Springer

KSME

Journal of Mechanical Science and Technology 24 (5) (2010) 1019~1027
www.springerlink.com/content/1738-494x
DOI 10.1007/s12206-010-0306-7

Optimum tool path generation for 2.5D direction-parallel milling with incomplete mesh model †

Hyun-Chul Kim*

High Safety Vehicle Core Technology Research Center, Department of Mechanical & Automotive Engineering,Inje University, Gimhae 621-749, Korea

(Manuscript Received May 23, 2008; Revised May 22, 2009; Accepted June 12, 2009)

Abstract

Many mechanical parts are manufactured by milling machines. Hence, geometrically efficient algorithms for tool path generation, along with physical considerations for better machining productivity with guaranteed machining safety, are the most important issues in milling. In this paper, an optimized path generation algorithm for direction-parallel milling, a process commonly used in the roughing stage as well as the finishing stage and based on an incomplete 2-manifold mesh model, namely, an inexact polyhedron widely used in recent commercialized CAM software systems, is presented. First of all, a geometrically efficient tool path generation algorithm using an intersection points-graph is introduced. Although the tool paths obtained from geometric information have been successful in forming desired shapes, physical process concerns such as cutting forces and chatters have seldom been considered. In order to cope with these problems, an optimized tool path that maintains a constant MRR for constant cutting forces and avoidance of chatter vibrations, is introduced, and verified experimental results are presented. Additional tool path segments are appended to the basic tool path by means of a pixel-based simulation technique. The algorithm was implemented for two-dimensional contiguous end milling operations with flat end mills, and cutting tests measured the spindle current, which reflects machining characteristics, to verify the proposed method.

Keywords: 2.5D milling; Constant cutting force; Direction-parallel tool path; Material removal rate

1. Introduction

Two and a half dimensional (2.5D) milling represents a two- and three-dimensional hybrid form of milling in which only two axes are continuous-path controlled after one plane has been selected. Generally, most CNC milling can be performed using 2.5D milling. According to Harenbrock [1], more than 80% of all mechanical parts can be machined by applying this path control concept. This is partially due to the facts that a surprisingly large number of mechanical parts have 2.5 dimensions and that more complicated objects are usually produced from a billet by 2.5D roughing (and 3D-5D finishing) [2]. Hence, finding an efficient tool path for 2.5D milling tasks is one of the most important issues in CAM.

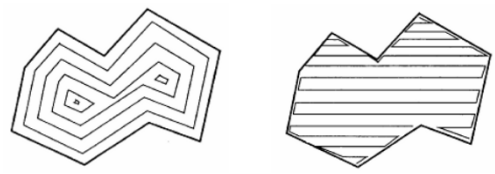
These days hundreds of commercial CAD/CAM software systems are used around the world in the metal-working industry. These systems have been developed based on different graphic kernels, bringing about the need for geometric data transfer. Generally, the geometry of a designed part is determined by surfaces that are, in most cases, defined by non-uniform rational B-splines (NURBS). The mathematical formulas that describe these NURBS are very complex, and each kernel implements those algorithms in its own way.

Although many studies have been conducted in this area and data transfer techniques have become more advanced, the fundamental differences between the numerical descriptions and accuracy of geometry still yield numerous problems: missing surface patches, distortion of boundary curves, overlapping of patches, gaps between patches, trimming errors, tangent discontinuities, and others [3]. In order to generate tool paths from a tessellated model, all design-surfaces have to be converted to meshes, but the tessellated model has numerous problems such as gaps and overlaps. Therefore, since machining areas for 2.5D milling, obtained by slicing a mesh model with guide surfaces, are not closed loops, a machining area detection process for obtaining closed loops is required.

In 2.5D milling, there exist two main types of tool path trajectory: contour parallel path (Fig. 1(a)) and direction-parallel path (Fig. 1(b)). Contour parallel path is generated by successive offsets of the input profile. Thus each successive offset is essential to the generation of the contour parallel tool path, which fact has been confirmed in studies [4, 5]. The direction-

† This paper was recommended for publication in revised form by Associate Editor Dae-Eun Kim

*Corresponding author. Tel.: +82 55 320 3988, Fax.: +82 55 324 1723
E-mail address: mechkhe@inje.ac.kr



(a) (b)

Fig. 1. (a) Contour parallel path; (b) direction-parallel path.

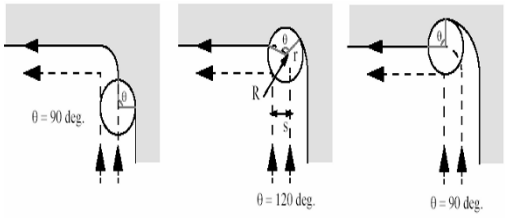


Fig. 2. Angle of engaged circumference.

parallel tool path trajectory, utilizing line segments that are parallel to an initially selected reference line, is simpler than the contour parallel path one.

In this paper, optimized tool path generation algorithms that take into account geometrical and physical considerations of direction-parallel milling are presented.

There have been many studies [1, 6-9] on direction-parallel path's geometric problems. Most of the available research has focused on minimizing the number of retractions and reducing the machining time with minimum total lengths. Held [1, 6] reported in-depth investigations into reference line selection. However, in most machining companies, the angle of inclination generally is determined according to operators' experience and subjective judgment, because these, despite their inherent limitations, are still regarded as the more optimal approaches. In the interests of providing a more certain and reliable tool path generation methodology, this paper presents a geometrically robust algorithm for direction-parallel path.

Since final products are obtained through real machining, physical consideration of geometric tool paths, which are obtained from a purely geometric perspective and cause many problems in real machining, is positively necessary.

Tlusty et al. [10] noted that radial engagement increases when a cutter moves into a corner region, but remains constant for straight path segments (Fig. 2). Consequently, momentary rises in cutting forces and chatter commonly are encountered during cornering.

In 2.5D machining, the tool path distance, s, and the angle of engagement between the tool and the workpiece, θ, must satisfy

$$\theta_{\max} = \pi - \cos^{-1}\left(\frac{2r^2 + 2R(s-r) - s^2}{2r(R-r)}\right) \quad (1)$$

where r is the tool radius and R is the radius of the arc.

Fig. 3 shows the relation between the engagement angle and

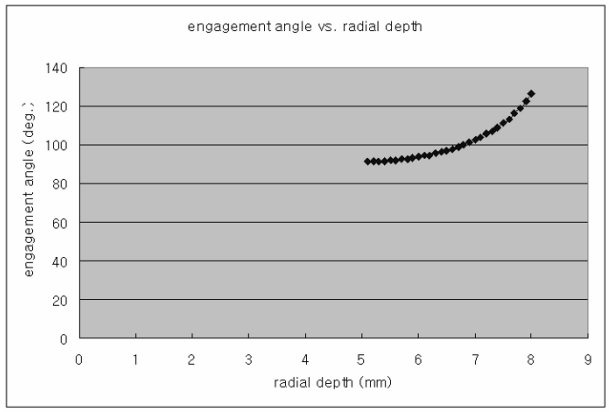


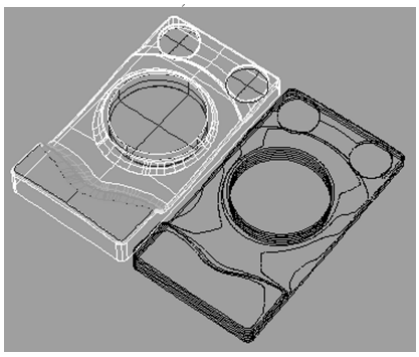
Fig. 3. Engagement angle vs. radial depth.

the radial depth of cut as s; accordingly, when the workpiece shown in Fig. 2 was machined with a 10ϕ flat end mill, the radial depth of cut increased. This increase of radial depth of cut with the increase of the engagement angle causes excessive cutting forces that incur tool breakage, tool deflection and chatter. Hence, from this physical point of view, tool paths for constant cutting force are required. In addition, Trusty, Smith, and Zamudio [10] and Smith, Cheng, and Zamudio [12] addressed the issues of stability in end milling and the importance of the proper selection of the axial and radial depths of cut for chatter-free machining. Since a given axial depth of cut is constant in 2.5D milling, the radial depth of cut plays an important role in determining stability; indeed, a change of radial depth of cut can cause cutter vibration even if cutting forces are constant.

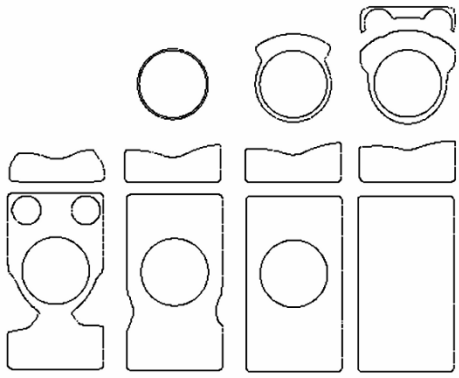
Using the feedrate scheduling approach, constant cutting forces can be achieved, but it remains impossible to keep the radial depth of cut constant for chatter-free paths without tool path modification. A more fundamental and optimum solution is to modify the tool path geometry. Therefore in this paper, direction tool paths maintaining both constant cutting forces and machining stability over the whole area of machining are considered the optimum ones.

Certainly, in order to generate optimum direction-parallel tool paths for 2.5D milling, constant cutting forces and radial depth of cut are required over the entire machining area.

Consequently in this paper, geometric tool path generation algorithms and tool path modification methods that obtain optimized paths by appending additional tool path segments to the basic tool path are presented. The remainder of this paper assumes the following order. In Section 2, a machining area detection step from an incomplete mesh model is introduced. In Section 3, geometric algorithms for direction-parallel path are presented. From the physical point of view, the necessity of a constant material removal rate (MRR) and the tool path modification method are explained in Section 4. The implementation of the method and cutting experiments are presented to verify the effect of the proposed method in Section 5. Section 6 treats the experimental results and discussion, and Section 7 offers concluding remarks.



(a) Speaker model and machining planes



(b) Intersecting lines

Fig. 4. Intersecting lines of meshes and machining planes.

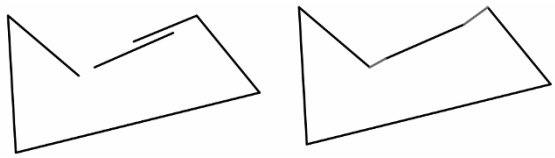
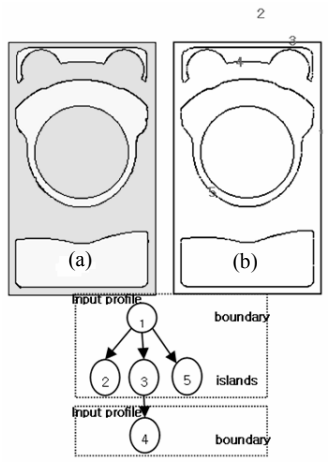


Fig. 5. Before and after adopting closed loop construction algorithm.



(c)

Fig. 6. Machining area detection.

2. Machining area detection

Direction-parallel milling takes place in different planes, but generally the Z plane, parallel to the machining plane. Therefore, machining areas can be obtained by calculating the intersections between the incomplete meshes and the machining planes. In order to calculate the intersections efficiently, the grid method [13] was applied in this study.

Fig. 4 shows a polyhedral model of a speaker and 2D lines obtained by intersecting it with horizontal planes at 5 mm intervals. Because a polyhedral model is composed of triangles or quad facets, the calculated intersection segments are lines. The obtained lines are discrete and have many gaps and overlaps, since the original meshes are incomplete. Therefore an algorithm for making closed loops is required. In order to avoid many gaps, the closest point in the allowable tolerance is searched from any point. At this time, a sudden direction change is considered, and the point is pushed in a stack for checking overlaps. If a closed loop without a sudden direction change is not constructed, a point is popped from a stack, and the process is repeated in order to obtain a closed loop. The algorithm is expressed as follows, and a simple example is illustrated in Fig. 5.

Closed Loop Construction Algorithm:

```
//Input: discrete lines with gaps and overlaps.//
//Select an arbitrary line as the start line.
While (a closed loop is not constructed)
{
if (search a closest point in allowable tolerance)
  if (check a sudden direction change)
  push the point in stack;
else
  pop in stack;
};
```

Actual areas to be machined are determined by defining the hierarchy among closed loops. Simple inclusion tests are executed for deriving all levels in a tree structure used to represent the hierarchy. Figs. 6(b) and (c) show constructed closed loops and their hierarchy. In the tree structure, every odd layer means boundaries and every even layer, islands. Areas to be machined are regions between boundaries and islands, and each is defined as an input profile for computations of direction-parallel paths, as in Fig. 6(c). In Fig. 6(a), the regions indicated with the shading colors are the areas to be machined.

3. Geometric tool path generation for direction-parallel milling

Direction-parallel tool path generation from geometric information consists of three steps: computation of intersection points, generation of intersection points-graph and graph traversal. Intersection points also are calculated by the grid method [13] to reduce computing time.

3.1 Intersection points-graph construction

The direction-parallel tool path is obtained by interconnecting intersection points calculated by intersections of parallel lines to a reference line and offset lines. Therefore intersection

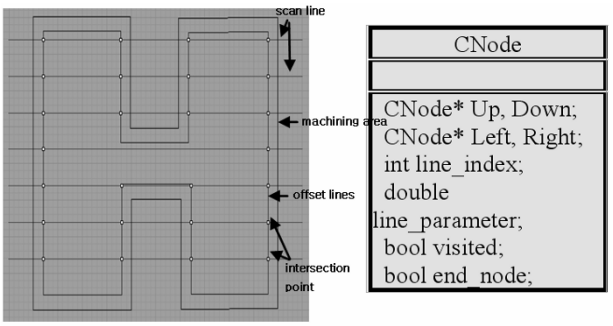


Fig. 7. Intersection points and CNode class data structure.

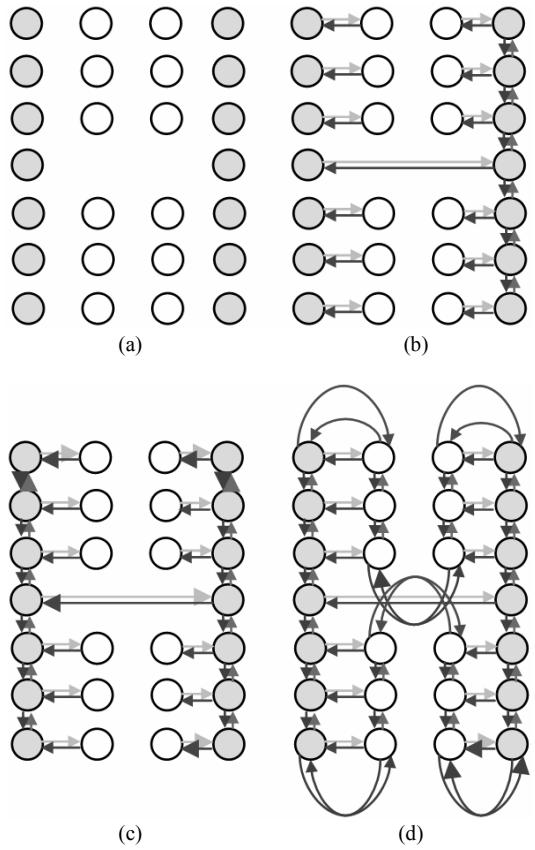


Fig. 8. Intersection points-graph construction.

points-graph construction is necessary in order to store calculated intersection points and their topology information. This process transforms the original direction-parallel tool path generation problem into a graph-theoretic problem.

Within a graph, each intersection point should have a data structure that can contain linking information. Intersection points and the CNode class data structure are illustrated in Fig. 7. Each CNode has four pointers to neighboring nodes (up, down, left and right nodes) and a boolean variable that can indicate whether or not the node is an end node representing an end point of path segments. Also, the boolean flag visited is used later in tool path generation.

Generation of an intersection points-graph is executed by the following steps.

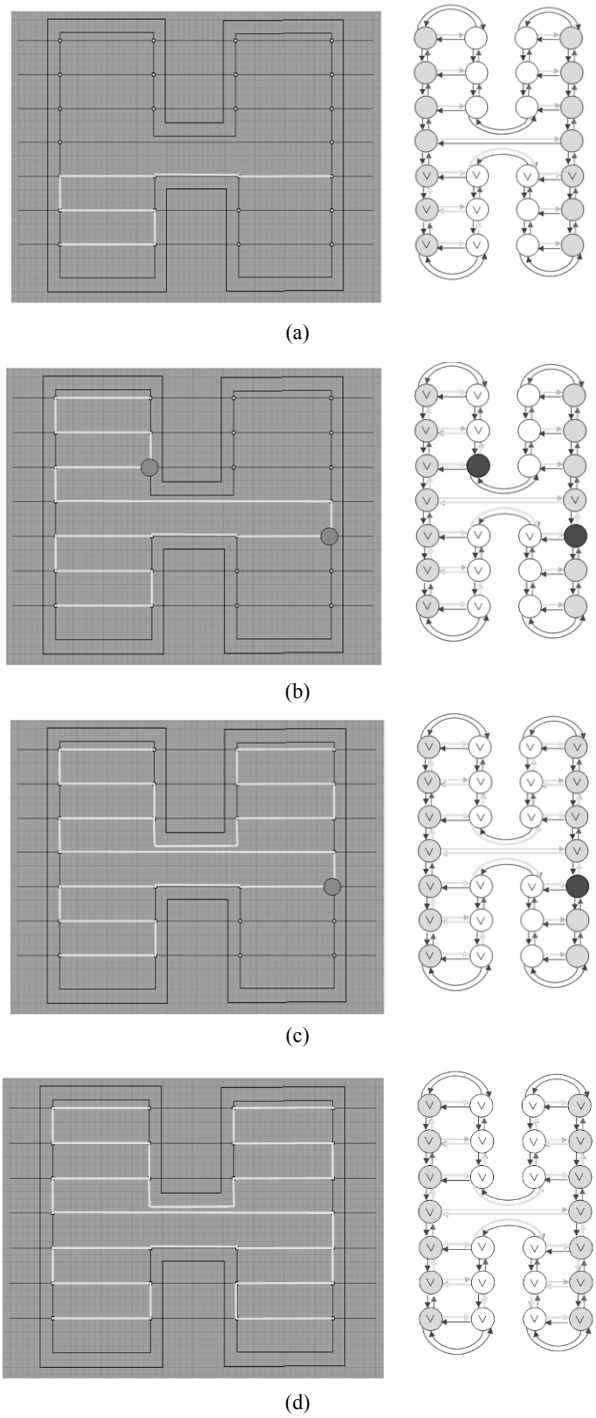


Fig. 9. Direction-parallel path generation through graph traversal.

1. indicating of end nodes (Fig. 8(a))
2. left, right connection
   A. linking from even intersection points to odd intersection points (Fig. 8(b))
3. up, down connection
   A. linking of end nodes (Fig. 8(c))
   B. linking of first detected intersection points in offset lines (Fig. 8(d))

The same steps are illustrated in Fig. 8.

3.2 Tool path generation

With the help of the graph structure, the direction-parallel tool path can be easily generated by simple traversing logic. Until there are not any nodes to move, a tool follows one path so as to minimize the number of retractions. In the case that the tool has to be lifted and moved in the air, a next node to be machined is popped from a stack. These processes are repeated until a stack is empty, and are summarized in the following five tasks.

1. set the beginning node and direction
2. check a current node as a visited node
3. traverse toward a given direction
4. transfer to the up or down node if there are no nodes to move. At that time, if there is any node not to be visited, push the node in a stack.
5. repeat the above 1-4 steps until a stack is empty

The overall algorithmic procedure is represented graphically with an example in Fig. 9. During traversing, all nodes not yet machined are pushed into the stack as shown in Fig. 9(b). The algorithms noted were implemented with C++ language on a PC and tested with various examples. Fig. 10 shows a path generation result for 2D lines obtained in a polyhedral model of a speaker. An actual machining simulation result by NC codes generation is illustrated in Fig. 11.

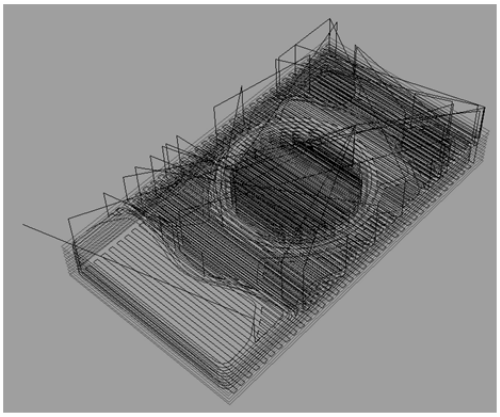


Fig. 10. Direction-parallel tool path of speaker model.

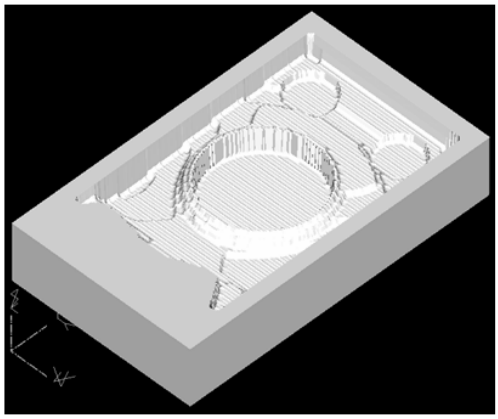


Fig. 11. Simulation result for direction-parallel path.

4. Physical consideration

As mentioned in Section 1, the direction-parallel tool path obtained from purely geometric information, though successfully used to form desired shapes, causes many problems such as tool breakage, tool deflection and chatter in real machining. There has been much research undertaken to find solutions to these problems. The literatures on controlling cutting forces and maintaining machining stability can be roughly divided into two major approaches: feedrate scheduling and tool path modification. The first approach focuses on controlling the cutting forces based on a cutting force model, specifically by adjusting feedrates so as not to exceed the reference cutting force. If the cutting force model employed is precise, constant cutting forces can be achieved. However, precise prediction of cutting force expressed as in Eq. (2) is very difficult because the specific cutting energy, k, varies according to cutting speed and radial depth.

$$F = kbt \tag{2}$$

where F is the cutting force, k is the specific cutting energy, b is the radial depth of cut and t is the axial depth of cut.

The goal of an approach by prediction of cutting forces is not to maintain the radial depth of cut as a constant. Rather, the rationale of this approach is that if the radial depth of cut is known, the feedrate can be controlled to achieve a constant cutting force.

Another control approach is modification of the tool path geometry. This method controls the radial depth of cut and cutting forces by using additional tool paths in corners. Iwabe et al. [14] attempted to reduce the chip load when cutting a corner region by using a looped cutter path, which removed the stock material incrementally in several passes. Tsai et al. [15] enhanced Iwabe's corner-looping tool path but still could not deal with complicated corner shapes. Further, there is a critical limit to for keeping cutting forces constant over the entire machining area by inserting additional paths in tool paths only in corners. As shown in Fig. 12, the direction-parallel path has a lot of overloaded areas besides corners, owing to the linking of path elements and geometric shape. Therefore direction-parallel tool paths considering both cutting forces and machining stability over the entire machining area are positively necessary.

4.1 Tool path modification for constant MRR

For the optimum direction-parallel tool path, not only cutting forces but also radial depth of cut must be maintained

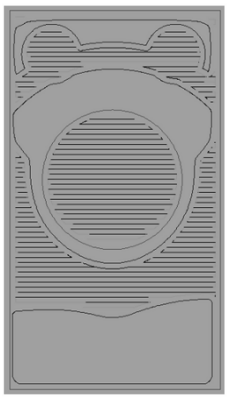


Fig. 12. Overloaded paths indicated by green lines.

constant over entire tool paths. In the case of 2.5D milling, tool paths to maintain MRR constant satisfy the conditions mentioned above. In Eq. (3), if the radial depth of cut is kept constant in fixed feedrates, the MRR would not be changed, because in 2.5D milling, the axial depth of cut is constant. In other words, with a constant MRR, one can maintain the radial depth of cut and cutting forces constant. Therefore it is important to maintain the MRR constant for optimum direction-parallel paths when considering constant cutting forces and machining stability.

$$Fv = kbtv = k \cdot MRR \tag{3}$$

where F is the cutting force, k is the specific cutting energy, b is the radial depth of cut, t is the axial depth of cut and v is the cutting speed.

In order to maintain a constant MRR at all times, additional tool paths are inserted into direction-parallel tool paths obtained by geometric algorithms. Since geometric information is changed according to geometric shape as in Fig. 12, the MRR cannot be computed analytically for arbitrary tool path trajectories. A pixel-based simulation method, however, can be used to overcome this problem. Fig. 13 illustrates the pixel-based simulation procedure for MRR calculation. As illustrated, both the geometry and the tool are discretized and represented as pixel squares. In the figure, the dark gray region represents materials remaining to be machined. The light gray region already-machined areas, and the present MRR is to be estimated with reference to the pixel numbers of the white region. The reference pixel numbers according to the nominal MRR can be computed as follows.

$$\text{reference MRR} = \left(\Delta s \big/ pixel\_resolution\right) \times \left(s \big/ pixel\_resolution\right) \tag{4}$$

where Δs is feed per tooth and s is the step over.

The size of the pixel is determined by the simulation accuracy, the most important parameter in machining simulation is accuracy. The expected error in radial depth of cut is $pixel\_size \div 2$. To achieve a simulation error of less than 5%, the following equation must be satisfied:

$$pixel\_size \div 2 \le step\_over \times 0.05 \tag{5}$$

Therefore, the size of the pixel is determined in order to improve the computation complexity as follows:

$$pixel\_size = step\_over \times 0.1 \tag{6}$$

The tool path is modified in the excessive MRR region; the procedure is shown in Fig. 14. As can be seen, as the MRR corresponding to an original path, line 1, is larger than reference MRR, a different path, line 2, is searched by changing the end point to the left of the movement direction. But, because line 2 also makes for an excessive MRR, line 3 is searched again. The process is repeated until the MRR for movement in the same direction as a previous path does not exceed the reference MRR. Finally, a return path, line 8, is appended. Correspondingly, the additional paths are lines 3, 5, 7 and 8.

MRR calculation for linear motion is shown in Fig. 15. The detailed algorithms for path modification can be expressed as follows.

Tool path modification for constant MRR:

```
//input: points [] representing closed 2D lines//
//output: modified offset curves//
for (int i=1; i<points.size (); ++i)
{
        start_point=points [i-1];
        end_point=points [i];
        Linear_Motion (start_point, end_point, pixel_num);
        If (pixel_num > reference_pixel_num)
          Additional_Path_Insertion (start_point, end_point);
}

Additional_Path_Insertion (start_point, end_point)
{
        Repeat {
            Repeat {
              change end_point to the left of moving direction;
              Linear_Motion (start_point, end_point, pixel_num);
            } Until (pixel_num <= reference_pixel_num);
            insert an additional path from start_point to end_point;
            extend an additional path with same length;
            start_point=end_point;
            replace end_point with the end point of extension line;
            Linear_Motion (start_point, end_point, pixel_num);
        } Until (pixel_num <= reference_pixel_num);
        insert a return path;
}

Linear_Motion (st_pt, end_pt, pixel_num)
{
        make boundingbox;
        while (lower_line < upper_line)
```

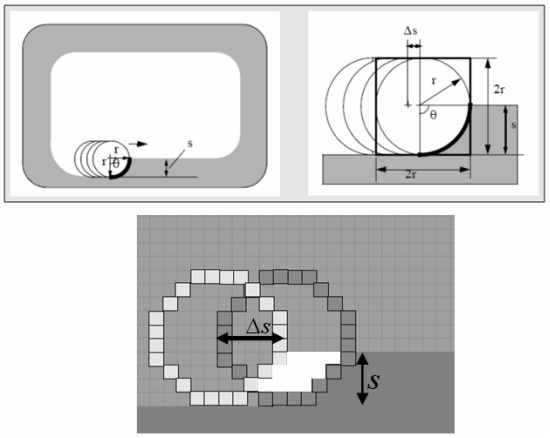


Fig. 13. Pixel-based MRR simulation.

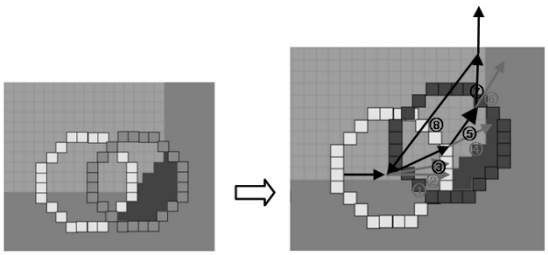


Fig. 14. Pixel-based tool path modification.

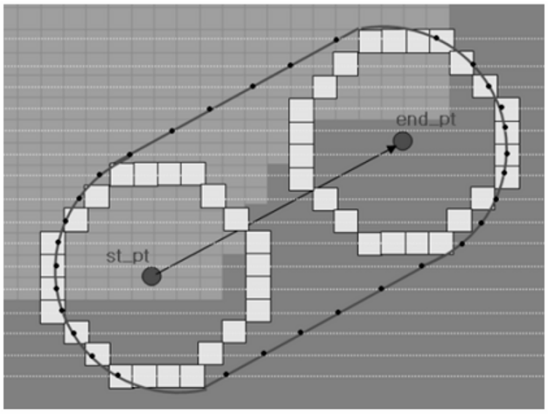


Fig. 15. MRR calculation for linear motion.

```
        {
            calculate intersection_pt;
            for ( ; st_x<end_x; ++st_x)
            if (pixel_map [st_x] [lower_line]!=visit)
            ++pixel_num;
            ++lower_line;
        }
}
```

5. Implementation and cutting experiments

The proposed path modification algorithm also was implemented with C++ language on a PC. As shown in Fig. 16, a direction-parallel cutter path generated from linear boundaries obtained by a polyhedral model, was used as the input for optimized paths. The result generated using the suggested algorithm is shown in Fig. 17. Additional paths obtained by the simulation approach consist of shot segments, and a machine tool thus could spend the majority of its time accelerating and decelerating. Therefore, in order to reduce accelerating and decelerating time, circular interpolation to the short segments was executed; the result is shown in Fig. 18.

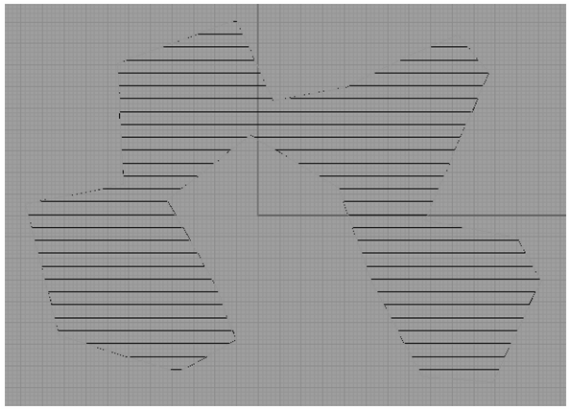


Fig. 16. Direction-parallel path used as input.

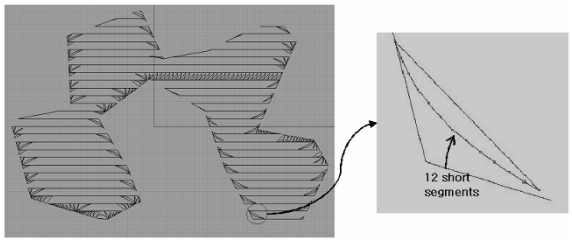


Fig. 17. Optimum path by short segments.

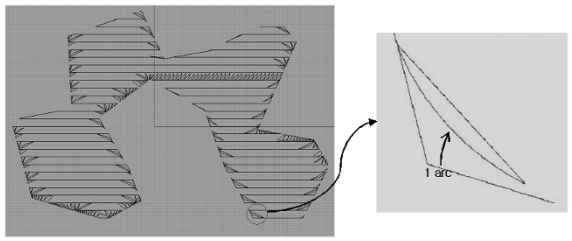


Fig. 18. Optimum path after circular interpolation.

5.1 Tool path modification for constant MRR

The spindle current was measured to investigate the real machining state and MRR, since the spindle current reflects the influences of the tool material, the characteristics of machine tools, the workpiece material and other factors [16], and thereby can be utilized in coping with machining situations. The relations between the spindle current and the cutting force and between the spindle current and the MRR, Z_M , from the equation of motion of the spindle of the machine tools, were

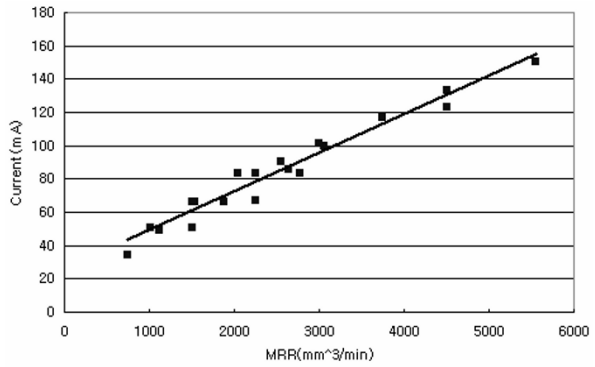


Fig. 19. Relation between current and MRR.

investigated. The relations can be simplified in linear form, as follows and as illustrated in Fig. 19 [17, 18].

$$I_M = a + bF_c \tag{7}$$

$$I_M = p_1 + H \cdot p_2 Z_M \tag{8}$$

where I_M is the spindle current, parameters a, b, p_1 and p_2 are constants related to the cutting tool, workpiece, CNC machine, and cutting conditions, respectively, F_c is the mean cutting force, and H is the coefficient from the hardness influence.

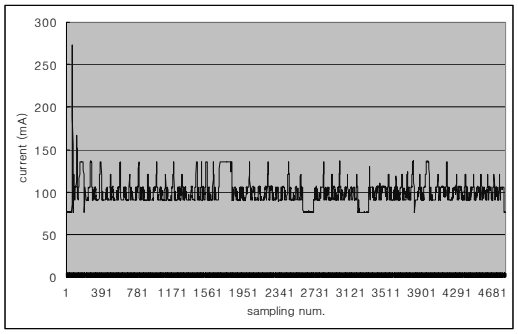
Actual cutting experiments were performed to contour parallel tool paths before and after the simulation approach was applied, in order to verify the constant MRR. The machining center is a Daewoo AL-40 with a Fanuc-0M controller. The spindle current was measured using a CNC spindle load meter. The experimental conditions are listed in Table 1 (note that the pixel size is 0.3 mm because the radial depth of cut is 3 mm).

6. Experimental results and discussion

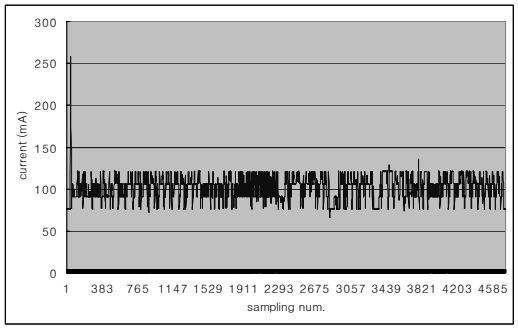
Fig. 20(a) shows the experimental results for the original direction-parallel tool paths of Fig. 16, size 120mm x 85mm, and Fig. 20(b) shows those for the algorithm-derived optimized paths. The maximum spindle current values of Fig. 20(b) are less than those of Fig. 20(a) even when the higher feedrate was applied in former case. Although the total length of the tool paths was increased, the machining time could be reduced in comparison with the original paths because the higher feedrate could be applied. As mentioned, momentary rises of cutting forces causes tool deflection, tool breakage, chatter, and other problems. To avoid unexpected tool breakage in 2.5D rough machining, which entails excessive cutting force variations, it is necessary to reduce the momentary excessive cutting force. To that end, in the practical field, a low feedrate is applied. However, in optimized paths, it is possible to apply a higher feedrate and also to reduce the momentary excessive cutting forces. Consequently, cutting forces and the radial depth of cut for stable machining situations are kept constant over the entire machining area, and moreover, the machining time can be reduced.

Table 1. Experimental conditions.

	Experimental Conditions
Workpiece	SM45C (200 x 200 x 50)
Spindle speed (RPM)	2000
Axial depth (mm)	0.5
Radial depth (mm)	3
Feedrate (mm/min)	500
Model size	120 x 85
Tool	10ϕ flat end mill (tungsten carbide tool)



(a) Machining time: 4 min 5sec (F500)



(b) Machining time: 3min 40sec (F1000)

Fig. 20. Experimental results.

7. Conclusions

This paper presents geometric algorithms for direction-parallel path, along with a simple but novel pixel-based simulation approach, for maintaining constant MRR at all times in consideration of cutting forces and machining stability.

Geometric tool path generation algorithms consist of three steps: computation of intersection points, generation of intersection points-graph, and graph traversal. Intersection points are calculated by the grid method to reduce the computing time, and an intersection points-graph is introduced to contain the calculated intersection points and their topology information. With the help of the graph structure, direction-parallel tool path can be easily generated by simple traversing logic.

In direction-parallel tool paths obtained by such geometric algorithms, varying radial depths of cut and MRR are accepted as inevitable by-products of path generation, and the worst case is that parameter selection is used inefficiently.

Although feedrate modifications have been employed to overcome this problem, such alteration cannot change radial depths, which affects machining stability.

Therefore, a simulation technique for maintaining a constant MRR over the entire machining area was introduced and implemented. Cutting tests entailing measurement of the MMR-proportional spindle current demonstrated that the MRR is kept constant at all times.

Acknowledgment

This work was supported by the 2009 Inje University research grant.

References

[1] M. Held, On the Computational Geometry of Pocket Machining, Berlin: Springer-Verlag, (1991).

[2] N. M. Patrikalakis and T. Maekawa, Shape Interrogation for Computer Aided Design and Manufacturing, Springer-Verlag Berlin Heidelberg New York, (2002).

[3] R. J. Goult and P. A. Shearar, Improving the Performance of Neutral File Data Transfers, Springer-Verlang, New York, (1990).

[4] H. Persson, NC Machining of Arbitrary Shaped Pockets, Computer-Aided Design, 10 (3) (1978) 169-174.

[5] A. Hansen and F. Arbab, An Algorithm for Generating NC Tool Path for Arbitrary Shaped Pockets with Islands, ACM Transaction on Graphics, 11 (2) (1992) 152-182.

[6] M. Held, A Geometry-Based Investigation of the Tool Path Generation for Zigzag Pocket Machining, Visual Computer, 7 (1991) 296-308.

[7] K. Tang, S. Chou and L. Chen, An Algorithm for Reducing Tool Retraction in Zigzag Pocket Machining, Computer-Aided Design, 30 (2) (1998) 123-129.

[8] S. C. Park and B. K. Choi, Tool-Path Planning for Direction-Parallel Area Milling, Computer-Aided Design, 32 (1) (2000) 17-25.

[9] M. Manuel and C. A. Rodriguez, Influence of Tool Path Strategy on the Cycle Time of High-Speed Milling, Computer-Aided Design, 35 (4) (2003) 395-401.

[10] J. Tlusty, S. Smith and C. Zamudia, New NC Routines for Quality in Milling, Annals of the CIRP, 39 (1) (1990) 517-521.

[11] T. R. Kramer, Pocket Milling with Tool Engagement Detection, J. Manufacturing Sys., 11 (2) (1992) 114-123.

[12] S. Smith, E. Cheng and C. Zamudia, Computer-Aided Generation of Optimum Chatter-Free Pockets, J. of Materials Processing Technology, 28 (1991) 275-283.

[13] D. Y. Lee, S. J. Kim, S. G. Lee and M. Y. Yang, Incomplete Mesh Based Tool Path Generation, Proceeding of the SMPE Spring Conference 2003, (2003) 844-847.

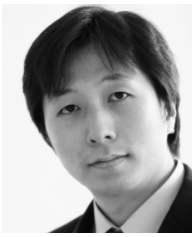
[14] H. Iwabe, Y. Fujii, K. Saito and T. Kisinami, Study on Corner Cut by End Mill Analysis of Cutting Mechanism and New Cutting Method at Inside Corner, J. of Japan Society of Precision Engineering, 99 (5) (1989) 841-846.

[15] M. D. Tsai, S. Takata, M. Inui, F. Kimura and T. Sata, Operation Planning Based on Cutting Process Models, Annals of CIRP, 40 (1) (1991) 95-98.

[16] Y. Liu, L. Zuo and C. Wang, Intelligent Adaptive Control in Milling Processes, Int. J. Computer Integrated Manufacturing, 12 (1999) 453-460.

[17] S. C. Kim and S. C. Chung, Robust Cutting Force Control Using Indirect Force and Disturbance Estimation in the End Milling Process, Proceedings of ASPE, 20 (1999) 248-251.

[18] M. Y. Yang and T. M. Lee, Hybrid Adaptive Control Based on the Characteristics of CNC End Milling, Int. J. Mach. Tools Manufact, 42 (2002) 489-499.



Hyun-Chul Kim received his B.S., M.S. and Ph.D. degrees in Mechanical Engineering from KAIST, KOREA, in 1999, 2001 and 2005, respectively. Dr. Kim is currently a Professor at the School of Mechanical & Automotive Engineering at Inje University in Gimhae, Korea. Dr. Kim's research interests include CAD/CAM, manufacturing, and precision machining.