

# Adaptive NC Simulation for Multi-axis Solid Machining

Hong T. Yau<sup>1</sup>, Lee S. Tsou<sup>2</sup> and Yu C. Tong<sup>3</sup>

<sup>1</sup>National Chung Cheng University, [imehty@ccu.edu.tw](mailto:imehty@ccu.edu.tw)

<sup>2</sup>National Chung Cheng University, [ltsou@cad.me.ccu.edu.tw](mailto:ltsou@cad.me.ccu.edu.tw)

<sup>3</sup>National Chung Cheng University, [pu@me.ccu.edu.tw](mailto:pu@me.ccu.edu.tw)

## ABSTRACT

For the machining of a complicated surface, a large amount of linear NC segments are usually generated to approximate the surface with precision. If inaccurate NC codes are not discovered until the end of cutting, time and expensive material would be wasted. However, accurate and view-independent verification of multi-axis NC machining is still a challenge. This paper emphasizes the use of adaptive octree to develop a reliable multi-axis simulation procedure which verifies the cutting route and the workpiece appearance during and after simulation. Voxel models with adaptive octree data structure are used to approximate the machined workpiece with specified resolution. Implicit functions are used to represent various cutter geometries for the examination of cutter contact points with speed and accuracy. It allows a user to do error analysis and comparison between the cutting model and the original CAD model. It can also verify the exactness of NC codes before machining is carried out by a CNC machine in order to avoid wasting material and to improve machining accuracy.

**Keywords:** NC Simulation, Solid Machining, Adaptive Voxel Model

## 1. INTRODUCTION

NC machining is a fundamental and important manufacturing process for the production of mechanical parts. Ideally, an NC machine would be running in unmanned mode. The use of NC simulation and verification is essential if programs are to be run with confidence during an unmanned operation [1]. Therefore, it is of vital importance to guarantee the exactness of NC paths before execution. From literature, NC simulation can mainly be divided into three major approaches [7], described as follows.

The first kind of approach uses direct Boolean intersections of solid models to calculate the material removal volumes during machining [2,3]. This approach is theoretically capable of providing accurate NC simulation, but the problem with using the solid modeling approach is that it is computationally expensive. The cost of simulation using constructive solid geometry is proportional to the fourth power of the number of tool movements  $O(N^4)$  [11].

The second kind of approach uses spatial partitioning representation to represent the cutter and the workpiece [4-8]. In this approach, a solid object is decomposed into a collection of basic geometric elements, which include voxels [4,5], dexels [6,7], G-buffers [8], and so on, thus simplifying the processes of regularized Boolean set operations. The third kind of approach uses discrete vector intersection [9-11]. This method is based on a discretization of a surface into a set of points. Cutting is simulated by calculating the intersection of vectors which pass through the surface points with tool path envelopes.

During multi-axis NC machining, the cutting tool frequently rotates so that it is very difficult to calculate a workpiece model that is view-dependent. Thus, in this paper, we use the voxel data structure to represent the workpiece model. But according to past literature, if precision is needed, a large number of voxels must be set up to carry out Boolean set operations. This consumes memory and time. Thus, our approach uses the octree data structure to represent the workpiece model. The octree can be adapted to create voxels with the desired resolutions that are needed. We utilize the octree to quickly search for voxels which have contact with the cutter. However, our approach uses an implicit function to represent the cutting tool because a cutter can be easily and exactly represented by implicit algebraic equations, and judging whether the cutter keeps in contact with the workpiece is also easy. Thus, our approach is reliable and precise.

The content of the paper is organized as follows. Section 2 discusses the workpiece representation using octree based voxel modes. Section 3 presents the formulation of implicit functions used to represent the geometry of various cutters.

Section 4 outlines the procedure of the proposed algorithm for 3-axis NC simulation. Section 5 shows how the proposed approach can be easily adapted to 5-axis simulation by extending the implicit functions to accommodate for the 5-axis rotation. Examples are given to demonstrate the effectiveness and simplicity of the proposed approach. Section 6 shows the experimental results of the required memory space and computation time for NC simulation. At the end, conclusions are made in Section 7.

## 2. VOXEL REPRESENTATION OF SOLID GEOMETRY

In this paper, we use a voxel data structure to represent the free form solid geometry of a milled workpiece because a voxel model has axis alignment and view-independent properties. At the same time we use the octree to avoid creating a large number of voxels. The method judges whether the cutter keeps in contact with the workpiece, finds all voxels which have such contact, and then subdivides these voxels into eight voxels in space recursively until the resolution reaches the desired precision level. Thus, if there is no voxel in contact with the cutter, there is no need to subdivide the voxel. Fig. 1. shows the Octree data structure and the voxel model it represents.

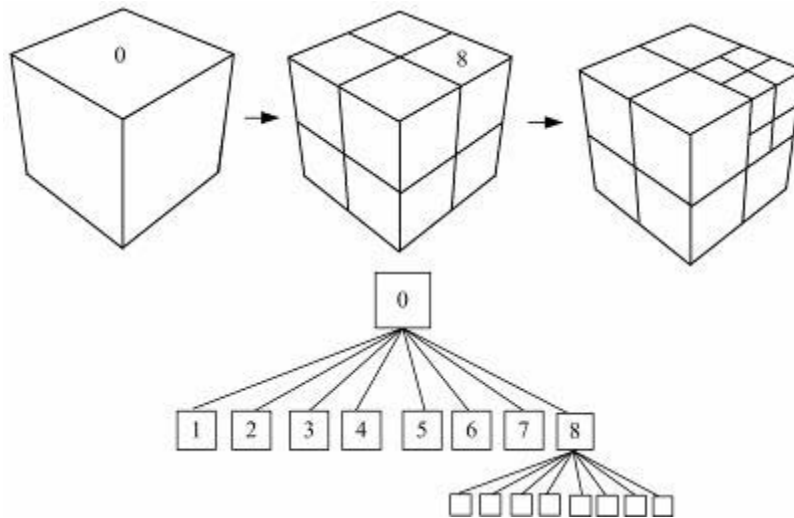


Fig. 1. Octree data structure and the associated voxel model

Traditionally, since machining simulation uses the uniform voxel data structure to represent a milled workpiece, when precision increases, great quantity of voxel data will be produced to represent the workpiece. This will make the machining simulation slow because a lot of computer memory is needed. Thus, we use the octree data structure to adaptively create voxels as needed during simulation.

## 3. REPRESENTATION OF CUTTER GEOMETRY USING IMPLICIT FUNCTIONS

The spatial partitioning approach with uniform voxels has failed to address multi-dimensional NC verification for workpieces of comparable complexity and accuracy. If high precision is needed, a large number of voxels must be set up to carry out Boolean set operations. This will consume considerable amount of memory and time. But our new approach for three-axis simulation uses the adaptive voxel model to represent a milled workpiece, and uses implicit functions to represent a solid model of the cutter. Since the cutter is not decomposed into a collection of basic geometric elements, high accuracy can be achieved. At the same time the workpiece model makes use of the octree model to reduce the number of unnecessary voxels. In the following, we describe the representation of various cutters using implicit functions.

Flat endmills can be represented by a cylinder. Fig. 2. shows the flat endmill aligned with the direction of cutting. Assuming the tool is parallel to the z-axis, and the coordinate system is translated such that the center point is located at the origin. Thus, the implicit function of a flat endmill is:

$$F(X, Y, Z) = \max \left\{ \text{abs} \left( Z - \frac{L}{2} \right) - \frac{L}{2}, X^2 + Y^2 - R^2 \right\} \quad \text{if } Z \geq 0 \quad (1)$$

This implicit function is used to determine whether a voxel is inside, outside, or intersected with the cutter without losing any accuracy. The judgment can be made by inserting the coordinates of the vertices of a voxel into the implicit function. Eqn. (2) describes the relationship between a vertex and the cutter, which is also illustrated in Fig. 3.

$$F(X, Y, Z) \begin{cases} < 0 & \text{lie inside the surface} \\ = 0 & \text{lie on the surface} \\ > 0 & \text{lie outside the surface} \end{cases} \quad (2)$$

where

R : the cutter radius

L : the distance measured from the center point along the cutter axis

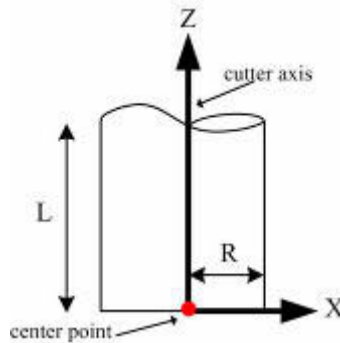


Fig. 2. Flat endmill and the associated coordinate system

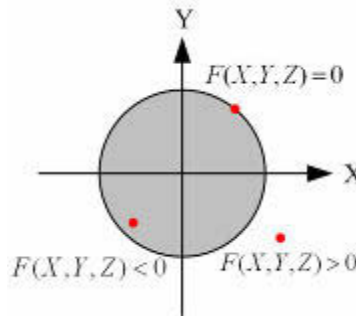


Fig. 3. Implicit function used to determine the interior or exterior of a cutter

Ball endmills can be represented by the union of a cylinder and a sphere, as shown in Fig. 4. Assuming the tool is parallel to the z-axis, and the origin of the coordinate system is translated to the center of the sphere, the implicit function of a ball endmill can be described as:

$$F(X, Y, Z) = \begin{cases} \max \{ \text{abs}(Z) - L, X^2 + Y^2 - R^2 \} & \text{if } Z \geq 0 \\ X^2 + Y^2 + Z^2 - R^2 & \text{otherwise} \end{cases} \quad (3)$$

where

R : the cutter radius

L : the distance measured from the center point along the cutter axis

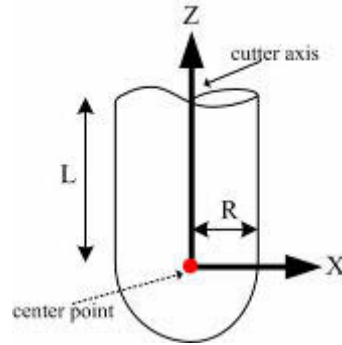


Fig. 4. Ball endmill and the associated coordinate system

Fillet endmills can be represented by the union of two cylinders and a torus. Assuming the tool is parallel to the z-axis, and the coordinate system is translated to the center point as shown in Fig. 5., the implicit function of a fillet endmill can be derived as:

$$F(X, Y, Z) = \begin{cases} \max \{ \text{abs}(Z) - L, X^2 + Y^2 - (R + r)^2 \} & \text{if } Z \geq 0 \\ \max \{ \text{abs}(Z) - r, X^2 + Y^2 - R^2 \} & \text{else if } \text{abs}(X) \leq R \\ (X^2 + Y^2 + Z^2 + R^2 - r^2)^2 - 4R^2(X^2 + Z^2) & \text{otherwise} \end{cases} \quad (4)$$

where

R : the radial distance from the cutter axis to the cutter corner center

r : the cutter corner radius

L : the distance measured from the center point along the cutter axis

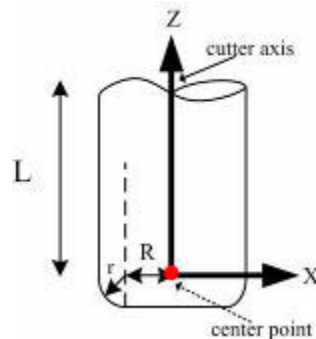


Fig. 5. Fillet endmill and the associated coordinate system

As simple as a flat endmill, or as complex as a fillet endmill, the implicit functions can be used to decide whether a point is inside or outside a cutter by a direct application of Eqn. (2). The use of implicit functions to represent a cutter is not only precise in geometry, simple in concept, and the programming of implicit functions is also very easy and straightforward. Thus, we can easily know the geometric relationship between the stock and the cutter by using the implicit function  $F(X, Y, Z)$ .

#### 4. SIMULATION OF THREE-AXIS NC MACHINING

After the implicit function of the cutting tool is formulated, an important task that needs to be performed is to determine which voxels need to be subdivided or deleted during the milling process. Fig. 6. shows the flow chart of a three-axis NC path simulation.

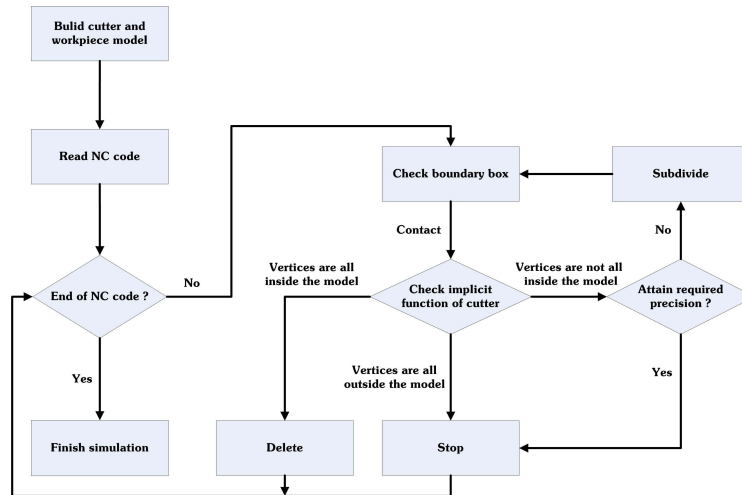


Fig. 6. Flow chart of three-axis machining simulation

The process of a three-axis NC path simulation is described as follows.

(1) The first step is to read the NC codes. Then for each NC segment, we can get the start and end cutter location (CL) points of a tool motion. Thus, the three-axis motion is modeled by a joint interpolation of CL points of the configuration of any two tool positions.

(2) The bounding box of the cutter is used preliminarily to judge which part of a voxel model the cutter is in contact with. The purpose is to get rid of voxels not in contact with the cutter. If it is determined that a voxel is in contact with the cutter, the voxel vertices will be substituted into the implicit function of the cutter to decide if the voxel vertices lie inside or outside the cutter.

(3) If all of the voxel vertices meet the conditions  $F(X, Y, Z) < 0$ , it is confirmed the voxels have been totally cut by the cutter; that is to say that the voxel falling in the cutter should be eliminated. If part of the vertices fall inside and the others outside, it means the voxel needs to be further divided. For each subdivided voxel, step (2) and step (3) will be carried out recursively until a predetermined precision level is reached.

After the current segment of the NC path is finished, step 1 is performed again, and the next segment of the NC code is read. The procedure is followed to the end until all NC codes have been read.

In the procedure for three-axis machining simulation mentioned above, it is clear that voxels are subdivided as needed according to the geometric relationship between the cutter and the workpiece; a large number of voxels are not created all at once in the beginning. Fig. 7. shows that voxels in contact only with a ball endmill will be subdivided by the octree. Voxels not in contact with the cutter will not be subdivided. Thus, this approach greatly reduces the number of voxels and saves memory during simulation.

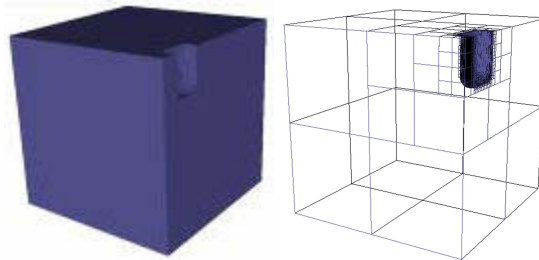


Fig. 7. Three-axis NC machining simulation.

In comparison with z-map model, NC simulation using voxel model can display multi-axis machining result. The dixel model has the restriction of being view-dependent, but the voxel model does not have this restriction. Thus, three-axis, five-side machining can utilize the three-axis simulation method of this paper, as Fig. 8. shows.

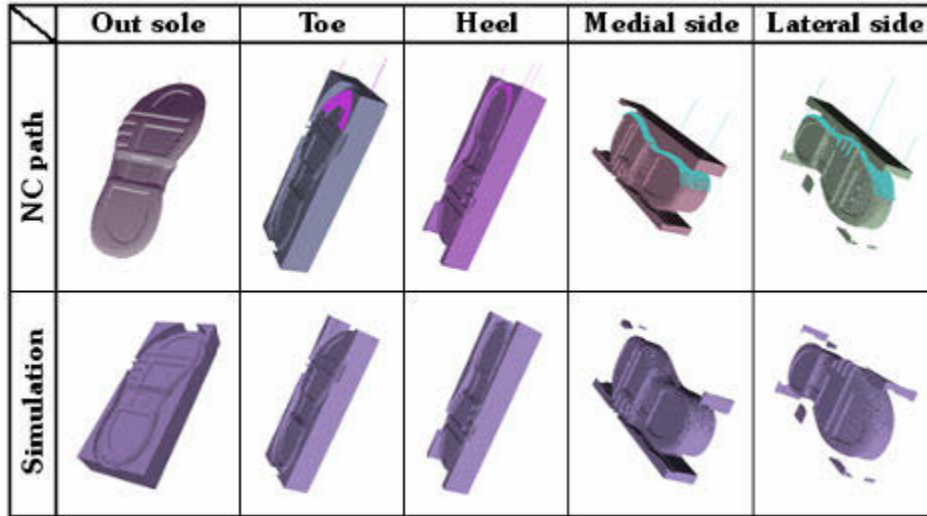


Fig. 8. Example of three-axis and five-side simulation

**5. SIMULATION OF FIVE-AXIS NC MACHINING**

In five-axis NC machining, in addition to the three translation movements, the tool axis can also be rotated. Therefore, our approach to five-axis simulation only revises the implicit function of the cutter. All other procedures are the same as the three-axis simulation. Thus, the three kinds of cutters can be expressed as follows:

Flat endmills can be represented by a cylinder. Fig. 9. shows that the tool axis is along  $\{\hat{n}\}$  and the center point is located at  $\{p\}$ . Thus, the implicit function of a flat endmill is:

$$F(X, Y, Z) = -(\{x\} - \{p\})^T [\vec{n}]^2 (\{x\} - \{p\}) - R^2 \quad \text{if } 0 \leq \{\hat{n}\}^T (\{x\} - \{p\}) \leq L \tag{5}$$

where

$R$  : the cutter radius

$\{x\} = \{X \ Y \ Z\}^T$  : the position of a voxel vertex

$\{\hat{n}\} = \{n_x \ n_y \ n_z\}^T$  : the unit vector of the tool axis

$\{p\} = \{p_x \ p_y \ p_z\}^T$  : the center point

$$[\vec{n}] = \begin{bmatrix} 0 & -n_z & n_y \\ n_z & 0 & -n_x \\ -n_y & n_x & 0 \end{bmatrix} \quad [\vec{n}]^2 = \begin{bmatrix} n_x^2 - 1 & n_x n_y & n_x n_z \\ n_x n_y & n_y^2 - 1 & n_y n_z \\ n_x n_z & n_y n_z & n_z^2 - 1 \end{bmatrix}$$

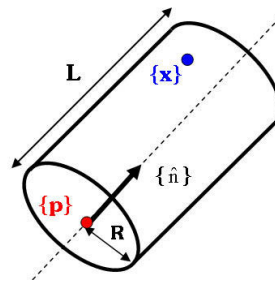


Fig. 9. Flat endmill rotated in five-axis mode

Ball endmills can be represented by the union of a cylinder and a sphere. Fig. 10. shows that the tool axis is along  $\{\hat{n}\}$  and the center point is located at  $\{p\}$ . Thus, the implicit function of a ball endmill is:

$$F(X, Y, Z) = \begin{cases} -(\{x\} - \{p\})^T [\hat{n}]^2 (\{x\} - \{p\}) - R^2 & \text{if } 0 \leq \{\hat{n}\}^T (\{x\} - \{p\}) \leq L \\ (\{x\} - \{p\})^T (\{x\} - \{p\}) - R^2 & \text{otherwise} \end{cases} \quad (6)$$

where

- $R$  : the cutter radius
- $\{\hat{n}\}$  : the unit vector of the tool axis

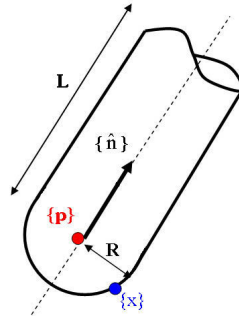


Fig. 10. Ball endmill rotated in five-axis mode

Fillet endmills can be represented by the union of two cylinders and a torus. Fig. 11. shows that the tool axis is along  $\{\hat{n}\}$  and that the center point is located at  $\{p\}$ . Thus, the implicit function of a round endmill is:

$$F(X, Y, Z) = \begin{cases} (-\{v\}^T [\hat{n}]^2 \{v\}) - (R+r)^2 & \text{if } 0 \leq \{\hat{n}\}^T \{v\} \leq L \\ (-\{v\}^T [\hat{n}]^2 \{v\}) - R^2 & \text{else if } 0 \leq \{\hat{n}\}^T (\{v\} - r\{\hat{n}\}) < r \text{ and} \\ & (-\{v\}^T [\hat{n}]^2 \{v\}) \leq R^2 \\ \left( (-\{v\}^T [\hat{n}]^2 \{v\}) + (\{\hat{n}\}^T \{v\})^2 + R^2 - r^2 \right)^2 + 4R^2 (\{v\}^T [\hat{n}]^2 \{v\}) & \text{otherwise} \end{cases} \quad (7)$$

where

- $R$  : the radial distance from the cutter axis to the cutter corner center
- $r$  : the cutter corner radius
- $\{\hat{n}\}$  : the unit vector of the tool axis
- $\{v\} = \{x\} - \{p\}$

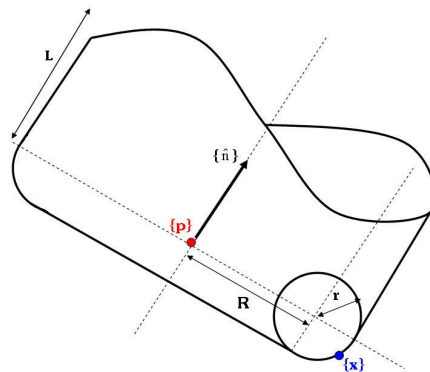


Fig. 11. Fillet endmill rotated in five-axis mode

Fig. 12. shows a simple example of a cutter axis rotated from 70 degrees to 50 degrees about the x-axis and moved along the x-axis. Fig. 13. shows the tool paths and simulation process used for five-axis machining of an impeller. Fig. 14. shows another example of five-axis machining simulation of a blade.

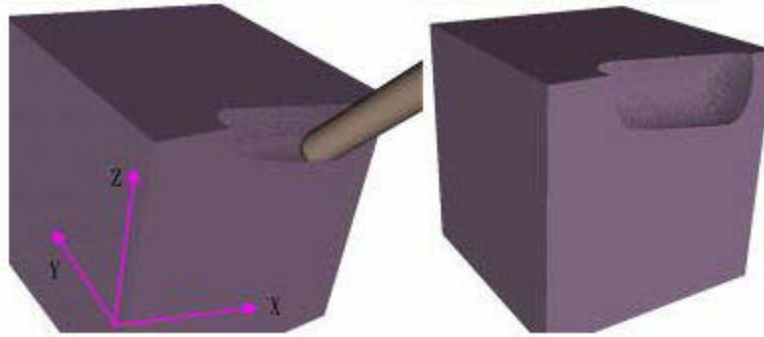


Fig. 12. Five-axis machining simulation.

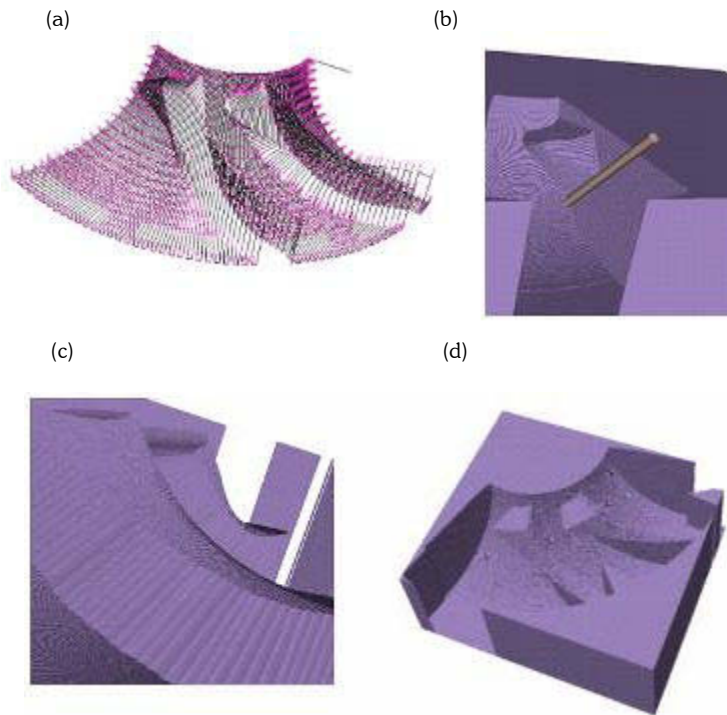


Fig. 13. Example of impeller in five-axis simulation. (a) Tool paths. (b)(c) In-process workpiece with a cutter. (d) Finished part.



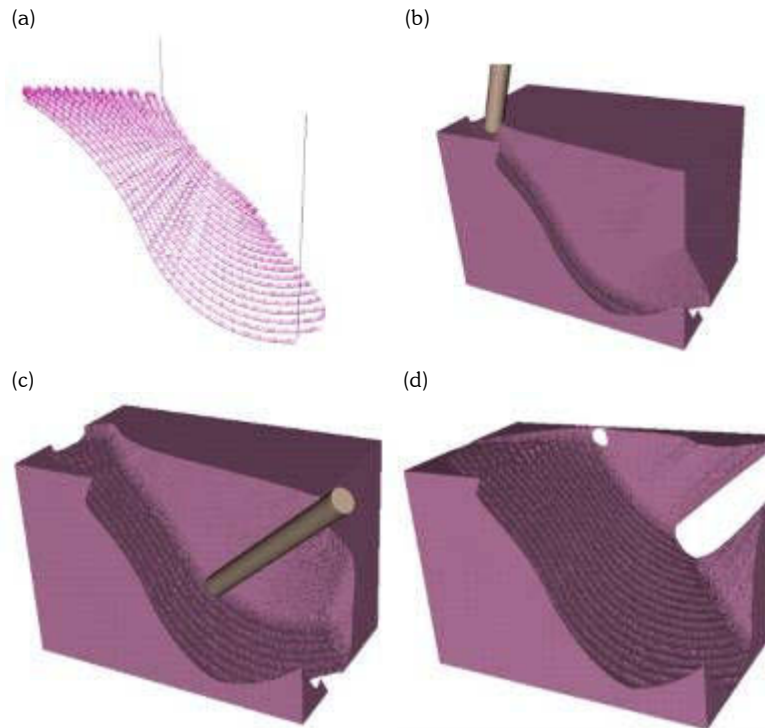










Fig. 14. Example of blade in five-axis simulation. (a) Tool paths. (b)(c) In-process workpiece with a cutter. (d) Finished part.

## 6. EXPERIMENTAL RESULTS

The proposed method has been implemented in C++ and some test cases were run on a 2.4 GHZ Pentium 4 computer. Tab. 1. gives a comparison of the required memory space and computation time for adaptive NC simulation. The first row shows the pictures of NC path for four different models. The second and third rows show the information of NC code. The fourth row is the resolution of the workpiece model. The cutter models are presented by implicit functions exactly, so there is no accuracy issue here. The fifth row shows the types of cutter being used. The last three rows are the required memory space, computation time, and the rendered simulation result.

Part	Impeller	Blade	Shoe	Bottle
NC path				
Number of NC code (line)	2654	2536	74803	3340
Length of NC code (mm)	24950	8642	27287	3060
Resolution (mm)	0.1	0.1	0.1	0.1
Cutting tool	Flat R3	Ball R10	Flat R1.5	Ball R1
Computation time (Sec)	562	387	296	209
Memory space (MB)	192	101	67	132
Simulation result				

Tab. 1. Required memory space and computation time for adaptive NC simulation.

Tab. 2. gives a comparison of the required memory space and computation time for NC simulation using uniform voxel models. The parameters remain the same as adaptive NC simulation. Under this condition, we are not able to simulate case1 (Impeller) and case3 (shoe) because such cases exceed our memory limitation. The results that can be observed are case 2 (blade) and case 4 (bottle). By comparison, the advantage of the adaptive NC simulation is clear. A great reduction of time and space can be achieved by using the adaptive NC simulation.

<b>Part</b>	<b>Impeller</b>	<b>Blade</b>	<b>Shoe</b>	<b>Bottle</b>
<b>Computation time (Sec)</b>	X	1491	X	721
<b>Memory space (MB)</b>	X	414	X	380

Tab. 2. Required memory space and computation time for NC simulation with uniform voxel model.

## 7. CONCLUSION

In this paper, we proposed a novel multi-axis simulation method. The objective of this paper was to use the adaptive voxel model to develop a reliable multi-axis simulation procedure which can simulate the cutting route and the workpiece appearance during and after the simulation. It allows the user to do error analysis and comparison between the cutting model and the original CAD model. It can verify the accuracy of NC codes before machining on a CNC machine in order to avoid wasting material and to improve machining accuracy.

In summary, the advantages of the multi-axis simulation method presented in this paper are as follows. (1) The simulation method uses less memory than other voxel-based simulation methods. (2) The simulation is view-independent. The dixel model has the restriction of being view-dependent, but the voxel model does not have this restriction. (3) The simulation is reliable and accurate. Regardless of whether it is the five-axis or three-axis simulation which uses the implicit function to represent a cutter, the whole method is simple and reliable.

## 8. REFERENCES

- [1] Choi, B. K., Jerard, R. B., *Sculptured Surface Machining: Theory and Applications*, Kluwer Academic Publishers, 1998.
- [2] Wang, W. P., Wang, K. K., *Geometric Modeling for Swept Volume of Moving Solids*, IEEE Computer Graphics & Applications, Vol. 6, No.12, 1986, pp 8-17.
- [3] Atherton, P. R., *A Scan-Line Hidden Surface Removal Procedure for Constructive Solid Geometry*, Computer Graphics, Vol. 17, No. 3, 1983, pp 73-82.
- [4] Kawashima, Y., Itoh, K., Ishida, T., Nonaka, S., Ejiri, K., *A Flexible Quantitative Method for NC Machining Verification Using a Space-Division Based Solid Model*, The Visual Computer, Vol. 7, 1991, pp 149-157.
- [5] Jang, D., Kim, K., Jung, J., *Voxel-Based Virtual Multi-Axis Machining*, Advanced Manufacturing Technology, Vol. 16, No. 10, 2000, pp 709-713.
- [6] Van Hook, T., *Real Time Shaded NC Milling Display*, Computer Graphics, Vol. 20, No. 4, 1986, pp 15-20.
- [7] Huang, Y., Oliver, J. H., *Integrated Simulation, Error Assessment, and Tool Path Correction for Five-Axis NC Milling*, Journal of Manufacturing Systems, Vol. 14, No. 5, 1995, pp 331-334.
- [8] Saito, T., Takahashi, T., *NC Machining with G-buffer Method*, Computer Graphics, Vol. 25, 1991, pp 207-216.
- [9] Chang, K. Y., Goodman, E. D., *A Method for NC Tool Path Interference Detection for A Multi-Axis Milling System*, ASME Control of Manufacturing Process, DSC-Vol.28/PED-Vol.52, 1991, pp 23-30.
- [10] Oliver, J. H., Goodman, E. D., *Direct Dimensional NC Verification*, Computer-Aided Design, Vol. 22, No. 1, 1990, pp 3-10.
- [11] Jerard, R. B., Drysdale, R. L., Hauck, K., Schaudt, B., Magewick, J., *Methods for Detecting Errors in Numerically Controlled Machining of Sculptured surface*, IEEE Computer Graphics & Applications, Vol. 9, No.1, 1989, pp 26-39.
- [12] Yamaguchi, K., Kunii, T., Fujimura, K., *Octree-Related Data Structures and Algorithms*, IEEE Computer Graphics Appl, Vol. 8, No. 3, 1984, pp 8-68.
- [13] Glaeser, G., Gröller, E., *Efficient Volume Generation During the Simulation of NC-Milling*, Mathematical Visualization, 1997, pp 315-328.
- [14] Chung, Y. C., Park, J. W., Shin, H., Choi, B. K., *Modeling the Surface Swept by a Generalized Cutter for NC Verification*, Computer-Aided Design, Vol. 30, No. 8, 1996, pp 587-94.