Automatic fitting of conical envelopes to free-form surfaces for flank CNC machining

Pengbo Bo[a], Michael Bartoň[*,b], Helmut Pottmann[c]

[a]*School of Computer Science and Technology, Harbin Institute of Technology, West Wenhua Street 2, 264209 Weihai, China*
[b]*BCAM – Basque Center for Applied Mathematics, Alameda de Mazarredo 14, 48009 Bilbao, Basque Country, Spain*
[c]*Center for Geometry and Computational Design, Vienna University of Technology, Wiedner Hauptstr. 8-10/104, A-1040 Vienna, Austria*

Abstract

We propose a new algorithm to detect patches of free-form surfaces that can be well approximated by envelopes of a rotational cone under a rigid body motion. These conical envelopes are a preferable choice from the manufacturing point of view as they are, by-definition, manufacturable by computer numerically controlled (CNC) machining using the efficient flank (peripheral) method with standard conical tools. Our geometric approach exploits multi-valued vector fields that consist of vectors in which the point-surface distance changes linearly. Integrating such vector fields gives rise to a family of integral curves, and, among them, linear segments that further serve as conical axes are quickly extracted. The lines that additionally admit tangential motion of the associated cone along the reference geometry form a set of candidate lines that are sequentially clustered and ordered to initialize motions of a rigid truncated cone. We validate our method by applying it on synthetic examples with exact envelopes, recovering correctly the exact solutions, and by testing it on several benchmark industrial datasets, detecting manufacturable conical envelope patches within fine tolerances.

Key words: 5-axis CNC machining, flank milling, free-form surface, tangential movability, shape manufacturing

1. Introduction & Motivation

Free-form surfaces are a popular modeling tool for engineers, architects, and designers in general [6]. Most commonly represented as non-uniform rational B-splines (NURBS), these surfaces are supported by a vast majority of the state-of-the-art computer-aided design (CAD) software. Using such software, the modeling stage of a free-form surface is intuitive via local adjustment of the control points. On the other hand, the manufacturing (or realization) stage is difficult, particularly because of the very diverse nature of a general free-form surface. An important step towards by-definition-manufacturable surfaces has been taken recently by proposing an *interactive* modeling tool with developable surfaces [28]. However, these surfaces are not generally free-form, and therefore the problem of accurate approximation of a general doubly-curved surface is still highly desirable.

Driven by industrial demands, efficient and high-precision manufacturing methods of free-form surfaces have been an active field for several decades [8, 24, 2, 15, 4, 5, 20, 1]. Manufacturing of free-form objects in a serial manner is typically realized using *molds* where a material (e.g., heated to become flexible) is pasted between the molds to form the desired shape. The leading manufacturing technology of free-form molds is 5-axis computer numerically controlled (CNC) machining which is a subtractive technology where a rotational tool is navigated along a workpiece, removing redundant material to shape the desired geometry. Depending on the contact between the machining tool and the design surface, various strategies are known. Whereas the most popular approaches of flat-end and rounded-end milling consider only point-point contact between the tool and the machined surface [19, 9, 34], a more efficient strategy is known to be *flank machining* [14], where the tool possesses a tangential contact with the design surface Φ along a 3D curve. The advantage of flank milling is not only the fact that one removes more material by a single sweep of the tool, but more importantly, this strategy is by definition scallop-free because of the tangential contact along the whole contact curve. To initialize the milling tool to meet the tangential contact condition along a whole 3D curve (infinitely many points), however, is theoretically impossible unless the surface is an exact *envelope*. Therefore, it is difficult to find a good approximation that *globally* minimizes the error between the design surface and its approximation when compared to single contact point approaches.

Our work belongs to the category of *"digital" reverse engineering*, where a general free-form surface is given as an input, and the goal is to find sub-patches that can be machined to high precision with flank milling. In this work, we restrict ourselves to *conical* tools and derive an automatic algorithm that detects *conical envelope patches*.That is, given an input surface Φ and parameters defining a conical cutter, we seek sub-parts of Φ

[*]Corresponding author
Email addresses: pengbo@hitwh.edu.cn (Pengbo Bo), mbarton@bcamath.org (Michael Bartoň), pottmann@geometrie.tuwien.ac.at (Helmut Pottmann)

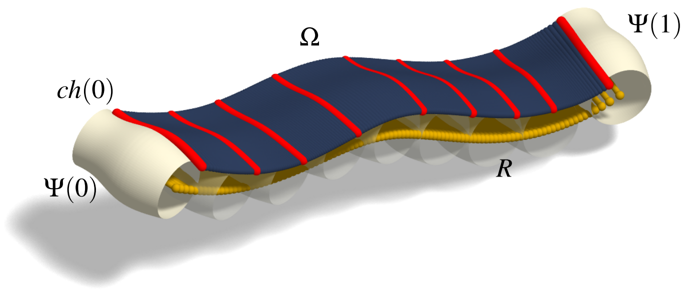


Figure 1: An envelope Ω of a one-parameter motion of a general surface of revolution Ψ. At each time instant t, $\Psi(t)$ touches Ω along a 3D curve known as characteristic $ch(t)$ (red). The trajectory of the rotational axis defines a ruled surface (yellow). High quality approximation of free-form surfaces by envelopes of general rotational surfaces has been studied recently [3] (figure courtesy of Bo et al. [3]).

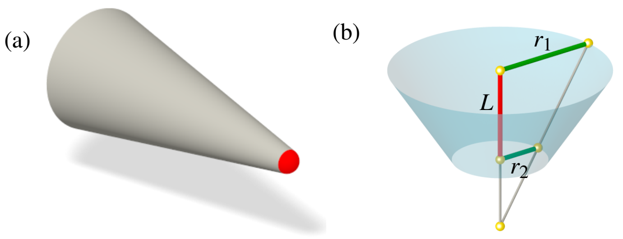


Figure 2: (a) A conical milling tool typically posses a rounded tip (red). In our considerations, we disregard this minor part and consider only the main body, i.e., the truncated cone. (b) Truncated cone is determined by three parameters, e.g., top and bottom radii r_1, r_2, and axis length L. Alternatively, a median value w of the distance function and its slope c can be used instead of the boundary radii.

that can be well-approximated by 3D motions of a truncated cone within a predefined tolerance.

2. Previous work and contributions

Our research belongs to the flank machining category [14], where the design surface and machining tool share a tangential contact along a 3D curve (known as characteristic), see Fig. 1. In particular, we focus on the case of conical milling tools [17, 18, 37, 20, 8, 27], which are the most frequently used tools for flank milling [26]. Typically, conical milling tools are provided by a ball tip, see Fig. 2, that for the simplicity of the argument we exclude from our considerations, and simplify the tool as a truncated cone. Truncated cones need only three degrees of freedom to specify the *meridian* which is a simplification when compared to cutters where the meridian is a general B-spline curve [3].

A special subclass are cylindrical cutters. Their axes have zero slope with respect to the tangent plane of the surface. Senatore et al. [25] provide analysis with respect to the maximal size of the cylindrical tool in order to cover large patches while satisfying the error between the machined and designed surface. Sprott and Ravani [27] present an algorithm that controls the under- and overcutting of the machined surface.

A lot of research has been devoted to ruled surfaces [8, 24, 21, 22, 12, 13, 32, 33, 29] and references cited in [14]. Ruled surfaces can be seen as limits of general envelopes when the moving surface of revolution degenerates to a single line. This fact, in the context of CNC-milling, makes a strong restriction on the milling tool, as to meet fine numerical tolerances for approximation of non-developable ruled patches, the milling tool must converge to a straight line. Elber and Fish [8] approximate a general free-form surface using piece-wise ruled surfaces. A subdivision strategy is applied, resulting in a large number of patches, each patch requiring the shape of the tool to be a cylinder with a tiny radius (a straight line in the limit). In contrast, we approximate the input surface by envelopes of conical tools of arbitrary given slope and thickness.

Optimization-based methods initialize the trajectory of the axis (a ruled surface), and sequentially optimize only the motion of the surface of revolution [24, 13, 35], or both the motion and the shape of the cutter [36, 20, 3]. Redonnet et al. [24] consider a cylindrical cutter for machining ruled surfaces and optimize its position such that it possess a tangential contact to three particular lines: one ruling of the surface and two tangent lines of the rail curves of the ruled surface.

Zhu et al. [20] simultaneously optimize the tool's motion and shape, considering physical constraints such as the stiffness of the cutter. The initialization stage, however, is achieved using the algorithm of Bedi et al. [2] that *locally* improves the position of the axis at one time instant to meet the tangential contact constraints at the endpoints of the tool. On the contrary, our strategy explores the space of axes *globally* to find the best tangentially-movable line segments. An algorithm that recognizes a surface of revolution from its implicit equation, and computes its axis and meridian, has been presented recently [31].

Aprroximation of free-form surfaces by envelopes of general surfaces of revolution were studied in [3] and conical envelopes were considered as a special case. A surface of revolution and its trajectory are simultaneously optimized to improve the approximation quality. The initialization stage, however, requires users' intervention to indicate the motion of the cutter. Such a human interaction in the automated machining process is highly undesirable as it requires a certain experience of the user, it is time consuming (some initial trajectories do not lead to high quality approximation), and is unrepeatable. In contrast, the research of this paper focuses on fully automatic initialization of the conical tool motions.

Gelfand and Guibas [11] present a reverse-engineering algorithm that seeks segmentation of the input data (point clouds) to find parts that can be well approximated by simple kinematic objects (planes, spheres, and cylinders). These kinematic surfaces admit tangential gliding motions, i.e., are so called *slippable*, which typically corresponds to movable mechanical components of CAD objects.

Contributions. We aim at a fully automatic algorithm that returns an *atlas* of conical envelopes that well approximate the input free-form surface. The main contribution (Section 4) is an initialization strategy that detects sequences of straight lines that can serve as discrete positions of a rigid truncated cone. We explore the space of candidate lines *globally* which is the

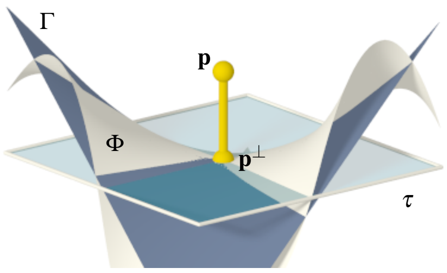


Figure 3: First and second order surface approximation. For a given point $\mathbf{p}$, and a free-form surface Φ, the approximation of the distance function $d(\mathbf{p},\Phi)$ can be efficiently computed using (8). The first (plane τ) and second (osculating paraboloid Γ) order approximation of Φ at $\mathbf{p}^{\perp}$ are shown.

main distinction to the previous work [20, 2, 3], where the initial motion of the cutter (a ruled surface) is given as an input. In contrast, our algorithm seeks automatically such ruled surfaces. In particular, we do the following:

- ⋆ Given a point $\mathbf{p}$ in 3-space and a design surface Φ, we use the second order approximation of Φ, see Fig. 3, analyze the point-surface distance, and determine directions in which the distance changes linearly, which is a necessary condition for a conical axis.
- ⋆ We introduce a multi-valued vector field and its integration gives rise to integral curves with linear distance function to Φ. Among these curves, we detect linear segments that admit tangential motion of the cone along Φ.
- ⋆ Candidate lines that correspond to a particular truncated cone are clustered and sequentially ordered to form an initial trajectory of the cone axis (a ruled surface). The motion of the cone then undergoes a global optimization to minimize the L^2-error between the design surface Φ and its approximation, a conical envelope Ω.

3. Distance function

Our aim is to detect several discrete positions of a rigid conical cutter Ψ (surface of revolution with a linear meridian) such that they all fit well the input surface Φ. Our algorithm is based on several facts from differential geometry. We first analyze the point-distance function.

3.1. Distance function of a surface

Let the design surface Φ be given and let d be the distance between $\mathbf{p}\in\mathbb{R}^3$ and Φ, i.e.,

$$\mathbb{R}^3 \quad \xrightarrow{d} \quad \mathbb{R}, \tag{1}$$

and let us consider its graph G, a hypersurface in $\mathbb{R}^4$.

At a fixed point $\mathbf{p}$, consider all 3D lines l passing through $\mathbf{p}$. We seek their directional vectors $\mathbf{v}$ such that the second directional derivative of d in $\mathbf{v}$ vanishes, i.e.,

$$\nabla_{\mathbf{v}}(\nabla_{\mathbf{v}} d) = \nabla^2_{\mathbf{v}} d = 0, \tag{2}$$

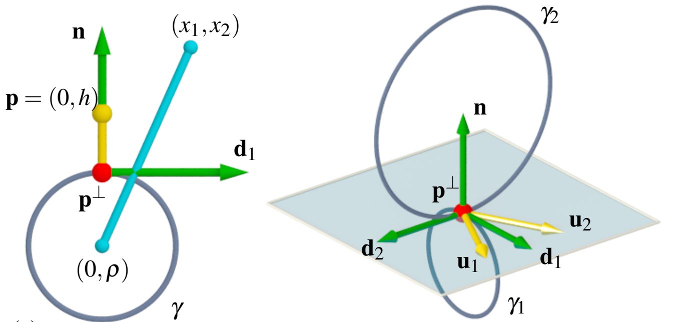


Figure 4: Geometry used for the approximation of the point-hypersurface distance function. (a) 2D scenario: a curve at a point $\mathbf{p}^{\perp}$ is approximated by its osculating circle γ with a radius ρ. The second order approximation of the distance function d between $\mathbf{x}=(x_1,x_2)^{\mathrm{T}}$ and γ is given by (6). (b) 3D analogy: a surface is approximated by an osculating paraboloid at a hyperbolic point (red). The principal frame (green) is shown together with the osculating circles (γ_1 and γ_2) lying in the principal normal planes $\{\mathbf{p}^{\perp},\mathbf{n},\mathbf{d}_1\}$ and $\{\mathbf{p}^{\perp},\mathbf{n},\mathbf{d}_2\}$, respectively. The asymptotic directions (yellow) satisfy (3).

which can be seen as the asymptotic directions on the graph of d. In the context of tangential-movability of a conical tool along Φ, l being its axis, condition (2) seeks the best candidates because the distance function changes linearly as we move from $\mathbf{p}$ in the direction $\mathbf{v}$. Therefore, the meridian is locally a straight line.

Eq. (2) is a quadratic constraint in $\mathbf{v}$, $\mathbf{v}\in\mathbb{R}^3$, and therefore the solutions form a quadratic cone. Moreover, we will show that this cone reduces to a pair of planes intersecting in the surface normal that passes through $\mathbf{p}$.

3.2. Degenerate cone of admissible directions

Firstly, we observe that Φ and any of its offsets Φ^{off} share the same distance functions (just translated by the value of the offset distance), and therefore we may assume that arbitrary $\mathbf{p}$ lies on some Φ^{off}.

At any point $\mathbf{p}$, the constraint (2) on candidate directions $\mathbf{v}$ reads as

$$\mathbf{v}^{\mathrm{T}} H \mathbf{v} = 0, \tag{3}$$

which is a quadratic form in $\mathbf{v}$, H being the *Hessian* matrix of d. This quadratic cone, however, will be shown to degenerate to a pair of planes (real or complex, depending on the sign of $\det(H)$).

If $\mathbf{p}$ is a hyperbolic point on Φ^{off}, (3) is a pair of real planes, and there are three obvious real directions emanating from $\mathbf{p}$ that satisfy (3): two asymptotic directions $\mathbf{u}_1$ and $\mathbf{u}_2$, and the surface normal vector $\mathbf{n}=\mathbf{u}_1\times\mathbf{u}_2$. Along the asymptotic lines, d is identically zero, while along $\mathbf{n}$ the distance varies in a linear fashion with unit speed ($\nabla_{\mathbf{n}} d = 1$).

Consider the principal frame $\{\mathbf{d}_1,\mathbf{d}_2,\mathbf{n}\}$ at $\mathbf{p}\in\Phi^{off}$. Since the distance function is (up to the second order) linear in the direction of $\mathbf{n}$, and constant in $\mathbf{u}_1$, and $\mathbf{u}_2$, (3) holds for all these three vectors. Moreover, expressing H with respect to the principal frame $\{\mathbf{d}_1,\mathbf{d}_2,\mathbf{n}\}$, (3) becomes

$$a_1v_1^2+a_2v_2^2+a_3v_3^2=0, \quad a_1,a_2,a_3\in\mathbb{R}, \tag{4}$$

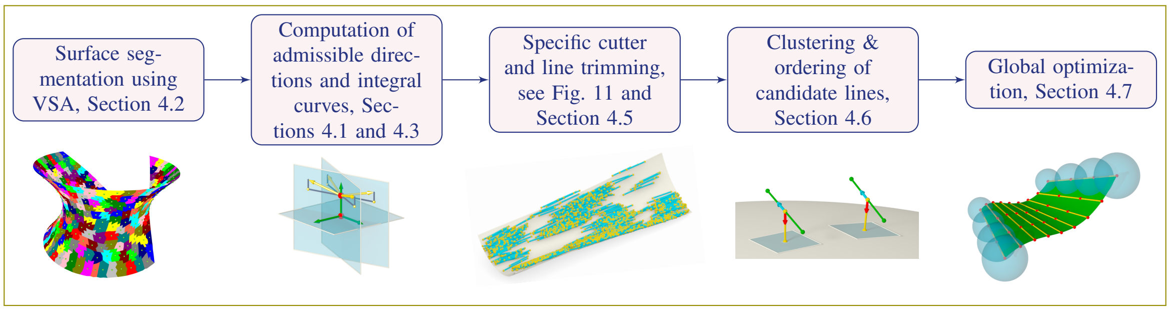

Surface segmentation using VSA, Section 4.2
Computation of admissible directions and integral curves, Sections 4.1 and 4.3
Specific cutter and line trimming, see Fig. 11 and Section 4.5
Clustering & ordering of candidate lines, Section 4.6
Global optimization, Section 4.7

Figure 5: Algorithm overview.

and since $\mathbf{n} = (0,0,1)^{\mathrm{T}}$ must comply with (4), it follows that $a_3 = 0$. Finally, this implies that the quadratic cone (3) decomposes to a pair of planes $\{\mathbf{p},\mathbf{n},\mathbf{u}_1\}$ and $\{\mathbf{p},\mathbf{n},\mathbf{u}_2\}$.

3.3. *Approximation of the distance function*

We want to approximate the distance function d of a surface Φ at a point $\mathbf{p}$, $\mathbf{p} \notin \Phi$. For the sake of simplicity, let us consider the 2D analogy first, see Fig. 4(a). Let $\mathbf{p}$ lie outside a curve ϕ at distance h and let γ be a second order approximation of ϕ, namely the osculating circle at the closest point $\mathbf{p}^{\perp}$. With $\mathbf{x} = (x_1,x_2)^{\mathrm{T}} \in \mathbb{R}^2$, the exact distance $d(\mathbf{x},\gamma)$ is

$$d(\mathbf{x},\gamma) = \rho + \sqrt{x_1^2 + (x_2-\rho)^2}, \tag{5}$$

$-\rho$ being the radius of γ. Direct computation of the Taylor expansion of d at $\mathbf{p} = (0,h)^{\mathrm{T}}$ gives

$$\widetilde{d}(\mathbf{x},\gamma) = x_2 + \frac{1}{2(h-\rho)}x_1^2 \tag{6}$$

which is the second order approximation of d, that is,

$$d(\mathbf{x},\gamma) = \widetilde{d}(\mathbf{x},\gamma) + \mathcal{O}(x_i^3), \quad i = 1,2. \tag{7}$$

Due to symmetries, the generalization of (6) to 3D becomes

$$\widetilde{d}(\mathbf{x},\Gamma) = x_3 + \frac{1}{2(h-\rho_1)}x_1^2 + \frac{1}{2(h-\rho_2)}x_2^2 \tag{8}$$

where Γ is the osculating paraboloid of Φ at $\mathbf{p}^{\perp}$, and $-\rho_{1,2}$ are the radii of the osculating circles in the principal normal planes, see Fig. 4(b).

4. Initialization algorithm for fitting conical envelopes

Our algorithm first investigates admissible vector fields in which the surface distance d changes linearly. Variational shape approximation is used to segment the input surface, and straight integral curves that admit tangential motion of the associated cone along the surface are computed. These line segments are clustered and ordered to define ruled surfaces, initial trajectories of a conical cutter. Finally, an optimization routine is applied to globally minimize the error between the input geometry and the conical envelope. An overview of our whole framework is shown in Fig. 5.

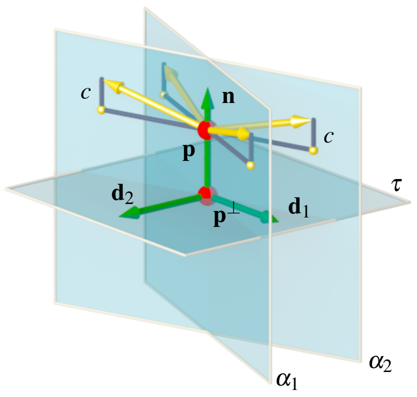


Figure 6: Admissible directions of a given slope. In general, at most four directions (yellow) of a constant slope emanating from $\mathbf{p}$ that solve (3) exist. All solutions of (3) form a singular cone, i.e., a pair of planes α_1 and α_2. The parameter c is the coordinate in the normal direction $\mathbf{n}$, and controls the slope of l with respect to the tangent plane τ of Φ at $\mathbf{p}^{\perp}$.

4.1. *Computation of admissible directions*

Let us consider $\mathbf{p} \notin \Phi$ to be expressed in the principal frame $\{\mathbf{d}_1,\mathbf{d}_2,\mathbf{n}\}$ of its footpoint (orthogonal projection) $\mathbf{p}^{\perp}$ on Φ, that is, $\mathbf{p} = (0,0,h)^{\mathrm{T}}$, see Fig. 6. Let us further consider a line l, the milling axis of a conical tool, passing through $\mathbf{p}$ such that the axis possesses a constant slope with respect to Φ. We wish to find lines such that the distance function d to Φ changes linearly (conical tool) when moving along l. This condition on the directional vector is exactly expressed by (3). Since (3) is a singular cone, i.e., a pair of planes passing through the surface normal $\mathbf{p}\mathbf{p}^{\perp}$, the computation of admissible directions $\mathbf{v}$ proceeds as follows.

Let $\mathbf{v}$, $\|\mathbf{v}\| = 1$ be expressed in the principal frame $\{\mathbf{d}_1,\mathbf{d}_2,\mathbf{n}\}$ and let c be the parameter to control the slope of l with respect to Φ, defined as $c = v_3$, $c \in [0,1]$. That is, $c = 0$ corresponds to lines parallel to Φ while $c = 1$ is the case when $l = \mathbf{p}\mathbf{p}^{\perp}$.

The Hessian of (8) is $\mathrm{diag}(\frac{1}{h-\rho_1}, \frac{1}{h-\rho_2}, 0)$, and therefore (3) becomes

$$\frac{v_1^2}{h-\rho_1} + \frac{v_2^2}{h-\rho_2} = 0, \tag{9}$$

which together with

$$v_1^2 + v_2^2 = 1 - c^2 \tag{10}$$

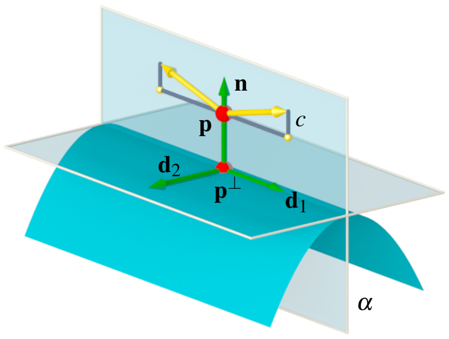


Figure 7: In the limit case when $\mathbf{p}^{\perp}$ is a parabolic point on Φ ($\rho_1 = \infty$), Eq. (11) gives only two admissible directions: $[\pm\sqrt{1-c^2}, 0, c]^{\mathrm{T}}$ (yellow) for fixed c. These directions lie in the normal plane $\alpha = \{\mathbf{p}^{\perp}, \mathbf{d}_1, \mathbf{n}\}$.

gives

$$\mathbf{v}(h,\rho_1,\rho_2,c) = \begin{pmatrix} \pm\sqrt{1-c^2}\sqrt{\frac{h-\rho_1}{\rho_2-\rho_1}} \\ \pm\sqrt{1-c^2}\sqrt{\frac{h-\rho_2}{\rho_1-\rho_2}} \\ c \end{pmatrix}. \tag{11}$$

That is, the admissible directions at $\mathbf{p}$ depend on the second order approximation of the surface Φ at its footpoint $\mathbf{p}^{\perp}$ (parameters ρ_1 and ρ_2), the distance h between $\mathbf{p}$ and $\mathbf{p}^{\perp}$, and the slope parameter c.

Observe that for $c = 0$, the four admissible directions in (11) degenerate to two: the asymptotic directions on Φ^{off}. If additionally $h = 0$, these directions are the asymptotic directions on Φ. If $\mathbf{p}^{\perp}$ is a hyperbolic point on Φ, Eq. (11) gives in general at most four real admissible directions at $\mathbf{p}$; these vectors lie in the spans of the normal planes α_1 and α_2, see Fig. 6. As one osculating radius increases, e.g., $\rho_1 \to \infty$, the two planes close their opening angle and become one in the limit (parabolic point), see Fig. 7.

Remark 1. Note that while hyperbolic points $\mathbf{p}^{\perp}$ on Φ define good local neighborhoods of Φ where Eq.(11) admits real solutions, there exist admissible directions also in the case when $\mathbf{p}^{\perp}$ is elliptic. In such a case $\rho_1\rho_2 > 0$ and, assuming $\rho_1 < \rho_2$, $\mathbf{p}$ must lie on the surface normal of Φ at $\mathbf{p}^{\perp}$ between the osculating centers, i.e., $h \in [\rho_1, \rho_2]$.

4.2. Variational shape approximation

We use variational shape approximation (VSA) from [7] to segment the reference geometry. Such a segmentation is based on a $\mathscr{L}^{2,1}$ metric that uses the normal deviation between the input surface and its polygonal approximation. We recall

$$\mathscr{L}^{2,1}(R_i, P_i) = \iint\limits_{x\in R_i} \|\mathbf{n}(x) - \mathbf{n}_i\|^2 \mathrm{d}x, \tag{12}$$

where R_i is the segmentation of Φ and P_i are so called planar *proxies*. These proxies $P_i = (\mathbf{p}_i, \mathbf{n}_i)$, i.e. pairs of points and normal vectors, are the unknowns in the variational formulation that seeks the segmentation of Φ that minimizes (12), see [7]. A segmentation of an exact envelope is shown in Fig. 8.

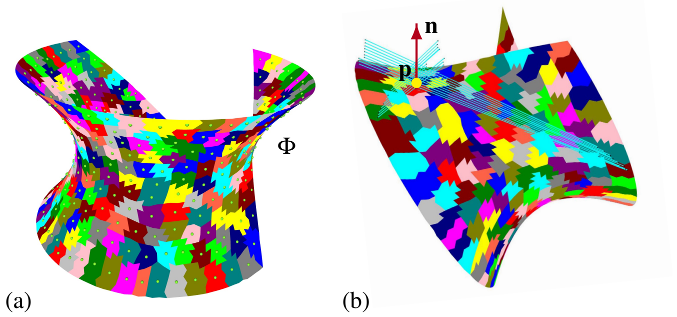


Figure 8: (a) A segmentation of Φ using VSA [7] with $n = 300$ proxies. Each polygonal segment (color coded) corresponds to one proxy; the barycenters of polygons are depicted as green dots. VSA of Φ offers a favorable sampling of the surrounding 3D space as the error metric (12) considers deviation of normal directions. (b) At each barycenter $\mathbf{p}$, we sample points along the surface normals to compute the candidate lines (cyan).

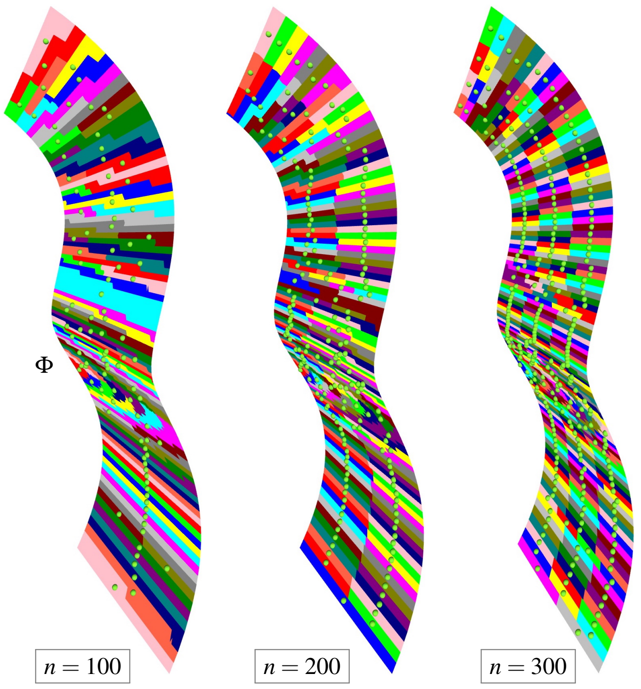


Figure 9: VSA of an exact envelope Φ, that was generated by a special motion where the tool and Φ are in a line contact. Observe that the VSA segmentation yields narrow rectangles that follow the direction of characteristics (rulings) on Φ and, therefore, serves also as a detector of these singular cases. Results for three particular numbers of proxies are shown.

Moreover, VSA helps us to detect singular parts of Φ that are developable surfaces. In such a case, the characteristic *ch* is a straight line, and the normal vector of Φ remains constant along *ch*. This results in low values of the $\mathscr{L}^{2,1}$ functional along *ch*. In the context of VSA, this scenario corresponds to a segmentation of Φ into narrow rectangles with the dominant edge aligned with *ch* as seen in Fig. 9. Such situations with linear characteristics need to be detected and avoided as the conical generator is not unique and any line that lies in the normal plane of Φ and contains *ch* is a good candidate.

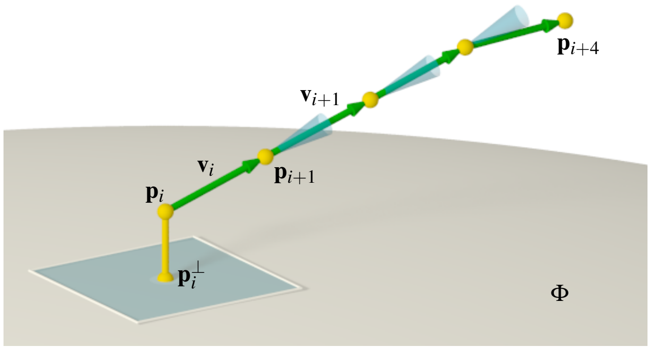


Figure 10: Computing integral curves. Starting at $\mathbf{p}_i$ in an admissible direction $\mathbf{v}_i$, the next iteration point is defiend as $\mathbf{p}_{i+1} := \mathbf{p}_i + \delta\mathbf{v}_i$. The admissible vectors (green) represent directions in which the surface distance d changes linearly (note several admissible directions exist at each point, see also Fig. 6). The next admissible direction is chosen such that the integral curve is as straight as possible. The numerical threshold is governed by an angular deviation between two consecutive admissible vectors (visualized as transparent cones). Three succesful integration steps are shown.

4.3. Candidate lines

Having the VSA segmentation of Φ, we sample the points along the surface normals passing through the barycenters of the VSA polygons. At each point on a surface normal, we compute the admissible directions $\mathbf{v}$ that satisfy (2), see also Fig. 6. This yields an initial value problem and its solutions are sets of space curves shown in Fig. 8(b). We recall that these curves have the property that the distance from Φ varies linearly.

Note that these integral curves are not, in general, straight lines. Among the integral curves, we seek linear segments. Starting at an initial point $\mathbf{p}_i$ in an initial admissible directional vector $\mathbf{v}_i$, see Fig. 10, we integrate the multi-valued vector field, looking for a consecutive admissible vector $\mathbf{v}_{i+1}$ that satisfies

$$\mathrm{angle}(\mathbf{v}_i, \mathbf{v}_{i+1}) < \varepsilon_{linear}, \tag{13}$$

where the parameter ε_{linear} directly controls the straightness. If not stated differently in the examples section, the default value is set $\varepsilon_{linear} = 2^\circ$. The integration terminates once (13) is violated. The result is a nearly straight curve that is approximated by a single linear segment, see e.g. [10, 23, 30] for curve fitting algorithms.

4.4. Movability analysis

So far, we have studied only metric relations between a line and a surface, i.e., we have not imposed the movability constraint yet. Observe that we aim to approximate Φ by an envelope of a conical tool and, therefore, the cone must be tangentially movable along Φ. We use the movability constraint now to filter out those lines that do not admit a rigid body motion along Φ.

We follow the prior work on movability analysis of general surfaces of revolution along free-form surfaces [3], and define an objective function expressing the tangential movability as

$$F(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} \langle \mathbf{v}_i, \frac{\mathbf{p}_i - \mathbf{p}_i^\perp}{\|\mathbf{p}_i - \mathbf{p}_i^\perp\|} \rangle^2 = \mathbf{x}\mathbf{A}\mathbf{x}^\mathrm{T} \to \min, \tag{14}$$

with the constraint

$$\mathbf{v}_\mathbf{m}\mathbf{v}_\mathbf{m}^\mathrm{T} = 1, \tag{15}$$

where n is the number of sampled points $\mathbf{p}_i$ on l, $\mathbf{p}_i^\perp$ are their orthogonal projections onto Φ, $\mathbf{v}_i$ are the velocity vectors associated to $\mathbf{p}_i$, and $\mathbf{v}_\mathbf{m}$ is the velocity at the midpoint $\mathbf{m}$ of l. The unknown $\mathbf{x}$ is a 5-dimensional vector that corresponds to an infinitesimal rigid body motion (a vector field that preserves length of l up to the first order), see e.g. Sections 3 and 4.1 in [3] for a detailed explanation.

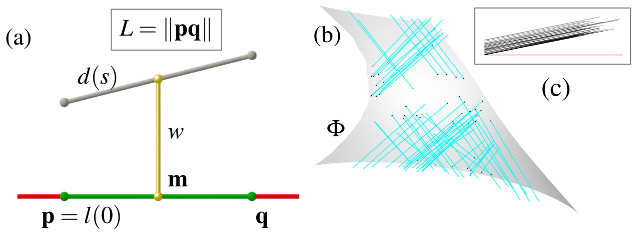


Figure 11: Distance function. (a) A conical axis $l(s) = \mathbf{pq}$, $s \in [0,1]$, L being its length and $\mathbf{m}$ its midpoint, $\mathbf{m} = l(\frac{1}{2})$, possesses a linear distance function $d(s)$ from Φ. The parameter c controls the slope of l with respect to Φ, see also Eq. (10) and Fig. 6, and therefore $c = (d(1) - d(0))/L$. (b) A family of candidate lines of the same parameter c. That is, their distance functions to Φ possess the same slope (c).

The objective function (14) is quadratic and has a unique non-trivial minimizer $\mathbf{x}^*$ that corresponds to the best possible instantaneous motion that moves the cone, l being its axis, as tangentially as possible along Φ. We evaluate F at $\mathbf{x}^*$ to define the *gliding energy* F_l. We recall that when Φ is an exact envelope, $\Phi \equiv \Omega$, $F_l \equiv 0$ at every time instant and, therefore, we look for lines with low values of F_l.

At this point, our algorithm returns a set of straight lines that correspond to cones that are tangentially movable along Φ where this movability is measured in terms of the gliding energy F_l. Straightness is controlled by ε_{linear}, Eq. (13), and movability is controlled by the upper bound $\varepsilon_{gliding}$ on the gliding energy F_l. The default value is set $\varepsilon_{gliding} = 0.005$. These straight lines all correspond to cutters with the same slope, yet these curves may have different lengths and mean of their linear distance functions, see Fig. 11(c).

4.5. Specific conical cutter

The conical cutter enters our algorithm as an input parameter since, typically, a set of milling tools is known a-priori the milling process. We now incorporate the three parameters that determine the specific truncated cone: the slope c, width w, and length L, see Fig. 11. Defining the width of the tool as the value of the distance function at the midpoint $\mathbf{m}$, see Fig. 11(a), our algorithm extracts lines that correspond to a particular cutter. Distance functions of a set of lines with the same slope are shown in Fig. 11(c).

Fig. 12 shows an example of movable lines as a function of the cutter width w and slope c. Our analysis reveals the truncated cone parameters that are the most convenient for a particular general free-form surface. At this point, our algorithm returns a set of line segments with *equal lengths* that correspond

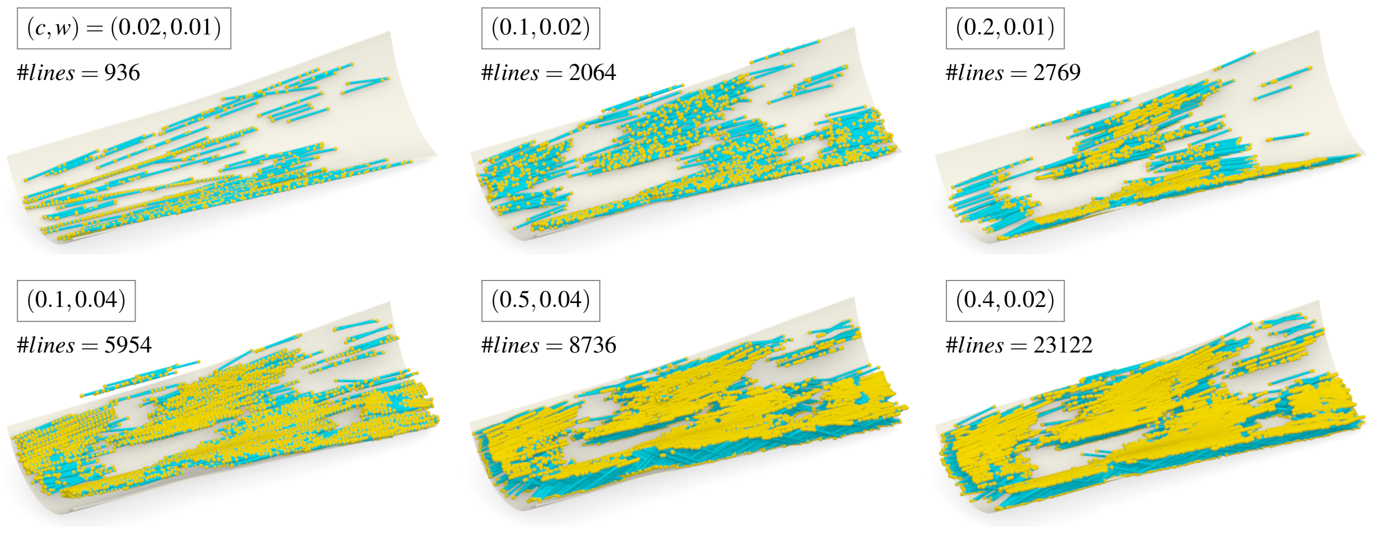


Figure 12: Families of the candidate lines as a function of the shape of the conical milling tool. Straight line segments (blue) of constant length $L = 0.1$ are shown; the line endpoints are highlighted in yellow. Two parameters, the slope c and width w, that control the shape of the truncated cone vary. For each family, the number of lines is displayed, indicating $(c,w) = (0.4, 0.02)$ as the most preferable cone for this specific free-form surface.

to discrete positions in 3D of a specific truncated cone (slope, width, and length) that is tangentially movable along Φ. The distance function $d(s)$ gives us the range $[d(0), d(1)]$ for sampling the points on the surface normals, see Section 4.3.

4.6. Line clustering and ordering

A set of candidate lines of the same length is further processed to extract subsets that correspond to *ordered* positions of a rigid conical cutter. In particular, this requires clustering lines to groups that represent a single approximation patch, and ordering the lines to define a smooth and fair initial ruled surface. We now describe these steps in detail.

Line proximity. In order to cluster nearby lines to groups, we introduce a *line proximity measure* that aims at categorizing lines according to proximity of the underlying conical tools. Let $\mathbf{m}_i$, $\mathbf{v}_i$, and $\mathbf{w}_i$ be in turn the line midpoint, unit directional vector, and unit vector pointing to Φ, i.e., $\mathbf{w}_i = (\mathbf{m}_i^{\perp} - \mathbf{m}_i)/w$, see Fig. 13. We define the proximity measure between two lines as

$$P(l_1, l_2) = \|\mathbf{m}_1 - \mathbf{m}_2\|^2 + \omega_1 \|\mathbf{v}_1 - \mathbf{v}_2\|^2 + \omega_2 \|\mathbf{w}_1 - \mathbf{w}_2\|^2, \quad (16)$$

where the weight ω_1 votes for parallel lines and ω_2 seeks motion with small deviation of the footpoint (characteristic).

k-means clustering. We perform k-means clustering sub-routine based on the measure (16) to reduce the number of candidate lines. Since this set may contain hundreds to thousands of lines, see Fig. 12, the clustering serves as a filter to obtain only moderate number of lines that can be quickly grouped and ordered.

Having n movable lines and an integer k that defines the desired number of clusters, the clustering problem seeks k lines (called centers) such that the mean square distance (16) between centers and the cluster elements (other lines) is minimized. We use the k-means clustering algorithm of Kanugo et al. [16] in our implementation, see Fig. 14.

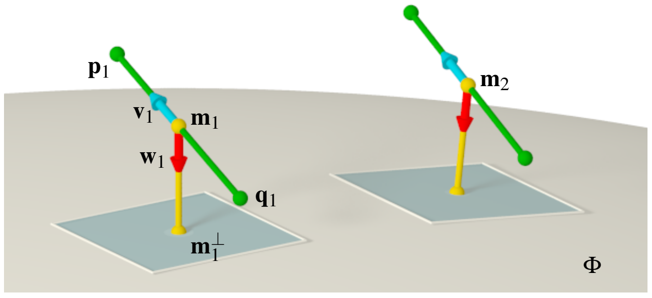


Figure 13: Entities appearing in the proximity measure (16). The proximity reflects the distance deviation between midpoints $\mathbf{m}_i$, directional vectors $\mathbf{v}_i$, and projection directions $\mathbf{w}_i$, $i = 1, 2$.

Line ordering. To order lines into meaningful sequences, see Fig. 14(d), we use again the VSA segmentation introduced in Section 4.2. Let l_i be a line from the set of candidate lines $\mathscr{M}$. We first detect all VSA-segments that contain footpoints of l_i. In our discrete setup, l_i is sampled by $n = 20$ points $\mathbf{p}_i$. This projection defines a part of Φ that can be well approximated by a single cone. Our aim is to approximate large portions of Φ by finding recursively lines from the set $\mathscr{M}$ that cover neighboring VSA-segments.

We use the VSA datastructure which, for each VSA-segment, stores the information of its neighboring patches. Let S_i be a set of patches associated to l_i, i.e., set of segments that contain footpoints of l_i. We say patch P is a *neighboring* VSA-segment of S_i if it shares a common edge with S_i. We denote by N_i the set of neighboring patches of S_i, see Fig. 15. A line $l^{\star}$ is said to be a neighbor of l_i, if one of its footpoints belongs to N_i.

Once a line l is provided by the set of its neighbors, we rank them according to (16) and parse this ranked list to select a line that admits motion which satisfies

$$\text{angle}(\mathbf{v}_i, \mathbf{u}_i) > \varepsilon_{shear}, \quad (17)$$

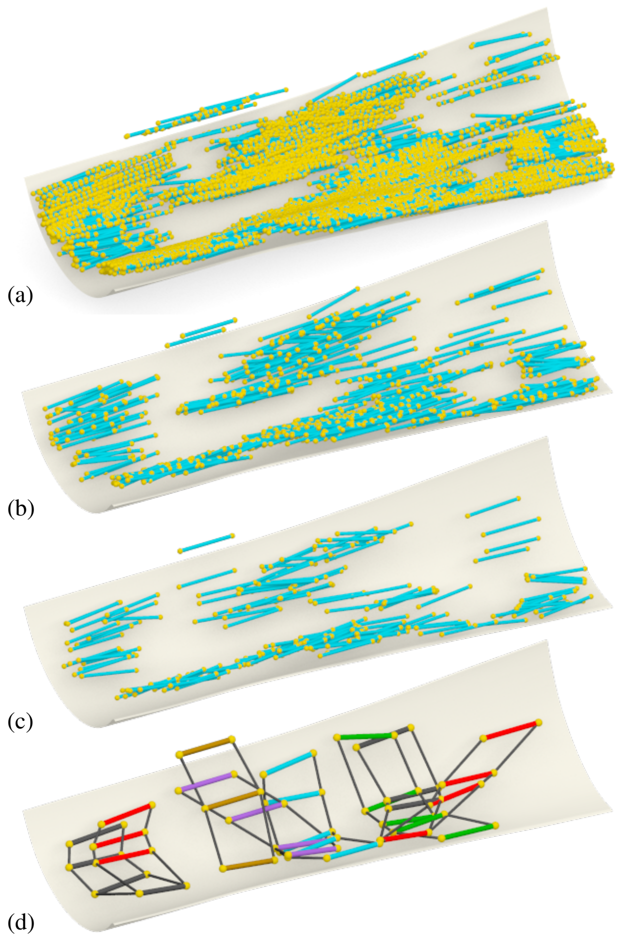


Figure 14: Clustering and ordering. (a-c) Two levels of line filtering based on k-means clustering with the line proximity measure (16) are shown: the initial number $n_0 \cong 5K$ of candidate lines (a) is in turn reduced to (b) $n_1 = 500$ and (c) $n_2 = 150$. (d) The lines are grouped (color-coded) and ordered (thin black) to form initial discrete ruled surfaces.

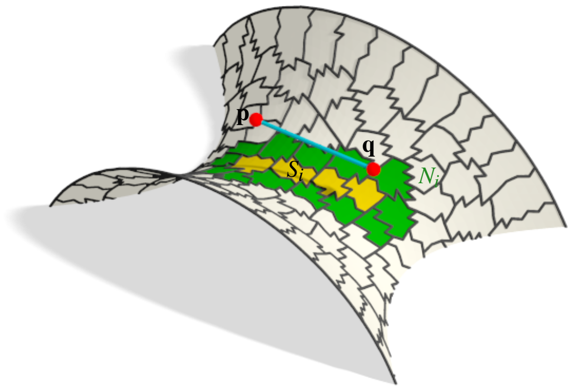


Figure 15: VSA-neighbors. A line $l_i = \mathbf{pq}$ is sampled and the union of VSA-segments containing their footpoints defines a set S_i (yellow). A set of all neighboring VSA-segments of S_i, N_i, is shown in green.

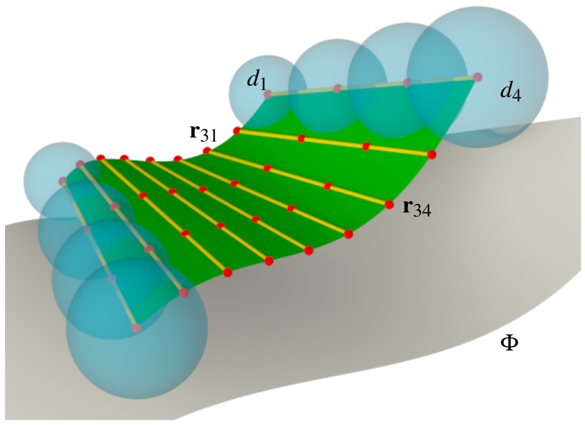


Figure 16: Global optimization. The ruled surface (green) is sampled by $m \times n$ points $\mathbf{r}_{ij}$ (here $m = 9$, $n = 4$). The prescribed distances $d_1, \ldots, d_4$ (transparent spheres) along the rulings change linearly and are determined by the meridian of the cone. The actual distances are obtained by computing the footpoints of $\mathbf{r}_{ij}$ on Φ.

where $\mathbf{u}_i$ is the unit vector determined by the midpoints, i.e., $\mathbf{u}_i = (\mathbf{m}_{i+1} - \mathbf{m}_i)/\|\mathbf{m}_{i+1} - \mathbf{m}_i\|$, see Fig. 13. This requirement eliminates the neighboring lines that would generate a shear motion, i.e., the line would move in the direction of its directional vector.

4.7. *Initial ruled surface and motion optimization*

Our initialization stage returns sets of ordered lines. For each set, a ruled surface that interpolates the ordered lines is constructed

$$R(t,s) = (1-s)\mathbf{p}(t) + s\,\mathbf{q}(t), \quad [t,s] \in [0,1] \times [0,1]. \tag{18}$$

where $\mathbf{p}(t)$ and $\mathbf{q}(t)$ are the two boundary curves, cubic B-splines in our implementation, and the input rulings correspond to uniformly distributed parameter values of t.

We follow the optimization approach introduced in [3] with the additional constraint on the known linear meridian. We uniformly sample the ruled surface R both in t and s parametric directions and obtain $\mathbf{r}_{ij} := R(t_i, s_j)$, $i = 1, \ldots, m$, $j = 1, \ldots, n$. At every time instant t, the distance function between the line and surface must obey the linear distance function $d(s)$, see Fig. 16. The tangential movability objective is formulated as a minimization problem

$$F_{\text{prox}}(\mathbf{p},\mathbf{q}) = \frac{1}{mn} \sum_{j=1}^{n} \sum_{i=1}^{m} (\text{dist}(\mathbf{r}_{ij}, \Phi) - d_j)^2 \to \min \tag{19}$$

subject to the rigidity constraints

$$F_{\text{rigid}}(\mathbf{p},\mathbf{q}) = \langle \mathbf{p}(t_i) - \mathbf{q}(t_i), \mathbf{p}(t_i) - \mathbf{q}(t_i) \rangle - L^2 = 0, \tag{20}$$

where $\text{dist}(,)$ is the actual point-surface distance and d_j, $j = 1, \ldots, n$ are the samples of the known distance function $d(s)$ in Fig. 11(a). The unknowns in the minimization are the control points of the two B-spline curves $\mathbf{p}(t)$ and $\mathbf{q}(t)$.

We further denote by $\mathbf{r}_{ij}^{\perp}$ the footpoints on Φ, and $\mathbf{n}_{ij}$ the unit normals at $\mathbf{r}_{ij}^{\perp}$ oriented towards $\mathbf{r}_{ij}$. The point-surface proximity constraint then reads as

$$\begin{aligned} F_{\text{point}}(\mathbf{p},\mathbf{q}) &= \tfrac{1}{mn} \sum_{j=1}^{n} \sum_{i=1}^{m} \|\mathbf{r}_{ij} - (\mathbf{r}_{ij}^{\perp} + d_j \mathbf{n}_{ij})\|^2, \\ F_{\text{plane}}(\mathbf{p},\mathbf{q}) &= \tfrac{1}{mn} \sum_{j=1}^{n} \sum_{i=1}^{m} (\langle \mathbf{r}_{ij} - \mathbf{r}_{ij}^{\perp}, \mathbf{n}_{ij} \rangle - d_j)^2, \end{aligned} \tag{21}$$

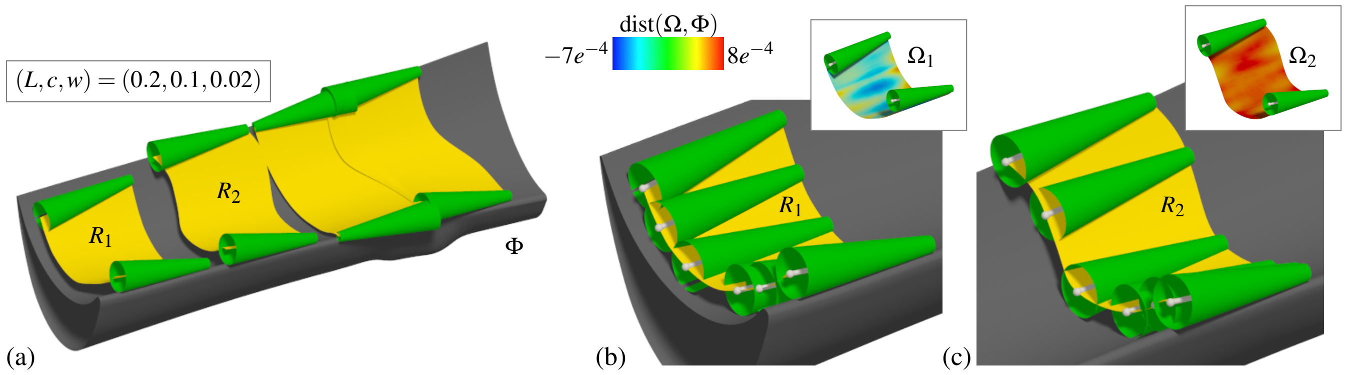


Figure 17: Approximation of a general free-form surface by conical envelopes. (a) The input surface Φ is approximated by four conical envelopes of a specific conical tool (determined by (L,c,w)). The ruled surfaces (yellow) represent the motions of the conical axis. (b,c) A zoom-in on the regions of R_1 and R_2, respectively. The framed figures show the corresponding envelopes Ω_i, $i=1,2$ color-coded by the signed error from the input surface Φ.

which correspond to a point-point and point-plane distance components, respectively.

To achieve a fair motion, we aim at fairness of the two boundary curves, expressed by

$$\begin{aligned} F_{\text{fair}}(\mathbf{p},\mathbf{q}) &= \frac{1}{m}\sum_{i=2}^{m-1}(\mathbf{p}(t_{i-1})-2\mathbf{p}(t_i)+\mathbf{p}(t_{i+1}))^2 \\ &+ \frac{1}{m}\sum_{i=2}^{m-1}(\mathbf{q}(t_{i-1})-2\mathbf{q}(t_i)+\mathbf{q}(t_{i+1}))^2. \end{aligned} \tag{22}$$

The final objective function becomes

$$\begin{aligned} F_{\text{approx}}(\mathbf{p},\mathbf{q}) &= \mu_1 F_{\text{plane}}(\mathbf{p},\mathbf{q}) + \mu_2 F_{\text{fair}}(\mathbf{p},\mathbf{q}) \\ &+ \mu_3 F_{\text{point}}(\mathbf{p},\mathbf{q}) + \mu_4 F_{\text{rigid}}(\mathbf{p},\mathbf{q}), \end{aligned} \tag{23}$$

adding the rulings rigidity (20) as a soft-constraint. The optimization problem is solved using the Gauss-Newton method in all examples shown in the paper. If not stated differently in Section 5, we use the default values $\mu_1=1$, $\mu_2=\mu_4=0.1$, and $\mu_3=0.001$.

5. Results, discussion, and limitations

Results obtained by our algorithm when applied to a general free-form surface are shown in Fig. 17. This CAD geometry is a benchmark industrial model for toolpath generation and material removal simulation algorithms. The parameters of the truncated cone are $L=0.2$, $c=0.1$, $w=0.02$ for a reference surface with normalized bounding box. The color coding represents the one-sided signed error ε between the envelope (Ω) and the designed surface (Φ), i.e.,

$$\varepsilon = \min_{i,j}(\text{dist}(\mathbf{r}_{ij},\Phi)-d_j), \tag{24}$$

measured over a discrete set of samples $\mathbf{r}_{ij}$ of the ruled surface R. Observe that more than 75% of the surface area is covered by only four large conical envelopes. We emphasize that all these four patches correspond to a single milling tool which is a considerable improvement, e.g., to [3, Figs. 14 and 18], where different tools are needed for each large patch. This fact is a significant advantage as it avoids the milling tool exchange that is typically slow and involves human interaction. Fig. 18 shows results on another industrial dataset, a blade of an impeller. Observe that our solutions meet fine numerical threshold that for a blade of a size of $10\,cm$ corresponds to the absolute machining error $0.04-0.06\,mm$. Sets of large envelopes (envelope area $>5\%$ of the surface area) that cover the same surface for two additional cone parameters are shown in Fig. 19.

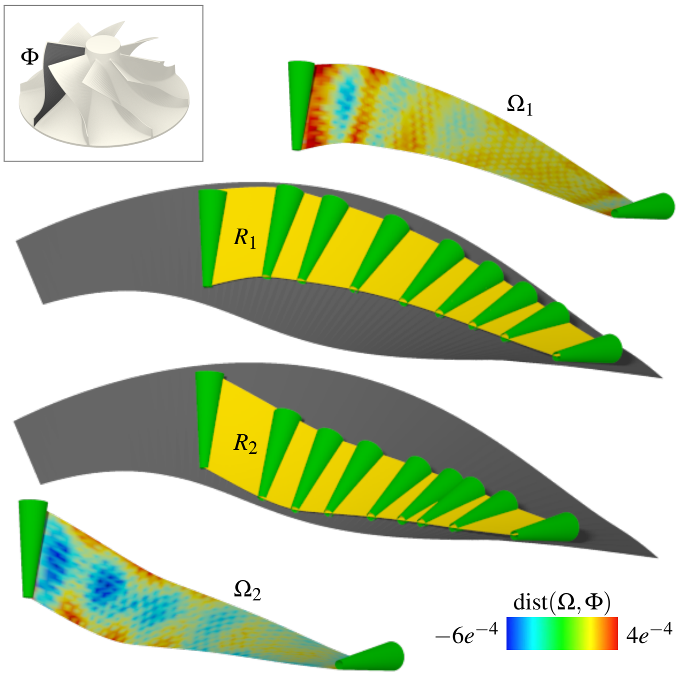
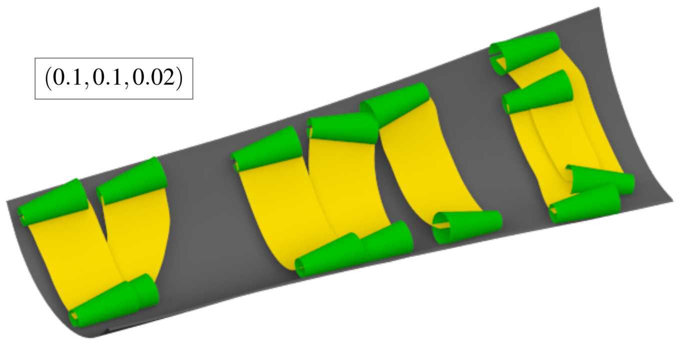


Figure 18: Impeller. Two largest motions of a conical cutter $(L,c,w)=(0.2,0.1,0.02)$ detected by our algorithm when applied to one blade of the impeller (framed). The envelopes are color-coded by the signed distance error (24) where the blue areas indicate overcutting.

A synthetic example where the input surface is an exact envelope is shown in Fig. 20. The envelope corresponds to a rotational motion where the rotational and conical axes are skew (non-singular case). The exact solution consists of two one-parameter families of straight lines that are correctly detected by our algorithm. Fig. 20 further shows that deviating the generating cone either in the c- or w-direction reduces the number



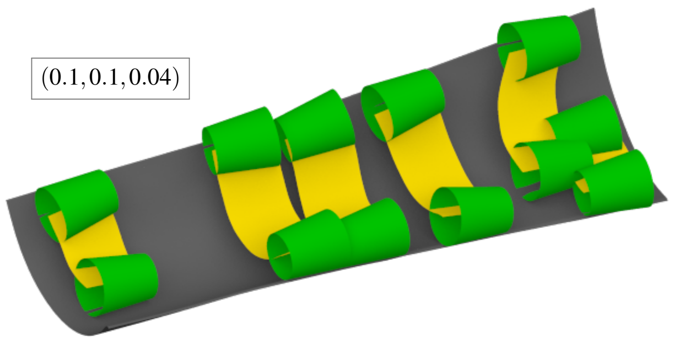


Figure 19: Coverage of the free-form surface shown in Fig. 17 by envelope patches for other two input parameters (L, c, w). For each patch, we show the initial and final position of the conical cutter (green) and the trajectory of the conical axis (yellow). All envelopes meet the error threshold $\varepsilon = 5e^{-4}$, see (24), measured at 3000 points, uniformly sampled in the parameter domain of the ruled surface.

of candidate lines.

All parameters and numerical thresholds of our algorithm are shown in Table 1. For all the examples shown in the paper, the total computation takes few minutes. For example, for the surface shown in Fig. 17, the computational load is: VSA segmentation and mesh preprocessing (13s), candidate lines (16s), movability analysis (7s), k-means clustering and line ordering (4s), and optimization (cca 40s per patch) on a laptop with 16G RAM memory, 4-core CPU, 2.2 GHz. The optimization parameters were set equally as in [3]. The number of VSA clusters was set based on the experiment showed in Fig. 9 that aimed at finding sufficient number of clusters to detect singular envelopes (developable patches). The numbers of samples on the ruled surface were set experimentally by roughly requiring the same distribution of sampling points in s- and t-directions. The total number of sampled points were desired to be moderate (few thousands) to keep the total computation times within a few minutes.

Our results approximate the input geometry within tolerances that meet the fine industrial requirements. In our examples, see e.g. Figs. 17 and 19, we show the largest patches since our algorithm is intended as a finalizing, scallop-free, stage of the milling process. The output of our algorithm, however, is an *atlas* of patches of various patch sizes and milling directions, but an automatic selection of a subset of these patches is not a part of this work. One may ask how to automatically select e.g., a smallest subset of patches that maximizes the coverage of Φ, but such an extraction seems to lead to an NP hard problem. Another limitation of our work stems in the fact that the algorithm does not guarantee to *fully* cover Φ. To mill the whole surface, our approach must be complemented by some other method, e.g., [8], to finalize the remaining residual parts of the surface.

6. Conclusion and Future Work

We have introduced an automatic algorithm that, in the context of CNC machining, initializes trajectories of conical cutters of given slopes and sizes such that their conical envelopes approximate well the input free-form surface. Our approach exploits multi-valued vector fields that correspond to directions in which the point-surface distance changes linearly according to the slope of the prescribed conical tool. Integrating these vector fields gives rise to integral curves, and among them, straight segments are detected. Linear segments that additionally admit tangential motion of the associated cutter along the surface form a set of candidate lines. These lines are sequentially clustered and ordered and their sequences define ruled surface patches that correspond to initial motions of the conical milling tool. Optimization is then applied to globally minimize the approximation error.

We have tested our algorithm against benchmark industrial datasets, as well as against exact envelopes with known solutions. Our algorithm recovers the exact solutions and for general free-form surfaces that are not exact envelopes returns approximate patches within fine tolerances. The algorithm requires several input parameters, e.g. the number of samples per ruling, but is fully automatic in the sense that it does not require intervention of the user to indicate the initial trajectory of the milling tool. We believe this automation is a significant step forward in the state-of-the-art milling algorithms as detecting large patches that can be milled by a single sweep may considerably reduce the total execution times.

Our future research aims at developing computer-aided modeling tools that will consider the manufacturing process directly in the modeling stage. In such a modeling interface, for example, a modification of a control point of a free-form surface would evoke an optimization based routine that would update both the milling tool and its trajectory to best approximate the designers' intentions by a surface that is by-definition manufacturable.

Acknowledgments. We thank Denys Plakhotnik for providing the datasets for Fig. 17 and 18. This research has been supported by the European Community's 7th Framework Programme under grant agreement 286426 (*GEMS*). The first author has been partially supported by The National Natural Science Foundation of China (Grant No. 61672187) and Natural Scientific Research Innovation Foundation in Harbin Institute of Technology (HIT.NSRIF.201711). The second author has been partially supported by the Basque Government through the BERC 2014-2017 program and by Spanish Ministry of Economy and

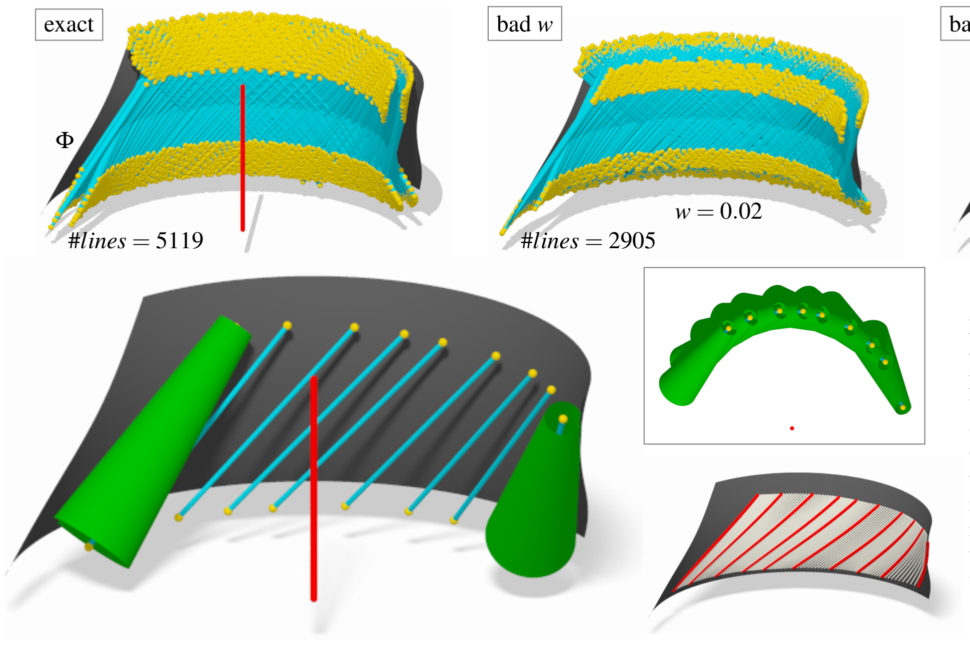


Figure 20: Exact envelope reconstruction. An exact envelope Φ was generated by applying a pure rotation (red axis) to a cone. In such a situation, there exist two one-parameter families of generating lines. Top row from left to right: Candidate lines detected by our algorithm for the exact solution $(c,w) = (0.1, 0.06)$, distorted in c-direction, and w-direction. Bottom left: final motion recovered by our algorithm and the top view (framed). Bottom right: The final envelope and the contact characteristics (red).

Table 1: Default parameters and thresholds of our algorithm for input surfaces with normalized bounding box. In the *'VSA'* columns, $\#n_{VSA}$ is the number of proxies, and $\#n_{nor}$ is the number of sampled points on the surface normal at each proxy barycenter. *'Lines & Movability'* columns contain the integration stepsize δ, see Fig. 10, angular deviation ε_{linear} that controls the line straightness and $\varepsilon_{gliding}$ that is a threshold on the gliding energy F_l in Section 4.4. In the *'Clustering & Ordering'* columns, k is the number of the k-means clusters, ω_1 and ω_2 are the weights in Eq. (16), and ε_{shear} is the angular threshold in (17) that prevents shear motions. In the *'Sampling'* columns, m and n are the number of samples on the ruled surface in the time and the ruling parametric directions, respectively. In the last category, we show the optimization parameters that affect Eq. (23).

VSA		Lines & Movability			Clustering & Ordering				Sampling		Optimization			
$\#n_{VSA}$	$\#n_{nor}$	δ	ε_{linear}	$\varepsilon_{gliding}$	k	ω_1	ω_2	ε_{shear}	m	n	μ_1	μ_2	μ_3	μ_4
400	20	0.01	2°	0.005	200	1	1	40°	100	30	1	0.1	0.001	0.1

Competitiveness MINECO: BCAM Severo Ochoa excellence accreditation SEV-2013-0323.

7. References

References

[1] M. Bartoň, H. Pottmann, and J. Wallner. Detection and reconstruction of freeform sweeps. *Computer Graphics Forum*, 33(2):23–32, 2014.

[2] S. Bedi, S. Mann, and C. Menzel. Flank milling with flat end milling cutters. *CAD*, 35(3):293–300, 2003.

[3] Pengbo Bo, Michael Bartoň, Denys Plakhotnik, and Helmut Pottmann. Towards efficient 5-axis flank CNC machining of free-form surfaces via fitting envelopes of surfaces of revolution. *Computer-Aided Design*, 79:1–11, 2016.

[4] C Brecher, M Weck, M Winterschladen, S Lange, O Wetter, T Pfeifer, D Dörner, E Brinksmeier, and L Autschbach. Manufacturing of free-form surfaces in optical quality using an integrated nurbs data interface. In *Proceedings of the ASPE Winter Topical Meeting*, 2004.

[5] Christian Brecher and O Wetter. Manufacturing of free-form surfaces using a fast tool servo (fts) and an online trajectory generator. In *Proc. ASPE Winter Topical Meeting*, 2005.

[6] Cristiano Ceccato, Lars Hesselgren, Mark Pauly, Helmut Pottmann, and Johannes Wallner. *Advances in Architectural Geometry 2010*. Springer Vienna Architecture, 2010.

[7] David Cohen-Steiner, Pierre Alliez, and Mathieu Desbrun. Variational shape approximation. In *ACM Transactions on Graphics (TOG)*, volume 23, pages 905–914. ACM, 2004.

[8] G. Elber and R. Fish. 5-axis freeform surface milling using piecewise rule surface approximation. *ASME Journal of Manufacturing Science and Engineering*, 119(3):383–387, 1997.

[9] J. Fan and A Ball. Flat-end cutter orientation on a quadric in five-axis machining. *Computer-Aided Design*, 53:126–138, 2014.

[10] S. Flöry. Fitting curves and surfaces to point clouds in the presence of obstacles. *Computer Aided Geometric Design*, 26:192–202, 2009.

[11] Natasha Gelfand and Leonidas J Guibas. Shape segmentation using local slippage analysis. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pages 214–223. ACM, 2004.

[12] H. Gong, C. Li-Xin, and L. Jian. Improved positioning of cylindrical cutter for flank milling ruled surfaces. *Computer-Aided Design*, 37:1205–1213, 2005.

[13] H. Gong and N. Wang. Optimize tool paths of flank milling with generic cutters based on approximation using the tool envelope surface. *Computer-Aided Design*, 41(12):981–989, 2009.

[14] R. F. Harik, H. Gong, and A. Bernard. 5-axis flank milling: A state-of-the-art review. *Computer-Aided Design*, 45(3):796–808, 2013.

[15] H. T. Hsieh and C. H. Chu. GPU-based optimization of tool path planning in 5-axis flank milling. In *2010 International Conference on Manufacturing Automation, ICMA 2010*, pages 143–150. IEEE, 2010.

[16] T. Kanungo, D. M Mount, N. Netanyahu, C. Piatko, R. Silverman, and A. Wu. An efficient k-means clustering algorithm: Analysis and implementation. *IEEE transactions on pattern analysis and machine intelligence*, 24(7):881–892, 2002.

[17] C. Li, S. Bedi, and S. Mann. Flank milling of a ruled surface with conical tools – an optimization approach. *Int. J. Adv. Manuf. Technol.*, 29:1115i–1124, 2006.

[18] C. Li, S. Bedi, and S. Mann. Flank millable surface design with conical and barrel tools. *Computer- Aided Design and Applications*, 5:461–470,

2008.

[19] S. X. Li and R. B. Jerard. 5-axis machining of sculptured surfaces with a flat-end cutter. *Computer-Aided Design*, 26(3):165–178, 1994.

[20] Y.A. Lu, Q.Z. Bi, and L.M. Zhu. Five-axis flank milling of impellers: Optimal geometry of a conical tool considering stiffness and geometric constraints. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, 228:1226–1236, 2014.

[21] M. Peternell, H. Pottmann, and B. Ravani. On the computational geometry of ruled surfaces. *Computer-Aided Design*, 31:17–32, 1999.

[22] H. Pottmann and J. Wallner. Approximation algorithms for developable surfaces. *Computer Aided Geometric Design*, 16(6):539–556, 1999.

[23] H. Pottmann, W. Wang, and Y. Liu. Fitting B-spline curves to point clouds by squared distance minimization. Technical report, Technical report, Department of Computer Science, The University of Hong Kong, 2004.

[24] J. Redonnet, W. Rubio, and G. Dessein. Side milling of ruled surfaces; optimum positioning of the milling cutter and calculation of interference. *The International Journal of Advanced Manufacturing Technology*, 14(7):459–465, 1998.

[25] J. Senatore, Y. Landon, and W. Rubio. Analytical estimation of error in flank milling of ruled surfaces. *Computer-Aided Design*, 40:595–603, 2008.

[26] J. Senatore, F. Moniès, and W. Rubio. 5-axis flank milling of sculptured surfaces. In *Machining of Complex Sculptured Surfaces*, pages 33–65. Springer, 2012.

[27] K. Sprott and B. Ravani. Cylindrical milling of ruled surfaces. *Int. J. Adv. Manuf. Technol.*, 38:649–656, 2008.

[28] Chengcheng Tang, Pengbo Bo, Johannes Wallner, and Helmut Pottmann. Interactive design of developable surfaces. *ACM Transactions on Graphics (TOG)*, 35(2):12, 2016.

[29] K. Tang and C.C.L. Wang. Modeling developable folds on a strip. *Journal of Computing and Information Science in Engineering*, 5(1):35–47, 2005.

[30] T. Várady and R. Martin. *Reverse Engineering. In: Handbook of computer aided geometric design*. Elsevier, 2002.

[31] Jan Vršek and Miroslav Lávička. Determining surfaces of revolution from their implicit equations. *Journal of Computational and Applied Mathematics*, 290:125–135, 2015.

[32] C.C.L. Wang and G. Elber. Multi-dimensional dynamic programming in ruled surface fitting. *Computer-Aided Design*, 51:39–49, 2014.

[33] C.C.L. Wang and K. Tang. Optimal boundary triangulations of an interpolating ruled surface. *Journal of Computing and Information Science in Engineering*, 5(4):291–301, 2005.

[34] Kim Y.J., Bartoň M., Elber G., and Pottmann H. Precise gouging-free tool orientations for 5-axis CNC machining. *Computer-Aided Design*, 58:220–229, 2015.

[35] G. Zheng, Q.-Z. Bi, and L.-M. Zhu. Smooth tool path generation for five-axis flank milling using multi-objective programming. In *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, volume 226, pages 247–254, 2012.

[36] L. Zhu, H. Ding, and Y. Xiong. Simultaneous optimization of tool path and shape for five-axis flank milling. *Comput. Aided Des.*, 44:1229–1234, 2012.

[37] L. Zhu, G. Zheng, H. Ding, and Y. Xiong. Global optimization of tool path for five-axis flank milling with a conical cutter. *Computer-Aided Design*, 42(10):903–910, 2010.