Direct NC path generation: from discrete points to continuous spline paths

Yu Liu Songtao Xia Xiaoping Qian [*]

Computational Design Laboratory

Department of Mechanical, Materials and Aerospace Engineering

Illinois Institute of Technology

Chicago, IL 60616

Abstract

Spline paths in NC machining are advantageous over linear and circular paths due to their smoothness and compact representation, thus are highly desirable in high-speed machining where frequent change of tool position and orientation may lead to inefficient machining, tool wear and chatter. This paper presents an approach for calculating spline NC paths directly from discrete points with controlled accuracy. Part geometry is represented by discrete points via an implicit point set surface (PSS). Cutter location (CL) points are generated directly from implicit part surfaces and interpolated by B-spline curves. A computing procedure for calculating maximum scallop height is given. The procedure is general and suitable for part surfaces in various surface representations provided that the closest distance from a point to the part surface can be calculated. Our results affirm that the proposed approach can produce high-quality B-spline NC paths directly from discrete points. The resulting spline paths make it possible for directly importing discrete points into CNC machines for high-speed machining.

1 INTRODUCTION

This paper presents an approach for calculating spline NC paths directly from discrete points by controlling the distribution and number of points interpolated by B-spline NC paths and the size of the side step between consecutive NC paths. Discrete point data has been widely used in product development and re-engineering due to the readily available three-dimensional data acquisition devices. Generating NC paths directly from discrete points avoids the time consuming and error prone tasks of surface fitting.

Among various representation of NC paths, spline NC paths are more advantageous over linear and circular paths due to their smoothness and compact representation, thus are highly desirable in high-speed machining where frequent change of tool position and

[*] All correspondence should be addressed to qian@iit.edu.

orientation may lead to inefficient machining, tool wear and chatter. Current CNC controllers can accept spline paths as input and generate motion commands on-line [1, 2]. Spline paths need fewer parameters for representing freeform curves than linear or circular paths to reach the same precision. Using spline paths alleviates memory burden of CNC controllers and decreases possible transmission errors between CAD/CAM and CNC systems. Spline paths also avoid the discontinuity between segments of linear or circular paths. The smoothness of spline paths is especially important for high-speed machining (HSM). The spindle in HSM has to slow down at C^0 points of tool paths, change its direction and accelerate to reach the desired maximum speed again. The deceleration/acceleration reduces the average machining speed, increases tool position error, and decreases motion repeatability.

Despite of the need for generating NC paths directly from discrete points and the advantages of spline paths, no reported work has been done for generating spline paths directly from discrete points. Existing approaches generate linear or circular paths instead of spline paths from discrete points [3, 4, 5, 6, 7, 8], or generate spline paths from continuous CAD surface models instead of discrete points [9, 10, 11, 12, 13].

In this paper, we present an approach that allows B-spline NC paths to be directly generated from discrete points. An ODE (ordinary differential equation) representation of NC paths is provided to define a nominal CC (cutter contact) path whose corresponding CL (cutter locations) path lies on a given tool driving plane. The ODE representation is amenable to computing of CC points and produces arc length parameterization of CL paths. Solutions of the ODE are numerically obtained by the fourth-order Runge-Kutta method and improved by our method to make sure the CL points strictly on the driving plane. The CL points on each path are interpolated by a B-spline curve. A compact set of interpolation points of each CL path is found by inserting or removing interpolation points according to deviations between the interpolation curve and the nominal CL path. The maximum side step between two consecutive CL paths is calculated by restricting the maximum scallop height formed by two CL paths to be below a given tolerance.

In our approach, the part surface is defined directly from discrete points via Amenta's moving least-squares (MLS) surface representation [14]. Our approach generates parallel paths represented by B-splines for finish machining. The MLS surface is an implicit representation, given in the implicit form $g(x, y, z) = 0$. In the past, NC path generation from implicit surfaces usually needs to go through a conversion process, which converts the implicit form to $z = f(x, y)$ or to a parametric form for the computation of tool paths [15], or go through a local Taylor series approximation in order to obtain explicit approximate representation of the implicit surface [9]. In contrast, our ODE representation of NC paths avoids such conversion by directly calculating path points from the implicit form. Note that our method is suitable for any point set surfaces

(PSSs) represented by implicit forms, not restricted to Amenta's MLS surface.

Compared with the method in [10] where B-spline path generation is based on parametric surfaces, our approach to calculating side steps is suitable not only for part surfaces represented by implicit surfaces, but also for all other surfaces provided that the closest distance from a point to the part surface can be calculated. Moreover, the method in [10] calculates the next CL path for each possible step size and the maximum scallop height formed by current CL path and each possible next CL path, while our approach only calculates possible step sizes at points on current CL paths. Thus our approach is potentially more efficient than the method in [10].

2 RELATED WORK

Substantial work has been done in the past concerning various issues of CNC machining, such as gouge avoidance, tool path generation and orientation identification, feedrate scheduling, and so on. The following review only covers related work of tool path generation from discrete points and B-spline NC path generation. A general review of recent development in CNC machining of freeform surfaces can be found in [16].

Lin and Liu [3] used a Z-map model to organize sequence points. Park and Chung [4] generated tool paths from similar sequence points to those in [3]. Park [5] generated iso-planar CL paths for triangular meshes. Triangular meshes can be obtained either from a parametric surface or a set of points. Feng and Teng [6] generated iso-planar piecewise linear CNC tool paths from discrete measured points. Input points are grouped by a projected cutter location net (CL-net). Chui et al. [7] obtained CL points by offsetting input points along normal vectors of 3D triangular mesh. CL paths are fitted from these CL points by a set of 3D arc splines. However, none of above methods discussed the specification of path parameters to control the accuracy of paths according to user specified tolerances. One possible reason would be the difficulty of evaluating the accuracy of tool paths when differential geometric properties such as normal vectors and curvatures are estimated directly from discrete points.

New surface representations of discrete points have appeared recently. Point-set surfaces (PSSs) are continuous surfaces defined directly from point sets. Since its original inception [17, 18, 14], the PSS and its many variants have been widely used in various graphics, visualization, geometric modeling and engineering applications (work in [19] and [8] for example). PSS has been applied in generating tool paths directly from discrete points [8]. However, no spline paths were generated in [8] and the path generation was based on cutter contact points.

On the other hand, B-spline path has been explored for some time. Lartigue et al. [10] generated B-spline paths from parametric part surfaces with the control of the maximum scallop height. Jin et al. [11] proposed a local cubic B-spline fitting algorithm with real time look-ahead scheme for consecutive micro-line blocks interpolation.

Erkorkmaz et al. [12] presented a quintic spline trajectory generation algorithm that produces continuous position, velocity, and acceleration profiles. Vijayaraghavan et al. [13] took a subdivision method of curves as an interpolation method to reduce trajectory generation time and increase feed rates. However, none of the above methods deals with the path generation from discrete points.

3 B-spline form of CL Path generation from discrete points

In this section, we first provide an ODE (ordinary differential equation) representation of the NC paths that defines a nominal CC (cutter contact) path whose corresponding CL (cutter locations) path lies on a given tool driving plane. We numerically solve the ODE by the fourth-order Runge-Kutta method and present a method that improves the numerical accuracy and makes CL points strictly on the driving plane. We then examine the deviation between the B-spline CL path and the nominal CL path defined according to the ODE model and obtain a compact set of interpolation points of the CL path by inserting or removing interpolation points.

3.1 Ordinary differential equation representation of CC and CL paths

In this paper, the part surface from discrete points is represented by an implicit MLS surface. For detailed description of how discrete points via MLS ~~surface~~ can be used in machining and rapid prototyping, please refer to our earlier work [8, 19]. In order to compute the points on the intersection curve between a given driving plane and the CL surface, we develop an ODE representation of corresponding CC path.

Given a point set $P = \{\boldsymbol{p}_i, i = 1, \ldots, n\}$, let the nominal part surface be represented by the PSS surface $g(\boldsymbol{x}) = 0$, where $\boldsymbol{x} = (x, y, z)^T$. The offset surface $\widehat{g}(\widehat{\boldsymbol{x}}) = 0$ of $g(\boldsymbol{x}) = 0$ consists of points

$$\widehat{\boldsymbol{x}} = \boldsymbol{x} + d\boldsymbol{n}(\boldsymbol{x}), \tag{1}$$

where d is a signed offset distance and $\boldsymbol{n}(\boldsymbol{x})$ denotes the unit normal

$$\boldsymbol{n}(\boldsymbol{x}) = \frac{(g_x, g_y, g_z)^T}{\left\|(g_x, g_y, g_z)^T\right\|}, \tag{2}$$

where $(g_x, g_y, g_z) = (\partial g(\boldsymbol{x})/\partial x, \partial g(\boldsymbol{x})/\partial y, \partial g(\boldsymbol{x})/\partial z)$. The direction of the normal is consistent such that $\boldsymbol{n}(\boldsymbol{x})$ points outside the surface $g(\boldsymbol{x}) = 0$.

The CL surface is the offset surface from $g(\boldsymbol{x}) = 0$ with offset distance $d = R$, where R is the cutter radius. The CL path is defined as a series of intersection curves between the CL surface and tool driving planes [10]. Let the current tool driving plane be defined by

$$\alpha x + \beta y + \gamma z + \delta = 0, \tag{3}$$

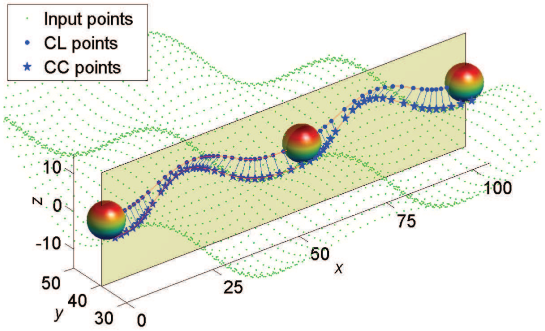


Figure 1: CC points and CL points defined by Eq. (4) along a path of a wavy surface.

where $(\alpha, \beta, \gamma)^T$ is the unit normal vector of the driving plane. A point $\widehat{\boldsymbol{x}}$ on the CL path should satisfy both Eq. (1) and Eq. (3). Then we have the following implicit representation for the CC points

$$\begin{cases} g(\boldsymbol{x}) = 0 \\ \left\langle (\alpha, \beta, \gamma)^T, \boldsymbol{x} + R\boldsymbol{n}(\boldsymbol{x}) \right\rangle + \delta = 0 \end{cases} \tag{4}$$

where $\langle\rangle$ denotes the dot product, the first equation means that the CC points lie on the part surface, and the second equation means that the offset points of the CC points by the cutter radius R lie on the driving plane. Note that Eq. (4) and Eq. (1) together define the nominal CL path on the given driving plane.

Fig. 1 shows CC points and CL points defined by Eq. (4) and Eq. (1) along a path of a wavy surface, where the green points, the blue pentacles, the blue points denote the input data, CC points, CL points respectively, the three spheres center at corresponding CL points and with the radius equal to the cutter radius. In the figure, each CL point is connected with corresponding CC point by a line segment. Eq. (4) possesses the following interesting properties:

1. Part surfaces are represented by implicit surfaces instead of parametric surfaces as in previous work [10].

2. The unknowns of Eq. (4) are coordinates of CC points. The CL points can be obtained immediately from these CC points by Eq. (1).

3. CL points determined by Eq. (4) lie on the driving plane, while in previous work [9] and [8] CC points instead of CL points lie on the driving plane. Since machine tools are driven along the CL path plane (the driving plane), our approach thus avoids potential jerk and ensures dynamically smooth paths.

Let the curve corresponding to the CC path be represented by $\boldsymbol{c}(t) = (x(t), y(t), z(t))^T$. From Eq. (1), we have

$$\frac{d\widehat{\boldsymbol{c}}(\boldsymbol{c}(t))}{dt} = (1 + R\Delta\boldsymbol{n}(\boldsymbol{x}))\frac{d\boldsymbol{c}}{dt}. \tag{5}$$

where $d\boldsymbol{c}/dt = (dx/dt, dy/dt, dz/dt)^T$,

$$\Delta \boldsymbol{n}(\boldsymbol{x}) = \begin{bmatrix} \frac{dn_x}{dx} & \frac{dn_x}{dy} & \frac{dn_x}{dz} \\ \frac{dn_y}{dx} & \frac{dn_y}{dy} & \frac{dn_y}{dz} \\ \frac{dn_z}{dx} & \frac{dn_z}{dy} & \frac{dn_z}{dz} \end{bmatrix} \tag{6}$$

Differentiating Eq. (4) with respect to the parameter t yields

$$\begin{cases} g_x \frac{dx}{dt} + g_y \frac{dy}{dt} + g_z \frac{dz}{dt} = 0 \\ \left\langle (\alpha, \beta, \gamma)^T, (1 + R\Delta \boldsymbol{n}(\boldsymbol{x})) \frac{d\boldsymbol{c}}{dt} \right\rangle = 0 \end{cases} \tag{7}$$

The solution to Eq. (7) is given by

$$\begin{cases} \frac{dx}{dt} = \xi (bg_z - cg_y) \\ \frac{dy}{dt} = \xi (cg_x - ag_z) \\ \frac{dz}{dt} = \xi (ag_y - bg_x) \end{cases} \tag{8}$$

where $(a, b, c) = (\alpha, \beta, \gamma) + R(\alpha, \beta, \gamma)\Delta \boldsymbol{n}(\boldsymbol{c})$, ξ is an arbitrary non-zero factor. Eq. (8) constitute the ODE representation of the CC path. The ODE representation of the CL path is obtained by substituting Eq. (8) into Eq. (5). The non-zero factor ξ can be chosen to provide the arc length parameterization of the CL path by letting $\|d\hat{\boldsymbol{c}}/dt\|^2 = 1$. Then from Eq. (5) and Eq. (8),

$$\xi = \pm \frac{1}{\left\| (1 + R\Delta \boldsymbol{n}(\boldsymbol{c})) (bg_z - cg_y, cg_x - ag_z, ag_y - bg_x)^T \right\|}. \tag{9}$$

3.2 Numerical procedure for computation of CL points

A sequence of CC points $(x_k, y_k, z_k)^T$, $k = 0, 1, \ldots$ can be computed by integrating the nonlinear differential Eq. (8) using, for example, the fourth-order Runge-Kutta method. CL points are obtained by substituting these CC points into Eq. 1. However, these CC points may not strictly satisfy Eq. (4) due to limited accuracy of numerically integrating Eq. (8). A method is introduced to improve the accuracy of the resultant CL points and make the CL points be strictly on the driving plane.

Without loss of generality, we assume driving planes are perpendicular to the y-direction of the part coordinate system and the current driving plane is defined by $y = y_c$, where y_c is a constant. Let the forward direction be parallel to the x-direction. Assume the coordinates of a CL point near $y = y_c$ are given and the x coordinate of the CL point is x_c.

Our method restricts the unknown CL point on the intersection line between planes $x = x_c$ and $y = y_c$. Let the intersection line be denoted by $L(x_c, y_c)$. Then the unknown CL point is the intersection point between $L(x_c, y_c)$ and the CL surface (shown in Fig. 2). And the problem to find the unknown CL point can be formulated by

$$\min_{z_c} \left[\left| \mathrm{Dis}\left((x_c, y_c, z_c)^T, S \right) \right| - R \right]^2, \tag{10}$$

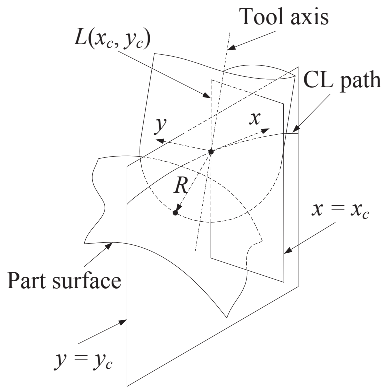


Figure 2: A point on a CL path.

where S denotes the part surface, $(x_c, y_c, z_c)^T$ is a point on the line $L(x_c, y_c)$, $\mathrm{Dis}(\ , S)$ is the closest distance from a point to the part surface.

The above problem contains the subproblem to find the closest point of $(x_c, y_c, z_c)^T$ on the implicit part surface $g(\boldsymbol{x}) = 0$. Since $g(\boldsymbol{x}) = 0$ is defined on point set $P = \{\boldsymbol{p}_i, i = 1, \ldots, n\}$, the initial closest point is specified by the closest point from $(x_c, y_c, z_c)^T$ to points of P. The initial closest point can be found efficiently by a data structure such as k-d tree.

Let the initial closest point be denoted by $\boldsymbol{p}$. The implicit surface $g(\boldsymbol{x}) = 0$ near $\boldsymbol{p}$ is parameterized locally by a plane passing through $\boldsymbol{p}$ and perpendicular to $\boldsymbol{n}(\boldsymbol{p})$, where $\boldsymbol{n}(\boldsymbol{p})$ is calculated by Eq. (2). Let $\boldsymbol{p}(u_0, v_0)$ be a point on the plane and has a parameter (u_0, v_0). And assume $\boldsymbol{u}$ and $\boldsymbol{v}$ are two orthogonal unit vectors in the plane. Then

$$\boldsymbol{p}(u_0, v_0) = \boldsymbol{p} + u_0\boldsymbol{u} + v_0\boldsymbol{v}.$$

The corresponding point of $\boldsymbol{p}(u_0, v_0)$ on the part surface is specified by the same parameter as $\boldsymbol{p}(u_0, v_0)$ and is denoted by $\boldsymbol{x}(u_0, v_0)$. The point $\boldsymbol{x}(u_0, v_0)$ is obtained from $\boldsymbol{p}(u_0, v_0)$ by the MLS projection process. The problem to find closest points on the part surface is formulated as

$$\begin{aligned} &\min_{u,v} \left[(x_c, y_c, z_c)^T - (x(u,v), y(u,v), z(u,v))^T\right]^2 \\ &\text{s.t.} \ \ g(x(u,v), y(u,v), z(u,v)) = 0 \end{aligned} \tag{11}$$

where $(x(u,v), y(u,v), z(u,v))^T$ are coordinates of a point with parameter (u,v) on the part surface.

3.3 B-spline curve interpolation of CL paths

For a fixed-degree B-spline path (cubic in this paper), the number of B-spline segments can be determined from the number of B-spline interpolation points. In order to make the resulting B-spline representation compact, the number of CL points to be interpolated needs to be controlled. The initial number of interpolation points is determined by the increment t_{inc} of the parameter t of Eq. (8).The t_{inc} is specified to be equal to the value of the cutter radius R in our experiments, which makes initial interpolation points be evenly distributed.

Let $\widehat{\boldsymbol{x}}_k$, $k = 1, 2, \ldots$ be initial CL points which are strictly on the corresponding driving plane after applying our method introduced in Section 3.2. Let $\widehat{\boldsymbol{c}}\,(u)$ be the interpolation curve of these initial CL points. Then $\widehat{\boldsymbol{c}}\,(u)$ is also on the driving plane. Assume $\widehat{\boldsymbol{c}}\,(u_c)$ is a point on $\widehat{\boldsymbol{c}}\,(u)$ and $\boldsymbol{c}\,(u_c)$ is the closest point of $\widehat{\boldsymbol{c}}\,(u_c)$ on the part surface $g\,(\boldsymbol{x}) = 0$. According to the nominal CL path defined by Eq. (1) and Eq. (4), the distance between $\widehat{\boldsymbol{c}}\,(u_c)$ and $\boldsymbol{c}\,(u_c)$ should be equal to the cutter radius R. Then the deviation of interpolation curve $\widehat{\boldsymbol{c}}\,(u)$ at point $\widehat{\boldsymbol{c}}\,(u_c)$ from the nominal CL path needs to satisfy

$$\left| \left\| \widehat{\boldsymbol{c}}\,(u_c) - \boldsymbol{c}\,(u_c) \right\| - R \right| \leq \overline{h}_{inter}. \tag{12}$$

where $\|\|$ denotes Euclidean norm, and $\overline{h}_{inter}$ is the given tolerance.

The accuracy of curve $\widehat{\boldsymbol{c}}\,(u)$ is usually checked at each midspan of the two successive parameter values [10], i.e. $u_c = (u_k + u_{k+1})\,/2$ in Eq. (12), $k = 1, 2, \ldots$, where u_k and u_{k+1} are the curve parameters of interpolation points $\widehat{\boldsymbol{x}}_k$ and $\widehat{\boldsymbol{x}}_{k+1}$ respectively. However, the accuracy of the curve segment between $\widehat{\boldsymbol{c}}\,(u_k)$ and $\widehat{\boldsymbol{c}}\,(u_{k+1})$ is not guaranteed by only examining the midspan point. In order to get an interpolation curve bounded by the tolerance $\overline{h}_{inter}$, a point with the maximum deviation on the curve segment between $\widehat{\boldsymbol{c}}\,(u_k)$ and $\widehat{\boldsymbol{c}}\,(u_{k+1})$ is explicitly obtained by solving

$$\max_{u_k \leq u \leq u_{k+1}} \left| \left\| \widehat{\boldsymbol{c}}\,(u) - \boldsymbol{c}\,(u) \right\| - R \right| \tag{13}$$

Assume the solution of the above problem is $u_k^{\max}$ and the maximum deviation is

$$h_{inter}\left(u_k^{\max}\right) = \left| \left\| \widehat{\boldsymbol{c}}\left(u_k^{\max}\right) - \boldsymbol{c}\left(u_k^{\max}\right) \right\| - R \right|. \tag{14}$$

The curve segment between $\widehat{\boldsymbol{c}}\,(u_k)$ and $\widehat{\boldsymbol{c}}\,(u_{k+1})$ is bounded by the tolerance $\overline{h}_{inter}$ if and only if

$$h_{inter}\left(u_k^{\max}\right) \leq \overline{h}_{inter}. \tag{15}$$

The interpolation points are inserted in an iterative process. In each iteration, let the curve segment with the highest deviation be the curve segment between $\widehat{\boldsymbol{c}}\,(u_k)$ and $\widehat{\boldsymbol{c}}\,(u_{k+1})$. If the maximum deviation of the curve segment satisfies Eq. (15), the process is ended. Otherwise, a new CL point is calculated by taking $\widehat{\boldsymbol{c}}\,((u_k + u_{k+1})\,/2)$ as an initial point for the method introduced in Section 3.2. Then old CL points and the new CL point are interpolated by a planar cubic B-spline curve again. The above process is repeated until there is no new CL point that needs to be inserted.

Although the above process can produce an error bounded interpolation curve by inserting more interpolation points, it may lead to excessive interpolation points. If all the curve segments of the interpolation curve still satisfy Eq. (15) after removing an interpolation point, the interpolation point is deemed unnecessary and can be removed.

Then the interpolation points can be removed in an iterative process. In each iteration, whether an interpolation point can be removed is examined by the maximum deviation of the curve segments of the interpolation curve after removing the interpolation point. Such maximum deviation is calculated for each interpolation point. The interpolation point with the minimum value of such maximum deviations is selected to be removed. The above process is repeated until none of points can be removed while retaining the given tolerance.

4 Path interval generation

In this section, a method is presented to find the step size between two consecutive CL paths such that the maximum scallop height along a scallop curve coincides with the prescribed tolerance. A point on a scallop curve is first calculated from a point of a known CL path such that the scallop height of the point on the scallop curve is equal to the prescribed tolerance, where the known CL path is a B-spline curve obtained by the method introduced in Section 3. Then a point on the consecutive unknown CL path is obtained according to the relationship between the scallop curve and the CL path. A step size is calculated from the two corresponding CL points on the known and unknown CL paths. The step size between the known and unknown CL paths are obtained by finding the minimum of the step size calculated at each point of the known CL path.

Let A and B be two points on two consecutive CL paths $\widehat{\boldsymbol{c}}^A(t)$ and $\widehat{\boldsymbol{c}}^B(\sigma)$ respectively (shown in Fig. 3), where the CL paths are represented by parameter curves. The side step l_{CL} is defined as

$$l_{CL} = \langle B - A,\ \boldsymbol{a}_{side} \rangle, \tag{16}$$

where $\boldsymbol{a}_{side}$ is the unit side direction.

The scallop curve is an intersection curve between two pipe surfaces [20]

$$P^A(t,\alpha) = \widehat{\boldsymbol{c}}^A(t) + R\left[\cos\alpha \boldsymbol{b}^A(t) + \sin\alpha \boldsymbol{n}^A(t)\right], \tag{17}$$

$$P^B(\sigma,\beta) = \widehat{\boldsymbol{c}}^B(\sigma) + R\left[\cos\beta \boldsymbol{b}^B(\sigma) + \sin\beta \boldsymbol{n}^B(\sigma)\right], \tag{18}$$

where $\boldsymbol{n}$ and $\boldsymbol{b}$ are the normal and binormal vectors of corresponding CL path respectively, α (or β) is the angle from the vector $\boldsymbol{b}^A$ (or $\boldsymbol{b}^B$) to the vector $P^A - \boldsymbol{c}^A$ (or $P^B - \boldsymbol{c}^B$). Since $\widehat{\boldsymbol{c}}^A(t)$ and $\widehat{\boldsymbol{c}}^B(\sigma)$ are on driving planes, the binormal vectors are perpendicular to driving planes.

As shown in Fig. 3, C is a point on the scallop curve and the distance from C to its closest point $\widetilde{C}$ on the part surface is equal to the scallop height h. The point C can be calculated from Eq. (17) if the h is specified. Assume $h = \overline{h}_{side}$, where $\overline{h}_{side}$ is the maximum scallop height. Then the point C can be obtained by solving

$$\min_{\alpha} F_1(t,\alpha) = \left[\left|\mathrm{Dis}\left(P^A(t,\alpha), S\right)\right| - \overline{h}_{side}\right]^2, \tag{19}$$

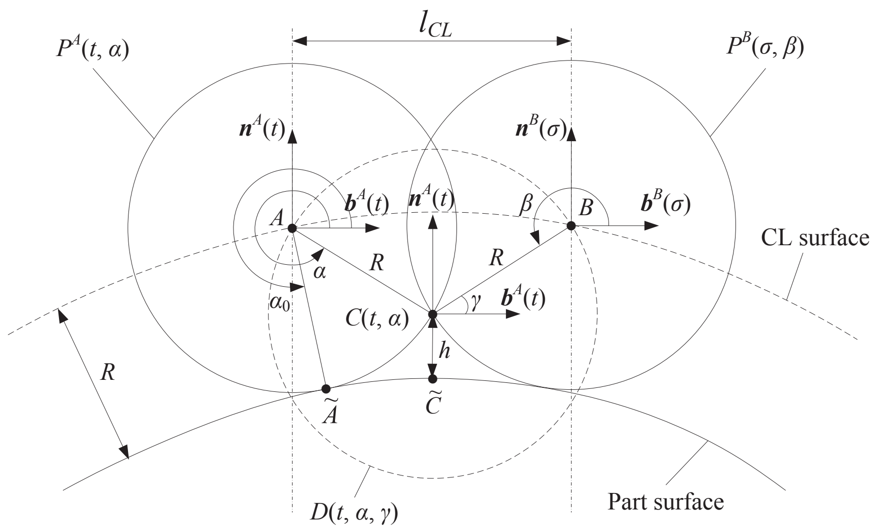


Figure 3: A and B are two ~~point~~ on two consecutive CL paths. The intersection curve of two pipe ~~surface~~ $P^A(t,\alpha)$ and $P^B(\sigma,\beta)$ is a scallop curve. $C(t,\alpha)$ is a point on the scallop curve. The scallop height at $C(t,\alpha)$ is h.

where $\text{Dis}\left(P^A(t,\alpha), S\right)$ is the distance from point $P^A(t,\alpha)$ to the part surface, t is the parameter of the point A on the CL path.

Assume $\widetilde{A}$ is the closest point of A to the part surface. Then $\widetilde{A}$ is the CC point of A. Assume the angle from the vector $\boldsymbol{b}^A(t)$ to the vector $\widetilde{A} - A$ is α_0. Then the α in Eq. (19) is restricted by $\alpha_0 \le \alpha \le \alpha_0 + pi/2$.

The side step l_{CL} and the next tool path $\boldsymbol{c}^B(\sigma)$ are both unknown now. Given the point C, point B should lie on the sphere centered at C and with the cutter radius R. The location of point B is not unique and all of them produce the same scallop hight with respect to point A. The location of point B which corresponds to the maximum side step is chosen since bigger side step means higher machining efficiency. Then we let the chord connecting A and B be on the plane spanned by $\boldsymbol{b}^A$ and $\boldsymbol{n}^A$ at point A, i.e. point B is on the great circle (shown in Fig. 3)

$$D(t,\alpha,\gamma) = C(t,\alpha) + R\left[\cos\gamma \boldsymbol{b}^A(t) + \sin\gamma \boldsymbol{n}^A(t)\right], \tag{20}$$

The point B should lie on the CL surface too. Then the point B can be obtained by solving

$$\min_{\gamma} F_2(t,\alpha,\gamma) = \left[\left|\text{Dis}\left(D(t,\alpha,\gamma), S\right)\right| - R\right]^2. \tag{21}$$

The range of γ is specified to exclude the point A because A is also on the circle $D(t,\alpha,\gamma)$.

The side step corresponding to the point A can be calculated by Eq. (16) now. A side step is calculated similarly for each point of the current CL path $\widehat{\boldsymbol{c}}^A(t)$. The side step between the consecutive known and unknown CL paths is selected as the minimum of all such side steps, i.e.

$$\min_{t} F_3\left(t, \alpha, \gamma\right) = \left\langle D\left(t, \alpha, \gamma\right) - \widehat{\boldsymbol{c}}^{A}\left(t\right),\ \boldsymbol{a}_{side} \right\rangle, \tag{22}$$

where α and γ are obtained by solving Eq. (19) and Eq. (21). Substituting Eq. (19) and Eq. (21) into Eq. (22), the problem to calculating the side step becomes

$$\min_{t} F_3\left(t, \operatorname*{argmin}_{\alpha} F1, \operatorname*{argmin}_{\gamma}\left(F_2, \operatorname*{argmin}_{\alpha} F_1\right)\right). \tag{23}$$

The position of the tool driving plane of the unknown CL path $\widehat{\boldsymbol{c}}^{B}\left(\sigma\right)$ is obtained by moving the current tool driving plane along the side direction by the value of the side step. Given the start and end positions of the tool driving plane, all the CL paths can be calculated. The start and end positions of the tool driving plane are determined by points on the CL surface. These points are sorted along the side direction in ascending order. The start and end tool driving planes pass through the first and last points respectively.

5 EXPERIMENTS

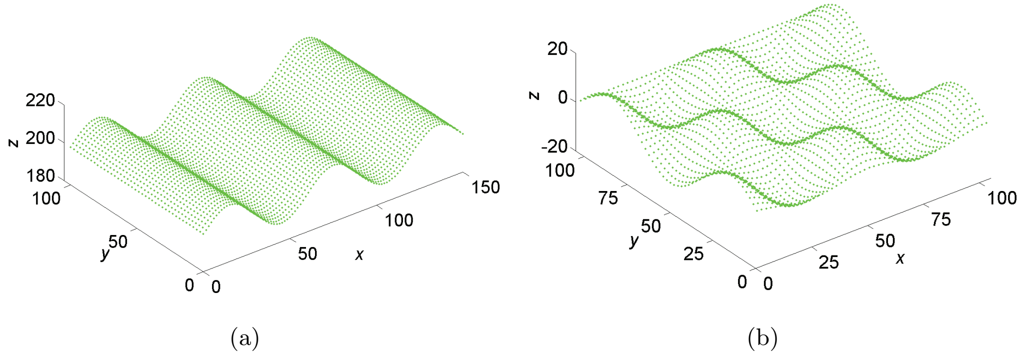
The above formulations for calculating B-spline paths' interpolation points and side steps have been implemented in Matlab. The optimization routines from Matlab are used. All the experiments are carried out on a computer with Intel Pentium 3GHz CPU and 2 GB of RAM. Experimental results on both synthetic and real scanned data are presented in this section. The unit of the length used in this paper is millimeter.

The synthetic data used in our experiments is sampled from two nominal surfaces. The first surface is a sinusoidal surface obtained by sweeping a sine curve along a line. The second surface is defined by $z = a\cos\left(bx\right)\sin\left(cy\right)$, where a, b, and c are constants and are specified by 10, 0.12, 0.09 respectively. Fig. 4 shows points sampled from the two surfaces. Table 1 gives characteristics of points sampled from the two surfaces, including the number of points and domain range of the data.

Table 1: ~~Characteristics of synthetic data~~

		Size		
~~Examples~~	Number of points	x	y	z
Sinusoidal surface	4,000	150.00	100.00	19.99
Wavy surface	2,500	102.63	102.63	19.98

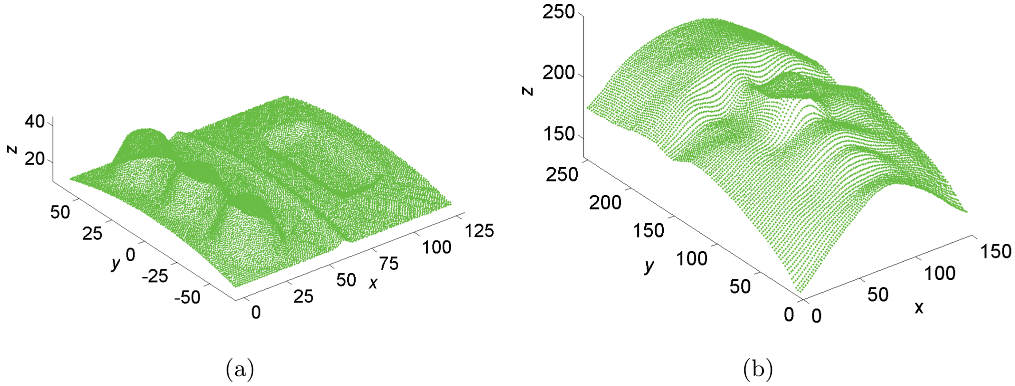
The application of our approach on real data obtained from 3D ~~scanners (such as~~ Minolta VIVID910) has also been conducted. Fig. 5 show a compound surface and a face~~-~~like surface used in the experiments. Table 2 gives characteritics of the two ~~surface~~s, where μ and σ denote the mean and standard deviation of the Gaussian noise respectively.



(a)

(b)

Figure 4: Two sets of synthetic data ~~used in the experiments~~: (a) a sinusoidal surface; (b) a wavy surface.



(a)

(b)

Figure 5: Two sets of real data ~~used in the experiments~~: (a) a compound surface; (b) a face like surface.

Table 2: ~~Characteristics of real data~~

		Size			Noise	
~~Examples~~	Number of points	x	y	z	μ	σ
Compound surface	20,302	127.00	127.16	26.71	0	0.1
Face~~-~~like surface	10,000	149.06	247.50	105.40	0	0.2

Table 3 gives ~~conditions~~ used to generate CL paths for the sinusoidal surface, the wavy surface, the compound surface, and the face like surface, where R is the cutter radius, ε_{nbor} defines the range of neighborhood of a driving plane, $\overline{h}_{inter}$ is the tolerance of B-spline interpolation, $\overline{h}_{side}$ is the maximum scallop height, $\overline{h}_{MLS}$ is the bandwidth of MLS projections.

Curvatures of the sinusoidal surface, the wavy surface, the compound surface, and the face like surface are calculated at points on PSSs corresponding to input points by the method presented in [21]. The maximum absolute values of principal curvatures of the sinusoidal surface, the wavy surface, the compound surface, and the face like surface are 0.102, 0.129, 0.311, and 0.207 respectively. Then all the cutter radius R given in Table 3 are smaller than the maximum limit of the cutter size determined by the the minimum curvature radius of corresponding part surface.

Table 4 gives some results obtained from the sinusoidal surface, the wavy surface, the compound surface, and the face~~-~~like surface, where n_{path} denotes the number of CL paths, $\overline{n}_{initial}$ denotes the average number of initial interpolation points of a CL path, $\overline{n}_{final}$ denotes the average number of final interpolation points of a CL path, $\overline{t}_{path}$ denotes the average computation time of a CL path, $\overline{t}_{side}$ denotes the average computation time of the side step from a CL path, the unit of $\overline{t}_{path}$ and $\overline{t}_{side}$ is second. The $\overline{t}_{path}$ is the sum of the time to calculate the initial interpolation points and the time to obtain a compact set of interpolation points under the given tolerance. Obtaining a compact set of interpolation points is much more time consuming than calculating the initial interpolation points in our experiment. The $\overline{t}_{side}$ consists of times to solve three ~~similar~~ optimization problems given by Eq. (19), Eq. (21), and Eq. (22). ~~The time costs of solving Eq. (19) and Eq. (21) are similar, which are included in the time cost of solving Eq. (22) because Eq. (19) and Eq. (21) need to be solved for each possible value of t of Eq. (22)~~.

5.1 Distribution of interpolation points and path intervals

Fig. 6 - Fig. 9 show ~~results~~ obtained from the sinusoidal surface, the wavy surface, the compound surface, and the face~~-~~like surface respectively. Fig. 6 (a) - Fig. 9 (a) show the isometric view of CL paths and positions of some CL paths that are plotted in 2D view in Fig. 6 (b) - Fig. 9 (b). Note that only half of CL paths of the compound surface and the face~~-~~like surface are plotted for display clarity.

Table 3: ~~Conditions~~ used to generate CL paths

~~Examples~~	R	ε_{nbor}	$\overline{h}_{inter}$	$\overline{h}_{side}$	$\overline{h}_{MLS}$
Sinusoidal surface	4.5	5	0.005	0.03	4.0
Wavy surface	4.5	5	0.005	0.03	4.0
Compound surface	3	2	0.005	0.03	2.0
Face like surface	4.5	5	0.005	0.03	4.0

Table 4: Results about CL paths and side steps

~~Examples~~	n_{path}	$\overline{n}_{initial}$	$\overline{n}_{final}$	$\overline{t}_{path}$	$\overline{t}_{side}$
Sinusoidal surface	101	44	65	362.54	27.22
Wavy surface	137	27	45	259.62	24.52
Compound surface	220	48	85	312.87	33.95
Face~~-~~like surface	300	40	58	148.08	39.55

As Fig. 6 (b) - Fig. 9 (b) show, the distribution of interpolated CL points is curvature adaptive, i.e. there are more points in regions with higher absolute values of curvatures. Fig. 6 (c), (d) - Fig. 9 (c), (d) confirm the above conclusion, where regions with high absolute values of normal curvatures and high densities of CL points are marked by ellipses.

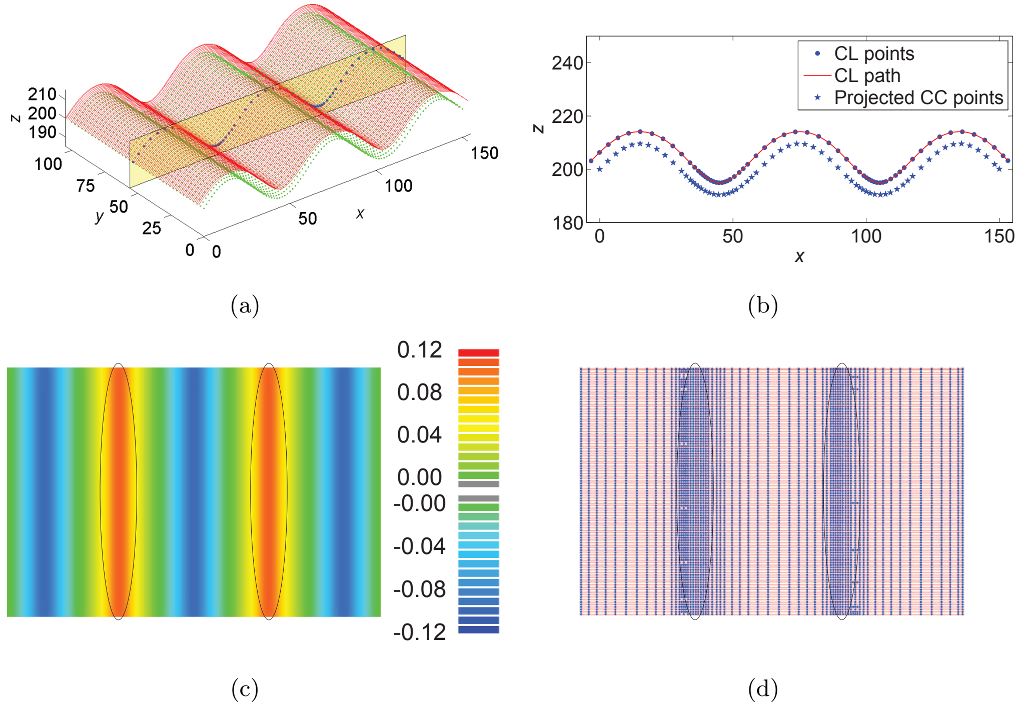
The normal curvature along the side direction are similar for each path of the sinusoidal surface. As expected, side steps of CL paths of the sinusoidal surface are also similar. Although the normal curvature along the side direction changes obviously for each path of the wavy surface, the range of the normal curvature is similar for each path of the wavy surface (shown in Fig. 7 (e)). Side steps of CL paths of the wavy surface are similar too (shown in Fig. 7 (f)).

Fig. 8 (e) and Fig. 9 (e) show normal curvatures along side directions of the compound surface and the face~~-~~like surface. Fig. 8 (f) and Fig. 9 (f) show the top view of CL paths of the compound surface and the face like surface. From these figures, we observe that

1. Smaller side step is produced in regions ~~with~~ higher normal curvature along the side direction.

2. Concave regions produce smaller side steps than convex regions even the normal curvatures along the side direction have the same absolute values.

The second observation is related to the definition of the scallop height. As Fig. 3 shows, the scallop height at a point on the scallop curve is the distance from the point to the part surface. If the side step l_{CL} is a constant, the scallop height to a concave region is bigger than that to a convex region. Moreover, a smaller scallop height needs a



(a)

(b)

(c)

(d)

Figure 6: CL paths and CL points generated from a sinusoidal surface: (a) isometric view of input points and CL paths; (b) the CL path, interpolated CL points, and corresponding CC points; (c) normal curvatures along the forward direction; (d) top view of CL points and CL paths.

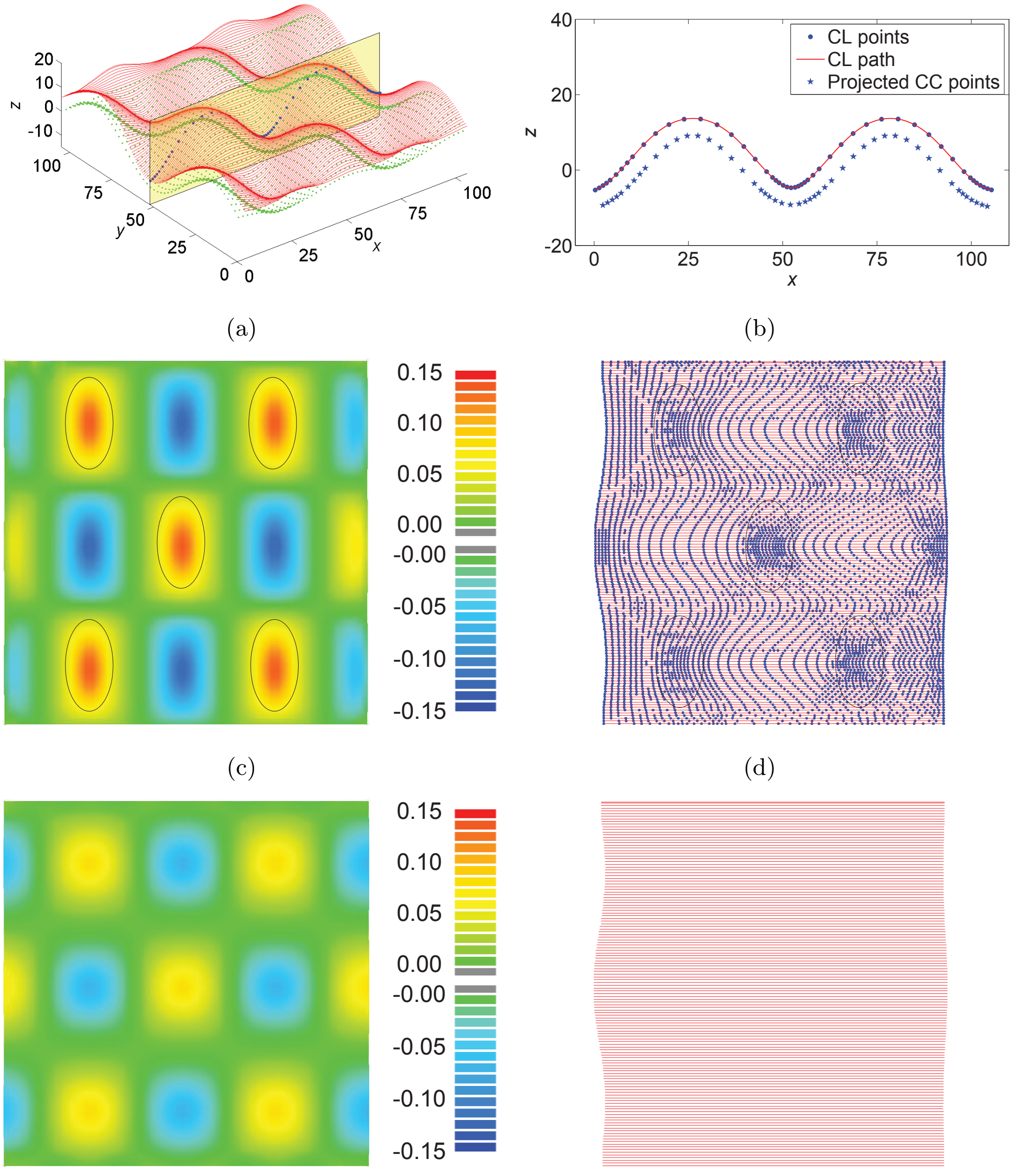
smaller side step. Then concave regions ~~produce~~ smaller side steps than convex regions if the scallop height is equal to the maximum scallop height $\overline{h}_{side}$.

5.2 Deviations of CL paths

Fig. 10 - Fig. 13 show deviations of CL paths~~, where~~ the horizontal line at 0.005 indicates the interpolation tolerance $\overline{h}_{inter} = 0.005$, each dot indicates the maximum deviation along a CL path. ~~The determination of midspan points has been adopted by previous work [10]. However, as Fig. 10 (a) - Fig. 13 (a) show, enforcing the tolerance only at midspan points may produce CL paths out of given tolerance. As Fig. 10 (b) - Fig. 13 (b) show, the CL paths obtained by our method are bounded by the given tolerance.~~

5.3 Number~~s~~ of interpolation points

Table 5 gives the total number of interpolation points for each example, where n_{In} denotes the number of interpolation points obtained by the process of only inserting interpolation points, n_{InRe} denotes the number of interpolation points obtained by processes of both inserting and removing interpolation points. As Table 5 indicates, unnecessary interpolation points are reduced by the process of removing interpolation points.



(a)

(b)

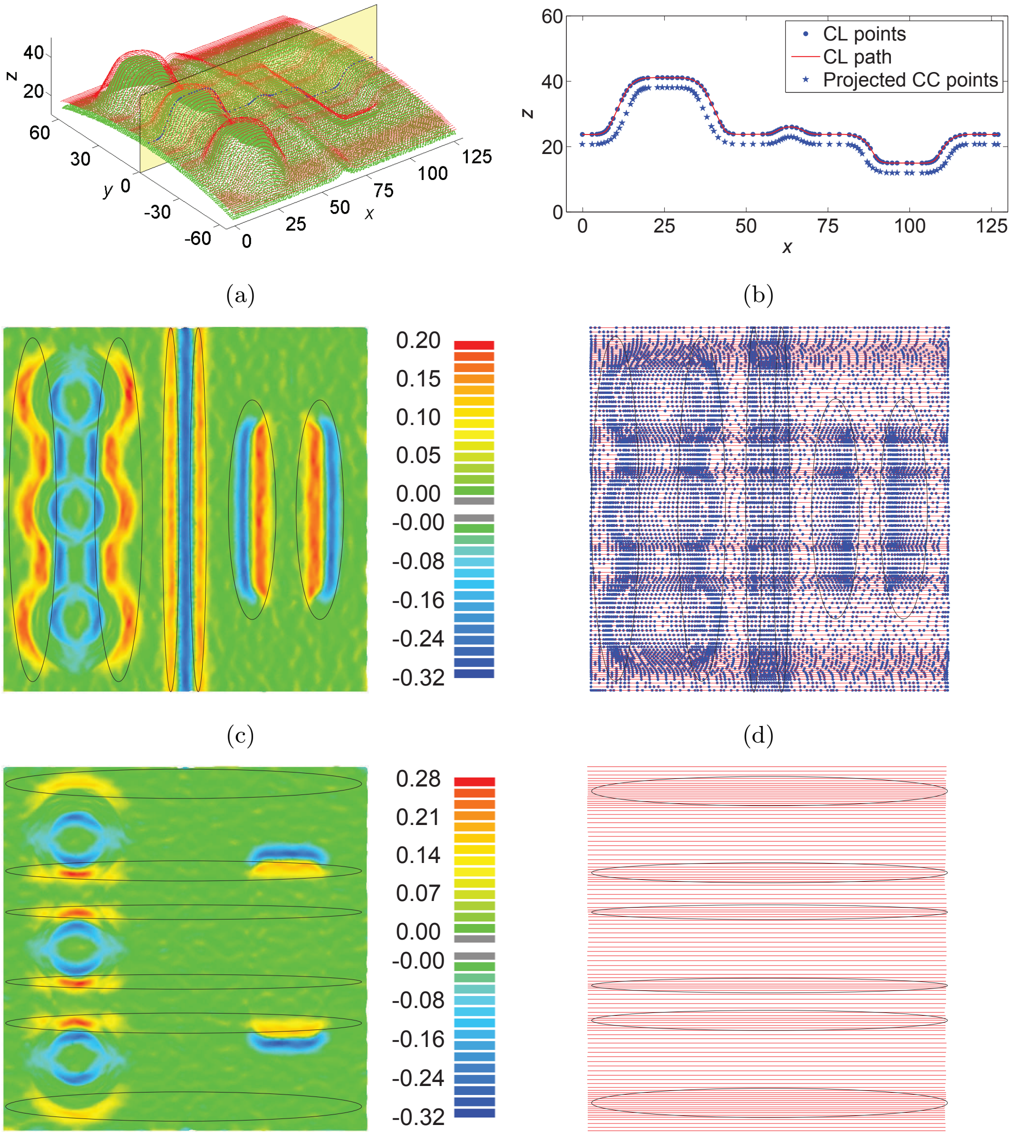
(c)

(d)

(e)

(f)

Figure 7: CL paths and CL points generated from a wavy surface: (a) isometric view of input points and CL paths; (b) the CL path, interpolated CL points, and corresponding CC points; (c) normal curvatures along the forward direction; (d) top view of CL points and CL paths; (e) normal curvatures along the side direction; (f) top view of CL paths.



(a)

(b)

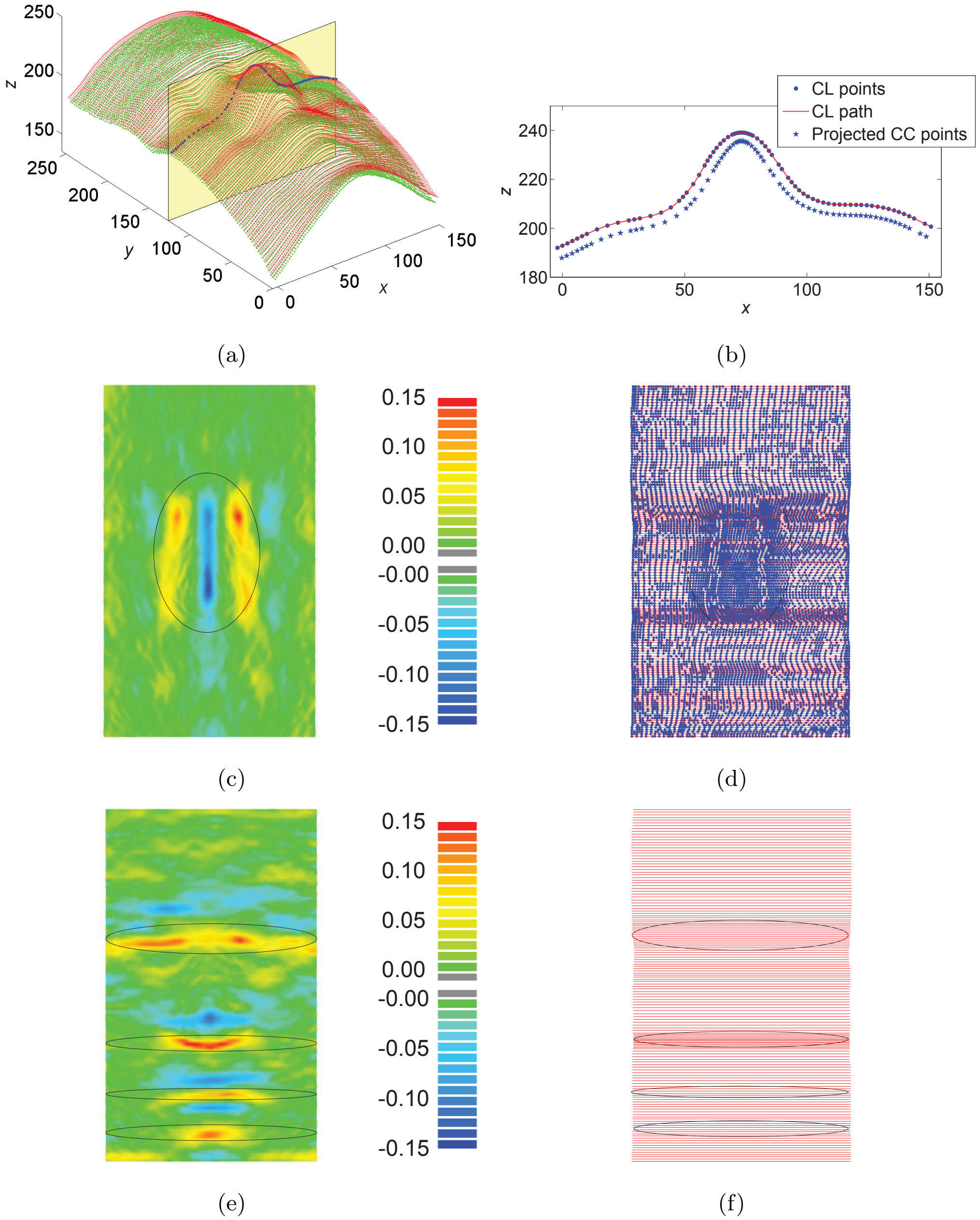
(c)

(d)

(e)

(f)

Figure 8: CL paths and CL points generated from a compound surface: (a) isometric view of input points and CL paths; (b) the CL path, interpolated CL points, and corresponding CC points; (c) normal curvatures along the forward direction; (d) top view of CL points and CL paths; (e) normal curvatures along the side direction; (f) top view of CL paths.



(a)

(b)

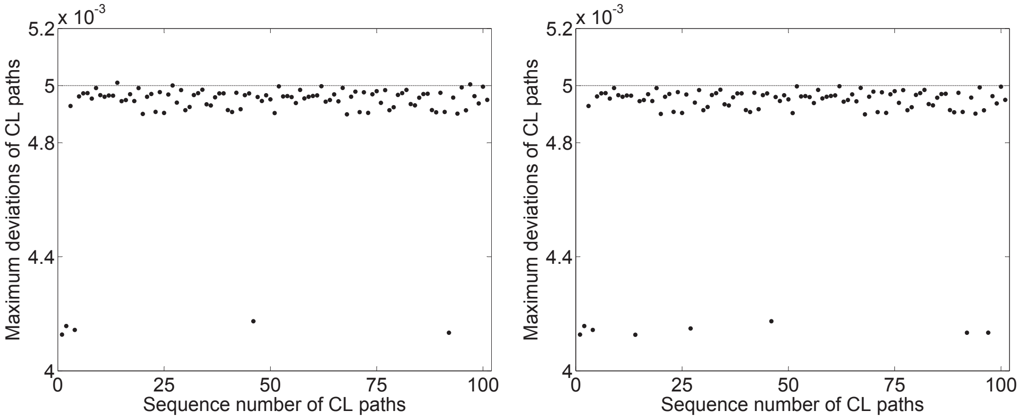
(c)

(d)

(e)

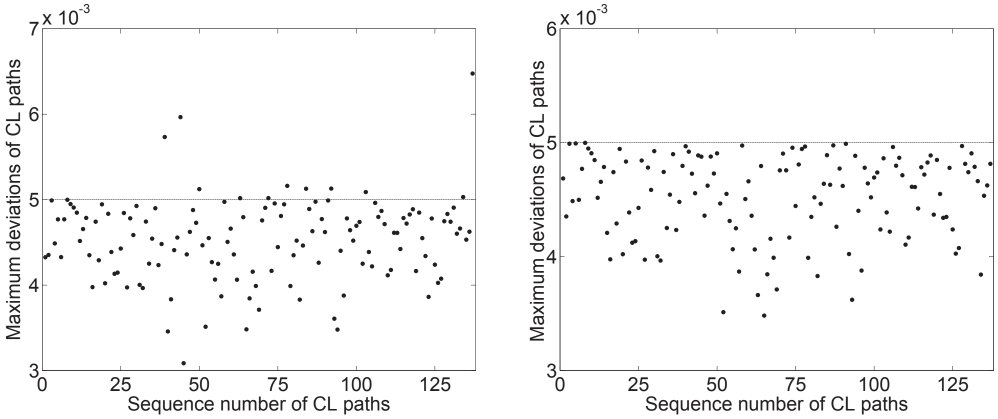
(f)

Figure 9: CL paths and CL points generated from a ~~face like~~ surface: (a) isometric view of input points and CL paths; (b) the CL path, interpolated CL points, and corresponding CC points; (c) normal curvatures along the forward direction; (d) top view of CL points and CL paths; (e) normal curvatures along the side direction; (f) top view of CL paths.



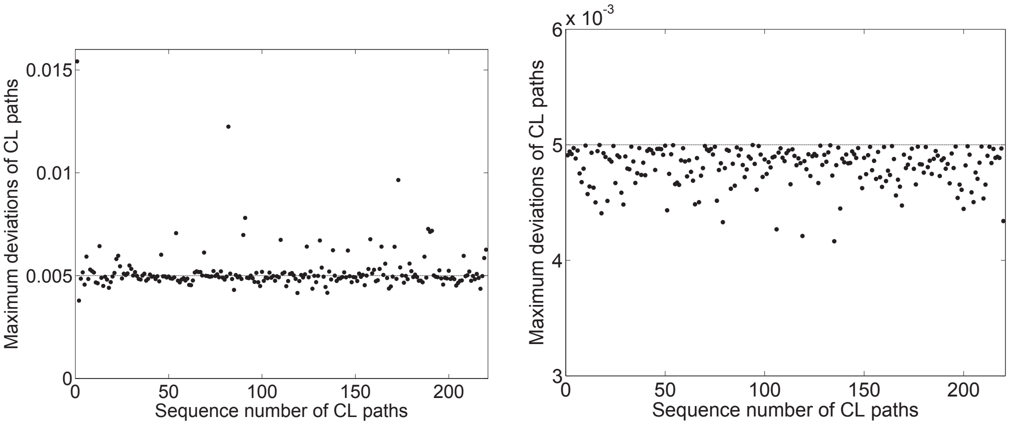
(a) (b)

Figure 10: ~~Deviations of CL paths generated from the sinusoidal surface: (a) CL paths obtained by enforcing the tolerance only at midspan points; (b) CL paths obtained by our method.~~



(a) (b)

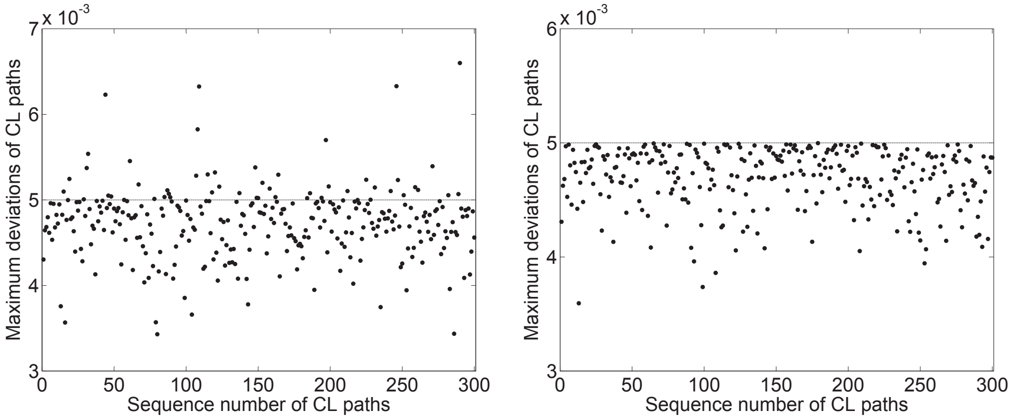
Figure 11: Deviations of CL paths generated from the wavy surface: (a) CL paths obtained by enforcing the tolerance only at midspan points; (b) CL paths obtained by our method.



(a)

(b)

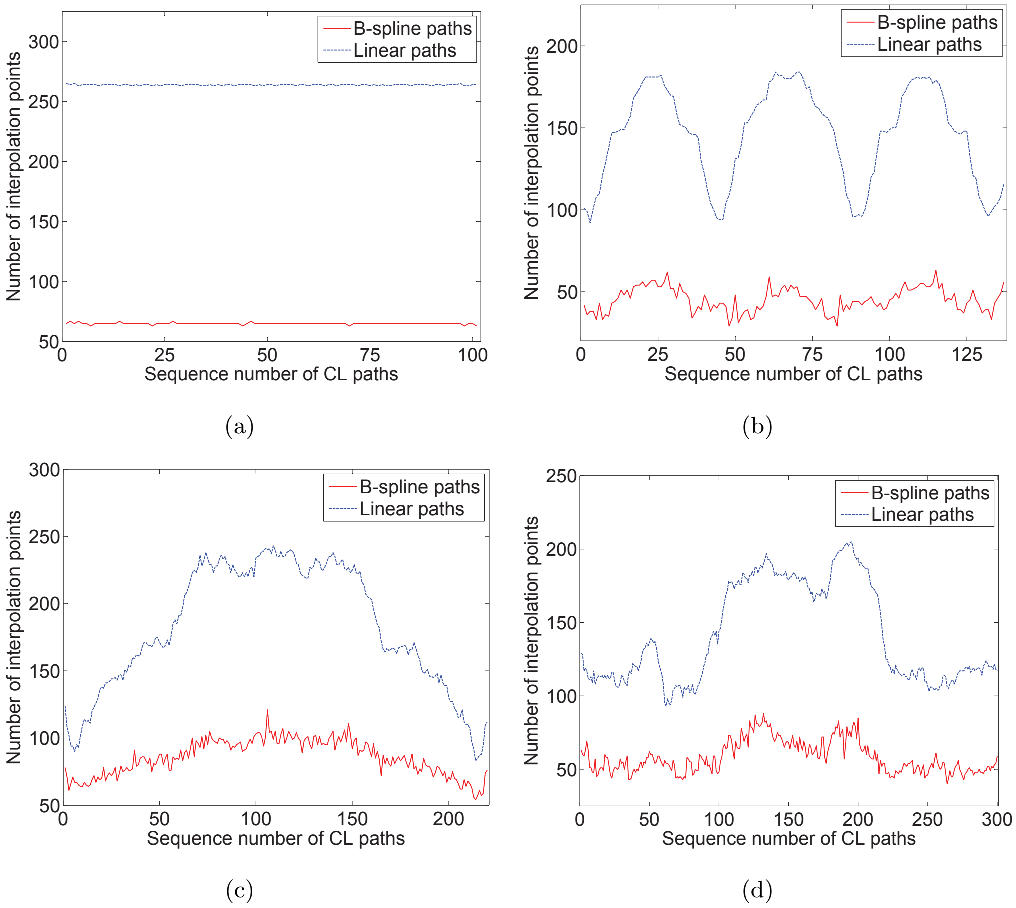
Figure 12: Deviations of CL paths generated from the compound surface: (a) CL paths obtained by enforcing the tolerance only at midspan points; (b) CL paths obtained by our method.



(a)

(b)

Figure 13: Deviations of CL paths generated from the face like surface: (a) CL paths obtained by enforcing the tolerance only at midspan points; (b) CL paths obtained by our method.



(a)

(b)

(c)

(d)

Figure 14: ~~Bounded by~~ the same interpolation tolerance, the number of interpolation points ~~of~~ B-spline CL paths ~~and~~ the number of interpolation points ~~of~~ linear CL paths: ~~Deviations of CL paths generated from a face like surface:~~ (a) the sinusoidal surface; (b) the wavy surface; (c) the compound surface; (d) the face like surface.

As Fig. 14 shows, the number of interpolation points ~~of~~ B-spline CL paths is smaller than the number of interpolation points ~~of~~ linear CL paths, where all the paths are bounded by the same tolerance. As Table 5 shows, the total number of interpolation points of our B-spline paths is less than 25% of that of linear paths for the sinusoidal surface, less than 31% of that of linear paths for the wavy surface, less than 48% of that of linear paths for the compound surface, and less than 42% of that of linear paths for the face like surface.

6 CONCLUSION

This paper presents an approach for generating B-spline CL paths directly from discrete points. The method for calculating the single tool path is suitable for any PSSs represented by implicit forms. An ODE representation of NC paths is introduced to produce CL points on the driving plane directly from the implicit MLS surface. By the processes of inserting and removing interpolation points, a compact set of interpolation

Table 5: Number of interpolation points ~~of~~ CL paths

		B-spline paths	
Examples	Linear paths	n_{In}	n_{InRe}
Sinusoidal surface	26,637	6,581	6,563
Wavy surface	19,993	6,270	6,185
Compound surface	39,951	20,276	18,785
Face like surface	42,424	18,564	17,467

points is obtained for each CL path such that the number of interpolation points is reduced and the B-spline path interpolated satisfies a given tolerance. Our approach is more accurate than conventional methods since it checks the point with the maximum deviation in each curve segment of CL paths instead of the point at the midspan of each curve segment. The method for calculating the side step between consecutive paths is suitable for part surfaces in different forms provided that the closest distance from a point to the part surface can be calculated. The step size is determined such that the maximum scallop height along a scallop curve coincides with a given tolerance.

Computational experiments demonstrate that the approach produces high-quality B-spline paths from a variety of noisy point cloud data. The resultant paths are curvature adaptive and satisfy specified tolerances.

Our future work would be to experimentally verify this method on high-speed CNC machines and to ~~run the formulations of the method on parallel processes~~ to improve the efficiency.

ACKNOWLEDGMENTS

We gratefully acknowledge the financial support from NSF grants (#0900597 and #1030347).

References

[1] Tikhon, M., Ko, T., Lee, S., and Sool Kim, H., 2004. “NURBS interpolator for constant material removal rate in open NC machine tools”. *International Journal of Machine Tools and Manufacture,* ***44***(2-3), pp. 237–245.

[2] Tsai, M., Nien, H., and Yau, H., 2008. “Development of an integrated look-ahead dynamics-based NURBS interpolator for high precision machinery”. *Computer-Aided Design,* ***40***(5), pp. 554–566.

[3] Lin, A., and Liu, H., 1998. “Automatic generation of NC cutter path from massive data points”. *Computer-Aided Design,* ***30***(1), pp. 77–90.

[4] Park, S., and Chung, Y., 2003. “Tool-path generation from measured data”. *Computer-Aided Design,* ***35***(5), pp. 467–475.

[5] Park, S., 2004. “Sculptured surface machining using triangular mesh slicing”. *Computer-Aided Design,* ***36***(3), pp. 279–288.

[6] Feng, H., and Teng, Z., 2005. “Iso-planar piecewise linear NC tool path generation from discrete measured data points”. *Computer-Aided Design,* ***37***(1), pp. 55–64.

[7] Chui, K., Chiu, W., and Yu, K., 2008. “Direct 5-axis tool-path generation from point cloud input using 3D biarc fitting”. *Robotics and Computer-Integrated Manufacturing,* ***24***(2), pp. 270–286.

[8] Zhang, D., Yang, P., and Qian, X., 2009. “Adaptive NC path generation from massive point data with bounded error”. *Journal of Manufacturing Science and Engineering,* ***131***, p. 011001.

[9] Tam, H., Xu, H., and Zhou, Z., 2002. “Iso-planar interpolation for the machining of implicit surfaces”. *Computer-Aided Design,* ***34***(2), pp. 125–136.

[10] Lartigue, C., Thiebaut, F., and Maekawa, T., 2001. “CNC tool path in terms of B-spline curves”. *Computer-Aided Design,* ***33***(4), pp. 307–319.

[11] Jin, Y., Wang, Y., Feng, J., and Yang, J. “Research on consecutive micro-line interpolation algorithm with local cubic B-spline fitting for high speed machining”. In Mechatronics and Automation (ICMA), 2010 International Conference on, IEEE, pp. 1675–1680.

[12] Erkorkmaz, K., and Altintas, Y., 2001. “High speed CNC system design. Part I: jerk limited trajectory generation and quintic spline interpolation”. *International Journal of machine tools and manufacture,* ***41***(9), pp. 1323–1345.

[13] Vijayaraghavan, A., Sodemann, A., Hoover, A., Rhett Mayor, J., and Dornfeld, D., 2010. “Trajectory generation in high-speed, high-precision micromilling using subdivision curves”. *International Journal of Machine Tools and Manufacture,* ***50***(4), pp. 394–403.

[14] Amenta, N., and Kil, Y., 2004. “Defining point-set surfaces”. *ACM Transactions on Graphics,* ***23***(3), pp. 264–270.

[15] Choi, B., and Jerard, R., 1998. *Sculptured surface machining: theory and applications.* Kluwer Academic.

[16] Lasemi, A., D., X., and Gu, P., 2010. “Recent development in CNC machining of freeform surfaces: A state-of-art review”. *Computer-Aided Design,* ***42***(7), pp. 641–654.

[17] Levin, D., 2003. “Mesh-independent surface interpolation”. *Geometric Modeling for Scientific Visualization,* ***3***, pp. 37–49.

[18] Alexa, M., Behr, J., Cohen-Or, D., Fleishman, S., Levin, D., and Silva, C., 2003. “Computing and rendering point set surfaces”. *IEEE Transactions on Visualization and Computer Graphics,* ***9***(1), pp. 3–15.

[19] Yang, P., and Qian, X., 2008. “Adaptive slicing of moving least squares surfaces: toward direct manufacturing of point set surfaces”. *Journal of Computing and Information Science in Engineering,* ***8***, p. 031003.

[20] Sarma, R., and Dutta, D., 1997. “The geometry and generation of NC tool paths”. *Journal of Mechanical Design,* ***119***, pp. 253–258.

[21] Yang, P., and Qian, X., 2007. “Direct computing of surface curvatures for point-set surfaces”. In Eurographics symposium on point-based graphics, pp. 29–36.

VITAE

Yu Liu is a postdoctoral of Huazhong University of Science and Technology and a research associate ~~of~~ Illinois Institute of Technology. He received an undergraduate degree in mechatronic engineering from Tianjin University, China in 2000 and a doctor’s degree in the mechatronic engineering ~~of~~ Huazhong University of Science and Technology, China in 2008. His research interests include reverse engineering, CAD/CAM, and digital manufacturing.

Songtao Xia is a PhD student at the department of Mechanical, Materials and Aerospace Engineering at IIT. He received his BS degree in mechanical engineering from Huazhong University of Science and Technology, China in 2008 and MS degree from The George Washington University in 2010. His current research interests focus on geometric processing of massive point-cloud data.

Xiaoping Qian is an associate professor in the Department of Mechanical, Materials and Aerospace Engineering at IIT. He received his Ph.D. in mechanical engineering from the University of Michigan, Ann Arbor in 2001. His research interests lie in the general area of computational design.