume the technique is of high quality and has suitable geometric constraints. Hence instead of defining a precise surface by an explicit equation, an equivalence class of surfaces are characterised that are practically the same.

The density of the surface points is dictated by the required tolerance of the characterised surface. In Figure 1.8, a sparse set of points characterising the shoe last is shown.

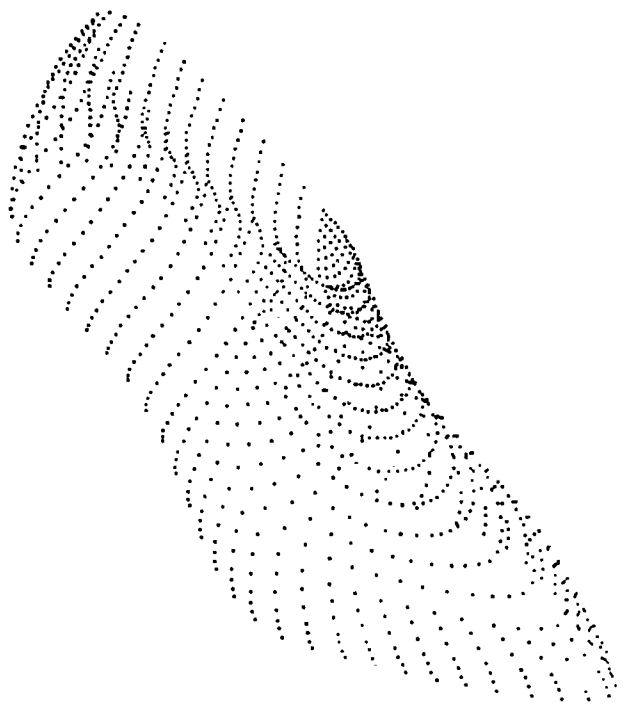


Figure 1.8: Points characterising the shoe last

1.2.2 Interpolatory Subdivision

To perform most operations on a part characterised by points we will generally require a much denser set of points. We will refer to a set of points as defining a part if they are of sufficient density to machine the part within tolerance using the suggested algorithm. To fill in the points such that they define the part we require an interpolatory subdivision algorithm, that is an algorithm that geometrically inserts intermediate points. There are a number of such algorithms available, a simple scheme is given by [Dyn *et al.* 1987] who also gives a good introduction to the area. [Kobbelt 1996] presents a scheme that can subdivide point sets with non-rectangular topologies using a *Catmull-Clark-type* split as shown in Figure 1.9. [Tookey 1997] describes a method that uses the Generalised Cornu Spiral, a locally fitted curve with monotonic curvature, to generate intermediate points.

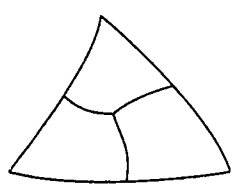


Figure 1.9: A *Catmull-Clark-type* split of a triangular region

For the purposes of this thesis we will assume we have an acceptable subdivision scheme. Figure 1.10 shows the point set resulting from subdividing the points characterising the shoe in Figure 1.8 using the algorithm suggested by Tookey.

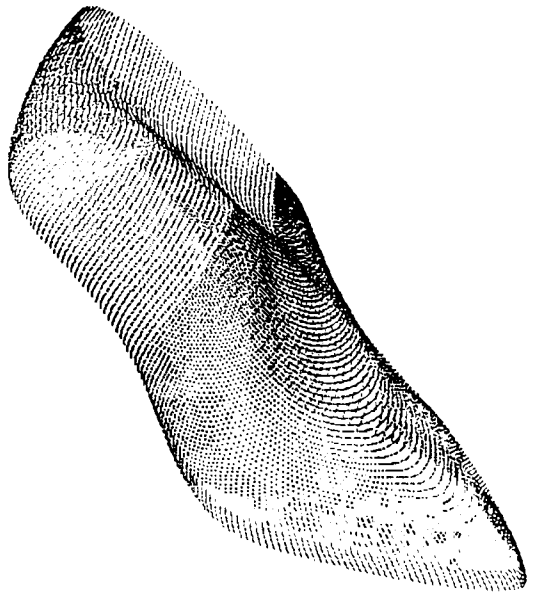


Figure 1.10: Points defining the shoe last

1.2.3 Our Approach to Generating Cutter Paths

The strategy employed is to generate cutter paths directly from a set of points that define the free-form surface we wish to machine. In particular there is no requirement to fit an assembly of parametric patches. Referring back to the classification presented by [Lin and Liu 1998], the surface modelling is removed from the process and a set of surface points becomes the initial model.

Employing this approach solves the problems stated in sub-section 1.1.2. Discrepancies at patch joins are not a concern as there are no patches, furthermore operating tolerances can be made consistent with the design approach. In reverse engineering, sparse points can be sampled from the part to characterise the shape and hence surface fitting is not required. Shapes with non-rectangular topologies can be represented by using particular patterns of points, see for example [Kobbelt 1996]. Also transferring a model between systems is easily done except in the case that the receiving system uses surface models and the sending system uses point definitions, then the problem is that of surface fitting. Another benefit is that localised surface modifications are easily made.

In this thesis we investigate implementing an algorithm employing this strategy which leads to a number of enhancements being suggested for that particular method. However it is the change in strategy that is the key message of this thesis.

1.3 Thesis Overview

In Chapter 2 we review the different techniques available for generating cutter paths. The methods are divided into three different categories dependent on the strategy they use to generate the cutter paths. The categories correspond to the 3 strategies highlighted in sub-section 1.1.2.

We first introduce the Inverse Offset Method (IOM) [Kishinami 1987] in Chapter 3 and adopt the IOM for a number of reasons. Firstly, it does not rely on an underlying

surface definition, it only requires the positional data of the surface points to perform. It is highly robust, the CL points generated are guaranteed not to cause the gouging of any surface points. The surface data is not required to have any particular topology, the algorithm can operate on completely unordered data. The only requirement of the surface data is that it is of sufficient density for the IOM to offset within tolerance. Finally, cutter paths for any tool shape can be generated but appropriate analysis is required to determine the tolerances.

The surface points do not necessarily have an order, hence we need to impose one for the purpose of generating cutter paths. We establish this order using an offset grid, that is a planar array of points, that after translation form the points of the cutter path.

We then introduce Powermill, the commercial CAM package against which the IOM is tested. Powermill was developed by Delcam who being the CASE industrial sponsors for this project allowed detailed support of its functions. The test-piece is a shoe last supplied by Clarks International. It is chosen because it is a doubly curved free-form surface; however it is noted that it does not contain pocket areas, planar surfaces nor irregular topologies.

Two shoes are machined, one by Powermill and the other by the IOM. These shoes are then measured using a CMM and surface stylus equipment. The results were used to confirm that the accuracy of surface shape and quality of surface finish produced by the IOM are comparable to those produced by Powermill.

The penultimate section in Chapter 3 gives the results of timing the two methods. Inevitably, since the IOM is operating directly on what is assumed to be scattered data, many surplus checks are made resulting in the IOM being much slower than Powermill. The results are discussed in section 3.6.

Before considering how to improve the speed of the IOM, Chapter 4 addresses a problem that occurs in regions containing a cliff-edge, i.e. where there is a tangent

discontinuity between two surfaces one of which is vertical. The error analysis given in Chapter 3 is only applicable in tangent continuous regions. However, nearly all parts will contain at least one tangent discontinuous region, namely at the boundary of the part. Hence we analyse the error in such regions and give a relation to calculate an appropriate point spacing that guarantees the error lies within tolerance.

Chapter 5 is concerned with optimising the method in order to improve its speed. By ordering the points we can reduce the number of surface points that are checked. We do this in two stages, first we optimise the algorithm to operate on nests of sections. This leads to a substantial improvement in speed. Then we optimise the algorithm to operate on parallel planar sections which gives a greater improvement in speed and allows the generation of cutter paths at a similar speed to Powermill.

In Chapter 6 we address some of the more prominent issues that arise when attempting to machine more complex parts. A number of the problems are dealt with inherently because of the geometric nature of the algorithm. For those that are not, suggestions are made as to possible solutions.

Finally conclusions and possible further work on generating cutter paths from points are given in Chapter 7.

Chapter 2

Literature Review

In this chapter we review some of the published literature concerned with generating cutter paths. The literature is divided into sections based on the classifications introduced in [Lin and Liu 1998] which depends on the fundamental geometric approach:

- Surface is approximated by a polyhedron,
- Offset surface or an approximation of it is calculated,
- CC data is calculated

or a hybrid of these. An example of a hybrid case is given in [Kuragno 1992] who generates points on the offset surface using CC data and then linearly interpolates this data to approximate the offset surface. In such cases the most prominent feature in the process dictates the section under which it is reviewed.

2.1 Methods Employing A Polyhedral Approximation of the Surface

In this section we review the algorithms that use a polyhedral approximation of the surface in the generation of the cutter paths.

[Duncan and Mair 1983] generate cutter paths from triangulations by generating CL points along the normal to the midpoint of each triangle. These points will generally lead to gouging and so an algorithm is provided that raises each point by an appropriate amount. However the cutter paths are fixed for a given triangulation and cutting tool because there is a one-to-one correspondence between the triangles and

the CL points. Hence a new triangulation is required to obtain cutter paths with either a new step forward or pass interval.

The same problem is true of the method proposed by [Choi and Jun 1989] whereby the vertices of the triangulation are offset. For a given vertex of the triangulation each of its neighbours are determined as being in either a convex, concave, parallel or inflection relation based on its relative position and normal. Each case is then dealt with appropriately to ensure the CL point does not lead to the gouging of the local triangles. In the generation of the cutter paths they also take into account the motion of the cutter as it interpolates the CL points and ensure the local triangles are not gouged. In areas where the radius of curvature of the surface is less than the radius of the cutter another algorithm is necessary to avoid gouging, Figure 2.1 shows an example of such a region.

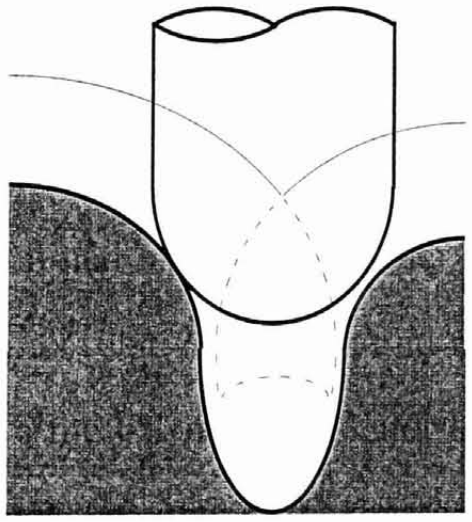


Figure 2.1: Curvature of surface leading to gouging

The CL points that lead to this type of interference, referred to as 'self-interference' [Kim and Kim 1995], are removed and two new CL points inserted. However a 'side-step' in the path, as illustrated by the cutter centre points shown in Figure 2.2, may occur since the cutter paths are non-planar. A side-step can lead to a relatively large change in direction for the cutter and produce an undesirable surface marking.

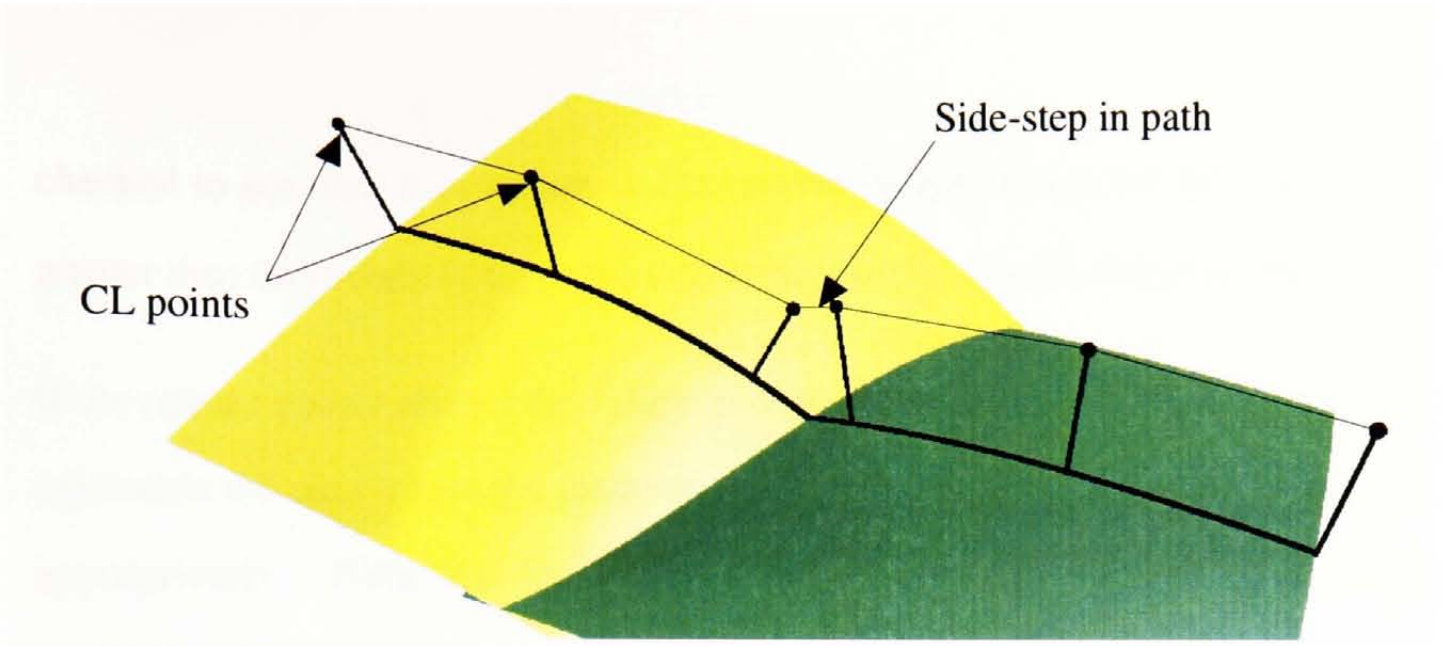


Figure 2.2: CL points leading to a side-step

[Kanda 1991], unlike [Duncan and Mair 1983] or [Choi and Jun 1989], does not let the triangulation dictate the density of the CL points. Given the (x_o,y_o) co-ordinates for the centre of the cutter, Kanda calculates the appropriate z-height such that each triangle is not interfered with. To do this he flags each triangle that lies either partially or wholly within the CC region. The CC region or cutter shadow is the area formed by projecting the tool parallel to the tool-axis onto the surface, Figure 2.3 shows an example of a CC region indicated by the grey area.

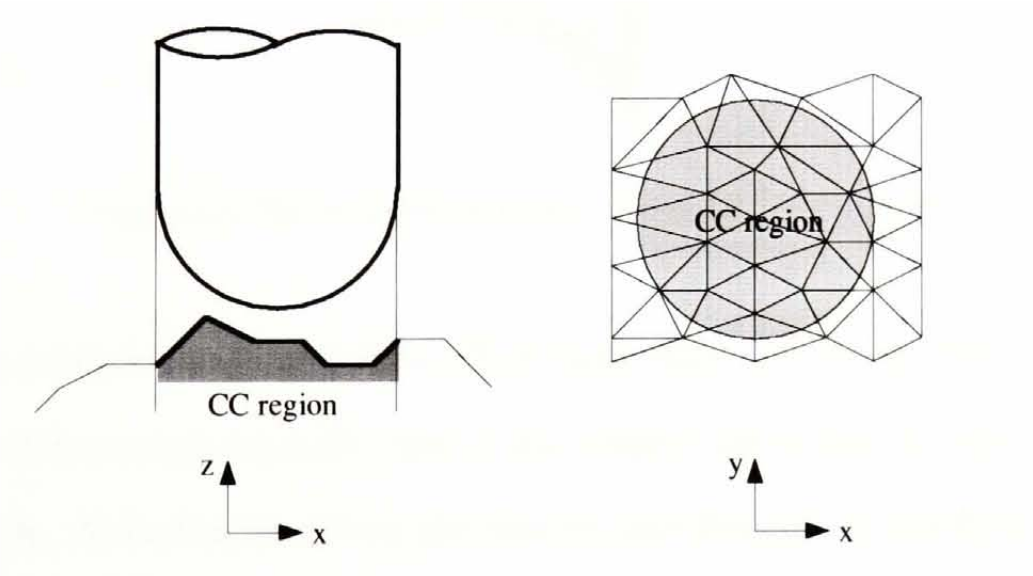


Figure 2.3: The CC region or cutter shadow

For each flagged triangle the cutter is placed above and tangent to the plane containing the triangle. If the point of contact between this plane and the sphere representing the cutter lies within the boundary of the triangle then the tool height is

checked to see how it compares to the current height setting for this (x_o,y_o). If it is greater then this height is set as the new cutter height, else the height is disregarded.

If the contact point between the sphere and the plane lies outside the triangle, then this represents the case where the cutter sits on an edge of the triangle and must be offset appropriately. Note that in this case the sphere will interfere with the plane containing the triangle.

The edge which is to be used in the offsetting is determined by which area the contact point falls into, Figure 2.4 highlights the three different regions.

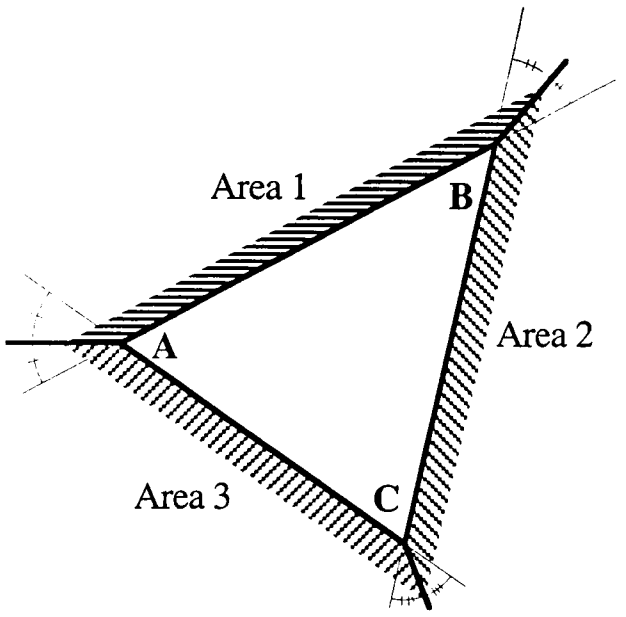


Figure 2.4: The areas that determine which edge to offset from

If the contact point lies in Area 1 then offset using edge **AB**, if the contact point lies in Area 2 then offset using edge **BC** and if the contact point lies in Area 3 then offset using edge **CA**. Note Kanda offsets the infinite line through **A** and **B**, rather than the line segment joining **A** and **B**, which means the exact offset is not generated. However this method does guarantee that the triangle is not gouged. Figure 2.5 shows the exact offset of a triangle on the left and the offset used by Kanda on the right. The red regions indicate the areas in which Kanda's algorithm fails to produce the correct offset.

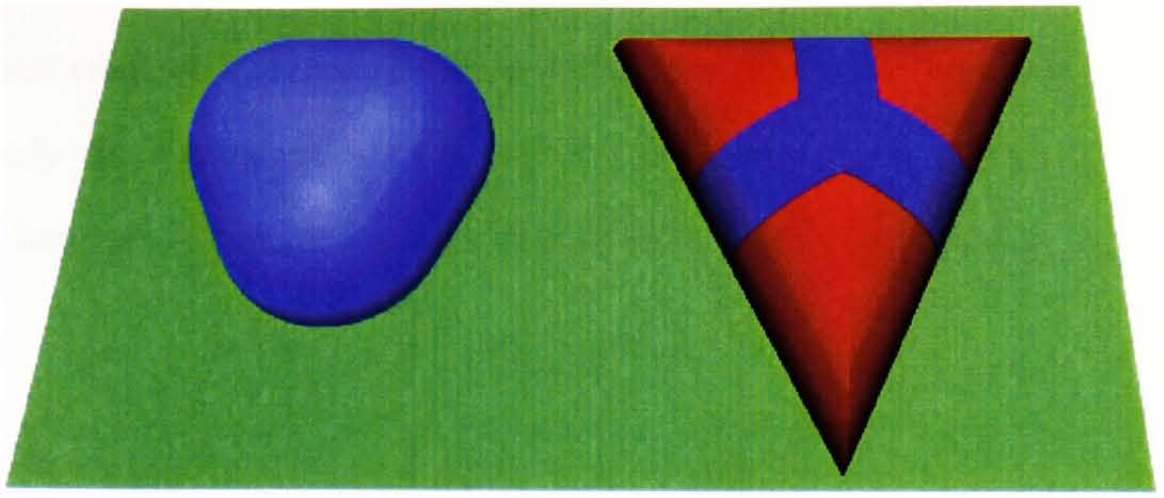


Figure 2.5: The exact offset and the offset generated by Kanda

To avoid the gouging caused by the linear interpolation of the tool, intermediate CL points are added whenever the shortest distance between the line joining two consecutive CL points and the triangulation is less than r_c. However there is no algorithm provided for determining a suitable position for the new CL point. A simple bisection method cannot be used because this will lead to the generation of many superfluous points as there is no allowance for tolerance.

[Hwang 1992]'s method is similar to the one proposed by [Kanda 1991]. However there are two notable differences that result in the actual offset of the triangulation being machined, these are:

- the vertices of the triangulation are initially offset,
- only the line segments representing the edge of the triangle are offset,

which is in contrast to offsetting the infinite line. Hwang also introduces algorithms for determining appropriate step forward and pass interval values to machine within a given tolerance. In a later publication, [Hwang and Chang 1998], the method is extended to generating cutter paths for both flat-end and filleted-end cutters.

[Lai and Wang 1994] generate the offset surface to a number of primitive shapes and to triangulated NURB patches. The offsets are intersected with the drive plane to generate CL-points. No details of the offsetting procedure are given, however in determining an appropriate pass interval the authors take into account the cutting

action of the shaft of the tool for steep surfaces unlike most previous authors who consider only the cutting action of the spherically shaped part. Figure 2.6 shows the region left between two passes of a ball-nosed cutter on a steep incline.

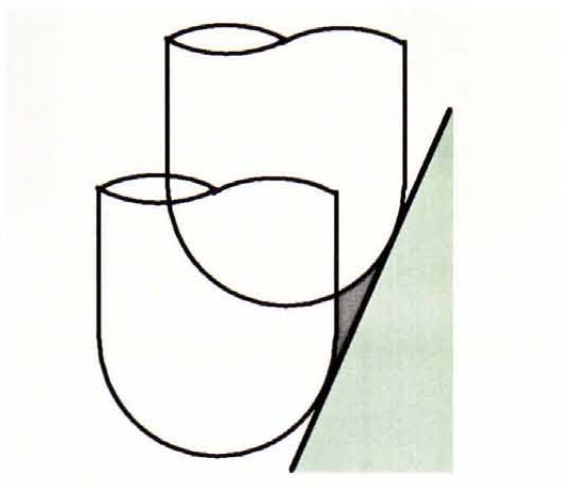


Figure 2.6: Shaft of tool affecting the shape of the cusp

[Li and Jerard 1994] consider generating cutter paths for a 5-axis milling machine using a flat-end cutter. They describe their work as an extension of [Duncan and Mair 1983] although the two algorithms bear little resemblance. Included in their paper is a clear introduction to the common problems associated with generating cutter paths.

The method starts by creating edge lists, i.e. strings of points generated by intersecting the surface triangulation with a plane. The front of the cutter is placed at each point in the edge list. This position will generally lead to gouging and hence the tool is inclined until the triangulation is not gouged. If this is impossible or results in too great a change in angle, the tool is raised slightly. However the method takes no account of the linear interpolation of the tool which in general will lead to gouged regions. Also the generation of alternative cutter paths is severely restricted since a new step forward requires a new triangulation.

The pass interval is calculated with respect to the effective cutting shape of the tool. If the flat-end tool cuts the surface with its tool axis not perpendicular to the tool motion, the effective cutting shape of the tool is an ellipse, Figure 2.7 shows the effective cutting shape of a flat-end tool.

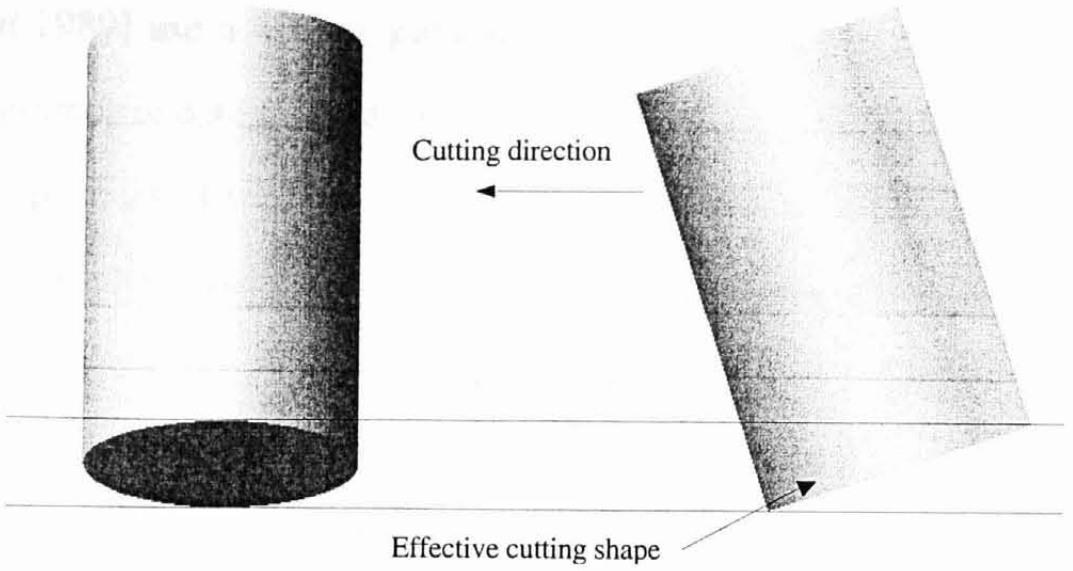


Figure 2.7: Effective cutting shape of a flat-end tool

To summarise, the polyhedral methods have developed considerably over the last fifteen years. The method by [Hwang 1992] seems particularly robust. However, it does not seem conceptually sound to generate cutter paths that lie on offset surfaces which have a greater degree of continuity than the actual part definition, Figure 2.7a illustrates this concept in 2D.

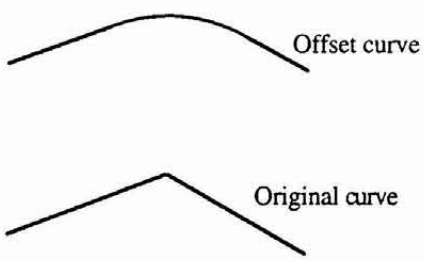


Figure 2.7a: The C^1 planar offset of a C^0 planar curve

[Li and Jerard 1994] note that using triangulation-based schemes is more efficient than point-based schemes when generating cutter paths for a flat-end cutter. This conclusion is arrived at due to the point spacing required to limit the gouging by the tool. Hence further research is required if a point-based scheme is to be effectively employed in the generation of cutter paths for flat-end cutters.

2.2 Methods Employing the Offset Surface

In this section we review some of the published methods that use the offset surface to generate cutter paths. Note that mostly an approximation of the offset surface, $\mathbf{o}_{app}$, is used as shown in Figure 2.8.

[Gao *et al* 1989] use a bicubic parametric patch to approximate the offset to another bicubic parametric patch. To do so, 16 offset data points (ODPs) are generated using the vector product of the partial derivatives at 16 points on the surface patch; these points are spaced at regular parametric intervals. The control polygon that defines $\mathbf{o}_{app}$ is determined by lining up parametrically sampled points on the patch, called approximate offset points (AOPs), to the ODPs. The same parametric intervals used to generate the ODPs are also used to generate the AOPs and then the respective AOPs and ODPs are aligned. However this method cannot guarantee a smooth surface and may lead to ripples. Figure 2.7a shows a single planar Bézier curve and its offset as approximated by another Bézier curve using the method described by [Gao *et al* 1989]. At the far left of the offset curve, the blue curve, the strict parameterisation has resulted in an inappropriate start tangent vector.

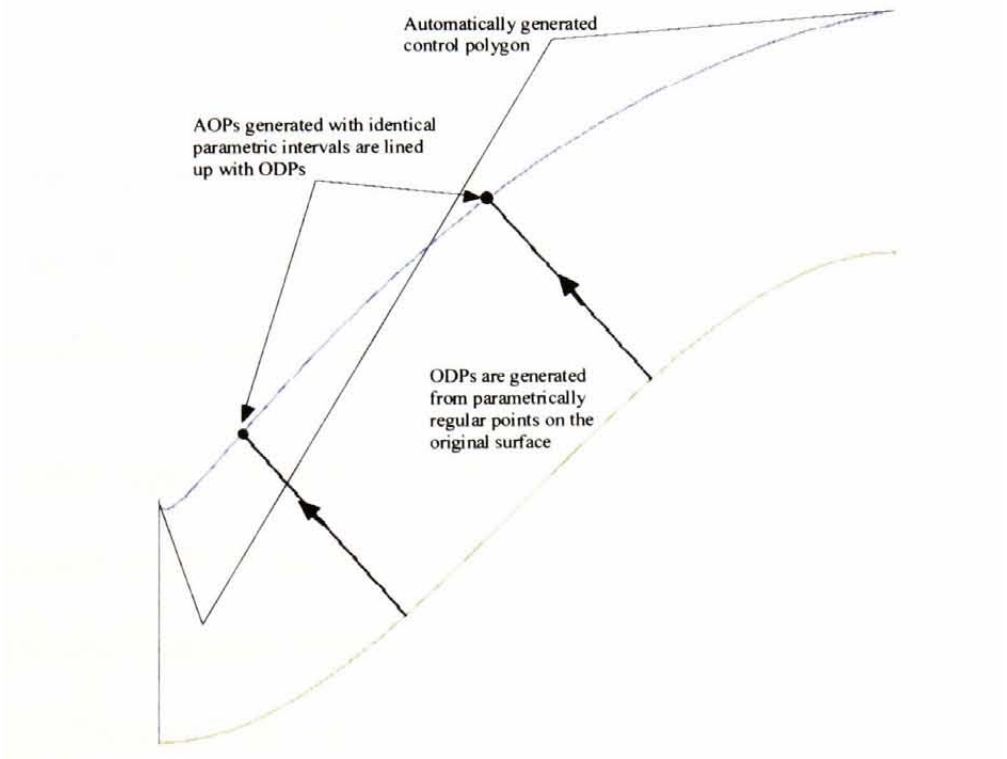


Figure 2.7a: Restrictive parameterisation of offset curve leading to a ripple

[Chen and Ravani 1987] suggest a more sophisticated method which generates a number of ODPs from the surface patch and parametrically corresponding AOPs on $\mathbf{o}_{app}$. The number of points used and their parametric spacing are left to the user to determine. Then $\mathbf{o}_{app}$ is fitted using a least-squares fitting technique. Next, the AOPs are adjusted by finding new parameter values such that they are closer to the ODPs

using a Newton-Raphson based search, Figure 2.8 illustrates the selection of a new AOP. This type of parameter correction technique used for surface fitting is further detailed in [Hoschek and Lasser 1993].

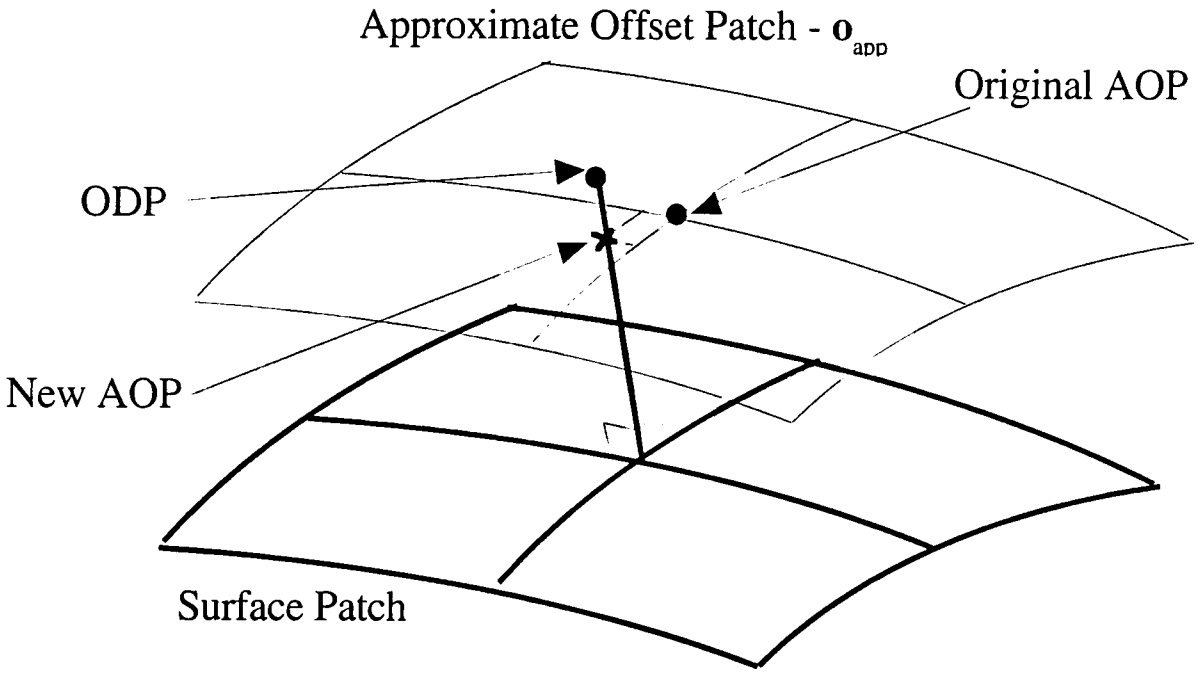


Figure 2.8: Parameter correction of AOP

A method for contouring is given to generate the CL points. Neither [Gao *et al* 1989] nor [Chen and Ravani 1987] give an error analysis of the offset surface approximation and hence a tolerance for the generated cutter paths cannot be determined.

[Kuragno 1992] approximates the offset surface with a rectangular grid. To define the part surface the author uses a rational bicubic parametric patch that was investigated by [Hosaka and Kimura 1980] and allows patches to be joined with tangent plane continuity. To generate the CL points a number of ODPs are generated along isoparametric lines and hence the resulting set of points has a rectangular topology. Neighbouring ODPs are joined with line segments corresponding to the grid-like structure and intersecting these line segments with a drive plane generates the cutter paths. Although the errors generated by the step-forward and pass-interval distances are examined, the errors generated by the grid approximation of the offset surface are not.

[Chen *et al* 1993] formulate the equation of the intersection curve between the offset surface of a parametric patch and the drive plane and then linearly approximate it in parametric space. Given the implicit equation of the plane, the equation for the intersection curve will be of the form:

$$A\mathbf{x}(u,v) + B\mathbf{y}(u,v) + C\mathbf{z}(u,v) + D = 0 \tag{2.1}$$

This defines a planar curve which is then linearly approximated in parameter space, Figure 2.9 shows an example of a linearly approximated parameter curve in parametric space.

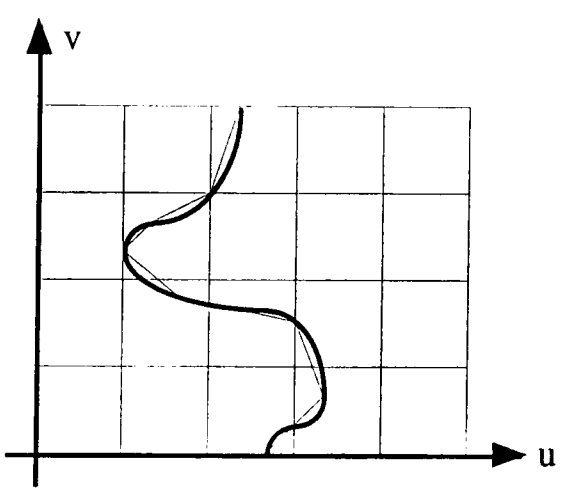


Figure 2.9: A planar surface curve linearly approximated in parametric space

This linear approximation defines a curve lying on the offset surface that is not planar. However the linear approximation is performed such that the deviation between this curve and the planar offset curve remains within tolerance. The vertices of the linear approximation are then used to generate the CL points of the cutter path. The set of piecewise linear curves defined by these CL points are then sorted and appropriate curves linked to form the cutter paths. Loops and gaps are easily detected and removed since the curves are planar. Figure 2.10 illustrates the method used to do this.

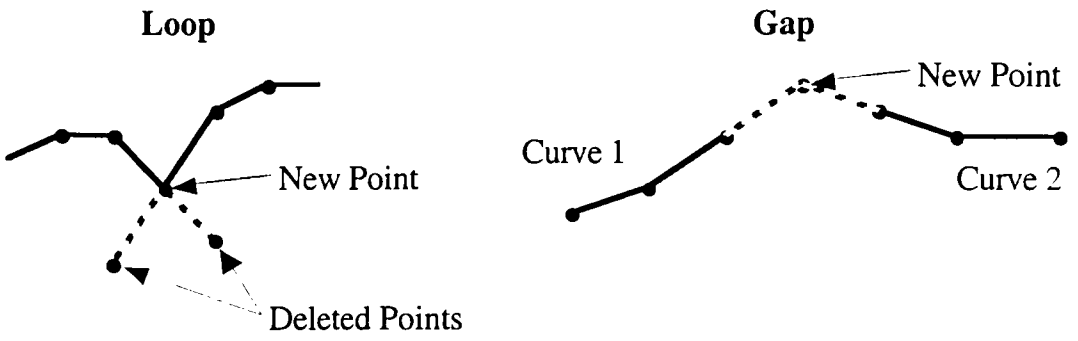


Figure 2.10: Illustration of method used to cope with loops and gaps

[Chen *et al* 1993]'s method seems generally sound and robust but will fail when the vector product of the partial derivatives is zero (e.g. at a cusp) as will the other methods that use ODPs. Also there is no suggested method for dealing with the boundary of a part which will generally lead to surplus stock being left as shown in Figure 2.11.

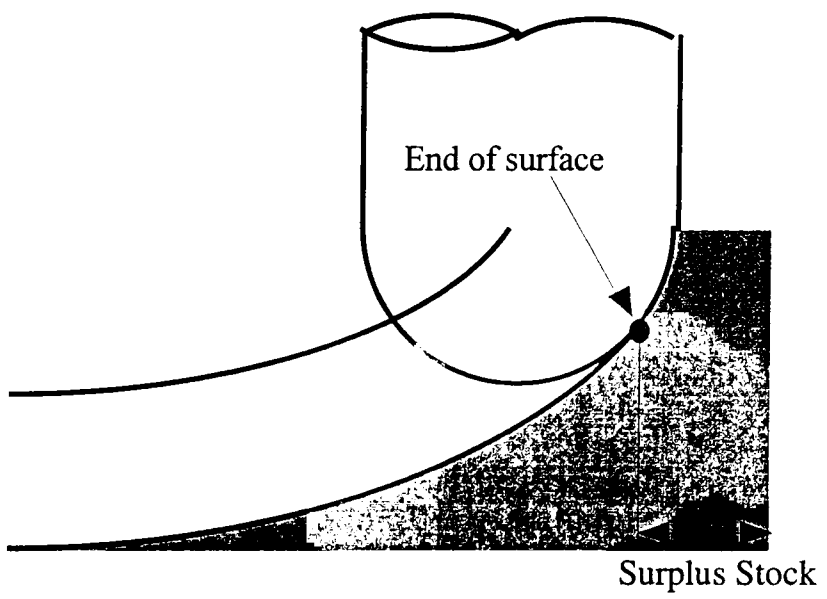


Figure 2.11: Result of cutter paths not being generated at boundary

[Tang *et al* 1995] solve the boundary problem by offsetting the surface and its boundary separately. To offset the boundary curve of a surface they first derive the guard surface of a point **P**, grd(**P**), that is the surface which the CL point of a tool must move along to machine **P**. This surface is the inverted tool that is defined as the image of the tool mirrored against a plane orthogonal to the tool axis. The concept of the inverted tool lies at the heart of the Inverse Offset Method, the focus of this thesis. A proof that the inverted tool is equivalent to grd(**P**) is given by [Kishinami *et al* 1987] and is reproduced later in Chapter 3.

The offset surface of a curve C is then defined as the envelope of the sweep of grd(**P**) along C, Figure 2.12 illustrates this. However, it is difficult to generate points on the envelope using this definition and hence the authors suggest an alternative but equivalent definition to the guard surface. The alternative definition, **E**(C), is given by intersecting the inverse tool with the plane orthogonal to the curve at each point on C. Hence the offset surface of C is defined by a sequence of curves as illustrated in Figure 2.13. The end points of C are dealt with separately, the offsets of these are defined by grd(C(0)) and grd(C(1)).

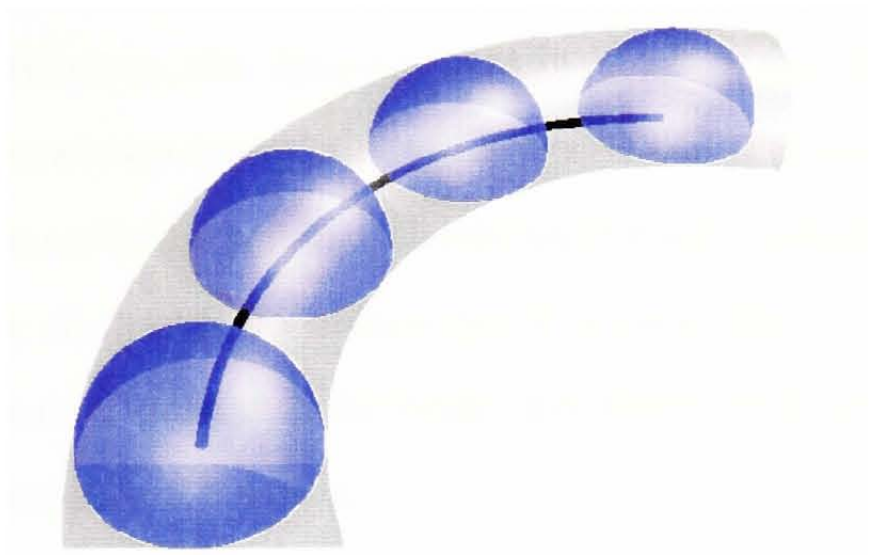


Figure 2.12: Sweep of grd(**P**) along C

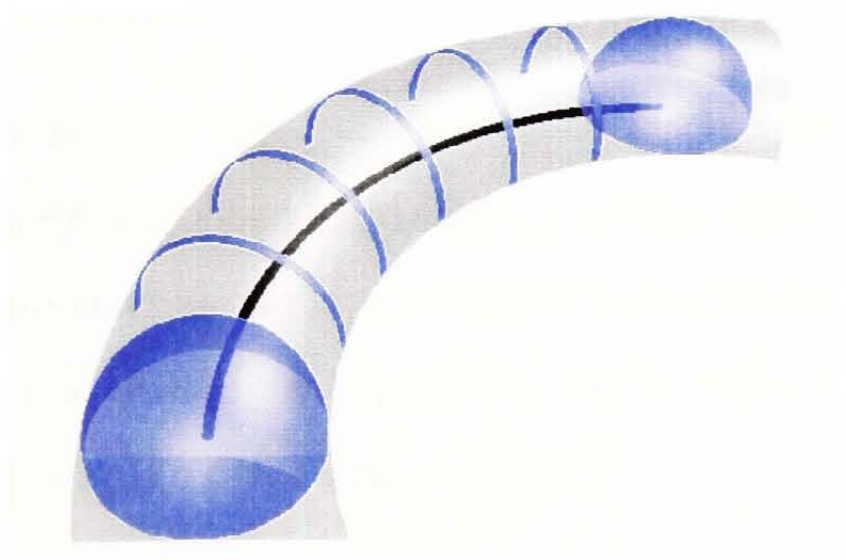


Figure 2.13: The alternative definition, **E**(C)

[Tang *et al* 1995] generate the cutter paths in a similar fashion to [Chen *et al* 1993] and use a Newton-Raphson based technique to find parameter values on the offset to give a planar curve. The algorithm should produce good results and is applicable to

all cutter shapes. However the algorithm that finds the appropriate parameter values has to be run on both the offset of the current patch and on the four offsets of the boundary curves, a computationally expensive process.

[Kim and Kim 1995] use a bicubic Coons patch to define $\mathbf{o}_{app}$ with respect to another bicubic Coons patch. Since the position and partial derivatives of its corner points define a Coons patch, appropriate data for the corner points of $\mathbf{o}_{app}$ are derived.

To generate the CL data, the authors use the parametric definition of the approximate offset surface. However, this does not solve many of the key problems associated with parameter based cutter paths, for example pass interval control. Also the authors suggest that to remove the problem of self-interference, cutters with radius small enough to fit into all areas of the surface should be used. Hence if a surface contains fillets with greater curvature than the cutter, then manual intervention is required to ensure that the surface is not gouged.

These methods do not give tolerances for the approximation of the offset surface that are essential when generating cutter paths. With appropriate error bounds these methods may prove useful.

The methods that generate points that lie on the exact offset surface rely on iterative root finding algorithms to find appropriate parameter values to generate the CL points. These algorithms lead to the methods being computationally expensive. Also all the methods presented in this section rely on the vector product of the partial derivatives being non-zero, which is not guaranteed.

2.3 Methods Employing CC Data

This section is essentially divided into two parts. The first part examines a selection of papers that use CC data to generate the cutter paths whilst the second part is concerned with the IOM, which is the focus of this thesis.

[Haapaniemi *et al* 1986] uses Gregory patches to describe the part surface. Gregory patches are modified bicubic Bézier patches similar to those used by [Kuragno 1992]. The CL points are calculated using the CC data from the isoparametric lines lying on the surface as illustrated in Figure 2.15.

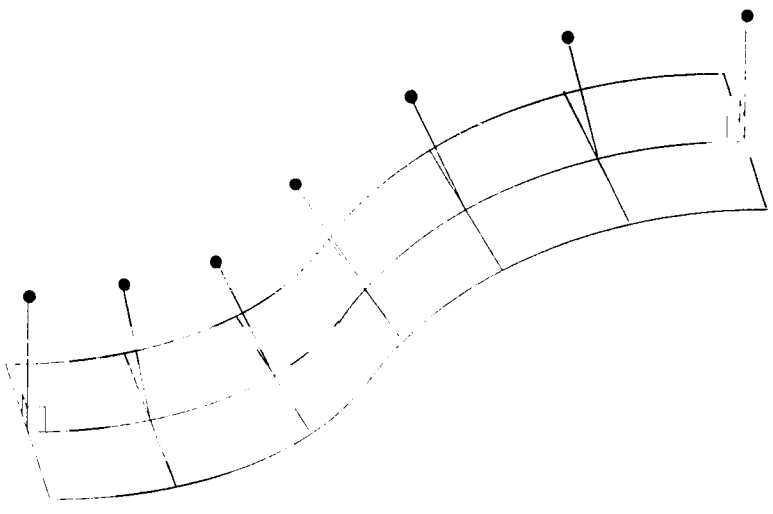


Figure 2.15: CL points generated directly from the CC data

The gouging of neighbouring patches is avoided by removing the offending points and inserting new CL points in a similar manner to [Choi and Jun 1989], however this leads to the same side-step problem as the generated cutter paths are non-planar. A problem highlighted by the authors is that, unless extra surfaces are defined at vertical regions, fouling will occur. Also there are no means given by which to calculate the error resulting from the step-forward or pass interval distance.

[Loney and Ozsoy 1987] generate cutter paths for bicubic parametric patches from CC data along isoparametric curves. An appropriate pass interval is calculated by locally approximating the surface perpendicular to the pass direction by the osculating circle, a method commonly employed in the literature. A similar approximation is commonly used to determine the step forward as well, however Loney and Ozsoy calculate the step forward based on the actual deviation of the isoparametric curve from the chord. Also included are the details of an algorithm for generating roughing

paths. However the authors do not consider the gouging caused to patches other than the one currently being machined, nor do they consider the self-interference problem.

[Choi *et al* 1988] present an algorithm to generate cutter paths for analytic compound surfaces, i.e. an assembly of surfaces each of which is represented by an equation of the form:

$$g(x, y, z)=0. \tag{2.2}$$

In particular they present the implementation of a constructive solid geometry (CSG) scheme that defines surfaces by Boolean operations on a selection of primitive shapes. The algorithm also accommodates the definition of a parametric base surface. They define two terms, CC-Cartesian and CL-Cartesian. Algorithms that use planar strings of CC points are CC-Cartesian and algorithms that generate planar cutter paths are CL-Cartesian. The algorithm described by [Choi *et al* 1988] is CC-Cartesian.

Both step forward and pass interval values are calculated based on local approximation of the surface by the osculating circle. However the method suffers from the gouge avoidance algorithm leading to the side-step problem which is also a problem with the follow on paper by the same leading author, [Choi and Jun 1989].

[Kim and Ko 1994] employ an original method in which bicubic Bézier surfaces are first subdivided until the 'thickness' of each patch is within some tolerance. The thickness of the patch is approximated using the control polyhedron and is measured along the vector given by:

$$\mathbf{N} = \frac{(\mathbf{V}_{33} - \mathbf{V}_{00}) \times (\mathbf{V}_{30} - \mathbf{V}_{03})}{\left\| (\mathbf{V}_{33} - \mathbf{V}_{00}) \times (\mathbf{V}_{30} - \mathbf{V}_{03}) \right\|}$$

Figure 2.16 shows the vectors $\mathbf{V}_{33}$-$\mathbf{V}_{00}$ and $\mathbf{V}_{30}$-$\mathbf{V}_{03}$ represented by the red lines, the normal to these vectors represented by the green arrow, and the measured thickness of the control polyhedron.

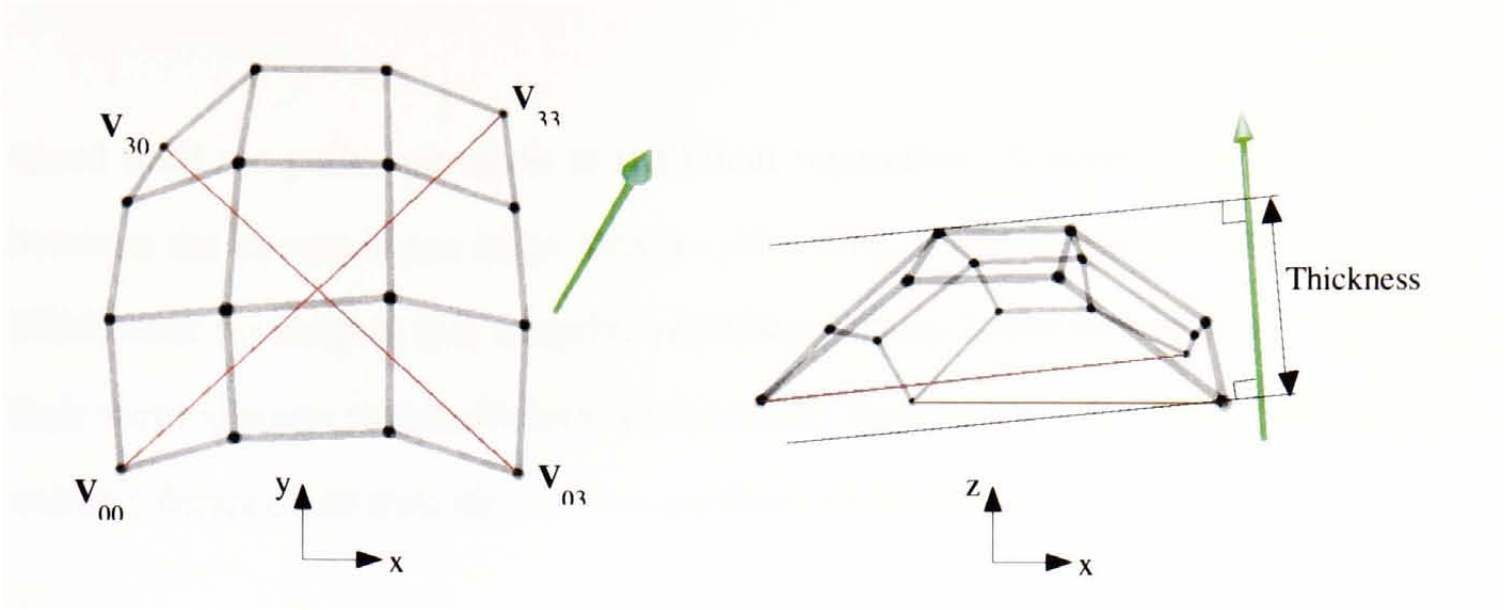


Figure 2.16: Thickness of a given control polyhedron

Once each patch's thickness is within tolerance it is then approximated by a bilinear tile. The error introduced by this approximation is restricted to being less than the thickness of the control polygon since the Bézier surface lies within the convex hull defined by the polygon. Intersecting the boundaries of these tiles with the drive plane generates the CC points. The author then suggests that this piecewise linear curve be compared against the 'true intersection curve'. Intersecting the control polygon with the drive plane generates the control vertices of this curve. However this process does not generate the true intersection curve and hence will create inaccurate error measures.

[Huang and Oliver 1994] use a CC-Cartesian algorithm for generating cutter paths for a parametric surface. They suggest the tool path is projected onto the surface to detect regions where there is too much gouging or undercut. This measure is used to alter the step forward appropriately. However such an operation is computationally expensive and will result in lengthy path generation times. The method used for gouge avoidance also suffers from the side-step problem.

[Hermann 1988] suggests two algorithms, the first is similar to [Haapaniemi *et al* 1986] except that the CC points are generated from the triangulation of the surface rather than the surface itself. The second algorithm is more interesting with the cutter contact points being generated along isoparametric curves. If two of these curves come within a prescribed distance of each other, the tool can be moved along at high

speed until the paths are again at sufficient separation. Conversely, if the distance between the curves is too large then an extra path is introduced. [Elber and Cohen 1994] later investigate this adaptive machining along isoparametric lines. However their method starts with sufficient isoparametric lines to machine the entire surface to within tolerance and then superfluous portions are removed.

[Lin and Liu 1998] address the generation of both rough and finish cuts from a large set of data points lying in parallel planes. The points are pre-processed for indexing purposes and to obtain certain geometric data. The algorithm essentially overlays a regular grid of points onto the measured data points. The height at each grid point can be quickly computed using the geometric data found in the pre-processing stage.

To create the cutter paths a CL point is generated above each point in the regular grid. Suppose we wish to calculate the height of the CL point in the m^{th} column and n^{th} row as shown in Figure 2.17.

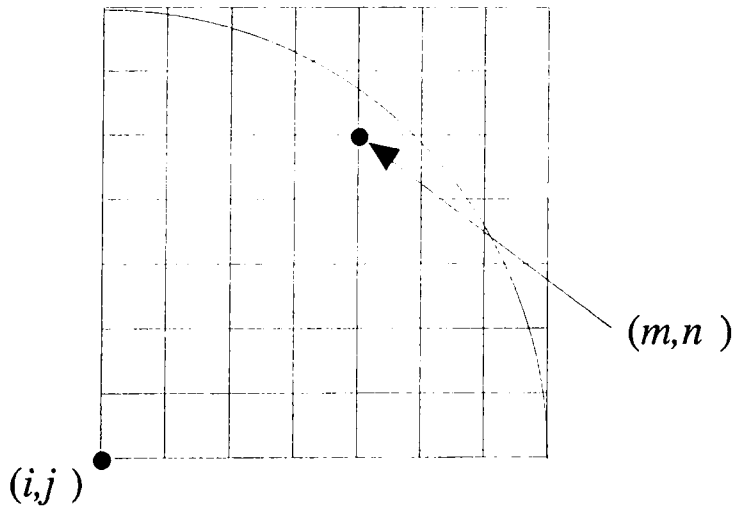


Figure 2.17: Points at which the height is pre-calculated

Let $z'(m,n)$ be the height of the offset point in the m^{th} column and n^{th} row of the grid, $z(i,j)$ be the part height at the i^{th} column and j^{th} row and R be the radius of the cutter. Then the height of the CL point is given by:

$$z'(m,n) = \max_{(i,j):z(i,j)\,in\,CC-region} \{z(i,j) + h(i,j,m,n,R)\}. \qquad (2.3)$$

where:

$$h(i,j,m,n,R) = \sqrt{R^2 - L^2} \tag{2.4}$$

where L is the Euclidean distance between the cutter centre and the mesh point in the x-y plane. To save processing time $h(i,j,m,n,R)$ is pre-calculated for every mesh point combination in a quarter of the cutter region. Only a quarter of the region needs pre-calculating because:

$$\begin{aligned} h(i,j,i+istep,j+jstep,R) &= h(i,j,i-istep,j+jstep,R) \\ &= h(i,j,i+istep,j-jstep,R) \\ &= h(i,j,i-istep,j-jstep,R) \end{aligned} \tag{2.5}$$

The problem with this method is that only cutter paths lying along a row or column of the mesh can be generated. Although an analysis of the error of this algorithm is not given, the author does suggest an algorithm that attempts to ensure the step forward does not lead to excessive undercut or gouging. The algorithm generates a CL point between the two current CL points and then checks the deviation of the new path from the original path. If it is not within tolerance then the new CL point is included and the process repeated on the new section of cutter path.

[Lin and Lin 1999] extend the work presented in [Lin and Liu 1998] by including a smoothing algorithm to cope with noisy data such as that from contact or non-contact CMMs.

The problem underlying nearly all of the methods based on CC data is the fact that non-planar cutter paths with loops, large gaps and self-interference portions may be generated. The detection of such regions is difficult and the fixing of them generally leads to the side-step problem. Also, like the offset surface methods the algorithm will fail when the vector product of the partial derivatives is zero. The method of [Lin and Liu 1998] does not suffer from these flaws and in strategy is similar to the Inverse Offset Method which we now consider.

The IOM is a method that generates cutter paths for 3-axis machining and is first presented in [Kishinami *et al* 1987] which includes a proof applicable to an arbitrarily shaped tool machining a single point in space. The surface upon which the tool's datum must move to machine the point is the tool shape reflected in the x, y and z directions. By observing that tool shapes are symmetrical about the tool axis, the inverse tool shape becomes the offset surface of the space point; key definitions and the details of this proof are reproduced in Chapter 3. Figure 2.18 shows the inverted tool shape for a filleted-end cutter.

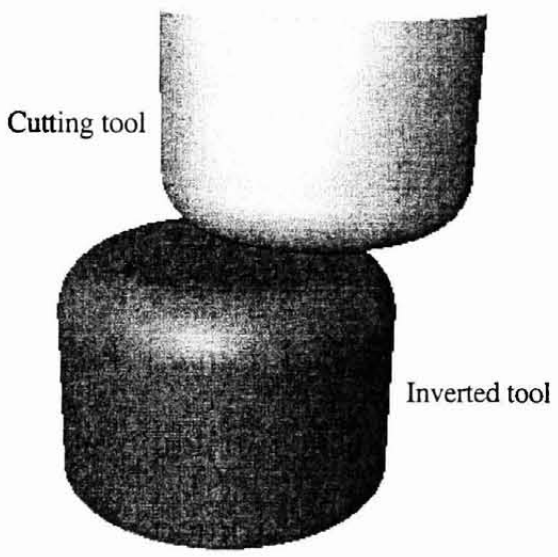


Figure 2.18: The inverted tool of a filleted-end cutter

The IOM is the application of this to a surface. However it is difficult to generate CL points from the surface defined by the envelope of the inverted tool as it is swept along the surface. This is because the analytical determination of each CL point on the envelope includes solving an integral equation, hence we discretise the surface to simplify the computation.

To store the CL points a 2D array is prepared, the elements of which have pre-defined x and y co-ordinates. The z-height of each is then determined using the IOM. The authors suggest a quadtree structure should be used to store the surface points to improve performance.

[Takeuchi *et al* 1989] describes a CAD/CAM system called P-CAPS for mould manufacture incorporating the IOM. For a small mould making company an

expensive and advanced CAD/CAM package would not be cost effective whereas this system would. Once the part has been defined, using either solid or surface geometry, it is converted into a spatial array model, Figure 2.19 illustrates a few elements from an example of such a model. This spatial array can then be operated on with specific use of the IOM for the generation of NC data. Also presented is the application of the IOM for generating fillets and cavity-core moulds.

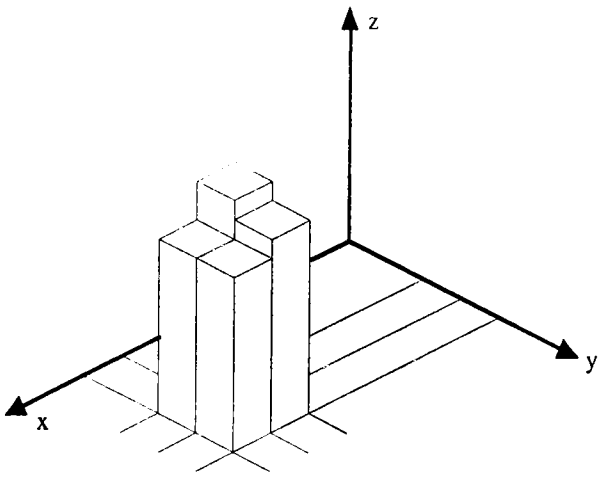


Figure 2.19: Spatial array representation of surface

[Suzuki *et al* 1991] then apply parallel processing to the IOM to enhance its performance. A detailed description of the optimisations provided by parallel processing are beyond the scope of this research, however it is made clear that the IOM naturally lends itself to parallel processing.

[Saito and Takahashi 1991] extend the G-buffer method to 3-axis NC machining. The G-buffer method was originally intended for rendering surfaces. A G-buffer is a 2D array, each 'pixel' of which contains various geometric information. This information can then be used to efficiently perform various rendering techniques. They point out that the application of this method to generating tool offsets is equivalent to the IOM.

The method is then applied to tool path verification, tool load evaluation, collision avoidance and identifying undercut regions. It is noted by [Saito and Takahashi 1991] that an error analysis of the IOM is required to generate finishes to within final product tolerances.

An application to rough cutting is also mentioned, but this amounts to increasing the cutter radius to define a virtual cutter. No mention is given to the machining of constant z-levels or to the optimisation of cutter path topology, important factors in rough machining [Li *et al.* 1994].

The greatest advantage of the method, as stated by the author, is its simplicity. The cutter path generation and simulation for a given part are based on the same methodology and rely on the same underlying definition. This leads to software packages that are more efficient.

Lying at the heart of the CAM system proposed in [Choi *et al* 1994] is the concept of the soft master model. The main requirement of the data structure for the soft master model is versatility as it is subjected to a variety of operations. It is suggested that a discrete representation scheme is used similar to that described by [Takeuchi *et al* 1989]. Again for the purposes of cutter path generation, the IOM is suggested due to its robustness. The authors also identify that little is understood about the approximation errors.

In summary the IOM is attractive due to the robustness it offers in generating gouge free planar/non-planar cutter paths in a direct and simple manner. Hence in this thesis we investigate the effectiveness of the IOM as a point-based algorithm for generating cutter paths. In Chapter 3 we provide an error analysis and compare the quality of a physically machined part produced by the IOM against that produced by Powermill.

Chapter 3

Evaluation of the IOM

The aim of this chapter is to demonstrate that it is possible to generate good quality cutter paths directly from a point definition of a part. It is required that the part should be machined to within tolerance and have a good surface finish.

The algorithm used is a slight modification of the Inverse Offset Method proposed by [Kishinami *et al* 1987] which was introduced in section 2.3. Their proof that this method gives an approximation of the tool offset is given in sub-section 3.1.1. The IOM has a number of advantageous properties that are listed in sub-section 3.1.2. An error analysis of the algorithm is given in sub-section 3.1.3. Note that this analysis assumes that the underlying part shape is tangent continuous. An error analysis applicable to regions containing a cliff is given in Chapter 4.

To demonstrate the effectiveness of the IOM its performance is compared against that of a commercially available software package, Powermill, the details of which can be found in section 3.2. Two shoes are machined using the cutter paths generated by each of these algorithms. In section 3.3 these shoes are measured on a CMM and compared to the original shoe last definition. Some of the discrepancies are found to lie slightly outside the machining tolerance on both machined parts. However in the case of the IOM shoe, it is shown that they are consistent with the error of the measuring process.

In section 3.4 we compare the surface finish of the two shoes. The results indicate that the IOM creates a slightly better surface finish than Powermill for equal pass interval distances. In section 3.5 we compare the generation periods of the two algorithms and the IOM is found to take significantly longer than Powermill to generate cutter paths. Finally, the results of this chapter are discussed.

3.1 The IOM

3.1.1 Assumptions, Proof and Algorithm Description

The research described in this chapter is based on the following assumptions:

- the part is defined by points,
- a denser definition is available if required to satisfy machining tolerances,
- the underlying part shape is tangent continuous and
- the minimum radius of curvature for the part is known.

We re-create here the proof given in [Kishinami *et al* 1987]. First it is proved that the tool offset of a single point in space is the tool shape reflected in the x,y and z axes. Then by observing that machine tools are symmetric about the tool axis, we deduce that the inverted tool gives the offset surface of the point. We then apply this theory to a discrete representation of a part to approximate the tool offset surface and generate the cutter paths using a 2-dimensional (2D) array called an offset grid. The grid points before being offset and the surface points are shown in Figure 3.1.

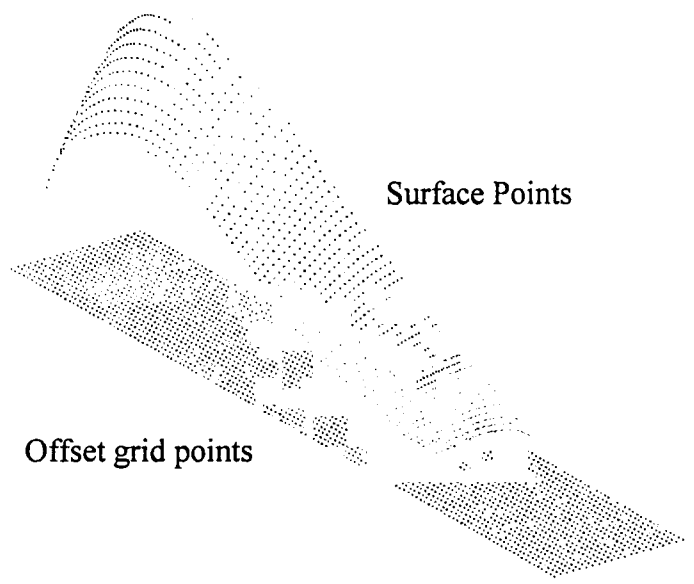


Figure 3.1: Offset grid and surface points

Suppose we wish to machine a space point, O_w, located at the origin of the workpiece co-ordinate system, O_w-X-Y-Z, using a tool with shape $S(u,v)$ where $u_0<u<u_1$ and $v_0<v<v_1$. Let $S(u,v)$ be defined in the tool co-ordinate system, O_t-x-y-z, and let the

workpiece and tool co-ordinate system have parallel axes. Let the origin of the tool co-ordinate system, O_t, be the reference point of $S(u,v)$. Then as the tool moves while keeping in constant contact with O_w, the following relation is obtained between the two co-ordinate systems:

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & -S_x(u,v) \\ 0 & 1 & 0 & -S_y(u,v) \\ 0 & 0 & 1 & -S_z(u,v) \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{3.1}$$

where $(S_x(u,v), S_y(u,v), S_z(u,v))$ is the vector from O_w to O_t. Figure 3.2 shows $S(u,v)$, the tool shape, and $S'(u,v)$, the locus of O_t as it moves around O_w for a fixed value of z. The tool has a distinctive shape to highlight the symmetry of the system. The surface on which this path lies is defined in the workpiece co-ordinate system by:

$$S'(u,v) = (-S_x(u,v), -S_y(u,v), -S_z(u,v)) \tag{3.2}$$

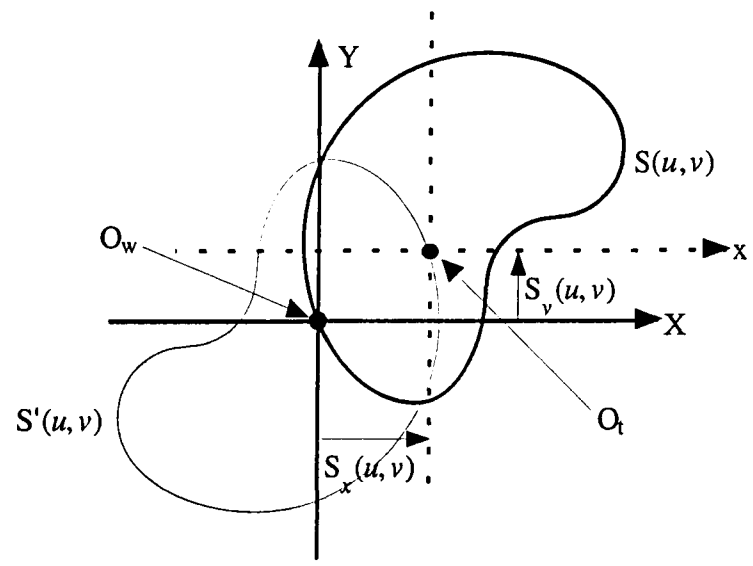


Figure 3.2: Tool path that tool with shape S must take to machine Ow for fixed z

We now consider the symmetry of the tool around the tool-axis. If the tool-axis lies along the z-axis of the tool co-ordinate system then:

$$(S_x(u,v), S_y(u,v), S_z(u,v)) \equiv (-S_x(u,v), -S_y(u,v), S_z(u,v)), \tag{3.3}$$

where $u_0<u<u_1$ and $v_0<v<v_1$.

This gives the tool offset surface of O_w as:

$$S'(u,v) \equiv (S_x(u,v), S_y(u,v), -S_z(u,v)). \tag{3.4}$$

That is the tool offset surface is the same shape as the inverted tool as shown in Figure 3.3.

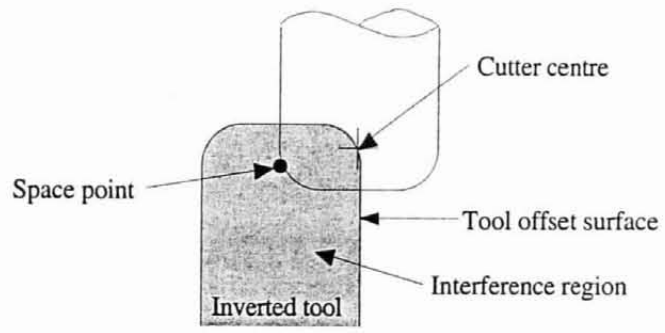


Figure 3.3: The inverted tool

The exact tool offset is the envelope of the volume generated by moving the inverted tool along the desired part surface. However it is in general difficult to generate cutter centre points from the definition of the exact envelope and hence we operate on a discrete definition of the part to generate the cutter centre points.

We now describe the method by which the cutter paths are generated using the inverted tool. By placing the cutter centre point of the inverted tool at each point in the definition of the surface we generate an approximation of the tool offset surface as shown in Figure 3.4.

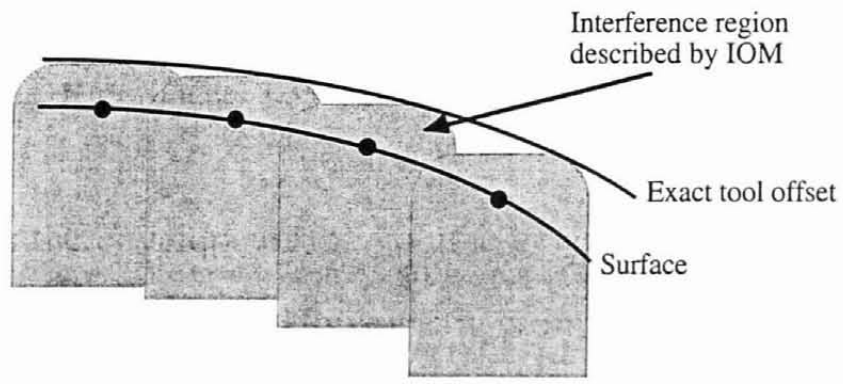


Figure 3.4: Approximation of tool offset surface

A 2D array of points, called an offset grid, is prepared which is used to store the cutter path. Through each 2D point in the CC-region, as defined in section 2.1, we construct a grid line running parallel to the tool-axis. To generate the cutter path points we intersect these grid lines with each inverted tool. The highest intersection point gives the cutter path point for each grid line. Figure 3.5 illustrates this procedure.

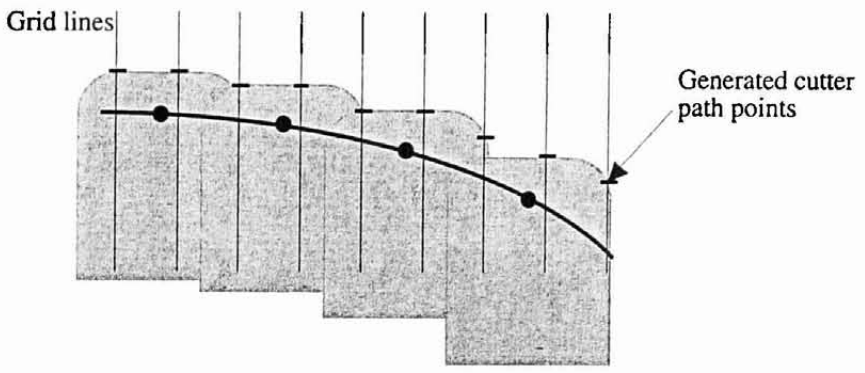


Figure 3.5: Generation of tool offset points

Suppose we wish to machine using a tool with general description as shown in Figure 3.6, where z_o is the height of the cutter centre point, R_1 is the radius of the flat part of the cutter, R_2 is the fillet radius, θ is the angle of slope between the fillet and the shaft of the tool and d is the diameter of the tool. Note that this definition requires that the tool have a flat bottom, which is a simplification of the cutter described in Chapter 1. Nevertheless this definition still allows the description of the most commonly used cutter types in industry.

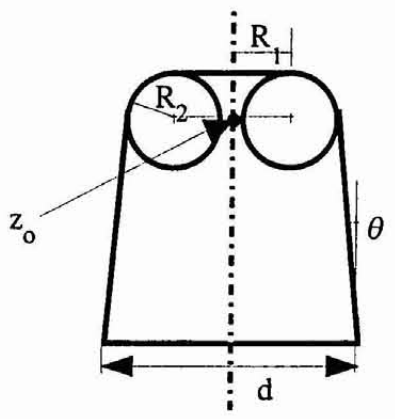


Figure 3.6: Parameters of inverted tool

The intersection points as shown in Figure 3.5 are given by the formulae:

$$z = z_o + R_2 \qquad r \le R_1 \tag{3.5}$$

$$z = z_o + \sqrt{R_2^2 - (r - R_1)^2} \qquad R_1 < r \le R_1 + R_2 \cos\theta \tag{3.6}$$

$$z = z_o + R_2 \sin\theta - \frac{(r - R_1 - R_2 \cos\theta)}{\tan\theta} \qquad R_1 + R_2 \cos\theta < r \le \frac{d}{2}. \tag{3.7}$$

where r is the Euclidean distance between the CC point and the grid line.

3.1.2 Advantageous Properties of the IOM

In addition to the advantages point-based systems have over surface-based systems as described in sub-section 1.2.3, the IOM has some inherent advantages:

- The tool offset surface for a tool of any shape can be generated using the method described in sub-section 3.1.1. The most common cutter types used in industry are catered for using relationships (3.5), (3.6) and (3.7). However appropriate error analysis is required to ensure machining is within tolerance; we give the error analysis for the case of a ball-nosed cutter in sub-section 3.1.3. Also note that [Li and Jerard 1994] highlight the limitations of this method applied to a flat-end cutter.

- The tool offset surface generated by the method is free of gaps and loops [Choi *et al* 1994]. For example the IOM will not cause the gouging of an overhanging region, nor will it gouge a region that has radius of curvature less than the radius of the cutter. Figure 3.7 shows the result of using the IOM on examples of surfaces with these characteristics. This particular feature is highly beneficial since the detection and handling of such regions is an involved task, see for example [Chen and Ravani 1987].

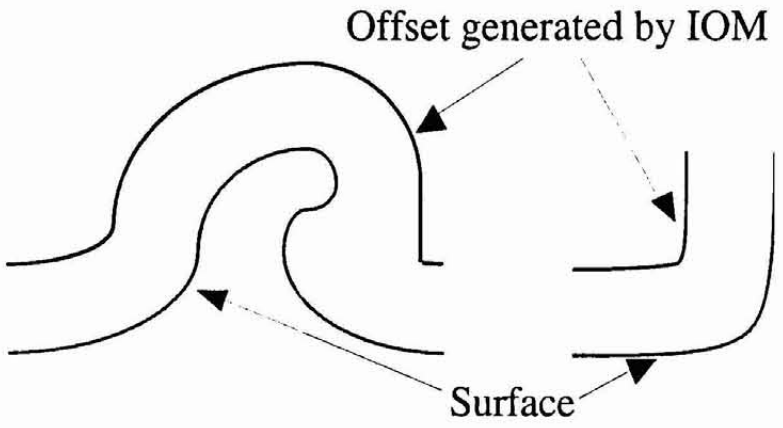


Figure 3.7: Offset generated by IOM for problematic areas

- The topology of the cutter path does not require special attention at the edge of the part. The whole part will be machined provided the offset grid extends beyond the boundary of the desired part by the cutter radius. Figure 3.8 shows the unmachined region that is left if the offset grid does not extend beyond the part boundary. Note that the cutting plane defines the default surface to cut when beyond the part boundary, for a full description of the cutting plane/surface see Chapter 4.

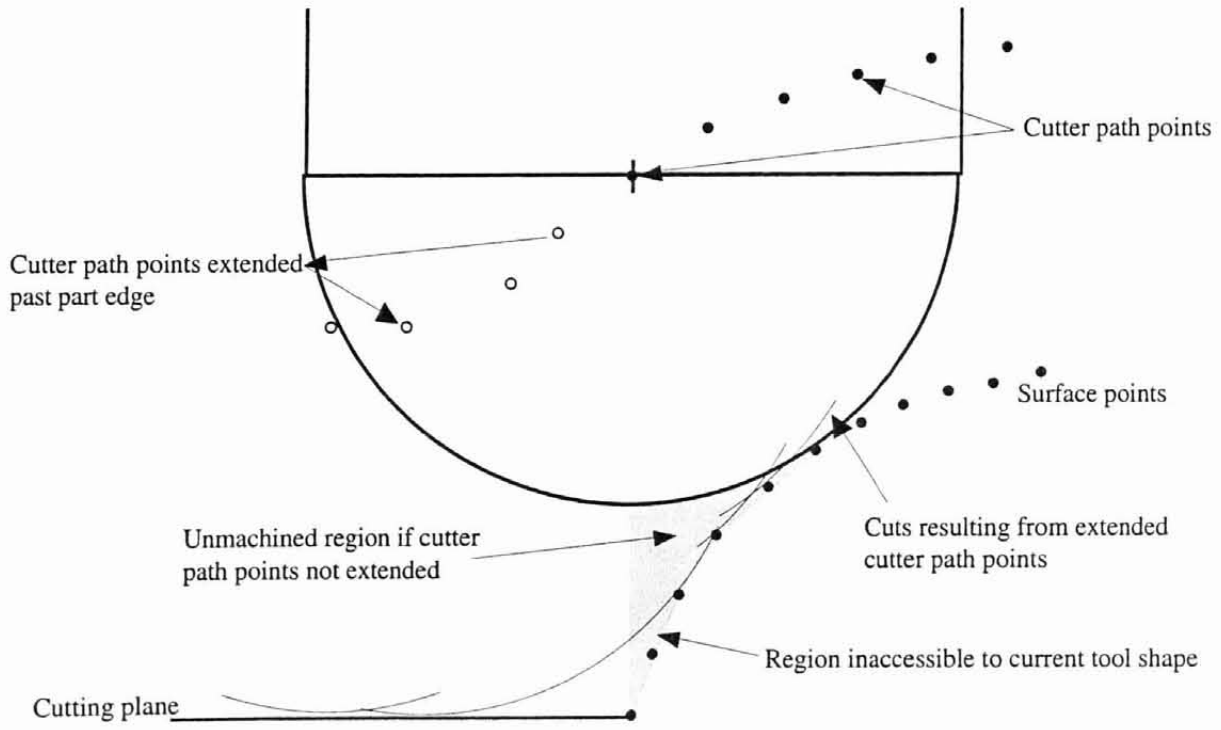


Figure 3.8: Unmachined region due to cutter path not extending past part boundary

- Surfaces defined by sets of points with arbitrary topology can be machined as readily as ones with a rectangular topology. Hence cutter paths can be generated for surfaces defined by nests of sections, triangulations and even scattered data. The only

requirement being that the data is of sufficient density to satisfy the machining tolerances.

3.1.3 Error Analysis

A minimum density of surface points is required to machine a part to some specified tolerance. In this sub-section we analyse the error of the IOM for a ball-nosed cutter in order to find the tolerance of a given set of point data. We then apply this analysis to a specific example, the shoe last, to determine the point density required by the IOM.

When machining there are a number of stages where error is introduced. Two of them are addressed in this sub-section, namely geometric machining error and algorithmic error. The milling machine can use linear, circular or spline interpolation to machine between points in the cutter path. The geometric machining error is the error introduced by this interpolation. We analyse the error caused by the machine linearly interpolating the points as this is the most commonly used method of interpolation. The total machining error would also include the error associated with the physical machining of the part but we do not consider it here.

Algorithmic error is the error caused by the offsetting algorithm not generating points that lie on the exact tool offset surface. The numerical error due to rounding is not considered; however with respect to the data employed in this research the error due to rounding will be less than 0.0001mm. After analysing the geometric machining error we consider the algorithmic error due to the IOM generating cutter paths for a ball-nosed cutter from a set of points. Given 3 points on which the cutter sits either 2 or 3 points will determine the maximum error depending on the arrangement of the points. We define r_t as the radius of the interpolating circle, that is the circle lying on the surface of the tool and passing through those points that lie on the tool whether it be 2 or 3, the example shown in Figure 3.9 represents the case for 3 points. The first case occurs when r_t is greater than the radius of the cutter; in this case just two of the

three points determine the error. The second case, as shown in Figure 3.9, occurs when r_t is smaller than the radius of the cutter.

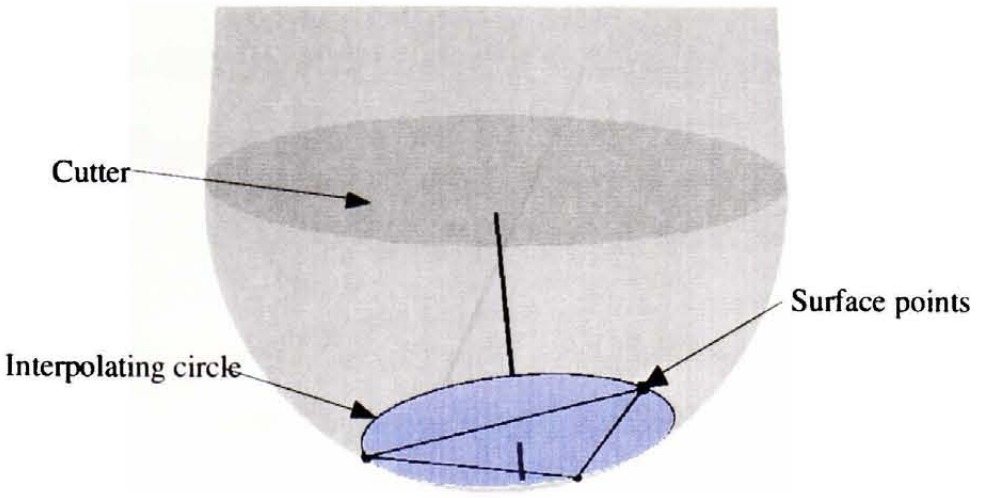


Figure 3.9: Interpolating circle has radius smaller than radius of cutter

In the following analysis the surface is locally approximated by a sphere with radius r_s. We approximate the radius using the method described by [Todd and McLeod 1986] who derive the radius by using an approximation of the Dupin indicatrix at the point in question using 8 nearby points. We also assume the surface is machinable everywhere by a ball-nosed cutter of radius r_c, this implies $r_c \leq r_s$.

We assume the surface is tangent plane continuous and hence for any given set of surface points, the maximum the surface can deviate from these points is bounded by the maximum deviation of the approximating sphere. We also assume the surface points have maximum separation $2s$ and the cutter path points have step forward interval $2c$.

Note that in calculating the geometric machining error the analysis is based on the 3D distance between the cutter path points. We simplify the problem by using the 2D distance between the cutter path points to approximate the 3D distance.

Geometric Machining Error

Suppose an offsetting technique has algorithmic error, e_a. We analyse the case where the surface is convex and the CL points have maximum error below the tool offset surface as shown in Figure 3.10.

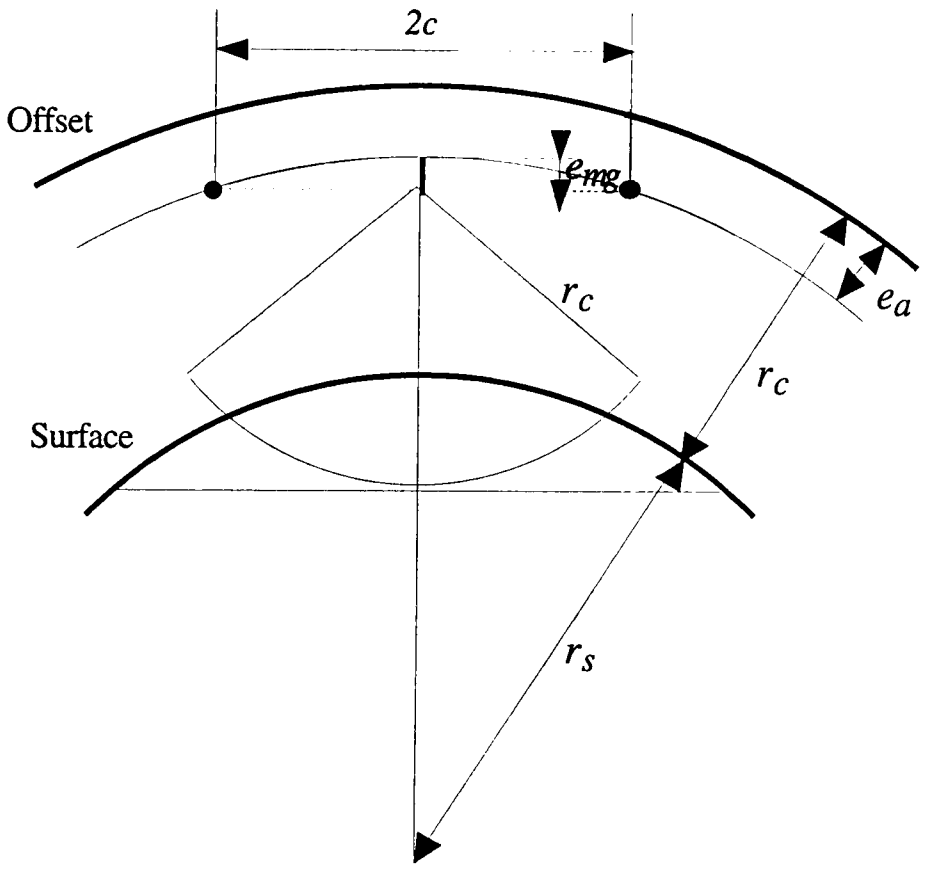


Figure 3.10: Geometric machining error

Figure 3.10 shows the surface, the offset and its error bound. Also shown is the linear interpolation of the milling machine. The gouge that can be caused by linear interpolation is given by:

$$e_{mg} = r_s + r_c - e_a - \sqrt{(r_s + r_c - e_a)^2 - c^2} \tag{3.8}$$

This relation gives the maximum gouging of the surface; the case for maximum undercut is given by:

$$e_{mu} = r_s - r_c + e_a - \sqrt{(r_s - r_c + e_a)^2 - c^2} \tag{3.9}$$

Note: there can be no undercutting with respect to algorithmic error due to the nature of the IOM and the assumption that $r_s \geq r_c$. Hence the overall maximum undercut can be calculated by substituting $e_a = 0$ into equation (3.9).

Algorithmic Error, Case 1: $r_t > r_c$

If r_t is greater than r_c then only two of the three points can touch the sphere in any given configuration. The maximum possible separation of these two points is $2s$ and hence the maximum gouging occurs when the cutter centre lies directly between the two points and the surface is convex, as shown in Figure 3.11. The gouging is indicated by e_{IO2} and given by:

$$e_{IO2} = r_s + r_c - \sqrt{r_s^2 - s^2} - \sqrt{r_c^2 - s^2} \tag{3.10}$$

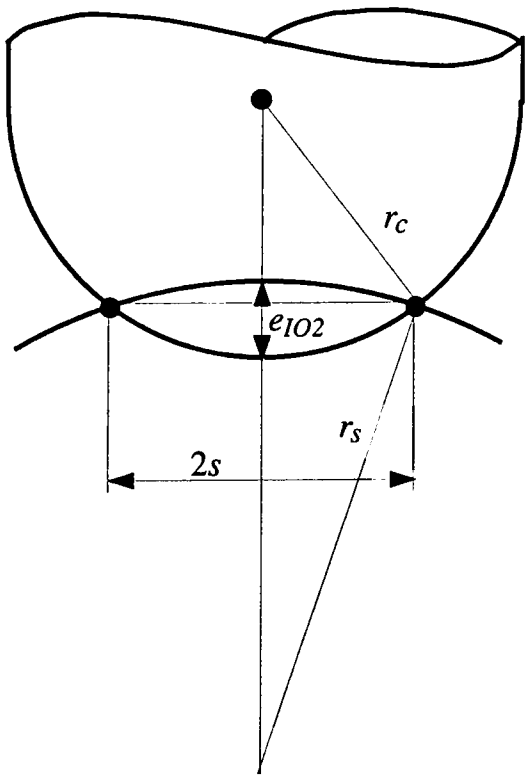


Figure 3.11: Maximum algorithmic error between two points

Algorithmic Error, Case 2: $r_t < r_c$

Given the three surface points, we position the ball-nosed cutter such that the maximum gouging is caused. This occurs when the cutter touches all three points as depicted in Figure 3.9.

The circle interpolating the three points forms a cross-section of the ball-nosed cutter. An upper bound for the gouging is calculated by finding the distance from the cross-section, away from the centre of the cutter, to the edge of the cutter. However by examining a case where this bound leads to exaggerated results, we find a more appropriate measure of the error.

Let **a** and **b** be vectors joining the three points as shown in Figure 3.12. The radius of the circle passing through the three points is given by [Faux and Pratt 1985] as:

$$r_t = \frac{|\mathbf{a}||\mathbf{b}||\mathbf{a}-\mathbf{b}|}{2|\mathbf{a}\wedge\mathbf{b}|} \tag{3.11}$$

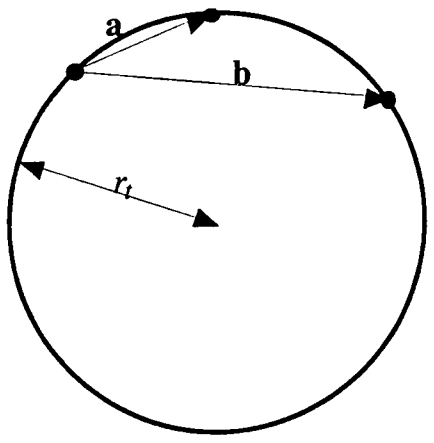


Figure 3.12: Vectors **a** and **b** used to find the radius of the circle

The upper bound for the error can be seen in Figure 3.13 and is given by:

$$e_{IO3} = r_c - \sqrt{r_c^2 - r_t^2} \tag{3.12}$$

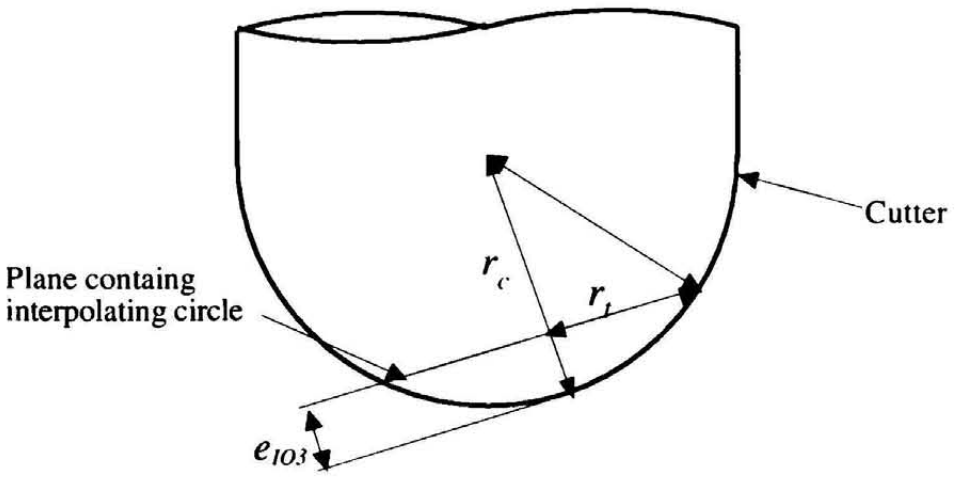


Figure 3.13: Bound of the error

However there is a problem with this measure as the triangle defined by the three points tends away from the centre of the interpolating circle, Figure 3.14 gives an illustration of such an occurrence. The gouge in the region of the triangle is much smaller than that caused at the centre of the cross-section.

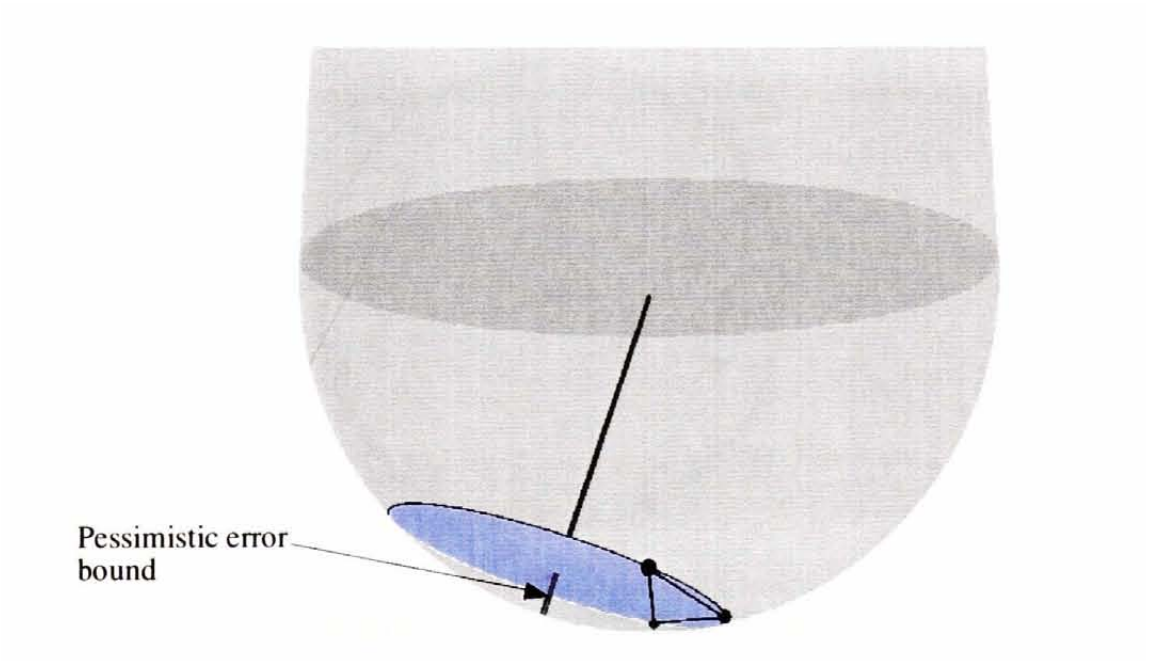


Figure 3.14: Error measured at the centre of the cross-section gives pessimistic results

To find the appropriate error we calculate the greatest gouge caused by the cutter within the boundary of the triangle. At this stage [Drysdale and Jerard 1987] choose to measure the maximum gouge normal to the cutter, see Figure 3.15. However, since it is the triangle that represents the surface and we wish to measure the gouge normal to the surface, an alternative method is presented.

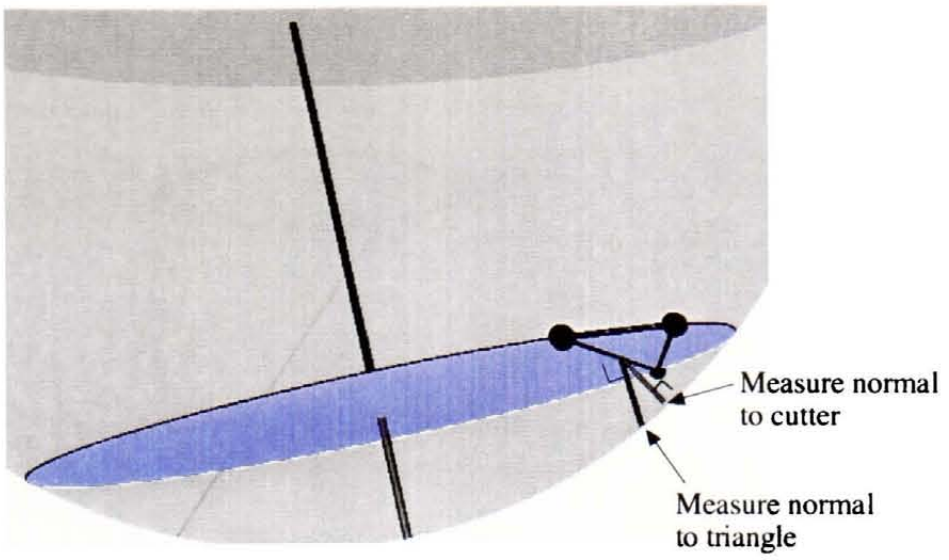


Figure 3.15: Gouge measure normal to the cutter

If the centre of the circular cross-section, **C**, does not lie within the triangle, Δ, then the greatest gouge will be caused at the point on Δ closest to **C**, we call this point **M**. Figure 3.16 shows the plane containing the circular cross-section of radius r_t and the point **M** on Δ.

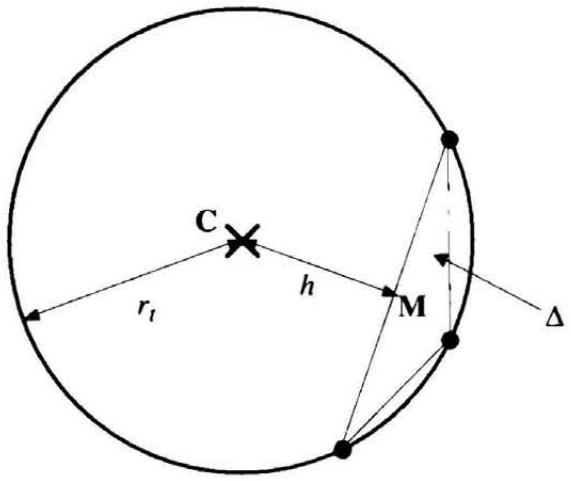


Figure 3.16: Point **M** on the triangle Δ

Figure 3.17 shows the cutter from the side, with gouge e_{IO3} caused in the region of Δ. The modified gouge depth is given by:

$$e_{IO3} = \sqrt{r_c^2 - h^2} - \sqrt{r_c^2 - r_t^2} \tag{3.13}$$

Note that (3.13) is consistent with (3.12) when h=0, i.e. when the projection of the centre of the cutter along the normal to the triangle lies within Δ.

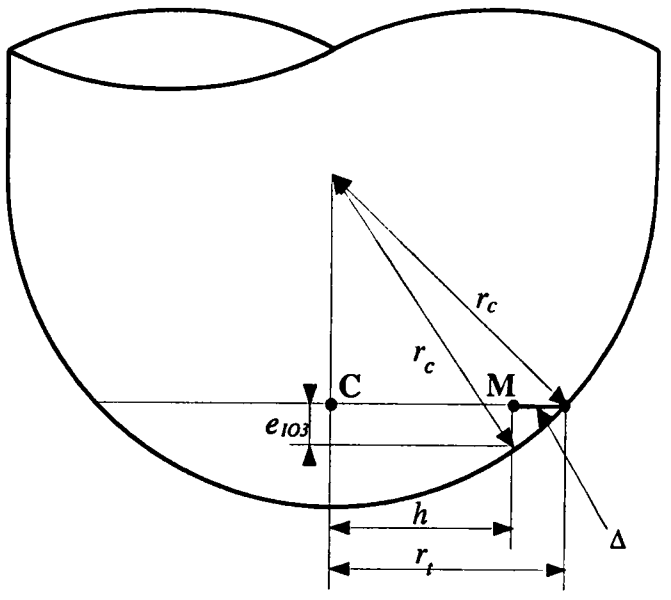


Figure 3.17: Gouge measured normal to the triangle.

Example: Application of error analysis to the shoe last

We now apply equations (3.8), (3.9), (3.10) and (3.13) to the example of the shoe last. Clarks International supplied the shoe last definition and Figure (1.2) shows the surface model defining it. Sampling a grid of 337x211 points at regular parametric intervals from the surface definition generated the point definition as shown Figure 3.18.

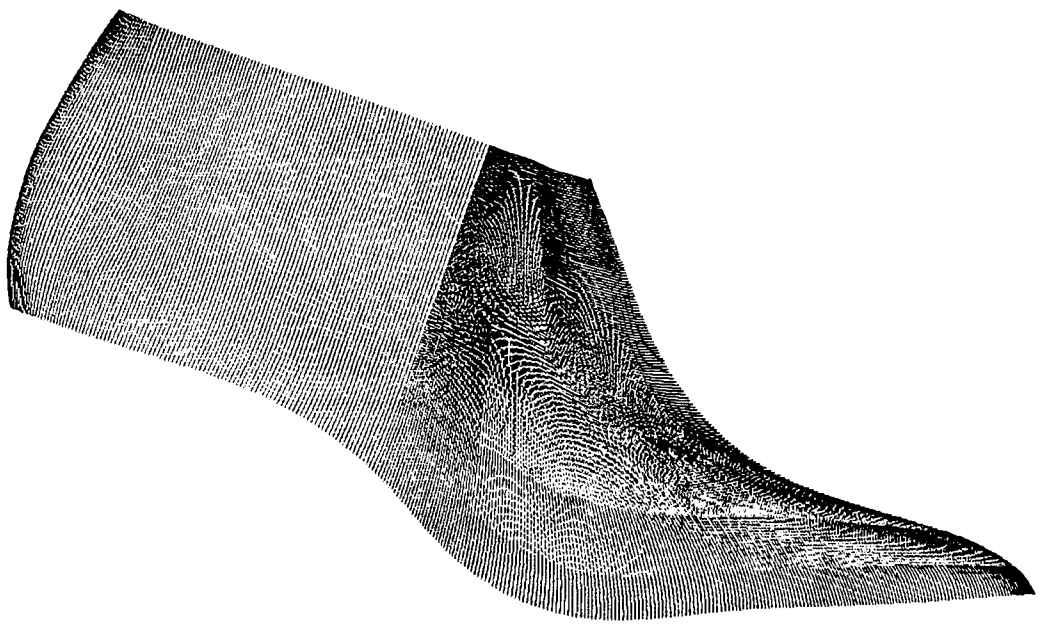


Figure 3.18: The point definition of the shoe last

Note that the point definition shown in Figure (1.7) was generated using an interpolatory subdivision scheme on a characterising set of points. We use points

sampled directly from the surface as it is the quality of the machining algorithm we wish to test, not the subdivision scheme.

To bound the algorithmic error we calculate the maximum gouging of a 5mm radius ball-nosed cutter into the point definition. We locally approximate the part by a sphere of radius 12mm, this is the minimum radius of curvature for the surface and hence gives an upper bound for the error. This value was obtained using the algorithm described by [Todd and McLeod 1986] for estimating principal curvatures.

The error calculated by (3.10) and (3.13) is the maximum a sphere of given radius can gouge the triangulation of the points. Hence the total algorithmic error is given by summing the error due to the cutter and that due to the sphere approximating the surface. The algorithm was implemented using C++ and run on the point definition. It was found that all combinations of three neighbouring points defined circles with radius less than that of the cutter, hence we use (3.13) to find the error due to a ball-nosed cutter of radius 5mm gouging the triangulation of the surface:

e_{IO3}=0.0713 mm.

Error due to a sphere of radius 12mm protruding between the surface triangulation:

e_{IO3}=0.0293 mm.

Hence the total algorithmic error is:

e_{IO}=0.1006 mm.

Using (3.8) and setting c=1mm we calculate the geometric machining error:

e_{mg}=0.0074 mm.

Which gives total maximum gouge:

e_{total}=0.1080 mm.

Although this total error is not below 0.1mm only a small region of the shoe last has radius of curvature 12mm. The minimum radius of curvature on most of the shoe last is greater than 16mm, which leads to a total error below 0.1mm.

The maximum undercut is given by equation (3.9):

$$e_{mu}=0.0455 \text{ mm.}$$

Hence we use this point definition of the shoe last to achieve an overall machining tolerance of 0.1mm.

3.2 Powermill

Powermill is a CAM software package developed by Delcam Ltd. Powermill is used throughout this thesis as the commercial package against which the performances of the implemented algorithms are compared. Powermill operates on triangulations of surfaces hence Duct 5.1, a CAD package also by Delcam Ltd, was used to triangulate the bi-sextic Bézier patch surface model that defines the shoe last. Associated with this triangulation is a tolerance bounding the maximum deviation of the triangles from the surface. After discussions with Delcam it was decided to allow half the overall machining tolerance to be used by the triangulation and half in the generation of the cutter paths.

Although Powermill has a number of alternative finishing patterns, planar cutter paths are used as they are the simplest to implement and allow an accurate comparison of the two parts.

3.3 CMM Results

The results presented in this section were obtained by measuring the two machined shoes using a CMM. The CMM uses a spherical probe of non-zero radius to contact the surface, a 1mm radius probe was used for the research presented here. When the

probe is pushed off-centre an electrical contact is broken and the co-ordinates stored. The tolerances of various set-ups on a CMM were investigated by [Miguel 1996] and for similar parameter values (e.g. stem length, probe radius, 2D/3D linear movement) the tolerances were found to be below 0.01mm, which is typical for most CMMs.

A contouring algorithm was used to sample points automatically on the shoe. The software samples points at some user-defined separation between user-defined start and end points on the shoe. These points must lie in a horizontal plane, i.e. a plane parallel to the XY plane in the CMM co-ordinate system. The shoe was positioned such that it had a similar orientation in the CMM co-ordinate system as in the CNC co-ordinate system. The sampling interval used in each profile was set at 2mm. Profiles were taken in horizontal planes at 5mm intervals.

The problem with the software is that it gives an inaccurate contact point and not the actual contact point in the output data file. The probe operates in a horizontal plane and approximates the contact point in this plane. As can be seen in Figure 3.19 there will generally be a discrepancy between the approximated contact point and the actual contact point. The maximum error this discrepancy can lead to is r_p, where r_p is the radius of the probe.

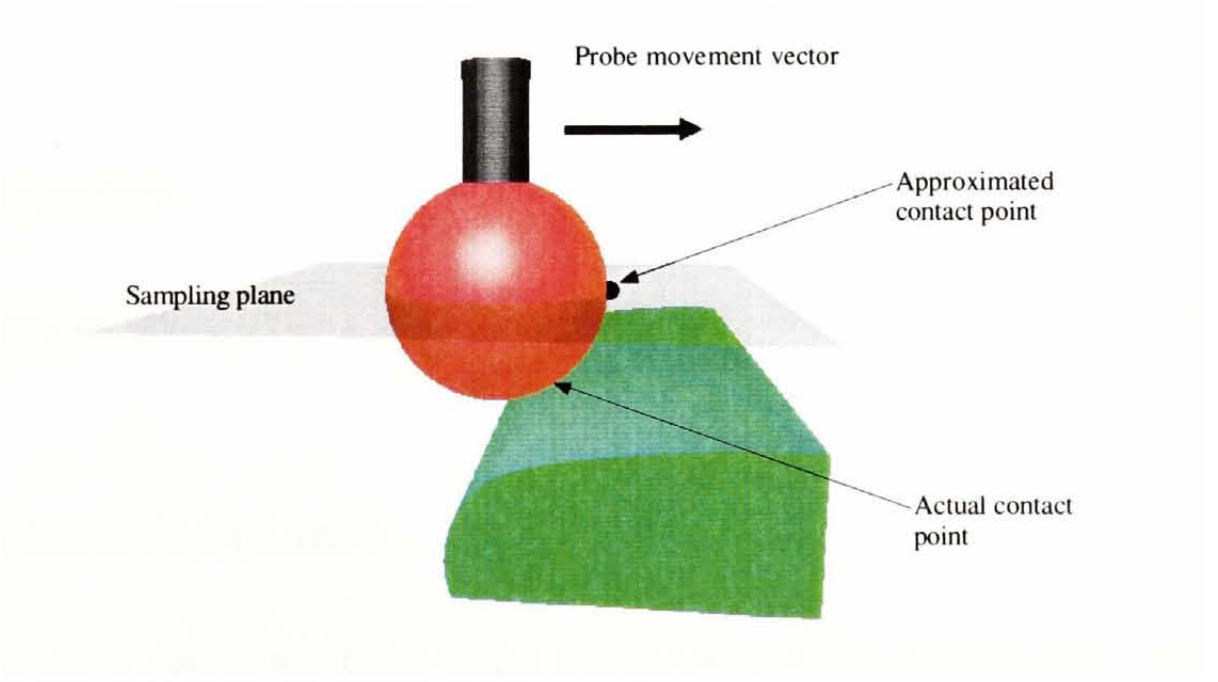


Figure 3.19: Approximated contact point with the surface

To overcome this problem we generate the probe centre. We know this point lies r_p away from the surface being measured and hence when comparing the measured data against the original definition the error is given by:

$$\text{error(sampled points)} = \left|\mathbf{P}_{pc} - \mathbf{P}_{np}\right| - r_p$$

where $\mathbf{P}_{pc}$ is the probe centre and $\mathbf{P}_{np}$ is the point on the surface definition nearest $\mathbf{P}_{pc}$. However, the algorithm does not include an option to output the probe centre and hence we must use a second piece of software to determine $\mathbf{P}_{pc}$.

We now investigate the stages in the measuring process where error could be introduced, Figure 3.20 highlights these different stages.

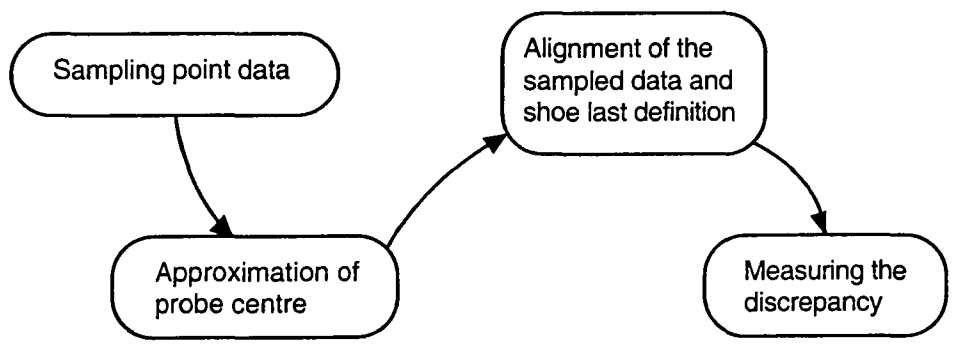


Figure 3.20: Stages at which error can be introduced

Sampling Point Data

The measuring accuracy of the CMM has already been discussed and has error below 0.01mm.

Approximation of Probe Centre

The CMM outputs the probe centre to the computer terminal, however the software used to automatically generate the profile data outputs an approximate contact point. As discussed above this process must then be reversed to regenerate the probe centre.

This involves degradation in the quality of the probe centre data. A bound on the error that this process may introduce is derived in Appendix 1 and is given by:

$$\text{error(approximation of probe centre)} \leq 2.\text{error(approximation of contact point)}$$

In Appendix 1 the error introduced by approximating the contact point is estimated at less than 0.03mm and hence the bound on the error introduced by approximating the probe centre is estimated at less than 0.06mm.

Note that the software used to regenerate the probe centre from the approximated contact points fits a spline. This software requires that boundary data is inputted manually via a graphical interface which may lead to large errors since the data will not be precise. Since the software splines the data it is assumed that the error will be localised to the boundaries.

Alignment of Data

To align the measured data to the shoe last co-ordinate system, the shoe last definition is offset by the probe radius. The probe centre is then aligned to this offset surface using a semi-automatic procedure within CopyCAD, a software package developed by Delcam Ltd. The error resulting from this process is difficult to evaluate and in future a datum will be machined in the workpiece which will remove this stage from the process.

Measuring the Discrepancy

CopyCAD uses a nearest point algorithm to calculate the discrepancy between the measured data and the surface definition but again it is difficult to evaluate the error in this measurement. Discussions with Delcam suggest though, that this error will be below 0.001mm.

The discrepancy plots are given in Figure 3.21a and Figure 3.21b. Each discrepancy has had 1mm subtracted to account for the probe radius. The colours represent different ranges, they are:

Blue	discrepancy $\leq$ 0.1mm
Green	0.1mm < discrepancy $\leq$ 0.2mm
Yellow	0.2mm < discrepancy $\leq$ 0.3mm
Red	0.3mm < discrepancy

Each shoe has some relatively large discrepancies at the boundary points. These are due to the software using the manually minimised boundary conditions. Also note that on the third contour from the top a measurement is missing on both, this is due to the CMM having difficulty locating the surface in this region.

The data measured from the Powermill shoe contains some non-boundary points with discrepancies greater than 0.2mm, if the Powermill shoe is machined to within tolerance this corresponds to a measurement error greater than 0.1mm outside the total machining tolerance. This would imply there is an alignment error of at least:

$$\text{error(alignment)} \geq 0.1 - 0.01 - 0.06 - 0.001 = 0.03\text{mm}$$

However it is not possible to determine whether these errors are due to the alignment procedure, or to out of tolerance cutter paths.

All of the non-boundary points on the IOM shoe lie within tolerance or within the next error band. This is consistent with a well-aligned model and the error analysis performed earlier in this section.

Figure 3.21a: Discrepancy plot of the sampled data from the Powermill shoe

Figure 3.21b: Discrepancy plot of the sampled data from the IOM shoe

3.4 Stylus Results

The previous section gave confidence that the IOM generates cutter paths to within the theoretical tolerances calculated. We now examine the surface finish produced by the IOM and compare it to that of Powermill. The finishing patterns of both are chosen as parallel planar paths of 1mm separation. Taylor-Hobson stylus equipment was used to sample points from the model. The equipment drags a stylus across the surface and the vertical displacement of the stylus is measured by a laser and recorded. The equipment can measure either a single profile (a planar cross-section) of the surface or a nest of profiles to approximate an area of the surface. The height of the surface is recorded at regularly spaced data points and the region is approximated by the linear interpolation of these points.

Two approximately rectangular regions, 12mm x 12mm, were measured from each shoe in relatively flat areas. The regions are required to be relatively flat and near horizontal because the measurement range of the stylus equipment is quite restricted. The number of data points that can be recorded by the equipment is fixed and hence the size of the sampling region dictates the length of the sampling interval. A 12mm x 12mm sampling region requires a sampling interval of 73 microns, which is sufficient to measure cusp heights at 1mm intervals.

To analyse the roughness of the surface it is necessary to approximate a profile by taking a planar cut through the linear interpolation of the data points. To get a fair result six profiles are approximated from each region and analysed. In sub-section 3.4.1 we describe the numerical measure of roughness that is used. In sub-section 3.4.2 the results of the measuring are given.

3.4.1 Description of Roughness Average, R_a

Roughness average, R_a, is defined as the arithmetic mean of the departures of the profile above and below the centre line along the sampling length. The centre line is taken as the arithmetic mean of all the heights as illustrated in Figure 3.22.

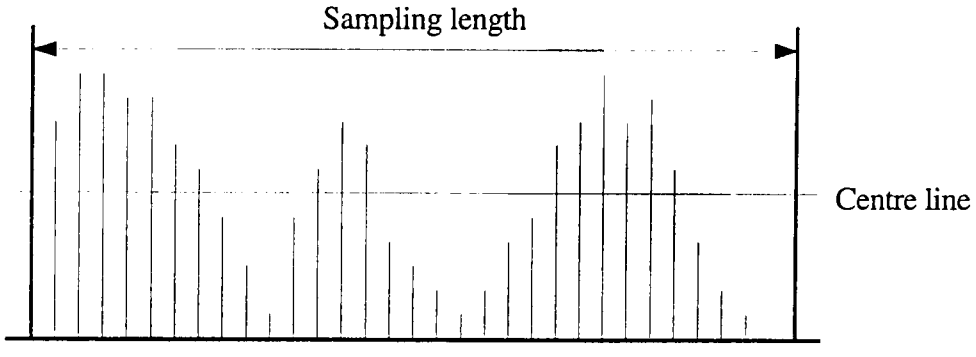


Figure 3.22: Centre line of a profile

The calculation of the average height is illustrated in Figure 3.23 and given by:

$$R_a = \frac{|h_1| + |h_2| + \ldots + |h_n|}{n} \tag{3.14}$$

Hence a smaller R_a indicates less departure from the centre line. Since the surface topology is consistent for all four surfaces, i.e. a series of cusps, a smaller R_a means a smoother surface and hence is desirable.

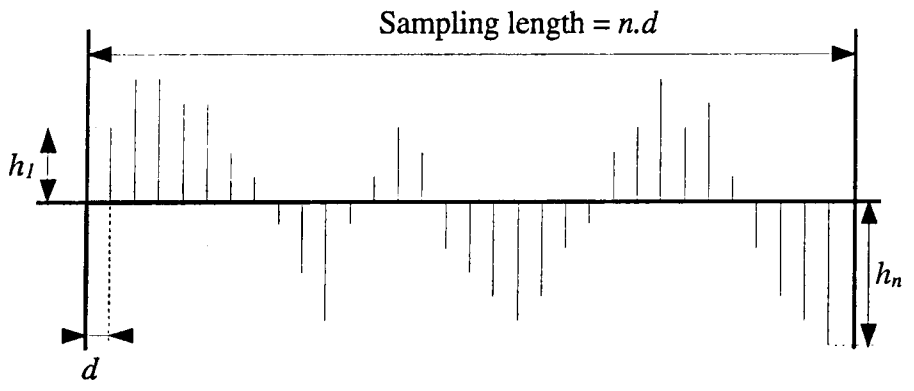


Figure 3.23: Illustration of average height calculation

3.4.2 Results of Measuring with Stylus Equipment

The two regions that were sampled are shown in Figure 3.24. The regions are located in relatively flat and near horizontal areas, as mentioned in section 3.4, this was a requirement of the equipment.

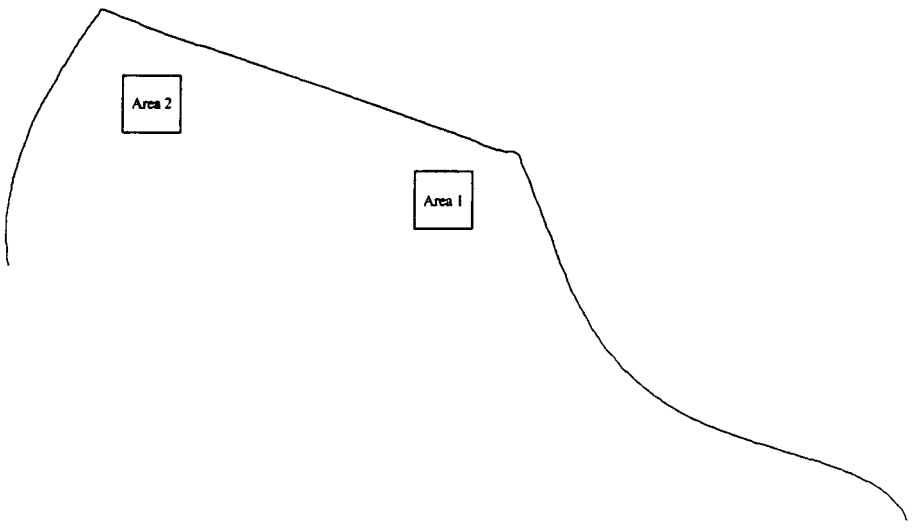


Figure 3.24: Sampled regions from each shoe

The 3D plots of each region are given in Figures 3.24a, 3.24b, 3.24c and 3.24d. Six profiles are taken from each region to analyse and their plots can be found in Appendix 2. Figure 3.26 shows the spacing at which the profiles were taken. To analyse and display the data, software was used to remove the wavelengths greater than 5mm. This value was one of a number of values available ranging from 0.8mm to 5mm. The largest was chosen to retain as much surface detail as possible.

It is difficult to compare the quality of the shoes visually using the 3D plots. However it can be seen that the consistency of the cusps in Area 1 is better than that of Area 2 for both shoes. This is reflected in the profile plots where each profile from Area 1 has greater regularity with respect to the cusp shape than those in Area 2.

Also there appears to be a slight flat spot at the centre of Area 2 and this again is reflected in the respective profiles. At the centre of Profile 3 on IOM-Area2 and right of centre of Profile 4 on PM-Area2, it appears that the cutter has missed the surface. It is difficult to be certain of the reason for this however the anomaly appears on both shoes which may suggest that the cause is geometric in nature.

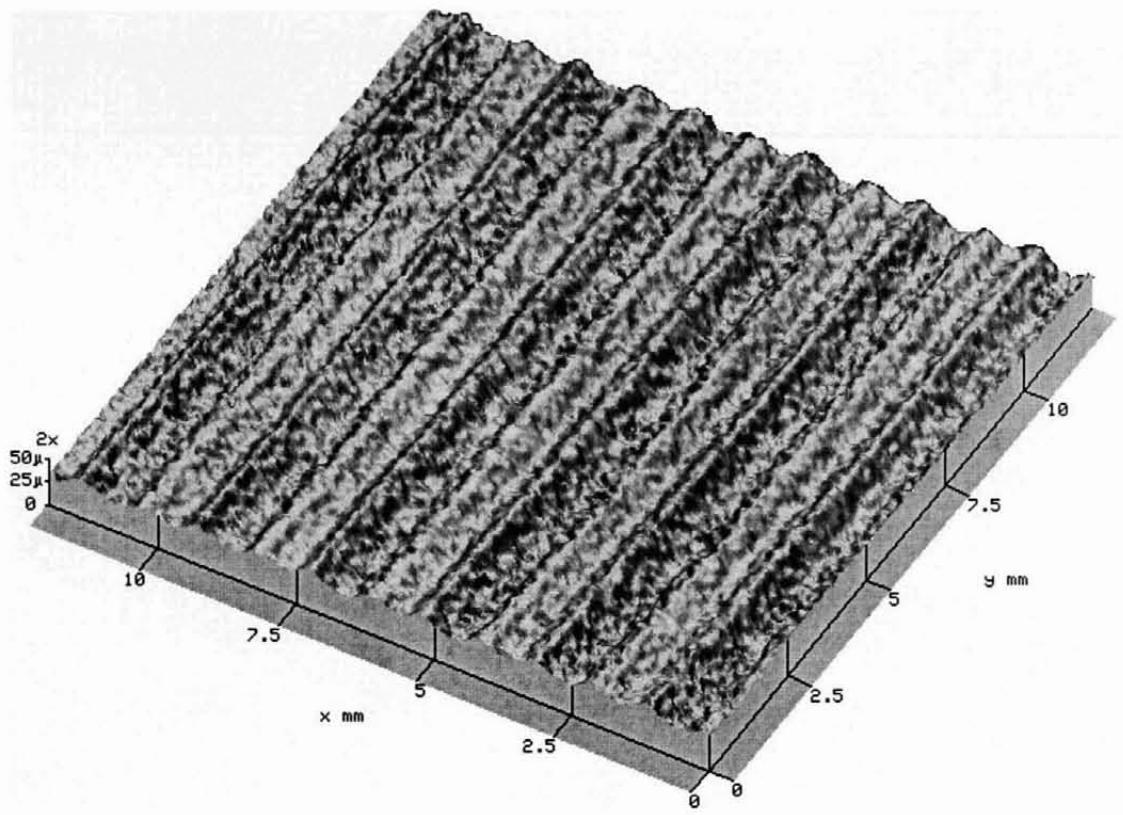


Figure 3.25a: IOM - Area 1

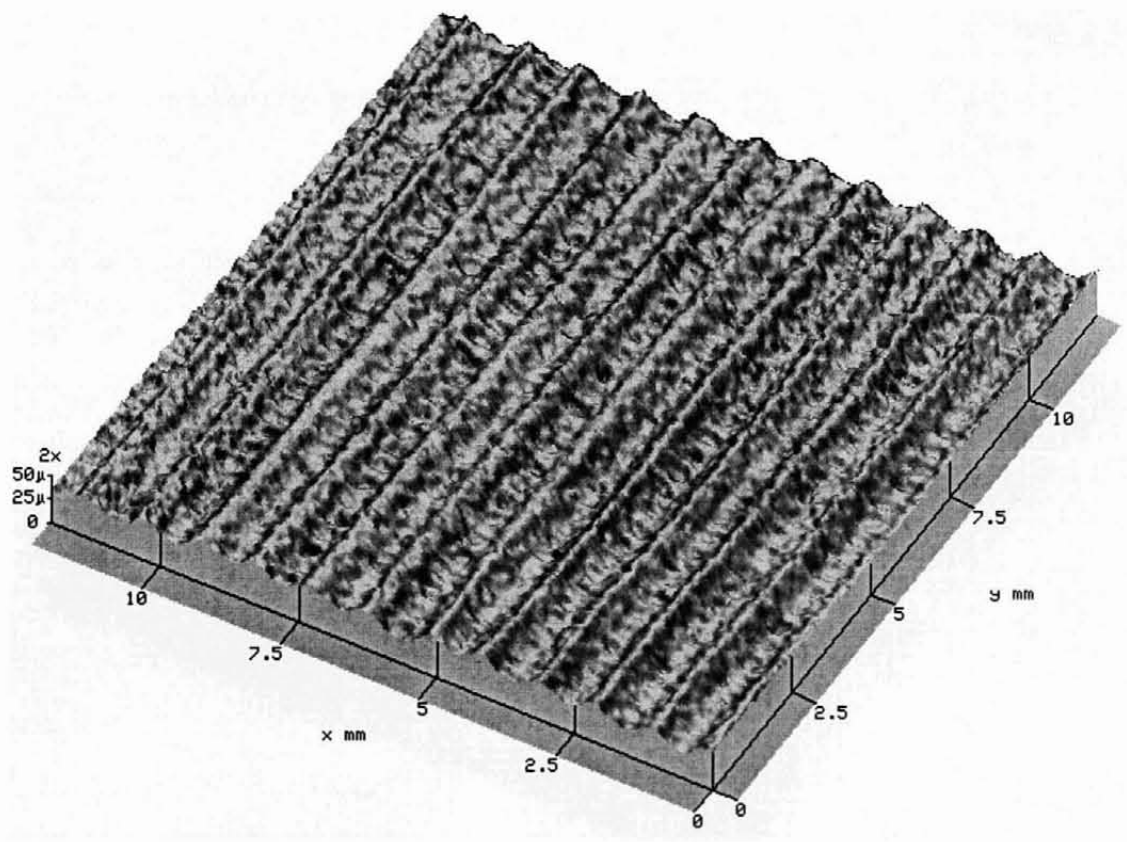


Figure 3.25b: Powermill - Area 1

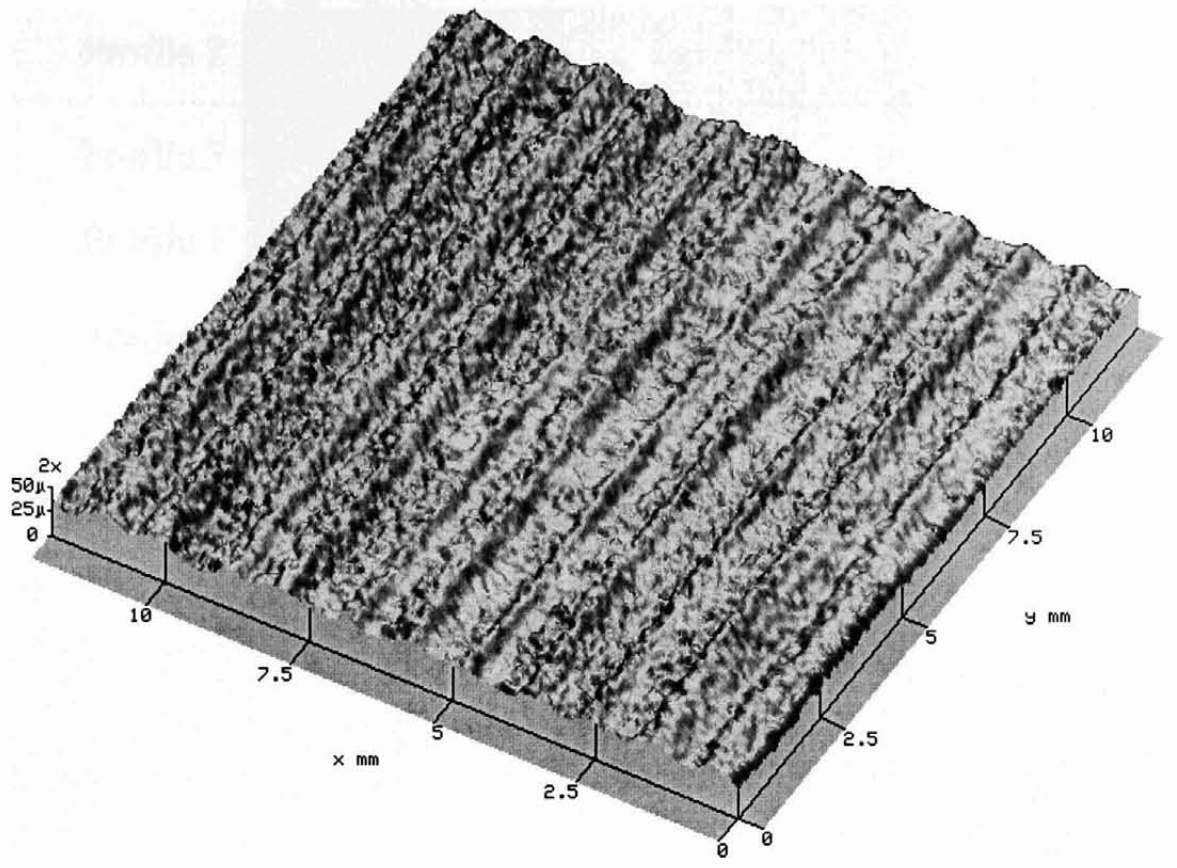


Figure 3.25c: IOM - Area 2

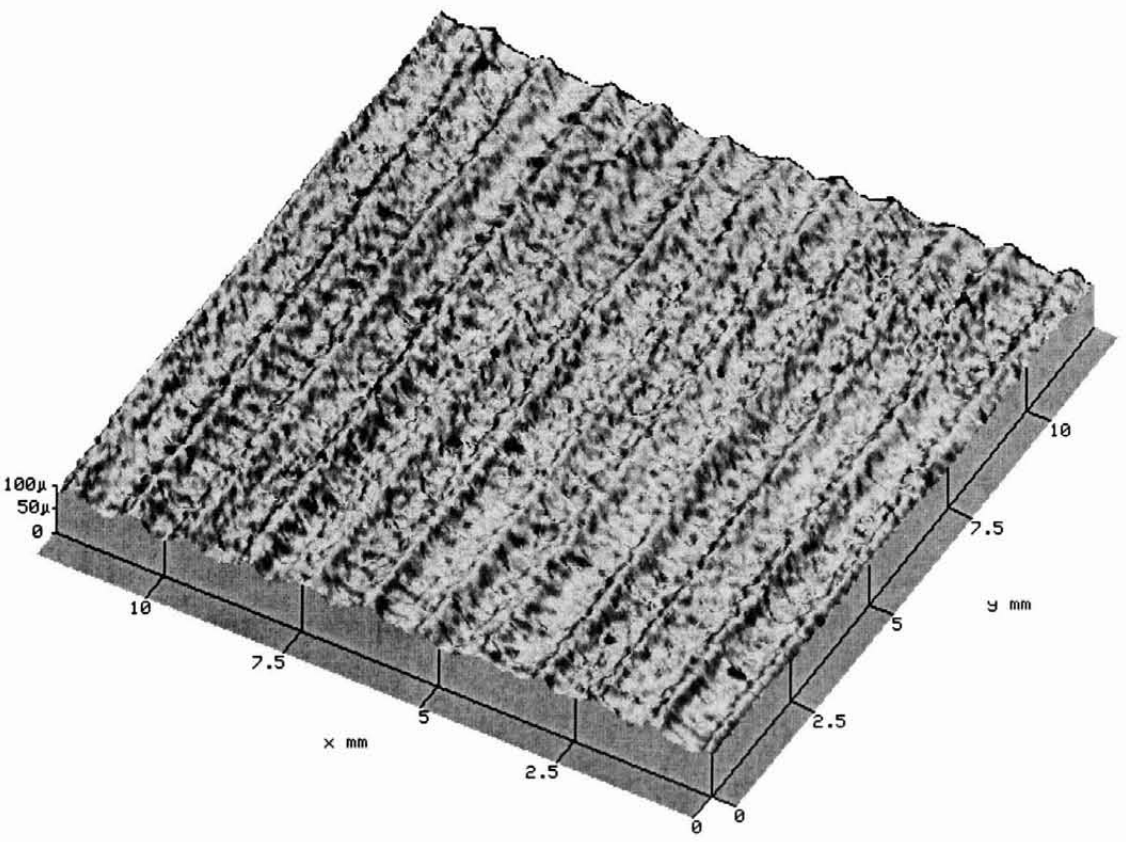


Figure 3.25d: Powermill - Area 2

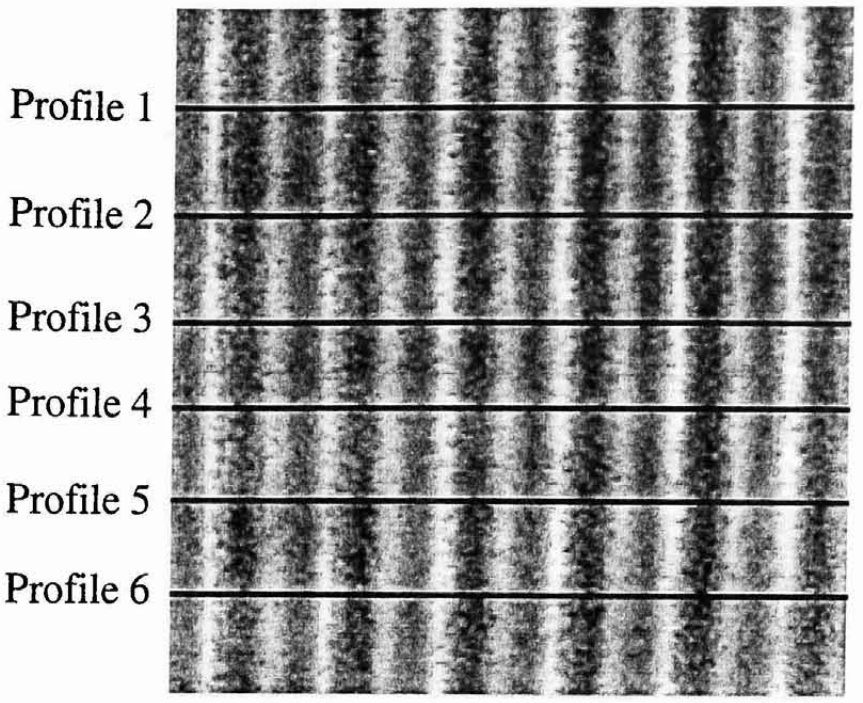


Figure 3.26: Spacing of profiles

[Loney and Ozsoy 1987] give a formula for calculating the pass interval, l, required to machine a plane with cusps of height h. Re-arranging the formula gives:

$$h = r_c - \sqrt{r_c^2 - \frac{l^2}{4}} \tag{3.15}$$

where the radius of the cutter is r_c. Using the respective values for the two machined shoes, r_c=5mm and l=1mm:

h=25 microns

It can be seen from the profile plots that the profiles have average height, from valley to peak, under 30 microns which corresponds quite closely to this theoretical value and gives confidence in the results produced by the stylus equipment.

The R_a value for each profile and the average R_a for each set of profiles can be found in Table 3.1. These R_a values indicate the IOM produces a slightly better finish than Powermill in both regions.

	IOM		Powermill	
	Area 1	Area 2	Area 1	Area 2
Profile 1	7.76	7.07	7.84	9.40
Profile 2	7.49	7.08	7.71	7.54
Profile 3	7.48	7.2	8.34	7.45
Profile 4	7.58	6.39	8.59	7.88
Profile 5	7.44	6.41	8.21	6.43
Profile 6	6.82	6.35	8.60	7.13
Average	7.43	6.75	8.22	7.64

Table 3.1: R_a values from the sampled areas for each shoe

3.5 Timing Results

The time taken by each algorithm to generate the cutter paths, called the generation period, are given in Table 3.2.

The times shown are the average of three measurements. The machining tolerance used for both algorithms was 0.1mm. The computer used to run both algorithms is a PIII 500 MHz PC with 256 Mb of RAM.

Method	Time (secs)
Powermill	11
IOM	1292

Table 3.2: Generation period of the two algorithms

3.6 Discussion of Results

In this chapter we have provided an error analysis of the IOM. The implementation of this algorithm and the subsequent measuring of the machined part confirmed the accuracy of this error analysis. Also, the comparison of the surface finish of the IOM shoe against that of the Powermill shoe demonstrates that the IOM produces satisfactory finishes. Finally, it is highlighted in section 3.5 that the main problem with the IOM is that it has an extensive generation period.

However before investigating this problem we need to address the generation of cutter paths at the boundary of a part. The error analysis contained in this chapter assumes the underlying shape is tangent continuous, but the boundary of a part will generally contain a tangent discontinuity. Machining this region to within tolerance is the subject of Chapter 4.

Chapter 4

The Cliff Problem

In the last chapter we established a method for generating cutter paths directly from points using the IOM. An error analysis was performed that allows the calculation of the theoretical tolerances of the method. We verified the accuracy of this error analysis by measuring a physically machined part. One of the assumptions made in the error analysis was that the part should not contain a tangent discontinuity. However, many parts will contain a cliff-edge around the boundary of the part which is an extreme case of a tangent discontinuity. In this chapter we extend the work presented in Chapter 3 to include an analysis of boundary regions containing a cliff. The strategy is to insert extra CL points into each pass to ensure the path is machined within tolerance in the region of the cliff. The positioning of these extra CL points is determined by the maximum allowable CL point separation.

Note that by inserting CL points we destroy the rectangular topology of the offset grid. However none of the theory presented thus far requires that the definition of the offset grid be so strict and so we now relax this definition to allow the rows of the grid to contain varying numbers of grid points.

We start by giving the definition and alternative point representations of a cliff. The algorithmic and geometric machining errors in the vicinity of the cliff are then calculated. The method that inserts extra CL points is described and then applied to a couple of examples. The chapter concludes with a discussion of possible extensions to this work. It is assumed throughout this chapter that the part is machinable and that the tool is a ball-nosed cutter.

4.1 A Cliff and its Point Representation

We define a cliff as a practically vertical surface on a part the summit of which is a tangent discontinuity referred to as the cliff-edge. In general the part will have a cutting surface associated with it, we assume this is the case. It is required that the cutting surface is distinct from the part definition so that the boundary of the part is defined. A cutting surface is usually a relatively simple shape that extends beyond the part and defines the shape that is to be machined outside the part boundary. Figure 4.1 shows the respective views of a simple part and cutting plane.

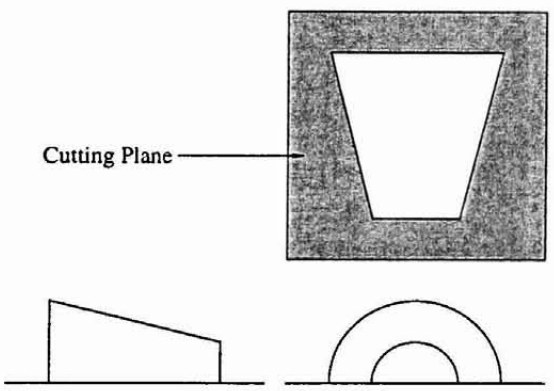


Figure 4.1: A cutting plane

It is assumed that there are no tangent discontinuities within the interior of the part and hence the cliff must lie between the part and the cutting surface. It is also required that the cliff drops down from the part surface and does not go up, this configuration is highlighted in Figure 4.2.

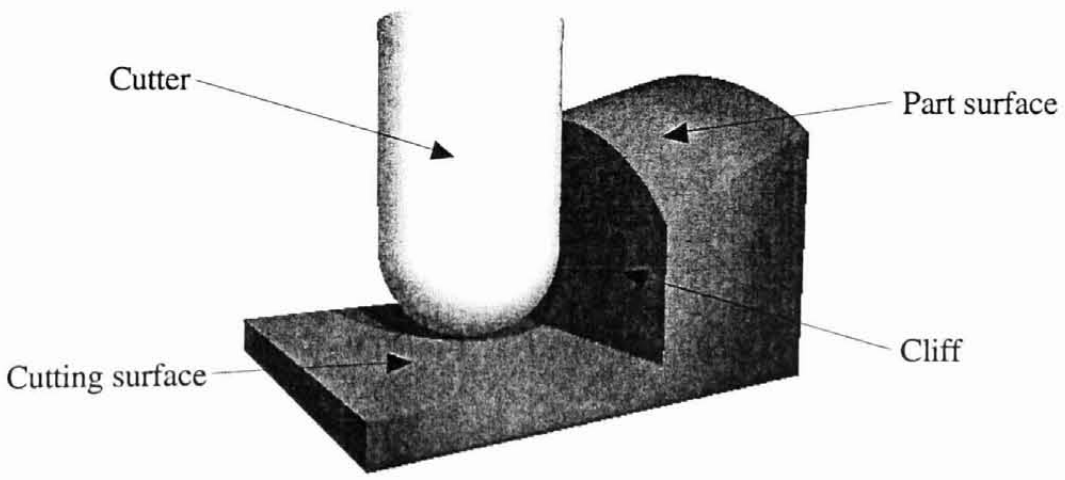


Figure 4.2: A cliff

Figure 4.3 shows some of the different shapes that constitute a cliff, in each case the cliff-edge is circled.

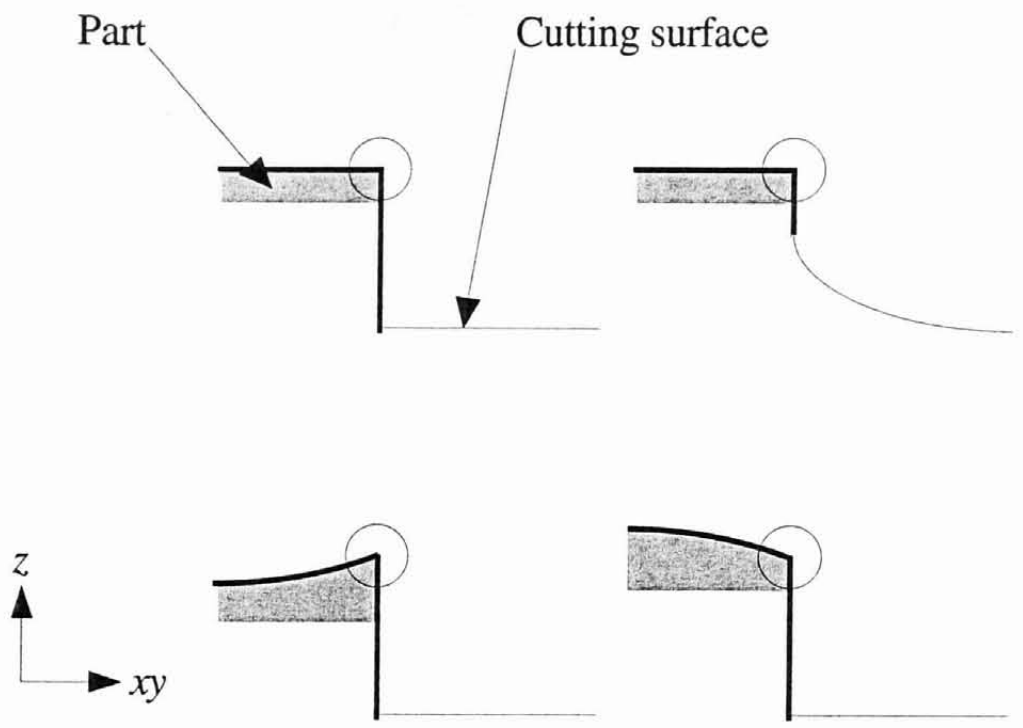


Figure 4.3: Possible cliff shapes

A cliff can be represented by four fundamentally different point combinations. The cliff can have points defining the cliff face or not, and points defining the cliff-edge or not. The part and cutting surface are necessarily defined by points. Figures 4.4b and 4.4c show the four generalised cases for the cliff cross-section shown in Figure 4.4a. The representations in Figure 4.4b represent the cases where the cliff face is defined by points. In both Figures 4.4b and 4.4c the representation on the left has a cliff-edge point indicated by a cross on the cliff-edge, and the one on the right does not.

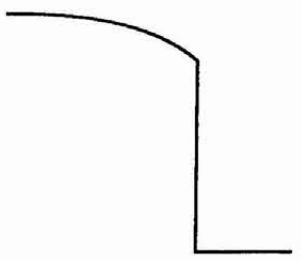


Figure 4.4a: Cliff cross-section

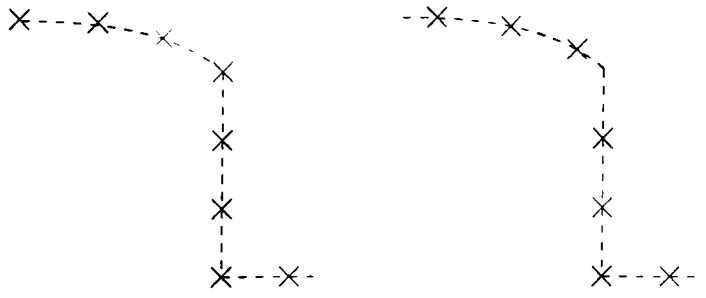


Figure 4.4b: Points lying on cliff face

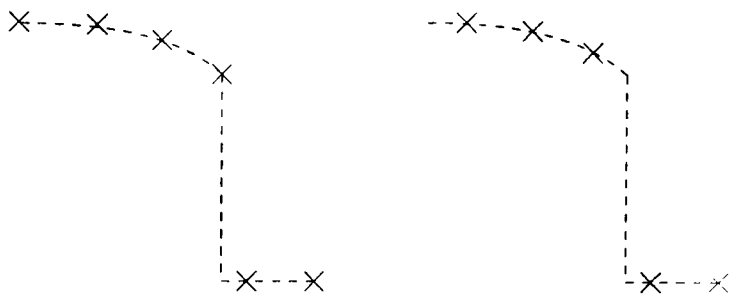


Figure 4.4c: Points not lying on cliff face

In general the discrete definition of a part will include points defining its boundary and so for the error analysis it is assumed that the cliff-edge is defined by points. This reduces the number of cases that need considering to just two. Furthermore, the points lying on the cliff face are made redundant because we are only considering 3-axis machining and the cliff is practically vertical; hence we are left with just one case to consider in the error analysis.

4.2 The Error Analysis

As in Chapter 3, we identify and analyse the two main sources of error, the algorithmic error and the geometric machining error. Note that the two errors are calculated independently of each other and the total error bound is given by their sum.

4.2.1 Cliff Algorithmic Error

Since we are machining from a discrete definition of the part the exact position of the cliff-edge will not generally be known. Hence, as shown in Figure 4.5, the algorithmic error may be as large as the maximum surface point separation, $2s$. The cutter path shown is that as would be generated by the IOM for this point definition.

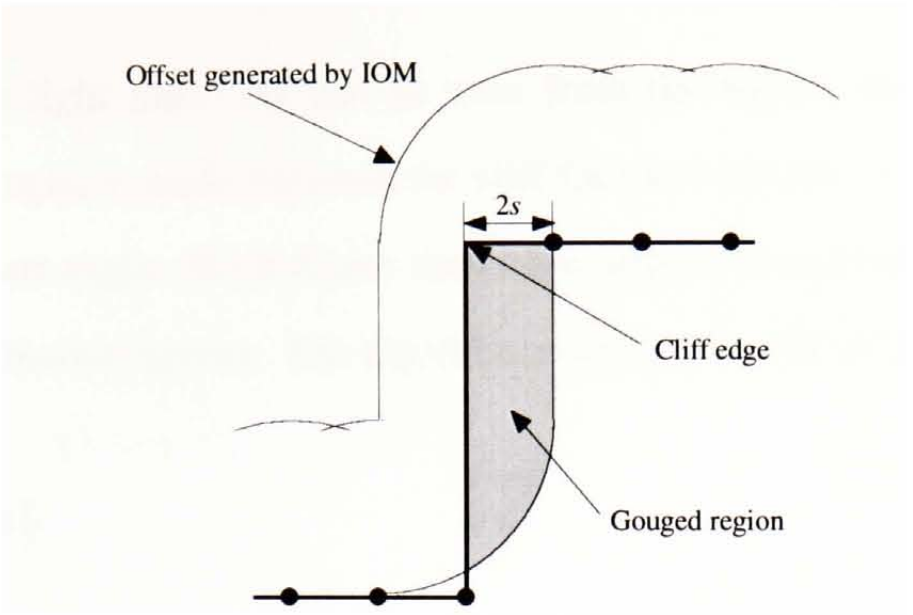


Figure 4.5: Discrete surface definition leading to large error

As stated in section 4.1, the discrete definition of a part will generally include points defining its boundary and so for the rest of this chapter this is assumed. Hence the algorithmic error is restricted to lying between the cliff-edge points, Figure 4.6 highlights the difference this restriction makes.

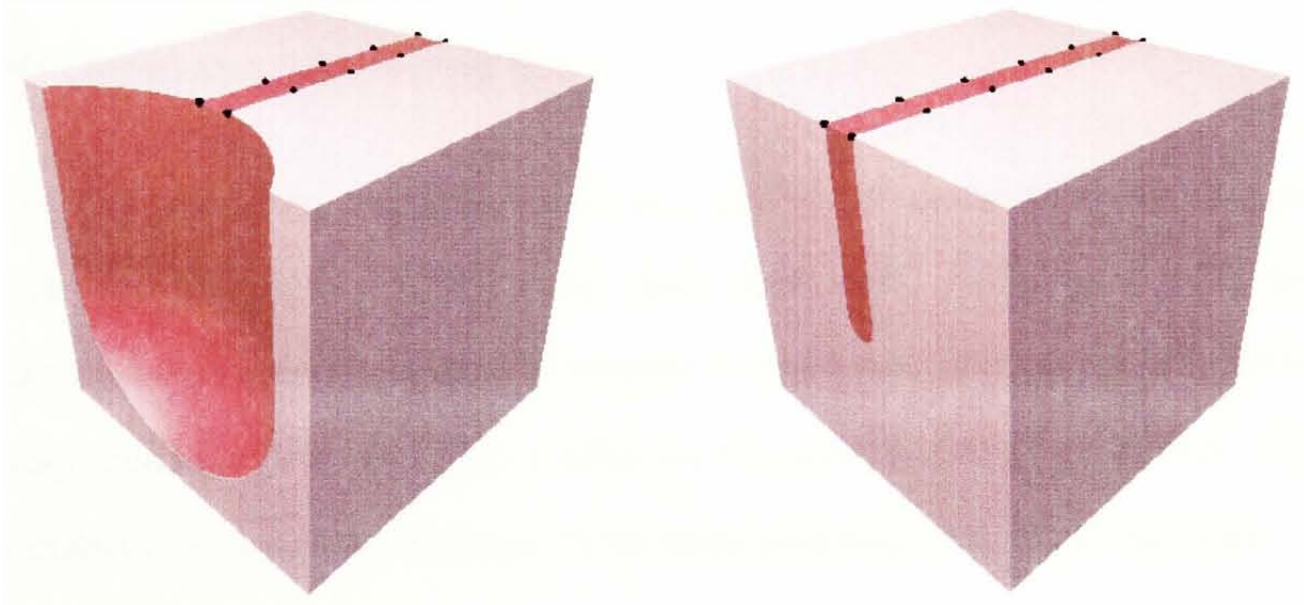


Figure 4.6: Points defining the cliff restrict the algorithmic error

We now calculate the algorithmic error at a cliff as a function of the algorithmic error at a tangent continuous region, i.e. as calculated in sub-section 3.1.3. Figure 4.7 shows the cross-sections of three different cliffs, the part surface is represented by the bold line and the maximum gouge that can be caused by algorithmic error is

represented by the light line. As can be seen from the figure, the error, e_{ca}, is dependent on the angle, γ, made between the cliff face and the part, we refer to this angle as the cliff-part angle. Each figure shows the respective algorithmic error, e_a, for the tangent continuous regions. The algorithmic error at the cliff is given by:

$$e_{ca} = 2e_a \cos\tfrac{\gamma}{2} \tag{4.1}$$

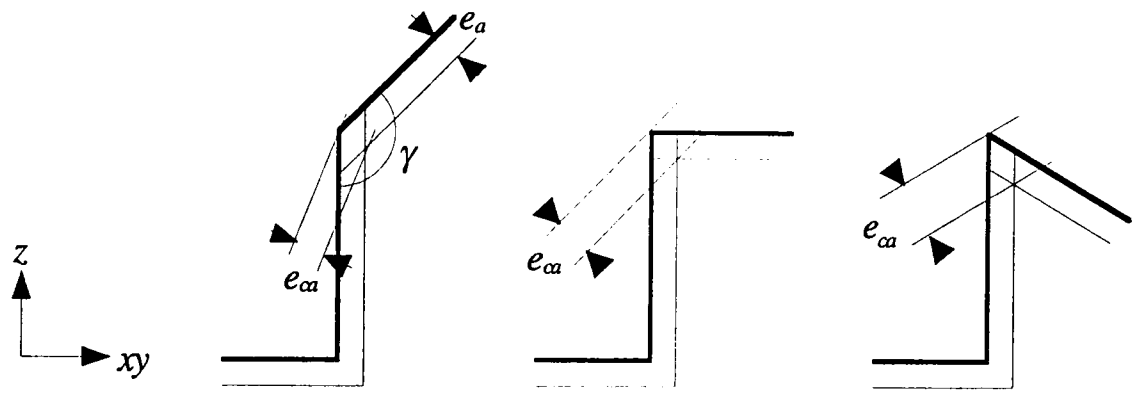


Figure 4.7: The algorithmic error at a cliff-edge

4.2.2 Cliff Geometric Machining Error

To calculate the geometric machining error at a cliff we find the maximum gouge caused by the discrete approximation of the exact cutter path. In doing so it will be useful to define a number of regions with respect to horizontal distance, $d_{hz} \geq 0$, from the part shape because in each region a different CL point spacing is required. By horizontal distance we mean the distance in the plane perpendicular to the tool-axis.

The outer region lies above the cutting surface and is further than the radius of the cutter away from the part boundary. The surface in this region is tangent continuous and hence is consistent with the assumptions made for the analysis performed in sub-section 3.1.3.

The local region also lies above the cutting surface and is less than r_c away from the part. In this region the cutter negotiates the cliff-edge and hence needs analysing.

The local region is separated from the outer region by the frontier which is the 2D offset of the part boundary by distance r_c. The CL points that straddle the frontier form the endpoints of the transition movement. The transition movement brings the cutter into contact with the part. This is a key movement and requires a separate error analysis to the other movements in the local region.

Finally the area above the part is called the inner region. The interior of the part is assumed to be tangent continuous and hence is consistent with the analysis performed in sub-section 3.1.3. These different regions are illustrated on an example in Figure 4.8. To summarise:

$d_{hz} = 0$	inner region,
$0 < d_{hz} < r_c$	local region,
$d_{hz} = r_c$	frontier,
$r_c < d_{hz}$	outer region,

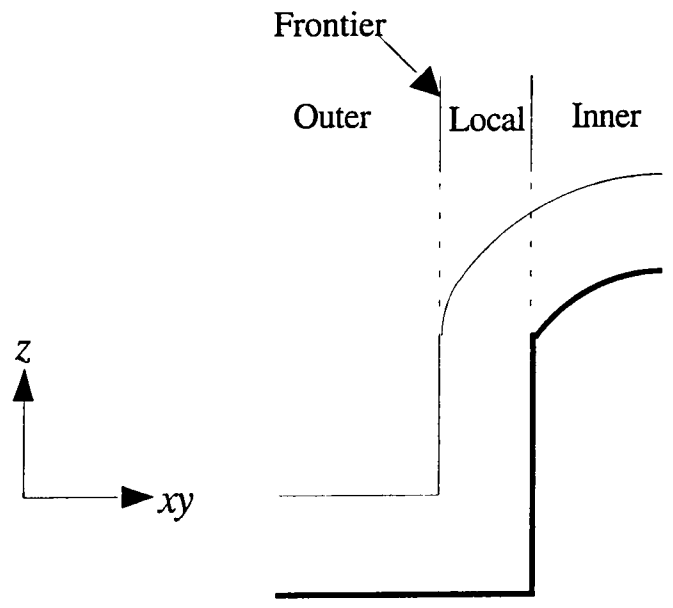


Figure 4.8: The different types of CL points

The Transition Movement Error

The current method of generating cutter paths from regularly spaced CL points (c.f. sub-section 3.1.3) will generally lead to excessive gouging of the cliff as can be seen in Figure 4.9. The figure demonstrates that most of the gouging is caused by the

transition movement of the tool as it crosses the frontier. The maximum gouge this movement can cause is now calculated.

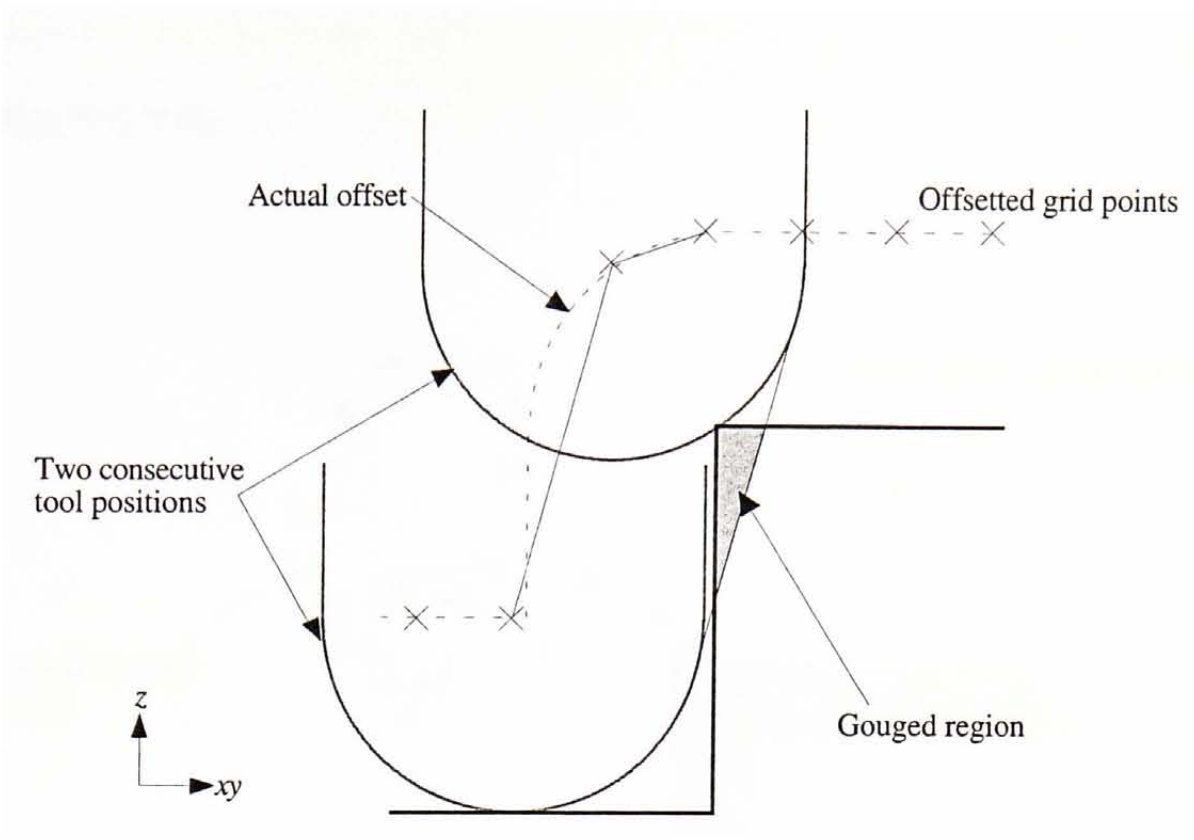


Figure 4.9: Transition movement leading to large gouge

The gouge is measured as the maximum discrepancy between the surface and its nearest point on the machined surface. Let h_1 be the cliff height and c_{hz} the horizontal separation of the CL points terminating the transition movement. It is assumed that the cliff height is greater than the cutter radius, i.e. $h_1 \geq r_c$. Figure 4.9a shows the cliff and the error, e_{cm1}. As can be seen from the figure, the maximum gouging occurs when the cutter centre passes the intersection of the transition movement and the line at angle θ to the horizontal. It is also evident from the figure that as $h_1 \rightarrow \infty$ the gouge increases to its supremum, c_{hz}. From Figure 4.9a the following expressions can be determined:

$$h_2 = \sqrt{r_c^2 - (r_c - c_{hz})^2} \tag{4.2}$$

$$\tan\theta = \frac{c_{hz}}{h_1 - r_c + h_2} \tag{4.3}$$

$$a_1 = \frac{c_{hz}\left(h_1 - r_c\right)}{h_1 - r_c + h_2} \tag{4.4}$$

$$a_2 = \left(r_c - a_1\right)\cos\theta \tag{4.5}$$

$$e_{cm1} = r_c - a_2 \tag{4.6}$$

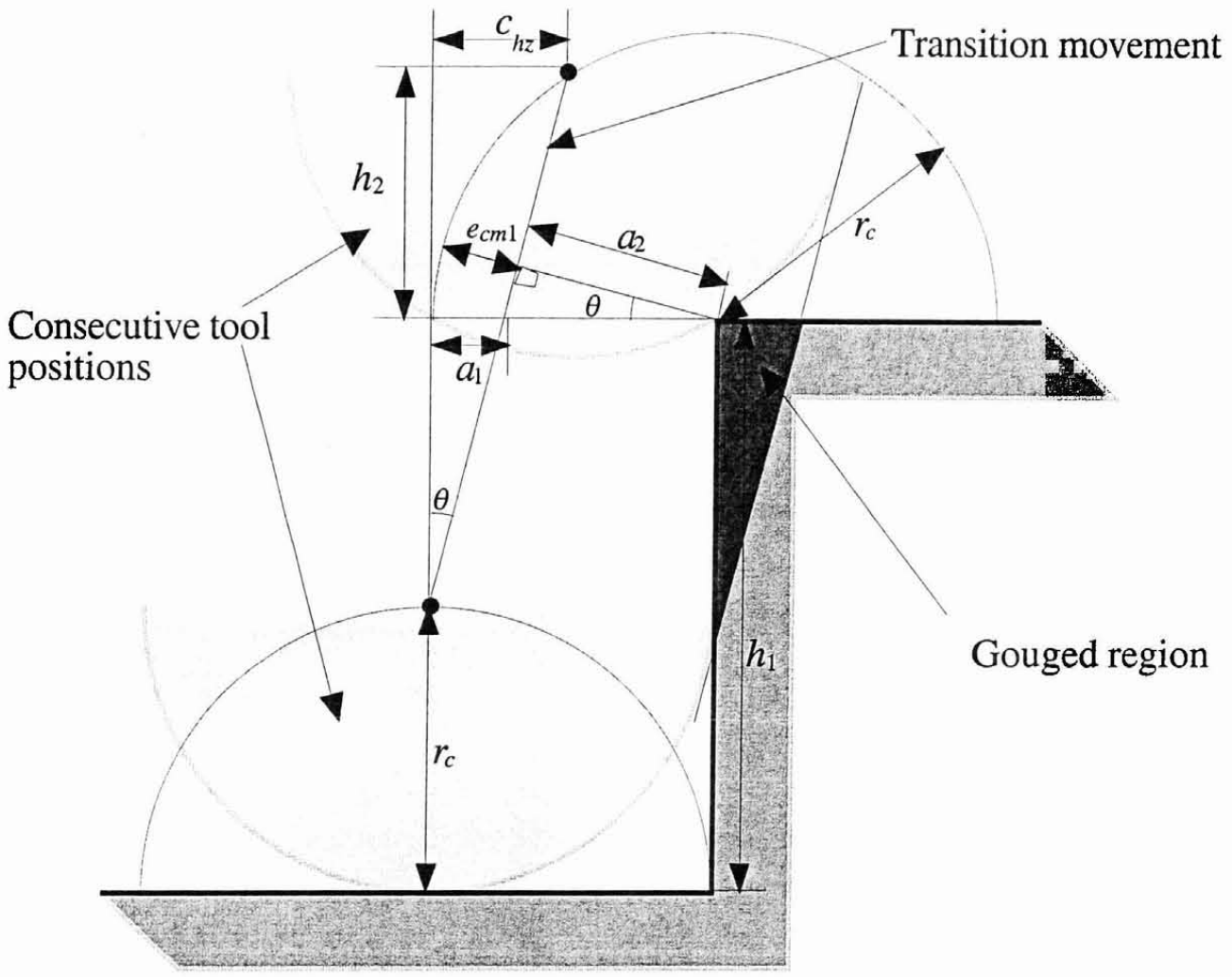


Figure 4.9a: Error due to the transition movement

Substituting (4.3) into (4.4) and (4.4) into (4.6) gives the result:

$$e_{cm1} = r_c + \left(-r_c + \frac{c_{hz}\left(h_1 - r_c\right)}{h_1 - r_c + h_2} \right)\cos\theta . \tag{4.7}$$

Note that equation (4.7) verifies the limit observed from the diagram; as $h_1 \to \infty$, $\theta \to 0$ and $e_{cm1} \to c_{hz}$ from below which gives the useful bound:

$$e_{cm1} < c_{hz}. \tag{4.8}$$

In general e_{cm1} will be much greater than the overall tolerance and hence to satisfy a given tolerance an appropriate horizontal CL point spacing is given by relation (4.8).

To ensure the transition movement does not gouge the cliff beyond tolerance its horizontal length must not be greater than c_{hz} as calculated by equation (4.8). Hence we place a CL point in the outer region, just beyond the frontier and another in the local region with separation from the first less than c_{hz}.

However this does not yet guarantee that the cliff is machined to within tolerance, the reason for this is that the local CL point may not be close enough to the cliff to be offset by the cliff-edge points. Figure 4.10 shows an example of such an occurrence on a cliff as viewed from above.

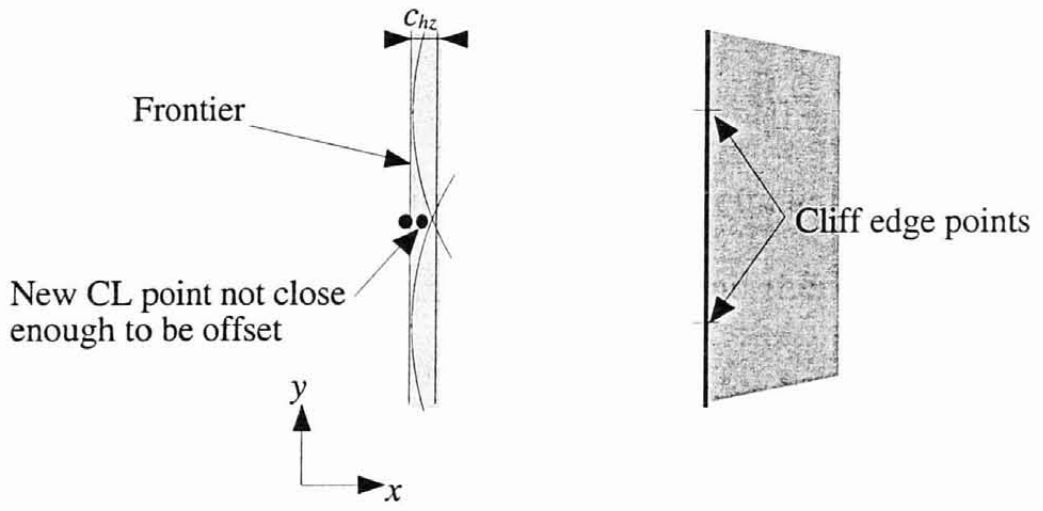


Figure 4.10: Frontier CL point not offset by cliff-edge points

To remedy this problem we require that the local CL point is placed such that its horizontal distance from the cliff is bounded by:

$$r_c - c_{hz} < d_{hz} < r_c - e_{IO2}, \tag{4.10}$$

where e_{IO2} is as calculated from sub-section 3.1.3. This ensures the CL point is offset because e_{IO2} is the maximum distance that the offsetting procedure can deviate from

the actual offset. Note that if $e_{IO2} > c_{hz}$ then the boundary of the part is insufficiently defined and more cliff point edge points are required.

Local Movement Error

We now determine the gouging caused by the interpolation of the local CL points. Figure 4.11 shows the error, e_{cm2}, caused by local movement that is given by:

$$e_{cm2} = r_c - \sqrt{r_c^2 - c^2}\,. \tag{4.11}$$

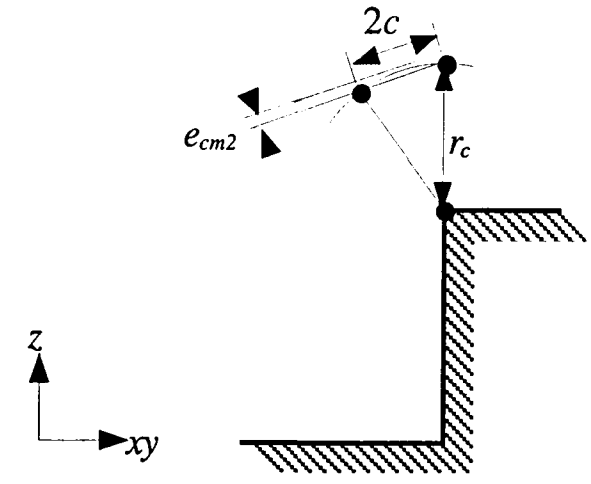


Figure 4.11: The error caused by the linear interpolation of two local CL points

If the current CL point spacing results in too large an error then a sufficiently small spacing to machine within tolerance is given by:

$$2c = 2\sqrt{r_c^2 - \left(r_c - e_{cm2}\right)^2}\,. \tag{4.12}$$

Although the geometric machining error caused by the interpolation of local CL points will always be larger than that caused by inner CL points, the total error will generally be smaller since the algorithmic error is smaller. This is because the algorithmic error in the local region is restricted by two points which are constrained to lie on the cliff-edge and the maximum gouge that can be caused between two points is less than can be caused between three points. See algorithmic error, cases 1 and 2 in sub-section 3.1.3 for the details.

4.3 Solving the Cliff Problem

We now detail the method for inserting CL points into the cutter paths to machine the cliffs to within a given tolerance. Suppose we have overall tolerance τ, we find the geometric machining error by subtracting the algorithmic error from τ. For each pass in the cutter path we insert a CL point just outside the frontier and another in the inner region such that relation (4.10) is satisfied. Extra CL points are inserted into the local region such that relation (4.12) is satisfied. Figure 4.12 shows a single pass on an example part and the regions into which the new CL points are placed. The red points terminate the transition movement and the white points represent the local CL points.

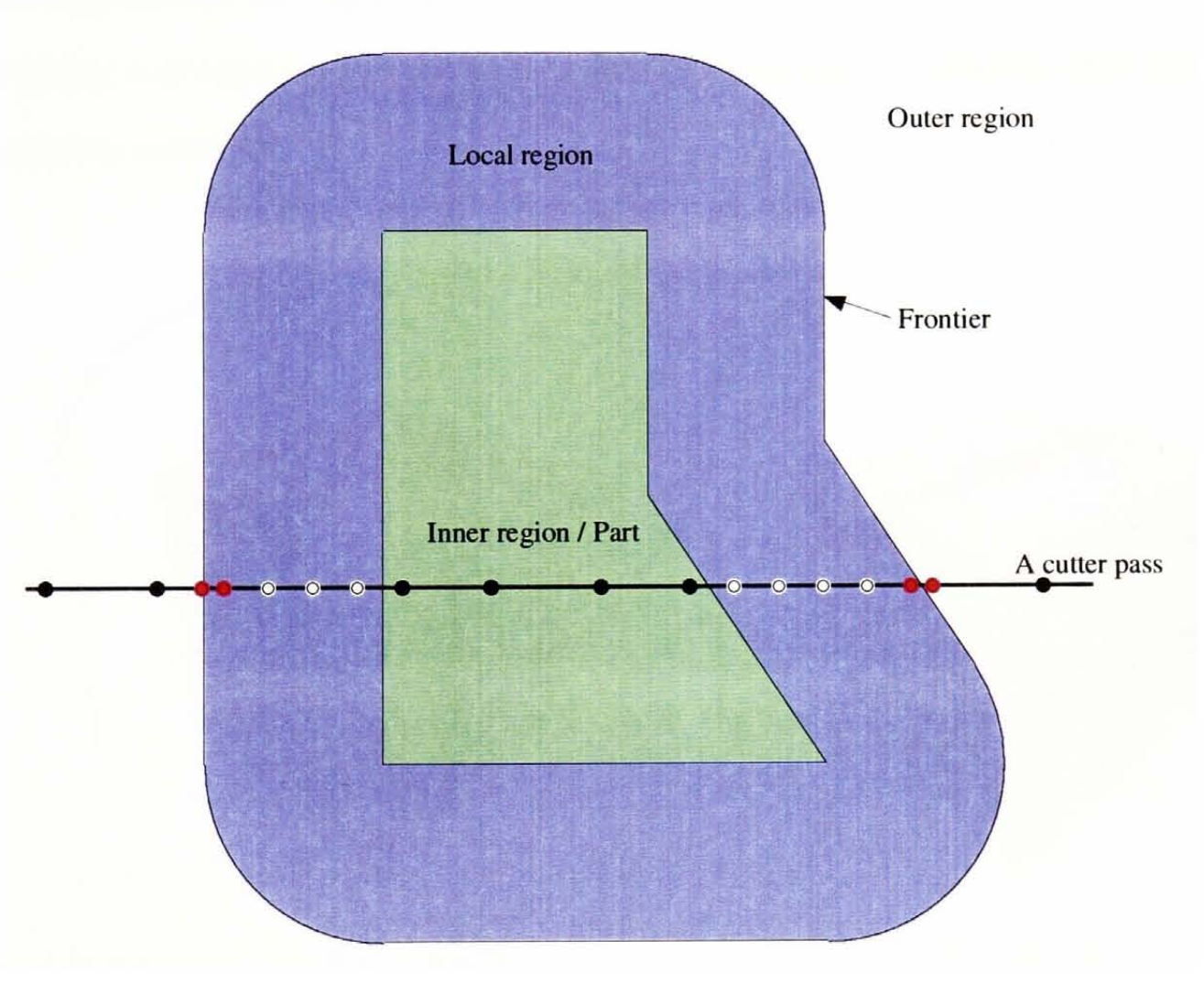


Figure 4.12: Placement of new CL points to machine cliff to within tolerance

4.4 Application of Error Analysis to Examples

We first show graphically the effectiveness of the above analysis on a simple example of a cliff. Then we apply the analysis to the shoe last to get an idea of the appropriate spacing values on an actual example.

4.4.1 A Simple Cliff

Figure 4.13a shows the example cliff as would be machined using the offset grid with regularly spaced points. The blue curve represents the gouge-free cutter path and the black dots joined by black line-segments represent the generated cutter path. It can be seen in the figure that the transition movement leads to a much larger gouge than that produced by the modified cutter path as shown in Figure 4.13b. Note that the local movement adjacent to the transition movement now causes the largest gouge in Figure 4.13b. This movement is slightly out of tolerance due to the simplification made in sub-section 3.1.3 and is because the horizontal separation of the CL points is set equal to the appropriate 3D separation, i.e. slightly larger than it should be. Note that by generating a circular arc in such regions the cliff geometric machining error would be completely removed.

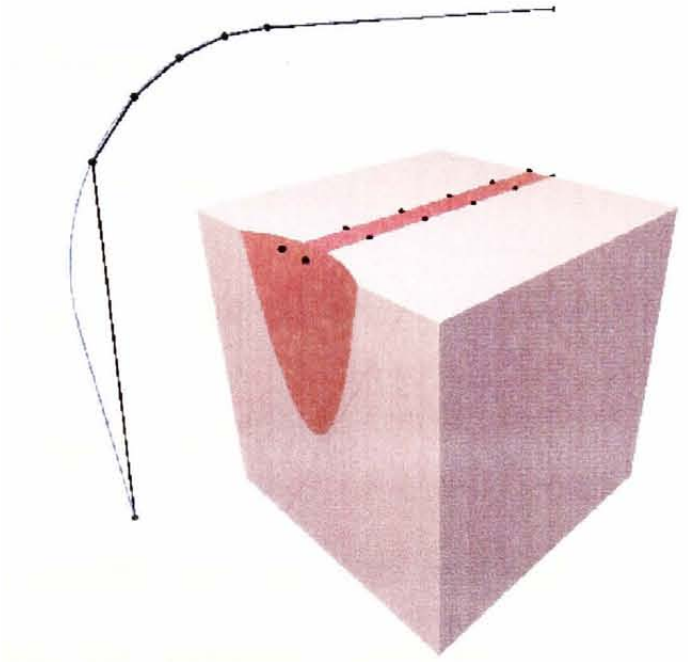


Figure 4.13a: Regular spaced CL points

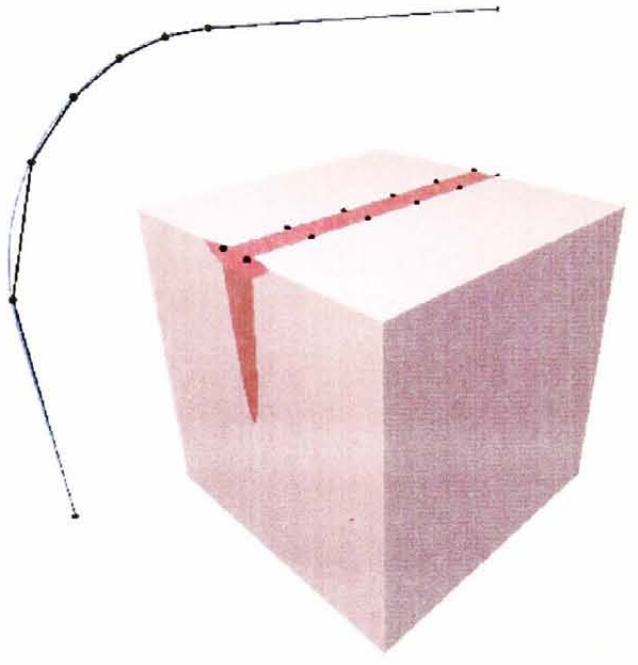


Figure 4.13b: Modified CL path

4.4.2 The Shoe Last

We first calculate the algorithmic error and then the two geometric machining errors for the machining of the cliffs on the shoe last. Equation (4.1) requires the cliff-part

angle in order to calculate a bound for the algorithmic error, from a visual inspection of the shoe it is clear that all the cliff-part angles are greater than 90° and hence:

$$e_{ca} \leq \sqrt{2}\, e_a. \tag{4.13}$$

Setting the minimum radius of curvature of the part equal to 16mm, as suggested in sub-section 3.1.3, and the cutter radius equal to 5mm, equations (3.9) and (4.13) give:

$$e_{ca} \leq 0.0465\text{mm}.$$

The geometric machining error of the transition movement is given by (4.7):

$$e_{cm1} = 0.9263\text{mm}.$$

This gives the total cliff error for the transition movement as:

$$e_{c1total} = 0.9728\text{mm}.$$

The geometric machining error due to local movements is given by (4.11):

$$e_{cm2} = 0.0251\text{mm}.$$

This gives the total cliff error due to local movements as:

$$e_{c2total} = 0.0716\text{mm}.$$

These results indicate that although local movements do not lead to excessive gouging, the transition movement does. Hence we calculate a suitable CL point spacing for the transition movement to ensure that the cliff is machined to within tolerance.

Let $e'_{cm1} = 0.0535\text{mm}$ since the sum of e_{ca} and e'_{cm1} must be less than our tolerance, 0.1mm. Then relation (4.8) gives:

$$c_{hz} = 0.0535\text{mm},$$

which guarantees $e'_{cm1} < 0.0535$mm. Hence for each CL path we must insert a new outer CL point just beyond the frontier and a local CL point with horizontal distance from the cliff satisfying:

$$r_c - c_{hz} = 4.9465 < d_{hz} < 4.9671 = r_c - e_{IOM2}$$

4.5 Discussion

In this chapter we have solved the cliff problem by extending the error analysis contained in Chapter 3 so that regionally dependent CL point separations can be calculated. Currently the process can only be applied to parts that contain cliffs on the boundary. To extend the process further and include parts containing internal cliffs, an algorithm is required to identify such features so that appropriate CL points can be inserted. In particular this extension would allow the machining of pocket regions.

We are now equipped to generate toleranced cutter paths for tangent continuous parts with boundary cliffs and so in Chapter 5 we focus on improving the performance on the algorithm, i.e. reducing the generation period.

Chapter 5

Optimisation of IOM for Ordered Data

In chapters 3 and 4 we developed the IOM, an algorithm that generates cutter paths directly from points. It was demonstrated that the resulting quality of machining matched that of a commercially available CAD/CAM package. However, as highlighted in section 3.5, our implementation of the IOM is relatively slow. This is because the implementation checks every grid point for every surface point and no account is taken of the coherence of the data sets. We use the term coherence to refer to the geometric "locality" of points, this is similar to the usage of the term spatial coherence in computer graphics [Foley and Van Dam 1983]. Hence the aim of this chapter is to modify the IOM such that superfluous checks are eliminated. We do this by using data arranged in sections and omitting from the checking procedure all sections that cannot contribute to a given iteration of the IOM.

Generally both the surface data and the pre-offset cutter paths, referred to as grid data, will be ordered sets. However this is not always the case and hence in section 5.1 we formulate the dual process to the IOM which gives us the choice of exploiting either the coherence of the surface data or the coherence of the grid data.

In section 5.2 we describe and subsequently analyse the two modifications optimised for data ordered into nests of sections. The first modification is suitable for all sectional data whereas the second requires that the points are monotonically ordered along the sections. In section 5.3 we do the same for the two methods optimised for nests of parallel planar sections. The chapter closes in section 5.4 with a discussion of the results obtained from each of the modifications.

5.1 The Dual Process of the IOM

We now describe the dual process which allows us to offset, exploiting the coherence of the grid data. The core process of the original IOM as described in sub-section 3.1.1 can be summed up as follows; recall that the CC-region is the region formed by projecting the tool along the tool axis onto the surface:

> Given a grid point, its final offset position is determined by all the surface points located in its CC-region and so we offset with respect to these points.

Figure 5.1 shows the CC-region of a given grid point and all the surface points that lie in it. Hence instead of checking every surface point for each CC-region, the process is made more efficient by exploiting the coherence of the surface data.

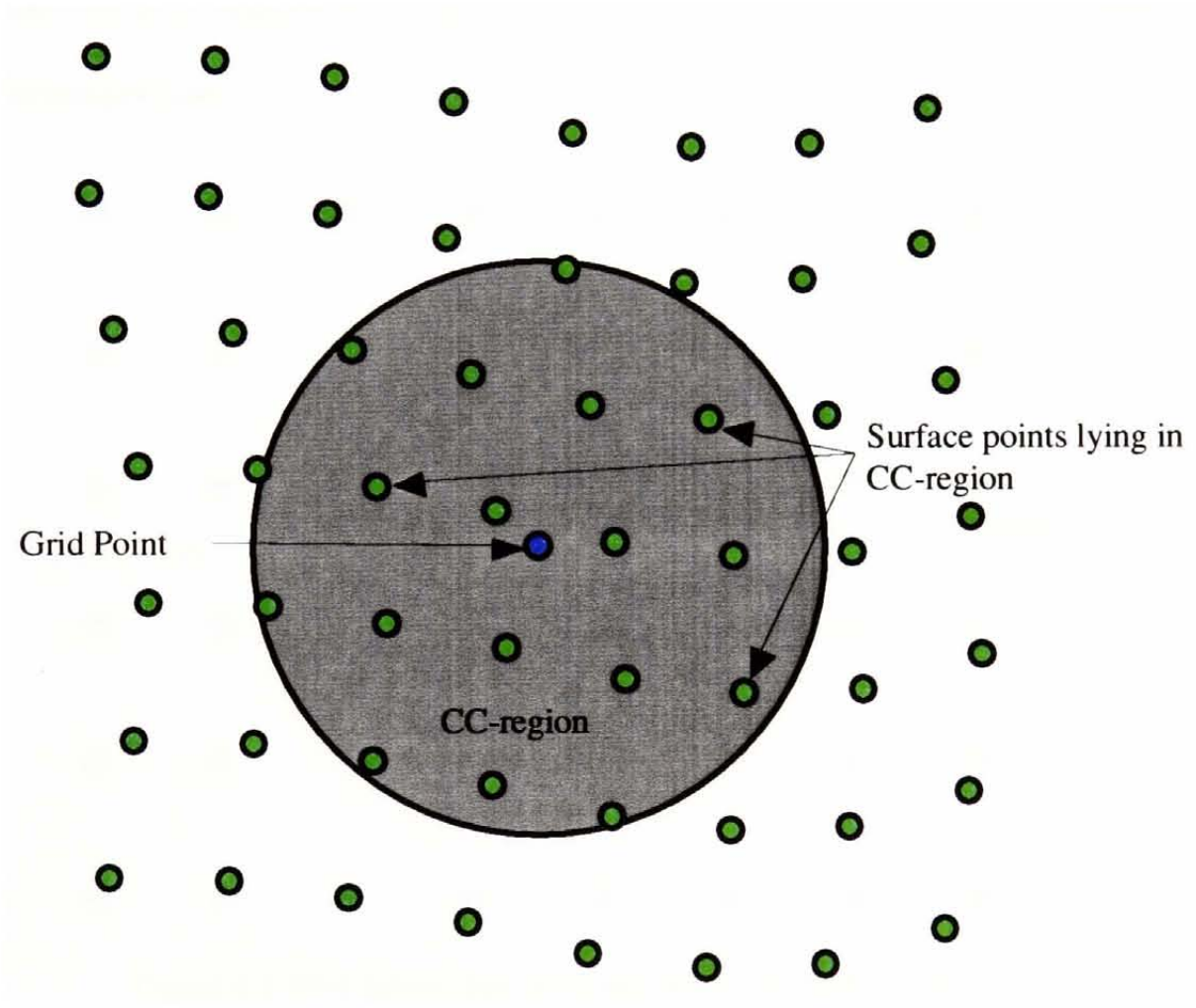


Figure 5.1: Surface points lying in the CC-region of a grid point

However we can achieve the same result as that of the IOM by using the dual process. The dual process effectively checks each relevant grid point for every surface point as opposed to the original IOM that checks each relevant surface point for every grid point. The core process of the dual process can be described as:

> Given a surface point, offset with respect to that point all grid points contained in its active zone.

The active zone of a surface point is analogous to the CC-region of a grid point. It is the region which contains all the grid points acted upon by a given surface point and hence consists of all the grid points not greater than the radius of the cutter away from the projected surface point onto the grid data plane. Figure 5.2 shows the grid points that lie in the active zone of the displayed surface point. Note that though the grid points are shown in a regular rectangular arrangement they are not constrained to lie in such an arrangement.

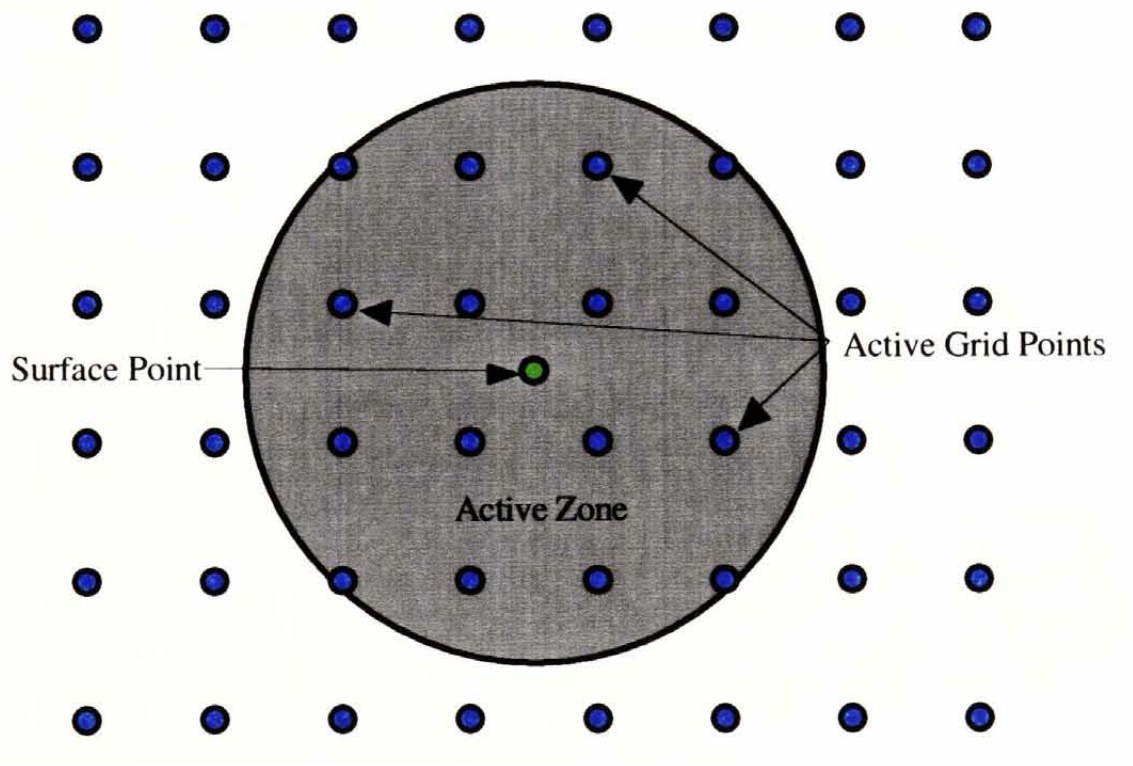


Figure 5.2: Grid points that lie in active zone of surface point

Hence by using the dual process we can exploit the coherence of the grid data to simplify the process instead. To highlight the difference between the two processes Figure 5.3a and Figure 5.3b show the result of one iteration of each algorithm

respectively. In Figure 5.3a, representing one iteration of the original IOM, the red grid line intersects with the blue inverted tools at the final height of the grid point. Note that only a few of the inverted tools are partially displayed for clarity.

In Figure 5.3b, representing one iteration of the dual process, all the grid points that are affected by a single surface point have been offset.

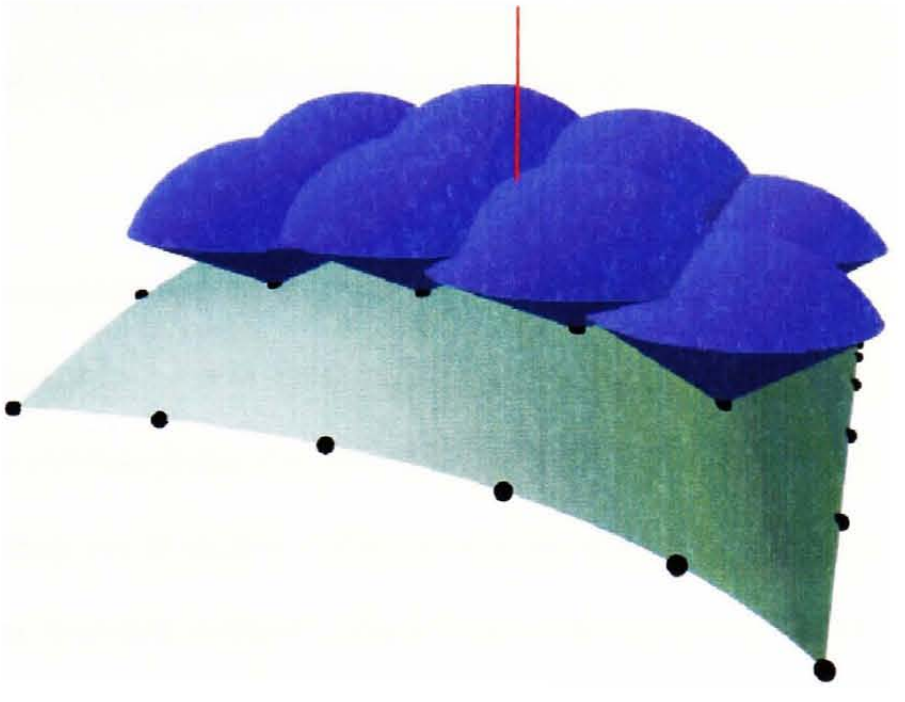


Figure 5.3a: One iteration of the original IOM process

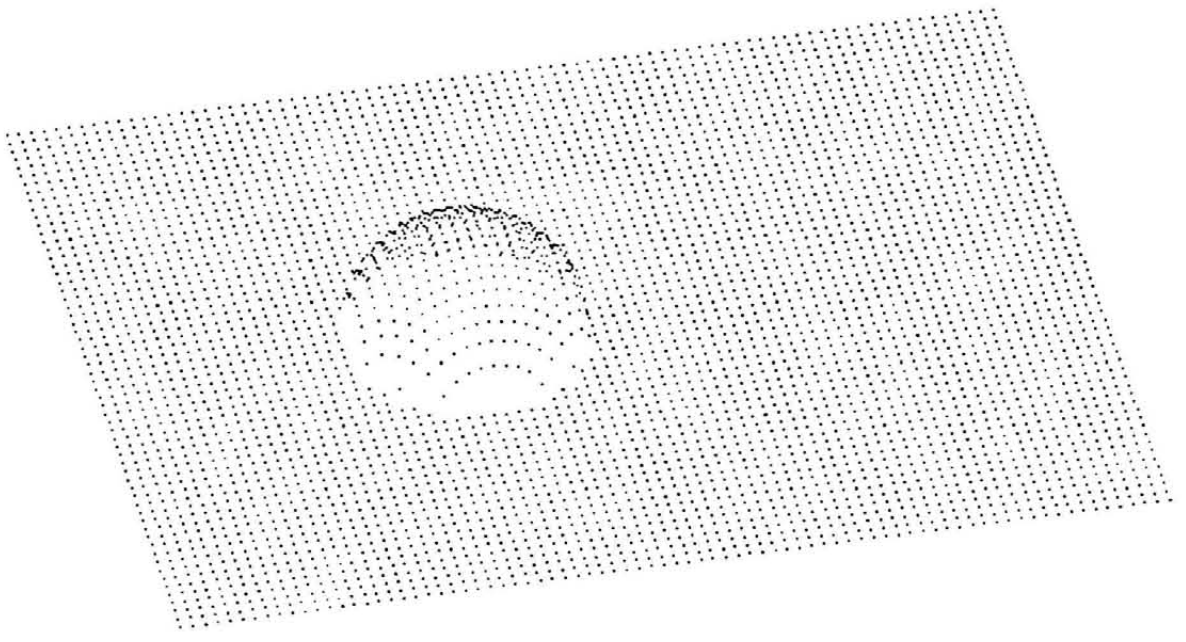


Figure 5.3b: One iteration of the dual process

Hence if either of the data sets is arranged into sections we may choose the process which will prove most beneficial.

5.2 Non-planar Section-Based IOM (S-IOM)

We now modify the IOM to exploit the features of a data set which is arranged in nests of sections. The modification is described for the dual process, i.e. for grid data arranged in nests of sections, but the modification for the original IOM, i.e. for surface data arranged in nests of sections, is easily derived. The reason for doing so is that it is anticipated that the grid data will have a greater degree of coherence than the surface data because the grid data represents the path a cutting tool will eventually follow.

We define an active point as a grid point that lies in the active zone of a given surface point; we define an active column as a grid column that contains an active grid point. To remove some columns from the checking process we require that a given surface point's corresponding set of active columns are all adjacent. This means that once we have established an inactive column either side of the active zone we can eliminate all further columns from consideration. This is not an unreasonable constraint as the points characterise curves and we are effectively requiring that these curves do not cross. Figure 5.4 shows a set of curves where the active columns are not all adjacent, remember that the grid columns do not necessarily have to be linear. The circles represent grid points that lie on active columns and crosses represent points that lie on inactive columns.

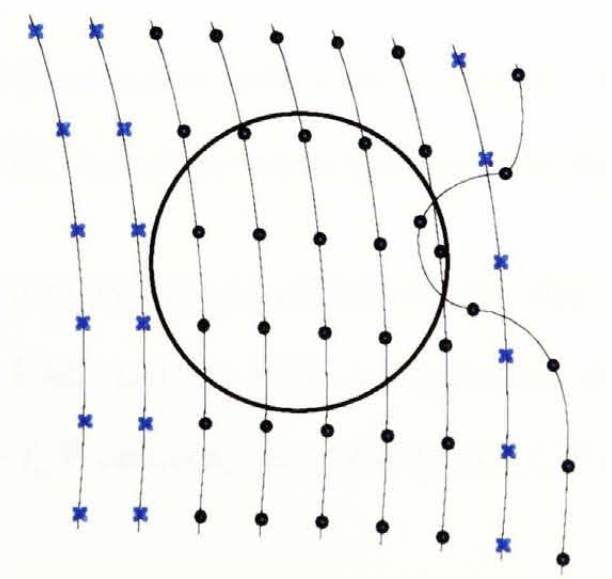


Figure 5.4: Sections that cross each other and fail to meet criterion

Since the curves are defined discretely a less pathological scenario may have the same result. Figure 5.5 shows such an example where the active zone fits between the points on one column and includes a point from the next. For clarity only a few of the curves are shown and the active columns are in bold. It can be seen from the figure that the points need to be spaced relatively far apart along the curve in relation to the spacing of the curves in order to fail this criterion. Generally the user will generate the grid data to ensure the criterion is satisfied but if this not practicable then the basic IOM as described in section 3.2 must be used.

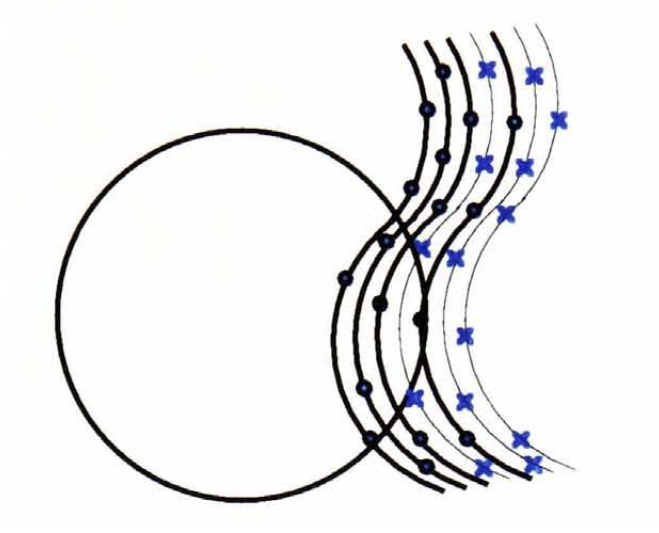


Figure 5.5: Sections that do not cross each other but still fail criterion

5.2.1 S-IOM Description

Phase 1: Initial Case

To start we must find an active column for the first surface point, i.e. find a column from the grid data that enters the active zone. This requires that we check each grid point on each column until an active grid point is found. This column is labeled as the first active column, FAC, and each active grid point in this column is offset.

The range of active columns is then established with the last active column being labeled LAC. Using the FAC and the LAC we calculate the middle active column, MAC; let the FAC be the $f_{ac}{}^{\text{th}}$ column, the LAC be the $l_{ac}{}^{\text{th}}$ column and the MAC be the $m_{ac}{}^{\text{th}}$ column using:

$$m_{ac} = \left[\tfrac{1}{2}\left(f_{ac} + l_{ac}\right)\right] \tag{5.1}$$

where the square brackets denote the integer part. Note that if there are no active columns for the first surface point we move onto the next surface point and repeat Phase 1 until an active column is found. Figure 5.6 shows the points that are checked in Phase 1 of the S-IOM.

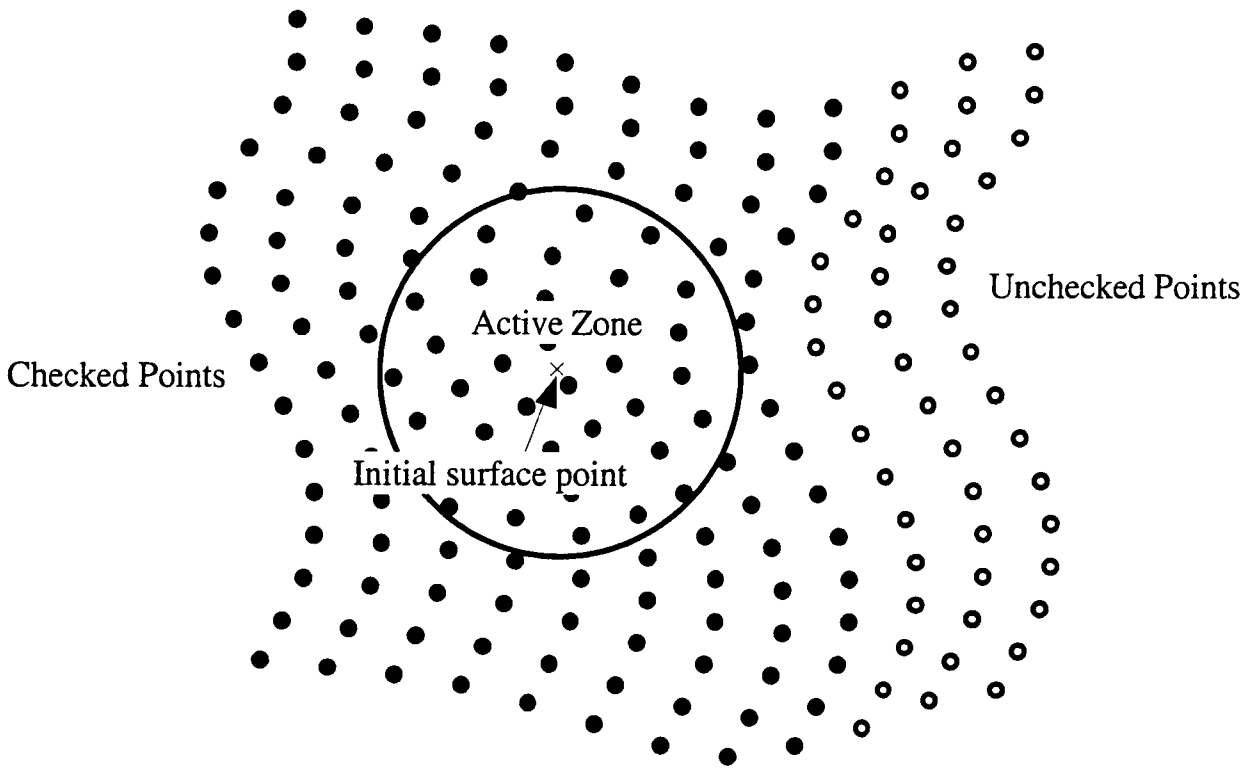


Figure 5.6: Points that are checked in Phase 1 of the S-IOM

Phase 2: Mainstream Case

We move onto the next surface point and by starting at the MAC establish the range of active columns labeling the new FAC and LAC as appropriate. The MAC is updated by substituting the new values for the FAC and the LAC into (5.1).

If the original MAC is not an active column then it is likely that an active column is not far away. Hence we search outwards checking each side of the MAC in turn until a new active column is found. This is labeled as the new MAC and we continue. If the next active column is not close this indicates a lack of coherence in the surface points, such an event will only delay the process and not cause the algorithm to fail.

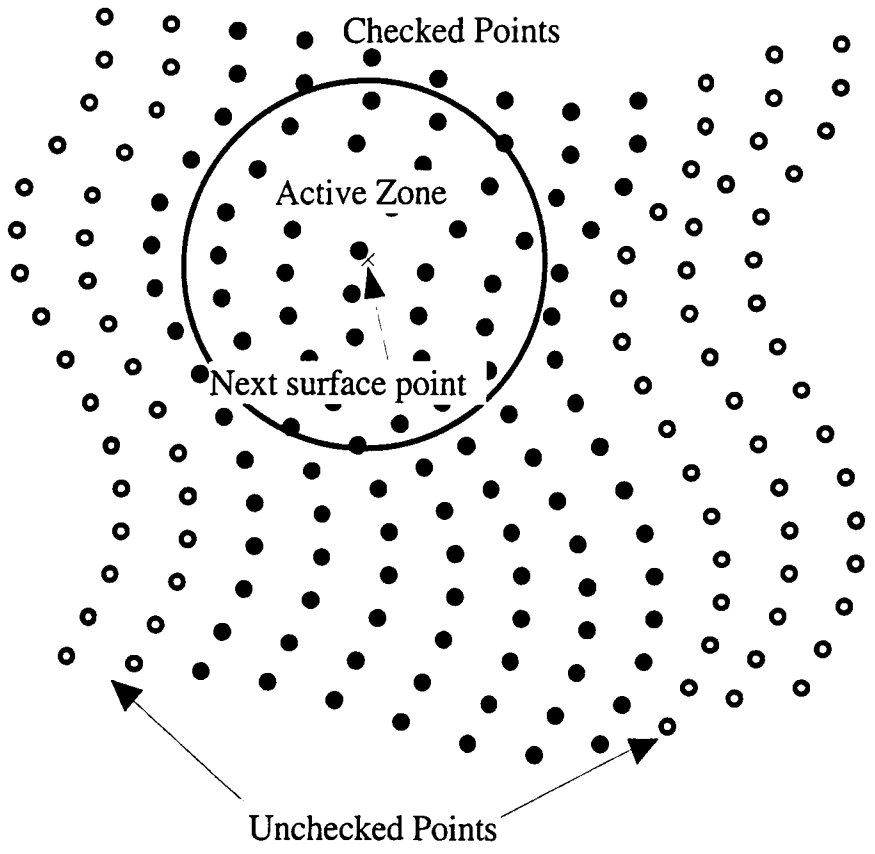


Figure 5.7: Points that are checked in Phase 2 of the S-IOM

5.2.2 Theoretical Improvement Achieved by the S-IOM

Using the S-IOM as opposed to the IOM reduces the number of performed operations. We now calculate the number of operations performed by the S-IOM and the IOM respectively in order to quantify the reduction in computation. There are two fundamental operations performed by the algorithms; a check to see if a given grid point is active and the inverse offset function. The same operation is used for both and hence a single instance of either is called a check. The relationships derived for each are applied to the example of the shoe last and the reduction in checks achieved by the S-IOM is given as a percentage of the number of checks performed by the IOM. The actual generation period of the S-IOM is also recorded.

The basic IOM checks every grid point for every surface point. Hence if there are p surface sections each containing an average of q entries and the offset grid has m columns and n rows, see Figure 5.8 for an example of these parameters on an example, then the number of checks made by the IOM is:

$$C_{IOM} = pqmn \tag{5.2}$$

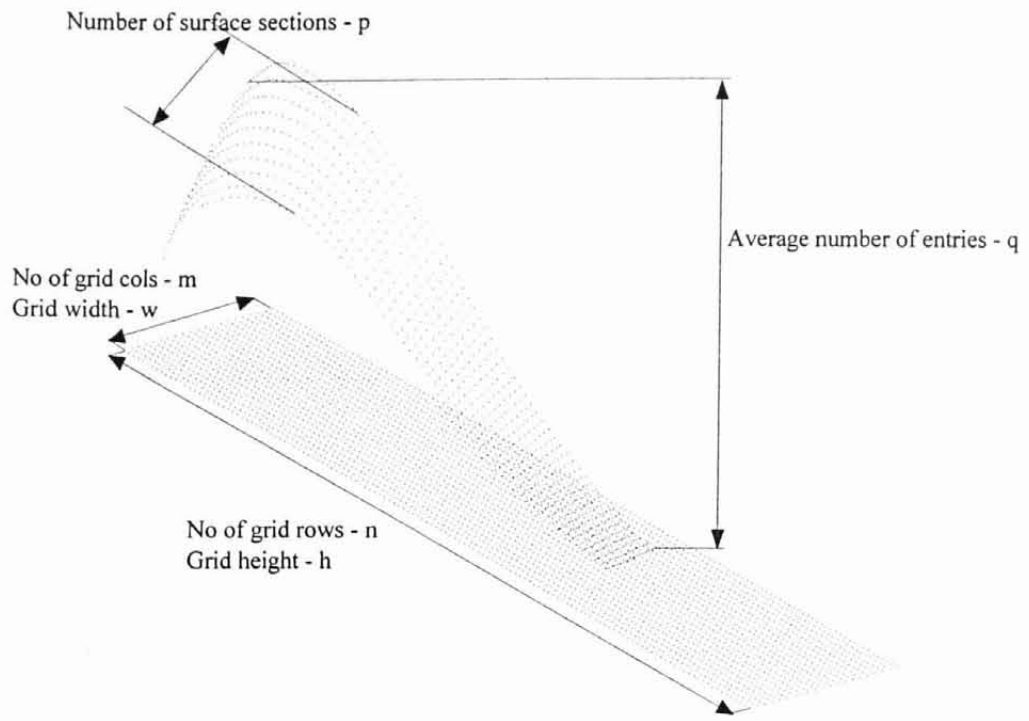


Figure 5.8: Parameters used to calculate theortical improvemtnt achieved by the S-IOM

To calculate the number of checks made by the S-IOM we simplify the problem. It is assumed the surface data and grid data occupy a rectangular region of space, of dimensions hxw, when viewed along the tool-axis. We also assume that the points in each data set are evenly dispersed. To facilitate the calculation of the number of checks, the points are divided into two sets. Figure 5.9 shows the two sets of points, Set A and Set B, which represent the grid points that are checked for any given surface point in Phase 2. Note that since only a rough approximation is required we ignore the additional points checked in Phase 1.

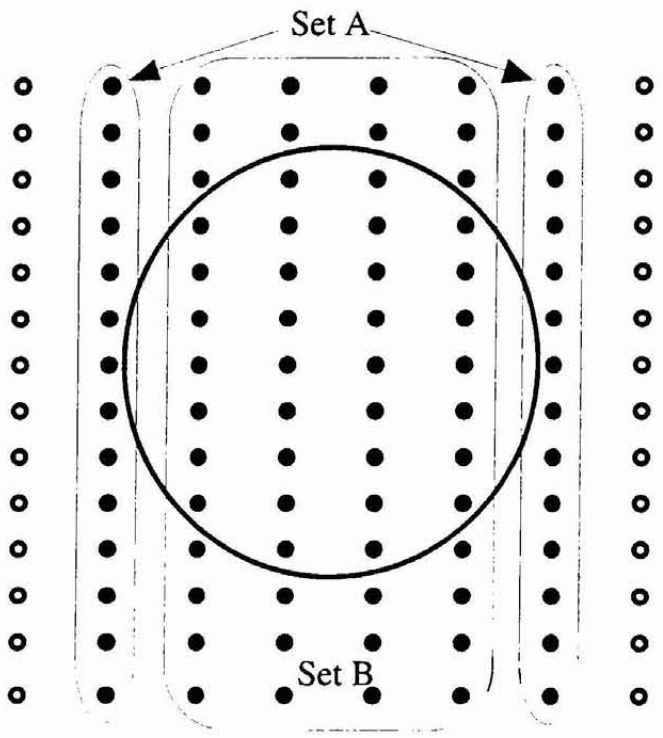


Figure 5.9: The sets of points checked for a given surface point

Set A consists of all the grid points that lie in the inactive sections either side of the active zone. Set A contains:

$2n$ grid points.

Set B is a box of points with height = h and width = $2r_c$. Hence Set B contains:

$$n\left[2r_c\frac{m}{w}\right] \text{ grid points,}$$

These points are checked for each surface point which gives:

$$pq\left(2n+n\left[2r_c\frac{m}{w}\right]\right) \text{ checks.}$$

However there is still a group of checks that may be made which we have not yet considered. Suppose a given surface point contains no grid points in its active zone, for such a surface point the S-IOM checks every grid point. This is equivalent to the surface point not lying in the CC-region of any grid point, such an instance may occur if we wish to machine only part of a given surface. Let α equal the ratio:

$$\alpha = \frac{\text{number of surface points not contained in any CC - region}}{\text{number of surface points}} \tag{5.3}$$

Then the total number of checks made by the S-IOM is:

$$C_{S-IOM} = pq\left(2n+n\left[2r_c\frac{m}{w}\right]\right)+\alpha pqmn \tag{5.4}$$

Hence if both data sets are arranged in nests of sections the choice of whether to employ the dual process or not can be decided by which results in a smaller α. We now apply the above calculations to the example of the shoe last. In doing so we note that the surface data is entirely contained within the grid data and hence choose to employ the dual process which results in $\alpha = 0$.

The shoe is contained within a rectangular region measuring 247mm x 151mm and defined by 337 sections each containing 211 entries. The grid data used contains 248 columns and 152 rows. These values are consistent with the error analysis performed

in Chapter 3 to achieve a 0.1mm tolerance and are shown in table 5.1 for ease of reference.

Variable name	Letter	Value
Grid width (in mm)	w	247
Grid height (in mm)	h	151
Number of surface sections	p	337
Average number of entries	q	211
Number of grid columns	m	248
Number of grid rows	n	152

Table 5.1: Variables used from shoe example

Substituting these values into equation (5.4):

$C_{S\text{-}IOM}$= 1.30 x10^8 checks (to 3 s.f.)

The number of checks performed by the IOM is given by equation (5.2):

C_{IOM}= 2.68 x10^9 checks (to 3 s.f.)

Hence the S-IOM performs only 4.8% of the checks performed by the IOM. The actual generation period of the S-IOM is given by the average of three runs as were all the methods:

35.5 seconds.

It is expected that the generation period of the modifications will be proportional to the number of checks made with some nominal period to account for the administrative operations performed by the computer, e.g. writing the output files. Hence once we have another result we can start predicting the generation period of the remaining algorithms.

5.2.3 S-IOM with Intra-Sectional Coherence (ES-IOM)

We now give the details for an enhancement to the S-IOM which exploits the intra-sectional coherence of the data. To do so it is required that once a section has left the active zone it cannot re-enter it. Hence two criteria are given which are sufficient to ensure that this requirement is met. Note that if we are considering surface sections we assume they have been projected along the tool-axis vector onto a plane perpendicular to the tool axis.

Criterion 1

The first criterion is that each column is strictly monotonic in some direction. This ensures the points are monotonically ordered along the column and that the column does not turn back on itself and re-enter the active zone.

Criterion 2

The second criterion is that any three consecutive points on a column lie on a circle with radius greater than that of the cutter. This criterion ensures the column does not leave the active zone and then turn back sharply to re-enter it as shown in Figure 5.10a. Note that if the columns fail to meet either of these criteria then the previous algorithm, the S-IOM, can be used. Figure 5.10a shows an example of a column that satisfies Criterion 1 but not Criterion 2 and Figure 5.10b shows a column that satisfies Criterion 2 but not Criterion 1.

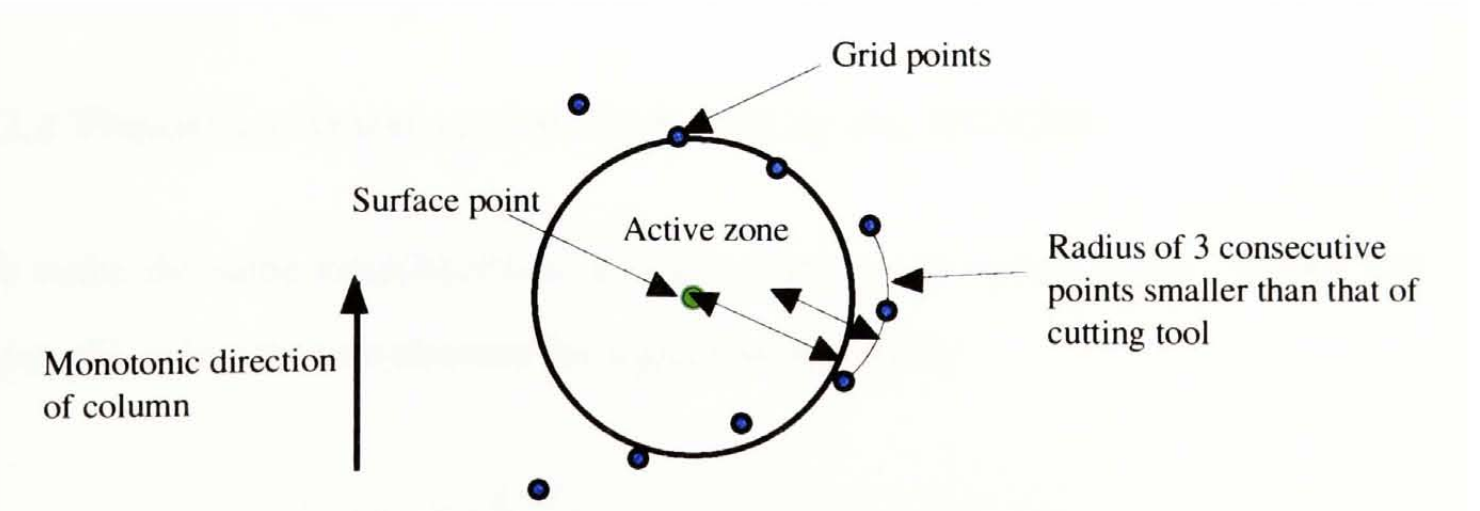


Figure 5.10a: Column satisfying Criterion 1 but not 2

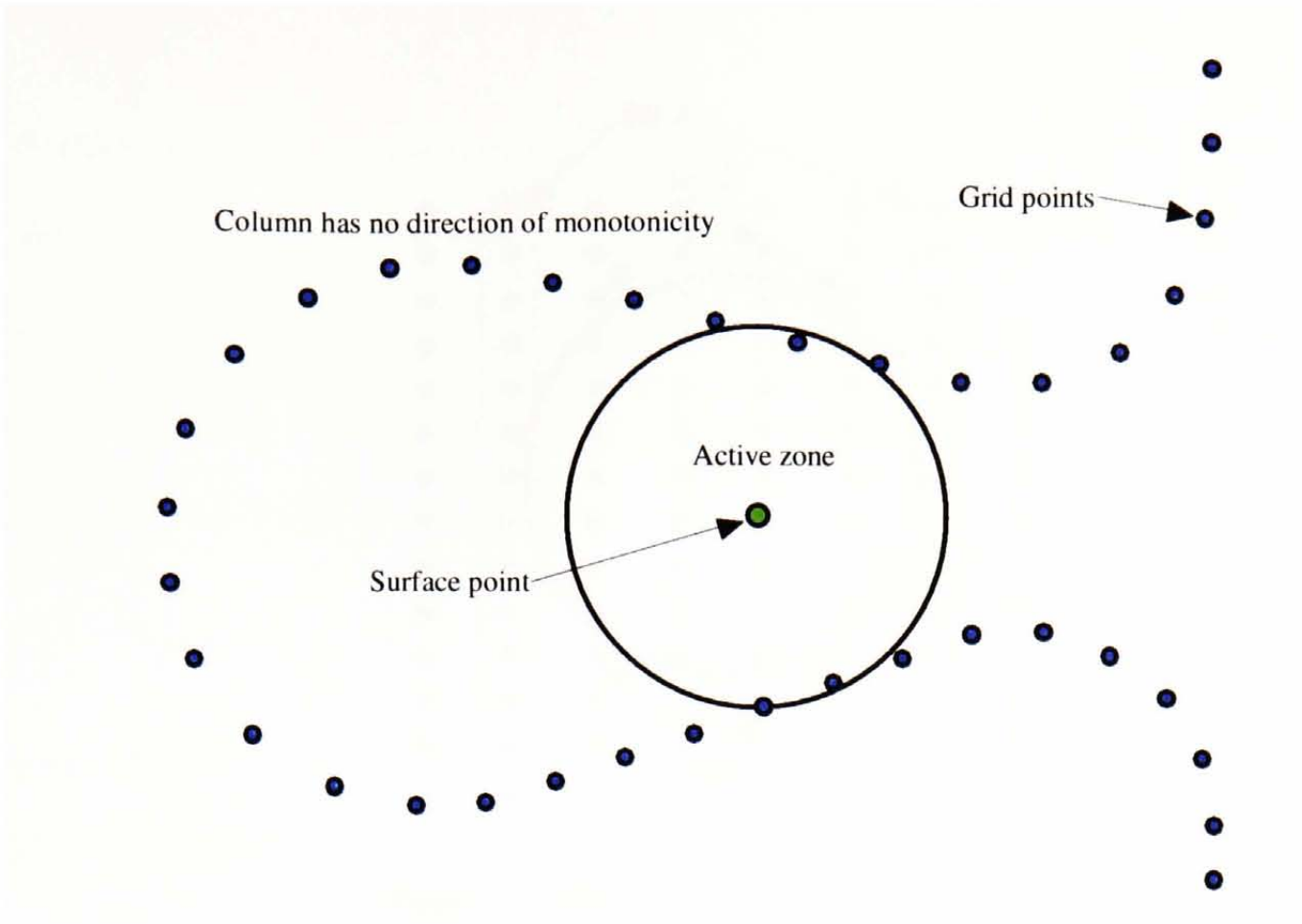


Figure 5.10b: Column satisfying Criterion 2 but not 1

Hence once a column has exited the active zone we can eliminate the rest of the grid points on that column from consideration.

The method is identical to the S-IOM except for the subroutine(offset contour). As each consecutive column is offset, the grid point that is initially checked is chosen such that the intra-sectional coherence is exploited.

Suppose that the $f_{agp}{}^{\text{th}}$ and $l_{agp}{}^{\text{th}}$ grid points were the first and last active grid points. Then the first grid point we check on the next column is the $m_{agp}{}^{\text{th}}$ point where:

$$m_{agp} = \left[\frac{f_{agp} + l_{agp}}{2} \right] \tag{5.5}$$

5.2.4 Theoretical Improvement Achieved by the ES-IOM

We make the same simplifications as were made in sub-section 5.2.2. Figure 5.11 shows the points that are checked for a given surface point.

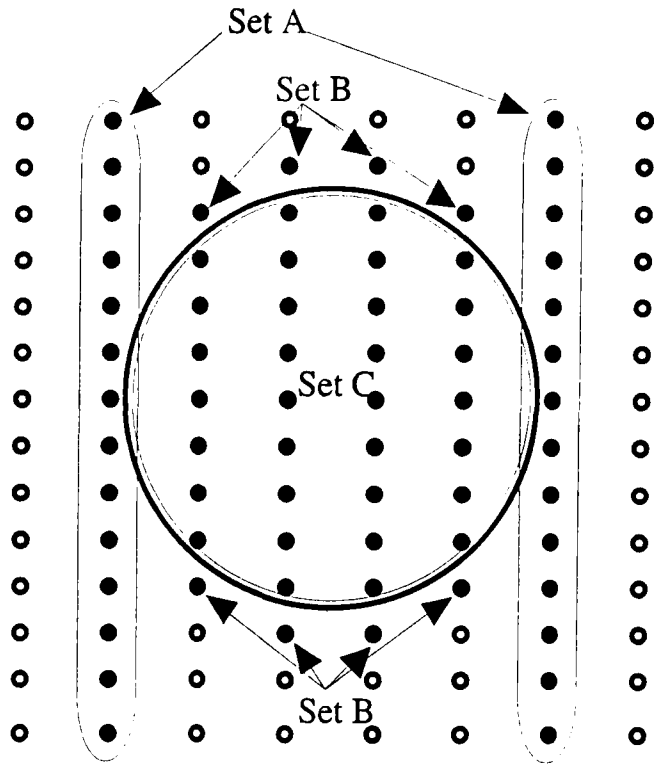


Figure 5.11: The points checked by the ES-IOM

Set A is as calculated in sub-section 5.2.2:

$2n$ grid points.

Set B consists of all the points checked as the algorithm exits the active zone. This is effectively a box containing:

$$2\left[2r_c\frac{m}{w}\right] \text{ grid points.}$$

Set C is the set of all active grid points, we approximate the density of points in this region by:

$$\sqrt{\frac{mn}{wh}}$$

Which gives an approximate size of Set C as:

$$\frac{\pi r_c^2 mn}{wh} \text{ grid points.}$$

Each of the above sets is checked for every surface point giving:

$$pq\left(2n+2\left[2r_c\frac{m}{w}\right]+\frac{\pi r_c^2 mn}{wh}\right) \text{ checks.}$$

Finally we take into account the ratio of surface points which contain no grid points in their active zone as calculated in sub-section 5.2.2 giving the total:

$$C_{ES-IOM} = pq\left(2n + 2\left[2r_c\frac{m}{w}\right] + \frac{\pi r_c^2 mn}{wh}\right) + \alpha pqmn \text{ checks.} \qquad (5.6)$$

Applying this formula to the shoe last:

$C_{ES\text{-}IOM}$= 2.42 x10^7 checks (to 3 s.f.)

This means that the ES-IOM performs 0.9% of the checks performed by the IOM. The actual generation period of the ES-IOM is:

Actual time = 13.5 seconds

The above two timings enable us to fit a linear model and hence to predict generation periods for the remaining two modifications.

5.3 Parallel Planar Section-Based IOM (PPS-IOM)

We now turn our attention to sections that lie in parallel planes, which are also referred to as contours. Note that grid data only qualifies as parallel planar sections if the finished cutter paths lie in parallel planes. Hence the grid data will either have linear rows or linear columns and the containing plane of the resulting cutter paths is described by a single row/column and the tool-axis. As in section 5.2 we describe the case for offsetting using the dual process to the IOM.

We define an active contour slightly differently to an active column to make the definitions consistent with the algorithm. A contour is active if the distance between the surface point and the plane containing the contour is less than the radius of the cutter. The main difference between an active column and an active contour is that an active contour does not necessarily contain an active grid point. Figure 5.12 shows an example of a section that would qualify as an active contour but not as an active column.

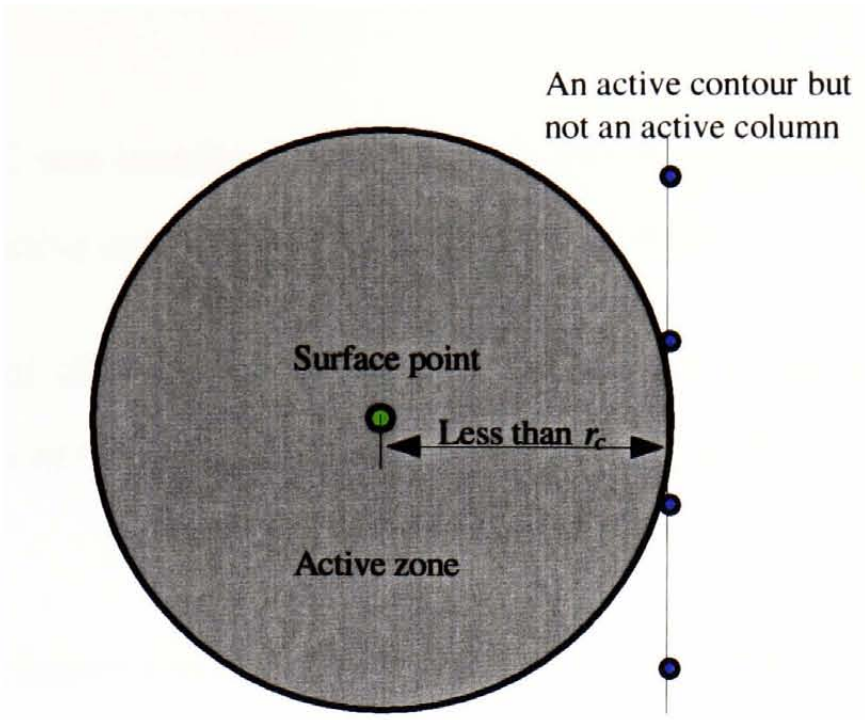


Figure 5.12: A section that enters the active zone but does not contain an active grid point

5.3.1 PPS-IOM description

Phase1: Initial Case

As in sub-section 5.2.1 we must find an active contour from the grid data. Given the normal to the plane containing the contour we can do this by checking only a single point from each contour. Once we have found an active contour we mark it as the FAC and then check along the contour offsetting any active grid points on it. If no active grid points are found we proceed to the next contour.

We then establish the range of active contours labeling the last as the LAC and offset each contour in this range.

Phase 2: Mainstream Case

Using equation (5.1) we calculate the MAC and use it as a start contour for offsetting with respect to the next surface point. If the MAC is an active contour then we offset all neighbouring active contours, stopping in each direction when an inactive contour is found. Update the MAC using the new FAC and LAC and then restart Phase 2 for the next surface point.

If the original MAC was inactive then we search, alternating between both sides of the MAC, until an active contour is reached and then continue.

The only substantial difference between this algorithm and the S-IOM is in the subroutine(find next MAC), which actually finds the next active column/contour and not the next MAC.

5.3.2 Theoretical Improvement Achieved by the PPS-IOM

To calculate the number of checks made by the PPS-IOM we simplify the problem as in sub-section 5.2.2. The points checked in a single iteration are divided into two sets as shown in Figure 5.13. Set A consists of the two points checked either side of the active zone and Set B consists of all the grid points in the active contours.

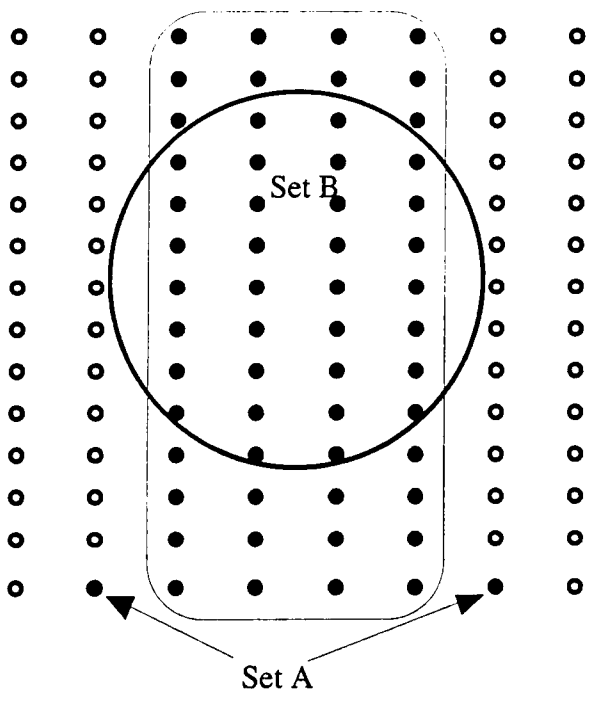


Figure 5.13: The sets of points checked by the PPS-IOM

Set A contains:

2 grid points.

Set B is a box of points containing:

$n\left[2r_c \frac{m}{w}\right]$ grid points.

These points are checked for each surface point giving:

$$C_{PPS-IOM} = pq\left(2 + n\left[2r_c \frac{m}{w}\right]\right) \text{ checks.} \qquad (5.7)$$

Note that we do not need to take into account the checks performed for surface points that do not lie in any CC-region as in sub-sections 5.2.2 and 5.2.4. This is because only a single check needs to be made per contour. We now apply equation (5.7) to the shoe last example.

$C_{PPS\text{-}IOM} = 1.08 \times 10^8$ checks. (to 3 s.f.)

Hence the PPS-IOM performs 4.0% of the checks performed by the IOM. Now using the previous two results we predict that the generation period of the PPS-IOM is roughly:

31.0 seconds.

The actual generation period of the PPS-IOM is:

30.8 seconds.

As the actual result shows, the linear fit provides a good model of the time complexity of the modifications.

5.3.3 PPS-IOM with Intra-Sectional Coherence (EPPS-IOM)

Following the lead of section 5.2 we enhance the PPS-IOM by exploiting the intra-sectional coherence of the data. This results in the elimination of nearly all the superfluous checks made along any of the contours. Since the contours lie in parallel planes it is sufficient that the points are monotonically ordered along the section to ensure the following enhancement is applicable.

5.3.4 Theoretical Improvement Achieved by the EPPS-IOM

Making the same simplifications as before, Figure 5.14 shows the points that are checked for a given surface point.

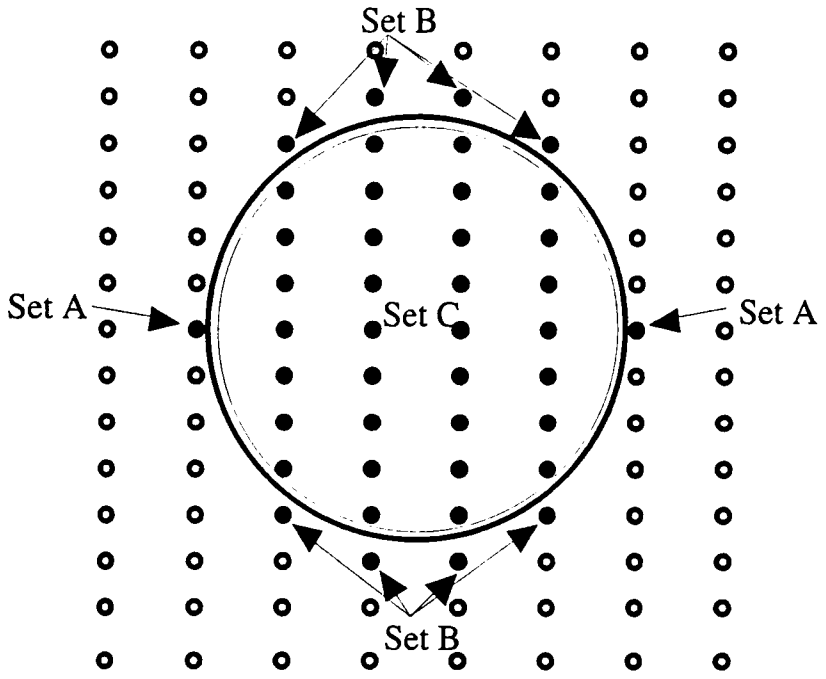


Figure 5.14: Points checked by the EPPS-IOM

The points checked in set A will generally be nearly in line with the centre of the active zone as it is in this region where each column is first checked and contains:

2 grid points.

Set B consists of all the points that are checked as the algorithm exits the active zone. This will generally be a box containing:

$$2\left[2r_c\frac{m}{w}\right] \text{ grid points.}$$

Set C is the set of all active grid points and we approximate the size of this set with:

$$\frac{\pi r_c^2 mn}{wh} \text{ grid points.}$$

Each set of points is checked for each surface point giving:

$$C_{EPPS-IOM} = pq\left(2 + 2\left[2r_c\frac{m}{w}\right] + \frac{\pi r_c^2 mn}{wh}\right) \text{ checks.} \tag{5.8}$$

This formula suggests that if there is a choice as to which orientation we put the grid rows and columns it would be beneficial to choose the direction of the grid columns such that they have least density. However in practice the machining strategy will take precedence, e.g. if we are machining a long thin shape the cutter paths will lie along the longest axis. Applying this equation to the example of the shoe last:

$C_{EPPS\text{-}IOM}$= 2.69 x10^6 checks. (to 3 s.f.)

Hence the EPPS-IOM performs 0.1% of the checks performed by the IOM. Using the same linear fit as in sub-section 5.3.2 we predict the generation period of the EPPS-IOM to be:

9.0 seconds.

The actual generation period of the EPPS-IOM is:

9.7 seconds.

Again the linear model provides a good prediction. In the next section we discuss the timing results of all the previous sections.

5.4 Discussion of Results

Table 5.2 shows the number of checks performed by each of the modifications as a percentage of those made by the IOM and their actual generation periods. It is worth noting that PowerMILL took 11 seconds to generate cutter paths of a similar quality.

Algorithm	Percentage checks	Actual result
S-IOM	4.8%	35.5 secs
PPS-IOM	4.0%	30.8 secs
ES-IOM	0.9%	13.5 secs
EPPS-IOM	0.1%	9.7 secs

Table 5.2: Percentage of checks made by modifications and generation periods

These results are shown in a graph in Figure 5.15. The red crosses represent the four actual results; from the left they are the EPPS-IOM, the ES-IOM, the PPS-IOM and the S-IOM. The blue line shows the linear fit that was used to predict the generation periods of the PPS-IOM and the EPPS-IOM. The intersection of the linear model with the y-axis can be interpreted as the administration period of the algorithm. The slope of the graph effectively represents the rate at which the computer can perform the checks. A faster processor would result in the graph becoming less steep.

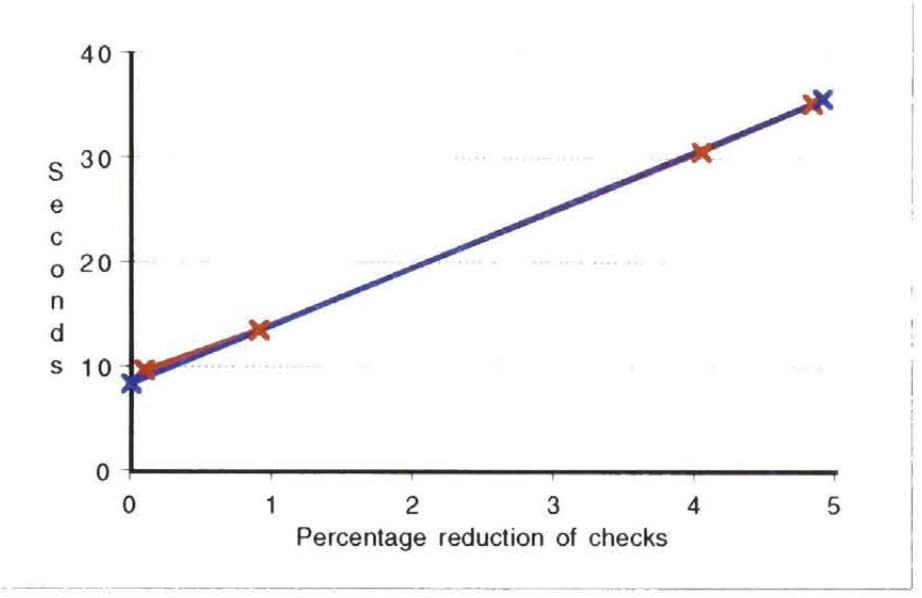


Figure 5.15: Actual results and linear model

Further tests were performed on data sets other than the shoe last. The data sets used were consistent with the simplifications made in sub-section 5.2.2 and of varying sizes. Firstly the administration period was found to be roughly proportional to the number of grid points; this was expected as the main function performed by the algorithm apart from offsetting is the writing of the output to a file.

Secondly, if we ignore the administration period and assume that the active zone contains only a small fraction of the grid points then the time complexity of the modifications are roughly given by:

$$\text{S-IOM} \propto pqmn/w \tag{5.9}$$

$$\text{PPS-IOM} \propto pqmn/w \tag{5.10}$$

$$\text{ES-IOM} \propto pqn \tag{5.11}$$

$$\text{EPPS-IOM} \propto pqmn/hw \tag{5.12}$$

These are derived from the formulae (5.4), (5.6), (5.7) and (5.8). From (5.11) we can infer that it is better to have more columns than rows in the offset grid when using the ES-IOM. Relation (5.12) indicates that if the density of the offset grid is maintained then the number of calculations performed by the EPPS-IOM will only increase proportionally to the number of extra surface points introduced. Also if we compare each of the above relations with equation (5.2) it is clear they all offer a much better time complexity than that of the basic IOM.

To conclude we note that we have generated cutter paths quickly and to within tolerance directly from point data. It remains to illustrate where our work is compatible with current machining practices. Hence in Chapter 6 we consider the process of generating cutter paths starting from a multi-patch IGES surface.

Chapter 6

Aspects of Operating on Multi-Surfaces

In the previous three chapters we investigated the IOM and developed a number of enhancements that allowed the efficient generation of cutter paths for a part whose shape is defined by a set of points. The basic IOM as described in Chapter 3 can operate on any set of points and generate paths to within tolerance assuming that the points are suitably dense and define a tangent continuous surface. A modification to deal with cliff-edges is described in Chapter 4. Finally in Chapter 5, a number of enhancements are presented that make the IOM more efficient by ordering the points in sections.

The work presented in these chapters has been undertaken with the tacit assumption that we are dealing with a straightforward point definition, that is a part whose underlying shape can be defined by a single valued surface with a rectangular topology. However not all part shapes are simple and defining a more complex shape can require a non-rectangular assembly of surfaces. We refer to such a definition as a multi-surface and in this chapter investigate the issues they raise but restrict ourselves to those that are generic to all CAM packages which are illustrated in Figure 6.1. The end nodes of the diagram are coloured to indicate the extent to which each has been addressed either directly or indirectly by this thesis so far. Items in green have been addressed but are revisited and described further; items in orange have been partially addressed; and items in red have not been considered at all.

The chapter is divided into six sections, the first five address the categories given in the primary nodes of the diagram, i.e. surface representation, parameterisation, trimmed surfaces, machining strategy and sampling interval. Note that for the rest of this chapter the IOM and its enhancements as presented in the previous chapters are simply referred to as the IOM.

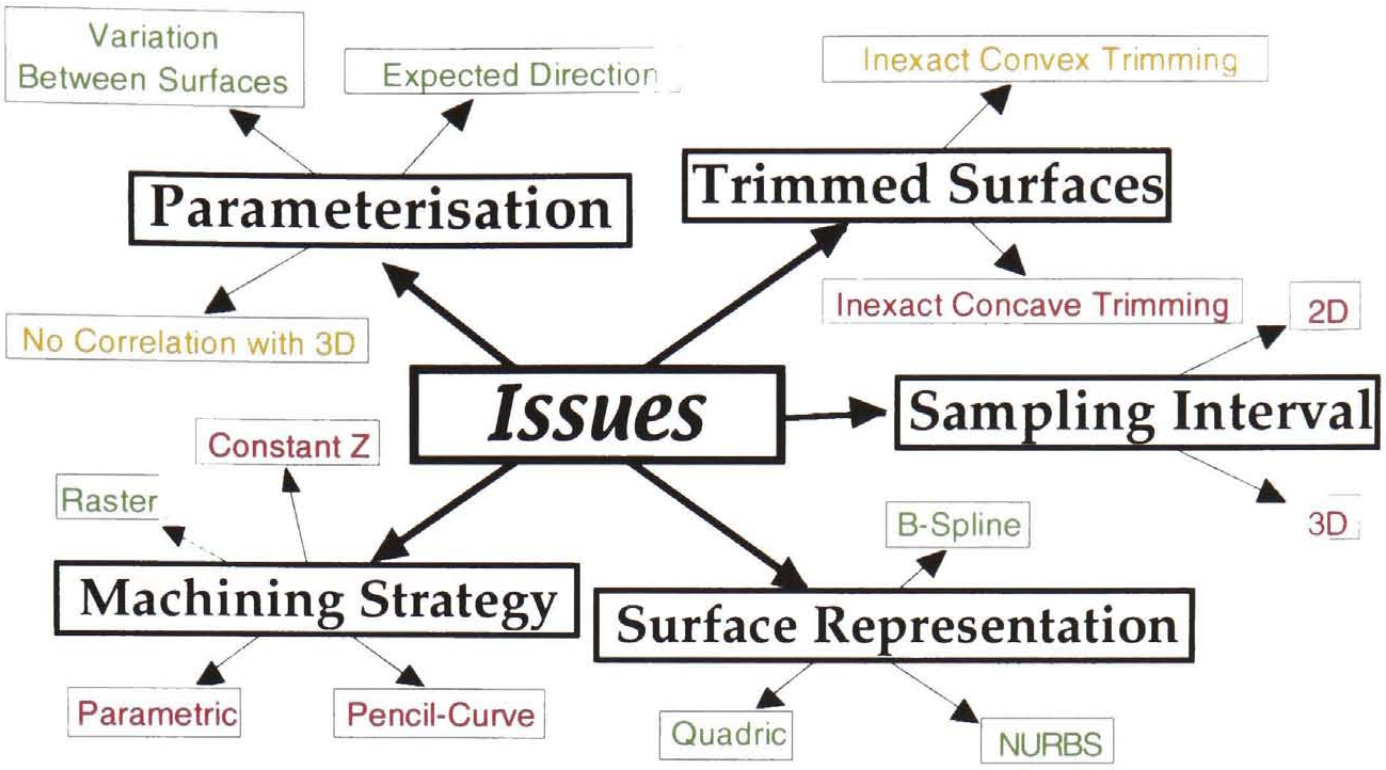


Figure 6.1: Problems tackled in this chapter

6.1 Surface Representation

The part that is to be machined may be defined originally in any one of a number of representations and it will be required that the CAM package perform a number of operations on the part. Suppose we have m representations and n operations, then there are nxm combinations to implement. However by converting the representation into a common one upon which all the CAM operations are performed, the number of combinations is reduced to $n+m$. We refer to the common representation as the intermediate representation. Figure 6.2a and Figure 6.2b illustrate the reduction of combinations that results from employing an intermediate representation.

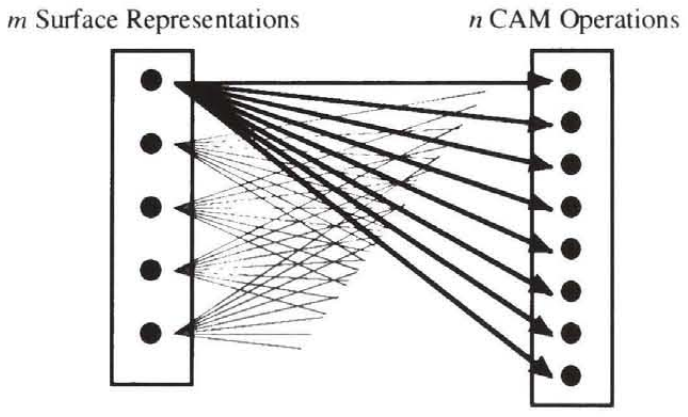


Figure 6.2a: mxn combinations

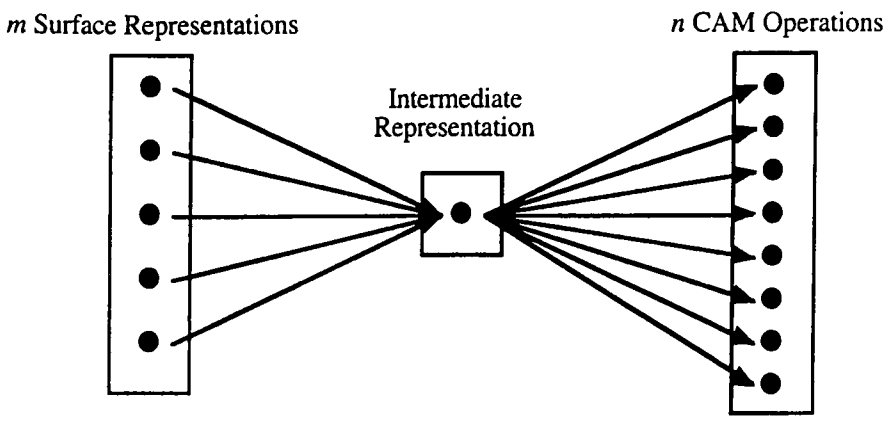


Figure 6.2b: *m+n* combinations

The intermediate representation chosen will generally reflect the strategy used to generate the cutter paths. This is reflected in Chapter 2 where the three main strategies described in the literature employed either:

- Polyhedral approximation of the part,
- Generation of the offset surface or an approximation it,
- Calculation of the surface points and normal data.

By definition the first strategy must employ a polyhedron as the intermediate representation. For the remaining two strategies, although they are not constrained to use any particular intermediate representation, it was found that the literature employing them typically used either a parametric or point representation.

The overall strategy as presented in Chapter 1 of this thesis is to move away from the various representations as the initial definition and simply use a characterising set of points to initially define the surface. However the characterising set of points can equally be regarded as the intermediate representation.

6.2 Surface Parameterisation

In Chapter 2 we mentioned a number of difficulties associated with generating parameter-based cutter paths, e.g. non-planar paths, self-intersection resulting in the

side-step problem. However these issues are made evident by the consideration of a straightforward surface definition. In this section we consider the issues that arise when considering the machining of a multi-surface and indicate the solutions provided by the IOM.

Underlying the natural way in which the IOM solves these issues is the fact that from the outset a geometrically motivated algorithm was sought. Hence as part of the concept of using a point-based strategy, the issues raised by parameterisation are inherently solved.

6.2.1 Correlation of Parametric Definition with 3D Shape

The first problem we address is that of correlation between the parametric definition of a part and its 3D shape. Although this topic is not strictly motivated by multi-surfaces, the increase in complexity of a part associated with the addition of more and more parametric surfaces leads to the parameterisation being less likely to reflect the 3D shape. This is in contrast to the parameterisation of a single patch which will generally, though not necessarily, correspond fairly closely to its 3D shape.

A lack of correlation between the parameterisation and the shape can create two distinct problems for a CAM package. Firstly, if the CAM package generates parameter based cutter paths, the part's surface finish may suffer from the cutting path pattern being unrelated to its shape. For example the isoparametric lines on a parametrically defined plane may deviate from the expected path as shown in Figure 6.3. The cutter paths generated by the IOM are inherited from an offset grid that is independent of the surface definition and hence the parameterisation cannot affect the cutter path pattern.

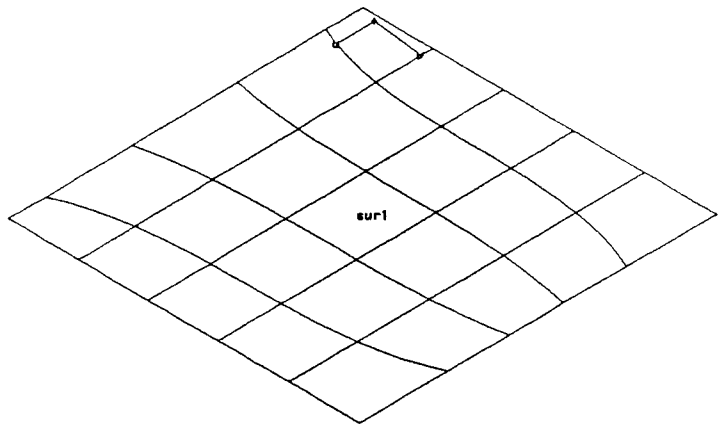


Figure 6.3: Isoparametric lines varying unexpectedly on a plane

Secondly, if the CAM package interrogates the parameterisation at regular sampling intervals with respect to 3D space, irregular definitions may lead to the sampling algorithm having difficulties. Similarly, the section-based modifications presented in Chapter 5 have a number of criteria to be satisfied by the point definition in order to employ the optimisations, however the IOM is not restricted to operating on data satisfying these criteria.

There is another issue which must also be considered, the spacing of the surface points has a direct impact on the tolerance of the algorithm and so if the parameterisation varies greatly over a surface, care must be taken when generating the point definition. In such cases a geometric method for sampling the points from the surface would be advantageous, e.g. with respect to arc-length. A study of geometric sampling is beyond the scope of this work but for further details see [Czerkawski 1996].

6.2.2 Variation of Parametric Definition between Surfaces

The constituent surfaces of a multi-surface will generally be required to meet with tangent continuity to within some tolerance. However this does not mean the parameterisations of neighbouring surfaces will meet up. Figure 6.4 shows two planar bi-cubic Bézier patches that meet with tangent continuity. The internal lines represent the isoparametric lines for u and v taking the values 0.2, 0.4, 0.6 and 0.8. As can be seen in the figure, the parameterisations of the two surfaces in the u-direction do not line up.

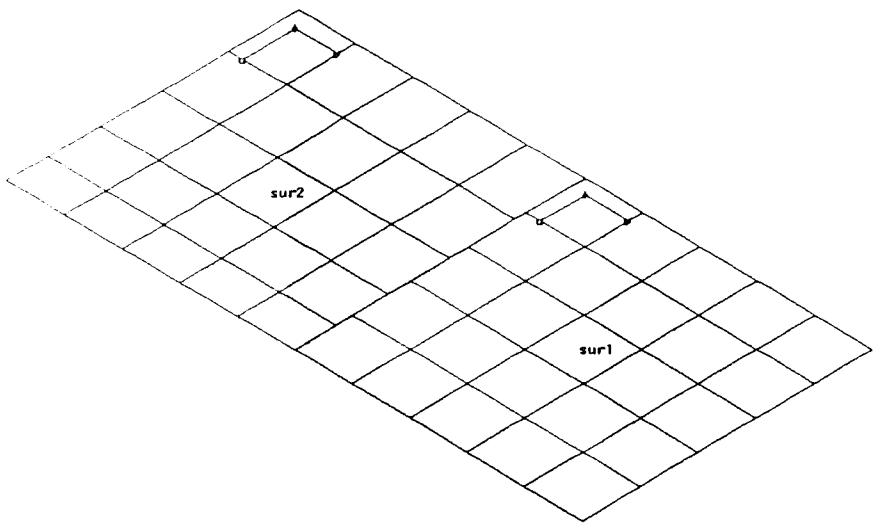


Figure 6.4: Tangent continuous surfaces with discontinuous parameter lines

The general situation will be more complicated with curved surfaces and a tolerance. Figure 6.5 shows a practical example taken from the definition of a car bonnet kindly supplied by Rover. The blue curves in the wire-frame pictures represent patch boundaries, the grey curves regular parametric intervals and the red curves highlight a parametric discontinuity across neighbouring patches. Note that unlike in Figure 6.4, the neighbouring surfaces in this example have different degrees but the net result is still the same – discontinuous parameter lines.

A CAM package that depends on the parameterisation to generate the cutter paths will face difficulties each time it exits a surface because to generate continuous paths the boundaries of each adjoining surface must be searched for a suitable parameter value.

To further complicate matters, once the parameter value giving positional continuity has been found, the parameter direction will not necessarily be continuous with that of the previous surface which may result in the undesirable feature of a sharp change of direction in the cutter path. However, the whole issue is solved by the IOM for the same reasons as described in sub-section 6.2.1.

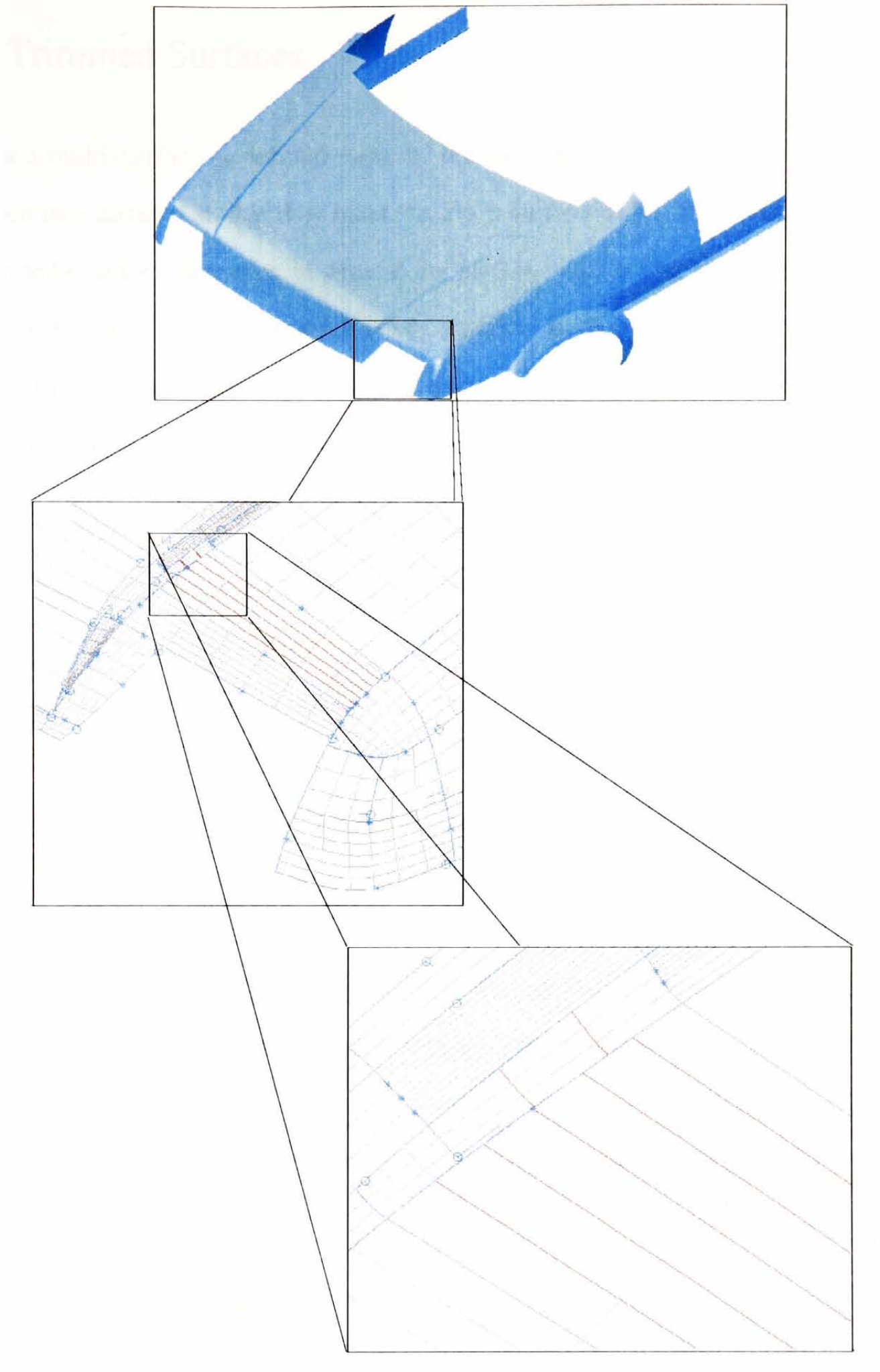


Figure 6.5: Parameterisation on neighbouring surfaces not necessarily continuous

6.3 Trimmed Surfaces

When a multi-surface is defined some of the surfaces will typically need trimming. To trim two surfaces so that they meet exactly requires that their intersection curve be generated exactly. However, in general the package will not be able to represent that curve and so it is usually approximated, typically by a poly-line. A poly-line is a series of points that lie on the intersection curve, the trimmed surface is defined by the linear interpolation of those points in the parameter space of the surface being trimmed.

Inexact trimming leads to an inaccurate definition of the part and consequently creates difficulties when generating cutter paths for it. We simplify the problem by taking cross-sections and noting that these cross-sections can be characterised by three inherently different configurations as illustrated in Figure 6.6. Also note that the configuration type of a cross-section may change along the length of the intersection curve of the two surfaces. The lines in the figure simply represent sets of points and not necessarily distinct sections. Type A cross-sections result from both surfaces extending past their intersection, type C cross-sections result from both surfaces being trimmed short of their intersection and type B cross-sections result from one surface extending past and the other being trimmed short. The next two sub-sections investigate the three types of inexact trim for both the concave and convex cases.

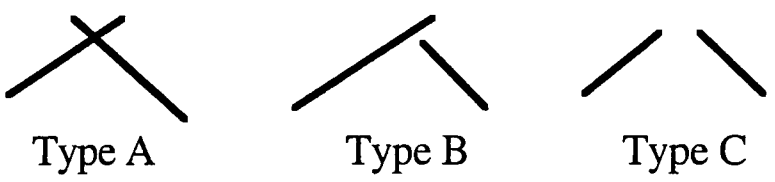


Figure 6.6: The three different types of inexact trimming

6.3.1 Inexact Concave Trim

The IOM handles inexact concave trims predictably and in most cases ideally. Figure 6.7 shows the path generated by the IOM for each case, which in general is identical to the path that would have been generated even if the patches had been trimmed exactly.

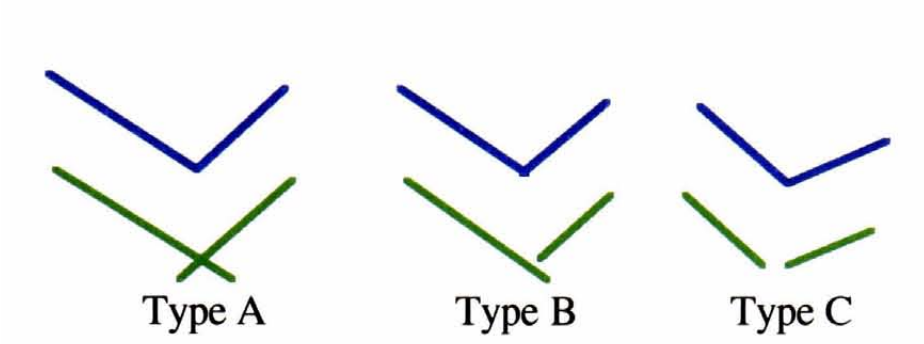


Figure 6.7: Result of offsetting inexact concave trims

The only instance in which the ideal path is not generated is if the trimming process creates a relatively large gap. We now calculate the maximum size the gap can be but still ensure that the IOM generates the ideal path. We only consider the case in 2D and assume that a nest of planar sections defines the surface. Also, since we are only considering a small region, we assume that the two surfaces can be locally approximated by planes. Figure 6.8 shows the cross-section of the cutter as it touches the both sides of the joint.

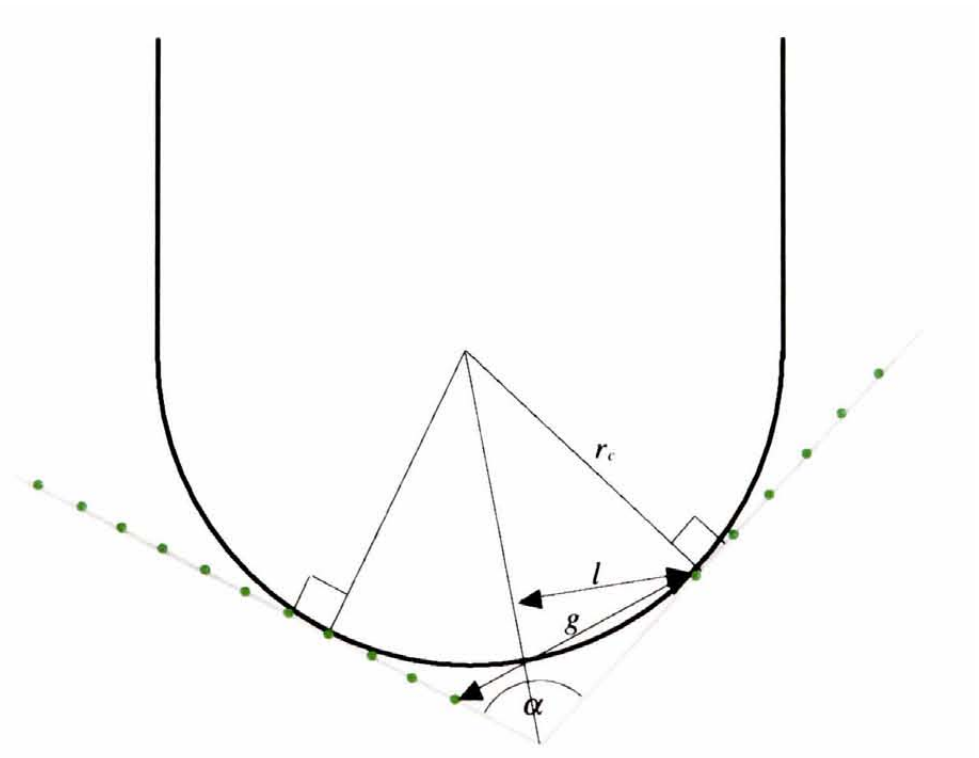


Figure 6.8: The maximum gap between two points at a concave joint

If the gap, g, is less than or equal to l then the IOM is guaranteed to generate the ideal path as illustrated in Figure 6.7. We can derive l from Figure 6.8:

$$l = r_c \sin\left(\frac{\pi - \alpha}{2}\right) \tag{6.1}$$

As an example, if we are machining with a tool of radius 5mm and at some concave joint we have $\alpha = \pi/2$ then:

$$l = 3.536\text{mm}.$$

If $l < g < 2l$ then the IOM may generate paths that lead to the gouging of the surface. If $g \geq 2l$, then the IOM is guaranteed to cause gouging.

6.3.2 Inexact Convex Trim

The IOM also handles inexact convex trims predictably but not generally ideally. The cross-sections of the offsets generated are shown in Figure 6.9. The thick blue curves represent the generated offset and the thin blue curves represent the offset that would have been generated had the surfaces been accurately trimmed.

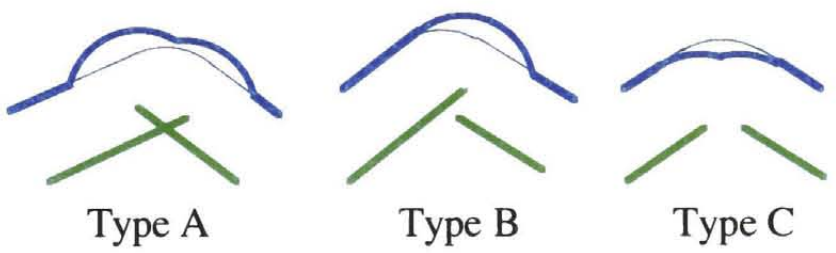


Figure 6.9: Result of offsetting inexact convex trims

As is immediately obvious from Figure 6.9, the result of offsetting a convex joint that has been inexactly trimmed is severe. However this is an issue that all CAM packages must tackle and so a solution may be derived from literature that addresses the issue for parametric methods although non are currently known of. Until such a solution is found it is important to ensure that any convex joints are trimmed to a tight tolerance.

6.4 Machining Strategy

So far in this thesis we have only been concerned with generating raster cuts. However there are many other types of finishing cuts used in industry. Therefore in this section we look at a selection of algorithms for generating: constant z-height cuts [Vadlamudi 1998], parametric cuts [Loney and Ozsoy 1987] and pencil-curve cuts [Park *et al* 1998]. Also, as in section 4.3, we observe that the grid-like structure imposed on the cutter paths is not a necessary restriction for any of the theory presented thus far and hence to facilitate generating economic cutter paths we allow cutter passes to contain any number of grid points.

6.4.1 Constant Z-Height Cuts

In this sub-section we present three approaches to generating constant z-height cuts highlighting the strengths and weaknesses of each. Note that when referring to z-height cuts we assume that the tool-axis lies parallel to the z-axis.

Cut-Off Technique

We start by generating the discrete definition of the offset surface by using the IOM. The offset points are then compared to the z-height plane in question and each point is determined as lying either above or below this plane. In Figure 6.10 the points that lie above the plane are coloured red and those below are coloured yellow. Joining the inner boundary of those points that lie below the plane forms the approximation to the contour. Doing so does not guarantee the surface will not be gouged but will minimise such gouging.

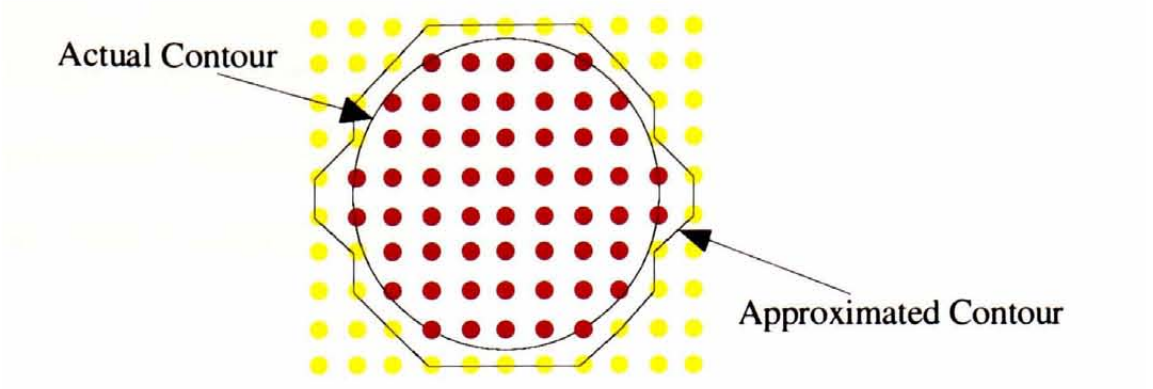


Figure 6.10: The contour generated by the cut-off technique.

This method is robust and provided that the offset grid covers the entire part and is sufficiently dense it will not miss any parts of the contour. The main weakness of the cut-off technique is that the contour is restricted to interpolating points that lie directly above the original offset grid which leads to aliasing effects, i.e. the sampling interval leads to irregular patterns in the contour. In Figure 6.10, this is most significantly apparent at the sides where the approximation to the circular contour pokes out.

Interpolation Technique

The interpolation technique is effectively an adaptation of the cut-off technique but with the quality of the contour improved by using a linear approximation of the offset surface. Again we start by generating the discrete definition of the offset surface by using the IOM. Then the offset data set is triangulated using a pattern like in Figure 6.11 and intersected with the appropriate planes to generate the constant z-height cutter paths. Each point in each pass is generated by the intersection of the plane with either an edge or a vertex of the triangulation.

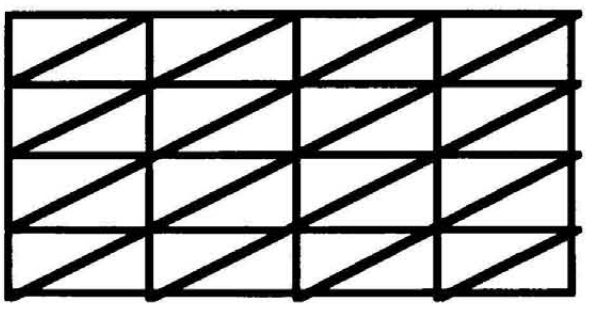


Figure 6.11: Possible triangulation pattern to be used on offset grid

The strengths and weaknesses of this method are akin to those for the cut-off technique; the method is robust but suffers from aliasing effects. The effect is most significant near curved vertical surfaces as highlighted in Figure 6.12a which shows the aliasing that occurs for a cylinder with a flat cap. The blue surface represents the triangulated offset grid and the red plane indicates the z-height. Figure 6.12b shows the top-view in which the effect can be seen clearly and can be compared to the contour generated by the cut-off technique in Figure 6.10. However the error of the interpolation technique is less than that of the cut-off technique because the contour points are generated from the triangulation of the offset data and are not restricted to lying directly above the offset grid.

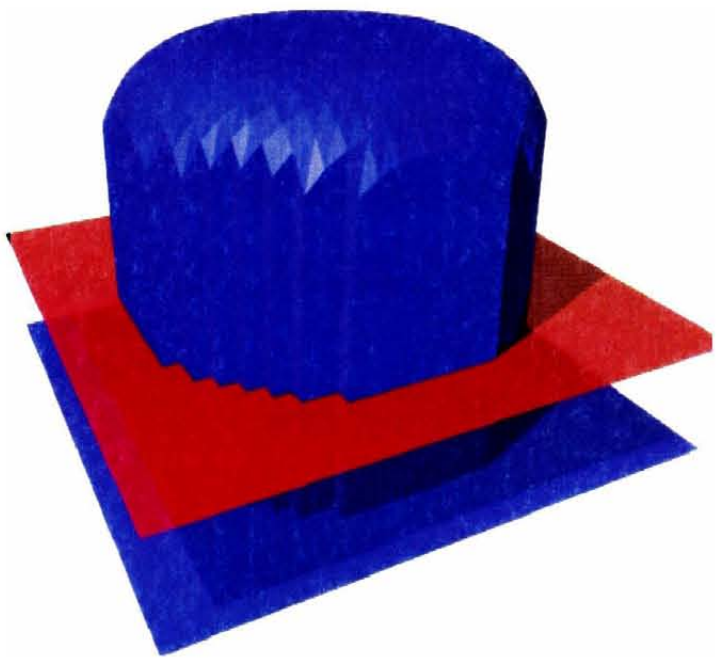


Figure 6.12a: Constant z-height contour generated by triangulation of offset grid

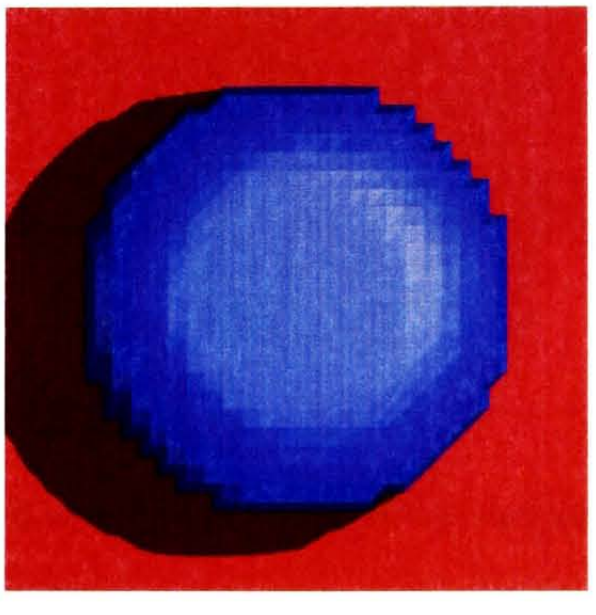


Figure 6.12b: Top-view of contour generated by triangulation method

Multiple Offset Directions Technique

By using the IOM to offset a part in a direction perpendicular to the z-axis we can generate an offset curve with constant z-height. Figure 6.13a highlights how a line of grid points is offset in the positive x direction to generate a constant z-height cut for the green cylinder. The points are moved onto the blue cylinder, which represents the offset of the green cylinder. As can be seen more clearly from the top-view in Figure 6.13b, only part of the contour is generated. We then offset again in another direction.

After offsetting in enough directions to ensure the whole contour is defined, any points that have not been offset, called non-offset points, are eliminated. The grid points are then concatenated to form a single continuous path. An algorithm to concatenate the separate paths has not yet been devised and will require further work. In both Figure 6.13a and Figure 6.13b the non-offset grid points are indicated by the points that do not lie on the offset contour/surface. We now consider the number of offsetting directions required.

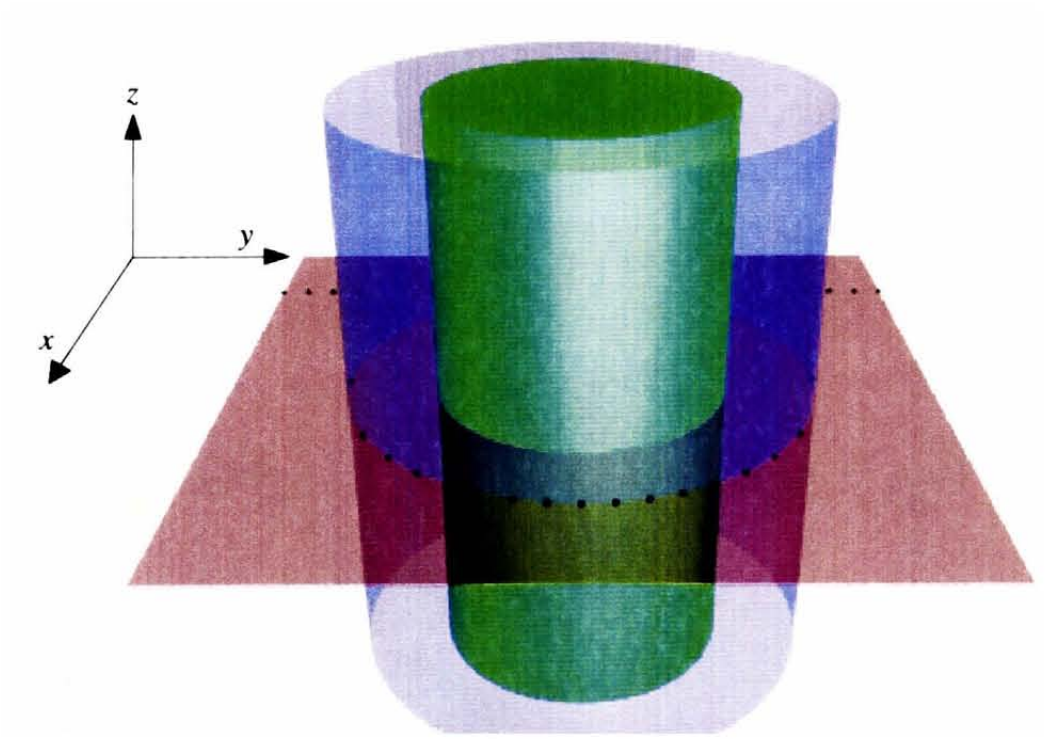


Figure 6.13a: Contour generated by offset perpendicular to z-axis

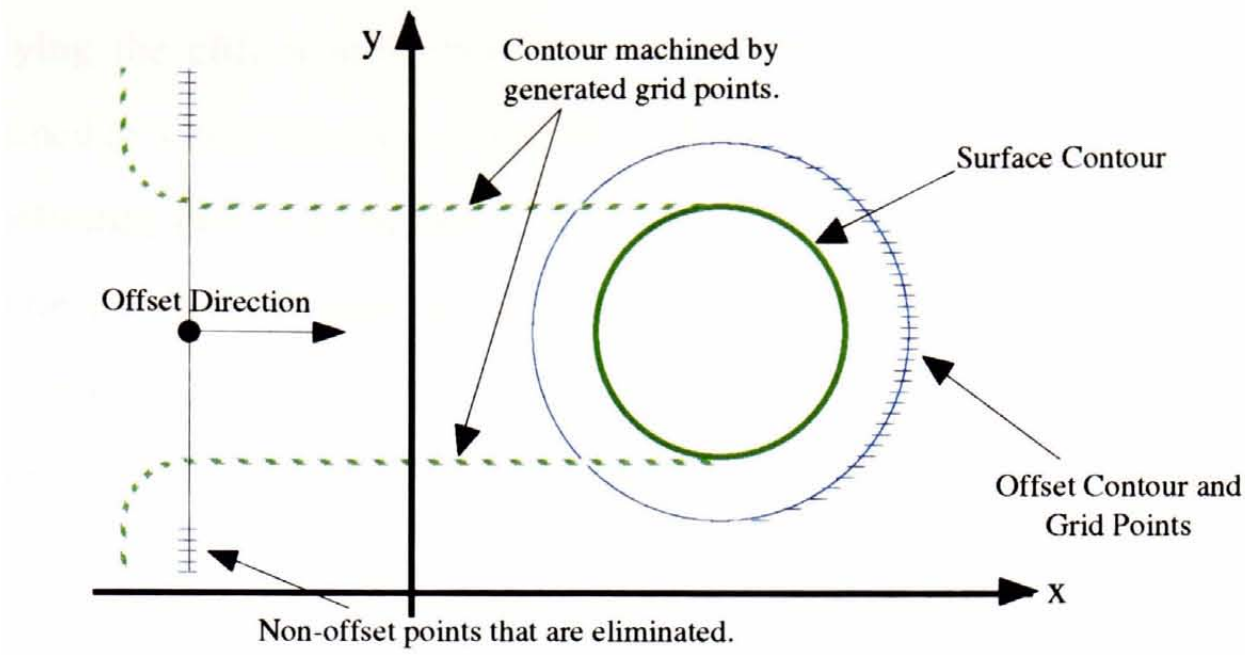


Figure 6.13b: Top-view of Figure 6.13a

To generate grid points along the entire contour the next logical step is to offset in the opposite direction to the first as illustrated in Figure 6.14. However if we consider the offsetting direction to be the positive z-axis then we are faced with a problem similar to the cliff problem as described in Chapter 4. The solution to the cliff problem, referred to as the 'cliff modification', works by placing one grid point just beyond the frontier and one just before. The resulting path then machines the part to within tolerance, Figure 6.15 shows the application of the cliff modification to the multiple offset directions technique.

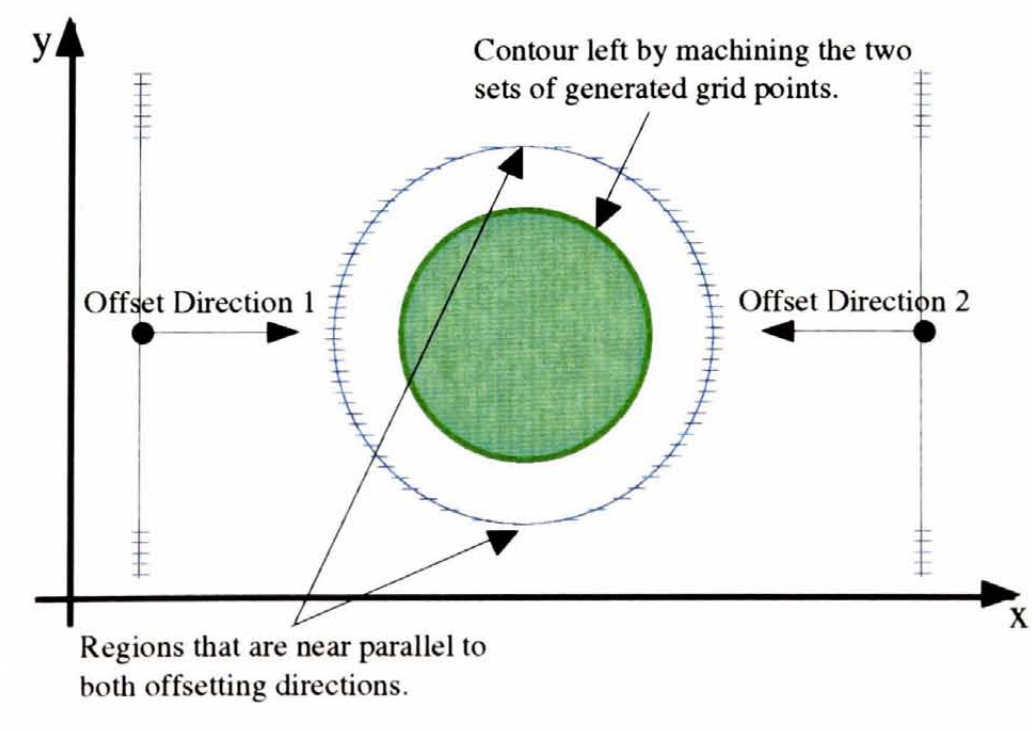


Figure 6.14: The contour generated by two offsetting directions

Applying the cliff modification in both directions results in the contour being machined to within tolerance, Figure 6.15 illustrates the insertion of the points in just one offsetting direction. However we do not wish to machine two distinct sections for a single contour and therefore remove the non-offset points. Concatenating the remaining points generates the final contour. The contour is defined to within tolerance because of the points inserted in the local region some small distance from the frontier as illustrated in Figure 6.16. The figure shows the two new points improving the definition of the contour.

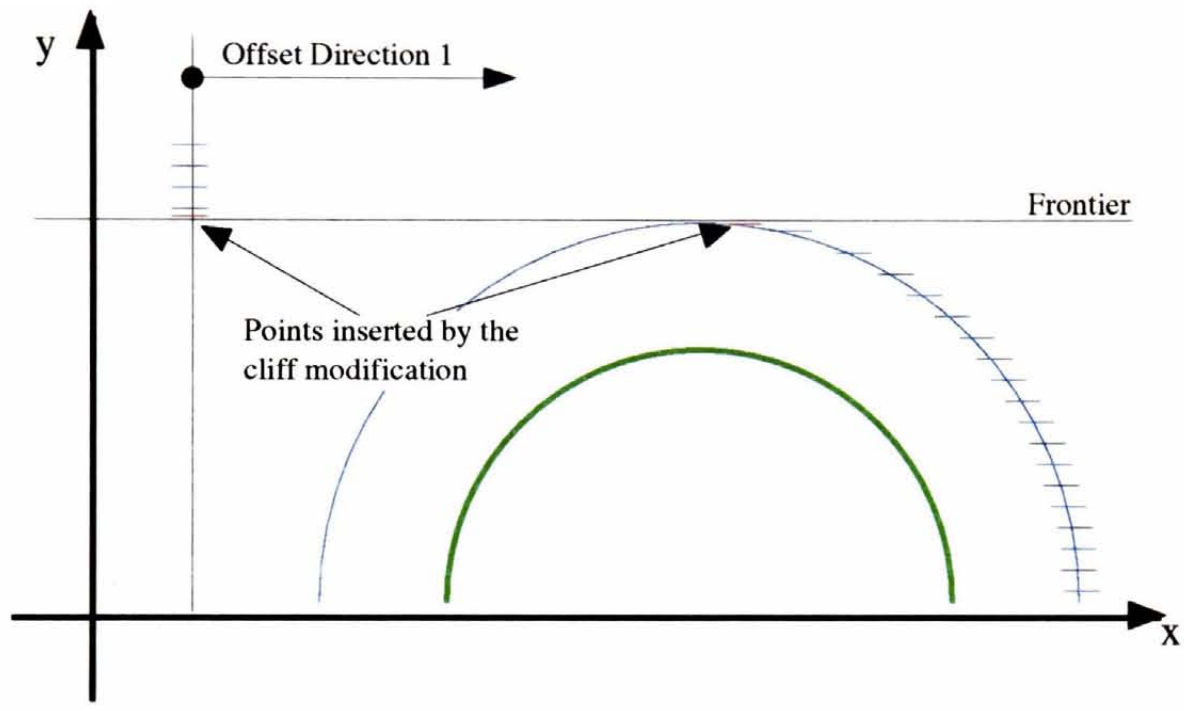


Figure 6.15: The cliff modification applied in one offsetting direction

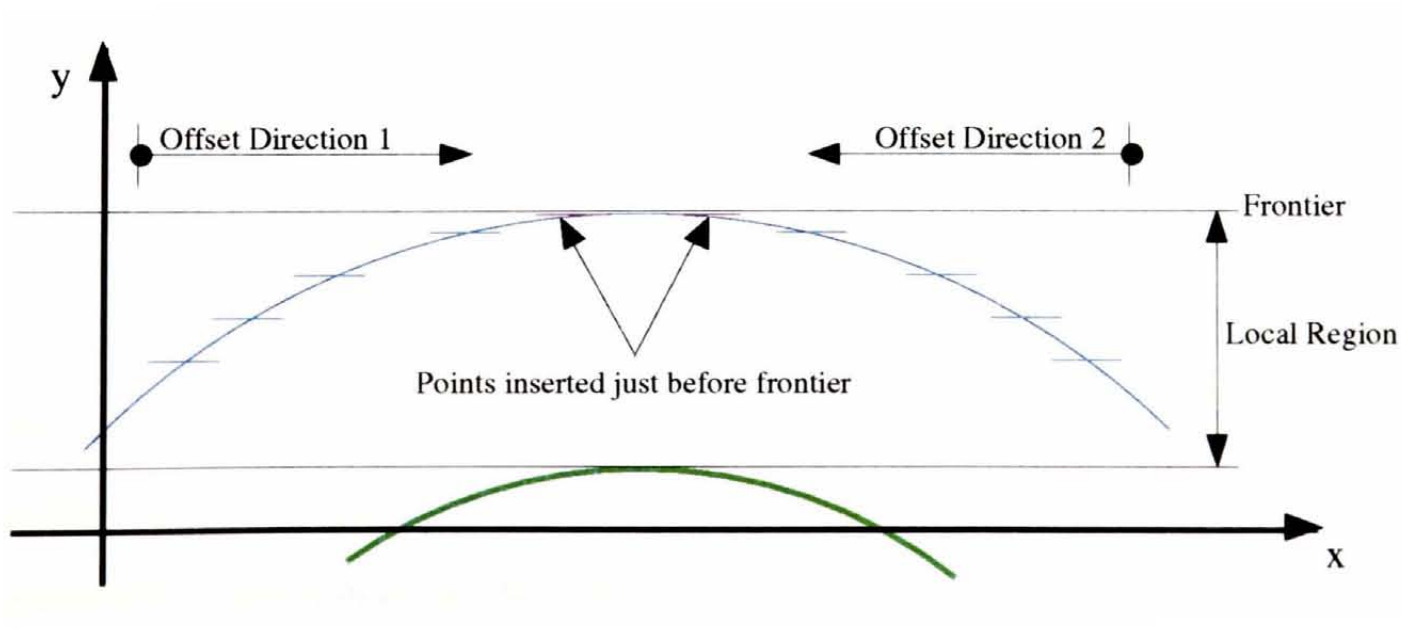


Figure 6.16: Final contour definition omits the non-offset points

However, there are times when two directions are insufficient. Figure 6.17 shows an example of a contour that requires three offset directions and indicates three suitable orientations for them. Further work is required to automate the process of selecting the most suitable number of offset directions and their orientation. Figure 6.18 shows an example of a contour where sufficient directions have been used but not orientated correctly.

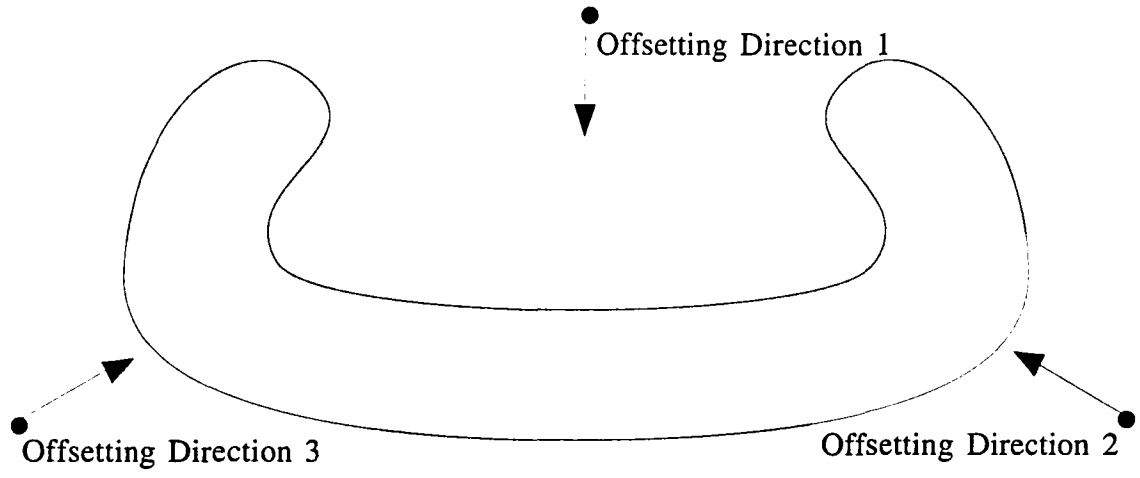


Figure 6.17: A contour shape that cannot be fully defined by using only 2 offset directions

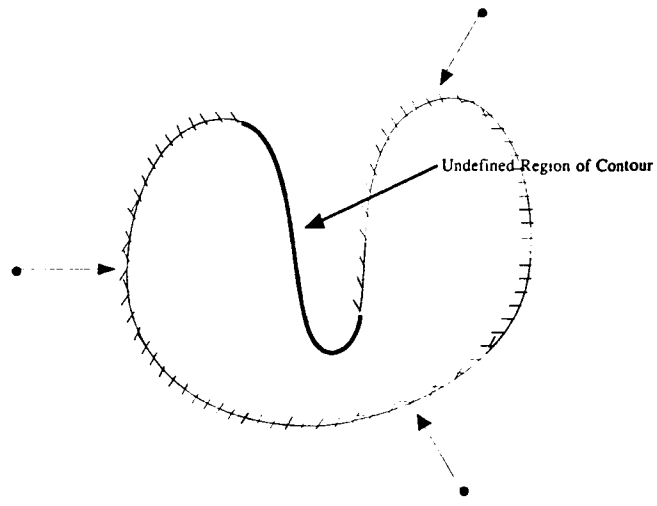


Figure 6.18: Incorrect orientation of the offset directions leading to an undefined region

Finally, there are some contour shapes that cannot be completely defined using this algorithm. The reason for this is there exists a region on the contour that for any given direction in the plane, the contour is not single-valued. That is if we consider the given offset direction as up, the undefined region will always contain an overhang.

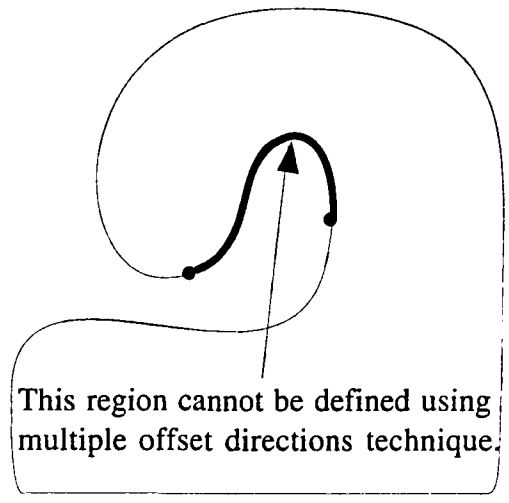


Figure 6.19: Example of a contour that contains an indefinable region

Assuming that the part shape contains no indefinable regions, to generate a complete set of contours we start by defining an offset grid for each offset direction. Each grid is perpendicular to its associated offset direction and has a row of points corresponding to each contour height. Each grid is then offset as described above to form the contour paths.

Note that the assumption that the part contains no vertical overhangs is also key to the application of this technique. If a part does contain an overhang then unlike for the cut-off or interpolation technique, the surface will be gouged as illustrated in Figure 6.20.

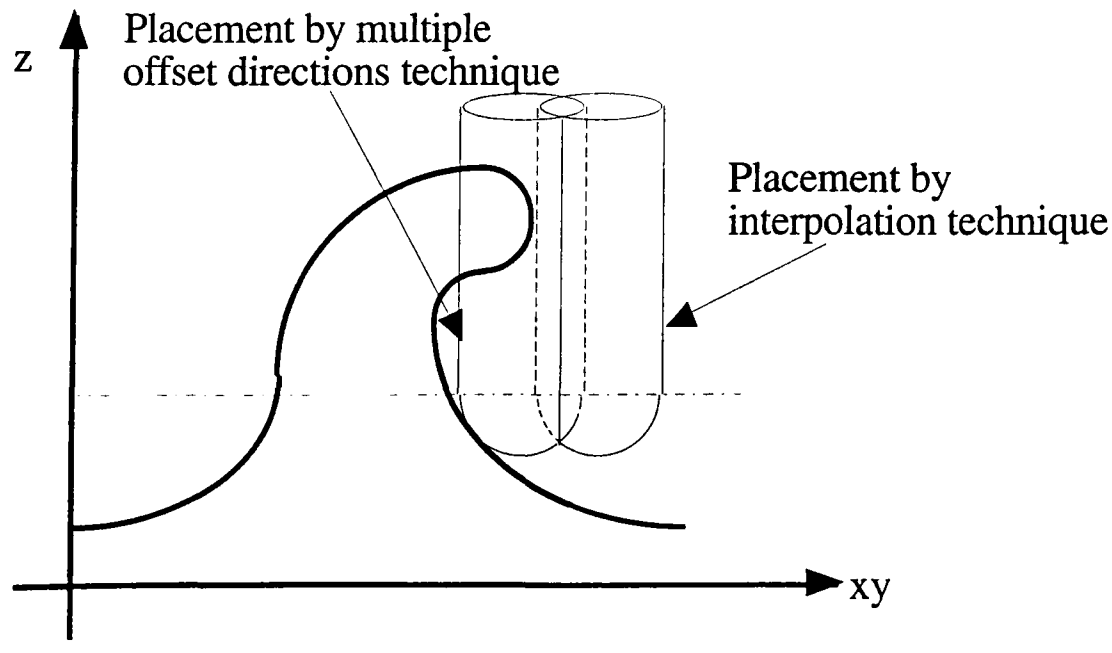


Figure 6.20: Overhang leads to algorithm failure

The strength of this method is that no additional approximation has been introduced, only the error as calculated for the IOM in Chapter 3 needs addressing. However there are a number of potential problems with using this technique. Firstly the cutter path points in the contour will have irregular spacing due to the multiple sections used. Also the procedure will prove computationally expensive since each set of contours requires at least two offsets. The most restricting feature however is the fact that this technique will fail on some non-convex or multi-path contours as illustrated in Figure 6.19.

Summary

In summary the interpolation technique is the most attractive option. It offers better quality contours for a given offset data set than the cut-off technique. Therefore it should prove more efficient than the cut-off technique because offsetting is a computationally expensive operation although further research is required to verify this.

The interpolation technique is better than the multiple offset directions technique for two reasons, firstly because it handles vertical overhangs appropriately whereas the multiple offset directions technique leads to the part being gouged. Secondly, the multiple offset directions technique cannot guarantee generating a complete contour due to indefinable regions.

6.4.2 Parametric Cuts

Although there are numerous difficulties associated with parameter-based machining patterns as mentioned in Chapter 2 and section 6.2, it may on occasion be the most appropriate choice. Hence we include a method of generating such cuts that employs the IOM. For the following work we assume the surface points have been generated at regular parametric intervals.

The standard way of generating parametric cuts is to calculate the normal at each surface point using the parametric definition and then generate the CL-data. Hence the analogous method would be to approximate the normal at each surface point using neighbouring surface points and then generate the CL-data.

However, we offer an alternative that generates pseudo-parametric cuts and is consistent with the IOM. To do so we define the lifted offset grid, an offset grid where the grid points are generated by projecting the surface points along the tool-axis onto the offset grid plane, i.e. when viewed along the tool-axis the grid points line up exactly with the surface points. The IOM is then used to generate the cutter paths. The main difficulty of using lifted offset grids lies at the boundary of the part. This is because the grid points will end wherever the definition ends. It is therefore necessary to extend the surface definition to include a cutting plane, see section 4.1, which will provide grid points beyond the boundary of the part. However the transition from above the surface to above the cutting plane may not be ideal since the part and cutting plane are defined distinctly from each other. Figure 6.21 illustrates the different paths generated by the standard parametric method and the lifted offset grid method. Note that, although not indicated in the figure, the lifted offset grid method will benefit from the cliff modification.

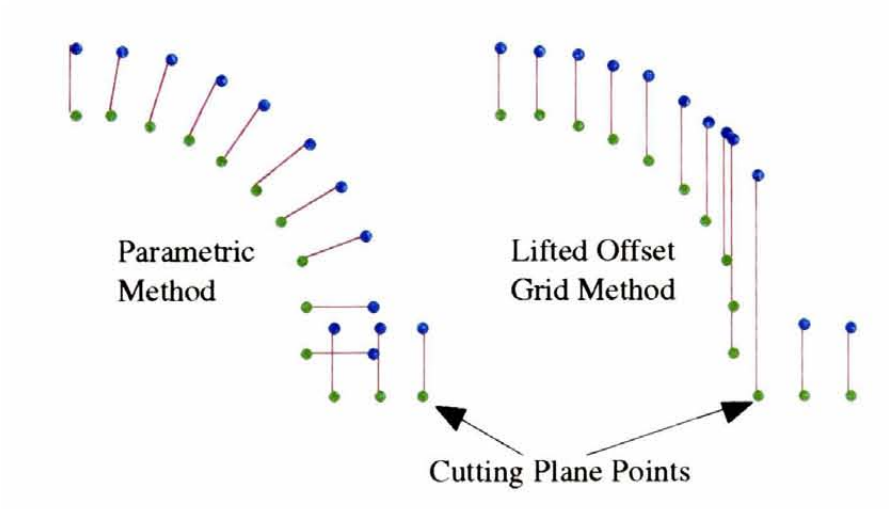


Figure 6.21: Two ways of generating parameter based cutter paths

6.4.3 Pencil-Curve Cuts

Pencil-curve machining consists of just a single pass that traces along tight concave edges of a surface. This operation is widely used in die-surface machining to either relieve the load a tool experiences in a subsequent cut or to clean-up uncut volumes in concave regions where previous cuts using larger radius cutters have missed. We give two alternative methods both taken from the literature and consistent with the IOM for tackling this problem.

The first method, described by [Saito 1991], can only be applied to generating clean-up pencil-curves and relies on the accurate identification of uncut regions. Firstly the IOM is used to generate the cutter paths for a given tool, Figure 6.22 shows a cross-section for one of the paths. The cutter paths themselves are then offset using the IOM to regenerate the surface definition. Any regions that do not correspond directly with the original surface indicate regions that cannot be accessed by the original tool.

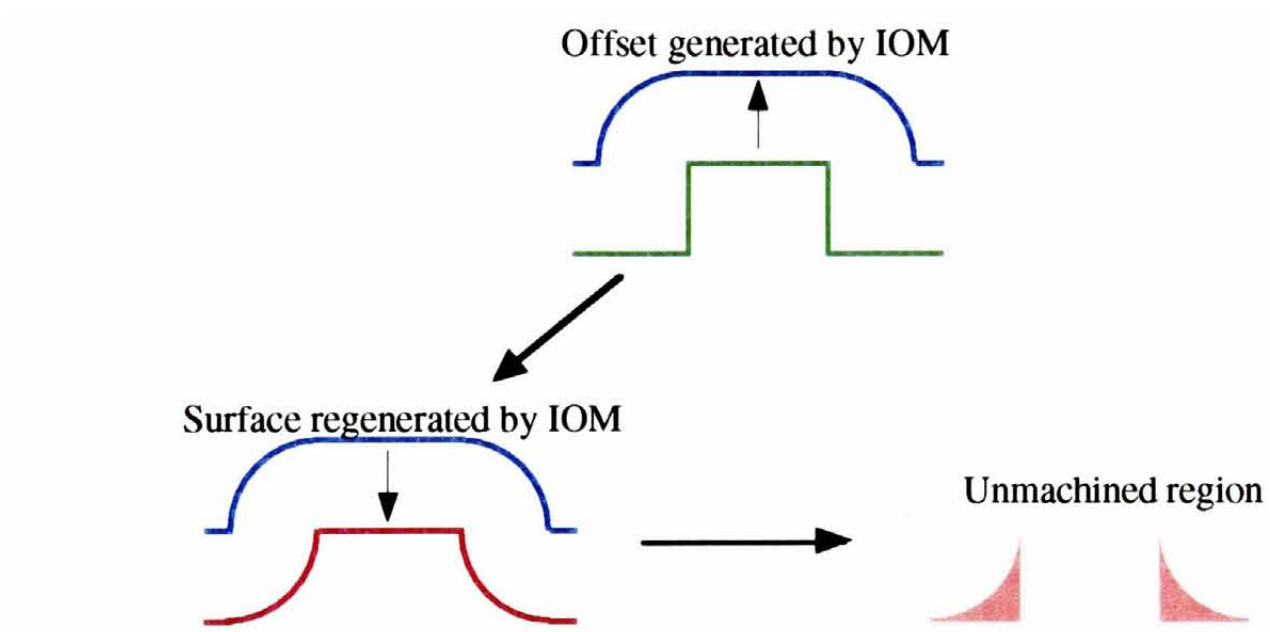


Figure 6.22: Identification of unmachined regions

These regions are then evaluated using a variable called the 'static milling volume' to determine whether they contain enough unmachined material to warrant a pencil-curve cut. If they do, then the peak values of unmachined volume are traced to define the cut. The author does not give typical values for the static milling volume and so although the method itself is simple to implement, further research is required to

establish a suitable threshold value for this variable above which it is desirable to machine the region.

The second method by [Park *et al* 1998] uses the IOM to generate the offset surface and then evaluates the quality of each offset point as a pencil point, i.e. a sharp concave point. The points are classified as Gold, Silver, Bronze or Clay; Gold representing a very good pencil point and Clay the points that are least pencil-like. Strings of good quality pencil points are generated to form pencil-curves that are then smoothed to generate the final pencil-curves. An adaptive sampling scheme is included as an option to efficiently increase the accuracy of the method.

The method by [Park *et al* 1998] includes a number of heuristic tests each of which requires a total of 9 'magic numbers' to be defined by the user. Though default values are suggested, these are motivated by the generation of cutter paths for stamping-dies in the automotive industry and hence to implement this method, extensive testing may be required to establish suitable values for the application in question.

To summarise both methods offer effective pencil cut generation routines and both are built on the IOM. Also the error analysis from Chapter 3 should prove helpful in establishing their relative efficiency and accuracy.

6.5 Sampling Interval

Crucial to the quality and efficiency of the IOM and the cutter paths generated is the optimisation of the sampling interval of the points. In this section we only consider optimising the sampling interval of the grid points and not the surface points. It should be noted however, that the surface definition could be recursed, i.e. filled in, if a denser set of points is required.

In sub-section 6.5.1 we restrict ourselves to two dimensions and consider only a single planar pass of the cutting tool. In sub-section 6.5.2 we highlight the consequences of extending naively from 2D to 3D by considering a number of neighbouring planar passes.

6.5.1 Sampling Theory

Cutter Path Point Frequency

In sampling theory the objective is typically to represent accurately the original pattern of some analogue data using discrete data. For example when digitising sound, the sampling rate should be sufficiently frequent to pick up the highest frequency contained within the sample. Nyquist's theorem states that the frequency of the sampling rate should be at least twice that of the highest frequency contained within the original sample. Figure 6.23 shows the result of sampling with a frequency less than this. The wavelength exhibited by the sampled points, λ_s, is three times that of the original, λ_o. Using the analysis contained in Chapter 3 it is assumed that the surface points are sampled at sufficient frequency to accurately represent the surface.

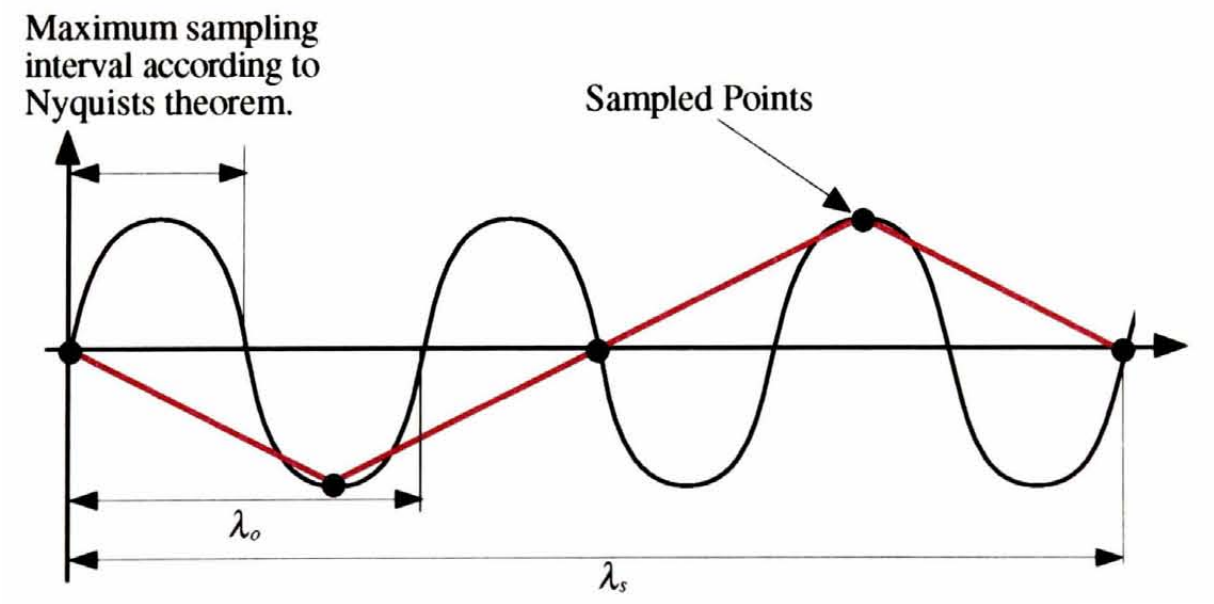


Figure 6.23: Result of sampling less frequently than suggested by Nyquist's theorem

However we are faced with a different objective from that of typical sampling theory for the cutter path points. The underlying pattern in the offset surface will reflect the shape of the inverted cutter, Figure 6.24 shows an example of this.

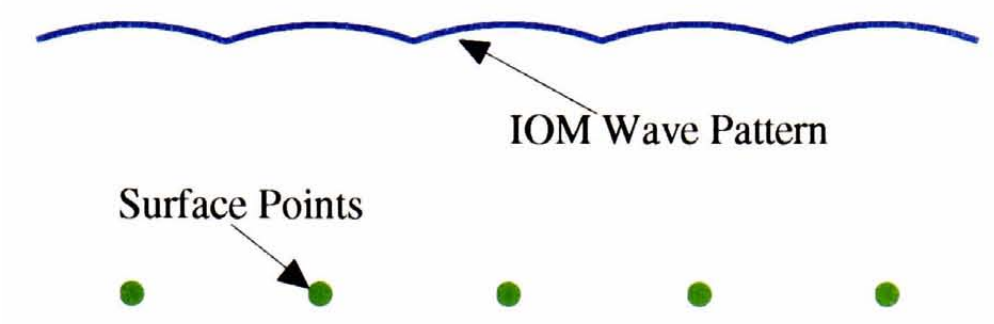


Figure 6.24: Wave pattern created by using the IOM with a ball-nosed cutter

The objective here is to avoid picking up the shape of the inverted tool. Hence using Nyquist's theorem we can deduce that:

$$\frac{1}{c} < \frac{2}{s} \Rightarrow c > \tfrac{1}{2}s \tag{6.2}$$

where c is the cutter path point separation and s is the surface point separation. The effect of the wave pattern can be further reduced by requiring:

$$c = is \qquad \text{where } i \in Z^{+} \tag{6.3}$$

Figure 6.25 shows the result of letting c=2s, as can be seen the result is that the sampled points completely miss the tool shape. In practice the wavelength will not be constant due to the surface varying in shape and gradient. However by making the ratio, s:c, as large as possible while satisfying the tolerances given in Chapter 3 we help even out the distribution of the error and hence improve the surface finish.

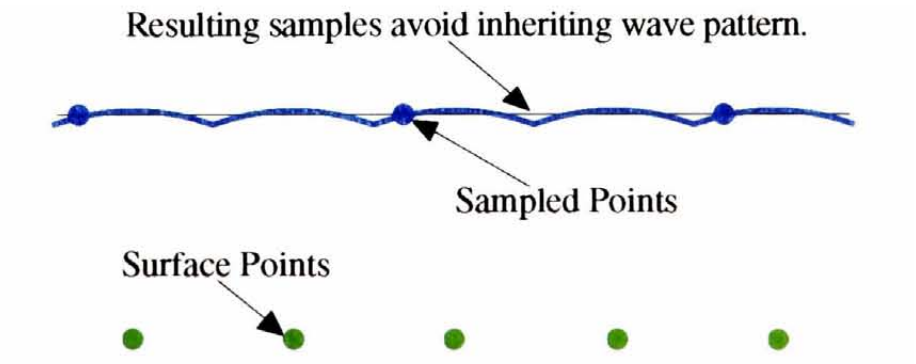


Figure 6.25: Relation (6.3) results in reducing the effect of the IOM wave pattern

Cutter Path Point Distribution

If we are generating paths for a part shape that contains flat regions and highly curved regions then selecting a single sampling interval for all of the cutter path passes will lead to high inefficiency. The work presented here can be seen as an extension to the work presented in Chapter 4 on the cliff problem. However instead of simply increasing the density of the cutter path points to handle tangent discontinuities, i.e. regions with infinitely large curvature, we propose a graded map of the part where the gradient of a region determines the density of the grid points. The grade is a function of the local surface curvature, normal direction and the cutter radius.

Figure 6.26 shows the cross-section of an example extruded-surface and the 2D version of a grade map. Grade 1 signified by a white region corresponds to a practically linear region of offset and hence needs no internal grid points at all, just one at either end of the region. The next three grades have increasing grid point density, up to grade 4 that is also suitable for tangent discontinuities. Using the analysis in Chapter 3 we can determine a suitable grid point separation for each region. The optimum number of grades that should be used to define a grade map requires further investigation but relation (6.3) from the previous section naturally determines the minimum intervals each grade should have.

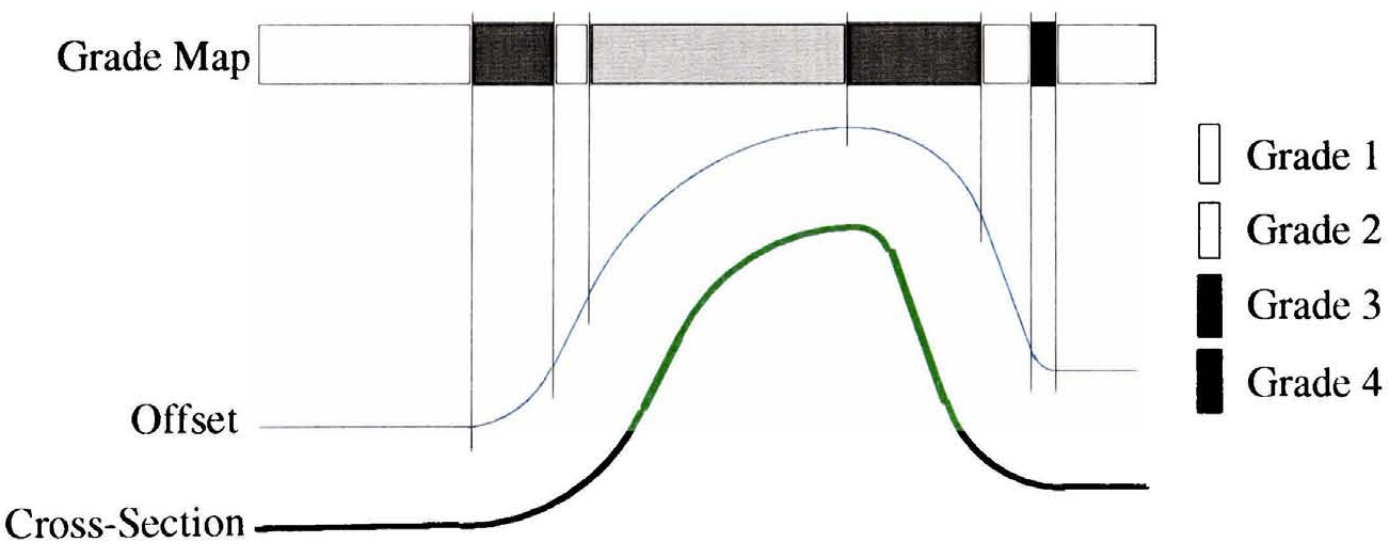


Figure 6.26: Cross section of extruded surface and grade map determining grid point separation

6.5.2 Extension to 3D

Using the above method each pass can be generated in isolation and cutter paths generated to within tolerance. However, we now highlight the result if care is not taken when generating multiple neighbouring passes. Suppose by using the method described in sub-section 6.5.1 we generate a grid for a part. Depicted in Figure 6.27 is a portion of this grid for a spherically shaped part of the surface.

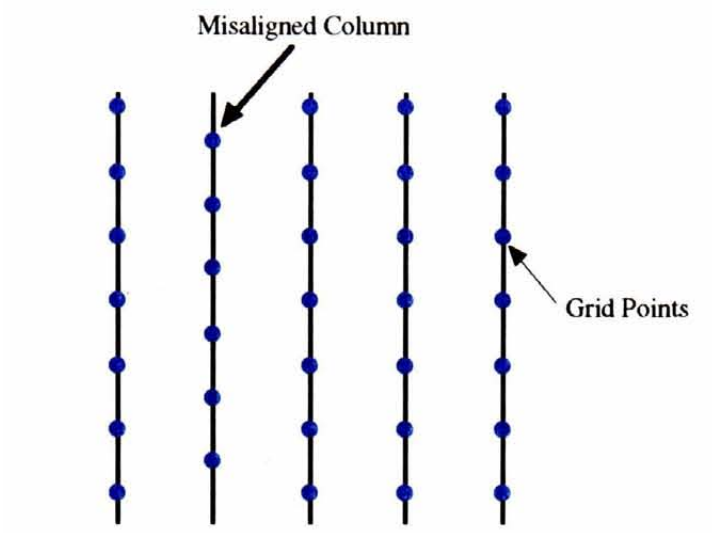


Figure 6.27: Example grid generated by method described in sub-section 6.5.1

The misalignment in column 2 could occur because of the shape of the part beyond this portion of the grid. Figure 6.28 shows a simulation of machining these cutter paths. As can be clearly seen the misaligned grid points result in the undesirable feature of a cusp line that zigzags across the surface. Even the aligned grid points result in a slightly stepped effect in the cusp line that is highlighted by its shadow.

This result is also apparent in the machining of the shoe produced by the IOM. In Figure 6.29, the circled region in both the picture and the simulation contains evidence of the effect artificially created in Figure 6.28. Solving this problem is not trivial and requires further research to provide a solution.

Figure 6.28: Simulation of machining using grid from Figure 6.27

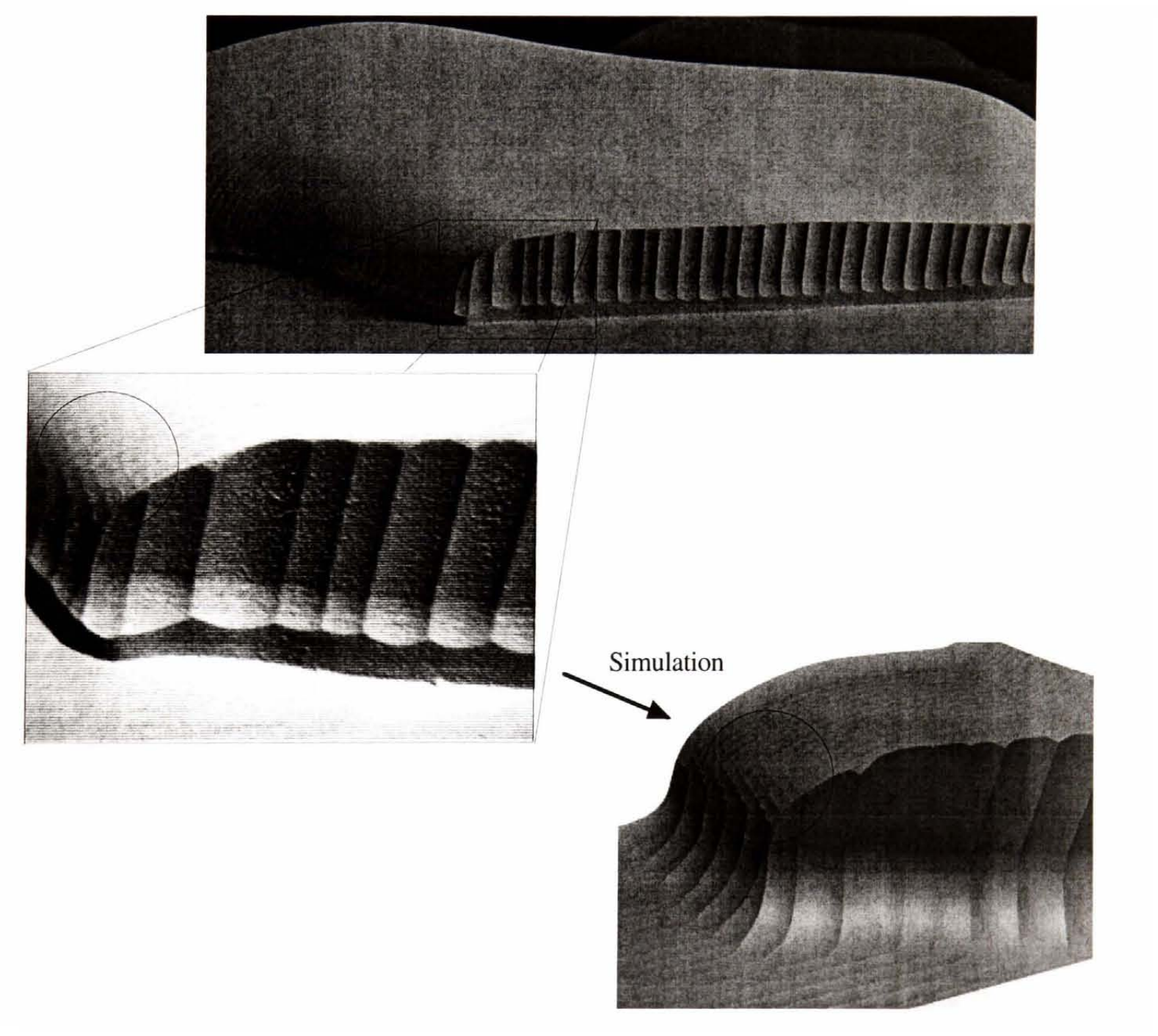


Figure 6.29: The machining of the shoe and its simulation

6.6 Summary

In this chapter we have briefly covered a number of issues associated with creating a CAM package capable of generating cutter paths for multi-surfaces. Undoubtedly there are other problems but we have restricted ourselves to considering those which are generic to all CAM packages.

It was found that in many cases the geometric nature of the point-based algorithm provided an immediate solution. In the remaining cases the results of preliminary investigations were given but still require further research to provide appropriate error analysis.

Chapter 7

Conclusions

In this thesis we have developed an algorithm for efficiently generating good quality cutter paths for a part whose shape is defined by a set of points. This work builds on the point-based approach to CAD/CAM and is motivated by a sensible consideration of the tolerances used throughout the design and manufacturing process. Current industrial practice is based on the parametric definition, which is regarded as being an exact definition even though it is the culmination of inexact design processes and the origin of inexact manufacturing processes. Our objective has been to provide the theory required for adding machining capability to a point-based CAD/CAM package.

The research began in Chapter 3 with an evaluation of the Inverse Offset Method as a possible algorithm for generating cutter paths from sets of points. An error analysis was performed and its accuracy confirmed by the machining of an actual example. For the error analysis it was assumed that a ball-nosed cutter would be used. Figure 7.1 shows the error for a ball-nosed cutter as calculated in sub-section 3.1.3. Extending the analysis to include other shaped tools remains an outstanding issue. It is anticipated that the main difficulties of such a generalisation would centre around tool profiles containing a tangent discontinuity as in the case of the flat-end cutter. Figure 7.2 illustrates the maximum error that results from machining with a flat-end cutter for a point set identical to that as shown in Figure 7.1 for the ball-nosed cutter.

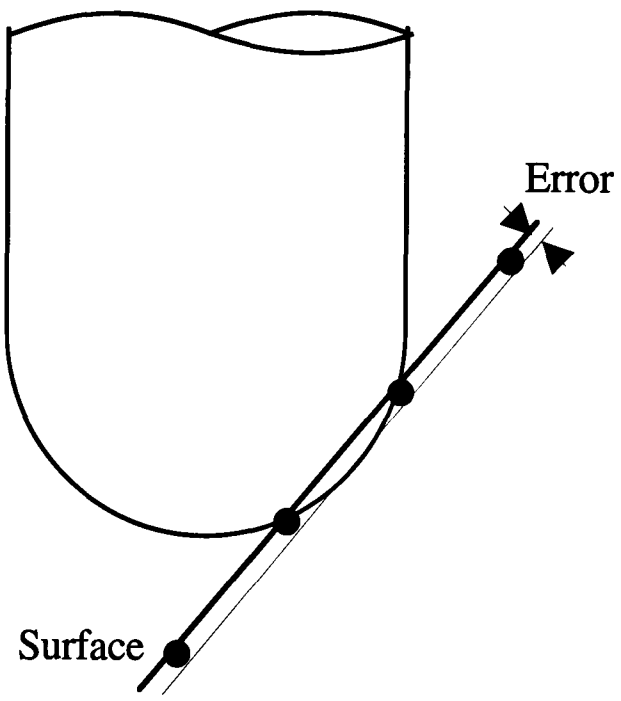


Figure 7.1: Error that results from using a ball-nosed cutter

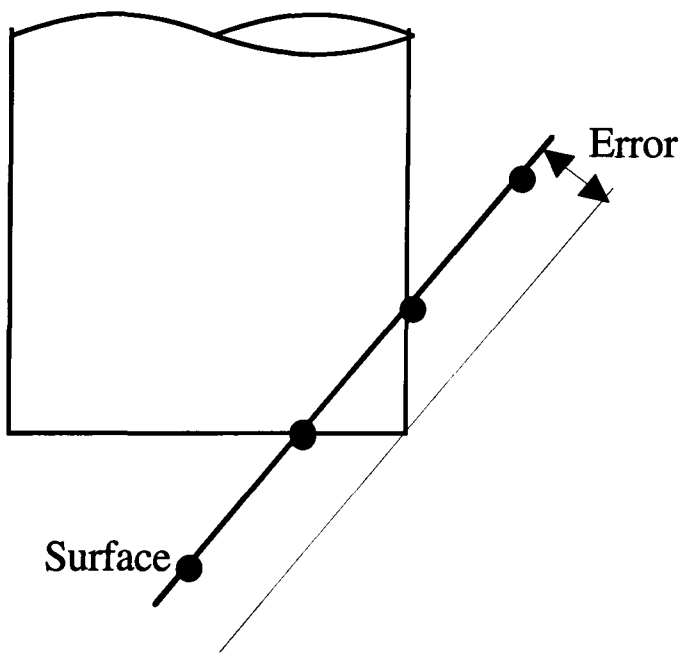


Figure 7.2: Error that results from using a flat-end cutter

In Chapter 4 we provided a modification to the IOM that allowed the machining of a cliff-edge to within a tolerance consistent with the rest of the part. Currently the modification is only applicable to cliff-edges that lie between the part and the cutting surface but the work could be adapted so as to be applicable to all tangent discontinuities. The key to such a generalisation lies in the locating of regions requiring special attention. Since the part is discretely defined it is anticipated that such an algorithm would need to flag regions containing sequences of points that exhibit curvature greater than some prescribed tolerance. Note that such a

modification is effectively just a sub-case of the theory presented in sub-section 6.5.1 in which it is suggested that an adaptive scheme be implemented for generating curvature dependent CL points to improve efficiency.

In Chapter 5 we presented four different optimisations for the IOM which all require that the point data be ordered in sections. We formulated the dual process which exploited the order of the cutter paths and not the surface data meaning that the optimised algorithms are applicable to all sets of surface points. Placing the constraints on the cutter paths is not restrictive because cutter paths generally lie in ordered sections. These optimisations were implemented and run on the test piece. The result was a significant improvement in performance; generation periods were achieved comparable to that of the commercial package. Further optimisations could be achieved by using a bucketing system such as that employed by [Drysdale and Jerard 1989] for their simulation algorithm. Alternatively by pre-ordering the data sets the optimisations could be used on all given point definitions.

Finally in Chapter 6 we considered some of the issues raised by multi-surfaces. At this point we reap a number of benefits from having employed a point-based strategy. The disassociation from parametric representations nullifies a number of difficulties that parameter-based CAD/CAM packages must still tackle, e.g. data exchange, correlation between parameterisation and shape, correlation of parameterisation between two surfaces. However two issues remain outstanding:

- Negotiation of inexact convex trims,
- Adaptive calculation of CL point intervals in 3D.

The latter issue is of prime concern and is crucial to the performance of any machining algorithm. The distribution of the points is critical to the surface finish and hence even though a part may be machined to within tolerance the result can still be unsatisfactory as illustrated in sub-section 6.5.2. This is an issue that must equally be

tackled by parameter-based algorithms which in turn may provide a comparable solution.

It is anticipated that a complete change to a point-based system may not be in the best interests of most companies and hence investigation into a hybrid system that incorporates both surfaces and points may be beneficial.

In summary we have developed a point-based algorithm that can efficiently generate good quality cutter paths. The IOM is presented as part of the point-based approach to CAD/CAM as an alternative to parameter-based approaches and has key advantages in reverse engineering, data exchange and manufacture. To conclude we have demonstrated that the point-based strategy is a feasible and positive direction for CAD/CAM from which the manufacturing industry should benefit.

Appendix 1

When sampling points using the CMM two software packages are used to generate the probe centre, in this appendix we derive the error introduced by them. The first package (Pack 1) outputs approximated contact points using the probe centre data provided by the CMM. However as explained in section 3.3, it is the probe centre, not an approximate contact point that is desired. Unfortunately an unknown algorithm is used by Pack 1 and so we cannot formulate the reverse process exactly. Instead we use a second piece of software, Pack 2, to approximate the probe centre.

We use a naïve algorithm to simulate the calculations performed by Pack 1 and assume that the error of our algorithm is greater than that of the software. It is assumed that Pack 2 introduces less error than Pack 1 because it has access to all the data given by Pack 1; whereas Pack 1 can only access previously sampled points from the CMM.

Illustrated in Figure A1.1 is the motion of the probe as it samples points along a profile. The probe starts at a prescribed clearance distance away from the surface. It then moves towards the surface until contact is made. The probe then moves away from the surface by the clearance distance and then along the tangent by the user set sampling distance. The process repeats until the user-defined end point is reached.

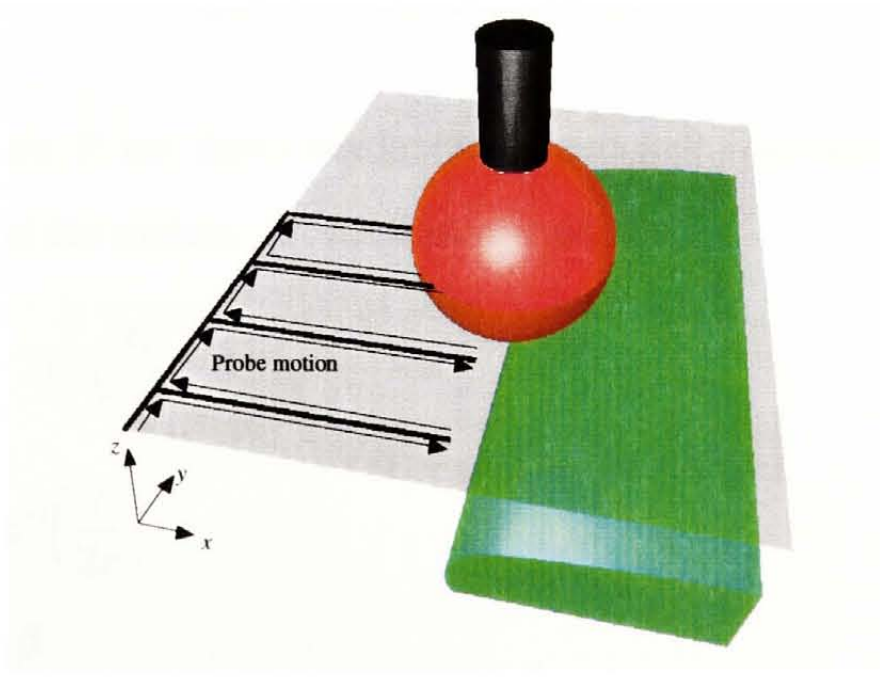


Figure A1.1: Illustration of the probe's motion

We now describe our simulation of Pack 1 and calculate an error bound for the approximation of the contact point.

Let $\mathbf{P}_0$ and $\mathbf{P}_1$ be the last two probe centre points sampled, r_s be the radius of the sphere used to approximate the surface locally and r_p be the radius of the probe. Figure A1.2 illustrates these parameters on an example. The tangent is approximated by joining $\mathbf{P}_0$ and $\mathbf{P}_1$. After retreating from the surface by the clearance distance, the probe moves along the approximated tangent by the sampling interval, l. Then the probe moves towards the surface, normal to the approximated tangent. When contact is made the new probe centre is used to approximate the contact point.

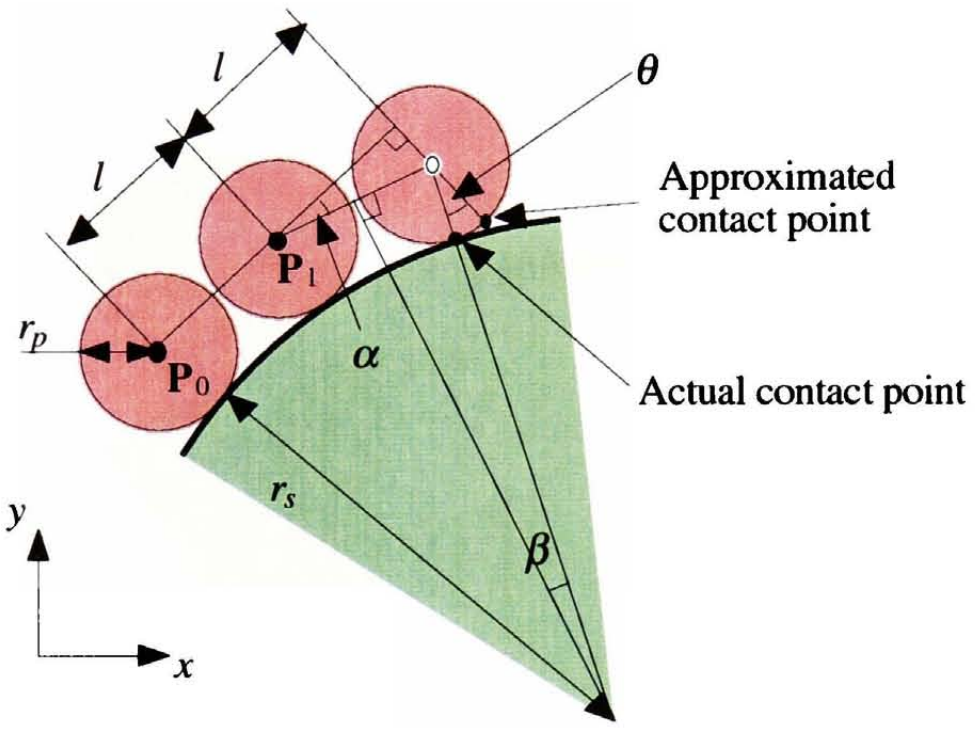


Figure A1.2: Illustration of parameters

First we calculate θ and then calculate the discrepancy between the approximated contact point and the surface. It can be shown that:

$$\alpha = \tan^{-1}\tfrac{1}{2}\left(\sqrt{\frac{4r^2}{l^2}-1}-\sqrt{\frac{4r^2}{l^2}-9}\right)$$

$$\beta = 2\sin^{-1}\left(\frac{l}{2r}\right)$$

$$\theta = \alpha + \beta$$

where $r = r_s + r_p$ if the surface is convex or $r = r_s - r_p$ if the surface is concave. To find the error bound appropriate for our measurements let l = 2mm, r_s = 16mm (which corresponds to a convex region) and r_p = 1mm:

$r = 17\text{mm}$

$\alpha = 6.7683^\circ$ (4 d.p.)

$\beta = 6.7445^\circ$ (4 d.p.)

$\theta = 13.5128^\circ$ (4 d.p.)

To calculate the error introduced by approximating the contact point we find the distance between the approximated point and the surface, Figure A1.3 illustrates this calculation.

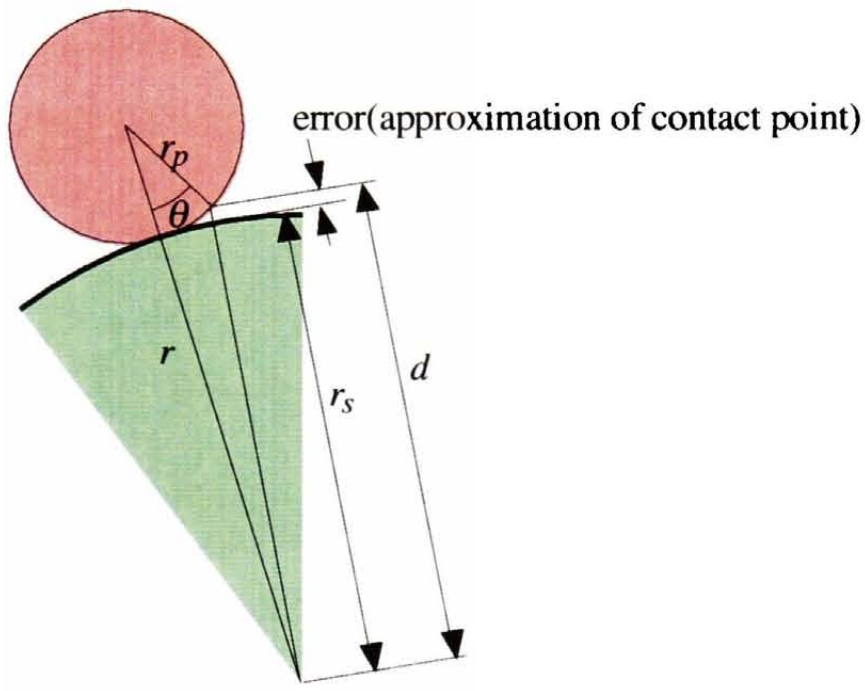


Figure A1.3: Calculating the error of the approximated point

Note that the corresponding calculation for a concave surface requires a slight modification to the procedure presented above. From the figure and the cosine law:

$$d = \sqrt{r^2 + r_p^2 - 2rr_p \cos\theta}$$

Again we use the parameters specific to our measurements to calculate the error:

$$\text{error(approximation of contact point)} = d - r_s$$

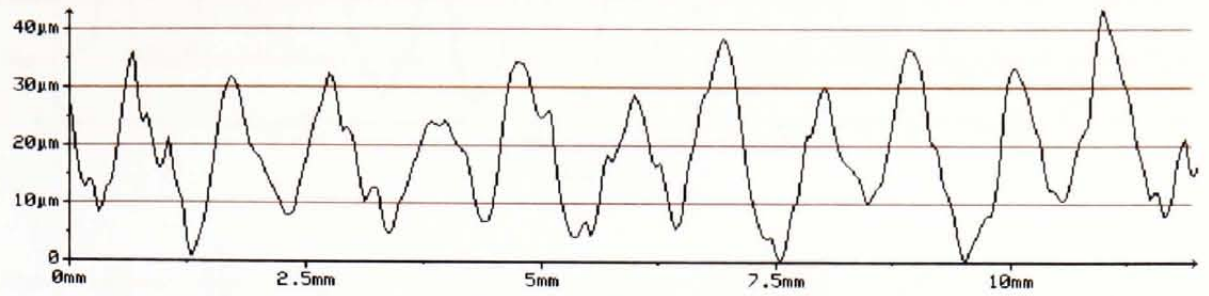
$$= 0.0294 \qquad (4 \text{ d.p.})$$

Assuming the error introduced by Pack 2 is less than the error introduced by Pack 1:

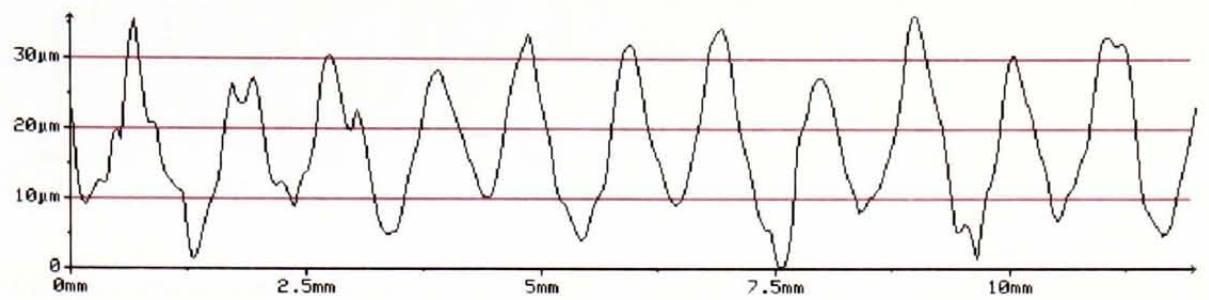
$$\text{error(approximation of probe centre)} \leq 2.\text{error(approximation of contact point)}$$

$$= 0.0588 \qquad (4 \text{ d.p.})$$

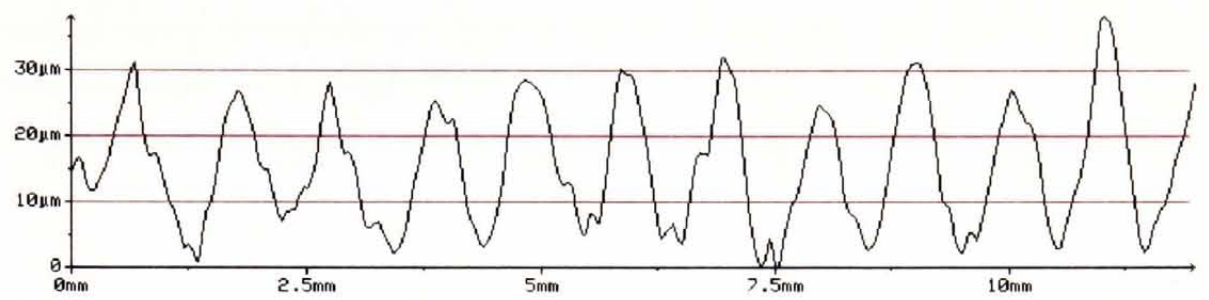
Appendix 2



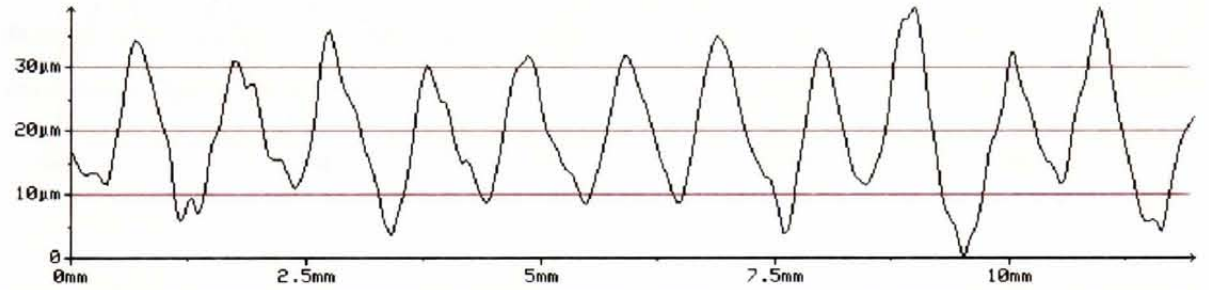
Profile 1



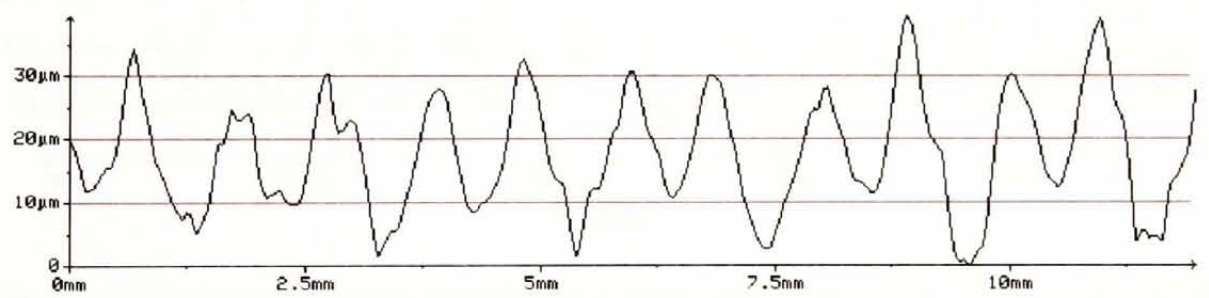
Profile 2



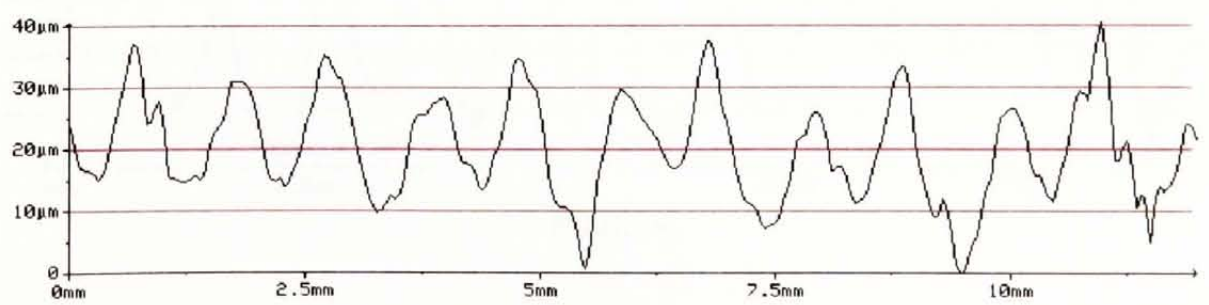
Profile 3



Profile 4

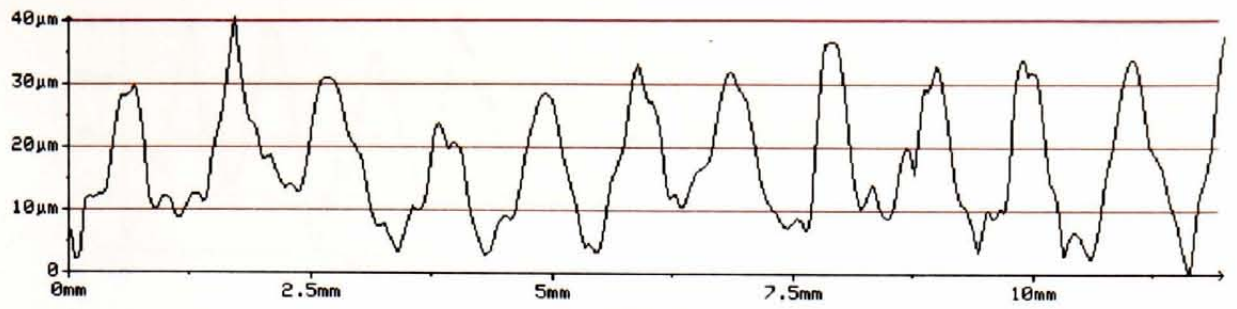


Profile 5

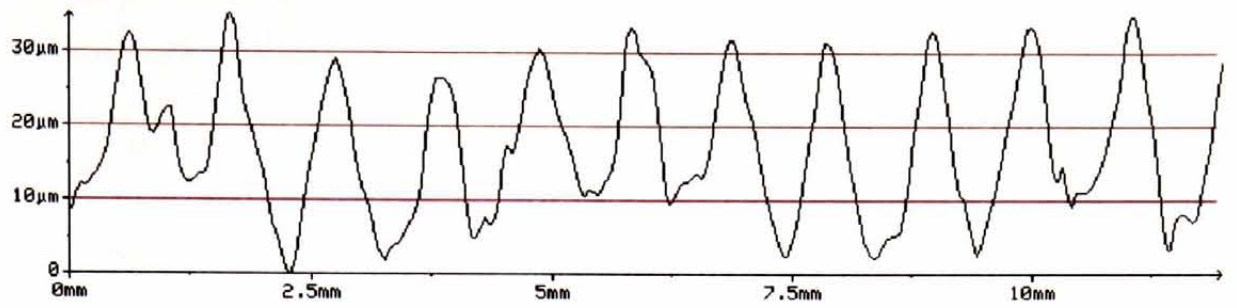


Profile 6

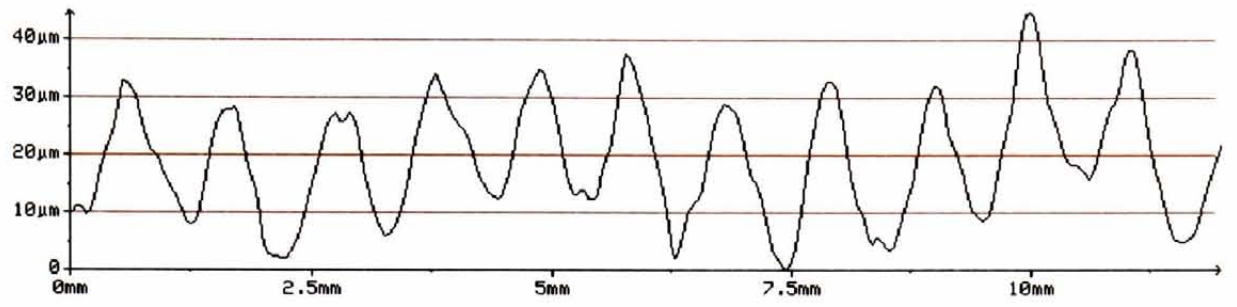
Figure A2.1: IOM – Area 1



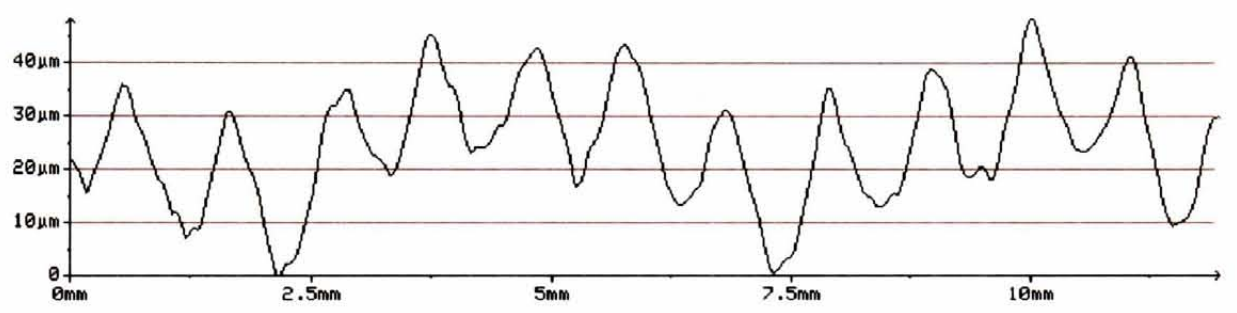
Profile 1



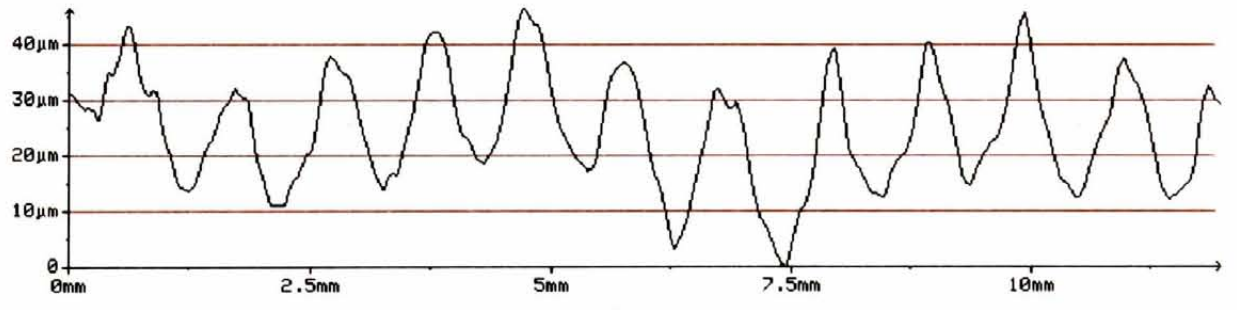
Profile 2



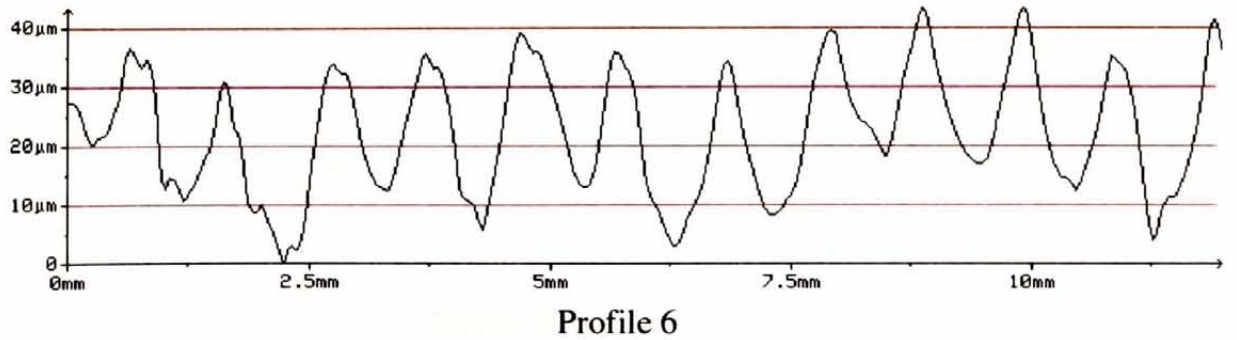
Profile 3



Profile 4

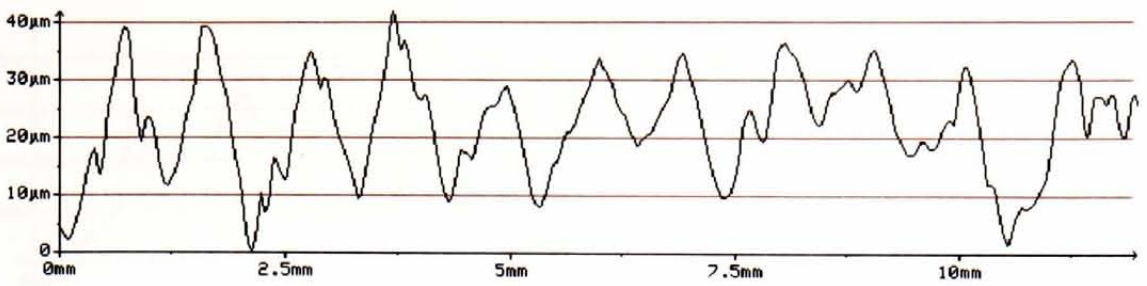


Profile 5

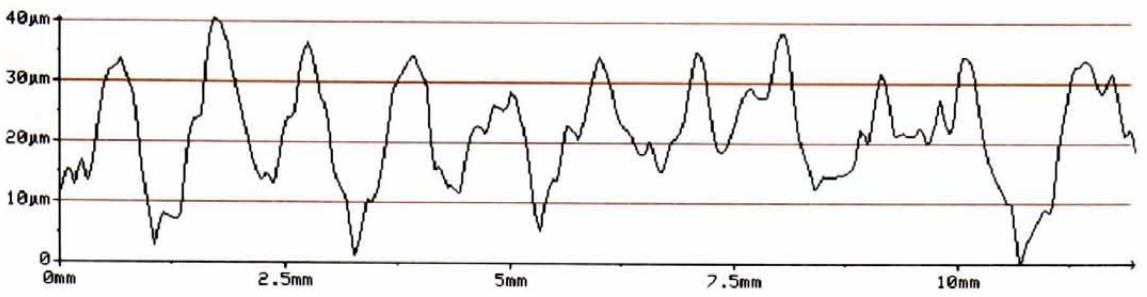


Profile 6

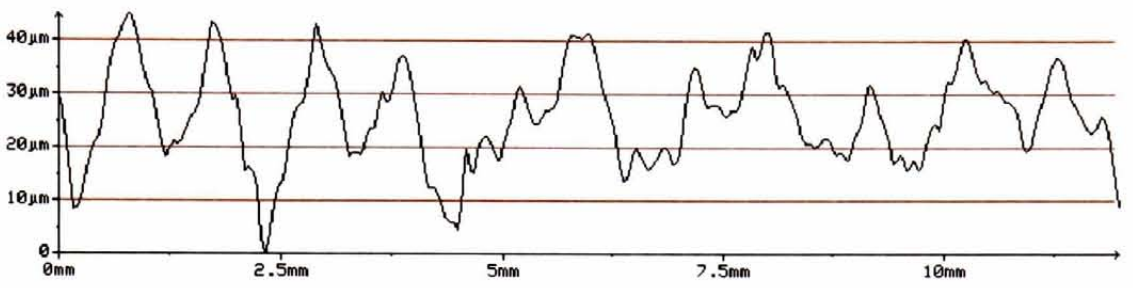
Figure A2.2: Powermill – Area 1



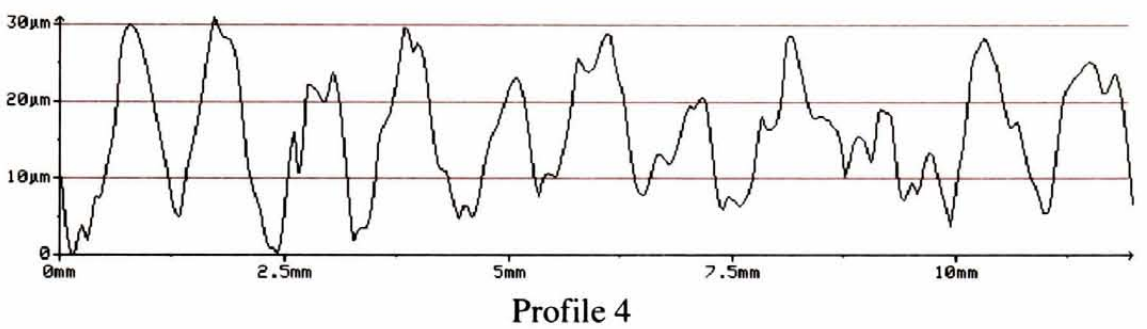
Profile 1



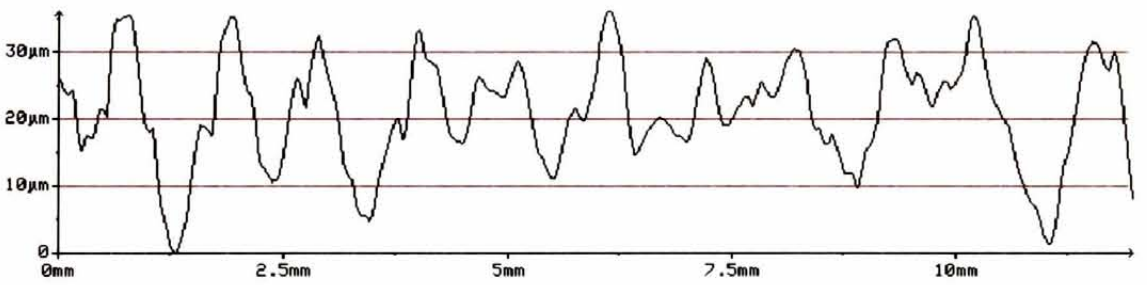
Profile 2



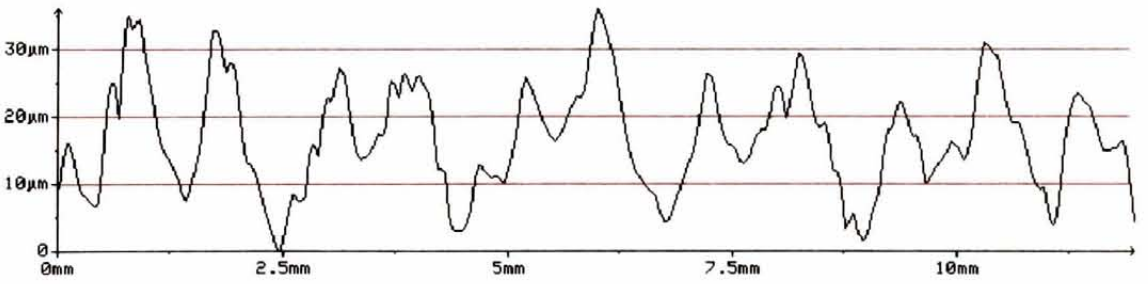
Profile 3



Profile 4

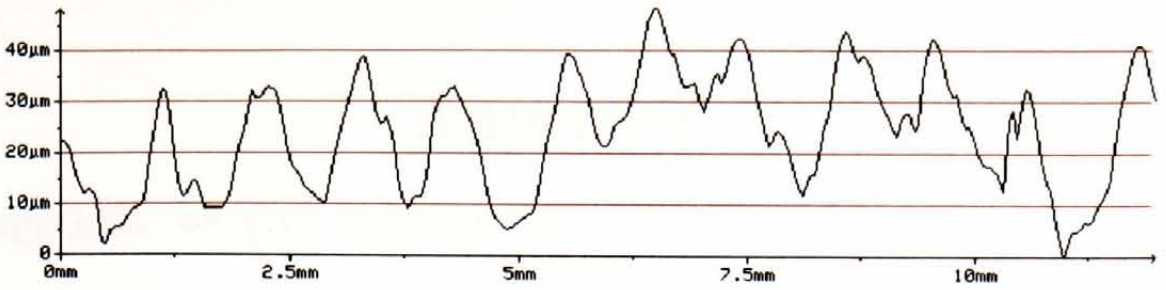


Profile 5

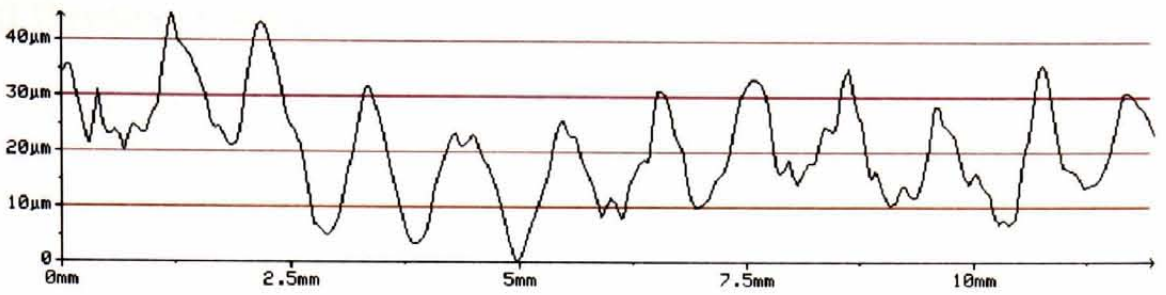


Profile 6

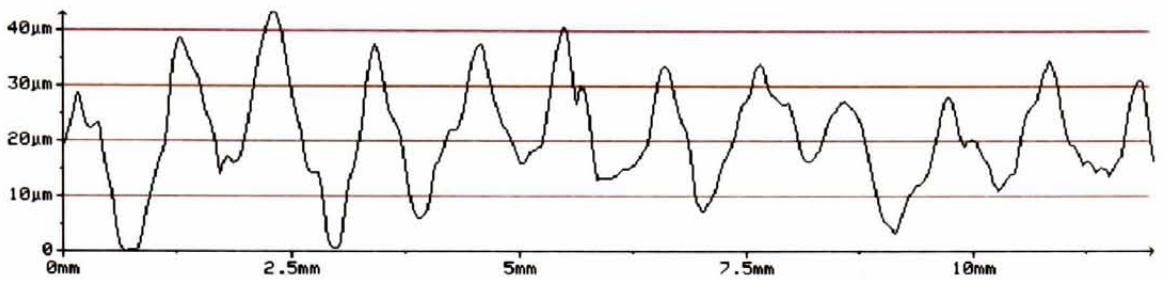
Figure A2.3: IOM – Area 2



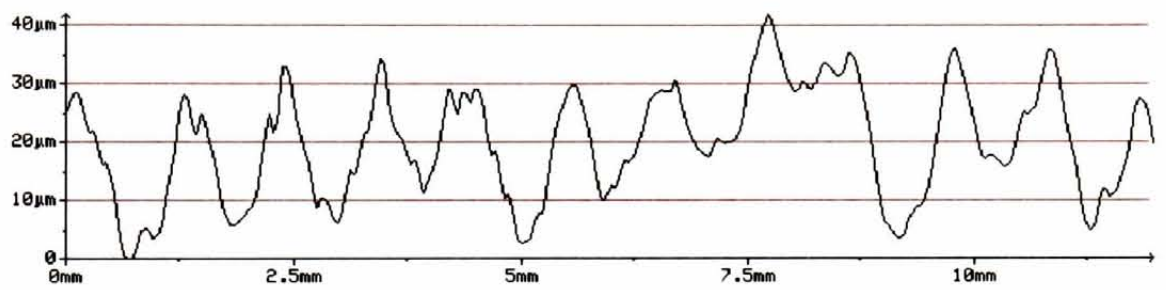
Profile 1



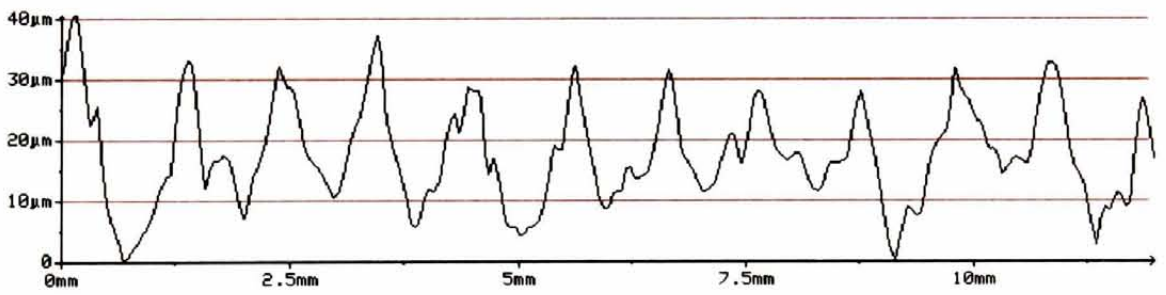
Profile 2



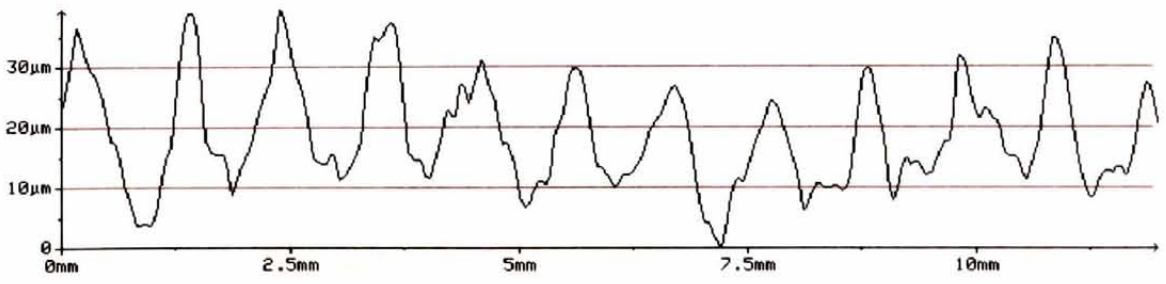
Profile 3



Profile 4



Profile 5



Profile 6

Figure A2.4: Powermill – Area 2

Appendix 3

In this appendix we give implementation details and program operation of the algorithms that were investigated in this thesis. All the algorithms were developed in Microsoft Developer Studio on Windows NT. All programs used the 'math.h' header file also. The IOM was written in C, while all the optimisations were written in C++. In Figure A3.1 a flow chart is given the program structure of the IOM.

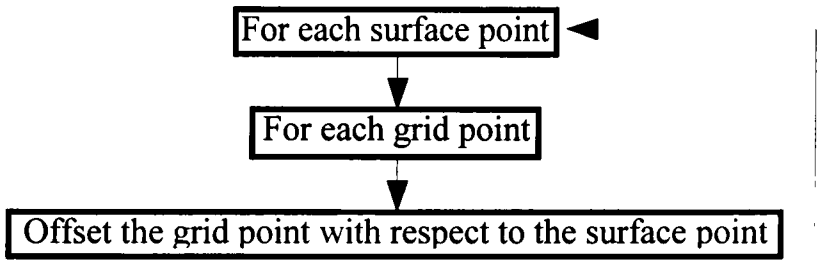


Figure A3.1: IOMs program structure

Figure A3.2 shows the program structure of the S-IOM and PPS-IOM. The only difference between the two is in the way that a section is checked to see if it is active.

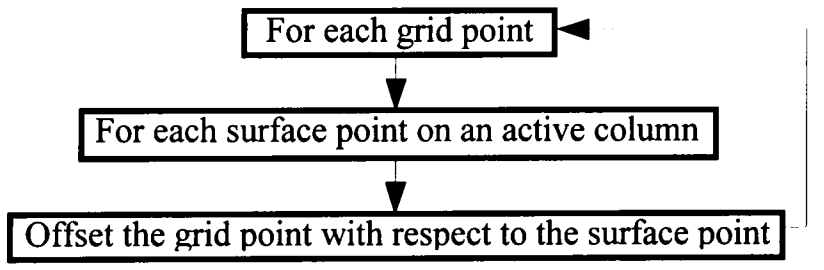


Figure A3.2: The S-IOM and PPS-IOMs program structure

Figure A3.3 shows the program structure of the ES-IOM and EPPS-IOM. As for the S-IOM and the PPS-IOM, the only difference between the two algorithms is the way in which a section is checked to see if it is active.

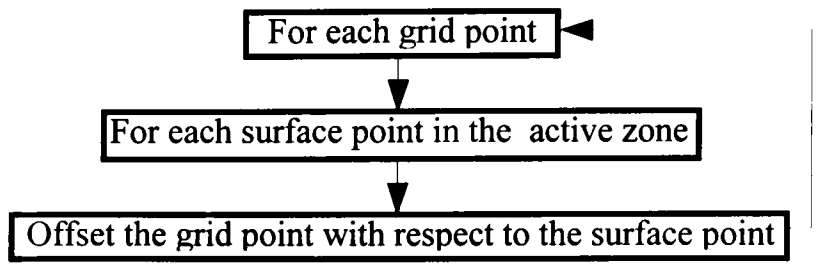


Figure A3.3:The ES-IOM and EPPS-IOMs program structure

References

Ball AA, [1994], Towards a geometric characterisation of shape. *Mathematics of surfaces V*, Oxford Univeristy Press, pp 91-98

Ball AA, [1997], CAD: Master or servant of engineering?. *Mathematics of surfaces VII*, Information Geometers, pp 17-23.

Baxter M, [1995], *Product design,* Chapman &Hall.

Bedworth DD, Henderson MR and Wolfe PM, [1991]. *Computer-integrated design and manufacturing*, McGraw-Hill, Inc.

Chang T-C, Wysk RA and Wang H-P, [1998], *Computer-aided manufacturing - 2nd Ed,* Prentice-Hall Inc.

Chen YD, Ni J and Wu SM, [1993], Real-time CNC tool path generation for machining IGES surfaces. *Transactions of the ASME*, **115**, pp 480-486.

Chen YJ and Ravani B, [1987], Offset surface generation and contouring in computer-aided design. *Journal of mechanisms, transmissions, and automation in design*, **109**, pp 133-142.

Choi BK, Chung YC, Park JW and Kim DH, [1994], Unified CAM-system architecture for die and mould manufacturing. *Computer-aided design*, **26**(3), pp 235-243.

Choi BK, Lee CS, Hwang JS and Jun CS, [1988], Compound surface modelling and machining. *Computer-aided design*, **20**(3), pp 127-136.

Choi BK and Jun CS, [1989], Ball-end cutter interference avoidance in NC machining of sculptured surfaces. *Computer-aided design*, **21**(6), pp 371-378.

Czerkawski AM, [1996], *Fitting procedures for curves and surfaces.* PhD. Thesis, University of Birmingham.

Drysdale RL and Jerard RB, [1987], Discrete simulation of NC machining. *Third annual symposium on computational geometry,* pp 126-135.

Duncan JP and Mair SG, [1983], *Sculptured surfaces in engineering and medicine*, Cambridge University Press.

Dyn N, Levin D and Gregory JA, [1987], A 4-point interpolatory subdivision scheme for curve design. *Computer aided geometric design*, **4**, pp 257-268.

Elber G and Cohen E, [1994], Toolpath generation for freeform surface models. *Computer-aided design*, **26**(6), pp 490-496.

Faux ID and Pratt MJ, [1987], *Computational geometry for design and manufacture,* John Wiley and Sons.

Foley JD and Van Dam A, [1983], *Fundamentals of Interactive Computer Graphics*, Addison Wesley.

Gao X, Harrison DK and Davies BJ, [1989], An approach to the low-cost computer aided manufacture of components embodying free-form surfaces. *Proceedings of the institution of mechanical engineers*, **203**, pp 119-126.

Haapaniemi A, Nagase H, Fukimoto M, Hiraoka H, Kimura F and Sata T, [1986], Development of a real time numerical controller for machining of sculptured surfaces. *Software for discrete manufacturing*, pp 205-214.

Hartquist EE, Menon JP, Suresh K, Voelcker HB and Zagajac J, [1999], A computing strategy for applications involving offsets, sweeps and Minkowski operation. *Computer-aided design*, **31**, pp 175-183.

Hermann G, [1988], Algorithms for real-time tool path generation. *Geometric modeling for CAD applications*, pp 295-305.

Hosaka M and Kimura F, [1980], A theory and methods for three dimensional free form shape construction. *Journal of information processing*, **3**(3), pp 140-151.

Hoschek J and Lasser D, [1993], *Computer aided geometric design,* AK Peters Ltd.

Huang Y and Oliver JH, [1994], Non-constant parameter NC tool path generation on sculptured surfaces. *The international journal of advanced manufacturing technology*, **9**, pp 281-290.

Hwang JS [1992], Interference-free tool-path generation in the NC machining of parametric compound surfaces. *Computer-aided design*, **24**(12), pp 667-676.

Hwang JS and Chang T-C [1998], Three-axis machining of compound surfaces using flat and filleted endmills. *Computer-aided design*, **30**(8), pp 641-647.

Jerard RB, Drysdale RL, Hauck KE, Schaudt B and Magewick J, [1989], Methods for detecting errors in numerically controlled machining of sculptured surfaces. *IEEE computer graphics & applications,* **9**(1), pp 26-39.

Kanda T, [1991], A cutter path determined by a given numerical model in NC metal mold processing. *IECON'91*, pp 1221-1225.

Kim K and Ko B, [1994], Generating cartesian NC tool paths for sculptured surface manufacture. *Computers and industrial engineering*, **26**(2), pp 359-367.

Kim KI and Kim K, [1995], A new machine strategy for sculptured surfaces using offset surface. *International journal for production research*, **33**(6), pp 1683-1697.

Kishinami T, Kondo T and Saito K, [1987], Inverse offset method for cutter path generation. *Proceedings of the 6th international conference on production engineering Osaka,* pp 807-812.

Kobbelt L, [1996], Interpolatory subdivision on open quadrilateral nets with arbitrary topology. *The international journal of the eurographics association*, **15**(3), pp 409-420.

Kuragno T, [1992], FRESDAM system for design of aesthetically pleasing free-form objects and generation of collision-free tool-paths. *Computer-aided design,* **24**(11), pp 573-581.

Lai J-Y and Wang D-J [1994], A strategy for finish cutting path generation of compound surfaces. *Computers in industry*, **25**, pp 189-209.

Lee K, Kim TJ and Hong SE, [1994], Generation of toolpath with selection of proper tools for rough cutting process. *Computer-aided design*, **26**(11), pp 822-831.

Li H, Dong Z and Vickers GW, [1994], Optimal toolpath pattern identification for single island, sculptured part rough machining using fuzzy pattern analysis. *Computer-aided design,* **26**(11), pp 787-795.

Li SX and Jerard RB, [1994], 5-axis machining of sculptured surfaces with a flat-end cutter. *Computer-aided design*, **26**(3), pp 165-178.

Lin AC and Liu H-T, [1998], Automatic generation of NC cutter path from massive data points. *Computer-aided design*, **30**(1), pp 77-89.

Lin AC and Lin S-Y, [1999], Computer-aided mold engraving: from point-data smoothing, NC machining, to accuracy checking. *Journal of materials processing technology*, pp 101-114.

Loney GC and Ozsoy TM, [1987], NC machining of free form surfaces. *Computer-aided design,* **19**(2), pp 85-90.

Marciniak K, [1991], *Geometric modelling for numerically controlled machining.* Oxford university press.

Miguel PAC, [1996], *CMM performance verification, considerations for a virtual artefact approach.* PhD. Thesis, University of Birmingham.

Park JW, Chung YC, Kim BH and Choi BK, [1998], Pencil curve tracing via virtual digitizing. *Machining Impossible Shapes*, Olling GJ, Choi BK and Jerard RB (ed.), pp 279-292.

Piegl LA, [1993], *Fundamental developments of computer-aided geometric modelling,* Academic press.

Saito T and Takahashi T, [1991], NC machining with G-buffer method. *Computer Graphics*, **25**(4), pp 207-216.

Suzuki H, Kuroda Y, Sakamoto M, Haramaki S and Brussel HV, [1991], Development of the CAD/CAM system based on parallel processing and the Inverse Offset Method. *Transputing'91*, pp 184-198.

Takeuchi Y, Sakamoto YA and Orita R, [1989], Development of a personal CAD/CAM system for mold manufacture based on solid modeling techniques. *Annals of the CIRP*, **38**(1), pp 429-432.

Tang K, Cheng CC and Dayan Y, [1995], Offsetting surface boundaries and 3-axis gouge-free surface machining. *Computer-aided design*, **27**(12), pp 915-927.

Todd PH and McLeod RJY, [1986], Numerical estimation of the curvature of surfaces. *Computer-aided design*, **18**(1), pp 33-37.

Tookey RM, [1997], Interpolatory subdivision and bounding box construction for grids of points. Internal report 97/03, GMG, University of Birmingham.

Tookey RM and Ball AA, [1995], A simple CAD/CAM system for small toolmaking companies. *Proceedings of 31st international MATADOR conference,* Kochhar AK (ed.), pp 571-577.

Vadlamudi RS, [1998], Cutting molds/dies from scan data. *Machining Impossible Shapes*, Olling GJ, Choi BK and Jerard RB (ed.), pp 1-7.