On the Generation of a Spiral Tool Path for High Speed Pocket Machining and Related Applications

Masterarbeit

Zur Erlangung des Diplomingenieurgrades an der Naturwissenschaftlichen Fakultät der Paris-Lodron-Universität Salzburg

Eingereicht von
Stefan de Lorenzo

Gutachter:
Ao.Univ.-Prof. Dipl.-Ing. Dr. Martin Held

Fachbereich:
Computerwissenschaften

Salzburg, November, 2016

Abstract

Based on prior work by Held and Spielberger [CAD 41/7, 2009][CADA 11/3, 2014], we present a modified tool-path generation strategy suitable for high speed machining (HSM). A raw tool path is constructed which is not self-intersecting, respects the initially specified, maximum step-over, starts in the interior, and ends at the boundary of the pocket.

Additionally, we formalize the idea of a dynamic boundary that changes its shape during the creation of the cutter trajectory, and results in a tool path with better kinematic features. By inserting bridges into the original pocket, the algorithm is able to handle pockets that include an arbitrary number of islands.

In order to generate a $\mathcal{G}^1$-continuous tool path, we approximate the raw cutter trajectory through a series of biarcs using the PowerApx-package developed by Held and Heimlich [IJCGA 18/3, 2008].

Complex pocket regions, i.e., cavities that contain a vast amount of narrow bottlenecks, result in exceedingly long tool paths. Thus, a decomposition scheme is discussed that subdivides the initial pocket into several sub-pockets which are milled out individually. This procedure tends to reduce the overall step-over variation, but has the drawback that each sub-pocket corresponds to a tool retraction.

Finally, the possibility of creating a double spiral, i.e., a curve that starts and ends at the pocket boundary, while simultaneously keeping all the aforementioned properties, is discussed. This scheme is unsuited for pocket milling, due to the fact that the cutter is highly engaged most of the time, but may find use in other applications, such as spray paintings, aerial surveillance, or path finding algorithms for rescue missions.

Acknowledgment

Foremost, I would like to thank my supervisor Prof. Martin Held for his continuous support, motivation, and patience. He always had time for insightful discussions, and spent a great amount of time with proof-reading this thesis. I could not have imagined having a better supervisor.

Furthermore, I would like to express my gratitude to the EMCO GmbH. In particular, I would like to thank Christian Klapf and Ronald Hinterbichler who helped my greatly in realizing a practical implementation.

Last but not least, I would like to thank my family and my girlfriend for supporting me spiritually throughout my studies.

Contents

1. Introduction

1.1. Motivation

Pocket milling is one of the most common tasks carried out in modern numerical control (NC) machining. In recent years, high speed machining (HSM) of pockets has become more and more popular. It shortens the machining time, and maintains a high material removal rate. HSM is considered one of the key technologies for higher productivity.

Two major approaches are widely used in conventional pocketing applications, namely, contour- and direction-parallel milling [Elber et al., 2005]. Both strategies yield good results when the length of the resulting cutter trajectory is the primary optimization criteria, but tend to struggle in a HSM environment. Due to the high feed rates and spindle speeds, discontinuities in the tool path become more critical. Sharp corners force the cutter to slow down rapidly and contribute greatly to tool wear. Therefore, it is essential that the cutter trajectory is at least $\mathcal{G}^1$-continuous. In order to prevent jolting, achieving $\mathcal{C}^2$-continuity would be even better [Held and Spielberger, 2009]. Furthermore, minimizing the variation of the cutter engagement is from utmost importance to guarantee a clean cut. New methods that produce cutter trajectories with adequate kinematic features have been developed, but often impose restrictions on the input pocket.

In this thesis a smooth spiral tool path suitable for HSM applications will be constructed from an input set consisting of arcs and straight-line segments. The following section deals with fundamental geometrical concepts as well as terms used in connection with pocket milling.

1.2. The fundamentals of pocket milling

1.2.1. Feeds and speeds

In pocket milling the feed rate defines the speed of the tool relative to the workpiece. It is usually measured in inches per minute (in/min) or millimeters per minute (mm/min). The spindle speed measures the rotational velocity of the tool in revolutions per minute (rpm).

1.2.2. Types of path control

There exist different types of path control based on the number of independently controllable axes. Figure 1.1 illustrates these different milling types.

- 2D milling supports linear and circular interpolation in a plane. This restricts 2D milling to a narrow field of possible applications.
- In 3D milling linear interpolation using all three available axes is possible, circular interpolation is only feasible in one of the three coordinate planes at a time.
- In addition to the three independent axes in 3D milling, 5D milling also supports two rotational axes. This offers a high amount of freedom and makes 5D milling well-suited for machining complex parts. Despite this advantage 5D milling is not widespread. This is due to the fact that it is not trivial to determine the cutter position and avoid collisions.
- 2½D milling is a hybrid form between 2D and 3D milling. Machining three dimensional shapes is possible, but only two pre-selected axes are able to move at the same time. Several working planes are introduced which are milled out one after another.

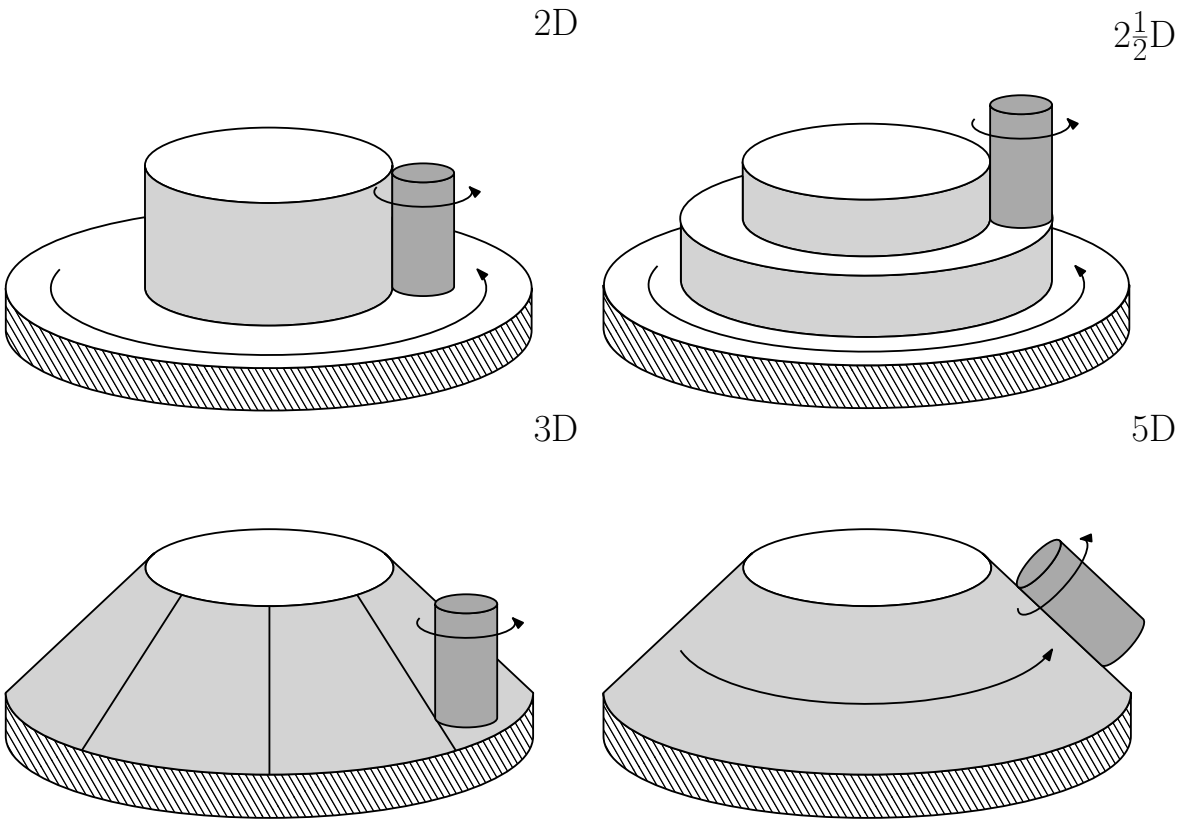


Figure 1.1.: Different types of NC path control.

1.2.3. Pocket representation

As mentioned before, a pocket in 2½D milling is represented as a series of working planes. From a geometrical standpoint, each of these working planes can be interpreted as an area

bound by a closed curve (the outer boundary) and possibly multiple inner contours (the islands). These curves are not self-intersecting and disjoint. Islands are areas inside the workpiece that should be left out. Figure 1.2 depicts a pocket containing two islands.

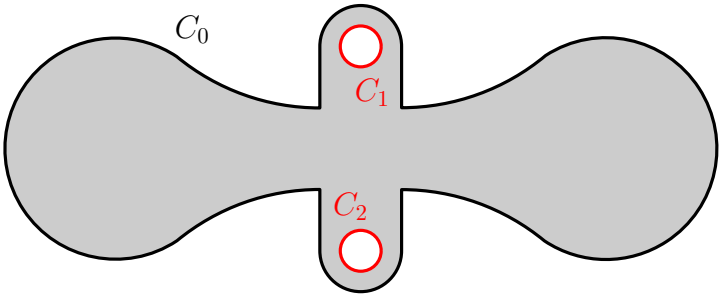


Figure 1.2.: A pocket outlined by the outer boundary C_0 with two islands C_1 and C_2.

1.2.4. Step-over

The step-over (or depth of the cut) measures the distance between two adjacent cutter passes. While conventional strategies have a constant step-over, that is, every pair of successive offset curves has the same distance, high speed machining approaches usually have varying step-over values. This is why a maximum step-over δ is specified, i.e., the distance between two neighboring laps has to be less than δ. See Figure 1.3 for an example of the depth of the cut at a specific point on the cutter pass.

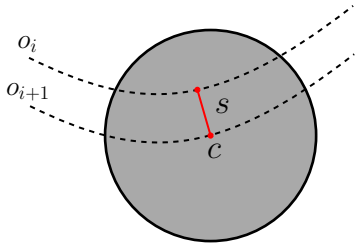


Figure 1.3.: The step-over s at point c on the cutter pass o_{i+1}.

1.2.5. Engagement angle

The area of contact between the cutter and the workpiece is defined by the engagement angle (see Figure 1.4). It directly impacts the cutting force. Abrupt changes of the direction of the tool path lead to rapid loading and unloading of the cutter. As a consequence, the engagement angle as well as the tool wear highly increases, the quality of the resulting surface suffers, and efficiency as well as productivity are lost.

1.2.6. Traditional milling strategies

Two strategies are prevalent in conventional $2^1/_2$D milling, namely, contour- and direction-parallel milling (see Figure 1.5). Both yield tool paths that are well suited for most standard

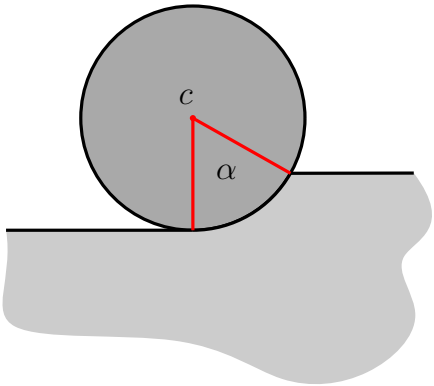


Figure 1.4.: The engagement angle α at c, the area shaded in light gray depicts the area not yet machined.

applications. The resulting cutter trajectory can be processed on every machine that supports linear as well as circular interpolation. Conventional pocketing has been studied widely, and robust algorithms for non-convex pockets without islands exist [Dragomatz and Mann, 1997].

Contour-parallel milling uses a set of successive, evenly spaced offset elements. These offsets can be created by using the Minkowski difference or Voronoi diagrams. The distance between those elements is defined by the step-over. After the offsets are connected this approach yields a spiral-like tool path.

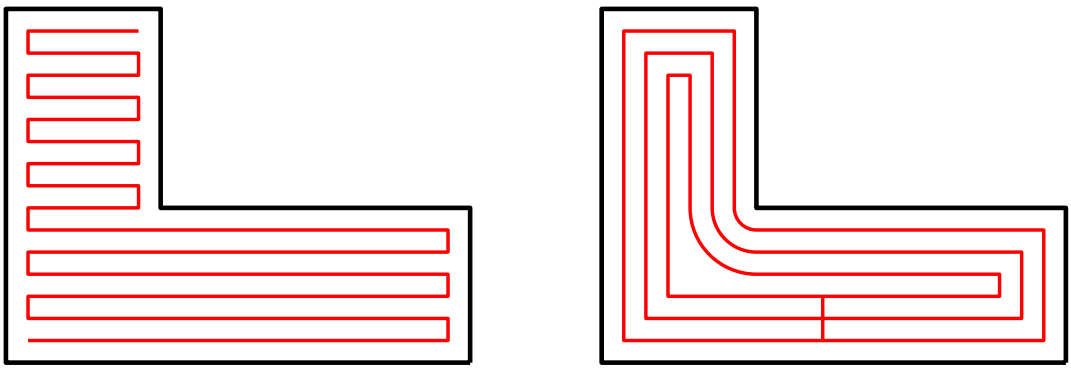
In direction-parallel milling a reference line is chosen initially. Parallel to this line, a series of uniformly distributed offset line segments is generated. Depending on how these line segments are generated two approaches can be distinguished.

In Zigzag milling the cutter passes the successive offsets alternating from left-to-right and right-to-left, with each alternation the cutter switches between conventional and climb milling, which contributes to tool wear. Zig milling is characterized by the fact that all offset segments are cut in the same direction, that is, either all are machined from left-to-right or from right-to-left. This strategy has the drawback that for every offset line that is created one retraction, i.e., a non-cutting tool movement, is necessary in order to process the surface. For a more detailed description of contour-parallel and direction-parallel milling see [Held, 1991].

The focus of the algorithms mentioned above is to minimize the overall tool-path length in order decrease the machining time. This is the reason why discontinuities in the resulting curve are accepted.

1.2.7. High speed machining

High speed machining caused a paradigm shift regarding the optimization criteria of tool paths. The continuity of the corresponding curve is now the most important benchmark, in contrast to the tool-path length in conventional pocket milling. Contour-parallel and direction-parallel strategies are not well suited for HSM applications, because the respective cutter trajectories include discontinuities. From a kinematic standpoint such a discontinuity



(a) (b)

Figure 1.5.: A simple pocket machined based on a contour-parallel (a) and a direction-parallel (b) strategy.

Notation	Description
S	A set of input sites in $\mathbb{R}^2$
$d(s,t)$	Euclidean distance between $s \in \mathbb{R}^2$ and $t \in \mathbb{R}^2$
$d(s,S)$	$\inf\{d(s,t) : t \in S\}$
$d(S,T)$	$\inf\{d(s,t) : s \in S, t \in T\}$, with $S \subseteq \mathbb{R}^2$ and $T \subseteq \mathbb{R}^2$
$b(p,q)$	Bisector between s and t, with $s,t \in S$
$\mathcal{VR}(s,S)$	Voronoi region of the input site $s \in S$
$\mathcal{VP}(s,S)$	Voronoi polygon defined by the input site $s \in S$
$\mathcal{VD}(S)$	Voronoi diagram of S
P	A region in $\mathbb{R}^2$ outlined by straight-line segments and circular arcs
$\mathcal{MA}(P)$	Medial axis of P
$HP(p,v)$	The half-plane $\{q \in \mathbb{R}^2 : qv \geq pv\}$, where v is a vector and $p \in \mathbb{R}^2$
v^{CCW}	Rotation of the vector v by $90°$ around the origin
v^{CW}	Rotation of the vector v by $-90°$ around the origin

Table 1.1.: Basic definitions and notations.

can be interpreted as a deceleration to null feed rate followed by a rapid acceleration. Apart from that, every discontinuity contributes to high variations of the tool engagement.

1.3. Basic geometric concepts

In Table 1.1 definitions and notations which are used in the course of this work are listed.

1.3.1. Voronoi diagram of points

The Voronoi diagram is a versatile tool in computational geometry. It is named after the Russian mathematician Georgy Voronoy. Intuitively, a Voronoi diagram consists of a set of regions, where each region corresponds to exactly one input site and contains all points which are nearest to this site. If the Voronoi diagram of a set of points is known, then it is possible

to solve a series of problems in linear time, for example:

- Triangulation
- Euclidean minimum spanning tree
- All nearest neighbor
- Maximum inscribed circle

In the following this intuitive characterization of a Voronoi diagram of points is formalized. The Definitions 1.1-1.7 follow the exposition of Held and Huber [2009].

Definition 1.1 (Voronoi region)**.** Let S be a set of input points. The Voronoi region $\mathcal{VR}(s, S)$ of a point $s \in S$ is defined as

$$\mathcal{VR}(s, S) := \{t \in \mathbb{R}^2 : d(s, t) \leq d(s, S)\}.$$

Definition 1.2 (Voronoi polygon)**.** The Voronoi polygon $\mathcal{VP}(s, S)$ of a point $s \in S$ is the boundary of the corresponding Voronoi region, i.e,

$$\mathcal{VP}(s, S) := \partial\mathcal{VR}(s, S).$$

Definition 1.3 (Voronoi diagram)**.** The Voronoi diagram $\mathcal{VD}(S)$ of a set S of points is defined by

$$\mathcal{VD}(S) := \bigcup_{s \in S} \mathcal{VP}(s, S).$$

An example of a Voronoi region as well as a Voronoi polygon can be found in Figure 1.6.

Definition 1.4 (Voronoi node)**.** A Voronoi node is a point on the Voronoi diagram that is equidistant to at least three different sites of the input set.

Definition 1.5 (Bisector)**.** A bisector between two points $s, t \in S$ is defined by

$$b(s, t) := \{p \in \mathbb{R}^2 : d(p, s) = d(p, t)\}.$$

1.3.2. Generalized Voronoi diagram

A generalized version of the Voronoi diagram is also able to handle straight-line segments and arcs as input sites, in addition to points. It restricts the individual sites to a corresponding cone of influence to avoid two dimensional bisectors.

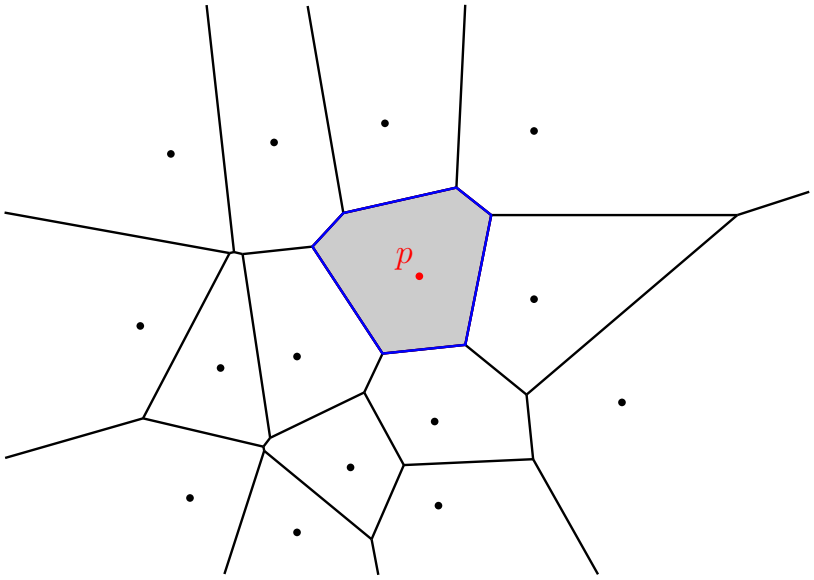


Figure 1.6.: The Voronoi region $\mathcal{VR}(p, S)$ (shaded in gray) and the Voronoi polygon $\mathcal{VP}(p, S)$ (outlined in purple) of a point p (highlighted in red).

Definition 1.6 (Cone of influence)**.** For an input site s the cone of influence $\mathcal{CI}(s)$ is defined as

$$\mathcal{CI}(s) := \begin{cases} \mathbb{R}^2 & \text{if } s \text{ is a point,} \\ HP(p, q-p) \cap HP(q, p-q) & \text{if } s \text{ is a line segment } \overline{pq}, \\ HP\left(c, (p-c)^{\text{CCW}}\right) \cap HP\left(c, (q-c)^{\text{CW}}\right) & \text{if } s \text{ is an arc centered at } c, \text{ with start point } p \text{ and end point } q. \end{cases}$$

The concept of the cone of influence is illustrated in Figure 1.7.

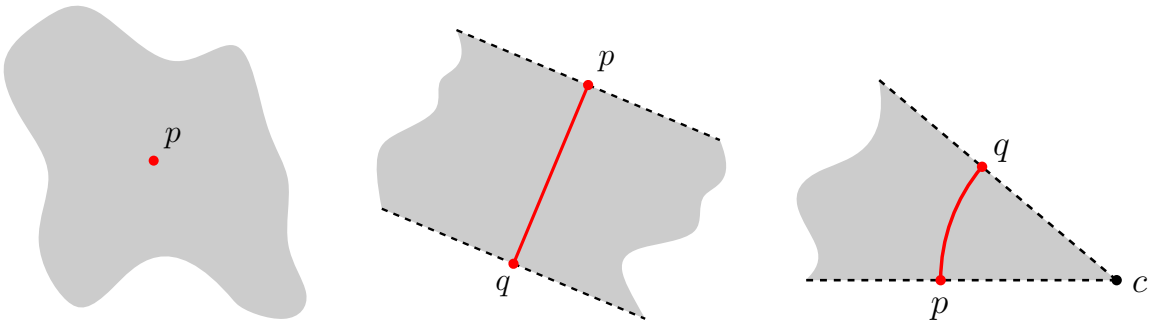


Figure 1.7.: The cone of influence of (a) a point, (b) a line segment, and (c) a circular arc.

Definition 1.7 (Voronoi region)**.** The Voronoi region $\mathcal{VR}(s, S)$ of a site $s \in S$ is given as

$$\mathcal{VR}(s, S) := \mathrm{cl}\{q \in \mathrm{int}(\mathcal{CI}(s)) : d(q, s) \leq d(q, S)\}.$$

Clearly, all Voronoi regions of input sites are (simply) connected, and the bisector between pairs of sites are given by portions of conics. That is, all bisectors are elliptic, hyperbolic, or parabolic arcs. For more information on the generalized Voronoi diagram see [Held and Huber, 2009].

1.3.3. Medial axis

The concept of the medial axis was first described in [Blum, 1967]. It is based on the notion of a clearance disk, inseparably connected with the Voronoi diagram, and can be extracted from it in linear time.

Definition 1.8 (Clearance disk)**.** A clearance disk of a point $p \in \mathbb{R}^2$ relative to a set S of input sites is the disk with maximum radius centered at p such that no point of S is contained in its interior.

Definition 1.9 (Medial axis)**.** The medial axis of an area P outlined by circular arcs and straight-line segments is the union of all points whose clearance disks touch the border of P in at least two distinct points.

1.3.4. Curves and continuity

The notion of a curve is important to describe the kinematic features of a tool path like the speed, curvature or continuity. Note that the Definitions 1.10-1.17 are taken from [Held, 2016].

Definition 1.10 (Parameterization of a curve)**.** A parameterization of a curve is a function $\gamma : S \to \mathbb{R}^2$ with $S \subseteq \mathbb{R}$, and $\gamma(S)$ is called the parametric curve parameterized by γ.

It is crucial to distinguish between the parameterization of a curve and the parametric curve, because different parameterizations correspond to a specific curve. For the sake of simplicity we will refer to a parameterization of a curve as curve. A curve is bound by two different points $a, b \in \mathbb{R}^2$, the start and endpoint respectively. If $a = b$, then the corresponding curve is called closed. A parameterization of a curve is called continuously differentiable if it is continuously differentiable in every point and is said to be regular if its first derivative is non-zero everywhere, and singular otherwise.

Definition 1.11 ($\mathcal{C}^r$-parameterization)**.** If $\gamma : I \to \mathbb{R}^n$ is r times continuously differentiable, then γ is called a parametric curve of class $\mathcal{C}^r$. It is called smooth if it is of class $\mathcal{C}^\infty$.

By assigning a parameterization of a curve to a specific differentiability class it is possible to measure its quality regarding continuity. Continuity depends on which parameterization of a specific curve is chosen.

There is also the notation of geometric continuity. In general a parameterization of a curve is $\mathcal{G}^k$-continuous if it can be reparameterized to have $\mathcal{C}^k$-continuity, that is, $\mathcal{C}^k$-continuity implies $\mathcal{G}^k$-continuity.

Definition 1.12 ($\mathcal{G}^1$-continuous)**.** Let $\beta : [a, b] \to \mathbb{R}^n$ and $\gamma : [c, d] \to \mathbb{R}^n$ be $\mathcal{C}^1$-curves, with $\beta(b) = \gamma(c) =: p$. β and γ join in a $\mathcal{G}^1$-continuous manner at p if

$$0 \neq \beta'(b) = \lambda \cdot \gamma'(c) \text{ for some } \lambda \in \mathbb{R}^+.$$

Definition 1.13 (Speed)**.** Let $\gamma : I \to \mathbb{R}^n$ be a $\mathcal{C}^1$-curve. The speed of γ at $t \in I$ is given by $||\gamma'(t)||$ and $\gamma'(t)$ is the corresponding velocity vector. Furthermore, γ is at unit speed if $||\gamma'(t)|| = 1$ for all $t \in I$.

A concepts that describes the curve at a point relative to its local neighborhood is the curvature. Intuitively, the curvature describes how quickly the curve changes its direction at a specific point relative to its speed.

Definition 1.14 (Curvature)**.** Let $\gamma : I \to \mathbb{R}^2$ be a $\mathcal{C}^2$-curve with $\gamma(t) = (x(t), y(t))$. The curvature at t is given by

$$\kappa(t) := \frac{x(t)'y(t)'' - y(t)'x(t)''}{(x(t)')^2}.$$

1.3.5. Distance measures

There are several ways to determine the distance between two curves. We are going to rely on the so-called (undirected) Hausdorff distance.

Definition 1.15 (Directed Hausdorff distance)**.** Let $A, B \subset \mathbb{R}^2$ with $A \neq \emptyset$ and $B \neq \emptyset$. Then the directed Hausdorff from A to B is defined as

$$h(A, B) := \sup_{a \in A}\{\inf_{b \in B}\{d(a, b)\}\},$$

where $d(a, b)$ is the Euclidean distance between a and b.

Lemma 1.16. In general $h(A, B) \neq h(B, A)$, i.e., the directed Haussdorff distance is not symmetric.

Definition 1.17 (Hausdorff distance)**.** If, again, A and B are two non-empty sets in $\mathbb{R}^2$, then the (undirected) Hausdorff distance $H(A, B)$ between A and B is given by

$$H(A, B) := \max\{h(A, B), h(B, A)\}.$$

1.4. Related work

As mentioned before, conventional tool paths yield discontinuities, that is, sharp corners. At these corners the feed rate has to be reduced to zero. That leads to a rapid deceleration followed by a rapid acceleration. Due to this, feed rate reduction the machining time increases. That means, in order to change the kinematic behavior of the machine the shape of the tool path has to be adjusted. The goal of this optimization process is to minimize the overall machining time.

One way of generating a path suitable for HSM is to rely on a conventional strategy and optimize sharp corners that were produced in the process, afterwards. Pateloup et al. [2004] took a conventional tool path and optimized discontinuities locally. To achieve this, the feed rate has to be predicted, particularly at points of high curvature. Furthermore, the pocket must be covered completely, i.e., there must not remain any material between two successive cutting passes.

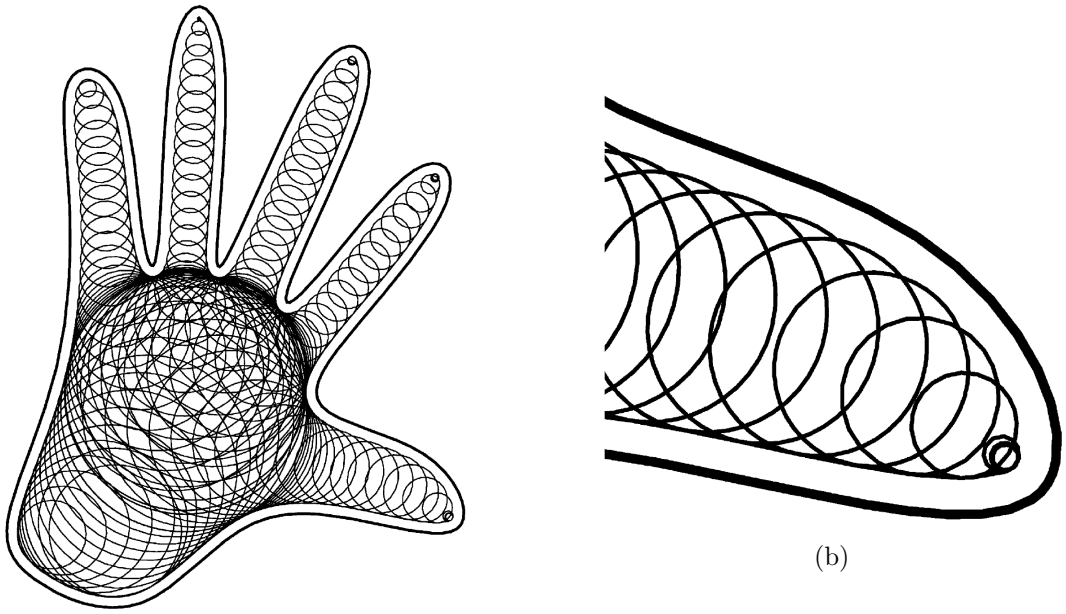
If the influence of the tool-path geometry on the kinematic behavior of the machine should be measured, then several features have to be considered, e.g., the acceleration capabilities and the maximum speed of each axis. It is insufficient to replace all corners with circular arcs to generate a path suitable for HSM applications. In order to reduce the machining time, the computed tool path has to be $\mathcal{C}^2$-continuous.

Zhao et al. [2007] produced a $\mathcal{G}^1$-continuous cutter trajectory by inserting biarcs in contour-parallel tool paths. This approach makes it possible to machine with step-over values that would otherwise lead to unmachined areas inside the pocket. Biarc transitions shorten the tool path compared to single-loop or double-loop transitions. Several self-intersections are introduced, and the engagement of the cutter changes rapidly in sections of the tool path where biarcs are inserted.

There are various approaches which use Voronoi diagrams during tool-path computation. Elber et al. [2005] used the medial axis, derived from the boundary of a $\mathcal{C}^1$-continuous pocket that is free of islands to generate a smooth tool path. The medial axis is used to acquire successive, equally spaced clearance disks. Trivially, these clearance disks minimize the radial acceleration, which is especially important for HSM applications.

Neighboring disks are connected through one of their common supporting lines. A supporting line of a curve divides the plane such that the curve itself lies completely inside one of the two resulting half-planes. Only the two outer supporting lines are considered during

this procedure. Depending on which one is chosen the orientation of the tool path changes. Finally the supporting lines are trimmed at the corresponding contact points. An example of a cutter trajectory produced by this scheme is displayed in Figure 1.8.



(a) (b)

Figure 1.8.: A tool path inside a hand-shaped pocket ([Elber et al., 2005]).

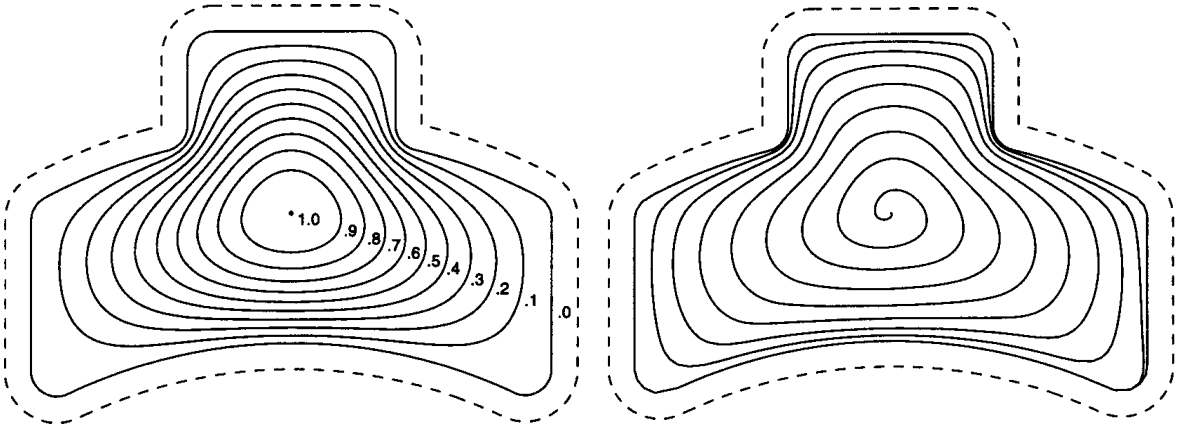
It is folklore that the medial axis of a pocket without islands forms a tree. Each leaf corresponds to one tool retraction. They can be avoided by moving the centers of the individual circles halfway between the medial axis and the boundary of the pocket. As a consequence of this approach, many additional self-intersection are introduced in the final tool path. In either way the constructed tool path includes a high number of non-cutting tool movements. More specifically, material is only removed about half of the time.

Otkur and Lazoglu [2007] proposed a strategy to predict cutting forces in trochoidal milling. This is helpful to identify regions of small engagement of the cutter and workpiece, where the feed rate can be incremented to increase efficiency. An analytical method for pockets without holes, as well as a numerical solution for general pockets are discussed.

Bieterman and Sandstrom [2002] presented an approach based on partial differential equations (PDEs). To be precise, a so-called scalar elliptic second order PDE is solved. According to the authors, this function was chosen based on personal experience rather than an actual machining model. This line of thought is derived from the fact that elliptic PDEs have nice smoothening properties.

The actual problem space is the two-dimensional pocket region. Its border contour is offset inwards. This is done by evaluating the PDE at different points in time (see Figure 1.9), such that the maximum step-over distance is respected. At some moment the curve described by the PDE will degenerate into a single point. This point is referred to as the center of

the pocket. An analytical solution of this problem is only possible for a limited number of input contours, this is why a numerical approach has to be taken. Using the finite element method (FEM) the problem domain is divided into smaller sub-problems that are solved individually. Afterwards, they are assembled into a system of equations [Farlow, 2012]. Once the solution contours are obtained they are connected through radial interpolation. This is the reason why only star-shaped pockets can be handled.



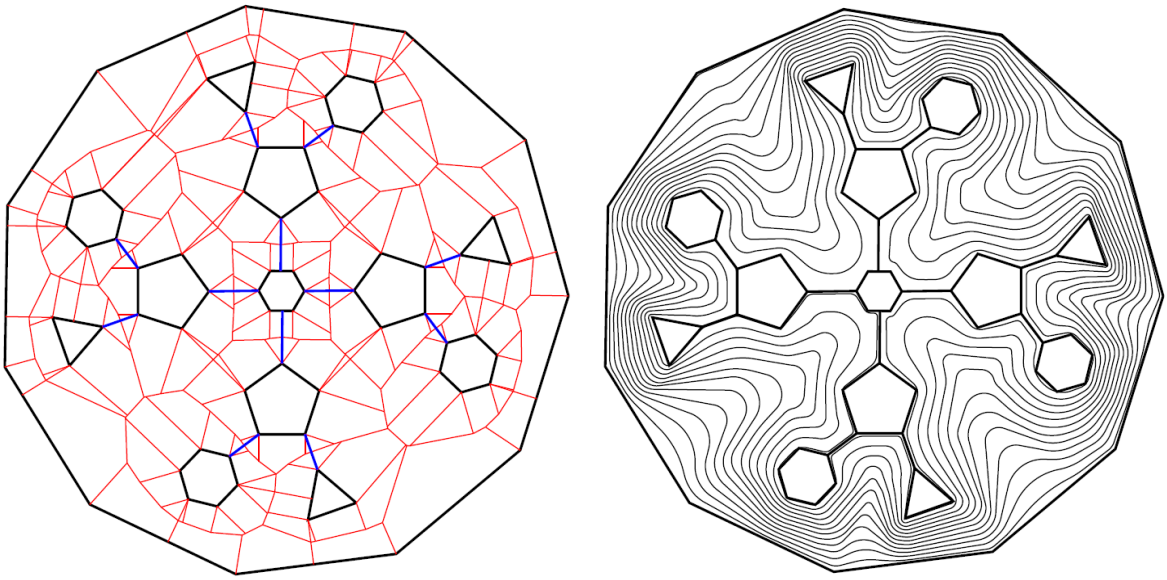
(a) (b)

Figure 1.9.: In (a) a series of solution contours is shown. Figure (b) illustrates the final tool path ([Bieterman and Sandstrom, 2002]).

Finally, the discrete tool path is smoothed. This smoothing operation results in a longer path, it also propagates discontinuities and regions of high curvature of the input boundary into the interior of the path. Path representation by splines yields a more compact output than an ordinary tool-path description through successive straight-line segments.

In order to generate a series of successive solution contours, Banerjee et al. [2012] used a similar approach solving the eigenvalue problem for an elliptic PDE. Neighboring contours are connected based on a winding angle parameterization. In addition, a method is proposed which makes it possible to deal with a single island near the center of the pocket. This is done by applying different boundary conditions to the inner and outer curve. The cut patterns are represented through biarcs [Bolton, 1975] and arc splines [Meek and Walton, 1992]. The reasoning behind this choice is based on the fact that NC controllers that support robust spline interpolation are not prevalent. This path representation yields $\mathcal{G}^1$-discontinuities at the intersection points of adjacent biarc and arc spline segments. Based on an existing mechanic force model [Kline et al., 1982] the cutting forces are calculated, and the feed rate can be optimized.

Chuang and Yang [2004] developed a Laplace-based tool-path generation method for arbitrary shaped pockets which supports selected configurations of islands. The boundary is divided into different segments. This makes it possible to apply a Laplace parameterization which basically resembles a grid. Afterwards, this grid is evened out. By linking together isoparametric loops the final tool path is produced.



(a) (b)

Figure 1.10.: A polygon with 13 islands (a) as well as the resulting spiral (b) is shown ([Abrahamsen, 2015]).

Spiral tool-path generation strategies based on PDEs have difficulties in controlling the distance between two adjacent contours. This means, it is hard to guarantee machining parameters during the concrete cutting process [Zhou et al., 2015].

Abrahamsen [2015] constructed a polyline spiral inside a pocket consisting of straight-line segments. Firstly, the medial axis tree is calculated. Afterwards, a sequence of uniformly distributed wavefronts is computed. This process basically represents an interpolation between a point inside the initial pocket and the corresponding boundary. Each wavefront is made up of a number of corners, that is, a number of points that are situated on the individual edges of the medial axis. Through manipulating the positions of these corners a spiral path is generated, in other words, an interpolation between consecutive wavefronts takes place. The resulting path is then smoothed by inserting circular arcs at sharp corners.

In order to deal with pockets that include a single island, an alternative approach is taken. Theory tells us that the Voronoi diagram of such a setup consists of a single cycle, the so-called central cycle, as well as a number of trees growing out of it. It is possible to define wavefronts for each subtree separately. By applying an interpolation procedure between neighboring wavefronts, a connected tool path is created. Whenever multiple islands are encountered, they are merged by inserting bridges, i.e., straight-line segments between them (see Figure 1.10). This procedure generates good results if the islands are clustered around the center of the pocket, but yields a sub-optimal cutter trajectory otherwise.

Zhao et al. [2016] generated multiple space filling curves (so-called Fermat spirals) inside an arbitrary, singly-connected 2D region. Due to the fact that the start as well as the end points of these spirals can be chosen arbitrarily over the boundaries of the respective sub-

regions, these curves can be easily merged into one another. In a final optimization step the connected Fermat spiral is smoothed. Thus, a connected, smooth curve is produced that covers the interior of the initially-specified 2D region.

2. Theoretical Approach

Held and Spielberger [2009] proposed a unique tool-path generation method for $2^1/_2$D milling that produces a $\mathcal{G}^1$-continuous cutter trajectory for (possibly) non-convex pockets with or without islands. This chapter attempts to give a basic intuition about this algorithm and discusses its key ideas.

2.1. Preliminaries

It is folklore that a pocket P describes an area of a workpiece where material should be removed. For the time being only pockets without islands are considered. In $2^1/_2$D milling the pocket is divided into several working planes. Each one of them is framed by a closed curve which is called the boundary contour or simply border of P, it will be denoted by ∂P.

We assume that ∂P is given by a set of points, straight-line segments, and circular arcs. Based on this representation it possible to derive the generalized Voronoi diagram $\mathcal{VD}(\partial P)$ from ∂P. Inside the boundary contour the medial axis $\mathcal{MA}(P)$ is defined. It is of central importance in this approach. Recall that if the Voronoi diagram is known, then the corresponding medial axis can be determined easily. In Figure 2.1 the medial axis inside a pocket outlined by points, straight-line segments, and circular arcs is shown.

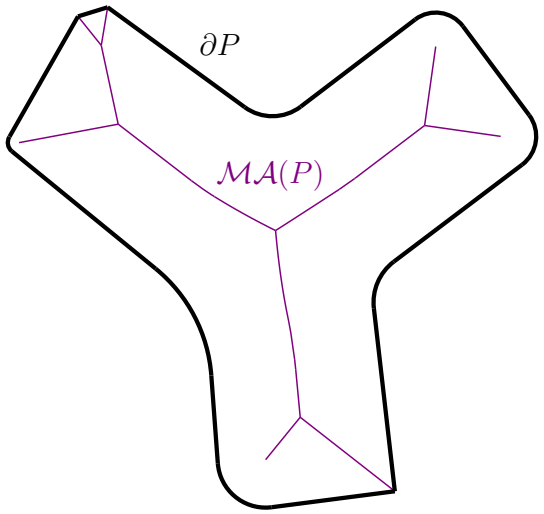


Figure 2.1.: The medial axis $\mathcal{MA}(P)$ (highlighted in purple) inside the pocket P.

2.2. Growing M-disks

Let P be a pocket. For the sake of simplicity, it is assumed that P does not include any islands. Imagine a circular disk centered at a point c on $\mathcal{MA}(P)$ which grows continuously over time. It is obvious that $\mathcal{MA}(P)$ divides this disk into several regions which are called M-parts. Let $p_{c,0}(t), p_{c,1}(t), ..., p_{c,n}(t)$ be the regions that have c on their boundary at time $t \in \mathbb{R}$. The corresponding M-disk $d_c(t)$ centered at c at moment $t \in \mathbb{R}$ is given as

$$d_c(t) := \bigcup_{i=0}^{n} p_{c,i}(t).$$

The circular arc that is part of the boundary of $p_{c,i}(t)$ is called the M-arc $a_{c,i}(t)$. In Figure 2.2 an example of a growing M-disk is displayed. It is apparent that a M-disk only consists of more than two M-parts if its center is at a node of the medial axis. As a consequence, it is possible to differentiate between a left and a right M-part relative to the orientation of the corresponding edge of the medial axis.

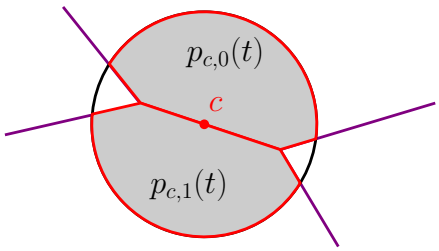


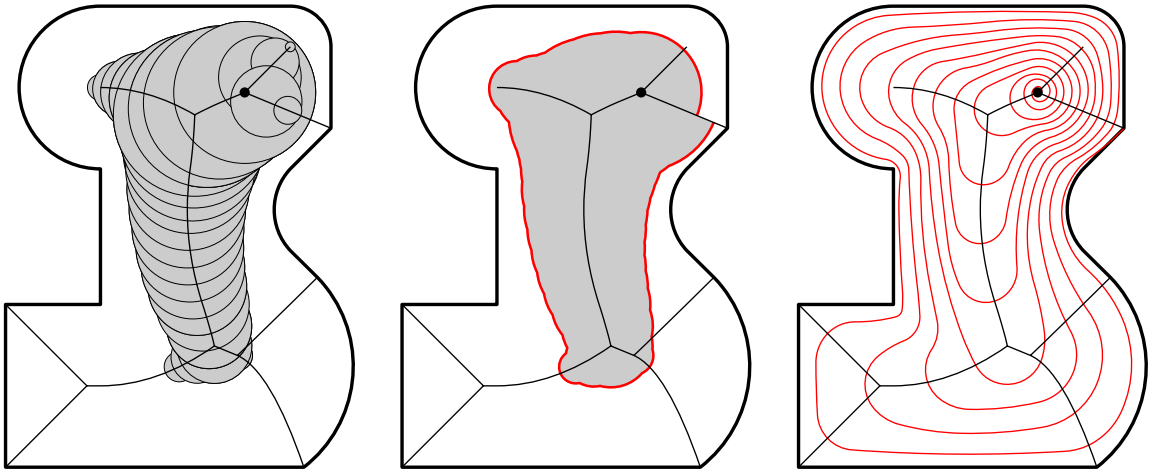
Figure 2.2.: The M-disk centered at c consists of the M-parts $p_{c,0}(t)$ as well as $p_{c,1}(t)$.

2.3. Impulse propagation

As stated above, the medial axis $\mathcal{MA}(P)$ inside a pocket P without islands forms a topological tree $T_r(P)$. Now, let r be the root of $T_r(P)$. Think of an impulse starting at r that spreads through $T_r(P)$, splitting at each node. An M-disk starts to grow when it is reached by the impulse. This process is called impulse propagation. Its speed depends on the length of the edge that is currently traversed. As soon as the impulse arrives at a leaf of $T_r(P)$ it stops.

2.4. The envelope

Another central concept is the so-called envelope. Intuitively, the envelope describes the border of the area of all non-null M-disks at a specific moment (see Figure 2.3).



(a) (b) (c)

Figure 2.3.: In (a) the envelope evaluated at a specific moment is displayed. Furthermore, a lap generated by interpolating between neighboring wavefronts (b) as well as the final tool path (c) are shown ([Held and Spielberger, 2009]).

Definition 2.1 (Envelope)**.** Let $d_c(t)$ be a growing M-disk centered at c with $t \in [0,1]$ with start time t_c. The region inside the envelope $E_P : [0,1] \to A$ of a pocket P is given by

$$E_P(t) := \bigcup_{c \in \mathcal{MA}(P) \wedge t \leq t_c} d_c(t),$$

where $A \subset \mathbb{R}^2$.

The value t has to be out of the closed interval $[0,1]$. It is easy to see that E_P at $t = 0$ degenerates into a single point, namely, the root of the medial axis tree, and at $t = 1$ it is equal to the region defined by the input pocket P. In practice, it is impossible to handle an infinite number of M-disks, this is why a discrete version of the envelope has to be established.

Definition 2.2 (Discrete envelope)**.** Let $d_c(t)$ be a growing M-disk centered at c with $t \in [0,1]$ with start time t_c. The region inside the discrete envelope $\widetilde{E}_P : [0,1] \to A$ of a pocket P is given by

$$\widetilde{E}_P(t) := \bigcup_{c \in C \wedge t \leq t_c} d_c(t),$$

where $A \subset \mathbb{R}^2$ and C is a finite set of uniformly distributed points on $\mathcal{MA}(P)$. The actual discrete envelope $\partial\widetilde{E}_P(t)$ is the border of $\widetilde{E}_P(t)$.

Trivially, $\partial\widetilde{E}_P(t)$ can be represented by a closed curve. It is a continuous sequence of circular arcs. An alternative way to interpret the envelope is as an interpolation between the root of $\mathcal{MA}(P)$ and ∂P.

2.5. Generating a spiral tool path

Through evaluating the discrete envelope $\partial\widetilde{E}_P$ at different points in time, it is feasible to derive a series of successive wavefronts W. How such wavefronts look in practice is shown in Figure 2.4. In order to generate a coherent cutter trajectory, it is necessary to interpolate

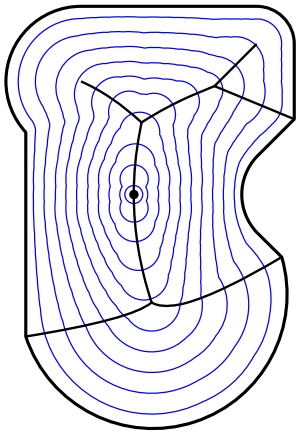


Figure 2.4.: A sequence of consecutive wavefronts. The root is depicted by a bold circle ([Held and Spielberger, 2009]).

between neighboring elements of W. First, the tool path is divided into several laps. Each lap consists of a set of consecutive M-parts, which are generated by traversing the medial axis tree once. The concrete interpolation is done by adjusting the extent of each M-part of a specific lap individually. Figure 2.3 shows the resulting spiral tool path. It is important to note that the difference in time between two successive laps stays exactly the same.

2.6. Smoothing the curve

Although a continuous tool path has already been obtained there still exist $\mathcal{C}^1$-discontinuities. In general, they appear at intersections between M-parts, and need to be dealt with if the tool path should be suitable for HSM. In this approach this is done through smoothing line segments (SLSs) (see Figure 2.5) and smoothing circular arcs (SCAs).

Definition 2.3 (Smoothing line segment)**.** Let b_i and b_{i+1} the contact points of the circular arcs a_i and a_{i+1}. A sharp corner between a pair of arcs (a_i, a_{i+1}) is smoothed by a SLS $\overline{a_{i-1}a_i} \cap \overline{a_i a_{i+1}}$ if and only if

- the portion $\partial\widetilde{E}_P$ between b_i and b_{i+1} is a sequence of counter-clockwise (CCW) arcs and straight-line segments,
- $\overline{a_{i-1}a_i} \cap \overline{a_i a_{i+1}} = \emptyset$ and $\overline{a_i a_{i+1}} \cap \overline{a_{i+1}a_{i+2}} = \emptyset$.

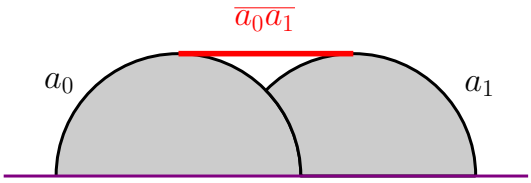


Figure 2.5.: A SLS $\overline{a_0 a_1}$ between the circular arcs a_0 and a_1.

Even after inserting SLSs at all valid locations there is the possibility that sharp corners remain. To deal with those cases SCAs are inserted, i.e., a circular arcs that start at the contact point of the first M-part and end at the contact point of the second arc. After all SLSs and SCAs are introduced, there are still potential $\mathcal{C}^1$-discontinuities, they occur if a SLS and a SCA converge. In this case smoothing repair arcs (REPs) have to be used.

This smoothing process does not influence the overall distance between two successive laps. Thus, the tool path remains free of self-intersections, because the inserted smoothing elements always stay inside the corresponding envelope. Additionally, the resulting curve is $\mathcal{G}^1$-continuous.

2.7. Discussion

The tool-path generation method discussed above yields a cutter trajectory which

- is $\mathcal{G}^1$-continuous,
- is not self-intersecting,
- has a cutting width greater than zero and less than the user-specified, maximum cutting width in all possible locations,
- and stops once the boundary is reached.

By the means of B-spline approximation, it is possible to boost the resulting curve to $\mathcal{C}^2$-continuity.

However, this strategy also has its drawbacks. In theory, the concept of the previously described M-disks is clear, but in practice they prove to be quite cumbersome. Various problems arise when it comes to implementing them.

- It is folklore that the generalized Voronoi diagram consists of straight-line segments as well as conic sections. As mentioned before, the individual M-disks have to be intersected with the medial axis in order to generate M-parts. This means that we have to intersect circles with conic sections. From a theoretical point of view this is trivial, but cannot be solved analytically in a discrete environment. So one has to rely on numerical methods which are costly. Apart from this, this procedure has to be done various times during the growth of a M-disk.

- The tool path has to be smoothed by inserting SLSs and SCAs in order to achieve $\mathcal{C}^1$-continuity, but this process is not straightforward in the general case. It involves tangential intersections between circular arcs and straight-line segments or other circular arc, which are notoriously hard to handle with only finite precision available. Additionally, it is difficult and time consuming to detect possible intersection of SLSs and SCAs in order to be able to insert REPs.

3. Obtaining a Raw Tool Path

Obtaining a practical implementation based on a theoretical description can prove to be challenging. It is folklore that modern computer systems only offer limited precision. Inaccuracies can lead to serious issues especially in geometry-based algorithms. Analytical methods for solving various problems are often too costly and sometimes simply not available. Thresholds are desperately needed to detect intersections. Even classical mathematical problems that may seem trivial bear caveats. The avoidance of numerical instabilities is from utmost importance in most real world applications. Unsurprisingly, this case forms no exception. In this chapter the, previously discussed, tool-path generation method is reviewed in more detail. We will suggest changes to the basic scheme and fill gaps in order to make a stable realization possible.

3.1. Overview

First of all, we want to summarize the individual steps of the tool-path generation strategy briefly.

Given: A simple, closed curvilinear chain ∂I which is oriented counter-clockwise, and the maximum step-over δ.

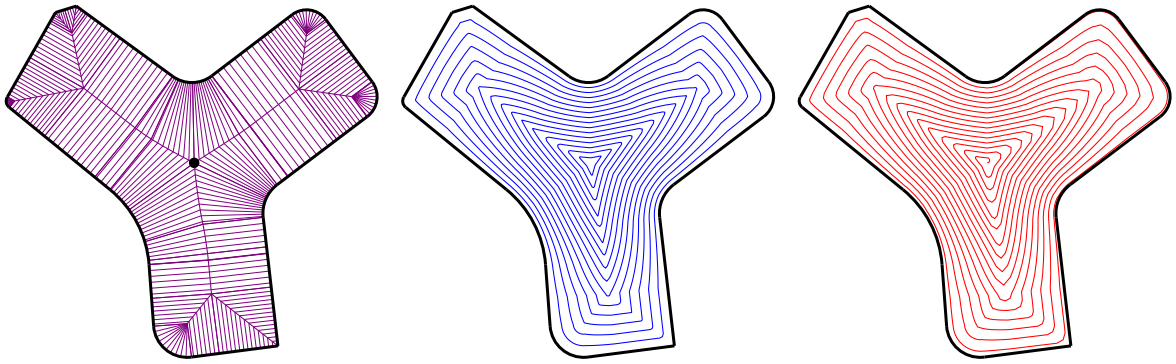
Determine: A tool path $\mathcal{TP}(P, \delta)$ that is not self-intersecting, and respects δ. Furthermore, it is required that the cutter stays inside the interior I of ∂P, and covers I entirely if δ is chosen appropriately, that is, smaller as the cutter diameter.

1. Initially, the generalized Voronoi diagram $\mathcal{VD}(\partial I)$ of ∂I is calculated.
2. Afterwards, the left offset ∂P is derived from ∂I. We refer to P as pocket.
3. Inside P the medial axis $\mathcal{MA}(P)$ is computed. It is assumed that P does not include islands. Thus, $\mathcal{MA}(P)$ can be interpreted as a tree $T_r(P)$ which is rooted at r, where r is an arbitrary point on $\mathcal{MA}(P)$.
4. A series of uniformly distributed sample points are placed on $T_r(P)$ that are utilized to construct the so-called clearance lines.
5. Those clearance lines are used to linearize $T_r(P)$ as well as ∂P.

6. The discrete medial axis tree $T'_r(P)$ is constructed by merging $T_r(P)$ with the corresponding clearance lines. In Figure 3.1a an example of a discrete medial axis tree is shown.

7. A sequence of well-chosen, consecutive wavefronts is computed (see Figure 3.1b). Each wavefront is represented by a closed polygonal chain. The corners of the individual wavefronts are situated on the discrete medial axis tree $T'_r(P)$.

8. The final tool path $\mathcal{TP}(P, \delta)$ is constructed based on the notion of laps. Each lap corresponds to a cutter pass. The first as well as the last lap are produced by interpolating between consecutive wavefronts. All other laps are generated by interpolating between those two initial laps. The result of this tool-path generation scheme is depicted in Figure 3.1c.



(a) (b) (c)

Figure 3.1.: A set of wavefronts (b) is derived from the discrete medial axis tree (a) in order to generate the final raw tool path (c).

3.2. General conditions

Firstly, we need to formalize the term contour, and define how a valid input sequence looks like.

Definition 3.1 (Polygonal chain)**.** A polygonal chain is a sequence of points $Z := (p_0, p_1, ..., p_k)$, where neighboring elements of Z are connected through straight-line segments. We call Z closed if $p_0 = p_k$.

Definition 3.2 (Curvilinear chain)**.** If $Z := (p_0, p_1, ..., p_k)$ is a sequence of points, where consecutive elements are connected by straight-line segments or circular arcs, then Z is called a curvilinear chain. Furthermore, Z is called closed if $p_0 = p_k$.

Subsequently, we will refer to a simple, closed curvilinear chain as a contour. Clearly, the term pocket is of fundamental importance in milling applications. A naive description of a pocket has already been given. Now it is time to discuss this topic in more detail and present

a formal definition. Therefore, let C_B and C_I be two contours. If C_I lies completely in the interior of C_B, then C_I is called an island relative to the outer contour C_O. An island is not allowed to include islands itself. Furthermore, we assume that the outer contour is oriented counter-clockwise and islands are oriented clockwise. A set consisting of one outer contour that includes k islands is called input set ∂I, with $k \geq 0$. By computing the left offset of all elements of ∂I in distance equal to the cutter radius, the respective pocket region P is generated which is bound by a set ∂P that consists of points, straight-line segments, and circular arcs (see Figure 3.2). These offset elements can be derived easily from the Voronoi diagram $\mathcal{VD}(\partial I)$ of ∂I which takes $O(n \log n)$ time to calculate, where n is the number of input sites. Note that this procedure does not necessarily preserve the topology of I. Throughout the course of this thesis it is assumed that we are dealing with pockets without islands (if not explicitly stated otherwise).

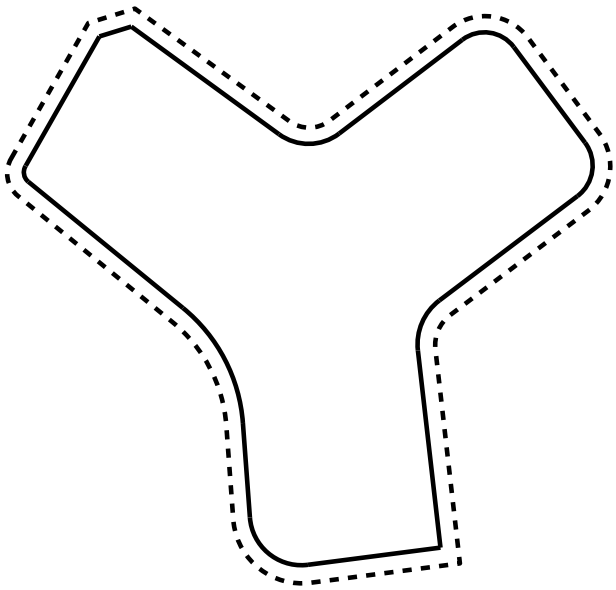


Figure 3.2.: A pocket boundary (illustrated by the solid curve) derived from the border of an input set (represented by the dashed curve).

The details on how to implement a numerically stable algorithm that gives us a basic version of our cutter trajectory will be explored in the sequel. It will still compromise $\mathcal{C}^1$-discontinuities, that is, further smoothing is needed in order to generate a satisfying result. How this optimization process looks like will be discussed subsequently. Thus, we only talk about a raw tool path for now which is represented by a polygonal chain, and has the following properties.

- The distance between successive cutter passes does not exceed an initially specified, maximum step-over δ.
- It shows no self-intersections.
- The tool path starts at the root of the medial axis tree, and ends at the boundary

contour of the pocket.

3.3. The medial axis

Theory tells us that by trimming the edges of $\mathcal{VD}(\partial I)$ and adjusting the clearance, it is possible to transform $\mathcal{VD}(\partial I)$ into $\mathcal{VD}(\partial P)$. Furthermore, the medial axis $\mathcal{MA}(P)$ of a pocket P is a subset of the generalized Voronoi diagram $\mathcal{VD}(\partial P)$. As a consequence, the medial axis can be computed efficiently if the corresponding Voronoi diagram is known. To simplify things, $\mathcal{MA}(P)$ is approximated by straight-line segments. The respective sampling strategy will be discussed later in detail. Nonetheless, it is necessary to outline some basic concepts related to this linearized version of the medial axis right away.

Every edge e of $\mathcal{MA}(P)$ is considered separately. Let $Q := (q_1, q_2, ..., q_k)$ be the sequence of those sample points in the order in which they appear when e is traversed from start point q_0 to end point q_k. The discrete edge e' is defined as

$$e' := \bigcup_{i=0}^{k-1} \overline{q_i q_{i+1}}.$$

The length $\text{len}(e')$ is specified by

$$\text{len}(e') := \sum_{i=0}^{k-1} d(q_i, q_{i+1}).$$

In general, a discrete edge will be shorter than its continuous counterpart. Trivially, the linearized medial axis $\mathcal{MA}'(P)$ is given by the union of all its discrete edges. The individual sample points are called centers of clearance lines (CCLs) on $\mathcal{MA}'(P)$.

3.4. Clearance lines

Due to the drawbacks of the, previously described, M-disks, we propose an alternative approach that is more amendable to implement and yields a more stable result. Instead of these M-disks, a set of clearance lines will be used to build a spiral tool path.

3.4.1. Fundamental concept

A clearance line is, as the name suggests, a straight-line segment. Its definition is based on the notion of a clearance disk situated on the medial axis of a pocket.

Definition 3.3 (Clearance line)**.** Let $\mathcal{MA}'(P)$ be the linearized medial axis of a pocket P. Furthermore, let d_p be a clearance disk centered at p, where p is a CCL on $\mathcal{MA}'(P)$. A clearance line l starting at p is a straight-line segment $\overline{pq}$, where q is an intersection point of d_p and ∂P.

Trivially, all clearance lines starting at the medial axis $\mathcal{MA}'(P)$ of a pocket P are mutually disjoint, except possibly at their common start or end point, respectively. Moreover, the number of clearance lines that share their start point p with the center of a specific clearance disk depends on the position of p on the medial axis.

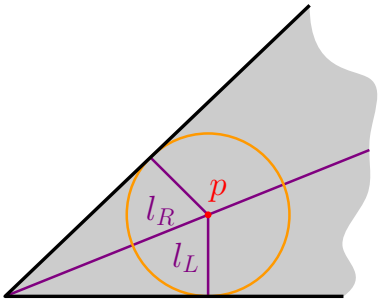


Figure 3.3.: Two clearance lines starting at p on an edge of $\mathcal{MA}'(P)$ defined by two straight-line segments (depicted in purple). The respective clearance disk is highlighted in orange.

An example how clearance lines look in practice can be found in Figure 3.3. Trivially, it is possibly to find at least two clearance lines starting at p such that one is positioned to the left and the other one to the right of their defining edge, at every point p on $\mathcal{MA}'(P)$.

3.4.2. Practical implementation

Obviously, each defining site of a point $p \in \mathcal{MA}'(P)$ corresponds to at least one clearance line starting at p. Thus, it is sufficient to only consider the defining sites in order to find all clearance lines starting at p. Furthermore, we know that a pocket consists of points, straight-line segments, and circular arcs. Based on this knowledge, it is possible to define a scheme used to construct all clearance lines at a specific CCL (see also Figure 3.4).

Given: A CCL $p \in \mathcal{MA}'(P)$.

Determine: All clearance lines starting at p.

A defining site s of p either corresponds to exactly one or infinitely many clearance lines. Firstly, let us consider the case that exactly one clearance line $l := \overline{pq}$ connects p with s, where q is a point on s. Based on the type of s it is possible to distinguish three cases.

- If s is a straight-line segment, then the respective end point of a clearance line starting at p is given by the orthogonal projection of p onto s.
- In the case that s is a circular arc and p is not equal to the center c of s, q is given as the intersection point of s and the line defined by p and c.
- Otherwise, s is a point. Trivially, the end point of l equals s.

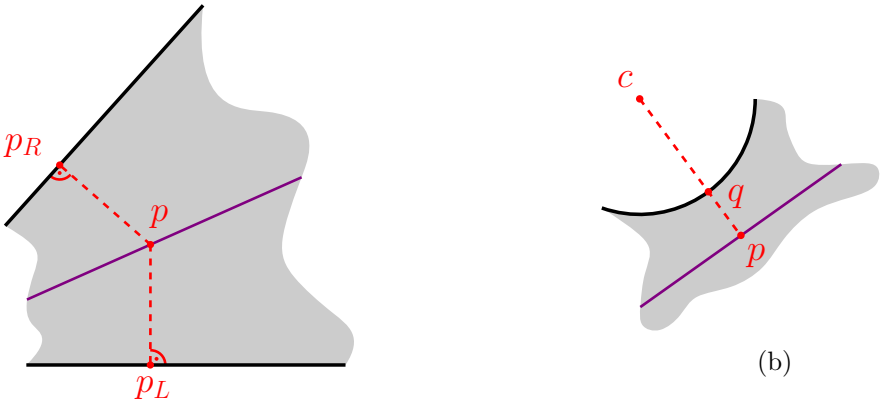


Figure 3.4.: Figure (a) shows the orthogonal projection of a point p on the medial axis onto the boundary of the pocket. In (b) a clearance line defined by a circular arc is shown.

The latter case remains to be treated. Imagine a CCL p on $\mathcal{MA}'(P)$ which is at the same position as the center of one of its defining arcs a, that is, the clearance disk d_p centered at p overlaps partially with a. We call a a concentric arc relative to d_p.

The following steps have to be taken to produce a satisfying cutter trajectory in such areas. A finite series S of uniformly distributed points on a is chosen. Through connecting p with each point in S, the sought clearance lines are produced. In Figure 3.5 this process is shown. Clearly, the higher the granularity of the distribution is chosen, the better the resulting approximation becomes.

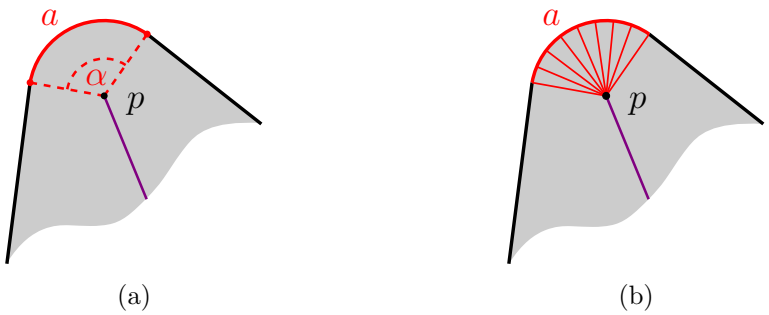


Figure 3.5.: In (a) a CCL p defined by the arc a as well as two straight-line segments is depicted. Figure (b) illustrates the insertion of multiple clearance lines starting at p.

3.5. Distributing the clearance lines

It remains to discussed how the linearized version of the medial axis $\mathcal{MA}'(P)$ is produced. Thus, every edge e of the continuous medial axis $\mathcal{MA}(P)$ is considered individually. A sequence of uniformly distributed sample points is placed at the interior of e in distance λ to

each other, with

$$\lambda := \frac{\rho\left(\mathcal{MA}(P)\right)}{\mu},$$

where $\mu \in \mathbb{N}$ is a user-specified value, the so-called clearance line granularity, and $\rho(\mathcal{MA}(P))$ is the Euclidean diameter of the medial axis $\mathcal{MA}(P)$. Clearly, a higher μ yields more sample points overall. It has to be ensured that the end points of successive clearance lines are not exceedingly far away from each other. A simple heuristic yields a satisfying result. Let l_0 and l_1 be two successive clearance lines starting at p_0 and p_1, respectively. Furthermore, let q_0 and q_1 the end points of l_0 and l_1. If $d(p_0, p_1) < 2\, d(q_0, q_1)$, then additional CCLs are introduced between p_0 and p_1. In Figure 3.6 the result of this distribution strategy is shown.

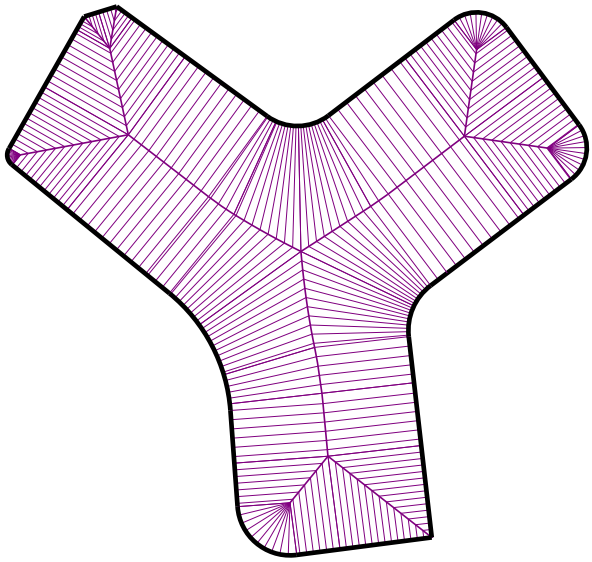
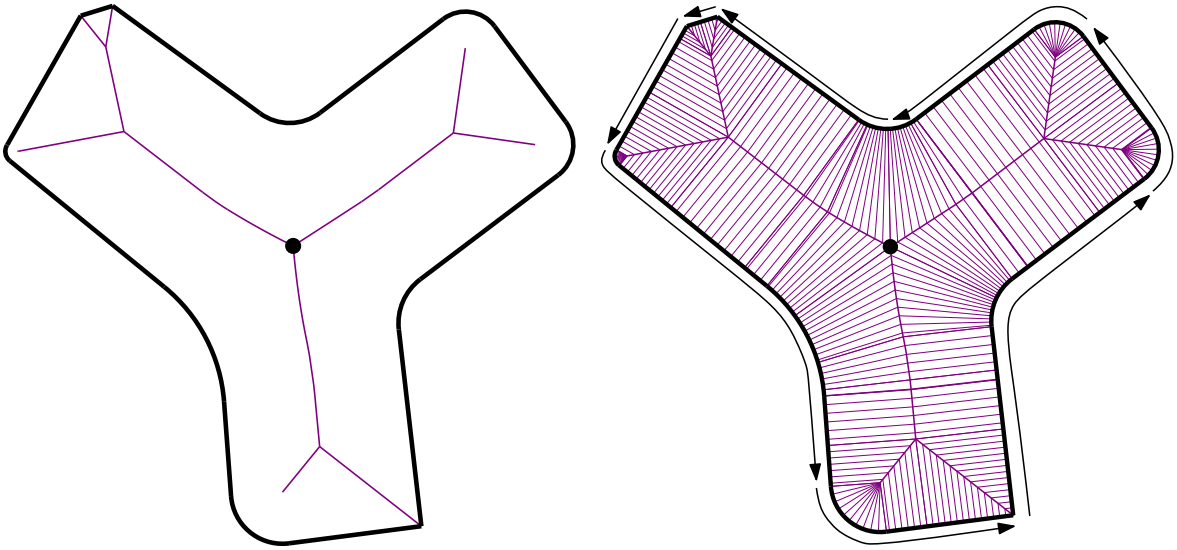


Figure 3.6.: Clearance lines as well as the medial axis inside a simple pocket (highlighted in purple).

3.6. Constructing the medial axis tree

Obviously, $\mathcal{MA}'(P)$ inside a pocket P without islands forms a tree $T_r(P)$, where r is an arbitrary point on $\mathcal{MA}'(P)$ (see Figure 3.7a). Trivially, the medial axis tree $T_r(P)$ inside P is not uniquely defined, i.e., there exist an infinite number of possible trees, because we are allowed to choose the root r freely on $\mathcal{MA}'(P)$. This opens space for optimization. For this reason, a root-selection strategy will be introduced in a later chapter.

Clearly, the distance $\text{dist}(p, q)$ between two points p and q along $T_r(P)$ is uniquely defined, with $p, q \in T_r(P)$. Furthermore, it is possible to determine the height of a node of $T_r(P)$. We will be using the so-called Euclidean height (see Definition 3.4) which was introduced by Held and Spielberger [2009].



(a) (b)

Figure 3.7.: The medial axis tree $T_r(P)$ inside a pocket P. The root r is depicted by the bold dot (a). In (b) the discrete medial axis tree $T'_r(P)$ as well as the respective traversal is illustrated.

Definition 3.4 (Euclidean height)**.** Let $T_r(P)$ be the medial axis tree rooted at r of the pocket P and let v be a node of $T_r(P)$. The height of v is defined by

$$h_{T_r(P)}(v) := \begin{cases} \max\{\text{dist}(v,w) + h_{T_r(P)}(w) : w \text{ is a child of } v\} & \text{if } v \text{ is not a leaf,} \\ \text{clr}(v) & \text{else,} \end{cases}$$

where $\text{clr}(v)$ is the clearance at v. The height $h_{T_r(P)}$ of $T_r(P)$ is given by $h_{T_r(P)}(r)$.

Lemma 3.5. It is possible to determine the height of $T_r(P)$ in $O(n)$ time, where n is the number of nodes of $T_r(P)$.

Proof. See Algorithm 3.1. □

It is easy to extend the definition of the Euclidean height to points which are situated in the interior of an edge. Let p be a point in the interior of the edge e which ends at the vertex w. The Euclidean height of p is given by $h_{T_r(P)}(w) + \text{dist}(p,w)$.

3.7. An alternative perspective

The set of clearance lines L on $T_r(P)$ together with $T_r(P)$ itself can be regarded as the Voronoi diagram inside a modified pocket. Based on this observation an alternative to the conventional medial axis tree can be constructed. Thus, the structure $T'_r(P)$ is defined as

$$T'_r(P) := T_r(P) \cup L.$$

Algorithm 3.1 Find the height of a node v inside the medial axis tree T.

```
1: function CalculateHeight(T, v, h)
2:     if v is a leaf of T then
3:         return clr(v)
4:     else
5:         for all c ∈ children(v) do
6:             k ←CalculateHeight(T, c, h)
7:             if k > h then
8:                 h = k
9:             end if
10:        end for
11:        return h + dist(v, c)
12:    end if
13: end function
```

Every start point of a clearance line in L is interpreted as a vertex in $T'_r(P)$. This extension preserves a key property of $T_r(P)$. Namely, $T'_r(P)$ still forms a tree. Recall that different clearance lines are disjoint except at their common start points. Thus, we will refer to $T'_r(P)$ is the discrete medial axis tree derived from $T_r(P)$. The Euclidean height of points on $T'_r(P)$ can be defined identically to the conventional case (see Definition 3.4).

3.8. Modifying the input pocket

In order to guarantee that the final cutter pass equals the boundary ∂P, the input pocket P has to be modified slightly. More precisely, ∂P will be linearized. That is, each site s that is part of ∂P is examined. If s forms a circular arc, then it is split at the end points of the respective clearance lines, and every portion of s that lies between two neighboring end points is replaced by a straight-line segment.

3.9. Impulse propagation

It is essential to determine the order in which the nodes of $T'_r(P)$ are stimulated, i.e., we have to calculate the point in time $t \in [0, 1]$ when they start growing. As mentioned previously, the impulse starts at the predefined root r of the medial axis tree $T'_r(P)$ and discharges at the leaves of $T'_r(P)$. Its speed changes depending on the edge it currently traverses. Three factors influence the velocity of the impulse at an edge e.

- The moment when the start point of e is stimulated, that is, when the impulse first reaches e. The earlier this happens, the slower the impulse propagates on the corresponding edge.
- The velocity at e grows directly proportional relative to the height of its end point in $T'_r(P)$.

- At a longer edge the impulse propagates faster.

We will define the impulse velocity similar to Held and Spielberger [2009].

Definition 3.6 (Impulse velocity)**.** Let e be an edge between the nodes p and q. The velocity v_e of the impulse at e is described by

$$v_e := \frac{h_{T'_r(P)}(q) + \text{len}(e)}{1 - t_p},$$

where $\text{len}(e)$ is the length of e, and t_p is the start time at p.

Based on the definitions of the height of a node of $T'_r(P)$ as well as velocity at an edge, it is possible to determine the moment when the impulse reaches a specific vertex. Imagine two neighboring nodes p and q in $T'_r(P)$ that are linked by an edge e. At r we set the start time $t_r = 0$. For all other vertices $q \in T'_r(P)$ the start time is given as

$$t_q = t_p + \frac{\text{len}(e)}{v_e},$$

where p is the parent node of q inside $T'_r(P)$.

Obviously, the impulse starts at the root r, and arrives at the individual nodes in the order they appear when traversing $T'_r(P)$ from r to the corresponding leaf node. Due to the fact that the velocity of the impulse stays the same over the whole edge e, the start time t_s of a point s on e is simply defined by

$$t_s = t_p + \frac{\text{dist}(p, s)}{v_e}.$$

The edge e is called active at time t if $t_p \leq t \leq t_q$.

3.10. Generating wavefronts

As the impulse flows through the (discrete) medial axis tree $T'_r(P)$, it covers an increasing portion of $T'_r(P)$. The last point the impulse reaches on a particular branch b of $T'_r(P)$ at time t will be called a corner at time t on b. Clearly, there exist as most as many corners as there are branches in $T'_r(P)$. Furthermore, it is possible and rather likely that two or more corners of distinct branches coincide at t.

By computing all corners at a specific moment, and arranging them appropriately, it is feasible to construct a closed polygonal chain, a so-called wavefront.

Definition 3.7 (Wavefront)**.** Let $T'_r(P)$ be a medial axis tree rooted at r inside a pocket P. Additionally, let $V := (v_0, v_1, ..., v_k)$, with $k \in \mathbb{N}_0$, be the sequence of all corners which are defined by the impulse at time $t \in [0, 1]$ in the order in which they appear when $T'_r(P)$ is traversed in depth-first manner. The wavefront $w(t)$ at t is a closed polygonal chain defined by V.

In Figure 3.7b the respective traversal is depicted. In order to guarantee that the initially specified, maximum step-over δ is respected, those wavefronts have to be chosen carefully. Therefore, a uniform decomposition of time $T := (t_0, t_1, ..., t_m)$, with $m \in \mathbb{N}_0$, and $0 = t_0 < t_1 < ... < t_m = 1$, will be calculated. The difference between two consecutive elements of T is denoted as Δt. Our goal is to choose Δt, such that

$$H(w(t_i), w(t_{i+1})) < \delta,$$

for $i \in \{0, 1, ..., m-1\}$.

Due to Definition 3.6, the velocity at every edge of $T'_r(P)$ is bound by $h_{T'_r(P)}(r)$. This implies that the impulse travels at most $\Delta s\, h_{T'_r(P)}(r)$ in time Δs along the medial axis. That is, we are able to establish an upper bound on the symmetric Hausdorff distance between $w(s_0)$ and $w(s_0 + \Delta s)$, with $s_0 \in [0, 1 - \Delta s]$, such that

$$H(w(s_0), w(s_0 + \Delta s)) < \Delta s\, h_{T'_r(P)}(r).$$

Hence, if we set

$$\Delta s := \frac{\delta}{h_{T'_r(P)}(r)},$$

then the impulse travels at most δ along $T'_r(P)$ during Δs. Afterwards, the interval $[0, 1]$ is subdivided into m equal parts, with

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

Finally, the difference Δt between $t_i, t_{i+1} \in T$, with $i \in \{0, 1, ..., m-1\}$, is defined as

$$\Delta t := \frac{1}{m}.$$

Due to construction, the Hausdorff distance between $w(t_i)$ and $w(t_{i+1})$ is at most δ.

3.11. A connected spiral tool path

Now that the fundamental concepts of the tool-path generation strategy have been discussed, we need to put the pieces together. The final tool path $\mathcal{TP}(P, \delta)$ is made up of $(m+1)$ so-called laps $R_0, R_1, ..., R_m$. Each of those laps consists of a series of corners, and forms a polygonal chain. The basic idea behind the construction scheme is to compute the first and the last lap by interpolating between well-chosen wavefronts. Subsequent laps are generated by gradually moving along $T'_r(P)$. Therefore, the maximum expansion of the impulse is limited such that the user-specified, maximum step-over is respected everywhere along $\mathcal{TP}(P, \delta)$.

Initially, R_0 is calculated by considering the wavefronts $w(t_0)$, i.e., the root r of $T'_r(P)$, and $w(t_1)$. As mentioned above, $w(t_1)$ is given by a sequence $V := (v_0, v_1, ..., v_k)$ of corners

which corresponds to a series of branches $B' := (b'_0, b'_1, ..., b'_k)$. For every element b'_j of B', with $j \in \{0, 1, ..., k\}$, a value τ_j is computed, with

$$\tau_j := \begin{cases} t_0 & \text{if } j = 0, \\ \tau_{j-1} + d_j & \text{else,} \end{cases}$$

where

$$d_j := \frac{d(v_{j-1}, v_j)}{o(w(t_1))} \Delta t,$$

and $o(w(t_1))$ is the accumulated length of all straight-line segments that are contained in $w(t_1)$. The corner c_j of b'_j at time τ_j defines the j-th corner of the lap R_0. Thus, R_0 is a polygonal chain. The final lap R_m is constructed analogously to R_0, with the difference that t_0 and t_1 are substituted by t_{m-1} and t_m, respectively. See Figure 3.8a for an illustration of this construction scheme.

At every corner p of R_0 we travel down $T'_r(P)$ until the closest corner q of R_m is reached. Let t_p and t_q the start times of p and q, respectively. If the difference between t_p and t_q is greater than $(m-1)\,\Delta t$, then q is moved closer towards p along $T'_r(P)$ until $t_q = t_p + (m-1)\,\Delta t$. Due to the construction of the individual wavefronts, it is never necessary to move q beyond the respective corner of $w(t_{m-1})$.

The remaining laps are produced recursively. The following procedure can be regarded as an interpolation between R_0 and R_m. Let R_i be a lap that has already been constructed, with $i \in \{0, 1, ..., m-2\}$.

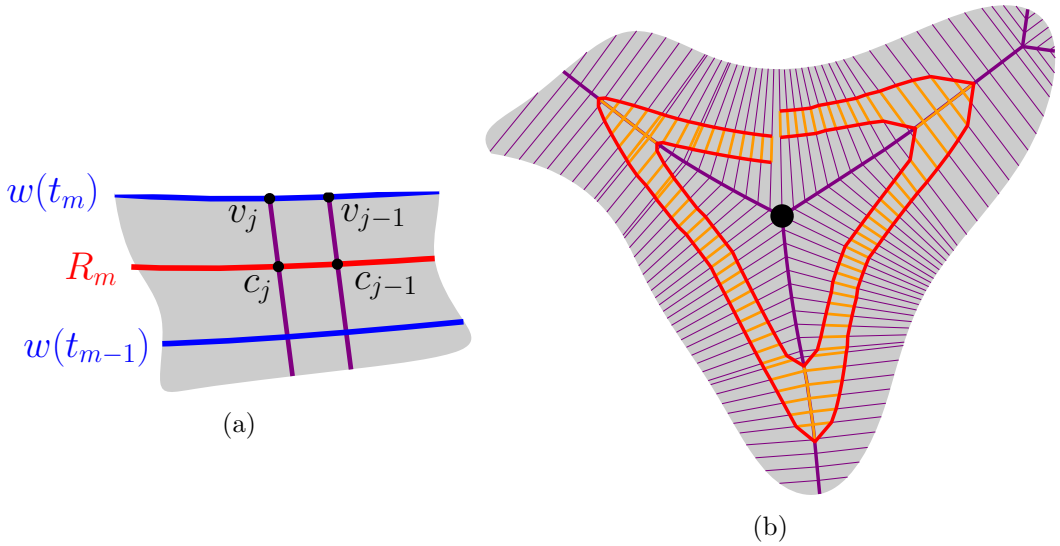


Figure 3.8.: Figure (a) shows a corner c_j (of the last lap R_m) that is computed based on the evaluation time of its predecessor c_{j-1} as well as the distance between the respective corners v_{j-1} and v_j of the subsequent wavefront $w(t_m)$. In (b) Two consecutive laps are illustrated (in red and blue). The expansion of the individual corners is highlighted in orange.

In order to generate the lap R_{i+1}, each corner p_i of R_i is examined individually. Firstly, the corner p_m of R_m is determined that is situated on the longest branch b starting at p_i. Based on this information, it is possible to move towards R_m while simultaneously controlling the step-over. Thus, the maximum expansion $\nu_i(p_i)$ at p_i is defined as

$$\nu_i(p_i) := \frac{h_{T'_r(P)}(p_i) - h_{T'_r(P)}(p_m)}{m - i}.$$

No matter where on $T'_r(P)$ the growth starts, the individual maximum expansions are always bound by δ. Let q be the point which is $\nu_i(p_i)$ away from p_i along b. Additionally, it is assumed that t_q is the start time of q. Afterwards, we travel down $T'_r(P)$ starting at p_i until the last active points at time t_q (on the respective branches) are reached. Those points define corners in R_{i+1}, and are at most δ away from p along $T'_r(P)$ due to the choice of the respective maximum expansions (see Figure 3.9).

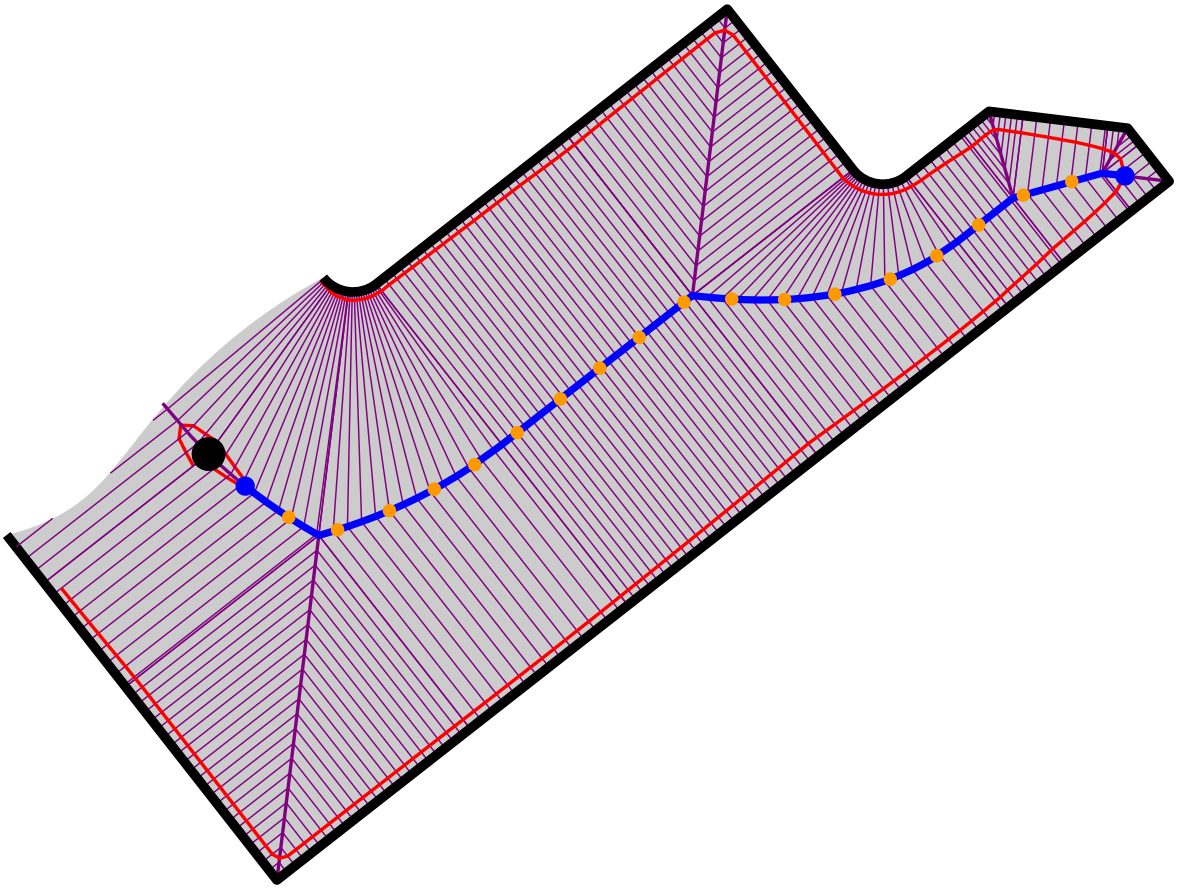


Figure 3.9.: The expansion of a corner along a branch (illustrated in blue) of the medial axis tree is shown. The first as well as the last lap are highlighted in red. Each one of the orange points belongs to a distinct lap of the final tool path.

This process yields a set of corners C. The elements of C are connected in the order in which they occur when $T'_r(P)$ is traversed in depth-first manner (see Figure 3.7b). Therefore, the polygonal chain R_{i+1} is generated. In Figure 3.8b this construction scheme is illustrated.

After all laps have been calculated, their start and end points, respectively, are connected. Thus, a polygonal chain is generated which is called a raw tool path $\mathcal{TP}(P, \delta)$ inside P (see Figure 3.10). Due to construction, $\mathcal{TP}(P, \delta)$ respects the maximum step-over δ. Trivially, $\mathcal{TP}(P, \delta)$ starts at r and ends on ∂P. This is guaranteed based on the construction of the first as well as the last lap. Furthermore, $\mathcal{TP}(P, \delta)$ is not self-intersecting, because we are gradually moving outwards starting at r until we arrive at ∂P during the tool-path generation.

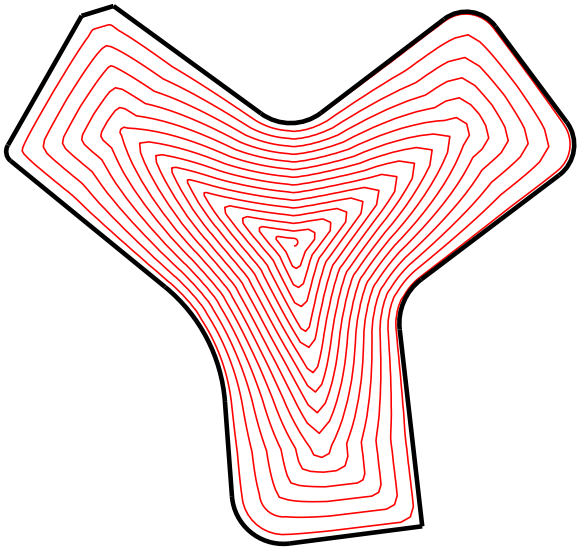


Figure 3.10.: A raw tool path (highlighted in red) inside a simple pocket is shown.

4. Dynamic Boundary

The raw tool path has a number of sharp turns, i.e., rapid direction changes. These artifacts occur based on three main reasons.

- The cutter trajectory moves through (or very close to) the center of a clearance disk.
- As the impulse spreads, it splits at the nodes of the medial axis which results in a change of the interpolation behavior at such locations.
- Discontinuities of the boundary reoccur in the resulting spiral.

In this chapter we propose a new method that improves the kinematic features of the cutter trajectory, the so-called dynamic boundary. As the name suggests, the border contour is not static, but changes during the tool-path generation. Intuitively, this approach can be described as a interpolation between the clearance disk centered at the root of the medial axis and the initial pocket.

Note that this procedure yields good results in practice, but does not guarantee that the initially specified, maximum step-over δ is respected everywhere along the modified tool path. For the sake of simplicity, we consider the medial axis tree and the clearance lines separately (instead of the discrete medial axis tree) throughout the course of this chapter.

4.1. Growing boundary

In all common $2^1/_2$D milling applications the input is specified by a fixed set of contours. They are assumed to be immutable over the course of the tool-path creation. In order to improve the raw cutter trajectory, we break with this convention, and construct our spiral tool path based on a progressively changing pocket. Intuitively, the region inside this dynamic boundary

- grows during interval $I := [0, 1]$,
- is the union of clearance disks centered at the medial axis of the initial pocket region,
- equals the clearance disk centered at the root of the medial axis tree at $t = 0$ and corresponds to the input pocket at time $t = 1$,
- and is strictly monotonously growing relative to $t \in [0, 1]$.

4.2. Boundary impulse

The dynamic boundary will be defined using the notion of the so-called boundary impulse φ, which controls its growth. To avoid ambiguities, we will from now on refer to the impulse that regulates the expansion of the clearance lines as the spiral impulse ψ. Both will be specified based on the area they occupy inside the medial axis tree $T_r(P)$ at a given moment $t \in [0, 1]$. Therefore, let us define what it means for an edge of $T_r(P)$ to be active at t.

Definition 4.1 (Active edge)**.** Let e be an edge of the medial axis tree $T_r(P)$ of the pocket P rooted at r between the nodes p and q, then e is called active at time t if

$$t_p \leq t \leq t_q,$$

where t_p and t_q are the start times of p and q.

Definition 4.2 (Active branch)**.** If $T_r(P)$ is the medial axis tree of the pocket P rooted at r and let e be an active edge at time $t \in [0, 1]$, then the longest path from r to c, such that c is situated on e, is called the active branch S_e associated with e at time t.

The area the impulse ψ visited at a specific point in time t can be now be described as the union of all active branches, i.e.,

$$\psi(t) = \bigcup_{e \in E} S_e,$$

where E is the set of all active edges at t. This forms the active subtree at t. We are able to define the position of the boundary impulse ψ_e at a specific branch S_e as the endpoint c of the corresponding path which starts at the root r of the medial axis tree. More formally $\psi_e(t) = c$, with

$$\text{dist}(r, c) = \max\{\text{dist}(r, p) : p \in e\}.$$

Similarly, it is possible to represent the location of the spiral impulse on S_e at time t by a point $\varphi_e(t)$.

Definition 4.3 (Boundary impulse)**.** Let $T_r(P)$ be the medial axis tree rooted at r inside a pocket P that is free of islands. The boundary impulse $\varphi : [0, 1] \to A$, where $A \subseteq T_r(P)$ is connected, is given by

$$\varphi(t) := \psi(t + \text{pen}(t)),$$

such that $\text{pen} : [0, 1] \to [0, 1]$ is the penalty function defined as

$$\text{pen}(t) := t^2(1 - t)^2.$$

Due to the choice of the penalty function the following assumptions hold.

- $\varphi(0) = \psi(0)$ and $\varphi(1) = \psi(1)$, i.e., both impulses start at the root r of the medial axis

tree $T_r(P)$ and discharge concurrently at the leaves of $T_r(P)$.

- $\varphi(t) \subset \psi(t) \quad \forall t \in (0,1)$, that is φ always lags behind ψ.

It is now possible to specify the dynamic boundary.

Definition 4.4 (Dynamic boundary)**.** Let P be a pocket. The dynamic boundary $\mathcal{DB}(t)$ at $t \in [0,1]$ relative to P is defined as

$$\mathcal{DB}(t) := \bigcup_{c \in \varphi(t)} d_c,$$

where d_c is the clearance disk centered at c.

In other words, the dynamic boundary is the union of all active clearance disks at a specific time, i.e., all clearance disks which are affected by the impulse. See Figure 4.1 for an example.

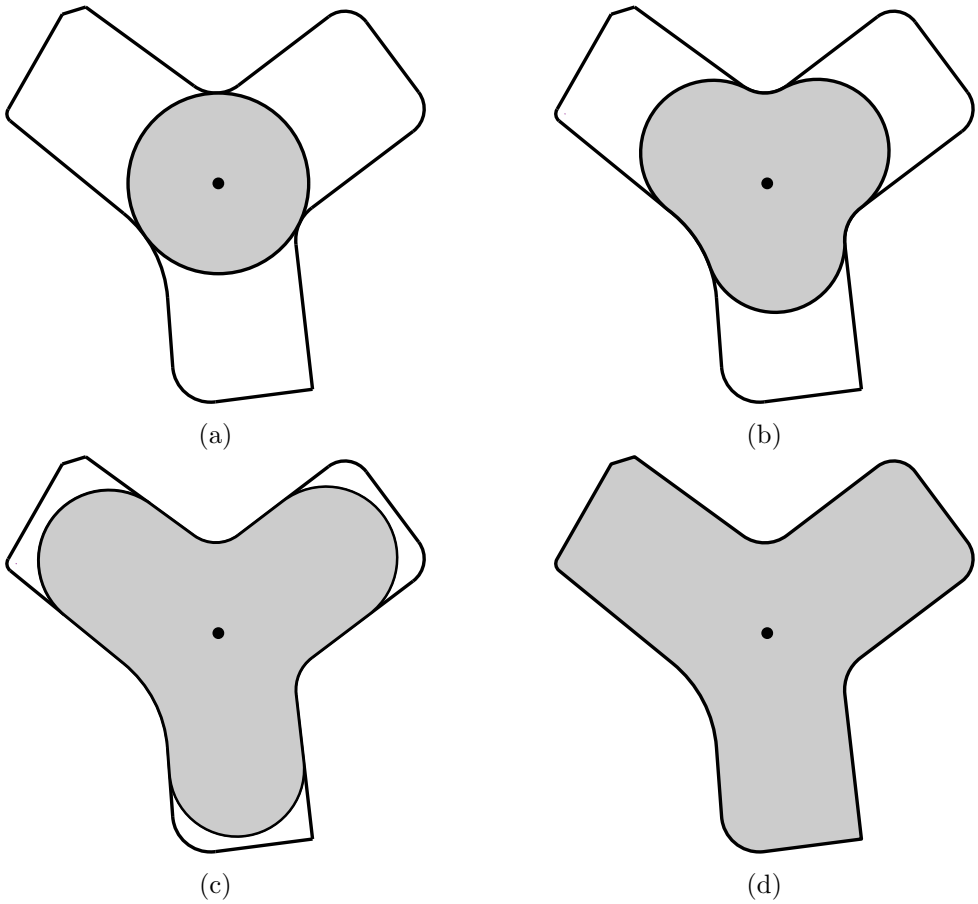


Figure 4.1.: The growing dynamic boundary expanding from time $t = 0$ until $t = 1$.

4.3. Maintaining the distance invariant

It is important to guarantee that

$$\operatorname{dist}(\psi_e(t), \varphi_e(t)) < \operatorname{clr}(\varphi_e(t)),$$

for all $t \in [0, 1]$. That is, the distance between the two impulses is not allowed to exceed the clearance of the boundary impulse in its local neighborhood. A violation can easily be recognized by backtracking starting at the last CCL which is reached by the spiral impulse until we arrive at the last CCL (on the same branch) that is reached by the boundary impulse. If such a violation of the distance invariant is detected, then the penalty has to be modified locally. Algorithm 4.1 calculates this adjusted penalty.

Algorithm 4.1 Calculate a new penalty value, such that the distance invariant is respected.

```
1: function AdjustPenalty(t)
2:     u ← ψ_e(t)
3:     v ← φ_e(t)
4:     s ← t
5:     while dist(u, v) ≥ clr(v) do
6:         s ← s/2
7:         v ← ψ_e(t + pen(s))
8:     end while
9:     return pen(s)
10: end function
```

4.4. Constructing the tool path

Finally, we are able to obtain a cutter trajectory based on the dynamic boundary. Therefore, a modified tool-path creation strategy will be applied. The following steps have to be taken.

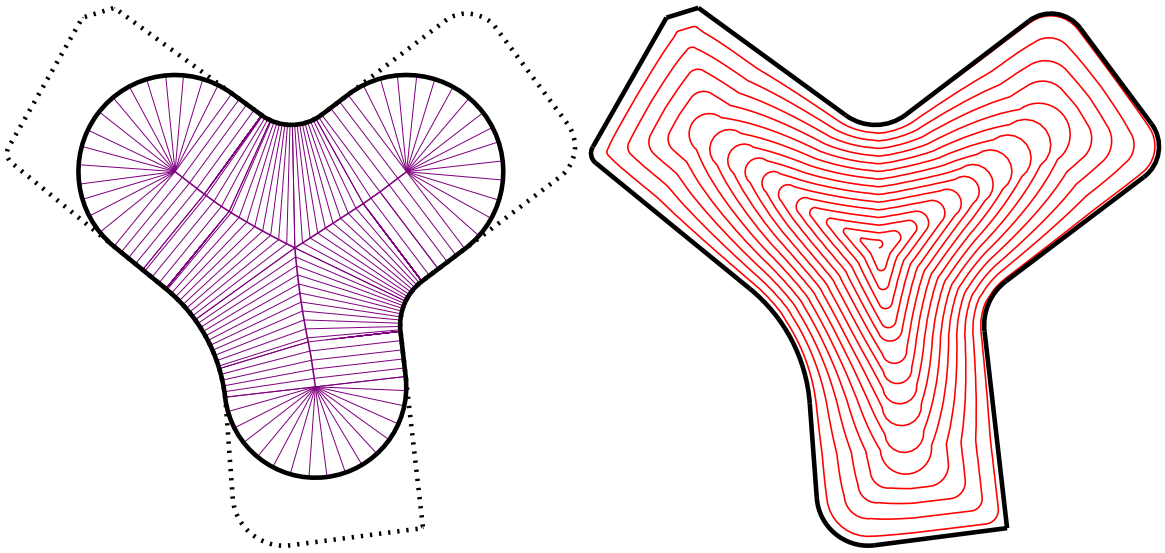
1. We calculate the medial axis tree $T_r(P)$ inside a pocket P rooted at r, as well as the respective clearance lines.

2. The i-th lap R_i, with $i \in \{0, 1, ..., m\}$, is created by calculating the corresponding corners as usual. The dynamic boundary only influences the spiral at turning points. Every turning point p of R_i marks the last point that is reached by the spiral impulse. Furthermore, it is easy to determine the starting time t_p of p. Starting at p we travel up the $T_r(P)$ until the last CCL q is found that is reached by the boundary impulse at time t_p. Obviously, the dynamic boundary defines a circular arc a that is centered at q at time t_p.

   Subsequently, multiple evenly spaced clearance lines which are all starting at q are inserted along a. All corners of R_i that are associated with regions of $T_r(P)$ that lie below q are substituted by corners which are produced by evaluating the clearance lines

starting at q at time t_p. In Figure 4.2a the result of this distribution scheme is depicted. Clearly, turning points which are removed during this procedure can be ignored in the further course.

3. After all laps $R_0, R_1, ..., R_m$ have been computed, they are merged as described before. Thus, a closed cutter trajectory $\mathcal{TP}^*(P, \delta)$ is created. An example of a cutter trajectory derived from the dynamic boundary is shown in Figure 4.2b.



(a) (b)

Figure 4.2.: In Figure (a) the state of dynamic boundary (depicted by the solid line), and the medial axis tree (together with the respective clearance lines) at a specific moment is shown. The initial pocket boundary is outlined by the dotted line. The final tool path derived from the dynamic boundary is illustrated in Figure (b).

5. Dealing with Islands

Until now we only cared about pockets without islands. As soon as they are introduced a number of new obstacles arise.

- The assumption that we deal with only one input curve no longer holds.
- Prohibited areas inside the border contour emerge, i.e., the tool is no longer allowed to move freely inside the boundary.
- We need to differentiate between two types of contours, the border and islands.
- Inside a pocket with islands the medial axis no longer forms a tree.

Hence, without special precautions the algorithm, discussed in the previous section, is not applicable for pockets with islands. Held and Spielberger [2014] proposed a solution to this problem. The basic idea is to link together the closed input curves, such that the interior of the modified pocket forms one connected component.

5.1. Constructing bridges

This procedure relies on the notion of bridges. Intuitively, a bridge is a special straight-line segment that connects exactly two input contours.

Definition 5.1 (Bridge)**.** Let C_i and C_j, with $i, j \in \mathbb{N}_0$ be two offsets of contours of a pocket P. A bridge between C_i and C_j is a pair of collinear clearance lines starting at a point $p \in \mathbb{R}^2$ such that

- The clearance at p is a local minimum.
- The bridge connects C_i and C_j.

We say that p defines a bridge in P.

That is, a bridge defined by a point p inside a pocket P can always be perceived as a straight-line segment l between the contact points of d_p with its defining segments.

Definition 5.2 (Adjacent contours)**.** Let $\mathcal{C} := \{C_1, C_2, ..., C_k\}$ be a set of islands, with $k \in \mathbb{N}$, within a pocket P. Further, let C_0 be the boundary contour of P. Two islands C_i and C_j, where $i, j \in \{1, 2, ..., k\}$, and $i \neq j$, are called adjacent if for a pair of points $p \in C_i$ and $q \in C_j$ there exists a straight-line segment $l := \overline{pq}$, such that l does not intersect $C^* \in \mathcal{C} \setminus \{C_i, C_j\}$.

Additionally, an island $C_i \in \mathcal{C}$ is called adjacent to C_0 if there exists a point p is on C_0, such that $\forall q \in C_i$ the straight-line segment $l := \overline{pq}$ does not intersect $C^* \in \mathcal{C} \setminus \{C_i\}$.

Definition 5.1 tells us that a bridge corresponds to a bottleneck inside the pocket region. The Voronoi diagram allows us to find all possible bridges, i.e., connections between adjacent contours, efficiently. Theory tells us that if the Voronoi diagram $\mathcal{VD}(\partial P)$ of a pocket P is known, then it is possible to figure out all local minima in clearance in $O(n)$ time, where n is the number of input sites. Recall that a Voronoi diagram has $O(n)$ many edges.

5.2. Choosing optimal bridges

Now that we have obtained all potential bridges, it is crucial to filter out unnecessary ones. To determine these redundant bridges we assign the quadruple (p, l, i, j) to every candidate. See Table 5.1 for further details.

Notation	Description
p	Defining point of the bridge.
l	Length of the bridge.
i, j	Indices of the contours joined by the bridge.

Table 5.1.: Description of a candidate bridge.

Based on this representation a graph is constructed. It is a fundamental tool to verify all required bridges, and resembles the adjacent input contours. An example of a collection of bridges can be found in Figure 5.1.

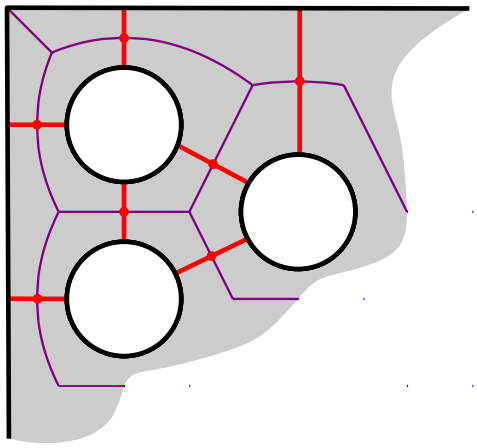


Figure 5.1.: All candidate bridges (highlighted in red) of a set of contours of a pocket.

Definition 5.3 (Contour graph)**.** Let $G(\mathcal{C}, B)$ be a graph and P be a pocket, where the edge set B consists of all possible bridges, and the vertex set $\mathcal{C}$ holds all contours P. We call $G(\mathcal{C}, B)$ the contour graph of P.

Graph theory tells us that it is always possible to transform a contour graph into a tree by breaking loops.

Definition 5.4 (Contour tree)**.** If $G(\mathcal{C}, B)$ is a contour graph of a pocket P. A graph $G'(\mathcal{C}, B')$ without loops is called a contour tree derived from $G(\mathcal{C}, B)$ inside P.

Clearly, a pocket with islands always defines multiple contour trees. The following observation helps us to select a good contour tree. Namely that a shorter bridge corresponds to a narrower bottleneck. Rapid changes of clearance affect the step-over variation negatively. Consequently, we try to close the tightest bottlenecks. This can be achieved by picking the contour tree with minimal edge length. Such a tree is known as an Euclidean minimum spanning tree (EMST).

Algorithm 5.1 Construct the minimum contour tree derived from a contour graph G of a pocket, where the vertex r of G is the starting node. Furthermore, it is assumed that Q is a priority queue, and pop(Q) yields the node with the lowest value inside Q.

```
1: function CalculateMinTree(G, r)
2:     Q ← nodes of G
3:     for all v ∈ Q do
4:         parent[v] ← 0
5:         value[v] ← ∞
6:     end for
7:     value[r] ← 0
8:     while Q is not empty do
9:         v ← pop(Q)
10:        for all u adjacent to v do
11:            if dist(u, v) < value[u] then
12:                parent[u] ← v
13:                value[u] ← dist(u, v)
14:            end if
15:        end for
16:    end while
17: end function
```

Definition 5.5 (Minimum contour tree)**.** Let $G(\mathcal{C}, B)$ is a contour graph of a pocket P and let $\mathcal{G}' := \{G'_0(\mathcal{C}, B'_0), G'_1(\mathcal{C}, B'_1), ..., G'_n(\mathcal{C}, B'_n)\}$, with $n \in \mathbb{N}_0$, be the set of all possible contours trees derived from $G(\mathcal{C}, B)$. The minimum contour tree $M(P)$ of P is a given by

$$M(P) := G'_i(\mathcal{C}, B'_i), \text{ where } G'_i(\mathcal{C}, B'_i) \in \mathcal{G}'$$

such that

$$\sum_{b \in B'_i} \text{len}(b) \leq \sum_{b \in B'_j} \text{len}(b) \text{ for all } G'_j(\mathcal{C}, B'_j) \in \mathcal{G}',$$

with $i, j \in \{0, 1, ..., n\}$.

Trivially, we are always able to find a minimum contour tree $M(P)$ inside an arbitrary

pocket P, although it may not be uniquely defined. Based on the algorithm described by Prim [1957], one may find $M(P)$ in $O(n \log n)$ time, where $n \in \mathbb{N}_0$ denotes the number of contours of P (see Algorithm 5.1).

Now that $M(P)$ is known, the only thing that remains to do is to connect the input contours through inserting the bridges. This procedure yields a modified pocket P' which meets the sought criteria.

- Trivially, the input pocket region P' contains no islands. Therefore, the medial axis forms a tree.
- In general, the area P' is not divided into multiple connected components and can be milled out without retraction moves.

An example of a tool path inside a modified pocket is shown in Figure 5.2.

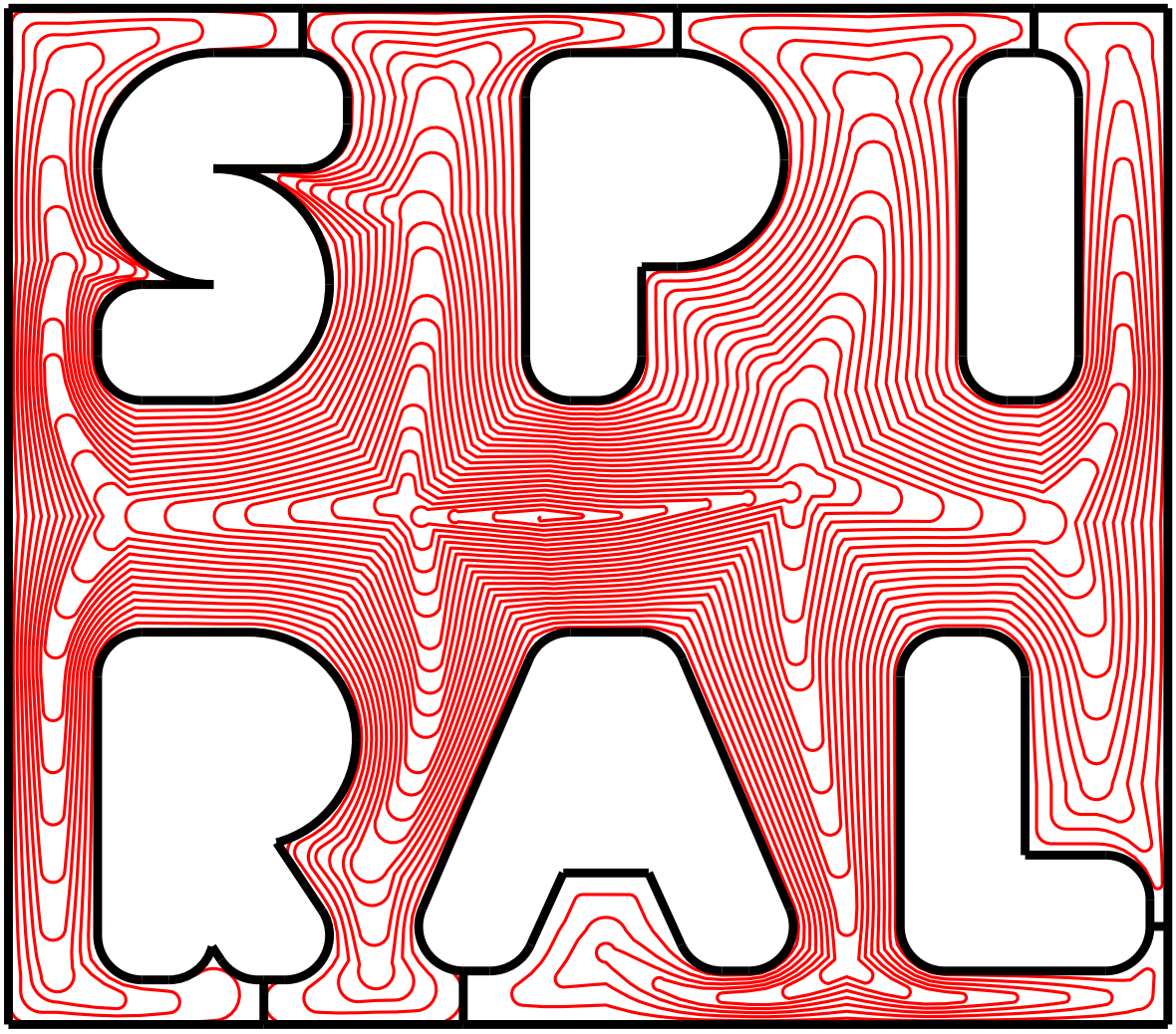
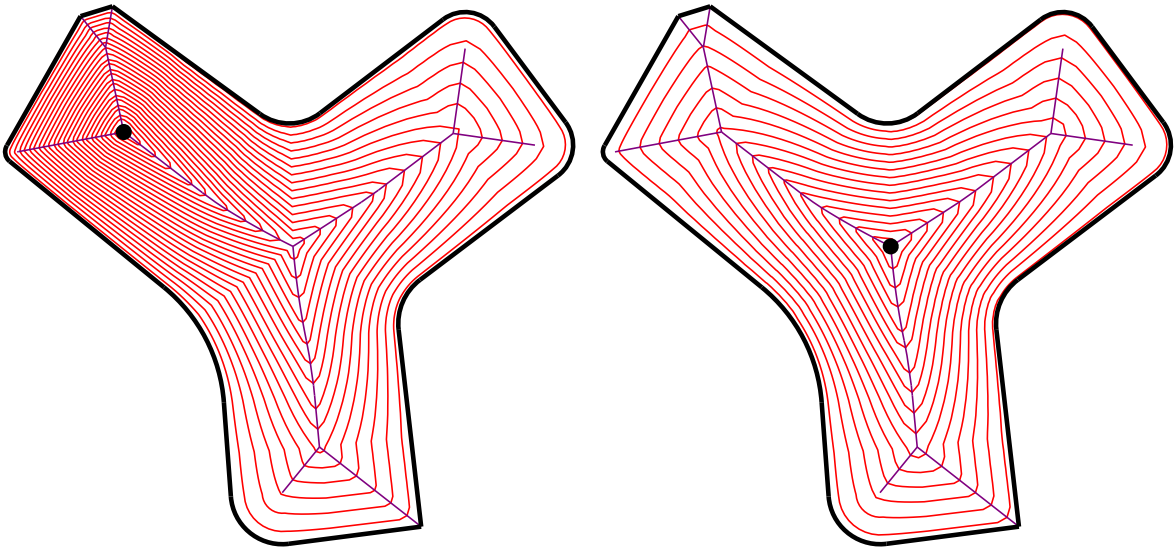


Figure 5.2.: A tool path (derived from the respective dynamic boundary) inside a modified pocket.

6. Finding the Optimal Root

In conventional pocket milling strategies the step-over is a fixed value that describes the distance between two adjacent cutter passes. In most cases the user is able to define the depth of the cut relative to the cutter diameter. In our approach the step-over is no longer static. That is, it is no longer guaranteed and rather likely that the distance between successive laps varies. In general, a minimal step-over variation is sought, because it reduces the tool wear and corresponds to a shorter tool path. In other words, it decreases the number of laps required to machine the interior of a given pocket.



(a) (b)

Figure 6.1.: A tool path whose start point has been chosen arbitrarily (a) as well as a cutter trajectory derived from a balanced medial axis tree (BMAT) (b) are displayed.

6.1. The balanced medial axis tree

Obviously, the potency of the impulse at a point p is directly proportional to the magnitude of the clearance lines starting at p at a specific time $t \in [0, 1]$. Based on this knowledge, Held and Spielberger [2014] presented the idea of balancing the medial axis tree to minimize the step-over variation.

Definition 6.1 (Balanced medial axis tree)**.** If $T'_r(P)$ is a medial axis tree rooted at r of a pocket P without islands, then $T'_r(P)$ is balanced if

$$\exists p, q \in T'_r(P), \text{ such that } \operatorname{dist}(p, r) = \operatorname{dist}(q, r) = \sup\{\operatorname{dist}(u, r) : u \in T'_r(P)\},$$

where $p \neq q$. We call r the height balanced root.

It is easy to formulate an algorithm that calculates the BMAT efficiently (see Algorithm 6.1). In Figure 6.1 the impact of the height balanced root is shown.

Algorithm 6.1 Calculate the height balanced root r of the medial axis $\mathcal{MA}'(P)$ derived from a pocket P. The initial input is given by an arbitrary node of $\mathcal{MA}'(P)$. The function $\operatorname{split}(v, d)$ computes a point p on the corresponding edge e between the vertex v and its parent u such that $\operatorname{dist}(u, p) = d$.

```
1:  function BalanceTree(r)
2:      d ← 0
3:      while height(r) > d do
4:          C ← children of r
5:          f ← n with height(n) = sup{height(c) : c ∈ C}
6:          s ← m with height(m) = sup{height(c) : c ∈ C \ {f}}
7:          if height(s) + dist(r, s) > d then
8:              d ← height(s) + dist(r, s)
9:          end if
10:         d ← d + dist(r, f)
11:         r ← f
12:     end while
13:     r ← split(r, d − height(r))
14:     return r
15: end function
```

Lemma 6.2. Consider the pocket region P without islands, and let $T'_p(P)$ be the corresponding medial axis tree inside P rooted at an arbitrary non-leaf node p. The algorithm outlined in Algorithm 6.1 computes the root $r \in \mathcal{MA}'(P)$ such that $T'_r(P)$ forms a BMAT in $O(n)$ time, where n is the number of nodes of $\mathcal{MA}'(P)$.

Proof. First of all, it is easy to see that r is uniquely defined (the points p, q of Definition 6.1 need not be uniquely defined, though). Obviously, the number of executions of the outer loop is bound by $O(n)$. The heights of all nodes in the medial axis tree are already known, i.e., they can be retrieved in $O(1)$ time. Finally, splitting the edge also consumes a constant amount of time. This yields a linear number of steps overall. □

7. Decomposition of Complex Pockets

In an abstract sense, the complexity of a pocket depends on the number of narrow passages it contains. Intuition tells us that a more complex pocket takes more laps to be milled out entirely. In the following section we will take a closer look at this issue.

7.1. Step-over variation

First of all, it has to be formalized how a complex pocket can be identified. A concept that seems well-suited for our purpose is the step-over variation, which was first described by Held and Spielberger [2014]. Let $\mathcal{TP}(P, \delta)$ be the cutter trajectory used to machine the pocket P, with $\delta \in \mathbb{R}^+$. Furthermore, let $\chi_{\max}(\mathcal{TP}(P, \delta))$ and $\chi_{\min}(\mathcal{TP}(P, \delta))$ be the maximum as well as the minimum step-over of $\mathcal{TP}(P, \delta)$. The step-over variation $\chi_{\text{var}}(\mathcal{TP}(P, \delta))$ is given by

$$\chi_{\text{var}}(\mathcal{TP}(P, \delta)) := \frac{\chi_{\max}(\mathcal{TP}(P, \delta))}{\chi_{\min}(\mathcal{TP}(P, \delta))}.$$

Note, we assume that $\mathcal{TP}(P, \delta)$ is derived from the medial axis tree $T'_r(P)$, where $T'_r(P)$ is height balanced.

7.2. Practical obstacles

The exact calculation of $\chi_{\max}(\mathcal{TP}(P, \delta))$ as well as $\chi_{\min}(\mathcal{TP}(P, \delta))$ is tied to high computational cost due to the simple fact that the actual tool path $\mathcal{TP}(P, \delta)$ has to be obtained before $\chi_{\max}(\mathcal{TP}(P, \delta))$ and $\chi_{\min}(\mathcal{TP}(P, \delta))$ can be acquired. An estimator for the step-over variation (which is based on the medial axis tree of the input pocket alone) is needed. The maximum step-over can be approximated by

$$\chi'_{\max}(T'_r(P)) := \frac{h_{T'_r(P)}(r)}{m},$$

where m is the estimated number of laps required to machine the interior of P. Similar to the maximum step-over, the minimum step-over inside an arbitrary pocket P is estimated by

$$\chi'_{\min}(T'_r(P)) := \frac{h_{T'_r(P)}(v)\,\text{clr}(v)}{h_{T'_r(P)}(r)\,m},$$

where v is the internal node on the medial axis of P with minimal clearance. After putting these pieces together, we are able to estimate the step-over variation of a tool path derived

from $T'_r(P)$ through

$$\chi'_{\text{var}}(T'_r(P)) := \frac{\chi'_{\max}(T'_r(P))}{\chi'_{\min}(T'_r(P))}.$$

For the sake of simplicity, we will call $\chi'_{\text{var}}(T'_r(P))$ the (estimated) step-over variation of $T'_r(P)$.

7.3. General idea

As already discussed, complex pockets have several properties that make them unsuited to be machined in a single cutter pass, that is, without retraction moves, using the aforementioned tool-path generation strategy. In general, it can be stated that a complex cutter trajectory

- has an exceedingly high step-over variation,
- and needs disproportional number of of laps to be milled out.

To overcome these problems, it is recommended to partition the initial pocket P into a number of sub-pockets. This technique tends to reduce the step-over variation of the tool paths inside these sub-pockets significantly, in respect to the cutter-trajectory which would be necessary to mill the superordinate cavity.

Keep in mind that every such subdivision introduces an additional retraction move, which is especially crucial in a HSM environment. This means that the number of resulting sub-pockets must not be ignored if we want to be able to machine the workpiece efficiently.

The actual decomposition is done through a simple recursive procedure, that stops as soon as the gain in terms of the step-over variation does no longer outweigh the overhead of the additional non-cutting tool motions.

The main differences between our scheme and the strategy presented in [Held and Spielberger, 2014] are that

- the separators which are used to subdivide the initial pocket are (for the sake of simplicity) defined by a single straight-line segment,
- we are using a quality function to determine the concrete separators from a set of candidates,
- and the termination criteria has been slightly modified, such that the granularity of the decomposition can be adjusted through a single variable.

7.4. Choosing the separators

In each decomposition step a straight-line segment, the so-called separator, has to be chosen. It partitions the original pocket P into two areas. We suppose that P is available such that the corresponding medial axis forms a tree, i.e., for pockets that include islands additional preprocessing is needed. Based on this assumption a separator is defined as follows.

Definition 7.1 (Separator)**.** Let $\mathcal{MA}(P)$ the medial axis of the pocket P. Further, let c be a point on $\mathcal{MA}(P)$, such that c does not coincide with a vertex of $\mathcal{MA}(P)$. If p and q are the two contact points of d_c with the corresponding defining sites, then $s := \overline{pq}$ is called a separator of P related to c.

Trivially, there exists an infinite number of possible separators inside an arbitrary pocket P. Therefore, it is not feasible to test all possibilities. A set S of candidate separators is created. This is done by iterating over all edges E of the medial axis tree $T'_r(P)$. For each edge $e \in E$, a candidate separator s related at p is generated, such that p is the point with minimal clearance on e. Afterwards, every individual element $s \in S$ is inspected, and its quality $q(s)$ is determined, with

$$q(s) := \frac{h_{T'_r(P)}(c)}{h_{T'_r(P)}(c) \cdot \text{len}(s) + 1},$$

where $h_{T'_r(P)}(c)$ is the Euclidean height of the center c of s. It is easy to see that the quality of a separator depends on two major factors.

- Obviously, the length of the separator has to be taken into account, due to the simple fact that shorter separators correspond to narrow bottlenecks. In general, closing tight corridors results in a significant reduction in terms of the overall step-over variation.
- Considering the Euclidean height of a separator guarantees that the initial pocket is split evenly.

The separator s' with the highest quality is chosen to be the concrete separator of P. It partitions P into two sub-pockets P'_0 and P'_1. This subdivision is identified by the triple (P, s', l), where l holds the decomposition level, which is incremented after every decomposition step.

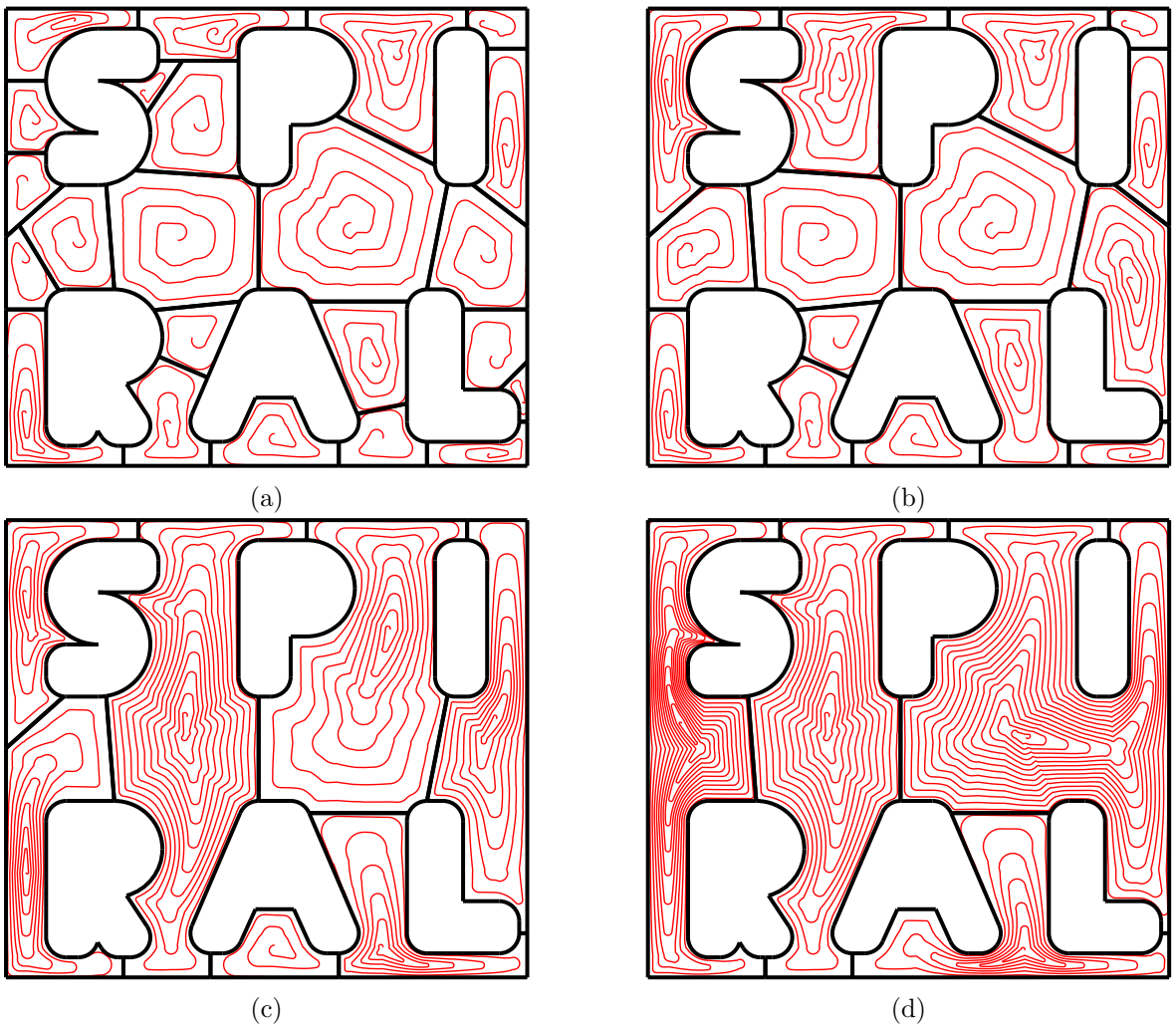
7.5. Termination criteria

As mentioned before, we seek to employ a recursive scheme to decompose the original pocket into smaller regions. This makes it necessary to formulate a termination criteria. It has to be determined whether a partition $D := (P, s', l)$ of a pocket P by the separator s' should be accepted. Remember that the goal of the decomposition algorithm is

- to reduce the step-over variation,
- while simultaneously keeping the number of sub-pockets low.

Based on these requirements the following heuristic was designed. It decides whether D is valid or not. Let $D := (P, s', l)$ be a subdivision of the pocket P by the separator s' into the sub-pockets P'_0 and P'_1, D is called an acceptable partition if the condition

$$\frac{1}{4}(2\eta l + 1)\left(\chi'_{\text{var}}\left(T'_{r'_0}(P'_0)\right) + \chi'_{\text{var}}\left(T'_{r'_1}(P'_1)\right)\right) < \chi'_{\text{var}}(T'_r(P))$$



(a) (b) (c) (d)

Figure 7.1.: The illustrations (a)-(d) show the result of increasing decomposition factors.

holds, where $\eta \in \mathbb{R}_0^+$. The value η controls the impact of the decomposition level on the actual subdivision. We call η the decomposition factor. Trivially, a higher η results in a greater number of retraction moves, but also tends to decrease the average step-over variation of the generated sub-pockets. In Figure 7.1 the result of increasing decomposition factors is displayed.

7.6. Decomposition algorithm

Now that the termination criteria has been established, it is possible to formulate a simple procedure that carries out the actual pocket decomposition.

Given: A pocket P without islands and as well as the desired decomposition factor η.

Determine: A decomposition of P that respects η.

1. Initialize the solution set $S = \emptyset$ and the decomposition level $l = 0$.

2. Compute the medial axis tree $T'_r(P)$ of the input pocket P.

3. Transform $T'_r(P)$ into a BMAT $T'_{r'}(P)$ through calculating the height balanced root r'.

4. Find the concrete separator s' of P, i.e., the candidate separator with the highest quality, and determine the corresponding subdivision $D := (P, s', l)$.

5. If D represents an acceptable decomposition, then set $S = \{S \setminus \{P\}\} \cup \{P'_1, P'_2\}$ (where P'_1 and P'_2 are two sub-pockets generated by inserting s' into P), increment l, and continue with step (2) for P'_1 as well as P'_2.

6. Finally, S holds the resulting sub-pockets.

The individual decomposition steps can be carried out in $O(n \log n)$ time, where n is the number of input sites, if the candidate separators are stored in a priority queue ordered based in their quality. Recall that there exist $O(n)$ many candidate separators.

8. Double Spiral

It is obvious that a spiral constructed by the algorithm discussed earlier always starts at the medial axis, and ends at the border ∂P of the corresponding pocket P. In this chapter an alternative approach is described, which produces a spiral that

- respects the user-specified maximum step-over δ,
- is not self-intersecting,
- and starts as well as ends on ∂P.

We will incorporate the idea of a double spiral to generate a path that suffices these requirements. Therefore, two interleaved, curled curves are created and linked together.

8.1. Generating the spirals

As already mentioned, we distinguish two spirals, namely, the inner spiral $\mathcal{SI}(P)$ as well as the outer spiral $\mathcal{SO}(P)$. The key idea behind the following strategy is to construct two initial laps by interpolating between wavefronts. Based on those initial laps an alternating sequence of laps whose elements belong to $\mathcal{SI}(P)$ and $\mathcal{SO}(P)$, respectively, is generated. Finally, $\mathcal{SI}(P)$ and $\mathcal{SO}(P)$ are connected. A more detailed description of this construction scheme follows.

1. Initially, the time interval $[0, 1]$ is subdivided into m parts such that $0 = t_0 < t_1 < ... < t_m = 1$, with $t_{i+1} - t_i = \Delta t$, and $i \in \{0, 1, ..., m-1\}$, where Δt is calculated as before.

2. Again, two laps R_1 and R_m are produced by interpolating between $w(t_0)$ and $w(t_2)$, as well as $w(t_{m-1})$ and $w(t_m)$. The lap R_1 is the first lap of $\mathcal{SI}(P)$.

3. Clearly, R_1 starts at r and ends at a point q on $T'_r(P)$. The start point of $\mathcal{SO}(P)$ is chosen to be the point p that lies halfway between r and q along $T'_r(P)$. The corner p marks the start point of $\mathcal{SO}(P)$.

4. Based on R_1 the laps of $\mathcal{SI}(P)$ and $\mathcal{SO}(P)$ are constructed alternately by expanding the individual corners gradually along $T'_r(P)$. This process is similar to the interpolation scheme we applied in the case of single spirals.

   For each corner p of R_1, it is checked if the difference in starting time between p and q is at most $(m-1)\,\Delta t$, where q is the respective corner on R_m. Whenever this condition

is violated, q is moved towards p along $T'_r(P)$ until $t_q = t_p + (m-1)\,\Delta t$ to guarantee that the distance invariant is respected.

Assume that the i-th lap R_i, with $i \in \{1, 2, ..., m-2\}$, has already been constructed. The $(i+1)$-th lap is produced by moving every corner p of R_i down $T'_r(P)$ until the maximum expansion $\upsilon_i(p)$ is reached.

5. Special precautions have to be taken as soon as R_{m-1} has been established. It is presumed that all previous laps have been generated. Additionally, all even and odd numbered laps (with an index lower than $m-1$) are connected successively at their start and end points, respectively. Furthermore, the lap R_{m-1} is connected to R_m.

   One loose end needs to be taken care of, namely, the end point v of R_{m-2}. The goal is to construct a lap R_{m+1} which starts at v and ends at w, where w is the end point of R_m. Trivially, w lies on ∂P. We start at v and construct R_{m+1} such that it moves progressively closer to R_m until both laps meet in w. During this process we have to pay attention, and verify that each corner of R_{m+1} is at most δ away from the nearest corner of R_m along $T'_r(P)$. If a violation of the distance invariant is detected, then the corresponding corner of R_m is moved closer to R_{m+1}. Note that it is never necessary to move corners of R_m above the respective corners of $w(t_{m-1})$.

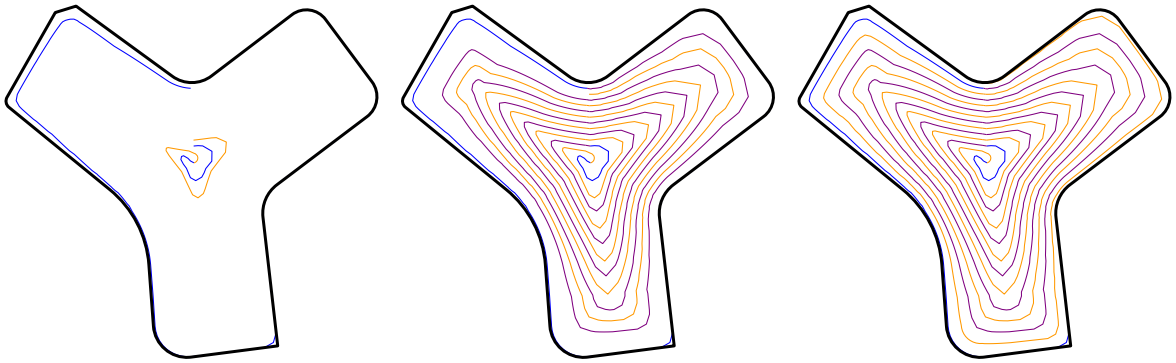
6. Finally, the start points of $\mathcal{SI}(P)$ and $\mathcal{SO}(P)$ are connected through a circular arc a. Trivially, $\mathcal{SI}(P)$ and $\mathcal{SO}(P)$ both form polygonal chains. By traversing the straight-line segments $\mathcal{SI}(P)$ in reverse order, followed by a, and the straight-line segments of $\mathcal{SO}(P)$, the final double spiral $\mathcal{DS}(P, \delta)$ is produced. See Figure 8.1 for an illustration of this strategy.

   Note that by choosing the interpolation steps during the construction of R_m and R_{m+1} accordingly, it is possible to place the end points of $\mathcal{DS}(P, \delta)$ freely on ∂P.

8.2. Dealing with pocket decomposition

It was already stated that the key disadvantage of the pocket decomposition strategy is that for each subdivision that is generated, a retraction move is introduced if single spirals are used. That is, spirals that start at the interior of a pocket P and end on ∂P. By using double spirals in the corresponding sub-pockets, it is possible to eliminate retraction moves. All sub-pockets can be covered using a single spiral which we will refer to as composite spiral $\mathcal{SC}(P, \delta)$.

Let $\mathcal{P} := \{P_0, P_1, ..., P_k\}$ be a set of sub-pockets produced by decomposing the pocket P. A graph $G := (V, E)$ is derived from $\mathcal{P}$. For each sub-pocket P_i, with $i \in \{0, 1, ..., k\}$, a vertex v_i is added to V. The edge set E holds undirected edges of the form $\{v_i, v_j\}$, with $i, j \in \{0, 1, ..., k\}$, and $i \neq j$, such that the ∂P_i and ∂P_j have at least one site in common. Additionally, we define a weight function $\omega : E \to \mathbb{R}_0^+$ which assigns a value $\omega(e)$ to each edge



(a) (b) (c)

Figure 8.1.: Firstly, the initial lap R_1 as well as the last lap R_m are calculated (a). Afterwards, the subsequent laps $R_2, R_3, ..., R_{m-1}$ of $\mathcal{SI}(P)$ and $\mathcal{SO}(P)$ are generated alternately (b). Finally, R_{m+1} is produced, which starts at the end point of R_{m-2}, and ends at the end point of R_m (c).

$e \in E$, where $\omega(e)$ equals the multiplicative inverse of the accumulated lengths of all common sites of these sub-pockets. Trivially, G together with ω forms a weighted graph. Hence, it is possible to compute the EMST $S(G)$ of G in $O(|V| \log |V|)$ time. A sequence of paths inside $S(G)$ is constructed. Moreover, an arbitrary leaf v^* is specified to be the root of $S(G)$.

We traverse $S(G)$ in depth-first order, and gather information as follows. Inside $S(G)$ the longest path M between v^* as well as a leaf v' is computed, and added to $\mathcal{Q}$, where $\mathcal{Q}$ is a set of paths. For every node w^* in M that has a degree higher than one, a new path Q is generated which starts at w^* and ends at a leaf w' of $S(G)$, with $w' \neq v'$. This process is repeated recursively for each vertex in Q. All paths which are created during this procedure are also added to $\mathcal{Q}$.

We end up with a set of paths $\mathcal{Q}$ inside $S(G)$. One element of $\mathcal{Q}$ is special, namely, the main path M. In particular, the start node v^* as well as the end node v' of M determine where the composite spiral starts and ends, respectively. Thus, the sub-pockets that correspond to v^* and v' are the only elements of $\mathcal{P}$ in which a single spiral is computed. In every other sub-pocket we generate a double spiral.

Lastly, these spirals need to be connected. All spirals whose corresponding vertices are neighbors in M are connected at one of their common sites. That is, the end point of the current spiral is connected to the start point of the next one until v' is reached. This is straightforward due to the fact that the start as well as the end points of these double spirals can be chosen freely on the border of the corresponding pocket.

If $|\mathcal{Q}| > 1$, then we look at each remaining path $Q \in \mathcal{Q}$ individually. Two consecutive elements $q_1, q_2 \in Q$ are considered at once. Let P_1 and P_2 be the respective sub-pockets. In both P_1 as well as P_2 a double spiral $\mathcal{DS}(P_1, \delta)$ and $\mathcal{DS}(P_2, \delta)$, respectively, is constructed. We choose the start point p as well as the end point q of $\mathcal{DS}(P_2, \delta)$ to be situated on the common site s of P_1 and P_2. Furthermore, it is assumed that the last orbit $\mathcal{DS}(P_1, \delta)$ moves along s. Therefore, it is possible to split $\mathcal{DS}(P_1, \delta)$ at p and q. The portion of $\mathcal{DS}(P_1, \delta)$ that

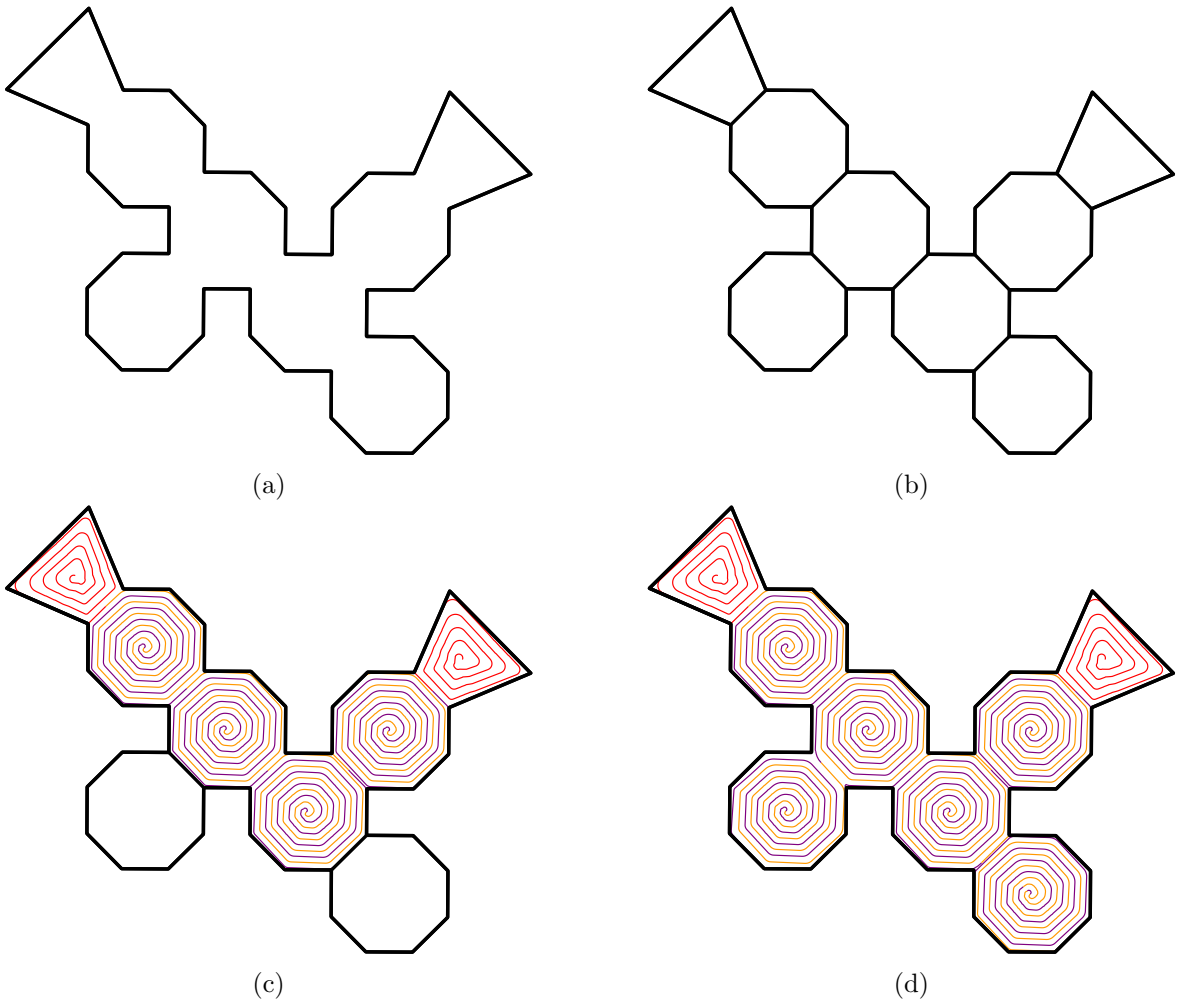


Figure 8.2.: The initial pocket (a) is decomposed in several sub-pockets (b). Afterwards, the spirals along the longest path are computed and connected (c). Finally, the spirals in the remaining sub-pockets are calculated (d).

lies between p as well as q is removed. Thus, connecting $\mathcal{DS}(P_1, \delta)$ and $\mathcal{DS}(P_2, \delta)$ becomes straight-forward.

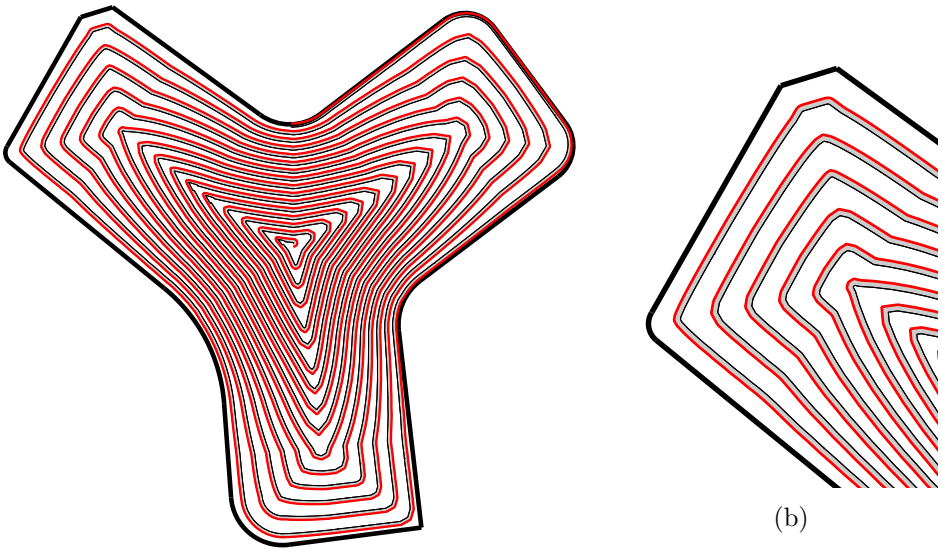
See Figure 8.2 for a typical application of this strategy. At the end of this procedure a composite spiral is generated which covers the entire pocket P.

9. Smoothing the Tool Path

For now we have obtained a tool path $\mathcal{TP}(P, \delta^*)$, where P is a pocket, and $\delta^* \in \mathbb{R}^+$, that is described by a polygonal chain. Obviously, $\mathcal{TP}(P, \delta^*)$ still shows discontinuities. From a kinematic standpoint each of these discontinuities forces the cutter to slow down rapidly only to accelerate immediately afterwards, i.e., the speed of the tool changes drastically at such locations. To avoid this behavior it is crucial that we boost our cutter trajectory to at least $\mathcal{G}^1$-continuity.

Therefore, we used the PowerApx-package developed by Heimlich and Held [2008]. Amongst other things, it supports the approximation of polygonal chains through biarcs and cubic Bézier curves.

9.1. Choosing the tolerance bands



(a) (b)

Figure 9.1.: The initial tool path as well as the corresponding tolerance band are shown in (a). In (b) a zoomed-in view of the upper left part of the pocket is illustrated.

The approximation procedure presented in [Heimlich and Held, 2008] derives a tolerance band $\mathcal{TB}(\mathcal{TP}(P, \delta^*), d_L, d_R)$ from $\mathcal{TP}(P, \delta^*)$. Intuitively, $\mathcal{TB}(\mathcal{TP}(P, \delta^*), d_L, d_R)$ describes the area in which the resulting approximation curve $\mathcal{TP}'(P, \delta)$ is situated, i.e.,

$$\mathcal{TP}'(P, \delta) \subseteq \mathcal{TB}(\mathcal{TP}(P, \delta^*), d_L, d_R),$$

where $d_L \in \mathbb{R}^+$ and $d_R \in \mathbb{R}$, such that $d_R < d_L$. Furthermore, we know that

$$H(\mathcal{TP}'(P, \delta), \mathcal{TP}(P, \delta^*)) < d_L.$$

Thus, it is important to choose suitable values for d_L as well as d_R, such that $\mathcal{TP}'(P, \delta)$ respects user-specified step-over value δ. We will be using a one-sided tolerance band to guarantee that $\mathcal{TP}'(P, \delta)$ stays inside ∂P. Therefore, we set $d_L = d$ and $d_R = 0$, with $d < \delta$. Obviously, if we set δ^* to be less than $\delta - d$, then the maximum step-over of $\mathcal{TP}'(P, \delta)$ is bound by δ. See Figure 9.1 for an example of a tolerance band obtained from a cutter trajectory.

9.2. Types of approximations

Once the tolerance band has been successfully specified, it is possible the compute the actual approximation. The aforementioned PowerApx-package was utilized to compute biarc (see Figure 9.2) as well as cubic Bézier approximations of the input tool path $\mathcal{TP}(P, \delta^*)$.

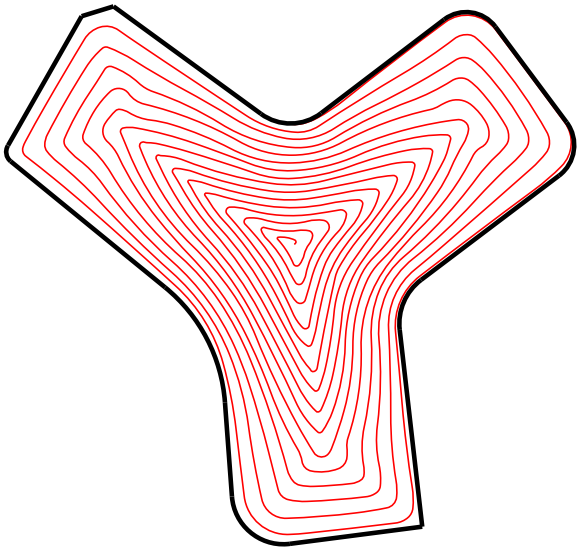


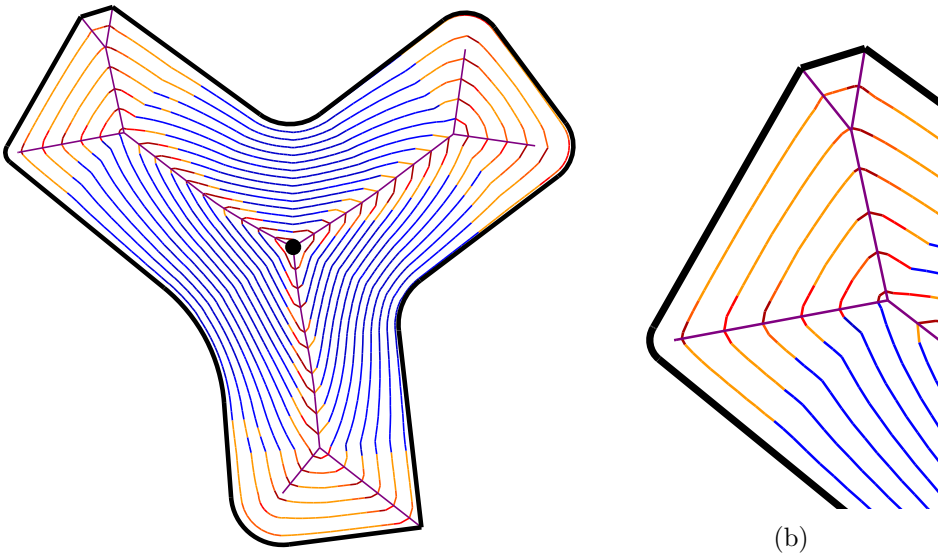
Figure 9.2.: A biarc approximation of a cutter trajectory generated by our tool-path generation strategy.

The resulting approximations are $\mathcal{G}^1$-continuous, which is crucial for HSM. Additionally, Held and Kaaser [2014] presented a strategy for approximating planar curvilinear profiles by uniform cubic B-splines. Using this scheme, it is even possible to boost our cutter trajectory to $\mathcal{C}^2$-continuity.

10. Discussion and Analysis

Amongst other things, the goal of our tool-path generation strategy is to minimize the overall cutter engagement angle. Furthermore, it is crucial to keep the step-over variance low to reduce the length of the resulting cutter trajectory.

We want to study the behavior of the step-over as well as the engagement angle on the example of the tool path $\mathcal{TP}_{\text{opt}}(P, \delta)$ portrayed in Figure 10.1. It is assumed that $\mathcal{TP}_{\text{opt}}(P, \delta)$ is represented by a curve parameterized over the closed time interval $[0, 1]$. Furthermore, let $\mathbb{T}$ be the set of all possible curves generated by our tool-path generation method. The corresponding evaluation functions are listed in Table 10.1. The step-over of a point p on the tool path is given by the Euclidean distance of p to the previous lap. The engagement of p is calculated by placing a circular disk d at p whose radius equals the cutter radius. Clearly, the part of the boundary of d that is not covered by the area already machine forms a circular arc a. We call the angle of a the engagement angle at p.



(a) (b)

Figure 10.1.: A tool paths inside a pocket is shown in (a), where locations of low as well as high cutter engagement are highlighted in blue or red, respectively. The maximum step-over is given by 90% of the cutter diameter. In (b) a zoomed-in view of the upper left part of the respective pocket is illustrated.

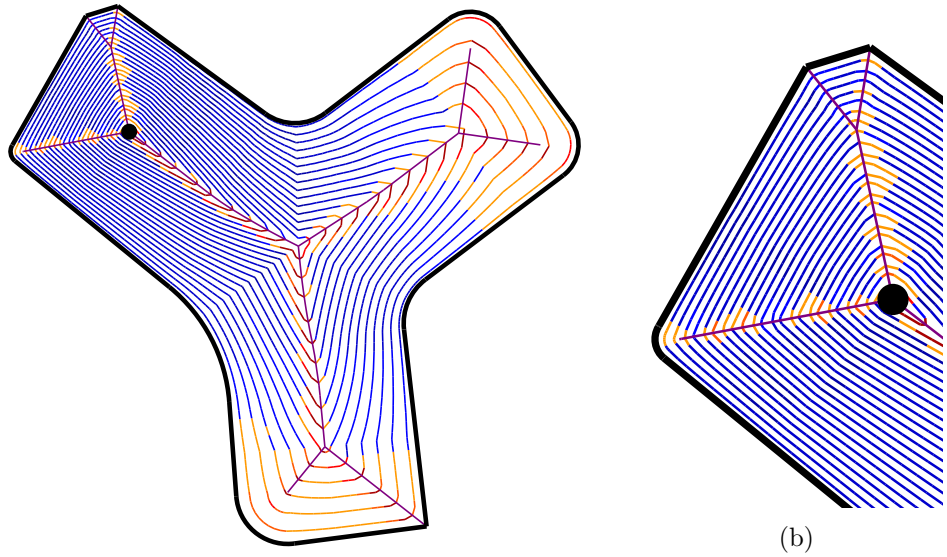
If every lap R of $\mathcal{TP}_{\text{opt}}(P, \delta)$ is considered individually, then it is obvious that the step-over is especially high at areas of R where the impulse is weak (see Figure 10.1).

Clearly, a higher step-over results in an increased engagement angle at these locations.

Function	Description
$\chi : \mathbb{T} \times [0,1] \to \mathbb{R}_0^+$	Step-over at time $t \in [0,1]$ of a tool path $\mathcal{TP}(P,\delta) \in \mathbb{T}$.
$\xi : \mathbb{T} \times [0,1] \to \mathbb{R}_0^+$	Engagement angle at time $t \in [0,1]$ of a tool path $\mathcal{TP}(P,\delta) \in \mathbb{T}$.
$\chi_{\min} : \mathbb{T} \to \mathbb{R}_0^+$	Minimum step-over of the tool path $\mathcal{TP}(P,\delta) \in \mathbb{T}$.
$\chi_{\max} : \mathbb{T} \to \mathbb{R}_0^+$	Maximum step-over of the tool path $\mathcal{TP}(P,\delta) \in \mathbb{T}$.
$\xi_{\min} : \mathbb{T} \to \mathbb{R}_0^+$	Minimum engagement angle of the tool path $\mathcal{TP}(P,\delta) \in \mathbb{T}$.
$\xi_{\max} : \mathbb{T} \to \mathbb{R}_0^+$	Maximum engagement angle of the tool path $\mathcal{TP}(P,\delta) \in \mathbb{T}$.
$\chi_{\text{var}} : \mathbb{T} \to \mathbb{R}_0^+$	$\chi_{\text{var}}(\mathcal{TP}(P,\delta)) := \frac{\chi_{\max}(\mathcal{TP}(P,\delta))}{\chi_{\min}(\mathcal{TP}(P,\delta))}$.
$\xi_{\text{var}} : \mathbb{T} \to \mathbb{R}_0^+$	$\xi_{\text{var}}(\mathcal{TP}(P,\delta)) := \frac{\xi_{\max}(\mathcal{TP}(P,\delta))}{\xi_{\min}(\mathcal{TP}(P,\delta))}$.

Table 10.1.: Evaluation functions used to analyze tool paths.

This observation is by no means surprising due to the simple fact that the impulse travels faster if it is further away from the root of the corresponding medial axis tree.

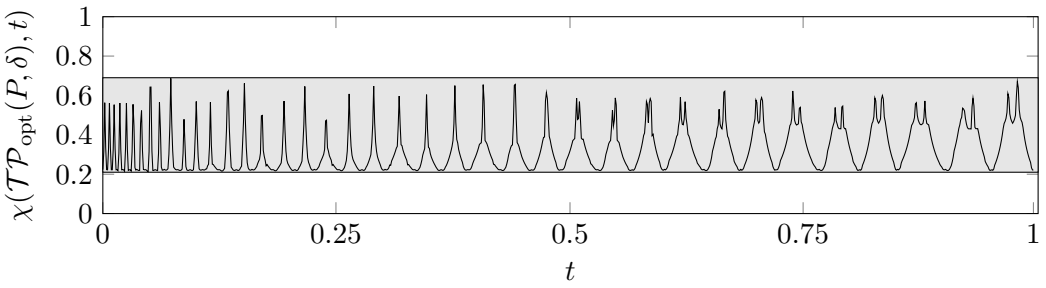


(a) (b)

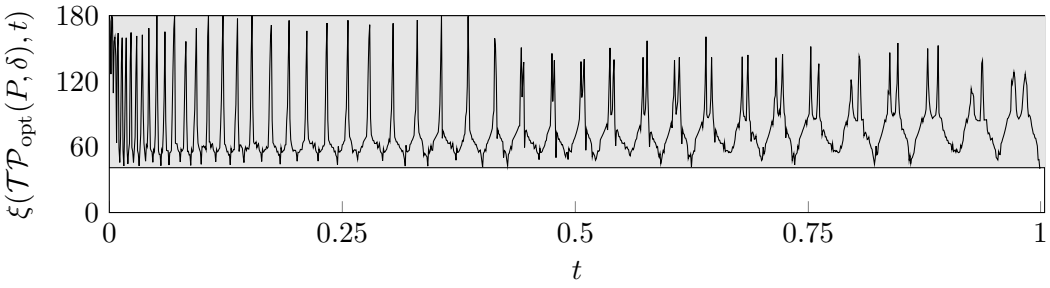
Figure 10.2.: A tool paths inside a pocket is shown in (a). The maximum step-over is given by 90% of the cutter diameter. In (b) a zoomed-in view of the upper left part of the respective pocket is illustrated.

The graphs displayed in Figure 10.3 show the engagement angle and the step-over along the cutter trajectory. Evidently, the initially defined, maximum step-over is respected everywhere. It is noteworthy that both the step-over values as well as the engagement angles are largely low.

Additionally, the root of the corresponding medial axis tree is height balanced. Thus, we end up with a cutter trajectory that has a nearly optimal step-over variation, in terms of our tool-path generation strategy. An alternative tool path $\mathcal{TP}_{\text{sub}}(P,\delta)$ with a sub-optimal root is displayed in Figure 10.2. Trivially, $\mathcal{TP}_{\text{sub}}(P,\delta)$ needs significantly more

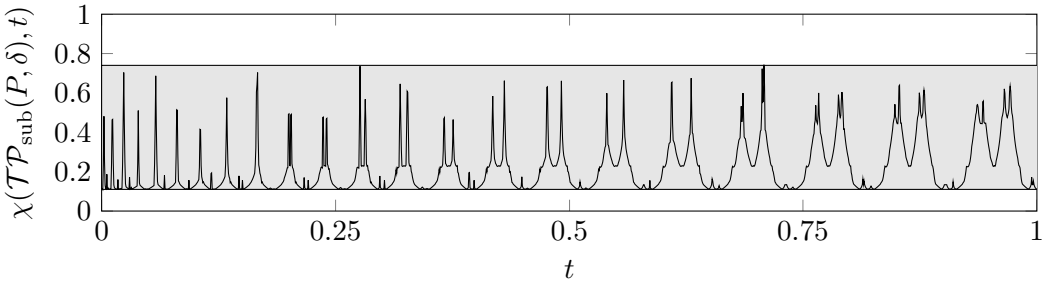


(a) The step-over along the entire tool path.

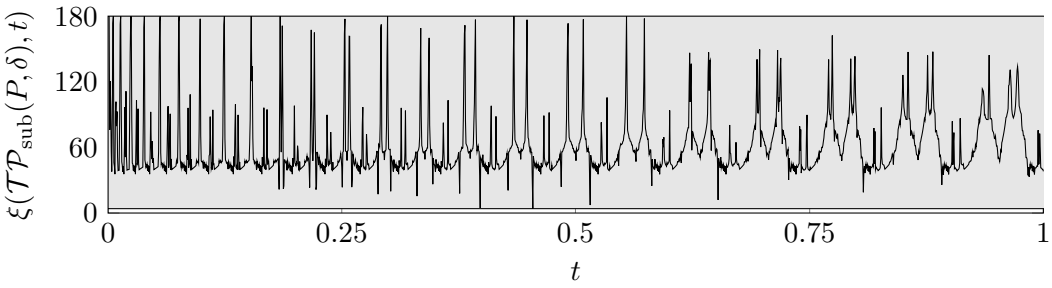


(b) The engagement angle along the cutter trajectory.

Figure 10.3.: Analysis of the tool path with the optimized start point.



(a) The step-over along the entire tool path.



(b) The engagement angle along the cutter trajectory.

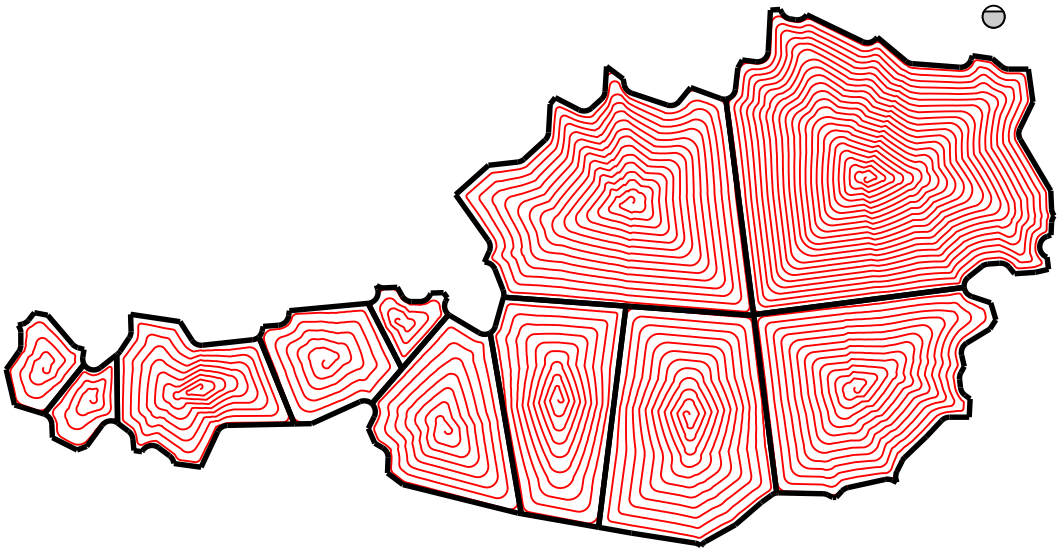
Figure 10.4.: The engagement angle as well as the step-over along the sub-optimal tool path.

laps the mill out P entirely, compared to $\mathcal{TP}_{\text{opt}}(P,\delta)$. As a consequence, $\mathcal{TP}_{\text{sub}}(P,\delta)$ is approximately 1.478 times longer than $\mathcal{TP}_{\text{opt}}(P,\delta)$. Furthermore, the step-over variation of $\mathcal{TP}_{\text{sub}}(P,\delta)$, with $\chi_{\text{var}}(\mathcal{TP}_{\text{sub}}(P,\delta)) \sim 6.862$, is considerably higher than that of $\mathcal{TP}_{\text{opt}}(P,\delta)$, with $\chi_{\text{var}}(\mathcal{TP}_{\text{opt}}(P,\delta)) \sim 3.223$. The unoptimized tool path has an engagement variation of

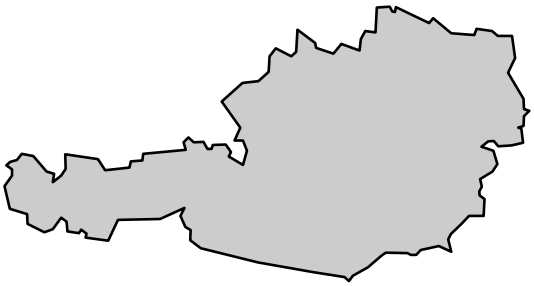
$\xi_{\text{var}}(\mathcal{TP}_{\text{sub}}(P,\delta)) \sim 46.986$, whereas the optimized cutter trajectory has an engagement variation of $\xi_{\text{var}}(\mathcal{TP}_{\text{opt}}(P,\delta)) \sim 4.436$ (see Figure 10.4).

A. Examples

Please note that the maximum step-over values of the subsequent tool paths have been chosen exceedingly high in order to display the cutter trajectories in a meaningful way. That is, it is advised to choose the maximum depth of the cut significantly lower for real world applications.

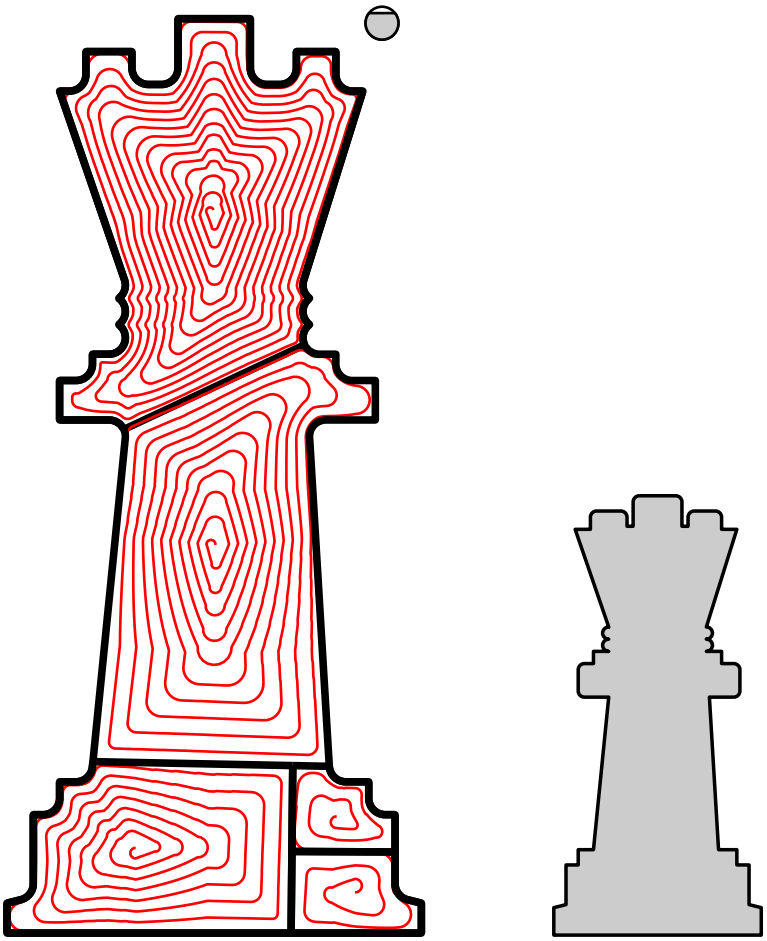


(a)



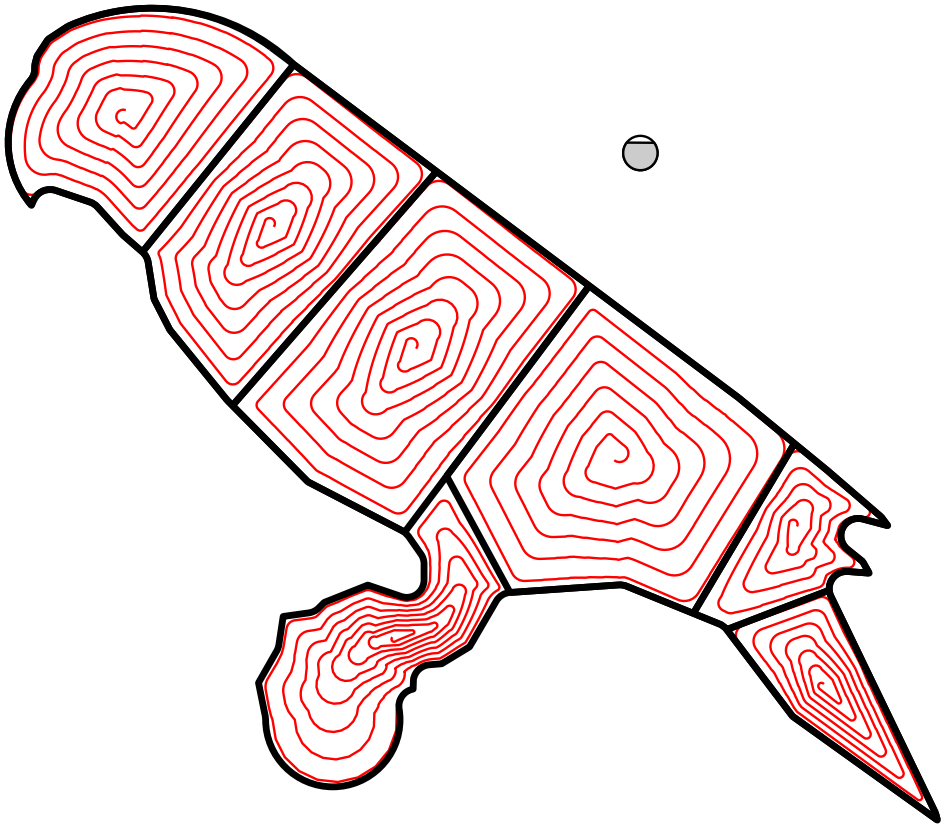
(b)

Figure A.1.: In (a) a decomposition of a complex pocket together with the corresponding tool paths (derived from the dynamic boundary) is illustrated. The maximum step-over δ was set to 80% of the cutter diameter. Furthermore, the decomposition factor η is chosen to be 0.5. The circle (partially) shaded in gray indicates the size of the cutter relative to the pocket. The shape of the resulting cavity is outlined in (b).

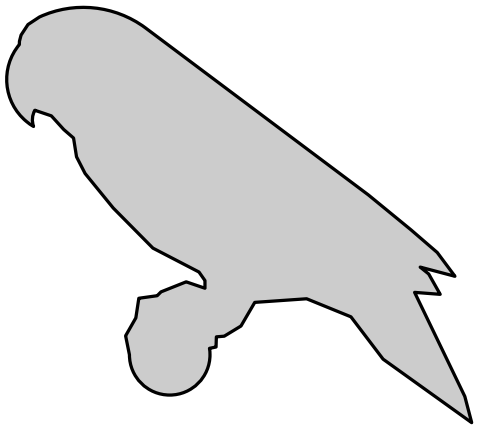


(a) (b)

Figure A.2.: In (a) a decomposition of a complex pocket together with the corresponding tool paths (derived from the dynamic boundary) is illustrated. The maximum step-over δ was set to 90% of the cutter diameter. Furthermore, the decomposition factor η is chosen to be 0.5. The circle (partially) shaded in gray indicates the size of the cutter relative to the pocket. The shape of the resulting cavity is outlined in (b).

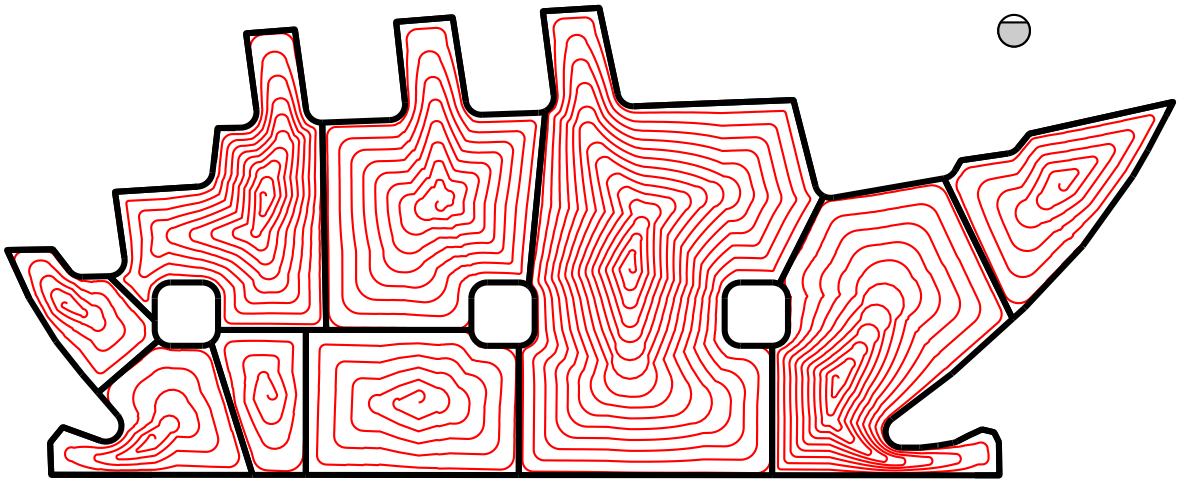


(a)

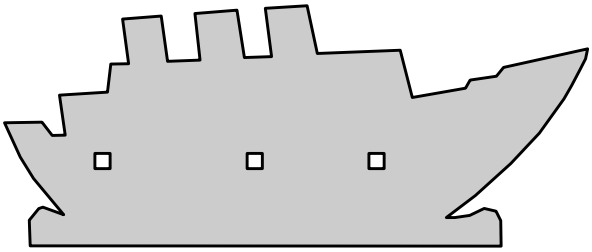


(b)

Figure A.3.: In (a) a decomposition of a complex pocket together with the corresponding tool paths (derived from the dynamic boundary) is illustrated. The maximum step-over δ was set to 90% of the cutter diameter. Furthermore, the decomposition factor η is chosen to be 0.5. The circle (partially) shaded in gray indicates the size of the cutter relative to the pocket. The shape of the resulting cavity is outlined in (b).



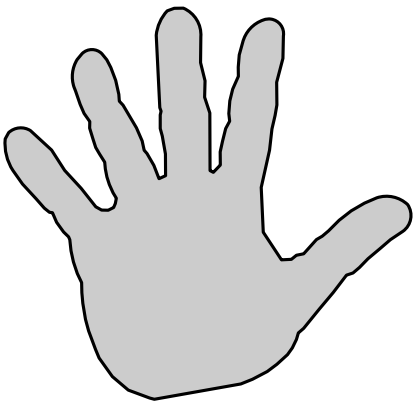
(a)



(b)

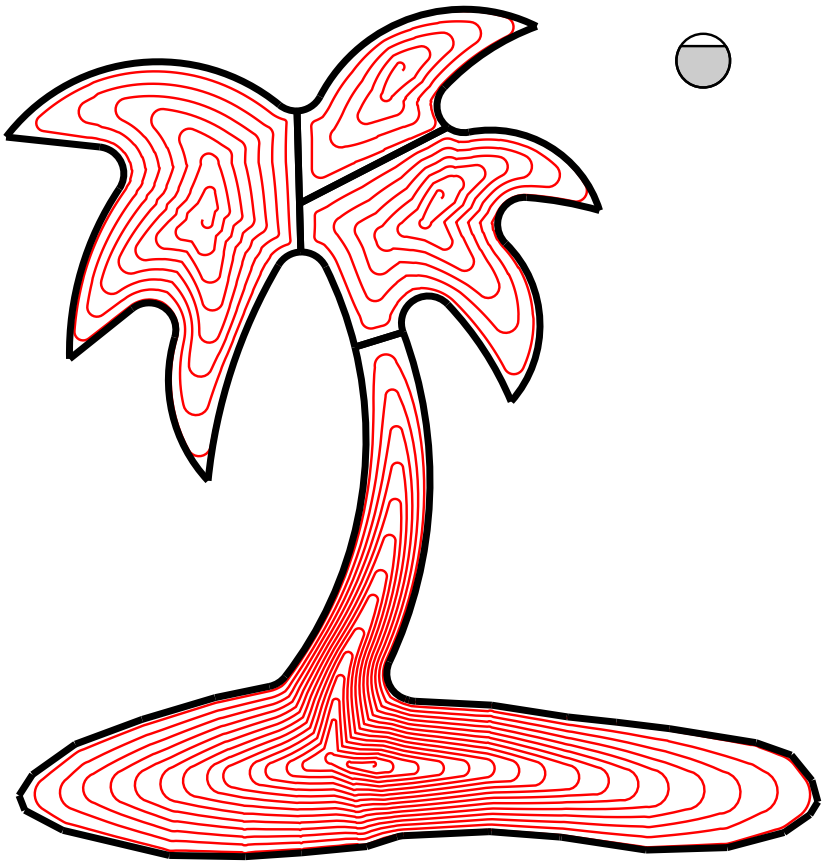
Figure A.4.: In (a) a decomposition of a complex pocket together with the corresponding tool paths (derived from the dynamic boundary) is illustrated. The maximum step-over δ was set to 90% of the cutter diameter. Furthermore, the decomposition factor η is chosen to be 0.5. The circle (partially) shaded in gray indicates the size of the cutter relative to the pocket. The shape of the resulting cavity is outlined in (b).

(a)

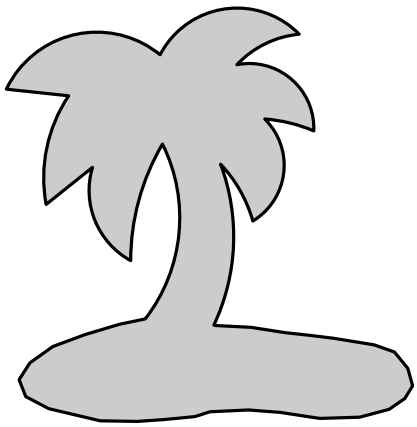


(b)

Figure A.5.: In (a) a decomposition of a complex pocket together with the corresponding tool paths (derived from the dynamic boundary) is illustrated. The maximum step-over δ was set to 90% of the cutter diameter. Furthermore, the decomposition factor η is chosen to be 0.5. The circle (partially) shaded in gray indicates the size of the cutter relative to the pocket. The shape of the resulting cavity is outlined in (b).

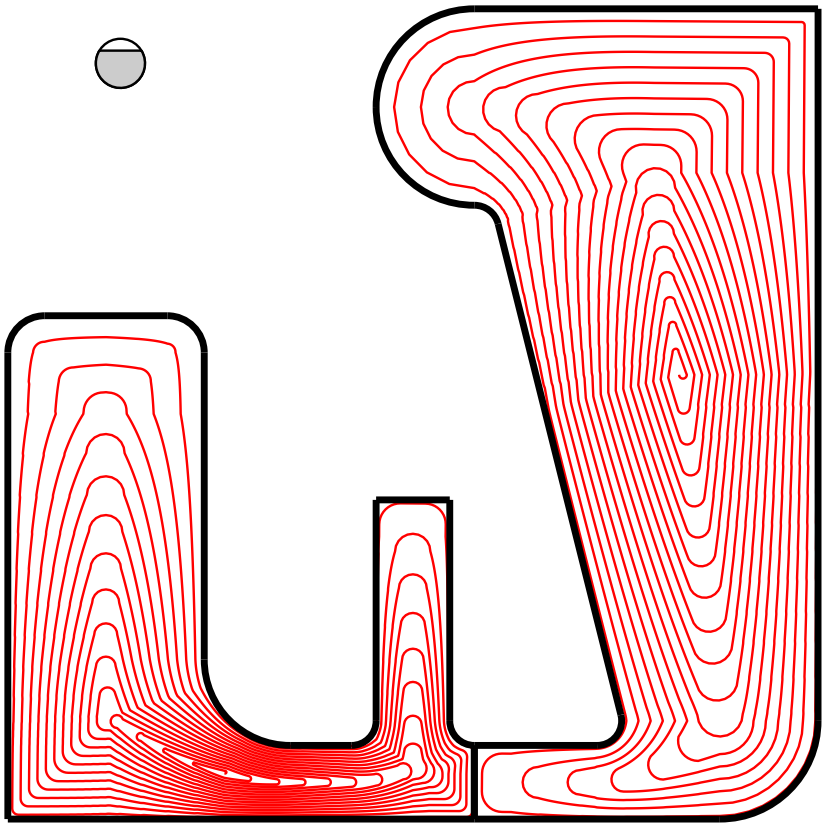


(a)

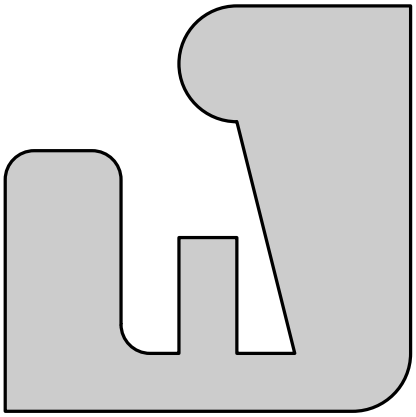


(b)

Figure A.6.: In (a) a decomposition of a complex pocket together with the corresponding tool paths (derived from the dynamic boundary) is illustrated. The maximum step-over δ was set to 80% of the cutter diameter. Furthermore, the decomposition factor η is chosen to be 0.5. The circle (partially) shaded in gray indicates the size of the cutter relative to the pocket. The shape of the resulting cavity is outlined in (b).

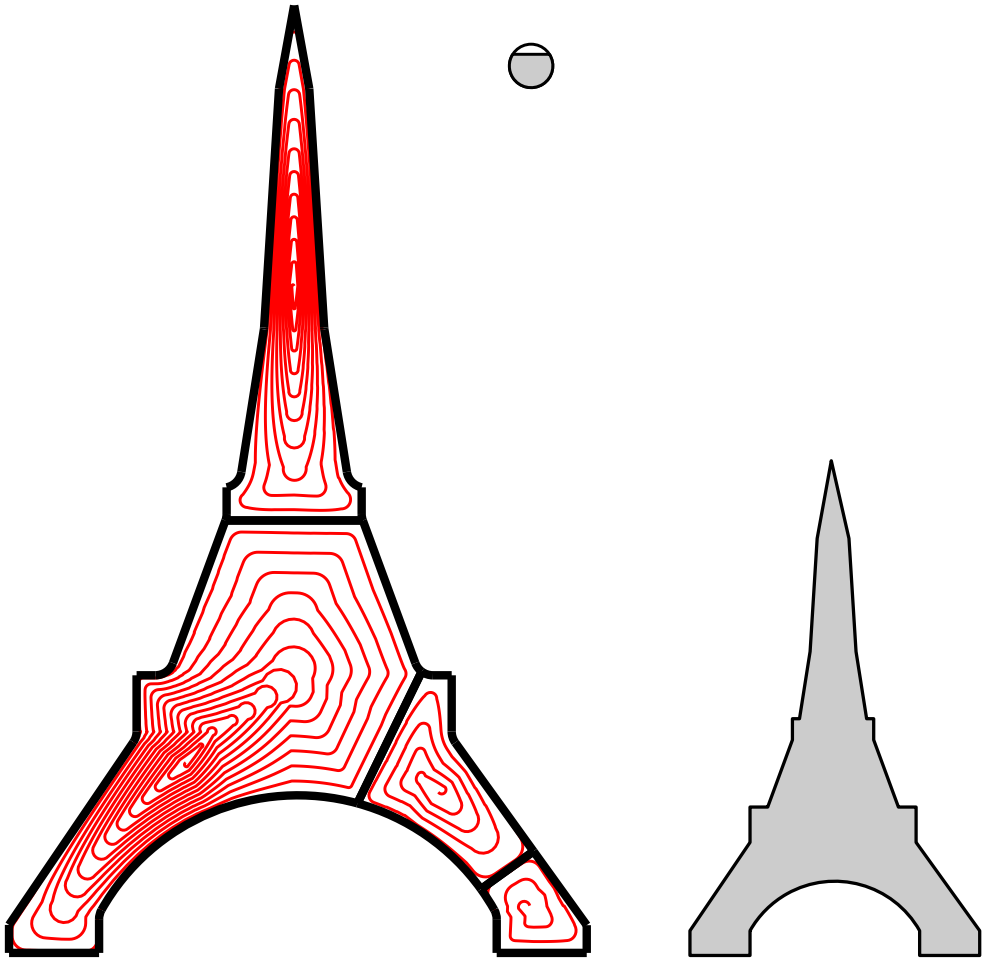


(a)



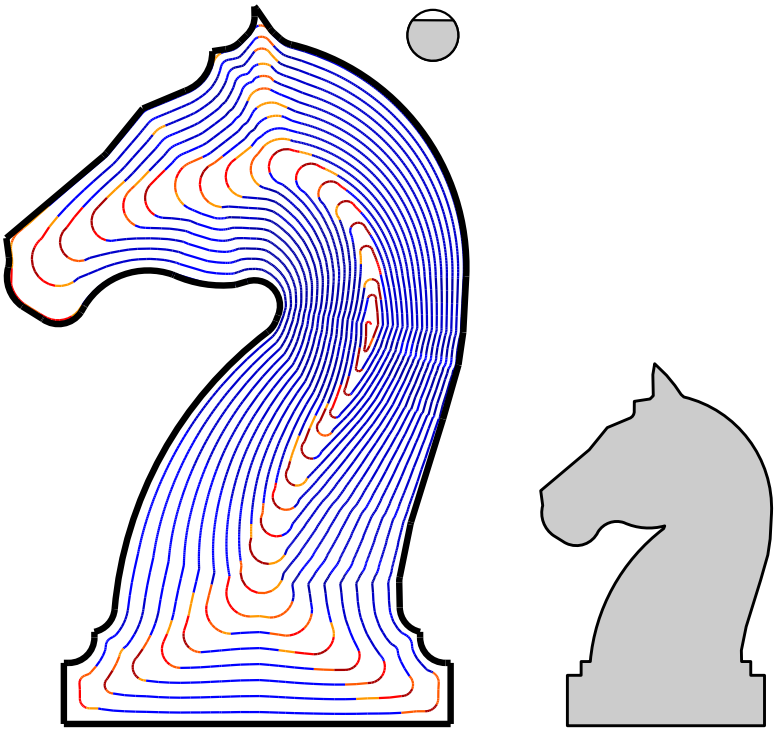
(b)

Figure A.7.: In (a) a decomposition of a complex pocket together with the corresponding tool paths (derived from the dynamic boundary) is illustrated. The maximum step-over δ was set to 80% of the cutter diameter. Furthermore, the decomposition factor η is chosen to be 0.5. The circle (partially) shaded in gray indicates the size of the cutter relative to the pocket. The shape of the resulting cavity is outlined in (b).

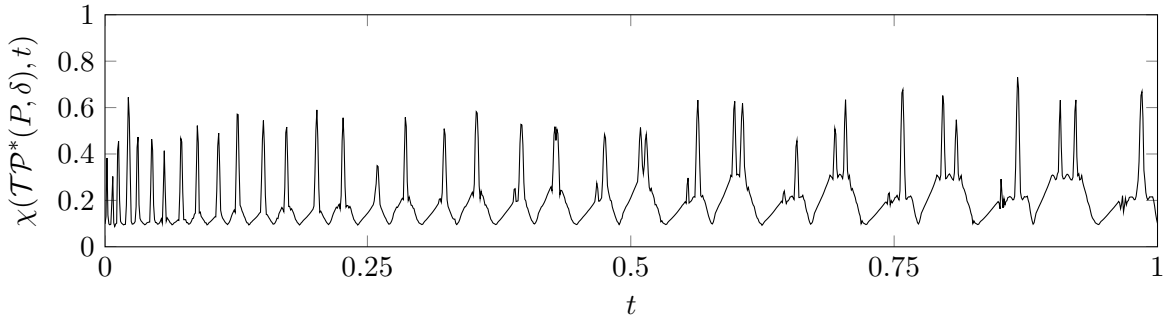


(a) (b)

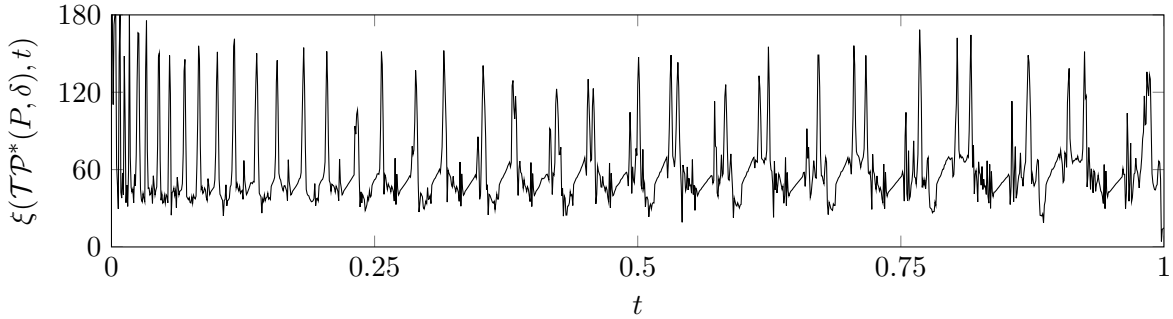
Figure A.8.: In (a) a decomposition of a complex pocket together with the corresponding tool paths (derived from the dynamic boundary) is illustrated. The maximum step-over δ was set to 80% of the cutter diameter. Furthermore, the decomposition factor η is chosen to be 0.5. The circle (partially) shaded in gray indicates the size of the cutter relative to the pocket. The shape of the resulting cavity is outlined in (b).



(a) (b)

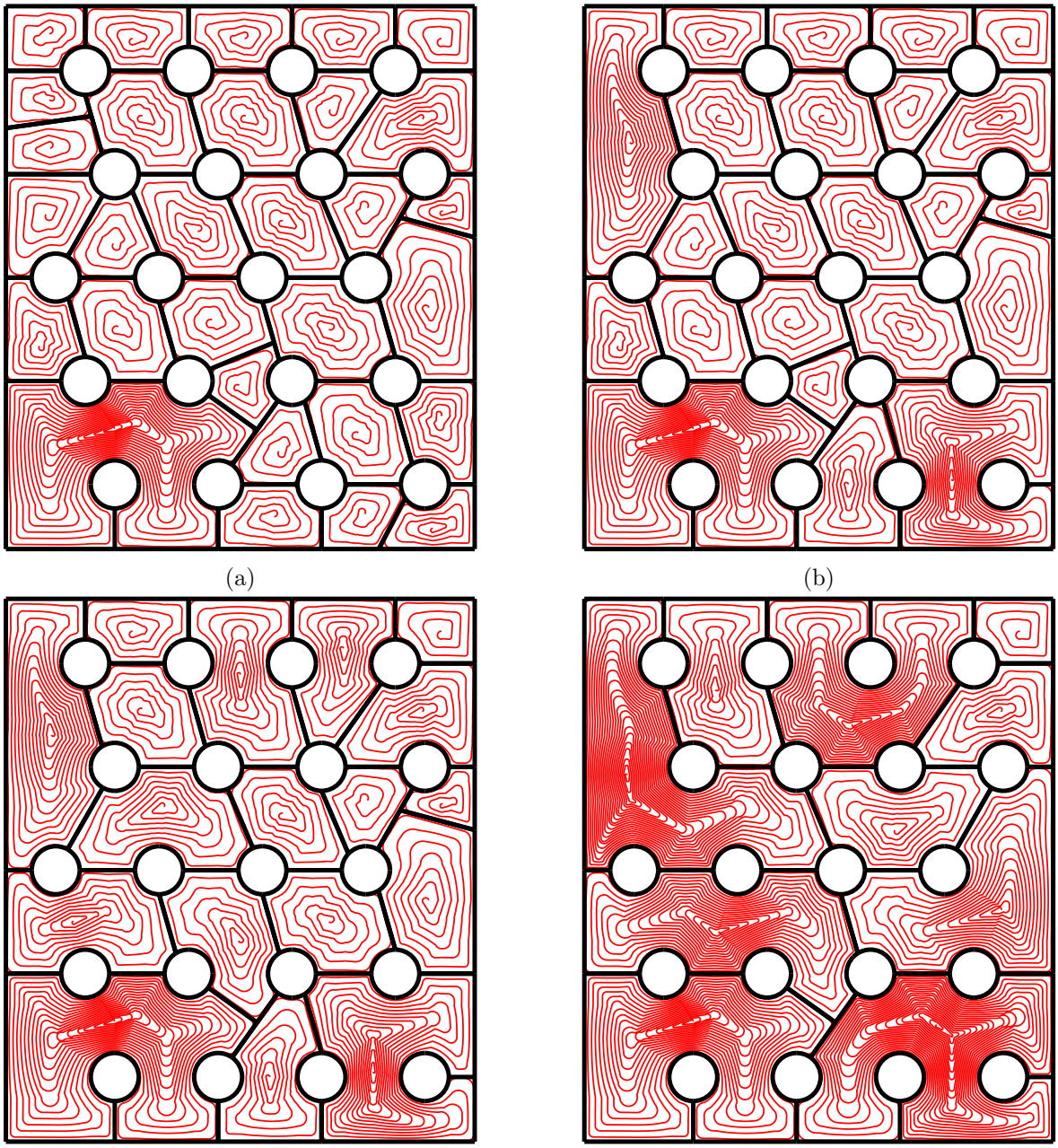


(c) The step-over along the entire tool path.



(d) The engagement angle along the cutter trajectory.

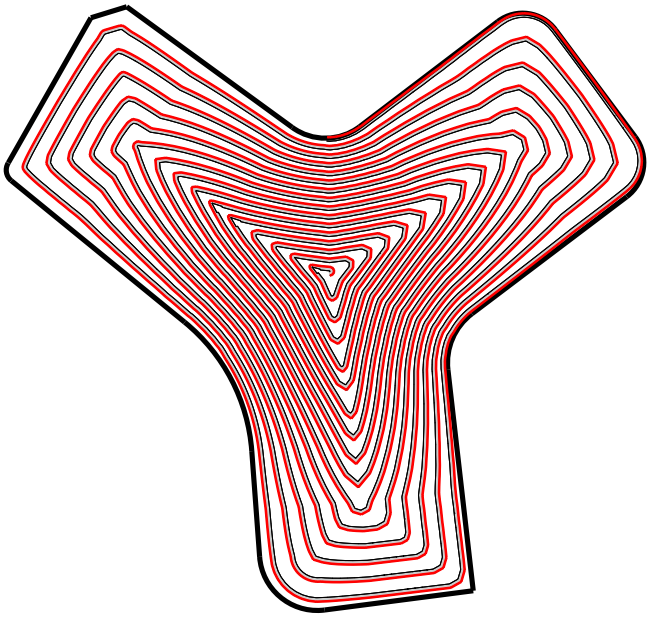
Figure A.9.: Analysis of a tool path $\mathcal{TP}^*(P,\delta)$ inside a simple pocket, where δ equals 80% of the cutter diameter. Locations of high (low) cutter engagement along the tool path are highlighted in red (blue). The circle (partially) shaded in gray indicates the size of the cutter relative to the pocket.



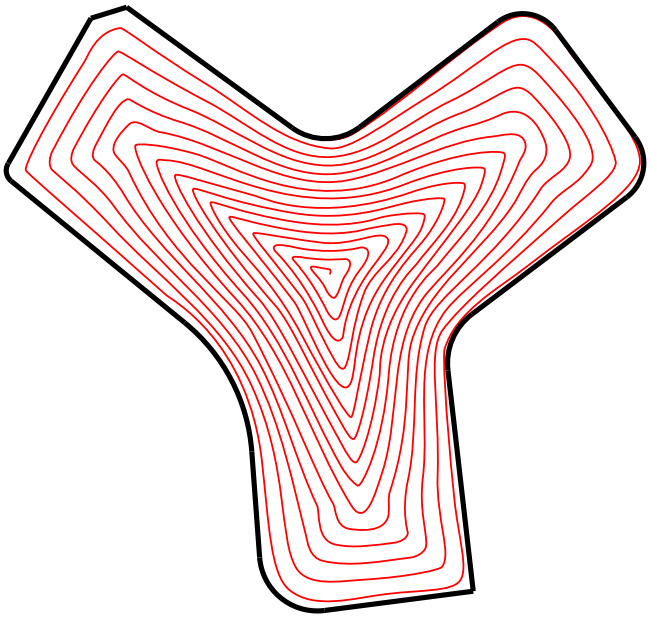
(a) (b)

(c) (d)

Figure A.10.: The figures (a)-(d) show the result of increasing decomposition factors, where decomposition factor η is chosen to be $0.2, 0.3, 0.4$, and 0.5. The maximum step-over δ is set to 80% of the cutter diameter.



(a)



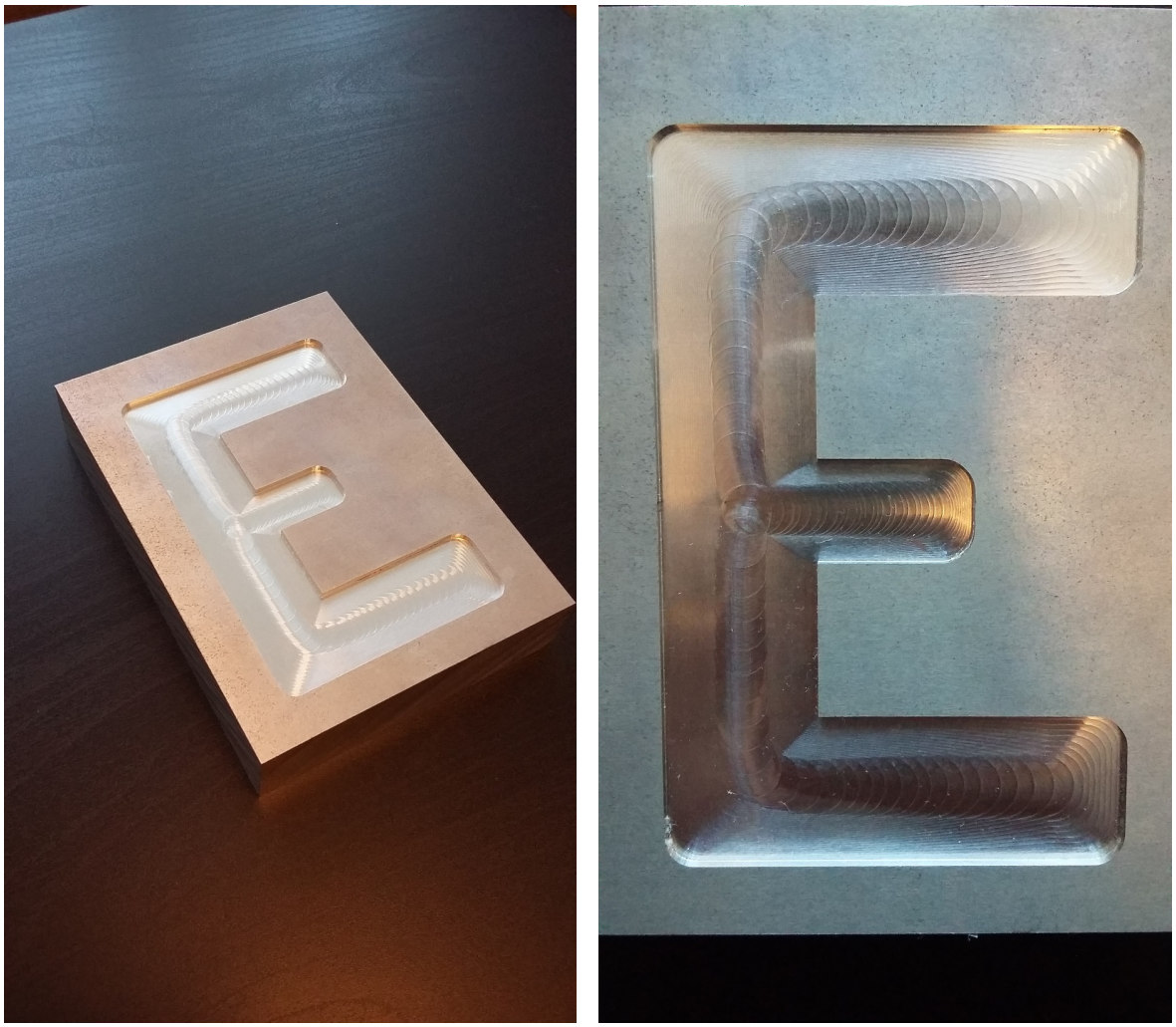
(b)

Figure A.11.: In (a) a tolerance band derived from a tool path $\mathcal{TP}(P, \delta^*)$ is shown, where δ^* equals 90% of the cutter diameter. The sub-figure (b) shows the final approximation of $\mathcal{TP}(P, \delta^*)$ by a cubic Bézier curve.

B. Workpieces

(a) (b)

Figure B.1.: Two views of a part machined in aluminum are shown.



(a) (b)

Figure B.2.: Two views of a part machined in aluminum are shown.

Acronyms

NC numerical control

HSM high speed machining

PDE partial differential equation

FEM finite element method

rpm revolutions per minute

in/min inches per minute

mm/min millimeters per minute

SLS smoothing line segment

SCA smoothing circular arc

REP smoothing repair arc

CCW counter-clockwise

EMST Euclidean minimum spanning tree

BMAT balanced medial axis tree

CCL center of clearance lines

Bibliography

M. Abrahamsen. Spiral toolpaths for high-speed machining of 2D pockets with or without islands. In *ASME 2015 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, pages V02BT03A017–V02BT03A017. American Society of Mechanical Engineers, 2015.

A. Banerjee, H.-Y. Feng, and E. V. Bordatchev. Process planning for floor machining of $2^1/_2$D pockets based on a morphed spiral tool path pattern. *Computers & Industrial Engineering*, 63(4):971–979, 2012.

M. B. Bieterman and D. R. Sandstrom. A curvilinear tool-path method for pocket machining. In *ASME 2002 International Mechanical Engineering Congress and Exposition*, pages 149–158. American Society of Mechanical Engineers, 2002.

H. Blum. A transformation for extracting new descriptors of shape. In W. Wathen-Dunn, editor, *Models for the Perception of Speech and Visual Form*, pages 362–380. MIT Press, Cambridge, 1967.

K. Bolton. Biarc curves. *Computer-Aided Design*, 7(2):89–92, 1975.

J.-J. Chuang and D. C. Yang. A Laplace-based spiral contouring method for general pocket machining. In *ASME 2004 International Mechanical Engineering Congress and Exposition*, pages 227–236. American Society of Mechanical Engineers, 2004.

D. Dragomatz and S. Mann. A classified bibliography of literature on NC milling path generation. *Computer-Aided Design*, 29(3):239–247, 1997.

G. Elber, E. Cohen, and S. Drake. Mathsm: Medial axis transform toward high speed machining of pockets. *Computer-Aided Design*, 37(2):241–250, 2005.

S. J. Farlow. *Partial differential equations for scientists and engineers*. Courier Corporation, 2012.

M. Heimlich and M. Held. 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, 2008.

M. Held. *On the computational geometry of pocket machining*, volume 500. Springer LNCS, 1991.

M. Held. Geometric modeling, 2016. URL `https://www.cosy.sbg.ac.at/~held/teaching/geom_mod/geom_mod.html`.

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

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

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

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

W. Kline, R. DeVor, and J. Lindberg. The prediction of cutting forces in end milling with application to cornering cuts. *International Journal of Machine Tool Design and Research*, 22(1):7–22, 1982.

D. S. Meek and D. J. Walton. Approximation of discrete data by G1 arc splines. *Computer-Aided Design*, 24(6):301–306, 1992.

M. Otkur and I. Lazoglu. Trochoidal milling. *International Journal of Machine Tools and Manufacture*, 47(9):1324–1332, 2007.

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

R. C. Prim. Shortest connection networks and some generalizations. *Bell System Technical Journal*, 36(6):1389–1401, 1957.

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

Z. Zhao, C. Wang, H. Zhou, and Z. Qin. Pocketing toolpath optimization for sharp corners. *Journal of materials processing technology*, 192:175–180, 2007.

B. Zhou, J. Zhao, and L. Li. CNC double spiral tool-path generation based on parametric surface mapping. *Computer-Aided Design*, 67:87–106, 2015.