RADIAL RAY REPRESENTATION FOR FAST ANALYSIS OF OPTIMAL CUTTING DIRECTION IN 3+2 AXIS MILLING

Masatomo Inui  
Dept. of Intelligent Systems Engr., Ibaraki Univ.  
Nakanarusa 4-12-1  
Hitachi, Ibaraki 316-8511, JAPAN  
masatomo.inui.az@mx.ibaraki.ac.jp

Shinji Nagano  
Dept. of Intelligent Systems Engr., Ibaraki Univ.  
Nakanarusa 4-12-1  
Hitachi, Ibaraki, 316-8511, JAPAN  
17nm934r@vc.ibaraki.ac.jp

Nobuyuki Umezu  
Dept. of Intelligent Systems Engr., Ibaraki Univ.  
Nakanarusa 4-12-1  
Hitachi, Ibaraki, 316-8511, JAPAN  
nobuyuki.umezu.cs@vc.ibaraki.ac.jp

ABSTRACT
In this paper, we propose a novel 3D shape representation method called “radial ray representation (3R)”. In this method, a solid shape is recorded by a set of dense rays radially expanding from a single point. This representation is especially useful for determining the optimal cutter posture in the 3+2 axis milling of a mold part. As a measure of the appropriateness of the cutter posture, the peak angle of the accessibility cone (AC) is used. In the appropriate postures, we select the optimal one as the tool posture with possible shortest cutter length without holder collisions. We developed 3R model based algorithms for computing the AC and for selecting the optimal cutter posture. We also propose a fast method for obtaining a 3R model using the polygon rendering function of the graphics processing unit. An experimental system is implemented and some computation results are demonstrated.

INTRODUCTION
In the automobile industry, molds with very deep shape are necessary for producing large plastic parts, such as instrument panels and bumpers. In the usual 3-axis milling, cutters with long shank is necessary to avoid collisions between the holder and the mold (see Fig. 1(a)). Since large deformation of the cutter is unavoidable with a long shank, it is difficult to realize stable machining in the 3-axis milling. To solve this problem, many manufacturers in Japan use 3+2 axis milling for the semi-finishing in the mold machining. In this method, the machine executes a 3-axis milling program with a cutter locked in a tilted position using its 2 rotational axes. In the 3+2 axis milling, shank length can be reduced by properly selecting the cutter posture as shown in Fig. 1(b). Different from the simultaneous 5-axis milling, the cutter in a fixed posture is more rigid so much accurate machining results can be realized.
milling machine (see Fig. 1(b)), a cone whose axis is coaxial to the spindle axis of the cutter and smoothly contacting the spherical part of the cutter is considered. The peak angle of the cone is enlarged until the cone touches the mold surface as shown in Fig. 2. Such cone with the maximum peak angle is called accessibility cone (AC) of the cutter at \( p \) in posture \((\phi, 0)\). AC represents the angular clearance between the mold and the cutter at a specific configuration (position and posture). To safely execute 3+2 axis milling in a specific posture, its corresponding AC must have a sufficiently large peak angle.

In the 3+2 axis milling, collision between the tool holder and the workpiece is another critical factor in the cutter posture determination. A cutter with shorter length is generally preferable for achieving higher milling accuracy, however collision between the tool holder and the workpiece is liable to occur. In this paper, we discuss an algorithm for computing the optimal cutter posture for the 3+2 axis milling. Our algorithm determines the optimal posture in a 2-step manner. In the first step, appropriate cutter postures are selected using their associating ACs. The optimal posture is selected from the appropriate ones in the second step as a posture with possible shortest cutter length without holder collisions.

To realize efficient computations of the AC and the selection of the optimal cutter posture, we propose a novel 3D shape representation method named “radial ray representation (3R)” where a solid shape is recorded by a set of dense rays radially expanding from a single point. In the next section, prior studies relating to the cutter posture determination in the 3+2 axis milling are briefly reviewed. In Section 3, the basic concept of 3R solid model and a conversion method from the usual boundary representation model to 3R model is explained. The outline of our AC computation algorithm using 3R model is illustrated in section 4. A method for selecting the optimal cutter posture with possible shortest cutter length is also given. In section 5, we propose a fast method for obtaining a 3R model for the optimal cutter posture determination using the polygon rendering function of the graphics processing unit (GPU). Experimental computation results are given in Section 6, and we summarize our conclusions in Section 7.

RELATED STUDIES

Van Hook initiated the ray representation for recording a solid shape, though he used a term “dexels” in the meaning of rays in his milling simulation system [3]. Menon et al. proposed the ray representation as a new solid shape representation framework in [4]. In the ray representation, object shape is recorded by a bundle of z-axis-aligned segments defined for each grid point of a square mesh in the xy-plane. In this representation, surfaces near-parallel to the ray direction (= z-axis) have inevitable large shape errors caused by the finite grid resolution. The triple-ray (or triple-dexel) model was proposed to overcome this non-uniformity of the representation accuracy [5]. In the usual ray representation, the object shape is recorded by using set of segments parallel to a specific axis direction. On the other hand, our radial ray representation records the object shape using a set of rays radially expanding from a single point.

Many research results are known for automatically determining the cutter optimal posture in the 3+2 axis milling [6]. Takeuchi et al. proposed a trial-and-error-based method for computing collision free cutter postures in the 5-axis milling [7, 8]. Morishige et al. developed a method for determining collision free cutting postures in the 5-axis milling using a C-space of the cutter posture [9, 10]. Kaneko et al. introduced the GPU technology for accelerating Morishige’s algorithm [11].

Determination of the collision-free cutter posture is related to the accessibility problem of a point to a certain region on the offset surface of the workpiece. The “visibility cone” is defined as the feasibility range of the cutter posture \((\phi, 0)\) for milling a surface point. Tseng and Joshi [12] and Kang and Suh [13] used the visibility cone to determine the cutter accessibility in the 5-axis milling. Spitz and Requicha developed a computation method of a visibility cone for the coordinate measurement machine using the perspective projection [14]. Morimoto and Inui extended Spitz and Requicha’s method for determining the cutter accessibility in the 3+2 axis milling [2].

In the computation of the accessibility cone, the offset surface of the object is necessary. Conventional techniques for offsetting 3D objects [15, 16, 17] are often computationally expensive, and its model reconstruction process can be unstable. Since a picture of the offset shape is only required in the perspective-projection-based computation of the AC [2], much simple and robust method is applicable. The picture of the offset shape of a polyhedral object can be obtained by rendering spheres, cylinders and thick plates (slabs) placed on the object surface [18].

![Figure 3: Radial ray representation solid model.](image)

RADIAL RAY REPRESENTATION (3R) SOLID MODEL

In the radial ray representation, object shape is defined by a set of rays radially expanding from a single point \( o \) as shown in Fig. 3(a). Position of the ray starting point \( o \) is variable according to the purpose of using the model. In our cutter posture determination purpose, the point is fixed somewhere in a space above the object surface. Details of the point positioning method is given in the following section.

An axis aligned cubic box is defined so that the center of the box is coincident to the ray starting point. The size of the box is determined so that the box completely contains the object within. For each square face of the bounding cube, a uniform square mesh is defined as shown in Fig. 3(b). Representation accuracy of the 3R models is highly depend on the mesh resolution. Higher resolution mesh gives accurate result, however much memory is consumed in the representation. In our cutter posture determination purpose, 768 x 768 resolution mesh generally gives good result.
For each grid point in the mesh, a ray expanding from \( o \) and connecting to the grid point is defined. For each ray, the intersection portions between the ray and the object are computed and intersecting segments are derived. Red segments in Fig.3 correspond to the intersecting segments. This intersection computation is repeated for all grid points of 6 meshes on the cube, and a 3R solid model of the object is obtained as a set of radial line segments.

**DETERMINATION OF OPTIMAL CUTTER POSTURE**

The input data of our algorithm consists of a polyhedral model that approximates the mold shape, radius \( r \) of a ball end cutter for milling, and a set of points representing the cutter positions in the milling operation. Most commercial CAD systems provide a function to output the model data as a group of triangular polygons, such as in the STL format. Cutter position data are obtained by using a conventional CAM system for 3-axis milling. We assume that the cutter posture \((\phi, 0)\) can be fixed in every one degree in a range between 0 to 360 degree for \( \phi \) and in a range between 0 to 90 degree for \( 0 \).

The output of the algorithm is a set of appropriate cutter postures with their corresponding ACs, necessary cutter length for milling operation without collisions for all appropriate postures, and the optimal cutter posture with the shortest cutter length.

**Computation of Accessibility Cones (Step 1)**

In the first step of the algorithm, AC is computed for each cutter posture and the appropriate postures with sufficiently large AC are derived. In the computation, a thin cutter of zero-radius and an expanded mold shape obtained by offsetting the mold surface by the cutter radius \( r \) (see Fig. 4). Point \( p_0 \) representing the cutter position in the milling locate on the offset surface. An AC for a normal cutter with respect to the mold shape and another AC for the zero-radius cutter with respect to the expanded mold shape have equal peak angle as shown in the figure. In the following discussion, we explain the AC computation based on the zero-radius cutter and the offset shape of the mold part using Morimoto and Inui’s method [2].

Consider a problem to judge whether a point \( p_0 \) is machinable with a zero-radius cutter in a certain posture. This cutter accessibility analysis can be achieved by using the perspective projection in the 3D computer graphics. A viewing point is placed on \( p_0 \) and a displaying screen in a certain background color is prepared in a sufficient distance from \( p_0 \). In Fig. 5, green is assigned as the background color. The offset shape of the mold part is rendered using the perspective projection. In the rendered image, regions in the background color represents a set of end points of segments corresponding to the zero-radius cutters machinable at \( p_0 \).

**Figure 5: Definition of a visibility pyramid and the determination of two ACs using the pyramid.**

Consider a “visibility pyramid” whose bottom face is the background color region in the display and whose peak point corresponds to \( p_0 \) (see Fig. 5). For each cutter posture \((\phi, 0)\) allowed in the ranges for \( \phi \) and \( 0 \), a straight line starting from \( p_0 \) and being extended in the cutter axis direction is checked. If the line reaches the bottom face of the pyramid, then the cutter in this posture is accessible to \( p_0 \). For each accessible line, a cone whose peak point is at \( p_0 \) and coaxial to the line is considered. The peak angle of such cone is enlarged until the cone surface touches some side faces of the visibility pyramid as shown in the figure. Obtained cone with the maximum peak angle allowed in the visibility pyramid becomes the AC for the cutter posture.

In our AC computation algorithm, the offset shape of a mold part must be rendered. Picture of the offset shape of a polyhedral object is obtained by placing spheres of radius \( r \), cylinders of radius \( r \), and slabs of thickness \( 2r \) on vertices, edges and triangles of the object, respectively and rendering them [18]. In the 3D computer graphics, the necessary cost for rendering an object is basically proportional to the number of polygons of the object. Since spheres and cylinders of the offset shape are finely tessellated before the rendering, their rendering cost (especially rendering cost of tessellated spheres) dominates the total rendering cost of the offset shape.

Figure 6 illustrates a rendering operation of the offset shape of a polyhedron with 8 vertices. For each vertex, a sphere of radius \( r \) (circle in the figure) is placed (see Fig. 6(a)). Viewing point for the perspective projection is placed on a point \( p_0 \). In the perspective projection, a viewing frustum is given to limit the visible range in the display. Light blue region in Fig. 6(b) corresponds to the viewing frustum. Visible portion of the offset surface is specified by red curves. Visibility pyramid (green shape in the figure) is constructed based on such visible surfaces. As shown in the figure, most spheres of the offset shape do not contribute the final image.
Figure 6: A polyhedron with 8 vertices and its offset shape (a). Visibility pyramid obtained by rendering the offset surface (b). Visibility pyramid obtained by using offsetting result of points in the visible surface (c).

Figure 7: 3R model of a part. The endpoints of the segments nearest to \( p_0 \) correspond to the visible points from \( p_0 \) in the perspective projection.

We developed a method named “visible surface offsetting” for reducing the rendering of such non-contributing spheres. In this method, a set of visible points on the part shape (not its offset shape) in the perspective projection is computed first. Red curves in Fig. 6(c) correspond to the visible points. Spheres of radius \( r \) are placed on such visible points in the display and their image is rendered by using the perspective projection. Visibility pyramid is finally constructed based on the rendering result of the spheres as shown in Fig. 6(c). Obtained pyramid shape is identical to the pyramid given in (b). In this method, the number of spheres to render is limited to the visible portion of the object surface. For a simple case as shown in Fig. 6, the difference of numbers of rendering spheres between (b) and (c) is small, however it becomes large for part with many polygons. See our original paper for detail analysis on the performance [1].

Solid model of a part in the radical ray representation (3R) is suitable for the visible surface offsetting. Fig. 7 shows a 3R model of the part where milling point \( p_0 \) is selected as the ray starting point. For each ray, the endpoint of the segments on the ray nearest to \( p_0 \) corresponds to the visible points of the object. Red points in the figure represents these endpoints. Visible surface offsetting is realized by placing the spheres on such visible endpoints and by rendering them in the perspective projection.

Figure 8: Definition of a holder shape.

Figure 9: Shortest cutter length determination method.

Computation of Cutter Length (Step 2)

After selecting the appropriate cutter postures using the AC, the optimal posture with possible shortest cutter length is determined in step 2. In the following algorithm, a cutter holder is assumed to have a single truncated cone shape as shown in Fig. 8(a). Actual holder has a shape of series of truncated cones (see Fig. 8(b)). Extension of the algorithm for handling such cutter holder with complex shape is straight forward.

Three parameters \( r_0, r_1 \) and \( t \) are used for defining the holder shape. \( r_0 \) and \( r_1 \) respectively represent the radii of the bottom and top side discs of the holder and \( t \) represents its thickness. \( r_1 \) is larger than \( r_0 \) or equal to \( r_0 \). Parameter \( l \) represents the cutter length, which is the distance between the cutter tip and the bottom side of the holder.

For each appropriate posture \((\phi, \theta)\), our algorithm determines the shortest cutter length \( l_{\text{min}} \) allowed for the holder without collisions between the holder and the mold shape. Consider a cutter with a holder and a mold part solid model in the radial ray representation. Since an AC is defined for a cutter in an appropriate posture \((\phi, \theta)\), we can place the cutter at \( p_0 \) without having holder collisions if \( l \) is sufficiently large. Our algorithm determines the minimum \( l \) value by reducing \( l \) to find out the first colliding point between the holder and the mold part (see Fig. 9).

The mold shape is represented by segments on the rays expanding from the ray starting point \( p_0 \), therefore possible collisions between the holder approaching to \( p_0 \) and the mold part can be classified to following two cases;
Figure 10: Possible collision between a holder and segments of 3R model of the part.

Figure 11: Cutter posture determination for cutting multiple points.

Case 1: Bottom disc or side conical surface of the holder collide to the endpoints of the segments of the mold part model. For each endpoint of the segments, its distance $d$ from the center axis of the cutter and radius $r_0$ and $r_1$ are compared (see Fig. 10(a)).
- If $d < r_0$ or $d = r_0$, then collision between the endpoint and the bottom side disc is checked.
- If $r_0 < d$ and $d < r_1$, then collision between the endpoint and the conical surface part of the holder is checked.

Case 2: Bottom disc or top disc of the holder collide to the middle point of a segment of the mold part model. Compare the apex angle $\alpha$ of the conical surface of the holder and angle $\beta$ between the segment and the cutter axis;
- If $\alpha > \beta$, then the collision between the boundary circle of the top side disc and the segment is checked (Fig. 10(b)).
- If $\alpha < \beta$, then the collision between the boundary circle of the bottom side disc and the segment is checked (Fig. 10(c)).

For each collision detection mentioned above, its corresponding cutter length $l$ is computed. Their largest value corresponds to the shortest cutter length $l_{\text{min}}$ allowed for the holder in a cutter posture ($\phi, \theta$). This computation is repeated for all appropriate cutter postures to determine the optimal cutter posture with the shortest cutter length.

The cutter length computations for each cutter posture are mutually independent; therefore, they can be performed using GPU in a parallel manner. The GPU is designed to have hundreds of small streaming processors (SPs) on a chip. These SPs can execute the same instructions with different data in parallel. In the implementation of the parallel cutter length computation software, Compute Unified Device Architecture (CUDA) is used [14].

Figure 12: Construction of 3R model using the depth buffer mechanism.

OPTIMAL CUTTER POSTURE FOR MULTIPLE MILLING POINTS

Consider the optimal cutter posture determination acceptable for multiple points given as the cutter positions. This problem can be converted to the optimal cutter posture determination for a single point. Select a representative point from the point set. In Fig. 11(a), five red points $p_0, p_1, p_2, p_3, p_4$ represent the cutter positions. In these points, $p_2$ is selected as a representative point. For each point in the cutting positions, a vector from the point to the representative point is computed and stored (see arrows in Fig. 11(a)). Before the optimal cutter posture computation, the mold part shape is translated by each stored vector, then Boolean union shape of all translated models is computed. Fig. 11(b) shows Boolean union shape of five translated models. Now the original model with five cutting points is converted to a cutting problem of the union shape at a single cutting position $p_2$.

In this method, Boolean union shape of translated part models is necessary. Input polyhedral model is translated by each stored vector, then it is converted to a model in the radial ray representation by using the representative point as a common ray starting point ($p_2$ for Fig 11 case). After the conversion, Boolean union shape of all translated models is computed. For each ray, segments of all models on the same ray are collected. Segments corresponding to Boolean union of the collected segments are computed and assigned as the new segments on the ray of the Boolean union model. This segment-wise Boolean operation is iterated for all the rays, and the result 3R model of the union shape is derived.

In the complex milling operation, the number of cutting positions is often more than 10,000, therefore huge number of segment-wise Boolean union computation must be executed to
obtain the result shape. To accelerate the computation, we develop an algorithm using the polygon rendering function of GPU. Result shape obtained by our algorithm is not exact as the Boolean union shape, but the shape is acceptable for our optimal cutter posture determination purpose.

Figure 12 illustrates our pseudo Boolean union computation algorithm. For each shift vector, polyhedral part model is translated and positioned. After positioning all translated models, their hidden-surface-removed image is rendered using the perspective projection and the depth buffer mechanism of GPU. In the rendering operation, the viewing point is fixed at the representative point ($p_2$ in the figure). Viewing direction is temporary fixed in the positive z-axis direction as shown in the figure. In the perspective projection, the resolution of the displaying window, the field of view (FOV) parameter and aspect ratio must be given consistently. In our algorithm, the resolution of the window is set to be the same resolution to the uniform grid of each cubic face of the radial ray representation model, which is 768x768 in our implementation. FOV and aspect ratio are 90 degree and 1.0, respectively.

Figure 13: Coordinate determination method of a visible point using a depth value.

After the rendering operation of all translated objects, the coordinates of the points corresponding to pixels of the visible surface are sampled (see Fig. 12(b)). Such coordinates are computed using the pixel location in the frame buffer and its corresponding depth information. In Fig. 13, a determination process of the coordinates of a visible point $q_0$ at pixel $(i, j)$ with depth value $dep_0$ in the frame buffer is illustrated. In the hidden surface removal using the depth buffer, two clipping planes named near and far perpendicular to the viewing direction are defined [20]. These two planes limit the visible range in the viewing direction. Depth value 0.0 and 1.0 are assigned to near and far planes, respectively. Consider a ray from the viewing point going through the pixel at $(i, j)$. Coordinates of a visible point $q_0$ on the ray can be determined by using the coordinates of the viewing point, grid position $i$ and $j$, positions of near and far planes, and the depth information $dep_0$ representing the relative position of $q_0$ with respect to near and far planes.

The depth buffer mechanism is usually used for sampling points on the object surface nearest to the viewing point. In OpenGL, GL_LESS function is used in the depth value comparison for obtaining the smallest depth value for each pixel, which corresponds to the visible point on the object surface for each pixel. The depth value comparison function can be switched to GL_GREATER to obtain the depth information of the opposite farthest side surface for the pixel from the viewing point. The translations are repeated for all shift vectors again, and their hidden surface removed image is generated. In this time, the depth value comparison function is changed to GL_GREATER and a point $q_1$ is obtained for the pixel at $(i, j)$ as the farthest point from the viewing point (see Fig. 13). Obtained points $q_0$ and $q_1$ are then connected by a line segment representing the internal portion of the Boolean union shape of the translated objects for a ray connecting the viewpoint and pixel $(i, j)$. In Fig. 12(c), red segments represent the connection result.

Figure 14: Perspective projection for additional five viewing directions.

Figure 15: Difference between exact 3R model (a) and a pseudo model obtained by using the depth buffer mechanism (b).

Since the viewing direction is set to +Z direction in the method explained above, limited part (upper part) of the 3R model is only generated. To obtain the complete 3R model, rendering operations and segment generation are repeated to additional five viewing directions which are +X, -X, +Y, -Y and -Z directions as shown in Fig. 14 and the obtained results are combined.

3R model obtained by using the rendering function mentioned above is not an exact one. For each ray, the nearest point and farthest point on the object surface from the viewing point are only sampled in this method, therefore generated 3R model cannot represent the object shape with some concaves such as holes as shown in Fig. 15. This “pseudo-union” model is, however acceptable for our cutter posture determination purpose in the following reasons.
In this model, the nearest point to the viewing point is accurately recorded on each ray, therefore the AC computation can be properly executed.

A single segment is defined on each ray in this model. Since this segment connects the nearest point and farthest point from the viewing point on the ray, the obtained model contains the exact model within. Therefore, the collision-free cutter length determined using this model is always in safety side.

**NUMERICAL EXPERIMENTS**

A system for computing the optimal cutter posture was implemented using Visual C++ and CUDA 8.0. This system computes appropriate cutter postures with their corresponding ACs first, then it derives necessary cutter length for all appropriate cutter postures to select the optimal cutter posture with the shortest cutter length. Series of computational experiments were performed using a PC with Intel Core i7 Processor (3.6 GHz), 32 GB memory, and an NVIDIA GeForce GTX-980 GPU.

We applied the system to four cases of polyhedral models of mold parts and points representing the cutter positions for semi-finishing them. First three cases (case A, B and C) are simple models for tests. Case D is actual mold cavity and cutter position data for machining it. Tab.1 shows number of polygons of the models, cutter radius and number of cutter positions. Required time for computing the appropriate cutter postures is given in the fifth column of the table. This time includes the time for computing the pseudo union shape of translated mold models in the radial ray representation with the depth buffer mechanism. The sixth column in the table shows the time for computing the necessary cutter length for all appropriate postures. In the experiments, a common holder shape is used whose \( r_0, r_1 \) and \( t \) parameters are 15mm, 25mm and 30mm, respectively

As shown in the table, more computation time is necessary for complex cases with many polygons and many cutter positions. Our system can determine the appropriate cutter postures and necessary cutter length for complex cases in a few minutes.

<table>
<thead>
<tr>
<th>Case</th>
<th>Number of polygons</th>
<th>Cutter radius (mm)</th>
<th>Number of cutter locations</th>
<th>Time for selecting appropriate postures with AC (sec)</th>
<th>Time for computing necessary cutter length (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>2,518</td>
<td>1.0</td>
<td>3,758</td>
<td>6.53</td>
<td>9.64</td>
</tr>
<tr>
<td>B</td>
<td>2,968</td>
<td>1.0</td>
<td>3,848</td>
<td>7.40</td>
<td>0.44</td>
</tr>
<tr>
<td>C</td>
<td>45,174</td>
<td>3.0</td>
<td>11,757</td>
<td>8.86</td>
<td>24.07</td>
</tr>
<tr>
<td>D</td>
<td>844,180</td>
<td>5.0</td>
<td>5,479</td>
<td>139.05</td>
<td>182.10</td>
</tr>
</tbody>
</table>

Table 1: Required time for computing appropriate cutter postures and necessary cutter length for all appropriate postures.

Computation results for case C and D are only illustrated in Fig. 16 and Fig. 17. In these figures, (a) shows the sample parts. Red points in the figures represent the cutter position data. (b) in Fig. 16 and Fig. 17 show the computation result of the appropriate cutter postures for 3+2 milling the parts. In these figures, colored spherical surface is a Gauss map representing the appropriate cutter postures. Color on the surface corresponds to the peak angle of the AC for each cutter posture (\( \phi, \theta \)). Red corresponds to cutter postures with 0.1 degree peak angle and blue corresponds to postures with the maximum peak angle allowed in the sample case. (c) in Fig. 16 and 17 show the necessary cutter length for each appropriate cutter posture in the Gauss map representation. In this figure, blue corresponds to the postures with the minimum cutter length, which is the optimal
cutter posture for milling the part. (d) shows the cutter in the optimal posture.

CONCLUSIONS

In this paper, we propose a new solid modeling framework named radial ray representation (3R) and its application for computing the optimal cutter posture for milling a mold part in the 3+2 axis milling. 3R model is suitable for computing the accessibility cones (AC) for selecting the appropriate cutter postures and for computing the necessary cutter length for each appropriate posture. In the optimal cutter posture determination for multiple milling positions, the union shape of the translated mold parts is necessary. We proposed a fast computation method of a pseudo union shape using the depth buffer mechanism of GPU. The shape obtained by our method is not exact but acceptable in the AC computation and the necessary cutter length determination purpose. An experimental system is implemented and some computation results are demonstrated. Our system can determine the optimal cutter posture for milling the mold part using 3+2 axis milling in a few minutes.

ACKNOWLEDGMENTS

This research work was supported by JSPS KAKENHI Grant Number 17K06075.

REFERENCES


