ANGLE CORRECTION ALGORITHMS FOR FIVE-AXIS MILLING MACHINE

M. Munlin 1 and S.S. Makhanov 2
1 Tapee College, 8/151 Srivichai Road, Muang District, Suratthani 84000, Thailand
2 School of Information and Computer Technology, Sirindhorn International Institute of Technology, Thammasat University, 131 Moo 5, Tiwanont Road, Bangkadi Pathum Thani 12000, Thailand
Email: mmunlin@gmail.com

ABSTRACT

When the tool of a five-axis milling machine travels across or around a stationary point the rotation angles jump considerably leading to unexpected deviations from the prescribed trajectories. Consequently, the inverse kinematics of five-axis milling machines produces large errors near the stationary points. We propose two new algorithms to repair the trajectories by adjusting the rotation angles in such a way that the kinematics error is minimized. The first algorithm is based on global optimization procedure to minimize the kinematics error with regard to feasible solutions of the inverse kinematics. We show that such optimization increases the accuracy of the milling operations and is the most appropriate in the case of a rough-cut. The optimization algorithm is based on the iterative shortest path scheme. The shortest path procedure can be applied to either the entire set of trajectories or to only the most inappropriate trajectories inside the work piece. The second algorithm is based on the equi distribution with regard to the rotation angles. The kinematics error near the stationary points is much more sensitive to the variation of the rotation angles than to the spatial variations. Therefore, the method inserts additional tool positions by finding numerically a grid of points equi distributed in the angular increments. We prove that near the stationary points the proposed method may require 10-15 times less additional points than the conventional scheme.

The efficiency of the two algorithms has been verified by a five-axis machine MAHO600E at the CIM Lab of Asian Institute of Technology of Thailand and HERMLE UWF920H at the CIM Lab of Kasetsart University of Thailand.

Keywords: Five-axis machines, inverse kinematics, tool path optimization, kinematics error, angle optimization, the shortest path.

1. INTRODUCTION

The five-axis machine is guided by axial commands $\Pi \in \mathbb{R}^5$ carrying three spatial coordinates of the tool-tip in the machine coordinate system and two rotation angles. The tool path $A = \{\Pi_0, \Pi_1, \ldots, \Pi_m\}$ is a sequence of positions $\Pi_i$, arranged into a structured spatial pattern such as the zigzag or the spiral pattern. The path could also be composed of a variety of unconventional patterns and include tool retractions. A full optimization scheme involves a model of cutting operations, topologies of the prescribed tool path patterns and an optimization procedure. Let $\mathbf{M}$ be a set of parameters related to the configuration of the machine and $\mathbf{T}$ parameters related to the tool. Let $S$ be the required surface. The model of the cutting operations, being fed with $\mathbf{M}$, $\mathbf{T}$, $S$ and $A$, produces a result of machining, the output surface $T$. The general optimization problem is then formulated by

$$\begin{align*}
\text{minimize}(c),
\Pi, \mathbf{M}, \mathbf{T}
\end{align*}$$

where $c$ denotes a criteria vector which may include the above mentioned parameters such as the error $c = ||S - T||$, the length of the path, the negative of the machining strip (strip maximization), the machining time, etc. The optimization could be subjected to constraints. The most important is 1) The scallop height constraint. The scallops between the successive tool tracks must not exceed a prescribed tolerance. 2) The local accessibility constraints. 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. 3) The global accessibility constraints. The
constraint ensures against the tool coming in contact with either machine parts (collision detection) or unwanted parts of the desired surface. It should be noted that minimization of a particular component of the criteria vector may be replaced by the constraint and vice versa. For instance the scallop height could be either minimized or constrained.

It may seem that the ultimate goal of five-axis optimization is simple: minimize the difference between the desired and the actual surface for a minimum time. However, mathematical formulations presented in the literature vary in terms of the error criteria and the set of optimized variables. The tool path is optimized with regard to the machining time, accuracy, the length of the tool path, the width of the machining strip, the volume of the removed material, the size of the remaining scallops, etc.[1,2,3]. Furthermore, the error analysis and optimization in the areas of large variations of the rotation angles have not been provided by commercial CAD/CAM systems such as Unigraphics, EdgeCam, Vericut, etc. Besides, only a few research papers deal with the subject. In [4] the authors analyze the sequence of rotations to minimize the number of the phase reverse steps at discontinuities of the first derivative of the surface (corners etc). However the case of the stationary points has not been analyzed. A method of avoiding singularities has been presented in [5]. However, it is not hard to give a counter-example of a surface on which such avoidance can not be performed. For instance, it is a continuous set of the stationary points on a “hill” on a flat surface [6].

Therefore, this paper presents two new optimization algorithms performed in the angular domain near stationary points of the required surface. The first algorithm optimizes the sequence of rotations of the five-axis milling machine which contribute to the inaccuracy of a machined workpiece at the vicinity of stationary points of the desired surface. It is based on global optimization with regard to feasible solutions of the inverse kinematics. We show that such optimization increases the accuracy of the milling operations and is the most appropriate in the case of a rough cut. The optimization algorithm is based on the iterative shortest path scheme. The shortest path procedure can be applied to either the entire set of trajectories or to only the most inappropriate trajectories (the so-called loops) inside the workpiece. The second algorithm is based on the equi distribution with regard to the rotation angles. The kinematics error near the stationary points depends on the variation of the rotation angles. The method allows inserting additional tool positions by finding numerically a grid of points equi distributed in the angular space. We prove experimentally that the proposed method requires 10-15 times less additional points than conventional schemes performed near the stationary points. The efficiency of the two algorithms has been verified by a five-axis machine MAHO600E at the CIM Lab of Asian Institute of Technology of Thailand and HERMLE UWF920H at the CIM Lab of Kasetsart University of Thailand. It has been also verified by solid modeling in Unigraphics V18 [7] as well as through the virtual milling machine simulator [8] developed by the authors.

2. Surface Singularity Position

Five-axis milling machine (Fig.1) is guided by the axial commands carrying the 3 spatial coordinates of the tool tip in the machine coordinate system M and the two rotation angles. The CAM software generates a successive set of coordinates (called cutter contact points or CC-points) in the workpiece coordinate system W. Typically, the CAM software distributes the CC-points along a set of curves, which constitutes the so-called zigzag or spiral pattern. An appropriate transformation into the M-system generates a set of the machine axial commands which provides the reference inputs for the servo-controllers of the milling machine.

Fig.1 Five-axis milling machine MAHO600E
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 of the tool path and $\mathcal{R}_p$ and $\mathcal{R}_{p+1}$ the corresponding rotation angles. In order to calculate the tool trajectory between $W_p$ and $W_{p+1}$ we, first, invoke the inverse kinematics to transform the part-surface coordinates into the machine coordinates $M_p = (X_p, Y_p, Z_p)$ and $M_{p+1} = (X_{p+1}, Y_{p+1}, Z_{p+1})$. Second, the rotation angles $\mathcal{R} = \mathcal{R}(t) = (a(t), b(t))$ and the machine coordinates $M = M(t) = (X(t), Y(t), Z(t))$ are assumed to change linearly between the prescribed points, namely,

\[
M(t) = tM_{p+1} + (1 - t)M_p,
\]

\[
\mathcal{R}(t) = t\mathcal{R}_{p+1} + (1 - t)\mathcal{R}_p,
\]

where $t$ is the fictitious time coordinate ($0 \leq t \leq 1$). Finally, invoking the transformation from $M$ back to $W$ (for every $t$) yields $W(t) = (x(t), y(t), z(t))$.

The kinematics are represented by matrix-functions $A = A(a(t))$, $B = B(b(t))$ associated with the rotations around the primary (the rotary table) and the secondary (the tilt table) axes respectively. Although the kinematics are specified by the structure of the machine, the resulting transformation is nothing else than successive rotations and translations designed to transport the tool to the desired point of the workpiece and with the specified orientation. For instance, the machine configuration depicted in Fig. 1 implies $M(t) = B(t)A(t)(W(t) + R) + T + C$, 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.

A simple analysis of the inverse kinematics equations reveals that a linear trajectory of the tool tip in the machine coordinates may produce a non-linear or circular arc trajectory in the workpiece coordinates (see Fig.2 generated by our simulation software). We shall call the deviation from the non-linear trajectory the kinematics error.

![Fig.2 Non-linearity of the tool-path in the workpiece coordinates $C_1$, the experimental cut $C_2$.](image)

Note that a fine cut of a smooth surface employing small spatial and angular steps may not demonstrate the detrimental effects near the singularity points. However, a rough cut characterized by large gradients could produce considerable errors.

It is because of the sharp angular jumps that the machine produces the loop-like trajectories of the tool. Moving along such trajectories may destroy the work piece and even lead to a collision with the machine parts. Furthermore, suppose that the tool vector changes the sign from positive to negative or vice versa, the inverse kinematics would produce the singularity position which any value for the rotary table is admissible [5,9]. Such singularity point on the surface presents a special case when $i$ and $j$ components of the tool vector are equal to zero. It is plain that in this case the rotation angles may jump considerably leading to unexpected deviations from the prescribed trajectory. The tool path deformation in [5] has made an attempt to by pass such singular point. However, such computation is complex, expensive and does not preserve the original CC points. In addition, they do not generate their own tool trajectory, instead using the build-in postprocessor from the NC machine. Therefore, the deformed tool path can not be verified either by the simulation or practical machining. In section 4, we will propose a simpler angle optimization method to cross such singularity either by finding the
optimum rotation path or using equally small rotation angles in which the original CC points are preserved. The proposed methods have been verified by both the simulation and practical machining.

As mentioned before, the tool trajectory is actually a circular arc that can be anywhere around the surface. This trajectory may be inside the actual surface that could not be detected by the typical gouging elimination [3,10,11]. In addition to angle optimizations, the good and the bad trajectory detection and correction is also needed. We address this issue in the next section.

3. Bad Trajectory Detection and Correction

The CAD software generates the CC points in the work piece coordinate system whereas the CAM software generates the NC program in the machine coordinate system from the prescribed CC points. The NC program guides the cutting tool of the five-axis machine to travel along a nonlinear trajectory to reach the required CC point. The nonlinear trajectories constitute a trajectory surface, which is slightly different from the actual surface. The error surface is estimated by computing the difference between the actual and trajectory surfaces.

Let four points $W_1(i,j), W_2(i+1,j), W_3(i+1,j+1)$ and $W_4(i,j+1)$ in Fig.3 be the grid $\{(u,v)_{i,j}\}$ which represents the actual surface $S(u,v)$. First, we apply the inverse kinematics (see section 2) to transform each point into the corresponding machine coordinate $M_1, M_2, M_3,$ and $M_4$ respectively. Then, we perform the linear interpolation procedure on the machine coordinate $M$ and invoke the inverse transformation from $M$ back to $W$ (for every $t$) yields the tool path $W(t) = (x(t), y(t), z(t))$. Next, the calculated tool path is mapped on the grid $\{(u,v)_{i,j}\}$ yields an approximation of the machined surface $T(u,v)$. The error surface is then calculated by $w_t(u,v) = |S(u,v) - T(u,v)|$, where $S(u,v) = (x(u,v), y(u,v), z(u,v))$ is the actual surface and $T(u,v) = (x_T(u,v), y_T(u,v), z_T(u,v))$, is the trajectory surface as shown in Fig.3. These errors are estimated and visualized graphically by our virtual machine [8].

![Fig.3 Error surface computation](image)

The trajectory generated by the inverse kinematics can be anywhere near the surface. If the trajectory is inside the surface, it is a bad trajectory or bad loop (denote by thick circular loop in Fig.4), which need to be repaired, otherwise it is a good trajectory or good loop (denote by normal circular arc in Fig.4). Since, the parametric surface is typically constructed in the $\{(u,v)_{i,j}\}$ domain and the tool trajectory is also in the same domain. Therefore, we can identify the good and bad trajectory by comparing the $z$-value of the actual surface and the trajectory. Although, the tool trajectory is interpolated from the two adjacent CC points, but its real trajectory may sweep through anywhere around the surface especially near the large milling errors. Therefore, we need to compute the $z$-value anywhere on the surface that corresponds to each interpolated trajectory point. This can be done by first, substitute the $x$ and $y$ position of the current trajectory into the surface equation to obtain the $u$ and $v$ parameters. Next, we substitute the $u$ and $v$ parameters into the surface equation to obtain the $z$-value on the surface. Finally, we compare this $z$-value with the $z$-value of the trajectory to identify the bad trajectory as display graphically in Fig.4. Obviously, such bad trajectory would damage the workpiece and is infeasible for machining. Typical CC points insertion or setting up the new workpiece configuration will not alter such trajectory. We propose two angle correction algorithms to alter such trajectory so that the sharp rotations are decrease and turn from the bad to the good trajectory. There is also a case in which the larger good loop may turn into a smaller bad loop after applying angle correction algorithm as shown in Fig.5. In this case, the bad loop detection must be combined with the angle correction to eliminate all the bad trajectories of the entire tool path.
4. Angle Correction Algorithms

Two angle correction algorithms namely, “angle switching” and “angle insertion” are proposed to repair the bad trajectories or the large milling error by adjusting the rotation angles in such a way that the kinematics error is minimized. The first algorithm is based on global optimization procedure to minimize the kinematics error with regard to feasible solutions of the inverse kinematics combined with good and bad trajectory detection. The second algorithm is based on the equi distribution with regard to the rotation angles near singularity position. The kinematics error near the singularity points is much more sensitive to the variation of the rotation angles than to the spatial variations. A special interpolation combined with the bisection methods is employed to insert additional points equally in angular space to obtain a uniform minimal kinematics error. Note that, prior to the proposed algorithms we perform the following angle adjustment [6].

\[
a_{i+1,\text{new}} = \begin{cases} 
    a_{i+1} - 2\pi, & \text{if } a_{i+1} - a_i > \pi, \\
    a_{i+1} + 2\pi, & \text{if } a_{i+1} - a_i < -\pi, \\
    a_{i+1}, & \text{otherwise.}
\end{cases}
\]
The angle adjustment eliminates the rotation angles exceeding \( \pi \), leaving the remaining \( |a_{i+1} - a_i| \leq \pi \). This order of angular distant still introduces considerable kinematics errors and therefore requires the further optimizations. Note that Modern five-axis machine such as UWF902H incorporates this algorithm into the controller. However, angle adjustment is still useful for the other machine such as MAHO600E.

### 4.1 Angle-Switching

According to our initial setup of the five-axis machine in Fig.1, the \( a \)-angle and \( b \)-angle are given by 
\[
a = \tan^{-1}\left(\frac{j}{i}\right), \quad 0 \leq a \leq 2\pi \quad \text{and} \quad b = -\sin^{-1}\left(\frac{k}{i}\right), \quad -\pi/2 \leq b \leq \pi/2
\]  
However, the normal vector \( i \) and \( j \) can be in any of the four quadrants (see Fig.6). Therefore, we can define
\[
a_{\text{base}} = \begin{cases} 
\tan^{-1}\left(\frac{j}{i}\right), & i > 0 \text{ and } j \geq 0, \\
\pi + \tan^{-1}\left(\frac{j}{i}\right), & i < 0, \\
2\pi + \tan^{-1}\left(\frac{j}{i}\right), & \text{otherwise.}
\end{cases}
\]
\[
b_{\text{base}} = -\sin^{-1}(k)
\]

![Fig.6 Computation of \( a_{\text{base}} \) in each quadrant](image)

Furthermore, there are four sets of \( a \)-angle and \( b \)-angles within the range \([0,2\pi]\) that can rotate the tool vector into the required orientation [6]. The set of the \( a \)-angles is defined by \( \{a_{\text{base}}, a_{\text{base}} - 2\pi, a_{\text{base}} - \pi, a_{\text{base}} + \pi\} \).

As mentioned earlier, the possible largest rotation is still in between \( |a_{i+1} - a_i| \leq \pi \). This variation may produce an unexpected motion that may damage the workpiece and probably the collision. The angle-switching algorithm is employed to select the shortest path from the feasible sequences \( \{p_i, p_{i+1}\} \) in such a way that \( \Sigma |a_{p_{i+1}} - a_p| + \Sigma |b_{p_{i+1}} - b_p| \) is minimized along a good trajectory. For instance, a position \( \{a_p, b_p\} \) could be followed by either \( \{a_{i+1}, b_{i+1}\} \), calculated by means of the formula above, \( \{a_{i+1} + \pi, -b_{i+1} - \pi\} \), \( \{a_{i+1} - \pi, -b_{i+1} + \pi\} \) or \( \{a_{i+1} - 2\pi, b_{i+1}\} \).

We can determine the best path by examining the minimum kinematics errors at each point along the good trajectory. As described in [5,9], the singular position such as minimum or maximum or saddle generates the large circular trajectories (loops). These loops represent the larger errors than any other points due to the longer distance the tool has to travel to cross the singular point. Therefore, we may perform the optimization with regards to this set. We will minimize the kinematics error traveled by the tool in the space \( (a, b) \) with the Euclidean distance. Let \( W_{p,p+1}(t) \), \( L_{p,p+1}(t) \) be the actual and the linear trajectory of the tool tip between CC point \( \Pi \) and \( \Pi_{p+1} \). Consider the following minimization problem
where the discrete functional \( \varepsilon_{\text{kinematic}} = \sum_p \varepsilon_{\text{kinematic}}^{p,p+1} \) represents the total kinematics error and \( \varepsilon_{\text{kinematic}}^{p,p+1} = \| W_{p,p+1} \|_2 - L_{p,p+1} \) is the kinematics error between \( \Pi_p \) and \( \Pi_{p+1} \). Note that, we have four set of feasible \((a, b)\) between \( \Pi_p \) and \( \Pi_{p+1} \) (the position of \( \Pi_p \) and \( \Pi_{p+1} \) does not change) and only one set of \((a, b)\) with the smallest error along the good trajectory is selected. We incorporate the bad trajectory detection algorithm into the Dijkstra’s shortest path algorithm and output only the shortest path along the good trajectory. The following steps describe the angle-switching algorithm.

1. Locate the singular position by checking the large errors and the sign of the tool vector to determine the source \(s\) and the destination \(t\) CC points.
2. Construct the error graph to represent four feasible paths from \(s\) to \(t\) using the adjacency list. The graph nodes represent the vertices and the arcs represent the error between two adjacent nodes of all four paths.
3. Apply the Dijkstra’s shortest path [12] and bad trajectory detection algorithm to compute the smallest error from \(s\) to \(t\), from the error graph in step 2.
4. Update all CC points from \(s\) to \(t\) using the output shortest path from step 3.

According to our experiment, the algorithm decreases the sharp rotation angle into a smaller ones (Fig.7) and will turn the bad trajectory (denote by thick line) into a good one if such smaller rotation is a bad loop (Fig.8). Although the resulting good loop is a larger trajectory but it is not damage the workpiece and would result in a better surface quality than the trajectory with the bad loop.

Fig.7 The original trajectory (left) and the repaired trajectory (right)

Fig.8 The bad loop (left) turns to the good loop (right)
4.2 Angle Insertion (Uniform Angular Grid)

The angle switching algorithms may still leave errors (e.g. when a smaller bad loop turns into a larger good loop), which can be further eliminated only by inserting additional CC points. In particular when the tool path crosses the stationary point in the direction parallel (or nearly so) to the secondary rotary axes. The workpiece often must be rotated by an angle close to $\pi$. Therefore, further optimization is required to reduce such angular jumps. If the trajectory lies far from singularities a conventional approach to insert spatially uniform grid of the CC points between the two given positions solves the problem. However, such interpolation does not remove the jumps near the singularities. In order to solve this problem, a special interpolation called the angle-insertion is proposed to partition the large angular interval into a grid having equal angular increments. Unfortunately, we can not merely split the angular interval in the machine coordinates. The resulting angles do not correspond to the actual CC points and orientations. In other words the trajectory does not follow the surface. The correct set of the CC points and the tool orientations can only be calculated from the surface equation. Therefore, we propose the following angle-insertion procedure.

1. Construct a uniform grid $a_i = a^i + i\Delta a$ (where $\Delta a$ is the angular step) near the singularity

2. For each $a_i$, find the CC point and the orientation that produce $a_i$ by the bisection method as follows:
   2.1 Compute a midpoint $(x_m, y_m)$.
   2.2 Invert the parametric equations $S_x(u, v) = x_m$, $S_y(u, v) = y_m$ to find the corresponding $u_m, v_m$ and consequently $z_m$.
   2.3 Calculate the orientation vector and obtain the corresponding $a_m$.
   2.4 Compare $a_m$ with the target $a_i$.

The bisection procedure above runs until it converges within the prescribed accuracy. Table 1 and Fig.9 illustrate the procedure. The angular interval has been partitioned into the almost equal sub intervals. Moreover, the loops at the singularity have been significantly reduced.

Note that usually only one of the two rotation angles changes sharply near the singularity. Therefore, the bisection should be applied in only one dimension. However, theoretically there exists a possibility when the both angles must be bisected. In this case one should use either a two dimensional version of the above procedure or to bisect a linear combination of the angles. The singular point can be located by finding the largest loop in which the tool vector changes the sign of the $i$ or the $j$ component.

5. Results and Discussion

Table 1 displays the kinematics errors and the rotation angles before and after applying the proposed angle insertion to the test surface 1 (Fig.9) given by

$$S(u, v) = \begin{cases} 
100u - 50 \\
100v - 50 \\
30((u - 0.5)^3 - (v - 0.5)^3) - 6 
\end{cases}$$

The largest loop turns to 8 small loops (Fig.9) in which each loop is 60 times smaller or 98% error reduction. Table 2 compares the performance of each angle correction algorithm on the test surface 2 (Fig.7) given by

$$S(u, v) = \begin{cases} 
100u - 50 \\
100v - 50 \\
-80v(v - 1)(3.55u - 14.8u^2 + 21.15u^3 - 9.9u^4) - 28 
\end{cases}$$

The most impressive results for the test surface 2 are obtained for MAHO600E in the case of a rough cut on 20 by 20 grid. The error has been reduced in more than 20 times after the angle switching and in more than 40 times after the angle insertion. It tells us that the error could be reduced by more than 80 times when then two methods are combined. Moreover the angle insertion works almost 10 times better than a standard CC point insertion. In the case of HERMLE UWF902H the angle switching creates another loop. Although it is 3 times smaller, it is a non-removable loop created by crossing the singular point parallel to the secondary rotary axis. The standard CC point insertion also does not have a significant impact on the error reduction (4 times smaller). It leaves two big loops, one before and another one after the singular point with error 6.104 mm each. However, the angle insertion produces an impressive result by reducing the error in almost 60 times. Finally, since the loops become smaller, the path length of the entire surface is also shorter as illustrated in Table 2.
Fig. 9 The original trajectory (left) and the repaired trajectory (right and bottom) using angle insertion.

Note that, the singularity can be obtained by searching the large loop in which the tool vector changes the sign into opposite direction. Table 1 illustrates the kinematics errors and the rotation angles before and after applying the angle insertion algorithm.

### Table 1. Kinematics error and angular interval for the test surface 1

<table>
<thead>
<tr>
<th>Test Surface1</th>
<th>Grid 20 by 20</th>
<th>NC program</th>
<th>A-Angle (degree)</th>
<th>C-Angle (degree)</th>
<th>Kinematics errors (mm)</th>
<th>Error Decrease (%)</th>
<th>The C-angular interval (degree)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original block 189 and 190</td>
<td>189</td>
<td>1.278687</td>
<td>135.018008</td>
<td>7.352</td>
<td>N/A</td>
<td>89.96398</td>
<td></td>
</tr>
<tr>
<td></td>
<td>190</td>
<td>1.278687</td>
<td>224.981984</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>The angle insertion between the original block 189 and 190</td>
<td>189.1</td>
<td>1.278687</td>
<td>135.018008</td>
<td>0.125</td>
<td>98.299</td>
<td>11.331202</td>
<td></td>
</tr>
<tr>
<td></td>
<td>189.2</td>
<td>1.086171</td>
<td>146.349210</td>
<td>0.128</td>
<td>98.258</td>
<td>11.550172</td>
<td></td>
</tr>
<tr>
<td></td>
<td>189.3</td>
<td>0.975995</td>
<td>157.899382</td>
<td>0.107</td>
<td>98.544</td>
<td>10.623209</td>
<td></td>
</tr>
<tr>
<td></td>
<td>189.4</td>
<td>0.922652</td>
<td>168.522591</td>
<td>0.125</td>
<td>98.299</td>
<td>11.477405</td>
<td></td>
</tr>
<tr>
<td></td>
<td>189.5</td>
<td>0.904027</td>
<td>179.999996</td>
<td>0.125</td>
<td>98.299</td>
<td>11.477406</td>
<td></td>
</tr>
<tr>
<td></td>
<td>189.6</td>
<td>0.922652</td>
<td>191.477402</td>
<td>0.107</td>
<td>98.544</td>
<td>10.623208</td>
<td></td>
</tr>
<tr>
<td></td>
<td>189.7</td>
<td>0.975995</td>
<td>202.100610</td>
<td>0.128</td>
<td>98.258</td>
<td>11.550159</td>
<td></td>
</tr>
<tr>
<td></td>
<td>189.8</td>
<td>1.086171</td>
<td>213.650769</td>
<td>0.125</td>
<td>98.299</td>
<td>11.331215</td>
<td></td>
</tr>
<tr>
<td></td>
<td>190</td>
<td>1.278687</td>
<td>224.981984</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
</tr>
</tbody>
</table>

### Table 2. Performance of the proposed methods for the test surface 2

<table>
<thead>
<tr>
<th>Test Surface2</th>
<th>Grid 20 by 20</th>
<th>Machine</th>
<th>MAHO600E at AIT</th>
<th>HERMLE UWF902H at KU</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Max Errors (mm)</td>
<td>Error Decrease (%)</td>
<td>Path Length (mm)</td>
</tr>
<tr>
<td>Angle Adjustment</td>
<td>6.388</td>
<td>N/A</td>
<td>2367.57</td>
<td>23.199</td>
</tr>
<tr>
<td>CC points insertion</td>
<td>1.209</td>
<td>81.07</td>
<td>2317.06</td>
<td>6.104</td>
</tr>
<tr>
<td>Angle insertion</td>
<td>0.150</td>
<td>97.65</td>
<td>2306.51</td>
<td>0.411</td>
</tr>
<tr>
<td>Angle switching</td>
<td>0.305</td>
<td>95.22</td>
<td>2301.03</td>
<td>8.974</td>
</tr>
</tbody>
</table>
6. Conclusions

We have presented two angle correction algorithms namely, “angle switching” and “angle insertion” to repair the bad trajectories or minimize the large milling error by adjusting the rotation angles in such a way that the kinematics error is minimized. The first algorithm employs the shortest path scheme combined with bad trajectory elimination to optimize tool path. The second algorithm is based on the equi distribution of the rotation angles near singularity position. A special interpolation combined with the bisection methods is employed to insert additional points equally in angular space to obtain a uniform minimal kinematics error. These algorithms are visually verified using our 3D interactive NC program generation (post processing) to generate and simulate the tool trajectory for the five-axis milling machine. Two of the five-axis machines, MAHO600E and HERMLE UWF920H are used to verify the algorithms. Depending on the type of machine, we can simulate and select the appropriate optimization algorithm for the particular surface. Our algorithms have been verified by surfaces having the saddle type regions, multiple extreme, steep regions with a high curvature etc. In particular, such optimization constitutes an efficient tool path in the case of rough machining in the five-axis mode. The numerical experiments demonstrate the accuracy increase ranging from 61 to 98 % depending on the surface and the configuration of the five-axis machine. The software produces impressive results and therefore can be used by the CAD/CAM industries to estimate the errors, to verify the tool path graphically and to produce and optimize the NC programs. The future work includes analysis and correction of the infeasible trajectories such as overcut trajectories, axis out of range and the infeasible angles.

7. Acknowledgements

This research is sponsored by the National Electronics and Computer Technology Center (NECTEC), National Science and Technology Development Agency (NSTDA) of Thailand and The Thailand Research Fund (TRF).

8. References