AUTOMATED PROCESS PLANNING FOR FIVE-AXIS POINT MILLING OF SCULPTURED SURFACES

GENG LIN

(B.Eng.)

A THESIS SUBMITTED

FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF MECHANICAL ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

2012

DECLARATION

I hereby declare that the thesis is my original work and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in the thesis.

This thesis has also not been submitted for any degree in any university previously.

GENG LIN

27/03/2013

Date

AKNOWLEDGEMENTS

I would like to express my sincere thanks to my supervisor A/Prof. ZHANG Yunfeng, for all the guidance, advice and discussion he has offered me in the past four years. It has been an inspiring and rewarding experience working under Prof. Zhang, from which I would benefit for the rest of my life.

I would also like to express my gratitude to Dr. LI Lingling and Dr. LI Haiyan for their excellent work which laid the foundation for my research and whose suggestions and advice has been invaluable to me.

Special thanks are given to Prof. WONG Yoke San and A/Prof. LU Wenfeng for their comments and suggestions on this thesis. I would also like to thank my fellow PH. D candidates and researchers for their lovely company and kind help.

Finally, I would thank my parents and my wife for their support, understanding and love throughout my PH. D candidature, without which I could never have come this far.

TABLE OF CONTENTS

SUMMARY

In this thesis, research efforts are presented for building an automated process planning system for 5-axis point milling of sculpture surfaces (finish cut) with optimized performance. As a continuation of our previous research, the proposed methods cover all three major tasks of process planning, i.e., accessibility evaluation, cutter(s) selection, and tool-path generation.

Firstly, as an extension of the accessibility evaluation algorithm developed in our previous research, a new representation scheme for the accessible posture range of a cutter at a surface point, called *posture chain*, is proposed. With this new formation, the accessible posture ranges at different surface points are constructed in the same global coordinate system and hence directly comparable. Methods are also developed to obtain the common accessible posture range at different surface points, which can be used for fast construction of accessible posture ranges through interpolation, thus alleviating the computational burden of the previously developed method.

Secondly, a novel method for tool-path length estimation for a given cutter and an accessible machining area is proposed to improve the existing multi-cutter selection algorithm. It makes use of neural network (NN) based on comprehensive data collection and system training. Compared with the existing heuristic based method, the NN-based method is able to achieve more accurate estimation, thus making the optimal multi-cutter selection more reliable.

Thirdly, for tool-path generation, methods are proposed to detect and eliminate possible machining interferences during the interpolation process between cutter locations (CLs). Such methods work as remedies for the existing heuristic-based posture assignment process, which may result in drastic posture changes between CLs.

The task is carried out by constructing the enveloping surface for the cutter's movement and conducting collision check between the enveloping surface and the workpiece. If interference occurs, the step-forward between CLs are accordingly adjusted. With the proposed method, the generated tool-paths are guaranteed to be interference-free, both at individual CLs and in-between.

Finally, evolutionary optimization methods are proposed for posture assignment to replace the existing heuristic-based method. Both tool-path smoothness and machining efficiency are considered in the objective function while interference avoidance and scallop height tolerance as constraints. Unlike the existing methods, the proposed method takes actual joint movements as the measurement of tool-path smoothness to eliminate drastic joint movements caused by the nonlinear kinematic structure of 5-axis machines. Two approaches are proposed for posture determination, based on the optimization tools of PSO and GA, respectively. Both approaches are proved effective with case studies and their pros-and-cons are analyzed.

The developed methods have been implemented as a system for generating finish-cut tool-paths with controlled smoothness and good machining efficiency. It represents an important step towards realizing automated and practical CAM system for 5-axis point-milling of complex sculptured surfaces.

LIST OF FIGURES

LIST OF TABLES

LIST OF GLOSSARY

λ: Inclination angle of a cutter posture at a surface point in the local frame

θ: Rotation angle of a cutter posture at a surface point in the local frame

R: Cutter major radius

r_f: Cutter torus radius

$\boldsymbol{S}_m$: Machining surface

$\boldsymbol{S}_{n,k}$; The kth non-machining surface

$\mathbf{C}_q$: Posture chain q

$\boldsymbol{x}_w$: Cutting direction

$\boldsymbol{f}$: Feeding direction

w_f, w_b: Forward and backward parts of a strip width as divided by a CC point

$J_{\max}$: Maximum amount of joint movement along a tool-path

τ: Surface finish tolerance

h: Tolerance for scallop height

$\mathbf{O}_G$-$\mathbf{X}_G\mathbf{Y}_G\mathbf{Z}_G$: Workpiece coordinate system

$\mathbf{O}_T$-$\mathbf{X}_T\mathbf{Y}_T\mathbf{Z}_T$: Cutter coordinate system

$\mathbf{O}_L$-$\mathbf{X}_L\mathbf{Y}_L\mathbf{Z}_L$: Local coordinate system at a surface point

A-map: Accessibility map at a surface point regarding a certain cutter

AR: Accessible region of a cutter on a machining surface

CAM: Computer-aided Manufacturing

CC point: Cutter Contact point

CC tool-path: Tool-path composing of CC points

CL tool-path: Tool-path composing of cutter center locations and cutter postures

CNC: Computer numerical Control

CSW: Cutting strip width

DT: Delaunay Triangulation

ESTP: Enveloping surface based tool-path

eAR: Effective Accessible Area

E-surface: Enveloping surface

GA: Genetic Algorithm

GC: Global Collision

LG: Local Gouging

MCP: Machining characteristic parameter

MR: Machining ratio

NN: Neural Network

NURBS: Non-Uniform Rational B-Spline

PTP: Preliminary tool-path

RG: Rear Gouging

CHAPTER 1

INTRODUCTION

In recent years, industries like aerospace, shipbuilding, automotive and even wood carpentering have seen a fast growing use of sculptured surfaces to meet functional or aesthetic needs. Such popularity stems from the excellent geometric properties of these surfaces, including high level of continuity, great flexibility for modeling free-form shapes and many more. However, these advantages come at a price. As sculptured surfaces usually lack an explicit mathematical description and have non-uniform curvature distribution, manufacturing of them turns out to be a difficult task.

Production of sculptured surfaces used to be carried out by skilful artisans or by using a trace mill based on a master model. This situation has changed with the advent of CNC (Computer Numerical Control) machining in the 1950s. Compared to the traditional method, CNC machining is an accurate, efficient, and reliable process. The movement of the cutting tool is controlled by NC commands, which eliminates the uncertainties of manual processes and achieves a high-level of automation. Another revolutionary change is brought about by the rise of various CAD (Computer-aided Design) and CAM (Computer-aided Manufacture) software packages. Together with CNC machining, they have combined design and production into an end-to-end process. Surface models are created in CAD software and then passed onto CAM software. CAM software analyzes the CAD model and generates cutter location (CL) data to produce the part. Based on the design of the CNC machine, the CL data is post-processed into NC commands that drive the machine to produce the desired surface shape. In this manner, the efficiency of the design-

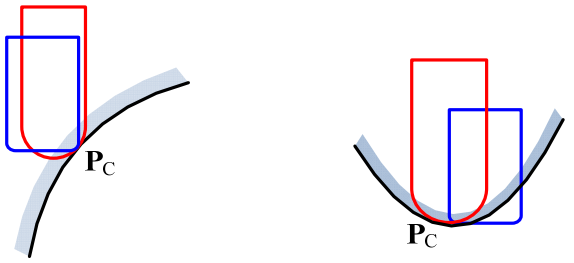
production cycle is greatly elevated.

The machining process for producing sculptured surfaces is end milling (Flank milling is only suitable for producing ruled surfaces and will not be discussed here). Traditionally, CNC machines for milling are only able to produce translational movements on the cutter in 3 orthogonal directions, hence the name 3-axis machining. Process planning for 3-axis machines is relatively easier as only CLs need to be provided. But this machining method lacks flexibility in face of complex surfaces, especially if the surface is blocked or partially blocked. Meanwhile, 5-axis machines have 2 more revolute joints than their 3-axis counterparts. Besides translational movements, they can also produce simultaneous rotational movement of the cutter around the 2 axes. Such a feature brings many advantages for 5-axis machining, such as set-up reduction, better surface finish, and higher machining efficiency. But process planning for 5-axis machining is inevitably complicated, as both CLs and cutter orientations (postures) need to be specified. Although process planning functions for 5-axis machining are already available in many CAM packages, they share some common shortcomings, such as heavy demand for human intervention and insufficient level of optimization.

In this thesis, efforts of the author to build an automated and optimized process planning system for 5-axis point milling (finish-cut) of sculptured surfaces are presented. As an introduction, this chapter covers the basics of 5-axis point milling of sculptured surfaces, followed by an introduction to automated process planning, including tasks, requirements and constraints. Further to that, by discussing the-state-of-art in commercial systems and published literatures, the motivation of this thesis is presented, followed by the detailed description of the research objectives and scope.

1.1. Five-axis Machining of Sculptured Surfaces

Traditionally, CNC machining of sculptured surfaces is carried out on 3-axis machines. The design of 3-axis machine is such that the orientation of the cutter is fixed in either a vertical or horizontal orientation. Relative movements between the cutter and the workpiece are produced by the three translational joints moving along 3 directions orthogonal to each other. Such a simple rule of movement effectively simplifies tool-path generation for 3-axis machining. Without any change in cutter orientation, the CL data for 3-axis point milling consist of a series of locations for the cutter center to trace.



Convex surface: both torus and ball-end cutters can be placed safely.

Concave surface: inability to adjusted posture rules out torus cutters

Figure 1.1 Limitations for use of torus cutters during 3-axis machining

However, the fixed cutter orientation in 3-axis milling also has some negative effects. For example, in a single set-up, only the area visible along the cuter axis can be machined. For surfaces that are blocked or partially blocked, multiple set-ups will be needed to complete the machining. Furthermore, the use of machining tools is limited for 3-axis machining. With the same major radius, flat-end and fillet-end (torus) cutters are preferred over ball-end cutters for their larger cutting profiles. However, for successful machining, the orientation of a flat-end or torus cutter should be adjusted at each individual surface point, which is beyond the capability of 3-axis machines. Using a fixed posture for the whole surface, especially for the concave regions, will cause mismatch of surface and cutter shapes, producing machining

interference (see Fig. 1.1). As a remedy, the relatively less efficient ball-end cutters become the dominant type used in 3-axis machining. This is because the effective cutting shape of a ball-end cutter is always a circular arc, invariant to the change in cutter posture. Such a feature reduces the process's vulnerability to interferences, especially local gouging. In summary, it can be seen that the inability of 3-axis machines to adjust cutter postures greatly limits the machining efficiency. Only for the so-called 'open' and conc sculptured surface without drastic changes in surface curvatures can 3-axis milling deliver satisfactory performance.

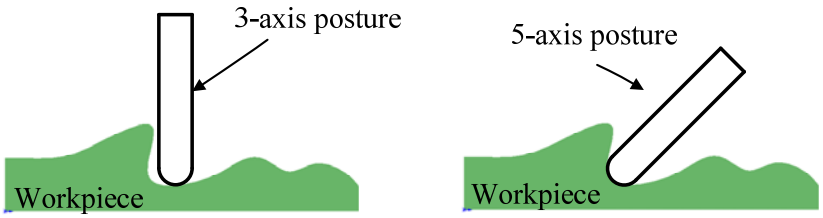


Figure 1.2 Accessibility comparison between 3-axis and 5-axis machining

Considering the limitation of 3-axis machines, 5-axis machines are introduced to provide enhanced cutter accessibility. In addition to the three translational joints, 5-axis machines are equipped with 2 extra revolute joints. This improved structure, combined with the advanced controller, makes it possible for 5-axis machines to conduct well coordinated, simultaneous movements on all 5 joints. In this manner, the cutter posture can be continuously adjusted during machining to deliver better performance. One advantage brought about by this increased flexibility is set-up reduction. As the cutter postures can be continuously adjusted, the accessible area on the machining surface is no longer limited to that visible in a certain direction (see Fig. 1.2), enabling a workpiece to be finished in fewer set-ups. This improvement not only saves processing time but also reduces or eliminates the inaccuracies produced by multiple set-ups. Meanwhile, 5-axis machining is also beneficial for machining

efficiency, as torus and flat-end cutters replace ball-end cutters as the dominant type used in 5-axis machining. The cutter posture can be adjusted continuously during machining, which produce cutting profiles that closely match the local surface shape at each individual point and therefore produce high material removal rate over the whole surface (see Fig. 1.3).

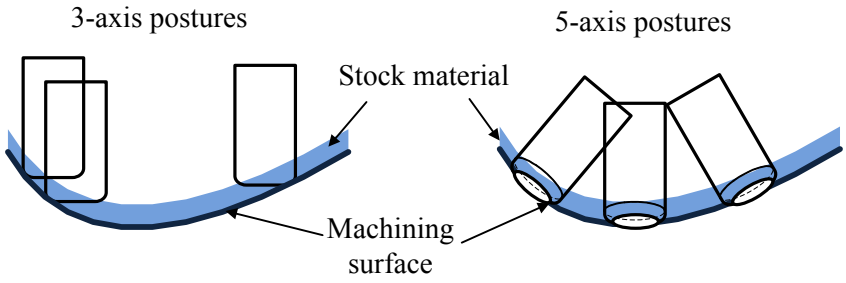


Figure 1.3 Material removal comparison between 3-axis and 5-axis machining

While 5-axis machining provides greater flexibility than 3-axis machining, process planning for 5-axis machining is inherently more complicated. The first limitation is imposed by cutter accessibility. During 5-axis machining, tool postures need to be specified for all CLs. However, due to the existence of machining interferences, not all cutter postures are feasible within the reachable space at a surface point. To ensure error-free machining, cumbersome and complicated algorithms should be implemented to detect and correct machining interferences. Further to this, during 5-axis machining, the movement of the cutter is complicated. Such movement should be carefully planned to conform to kinematic constraints as well as interference avoidance requirements. Finally, to get the most out of the huge potential of 5-axis machining, optimization should be carried out regarding all aspects of the process, such as cutter selection, cutting direction selection, and posture determination. Several interrelated yet not completely mutually-exclusive optimization objectives exist side by side for 5-axis machining (e.g., machining

efficiency and tool-path smoothness) while one optimization objective could be affected by several other selected parameters, e.g., machining efficiency is affected by cutter size, cutting direction, and cutter postures. Due to these factors, optimization of 5-axis process planning is a highly challenging task.

Nowadays, 5-axis machining centers and the accompanying software packages are becoming more and more affordable. The bottleneck limiting its wider application is the lack of satisfactory process planning systems. Existing CAM systems have various kinds of flaws, such as the need for significant human intervention, incomprehensive interference check, and lack of optimization. There is a need for an automated process planning system that overcomes these shortcomings and produce optimized manufacturing plans for point milling of sculptured surfaces.

1.2. Process Planning for Sculptured Surface Machining

Given a machining surface, a set of cutters and a specific 5-axis machining center, the process planning task is to (1) select one or several cutters, (2) allocate the *machining region* for each cutter involved (only applicable to the situation where multiple cutters are used) and (3) generate the tool-paths in the form of CL data and postures for each cutter/region. The basic requirements for the generated tool-paths are (1) the surface finish quality meets the specified shape error tolerance and (2) no machining interference occurs.

As shown in Fig. 1.4, process planning for 5-axis point milling (finish-cut) of sculptured surfaces can be divided into two stages, cutter selection and tool-path generation. Both stages can be divided further into several sub-stages. Each stage and sub-stage is subject to different constraints and optimization can be achieved according to different objectives.

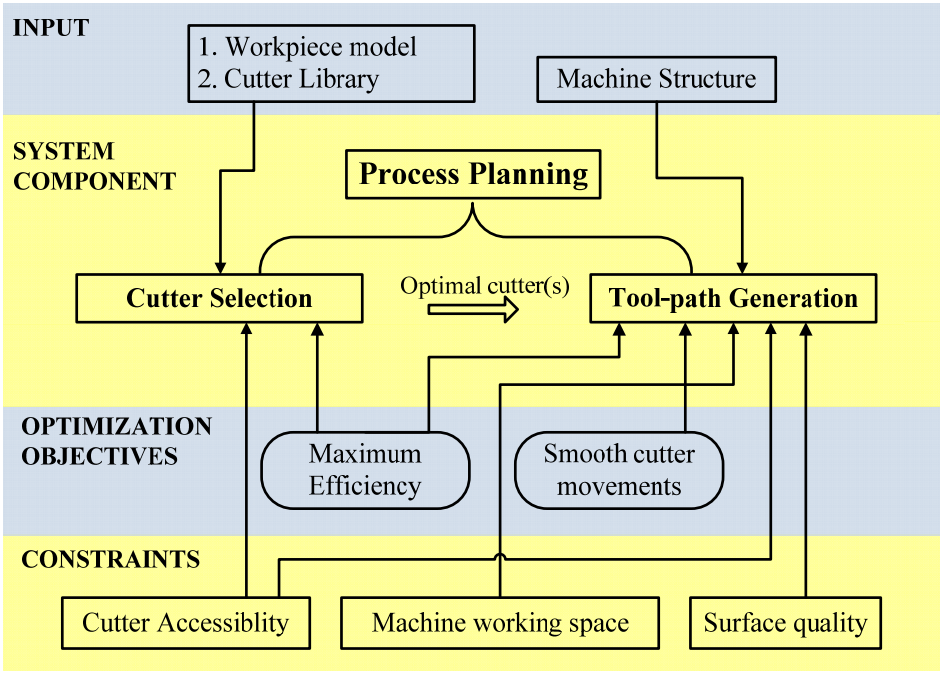


Figure 1.4 Process planning for 5-axis point milling of sculptured surfaces

Cutter selection starts with the accessibility evaluation for all the cutters in the cutter library. For a cutter at a point on the machining surface (called surface point), there is an accessible posture range inside which the cutter does not produce any interference with the workpiece. Basically, the cutter's size determines its accessibility. As the size of the cutter gets larger, the accessible posture range of the cutter shrinks. When the size of cutter reaches a certain level, there will be no posture with which the cutter can reach the target point without causing interference, meaning the cutter is *inaccessible* to the surface point. Accessibility evaluation for the cutter is of vital importance. For machining with a single cutter, the cutter must be accessible to all the points on the surface. For machining with multiple cutters, knowledge of the cutter's accessibility information is needed for allocation of machining regions to each cutter.

The actual cutter selection takes place after the accessibility of the all the cutters are obtained. The purpose is to select the cutter(s) with the highest machining

efficiency. For single cutter machining, this means to find the *largest accessible cutter*, as generally it is believed large cutters produce high material removal rates, hence the high machining efficiency. For multi-cutter machining, more complicated methods are needed to evaluate the performance of combinations of cutters regarding their machining efficiency.

Tool-path generation takes place after one or more cutters are selected for machining. For multi-cutter machining, since each selected cutter is already allocated with a specific region on the machining surface, the tool-path generation problem can be treated as generating tool-paths for every single cutter/region. For a cutter/region, the task of tool-path generation is two-fold, i.e. generation of cutter contact (CC) points and assignment of cutter posture to each CC point. CC points are surface points where the cutting edge meets tangentially with the machining surface. Normally, generation of CC points follows a certain path topology along a certain cutting direction. As CC points are to be traced by the cutter to produce the surface shape, their density and distribution greatly affects the surface finish quality. This is because the CC points are discrete. Interpolation will be conducted by the machine's controller to guide the cutter's movement between CC points. The deviation between the cutter trajectory and the machining surface should hence stay below the specified shape error tolerance.

The next crucial task for tool-path generation is the determination of cutter postures at all the CC points. The performance of 5-axis tool-paths is largely determined by this step. Interference avoidance is still the basic requirement for this process, which means the assigned cutter postures must come from the accessible posture ranges at the corresponding CC points. Under this constraint, cutter postures can be optimized regarding a series of objectives that affect the performance of tool-

paths, such as material removal rate and kinematic performance. Besides, as the kinematic performance of the tool-path is closely related to the structure and controller design of a specific machine, these factors should also be considered during the optimization process. After the cutter postures at all the CC points are determined, they will be combined with the CC point locations to form the CL data, which is the final product of process planning. The CL data will then be converted to NC codes (e.g., G-codes) that will drive the machine joints to finish the machining.

As introduced above, process planning for 5-axis machining is a complicated decision-making process that requires well-planned strategies, delicate heuristics, and extensive optimization efforts. Currently, most commercially available CAM software packages are inadequate in their level of automation and optimization. Common flaws include heavy reliance on human input and adoption of a trial-and-error approach. The user is often required to select cutter size, cutting direction, tool-path topology, and even cutter postures. When faced with such difficult tasks, users are often forced to make conservative selections due to lack of information. Cutter postures in available CAM systems are usually assigned by following a pre-set heuristic, and will be modified iteratively when interference is encountered. Most of the time, the heuristic is not developed for better tool-path performance, but for ease of computation, e.g., many CAM systems assign cutter postures (e.g., tilt angle) to have a fixed angle with the surface normal. These shortcomings greatly limit the performance of 5-axis machining while adding to the burden of users. Therefore, it is highly desirable and necessary to develop fully automatic process planning systems with optimized performance to make use of the full potential of 5-axis machining.

1.3. State-of-the-art in Process Planning for Sculptured Surface Machining

Process planning for 5-axis machining of sculptured surfaces has been studied extensively by many researchers since late 1980s. A number of reviews and surveys are available that effectively summarizes the solutions to the various issues of 5-axis machining. Choi and Jerard (1998) gives an extensive introduction of 5-axis sculptured surface machining from several aspects, such as fundamental mathematics, avoidance of machining interference, and principles of CNC machining. Lasemi *et al.* (2010) provides a detailed review of those recently proposed methods for 5-axis machining. The methods are grouped into several categories to cover different stages of process planning, such as tool-path topology, tool orientation identification, and tool geometry selection. Makhanov (2010) presents a survey on 5-axis machining optimization methods. Methods on optimizing cutter positions and orientations are summarized and reviews are also provided for different kinds of tool-path patterns.

In the following sections, a general review of recent research efforts on process planning for 5-axis machining is to be presented. The relative literature is grouped into several categories based on the areas of their focus.

- **Evaluation of cutter accessibility**

While cutter accessibility is greatly enhanced in 5-axis machining, the process is also more vulnerable to the harm of machining interferences. The consequences of machining interferences can be anything from beyond-tolerance surface error to a paralyzed machining center. Thus, for all process planning tasks that require knowledge of cutter posture, accessibility information of the cutter would play an essential role. Over the years, continuous efforts have been made by researchers to develop a reliable and comprehensive method to evaluate the accessibility of cutters for 5-axis machining. Among the different types of machining interferences, *local*

gouging is mainly detected by means of curvature analysis (Chen *et al.*, 2005; Chiou, 2004; Gong *et al.,* 2008; Wang and Tang, 2008). The other two kinds of machining interferences, i.e., *rear gouging* and *global collision*, are collisions between two rigid bodies in nature. Thus, detection methods for them usually utilize discretized representations of the workpiece, such as tessellated surface used by Li and Jerard (1994), triangular polyhedron by Kiswanto *et al.* (2007) and Xu *et al.* (2002), point cloud by Li and Zhang (2006) and bounding boxes by Ding *et al.* (2004). In terms of completeness, some researchers target at individual cutter postures and use the detection-correction approach to find a single interference-free posture, such as Lee (1997); Pi *et al.* (1998) and Wang and Yu (2002). Meanwhile, the other group of researchers aim at building the complete accessible posture range of a cutter at a surface point in various forms, including Balasubramaniam *et al.* (2003); Bi *et al.* (2011) and Jun *et al.* (2003). Apparently, the latter approach is more suitable for optimization of 5-axis tool-paths, as the accessible posture range would make up the feasible search space for cutter postures at the corresponding surface point.

- **Cutter Selection**

For cutter selection of sculptured surface machining, each selected cutter should be accessible everywhere within its allocated machining region. Under this premise, large cutters are preferred as they are believed to be able to produce better machining efficiency. For machining with a single cutter, this rule equals finding the largest accessible cutter for a machining surface. A series of trial-and-error methods are thus proposed by researchers like Jensen *et al.* (2002) and Lee and Chang (1996). In our previous work by Li and Zhang (2006), a cutter selection algorithm with accessibility evaluation for all the candidate cutters is developed. In this way, the optimal cutter can be easily identified.

For surfaces with more complex geometries (significantly different curvatures, machining obstacles, etc), it is more efficient to use multiple cutters for machining. Cutters of different sizes are assigned to different machining regions to produce better overall machining efficiency. The difficulties of this approach lie in allocation of machining regions and measuring the performance of different cutter combinations. Existing literature for this subject is quite limited. Elber (1995) presented an algorithm of surface decomposition that could be used for allocation of machining regions but with no algorithm for choosing cutter parameters. Yang and Han (1999) selected optimal cutter combination for 3-axis machining with minimized machining time by generating all the actual tool-paths. This trial-and-error approach is obviously not a practical approach for 5-axis machining. In our previous work, an heuristic based algorithm for multi-cutter selection of 5-axis machining was proposed by Li and Zhang (2009). The result of accessibility evaluation is utilized for machining region allocation. A method for predicting tool-path length is also proposed to select the optimal cutter combination. Details and improvements of this algorithm will be covered in section 3.

- **Tool-path generation: tool-path pattern**

Cutter contact (CC) points are the points on the machining surface for the cutter to trace during machining. According to Kim and Choi (2002), CC point generation lays the basis for the following steps in tool-path generation and thus would determine the overall performance of the resultant tool-paths. Normally, generation of the CC path should follow a certain pattern. Over the years, many tool-path patterns have been proposed by researchers, such as iso-parametric pattern, iso-planar pattern, and contour-pattern. Among them, the contour-pattern is mainly intended for workpiece with steep walls or islands, for which an example is provided

by Lee (2003). Iso-parametric pattern is the most straight-forward pattern. With this pattern, CC points are generated by fixing the value of *u* or *v* in the parametric domain (Elber and Cohen, 1994; Rao *et al.,* 2000). However, the control of surface quality becomes quite difficult, as the mapping between the Cartesian frame and the parametric frame is non-linear. For the iso-planar pattern, CC points are generated on the intersection curves between the machining surface and a series of parallel cutting planes. This method is simple and efficient with easy control of scallop height and thus widely used by many researchers such as Jensen and Anderson (1993); Li and Zhang (2006); Li and Jerard (1994) and Pi *et al.* (1998). In recent years, some untraditional patterns have been proposed, such as the iso-scallop pattern by Can and Ünüvar (2010); Li and Feng (2004); Tournier and Duc (2002), machining potential field method by Chiou and Lee (2002), and space filling curves pattern by Anotaipaiboon and Makhanov (2005). With these methods, CC points are selected only with regard to the forward-step and side-step requirements. Although proven to be effective under certain circumstances, such novel patterns are not widely used due to their high complexity or specialized purpose. Iso-planar pattern is still mostly used for 5-axis machining for its simplicity and robustness according to Lasemi *et al.* (2010).

- **Tool-path generation: cutter posture assignment**

Posture assignment is the most difficult yet most important task for process planning of 5-axis machining. This task is subject to many constraints, such as cutter accessibility and surface finish quality. Meanwhile, it can also be optimized regarding many objectives to improve the performance of 5-axis tool-paths. In early days, assignment of cutter postures follows a fixed heuristic. Although such heuristics do take machining efficiency into consideration, they are mainly developed for the sake

of easy computation. A typical approach by many researchers such as Warkentin *et al.* (2000) and Yuan-Shin (1997) is to keep the cutter posture inside the plane containing the cutting direction and the surface normal and keep it at a fixed angle from the surface normal (Sturz angle). Later, methods were proposed to optimize individual cutter postures to produce better machining efficiency. Machining interferences, mostly local and rear gouging, were also taken into consideration as the constraints. The basic idea behind such methods is to adjust the cutter posture within the accessible posture range so that the effective cutting shape can closely match the local surface shape to produce higher material removal rate. Examples include the works of Barakchi Fard and Feng (2009); Fard and Feng (2011); Yoon (2003); Yoon *et al.* (2003) and Zhong *et al.* (2002). It is worth noting that in these reported methods, the cutter posture is still adjusted within the plane determined by cutting direction and surface normal. Generally, the accessible posture range is not built and the optimal posture is often obtained with trial-and-error search.

Recently, the kinematic performance of the tool-paths has caught the attention of many researchers. Tool posture assignment methods are proposed to produce smooth cutter movements for higher machining efficiency. For this purpose, the accessible posture ranges are often obtained beforehand as the search space for cutter postures, as in the works of Bi *et al.* (2009); Bi *et al.* (2011); Ho *et al.* (2003); Jun *et al.* (2003) and Wang and Tang (2007). In these proposed methods, smoothness of the tool-path is measured by the angular difference between cutter postures. However, the CL paths must be transformed into NC commands before they can be executed on an actual 5-axis machine center. This means that the kinematic performance of tool-paths should be measured by joint movements in the machine frame, instead of posture difference in the workpiece frame. Although the structure of 5-axis machines has been

studied thoroughly by researchers such as She and Chang (2007) and She and Lee (2000), research on tool-path optimization in the machine frame is still quite limited. An example is the work of Castagnetti *et al.* (2008), in which the authors transformed the accessible posture range into the machine frame regarding the two revolute joints. Optimization is then carried out to select the postures such that the amount of joint movement is kept minimal.

1.4. Research Motivation

Process planning is the bottle neck limiting the wider application of 5-axis machining. Especially for the machining of freeform surfaces, available commercial CAM packages (e.g., MASTERCAM and HYPERMILL) still take a conservative approach. Most of them rely on the user for cutter selection, machining region allocation, and even for providing key parameters for tool-path generation, such as path intervals and cutting direction. Assignment of tool postures usually follows a simple heuristic while a trial-and-error approach is adopted to deal with possible machining interferences.

Meanwhile, in our previous research, an automatic process planning system for 5-axis machining is developed (Li, 2007). With an effective interference checking algorithm, the accessible posture range can be obtained at any point on the machining surface. With this algorithm, selection of single and multiple cutters can be carried out based on the cutters' accessibility information over the whole surface. For tool-path generation, a set of heuristics have been developed to generate the tool-paths aiming at maximized machining efficiency while meeting the requirement for surface finish quality. The process planning system was extensively tested with case studies and proved to have achieved a certain level of robustness and efficiency.

However, the current process planning system still has great potential for improvement: many steps in the system can be optimized to have lower computational

cost or to produce results with better quality. Many factors should be taken into consideration to produce tool-paths with superior performance and ease of implementation. Different modes of process planning should be provided to cater for the user's specific needs. The system should be modified to cover actual machining with NC code. As an extension of our previous research, the work presented in this thesis includes a series of optimization efforts for each stage in the existing system. The target is to improve the existing system in terms of efficiency, reliability, optimality, and versatility.

1.5. Research Objectives and Scope

The purpose of the research proposed in this thesis is to develop an optimized process planning system for 5-axis point milling of sculptured surfaces in the finish-cut stage. The processes subject to optimization are accessibility evaluation, cutter selection, and tool-path generation. New methods are to be proposed for these tasks to achieve lower computational cost and better machining performance. To this end, three specific objectives are set as follows:

1) To develop a fast algorithm for identifying accessible posture range, based on the existing algorithm proposed in our previous research.
2) Develop a new measurement for the performance of multi-cutter set cutter and effective algorithms for multi-cutter selection.
3) Develop new algorithms for tool-path generation. The performance of the tool-paths should be optimized in respect to different objectives.

On the other hand, the research scope has been recognized as follows.

For the 1st objective, the research scope covers:

- Develop a new representation scheme for accessible posture range that is invariant to transformation, as the current representation named *Accessibility-map* or *A-map* is confined to the local coordinate system at the target point.
- Design algorithms that can produce a reliable and quick approximation of the A-map at a surface point based on existing A-maps of the sampled points within a certain neighborhood of the target point.

For the 2nd objective, the research scope covers:

- Develop new criteria for evaluating the performance of multi-cutter sets without generating actual tool-paths.
- Develop methods to extract useful data regarding the above mentioned criteria from the multi-cutter sets and machining surface.
- Develop algorithms to process the extracted data and select the optimal multi-cutter set with acceptable accuracy.

For the 3rd objective, the research scope covers:

- Propose methods to detect interferences during the interpolation process between CC points and develop methods to eliminate such interferences.
- Modify the algorithm for generating the CC point locations to accommodate the cutter postures assigned by an optimization algorithm instead of the existing heuristic.
- Develop methods to evaluate the kinematic performance of tool-path regarding joint movements.
- Develop multi-objective optimization methods to optimize the posture assignment process regarding multiple optimization objectives, i.e., machining efficiency and tool-path smoothness.

1.6. Organization of the Thesis

In the remaining of this thesis, each chapter will cover an individual topic of process planning. Each topic is approached in a self-contained manner containing the following steps: introduction of background and relevant literature, specification of inputs and outputs, illustration of methodology, case studies and discussion. A brief overview is provided as follows.

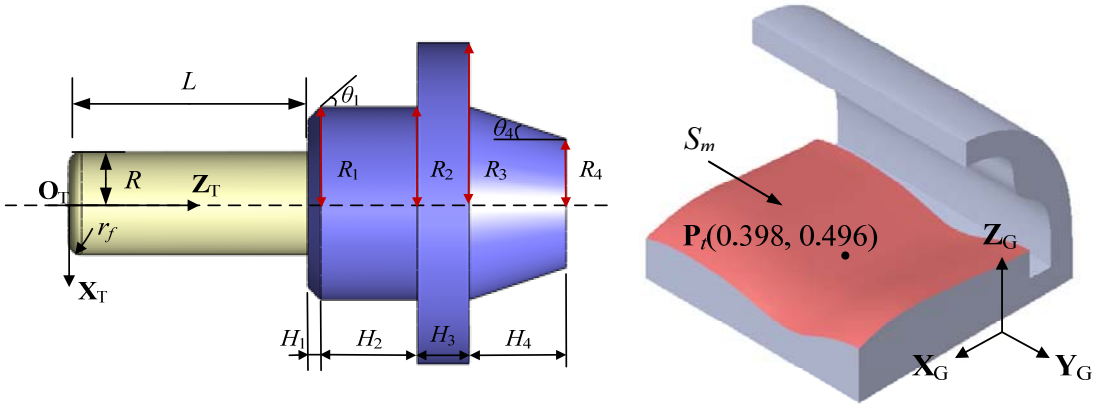
Chapter 2 presents the newly improved cutter accessibility evaluation algorithm. The existing A-map algorithm is briefly introduced first. A new representation scheme of A-map that is invariant to transformation is proposed, which is useful for providing a quick and reliable approximation for the accessible posture range at a surface point. Chapter 3 deals with the cutter selection process. Existing methods for single and multi-cutter selection, together with their problems and flaws, are introduced first. A novel criterion for evaluating the performance of a multi-cutter set is then proposed. As the evaluation requires the tool-path length for a specific cutter, a prediction method based on neural networks is developed. Chapter 4 provides the complete workflow of iso-planar tool-path generation. The existing pro-efficiency heuristic for posture assignment is also introduced in this chapter. In Chapter 5, as an improvement for the previous tool-path generation algorithm, a method to detect and correct possible collision during the interpolation process between CC points is proposed. First, the enveloping profiles of the cutter's movement during interpolation are identified, which are taken to build the enveloping surface of the cutter's movement. Collision query is then conducted between the enveloping surface and the workpiece. Such collision, if there is any, is then eliminated by increasing the density of CC points. In Chapter 6, optimization algorithms for posture determination are introduced. The kinematic structure of 5-axis machines is studied to obtain joint

movements from CL data. Then the posture determination task is modeled as a constrained multi-objective optimization problem. Evolutionary computation algorithms such as PSO and GA are used to solve this optimization problem. The target is to produce tool-paths with good performance regarding both machining efficiency and tool-path smoothness. Application examples for these methods are provided in Chapter 7. Chapter 8 draws the conclusions by discussing the achievements and limitations of the research, as well as direction of possible future work.

CHAPTER 2

CUTTER ACCESSIBILITY EVALUATION

Process planning for 5-axis point milling of sculptured surfaces is complicated, largely due to the process's vulnerability to machining interferences, which may cause irreversible damages to the workpiece, the cutter, and the machine. Therefore, evaluation of a cutter's accessibility is of highest priority in all process planning systems. The important tasks, such as cutter selection and tool-path generation, rely heavily on information obtained in this process. For 5-axis point milling, accessibility of a cutter should be evaluated at an individual surface point. 'Accessible' is synonym for 'interference-free' here, which means at the target surface point, there is at least one cutter posture with which the cutter does not produce any interference with the workpiece.



(a) Cutter and holder model (b) Workpiece model

Figure 2.1 Inputs models for accessibility evaluation

The input to accessibility evaluation is the information of the cutter and the workpiece. A general description of the cutter and workpiece models to be used in this study are shown in Fig. 2.1. The cutter is modeled as a cylinder with a torus end,

which is described by its major radius R, minor radius r_f, and length L. The cutter's holder is modeled as a series segments, i.e., cylinders and/or truncated cones. A segment is described by its height H, base radius R and angle θ (if it is not zero). The tool coordinate system $\mathbf{O}_T$-$\mathbf{X}_T\mathbf{Y}_T\mathbf{Z}_T$ is assigned to the cutter with $\mathbf{O}_T$ coincident with the cutter bottom center and $\mathbf{Z}_T$ along the cutter axis as shown in Fig. 2.1a. It is assumed that the length of the cutter plus the holder would be enough to eliminate the possibility of collision between the spindle or other moving parts on the machine and the workpiece. Thus they are not considered in this model. The workpiece (part) model is described by its bounding surfaces, categorized as machining surface $\boldsymbol{S}_m$ and non-machining surfaces $\{\boldsymbol{S}_{n1}, \boldsymbol{S}_{n2} \ldots, \boldsymbol{S}_{nk}\}$ in the form of NURBS surface patches with the workpiece coordinate system $\mathbf{O}_G$-$\mathbf{X}_G\mathbf{Y}_G\mathbf{Z}_G$. An arbitrary surface point $\mathbf{P}$ from $\boldsymbol{S}_m$ can be located using the parametric form of the surface with the two parameters u and v (see Fig 2.1b).

2.1. Background

The purpose of accessibility evaluation is to identify the range of postures with which a cylindrical cutter can access a specific surface point on $\boldsymbol{S}_m$ without causing any interference. The first limit for the search of accessible postures is imposed by the design of cylindrical cutting tools. For point milling, the torus portion serves as the cutting edge (the hemisphere portion for ball-end cutters, and the bottom circle for flat-end cutters). Meanwhile, to avoid gouging, the cutting edge is supposed to be tangent to the machining surface ($\boldsymbol{S}_m$) at the cutter contact point ($\mathbf{P}_C$ in Fig. 2.2). Considering these two requirements, the allowable posture range is limited to the hemisphere centered on the surface normal at $\mathbf{P}_C$. Besides, an accessible posture should be within the workspace of a specific 5-axis machine (given in the machine specification), which puts another boundary on the allowable posture range. The

remaining constraints are imposed by the requirement for interference avoidance between the cutter and the part.

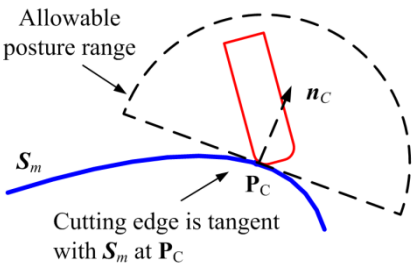


Figure 2.2 Posture range determined by cutter geometries

Generally, there are three kinds of machining interferences for 5-axis machining, i.e., local gouging, rear gouging, and global collision. When the cutter is placed at $\mathbf{P}_C$, local gouging occurs when the curvature of the cutter is smaller than the minimum principal curvature at $\mathbf{P}_C$ such that the cutter cuts off excess material beyond tolerance. Rear gouging refers to the situation when the bottom of the cutter (not intended for cutting) protrudes into the machining surface and removes material. Global collision occurs when the shank or holder of the cutter collides with the machining or non-machining surfaces on the workpiece. A simple illustration of the mechanisms for all three kinds of machining interferences is provided in Fig. 2.3. An accessible cutter posture should be free from all these 3 types of machining interferences. The set of accessible postures within the allowable posture range makes up the accessible posture range of the cutter at the surface point.

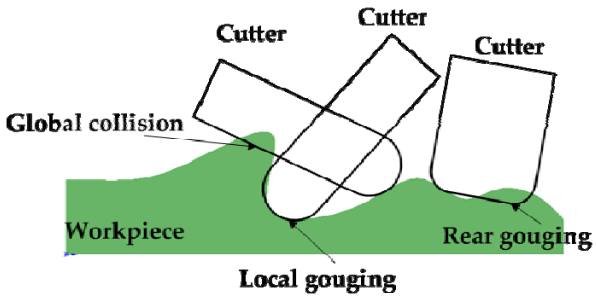


Figure 2.3 Mechanisms for machining interferences in 5-axis machining

2.2. Related Works

Over the years, many accessibility checking methods for cylindrical cutters during 5-axis point milling of free-form surfaces have been proposed. Based on purpose and usage, the proposed methods fall into two categories. The first category aims at judging whether interference exists for a specific cutter posture or identifying a single interference-free cutter posture. Such algorithms can be referred to as 'single posture methods' and are often used with a trial-and-error approach for posture determination during tool-path generation or modifying problematic tool-paths with interferences. A typical example of this category is the work of Chiou and Lee (2005), in which the authors identified the profile curves on the sweeping volume produced by the cutter posture along a certain cutting direction. Intersections between the profile curves and the workpiece model are then searched for to determine whether the posture in question is accessible. The rolling ball method proposed by Gray *et al.* (2003) is another example in this category. The local surface shape is approximated by a spherical surface, whose radius is determined by the point with the most critical curvature under the cutter. Then the cutter is oriented in such a posture that the cutter bottom surface rests on the sphere. Also included in this category are the arc-intersect method proposed by Gray *et al.* (2005) and the penetration elimination method proposed by Hosseinkhani *et al.* (2007).

The second category, on the other hand, takes a more comprehensive approach by trying to find the interference-free posture range for a cutter at a surface point, hence named 'posture range methods'. The obtained accessible posture ranges contain all the accessible postures of the cutter at the target point, thus can be used as search spaces for cutter postures during tool-path optimization. Some early works of this category is based on the idea of matched curvature machining, in which the authors

attempted to find a condition for a cutter to be free of local gouging by comparing the curvature of the cutter and the workpiece surfaces in every possible direction. Works of this category include those of Rao and Sarma (2000) and Yoon *et al.* (2003). However, since only local gouging avoidance is considered, the obtained posture range can only be used on the so called 'open' free-form surfaces. To build a complete interference-free posture range at a surface point, many researchers proposed their methods based on the idea of configuration space (C-space), including Jun *et al.* (2003); Morishige *et al.* (1997) and Morishige *et al.* (1999). C-space is a description of the solution space for cutter postures. The obstacles that the cutter should avoid during machining can also be mapped into this space. Then the space not occupied by obstacles would be collision-free. Although the idea is quite straight-forward, a major difficulty of the C-space approach is the computational intractability when mapping the obstacles into the C-space. Another group of researchers, e.g. Balasubramaniam *et al.* (2003), tried to evaluate cutter accessibility as the visibility of the target point by treating cutter as a ray of light. An advantage of this approach is that the computation of visibility cones can be accelerated by the usage of graphic hardware. However, visibility is only the necessary but not sufficient condition for accessibility, making extra measures necessary to take cutter radius as well as possibilities of gouging into consideration according to Wang *et al.* (2007). Further to this, all the posture range methods generally suffer from heavy computational load, which is why single posture methods are still quite useful and cannot be discarded.

In this chapter, an accessibility evaluation method that falls into the second category developed in our previous research will be introduced. Several improvements designed to achieve ease of handling and reduce computation load will

be presented. Such improvements are of great importance for following process planning tasks, such as tool-path generation and optimization.

2.3. A-maps - Finding a Cutter's Accessible Posture Range

In our previous work, an accessibility evaluation method has been developed. The method takes a cylindrical cutter, the workpiece and the target point on a sculptured surface as inputs. The output is the interference-free posture range at the point, which is named *accessibility map* (A-map) of the cutter at the surface point. For completeness, the existing A-map algorithm will be introduced briefly in this section before improvements and extensions of the algorithm are presented.

The basic idea behind the A-map algorithm is to identify the three interference-free posture ranges regarding each type of machining interference for a specific cutter at a surface point. The intersection of these 3 ranges plus the range determined by the joint travel limits of 5-axis machines will make up the accessible posture range of the cutter.

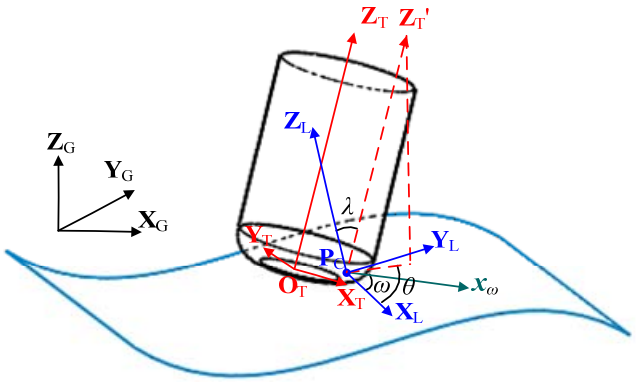


Figure 2.4 The local and tool frames for accessibility analysis

In the proposed method, a local frame, denoted as $\mathbf{O}_L$-$\mathbf{X}_L\mathbf{Y}_L\mathbf{Z}_L$ is established at the target point $\mathbf{P}_C$ as shown in Fig. 2.4. The origin $\mathbf{O}_L$ is coincident with $\mathbf{P}_C$; $\mathbf{Z}_L$ is coincident with the surface normal at $\mathbf{P}_C$; $\mathbf{Y}_L$ and $\mathbf{X}_L$ are along the maximum and

minimum principal directions at $\mathbf{P}_C$, respectively. Meanwhile, the tool frame $\mathbf{O}_T$-$\mathbf{X}_T\mathbf{Y}_T\mathbf{Z}_T$ introduced in Fig. 2.1 is also used here ($\mathbf{O}_T$ coincident with the center of the cutter bottom, $\mathbf{Z}_T$ along the cutter axis, $\mathbf{X}_T$ in the plane determined by $\mathbf{P}_C$ and $\mathbf{Z}_T$, pointing towards $\mathbf{P}_C$). With $\mathbf{O}_L$-$\mathbf{X}_L\mathbf{Y}_L\mathbf{Z}_L$ and $\mathbf{O}_T$-$\mathbf{X}_T\mathbf{Y}_T\mathbf{Z}_T$, a cutter's posture can be defined by an angle pair (λ, θ). λ, known as the inclination angle, is defined as the angle between $\mathbf{Z}_L$ and $\mathbf{Z}_T$ (or $\mathbf{Z}_T$', which passes through $\mathbf{P}_C$ and is parallel to $\mathbf{Z}_T$). θ is the rotational angle that the cutter rotates about $\mathbf{Z}_L$, which equals the angle between $\mathbf{X}_L$ and the projection of $\mathbf{Z}_T$' on plane $\mathbf{X}_L$-$\mathbf{Y}_L$.

2.3.1. Accessible range regarding local gouging

At $\mathbf{P}_C$, local gouging (LG) occurs when the curvature of the cutter surface is smaller than that of the machining surface. Therefore, given a posture (λ, θ), the normal curvatures of the cutter and the machining surface in every possible direction need to be compared to detect LG. Suppose $\boldsymbol{x}_\omega$ is a unit vector on $\mathbf{X}_L$-$\mathbf{Y}_L$ (the surface's tangent plane at $\mathbf{P}_C$). The angle between $\mathbf{X}_L$ and $\boldsymbol{x}_\omega$ is given by ω ($0 \le \omega < 2\pi$). As shown in Fig. 2.5, the curvature of the machining surface at $\mathbf{P}_C$ on plane $\boldsymbol{x}_\omega$-$\mathbf{Z}_L$ is:

$$\kappa_{s\omega} = \kappa_{\max} \cos^2 \omega + \kappa_{\min} \sin^2 \omega \tag{2.1}$$

where $\kappa_{\max}$ and $\kappa_{\min}$ are the maximum and minimum principal curvatures of the machining surface at $\mathbf{P}_C$, respectively. On the same plane, the curvature of the cutter is given as (Jensen *et al.,* 2002):

$$\kappa_{t\omega} = \frac{\cos^2(\omega-\theta)}{r_f} + \frac{\sin^2(\omega-\theta)}{\frac{(R-r_f)}{\sin\lambda} + r_f} \tag{2.2}$$

To make sure the cutter is LG-free, for any ω, $\kappa_{t\omega} - \kappa_{s\omega} > 0$. Combining Eq. (2.1) and Eq. (2.2), the following 2 inequalities that are independent of ω can be obtained:

$$\sin\lambda > \frac{r_1(r_f\kappa_{\max} - \cos^2\theta)}{r_f(1 - r_f\kappa_{\max})} \tag{2.3}$$

$$\sin\lambda > \frac{r_1\kappa_{\max}(1 - r_f\kappa_{\min}) - r_1(\kappa_{\max} - \kappa_{\min})\cos^2\theta}{(1 - r_f\kappa_{\min})(1 - r_f\kappa_{\max})} \tag{2.4}$$

where $r_1 = R\text{-}r_f$. Given a value of θ, 2 minimum values, $\lambda_{min\text{-}1}$ and $\lambda_{min\text{-}2}$, if there is any, can be obtained. The accessible range is $[\theta, (\lambda_{\theta\text{-}lg}, 90°)]$, where $\lambda_{\theta\text{-}lg} = \max(\lambda_{min\text{-}1}, \lambda_{min\text{-}2})$.

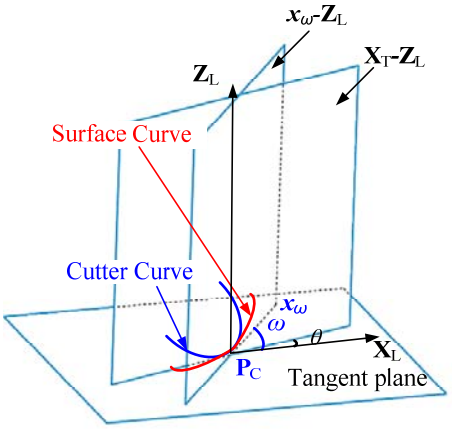
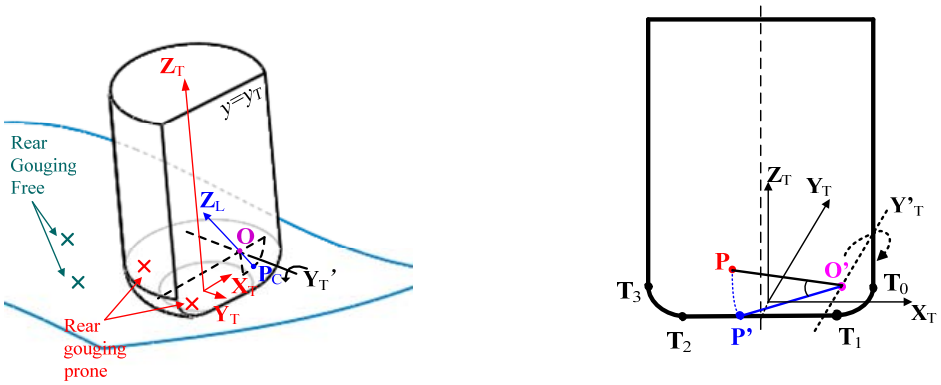


Figure 2.5 Comparison of cutter and surface curvatures on $\boldsymbol{x}_\omega$-$\mathbf{Z}_L$

2.3.2. Accessible range regarding rear gouging

At $\mathbf{P}_C$, rear gouging (RG) refers to the intrusion of the cutter bottom surface into the part surface. The detection of RG employs a discrete approach. Firstly, the machining surface is sampled into a high-density point set. For each value of θ, regarding each sampled point $\mathbf{P}i$ (i = 1, 2,…, n), an accessible range ($\lambda_{\theta\text{-rg1-}i}$, $\lambda_{\theta\text{-rg2-}i}$) can be obtained at $\mathbf{P}_C$. The intersection of all the ($\lambda_{\theta\text{-rg1-}i}$, $\lambda_{\theta\text{-rg2-}i}$) ($i$ = 1, 2, …, n) is taken as the ($\lambda_{\theta\text{-rg1}}$, $\lambda_{\theta\text{-rg2}}$), which gives RG-free range of λ at θ. To reduce computation load, the RG-free range is not built at every sampled point, but at the *RG-prone* points only. As shown in Fig. 2.6a, for a given θ, when λ changes, the cutter rotates around $\mathbf{Y}_T$' with point **O** as the pivot point (**O** is the point along surface normal with distance r_f from $\mathbf{P}_C$; $\mathbf{Y}_T$' passes through **O** and is parallel with $\mathbf{Y}_T$). Based on this observation, the following 3 conditions are established for a sampled point $\mathbf{P}(x_T, y_T, z_T)$ to be RG-prone (note that

the point coordinate is expressed in $\mathbf{O}_T$-$\mathbf{X}_T\mathbf{Y}_T\mathbf{Z}_T$): 1) $|\mathbf{OP}| \le 2R - r_f$; 2) $\mathbf{P}_C\mathbf{P}\cdot\mathbf{Z}_L > 0$; 3) $-R \le y_T \le R$.



(a) Gouging prone point and the cutter (b) Section curve on the cutter

Figure 2.6 Identify accessible posture range regarding rear-gouging

As shown in Fig. 2.6b, when $\lambda = 0$ and plane $y=y_T$ is used to section the cutter, a section curve of 3 segments is produced on the cutter bottom: two arcs $\mathbf{T}_0\mathbf{T}_1$ and $\mathbf{T}_2\mathbf{T}_3$ (torus part) and one horizontal line $\mathbf{T}_1\mathbf{T}_2$ (bottom). If **P** is above the section curve, RG occurs. If we increase λ by rotating the cutter about $\mathbf{Y'}_T$, **P** tends to move towards underneath the section curve. Therefore, we need to find the minimum λ to move **P** to the cutter's outer surface at position **P**'. Depending on the segment **P**' falls onto, calculation of the increment $\Delta\lambda$ that moves **P** to **P**' is different. We define d as the distance between **P** and $\mathbf{Y'}_T$, and use d_0, d_1, d_2, and d_3 to represent the distances from $\mathbf{T}_0$, $\mathbf{T}_1$, $\mathbf{T}_2$, and $\mathbf{T}_3$ to $\mathbf{Y'}_T$, respectively. The calculation of $\Delta\lambda$ is given as:

(1) When $d_1 \le d \le d_2$, **P**' falls between $\mathbf{T}_1$ and $\mathbf{T}_2$, and its coordinates in the tool frame are given as $\mathbf{P}'(x_T, y_T, z_T) = \left(r_1 - \sqrt{d^2 - r_f^2}, y_T, 0\right)$ and $\Delta\lambda$ is calculated as:

$$\Delta\lambda = \cos^{-1}\left(\frac{r_f - z_T}{d}\right) - \cos^{-1}\left(\frac{r_f}{d}\right) \tag{2.5}$$

(2) When $d_0 \le d \le d_1$ or $d_2 \le d \le d_3$, **P**' falls between $\mathbf{T}_0$ and $\mathbf{T}_1$, $\mathbf{T}_2$ and $\mathbf{T}_3$, respectively. Its coordinates in the tool frame is given as:

$$\mathbf{P}'(x_T, y_T, z_T) = \left(a + \frac{a^2 + {y_T}^2 - r_1^2}{2(r_1 - a)}, y_T, r_f \left(1 - \sqrt{1 - \left(\frac{a^2 + {y_T}^2 - r_1^2}{2r_f (r_1 - a)} \right)^2} \right) \right) \quad (2.6)$$

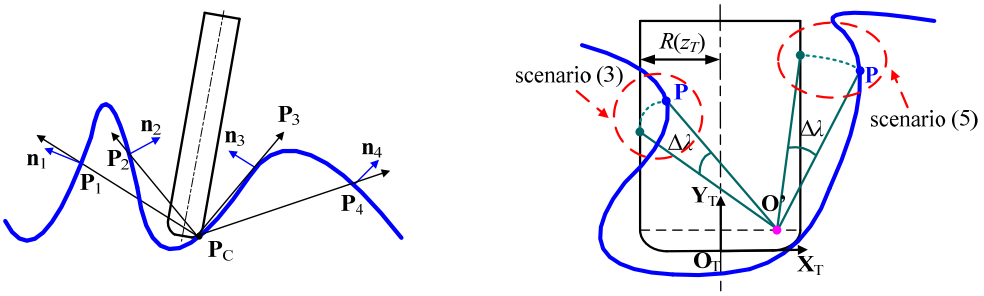
where $a = \dfrac{2r_1^2 + {r_f}^2 - d^2 - {y_T}^2}{2r}$. $\Delta\lambda$ is then calculated as:

$$\Delta\lambda = \sin^{-1}\left(\frac{z_T - r_f}{d} \right) + \sin^{-1}\left(\frac{r_f - {z'}_T}{d} \right) \quad (2.7)$$

The RG-free posture range of $\mathbf{P}_C$ at θ is [$\Delta\lambda$, 90º]. After applying the algorithm to all the RG-prone points, the accessible region for rear gouging avoidance at θ is [θ, ($\lambda_{\theta\text{-rg}}$, 90°)], where $\lambda_{\theta\text{-rg}} = \max\{\Delta\lambda_i | i=1,2,\ldots, n\}$ (n is the number of RG-prone points).

2.3.3. Accessible range regarding global collision

Global collision (GC) occurs when the non-cutting parts of the cutter (cutter shaft, holder) intersect with the machining surface or the non-machining surfaces. To find the GC-free posture range, a similar approach as that used with RG is taken. At $\mathbf{P}_C$ and a fixed θ, let the cutter take the posture of (θ, λ_0=0°). Meanwhile, let **n** be the surface normal at an arbitrary point **P**(x_T, y_T, z_T) on the machining/non-machining surface. For **P** to be a candidate point for GC, the following condition should be met, $\mathbf{n} \bullet \mathbf{P}_C\mathbf{P} < 0$ (see Fig. 2.7a).



(a) Collision-free and collision- prone points (b) Section curve on the cutter

Figure 2.7 Rear-gouging free posture range

Similarly, we use plane $y = y_T$ to section the cutter. There are 2 possible scenarios: 1) **P** falls inside the section curve of the cutter and GC exists for posture (θ,

λ_0); 2) **P** falls outside the section curve and posture (θ, λ_0) is GC-free. Unlike the cases for LG and RG, the upper limit of λ regarding global collision is no longer 90º. So we need to find both the minimum $\Delta\lambda$ to rotate the cutter away from global collision (1st condition) and the minimum $\Delta\lambda$ to induce global collision (2nd condition). Accordingly, the relative positional relationship between **P** and the section curve can be categorized into the following 5 scenarios:

(1) $z_T < r_f$. Cutter is GC-free with **P** and the accessible range of λ is [0°(λ_0), 90°].

(2) $x_T < -R(z_T)$, and $\mathbf{n} \bullet \boldsymbol{X}_T > 0$ ($z_T \geq r_f$). Cutter is GC-free with **P** and the accessible range of λ is [0°, 90°].

(3) $x_T \geq -R(z_T)$, and $\mathbf{n} \bullet \boldsymbol{X}_T > 0$ ($z_T \geq r_f$), GC exists (see Fig. 2.7b). The minimum $\Delta\lambda$ that the cutter must be rotated clockwise to avoid GC is:

$$\Delta\lambda = \cos^{-1}\frac{r_1 - x_T}{d} - \cos^{-1}\frac{r_1 + \sqrt{R(z_T)^2 - {y_T}^2}}{d} \tag{2.8}$$

where d is the distance from **P** to **O'**. The accessible range of λ is [$\Delta\lambda$, 90°].

(4) $x_T < r$ and $\mathbf{n} \bullet \boldsymbol{X}_T < 0$ ($z_T \geq r_f$), the accessible range of λ is NULL.

(5) $x_T \geq r$, and $\mathbf{n} \bullet \boldsymbol{X}_T < 0$ ($z_T \geq r_f$), **P** is GC-free (see Fig. 2.7b). The minimum $\Delta\lambda$ that the cutter must be rotated clockwise to cause global-collision is given as:

$$\Delta\lambda = \cos^{-1}\frac{r_1 - x_T}{d} - \cos^{-1}\frac{r_1 + \sqrt{R(z_T)^2 - {y_T}^2}}{d} \tag{2.9}$$

The accessible range of λ is [0, $\Delta\lambda$].

Using this method, the accessible ranges for all GC-prone points can be obtained as $[\lambda_{\theta\text{-gc1-}i}, \lambda_{\theta\text{-gc2-}i}] \mid i = 1, 2, \ldots, n$. The overall GC-free posture range is $[\theta, (\lambda_{\theta\text{-gc1}}, \lambda_{\theta\text{-gc2}})]$, where $\lambda_{\theta\text{-gc1}} = \max\{\lambda_{\theta\text{-gc1-}i} | i = 1, 2, \ldots, n\}$ and $\lambda_{\theta\text{-gc2}} = \min\{\lambda_{\theta\text{-gc2-}i} | i = 1, 2, \ldots, n\}$. However, if $\lambda_{\theta\text{-gc1}} > \lambda_{\theta\text{-gc2}}$, the accessible range for the cutter regarding θ is null.

2.3.4. Construction of A-map

Using the methods introduced in sections 2.3.1-2.3.3, for a certain θ, the accessible range of λ at $\mathbf{P}_C$ can be obtained as the intersection of the 3 accessible ranges. Based on this method, θ is sampled evenly in the range of $[\theta_{min}, \theta_{max}]$ and the feasible range of λ at every discrete value of θ can be obtained. In this manner, a numerical approximation of the accessible posture range at $\mathbf{P}_C$, i.e., the A-map of the cutter at $\mathbf{P}_C$, can be obtained. The complete algorithm for obtaining the A-map is given as follows.

Algorithm: Finding the A-map of a cutter at a CC point P_C

Input: *(a) Sampled point set* $\{\boldsymbol{P}_k, k = 1, 2, \ldots, m\}$ *and CC point* $\mathbf{P}_C$

(b) A torus end miller (R, r_f, L)

(c) Titling angle range $[\lambda_{min}, \lambda_{max}]$*, rotational angle range* $[\theta_{min}, \theta_{max}]$

Output: *A-map and the accessibility of the cutter at* $\mathbf{P}_C$

Begin

(1) Uniformly sample $(\theta_{min}, \theta_{max})$ into k angles, set $i = 0$.

(2) IF $i \le (k\text{-}1)$, $\theta_i = \theta_{min} + (\theta_{max} - \theta_{min})(i/(k\text{-}1))$; otherwise, go to (6).

(3) Find the LG-free range $[\theta_i, (\lambda_{\theta\text{-lg}}, \lambda_{max})]$. If the accessible range is NULL, $i = i + 1$, go to (2).

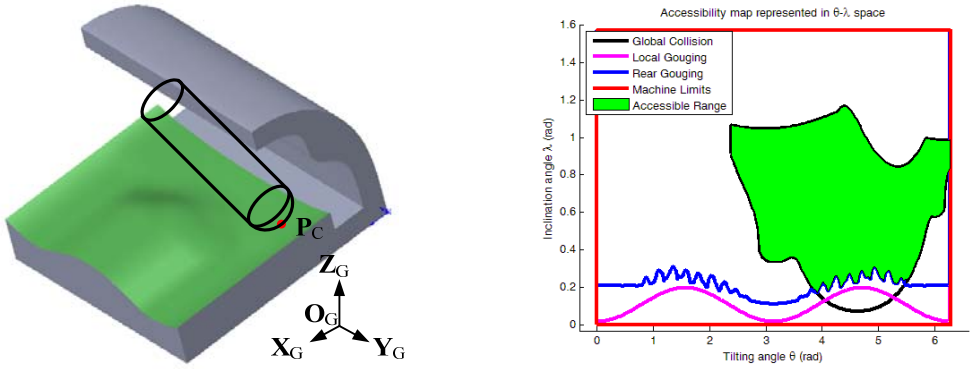
(4) Find the RG-free range from $(\lambda_{\theta\text{-lg}}, \lambda_{max})$. The current accessible range is $[\theta_i, (\lambda_{\theta\text{-rg}}, \lambda_{max})]$ $(\lambda_{\theta\text{-rg}} \ge \lambda_{\theta\text{-lg}})$. If the accessible range is NULL, $i = i + 1$, go to (2).

(5) Find the GC-free ranges from $(\lambda_{\theta\text{-rg}}, \lambda_{max})$ at $\{\mathbf{P}_k, k = 1, 2, \ldots, m\}$. The final accessible range is $[\theta_i, (\lambda_{\theta\text{-gc1}}, \lambda_{\theta\text{-gc2}})]$ $(\lambda_{\theta\text{-gc1}} \ge \lambda_{\theta\text{-rg}}, \lambda_{\theta\text{-gc2}} \le \lambda_{max}$. Return NULL if such a range does not exist. $i = i + 1$. Go to (2).

(6) If the A-map is not NULL, output the A-map. Else, $\mathbf{P}_C$ is not accessible.

End

This A-map construction algorithm is further illustrated by the following example, in which the A-map for $\mathbf{P}_C$ (u=0.2, v=0.8) on the workpiece shown in Fig. 2.8a is obtained. The cutter is of R = 5 mm, r_f=0.5 mm, L = 60 mm. The A-map is plotted in Fig. 2.8b. The range for θ is taken as [0, 2π] and θ is evenly sampled into 72 discrete values. Accessible ranges regarding the 3 kinds of interferences and the posture range determined by joint limits are also indicated.



(a) Workpiece and target point $\mathbf{P}_C$ (b) A-map as the intersection of accessible ranges

Figure 2.8 An example of A-map construction

In summary, the existing A-map algorithm takes a numerical approach to construct a cutter's accessible posture range to a surface point in terms of the cutter's inclination and rotational angles. The requirements for interference (LG, RG, GC) avoidance have been effectively addressed. For more details on A-map construction, readers can refer to the work of Li and Zhang (2006).

2.4. A New A-map Representation Scheme

With the A-map algorithm, the accessibility of a cutter at a surface point is obtained by specifying the allowable range for the inclination angle λ at all the discretely-sampled values of the rotational angle θ. Both λ and θ are calculated in the local frame $\mathbf{O}_L$-$\mathbf{X}_L\mathbf{Y}_L\mathbf{Z}_L$ at the surface point. Such a setting is convenient for the geometric analysis in accessibility evaluation. When a torus cutter is placed at a surface point $\mathbf{P}_C$

without gouging, the torus surface and the machining surface are tangent to each other at $\mathbf{P}_C$. As long as this tangency is maintained, any change in λ with a fixed θ (θ_i) can be viewed as a rotation of the cutter around a pivot point given as $\mathbf{P}_{pivot} = \mathbf{P}c + \boldsymbol{n}_C \cdot r_f$, where $\boldsymbol{n}_C$ is the surface normal at $\mathbf{P}c$ (see Fig. 2.6a). This guarantees that the accessible range of λ at θ_i is continuous and the accessible posture range can be represented using the upper and lower bounds of λ, given as (θ_i, $\lambda_{i,min}$~$\lambda_{i,max}$).

The representation of A-map using (θ_i, $\lambda_{i,min}$~$\lambda_{i,max}$) is convenient for accessibility evaluation at individual surface points, but not so when the A-map is used in the subsequent process planning steps for the following reasons:

1) A-map is discrete and contains little topological information. This makes it troublesome to test the feasibility of a given posture, $\boldsymbol{p}$ in the global frame. Suppose $\boldsymbol{p}$ corresponds to (θ_p, λ_p) in the local frame at the point in question and θ_p falls between 2 sampled values of θ. The accessible range of λ can only be obtained via interpolation, which may not be reliable, especially when the accessible range changes greatly between the 2 neighboring values of θs.
2) In the proposed process planning system, a comprehensive accessibility evaluation for all the cutters needs to be conducted in which the A-maps at all the sampled points on the machining surface will be constructed. The construction of the A-map at an arbitrary point can then be approximated via interpolation. However, such a task is virtually impossible with the existing representation of A-map due to lack of topological information and the different frames ($\mathbf{O}_L$-$\mathbf{X}_L\mathbf{Y}_L\mathbf{Z}_L$) used at different sampled points.

Therefore, in this study, a new representation of A-map is proposed to overcome the aforementioned problems. In the following sections, the new scheme as well as the construction method will be introduced.

2.4.1. Boundary posture chain

In the new representation scheme, the A-map is specified by the boundary postures (BPs) (in sequence) of the accessible range, named *BP chain*. An example is shown in Fig. 2.9 in which the red circles specify the BPs. The chain made up of BPs provides an approximation of the boundary of the accessible posture range. With the new form, the accessibility of a posture can be determined by checking whether this posture is enclosed by the BP chain.

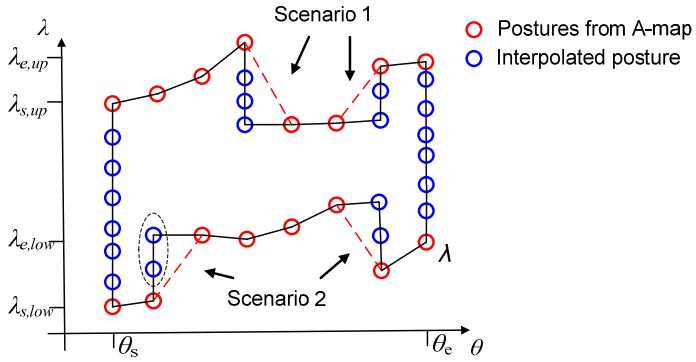


Figure 2.9 Adding extra BPs to form the complete BP chain

With the previously A-map construction, any number of consecutive sampled θs, whose accessible range of λ is not null, indicate the existence of an accessible posture range. The postures making up the A-map are the available BPs without any topological relationship. For an accessible posture range spanning from θ_s to θ_e, we can cetegarize the postures into 2 sub-groups, i.e., the lower boundary and the upper boundary. The lower boundary is given as $\left\{ \boldsymbol{p}_i \left(\theta_i, \lambda_{i,\min} \right) \middle| i = s, s+1, ..., e \right\}$, while the upper boundary is given by $\left\{ \boldsymbol{p}_i \left(\theta_i, \lambda_{i,\max} \right) \middle| i = s, s+1, ..., e \right\}$. However, these postures alone may not be able to represent the boundary accurately.

To achieve an acceptable level of accuracy, it is necessary to ensure the density of cutter postures stays above a certain level. In other words, the angular difference between the neighboring BPs should be kept below a certain threshold.

Here, the interval ($\Delta\theta$) between two neighboring sampled θs is used as the threshold for the angular difference between neighboring BPs. For any violation of this threshold, additional BPs need to be added through interpolation.

Given 2 consecutive BPs (θ_1, λ_1) and ($\theta_2=\theta_1+\Delta\theta$, λ_2), the 2 unit vectors are: $\boldsymbol{p}_1 = \left(\cos\theta_1 \sin\lambda_1, \sin\theta_1 \sin\lambda_1, \cos\lambda_1\right)$ and $\boldsymbol{p}_2 = \left(\cos\theta_2 \sin\lambda_2, \sin\theta_2 \sin\lambda_2, \cos\lambda_2\right)$. The angular difference (σ) between $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ is given as:

$$\sigma = \cos^{-1}\left(\sin\lambda_1 \sin\lambda_2 \cos\Delta\theta + \cos\lambda_1 \cos\lambda_2\right) \tag{2.10}$$

To satisfy the condition $\sigma \le \Delta\theta$, we have,

$$\cos(\Delta\theta) + \frac{\cos\lambda_1 \cos\lambda_1}{\sin\lambda_1 \sin\lambda_2 - 1} \le 0 \tag{2.11}$$

It is found that a sufficient condition for this inequality to hold is $\frac{\cos\lambda_1 \cos\lambda_1}{\sin\lambda_1 \sin\lambda_2 - 1} = -1$, which gives $\lambda_1 = \lambda_2$. Therefore, it can be concluded that as long as the value of λ is the same at neighboring postures, the angular difference is guaranteed to stay below $\Delta\theta$. This finding can guide the addition BPs where necessary in an efficient manner.

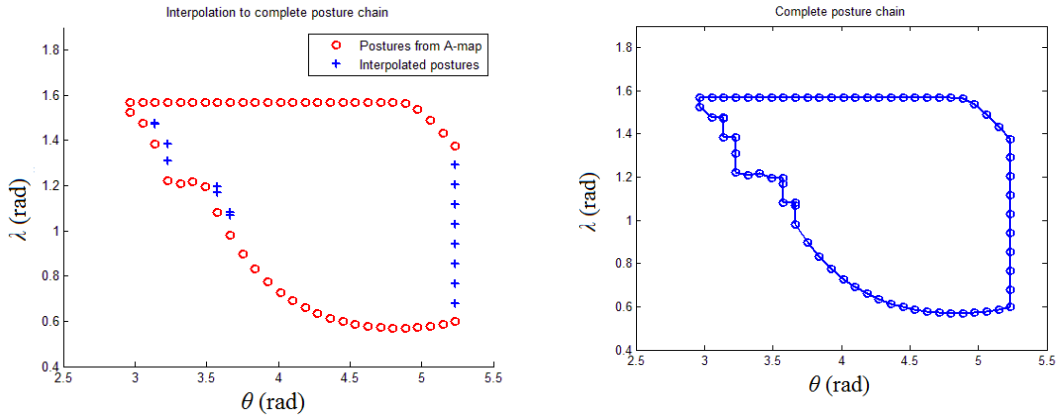
The construction of the BP chain starts with the checking of every pair of BPs on the upper and lower boundaries, respectively. All the cases of violation, i.e., when the angular difference between 2 BPs exceeds $\Delta\theta$, can be categorized into 2 scenarios. The interpolation strategies for adding extra BPs are as follows (see Fig. 2.9):

1) The BP pair is taken from the upper boundary, denoted as $\{\boldsymbol{p}_1(\theta_1, \lambda_1), \boldsymbol{p}_2(\theta_2, \lambda_2)\}$ and satisfies $\lambda_1 > \lambda_2$ (Be noted that it is not necessary that $\theta_1 < \theta_2$) Addition of new BPs is conducted at θ_1 by deducting $\Delta\theta$ from λ_1 iteratively to form a new BP, until λ_2 is reached. As a result, the pair $\{\boldsymbol{p}_1(\theta_1, \lambda_1), \boldsymbol{p}_2(\theta_2, \lambda_2)\}$ is replaced by $\{\boldsymbol{p}_1(\theta_1, \lambda_1), \boldsymbol{p}(\theta_1, \lambda_1\text{-}\Delta\theta,), \boldsymbol{p}(\theta_1, \lambda_1\text{-}2\Delta\theta,), \ldots, \boldsymbol{p}(\theta_1, \lambda_2)\ \boldsymbol{p}_2(\theta_2, \lambda_2)\}$. It can be

seen that the newly defined accessible range is well within the original range specified by linking $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ directly (see Fig. 2.9).

2) The BP pair are taken from the lower boundary, denoted by $\{\boldsymbol{p}_1(\theta_1, \lambda_1), \boldsymbol{p}_2(\theta_2, \lambda_2)\}$ and satisfies $\lambda_1 < \lambda_2$ (Still, be noted that it is not necessary that $\theta_1 < \theta_2$). Addition of new BPs is conducted at θ_1 by adding $\Delta\theta$ from λ_1 to form a new BP, until λ_2 is reached. As a result, the pair $\{\boldsymbol{p}_1(\theta_1, \lambda_1), \boldsymbol{p}_2(\theta_2, \lambda_2)\}$ is replaced by $\{\boldsymbol{p}_1(\theta_1,\lambda_1), \boldsymbol{p}(\theta_1, \lambda_1+\Delta\theta,), \boldsymbol{p}(\theta_1, \lambda_1+2\Delta\theta,), \ldots, \boldsymbol{p}(\theta_1, \lambda_2), \boldsymbol{p}_2(\theta_2, \lambda_2)\}$. Similarly, it can be seen that the new defined accessible range is well within the original range specified by linking $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ directly (see Fig. 2.9).

Furthermore, at both ends, the BP pairs: $\{(\theta_s, \lambda_{s,low}), (\theta_s, \lambda_{s,up})\}$and $\{(\theta_e, \lambda_{e,low}), (\theta_e, \lambda_{e,up})\}$ are compared. If violation occurs, addition of new BPs is conducted at θ_s or θ_e at intervals of $\Delta\theta$, starting from $\lambda_{s,low}$ or $\lambda_{e,low}$ until $\lambda_{s,up}$ or $\lambda_{e,up}$ is reached. Finally, all the BPs belonging to the same accessible region are connected clockwise to form the BP chain. In this way, A-map at a single surface point can be represented by one or several BP chains. An example of this process is shown in Fig. 2.10. Fig. 2.10a shows the BP interpolation result and Fig. 2.10b the final completed BP chain.



(a) Interpolation (b) The complete BP chain

Figure 2.10 An example of BP chain construction

2.4.2. Accessibility check for a posture

Given a BP chain by $\boldsymbol{C}_k = \left\{ \boldsymbol{p}_1^k(\theta_1, \lambda_1), \boldsymbol{p}_2^k(\theta_2, \lambda_2), ..., \boldsymbol{p}_n^k(\theta_n, \lambda_n) \right\}$ and a posture $\boldsymbol{p}_j$ (θ_j, λ_j), the accessibility of $\boldsymbol{p}_j$ can be determined by identifying the location relationship between them in the λ-θ space. Effectively, this is equivalent to the point-in-polygon (PIP) problem, i.e., determining whether $\boldsymbol{p}_j$ lies inside the region defined by the polygon $\boldsymbol{C}_k$. The ray-casting (crossings test) algorithm (Arvo, 1991) is adopted here to solve this problem. The idea is to draw an arbitrary ray starting from $\boldsymbol{p}_j$ that may intersect multiple times with the edges of $\boldsymbol{C}_k$. If the number of intersections is an odd number, it means the point is inside the polygon. Since arbitrary rays can be used, for simplicity, we select the ray given by $\lambda = \lambda_j$ and $\theta \geq \theta_j$. The task then becomes to identify, from $\boldsymbol{C}_k$, all the instances of $\boldsymbol{p}_j$ that satisfies $\left\{ \theta_{i+1}^k > \theta_j, \theta_i^k > \theta_j, \left(\lambda_i^k - \lambda_j\right)\left(\lambda_{i+1}^k - \lambda_j\right) \leq 0 \right\}$.

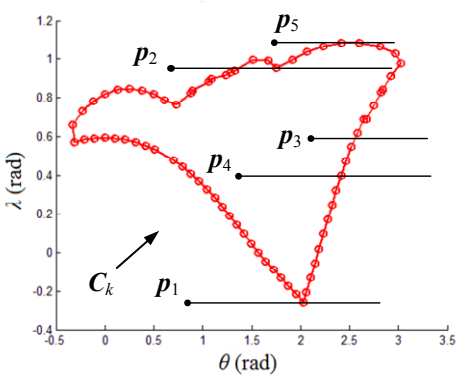


Figure 2.11 Accessibility checking formulated as a PIP problem

2.4.3. Intersection of A-maps

In some situations, it is necessary to find the common posture range (i.e., intersection), between A-maps. Since θ and λ of the BPs are obtained in the local frame, we need to convert all the postures in (θ, λ) pair to (α, β) pairs in the global frame $\mathbf{O}_G$-$\mathbf{X}_G\mathbf{Y}_G\mathbf{Z}_G$.

As shown in Fig. 2.12, for a cutter posture $\boldsymbol{p}$, β is the angle between $\boldsymbol{p}$ and plane $\mathbf{X}_G\mathbf{Y}_G$ and α the angle between the projection of $\boldsymbol{p}$ on plane $\mathbf{X}_G\mathbf{Y}_G$ and $\mathbf{X}_G$.

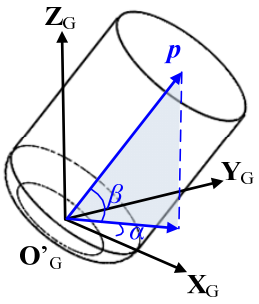
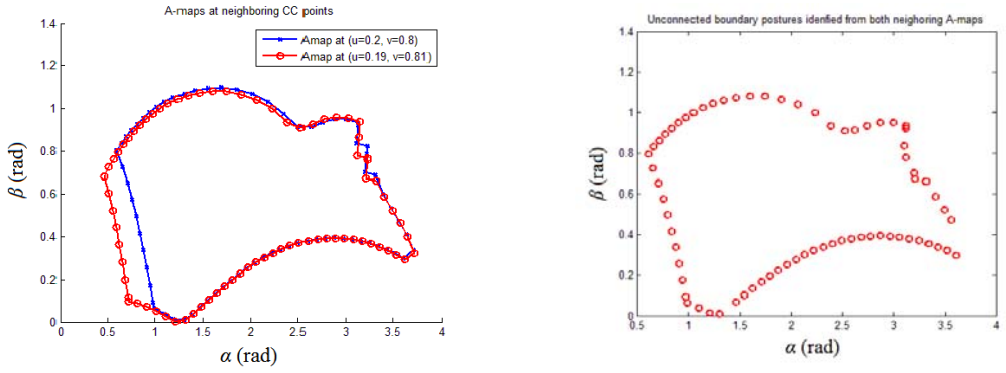


Figure 2.12 A posture $\boldsymbol{p}(\alpha, \beta)$ in the global frame

Given a BP chain $\boldsymbol{C}_i$ and a posture $\boldsymbol{p}$, $\boldsymbol{p} \in \boldsymbol{C}_i$ if $\boldsymbol{p}$ is on $\boldsymbol{C}_i$; $\boldsymbol{p} \Subset \boldsymbol{C}_i$ if $\boldsymbol{p}$ is inside $\boldsymbol{C}_i$. For a group of BP chains, $\boldsymbol{C}_1$, $\boldsymbol{C}_2$, …, $\boldsymbol{C}_k$, the BPs that form the intersection among $\boldsymbol{C}_1$, $\boldsymbol{C}_2$, …, and $\boldsymbol{C}_k$ is given as: $\boldsymbol{Q} = \left\{ \boldsymbol{q} \middle| \boldsymbol{q} \in \boldsymbol{C}_m, \boldsymbol{q} \Subset \boldsymbol{C}_i, i = 1,2,...,k, i \neq m \right\} \middle| m = 1,2,...,k \}$. In other words, a posture $\boldsymbol{q}$ of $\boldsymbol{Q}$ (the set of BPs making up the intersection of $\boldsymbol{C}_1$, $\boldsymbol{C}_2$, …, $\boldsymbol{C}_k$) should be inside each BP chain ($\boldsymbol{C}_i$) except the one to which it belongs to ($\boldsymbol{C}_m$). An example on how to obtain the BPs of the intersection among A-maps is shown in Fig. 2.13. Two A-maps in the form of BP chain are given in Fig. 2.13a and the BPs of their intersection is shown in Fig. 2.13b.



(a) Two BPs (b) BPs making up the intersection

Figure 2.13 The intersection of two A-maps in the form of BPs

At this stage, the BPs of the intersection among the given A-maps do not carry any topological information. Therefore, they have to be connected to form a BP chain $\boldsymbol{C}_q$. The problem can be described as: given a set of points on a plane $\boldsymbol{Q} = \{\boldsymbol{q}_1, \boldsymbol{q}_2 \ldots \boldsymbol{q}_m\}$, construct a non-intersecting boundary that passes through every point. Here, the boundary construction algorithm proposed by Veltkamp (1994) is adopted in which *Delaunay triangulation* (de Berg *et al.*, 2008) is conducted to construct a set of triangles, denoted as DT($\boldsymbol{Q}$). A convex hull of $\boldsymbol{Q}$ is subsequently constructed, which is taken as the initial solution for $\boldsymbol{C}_q$. The example shown in Fig. 2.14 illustrates this process.

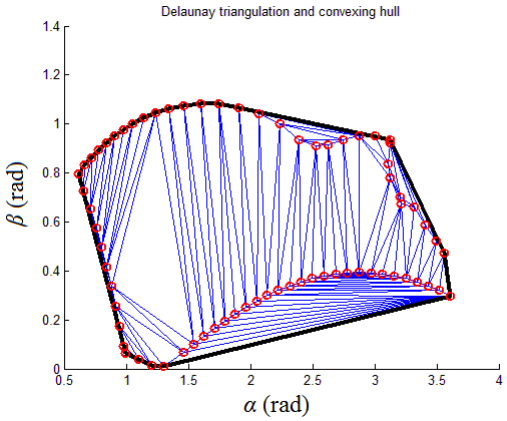


Figure 2.14 Delaunay triangulation and convex hull for the BPs

Every edge taken from $\boldsymbol{C}_q$, denoted as $\overline{\boldsymbol{q}_i\boldsymbol{q}_{i+1}}$, is contained by one or more triangles from DT($\boldsymbol{Q}$), denoted as $\Delta\boldsymbol{q}_i\boldsymbol{q}_k\boldsymbol{q}_{i+1}$, where $\boldsymbol{q}_k$ is the third vertex. If $\boldsymbol{q}_k$ is not on $\boldsymbol{C}_q$ already, $\overline{\boldsymbol{q}_i\boldsymbol{q}_{i+1}}$ could be replaced by two edges $\overline{\boldsymbol{q}_i\boldsymbol{q}_k}$ and $\overline{\boldsymbol{q}_k\boldsymbol{q}_{i+1}}$. In this way, $\boldsymbol{q}_k$ is added into $\boldsymbol{C}_q$. In the work of Veltkamp (1994), he provided a heuristic for selecting the next $\boldsymbol{q}_k$ to add based on the current solution of $\boldsymbol{C}_q$, after which $\boldsymbol{C}_q$ will be updated. With this selection-update cycle, an iterative process can be established to construct the final BP chain that includes all the BPs from $\boldsymbol{Q}$. In Fig. 2.15, the final BP chain constructed from the initial solution given in Fig. 2.14 is shown.

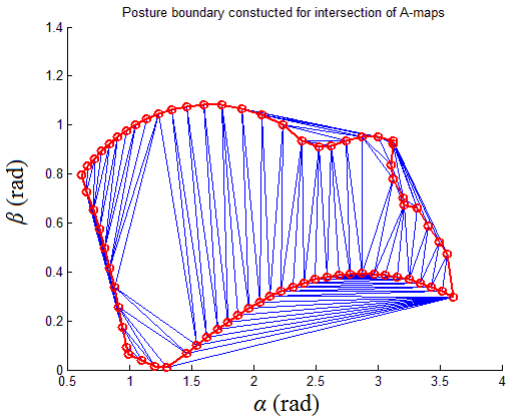


Figure 2.15 The final BP chain

2.4.4. A-map construction through interpolation

In the proposed process planning system, a comprehensive accessibility evaluation for various available cutters on the machining surface takes place before all other tasks. This evaluation starts with the generation of a high-density point set sampled from the machining surface followed by the construction of A-maps at all the sampled points (SPs). Although the A-map construction is time consuming, this evaluation is critical as the cutter selection entirely depends on the result. The importance for rapid approximation of A-maps arises from the tool-path planning stage where the A-maps at the cutter contact (CC) points are required. Since the density of the SPs is generally very high (but they are different from the CC points), it is believed that the A-map at a CC point can be approximated through interpolation, i.e., the intersection of A-maps at the neighboring SPs.

The generation of the SPs is carried out in the parametric frame of $\boldsymbol{S}_m$ at even intervals of Δu and Δv ($0 \le u \le 1$, $0 \le v \le 1$). Meanwhile, to make the interpolation reliable and to keep the computational load acceptable, the density of the SPs should be determined based on actual size of $\boldsymbol{S}_m$ in the Cartesian space. As a rule of thumb, the average interval (d) between neighboring SPs should satisfy $d < 0.1R_{min}$, where R_{min} is the major radius of the smallest cutter from the cutter library. Suppose $\boldsymbol{S}_m$ is bounded

by a rectangle of size l_1 by l_2, the following heuristic is given regarding the sampling interval Δu (Δv):

$$\Delta u(\Delta v) \le \left(\frac{0.1R_{\min}}{\max\{l_1, l_2\}} \right) \tag{2.12}$$

To construct the A-map at a target point **P** by interpolation, we need to identify a set of SPs that makes up the neighborhood of interpolation for **P**, denoted by **N**. SPs from **N** should meet requirements of both proximity and similarity, i.e., the points from **N** should be close enough to **P** in the Cartesian frame while having similar geometric properties with **P** (the similarity in geometric properties is measured by the angular difference between surface normal at the two points in question). Based on these requirements, the following steps are taken to construct **N**:

1) From all the SPs, identify those within a distance of R_{min} from **P** and put them into **N**. By using R_{min} as the low bound, we make sure that the proximity condition is satisfied and there are enough candidates in **N** to work with.
2) Calculate the angular difference between the surface normal at each SP from **N** (denoted as $\boldsymbol{n}_k$) and **P**, given as $\varphi_k = \cos^{-1}(\boldsymbol{n}_k \cdot \boldsymbol{n})$. Rank the points in **N** with φ_k (k=1, 2,…, n) in ascending order (n is the current number of points in **N**).
3) Take the top 4 SPs in **N** to form the final interpolation pool **N.**

   If $\varphi_4 > \varepsilon$ (ε is the lower bound of the angular difference (e.g., 3°)

   Abort interpolation and build the A- map using the analytical method

   Else

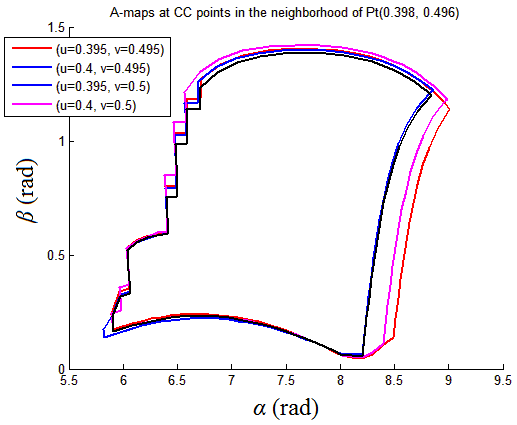
   Output **N** and exit.

   End If

Only 4 points are kept in **N** to make sure the approximation does not become over-conservative, causing too much accessible range to be misclassified as inaccessible.

With the points in **N** and their A-maps in the forms of BP chain, the intersection of these A-maps are obtained using the procedures introduced in Section 2.4.3. The finally obtained intersection in BP chain is the A-map at **P**.

A complete example for obtaining the A-map through interpolation is shown here. The machining surface is shown in Fig. 2.1b with an area of 100mm × 100mm, and the cutter is of (R = 8mm, r_f = 2mm, L = 80mm). A total of 201 × 201 points are sampled uniformly in both u and v directions with $\Delta u = \Delta v = 0.005$. The target point **P** is at (0.398, 0.496). The neighborhood for interpolation is given as **N** = {$\mathbf{P}_1$(0.395, 0.495), $\mathbf{P}_2$(0.4, 0.495), $\mathbf{P}_2$(0.395, 0.5), $\mathbf{P}_4$(0.4, 0.5)} and their A-maps are shown in Fig. 2.16a in the form of BP chains. The approximated A-map at **P** is obtained through interpolation as shown in Fig. 2.6b.



(a) A-maps at the sampled points

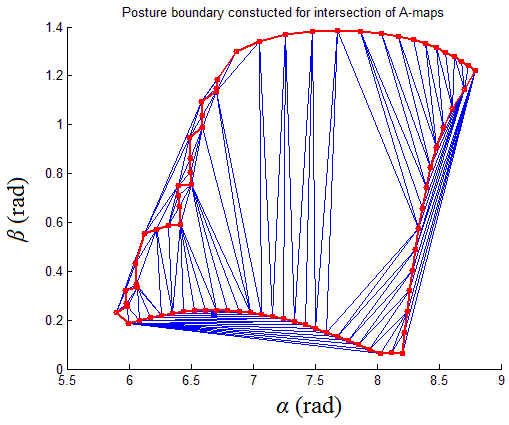


Figure 2.15 A-map at **r**(0.398, 0.496)

Figure 2.16 Obtaining an A-map through interpolation

For comparison, the A-map at **P** is also constructed directly using the analytical method introduced in Section 2.3 and plotted together with the approximated A-map in Fig. 2.17. It can be seen that the approximated A-map is very close to the "exact" one. On the other hand, it is also relatively conservative, thus providing a safe approximation.

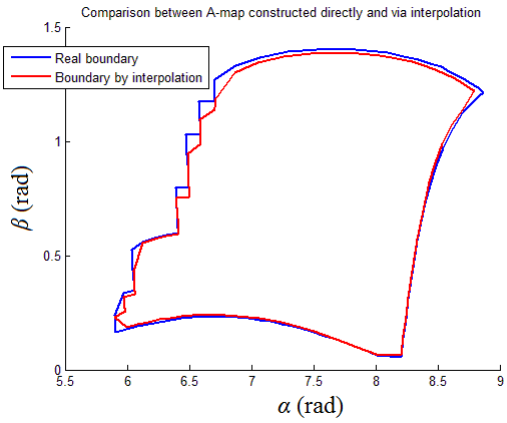


Figure 2.17 Comparison between A-map constructed directly and via intersection

2.4.5. Computational complexity analysis on A-map construction

The computation complexity of both A-map construction methods (*exact* and *interpolation*) is compared here. Given N SPs on the machining surface and M sampled θs, the computational complexity of the *exact method* is as follows:

1) O(1) for finding the LG-free posture range.
2) $\mathrm{O}(N)$ for identifying the RG-prone points and $\mathrm{O}(MN)$ for finding the RG-free posture range at the RG-prone points (worst case scenario). The overall computational complexity for obtaining the RG-free posture range is $\mathrm{O}(MN)$.
3) $\mathrm{O}(N)$ for identifying the GC-prone points and $\mathrm{O}(MN)$ for finding the GC-free posture range at the GC-prone points (worst case scenario). The overall computational complexity for finding the GC-free posture range is $\mathrm{O}(MN)$.

Combining the computational complexity of all 3 tasks, the overall computation complexity for building the A-map at a surface point is $\mathrm{O}(MN)$.

Meanwhile, for the *interpolation method*, the A-map at the new point can be viewed as the intersection of k BP chains. At the same time, the number of BPs in each BP chain is proportional to M. Hence, the number of BPs in the ith BP chain can

be represented as K_iM ($0<K_i<2$). The computational complexity for obtaining the intersection of these A-maps is $\mathrm{O}\left(\sum_{i=1}^{k}\sum_{j=1,j\neq i}^{k} K_iK_jM^2\right)=\mathrm{O}\left(M^2\right)$. On the other hand, according to Amenta *et al.* (2007), the Delaunay triangulation algorithm has the complexity of $\mathrm{O}\left(M\right)$ under the worst case scenario. Furthermore, the complexity of the boundary extraction algorithm is $\mathrm{O}\left(M^2\right)$ under the worst case scenario. Therefore, the overall computation complexity for the interpolation construction method is $\mathrm{O}\left(M^2\right)$.

Normally, the number of SPs (N) is much larger than the number of discrete θs (M). In the example shown in the last section, for instance, $N = 4\times10^4$ while $M = 72$ (with an interval of 5°). Clearly, the *interpolation method* is much faster than the *exact method*, thus increasing the computational efficiency significantly.

2.5. Discussion

In process planning for 5-axis sculptured surface milling, the accessibility information of the cutters is of uttermost importance for cutter selection and tool-path generation. In this chapter, the previously developed accessibility evaluation method, named the *exact A-map method*, is first introduced. The exact approach is heavy in computation. To make an improvement in this aspect, a new representation of A-map in the form of boundary posture (BP) chain is proposed. Based on this new representation scheme, an *interpolation*-based A-map construction method is proposed, which include algorithms for neighborhood identification, A-map intersection, and BP sorting. Compared with the A-map from the *exact method*, the A-map from the *interpolation method* is very close to the exact one but with some conservativeness. On the other hand, the *interpolation method* has a much lower computational complexity. For the

process planning system proposed in this thesis, the A-maps at a group of densely sampled points from the machining surface will be generated using the *exact method*. Subsequently, when the A-map at a new point is required, the interpolation method will be used based on the A-maps at the sampled points. It is believed that by doing so, the computational efficiency of the process planning will be improved tremendously while the reliability on the accessibility of cutter postures will not suffer.

CHAPTER 3

OPTIMAL CUTTER SELECTION FOR 5-AXIS MILLING

Cutter selection is a crucial task for process planning of 5-axis milling of sculptured surfaces. The goal is to select cutter(s) that will be able to produce the surface with the specified shape error tolerance with the best machining efficiency. Generally, larger cutters are preferred as they have higher material removal rate thus better machining efficiency. However, such a rule is only applicable when the machining region is accessible to the cutter. Based on the *exact* A-map construction algorithm, a cutter's accessibility information to a machining surface can be obtained at all the sampled points. Subsequently, cutter selection can be performed in two different modes: *single*-cutter and *multi*-cutter.

Single-cutter selection refers to the selection of a single cutter to finish the given machining surface. The objective is to identify the best cutter, from all the accessible ones to the machining surface, that has the best machining efficiency. This task is relatively straight-forward. Given two cutters that are both accessible to the machining surface, the cutter with a larger R is preferable. This is because the cutter with larger major radius will produce a bigger cutting profile, leading to better material removal rate. On the other hand, with the same R, the cutter with smaller r_f are preferable. This is based on the observation that a flat-end cutter is more efficient than a ball-end cutter of the same R.

Multi-cutter selection refers to the selection of a multi-cutter set (with a number of cutters) to finish a given surface. The optimal set can collectively finish the

whole surface with the best machining efficiency. This means that the whole surface is distributed (partitioned into different machining-areas) to the cutters in the set. The overall performance of a multi-cutter set is the combination of the performance of each cutter on its assigned machining area. This requires the development of algorithms for machining area distribution (assignment) to different cutters and performance prediction for each {cutter/machining-area} combination without generating the actual tool-paths. These issues will be addressed in this chapter.

3.1. Related Work on Cutter Selection

Most of the commercial CAM packages rely on the user for cutter selection. When the cutter assigned by user is found to be incapable of interference avoidance during tool-path generation, tool-path generation will be aborted and the user will be prompted to select a smaller cutter. Such a process is obviously non-optimal and non-efficient. Over the years, cutter selection algorithms have been proposed by many researchers, but mainly for 3-axis milling. Accessibility checking for 3-axis milling with ball-end cutters is basically a curvature matching problem, as the cutting profile of ball-end cutters is invariant in all cutting directions. As long as the cutter curvature is larger than the maximum principal curvature on the surface, the cutter will be accessible to the whole surface. Such an approach has been used by Ding *et al.* (2001); Lim *et al.* (2001); Pottmann *et al.* (1999) and You *et al.* (2007). Based on this simple rule, a sculptured surface can be partitioned into accessible and inaccessible regions for any specific cutter. Also due to the simple cutting profiles of ball-end cutters, the tool-path length of a cutter within its machining-region can be accurately estimated. After the tool-path length for each cutter within a cutter combination is obtained, the performance of different multi-cutter sets can be easily compared based on total tool-path length (Yang and Han, 1999).

For 5-axis milling, the cutter can access the machining surface with many different postures. At the same time, torus- and flat-end cutters with more complex cutting profiles are used instead of ball-end cutters. These factors have made the job of cutting performance evaluation for the cutter more complicated. To make sure the selected cutter is accessible everywhere on the machining surface, Lee and Chang (1996) proposed to evaluate a cutter's accessibility at a large number of points sampled from the machining surface. At each sampled point, the posture with the maximum effective cutting radius is identified from the feasibility cone at the point. In terms of accessibility evaluation, only local gouging avoidance is taken into consideration here. Jensen *et al.* (2002) proposed a trial-and-error cutter selection algorithm for torus cutters based on curvature matched machining with consideration for all 3 kinds of interferences. The algorithm starts with the largest cutter in the cutter library. An interference detection and correction algorithm is applied at the all sampled points to find an interference-free orientation for the cutter in question. If at a certain point, such an orientation cannot be found, a smaller cutter (a cutter with larger minor radius or smaller major radius) will be taken to repeat this process until a feasible cutter is identified. The above mentioned methods are both intended for single cutter selection for 5-axis machining. On the other hand, little literature is available on multi-cutter selection for 5-axis milling, possibly due to difficulties in evaluation of cutter accessibility or allocation of machining surfaces.

In our previous work, heuristic-based methods for both single- and multi-cutter selection have been developed. Both methods are implemented based a comprehensive accessibility evaluation on a set of sampled points with high-density from the machining surface. For the single-cutter mode, the largest cutter with non-empty A-maps at all the sampled points is selected (Li and Zhang, 2006). For the

multi-cutter mode, candidate multi-cutter sets are firstly formed based on the accessibility information and the optimal set is extracted based on a heuristic (Li and Zhang, 2009). In the following sections, these two existing algorithms will be introduced first, followed by extensions and improvements made in this study.

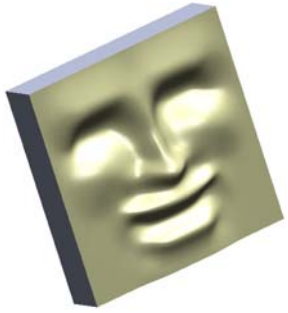
3.2. Evaluating a Cutter's Accessibility to a Machining Surface

For a cutter to be feasible for a given machining a surface, it has to be interference-free at every point on the surface. In practice, a set of sampled points (SPs) are generated to represent the surface approximately. If the cutter is accessible to all the SPs (or the cutter has a non-empty A-map at every sampled point), the cutter is deemed accessible to the surface.

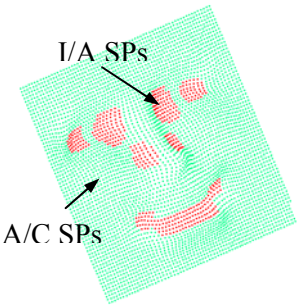
Based on accessibility to the given surface, all the available cutters in the library can be grouped into 3 categories: *accessible (A/C) cutters*, *partially accessible (P/A) cutters*, and *inaccessible (I/A) cutters*. The A/C cutters have non-empty A-maps at all the SPs; the P/A cutters have non-empty A-maps at some of the SPs and the I/A cutters have no non-empty A-maps at any of the SPs. In process planning, the A/C cutters can be used for single-cutter mode; the P/C cutters can be combined with A/C ones to form multi-cutter sets for multi-cutter mode; the I/A cutters, on the other hand, will not involve in the machining of this surface.

For a A/C or P/A cutter, based on the A-map information at the SPs, the SPs can be categorized into two types: A/C and I/A. Accordingly, the surface can be divided into A/C and I/A regions by grouping all the A/C SPs and extracting their boundaries. However, due to discrete nature of the surface representation and uneven curvature distribution, discontinuities may exist in the initial partitioning. An illustration example is given in Fig. 3.1a (the "FACE" surface) and b (categorization of SPs for cutter: R = 8mm, r_f= 0.5mm, L = 60mm). Such discontinuities will cause

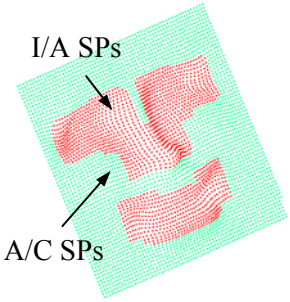
inconvenience for machining region allocation and tool-path generation. A refinement process is, therefore, necessary. In the refinement, the SPs within a distance of d_{safe} from any I/A SP are taken as I/A. This effectively adds a safety margin around each I/A SP to make the partition more reliable. For the example shown in Fig. 3.1, d_{safe} is set as $10d$, where d the sampling interval. The refined result is shown in Fig. 3.1c. Based on the refined result, the boundary extraction method by Park and Choi (2001) is adopted to extract the boundaries of the A/C regions for the cutter. The extracted A/C regions of the "FACE" for the cutter are shown in Fig. 3.1d.



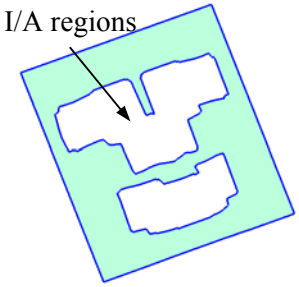
(a) The "FACE"



(b) Original A/C and I/A SPs



(c) A/C and I/A SPs after refinement



(d) Boundary extraction for accessible area

Figure 3.1 Machining surface partition based on cutter's accessibility

For single cutter selection, the cutter size is a direct indicator for the cutter's optimality. Thus, single cutter selection starts with accessibility evaluation for the largest cutter in the given cutter library, one at a time, and stops when the first accessible cutter is identified. In the meantime, the A-maps of the selected cutter at all the SPs are also obtained. Meanwhile, the algorithm for multi-cutter selection is much

more complicated. Details of the existing algorithm and its extensions will be introduced in the following section.

3.3. Multi-cutter Selection Based on Heuristics

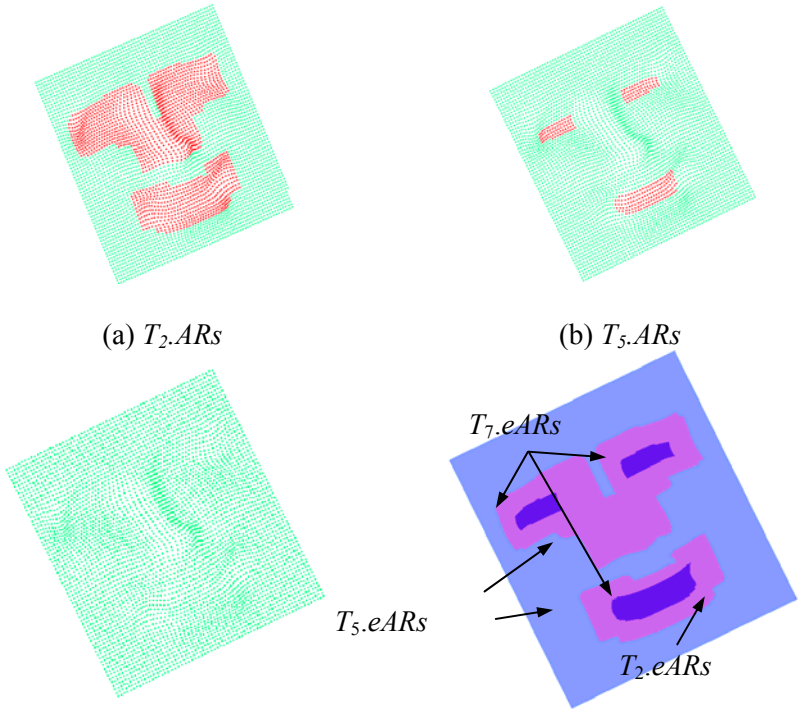
Although the selection of a single cutter to machine a given surface is straight-forward, such a choice may not be optimal, as the selected cutter is always confined by the most critical feature on the surface. For surfaces with a large portion of non-critical area, low machining efficiency is inevitable. This is why it is desirable to use multiple cutters to machine such surfaces. The larger cutters can be used to machine the flat and/or convex areas while smaller cutters for the critical concave or saddle areas. Considering that most 5-axis machines are equipped with the fast tool change mechanism, the time gain from better machining efficiency will be more significant.

As introduced in the last section, the two groups of cutters that can be used for multi-cutter machining are A/C and P/A cutters. To finish the whole surface without interference, a feasible multi-cutter set must contain at least one A/C cutter. At the same time, larger cutters are preferred for their better material removal rate. Thus, if there is more than one A/C cutter in a cutter library, only the largest one, denoted as $T_{a/c}$, should be taken to form a feasible set. Any feasible cutter multi-cutter set contains $T_{a/c}$ and one or more P/A cutters, given as T-Set = $\{T_1, T_2, \ldots, Tn, T_{a/c}\}$, where $T_1, T_2, \ldots, Tn$ are P/A cutters, from large to small.

3.3.1. Allocation of machining regions within a feasible multi-cutter set

The accessible region (*AR*) of a cutter is obtained after accessibility evaluation. The *ARs* of a cutter can be formed by one or several regions on the surface. Given a feasible multi-cutter set, the whole surface needs to be partitioned and allocated to all the cutters. The regions that are allocated to a cutter are called the cutter's *effective*

accessible regions (*eARs*), which is a subset of its original *ARs*. For better machining efficiency, surface partition and distribution follows the heuristic: it is preferable to allow the larger cutter to machine as much of its *ARs* as possible. As a result, the *eARs* for the largest cutter in the multi-cutter set is its original *ARs*, i.e., $T_1.eARs = T_1.ARs$. The remaining surface after updating is then ready to be partitioned by the updated multi-cutter set$\{T_2, \ldots, Tn, T_{a/c}\}$, and $T_2.eARs$ is obtained in the same fashion. In this way, the *eARs* of the remaining cutters in the set can be worked out one at a time. Obviously, the *eARs* of a smaller cutter are smaller than its original *ARs*. An example of this surface partition and distribution process is shown in Fig. 3.1. The *ARs* of the 3 cutters in the set of $\{T_2, T_5, T_7\}$ on the "FACE" are shown in Fig. 3.2a, b, and c, respectively. The *eARs* of these 3 cutters are shown in Fig 3.1d. Proportionally, $T_2.eARs$, $T_5.eARs$, and $T_7.eARs$ take up 52.7%, 37.2%, and 10.1%, of the whole surface respectively.



(a) $T_2.ARs$ (b) $T_5.ARs$

(c) $T_7.ARs$ (d) *eARs* of the 3 cutters

Figure 3.2 *ARs* and *eARs* of cutters in the multi-cutter set $\{T_2, T_5, T_7\}$

3.3.2. Construction of the candidate multi-cutter set

Although a larger cutter generally results in higher material removal rate, putting too many cutters in one set may cause the machining efficiency to drop. This is because much time is wasted in the air travel needed for tool change. Besides, too many cutters may cause inconsistencies at the boundary between *eARs* for different cutters, resulting in poor surface quality. Here, a heuristic is developed to select the *candidate* multi-cutter sets from the feasible ones by checking the machining area ratio (MR) of each *eAR*, which refers to the ratio of the area of the *eAR* to the whole surface area. By setting a minimum threshold value of MR (MR_{min}) for all the P/A cutters in a set, any *eAR* with its MR < $MR_{\min}$ will be removed from its corresponding cutter. If all the *eARs* of a cutter are removed, the cutter itself will be removed from the set. Such a measure can effectively limit the number of cutters in a multi-cutter set and the number of candidate sets. The algorithm for this process is as follows:

Algorithm: Construction of the candidate multi-cutter sets

Input: *(a) All the feasible multi-cutter sets; eARs of all the cutters in each cutter set.*

(b)The whole surface area A; MR_{min}

Output: *The candidate multi-cutter sets*

Begin

(1) Pick a feasible multi-cutter set as the current set (*C-set*) and set the first cutter in *C-set* as the current cutter (*C-cutter*).

(2) If *C-cutter* is not the last cutter in *C-set*, check the area of each *eAR*;

a) Remove the *eAR* if its area is smaller than ($MR_{min} \times A$).

b) If there are no *eARs* left for *C-cutter*, remove *C-cutter* from *C-set*. Re-calculate the *eARs* of the remaining cutters and update *C-set*. Go to (d).

c) If there are still remaining *eARs*, place *C-cutter* into the corresponding candidate set, together with its *eARs*.

d) If C-cutter is not the last cutter in C-set, set next cutter in *C-set* as *C-cutter*, repeat (2). Otherwise, close this candidate set.

(3) If *C-set* is not the last feasible set, go to (1).

(4) Output all the candidate multi-cutter sets. Stop.

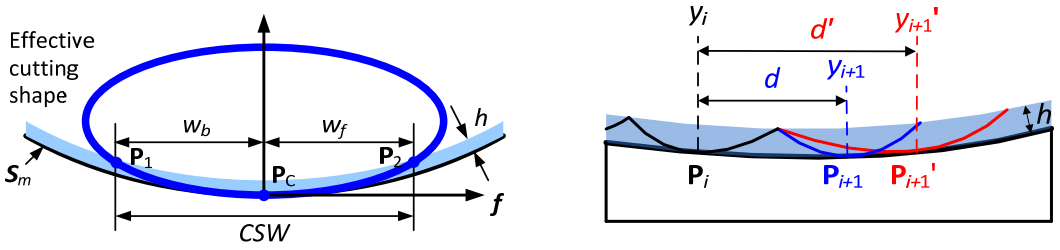
End

3.3.3. Heurist-based tool-path length estimation for multi-cutter set selection

The criterion for identifying the optimal cutter combination is machining efficiency, for which the machining time is directly measurement. However, in 5-axis machining, feed-rate is not constant, as the speeds of the two revolute joints need to be coordinated to reach the target simultaneously. As a compromise, the total tool-path length is taken as a reasonable indicator for machining efficiency. The cutter selection task can thus be summarized as 'selecting the multi-cutter set with the shortest total tool-path length from the candidate sets'. However, at the stage of cutter selection, tool-paths are not available and it is impractical to generate the tool-paths for all the candidate cutter sets. As a remedy, methods need to be developed to produce estimations of the total tool-path lengths without generating them. In our early work, a heuristic-based method was developed.

The estimation method is designed for parallel cutting pattern. Given a cutter and an accessible surface region (with a total area of A), suppose the average cutting-strip width at all the CC points is W_{ave}, A/W_{ave} should serve as a close approximation of the total tool-path length. A/W_{ave} is given the name *tool-path length index* (L_{index}). Obviously, the key for obtaining L_{index} is to estimate the average cutting-strip width at all the CC points on the surface.

The cutting-strip width (CSW) at a CC point $\mathbf{P}_C$ is calculated as the distance between the 2 intersection points between the effective cutting shape of the cutter and the offset surface (see $\mathbf{P}_1$ and $\mathbf{P}_2$ in Fig. 3.3a). The effective cutting shape of the cutter is co-determined by cutter posture and cutting direction at $\mathbf{P}_C$. The size of CSW determines the side-step between tool-paths and hence greatly affects the total tool-path length (see Fig. 3.3b). Thus, the cutter postures are usually adjusted based on the cutting direction to produce as large CSWs as possible. In this manner, larger path intervals will be obtained, resulting in shorter tool-path lengths.



(a) Effective cutting shape and CSW (b) Influence of CSW on path intervals

Figure 3.3 Influence of cutter posture on CSW and machining efficiency

However, at the time of cutter selection, none of cutting direction, CC point locations or cutter postures is available. Thus, some assumptions need to be made to conduct the approximation. First, W_{ave} is calculated as the average of CSW at all the SPs instead of CC points. As both CC points and SPs are densely distributed over the whole surface, such an assumption is reasonable Secondly, the cutting direction is assumed to be along the minimum principal direction at the SP, a direction that maximizes material removal rate. The third assumption is regarding the cutter posture. In our previous research, a heuristic was developed to select the posture with the near-maximum CSW from the A-map at a point, i.e., 1) align the cutter with the cutting direction as much as possible and 2) keep the inclination angle λ as small as possible.

For estimation of total tool-path length, such a posture is assumed at each SP based on the cutting direction specified in the second assumption.

Based on these 3 assumptions, the CSW at a SP can be calculated. Fig. 3.4 gives the illustration of a cutter placed at a SP in which $\boldsymbol{x}_\omega$ is the assumed cutting direction while $\boldsymbol{f}$ the feed direction. The angle between $\boldsymbol{x}_\omega$ and $\mathbf{X}_L$ is ω. Plane $\mathbf{Z}_L$-$\boldsymbol{f}$ intersects with both the cutter's torus surface and the machining surface, producing the cutter and surface curves, respectively. The curvature of cutter curve is given as:

$$\kappa_t = \frac{1}{\left(R - r_f\right)/\sin\lambda + r_f}\cos^2\omega + \frac{1}{r_f}\sin^2\omega \tag{3.1}$$

The curvature of the surface curve is given as:

$$\kappa_s = \kappa_{s\max}\cos^2\omega + \kappa_{s\min}\sin^2\omega \tag{3.2}$$

where $\kappa_{s\max}$ and $\kappa_{s\min}$ are the maximum and minimum principal curvatures at the point.

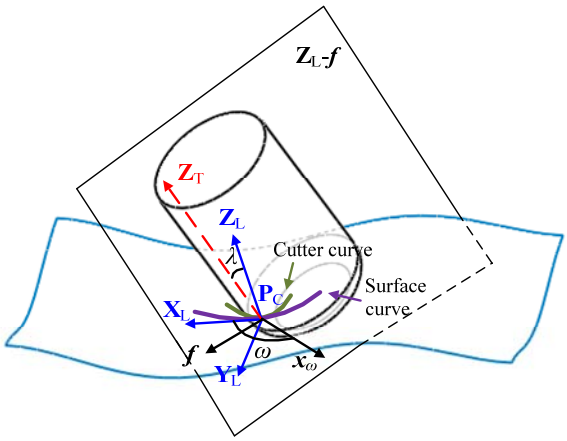


Figure 3.4 Estimation of CSW at a sampled surface point

Based on the second assumption on cutting direction, $\boldsymbol{x}_\omega$ is coincident with $\mathbf{X}_L$, i.e., ω=0. The curvatures of the cutter and the surface curves can thus be given as $\kappa_t = \dfrac{1}{(R - r_f)/\sin\lambda + r_f}$ and $\kappa_s = \kappa_{s\min}$. According to Lee and Chang (1996), the CSW at $\mathbf{P}_C$ can be approximated as:

$$CSW = \sqrt{\frac{8h}{\kappa_t - \kappa_s}} \tag{3.3}$$

where h is the tolerance for scallop-height. The gain from using such an approximation is low computational cost. However, accurate methods for obtaining CSW, such as the iterative method introduced in chapter 4 or the enveloping-profile-based method provided by Sarma and Dutta (1997) can also be used.

Combining Eqs. (3.1)~(3.3), the CSW at $\mathbf{P}_C$ can be obtained as:

$$CSW = \sqrt{\frac{8h}{\frac{1}{\left(R - r_f\right)/\sin\lambda + r_f} - \kappa_{s\min}}} \tag{3.4}$$

Suppose a total of m SPs are sampled from the surface, the tool-path length index is:

$$L_{\text{index}} = \frac{A}{\frac{1}{m}\sum_{i=1}^{m} CSW_i} \tag{3.5}$$

Based on Eq. (3.5), the L_{index} of every cutter/surface in a multi-cutter set can be calculated and therefore the $\sum(L_{\text{index}})$ of the multi-cutter set. The multi-cutter set with the smallest $\sum(L_{\text{index}})$ is the optimal one.

The heuristic-based method was extensively tested with different machining surfaces. For each surface, the actual tool-paths are generated for all the candidate multi-cutter sets. Then the ranking of multi-cutter sets based on actual total tool-path length is compared with the ranking based on L_{index}. It was observed that the ranking based on L_{index} sometimes deviates a lot from the actual ranking. This may be due to over-simplification in the formulation of L_{index} or due the three assumptions made for calculating the CSW. Hence, developing a new estimation method with better accuracy becomes a task in this study.

3.4. Tool-path Length Prediction Using Neural Networks

Considering that the relationship between total tool-path length and cutter combinations may be too vague or too complicated to be described using an explicit form, the tool of neural networks (NNs) is proposed to be used for tool-path length estimation regarding a cutter and an accessible surface region.

Neural networks (NN) are a group of artificial intelligent systems. They are generally composed of several layers of inter-connected neurons that mimic the performance of biological neurons. Many variations of neural networks exist. For the neural network used in this study, it will be exposed repeatedly to a training data set that composes of inputs and the known correct outputs. Such a process is called 'supervised learning'. During this process, the connection weights between neurons on different layers will be iteratively adjusted by a feed-back scheme. After the training converges, the NN should be mimicking the properties of the system or function that is being simulated. All the information of the target system would be stored in the connection weights. When fed with new inputs that have not been encountered before, the trained NN could still produce the correct result or a close approximation. For more information on this tool, readers can refer to the work of Haikin (1998). The task of NN in multi-cutter selection is to predict the tool-path length for a cutter/surface combination.

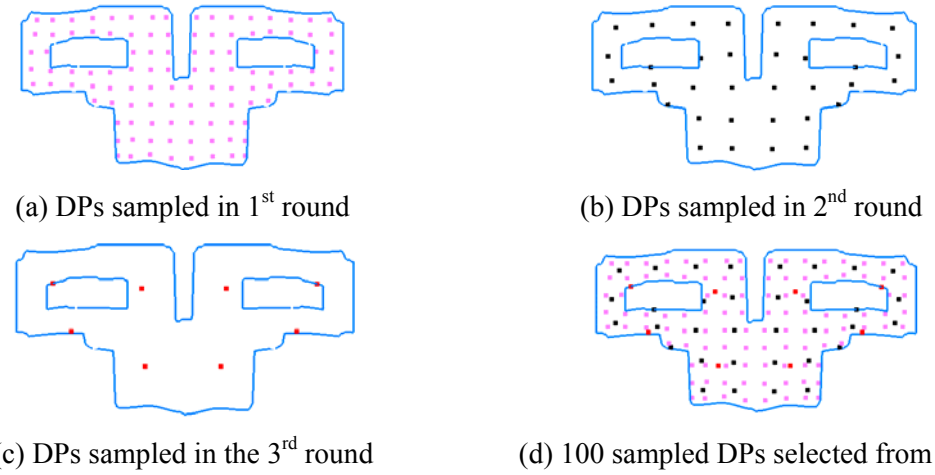
Before details of the NN algorithm are introduced, it is worth noting that one important difference between the NN-based method and the heuristic-based method is that the cutting direction $\boldsymbol{x}_\omega$ is provided for tool-path length estimation. As will be introduced in Chapter 4, with the accessibility information at all the SPs, an optimal cutting direction can be identified before multi-cutter selection.

3.4.1. Input identification for neural network

Tool-path length is determined by the surface area and the CSWs over the whole surface. To collect sufficient information for tool-path length estimation, the inputs should be extracted at the SPs over the whole surface regarding these two parameters. For this purpose, a machining character parameter (MCP) is calculated for each selected SP, which is given as *A/CSW*, where *A* is the surface area and *CSW* is the near-maximum CSW at the SP. The posture is selected using the heuristic introduced in section 3.3.3 based on the provided cutting direction $\boldsymbol{x}_{\omega}$.

Generally, a NN takes in a fixed number of inputs while the total number of SPs varies for surfaces of different sizes. In this study, a fixed number of SPs are selected as data points (DPs) for a given surface. To select a fixed number of DPs evenly from the SPs, a special selection scheme is developed. Suppose a total number of *N* DPs are to be extracted, the bounding box of the surface on the $\mathbf{X}_G$-$\mathbf{Y}_G$ plane is firstly segmented into *N* grids and all the SPs of the surface will fall into these grids. In each grid that has SPs inside, the centroid of the sampled points is calculated and the SP that is the nearest to the centroid is selected as a DP.

As the surface may have an irregular shape, not all the grids will have SPs inside. After one round of DP selection, the number of empty grids is recorded as N_e. Then sampling will be carried out again with N_e grids. This process will be repeated till there is no empty grid left. An example is shown in Fig. 3.5 showing the running of this algorithm to the first *eAR* of T_5 (see Fig. 3.2). A total number of 100 DPs are extracted in 3 passes (see Figs. 3.5a-c) with the final DPs shown in Fig. 3.5d.



(a) DPs sampled in 1st round (b) DPs sampled in 2nd round

(c) DPs sampled in the 3rd round (d) 100 sampled DPs selected from SPs

Figure 3.5 Grid sampling process to select DPs from SPs

At each DP, a cutter posture is selected from the A-map to calculate its MCP. The number of DPs is selected as 100 for each surface. The total tool-path length *TP* is obtained from the actual tool-paths. A complete training data set extracted from a cutter/surface combination indexed as *j* is given as $\left\{MCP_j^0, MCP_j^1, ..., MCP_j^{99} \rightarrow TP_j\right\}$.

3.4.2. Building and training the neural network

The classic feed-forward back-propagation neural network (BPNN) has been chosen in this study. It is implemented with the MATLAB Neural Network Toolbox. The variance in our training sets is quite large. For faster convergence, all the inputs (*MCPs*) from the training set have been normalized to the range of [-1, 1] (the same normalization setting should be applied to the inputs from the testing data). In the toolbox provided by MATLAB, the error of the network is calculated as the mean absolute error (MAE). Suppose there are *m* sets of training data. Let the neural network's prediction of tool-path length be denoted as *TPN* and the real tool-path length be denoted as *TP*, the MAE is calculated as:

$$\text{MAE} = \frac{1}{m}\sum_{i=1}^{m}\left|TPN_i - TP_i\right| \tag{3.6}$$

During the training, the error goal is set as 0.0, which can never be reached, so that the capabilities of the network can be exploited. The maximum number of training epochs is set as 20,000, which as our training result shows, is enough for the networks to converge.

In the proposed NN, only 1 hidden layer is used. To determine the number of hidden neurons, a trial-and-error approach is used. Neural networks with 3, 7, 13, 25 and 40 hidden neurons were constructed and trained. As the tool-path length estimation algorithm is intended for multi-cutter selection, it is the relative prediction error instead of the absolute error that is more significant: The same level of absolute error may seem acceptable for a set of long tool-paths but unacceptable for a set of shorter tool-paths. For this reason, the best network is selected as the one that produces the minimum mean relative error MRE, which is calculated as:

$$\mathrm{MRE} = \frac{1}{m}\sum_{i=1}^{m}\frac{\left|TPN_i - TP_i\right|}{TP_i} \tag{3.7}$$

Training results show that the neural network with 13 hidden neurons produces the smallest MRE.

The transfer functions on each layer of the neural network also play a vital role in determining NN's performance. The transfer function for the hidden layer is selected as sigmoid while the transfer function for the output layer is selected as linear. In the BPNN, the amount of change in the connection weights is proportional to the difference between the desired output and the actual output, controlled by the learning rate. For guaranteed convergence, a small value of 0.05 is selected for learning rate. The NN's structure is shown in Fig. 3.6 and the parameters used in the NN are summarized in Tab. 3.1.

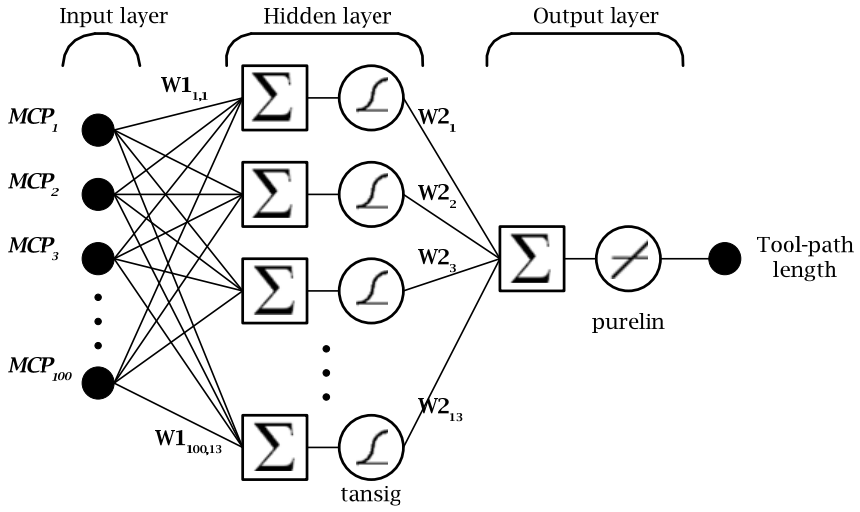


Figure 3.6 Structure of the proposed neural network

Table 3.1 Parameters of the neural network

Parameters	Input neurons	hidden layer	hidden neurons	Learning rate	Training algorithm	Transfer function	*Max epoch*
Values	100	1	13	0.05	Resilient BP	*tansig*+ *linear*	20,000

To generate the training data, 15 surfaces were prepared. For each surface, several candidate multi-cutter sets were selected. Each cutter set is coupled with a randomly selected cutting direction. For each cutter set, the surface was partitioned into several regions, each with its allocated cutter. One set of training data (100 inputs vs. 1 output) can be extracted from each cutter/region combination. We managed to have collected 302 sets of training data. Moreover, another 5 machining surfaces were created as testing samples, and 112 sets of testing data were generated. These testing data will not be exposed to the NN during the training process. Snap views of these surfaces prepared for training and testing are provided in Fig. 3.7.

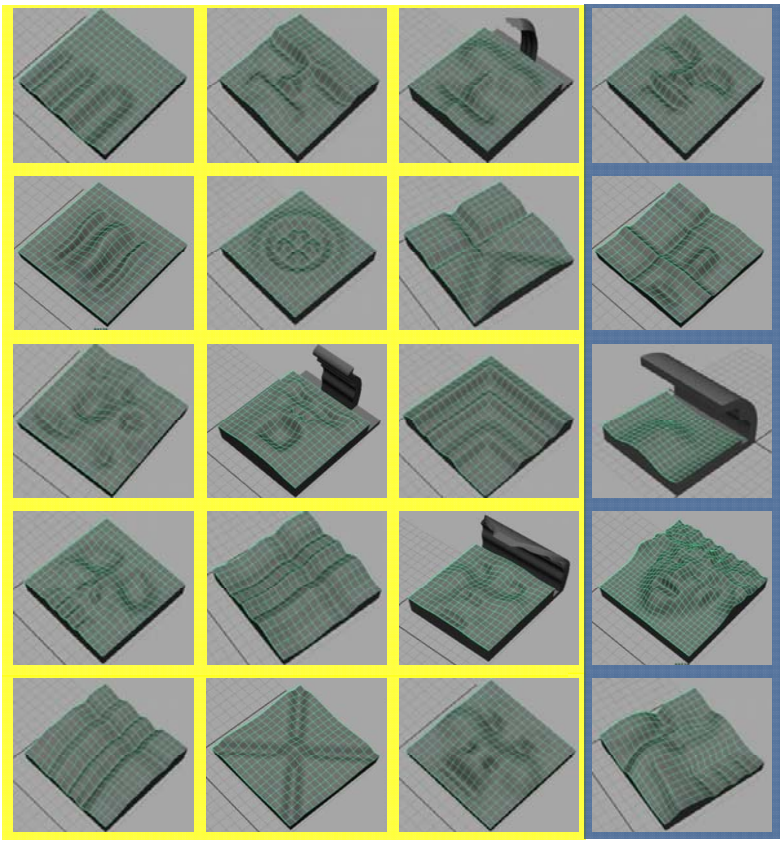


Figure 3.7 Machining surfaces for NN training and testing

The training curve (MAE vs. epochs) of the chosen NN is shown in Fig. 3.8. The absolute error (AE) and relative error (RE) for all the training samples are shown in Fig. 3.9a. After training, testing is carried out using the normalized inputs from the testing data. The AE and RE for testing data are shown in Fig. 3.9b. The training and testing results are also summarized in Tab. 3.2.

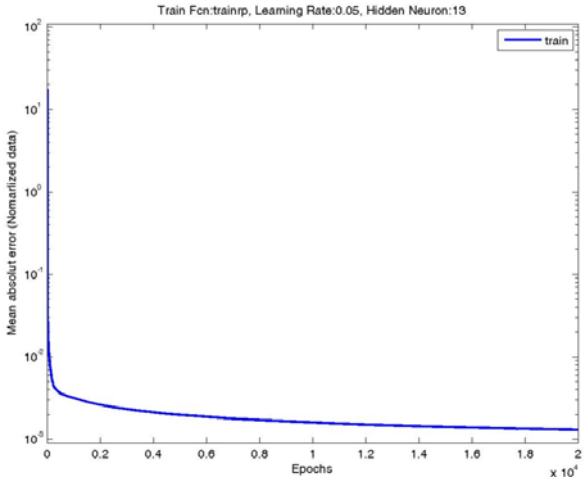
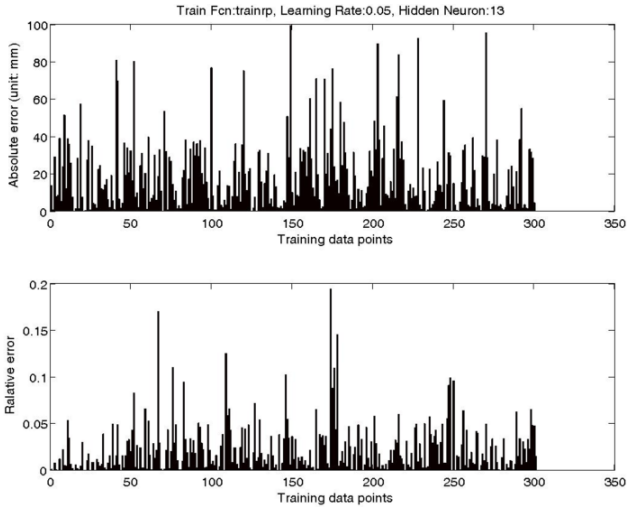


Figure 3.8 Training record for NN with 13 hidden neurons (MAE vs. Epochs)

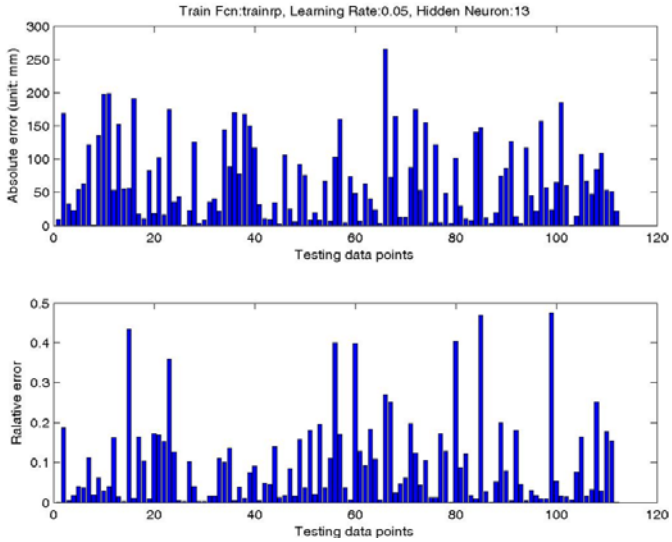
Table 3.2 Performance of the network with training and testing data

Parameters	MAE	MRE	*Max* AE	*Max* RE	*Samples with* RE>20%
Training	19.19 mm	0.017	99.2 mm	0.152	N/A
Testing	84.25 mm	0.143	275.8 mm	0.472	10/112

Regarding the reliability of the developed NN, the prediction result is deemed unacceptable when the relative error exceeds 20%. Out of the 112 testing data sets, only 10 have RE beyond 20%. It is observed that the tool-paths with RE>20% are relatively short. They all fall inside the range [54.2, 1024.5] while the longest tool-path in the training set stands at 30975.2 mm. This is probably because during training, the weights and bias of the network are updated based on the MAE of the whole training set. Shorter tool-paths tend to have less influence as they are more likely to produce smaller AE, even though the corresponding RE can be quite big. However, this shortcoming will have limited negative effects for selecting the optimal cutter combination. As we have put a size limit (low bound) on *eAR*s, there cannot be many regions with very short tool-paths. Besides, these short tool-paths will only take up a small percentage of the overall tool-path length for any multi-cutter set. Therefore, we believe the resultant inaccuracies will be quite limited.



(a) Absolute and relative errors of the NN with the training data



(b) Absolute and relative errors of the NN with the testing data

Figure 3.9 Performance the neural network with training and testing data

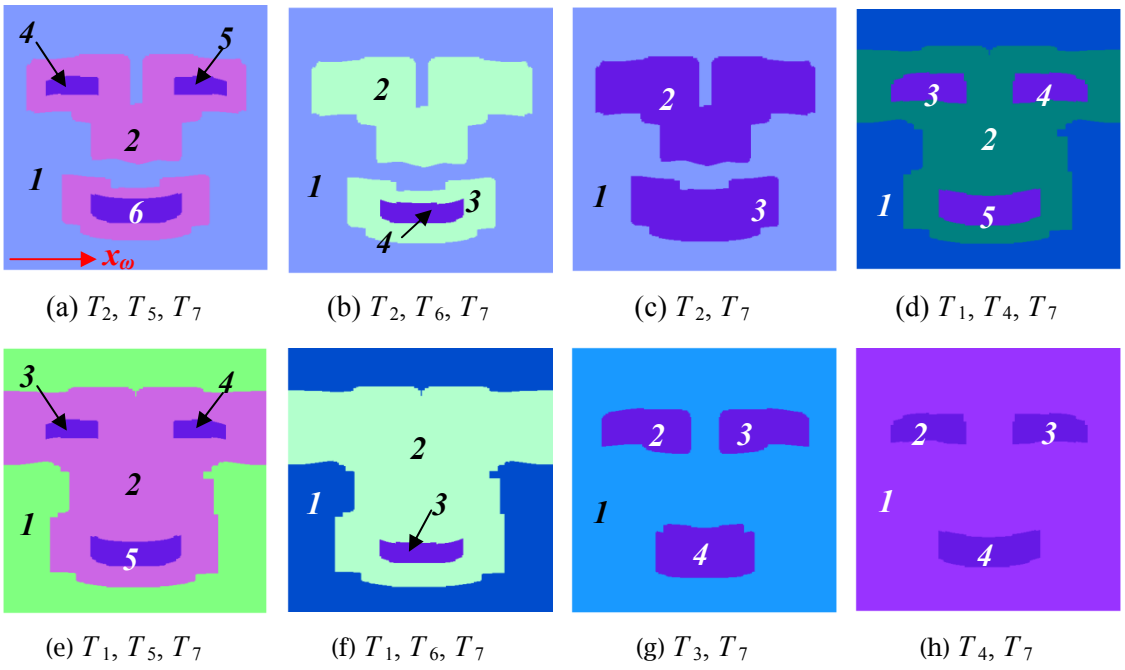
With the trained NN for tool-path length prediction, selection of optimal multi-cutter sets from the candidate cutter sets can be carried out in an efficient manner.

3.4.3. Multi-cutter Selection with NN: A Case Study

In this section, the multi-cutter selection process for the "FACE' (see Fig. 3.1) is presented as an example. The cutter library used for this mission is given in Tab. 3.3.

Table 3.3 Cutter library used for machining the 'FACE'

Cutter	R (mm)	r_f (mm)	L (mm)	Cutter	R (mm)	r_f (mm)	L (mm)
#1	10	0.5	60	#5	3	0.5	60
#2	8	0.5	60	#6	2.5	0.2	60
#3	6	0.5	60	#7	1.5	0.2	60
#4	4	0.5	60				



(a) T_2, T_5, T_7 (b) T_2, T_6, T_7 (c) T_2, T_7 (d) T_1, T_4, T_7

(e) T_1, T_5, T_7 (f) T_1, T_6, T_7 (g) T_3, T_7 (h) T_4, T_7

Figure 3.10 Machining regions for all candidate multi-cutter sets

Based on accessibility analysis, T_7 is the only cutter accessible to the whole surface. For candidate cutter set generation, the limit on MR for an *eAR* is set as 20%. Allocation of machining regions is shown in Fig. 3.10. For a clearer view, regions allocated to different cutters are plotted with different colors. Within a multi-cutter set, all the regions are numbered for easy reference. The cutting direction $\boldsymbol{x}_\omega$ is selected to be along one of the boundaries of the 'FACE', as shown in Fig. 3.10a. For each cutter/region combination, 100 DPs are extracted and the corresponding MCPs calculated.

The trained NN is then taken to estimate the tool-path lengths for all the regions of each candidate cutter set. The overall tool-path length for a cutter set is thus

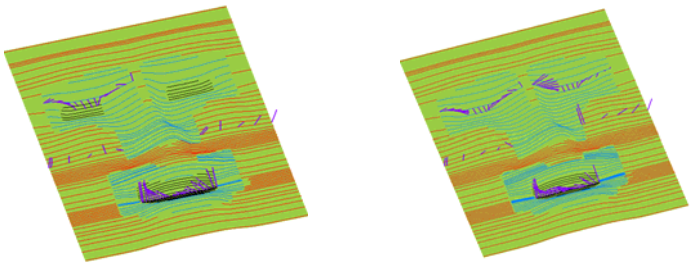
obtained. On the other hand, tool-path generation is carried out along $\boldsymbol{x}_\omega$ to obtain the accurate lengths of the tool-paths for each cutter/region combination. During tool-path generation, cutter postures are selected following the same heuristic for MCPs. The results of the "actual" and "estimation" for each multi-cutter set are given in Tab. 3.4.

Table 3.4 Tool-paths lengths: "actual" and "estimation" (unit: mm)

CSet	Reg. #	Reg. 1	Reg. 2	Reg. 3	Reg. 4	Reg. 5	Reg. 6	Total	***Rank***
T_2, T_5, T_7	*Actual*	3438.7	1481.8	592.8	280.1	N/A	N/A	5793.4	**1**
	NN	3647.1	1603.6	697.1	314.7	N/A	N/A	6262.5	2
T_2, T_6, T_7	*Actual*	3438.7	1045.8	554.9	214.3	226.8	321.6	5808.5	2
	NN	3647.1	942.1	694.4	184.5	164.8	328.2	5961.1	**1**
T_2, T_7	*Actual*	3438.7	2047.2	1035.5	N/A	N/A	N/A	6521.4	3
	NN	3647.1	2388.5	1217.4	N/A	N/A	N/A	7253.0	4
T_1, T_4, T_7	*Actual*	1379.3	4833.7	235.4	251.2	335.1	N/A	7034.6	4
	NN	1269.0	4547.5	195.5	275.5	286.4	N/A	6573.9	3
T_1, T_5, T_7	*Actual*	1379.3	5915.7	214.3	226.8	321.6	N/A	8057.7	5
	NN	1269.0	6510.2	244.5	284.8	198.2	N/A	8506.7	5
T_1, T_6, T_7	*Actual*	1379.3	6656.9	280.1	N/A	N/A	N/A	8316.3	6
	NN	1269.0	7330.8	310.5	N/A	N/A	N/A	8910.3	7
T_3, T_7	*Actual*	7225.4	438.3	415.1	754.7	N/A	N/A	8833.5	7
	NN	7308.2	395.2	401.2	690.1	N/A	N/A	8794.7	6
T_4, T_7	*Actual*	9053.1	235.3	251.2	335.1	N/A	N/A	9874.7	8
	NN	10007.1	195.5	275.5	286.4	N/A	N/A	10764.5	8

In Tab. 3.4, column "*Rank*" shows the ranking of multi-cutter sets based on "actual" and "estimation" tool-path lengths. It can be seen that the developed NN can predict tool-path length with a certain level of accuracy. This level of accuracy may not be enough to identify the cutter set with the shortest tool-path length every time, but is enough to make sure the one that gets selected is one of the best candidates. In our example, the actual tool-path length of optimal cutter set ($\{T_2, T_5, T_7\}$) is quite close to the one that gets selected by NN ($\{T_2, T_6, T_7\}$).

Finally, for completeness, the tool-paths of using these 2 cutter sets are shown in Fig. 3.11. For a clearer view, the postures of only a few CLs are plotted.



(a) $\{T_2, T_5, T_7\}$ (b) $\{T_2, T_6, T_7\}$

Figure 3.11 Tool-paths of using the top 2 multi-cutter sets

3.5. Summary

In this chapter, the complete algorithms for multi-cutter selection in 5-axis machining of sculptured surfaces are described. It starts with a comprehensive accessibility evaluation for all the available cutters. During this process, the machining surface is discretized into a set of SPs and A-maps are constructed for each cutter at all the SPs. For single cutter or multi-cutter selection, the key is to estimate the tool-path length without generating the actual path. To this end, the previously developed heuristic-based method is firstly introduced which suffers from poor accuracy in terms of tool-path estimation.

In this study, the tool of NN is used for tool-path length estimation of any given cutter/surface combination. The method starts from selecting a fixed number of evenly distributed SPs from the region and calculating the machining characteristic parameter at every DP to form the input for the NN. The NN has been extensively trained and tested. Testing results suggest that the NN produces the level of accuracy that is adequate for selecting the optimal/near-optimal multi-cutter set with shortest overall tool-path length. This NN can be used for both single- and multi-cutter selection.

CHAPTER 4

WORKFLOW FOR GENERATION OF CL PATHS

In the proposed process planning system, tool-path generation takes place after one or several cutters are selected for machining a surface on a part. At the beginning, only cutter contact (CC) points are generated, which are the points of contact between the cutter and the target surface. Tool-paths made up of CC points only are called *CC paths*. At each CC point, a cutter posture needs to be assigned. Given a CC point and the corresponding cutter posture, the location of the cutter center can be worked out. The combination of a cutter center location and the cutter posture makes a *cutter location* or CL in short. The tool-paths made up of CLs are referred to as the *CL tool-paths* or CL paths, which carry all the information needed to describe the cutter's movement in the workpiece frame. Finally, based on the configuration of the specific 5-axis machine used, the CL paths are converted to NC codes via post-processing. Tool-paths at this stage are referred to as NC tool-paths, which are the final product for tool-path generation. NC tool-paths are made up of a series of joint locations. The joints will thus guide the cutter to follow the CLs specified by CL paths. Chapters 4 to 7 of this thesis are dedicated to the subject of tool-path generation and optimization. This chapter introduces the workflow of CL path generation covering the following aspects: 1) generation of CC points on a single tool-path and 2) the heuristic for selecting cutter posture from the A-map with near-maximum CSW, and 3) determination of path intervals. It has to be pointed out that, due to the heuristic nature of the posture determination algorithm, the tool-paths produced are called *preliminary* tool-paths that are in a non-optimal state. Methods on correction,

improvement, and optimization of these *preliminary* tool-paths will be introduced in Chapters 5~7.

4.1. Background

The algorithm for CL path generation is intended for one cutter/surface combination. Therefore, there is not much difference between single-cutter and multi-cutter modes. The inputs for CL path generation include the machining surface (region), the cutter dimension and the cutter's A-maps at all the SPs. The outputs are complete tool-paths for machining the surface/region in the form of CL data. Due to the discrete nature of tool-paths, deviations will be created between the cutter's trajectory and the machining surface. At the same time, a scallop will be created between CC paths. These are all sources for surface error, which should stay below the surface finish tolerance. Hence, the surface finish tolerance, plus the requirement for interference avoidance at all the CC points, is taken as the constraints for the whole tool-path generation process. Meanwhile, the generated tool-paths could be optimized regarding several objectives, such as machining efficiency and tool-path smoothness.

The tool-path pattern used in the proposed system is iso-planar, also known as parallel cutting. Iso-planer tool-paths are generated by intersecting the given surface with multiple parallel cutting planes. Over the years, study on iso-planar tool-path generation has received much attention, especially for composite (Cho *et al.,* 2000) or trimmed surfaces (Hatna *et al.,* 2000; Yang *et al.,* 2003). Although many novel tool-path patterns have been proposed in the recent years (Makhanov, 2010), the iso-planar pattern is still perhaps the most widely used in CAM software and by researchers owing to its robustness and simplicity according to Apro (2008).

A problem unique to iso-planar pattern is the selection of an optimal cutting direction. However, such a problem has often evaded the attention of researchers. In

existing literatures on iso-planar tool-paths, researchers mainly focus on generating tool-paths with high level of efficiency and accuracy. Meanwhile, it is a common practice to assume that a path direction is already specified by the user (Ding *et al.,* 2003; Park and Choi, 2000; Xu *et al.,* 2002). Although there is little literature on cutting direction selection regarding a whole machining surface/region, locally optimal cutting direction at a point has been identified by many researchers to be used in a group of tool-path generation methods based on field theories, such as vector field method proposed by My *et al.* (2005) and Stanislav (2007) or machining potential field method by Chiou and Lee (2002). The idea behind such methods is to identify an optimal cutting direction at each surface point. In this way, the surface can be partitioned into several machining regions by clustering the points with similar cutting directions, for which different tool-path generation strategies can be applied. Although the local approach is not directly applicable to cutting direction selection for iso-planar tool-paths, it provides valuable insights for the task. In our previous work, a heuristic-based method was proposed to select the optimal cutting direction for a machining surface to produce *smooth* cutter movement. The possible cutting directions at a point are firstly generated through uniform sampling. For each sampled cutting direction, a special parameter called *posture change rate* (PCR) is calculated based on information collected at all the sampled points. The direction with the minimum PCR would be selected as the cutting direction to produce smooth cutter movements (Li and Zhang, 2006). This method is proved to be able to produce satisfactory results and is adopted in the proposed process planning system for cutting direction selection.

Another important issue to address during CC tool-path generation is determination of cutter postures. Two groups of methods exist for this category. The

first takes a trial-and-error approach. Knowledge of tool-path performance is usually built into accessibility checking algorithms. The ideal posture will be identified first based on heuristic and will be subject to iterative adjustment if interference is detected (Gian *et al.,* 2003; Gray *et al.,* 2005; Hosseinkhani *et al.,* 2007). A typical example of this category is the work of Lauwers *et al.* (2003), in which tool postures are first assigned based on a heuristic to produce maximum material removal rate and ignoring all constraints. Detection and correction of machining interferences is then conducted in the joint frame of the machine. The amount of adjustment will be kept minimal to stay close to the original heuristic as long as machining interference is eliminated. For the second group, the search space for cutter postures, i.e., the accessible posture ranges, will be built at the CC points. The cutter postures can then be searched or directly selected from the accessible ranges, as in the works of Jun *et al.* (2003); Li and Zhang (2006) and Wang and Tang (2008). In the work of Jun *et al.* (2003), the feasible posture range in the form of C-space is first obtained at all the CC points. For maximized material removal rate, feasible postures are searched for along the boundary of the C-space to closely match the surface shape. After the initial tool-paths are obtained, the cutter postures are further optimized to produce better smoothness of the tool-path. This piece of work reveals an important advantage of the latter group of methods over the trail-and-error approach: with the feasible search space for cutter postures at all the CC points, optimization regarding global properties of the tool-paths, such as tool-path smoothness is made possible.

4.2. The Proposed Approach for Generation of Iso-planar Tool-paths

With the iso-planar pattern, a series of cutting planes along the cutting direction are taken to intersect the machining surface, producing a series of intersection curves representing the tool-paths. The cutting direction can be assigned by user or can be

selected using the algorithm proposed by Li and Zhang (2006). In the remaining of the thesis, it is assumed that the cutting direction $\boldsymbol{x}_w$ is along axis $\mathbf{X}_G$ of the workpiece frame. In case this is not the actual condition, a coordinate transformation will be conducted for alignment. Tool-path generation starts from the first tool-path located at a small offset Δy_0 from the surface boundary y_{max} (see Fig. 4.1). With the proposed method, tool-path generation consists of 3 steps that are carried out iteratively, which is single tool-path generation, posture determination at CC points and path interval (side-step) calculation. The steps for this process are provided as follows:

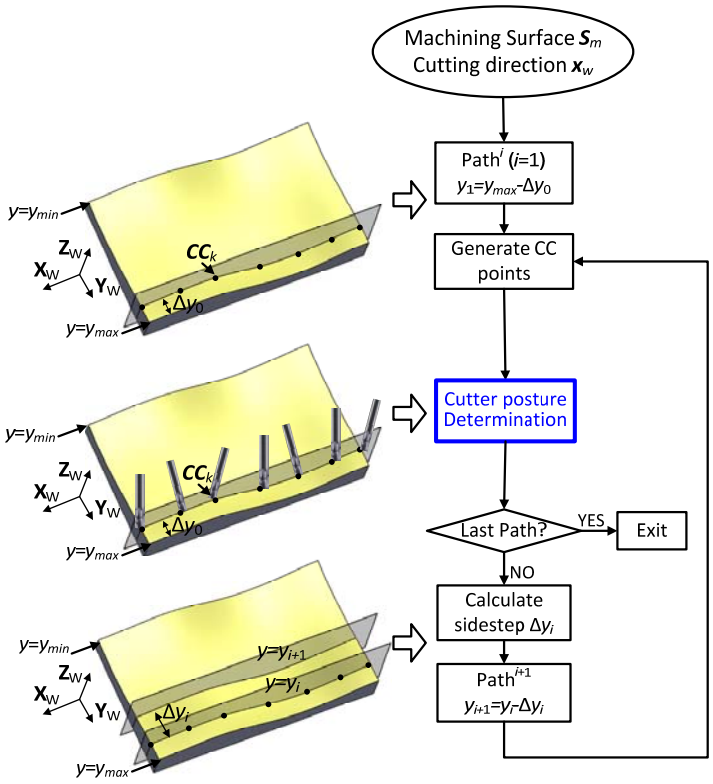


Figure 4.1 The overall approach for generation of iso-planar tool-paths

1) Set the first cutting plane to be just off the surface edge with a small distance Δy_0. The location of the current cutting plane is then $y_i = y_{max} - \Delta y_0$ ($i = 1$). The intersection curve represents the current path.

2) Generate the CC points on the current path. During this process, the deviation between the cutter trajectory (determined by interpolator) and the path is kept just below the given surface finish tolerance.
3) At each CC point, following the heuristic to produce near-maximum CSW, select cutter postures from the A-map and assign to the CC point.
4) If path^i is the last path, output the CL data and exit. Else, go to (5).
5) Calculate the CSW on the current path. Determine the side-step Δy_i between the current and the next paths. The scallop height tolerance h is imposed as a constraint.
6) Set $i = i+1$ and $y_i = y_{i-1}-\Delta y_{i-1}$. If $y_i - y_{min} \leq \mu$, set $y_i = y_{min}+\mu$ (μ is a small offset from the surface boundary, e.g. 0.1mm) and mark path^i as the last path. Go to step 2).

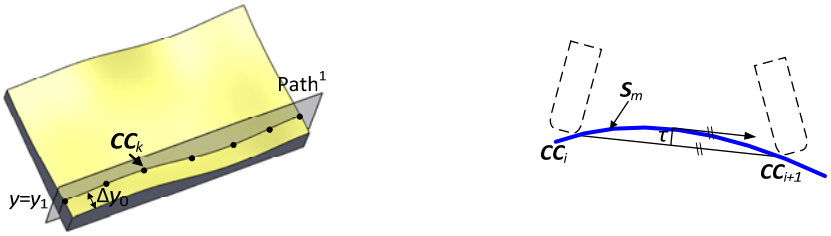
4.3. Generation of the First Tool-path

Generation of the 1st tool-path (path^1) takes place at an offset of Δy_0 from the surface boundary on plane $y = y_{max}-\Delta y_0$. The locations of next tool-paths, on the contrary, are determined based on that of the path ahead of it. The CC point generation and posture determination methods introduced in this section applies to other paths as well.

4.3.1. Generation of CC points on an individual tool-path

Generation of path^1 starts with the generation of CC points. Once the path location is determined, the first CC point can be easily calculated as the intersection point between the machining surface and the cutting plane. The rest of the CC points would be worked out iteratively. Two factors affect this process. First, CC points are discrete and the trajectory of the cutter is actually determined by interpolators on 5-axis machines. Assuming linear interpolators are used, the trajectory of the cutter between

CC points would be a straight line segment. As a result, deviations would be created between the cutter trajectory and the surface curve. Such deviations should not exceed the surface finish tolerance. This requirement puts a limit on the step-forward distance between CC points. Secondly, during actual machining, each CC point location will require one time of acceleration and deceleration of the machining joints. For shorter machining time, it is preferable to keep the number of CC point minimal. Regarding a single path, this means that the step-forward distance between CC points should be maximized.



(a) CC points on a single path (b) Chord deviation between 2 CC points

Figure 4.2 Calculate CC point locations based on surface finish tolerance

Considering both surface finish and machining efficiency requirements, determination of step-forward between CC points can be formulated as an optimization problem. The target is to make sure the step-forward distance are maximized while the constraint is that the maximum chord deviation between the cutter trajectory and surface curve stay below the surface finish tolerance τ. Given the current CC point $\mathbf{P}_i$ and cutting plane $y = y_1$, the above problem can be summarized as finding the next CC point $\mathbf{P}_{i+1}$ on plane $y = y_1$ that satisfies (see Fig. 4.2):

$$(1-\delta)\tau \le d_{\max}\left(\mathbf{P}_i\mathbf{P}_{i+1}, \boldsymbol{S}_m\right) \le \tau \tag{4.1}$$

Where $d_{max}(\mathbf{P}_i\mathbf{P}_{i+1},\boldsymbol{S}_m)$ represents the maximum chord deviation and δ is a small number such as 0.05. To carry out the optimization mentioned above, the following iterative approach is adopted:

1) Set the initial value for step-forward length (denoted as L_i) between $\mathbf{P}_i\mathbf{P}_{i+1}$ regarding local surface geometry at $\mathbf{P}_i$;
2) Search for an estimated point $\mathbf{P}_{i+1}$ from $\mathbf{P}_i$ based on the step-forward distance. Check whether the deviation between the tool trajectory and surface curve is within range (1-δ) $\tau \le d \le \tau$, where δ is a pre-defined small value (e.g. 0.05).
3) If the condition is not satisfied, the step-forward length is accordingly changed, and steps (2) and (3) are repeated till the suitable $\mathbf{P}_{i+1}$ is obtained.

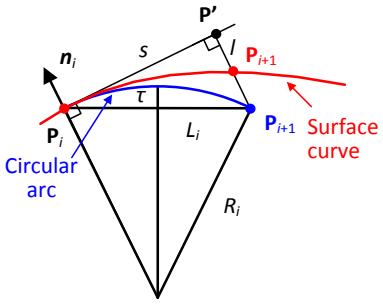


Figure 4.3 Obtain next CC point based on initial estimation of step-forward

To set the initial value for step-forward at a CC point, the local curvature should be taken into consideration. For simplicity, the surface shape in the vicinity of $\mathbf{P}_i$ is approximated by a circular arc, whose radius is the reciprocal of the curvature κ at $\mathbf{P}_i$ on plane $y=y_1$ (see Fig. 4.3). In this way, the initial estimation of step-forward distance at $\mathbf{P}_i$ is given as:

$$L_i = \sqrt{8R\tau - 4\tau^2} = 2\sqrt{\frac{\tau(2-\kappa\tau)}{\kappa}} \tag{4.2}$$

Based on the estimated value of L_i, we proceed to search for the next CC point $\mathbf{P}_{i+1}$ on the path from $\mathbf{P}_i$. The method proposed by Hwang (1992) is used. Firstly, based on the approximated circular arc, a tangent distance s is calculated as

$s = (R-\tau)L_i/R$ (See Fig. 4.3). This gives us $\mathbf{P}' = \mathbf{P}_i + s\boldsymbol{f}_i$, where $\boldsymbol{f}$ is the tangent direction at $\mathbf{P}_i$. $\mathbf{P}_{i+1}$is then obtained to be the intersection point between the surface curve and a line through $\mathbf{P}'$ and along the opposite of $\boldsymbol{n}_i$ (see Fig. 4.3), given as $\mathbf{P}_{i+1} = \mathbf{P}' - l\boldsymbol{n}_i$, where l is the distance of $\mathbf{P}_{i+1}$ from $\mathbf{P}'$ and calculated using the method proposed by Scherrer and Hillberry (1978).

After a tentative location for $\mathbf{P}_{i+1}$ is obtained, the chord deviation between the cutter trajectory and the machining surface should be evaluated. On the surface curve between $\mathbf{P}_i\mathbf{P}_{i+1}$ there is a point $\mathbf{P}$ where the curve's tangent vector parallel to $\mathbf{P}_i\mathbf{P}_{i+1}$. This means the chord deviation has reached its maxima at $\mathbf{P}$ (see Fig. 4.4).

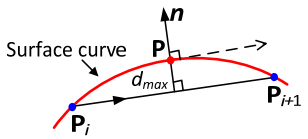


Figure 4.4 Point of maximum deviation on the surface curve

After obtaining $\mathbf{P}$, $d_{max}(\mathbf{P}_i\mathbf{P}_{i+1},\boldsymbol{S}_m)$ is calculated as:

$$d_{\max}\left(\boldsymbol{P}_i\boldsymbol{P}_{i+1},\boldsymbol{S}_m\right) = \left|(\mathbf{P}-\mathbf{P}_i) - \left[(\mathbf{P}-\mathbf{P}_i)\cdot\frac{\mathbf{P}_{i+1}-\mathbf{P}_i}{\left|\mathbf{P}_{i+1}-\mathbf{P}_i\right|}\right]\frac{\mathbf{P}_{i+1}-\mathbf{P}_i}{\left|\mathbf{P}_{i+1}-\mathbf{P}_i\right|}\right| \tag{4.3}$$

Based on $d_{max}(\mathbf{P}_i\mathbf{P}_{i+1},\boldsymbol{S}_m)$, the value of L_i should be adjusted accordingly. This process is carried out iteratively till the requirement on surface deviation is met. After the location of the new CC point $\mathbf{P}_{i+1}$ is determined, it will be taken as the current CC point to carry the search forward. This iterative process will terminate when the boundary of the machining surface/region is reached, which concludes the generation of CC points along an individual tool-path.

4.3.2. Heuristic for cutter posture determination from A-map

After the CC points are generated, cutter postures need to be assigned to them. Before this, the A-map at each CC point is obtained using the fast interpolation method introduced in Section 2.4.4. In this section, a heuristic based method to select a cutter posture producing near-maximum CSW will be introduced.

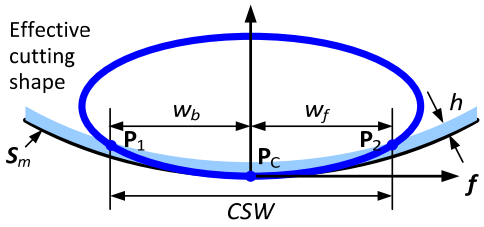


Figure 4.5 Effective cutting shape of the cutter at a surface point

When a torus cutter is placed at a point $\mathbf{P}_C$, the effective cutting shape of the cutter is the circumferential circle passing through $\mathbf{P}_C$ on the torus surface. When projected onto the plane normal to the cutting direction $\boldsymbol{x}_\omega$, the effective cutting shape takes the shape of an ellipse (see Fig. 4.5). Suppose the machining surface is offset outward along the surface normal at $\mathbf{P}_C$ by a distance of h (tolerance for scallop height), the effective cutting shape of the cutter will produce two intersection points $\mathbf{P}_1$ and $\mathbf{P}_2$ with the offset surface. The distance between $\mathbf{P}_1$ and $\mathbf{P}_2$ along the feed direction, i.e., $\overline{\mathbf{P}_1\mathbf{P}_2} \cdot \boldsymbol{f}$, would be the CSW of cutter left at $\mathbf{P}_C$.

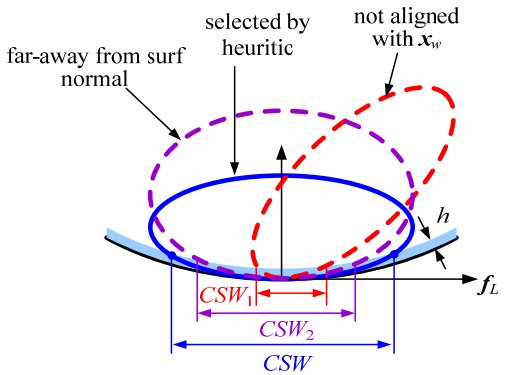


Figure 4.6 Influence of the cutter posture on cutting strip width

As the effective cutting shape of the cutter is determined by cutter posture, given a fixed cutting direction, the CSW at the point is solely determined by the cutter posture. For a posture represented by angle pair (λ, θ), increasing λ equals increasing the length of the short axis of the ellipse. As a result, the center of the ellipse is drawn further away from the CC point. The CSW produced by the cutter will therefore be reduced (see Fig. 4.6). Meanwhile, for a fixed λ, when θ is aligned with the cutting direction angle ω on the $\mathbf{X}_L$-$\mathbf{Y}_L$ plane (see Fig. 4.6), the CSW would be at its maxima w_{max}. When θ deviates from ω, the cutter will be facing the cutting direction in a tilted position, causing the actual CSW, w, to shrink to $w = w_{max}\cos(\omega - \theta)$ (see Fig. 4.6).

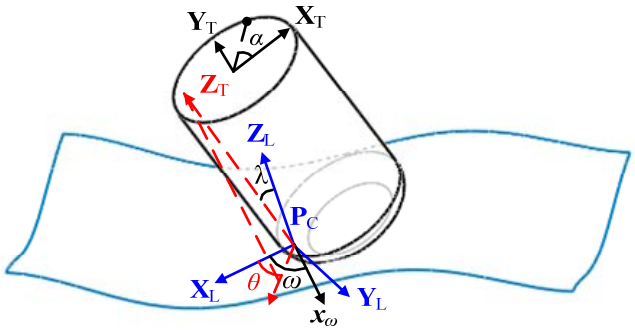


Figure 4.7 Cutting direction angle ω and angular position angle α

Based on the above observations, a heuristic named the Max-CSW heuristic is proposed for selection of cutter posture from the A-map (Li and Zhang, 2006). The purpose is to find a posture that produces a near-maximum CSW without tedious calculation. To conduct posture selection, the cutting direction $\boldsymbol{x}_\omega$ is first transformed into the local coordinate system to obtain ω (see Fig. 4.7). Then, from the A-map at the point, θ_ω, the value of θ that is closest to ω and has non-empty accessible range of λ is identified. From the accessible range of λ at θ_ω, the maximum value λ_{max} is taken to form the cutter posture $\boldsymbol{p}(\theta_\omega, \lambda_{max})$. Alternatively, the heuristic can be described as: align the cutter posture with the cutting direction as much as possible and keep the

cutter's axis as close as possible to the surface normal. With such a strategy, the posture selection problem is localized to selection of λ at a fixed value of θ.

It is worth noting that the optimality of $\boldsymbol{p}(\theta_\omega, \lambda_{\max})$ depends on the accessible range of λ at θ_ω. Only under certain circumstance, the posture identified by the Max-CSW heuristic is truly optimal. In most cases, when the accessible range of λ at θ_ω is limited by the existence of machining obstacles or local gouging, the selected posture is a sub-optimal solution only. The advantage of this approach, however, is its very low computational load, as long as the A-map is available at the surface point.

4.4. Generation of the Next Tool-paths (2nd~last)

After path[1] is generated, generation of the remaining tool-paths can be carried out in an iterative manner. Procedures for CC point generation and posture determination are the same as that of path[1]. This section will focus on the determination of the location of the next tool-path under the constraint of scallop height tolerance.

Like generation of a single tool-path, requirements for side-steps determination are two-fold. First, between two tool-paths, a scallop will be created. The height of such a scallop should stay below the scallop height tolerance h. Given the cutter postures on neighboring paths, the scallop height is solely determined by the side-step between them: the smaller the side-step, the lower the scallop. On the other hand, it is preferable to machine a region with the least number of tool-paths. For this reason, the side-step between tool-paths should be maximized. Considering these two requirements, an optimization problem is formulated: to maximize the side-step between two paths without breaching the limit imposed on scallop height.

4.4.1. Determination of path interval based on CSW

To keep the scallop height between neighboring paths below the given tolerance, a certain level of overlap should be kept between the CSWs on neighboring paths. Suppose for one CC point $\mathbf{P}_m$ on the current path, path^i, there is a corresponding CC point $\mathbf{P}_n$ on path^{i+1} along the feeding direction $\boldsymbol{f}$ (see Fig. 4.8). On the plane normal to the cutting direction (e.g., plane $x = x_m$ in Fig. 4.8), the effective cutting shapes of the cutter at $\mathbf{P}_m$ and $\mathbf{P}_n$ are shown in Fig. 4.9, with their CSWs overlapped.

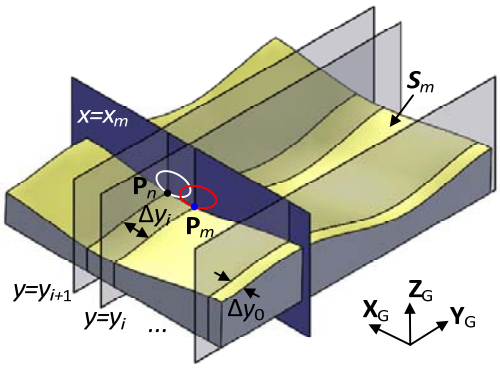


Figure 4.8 Determination of side-steps between tool-paths

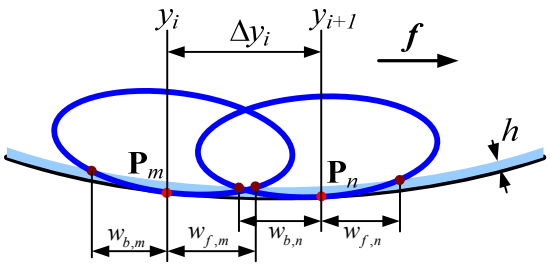


Figure 4.9 Control of scallop height with side-step

It can be seen that the scallop is formed by the intersection of effective cutting shapes on neighboring paths. The cutting strip is divided into two parts at a CC point. The part ahead of the CC point along $\boldsymbol{f}$ is the forward CSW (F-CSW, w_f) while the part behind the CC point is the backward CSW (B-CSW, w_b). To keep the scallop height between path^i and path^{i+1} below h, the side-step Δy_i must meet the following:

$$w_{f,m} + w_{b,n} > \Delta y_i \tag{4.4}$$

Such a condition should be met everywhere along the tool-path. However, the CC points on two neighboring tool-paths may not be perfectly aligned, i.e., $\mathbf{P}_m$ may not always exist on path^{i+1} corresponding to $\mathbf{P}_n$ on path^{i}. As a safe approximation, the following condition is used instead,

$$w^i_{f,\min} + w^{i+1}_{b,\min} > \Delta y_i \tag{4.5}$$

where $w^i_{f,\min}$ and $w^{i+1}_{b,\min}$ are the minimum F-CSW and B-CSW on path^{i} and path^{i+1}, respectively.

The condition shown in Eq. (4.5) only puts a high bound for the side-step. To maximize the machining efficiency, it is desirable to achieve the side-step resulting scallop height just below the tolerance. Hence, the following condition is proposed:

$$(1-\eta)\left(w^i_{f,\min} + w^{i+1}_{b,\min}\right) < \Delta y_i < \left(w^i_{f,\min} + w^{i+1}_{b,\min}\right) \tag{4.6}$$

where η is a pre-defined value controlling the level of overlapping. For better machining efficiency, a small value, e.g., 0.05, should be used.

Based the location of path^{i}, the location of path^{i+1} can be searched for by following the condition in Eq. (4.6) using an iterative approach. The initial side-step Δy_i is estimated to be the minimum CSW on the current path. The procedure is given as follows:

1) Set the initial value of Δy_i as $w_{i,min}$ on path^{i}; $y_{i+1} = y_i - \Delta y_i$.
2) Obtain the CC points on path^{i+1} and assign the postures using the Max-CSW heuristic. Calculate the CSWs at all the CC points and obtain $w^{i+1}_{b,\min}$.
3) Test the condition in Eq. (4.6). If the condition is not met, adjust Δy_i accordingly (increasing or decreasing) by a small proportion and go to step (2). Otherwise, Δy_i is found.

4) Set $y_{i+1} = y_i - \Delta y_i$. If $y_i - y_{\min} \leq \mu$, set $y_i = y_{min} + \mu$. Stop.

In step (4), μ is a small safety value, e.g., 0.1mm, which is used to avoid the situation when more than two CC points fall on the surface boundary, causing complications for identifying the starting and ending CC points.

4.4.2. Calculation of cutting strip width

In this section, an accurate method to calculate CSW for path interval determination will be introduced ('accurate' as opposed to the approximation method introduced in Section 3.3.3). When a torus cutter is placed at a point with posture $\boldsymbol{p}(\theta, \lambda)$, based on the tangential condition, the effective cutting radius, i.e., the radius of the circle passing through the CC point on the torus surface, is given as $r = (R - r_f) + \sin\lambda \cdot r_f$. The CSW of the torus cutter can thus be calculated as that of a dummy flat-end cutter of major radius r. Such an approximation is conservative as the dummy flat-end cutter is totally contained in the original torus cutter (see Fig. 4.10).

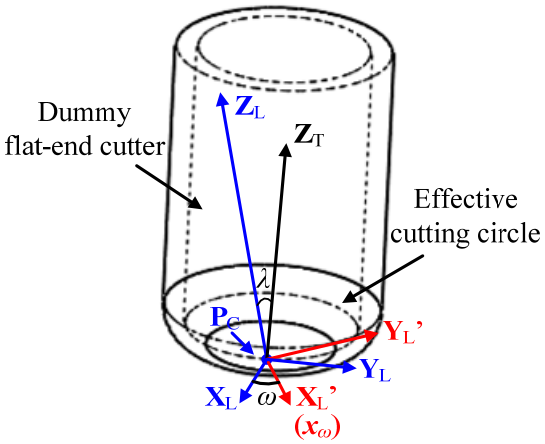


Figure 4.10 Calculation of CSW with dummy flat-end cutter

For easy reference, a new local frame is proposed at the CC point denoted as $\mathbf{O}_L$-$\mathbf{X}_L$'$\mathbf{Y}_L$'$\mathbf{Z}_L$. $\mathbf{X}_L$' is the cutting direction $\boldsymbol{x}_\omega$ transformed into $\mathbf{O}_L$-$\mathbf{X}_L\mathbf{Y}_L\mathbf{Z}_L$, $\mathbf{Z}_L$ stays the same and $\mathbf{Y}_L$' is determined accordingly (see Fig. 4.10). An angle pair (θ, λ) in the new local frame is actually $(\theta+\omega, \lambda)$ in the original local frame. The advantage with

the new local frame is that the CSW can be directly calculated on plane $\mathbf{Y}_L$'-$\mathbf{Z}_L$. In the rest of this section, the notion of angle pair (θ, λ) refers to the pair calculated in the new local frame. Let a cutter be placed at point $\mathbf{P}_C$ with posture $\boldsymbol{p}(\theta, \lambda)$, the effective cutting edge of the cutter projected onto $\mathbf{Y}'_L$–$\mathbf{Z}'_L$ plane is given by Sheltami *et al.* (1998) as:

$$\boldsymbol{E}_L(\alpha) = \begin{pmatrix} 0 \\ r\cos\lambda\sin\theta\cos\alpha + r\cos\theta\sin\alpha - r\cos\lambda\sin\theta \\ -r\sin\lambda\cos\alpha + r\sin\lambda \end{pmatrix} \tag{4.7}$$

where $\alpha \in [0°, 360°]$ gives the angular position of the point on the cutting edge (see Fig. 4.7).

To calculate the CSW, the intersection points ($\mathbf{P}_1$ and $\mathbf{P}_2$) between the effective cutting shape and the offset part surface $\boldsymbol{S}_{offset}$ on $\mathbf{Y}'_L$–$\mathbf{Z}'_L$ plane should be identified. Using the second order Tailor expansion, the surface curve on $\mathbf{Y}'_L$–$\mathbf{Z}'_L$ plane can be approximated as $z_L = \kappa_n y_L^{\,2}/2$ (DoCarmo, 1976), where κ_n is the curvature of surface at $\mathbf{P}_C$ on plane $\mathbf{Y}'_L$–$\mathbf{Z}'_L$. On the other hand, a point on the exact offset surface $\boldsymbol{S}_{offset}$ is given as $\mathbf{P}_{offset} = \mathbf{P}_s + h\boldsymbol{n}(\mathbf{P}_s)$, where $\mathbf{P}_s$ is any point between $\mathbf{P}_1$ and $\mathbf{P}_2$. Such a definition, although strict, is inconvenient for calculation, as $\boldsymbol{n}(\mathbf{P}_s)$ is always changing on the surface curve. The problem may be substantially simplified if a fixed direction is used, such as the unit normal vector $\boldsymbol{n}(\mathbf{P}_C)$ at $\mathbf{P}_C$. In this way, a point from $\boldsymbol{S}'_{offset}$ is given as $\mathbf{P}_h = \mathbf{P}_s + h\boldsymbol{n}(\mathbf{P}_C)$. It is suggested by Yoon *et al.* (2003) that it is on the safe side working with this approximation. In this way, on plane $\mathbf{Y}'_L$–$\mathbf{Z}'_L$, $\boldsymbol{S}'_h$ is given as:

$$z_L = \frac{\kappa_n y_L^2}{2} + h \tag{4.8}$$

Combining Eqs. (4.7) and (4.8), we can obtain the intersection points $\mathbf{P}_1$ and $\mathbf{P}_2$, and then the CSW at $\mathbf{P}_C$.

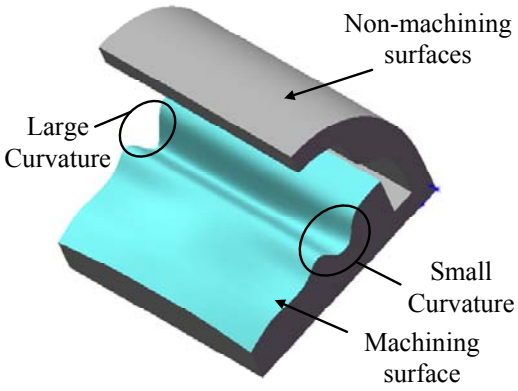
After the location of a new path is determined using the methods introduced in Sections 4.4.1 and 4.4.2, CC point generation and posture determination will be conducted in the same way as for path[1]. The whole process will stop after the last path is generated.

4.5. Case Study

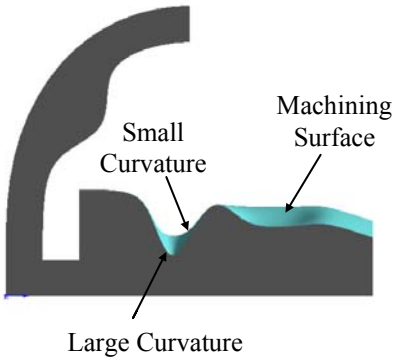
A case study is shown here to illustrate the effectiveness of the proposed heuristic-based tool-path generation algorithm. The machining surface, shown in Fig. 4.11a and b, is machined under both single and multi-cutter modes. The machining surface is quite complicated with non-machining surfaces acting as obstacles and critical areas. The available cutters are shown in Table 4.1. After cutter selection, the optimal cutter set consists of 2 cutters: T_5 and T_9, in which T_9 is the only accessible cutter. The cutting direction is selected to be along the $\mathbf{X}_G$ axis and the tool-paths of the 2 cutters are shown in Fig. 4.11c. To make a comparison, tool-paths are also generated using a single cutter T_9 as shown in Fig. 4.11d. The multi-cutter tool-paths have a total of 69 paths with a total length of 6732.88 mm (T_5: 33 paths with a length of 3306.16 mm, T_9: 36 paths with a length of 3426.72 mm). Meanwhile, the single-cutter tool-paths have 115 paths with a total length of 11532.3 mm. The machining time saving achieved by using the multi-cutter mode is tremendous. Such results demonstrate the superior machining efficiency of multi-cutter machining over single-cutter machining. Besides, machining simulations are carried for both sets of tool-paths without finding any machining interferences (see Figs. 4.11e and f).

Table 4.1Cutter library used for case study

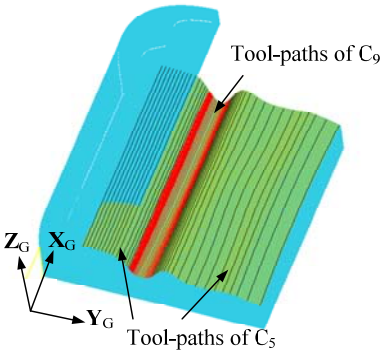
Cutter index	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	T_9
Major radius R (mm)	12	10	8	6	4	3	2	1.5	1
Minor radius r_f(mm)	0.5	0.5	0.5	0.5	0.5	0.2	0.2	0.2	0.2
Length L (mm)	85	80	70	60	50	45	45	45	45



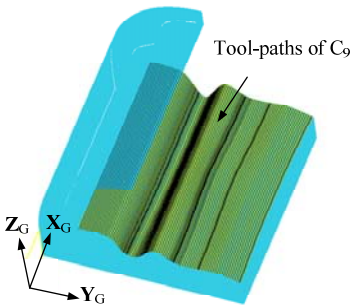
(a) Workpiece model



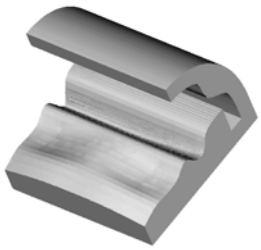
(b) Workpiece model (side view)



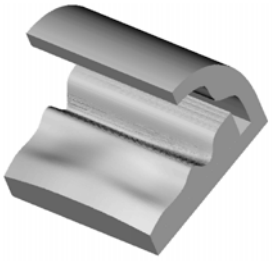
(c) Tool-paths with $\{T_5, T_9\}$



(d) Tool-paths with $\{T_9\}$



(e) Machining simulation with $\{T_5, T_9\}$



(f) Machining simulation with $\{T_9\}$

Figure 4.11 Case study-5-axis machining with single & multi-cutters

4.6. Discussion

In this chapter, the complete algorithms for generating iso-planar tool-paths are presented. The algorithm consists of three steps: generation of CC points, posture determination, and path interval determination. With the proposed methods, CC point generation is formulated as an iterative search process that produces maximized

forward-step between CC points under the constraint of surface finish quality. For posture determination, the Max-CSW heuristic is used to select from the A-map at a CC point the posture that produces near-maximum CSW. To determine the location of the next path, an iterative search algorithm is used to find the near maximum side-step, while the constraint on the scallop-height tolerance is satisfied. The major contribution for the work in this chapter is the establishment of workflow for generation of iso-planar tool-paths, which will still be followed when optimization methods are used for tool-path generation.

As shown in this chapter, extensive optimization efforts have been made regarding the tasks CL tool-path generation. Yet, one non-optimal process remains as the Max-CSW heuristic for posture determination and yet this heuristic plays a deterministic role for the performance of tool-paths. The disadvantages of the current method are two-fold. Firstly, performance of the Max-CSW heuristic is not always satisfactory. As a heuristic instead of an optimization method, it cannot find the true optimal solution. Sometimes, when faced with complicated surface shapes or machining environments, unsatisfactory results may be obtained. Secondly, as only machining efficiency is considered in the Max-CSW heuristic, it may prove ineffective when emphasis is placed on other aspects of tool-path performance (e.g., smoothness). Considering these disadvantages of the heuristic-based method, optimization methods for posture determination will be introduced in the following chapters.

CHAPTER 5

TOOL-PATH CORRECTION FOR INTERFERENCE AVOIDANCE DURING INTERPOLATION

As determined by the mechanism of CNC machining, the obtained CL paths are discrete in nature, containing only individual CC point locations and cutter postures. This means that checking for machining interferences is only conducted at the discrete CC points. The process when the cutter travels between CC points is thus not covered. Meanwhile, the proposed Max-CSW heuristic is pro-efficiency only, with no consideration for smoothness of the cutter postures. As a result, when facing complicated machining surfaces or machining obstacles, drastic posture changes may be produced between CLs, leading to possible interferences while the cutter completes its course of movement. Interference avoidance is viewed as an essential requirement for 5-axis tool-paths. This requirement also applies for the interpolation process between CC points. In this chapter, algorithms are going to be presented to detect and correct *interferences during interpolation* (IDI).

To detect IDI, the *enveloping surface* (e-surface) of the cutter's movement between CC points should be constructed. Collision query between the e-surface and the workpiece needs to be conducted to detect possible interferences. To eliminate the interferences, a straightforward approach is proposed: the posture selection heuristic remains unchanged while the step-forward between CC points is adjusted. This is based on the belief: it is safe to assume that when two CC points are placed sufficiently close, due to the similarity of local geometries, any possible IDI could be eliminated. However, when such an approach is used, the machining efficiency will

be sacrificed, as more CC points are needed for a single path. Thus, an iterative approach is used for adjusting CC points that maximizes step-forward on the premise of no IDI. In this way, the side effects regarding machining efficiency can be minimized.

5.1. Background

The e-surface for a milling cutter in motion can be viewed as the set of points that form the boundary for the cutter's swept volume. The volume contained by the e-surface is hence the maximum volume the cutter can remove in its course of movement. Currently, while the idea of e-surface has been widely used for NC simulation, limited work has been reported on its application in tool-path generation.

Application of e-surface for tool-path generation is firstly found in the area of 5-axis flank milling, in which the cutting edges are located on the cylindrical part of the cutter and the rule of movement is relatively simple. As a result, the obtained e-surfaces have much simpler shapes than that of 5-axis point milling. During machining, it usually takes only one pass for a flank miller to finish a surface, which means one single surface patch would be enough to describe the surface-shaping movements of the cutter. Machining errors can thus be easily obtained by comparing the cutter's e-surface with the machining surface. In the work of Chiou (2004), the machining errors are obtained as the distance between machining and enveloping surface along surface normal of the machining surface. Positioning strategies designed for rough and finish milling are then applied to reduce such errors. A more recent work is that of Gong and Wang (2009), in which the machining errors are obtained in a similar manner. The cutter locations are then adjusted using the least square method to produce minimized surface errors. More works of this category include those of Chiou (2004); Gong *et al.* (2005); Lartigue *et al.* (2003) and Senatore

et al. (2005). In the case of 5-axis point milling, however, multiple passes are needed for machining a single surface/region. Maximum machining errors are often produced between paths, where the e-surfaces of different paths will intersect. The computation cost would be very high if surface error is calculated using the same approach as for flank milling.

For 5-axis point milling, e-surface of the cutter has been used as a tool for interference detection. In the work of Gong *et al.* (2008), the authors produced a second-order approximation of the tool e-surface at a single CC point. The curvature information of the e-surface is then compared to that at the CC point to detect possible local gouging. Measures are then provided to adjust the posture to be interference-free. However, it is obvious that such a method is also intended for discrete postures at individual CC points. The interpolation process between CC points is not covered.

As to the construction of e-surfaces, the basic tangency principle was firstly proposed by Wang and Wang (1986) in late 1980s. After several numerical methods to solve the problem proved to be too expensive in computation, Roth *et al.* (2001) proposed a simple method to analytically calculate the sweeping profiles of toroidal cutters undergoing 5-axis movement. Later, an explicit, closed-form solution for the same problem was provided by Chiou and Lee (2002), who also considered the kinematic structure of 5-axis machines during the construction process. Finally, the method was enhanced by Weinert *et al.* (2004) and Du *et al.* (2005) with the introduction of a moving frame and simplified computation. The method for constructing e-surface in this chapter is a modified version of the methods provided by Du *et al.* (2005); Wang and Wang (1986) and Weinert *et al.* (2004).

5.2. Construction of the Enveloping Surface for Cutter Movement

Between starting and ending CLs of a single path, the cutter's enveloping surfaces (e-surfaces) make up the boundaries of the cutter's swept volume. At any instance between two CLs, the cutter would be tangent to the e-surface while being contained by the cutter's swept volume. As the cutter's movement is complicated between neighboring CC points, an explicit form does not exist for e-surfaces. Thus, an approach based on surface fitting is used here to construct the e-surfaces. First, several intermediate CLs are obtained based on the rule of movement. At these locations, the cutter's sweeping profiles (also known as grazing lines), which are the curves of tangency between the e-surfaces and the cutters, are obtained. A NURBS surface is then taken to fit these sweeping profiles as an approximation of the e-surface.

5.2.1. Cutter's movement and the moving frame

The shape of e-surfaces is closely related to the cutter's movement between CC points. For calculation of step-forward distance, it is assumed that the cutter travels linearly between CC points. The same assumption is used here. With CC point $\mathbf{P}_C$ and the cutter posture $\boldsymbol{p}$, the location of the cutter center $\boldsymbol{Q}_C$ can be calculated, given as:

$$\boldsymbol{Q}_C = \mathbf{P}_C + r_f\left(\mathbf{n}_C - \boldsymbol{p}\right) + \left(R - r_f\right)\left(\boldsymbol{p} \times \mathbf{n}_C \times \boldsymbol{p}\right) \tag{5.1}$$

where $\mathbf{n}_C$ is the surface normal at $\mathbf{P}_C$. For two neighboring CLs, let their corresponding cutter center locations can be denoted as $\boldsymbol{Q}_s$ and $\boldsymbol{Q}_e$. For calculation of e-surface, it is assumed that the cutter center travels on the straight line $\boldsymbol{Q}_s\boldsymbol{Q}_e$ at a constant speed. Based on such an assumption, at a time instant t during the cutter's travel from $\boldsymbol{Q}_s$ to $\boldsymbol{Q}_e$ ($0 < t < 1$), the location of the cutter center is given as:

$$\boldsymbol{Q}(t) = \boldsymbol{Q}_s + t\left(\boldsymbol{Q}_e - \boldsymbol{Q}_s\right) \tag{5.2}$$

The velocity of the cutter center can also be obtained as (in the workpiece frame):

$$\boldsymbol{V}_0(t) = \boldsymbol{Q}_e - \boldsymbol{Q}_s \tag{5.3}$$

For the sake of convenience, sweeping profiles are calculated in a moving frame $\mathbf{O}_E$-$\mathbf{X}_E\mathbf{Y}_E\mathbf{Z}_E$, defined based on the cutter posture and the velocity of the cutter center: $\mathbf{O}_E$ is coincident with the cutter center; $\mathbf{Z}_E$ is along the cutter axis; $\mathbf{Y}_E$ is the projection of the cutter center velocity $\boldsymbol{V}_0$ onto the cutter bottom plane and $\mathbf{X}_E$ is determined accordingly (see Fig. 5.1).

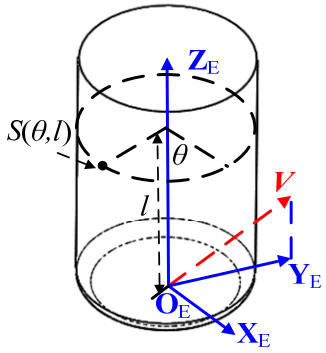


Figure 5.1 Definition of moving frame and point on cutter surface

Referring to the global frame $\mathbf{O}_G$-$\mathbf{X}_G\mathbf{Y}_G\mathbf{Z}_G$, the orientations of the moving frame are defined by three angles, i.e., the roll-pitch-yaw angles denoted by γ, β and α respectively. γ, β and α represent the amount of rotation of the moving frame about $\mathbf{X}_G$, $\mathbf{Y}_G$ and $\mathbf{Z}_G$, respectively (see Fig. 5.2). The rotational matrix between these two frames are given as:

$$\overset{G}{E}\boldsymbol{R}(\alpha,\beta,\gamma) = \begin{bmatrix} c\alpha c\beta & c\alpha s\beta s\gamma - s\alpha c\gamma & c\alpha s\beta c\gamma + s\alpha s\gamma \\ c\alpha c\beta & s\alpha s\beta s\gamma + c\alpha c\gamma & s\alpha s\beta c\gamma - c\alpha s\gamma \\ -s\beta & c\beta s\gamma & c\beta c\gamma \end{bmatrix} \tag{5.4}$$

where s and c represent *sin* and *cos* functions. Corresponding to a certain group of γ, β and α, the cutter posture is given as:

$$\boldsymbol{p}(\alpha,\beta,\gamma) = \begin{bmatrix} c\alpha s\beta c\gamma + s\alpha s\gamma \\ s\alpha s\beta c\gamma - c\alpha s\gamma \\ c\beta c\gamma \end{bmatrix} \tag{5.5}$$

which is last column of the transformation matrix.

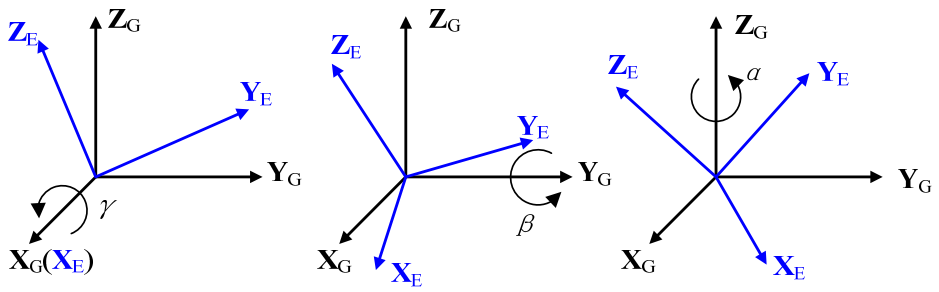


Figure 5.2 Roll-pitch-yaw angles of the moving frame regarding the workpiece frame

When the cutter center travels from $\boldsymbol{Q}_s$ to $\boldsymbol{Q}_e$, the cutter's posture also changes from $\boldsymbol{p}_s$ to $\boldsymbol{p}_e$. As a result, the orientations of the moving frame $\mathbf{O}_E$-$\mathbf{X}_E\mathbf{Y}_E\mathbf{Z}_E$ also changes. For simplicity, it is assumed that the three angles change linearly between $\boldsymbol{p}_s$ and $\boldsymbol{p}_e$ with t. In this way, at time instant t ($0 < t < 1$), the three angles are given as:

$$\begin{pmatrix} \alpha(t) \\ \beta(t) \\ \gamma(t) \end{pmatrix} = \begin{pmatrix} \alpha_s \\ \beta_s \\ \gamma_s \end{pmatrix} + t \begin{pmatrix} \alpha_e - \alpha_s \\ \beta_e - \beta_s \\ \gamma_e - \gamma_s \end{pmatrix} \tag{5.6}$$

Combining Eq. (5.5) and Eq. (5.6), the cutter posture $\boldsymbol{p}(t)$ at an arbitrary time instant t can be obtained. With $\boldsymbol{Q}(t)$ and $\boldsymbol{p}(t)$, the time instant t will be sampled evenly in the range of 0~1 to obtain the intermediate CLs.

5.2.2. Find grazing points on sweeping profiles

At an intermediate CL, the sweeping profile refers to the curve of tangency between the cutter and e-surface. The points on the sweeping profiles are called *grazing* points. The approach adopted here is to obtain a group of grazing points and connect them to produce an approximation. In the work of Wang and Wang (1986), a tangency condition is proposed to identify grazing points on any moving rigid body, which states that the velocity at a grazing point should be perpendicular to the surface normal at the point.

It is worth noting that the tangency rule is invariant to coordinate transformations. As the cutter geometries are easier to describe in the moving frame,

calculation of grazing points will be conducted in the moving frame, where the tangency condition is expressed as $\boldsymbol{V}_E \cdot \boldsymbol{N}_E = 0$, where $\boldsymbol{V}_E$ is the velocity at the grazing point and $\boldsymbol{N}_E$ the corresponding surface normal. For an arbitrary point on the cutter surface, its velocity is given as (in the moving frame):

$$\boldsymbol{V}_E(\theta,l) = \boldsymbol{V}_{E,0} + \boldsymbol{\omega}_E \times \boldsymbol{S}_E(\theta,l) = {}^{E}_{G}\boldsymbol{T}\boldsymbol{V}_0 + \left({}^{E}_{G}\boldsymbol{T}\boldsymbol{\omega}_G\right) \times \boldsymbol{S}_E(\theta,l) \tag{5.7}$$

Where $\boldsymbol{V}_0$ is the velocity of the cutter center and $\boldsymbol{\omega}_G$ is the angular velocity of the cutter, both expressed in the global frame. ${}^{T}_{G}\boldsymbol{T}$ is the transformation matrix between the global frame and the moving frame. Given $\boldsymbol{Q}(t)$ and $\boldsymbol{p}(t)$, ${}^{T}_{G}\boldsymbol{T}$ is easily obtainable. Meanwhile, $\boldsymbol{S}_E$ is the location of the point on the cutter surface in the moving frame. On a torus cutter, there are 3 kinds of surfaces, cylindrical (cutter shank), toroidal (cutter torus), and flat (cutter bottom). For a revolving body such as the cutter, it is more convenient to describe the surface using θ and l, which are the circumference angle and the height of a surface point respectively as shown in Fig. 5.1. The locations and surface normal each type of the cutter surface are summarized in Tab.5.1.

Table 5.1 Surface point and surface normal as a function of (θ, l) in the moving frame

Shank	$\boldsymbol{S}_E(\theta,l) = \left(R\cos\theta, R\sin\theta, l\right),\ r_f \le l \le L$
	$\boldsymbol{N}_E(\theta) = \left(\cos\theta, \sin\theta, 0\right)$
Torus	$\boldsymbol{S}_E(\theta,l) = \left(D\cos\theta, D\sin\theta, l\right)$ $D = \left[(R-r_f) + \sqrt{{r_f}^2 - \left(r_f - l\right)^2}\right],\ 0 \le l \le r_f$
	$\boldsymbol{N}_E(\theta,l) = \left(\cos\theta\sqrt{\frac{2l}{r_f} - \frac{l^2}{r_f^2}}, \sin\theta\sqrt{\frac{2l}{r_f} - \frac{l^2}{r_f^2}}, \frac{l}{r_f} - 1\right)$
Bottom	$\boldsymbol{S}_E(r,\theta) = \left(r\cos\theta, r\sin\theta, 0\right), \left(0 < r < R - r_0\right)$
	$\boldsymbol{N}_E = \left(0,0,-1\right)$

Meanwhile, the angular velocity $\boldsymbol{\omega}_G$ of the cutter can be arranged in a skew-symmetric matrix to form the *angular velocity tensor*, given as:

$$\mathbf{\Omega} = \begin{bmatrix} 0 & -\omega_{G,z} & \omega_{G,y} \\ \omega_{G,x} & 0 & -\omega_{G,x} \\ -\omega_{G,y} & \omega_{G,z} & 0 \end{bmatrix} \tag{5.8}$$

Based on Euler's rule of infinitesimal rotation, the angular velocity tensor can be calculated as (Hibbeler, 2009):

$$\mathbf{\Omega} = \frac{d\left({}^{E}_{G}R\right)}{dt} \cdot {}^{E}_{G}R^{-1} = \left[\frac{\partial\left({}^{E}_{G}R\right)}{\partial\alpha} \cdot \frac{d\alpha}{dt} + \frac{\partial\left({}^{E}_{G}R\right)}{\partial\beta} \cdot \frac{d\beta}{dt} + \frac{\partial\left({}^{E}_{G}R\right)}{\partial\gamma} \cdot \frac{d\gamma}{dt} \right] \cdot {}^{E}_{G}R^{-1} \tag{5.9}$$

Combining Eqs. (5.4), (5.6), (5.8) and (5.9), the angular velocity of the cutter in the workpiece frame $\boldsymbol{\omega}_G$ can be obtained.

Knowing $\boldsymbol{V}_0$, $\boldsymbol{\omega}_G$ and the transformation matrix ${}^{T}_{G}\boldsymbol{T}$, based on Tab 5.1, the velocity of an arbitrary point on the cutter can be obtained. Combining Eq. (5.7) with the tangency rule, the locations of the grazing points can be calculated as shown in Tab. 5.2. In the rest of this section, the center velocity and angular velocity in the moving frame will be referred to as $\boldsymbol{V}_E = (V_x, V_y, V_z)$ and $\boldsymbol{\omega}_E = (\omega_x, \omega_y, \omega_z)$.

Table 5.2 Locations grazing points and their existing conditions

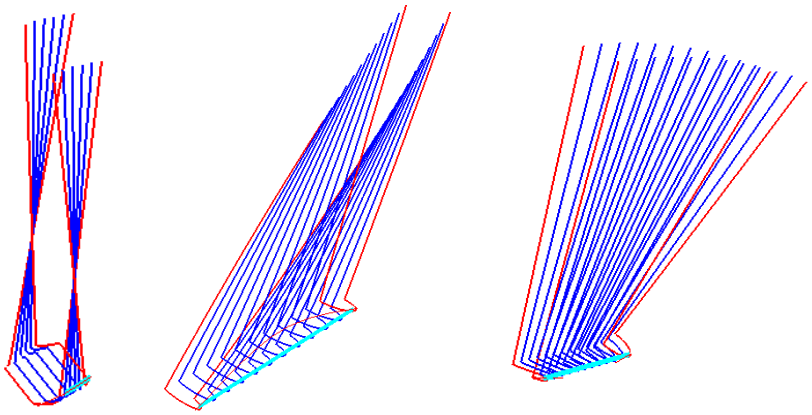
Cylinder	Grazing point	$\tan\theta = \dfrac{\omega_y l + V_x}{\omega_x l - V_y}$
	Subject to:	$l \geq r_f$
Shank	Grazing point	$\tan\dfrac{\theta}{2} = \dfrac{B \pm \sqrt{A^2 + B^2 - C^2}}{A + C}$ $A = \left(R - r_f\right)\left(r_f - l\right)\omega_y + V_x\sqrt{2r_f l - l^2}$ $B = -\left(R - r_f\right)\left(r_f - l\right)\omega_x + V_y\sqrt{2r_f l - l^2}, C = \left(r_f - l\right)V_z$
	Subject to:	$\left(A^2 + B^2 - C^2 \geq 0\right)$ $0 \leq l \leq r_f$
Bottom	Grazing point	$r = \dfrac{V_z}{\omega_y \cos\theta - \omega_x \sin\theta}$
	Subject to:	$R - r_f \geq \sqrt{\dfrac{{v_Z}^2}{\left(\omega_x r_f\right)^2 + \left(\omega_y r_f\right)^2}}$

When calculating the grazing points, it should be noted that grazing points may or may not exist on the cutter bottom, subject to the condition given in Tab. 5.2. However, if there are grazing points on the cutter bottom, they will fall into a straight line, as the equation for grazing point location indicates (see Tab 5.2). After the grazing points are identified in the moving frame, they can be transformed back to the global frame for constructing the sweeping profiles.

5.2.3. Construction of e-surfaces

Among the grazing points obtained for each intermediate CL, a certain number of them will be selected to produce an approximation for the sweeping profiles. On the cylindrical part of the cutter, regarding a certain value of l, two solutions of θ with an interval of π can be obtained. In this way, a total number of $2 \times n_{\text{cylinder}}$ points can be collected evenly at n_{cylinder} intervals of l, which will make up two grazing lines located on separate sides of cutter. For the torus part, it is very convenient to use the same approach. However, it was found that when $l \rightarrow 0$, θ may change drastically. Thus, if l is sampled evenly, the intervals between the resultant values of θ will be quite large, causing deformations of the grazing lines. Thus, for sampling of points on the torus part, the upper and lower bounds of θ will be obtained first based on the conditions given in 5.2. Then θ is sampled evenly to obtain the grazing points. Two grazing lines will be obtained on the torus part, which are connected to the two grazing lines on both sides of the cutter shank. Finally, if grazing points exist on the cutter bottom plane, only the one nearest to the cutter center will be sampled, as the sweeping profile on the cutter bottom is a straight line. If grazing points do not exist, the two sweeping profiles on the cutter torus will share a common grazing point. One way or another, a loop will be formed between the grazing lines on both sides of the cutter.

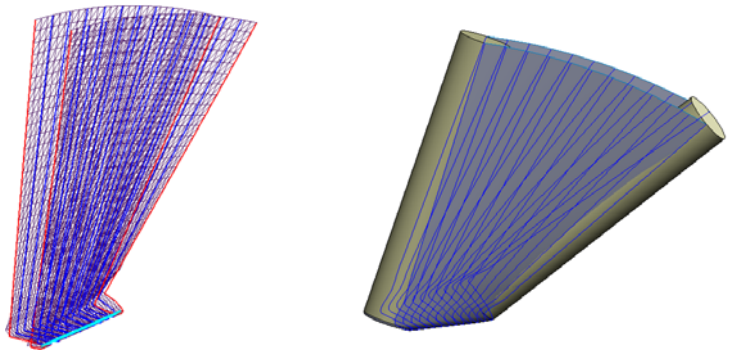
For constructing e-surfaces, the number of intermediate CLs is proportional to the distance between the starting and ending cutter center locations ($\boldsymbol{Q}_s\boldsymbol{Q}_e$) to maintain a reasonable density. The lower bound for the number of intermediate CLs is two, as a cubic NURBS surface is going to be used to fit these curves. Some examples are shown in Fig. 5.3. The sweeping profiles for the starting and ending CLs are plotted in red and the intermediate sweeping profiles are plotted in blue. The cyan straight line connects the starting and the ending CC points.



(a) (b) (c)

Figure 5.3 Examples of the sweeping profiles

After the sweeping profiles are obtained, a cubic NURBS surface will be taken to fit the sweeping profiles. The simple global interpolation approach for surface fitting provided in the work of Les and Tiller (1997) is used here.



(a) Cutter's enveloping surface (b) Cutter's swept volume

Figure 5.4 Fitting sweeping profiles with NURBS surface

For the set of sweeping profiles shown in Fig. 5.3c, the e-surfaces are constructed as an example. The generated NURBS surface is provided in a triangulated format as shown in Fig. 5.4a. For a better illustration, the generated surface as well as the cutters at the first and last CLs is imported into SOLIDWORKS®. Together they make up the swept volume of the cutter during its movement as shown in Fig. 5.4b.

5.3. Detection and Correction of IDI

During the cutter's movement between CC points, possible machining interferences include overcut and collision. Overcut refers to the situation when the cutter removes more material than specified by the surface finish tolerance and collision refers to the unintended engagement of the cutter with the machining/non-machining surfaces during its course of movement. These two kinds of interferences can both be detected by intersection checking between the e-surface and the machining/non-machining surfaces on the workpiece.

5.3.1. Detection of interferences with e-surfaces

Collision is detected when any of the machining/non-machining surfaces on the workpiece intersects with the e-surface. On the other hand, detection of overcut employs a more complicated approach. At a surface point, surface error is measured as the distance between the machined surface and the design surface along the surface normal. As the machined surface is shaped by the cutter's movement, the cutter's enveloping surface can take the place of the machined surface in calculating the surface error. At a surface point $\mathbf{P}_C$, suppose the surface finish tolerance is given as τ, the lower error bound for the machining surface, denoted as $\boldsymbol{S}_{lower}$, can be obtained by offsetting the machining surface $\boldsymbol{S}_m$ along the opposite of the surface normal $\boldsymbol{n}_C$ at $\mathbf{P}_C$

by τ. As long as the e-surface has no intersection with $\boldsymbol{S}_{lower}$, there will not be any overcut at $\mathbf{P}_C$.

Theoretically, when testing for overcut in between two CC points $\mathbf{P}_s$ and $\mathbf{P}_e$, the above procedure should be repeated for every point on the surface curve between $\mathbf{P}_s$ and $\mathbf{P}_e$, which is not realistic. To reduce computational cost, such a procedure is only conducted at the mid-point $\mathbf{P}_m$ on the curve segment between $\mathbf{P}_s$ and $\mathbf{P}_e$ with an imposed safety factor on the surface finish tolerance τ. The $\boldsymbol{S}_{lower}$ will be obtained by offsetting $\boldsymbol{S}_m$ with a distance of $\eta\tau$, where $0 < \eta < 1$ (see Fig 5.5).

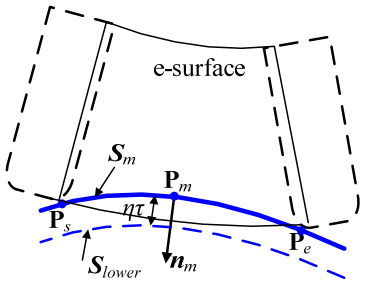
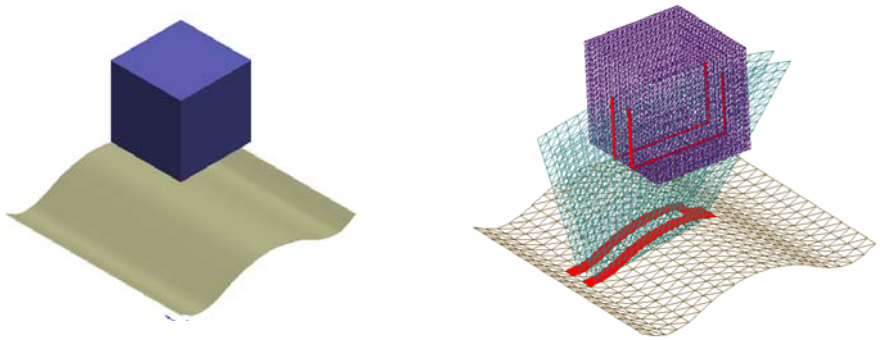


Figure 5.5 Detection of overcut between CC points

For detection of IDI, collision query between the workpiece and e-surfaces are conducted with a third-party package called RAPID (Gottschalk *et al.*, 1996), while the package is open-source and downloadable at http://gamma.cs.unc.edu/OBB/. To fit into the RAPID package, both the e-surface and the workpiece surfaces will be triangulated. An example for collision detection with RAPID between e-surfaces and a benchmark workpiece is presented in Fig. 5.6. The surface patches where the e-surfaces intersect with the workpiece are highlighted in red.

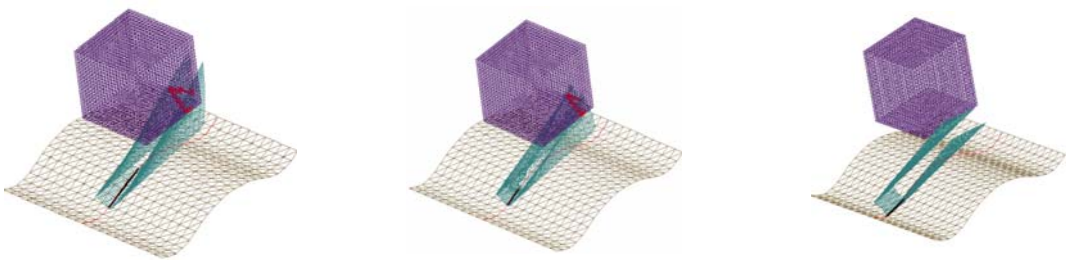


(a) Workpiece with simple obstacle (b) Collision detection result

Figure 5.6 Collision query between e-surfaces and workpiece with RAPID

5.3.2. Determination of step-forward based on avoidance of IDI

In the CL path generation method introduced in Chapter 4, cutter postures are assigned based on a fixed heuristic which only takes local surface geometries into consideration. As a result, interferences may be produced when the surface geometry changes drastically between CC points. On the other hand, due to the high level of continuity of sculptured surfaces, the local surface geometries will be more similar when two CC point are placed sufficiently close, so will be the postures assigned by heuristic. Following this logic, any possible IDI could be eliminated when the CC points are placed close enough. An example is given in Fig. 5.7, showing how collision is reduced when the distance between CC points is continuously reduced: (a) with the least number of CC points, (c) with the most number of CC points.



(a) (b) (c)

Figure 5.7 Interference elimination through reducing step-forward distance

Based on this principle, the algorithm for step-forward distance determination described in Chapter 4 is modified to incorporate IDI detection and correction. Details of the modified algorithms are provided as follows:

Algorithm: Determining position and posture at the next CC point

Input: (1) Current CC point $\mathbf{P}_i$ and its posture $\boldsymbol{p}_i$

(2) The machining surface $\boldsymbol{S}_m$ and surface error tolerance τ.

(3) All the non-machining surfaces $\boldsymbol{S}_{n,1}, \boldsymbol{S}_{n,2},\ldots, \boldsymbol{S}_{n,k}$.

Output: Next CC point $\mathbf{P}_{i+1}$ and cutter posture $\boldsymbol{p}_{i+1}$.

Begin

(1) Estimate the initial step-forward distance, denoted by D with the method provided in section 4.3.1. Set $\Delta D = 0.5D$.

(2) Obtain the next CC point $\mathbf{P}_{i+1}$ using D. Assign cutter posture $\boldsymbol{p}_{i+1}$ to $\mathbf{P}_{i+1}$. Find $\mathbf{P}_m$ on the surface curve in the middle of $\mathbf{P}_i$ and $\mathbf{P}_{i+1}$. Offset $\boldsymbol{S}_m$ along $-\boldsymbol{n}_m$ by $\eta\tau$ to obtain $\boldsymbol{S}_{lower}$.

(3) Construct the e-surface $\boldsymbol{S}_E$.

(4) Check for IDI using $\boldsymbol{S}_E$, $\boldsymbol{S}_{lower}$, $\boldsymbol{S}_m$ and $\boldsymbol{S}_{n,1}, \boldsymbol{S}_{n,2},\ldots, \boldsymbol{S}_{n,k}$.

a) Overcut detection by checking intersection between $\boldsymbol{S}_E$ and $\boldsymbol{S}_{lower}$.

b) Collision detection by collision query between $\boldsymbol{S}_E$ and $\boldsymbol{S}_m$ and $\boldsymbol{S}_{n,1}, \boldsymbol{S}_{n,2}\ldots \boldsymbol{S}_{n,k}$.

If there is IDI, go to (6).

(5) Calculate the maximum deviation d using the method introduced in 4.3.2. If $d > \tau$, go to (6). Else, go to (7).

(6) Set $\Delta D = -0.5|\Delta D|$ and $D = D+\Delta D$. Obtain $\mathbf{P}_{i+1}$ and go to step (3).

(7) If $|\Delta D|/D<0.01$, output current $\mathbf{P}_{i+1}$ and $\boldsymbol{p}_{i+1}$ and exit. Else if $\Delta D<0$, set $\Delta D = 0.5\cdot\Delta D$. Set $\Delta D = |\Delta D|$ and $D = D+\Delta D$. Obtain $\mathbf{P}_{i+1}$ and go to step (3).

End

5.3.3. Testing Examples

In this section, case studies are presented to demonstrate the effectiveness of the developed algorithms. Workpieces are specially designed to induce IDI, so that the performance of CC point generation algorithms with/without consideration for IDI can be compared. For easy reference, tool-paths generated using the method in Chapter 4 are referred to as *preliminary* tool-paths (PTPs) while tool-paths generated with the method proposed in the chapter are referred to as enveloping-surface-based tool-paths (ESTPs). A cutter of size (R = 6, r_f= 2, L = 40 mm) plus holder is chosen to complete the machining.

(a) (b)

Figure 5.8 CC tool-path and cutter postures on (a) PTP (b) ESTP

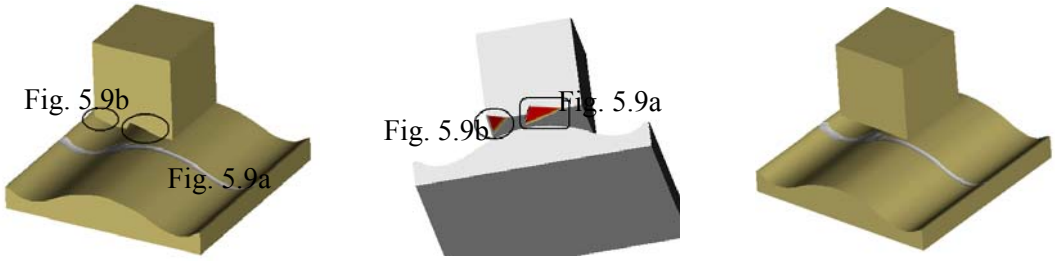
(a) (b)

Figure 5.9 Collisions in PTP between CC points (a) 1 and 2 (b) 8 and 9

The first case study takes the workpiece shown in Fig. 5.6 , for which only one tool-path is generated in the middle of the machining surface, right below the obstacle. The generated CL tool-paths are shown in Fig. 5.8 with cutter postures in purple. Enveloping surfaces are then constructed for all the neighboring CC points on PTP and interference detection is conducted to search for IDI. For the PTP tool-path, two

cases of collision are detected between CC points 1 and 2 and CC points 8 and 9 (see Fig. 5.9, CC points are number from 1 to 22 from left to right). The triangles engaged with the e-surface on the non-machining surfaces are highlighted in red.

To verify this result, simulations with these two sets of tool-paths are performed in VERICUT®. Simulation results prove that there is no interference with ESTP tool-paths (see Fig. 5.10c). For PTP tool-paths, two cases of collision are detected. The material removed by the cutter on the non-machining surface has the same location and shape as predicted using the e-surface of the cutter's movement (see Figs. 5.10a and b).



(a) Collision in PTP (b) Collisions in PTP (c) No collsion with (ESTP)

Figure 5.10 Collision detection results with VERICUT for PTP and ESTP

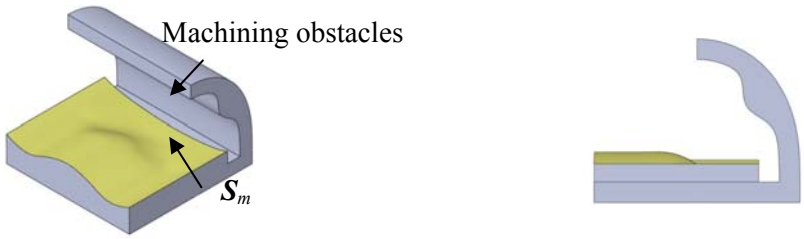
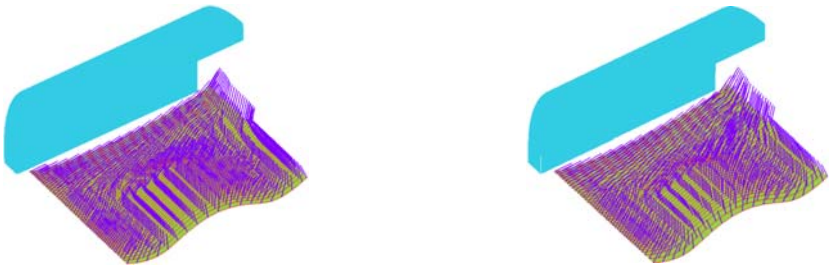


Figure 5.11 Machining surface with arch-shaped overhang

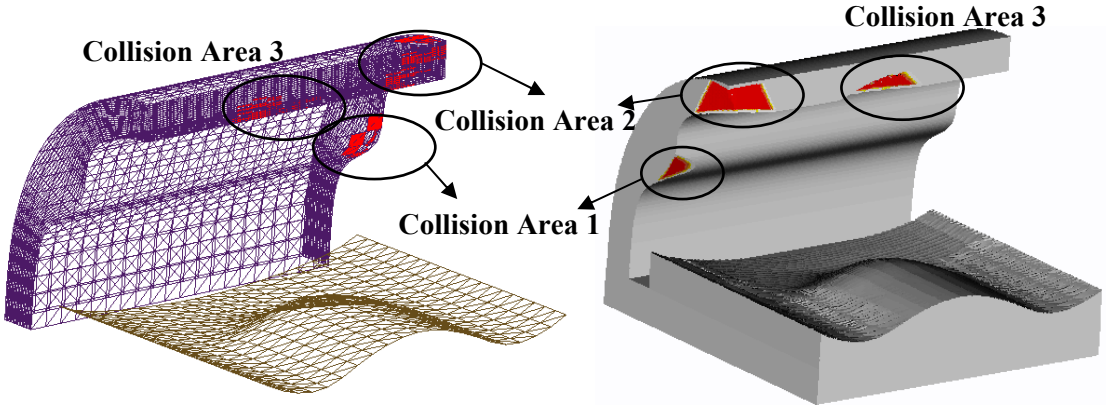
The proposed method's capability is further tested using a more complicated workpiece as shown in Fig. 5.11. It can be seen that the non-machining surfaces of the workpiece form an arch-shaped overhand above the machining surface to induce more obstacles. The generated PTPs and ESTPs are shown in Fig. 5.12.



(a) (b)

Figure 5.12 PTPs and ESTPs for 2nd test workpiece

By checking the two sets of tool-paths, it was found that there is no IDI for the ESTP tool-paths. For the PTP tool-paths, however, 17 cases of interferences were detected. For a clearer presentation, the triangular patches on the non-machining surfaces that have interference with the enveloping surfaces are highlighted in red (see Fig. 5.13). Simulation with VERICUT® also produces the same result. The locations and sizes of interferences detected by simulation are in accordance with those detected by our method. Meanwhile, for ESTP tool-paths, simulation reveals no detectable interference.



(a) (b)

Figure 5.13 Collision detected by e-surface and by VERICUT® for PTP

A complete comparison between these two sets of tool-paths is given in Tab 5.3. It can be seen that with the proposed method, the machining efficiency is slightly affected, i.e., the average step-forward step is reduced while the number of CC points and the overall tool-path length both increase. Meanwhile, the average angular

difference between postures at neighboring CC points is also reduced, meaning the tool-path is smoother. However, the most important significance of the addition of IDI detection and correction is that the proposed method can, to a large extent, generate tool-paths that are interference-free in actual machining. It is also worth mentioning that the proposed method is inevitably more computationally expensive compared to the original discrete method, due to the extra interference checking procedures. For each possible value of step-forward, k (number of non-machining surfaces)+1($\boldsymbol{S}_m$)+1($\boldsymbol{S}_{lower}$) collision queries will be needed to determine its feasibility in terms of IDI. As a result, the difference in time consumption for generating PTP and ESTP are enormous. While the tool-path generation of the Max-CSW heuristic takes only 6 minutes on a PC a 2.33GHZ CPU and 2 GB RAM, the ESTP takes more than 3 hours.

Table 5.3 Performance comparison between PTP and ESTP

Parameters	*PTP*	*ESTP*
No. of paths	44	43
No. of CC points	776	828
Overall tool-path length	4240.40	4335.12
Average Step-forward (mm)	5.83	5.24
Average Angle Change (°)	11.92	9.28
IDI	17	0

CHAPTER 6

OPTIMAL POSTURE DETERMINATION USING EVOLUTIONARY ALGORITHMS

In the existing version of the proposed process planning system, cutter posture assignment at the generated CC points on a single path is based on the Max-CSW heuristic. This heuristic has two kinds of weaknesses. Firstly, although designed to find a posture with near-maximum CSW, the heuristic does not guarantee the quality of the solution. Secondly, the heuristic lacks consideration for tool-path smoothness, which is also of great concern when evaluating the performance of tool-paths. As a result, for machining surfaces with highly varied curvatures, drastic posture changes may be produced, which could lead to poor surface quality on the machined surface and possible machining interferences during the interpolation process. It is, therefore, desirable to use a more comprehensive method to replace the Max-CSW heuristic for posture determination at CC points.

Since A-maps represent the feasible search space for cutter postures at the CC points, the posture assignment problem can be formulated as a constrained optimization problem. The optimization objective function could cover measures of machining efficiency as well as tool-path smoothness.

6.1. Background

Tool-path smoothness is considered as an important performance criterion for 5-axis tool-paths. Unsmooth tool-paths with drastic joint movements (in the NC codes) may cause the following problems in actual machining:

1) Interference occurs during the interpolation from one posture to the next, a process that is generally not covered by interference checking. The larger the joint movement, the more likely the machining interference will happen.
2) There are speed limits on the revolute joints of 5-axis machines. When the joint movement is too drastic, achieving the designed feed-rate of the machine will be hindered.

There are several causes for drastic joint movements in NC codes, the most common being unsmooth CL data. If there is large angular difference between neighboring CLs, large joint movements will most definitely be produced to carry out the cutter movement. Meanwhile, due to the non-linear kinematic structure of 5-axis machines, smooth CL data will not necessarily result in smooth NC codes. Furthermore, there are two extreme scenarios affecting the joint movement. First, all 5-axis machines have a singularity configuration, under which the primary revolute joint will lose control of the cutter orientation. When the machine is operating near the singularity point, a rapid change in the primary revolute joint up to 180° is often needed between two cutter postures with a small angular difference. Secondly, many 5-axis machines have joint limits on the revolute joints. When the revolute joints are working near the boundaries of their travel limits, a small posture change will cause the revolute joint to literally 'run out' and an axis rewinding process is needed to reset the joint to the other end of its working range. Due to existence of these possibilities, smooth CL data is only the necessary but not sufficient condition for smooth NC tool-paths.

For reasons given above, it can be seen that the angular difference between CLs does not truly measure the tool-path smoothness. Instead, we propose a new measure of tool-path smoothness in terms of the *joint travel* needed to take the cutter from one CL to the next. Smooth CL tool-paths should thus be termed as 'tool-paths

that will produce joint movement below a certain level after transformed into NC tool-paths'. This requires detecting and eliminating possible drastic joint movements during CL tool-path generation. To carry out such a task, cutter postures need to be transformed into joint variables during CL tool-path generation, which requires information of the machine configuration as an extra input. On the other hand, the machining efficiency of the tool-path can still be treated as an optimization objective for posture determination.

In recent years, many attempts have been made by researchers to optimize tool-paths regarding tool-path smoothness. For example, a greedy search approach was used Jun *et al.* (2003): at each CC point, the selected cutter posture produces the minimum posture change from the previous posture. Such an approach, though straight-forward, cannot guarantee the optimality of the solution. Ho *et al.* (2003) used a quaternion interpolation method to generate smooth tool-paths. However, the whole method is trial-and-error in nature with iterative accessibility checking on the postures generated by interpolation. Wang and Tang (2007) proposed bi-directional search algorithm in C-space to find a smooth posture sequence Limits were imposed on the angular posture changes between postures, which produced satisfactory results. Bi *et al.* (2009) proposed to use Dijkstra's algorithm to optimize the tool-path towards minimum total angular movement. Although this method could reduce total posture change, it ignores the possibility of drastic posture change between individual pairs of CC points. In our previous work, a novel tool-path topology is proposed with which the cutting direction keeps changing towards to the direction with minimum posture change. The shortcoming is that the tool postures are all selected based on a fixed heuristic, leaving no room for adjustment (Li *et al.,* 2010). Besides, a common problem shared by these methods is that posture change is evaluated in the workpiece

frame. The non-linear mapping between the workpiece frame and the machine frame as a possible cause of drastic joint travel is thus ignored.

Due to the relatively simple geometric constraints, optimization of tool-paths regarding multiple objectives is first proposed for 5-axis flank milling of ruled surfaces. The optimization objectives are usually selected as the finishing quality and tool-path smoothness, as shown in the works of Chu and Hsieh (2010); Chu *et al.* (2010); Gong and Wang (2009) and Zhu *et al.* (2010). Many optimization methods have been used, such as dynamic programming (Zhu *et al.*, 2010), particle swarm optimization (Chu and Hsieh, 2010), ant system algorithms (Chu *et al.,* 2010), etc. Although tangible improvements have been achieved, such methods cannot be directly transferred to 5-axis point milling due to more complicated geometric constraints.

The kinematic properties for different kinds of 5-axis configurations have been well studied. Tools such as inverse kinematic transformation are well established in works of Bohez (2002); She and Chang (2007); Tutunea-Fatan and Feng (2004), etc. In the area of NC tool-path optimization, Makhanov (2007) proposed a series of optimization algorithms in his works. Each of these algorithms has a specific optimization target, such as kinematic error, total tool-path length, workpiece set-up. Case studies show that these algorithms produce satisfactory results. However, Makhanov only considered machining of the so called ‘open’ surfaces and the cutter postures are determined by heuristics. Similar work includes that of Anotaipaiboon *et al.* (2006), who attempt to find the best workpiece setup orientation to minimize kinematic errors. In this work, only workpiece set-up has been optimized without involvement of the tool-paths. In the work of Lavernhe *et al*. (2008), tool-postures are optimized to produce the maximum feed rate. However, the researchers assumed that

the tool takes the same posture at all the cutter contact points, an approach which cannot fully explore the capabilities of 5-axis machines. An exception is work of Castagnetti *et al.* (2008), in which the accessible posture range is built at each CC point in the form of DAO. A gradient-based optimization method is then taken to produce a posture sequence with smooth joint movements. However, the method is also limited to optimization of a single objective with no consideration for other optimization objective like machining efficiency.

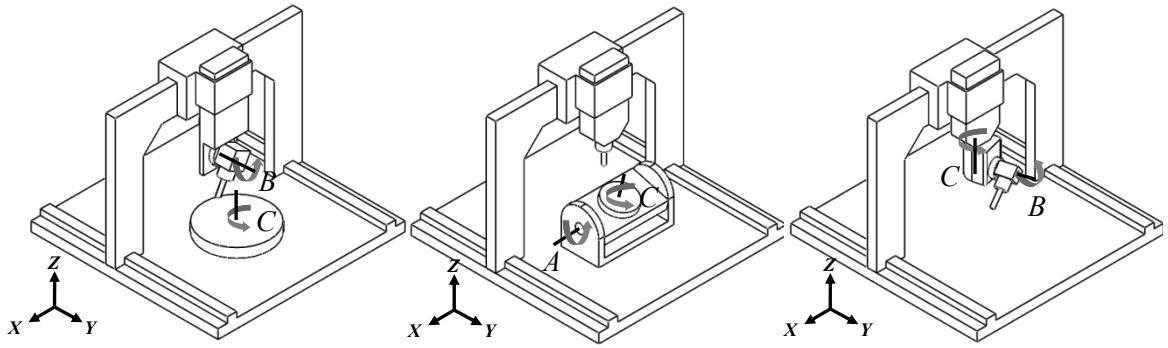
In the remaining of this chapter, the inverse kinematic transformations for 5-axis machines will be briefly analyzed first, followed by the method to calculate joint movements along a tool-path. Two approaches for tool-path optimization will be introduced. The first one, called the *repairing* approach, starts by identifying the problematic segments within a preliminary CL path generated using the Max-CSW heuristic and replace them with smooth ones. The second one takes a dual-objective optimization approach that seeks to optimize a complete CL path regarding machining efficiency and tool-path smoothness simultaneously.

6.2. Machine Configurations and Inverse Kinematic Transformation

To measure tool-path smoothness by joint movements, CL data should be transformed into the machine frame to obtain the corresponding joint locations for every CL. Such a task requires knowledge of machine configuration, which is taken as an extra input for the tool-path optimization algorithms proposed in this chapter. Knowing the cutter center location and cutter posture, the process of obtaining the corresponding joint locations regarding a specific machine configuration is called inverse kinematic transformation (IKT).

6.2.1. Inverse kinematic transformation of 5-axis machines

There are various configurations for 5-axis machines. Depending on the relative positions of the two revolute joints, configurations for 5-axis machines mainly fall into three categories: (1) Head/Head (see Fig. 6.1c, both revolute joints are connected to the spindle), (2) Head/Table (see Fig. 6.1a, one revolute joint connected to spindle and another to the machine table) and (3) Table/Table (see Fig. 6.1b, both revolute joints are connected to the machine table). Furthermore, based on the choice of axes for the two revolute joints (*A*, *B*, or *C*), the configurations of 5-axis machines can be further sub-classified, e.g., Fig. 6.1a shows a Head-Table structure with *B* (spindle) - *C* (table) axis configuration.



(a) (b) (c)

Figure 6.1 Machine structures: (a) Head/Table (b) Table/Table (c) Head/Head

Regardless of configurations, a 5-axis machine can always be viewed as a kinematic chain with three translational joints *X*, *Y*, *Z* and two revolute joints *B*, *C* (or any other combinations). The first link in the chain is the workpiece and the end effecter is the machine tool. Each joint corresponds to one time of coordinate transformation between the 2 connected links. Following this observation, coordinate systems can be assigned to each link using the Denavit-Hartenberg approach (Craig, 2005). In this manner, the influence of a machine joint on cutter location is reflected by the transformation matrix between the 2 coordinate systems on both sides of the

joint. An example is provided in Fig. 6.2 that shows the D-H system of coordinate systems constructed for a Head/Head 5-axis machine with *B/C* axis configuration.

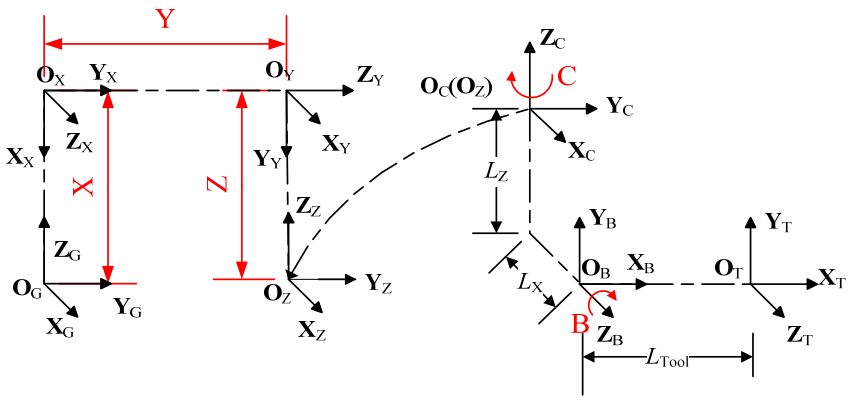


Figure 6.2 D-H frames for a Head/Head 5-axis machine with B-C axis configuration

For the coordinate systems assigned using the D-H rule, the naming of frame follows that of the machine joint in front of it. For example, $\mathbf{O}_X$-$\mathbf{X}_X\mathbf{Y}_X\mathbf{Z}_X$ is attached to the link right behind translational joint *X*. The transformation matrix ${}^{X}_{G}T$ between $\mathbf{O}_G$-$\mathbf{X}_G\mathbf{Y}_G\mathbf{Z}_G$ and $\mathbf{O}_X$-$\mathbf{X}_X\mathbf{Y}_X\mathbf{Z}_X$ is then determined by the joint variable *X*. In Fig. 6.2, L_Z and L_X represent the offset between the 2 rotational axes in $\mathbf{Z}_C$ and $\mathbf{X}_C$ directions while L_{Tool} the tool length. Meanwhile, the offsets between the translational joints and between the workpiece and the machine reference point have been absorbed by the three translational joint variables for simplicity. Based on the nature of movement for each joint (translation or rotation), transformation matrices are derived for each every pair of neighboring coordinate systems, given as (homogenous form):

$$
{}^{G}_{X}T\,{}^{X}_{Y}T\,{}^{Y}_{Z}T = \begin{bmatrix} 1 & 0 & 0 & X \\ 0 & 1 & 0 & Y \\ 0 & 0 & 1 & Z \\ 0 & 0 & 0 & 1 \end{bmatrix}
\quad {}^{Z}_{C}T = \begin{bmatrix} \mathrm{c}C & -\mathrm{s}C & 0 & 0 \\ \mathrm{s}C & \mathrm{c}C & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
\quad {}^{C}_{B}T = \begin{bmatrix} 0 & 0 & 1 & L_X \\ \mathrm{c}B & -\mathrm{s}B & 0 & 0 \\ \mathrm{s}B & \mathrm{c}B & 0 & -L_Z \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^{B}_{T}T = \begin{bmatrix} 1 & 0 & 0 & L_{Tool} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6.1}
$$

where *s* and *c* are short for *sin* and *cos* respectively. These transformation matrices multiplied over each other could give us the transformation matrix ${}^{G}_{T}T$ between the tool frame and the workpiece frame as:

$$ {}^{G}_{T}T = {}^{G}_{X}T\,{}^{X}_{Y}T\,{}^{Y}_{Z}T\,{}^{Z}_{C}T\,{}^{C}_{B}T\,{}^{B}_{T}T \tag{6.2} $$

In the tool frame $\mathbf{O}_T$-$\mathbf{X}_T\mathbf{Y}_T\mathbf{Z}_T$, the cutter posture is given as (homogeneous form) $\boldsymbol{p}_T = (0, 0, 1, 0)^T$. Assuming the cutter posture has been obtained in the global frame as $\boldsymbol{p}_G = (i, j, k, 0)^T$, the following condition can be established:

$$ \boldsymbol{p}_G = {}^{W}_{T}T\boldsymbol{p}_T \tag{6.3} $$

Combining Eqs. (6.1) ~ (6.3) gives us:

$$ \boldsymbol{p}_G = \boldsymbol{p}_G(C,B) = \begin{bmatrix} -\sin C \cos B \\ \cos C \cos B \\ \sin B \end{bmatrix} = \begin{bmatrix} i \\ j \\ k \end{bmatrix} \tag{6.4} $$

It can be seen from Eq. (6.4) that $\boldsymbol{p}_G$ is the function of the two revolute joint variables only. Given the travel limits of the two revolute joints, the two joint variables can be solved for.

The fact that the cutter posture is solely determined by the two revolute joints is no coincidence, but determined by the structures of 5-axis machines. As a general rule, the transformation matrix between tool frame and workpiece frame always come in the form (assuming the 5-axis machine has a *B/C* axis combination, other axis combinations can be used as well):

$$ {}^{G}_{T}T = \begin{bmatrix} r_{11}(B,C) & r_{12}(B,C) & r_{13}(B,C) & t_1(X,Y,Z,B,C) \\ r_{21}(B,C) & r_{22}(B,C) & r_{23}(B,C) & t_2(X,Y,Z,B,C) \\ r_{31}(B,C) & r_{32}(B,C) & r_{33}(B,C) & t_3(X,Y,Z,B,C) \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} \tag{6.5} $$

Combining Eq. (6.5) and Eq. (6.3), the cutter posture in the workpiece frame is:

$$ \boldsymbol{p}_G = \left(r_{13}\left(B,C\right), r_{23}\left(B,C\right), r_{33}\left(B,C\right), 0\right) \tag{6.6} $$

where r_{12}, r_{23} and r_{33} are all trigonometric expressions of the two revolute joint variables.

Based on Eq. (6.6), given a cutter posture, the revolute joint variables corresponding to any type of 5-axis configuration can be easily obtained. Besides, it should be noted that the example provided above is based on machines with orthogonal revolute joint axes. For the any non-orthogonal configurations, derivation will be more complicated but the general form given for ${}^{T}_{G}T$ will not change. Hence, the conclusion drawn above will not be affected.

6.2.2. Transforming CL path to joint locations

For machines with travel limits on the two revolute joints, there are several solutions of revolute joint variables corresponding to a single cutter posture. For machines without travel limits, there is an indefinite number of solutions. Post-processors of 5-axis machines usually have certain heuristics to tackle the problem of solution multiplicity. To evaluate the smoothness of a tool-path accurately, the methods to calculate the joint variables have to be in accordance with the settings of the post-processor for the machine. In our case, the post-processor takes a greedy approach, i.e., for any CL, the joint variables are selected as the pair that produces the minimum joint movement from the last joint location. Suppose there are n CC points along a tool-path and the 5-axis machine has a *B/C* axis configuration like the one shown in Fig. 6.1c, the process of converting a candidate posture sequence given as $\{\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_n\}$ to the corresponding joint variables is as follows (the method can be easily adapted for other axis combinations).

1) At the beginning, re-set the revolute joints to their zero positions, i.e., $B_0 = 0$ and $C_0 = 0$. Set $i = 1$.

2) Find all candidate solutions corresponding to $\boldsymbol{p}_i$ within the travel limit of the machine, denoted as $\left\{\left(B_i^1, C_i^1\right), \left(B_i^2, C_i^2\right), ..., \left(B_i^k, C_i^k\right)\right\}$.

3) Calculate the joint movement between each candidate solution and the last joint locations ($B_{i\text{-}1}$, $C_{i\text{-}1}$), given as $J_{i,i-1}^m = \left|B_i^m - B_{i-1}\right| + \left|C_i^m - C_{i-1}\right|$. Select the solution that produces the minimum, i.e. $J_{i,i-1} = \min\left\{J_{i,i-1}^1, J_{i,i-1}^2, ..., J_{i,i-1}^k\right\}$.

4) If $i = n$, stop and output the joint variables; else, set $i = i+1$ and go to step (2).

Finally, the maximum joint movement $J_{\max}$ along a CL path is given as $J_{\max} = \max\left\{J_{i,i-1} \middle| i = 1, 2, ..., n\right\}$.

6.3. Cutter Posture Optimization: The Overall Approach

In this chapter, two approaches for cutter posture optimization are to be introduced. The first takes a 'repairing' approach, for which an upper-limit will be set for the amount of joint movement between neighboring CLs. At a CC path, the cutter postures are first assigned by the Max-CSW heuristic. This *preliminary* tool-path is then transformed into joint locations to be checked for beyond-limit joint movements. The identified 'unsmooth' postures are grouped into *unstable CL clusters* and passed to a hybrid Particle Swarm Optimization (PSO) algorithm for repairing (smoothing). Meanwhile, the second one employs a direct dual-objective optimization approach, where a search of the optimal posture sequence for the CC points is conducted with a genetic algorithm (GA) having an objective function covering the two objectives. The workflow for tool-path generation, as well as the search constraints are the same for these two approaches, and will be introduced in the rest of this section.

6.3.1. Workflow of tool-path generation with posture optimization

The modified workflow for tool-path generation incorporating the optimization means is shown in Fig. 6.3. The tool-path generation starts with the first path, path[1], located at a small distance Δy_0 from the surface boundary. Along the CC path, cutter postures are determined using the optimization methods (instead of the max-CSW heuristic) with machine configuration as an extra input. After the cutter postures are determined, the location for the next tool-path will be searched for, where single tool-path generation will be carried out again. The complete steps are provided in the following:

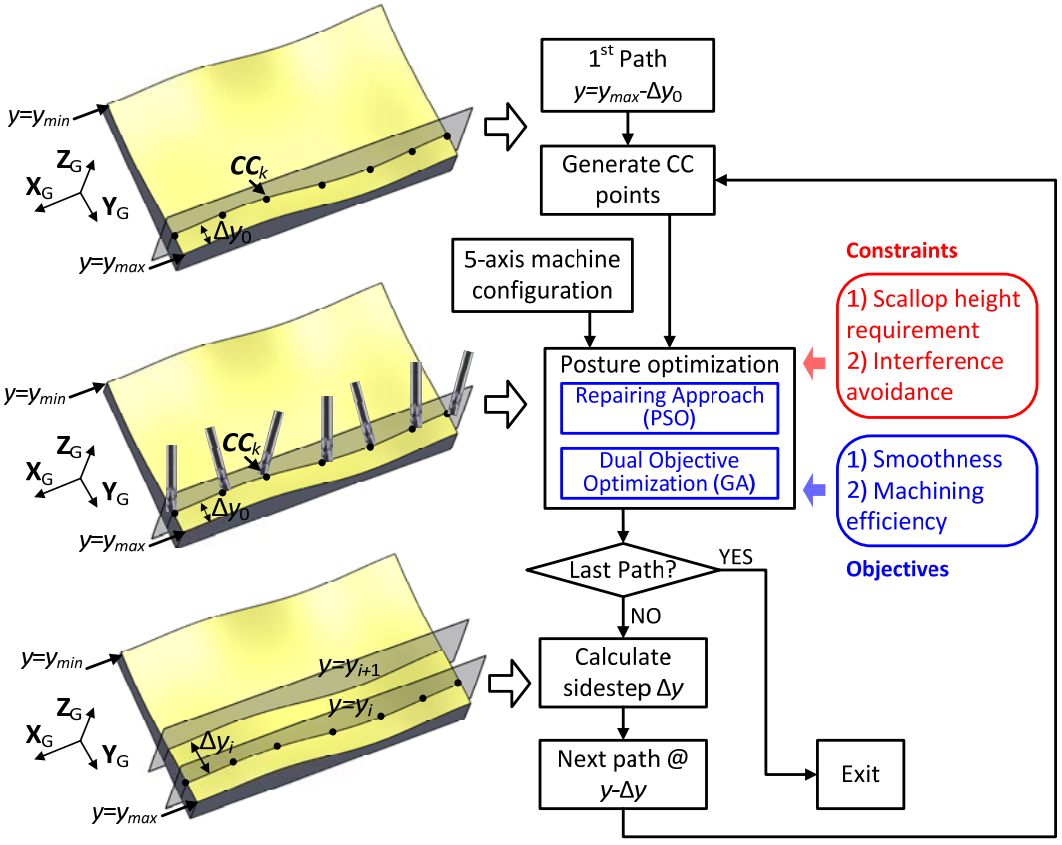


Figure 6.3 Workflow of tool-path generation with posture optimization

1) Set the first cutting plane to be just off the surface edge with a small distance Δy_0. The location of the current cutting plane is $y_i = y_{max} - \Delta y_0$ ($i = 1$). The intersection curve represents the *current path*.

2) Generate the CC points on the current path. During this process, the deviation between the cutter trajectory (determined by interpolator) and the path is kept just below the given surface finish tolerance.
3) Build a posture sequence using either of the optimization methods

   If the **repairing approach** is taken:

   a) Build A-maps at all the CC points. Using the Max-CSW heuristic, assign cutter postures to the CC points to make up the *preliminary* tool-path.
   b) Transform the postures into joint variables. Calculate the joint movement between each pair of cutter postures. Mark all cutter postures with beyond-limit joint movement in-between as unstable. Group consecutive unstable postures into unstable clusters.
   c) For each unstable cluster, randomly pick postures from A-maps of the corresponding CC points to form a random solution. Repeat these process N times to obtain the *initial swarm*. With the custom update rule of the developed PSO algorithm, the swarm will evolve and the best solution of the swarm will be updated. During this process, cutter accessibility and scallop height tolerance act as constraints. When the PSO algorithm terminates, the best solution identified will replace the unstable CL cluster. This process will be repeated for all unstable clusters along a tool-path.

   If the **direct optimization** approach is taken:

   a) Randomly select a cutter posture from the A-map at each CC point to form a posture sequence. Repeat this process to produce N posture sequences to form the initial population (solution pool). The candidate posture sequences then evolve with the GA operators. The reproduction mechanism of the GA ensures that the candidate solutions with better

performance have a better chance of reproduction. When the GA terminates, the candidate posture sequence with the best performance is identified and assigned to the CC points to form the CL path.

4) If path^i is the last path, output the CL data and exit. Else, go to (5).
5) Determine the side-step Δy_i between the current and the next paths. The scallop height tolerance is imposed as a constraint.
6) Set $i = i+1$ and $y_i = y_{i-1}-\Delta y_{i-1}$. If $y_i - y_{\min} \leq \tau$, set $y_i = y_{min}+ \tau$ (τ is a small offset from the surface boundary, e.g., 0.1mm. Go to (2).

6.3.2. Search constraints for cutter posture optimization

The search for the optimal cutter posture sequence is subject to the following two constraints:

1) The cutter posture at each CC point must be interference-free, i.e., the posture should come from the A-map at the CC point.
2) The cutter postures must not produce beyond-the-tolerance scallop height.

While the first constraint is straight-forward, the second constraint has to be interpreted as a constraint on B-CSW at the CC points. For path^1, Δy_0 makes the low bound for B-CSW at any CC point, given as:

$$LB\left(w_b^1\right) = \Delta y_0 \tag{6.7}$$

Any posture failing to produce w_b larger than $LB\left(w_b^1\right)$ will produce scallop height beyond tolerance h, as shown in Fig. 6.4.

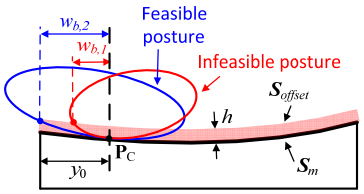


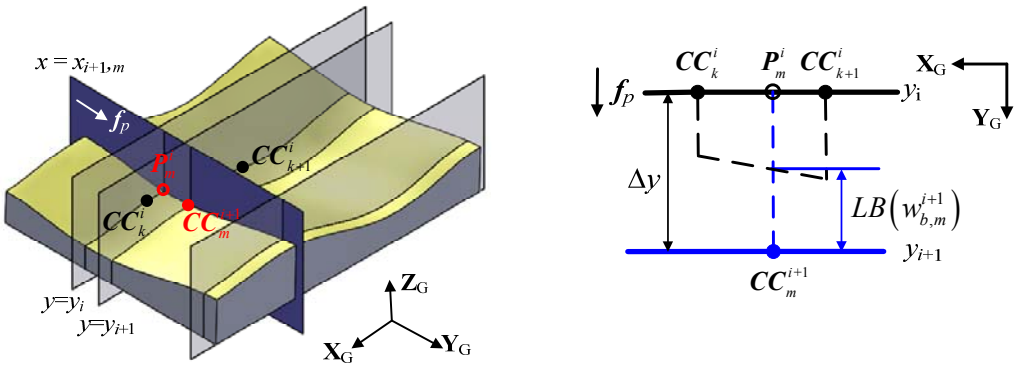
Figure 6.4 Constraint for w_b for postures on the 1st tool-path

For tool-paths other than path[1], the low-bound on w_b is calculated in a different manner. Let a new path be denoted by path^{i+1} and an arbitrary CC point on it be given as $\boldsymbol{CC}_m^{i+1}\left(x_{i+1,m}, y_{i+1}\right)$. The plane along the feeding direction $x = x_{i+1,m}$ will intersect with path^i at $\mathbf{P}_m^i\left(x_{i+1,m}, y_i\right)$, as shown in Fig. 6.5a. The F-CSW at $\mathbf{P}_m^i$ on path^i, $w_f\left(\mathbf{P}_m^i\right)$, together with the path step Δy_i sets a low-bound for $w_{b,m}^{i+1}$ at $\boldsymbol{CC}_m^{i+1}$, given as:

$$LB\left(w_{b,m}^{i+1}\right) = \Delta y_i - w_f\left(\mathbf{P}_m^i\right) \tag{6.8}$$

The B-CSW at $\boldsymbol{CC}_m^{i+1}$ should thus satisfy $w_{b,m}^{i+1} > LB\left(w_{b,m}^{i+1}\right)$. If $\mathbf{P}_m^i$ is coincident with any CC point on path^i, $w_b\left(\mathbf{P}_m^i\right)$ would be readily available. In case that $\mathbf{P}_m^{i+1}$ falls between two CC points, $\boldsymbol{CC}_k^i\left(x_k^i, y_k^i\right)$ and $\boldsymbol{CC}_{k+1}^i\left(x_{k+1}^i, y_{k+1}^i\right)$ as shown in Fig. 6.5a, $w\left(\mathbf{P}_m^i\right)$ is estimated through linear interpolation and Eq. (6.8) is re-written as:

$$LB\left(w_{b,m}^{i+1}\right) = \Delta y_i - w_f\left(\mathbf{P}_m^i\right) = \Delta y_i - \left[w_{f,k+1}^i + \frac{x_{i+1,m} - x_{i,k}}{x_{i,k+1} - x_{i,k}}\left(w_{f,k+1}^i - w_{f,k}^i\right) \right] \tag{6.9}$$



(a) (b)

Figure 6.5 The constraint on w_b at a CC point on path^{i+1}

With Eqs. (6.7) ~ (6.9), for any CC point during tool-path generation, $LB(w_b)$ can be calculated. However, with the current algorithm for path-interval determination, the feasible search space determined by $LB(w_b)$ would be quite limited, resulting in solutions with poor quality. As introduced in Section 4.4.1, the path interval is

determined based on the minimum F-CSW and B-CSW on neighboring paths denoted as path^i and path^{i+1}. The condition is reproduced below for easy reference as:

$$(1-\eta)\left(w^i_{f,\min} + w^{i+1}_{b,\min}\right) < \Delta y_i < \left(w^i_{f,\min} + w^{i+1}_{b,\min}\right) \tag{6.10}$$

Where η is a small number such as 0.05. When the condition given in Eq. (6.10) is met, the path interval Δy_i would be near its maxima. However, it should be noted that when determining the path interval, the cutter postures at the tentative location of path^{i+1} is assigned by the Max-CSW heuristic. As a result, $w^{i+1}_{b,\min}$ is at or near its maximum. By satisfying the condition in Eq. (6.10), the search space for the postures of path^{i+1} would be greatly limited. This is because the B-CSWs at all the CC points of path^{i+1} must be large enough to fill the gap between the cutting strips of path^i and path^{i+1}. To resolve this problem, Eq. (6.10) is modified into the following:

$$\delta(1-\eta)\left(w^i_{f,\min} + w^{i+1}_{b,\min}\right) < \Delta y_i < \delta\left(w^i_{f,\min} + w^{i+1}_{b,\min}\right) \tag{6.11}$$

Where δ is the number in the range of (0.5~1), introduced for the following reasons:

1) A relative large η will still result in tool-paths with relatively good efficiency.
2) The search space for cutter postures are enlarged, allowing for good chance of achieving better tool-path smoothness, due to the fact that scallop-height tolerance will not be easily violated.

In this study, $\delta = 0.7$ is employed.

In summary, the procedure of tool-path generation with optimization means remain basically unchanged except for a more relaxed path interval determination algorithm. Meanwhile, during tool-path optimization, at any path location, a single posture $\boldsymbol{p}_i$ is subject to the following two constraints:

1) $\boldsymbol{p}_i$ is interference-free.
2) At every CC point, $w_{b,i} \geq LB(w_{b,i})$

In the following sections, the two optimization approaches for posture determination will be introduced in details.

6.4. Generation of Smooth Tool-path with Particle Swarm Optimization

In this section, a PSO optimization method designed to improve the smoothness of preliminary tool-paths is introduced. With the proposed method, CLs from the preliminary tool-path are converted into joint variables and checked for beyond-limit joint movements. Based on the checking result, the CLs in the preliminary tool-path are divided in to *stable* and *unstable* clusters. The unstable CL clusters will go through a smoothing process, where a posture sequence with smooth joint movements will be obtained to replace the unstable cluster. The search algorithms are based on particle swarm optimization algorithms (PSO). Customized search and update rules are developed to enhance the classic PSO for better performance.

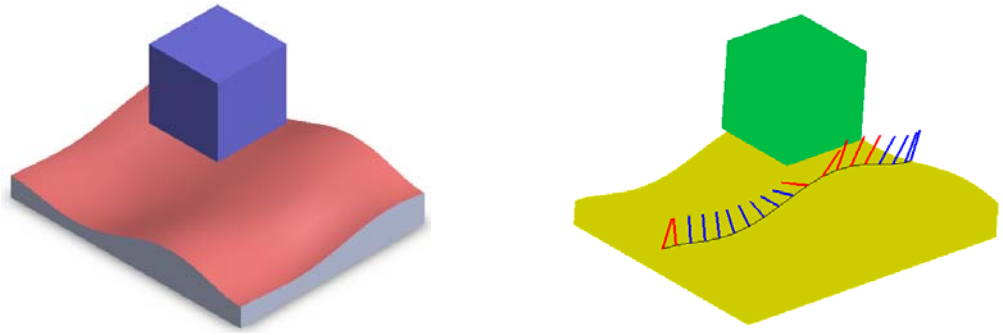
6.4.1. Identification of unstable CL clusters

For the CLs in a given preliminary tool-path, the postures are transformed into joint variables using the method introduced in Section 6.2. Suppose there are N CLs on the tool-path. The criteria for identifying unstable CLs are as follows:

1) For neighboring postures $\boldsymbol{p}_k$ and $\boldsymbol{p}_{k+1}$ (k = 1, 2, …, N-1), if the corresponding joint travel $J_{k,k+1}$ exceeds the given limit J_{lim}, both $\boldsymbol{p}_k$ and $\boldsymbol{p}_{k+1}$ are marked as *unstable*.
2) Let $\boldsymbol{p}_s$ and $\boldsymbol{p}_{s+n}$ are two stable postures with only unstable postures in-between, if $J_{s,s+n} > (n\text{-}1)J_{lim}$, $\boldsymbol{p}_{s+n}$ will be marked as *unstable*.

The second criterion is to ensure that the average joint travel among a consecutive set of unstable CLs must not exceed J_{lim}, thus making the smoothing target reachable. After all the unstable CLs are identified, the consecutive unstable CLs are grouped

into an unstable CL cluster. An example is shown in Fig. 6.6a, where a cube is placed above the workpiece to induce drastic joint movements. A single preliminary tool-path is generated right below the obstacle, for which two unstable CL clusters are identified and plotted in red (see Fig. 6.6b).



(a) (b)

Figure 6.6 Unstable CL clusters in a pre-liminary tool-path

6.4.2. Problem Formulation and initialization

PSO is a population-based search method motivated by the simulation of the social behavior of bird flocking. A PSO algorithm maintains a swarm of particles, each representing a feasible solution. During the search process, the location of the particle is updated based on both the social knowledge (location of the best particle in the swarm) and the particle's own experience (best solution discovered by the particle itself). It is expected that after a certain number of iterations, the particles will converge to a single location supposed to be the optimal solution. For more background information the method, readers can refer to the work of EngelBrecht (2005).

For the CL *repairing* problem, the location of a particle represents a possible posture sequence for an unstable CL cluster. The particle's location is updated with the discrete time step t throughout the search. At t, if the swarm size is M, a particle

position is represented by $\boldsymbol{X}_i(t) = \{\boldsymbol{p}_{i,m}, \boldsymbol{p}_{i,m+1}...\boldsymbol{p}_{i,m+l}\}$, i = 1, 2, …, M, where m is the index of the first unstable CL and l is the length of the unstable CL cluster. The search in PSO starts with the initial swarm (t=0), which consists of only randomly generated initial solutions. To make sure the initial swarm contains only feasible solutions, a trial-and-error approach with the following steps is used for initialization of the swarm:

1) Set k=m (index of 1st CC point in the unstable cluster) and i=1.
2) Calculate $LB(w_{b,k})$ based on the current path location and path interval.
3) Randomly pick a posture $\boldsymbol{p}$ from the A-map at CC point $\mathbf{P}_k$. Calculate w_b.
4) IF $w_b > LB(w_{b,k})$, add $\boldsymbol{p}$ to solution $\boldsymbol{X}_i$ as $\boldsymbol{p}_{i,k}$. Set $k = k+1$. ELSE, repeat step 3).
5) IF $k=m+l$ (index of last CC point in the unstable cluster), set $k=m$ and $i=i+1$.ELSE, go to step 2).
6) IF $i=M$, stop. ELSE, go to step 2).

6.4.3. Customized update rule for PSO

Starting from the initial swarm at t=0, the swarm is continuously updated using our customized update rule till a satisfactory solution is obtained. For a single particle i, the best solution obtained up to time t is recorded as the particle's *personal best*, i.e., $\boldsymbol{PB}_i(t) = Best\{\boldsymbol{X}_i(0), \boldsymbol{X}_i(1), \ldots, \boldsymbol{X}_i(t)\}$. Out of all the personal best in the swarm at time t, the best solution is recorded as the *global best*, i.e., $\boldsymbol{GB}(t) = Best\{\boldsymbol{PB}_1(t), \boldsymbol{PB}_2(t), \ldots, \boldsymbol{PB}_M(t)\}$. At the subsequent time instant t+1, $\boldsymbol{X}_i$ will be updated using our customized updating rule. The location of a particle is updated based on the *last update direction*, the *personal best location*, $\boldsymbol{PB}_i(t) = \{\boldsymbol{pb}_{i,m}, \boldsymbol{pb}_{i,m+1}\ldots \boldsymbol{pb}_{i,m+l}\}$, and the *global best location* $\boldsymbol{GB}(t) = \{\boldsymbol{gb}_m, \boldsymbol{gb}_{m+1}\ldots \boldsymbol{gb}_{m+l}\}$. Following this principle, for a posture $\boldsymbol{p}_{i,j}(t)$ taken from $\boldsymbol{X}_i$, it will be updated 3 times based on $\boldsymbol{p}_{i,j}(t\text{-}1)$, $\boldsymbol{pb}_{i,j}$, and $\boldsymbol{gb}_j$ consecutively, one at a time. As a general rule, when a posture $\boldsymbol{p}$ is being updated based on a target

posture $\boldsymbol{p}_{target}$, the resultant posture $\boldsymbol{p}_{new}$ is on the plane determined by $\boldsymbol{p}_{target}$ and $\boldsymbol{p}$, given as:

$$\boldsymbol{p}_{new} = f\left(\boldsymbol{p}, \boldsymbol{p}_{target}, c\right) = \left(\boldsymbol{p} \times \boldsymbol{p}_{target} \times \boldsymbol{p}\right)\sin\left(c\beta\right) + \boldsymbol{p}\cos\left(c\beta\right) \tag{6.12}$$

where β is the angle between $\boldsymbol{p}$ and $\boldsymbol{p}_{target}$ and c is a parameter controlling the step size of updating. The principle of update is illustrated in the Fig. 6.7.

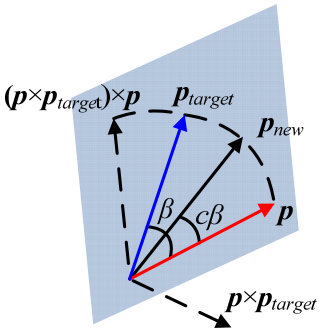


Figure 6.7 General update rule regarding current posture $\boldsymbol{p}$ and target posture $\boldsymbol{p}_{target}$

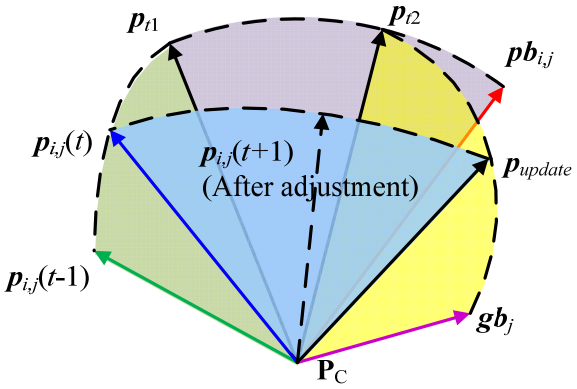


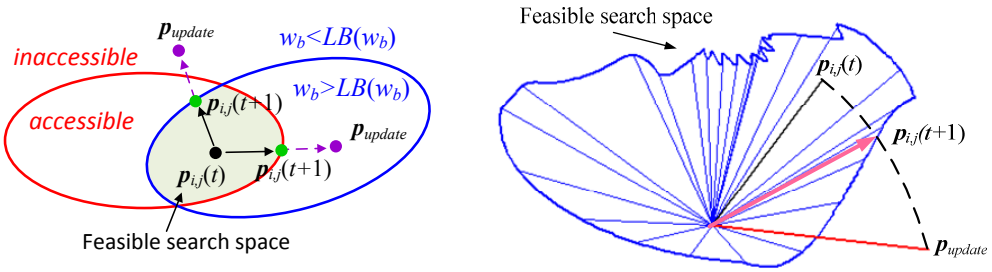
Figure 6.8 3-step update rule cutter posture $\boldsymbol{p}_{i,j}(t)$

At time step t, the above update rule will be applied three times for a cutter posture $\boldsymbol{p}_{i,j}(t)$ from particle $\mathbf{X}_i$. $\boldsymbol{p}_{i,j}(t\text{-}1)$ is used as $\boldsymbol{p}_{target}$ in the first update while $\boldsymbol{pb}_{i,j}$ and $\boldsymbol{gb}_j$ are used as the $\boldsymbol{p}_{target}$ for the following 2 updates. The 3-step updating process is given as:

$$\begin{aligned}
&(i)\ \boldsymbol{p}_{t1} = f\left(\boldsymbol{p}_{i,j}\left(t\right), \boldsymbol{v}_{i,j}\left(t-1\right), c_0\right)\\
&(ii)\ \boldsymbol{p}_{t2} = f\left(\boldsymbol{p}_{t1}, \boldsymbol{pb}_{i,j}, r_1 c_1\right)\\
&(iii)\ \boldsymbol{p}_{update} = f\left(\boldsymbol{p}_{t,2}, \boldsymbol{gb}_j, r_2 c_2\right)
\end{aligned} \tag{6.13}$$

where $\boldsymbol{p}_{t1}$ and $\boldsymbol{p}_{t2}$ are intermediate products of the update, c_0, c_1 and c_2 are step sizes assigned to the three updates and r_1 and r_2 are random numbers in the range of (0~1) designed to add a stochastic element to the search. Please note that the step size for the first update c_0 is a negative number in the range of (-2, -1), so that the search will follow the last update direction instead its opposite. This 3-step updating process is shown in Fig.6.8.

After the 3-step update process, the obtained cutter posture is referred to as $\boldsymbol{p}_{update}$. If $\boldsymbol{p}_{update}$ is tested feasible against the two constraints, (interference avoidance and low bound on w_b), $\boldsymbol{p}_{update}$ will be recorded as $\boldsymbol{p}_{i,j}(t+1)$. Update will then go on for the other cutter postures of the particle. However, as no measures have been taken to guarantee the feasibility of $\boldsymbol{p}_{update}$, it is very likely that $\boldsymbol{p}_{update}$ will breach one or both of the search constraints. In such cases, $\boldsymbol{p}_{update}$ should be adjusted to become feasible.



(a) (b)

Figure 6.9 Adjusting update step to guarantee feasibility of solution (a) 2D (b) 3D

Assuming both cutter accessibility and CSW changes continuously with cutter posture, two posture ranges exist around $\boldsymbol{p}_{i,j}(t)$, i.e., the accessible posture range and the posture range that satisfies the limit imposed by $LB(w_b)$. The intersection of these two posture ranges makes up the feasible search space for $\boldsymbol{p}_{i,j}(t+1)$. With such a feasible search space around $\boldsymbol{p}_{i,j}(t)$**,** when $\boldsymbol{p}_{i,j}(t+1)$ is close enough to $\boldsymbol{p}_{ij}(t)$, a feasible posture would always exist (see Fig. 6.9a). Thus, in case of violated constraints, $\boldsymbol{p}_{i,j}(t+1)$ can be found on the plane determined by $\boldsymbol{p}_{update}$ and $\boldsymbol{p}_{i,j}(t)$ by reducing the

angle between $\boldsymbol{p}_{i,j}(t+1)$ and $\boldsymbol{p}_{i,j}(t)$ (see Fig. 6.9b). On the other hand, for better exploration of the search space, the update step should be as large as possible, i.e., $\boldsymbol{p}_i(t+1)$ should be as close to $\boldsymbol{p}_{update}$ as possible. For this purpose, when $\boldsymbol{p}_{update}$ violates any of the search constraint, $\boldsymbol{p}_{i,j}(t+1)$ will be continuously adjusted using the bi-section approach till the change is sufficiently small, which is an indication that $\boldsymbol{p}_{i,j}(t+1)$ is at the boundary of the feasible search space around it (see Fig. 6.9a). The update process for the complete swarm is given as follows:

For $i = 1{:}M$ (swarm size)

For $j=m{:}m+l$ (index of the CC points from an unstable cluster)

1) Take $\boldsymbol{p}_{i,j}(t)$ from the particle location $\boldsymbol{X}_i(t)$ and obtain $\boldsymbol{p}_{update}$ through the update process given in Eq. (6.13).
2) Test $\boldsymbol{p}_{update}$ against the constraints given in 6.4.2.

If $\boldsymbol{p}_{update}$ is feasible, set $\boldsymbol{p}_{i,j}(t+1) = \boldsymbol{p}_{update}$. **Else**, set c=0.5. Go to step 3).

3) Conduct the update $\boldsymbol{p}_{i,j}(t+1) = f(\boldsymbol{p}_{i,j}(t), \boldsymbol{p}_{update}, c)$.
4) **If** $\boldsymbol{p}_{i,j}(t+1)$ is feasible, calculate the angle between $\boldsymbol{p}_{update}$ and $\boldsymbol{p}_{i,j}(t+1)$, given as $\beta=\cos^{-1}(\boldsymbol{p}_{update}\cdot\boldsymbol{p}_{i,j}(t+1))$.

If β <0.1°, accept $\boldsymbol{p}_{i,j}(t+1)$.

Else set $\boldsymbol{p}_{i,j}(t) = \boldsymbol{p}_{i,j}(t+1)$. Go to step 3)

Endif

Else set $\boldsymbol{p}_{update} = \boldsymbol{p}_{i,j}(t+1)$. Go to step 3)

Endif

End

End

6.4.4. Hybrid PSO with mutation operator

Classic PSO is notorious for its tendency of pre-mature convergence, which is why a

hybrid PSO with a mutation operator is proposed here for our algorithm. Regarding a cutter posture, the mutation operator will generate a child posture randomly within a certain range (in terms of angular difference) around the original posture. There are 2 levels of mutations, i.e., particle-level and swarm-level. Particle-level mutation happens throughout the search at probability $p_{mutation}$. It re-generates certain postures of a certain particle around their current locations. The introduction of the operator is pro-exploration, i.e., to provide more diversity for the search. The particle-level mutation operator will work along with the customized update operator to produce the postures within a particle. The two constraints governing the update process also apply for the particle-level mutation operator.

On the other hand, swarm-level mutation only happens when the particles have converged around the global best location. It is basically a re-initialization of the swarm around the obtained ***GB*** to provide better exploitation of the search space around the current ***GB*.**

The maximum angular difference between the parent and child postures, denoted by S, is referred to as the mutation step. As the beginning of the search, exploration is favored so that our algorithm can escape local optima easily. As a result, large mutation step is desired. On the contrary, towards the end of the search, we want the search to focus on the vicinity of ***GB*** to produce improvement. For this reason, exploitation is favored and smaller mutation steps are desired. To balance exploration and exploitation, S should shrink with the time step t and yet remain above a certain level towards the end of the search. Based on experiments, the following rule is proposed (S_{max} and S_{min} are user defined maximum and minimum mutation step sizes, such as 90° and 5° respectively):

$$S(t) = e^{-t/t_{\max}} \left(\frac{F(\boldsymbol{GB}(t))}{F(\boldsymbol{GB}(0))} (S_{\max} - S_{\min}) + S_{\min} \right) \tag{6.14}$$

6.4.5. Cost function, replacement mechanism and stopping criterion

Apart from the update rules, other important aspects of the PSO algorithm are given as follows:

(1) **Cost Function:** The optimization objective for the PSO is to minimize the maximum joint travel along an unstable CL cluster. For a candidate solution $\boldsymbol{X}_i$, the cost function is given as: $F(\boldsymbol{X}_i) = max\{J_{k,k+1} | \ k=m\text{-}1,\ldots, m+l\}$. It is worth noting that the joint travel between the unstable cluster and the stable postures at both ends of it has also been taken into account in this cost function.

(2) **Replacement scheme:** After a new particle location $\boldsymbol{X}_i(t+1)$ is obtained, its cost function $F(\boldsymbol{X}_i(t+1))$ will be compared with that of $F(\boldsymbol{PB}_i(t))$ and $F(\boldsymbol{GB}(t))$ respectively to decide if replacement is needed:

   (a) $F(\boldsymbol{X}_i(t+1)) > F(\boldsymbol{PB}_i(t))$, no replacement happens.

   (b) $F(\boldsymbol{X}_i(t+1)) < F(\boldsymbol{PB}_i(t))$ and $F(\boldsymbol{X}_i(t+1)) > F(\boldsymbol{GB}(t))$, $\boldsymbol{PB}_i(t+1)=\boldsymbol{X}_i(t+1)$.

   (c) $F(\boldsymbol{X}_i(t+1)) < F(\boldsymbol{GB}(t))$, $\mathbf{GB}(t+1)=\boldsymbol{X}_i(t+1)$ and $\mathbf{PB}_i(t+1)= \boldsymbol{X}_i(t+1)$.

(3) **Stopping criterion:** The search will stop when ***GB*** remains the same for t_{null} time steps or when the pre-set maximum time step t_{max} is reached.

(4) The parameters used with the implemented PSO are given in Tab. 6.1.

Table 6.1 Parameters used in the hybrid PSO

Swarm Size	c_0	c_1	c_2	*Maximum iteration* t_{max}	*Maximum null iteration* t_{null}	
20	-1.4	0.8	0.5	200	25	
	$p_{mutation}$				S_{max}	S_{min}
	0.2				90º	5º

6.4.6. The overall algorithm for tool-path smoothing with PSO

When the stopping criterion is met, **GB,** which is the best solution identified by PSO will replace the unstable cluster in the posture sequence. It should be noted that there might be multiple unstable clusters inside a posture sequence assigned by the Max-CSW heuristic, each of which should be smoothed. The complete steps for smoothing the postures sequence of a preliminary tool-path is given as follows:

1) Suppose N CC points along the tool-path and postures are $\{\boldsymbol{p}_1, \boldsymbol{p}_2,\ldots, \boldsymbol{p}_N\}$.
2) Scan the posture sequence $\{\boldsymbol{p}_1, \boldsymbol{p}_2,\ldots, \boldsymbol{p}_N\}$ and obtain H unstable clusters, the starting CC point indices and length of which are given as $\{m_1, m_2,\ldots, m_H\}$ and $\{l_1, l_2, \ldots, l_H\}$, respectively. Set k=1.
3) Set t = 0. For the kth unstable cluster, initialize a swarm of M particles using the method introduced in Section 6.5.2, given as $\{\boldsymbol{X}_1(t), \boldsymbol{X}_2(t),\ldots, \boldsymbol{X}_M(t)\}$. Calculate $\boldsymbol{GB}(t)$ of the swarm and $\{\boldsymbol{PB}_1(t), \boldsymbol{PB}_2(t),\ldots, \boldsymbol{PB}_M(t)\}$.
4) **do**{
    a. **For** i=1: M
        For $j=m_k : m_k+l_k$
            Generation random number *rand*. If $rand<p_{mutation}$ (probability for mutation), generate $\boldsymbol{p}_{i,j}$ (t+1) using the particle-level mutation operator. Else, produce $\boldsymbol{p}_{i,j}$ (t+1) using the custom update rule introduced in Section 6.4.3.
        End
    End
    b. Set $t = t$+1. Update $\boldsymbol{GB}(t)$ and $\{\boldsymbol{PB}_1(t), \boldsymbol{PB}_2(t),\ldots, \boldsymbol{PB}_M(t)\}$.

c. Test if solutions have converged around ***GB***(t). If so, conduct swarm level mutation to re-initialize the search around ***GB***(t) within the mutation step.

} **while** (stopping criterion is not met)

5) Replace the unstable cluster with ***GB***(t). Set k=k+1. If k=H+1, stop and output the smooth posture sequence. Otherwise, go to step (3).

For the single tool-path shown in Fig.6.5b, the PSO smoothing algorithm is applied. The smoothed tool postures are shown in Fig. 6.10. It can be seen that the smoothness of the unstable clusters has greatly improved. For the unstable cluster containing 5 postures, Fig. 6.11 shows the search process (cost vs. iterations) for both the classic PSO and the developed hybrid PSO. It is obvious that the hybrid PSO has the edge as the classic PSO was trapped in local minima early in the search. This clearly demonstrates the importance of the proposed mutation operator.

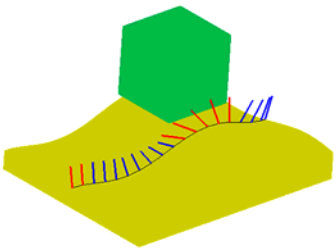


Figure 6.10 The tool-path after smoothing

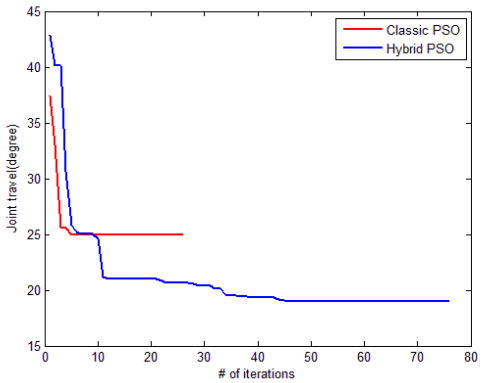


Figure 6.11 Comparison of performance between classic PSO and hybrid PSO

6.5. Cutter Posture Optimization using Genetic Algorithms

In the PSO algorithm, the preliminary tool-path with its postures assigned by the MAX-CSW heuristic is used as the start point. The weakness of such an approach lies with (1) no optimization efforts are made to improve the machining efficiency of the tool-paths and (2) The room for improvement is limited regarding tool-path smoothness, as optimization is conducted within unstable CL clusters only. Considering these shortcomings, a multi-objective optimization method based on genetic algorithm (GA) is proposed in this chapter. Optimization is conducted regarding the whole tool-path aiming at improving machining efficiency and tool-path smoothness simultaneously.

6.5.1. Optimization objectives

The optimization algorithm proposed in this section considers two measures of a tool-path: tool-path smoothness and machining efficiency. Specifically, the following objectives are considered regarding the two measures:

1) The maximum joint movement J_{max} along the tool-path should be minimized or should be kept small.
2) The minimum F-CSW, $w_{f,min}$, along the tool-path should be maximized.

In the proposed method, machining efficiency is measured by the minimum F-CSW on the current tool-path, denoted as $w_{f,min}$. This is due to the fact that a larger $w_{f,min}$ will lead to larger path interval between the current and the next one. Two neighboring tool-paths at y_i and y_{i+1} are shown in Fig. 6.12. When the CSW at y_{i+1} remains unchanged and the minimum F-CSW at y_i increases from $w_{f,min}$ to $w'_{f,min}$, the resultant scallop height h' will be smaller than the scallop height tolerance h. To achieve the scallop height close to h, the path interval Δy_i will need to be increased

(from Δy_i to $\Delta y_i'$). Therefore, it can be concluded that larger $w_{f,min}$ on the current tool-path at y_i will lead to better machining efficiency.

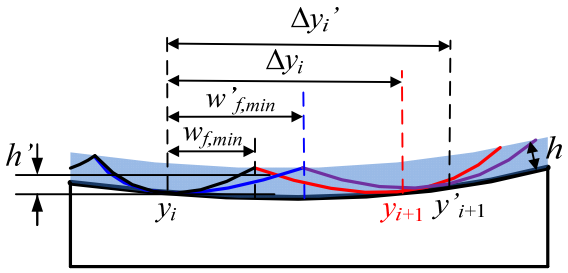


Figure 6.12 Larger $w_{f,min}$ leading to larger path interval

6.5.2. Problem formulation, initialization and fitness function

Genetic algorithm (GA) is a search heuristic that simulates the evolving process of species in nature. In a GA, a solution pool called a *population* is maintained at every generation. Each individual in the population, called a *chromosome*, is a candidate solution encoded as a string. At first, the chromosomes are randomly initialized from the solution space and their fitness is evaluated. Various *GA operators* (crossover, mutation, etc) will then be applied to the existing population to produce new chromosomes for the next generation. These operators are designed following the rule of natural selection, so that chromosomes of the stronger individuals (i.e., solution with better fitness) are more likely to be preserved and passed onto the next generation. In the meantime, new chromosomes (new random solutions) keep being introduced into the population. This evaluation-reproduction cycle will be carried out iteratively till convergence or a satisfactory solution is achieved. GA is considered a powerful search tool with many applications. To get its full potential, the algorithm has to be tailored to the nature of our problem.

- **Chromosome structure**

For posture assignment optimization, a candidate solution (chromosome) is made up of the complete posture sequence at all the CC points along a tool-path. Suppose there are N CC points on the tool-path (the kth), the chromosome is given as $\boldsymbol{g}_k = \left\{ \boldsymbol{p}_1^k, \boldsymbol{p}_2^k, ..., \boldsymbol{p}_N^k \right\}$. The cutter posture at the jth CC point on the tool-path, denoted as $\boldsymbol{p}_j^k$, would be referred to as a *gene* of the $\boldsymbol{g}_k$. As mentioned earlier, a population of chromosomes will be maintained during the search. Suppose there are a total of M chromosomes inside the population, the population is represented as: $\boldsymbol{P} = (\boldsymbol{g}_1, \boldsymbol{g}_2, ..., \boldsymbol{g}_M)$.

- **Initialization**

Like the PSO method, the initial solution pool with GA should also contain only feasible solutions. The trail-and-error approach is used here with the following steps:

1) Set i=1 (index of CC point) and k=1 (index of random solution).
2) Randomly pick a posture $\boldsymbol{p}$ from the A-map at CC point i. Calculate w_b.
3) IF $w_b > LB(w_b)$, append $\boldsymbol{p}$ to solution $\boldsymbol{g}_k$ as $\boldsymbol{p}_i^k$. Set $i = i+1$. ELSE, repeat step 2).
4) IF $i=N+1$, set $k=k+1$ and i=1.ELSE, go to step 2)
5) IF $k=M$, exit. ELSE, go to step 2).

- **Fitness Function**

A simple approach for evaluating the fitness of a solution is to assign different weights to J_{max} and $w_{f,min}$, and sum them up in a single fitness function. However, such an approach does not make sense in terms of physical meanings. Another shortcoming is that the weights of both measures would have to be determined by experiment and would be case-sensitive. Thus, a new approach is proposed to form the fitness function.

In the GA algorithm, the evaluation of solution fitness serves two purposes: (1) to rank the solutions of a new generation, so that their probability for reproduction can be determined and (2) to identify the best solution $\boldsymbol{g}_{g\_best}$ of the generation. These two tasks do not require a uniform fitness function throughout the whole search but one that works at generation level. For this purpose, we propose to normalize all the $J_{\max}^{-1}$ and $w_{f,min}$ to the range of [0~1], denoted as $n(J_{\max}^{-1})_i$ and $n(w_{f,min})_i$, $i = 1,\ldots,M$. For solution $\boldsymbol{g}_k$ of the generation, the normalized optimization measures are given as,

$$
\begin{aligned}
n\left(J_{\max}^{-1}\right)_k &= \frac{J_{\max,k}^{-1} - \min\left\{J_{\max,i}^{-1}\Big|_{i=1,\ldots,M}\right\}}{\max\left\{J_{\max,i}^{-1}\Big|_{i=1,\ldots,M}\right\} - \min\left\{J_{\max,i}^{-1}\Big|_{i=1,\ldots,M}\right\}} \\
n\left(w_{f,\min}\right)_k &= \frac{w_{f,\min}^{k} - \min\left\{w_{f,\min}^{i}\Big|_{i=1,\ldots,M}\right\}}{\max\left\{w_{f,\min}^{i}\Big|_{i=1,\ldots,M}\right\} - \min\left\{w_{f,\min}^{i}\Big|_{i=1,\ldots,M}\right\}}
\end{aligned}
\tag{6.15}
$$

The fitness of $\boldsymbol{g}_k$ within the generation is hence given as:

$$
f\left(\boldsymbol{g}_k\right) = c_1 n\left(JT_{\max}^{-1}\right)_k + c_2 n\left(w_{f,\min}\right)_k,\ c_1 + c_2 = 1 \tag{6.16}
$$

c_1 and c_2 are the weights assigned to the two measures, indicating the importance placed on each optimization objective by the users (c_1+c_2=1).

6.5.3. Reproduction: immigration, elitism and cross-over

The chromosomes obtained by initialization make up the initial population. The population will keep being updated during the search with the help of reproduction operators. The operators used in this paper include *elitism*, *immigration*, and *cross-over*. The idea of elitism is the simplest, i.e., the best chromosome(s) of the current generation $\boldsymbol{P}_{current}$ is preserved and passed onto the next generation $\boldsymbol{P}_{next}$. The immigration operator, on the other hand, is aimed at increasing the diversity of the population. It simply generates a new chromosome randomly for $\boldsymbol{P}_{next}$.

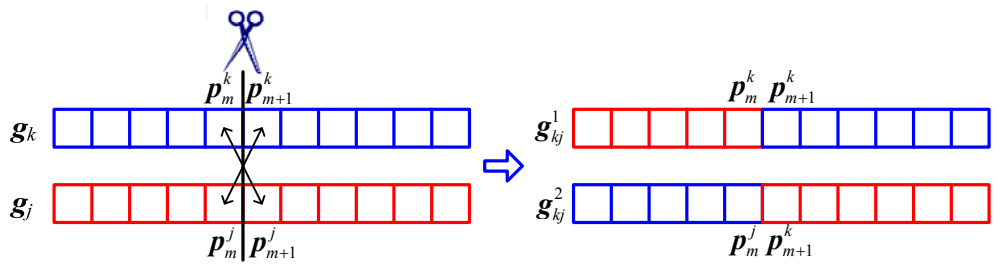


Figure 6.13 Mechanism of the cross-over operator in the proposed GA

Besides the chromosomes obtained via elitism and immigration, the rest of the chromosomes in $\boldsymbol{P}_{next}$ are produced by cross-over. The idea behind cross-over is that a child chromosome can have better fitness if it takes the good characteristics from both of its parents. To produce improvement in the next generation, we must make sure the 'good' chromosomes have a better chance of reproduction. In our algorithm, the simple *roulette selection* is used based on the nominal fitness of the chromosomes within a generation. Suppose there are a total of M individuals in the population, the probability for a chromosome $\boldsymbol{g}_k$ to be selected for cross-over is:

$$P(\boldsymbol{g}_k) = \frac{fn(\boldsymbol{g}_k)}{\sum_{i=1}^{M} fn(\boldsymbol{g}_i)} \tag{6.17}$$

A cross-over operator takes two parent chromosomes. During cross-over, both parent chromosomes will be severed at a randomly selected break point and each chromosome is combined with the corresponding segment from its spouse (see Fig. 6.13). In this way, two child genes will be created. For example, for two parent genes $\boldsymbol{g}_k = \{\boldsymbol{p}_1^k, \boldsymbol{p}_2^k, ..., \boldsymbol{p}_N^k\}$ and $\boldsymbol{g}_j = \{\boldsymbol{p}_1^j, \boldsymbol{p}_2^j, ..., \boldsymbol{p}_N^j\}$, let the break point be at m, the 2 child genes obtained are $\boldsymbol{g}_{kj}^1 = \{\boldsymbol{p}_1^k, \boldsymbol{p}_2^k, ..., \boldsymbol{p}_{m-1}^k, \boldsymbol{p}_m^j, ..., \boldsymbol{p}_N^j\}$ and $\boldsymbol{g}_{kj}^2 = \{\boldsymbol{p}_1^j, \boldsymbol{p}_2^j, ..., \boldsymbol{p}_{m-1}^j, \boldsymbol{p}_m^k, ..., \boldsymbol{p}_N^k\}$. The nominal fitness of the two offspring will be evaluated using the normalization setting of $\boldsymbol{P}_{current}$. Between $\boldsymbol{g}_{kj}^1$ and $\boldsymbol{g}_{kj}^2$, the one with better fitness will be selected and placed into $\boldsymbol{P}_{next}$.

Except elitism, which is always called to pass on the best solution of the current population, the activation of immigration and cross-over operators are probabilistic events. Suppose the probability for the immigration operator is given as $p_{immigration}$, a random number *rand* will be generated in the range of [0~1] first. When $rand<p_{immigration}$, the immigration operator will be called to generate a new chromosome for the new population. Otherwise, the cross-over operator will be called. Based on this rule, reproduction of the population can be carried out iteratively in the following steps:

1) Set $\boldsymbol{P}_{next}$ = NULL. Rank the random solutions of $\boldsymbol{P}_{current}$ based on nominal fitness. Identify the best solution of the population $\boldsymbol{g}_{l\_best}$. Put $\boldsymbol{g}_{l\_best}$ of $\boldsymbol{P}_{current}$ into $\boldsymbol{P}_{next}$. Set k= 1.
2) Generate random number *rand* in the range of (0~1).

   If $rand < p_{immigration}$

   Apply the immigration operator to generate a new solution for $\boldsymbol{P}_{next}$.

   Else

   Based on Eq. (6.17), randomly select two solutions from $\boldsymbol{P}_{current}$ using roulette selection and conduct cross-over. Out of the two offspring, select the one with better fitness and put it into $\boldsymbol{P}_{next}$.

   Endif

   Set $k=k+1$.
3) IF $k=n+1$, $\boldsymbol{P}_{next}$ is fully filled. Exit. Else, go to (2).

6.5.4. Knowledge-based mutation

Besides the reproduction operators introduced in Section 6.5.3, *mutation* is also an important operator for classic GA as a source of solution diversity. The mutation operator works at gene level and will alter a gene randomly with a certain probability.

For our case, this means the change of one or several postures (with a certain probability) in a selected posture sequence (see Fig. 6.14). Due to the high complexity of our problem, this simple mutation mechanism, randomly selected a posture from the A-map to replace the current one, has limited effect. Hence, a more advanced mutation strategy with built-in knowledge is proposed.

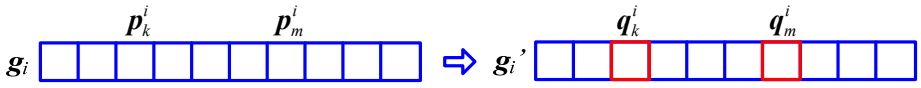
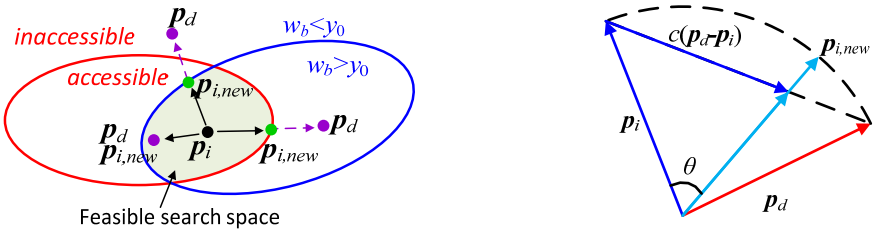


Figure 6.14 Mechanism of mutation operator in the proposed GA

The proposed mutation strategy is based on the assumption that both cutter accessibility and CSW changes continuously with cutter posture. Accordingly, there should be two regions around a feasible posture: (1) the accessible posture range, (2) the posture range that satisfies the limit imposed on the B-CSW (see Fig.6.15a). The intersection of these two regions forms the feasible search space (for mutation) around the current posture.



(a) Search for $p_{i,new}$ constrained by feasible search space (b) Search for $p_{i,new}$ based on p_d

Figure 6.15. Illustration of mutation with built-in knowledge

When conducting mutation to a posture p_i, it is preferable to replace it with a *desirable posture*, p_d, that produces better fitness than p_i (assuming p_d is known). In case that p_d does not fall into the feasible range around p_i, a strategy is proposed to "push" p_i towards p_d until the new p_i reaches the boundary of the feasible search space

(see Fig.6. 15a). Accordingly, an iterative search method is proposed to find the new $\boldsymbol{p}_i$ using the following rule,

$$\boldsymbol{p}_{i,new} = \frac{\boldsymbol{p}_i + c(\boldsymbol{p}_d - \boldsymbol{p}_i)}{\left|\boldsymbol{p}_i + c(\boldsymbol{p}_d - \boldsymbol{p}_i)\right|} \tag{6.18}$$

where $c \in (0\sim1)$ (see Fig. 6.15b). If $\boldsymbol{p}_d$ turns out to be infeasible, search for $\boldsymbol{p}_{i,new}$ will be conducted during which the two constraints will be checked against. The search process is as follows:

1) Check the feasibility of $\boldsymbol{p}_d$. If $\boldsymbol{p}_d$ is feasible, set $\boldsymbol{p}_{i,new} = \boldsymbol{p}_d$ and exist. Otherwise, set c=0.5.
2) Obtain $\boldsymbol{p}_{i,new}$ based on Eq. 6.18. If $\boldsymbol{p}_{i,new}$ meets both constraints, go to (3). Else, go to (4).
3) Calculate $\theta = \cos^{-1}(\boldsymbol{p}_i \cdot \boldsymbol{p}_{i,new})$. If $\theta < 0.5°$, accept the current $\boldsymbol{p}_{i,new}$ and stop. Else, set $\boldsymbol{p}_i = \boldsymbol{p}_{i,new}$ and c=0.5. Go to (2).
4) Set c=0.5c and go to (2).

It can be seen that in the proposed mutation strategy, the key is to find the *desirable posture* $\boldsymbol{p}_d$ for a given $\boldsymbol{p}_i$. It is observed that the assignment of $\boldsymbol{p}_d$ depends on the characteristic of $\boldsymbol{p}_i$. The categorization of $\boldsymbol{p}_i$ and the corresponding strategy for $\boldsymbol{p}_d$ assignment are described as follows:

1) $\boldsymbol{p}_i$ being a critical posture - producing J_{max}. The probability of mutation for this scenario is 1.0. The two neighboring postures corresponding to J_{max} are denoted as $\boldsymbol{p}_i$ and $\boldsymbol{p}_{i+1}$. The objective of mutation for this scenario is to reduce the joint movement and not to cause damage to machining efficiency. Hence, $\boldsymbol{p}_d = 0.5(\boldsymbol{p}_{i-1}+\boldsymbol{p}_{i+1})$ for $\boldsymbol{p}_i$ or $\boldsymbol{p}_d = 0.5(\boldsymbol{p}_i+\boldsymbol{p}_{i+2})$ for $\boldsymbol{p}_{i+1}$. At the same time, an extra constraint is applied to the search of $\boldsymbol{p}_{i,new}$, i.e., w_f at $\boldsymbol{p}_{i,new}$ must be no less than $w_{f,min}$ of the current posture sequence.

2) $\boldsymbol{p}_i$ being a critical posture - producing $w_{f,\min}$. The probability of mutation for this scenario is also 1.0. The objective of mutation for this scenario is to increase the machining efficiency without causing rise to joint movement. Firstly, within the feasible search space of $\boldsymbol{p}_i$, through trial-and-error, a posture $\boldsymbol{p}_i'$ can be found with its w_f larger than the *second* smallest forward strip-width $w'_{f,\min}$. $\boldsymbol{p}_i'$ is used as the new $\boldsymbol{p}_i$ to start with. Secondly, set $\boldsymbol{p}_d = 0.5(\boldsymbol{p}_{i-1}+\boldsymbol{p}_{i+1})$. As shown in Fig. 6.16, the new posture $\boldsymbol{p}_{i,new}$ is found by mutating $\boldsymbol{p}_i'$ towards $\boldsymbol{p}_d$ using $w'_{f,\min}$ as the constraint on the forward strip-width. This is because any w_f larger than $w'_{f,\min}$ will no longer be able to improve the minimum strip-width. Meanwhile, the mutation towards $\boldsymbol{p}_d$ can intuitively keep the joint movement between $\boldsymbol{p}_{i,new}$ and $\boldsymbol{p}_{i-1}$ and $\boldsymbol{p}_{i+1}$ small. Sometimes, the search for $\boldsymbol{p}_i'$ may not succeed. To prevent a dead lock, a maximum number of iteration is set.

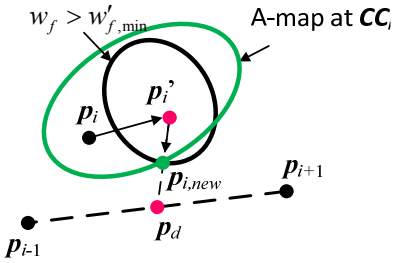


Figure 6.16 Mutation strategy regarding $w_{f,min}$

3) $\boldsymbol{p}_i$ being a non-critical posture in the chromosome. It is observed that a guided mutation towards $\boldsymbol{p}_d = 0.5(\boldsymbol{p}_{i-1}+\boldsymbol{p}_{i+1})$ often helps accelerate the converging of the GA. To preserve diversity of the population, probability of the guided mutation is set at a low level, e.g., 0.2. At the same time, for the sake of increasing diversity, random mutation, during which the gene (posture) is replaced with a randomly generated posture, is also conducted at a low

probability, e.g., 0.1. Whether random, guided, or no mutation will be conducted is determined randomly following the *roulette selection* algorithm.

Finally, it is worth mentioning that the genes of a chromosome should be picked for mutation in a randomized sequence. If a fixed sequence is used, the search would tend to be trapped in local optima. In summary, the developed knowledge-based mutation procedure is as follows (note that mutation is applied to $\boldsymbol{P}_{next}$, generated by elitism, immigration, and cross-over):

1) Suppose there are M chromosomes in the population and each chromosome is made up of N genes (postures). Set $k = 0$; $m = 0$.
2) Take $\boldsymbol{g}_k$ from $\mathbf{P}_{next}$.
3) Randomly pick a posture $\boldsymbol{p}$ from $\boldsymbol{g}_k$.

   If $\boldsymbol{p}$ is a critical posture

   Conduct mutation following strategy (1) or (2) accordingly.

   Else

   Determine the mutation strategy to use (guided, random, or no mutation) and proceed accordingly.

   Endif

   Set $m=m+1$
4) If $m<N$, go to (3); otherwise, mutation is completed for the current chromosome, go to (5).
5) Set $k = k+1$. IF $k = M+1$, stop; otherwise, set $m = 0$ and go to (2).

6.5.5. Replacement mechanism and stopping criterion

After $\boldsymbol{P}_{next}$ is generated with reproduction and mutation operators, the fitness of solutions within $\boldsymbol{P}_{next}$ will be evaluated and the best solution of the generation $\boldsymbol{g}_{l\_best}$ will be identified. $\boldsymbol{g}_{l\_best}$ will be compared with the best solution identified in the

whole search $\boldsymbol{g}_{g\_best}$, to decide whether $\boldsymbol{g}_{g\_best}$ need to be updated. Two scenarios exist for such a task:

1) In terms of both optimization objectives, $\boldsymbol{g}_{l\_best}$ is better than $\boldsymbol{g}_{g\_best}$, i.e., $J_{\max,l\_best} < J_{\max,g\_best}$ and $w_{f,\min}^{l\_best} > w_{f,\min}^{g\_best}$. This means $\boldsymbol{g}_{l\_best}$ is closer to the pareto-front than $\boldsymbol{g}_{g\_best}$ and $\boldsymbol{g}_{g\_best}$ will be replaced with $\boldsymbol{g}_{l\_best}$. On the contrary, if we have $J_{\max,l\_best} > J_{\max,g\_best}$ and $w_{f,\min}^{l\_best} < w_{f,\min}^{g\_best}$, $\boldsymbol{g}_{g\_best}$ will be unaffected.

2) $\boldsymbol{g}_{l\_best}$ excels regarding only one optimization objective. In such circumstances, we propose to evaluate the relative fitness of $\boldsymbol{g}_{l\_best}$ over $\boldsymbol{g}_{g\_best}$. The idea is to check whether the improvement regarding one optimization objective is enough to counterbalance the deterioration regarding the other objective. The relative fitness of $\boldsymbol{g}_{l\_best}$ over $\boldsymbol{g}_{g\_best}$ is given as:

$$fr\left(\boldsymbol{g}_{l\_best}, \boldsymbol{g}_{g\_best}\right) = c_1 \frac{J_{\max,l\_best}^{-1} - J_{\max,g\_best}^{-1}}{J_{\max,g\_best}^{-1}} + c_2 \frac{w_{f,\min}^{l\_best} - w_{f,\min}^{g\_best}}{w_{f,\min}^{g\_best}} \qquad (6.19)$$

If $fr(\boldsymbol{g}_{l\_best}, \boldsymbol{g}_{g\_best})>0$, it means $\boldsymbol{g}_{l\_best}$ is the better solution and replacement will take place.

After updating for $\boldsymbol{g}_{g\_best}$ is finished, we will set $\boldsymbol{P}_{current} = \boldsymbol{P}_{next}$ and search continues. Meanwhile, the search will terminate when $\boldsymbol{g}_{g\_best}$ shows no improvement for a consecutive M iterations or when the maximum iteration is reached.

6.5.6. The overall GA algorithm

In summary, the complete procedure for optimization of cutter postures over a single tool-path with the proposed GA algorithm is given as follows (some of the parameters used for the GA are summarized in Tab. 6.2.):

1) Using the method introduced in Section 6.5.2, initialize a population of M individuals, each containing N genes.

2) Set the initial population as $\boldsymbol{P}_{current}$ and set $\boldsymbol{P}_{next}$ = NULL. Rank the random solutions of $\boldsymbol{P}_{current}$ based on nominal fitness. Identify the best solution of $\boldsymbol{P}_{current}$ as $\boldsymbol{g}_{l\_best}$. Set the global best solution $\boldsymbol{g}_{g\_best} = \boldsymbol{g}_{l\_best}$.
3) Fill $\boldsymbol{P}_{next}$ through reproduction using the method introduced in Section 6.5.3.
4) Conduct mutation for solutions of $\boldsymbol{P}_{next}$ using the method introduced in Section 6.5.4.
5) Set $\boldsymbol{P}_{current} = \boldsymbol{P}_{next}$. Rank the solutions of $\boldsymbol{P}_{current}$ and update $\boldsymbol{g}_{l\_best}$. Compare $\boldsymbol{g}_{l\_best}$ with $\boldsymbol{g}_{g\_best}$ and update $\boldsymbol{g}_{g\_best}$ if $\boldsymbol{g}_{l\_best}$ has better fitness.
6) Check against the stopping criteria. IF the criterion is met, exit and output the best solution $\boldsymbol{g}_{g\_best}$. ELSE, set $\mathbf{P}_{next}$ = NULL. Go to (3).

Table 6.2 Parameters used for the developed GA algorithm

Population size	**Cost function weight**		**Mutation probability**		
	c_1	c_2	Group 1	Group 2	Group 3
	0.5	0.5	1.0	1.0	Guided: 0.2 Random: 0.1
20	**Reproduction probability**			**Stopping criteria**	
	Elitism 0.05	Immigration 0.1	Cross-over 0.85	# of null generations > 20 Max Iteration = 100	

6.6. Discussion

In this chapter, methods are proposed to generate optimal tool-paths with respect to machining efficiency and tool-path smoothness. A significant feature of the proposed method is that tool-path smoothness is measured by the joint movements of 5-axis machines, instead of the angular difference between cutter postures in the workpiece frame. Such a measure truly reflects the 'real' smoothness of tool-paths. To this end, the inverse kinematic transformation of general 5-axis machines is established first, and the methods to obtain joint locations for evaluation of tool-path smoothness are provided. The proposed methods are subject to the constraints of both interference

avoidance and scallop height tolerance. While the first constraint is straight-forward, methods to obtain the low bound on B-CSW regarding the second constraint are provided. The heuristic for determining path interval between tool-paths is also modified to provide larger search space for posture optimization.

Two methods for optimized posture assignment have been developed. The first method takes a 'repairing' approach. Preliminary CL paths are generated based on the Max-CSW heuristic. The CLs will then be converted to joint locations and divided into stable and unstable clusters. Adjustment for the unstable clusters is then carried out using a PSO-based search algorithm. Efforts have been made to customize the PSO to suit the nature of our problem, including update methods and introduction of mutation operators.

The second approach aims at optimizing the whole posture sequence regarding the 2 objectives, the maximum joint travel J_{max} and minimum F-CSW $w_{f,min}$, simultaneously. A GA-based optimization algorithm is developed with a specially designed fitness function to accommodate both objectives. Moreover, knowledge-based mutation operators are incorporated into the algorithm, on top of the classic GA operators, to guide the search towards better solutions more efficiently.

CHAPTER 7

CASE STUDIES AND DISCUSSIONS

The developed PSO and GA algorithms, introduced in Chapter 6, have been implemented using C++. In this chapter, several case studies are presented for the testing of the two posture optimization algorithms. To demonstrate the capabilities of the proposed algorithms, workpieces are purposely designed to induce drastic joint movements. Performance comparison is conducted between the method based on the Max-CSW heuristic and the two optimization methods.

As shown in Fig. 7.1, the machine configuration used in this study is taken from a CMS POSEIDON® 5-axis machining center. This machine is of the HEAD/HEAD configuration with B(secondary)-C(primary) axis configuration. The travel limits on the two revolute joints are given as: B (-120º ~ 120º) and C (-270º ~ 270º).

Figure 7.1 Machine structure and axis configuration of CMS machine

Due to time and cost reasons, actual machining is not conducted on the CMS machine. Instead, a virtual machine (an exact replica of the CMS machining center) has been built using the simulation software VERICUT® as shown in Fig. 7.2. The post-processing of the CL data is conducted with a custom built post-processer

developed in MASTERCAM®. The assumptions made about post-processors in Chapter 6 are strictly followed, i.e., the machine joints are re-set to zero position at the beginning of every tool-path through fast movement and each joint location is selected as the one with minimum joint movement from the previous location.

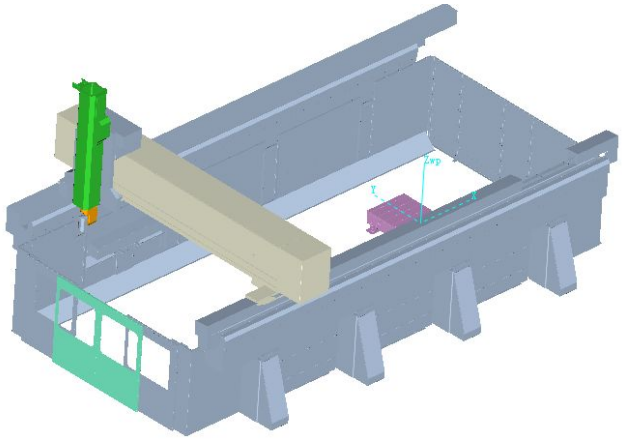
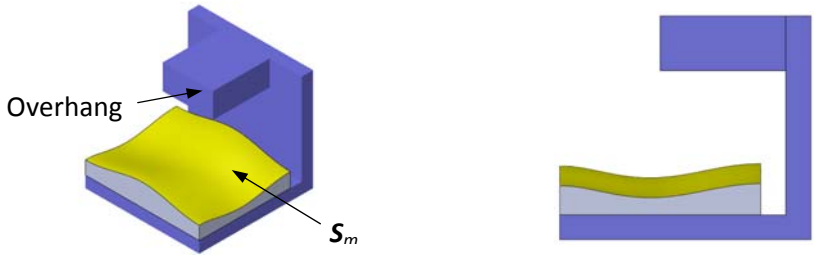


Figure 7.2 The virtual machine of CMS POSEIDON built in VERICUT®

7.1. Case Study 1: Posture Repair with the PSO-based Algorithm

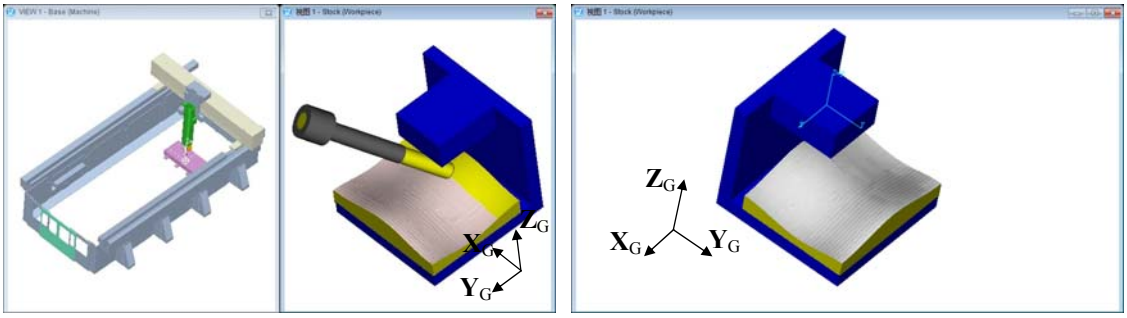
The workpiece involved in the first case study is shown in Fig. 7.3 with a cutter of {R = 6, r_f= 2, L = 40}. The overhang was designed to induce drastic joint movement that only covers half of the workpiece. As a result, the preliminary tool-paths generated on the outer half of the machining surface are smoother with fewer cases of beyond-limit joint travel (the upper-limit set for joint movement is 30°). On the contrary, the tool-paths near or below the overhang have many violations for smoothing (see Fig. 7.5).



(a) (b)

Figure 7.3 Test workpiece with overhang (a) isometric view (b) side view

During tool-path modification using the PSO algorithm, the surface finish tolerance and the scallop height tolerance are both set as 0.1mm. Running on a PC (CPU of 2.33GHZ and RAM of 2 GB), a total of 63 CL paths ware generated in 47 minutes. The tool-paths are put to testing on the virtual machine in VERICUT. Machining simulation reveals no machining interference (see Fig. 7.4).

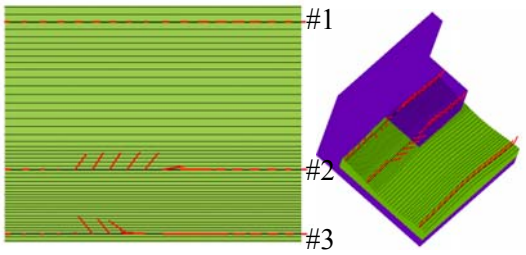


(a) Machining center and cutter (b) Workpiece after machining

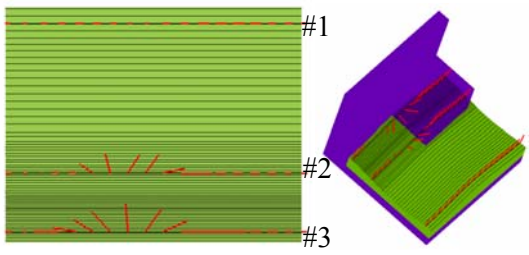
Figure 7.4 Machine simulation with the test workpiece

In Fig. 7.5a, for the sake of clearer view, 3 preliminary tool-paths from different regions are shown to illustrate the influence of the overhang on cutter posture determination using the Max-CSW heuristic (for other tool-paths, only the positions are shown). In the 'open' area, the preliminary cutter postures generated by heuristic are almost identical along tool-path #1 (the tool-path furthest from the overhang). Meanwhile, under the overhang, drastic posture change occurs when the tool-path enters and exists from the shadow of the overhang. The preliminary tool-paths were then corrected with the PSO smoothing algorithm and the resulted CL paths are shown in Fig. 7.5b for a comparison (only 3 CL paths at approximately the same position of #1, #2, and #3 are shown with their postures). It can be seen that tool-paths in the 'open' area are identical (tool-path intervals, cutter postures, etc) for the two methods while for the tool-paths under the overhang, the PSO smoothed tool-paths clearly have better smoothness. The maximum joint travel on each tool-path

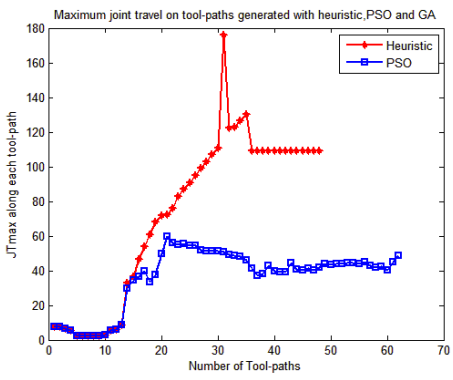
generated with and without the PSO-smoothing algorithm is plotted in Fig. 7.5c. Presented in Fig. 7.5d are the strip widths along the pick-feed direction. It can be seen that the machining efficiency is affected by the smoothing algorithm, as the smoothed tool-paths may produce smaller $w_{f,min}$, causing the path interval to become smaller. The preliminary tool-paths have 48 single paths while the smoothed tool-paths consist of 63 single tool-paths, which features an increase of 23.8%. Another negative effect from smoothing is related to computation efficiency. On the same PC, the preliminary tool-paths took 7 minutes to generate, which is only of a fraction of that used by the PSO algorithm.



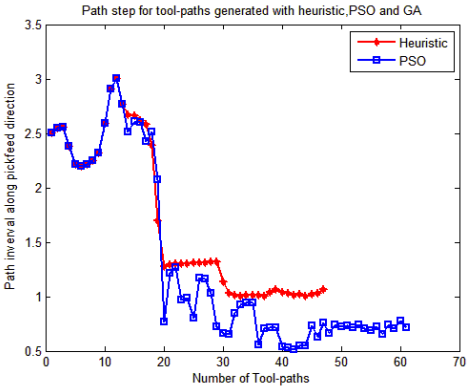
(a) The preliminary



(b) The PSO smoothed



(c) Based on max. joint travel

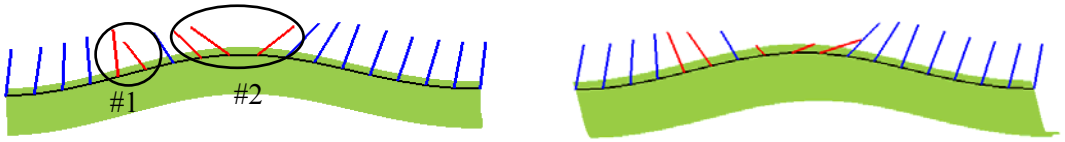


(d) Based on path interval

Figure 7.5 Comparison between preliminary and PSO smoothed tool-paths

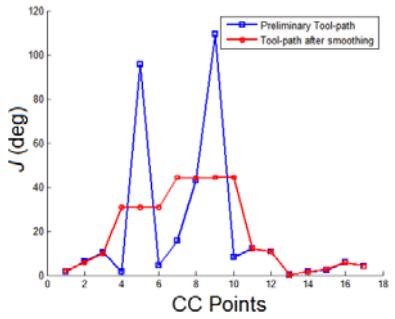
For a more in-depth comparison, a single tool-path below the overhang is taken as an example. The preliminary and smoothed tool-paths are shown in Fig. 7.6 (only the machining surface is shown for a clearer view). The two unstable clusters

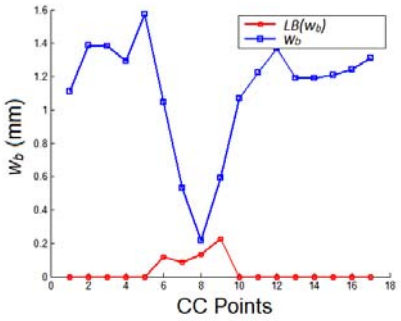
on the preliminary tool-paths are highlighted in red in Fig. 7.6a. On the smoothed tool-path shown in Fig. 7.6b, both unstable clusters are eliminated.



(a) Postures before smoothing

(b) Postures after smoothing





(c) Joint travels

(d) w_b vs. $LB(w_b)$ on B-CSW

Figure 7.6 A single tool-path before/after smoothing

Shown in Fig. 7.6c are the joint movements on the tool-path before and after smoothing. It can be seen that on the preliminary path, the joint travel within a stable cluster is quite small. However, in face of sudden change of machining environment (machining obstacles in this case), drastic joint movements occurred. What the PSO smoothing algorithm does is to spread the drastic joint movements evenly among several CLs to reduce the maximum smooth joint movement. Following this principle, it's worth noting that, as limited by the stable CLs on both side of an unstable cluster, the pre-set upper limit for joint movement (30°) cannot always be met, which is a limitation of this repairing approach.

The B-CSWs (w_b) at all the CC points on the smoothed tool-path are shown in Fig. 7.6d. The red line gives the $LB(w_b)$ on the B-CSW at all the CC points. It can be seen that w_b stays above the limit at all the CC points, which guarantees that the tolerance on scallop height is not exceeded. It is worth noting that at several CC

points, the $LB(w_b)$ is 0, which means that the F-CSWs at the CC points on the previous path is larger than the path interval. This is because the path interval is determined based on the minimum F-CSW ($w_{f,min}$) on the previous path.

To test the repeatability of the PSO algorithm, for the unstable cluster #2 shown in Fig. 7.6a, the PSO algorithm was run for a consecutive of 50 times, with a maximum number of 300 iterations for each run. The maximum joint travels of the 50 resultant tool-paths are shown in Fig. 7.7. The average maximum joint travel obtained is 43.6° (the minimum is 42.6° and the maximum is 49.2°). This clearly demonstrates the high level of repeatability of performance from the PSO algorithm.

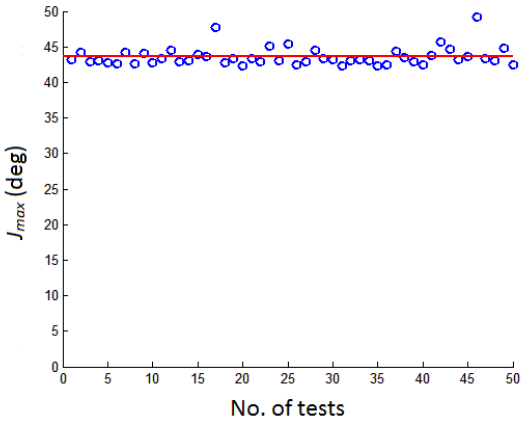


Figure 7.7 Repeatability test results with a single unstable cluster

7.2. Case Study 2: Posture Optimization with the GA-based Algorithm

In this section, the workpiece used in case study 1 is firstly used to test the GA-based tool-path optimization algorithm and a comparison between the performance of GA and PSO algorithms on this example is presented. In the second part, a benchmark example specially designed for this optimization algorithm with duo objectives is presented.

7.2.1. Comparison between the GA- and PSO-based algorithms

The major limitation of the repairing approach with the PSO-based algorithm is imposed by the stable posture clusters in the preliminary posture sequence. As such postures are not subject to adjustment, the repairing can only be carried out in a local manner. At the same time, the PSO–based algorithm also benefits from such a limitation, as the dimension of the solution space is relatively low, compared to that of the GA-based method (unstable cluster vs. whole tool-path). This can be seen from running the two algorithms against the same case in Fig. 7.3 in which the GA-based algorithm took 72 minutes to complete while the PSO-based algorithm took 47 minutes.

The GA generated tool-paths are shown in Fig. 7.8. Unlike the tool-paths generated with PSO, where only the unstable clusters are adjusted, optimization is conducted regarding all the cutter postures along a single tool-path. As a result, the search space would be enlarged, leading to better tool-path smoothness along the whole tool-path that even can be directly observed, compared with the PSO generated tool-paths shown in Fig. 7.6.

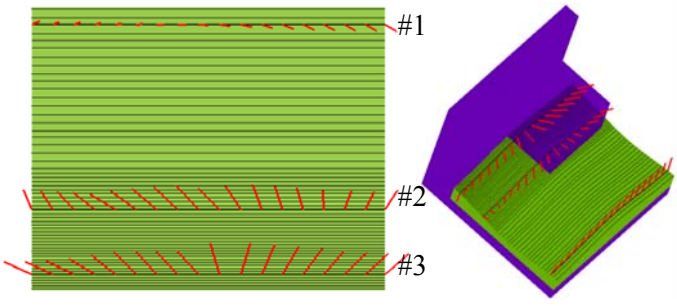
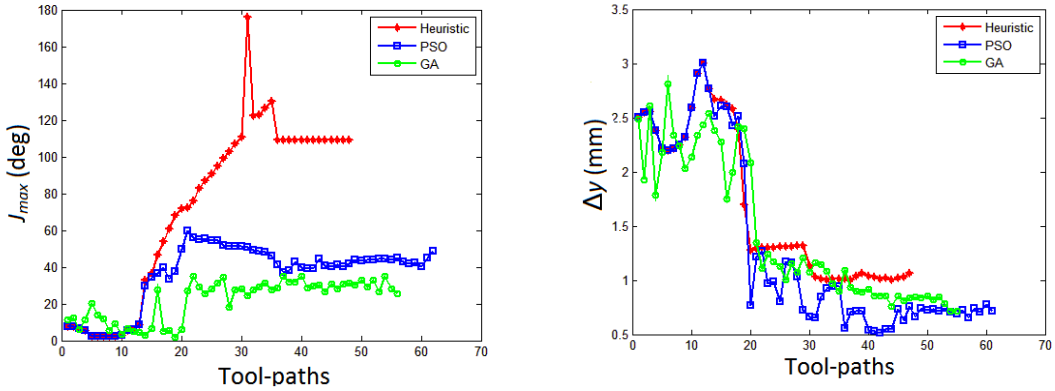


Figure 7.8 Tool-path generated with the dual-objective optimization based on GA

The performance of the GA generated tool-paths is plotted with that of the PSO generated tool-paths and the Max-CSW generated tool-paths in Fig. 7.9. In terms of joint movement, the GA tool-paths have the best overall performance. In terms of

machining efficiency (path interval), the GA tool-paths are shorter than the PSO tool-paths (56 vs. 63), although the max-CSW tool-paths are the shortest with 48 passes, which is expected. It is worth noting that in the 'open' area unblocked by the overhang, the Max-CSW algorithm actually produces slightly better performance than the GA algorithm. This is due to the randomized nature of GA, causing it difficult to lock on to the real optimal solution but on to a near optimal solution. In the area covered by the overhang, as indicated by the performance curves, the GA-based algorithm clearly has the edge.



(a) Max. joint movement on a path J_{max} (b) Path interval Δy_i

Figure 7.9 Performance comparison among Max-CSW, PSO, and GA

Finally, the machining simulation using the GA tool-paths was conducted using VERICUT and the result is shown in Fig. 7.10, in which no machining interferences was found.

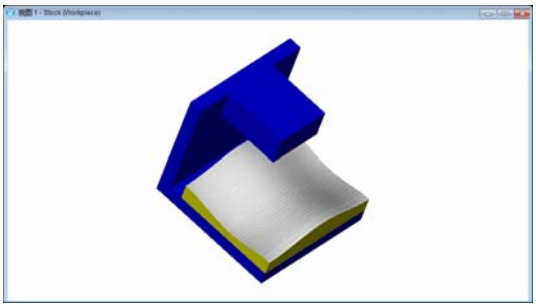
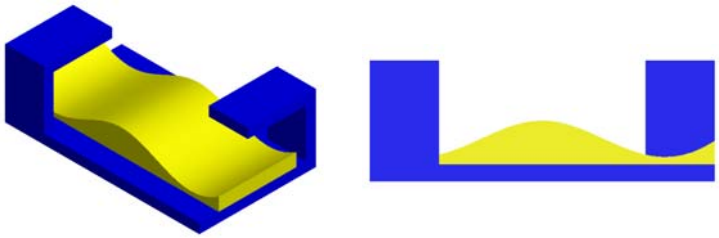


Figure 7.10 Machining simulation for the GA tool-paths

7.2.2. Test of the GA algorithm on a benchmark workpiece

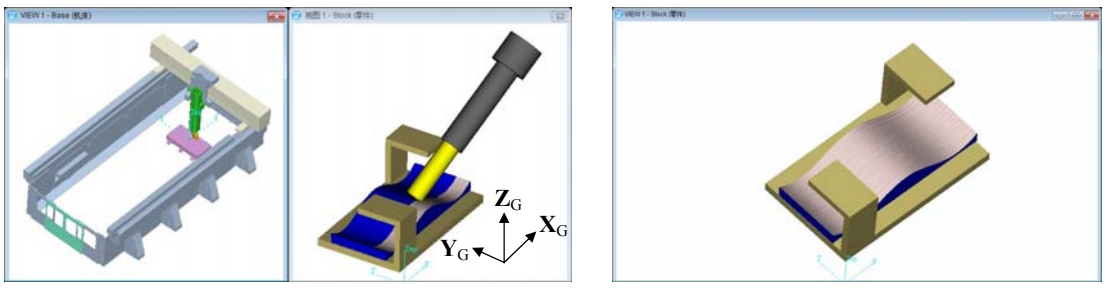
The benchmark workpiece designed for the GA algorithm is shown in Fig. 7.11. It comes with a specially designed overhang to test the GA algorithm. The overhang is introduced to make sure the CC points at the two ends of the workpiece share as little common accessible posture range as possible, so that the capabilities of the proposed algorithm can be fully demonstrated.



(a) Isometric view (b) Side view

Figure 7.11 Workpiece with specially designed overhang

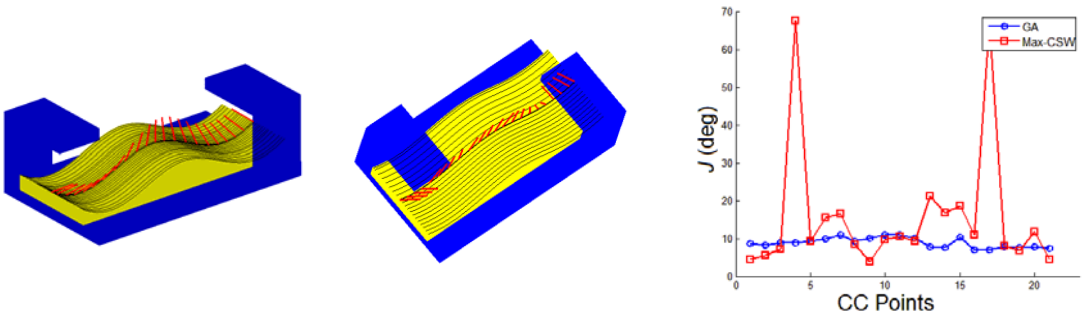
The tool-path direction is along $\mathbf{X}_G$ of the workpiece frame. A torus cutter of $\{R = 6, r_f = 2, L = 40\}$ plus holder is used for the machining. Both the surface shape tolerance τ and scallop tolerance h were set as 0.1mm. It took 35 minutes for the GA algorithm to complete the generation of 28 CL paths. Machining simulation was conducted using VERICUT (see Fig. 7.12a) and the finished part is shown in Fig. 7.12b. Subsequent inspection by the VERICUT shows no interference and no undercut or overcut beyond the given tolerance.



(a) (b)

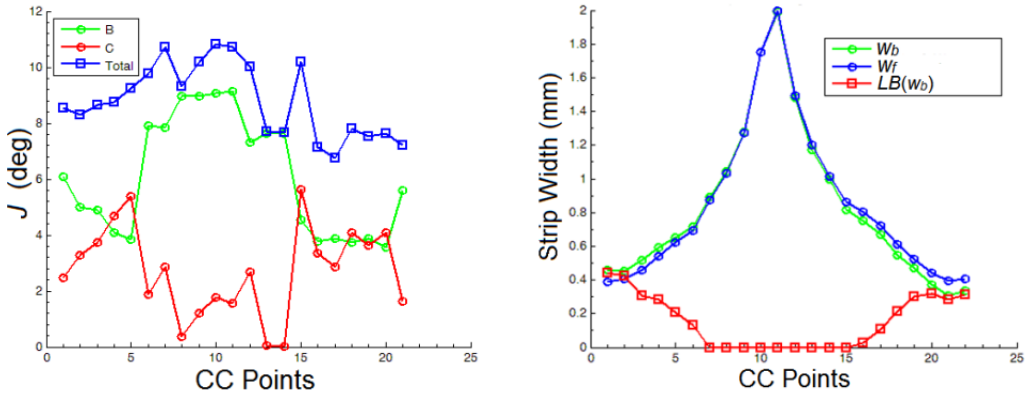
Figure 7.12 Machine simulation of the case study

For a clearer view, an individual CL path at the center of the workpiece is displayed with the tool-postures at the CC points as shown in Fig.7.13a. For the remaining CL paths, only their locations are shown. The effect of the two over-hanged obstacles at the two opposite corners on the cutter postures is clearly demonstrated.



(a) CL path (GA) (b) CL path (Max-CSW) (c) Comparison of joint movement

Figure 7.13 Comparison of CL paths from GA and Max-CSW heuristic



(a) Joint movement at CC points (b) CSWs at CC points

Figure 7.14 Performance of the CL path generated using GA

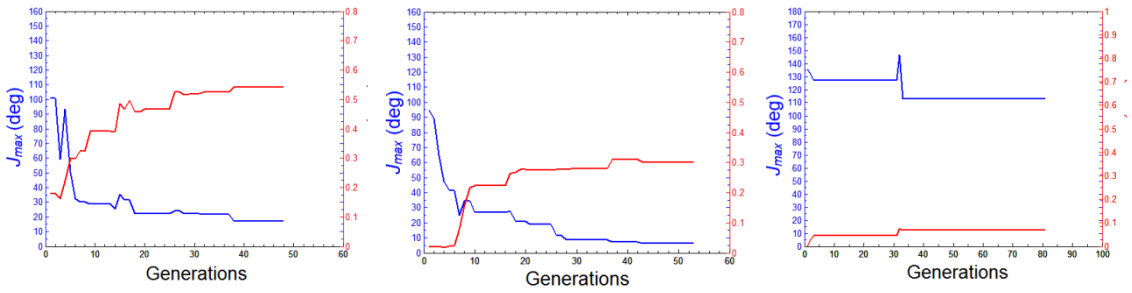
A more in-depth analysis on the joint movement between neighboring CC points of the center CL path is shown in Fig. 7.14a (B, C, and the total). It can be seen that the total joint movement is maintained between 7˚-11˚ throughout. At the same time, the cutting strip-widths (w_b and w_f) at the CC points along the CL path are

shown in Fig. 7.14a. The red line indicates the low-bound of w_b at every CC point as a requirement for scallop. Clearly, this constraint is well maintained. It is also observed that at some CC points the low-bound equals zero, indicating that the corresponding w_f at the previous CL path is larger than the path interval.

To investigate the quality of the generated CL path in terms of joint movement, another set of CL paths along the same direction is generated using the Max-CSW heuristic. As shown in Fig. 7.13b, the Max-CSW solution has a total of 20 individual CL paths, showing some decent reduction in total tool-path length. In terms of tool-path smoothness, a direct comparison is made between the GA path and Max-CSW path at the center location, as shown in Fig. 7.13c. It can be clearly seen that, for the Max-CSW path, at the border between the "blocked" and the "open" on the machining surface, there are sudden huge jumps of the joint movement (up to nearly 70˚). This is a clear indication that the GA-base algorithm is able to generated CL paths with much improved smoothness.

The influence of weight factors assigned to the two optimization objectives in the fitness function of GA was also studied. Two sets of weight factors, (c_1 = 0.5, c_2 = 0.5) and (c_1 = 0.8, c_2 = 0.2), were used to generate the center path separately. The evolution of the best chromosome is displayed with $J_{\max}$ and $w_{f,min}$ in Fig. 7.15a and b, respectively. Clearly, $J_{\max}$ (c_1 = 0.5) > $J_{\max}$ (c_1 = 0.8) and $w_{f,min}$ (c_2 = 0.5) > $w_{f,min}$ (c_2 = 0.2). Hence, the weight assignment can be used as an effective tool to favor the search towards a particular optimization objective. The effect of the knowledge-based mutation operator was also studied by running the GA for the center path (c_1 = c_2 = 0.5) without the guided mutation. The evolution of the best chromosome is shown in Fig. 7.15c. Compared with the final solution shown in Fig. 7.15a, the solution is

poorer in both $J_{\max}$ and $w_{f,min}$. Therefore, the importance of the developed mutation operator is clearly demonstrated.



(a) c_1 =0.5, c_2 = 0.5 (b) c_1 = 0.8, c_2 = 0.2 (c) GA without mutation

Figure 7.15 Characteristics of the developed GA

To verify the repeatability of the GA algorithm, a test consisting of 50 runs of the GA (c_1 = 0.5, c_2 = 0.5) with the center path was conducted. The distribution of the solutions in terms of $J_{\max}$ and $w_{f,min}$ is shown in Fig. 7.16. Based on the study of all the obtained CL paths, any solution featuring $w_{f,min}$ less than 0.5 or J_{max} larger than 30° is deemed unsatisfactory (the boundary is marked with dashed lines). In this case, only 3 solutions out of 50 (less than 10%) are considered unsatisfactory. Therefore, the repeatability of the developed GA is considered satisfactory.

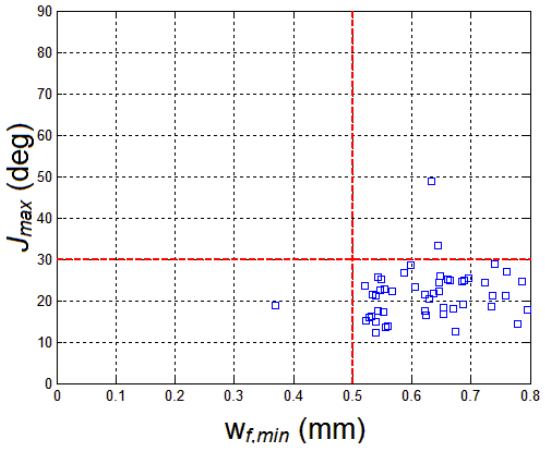


Figure 7.16 Repeatability test for the GA with the center path

7.3. Discussion

In this chapter, testing is conducted for the two developed optimization algorithms based on PSO and GA respectively. As demonstrated by the testing results, both methods produced tool-paths with superior performance over the Max-CSW heuristic-base algorithm. Although the Max-CSW heuristic works well for 'open' surface areas with non-critical surface property, it will produce drastic joint movement when faced with sudden change of machining environment. With the proposed PSO method, problematic tool-path segments can be identified and smoothed. The effect of smoothing is that the drastic joint movement corresponding to a certain pair of CLs will be spread evenly over the whole unstable cluster.

In the meantime, the GA-based algorithm attempts to optimize posture assignment regarding both machining efficiency and tool-path smoothness simultaneously. The attempt is successful as significant improvements regarding both objectives are observed during the search. When tested on the same workpiece with the repairing approach, the GA-based method produces results that excel regarding both objectives. Meanwhile, in terms of computational cost, the repairing approach outperforms the GA-based method simply because only unstable clusters are taken for modification.

CHAPTER 8

CONCLUSIONS AND FUTURE WORK

In this thesis, research efforts are made to tackle the various problems and difficulties present in automatic process planning for 5-axis point milling of sculpture surfaces (finish cut). The tasks of process planning for 5-axis machining are generally two-fold, i.e. selection of cutters and generation of tool-paths. For the first task, machining efficiency is taken as the major performance criterion. For the second task, besides machining efficiency, tool-path smoothness is also taken into consideration. Meanwhile, both cutter selection and tool-path generation are subject to the constraints of cutter accessibility. Tool-path generation is also constrained by surface finish requirements. The objective of this research is to find optimized solutions for the various tasks in process planning, so that satisfactory performance can be delivered during the actual machining. In this chapter, research presented in this thesis will be summarized and possible directions for future research will be pointed out.

8.1. Conclusions

The research achievements presented in this thesis are summarized in the following:

- **Fast construction of accessible posture range**

The A-map algorithm proposed in our previous research is capable of identifying the complete accessible posture range for a cutter at a surface point. However, due to its representation with angle pairs of (λ, θ) in the local frame, its application is limited to single CC points. In this thesis, a new A-map representation scheme based on posture chains is proposed, which is in the workpiece frame. The most important change brought about by such a transformation is that the A-maps at

different points are now directly comparable. With a dedicated method for checking the accessibility of single cutter postures, the intersection of A-maps at different surface points can be easily obtained.

Based on the A-maps at the sampled points, the posture chain representation scheme allows the A-map at any arbitrary surface point to be obtained through interpolation in a conservative but safe manner. Such a function makes the most out of the existing accessibility information and greatly saves computational load, It is especially important for tool-path optimization, where the A-maps at all the CC points are needed as the feasible search space for cutter postures.

- **Tool-path length prediction method for multi-cutter selection**

The major difficulty of multi-cutter selection is the lack of a reliable estimation method for total tool-path length for a given cutter/surface region combination. The existing heuristic-based method from our previous research suffers from poor accuracy.

The new tool-path length prediction method proposed in this thesis is based on neural network (NN) in simulating the implicit relationship between cutter size, machining surface and the final tool-path length. By collecting inputs at dispersed data points over the whole machining surface/region, comprehensive information on surface geometry can be collected. Meanwhile, through the design of a machining characteristic parameter, information on machining strip width over the whole surface is also obtained. Thanks to these two designs, the developed neural network is able to make much more accurate tool-path length estimation than the heuristic-based method.

- **Improvement of CC tool-path generation procedures**

In our previous work, a complete system for CC tool-path generation has been proposed. Tasks covered in the system are cutting direction selection, CC point

generation, posture assignment and side-step calculation. All these tasks are carried out based on a fixed heuristic for selecting the posture with maximum/near-maximum machining efficiency from the A-map at surface point. Assuming the same heuristic is used, improvements of CC tool-path generation procedures are performed in the following aspects.

The possibility of interference during interpolation (IDI) is taken into consideration with a new method for step-forward distance calculation. The detection of IDI is carried out by performing collision query between the enveloping surface (e-surface) of the cutter's movement and the workpiece. As elimination of IDI can be achieved through adjusting the interval between CC points, the function for detection and elimination of IDI is built into the step-forward determination method. In this way, the process of determining CC point locations is subject to the requirements of both surface finish and avoidance of IDI. The consideration for IDI serves as an effective remedy for the problem caused by the fixed heuristic for posture assignment.

- **Evolutionary algorithms for optimized posture determination**

Considering the shortcomings of the heuristic based method for posture assignment, such as solution lacking optimization and lack of consideration for tool-path smoothness, optimized methods are proposed for posture assignment at CC points. The objectives include machining efficiency and tool-path smoothness. The most significant improvement over previous research is that tool-path smoothness is measured as the amount of joint movements to be conducted on 5-axis machines of a specific configuration. In this way, 'real' smooth tool-paths can be generated. Drastic joint movements as a result of singular configuration or joint rewinding, as well as those caused by unsmooth CL data are eliminated altogether.

Two evolutionary approaches are proposed for optimized posture assignment. The first tries to 'repair' problematic tool-path segments in the preliminary tool-paths. The second attempts to optimize a whole posture concerning machining efficiency and path smoothness simultaneously. Different optimization tools (PSO vs. GA) are used for the two approaches with newly developed knowledge based operators. Case studies show that the optimization targets are achieved in both approaches. Through comparison of performance, the second approach produces tool-paths of better quality regarding both optimization objectives. For the repairing approach, the level of optimization is not as high, as machining efficiency is not taken as an objective for optimization and optimization regarding smoothness is constrained by the preliminary tool-paths. On the other hand, as only unstable clusters are optimized, the computational cost for the repairing approach is relatively low.

With methods proposed in this thesis, the existing system for process planning of 5-axis machining is significantly enhanced. The complete workflow of the developed system is provided in Fig. 8.1. It can be seen process planning starts with a comprehensive accessibility evaluation over a high-density point cloud. Based on the obtained accessibility information, cutter selection and tool-path generation can be carried out. Some non-optimal procedures, such as posture assignment by heuristic, are still preserved in the system, as they can be used for simpler machining jobs to save computational cost. The process planners can chose which procedures to use based on the complexity of the workpiece. Except for such decisions, the system requires no human interference, which is a crucial benefit from the high level of automation of the proposed system.

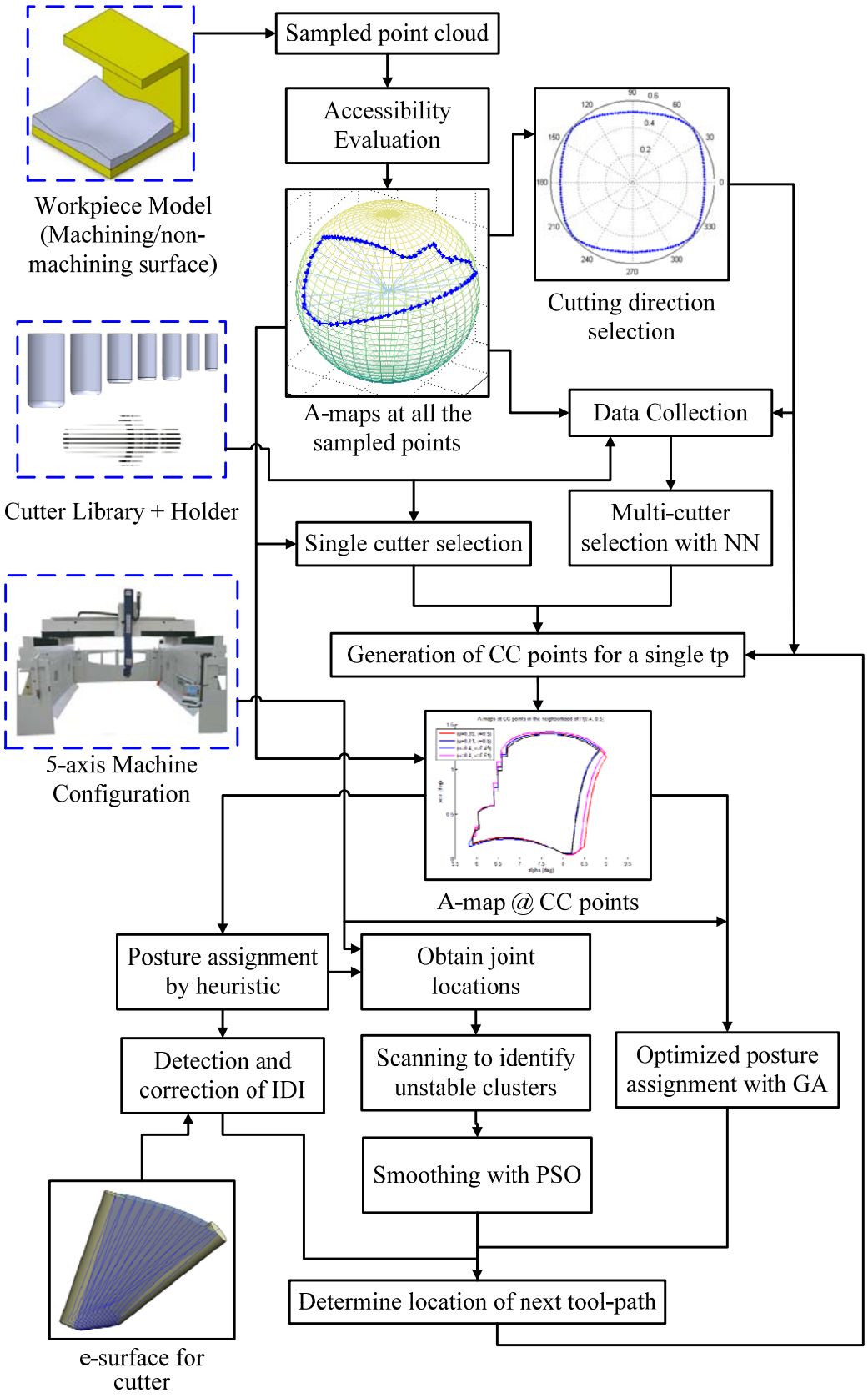


Figure 8.1 Proposed process planning system for 5-axis machining

8.2. Recommendations for Future Work

Several limitations still exist for the current process planning system, which may indicate possible directions for future research:

- The tool-path length prediction method is only intended for tool-path generation with postures assigned by the fixed heuristic. When other cutter posture assignment strategies are used, such as the optimized posture assignment method proposed in Chapter 6, the solution accuracy will deteriorate. Thus the method should be extended to cover different posture assignment strategies. Moreover, as tool-path smoothness is also an important aspect of tool-path performance, it should also be considered in multi-cutter selection.
- In this thesis, tool-path interval is determined by the minimum backward and forward cutting strip widths on neighboring paths. Even if some postures may produce larger material removal rate, they cannot contribute to improve the overall machining efficiency. This problem is unique to iso-planar tool-paths. For better machining efficiency, other tool-path patterns should be explored, such as the iso-scallop-height pattern. In that case, the optimization methods for posture assignment should also be re-designed in order to produce satisfactory performance.
- In this thesis, optimization is carried out regarding the amount of movements on the machine joints. However, the kinematic performance of tool-paths covers more than tool-path smoothness only. When a 5-axis machine moves the cutter between the cutter locations, the movements of the joints are quite complicated, composing of a series of acceleration and deceleration phases. Meanwhile, the allowable speed, acceleration and jerk for a machining joint are limited by the servo mechanism driving the joint. Inappropriate CL data may require the machine joints to exceed such limits, which is physically impossible. As a result, feed-rate drop or chatter may be

produced, harming the tool-path's performance. For future work, the travelling schedule for machine joints could be looked into. Optimization could be carried out regarding the CL data, so that the generated tool-path is complaint with the physical capabilities of the machine joints.

- Cutter postures also affect the engagement between the cutter and the workpiece, which in turn determines the cutting force. Cutting force, on the other hand, changes in cyclic patterns, causing vibration and chatter during the machining process. Such a fact should also be considered for posture assignment. Optimization efforts could be made to produce cutter postures with minimized radial cutting force for better tool-path performance.

REFERENCES

Amenta, N., D. Attali and O. Devillers (2007). Complexity of Delaunay triangulation for points on lower-dimensional polyhedra. Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms. New Orleans, Louisiana, Society for Industrial and Applied Mathematics**:** 1106-1113.

Anotaipaiboon, W. and S. S. Makhanov (2005). "Tool path generation for five-axis NC machining using adaptive space-filling curves." International Journal of Production Research **43**(8): 1643-1665.

Anotaipaiboon, W., S. S. Makhanov and E. L. J. Bohez (2006). "Optimal setup for five-axis machining." International Journal of Machine Tools and Manufacture **46**(9): 964-977.

Apro, K. (2008). Secrets of 5-Axis Machining. New York, Industrial Press, Inc.

Arvo, J. (1991). Graphic Gems II. Ithaca, AP Professional.

Balasubramaniam, M., S. E. Sarma and K. Marciniak (2003). "Collision-free finishing toolpaths from visibility data." Computer-Aided Design **35**(4): 359-374.

Barakchi Fard, M. and H.-Y. Feng (2009). "Effect of tool tilt angle on machining strip width in five-axis flat-end milling of free-form surfaces." The International Journal of Advanced Manufacturing Technology **44**(3): 211-222.

Bi, Q.-Z., Y.-H. Wang and H. Ding (2009). "A GPU-based algorithm for generating collision-free and orientation-smooth five-axis finishing tool paths of a ball-end cutter." International Journal of Production Research **48**(4): 1105 - 1124.

Bi, Q. Z., Y. H. Wang, L. M. Zhu and H. Ding (2011). "Generating collision-free tool orientations for 5-axis NC machining with a short ball-end cutter." International Journal of Production Research **48**(24): 7337 - 7356.

Bohez, E. L. J. (2002). "Five-axis milling machine tool kinematic chain design and analysis." International Journal of Machine Tools and Manufacture **42**(4): 505-520.

Can, A. and A. Ünüvar (2010). "A novel iso-scallop tool-path generation for efficient five-axis machining of free-form surfaces." The International Journal of Advanced Manufacturing Technology **51**(9): 1083-1098.

Castagnetti, C., E. Duc and P. Ray (2008). "The Domain of Admissible Orientation concept: A new method for five-axis tool path optimisation." Computer-Aided Design **40**(9): 938-950.

Chen, T., P. Ye and J. Wang (2005). "Local interference detection and avoidance in five-axis NC machining of sculptured surfaces." The International Journal of Advanced Manufacturing Technology **25**(3): 343-349.

Chiou, C.-J. and Y.-S. Lee (2002). "A machining potential field approach to tool path generation for multi-axis sculptured surface machining." Computer-Aided Design **34**(5): 357-371.

Chiou, C. J. and Y. S. Lee (2002). "Swept surface determination for five-axis numerical control machining." International Journal of Machine Tools and Manufacture **42**(14): 1497-1507.

Chiou, J. C. J. (2004). "Accurate tool position for five-axis ruled surface machining by swept envelope approach." Computer-Aided Design **36**(10): 967-974.

Chiou, J. C. J. and Y. S. Lee (2005). "Optimal Tool Orientation for Five-Axis Tool-End Machining by Swept Envelope Approach." Journal of Manufacturing Science and Engineering **127**(4): 810-818.

Cho, J. H., J. W. Kim and K. Kim (2000). "CNC tool path planning for multi-patch sculptured surfaces." International Journal of Production Research **38**(7): 1677-1687.

Choi, B. K. and R. B. Jerard (1998). Sculptured surface machining: theory and applications. Boston, Kluwer Academic Publishers

Chu, C.-H. and H.-T. Hsieh (2010). "Generation of reciprocating tool motion in 5-axis flank milling based on particle swarm optimization." Journal of Intelligent Manufacturing: 1-9.

Chu, C.-H., C.-T. Lee, K.-W. Tien and C.-J. Ting (2010). "Efficient tool path planning for 5-axis flank milling of ruled surfaces using ant colony system algorithms." International Journal of Production Research **49**(6): 1557 - 1574.

Craig, J. J. (2005). Introduction to robotics: mechanics and control. Upper Saddle River, NJ, Pearson Education.

de Berg, M., O. Cheong, M. van Kreveld and M. Overmars (2008). Computational Geometry: algorithms and applications. Berlin, Springer-Verlag.

Ding, S., M. A. Mannan and A. N. Poo (2004). "Oriented bounding box and octree based global interference detection in 5-axis machining of free-form surfaces." Computer-Aided Design **36**(13): 1281-1294.

Ding, S., M. A. Mannan, A. N. Poo, D. C. H. Yang and Z. Han (2003). "Adaptive iso-planar tool path generation for machining of free-form surfaces." Computer-Aided Design **35**(2): 141-153.

Ding, X. M., J. Y. H. Fuh and K. S. Lee (2001). "Interference detection for 3-axis mold machining." Computer-Aided Design **33**(8): 561-569.

DoCarmo, M. (1976). Differential Geometry of Curves and Surfaces. New Jersy, Prentice-Hall Inc.

Du, S., T. Surmann, O. Webber and K. Weinert (2005). "Formulating swept profiles for five-axis tool motions." International Journal of Machine Tools and Manufacture **45**(7-8): 849-861.

Elber, G. (1995). "Freeform surface region optimization for 3-axis and 5-axis milling." Computer-Aided Design **27**(6): 465-470.

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

EngelBrecht, A. P. (2005). Fundamentals of computational swarm intelligence. Hoboken, New Jersey, Wiley.

Fard, M. J. B. and H.-Y. Feng (2011). "New criteria for tool orientation determination in five-axis sculptured surface machining." International Journal of Production Research **49**(20): 5999-6015.

Gian, R., T. W. Lin and A. C. Lin (2003). "Planning of tool orientation for five-axis cavity machining." The International Journal of Advanced Manufacturing Technology **22**(1): 150-160.

Gong, H., L.-X. Cao and J. Liu (2005). "Improved positioning of cylindrical cutter for flank milling ruled surfaces." Computer-Aided Design **37**(12): 1205-1213.

Gong, H., L.-X. Cao and J. Liu (2008). "Second order approximation of tool envelope surface for 5-axis machining with single point contact." Computer-Aided Design **40**(5): 604-615.

Gong, H. and N. Wang (2009). "Optimize tool paths of flank milling with generic cutters based on approximation using the tool envelope surface." Computer-Aided Design **41**(12): 981-989.

Gottschalk, S., M. C. Lin and D. Manocha (1996). OBBTree: a hierarchical structure for rapid interference detection. Proceedings of the 23rd annual conference on Computer graphics and interactive techniques, ACM.

Gray, P., S. Bedi and F. Ismail (2003). "Rolling ball method for 5-axis surface machining." Computer-Aided Design **35**(4): 347-357.

Gray, P. J., S. Bedi and F. Ismail (2005). "Arc-intersect method for 5-axis tool positioning." Computer-Aided Design **37**(7): 663-674.

Haikin, S. (1998). Neural Networks: A Comprehensive Foundation, Pearson Education.

Hatna, A., R. J. Grieve and P. Broomhead (2000). "NC Machining of Trimmed Surfaces Maintaining Adjacent-Surfaces Integrity." The International Journal of Advanced Manufacturing Technology **16**(3): 196-204.

Hibbeler, R. C. (2009). Engineering Mechanics. Upper Saddle River, New Jersey, Pearson Prentice Hall.

Ho, M.-C., Y.-R. Hwang and C.-H. Hu (2003). "Five-axis tool orientation smoothing using quaternion interpolation algorithm." International Journal of Machine Tools and Manufacture **43**(12): 1259-1267.

Hosseinkhani, Y., J. Akbari and A. Vafaeesefat (2007). "Penetration–elimination method for five-axis CNC machining of sculptured surfaces." International Journal of Machine Tools and Manufacture **47**(10): 1625-1635.

Hwang, J. S. (1992). "Interference-free tool-path generation in the NC machining of parametric compound surfaces." Computer-Aided Design **24**(12): 667-676.

Jensen, C. G. and D. C. Anderson (1993). "Accurate tool placement and orientation for finish surface machining." Journal of Design and Manufacturing **59**(3): 251-261.

Jensen, C. G., W. E. Red and J. Pi (2002). "Tool selection for five-axis curvature matched machining." Computer-Aided Design **34**(3): 251-266.

Jun, C.-S., K. Cha and Y.-S. Lee (2003). "Optimizing tool orientations for 5-axis machining by configuration-space search method." Computer-Aided Design **35**(6): 549-566.

Kim, B. H. and B. K. Choi (2002). "Machining efficiency comparison direction-parallel tool path with contour-parallel tool path." Computer-Aided Design **34**(2): 89-95.

Kiswanto, G., B. Lauwers and J. P. Kruth (2007). "Gouging elimination through tool lifting in tool path generation for five-axis milling based on faceted models." The International Journal of Advanced Manufacturing Technology **32**(3): 293-309.

Lartigue, C., E. Duc and A. Affouard (2003). "Tool path deformation in 5-axis flank milling using envelope surface." Computer-Aided Design **35**(4): 375-382.

Lasemi, A., D. Xue and P. Gu (2010). "Recent development in CNC machining of freeform surfaces: A state-of-the-art review." Computer-Aided Design **42**(7): 641-654.

Lauwers, B., P. Dejonghe and J. P. Kruth (2003). "Optimal and collision free tool posture in five-axis machining through the tight integration of tool path generation and machine simulation." Computer-Aided Design **35**(5): 421-432.

Lavernhe, S., C. Tournier and C. Lartigue (2008). "Optimization of 5-axis high-speed machining using a surface based approach." Computer-Aided Design **40**(10-11): 1015-1023.

Lee, E. (2003). "Contour offset approach to spiral toolpath generation with constant scallop height." Computer-Aided Design **35**(6): 511-518.

Lee, Y.-S. (1997). "Admissible tool orientation control of gouging avoidance for 5-axis complex surface machining." Computer-Aided Design **29**(7): 507-521.

Lee, Y.-S. and T.-C. Chang (1996). "Automatic cutter selection for 5-axis sculptured surface machining." International Journal of Production Research **34**(4): 977 - 998.

Les, P. and W. Tiller (1997). The NURBS Book. Berlin, Springer-Verlag.

Li, H. and H. Y. Feng (2004). "Efficient five-axis machining of free-form surfaces with constant scallop height tool paths." International Journal of Production Research **42**(12): 2403-2417.

Li, H. Y. and Y. F. Zhang (2009). Automatic tool-path generation in 5-axis finish cut with multiple cutters. Proceedings of the 2009 IEEE international conference on Virtual Environments, Human-Computer Interfaces and Measurement Systems, Hong Kong, China, IEEE Press.

Li, L., Y. Zhang, H. Li and L. Geng (2010). "Generating tool-path with smooth posture change for five-axis sculptured surface machining based on cutter's accessibility map." The International Journal of Advanced Manufacturing Technology: 1-11.

Li, L. L. (2007). Process planning for five-axis milling of sculptured surfaces. Department of mechanical engineering, National University of Singapore.

Li, L. L. and Y. F. Zhang (2006). "Cutter selection for 5-axis milling of sculptured surfaces based on accessibility analysis." International Journal of Production Research **44**(16): 3303 - 3323.

Li, L. L. and Y. F. Zhang (2006). "An integrated approach towards process planning for 5-axis milling of sculptured surfaces based on cutter accessibility map." Computer Aided Design & Applications **1~4**(3): 249-258.

Li, S. X. and R. B. Jerard (1994). "5-axis machining of sculptured surfaces with a flat-end cutter." Computer-Aided Design **26**(3): 165-178.

Lim, T., J. Corney, J. M. Ritchie and D. E. R. Clark (2001). "Optimizing tool selection." International Journal of Production Research **39**(6): 1239-1256.

Makhanov, S. (2007). "Optimization and correction of the tool path of the five-axis milling machine: Part 1. Spatial optimization." Mathematics and Computers in Simulation **75**(5-6): 210-230.

Makhanov, S. (2007). "Optimization and correction of the tool path of the five-axis milling machine: Part 2: Rotations and setup." Mathematics and Computers in Simulation **75**(5-6): 231-250.

Makhanov, S. (2010). "Adaptable geometric patterns for five-axis machining: a survey." The International Journal of Advanced Manufacturing Technology **47**(9): 1167-1208.

Morishige, K., K. Kase and Y. Takeuchi (1997). "Collision-free tool path generation using 2-dimensional C-space for 5-axis control machining." The International Journal of Advanced Manufacturing Technology **13**(6): 393-400.

Morishige, K., Y. Takeuchi and K. Kase (1999). "Tool Path Generation Using C-Space for 5-Axis Control Machining." Journal of Manufacturing Science and Engineering **121**(1): 144-149.

My, C. A., E. L. J. Bohez, S. S. Makhanov, M. Munlinb, H. N. Phien and M. T. Tabucanon (2005). "On 5-Axis Freeform Surface Machining Optimization: Vector Field Clustering Approach." International Journal of CAD/CAM **5**(1): 1-14.

Park, S. C. and B. K. Choi (2000). "Tool-path planning for direction-parallel area milling." Computer-Aided Design **32**(1): 17-25.

Park, S. C. and B. K. Choi (2001). "Boundary extraction algorithm for cutting area detection." Computer-Aided Design **33**(8): 571-579.

Pi, J., E. Red and G. Jensen (1998). "Grind-free tool path generation for five-axis surface machining." Computer Integrated Manufacturing Systems **11**(4): 337-350.

Pottmann, H., J. Wallner, G. Glaeser and B. Ravani (1999). "Geometric criteria for gouge-free three-axis milling of sculptured surfaces." Journal of Mechanical Design, Transactions of the ASME **121**(2): 241-248.

Rao, A. and R. Sarma (2000). "On local gouging in five-axis sculptured surface machining using flat-end tools." Computer-Aided Design **32**(7): 409-420.

Rao, N., F. Ismail and S. Bedi (2000). "Integrated tool positioning and tool path planning for five-axis machining of sculptured surfaces." International Journal of Production Research **38**(12): 2709-2724.

Roth, D., S. Bedi, F. Ismail and S. Mann (2001). "Surface swept by a toroidal cutter during 5-axis machining." Computer-Aided Design **33**(1): 57-63.

Sarma, R. and Dutta, D. (1997).). "The geometry and generation of NC tool paths." Journal of Mechanical Design **119**(2): 253-258.

Scherrer, P. K. and B. M. Hillberry (1978). "Determining distance to a surface represented in piecewise fashion with surface patches." Computer-Aided Design **10**(5): 320-324.

Senatore, J., F. Monies, J.-M. Redonnet and W. Rubio (2005). "Analysis of improved positioning in five-axis ruled surface milling using envelope surface." Computer-Aided Design **37**(10): 989-998.

She, C.-H. and C.-C. Chang (2007). "Design of a generic five-axis postprocessor based on generalized kinematics model of machine tool." International Journal of Machine Tools and Manufacture **47**(3-4): 537-545.

She, C.-H. and R.-S. Lee (2000). "A Postprocessor Based on the Kinematics Model for General Five-Axis Machine Tools." Journal of Manufacturing Processes **2**(2): 131-141.

Sheltami, K., S. Bedi and F. Ismail (1998). "Swept volumes of toroidal cutters using generating curves." International Journal of Machine Tools and Manufacture **38**(7): 855-870.

Stanislav, M. (2007). "Optimization and correction of the tool path of the five-axis milling machine: Part 1. Spatial optimization." Mathematics and Computers in Simulation **75**(5–6): 210-230.

Tournier, C. and E. Duc (2002). "A Surface Based Approach for Constant Scallop HeightTool-Path Generation." The International Journal of Advanced Manufacturing Technology **19**(5): 318-324.

Tutunea-Fatan, O. R. and H.-Y. Feng (2004). "Configuration analysis of five-axis machine tools using a generic kinematic model." International Journal of Machine Tools and Manufacture **44**(11): 1235-1243.

Veltkamp, R. C. (1994). Closed Object Boundaries from Scattered Points, Springer-Verlag.

Wang, N. and K. Tang (2007). "Automatic generation of gouge-free and angular-velocity-compliant five-axis toolpath." Computer-Aided Design **39**(10): 841-852.

Wang, N. and K. Tang (2008). "Five-axis tool path generation for a flat-end tool based on iso-conic partitioning." Computer-Aided Design **40**(12): 1067-1079.

Wang, Q. H., J. R. Li and H. Q. Gong (2007). "Graphics-assisted cutter orientation correction for collision-free five-axis machining." International Journal of Production Research **45**(13): 2875-2894.

Wang, W. P. and K. K. Wang (1986). "Geometric Modeling for Swept Volume of Moving Solids." Computer Graphics and Applications, IEEE **6**(12): 8-17.

Wang, X. C. and Y. Yu (2002). "An approach to interference-free cutter position for five-axis free-form surface side finishing milling." Journal of Materials Processing Technology **123**(2): 191-196.

Warkentin, A., F. Ismail and S. Bedi (2000). "Comparison between multi-point and other 5-axis tool positioning strategies." International Journal of Machine Tools and Manufacture **40**(2): 185-208.

Weinert, K., S. Du, P. Damm and M. Stautner (2004). "Swept volume generation for the simulation of machining processes." International Journal of Machine Tools and Manufacture **44**(6): 617-628.

Xu, X. J., C. Bradley, Y. F. Zhang, H. T. Loh and Y. S. Wong (2002). "Tool-path generation for five-axis machining of free-form surfaces based on accessibility analysis." International Journal of Production Research **40**(14): 3253-3274.

Yang, D. C. H., J. J. Chuang and T. H. OuLee (2003). "Boundary-conformed toolpath generation for trimmed free-form surfaces." Computer-Aided Design **35**(2): 127-139.

Yang, D. C. H. and Z. Han (1999). "Interference detection and optimal tool selection in 3-axis NC machining of free-form surfaces." Computer-Aided Design **31**(5): 303-315.

Yoon, J.-H. (2003). "Tool tip gouging avoidance and optimal tool positioning for 5-axis sculptured surface machining." International Journal of Production Research **41**(10): 2125-2142.

Yoon, J.-H., H. Pottmann and Y.-S. Lee (2003). "Locally optimal cutting positions for 5-axis sculptured surface machining." Computer-Aided Design **35**(1): 69-81.

You, C.-F., B.-T. Sheen and T.-K. Lin (2007). "Selecting optimal tools for arbitrarily shaped pockets." The International Journal of Advanced Manufacturing Technology **32**(1): 132-138.

Yuan-Shin, L. (1997). "Admissible tool orientation control of gouging avoidance for 5-axis complex surface machining." Computer-Aided Design **29**(7): 507-521.

Zhong, Y., J. Zhou and T. Chen (2002). "Determination of Cutter Orientation for Five-Axis Sculptured Surface Machining with a Filleted-End Cutter." The International Journal of Advanced Manufacturing Technology **20**(10): 735-740.

Zhu, L., G. Zheng, H. Ding and Y. Xiong (2010). "Global optimization of tool path for five-axis flank milling with a conical cutter." Computer-Aided Design **42**(10): 903-910.