Determining Rotational Setups from Visibility of Slice Files for Rapid CNC Machining

Matthew Frank
Department of Industrial and Manufacturing Systems Engineering
Iowa State University
Ames, IA 50011

Richard A. Wysk          Sanjay B. Joshi
Department of Industrial and Manufacturing Engineering
Penn State University
University Park, PA 16802

Abstract
A method for Rapid CNC machining is being developed in an effort to automatically create functional prototypes and parts in a wide array of materials. The method uses a plurality of simple 2½-D toolpaths from various orientations about an axis of rotation in order to machine the entire surface of a part without refixturing. It is our goal to automatically create these toolpaths for machining, and eliminate the complex planning traditionally associated with CNC machining. In this paper, we consider the problem of visibility to the surface of a model that is rotated about a 4th axis. Our approach involves slicing the CAD model orthogonal to the axis of rotation. The slice geometry is used to calculate 2-D visibility maps for the set of polygons on each slice plane. The visibility data provides critical information for determining the minimum number and orientation of 2½-D toolpaths required to machine the entire surface of a part.

Keywords: Visibility, Rapid Prototyping, Machining, Slice Geometry

Introduction
The labor intensive and time consuming task of manual process planning is recognized as the main factors prohibiting CNC machining from being used as a rapid prototyping (RP) process (Wang, et al. 1999). Existing commercialized RP processes are capable of creating physical models from CAD with little human intervention. Likewise, if CNC machining is to be employed as a rapid process, one will need to automate the steps involved in creating process and fixture plans. A challenging problem for CNC process planning is to determine the orientation or set of orientations that will allow all the features/surfaces of the model to be machined. A desired goal is to machine the part with the fewest number of orientations or setups. Each additional setup requires either a human or a robot to unclamp, reorient and then re-clamp the part. Each new setup requires additional time, but more important, there is the risk of locational errors if the part is not refixtured properly.

As a solution to this problem, a new method for machining complex models using a 3-axis milling machine with a 4th axis indexer is being developed (Frank, et al. 2003). The method involves executing layer-based toolpaths from a plurality of orientations in order to machine the surfaces of a model. These toolpath orientations are about an axis of rotation and are indexed using a 4th axis on the milling machine. This method simplifies the problem of toolpath planning by taking a feature-free approach, whereby the goal is to simply machine the visible surfaces from each orientation rather than planning tool paths for each model feature. In addition, the
problem of fixturing is simplified by borrowing from the concept of sacrificial supports, as used in other RP processes. Throughout the process, the model is secured to the remainder of the stock material by small cylinders attached to the ends of the model along the axis of rotation. The cylinders are cut in order to remove the model after machining. Figure 1 illustrates the setup and steps involved in this new method for rapid CNC machining.

The critical data required for processing a part using this method is the number and orientation of the $2\frac{1}{2}$-D tool paths necessary to machine all the surfaces. It is our goal to automatically create these tool paths for machining, and eliminate the complex planning traditionally associated with CNC machining.

The reachability of the surface for machining can be abstracted to the geometric problem of visibility. We require that any surface need not be completely visible from only one direction, but there must exist a set of orientations that make the surface completely visible.

In order for the surfaces to be machined, they must be visible in the tool approach direction. Other sufficiency conditions must be resolved such as determining a proper tool length and diameter; however, these problems will not be addressed in this paper. In this paper, we consider the problem of visibility to the surface of a model that is rotated about a $4^{th}$ axis. The problem is two-fold: 1) Determine whether all the surfaces of the model can be reached with rotations about the selected axis and if so, 2) Calculate the minimum number of orientations required to machine the part. An open problem is to determine the axis or multiple axes of rotation required to machine all surfaces. This problem will not be addressed in the current paper.

![Diagram of processing steps for rapid machining](image)
Review of Related Work

Many approaches to machinability and visibility analyze the model using surface normal calculations (i.e. Gaussian mapping). Notably, Chen and Woo (1992) performed seminal work with visibility cones. Gan et al. (1994) discuss the properties of spherical maps. Tang et al. (1992) and Chen et al. (1993) use spherical visibility maps to find a 4th axis of rotation such that the maximal number of surfaces can be machined. Suh and Kang (1995) also use spherical visibility maps to create point visibility cones in order to calculate the number of, and range for, controllable CNC machine axes and the workpiece setup.

These approaches use portions of the part surface, most likely pockets and other part features. For each feature, the visibility cone is generated and represented on the unit sphere. Given a CAD model, this requires either some arbitrary surface partitioning or the use of feature recognition software (Joshi and Chang, 1988; Ferreira and Hinduja, 1990; Vanderbrande and Requicha, 1993). Currently Stereolithography (STL) representations for part models have become popular for most RP applications. Unfortunately, feature information is unavailable from an STL model. Recently, Balasubramanium et al. (2000) use a tessellated representation of the model surface for generating toolpaths. They note that visibility cones represent likely access directions, although obstruction from other surfaces may still prohibit tool access. These visibility maps are created for a section of the 3D surface and therefore represent local visibility for that particular section of the part surface.

Using visibility data, one can determine desirable orientations for the workpiece material. If possible, it is best if all processing operations can be done in just one orientation (setup). Problems can occur with maintaining accurate reference locations when a part is processed in multiple steps and has to be re-clamped/fixtured. Haghpassand and Oliver (1995) present algorithms that calculate workpiece orientations for 3- and 4-axis machining. They consider both the part and tool geometry and use an optimization routine to minimize costs. Sarma and Wright (1996) consider the problem of minimizing setups and the number of tool changes. They assume that parts are “simply fixturable”, meaning there are only six orientations that need to be considered (flipping a prismatic workpiece). Joneja et al. (1999) introduce more information into the workpiece setup problem, namely, information about the fixtures. They consider both the geometric and functional characteristics of available fixture elements when designing a setup.

Visibility approaches have been broadly applied to the setup reduction problem as well as toolpath planning for CNC machining. Typically, visibility is based on line-of-sight to a surface and on the local geometry. This is a critical requirement for the machining of any surface. In toolpath planning, other problems exist. Cutting tools obviously do not have zero diameter, and problems with local and global gouging or collision with surrounding surfaces may exist. A difficult problem is handling the collision of other parts of the machine tool (e.g., holders, spindle, etc.) with the workpiece or fixtures. The problem is to determine which directions and orientations will ensure that a tool of some diameter will be able to contact a surface without collision. Tseng and Joshi (1991) developed a method to determine tool-approach directions for machining Bezier curves and surfaces. Local and global gouge-free toolpath planning for a 3-axis machine has also been presented (Glaeser et al., 1999; Yang and Han, 1999). In both papers, the authors discuss tool selection in addition to collision avoidance. Although 4- and 5-axis machining provides more capability in most cases (with respect to orienting the tool), path planning is complicated. Avoiding collisions is quite difficult simply due to the increase in degrees of freedom. Suh et al. (1998a, 1998b) describe an approach to machining with
“additional-axes”. The concept of additional-axis machining is simple: use 3-axis machining, then reorient the part using a 4th and/or 5th axis.

There are several works related to collision and gouge avoidance in 5-axis machining. For example, Yin et al. (2000), discussed visibility approaches and applications to manufacturing. They discuss some downfalls of approaches like visibility cones and visibility mapping where the assumption is that visibility cones exist. For example, if the pocket/feature is not completely visible in one orientation, then a visibility map (intersection of all point visibility cones) will be the null set. Visibility approaches to 5-axis machining is discussed in other works (Lee and Chang, 1995b; Yang et al., 1999).

More recently, researchers have worked on using tessellated representations for 5-axis toolpath planning (Li and Jerard, 1994; Xu et al., 2002). Their work presents local and global gouge avoidance using triangular patches of the part surface rather than parametric surface representations. Balasubramaniam et al. (2003) use computer graphic techniques in conjunction with a tessellated representation in their work.

There is a large amount of published work in 2-D visibility problems. In particular, polygon visibility problems have received much attention (Lee, 1983; Shin and Woo, 1989; Ghosh and Mount, 1991; Gewali and Naftos, 1998; Everett et al., 1999; Kapoor and Maheshwari, 2000). Others present work on the popular Art Gallery Problem, which looks at the minimum number of interior points with which all edges of a polygon (walls of an art gallery) can be viewed (Shin and Woo, 1989; Naftos and Gewali, 1994; and Laurentini, 1999). A variant of the art gallery problem is the Fortress Guard Problem, in which the goal is to find the minimum set of points (guards) placed on the exterior of a polygon (fortress) such that every segment of the polygon is visible from at least one point (O’Rourke, 1987). Peshkin and Sanderson (1986) present the Convex Rope algorithm, which determines the externally visible ranges for vertices of a polygon. Unfortunately, the method is suitable for only a single polygon and does not consider other obstacles in the plane.

2-D visibility cones can be created to represent the visible ranges for a point on a polygon. These cones can be created using Euclidean Shortest Path algorithms (Lee and Preparata, 1984). Guibas et al. (1987) presented several algorithms for visibility and shortest path problems. More recently, Stewart (1999) uses similar approaches to determine 2-D visibility cones for folded surfaces; however, in his approach, the polygonal regions to be investigated must first be triangulated (Toussaint, 1991; Held, 1998). As noted in the work by Stewart, there are difficulties that arise from this type of approach. The main problem is that the connected regions created after the convex hull algorithm often will contain “holes” (polygons with no endpoints on the convex hull). Holes are ignored during triangulation, and therefore also during SEP calculations. For each endpoint, the visible ranges blocked by the holes is calculated and subtracted. In the case of multiple polygons in the plane, several steps were involved including: 1) finding the convex hull of the set of polygons, 2) determining the set of connected regions (polygons) within the convex hull, 3) triangulating this set of polygonal regions, 4) computing the Euclidean shortest paths to each point on the polygons, and 5) separately analyzing holes within regions.

In the work of this paper, several issues make the past 2-D visibility approaches inappropriate. One obvious difference is that we require visibility to the surface of the model, so we must explicitly work with the segments of the polygonal chains, not points. Therefore, somewhat different visibility cones that originate at a line rather than at a cusp must be generated. Secondly, since STL representations can be generated with varying granularity, the
number of points (segments) representing each slice may lead to erroneous visibility results. As such, we wish to add points to the polygonal chains during visibility calculations if either local or global visibility does not exist to a particular segment.

If an existing visibility approach is used, then for every point added polygonal regions to be investigated need to be recreated and the triangulation algorithms rerun. This could be very computationally expensive. Since this approach does not rely on creating these polygonal regions, the problem of dealing with holes is avoided. The only holes, per se, would be any interior chains within the polygons of the slice geometry. These interior chains are obviously not visible and can be detected simply by their chain ordering (clockwise or counter-clockwise), depending on the convention of the slicing algorithm. Furthermore, the existing approaches generate visibility cones for global visibility in one step. In our approach, we wish to separately generate local visibility cones and then calculate blocked visibility, which can be subtracted to determine global visibility. In this manner, if local but not global visibility exists to a given segment, then a new sub-segment is created and analyzed.

**General Methodology**

In our approach to rapid CNC machining, accessibility is limited to one rotation axis; therefore, it is much simpler to solve a series of 2-D visibility problems. Similar to rapid prototyping methods where models are created layer by layer, the algorithm presented in this paper analyzes the CAD model layer by layer. We assume that a proposed axis of rotation is given by the user, similar to choosing a build orientation in the current RP methods. Unlike other approaches, we do not require that all points on any arbitrary section of the surface are simultaneously visible. In other words, it is not a feature-based approach whatsoever. For example, consider the surface illustrated in Figure 2.

![Figure 2](image)

Using an approach like Gaussian mapping, one would conclude that the surface is not visible since the intersection of the visibility cones would obviously yield the null set. However, if we only require that all surfaces are visible in some orientation, then we simply need to calculate the visible ranges for each segment on the polygonal chain.

Since tool access is restricted to directions orthogonal to the rotation axis, 2-D visibility maps for a set of cross sections of the surface of the model are used for visibility mapping. This procedure approximates visibility to the entire surface of the model. For example, consider the part illustrated in Figure 3.
Cross sectional slices of the geometry from an STL model provide polygonal chains that are used for 2-D visibility mapping. A simultaneous visibility solution for all cross sections of the model will approximate visibility to the entire surface. For this simple model and the slice shown in Figure 3a, the chain of edges in the polygon can be “seen” from many different views. If the views in Figure 3b illustrated by the block arrows are chosen, four rotations could be used to machine the part. This implies that four orientations (index rotations) are used and all visible material from each view is removed. If the two orientations noted by the lightening arrows are used, then only two rotations are needed. In this case, two rotations is the fewest number required.

For the method developed in this research, visibility for each polygonal chain is determined by calculating the polar angle range that each segment of the chain can be seen (Figure 4a). Since there can be multiple chains on each slice, one must consider the visibility blocked by all other chains. Therefore, the visibility data for each segment can be a set of ranges (Figure 4b).
If a visible range exists for every segment on each chain, for all slices in the set, then the remaining problem is to determine the minimum set of polar orientations such that every segment is visible in at least one orientation.

Figure 5 illustrates visibility to a set of polygons from an orientation about the axis of rotation. To be applicable for machining, we consider visibility from a line tangent to a circle with diameter at least equal to the diameter of the polygon endpoint set, instead of from an exterior point, as is typically done in 2-D visibility problems.

![Figure 5 – Visibility to a set of polygons from one orientation](image)

The problem of finding the set of rotations sufficient to see every surface of the model can be formulated as a Minimum Set Cover problem. The solution of the set cover provides the minimum set of angles from the set \([0^\circ, 360^\circ]\) such that, for every segment, at least one angle is contained in one of its visibility ranges. However, other criteria will need to be considered in order to determine a minimum, yet sufficient, number of 2½-D toolpaths necessary to machine all surfaces of the part. Tool diameter and length, and the processing sequence for the indexing operations need to be considered. Furthermore, one needs to determine the axis or axes of rotations necessary to machine all the surfaces.

Using an STL file for visibility mapping presents some practical challenges. Depending on the accuracy desired, the STL could have few or many triangular facets representing the
surface. Therefore, each slice will have few or many segments for each polygonal chain. The granularity of the STL representation is typically controlled by specifying the maximum amount that any triangle on the surface can deviate from the actual part surface. A common parameter used to control this is called Chord Height (CH) deviation. Suppose a coarse STL is used, and the slice geometry appears like the one in Figure 6. Notice how visibility does not exist to the segment (uv) shown in Figure 6a; however, if a midpoint is added, then the new sub-segment (uv’) becomes visible (Figure 6b).

If a smaller CH deviation had been chosen, the problem may not have occurred. Of course, if the part has planar surfaces, then modifying the CH parameter will make no difference. For example, a four-sided planar surface will always be represented by only two triangles.

For practical purposes, the approach to visibility for rapid machining will need to be able to handle problems such as STL granularity. In this manner, the visibility algorithm needs to be adaptive depending on the visibility conditions. The addition of midpoints to non-visible segments is an approach that can modify the chain representation dynamically such that a finer mapping of the visibility of the surface can be obtained. In other approaches, the assumption is that the surface representation (set of polygons) is fixed, and the algorithm continues whether visibility ranges are found or not.
Adding additional points using other 2-D approaches would result in a significant amount of additional computation. For each additional point, one would need to recalculate all of the connected regions and then each would need to be re-triangulated. These two significant steps would need to be redone iteratively, each time a midpoint is added. The midpoint method can be executed until a visible segment is found, or the segment length is less than a given stopping criteria. In the current work, the midpoint approach is run until the new segment length is equal to or less than the tool diameter. In general, a segment with length smaller than the tool diameter will not be accessible, even if line-of-sight visibility exists.

Our approach to visibility is unique with respect to two particular characteristics. For one, the approach is completely feature-free. 3D visibility approaches typically need to partition the surface into several surface features. This implies that visibility must exist for some arbitrary section of the surface. In the proposed approach, it is only important that all surfaces of the part geometry are visible in some direction. Since the proposed methodology uses segments of polygons, this implies that each segment must be visible from some polar direction, regardless of any other segments around the one being investigated. The most significant difference is that the proposed methodology is adaptive depending on the visibility of the segments. As described previously, if only a portion of a segment is visible, then the segment is divided iteratively until the visible sub-segments are found and their visible ranges are mapped.

The visibility algorithms operate on slice files generated from the STL model orthogonal to the axis of rotation. Each slice is comprised of multiple simple polygons represented by the endpoints of the polygon segments (edges of the polygon). Each slice is analyzed independently of other slices, and the visible ranges for each segment within each slice are calculated. The first step in the visibility-mapping calculates the visible range for each segment with respect to the chain (polygon) in which it resides.

Next, ranges blocked by every other chain on the slice are subtracted from the visible range. An output file is generated containing an entry for each segment of all slices and the corresponding set(s) of polar angle ranges with which the segment is visible. The output is modified to represent the sets of segments visible from each polar angle. This data is used to formulate a set cover problem, which yields the minimum number of rotations required to view the entire surface.
Visibility Algorithms

It is appropriate to present the visibility mapping in two phases: 1) calculating the visible range for a segment with respect to its chain on which it resides, and 2) calculating the ranges blocked by obstacles (other chains) on the same slice plane. This is done to separate the visibility analysis into two steps; one that defines local visibility and one that defines the ranges of visibility blocked from obstacles, resulting in global visibility directions.

Visibility of a segment with respect to its own chain

Visibility to every segment on each surface slice chain is a necessary condition for the machining of all surfaces in Rapid Machining. Visibility to a point on the surface slice chain will first be presented. This formulation will then be extended to segments defined by consecutive endpoints on the polygonal chain.

Simply stated, visibility to any point \( P_i \) on a polygon \( P \) exists if a line can be drawn from \( P_i \) to infinity in some direction, such that the line does not pass through the interior of \( P \). Consider the polygon in Figure 7. Lines drawn from \( P_i \) through all endpoints in \( P \) yield only two that do not intersect the interior of \( P \). The polar angle from \( P_i \) to the two lines, \( L_1 \) and \( L_2 \), define the range for which \( P_i \) is visible from the exterior of the 2D slice. However, it is not necessary to investigate lines to all points in \( P \) from \( P_i \) since the visibility range has obvious bounds.

Consider the polygon \( P \) and its convex hull (CH), \( S \), in Figure 8. It can easily be seen that all points on the convex hull \( S \) are visible for a viewing range of at least 180\(^\circ\). For any point \( P_i \) not on \( S \), the visible range can be found by investigating points from the adjacent counterclockwise (CCW) convex hull point to the adjacent clockwise (CW) convex hull point. These points will be denoted the left and right convex hull points of \( P_i \), \( LCHP(P_i) \) and \( RCHP(P_i) \), respectively. If one considers the pocket in Figure 8 with the lid formed by the \( LCHP \) and \( RCHP \), visibility is not possible through any points CCW of \( LCHP \) or CW of \( RCHP \).
For any point \( P_i \) not on the \( CH \) of \( P \), a line drawn through a point not in the set \([LCHP, RCHP]\) would have to pass through the interior of \( P \). With that consideration, it is only necessary to calculate the polar angles from \( P_i \) to the points in the set \([LCHP, RCHP]\), excluding \( P_i \). This set is divided into two sets, \( S_1 \) and \( S_2 \) where:

\[
\begin{align*}
S_1 &: [LCHP, P_{i-1}] \\
S_2 &: [P_{i+1}, RCHP]
\end{align*}
\]

Now, the visible range for a point is bounded by the minimum polar angle from \( P_i \) to points in \( S_1 \) and the maximum polar angle from \( P_i \) to points in \( S_2 \). This is the visibility range for the point \( P_i \) with respect to the boundary of its own chain and is denoted \( V(P_i) \), where:

\[
\begin{align*}
RV(P_i) &= \max_{X \in S_2} (\angle P_i X) \quad \text{(The "right" visible bound for } P_i) \\
LV(P_i) &= \min_{Y \in S_1} (\angle P_i Y) \quad \text{(The "left" visible bound for } P_i) \\
V(P_i) &= [\max_{X \in S_2} (\angle P_i X), \min_{Y \in S_1} (\angle P_i Y)]
\end{align*}
\]

Figure 9 illustrates the angles from \( P_i \) to all points in \( S_1 \) and \( S_2 \), highlighting the visibility range \( V(P_i) \).
Using this procedure, it is only necessary to analyze segments of each polygonal chain on
the slice in order to determine visibility to the surface. If visibility to all segments exists and all
polygonal chains are simple polygons, then visibility exists to the polygon. Likewise, visibility
to all polygonal chains on all slices in the set approximates visibility to the entire surface of the
3D model.

Consider the segment $uv$ defined by points $u$ and $v$ in $P$, where:

$$u: P_i \text{ and } v: P_{i+1}$$

The intersection of visibility ranges for the points $u$ and $v$ and the 180º range about the segment
define a feasible range of polar angles in which the segment could be reached. Intersecting the
visibility ranges for each point with the 180º range about the segment is done since visibility to
the segment obviously cannot exist from any direction “behind” the segment. The 180º range
about the segment is the set of angles: $[\angle vu, \angle uv]$. In Figure 10, the ranges are illustrated
($[RV_v, LV_v],[RV_u, LV_u],[\angle vu, \angle uv]$).

![Figure 10 – Ranges used for segment visibility calculation](image)

The intersection of the visibility of $u$ and the visibility of $v$ will have bounds of $RV_v$ and $LV_u$:

$$(V_u \cap V_v) = [RV_u, LV_u] \cap [RV_v, LV_v] \rightarrow [RV_v, LV_u]$$

The sets $S1$ and $S2$ are thus redefined:
The ends of the visibility range are denoted $RV(\overline{uv})$ and $LV(\overline{uv})$, the right and left visibility bounds of the segment $\overline{uv}$, where:

$$RV(\overline{uv}) = \max_{i \in S_1} \angle \overline{vu}, \quad LV(\overline{uv}) = \min_{i \in S_2} \angle \overline{vu}$$

Visibility to the segment $\overline{uv}$ is defined as: $V(\overline{uv}) = [RV(\overline{uv}), LV(\overline{uv})]$ (see Figure 11).

Since not all surfaces will have a simple open pocket as shown in Figure 11, it is necessary to investigate the characteristics of the pocket in order to determine proper bounds for the visibility range, if indeed one exists. There are cases where the minimum angle to points in $S_1$ or the maximum angle to points in $S_2$ is outside of $180^\circ$ range above the segment. In this case, $RV$ or $LV$ is set to the extremes of $[\angle \overline{vu}, \angle \overline{uv}]$, either $(\angle \overline{vu})$ or $(\angle \overline{uv})$, respectively. There is the possibility that no visibility exists as defined by the range $[RV, LV]$ due to severe undercuts or overlapping surfaces above the segment. Figure 12 illustrates several problem surfaces with respect to establishing the visibility bounds.
In each of the cases illustrated in Figure 12, problems occur from naively setting visibility to \([RV, LV]\). This can be avoided by investigating the characteristics of the pocket where the segment \(uv\) resides. The two points in \(S1\) and \(S2\) where the bounds \(RV\) and \(LV\) are calculated are used and are denoted as \(I_1\) and \(I_2\), respectively;

\[ I_1 = P_x \text{ where } x = \arg \min_{x \in S1} (\angle uX) \text{ and } I_2 = P_y \text{ where } y = \arg \max_{y \in S2} (\angle uX) \]

(refer to Figures 12 and 13). The geometric relationships between \(I_1, I_2, u\) and \(v\) can be used to determine if any of the three following cases exist:

1. Whether the entrance to the pocket has an overlapping rim that makes visibility impossible.
2. Whether \(RV\) and/or \(LV\), as calculated, are outside of the 180\(^\circ\) range.
3. Whether the range defined by \(RV\) and \(LV\) defines an opening that permits visibility to the entire segment from one orientation.
Values of particular vector cross products defined by the four points \( u, v, I_1, \) and \( I_2 \) will be employed to test for the three cases described previously:

\[
\begin{align*}
\overrightarrow{I_2v} \times \overrightarrow{I_2u} & \quad \overrightarrow{I_1v} \times \overrightarrow{I_1u} & \quad \overrightarrow{uI_1} \times \overrightarrow{uI_2} & \quad \overrightarrow{vI_1} \times \overrightarrow{vI_2}
\end{align*}
\]

The following algorithm determines feasible values for \( LV \) and \( RV \) or whether visibility exists at all (Figure 14).

\[
\begin{align*}
&\text{IF} \quad \overrightarrow{uI_1} \times \overrightarrow{uI_2} > 0 \\
&\quad \text{IF} \quad \overrightarrow{vI_1} \times \overrightarrow{vI_2} > 0 \\
&\quad \text{Segment} \overrightarrow{uv} \text{ is NOT visible (Overlapping rim, see Figure 12c)}
\end{align*}
\]

\[
\begin{align*}
&\text{IF} \quad \overrightarrow{I_1v} \times \overrightarrow{I_1u} > 0 \\
&\quad \text{IF} \quad \overrightarrow{I_2v} \times \overrightarrow{I_2u} < 0 \\
&\quad \quad LV = \angle \overrightarrow{uv}
\end{align*}
\]

\[
\begin{align*}
&\text{ELSE} \quad \overrightarrow{uv} \text{ is NOT visible (Both LV and RV not in 180° range, see Figure 12b)}
\end{align*}
\]

\[
\begin{align*}
&\text{ELSE} \\
&\quad LV = \angle \overrightarrow{uI_1} \quad (1)
\end{align*}
\]

\[
\begin{align*}
&\text{IF} \quad \overrightarrow{I_2v} \times \overrightarrow{I_2u} > 0 \\
&\quad \text{IF} \quad \overrightarrow{I_1v} \times \overrightarrow{I_1u} < 0 \\
&\quad \quad RV = \angle \overrightarrow{uv} \quad (RV \text{ is outside range, LV is not, see Figure 12a)}
\end{align*}
\]

\[
\begin{align*}
&\text{ELSE}
\end{align*}
\]
Case (3) can exist due to a long segment inside a pocket (see Figure 15a). However, if the segment is divided into two or more smaller segments, visibility may exist to each. To do this, a midpoint is added to the segment, and the visibility algorithm rerun, backtracking to the first new segment. In Figure 15b, a new segment is created by adding a midpoint, and the new segment becomes visible. If the condition persists after one iteration, another midpoint is added to the new segment that is not visible. This is repeated until a stopping criteria (e.g., a minimum segment length) is invoked. We currently use a minimum segment length equal to the diameter of the cutting tool. This is done since, if a segment of length less than the tool diameter still has this visibility condition, then the tool cannot reach through the opening to the pocket regardless (see Figure 15c).
Another case of a problem geometry is a “spiral pocket”, as shown in Figure 16. A spiral pocket can be detected early, when the values for the polar angles of RV and LV are being calculated. While calculating the polar angles from \( u \) or \( v \) to the point in \( S1 \) and \( S2 \), respectively, one can track the cumulative angle from the segment’s endpoints to the points in the corresponding set, starting from the respective convex hull point. Note that if a segment resides in the interior of a spiral pocket, the cumulative angle from one of its endpoints rotated through the points in \( S1 \) or \( S2 \) will exceed 180°. If this condition is detected, then the segment is not visible since it resides in a spiral pocket, and the visibility for the segment is set to null.

The following algorithm is used to detect a spiral pocket early, namely during the calculation of the visibility bounds for the segment. The example shows the calculation of \( LV \); however, a similar approach is taken for \( RV \) (see Figure 17).

\[
\begin{align*}
\text{CurrentMin} &= \angle u (LCHP) \\
\text{LV}(u) &= \text{CurrentMin} \\
\text{Cumangle} &= 0 \\
\text{For all } x \in S1: \\
\text{CurrentMin} &= \text{CurrentMin} + \Theta_i(\overrightarrow{ux} \oplus u(x+1)) \text{where } \Theta_i = \angle x, u, x+1 \text{ for } x \in S1 \\
\text{Cumangle} &= \text{Cumangle} + \Theta_i(\overrightarrow{ux} \oplus u(x+1))
\end{align*}
\]
Note that the preceding investigations are for the case where both points \( u \) and \( v \) are not convex hull points (CHPs). If either or both are CHPs, then determining \( RV \) and \( LV \) is straightforward (see Figure 18).

If \( u \) and \( v \) are both CHPs:
\[
RV = \angle uv \text{ and } LV = \angle vu \quad (Figure \, 18a)
\]

If \( u \) is a CHP and \( v \) is not:
\[
RV = \angle vI \text{ and } LV = \angle vu \quad (Figure \, 18b)
\]

If \( v \) is a CHP and \( u \) is not:
\[
RV = \angle uv \text{ and } LV = \angle uI \quad (Figure \, 18c)
\]

![Figure 18 – Cases where at least one endpoint of the segment \( uv \) is a convex hull point](image)

Visibility blocked by obstacles on the slice plane

The algorithms described in the previous section provide a necessary condition for the visibility criteria of rapid machining; that \( V(\overline{uv}) \) must exist for all segments. This is interpreted as the local visibility of the segment. Other geometric conditions also exist that must be taken into account. For instance, the range \( V(\overline{uv}) \) only considers the visible range with respect to the
chain on which the segment resides. However, obstacles in the slice plane can also block visibility $V(\overrightarrow{uv})$.

The problem is to define the set of ranges where a segment is visible in the presence of other chains on the slice. Each slice contains a set of chains, $j \in J$ where $J = \{j | j = 1, \ldots, n\}$. For any segment on a slice containing $n$ chains, there could be as many as $n$ visible ranges for the segment. We will denote $V(\overrightarrow{uv})_{j^*}$ as the visibility with respect to the chain $j$ on which $\overrightarrow{uv}$ resides, denoted $j^*$. The set of ranges for which $\overrightarrow{uv}$ is visible from the exterior will be called $VIS(\overrightarrow{uv})$ and represents the global visibility of the segment. It is calculated as the visibility of $\overrightarrow{uv}$ with respect to chain $j^*$ minus the set of ranges blocked by other chains on the slice.

For all obstacle chains $j \in J \setminus \{j^*\}$, the polar range blocked by the chain is denoted $VB(\overrightarrow{uv})_j$; Visibility blocked to the segment by another chain on the slice. This set of visible ranges for the segment $\overrightarrow{uv}$ is defined:

$$VIS(\overrightarrow{uv}) = V(\overrightarrow{uv})_j - \sum_{j \in J \setminus \{j^*\}} VB(\overrightarrow{uv})_j$$

Visibility blocked to the segment $\overrightarrow{uv}$ by chain $j$ is the union of the visibility blocked by chain $j$ to point $u$ and the visibility blocked by chain $j$ to point $v$, intersected with the range $[\angle \overrightarrow{uv}, \angle \overrightarrow{vu}]$ about the segment $\overrightarrow{uv}$ (Figure 19). The set of angles blocked to the segment $\overrightarrow{uv}$ are:

$$VB(\overrightarrow{uv})_j = \{[[VB(u)_j] \cup [VB(v)_j]] \cap [\angle \overrightarrow{uv}, \angle \overrightarrow{vu}]\}$$

Where, the set of angles blocked to points $u$ and $v$ are:

$$VB(u)_j = [RB_u, LB_u]$$ (The right and left bounds of the range blocked to $u$ by obstacle $j$)

$$VB(v)_j = [RB_v, LB_v]$$ (The right and left bounds of the range blocked to $v$ by obstacle $j$)
Considering the condition that blocked visibility is only valid within the range \([\angle \vec{u}v, \angle \vec{vu}]\) about the segment, then the union operation yields the following:

\[
(VB_u \cup VB_v) = [RB_u, LB_u] \cup [RB_v, LB_v] \rightarrow [RB_u, LB_u]
\]

Calculating \(RB_u\) and \(LB_v\) is straightforward, as \(RB_u\) is simply the minimum polar angle from \(u\) to all points on the blocker chain and \(LB_v\) is the maximum polar angle from \(v\) to all points on \(P_j\), where \(P_j\) is the set of points for the blocker chain.

\[
RB_u = [\text{Min}(\angle \vec{ux})] \quad \text{and} \quad LB_v = [\text{Max}(\angle \vec{vy})]
\]

It must be determined whether the obstacle is partially or wholly within the range \([\angle \vec{uv}, \angle \vec{vu}]\) about the segment in order to calculate the blockage range. As in the calculation of visibility in the previous section, the characteristics of the blocking chain can be determined by evaluating the locations of four particular points: \(u\), \(v\), \(I_1\) and \(I_2\). In this case, \(I_1\) and \(I_2\) are the points of intersection on the blocker chain for the vectors defining the bounds \(RB_u\) and \(LB_v\), respectively:

\[
I_1 = P_x \quad \text{where} \quad x = \arg \min_{y \in P_j}(\angle \vec{ux}) \quad \text{and} \quad I_2 = P_y \quad \text{where} \quad y = \arg \max_{y \in P_j}(\angle \vec{vy})
\]

Figure 20 illustrates various blocker chain locations.
In addition to the location of the blocker chain, there is the special case where a blocker is in the shape of a "horseshoe" and completely envelopes the segment, which also needs to be detected (Figure 21).

The following algorithm determines the values for $RB_u$ and $LB_v$ (Figure 22). Consider the values of the following vector cross products:

$$\mathbf{v}_1 \times \mathbf{v}_u, \mathbf{v}_u \times \mathbf{v}_1, \mathbf{v}_1 \times \mathbf{v}_u$$

Figure 21 - "Horseshoe" obstacle envelopes the segment.

- IF $\mathbf{v}_u \times \mathbf{v}_1 > 0$
  - IF $\mathbf{v}_u \times \mathbf{v}_1 > 0$
    - IF $\mathbf{v}_1 \times \mathbf{v}_u > 0$ (Horseshoe blocker, see Figure 21)
      - $RB_u = \angle \mathbf{uv}$
      - $LB_v = \angle \mathbf{vu}$
    - ELSE
      - Blocker outside of range (see Figure 20, Location A)
  - ELSE
    - $RB_u = \angle \mathbf{uv}$
    - $LB_v = \angle \mathbf{v}_1 \mathbf{v}_2$ (see Figure 20, Location B)
At this point, all data is available for calculating the sets of global visibility ranges for each segment:

\[
\text{VIS}(uv) = V(uv), - \sum VB(uv)_j \text{ for all } j | j \in J \setminus \{j^*\}
\]

However, it is possible that \( \text{VIS}(uv) = \emptyset \) (null set) while \( V(uv), \emptyset \neq \emptyset \) in which case local visibility exists but the segment is blocked by an obstacle(s). Again, a midpoint is added to the segment, and the algorithm is rerun for the two new segments formed. If the condition persists after one iteration, another midpoint is added to the new segment that is not visible. This is repeated until a stopping criteria (minimum segment length) is invoked. This visibility condition was illustrated in an earlier section and is shown in Figure 6.

The output of the visibility algorithm is the collection of visible ranges given in polar angle about the axis of rotation, as follows:

\[
\text{VIS}_{jk} : \{\Theta_a, \Theta_b, \}, \{\Theta_a, \Theta_b, \}, \ldots, \{\Theta_a, \Theta_b, \}
\]

where:

- \( r_{\text{MAX}} = n \) (number of chains on slice \( k \))
- \( t \) is the segment for \( t \in T \) where \( T = \{\{ \} | t = 1, \ldots, q \} \)
- \( j \) is the chain, for \( j \in J \) where \( J = \{j | j = 1, \ldots, n \} \)
- \( k \) is the slice, for \( k \in K \) where \( K = \{k | k = 1, \ldots, p \} \)
A necessary and sufficient condition for the visibility criteria of rapid machining is that visibility as defined by \( VIS_{ijk} \) exists for all segments on all chains for all slices of the surface geometry:

\[
VIS_{ijk} \neq \emptyset \] (null set), for all segments \( r \), on all chains \( j \), of all slices \( k \).

If this condition is not satisfied, then the entire surface of the part cannot be machined using the proposed method, or at least not using the axis of rotation selected.

**Calculating the Minimum Set of Orientations**

A desired goal of rapid machining is to find the minimum set of index rotations such that all surfaces of the model are machined in at least one orientation. The motivation to minimize the number of rotations is that the machining removal process is executed for each orientation and more orientations results in longer machining time. From the visibility information, a reverse mapping of the sets of segments visible from orientations about the rotation axis is calculated. This mapping is used to derive the minimum set of orientations, such that all surfaces are machined in at least one orientation.

The continuous set of polar orientations \([0^\circ, 360^\circ)\) can be discretized arbitrarily based on a reasonable step size for axis rotations. For illustration, we will assume a step size is of \( 1^\circ \). From the data in \( \{VIS\}_{ijk} \), one can formulate a set corresponding to the segments visible from a given angle \( \Theta_s \). The set \( \{ \Theta \} \) contains the segments visible from each angle \( \Theta_s \) where: \( s = 1^\circ - 360^\circ \).

\[
\Theta_s = \{ \{SEG_{ijk} \} | (\Theta_a \leq \Theta_s \leq \Theta_b) \text{ for some range, } [\Theta_a, \Theta_b], \in VIS_{ijk} \}
\]

The formulation of the Minimum Set Cover problem is straightforward:

---

**Given:** A collection of subsets \( \Theta_s \) of a finite set \( \{SEG\} \) (the set of all segments)

**Solution:** A set cover for \( \{SEG\} \), i.e., a subset \( S' \subseteq S \) such that every element in \( \{SEG\} \) belongs to at least one member of \( \Theta_s \) for \( s \in S' \).

It is noted that the Minimum Set Cover problem is NP-Hard, so we use a Greedy Heuristic to achieve an approximately optimal solution quickly (Chvátal, 1979).
Implementation

The visibility algorithms were implemented in C and tested on a Pentium IV, 2.0Ghz PC, running Windows XP. The software accepts slice files as input and returns several critical process parameters: 1) The minimum number of orientations, 2) The minimum stock diameter, and 3) The distance to the minimum and maximum layer depth for each orientation. The first model investigated is a toy “Jack”. An axis of rotation parallel to the tapered shaft of the Jack was chosen. (See Figure 23a, x-axis alignment) In this manner, it is assumed that the model will be machined by a tool oriented along the z-axis and the stock material is reoriented with index rotations about the x-axis. Cross sectional slices of the model were generated orthogonal to the x-axis, as shown in Figure 23b.

Figure 23 – STL model of the jack and sample cross section
Several trials were run to evaluate the computation time for the visibility algorithm. Computation time is dependent on two main factors: 1) The distance between successive cross sectional slices (i.e., the number of slices) and 2) The granularity of the STL model (i.e., the number of segments per slice). Both factors obviously affect the accuracy with which the set of cross sections approximates the 3-D model; a finer STL file and smaller slice spacing achieves a better approximation of the surface. The granularity of the STL model is controlled by the Chord Height Deviation (CH) and Angle Control (AC). In our tests, AC was held constant while CH was modified. This resulted in several models of increasing granularity, approximated by the number of triangular facets in the STL model. As expected, the computation time for the visibility algorithm increased proportionally with the number of slices. However, results also indicate that the processing time increased linearly as the model granularity was increased. As the number of triangular facets increase, so do the number of segments on each chain of the slice geometry.

The following table presents data on the processing time for numerous sample models of the Jack. The STL model granularity, presented as a range from “extra coarse” to “extra fine”, was generated by adjusting CH. Slice intervals were taken from 0.0025” between each slice, up to 0.040”. In each column of the table, the number of facets, CH, AC, and total number of segments in the model are listed, along with the corresponding computation times for the visibility algorithm. Using the visibility data from the algorithm, a Greedy solution gave a minimum set of orientations required to machine the Jack. The solution is illustrated in Figure 24.

Table 1 – Processing times for the visibility algorithm analyzing the “Jack” model

<table>
<thead>
<tr>
<th>CH</th>
<th>A.C.</th>
<th>Facets</th>
<th>Slice (in)</th>
<th>#sgmts</th>
<th>time (s)</th>
<th>#sgmts</th>
<th>time (s)</th>
<th>#sgmts</th>
<th>time (s)</th>
<th>#sgmts</th>
<th>time (s)</th>
<th>#sgmts</th>
<th>time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0025</td>
<td>0.0075</td>
<td>19,566</td>
<td>0.005</td>
<td>27,285</td>
<td>22.750</td>
<td>25,812</td>
<td>29.390</td>
<td>49,975</td>
<td>36.623</td>
<td>69,212</td>
<td>47.122</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0050</td>
<td>9.772</td>
<td>11,230</td>
<td>0.005</td>
<td>13,553</td>
<td>11.230</td>
<td>18,178</td>
<td>14.671</td>
<td>25,044</td>
<td>18.640</td>
<td>34,458</td>
<td>23.389</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0100</td>
<td>4,850</td>
<td>5.687</td>
<td>0.005</td>
<td>6,781</td>
<td>5.687</td>
<td>9,054</td>
<td>7.405</td>
<td>12,476</td>
<td>9.297</td>
<td>17,306</td>
<td>11.843</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0200</td>
<td>2,375</td>
<td>2.875</td>
<td>0.005</td>
<td>3,409</td>
<td>2.875</td>
<td>4,597</td>
<td>3.907</td>
<td>6,269</td>
<td>4.859</td>
<td>8,683</td>
<td>6.281</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0400</td>
<td>1,182</td>
<td>1.453</td>
<td>0.005</td>
<td>1,655</td>
<td>1.453</td>
<td>2,159</td>
<td>2.032</td>
<td>2,974</td>
<td>2.453</td>
<td>4,123</td>
<td>3.141</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 24 - Rotations required to machine the “jack”
Now, toolpath plans can be generated using CAM software using the set of orientations, max/min depths of cut, and stock diameter from the visibility algorithms. Several metal prototypes have been machined in the laboratory. Although the intent is to integrate the visibility software with CAM and automate process planning tasks, at present the steps of toolpath processing are done manually. The steps that were executed are as follows:

1. Visibility software is executed.
2. The CAD model, oriented about the intended axis, is rotated through each of the orientations of the visibility solution.
3. The toolpath containment boundary is created using the stock and tool diameter, and the length of the part.
4. For each orientation, rough surface pocket toolpaths (MasterCAM) are generated with min. depth at the stock radius and max. depth set to the parameters given from the visibility software.
5. NC code for each orientation is combined manually into a file with 4\textsuperscript{th} axis rotation commands.

The first prototype is the toy Jack. The Jack was machined on a Haas VF-0 3-axis machining center. The number of orientations provided by the visibility method was five. An indexer was used to rotate the part about an axis parallel to the machine x-axis. A tailstock with a dead center was used to provide rigidity. (A tailstock with a 3-jaw chuck, was not available at the time of testing.) The material used was 6061 Aluminum in the form of 1.375” bar stock. Layer thickness was set at 0.005” while the spindle speed and feedrate were set to 7500rpm and 350ipm, respectively (limits of the machine). A 1/8” flat-end mill at 1.5” length was used.

The part was created in approximately 3 hours. Sacrificial supports were added to the ends of the model (0.1” diameter by 0.13” long cylinders). Figure 25 shows the prototype of the Jack in between machining operations. In Figure 26, the Jack is shown after being cut from the stock at the sacrificial supports once all orientations were machined.

Figure 25 – The Jack between orientations

Figure 26 – The Jack prototype
The next model is of a human leg bone, the femur, which was also machined from 6061 2.25” round Aluminum stock. The spindle speed was 7500rpm and the feedrate was 350ipm. A 1/4” flat-end mill was used and the layer thickness was 0.005”. For convenience (automatic rotations), this prototype was machined on a Haas VF-3, 5-axis machining center. The 4th axis was rotated to align with the y-axis, and the 5th axis was used for rotations of the part. Figure 27 presents several views of the completed femur prototype after machining from 3 orientations. Total machining time was approximately 5 ½ hours.

![Figure 27 – The femur prototype]
Future Work and Conclusions

In its current implementation, the visibility method accepts a model oriented by the user and generates visibility information based on that axis of rotation. The next research will be in a method for evaluating multiple orientations and guiding the user at least semi-automatically to a “better” axis of rotation. There are practical issues in regard to machining conditions that need to also be addressed. The sequence of machining operations is very important to avoid poor material conditions. A sequencing approach for the solutions from the visibility is being considered. This includes the addition of extra passes in an effort to avoid thin stock material conditions.

The visibility method presented performs a critical function in automated process planning for rapid machining. The approach provides the data necessary for determining the minimum number of 2½-D toolpaths oriented about an axis of rotation needed to machine the entire surface of a model. Using slice file information as input, the method avoids the problems of feature extraction and identification; an area that has not yielded the automated, robust, solutions we require for rapid machining. This method is also an improvement over existing methods because it modifies the representation of the slice geometry in an effort to seek a feasible visibility solution.

It is envisaged that the visibility algorithm and related functions could be implemented as an additional module in a commercial CAM system. The CAM system could then be utilized in one instance as a rapid prototyping process planner, and then production plans could be developed after the prototype is tested and refined. This method contributes a general visibility methodology that is applicable outside of rapid machining for uses like inspection and reverse engineering with mechanical and optical scanning.