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

Authors:
Masatomo Inui, masatomo.inui.az@vc.ibaraki.ac.jp, Ibaraki University
Masayoshi Kobayashi, 17nm918g@vc.ibaraki.ac.jp, Ibaraki University
Nobuyuki Umezu, nobuyuki.umezu.cs@vc.ibaraki.ac.jp, Ibaraki University

Keywords:
Milling Simulation, Cutter Engagement Feature, Triple-Dexel Representation, Parallel Processing, GPU

DOI: 10.14733/cadconfP.2018.189-193

Introduction:

Cutting force simulation is helpful for optimizing the cutting process, because it enables proper control of the cutter feed rate to achieve a milling operation with a constant cutting force. Recent simulation systems yield a precise cutting force estimation based on analysis of the cutter engagement feature (CEF), representing a contact area between the cutter and the workpiece during the milling process. The contribution of each small part of a flute during cutting is examined, based on the CEF, and the corresponding cutting force (dF_{ri}, dF_{ti}, dF_{ai}) is computed (see Fig. 1). The total cutting force is determined by integrating the cutting forces along the flute [3].

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. A cutter path for milling a large object, such as a stamping die for an automobile, is often several tens of meters in length. Consequently, a large number of CEF computations are necessary. Thus, fast computation of the CEF is critical to realizing a cutting force simulation with a high practical applicability.

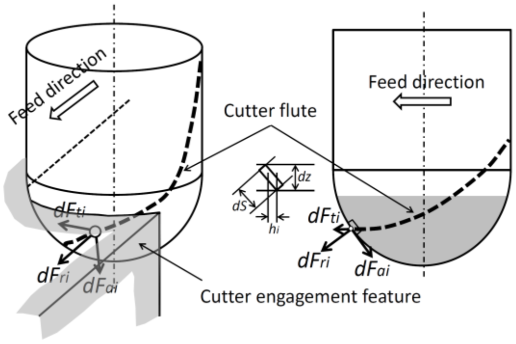

Fig. 1: Basic concept of the cutting force analysis using CEF.

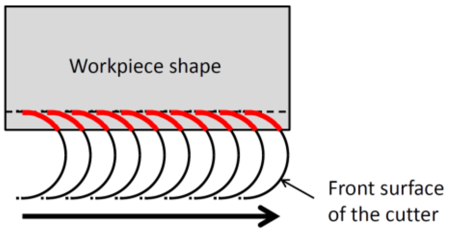

Fig. 2: Parallel CEF computations for a cutting operation along a linear trajectory.

We previously proposed a method for accelerating the CEF computation by using the parallel processing capabilities of a graphics processing unit (GPU) [2]. CEF is typically computed based on the contact analysis between the cutter and 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 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 spherical surface area of a ball-end cutter.

Main Idea:

Parallel Milling Simulation and CEF Computation

In the following discussion, we consider the CEF computation for a milling process with a flat-end or ball-end cutter in a fixed spindle axis direction. The input data of the algorithm consists of a polyhedral STL model that represents the workpiece shape, the shape data of the cutter, and the cutter path data representing the trajectory of the center point of the cutter during the milling process.

In contrast, for a typical computer processing unit, the GPU is designed with a many-core architecture. The main factor contributing to the acceleration gain of a GPU is the substitution of the iterative execution of an operation in a loop with the simultaneous execution of its equivalent threads on the processing cores. The replacement of an operation by threads is generally not applicable to a milling simulation, in which the results of a prior cutting operation affects proceeding 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 by examining the intersection area between the workpiece shape and front surface of the cutter model. This computation can be done for multiple places simultaneously, as shown in Fig. 2.

Based on this consideration, we developed a parallel algorithm for CEF computation [2]. It iterates the following two-step operation for each linear segment in the cutter path (see Fig. 3).

Step1: Cutter models are placed in the workpiece model at a small constant interval along the cutter path segment. The CEF in the surfaces of the cutter at each cutter placement are computed based on the intersection analysis between the surfaces and workpiece model.

Step2: Computes the swept volume of the cutter moving along the linear segment. The workpiece shape is then modified by subtracting the cutter swept volume from the workpiece model to obtain the result shape of the cutting.

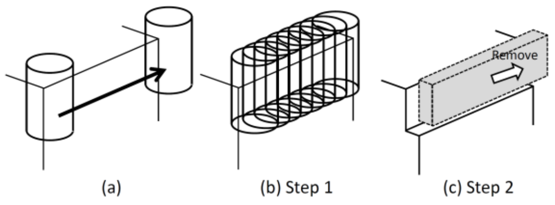

Fig. 3: Steps of our CEF computation algorithm.

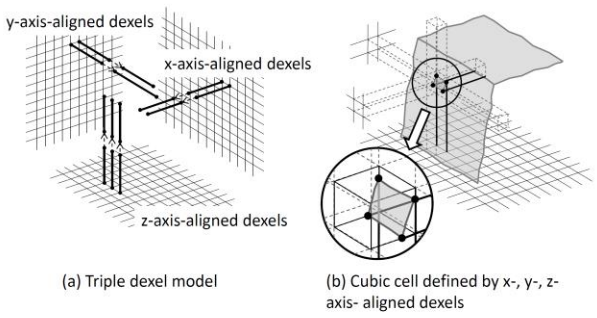

Fig. 4: 3D shape modeling using triple-dexels.

Triple-Dexel Model for Representing Workpiece Shape

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. 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 non-uniformity of the representation accuracy. 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)) [1].

In a 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 shown in Fig. 4(b). We define a regular spatial grid with a cubic cell structure in an axis-aligned rectangular box which 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. Lines starting from the grid points in each plane, perpendicular to the plane, corresponding to the grid lines organizing the regular 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.

Fig. 4(b) illustrates one cubic cell in the spatial grid structure. Because this cell finds a location in the middle of the surface of the workpiece shape, four segments of the cell have intersection points with the surface. These points correspond to the end points of x-, y-, or z-axis-aligned dexels. Eight vertices of the cell can be classified to the internal one or external one with respect to the workpiece shape, by using the dexel information. Based on the classification, small polygons representing a part of the workpiece surface are obtained in the cell by properly connecting the intersection points. Marching cubes software, accelerated with a GPU, is used for generating polygons.

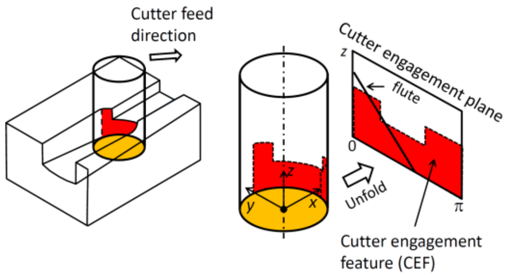

Fig. 5: Definition of the cutter engagement feature.

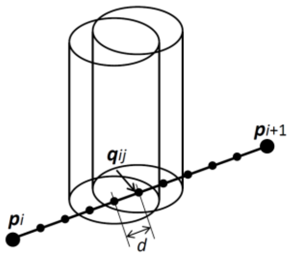

Fig. 6: Placement of multiple cutter models.

Step1 Cutter Contact Analysis with Triple-Dexel Model

Fig. 5 illustrates a milling operation with a flat-end cutter following a downward slope. A local coordinate frame is fixed on the cutter so that its 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 feed direction of the cutter. The x-axis is finally defined as the cross product of the y-axis and z-axis. The red region represents a CEF between the side surface of the cutter and the workpiece, and 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.

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 intervals, *d*. The cutter-shape models are placed on the segment so that the center point of each cutter model and a point on the segment become coincident, as shown in Fig. 6. In the following explanation, extraction of the CEF of a cutter placed on a point, $\boldsymbol{q}_{ij}$, is considered.

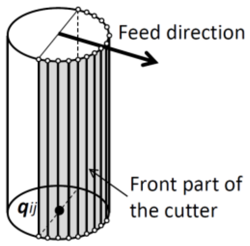

Fig. 7: Placement of segments surrounding the "front" part of the cylindrical surface of the cutter.

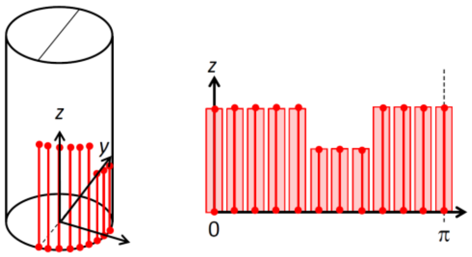

Fig. 8: CEF as a series of thin rectangles.

The CEF computation for the side surface of the cutter is explained first. The cylindrical surface of the cutter can be subdivided into two parts: the "front" part facing the feed direction of the cutter, and the "back" part. The CEF is generated only in the front part. For preparation, a set of segments parallel to the cutter axis direction is placed on the cylindrical surface so that it surrounds the front part of the surface, as shown in Fig. 7. In our current implementation, 180 segments are placed on the surface, because the circular distance between two adjacent segments becomes $r\pi/180$, where r is the radius of the cutter.

Each segment is further subdivided into a series of smaller segments. An axis-aligned bounding box (AABB) is defined for each smaller segment, then potential overlap between the segment and the cubic cells of the triple-dexel model of the workpiece shape is examined. The intersecting points between the segment and the surface polygons in the cells are computed. The obtained points and two end points of the segments are sorted, according to the spindle axis direction of the cutter, to determine the intersection portion. Such intersecting segments are arranged side-by-side, as indicated in Fig. 8; then they are replaced to thin rectangles of the same vertical length. The CEF on the side surface of the cutter is finally obtained as a series of rectangles. In our implementation of the parallel CEF computation software, one thread is assigned to each segment on the cylindrical surface of the cutter for computing the intersecting portion on the segment.

With flat-end milling, the CEF computation for the bottom surface is necessary only when the cutter moves along a downward slope. Concentric circles at an equal distance are generated in the bottom surface of a flat-end cutter. For each circle, a set of points are generated, as shown in Fig. 9(a) and (b). A total of 360 points are generated in each circle, because the circular distance between two adjacent points becomes $r\pi/180$, where r is the radius of the circle. For each point, its position with respect to the workpiece shape is checked. Three neighboring points are connected by a triangle if they are contained within the workpiece shape, and a CEF on the bottom surface of the cutter is generated as a set of triangles. One thread is assigned to each point on the bottom surface to check its position with respect to the workpiece shape.

A similar method is used for computing the CEF on the spherical surface of a ball-end cutter. Unlike the flat-end milling case, the CEF computation for the spherical surface is not limited to the milling in the downward motion. For the ball-end milling case, parallel circles on the spherical surface are used as guide-lines for generating the points, as shown in Fig. 9(c).

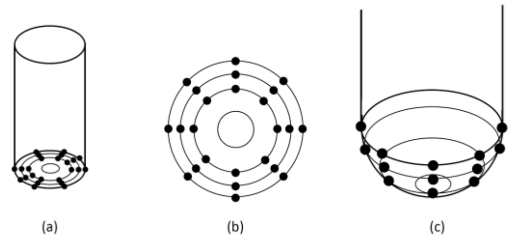

Fig. 9: Point generation on the bottom surface.

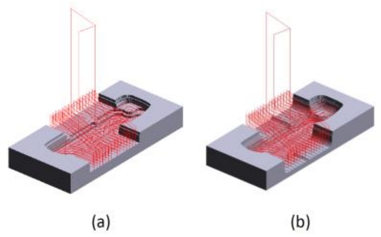

Fig. 10: Milling simulation results.

Step2 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 $\boldsymbol{p}_i$ and $\boldsymbol{p}_{i+1}$. This operation is achieved in the following two sub-steps.

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 from the operations 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.

Computational Experiments

A milling simulation system with a CEF computation capability is implemented using VisualStudio 2010 and CUDA 8.0. A 64-bit PC with an Intel Core i7 Processor (4.0GHz), 32 GB memory, and an nVIDIA GeForce GTX-1080 GPU is used in the experiments. The size of the stock object is 150 x 70 x 20mm. Triple-dexels for representing the workpiece shape is generated using three square meshes in the xy, yz, and zx planes which are obtained by projecting a spatial grid structure of 1925 x 901 x 261 resolution. The total length of the cutter path is 10,149.15mm. It consists of 9,522 linear segments. In the experiments, flat-end and ball-end cutters with 5mm radii are used. The interval distance, *d*, for computing the CEF, is set to be 2.5, 2.0, 1.5, 1.0, 0.5, 0.2, and 0.1 mm. Figs. 10(a) and (b) illustrate the milling results with a flat-end and ball-end cutter, respectively.

Fig. 11 depicts the total time required for the geometric milling simulation and CEF computation. 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 was required for computing CEFs at 100,000 cutter positions. On average, 0.12 ms was consumed for computing a single CEF. Fig. 12 displays an example of the extracted CEFs for a ball-end cutter at one milling position. Red shape in a circle represents a CEF on the spherical surface of the cutter.

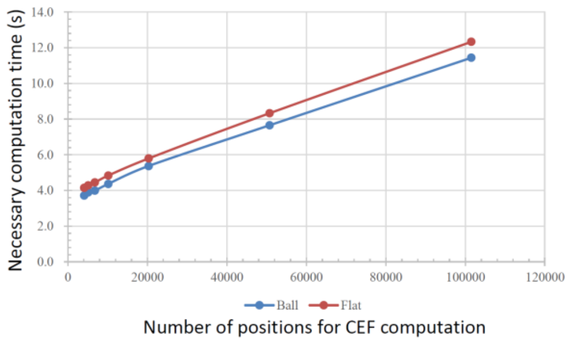

Fig. 11: Necessary time for computing CEFs.

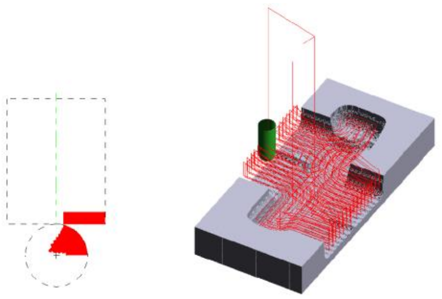

Fig. 12: CEF computation result.

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 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 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 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 was consumed for computing a single CEF.

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] 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

[3] 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