FSP Report No. 105

**Medial Axis Computation using a Hierarchical Spline Approximation of the Signed Distance Function**

B. Jüttler, A. Poteaux, Xinghua Song

July 2010
Medial Axis Computation using a Hierarchical Spline Approximation of the Signed Distance Function

Xinghua Song¹, Bert Jüttler², Adrien Poteaux²
¹Projet GALAAD, INRIA Sophia Antipolis, France,
²Institute of Applied Geometry, Johannes Kepler University of Linz, Austria

Abstract
We present a new method for computing the medial axis of a given closed smooth curve or surface. It is based on an algorithm for approximating the signed distance function using polynomial splines over hierarchical T-meshes, which has recently been presented in [36]. Since the signed distance function is not differentiable along the medial axis, the hierarchical T-meshes is automatically refined along the medial axis. Based on this observation, we generate an approximation of the medial axis by applying suitable thinning, cleaning and smoothing algorithms to the voxels in the deepest level of the hierarchical T-mesh. Finally, the accuracy of the result is validated by analyzing the Hausdorff distance between the original boundary surface and the boundary of the volume which is represented by the approximate medial axis transform.

Keywords: signed distance function, hierarchical T-spline, medial axis computation

1. Introduction
Blum introduced the medial axis transform (MAT) as a concept for efficient shape description [5]. The MAT is defined as the set of all centers of maximum inscribed balls of a domain (which form the medial axis) along with the distance of each point on the medial axis to the boundary of the shape (i.e., the radius of the ball). The medial axis was soon recognized as a very important concept and is now being used in several different contexts, such as global shape interrogation and representation, finite element mesh generation or offset curve and surface trimming, see e.g. [4, 16, 25, 29, 30, 37, 38].

Several algorithms for computing the medial axis of a given shape are described in the rich literature of this topic. In the literature on Computational Geometry and Geometric Computing, they are mostly based on two approaches.

The first approach exploits the intimate relationship between the medial axis and the Voronoi diagram. Based on it, an approximation of the medial axis is computed as a certain subset of the Voronoi diagram of a set of sample points on the domain boundary [2, 6, 32, 34]. As an application, Amenta et al. [2] describe the Powercrust algorithm for 3D surface reconstruction with guarantees, which is based on the medial axis transform. If the medial axis is to be computed from the Voronoi diagram of sample points, then additional pruning are required in order to identify the most stable part of the medial axis, see [3] and the references cited therein.

Instead of using point samples, the second approach works directly with the given boundary representation, which is assumed to consist of certain special primitives (e.g. line segments or circular arcs), and computes the exact medial axis from it. Note that – even for domains with piecewise linear boundaries – the computation of the exact medial axis is a challenging topic [13]. Consequently, the existing methods deal mostly with piecewise linear or circular boundaries [1, 22, 30]. Several simplified variants of the medial axes and methods for computing it have also been described in the literature, e.g. [18, 27].

Several authors focus on the computation of bisectors and medial axes for domains with smooth curved boundaries [12, 31]. The main idea in these approaches revolves around tracing the bisector of pairs of boundary entities and trimming the bisector to obtain the correct topology of the MAT. Choi et al. [10] present a framework based on the domain decomposition method. Using this paradigm, one may break up the complicated domain into smaller and simpler pieces in order to compute the medial axis. In the case of spline curves, the computation of Voronoi diagrams (which are closely related to the medial axis) can be re-formulated as the problem of solving a multivariate polynomial systems, and efficient numerical techniques for solving it are available [19, 20, 33].
A survey about the stability and computation of the medial axis is given by Attali et al. in [3]. Another survey [11] is dedicated to the properties of the medial axis.

The present paper proposes an entirely different approach to medial axis computation, which is based on the signed distance function of a closed curve or surface $\Gamma$ in the plane or in three-dimensional space. This function assigns to any point $x$ the shortest distance between $x$ and $\Gamma$, with a positive sign if $x$ is inside $\Gamma$ and a negative one otherwise. It is a highly useful representation in a large number of problems in geometric computing. The signed distance function is also closely related to the concept of level set methods [28]. Kimmel and Bruckstein [23] used level sets for shape offsetting in the plane, in particular for computing trimmed offsets.

A new method to compute a hierarchical approximation of the signed distance function was presented recently in [36]. We interpolate the signed distance function of the boundary of the given shape by using a spline function defined over a hierarchical T-mesh.

In the present paper, we use the deepest level of the hierarchical T-mesh (computed for the signed distance function approximation) as an initial approximation of the medial axis. We then improve the dual mesh of this pixel/voxel set by successive thinning and smoothing algorithms, in order to get our final approximation of the medial axis of the given shape. Finally, in order to quantify the accuracy of the computation, we analyze the Hausdorff distance between the given shape boundary and the boundary of the (perturbed) domain which is associated with the approximation of the medial axis.

2. Preliminaries

We briefly recall earlier results from [36] concerning the approximation of the signed distance functions using spline functions over hierarchical grids.

2.1. PHT-splines

Spline functions over T-meshes have recently been analyzed in [15]. Our application is based on results for the special case of $C^1$-smooth bi- and tricubic polynomial splines over hierarchical T-meshes, which is summarized in this section. We will call these spline functions PHT-splines for short.

A two-dimensional $T$-mesh is a partition of an axis-aligned box (e.g., the unit square) into smaller axis-aligned boxes. The edges of the boxes form a rectangular grid that may possess T-junctions. If a grid point of a two-dimensional T-mesh is a crossing vertex (i.e., it possesses valency 4), or belongs to the boundary of the domain, then we call it a base vertex.

A three-dimensional T-mesh extends this concept to the three-dimensional space. It is a partition of an axis-aligned box such that each cell is another axis-aligned box. A vertex of a 3D T-mesh is called a base vertex if it either

- belongs to a boundary edge,
- is a crossing vertex on a boundary facet, or
- possesses valency 6.

In this paper, we consider only hierarchical meshes. More precisely, we assume that the T-mesh has been obtained by repeatedly applying a splitting step (which subdivides a box into two smaller boxes) to the original axis-aligned box representing the entire domain of the spline functions and to the boxes obtained by subdividing it.

For a given hierarchical T-mesh $\mathcal{T}$, a $C^1$-smooth bi- or tricubic PHT-spline $f(x)$ is a function which is a bicubic or tricubic polynomial within each cell of $\mathcal{T}$ for the two- and three-dimensional case, respectively, such that the collection of these polynomial segments forms a globally $C^1$-smooth function. The space of these functions admits a simple local construction, as follows.

At each base vertex, we specify the value, the first derivatives, the mixed second derivative(s), and (in the 3D case) the mixed third derivative of the function. Thus, we specify 4 or 8 data at each base vertex. Then, using Hermite interpolation by cubic polynomials, we evaluate these data of $f(x)$ at the other vertices of the hierarchical T-mesh $\mathcal{T}$. For any point $x$ in any cell $C$ of $\mathcal{T}$, the value of $f(x)$ is determined by the value and derivative data at the 4 resp. 8 vertices of the cell $C$.

More precisely, we obtain a bicubic or tricubic polynomial for each cell, which matches the value and derivative data at the vertices. Thus, a PHT-spline defined over a given T-mesh $\mathcal{T}$ can be determined by the value and derivative data at the base vertices of $\mathcal{T}$. This is in agreement with the dimension formulas for PHT-spline functions [14].

2.2. The signed distance function

Consider a closed curve or surface $\Gamma$, which does not possess any self-intersections, in the Euclidean space $\mathbb{R}^n$ ($n = 2, 3$). In the remainder of this paper, we shall refer to $\Gamma$ as a surface and assume that this includes the case of curves. The surface $\Gamma$ divides $\mathbb{R}^n$ into three parts: the interior $\Gamma^+$, the exterior $\Gamma^-$ and $\Gamma$. A point is contained in the interior (or exterior) of $\Gamma$, if a generic line through it possesses an odd (or even) number of intersection points with $\Gamma$ on either side of the point.

The signed distance function (SDF) $d$ of $\Gamma$ is the scalar field

$$d(x) = \begin{cases} ||x - p|| & \text{if } x \in \Gamma^+ \\ -||x - p|| & \text{if } x \in \Gamma^- \\ 0 & \text{if } x \in \Gamma \end{cases}$$

(1)

A survey about the stability and computation of the medial axis is given by Attali et al. in [3]. Another survey [11] is dedicated to the properties of the medial axis.
where \( \mathbf{p} = \mathbf{p}(\mathbf{x}) \) is the closest point of \( \mathbf{x} \) on \( \Gamma \). The partial derivatives of the signed distance function with respect to the coordinates can be computed by differentiating (1). In particular, the derivatives of the closest point are needed, which can be found from the systems of equations which express the conditions for the closest point. See [36] for more details.

In order to be able to evaluate the derivatives of \( d \), we need to assume that the boundary surface \( \Gamma \) is sufficiently smooth. Note that the signed distance function does not inherit this smoothness, due to the presence of several closest points along the medial axis. We assume that the signed distance function in the interior \( \Gamma^+ \) is at least \( C^2 / C^3 \) smooth for \( n = 2 \) and \( n = 3 \), respectively, except for the points on the medial axis. For instance, this is guaranteed if \( \Gamma \) is \( G^2 \) or \( C^3 \) for the curve or surface case, respectively, except for points on convex edges or vertices (which never occur as closest points of points in the interior).

### 2.3. Adaptive SDF fitting

We generate an approximation of the signed distance function \( d \) of a given surface by interpolating the value and derivative data using a PHT-spline. The algorithm consists of three steps.

1. We choose a bounding box of the given surface that is used as the initial T-mesh \( T^0 \). We compute the closest point of every base vertex in \( T^0 \) and evaluate the required derivatives at the vertices. Using this information we generate an initial bicubic (2D) or tricubic (3D) PHT-spline \( f^0(\mathbf{x}) \). We put \( k = 0 \).

2. We compute the fitting errors of the PHT-spline \( f^k(\mathbf{x}) \) over the T-mesh \( T^k \) at level \( k \). If the fitting errors of all cells in \( T^k \) are less than a given threshold \( \epsilon \), or if \( k \) has reached the maximum allowed subdivision level \( k_{\text{max}} \), then we output the result. Otherwise, we subdivide the cells whose fitting errors are greater than \( \epsilon \) to form a new T-mesh \( T^{k+1} \) at level \( k + 1 \).

3. We compute the closest point of every new base vertex in \( T^{k+1} \) and evaluate the required derivatives. This gives us a new PHT-spline \( f^{k+1}(\mathbf{x}) \) over the T-mesh \( T^{k+1} \) at level \( k + 1 \), which approximates the signed distance function better than \( f^k(\mathbf{x}) \). Let \( k \leftarrow k + 1 \) and continue with step 2.

More details of the method – in particular concerning the closest point computation (which uses a Newton-type method which is initialized by several seed points) and the fitting error estimation – are described in [36].

### 2.4. Examples

We demonstrate the performance of the algorithm by two examples, see Fig. 1. The figure shows the given curves (top row), the given T-meshes obtained by the adaptive refinement procedure (center row) and the graph surfaces of the signed distance functions (bottom row). The graph surfaces are smooth everywhere, except for the points on the medial axis of the interior and exterior regions \( \Gamma^+ \) and \( \Gamma^- \). The PHT-spline fitting algorithm correctly identifies these regions and refines the T-mesh accordingly.

The computation time for the two examples was in the order of less than 10 seconds on a standard PC, see Section 5.2 for details. Once the signed distance function has been approximated, the trimmed offsets of the domain boundary can easily be approximated by its level sets, see [36] for examples.

### 3. Medial axis computation

The adaptive PHT spline approximation correctly identifies the regions with non-smooth signed distance function, which are located along the medial axis. Thus, we propose to use the cells which are obtained in the deepest level of subdivision of the the hierarchical T-mesh as an approximation of the medial axis of the given boundary surface \( \Gamma \). We will denote them with \( T^* \).

Clearly, this is a volumetric representation of the medial axis as a set of pixels (in the case of curves) or voxels (in the case of surfaces). In order to represent the medial axis as a collection of polygons or surfaces, we apply a thinning and a smoothing step. These procedures are explained in the following two sections.

#### 3.1. Planar domains

Starting from \( T^* \), we apply the following thinning algorithm to delete all the facets in \( T^* \).

1. **Construction of the dual graph:** We first construct the dual graph \( \hat{T}^* \) of \( T^* \). More precisely, we use the center of each facet of \( T^* \) as a vertex of \( \hat{T}^* \). In order to obtain the edges of \( \hat{T}^* \), we connect the centers of every two adjacent facets of \( T^* \) by a straight line segment.

   Every closed loop in the dual graph which consists of four different edges is called a facet of \( \hat{T}^* \). If an edge of \( \hat{T}^* \) belongs to one and only one facet, then we call it a **boundary edge**; the associated facet is then called a **boundary facet**.

2. **Thinning of the edges of the medial axis:** If there exists a boundary facet \( F_0 \) of \( \hat{T}^* \) which contains a vertex of valency two, then we delete this vertex, its adjacent edges and the facet \( F_0 \), but we keep the other vertices and edges of \( F_0 \). We repeat this step recursively until all facets are deleted or the valencies of all the vertices are greater than two. In the first case, we output \( \hat{T}^* \). Otherwise, we continue with step 3.
3. Thinning of the nodes of the medial axis: We delete an arbitrary facet of $\hat{\mathcal{T}}^*$ and one arbitrary edge of this facet, but keep all the other edges and vertices. Then, we return to step 2.

After this thinning process, $\hat{\mathcal{T}}^*$ consists of line segments and does not contain any facet. Nevertheless, there may exist some additional short branches (topological noise) in $\hat{\mathcal{T}}^*$ which we do not want to keep. In order to eliminate them, we apply a cleaning procedure.

We identify two special classes of vertices in $\hat{\mathcal{T}}^*$, the character vertices and boundary vertices which possess a valency greater than two and less than two, respectively. The character vertices represent the nodes of the medial axis, and the boundary vertices represent points on the branches of the approximate medial axis.

If no character vertex in $\hat{\mathcal{T}}^*$ exists, then the approximate medial axis is a single open polygon and it is already the final result. Otherwise, for every boundary vertex, we find a unique path $r$ which connects this vertex with the nearest character vertex. If the length of this path $r$ is less than a given threshold, then we delete all the edges and vertices on this path except for the character vertex.

We take the final dual graph $\hat{\mathcal{T}}^*$, which has been modified by applying the thinning and cleaning algorithm, as an approximation of the medial axis of the given curve. Clearly, the edges of this graph are still aligned with the axes of the coordinate system. In order to obtain a smoother representation, one may apply an additional fairing step to these polygons.

3.2. Planar examples

We demonstrate the performance of our algorithm in the planar case by several examples. We used the smallest boxes in the T-meshes shown in the second row of Fig. 1 as initial meshes for computing an approximate medial axis. The results are shown in Fig. 2. As demonstrated by these examples, the methods produces a good approximation of the medial axis, even if corners of the domain boundary are present.

All computations were performed using a PIV-2.2GHz PC with 2.0GB RAM. The computation times are reported in Table 1. In the planar case, the time needed for processing the dual graph is negligible, since the entire computation time is dominated by PHT spline approximation of the signed distance function.

It is well known that the medial axis of a planar domain can be approximated by a subset of edges of the Voronoi diagram of a set of sample points on the boundary [7].
Table 1: Times needed for computing the signed distance field approximation $T_f$ and the medial axis approximation $T_r$ for the two examples in Figures 1 and 2 (in seconds).

<table>
<thead>
<tr>
<th>Example</th>
<th>$T_f$ (s)</th>
<th>$T_r$ (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Heart</td>
<td>3.609</td>
<td>0.01</td>
</tr>
<tr>
<td>Rectangle</td>
<td>4.638</td>
<td>0.01</td>
</tr>
</tbody>
</table>

In particular, one may consider the edges which do not intersect the boundary. Figure 3 compares the result of our method (where the set of sample points was used to initialize the closest point computations) with the Voronoi diagram-based approach. Clearly, our method gives a far more accurate approximation of the medial axis. However, one may observe that our method slightly exaggerates the leaves of the medial axis (which correspond to local maxima of the curvature). This effect is due to the large variation of the gradient of the signed distance function in the vicinity of the leaves.

The final example is a B-spline curve representing the boundary of a leaf of a phoenix tree. The computation time for this example is 42 seconds. The same example has also been considered by Cao and Liu in [9] using a different approach, where the computation (though on different hardware) took approx. 10 minutes.

3.3. Closed surfaces

In the case of closed surfaces, the situation is more subtle than that in the planar case, since the medial axis may possess components of different dimensions. In the generic case, it consists of sheets (parts of bisector surfaces), seams (segments of trisector curves) and junctions, where the sheets meet in seams, and the seams meet in vertices (see e.g. [35, section 3] for more details). However, in special cases, such as objects having rotational symmetries, one may also encounter one-dimensional components without sheets intersecting in them.

In order to obtain a medial axis approximation from the set of smallest boxes, we use the following (partially heuristic-based) combination of thinning, cleaning and smoothing algorithms (see [8, 26] for similar algorithms):

1. Construction of the dual graph: As in the 2D case, we construct the dual graph $\hat{T}^*$ of $T^*$. The center of each cell of $T^*$ defines a vertex of $\hat{T}^*$. We connect the centers of every two adjacent cells of $T^*$ by a straight line to form the edges of $\hat{T}^*$.

   We also construct the facets and cubes of $\hat{T}^*$ by traversing the vertices of $\hat{T}^*$ with the help of the topology of the edges. If an edge of $\hat{T}^*$ has only one neighboring facet, then we call it a boundary edge, and the associated facet is called a boundary facet; a cube of $\hat{T}^*$ which has a boundary facet is called a boundary cube.

2. Thinning of the sheets of the medial axis: If the valency of one vertex of a boundary cube $C_0$ of $\hat{T}^*$ is three, then we delete it, its adjacent edges and facets, and the cube $C_0$, but keep the other vertices, edges and facets of $C_0$. We repeat this step recursively until all the cubes are deleted or the valencies of all the vertices are greater than three.

   In the first case, we output $\hat{T}^*$. Otherwise, we consider two cases: If there is a cube of $\hat{T}^*$ contains an edge $E_1$ with two adjacent facets, then we go to step 3; otherwise, we go to step 4.

3. Edge-based boundary cube deletion: Let $C_1$ be an arbitrary cube of $\hat{T}^*$ such that it contains an edge $E_1$ with two adjacent facets. We delete $E_1$, its two adjacent facets and the cube $C_1$ but keep all the other facets, edges and vertices of $C_1$. Then, we return to step 2.

4. Face-based boundary cube deletion: Let $C_2$ be an arbitrary cube of $\hat{T}^*$ which contains a facet $F_2$ with
only one adjacent cube. We delete $F_2$ and $C_2$, but keep all the other facets, edges and vertices of $C_2$. We then return to step 2.

The thinned dual graph $\hat{T}^*$ constructed by the previous algorithm may contain some unwanted branches (topological noise) and is generally not smooth, since all edges are aligned with the coordinate axes. We apply additional cleaning and smoothing steps to $\hat{T}^*$, as follows.

An edge of $\hat{T}^*$ which has more than 2 neighboring facets is called a non-manifold edge. After the thinning step, the dual graph consists of one or several two-dimensional components (collections of faces of the dual graph), which meet in non-manifold edges. Some of these components may be small and can be considered as topological noise. We use a cleaning procedure to eliminate these components.

This cleaning procedure is based on the distance between the boundary edges to the nearest non-manifold edges. More precisely, for every boundary facet $F_b$ of $\hat{T}^*$, we first find the nearest non-manifold edge $E_{nm}$ along the direction from any boundary edge of $F_b$ to its opposite edge. The length of the path which connects $F_b$ with $E_{nm}$ is denoted by $D(F_b)$.

Given a positive integer $\tau$, we delete the extra facets of $\hat{T}^*$ as follows:

1. Identification of non-manifold edges: We first mark the non-manifold edges of $\hat{T}^*$.
2. Deletion of faces with three boundary edges: We select an arbitrary facet $F_0$ of $\hat{T}^*$, which has 3 boundary edges, and compute the distance $D(F_0)$ of $F_0$ to the non-manifold edges. If $D(F_0) \leq \tau$, we delete $F_0$ from $\hat{T}^*$. We repeat this step until the distances of all the remaining facets which have three boundary edges are greater than $\tau$.
3. Deletion of faces with two boundary edges: Let $F_1$ be an arbitrary facet with 2 boundary edges; we compute the distance $D(F_1)$ of $F_1$ to the non-manifold edges. If $D(F_1) \leq \tau$, we delete $F_1$ from $\hat{T}^*$. We repeat this step until the distances of all the remaining facets which have 2 boundary edges are greater than $\tau$.
4. Deletion of faces with one boundary edge: If a facet $F_2$ has 1 boundary edge, we compute the distance $D(F_2)$ of $F_2$ to the non-manifold edges. If $D(F_2) \leq \tau$, we delete $F_2$ from $\hat{T}^*$. We repeat this step until the distances of all the remaining facets which have one boundary edge are greater than $\tau$.

Nevertheless, even after this step, the edges of the thinned and cleaned dual graph are still aligned with the axes of the coordinate system. In order to obtain a more faithful representation of the medial axis, we apply local fairing (based on a simple discretization of the Laplace operator [24]) to the vertices of the dual graph.

The mesh $\hat{T}^*$ obtained after applying the thinning, cleaning and smoothing algorithms is used as the approximation of the medial axis of the given closed surface.

3.4. Surface examples

We present several computed examples.

In order to validate the output of our method, we applied it to an axis-aligned box, where the medial axis is known. It consists of 13 sheets, 12 seams and four junctions, where all sheets are planar and all seams are straight line segments.
We used an implicit representation of the box boundary and used our algorithm to compute the PHT spline approximation of the signed distance function and then -based on it - the approximate medial axis. The results are shown in the left column of Figure 5. One may see that our approach correctly reproduces the structure and the geometry of the medial axis in this case.

As a second and third example, we applied the medial axis computation procedure to the double torus model, see the middle column of Fig. 5, and to the “dolphin” model, see the right column of Fig. 5.

Fig. 6 presents the medial axis obtained for two more complex objects. In order to apply our method to these data, which were taken from the AIM@SHAPE repository, we first approximated them by implicitly defined tensor-product B-spline surfaces using the method described in [21]. Even though the results of this approximation are not optimal, due to the presence of sharp features, the medial axis computation produces fairly reasonable results.

<table>
<thead>
<tr>
<th>Example</th>
<th>$T_f$</th>
<th>$T_c$</th>
<th># facets</th>
</tr>
</thead>
<tbody>
<tr>
<td>Box</td>
<td>81.53</td>
<td>95.6</td>
<td>119,800</td>
</tr>
<tr>
<td>“Dolphin”</td>
<td>49.58</td>
<td>30.1</td>
<td>62,168</td>
</tr>
<tr>
<td>Bunny</td>
<td>717</td>
<td>377</td>
<td>164,210</td>
</tr>
<tr>
<td>Fandisk</td>
<td>338</td>
<td>271</td>
<td>132,032</td>
</tr>
</tbody>
</table>

Finally, Table 2 reports the computation times (in seconds) which were needed for generating the PHT spline approximation of the medial axis ($T_f$) and for the extraction of the medial axis approximation from the set of smallest boxes in the T-mesh ($T_c$). Unlike the case of curves, the latter time now makes a significant contribution to the total computation time, due to the more complex nature of the geometry. The last column gives the number of facets of the initial dual mesh (before thinning, cleaning and smoothing).

<table>
<thead>
<tr>
<th>Example</th>
<th>$T_f$</th>
<th>$T_c$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Double-Torus</td>
<td>81.53</td>
<td>95.6</td>
</tr>
<tr>
<td>“Dolphin”</td>
<td>49.58</td>
<td>30.1</td>
</tr>
<tr>
<td>Bunny</td>
<td>717</td>
<td>377</td>
</tr>
<tr>
<td>Fandisk</td>
<td>338</td>
<td>271</td>
</tr>
</tbody>
</table>

Table 2: Times needed for computing the signed distance field approximation $T_f$ and the medial axis approximation $T_c$ for the five surface examples in Figures 5 and 6 (in seconds).

4. Error estimation

We estimate the fitting error of the approximate medial axis by analyzing the Hausdorff distance between the surface $Z(D)$ (see below), which is determined by the approximate medial axis $\hat{\mathcal{F}}^*$, and the given boundary surface $\Gamma$. Thus, we consider the approximate medial axis generated by our algorithm as the medial axis of a perturbed domain with boundary $Z(D)$. In order to quantify the accuracy of our computation, we analyze the distance between $\Gamma$ and the perturbed domain boundary.

4.1. Estimating the Hausdorff distance

More precisely, the approximate medial axis transform of $\Gamma$ is obtained by adding the radius information to the approximate medial axis $\hat{\mathcal{F}}^*$ obtained after thinning, cleaning and smoothing,

$\mathcal{M} = \{ (c, r) \in \mathbb{R}^d \mid c \in \hat{\mathcal{F}}^*, r = d(c) \}$.

It defines an associated signed distance function $D(x)$ via

$D(x) = \max \{ \hat{D}(c, r)(x) \mid (c, r) \in \mathcal{M} \}$,

where

$\hat{D}(c, r)(x) = r - \| x - c \|$ is the signed distance function of a ball with center $c$ and radius $r$. In practice, we evaluate $\mathcal{M}$ only at a finite number of sample points. We obtain a discrete approximation

$D^*(x) = \max \{ \hat{D}(c, r)(x) \mid (c, r) \in \mathcal{M}^* \}$

of $D(x)$, where $\mathcal{M}^*$ is the finite subset of $\mathcal{M}$ that is obtained by taking the vertices of the mesh which represents the approximate medial axis $\hat{\mathcal{F}}^*$.

The implicitly defined surface

$Z(D) = \{ x \in \mathbb{R}^3 \mid D(x) = 0 \}$

is an approximation of the given boundary surface $\Gamma$. Recall that the Hausdorff distance is defined as

$d_H(Z(D), \Gamma) = \max \{ \sup_{x \in Z(D)} \inf_{y \in \Gamma} \| x - y \|, \sup_{y \in \Gamma} \inf_{x \in Z(D)} \| x - y \| \}$

where $\Omega$ is a tubular neighborhood of the given boundary surface $\Gamma$, which contains $Z(D)$.

The first term in the second line of Eq. (2) covers the error introduced by using the discrete approximation of $D$. For any point $x \in \Omega$, we have that

$D(x) = r - \| x - c_x \| = d(c_x) - \| x - c_x \|,$

where $c_x \in F$ and $F$ is a facet of $\hat{\mathcal{F}}^*$. Recall that we use the vertices of $F$ in order to define the discrete approximation $D^*$. Let $c_0$ be the vertex of $F$ which is nearest to $c$. Then

$|D(x) - D^*(x)| = D(x) - D^*(x) \leq d(c_x) - \| x - c_x \| - (d(c_0) - \| x - c_0 \|)$

(3)

$\leq d(c_x) - d(c_0) + \| c_x - c_0 \|.$

We now denote the closest points of $c_x$ and $c_0$ on $\Gamma$ with $w$ and $v$, respectively. Then

$d(c_x) - d(c_0) = \| c_x - w \| - \| c_0 - v \| \leq \| c_x - c_0 \|.$

(4)
Summing up, we arrive at

$$|D(x) - D^*(x)| \leq 2||c_x - c_0|| \leq \delta,$$

where $\delta$ is the maximum diameter of the facets in the approximate medial axis $\hat{T}^*$. Now we substitute Eq. (5) into Eq. (2), which gives

$$d_H(Z(D), \Gamma) \leq \delta + \sup_{x \in \Omega} |D^*(x) - d(x)|.$$

### 4.2. Examples

We estimate the second term on the right-hand side of (6) by evaluating it at a large number of sample points in $\Omega$. Then, the maximum value of $|D^*(x) - d(x)|$ plus $\delta$ are used as an estimate of the fitting error of the approximate medial axis.

The estimated fitting errors of the approximate medial axis are shown in Tables 3 and 4. The estimated errors are given in relation to the diameter of the bounding box.

### 4.3. The inverse problem viewpoint

The computation of the MAT from the boundary of a domain is a typical example of an ill-posed problem, cf. [17]. Such a problem has the form $G(u) = d$, where the unknown object $u$ (in our case the medial axis and the radius function) is to be computed from (possibly noisy) observations / data $d$ (in our case signed distance function which is associated with the boundary surface of the two- or three-dimensional domain). The data and the unknowns are linked by an operator $G$ (in our case the operator which generates the boundary surface as the envelope of the maximum inscribed balls of the domain).

It is well known that small variations of $d$ may lead to large changes of $u$. In the framework of inverse problems, this problem is addressed by computing an approximate solution satisfying $G(U) = D$ for perturbed data, $\|D - d\| = \delta$. The approximate medial axis generated by our
algorithm fits into this framework, and $\delta$ is the Hausdorff distance between the given domain and its approximation.

As a matter of future research we will investigate regularization techniques for medial axis computation. In the framework of inverse problems, an “optimal” approximate solution is found by solving a regularized problem of the form

$$\|G(U) - D\|^2 + \lambda R(U) \to \min,$$

where $R$ is a (problem-dependent) regularization term and $\lambda$ is the (noise-dependent) regularization parameter. By increasing $\lambda$, the solution becomes more regular, but simultaneously the error $\delta = \|G(U) - D\|$ increases. Clearly, the norm $\|\cdot\|$ which is used here should be problem-specific.

5. Conclusion

Based on a new method to fit the signed distance function of a given boundary curve of surface by using a simple but efficient PHT-spline (bi- and tricubic piecewise polynomial functions over hierarchical T-meshes) Hermite interpolation method, which was presented in [36], we show how to construct an approximation of the medial axis of the given domain: starting from the deepest level of the hierarchical T-mesh that we obtain in our fitting algorithm, we apply a thinning, cleaning and smoothing algorithm in order to get the medial axis for the given boundary.

The accuracy of the result has been quantified by analyzing the Hausdorff distance between the given domain boundary and the domain which is represented by the approximate medial axis. Future research should be devoted to the enhancement of the resulting piecewise linear approximation of the medial axis, to more sophisticated termination criteria for the hierarchical spline approximation of the signed distance function, and to the investigation of regularization techniques.

Acknowledgments. The author acknowledge the support of the Austrian Science Fund (NFF S92 Industrial Geometry) and of the European Union (PITN-GA-2008-214584 SAGA). The work of Xinghua Song was partially supported by NSF of China (No.60873109).

References


