www.cadanda.com
Computer-Aided Design AND APPLICATIONS

Cutter Engagement Feature Extraction Using Triple-Dexel Representation Workpiece Model and GPU Parallel Processing Function

Masatomo Inui[1], Masayoshi Kobayashi[2] and Nobuyuki Umezu[3]

[1]Ibaraki University, masatomo.inui.az@vc.ibaraki.ac.jp
[2]Ibaraki University, 17nm918g@vc.ibaraki.ac.jp
[3]Ibaraki University, nobuyuki.umezu.cs@vc.ibaraki.ac.jp

Corresponding author: Masatomo Inui, masatomo.inui.az@vc.ibaraki.ac.jp

ABSTRACT

For an accurate analysis of the cutting force, the cutter engagement feature (CEF) representing the contact area between the cutter and workpiece must be extracted for each small feed motion of the cutter. We previously proposed a method for accelerating the CEF computation using the parallel processing function of a graphics processing unit. To improve the stability in the CEF computation, we propose a novel CEF extraction algorithm using an intersection analysis between the cutter and the workpiece shape. A triple-dexel model is adopted to represent the workpiece shape to realize a highly accurate computation. The algorithm is extended to compute the CEF, not only for the cylindrical surface portion of a cutter but also for the flat area of a flat-end cutter and the spherical surface area of a ball-end cutter. Our cutting simulation system based on this algorithm can compute a single CEF in 0.12 ms to 0.25 ms.

Keywords: Cutting force analysis, cutter-workpiece engagement, intersection computation, parallel processing
DOI: https://doi.org/10.14733/cadaps.2019.89-102

1 INTRODUCTION

Cutting force simulation is helpful for optimizing the cutting process because it enables proper control of the cutter feed rate to realize a milling operation with a constant cutting force and achieve maximum efficiency. A majority of the commercial cutting force simulation systems conduct analyses based on the cutter removal volume of the workpiece after every small feed motion of the cutter (for example [3]). Although the computation cost of a removal volume-based analysis is small, the predicted result of the cutting force is frequently different from the actual value. Recent simulation systems yield a more precise cutting force estimation based on the analysis of the cutter engagement feature (CEF), representing the contact area between the cutter and workpiece during the milling process. This area is known as the cutter-workpiece engagement (CWE) in some literatures. The contribution of each small part of a flute during cutting is examined based on the CEF, and the corresponding cutting force (*dFri*, *dFti*, *dFai*) is computed (see Fig. 1). The total value and direction of the cutting force are determined by integrating the cutting forces along the flute [12].

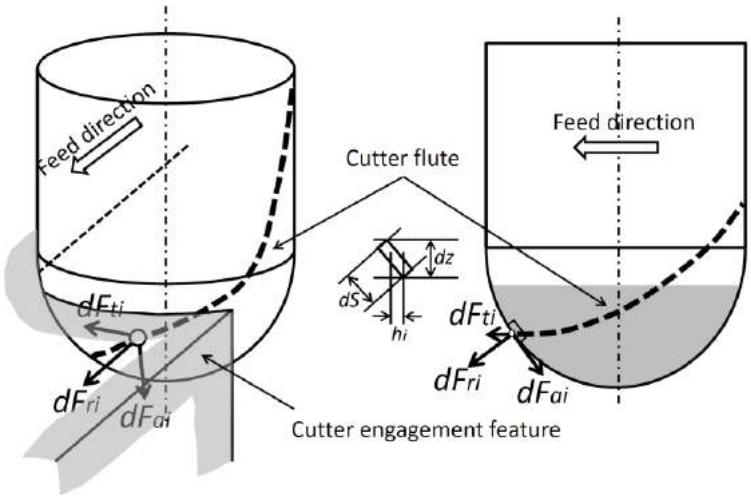

Figure 1: Basic concept of cutting force analysis using CEF.

In this method, the CEF must be computed for each constant feed motion of the cutter. For a precise analysis, the CEF should be computed at sufficiently short intervals. Consequently, a large number of CEF computations are necessary. Fast and accurate computation of the CEF is critical to realizing a cutting force simulation with a high practical applicability. We previously proposed a method for accelerating the CEF computation using the parallel processing capability of a graphics processing unit (GPU) [16]. Improvements to our prior method are proposed in this paper. This method is applicable to a cutting operation with a cutter in a fixed spindle axis direction, such as 3-axis milling or 3+2-axis milling. CEF is typically computed based on the contact analysis between the cutter and the in-process workpiece model. To improve the stability in the CEF computation, we propose a new CEF extraction algorithm using an intersection analysis between the cutter and the workpiece shape before the machining. In our prior method, a conventional dexel model was used to represent the workpiece shape. To realize a highly accurate computation, a triple-dexel model is adopted herein to represent the workpiece shape. The algorithm is extended to compute the CEF, not only for the cylindrical surface portion of a cutter (i.e., our prior method) but also the flat area of a flat-end cutter and the spherical surface area of a ball-end cutter.

In the next section, related studies on milling simulations are briefly explained. In Section 3, the basic concept of our parallel CEF computation method is illustrated. Explanations regarding CEF and the three-dimensional (3D) shape in the triple-dexel representation are provided in the same section. Our novel contributions are also offered. The details of the proposed CEF extraction algorithm are explained in Section 4. The experimental computation results are provided in Section 5, and we summarize our conclusions in Section 6.

2 RELATED STUDIES

Research works on milling simulation can be classified into two groups, namely, geometric milling simulation and simulation of the cutting force in the milling process. The main purpose of geometric milling simulation is to visualize the workpiece shape change and to detect possible gouges and collisions between a holder and the workpiece. A milling operation is geometrically equivalent to a Boolean subtraction of the swept volume of a cutter moving along a cutter path from a geometric model representing the stock shape. The majority of systems realize milling simulation based on this idea. One of the major differences among the systems is the representation scheme of the workpiece shape. CSG, Z-maps, dexels, voxels, octree, and vector representations have been used in prior studies [4, 6, 8, 11, 13, 17, 20, 21, 22, 25]. Using the parallel processing capability of a GPU for accelerating the simulation has also been studied, for example [14].

Cutting force simulation uses the result of the geometric milling simulation as the basic data for the analysis. In an early work, the cutting force was estimated using the cutter removal volume, for example [23]. Cutting force simulation using the CEF has become popular as a method for performing an accurate analysis. Analytical CEF computation methods are applicable to only 2.5-dimensional machining such as pocket milling (for example [10]). In recent studies, the CEF is obtained from the geometric milling simulation result. Gong and Feng developed a method using the cutter swept volume and cutter removal volume in the triangle mesh representation [9]. Erdim

and Sullivan realized a CEF computation algorithm using their geometric milling simulation software. In their system, the CEF is computed based on the contact analysis between the cutter surface and in-process workpiece model in the composite adaptively sampled distance field [7].

Kaneko and Horio proposed a parallel CEF estimation algorithm using a GPU [18]. Unlike the conventional CEF computation method, their method obtains CEFs without using geometric milling simulation. A sufficiently large number of points are arranged on the surface of a cutter. A CEF on the cutter in a certain position is computed by checking the intersection between each point on the cutter surface and the cutter removal volume corresponding to each small motion of the cutter. To efficiently detect the point-volume intersection for many surface points and many cutter removal volumes, the simultaneous thread execution capability of the GPU is used. Their prototype system generally needs 100 to 200 ms for computing a single CEF.

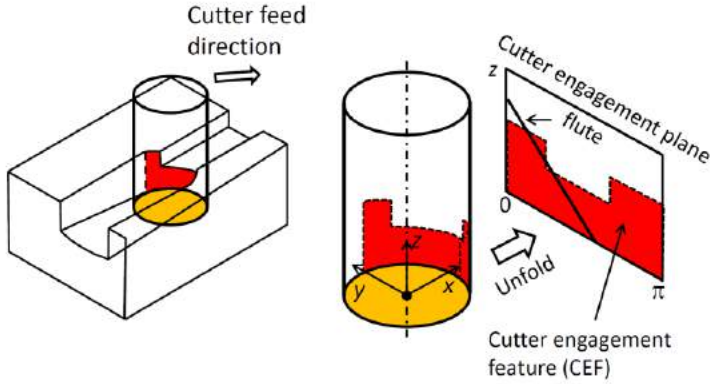

Figure 2: Cutter engagement feature in side surface and bottom surface of a flat-end cutter.

3 ALGORITHM OUTLINE

In the following discussion, we consider the CEF computation for a flat-end cutter or a ball-end cutter in a conventional 3-axis or 3+2-axis milling process. The extension of the method for addressing other cutters in different shapes is straightforward. In these milling processes, the spindle axis of the cutter is fixed in a certain posture. Milling operations with a cutter in a fixed axis direction are suitable for a rough milling process where the efficient removal of excess material is of the highest priority. This cutting method is also frequently used in the finishing process of precision molds for plastic parts.

3.1 Cutter Engagement Feature

Fig. 2 illustrates a milling operation with a flat-end cutter following a downward slope. A local coordinate frame is fixed on the cutter such that the frame's origin is at the center point of the end part of the cutter, and its z-axis is on its spindle axis. The y-axis is defined as the cross product of the z-axis and the feed direction of the cutter. The x-axis is then defined as the cross product of the y-axis and the z-axis. The red region represents a CEF between the side surface of the cutter and the workpiece; the yellow region represents a CEF for its bottom surface.

The contact area on the cylindrical surface of the cutter is mapped onto the cutter engagement plane, for which the axial depth of the cut is the vertical axis and the angular position measured from the y-axis in the clockwise direction is the horizontal axis. The mapped area is usually represented as a series of vertical rectangles of different heights. The CEF for the bottom surface of a flat-end cutter is analyzed for concentric circles in the bottom surface. The origin point of the local coordinate frame corresponds to the common center point of the circles. For each circle, the contacting area to the workpiece shape is analyzed as indicated in Fig. 3(a). In the ball-end cutter case, the CEF for the spherical bottom surface is analyzed using coaxial circles in the surface whose common axis corresponds to the spindle axis of the cutter (see Fig. 3(b)). For each circle, the contacting area to the workpiece is analyzed.

3.2 Triple-Dexel Representation

The triple-dexel model is used to represent the workpiece shape in our geometric milling simulation. Dexel modeling is known as a discrete method for representing a 3D shape [24]. In this method, the object shape is represented by a bundle of z-axis-aligned segments defined for each grid point of a square mesh in the xy plane. With dexel modeling, near-vertical surfaces have inevitable large shape errors caused by a finite grid resolution. The triple-dexel model was proposed to overcome this nonuniformity of the shape errors and to realize accurate shape representation [1]. In this representation, the 3D shape is not only defined by a z-axis-aligned dexel model but also by an x-axis-aligned dexel model based on a square mesh in the yz plane and a y-axis-aligned dexel model based on a mesh in the zx plane (see Fig. 4(a)).

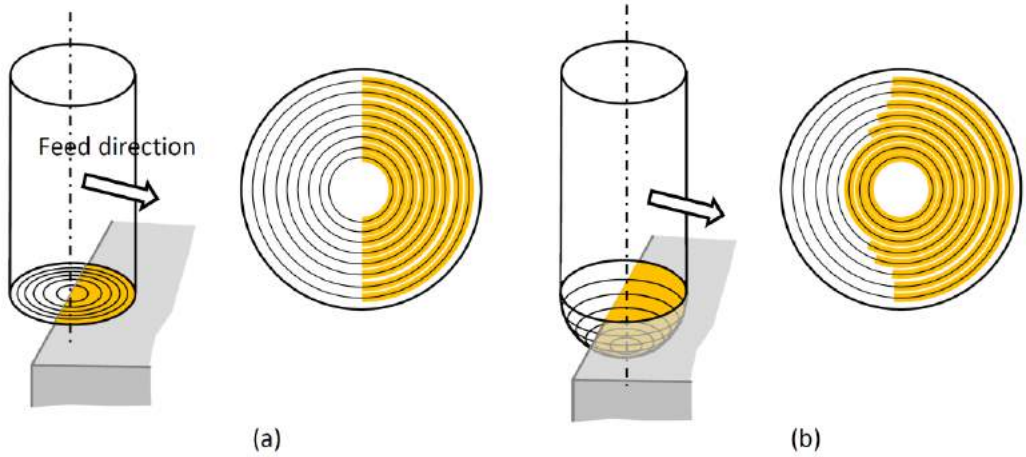

Figure 3: Definition of CEF in bottom surface of cutter: (a) flat-end cutter case; (b) ball-end cutter case.

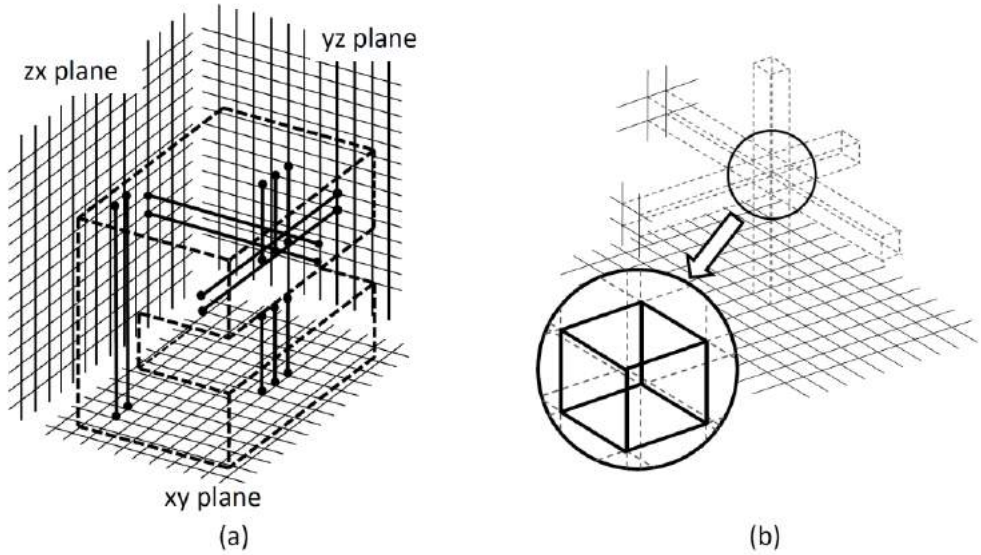

Figure 4: Triple-dexel representation of solid model (a) and cubic cell defined by properly aligned dexels (b).

In the triple-dexel representation, the simulation space can be divided into a set of cubes defined by properly positioned x-, y-, and z-axis-aligned dexels, as displayed in Fig. 4(b). We define a regular spatial grid with a cubic cell structure in an axis-aligned rectangular box that holds the workpiece shape within. The grid is projected to the xy, yz, and zx planes to define an axis-aligned square mesh in each (see Fig. 5). Lines starting from the grid

points in each plane, perpendicular to the plane, correspond to the grid lines organizing the spatial grid structure in the box. x-, y-, and z-axis-aligned dexels are defined using these meshes in the xy, yz, and zx planes, respectively.

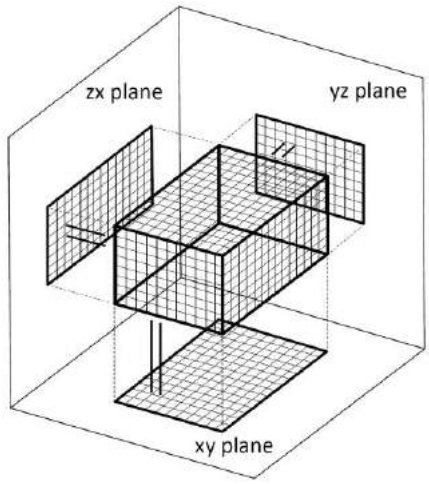

Figure 5: Regular spatial grid for defining square meshes in xy, yz, and zx planes.

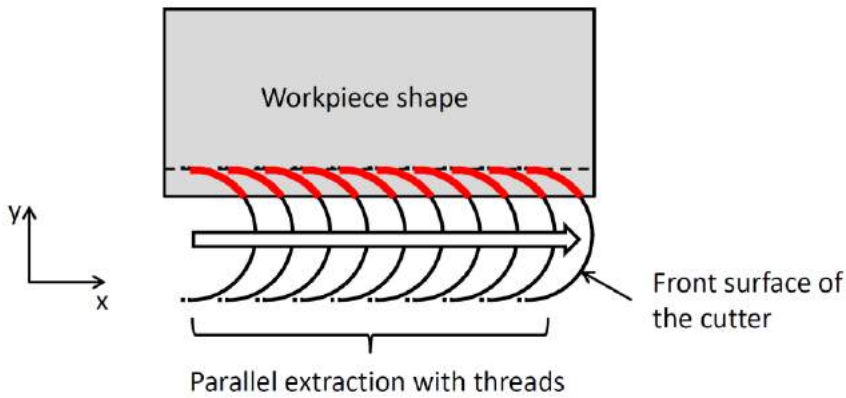

Figure 6: Parallel computation of multiple CEFs in the linear motion of cutter.

3.3 Parallel CEF Computation Algorithm

Unlike a typical processing unit for computers, a GPU is designed as a many-core architecture in which maximal small and simple processing units are integrated on a single chip. The GTX-1080 processor, a new GPU developed by the nVIDIA Corporation, has 2,560 processing units on a single chip. They are called Compute Unified Device Architecture (CUDA) cores. The main factor for the acceleration gain of a GPU is the substitution of the iterative execution of an operation in a loop with a simultaneous execution of its equivalent threads on the processing cores.

The replacement of an operation by threads is typically not applicable to a milling simulation, in which the results of a prior cutting operation affect the ensuing operations. This dependency on the cutting sequence can be ignored for a cutting operation along a linear trajectory because the cutting result is always located behind the cutter. It cannot affect the CEF in the subsequent cutting operations. In this case, the CEF computation at each cutting position can be achieved by simply placing a shape model of the cutter at that position and examining the intersection area between the workpiece shape and the front side surface and bottom surface of the cutter model. This computation can be accomplished for multiple places simultaneously with threads. Fig. 6 illustrates a parallel extraction of ten CEFs in the front side surface of a cutter in a straight motion.

Based on this consideration, we propose a novel parallel algorithm for the CEF computation. This algorithm is an improvement of our prior algorithm discussed in [16]. Novel contributions of the new algorithm are explained in the next subsection. The input data of the algorithm consists of a polyhedral STL model that approximates the workpiece shape before the milling, shape data of the cutter, and cutter path data representing the trajectory of the center point of the cutter in the milling process. Most commercial CAD systems provide a function for exporting the native model data in the STL format; therefore, acceptance of only the STL model is not a limitation of our method. The workpiece shape is converted to an equivalent 3D model in the triple-dexel representation. In the conversion, our improved ray algorithm is used [15]. In computer-controlled machining, the cutter path data is represented as a series of linear segments. The proposed CEF computation algorithm iterates the following two-step operation for each linear segment in the cutter path (Fig. 7(a)).

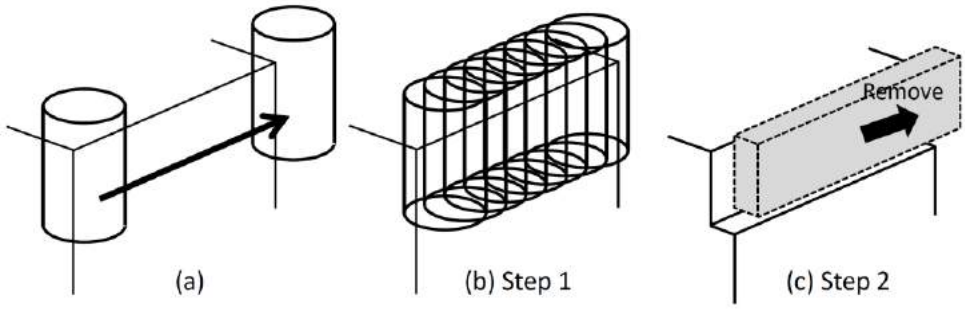

Figure 7: Basic processing flow of proposed parallel CEF computation algorithm.

Step 1: Cutter models are placed in the workpiece model at a small constant interval along the cutter path segment. The CEF in the side surface and bottom surface of the cutter at each cutter placement are computed based on the intersection analysis between the surfaces and the workpiece model (see Fig. 7(b)). The CEF computation at each cutter placement is mutually independent; therefore, the parallel processing capability of the GPU is applied to accelerate the computation. After the computation, obtained data are transferred from the graphics memory of the GPU to the main memory of the CPU to complete the CEF computation.

Step 2: The swept volume of the cutter moving along the segment is computed. The workpiece shape is then modified by subtracting the cutter swept volume from the workpiece model to obtain the result shape of the cutting (see Fig. 7(c)). In the proposed algorithm, the cutter swept volume is represented in the triple-dexel representation. The model subtraction operation in the triple-dexel representation can be decomposed to a set of dexel-wise subtractions. As all dexel-wise subtractions are mutually independent, the parallel processing capability of the GPU can be used for accelerating the operation.

Parallelism of the computation in the proposed algorithm is dependent on the length of each linear segment in the cutter path and the interval distance between the CEF computations on the linear segment. To maximize the parallelism, a cutter path with long linear segments and a short interval distance between the CEF computations is preferable. A cutting force analysis is typically used for optimizing the feed control in a rough cutting operation in which the cutter path is known to contain rather long linear segments. The computation cost of the CEF becomes critical when the interval distance is set to an extremely small value for achieving an accurate cutting force analysis. Because of these reasons, the proposed method is considered an effective solution for accelerating the CEF computation.

3.4 Our Contributions

The proposed algorithm has the following three novel features compared to prior works:

Parallel CEF computation: As with our previous research [16], the method proposed in this paper realizes high-speed CEF computation by fully utilizing the many-core architecture of a GPU. The proposed algorithm is applicable to only 3-axis and 3+2-axis milling in which the posture of the cutter is not changed during the operation. In precision die machining, in which cutting force simulation is highly beneficial, the majority of cutting work continues to use 3-axis machine tools and hence, this algorithm has high effectiveness. The proposed parallel CEF computation algorithm is different from the algorithm proposed by Kaneko and Horio [18]. Computation experiments confirm that a CEF computation system based on the proposed algorithm can derive a single CEF in 0.2 ms using the latest GPU and is more than 100 times faster than the prior method.

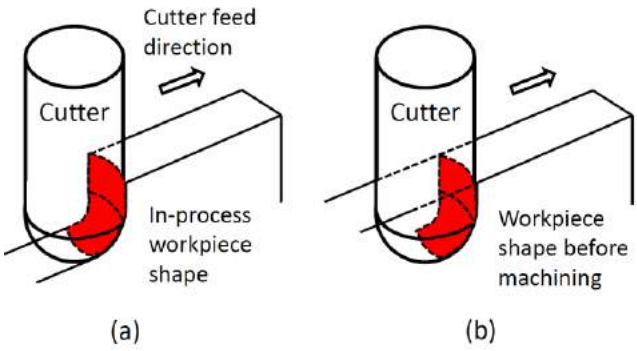

Figure 8: Contact between cutter and workpiece.

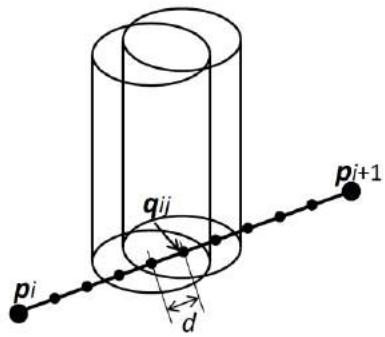

Figure 9: Subdivision of linear cutter path segment.

Robust CEF computation: The majority of computation algorithms to date (including our prior algorithm) obtain a CEF based on a contact analysis between the cutter and the in-process workpiece or a contact analysis between the cutter and the cutter removal volume. Because the surface geometry of the cutter and the surface of the in-process-workpiece/cutter-removal-volume are virtually identical (red surface region in Fig. 8(a)), the result of the contact analysis is susceptible to inevitable shape errors in the in-process workpiece model or cutter removal volume. To reliably obtain the contact area, a method has been proposed in which the shape of the cutter is marginally expanded to sufficiently penetrate into the workpiece; however, a reduction in the calculation accuracy is unavoidable [9]. The CEF computation for a straight cutter motion is independent of the cutting sequence; therefore, the proposed system can compute the CEF based on a robust intersection analysis between the workpiece shape before the milling operation and the cutter shape sufficiently penetrating to the workpiece shape as indicated in Fig. 8(b).

Accurate CEF computation: A conventional dexel model was used for representing the workpiece shape in our prior method [16]. Because shape errors caused by the limited grid resolution are unavoidable in the dexel representation, our prior method is not applicable for analyzing the CEF in the finish-milling operation with a cutting depth less than the grid size. To overcome this accuracy limitation, the triple-dexel model is introduced as the workpiece shape representation scheme. Boess *et al.* also proposed the use of the triple-dexel model for accurate cutter contact zone analysis; however, their method focused on a simple milling process in a micro-scale [2]. The CEF on the side surface of the cutter is only computed in our prior method. We extended the algorithm for computing the CEF on the bottom surface of the cutter.

4 PARALLEL CUTTER ENGAGEMENT FEATURE EXTRACTION

4.1 Step 1: Cutter Contact Analysis with a Triple-Dexel Model

Consider a parallel computation of multiple CEFs for a cutter motion along a linear segment connecting points $\boldsymbol{p}_i$ and $\boldsymbol{p}_{i+1}$ in the cutter path. A set of points is generated in the segment at sufficiently small and constant interval d. The cutter shape models are placed on the segment such that the center point of each cutter model and a point on the segment become coincident as indicated in Fig. 9. In this section, the extraction of the CEF on the surface of a cutter placed on a point $\boldsymbol{q}_{ij}$ is explained.

4.1.1 CEF on the side surface of the cutter

The cylindrical surface of the cutter can be subdivided into two parts, namely, the “front” part facing the feed direction of the cutter and the “back” part. The CEF is only generated on the front part of the side surface. For preparation, a set of segments parallel to the cutter axis direction are placed on the cylindrical surface such that they surround the front part of the surface as displayed in Fig. 10(a). In our current implementation, 180 segments are placed on the surface, as the circular distance between two adjacent segments becomes $r \cdot \pi/180$, where r is the radius of the cutter. Each segment is further subdivided into a series of smaller segments by inserting points in the segment. The distance between neighboring points is set equal to the gird size of the square mesh of the triple-dexel model. For each small segment, the axis-aligned bounding box (AABB) holding the segment is defined as indicated in Fig. 10(b).

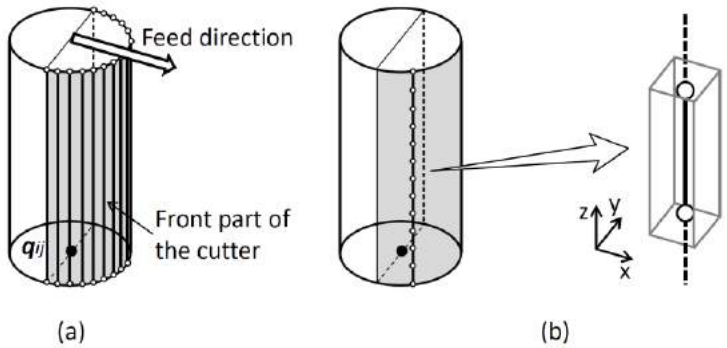

Figure 10: Definition of short segment in side surface of cutter.

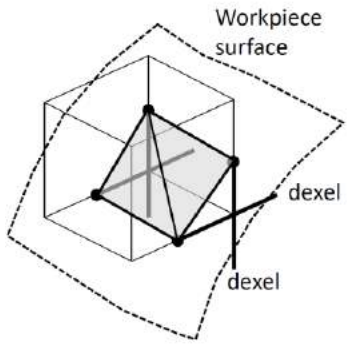

Figure 11: Placement of small polygons in cell.

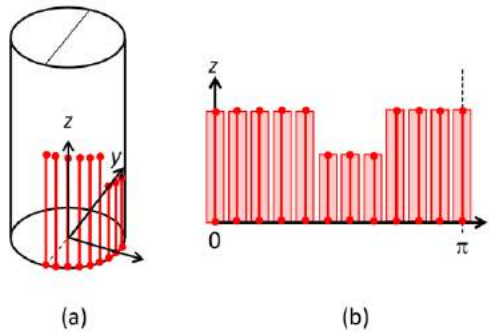

Figure 12: CEF as series of thin rectangles.

As explained in Section 3, a triple-dexel model is defined based on a regular spatial grid with a cubic cell structure. Square meshes in the xy, yz, and zx planes correspond to the projections of the grid to the planes. x-, y-, and z-axis-aligned dexels of the workpiece model are defined on the grid lines. Fig. 11 illustrates one cubic cell in the spatial grid. Because this cell is located in the middle of the workpiece surface, four segments of the cell have intersection points with the surface. Small black points in the figure correspond to the intersection points. The intersection points between the segments and the workpiece surface correspond to the end points of the x-, y-, or z-axis-aligned dexels. The eight vertices of the cell can be classified as internal or external with respect to the workpiece shape using the dexel information. If a vertex is contained within one of the dexels, then this vertex is judged as an internal vertex; otherwise, the vertex is recognized to be external. After the classification of the eight vertices, the result is converted to an 8-bit pattern. Using this pattern as a key, a set of surface polygons (two triangle polygons in Fig. 11) are retrieved from the database of the marching cube algorithm [19].

Using the geometric information of the AABB holding a small segment, spatial cells intersecting the AABB are verified. The workpiece surface intersecting the cells is computed as a set of small polygons using the above algorithm. The intersection points between the small segment and polygons are then derived. This operation is repeated for all small segments in each long segment. The obtained points and the two end points of the long segment are sorted according to the segment direction, to determine segment portions that are contained within the workpiece. Such intersecting portions are arranged side-by-side as indicated in Fig. 12; then, they are replaced by thin rectangles of the same vertical length. The CEF on the side surface of the cutter is finally obtained as a series of thin rectangles.

In our implementation of the parallel CEF computation software, one GPU thread is assigned to each small segment in the front side surface of the cutter. Each thread determines the intersecting points between the segment and the workpiece surface. In the implementation of a program using the thread, CUDA is used [5]. The obtained points are transferred to the CPU to determine the intersecting portion between the segment in the cutter surface and the workpiece, and to generate a CEF in the side surface of the cutter.

4.1.2 CEF on the bottom surface of the cutter

With flat-end milling, the CEF computation for the bottom surface is necessary only when the cutter moves along a downward slope. Equally spaced concentric circles are generated on the bottom surface of a flat-end cutter. In the current implementation, a fixed number of circles (ten circles) are generated. It would not be difficult to improve the algorithm to permit the specification of the interval distance between the concentric circles. For each circle, a set of points are generated, as indicated in Fig. 13(a). 360 points are generated in each circle, because the circular distance between two adjacent points becomes $r \cdot /180$, where r is the radius of the circle. By inserting the points, each circle is subdivided to a set of connecting circular segments. For each circular segment, an AABB holding the segment within is defined as displayed in Fig. 13(b).

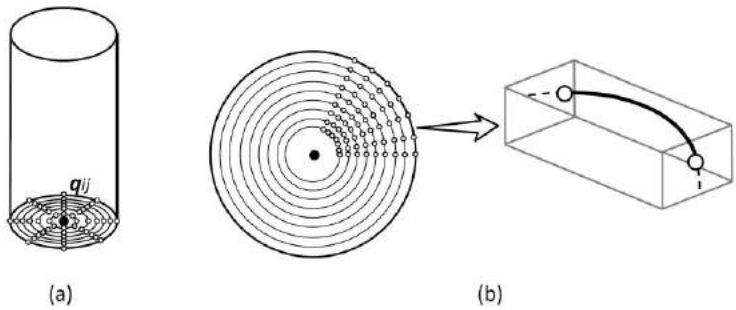

Figure 13: Definition of short segment in bottom surface of cutter.

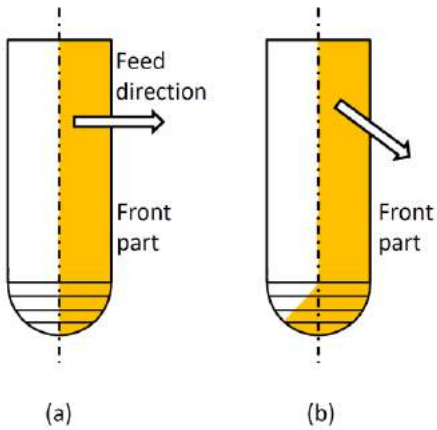

Figure 14: Front part of spherical bottom surface of ball-end cutter.

As in the CEF computation for the side surface of the cutter, spatial cells intersecting the AABB are selected. The workpiece surface intersecting the cells is computed using the marching cubes algorithm and the intersection points between the small circular segment and polygons are derived. This operation is executed for all small segments in a circle to obtain a set of intersection points between the circle and the workpiece. The obtained points in the circle are sorted according to their central angles to determine the portions in the segment that are contained within the workpiece; these intersecting portions are expanded to be a circular belt shape. The CEF on the bottom surface of the flat-end cutter is finally obtained as a set of concentric circular belts (see yellow belts in Fig. 3(a)). As in the CEF computation for the side surface, one GPU thread is assigned to each small circular segment. The thread determines the intersection points between the segment and the workpiece surface. The obtained points are transferred to the CPU to determine the intersecting belt shape.

With ball-end milling, virtually the same algorithm is used; however, coaxial circles in the bottom surface of the cutter are used in the CEF computation, instead of the concentric circles for the flat-end cutter case. Unlike flat-end milling, the CEF computation for the spherical surface is not limited to milling in the downward motion. In Fig. 14(a), the side view of a ball-end cutter in a horizontal motion is illustrated. In this case, the spherical bottom surface of the cutter can be subdivided into a front part and a back part by a vertical plane containing the tool axis

and perpendicular to the feed motion of the cutter. The CEF in the bottom surface is generated only in the front part of the surface. The front part in the bottom surface corresponds to a set of points whose normal vector has a feed direction component. If the cutter moves downwards, the front side of the cutter is changed to a yellow range as displayed in Fig. 14(b). After computing a circular belt shape, the part of the belt that is contained within the front part of the bottom surface of the cutter is cropped and used as the CEF.

In our current system, CEF extraction software using the circular segments in the bottom surface is not yet available. Our current software checks the position of each point in the circles with respect to the workpiece shape. After checking all the points in the bottom surface, three neighboring points are connected by a triangle if they are in the front region of the bottom surface and they are contained within the workpiece shape. A CEF on the bottom surface of the cutter is consequently obtained as a set of triangles. One GPU thread is assigned to each point on the bottom surface to check its position with respect to the workpiece shape.

4.2 Step 2: Geometric Milling Simulation

In the second step of the algorithm, a geometric milling simulation is executed to obtain a workpiece shape after a cutting operation with the cutter moving from p_i and p_{i+1}. This operation is achieved in the following two substeps.

Step 2.1: Define the swept volume of the cutter moving along a linear segment. The swept volume is converted to its equivalent form in a triple-dexel representation.

Step 2.2: Execute dexel-wise subtraction of the cutter swept volume from the workpiece model for x-, y-, and z-axis-aligned dexels to obtain the resultant shape of the workpiece after milling.

In Step 2.2, the dexel-wise Boolean subtraction is computed between a series of dexels in the workpiece model on a grid point and a dexel in the cutter swept volume on the same grid point. The Boolean subtraction of the dexels on a grid point is independent of the operations performed on the other grid points. Therefore, the dexel-wise subtraction can be conducted in a parallel manner for all grid points. A single GPU thread is assigned to each grid point for executing the dexel-wise Boolean subtraction on the grid point.

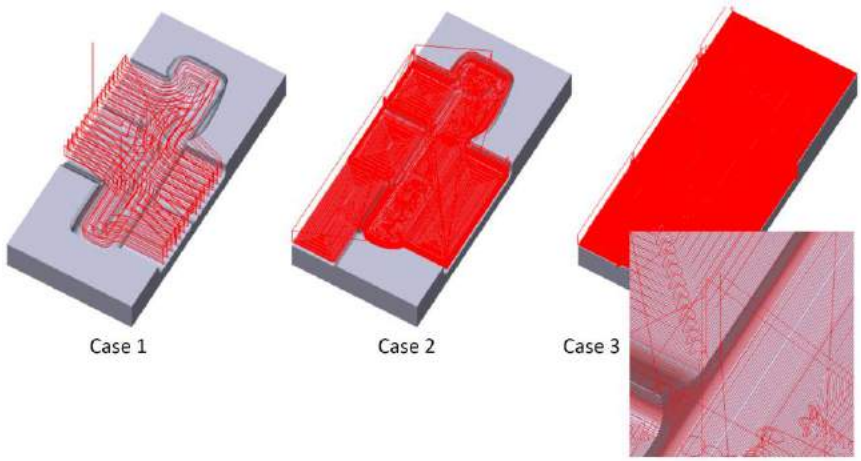

Figure 15: Three cutter path data for numerical experiments.

Milling case	Case 1	Case 2	Case 3
Total length	10,149.15 mm	34,507.83 mm	88,043.69 mm
Number of linear segments	9,522	35,389	156,797
Cutter types	Flat r = 5 mm Ball r = 5 mm	Flat r = 3 mm Ball r = 3 mm	Flat r = 1 mm Ball r = 1 mm

Table 1: Simulation conditions.

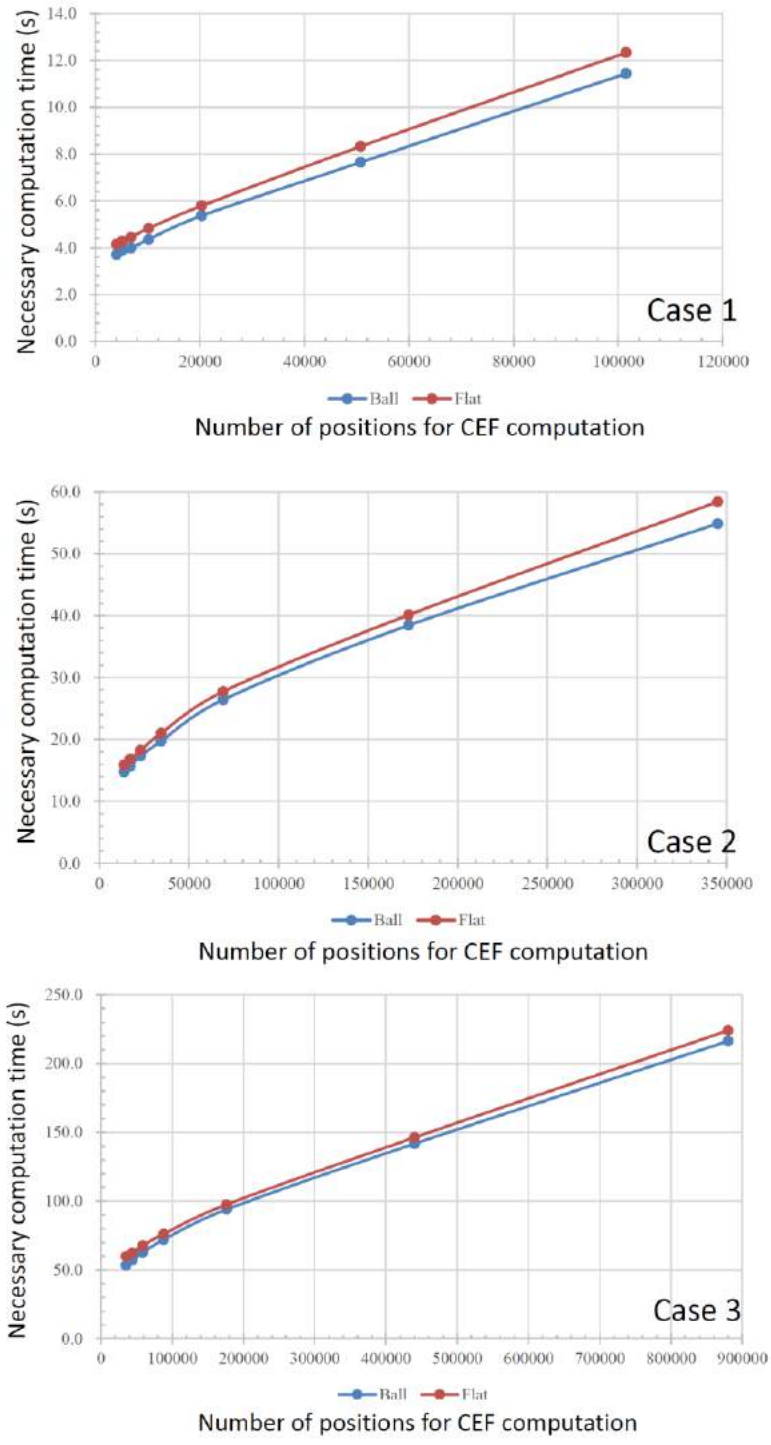

Figure 16: Required computation time.

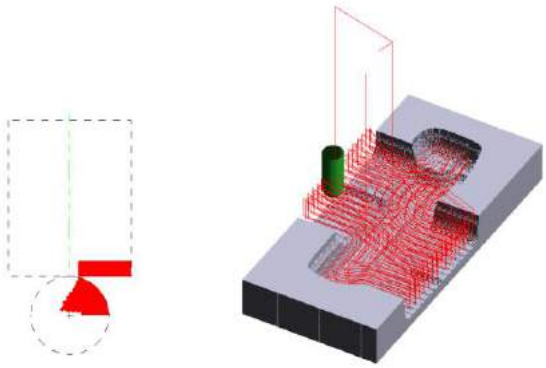

Figure 17: Example of computed CEF.

5 IMPLEMENTATION AND NUMERICAL EXPERIMENTS

A triple-dexel-based milling simulation system with a CEF computation capability was implemented using Visual Studio 2010 and CUDA 8.0. A 64-bit PC with an Intel Core i7 Processor (4.0 GHz), 32 GB memory, and an nVIDIA GeForce GTX-1080 GPU was used in the experiments. The size of the stock object was 150 × 70 × 20 mm. Triple-dexels for representing the workpiece shape were generated using three square meshes in the xy, yz, and zx planes, which were obtained by projecting a spatial grid of 1,925 × 901 × 261 resolution. We applied the system for simulating three milling cases. The first case was with a cutter path for rough machining. The second and third cases were for semi-finishing and finishing milling, respectively. Fig. 15 illustrates these three cutter path data with a workpiece model corresponding to the milling result. Table 1 presents the simulation conditions: total length of the cutter path, number of linear segments in the path, and cutter types used in the simulation.

Fig. 16 depicts the total time required for the geometric milling simulation and CEF computation. The interval distance, *d*, for computing the CEF, is set to 2.5, 2.0, 1.5, 1.0, 0.5, 0.2, and 0.1 mm. In the graph, the horizontal axis corresponds to the number of CEF computations. The required computation time increased near-linearly according to the number of the CEF computations. It can be observed that 12 s were required for computing CEFs at 100,000 cutter positions for case 1. 60 s were necessary for 350,000 cutter positions for case 2, and 230 s were necessary for 900,000 cutter positions for case 3. On average, 0.12 to 0.25 ms were consumed for computing a single CEF. Fig. 17 displays an example of the extracted CEFs for a ball-end cutter at one milling position with a cutter path for rough machining (case 1). The red shape in the circle represents a CEF on the spherical surface of the cutter.

6 CONCLUSION

In this paper, we proposed a novel method for accelerating the CEF computation using the parallel processing capability of a GPU. The CEF computation software proposed in our prior paper exhibited inevitable shape errors owing to the grid resolution in the horizontal plane. To overcome this accuracy limitation, we introduced a triple-dexel model for representing the workpiece shape. Prior works computed a CEF based on the contact analysis result between the cutter and the in-process workpiece model. This method is numerically unstable and the obtained CEF frequently includes significant errors. To improve the stability in the CEF computation, we developed a new CEF extraction method using an intersection analysis between the cutter and the workpiece shape. The algorithm was extended to compute the CEF not only for the cylindrical surface portion of a cutter but also the flat area of a flat-end cutter and the spherical surface area of a ball-end cutter. A milling simulation system with a CEF computation capability was implemented to demonstrate the performance of the algorithm. On an average, 0.12 ms to 0.25 ms were consumed for computing a single CEF. We are now completing our system for precisely computing the CEF in the bottom surface. Extension of the method for analyzing the CEF for 5-axis milling is also our future research subject.

ACKNOWLEDGEMENTS

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

Masatomo Inui, http://orcid.org/0000-0002-1496-7680
Masayoshi Kobayashi, http://orcid.org/0000-0002-1225-728X
Nobuyuki Umezu, http://orcid.org/0000-0002-7873-7833

REFERENCES

[1] Benouamer, M.O.; Michelucci, D.: Bridging the Gap between CSG and Brep via a Triple Ray Representation, Proceedings of ACM Symposium on Solid Modeling and Applications, 1997, 68-79. https://doi.org/10.1145/267734.267755
[2] Boess, V.; Ammermann, Chr.; Niederwestberg, D.; Denkena, B.: Contact Zone Analysis Based on Mutidexel Workpiece Model and Detailed Tool Geometry Representation, Procedia CIRP, 4, 2012, 41-45. https://doi.org/10.1016/j.procir.2012.10.008
[3] CGTech, http://www.cgtech.com/products/about-vericut/optipath/.
[4] Choi, B.K.; Jerard, R.B.: Sculptured Surface Machining, Theory and Applications, Kluwer Academic Publishers, 1998, 255-258.
[5] CUDA Compute Unified Device Architecture Programming Guide, http://docs.nvidia.com/cuda/pdf/CUDA_C_Programming_Guide.pdf, NVIDIA.
[6] Drysdale, R.L.; Jerard, R.B.; Schaudt, B.; Hauck, K.: Discrete simulation of NC machining, Algorithmica, 4(1), 1989, 33-60. https://doi.org/10.1007/BF01553878
[7] Erdim, H.; Sullivan, A.: Cutter Workpiece Engagement Calculations for Five-axis Milling Using Composite Adaptively Sampled Distance Fields, Procedia CIRP, 8, 2013, 438-443. https://doi.org/10.1016/j.procir.2013.06.130
[8] Fridshal, R.; Cheng, K.P.; Duncan, D.; Zucker, W.: Numerical control part program verification system, Proceeding of Conference on CAD/CAM Technology in Mechanical Engineering, 1982, 236-254.
[9] Gong, X.; Feng, H.-Y.: Cutter-workpiece engagement determination for general milling using triangle mesh modeling, Journal of Computational Design and Engineering, 3(2), 2016, 151-160. https://doi.org/10.1016/j.jcde.2015.12.001
[10] Guerrero-Villar, F.; Dorado-Vicente, R.; Romero-Carrillo, P.; López-García, P.; Mercado-Colmenero, J.: Computation of Instantaneous Cutter Engagement in 2.5D Pocket Machining, Procedia Engineering, 132, 2015, 464-471. https://doi.org/10.1016/j.proeng.2015.12.520
[11] Huang Y.; Oliver, J.H.: NC milling error assessment and tool path correction, SIGGRAPH 1994 Proc. 21st Annual Conference on Computer Graphics and Interactive Techniques, 1994, 287-294. https://doi.org/10.1145/192161.192231
[12] Merdol, D.; Altintas, Y.: Virtual cutting and optimization of three axis milling processes, International Journal of Machine Tools and Manufacture, 48(10), 2008, 1063-1071. https://doi.org/10.1016/j.ijmachtools.2008.03.004
[13] Inui, M.; Kakio, R.: Fast visualization of NC milling result using graphics acceleration hardware, in Proc. IEEE International Conference on Robotics and Automation, 2000, 3089-3094. https://doi.org/10.1109/ROBOT.2000.845138
[14] Inui, M.; Umezu, N.; Shinozuka, Y.: A Comparison of Two Methods for Geometric Milling Simulation Accelerated by GPU, Transactions of the Institute of Systems, Control and Information Engineers, 26(3), 2013, 95-102. https://doi.org/10.5687/iscie.26.95
[15] Inui, M.; Umezu, N.; Tsukahara, M.: Simple offset algorithm for generating workpiece solid model for milling simulation, Journal of Advanced Mechanical Design, Systems, and Manufacturing, 11(4), 2017. https://doi.org/10.1299/jamdsm.2017jamdsm0042
[16] Inui, M.; Kobayashi, M.; Umezu, N.: Fast Extraction of Cutter Engagement Features by Using the Parallel Processing Function of a GPU, Proc of 13th IEEE Conference on Automation Science and Engineering (CASE), 2017. 668-673. https://doi.org/10.1109/COASE.2017.8256180
[17] Jerard, R.B.; Hussaini, S.Z.; Drysdale, R.L.; Schaudt, B.: Approximate methods for simulation and verification of numerically controlled machining programs, The Visual Computer, 5(6), 1989, 329-348. https://doi.org/10.1007/BF01999101
[18] Kaneko, J.; Horio, K.: Fast cutter workpiece engagement estimation method for prediction of instantaneous cutting force in continuous multi-axis controlled machining, International Journal of Automation Technology, 7(4), 2013, 391-400. https://doi.org/10.20965/ijat.2013.p0391
[19] Lorensen, W.E.; Cline, H.E.: Marching Cubes: A High Resolution 3D Surface Construction Algorithm, Computer Graphics, Proceedings of the 14th annual conference on Computer graphics and interactive techniques, 1987, 163-169. https://doi.org/10.1145/37401.37422

[20] Menon, J.P.; Robinson, D.M.: Advanced NC verification via massively parallel raycasting: extensions to new phenomena and geometric domains, ASME Manufacturing Review, 6(2), 1993, 141-154.
[21] Oliver, J.H.; Goodman, E.D.: Direct dimensional NC verification, Computer-Aided Design, 22(1), 1990, 3-10. https://doi.org/10.1016/0010-4485(90)90023-6
[22] Saito, T.; Takahashi, T.: NC machining with G-Buffer method, in SIGGRAPH 1991 Proceedings of 18th Annual Conference on Computer Graphics and Interactive Techniques, 25(4), 1991, 207-216. https://doi.org/10.1145/122718.122741
[23] Takata, S.; Tsai, M.D.; Inui, M.; Sata, T.: A cutting simulation system for machinability evaluation using a workpiece model, CIRP Annals, 38(1), 1989, 417-420. https://doi.org/10.1016/S0007-8506(07)62736-X
[24] Van Hook, T.: Real-time shaded NC milling display, SIGGRAPH 1996 Proceedings of 13th Annual Conference on Computer Graphics and Interactive Techniques, 20(4), 1996, 15-20. https://doi.org/10.1145/15922.15887
[25] Wang, W.P.; Wang, K.K.: Geometric modeling for swept volume of moving solids, IEEE Computer Graphics and Applications, 6(12), 1987, 8-17. https://doi.org/10.1109/MCG.1986.276586