Available online at www.sciencedirect.com
ScienceDirect
Robotics and Computer-Integrated Manufacturing 24 (2008) 270–286
Robotics and Computer-Integrated Manufacturing
www.elsevier.com/locate/rcim

Direct 5-axis tool-path generation from point cloud input using 3D biarc fitting

K.L. Chui, W.K. Chiu, K.M. Yu*

Department of Industrial and Systems Engineering, The Hong Kong Polytechnic University, Hong Kong, China

Received 21 October 2005; received in revised form 4 October 2006; accepted 24 November 2006

Abstract

In reverse engineering, geometrical information of a product is obtained directly from a physical shape by a digitizing device. To fabricate the product, manufacturing information (usually tool-path) must be generated from a CAD model. The data digitized must be processed and in most cases, a surface model is constructed from them using some of the surface fitting technologies. However, these technologies are usually complicated and the process for constructing a surface patch from a massive digitizing data is time-consuming. To simplify the process for getting tool-path information, a simple algorithm is proposed in this paper. The algorithm is used to generate a 5-axis machining tool-path. Instead of implementing any complicated surface fitting techniques, a direct method is proposed for constructing three-dimensional (3D) triangular mesh from the digitizing data with the mesh points considered as the tool contact locations. Depending on the locations of the points digitized, a decimation procedure is applied such that some of the digitizing data will be filtered out. Then, the tool axis orientations which must be determined in 5-axis tool-path are calculated and the tool center locations are determined accordingly. A 3D biarc fitting technique is applied for all the tool center locations so that a complete 5-axis tool-path is obtained.

Keywords: Machining; Reverse engineering; 5-Axis tool-path; 3D biarc; Point cloud; Triangular mesh

1. Introduction

In product development, stages such as conceptual design, prototype making, CAD model construction, tooling design, etc. are involved. Either forward engineering or reverse engineering can be employed (Fig. 1). In general, in forward engineering, a 3D solid model of a product is first designed in a CAD platform and CAD information is obtained. Then corresponding tool-path information is generated and the product is produced using an appropriate manufacturing process, such as CNC machining. However, in reverse engineering, geometrical information is needed to be obtained from a physical shape directly and these information are converted into a computable format for other downstream processes. In most cases, a digitizing device is used to get the data and the output data is usually a set of scattered points (called a point cloud). Due to the characteristics of the digitizing device, the point cloud can be divided into two main types—a regular point cloud and an irregular point cloud. In the former, intervals between adjacent digitizing points are identical while they are not in the latter. No matter which type of point cloud is obtained, surface fitting techniques are usually applied and a surface model is constructed. Based on the surface model, CAM tool-path information is generated accordingly. This procedure is termed as an indirect machining process and the technologies for fitting surfaces onto a point cloud are essential. Different surface fitting technologies have been developed and some of them will be reviewed. However, surface fitting process is usually time-consuming. In this case, if tool-path information can be achieved directly from the point cloud, the time-consuming surface fitting process can be avoided.

In [1], Chui et al. proposed a direct tool-path generation strategy for a 3-axis tool-path from a massive point input. However, 5-axis tool-path generation was not taken into consideration. Theoretically, there are many advantages of

*Corresponding author. Tel.: +852 27666603; fax: +852 23625267.
E-mail addresses: mfwkchiu@polyu.edu.hk (W.K. Chiu), mfkmyu@polyu.edu.hk (K.M. Yu).

0736-5845/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.rcim.2006.11.004

using 5-axis machining of freeform surfaces over 3-axis machining, such as better surface finish, faster removal rate, higher surface quality, etc. When a sculpture product is machined, a 5-axis machining strategy may be applied. In this case, the tool axis orientation in each tool contact location between the tool and the machined surface must be determined. In general, the tool axis is oriented close to the surface normal. Li et al. [2] and Bohez et al. [3,4] used differential geometry to calculate the corresponding initial tool axis location and orientation. In addition, the tool axis orientation can be defined using different strategies, such as "go through a point" and "go through a circular curve" [5] (Fig. 2). However, for a point cloud obtained from a digitizing device, there is no surface normal information. To solve this problem, a new algorithm is needed. A three-dimensional (3D) biarc fitting technology is employed for fitting the tool centers and the tool axis orientation are inferred from the point cloud so that a complete 5-axis tool-path can be generated.

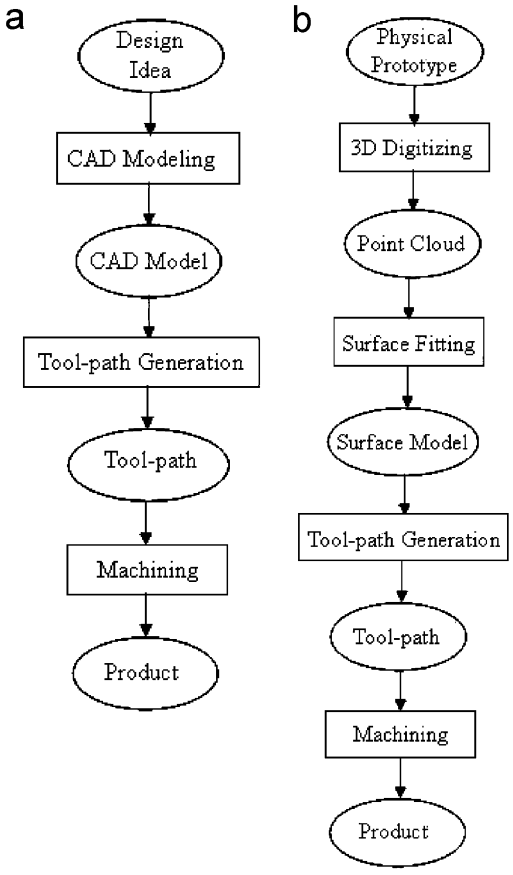

Fig. 1. General workflow of (a) forward engineering and (b) reverse engineering.

2. Reviews

2.1. Surface fitting technologies

There are different surface fitting technologies in different literatures and some of them are reviewed below. Lee et al. [6] employed a Ferguson surface model to fit the measured point data. The original scanning profile is assumed to be a zero twist surface at the four corners. As shown in Fig. 3, the surface can be formed from the four corner points by the following equations:

$$P(u,v) = \begin{bmatrix} F_1(u) & F_2(u) & F_3(u) & F_4(u) \end{bmatrix} \times \begin{bmatrix} P_{00} & P_{01} & P^v_{00} & P^v_{01} \\ P_{10} & P_{11} & P^v_{00} & P^v_{01} \\ P^u_{00} & P^u_{01} & P^{uv}_{00} & P^{uv}_{01} \\ P^u_{10} & P^u_{11} & P^{uv}_{10} & P^{uv}_{11} \end{bmatrix} \times \begin{bmatrix} F_1(v) & F_2(v) & F_3(v) & F_4(v) \end{bmatrix}^{\mathrm{T}},$$

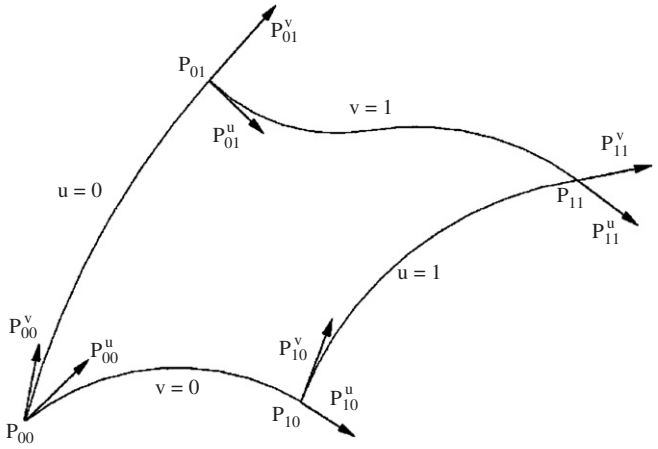

Fig. 3. Ferguson surface patch.

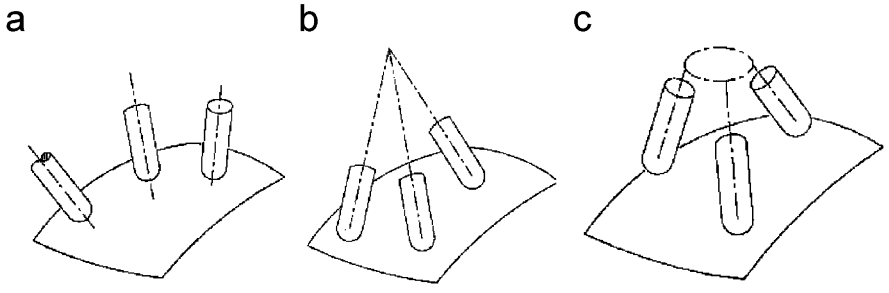

Fig. 2. Different tool axis orientation strategies: (a) normal to surface, (b) go through a point, (c) go through a circular curve.

where P_{00}, P_{01}, P_{10} and P_{11} are the four corner points of the surface and $P_{00}^{uv} = P_{10}^{uv} = P_{01}^{uv} = P_{11}^{uv} = 0$.

Based on the constructed surface, the machining tool-path can be generated.

Due to the excellent shape modification flexibility of B-spline, many researchers tried to fit a B-spline surface to a point cloud. Bradley et al. [7,8] used a laser-scanning technique to obtain the information from a physical product. The laser scanning data was fitted and B-spline freeform surfaces are constructed. In order to keep G^1 continuity at each data point, they used forward differentiation to approximate the instantaneous tangent direction, as shown below:

$$\frac{\partial \mathbf{P}}{\partial u} = \lim_{\Delta u \to 0} \frac{\mathbf{P}(u+\Delta u, v) - \mathbf{P}(u, v)}{\Delta u}.$$

The surface can be calculated by the standard B-spline equation:

$$P(u,v) = \sum_{i=0}^{n} \sum_{j=0}^{m} N_{i,k}(u) N_{j,l}(v) P_{i,j},$$

where

$$N_{i,k}(u) = \frac{(u - t_i) N_{i,k-1}(u)}{t_{i+k-1} - t_i} + \frac{(t_{i+k} - u) N_{i+1,k-1}(u)}{t_{i+k} - t_{i+1}}$$

and

$$N_{i,1} = 1 \quad \text{if} \quad t_i \leqslant u < t_{i+1},$$
$$N_{i,1} = 0 \quad \text{otherwise.}$$

For knot vector

$$t_i = 0 \quad \text{if} \quad i < k,$$
$$t_i = i - k + 1 \quad \text{if} \quad k \leqslant i \leqslant n,$$
$$t_i = n - k + 2 \quad \text{if} \quad i > n.$$

Similarly, Kawabe et al. [9] used an offsetting algorithm and a degree three Bezier surface to fit the scanning data points. The following equation was employed:

$$P(u,v) = \sum_{i=0}^{3} \sum_{j=0}^{3} B_{i,3}(u) B_{j,3}(v) P_{i,j} \quad u, v \in [0,1],$$

where

$$B_{i,3}(u) = \frac{3!}{i!(3-i)!} u^i (1-u)^{3-i} \quad \text{and}$$
$$B_{j,3}(v) = \frac{3!}{j!(3-j)!} v^j (1-v)^{3-j}.$$

In their algorithms, the Bezier curves passed through the measured points. After that the same set of measured points is used to fit the new Bezier curves, which intersect orthogonally with the original curves, as shown in Fig. 4. These Bezier curves serve as the boundaries of the surface patches. From these boundaries, the surface model of the measured physical surface is generated. In general, B-spline possesses the local region editing properties. For any control point, only adjacent curve profile will be affected if its position is altered while this is not true for the Bezier curve.

For these surface fitting technologies, a regular point cloud or a point set in a special pattern is usually required. Apart from the mentioned surface fitting methods for forming a smooth surface, another approach [10] for creating a polygonal mesh from a point set can also be applied. Different triangulation methodologies were developed [10–12]. Since any three non-colinear points can be used to form a triangular facet, the resultant triangular mesh can be achieved by sewing the triangular facets together (Fig. 5).

In [10,11], Choi et al. and Park et al. have proposed similar algorithms. For obtaining a regular mesh, a uniform shape patch pattern has to be set. Choi [12] used the Max–Min angle critirion to control the shapes of the triangles. As shown in Fig. 6, $\angle$**BDC** is smaller than $\angle$**BAC**, and the triangulation result in Fig. 6(b) is preferred. Using the triangulation method, the complicated calculation in creating freeform surfaces can be avoided. However, the accuracy of the mesh with reference to the original shape may be sacrificed.

Using these surface fitting technologies, a surface model is formed and the tool-path can be generated based on the constructed surfaces. Apart from the tool-path generation

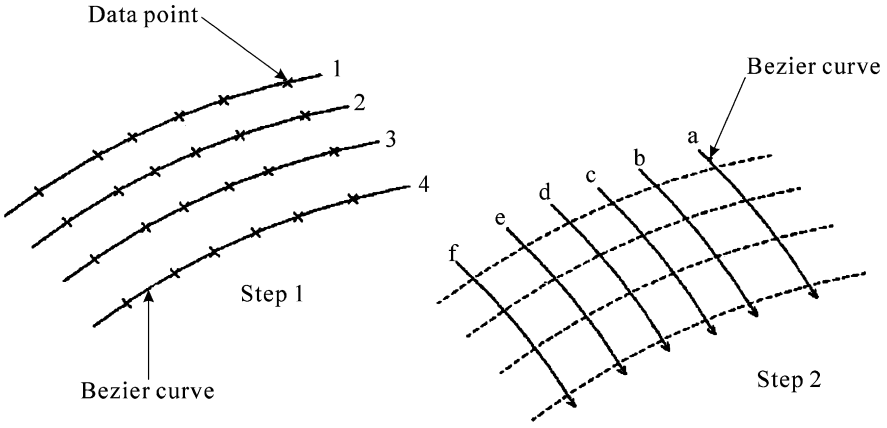

Fig. 4. Surface generating from scanning data.

process, the surface model can also be applied in other applications, such as rendering in product presentation and computer-aided engineering analysis. However, this workflow has also some drawbacks. First, there is the accuracy problem. Second, in view of computation efficiency, long computation time has to be spent and a large amount of memory has to be drawn in the surface fitting process. Therefore, some researchers tried to develop a direct surface machining process for reverse engineering. Generally speaking, a direct machining process may be informally defined as: given a file of digitized data (or a point cloud), a machining tool-path is generated directly without the need to construct any surface patch.

2.2. Direct tool-path generation

Lin [13] employed an xy interpolation method to rearrange a point series in a regular manner. First, a rectangular grid plane is assigned and projected onto the xy-plane. The first row of the points is adjusted so that they are aligned to the first grid line (Fig. 7). The points are then arranged or located in each individual row. Using interpolation technique and assuming $\mathbf{P}_4$, $\mathbf{P}_5$ are on the same z–x plane, the following relation can be established:

$$\mathbf{P}_5 = \mathbf{P}_4 + si - \frac{sh}{k}\mathbf{k}.$$

After the points in each individual row are properly arranged, a similar step is applied to obtain the arrangement order between successive rows. As shown in Fig. 8, the 2D grid points are projected onto the 3D mesh and the projected grid points can be calculated by a linear interpolation or extrapolation from non-aligned digitized points. The tool-path is generated by connecting the points with straight line segments. However, there is one disadvantage in this method. The space intervals between the digitized points are in general very close. The interpolation technique being applied for calculating the positions of the corresponding grid points may result in over-information.

Other than Lin's approach, Lai et al. [14] suggested a different algorithm where the tool-path should be generated from a regular point pattern. A point interpolation process is used to fit each row of points with cubic splines (Fig. 9) and a tool-path can be generated using the cubic splines. Apart from the tool-path, surface patches can also be constructed from the cubic spline curves.

From the reviewed direct tool-path generation methods, a regular point pattern is generally required and interpolation is engaged for the point data regularization. Also, the machining direction is determined from the orientation of the grid being used in the interpolation procedure. In these methods, only 3-axis tool-paths are studied and cubic spline curves and straight line segments are used as the tool-path.

3. The algorithm

In 5-axis machining tool-path generation, the alignment of the tool axis with the surface normal at the point of contact between the surface and the tool must be determined. In the proposed algorithm, the apparent surface normals must be calculated from the point cloud. Then the tool centers and the tool axis orientations along the tool-path can be determined by a solid coordinate geometric analysis.

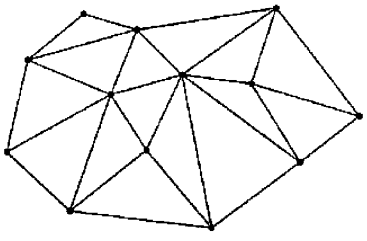

Fig. 5. Triangular patches formed by joining scattered points.

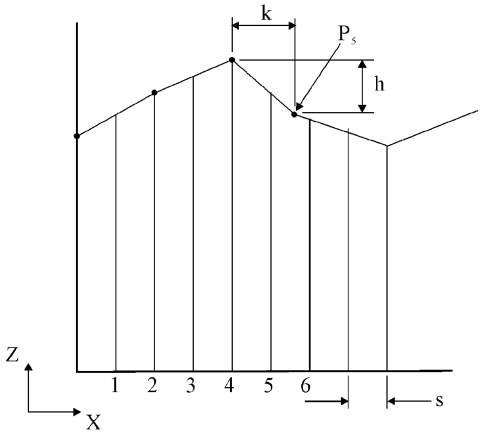

Fig. 7. Grid points interpolation.

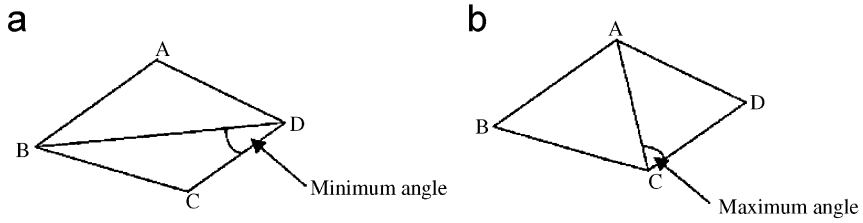

Fig. 6. Maximum and minimum angle critirion for forming triangular patches.

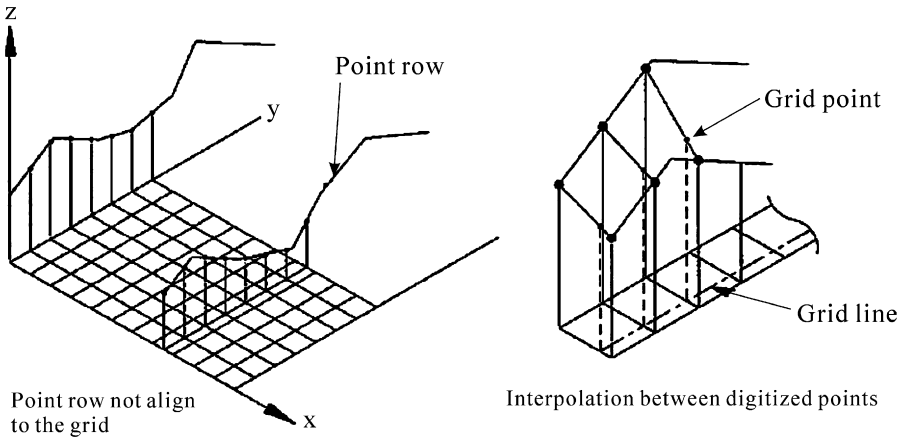

Fig. 8. Interpolation between rows of points.

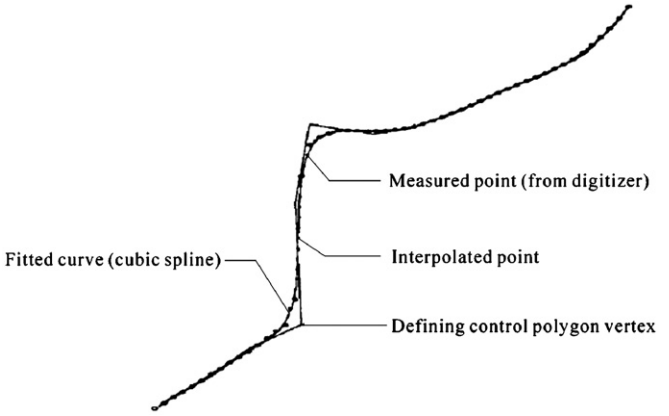

Fig. 9. Point interpolation process to fit a cubic spline onto a row of points.

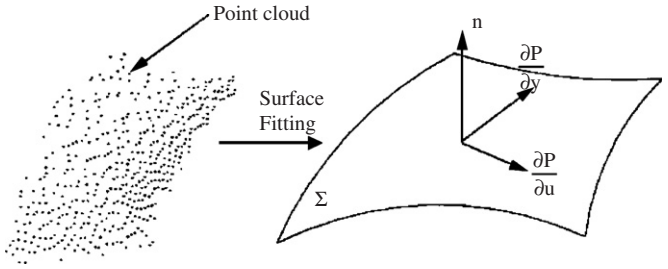

Fig. 10. A surface $\sum$ constructed from a point cloud and the surface normal of an arbitrary point on $\sum$.

In reverse engineering, a smooth surface patch (or patches) is usually constructed from a point cloud and the points are used as the tool contact locations (Fig. 10). The local surface normal for each contact point will then be calculated by taking differentiations on the surface, i.e.

$$\mathbf{n} = \frac{(\partial \mathbf{P}/\partial u) \times (\partial \mathbf{P}/\partial v)}{||(\partial \mathbf{P}/\partial u) \times (\partial \mathbf{P}/\partial v)||}. \quad (1)$$

However, in the proposed algorithm, a 5-axis tool-path is generated without the construction of a surface patch. Since the points being sampled are discrete (discontinuous) and cannot be differentiated, a local surface normal cannot be associated with each point. To obtain the local surface normal of a point, the point and its neighbor points will be used to form a number of triangular facets first. Then its local surface normal is determined based on the normal directions of its neighbor triangular facets and the tool axis inclination in each tool location (i.e. each digitizing point) is calculated. By applying 3D biarc fitting, all the tool centers are fitted by 3D arc splines and a 5-axis G^1 tool-path is formed. In Fig. 11, the workflow of the proposed algorithm is shown and the details will be discussed in subsequent section. In the algorithm, a ball nose tool is assumed to be used.

3.1. Point cloud registration

Since the proposed algorithm is orientation independent, there are no special requirements on the alignment of the physical profile on the digitizing device and so does the pattern of the point cloud. After a point cloud is sampled, a coordinate system (in 3D space) will first be assigned and the point cloud is denoted as $\mathscr{P} = \{\mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_3, \ldots, \mathbf{P}_n\}$. Then a corresponding set of point cloud which is a projection of $\mathscr{P}$ onto the xy-plane is formed and the projection set is denoted as $\mathscr{P}_J = \{\mathbf{P}_{J1}, \mathbf{P}_{J2}, \mathbf{P}_{J3}, \ldots, \mathbf{P}_{Jn}\}$. Each point $\mathbf{P}_{Ji}$ is the projected point of $\mathbf{P}_i$ in $\mathscr{P}$. For the projection set $\mathscr{P}_J$, in general, the "lowest leftmost" point is assigned as the origin. If no appropriate point exists, the position of the y-axis is defined by the point that possesses the minimum x value ($\mathbf{P}_{Jx\text{-min}}$) and the position of the x-axis is defined by the one with the minimum y value ($\mathbf{P}_{Jy\text{-min}}$).

3.2. 3D Triangular mesh construction

In the second step of the algorithm, a 3D triangular mesh, which consists of a number of triangular facets will be formed from the point cloud. To construct the 3D

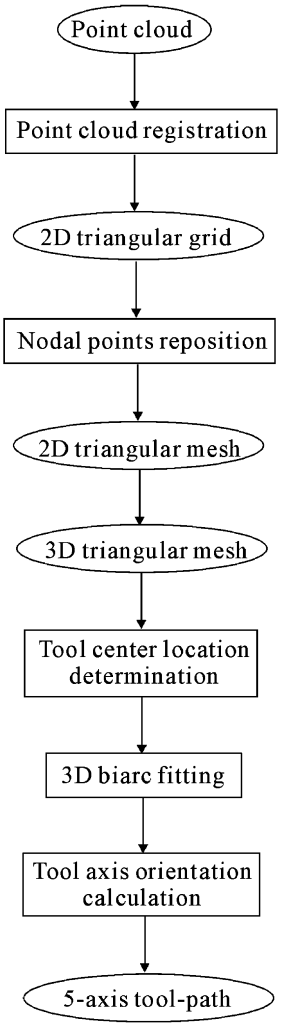

Fig. 11. Workflow of the proposed algorithm.

triangular mesh, corresponding 2D triangular mesh must be formed first. To do this, a 2D triangular grid covering the set $\mathscr{P}_J$ is generated. A pseduo minimum boundary parallelogram (PMBP) is introduced to obtain the two-dimensional (2D) triangular grid.

3.2.1. Pseduo minimum boundary parallelogram (PMBP)

A PMBP is calculated for tracing the 2D triangular mesh. It possesses the following properties:

1. The parallelogram covers all the projected points in $\mathscr{P}_J$.
2. Each element is an equilateral triangle.
3. An oblique coordinate system is established as the global coordinate system for $\mathscr{P}_J$.

The procedures for the formation of the PMBP are shown in Fig. 12. The size is based upon four main projected points in $\mathscr{P}_J$. They are the points having (a) maximum x value, (b) minimum x value, (c) maximum y value and (d) minimum y value. They define a rectangular shape boundary that covers all the projected points.

The lower left corner of PMBP can be defined by simple geometric relationship:

$$\mathbf{P}_{J\text{lower\_left\_corner}} = (\mathbf{P}_{J\text{lower\_left\_corner}} \cdot i)i + (\mathbf{P}_{J\text{lower\_left\_corner}} \cdot j)j,$$

$$\mathbf{P}_{J\text{lower\_left\_corner}} \cdot i = \mathbf{P}_{Jx\text{-min}} \cdot i - \frac{(\mathbf{P}_{Jy\text{-max}} - \mathbf{P}_{Jy\text{-min}}) \cdot j}{\tan(\pi/3)},$$

$$\mathbf{P}_{J\text{lower\_left\_corner}} \cdot j = \mathbf{P}_{Jy\text{-min}} \cdot j. \quad (2)$$

The upper right corner can be obtained using similar approach:

$$\mathbf{P}_{J\text{upper\_right\_corner}} = (\mathbf{P}_{J\text{upper\_right\_corner}} \cdot i)i + (\mathbf{P}_{J\text{upper\_right\_corner}} \cdot j)j,$$

$$\mathbf{P}_{J\text{upper\_right\_corner}} \cdot i = \mathbf{P}_{Jx\text{-max}} \cdot i - \frac{(\mathbf{P}_{Jy\text{-max}} - \mathbf{P}_{Jy\text{-min}}) \cdot j}{\tan(\pi/3)},$$

$$\mathbf{P}_{J\text{upper\_rigt\_corner}} \cdot j = \mathbf{P}_{Jy\text{-max}} \cdot j, \quad (3)$$

where $\mathbf{P}_{Jx\text{-min}}$ is the point possesses minimum x value, $\mathbf{P}_{Jy\text{-min}}$ the point possesses minimum y value, $\mathbf{P}_{Jx\text{-max}}$ the point possesses maximum x value, $\mathbf{P}_{Jy\text{-max}}$ the point possesses maximum y value.

For the upper left corner point, it equals to the intersection from the horizontal line drawn from $\mathbf{P}_{Jy\text{-max}}$ and vertical line drawn from $\mathbf{P}_{Jx\text{-min}}$. Similarly, for the lower right corner point, it comes from the intersection point from the horizontal line drawn from $\mathbf{P}_{Jy\text{-min}}$ and vertical line drawn from $\mathbf{P}_{Jx\text{-max}}$:

$$\mathbf{P}_{J\text{upper\_left\_corner}} = (\mathbf{P}_{J\text{upper\_left\_corner}} \cdot i)i + (\mathbf{P}_{J\text{upper\_left\_corner}} \cdot j)j,$$

$$\mathbf{P}_{J\text{upper\_left\_corner}} \cdot i = P_{Jx\text{-min}} \cdot i,$$
$$\mathbf{P}_{J\text{upper\_left\_corner}} \cdot j = \mathbf{P}_{Jy\text{-max}} \cdot j, \quad (4)$$

$$\mathbf{P}_{J\text{lower\_right\_corner}} = (\mathbf{P}_{J\text{lower\_right\_corner}} \cdot i)i + (\mathbf{P}_{J\text{lower\_right\_corner}} \cdot j)j,$$

$$\mathbf{P}_{J\text{lower\_right\_corner}} \cdot i = \mathbf{P}_{Jx\text{-max}} \cdot i,$$
$$\mathbf{P}_{J\text{lower\_right\_corner}} \cdot j = \mathbf{P}_{Jy\text{-min}} \cdot j. \quad (5)$$

Once the boundary area is defined, the second step is the grid line assignment. The grid is constructed from horizontal, inclined 60° and inclined 120° lines and the grid spacing is equal to the length of the triangle.

The horizontal line will be drawn upwards from the lower boundary line with spacing t along the slant 60° direction. It should be noted that the upper boundary may not pass through $\mathbf{P}_{Jy\text{-max}}$. This is because the length of the slant side of PMBP may not be an integral multiplier of t (Fig. 12(f)). The number of rows of triangle in PMBP is determined from

$$h = \frac{\mathbf{P}_{Jy\text{-max}} \cdot j - \mathbf{P}_{Jy\text{-min}} \cdot j}{t \sin(\pi/3)}, \quad (6)$$

where $\mathbf{P}_{jy\text{-max}} \cdot j$ is the y coordinate value of point $\mathbf{P}_{Jy\text{-max}}$, $\mathbf{P}_{jy\text{-min}} \cdot j$ the y coordinate value of point $\mathbf{P}_{Jy\text{-min}}$.

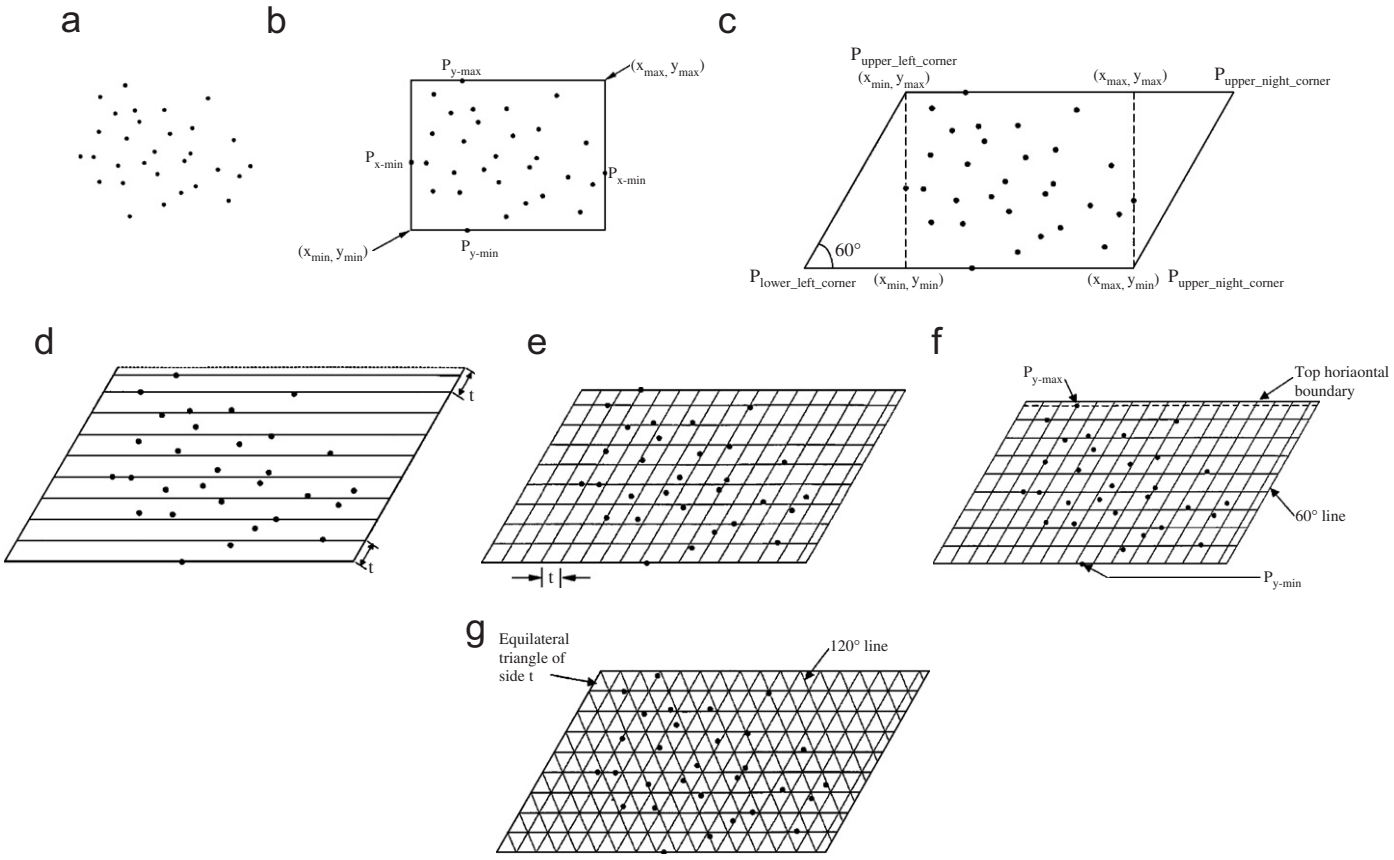

Fig. 12. Procedures for forming a PMBP from a point cloud.

If h is an integer, the upper boundary passes through $\mathbf{P}_{Jy\text{-max}}$; whereas if h is not an integer, the upper boundary does not pass through $\mathbf{P}_{Jy\text{-max}}$. For the latter case, an appropriate value is defined by rounding up the value of h. This guarantees that all points will be covered by the grid. In the next step, the 60° inclined line is drawn from the left hand side to the right hand side. The 120° inclined line will be drawn, from "top left" to "bottom right ". In a PMBP, the lower left corner is defined as the origin position of an oblique coordinate system. For any grid point $\mathbf{P}_{gi,j}$, it will have a position vector $\mathbf{P}_{gi,j} = h_1 e_1 + h_2 e_2$ (Fig. 13) where h_1, h_2 are real.

Since the grid is constructed by equilateral triangle

$$e_1 = i \quad \text{and} \quad e_2 = i + \sqrt{3}j.$$

For discrete grid point (i,j),

$$\mathbf{P}_{gi,j} = ie_1 + je_2.$$

For example, the corresponding point for $(i,j) = (1,2)$ is

$$\mathbf{P}_{g1,2} = i + 2(i + \sqrt{3}j) = 3i + 2\sqrt{3}j.$$

Since the grid structure is defined by an oblique coordinate system with horizontal and inclined iso-parametric lines, each node will have the corresponding indexing. For the ease of subsequent tool-path planning process, the vertex of the triangle for each row will be stored in two groups. The triangles in the first row which point upwards will be in one group and those point downwards will be in another group, as shown in Fig. 14. As a result, a triangular grid for the set $\mathscr{P}_J$ is formed. The corresponding triangular facets of the 3D triangular mesh will follow the 2D grouping results.

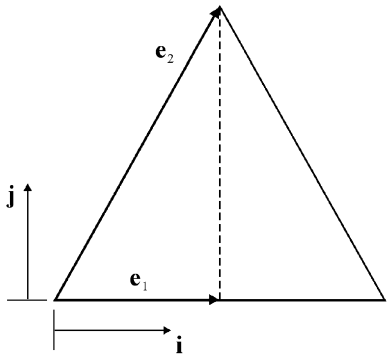

Fig. 13. An oblique coordinate system.

3.2.2. Nodal point reposition

Nodal point reposition is in essence a filtering process. At each grid node (also called nodal point), the projected point that is closest to the pre-defined grid node is

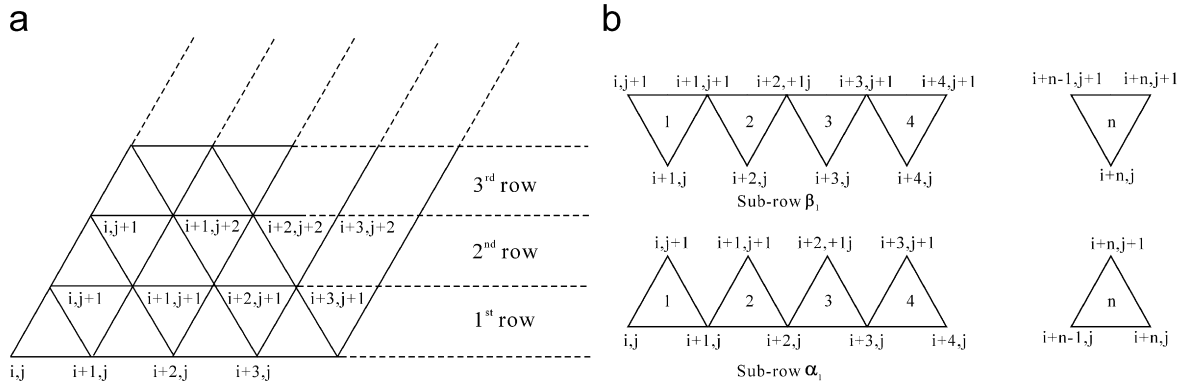

Fig. 14. Grouping of rows of triangles.

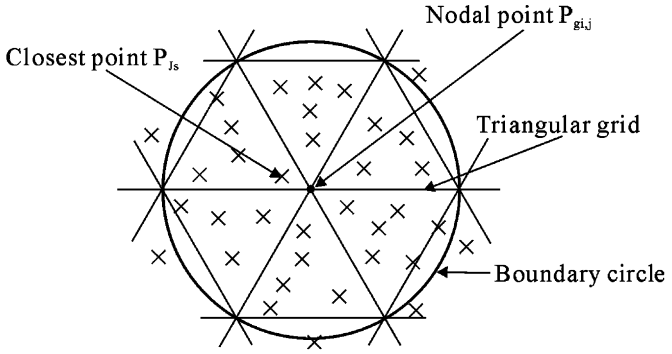

Fig. 15. Point selection approach.

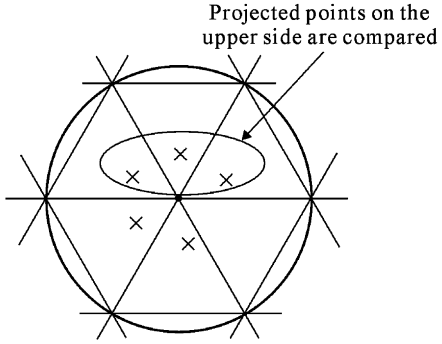

Fig. 16. An example case for choosing projected points using Critirion A.

identified. For all nodal points in the grid, a circle is drawn with its radius equals to the grid interval. For a grid node, only the projected points inside its circle are considered and the projected point that has the shortest distance to the grid node will be recorded as the vertex point for the corresponding node. The procedure is repeated for other grid nodes and points, and those that have not been selected will be deleted.

For example, in Fig. 15, the closest projected point $\mathbf{P}_{Js}$ will be considered as the new position for the nodal point $\mathbf{P}_{gi,j}$. However, in case two or more points satisfying the shortest distance condition, the following criteria will be applied.

Critirion A. If two or more projected points are equal in distance to $\mathbf{P}_{gi,j}$ but are located in different triangles, the projected points in the upper side (i.e. larger y) will be chosen. For the example in Fig. 16, five projected points are in equal distance to the nodal point. Since more than one projected point are on the upper side, the new position of the nodal point will be clarified using Critirion B.

Critirion B. If two or more projected points are equal in distance, on the same sub-row but in different triangles, the projected points in the middle triangle will be chosen as the new position of the nodal point under consideration (Fig. 17).

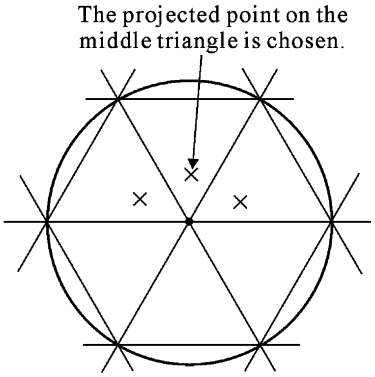

Fig. 17. A projected point is selected as the new nodal point position using Critirion B.

Critirion C. If two or more projected points are equal in distance, on the same sub-row and on the same triangle, the projected point having the largest x value will be chosen (Fig. 18). From the newly positioned vertex points, a series of triangular facets can be traced out such that each grid node will have its corresponding projected point (Fig. 19).

After all the nodal points of the triangular grid are repositioned, a 2D triangular mesh with all the mesh vertices being the member of the set $\mathscr{P}_J$ is constructed. This 2D triangular mesh will be used to form a 3D mesh in the next step.

3.2.3. 3D triangular mesh formation

For all vertices in the 2D triangular mesh, the corresponding digitizing points can be found in the set $\mathscr{P}$. A 3D triangular mesh can then be constructed by arranging points in $\mathscr{P}$ in the same order as the 2D triangular mesh vertices in $\mathscr{P}_J$, as shown in Fig. 20. The mesh vertices are considered as the tool contact locations of the machining tool-path. In the proposed algorithm, a zigzag tool-path is used. In the next step, the tool axes and the tool centers for these tool positions on the 3D triangular mesh will be determined.

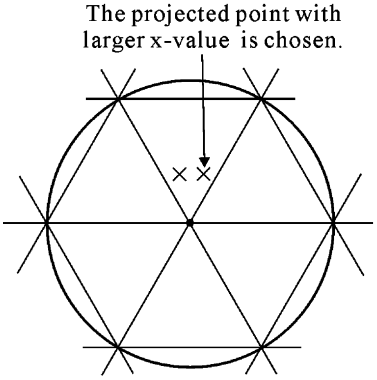

Fig. 18. Projected point being chosen using Critirion C.

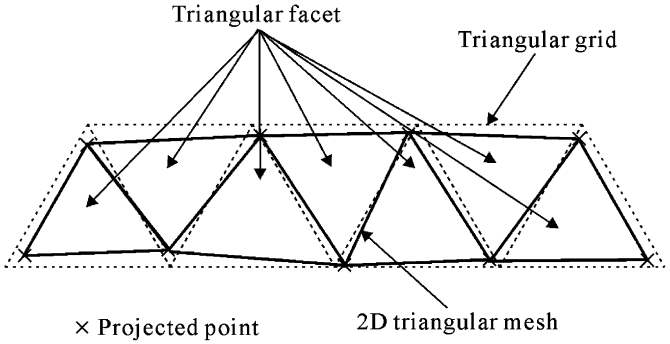

Fig. 19. Formation of a 2D triangular mesh from the projected points after a nodal reposition procedure.

3.3. Tool center location determination

The local surface normals at the tool contact locations are calculated by an approximation method. For instance, the tangent value can be approximated by the central difference equation:

$$f'(x) = \frac{f(x+\Delta x) - f(x-\Delta x)}{2\Delta x},$$

where Δx stands for the interval between points.

However, this approach is applicable to a regular point cloud only and not for the general irregular case. An alternative is the Gouraud shading algorithm [15] where the vector sum from adjacent points can be used to calculate the local surface normal of a point. As shown in Fig. 21, the normal vector of the point under consideration is equal to the average of its adjacent points' normal vectors, i.e.

$$\mathbf{n} = \frac{\mathbf{n}_1 + \mathbf{n}_2 + \mathbf{n}_3 + \mathbf{n}_4}{4}. \tag{7}$$

However, the result is affected by the criteria being used in selecting the adjacent points.

In the proposed algorithm, the local surface normal $\mathbf{n}_i$ of point $\mathbf{P}_i$ is obtained from the normal vectors $\mathbf{n}_{mj}$ of its incident triangular facets rather than the normal vectors of its adjacent points (Fig. 22). Eq. (7) is thus modified as

$$\mathbf{n}_i = \frac{\sum_{j=1}^{q} \mathbf{n}_{mj}}{q}, \tag{8}$$

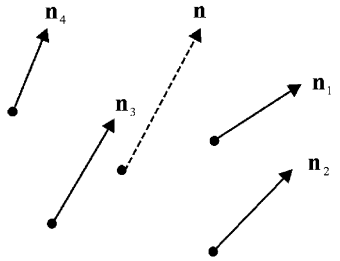

Fig. 21. Calculation of the normal vector $\mathbf{n}$ of a point from its neighbor points normals.

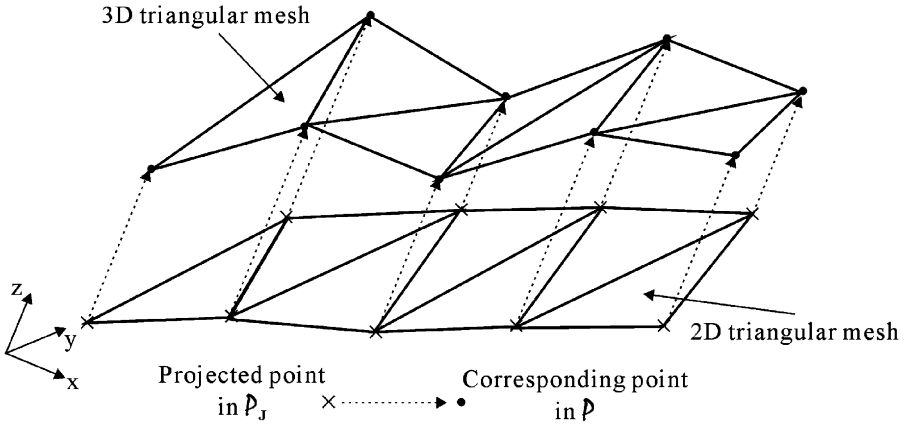

Fig. 20. 3D triangular mesh is constructed based on its corresponding 2D triangular mesh.

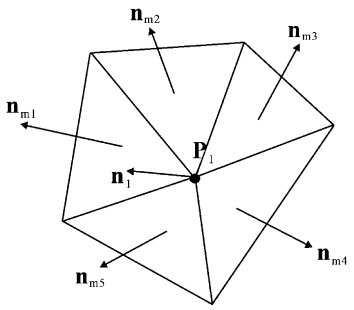

Fig. 22. Local surface normal of a point $\mathbf{P}_1$ is inferred from its neighbor facets normals.

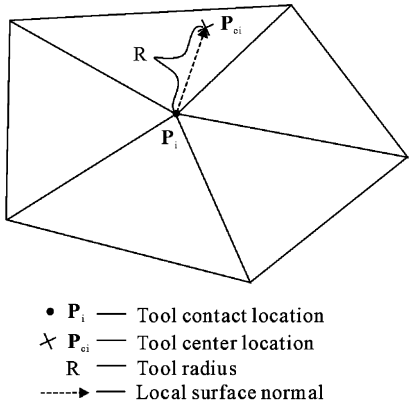

Fig. 23. Offsetting the tool contact point to find the corresponding tool center location.

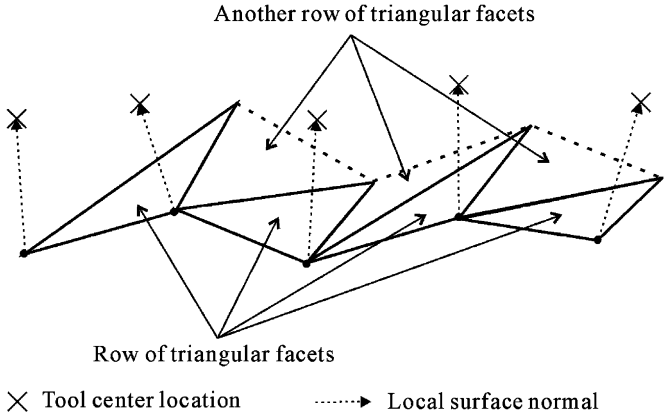

Fig. 24. A row of tool centers associated with a row of triangular facets.

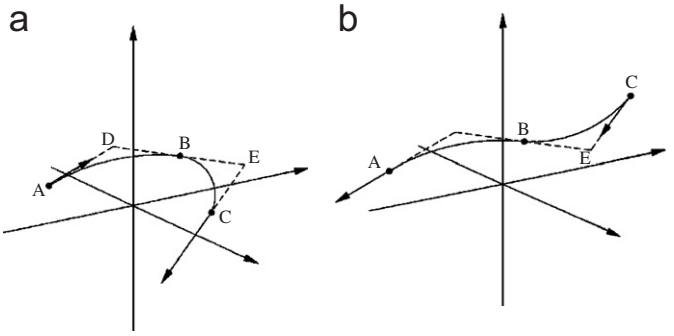

Fig. 25. C-type and an S-type biarcs.

where $\mathbf{n}_{mi}$ is the normal vector of triangular facet j incident to $\mathbf{P}_i$ and q is the number of triangular facets.

Using Eq. (8), the local surface normals $\mathbf{n}_i$ for all points $\mathbf{P}_i$ on the 3D triangular mesh can be calculated. The tool center locations can also be calculated by offsetting the tool contact points along the local surface normals by an amount equal to the tool radius (Fig. 23). The tool axis orientation will then be determined after the tool center locations are fitted by 3D biarc curves.

3.4. 3D arc spline tool-path fitting

As mentioned in Section 3.2.1, the triangular facets are grouped into rows. Each row of tool centers is associated with a row of triangular facets, as shown in Fig. 24. In general, the post-processor of NC machine can read and write straight line and circular arc segments. If complex or higher degree curves are used as tool-path and this tool-path is input to the machine, the tool-path has to be broken down into or approximated by arc splines. Sharrock et al. [16] suggested an algorithm of 3D arc spline interpolation.

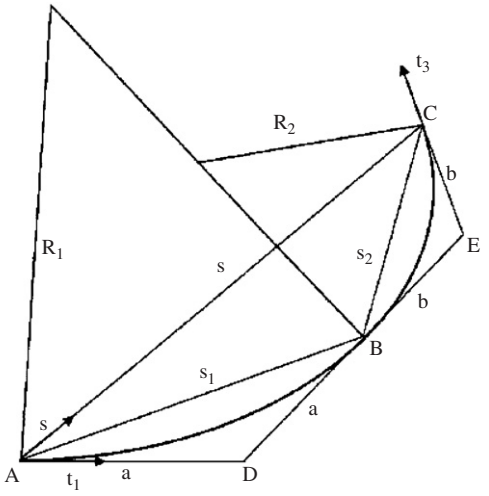

Fig. 26. Construction of a 3D biarc.

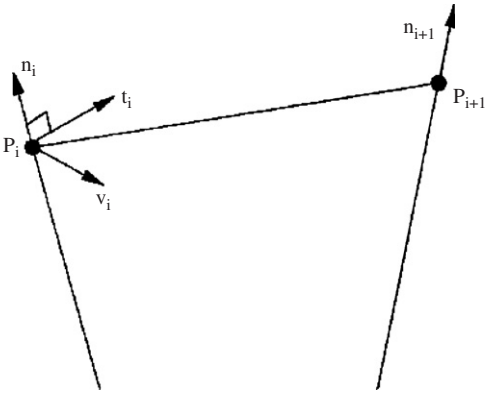

Fig. 27. Tangent vector calculated from cross product.

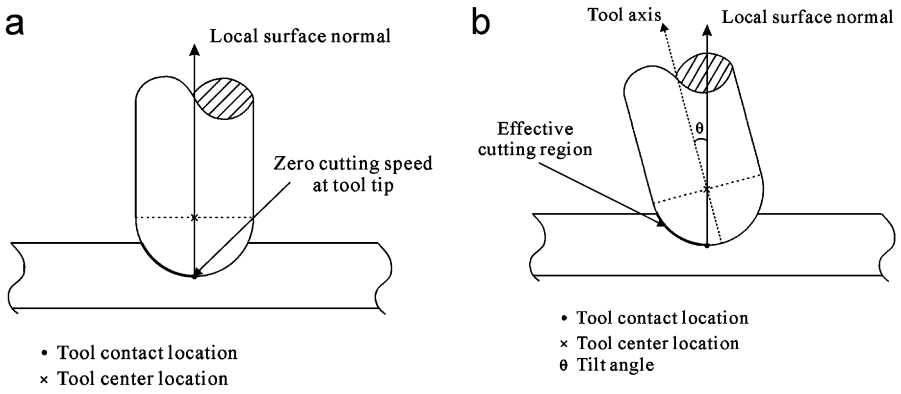

Fig. 28. Change of effective cutting region by varying the tilt angle.

However, their algorithm has not been applied to 5-axis tool-path generation. Normally, a 3D biarc is constructed from two planar circular arcs lying on the same plane. Similar to 2D biarc fitting, there are two main categories—C-type and S-type (Fig. 25) where the endpoint tangent vectors will control the shape of the arc. As a result, the tool-path can be created by joining the tool contact points directly with 3D arc splines.

From Fig. 26, the curvature of a 3D biarc can be found by the following equations:

$$\kappa_1 = \frac{\|\mathbf{t}_1 \times s - b\mathbf{t}_1 \times \mathbf{t}_3\|}{s^2/2 - b(\mathbf{s}\cdot\mathbf{t}_3)},$$

$$\kappa_2 = \frac{\|\mathbf{s} \times \mathbf{t}_3 - a\mathbf{t}_1 \times \mathbf{t}_3\|}{\mathbf{s}^2/2 - a(\mathbf{s}\cdot\mathbf{t}_1)},$$

where

$$a = (-\mathbf{s}\cdot\mathbf{t}_3) + u/w,$$
$$b = (-\mathbf{s}\cdot\mathbf{t}_1) + u/w$$

and

$$u^2 = (\mathbf{s}\cdot\mathbf{t}_1)(\mathbf{s}\cdot\mathbf{t}_3) + w(\mathbf{s}^2/2),$$
$$w = 1 - (\mathbf{t}_1\cdot\mathbf{t}_3).$$

Then the radii for the two arcs become

$$R_1 = 1/\kappa_1 \quad \text{and} \quad R_2 = 1/\kappa_2.$$

(The detailed derivations are given in Appendix A.)

The tangent vector (i.e. $\mathbf{t}_1$ and $\mathbf{t}_3$) at each tool position (i.e. **A** and **C**) should be calculated prior to the construction of the 3D biarc. At point $\mathbf{P}_i$, a line segment is drawn to join with its adjacent point $\mathbf{P}_{i+1}$ and the cross product ($\mathbf{v}_i$) is computed (Fig. 27), i.e.

$$\mathbf{v}_i = \mathbf{P}_i\mathbf{P}_{i+1} \times \mathbf{n}_i.$$

Then the tangent vector ($\mathbf{t}_i$) at point $\mathbf{P}_i$ can be found as

$$\mathbf{t}_i = \mathbf{n}_i \times \mathbf{v}_i = \mathbf{n}_i \times (\mathbf{P}_i\mathbf{P}_{i+1} \times \mathbf{n}_i).$$

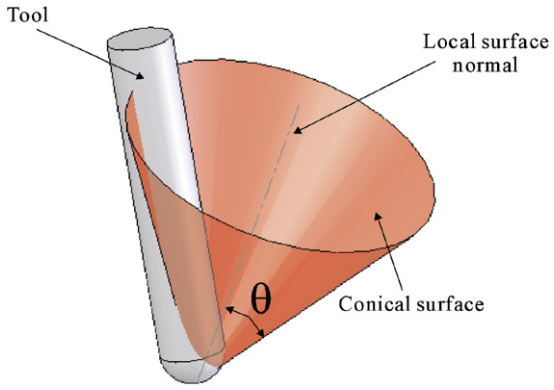

Fig. 29. Tool axis is on the conical surface with the local surface normal as the surface's axis.

3.5. Inclined tool axis orientations calculation

In the proposed algorithm, the local surface normal will not be the tool axis orientation. This is because in actual machining process, the cutting speed at the tool tip is zero and no material will be cut off. In practice, the tool should be oriented at a tilt angle to the local surface normal so that the surface can be cut by the tool blades, as shown in Fig. 28. In general, the tilt angle would be around 15° and this provides a guideline for setting the actual tool axis orientation in the tool-path generation process.

However, the exact tool axis orientation cannot be fixed if only the tilt angle is given (Fig. 29). In order to determine the exact tool axis orientation, a method using a moving Frenet frame [17] along the curve fitting the tool center locations is proposed, as shown in Fig. 30. The tool axis orientation is calculated as follows.

Assume $\mathbf{P}(u)$ to be the tool center fitting curve, then tangent $\mathbf{t}(u_i)$ at the ith tool center is

$$\mathbf{t}(u_i) = \frac{\dot{\mathbf{P}}(u_i)}{\dot{s}},$$

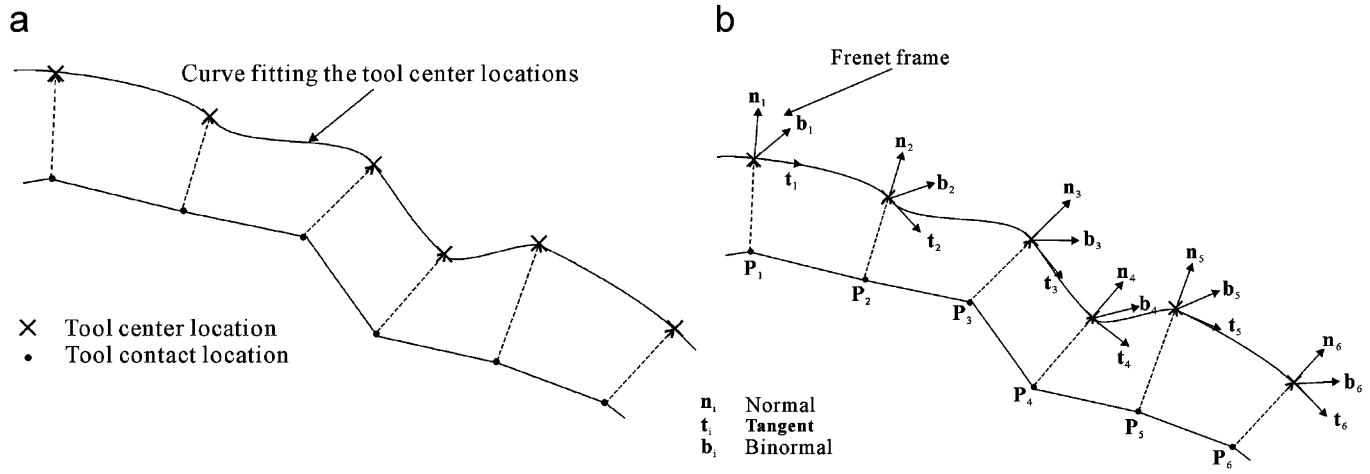

Fig. 30. (a) Tool center locations are fitted with a curve, (b) Frenet frame along the curve fitting the tool centers.

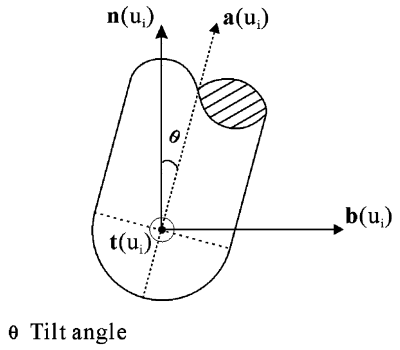

Fig. 31. Tilt angle between tool axis orientation and normal.

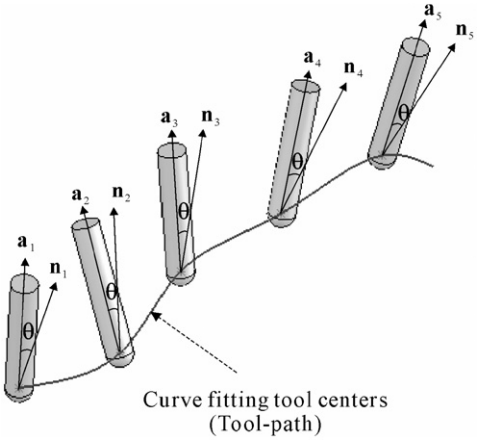

Fig. 32. 5-Axis tool-path with the tool axis tilts at an angle to the normal.

where u is the curve defining parameter, s the arc length and $\dot{s} = (\mathrm{d}s/\mathrm{d}u)$ and

$$\mathbf{n}(u_i) = \frac{\dot{\mathbf{t}}(u_i)}{\kappa_i \dot{s}},$$

where κ_i is the curvature of $\mathbf{P}(u)$ at the ith tool center.

If $\mathbf{a}(u_i)$ is the tool axis orientation at the ith tool center, then

$$\mathbf{a}(u_i) = \mathbf{a}(u_i)\cdot\mathbf{t}(u_i)\mathbf{t}(u_i) + \mathbf{a}(u_i)\cdot\mathbf{n}(u_i)\mathbf{n}(u_i) + \mathbf{a}(u_i)\cdot\mathbf{b}(u_i)\mathbf{b}(u_i), \tag{9}$$

where $\mathbf{n}(u_i)$, $\mathbf{t}(u_i)$ and $\mathbf{b}(u_i)$ are the normal, tangent and binormal vectors, respectively.

To simplify the calculation and maintain a smooth transition of the tool axis orientation during machining, the tool axis orientation vector is assumed to be on the nb-plane, i.e.

$$\mathbf{a}(u_i)\cdot\mathbf{t}(u_i) = \cos 90^\circ = 0.$$

Then Eq. (9) is modified as

$$\mathbf{a}(u_i) = \mathbf{a}(u_i)\cdot\mathbf{n}(u_i)\mathbf{n}(u_i) + \mathbf{a}(u_i)\cdot\mathbf{b}(u_i)\mathbf{b}(u_i). \tag{10}$$

Assume θ is the tilt angle between the tool axis orientation and the normal (Fig. 31), then

$$\mathbf{a}(u_i)\cdot\mathbf{n}(u_i) = \cos\theta \quad \text{and} \quad \mathbf{a}(u_i)\cdot\mathbf{b}(u_i) = \cos(90^\circ - \theta). \tag{11}$$

Substituting Eq. (11) into (10), Eq. (10) becomes

$$\mathbf{a}(u_i) = \cos\theta\,\mathbf{n}(u_i) + \cos(90^\circ - \theta)\mathbf{b}(u_i). \tag{12}$$

When the tilt angle θ is set by the user, the actual tool axis orientations for all the tool centers can be calculated using Eq. (12). As a result, a practical 5-axis tool-path is generated (Fig. 32).

4. Result

To verify the proposed algorithm, a sample model, which is the front panel of a model car is used as an example. The size of the model is 74 mm (x) × 54 mm (y) × 15 mm (z). The triangular grid size is 4 mm, the tool radius of the ball nose tool being used in this example is 8 mm and the depth of cut is 0.3 mm. The number of points input after a 3D scanning process is 17,036 points and this

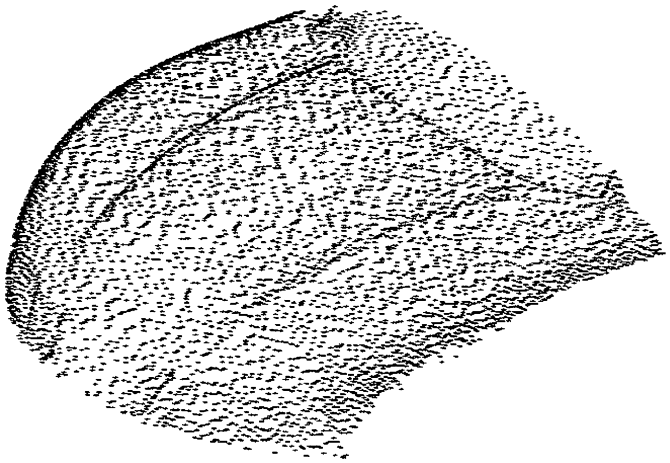

Fig. 33. Point cloud obtained from scanning a physical shape.

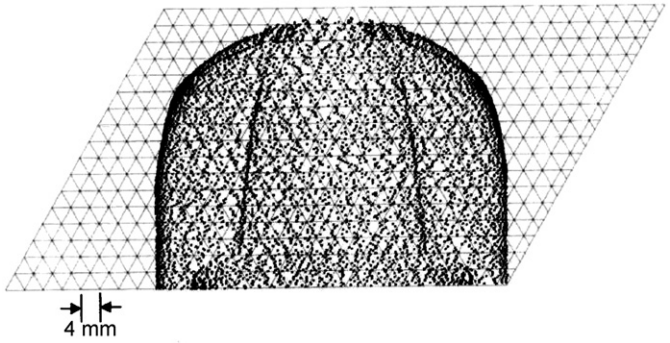

Fig. 34. Formation of a PMBP and the triangular grid for the point cloud.

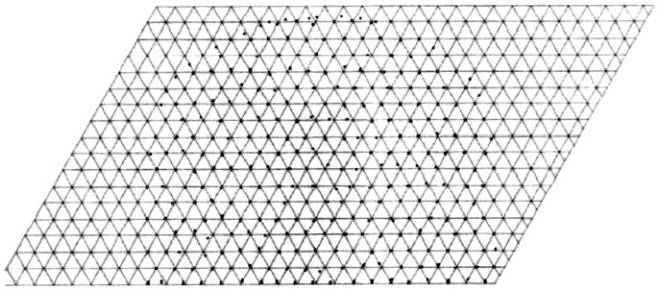

Fig. 35. Points retained after the nodal point reposition procedure.

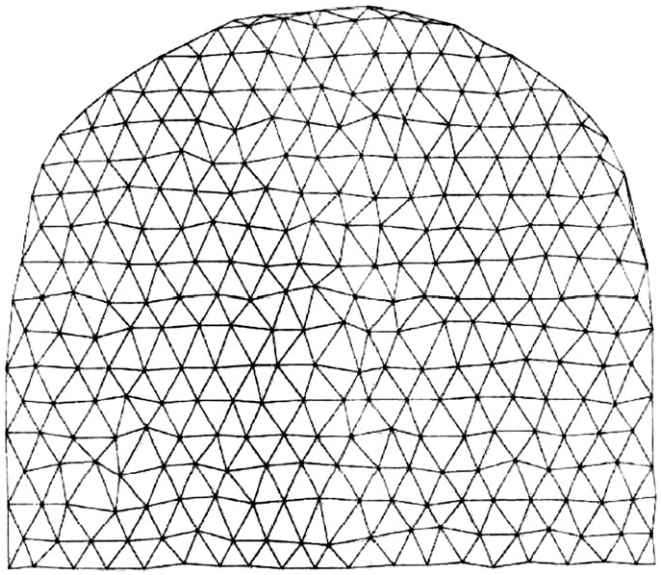

Fig. 36. 2D triangular mesh.

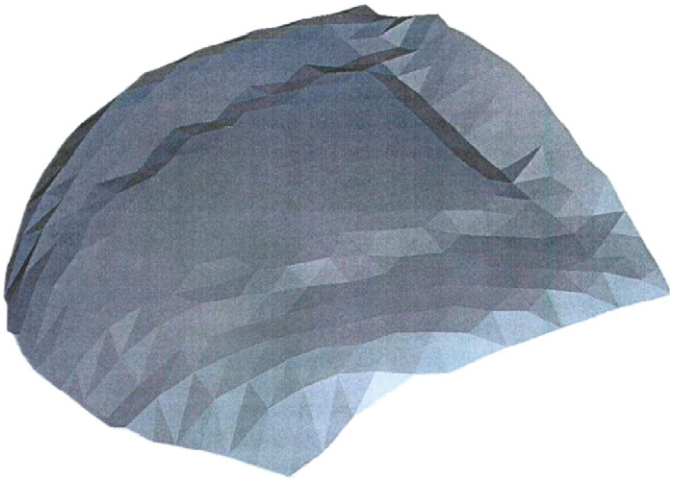

Fig. 37. 3D triangular mesh.

point cloud is assigned with a coordinate system (Fig. 33). The point cloud is projected onto an xy-plane and formed a projected set $\mathscr{P}_J$. According to the algorithm, a PMBP is found for $\mathscr{P}_J$ and a 2D triangular grid is constructed (Fig. 34). For each grid point, the closest projected point is identified and projected points other than those recorded points are filtered out. Eventually, 343 points are recorded (Fig. 35). The 2D triangular grid is altered such that each grid point is relocated at its closest projected point and a 2D triangular mesh for $\mathscr{P}_J$ is formed (Fig. 36). The projected points are the mesh points of this 2D mesh and their corresponding points (in $\mathscr{P}$) are used to form a 3D triangular mesh. The digitizing points are also the vertices of the triangular facets, as shown in Fig. 37. For the 3D triangular facets, their normal vectors are calculated (Fig. 38) and the local surface normals of the vertices are calculated using Eq. (8). Once the local surface normals are defined, the tool centers can be found. At each row of triangular facets, 3D biarc is used to join the corresponding tool centers with tool orientations are calculated using Eq. (12) (Fig. 39). A 5-axis tool-path is thus generated. In Fig. 40, the NC verification result of the 5-axis tool-path is shown.

Another sample model which is a part of a human face is also used to verify the algorithm. The size of the model is 90 mm (x) $\times$ 165 mm (y) $\times$ 60 mm (z). The number of points input after a 3D scanning process is 26,493 points and the point cloud is shown in Fig. 41. The PMBP of the points is formed and 1099 points are retained after the nodal point reposition procedure. The corresponding 2D

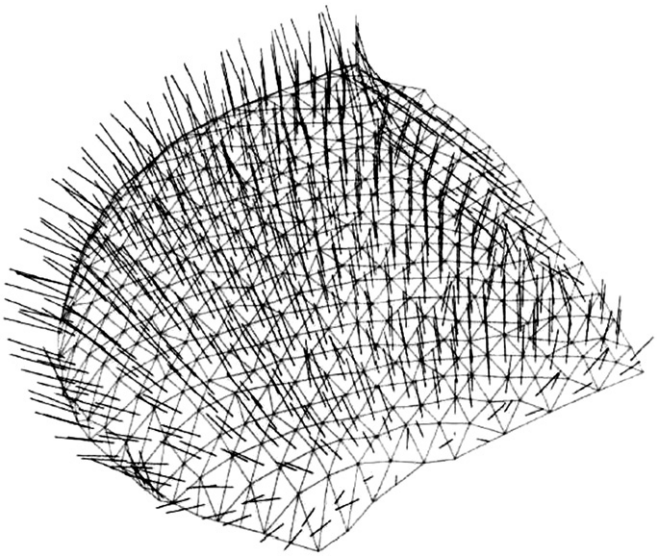

Fig. 38. Local surface normal vectors.

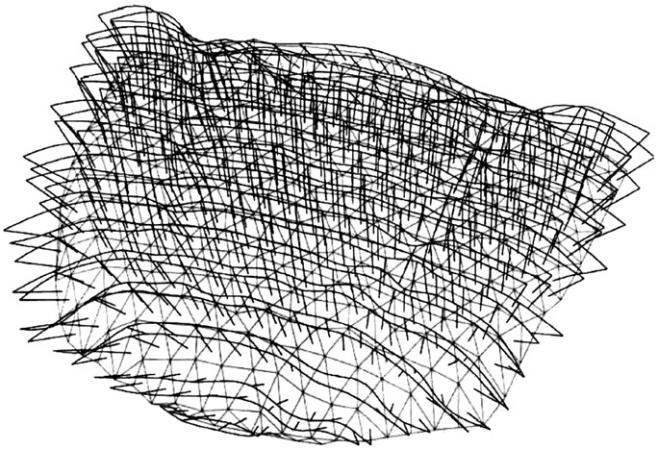

Fig. 39. 3D biarcs tool-path.

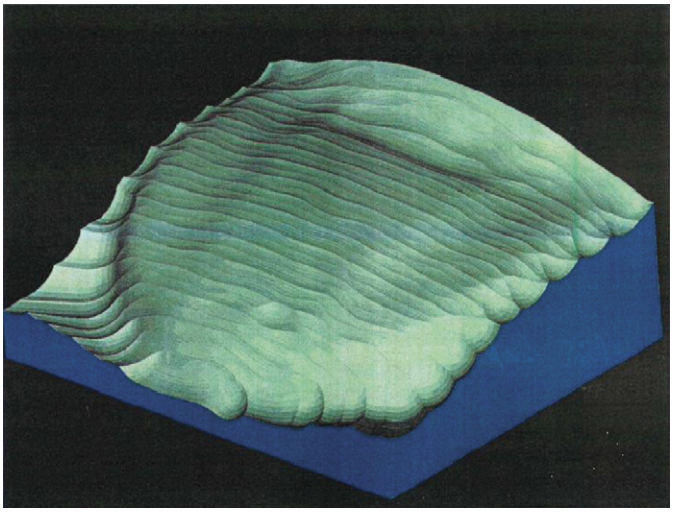

Fig. 40. NC tool-path verification.

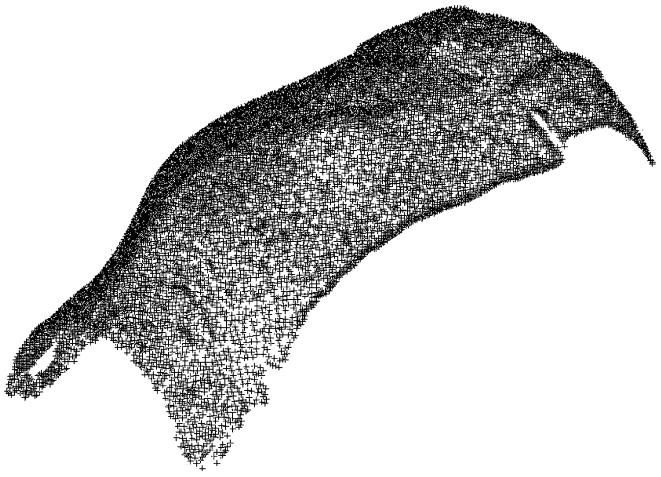

Fig. 41. Point cloud of a human face.

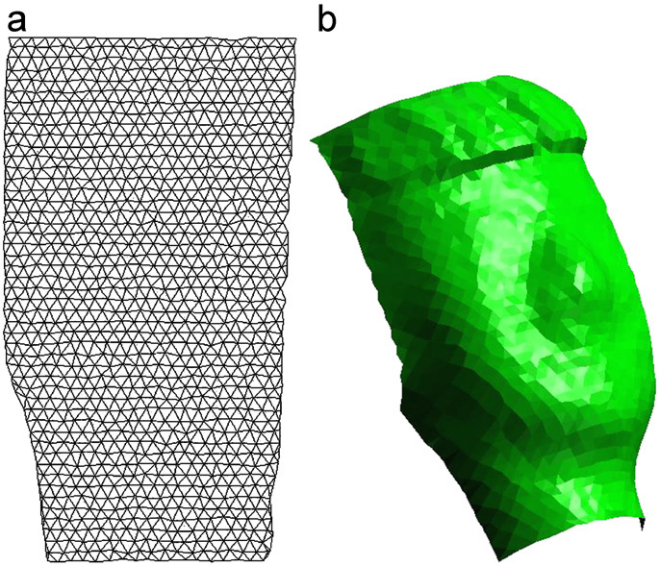

Fig. 42. (a) 2D triangular mesh, (b) 3D triangular mesh.

and 3D triangular meshes are shown in Fig. 42. In this example, the tool radius of the ball nose tool being used is 10 mm and the depth of cut is 0.5 mm. A 5-axis tool-path is generated following the procedures of the proposed algorithm and the NC verification result is shown in Fig. 43.

5. Discussion

In this paper, an algorithm is proposed for generating a 5-axis tool-path from a point cloud obtained from a digitizing device. However, there are some drawbacks which must be considered. In the algorithm, a 2D triangular grid is constructed and the size of the grid has to be set. However, the fineness of the grid would affect the accuracy of the machined surface to the actual shape. It can be seen that the number of digitizing points being filtered out in the nodal point reposition step is affected by

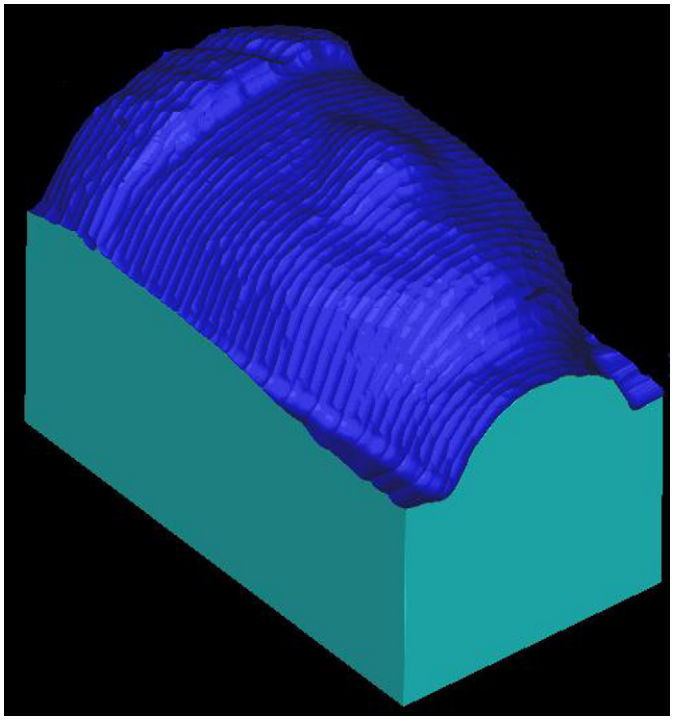

Fig. 43. NC tool-path verification.

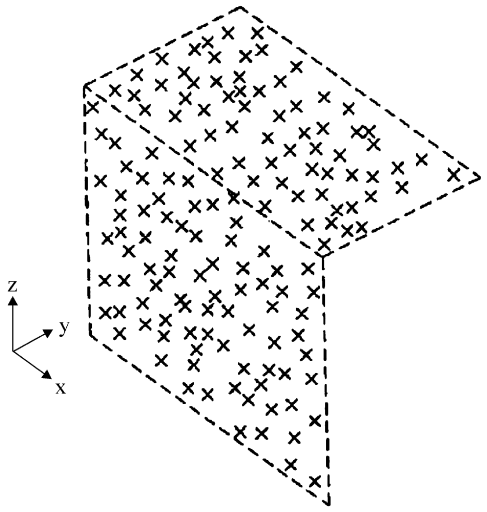

Fig. 44. Point cloud of a vertical surface.

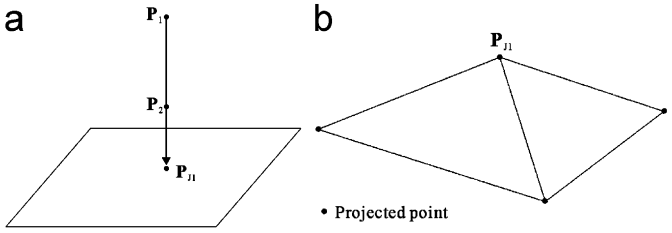

Fig. 45. (a) Two scanning points ($\mathbf{P}_1$ and $\mathbf{P}_2$) being projected onto a same projected point $\mathbf{P}_{J1}$; (b) 2D triangular mesh if $\mathbf{P}_{J1}$ is retained on the mesh.

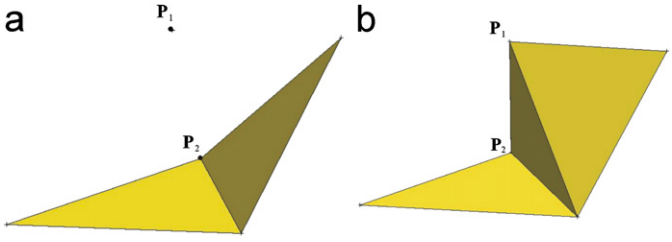

Fig. 46. (a) Vertical surface which $\mathbf{P}_1$ and $\mathbf{P}_2$ are on cannot be retained in the 3D triangular mesh. (b) The vertical surface can be formed in the mesh.

the grid fineness. If a coarse grid is used (i.e. the grid size is in the order of several mm), too many digitizing points may be filtered out. A rough tool-path is resulted and a smooth surface cannot be machined out.

When a surface with high geometrical complexity is scanned, the surface (and also its point cloud) may be subdivided into a number of sub-surfaces (and corresponding sub-point clouds) such that the complexity of each sub-surface is reduced. The proposed algorithm can then be used to generate tool-paths for the relatively simple sub-surfaces based on the corresponding sub-point clouds.

Apart from grid fineness, the criterion for filtering out the digitizing points should also be considered. In the proposed algorithm, the distances of the projected points from the nodal points in the 2D triangular grid are used to determine whether a projected point (and its corresponding digitizing point) is filtered out or not. The 3D distances between the digitizing points and the nodal points are not taken into consideration. In case there are some tiny features between the nodal points, the features may be filtered out.

For a vertical (or nearly vertical) surface on the digitizing profile, most of the digitizing points on the vertical surface will be filtered out in nodal points reposition (Fig. 44) since the digitizing points are projected onto the *xy*-plane. The accuracy of the resultant machined surface may be affected. Also, sharp edges in the digitizing profile cannot be recognized in the proposed algorithm and thus cannot be retained in the machined surface.

To handle the vertical surfaces on the scanning object, some modifications on the algorithm must be carried out. If more than one scanning points are projected onto the same projected point (Fig. 45(a)), the corresponding scanning points have to be registered with this projected point. If this projected point is retained in the 2D triangular mesh (Fig. 45(b)), the procedure for generating the corresponding 3D triangular mesh has to be modified so that the vertical triangular facet can be formed. In Fig. 46(a), it can be seen that if only one scanning point ($\mathbf{P}_1$ or $\mathbf{P}_2$) is retained in the 3D triangular mesh, the vertical surface is replaced by a slanting facet. However, this will not be the case if another scanning point ($\mathbf{P}_1$ in this case) is retained (Fig. 46(b)). More works have to be done in order to incorporate this modification into the proposed algorithm for solving the mentioned problem.

In the proposed algorithm, a 3D triangular mesh has to be constructed in order to determine the local surface normal of each tool contact location. When comparing to traditional triangulation methods, the proposed procedure is much simpler and the mesh is constructed from a regular grid by calculating the distances between the projected points and the nodal points. The computational effort is much lighter than those of surface fitting techniques and the algorithm will not be complicated by the proposed triangular mesh construction procedure.

6. Conclusion

The proposed algorithm is used to provide another approach for generating CAM information from reverse engineering. Instead of the complicated surface fitting technologies, a simple triangular mesh construction method is proposed and this is used for determining the local surface normals of the digitizing points. A 3D biarc fitting technique is used for joining the tool center locations and a smooth surface can be machined out. As a result, sufficient information can be provided for the generation of a 5-axis tool-path.

Acknowledgments

The work described in this paper was substantially supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region (Project No.: B-Q601).

Appendix A. 3D biarc

A 2D biarc is constructed by two arcs such that the corresponding tangent vectors existed in coplanar condition. However, in 3D biarc, the planes of the two consecutive arc splines are now different, and there is a twist angle between them. The basic construction is shown in Fig. A1. In plane $\prod_1$, unit tangent vectors $\mathbf{t}_1$ and $\mathbf{t}_3$ form a plane. Similarly, unit tangent vector $\mathbf{t}_3$ and $\mathbf{t}_5$ form a plane in plane $\prod_2$.

In Fig. 26, $\mathbf{t}_1$ and $\mathbf{t}_3$ are unit tangent vectors of the given end points of the arc spline.

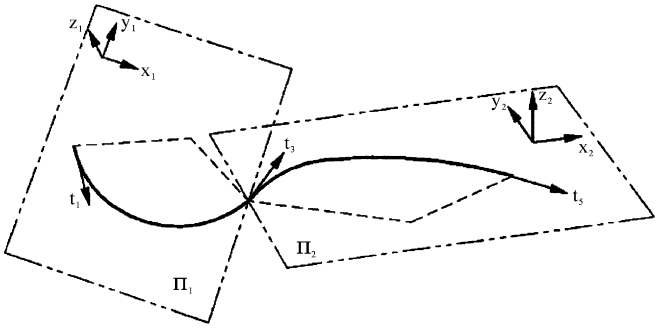

Fig. A1. Basic construction of a 3D biarc.

Let chord **AC** (**A**, **C** are given)

$$\mathbf{AC} = s, \tag{A.1}$$

For chord **AB** (**B**, $\mathbf{t}_2$ are not given),

$$\mathbf{AB} = s_1 = a(\mathbf{t}_1 + \mathbf{t}_2). \tag{A.2}$$

Also for chord

$$\mathbf{BC} = s_2 = b(\mathbf{t}_2 + \mathbf{t}_3). \tag{A.3}$$

Therefore,

$$\begin{aligned}\mathbf{AC} = s &= s_1 + s_2 = a(\mathbf{t}_1 + \mathbf{t}_2) + b(\mathbf{t}_2 + \mathbf{t}_3)\\ &= a\mathbf{t}_1 + (a+b)\mathbf{t}_2 + b\mathbf{t}_3,\end{aligned} \tag{A.4}$$

Rewrite (A.4) as

$$(a+b)\mathbf{t}_2 = \mathbf{s} - a\mathbf{t}_1 - b\mathbf{t}_3. \tag{A.5}$$

Squaring and simplifying

$$2ab(1 - \mathbf{t}_1 \cdot \mathbf{t}_3) + 2bs \cdot \mathbf{t}_3 = \mathbf{s}^2 - 2a\mathbf{s} \cdot \mathbf{t}_1 \tag{A.6}$$

then

$$b = \frac{\mathbf{s}^2 - 2a\mathbf{s} \cdot \mathbf{t}}{2a(1 - \mathbf{t}_1 \cdot \mathbf{t}_3) + 2\mathbf{s} \cdot \mathbf{t}_3},$$

$$a + b = a + \frac{\mathbf{s}^2 - 2a\mathbf{s} \cdot \mathbf{t}}{2a(1 - \mathbf{t}_1 \cdot \mathbf{t}_3) + 2\mathbf{s} \cdot \mathbf{t}_3}. \tag{A.7}$$

In order to find the minimum value of $(a+b)$, let $m = 1 - t_1 t_3$ and $f = a + b$

$$\begin{aligned}\frac{\mathrm{d}f}{\mathrm{d}a} &= 1 - \frac{(am + \mathbf{s} \cdot \mathbf{t}_3)(-\mathbf{s} \cdot \mathbf{t}_1) - \left((\mathbf{s}^2/2) - a\mathbf{s} \cdot \mathbf{t}_1\right)(m)}{(am + \mathbf{s} \cdot \mathbf{t}_3)^2}\\ &= 1 - \frac{\left(\mathbf{s}^2\mathbf{t}_1 \cdot \mathbf{t}_3 + (m\mathbf{s}^2/2)\right)}{(am + \mathbf{s} \cdot \mathbf{t}_3)^2}.\end{aligned}$$

Let

$$\frac{\mathrm{d}f}{\mathrm{d}a} = 0.$$

Then

$$a = \frac{\sqrt{\left(\mathbf{s}^2\mathbf{t}_1 \cdot \mathbf{t}_3 + (m\mathbf{s}^2/2)\right)} - \mathbf{s} \cdot \mathbf{t}_3}{m}. \tag{A.8}$$

The value of b can be calculated by the substitution of value a to Eq. (A.7).

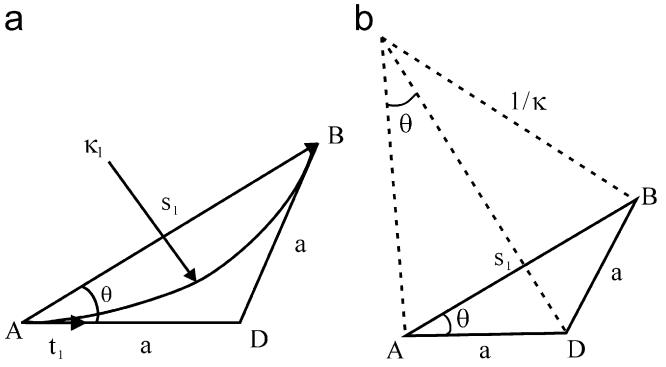

Fig. A2. Triangle **ADB**.

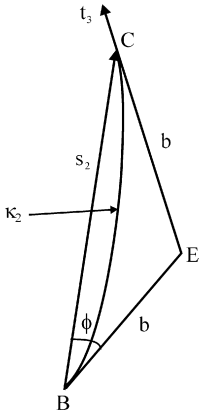

Fig. A3. Triangle **BEC**.

From triangle **ADB** in Fig. 26, the curvature κ_1 becomes

$$\kappa_1 = \left(\frac{2}{s_1}\right) \sin\,\theta. \tag{A.9}$$

From Fig. A2(a), Eq. (A.9) becomes

$$\kappa_1 = \frac{2\,\sin\,\theta}{2a\,\cos\,\theta} = \frac{\tan\,\theta}{a} = \frac{\|\mathbf{t}_1 \times \mathbf{s}_1\|}{a\mathbf{t}_1 \cdot \mathbf{s}_1}. \tag{A.10}$$

Using similar approach for triangle **BEC** (Fig. A3),

$$\kappa_2 = \left(\frac{2}{s_2}\right) \sin\,\phi = \frac{2\,\sin\,\phi}{2b\,\cos\,\phi} = \frac{\tan\,\phi}{b} \tag{A.11}$$

$$= \frac{\|\mathbf{t}_3 \times \mathbf{s}_2\|}{b\mathbf{t}_3 \cdot \mathbf{s}_2}. \tag{A.12}$$

Therefore, the radii of curvatures for the two arcs are

$$R_1 = \frac{1}{\kappa_1},$$

$$R_2 = \frac{1}{\kappa_2},$$

References

[1] Chui KL, Yu KM, Lee TC. Direct tool-path generation from massive point input. Proc Inst Mech Eng Part B: J Eng Manuf 2002;216(2):199–206.
[2] Li XS, Jerard RB. 5-Axis machining of sculptured surface with a flat-end cutter. Comput-Aided Des 1994;26(3):165–78.
[3] Bohez ELJ. Computer aided manufacturing of rotors for centrifugal compressors on a four axis machine. In: Proceeding of the 13th International Conference on Production Research, Jerusalem, Israel, 1995. p. 32–4.
[4] Bohez ELJ, Senadhera SDR, Ole K, Duflou JR, Tar T. A geometric modeling and five axis machining algorithm for centrifugal impellers. J Manuf Syst 1997;16(6):422–36.
[5] Elber G, Cohen E. A unified approach to verification in 5-axis freeform milling environments. Comput-Aided Des 1999;31(13):795–804.
[6] Lee AC, Chen DP, Lin CL. A CADCAM system from 3D coordinate measuring data. Int J Product Res 1990;28(12):2353–71.
[7] Bradley C, Vickers GW. Automated rapid prototyping utilizing laser scanning and free form machining. Ann CIRP 1992;41:437–40.
[8] Bradley C, Vickers GW. Freeform surface re-construction for machine rapid prototyping. Opt Eng 1993;32(9):2191–220.
[9] Kawabe S, Kimura F, Sata T. Generate the NC commands for sculptured surface machining from 3-coordinate measured data. Ann CIRP 1980;29(1):369–72.
[10] Choi BK, Shin HY, Yoon YI, Lee JW. Triangulation of scattered data in 3D space. Comput-Aided Des 1988;20(5):239–48.
[11] Park HJ, Kim KS. An adaptive method for smooth surface approximation to scattered 3D points. Comput-Aided Des 1995;27(12):929–39.
[12] Choi BK, Robert B. Sculptured surface machining. Dordrecht: Kluwer Academic Publishers; 1998 [Chapter 11, p. 224–6].
[13] Lin CA, Liu HT. Automatic generation of NC cutter path from massive data points. Comput-Aided Des 1998;30(1):77–90.
[14] Lai JY. Interference free cutter-path generation based on scanning data. Int J Adv Manuf Technol 1997;13(8):535–47.
[15] Ronald H. Computer graphics. Englewood Cliffs, NJ: Prentice-Hall Inc.; 1986 [Chapter 14].
[16] Sharrock TJ, Martin RR. Biarc in three dimensions, mathematics of surfaces II. Oxford: Oxford University Press; 1996.
[17] Henderson DW. Differential geometry—a geometric introduction. Englewood Cliffs, NJ: Prentice-Hall Inc.; 1998.