Virtual Five-Axis Milling Machine: Tool Path Generation and Simulation

Mud-Armeen Munlin
Department of Information Technology,
Sirindhorn International Institute of Technology,
Thammasat University, Rangsit Campus,
Pathum Thani, 12121, Thailand,
amin@siit.tu.ac.th

Abstract

This paper presents the algorithms to generate and simulate non-linear tool path of the five-axis milling machine. The simulator is based on 3D representation and employs an inverse kinematics approach to derive the corresponding rotational and translation movement of the mechanism. The simulator makes it possible to analyze an accuracy of a 3D tool path based on a prescribed set of the cutter location (CL) points as well as a set of the cutter contact (CC) points and the tool inclination angle. The resulting trajectory of the tool path is not unique and depends on the initial set up of the machine which in turn is problem dependent. Furthermore, the simulator can be used to simulate the milling process, verify the final cut and estimate the errors of the actual tool path before the real workpiece is actually cut with the real machine. Thus, it reduces the cost of iterative trial and error.

Tool path generation and simulation is verified by a series of cutting experiments performed by means of the proposed software and the accuracy of milling is estimated. It has been shown that the proposed graphical 3D software presents an efficient interactive approach to the modification of a tool path based on an appropriate set of transformations as well as verification of the tool path optimization algorithms. The result of the simulation has been tested using the Maho600E 5-Axis Milling Machine at Computer Integrated Manufacturing Laboratory at the Asian Institute of Technology.

Keywords: Inverse kinematics, five-axis CNC machines, Tool path simulation and optimization.

1. Introduction

Simulation of 5-axis milling is a must. The software must include tool path tracing, dynamic display of all moving elements combined with a realistic solid modeling. This paper introduces software for constructing a tool path (G-Code) from given CL-points and simulating the actual tool trajectory of the cutting operations of a five-axis milling machine. The tool path is constructed based on the kinematics of the machine. The simulation process employs the 3D geometric modeling approach derived in the framework of inverse kinematics of the five-axis milling machine. The proposed software constitutes a basis to develop a solid modeling system for simulation, verification and optimization of the cutting operations. In particular, the system allows for an efficient simulation of kinematics of multi axis-milling machines. Moreover, it is possible to build a virtual environment that enables the user to interactively evaluate the kinematics of the mechanism and estimate the geometric errors. Several physical phenomena, such as machine kinematics, thermal effects, static and dynamic loading, common-cause failures often affect the quality of the surfaces produced by five-axis machining. However, the particular effect of machine kinematics-geometric errors seems to be the most significant [1,2].

Consider a typical configuration of the five-axis milling machine with the rotary axis on the table (Fig.1). The machine is guided by axial commands $\Pi=(W,\Re)\in\mathbb{R}^5$ carrying the three spatial coordinates $W\equiv(x,y,z)$ of the tool tip in the machine coordinate and the two rotation angles $\Re=(a,b)$. The tool path $\Pi=\{\Pi_0,\Pi_1,...,\Pi_m\}$ is a sequence of coordinates in the five- dimensional space. The spatial coordinates of the tool path usually (but not necessarily) lie on the required surface $S\equiv S(u,v)$. Usually, the tool visits the

positions Π_p following a structured spatial pattern such as a zigzag or a spiral pattern. However, the path could be also composed from a variety of unconventional patterns and include tool retraction [3].

In order to ensure a prescribed tolerance, the standard CAM software estimates the local errors and incorporates additional points (if applicable) into a single G-Code block. However, such a strategy invokes a substantial increase of the CL-points and consequently a substantial increase of the machining time [4]. Therefore, recent papers have displayed a number of sophisticated methods to optimize a zigzag or spiral pattern combined with techniques dealing with the geometric complexity of the workpiece [5,6]. Besides, there exist a variety of off-line methods to generate a suitable non-uniform tool-path, for instance: the neural network modeling approach [7] and the Voronoi diagram technique [8]. Verification of this method requires an up to date software involving appropriate non-linear kinematics as well as a solid model of the milling machine.

A full optimization scheme involves a model of cutting operations, topologies of the prescribed tool path patterns and an optimization procedure. Let p_c be the parameters related to the configuration of the machine (such as coordinates of the centers of rotation, workpiece offset relative to the machine coordinates, etc.) and p_t the parameters related to the tool (such as the diameter, length, shape, etc). The model of the cutting operations, being fed with p_c, p_t, S and Π, produces a result of machining, namely, the output surface $T \equiv T(u,v)$. The optimization is usually performed with regard to Π and p_t. The cutting operations could be optimized with regard to the machine configuration p_c as well. However, the optimal machine is often a purely theoretical issue [9]. Let $S \equiv S(u,v)$ be the required surface. The general optimization problem is then formulated by

$$\underset{\Pi,\, p_t}{\text{minimize}} \; \|\varepsilon\|, \qquad (1)$$

where ε denotes the cost function representing the error given by

$$\varepsilon(u,v) \equiv |S(u,v) - T(u,v)|, \qquad (2)$$

where $\|\ \|$ is an appropriate norm. Optimization (1) is subjected to the following constraints:

1) The scallop height constraint. The scallop between the successive tool tracks must not exceed the prescribed tolerance [10].

2) The local accessibility constraint. The constraint insures against the removal of an excess material when the tool comes in contact with the desired surface due to the so-called curvature interference and the surface interference [4,12].

3) The global accessibility constraint. The constraint ensures that the tool does not come in contact with either machine parts (collision detection) or unwanted parts of the desired surface [6].

Given the general context above, we tackle a particular but important problem of optimization and simulation of the rotation angles in the vicinity of stationary points of the desired surface. It should be noted that there have been a variety of research focused on the *orientation of the cutting tool*. However, the accuracy is also affected by *the way the orientations are being achieved*. In other words, the kinematics error depends not only on the characteristics of the surface versus the tool orientation but on the previous rotations as well. It is not hard to demonstrate that the history of rotations becomes particularly important in the vicinity of the stationary points of the desired surface. However, to the best of our knowledge such analysis is not provided by commercial CAD/CAM software such as Unigraphics, EdgeCam, Vericut, etc.

Therefore, we propose a global optimization procedure to minimize the kinematics error [11] with regard to the feasible angles performed in the vicinity of the stationary points as well as to simulate such errors using the actual tool trajectory generated by our post processor. The optimization is performed within our post processor and the tool trajectory is calculated by applying the inverse transformation from the machine coordinate back to the workpiece coordinate system as described in the next section.

2. Tool Path Generation

Consider a typical configuration of the five-axis milling machine with the rotary axis on the

table as shown in Fig.1. Recall that the machine is guided by axial commands carrying the three spatial coordinates of the tool tip in the machine coordinate system M and the two rotation angles. The CAM software generates a set of successive coordinates called cutter location points or CL-points (X,Y,Z,I,J,K) in the workpiece coordinate system W. Typically, the CAM software distributes the CL-points along a set of curves, which constitutes the so-called zigzag or spiral pattern. A post processing which includes a transformation into the M-system generates a set of machine axial commands which provide the reference inputs for the servo-controllers of the milling machine.

Consider how the axial command translates the centers of rotation and simultaneously rotates the W-coordinates. Let W_p and W_{p+1} be two successive spatial positions belonging to the tool path and $\Re_p$, $\Re_{p+1}$ the corresponding rotation angles. In order to calculate the tool trajectory between W_p and W_{p+1} we, first invoke the inverse kinematics [5,9] to transform the part-surface coordinates into the machine coordinates $M_p \equiv (X_p, Y_p, Z_p)$ and $M_{p+1} \equiv (X_{p+1}, Y_{p+1}, Z_{p+1})$. Second, the rotation angles $\Re \equiv \Re(t) = (a(t), b(t))$ and the machine coordinates $M \equiv M(t) \equiv (X(t), Y(t), Z(t))$ are assumed to change linearly between the prescribed points, namely:

$$\begin{aligned} M(t) &= tM_{p+1} + (1-t)M_p, \\ \Re(t) &= t\Re_{p+1} + (1-t)\Re_p, \end{aligned} \qquad (3)$$

where t is the fictitious coordinate ($0 \le t \le 1$). Finally, invoking the transformation from M back to W (for every t) yields $W(t) \equiv (x(t), y(t), z(t))$. Note that, $W(t)$ represents the actual tool trajectory in the workpiece coordinate. Our tool path simulation is therefore, different form the commercial CAD/CAM systems where they simulate the tool path directly from the CL-points and hence it does not represent the actual tool trajectory. As a result, kinematics errors can not be detected and minimized.

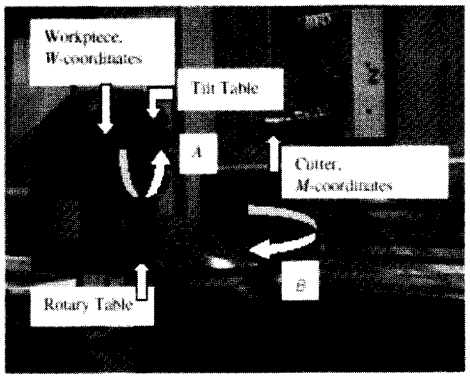


Figure 1. Five-axis milling machine

The kinematics are represented by the functions $A \equiv A(a(t)), B \equiv B(b(t))$ associated with the rotations around the primary (the rotary table) and the secondary (tilt table) axes respectively. They are specified by the structure of the machine. For the five-axis machine in Fig.1, the kinematics involving two rotations and three translations are given by:

$$M(t) = B(t)(A(t)(W(t)+R)+T)+C, \qquad (4)$$

where R, T, and C are respectively the coordinates of the origin of the workpiece in the rotary table coordinates, coordinates of the origin of the rotary table coordinates in the tilt table coordinates and the origin of the tilt table coordinates in the cutter center coordinates. The general inverse kinematics are given by:

$$W = (A^{-1}(B^{-1}(M-C)-T)) - R. \qquad (5)$$

In this work, we develop our virtual machine for generating and simulating the tool trajectory using the following sequence of transformations to represent the inverse kinematics of the real five-axis machine.

1. Transform a CL-point P_1 (x, y, z, i, j, k) into the local coordinate system (LCS) of the rotary table by means of the rotation matrix R_1 and the translation vector T_1. The new position is $P_2 = R_1(P_1 + T_1)$.

2. Transform P_2 into the LCS of the tilt table by translation vector T_2 and rotation R_2 and R_3. The new position is $P_3 = R_3 R_2 (P_2 + T_2)$.

3. Transform P_3 until it is coincident with the LCS of the tool tip by translation vector T_3. The final position is $P_4 = P_3 + T_3$.

The translation vectors are given by $T_1 = [t_{1x}, t_{1y}, t_{1z}]$, $T_2 = [t_{2x}, t_{2y}, t_{2z}]$, $T_3 = [t_{3x}, t_{3y}, t_{3z}]$, where t_{ix}, t_{iy}, t_{iz} are the corresponding offsets which depend on the reference position of the machine. The rotation matrices are given by:

$$R_1 = \begin{pmatrix} \cos(A) & \sin(A) & 0 \\ -\sin(A) & \cos(A) & 0 \\ 0 & 0 & 1 \end{pmatrix},$$

$$R_2 = \begin{pmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix},$$

$$R_3 = \begin{pmatrix} \cos(B) & 0 & -\sin(B) \\ 0 & 1 & 0 \\ \sin(B) & 0 & \cos(B) \end{pmatrix}$$

The rotation angles are:

$$A = \tan^{-1}(j/i),\ 0 \le A \le 2\pi \qquad (6)$$

$$B = -\sin^{-1}[k],\ -\pi/2 \le B \le \pi/2 \qquad (7)$$

However, if the A-angle jumps from a small value to 360^0 or vice versa, then the sequence a_p should be adjusted in order to minimize the difference between the successive values [11]. Thus, it eliminates the sharp variations of the A-angle and prevents from unexpected motion and collision.

Furthermore, given the M-coordinates of two consecutive CL-points $C_1=(x_1, y_1, z_1)$ and $C_2=(x_2, y_2, z_2)$ and the two rotation angles representing an orientation of the tool, the virtual machine will perform the required motion based on equation **(5)**. The results of the tool trajectory can be visualized and used to inspect whether or not the virtual machine performs the correct movements. According to our experiment, all surfaces have been machined with satisfactory result in a single cut. Finally, it is worth noting that the results surprised the technician who had been running the machine for many years. He rarely witnessed machining in a single cut by means of other post processing software.

Our simulation software has demonstrated that a simple analysis of inverse kinematics in equation **(5)** reveals that a linear trajectory of the tool tip in the machine coordinates produces a non-linear trajectory in the workpiece coordinates (curve C_1) as shown in Fig.2.

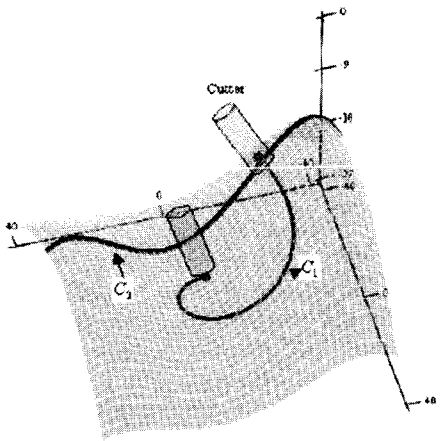
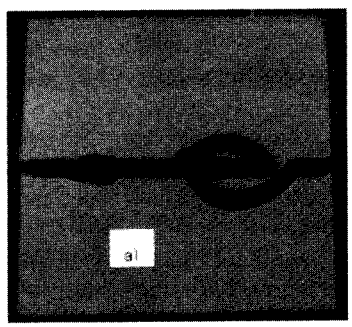


Figure 2. Non linear trajectory of the tool path

It is because of the sharp angular jumps that the machine produces the loop-like trajectory of the tool. Moving along such trajectory could destroy the workpiece and even lead to a collision with the machine parts. Fig.3 demonstrates such a trajectory in the case of machining a single curve belonging to the surface presented in Fig.2 (curve C_2). Fig.3 (a, b) shows that as opposed to a linear version of the tool path, the real machining produces a loop-like trajectory induced by the large angular steps. We have demonstrated that such trajectory could be repaired (see Fig.3 (c, d)) by adjusting the rotation angles in such a way that the kinematics error is minimized [11].



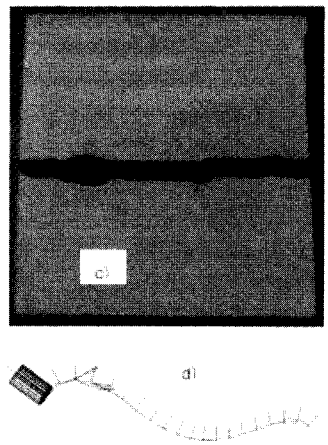


Figure 3. A loop-like trajectory of the tool path, (a) real machining, and (b) virtual machining, and a repaired trajectory of the tool path, (c) real machining, and (d) virtual machining

Note that, without tool trajectory generation and simulation, such errors would not be detected. Our virtual machine has demonstrated that it is able to verify and modify the G-Code before the real machining. We have proved that the G-Code generated by our virtual machine performs the real machining exactly in the same way as the virtual machine does.

3. Computation of CC-Points and Tool Inclination

The CAM software usually generates a successive set of CL-points, which are converted into the G-code to represent the machine motion in which the tool tip coincides with the CL-point. In the case of a non-planar surface cutting with CL-points, this usually generates undercuts and overcuts. Therefore optimization of the tool angle position at each cutter contact point (CC-point) is required to minimize the undercuts and overcuts. The tool geometry is a cylinder (flat end mill as shown in Fig.4). Since we employ the parametric model of the surface $\{x(u,v),\ y(u,v),\ z(u,v)\}$, the parametric equations produce the point coordinates and the normal vector. The CL-point is positioned at the centerline of the tool along the surface normal. Such a representation must be changed to a CC- point positioned in the tool tip plane and on the circle produced by intersection of the tool cylinder and the tool tip plane (see Fig.4).

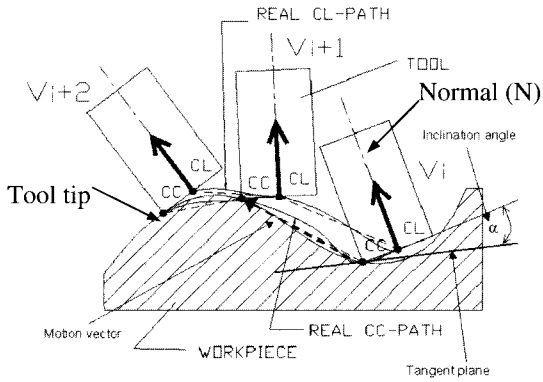


Figure 4. Tool and workpiece contact zone

Given the input CL-point (x,y,z,i,j,k), we outline the algorithm for constructing a CC-point as the following:

1. Construct the tangent plane perpendicular to the tool vector (normal vector):

$$\tau(x,y) = -\frac{i(x-x_1)}{k} - \frac{j(y-y_1)}{k} + z_1, \quad \textbf{(8)}$$

where $CC = (x_1, y_1, z_1)$ represents the CC-point and (i,j,k) is the tool orientation.

2. Compute two coplanar vectors v_1, v_2 at the tangent plane given by:

$$v_1 = (1,0,-i/k), v_2 = (0,1,-j/k). \quad \textbf{(9)}$$

3. Compute a projection of the motion vector M onto τ.

$$P = (M \bullet v_1)v_1 + (M \bullet v_2)v_2, \quad \textbf{(10)}$$

where $M = (x_2 - x_1, y_2 - y_1, z_2 - z_1)$ and (x_2, y_2, z_2) is the next successive point of the tool path.

4. Translate the CC-point by r in the direction opposite to the positive direction of the projection vector to get the new CL-point, where r denotes the tool radius.

$$CL = CC - rP. \quad \textbf{(11)}$$

Note that, the new CL-point represents the CC-path in which the tool tip is in contact with the workpiece. Fig.5 illustrates the trajectory (in

gray color) of the CC-path of the saddle surface generated from equation **(11)**, **(3)** and **(5)** respectively, whereas the original CL-path displays in dark color.

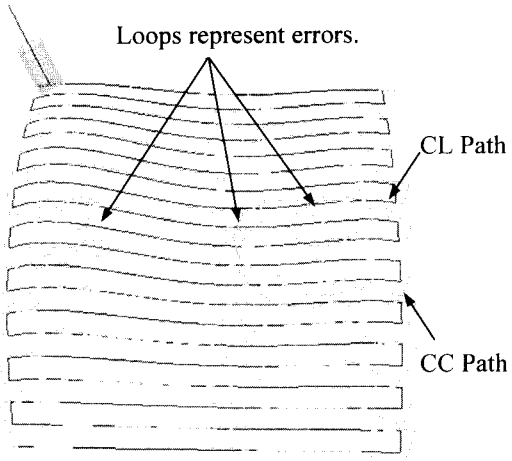


Figure 5. Trajectory of CC and CL-points

In the case of a concave surface, the translation of the tool along the tangent plane may not eliminate the undercut. We tackle the problem by changing the tool inclination angle with regard to the tangent plane (see Fig.4).

It should be noted that advanced algorithms [13,14,16] are required to compute the curvature interference between the tool and the surface to determine the optimal tool inclination angle (α). However at this stage we use a heuristic α-degree inclination and compare the errors with the case without inclination. An algorithm for constructing the CC-point with tool inclination is presented below.

1. Compute vector O orthogonal to both projection vector P and normal vector N.

$$O = N \times P \qquad \textbf{(12)}$$

2. Rotate the tool about O by α degree using transformation matrix T:

$$T = \begin{pmatrix} P_x & O_x & i \\ P_y & O_x & j \\ P_z & O_y & k \end{pmatrix} \begin{pmatrix} \cos(\alpha) & 0 & \sin(\alpha) \\ 0 & 1 & 0 \\ -\sin(\alpha) & 0 & \cos(\alpha) \end{pmatrix}$$

3. Translate the CC-point by the distance of the tool radius r in the opposite direction of the projection vector to get the new CL-point.

$$CL = \begin{pmatrix} P_x & O_x & i \\ P_y & O_x & j \\ P_z & O_y & k \end{pmatrix} \begin{pmatrix} \cos(\alpha) & 0 & \sin(\alpha) \\ 0 & 1 & 0 \\ -\sin(\alpha) & 0 & \cos(\alpha) \end{pmatrix} \begin{pmatrix} -r \\ 0 \\ 0 \end{pmatrix} + CC \quad \textbf{(13)}$$

Fig. 6 illustrates the repaired trajectory of the CC-points with 5-degree tool inclination. Note that, generation of tool path using CC-points reduces the size of loops at the surface stationary points. This implies that the errors are also reduced (see Table 1).

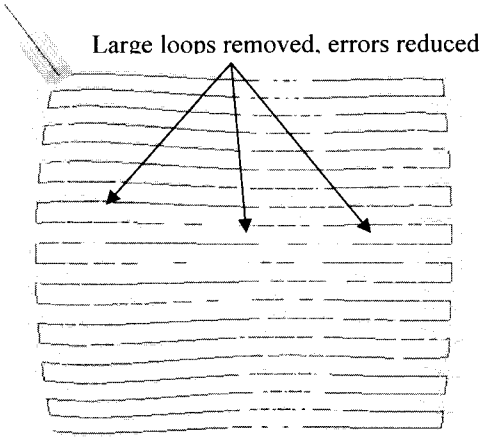


Figure 6. Repaired trajectory of CC-points with 10-degree tool inclination

Table 1 and 2 show the error before and after optimization performed for the saddle surface when the tool is inclined by 5^0 and 15^0 respectively. Although the error decrease is no longer monotone it is of the same order. It is clear that with the kinematics error that large, errors due to curvature interference, local gouging as well as the thermal errors are negligible.

4. Virtual and Practical Machining

The objective of the pilot cutting experiments is the calibration of the parameters involved in the inverse kinematics. The inverse kinematics transforms the tool reference vector *(X, Y, Z, I, J, K)* fixed to the workpiece *(W)* into the machine coordinates *(X, Y, Z, A, B)* fixed to the machine frame *(M)*. Since the inverse transformation contains specific machine

dependent geometric parameters, cutting a convex bell shape surface (Section 4.1) allows for simultaneous calibration of the rotation angles *A* and *B*. If the angle *A* or *B* changes unexpectedly from minimum to maximum or vice versa, this may cause an expected collision among each machine's axis. Therefore, the angle adjustment algorithm is implemented to produce the continuous variation of the angle [11]. The experiments result in excellent surface quality without an undercut. Next (Section 4.2), we analyze effects of undercut/overcut in the case of a convex/concave surface using a two-bell surface. We observe the problem that the tool interferes with the workpiece and removes an excessive amount of the material. Finally (Section 4.3), we demonstrate the trajectory simulation of the custom tool path where the virtual machine will read the input CL-points from any CAD systems and convert into the G-Code and simulate the tool trajectory. Once, the result is satisfactory, then the actual machining can be performed.

4.1 Calibration of simultaneous rotation of A and B axis and correction of the angles.

This experiment introduces a convex parametric bell surface.

$$P(u,v)=\begin{bmatrix} 20u-10 \\ 20v-10 \\ -15\left[(u-0.5)^2+(v-0.5)^2\right] \end{bmatrix} \quad \textbf{(14)}$$

The normal to the surface is computed using the derivative with regard to u and v.

We then construct a zigzag tool path of

$$N(u,v)=\frac{dP/du \times dP/dv}{\left|dP/du \times dP/dv\right|} \quad (15)$$

40x40 points. The original CL-points *(X, Y, Z, I, J, K)* are calculated from:

$$\begin{aligned} X_{i,j} &= P(u_i,v_j)_0 & I_{i,j} &= N(u_i,v_j)_0 \\ Y_{i,j} &= P(u_i,v_j)_1 & J_{i,j} &= N(u_i,v_j)_1 \\ Z_{i,j} &= P(u_i,v_j)_2 & K_{i,j} &= N(u_i,v_j)_2 \end{aligned} \quad \textbf{(16)}$$

The angle-*A* and angle-*B* are computed using equation **(6)** and **(7)** respectively. *A* and *B* axis are rotating simultaneously ranging from 0 to 360^0 and from -90^0 to 90^0 respectively.

Fig.7a shows the required surface, Fig.7b shows the corresponding graphs of the rotation angles-*A*. The *A*-angle requires an adjustment shown in Fig.7c. Note that the angle adjustment is required to eliminate sharp variations of the rotation angles near 0 or 360^0. For instance if *A* changes from 5^0 to 355^0 the algorithm replaces 355^0 by -5^0 etc. Note that although the general case requires an adjustment of *B* as well, the case of a bell shaped surface implies that $B \in [-90,90]$. Therefore the adjustment is not required. Fig.8 displays the corresponding tool trajectory produced by the virtual machine. We observe that only a little error is detected. Fig.9 shows the finished part of the bell surface using 60x60mm-workpiece and 10mm-tool size. The result is an excellent surface quality without overcut and undercut.

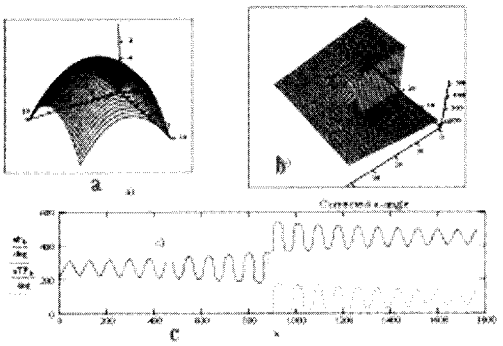


Figure 7. The parametric bell surface

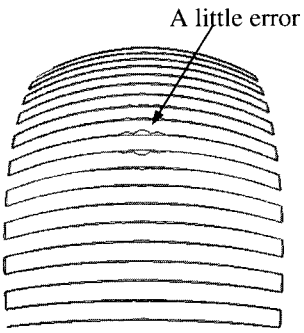
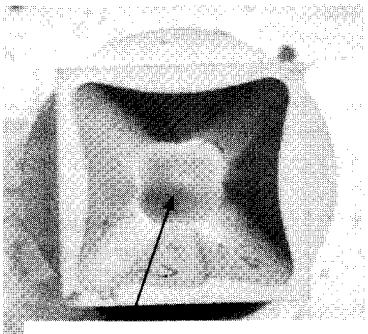


Figure 8. Tool trajectory simulation



Surface with a little error.

Figure 9. The finished bell surface

4.2 Concave and Convex regions and undercut.

This experiment employs a parametric two-bell surface for an analysis of an undercut in the case of a convex-concave surface.

$$P(u,v)=\begin{bmatrix} 20u-10 \\ 20v-10 \\ -20\left[(u-0.3)^2+(v-0.3)^2\right] \\ e \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ -20\left[(u-0.7)^2+(v-0.7)^2\right] \\ e \end{bmatrix} \quad (17)$$

The zigzag tool path and the original CL-points *(X, Y, Z, I, J, K)* and the angle-*A* and angle-*B* are calculated in a similar manner as in section 4.1.

The experiment displays saddle type regions that lead to substantial milling errors. The first type of the error is due to the large size of the tool. The second type is due to the removal of the material by a side of the tool (Fig.10, the tool removes the materials from the first bell as well as cutting the second bell).

Note that the two-bell case requires a substantial adjustment of the angle. Fig.11 displays the tool trajectory for the parametric two-bell surface with large milling errors detected by the virtual machine. The finished real surface is shown in Fig.12 using 60x60mm-workpiece and 10mm-tool size. The result is an average surface quality, with a substantial undercut. This experiment produces a real result that is quite different from the simulator. The reason is due to the fact that we have not yet implemented the workpiece removal model, which takes into account the tool size, the gouging, the curvature and surface interference [12,13].

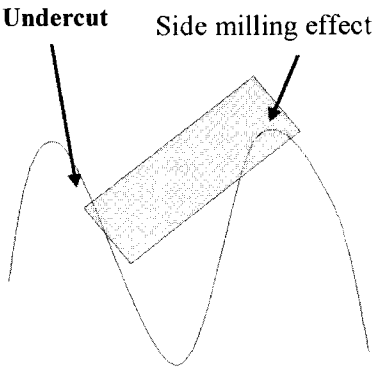


Figure 10. Side milling effect

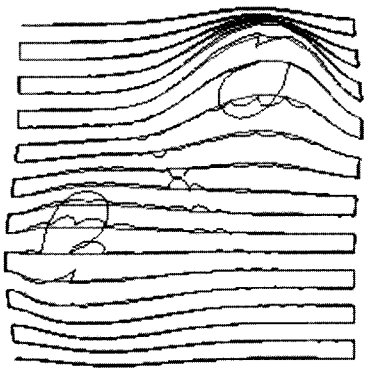
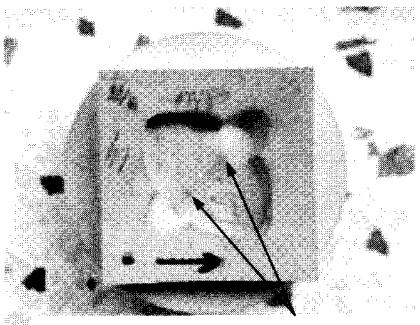


Figure 11. Tool trajectory simulation



Side milling effect removes the bell apex.

Figure 12. The finished the two-bell surface

4.3 Simulation of the custom tool path

This experiment is performed to debug the virtual machine as applied to a curvilinear tool path produced by the grid generation procedure. The input CL-file (*X, Y, Z, I, J, K)* is read from the text file generated by any CAD/CAM systems. In order to suppress an impact of overcut or undercut we perform a three-axis milling on the five-axis milling machine. The rotation angles are fixed, namely *A=0, B= -90.* Note that although we obtain a better surface quality in the zone of sharp variations without under and overcut, the total quality of the surface is average (due to three-axis machining).

However the experiment clearly shows that the virtual machine is applicable to any curvilinear zigzag pattern and able to process the CL-points in the form of *X, Y, Z, I, J, K* from any CAD systems.

Fig.13 displays the corresponding tool trajectory in workpiece coordinate. Fig.14 displays a graphical representation of the G-Code in machine coordinate. The graphical G-Code and the tool trajectory are the same in the case of three-axis milling. The result is automatically verified by the virtual machine. This ensures that the virtual machine is also capable of all three-axis milling. The finished real surface is shown in Fig.15 using 60x60mm-workpiece and 10mm-tool size. The real surface is exactly the same as the virtual surface produced by the virtual machine.

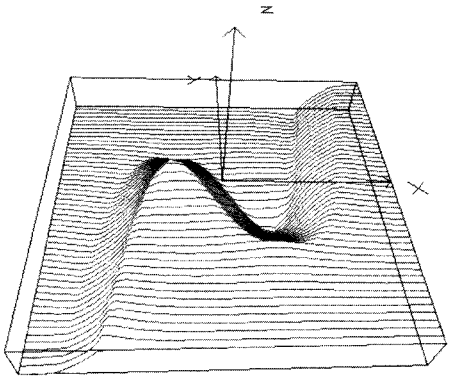


Figure 13. Tool trajectory simulation

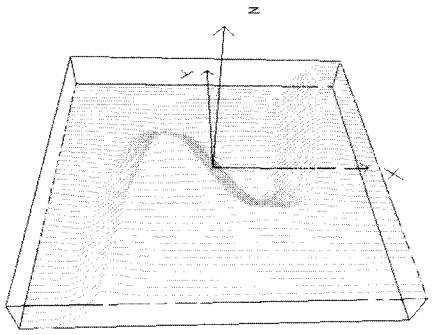
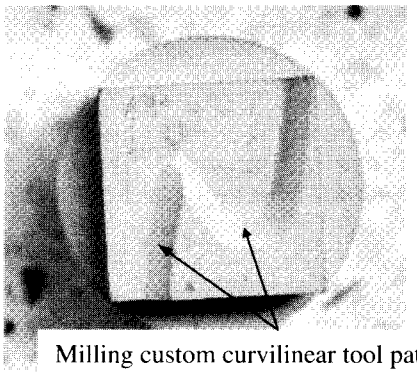


Figure 14. The graphical G-Code



Milling custom curvilinear tool path.

Figure 15. The finished custom surface

5. Conclusions and Future Work

We have developed 3D interactive G-Code (post processing) and tool path simulation software capable of generating and simulating the tool trajectory for the five-axis milling machine. The software provides a 3D graphical environment to manipulate (translate, rotate, zoom) the CL-points, the G-Code as well as the tool trajectory. It evaluates a correct position of the tool in the five-dimensional space, i.e. the location, the orientation and the heuristic inclination of the tool for an arbitrary sculptured surface and graphically simulates such trajectory to examine the preliminary results. The software has been verified by surfaces having the saddle type regions, multiple extreme, steep regions with a high curvature, etc. Currently, the virtual machine provides extraordinary results that can be used to estimate the errors and verify the tool path graphically. Future work includes the development of more accurate numerical error computation methods to simulate and estimate the errors so that the tool trajectory can be further modified to minimize the errors. The inverse kinematics combined with the solid

model of the machine will provide a realistic and powerful simulator for simulation, verification and optimization of cutting operations. The workpiece simulation model [15] will be combined with advance curvature and surface interference detection [13] as well as machining strip width and dynamic tool inclination [16] to simulate the realistic workpiece removal process. This will solve the problem of side milling effect and undercut/overcut. Finally, the software will be linked with the grid generation technique [17] to optimize the tool path in the regions of sharp variations of the curvature of the surface.

6. Acknowledgement

The author wishes to thank the Thailand Research Fund (TRF) and National Electronics and Computer Technology Center, National Science and Technology Development Agency of Thailand for their financial support. I also would like to thank Dr. S.S. Makhanov for his helpful ideas and suggestions.

7. References

[1] Y. Huang and J. H. Oliver, *Integrated Simulation, Error Assessment, and Tool Path Correction for 5-Axis NC Milling*, Journal of Manufacturing, Vol. 14, No. 5, pp. 331-344, 1995.

[2] E. A. Gani, B. Lauwer, P. Klewais and J. Detand, *An Investigation of the Surface Characteristics in 5-Axis Milling with Toroid Cutters,* Proc. Pacific Conference on Manufacturing, pp. 53-60, 1994.

[3] R. Sarma, *An Assessment of Geometric Methods inTrajectory Synthesis for Shape Creating Manufacturing Operations,* Journal of Manufacturing Systems, No. 19, pp. 59-72, 2000.

[4] K. Kim, J. Jeong, *Finding Feasible Tool-approach Directions for Sculptured Surface Manufacture*, IIE Transactions, No. 28(1), pp. 829-836, 1996.

[5] Y. Koren, *Five-axis Iinterpolators*, Annals of CIRP, No. 44(1), pp. 379-382, 1995.

[6] J.-P. Kruth, P. Klewais, *Optimization and Dynamic Adaptation of the Cutter Inclination During Five-axis Milling of Sculptured Surfaces,* Annals of CIRP, No. 43(1), pp. 443-448, 1994.

[7] S.-H. Suh and Y.-S. Shin, *Neural Network Modeling for Tool Path Planning of the Rough Cut in Complex Pocket Milling,* Journal of Manufacturing Systems, Vol. 15, No. 5, pp. 295-324, 1996.

[8] J. Jeong and K. Kim, *Generation of Tool Paths for Machining Free-Form Pockets with Islands Using Distance Maps*, Advanced Manufacturing Technology, No. 15, pp. 311-316, 1999.

[9] E. L. J. Bohez, *Compensating for Systematic Errors in Five-axis NC Machining,* Computer-Aided Design, No. 34, pp. 391-403, 2002.

[10] S. S. Makhanov, *An Application of the Grid Generation Techniques to Optimize a Tool-path of Industrial Milling Robots,* Journal of Computational Mathematics and Mathematical Physics, No. 39, pp. 1524-1535, 1999.

[11] M. Munlin, *Virtual Five-Axis Milling Machine: Error Estimator,* Thammasat International Journal of Science and Technology, Vol. 7, No. 1, pp. 40-49, 2002.

[12] A. Rao, R. Sarma, *On Local Gouging in Five-axis Sculptured Surface Machining using Flat end Tools,* CAD No. 32, pp. 409-420, 2000.

[13] D. Yu, J. Deng, Z. Duan and J. Liu, *Generation of Gouge-free Cutter Location Path on Freeform Surfaces for Non-spherical Cutters*, Computer in Industry Vol. 28, pp. 81-94, 1996.

[14] N. Rao, F. Ismail and S. Bedi, *Tool Path Planning for Five-axis Machining using the Principal axis Method,* International Journal of Machine Tools Manufacturing, Vol. 37, No. 7, pp. 1025-1040, 1997.

[15] U. Roy and Y. Xu, *Computation of a Geometric Model of a Machined Part From its NC Machining Programs,* CAD, No. 31, pp. 401-411, 1999.

[16] C.-C Lo, *Efficient Cutter-path Planning for Five-axis Surface Machining with a Flat-end Cutter,* CAD, No. 31, pp. 557-566, 1999.

[17] S. S. Makhanov, E.J. Vanderperre and K. Sonthipaumpoon, *On Generation of Curvilinear Grids*, South-East Asian Bull.of Math, No. 23, pp. 231-241, 1999.

Table 1. Kinematics errors for the optimized and non-optimized tool path with inclination angle 5^0

Number of the CL-points	No optimization error (mm) avg/max	Optimization error, (mm) avg/max	Error decrease (%)	Path length Non-opt/opt (mm)
20 x 20	0.236/13.986	0.180/5.199	62.83	2485.82/2224.49
30 x 20	0.128/7.871	0.098/4.299	45.38	2368.56/2160.61
40 x 20	0.084/7.981	0.065/3.652	54.24	2288.54/2117.87
50 x 20	0.060/5.655	0.053/3.629	35.83	2192.93/2113.41
60 x 20	0.049/6.117	0.043/2.678	56.22	2165.54/2084.19
70 x 20	0.041/5.235	0.037/2.276	56.52	2135.49/2070.56
100 x 20	0.031/3.062	0.031/3.062	0.00	2070.54/2070.54
130 x 20	0.028/1.886	0.028/1.886	0.00	2034.90/2034.90

Table 2. Kinematics errors for the optimized and non-optimized tool path with inclination angle 15^0

Number of the CL-points	No optimization error (mm) avg/max	Optimization error (mm) avg/max	Error decrease (%)	Path length non-opt/opt (mm)
20 x 20	0.249/11.174	0.194/4.569	59.11	2555.35/2286.40
30 x 20	0.129/6.293	0.121/4.374	30.49	2326.21/2265.82
40 x 20	0.091/8.515	0.084/4.471	47.49	2293.42/2218.14
50 x 20	0.071/7.906	0.065/3.430	56.62	2256.84/2179.24
60 x 20	0.057/5.402	0.056/2.840	47.43	2199.05/2174.77
70 x 20	0.050/6.537	0.048/3.383	48.29	2182.04/2149.31
100 x 20	0.039/5.051	0.038/3.247	35.72	2136.00/2123.82
130 x 20	0.034/3.941	0.034/3.941	0.00	2106.55/2106.55