Computer-Aided Design 79 (2016) 1–11

Contents lists available at ScienceDirect

Computer-Aided Design

journal homepage: www.elsevier.com/locate/cad

Towards efficient 5-axis flank CNC machining of free-form surfaces via fitting envelopes of surfaces of revolution

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

[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] ModuleWorks GmbH, Henricistr. 50, 52072 Aachen, Germany
[d] Center for Geometry and Computational Design, Vienna University of Technology, Wiedner Hauptstr. 8-10/104, A-1040 Vienna, Austria

ARTICLE INFO

Article history:
Received 18 November 2015
Accepted 8 April 2016

Keywords:
5-axis CNC machining
Flank milling
Free-form surface
Tangential movability
Shape manufacturing

ABSTRACT

We introduce a new method that approximates free-form surfaces by envelopes of one-parameter motions of surfaces of revolution. In the context of 5-axis computer numerically controlled (CNC) machining, we propose a flank machining methodology which is a preferable scallop-free scenario when the milling tool and the machined free-form surface meet tangentially along a smooth curve. We seek both an optimal shape of the milling tool as well as its optimal path in 3D space and propose an optimization based framework where these entities are the unknowns. We propose two initialization strategies where the first one requires a user's intervention only by setting the initial position of the milling tool while the second one enables to prescribe a preferable tool-path. We present several examples showing that the proposed method recovers exact envelopes, including semi-envelopes and incomplete data, and for general free-form objects it detects envelope sub-patches.

1. Introduction & motivation

Free-form, aka sculptured, objects appear in our daily life ranging from small coffee mugs, over car cockpits, to modern free-form buildings. These curved objects are aesthetically pleasing and a result of combined efforts of designers and engineers, the latter being responsible for optimal functionality. The production of free-form shapes often involves computer numerically controlled (CNC) machining, which can follow various strategies of different efficiency. The choice of machining strategy is related to the shape to be produced. To find the most efficient CNC machining strategy for a given free-form object is still an open and very challenging task [1,2].

Our work is a step into this direction. We derive new computational methodology for manufacturing that reduces the time and cost and increases the quality of fabrication processes, particularly for free-form (NURBS) objects. The technology we have in mind is 5-axis CNC machining (milling) because it is the leading manufacturing technology [2]. A revolving milling tool is navigated along a workpiece, removing the undesired material and producing the desired shape. In particular, we consider the case of *flank* milling [3,4], i.e., the case when the milling tool touches the workpiece along a whole curve, in contrast to conventional multi-axis milling, where the designed surface and the milling tool share only a single contact point. Flank machining is more efficient because the machining strips are usually much wider since the cutting strip width is not bounded by the tool's diameter [5]. However, it is a more complicated task to generate toolpaths for flank milling in which the milling tool is aligned to and tangentially movable through the whole design surface without gouging.

Geometrically, the problem considered in this work is to approximate the given free-form object by a set of manufacturable patches. The object is represented by a free-form surface Φ and the patches are required to approximate it within a fine user-predefined tolerance. Each manufacturable patch is an envelope of a one-parameter motion of a surface of revolution. Our goal is to find both the optimal shape of the milling tool (surface of revolution) as well as its motion in 3D.

2. Previous work and contributions

The research presented in this work belongs to the flank category [5] of 5-axis CNC milling, i.e., the milling tool moves

* Corresponding author.
E-mail addresses: pengbo@hitwh.edu.cn (P. Bo), mbarton@bcamath.org (M. Bartoň), denys@moduleworks.com (D. Plakhotnik), pottmann@geometrie.tuwien.ac.at (H. Pottmann).

http://dx.doi.org/10.1016/j.cad.2016.04.004
0010-4485/

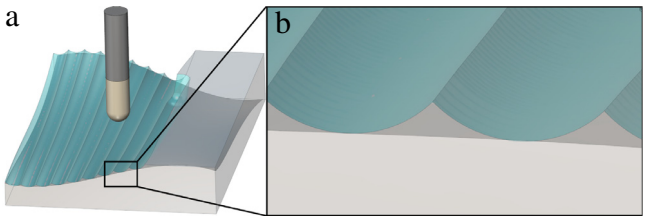

Fig. 1. Scallop cusps. (a) Ball-end milling is shown. A design surface (dark gray) and machined surface (light blue), consisting of several machined strips, are shown. (b) A zoom-in of the machined strips. The intersection of two neighbor strips introduces a sharp edge (cusp) and its distance from the design surface is known as the scallop height. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

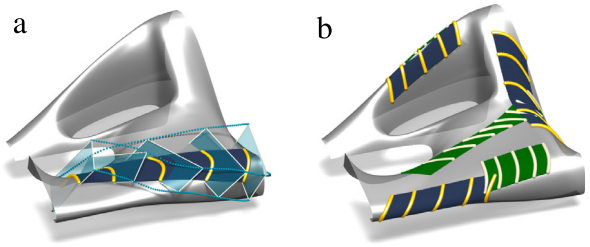

Fig. 2. Approximation of free-form skin of the Heydar Aliev cultural center, Baku, by a set of planar sweeps [29]. (a) For one large detected patch, the corresponding motion of a planar profile is shown. (b) Several larger patches together with their generating planar profiles.

tangentially along the input (designed) free-form surface, staying with it in a tangential contact along a 3D curve.

Typically, the shape and size of the milling tool is given as an input (mostly cylindrical or conical) [2,6]. A lot of research is devoted to ruled surfaces [7–15] and references cited in [5]. Using a truncated conical milling tool, Elber and Fish [7] approximate a general free-form surface using piece-wise ruled surfaces. For a general shape, however, a large number of patches is obtained and each of these ruled patches requires its own conical milling tool. Redonnet et al. [8] consider a cylindrical cutter for machining ruled surfaces and optimize its position such that it possesses a tangential contact to three particular lines: one ruling of the surface and two tangent lines of the rail curves of the ruled surface. This reduces the machining error by order of magnitude when compared with standard two-tangential arrangements. Gong and Wang [12] optimize the tool axis trajectory to minimize the error along the surface normal directions. However, the milling tool is an input as well as the initial axis trajectory (ruled surface). In contrast, in this work, we also initialize the ruled surface and search for the optimal milling tool.

The proximity of the designed and machined surfaces need not to be necessarily the only objective. The smoothness of the motion of the milling tool is equally important for machining efficiency [16–18]. Lavernhe et al. [16] optimize the tool axis orientation to satisfy certain kinematic constraints which results in a higher speed of the tool and consequently in shorter machining time. Pechard et al. [17] minimize the distance between the designed and machined surfaces while preserving the smoothness of the tool's trajectory. Zheng et al. [18] use the strain energy of the tool axis trajectory to describe the geometric smoothness and formulate it as a multi-objective programming problem. All three works, however, define a milling tool as the input and do not consider its optimization.

In the direction of selecting an optimal *general* cutter, to the best of our knowledge, it is very little known so far. Senatore et al. [6] provide analysis with respect to the size of a cylindrical tool. In order to cover large patches, a maximum radius is computed while keeping the predefined geometric error between the machined and designed surface. The work of Zhu et al. [19] is the closest one to our research as it also deals with flank milling and the simultaneous optimization for tool's motion and shape. This approach has been extended in [20], considering additional constraints such as conical tools and their stiffness. In both cases, the primary objective is to improve the machining process by reducing deflection and vibrations of the cutter. The shape of the cutter is optimized towards higher stiffness, while the trajectory surface is computed by interpolating the cutter axes. In our work, we take into account mainly geometric objectives (the forces applied to the cutter are not considered). We present an alternative formulation of the optimization where the cutter and its trajectory are both the unknowns, and a new way for quickly deciding whether a given position of the tool axis possesses a good tangential movability with respect to the reference geometry or not.

Optimization-based approaches that subdivide the design surface into sub-patches have been proposed recently [13,21]. On each such a patch, Liu et al. [21] apply a rank-two tensor field to compute the machining strip width and the optimal machining direction. Wang and Elber [13] use multi-dimensional dynamic programming to find optimal subdivisions and fit a ruled surface to each subdivided patch.

The research proposed here aims at the finishing stage of machining when the motion of the milling tool completes the desired shape of the workpiece. Typically, this is a very time demanding process because small-radii milling tools are used to eliminate or reduce the remaining scallop cusps, see Fig. 1. To optimize performance of this operation, various approaches varying tool orientation have been developed. The idea is to adapt the milling tool, usually a torus or a cylinder, such that the contact circle possesses higher order contact with the surface. This technique is known as *curvature matched machining*, see [22–27] and other references cited in [26]. However, it is possible to consider non-traditional shapes of the milling tool [28,29] to approximate Φ by a smaller number of larger, geometrically simpler, but yet-sufficiently complex patches. Conceptually, our research belongs to this modern family of techniques.

In Fig. 2 we show one of our recent results [29] where the idea of representing a complex shape by several large simpler patches was realized on a large scale, mainly for real architectural models. The patches are kinematic surfaces generated by motions of planar profiles. For purposes of CNC machining, however, we cannot in general move a planar profile so that it carves out a desired shape. We need to work with a rotary cutting tool and thus consider envelopes of rotational surfaces instead of sweeps of planar profiles.

Contributions. We investigate a class of surfaces, namely *envelopes*, generated by one-parameter motions of surfaces of revolution. By definition, these envelopes have a tangential contact with their generating surfaces of revolution at any time instant. In other words, the surface of revolution glides tangentially along its envelope. This fact makes envelopes perfect candidates for flank CNC machining and *thus we propose an algorithm that seeks a set of envelopes that well approximate the input free-form shape.* This is the main contribution of our paper and is presented in Section 4. In particular, we do the following:

- ⋆ Given a line l in space and a design surface Φ, this uniquely defines a rotational surface Ψ with l as its axis, as well as it defines the contact curve between Ψ and Φ. We seek an envelope Ω, that is, Ψ and its rigid body motion, such that Ω well approximates Φ. Therefore a good candidate line l yields a surface Ψ which is tangentially movable along Φ. We consider this tangential motion up to first order and explore the space of lines for those which lead to good tangential movability. Alternatively, a user defines a preferable trajectory of the tool axis.

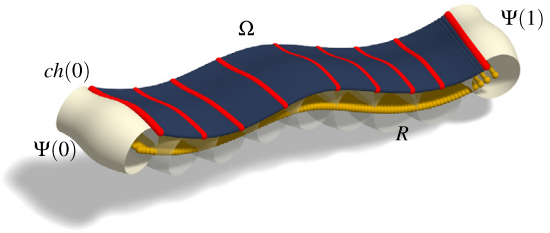

Fig. 3. A one-parameter motion of a surface of revolution. A surface of revolution Ψ moves from its starting position $t = 0$ to the end configuration $t = 1$, generating an envelope Ω (blue). At each time instant t, $\Psi(t)$ touches Ω along a characteristic curve $ch(t)$ (red). The ruled surface R swept by the axis of Ψ is shown in yellow. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

- ⋆ The initial guess described above is optimized towards the proximity objective between the cutter and the designed surface Φ during the whole motion. In this step, both the cutting tool (surface of revolution) and its motion are optimized.
- ⋆ We show that our algorithm reproduces exact envelopes and discuss to what extent the desired shape can deviate from an exact envelope to be still well approximated.
- ⋆ The main goal of this research is to detect large parts of free-form surfaces that are manufacturable by a *single sweep* of a rotary cutter. We emphasize here again that such a patch is scallop-free, cf. Fig. 1, because of the tangential contact between the two surfaces (the cutter and the material block), and therefore is desirable for CNC machining because it does not require any post-process smoothing.

3. Basic facts from geometry and kinematics

We now briefly recall a few preliminaries on surfaces of revolution and their envelopes under a Euclidean rigid body motion. We also discuss the velocity vector fields attached to moving lines.

3.1. Envelopes of surfaces of revolution

Consider a surface of revolution Ψ with an axis l under a one-parameter motion in space. Denote by Ω its *envelope*, see Fig. 3, and consider Ψ as a function of time (or pseudo-time), $\Psi(t)$, $t \in [0, 1]$, $\Psi(0)$ being the start and $\Psi(1)$ the end position, respectively.

The contact curve $ch(t)$ shared by $\Psi(t)$ and Ω is called a *characteristic*. By definition, since Ω envelopes the set of Ψs, Ω and $\Psi(t)$ share also the tangent planes along $ch(t)$. Ω has typically two disconnected components (upper and lower envelopes); we consider only one of its components in our application scenario. Moreover, we also assume that $ch(t)$ associated to the lower (upper) envelope consists of a single continuous branch. As explained later in Remark 1, $ch(t)$ with more components restricts movability and therefore is not desirable for machining considerations.

Remark 1. We recall that *ch* is the locus of footpoints (closest points) of all $\mathbf{p}$, $\mathbf{p} \in l$, on Φ. In the case when *ch* has more than one branch (continuous component) while Ψ moves along Φ, the discontinuity implies the existence of a point $\mathbf{p}_d$, $\mathbf{p}_d \in l$, which has two different closest points on Φ, see Fig. 4. That is, the sphere centered at $\mathbf{p}_d$ touches Φ at these two footpoints which decreases its movability along Φ. Whereas a sphere with a single contact point has two degrees of freedom (DoF) to tangentially glide along Φ, the double tangential contact demands (up to scaling) a unique moving direction. This is a strong movability restriction and, since we look for lines with good movability, these configurations are, if detected, ignored.

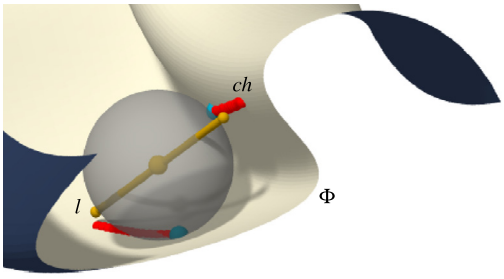

Fig. 4. Double-tangential contact. The footpoints of a line (yellow) form a characteristic (red) which, in complex geometries, may be discontinuous. Such a scenario is inappropriate for machining purposes because the tangential movability of l along the reference geometry Φ is limited by the sphere (transparent) that possess a double-tangential contact (blue) with Φ. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

3.2. Lines in 3D

Exploring the space of surfaces of revolution that possess tangential contact with Φ is equivalent to searching among all possible axes, that is, lines. The space of 3D lines is of dimension four (with the structure of a quadric in 5-dimensional projective space [30]). However, this is the space of *infinite* lines, while here we have to consider also the start and end points of *finite* lines. This results in a 6-dimensional search space. To explore exhaustively this huge space would not be possible, particularly because comparison and ordering of good candidates (surfaces of revolution) grows exponentially with the problem dimension. Therefore, a strategy analogous to our previous work [29, Section 2.1] is not expected to be efficient. Thus, we introduce a different approach which does not require shape matching of the tool profiles.

3.3. Velocity vector fields attached to lines

A rigid body motion in 3D-space has six DoFs. Because we move a surface of revolution Ψ and thus can ignore rotations about its axis l, we only need to consider the motion of the straight line l, which reduces the number of DoFs to five. We consider the velocity vectors of points on a line l which is spanned by points $\mathbf{a}$ and $\mathbf{b}$, see Fig. 6. During the motion, these points run along their trajectories $\mathbf{a}(t)$, $\mathbf{b}(t)$ so that their distance $d = \|\mathbf{a}(t) - \mathbf{b}(t)\|$ remains constant. Differentiating this condition leads to the well known *projection rule* for their velocity vectors $\mathbf{v_a} = \dot{\mathbf{a}}(t)$, $\mathbf{v_b} = \dot{\mathbf{b}}(t)$,

$$\langle \mathbf{v_a}, \mathbf{b} - \mathbf{a} \rangle = \langle \mathbf{v_b}, \mathbf{b} - \mathbf{a} \rangle . \tag{1}$$

The vector field is linear along l, i.e., for any point $\mathbf{p} \in l$, $\mathbf{p} = (1 - s)\mathbf{a} + s\mathbf{b}$, we have

$$\mathbf{v_p} = (1 - s)\mathbf{v_a} + s\,\mathbf{v_b}, \quad s \in [0, 1]. \tag{2}$$

To define the vector field, and therefore to prescribe the instantaneous motion of l, one has to determine five parameters: the three coordinates of the vector at line's start point $\mathbf{v_a} = (v_x^{\mathbf{a}}, v_y^{\mathbf{a}}, v_z^{\mathbf{a}})$ and two additional parameters λ_1 and λ_2 which control the position of $\mathbf{v_b}$, satisfying the projection rule Eq. (1). In particular, we write

$$\mathbf{v_b} = \mathbf{v_a} + \lambda_1 \mathbf{n}_1 + \lambda_2 \mathbf{n}_2, \tag{3}$$

where $\mathbf{n}_1$ and $\mathbf{n}_2$ span the space perpendicular to l, see Fig. 6.

Since any non-zero scalar multiple of the vector field gives the same instantaneous motion of l, we normalize the field so that at the midpoint $\mathbf{m} = (\mathbf{a} + \mathbf{b})/2$ of l we have $\|\mathbf{v_m}\| = 1$. After this normalization, the number of free parameters that control the instantaneous motion of l is four.

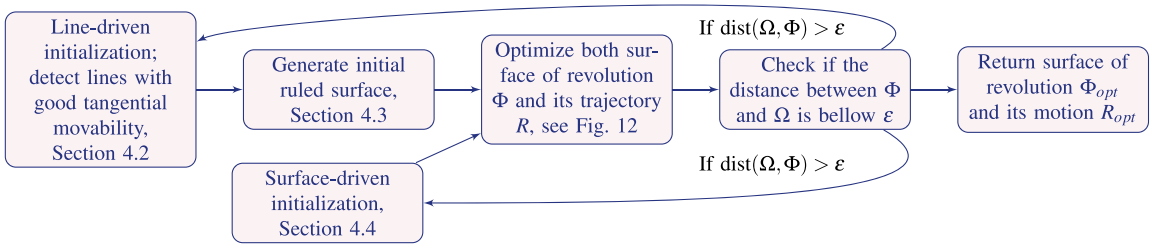

Fig. 5. Algorithm overview.

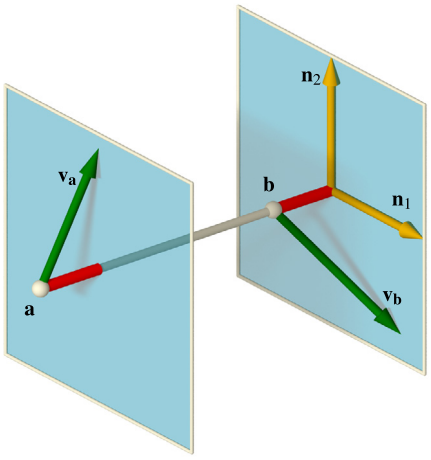

Fig. 6. Projection rule. The instantaneous motion of a line $l = \mathbf{ab}$ is determined by the velocity vectors (green) associated to the endpoints. The oriented projections of $\mathbf{v_a}$ and $\mathbf{v_b}$ onto l are equal (red). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

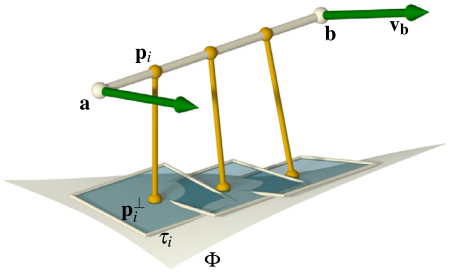

Fig. 7. The reference geometry Φ is approximated by a set of footpoints $\mathbf{p}_i^\perp$ and tangent planes τ_i. A line $l = \mathbf{ab}$ is required to move as parallel as possible to the set of tangent planes. The sought-after vector field (green) is computed from Eq. (5). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

4. Fitting envelopes to free-form shapes

In this section, we show how to detect parts of a free-form surface Φ that can be approximated by an envelope Ω. Our algorithm consists of the following steps, see also Fig. 5 for a detailed overview of the algorithm:

- ⋆ *Line-driven initialization.* We explore the space of lines and quickly detect candidate lines with good local tangential movability with respect to Φ. Consequently, we evolve a candidate line along Φ to generate an initial motion of the axis. The surface swept by the moving axis is a ruled surface R.
- ⋆ *Surface-driven initialization.* Alternatively, we initialize the ruled surface R directly. The reason is to avoid an exhaustive search of lines and to better control the motion.
- ⋆ *Optimization.* Since the meridian (profile) of the cutting tool still varies along R, an optimization scheme is proposed to unify the meridian's shape and to minimize the error between the envelope Ω and the given surface Φ.

4.1. Tangential movability along surface

Given a free-form surface Φ, we seek finite lines that possess the property of local tangential movability along Φ. We call a line l *tangentially movable* if any point $\mathbf{p}_i \in l$ has a velocity vector $\mathbf{v}_i$ which is parallel to the tangent plane of its footpoint $\mathbf{p}_i^\perp$, see Fig. 7.

Given a line l, we have to quickly test how tangentially movable l is with respect to Φ. Denoting by

$$\mathbf{x} := (v_x^{\mathbf{a}}, v_y^{\mathbf{a}}, v_z^{\mathbf{a}}, \lambda_1, \lambda_2) \tag{4}$$

the five parameters that define a velocity field associated with l, we formulate the search for a tangentially movable vector field as a constrained least squares problem,

$$F(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n}\left\langle \mathbf{v}_i, \frac{\mathbf{p}_i - \mathbf{p}_i^\perp}{\|\mathbf{p}_i - \mathbf{p}_i^\perp\|}\right\rangle^2 = \mathbf{x}\mathbf{A}\mathbf{x}^\mathsf{T} \to \min, \tag{5}$$

with the constraint

$$\mathbf{v_m}\mathbf{v_m}^\mathsf{T} = 1, \tag{6}$$

where n is the number of sample points on l. With this setting, the projection rule (1) is automatically satisfied. Note that $\mathbf{v}_i$ is linear in $\mathbf{x}$ and therefore Eq. (5) is a quadratic form in $\mathbf{x}$ with a trivial minimizer $\mathbf{x} = \mathbf{0}$. Constraint (6) is used to demand a non-trivial solution. It is well known that this amounts to the solution of a generalized eigenvalue problem.

Once the best tangential vector field is computed from (5), we take the function value $F(\mathbf{x}^*)$ at the minimizer $\mathbf{x}^*$ as a movability measure of l and denote it by F_l. Observe that for the cases when Φ is an exact envelope, $\Phi \equiv \Omega$, $F_l \equiv 0$ at every time instant. Therefore we search for lines with good movability by looking for low values of F_l.

4.2. User-driven exploration of the space of movable lines

We conduct a user-driven exploration of the space of lines. Recall that the space of finite lines is 6-dimensional (4 DoFs for an infinite line and 2 DoFs for the positions of the endpoints). Typically, there is a preferable region to start the machining process and the user's intervention at this stage is usual. The user sets an initial finite line by selecting its endpoints. By sampling and computing the corresponding footpoints on Φ, see Fig. 7, the line defines a characteristic and simultaneously Ψ. However, such a line is not in general tangentially movable along Φ. Therefore, we repeatedly sample the neighborhood of both endpoints and select a line l with the best tangential movability energy F_l, see Fig. 8. This adaptive sampling improves the initial line which sequentially serves as an input of the ruled surface generation.

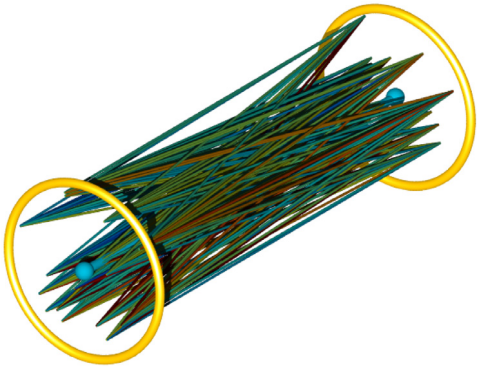

Fig. 8. Local exploration of line movability. A user-defined line l (blue endpoints) is evaluated by the tangential movability F_l. At the normal planes passing through the line endpoints, their alternations are randomly sampled in a certain neighborhood (yellow circles) to form candidate lines (here 15 × 15 lines are shown). These lines are color coded according to F_l and the minimizer is taken as an input in Section 4.3 to generate an initial ruled surface. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

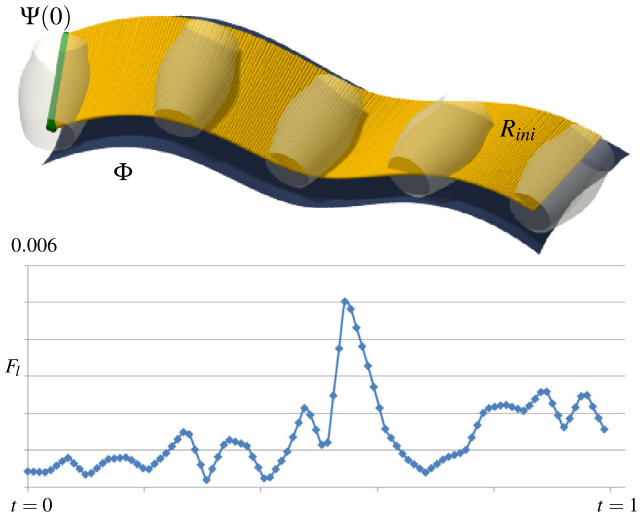

Fig. 9. Generation of initial ruled surfaces. Top: a line with a low tangential movability energy F_l (green) is propagated along Φ in the as-tangential-as-possible manner, i.e., by iteratively computing the best tangential vector field using Eq. (5). The initial ruled surface R_{ini} (yellow) is generated by integrating the best tangential vector fields. The propagation is terminated either when a numerical threshold on F_l is reached or, like in this case, when the surface boundary is encountered. Note that Ψ varies in time; several non-congruent positions are shown. Bottom: the histogram displays the distribution of tangential movability energy F_l of R_{ini}. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

4.3. Initialization from lines with good movability

At this stage of the algorithm, we have a set of locally movable finite lines. Each such line is propagated tangentially along Φ to uniquely define a ruled surface, see Fig. 9. The initial position of the line determines a characteristic and therefore an initial surface of revolution $\Psi(0)$. The line is moved along Φ by iteratively computing the best tangential vector field with respect to Φ using (5), while preserving its length L. The stepsize is set 0.001 for the normalized velocity vector of the line's midpoint. At each time instant, the movability value F_l is computed and the propagation terminates if F_l exceeds a predefined threshold ε, see Fig. 9 bottom.

Such an initialization aims at finding the best tangential motion of a rigid line. Its trajectory, a ruled surface R_{ini}, provides an initial guess for the optimization stage, explained later in Section 4.5. The reason is that by now, the surfaces of revolution associated to l vary over time and the optimization seeks for congruent surfaces of revolution, see Fig. 10.

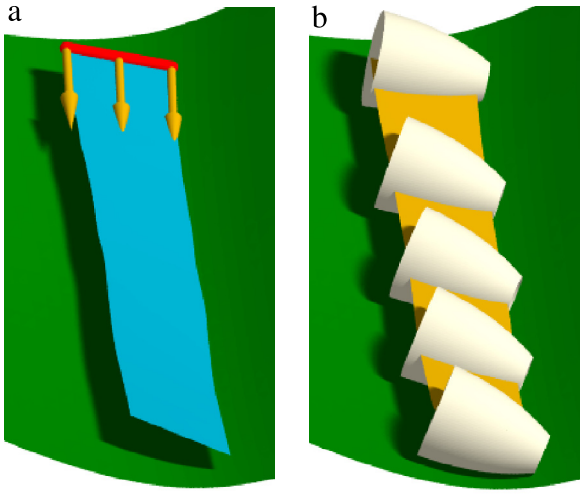

Fig. 10. Line propagation. (a) A line with good movability (red) is moved along the design surface (green) by iteratively computing the best tangential vector field (yellow), see (5). Its rigid body motion generates an initial ruled surface (blue). (b) The blue ruled surface is used as an input for the optimization, see Section 4.5, resulting in the final shape of the machining tool as well as its optimized motion (yellow). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

4.4. Surface-driven initialization

An exploration of the space of lines is time demanding and one usually desires to get a fast feedback whether there exists a part of Φ that can be well approximated by some Ω. Moreover, in many situations of CNC machining, there are preferable paths of the milling tool and, for such a case, a surface-driven instead of line-driven initialization is typically desirable. In our interface, a user can specify several lines which roughly correspond to the preferable positions of the axis of the milling tool. This ordered sequence of lines determines the initial ruled surface that enters the optimization stage, see Section 4.5. In our implementation, the endpoints of the lines serve as control polygons of two boundary cubic B-spline curves. The user specifies then one boundary curve as a trajectory of one end-point of the ruling (finite line), and also prescribes the ruling's length L. With this setting, the initial ruled surface corresponds to a rigid body motion of a rigid finite line.

4.5. Final optimization

During the motion of a rigid rotary cutting tool, each point of the moving cutter axis l must keep its distance to Φ. This implies that velocities of all points on l have to be parallel to the tangent planes at their footpoints, and this has to hold for all time instances. For a general Φ (that is not an exact envelope), each position of l leads to a different milling tool Ψ. We now present an optimization algorithm which simultaneously improves the tool axis motion, and unifies the milling tool shape.

Let us consider the initial ruled surface

$$R(t, s) = (1 - s)\mathbf{a}(t) + s\,\mathbf{b}(t), \quad [t, s] \in [0, 1] \times [0, 1]. \tag{7}$$

Here, we represent the two boundary curves $\mathbf{a}(t)$ and $\mathbf{b}(t)$ as cubic B-spline curves. Further, we equidistantly sample $s \in [0, 1]$ and thus obtain trajectories $R(t, s_j), j = 1, \ldots, n$, of n uniformly sampled points on l (see Fig. 11). We denote these trajectories by $C_j(t)$ and the vector of unknown distances by $\mathbf{d} := (d_1, \ldots, d_n)$.

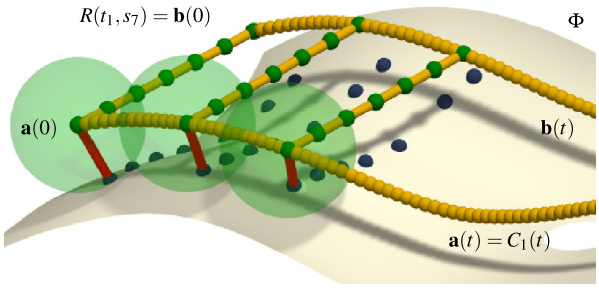

Fig. 11. Optimization settings. The initial ruled surface $R(t, s)$, see (7), is uniformly discretized along the rulings (s-direction) with $n = 7$ samples (green dots). Their footpoints on the reference surface Φ are shown in blue. For a fixed s-value (j = const), a distance d_j (d_1 shown in red) between $R(t_i, s_j)$ and Φ is required to be constant along $C_j(t)$. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Our objective is that each curve $C_j(t)$ lies at a constant distance d_j from Φ. We discretize also the t-space (position of rulings in time) by m samples and thus arrive at the problem to minimize

$$F_{\text{prox}}(\mathbf{a}, \mathbf{b}, \mathbf{d}) = \frac{1}{mn} \sum_{j=1}^{n} \sum_{i=1}^{m} (\text{dist}(R(t_i, s_j), \Phi) - d_j)^2 \to \min \quad (8)$$

subject to the rigidity constraints

$$F_{\text{rigid}}(\mathbf{a}, \mathbf{b}) = \langle \mathbf{a}(t_i) - \mathbf{b}(t_i), \mathbf{a}(t_i) - \mathbf{b}(t_i) \rangle - L^2 = 0, \quad (9)$$

where dist(,) is a point–surface distance function. The free parameters are the control points of the two B-spline curves $\mathbf{a}(t)$ and $\mathbf{b}(t)$, and also the distances $\mathbf{d}$. We further denote $\mathbf{p}_{ij} := R(s_i, t_j)$, $\mathbf{p}_{ij}^{\perp}$ their footpoints on Φ, and $\mathbf{n}_{ij}$ the unit normals at $\mathbf{p}_{ij}^{\perp}$ oriented towards $\mathbf{p}_{ij}$. The proximity objective has two components

$$\begin{aligned} F_{\text{point}}(\mathbf{a}, \mathbf{b}, \mathbf{d}) &= \frac{1}{mn} \sum_{j=1}^{n} \sum_{i=1}^{m} \|\mathbf{p}_{ij} - (\mathbf{p}_{ij}^{\perp} + d_j \mathbf{n}_{ij})\|^2, \\ F_{\text{plane}}(\mathbf{a}, \mathbf{b}, \mathbf{d}) &= \frac{1}{mn} \sum_{j=1}^{n} \sum_{i=1}^{m} (\langle \mathbf{p}_{ij} - \mathbf{p}_{ij}^{\perp}, \mathbf{n}_{ij} \rangle - d_j)^2, \end{aligned} \quad (10)$$

which correspond to a point–point and point–plane distance constraints, respectively.

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

$$\begin{aligned} F_{\text{fair}}(\mathbf{a}, \mathbf{b}) = &\frac{1}{m} \sum_{i=2}^{m-1} (\mathbf{a}(t_{i-1}) - 2\mathbf{a}(t_i) + \mathbf{a}(t_{i+1}))^2 \\ &+ \frac{1}{m} \sum_{i=2}^{m-1} (\mathbf{b}(t_{i-1}) - 2\mathbf{b}(t_i) + \mathbf{b}(t_{i+1}))^2. \end{aligned} \quad (11)$$

The final objective function is

$$\begin{aligned} F_{\text{motion}}(\mathbf{a}, \mathbf{b}, \mathbf{d}) = &\mu_1 F_{\text{plane}}(\mathbf{a}, \mathbf{b}, \mathbf{d}) + \mu_2 F_{\text{fair}}(\mathbf{a}, \mathbf{b}) \\ &+ \mu_3 F_{\text{point}}(\mathbf{a}, \mathbf{b}, \mathbf{d}) + \mu_4 F_{\text{rigid}}(\mathbf{a}, \mathbf{b}), \end{aligned} \quad (12)$$

adding the rulings rigidity (9) as a soft-constraint. We emphasize that distances d_j are updated in every iteration and therefore are also considered as unknowns. We set the optimization such that all three types of unknowns $\mathbf{a}$ and $\mathbf{b}$ (ruled surface) and $\mathbf{d}$ (surface of revolution) are optimized simultaneously. One could eventually split the optimization loop into two separate steps, and optimize interchangeably $(\mathbf{a}, \mathbf{b})$ and $\mathbf{d}$. This would result in only quadratic objective function (12), however, alternating optimizations often suffer with slow convergence, and therefore we optimize all unknowns simultaneously. We did not experiment with various optimization techniques; the optimization problem is solved using the Gauss–Newton method in all examples shown in the paper.

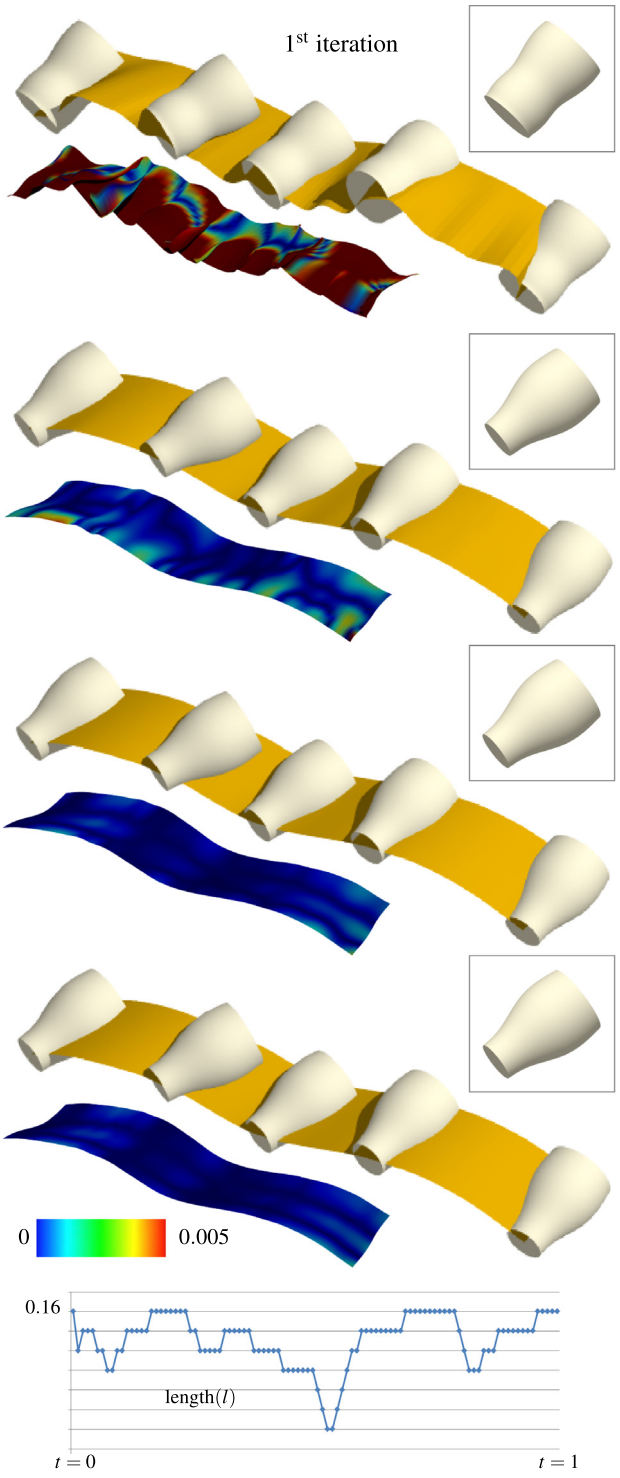

Fig. 12. Optimization. Top: four iterations of the optimization tested on an exact envelope surface are shown. In every iteration, the optimized milling tool (framed) and its motion (yellow) are displayed. The actual envelopes are color-coded by the absolute value of the one-sided distance to the designed surface which shows a fast convergence to the exact solution. Bottom: a histogram of the length of the axis l during one round of optimization is shown. The distance between two neighboring horizontal lines corresponds to $\varepsilon = 10^{-5}$. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

If not stated differently in Section 5, we use the default values $\mu_1 = 1$, $\mu_2 = \mu_4 = 0.1$, and $\mu_3 = 0.001$. Four iterations of the optimization cycle are shown in Fig. 12.

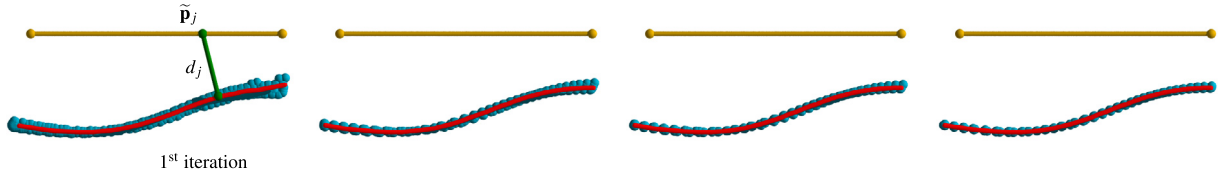

Fig. 13. Half-meridian update. From the optimized distances d_js, an updated half-meridian is computed (red). The axis of revolution is shown in yellow. The blue points correspond to the footpoints computed in the optimization stage, cf. Figs. 7 and 11, and mapped to a common plane. Four iterations of the half-meridian update are shown. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Meridian update. The distances d_j change in every iteration of the optimization loop. Their initial values are obtained from the initial smooth meridian which is the least squares fit to all the footpoints, mapped to a common plane, see Fig. 13 left. We use a cubic spline curve with ten control points, uniformly sample the axis by 30 points, and obtain the initial d_j's by computing their footpoints on the meridian.

As the ruled surface R is changed, the surface of revolution Ψ is also updated by recomputing its meridian. In our discrete setting, the optimization returns the actual values of d_js and therefore the meridian is also treated in a discrete fashion as a polyline by computing a planar envelope of a set of circles centered along l having d_js as radii. Fig. 13 shows four iterations of the half-meridian update.

4.6. Additional constraints

One can incorporate additional constraints on both the shape of the milling tool and its motion. We considered two categories of the milling tools: a general one where the half-meridian is a general spline curve (no constraint) and a conical one where the half-meridian is constrained to a straight line. This linear constraint is applied when the half-meridian is updated, see Section 4.5, by setting the polynomial degree to one and requiring only two control points. An example with conical cutters is shown in Fig. 14. If needed, one could consider, e.g., hyperbolic cutters by requiring a proper rational quadratic half-meridian, or to set additional constraints on the meridian curve like preventing high curvature or concavity.

In our approach, the smoothness of the motion is controlled by the fairness term (11) applied on the trajectories of the endpoints of the ruling. If the objective is to perform a high-speed milling as in [16], (11) could be eventually modified to comfort the proper velocity constraints. Since our primary objective was the approximation quality in terms of the minimum one-sided distance between Φ and Ω, we did not experiment with high-speed performance.

5. Results and discussion

In this section, we present several examples of envelope detection. An example with an exact envelope as the input surface is shown in Fig. 15 top. The color-coding reflects the one-sided absolute error ε between the machined (Ω) and the designed (Φ) surfaces, i.e.,

$$\varepsilon = \min_{i,j} |\text{dist}(R(t_i, s_j), \Phi) - d_j|, \tag{13}$$

considered over a discrete set of samples of the ruled surface R, see also Fig. 11 as a reference for this sampling. If not stated differently, the design surface Φ is normalized such that the diagonal of its bounding box is one. Additionally in Fig. 15, an exact envelope was deformed by locally destroying the exact envelope property. The results show that our algorithm still detects a dominant sub-patch of the exact envelope, but optimizes the surface of revolution and its trajectory accordingly.

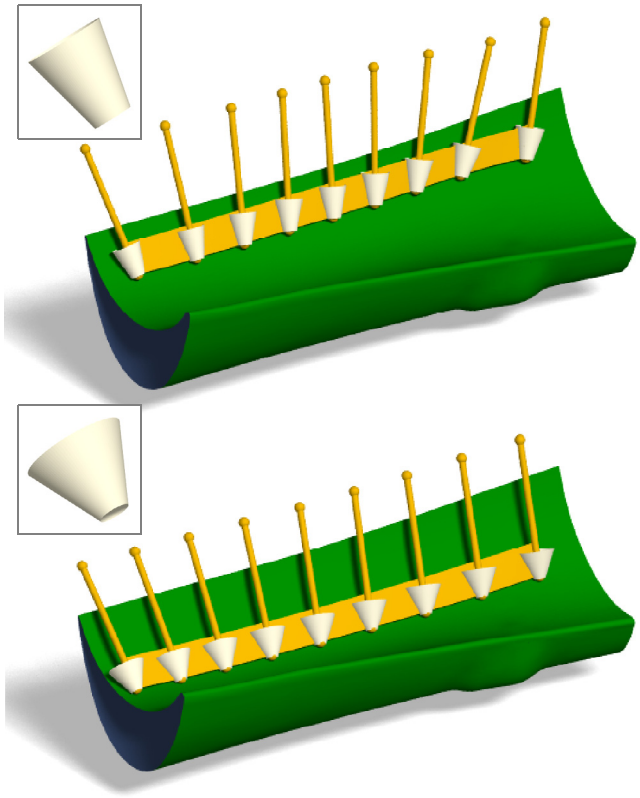

Fig. 14. Conical cutters. Unlike the milling tools of general shapes shown in Fig. 20, one may prefer simpler ones with linear meridians, i.e., conical milling tools. Two optimal milling tools with their penetration-free trajectories are shown.

The results of another test conducted on the exact envelope are shown in Fig. 16 where a random noise was applied on the exact envelope. The optimized solution differs marginally ($\varepsilon < 0.1\%$) from the one without noise, cf. Fig. 15 top, which shows a very good stability of our algorithm.

Fig. 17 shows another reconstruction from distorted data where an exact, yet incomplete, envelope Φ is given as the input. Our method recovers the exact solution within a very fine threshold on the distance error, see Eq. (13); $\varepsilon < 0.1\%$ of the bounding box of Φ.

An example testing our algorithm on industrial data is shown in Fig. 18. The depicted geometry is a reference surface that is used for benchmarking toolpath generation and material removal simulation algorithms in industrial settings. A single sweep approximation is compared with an approach using several patches. A surface-driven initialization was used and the direction of the rulings was set roughly parallel to the long edge of the design surface. A linear constraint on the meridian of the milling tool was applied, see Section 4.6, resulting in optimized milling tool of a semi-cylindrical shape.

Another example with industrial data is shown in Fig. 19. The initialization was surface-driven and the shape of the tool was

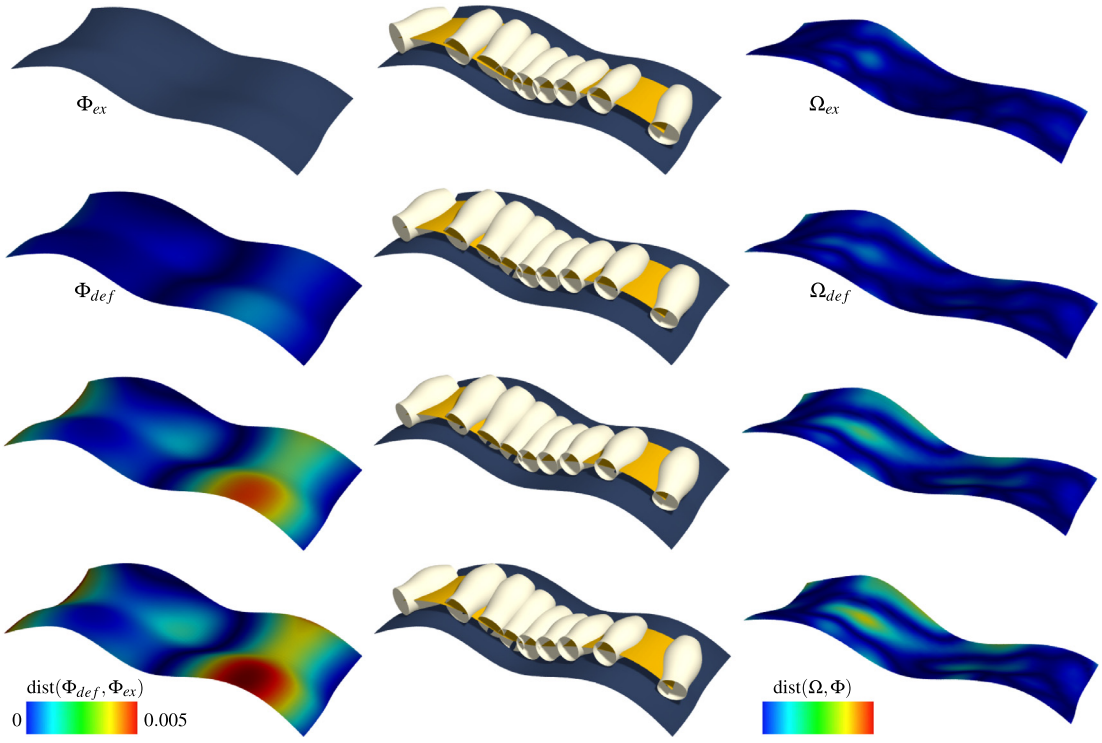

Fig. 15. Stability. Left: an exact envelope Φ_{ex} (top-left) was continuously deformed, destroying the exact envelope property. The color coding shows the error between the exact envelope, with normalized diameter of its bounding box, and the deformed surfaces. Middle: the solutions obtained by our algorithm for the exact (top) and the distorted surfaces. A few positions of the milling tools together with their axes trajectories (yellow) are shown. Right: the resulting (machined) envelopes Ω. These patches are sub-patches of those shown in the left column, and are the envelopes of the cutters' motion shown in the middle column. The color-coding reflects the absolute one-sided error, see (13), to the corresponding designed surfaces Φ. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

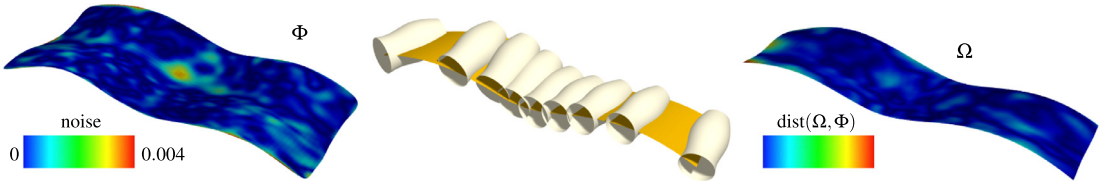

Fig. 16. Noisy data. Left: A random noise was applied on the vertices of the mesh (an exact envelope) shown Fig. 15. The magnitude of the noise is relative to the normalized bounding box of Φ. The solution obtained by our algorithm (middle) and the machined envelope color-coded by the absolute one-sided error to the designed surface are shown.

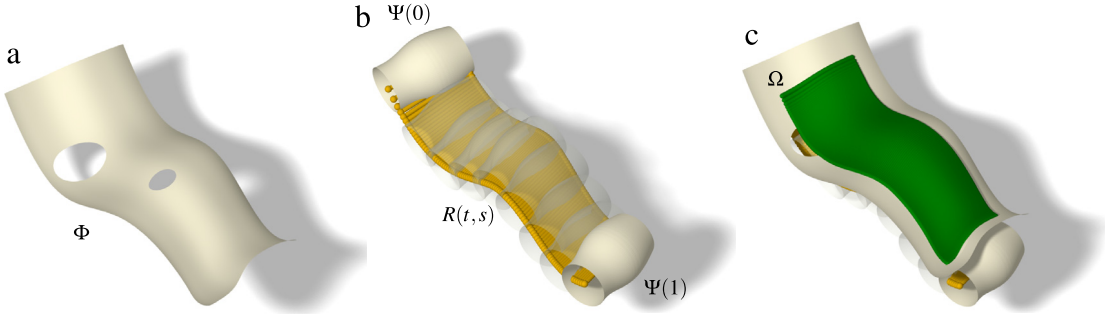

Fig. 17. Detection of envelopes from incomplete data. (a) The input surface Φ is an exact envelope, corrupted by two trimmed holes. (b) The optimized ruled surface (yellow) and the surface of revolution, Ψ, are shown. (c) The recovered exact solution Ω (green) is shown; it approximates the original surface Φ within the one-sided distance error $\varepsilon < 0.1\%$ of the bounding box of Φ. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

constrained to be linear. Two envelopes of the optimal conical cutters that approximate the input geometry within a very fine threshold $\varepsilon = 0.00045$ are shown.

Another example with linear constraints is shown in Fig. 14. In this experiment, a surface-driven initialization was used, now requiring the axis direction to be roughly perpendicular to the dominant edge of the reference surface. The optimal conical cutters and their motions derived by our algorithm are displayed.

A comparison of our two initialization strategies is shown in Fig. 21: the line-driven (Section 4.3) and the surface-driven (Section 4.4) initialization of the motion of the milling axis. An example of the approximation of a free-form blade using a conical

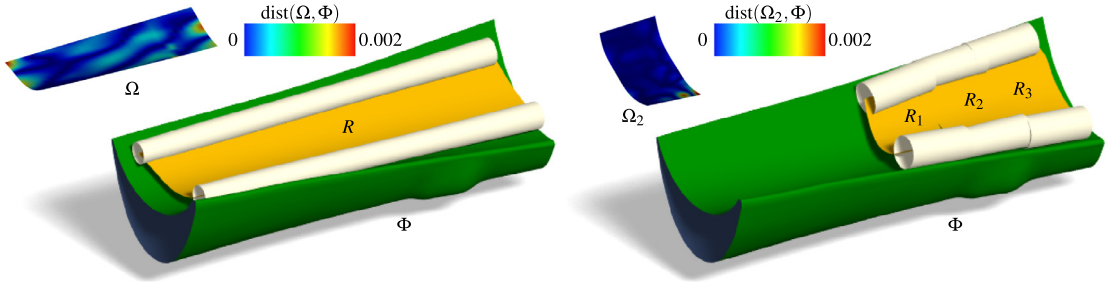

Fig. 18. Number of patches vs. accuracy. Left: the designed surface Φ is approximated by a single patch, compared to the case when three smaller initial patches were assigned (right). The optimized ruled surfaces R are shown in yellow. The meridians of all the milling tools were constrained to be linear, resulting in semi-cylindrical shapes (conical, with the slope close to zero). The color coding visualizes the one-sided error between Φ and Ω, see (13), and shows the trade off between the errors vs. the number of patches. The approximation error is worse for the large patch (left) when compared to the smaller patches, obtained by three mutually different cutters. The smaller patch with the worst error is the middle patch Ω_2.

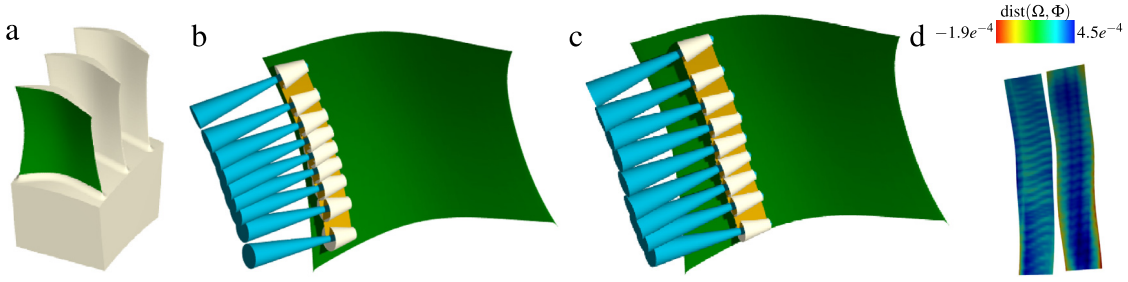

Fig. 19. Industrial data. (a) A "blisk" model is used for benchmarking toolpath generation and material removal simulation algorithms in industrial settings. The machined patch under consideration is shown in green. (b, c) Two penetration-free solutions using different conical cutters are shown. The device to carry the milling tool is visualized as a blue cone. (d) The machined strips, color-coded by the signed distance between the machined envelope Ω and the design surface Φ, are shown. The negative values correspond to overcutting (red), while the maximum error $\varepsilon = 4.5e^{-4}$ is attained in subregions where undercutting occurs (blue). The values are relative to the normalized bounding box of the design surface Φ (green). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

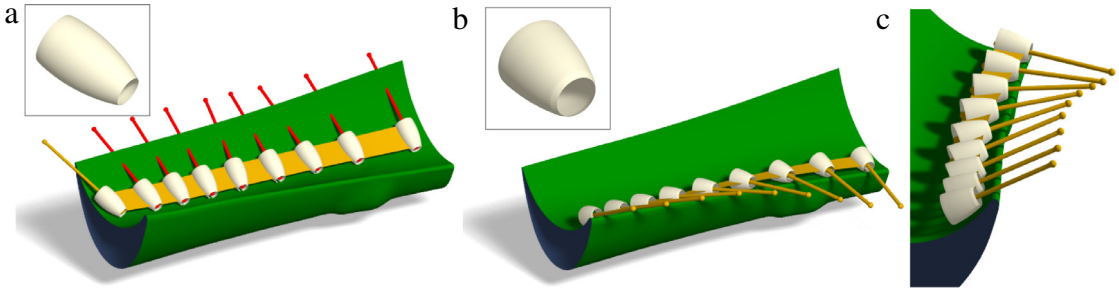

Fig. 20. Collision detection. (a) An optimal tool shape (framed top) is derived together with its motion (yellow ruled surface). Our results are tested a-posteriori if the designed motion is feasible in terms of accessibility. The extended half-axis (yellow) is required to be penetration-free with Φ. The patch showed in (a) failed the collision detection test (red), while in (b, c) the derived motion is feasible. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

envelope is shown in Fig. 22. The initialization was surface-driven; the result satisfies a very fine tolerance ($\varepsilon < 0.03\%$).

Implementation. The current implementation supports meshes, however, it can be easily adapted to smooth input. The reasons for choosing discretized input were mainly computational issues. For example, the closest point computation that is used very frequently (see Sections 4.1 and 4.5) is for a mesh (provided by a volume hierarchy in a preprocessing stage) significantly less expensive than in the smooth setting. We summarize all the statistics of the examples shown in the paper in Table 1. Timings refer to a desktop PC with 4G RAM memory, 4-core CPU, 3.4 GHz.

Collision detection. The machining device that carries the tool must not penetrate the designed surface. Therefore the corresponding half-axis and its sufficient offset (typically a cylinder or cone) must be penetration-free with Φ. Since our objective was to approximate Φ with one, or several, large patches of the size ideally equal the designed surface, we did not consider the collision detection in our optimization framework. We conduct collision detection as a post-process, by testing penetration of the half-axes, provided by some thickness, with Φ. Fig. 20 shows two results of our algorithm where one solution passed this test whilst the other failed.

6. Conclusion and future work

We have investigated a problem of approximating free-form surfaces by a set of envelopes of surfaces of revolution. The proposed algorithm is intended for the finishing stage of CNC flank machining where large scallop-free parts of the machined surface can be generated by a single sweep of the milling tool.

The tangential movability analysis of a 3D line with respect to a free-form surface has been conducted to detect candidate lines with good movability from which initial ruled surfaces arise. Alternatively, we support also a surface-driven initialization to prescribe directly a preferable motion of the machining tool.

Table 1
A summary of our results. The first columns show the number of mesh vertices used to represent the reference surfaces and the preprocessing time needed to build the hierarchy structure (KD-tree). In the "Initialization" columns, we show the type of initialization; whether it was line-driven (L-D) or surface-driven (S-D). The error ε^{ini} refers to the one sided distance between the initial envelope and the reference surface, see (13). In the "Optimization", we show the number of points **p** sampled on the ruled surfaces in the s and t-parameter directions, respectively, and the errors after optimization. Finally, the timings of both main stages of our algorithm are displayed.

Reference shape		Prep. [s]	Initialization		Optimization		Time [s]	
Fig. No.	#vertices		type	ε^{ini}	#**p**	ε^{opt}	T^{ini}	T^{opt}
15 top	13,234	0.8	L-D	$1.1e^{-2}$	30×100	$1.7e^{-3}$	25	21
15 bottom	13,234	0.8	S-D	$3.5e^{-2}$	30×100	$3.3e^{-3}$	50	24
17	12,921	0.7	L-D	$8.1e^{-3}$	30×100	$2.1e^{-3}$	25	25
19(b)	3,132	0.2	S-D	$7.0e^{-3}$	30×100	$3.5e^{-4}$	61	4
19(c)	3,132	0.2	S-D	$8.1e^{-3}$	30×100	$4.5e^{-4}$	70	4
21(a)	3,592	0.2	L-D	$1.3e^{-3}$	30×100	$1.6e^{-4}$	30	3
21(b)	3,592	0.2	S-D	$1.1e^{-3}$	30×100	$6.5e^{-4}$	65	13
22	228	0.1	S-D	$8.0e^{-3}$	30×100	$2.7e^{-4}$	90	25

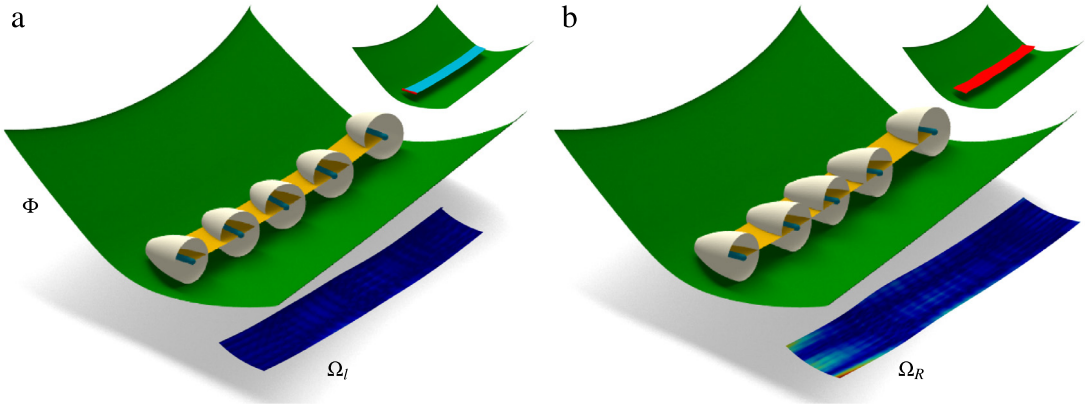

Fig. 21. Different initialization strategies. The optimal shape of the milling tool (white) and its optimal tangential motion (yellow) along Φ are shown. (a) The initial ruled surface was generated by the line-driven strategy, introduced in Section 4.2, while in (b) the initialization is surface-driven, see Section 4.4. The resulting envelopes Ω_l and Ω_R are color-coded by the one-sided distance error, see (13), from Φ. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

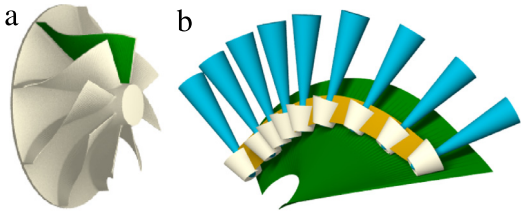

Fig. 22. Impeller. (a) The CAD model of an impeller and its free-form blade (green) to be milled are shown. (b) The optimized motion of the milling tool along the reference surface is displayed. The machining device that carries the tool is visualized as a blue cone. The maximum error of the machined strip is $\varepsilon = 0.00027$ for the reference surface (green) with normalized bounding box. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

We have introduced an optimization framework which improves both the toolpaths and the shape of the milling tool itself. The initial motion of the tool axis and the meridian curve of the surface of revolution are iteratively updated to minimize the one-sided distance between the designed and machined surfaces.

We have validated our algorithm on exact envelopes as well as on real industrial data and have shown that the proposed algorithm detects large parts of the input surfaces by a single motion of the milling tool, while satisfying fine distance thresholds. We believe the presented paper shows the possibilities towards "custom-tools" computer-aided modeling, where an analysis of the design surface reveals a suitable set of machining tools.

As a future research, we point out that currently the initialization is in both cases user-driven and it is still very challenging to set up the initialization fully automatically.

Acknowledgments

This research has been supported by the European Community's 7th Framework Programme under grant agreement 286426 (*GEMS*). The first author was partially supported by The National Natural Science Foundation of China (Grant No. 61202275) and the Fundamental Research Funds for the Central Universities (HIT.NSRIF.201711). The second author has been partially supported by Bizkaia Talent under Grant AYD-000-270 and by the Basque Government through the BERC 2014–2017 program.

References

[1] Warkentin A, Ismail F, Bedi S. Comparison between multi-point and other 5-axis tool positioning strategies. J Mach Tools Manuf 2000;40:185–208.
[2] Lasemi A, Xue D, Gu Peihua. Recent development in CNC machining of freeform surfaces: A state-of-the-art review. Comput Aided Des 2010;42(7):641–57.
[3] Wu CY. Arbitrary surface flank milling of fan, compressor, and impeller blades. Trans ASME, J Eng Gas Turbines Power 1995;117:534–9.
[4] Menzel Cornelia, Bedi Sanjeev, Mann Stephen. Triple tangent flank milling of ruled surfaces. Comput Aided Des 2004;36(3):289–96.
[5] Harik RF, Gong H, Bernard A. 5-axis flank milling: A state-of-the-art review. Comput Aided Des 2013;45(3):796–808.
[6] Senatore J, Landon Y, Rubio W. Analytical estimation of error in flank milling of ruled surfaces. Comput Aided Des 2008;40:595–603.
[7] Elber G, Fish R. 5-axis freeform surface milling using piecewise rule surface approximation. ASME J Manuf Sci Eng 1997;119(3):383–7.
[8] Redonnet J, Rubio W, Dessein G. Side milling of ruled surfaces; optimum positioning of the milling cutter and calculation of interference. Int J Adv Manuf Technol 1998;14(7):459–65.
[9] Peternell M, Pottmann H, Ravani B. On the computational geometry of ruled surfaces. Comput Aided Des 1999;31:17–32.
[10] Pottmann H, Wallner J. Approximation algorithms for developable surfaces. Comput Aided Geom Design 1999;16(6):539–56.

[11] Gong H, Li-Xin C, Jian L. Improved positioning of cylindrical cutter for flank milling ruled surfaces. Comput Aided Des 2005;37:1205–13.
[12] Gong H, Wang N. Optimize tool paths of flank milling with generic cutters based on approximation using the tool envelope surface. Comput Aided Des 2009;41(12):981–9.
[13] Wang CCL, Elber G. Multi-dimensional dynamic programming in ruled surface fitting. Comput Aided Des 2014;51:39–49.
[14] Wang CCL, Tang K. Optimal boundary triangulations of an interpolating ruled surface. J. Comput. Inf. Sci. Eng. 2005;5(4):291–301.
[15] Tang K, Wang CCL. Modeling developable folds on a strip. J. Comput. Inf. Sci. Eng. 2005;5(1):35–47.
[16] Lavernhe S, Tournier C, Lartigue C. Optimization of 5-axis high-speed machining using a surface based approach. Comput Aided Des 2008;40:1015–23.
[17] Pechard PY, Tournier C, Lartigue C, Lugarini JP. Geometrical deviations versus smoothness in 5-axis high-speed flank milling. Int J Mach Tools Manuf 2009;49:453–61.
[18] Zheng G, Bi Q-Z, Zhu L-M. Smooth tool path generation for five-axis flank milling using multi-objective programming. Proc Inst Mech Eng B 2012;226:247–54.
[19] Zhu L, Ding H, Xiong Y. Simultaneous optimization of tool path and shape for five-axis flank milling. Comput. Aided Des. 2012;44:1229–34.
[20] Lu YA, Bi QZ, Zhu LM. Five-axis flank milling of impellers: Optimal geometry of a conical tool considering stiffness and geometric constraints. Proc Inst Mech Eng B 2014;228:1226–36.
[21] Liu X, Li Y, Ma S, Lee C. A tool path generation method for freeform surface machining by introducing the tensor property of machining strip width. Comput Aided Des 2015;66:1–13.
[22] Bartoň M, Flöry S, Pottmann H. Surface approximation with circular sweeps for applications in CNC machining and freeform architecture. Technical report. KAUST; 2013.
[23] Wang XC, Ghosh SK, Li YB, Wu XT. Curvature catering — a new approach in manufacture of sculptured surfaces (part 1. theorem). J Mater Process Technol 1993;38(1–2):159–75.
[24] Wang XC, Ghosh SK, Li YB, Wu XT. Curvature catering — a new approach in manufacture of sculptured surfaces (part 2. methodology). J Mater Process Technol 1993;38(1–2):177–93.
[25] Jensen CG, Red WE, Ernst C. Machining free-form surface cavities using a combination of traditional and non-traditional multi-axis machining methods. Comput Aided Des Appl 2008;5:241–53.
[26] Kim YJ, Bartoň M, Elber G, Pottmann HPrecise. gouging-free tool orientations for 5-axis CNC machining. Comput Aided Des 2015;58:220–9.
[27] Fan J, Ball A. Flat-end cutter orientation on a quadric in five-axis machining. Comput Aided Des 2014;53:126–38.
[28] Bartoň M, Shi L, Pottmann H, Kilian M, Wallner J. Circular arc snakes and kinematic surface generation. Comput. Graph. Forum 2013;32(1):1–10.
[29] Bartoň M, Pottmann H, Wallner J. Detection and reconstruction of freeform sweeps. Comput. Graph. Forum 2014;33(2):23–32.
[30] Pottmann H, Wallner J. Computational line geometry. Springer; 2001.