

Optimization of Three-Axis Vertical Milling of Sculptured Surfaces

by

Gerardo Salas Bolaños

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Mechanical Engineering

Waterloo, Ontario, Canada, 2010

© Gerardo Salas Bolaños 2010

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

A tool path generation method for sculptured surfaces defined by triangular meshes is presented in this thesis along with an algorithm that helps determine the best type of cutter geometry to machine a specific surface.

Existing tool path planning methods for sculptured surfaces defined by triangular meshes require extensive computer processing power and result in long processing times mainly since surface topology for triangular meshes is not provided. The method presented in this thesis avoids this problem by offsetting each triangular facet individually.

The combination of all the individual offsets make up a cutter location surface. A single triangle offsetting results in many more triangles; many of these are redundant, increasing the time required for data handling in subsequent steps.

To avoid the large number of triangles, the proposed method creates a bounding space to which the offset surface is limited. The original surface mesh describes the bounding surface of a solid, thus it is continuous with no gaps. Therefore, the resulting bounding spaces are also continuous and without gaps. Applying the boundary space limits the size of the offset surface resulting in a reduction in the number of triangular surfaces generated. The offset surface generation may result in unwanted intersecting triangles. The tool path planning strategy addresses this issue by applying hidden-surface removal algorithms. The cutter locations from the offset surface are obtained using the depth buffer. The simulation and machining results show that the tool paths generated by this process are correct. Furthermore, the time required to generate tool paths is less than the time required by other methods.

The second part of this thesis presents a method for selecting an optimal cutter type.

Extensive research has been carried out to determine the best cutter size for a given machining operation. However, cutter type selection has not been studied in-depth. This work presents a method for selecting the best cutter type based on the amount of material removed. By comparing the amount of material removed by two cutters at a given cutter location the best cutter can be selected. The results show that the optimal cutter is highly dependant on the surface geometry. For most complex surfaces it was found that a combination of cutters provides the best results.

Acknowledgements

The following thesis, while an individual work, benefited from the insights and direction of several people. I have worked with a great number of people whose contributions to my research deserve special mention. It is a pleasure to convey my gratitude to them.

I would like to express the deepest appreciation to my supervisor, Dr. Sanjeev Bedi, for his supervision, advice, and guidance from the very early stages of my education and for providing extraordinary experiences throughout the time we worked together. Without his guidance and persistent help this thesis would not have been possible. For all his help throughout the years, I am indebted to him.

I would like to express my gratitude to Dr. Stephen Mann and Dr. Behrad Khamesee for being my thesis readers. In addition, I would like to thank Dr. Stephen Mann for sharing his knowledge of computer graphics with me and Dr. Behrad Khamesee for trusting me as a teaching assistant.

I would like to thank Dr. Armando Roman for helping me when I needed it, for bringing me into the field of CNC machining, for the the advice and guidance he has provided throughout the years, and for his friendship

To the group of students in E3-3134 I would like to express my gratitude. It has been a pleasure working with all of you. Special thanks to Kandarp and Rajnish for the papers we worked on together and to Carlos for helping me relearn C++ and for being my sidekick in the India adventure.

I would like to thank my friends: Mauricio S., Mauricio B., Kevin, Chris B., Chris C., Alex, and Darius for putting up with me all these years. Thank you for all the help and support you have provided throughout the years.

Finally, I would like to thank my family. Mayela and Fabiola, thanks for being supportive and caring sisters. My parents deserve special mention for their constant source of support - emotional, moral, and financial. This thesis would certainly not have existed without them. To my mother, thank you for all the love you sent in form of cakes, cookies and gifts; I always knew I had someone praying for me. To my father, thank you for always pushing me to be the best and teaching me that hard work and perseverance always lead to success. For all their support and encouragement, it is to my parents that I dedicate this work.

To my parents...

Contents

List of Tables	ix
List of Figures	xi
1 Introduction	1
1.1 Surface Representation in STL Format	5
1.2 Proposed Strategy	6
1.3 Research Objectives	7
1.4 Thesis Layout	8
2 Literature Review	10
2.1 Toolpath Generation for Surfaces Defined by Triangular Meshes	11
2.2 Offset Surface	12
2.2.1 General Offset Surface Methods	12
2.2.2 Offset Surface for CNC Machining	14
2.3 Tool Selection	15
2.3.1 Prismatic or 2.5D Parts	15
2.3.2 Complex or 3D parts	16
2.3.3 Tool Selection for 5-Axis CNC Machining	16
2.4 Summary	17

3	Tool Path Generation: Offset Surface	19
3.1	Offset Surfaces in Tool Path Planning	20
3.2	Offset Surface Approach	25
3.3	Offset Surface for a Generalized Cutter	33
3.3.1	Face Offset	33
3.3.2	Edge Offset	36
3.3.3	Vertex Offset	38
3.3.4	Examples of Offset Surfaces	41
3.4	Bounding space	43
3.5	Tool Path Generation	47
3.5.1	Depth Buffer	48
3.5.2	Cutter Locations from Depth Buffer	49
3.6	Results	51
3.6.1	Comparison of Offset Surface Method with Ball Drop Method	57
4	Optimal Tool Selection	59
4.1	Material Removal	60
4.2	Scallop Height	64
4.3	Results	65
4.4	Confirmation Test	73
5	Conclusion	77
5.1	Future Considerations	78
	Appendices	80
	Appendix A STL File Format	80
	Appendix B APT Cutter Geometry	83
	References	89

List of Tables

3.1	Comparison of offset surface methods	32
4.1	Scallop heights of four surfaces	67
4.2	Scallop heights for custom product	75

List of Figures

1.1	Sample wood sign created by WatSign	4
3.1	Tool path foot print and gouge free tool path	20
3.2	Tool path planning method comparison	22
3.3	Offset surfaces for each type of tool	24
3.4	Offset surfaces with voids and gaps	26
3.5	Offset surface for a sharp vertex	27
3.6	Offset surface for a sharp vertex using vertical planes	28
3.7	Offset surface for a sharp vertex using vertical planes	29
3.8	Effect of trimming face offset on tool path	30
3.9	Handling intersecting offset surfaces	31
3.10	Face offset geometry for a generalized cutter	34
3.11	Face offset for ball, flat, and radiused end mill	35
3.12	Edge offset geometry for a generalized cutter	36
3.13	Edge offset for ball, flat, and radiused end mill	38
3.14	Vertex offset geometry for a generalized cutter	39
3.15	Vertex offset for ball, flat, and radiused end mill	40
3.16	Offset surface for square pyramid	42
3.17	Offset surface for a sharp vertex	43
3.18	Comparison of complete and partial edge offset	44
3.19	Vertical planes positive direction	45
3.20	Starting and ending point for offset surface generation	45

3.21	Generation of partial surface	46
3.22	Getting tool path by intersecting planes with offset surface	47
3.23	Depth buffer	49
3.24	Depth buffer values in view volume	50
3.25	Depth buffer pixels	51
3.26	Simulation test of sculptured surface	54
3.27	Simulation and machining result for custom name plaque, Sanjeev Bedi . .	55
3.28	Simulation and machining result for custom name plaque, Stephen Mann .	56
3.29	Tool path generation time comparison	58
4.1	Comparison of area removal for ball nose and radiused cutter	61
4.2	Parameters for calculating area removal	62
4.3	Volume removal of radiused cutter	63
4.4	Orientation of scallop height calculation	65
4.5	Scallop heights and volume removal for a spherical surface	69
4.6	Scallop heights and volume removal for a toroidal surface	70
4.7	Scallop heights and volume removal for a spline surface	71
4.8	Scallop heights and volume removal for a spline surface	72
4.9	Scallop heights and volume removal for a custom plaque	74
1	Conversion of solid model to tessellated model	80
2	Orientation of triangular facet	81
3	Vertex-to-vertex rule	82
4	Parameters for generalized APT cutter geometry	84
5	Selection of cutter shapes based on APT definition	85

Chapter 1

Introduction

The need to increase productivity and survivability in an increasingly competitive manufacturing market requires the integration between computer-aided design (CAD) and computer-aided manufacturing (CAM); this can be achieved by computer-aided process planning (CAPP). CAPP is the collection of activities that translate a part's design specifications from engineering drawings into the manufacturing instructions required to produce it.

Manual process planning is time-consuming and the results vary based on the person doing the planning. By using computer systems the process planning is simplified, optimum process plans are produced quickly and consistently, and more efficient use of manufacturing resources is achieved [51]. CAPP successfully links engineering design and shop floor manufacturing. Still, the intricacy and interdependent nature of manufacturing processes make the effective implementation of CAPP in industry difficult, particularly those industries involved with cutting operations [18].

Process planning is based on a manufacturing engineer's experience and knowledge of production facilities, equipment, capabilities, processes and tooling [9]. Transferring this

knowledge and experience into a computerized system is not an easy task especially since some of this knowledge is empirical. The goal of a CAPP in industry is to reach a generative stage where the system can produce a complete process plan from part classification and other design data. This involves the use of artificial intelligence type capabilities to produce process plans as well as be fully integrated in a computer integrated manufacturing environment. These reasons make successful implementations of CAPP in industry challenging.

Even more challenging is the task of creating an automated CAPP system because CAD and CAM data are heterogeneous and incompatible. There are many problems to face in terms of compatibility of different computer systems and hardware, the compatibility of computer languages, and the compatibility of software packages. This makes a single universally applicable process plan for all parts in manufacturing unlikely to be attained, or even attempted.

The difficulty of implementation is not the only challenge associated with a successful implementation of CAPP system- there is also a high cost involved due to CAD and CAM. The costs are in the form of CAD/CAM software, operator salary, and regular training required to keep up to date with changes in software and manufacturing processes. In large manufacturing companies the costs associated with CAD and CAM are justified since these are distributed among a large number of products. In industries where an item is manufactured over and over the costs of CAD and CAM are also reasonable. However, in small manufacturing industries or in the manufacturing of customizable products the costs associated with CAD/CAM are not justified.

Use of automation in CAD and CAM systems in custom product markets has been limited due to the high-costs associated with these [23]. In custom products such as bio-

medical inserts, the cost of CAD and CAM results in high priced products. In these cases; however, the price is justified by the benefit. The cost of designing each part individually can be a prohibitive factor for products like custom address plaques and custom table legs. The cost of CAD, CAM, and CNC machining can be reduced by coupling these three elements in an automated system. The implementation of universal CAPP system is difficult, if not impossible; but a system targeted for a specific type of product is a viable solution. A combined CAD-CAM-CNC system solves the issue by offering functionality of designing and manufacturing in a single package.

Systems such as these have already been created. Such is the case of *WatCAD/CAM*: a web-based CAD and CAM system that allows users to easily design and manufacture table legs. The software is built around a solid modelling engine and a custom CAM package that generates tool paths that are used in the CNC milling lathe designed specifically to carve wooden legs [38]. The system is created in such a manner that the CAD software allows designers to create parts containing features only when they can be manufactured afterwards. By setting into place these design constraints, the CAM package can create tool paths automatically for any design created in the CAD packages. In the same manner, the CAM package will generate the machine-appropriate instructions for the CNC milling lathe to manufacture the table leg. Other examples of such systems are *CyberCut* [1, 41], used in the design and manufacture of simple 2.5D parts; DELCAM's *ArtCAM* [2], which provides a combined CAD/CAM package for designing customized wooden and metal products; and *WatSign* [23], a web-based custom wooden plaque consisting of a web-based solid modeller paired with a CAM software which allows users to easily design plaques, like the one shown in Figure 1.1, and download the tool paths required to machine the plaque in a 3-axis CNC milling machine. It is in products like custom address plaques that the research described



Figure 1.1: Sample wood sign created by WatSign [23]

in this thesis can be applied.

The 3-axis CNC machining of sculptured surfaces, such as the ones generated by *WatSign*, is not an easy task: the generation of tool paths is a complicated task and may not always have optimal results. The main goal in sculptured surface machining is to have a side step length that is as large as possible. Having a large side step reduces the machining time as well as the amount of data generated [6]. However, the side step cannot be too large because the resulting scallop heights formed by two adjacent machined paths may not provide sufficient surface quality. Predicting scallop heights for a sculptured surface is a complicated process. Another issue that must be addressed in the tool path planning of sculptured surfaces is local tool avoidance (gouging). Two approaches are commonly used to avoid gouging. The first is to use a cutter that has a radius smaller than the smallest radius of curvature on the surface; however, having a small radius results in longer machining times since it increases as the cutter radius decreases. The second approach is to use a large cutter and skip the areas where the tool would gouge the surface. This can leave a lot of uncut material and detection and correction of the local cutter interference represents an additional problem.

For these reasons, a tool path generation method for sculptured surfaces defined by triangular meshes is presented in this thesis along with an algorithm that determines what type of cutter geometry is best suited to machine a specific surface. The software that is described represents the CAM component of CAPP system. It is designed in such a manner that it can be paired with any CAD system and CNC machine to automatically generate tool paths for a family of sculptured surfaces that will be machined in a 3-axis CNC mill.

1.1 Surface Representation in STL Format

The use of triangulated surfaces stored in stereolithography (STL) files for design and manufacturing applications started in the field of rapid prototyping, selective laser sintering, laminating object manufacturing, and three-dimensional printing. Given their simplicity STL files are now commonly used in other fields of engineering such as CNC machining. Moreover, STL files are used due to the necessity of neutral data files required for transferring models between different CAD/CAM systems. Other neutral data file formats such as the Initial Graphics Exchange Specification (IGES) and Standard for Exchange of Product Data (STEP) are also used for communicating product data among dissimilar CAD/CAM systems. However, the translation of CAD models using IGES and STEP is not easy because most CAD systems use different internal representations and the conversion is not always error free [49]. In contrast to IGES and STEP, the STL format is simple and its implementation is easy. Even though the STL contains less geometric information than an other file formats, it is enough information for CNC machining.

In the past, STL files were not used given the large memory allocation required and

the long processing time associated with large STL files. However, since the cost of central processing units (CPU) and memory chips continues to decrease while their power increases this is no longer a problem. Most 3D modelling CAD software are now equipped with efficient tessellation algorithms that can create valid triangular meshes from any valid solid. These advantages along with the advantage that STL files are neutral data files make them the best option for transferring data effortlessly among various CAD and CAM packages.

1.2 Proposed Strategy

A tool path planning method for sculptured surfaces is presented in this work. This method uses the offset surface of a solid model represented by a triangular mesh to generate tool paths. This representation was chosen as it can be stored in a neutral data file easily transferable between CAD and CAM packages: STL files. Triangular meshes are commonly used as representations of sculptured surfaces; however, two disadvantages make it difficult to generate tool paths for these meshed surfaces:

1. Topological information is not provided for the triangular facets.
2. An accurate representation requires a large number of triangular facets.

Previous research in NC machining of sculptured surfaces represented by triangular meshes has shown that tool path planning results in long processing times since the time required to process the entire triangular mesh as a whole is large [33, 49, 34]. Therefore, this thesis presents a new method in which each triangle is handled independently of the other triangles.

When offsetting a single triangular facet its area of influence is bound by its three edges. If three vertical planes were to be drawn, one at each edge of the triangle, the area inside the planes represents the area of influence of that given triangle; thus, during the offsetting procedure care is taken that this area is completely covered by the offset surface. Triangles outside the region are ignored. As the original triangular model represented a connected volume, the offset surface will also be a connected volume. This thesis presents how this method results in a more efficient gouge free tool path generation method without any sacrifices in accuracy.

The second part of this thesis presents a method for selecting an optimal cutter type. Extensive research has been carried out to determine the best cutter size for a given machining operation. However, cutter type selection has not been studied in-depth. This work presents a method for selecting the best cutter type based on the amount of material removed. By comparing the amount of material removed by two cutters at a given cutter location (CL) the best cutter can be selected: the cutter that removes the most material is best since the machining resulting from this cutter will be closest to the solid model. This material removal comparison is only possible since the offset surface tool path planning method guarantees that the tool path is gouge free; therefore, a cutter can be selected with maximum material removal with out gouging.

1.3 Research Objectives

The utilization of CNC machines to manufacture complex surfaces has driven extensive research work in the area of tool path generation. Two criteria are generally used to evaluate the generated tool paths. The first deals with the validity of the tool paths and

the second deals with their optimality. Both these criteria are address in this research.

There are two main objectives in this thesis. The first, is to develop a 3-axis milling tool path planning strategy for a model defined by a triangular mesh. The tool path is to be generated using the offset surface technique. The offset surface scheme should work for a generalized cutter and topology of the triangular mesh should not be a requirement. The goal is to produce valid tool paths while decreasing the processing time required to generate the tool path without any sacrifices in terms of accuracy.

The second objective is to develop a methodology for determining which cutting tool geometry is best suited to machine a sculptured surface. Special emphasis is to be given on cutter type rather than cutter size. Extensive research has been performed in determining the optimal cutter size, but there is little research in determining the cutter type. The goal is to develop a method for determining numerically which tool type gives the best results; i.e., results in the machined part that is closest to solid model. This portion of the research addresses the optimality of the tool paths by ensuring high quality and efficient machining.

1.4 Thesis Layout

First, a literature review covering offset surface methods in CNC machining and tool selection methods is presented in Chapter 2. The shortcomings of the published research described in this chapter will point out the necessity of the research described in this thesis.

Chapter 3 provides an in-depth description of the use of offset surfaces in CNC machining covering the theory as well as its benefits and disadvantages. The *vertical planes*

approach taken to offset a triangular mesh is described explaining the geometry of an offset surface for a generalized cutter. Finally, the generation of tool paths based on an offset surface using the graphics *z-buffer* is discussed along with results.

The need for an optimal tool selection methodology is presented in Chapter 4 along with the formula that was developed to determine which cutting tool geometry is best suited for a specific surface. Results of test performed on various surface types and actual consumer products are presented.

In Chapter 5, conclusions and recommendations for future improvements are presented.

Chapter 2

Literature Review

The improvement in tessellation algorithms and the increase in CPU processing power has lead to an increasing number of complex surfaces in industrial applications that are defined using triangular meshes. To address this issue, algorithms capable of generating tool paths to machine these surfaces have been developed. This chapter will familiarize the reader with existing tool path planning methodologies along with their limitations. This will show the need for a new and more efficient tool path planning technique.

This chapter will also present previous research in the area of optimal tool selection for NC machining. To have an efficient machining process it is essential to select the appropriate cutter. This chapter will show how the research that has been carried out in tool selection has mainly focused on determining tool size and not tool type.

2.1 Toolpath Generation for Surfaces Defined by Triangular Meshes

Manos *et al.* [33] developed a gouge-free method for machining a surface defined by a triangular mesh using a ball nose cutter. The method, known as “ball drop”, consists in “dropping” a sphere at every location on the surface. The first point at which the sphere touches the surface is defined as the cutter location point. The advantage of this method is that the resulting tool path is gouge free. However, the processing time is large since each ball drop requires a series of checks that must be carried out to determine the first point at which the sphere touches the surface. These checks involve large processing time since it involves solving high-order polynomial equations. Furthermore, for a given point on the surface the triangles that must be analysed are not known since the topological information is not given by the file defining the triangular mesh (STL-file) and every triangle must be taken into account. To minimize the processing time this method is optimized using a bucketing algorithm [37] which creates partial topological information of the triangular mesh. The interconnectedness of the triangles is not known, but for a given point on the surface the “bucket” of triangles that are nearby is known. This reduces the processing time but the checks required are still the same.

Yau *et al.* [49] developed a similar algorithm as the one by Manos *et al.* In this case rather than just using a ball nose cutter, they have developed an algorithm using generalized automatically programmed tools (APT) cutter geometry. This generalized cutter geometry encompasses ball nose, flat, and radius end mills. This method also uses a bucketing algorithm to reduce processing time. However, the numbers of checks to find the cutter location are many and the processing time is large.

Park[34] proposed computing a tool path by slicing a CL-surface. The method involves two steps: obtaining a set of line segments by slicing the triangular mesh with two-dimensional geometric elements and extracting a valid tool path from the line segments by removing invalid proportions. The author claims that this method is more efficient than other methods yet no comparison is provided. Furthermore, this method does not allow Z-level machining, clean-up machining, and pencil machining.

2.2 Offset Surface

2.2.1 General Offset Surface Methods

Kim *et al.* [25] describe a method in which the triangular mesh is offset by moving the vertex along the multiple normal vectors of a vertex computed by the normal vectors of the faces surrounding the vertex. The multiple normal vectors of a vertex are set the same as the normal vectors of the faces surrounding the vertex. The offset surface generated by this method does not present gaps or overlapping triangles at the smooth edges. It deals with sharp edges by moving the vertices to the normal directions of the faces and joining them by a blend surface. This method has shown to decrease computational time in creating the offset surface; however, it has several disadvantages. First, it requires a complete solid to generate the multiple normals of a vertex. Second, the resultant offset distance is not always the desired one. Finally, in order to be able to calculate multiple normals of the vertex the topology of the triangular mesh must be known and thus this information must be generated.

Qu and Stucker [36] developed a method for offsetting a triangular mesh by offsetting

the vertices of the triangle rather than the facets. The magnitude and direction of each vertex is offset is calculated using the weighted sum of the normals of the facets connected to the vertex. The main advantage of this method is that the offset surface resulting from this method does not present self-intersections and gaps; however, it has many disadvantages. In many cases the length of the offset vector calculated using the weighted sum is much larger than the offset distance; while this might be acceptable for rapid prototyping it is not for CNC machining. This method only works for small offset values; otherwise, self-intersection becomes an issue and post-processing is required. Finally, the connectivity of the triangular mesh must be known to calculate the sum of normals; this results in extra processing time to generate the topological information.

Yi *et al.* [50] calculate the magnitude and direction of each vertex using a modified version of the quadric error metric. This minimizes the sum of the squared distance error from the faces around the original vector; nonetheless, it still presents the same problems as the Qu and Stucker's method.

Koc and Lee [27] used a method of non-uniform offsetting, biarcs fitting, and averaged surface normals to find the correct offset surface. This method generates a gap and self-intersection free surface; however, the offset distance is not uniform.

The research so far described is not exclusive to machining. Many of these techniques are used for rapid prototyping. That is why in many cases the offset distance not being uniform is accepted. Many of these processes can be optimized by techniques described in [20, 44, 39] such as decimation of triangle meshes, re-tiling polygonal surfaces, and self-intersection removal in triangular mesh offset. Decimation and re-tiling of polygonal surfaces reduces the number of triangles used to define the sculptures surface. While this will reduce the processing time the accuracy of the definition of the surface is compromised.

Self-intersection removal is a time consuming task.

2.2.2 Offset Surface for CNC Machining

Kim and Yang [26] developed a method to create a triangular mesh offset for a generalized cutter based on the APT definition. The offset surface is generated based on the type of cutter that will be used for the machining process. This method is successful at offsetting the triangular facets based on the normal vector of the facet and the type of tool being used. However, to deal with the gaps created by the facet offset the multiple normal vectors of vertices is used. The multiple normal vectors of vertices is not uniform throughout the offset surface and this method only works for smooth edges and vertices. It addresses this problem by recursively dividing two vertex normals in the case of an edge and three or more vertex normals in the case of a vertex. The result is a blend surface that is within an acceptable error range but not exact.

Finally, Jun et al. [19] proposed a curve-based approach to gauge free tool paths. The triangular mesh is offset by a local offsetting scheme. The resulting offset elements (triangular facets, trimmed cylinders, and trimmed spheres) are sliced by a series of drive planes. This method results in a gouge-free tool path in both convex and concave regions. The two disadvantages are that STL-file topology is required and trimming and linking tool path curve is required.

2.3 Tool Selection

Research on tool selection for three axes NC milling operations is limited and mostly focused on pocketing operations using flat end mills.

2.3.1 Prismatic or 2.5D Parts

Bala and Changs [3] method selects finishing and roughing tools based on minimum cutter motion criterion and uses algorithms to determine the area in which the cutter centre can move. This method is limited to features found in prismatic parts such as slots, steps, and projections. A similar method is described by Yao *et al.* [48, 47] for tool selection in machining of 2.5D parts. The optimal tool selection is found by analyzing the area that can be cut by a given cutter. The sequence of cutters is selected using Dijkstras shortest path planning algorithm.

Kyoung *et al.* [46] developed another method for tool selection for pocket machining. This research considers tool size the most important factor in an optimal process. Therefore, it focuses in selecting the optimal tool sizes for pocket machining using the branch and bound method.

Lim *et al.* [32] propose a method for optimizing tool selection by considering residual material that is inaccessible to oversized cutters and the relative clearance rates of cutters that can access these regions of the selected machining features. The method was only tested in 2.5D parts and mostly for pocketing operations.

The methods described are focused on the selection of tool radius for flat end milling operations. No consideration is given to the actual tool shape.

2.3.2 Complex or 3D parts

Tool selection for the machining of complex surfaces has also been studied. Glaeser *et al.* [12] developed an algorithm based on an evaluation of the surface curvature that yields a differential inequality for determining the meridian curve of a cutting tool. This inequality is only fulfilled if the cutting tool is able to machine the entire surface. Solving the inequality yields the optimal cutter in the sense that it can machine the entire surface but has the largest possible curvature radii. This method only works for sculptured surfaces and the optimal shape of the cutter may not be one of a commercial cutter.

Chen *et al.* [7] describe two optimization methods for minimizing machining time. The integer programming method generates an upper bound for the problem of cutter selection and the dynamic programming method selects the optimal cutter and machine plane. Both the methods are limited to pocketing of complex surfaces and only using flat end mills.

Lee *et al.* [31] determine cutter size for sculptured surface cavity machining by considering geometric constraints (determined using hunt planes), maximum material removal rate in the roughing process, and minimum cutter movement with the required accuracy in the finishing process. The machining strategy uses large flat and ball end mills for roughing and small ball nose end mills for finishing. The reason as to why a ball nose is used for finishing is not provided.

2.3.3 Tool Selection for 5-Axis CNC Machining

Vickers and Quan [45] carried out an extensive analysis comparing ball nose and flat end mill cutters for the machining of low curvature surfaces. The analysis took into account

cutter geometry (effective radius), surface roughness, number of passes, and cutting speed. The result was that flat end mills are a faster means of machining a wide class of low curvature surfaces.

Lee and Chang [30] presented a methodology for finding the optimal cutter size for 5-axis sculptured surface machining. The appropriate cutter size is determined based on the effective cutter radius and the range of feasible cutter radius. The effective cutter radius is determined as a function of physical cutter size and tool orientation.

Bedi *et al.* [4] compared the effects of using radiused end mills with ball nose and flat bottom end mills. Numerical and experimental studies showed that radiused cutters inherit the advantages of both ball and flat end mills. Radiused cutters lead to smaller scallops compared to ball nose end mills, and they generate surface roughness along the feed direction that is superior to that produced by flat end mills.

Jensen *et al.* [17] introduced an automatic tool selection method for radiused cutters based on cutter radius, cutter corner radius and cutter effective length. The selection of cutter is based on curvature matching, while the cutter effective length is computed by using global tool interference detection.

2.4 Summary

Tool path planning algorithms for tessellated surfaces have been presented. Methods like the *ball drop* algorithm are successful at accurately generating tool paths; however, processing time is long. Methods that use surface offsetting for tool path planning are more efficient at creating tool paths but struggle since they require surface topology. In an at-

tempt to optimize tool path generation time, these methodologies make sacrifices in the accuracy of the offset surface resulting in an inaccurate machining process.

The tool selection methods presented have mostly focused in determining the largest possible cutter radius in order to minimize machining time. The research in optimal tool selection for three-axis machining has not studied the effect of surface curvature or topology in tool selection. Surface curvature has been analysed in five-axis machining resulting in more efficient machining and better surface finish.

Chapter 3

Tool Path Generation: Offset Surface

In the literature review provided in this thesis, several tool path generation methods for triangular meshed surfaces are presented. Manos *et al.* [33] and Yau *et al.* [49] developed tool positioning strategies by performing checks to find the first point at which a tool touches the surface. These have the disadvantage of having a long computational time because of the time required to perform the checks for each CL [35]. This issue can be addressed using bucketing algorithms to optimize the process; however, the tool path planning process is still long. A sample model containing 82,345 triangular facets requires 651 seconds to generate a tool path. A common tool positioning strategy used to avoid the long computation times is the offset surface method. This method is widely used since an offset surface represents the cutter center locations for all points on the part and can provide a gouge free tool path.

3.1 Offset Surfaces in Tool Path Planning

A part is machined by moving a tool across the surface. The trajectory of the tool is dictated by a path designed on a plane perpendicular to the tool axis. This trajectory is called the tool path foot print. The tool path foot print for machining the surface shown in Figure 3.1 is a zig-zag path. The path is characterized by its side step, i.e., the distance between adjacent straight line paths. To create a tool path, the foot print is discretized into small moves connecting gouge free CL-points. The tool is moved linearly between the the CL-points to machine the part. If the foot print is discretized finely the surface can be machined precisely. Tool path generation algorithms are used to find gouge free CL-points on the tool path foot print.

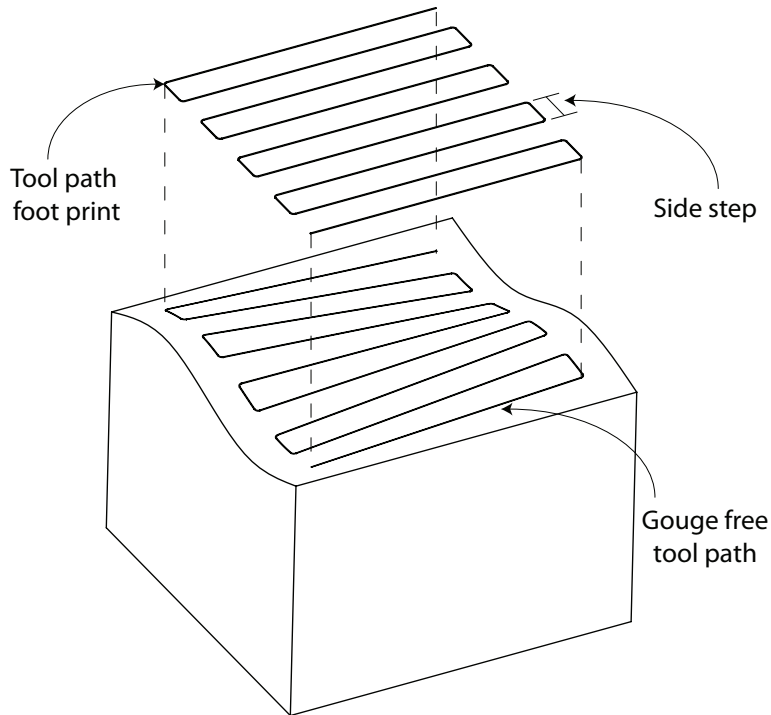


Figure 3.1: Tool path foot print and gouge free tool path

Based on tool interference removal, tool path generation procedures can be classified into two groups: cutter contact (CC) point methods and cutter locations (CL) point methods.

- **CC-point Method** - In the CC-based method, the CC-points are found based on a discretized tool path foot print. The CL-points are then obtained by offsetting the CC points along the surface normal unit vector, as shown in Figure 3.2a. This method has the advantage of accurately reflecting the design surface in the CC-points. However, it has the disadvantage of having a long computation time since the process of going from CC-points to CL-points is a three-dimensional problem requiring numerical methods to solve the geometric equations for the problem. An example of a CC-point based method is the ball drop method by Manos *et al.* [33].
- **CL-point Method** - In the CL-based method, an intersection-free surface, i.e., offset surface, is first constructed based on the design surface. The CL-points are then found directly from the offset surface, 3.2b. This method has the advantage that tool interferences are removed when the offset surface is generated.

The offset surface can be either a free-form surface (nonuniform rational B-spline, NURBS), or a mesh surface (triangular mesh). Most CAD systems use NURBS surfaces to design 3D models of sculptured surfaces. Tool path generation systems that offset NURBS surfaces have previously been created [13, 21]. These methods have the advantage of using accurate surface information; however, implementation of surface offset is not simple because it is difficult to represent an exact offset of a NURBS [25] and surface intersection detection is difficult. To overcome these difficulties the surfaces are defined as triangular meshes. The triangular mesh based NC tool path generation approach is numerically more

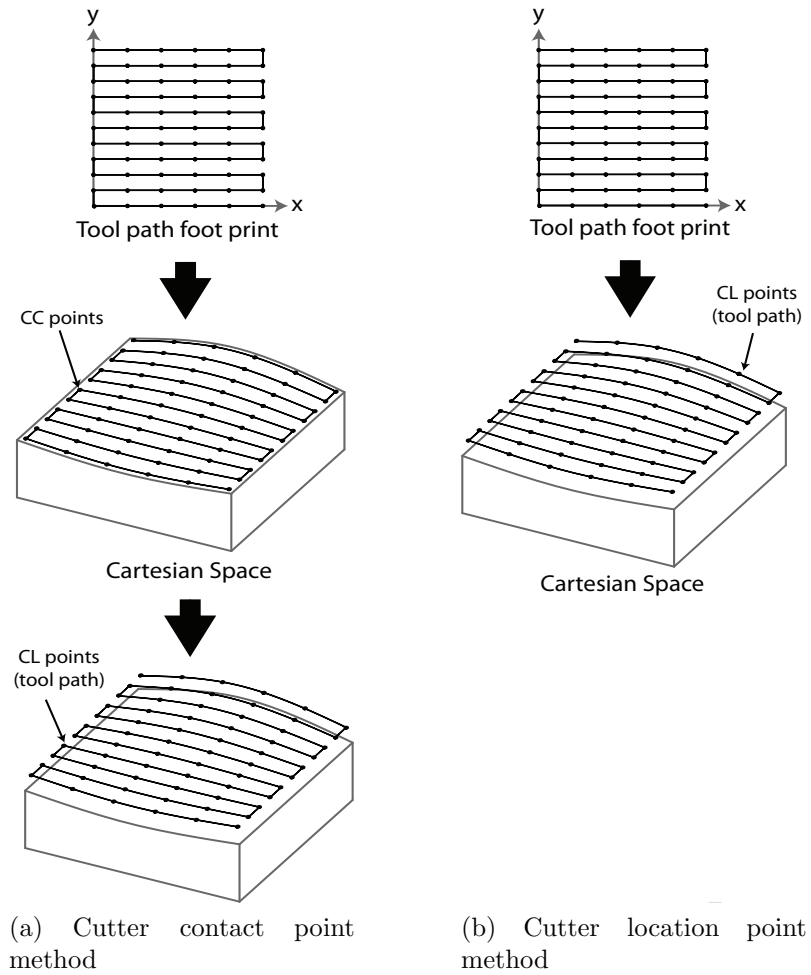


Figure 3.2: Tool path planning method comparison [24]

stable than surface-based tool path generation in terms of gouge check, pencil cut, and remaining cut process. If the triangles are small enough any degree of accuracy can be achieved.

There are two main methods used for offsetting a triangular mesh: vertex offset and face offset.

- **Vertex offset** - This method offsets a triangular mesh by moving the vertex of each

triangle to the average surface normal direction. It has the advantage that it does not create gaps and local gouges at small offset distances. However, it is not accurate enough for NC machining since the offset distance is not constant throughout the surface which results in an inaccurate machining process. Vertex offset is used in rapid prototyping processes and finite element method analysis.

- **Face offset** - The offset surface is generated by moving all faces along the normal direction of the faces and filling gaps at convex edges and vertices using cylindrical and spherical surfaces. This process results in an accurate offset surface that can be used for NC machining; however, previous research [26, 19] have difficulty in the following areas:

1. *Offset of sharp edges and vertices*: the big difference in surface normals at sharp edges and vertices makes offsetting of these a difficult task. The result of offsetting sharp edges and vertices results in voids (at convex surfaces) or intersecting triangles (at concave surfaces). This issue has been addressed by generating blend surfaces on sharp edges and vertices [26]; however, the blend surface is an approximation and thus not always accurate.
2. *Surface topology*: many face offset methods rely heavily on the surface topology of the triangular mesh, e.g., it is required to generate the blend surfaces. As mentioned previously, the surface topology is not provided for the triangular meshes. Thus, the topology must first be generated and then offset the surface. This translates into a lengthy offset surface generation process. A preliminary implementation resulted in topology generation time of 94 seconds for a triangular mesh containing 4200 triangular facets. The time required to generate the topology increases rapidly as the number of triangle increases.

3. *Tool type*: Most research has focused on creating offset surfaces for ball nose cutters since the offset distance for this cutter is only dependant on tool radius and the distance is uniform throughout the entire surface [25, 36, 50]. For an effective machining of a designed model, many cutters are used from roughing to finish cutting. Therefore, offset surface generation for different cutter geometries are required for effective 3-axis tool path generation. The offset surface distance for other cutters, such as radiused and flat, is dependant on both the surface normal and the tool geometry as shown in Figure 3.3.

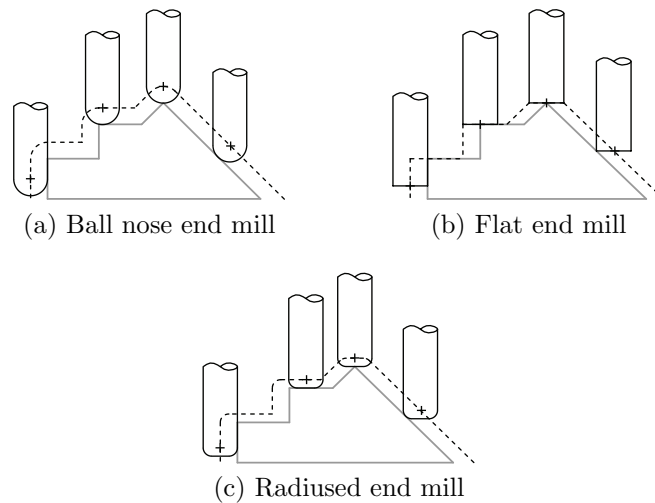


Figure 3.3: The offset surface (dashed line) of a part (solid gray line) is created for ball nose, flat, and radiused end mills

This chapter presents a new geometric solution for the generation of an offset surface that addresses the issues that have been presented. The offset surface that is generated is specifically for 3-axes NC machining. This method takes advantage of the fact that in 3-axes machining the tool axis does not change. The generated offset surface is void free and the offset distance is not approximated at any point on the offset surface. The offset

surface generation does not require surface topology and is optimized by limiting the area of influence of each individual triangular facet.

3.2 Offset Surface Approach

The successful offsetting of a triangulated surface is not an easy task. When two neighbouring facets in a concave section are offset, facet intersections occur. Similarly, voids are created when two neighbouring triangles in a convex segment are offset as shown in Figure 3.4a. The voids and intersections occur at the edges and vertices. The voids are usually handled by filling them with cylinders that model the offset of the edges and spheres that model the offset of the vertices, as shown in Figure 3.4b. To eliminate intersecting triangles surface topology is required to determine that the facets are intersecting. The line of intersection must then be found to trim the facets and eliminate the unwanted areas of the facets. The processes of generating surfaces at edges and vertices and trimming triangles are time consuming and computationally intensive for the computer since an excess amount of triangles results from creating cylinders and spheres and trimming of triangles requires surface topology.

The method described in this chapter focuses on one triangle at a time; sorting or preprocessing of triangles is not required. The offset of each triangle is generated by separately offsetting triangles in three steps: face offset, edge offset, and vertex offset. The first step is simply offsetting the triangular facet along the surface normal. The second step, edge offset, fills the gaps created by the face offset at the edges using cylindrical surfaces. Finally, the third step, fills the gaps created by the face offset at the vertices using spherical surfaces. Treating each triangle individually has two advantages. First, the

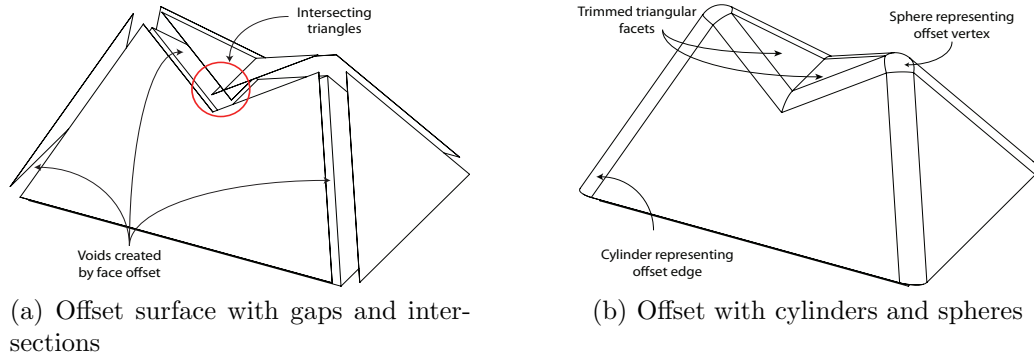


Figure 3.4: Offset surfaces with voids and gaps

need for surface topology is eliminated. Second, since each facet is treated individually the issue of sharp edges or vertices does not affect this methodology.

The separation of the offsetting procedure results in an offset surface comprising of a triangular face, three cylindrical surfaces, and three spherical surfaces respectively representing the offset of the face, edge, and vertices. As the sphere and cylinders are modelled with triangular faces, the offset of a single triangle results in many more triangles (in the excess of 7,000 triangles for a $0.118mm$ tolerance). A larger number of triangles increases the time required for data handling in subsequent steps.

The proposed method creates a bounding space by projecting the triangle in question onto a plane perpendicular to the tool axis, as shown on Figure 3.5a, and drawing vertical planes on the projected edges, Figure 3.5b. This creates a column around the projection; the volume enclosed by the three vertical planes is defined as the bounding space of the offset surface. The bounding spaces of the adjoining triangular faces do not intersect with each other and are distinct. Furthermore, if the original triangulation describes the bounding surface of a solid then it is continuous and with no gaps. The resulting bounding spaces are also continuous and without gaps. As the bounding space subdivides the part

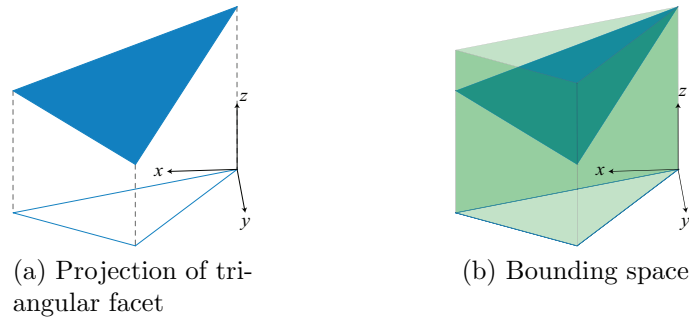


Figure 3.5: Offset surface for a sharp vertex

space, the offsetting procedure is designed to ensure that no gap is left in the bounding space when a triangle is offset. It may happen that when offsetting a triangle some offset parts may fall in the boundary space of other triangles; these are dealt with in the tool path planning strategy and described later. The vertical boundary planes are used to trim the spheres and cylinders thereby eliminating unnecessary triangles.

The need for a bounding space is exemplified in Figure 3.6. The surface offset for a solid model is created. Figure 3.6b shows the offset surfaces generated for each triangular facet in the solid model. The offset surfaces have been shifted so that each individual offset can be easily visualized. If the surface offsets were in the correct position, note how for the top vertex each individual offset generates a sphere resulting in four overlapping spheres for this example. This process results in redundant and unnecessary operations since only one sphere is needed and not four. Each edge offset also generates two cylinders as it is shared by two facets.

The objective of using a bounding space is to avoid these redundant offset surfaces. The offset surface for the same solid model, but now using a bounding space, is shown in Figure 3.7. By generating a bounding space for each of the triangular facets (Figure 3.7a) it is possible to limit the individual offset surfaces (Figure 3.7b) to a specific area

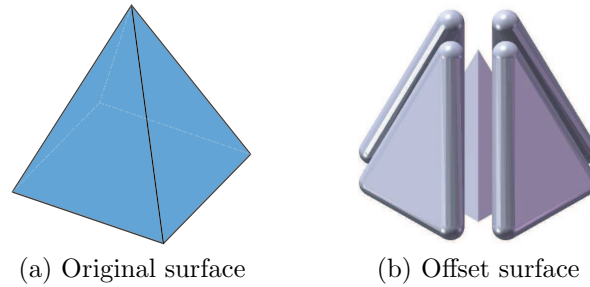


Figure 3.6: Offset surface for a sharp vertex using vertical planes

and avoid redundant surfaces. When the trimmed individual offsets are placed together they create a complete and gap free offset surface, Figure 3.7c. Notice how the surface resulting from the offsetting procedure does not have any gaps, Figure 3.7d. This example shows the advantages of using a bounding space to limit the individual offset surfaces. In this example the number of triangular facets used to describe the offset surface is reduced from 29,068 to 2,636. Offset surfaces result in a large number of triangular facets since very small triangles have to be used to describe the surface accurately.

The effect of trimming the face offset with the vertical planes is explained using Figure 3.8a. The cross-section of three triangles is shown along with their offset surfaces (dashed lines) and the vertical planes (dotted lines). The bounding space for each triangle is also shown with hatched lines. The individual offset surface for triangles *A* and *C* is simply the face offset since these facets are flat, i.e., perpendicular to the tool axis, and in these cases the face offset alone covers the entire bounding space. Triangle *B* requires an edge and vertex offset in addition to the face offset. The cross section of the edge and vertex offset is shown by the red dashed line. Trimming Triangle *B* such that it is within the boundary space results in a discontinuous offset surface, as shown in the figure. The discontinuity in the offset surface translates into an incorrect tool path.

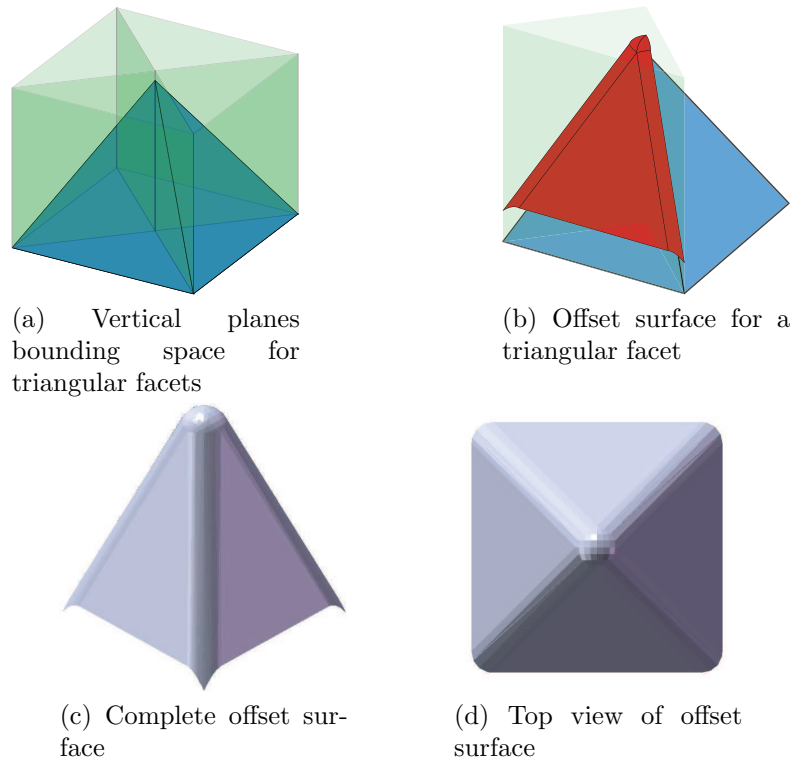


Figure 3.7: Offset surface for a sharp vertex using vertical planes

To avoid this, the bounding space is only used to trim surfaces generated from edge offset and vertex offset, and not the face offset. Not trimming the face offset means that the face offset for Triangle B will invade the bounding space of Triangle C and as a result the offset surfaces for these two triangles will intersect each other, as shown in Figure 3.8b. The intersection represents surface areas with dual representation (shown in blue line) that are invalid because they are not at the correct offset distance. Since the offset surface is used to create a tool path, the invalid surfaces translate into a tool path that gouges the surfaces during the machining process. It is impossible to detect intersecting facets without generating surface topology which is a time consuming process.

To avoid generating surface topology, a tool path planning strategy that can address

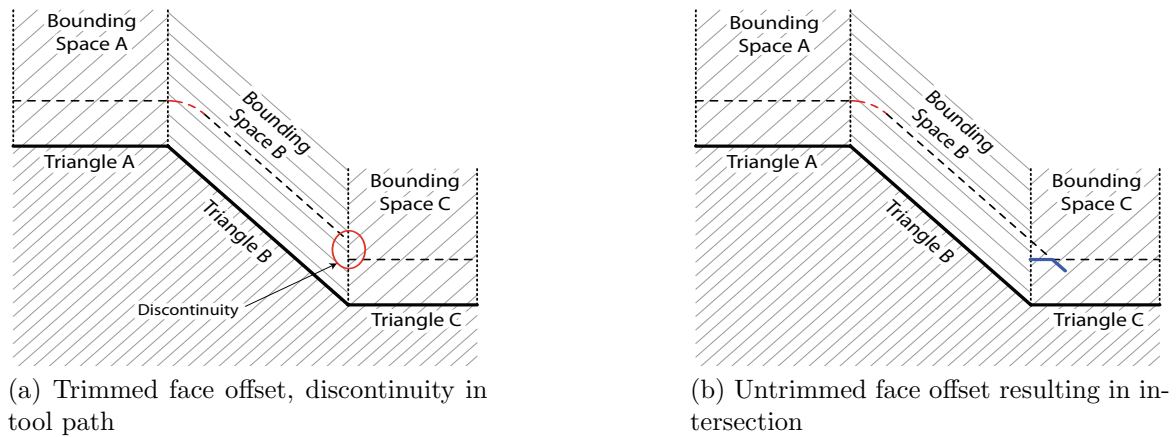


Figure 3.8: Effect of trimming face offset on tool path

this issue was designed. Figure 3.9a shows the cross section of an offset surface containing intersecting triangular surfaces. The surfaces, shown in red, must be eliminated to generate a correct tool path. Notice that if the offset surface is viewed from the direction of the tool axis (z -axis), the invalid surfaces are hidden by the valid surface, i.e., only the valid surfaces are visible from the direction of the tool axis. Thus, if the surface is rendered, computer graphics hidden-surface elimination algorithms will eliminate the hidden surfaces (invalid surfaces resulting from intersections) and only display those surfaces that are visible, that is the valid offset surface.

For a rendered surface, the distance to the object that is being displayed can be obtained from the graphics hardware. The hidden-surface elimination algorithm ensures that only the visible surfaces are displayed; thus, the distance information obtained is that of the valid surface only, i.e., the information from the hidden surfaces is ignored. The distance or depth information is stored in a block of memory called the *depth buffer*. Using the depth information a z -value can be found for any given x - y location, thus finding valid cutter locations for the tool path.

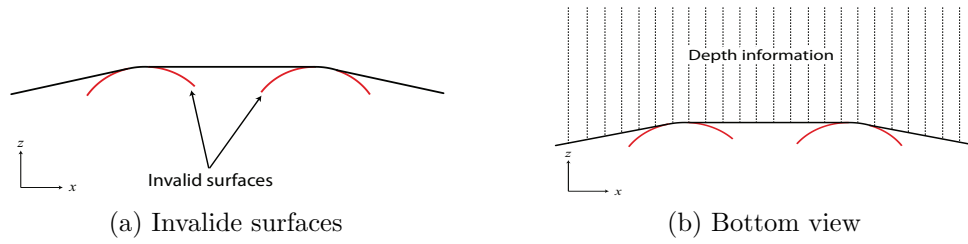


Figure 3.9: Handling intersecting offset surfaces

In addition to optimizing the offset surface generation by defining a bounding space, the offset surface can be generated for any cutter in the APT geometric definition of cutters. The face, edge, and vertex offset are found based on the surface normal and tool parameters (r_1 and r_2). By taking the tool parameters into account in the procedure it is possible to generate offset surfaces for different tool types, i.e., ball nose ($r_2 = 0$), flat ($r_1 = 0$), and radiused end mill. This allows creating tool paths for the different tools that are used in the NC machining process. It is essential for the method to allow offset surface for different types of cutters since different types of cutters are used in machining processes, e.g., using a flat end mill for roughing and ball nose end milling for the finishing pass.

A comparison of the current offset surface method with previous methods is shown in Table 3.1. Qu and Stucker's method is the only one that does not require edge and vertex offset. This method; however, is an inaccurate method, requires surface topology, and only generates offset surfaces for a ball nose cutter. Kim *et al.* have a method based on the multiple normal vectors of a vertex. This method requires surface topology and the offset surface results in intersections that require post processing. Furthermore, it is only designed for ball nose cutter. Kim and Yang's method has the advantage of working for generalized cutter and no intersections are created in the offset surface. However, it has the disadvantage of requiring surface topology and the offset surface is not exact since it

makes approximations at sharp edges and vertices. The current research method has the advantages of not requiring surface topology, works for any APT cutter, and results in an exact offset surface. The only disadvantage it has is the intersections that occur in the offset surface. As mentioned previously, this is handled in the tool path planning strategy.

Table 3.1: Comparison of offset surface methods

	Requires surface topology	Edge and vertex offset required	Intersections	Generalized APT Cutter	Exact
Qu and Stucker[36] <i>(Vertex offset, weighted normals)</i>	✓	✗	✗	✗	✗
Kim <i>et al.</i> [25] <i>(Multiple normal vectors of a vertex)</i>	✓	✓	✓	✗	✗
Kim and Yang [26] <i>(Multiple normal vectors of a vertex)</i>	✓	✓	✗	✓	✗
Salas Bolanos <i>(Face offset)</i>	✗	✓	✓	✓	✓

3.3 Offset Surface for a Generalized Cutter

The three steps required to generate an offset surface for a generalized cutter are presented in this section. The first step is a face offset that consists of moving the triangular facets along the offset vector. This leaves gaps at convex edges and vertices on the surface. In the second step, cylindrical surfaces are drawn to fill in the gaps created by offsetting convex edges. Finally, the third step fills in the gaps created by offsetting convex vertices by creating spherical surfaces at these points.

3.3.1 Face Offset

The offset of triangular facets is obtained by translating the vertices along the offset vector \bar{T}_c . Given that the offset surface represents the CL point for machining, the objective is to use \bar{T}_c to translate the vertices of the triangular facet to their respective CL points. Offset vector \bar{T}_c is the result of adding two vectors since the APT cutter used in this research is defined by parameters: r_1 (circular insert) and r_2 (core radius). As shown in Figure 3.10 vector \bar{T}_c is the result of adding \bar{T}_1 and \bar{T}_2 . \bar{T}_1 translates the vertices from their original position to center of the circular insert. It is defined by:

$$\bar{T}_1 = \hat{n} \cdot r_1 \quad (3.1)$$

where, \hat{n} is equal to surface normal of the facet.

Vector \bar{T}_2 then translates the vertex to reference point of the cutting tool. This is obtained by projecting the surface normal, \hat{n} , onto the tool axis, \hat{T} , resulting in vector \hat{T}_{proj} . \bar{T}_2 is the result of subtracting \hat{T}_{proj} from \hat{n} . \bar{T}_2 can be expressed as:

$$\bar{T}_2 = \frac{\hat{n} - \hat{T}_{proj}}{|\hat{n} - \hat{T}_{proj}|} \cdot r_2 = \frac{\hat{n} - (\hat{n} \cdot \hat{T}) \hat{T}}{|\hat{n} - (\hat{n} \cdot \hat{T}) \hat{T}|} \cdot r_2 \quad (3.2)$$

where the tool axis, \hat{T} , is equal to $[0 \ 0 \ 1]^T$.

Thus, the offset vector, \bar{T}_c , is given by:

$$\bar{T}_c = \bar{T}_1 + \bar{T}_2 = \hat{n} \cdot r_1 + \frac{\hat{n} - (\hat{n} \cdot \hat{T}) \hat{T}}{|\hat{n} - (\hat{n} \cdot \hat{T}) \hat{T}|} \cdot r_2 \quad (3.3)$$

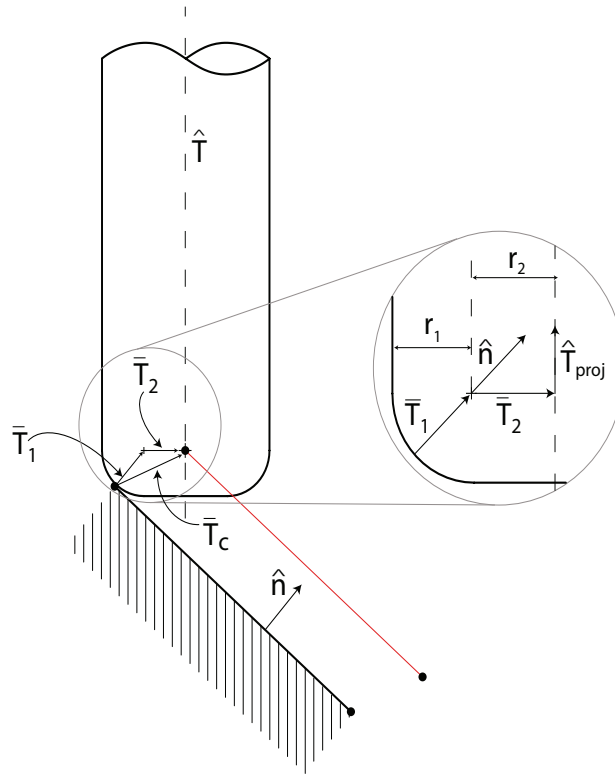
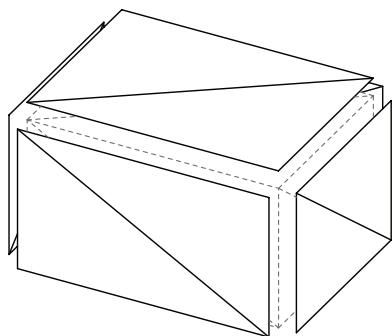


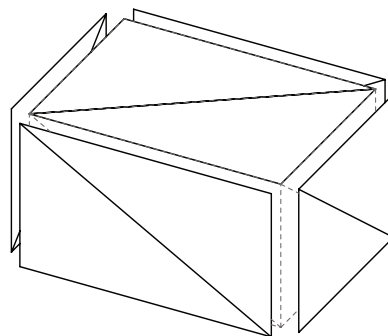
Figure 3.10: Face offset geometry for a generalized cutter

Figure 3.11 shows the result of applying the face offset to a box. In Figure 3.11a the

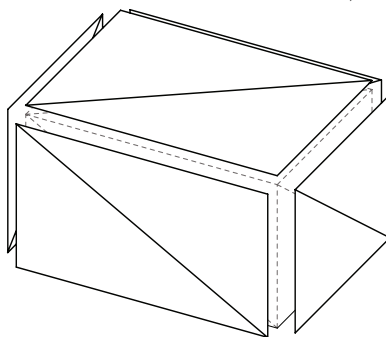
offset distance is uniform regardless of the facet orientation since it is the offset surface for a ball nose end mill. Figure 3.11b shows the offset surface for a flat end mill. Note how in this case the top face's offset distance is zero. This will occur in cases where the original facet is flat since the reference point for the flat end mill is at the bottom of the cutter, i.e. $r_1 = 0$. Finally, Figure 3.11c shows the offset surface for a radiused end mill. In this case the offset distance for the top faces is $1.25mm$ compared to $2.5mm$ of the ball nose end mill due to the value of r_1 .



(a) Ball end mill face offset ($r_1 = 2.5mm, r_2 = 0.0mm$)



(b) Flat end mill face offset ($r_1 = 0.0mm, r_2 = 2.5mm$)



(c) Radiused end mill face offset ($r_1 = 1.25mm, r_2 = 1.25mm$)

Figure 3.11: Face offset for ball, flat, and radiused end mill

3.3.2 Edge Offset

Gaps created on convex edges are a result of the triangular face offset. These gaps are filled by applying an edge offset. The edge offset consists of surfaces along the edges of the triangular facets. In the case of the ball nose end mill, the surface created is exactly a cylinder with a radius equal to the radius of cutter. However, in the case of the flat end mill and the radiused end mill the offset surfaces are not a cylinder. The shape of these surfaces change with varying surface normal. The approach that is described in this section works for any type of edge: concave, convex, and flat.

Similar to the face offset, the offset vector \bar{T}_c is the result of adding two vectors: \bar{T}_1 and \bar{T}_2 as can be seen in Figure 3.12. Vector \bar{T}_1 generates a cylinder with radius r_1 along the edge and vector \bar{T}_2 translates these points to the reference point of the cutter.

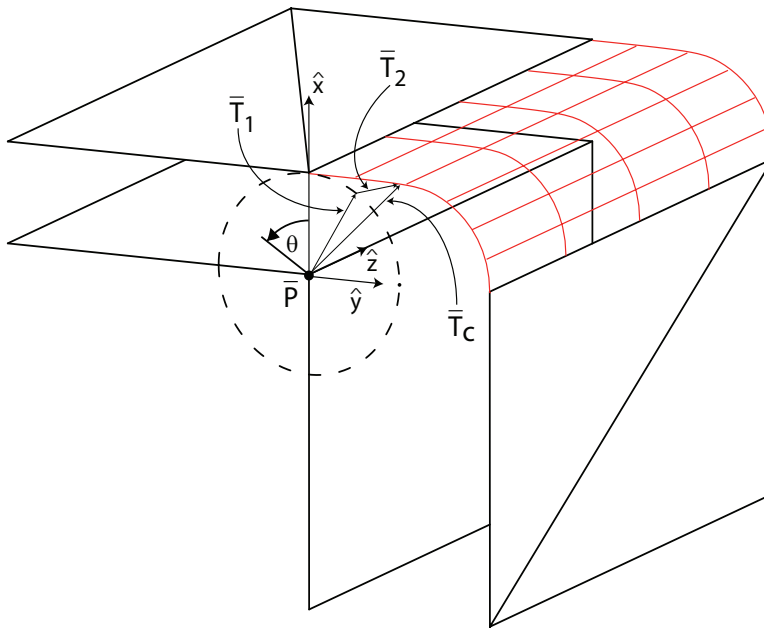


Figure 3.12: Edge offset geometry for a generalized cutter

Vector \bar{T}_1 is calculated by first creating a local coordinate system on of the vertices of

the edge. \hat{z}_1 is aligned with the edge for which the offset is being generated. \hat{y}_1 and \hat{x}_1 are found based on the cross products of \hat{z}_1 and the tool axis \hat{T} .

$$\hat{y}_1 = \frac{\hat{z}_1 \times \hat{T}}{|\hat{z}_1 \times \hat{T}|} \quad (3.4)$$

$$\hat{x}_1 = \frac{\hat{y}_1 \times \hat{z}_1}{|\hat{y}_1 \times \hat{z}_1|} \quad (3.5)$$

Using the values of \hat{x}_1 and \hat{y}_1 in the equation of a cylinder, vector \bar{T}_1 is calculated.

$$\bar{T}_1 = r_1 \cdot \hat{x}_1 \cdot \cos \theta + r_1 \cdot \hat{y}_1 \cdot \sin \theta \quad (3.6)$$

where θ is the central angle of the surface being generated. Vector \bar{T}_2 is calculated using the same procedure that was used for the face offset:

$$\bar{T}_2 = \frac{\hat{z} - (\hat{z} \cdot \hat{T}) \hat{T}}{|\hat{z} - (\hat{z} \cdot \hat{T}) \hat{T}|} \cdot r_2 \quad (3.7)$$

Then, the edge offset vector \bar{T}_c is given by

$$\bar{T}_c = \bar{T}_1 + \bar{T}_2 = r_1 \cdot \hat{x}_1 \cdot \cos \theta + r_1 \cdot \hat{y}_1 \cdot \sin \theta + \frac{\hat{z} - (\hat{z} \cdot \hat{T}) \hat{T}}{|\hat{z} - (\hat{z} \cdot \hat{T}) \hat{T}|} \cdot r_2 \quad (3.8)$$

Figure 3.13 shows the result of applying the edge offset along with the face offset to a box. Figure 3.13a depicts the offset for a ball nose cutter. In this case the edge offset results in a cylinder drawn along the edges. Figure 3.13b shows the edge offset for a flat end mill. In this case the offset is flat since the face is aligned with the tool axis. However, this

will not always be the case. Finally, Figure 3.13c shows how the radius for the *cylindrical* surface generated by the edge offset transitions from r_1 to $r_1 + r_2$.

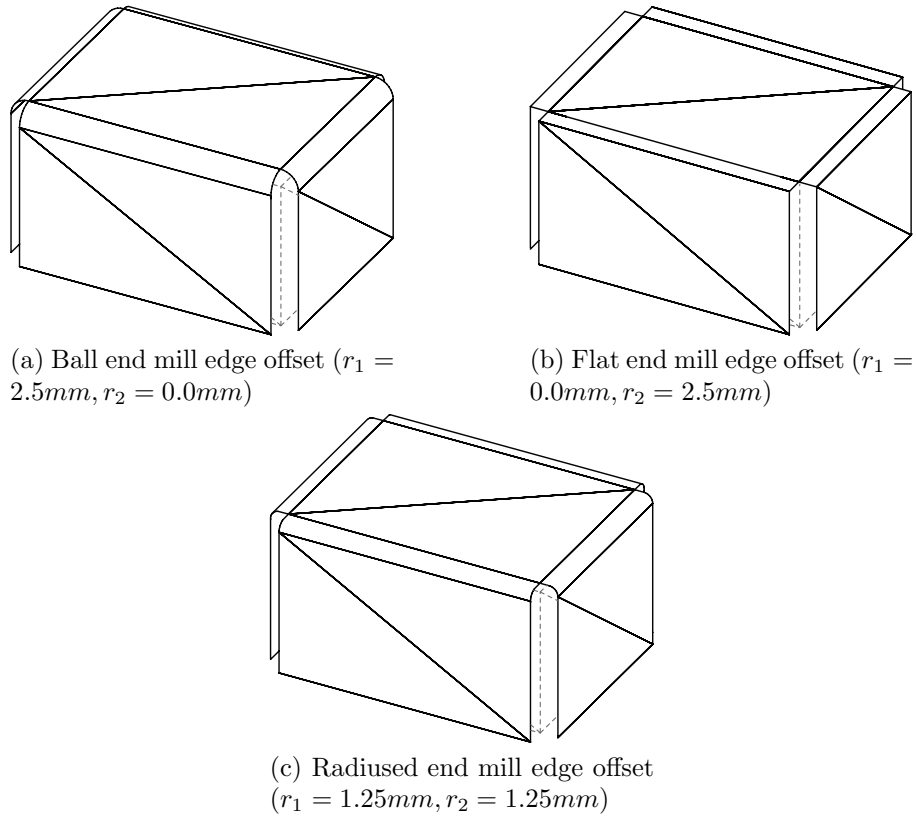


Figure 3.13: Edge offset for ball, flat, and radiused end mill

3.3.3 Vertex Offset

The last step in generating the offset surface for a triangular mesh is the vertex offset. The objective of the vertex offset is to fill in the gaps created at convex vertices during the face offset. In the edge offset process surfaces were created at the edges. In the same manner, surfaces are created as vertices are offset uniformly. The result of the vertex offset for a

ball nose end mill, a type of generalized cutter, is a sphere. However, for the flat end mill, and the radiused end mill the surface resulting from this process will vary with changing surface normal.

The offset vector \bar{T}_c results from adding \bar{T}_1 and \bar{T}_2 as can be seen in Figure 3.14. Vector \bar{T}_1 generates a sphere with radius r_1 along the edge and vector \bar{T}_2 translates the sphere points to the reference point of cutter. In the case of the ball nose cutter $r_2 = 0$; thus, $\bar{T}_2 = 0$ and a perfect sphere is generated. Vector \bar{T}_1 based on the parametric equation of

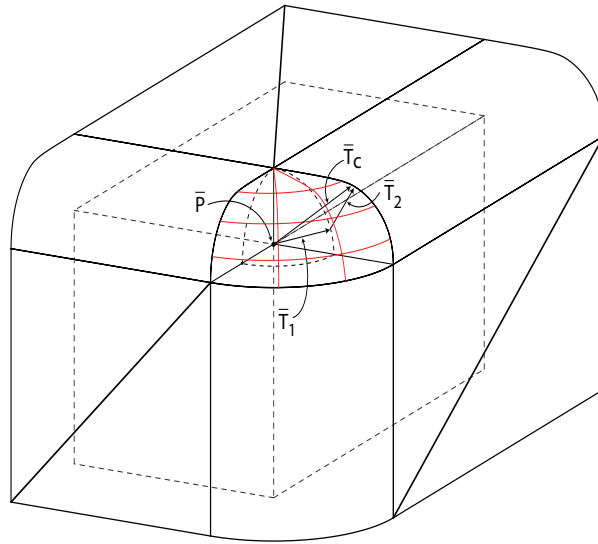


Figure 3.14: Vertex offset geometry for a generalized cutter

a sphere. Vector \bar{T}_1 will result in a sphere with radius equal to r_1 .

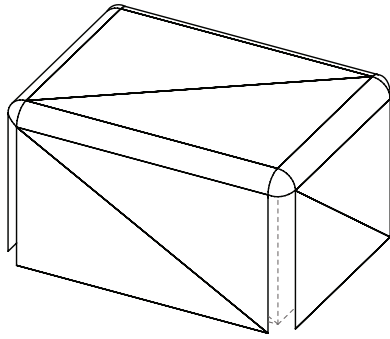
$$\bar{T}_1 = r_1 \cdot \cos \phi \cdot \cos \theta \cdot \hat{i} + r_1 \cdot \cos \phi \cdot \sin \theta \cdot \hat{j} + r_1 \cdot \sin \phi \cdot \hat{k} \quad (3.9)$$

Vector \bar{T}_2 is calculated using the same procedure that was used for the face and edge offset. In this case; however, rather than projecting the surface normal onto the tool axis, vector \bar{T}_1 is projected onto the tool axis. The value of \bar{T}_2 is calculated by:

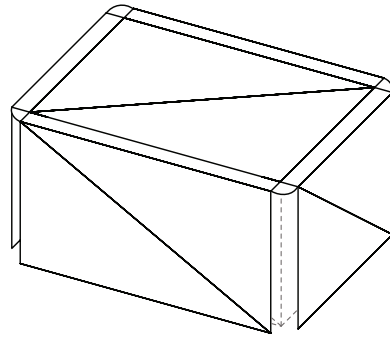
$$\bar{T}_2 = \frac{\bar{T}_1 - (\bar{T}_1 \cdot \hat{T}) \hat{T}}{|\bar{T}_1 - (\bar{T}_1 \cdot \hat{T}) \hat{T}|} \cdot r_2 \quad (3.10)$$

Therefore, the edge offset vector \bar{T}_c is given by:

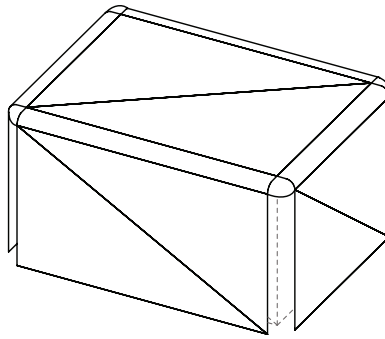
$$\bar{T}_c = r_1 \cdot \cos \phi \cdot \cos \theta \cdot \hat{i} + r_1 \cdot \cos \phi \cdot \sin \theta \cdot \hat{j} + r_1 \cdot \sin \phi \cdot \hat{k} + \frac{\bar{T}_1 - (\bar{T}_1 \cdot \hat{T}) \hat{T}}{|\bar{T}_1 - (\bar{T}_1 \cdot \hat{T}) \hat{T}|} \cdot r_2 \quad (3.11)$$



(a) Ball end mill vertex offset
($r_1 = 2.5mm, r_2 = 0.0mm$)



(b) Flat end mill vertex offset
($r_1 = 0.0mm, r_2 = 2.5mm$)



(c) Radiused end mill vertex offset
($r_1 = 1.25mm, r_2 = 1.25mm$)

Figure 3.15: Vertex offset for ball, flat, and radiused end mill

3.3.4 Examples of Offset Surfaces

To verify the offset surface methodology, offset surfaces for two geometries were created. The two geometries chosen are a square pyramid and a model containing a saddle vertex. These geometries are chosen as they contain a combination of convex and concave regions. It is at the edges and vertices of these regions that previous offset surface methodologies struggle at generating uniform and accurate offset surfaces.

Figure 3.16a shows the original definition of the square pyramid. Figures 3.16b-d show the result of creating offset surfaces for a ball nose, flat, and radiused cutter. The offset surfaces are uniform and complete, i.e., with no gaps at the sharp edges or vertices. It can be seen how the edge and vertex offset for the ball nose cutter results in nicely defined cylinders and spheres regardless of orientation. Whereas in the case of the flat and radiused the shape of the cylindrical and spherical surfaces vary with the orientation of the edges and vertices respectively.

Figure 3.17 shows the result of offsetting a surface containing a saddle point. Previous offset surface methodologies struggled with saddle points given that the faces meeting at this vertex can be both convex and concave. The results shown on Figure 3.17 prove that this method is successful at creating an offset surface for saddle vertices.

The method described is successful at offsetting a surface. However, the offset surface results in overlapping and unnecessary triangles since complete cylindrical and spherical surfaces are being created at each edge as shown previously in this chapter. To generate the cylindrical and spherical surfaces within an acceptable tolerance a large number of triangular facets are required. For example, to generate a cylinder ($5mm$ radius) within a $0.118mm$ tolerance a total of 70 triangular facets are required. In addition, to generate a

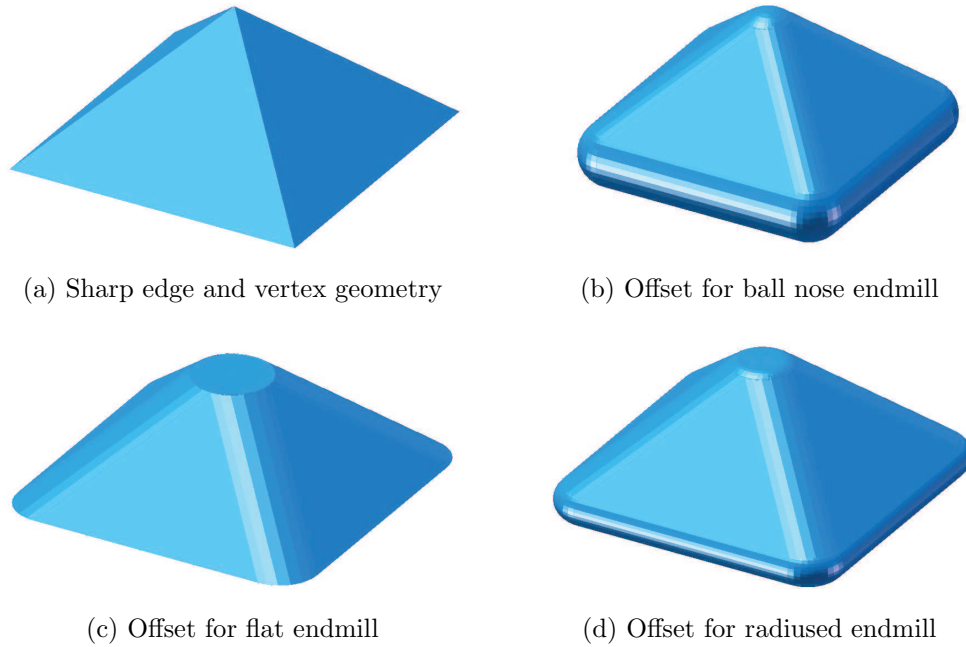


Figure 3.16: Offset surface for square pyramid

sphere ($5mm$ radius) with the same tolerance, a total of 2352 triangles are needed. In total, a single triangular offset surface would require 7266 triangular facets. If a design originally contains 50,000 triangles, an average amount for a complex surface model, then a total of 363,300,000 triangular facets would be required. This results in large memory requirements and very long processing time. Even more, in the most extreme case (vertical triangular facet) only a quarter of the 7266 triangles are required. Therefore, steps were taken to avoid the generation of the unnecessary triangles and optimize the tool path generation process. For this, the concept of *bounding space* created based on the triangle edges is introduced in the next section.

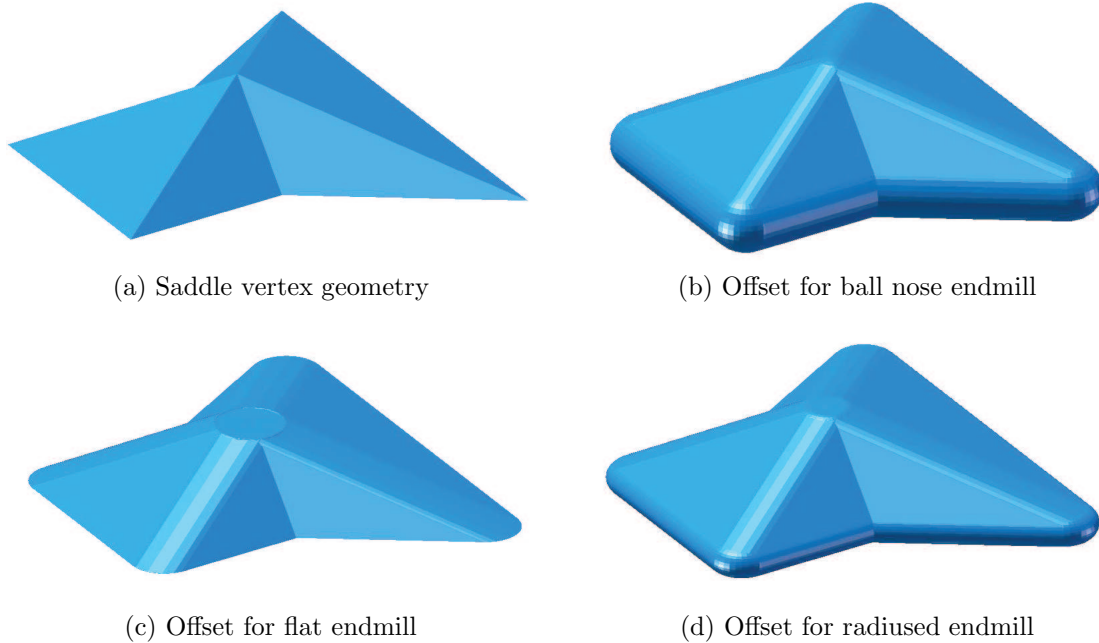


Figure 3.17: Offset surface for a sharp vertex

3.4 Bounding space

To generate the partial cylindrical and spherical surfaces there are two options. The first is to generate the entire surface and then trim the triangles that are outside the bounding space. This option results in calculations required to generate the entire surface and additional calculations for trimming each surface. The second option is to simply stop generation of the surfaces when the vertices of the triangles are outside of the bounding space. Even though this implies that a check must be performed for every triangle that is generated, this section shows how the amount of calculations is still less than the first option. Figure 3.18a shows a complete edge offset. Figure 3.18b shows a partially generated edge offset. By stopping the generation of the cylindrical surface it was ensured that the edge offset is within the bounding space.

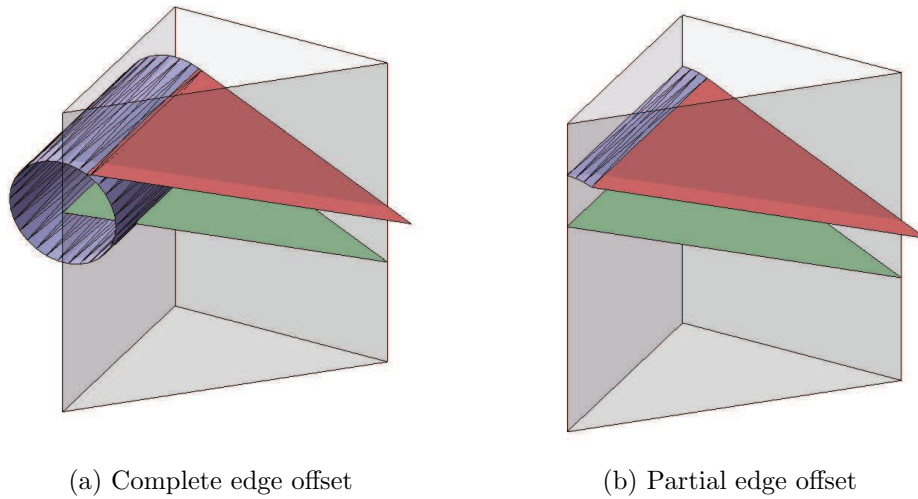


Figure 3.18: Comparison of complete and partial edge offset

The plane equation is $Ax + By + Cz = d$, where A , B , and C are the x , y , and z components of the surface normal of a plane respectively, and d is the distance from the origin to the point on the plane which is nearest to the origin. Any given point can be checked to see which side of the plane it lies on by calculating the dot product of the surface normal and the point, and comparing the value with d . If it is greater than d , the point is on one side, if it is smaller, it is on the other side. Figure 3.19 shows the normal direction of the three vertical planes; a point is on the positive side of a plane if it lies on the same side as the plane normal.

Figure 3.20a shows the surface from Figure 3.18, in this case viewed from the direction of the tool axis. The area that the edge offset must cover is marked by the red hatch. When the surface is viewed from the side, Figure 3.20b, the starting and ending points for the edge offset can be easily viewed. The start point is given by the face offset vector and the end point is found using the vertical planes.

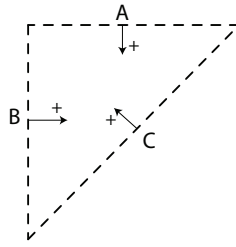


Figure 3.19: Vertical planes positive direction

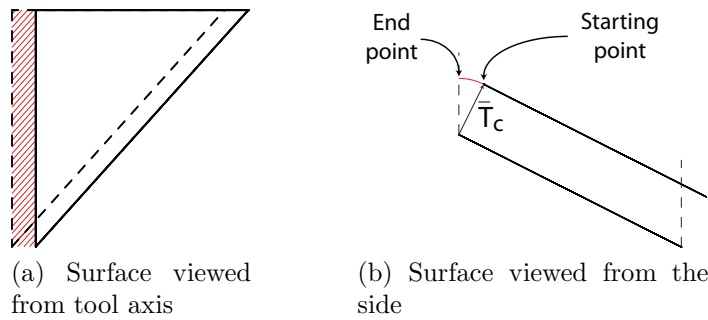


Figure 3.20: Starting and ending point for offset surface generation

The process to find the end point for the edge offset is shown in Figure 3.21: the red points are vertices for triangular facets that are being generated, the dotted lines represent the bounding space when viewed from the direction of the tool axis, and the solid black line represents the face offset. In this case the edge offset must be on the positive side of vertical plane B to be within the bounding space. The starting point of edge offset is given by the face offset vector. Then, vertices for the triangular facets required to generate an accurate cylindrical surface are generated. Each time a vertex is generated, the dot product of plane B and the vertex is calculated to ensure that it is on the positive side of plane B . The edge offset is stopped when the dot product indicates that the vertex is on the negative side of plane B .

Applying this procedure to every edge and vertex offset will result in partial surfaces

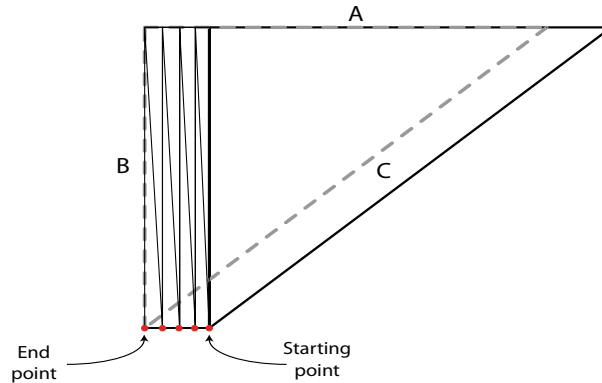


Figure 3.21: Generation of partial surface

enclosed by the bounding space. In the vertex offset, the points have to be checked with respect to two planes. Checking each vertex involves performing a dot product for each plane. These operations are performed quicker than the operations required to generate the entire surface and the amount of triangular facets for subsequent operations is significantly reduced. This results in a reduction in processing time.

Once the offset surface has been generated the part can be machined by placing the cutter reference point on the offset surface and moving it along the surface as dictated by the tool path footprint. However, the surface definition (STL file) cannot be directly used by CNC machines to machine the part. The offset surface information must be converted into a series of points that the cutter reference point must visit. The process of converting the offset surface into a tool path is described in the next section.

3.5 Tool Path Generation

A common approach to generate a tool path from an offset surface is to intersect a vector aligned with the tool axis at different x and y locations and find the corresponding z coordinate. This is a simple implementation but time consuming given that for a given (x,y) location the triangles intersecting the vector are not known. Thus, the vector must be checked with every triangle or bucketing algorithms must be used to create a partial surface topology of the triangular mesh to expedite the process [37]. Another approach is to intersect the offset surface with a series of planes, as shown in Figure 3.22. The line resulting from the intersection of the plane and the offset surface is the path along which the cutter reference point must travel. This process also requires surface topology since the triangular facets that the plane is intersecting at a given position have to be known; furthermore, given that the offset surface contains intersecting triangles the resulting path would require post processing to eliminate the unwanted line segments. This section describes a methodology for obtaining a valid tool path from the offset surface without the need for trimming operations thereby speeding up the computation.

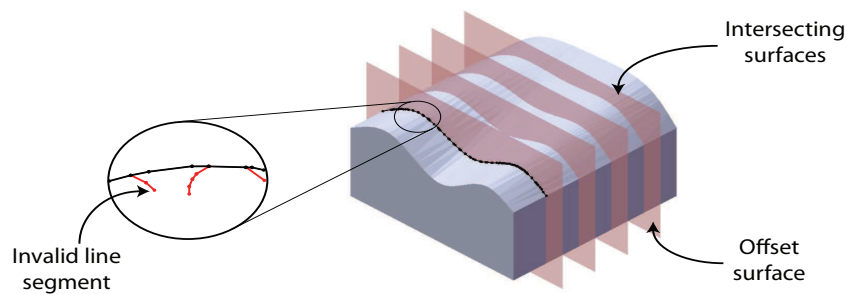


Figure 3.22: Getting tool path by intersecting planes with offset surface

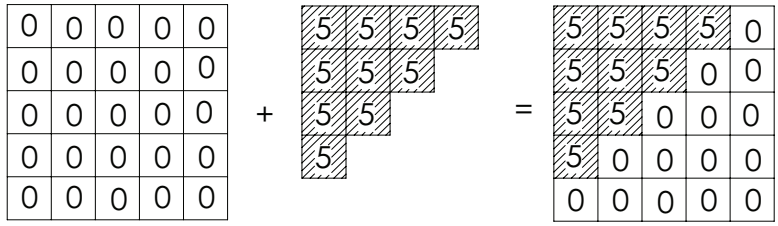
Once the offset surface is generated it can be rendered using standard graphic libraries. Render engines use hidden surface removal algorithms to determine which surfaces are not

visible from a certain viewpoint. If the offset surface is rendered with the viewpoint looking down on the surface from the direction of the tool axis, then the invalid surfaces are not shown since they are hidden by the offset surface. Information such as color and depth for any point on the displayed image can be accessed from the graphics hardware. As the information is only available for the displayed surfaces, the information accessed is that of the valid offset surface and not the invalid surfaces resulting from intersections. The depth for every point on the image is stored in the *depth buffer*. The depth buffer holds the distance to the object that is being displayed from the viewpoint. Using the depth information the z coordinate for a given x and y position can be found; thus, the cutter locations along the tool path foot print can be found.

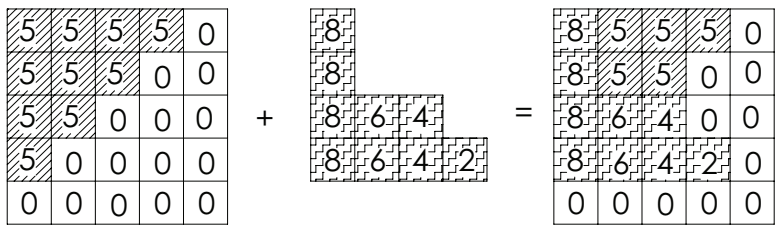
3.5.1 Depth Buffer

The goal of every graphics program is to draw pictures on a screen. A screen is composed of a rectangular array of pixels. Each of these pixels is capable of displaying a tiny square of color. To do this, the pixels require information such as color and depth. Whenever data is stored uniformly for each pixel, the storage is called a *buffer*. Examples of buffers are color, depth, stencil and accumulation. The depth buffer (also called z -buffer since the z -value measures the distance perpendicular to the screen) stores a depth value of the nearest or visible surface surface at each pixel [40]. The depth buffer records the information associated with the largest z for each (x,y) as shown on Figure 3.23.

Depth buffers have already found applications in NC machining such as tool path planning and mechanistic modelling of milling processes [10, 14, 5]. The use of depth buffers is becoming common in manufacturing due to the benefits associated it with it. First, the



(a) Adding polygon of constant z to empty z -buffer



(b) Adding polygon that intersects the first polygon

Figure 3.23: Depth buffer

cost of high end graphics hardware has decreased drastically due to the popularity of video games. Second, rendering engines have been optimized and many operations are performed directly in the graphics hardware leaving CPU open to other tasks.

3.5.2 Cutter Locations from Depth Buffer

Once the offset surface has been generated, it can easily be rendered using a graphics library since the geometry is comprised of triangular facets. The rendered model is oriented such that the viewing direction is aligned with the tool axis. Once the model has been rendered the depth buffer value for any pixel can be read. Each component is converted to floating point such that the minimum depth value maps to 0 and the maximum depth value maps to 1 as shown on Figure 3.24. The depth buffer is converted into the corresponding z -

coordinate by applying the formula:

$$z = (\text{depth buffer}) \times (\text{max}Z - \text{min}Z) \quad (3.12)$$

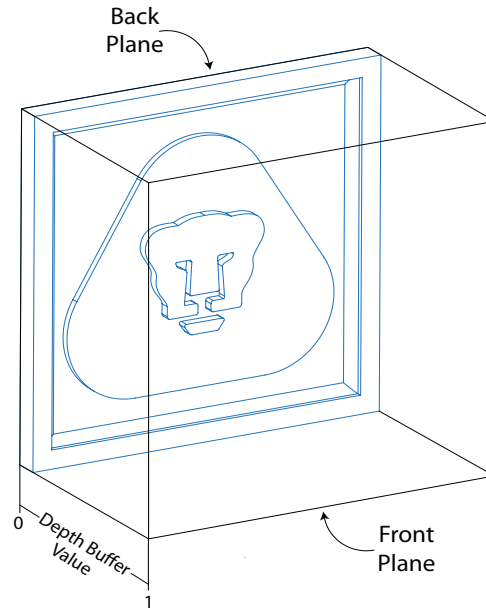


Figure 3.24: Depth buffer values in view volume

By finding the depth buffer of the entire window the CLs for the tool are found. Since the depth buffer can only be found for a given pixel, an important step for using the depth buffer to obtain the cutter locations is determining the size of the window on which the part is displayed. The size of the window must be determined based on the range of x and y values and the desired side step in both directions. The size of the display window can be found using the formulas:

$$xPixels = (\text{max}X - \text{min}X) / \text{sidestep}X \quad (3.13)$$

$$yPixels = (maxY - minY)/sidestepY \quad (3.14)$$

These equations ensure that the area of the model rendered in a pixel is equal to *side step* $x \times$ *side step* y and thus the desired side step is achieved as shown on Figure 3.25. This means that this method is dependant on the screen resolution on which the model is being rendered. Based on the resolution of the screen there is a limit on the minimum side step possible.

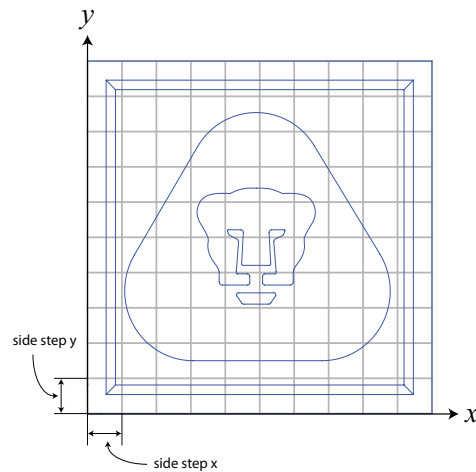


Figure 3.25: Depth buffer pixels

3.6 Results

The offset surface method is used to create toolpaths for several complex surfaces and verified by simulation and NC machining. An in-house NC machining simulator developed

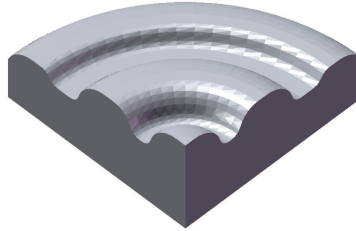
using C++ and OpenGL libraries was used for the machining simulations. This simulator has been found to successfully simulate metal and wood milling operations [15, 42]. The machining tests were carried out on a 3-axis CNC router with linear accuracy of $\pm 0.001in$. The machining tests were performed on different types of wood.

The model shown in Figure 3.26a is a complete triangular mesh generated from a solid model using SolidWorks 2007. The number of triangular surfaces is 1548 with a tolerance of 0.01 mm. Figure 3.26b-c show the offset surface and machining simulation results for a ball nose cutter with radius equal to 3.175mm. Similarly, Figure 3.26d-e and Figure 3.26f-g show the offset surface and machining simulation for a flat endmill and a radiused endmill, respectively. The offsetting of the surfaces takes 10.2 seconds when offset on an Intel® Core2 Quad CPU @2.83GHz.

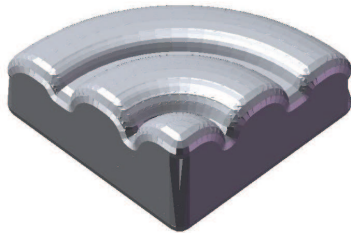
The toolpath for machining two custom name plaques was created using the proposed offset surface algorithm. The first plaque is shown in Figure 3.27. The design is made up by the text “*Sanjeev Bedi*” written on top of a traditional Indian background. The overall dimensions of the plaque are $200mm \times 140mm$, and the STL definition is comprised of 56,170 triangular facets. The plaque was machined on purple heart wood and required two roughing passes using a 0.5in ball nose cutter, and a finishing pass using a 0.125 in ball nose cutter with a 0.3mm side step. The simulated finishing pass is shown in Figure 3.27b and the machined result is shown in Figure 3.27c.

The second plaque is shown in Figure 3.28. The design is made up by the text “*Stephen Mann*” written on top of a frieze pattern. The plaque was machined on paddock wood and required two roughing passes and a finishing pass with a 0.4mm side step. The simulated finishing pass is shown in Figure 3.28b and the machined result is shown in Figure 3.28c. The overall dimensions of this plaque are $120mm \times 100mm$. The STL representation

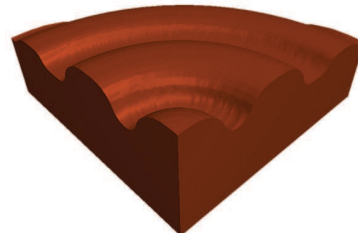
of the model required 28,720 triangles and processing time to generate the finishing tool path was 53.1 seconds. The simulation and machining results prove that the offset surface methodology described works properly.



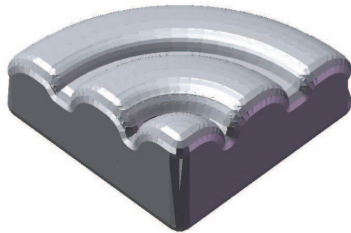
(a) STL model



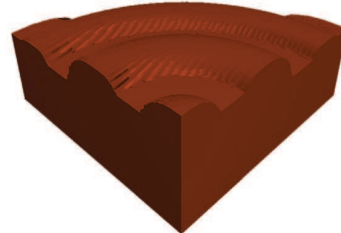
(b) Offset for ball nose endmill $r_1 = 3.175mm$ and $r_2 = 0.0mm$



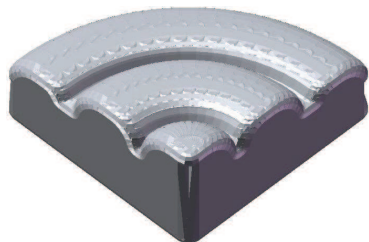
(c) Simulation of machining using ball nose endmill



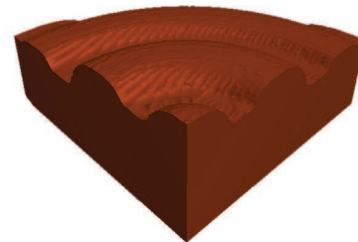
(d) Offset for flat endmill $r_1 = 0.0mm$ and $r_2 = 3.175mm$



(e) Simulation of machining using flat endmill



(f) Offset for radiused endmill $r_1 = 1.5875mm$ and $r_2 = 1.5875mm$



(g) Simulation of machining using radiused nose endmill

Figure 3.26: Simulation test of sculptured surface



(a) STL model of plaque

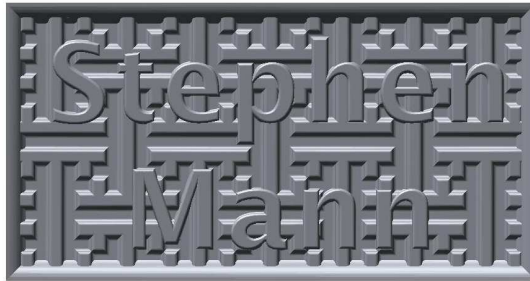


(b) Simulation of finishing pass using ball nose cutter, $r_1 = 1.5875mm$ and $r_2 = 0.0mm$

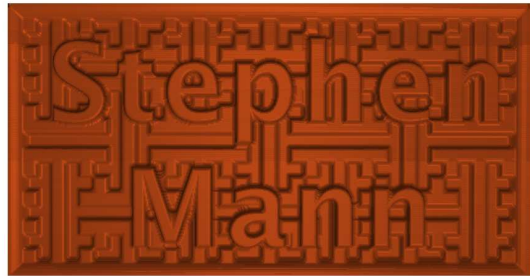


(c) Machined plaque, $r_1 = 1.5875mm$ and $r_2 = 0.0mm$

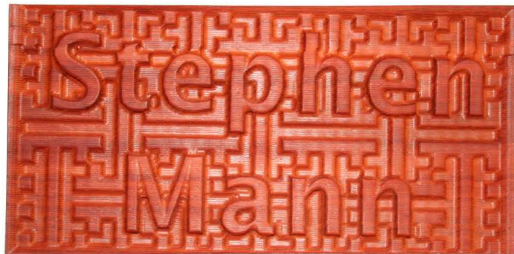
Figure 3.27: Simulation and machining result for custom name plaque, Sanjeev Bedi



(a) STL model of plaque



(b) Simulation of finishing pass using ball nose cutter, $r_1 = 1.5875mm$ and $r_2 = 0.0mm$



(c) Machined plaque, $r_1 = 1.5875mm$ and $r_2 = 0.0mm$

Figure 3.28: Simulation and machining result for custom name plaque, Stephen Mann

3.6.1 Comparison of Offset Surface Method with Ball Drop Method

The time required to generate a tool path using the offset surface method was compared to the time required to generate a tool path using the *ball drop* method [33]. Tool paths for models with varying number of triangular facets were generated and timed. A second degree polynomial regression was applied to the data, the results are shown in Figure 3.29.

When the number of triangles is small, both these methods have approximately the same processing speed. However, as the number of triangles increases the offset surface method begins to out perform the ball drop method. The processing time increases linearly for the offset surface method and quadratically for the ball drop method. The reason this happens is that the offset surface method handles each triangle individually; thus, an increase in number of triangles results in a linear increase in time. In the case of the ball drop method, an increase in the number of triangles affects the bucketing time, number of triangles in a bucket, and the number of triangles that have to be checked. Thus, the time increases much more rapidly.

The offset surface method is much faster at generating tool paths since some of the calculations are carried out in the graphics processing unit (GPU) which results in hardware acceleration. The total processing time is also less since the calculations required for the offset surface are simple and less are required. In the drop ball method there is a greater number of calculations required and in some cases second degree polynomials must be solved to find a valid tool path.

In addition to the total time for generating a tool path intermediate times were also checked. For the offset surface it was found that the average time required to offset a triangular facet is approximately 1.43 milliseconds. This time varies, but not significantly,

depending on the number of triangles that are generated in the edge and vertex offset. Whereas, for the ball drop method it was found that the processing time for a given tool path location can vary significantly and is highly dependant on the concentration of triangles at a given location. For example, in a flat area of the model where the number of triangles in a bucket is very small the processing time is fast. However, in areas such as fillets where the concentration of triangles is very large, the processing time increases. This is a big disadvantage for the ball drop method since sculptured surfaces are prone to high concentration of triangles due to there complex geometries.

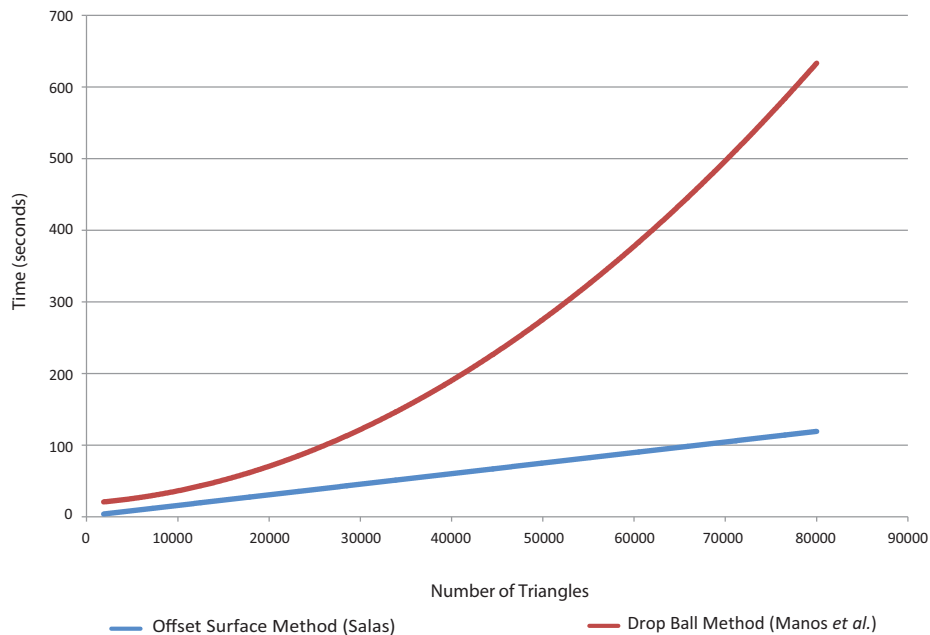


Figure 3.29: Tool path generation time comparison

Chapter 4

Optimal Tool Selection

Selection of an optimal cutting tool is a difficult task given the many parameters that are used to evaluate the effectiveness of the tool. Extensive research has been carried out in tool selection based on tool life, manufacturing cost, and required surface finish. Most of this research has mainly focused on determining the optimal tool size and not the tool type. Selection of an appropriate tool type is important to obtain an optimal NC machining process, specially for complex surfaces.

When machining a curved surface a ball-nosed cutter is usually used. The reasons behind this are: ball nose end mills are easy to position in relation to curved surfaces and tool paths for this type of tool are easily generated. For the machining of plane surfaces flat end mills are used. This because flat-ended cylindrical end mills match the surface being machined exactly [45].

Optimal tool selection helps achieve two conflicting objectives: quality and efficiency [11]. This chapter presents a method for determining which tool type, ball nose or radiused end mill, is best suited to machine a sculptured surface. The research presented determines the amount of material removed by the machining process and the resulting scallop heights.

Based on these two factors an optimal tool is selected for a specific surface. By selecting an optimal tool type an NC machining process can be optimized by reducing machining time and improving surface quality.

4.1 Material Removal

When machining a surface it is desired that the amount of stock material leftover in the finished product is minimal. A comparison of volume removed by each cutter is calculated to determine which cutter is best suited for a specific CL point. At any given position on the surface being machined the tool position of a radiused cutter and a ball nose cutter can be found using the tool positioning method described in the previous chapter. The offset surface guarantees that the CL for a ball nose and radiused cutter is found at the lowest point possible without gouging the surface being machined. This means that the cutter position chosen represents minimum volume left over by each tool without gouging the part.

Figure 4.1 shows the ball nose and radiused end mills at one point. The two tool positions have been superimposed. The ball nosed cutter is shown to the left and the radiused end mill is shown to the right. The distance between the center of the ball and the torus is designated as d . At this point the ball nosed cutter machines the hatched area that cannot be cut by the radiused end mill. Similarly, the honeycomb hatch shows the volume machined by the radiused end mill that cannot be machined by a ball nosed cutter. If the hatched area is larger than the honeycomb a ball nose cutter should be used and vice versa.

The objective is to maximize the material removal. A reference value, d base, is found

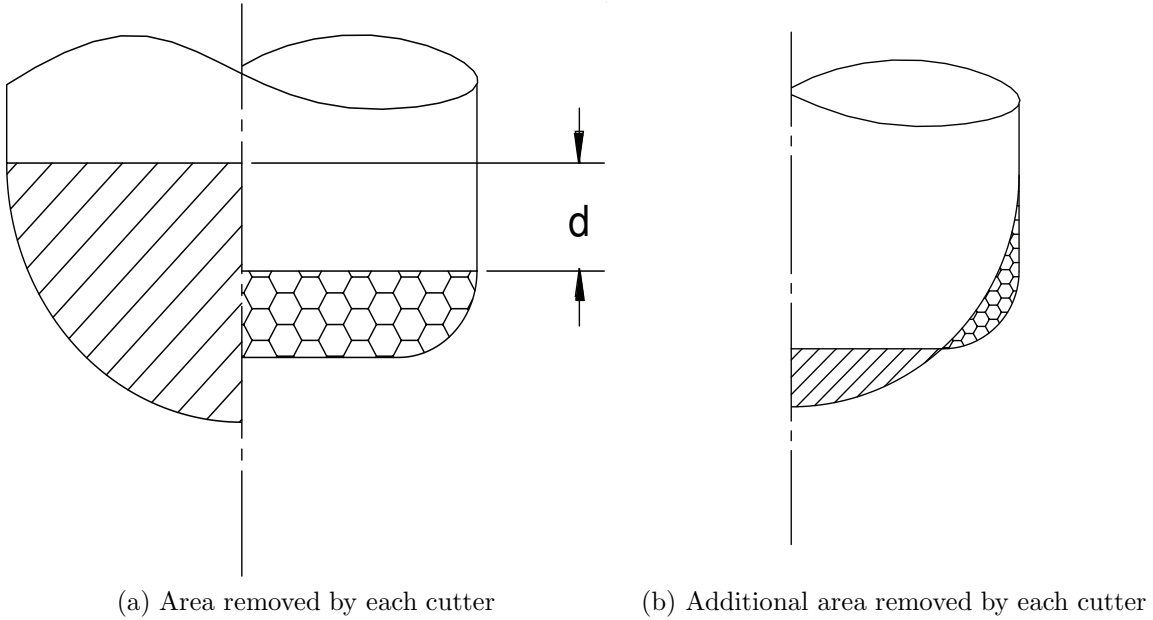


Figure 4.1: Comparison of area removal for ball nose and radiused cutter

at the point at which the area removal for both tools is the same, i.e., the hatched area equals the honeycomb hatched area. Based on the parameters shown in Figure 4.2 the removal area of the ball nose cutter and the radiused cutter is found. The area of the ball nose cutter is simply a quarter circle and the area of the radiused cutter is composed of two rectangular sections and a quarter circle.

$$\text{Area of ball nose cutter: } \frac{\pi}{4} \cdot (r_1 + r_2)^2 \quad (4.1)$$

$$\text{Area of toroidal cutter: } (r_1 \cdot r_2) + \pi \cdot \frac{r_2^2}{4} + d \cdot (r_1 + r_2) \quad (4.2)$$

Setting the areas equal to each other and solving for d yields:

$$d_{base} = \frac{\frac{\pi}{4} \cdot (r_1 + r_2)^2 - (r_1 \cdot r_2) - \pi \cdot \frac{r_2^2}{4}}{(r_1 + r_2)} \quad (4.3)$$

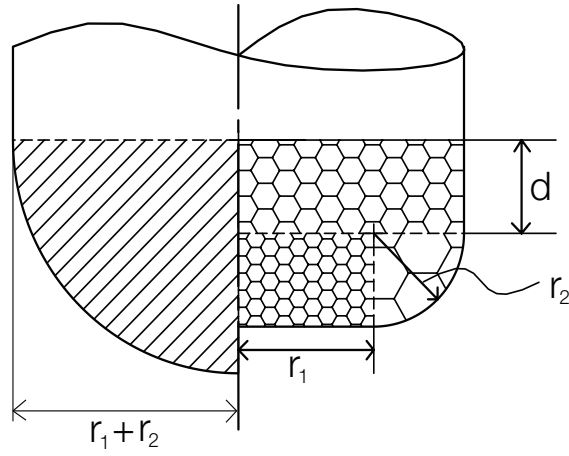


Figure 4.2: Parameters for calculating area removal

Based on this value it can be determined which tool removes more material. If the value of d at a cutter location is greater than the d base then the radiused cutter removes more material and is preferred. Otherwise, the ball nose cutter has greater material removal and is preferred.

d base was calculated based on a sectional view of the cutter locations. In reality the tools are 3D objects and volumetric material removal is a better indication of the quality of machining. The volume removal of a ball nose is half the volume of a sphere and is represented by:

$$\text{Volume removal of ball nose cutter: } \frac{2\pi}{3} \cdot (r_1 + r_2)^3 \quad (4.4)$$

To calculate the volume removal of a radiused cutter it was split in three sections as shown in Figure 4.3. The volume of the three sections is given by:

$$\text{Section 1: } \pi \cdot (r_1 + r_2) \cdot d \quad (4.5)$$

$$\text{Section 2: } \pi \cdot r_1^2 \cdot r_2 \quad (4.6)$$

$$\text{Section 3: } \frac{2\pi}{3} \cdot r_2^3 + \frac{\pi^2}{2} \cdot r_1 \cdot r_2^2 \quad (4.7)$$

$$\text{Volume of removal of radiused cutter: } \pi \cdot (r_1 + r_2) \cdot d + \pi \cdot r_1^2 \cdot r_2 + \frac{2\pi}{3} \cdot r_2^3 + \frac{\pi^2}{2} \cdot r_1 \cdot r_2^2 \quad (4.8)$$

Setting the volume removals for ball nose cutter and radiused cutter equal and solving for d yields:

$$d_{base} = \frac{\frac{2\pi}{3} \cdot (r_1 + r_2)^3 - \pi \cdot r_1^2 \cdot r_2 - \frac{2\pi}{3} \cdot r_2^3 + \frac{\pi^2}{2} \cdot r_1 \cdot r_2^2}{\pi \cdot (r_1 + r_2)} \quad (4.9)$$

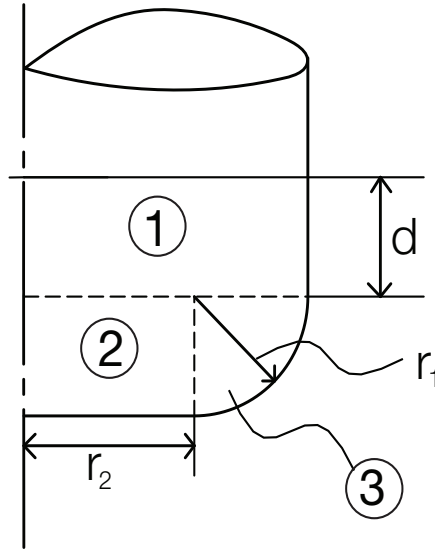


Figure 4.3: Volume removal of radiused cutter

As in the case of area removal if the value of d at a cutter location is greater than the d_{base} then the radiused cutter removes more material. Otherwise the ball nose cutter removes more material.

Volume removal is a good way to determine which tool is better at a given point;

however, tool paths require tools to move. Material left behind in one tool position can be removed at another tool position. For a global perspective a second factor should be used to evaluate the surface quality. This factor is the height of the scallops resulting from the machining process.

4.2 Scallop Height

The method presented here obtains minimal scallop height by selecting an appropriate tool. This section describes a method for calculating scallop height before actual machining takes place. This information is then used as a parameter in optimal tool selection.

Given that the feed direction of the tool path is along the x -axis the scallop heights are measured on the yz -plane as shown in Figure 4.4 . The ball nose cutter is modelled as a two dimensional circle with radius r . The inserts in the radiused end mill are also modelled as two dimensional circles with radius r_1 .

To determine scallop heights the method described by Patel *et al.* [35] is used. A tangent line connecting two successive tool positions along the yz -plane is found. By finding the maximum distance along the z -axis, between the intersection point and the tangent line the scallop height between two tool positions can be found. This procedure is then repeated for all other tool positions.

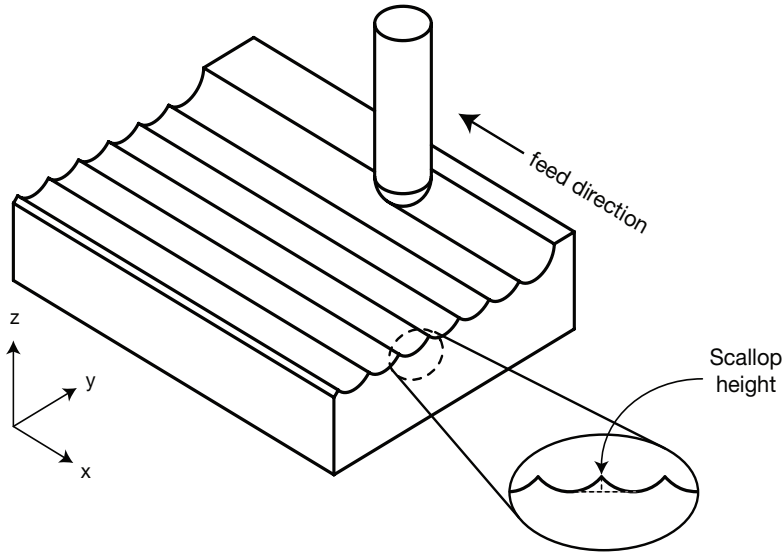


Figure 4.4: Orientation of scallop height calculation

4.3 Results

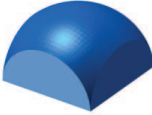
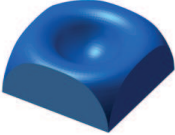
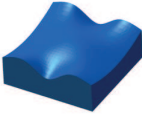
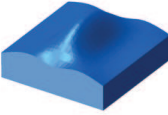
To evaluate the performance of each type of tool four surfaces (sphere, torus and two spline surfaces) were machined. The surfaces, shown in Table 4.1, were machined using both tools. Scallop height and volume removal are calculated for each of these surfaces using the procedures described in the previous section. Table 4.1 shows the mean, maximum and minimum scallop height resulting from the machining of the surfaces using both cutters. The distribution of scallop heights for both cutters is shown in Figure 4.5-Figure 4.8. Figure 4.5(a) and Figure 4.5(b) show the scallop heights for the ball nose and radiused cutter respectively. The volume removal comparison for each surface is shown in Figure 4.5(c). This is obtained by comparing the value of d (volume removal) for both tools at a tool position and selecting the one with the highest material removal. Values higher than d_{base} have higher material removal when machined with the radiused cutter and vice versa. Figure 4.5(d) shows the scallop heights resulting from machining the surface based

on the results of the volume comparison. Both cutters are used to machine the surface, for each cutter location the cutter with the highest volume removal is selected. The result is a machined part that will have the least material leftover. The tool movement is not considered. Finally, Figure 4.5(e) shows the surface being machined but in this case the tool that results in the smallest scallop height is selected. The result is a machined part with minimal scallop heights. Scallops depend on the direction of tool movement. This method assumes that the tool path footprint is specified by the user.

The results on Table 4.1 show that the ball nose cutter has a lower mean and maximum scallop height for all the surfaces. The radiused cutter has a minimum scallop height of $0.0mm$; this scallop height will occur whenever the normal of the surface being machined is aligned with the tool axis and the side step is less than the core radius (r_2) of the cutter. For example, in the case of the sphere the zero scallop height case occurs at the top the sphere where the triangles that are used to model the sphere have their normal vectors aligned with the tool axis. Figure 4.5 and Figure 4.6 show that the top of the spheres and torus are best machined with a radiused end mill whereas the side is more efficiently machined with a ball nosed tool. The transition occurs when the surface normal makes an angle of 35° with the tool axis. The example shows how the portions of the spherical and toroidal surface making an angle more than 35° with the tool axis are best machined with a ball nose cutter and similarly those portions that make an angle less than 35° are best machined with a radiused end mill.

The relationship among the angle between the tool axis and the surface normal is a common result in the machining of these four surfaces as can be seen in the figures. Since the spline surfaces shown in Figure 4.7 and 4.8 are mostly composed of areas where the angle between the surface normal and the tool axis is large the ball nose was determined

Table 4.1: Scallop heights of four surfaces machined using ball nose and radiused end mills.

Type of Surface		Ball Nose End Mill			Radiused End Mill		
		$r_1 = 1/16 \text{ in}$			$r_1 = 1/32 \text{ in}, r_2 = 1/32 \text{ in}$		
		Scallop Height [mm]					
		Max	Min	Mean	Max	Min	Mean
Sphere		0.2849	0.0138	0.0430	0.3945	0.0	0.0870
Torus		0.2941	0.0079	0.0415	0.4052	0.0	0.0838
Spline 1		0.0421	0.0073	0.0329	0.0884	0.0	0.0680
Spline 2		0.0538	0.0073	0.0340	0.1180	0.0	0.0669

to be better suited for these surfaces. It was found that as the angle increases scallop heights increase for both types of tools. For the volume removal calculation it was found that at 35° the ball nose tool starts removing more material than the radiused cutter. The transition in volume removal from a radiused cutter to a ball nose cutter can be explained by that fact that at 35° the removal volume on the radiused end mill is at most $\frac{1}{4}$ of a torus geometry defining the cutter, while the volume of removal for the ball nose end mill is half a sphere as was explained previously.

The variation in scallop height can be explained as follows. As the angle between the surface normal and the tool axis increases the motion along the *z-axis* also increases. In other words, when a surface is flat the side step only causes motion along the *y-axis*; however, when the angle increases the side step causes motion in the *y-axis* and the *z-axis*. This means that the distance between the two centres will be bigger. In the case of the ball nose end mill the effect is not that significant since the contact radius is large in comparison with the contact radius of the radiused end mill. Only the radius of insert is considered at this point since for any angle greater than 0° only the insert is in contact with work piece.

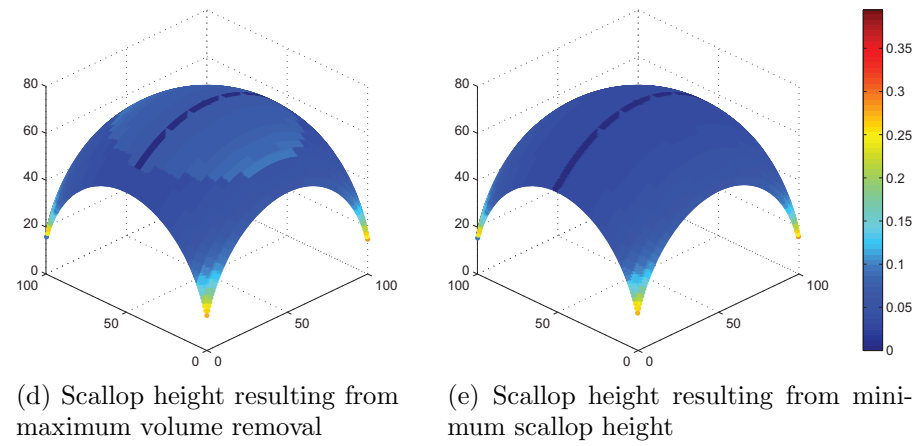
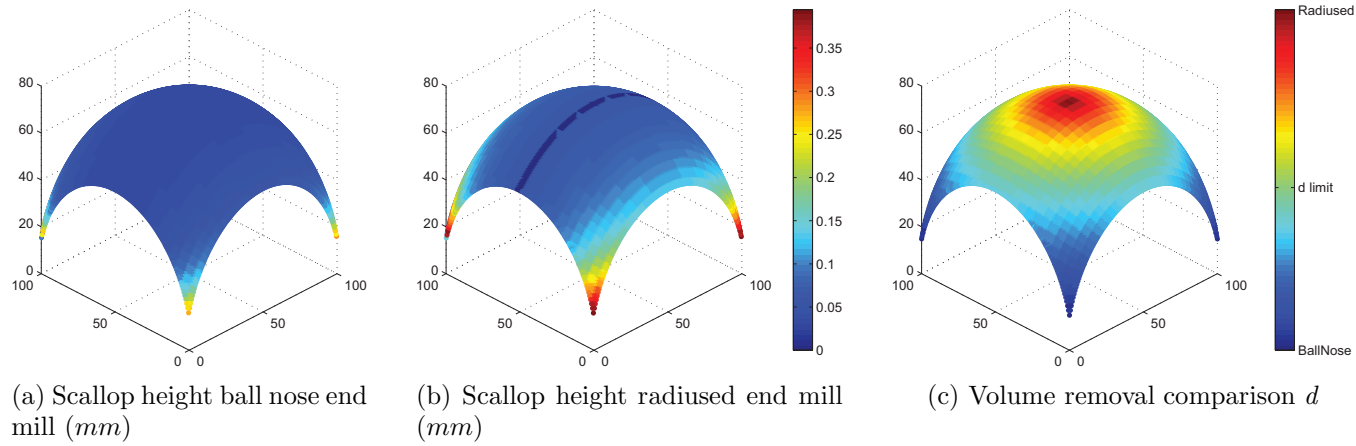


Figure 4.5: Scallop heights and volume removal for a spherical surface

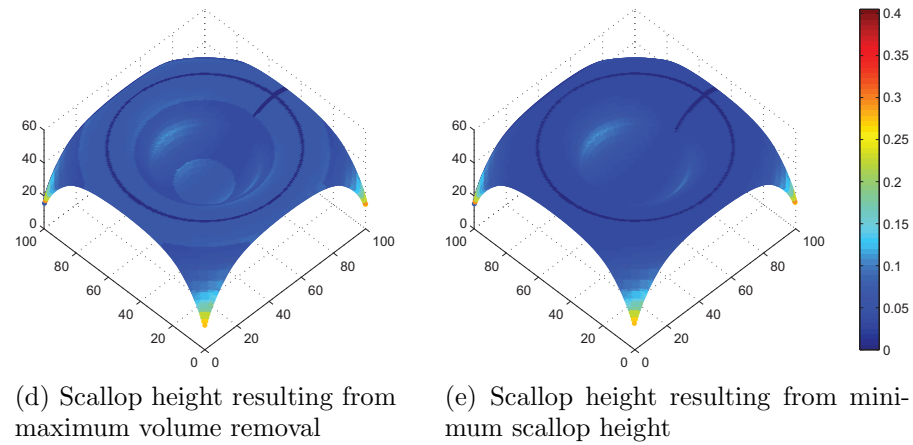
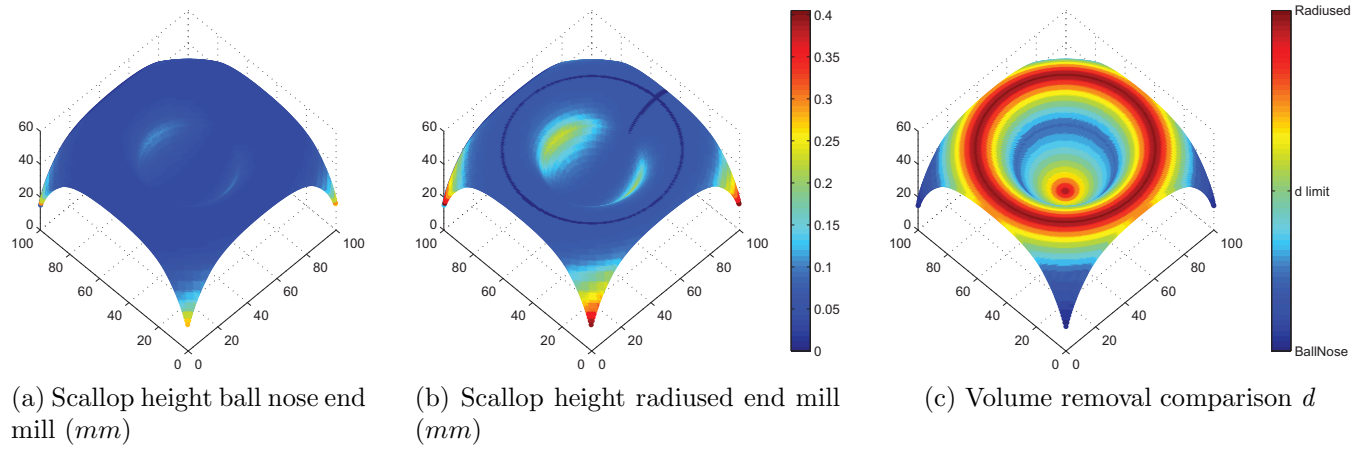


Figure 4.6: Scallop heights and volume removal for a toroidal surface

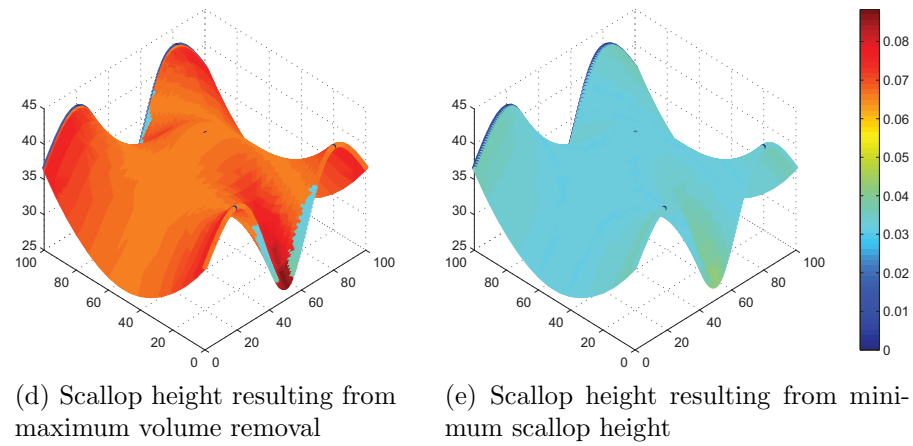
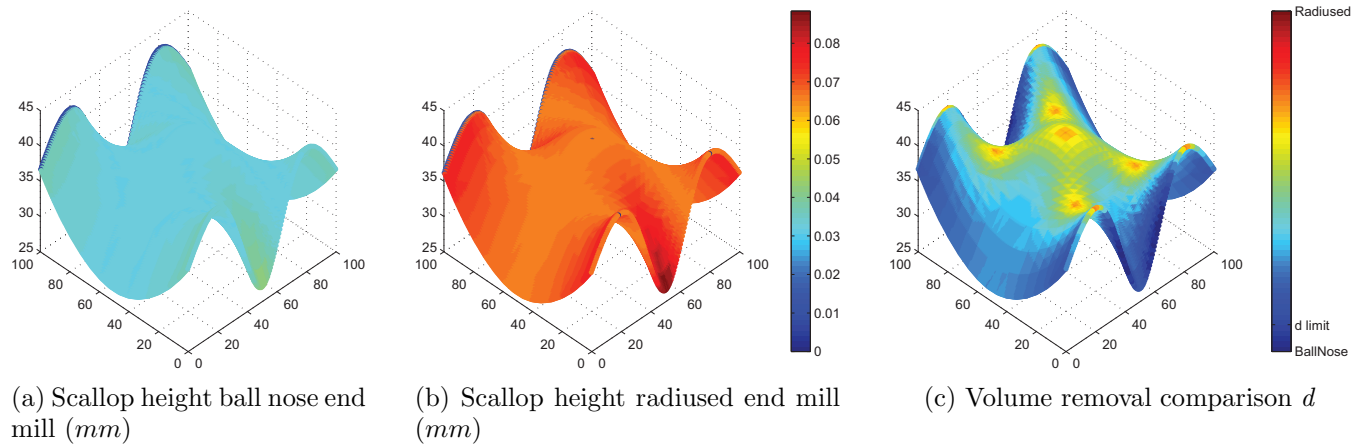


Figure 4.7: Scallop heights and volume removal for a spline surface

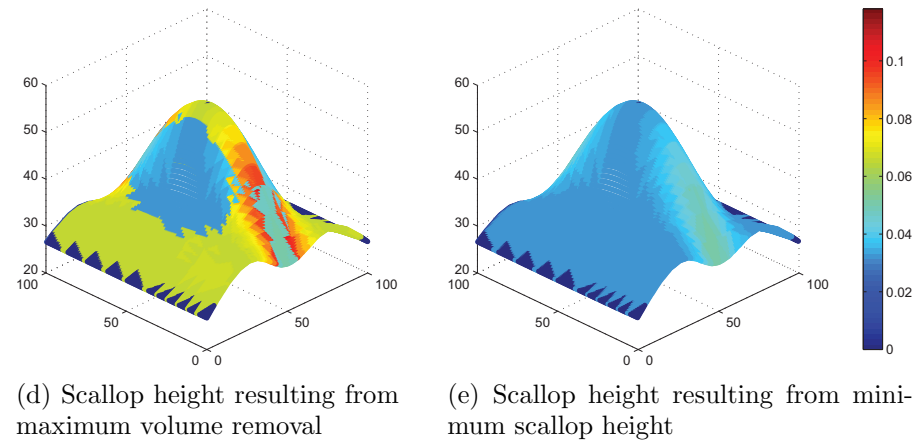
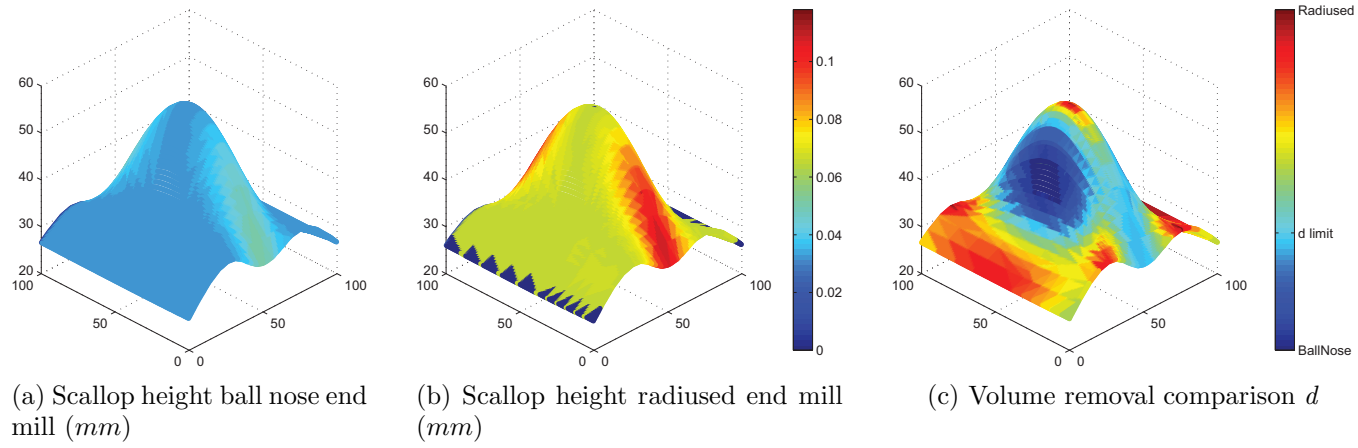


Figure 4.8: Scallop heights and volume removal for a spline surface

4.4 Confirmation Test

Tests were conducted on a sample part, shown in Table 4.2, to verify the conclusions drawn from the results. The sample part is comprised of both curved surfaces and flat surfaces. The objective of this test was to examine the performance of the described methodology in an actual customizable product. A similar analysis, as the one performed on the four sections, was carried out for the sample part. The results of the test are shown in Table 4.2 and Figure 4.9 shows the scallop height distributions.

The results from this experiment show that in average the scallop heights for the ball nose cutter are lower than the scallops for the radiused cutter. However, it was found that that based on material removal the radiused cutter was better fit for this surface. Based on what is desired the model can be machined using either one of the cutters or both them. If it is desired to have a good surface finish then a ball nose cutter should be used since it will result in smaller scallop heights. If a machining process that is as close as possible to the original model is desired then a radiused cutter should be used.

The results from this experiment as well as the results from the sample geometric surfaces prove that tool type selection is essential to guarantee an optimal machining process. Much research had been done in selecting an optimal cutter for a milling operation; however, as mentioned previously, these had failed to take into account surface geometry and mostly focused on cutter size and not cutter type. The methodology described in this chapter successfully compares a radiused and ball nose cutter based on scallop height and volume removal of each. An analysis of just the volume removal is not enough to analyze a machining process. Volume removal only determines the material removal at a certain cutter location but tools move during a machining process. It is necessary to study the

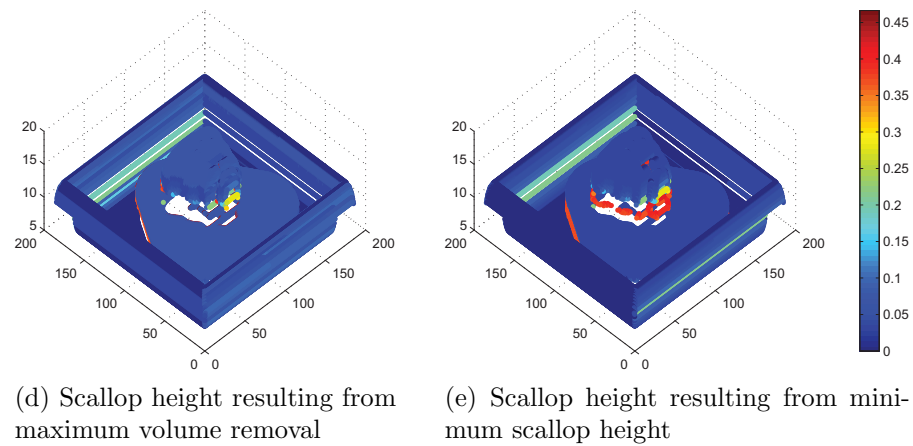
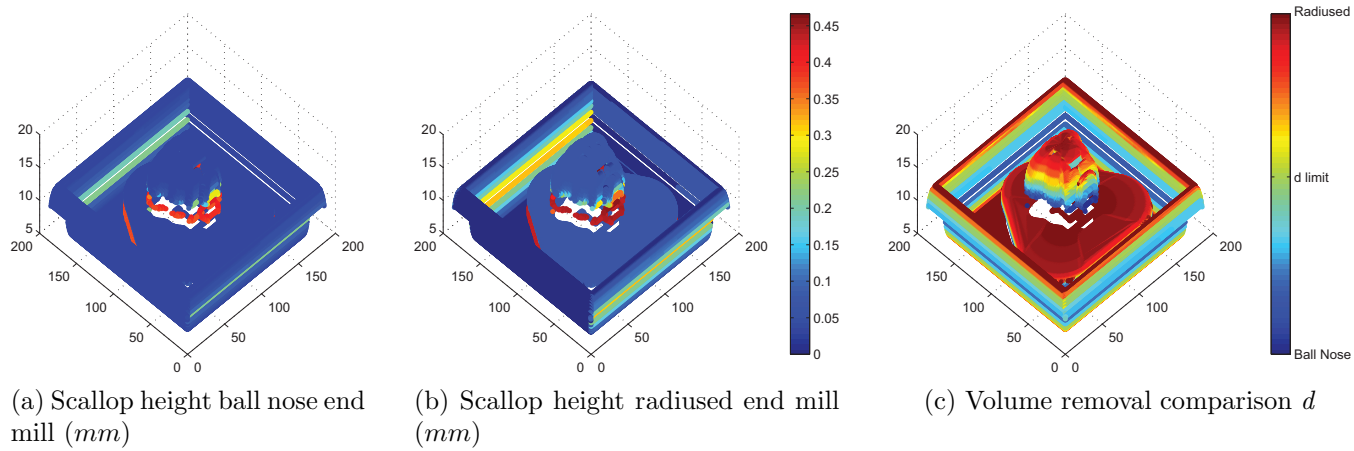


Figure 4.9: Scallop heights and volume removal for a custom plaque

Table 4.2: Scallop heights machined using ball nose and radiused end mills.

Ball Nose End Mill $r_1 = 1 \text{ mm}$			Radiused End Mill $r_1 = 0.5 \text{ mm}, r_2 = 0.5 \text{ mm}$		
Scallop Height [mm]					
Max	Min	Mean	Max	Min	Mean
0.4456	0.0313	0.0388	0.4664	0.0000	0.0500



effects caused by the movement of the tool. The focus of this work is on presenting a method for selecting a tool based on volume removal. More work is required to consider tool movement. A simple tool movement based tool selection has been presented in terms of the scallop height method. This method assumes a zig zag tool path and does not allow change of path after tool selection shows the different regions that should be machined with different tools.

The results show that material removal and scallop are not necessarily related. While a tool might be removing a large amount of material at a cutter location this does not

necessarily mean that the scallop height will be lower. Nonetheless, based on this method one can either have a surface that is machined with minimal scallop height or maximum material removal if only one type of tool is used.

This research also shows that for sculptured surfaces there is not a specific type of tool suited for machining the entire surface. For example, in the confirmation test that was presented in this chapter a radiused end mill can be used to machine the entire surface to ensure that the machined part will be as close as possible to the original model. A ball nose end mill can then be used to clean up the areas where the scallop heights are outside the tolerance. By using a combination of these tools an optimal machining process can be achieved.

Chapter 5

Conclusion

A successful implementation of an automated CAPP system for NC machining in industry requires a CAM system that can generate tool paths quickly and guarantee an optimal machining process. The research described in this thesis addresses two main components of CAM system: tool path generation and selection of tool type.

The new offset surface methodology described has improved on previous tool path planning techniques. As a result of this work tool paths are accurately generated for triangulated sculptured surfaces. Furthermore, knowing that more than one type of tool is required to accurately machine a surface, this offset surface technique was created for any cutter in the APT definition. Finally, the tool path generation process was optimized by eliminating unnecessary elements in the offset surface and by doing part of the calculations in the graphics card resulting in hardware acceleration. The simulations and machining test that were carried out on diverse sculptured surfaces prove that the tool paths generated by the offset surface method are correct. Proving that this methodology is adequate for machining sculptured surfaces.

The selection of an optimal tool type for machining a sculptured surface was achieved

by comparing the material removal of ball nose and radiused cutters. The results show the end result of a machining process is highly dependant on the tool type that is used. The material removal and the scallop heights are both affected by the type of tool used but are not necessarily related. Results also show that orientation of the surfaces affects the tool type that should be used. It was found that by using the two types of cutters the end result would be better, i.e., better surface finish and machined part being close to solid model representation.

5.1 Future Considerations

The results obtained from this research are only a small part of CAM system. Future research should include ways in which these methodologies can be integrated successfully into a CAM system. Research should focus on:

1. Integration of the offset surface methodology for tool path generation with a CAD system that generates solid models of customizable parts.
2. For the offset surface, only part of the calculations are carried out in the graphics card. If additional calculations are transferred to the graphics processing unit the speed at which tool paths are generated would be increased due to hardware accelerated processing.
3. Research on ways to avoid the small amount of intersecting triangular facets should be undertaken.
4. The offset surface method and the tool selection algorithms should be integrated. In

such a way that by simply specifying tools available and desired surface finish the optimal tool type and tool path foot print can be generated.

Appendix A STL File Format

In order to generate an STL file an object is first designed by a solid modeller. A tessellation algorithm is then applied creating a boundary representation that covers the surface of the solid with a mesh. [43] This mesh is made up by connected “three-dimensional” triangles as shown in Figure 1. Although a triangle is a two dimensional object, the “three-dimensional” terminology applies to the X , Y , and Z coordinates of three ordered endpoints of the triangle’s edges. The three endpoints along with an outward normal are used to define each triangle. These triangular meshes are stored in STL format and are used as definitions of geometry of real solids in many industrial applications. An STL file is

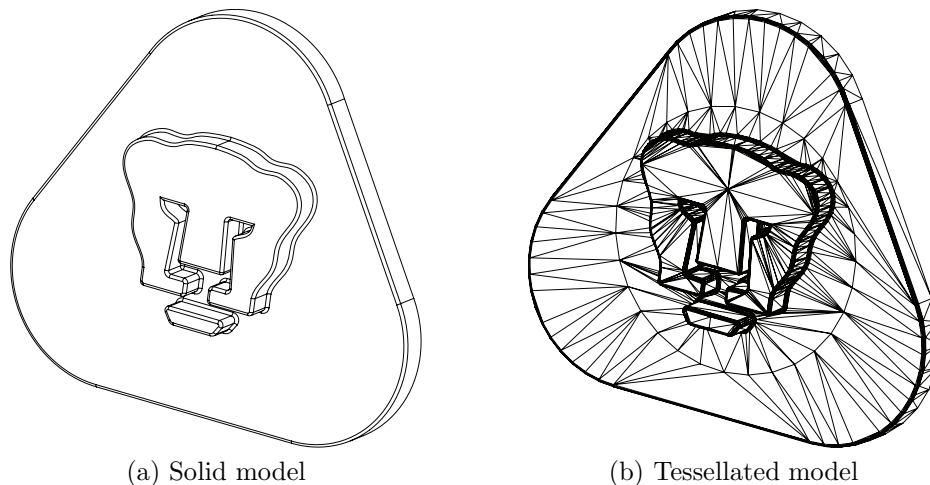


Figure 1: Conversion of solid model to tessellated model

list of facet data: a normal (n_x, n_y, n_z) and each vertex of the triangular facet are specified by three coordinates each (p_{kx}, p_{ky}, p_{kz}) ; therefore, there is a total of 12 numbers stored for each facet.

$$\begin{array}{ccc} n_x & n_y & n_z \\ p_{1x} & p_{1y} & p_{1z} \\ p_{2x} & p_{2y} & p_{2z} \\ p_{3x} & p_{3y} & p_{3z} \end{array}$$

The facets define the surface of a 3-dimensional object. As such, each facet is part of the boundary between the interior and the exterior of the object. The orientation of the facets (which way is “out” and which way is “in”) is specified in the following two ways which must be consistent. First, the direction of the normal is outward. Second, the vertices are listed in counter-clockwise order when looking at the object from the outside (right-hand rule). These rules are illustrated in Figure 2.

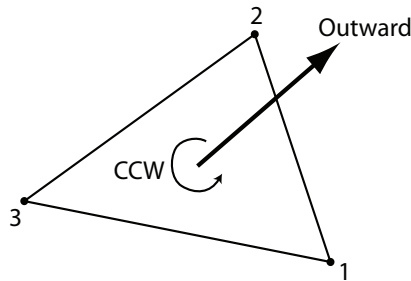


Figure 2: Orientation of facet determined by unit normal and order in which vertices are listed.

Triangles in an STL file must all mate with other triangles at the vertices; this is known as the “vertex to vertex” rule [16]. In other words, a vertex of one triangle cannot lie on the side of another. This is illustrated in Figure 3. The ASCII format is primarily intended for

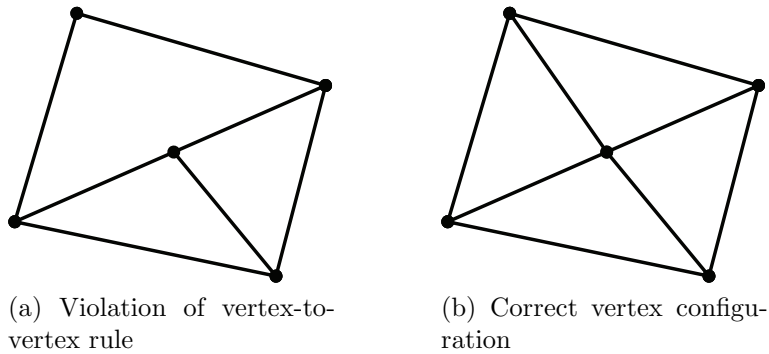


Figure 3: The vertex-to-vertex rule. The left figure shows a violation of the rule. A correct configuration is shown on the right

testing CAD interfaces given its simplicity. The syntax for an ASCII STL file is as follows:

<pre> solid <i>name</i> facet normal n_x n_y n_z outer loop vertex p_{1x} p_{1y} p_{1z} vertex p_{2x} p_{2y} p_{2z} vertex p_{3x} p_{3y} p_{3z} endloop endfacet endsolid <i>name</i> </pre>	<pre> solid pumas facet normal 0.000000e+000 0.000000e+000 1.000000e+000 outer loop vertex 1.605438e+002 8.668300e+001 4.000000e+001 vertex 1.608682e+002 8.646623e+001 4.000000e+001 vertex 1.612509e+002 8.639011e+001 4.000000e+001 endloop endfacet endsolid pumas </pre>
---	--

The STL file does not contain any topological information such as links, pointers to another element, or proximity. Each vertex is written by its coordinates in the file as many times as it occurs in the mesh. For a triangular mesh to be considered correct it must meet the following criteria:

1. Each edge is shared at most by two triangles.
2. A vertex can be shared by any number of triangles.
3. No triangle has intersection with the interior of any other triangle.

Appendix B APT Cutter Geometry

An appropriate cutter description is required so that proper tool path coordinates may be computed. Most of the cutter geometry is usually defined by the automatically programmed tools (APT) cutter definition for CNC machining. The cutter defined by the APT consists of a lower cone (defined by a lower line segment), the outer surface of a torus, and an upper cone (defined by an upper line segment). The generalized cutter geometry can be described fully by the following parameters [28]:

d The cutter diameter, which is twice the radial distance from the tool axis to the intersection of the lower and upper line segments.

r The radius of the corner circle.

e The radial distance from the tool axis to the center of the corner circle.

f The distance from the tool endpoint to the center of the corner circle measured parallel with the tool axis.

α The angle from a radial line through the tool endpoint to the lower line segment.

β The angle between the upper segment and the tool axis.

h The cutter height measured from the tool endpoint along the tool axis.

The geometric interpretation of all parameters is given in Figure 4 , where the bold outline represents a cross section of the cutter.

The cutter parameter values must be consistent among themselves and not violate

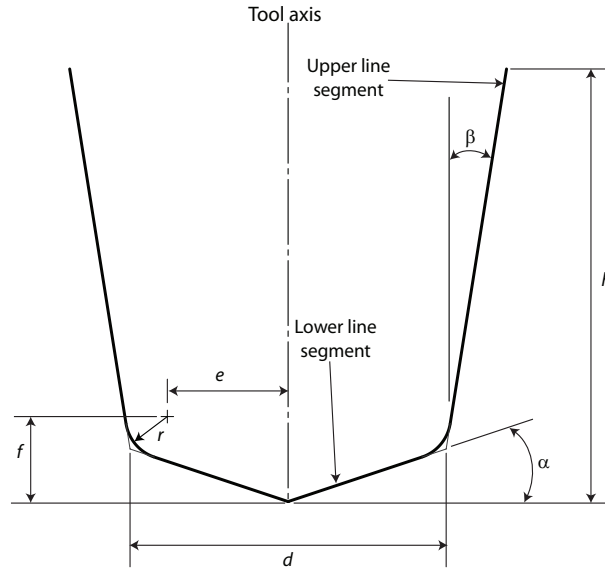


Figure 4: Parameters for generalized APT cutter geometry

certain restrictions so that permissible geometries are properly described [28]. This work will focus on three common configurations of the APT cutter geometry: ball nose, radiused, and flat end mill. These cutters were selected as they are the ones most commonly used in industry and are most commercially available. The cutter shape geometric definition for these three cutters is shown in Figure 5 below.

The reference point of each of the cutters is marked by the red cross in the figures. This reference point is used to place the cutter at the CL. The reference point of the flat end mill is located in the center point of the bottom of the cutter, for the ball nose end mill it is located in the center of the end sphere, and for the radiused end mill the reference point is located in the center point of the torus.

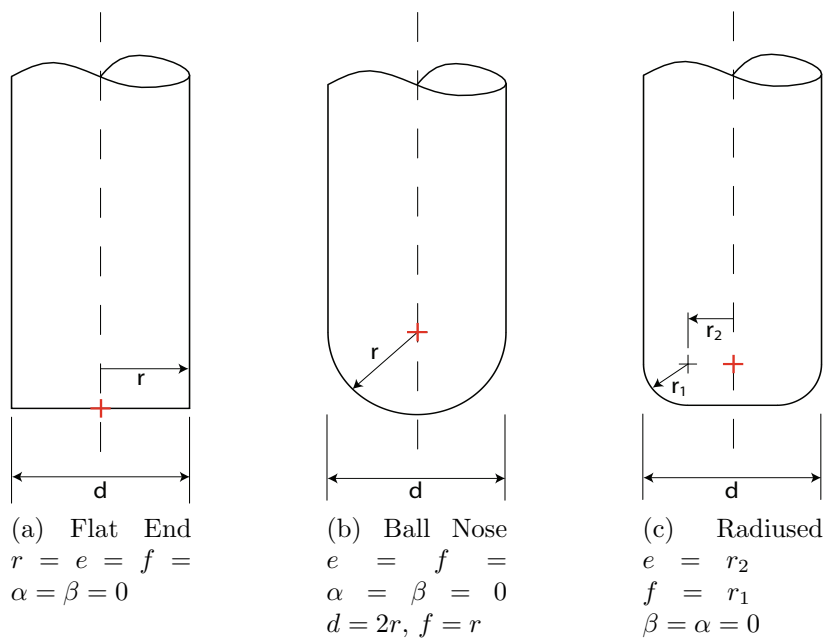


Figure 5: Selection of cutter shapes based on APT definition

References

- [1] S. H. Ahn, V. Sundararajan, C. Smith, B. Kannan, R. D'Souza, G. Sun, A. Mohole, P. K. Wright, J. Kim, S. McMains, J. Smith, and C. H. Squin. Cybercut: An internet-based cad/cam system. *Journal of Computing and Information Science in Engineering*, 1(1), 201. 3
- [2] ArtCAM. Delcam plc. Accessed June 18th, 2009. 3
- [3] M. Bala and T. Chang. Automatic cutter selection and optimal cutter path generation for prismatic parts. *International Journal of Production Research*, 29(11):2163–2176, 1991. 15
- [4] S. Bedi, F. Ismail, M. J. Mahjoob, and Y. Chen. Toroidal versus ball nose and flat bottom end mills. *The International Journal of Advanced Manufacturing Technology*, 13(5):326–332, 1997. 17
- [5] D. K. BK Choi and R. Jerard. C-space approach to tool-path generation for die and mould machining. *Computer-Aided Design*, 29(9):657–669, 1997. 48
- [6] T. Chen and P. Ye. A tool path generation strategy for sculptured surfaces machining. *Journal of Materials Processing Technology*, 127:369–373, 2002. 4
- [7] Y.-H. Chen, Y.-S. Lee, and S.-C. Fang. Optimal cutter selection and machining plane determination for process planning and nc machining of complex surfaces. *Journal of Manufacturing Systems*, 17(5):371–388, 1998. 16
- [8] B. Choi, Y. Chang, J. Park, and D. Kim. Unified cam-system architecture for die and mould manufacturing. *Computer-Aided Design*, 26:235–243, 1994.
- [9] K. A. Crow. Computer-aided process planning. Accessed March 18th, 2010. 1
- [10] F. I. David Roth and S. Bedi. Mechanistic modelling of the milling process using an adaptive depth buffer. *Computer-Aided Design*, 35:1287–1303, 2003. 48

- [11] H.-Y. Feng and H. Li. Constant scallop-height tool path generation for three-axis machining sculptured surface machining. *Computer Aided Design*, 34:647–654, 2001. 59
- [12] G. Glaeser, J. Wallner, and H. Pottmann. Collision-free 3-axis milling and selection of cutting tools. *Computer-Aided Design*, 31(3):225–232, 1999. 16
- [13] B. P. G.V.V.R Kumar, K.G. Shastry. Computing non-self intersecting offsets of nurbs surfaces. *Computer Aided Design*, 36(3), 2002. 21
- [14] T. V. Hook. Real-time shaded in nc milling display. *ACM SIGGRAPH Computer Graphics*, 20(4):15–20, 1986. 48
- [15] G. Israeli. Software simulation of numerically controlled machining. Master’s thesis, University of Waterloo, 2006. 52
- [16] P. F. Jacobs. *Rapid Prototyping and Manufacturing*. McGraw Hill, New York, New York, 1992. 81
- [17] C. Jensen, W. Red, and J. Pi. Tool selection for five-axis curvature matched machining. *Computer-Aided Design*, 34(3):251–266, 2002. 17
- [18] F. Ju and G. Barrow. A technologically oriented approach for the economic tool selection and tool balancing of milled components. *International Journal of Advanced Manufacturing Technology*, 14(5):307–320, 1998. 1
- [19] C.-S. Jun, D.-S. Kim, and S. Park. A new curve-based approach to polyhedral machining. *Computer-Aided Design*, 34(5):379–389, 2002. 14, 23
- [20] W. Jung, H. Shin, and B. K. Choi. Self-intersection removal in triangular mesh offsetting. *Computer Aided Design and Applications*, 1(1-4):477–484, 2004. 13
- [21] T. Kai, C. Cheng, and Y. Dayan. Offsetting surfaces boundaries and 3-axis gouge-free surface machining. *Computer Aided Design*, 27(12), 1995. 21
- [22] Y. D. Kai Tang, Charles C Cheng. Offsetting surface boundaries and 3-axis gouge-free surface machining. *Computer Aided Design*, 27(12), 1995.
- [23] T. Kandala. Implementation of user-defined features in web-based CAD applications. Master’s thesis, University of Waterloo, 2009. 2, 3, 4
- [24] K. Kim and K. Kim. A new machine strategy for sculptured surfaces using offset surface. *International Journal of Production Research*, 33(6), 1995. 22

- [25] S.-J. Kim, D.-Y. Lee, and M.-Y. Yang. Offset triangular mesh using the multiple normal vectors of a vertex. *Computer Aided Design and Applications*, 1(1-4):285–291, 2004. 12, 21, 24, 32
- [26] S.-J. Kim and M.-Y. Yang. Triangular mesh offset for generalized cutter. *Computer-Aided Design*, 37(10):999–1014, 2005. 14, 23, 32
- [27] B. Koc and Y.-S. Lee. Non-uniform offsetting and hollowing objects by using biarcs fitting for rapid prototyping processes. *Computers in Industry*, 47(1):1–23, 2002. 13
- [28] I. H. Kral. *Numerical Control Programming in APT*. Prentice-Hall, Englewood Cliffs, New Jersey, 1986. 83, 84
- [29] K.Suresh and D. Yang. Constant scallop-height for machining of free-form surfaces. *Journal of Engineering for Industry*, 116:253–259, 1994.
- [30] Y. Lee and T. Chang. Automatic cutter selection for 5-axis sculptured surface machining. *International Journal of Production Research*, 34(4):977–998, 1996. 17
- [31] Y. Lee, B. Choi, and T. Chang. Cut distribution and cutter selection for sculptured surface cavity machining. *International Journal of Production Research*, 30(6):1447–1470, 1992. 16
- [32] T. Lim, J. Ritchie, and D. Clark. Optimizing tool selection. *International Journal of Production Research*, 39(6):1239–1256, 2001. 15
- [33] N. P. Manos, S. Bedi, D. Miller, and S. Mann. Single controlled axis lathe mill. *International Journal of Advanced Manufacturing Technology*, 32(1-2):55–65, 2007. 6, 11, 19, 21, 57
- [34] S. C. Park. Sculptured surface machining using triangular mesh slicing. *Computer Aided Design*, 36(3):279–288, 2004. 6, 12
- [35] K. Patel, G. Salas Bolanos, R. Bassi, and S. Bedi. Optimal tool selection. 2010. 19, 64
- [36] X. Qu and B. Stucker. A 3d surface offset method for stl-format models. *Rapid Prototyping Journal*, 9(3):133–141, 2003. 12, 24, 32
- [37] S. J. Rock and M. J. Wozny. Generating topological information from a “bucket of facets”. In *Solid Freeform Fabrication Symposium Proceedings*, pages 251–259, Austin, Texas, United States America, 1992. 11, 47
- [38] G. Salas Bolanos, S. Mann, and S. Bedi. Targeted webcad. *Computer Aide Design and Applications*, 6(5), 2009. 3

- [39] W. J. Schroeder, J. A. Zarge, and W. E. Lorensen. Decimation of triangle meshes. *Computer Graphics Association for Computing Machinery*, 26(22):65–70, 1992. 13
- [40] D. Shreiner. *OpenGL Programming Guide*. Addison Wesley, United States of America, 2009. 48
- [41] C. S. Smith and P. K. Wright. Cybercut: A world wide web based design-to-fabrication tool. *Journal of Manufacturing Systems*, 15(6), 1996. 3
- [42] G. I. X. Z. Stephen Mann, Sanjeev Bedi. Machine models and tool motions for simulating five-axis machining. *Computer-Aided Design*, 42:231–237, 2010. 52
- [43] M. Szilvasi-Nagi. Analysis of stl files. *Mathematical and Computer Modelling*, 38(7-9):945–960, 2003. 80
- [44] G. Turk. Re-tiling polygonal surfaces. *ACM SIGGRAPH Computer Graphics*, 26(2):55–64, 1992. 13
- [45] G. Vickers and K. Quan. Ball-mills versus end-mills for curved surface machining. *ASME Journal of Eng. for Industry*, 111:22–26, 1989. 16, 59
- [46] K. K. C. Y. M. Kyoung and and C. S. Jun. Optimal tool selection for pocket machining in process planning. *Computers and Industrial Engineering*, 33(3-4):505–508, 1997. 15
- [47] Z. Yao, S. K. Gupta, and D. S. Nau. A geometric algorithm for selecting optimal set of cutters for multi-part milling. In *Proceedings of the sixth ACM symposium on Solid modeling and applications*, pages 130–139, Ann Arbor, Michigan, United States, 2001. 15
- [48] Z. Yao, S. K. Gupta, and D. S. Nau. Algorithms for selecting cutters in multi-part milling problems. *Computer-Aided Design*, 35(9):825–839, 2003. 15
- [49] H. Yau, C. Chuang, and Y. Lee. Numerical control machining of triangulated sculptured surfaces in a stereo lithography format with a generalized cutter. *International Journal of Production Research*, 42(13):2573–2598, 2004. 5, 6, 11, 19
- [50] I. L. Yi, Y.-S. Lee, and H. Shin. Mitered offset of a mesh using qem and vertex split. In *Proceedings of the 2008 ACM symposium on Solid and physical modeling*, pages 315–320, Stony Brook, New York, 2008. 13, 24
- [51] I. Zeid. *Mastering CAD/CAM*. McGraw Hill, New York, New York, 2005. 1