2-2006

Determining Setup Orientations From the Visibility of Slice Geometry for Rapid Computer Numerically Controlled Machining

Matthew C. Frank
Iowa State University, mfrank@iastate.edu

Richard A. Wysk
Penn State University

Sanjay B. Joshi
Penn State University

Determining Setup Orientations From the Visibility of Slice Geometry for Rapid Computer Numerically Controlled Machining

Abstract
A method for rapid computer numerically controlled (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 two-and-a-half-dimensional (21/2-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 a problem that arises in automating this process - visibility to the surface of a model that is rotated about a fourth axis. Our approach involves slicing the computer-aided design (CAD) model orthogonal to the axis of rotation. The slice geometry is used to calculate two-dimensional 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 21/2-D toolpaths required to machine the entire surface of a part.

Keywords
machining, rapid prototyping, slice geometry, visibility

Disciplines
Industrial Engineering | Systems Engineering

Comments
This article is from *Journal of Manufacturing Science and Engineering, Transactions of the ASME* 128 (2006): 228, doi: 10.1115/1.2039100. Posted with permission.

Determining Setup Orientations From the Visibility of Slice Geometry for Rapid Computer Numerically Controlled Machining

Matthew C. 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

A method for rapid computer numerically controlled (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 two-and-a-half-dimensional ($2\frac{1}{2}$-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 a problem that arises in automating this process—visibility to the surface of a model that is rotated about a fourth axis. Our approach involves slicing the computer-aided design (CAD) model orthogonal to the axis of rotation. The slice geometry is used to calculate two-dimensional 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 $2\frac{1}{2}$-D toolpaths required to machine the entire surface of a part. [DOI: 10.1115/1.2039100]

Keywords: visibility, rapid prototyping, machining, slice geometry

Introduction

The labor-intensive and time-consuming task of manual process planning is recognized as the main factor prohibiting computer numerically controlled (CNC) machining from being used as a rapid prototyping (RP) process [1]. Existing commercialized RP processes are capable of creating physical models from computer-aided design (CAD) with little human intervention. Likewise, if CNC machining is to be employed as a rapid prototyping process, one will need to automate the steps involved in creating process and fixture plans. A brief overview of the approach is illustrated in Fig. 1. A method for machining complex models using a three-axis milling machine with a fourth axis indexer is being developed [2,3]. 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 fourth 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 toolpaths 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 at the end of machining in order to remove the model.

A challenging problem for CNC-RP process planning is to determine the orientation or set of orientations that will allow all the 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 reclamp the part. Each new setup requires additional time, but more importantly there is the risk of locational errors if the part is not refixtured properly. The critical data required for processing a part using this method is the number and orientation of the two-and-a-half-dimensional ($2\frac{1}{2}$-D) toolpaths necessary to machine all the surfaces. It is our goal to automatically create these toolpaths for machining and eliminate the complex planning traditionally associated with CNC machining.

The accessibility 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 fourth axis. The problem is twofold: (i) Determine whether all the surfaces of the model can be reached with rotations about the selected axis and if so, (ii) 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.

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 performed seminal work with visibility cones [4]. Gan et al. discuss the properties of spherical maps [5]. Tang et al. and Chen et al. use spherical visibility maps to find a fourth axis of rotation such that the maximal number of surfaces can be machined [6,7]. Suh and Kang also use spherical visibility

Contributed by the Manufacturing Engineering Division of ASME for publication in the Journal of Manufacturing Science and Engineering. Manuscript received April 9, 2004; final manuscript received March 15, 2005. Review conducted by D.-W. Cho.

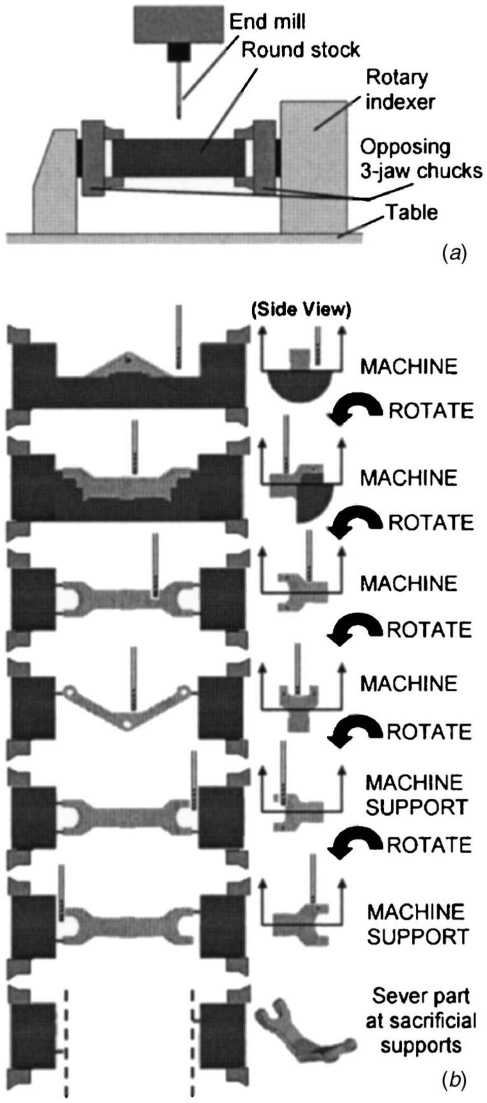


Fig. 1 Rapid machining: (*a*) setup and (*b*) process steps

maps to create *point visibility cones* in order to calculate the number of, and range for, controllable CNC machine axes and the workpiece setup [8].

These approaches only consider portions of the part surface, typically 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 [9–11]. Recently, Balasubramanium et al. used a tessellated representation of the model surface for generating toolpaths [12]. They noted 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 three-dimensional (3D) surface and, therefore, represent local visibility for that particular section of the part surface.

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. Tseng and Joshi developed a method to determine tool-approach directions for machining Bezier curves and surfaces [13]. Local and global gouge-free toolpath planning for a three-axis machine has also been presented [14,15]. In both papers, the authors discuss tool selection in addition to collision avoidance. Although four- and five-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. and Suh and Lee describe an approach to machining with "additional axes" [16,17]. The concept of additional-axis machining is simple: use three-axis machining, then reorient the part using a fourth- and/or fifth-axis indexer. More recently, researchers have worked on using tessellated representations for five-axis toolpath planning [18,19]. Their works present local and global gouge avoidance using triangular patches of the part surface rather than parametric surface representations. Balasubramanium et al. use computer graphic techniques in conjunction with a tessellated representation in their work [20].

There is a large amount of published work in 2D visibility problems. In particular, polygon visibility problems have received much attention [21–26]. 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 [22,27,28]. 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 [29]. Peshkin and Sanderson present the convex rope algorithm, which determines the externally visible ranges for vertices of a polygon [30]. Unfortunately, the method is suitable for only a single polygon and does not consider other obstacles in the plane.

Two-dimensional 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 [31]. Guibas et al. presented several algorithms for visibility and shortest path problems [32]. More recently, Stewart uses similar approaches to determine 2D visibility cones for folded surfaces [33]; however, in his approach, the polygonal regions to be investigated must first be triangulated [34,35]. As noted in the work by Stewart [33] 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 are calculated and subtracted in a secondary step.

General Methodology

In this approach to rapid CNC machining, accessibility is limited to one rotation axis; therefore, it is much simpler to solve a series of 2D 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. However, 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 Fig. 2. Using an approach such as 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, 2D 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 Fig. 3. Cross-sectional slices of the geometry from a stereolithography (STL) model provide polygonal chains that are used for 2D visibility mapping. A simultaneous visibility solution for all cross sections of the model

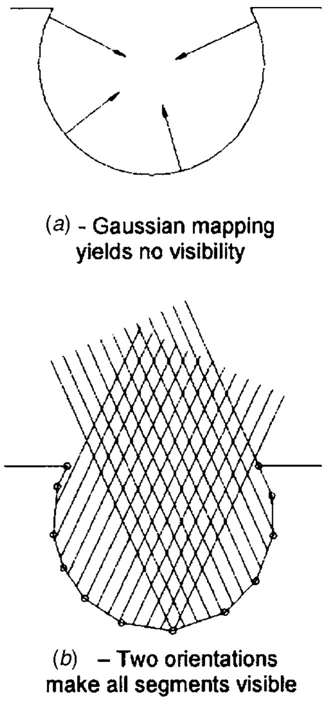

(a) - Gaussian mapping yields no visibility
(b) – Two orientations make all segments visible

Fig. 2 Comparison of visibility

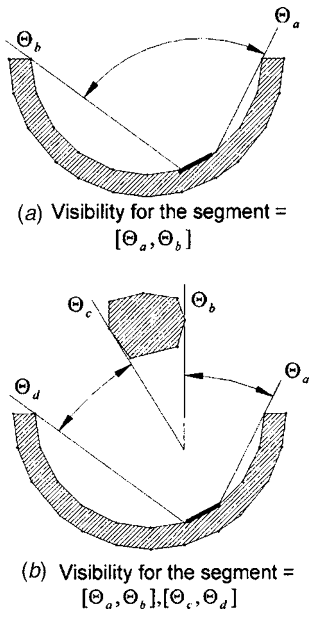

Θ_a
Θ_b
(a) Visibility for the segment = $[\Theta_a,\Theta_b]$
Θ_c
Θ_d
(b) Visibility for the segment = $[\Theta_a,\Theta_b],[\Theta_c,\Theta_d]$

Fig. 4 Visible ranges for segment of polygonal chain

will approximate visibility to the entire surface. For this simple model and the slice shown in Fig. 3(*a*), the chain of edges in the polygon can be "seen" from many different orientations. If the orientations in Fig. 3(*b*) 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 *lightning* 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 (Fig. 4(*a*)). 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 (Fig. 4(*b*)).

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 2D visibility problems.

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, 360 deg) such that, for every

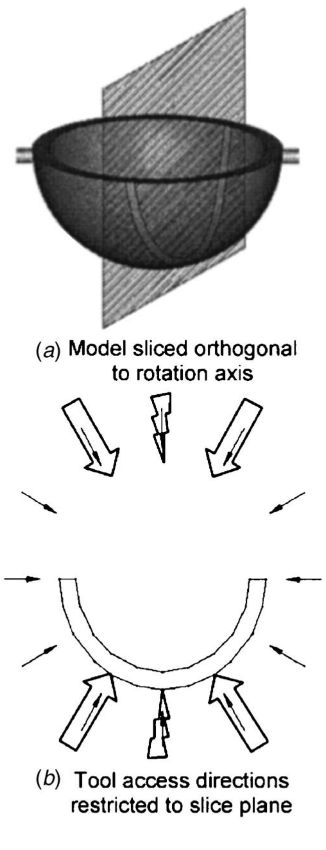

(a) Model sliced orthogonal to rotation axis
(b) Tool access directions restricted to slice plane

Fig. 3 Sample model with cross section for visibility mapping

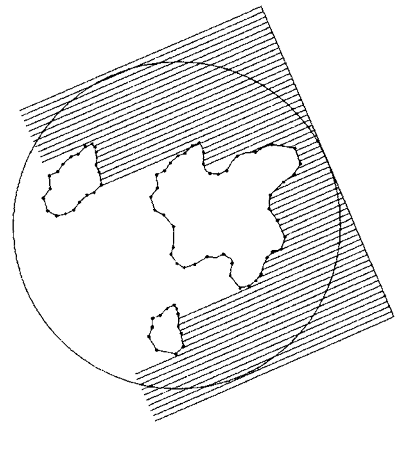


Fig. 5 Visibility to a set of polygons from one orientation

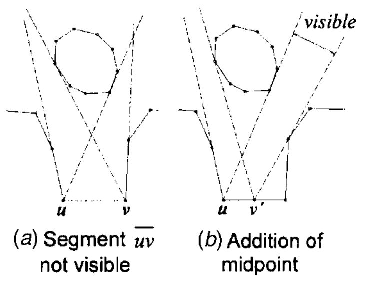


Fig. 6 Midpoint yields visibility to newly formed segment

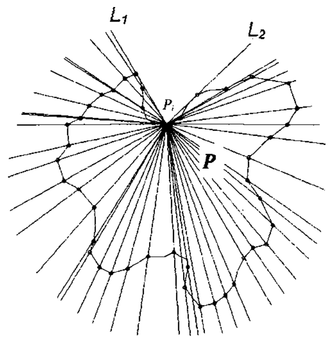


Fig. 7 Lines (L_1, L_2) from P_i through points in P

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 $2\frac{1}{2}$-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 Fig. 6. Note how visibility does not exist to the segment ($\overline{uv}$) shown in Fig. 6(*a*); however, if a midpoint is added, then the new subsegment ($\overline{uv'}$) becomes visible (Fig. 6(*b*)).

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 nonvisible 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 2D 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 retriangulated. These two significant steps would need to be redone iteratively, each time a midpoint is added. The midpoint method would 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 after midpoint addition.

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 subsegments 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, polar angle 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: (i) calculating the visible range for a segment with respect to the chain on which it resides and (ii) 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 $\boldsymbol{P}$ 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 $\boldsymbol{P}$. Consider the polygon in Fig. 7. Lines drawn from P_i through all endpoints in $\boldsymbol{P}$ yield only two that do not intersect the interior of $\boldsymbol{P}$. 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 $\boldsymbol{P}$ from P_i since the visibility range has obvious bounds.

Consider the polygon $\boldsymbol{P}$ and its convex hull (CH) $\boldsymbol{S}$ in Fig. 8. It can easily be seen that all points on the convex hull $\boldsymbol{S}$ are visible for a viewing range of at least 180 deg. For any point P_i *not* on $\boldsymbol{S}$, 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 de-

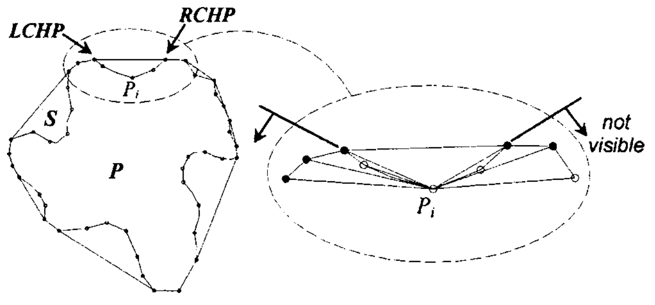


Fig. 8 A point P_i and its adjacent convex hull points

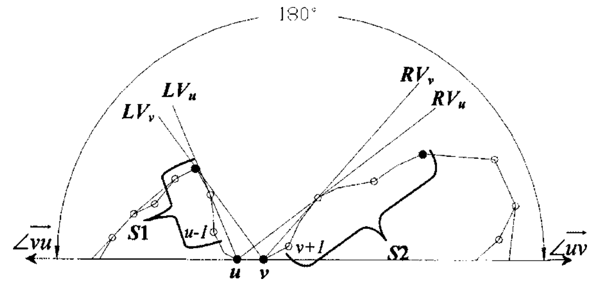


Fig. 10 Ranges used for segment visibility calculation

noted the *left* and *right* convex hull points of P_i, LCHP(P_i) and RCHP(P_i), respectively. If one considers the pocket in Fig. 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 $\boldsymbol{P}$, a line drawn through a point *not* in the set [LCHP, RCHP] would have to pass through the interior of $\boldsymbol{P}$. 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, $S1$ and $S2$ where $S1:[\mathrm{LCHP},P_{i-1}]$ and $S2:[P_{i+1},\mathrm{RCHP}]$.

Now, the visible range for a point is bound by the minimum polar angle from P_i to points in $S1$ and the maximum polar angle from P_i to points in $S2$. 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: $\mathrm{RV}(P_i)=\max_{X\in S2}(\angle \overrightarrow{P_iX})$ (The *right* visible bound for P_i), $\mathrm{LV}(P_i)=\min_{Y\in S1}(\angle \overrightarrow{P_iY})$ (The *left* visible bound for P_i), and $V(P_i)=[\max_{X\in S2}(\angle \overrightarrow{P_iX}),\min_{Y\in S1}(\angle \overrightarrow{P_iY}\,]$. Figure 9 illustrates the angles from P_i to all points in $S1$ and $S2$, highlighting the visibility range $V(P_i)$.

Utilizing 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 $\overline{uv}$ defined by points u and v in $\boldsymbol{P}$, where $u=P_i$ and $v=P_{i+1}$. The intersection of visibility ranges for the points u and v and the 180 deg 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 180 deg range about the segment is done since visibility to the segment obviously cannot exist from any direction "behind" the segment. The 180 deg range about the segment is the set of angles: $[\angle\overrightarrow{uv},\angle\overrightarrow{vu}]$. In Fig. 10, the ranges are illustrated $([\mathrm{RV}_v,\mathrm{LV}_v],[\mathrm{RV}_u,\mathrm{LV}_u],[\angle\overrightarrow{uv},\angle\overrightarrow{vu}])$. The intersection of the visibility of u and the visibility of v will have bounds of RV_v and LV_u: $(V_u\cap V_v)=[\mathrm{RV}_u,\mathrm{LV}_u]\cap[\mathrm{RV}_v,\mathrm{LV}_v]\rightarrow[\mathrm{RV}_v,\mathrm{LV}_u]$. The sets $S1$ and $S2$ are thus redefined as $S1:[\mathrm{LCHP}(u),(u-1)]$ and $S2:[(v+1),\mathrm{RCHP}(v)]$. The ends of the visibility range are denoted $\mathrm{RV}(\overline{uv})$ and $\mathrm{LV}(\overline{uv})$, the *right* and *left* visibility bounds of the segment $\overline{uv}$, where: $\mathrm{RV}(\overline{uv})=[\max_{x\in S2}(\angle\overrightarrow{vx})]$ and $\mathrm{LV}(\overline{uv})=[\min_{y\in S1}(\angle\overrightarrow{uy})]$. Visibility to the segment $\overline{uv}$ is defined as: $V(\overline{uv}):[\mathrm{RV}(\overline{uv}),\mathrm{LV}(\overline{uv})]$ (Fig. 11).

Since not all surfaces will have a simple open pocket as shown in Fig. 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 $S1$ or the maximum angle to points in $S2$ is outside of the 180 deg range above the segment. In this case, RV or LV is set to the extremes of $[\angle\overrightarrow{uv},\angle\overrightarrow{vu}]$, either $(\angle\overrightarrow{uv})$ or $(\angle\overrightarrow{vu})$, 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 Fig. 12, problems occur from naively setting visibility to [RV, LV]. This can be avoided by investigating the characteristics of the pocket where the segment $\overline{uv}$ resides. The two points in $S1$ and $S2$ where the bounds RV and

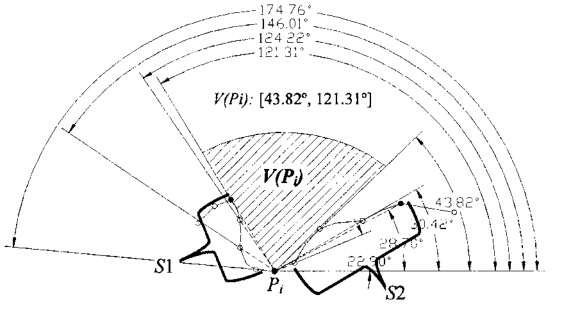


Fig. 9 Visibility of a point with respect to its chain

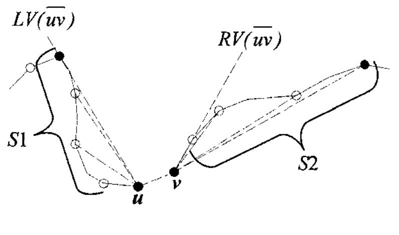


Fig. 11 Visibility range $[\mathrm{RV}(\overline{uv}),\mathrm{LV}(\overline{uv})]$

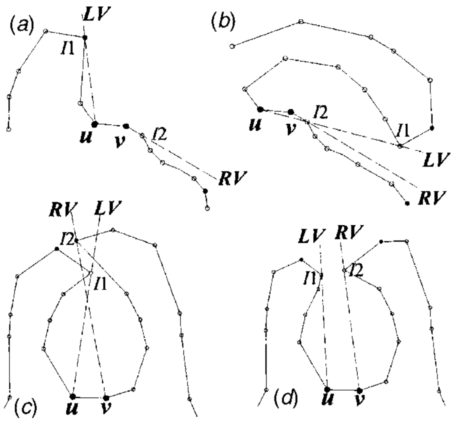


Fig. 12 Problem geometries (*a*) RV is outside the 180 deg range, (*b*) both RV and LV are outside the 180 deg range, (*c*) no visibility due to overlapping, and (*d*) visibility to entire segment not possible since RV>LV

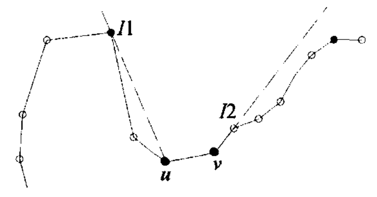


Fig. 13 Intersections in $S1(I1)$ and $S2(I2)$ to calculate RV and LV

LV are calculated are used and are denoted as $I1$ and $I2$, respectively; $I1 = P_x$, where $x = \arg\min_{Y_x \in S1}(\angle \overrightarrow{vY_x})$ and $I2 = P_y$, where $y = \arg\max_{X_y \in S2}(\angle \overrightarrow{uX_y})$ (Figs. 12 and 13).

The geometric relationships between $I1$, $I2$, u, and v can be used to determine if (i) the entrance to the pocket has an overlapping rim that makes visibility impossible, (ii) RV and/or LV, as calculated, are outside of the 180 deg range, or (iii) 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, $I1$, and $I2$ are employed to test for the three cases described above ($\overrightarrow{I2v} \otimes \overrightarrow{I2u}, \overrightarrow{I1v} \otimes \overrightarrow{I1u}, \overrightarrow{uI1} \otimes \overrightarrow{uI2}\ \overrightarrow{vI1} \otimes \overrightarrow{vI2}$).

The algorithm in Fig. 14 determines feasible values for LV and RV or whether visibility exists at all. Condition 3 in Fig. 14 can exist due to a long segment inside a pocket (see Fig. 15(a)). 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 Fig. 15(b), 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 (Fig. 15(c)).

```
IF $\overrightarrow{uI1} \otimes \overrightarrow{uI2} > 0$
  IF $\overrightarrow{vI1} \otimes \overrightarrow{vI2} > 0$
    $\overline{uv}$ is NOT visible   (Overlapping rim, Figure 12c)
IF $\overrightarrow{I1v} \otimes \overrightarrow{I1u} > 0$
  IF $\overrightarrow{I2v} \otimes \overrightarrow{I2u} < 0$
    $LV = \angle\overrightarrow{vu}$
  ELSE
    $\overline{uv}$ is NOT visible   (Both LV and RV not in 180° range, Figure 12b)
ELSE
  $LV = \angle\overrightarrow{uI1}$ (1)
IF $\overrightarrow{I2v} \otimes \overrightarrow{I2u} > 0$
  IF $\overrightarrow{I1v} \otimes \overrightarrow{I1u} < 0$
    $RV = \angle\overrightarrow{uv}$   (RV is outside range, LV is not, Figure 12a)
  ELSE
    $\overline{uv}$ is NOT visible
ELSE
  $RV = \angle\overrightarrow{vI2}$ (2)   (Check if visibility range is feasible, i.e., LV>RV)
IF (1) & (2)
  IF $(\angle I1, u, v) + (\angle I2, v, u) < 180°$
    $\overline{uv}$ is NOT visible (Figure 12d) (3)
```

Fig. 14 Algorithm 1: Determining values for LV and RV

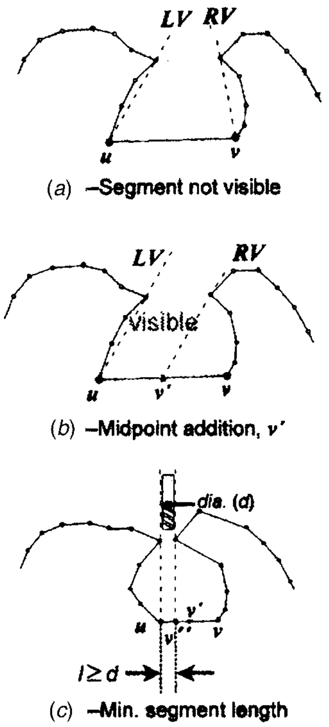


Fig. 15 Midpoint addition yields visibility to new segments

Another case of a problem geometry is a "spiral" pocket, as shown in Fig. 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 deg. 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*. An 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 (Fig. 17).

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 (Fig. 18). If u and v are both CHPs, then $\text{RV} = \angle\overrightarrow{uv}$ and LV

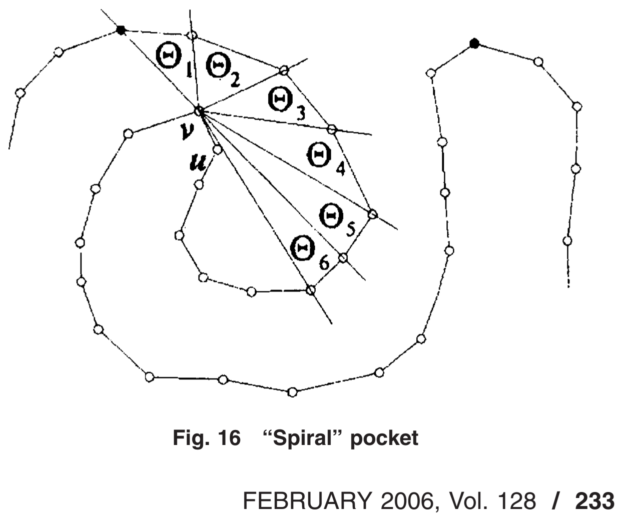


Fig. 16 "Spiral" pocket

```
$CurrentMin = \angle u(\overline{LCHP})$
$LV(\overline{uv}) = CurrentMin$
For all $x \in S1$:
 $\Theta_t = \angle \overrightarrow{x,u,x+1}$ for $x \in S1$
$CurrentMin = CurrentMin + \Theta_t(\overrightarrow{ux} \otimes \overrightarrow{u(x+1)})$
$Cumangle = Cumangle + \Theta_t(\overrightarrow{ux} \otimes \overrightarrow{u(x+1)})$
 IF $Cumangle \geq 180$
   $\overline{uv}$ is NOT VISIBLE
 ELSE
   IF $CurrentMin \leq LV(\overline{uv})$
     $LV(\overline{uv}) = CurrentMin$
```

Fig. 17 Algorithm 2: Detecting *spiral* pockets

$=\angle\overrightarrow{vu}$ (Fig. 18(*a*)). If u is a CHP and v is not, then RV$=\angle\overrightarrow{vI2}$ and LV$=\angle\overrightarrow{vu}$ (Fig. 18(*b*)). If v is a CHP and u is not, then RV $=\angle\overrightarrow{uv}$ and LV$=\angle\overrightarrow{uI1}$ (Fig. 18(*c*)).

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 $V(\overline{uv})$.

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 $V(\overline{uv})_{j^*}$ as the visibility with respect to the chain j on which $\overline{uv}$ resides, denoted j^*. The set of ranges for which $\overline{uv}$ is visible from the exterior will be called VIS$(\overline{uv})$ and represents the *global* visibility of the segment. It is calculated as the visibility of $\overline{uv}$ with respect to chain j^* minus the set of ranges blocked by other chains on the slice.

For all obstacle chains $j \in J\backslash(j^*)$, the polar range *blocked* by the chain is denoted VB$(\overline{uv})_j$; *Visibility blocked to the segment by another chain on the slice*. This set of visible ranges for the segment $\overline{uv}$ is defined as VIS$(\overline{uv})=V(\overline{uv})_{j^*}-\Sigma$VB$(\overline{uv})_j$ for all $j \in J\backslash(j^*)$. Visibility blocked to the segment $\overline{uv}$ 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 $\overline{uv}$ (Fig. 19). The set of angles blocked to the segment $\overline{uv}$ are VB$(\overline{uv})_j$ $=\{[[\text{VB}(u)_j]\cup[\text{VB}(v)_j]]\cap[\angle\overrightarrow{uv},\angle\overrightarrow{vu}]\}$. Where, the set of angles blocked to points u and v are VB$(u)_j=[\text{RB}_u,\text{LB}_u]$ (The *right* and *left* bounds of the range blocked to u by obstacle j) and VB$(v)_j$ $=[\text{RB}_v,\text{LB}_v]$ (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 $[\angle\overrightarrow{uv},\angle\overrightarrow{vu}]$ about the segment, then the union operation yields the following: $(\text{VB}_u\cup\text{VB}_v)$ $=[\text{RB}_u,\text{LB}_u]\cup[\text{RB}_v,\text{LB}_v]\rightarrow[\text{RB}_u,\text{LB}_v]$. 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 $\boldsymbol{P}_j$, where $\boldsymbol{P}_j$ is the set of points for the blocker chain (RB$_u=[\min_{x\in P_j}(\angle\overrightarrow{ux})]$ and LB$_v$ $=[\max_{y\in P_j}(\angle\overrightarrow{vy})]$). It must be determined whether the obstacle is partially or wholly within the range $[\angle\overrightarrow{uv},\angle\overrightarrow{vu}]$ about the segment in order to calculate the blockage range. Figure 20 illustrates various locations for the blocker chains. 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, $I1$, and $I2$. In this case, $I1$ and $I2$ are the points of intersection on the blocker chain for the vectors defining the bounds RB$_u$ and LB$_v$, respectively; $I1=P_x$, where x $=\arg\min_{Y_x\in P_j}(\angle uY_x)$ and $I2=P_y$, where $y=\arg\max_{X_y\in P_j}(\angle vX_y)$. 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 (Fig. 21). The algorithm in Fig. 22 determines the values for RB$_u$

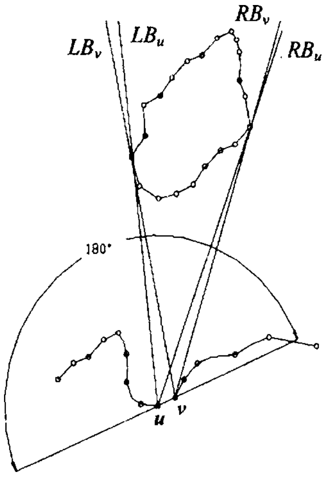
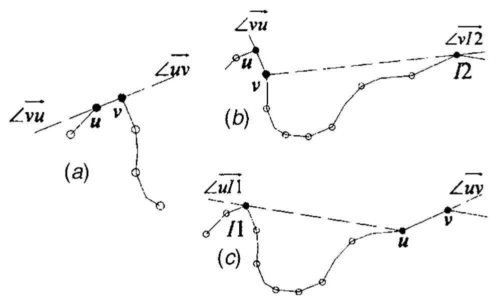


Fig. 19 Visibility blocked to $\overline{uv}$



$\angle\overrightarrow{vu}$
$\angle\overrightarrow{uv}$
u
v
(*a*)
$\angle\overrightarrow{vu}$
u
v
$\angle\overrightarrow{vI2}$
$I2$
(*b*)
$\angle\overrightarrow{uI1}$
$I1$
u
$\angle\overrightarrow{uv}$
v
(*c*)

Fig. 18 Cases where at least one point of segment $\overline{uv}$ is a convex hull point

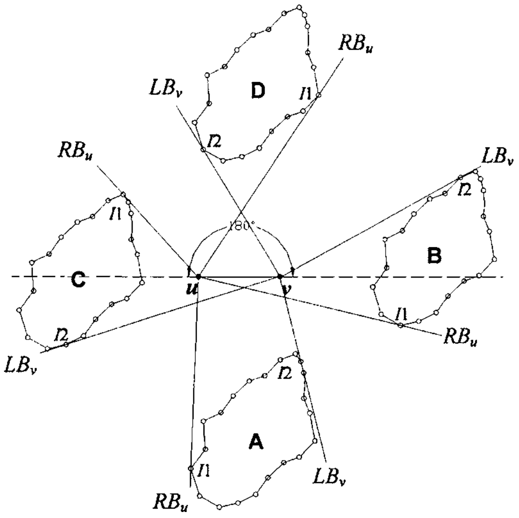


Fig. 20 Locations of blocker chains with respect to $\overline{uv}$

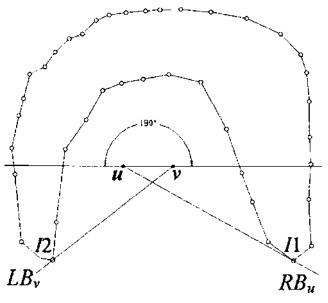


Fig. 21 "Horseshoe" obstacle envelopes the segment

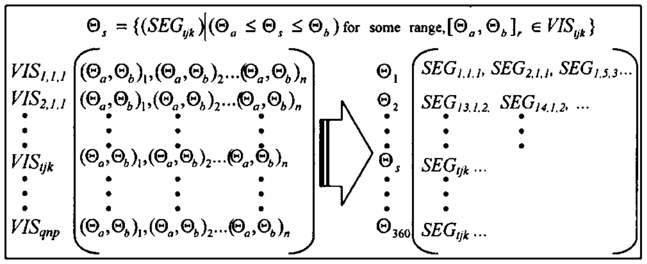
and LB_v.

At this point, all data is available for calculating the sets of *global* visibility ranges for each segment: $\text{VIS}(\overline{uv}) = V(\overline{uv})_{j^*} - \Sigma \text{VB}(\overline{uv})_j$ for all $j[j \in J \backslash (j^*)]$. However, it is possible that $\text{VIS}(\overline{uv}) = [\Phi]$ (*null* set) while $V(\overline{uv})_{j^*} \neq [\Phi]$, 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 Fig. 6.

The output of the visibility algorithm is the collection of visible ranges given in polar angle about the axis of rotation, as follows: $\text{VIS}_{tjk}:[\Theta_a,\Theta_b,]_1,[\Theta_a,\Theta_b,]_2,\ldots[\Theta_a,\Theta_b,]_r$ where: $r_{\max}=n$ (number of chains on slice k), t is the segment for $t \in T$ where $T=\{t|t=1,\ldots,q\}$, j is the chain, for $j \in J$ where $J=\{j|j=1,\ldots,n\}$, and 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_{tjk} exists for all segments on all chains for all slices of the surface geometry: $\text{VIS}_{tjk} \neq [\Phi]$ (*null set*), for all segments (t), 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.

```
IF $\overrightarrow{vu} \otimes \overrightarrow{vI1} > 0$
  IF $\overrightarrow{vu} \otimes \overrightarrow{vI2} > 0$
    IF $\overrightarrow{vI2} \otimes \overrightarrow{vI1} > 0$
      $RB_u = \angle\overrightarrow{uv}$  ("Horseshoe",
      $LB_v = \angle\overrightarrow{vu}$   Figure 21)
  ELSE
  Blocker outside range  (Figure 20,
                          Location A)
IF $\overrightarrow{vu} \otimes \overrightarrow{vI1} > 0$
  IF $\overrightarrow{vu} \otimes \overrightarrow{vI2} < 0$
      $RB_u = \angle\overrightarrow{uv}$  (Figure 20,
      $LB_v = \angle\overrightarrow{vI2}$  Location B)
IF $\overrightarrow{vu} \otimes \overrightarrow{vI1} < 0$
  IF $\overrightarrow{vu} \otimes \overrightarrow{vI2} > 0$
      $RB_u = \angle\overrightarrow{uI1}$  (Figure 20,
      $LB_v = \angle\overrightarrow{vu}$   Location C)
IF $\overrightarrow{vu} \otimes \overrightarrow{vI1} < 0$
  IF $\overrightarrow{vu} \otimes \overrightarrow{vI2} < 0$
      $RB_u = \angle\overrightarrow{uI1}$  (Figure 20,
      $LB_v = \angle\overrightarrow{vI2}$  Location D)
```

Fig. 22 Algorithm 3: Determining values for RB_u and LB_v



$\Theta_s = \{(SEG_{tjk})|(\Theta_a \leq \Theta_s \leq \Theta_b)$ for some range, $[\Theta_a,\Theta_b]_r \in VIS_{tjk}\}$
$VIS_{1,1,1}$ $(\Theta_a,\Theta_b)_1,(\Theta_a,\Theta_b)_2\ldots(\Theta_a,\Theta_b)_n$
$VIS_{2,1,1}$ $(\Theta_a,\Theta_b)_1,(\Theta_a,\Theta_b)_2\ldots(\Theta_a,\Theta_b)_n$
VIS_{tjk} $(\Theta_a,\Theta_b)_1,(\Theta_a,\Theta_b)_2\ldots(\Theta_a,\Theta_b)_n$
VIS_{qnp} $(\Theta_a,\Theta_b)_1,(\Theta_a,\Theta_b)_2\ldots(\Theta_a,\Theta_b)_n$
Θ_1 $SEG_{1,1,1}, SEG_{2,1,1}, SEG_{1,5,3}\ldots$
Θ_2 $SEG_{13,1,2}, SEG_{14,1,2}, \ldots$
Θ_s $SEG_{tjk}\ldots$
Θ_{360} $SEG_{tjk}\ldots$

Fig. 23 Mapping of visible segments to orientations

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, 360 deg) can be discretized arbitrarily based on a reasonable step size for axis rotations. For illustration, we will assume the step size is of 1 deg. From the data in $\{\text{VIS}\}_{tjk}$, one can formulate a set corresponding to the segments visible from a given angle. In other words, out of the complete set of segments {SEG} some SEG_{tjk} are visible from a given angle Θ_S. The set $\{\Theta\}$ contains the segments visible from each angle Θ_S where: $s=1-360$ deg (Fig. 23).

The formulation of the *minimum set cover* problem is straightforward: *Given:* A collection of subsets Θ_s of a finite set {SEG} *(the set of all segments). Solution:* A set cover for {SEG} i.e., subset $S' \subseteq S$ such that every element in {SEG} belongs to at least one member of Θ_s for $s \in S'$. It is noted that the minimum set cover problem is NP-Hard therefore a greedy heuristic was employed to achieve an approximately optimal solution quickly [36].

Implementation

The visibility algorithms were implemented in C and tested on a Pentium IV, 2.0 Ghz PC, running Windows XP. The software accepts slice files as input and returns several critical process parameters: (i) the minimum number of orientations, (ii) the minimum stock diameter, and (iii) 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 Fig. 24(*a*), *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 Fig. 24(*b*).

Several trials were run to evaluate the computation time for the visibility algorithm. Computation time is dependent on two main factors: (i) the distance between successive cross sectional slices (i.e., the number of slices) and (ii) 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 3D model; a finer STL file and smaller slice spacing achieves a better approximation of the surface. One must consider the spacing of the slices that should be used. Of course, an infinitely thin slice spacing approaches the true 3D shape of the geometry; however, the computation times would explode.

Reasonable spacing for slices can be considered based on the positional accuracy of the CNC machine and/or reasonable assumptions about the geometry of the part features. The issue is

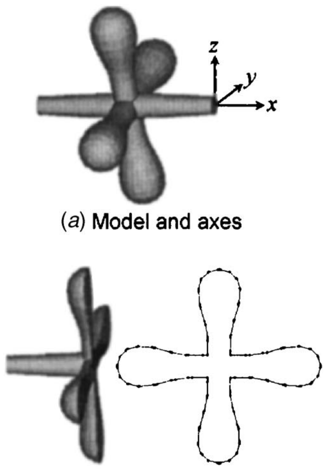


Fig. 24 STL model and sample cross section

whether features of the part model will be missed if the slice spacing is too large. Take, for example, a part with a small hole feature. If the slice spacing is larger than the diameter of the hole, it is likely that the geometry of the hole will not be considered during the visibility mapping. However, if a minimum diameter tool is known, then one would not need to reduce the slice spacing to less than the minimum tool diameter. Even if the hole were missed during the slicing operation, if the minimum tool diameter is greater than the hole diameter, then it would not be able to be completely machined regardless. If the feature is a protrusion from the part surface, then the feature could have dimensions less than the minimum tool diameter (i.e., a thin web feature) and still be machinable. In the example of the thin web, one could assume that any feature less than a particular web "thickness" would be unmachinable due to problems with material deflection. In this case, the upper bound for slice spacing could be set equal to the thinnest web thickness that is considered machinable given the material used. 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.

Table 1 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 in. (0.0635 mm) between each slice, up to 0.040 in. (1.016 mm). 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 Fig. 25. A second example is a model of the human femur bone with an axis of rotation selected along the length of the bone. The visibility algorithm determined that the bone was 99.97% visible and completed the visibility mapping in 9.4 s. A final example is of a model turbine blade. Visibility mapping required 7.4 s, and it was determined that the turbine is 91.29% visible using its designed axis of rotation. The low-visibility result should be expected, since the cylindrical hub of the turbine would obviously not be visible from the circumference of the part. For each of these models the greedy solutions are illustrated in Fig. 26. One will note the two orientations grouped beside each other on the right side of the turbine blade (Fig. 26(*b*)). This is because the greedy heuristic does not guarantee optimality and the extra orientation might be avoided if a new seed orientation were chosen; however, optimizing the results is outside the scope of the current work.

Once the set of orientations is known, toolpath plans can be generated using CAM software using the orientation angles, max/min depths of cut, and stock diameter from the visibility algorithms. Several 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 following steps were executed: (i) visibility software is executed; (ii) the CAD model, oriented about the intended axis, is rotated through each of the orientations of the visibility solution; (iii) the toolpath containment boundary is created using the stock and tool diameter, and the length of the part; (iv) for each orientation, *rough surface pocket* toolpaths (MasterCAM) are generated with minimum depth at the stock radius and maximum depth set to the parameters given from the visibility software; and (v) NC code for each

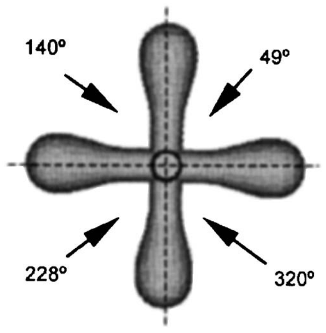


Fig. 25 Orientations required for machining

Table 1 Process times for visibility algorithm

	STL resolution					
	Coarse		*Medium*		*Fine*	
C.H.	0.0075″		0.0025″		0.000625″	
facets	865		1990		6578	
Slice (in)	*No. segments*	*time*(*s*)	*No. segments*	*time* (*s*)	*No. segments*	*time* (*s*)
0.0025	19 566	22.750	36 199	29.390	69 212	47.122
0.0050	9 772	11.230	18 178	14.671	34 458	23.389
0.0100	4 850	5.687	9 054	7.405	17 306	11.843
0.0200	2 375	2.875	4 597	3.907	8 683	6.281
0.0400	1 182	1.453	2 159	2.032	4 123	3.141

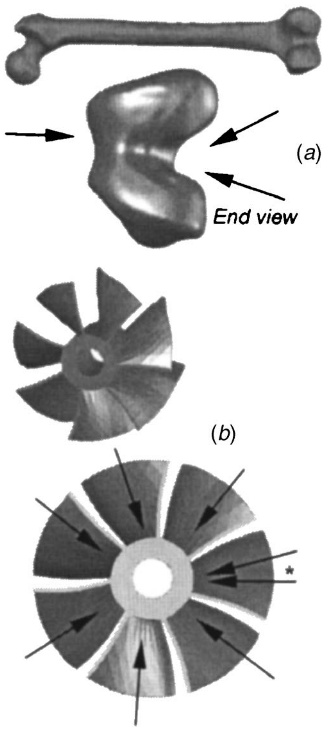


Fig. 26 Sample models and visibility map results

orientation is combined manually into a file with fourth-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 four. 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 three-jaw chuck, was not available at the time of testing.) The material used was 6061 Aluminum in the form of 1.375 in. (35 mm) bar stock. Layer thickness was set at 0.005 in. (127 m) while the spindle speed and feed rate were set to 7500 rpm and 350 ipm (8890 mm/min), respectively (limits of the machine). A 1/8 in. (3.175 mm) flat-end mill at 1.5 in. (38.1 mm) length was used. The part was created in ~3 h. Sacrificial supports were added to the ends of the model (0.1 in. (2.54 mm) diam cylinders). Figure 27(a) shows the prototype of the jack in between machining operations, and the finished part is shown in Fig. 27(b).

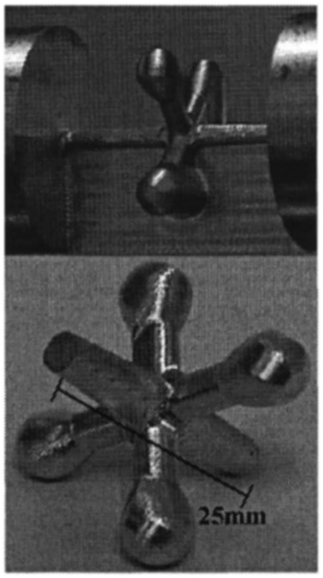


Fig. 27 Example prototype: "The jack"

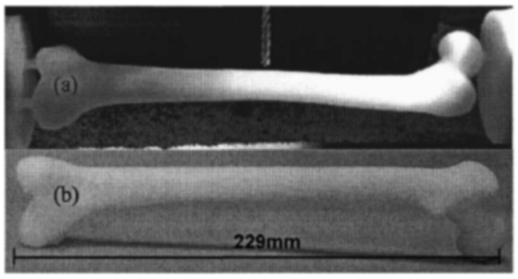


Fig. 28 Example prototype: Femur bone

The next model is of a half-scale human leg bone, the femur, which was machined from 2 in. (50.8 mm) round Delrin (Acetal) stock material on a Fadal VM15. The spindle speed was 7500 rpm and the feed rate was 300 ipm (7620 mm/min). A 1/8 in. (3.175 mm) flat-end mill was used and the layer thickness was 0.003 in. (76 m). Figure 28(a) shows the femur bone model still attached to the stock via three sacrificial supports while Fig. 28(b) is the completed model. Total machining time was ~12 h.

Conclusions

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 $2\frac{1}{2}$-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 that are required for rapid machining. This method is also a significant improvement over existing methods because it modifies the representation of the slice geometry in an effort to seek a feasible visibility solution.

References

[1] Wang, F. C., Marchetti, L., and Wright, P. K., 1999, "Rapid Prototyping Using Machining," SME Technical Paper, PE99-118.
[2] Frank, M. C., Wysk, R. A., and Joshi, S. B., 2003, "Rapid Prototyping Using CNC Machining," *Proc. of ASME Design Engineering Technical Conf. and Computers and Information in Engineering Conf., Chicago*, IL, ASME, New York.
[3] Frank, M. C., Joshi, S. B., and Wysk, R. A., 2003, "Rapid Prototyping as an Integrated Product/Process Development Tool: An Overview of Issues and Economics," J. Chin. Inst. Eng., **20**(3), pp. 239–245.
[4] Chen, L. L., and Woo, T. C., 1992, "Computational Geometry on the Sphere With Application to Automated Machining," Trans. ASME, **114**(2), pp. 288–295.
[5] Gan, J. G., Woo, T. C., and Tang, K., 1994, "Spherical Maps: Their Construction, Properties, and Approximation," J. Mech. Des., **116**(2), pp. 357–363.
[6] Tang, K., Woo, T. C., and Gan, J., 1992, "Maximum Intersection of Spherical Polygons and Workpiece Orientation for 4- and 5-Axis Machining," J. Mech. Des., **114**(3), pp. 477–485.
[7] Chen, L., Chou, S., and Woo, T. C., 1993, "Separating and Intersecting Spherical Polygons: Computing Machinability on Three-, Four-, and Five-Axis Numerically Controlled Machines," ACM Trans. Graphics, **12**(4), pp. 305–326.
[8] Suh, S. H., and Kang, J. K., 1995, "Process Planning for Multi-Axis NC Machining of Free Surfaces," Int. J. Prod. Res., **33**(10), pp. 2723–2738.
[9] Joshi, S. B., and Chang, T. C., 1988, "Graph-Base Heuristic for Recognition of Machined Features From a 3-D Solid Model," Comput.-Aided Des., **20**(2), pp. 58–66.
[10] Ferreira, J. C. E., and Hinduja, S., 1990, "Convex Hull-Based Feature Recognition Method for 2.5D Components," Comput.-Aided Des., **22**(1), pp. 41–49.
[11] Vanderbrande, J. H., and Requicha, A. A. G., 1993, "Spatial Reasoning for the Automatic Recognition of Machinable Features in Solid Models," IEEE Trans. Pattern Anal. Mach. Intell., **15**(12), pp. 1269–1285.
[12] Balasubramanium, M., Laxmiprasad, P., Sarma, S., and Shaikh, Z., 2000,

"Generating 5-Axis NC Roughing Paths Directly From a Tessellated Representation," Comput.-Aided Des., **32**(4), pp. 261–277.
[13] Tseng, Y. J., and Joshi, S. B., 1991, "Determining Feasible Tool-Approach Directions for Machining Bezier Curves and Surfaces," Comput.-Aided Des., **23**(5), pp. 367–379.
[14] Glaeser, G., Wallner, J., and Pottman, H., 1999, "Collision Free 3-Axis Milling and Selection of Cutting Tools," Comput.-Aided Des., **31**(3), pp. 225–232.
[15] Yang, D. C. H., and Han, Z., 1999, "Interference Detection and Optimal Tool Selection in 3-Axis NC Machining of Free-Form Surfaces," Comput.-Aided Des., **31**(5), pp. 303–315.
[16] Suh, S. H., Lee, J. J., and Kim, S. K., 1998, "Multiaxis Machining With Additional-Axis NC System: Theory and Development," Int. J. Adv. Manuf. Technol., **14**(12), pp. 865–875.
[17] Suh, S., and Lee, J., 1998, "Five Axis Part Machining With Three Axis CNC Machine and Indexing Table," ASME J. Manuf. Sci. Eng., **120**(1), pp. 120–128.
[18] Li, S., and Jerard, R., 1994, "5-Axis Machining of Sculptured Surfaces With a Flat-End Cutter," Comput.-Aided Des., **26**(3), pp. 165–178.
[19] Xu, X. J., Bradley, C., Zhang, Y. F., Loh, H. T., and Wong, Y. S., 2002, "Tool-Path Generation for Five-Axis Machining of Free-Form Surfaces Based on Accessibility Analysis," Int. J. Prod. Res., **40**(14), pp. 3253–3274.
[20] Balasubramanium, M., Sarma, S., and Marciniak, K., 2003, "Collision-Free Finishing Toolpaths From Visibility Data," Comput.-Aided Des., **35**(4), pp. 359–374.
[21] Lee, D. T., 1983, "Visibility of a Simple Polygon," Comput. Vis. Graph. Image Process., **22**(2), pp. 207–221.
[22] Shin, S. Y., and Woo, T. C., 1989, "An Optimal Algorithm for Finding all Visible Edges in a Simple Polygon," IEEE J. Rob. Autom., **5**(2), pp. 202–207.
[23] Ghosh, S. B., and Mount, D. M., 1991, "An Output-Sensitive Algorithm for Computing Visibility Graphs," SIAM J. Comput., **20**(5), pp. 888–910.
[24] Gewali, L. P., and Ntafos, S., 1998, "Watchman Routes in the Presence of a Pair of Convex Polygons," Inf. Sci. (N.Y.), **105**(1–4), pp. 123–149.
[25] Everett, H., Hurtado, F., and Noy, M., 1999, "Stabbing Information of a Simple Polygon," Discrete Appl. Math., **91**(1–3), pp. 67–82.
[26] Kapoor, S., and Maheshwari, S. N., 2000, "Efficiently Constructing the Visibility Graph of a Simple Polygon With Obstacles," SIAM J. Comput., **30**(3), pp. 847–871.
[27] Ntafos, S., and Gewali, L., 1994, "External Watchman Routes," Visual Comput., **10**(8), pp. 474–483.
[28] Laurentini, A., 1999, "Guarding the Walls of an Art Gallery," Visual Comput., **15**(6), pp. 265–278.
[29] O'Rourke, J., 1987, *Art Gallery Theorems and Algorithms*, Oxford University Press, New York.
[30] Peshkin, M. A., and Sanderson, A. C., 1986, "Reachable Grasps on a Polygon: The Convex Rope Algorithm," IEEE J. Rob. Autom., **RA-2**(1), pp. 53–58.
[31] Lee, D. T., and Preparata, F. P., 1984, "Euclidean Shortest Paths in the Presence of Rectilinear Barriers," Networks, **14**(3), pp. 393–410.
[32] Guibas, L. J., Hershberger, J., Leven, D., Sharir, M., and Tarjan, R. E., 1987, "Linear Time Algorithms for Visibility and Shortest Path Problems Inside Triangulated Simple Polygons," Algorithmica, **2**(2), pp. 209–233.
[33] Stewart, J. A., 1999, "Computing Visibility From Folded Surfaces," Comput. Graph., **23**(5), pp. 693–702.
[34] Toussaint, G., 1991, "Efficient Triangulation of Simple Polygons," Visual Comput., **7**(5–6), pp. 280–295.
[35] Held, M., 1998, *Efficient and Reliable Triangulation of Polygons*, Computer Graphics International, Hannover, Germany, pp. 633–643.
[36] Chvátal, V., 1979, "A Greedy Heuristic for the Set-Covering Problem," Math. Op. Res., **4**(3), pp. 233–235.