

2009

Geometric process planning in rough machining

Joseph Edward Petrzelka
Iowa State University

Follow this and additional works at: <https://lib.dr.iastate.edu/etd>

 Part of the [Industrial Engineering Commons](#)

Recommended Citation

Petrzelka, Joseph Edward, "Geometric process planning in rough machining" (2009). *Graduate Theses and Dissertations*. 10787.
<https://lib.dr.iastate.edu/etd/10787>

This Thesis is brought to you for free and open access by the Iowa State University Capstones, Theses and Dissertations at Iowa State University Digital Repository. It has been accepted for inclusion in Graduate Theses and Dissertations by an authorized administrator of Iowa State University Digital Repository. For more information, please contact digirep@iastate.edu.

Geometric process planning in rough machining

by

Joseph E. Petrzelka

A thesis submitted to the graduate faculty
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Major: Industrial Engineering

Program of Study Committee:
Matthew Frank, Major Professor
Frank Peters
Eliot Winer

Iowa State University
Ames, Iowa

2009

Copyright © Joseph E. Petrzelka, 2009. All rights reserved.

Table of Contents

List of Figures	iv
List of Tables.....	vi
Abstract	vii
Chapter 1: General Introduction.....	1
Introduction.....	1
Milling Stages.....	1
Milling Machine Configurations	2
Three-axis Rough Machining	4
Multi-setup Rough Machining	6
Multi-axis Rough Machining	6
Problems in Multi-setup and Multi-axis Rough Machining.....	7
Thesis Organization	10
Literature Review	11
Three-axis Roughing	11
Multi-setup / Multi-axis Roughing.....	12
Summary of Literature	14
References.....	15
Chapter 2: A Remaining Stock Algorithm for Multiple Setup Subtractive Processes	18
Abstract.....	18
1 Introduction.....	18
Background and Motivation	19
2 Related Work	23
Vector Based:	24
Spatial Discretization:	24
Solid / Parametric Surface Based:	25
Integrating Simulation and Process Planning:.....	26
3 Overview of Methodology	26
Slice Approximation.....	28
Slice Shadowing	29

Polyhedral Reconstruction.....	29
4 Detailed Presentation of Methodology	31
4.1 Slice Approximation.....	31
4.2 Slice Shadowing	33
4.3 Polyhedral Reconstruction.....	37
5 Implementation	49
6 Limitations and Future Work.....	52
7 Conclusions.....	54
8 Acknowledgment	55
9 References.....	55
Chapter 3: General Conclusion	57
Review of Contribution	57
Future Research Directions.....	58
Application of Accessibility	58
Set-cover Analysis for Roughing Volume.....	59
Thin Web Detection	59
References.....	59

List of Figures

Figure 1. (a) Flat end mill and (b) ball end mill, showing different swept volume profiles.....	1
Figure 2. Component with undercut geometry that cannot be reached with a single setup in a three-axis mill	3
Figure 3. Common configurations of (a) three-axis, (b) four-axis, and (c) five-axis vertical milling machines.....	4
Figure 4. (a) Pocket milling and (b) island milling, both accessible using three-axis mills.....	5
Figure 5. Centrifugal impeller design, common in five-axis machining research [21]	7
Figure 6. Overlapping swept volumes from different approach orientations lead to tool path inefficiency	8
Figure 7. (a) Beginning stock and (b) desired component; (c) and (d) illustrate the thin web that emerges when cutting from opposing orientations, as in this case of 0-180 degree orientations.....	9
Figure 8. Thin string emerges (c) when cutting orientations are not directly opposing, as in this case of 0-120-240 degree orientations.....	9
Figure 9. Multiple setups required (even in multi-axis milling) to create component via a subtractive process; (a) shows initial stock geometry and (b-d) show the results after iterative cutting operations until the component fully emerges (d).....	19
Figure 10. (a) CNC-RP Implementation; (b) Process sequence of steps (b.1-b.4) to expose component geometry and (b.5-b.6) to expose sacrificial supports.....	21
Figure 11. (a) Naïve tool path volume sweep; (b) Volume sweep of tool path with knowledge of prior operations	22
Figure 12. Method for remaining stock computation is composed of three distinct tasks, each accomplished through new algorithms	27
Figure 13. An example component (a) and reconstruction of approximating slices (b) showing elimination of inaccessible holes and slots; lines represents axis of rotation.....	29
Figure 14. Correspondence types between (A) 1:1, (B) 2:1, and (C) 2:2 contour sets.....	30
Figure 15. Each slice (a) is represented as its own convex hull polygon (b)	32
Figure 16. Each slice (b) is also redefined as the union of all slices within one tool radius (a); for clarity, the set of slices in (a) are those surrounding the bold slice in (b)	32

- Figure 17. \mathbb{A}_j (d) is the set of slices $A_{i,j}$ (c) representing a visibility shadow of C_i at angle α_j (a) trimmed to stock cross-section S_i (b)..... 35
- Figure 18. \mathbb{B}_j (d) is the set of slices $B_{i,j}$ (c) composed of stock material unreachable from angle α_j to depth δ_j , computed by trimming the unreachable ‘halfspace’ (a) to the stock cross-section (b) 36
- Figure 19. Single-cut effect is represented by slice set \mathbb{R}_j (d), where each $R_{i,j}$ (c) is computed by joining shadow $A_{i,j}$ (a) and stock material $B_{i,j}$ (b)..... 36
- Figure 20. Each $\text{ReSt}_{i,j}$ (c) in \mathbb{ReSt}_{j+1} (d) is computed as the cumulative intersection of preceding single-cut effects $R_{i,j}$ (a,b), allowing independent computation of $R_{i,j}$; it (d) is also the final set of slices that is reconstructed to form a polyhedral model 37
- Figure 21. Illustration of shadow intersection: (a) where $\alpha_{x+1} = (\alpha_{x+2} + \pi)$ and the intersection is null, and (b) where the intersection is adjacent to C_i 41
- Figure 22. (a) $A_{i,x+1}$ and $B_{i,x+2}$ share (bold) boundary edges from S_i , (b) only two edges (neither adjacent to the other) of $A_{i,x+1}$ are not in either C_i or $B_{i,x+1}$ 41
- Figure 23. Method of augmenting the contour set to facilitate branching (A and B form an augmented contour, C , by joining their nearest points); actual tiling results are shown with augmented contour C 43
- Figure 24. (a) Method of determining normal vector for landmark points, (b) landmark normal vectors, with real and imaginary multipliers, for an arbitrary polygon (note that without the multipliers, matching vectors u and v via dot product could provide incorrect results)..... 46
- Figure 25. Typical tiling results using convex and concave landmark point matching (note good results along both convex and concave regions) 48
- Figure 26. Illustrations of remaining stock for the example component from seven setup orientations: (a) beginning stock, (b-e) setups 4-7, respectively (setups 1-3 omitted)..... 50

List of Tables

Table 1. Implementation results (coupling with multi-axis tool path generation) for two different geometries	51
--	----

Abstract

This thesis examines geometric process planning in four-axis rough machining. A review of existing literature provides a foundation for process planning in machining; efficiency (tool path length) is identified as a primary concern. *Emergent structures* (thin webs and strings) are proposed as a new metric of process robustness. Previous research efforts are contrasted to establish motivation for improvements in these areas in four-axis rough machining.

The original research is presented as a journal article. This research develops a new methodology for quickly estimating the remaining stock during a plurality of $2\frac{1}{2}$ D cuts defined by their depth and orientation relative to a rotary fourth axis. Unlike existing tool path simulators, this method can be performed independently of (and thus prior to) tool path generation. The algorithms presented use polyhedral mesh surface input to create and analyze polygonal slices, which are again reconstructed into polyhedral surfaces. At the slice level, nearly all operations are Boolean in nature, allowing simple implementation. A novel heuristic for polyhedral reconstruction for this application is presented. Results are shown for sample components, showing a significant reduction in overall rough machining tool path length.

The discussion of future work provides a brief discussion of how this new methodology can be applied to detecting thin webs and strings prior to tool path planning or machining.

The methodology presented in this work provides a novel method of calculating remaining stock such that it can be performed during process planning, prior to committing to tool path generation.

Chapter 1: General Introduction

Introduction

Machining is a common and economical means of producing accurate components with good surface finish. In this material removal process, the relative motion between a cutting tool and work piece shears away stock material through chip formation. One common method of machining is end milling, where a cylindrical tool with several flutes (cutting surfaces) spins rapidly and is directed through the work piece. Two of the most common tools are flat end mills and ball end mills, each affecting the profile of material removal differently (Figure 1).

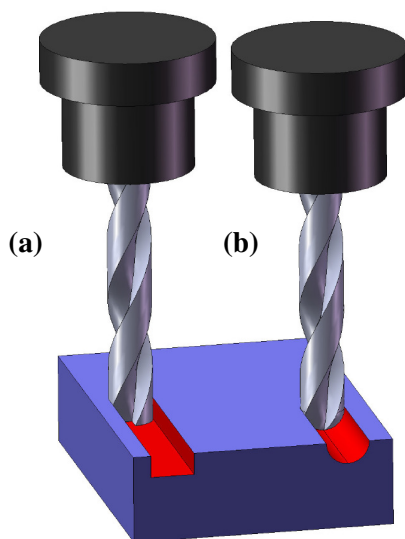


Figure 1. (a) Flat end mill and (b) ball end mill, showing different swept volume profiles

Milling Stages

In most milling applications, two distinct stages of processing occur: rough and finish machining.

Rough machining is the process of removing gross amounts of material from a piece of billet or raw

casting to form the general shape of a final component. This allows for a subsequent finishing operation to remove a very thin layer of remaining material, creating the appropriate surface finish and dimensional characteristics. In essence, the roughing stage focuses on removing a volume of material, while the finishing stage focuses on covering a surface area.

The specific roughing strategy employed is critical to the efficiency of the machining operation. According to a variety of sources, the roughing stage accounts for 50% to 90% of total machining time [8, 11] and is the most important process affecting machining time and product accuracy [5, 21]. Further, if the roughing process planning is not robust, the finishing process could encounter much more material than intended, resulting in excessive cutting forces and potentially catastrophic failure.

Milling Machine Configurations

While there are a variety of milling machine configurations available, this thesis focuses on the more common vertical milling machine, where the rotating tool is held in a vertical position. A basic computer numerical control (CNC) mill has three linear axes: x , y , and z (Figure 3a). These axes can be simultaneously controlled to move the tool along complex paths relative to the component surface. However, the *accessibility* of the tool is limited; undercut geometry is impossible to create without manually repositioning the component in the mill (Figure 2). This re-fixturing step is undesirable because it requires both additional time and introduces inaccuracies due to compounding setup errors.

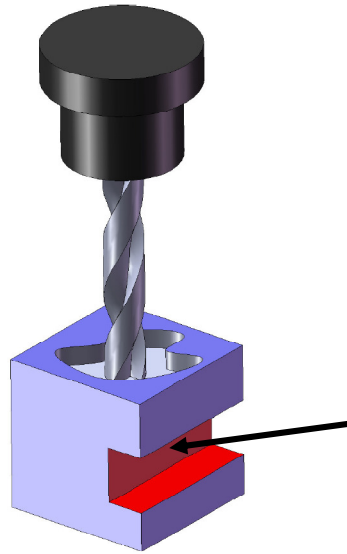


Figure 2. Component with undercut geometry that cannot be reached with a single setup in a three-axis mill

Fitting a standard three-axis CNC machine with additional rotary axes alleviates, to a certain extent, the problems associated with complex geometry. Work pieces can be automatically rotated to new cutting positions with little additional time and virtually no loss of accuracy. Alternatively, the rotary axes can be controlled simultaneously with the three linear axes to create even more complex tool motions. That is, in addition to controlling tool position, the tool posture (axial vector relative to the work piece) can be controlled with rotational degrees of freedom. Common four- and five- axis mill configurations are illustrated in Figure 3.

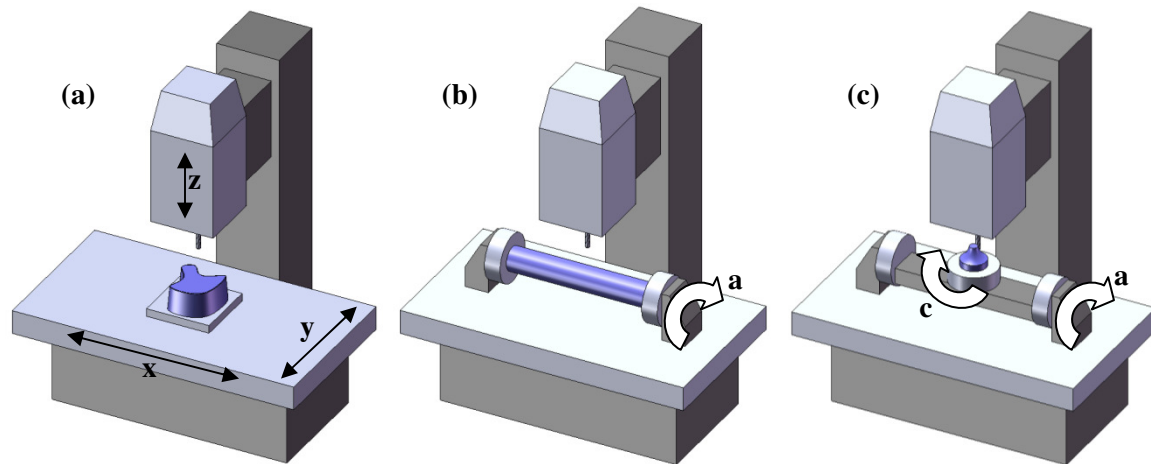


Figure 3. Common configurations of (a) three-axis, (b) four-axis, and (c) five-axis vertical milling machines

It is important to note that even with four- and five- axis mills, accessibility is still constrained due to the physical location of the rotary axes. While three-axis mills provide a single line of accessibility, four-axis mills provide a full circle of accessibility and five-axis mills can provide a hemisphere of accessibility (unrestricted accessibility would be represented as a full sphere).

Three-axis Rough Machining

Perhaps the most common example of three-axis rough machining is in the manufacturing of molds and dies. In these products, either a deep pocket (cavity) or a large island (core) needs to be machined from a large block of steel or aluminum (Figure 4). While the finishing process may be performed using more complex multi-axis tool motions to control the posture of the finishing tool, the rough machining can easily be accomplished on a three-axis mill since the majority of molds and dies have complete accessibility from a single orientation (assuming sufficient tool length).

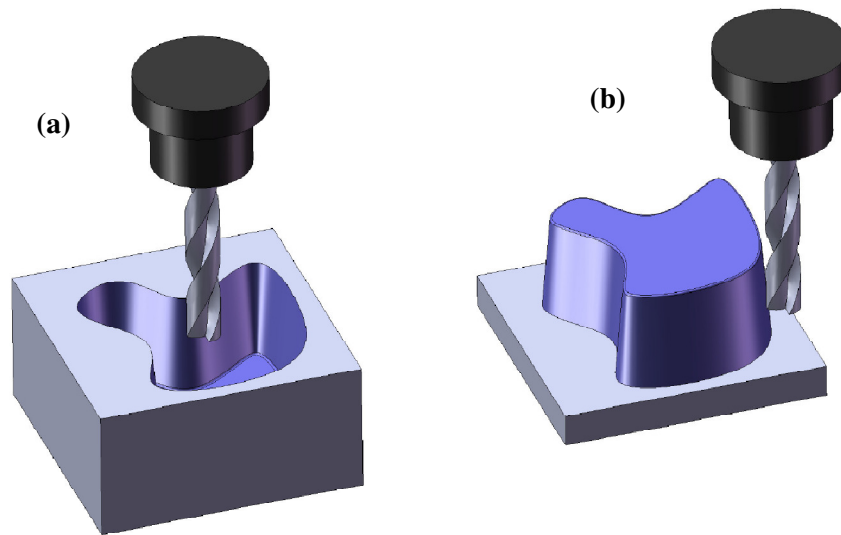


Figure 4. (a) Pocket milling and (b) island milling, both accessible using three-axis mills

In nearly all three-axis roughing strategies, the three dimensional (3 D) geometry is decomposed into 2 ½ dimensional (2 ½ D) slices or ‘slabs.’ This allows both easy computation of the tool path in computer aided manufacturing (CAM) software packages and efficient cutting in the machine tool. When the geometry is decomposed into individual slices, the problem simplifies to that of covering a given planar area with the tool profile. The result of this strategy resembles a terracing effect, also called a ‘waterline’ strategy. One advantage of this method is that as the tool cuts into the stock material (decreasing depth), it is known what material has been removed on previous slabs and collisions between the tool and remaining stock material can be easily avoided.

With this in mind, much of the research in three axis roughing has focused on how to best cover the planar slabs with a tool path. A variety of strategies have been proposed; some use simple zigzag motions while others use more complex motions involving offsets of the tool or stock material. In many of these strategies, the profile of the stock material in the cutting slab must be exactly known. For single setup three axis work pieces, this is typically a trivial matter.

Multi-setup Rough Machining

In low-volume applications it may be practical to fully machine a component from a large piece of available stock (either prismatic or round). Unlike molds and dies, these components typically need to be machined from multiple setup orientations to fully create the desired geometry. In this situation, it would be common to perform a plurality of three-axis roughing and finishing operations - from different setup orientations - to transform the stock into the desired component (Figure 9).

Though more than one three axis roughing tool path would be used, the strategy in each would remain the same as single setup three-axis roughing: a series of 2 ½ D tool passes would sequentially remove 'slabs' of the stock.

Multi-axis Rough Machining

Certain components cannot be machined via three-axis milling due to accessibility problems; an example often used is an impeller blade (Figure 5). In the impeller blade, the high and changing curvature of each blade obscures a considerable volume of material if viewed from a single orientation. Though this geometry could conceivably be produced using multi-setup three axis milling, the number of setups required would be prohibitively large. In cases like this, the accessibility circle or hemisphere of four- or five-axis mills (respectively) gives more flexibility and allows the tool to completely and efficiently remove material.

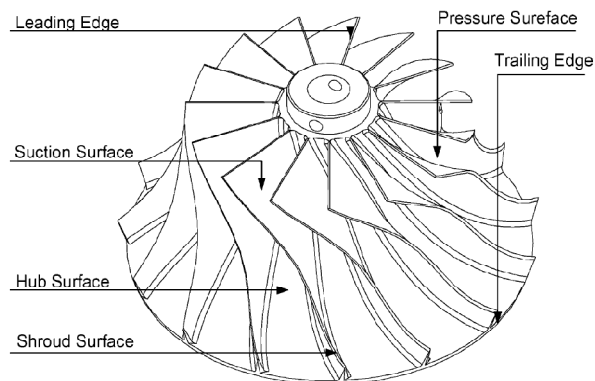


Figure 5. Centrifugal impeller design, common in five-axis machining research [21]

Often, an attempt is made to simultaneously coordinate all five axes of the machine in rough machining. Though quite good in three axis environments, current computer aided manufacturing (CAM) software packages are not adept at generating robust tool paths for general roughing cases in four- and five-axis situations [21]. In fact, certain research has found that it is more efficient (both for process planning and actual processing) to treat the rotary fourth and fifth axes as automated setup mechanisms [8]. In this fashion, the fourth and/or fifth axes rotate to a static position, a three-axis roughing strategy is executed, then the rotary axes reposition the work piece for the next three-axis tool path. Doing this, the robust nature of three-axis roughing tool paths can be utilized while automating setup tasks that would otherwise be overwhelming. In essence, this method becomes an automated case of multi-setup roughing.

Problems in Multi-setup and Multi-axis Rough Machining

Multi-setup and multi-axis roughing present new challenges not found in single setup machining. Both tend to rely on basic 2 ½ or 3 D tool path strategies, and the most efficient of these strategies rely on knowledge of exact stock material boundaries [10, 16].

Single setup three-axis roughing, such as that encountered in mold and die manufacturing, has a well known stock boundary. Because ‘slabs’ of material are removed in a top-down fashion, the stock perimeter in a slab is not changed by preceding slab removal. However, in situations where a plurality of approach orientations are used (either multi-setup or multi-axis), the stock perimeter is inevitably changed by removal of slabs at preceding approach orientations. This often causes over-coverage of volumes (‘air’ cutting) and unnecessarily long machining times (Figure 6).

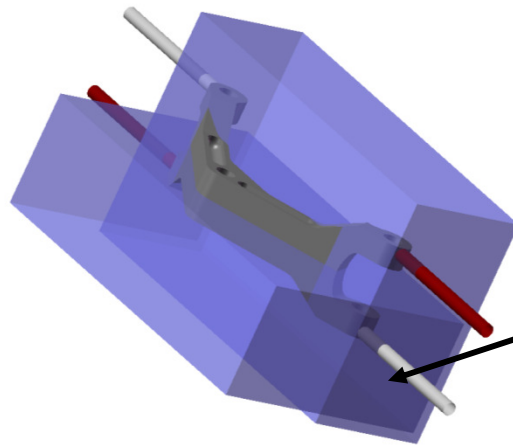


Figure 6. Overlapping swept volumes from different approach orientations lead to tool path inefficiency

The interfering volume of slabs removed from different approach orientations also causes problems with *emergent structures*: thin *webs* and *strings*. When approach orientations are directly opposing, the last few slabs to be removed can form a very thin *web* structure, which is not rigid enough to support cutting forces and standard chip formation (Figure 7). Instead, the web will tend to chatter or wrap and bind the tool, causing either tool breakage or work piece damage.

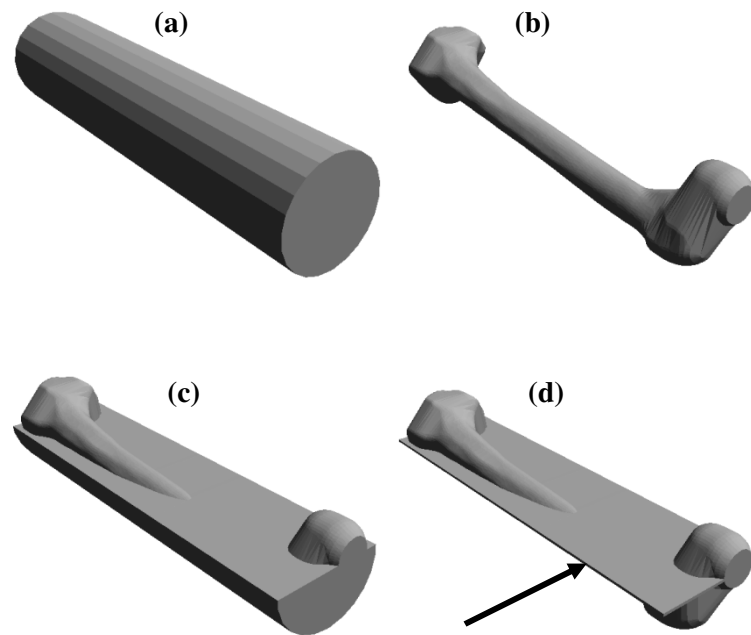


Figure 7. (a) Beginning stock and (b) desired component; (c) and (d) illustrate the thin web that emerges when cutting from opposing orientations, as in this case of 0-180 degree orientations

Similar problems result when two approach orientations leave a *string* of material that is not attached to the final component geometry (Figure 8). When a subsequent approach attempts to machine away this string of stock material, catastrophic results occur as in web removal.

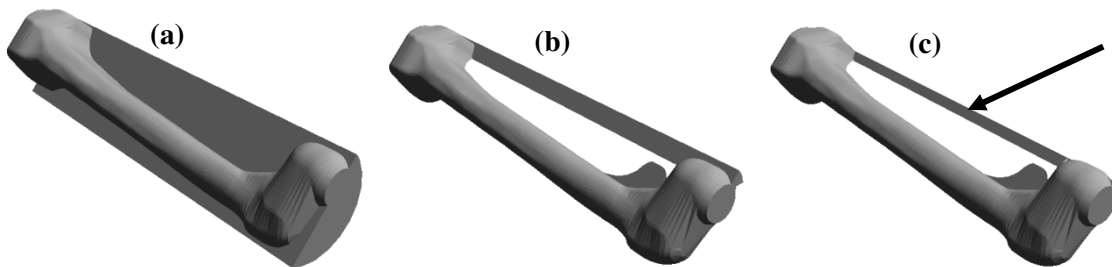


Figure 8. Thin string emerges (c) when cutting orientations are not directly opposing, as in this case of 0-120-240 degree orientations

These *emergent structures* are presented here as a new metric of process robustness; an ideal process plan not allow these structures to form. If the exact profile of the stock material were understood, these conditions could be avoided by altering the approach orientation sequence and strategy.

No method exists to efficiently calculate the condition of the stock material in multi-setup and multi-axis roughing during process planning. A variety of tool path simulation methods and software packages are available, but these methods require generating tool paths first. The computational cost of integrating such systems into tool path planning would be very high and would require many iterations of tool path planning to avoid thin webs and strings. A more desirable solution would be able to identify the changes to the stock profile *before* exact path planning by applying a basic understanding of 2 ½ and 3 D material removal. This thesis will present an efficient new method of modeling stock material that enables simulation of 2 ½ D cuts without prior knowledge of exact tool paths.

Thesis Organization

The remainder of the general introduction provides a literature review of the state of three-axis and multi-axis roughing strategies. This review demonstrates the need for a new process planning methodology to solve the problems of efficiency and robustness in multi-setup and multi-axis roughing applications. To fill this void, Chapter 2 presents original work in paper format: a novel remaining stock simulation method. The final chapter of the thesis provides general conclusions and an outline for future research directions.

Literature Review

This review of roughing literature is divided into two parts: (1) three-axis milling and (2) multi-setup and multi-axis milling.

Three-axis Roughing

A large body of literature is devoted to three-axis roughing strategies, where several tool path routines have been proposed. The two most common types of three-axis roughing routines are *offset machining* and *contour-map machining* [16], though *plunge milling* has been introduced more recently [6].

In offset machining, a ball mill is used to traverse decreasing 3 D offsets of the component geometry until the component surface is exposed. This approach becomes very inefficient when the stock and component geometries are not similar and, furthermore, ball milling is not capable of the higher material removal rate of flat end mills [17]. [18] proposes plunge milling for finishing vertical walls and [6] extends this idea to optimize the plunge strategy for roughing, though large scallops remain in free-form surfaces. By far the most popular strategy is the contour-map approach, where the stock volume is divided into slabs that can be sequentially machined in 2 ½ D operations.

Within the contour-map approach of three-axis roughing, a multitude of methods have been used to create the tool path on each slab. [14] discretize the stock and component volumes into ‘blocks’ of fixed size (approximately the tool diameter) and removes only those blocks that exist in the stock (but not component) volume. [22] use an octree approach in a similar fashion, though with the advantage of variable ‘block’ resolution. This octree approach decomposes to a quadtree problem on the planar (slab) level. [12] and [20] discuss generating 2 ½ D roughing paths directly from point clouds, such

as those obtained from reverse-engineering technologies. [11] shows preliminary work in generating 3-axis roughing tool paths directly from the standard STEP file format. They also decompose the problem into a set of 2 ½ D slabs, though their work is limited to components with planar faces.

The most accepted method of creating 2 ½ D slab tool paths has been *contour offsetting* [4, 9, 10, 16, 19]. In this methodology, ‘slices’ of the three-dimensional stock and component are taken. The problem then becomes that of covering a planar area with a tool profile. This creates an extremely robust tool path where, at each slice, the material removed from preceding slices is exactly known. Much of the literature has focused on optimal patterns for covering these planar areas; notable patterns include the *stock offset*, *component offset*, *stock/component hybrid offset*, *parallel offset*, *proportional blending offset*, and *max-min offset* [9, 10, 16, 19].

Optimization of three axis roughing has centered on the contour offsetting approach. Within this area, tool selection and offset patterns have been shown to be very influential on the efficiency of a tool path. [1] provides a review of tool selection methods along with their original optimization approach. [9], [10], [16], and [19] investigate optimal offset patterns for each 2 ½ D slab. Of particular importance to this research, [10] and [16] show that the selection of the optimal offsetting pattern is dependent on the relationship between the stock and component. In fact, for an example set of stock and component contours it was shown that the optimal strategy results in a 40% path reduction [16]. Thus, if the stock profile is not exactly known, tool path efficiency suffers greatly.

Multi-setup / Multi-axis Roughing

More recently, tool path generation research has shown an interest in five-axis finishing strategies. Unlike three-axis milling, the additional degrees of freedom often allow infinite tool postures at any

point on the surface of a component. Thus, there is a much more significant problem in detecting collisions and gouges in five-axis finishing than in similar three-axis applications.

Five-axis roughing encounters the same problem, though the literature has focused almost entirely on the unique geometry of impeller blades [3, 5, 8, 13, 15, 21] and failed to provide any general solutions for five-axis roughing strategies. Very recently, [21] noted that CAM packages tended to lack generality in five-axis tool path planning and left much to be desired in tool path planning for multi-axis roughing. [2] notes that five-axis CAM modules typically only consider three degrees of freedom, rather than the full five possible.

This same criticism applies to the literature cited here. While the aforementioned impeller component has accessibility issues that certainly warrant five-axis machining, the roughing strategies proposed in the literature rely heavily on the nature of the ruled surfaces of impeller blades and the repetitive pocket geometry encountered. [3], [5], and [21] use offsets of the impeller blades to create tool paths that sequentially deepen the pocket between adjacent blades. [13] uses a layer-based approach with decreasing z depth, using the extra degrees of freedom to counter accessibility problems while traversing the x - y plane.

[2] provides one of the most general five-axis roughing algorithms found in the literature. They utilize four degrees of freedom in their process planning, using the two rotational axes to alter tool posture in x - y slices. In this manner, the efficiency and robustness (emergent structure) problems presented in this thesis are basically overcome. However, this is done with an unfortunate loss of generality since only four of the five degrees of freedom are employed. In essence, this approach is similar to single setup three-axis applications, but with variable tool posture.

Interestingly, even in multi-axis milling, a multi-setup approach using 2 ½ D tool paths has been proven quite practical. In this manner, the fourth and fifth rotational axes are used to automatically position the work piece for a particular setup, the three linear axes (x , y , and z) are used to execute a tool path, then the rotational axes automatically reposition the work piece for the next pass. This often increases the effective rate of machining because the linear axes are capable of responding much faster than the rotary axes [8], [13]. [8] and [15] use this type of three-axis control with two-axis positioning for machining impeller blades. [8] shows that this strategy actually results in both considerably shorter tool paths and a higher material removal rate than full five-axis control. [7] employ a similar strategy in four-axis machining, using the rotary axis for automatic setup execution between 2 ½ D tool paths for three-axis control.

When considering discrete rotary axis positioning, an additional problem is introduced: the required tool vectors must be determined to drive positioning requirements. [7] solves this problem using a visibility map and corresponding set cover analysis. [8] uses the projections of ruled impeller blade profiles to determine visible and accessible regions. In the application of impeller blades, the efficiency and robustness problem is not a large issue because machining is limited to relatively independent pockets. In the more general case presented by [7], ‘island’ machining causes major efficiency and robustness problems.

Summary of Literature

The literature reviewed here has several impacts on the direction of this thesis. First, the use of 2 ½ D tool paths with flat end mills is very popular since it provides efficient and robust process planning for three-axis machining. Further, knowledge of the exact stock boundary is critical for generating efficient tool paths. Also, the use of rotary axes as automatic setup mechanisms, rather than full

machine control, has been proven efficient and validated in the literature. Finally, there has been no general method presented in the literature for efficient and robust multi-axis roughing using this approach of rotary axis setup automation.

References

1. Balasubramaniam, M.; Y. Joshi; D. Engels; S. Sarma; Z. Shaikh. "Tool selection in three-axis rough machining." *International Journal of Production Research*, v39 n18 p4215-4238 2001.
2. Balasubramaniam, M.; P. Laxmiprasad; S. Sarma; Z. Shaikh. "Generating 5-axis NC roughing paths directly from a tessellated representation." *Computer Aided Design*, v32 n4 p261-277 2000.
3. Bohez, E.L.J.; S.D.R. Senadhera; K. Pole; J.R. Duflou; T.Tar. "A Geometric Modeling and Five-Axis Machining Algorithm for Centrifugal Impellers." *Journal of Manufacturing Systems*, v16 n6 p422-436 1997.
4. Chan, K. W.; W. K. Chiu; S. T. Tan; T. N. Wong. "A high-efficiency rough milling strategy for mould core machining." *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, v217 n3 p335-348 2003.
5. Chuang, L.C.; H.T. Young. "Integrated rough machining methodology for centrifugal impeller manufacturing." *International Journal of Advanced Manufacturing Technology*, n34 p1062-1071 2007.
6. El-Midany, T.T.; A. Elkeran. "Optimal CNC Plunger Selection and Toolpoint Generation for Roughing Sculptured Surfaces Cavity." *Journal of Manufacturing Science and Engineering*, v128 p1025-1029 2006.
7. Frank, M., R. Wysk, S. Joshi. "Determining setup orientations from the visibility of slice geometry for rapid computer numerically controlled machining", *Journal of Manufacturing Science and Engineering, Transactions of the ASME*, v128 n1 2006.
8. Heo, E.Y.; D.W. Kim; B.H. Kim; D.K. Jang; F.F. Chen. "Efficient rough-cut plan for machining an impeller with a 5-axis NC machine." *International Journal of Computer Integrated Manufacturing*, v21 n8 p971-983 2008.

9. Hu, Y.N.; W.C. Tse; Y.H. Chen; Z.D. Zhou. "Tool-Path Planning for Rough Machining of a Cavity by Layer-Shape Analysis." *International Journal of Advanced Manufacturing Technology*, n14 p321-329 1998.
10. Li, H.; Z. Dong; G.W. Vickers. "Optimal toolpath pattern identification for single island, sculptured part rough machining using fuzzy pattern analysis." *Computer Aided Design*, v26 n11 p787-795 1994.
11. Liang, M.; S. Ahamed; B. van den Berg. "A Step based tool path generation system for rough machining of planar surfaces." *Computers in Industry*, v32 p219-231 1996.
12. Lin, A.C.; R. Gian. "A Multiple-Tool Approach to Rough Machining of Sculptured Surfaces." *International Journal of Advanced Manufacturing Technology*, v15 p387-398 1999.
13. Morishige, K.; Y. Takeuchi. "5-Axis Control Rough Cutting of an Impeller with Efficiency and Accuracy." *Proceedings of the 1997 IEEE International Conference on Robotics and Automation*, v2 p1241-1246 1997.
14. Trochu, F.; Y. Abong; M. Balazinski; P. Larbrisseau. "Rough machining of free-form surfaces defined by dual kriging." *International Journal of Production Research*, v34 n6 p1603-1623 1996.
15. Umehara, T.; T. Ishida; K. Teramoto; Y. Takeuchi. "Effective tool pass generation method for roughing in multi-axis control machining." *Transactions of the Japan Society of Mechanical Engineers*, v73 n8 p2387-2393 2007.
16. Vickers, G.W.; H. Li; Z. Dong. "Automated Rough Machining of Curved Surfaces." *Transactions of the Canadian Society for Mechanical Engineering*, v17 n4 p647-657 1993.
17. Vickers, G.W.; K.W. Quan. "Ball-Mills Versus End-Mills for Curved Surface Machining." *Journal of Engineering for Industry*, v111 p22-26 1989.
18. Wakaoka, S.; Y. Yamane; K. Sekiya; N. Narutaki. "High-speed and high-accuracy plunge cutting for vertical walls." *Journal of Materials Processing Technology*, v127 n2 p246-250 2002.
19. Wang, H.; H. Chang; R.A. Wysk; A. Chandawarkar. "On the Efficiency of NC Tool Path Planning for Face Milling Operations." *Journal of Engineering for Industry*, v109 p370-376 1987.
20. Wu, S.; C. Wang; J. Fan. "Generate rough tool-paths from unorganized point-cloud directly." *Chinese Journal of Mechanical Engineering*, v20 n5 p1-4 2007.

21. Young, H.T.; L.C. Chuang; K. Gershwiler; S. Kamps. “A five-axis rough machining approach for a centrifugal impeller.” *International Journal of Advanced Manufacturing Technology*, v23 p233-239 2004.
22. Yuen, M.F.; S.T. Tan; W.S. Sze; W.Y. Wong. “Octree approach to rough machining.” *Proceedings of the Institution of Mechanical Engineers, Part B, Management and engineering manufacture*, v201 nB3 p157-163 1987.

Chapter 2: A Remaining Stock Algorithm for Multiple Setup Subtractive Processes

A paper submitted to the *ASME Journal of Manufacturing Science and Engineering*

Joseph E. Petrzela and Matthew C. Frank

Abstract

This paper presents an algorithm for machining analysis that provides a 3D model of the incrementally changing shape of stock material as it is processed with a series of subtractive operations. Contrary to existing verification and simulation techniques, this manufacturing analysis can be performed prior to tool path planning. The method provides a robust calculation of material conditions as a function of cut depth and orientation in three or four axis milling environments. The method is based on analysis and modification of cross section slice data from the model. The algorithms provide cross sections and surface models that can be used for mathematical or qualitative analysis. This method augments traditional tool path planning by allowing more efficient processing by using more accurate current stock condition to set containment boundaries during process planning. The method is also effective in avoiding stock conditions, such as thin webs, that could cause poor machining or catastrophic failure.

1 Introduction

This paper presents a new method for subtractive manufacturing analysis. Building on concepts of computational geometry, this robust and efficient algorithm computes remaining stock material for subtractive processes. Based on inputs of part geometry, initial stock geometry, and cutting direction and depth, a polygonal approximation of the incrementally modified model is created to represent machining progress. The unique nature of the method is that it is independent of tool path generation. Specific

contributions include a unique finite range visibility shadowing algorithm and a novel method of polyhedral reconstruction from polygonal cross sections.

Background and Motivation

Subtractive processes are extremely versatile and can form a variety of complex geometries. However, to form arbitrary geometry from stock, material must be removed from a variety of orientations, or setups. For example, if one were to mill a part from a block of stock material, several setups would be required; one possibility, though not necessarily optimal, would be to mill the part from orthogonal angles (Figure 9).

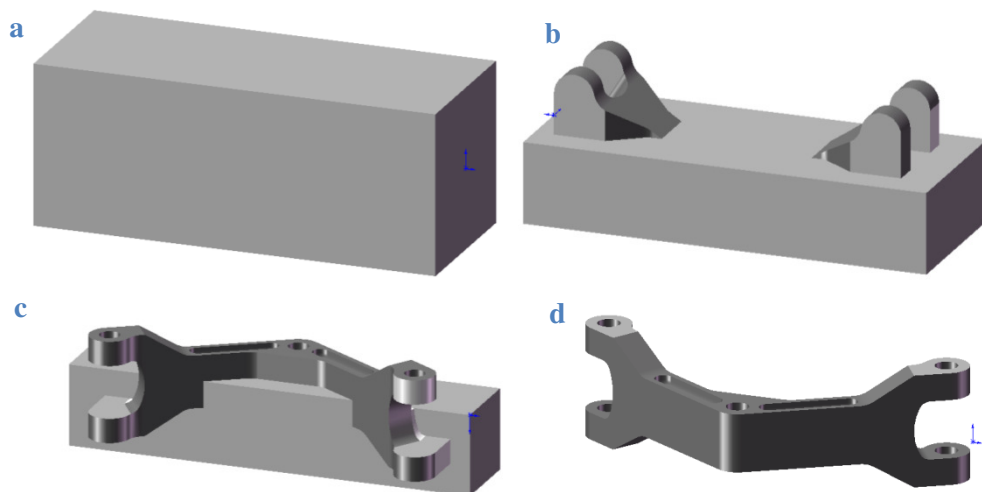


Figure 9. Multiple setups required (even in multi-axis milling) to create component via a subtractive process; (a) shows initial stock geometry and (b-d) show the results after iterative cutting operations until the component fully emerges (d)

In situations such as this, it becomes very difficult to analyze the remaining stock after a certain sequence of operations. The problem is far from trivial; component geometry will often occlude accessibility to certain portions of the stock material. A variety of tool path simulation and verification techniques have been presented in the literature; however, these techniques focus computational effort on surface finish and conformity rather than gross shape analysis.

Little research has been conducted that specifically evaluates the effect of multiple-setup roughing operations; in fact, many typical situations do not require this type of analysis. In usual production settings, a near-net shape component is created (i.e. casting), and then critical features are machined. In die and mold shops, a single setup in a multi-axis mill is typically sufficient to achieve the geometry of the die or mold cavity. Regardless, there exist a variety of applications where multiple setup subtractive analysis is a requirement, especially for creating custom or unique parts from nominally sized stock material.

In this research, we consider a subset of this problem where all setups can be performed by rotation about a single axis. The manner of iteration is not important; the work piece could be fixtured in a four-axis machine and automatically rotated to new orientations, or the work piece could be manually fixtured in a three-axis mill such that each subsequent orientation was about a theoretical axis (as in Figure 9).

Recent development of a rapid prototyping process using CNC machining by the authors [8] has enabled automated process planning for custom, very low volume, or prototype components. In this system, called CNC-RP, sacrificial support structures provide fixturing for machining in a four-axis milling machine. Visibility analysis of cross sectional slice data provides a basis for automated setup planning about a single axis. The implementation uses a modified Greedy set cover algorithm to determine discrete orientations; depth requirements are generated by 'bucketing' visible slice segments for each orientation. In its full implementation, CNC-RP safely and completely machines arbitrary components from bar stock in a four-axis mill. In effect, the fourth axis serves as a rotary indexer to automate setups between independent three-axis tool paths (Figure 10).

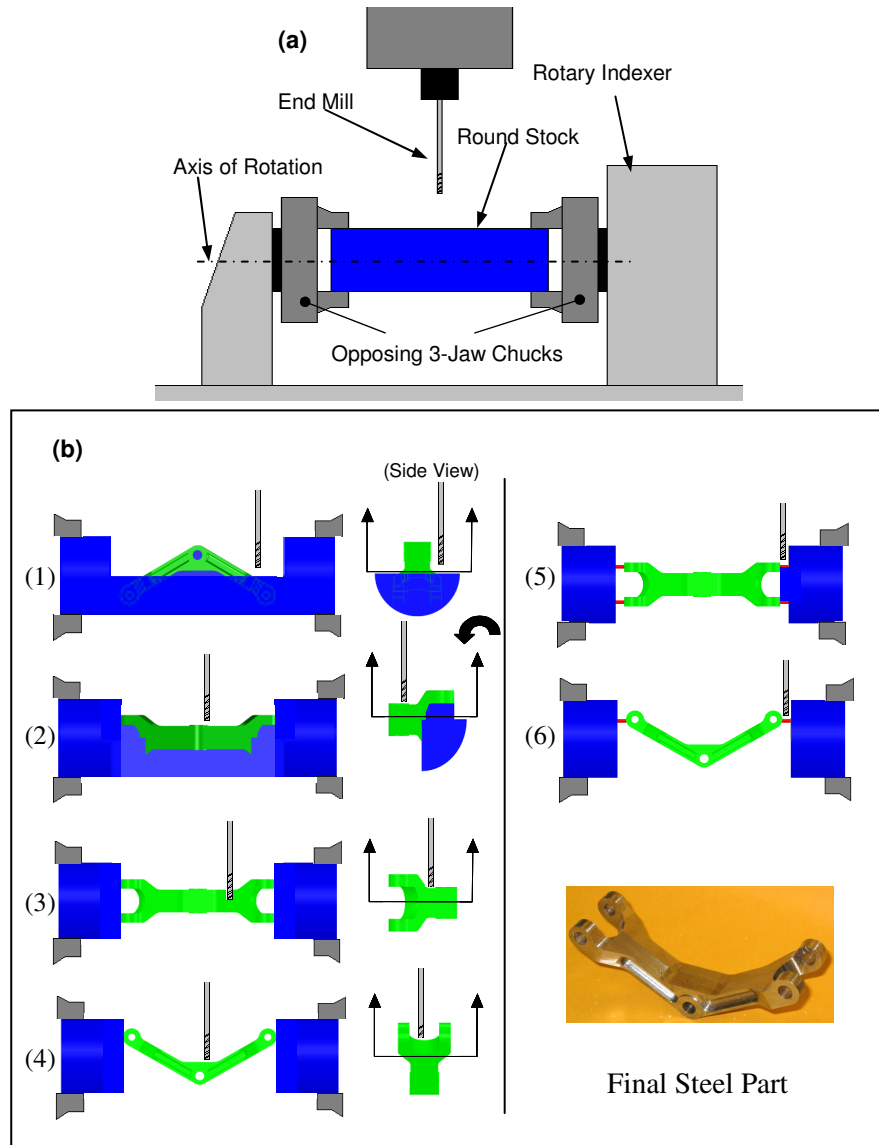


Figure 10. (a) CNC-RP Implementation; (b) Process sequence of steps (b.1-b.4) to expose component geometry and (b.5-b.6) to expose sacrificial supports

While this system has proven capable and robust, it generates each tool path independently of others, basing it only on set cover angles and depth results. The conservative tool path planning method results in efficiency sacrifices for this multiple orientation process. Because each machining orientation is performed independent of others, there is no knowledge of prior operations during roughing procedures. This naïve approach leads to excessively long tool paths and redundant volume sweeps (Figure 11). If an

effective means existed to predict the stock geometry resulting from each step (setup) in the process, one could generate much more effective and efficient tool paths.

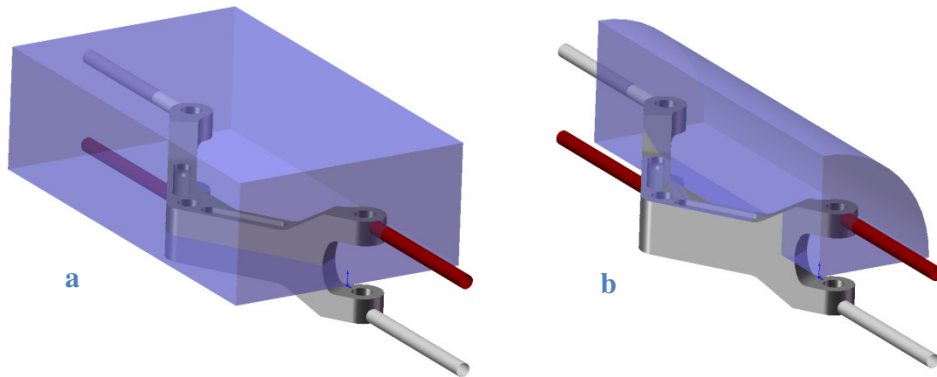


Figure 11. (a) Naïve tool path volume sweep; (b) Volume sweep of tool path with knowledge of prior operations

Saito [18] cites machining time as one of five critical factors to successfully producing complex geometries; naïve tool path generation fails to minimize this factor. To characterize the tradeoff between robust (naïve) and fast characteristics, we define two types of error that may occur when generating the in-process stock material shape:

Type A: Remaining stock is underestimated. Failure mode: possible collision between tool and large amount of stock.

Type B: Remaining stock is overestimated. Failure mode: marginal increase in tool path length and machining time.

Noting the failure modes and the desire to develop a robust system, it is asserted that Type A error must be avoided at all costs. For example, in the lights-out operation of the CNC-RP method, it is absolutely necessary that tool crashes do not occur during untended operations, often done overnight. Without simulation of previous operations (three axis tool paths at different orientations), each subsequent tool path is required to assume worst-case conditions (Type B error). While little original stock material may actually remain, a large volume is swept by the tool to maintain a robust process (Figure 11a).

Jerard [12] suggests that NC simulation, verification, and correction be used to increase the feed rate of a tool if it is found to be traversing outside the boundary of remaining stock material. Although this can reduce processing time for an existing tool path, it would be far more efficient to reduce or eliminate the generation of tool moves outside the remaining stock material.

Commercially available CAM packages can accept STL (triangle mesh surface) models as stock material input to constrain tool path generation to the correct regions in space. Attempts to utilize existing simulation techniques in CAM packages to generate this driving STL model have not been successful. Z-buffer based simulations provide reasonably accurate results, though not robust. After several iterations between the Z-buffer dexel structure to a portable STL format (for use simulating several sequential machining operations), the data becomes unreasonably large and prone to corrupt STL files. Furthermore, this and similar simulation techniques rely on pre-generated tool paths.

These difficulties create the need for a simple remaining stock calculation for multiple setup environments that is (1) mathematically defined, with portable results, (2) computationally inexpensive for application to many unique components, and (3) able to predict process performance prior to computationally expensive tool path planning.

2 Related Work

Computer Aided Manufacturing (CAM) algorithms can be classified as (1) tool path generation methods or (2) simulation / verification methods [18]. Advances in verification techniques (beyond graphical simulation) focus on gouge and under-cut detection. The literature fails to address the area of 'lightweight' pre-NC computation (gross shape analysis only, neglecting surface scallops) for process planning in a multiple-setup environment. One area of recent research has focused on five-axis simulation and verification techniques, though the examples provided in literature are single-setup

simulations with variable tool posture. Possible applications of these techniques include complex pocket milling for molds and dies; these types of five-axis problems are very different from and cannot be generalized to a multiple setup situation where complete work piece rotation is feasible.

Tool path simulation is similar to the remaining stock calculation desired; it creates a model of the stock by subtracting the tool's swept volume from a specific tool path. Verification techniques go a step beyond to detect gouges (over-cutting during feed motion), under-cutting (failing to remove all material from a component surface), and collisions (driving to tool into a part or fixture during rapid travel). By definition, simulation and verification calculate (in various manners) the Boolean effect of a tool path. Three primary groups of verification / simulation are evident in the literature: vector based, spatial discretization, and solid/parametric surface based methods.

Vector Based:

Chappel [5] and Oliver/Goodman [16] developed vector based approaches to track machining progress. After populating component surfaces with normal vectors (the density determined relative accuracy and computation time, though in $O(n)$), the vectors are trimmed as appropriate for each tool instance. Jerard [12] extended this simulation technique to verification and cutter location (CL) data correction. Simulations from this method would be difficult to translate to a generic model (i.e. STL), though they have the advantage of a precisely defined tool-surface proximity measurement (subject to vector density) for exact gouge or undercut detection.

Spatial Discretization:

The most popular spatial discretization method – the Z-buffer - was originally proposed by Van Hook [19]. A graphics based approach, the Z-buffer uses x - y 'dexels' in place of pixels; the dexels contain depth (near and far surface) characterization and a pointer (for hidden bodies). This method is extremely

efficient and scales well, with computational complexity of $O(n)$. Van Hook's original method was view-dependent and required recalculation if the simulation was to be rotated for a different view. Huang and Oliver [11] extended this simulation technique to verification by detecting gouging and undercutting. They also fit a triangle mesh to the dixel structure to allow efficient graphics translation without recalculating the simulation. Speeds as high as 68.6 instances per second were reported for 3-axis roughing (where each tool instance is one dixel unit from the previous instance). Saito and Takahashi [18] presented a similar graphics-based approach (G-buffer method) that is capable of both tool path generation and simulation. Bohez et al. [3] discretize space into x - y planes in their method. Using slice (cross-section) data from stock geometry, a sweep plane in the z -direction performs Boolean subtraction of the tool swept volume. The modified slices of the stock geometry can then be compared with slice data of the reference (desired) geometry using graphics-based stencil operations. This method can be adapted to five-axis problems, but is in general very slow.

Solid / Parametric Surface Based:

Solid modeling would seem to be ideal for NC simulation and verification; an exact representation of remaining material could be constructed. However, constructive solid geometry (CSG) application becomes $O(n^4)$ and therefore unacceptable for any simulation of practical scale [12]. There does not appear to be extensive recent research in this area; most efforts have been focused on parametric boundary representation (Brep) methods (effectively a transition from solid to surface modeling). Using Brep methods, Weinart [21] achieves 5-axis swept volume approximation resulting in parametric NURBS surfaces. Boolean subtraction of this swept volume from a stock model results in a tool path simulation. Weinart still interpolates between discrete tool instances (similar to discretization methods) and shows only a small example with no large scale implementation. Jütler [13] presents a similar method, though he used discrete points along a rational B-spline curve to drive tool instances. Jütler presents purely empirical research with no successful implementation. Blackmore [2] generates triangle mesh

representations of swept volume using differential equations applied to continuous tool paths and boundaries. This method would not apply to the interpolated linear moves (discontinuities exist) common in CAM generated tool paths. All of these Brep methods are slow [3] and do not appear to be mature technologies based on the lack of full scale implementations in the literature.

Integrating Simulation and Process Planning:

Though these traditional simulation methods evaluate existing tool paths, several attempts have been made to integrate simulation with tool path generation to autonomously create more robust tool paths. Lauwers [15] and Huang [11] both use simulation results to correct CL data (tool position & posture). Pal [17] presents a remaining stock computation to drive more efficient tool paths, but his algorithm is designed only for two-stage three axis milling with sequentially smaller ball mills. Many of these simulation techniques are well suited to verify gouging and undercutting in surface finishing operations. However, most are very slow (Z-buffer methods are the only that can be considered fast [3]) and all rely on computational investment in tool path generation. The literature does not present methods suitable for gross stock remainder calculations for more efficient roughing processes. Only vector based approaches offer mathematically defined tool tracking with respect to the surface (though some other methods can derive it from discretized maps).

3 Overview of Methodology

In this section, we provide a general overview of the proposed methods for remaining stock calculation. The approach involves three major tasks; (1) the model of the stock material is broken into cross sectional slices and a set of operations performed to create a factor of safety for avoiding Type A errors, (2) slices are modified to simulate the iterative changes to the stock, and finally (3) a reconstruction of the slice data into a surface model representation of the stock for each iterative step in the manufacturing process.

The method presented computes the remaining stock for a set of subtractive processes where all setups occur about a single theoretical axis. In particular, this analysis computes the net effect of a tool path from a given orientation to a given depth, but without requiring exact tool path data. If a robust tool path generator is used, the only gain from considering tool instances is the ability to analyze surface scallops; for a roughing operation this is considerably less important than in finishing operations. Our method consists of three distinct algorithms: slice approximation, slice shadowing, and polyhedral reconstruction (Figure 12).

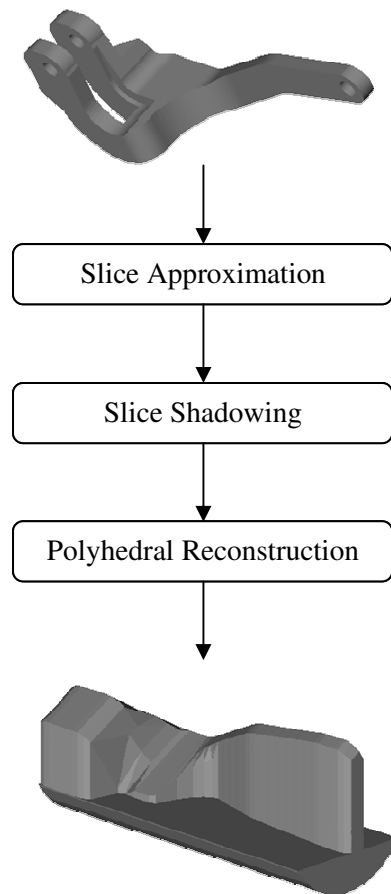


Figure 12. Method for remaining stock computation is composed of three distinct tasks, each accomplished through new algorithms

A basic description of the three algorithm steps is provided next, with a detailed presentation in Section 4.

Slice Approximation

Slice approximation is a pre-processing step used to convert a surface model of the part geometry to cross sectional polygonal data. Since we assume all setups occur about a single theoretical axis, the problem can be represented as a set of cross sections perpendicular to and along that axis. This provides the distinct advantage of reducing the overall problem to independent analyses of two-dimensional data from a three dimensional model.

However, the preprocessing involves more than simply slicing the model. Subsequent algorithms in the method use a visibility based shadow to approximate remaining stock, so the slices must account for the difference between *visible* and *accessible* regions. If a visibility approach is used on the exact cross section of the part, Type A error will occur for all but the simplest geometries (remaining stock will be underestimated). To avoid this, the cross sectional data must exclude small concave features (i.e. holes and slots) that a tool of specific finite diameter may not be able to access. The approach to counter visibility underestimation is twofold: cross sections are (1) reduced to their own convex hull polygons and (2) redefined as the union of all slices within one tool radius along the axis of rotation. The first measure has the effect of closing concave geometry that is radial relative to the axis of rotation, while the second measure closes concave geometry along the axis (Figure 13). As the final step of slice approximation, the resulting cross sections can be offset in order to approximately account for scallop effects. Since specific tool instances are assumed unknown, the location of scallops is likewise unknown. To this point, offsetting the polygonal cross sections allows compensation for possible scallops and for any amount of intentional under-cutting (e.g. leaving some set amount of material for finishing operations).

The net effect of the slice approximation steps described above is shown in Figure 13 with a reconstructed model. In particular, note that the small holes and slots inaccessible by a roughing tool have been eliminated; however, thick features have been slightly over approximated (Type B error). This reduction

in accuracy is the cost of using a robust form of visibility analysis during slice alteration; however, it provides a critical degree of safety to ensure tool collisions will not occur.

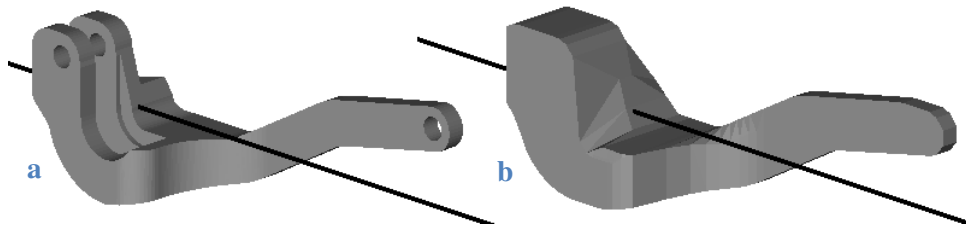


Figure 13. An example component (a) and reconstruction of approximating slices (b) showing elimination of inaccessible holes and slots; lines represents axis of rotation.

Slice Shadowing

Slice shadowing, the second of three algorithms, is central to the remaining stock analysis. After the model has been represented as a set of cross sections, each cross section is independently modified with a unique ‘limited visibility shadow’. Unlike infinite visibility from a given orientation, we explicitly define a finite visibility range to emulate the finite cutting depth common to subtractive processes such as machining. This shadow is readily defined by Boolean operations between the part cross sections and initial stock cross sections. The algorithm is generalized to allow multiple ‘cuts’ to different depths from different orientations.

Polyhedral Reconstruction

The resulting contours from the shadowing operation compose a rich data set, though not portable in any widely accepted format. Hence, the third and final step is a polyhedral reconstruction of the stock geometry from the modified slice data, which converts the data to a common format (STL model) for graphics rendering and use in CAD/CAM packages. Though polyhedral reconstruction from polygonal cross sections is a relatively well known problem, we present new methods unique to this application.

Our work addresses the three primary problems of reconstruction: *correspondence*, *branching*, and *tiling* [1].

The correspondence problem must be solved when multiple contours exist in adjacent cross sectional slices or when the distance between adjacent cross sectional slice planes is significantly large. In these situations, one cannot be certain of the exact ‘correspondence,’ or mapping, of contours between two slice planes (determining *which* slice contours correspond to *which* features from the sliced stock/part model). The general cases for up to two contours in each slice plane are illustrated in Figure 14. For this application, we present a set of proofs that reduce the correspondence problem between multiple contours to a concise and well-defined solution set.

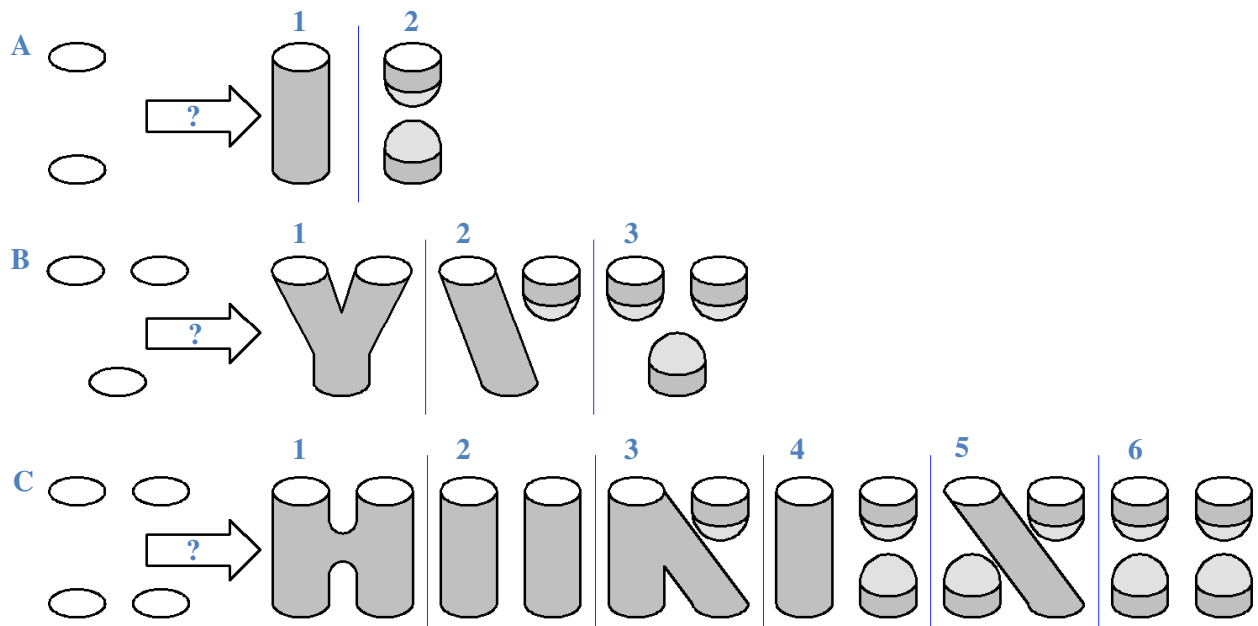


Figure 14. Correspondence types between (A) 1:1, (B) 2:1, and (C) 2:2 contour sets

Once correspondence is determined, the branching problem deals with determining which topology is correct at the juncture of multiple contours (e.g. Figure 14.B.1). The literature supports several approaches for this problem, and a simple yet effective heuristic is adopted for our work. After solving the branching and correspondence problems, facets must be placed between corresponding contours to

create manifold topology. While a number of solutions to the problem have been proposed, we present a new method of tiling that outperforms other heuristics. A new two-stage approach is employed: in the first stage, ‘landmark’ contour points are matched by finding the longest path in a toroidal graph, and in the second stage previously published heuristics are synthesized to tile between corresponding landmark points.

4 Detailed Presentation of Methodology

A detailed presentation of the entire methodology, including all three algorithmic steps, is given in this section. To augment the discussion of each of these algorithms, results from an example component (Figure 13) will be shown as appropriate. The example component will be shown in multiple forms, as it begins from the existing model of both the part itself and the corresponding stock, through each of several steps in a multi-setup subtractive operation.

4.1 Slice Approximation

To begin, a stereolithography (STL) model of a part and fixturing geometry (representing the space set in which no machining should take place) is sliced into sets of contours using existing slicing methods. For multiple setup processes, the cross sectional slice planes are set normal to the axis of rotation.

Define:

$\mathcal{C} : \{C_1, C_2, \dots, C_n\}$	Component data set \mathcal{C} of n slices, each denoted C_i
l	Total length of component along rotational axis
d	Distance between slices
r	Tool radius

Since this methodology is later used to drive tool path planning, it must never have Type A error so that collisions do not occur; equivalently, the amount of remaining stock must never be underestimated. To maintain a lightweight model and avoid the computational complexity associated with collision or machinability algorithms, two steps are taken to further process the slices and avoid underestimating material removal in visible regions that a tool could not actually reach. First, each cross section is reduced to its own convex hull.

$$C_i = \text{ConvexHull}(C_i)$$

$$i = 1 \dots n$$

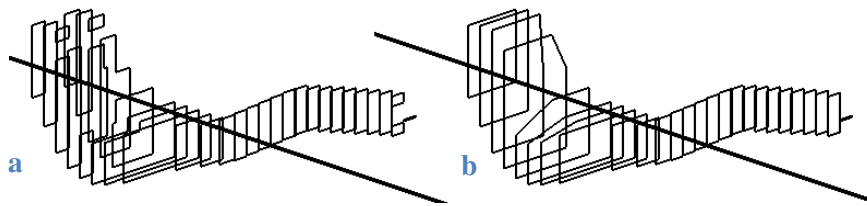


Figure 15. Each slice (a) is represented as its own convex hull polygon (b)

This step has the effect of eliminating slots or other internal geometry radial with respect to the axis of rotation (any given convex hull is completely visible and accessible by any convex 2D tool silhouette).

Second, each slice is redefined as the union of itself and all slices within one tool radii.

$$C_i = (C_{i-(r/d)}) \cup (\dots) \cup (C_i) \cup (\dots) \cup (C_{i+(r/d)}) \quad i = 1 \dots n$$

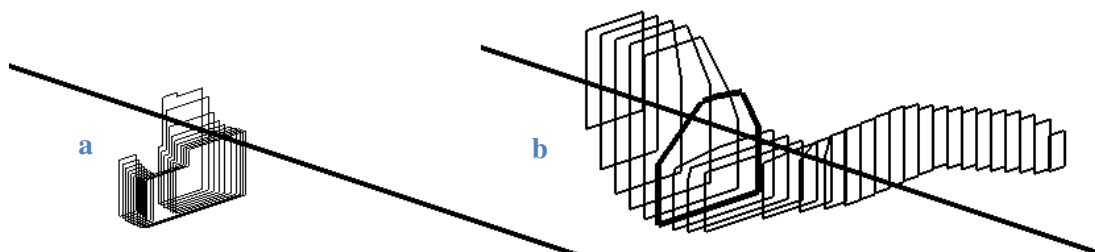


Figure 16. Each slice (b) is also redefined as the union of all slices within one tool radius (a); for clarity, the set of slices in (a) are those surrounding the bold slice in (b)

This step offsets all possible collision surfaces by one tool radius and closes small gaps and slots along the axis of rotation. Both of these steps contribute Type B error, where the remaining stock is over-estimated (though not detrimentally so; the resulting tool path efficiency is discussed in Section 5).

Furthermore, if the component is being intentionally under-machined (i.e. leaving extra stock above the true surface for finishing operations), each slice can be offset by the amount of under-machining. Offsetting by the machining step down will also account for surface scallops without exact modeling. This external offset of a convex polygon can be performed very simply and efficiently in $O(n)$ time; the exact algorithm is not presented here.

$$C_i = \text{Offset}(C_i) \quad i = 1 \dots n$$

In addition to the part geometry, slices must also be created to represent initial stock geometry. Slices can be generated automatically to represent homogenous stock material (bar or prismatic stock), or a model of the beginning stock can be sliced to create cross-sections. Regardless, the beginning stock is assumed to be of convex cross section along the axis of rotation.

Define:

$$\mathcal{S} : \{S_1, S_2, \dots, S_n\} \quad \text{Stock data set } \mathcal{S} \text{ of } n \text{ slices, each denoted } S_i$$

Note that for stock material that is homogenous along the axis of rotation (i.e. bar or round stock), all S_i in \mathcal{S} are equivalent.

4.2 Slice Shadowing

Once the component has been approximated as a series of slices, \mathcal{C} , the effect of cutting is simulated on each slice C_i independently. For this simulation, we note that the remaining stock will be equivalent to

the union of the stock material (cut to a certain depth), the component, and the visibility shadow of the component (where the tool could not reach). Recall that measures were taken during ‘Slice Approximation’ to account for the difference between visibility and accessibility analysis. For the purposes of detailed discussion, we further define:

m	Number of cutting orientations
$\alpha : \{\alpha_1, \alpha_2, \dots, \alpha_m\}$	Set of cutting angles
$\delta : \{\delta_1, \delta_2, \dots, \delta_m\}$	Set of cutting depths, relative to axis of rotation viewed from angles α
$A_j : \{A_{1,j}, A_{2,j}, \dots, A_{n,j}\}$	Set of n intermediate slices, representing the visibility shadow from angle α_j
$B_j : \{B_{1,j}, B_{2,j}, \dots, B_{n,j}\}$	Set of n intermediate slices, representing stock material (without component or shadow) from angle α_j to depth δ_j
$R_j : \{R_{1,j}, R_{2,j}, \dots, R_{n,j}\}$	Set of n slices, representing effect of a single cut from angle α_j to depth δ_j
$ReSt_j : \{ReSt_{1,j}, ReSt_{2,j}, \dots, ReSt_{n,j}\}$	Set of n slices, representing effect of cumulative cuts from 1 ... j

With the final goal of achieving slice set $ReSt_j$, sets A_j , B_j , and R_j are computed. Each is presented below:

Slice set A_j represents a visibility shadow from angle α_j cast onto set \mathcal{C} . Because each C_i is a known polygon, a visibility shadow can be easily determined by finding points in C_i tangent to α_j . To find these points, some vector in the direction of α_j is determined (denote $\vec{\alpha}_j \in \mathbb{R}^2$). Then, the cross product between this vector and each segment (span between polygon vertices p_k and p_{k+1}) in the cross section is found ($\vec{\alpha}_j \times \overrightarrow{p_{k+1} - p_k}$); if the sign of this cross product changes between two segments, their shared endpoint is tangent to α_j .

In the case presented, the polygonal cross section is known to be convex, thus, there are exactly two points tangent to any given direction vector. Two new vertices, $p_{u,j}$ and $p_{v,j}$, can be added to the polygon point set at some arbitrarily large distance in the direction of α_j from tangency points p_u and p_v ($p_{u,j} = p_u + \vec{\alpha}_j$). Treating the polygon vertices as a point cloud and recomputing the convex hull yields a ‘shadow’ contour to some finite distance, $|\vec{\alpha}_j|$ (the general case of non-convex polygons would require a simple modification to detect the intersection of the shadow ray and other polygon segments). Finally, this ‘shadow’ is trimmed to the stock cross section via intersection and the result is stored as $A_{i,j}$.

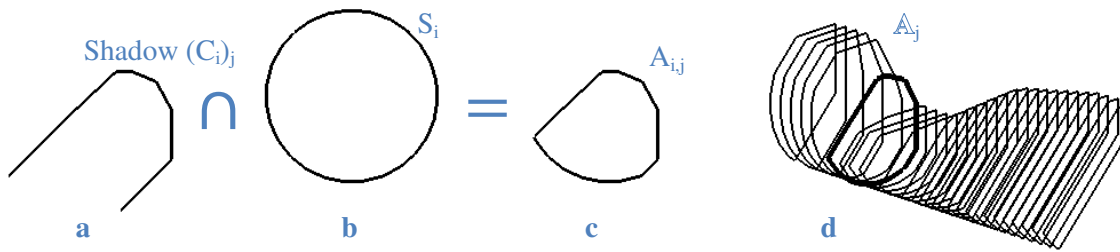


Figure 17. A_j (d) is the set of slices $A_{i,j}$ (c) representing a visibility shadow of C_i at angle α_j (a) trimmed to stock cross-section S_i (b)

Slice set \mathcal{B}_j represents the effect on stock material of a cut from angle α_j to depth δ_j , assuming for now that no component geometry restricts tool access (we are only considering stock cutting for \mathcal{B}_j). First, an arbitrarily large polygon is created to represent the half-space not accessible by a tool at depth δ_j at angle α_j . In practice, this is done by forming a sufficiently large square centered on the origin, translating this square polygon so that the top segment is at depth δ_j , and rotating the polygon by angle α_j about the origin. Then, each $B_{i,j}$ can be computed by trimming, via intersection, the stock cross section (S_i) to the appropriate ‘half-space’ polygon. In the special case that the stock is uniform along the slicing (cross sectional) axis, all $B_{i,j}$ are equivalent for each cut j and only one $B_{i,j}$ need be computed.

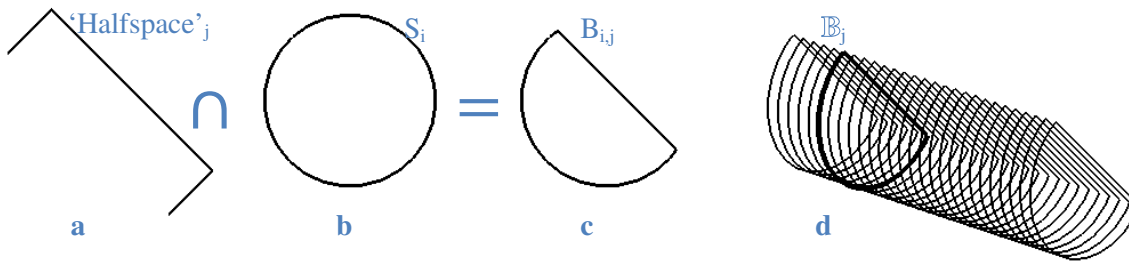


Figure 18. \mathbb{B}_j (d) is the set of slices $B_{i,j}$ (c) composed of stock material unreachable from angle α_j to depth δ_j , computed by trimming the unreachable 'halfspace' (a) to the stock cross-section (b)

The union of respective slices in \mathbb{A}_j and \mathbb{B}_j yields \mathbb{R}_j , representing the net effect of a single cut from angle α_j to depth δ_j . Without formal proof, we assert that this represents remaining stock assuming that a tool path is used that has no gouge (removing too much material) or undercut (failing to remove all desired material) motion.

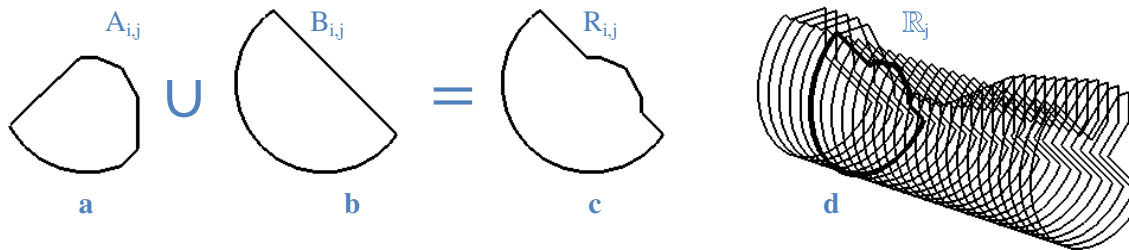


Figure 19. Single-cut effect is represented by slice set \mathbb{R}_j (d), where each $R_{i,j}$ (c) is computed by joining shadow $A_{i,j}$ (a) and stock material $B_{i,j}$ (b)

Finally, to evaluate the cumulative effect of cutting operations, the set $ReSt_j$ is computed. In the natural sense, one would use each \mathbb{R}_j as the beginning stock material for computing the effect of cut $j+1$. However, this makes computations dependent on each other; it would be more appropriate to compute each $ReSt_j$ in terms of the preceding $\mathbb{R}_1 \dots \mathbb{R}_j$ such that all \mathbb{R}_j can be computed independently for large scale simulations. In fact, we can prove this possible by considering the case of $ReSt_{1,2}$. Resetting S_i equal to $R_{i,1}$, we obtain:

$$ReSt_{i,2} = (Shadow(C_i)_2 \cap R_{i,1}) \cup (Halfspace_2 \cap R_{i,1})$$

which can be shown to equal:

$$ReSt_{i,2} = (A_{i,1} \cup B_{i,1}) \cap (A_{i,2} \cup B_{i,2}) = R_{i,1} \cap R_{i,2}$$

Similarly, it can be shown that:

$$ReSt_{i,j} = R_{i,1} \cap R_{i,2} \cap \dots \cap R_{i,j}$$

Thus, the cumulative effect of cuts ($ReSt_i$) is computed as the cumulative intersection of single-cut effects (R_j), allowing independent calculation of single-cut effects regardless of operation order.

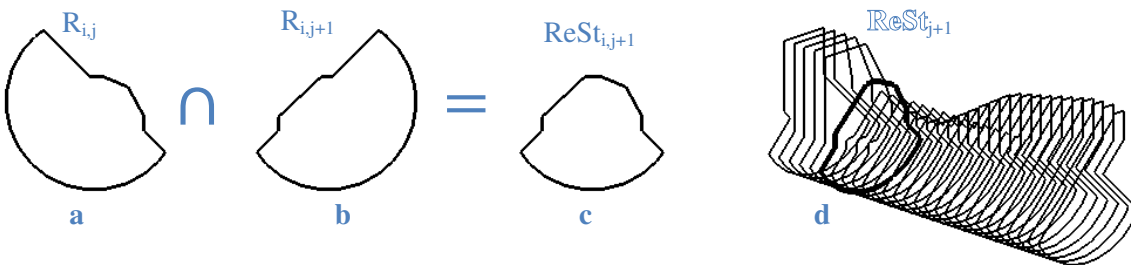


Figure 20. Each $ReSt_{i,j}$ (c) in $ReSt_{j+1}$ (d) is computed as the cumulative intersection of preceding single-cut effects $R_{i,j}$ (a,b), allowing independent computation of $R_{i,j}$; it (d) is also the final set of slices that is reconstructed to form a polyhedral model

4.3 Polyhedral Reconstruction

The final step in the method is to create a polyhedral model of the remaining stock after all operations have been simulated. To do this, each $ReSt_i$ in $ReSt$ is used to reconstruct a polyhedral representation (in STL format) of $ReSt$. Extensive literature has been published on this specific problem, focusing generally on three individual problems of reconstruction from planar cross sections: the *correspondence* problem, the *branching* problem, and the *tiling* problem [1].

Correspondence

The correspondence problem arises from two situations: first, when planar sampling is relatively sparse and the geometry between planar sets is unknown, and second, when multiple contours exist in the same cross section. The general cases (of up to two contours) are illustrated in Figure 14. A variety of algorithms, rules, and methods have been presented to correctly determine correspondence between contours in adjacent planes. However, we can make certain assumptions based on characteristics of this application that allow consideration of exactly three well-defined cases. In particular, we make the following clarification:

If m total operations occur, any number of operations may occur to reduce the general size of the stock to a prismatic form *as long as* none go to a depth that would allow component geometry to emerge (cuts $1\dots x$). This allows more aggressive cutting parameters for removing much of the stock material surrounding a component. Once a cut exposes component geometry, the next two cuts must finish exposing the convex hull of the component (cuts $x+1\dots x+3$). Any number of remaining cuts may be taken after this to finish forming concave geometry (cuts $x+4\dots m$).

Two proofs are given below. By Proof 1 below, considering the generic cases shown in Figure 14 is sufficient; each slice will have at least one and at most two contours. Then by Proof 2, we can disregard all correspondence types except A1, B1, and C2 (Figure 14). Thus, the following rules are proposed for correspondence:

- If each slice has one contour: Tile between these contours.
- If each slice has two contours: Map each contour across slices (using centroid proximity) and tile between respective contours.

- If one slice has one contour, and the other has two: Apply a branching algorithm and tile between contours.

Proof (1); that at least 1 and at most 2 bodies (polygons) exist per slice:

For this proof, consider four cases: cuts $\{1\dots x\}$, $x+1$, $x+2$, and $\{x+3\dots m\}$:

For cuts $j \in \{1\dots x\}$:

Assuming that the stock geometry begins as a convex shape, these operations simply reduce the remaining stock to a smaller convex shape. Since the depth of cut does not expose C_i , then

$$R_{i,j} = A_{i,j} \cup B_{i,j} = B_{i,j}$$

Thus, the computation of $ReSt_{i,j}$ becomes:

$$ReSt_{i,j} = R_{i,1} \cap R_{i,2} \dots R_{i,j} = B_{i,1} \cap B_{i,2} \dots B_{i,j} \quad (\text{for } j \leq x)$$

B can be reformulated as $D \cap S$, where D is the half-space unreachable by a tool at depth δ and angle α .

Then $ReSt_{i,j}$ becomes:

$$\begin{aligned} ReSt_{i,j} &= (D_1 \cap S_i) \cap (D_2 \cap S_i) \cap \dots \cap (D_j \cap S_i) \\ &= S_i \cap (D_1 \cap D_2 \cap \dots \cap D_j) \end{aligned} \quad (\text{for } j \leq x)$$

Each $D_{i,j}$ incrementally trims the material (beginning with S_i) to a smaller polygon. Because $D_{i,j}$ can be thought of as an infinitely large convex polygon, and S_i is convex, the intersection of the two is either empty or another convex polygon. Also, because each $D_{i,j}$ contains C_i (the depth of cut does not expose geometry for operations $j \leq x$) and C_i is contained in S_i (the beginning stock material completely encloses the component), in the least material condition (LMC):

$$ReSt_{i,j} = S_i \cap (D_1 \cap D_2 \cap \dots \cap D_j) \geq S_i \cap C_j = C_j \quad (\text{for } j \leq x)$$

So $ReSt_{i,j} = S_i \cap (D_1 \cap D_2 \cap \dots \cap D_j)$ is never an empty set and must remain a single convex polygon. •

For cut $j=x+1$ (first operation to expose geometry):

$ReSt_{i,x+1} = A_i \cup B_i$. If B_i is the null set, then $ReSt_i = A_i$ and is a single contour. If B_i exists, then:

$$B_i = S_i \cap D_i$$

Because A_i shares the edge of S_i in the direction of α_{x+1} , and D_i is the half-space in the direction of α_{x+1} , B_i and A_i must share a common edge along S_i in the direction of α_{x+1} . Therefore, the union of A_i and B_i must be exactly one contour. •

For cut $j=x+2$ (second operation to expose geometry):

Remaining stock can be modeled as:

$$ReSt_{i,x+2} = (A_{i,x+1} \cup B_{i,x+1}) \cap (A_{i,x+2} \cup B_{i,x+2})$$

Let

$$A_{i,j} = C_i \cup X_{i,j}$$

Where $X_{i,j}$ represents the complement of C_i in $A_{i,j}$. Then remaining stock can be shown to equal:

$$ReSt_{i,x+2} = C_i \cup (B_{i,x+1} \cap B_{i,x+2}) \cup (X_{i,x+1} \cap X_{i,x+2}) \cup (X_{i,x+1} \cap B_{i,x+2}) \cup (X_{i,x+2} \cap B_{i,x+1})$$

Equation 1

Examining Equation 1, there must be at least one contour since C_i will never be null. There may be at most two contours if $(B_{i,x+1} \cap B_{i,x+2})$, representing the remaining stock without shadow effects, is disjoint from C_i . To prove that there are at most 2 contours, we show that the remaining three terms of Equation 1 represent regions that are either adjacent to or overlapping C_i or $(B_{i,x+1} \cap B_{i,x+2})$:

In this situation, $(X_{i,x+1} \cap X_{i,x+2})$ represents the shadow region. If $\alpha_{x+1} = (\alpha_{x+2} + \pi)$ then this region is null. Otherwise, it is adjacent to C_i and does not create a separate contour (Figure 21).

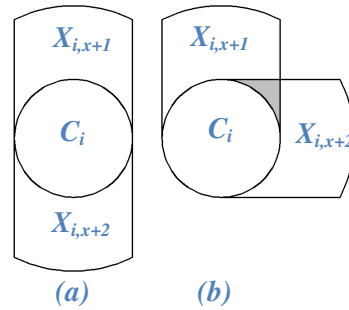


Figure 21. Illustration of shadow intersection: (a) where $\alpha_{x+1} = (\alpha_{x+2} + \pi)$ and the intersection is null, and (b) where the intersection is adjacent to C_i

The fourth term, $(X_{i,x+1} \cap B_{i,x+2})$, can be rewritten as $(X_{i,x+1} \cap B_{i,x+2}) \cup (C_i \cap B_{i,x+2})$ without affecting the balance of Equation 1. This can be further reduced to:

$$B_{i,x+2} \cap (X_{i,x+1} \cup C_i) = B_{i,x+2} \cap A_{i,x+1}$$

Because $A_{i,x+1}$ and $B_{i,x+2}$ are convex sets, their intersection must be either null or have at least two edges that are edges in $A_{i,x+1}$ (recall that $A_{i,x+1}$ and $B_{i,x+2}$ will share boundary edges from S_i , Figure 22a). Each edge in $A_{i,x+1}$ is also in either C_i or $B_{i,x+1}$ except for two (Figure 22b), and these two cannot be adjacent (figure ???). Thus, $B_{i,x+2} \cap A_{i,x+1}$ is either null or must contain edges in, and thus be adjacent or overlapping, $(B_{i,x+1} \cap B_{i,x+2})$ or C_i .

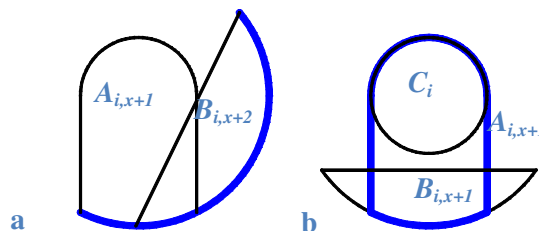


Figure 22. (a) $A_{i,x+1}$ and $B_{i,x+2}$ share (bold) boundary edges from S_i , (b) only two edges (neither adjacent to the other) of $A_{i,x+1}$ are not in either C_i or $B_{i,x+1}$

A similar proof could be shown for the remaining term, $(X_{i,x+2} \cap B_{i,x+1})$. Therefore, $ReSt_{i,x+2}$ has at least one and at most two contours. •

For cut $j=\{x+3 \dots m\}$ (third and subsequent cuts to expose geometry):

For any cut after $j=x+3$, there remains exactly one contour in $ReSt_{i,j}$. Because the convex hull of the component has been exposed by cut $j=x+3$, each $ReSt_{i,j}$ has been reduced to C_i by $j=x+3$. •

Proof (2); that we can disregard all correspondence types except A1, B1, and C2 from the previous presentation (Figure 14):

For cuts $j=\{1 \dots x+1, x+3 \dots m\}$:

According to proof 1, each $ReSt_{i,j}$ contains exactly one contour, and this contour contains C_i . Because C_i can be assumed to be continuously connected across all slices, the correspondence type *must* be 1:1 continuous (Figure 14.A.1). •

For cut $j=\{x+2\}$:

Each $ReSt_{i,j}$ has at least one and at most two contours by analysis of Equation 1. At a minimum, each $ReSt_{i,j}$ will contain term C_i (which will always exist); term $(B_{i,x+1} \cap B_{i,x+2})$ will either be null or exist. If term $(B_{i,x+1} \cap B_{i,x+2})$ is adjacent to C_i for some slice i , there will be one contour; otherwise there will be two.

Because C_i is continuously connected across all slices and $(B_{i,x+1} \cap B_{i,x+2})$ is continuously connected across all slices, there is continuous correspondence of either type 1:1, 1:2, or 2:2 (Figure 14.A.1, -B.1, or -C.2). •

Branching

Once correspondence is determined, the branching problem deals with determining which topology is correct at the juncture of multiple contours. A simple new form of branch reconstruction is utilized in this method, similar to that proposed by Ekoule [7]. Given a single contour in slice S_i and two in S_{i+1} , an augmented contour is formed in slice S_{i+1} by joining the two nearest points of the contours in S_{i+1} . In the example provided (Figure 23), contours A and B in S_{i+1} are joined into an augmented contour, C . Note that segments (C_3, C_4) and (C_7, C_8) are coincident, and this will cause tiles to form a ‘saddle’ at this point to maintain a closed manifold surface.

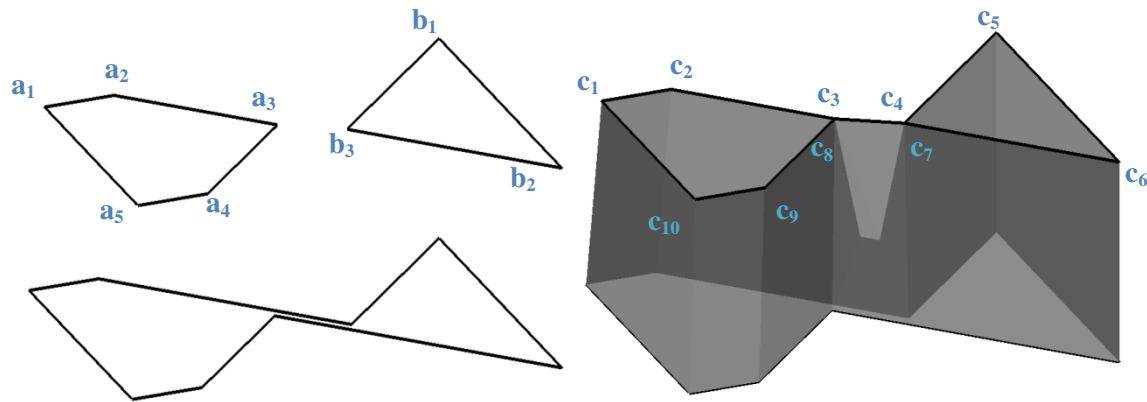


Figure 23. Method of augmenting the contour set to facilitate branching (A and B form an augmented contour, C, by joining their nearest points); actual tiling results are shown with augmented contour C

Finally, tiling is performed between the contour on S_i and the augmented contour C on S_{i+1} . The original contours A and B are used for tiling between S_{i+1} and S_{i+2} .

Ekoule’s method of forming an augmented contour joins the original contours via their centroids, not closest point pair as we propose in this paper. The closest point pair is more appropriate for this application; because it does not add points to the contours and the strict format rules of polyhedral STL models can be maintained more easily. Furthermore, for the type of models that we are reconstructing, it is known that at most two contours will exist in the same slice and that these two contours likely have

well-defined nearest points (intuitively, the contours would form by a machine tool removing material, as in Figure 23). In the general case, Ekoule's original method may be superior (and admittedly other elegant methods have been presented in the literature), but our simplified method is specifically appropriate for the types of branching encountered in this application.

Tiling

In general, the problem is formulated by decomposition into separate problems between respective contours on planes i and $i+1$. Boissonnat [4] poses the problem as a volume calculation and solves by trimming tetrahedra from the associated Delaunay triangulation between the contours. Keppel [14], Fuchs et al. [9], and Wang & Aggarwal [20] combine graph theory and optimization approaches to solve for meshes in the optimal sense. These methods are $O(n^2 \log n)$ complex [1]. Keppel [14] provides the first reference to decomposition of contours into convex and concave components for purposes of matching and tiling. Christiansen [6] proposes a heuristic method that joins contours (denote P and Q) based on a Greedy approach. Christiansen's approach normalizes polygons to containment in a unit square to reconcile contours of differing scale or location, then sequentially adds facets at $|P_i, P_{i+1}, Q_j|$ or $|P_i, Q_j, Q_{j+1}|$ depending on whether the distance from $P_{i+1} \rightarrow Q_j$ or $P_i \rightarrow Q_{j+1}$ is shorter, respectively. Ganapathy et al. [10] present a similar method, though they normalize the perimeter of each contour to unity, then move along each contour in an equidistant fashion by examining the percent of each contour traversed rather than proximity to points in the opposite contour. The difficulty in this method is choosing the correct "seed" location to begin tiling and no elegant method is proposed to select this point. These methods fail for very complex (many concave regions) or dissimilar shapes. Ekoule et al. [7] modify Christiansen's heuristic by decomposing contours into convex portions (similar to that initially presented by Keppel [14]) and mapping concave points onto the convex hull, and then using a similar Greedy approach to form facets.

The method presented here combines aspects of several existing methods in a new manner that considers a more appropriate tiling metric of surface normal characterization. For the purpose of the proceeding discussion of tiling, we define:

- P : A contour of n points in one slice plane, with g landmark points
 Q : A contour of m points in an adjacent slice plane, with h landmark points
 i : An index used in reference to P , from 1 ... n
 j : An index used in reference to Q , from 1 ... m

Tiling, in this case, is reduced to a problem of placing chords between two polygonal contours, P and Q , such that adjacent chords form triangles (with the third side of the triangle being a contour segment). To accomplish this, a two-pass approach is employed. First, landmarks on each contour are identified and primary chords are placed between respective landmark locations. Next, secondary chords are placed between the primary chords to complete the triangle mesh.

This approach is unique both in the use of landmark points and the method by which they are identified and matched. Rather than other metrics of best fit, we assume that points of similar curvature should match between adjacent contours. Thus, landmark points are defined as two types: first, those that form the convex hull set, and second, those that form concave sets between convex hull points. To match these landmark points, we examine the contour normal (\hat{n}_{P_i}) of each.

To compute the contour normal, polygons are decomposed into convex and concave portions similar to Ekoule's method. Then, for each convex or concave point (P_i), a contour normal (\hat{n}_{P_i}) is defined (Figure 24a):

$$\hat{t}_{P_{i+}} = \frac{P_i - P_{i+1}}{|P_i - P_{i+1}|}$$

$$\hat{t}_{P_{i-}} = \frac{P_i - P_{i-1}}{|P_i - P_{i-1}|}$$

$$\hat{n}_{P_i} = \frac{\hat{t}_{P_{i+}} + \hat{t}_{P_{i-}}}{|\hat{t}_{P_{i+}} + \hat{t}_{P_{i-}}|} \quad (\text{convex landmark})$$

$$\hat{n}_{P_i} = -i \cdot \frac{\hat{t}_{P_{i+}} + \hat{t}_{P_{i-}}}{|\hat{t}_{P_{i+}} + \hat{t}_{P_{i-}}|} \quad (\text{concave landmark})$$

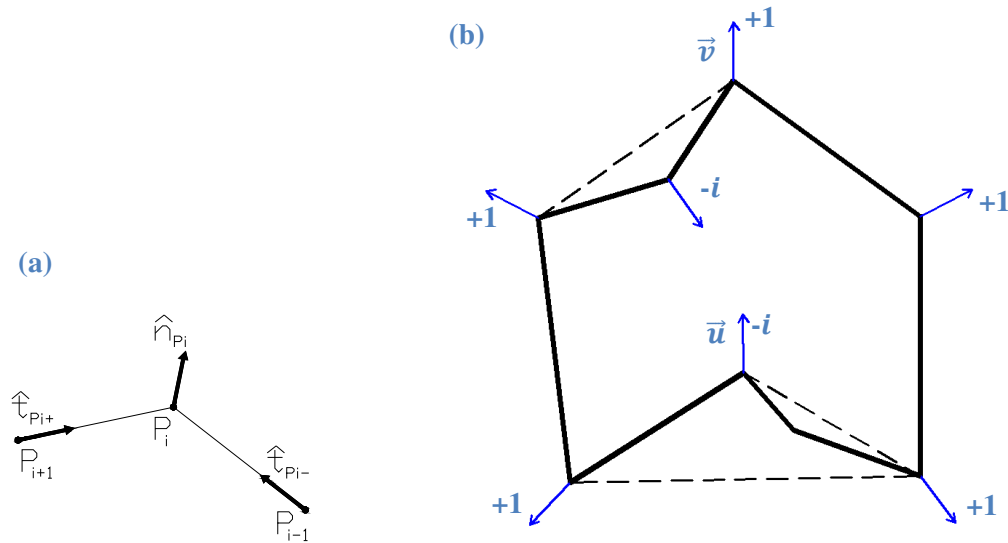


Figure 24. (a) Method of determining normal vector for landmark points, (b) landmark normal vectors, with real and imaginary multipliers, for an arbitrary polygon (note that without the multipliers, matching vectors u and v via dot product could provide incorrect results)

Where P_i and \hat{n}_x are vectors in \mathbb{R}^2 and P_{i-1} , P_i , and P_{i+1} are all in the same convex or concave decomposition set. Note that the contour normal of convex points will point to the exterior of the polygonal contour, while the contour normal of concave landmark points will face the interior.

Next, we wish to find points in P and Q such that \hat{n}_{P_i} and \hat{n}_{Q_j} are similar. This can be detected by maximizing the cosine of the included angle, by definition of the dot product:

$$\max \cos \theta = \frac{\hat{n}_{P_i} \cdot \hat{n}_{Q_j}}{|\hat{n}_{P_i}| |\hat{n}_{Q_j}|} = \frac{\hat{n}_{P_i} \cdot \hat{n}_{Q_j}}{1 \cdot 1} = \hat{n}_{P_i} \cdot \hat{n}_{Q_j}$$

Thus, the proximity, or ‘closeness’ of two landmark points in contours P and Q is quantified as $\hat{n}_{P_i} \cdot \hat{n}_{Q_j}$.

Though convex and concave landmark points will point in different directions (locally), the contour

normals of concave and convex points on opposite sides of a contour may (undesirably) match closely (see the case of \vec{u} and \vec{v} in Figure 24). To avoid this issue, \hat{n}_{p_i} includes a factor of $-i$ for concave landmarks; in effect, convex landmark points are considered real while concave landmark points are considered imaginary (Figure 24b). Then, when the dot product is computed between landmark points in adjacent contours, the result will be a real number when points of the same nature (convex or concave) are compared, and imaginary otherwise.

To efficiently match landmark points, a toroidal graph (first suggested for this application by Fuchs et al.) is constructed, where each node represents a possible primary chord. Where P contains g landmark points and Q contains h landmark points, the toroidal graph takes the following form:

$$\begin{array}{c} P_1 \\ P_2 \\ \vdots \\ P_g \end{array} \begin{bmatrix} Q_1 & Q_2 & \dots & Q_h \\ \hat{n}_{P_1} \cdot \hat{n}_{Q_1} & & \dots & \\ & \vdots & \ddots & \dots \\ & & & \hat{n}_{P_g} \cdot \hat{n}_{Q_h} \end{bmatrix}$$

The problem then becomes choosing the appropriate chords from the graph, amounting to a longest path search. The problem is simplified by noting that the graph is directed (the contours progress in the same direction) and that, as a condition of feasibility, at most one chord may be chosen in any given row or column of the graph (to avoid intersecting chords and infeasible tiling solutions). Rather than implementing a longest path algorithm, a very efficient heuristic (similar to a g-h scan) is used to select the chord locations with the highest proximity values:

Step1: Declare all nodes 'feasible'

Step 2: Select the node with largest real (not imaginary) proximity value

Step 3: Check to see if the graph remains feasible

- a. If so, set as a ‘chord location’ and set all other nodes in that row and column ‘infeasible’
- b. If not, set the chosen node ‘infeasible’

Step 4: Repeat (2) and (3) while ‘feasible’ nodes exist

While the solutions from this fast heuristic are not guaranteed optimal, the results are very good and the approach scales well. Furthermore, this approach lends itself to future extension if a more optimal longest path algorithm is employed.

In the second pass of the algorithm, the spans between landmark chords must be tiled to complete the surface mesh. To achieve this, landmark chords provide a known beginning and end for the tiling and all points are considered. A greedy tiling algorithm (Christiansen [6]) is used to place remaining chords between contours. For example, if a landmark chord exists between points P_i and Q_j , a new tile will be added with its third vertex at P_{i+1} or Q_{j+1} . This determination is made from the shortest distance between P_{i+1}, Q_j or Q_{j+1}, P_i , respectively. Finally, the results from tiling are written to a standard ASCII STL format file. An example tiling result is given in Figure 13.

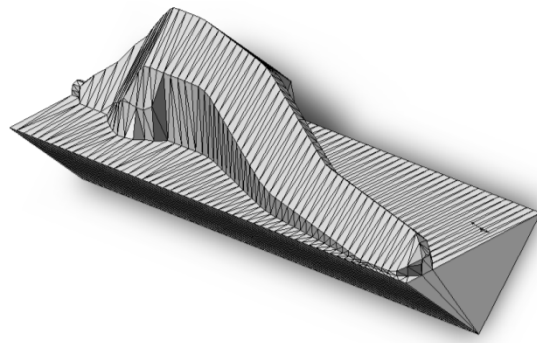


Figure 25. Typical tiling results using convex and concave landmark point matching (note good results along both convex and concave regions)

5 Implementation

The algorithms presented above have been implemented in software and utilized in the laboratory for four-axis subtractive rapid prototyping (CNC-RP). These algorithms are ideally suited for this system because they provide a robust, reliable estimation of remaining stock. In the CNC-RP process, it is more important to achieve safe and effective results for first-pass success in low volume applications than it is to optimize tool paths at a high level (the expectation is that only one, or very few parts are desired). This remaining stock calculation serves exactly that purpose: while the stock is overestimated in certain situations, it is done so to ensure the integrity of results and eliminate collisions between the tool and underestimated stock material while the machine is allowed to run unattended

To illustrate the results typical in the CNC-RP implementation, reconstructed models for a single component (Figure 13), machined in seven setups, are illustrated (Figure 26, where operations 1-3 are omitted for clarity).

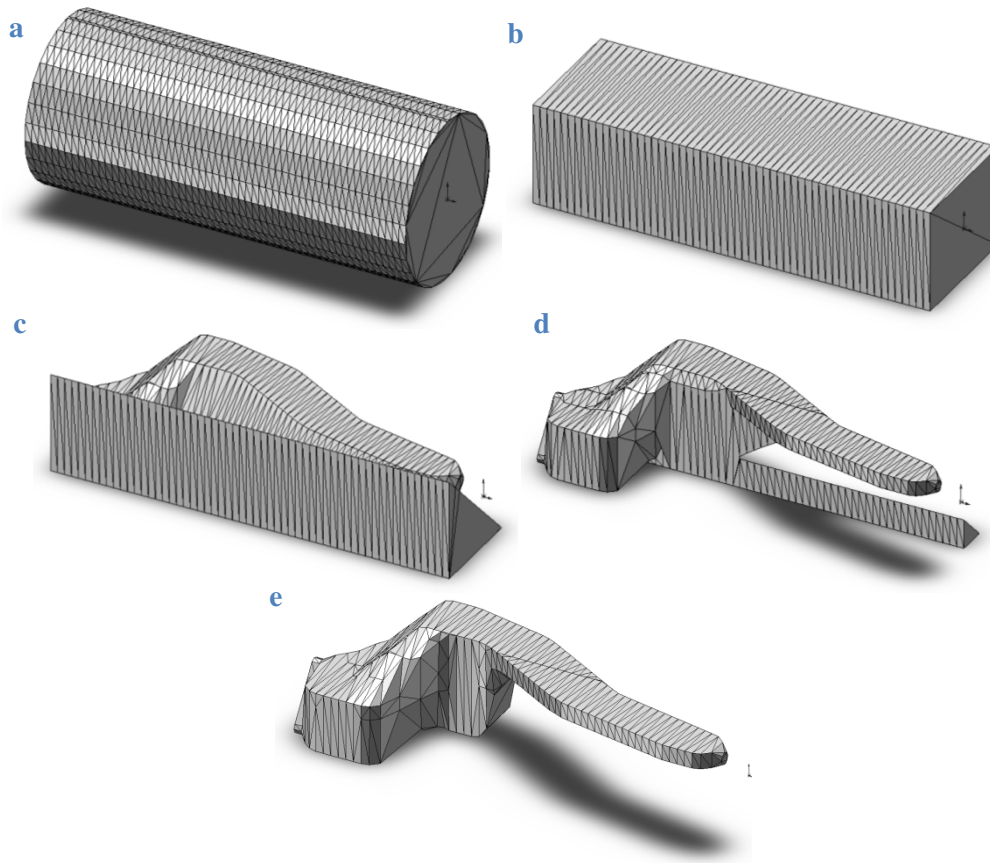


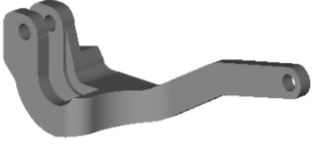


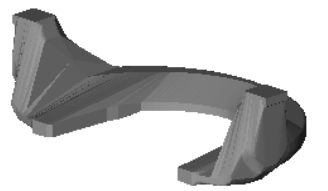
Figure 26. Illustrations of remaining stock for the example component from seven setup orientations: (a) beginning stock, (b-e) setups 4-7, respectively (setups 1-3 omitted)

The efficiency gains from this implementation have been very significant. While a variety of factors may influence tool path inefficiency, one of the largest contributors is adopting a naïve and overly conservative remaining stock assumption. Overall tool path length has been observed to decrease as much as 65% when utilizing this new remaining stock analysis during tool path generation.

Improved efficiencies are presented for two sample parts (Table 1). Each was assumed to be machined from three inch diameter bar stock with sacrificial supports (consistent with the CNC-RP implementation) using a 0.5 inch flat end mill, 0.04 inch step down, and 75% step over. In laboratory testing, the tool was previously observed performing redundant ‘air’ cutting, but with the remaining stock implementation it remains engaged in the material during nearly the entire roughing routine. Numerous laboratory tests on

a variety of geometries have not revealed any cases of Type A error (failure mode of collision with underestimated remaining stock).

Table 1. Implementation results (coupling with multi-axis tool path generation) for two different geometries

CAD Model	Reconstructed Remaining Stock Model	Total Setups	Tool Path Length Reduction	Computation Time (3 GHz CPU)
		9	53%	640 milliseconds
		8	64%	640 milliseconds

6 Limitations and Future Work

This work is limited primarily by the assumption that setups must be about a single axis. However, in practice, it is observed that most geometries can be achieved in this manner (from machinery components to free-form biologic shapes such as human bones). This fact may be due to the motivation behind design for manufacturing to reduce setups for industrial components. Using visibility software in the lab, the authors have found few bones of the human anatomy (i.e. the entire skull and pelvis) that cannot be manufacturing about one axis of rotation.

The steps taken during slice preprocessing to avoid underestimating the remaining material (approximation as a convex hull) cause slight overestimation in many cases. This constraint could be relaxed if a machinability algorithm were incorporated and a more robust correspondence algorithm implemented for polyhedral reconstruction. While this would result in a more exact approximation, the machinability algorithm would substantially increase computation time.

As a final limitation, the method presented does not convey scallop effects or surface finish in the remaining stock estimation. However, the method presented is not intended to replace conventional verification and simulation techniques for finishing operations; rather, its strength is in advanced, efficient process planning.

Jerard [12] proposes a framework for automated NC code generation via an iterative process between setup orientations, setup sequencing, tool path generation, and tool path verification. With the algorithms we present, setup orientation and sequencing can be improved *without* iteration between computationally expensive tool path generation and verification algorithms. The robust and simple nature of this algorithm lends itself to future simulation or optimization techniques for determining

optimal setup strategies. The Z-buffer method was the fastest pre-existing simulation method [3]; Huang and Oliver [11] cite a computation speed of 68.8 instances / second for a three-axis roughing routine (analogous to the operations analyzed here). Even at this rate, the tool path for the example part would have taken several minutes to simulate (for 9069 linear inches of feed move in the example component's NC program), even at modern computing speeds. However, by evaluating the net effect of the tool path rather than each individual move, our method requires only 0.6 seconds to simulate nine different operations from different setup orientations (considering the example part on a 3 GHz CPU). Because the net effect of a tool path requires only milliseconds to calculate, this algorithm is appropriate for application to optimization techniques.

One such application for focus of future work is for the detection of thin material conditions. If operations are not sequenced optimally, machining from opposing setup angles (at or near 180°) can form thin webs or thin strings. These conditions can be readily detected by our algorithm at the slice level.

The method presented can also be applied to design analysis. The shadow method can be used to augment visibility algorithms. While visibility and accessibility algorithms investigate the invisible / inaccessible surface area of a component, our new method can be used to evaluate inaccessible *volume* from a given set of angles. One possible application would be evaluating the amount of material that must be added to regions of a component to make molding or forging possible from a minimal number of molds.

Another future application is to simulate wire EDM processes since, in fact, wire EDM cannot readily create concave geometry and it is very nearly a line of sight accessible process (for small wire

diameters). Thus, the assumptions made for the current algorithm would map directly to a wire EDM process.

7 Conclusions

This work presented a new method for gross analysis of subtractive processes in multiple setup environments. As opposed to existing methods, focusing on single setup (albeit multi-axis) surface finishing verification, our method provides a robust and efficient means of evaluating subtractive processes. Specific results of interest are a new limited visibility shadowing algorithm, advances in polyhedral reconstruction heuristics, and efficient analysis of subtractive processes independent of specific tool paths.

The limited visibility shadowing algorithm can be applied to a variety of other situations. Other visibility analyses evaluate surface visibility, and a set cover problem can be formulated to solve for optimal setup angles. Using this algorithm, the problem could be formulated differently to achieve the maximum *accessible volume* in a set cover problem rather than maximum accessible surface area.

The new method of polyhedral reconstruction, especially our tiling metrics and toroidal graph formulation, should advance the research in this area. Rather than previous metrics of shortest chord length or smallest surface area, we match surface normals, creating a more smooth and ‘correct’ surface. Applications are abundant in biomedical imaging reconstruction, reverse engineering, and other areas where planar cross sections must be reconstructed to form a manifold surface. Moreover, the extremely fast nature of this algorithm lends itself to iterative optimization approaches; which has significant implications to improved process planning. If an objective function can be formulated that uses metrics to identify advantageous or detrimental characteristics of process progress, the

millisecond time scale of our algorithms will allow many iterations through setups to identify (at least locally) optimal setup sequences.

8 Acknowledgment

Funding for this research was provided by Deere and Company under the project *A Rapid Machining System for Service Parts*, account #400-60-41.

9 References

- [1] Bajaj, C., E. Coyle, and K.-N. Lin. "Arbitrary Topology Shape Reconstruction from Planar Cross Sections", *Graphical Models and Image Processing*, v58 n6 November 1996.
- [2] Blackmore, D., M. C. Leu, and L. P. Wang. "The sweep-envelope differential equation algorithm and its application to NC machining verification", *Computer-Aided Design*, v29 n9 1997.
- [3] Bohez, E., N. Minh, B. Kiatsrithanakorn, P. Natasukon, H. Ruei-Yun, L. Son. "The stencil buffer sweep plane algorithm for 5-axis CNC tool path verification", *Computer-Aided Design*, v35, n12, October 2003.
- [4] Boissonnat, J.-D. "Shape Reconstruction from planar cross-sections", *Computer Vision, Graphics, and Image Processing*, v44 n1 October 1988.
- [5] Chappel, L. "The use of vector to simulate material removed by numerically controlled milling", *Computer-Aided Design*, v15 n3 1983.
- [6] Christiansen, H. and T. Sederberg. "Conversion of Complex Contour Line Definitions into Polygonal Element Mosaics", *Computer Graphics*, v12 n3 1978.
- [7] Ekoule, A., F. Peyrin, and C. Odet. "A Triangulation Algorithm from Arbitrary Shaped Multiple Planar Contours",
- [8] Frank, M., R. Wysk, S. Joshi. "Determining setup orientations from the visibility of slice geometry for rapid computer numerically controlled machining", *Journal of Manufacturing Science and Engineering, Transactions of the ASME*, v128 n1 February 2006.

- [9] Fuchs, H., Kedem, Z., Uselton, S. "Optimal Surface Reconstruction from Planar Contours", *Communications of the ACM*, v20 n10 October 1977.
- [10] Ganapathy, S. and T. Dennehy. "A New General Triangulation Method for Planar Contours", *Computer Graphics*, v16 n3 July 1982.
- [11] Huang, Y. and J. Oliver. "NC milling error assessment and tool path correction", *Proceedings of the ACM SIGGRAPH Conference on Computer Graphics*, 1994.
- [12] Jerard, R., R. Drysdale, K. Hauck, B. Schaudt, and J. Magewick. "Methods for detecting errors in numerically controlled machining of sculptured surfaces", *IEEE Computer Graphics and Applications*, v9 n1 January 1989.
- [13] Jütler, B. and M. G. Wagner. "Computer-Aided Design with Spatial Rational B-spline Motions", *Journal of Mechanical Design, Transactions of the ASME*, v118 n2 June 1996.
- [14] Keppel, E. "Approximating complex surfaces by triangulation of contour lines", *IBM Journal of Research and Development*, v19 n1 January 1975.
- [15] Lauwers, B., J.-P. Kruth, P. Dehonghe, R. Vreys. "Efficient NC-Programming of Multi-Axes Milling Machines through the Integration of Tool Path Generation and NC-Simulation", *CIRP Annals – Manufacturing Technology*, v49 n1 2000.
- [16] Oliver, J. and E. Goodman. "Color graphic verification of N/C milling programs for sculptured surface parts", *First Symposium on integrated intelligent manufacturing*, New York, ASME 1986.
- [17] Pal, P. "Remaining Stock Computation for 3D-Machining in Parametric Regime", *Journal of Manufacturing Science and Engineering*, v127 n4 November 2005.
- [18] Saito, T. and T. Takahashi. "NC Machining with G-buffer Method", *Computer Graphics*, v25, n4, July 1991.
- [19] Van Hook, T. "Real-Time Shaded NC Milling Display", *Computer Graphics*, v20 n4 August 1986.
- [20] Wang, Y. and J. Aggarwal. "Surface Reconstruction and Representation of 3-D Scenes", *Pattern Recognition*, v19 n3 1986.
- [21] Weinert, K., S. Du, P. Damm, M. Stautner. "Swept volume generation for the simulation of machining processes", *International Journal of Machine Tools & Manufacture*, v44 n6 May 2004.

Chapter 3: General Conclusion

A review of literature on rough machining reveals the popularity of using flat end mills in 2 ½ D tool paths to remove volumes of material. This method is typically used in three-axis roughing strategies. In multi-axis roughing, the literature has presented and validated the use of three-axis machine control from a plurality of setup orientations, rather than simultaneous control of four or five axes.

In this thesis, two primary problems are established for these multi-setup roughing strategies.

First, efficiency is paramount and the body of work in 2 ½ D roughing has shown that this is dependent on absolute knowledge of the stock boundary. However, this boundary is often ill-defined after the first few machining setups. In a further review of literature (Chapter 2), it is shown that there are no existing fast simulation techniques for evaluation of stock boundaries independent of exact tool path knowledge.

Second, robust tool paths are essential for multi-axis machining. Emergent structures in the form of thin webs and strings can cause tool or work piece failure rather than proper chip formation.

Review of Contribution

This thesis provides a unique and capable method of representing work piece stock in machining. By reducing the four-axis rough machining problem to a plurality of three-axis tool paths, the progress of machining can be modeled by planar polygons. While this critical assumption may seem limiting, it follows the same direction as existing research.

Representing a work piece as planar polygons allows efficient computation of changing volume (represented as area in each polygon). Just as these polygons are generated from a three-dimensional

polyhedron, they can be reconstructed into a polyhedron for use as stock models in commercial CAM software.

Application of this method results in significant reductions in tool path lengths (64% in one example) but requires only milliseconds of computation time. This low computation time makes this methodology fast enough for more intense applications, such as optimization or iterative simulation.

Future Research Directions

The method of stock representation developed in this research allows for a variety of future directions. Three promising areas of future research are discussed here.

Application of Accessibility

In this work, the problem of accessibility is circumvented by applying an overly conservative approach. The 2 ½ D convex hull of the component is considered the minimal reduction of stock material, though tool path generation is still driven by the true component surface. While this method is very fast and can be applied as a pre-processing step, its conservative nature limits the efficiency of tool paths.

An efficient means of accessibility analysis should be developed and applied to this methodology. While accessibility algorithms exist, their computational requirements would greatly slow the speed of this algorithm. Further, assumptions made in the correspondence and branching problems of polyhedral reconstruction (Chapter 2) would need to be re-evaluated if the 2 ½ D convex hull were replaced with a more elaborate approach.

Set-cover Analysis for Roughing Volume

The set of orientations used in this work is derived from visibility set cover analysis [2]. However, application of visibility polygons (see [3] for a review) to this polygonal stock representation would evaluate visible volume rather than visible surface. This type of analysis would be more useful for driving roughing orientations, where the objective is to remove as much volume as possible (and where visible surface area is less important).

An important area of research in this direction would be appropriately formulating the set cover problem and efficiently deriving data and metrics from the polygonal model.

Thin Web Detection

While existing research has investigated machining of thin webs and ribs incorporated in the design of components [1, 4, 5], there is no literature identifying or investigating *emergent* thin webs and strings (Figure 7, Figure 8). These emergent structures are undesirable and appear when rough machining from opposing setup orientations. An important aspect of effectively handling emergent structures is the ability to recognize them during process planning so that counter-measures can be taken. Shape analysis of the stock material polygons ($ReSt_{i,j}$) could identify emergent structures.

References

1. Agba, E.; D. Ishee; J. Berry. "High speed machining of unsupported thin-walled structures", *Society of Manufacturing Engineers*, 1999.
2. Frank, M.; R. Wysk; S. Joshi. "Determining setup orientations from the visibility of slice geometry for rapid computer numerically controlled machining", *Journal of Manufacturing Science and Engineering, Transactions of the ASME*, v128 n1 2006.

3. Heffernan, P.J.; J.S.B. Mitchell. "Optimal algorithm for computing visibility in the plane." *SIAM Journal on Computing*, v24 n1 p184-201 1991.
4. Ning, H.; W. Zhigang; J. Chengyu; Z. Bing. "Finite element method analysis and control stratagem for machining deformation of thin-walled components." *Journal of Materials Processing Technology*, v139 2003.
5. Smith, S.; D. Dvorak. "Tool path strategies for high speed milling aluminum workpieces with thin webs." *Mechatronics*, v8 n4 1998.