Computing axes of rotation for 4-axis CNC milling machine by calculating global visibility map from slice geometry

by

Guangyu Hou

A thesis submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Major: Industrial Engineering

Program of Study Committee:
Matthew Frank, Major Professor
Frank Peters
James Oliver

Iowa State University

Ames, Iowa

2015

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my advisor, Dr. Matthew Frank, for his excellent and consistent guidance, understanding, patience, and providing me with a wonderful atmosphere for doing research. I would like to thank my committee members, Frank Peters and James Oliver, for their guidance and support throughout the course of this research.

In addition, I would like to thank my senior colleagues Shuangyan Lei, Ashish Joshi, Prashant Barnawal and all other colleagues and friends in the RMPL lab for their kindly help in the development of this thesis. I would also like to thank the department faculty and staff for making my time at Iowa State University a wonderful experience.

Finally, thanks to my parents and brother for their consistent encouragement and to my wife Mei Huang for her hours of patience, respect, and love.

ABSTRACT

This thesis presents a new method to compute a global visibility map (GVM) in order to determine feasible axes of rotation for 4-axis CNC machining. The choice of the 4th-axis is very important because it directly determines the critical manufacturing components; visibility, accessibility and machinability of the part. As opposed to the considerable work in GVM computation, this thesis proposes an innovative approximation approach to compute GVM by utilizing slice geometry. One advantage of the method is that it is feature-free, thus avoiding feature extraction and identification. In addition, the method is computationally efficient, and can be easily parallelized in order to vastly increase speed. In this thesis, we further present a full implementation of the approach as a critical function in an automated process planning system for rapid prototyping.

CHAPTER 1

INTRODUCTION

Workpiece setup planning is one important part of process planning in CNC

machining as it is directly related to a part's manufacturability and resulting quality of the

machined surface [1]. In a 4-axis CNC mill machine, optimization in setup planning includes

optimizing the axis of rotation, a set of rotation angles, support positions, etc. in order to

properly machine the part. Among all these setups, axis of rotation is the first to be

determined and has a significant impact on the rest. Although general methods for visibility

mapping and some setup planning exists, there is little work in the specific problem of axis of

rotation selection and time efficiency is a major challenge for the previous approaches [1, 2].

1.1 CNC Milling Machines

Milling is a machining process using rotary cutters to remove material from a

workpiece, advancing (or feeding) in a direction with the axis of the tool[3-5] (Figure 1a).

The capability of machining parts to precise sizes and shapes and creating a large variety of

features has made milling one of the most commonly used processes in industry. Computer

Numerical Control (CNC), with its advent in the 1950s, has evolved milling machine to

machining *centers*, making the milling process a much more automated and precise tool.

Mill configurations

Mill orientation is the primary classification for milling machines. The two basic

configurations are vertical and horizontal. Based on the motion of the spindle, vertical mills

divide into two subcategories: turret mill and bed mill (Figure 1b and c). In this thesis, we

focus on the bed vertical mill (Which means the spindle moves parallel to its axis (Figure 1c)

compared to a stationary spindle (Figure 1b).

(a)   (b)   (c)
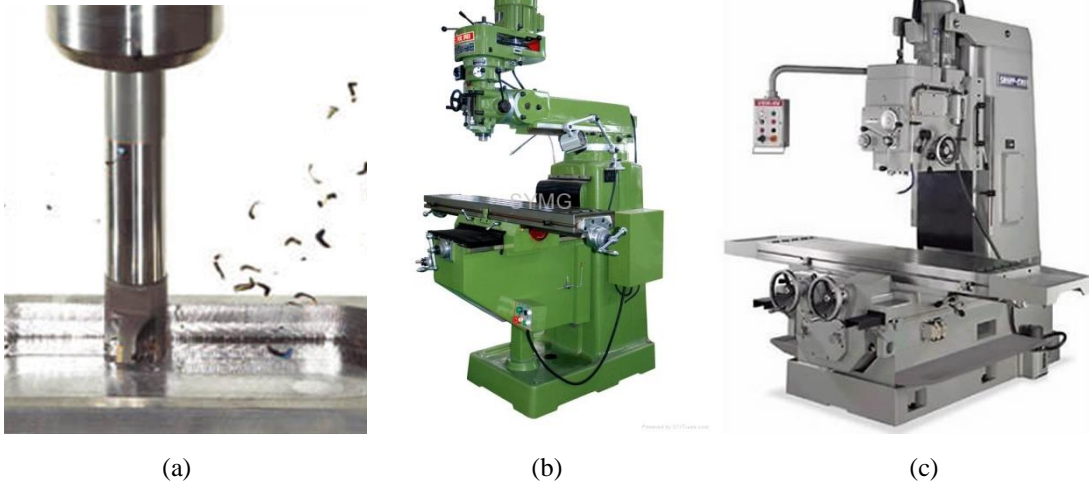
**Figure 1 – Milling process examples; (a) End milling tool[6] (b) Turret Vertical Mill[7] (c)
Bed Vertical Mill[8]**

Multi-axis CNC milling machines

A very basic vertical mill is typically in a 3-axis configuration (Figure 1b and c). For

example, the bed vertical mill has an X-Y table and the vertical moving spindle provides Z-

axis motion. The limitation of a 3-axis mill is that the tool has limited access to the part thus

cannot machine a part with complex geometry (e.g. undercuts) in one operation. Multiaxis

CNC machines are machines that support not only translation in 3 axes, but also rotation

around one or multiple axes, deemed the 4th-axis and 5th-axis (Figure 2) , where a 5th axis is

generally considered the maximum degree of freedom for milling.

Figure 2 – Axis configurations; (a) 4-axis CNC mill[9] (b) 5-axis CNC mill[10]

Four-axis CNC milling machines

The 4-axis CNC vertical milling machine has one rotary axis in addition to the standard three linear axes (x, y and z axis) as shown in Figure 2a. The 4th axis can be controlled simultaneously to move the tool in complex paths, but it can also be used to simply rotate the component between setups. This latter approach is more desirable than re-fixturing the component manually using a 3 axis CNC mill, which introduces re-fixture errors, requires a proper fixture posture and needs extra setup time and human intervention.
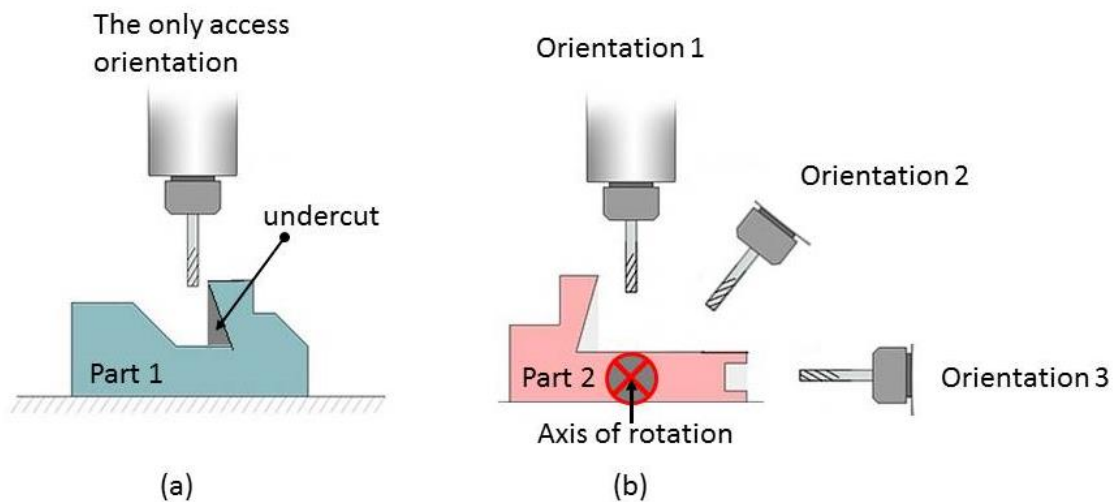
Figure 3 – Milling undercut problem; (a) 3-axis mill (b) 4-axis mill [11]

For a given part, the choice of axis to orient the part is critical, in that it directly determines whether the part is machinable and further affects the rotation (setup) angles of the machining operations about that axis.
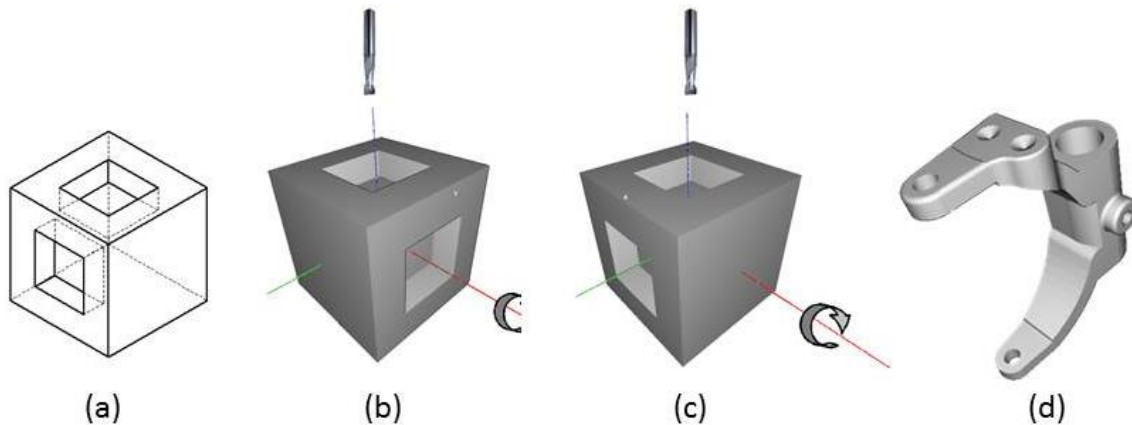
Figure 4 – The choice of axis of rotation; (a) Cube with two square pocket (b) axis of rotation 1 (c) axis of rotation 2 (d) industrial bracket

As shown in Figure 4, a cube with square pockets on two of its faces has two optional axes of rotation (Figure 4b and 4c). As the machine tool is perpendicular to the axis of rotation, it is easy to see that the pocket in Figure 4b is an undercut whereas the same pocket oriented in Figure 4c is fully machinable. The cube example is simple in that the cube and all pockets are aligned with the principle axes. However, when parts become more complex (Figure 4d) the choice is not obvious and a systematic solution to find the proper axis of rotation is in need.

1.2 Visibility in 4-axis CNC Milling

In order to solve the axis selection problem, visibility is introduced. In general, if a surface patch on a component is visible from some orientation, it means that a line of light along that orientation can reach the surface patch without obstacle. The complete visibility of

that surface patch can then be defined as *all* the orientations that make the surface patch

visible. In 3-dimensional space, we define such a visibility region as visibility cone. The

space where the visibility cone resides is defined as visibility sphere. In 2-dimensional

scenarios, we define such visibility regions as visibility arcs (Figure 5).

(a)   (b)
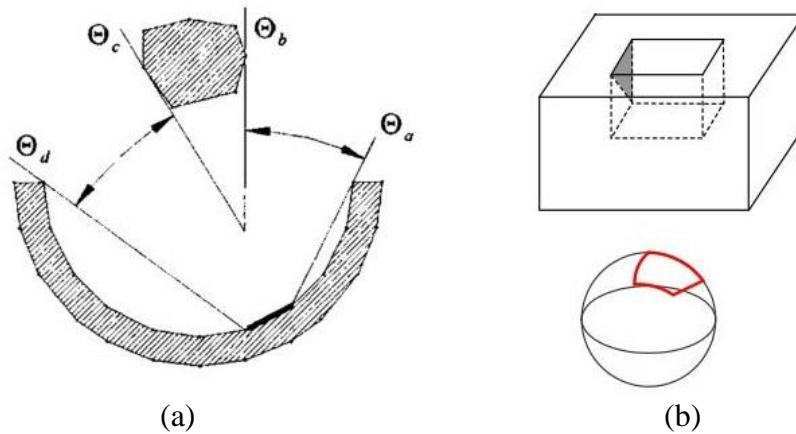
Figure 5 – Visibility used in 2D and 3D; (a) 2D visibility arcs[12] and (b) 3D visibility cone [13]

Using the visibility cone of each surface patch of a part, the solution of visibility in

CNC setup planning is to find a viewpoint or a set of viewpoints from which the entire set of

part surface are visible. These viewpoints/orientations give us the feasible postures of the

component with respect to the machine tool. Furthermore, they tell us what potential axes of

rotation might be feasible for a given part.

It is necessary to note that, in practice, the tool may not be able to use the exact

orientation given by the visibility solution because the tool has a diameter, whose

accessibility cannot be treated as "line of sight". Therefore, the real accessibility of a tool is

more restricted and will be defined as machinability instead. The visibility cone is not

sufficient, but necessary in that it gives a tight upper bound of a region of feasible

orientations therefore greatly reduces undesirable orientations from consideration.

1.3 Research Objectives

The overarching objective of this thesis is to develop a method that determines feasible axes of rotation for parts to be machined using a 4-axis CNC mill. The work is divided into two major sub-objectives. The first sub-objective is to compute a Global Visibility Map (GVM), while the second sub-objective is to compute axes of rotation from the GVM. The following chapter will provide a review of the relevant literature and then followed by the proposed new methods.   Finally, we will present an implementation and testing of the method and application within CAD/CAM software for a rapid machining system.

CHAPTER 2

LITERATURE REVIEW

2.1 Overview

The main concern of setup planning is providing accessibility for the cutting tool. Since visibility is a necessary condition for accessibility, it is broadly used by researchers in setup planning.

The visibility problem is called the *visible set problem* in some articles [14]. It is defined as: given an entity (surface patch) in the scene, find the set of view directions along which the given entity is completely visible. As defined by previous work [15, 16], the mapping of the visible directions on the unit sphere is a spherical image called the global visibility map (GVM).

There have been many articles addressing the problem of visibility, but few articles address the axis of rotation problem directly. Among the many visibility approaches, a widely known method to construct a visibility map is using Gaussian mapping [15]. Many researchers utilized Gaussian maps to compute setup orientations for 4- and 5- axis machining [16-18]. However, visibility obtained from a Gaussian map ignores the fact that other surfaces of the object may occlude visibility of the target facet[14]. Thus, it is a *local* visibility which cannot be used directly to determine axis of rotation for 4-axis CNC setups. Suh and Kang[2] discretized the visibility sphere into spherical triangles and used an occupation test to obtain the visibility cone. Although it obtained a global visibility, such a method has disadvantages in that; 1) it uses the centroid of the part surface triangle to approximate the triangle, resulting in an approximated visibility, and 2) The occupation test

leads to a $n \times (n-1) \times m$ time complexity in computation thus is very inefficient. In this case, n is the number of surface patches and m is the size of the approximated visibility sphere. Li and Frank[1] later introduce an approach that computes visibility by an occlusion calculation between a pair of polyhedral facets. This method provides global visibility and is accurate but still has a time complexity of $O(n^2)$; n being the number of facets.

These visibility approaches, along with several others are categorized and described in detail below. Among them, the common challenges are accuracy, time complexity, and local vs. global visibility.

2.2 Gaussian Mapping

Gaussian Mapping is the first approach introduced to solve the visible set problem. One early work used Gaussian mapping to calculate the visibility map of Bezier surfaces[19]. This work was further extended by [20] to include visibility calculation for free-form surfaces. The basic idea to obtain a visibility map in these works is to compute the dual image of the Gaussian map on a unit sphere. However, these works only consider the surface visibility locally, ignoring that the visibility cone of a designated entity might be occluded by other surfaces. Thus applications are limited to the visibility of certain features of the component [17, 21].

2.3 Approximation Methods

2.3.1 Hidden surface removal

Hidden surface removal methods often need to discretize the component surface into discrete entities (like triangle meshes) and then establishes a discrete visibility space. It builds a visibility mapping from the discrete entity to the discrete visibility viewing sphere;

such mapping is finite. The discretion resolution impacts visibility accuracy. One way to obtain such discrete visibility is to use ray-casting techniques [22], while other related work utilizes a z-buffer method through graphics hardware[23-25]. The disadvantages of approximation methods are obvious – approximation error. The true visibility cone might turn to be larger than obtained considering some viewing points are not in the sampled region.

2.3.2 Slice geometry

Slice based methods have been developed to compute visibility, especially for 4-axis CNC mills. They address the visible set problem indirectly by taking advantage of the slice geometry. In the approach of [12] they divide the component into a set of 2 dimensional slices along one direction (phase1); compute the 2D visibility of each slice (phase2), and then combine them to construct visibility for the axis. The advantage is that its experiment shows processing time in visibility computation (phase2) is linear to the number of facets. Therefore, it is a very efficient algorithm. One of the drawbacks is that the slicing process (phase1) has time complexity of $O(n^2 + k^2)$ for model with n triangles and k slicing planes. Another drawback of this approach is that the final visibility corresponds to a certain axis of rotation instead of a GVM.

However, an asymptotically optimal slicing algorithm has been developed recently that has time complexity of $O(n\bar{k})$, where $\bar{k}$ is the average number of slices cutting each triangle, which is asymptotically the best that can be achieved under certain common assumptions[26].

Accordingly, phase 1 in the original method can be dramatically improved using such asymptotically optimal slicing algorithm.

2.4 Exact Solution Methods

There are also a few approaches that aim at obtaining an exact solution for a global visibility map[13, 27-30]. [29] proposed an algorithm that calculated global a visibility map for a triangular mesh by an occlusion calculation between a pair of triangle facets. [13] extends the work to the occlusion calculation between a pair of convex facets. A similar approach also calculates the occlusion between a pair of convex facets by a boundary tracing method [30]. However these approaches share a common bottleneck: due to a pairwise occlusion computing, the algorithm's time efficiency is largely limited by the size of the mesh. In addition, the union operation among all spherical polygons before generating the GVM is also an expensive computing task. To improve efficiency in the latter problem, [14] has further extended their previous work by introducing a Minkowski sum.

2.5 Mapping from GVM to Axes of Rotation

Methods to apply GVM to 4-axis CNC machining setup planned have also been studied. Li and Frank have established a relation from point visibility, arc visibility and cone visibility to axes of rotation, thus forming a mapping from GVM to axes of rotation[1]. The time complexity of such mapping is $O(n)$ for an n-facets model.

In summary, Gaussian mapping alone is not sufficient for a GVM. Considering previous approaches, there is always a tradeoff between accuracy and time efficiency (either

approximated or exact methods). There is still space for improvement of these approaches, and, considering that time efficiency is a major concern in CNC setup planning, an approximated approach with good time efficiency may be a reasonable choice.

CHAPTER 3

SOLUTION METHOD

3.1 General Overview of the Method

To introduce, the method is decomposed into two main phases based on their precedence; computing global visibility and then computing axes of rotation. These two main phases are further decomposed into several sequential subroutines based on their functionality.
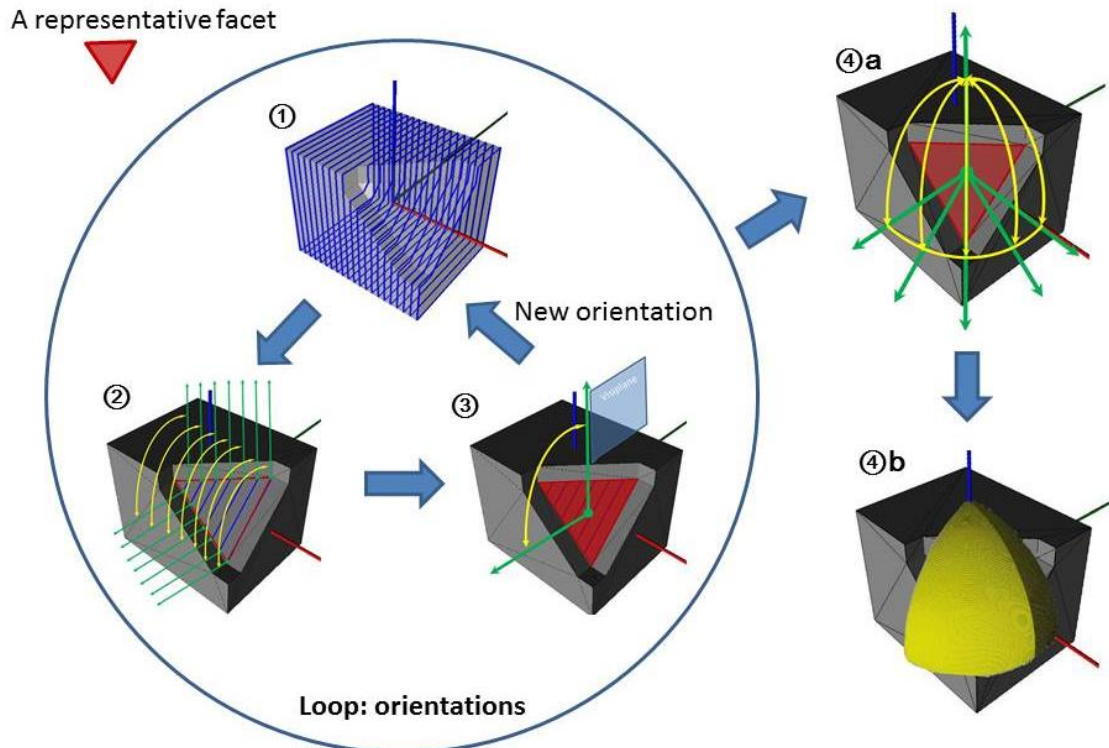
Figure 6 – Phase 1 global visibility computation flow; 1) model Slicing, 2) segment visibility computation, 3) conversion to facet visibility and obtainment of a 2D visibility map in one orientation, 4a) union of 2D visibility maps of *all orientations*, 4b) obtainment of GVM, the union of *densely spaced* 2D visibility maps (angle between two maps = 1°).

Phase I Global visibility computation

1. Slicing the model.

2. 2D visibility computation.

3. Intersection of segments visibility for facet visibility.

4. Union of 2D visibility maps of all orientations into a global visibility map.

Phase II Axis of rotation computation

1. Computing corresponding axis region of each facet.

2. Obtain final axes by intersecting the axis regions of all facets.

An example part is used in Figure 6 and Figure 7 to show the flow of the two main phases respectively, while Figure 8 illustrates the complete process in the form of a flow chart.
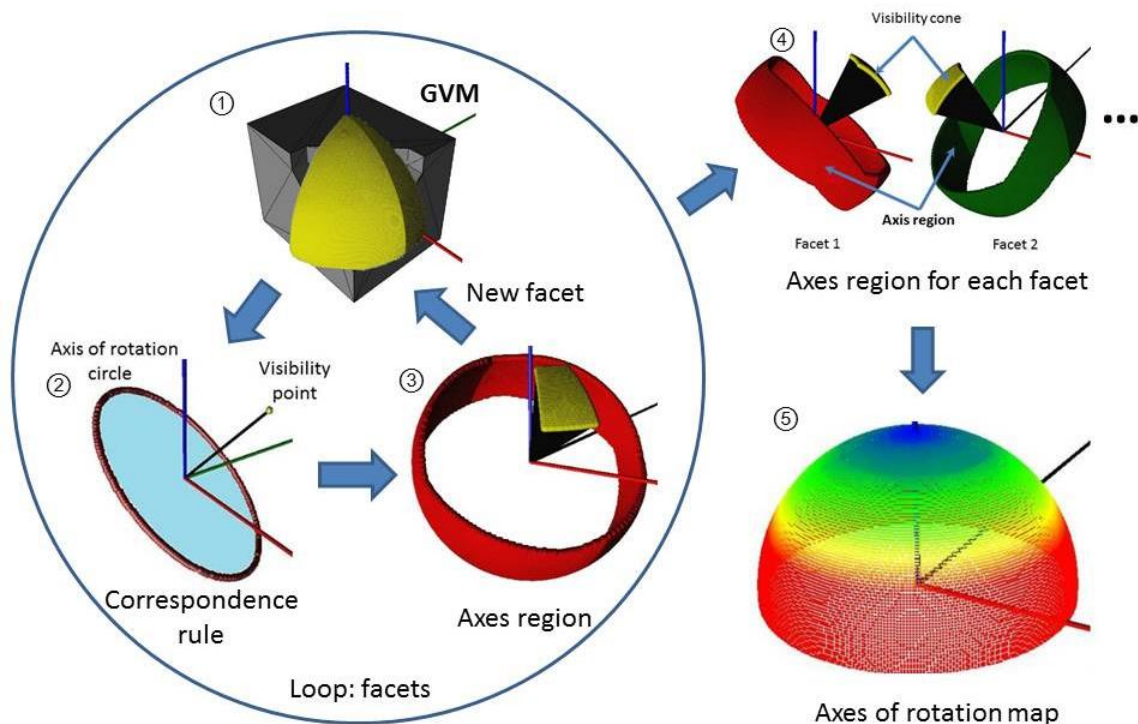
Figure 7 – Phase 2 axis of rotation computation flow; 1) an existing GVM of a facet, 2) the correspondence rule of point visibility to axes of rotation, 3) obtainment of axes of rotation using the correspondence rule, 4) axes regions for each facet, 5) the axis of rotation map.
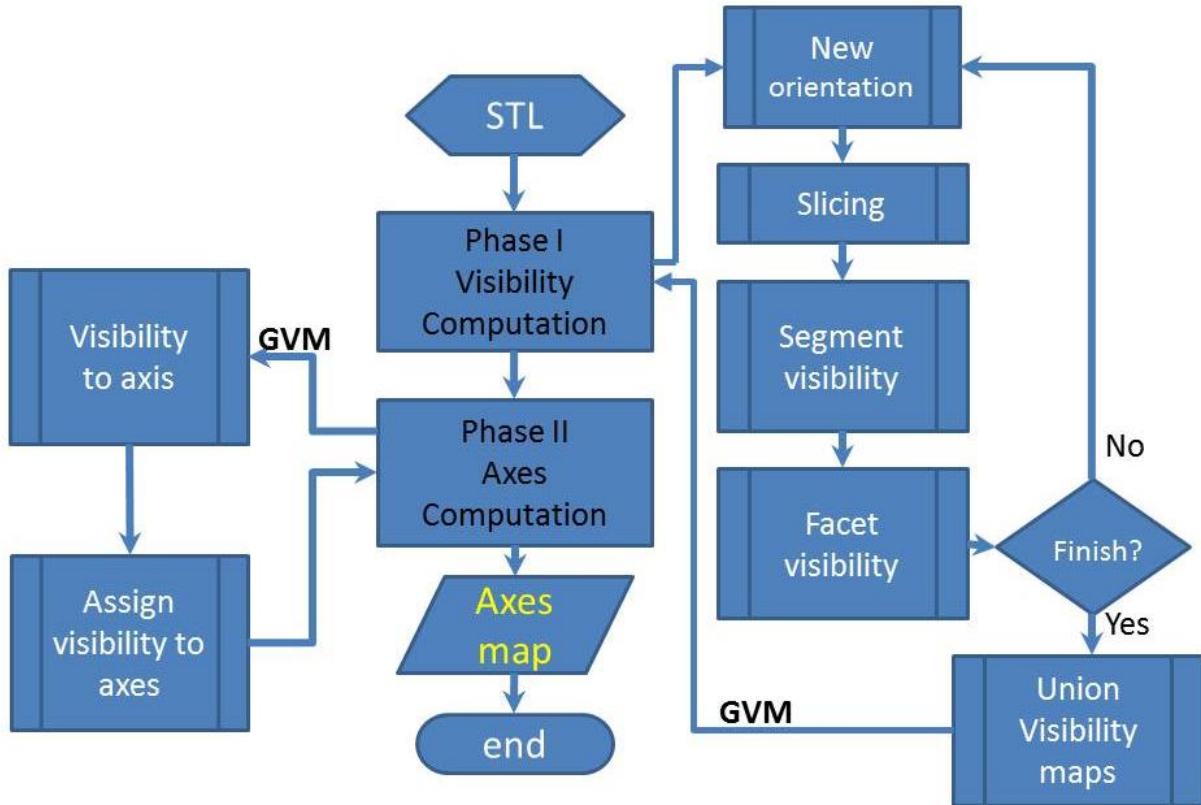
Figure 8 – Flowchart of all subroutines

3.2 Computing the Global Visibility Map

The creation of a Global Visibility Map (GVM) is the objective of phase 1. A GVM

is a spherical image created to describe the complete set of global visible view directions for

a surface [14]. The GVM of any surface patch is in the form of a spherical polygon. Under

spherical coordinate with unit radius, a GVM has a freedom of 2.

Limitations using segment visibility

The method in [12] provides segments' visibility about a given slice orientation. If

they are presented on a sphere, they form arcs of a fixed great circle. Accordingly, they have

a freedom of 1 under spherical coordinates with unit radius. Considering that a GVM has a

freedom of 2, such visibility actually only considers a portion of the GVM.   If a GVM is to

be derived from such visibility, the only option is to combine multiple such visibilities from different slice orientations. The hope would be that the original arcs combine to form a sufficiently dense portion of a spherical polygon. However, segment visibility is rather "exclusive" due to slicing. The segments resulting from two different slicing orientations are irrelevant to each other and cannot be matched. Therefore, visibility in two settings, having no common reference, cannot be combined. This characteristic makes segment visibility "non-incremental" (or unable to accumulate) among slicing orientations (Figure 9).
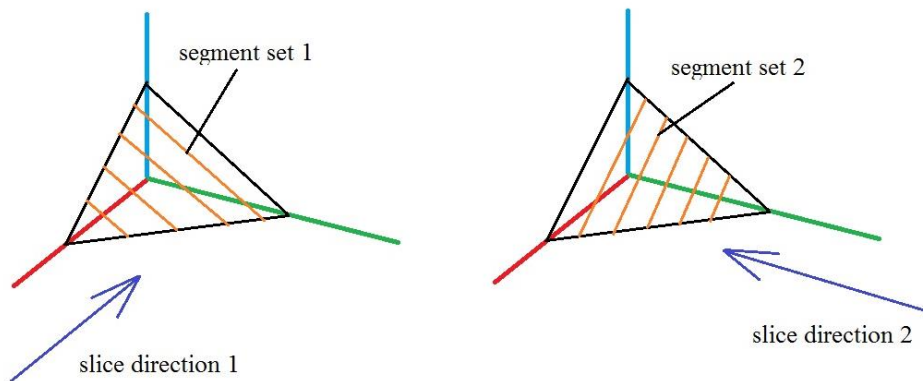
Figure 9 – Two different slice orientations and resulting segments of the same facet

Solution

The key to constructing a GVM from a set of "exclusive" visibility maps is to convert the segment visibility to facet visibility. This is because the facet, unlike segments, remains intact during multiple slicing settings. Therefore, the gradually formed visibility map can always refer to some particular facet. Note that by "converting", the method will still use segment visibility, but as an intermediate subroutine.

3.2.1 Computing segments' visibility

A method is used to compute visibility to the surface of a model that is rotated about a fourth axis[12] assuming a proposed axis of rotation is given. Since the tool access is restricted to directions orthogonal to the rotation axis, 2D visibility maps for a set of cross sections of the surface of the model can be used together to approximate visibility to the entire surface of the model. The method divides a 3D problem into 2D problems and conquers each of them separately (Figure 10).
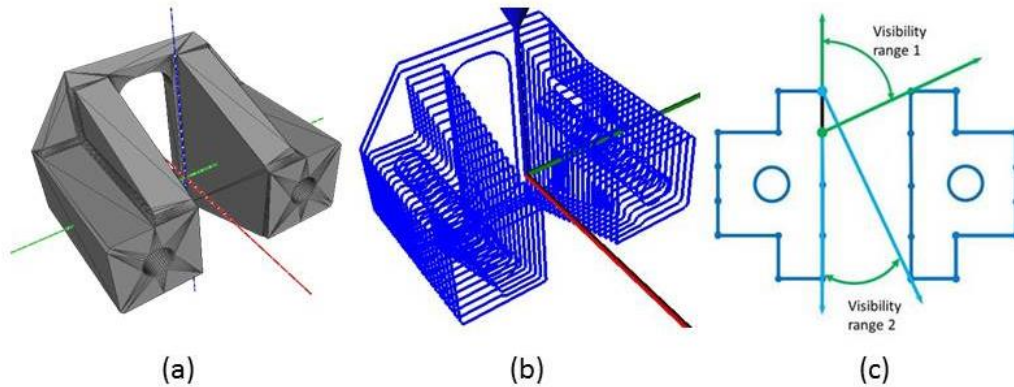
Figure 10 – Slice geometry; (a) Original model (b) slices (c) visibility calculation of a segment

For each segment, this visibility algorithm generates a collection of visible ranges given in polar angle about the axis of rotation:

$$
\begin{array}{l}
VIS_{1,1,1} \\
VIS_{2,1,1} \\
\vdots \\
VIS_{ijk} \\
\vdots \\
VIS_{qnp}
\end{array}
\left(
\begin{array}{cccc}
(\Theta_a,\Theta_b)_1, (\Theta_a,\Theta_b)_2 \ldots (\Theta_a,\Theta_b)_n \\
(\Theta_a,\Theta_b)_1, (\Theta_a,\Theta_b)_2 \ldots (\Theta_a,\Theta_b)_n \\
\vdots \quad \vdots \quad \vdots \\
(\Theta_a,\Theta_b)_1, (\Theta_a,\Theta_b)_2 \ldots (\Theta_a,\Theta_b)_n \\
\vdots \quad \vdots \quad \vdots \\
(\Theta_a,\Theta_b)_1, (\Theta_a,\Theta_b)_2 \ldots (\Theta_a,\Theta_b)_n
\end{array}
\right)
$$

Figure 11 – Visibility of segments [12]

Where VIS_{tjk} denotes the visibility collection of the tth segment of the jth contour of the kth slice. $(\Theta_a, \Theta_b)_i$ denotes one of its visibility ranges.

3.2.2 Computing facets' visibility

In a certain slicing setting, the visibility of a facet is calculated as the intersection of visibilities of all segments contained in that particular facet (Figure 13). To facilitate an error estimation which will be introduced later, all approximations are discussed in detail next.

Approximation One

The segment visibility is an average visibility over Δx considering the unneglectable slicing interval Δx, or the *finite* number of segments (Figure 12). The ideal situation is that each facet is represented by an infinite number of segments. This approximation inherits from the method in [4].
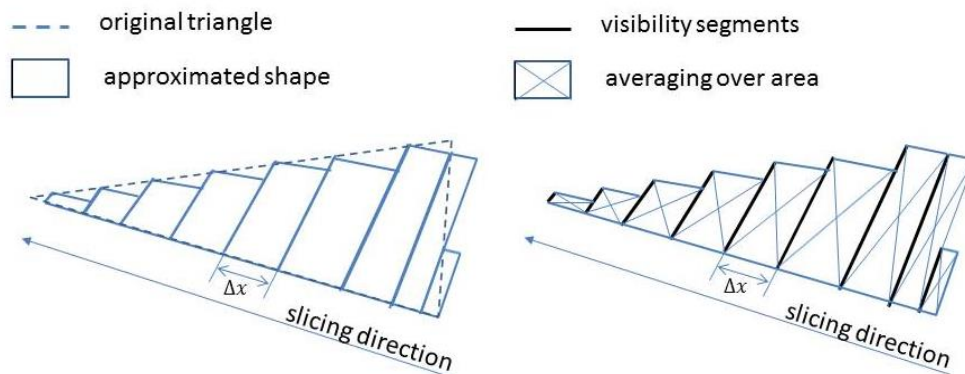
Figure 12 – Representing a facet by a series of segments from the perspective of visibility

Therefore, the visibility of the original segment reflects visibility over an area (called averaging over area) defined by the slicing interval Δx and segment length. The bounds of the visibility range of an area are two bands (Figure 13).
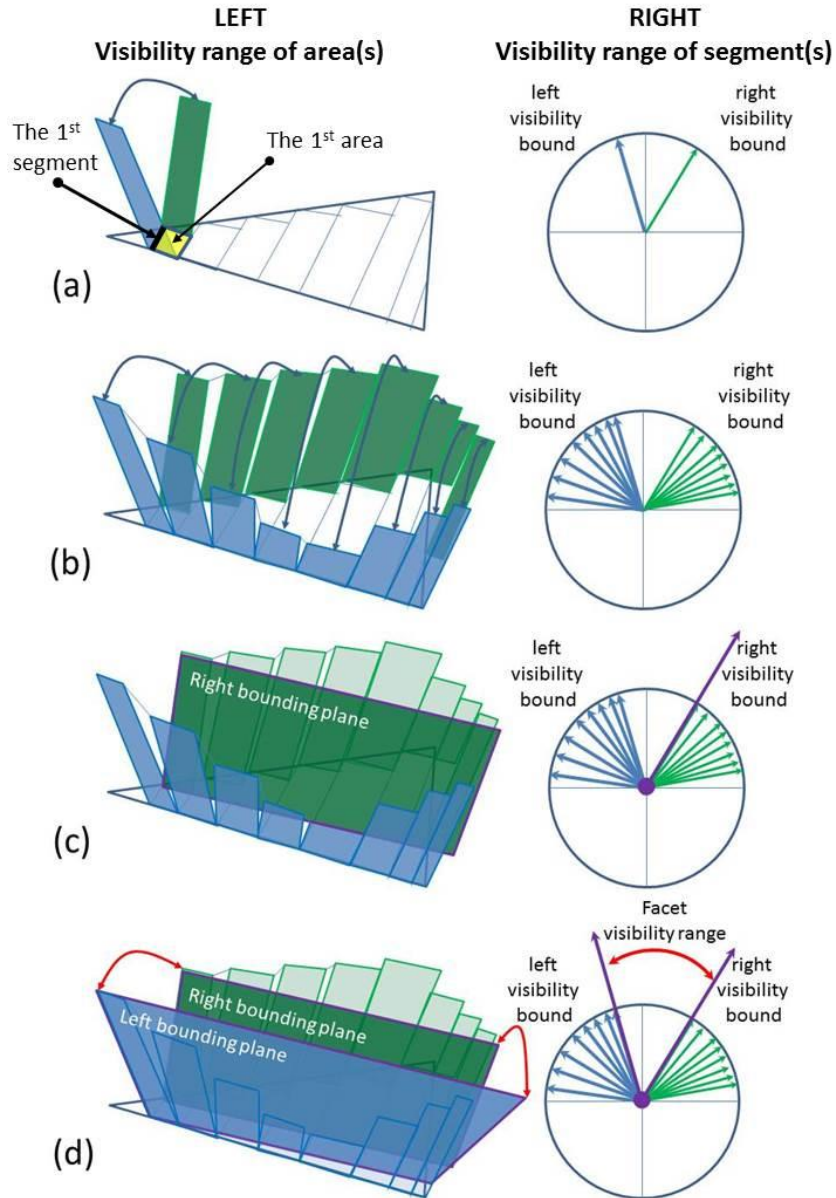
Figure 13 – Facet visibility obtained by <u>intersecting</u> segments' visibility; (a) **LEFT: the visibility range of the 1st area, RIGHT: the visibility range of the 1st segment, (b) LEFT: the visibility ranges of all areas, RIGHT: the visibility ranges of all segments; (c) LEFT&RIGHT: the right bound of the facet visibility (d) LEFT&RIGHT: the left bound of the facet visibility. The two visibility bounds of the facet define its range.**

<u>Approximation Two</u>

It is a compromise of visibility to treat a facet as an intersection of the segments it

contains. An example is that a facet is deemed invisible when some fraction of its containing

segments are still visible. In such case, visibility is not adequately recovered. Such

disadvantage comes from the nature of approximation – it has error.

Error Estimation and Reduction

This work proposes a way to improve approximation and reduce error. All that is

needed is a way of estimating the specific error and a direction of improvement.

Visibility volume

As for error estimation, a concept of visibility volume is introduced (Figure 14). We

define the visibility volume of segment j as:

$$VVS_j = \frac{1}{2} \cdot \Delta\theta \cdot 1^2 \cdot \Delta x = \frac{1}{2}\Delta\theta\Delta x$$

where $\Delta\theta$ is the range of visibility of segment j and Δx is the slicing interval for segment j.

And the visibility volume of facet $\boldsymbol{k}$ is defined as:

$$VVF_k = \frac{1}{2} \cdot \Delta\theta_{intersecting} \cdot \Delta L$$

Where $\Delta\theta_{intersecting}$ is the visibility range of facet $\boldsymbol{k}$ and ΔL is the length of facet $\boldsymbol{k}$ in the

slicing direction.

(a) Visibility range of an Area A/segment A   (b) Visibility volume of Area A/segment A
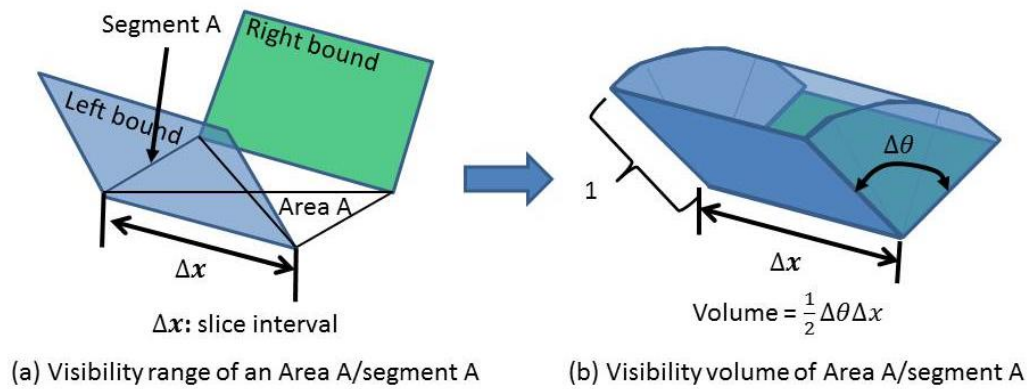
Figure 14 – Concept of visibility volume by an example; the visibility volume is decided by a segment's visibility range ($\Delta\theta$) and slice interval (Δx)

Converting quality indicator

The difference in visibility volume between a facet and the segments it contains

indicates the visibility loss due to converting, thus can serve as a converting quality indicator

(Figure 15). We define converting quality of facet k to be:

$$q_k = \frac{VVF_k}{\sum_1^{n_k} VVS_j} = \frac{\Delta\theta_{intersecting} \cdot \Delta L}{\sum_1^{n_k} \Delta\theta_j \Delta x_j}$$

Where n_k is the number of segments that facet k contains. The overall converting quality of

the model is calculated as:

$$q_{overall} = \frac{\sum_1^n q_k \cdot A_k}{\sum_1^n A_k}$$

Where A_k is the area of facet k.

Visibility volume
Of segment 1=VVS_1

Visibility volume
Of segment 2 =VVS_2

Visibility volume
Of segment 3 =VVS_3

(a) Sum of all segments' visibility volumes = $\sum_1^3 VVS_j$

$$q_f = \frac{VVF}{\sum_1^3 VVS_j}$$

(b) Visibility volume of the facet = VVF
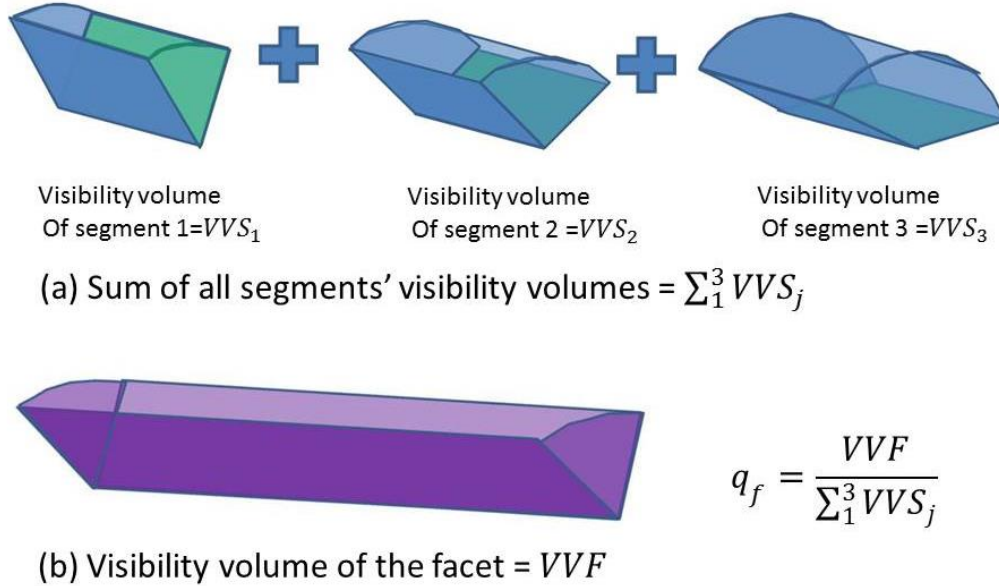
Figure 15 – An example of the calculation of converting quality of a facet, the example facet has 3 segments; (a) The sum of all 3 segments' visibility volumes, (b) visibility volume of the facet

Error-reduction

As for error-reduction, a mesh refinement is implemented. That way, the sum of the

visibility volumes of the smaller facets is at least as much as the big facet containing them,

and with a chance to be improved (Figure 16). In the implementation section, a tolerance

TOL_c is preset and visibility volume loss is calculated for every facet, as long as $q_{overall} \leq$

$1 - TOL_c$, a mesh refinement is implemented to increase $q_{overall}$.

Once $q_{overall} > 1 - TOL_c$, then visibility loss is within the acceptable range and it

is concluded that the facet visibility is obtained with negligible loss.
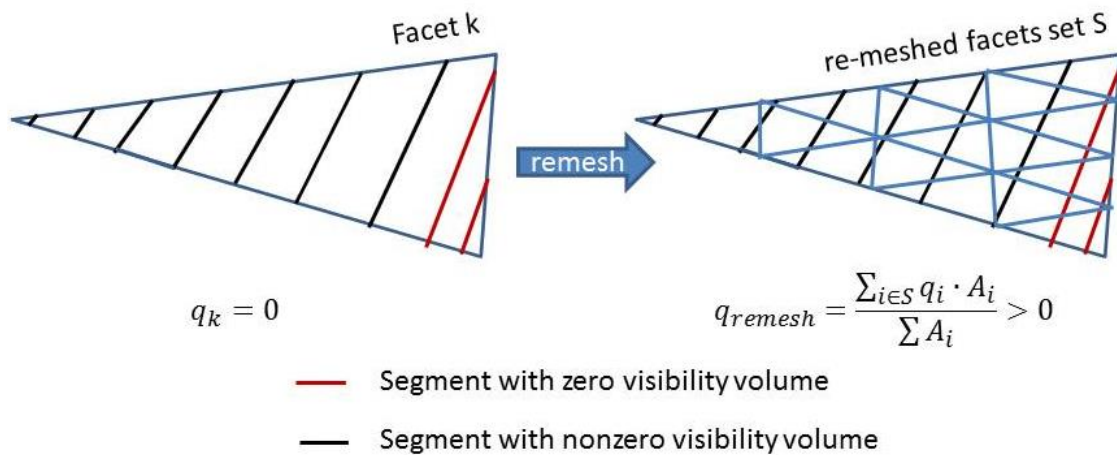
Figure 16 – Mesh refinement to improve visibility converting quality

3.2.3 Union of visibility maps

A spherical polygon is a closed geometric figure on the surface of a sphere formed by

the arcs of great circles (Figure 17). In spherical geometry, geodesics are arcs of great circles.

(a) spherical triangle   (b) spherical rectangle
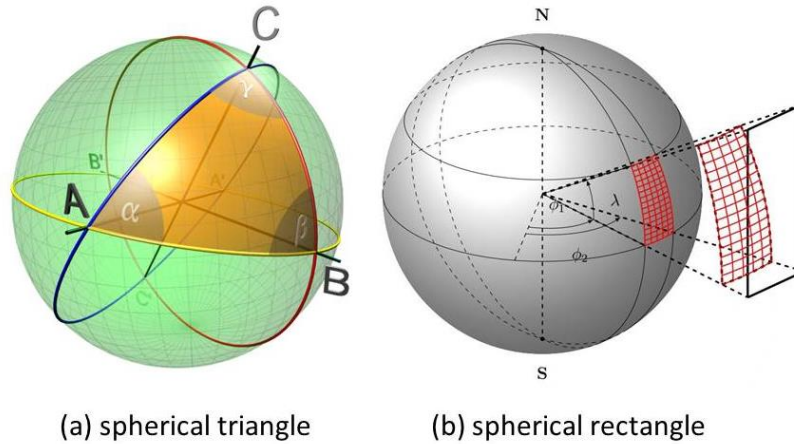
Figure 17 – An example of spherical polygon [31] and [32]

As facet visibility using slice geometry is in the form of geodesics and GVM is in the form of spherical polygons, constructing GVM from the former visibility is possible.

If sliced only in one orientation, the facet will still have visibility in the form of arcs on a fixed great circle, namely, the 2D visibility map (Figure 18). To obtain the facet's GVM, it is required to compute all its 2D visibility maps (in various slicing orientations) and union all of them.
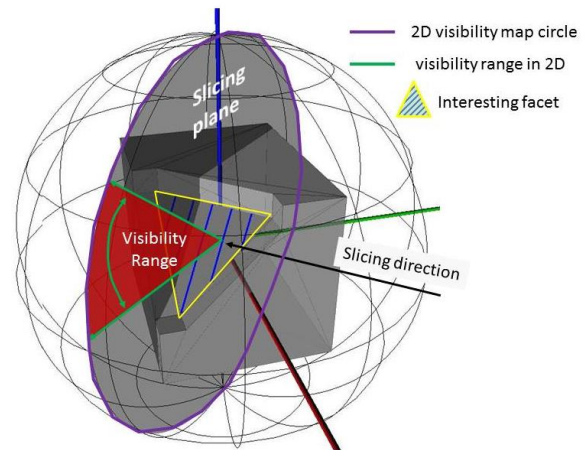
Figure 18 – 2D visibility map of a facet

Geometrically speaking, it is equal to the union of all geodesics into a spherical polygon. An example is given in Figure 19 to illustrate the procedure of obtaining a GVM by a union of 2D visibility maps.
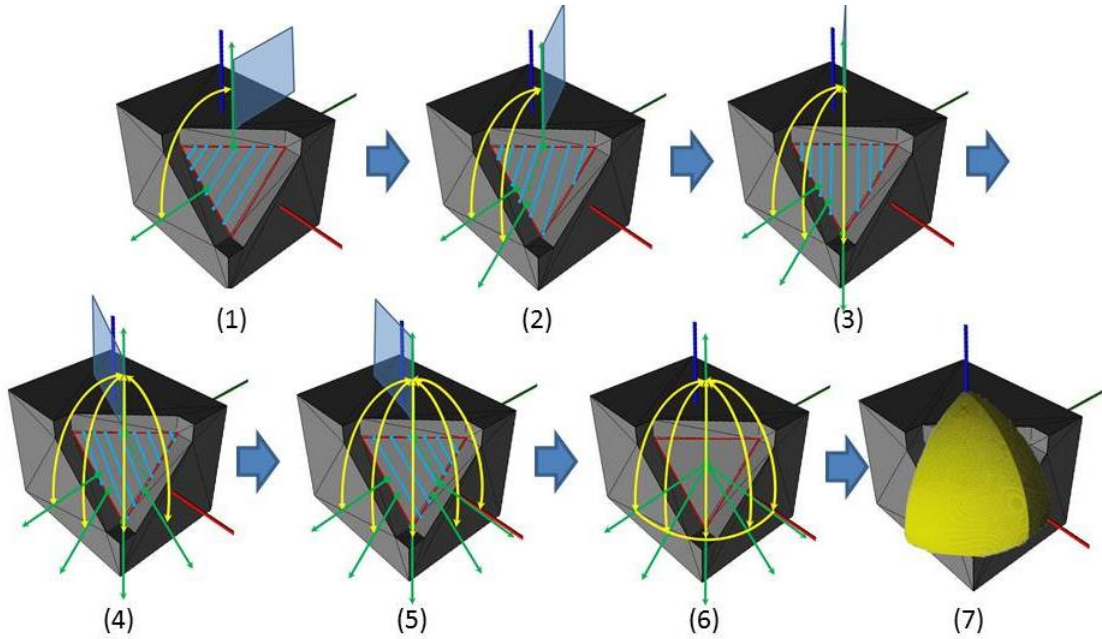
Figure 19 – Union of 2D visibility maps; 1-5) slicing the highlighted facet in 5 different orientations within 90°, 6 &7) visibility union; 1-6) yellow curves denote accumulated 2D visibility maps, blue transparent planes denotes slicing orientations, 6-7) the slicing interval decreases to 1° resulting in a dense GVM.

Capturing the complete GVM

To capture all spherical polygons in a GVM, it is required that the 2D visibility maps cover the entire sphere (Figure 20). One good way is to select a set of coaxial and evenly spaced 2D visibility maps.
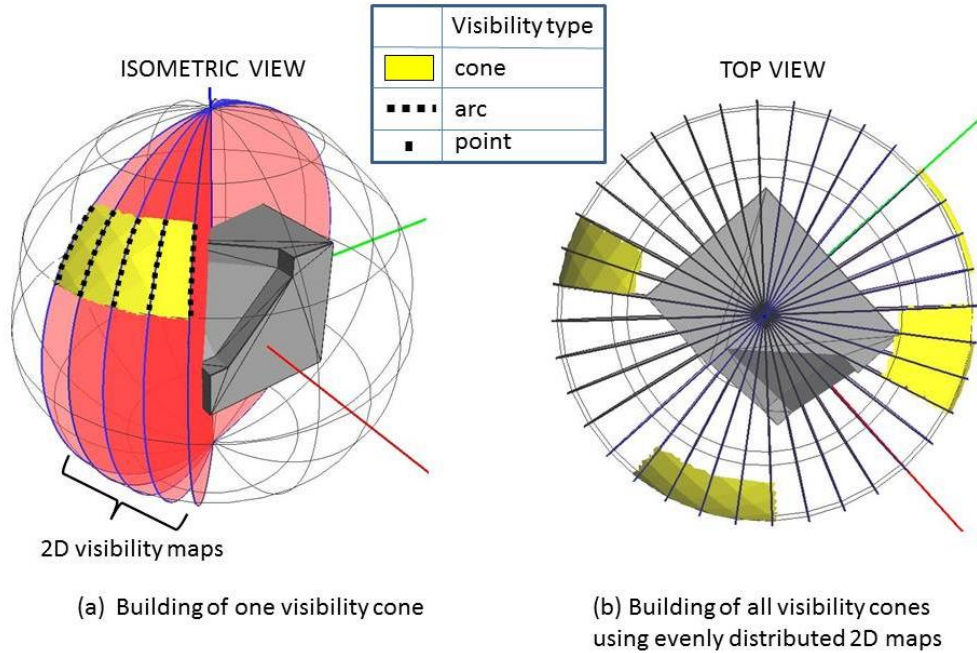
Figure 20 – Construction of visibility cone(s) using 2D visibility maps; (a) building one visibility cone using five 2D visibility maps, (b) build all visibility cones using 2D visibility maps that cover the entire sphere.

Distribution of 2D visibility maps

In this method, all 2D visibility maps are chosen to be coaxial about the Z-axis. With a map-to-map spacing of $\Delta\theta$, a total of $\frac{180}{\Delta\theta}$ 2D visibility maps will be generated. By default, $\Delta\theta = 1$ and the number of 2D visibility maps is 180.

Approximation Three

In the method, a visibility arc is the union of a **_finite_** number of visibility points, and a GVM is the union of **_finite_** number of such visibility arcs. Therefore, the result GVM is an approximation. An example is given in Figure 21 to show the approximation procedure of an exact GVM.
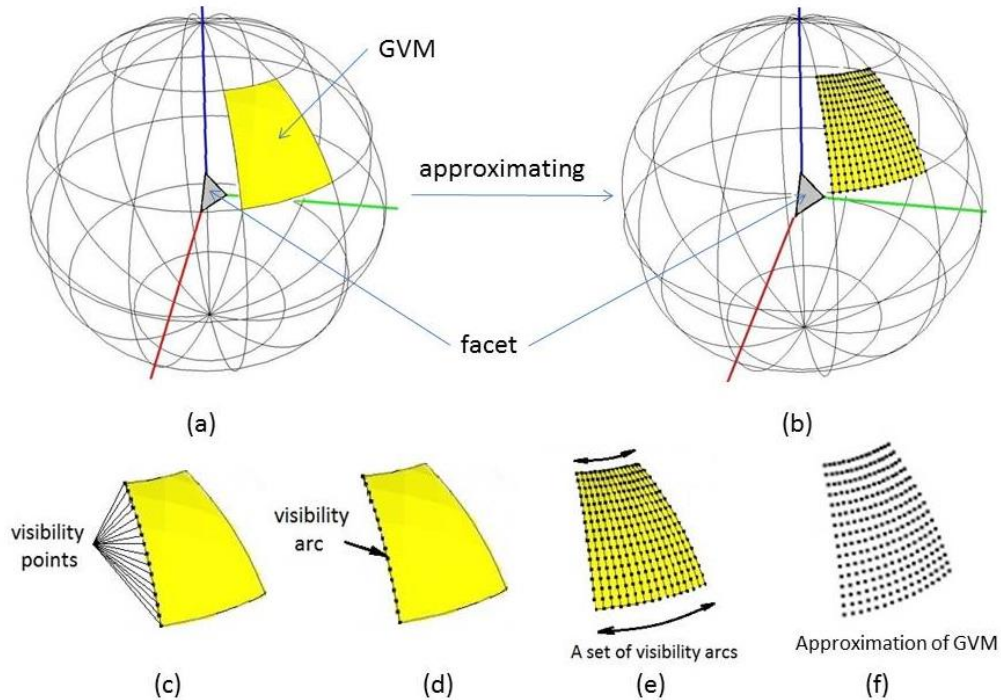
Figure 21 – The procedure to approximate an exact GVM; (a) a facet with an exact GVM (yellow surface patch), (b) the approximated GVM represented as a grid of points, (c)-(d) a visibility arc consists of multiple visibility points, (e)-(f) a set of visibility arcs build a GVM.

3.2.4 Running time estimate

Essentially, the method combines the 2D visibility algorithm and segment- to-facet visibility converting. Thus its running time is generalizes as:

$$\frac{180}{\Delta\theta} \cdot (\, 2\text{DVisibilityAlgorithm} + \text{visibilityConvertingAlgorithm} \,) + \text{Union of 2D maps}$$

3.3 Computing Axis of Rotation

Point visibility to axes of rotation

As the GVM is available, it is used to generate its corresponding axes of rotation. The correspondence of a point visibility to its axes is: Each point visibility corresponds to one

great circle of axes of rotation[1]. This correspondence is shown in Figure 22, while a formal proof is given in the Appendix.
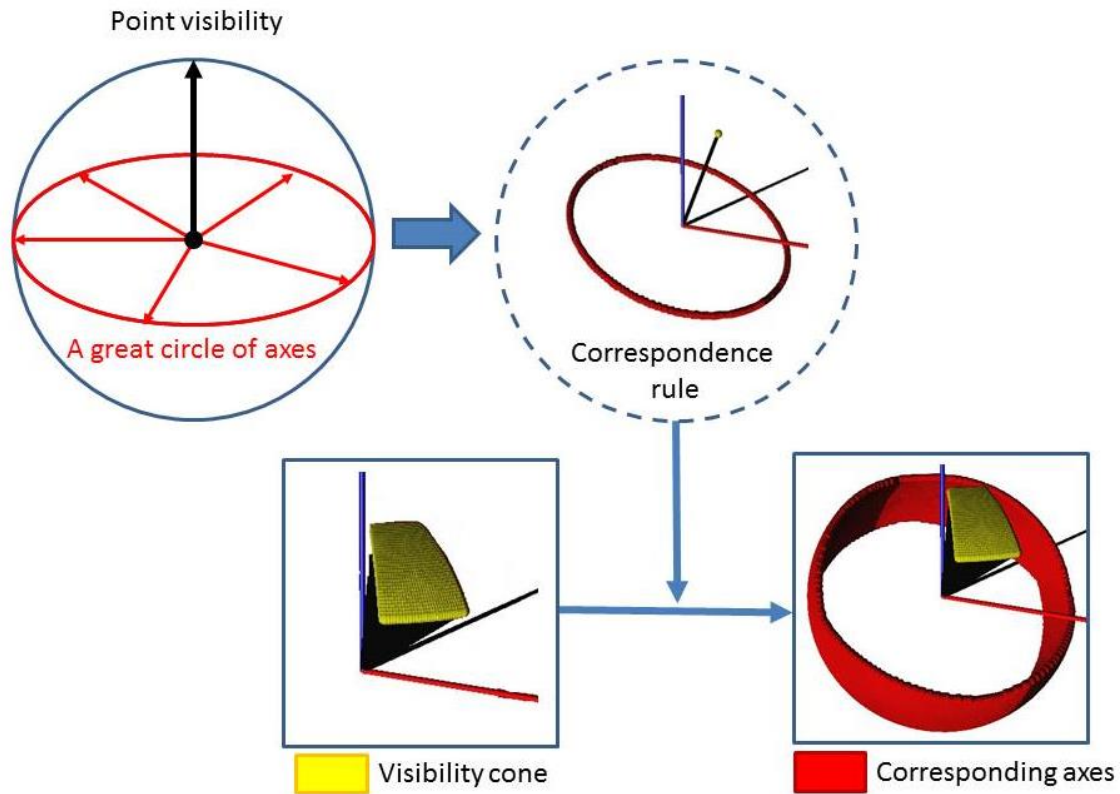
Figure 22 – Finding corresponding axes for a visibility cone.

A region of axes of rotation is formed by applying the correspondence rule to all the visibility points of a facet. **To simplify, an axis of rotation is called <u>axis</u> and this axis region is called <u>axes of a facet</u>.**

Finding the axes of rotation of a model

As we know that 1) a facet is visible from its axes, 2) a model consists of multiple facets, and 3) a model is visible if all its facets are visible from some axes. Therefore, it is concluded that a model is visible if all its facets share at least one common axis. The set of

shared axes is an intersection of axes of all facets. An example is given in Figure 23 to show

the intersection between axes of facet 1 and axes of facet 2.
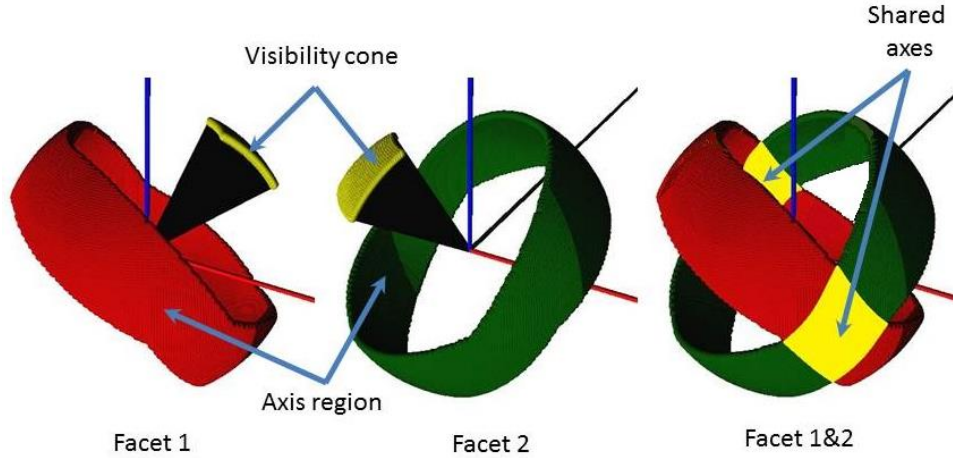
Figure 23 – Intersecting the axis regions of two facets

Formal description

Let $A_{i,j}$ represents the state whether facet j is visible from axis i and V_i represents the

number of facets an axis i can see, and we have the following:

$$A_{i,j} = \begin{cases} 1 \text{ , if facet } j \text{ is visible from axis } i \\ 0 \text{ ,} \qquad\qquad \text{otherwise} \end{cases}$$

$i = 1, \dots, N$ where N is the total number of points on a sphere

$j = 1, \dots, F$ where F is the total number of facets of a model

$$V_i = \sum_{j=1}^{F} A_{i,j} \quad i = 1, \dots, N$$

The final solution is a set $S = \{ i \mid V_i = F, i \in \{1, \dots, N\}\}$

Visibility by surface area

In the case S is a null set, (these is no such axis that all facets are visible), partial

visibility should be considered. Essentially, the axes region for facets provides a

map: facets→axes; while an inverse mapping will provide a map: axes →facets (Figure

24). Using the inverse map, we can define the visibility for each axis.

Map: facets → axes   **Inverse Map: axes → facets**

$$\begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ \vdots \\ f_n \end{bmatrix} \longrightarrow \begin{bmatrix} axis_1 & axis_2 & \cdots & axis_k \\ axis_3 & axis_5 & \cdots & axis_l \\ axis_4 & axis_1 & \cdots & axis_a \\ & \vdots & \ddots & \vdots \\ axis_7 & axis_5 & \cdots & axis_b \end{bmatrix} \qquad \begin{bmatrix} axis_1 \\ axis_2 \\ axis_3 \\ \vdots \\ axis_m \end{bmatrix} \longrightarrow \begin{bmatrix} f_1 & f_3 & \cdots & f_c \\ f_1 & f_5 & \cdots & f_d \\ f_2 & f_6 & \cdots & f_e \\ & \vdots & \ddots & \vdots \\ f_7 & f_9 & \cdots & f_g \end{bmatrix}$$
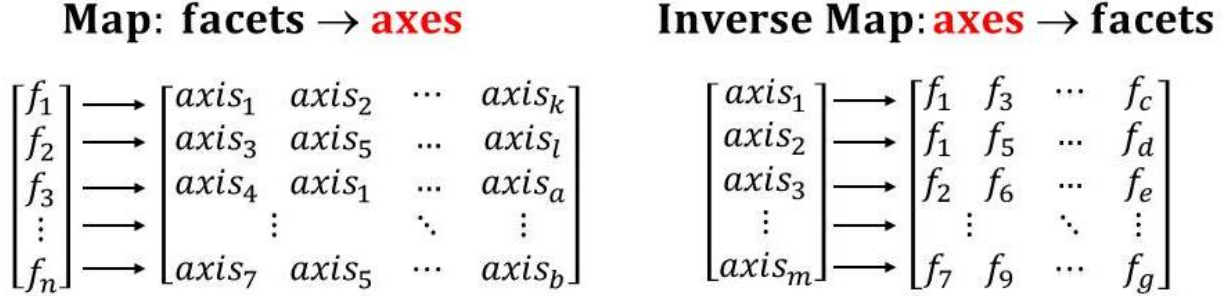
Figure 24 – facets to axes map and the inverse map

Suppose an axis k is mapped to a set of facets: $axis_k \rightarrow S = \{f_1, f_2 \dots f_m\}$. The

visibility of axis k is defined as the percentage of the surface area of the model that axis k

can see:

$$\boldsymbol{Visibility_{axis-k}} = \frac{Area_k}{Area_{total}} \times 100\% = \frac{\sum_{f_i \in S} area_{f_i}}{\sum_{\forall j} area_{f_j}} \times 100\%$$

where $area_{f_j} :=$ area of facet f_j

3.4 Discretization of the Unit Sphere for Visibility and Axis of Rotation Maps

In this thesis, the sphere is discretized into a grid of evenly spacing points for

approximation purposes. There is a 1 degree spacing in both longitude and latitude directions

among points. In total, there are 64442 (=179*360+2) points (Figure 25).

Figure 25 – Discretization of the sphere for visibility and axis of rotation maps

3.5 Summary

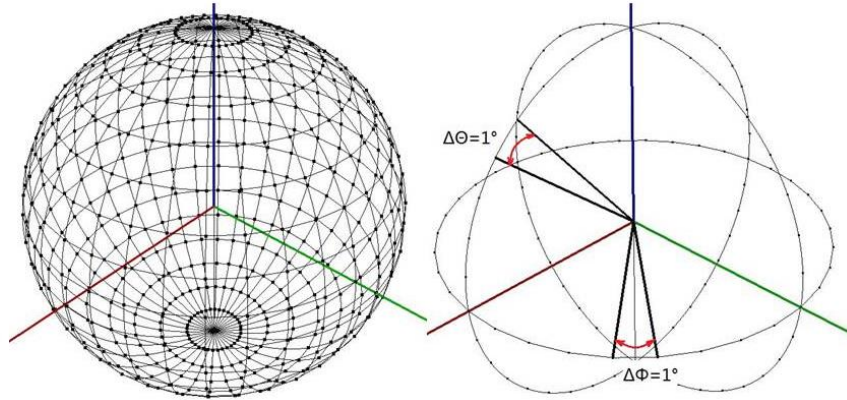To summarize, the proposed method works in the following pattern: First, a facet's GVM is obtained and the GVM's corresponding axis region is mapped. This identical process repeats for every facet of the model and then then a <u>facets to axes</u> mapping is formed. Next, a reverse mapping is conducted such that an <u>axes to facets</u> map is formed. From this <u>axes to facets</u> map, feasible axes are identified as the ones whose size is the number of the facets of the model. Moreover, visibility is calculated for each axis by summing up the surface area of its containing facets and then dividing by the model's total surface area. At this point, the entire computation of the method is done. A program was written to implement the method described above. A graphic user interface is also implemented to visualize the result and allow the user to interact with the program. The implementation and test results will be provided in the next chapter.

CHAPTER 4

IMPLEMENTATION AND TESTING

The methods of this thesis have been implemented and tested in a laboratory setting. A program was written in C++ in Visual Studio 2010. All the tests are conducted using a PC of the following configuration:

- intel(R) Core(TM) i5-2300 CPU @ 2.8GHz Quad-Core, 6GB RAM running Windows 7/64bit on an SSDSC2CT180A3 hard drive.

In order to evaluate the effectiveness and performance of the method, several tests will be presented. One, it is important to verify that the method provides expected results for a known part solution. Two, the visibility map will be measured to determine if the proper number and shape of visibility polygons are found. Three, the method will be compared against known measured to determine its accuracy. Finally, the method will be run for a variety of complex practical part examples to evaluate its performance and robustness. The final sections of the chapter will illustrate how the method has been applied to a current software package and further improvements in processing speed using multi-core parallel computing.

4.1 Initial Verification.

In the example of Figure 26, three simple parts with known solutions were tested. These parts are cubes with one, two or three square pockets, respectively. The cube with one square pocket's axis region is expected to be a great circle. The cube with two orthogonal square pockets' axis region is expected to be a point. And, the cube with three orthogonal

square pockets is expected to have no axis region. In the last case, an additional complete

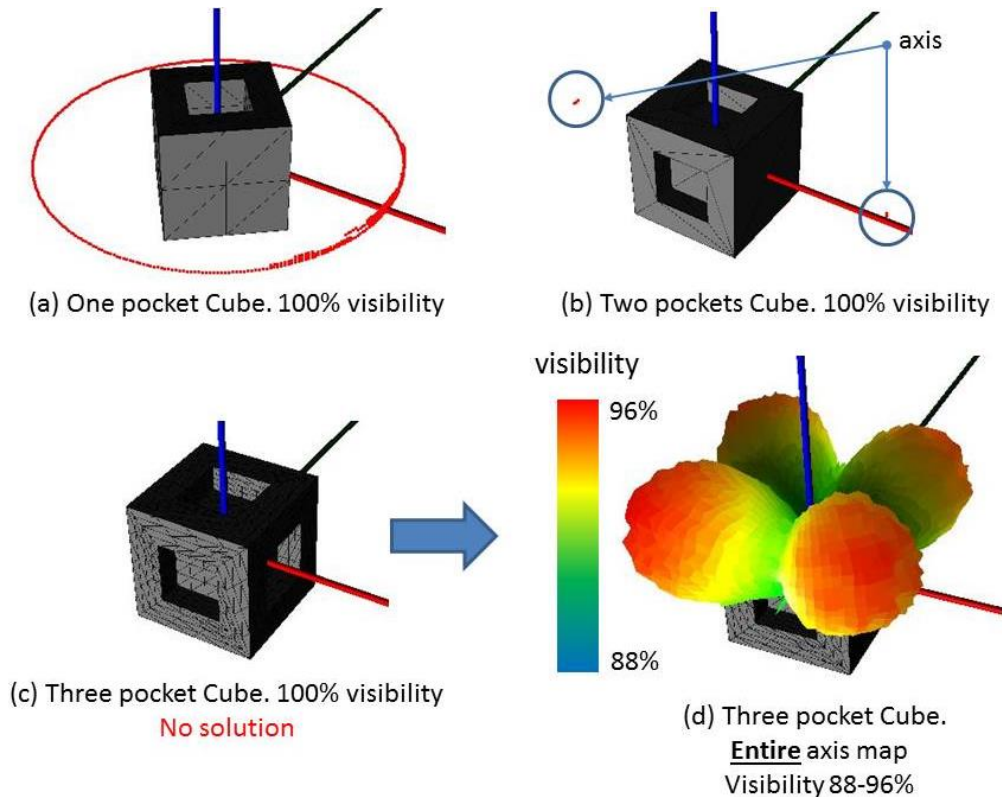axis map is given (*Axis region* is defined to be axes of 100% visibility).

(a) One pocket Cube. 100% visibility

(b) Two pockets Cube. 100% visibility

(c) Three pocket Cube. 100% visibility
No solution

(d) Three pocket Cube.
Entire axis map
Visibility 88-96%

Figure 26 – Method results; (a) One pocket cube's axis region, (b) Two pockets cube's axis region, (c) Three pockets' cube's axis region(none), (d) Three pockets cube's entire axis map

The test result shows that the method's solution is in accordance with the expected

solution. The method provides an axis region if there is one. Otherwise, it provides an axis

map where axes are distinguished by their visibility.
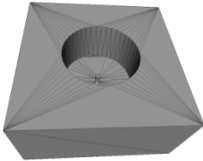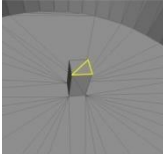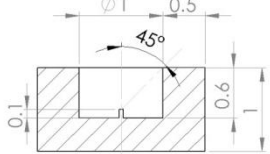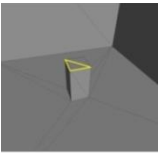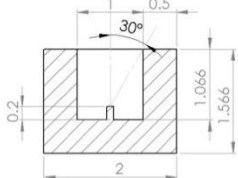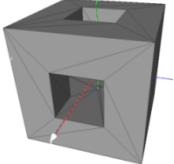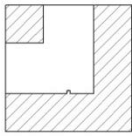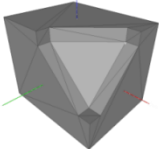
4.2 Testing the Visibility Results

Spherical map test

One of the two fundamentals of the method is computing the visibility map. In the

example of Table 2, four models with expected solutions are tested. More specifically,

models #1 and #2 are used to evaluate the accuracy of the visibility polygon results, while
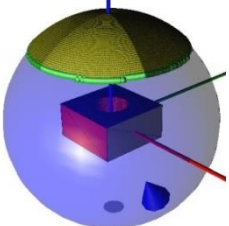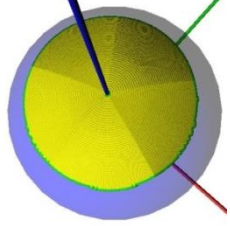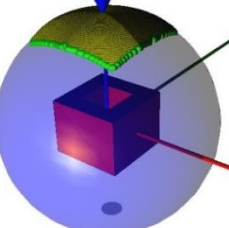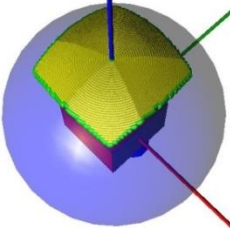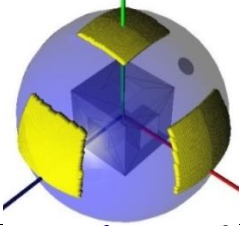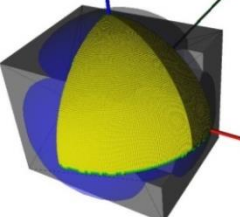
models #3 and #4 are used to verify that the number and the shape of the polygon is correct,
respectively. The dimensions of these models are shown in Table 1. The results refer to a
reference point/facet, which is where visibility is evaluated. For Model #1, #2 and #3, it is at
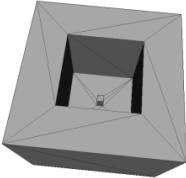the center of the pocket. For Model #4, it is the front triangle. Models #1 and #2 are chosen
because the expected visibility polygon in such configuration has a simple shape and
theoretical value. Thus we can evaluate the error of the proposed method's result easily. In
order to test results, we will create a very small facet in the center of the region to be
evaluated in order to approximate a "point" (since the method is only developed to evaluate
facets).

Table 1 Dimensions of test models

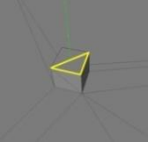| Model#	Purpose	Overview	Facet to test	Section view & Included angle
1	Accuracy			
2	Accuracy			
3	Polygon number			
4	Polygon shape			

Results are shown in Table two, wherein; 1) a transparent blue sphere denotes the unit sphere, 2) the yellow surface (consisting of a set of points) denotes the visibility polygon, 3) the green curve on the yellow surface denotes its boundary, and the analysis is to measure the opening angle of the visibility polygon

Table 2 Method results of spherical map and verification

Model	Test facet	Visibility polygon (green curves denote polygon boundary)		Expected	Algorithm output
		View 1	View 2	Theoretical angle	Measured angle
		Accuracy test			
1	Center of sink			45 °	45 °~47 °
2	Center of sink			51 °~60 °	54 °~63 °
		Spherical polygon number & shape test			
3	Center of Cube			3 polygons	3 polygons
4	Front big facet			One eighth of a sphere	One eighth of a sphere

The result shows that the method performs as expected; where the observed error is mainly due to approximating a point by a facet. Another way to evaluate visibility is showing visible facets from viewpoints. A visibility test under such criteria is shown in Table 3.

Visible facets test

The test model # 5 is axial symmetrical about the x-axis and has six oblique channels specially designed to show visibility difference if viewed from different viewpoints. The tested STL has 3,564 triangles.
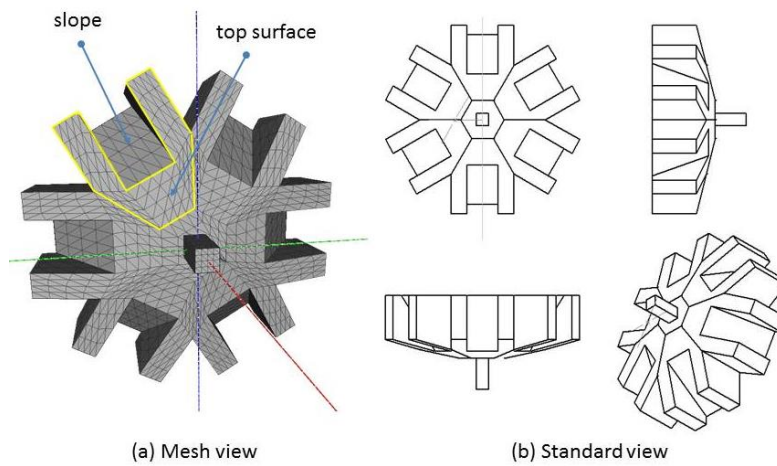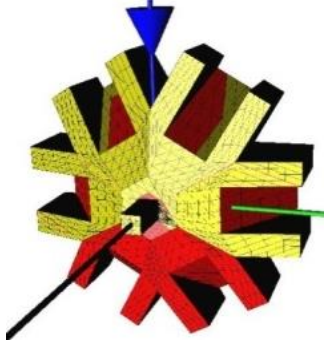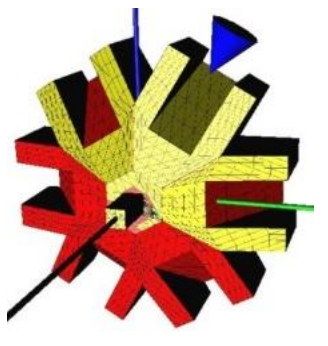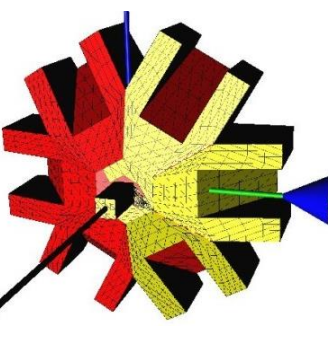
Figure 27 – Test model #5's configuration

As seen in Table 3, 1) the blue cone denotes the view pointing to the origin, 2) the viewpoint lies in YZ-plane and rotates about the X-axis, 3) the model's facet is colored either in red if it is not visible from the viewpoint or yellow if it is visible from the view point, and 4) the view angle is defined as the included angle between line of sight and the Z-axis.

Table 3 Visible facets test on model #5

View angle	0 °	30 °	90 °
Visible facets			

The result of the method is in accordance with experience. When the viewpoint is in between channels (0 °, 60 °), the top surfaces (in positive X direction) of four channels and portion of the neighbor slopes are visible. When the view point is pointing to a channel (30 °, 90 °), this specific channel and its neighbor channels' top surfaces are visible.

4.3 Accuracy Testing

The other fundamental goal of the method is computing the axis region. To test the accuracy of the axis region, an example part with a known and complex axis region is tested (Figure 31).
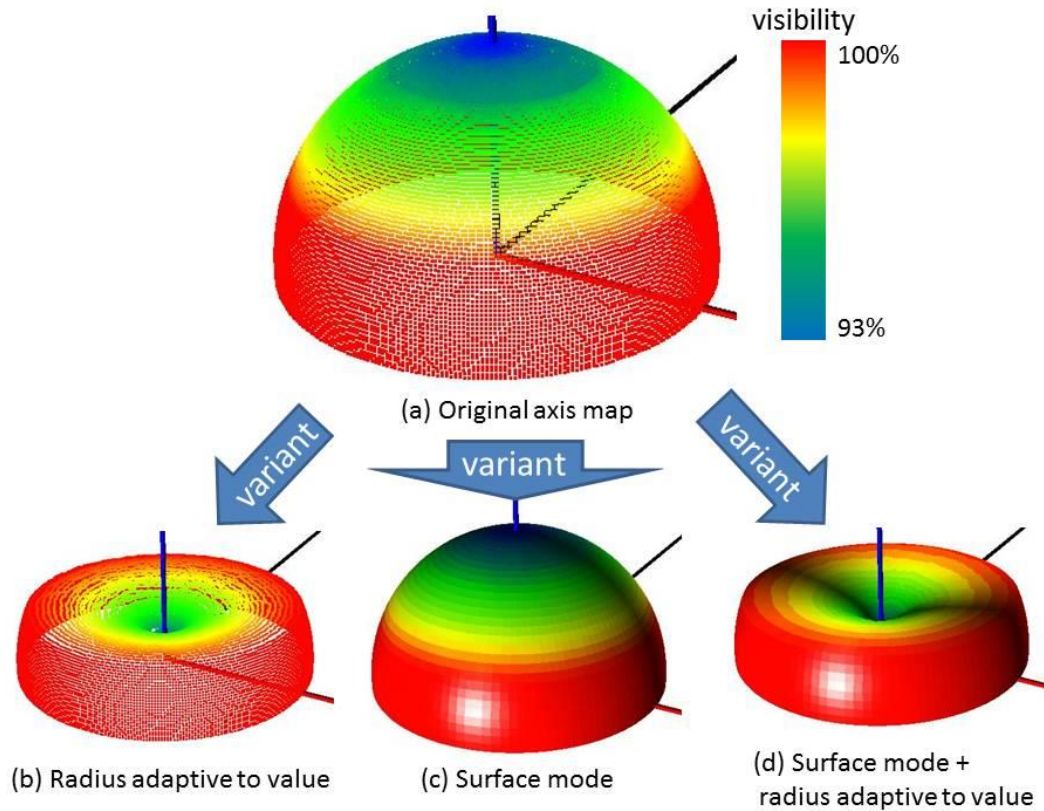
Figure 28 – An Example of axis map (original map and its three variants)

On the axis map of Figure 27, "Axis" is short for "Axis of rotation" and each point on the map corresponds to an axis. The method assigns each point on the map a visibility value:

Visibility V_k is calculated as $V_k = \dfrac{S_k}{S_{total}}$, where:

S_k = visible surface area of axis k

S_{total} = model's total surface area.

A color scheme is used to present the visibility for each axis. The color of a point indicates its visibility, and optionally the visibility of a point can be reflected on its distance to the origin (Figure 28b, d). Finally, the axis map is interpolated to a surface in Figure 28c

& d. One will note that, as the axis map is symmetric about the origin, only the upper hemisphere is shown.

Axis region of known limit
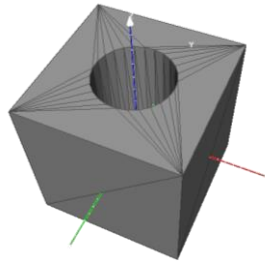
Figure 29 – Test model #6. Cube 2x2x2; Cylinder ⌀1×2

From a feature-based method, it is expected that one type of axis (type 1) will be found; in this case, an axis that is perpendicular to the center line of a through hole (Figure 30a). Theoretically, this model has more feasible axes (type 2) by which the through hole cannot be machined in one setup alone, but could be in several (Figure 30b). Type 2 axes are tilted from the type 1 axes, and in this case, the known limit of the tilted angle θ is 26.57°. Since the part is axial symmetric, such tilted angle results in an axis region of fixed width.
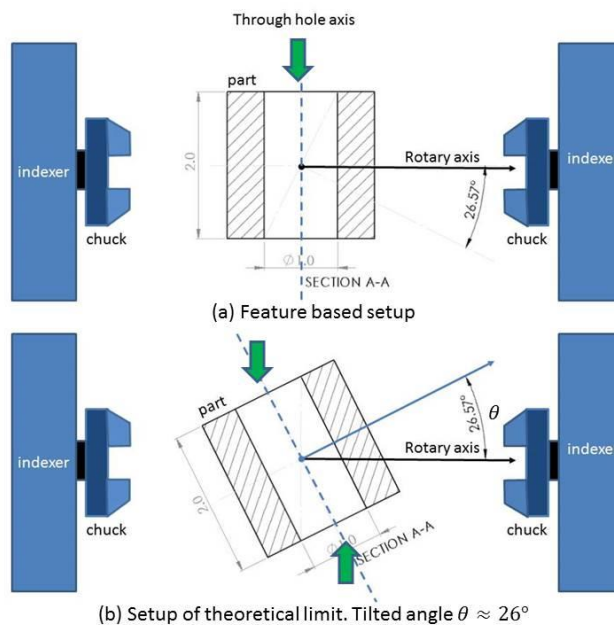
Figure 30 – Setup of theoretical limit for model in Figure 29. Green arrows : access directions.

Method results

Figure 31 shows the axis region results of Model #6. The feasible axis region forms a ring with an opening angle of 26°. The result is close to the expected limit with an error of no more than 1 degree. The subfigures in Figure 31 (b, c and d) show axis rings of different visibility ranges giving a trend of how the axis's visibility changes on the sphere.

(a) Axes ring of 100% visibility

(b) Axes ring of 98-99% visibility

(c) Axes ring of 95-98% visibility

(d) Axes ring of 93-95% visibility
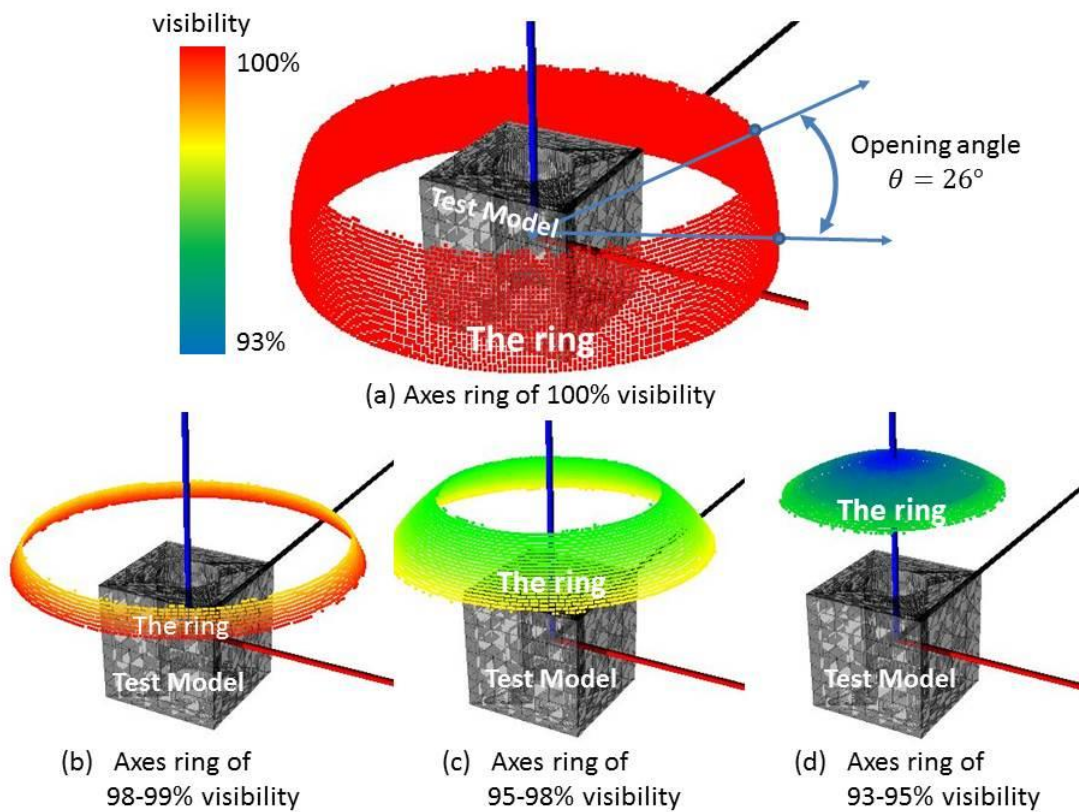
Figure 31 – Axis rings of different visibility ranges; (a) Axes ring of 100% visibility, (b) Axes ring of 98-99% visibility, (c) Axes ring of 95-98% visibility, (d) Axes ring of 93-95% visibility

4.4 Example Industrial Part Models

To evaluate the performance of the method on complex models, three industrial brackets are tested (results shown in Figure 32). The results show that the conventional axes of these industrial brackets are correctly captured. In addition, axes with near complete

visibility are also discovered. The potential value of knowing these near-complete visibility

axes is, when visibility is a necessary condition but not the only selecting criterion, these

axes can be reused as input for other selection schemes.
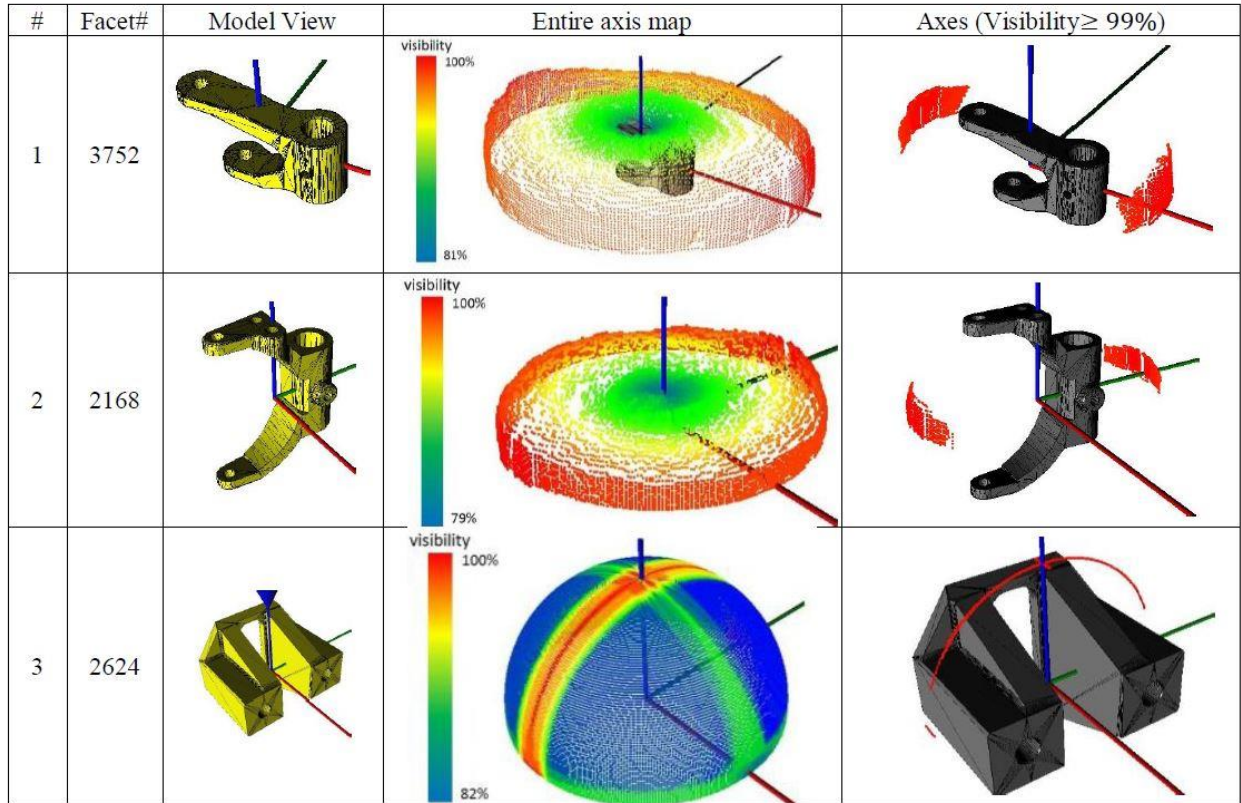
Figure 32 – Axis maps of three complex models. High visibility region is shown separately in the last column.

4.5 Application of Method into Arbitrarily Orientated Model

Motivation

One application of the method is to rectify a model from a bad orientation in terms of machinability. A "bad" orientation may originate from the drawing of the part, the scanning setup, the nature of a biological part or it can be caused by accidentally re-orientating an engineering model in space. In such cases, a part may be inadvertently deemed infeasible for 4-axis CNC machining if only the three principle axis are examined as axis of rotation options (quite typical in practice). Using the method, an axis of maximum visibility can always be found no matter how bad the orientation is. An example is given below for a cube with two orthogonal square pockets; which has a known axis that is perpendicular to both square pocket center lines. Suppose this cube is reoriented to a new orientation in the world frame by first rotating about the y-axis by 45° and then about z-axis by 45°. The new orientation is shown in Figure 33b. Note that a cuboid bulge is purposely added to the cube to denote the its orientation clearly.

(a) Original orientation   (b) New orientation
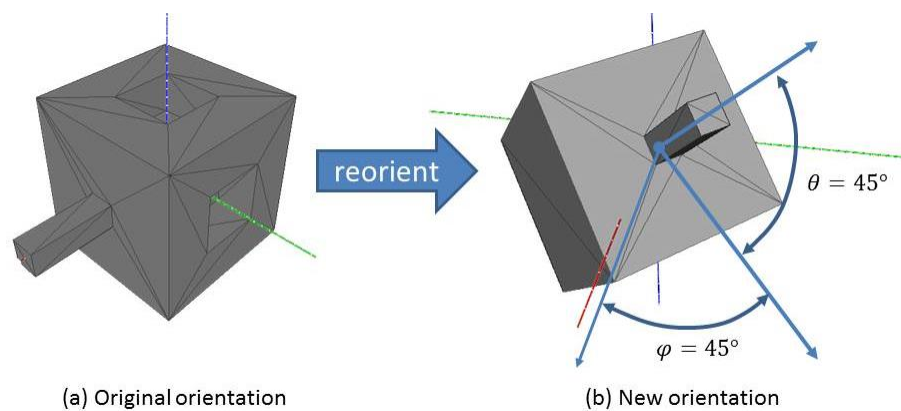
Figure 33 – Model reorientation; (a) original orientation of the two square pockets cube and (b) its new orientation
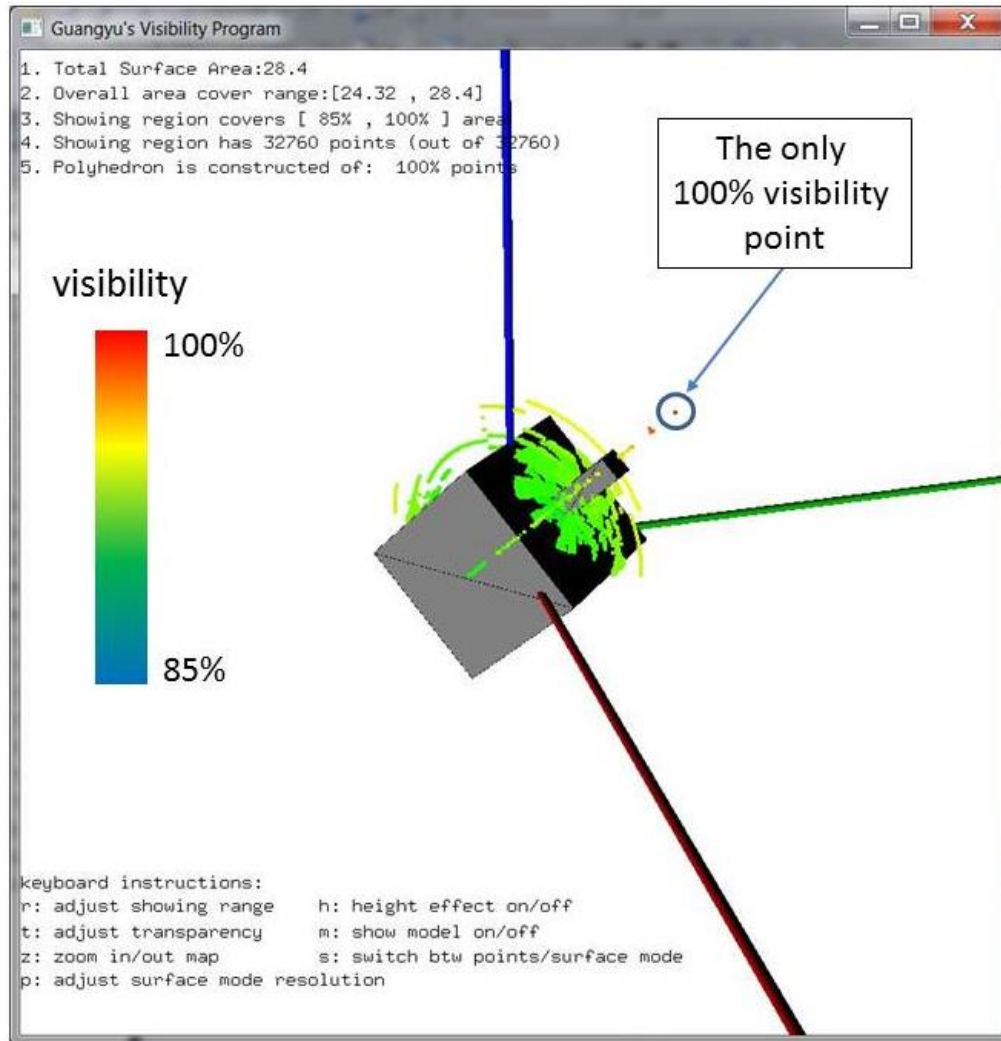
Figure 34 – Axis map of the two square pockets cube with tilted orientation. Map is shown in points under distance effect.

Method results

The results for the cube under new orientation are shown in Figure 34. In this example, a complete program interface is presented. The feasible axis region is a point with spherical coordinates $(R, \varphi, \theta) = (1.5, \ 45°, 45°)$. Using the coordinates (φ, θ), the part can be re-oriented to be suitable for 4-axis machining.

4.6 Application of Method into Current Software

Motivation

In practice, the axis selection scheme for a 4-axis CNC machine takes criteria like machinability, tooling, and sacrificial supports and visibility into account. The CNC-RP software developed is one example, where it uses slice geometry to determine visibility and supports' quality. The limitation is: it estimates a designated axis at a time. The exact optimization of axes is not practical due to evaluating each axis in the design space as large as a hemisphere. Although an approximated solution is available where the design space is reduced to sparse discrete points, the approximation can only improve with considerable cost of time. The reason that visibility is considered first is that visibility of an axis is a *necessary* condition for machinability thus is of highest priority among all criteria and can help us reduce the effort of blindly looking for good axes. The idea is that if a large portion of axes can be first ruled out by visibility, the time budget for other tasks on these axes will be entirely eliminated. The method of this thesis is an improvement since, instead of enumerating every axis in the design space and evaluating them, it only evaluates a small number of axes (in the current setting, the number is 180) and produces a global axis map. The essential explanation of such improvement is that this method utilizes the underlying connection of slices of any two orientations while simply enumerating does not.

Outputs of the method

The method outputs a set of axes. The set is resizable by controlling the visibility threshold. Based on the requirement, axes of 100% visibility or axes of [threshold, 100%] visibility can be returned. In the latter case, the part might not be completely machined. An

example is illustrated in Figure 36 for the part shown in Figure 35 (a simplification of a

Chinese ancient cooking vessel).
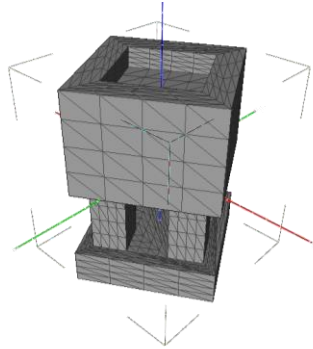
Figure 35 – A Chinese ancient cooking vessel
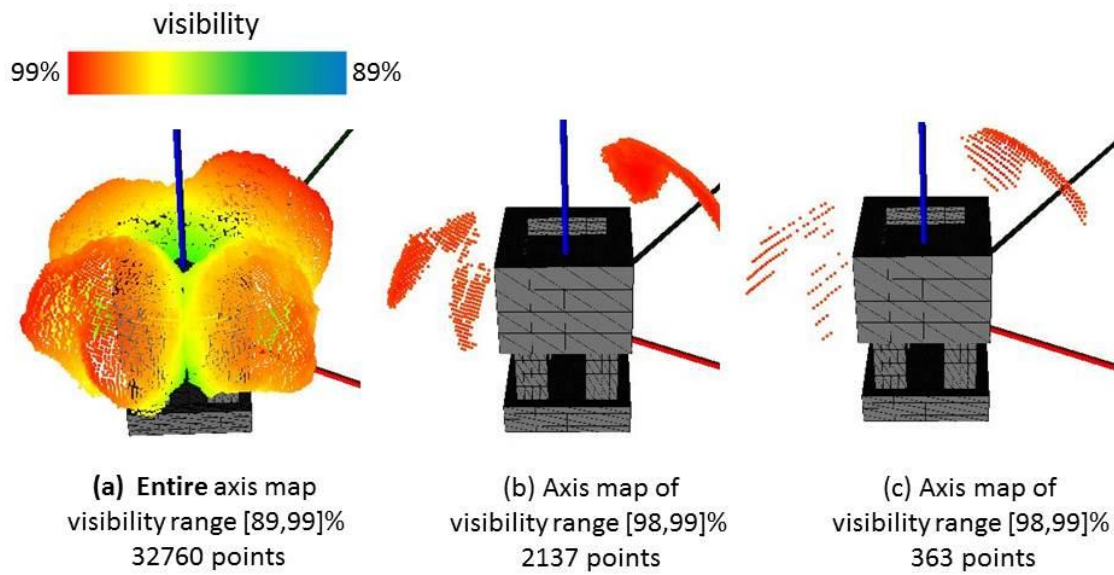
Figure 36 – Chinese ancient cooking vessel's axis maps of different configurations; (a) the entire axis map, (b) axis map of the visibility range [98,99]% and 2137 points, (c) axis map of the visibility range [98,99]% and 363 points.

As can be expected, this part is not completely visible; the outputs are axes of high

visibility only. To facilitate a faster calculation in CNC-RP, the number of output axes is

reduced from 2137 (Figure 36b) to 363 (Figure 36c) but still in an evenly distribution. These

axes are prepared to be loaded into the CNC-RP software interface for machining process

planning.

Implementation in CNC-RP software

The Axis Analysis Interface (Figure 37) in CNC-RP consists of 9 graphical units. Each graphical unit denotes an axis of rotation choice under certain criteria. The three units in the first row denote X-axis, Y-axis and Z-axis as axis of rotation. The three units in the second row denote best axis when visibility, stock diameter or support quality is the sole consideration, respectively. The 7th unit denotes a weighted best axis over visibility, stock diameter and support quality combined. The 8th unit denotes a user picked axis among the imported axes. The last unit provides the functionality to arbitrarily orientate the part (by the user) and monitor the result. In terms of display, each graphical unit has three modules; 1) an OpenGL window to show the part's current orientation, 2) a text below to show visibility, stock diameter and supports' score, and 3) a progress bar to show visibility.

Introduction to CNC-RP Axis Analysis: The criteria

Four criteria (visibility, stock diameter, support quality, overall score) of an axis form an item. All items are collected in a list where they can be sorted by each criterion (Figure 38). The last criterion (overall score) is calculated as:

$$Overall\ Score = \frac{1}{3} \cdot \left(visibility + \left| \frac{d - d_{max}}{d_{max} - d_{min}} \right| + \left| \frac{s - s_{max}}{s_{max} - s_{min}} \right| \right) \times 100$$

Where $visibility \in [0,1]$; d_{max}, d_{min} is the max and min diameter of all axes imported; s_{max}, s_{min} is the max and min support quality of all axes imported.
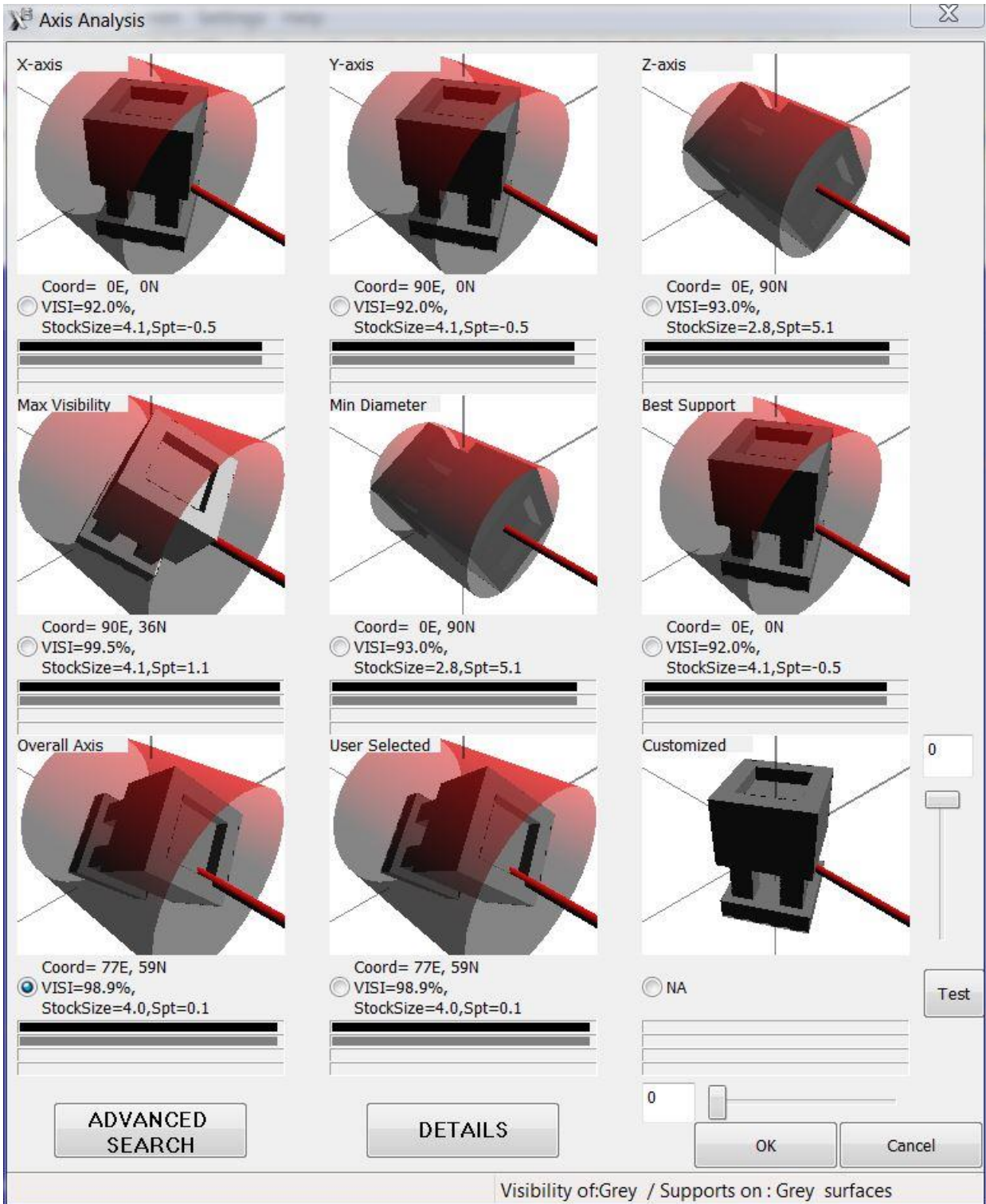
Figure 37 – CNC-RP Axis Analysis Interface

Figure 38 – Axes list (sorted by overall score)

<u>CNC-RP results</u>

From Figure 37, the axis of best visibility has coordinate (90E, 36N) and the axis of best overall score has coordinate (77E, 59N). These two axes are located at irregular orientations that will not be easily discovered. The axis of best overall score is selected, and, after adding sacrificial supports, the part is loaded into the MasterCAM main interface.

(a) Add supports   (b) Load in MasterCAM
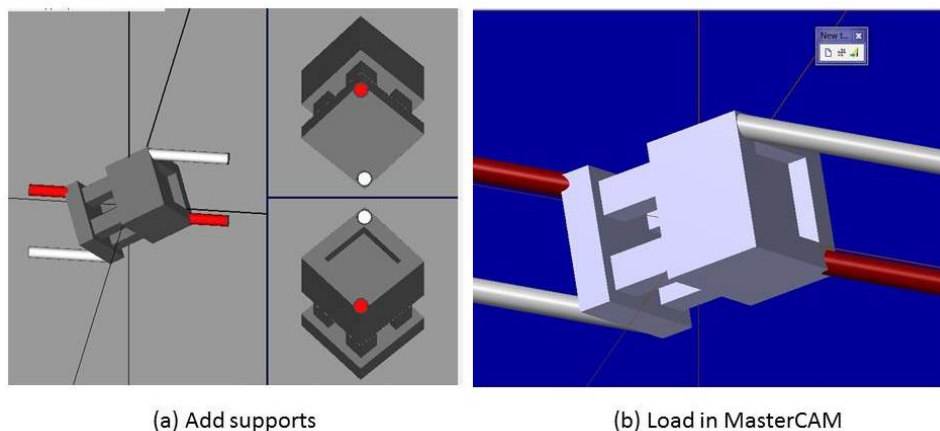
Figure 39 – Post processes after selecting the axis; (a) adding sacrificial supports to the part, (b) loading the part in MasterCAM for further operations.

Table 4 provides the time spent on the entire axis-selection process. Table 5 provides the estimated equivalent time it would have to take if using an enumerating method (as used in CNC-RP Axis Analysis).

Table 4 Time spend using the method + CNC-RP Axis Analysis

	The method in this thesis	CNC-RP Axis Analysis (on 363 axes)	Total
Time	262s	207s	**About 8mins**

Table 5 Estimated equivalent time using CNC-RP Axis Analysis

	Number of axes to examine	Time per axis	Total
Time	5460	0.57s	About 50mins

In table 5, the number of axes to examine is 5460 because the method reduce the size of its output to only one-sixth of its original size; the equivalent discretization of a hemisphere then has 32760/6 = 5460 axes. The time per axis is estimated in Table 4 by 207s/363axes=0.57s/axis.

It is clear that using the method in this thesis to pre-calculate a pool of visibility-qualified axes and then conducting further analysis on them has saved significant time.
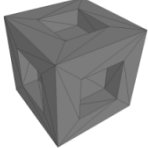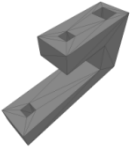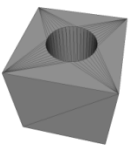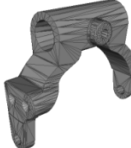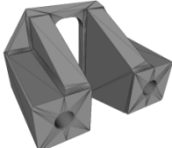
Limitations

It is worthy to note that since the method in this thesis evaluates visibility only, the solution may not be optimal if more criteria are considered. For example, such solution may not be appealing from the perspective of true machinability. Nevertheless, since the method's solution is not unique, criteria of lesser precedence can always be considered later.

4.7 Implementing Parallel Computing

The core of phase 1 – global visibility computation is the computations of a set of 2D visibility maps. These maps lie on 180 planes evenly spaced and are coaxial about z-axis. The computation of each of these 2D visibility maps is independent to each other [4]. Therefore, it is desirable to parallel compute these 2D visibility maps to make use of the independency. In this thesis, the algorithm is rewritten to facilitate OpenMP multicore functionality. A considerable speedup is achieved. The result is shown in Table 6:

Table 6 Process time on computing global visibility

Part name	Sample1	Sample2	Sample3	Sample4	Sample5
Part preview					
# of facets	60	88	176	2168	2624
Dimension (inch)	2.0x2.0x2.0	4.0x1.0x2.2	2.0x2.0x2.0	1.4x3.4x4.3	3.5x5x4.5
Single thread (s)	15	35	33	591	2762
Multithreads (s)	9	15	16	104	672
Speedup (x)	**1.7**	**2.3**	**2.1**	**5.7**	**4.1**

4.8 Summary

The method has been tested in various ways, including the verification of several aspects of a GVM and the accuracy of the axis of rotation map using parts with known solutions. Also, the method's performance with more complex geometry models has been examined. The results overall are satisfactory, and the method has also shown its usage by two applications, both of which are of practical prospect. Finally, method shows significant

potential to be improved with respect to processing time using parallel computing. As the

idea of visibility mapping is widely used in the CAD/CAM world, the method could be

applicable in many different applications beyond that of this work.

CHAPTER 5

CONCLUSIONS AND FUTURE WORK

5.1 Conclusions

This thesis presented a new method to compute a GVM in order to determine feasible axes of rotation for 4-axis CNC machining. The method makes use of slice geometry, which decomposes the GVM problem into a set of 2D problems thus facilitating the use of parallel computing. Testing has shown that the GVM computed is accurate and the algorithm's speed is acceptable. It is further argued that the algorithm discovers a lot more feasible axes than a feature-based method would. Besides, the axis map also gives an intuition of the how the design of the model may affect its visibility. Furthermore, the output of the method can be used for a more advanced analysis that relies on visibility. The fact that axes with poor visibility are ruled out in the output saves computational time for advanced analysis. Last, since the developed method is intended for polygonal models (such as STL), then it is usable across a wide array of CAD platforms, as nearly any can output an STL model.

5.2 Future Work

One of the highlights of this method is that it can be profoundly parallelized (embarrassingly parallel). In this work, parallel computation is implemented only among a set of slicing orientations. In fact, both the fundamental 2D slicing process and the 2D visibility process can be parallelized. As these two processes are the foundations of the entire method and are repeated many times, the expected speed improvement is very significant. Furthermore, the process that converts segment visibility to facet visibility and the mapping

from point visibility to axes of rotation can also be parallelized among facets because essentially it is just repeating the same processes over every facet.

From a design perspective, we can detect which facets are causing the greatest trouble to visibility, thus giving feedback to a designer about which *regions* of the model needs redesign for improved manufacturability. It is possible because we have a mapping from axis to facets. An inverse mapping would give us which axes that a facet is visible from; therefore, using the inverse mapping, we can form a mapping from each facet to the number of axes it relates to. Intuitively, the facets with the least number of axes are causing the greatest trouble. Thus we can mark them for redesign. These are a few examples of future improvements and uses of the visibility and axis searching methods of this thesis.

REFERENCES

[1]   Li, Y., and Frank, M. C., 2012, "Computing Axes of Rotation for Setup Planning Using Visibility of Polyhedral Computer-Aided Design Models," J Manuf Sci E-T Asme, 134(4).

[2]   Suh, S. H., and Kang, J. K., 1995, "Process Planning for Multiaxis Nc Machining of Free Surfaces," Int J Prod Res, 33(10), pp. 2723-2738.

[3]   Usher, J. T., 1896, The Modern Machinist: A Practical Treatise on Modern Machine Shop Methods, N. W. Henley.

[4]   Brown, and Company, S. M., 1914, Practical Treatise on Milling and Milling Machines, Brown & Sharpe Manufacturing Company.

[5]   Company, C. M. M., 1922, A Treatise on Milling and Milling Machines ... 14th Thousand, Cincinnati Milling Machine Company.

[6]   galleryhip.com, 2015, "Toroidal milling cutter / insert / cutting - 30 - 120 mm ", http://galleryhip.com/milling.html. Access Date: 11/10/2015

[7]   Jenny, 2015, "Vertical Turret Milling Machine 3M4M5M6M," Global B2B Trading Platform, http://www.diytrade.com/china/pd/8234033/Name_Brand_Vertical_Turret_Milling_Machine_3M4M5M6M.html. Access Date: 11/10/2015

[8]   Inc., T. T. C., 2015, "SHARP KMA-1 VERTICAL BED TYPE MILLING MACHINE," web, http://www.machinetools4sale.com/shop/item.asp?itemid=941. Access Date: 11/10/2015

[9]   katran, 2013, "CNC rotary fixture , modular , belt driven," http://www.cnczone.com/forums/uncategorised-metalworking-machines/194598-cnc-rotary-fixture-modular-belt-driven-4.html. Access Date: 11/10/2015

[10]   Corporation, O. A., 2013, "Okuma's 5-Axis Vertical Machining Center, MU-500VA," Video Website, https://www.youtube.com/watch?v=CqePrbeAQoM. Access Date: 11/10/2015

[11]   Zalewski, M., 2013, "Guerrilla guide to CNC machining, mold making, and resin casting," Personal, http://lcamtuf.coredump.cx/gcnc/ch2/. Access Date: 11/10/2015

[12]   Frank, M. C., Wysk, R. A., and Joshi, S. B., 2006, "Determining setup orientations from the visibility of slice geometry for rapid computer numerically controlled machining," J Manuf Sci E-T Asme, 128(1), pp. 228-238.

[13]   Liu, M., and Ramani, K., 2007, "Computing an exact spherical visibility map for meshed polyhedra," Proceedings of the 2007 ACM symposium on Solid and physical modeling, ACM, Beijing, China, pp. 367-372.

[14]   Liu, M., Liu, Y. S., and Ramani, K., 2009, "Computing global visibility maps for regions on the boundaries of polyhedra using Minkowski sums," Comput Aided Design, 41(9), pp. 668-680.

[15]   Gan, J. G., Woo, T. C., and Tang, K., 1994, "Spherical Maps - Their Construction, Properties, and Approximation," J Mech Design, 116(2), pp. 357-363.

[16]   Tang, K., Woo, T., and Gan, J., 1992, "Maximum Intersection of Spherical Polygons and Workpiece Orientation for 4-Axis and 5-Axis Machining," J Mech Design, 114(3), pp. 477-485.

[17]   Chen, L. L., Chou, S. Y., and Woo, T. C., 1993, "Separating and Intersecting Spherical Polygons - Computing Machinability on 3-Axis, 4-Axis and 5-Axis Numerically Controlled Machines," Acm T Graphic, 12(4), pp. 305-326.

[18]   Haghpassand, K., and Oliver, J. H., 1995, "Computational Geometry for Optimal Workpiece Orientation," J Mech Design, 117(2A), pp. 329-335.

[19]   Kim, D. S., Papalambros, P. Y., and Woo, T. C., 1995, "Tangent, Normal, and Visibility Cones on Bezier Surfaces," Comput Aided Geom D, 12(3), pp. 305-320.

[20]   Elber, G., and Cohen, E., 1995, "Arbitrarily precise computation of Gauss maps and visibility sets for freeform surfaces," Proceedings of the third ACM symposium on Solid modeling and applications, ACM, Salt Lake City, Utah, USA, pp. 271-279.

[21]   Chen, L.-L., Chou, S.-Y., and Woo, T. C., 1993, "Parting directions for mould and die design," Comput Aided Design, 25(12), pp. 762-768.

[22]   Tarbox, G. H., and Gottschlich, S. N., 1995, "Planning for Complete Sensor Coverage in Inspection," Comput Vis Image Und, 61(1), pp. 84-111.

[23]   Balasubramaniam, M., Laxmiprasad, P., Sarma, S., and Shaikh, Z., 2000, "Generating 5-axis NC roughing paths directly from a tessellated representation," Comput Aided Design, 32(4), pp. 261-277.

[24]   Spitz, S. N., and Requicha, A. A. G., 2000, "Accessibility analysis using computer graphics hardware," Ieee T Vis Comput Gr, 6(3), pp. 208-219.

[25]   Khardekar, R., Burton, G., and McMains, S., 2006, "Finding feasible mold parting directions using graphics hardware," Comput Aided Design, 38(4), pp. 327-341.

[26]   Gregori, R. M. M. H., Volpato, N., Minetto, R., and Da Silva, M. V. G., 2014, "Slicing Triangle Meshes: An Asymptotically Optimal Algorithm," Computational Science and Its Applications (ICCSA), 2014 14th International Conference on, pp. 252-255.

[27]   Nirenstein, S., Blake, E., and Gain, J., 2002, "Exact from-region visibility culling," Proceedings of the 13th Eurographics workshop on Rendering, Eurographics Association, Pisa, Italy, pp. 191-202.

[28]   Keeler, T., Fedorkiw, J., and Ghali, S., 2007, "The spherical visibility map," Comput Aided Design, 39(1), pp. 17-26.

[29]   Dhaliwal, S., Gupta, S. K., Huang, J., and Priyadarshi, A., 2003, "Algorithms for Computing Global Accessibility Cones," Journal of Computing and Information Science in Engineering, 3(3), pp. 200-209.

[30]   Li, Y., and Frank, M. C., 2007, "Computing non-visibility of convex polygonal facets on the surface of a polyhedral CAD model," Comput Aided Design, 39(9), pp. 732-744.

[31]   Chapa, D. H. a. R., 2009, "Non-Euclidean Geometry," Rice University, Mathematics Leadership Institute.

[32]   Miani, M., 2009, "Example: Spherical and Cartesian grids," http://www.texample.net/tikz/examples/spherical-and-cartesian-grids/. Access Date: 11/10/2015

APPENDIX

PROOF OF CORRESPONDENCE FROM VISIBILITY POINT TO AXIS OF ROTATION

The question is:

Given a visibility point P, how to find any feasible axes of rotation?

The solution is:

Find a plane A that passes the origin and is perpendicular to the vector OP. The great circle formed by intersecting plane A and the unit sphere centered at origin contains all the feasible axes.

U: unit sphere

O: the origin

P, T: Points on U

A: a Plane $\perp OP$

C_1: a Circle $= A \cap U$

R: Point on C_1

B: a Plane $\perp OR$

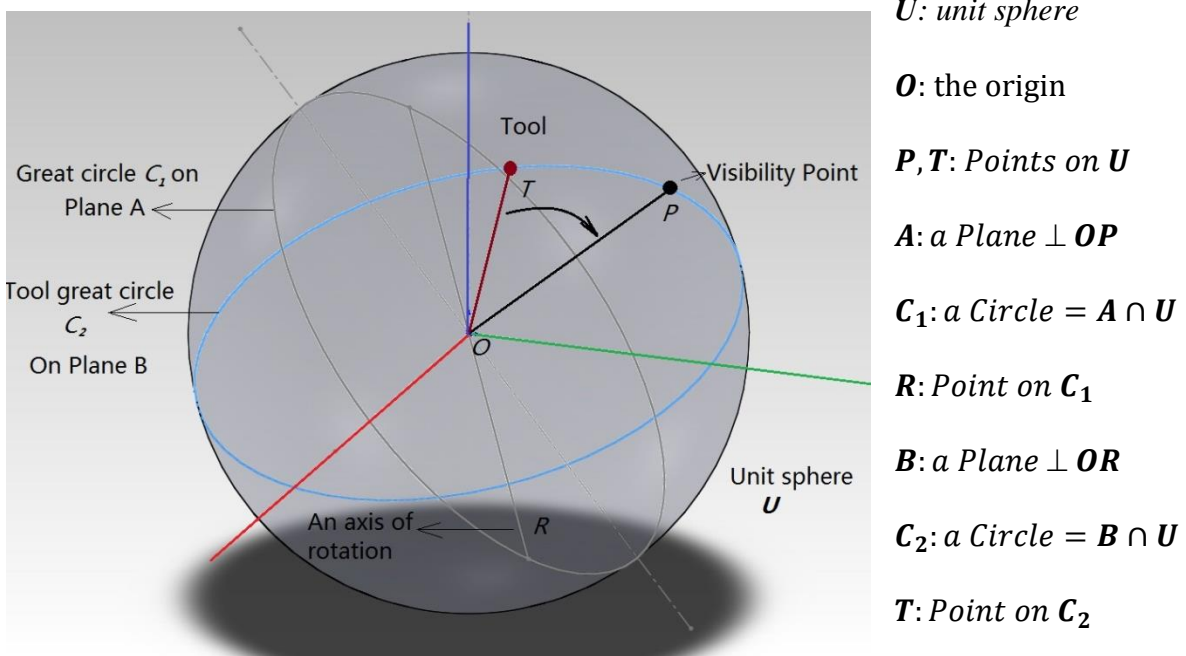C_2: a Circle $= B \cap U$

T: Point on C_2

Figure 40 – Point visibility and corresponding axes of rotations

As illustrated in Figure 40, the procedure to find axes of rotation is described as following:

(1) Given a visibility point P on unit sphere U.

(2) Find a plane A (passing O) and $A \perp OP.$

(3) Find Circle $C_1 = A \cap U.$ C_1 is the solution of axes of rotation.

(4) Pick an arbitrary point R on C_1, OR represents an feasible axis of rotation.

The formal proof

(5) We know the fact that the tool access is always perpendicular to the rotary axis which is OR, thus we can get a Plane $B \perp OR$ where the tool resides.

(6) We know the tool access is a great circle $C_2 = B \cap U,$ where the tool can be any point T on $C_2.$

(7) From (5), (6) we have $OT \perp OR.$

(8) From (2), (3), (4) we have $OP \perp OR.$

(9) From (7), (8) we can conclude that OP and OT are on the same plane B and have an included angle.

Q.E.D

Summary

Statement (9) essentially tells us that as long as we follow the procedure (1)-(4) to find axes of rotation, we can always reorient the tool OT , no matter where they are, to coincide with the visibility point by a rotation defined by axis OR and angle $\angle TOP.$ Since point R is chosen arbitrarily on the circle C_2, all axes of rotation in C_2 will make the facet reachable from its visibility point $P.$