Iso-level tool path planning for free-form surfaces

Qiang Zou[a,c], Juyong Zhang[a,*], Bailin Deng[b], Jibin Zhao[c]

[a] *School of Mathematical Sciences, University of Science and Technology of China, Anhui, 230026, China;*
[b] *Computer Graphics and Geometry Laboratory, École Polytechnique Fédérale de Lausanne, Lausanne, CH-1015, Switzerland;*
[c] *Shenyang Institute of Automation, Chinese Academy of Sciences, Liaoning, 110016, China.*

Abstract

The aim of tool path planning is to maximize the efficiency against some given precision criteria. In practice, scallop height should be kept constant to avoid unnecessary cutting, while the tool path should be smooth enough to maintain a high feed rate. However, iso-scallop and smoothness often conflict with each other. Existing methods smooth iso-scallop paths one-by-one, which makes the final tool path far from being globally optimal. In this paper, a new framework of tool path optimization is proposed, where different objectives can be considered simultaneously, to plan globally optimal tool path with respect to iso-scallop and smoothness. The idea is to view a family of iso-level curves of a scalar function over the surface as the tool path so that desired tool path can be generated by finding the function that minimizes certain energy functional. The energy functionals for planning iso-scallop, smoothness, and optimal tool path are respectively derived, and the path topology is studied too. Experimental results are given to show effectiveness of the proposed methods.

Keywords: iso-level tool path, globally optimal, PDE, differential geometry

*Corresponding author. Tel.: +86-551-63600673.
Email addresses: john.qiangzou@gmail.com (Qiang Zou), juyong@ustc.edu.cn (Juyong Zhang), bailin.deng@epfl.ch (Bailin Deng), jbzhao@sia.cn (Jibin Zhao)

1. Introduction

The terminology "tool path" refers to a specified trajectory along which machine tools move their ends (i.e., cutter and table) to form desired surfaces. The automatic generation of such trajectories are of central importance of modern CAD/CAM systems. Automatic tool path generation mainly considers two criteria: precision and efficiency. Precision refers to the error of approximating a surface with a family of curves, and approximating a curve with a family of segments or arcs. Efficiency concerns the time of machining along the tool path. The aim of tool path planning is to maximize the efficiency under the given precision criteria. In this paper, we propose to generate a globally optimal tool path which takes these two criteria into consideration together.

1.1. Related works

For a given precision tolerance (i.e., the scallop height and chord deviation), the tool path is always supposed to be as smooth and short as possible. In this paper, the smoothness of the tool path is measured by its curvature in the 3D space. If the tool path is smooth enough, there is less repeated acceleration/deceleration, which makes it possible to maintain a high feed rate. Meanwhile, the shorter the tool path is, the less time the machining takes. Theoretically, tool paths following the direction of maximum machining strip width are the shortest in total length, since they maximize material removal. But such strategy often leads to irregular tool paths which are neither direction/contour parallel nor spiral, as shown in [1, 2]. Therefore, in practice, a weaker condition that the tool path has no unnecessary (also called redundant) cutting is adopted. To achieve this purpose, the scallop height should be kept constant along the path. Hence, tool paths with iso-scallop and smooth properties are preferable.

Last decade has seen a great deal of literature on tool path planning for free-form surfaces, such as iso-parametric method [3–5], iso-planar method [6–8], iso-scallop method [9–15], iso-phote method [16] and C-space method [17], to name a few. Surveys of much more work about tool path planning research can be found in [18, 19]. Since we aim at optimal tool paths with respect to iso-scallop and smoothness, we put special interest in the iso-scallop method, which means the height of the points at the scallop curves remains as high as a given tolerance so that the tool path has no unnecessary cutting. Conventionally, constant scallop height is obtained by varying the

offset magnitude along each path. A mathematical method for generating iso-scallop tool paths with such strategy was first proposed by Suresh et al. [9]. Afterwards, methods to improve the computing efficiency [10, 13] and accuracy [11, 12, 15] were proposed. In 2007, Kim [14] reformulated the iso-scallop tool path as geodesic parallel curves on the design surface by defining a new Riemannian metric.

Despite the non-redundance property, tool paths of constant scallop height tend to have sharp corners, as illustrated in Fig. 1. It means that smoothness and iso-scallop requirements often conflict with each other. And the tradeoff between them is a major concern in tool path planning. A widely adopted solution is postprocessing: first a new path with constant scallop is generated by varying the offset magnitude along current path, then it is smoothed by replacing its corners with circular arcs [20, 21]. An alternative way is to employ the level set method to offset the paths while keeping them smooth [22]. Similar to the image segmentation method by Paragios et al. [23], we can add a curvature term into the evolution equation so that points of higher curvature are offset less while those of lower curvature are offset more. However, on one hand, for a path subject to desired precision, the modification would introduce error. On the other hand, if the path is offset less than the desired tolerance to avoid such error, then hardly can we choose a proper offset magnitude since we usually do not have an overall picture of the tool path. For example, in [22], because a curvature item is introduced into the normal velocity, it is still unknown how to choose a proper evolution step that determines the distance between neighboring paths. In fact, such local modification is in general unable to gain a globally optimal tool path, since it cannot take the ungenerated paths into account when operating on (or optimizing) one path. All previous works with offset based methods generate tool paths one-by-one, and thus inherit the drawback of non-optimal results.

There also exist some efforts to generate smooth tool paths without considering the overlap between neighboring machining strips (i.e., the iso-scallop condition). Generally, such methods are based on the Laplacian. For example, Bieterman and Sandstrom [24] proposed a Laplacian based contour parallel tool path generation method by selecting the level sets of the harmonic function over a pocket as the tool path. It remains an unsolved problem to choose the level sets with this method, namely there is no formula for path interval calculation so far. Similarly, Chuang and Yang [25] combined the Laplacian method and iso-parametric method to generate tool paths for

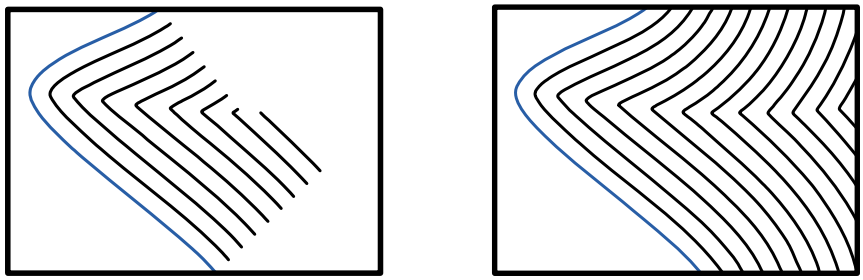
pockets with complex topology, i.e., complex boundaries and islands. However, the smoothness of the tool path cannot be guaranteed through Laplacian energy as small Laplacian value does not necessarily mean small curvature of the level set curves. And solving a Laplace equation over a surface can only generate a unique and uncontrollable scalar function (scaling has no impact on the shape of tool path). Another drawback of the Laplacian approach is that the severe overlapping between machining strips for neighbor paths, especially for paths near the boundary, which results in much redundant machining.

1.2. Our approach

In this paper, we aim to plan optimal tool path regarding iso-scallop and smoothness. We propose a framework that is able to obtain a globally optimal tool path by considering several objectives together. The tool path is represented as a family of level set curves from a scalar function defined on the surface, and our method computes an optimal scalar function by solving a single optimization problem, instead of generating the curves one-by-one. We refer to the level sets as *iso-level curves*, and the proposed tool path planning method as *iso-level method*, in order to be consistent with other terminologies in the literature such as iso-parametric, iso-planar, iso-scallop and iso-phote.

As the tool path is represented by the iso-level curves of the optimized scalar function, the desired properties of the tool path are encoded into the properties of the scalar function. In this work, we give the details of how to control the scalar function so that the desired tool path, e.g., iso-scallop tool path, can be generated. We first propose an iso-scallop condition for the target function, which shapes two neighboring iso-level curves to be iso-scallop. Then we propose a smoothness objective. Finally we combine them together to form the objective energy functional so that its minimizer corresponds to an optimal tool path with respect to iso-scallop and smoothness. To the best of our knowledge, this paper is the first work where these formulas are given, through which interval between iso-level curves and their smoothness can be controlled globally. The minimizer of the iso-scallop objective can not only be exploited to plan tool path of constant scallop, but also has an interesting machining meaning that the level increment of two neighbor iso-level curves equals to the square root of scallop height generated by them. In addition, the optimal scalar function can be reused to generated tool path of different scallop height tolerances.

Compared with existing tool path generation methods, our proposed method solves the tool path planing problem in a global optimization way. Besides, the proposed iso-level tool path planning method can free us from the tedious post-processing step for self-intersection and disjunction, which will be demonstrated in Section 2.4. Since the scalar function is defined all over the surface, the model is completely covered by the iso-level curves so that there are no regions which are not machined, as opposed to the offset based methods (illustrated in Fig. 1). Our optimization framework can also be easily extended to include other objectives, such as tool wear, machine kinematics and dynamics.



(a) (b)

Figure 1: Iso-scallop tool path. (a) offset based iso-scallop tool path; (b) scale function based iso-scallop tool path.

The remainder of this paper is organized as follows: Section 2 describes the optimization models for the iso-level method, including iso-scallop tool paths (Section 2.1), smooth tool paths (Section 2.2), optimal tool paths (Section 2.3), followed by a discussion on the resulting tool path topology (Section 2.4). In Section 3, we present the numerical solution to the optimization problems. Section 4 summarizes the overall procedures for planning iso-level tool paths. Section 5 shows some experimental results. Finally, we conclude the whole paper in Section 6.

2. Optimal iso-level tool path

Consider a surface S embedded in $\mathbb{R}^3$ and a scalar function $\varphi : S \to \mathbb{R}$ defined over it. The curves on S which correspond to a set of values $\{l_i\}_{i=1}^n$ bounded by the range of the scalar function are selected as tool path for the surface. There are two problems to consider when generating tool

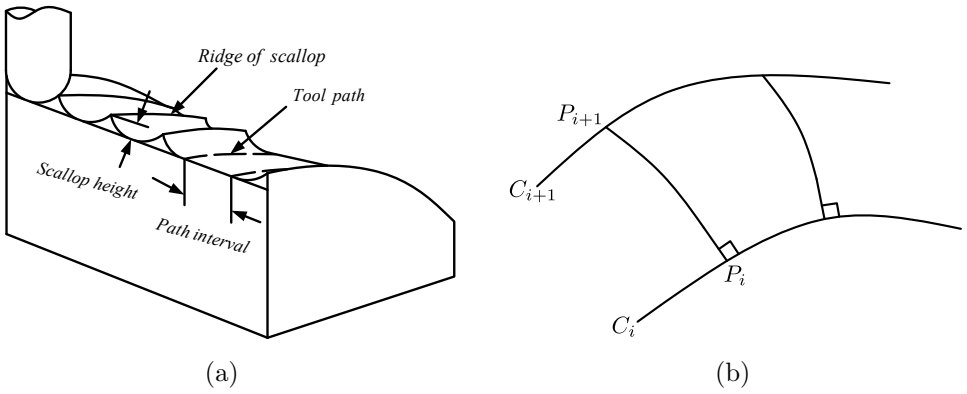


Figure 2: Illustrations of path parameters. (a) scallop height and path interval; (b) path interval.

path with this strategy: the design of φ and the mathematical method for determining $\{l_i\}$. In this section, we describe our solution to these problems, and demonstrate how to plan iso-level tool paths.

2.1. Iso-scallop tool path generation

In general, a tool path is discretized as a family of curves on the surface. Scallop refers to the remaining material that is generated when the cutter sweeps along two neighbor paths, which results in deviation between the machined surface and the design surface. Generally, we use the height of the points at the ridge of the scallop to the design surface to quantify such error, as illustrated in Fig. 2(a). On one hand, the closer the two neighboring curves are, the lower the scallop height becomes. On the other hand, closer curves may lead to longer path and time to machine the whole surface. The iso-scallop method generates tool paths with the scallop height as high as a specific tolerance, which results in no redundant machining and achieves higher efficiency. The scallop height is determined by the interval between two neighbor paths and they are related with the following formula [10]

$$h = \frac{\kappa_s + \kappa_c}{8} w^2 + \mathrm{O}\left(w^3\right), \tag{1}$$

where w denotes the interval $p_i p_{i+1}$, h is the scallop height, κ_s is the normal curvature along the direction normal to path C_i, as shown in Fig. 2(b), and κ_c is the curvature of the cutter.

Let C_i, C_{i+1} be the iso-level curves $\{p \in S \mid \varphi(p) = l_i\}$ and $\{p \in S \mid \varphi(p) = l_{i+1}\}$, respectively. Then we have the following Taylor expansion

$$l_{i+1} - l_i = (\nabla\varphi)^T (p_{i+1} - p_i) + \mathrm{O}\left(\|p_{i+1} - p_i\|^2\right). \tag{2}$$

The gradient $\nabla\varphi$ is a vector in the tangent plane of the surface at point p_i, and is normal to the path. Therefore, the expansion can be rewritten as

$$|l_{i+1} - l_i| = \|\nabla\varphi\| \cdot \|p_{i+1} - p_i\| + \mathrm{O}\left(\|p_{i+1} - p_i\|^2\right). \tag{3}$$

Thus

$$\|\nabla\varphi\| = \lim_{\|p_{i+1}-p_i\|\to 0} \frac{|l_{i+1} - l_i|}{\|p_{i+1} - p_i\|}. \tag{4}$$

If the level increment $|l_{i+1} - l_i|$ of the scalar funciton is endowed with a machining meaning by letting it equal to the square root of scallop height, Eq. (4) will be

$$\|\nabla\varphi\| = \sqrt{\frac{\kappa_s + \kappa_c}{8}}, \tag{5}$$

and the scallop height between two neighbor iso-level paths will be constant and equal to the square of the increment. This can be easily derived by substituting Eq. (1) into Eq. (4).

Thus an iso-scallop tool path can be generated by finding a scalar function satisfying Eq. (5). And we obtain such φ by solving a nonlinear least square problem

$$\min_{\varphi} \quad E_w(\varphi) = \int_{\mathcal{S}} \left(\|\nabla\varphi\| - \sqrt{\frac{\kappa_s + \kappa_c}{8}} \right)^2 d\mathcal{S} \tag{6}$$

where κ_c is a user input and κ_s is computed by

$$\kappa_s = \left(\frac{\nabla\varphi}{\|\nabla\varphi\|} \right)^T T \left(\frac{\nabla\varphi}{\|\nabla\varphi\|} \right) \tag{7}$$

with T denoting the curvature tensor (see [26, 27] for its definition and numerical computation).

Finally, the iso-level curves corresponding to level values $\left\{ i\sqrt{h} \right\}_{i=1}^{n}$ are iso-scallop tool path with constant height h, that is, level increments between neighboring iso-level curves all equal to $\sqrt{h}$. Thus, a large level increment can generate a tool path for rough machining, and a small increment for finish machining. The novelty here is that they share the same scalar function. We refer to this as multiresolution property.

2.2. Smooth tool path generation

As explained in the introduction section, a smooth tool path is preferred as we can get a nearly constant feed rate along it. For a curve in 3D space, its curvature measures how much it bends at a given point. This is quantified by the norm of its second derivative with respect to arc-length parameter, which measures the rate at which the unit tangent turns along the curve [26]. It is the very metric to measure the smoothness of the curve.

As the tool path is embedded on the design surface, its second derivative with respect to arc-length parameter can be decomposed into two components, one tangent to the surface and the other normal to the surface (see Fig. 3) [26]. The norms of these components are called the geodesic curvature and the normal curvature respectively, and they are related to the curve curvature by

$$\kappa^2 = \kappa_g^2 + \kappa_n^2, \tag{8}$$

where κ is the curve curvature, and κ_g, κ_n are geodesic curvature and normal curvature respectively.

For an iso-level curve $\phi = const$, its normal curvature can be computed by

$$\begin{aligned} \kappa_n &= \left(n \times \tfrac{\nabla\varphi}{\|\nabla\varphi\|}\right)^T T \left(n \times \tfrac{\nabla\varphi}{\|\nabla\varphi\|}\right) \\ &= \left(A\tfrac{\nabla\varphi}{\|\nabla\varphi\|})^T T (A\tfrac{\nabla\varphi}{\|\nabla\varphi\|}\right) \\ &= \left(\tfrac{\nabla\varphi}{\|\nabla\varphi\|})^T T' (\tfrac{\nabla\varphi}{\|\nabla\varphi\|}\right), \end{aligned} \tag{9}$$

where T is the curvature tensor, $n = (n_x, n_y, n_z)^T$ is the normal vector, and

$$A = \begin{bmatrix} 0 & -n_z & n_y \\ n_z & 0 & -n_x \\ -n_y & n_x & 0 \end{bmatrix}, \quad T' = A^T T A. \tag{10}$$

The geodesic curvature can be computed by

$$\kappa_g = \mathrm{div}\left(\frac{\nabla\varphi}{\|\nabla\varphi\|}\right), \tag{11}$$

where $\mathrm{div}(\cdot)$ is the divergence operator. For a planar curve, its normal curvature is zero and we have

$$\kappa = \kappa_g = div(\frac{\nabla\varphi}{\|\nabla\varphi\|}) \neq div(\nabla\varphi) = \nabla^2(\varphi). \tag{12}$$

Therefore, Laplacian can't ensure smoothness of a tool path for pocket milling.

To guarantee the smoothness of all iso-level curves on surface S, we define the smoothness energy as

$$E_\kappa(\varphi) = \int_{\mathcal{S}} \kappa^2 d\mathcal{S} = \int_{\mathcal{S}} \kappa_g^2 d\mathcal{S} + \int_{\mathcal{S}} \kappa_n^2 d\mathcal{S}. \tag{13}$$

In Section 2.1, we employ the formula $|l_{i+1} - l_i| = \sqrt{h}$ to generate iso-level tool path. But for smooth tool path, the following strategy is exploited: First, a certain number of points are sampled from iso-level curve C_i; Then the level increment $|l_{i+1} - l_i|$ is computed for each point with respect to a given scallop height h using Eq. (1) and Eq. (3); Finally, the smallest level increment is chosen to be the level increment between C_i and its next path C_{i+1}. This results in level increments of different values as opposed to the iso-scallop method, while the scalar function remains unchanged, i.e., the multiresolution property still holds.

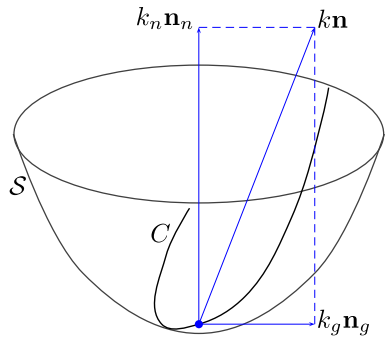


Figure 3: The curvature vector $k\mathbf{n}$ of curve C on $\mathcal{S}$ has two orthogonal components: the normal curvature vector $k_n\mathbf{n}_n$ and the geodesic curvature vector $k_g\mathbf{n}_g$.

2.3. Optimal tool path generation

The width term Equ. (5) and the smoothness term Equ. (8) can control the interval between neighboring paths and smoothness of the paths, respectively. Thus an optimal tool path in terms of iso-scallop and smoothness can be obtained by computing φ through a nonlinear least square optimization which minimizes a linear combination of the two objectives

$$E(\varphi) = E_w(\varphi) + \lambda E_\kappa(\varphi) \tag{14}$$

where λ is a positive weight controlling the trade-off between the two terms. And in order to ensure the tool path is regular (i.e., either contour parallel or direction parallel), we introduce a hard constraint $\|\nabla\varphi\| > 0$. The impact of this constraint is demonstrated in Section 2.4. Then the optimization problem becomes

$$\begin{aligned} &\min_{\varphi} \int_S \left(\|\nabla\varphi\| - \sqrt{\frac{\kappa_s + \kappa_c}{8}} \right)^2 + \lambda \left(\kappa_g^2 + \kappa_n^2 \right) d\mathcal{S} \\ &\text{s.t.} \quad \|\nabla\varphi\| > 0. \end{aligned} \tag{15}$$

However, because of the smoothness energy, the optimization result may violate Equ. (5), and the formula $|l_{i+1} - l_i| = \sqrt{h}$ would be invalid. Therefore, we employ the method described in Section 2.2 to select iso-level curves with respect to a certain scallop height tolerance. Note that we can use the same scalar function for planning tool paths of different scallop height tolerances, which shows the multiresolution property of our approach.

Since different machine tools have different feed rate capability, for those of good capability we can choose a lower weight on smooth term. Thus the freedom of choosing weights λ provides the possibility of applying the proposed method to various machine tools.

2.4. Path topology

In this section, we will show that each iso-level curve generated by the proposed method is either a loop or a curve segment without self-intersection and disjunction. In addition, this kind of path topology can be exploited to quickly extract iso-level curves.

Lemma 1. *For a given scalar function φ over a surface S, if the norm of its gradient does not vanish anywhere, the endpoints of iso-level curves (if they exist) are on the boundary.*

Proof. For a interior point p, $\|\nabla\varphi\| \neq 0$ implies that along the two directions $\triangle p_1, \triangle p_2$ orthogonal to $\nabla\varphi$, we have, in a small range, the following expression

$$\varphi(p + \triangle p_i) - \varphi(p) = (\nabla\varphi)^T \triangle p_i = 0 \quad \text{for} \quad i = 1, 2. \tag{16}$$

Namely, each interior point has exactly two directions sharing the same level value with it. Therefore, the endpoints can only be on the boundary. □

Lemma 2. *For the scalar function φ, its iso-level curves never intersect with each other and do not have self-intersections.*

Proof. Since each point corresponds to a unique value, iso-level curves for different values do not intersect with each other. Generally, we have two types of self-intersections, as shown in Fig. 4. The difference between them is that in (b) the self-intersection is tangential. For case (a), the two curve segments have different tangent directions at the self-intersection point. It is well-known that the gradient direction at a point is orthogonal to the tangent direction of the iso-level curve. Thus the two different tangent directions at the self-intersection point results in a contradiction that there are two different gradient direction at that point. For case (b), we view the self-intersection as two iso-level curves that are of same level value and tangential at the intersection point and separate from each other in its neighborhood. But $\|\nabla\varphi\| \neq 0$ along an iso-level curve implies that iso-level curves near it are of different level values, which again leads to a contradiction. □

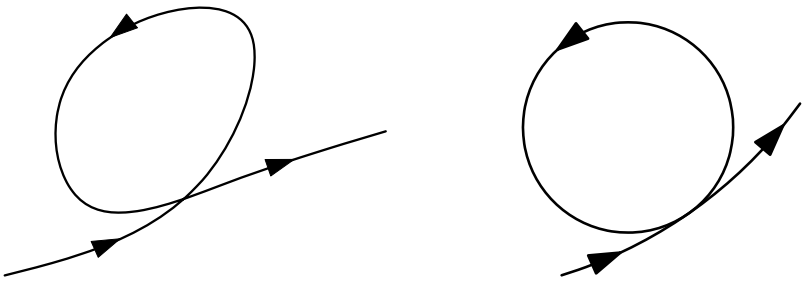


Figure 4: Self-intersection illustrations

Note that these properties are employed to extract each desired iso-level curves in the following sections so that traversal is not needed. And it follows immediately from Lemma 1 and Lemma 2 that:

Proposition 1. *Each iso-level path generated by the proposed method is either contour parallel or direction parallel and free from self-intersection and disjunction.*

3. Numerical solution

The iso-level method described in the above section can be applied to any domain with a discrete gradient operator ∇, divergence operator $\mathrm{div}(\cdot)$, and curvature tensor T. In this work, we focus on computing the tool path on triangular meshes. However, this method can be easily extended to other domains, such as polygonal surfaces and point clouds.

Assume that $M \subset \mathbb{R}^3$ is a compact triangulated surface with no degenerate triangles. Let $N_1(i)$ be the 1-neighborhood of vertex v_i, which is the index set for vertices connecting to v_i. Let $D_1(i)$ be the 1-disk of the vertex v_i, which is the index set for triangles containing v_i. The dual cell of a vertex v_i is part of its 1-disk which is more near to v_i than its $N_1(i)$. Fig. 5 (a) shows the dual cell C_i for an interior vertex v_i, while Fig. 5 (b) shows the dual cell for a boundary vertex. A function φ defined over the triangulated surface M is considered to be a piecewise linear function, such that φ reaches value φ_i at vertex v_i and is linear within each triangle. Based on these, the energies shown in Equ. (6), Equ. (13), and Equ. (15) are computed by integrating the width term and smooth term over the whole mesh domain, while the mesh domain can be decomposed into a set of triangles or a set of dual cells. To compute the width term and the smooth term on a mesh, we need to discretize the gradient, the divergence, and the curvature tensor, which we briefly discuss below, since they are basic in the Finite Element Method (FEM).

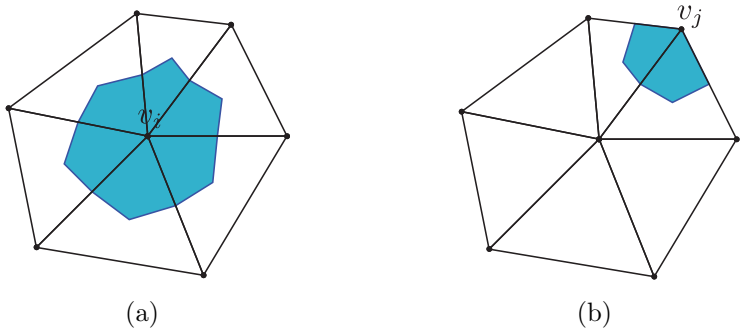


Figure 5: Dual cells. (a) dual cell of an interior vertex; (b) dual cell of a boundary vertex.

The gradient of φ over each triangle is constant as the function φ is linear

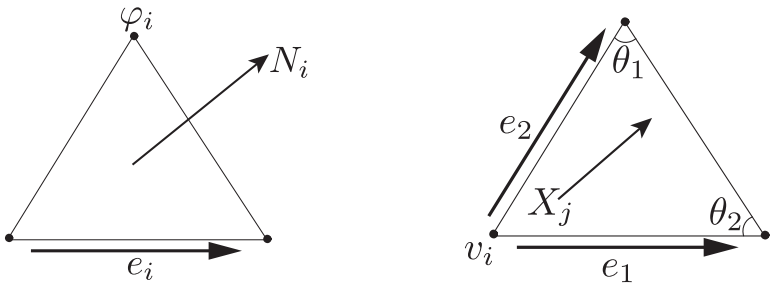


Figure 6: Triangles of gradient (a) and divergence (b).

within the triangle. The gradient in a given triangle can be expressed as

$$\nabla\varphi(f) = \frac{1}{2A_f}\sum_i \varphi_i(N \times e_i), \tag{17}$$

where A_f is the area of the face f, N is its unit normal, e_i is the $i-$th edge vector (oriented counter-clockwise), and φ_i is the opposing value of φ as shown in Fig. 6.

According to the Stokes' theorem, the integral of divergence over the dual cell is equal to the outward flux along the boundary of the dual cell. Thus the divergence operator associated with vertex v_i is dicretized by dividing the outward flux by the dual cell area

$$\mathrm{div}(X) = \frac{1}{2C_i}\sum_{j\in D_1(i)} \cot\theta_1(e_1 \cdot X_j) + \cot\theta_2(e_2 \cdot X_j), \tag{18}$$

where the sum is taken over incident triangles f_j with a vector X_j, e_1 and e_2 are the two edge vectors of triangle f_j containing vertex v_i, θ_1 and θ_2 are the opposing angles, and C_i is the dual cell area for vertex v_i. Accordingly, the geodesic curvature value of curve $\varphi = const$ can be computed by

$$\kappa_g(v_i) = \frac{1}{2C_i}\sum_{j\in D_1(i)} \frac{1}{\|(\nabla\varphi)_j\|}(\cot\theta_1(e_1 \cdot (\nabla\varphi)_j) + \cot\theta_2(e_2 \cdot (\nabla\varphi)_j)). \tag{19}$$

The curvature tensor (second fundamental tensor) T is defined in terms of the directional derivatives of the surface normal:

$$T = \begin{pmatrix} D_u n & D_v n \end{pmatrix} = \begin{pmatrix} \frac{\partial n}{\partial u}\cdot u & \frac{\partial n}{\partial v}\cdot u \\ \frac{\partial n}{\partial v}\cdot u & \frac{\partial n}{\partial v}\cdot v \end{pmatrix}, \tag{20}$$

where (u, v) are the directions of an orthogonal coordinate system in the tangent frame (the sign convention used here yields positive curvatures for convex surfaces with outward-facing normals). Multiplying this tensor by any vector in the tangent plane gives the derivative of the normal in that direction. Although this definition holds only for smooth surfaces, we can approximate it in the discrete case using finite difference. In this work, the curvature tensor for each face is computed by the method in [27].

Then the whole optimization model can be formulated as

$$\begin{aligned} \min_{\varphi} \quad & \sum_{j=1}^{|F|} A_{f_j} \left(\|\nabla\varphi\| - \sqrt{\frac{\kappa_s + \kappa_c}{8}} \right)^2 + \lambda \left(\sum_{j=1}^{|F|} A_{f_j} \kappa_n^2 + \sum_{i=1}^{|V|} C_i \kappa_g^2 \right) \\ \text{s.t.} \quad & \|\nabla\varphi(f_j)\| > 0 \end{aligned} \tag{21}$$

where $|F|$ is the number of faces and $|V|$ denotes the number of vertices. This is a well established nonlinear least square optimization problem with inequality constraints, which can be easily solved by the interior point method [28–30]. The interior point solver requires the gradient of the target function and the constraint functions. The gradient calculation boils down to computing the gradient of $\nabla\varphi$ and $\nabla\varphi/\|\nabla\varphi\|$, which we do as follows.

As demonstrated previously, the gradient of a piecewise linear scalar function in a given triangle is a linear combination of constant vectors $N \times e_i$, and thus, the gradient of $\nabla\varphi$ is

$$\frac{\partial}{\partial\varphi_j}\nabla\varphi(f) = \frac{1}{2A_f}\frac{\partial}{\partial\varphi_j}\sum_i \varphi_i(N \times e_i) = \frac{1}{2A_f}\sum_i \delta_{ij}(N \times e_i) \tag{22}$$

where $\delta_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$ is the Kronecker delta function.

As for the gradient of $\nabla\varphi/\|\nabla\varphi\|$, it is

$$\frac{\partial}{\partial\varphi_j}\left(\frac{\nabla\varphi(f)}{\|\nabla\varphi(f)\|}\right) = \frac{\left(\frac{\partial}{\partial\varphi_j}\nabla\varphi(f)\right)\|\nabla\varphi(f)\| - \nabla\varphi(f)\frac{(\nabla\varphi(f))^T\frac{\partial}{\partial\varphi_j}\nabla\varphi(f)}{\|\nabla\varphi(f)\|}}{\|\nabla\varphi(f)\|^2} \tag{23}$$

The final solution to the optimization problem Equ. (21) depends on the initial value. In this work, we initialize the tool path by paths from [31].

4. Tool path planning algorithm

Planning tool-path is to represent a surface with a series of curves against some error criteria (i.e., chord deviation and scallop height). We next summarize the overall process for generating such curves on a surface by the iso-level method as follows:

1. Select an initial curve C_0 on the surface S and fix its level value to zero, i.e., $l_0 = 0$. C_0 is a part of boundary for direction parallel tool path and the whole boundary for contour parallel tool path.
2. Find the optimizer of the model Equ. (6), Equ. (13), and Equ. (15), including meshing and numerical optimization.
3. Select level values $\{l_i\}_{i=1}^{n}$, where $l_{(n-1)} < \varphi_{max}, \quad l_{(n)} \geq \varphi_{max}$, with the method described in Section 2.1 for iso-scallop tool path and the method in Section 2.2 for smooth or optimal tool path. For direction parallel tool path the last level value $l_n = \varphi_{max}$, while for contour parallel tool path the last level value is l_{n-1}. Then fastly extract iso-level curves on the triangular mesh based on the path topology described in Section 2.4.
4. Convert the iso-level curves on mesh which actually are polygons to surface S. The vertices of an iso-level curve on the mesh are either vertices of the mesh or points on edges of the mesh. For the former case, the vertices are also on S. For the latter case, a vertex is first proportionally mapped to the parameter domain with respect to the two ends of the edge it is on and then find its corresponding point on the surface.
5. Greedily merge short segments of the polygons to approach the chord deviation tolerance as closely as possible. And finally, these reduced iso-level curves (polygons) are desired tool path.

5. Experimental results

In this section, the proposed tool path planning method is applied on real data. A free-form surface and a human face are chosen to illustrate the effectiveness of it, as in Fig. 7. The free-form surface is exploited to show the generation of direction parallel tool path. The human face was generated by a coordinate measuring machine. We utilize it to show the generation of contour parallel tool path.

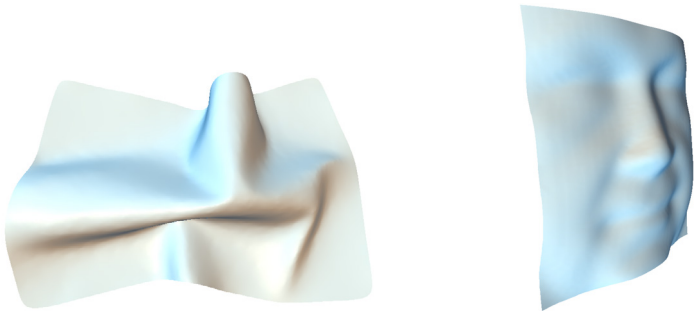
For planning iso-level tool path, the first thing to do is to construct a proper scalar function over the surface. Since the Finite Element Method is employed to find the optimizer of the optimization models, meshing is needed. We choose the element to be triangular. Fig. 8(a) shows the meshing results of the free-form surface and Fig. 9(a) shows that of the human face. The optimal scalar functions are illustrated in Fig. 8(b) and Fig. 9(b) by varying color. Fig. 8(b) shows the scalar function of the free-form surface for generating direction parallel tool path and Fig. 9(b) shows that of the human face for generating contour parallel tool path. And the varying from blue to red represents the rising of level value.

As the optimal scalar functions have been constructed for both surfaces, tool path that is optimal with respect to iso-scallop and smoothness can be generated. A ball-end cutter with radius 4mm is chosen to show the path generation so that tool orientation doesn't matter. The limited scallop height is 1mm and chord deviation is 0.01mm. In order to clearly show tool paths, the error criterion (i.e., scallop height) is set to be much greater than those in real cases. Fig. 8(c) shows the optimal direction parallel paths on the free-form surface and Fig. 9(c) shows corresponding result of contour parallel tool path on the human face. Their weights are both $\lambda = 1$.

We next show some comparisons of the generated tool path. According to the demonstration in [32], the contour parallel tool path will be the emphatic one. Fig. 10 shows the tool paths form smooth to iso-scallop generated by the proposed method. Fig. 10(a) shows the smooth contour parallel tool path generated by the proposed method with $\lambda = 10$, Fig. 10(b) shows the optimal tool path, and Fig. 10(c) shows the iso-scallop tool path (i.e., $\lambda = 0$). As the figures show, for the optimal tool path, the intervals are rather even and thus the machining strips are less overlapped for neighbor paths, as opposed to the path in Fig. 10(a), in addition, there are no sharp corners for the optimal tool path, as opposed to iso-scallop tool path in Fig. 10(c). We also compare our method with the Laplacian based method in Fig. 11 from which we can find that the Laplacian tool path is severely overlapped for neighbor paths, especially for paths near the boundary.

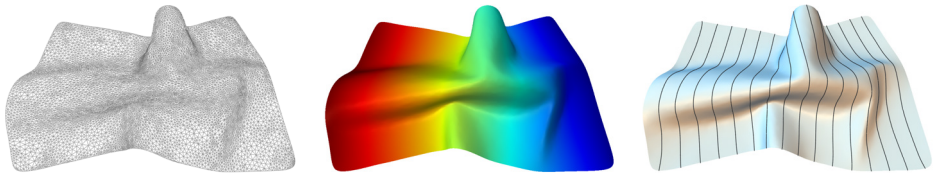
6. Conclusion

In this paper, a new framework of tool path planning is proposed. The novelty of our method is that it allows several objectives to be considered in a unified framework and thus making global optimization of tool path-



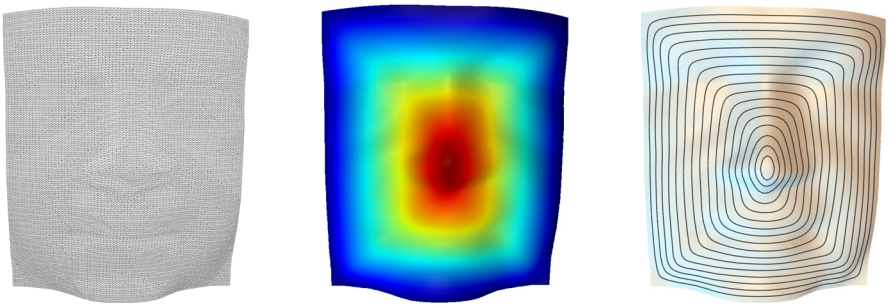
(a) (b)

Figure 7: Tested models. (a) free-form surface; (b) human face.



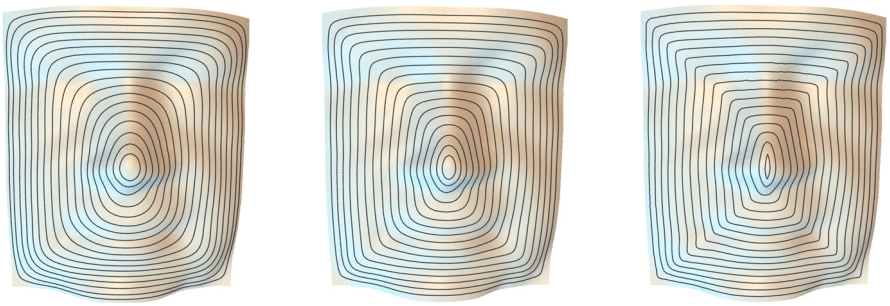
(a) (b) (c)

Figure 8: Direction parallel tool path for the free-form surface. (a) meshing result; (b) optimal scalar function; (c) the generated tool path.



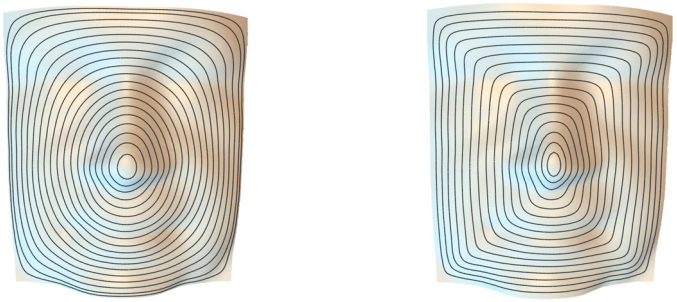
(a) (b) optimal scalar function (c) the generated tool path

Figure 9: Contour parallel tool path for the face model. (a) meshing result; (b) optimal scalar function; (c) the generated tool path.



(a) (b) (c)

Figure 10: Tool path from smooth to iso-scallop. (a) smooth tool path; (b) optimal tool path; (c) iso-scallop tool path.



(a) (b)

Figure 11: Comparison of (a) Laplacian based tool path and (b) optimal tool path by the proposed method.

s possible. Moreover, the scalar function only has to be constructed once, then it can be utilized to generate tool paths for machining from rough to fine. The proposed framework is applied to find an optimal tool path that takes smoothness and iso-scallop requirements into consideration simultaneously. Equ. (5) for controlling interval between neighbor iso-level curves and Equ. (8) for measuring curvature of an iso-level curve are derived to lay a foundation for the following formulation of optimization models.

It is likely that this theory has further potential in planning other optimal tool path, and the derived formulas can also be directly applied to level set based tool path planning methods, e.g., [22].

7. Acknowledgement

The authors are grateful for the support provided by National Key Basic Research Project of China (No. 2011CB302400), National Natural Science Foundation of China (No. 61303148, 51175495), Ph.D. Programs Foundation of Ministry of Education of China (No. 20133402120002), and Swiss National Science Foundation (No. 200021_137626).

References

[1] T. Kim, S. E. Sarma, Toolpath generation along directions of maximum kinematic performance; a first cut at machine-optimal paths, Computer-Aided Design 34 (6) (2002) 453–468.

[2] G. H. Kumazawa, Generating efficient milling tool paths according to a preferred feed direction field, Master's thesis, University of British Columbia (2012).

[3] G. C. Loney, T. M. Ozsoy, Nc machining of free form surfaces, Computer-Aided Design 19 (2) (1987) 85–90.

[4] S. Yuwen, G. Dongming, W. Haixia, et al., Iso-parametric tool path generation from triangular meshes for free-form surface machining, The International Journal of Advanced Manufacturing Technology 28 (7-8) (2006) 721–726.

[5] Q. Zou, J. Zhao, Iso-parametric tool-path planning for point clouds, Computer-Aided Design 45 (11) (2013) 1459–1468.

[6] Y. Huang, J. H. Oliver, Non-constant parameter nc tool path generation on sculptured surfaces, The International Journal of Advanced Manufacturing Technology 9 (5) (1994) 281–290.

[7] S. Ding, M. Mannan, A. N. Poo, D. Yang, Z. Han, Adaptive iso-planar tool path generation for machining of free-form surfaces, Computer-Aided Design 35 (2) (2003) 141–153.

[8] H.-Y. Feng, Z. Teng, Iso-planar piecewise linear nc tool path generation from discrete measured data points, Computer-Aided Design 37 (1) (2005) 55–64.

[9] K. Suresh, D. Yang, Constant scallop-height machining of free-form surfaces, Journal of Engineering for Industry 116 (2) (1994) 253–259.

[10] Y. Koren, R. Lin, Efficient tool-path planning for machining free-form surfaces, Transactions of the ASME Journal of Engineering for Industry 118 (1996) 20–28.

[11] R. Sarma, D. Dutta, The geometry and generation of nc tool paths, Journal of mechanical design 119 (2) (1997) 253–258.

[12] H.-Y. Feng, H. Li, Constant scallop-height tool path generation for three-axis sculptured surface machining, Computer-Aided Design 34 (9) (2002) 647–654.

[13] J.-H. Yoon, Fast tool path generation by the iso-scallop height method for ball-end milling of sculptured surfaces, International journal of production research 43 (23) (2005) 4989–4998.

[14] T. Kim, Constant cusp height tool paths as geodesic parallels on an abstract riemannian manifold, Computer-Aided Design 39 (6) (2007) 477–489.

[15] H. Li, S. Yao, G. Li, Y. Liu, L. Zhang, Power series solution for isoscallop tool path generation on free-form surface with ball-end cutter, Mathematics in Computer Science 6 (3) (2012) 281–296.

[16] Z. Han, D. C. Yang, Iso-phote based tool-path generation for machining free-form surfaces, Journal of Manufacturing science and engineering 121 (4) (1999) 656–664.

[17] B. K. Choi, C-space approach to tool-path generation for sculptured surface machining, Geometric Modelling: Theoretical and Computational Basis Towards Advanced CAD Applications 75 (2001) 85–97.

[18] D. Dragomatz, S. Mann, A classified bibliography of literature on nc milling path generation, Computer-aided design 29 (3) (1997) 239–247.

[19] A. Lasemi, D. Xue, P. Gu, Recent development in cnc machining of freeform surfaces: A state-of-the-art review, Computer-Aided Design 42 (7) (2010) 641–654.

[20] V. Pateloup, E. Duc, P. Ray, Corner optimization for pocket machining, International Journal of Machine Tools and Manufacture 44 (12) (2004) 1343–1353.

[21] V. Pateloup, E. Duc, P. Ray, Bspline approximation of circle arc and straight line for pocket machining, Computer-Aided Design 42 (9) (2010) 817–827.

[22] S. Dhanik, P. Xirouchakis, Contour parallel milling tool path generation for arbitrary pocket shape using a fast marching method, The International Journal of Advanced Manufacturing Technology 50 (9-12) (2010) 1101–1111.

[23] N. Paragios, R. Deriche, Geodesic active regions: A new framework to deal with frame partition problems in computer vision, Journal of Visual Communication and Image Representation 13 (1) (2002) 249–268.

[24] M. B. Bieterman, D. R. Sandstrom, A curvilinear tool-path method for pocket machining, Journal of manufacturing science and engineering 125 (4) (2003) 709–715.

[25] J.-J. Chuang, D. C. Yang, A laplace-based spiral contouring method for general pocket machining, The International Journal of Advanced Manufacturing Technology 34 (7-8) (2007) 714–723.

[26] M. P. D. Carmo, Differential Geometry of Curves and Surfaces, Prentice-Hall, 1976.

[27] S. Rusinkiewicz, Estimating curvatures and their derivatives on triangle meshes, in: 2nd International Symposium on 3D Data Processing, Visualization and Transmission., IEEE, 2004, pp. 486–493.

[28] T. F. Coleman, Y. Li, An interior trust region approach for nonlinear minimization subject to bounds, SIAM Journal on Optimization 6 (1996) 418–445.

[29] A. Wächter, L. T. Biegler, On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming, Mathematical Programming 106 (2006) 25–57.

[30] F. E. Curtis, J. Huber, O. Schenk, A. Waechter, A note on the implementation of an interior-point algorithm for nonlinear optimization with inexact step computations., Math. Program. 136 (1) (2012) 209–227.

[31] K. Crane, C. Weischedel, M. Wardetzky, Geodesics in heat: A new approach to computing distance based on heat flow, ACM Transactions on Graphics (TOG) 32 (5) (2013) 152.

[32] B. H. Kim, B. K. Choi, Machining efficiency comparison direction-parallel tool path with contour-parallel tool path, Computer-Aided Design 34 (2) (2002) 89–95.