Journal of Computational Design and Engineering 5 (2018) 348–357

Contents lists available at ScienceDirect

Journal of Computational Design and Engineering

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

On the generation of spiral-like paths within planar shapes

Martin Held, Stefan de Lorenzo *

Universität Salzburg, FB Computerwissenschaften, 5020 Salzburg, Austria

ARTICLE INFO

Article history:
Received 17 August 2017
Received in revised form 15 November 2017
Accepted 23 November 2017
Available online 24 November 2017

Keywords:
Spiral-like path
Double spiral
Composite spiral
Medial axis
Smoothing
High-speed machining

ABSTRACT

We simplify and extend prior work by Held and Spielberger [CAD 2009, CAD&A 2014] to obtain spiral-like paths inside of planar shapes bounded by straight-line segments and circular arcs: We use a linearization to derive a simple algorithm that computes a continuous spiral-like path which (1) consists of straight-line segments, (2) has no self-intersections, (3) respects a user-specified maximum step-over distance, and (4) starts in the interior and ends at the boundary of the shape. Then we extend this basic algorithm to double-spiral paths that start and end at the boundary, and show how these double spirals can be used to cover complicated planar shapes by composite spiral paths. We also discuss how to improve the smoothness and reduce the curvature variation of our paths, and how to boost them to higher levels of continuity.

1. Introduction

1.1. Motivation

Several applications require to cover a planar shape by moving a circular disk along a path. E.g., in machining applications the disk models the cross-section of a tool and the area models a so-called pocket. Similarly, the disk may represent the area covered by a spray nozzle or the area of visibility of a camera device used for aerial surveillance. In our study the planar shape may be bounded by one outer contour and possibly a number of island contours (contained within the outer contour), where each contour is formed by straight-line segments and circular arcs.

Traditional strategies for path generation include zigzag patterns and the use of offset curves to form contour-parallel patterns. See, e.g., Held (1991) for a detailed discussion of both strategies in the context of pocket machining.

Common to these traditional strategies is the fact that the resulting paths contain lots of sharp corners, i.e., abrupt changes of the direction. The higher the speed or the moment of inertia of the moving object represented by the disk, the more these directional discontinuities cause problems. E.g., for a high speed machining (HSM) application, an abrupt change of direction requires the tool to slow down to near-zero speed, change its direction and then accelerate until the desired maximum speed is reached again. In a machining application, sharp corners also lead to a high variation of the tool load.

1.2. Prior work

One way of generating a smooth continuous path is to rely on a traditional strategy and to reduce sharp directional discontinuities in a post-processing step: Pateloup, Duc, and Ray (2004) and Zhao, Wang, Zhou, and Qin (2007), Zhao, Liu, Zhang, Zhou, and Yu (2009) take a conventional tool path and smooth it by inserting circular fillet arcs.

Spiral-like paths are widely regarded as a suitable means for avoiding sharp directional discontinuities. Bieterman and Sandstrom (2002) present an approach based on partial differential equations (PDEs) to compute a spiral-like path inside a star-shaped pocket. Its border contour is successively offset inwards by evaluating the PDE at different points in time. Then these solution contours are connected through radial interpolation. Banerjee, Feng, and Bordatchev (2012) use a similar approach and solve the eigenvalue problem for an elliptic PDE. Neighboring contours are connected based on a winding-angle parameterization. In addition, they explain how to deal with one single island near the center of the planar shape.

Zhou, Zhao, Li, and Xia (2016) propose a strategy that produces smooth, double spirals which start as well as end at the boundary of the planar shape. A series of isothermal lines is derived from a parabolic PDE. By interpolating between successive isothermal lines a closed spiral-like path is produced. Embedding a second spiral-like path between adjacent revolutions of the initial one

Peer review under responsibility of Society for Computational Design and Engineering.
* Corresponding author.
E-mail addresses: held@cosy.sbg.ac.at (M. Held), slorenzo@cosy.sbg.ac.at (S. de Lorenzo).

https://doi.org/10.1016/j.jcde.2017.11.011
2288-4300/

yields the final double spiral. Multiply-connected input shapes, i.e., regions which contain islands, are subdivided into (nearly star-shaped) sub-shapes. One connected path is produced by computing a double spiral inside every sub-shape, and linking neighboring ones together at their ends.

Zhao et al. (2016) suggest space-filling curves (''Fermat spirals'') to cover planar shapes. Another strategy which is based on space-filling curves is introduced by Romero-Carrillo, Torres-Jimenez, Dorado, and Díaz-Garrido (2015): An Archimedean spiral is deformed through linear morphing, and embedded into a convex two dimensional region.

Abrahamsen (2015) constructs a polygonal spiral-like path inside a planar shape bounded by straight-line segments. After calculating an enhanced medial axis tree, a sequence of uniformly distributed wavefronts is computed. Each wavefront is given by a sequence of vertices which are situated on the edges of the medial axis. A closed spiral-like path is generated by manipulating the positions of these vertices. The resulting path is then smoothed by inserting circular arcs at sharp corners.

Held and Spielberger (2009, 2014) generate spiral-like paths for general non-convex planar shapes with or without islands. A series of circles are placed on the medial axis whose radii increase as time progresses. Portions of these circles are interpolated and connected by other circular arcs to form a G^1-continuous path.

While our own work was originally motivated by an HSM application at our industrial partner, we note that a need for paths that cover specific areas while avoiding sharp directional discontinuities arises also outside of CAD/CAM: E.g., Chandler, Rasmussen, and Pachter (2010) insert fillet arcs into a polygonal path in order to take care of maneuverability constraints of an unmanned aerial vehicle. We refer to Keller (2017) for a recent detailed discussion of smooth paths for aerial surveillance.

1.3. Our contribution

Since the algorithm by Held and Spielberger (2009, 2014) is difficult to analyze theoretically and even more difficult to implement reliably, in this work we pick up their basic idea and simplify it significantly: A linearization of the medial axis of the input shape allows to come up with an algorithm for a polygonal spiral-like path that is easy to implement. (And, indeed, an implementation of this algorithm is already in commercial use at our industrial partner.) The path is continuous, without self-intersections, and respects a user-specified maximum ''step-over'' distance. This initial path is then smoothed by refining the positions of its vertices, which helps to reduce the curvature variation. It can be boosted to G^1-continuity or C^2-continuity by using an approximation by biarcs or cubic B-splines. (We use the PowerApx package (Heimlich & Held, 2008; Held & Kaaser, 2014).)

As in the work by Held and Spielberger (2009, 2014), our spiral-like paths start in the interior of the pocket and end at its boundary. The simplicity of our approach allows to generalize this scheme and to devise double-spiral paths that start and end at arbitrary points on the boundary. This makes it easier to cover a complex shape by one continuous spiral-like path by (1) decomposing the shape into simpler sub-areas, (2) computing a (double) spiral within every sub-area, and (3) linking the individual paths to form one continuous path. While such a double-spiral path is unsuited for machining, it may find use in other applications, such as layered manufacturing, spray painting, aerial surveillance, or path finding algorithms for search&rescue missions.

Our approach relies heavily on the medial axis of the input: (1) It serves as the key tool for capturing the geometry of the shape and for computing offset-like curves which form the basis for our paths. (2) It allows to determine an upper bound on the ''step-over'' distance between portions of the spiral. Besides its inherent simplicity, a major advantage of our approach is its generality: It can deal with arbitrarily complex planar shapes with and without islands, thereby guaranteeing a maximum ''step-over'' distance.

2. Preliminaries

Consider a planar input shape that is bounded by straight-line segments and circular arcs, and suppose that we want to move a disk of radius ρ inside this shape such that the area swept by the disk equals (most of) the shape. If the disk has to stay within the shape during the entire movement then it is obvious that its center can never get closer to the boundary of the shape than ρ, even if this constraint results in some areas of the shape being uncovered. (E.g., for a polygonal shape this will happen at convex vertices of the shape.) The loci of all permissible positions of the center of the disk can be obtained as the Minkowski difference of the shape and a disk of radius ρ centered at the origin. (The Minkowski difference $A - B$ of two sets A, B of position vectors in the Euclidean plane $\mathbb{R}^2$ is defined as $A - B := \{c \in \mathbb{R}^2 : c + B \subseteq A\}$.).

We call this set of permissible center positions a *pocket*, P. (But, again, our work is not necessarily restricted to traditional pocket machining applications.) It is well-known that (1) the boundary ∂P of P consists of $O(n)$ straight-line segments and circular arcs if the initial shape was bounded by n straight-line segments and circular arcs, and that (2) it can be obtained in $O(n \log n)$ time via Voronoi-based offsetting (Held, 1991). We use the VRONI/ArcVRONI (Held, 2001; Held & Huber, 2009) package to compute Voronoi diagrams, medial axes[1] and offsets.

Of course, in order to cover as much of P as possible, the disk will have to be moved along the boundary ∂P of P once during a finishing pass. In an actual machining application one may want to consider a Minkowski difference of the input shape and a disk of radius $\rho + \varepsilon$, for some $\varepsilon > 0$, thus pushing ∂P further inwards. This will help to avoid that the tool gets very close to the boundary of the input shape while traveling along our spiral-like path and leaves only a thin amount of material along the boundary for the finishing pass.

We assume that P is path-connected and simply-connected. If P were disconnected then we would run our algorithm separately for every connected component of P. If P contains islands—i.e., is multiply-connected—then we follow (Held & Spielberger, 2014) and convert it to a simply-connected area by introducing bridges, see Fig. 1. (Needless to say, this is a rather complicated pocket that is difficult to cover decently by only one spiral-like path.) Every bridge corresponds to two straight-line segments which have opposing orientations and which are added to the boundary of P in an appropriate way such that one single boundary contour is obtained. Human guidance in the selection of ''good'' bridges (relative to the intended application) is possible but, of course, the algorithm explained in Held and Spielberger (2014) can compute all bridges automatically without human interaction.

It is natural to break a spiral that winds around a point r for k times into a sequence of k individual portions, where each portion corresponds to one full turn around r. We call such a portion of a spiral a *lap*. Then the *step-over distance* at point p of lap L_{i+1} is the minimum distance from p to the next inner lap L_i, cf. Fig. 2. It is obvious that, in general, the step-over distance has to be less than the diameter of the disk which is being moved in order to avoid regions of P that are not covered. In practice, considerably smaller step-over distances are used, though. For HSM a good

[1] Since no efficient algorithm to compute the medial axis of a NURBS curve (or other freeform curve) is known, any freefrom input boundary would have to be approximated by straight-line segments and circular arcs prior to the application of our algorithm.

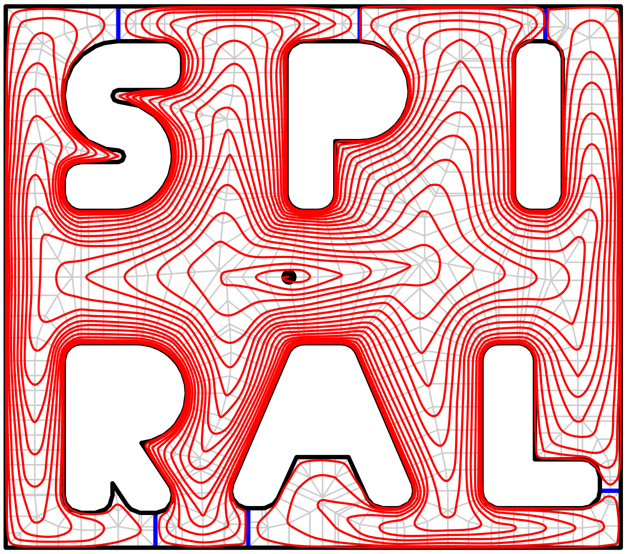


Fig. 1. A cubic B-spline as a spiral-like path inside a multiply-connected planar shape which was converted to a simply-connected shape by means of bridges (shown in blue).

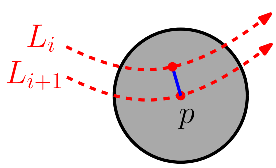


Fig. 2. The local step-over distance at a point p on lap L_{i+1} of a spiral is the minimum distance from p to the next inner lap L_i.

step-over value is a rather small fraction of the diameter that depends on the material of the cutter as well as on the workpiece. (It is largely independent of the geometry of the pocket.) E.g., for aluminum or (non-hardened) steel a typical maximum step-over is given by about 15% of the diameter. In any case, it is important that the user can control the maximum step-over Δ of a spiral path.

We note that in mathematics the term "spiral" has come to mean a curve that emanates from a center point c and winds around c at a monotonically increasing curvature and distance. Hence, every lap of a spiral lies between an inner circle and an outer circle centered at c: Every lap starts at its inner circle and reaches its outer circle after winding around c once. Thereby the fraction $\frac{d_o}{d_i}$ of the distance d_o to the outer circle over the distance d_i to the inner circle decreases monotonically.

In the sequel we investigate "spiral-like" paths that can be seen as a generalization of standard spirals. Our paths also start at a center point, r, and wind around it. And every lap of such a spiral-like path starts at an inner boundary curve and reaches its outer boundary curve after winding around r once, thereby also decreasing the fraction $\frac{d_o}{d_i}$ monotonically. However, for our spiral-like paths we allow general nested Jordan curves[2] in lieu of the concentric circles as boundary curves. Hence, the distances to r and to the inner boundary curve as well as the curvature may also decrease along a lap of such a path. Still, for the sake of terminological simplicity, we prefer to apply the term "spiral" also to our spiral-like paths in the rest of this paper.

[2] A Jordan curve is a closed curve that is simple, i.e., which has no self-intersections.

3. The medial axis tree

According to standard definition the medial axis $\mathcal{MA}(P)$ of the pocket P is the locus of all points inside P which have more than one closest point on the boundary of P, cf. Fig. 3(a). It is known to be a subset of the Voronoi diagram of P, and consist of straight-line segments and portions of conics as edges.

In order to simplify the algorithm by Held and Spielberger (2009) we approximate every edge of the medial axis $\mathcal{MA}(P)$ by a polygonal chain. The vertices of such a polygonal chain are obtained by placing uniformly distributed sample points on the edge such that the maximum length of a segment of the chain is less than a user-supplied or heuristically determined value λ. This process yields the discrete medial axis $\mathcal{MA}'(P)$. We refer to the sample points on $\mathcal{MA}(P)$ and the original nodes of $\mathcal{MA}(P)$ as *nodes* of $\mathcal{MA}'(P)$.

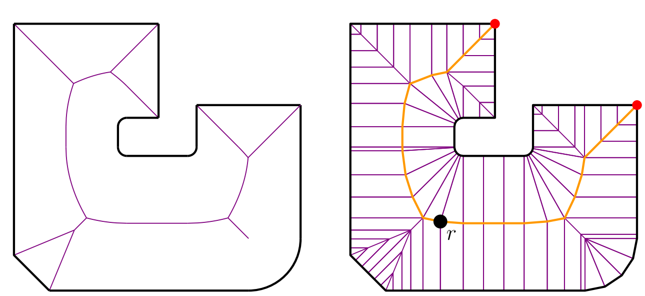
As usual, the *clearance*, $\mathrm{clr}(p)$, of a point p on $\mathcal{MA}'(P)$ is the radius of the largest disk ("clearance disk") centered at p that fits into P. For every node p of $\mathcal{MA}'(P)$ we consider the points $p_1, p_2, \ldots, p_k$ where the clearance disk of p touches the boundary ∂P of P, and construct the clearance line segments $\overline{pp_1}, \overline{pp_2}, \ldots, \overline{pp_k}$. If p happens to be the center of a circular arc a of ∂P then we select finitely many points on a which are uniformly spaced, with a spacing less than λ. Note that some clearance lines might share the same reflex vertex of the boundary ∂P of P as start point.

We add the set of all clearance line segments to $\mathcal{MA}'(P)$ and get the new (planar straight-line graph) $\mathcal{MA}''(P)$. The medial axis $\mathcal{MA}(P)$ is known to form a tree because P does not contain islands. This property carries over to $\mathcal{MA}''(P)$ if we regard the start points of two different clearance lines as different nodes even if they coincide at the same reflex vertex of the boundary of P. Hence, by choosing one (inner) node r of $\mathcal{MA}''(P)$ as root we can turn $\mathcal{MA}''(P)$ into a rooted tree $\mathcal{T}_r$, the *discrete medial axis tree* derived from $\mathcal{MA}''(P)$. (Since we will use this symbol for the discrete medial axis tree of P at various places and also within mathematical equations we keep the notation simple and do not make the dependence of $\mathcal{T}_r$ on P explicit in the notation.) All points that correspond to the leaves of $\mathcal{T}_r$ lie on ∂P. In particular, every start point of a clearance line on ∂P forms a leaf node of $\mathcal{T}_r$.

Since all edges of $\mathcal{T}_r$ are given by line segments, it is easy to compute the (Euclidean) length $d_{\mathcal{T}_r}(p, q)$ of the unique path along $\mathcal{T}_r$ between two nodes p, q of $\mathcal{T}_r$. This allows us to define the *Euclidean height* of a node p of $\mathcal{T}_r$ as

$$h_{\mathcal{T}_r}(p) := \max_q d_{\mathcal{T}_r}(p, q),$$

where the maximum is taken over all nodes q of the sub-tree(s) of $\mathcal{T}_r$ rooted at p.



(a) (b)

Fig. 3. (a) Medial axis of a pocket P; (b) the height-balanced discrete medial axis tree $\mathcal{T}_r$ rooted at r, with the two leaves that define the Euclidean height $h_{\mathcal{T}_r}(r)$ of r shown in red and the corresponding two radial paths shown in orange.

As in Held and Spielberger (2014) we assume that $\mathcal{T}_r$ is *height-balanced*: We assume that $h_{\mathcal{T}_r}(r)$ is defined by at least two different leaves of $\mathcal{T}_r$. That is, we assume that there exist $k \geqslant 2$ distinct leaf nodes $v_1, v_2, \ldots, v_k$ of $\mathcal{T}_r$ such that

$$h_{\mathcal{T}_r}(r) = d_{\mathcal{T}_r}(r, v_1) = d_{\mathcal{T}_r}(r, v_2) = \cdots = d_{\mathcal{T}_r}(r, v_k).$$

Every path from r to such a leaf v_i is called a *radial path* of $\mathcal{T}_r$. See Fig. 3. (For the sake of visual clarity we show this toy example with a very coarse discretization and (in subsequent figures) with an unrealistically large step-over distance.) If no such node r exists in $\mathcal{T}_r$ then we insert a new node within an edge of $\mathcal{T}_r$ in order to achieve such a perfect height balance. The computation of all Euclidean heights of the nodes of $\mathcal{T}_r$ and the height-balancing can be done easily in time linear in the number of edges of $\mathcal{T}_r$ (Held & Spielberger, 2014). In particular, no human interaction is needed for choosing the root r. In the sequel we will use r as the start point for our spiral-like paths.

Of course, the algorithms explained in the rest of our paper remain applicable if a point other than the height-balanced root r is chosen as start point. As a matter of principle, any point p in the interior of the pocket P could be chosen as start point of the spiral-like path and root r of the medial-axis tree. If p does not lie on $\mathcal{MA}''(P)$ then we consider the (closest) projection of p onto the boundary ∂P of P, and add the elongation of this projection between $\mathcal{MA}''(P)$ and ∂P as dummy Voronoi edge (Held & Spielberger, 2009). We note, though, that choosing a start point other than the height-balanced root r will result in (1) an increased length of the final spiral-like path, (2) in an increased number of laps, and (3) in a highly irregular spacing of the laps. See Fig. 4 for sample (polygonal) spirals computed by the algorithm presented in Section 5 for five different start points on $\mathcal{MA}''(P)$. Note that the same maximum step-over distance Δ was used for all five paths. We refer to Held and Spielberger (2014) for a detailed discussion of the impact of a variation of the start point.

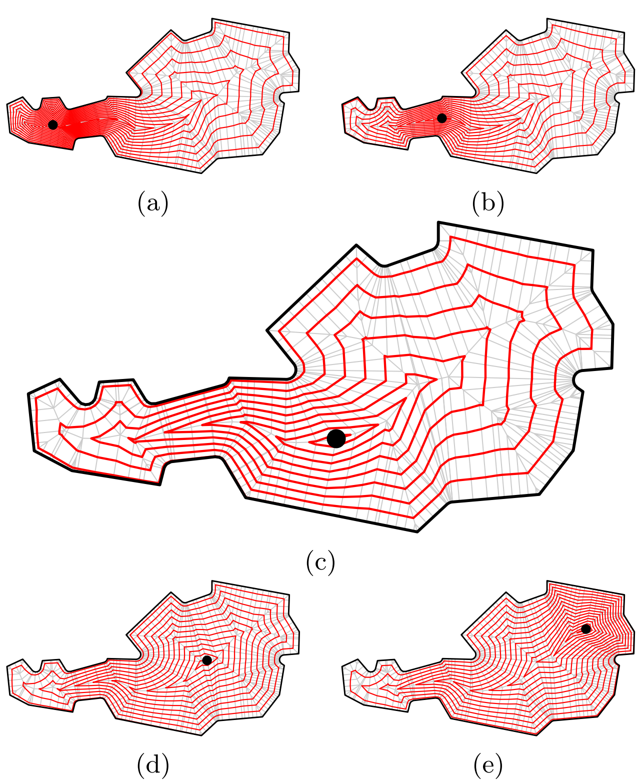


Fig. 4. Moving the start point of a spiral path away from the height-balanced root r may have a significant impact on the length of the path and on the spacing of its laps. The middle (larger) figure shows the path that starts at r.

4. Impulse propagation

Similar to Held and Spielberger (2009) we consider an impulse which is active during the time interval $[0, 1]$, which starts at the root r of the discrete medial axis tree $\mathcal{T}_r$ at time $t := 0$, and discharges concurrently at all leaves of $\mathcal{T}_r$ at time $t := 1$. Suppose that we want the impulse to travel along every radial path of $\mathcal{T}_r$ with constant velocity. Then the impulse has to cover a distance of $h_{\mathcal{T}_r}(r)$ within unit time, which implies that the velocity v of the impulse along every edge of a radial path equals $h_{\mathcal{T}_r}(r)$. Hence, a node p on a radial path of $\mathcal{T}_r$ is reached at the "start time"

$$t = \frac{h_{\mathcal{T}_r}(r) - h_{\mathcal{T}_r}(p)}{h_{\mathcal{T}_r}(r)}.$$

This simple observation can be used in a recursive manner to determine the time when the impulse reaches a specific node (or even any point within an edge of $\mathcal{T}_r$) together with the impulse velocity for all edges of $\mathcal{T}_r$. Initially, the start times for all nodes on the radial paths of $\mathcal{T}_r$ are known. (Recall that the start time t_r of the root r was set as $t_r := 0$.) Now imagine removing all edges of all radial paths from $\mathcal{T}_r$. Peeling off these "longest branches" splits $\mathcal{T}_r$ into a number of rooted sub-trees, where every sub-tree is rooted at a node of a radial path. Let p be the root of the sub-tree T_p, and let us denote its start time by t_p. We choose a leave node p' in T_p such that

$$d_{\mathcal{T}_r}(p, p') = \max_{v \in T_p}\{d_{\mathcal{T}_r}(p, v)\},$$

with ties being broken arbitrarily. That is, the path from p to p' is a longest path in T_p (and also in $\mathcal{T}_r$) from p to a leaf of T_p. Let q be the child of p on this path, and let l_e denote the length of the edge e between p and q. The length $d_{\mathcal{T}_r}(p, p')$ of the entire path from p to p' is denoted by l_b. Since the impulse has to reach p' at time $t := 1$, the (constant) velocity of the impulse along e and all other edges of the path from p to p' is given by

$$v_e = \frac{l_b}{1 - t_p} = \frac{h_{\mathcal{T}_r}(q) + l_e}{1 - t_p}.$$

We conclude that the impulse reaches q at the start time

$$t_q = t_p + \frac{l_e}{v_e}.$$

Similarly, due to the fact that the velocity of the impulse stays constant along the whole edge e, the start time t_s of a point s within (the relative interior of) e is simply given by

$$t_s = t_p + \frac{d_{\mathcal{T}_r}(p, s)}{v_e}.$$

As in the case of the nodes on the radial paths, the start times of all other nodes on the path from p to p' can be computed easily, too. Once all these start times are known we remove from T_p all edges of the path from p to p', thereby splitting T_p into a number of sub-trees. Then we apply this scheme recursively to these newly generated sub-trees.

Note that we have $v_e \leqslant v$, where v is the velocity along a radial path of $\mathcal{T}_r$. Furthermore, the equality $v_e = v$ holds only if the path from r to p' forms a radial path, too.

This recursive scheme allows us to determine all edge velocities and start times in time linear in the number of edges of $\mathcal{T}_r$. It is an easy exercise to prove that this scheme guarantees that the impulse will reach all leaves of $\mathcal{T}_r$ at time $t := 1$. Effectively, this scheme splits $\mathcal{T}_r$ into a number of branches, with constant impulse velocity per branch. See Fig. 5. We denote this set of branches by B.

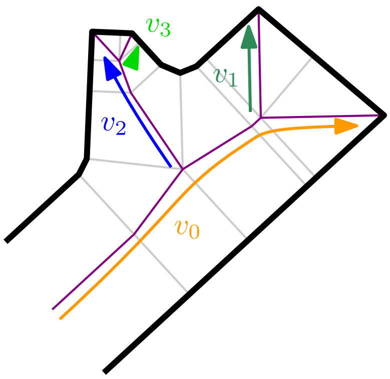


Fig. 5. The velocities on some branches of $\mathcal{T}_r$.

As the impulse flows through $\mathcal{T}_r$, it covers an increasing portion of $\mathcal{T}_r$. The point which the impulse reaches at time t on its way from r to some leaf of $\mathcal{T}_r$ is called a *vertex* at time t. Clearly, for any time $t \in [0,1]$ there exist at most as many vertices as there are leaves in $\mathcal{T}_r$. By computing all vertices at a specific moment in time, and arranging them in the order in which they appear when $\mathcal{T}_r$ is traversed in depth-first manner, it is possible to construct a closed polygonal chain, a so-called *wavefront* $w(t)$ at time t.

The spacing of the wavefronts has to be chosen carefully in order to guarantee that the user-specified maximum step-over Δ is respected. Consider a uniform decomposition of time $(t_0, t_1, \ldots, t_m)$, for some (yet unknown) $m \in \mathbb{N}$, with $0 = t_0 < t_1 < \cdots < t_m = 1$. The vertices of the wavefront $w(t_i)$ are given by the positions of the impulse at time t_i, see Fig. 6.

Let $t^* := t_{i+1} - t_i$ denote the constant time difference between the times of two neighboring wavefronts. Recall that the (symmetric) Hausdorff distance $H(X,Y)$ between two closed and bounded sets $X, Y \subset \mathbb{R}^2$ is defined as

$$H(X,Y) := \max\left\{ \max_{x\in X} \min_{y\in Y} d(x,y),\ \max_{y\in Y} \min_{x\in X} d(x,y) \right\},$$

where $d(x,y)$ denotes the standard Euclidean distance of two points $x, y \in \mathbb{R}^2$. Our goal is to choose t^*, such that

$$H(w(t_i), w(t_{i+1})) \leqslant \Delta \quad \text{for all} \quad i \in \{0, 1, \ldots, m-1\}.$$

We recall that the impulse velocity is bound by $h_{\mathcal{T}_r}(r)$ for every edge of $\mathcal{T}_r$. This implies that the impulse travels a distance of at most $s \cdot h_{\mathcal{T}_r}(r)$ in time s along $\mathcal{T}_r$. Hence, we are able to establish an upper bound on the symmetric Hausdorff distance between $w(s_0)$ and $w(s_0+s)$, with $s_0 \in [0, 1-s]$, as follows:

$$H(w(s_0), w(s_0+s)) \leqslant s \cdot h_{\mathcal{T}_r}(r).$$

This implies that the impulse travels a distance of at most Δ along $\mathcal{T}_r$ during the time s if we set

$$s := \frac{\Delta}{h_{\mathcal{T}_r}(r)}.$$

Summarizing, in order to ensure $H(w(t_i), w(t_{i+1})) \leqslant \Delta$ for all $i \in \{0, 1, \ldots, m-1\}$, it suffices to set m as

$$m := \left\lceil \frac{1}{s} \right\rceil.$$

This gives

$$t^* := \frac{1}{m}$$

as the constant time distance between two impulse times that correspond to neighboring wavefronts. Note that this construction implies that the radial paths are split by the wavefronts into sections with length at most Δ.

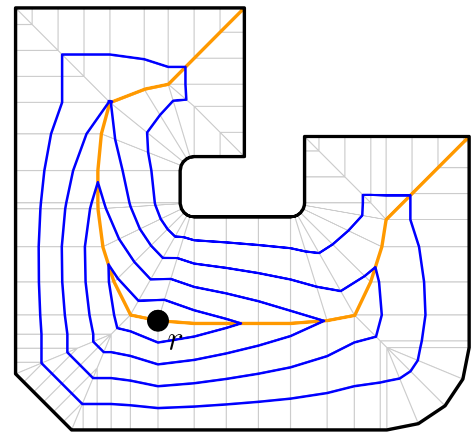


Fig. 6. A series of uniformly spaced wavefronts inside the pocket for $m := 5$. The wavefront $w(t_0)$ equals r, and $w(t_5)$ coincides with the boundary of P; both are not shown. The two radial paths in $\mathcal{T}_r$ between r and leaves of $\mathcal{T}_r$ are shown in orange.

5. Generating one spiral

We now focus on the generation of the actual spiral path, which is fundamentally different to the strategy applied by Held and Spielberger (2009). We explain and depict counter-clockwise (CCW) spiral paths; the modifications needed to obtain clockwise (CW) spirals are trivial. A spiral path $\mathcal{S}(P, \Delta)$ is made up of m laps $L_1, L_2, \ldots, L_m$. In addition, we have $L_0 := \{r\}$ and $L_{m+1} := \partial P$ as two "trivial" laps. Each of the laps is a polygonal chain whose vertices lie on $\mathcal{T}_r$. In a nutshell, we compute the innermost (non-trivial) lap L_1 by interpolating between the wavefronts $w(t_0)$, i.e., the root r of $\mathcal{T}_r$, and $w(t_1)$. Similarly, L_m is formed by interpolation between $w(t_{m-1})$ and $w(t_m)$, i.e., ∂P. See Fig. 7. All other (non-trivial) laps are formed by interpolations between L_1 and L_m. Every lap starts and ends at one specific clearance line incident at r. The important technical issue is to generate these laps in such a way that the step-over distance between neighboring laps does not exceed the user-specified maximum step-over Δ.

We start with explaining how L_1 is generated, see Fig. 8. Recall that $w(t_0)$ degenerates to r. Suppose that q_0 is the vertex of $w(t_1)$ that is intersected by the clearance line $\overline{rv_0}$, on which all laps start and end. Thus, L_1 starts at r and ends at q_0. We number the vertices of $w(t_1)$ in CCW order, starting at q_0. Now consider some vertex of $w(t_1)$, e.g., q_4 in Fig. 8. Let d denote the circumference of $w(t_1)$, let d_4 denote the length of the polygonal chain $q_0 q_1 \ldots q_4$, and let $\delta_4 := d_{\mathcal{T}_r}(r, q_4)$, i.e., the distance from r to q_4 along $\mathcal{T}_r$. Then a candidate corner c of L_1 is placed on the path from q_4 to r at a distance (along $\mathcal{T}_r$) of

$$\left(1 - \frac{d_4}{d}\right) \cdot \delta_4$$

from q_4. We store c at the corresponding edge of $\mathcal{T}_r$. Note that some vertices of $w(t_1)$ might end up storing candidate corners on the same edge or path towards r. These candidate corners are classified as "type-1" candidate corners.

After setting the weight d to the circumference of ∂P and letting the vertices of $w(t_{m-1})$ play the role of r, we obtain type-1 candidate corners for L_m in a similar way by moving from the vertices of $w(t_m)$, i.e., ∂P, towards vertices of $w(t_{m-1})$. If required, we can also let L_m wind around r a bit more than once, and let it end at some point on ∂P other than v_0, by making d larger than the circumference of ∂P.

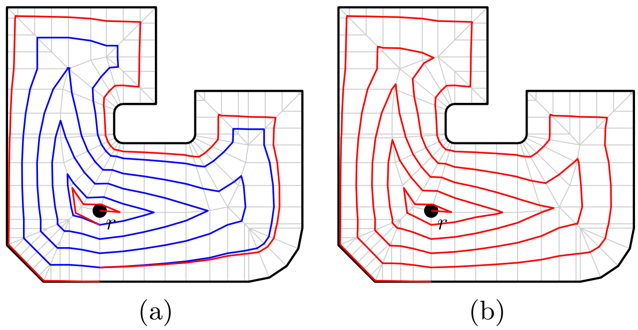


Fig. 7. (a) The first and the last lap are derived by interpolating neighboring wavefronts. (b) The final spiral path that starts at r and ends on ∂P.

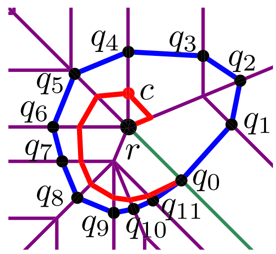


Fig. 8. The first lap starts at the root r of $\mathcal{T}_r$ and ends at a vertex q_0 of $w(t_1)$ on a clearance line (shown in green), on which all laps start and end.

In order to actually generate L_1 we scan $\mathcal{T}_r$ in a depth-first order, starting at r and moving along $\overline{rv_0}$ as first branch of $\mathcal{T}_r$. The recursive scan stops whenever a candidate corner for L_1 is encountered. This depth-first scan establishes all vertices of L_1 in the desired (CCW) order.

Now we start a new depth-first scan towards the leaves of $\mathcal{T}_r$ at every vertex q of L_1. The recursion of the depth-first scan is stopped whenever we get to a distance $(m-1)\cdot\Delta$ from q along $\mathcal{T}_r$ or, trivially, if we reach the boundary ∂P. At every such stopping point of the recursion a new "type-2" candidate corner for L_m is placed. Then another depth-first scan starting at r reveals all vertices of L_m by stopping the recursion whenever a candidate corner for L_m (of either type-1 or type-2) is encountered.

Our construction implies the following two distance properties:

$$H(L_0, L_1) \leqslant \Delta \quad \text{and} \quad H(L_1, L_m) \leqslant (m-1)\cdot\Delta.$$

We now argue that L_m is guaranteed to be contained in the annulus defined by $w(t_{m-1})$ and $w(t_m)$: Every type-1 candidate corner for L_m lies in this annulus since it is generated by an interpolation between $w(t_{m-1})$ and $w(t_m)$. Every type-2 candidate corner which does not lie on ∂P is at a distance of $(m-1)\cdot\Delta$ along $\mathcal{T}_r$ from a vertex of L_1 and, thus, at a distance of at least $(m-1)\cdot\Delta$ from r. However, all vertices of $w(t_{m-1})$ are at a distance of at most $(m-1)\cdot\Delta$ from r. Thus, also every type-2 candidate corner lies within the annulus defined by $w(t_{m-1})$ and $w(t_m)$. As a result, L_m lies also in this annulus. (More precisely, all of L_m lies within the interior of this annulus except for the start point and end point of L_m.) In particular, we get

$$H(L_m, \partial P) = H(L_m, L_{m+1}) \leqslant \Delta$$

as the third distance property.

The remaining laps $L_2, \ldots, L_{m-1}$ can be generated similar to the generation of the initial wavefronts if we take the freedom to regard one lap as a special type of wavefront between L_1 and L_m: Again we let an impulse propagate along $\mathcal{T}_r$. However, this modified impulse propagation starts at time $t := 0$ at the vertices of L_1, and ends at time $t := 1$ at the vertices of L_m. Then, for properly chosen velocities of the impulse on the edges of $\mathcal{T}_r$, the "wavefront" that corresponds to the time $i/m-2$ forms the lap L_{i+1}, for $i \in \{1, 2, \ldots, m-2\}$.

By connecting all non-trivial laps $L_1, L_2, \ldots, L_m$ in the natural way we obtain a polygonal path $\mathcal{S}(P, \Delta)$ inside P. Trivially, $\mathcal{S}(P, \Delta)$ starts at r and ends on ∂P. Furthermore, $\mathcal{S}(P, \Delta)$ is not self-intersecting because we move outwards in a strictly monotonic fashion, starting at r, until we arrive at ∂P. And due to the construction, $\mathcal{S}(P, \Delta)$ respects the maximum step-over Δ: The $m-2$ laps $L_2, \ldots, L_{m-1}$ split a distance (along $\mathcal{T}_r$) of at most $(m-1)\cdot\Delta$ into $m-1$ portions of length at most Δ. Hence, the Hausdorff distance between L_i and L_{i+1} is at most Δ for all $i \in \{1, 2, \ldots, m-1\}$.

We summarize our result as follows:

$$H(L_i, L_{i+1}) \leqslant \Delta \quad \text{for all} \quad i \in \{0, 1, \ldots, m\},$$

which settles the claim that our spiral path $\mathcal{S}(P, \Delta)$ obeys the user-specified maximum step-over Δ. We note that Δ forms an upper bound on the true maximal step-over distance: We do not determine the actual Hausdorff distance but only measure distance along (possibly curved) edges of the medial axis of P. (An algorithm by Alt, Behrends, & Blömer (1995) would allow to compute one Hausdorff distance between polygonal curves with a total of n vertices in $O(n \log n)$ time but there is no obvious way for applying this algorithm to the laps of our spiral path under generation.)

6. Improving and smoothing a spiral

6.1. Impulse modification

Recall that the impulse moves with constant velocity per branch of B, cf. Fig. 5. In particular, it is constant within every edge of $\mathcal{T}_r$. Hence, the velocity of the impulse might change rapidly at some nodes of $\mathcal{T}_r$. This leads to exceedingly sharp corners along the spiral path. We now explain how to remedy this problem by modifying the impulse propagation.

In order to mitigate the effects of rapidly changing velocities whenever a shorter branch starts, we part from the simple scheme of using constant velocities and assign a linear velocity function to every element of B. As in Section 4, the dynamic velocity of r is set to $h_{\mathcal{T}_r}(r)$ and its start time t_r is set to 0. The branches in B are, again, considered in the order in which they appear when $\mathcal{T}_r$ is traversed in depth-first manner. Let b be the branch that is currently inspected, with p as its start node, p' as its end (leaf) node, and l_b as its length. According to Section 4, the constant "average" impulse velocity assigned to all edges of b is given by

$$v_{\text{avg}} = \frac{l_b}{1 - t_p},$$

where t_p denotes the start time at p. Roughly, the new idea is to start with an initial velocity along b that (ideally) is identical to the velocity v_p with which the impulse reached p, and to decrease this velocity linearly as one gets closer to ∂P. Of course, even after this modification the impulse will have to travel a distance of l_b within time $1 - t_p$.

We define the start velocity along b as

$$v_{\text{start}} := \begin{cases} v_p & \text{if } 2v_{\text{avg}} \geqslant v_p, \\ 2v_{\text{avg}} & \text{else.} \end{cases}$$

Furthermore, the end velocity v_{end} along b is defined as

$$v_{\text{end}} := \begin{cases} 2v_{\text{avg}} - v_p & \text{if } 2v_{\text{avg}} \geqslant v_p, \\ 0 & \text{else.} \end{cases}$$

The corresponding linear velocity function ϑ^b for the velocity along b is given by

$$\vartheta^b(s) := v_{\text{start}} - (v_{\text{start}} - v_{\text{end}})s,$$

with $s \in [0, 1]$. Obviously, the velocity along b at a specific time t, with $t_p < t \leqslant 1$, is given by

$$\vartheta^b\left(\frac{t - t_p}{1 - t_p}\right).$$

Finally, at time t the impulse has travelled a distance of

$$\frac{v_{\text{start}} + v_q}{2}(t - t_p)$$

along b. We note that the distance travelled by the impulse equals l_b for $t := 1$, for both cases in the settings of v_{start} and v_{end}.

We can now use this modified linear impulse velocity and apply the schemes discussed in Sections 4 and 5 to compute the wavefronts as well as the spiral path, see Fig. 9. We note that the modified impulse travels with the (standard) constant velocity $v = h_{\mathcal{T}_r}(r)$ along all radial paths of $\mathcal{T}_r$. Along all other branches the velocity varies but never exceeds v. This fact implies that the distance analysis of Section 5 is still applicable and that the maximum step-over Δ is respected everywhere along the final spiral path even for the modified impulse setting.

In order to reduce directional discontinuities even further we keep in mind that a vertex v of lap L_i of the spiral path could be moved along $\mathcal{T}_r$ towards ∂P as long as this movement does not (1) result in a violation of the maximum step-over Δ or (2) cause v to run over L_{i+1}. One could even require that v keeps a certain minimum distance from L_{i+1} in order to avoid that laps get extremely close to each other. In any case, every vertex v has a range of positions which are permissible for an outwards shift of v. (This range can also be empty for some particular vertex.)

Let (v_1, v_2, v_3) be a triple of consecutive vertices of the spiral. We say that the angle at v_2 is convex if v_2 lies to the left of the ray from v_1 to v_3, reflex if it lies to the right of this ray, and tangential otherwise. We compute the deviation of the angle at v_2 from 180°, and insert its absolute value into a priority queue PQ. We also keep a link from v_2 into the position of this value in PQ, and from it back to v_2. This is done for all vertices of the spiral path. The priority queue PQ is organized such that it maintains the maximum angular deviation at its top.

Once PQ has been filled we are ready to shift some vertices. Let v_2 be the vertex that is linked to the angular deviation currently fetched from PQ. If the angle at v_2 is convex then we shift v_2 outwards. If it is reflex then we shift its predecessor v_1 and its successor v_3 outwards. Of course, the shifting of one or two vertices of the triple (v_1, v_2, v_3) shall not result in deviations of the angle(s) from 180° at the unshifted vertices which are greater than the one which we try to reduce at v_2. In theory, the optimum amount(s) for shifting could be determined by solving a (non-linear) optimization problem. We resort to a much simpler approach and sample 10 uniformly distributed positions within the maximum permissible range of new positions. (The sample number 10 turned out to be good enough for our purposes; there is no theoretical justification for it.) If the optimal shift determined this way does indeed reduce the maximum absolute deviation of the angles at v_1, v_2 and v_3 from 180° then we delete the three entries for v_1, v_2, v_3 from PQ and insert the three absolute values of the new deviations from 180° at v_1, v_2 and v_3 into PQ. Otherwise, the entry for v_2 is deleted from PQ but no shift is carried out. See Fig. 10(a) for a result of this shifting strategy applied to the setting of Fig. 9(b). Additional sample paths are shown in Fig. 11; the polygonal path derived from a constant impulse propagation for the sample pocket of Fig. 11(b) is shown in Fig. 4.

A minor technical problem is given by the fact that shifting a vertex towards ∂P might cause it to run over a node of $\mathcal{T}_r$. In such a case we have to split the vertex into several individual copies that move independently towards ∂P.

6.2. Higher-order smoothing

For now we have obtained a spiral path which is described by a polygonal chain. Practical experiments made it apparent quickly

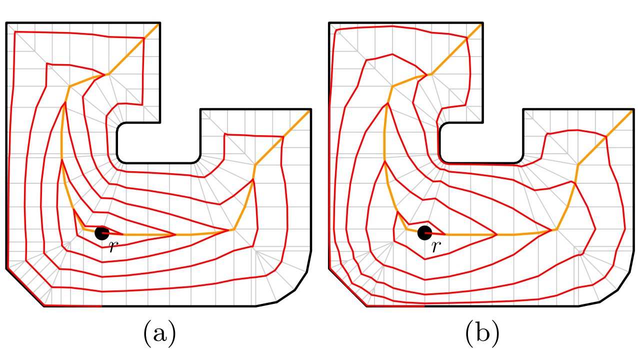


Fig. 9. (a) Spiral path according to piecewise constant velocities of the impulse, cf. Section 5. (b) Spiral path according to the modified linear velocities of the impulse.

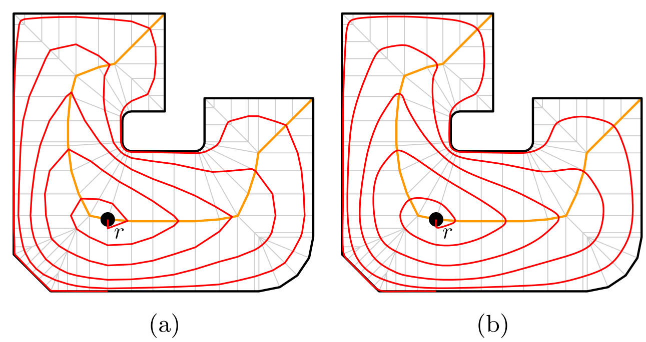


Fig. 10. (a) Spiral path after some vertices were shifted outwards; (b) approximation of this path by a cubic B-spline.

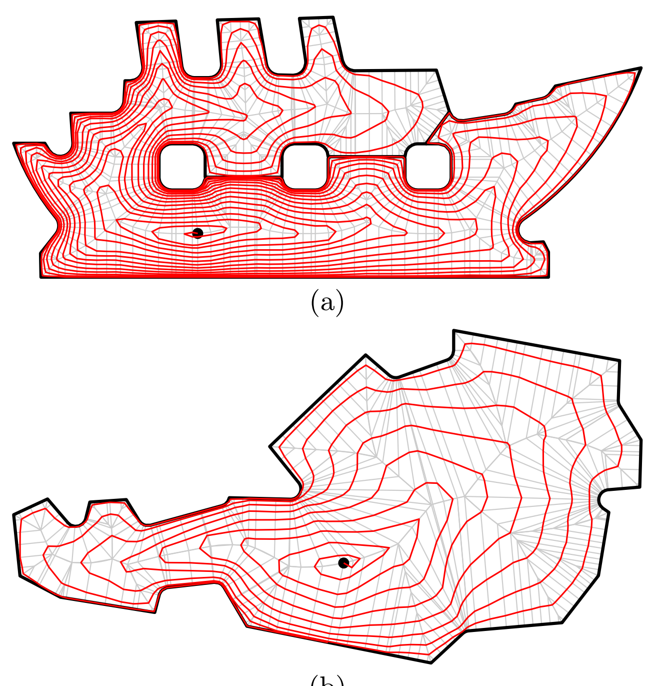


Fig. 11. Sample polygonal spiral paths generated based on the modified impulse propagation.

that there is nothing to gain by employing non-linear functions for the impulse velocity: The higher the algebraic degree of the velocity function, the more "tricky" freedom for choosing "good" parameters and the more work to implement such a function.

Experiments made it also apparent that resorting to a very fine sampling of the medial axis and, thus, to a large amount of clearance lines does not help to make the spirals look smoother. Rather, the finer the sampling, the more the resulting spirals seemed to "converge" to some limit curve. This can be understood if one analyzes the mathematics of the impulse propagation in the neighborhood of a sharp corner of a spiral: For parallel clearance lines the propagation of the impulse obeys the intercept theorem and, thus, the wavefront locally follows a straight-line segment even if the sampling rate is increased significantly.

As a rule of thumb, using up to five times as many clearance lines as we used in our sample Fig. 10 seems to yield decent results. The sampling can be coarser along straight-line edges of the medial axis of P and should be finer along conic edges.

In any case, a purely polygonal path will always show directional discontinuities at its corners, no matter how much effort were invested in an improved impulse propagation. Hence, it seems natural to resort to an approximation of our polygonal spirals by higher-order primitives if the smoothness of the path is of a concern.

Of course, an approximation of a spiral path should still have Δ as maximum step-over distance, and it must not leave the pocket P. These two requirements place constraints on an approximation. Suppose that our spiral path $\mathcal{S}(P,\Delta_1)$ has a maximum step-over distance of Δ_1. If we can guarantee $H(\mathcal{S}(P,\Delta_1),\mathcal{A}) \leqslant \Delta_2$ for its approximation $\mathcal{A}$ then we know that $\mathcal{A}$ has a maximum step-over distance of $\Delta_1+\Delta_2$. Hence, we can proceed as follows: (1) We choose an approximation threshold ε with $0<\varepsilon<\Delta$, (2) we compute $\mathcal{SP} := \mathcal{S}(P,\Delta-\varepsilon)$, and (3) we compute an approximation $\mathcal{A}$ of $\mathcal{SP}$ such that $H(\mathcal{SP},\mathcal{A}) \leqslant \varepsilon$. This approach ensures that the maximum step-over distance Δ is not exceeded by $\mathcal{A}$. In order to guarantee that $\mathcal{A}$ does not leave P it suffices to ensure that the approximation of the last lap stays locally on the left side of that lap. All other laps can be approximated using a symmetric tolerance.

For this work we used the PowerApx-package (Heimlich & Held, 2008; Held & Kaaser, 2014). Amongst other things, it supports the approximation of polygonal chains by biarcs and cubic B-splines, thus achieving G^1 continuity or even C^2 continuity. The approximation curve $\mathcal{A}$ is guaranteed to lie within a user-specified tolerance of the original input $\mathcal{SP}$, and $\mathcal{SP}$ is guaranteed to lie within a user-specified tolerance of $\mathcal{A}$. Hence, a bound on the Hausdorff distance between $\mathcal{A}$ and $\mathcal{SP}$ can be established. These tolerances can be either symmetric, asymmetric, or even one-sided. (A one-sided tolerance is used for the last lap of $\mathcal{SP}$.)

In Fig. 10(b) we see the approximation of the spiral path shown in Fig. 10(a) by a cubic B-spline. For the sake of simplicity, we subjected the actual spiral of Fig. 10(a) to the approximation, without reducing the maximum step-over distance Δ. Hence, although we used a tiny approximation threshold ε which, if plotted, would hardly exceed the pen width used for drawing ∂P, the step-over distance of the resulting cubic B-spline might exceed Δ ever so slightly. Other sample cubic B-Spline spirals are shown in Figs. 1 and 12.

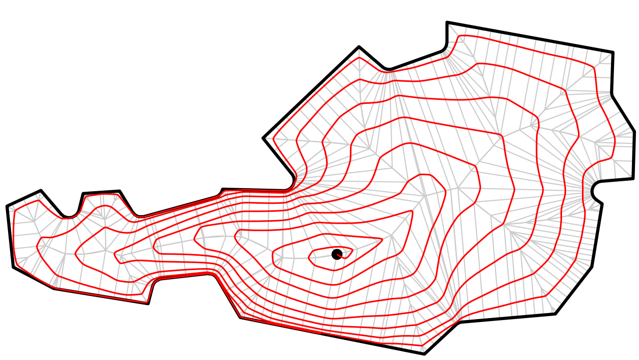


Fig. 12. Cubic B-spline approximation of the polygonal spiral path of Fig. 11(b).

7. Double and composite spirals

7.1. Double spiral

All spirals discussed so far have one fact in common: They start at some point of the medial axis and end at the boundary ∂P of the pocket P. We now generalize our approach to a double spiral that starts and ends at the boundary ∂P.

As in the case of a single spiral, the user-specified step-over Δ implies a certain minimum number of wavefronts. For the sake of descriptional simplicity, suppose that this number is odd and that we have $2k+1$ wavefronts $w(t_0), w(t_1), \ldots, w(t_{2k})$, with $w(t_0)$ equal to r and $w(t_{2k})$ equal to ∂P. We use the algorithm of Section 5 to compute one single "inner" spiral with maximum step-over 2Δ which starts at r and ends at v_0 on ∂P. Let $L_1, L_3, \ldots, L_{2k-1}$ denote the successive laps of this spiral. Hence, L_1 starts at r and ends at the intersection q of $w(t_2)$ with $\overline{rv_0}$, L_3 starts at q and ends on $w(t_4)$, and so on. In particular, L_{2k-1} ends at v_0 on ∂P.

Let L_{2k+1} be identical to ∂P. For $i \in \{1,3,\ldots,2k-1\}$, we plant an impulse at every vertex of lap L_i that moves along $\mathcal{T}_r$ towards the leaves of $\mathcal{T}_r$, starting on L_i at time $t := 0$ and reaching L_{i+2} at time $t := 1$. Stopping the impulse at time $t = 1/2$ yields the vertices of the laps $L_2, L_4, \ldots, L_{2k}$ of the "outer" spiral, where L_2 starts at q and L_{2k} ends at v_0 on ∂P. As for a single spiral, the positions of the end-points of L_{2k-1} and L_{2k} on ∂P can be adjusted to meet specific needs. In Fig. 13(a), the outer spiral and the vertices of the inner spiral are shown.

In order to connect the start of L_2 at q with the start of L_1 at r we move from the vertices of L_1 towards r along $\mathcal{T}_r$ for a distance of Δ, thus obtaining candidate corners of a polygonal path that connects L_1 and L_2. (This is similar to the generation of L_1 in Section 5.) We note that this construction ensures that the resulting double spiral is not self-intersecting and respects the maximum step-over Δ. In Fig. 13(b), a full double spiral is shown for our sample pocket.

Of course, the smoothing operations of Section 6.1 are applicable again. Fig. 14 shows the outer polygonal spiral computed according to the modified impulse propagation and smoothing, and Fig. 14 shows an approximation of the full double spiral by a cubic B-spline. The outer spiral was stopped in the upper-left corner of ∂P. (Again, we used the POWERAPX package (Heimlich & Held, 2008; Held & Kaaser, 2014) to obtain this approximation.)

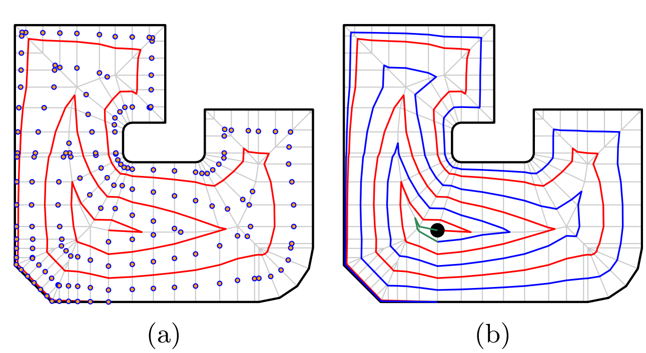


Fig. 13. (a) The vertices of the outer spiral (highlighted by blue circles) are placed halfway between the corresponding vertices of the inner spiral. (b) Final double spiral consisting of the inner spiral (red), outer spiral (blue) and connecting polygonal path (green).

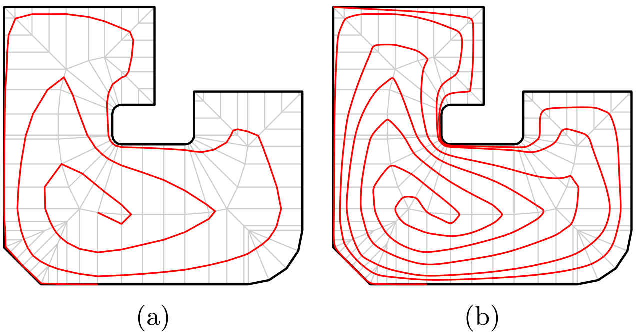


Fig. 14. (a) Outer polygonal spiral generated based on the modified impulse propagation. (b) Resulting double spiral as a cubic spline.

7.2. Composite spiral path

As suggested in Held and Spielberger (2014), we can decompose a complex (possibly multiply-connected) pocket into simpler sub-pockets and then compute spiral paths within these sub-shapes. The obvious disadvantage of having multiple spirals is the need to link them into one path. In general, this will require the application to pause during these linking portions of the path, like during retraction moves in machining.

We now employ our machinery for computing single and double spirals to obtain composite spiral path. In a nutshell, we compute suitable spirals within every sub-pocket and splice them together appropriately.

Let $\mathcal{D}$ be a set of sub-pockets obtained by decomposing the pocket P by some means. (See, e.g., Held & Spielberger, 2014 for methods to achieve a decent decomposition.) The common boundary between two sub-pockets is called a decomposition edge. We derive a graph G from $\mathcal{D}$ in the following way: The nodes of G represent the sub-pockets of $\mathcal{D}$. Two nodes are linked by an edge of G if the corresponding sub-pockets share a decomposition edge. For the sake of descriptional simplicity we assume that G is a tree. (Recall that we can use bridge edges to convert a multiply-connected shape into a simply-connected shape.) A sample pocket together with its decomposition and resulting graph G are shown in Fig. 15(a).

We start with computing two leaf nodes v_1, v_2 of G which determine the diameter of G. That is, no path in G between any pair of nodes of G contains more edges than the path between v_1 and v_2. The sub-pockets that correspond to v_1 and v_2 are the only ones in which a single spiral is computed, cf. Fig. 15(b). In every other sub-pocket we generate a double spiral. Now recall that we can let our spirals end at arbitrary points on the pocket boundary. In particular, we can make them start and end on the decomposition edges. This makes it easy to link all spirals within the sub-pockets that correspond to the diameter path between v_1 and v_2 into one composite spiral path.

In a similar way, the other spirals can be linked to paths and spliced into the composite spiral path obtained so far. We do not go into details of the linking since the actual geometry of the linking portions of the spirals depends on the geometry of the decomposition edges. (For the sake of simplicity, in our own work we use straight-line segments as decomposition edges.) See Fig. 15(c) for a full composite spiral path.

8. Discussion and conclusion

We introduce a simple and easy-to-implement algorithm for computing polygonal spirals to cover planar shapes bounded by straight-line segments and circular arcs. The paths do not self-intersect and respect a user-specified maximum step-over distance. Smoothing heuristics help to prevent excessively sharp corners, thus avoiding a drastic variation of the curvature. If our paths are applied in an HSM application then smoothing will also help to avoid a rapid change of the engagement angle. And, indeed, at least our single spirals have already mastered a practical test at the shop-floor level. See Fig. 16 for two pockets machined by our industrial partner using flat-end milling.

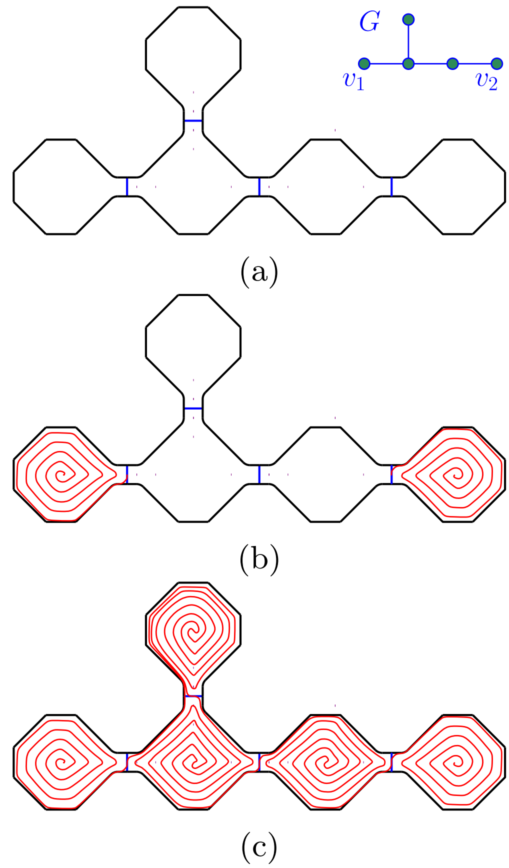


Fig. 15. (a) Subdivision into five sub-pockets and resulting graph G (in the top right corner); (b) first and last single spiral; (c) cubic B-spline as full composite spiral path.

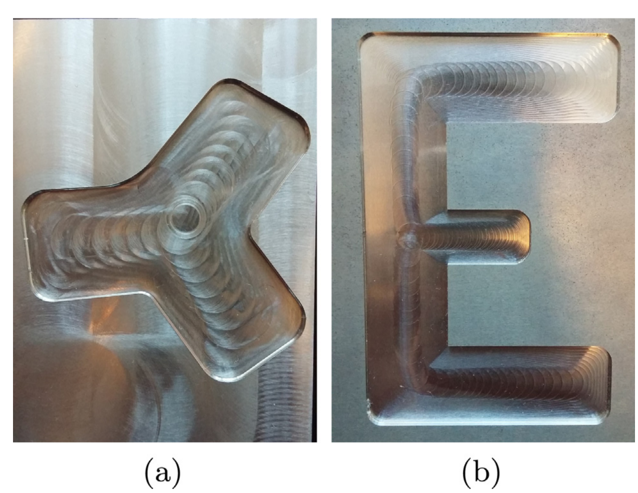


Fig. 16. Two parts machined in aluminum.

Currently we use PowerApx to approximate a polygonal spiral by biarcs or cubic B-splines. While using a package like PowerApx is certainly the simplest approach to boost a polygonal spiral to higher continuity, it is not necessarily the best approach: PowerApx is a general-purpose tool which "blindly" approximates a polygonal path such that specific tolerances are met. As discussed, this allows to obtain smooth spirals that still respect a user's maximum step-over distance Δ. However, it cannot take advantage of the fact that some portions of our spirals would allow a much coarser approximation since we are still far from exceeding Δ. Trying to exploit this additional information for a better approximation that either has fewer approximation primitives or a lower variation of the curvature seems to be a promising avenue for future research.

Conflict of interest

None.

Acknowledgements

We would like to thank the Austrian Science Fund (FWF, Grant P25816-N15) as well as EMCO GmbH (Hallein, Austria) for supporting this research.

References

Abrahamsen, M. (2015). Spiral toolpaths for high-speed machining of 2D pockets with or without islands. In: *Proceedings of the ASME IDETC/CIE 2015 conference* (pp. V02BT03A017–V02BT03A017).

Alt, H., Behrends, B., & Blömer, J. (1995). Approximate matching of polygonal shapes. *Annals of Mathematics and Artificial Intelligence, 13*(3), 251–265. https://doi.org/10.1007/BF01530830.

Banerjee, A., Feng, H.-Y., & Bordatchev, E. V. (2012). Process planning for floor machining of 21/2D pockets based on a morphed spiral tool path pattern. *Computers & Industrial Engineering, 63*(4), 971–979.

Bieterman, M. B., & Sandstrom, D. R. (2002). A curvilinear tool-path method for pocket machining. In: *Proceedings of IMECE2002, ASME international mechanical engineering congress* (pp. 149–158).

Chandler, P., Rasmussen, S., & Pachter, M. (2010). UAV cooperative path planning. In: *AIAA guidance, navigation and control conference*.

Heimlich, M., & Held, M. (2008). Biarc approximation, simplification and smoothing of polygonal curves by means of Voronoi-based tolerance bands. *International Journal of Computational Geometry & Applications, 18*(03), 221–250.

Held, M. (1991). *On the computational geometry of pocket machining. Lecture notes in computer science* (Vol. 500). Springer-Verlag. ISBN 3-540-54103-9.

Held, M. (2001). VRONI: An engineering approach to the reliable and efficient computation of Voronoi diagrams of points and line segments. *Computational Geometry: Theory and Applications, 18*(2), 95–123.

Held, M., & Huber, S. (2009). Topology-oriented incremental computation of Voronoi diagrams of circular arcs and straight-line segments. *Computer-Aided Design, 41*(5), 327–338.

Held, M., & Kaaser, D. (2014). C2 approximation of planar curvilinear profiles by cubic B-splines. *Computer-Aided Design and Applications, 11*(2), 206–219.

Held, M., & Spielberger, C. (2009). A smooth spiral tool path for high speed machining of 2D pockets. *Computer-Aided Design, 41*(7), 539–550.

Held, M., & Spielberger, C. (2014). Improved spiral high-speed machining of multiply-connected pockets. *Computer-Aided Design and Applications, 11*(3), 346–357.

Keller, J. F. (2017). *Path planning for persistent surveillance applications using fixed-wing unmanned aerial vehicles*. PhD dissertation available from ProQuest. AAI10247340. URL <http://repository.upenn.edu/dissertations/AAI10247340>.

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

Romero-Carrillo, P., Torres-Jimenez, E., Dorado, R., & Díaz-Garrido, F. (2015). Analytic construction and analysis of spiral pocketing via linear morphing. *Computer-Aided Design, 69*, 1–10.

Zhao, Z., Liu, B., Zhang, M., Zhou, H., & Yu, S. (2009). Toolpath optimization for high speed milling of pockets. In: *Second international conference on information and computing science* (Vol. 1, pp. 327–330).

Zhao, H., Gu, F., Huang, Q.-X., Garcia, J., Chen, Y., Tu, C., Benes, B., Zhang, H., Cohen-Or, D., Chen, B., et al. (2016). Connected fermat spirals for layered fabrication. *ACM Transactions on Graphics, 35*(4), 100.

Zhao, Z., Wang, C., Zhou, H., & Qin, Z. (2007). Pocketing toolpath optimization for sharp corners. *Journal of Materials Processing Technology, 192*, 175–180.

Zhou, B., Zhao, J., Li, L., & Xia, R. (2016). Double spiral tool-path generation and linking method for complex pocket machining. *Machining Science and Technology, 20*(2), 262–289.