Distance Functions and Skeletal Representations of Rigid and Non-Rigid Planar Shapes*

Ata A. Eftekharian and Horea T. Ilieş†
Department of Mechanical Engineering
University of Connecticut

May 29, 2009

Abstract

Shape skeletons are fundamental concepts for describing the shape of geometric objects, and have found a variety of applications in a number of areas where geometry plays an important role. Two types of skeletons commonly used in geometric computations are the straight skeleton of a (linear) polygon, and the medial axis of a bounded set of points in the k-dimensional Euclidean space. However, exact computation of these skeletons of even fairly simple planar shapes remains an open problem.

In this paper we propose a novel approach to construct exact or approximate (continuous) distance functions and the associated skeletal representations (a skeleton and the corresponding radius function) for solid 2-dimensional semi-analytic sets that can be either rigid or undergoing topological deformations. Our approach relies on computing constructive representations of shapes with R-functions that operate on real-valued half-spaces as logic operations. We use our approximate distance functions to define a new type of skeleton, i.e, the *C-skeleton*, which is piecewise linear for polygonal domains, generalizes naturally to planar and spatial domains with curved boundaries, and has attractive properties. We also show that the exact distance functions allow us to compute the medial axis of any closed, bounded and regular planar domain. Importantly, our approach can generate the medial axis, the straight skeleton, and the C-skeleton of possibly deformable shapes within the same formulation, extends naturally to 3D, and can be used in a variety of applications such as skeleton-based shape editing and adaptive motion planning.

1 Introduction

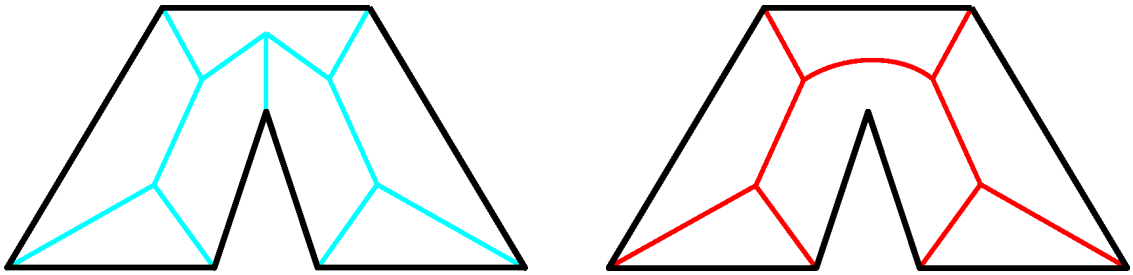
Geometric skeletons are widely used in a variety of applications that deal with shapes and their properties, including engineering design and manufacturing. However, there are multiple ways to define a skeleton of a given set, and a variety of definitions have been proposed for different applications. Two of the most widely used geometric skeletons are the Medial Axis ($\mathcal{MA}$) and the so-called straight skeleton ($\mathcal{SS}$).

The Medial Axis has been introduced by Blum [1] for biological shape measurement and description, and has become an important tool in computational geometry and geometric modeling. The concept of medial axis has been described with the help of the fire grass concept (see for example [2]) as follows: if a fire starts from all points of a planar curve at the same time and moves

*To appear in Computer Aided Design, accepted May 29, 2009.
†Corresponding author.

with constant velocities in all directions in the same plane, then the $\mathcal{MA}$ is the locus of points where the fire (in fact a moving front) meets itself. The concept can be extended to k-dimensional geometric shapes in $\mathbb{R}^k$, case in which the $\mathcal{MA}$ becomes a set of dimension $k-1$. Intuitively, the points on the medial axis are equally distant to at least two points of the curve as illustrated in Figure 1(b). Since its formulation, the medial axis has been used as an alternative solid modeling representation [3], as well as in many other applications such as shape matching and reconstruction [4, 5, 6, 7, 8, 9], dimensional reduction in boundary value problems [10], representing and classifying 2D shapes [11, 12, 13], human vision [14], pattern analysis and shape recognition [15, 16], and mesh generation [17, 18, 19].

A concept that is closely related to the $\mathcal{MA}$ of a point set is that of a straight skeleton, which has been defined in [20] for polygons. In 2D, the straight skeleton of a polygon consists of line segments obtained from trimming the bisectors of the polygon edges, which can lead to ambiguous constructions (see [20] for details). The $\mathcal{SS}$ partitions an n-gon into n polygons that contain only one of the edges of the original polygon. The straight skeleton and medial axis are identical for convex polygons, but are different for non-convex polygons. Straight skeletons have been used as folding directions in origami related problems in computational geometry [21], surface interpolation between polygonal curves [22], image processing [23] or shape manipulation [24].



(a) (b)

Figure 1: Straight skeleton (a) and the Medial Axis $\mathcal{MA}$ (b) of a planar polygon.

1.1 Defining the Medial Axis and Straight Skeleton

As implied above, there is no agreed upon definition of the medial axis in the literature. In this paper we use the following definition of the $\mathcal{MA}$ if Ω is a closed, bounded, regular and semi-analytic subset of $\mathbb{R}^k$, then the medial axis $\mathcal{MA}(\Omega)$ is the set of points that have at least two closest points in $\partial\Omega$ – the boundary of Ω. Thus, $\mathcal{MA}$ is the set of all centers of the closed maximal balls, and is a continuous version of the usual Voronoi diagram.

One can construct examples in which Ω is bounded and its boundary is C^∞-smooth yet the $\mathcal{MA}$ has infinitely many branches [2]. However, the finiteness of the medial axis has been proven in [25] for semianalytic bounded open sets $\Omega \in \mathbb{R}^k$. In the same paper [25], it is proven that the medial axis of semianalytic sets admits stratification, i.e., it can be decomposed into finite number of strata, each a connected i-manifold with boundary for $i < k$. This is particularly important since a wide variety of shapes encountered in science and engineering, including all solid models, can be described as semi-analytic sets [26]. Another important result described in [27] states that a shape and its medial axis are connected the same way regardless of the dimension of the space (i.e., they are homotopy equivalent), which is exploited in many applications, including those that focus on shape similarity.

The straight skeleton has not been defined using a distance function, but through a process of "skinning" a polygon such that all edges move with the same speed in a "self-parallel" manner, and each vertex moves along the bisector of its incident edges. Thus, the straight skeleton $\mathcal{SS}$ is defined as the union of the bisector segments that are traced by the polygon vertices as the polygon shrinks, and has a lower combinatorial complexity than that of the medial axis for non-convex polygons [20]. Despite of that, computing straight skeletons of polygonal shapes appears to be slower than $\mathcal{MA}$ computations for which linear time algorithms are known [28].

Importantly, both the medial axis and the straight skeleton are uniquely defined for a given shape, and induce a disjoint partitioning of the shape.

1.2 Prior Art in Computing these Skeletons

There is a massive body of work related to the mathematical and computational properties of the medial axis, and good surveys can be found in several well written papers on this topic [2, 25, 29]. Here we will briefly discuss some of the relevant research that is closely related to the focus of this paper.

Although it is possible in principle to compute the medial axis exactly for general semi-algebraic/analytic sets, we do not seem to have any algorithms for doing so. The most advanced algorithms make significant simplifying assumptions on the geometry of these sets by focusing on planar or piecewise linear shapes. Algebraic planar curve segments whose bisectors admit rational parametrizations are examined in [30], while exact computation of $\mathcal{MA}$ for polyhedra is described in [31]. The reasons behind these restrictions become apparent once the algebraic difficulties in computing MA are examined [2].

Consequently, these difficulties promoted a variety of algorithms that approximate the complex shape by a simpler set for which the $\mathcal{MA}$ can be computed exactly, followed by the extraction of the $\mathcal{MA}$ of the approximating shape – the so called pruning step. The computed $\mathcal{MA}$ has to be post-processed in order to eliminate the branches that appear in the medial axis due to the shape approximation. The main approaches to approximately compute the medial axis for piecewise linear shapes rely on: Voronoi/Delaunay decompositions of space [32, 33, 34] for piecewise linear approximations of the shape or of planar shapes with curved boundaries [35] where the medial axis is the Voronoi graph; solutions of partial differential equations (such as diffusion or Hamilton-Jacobi equations) [36, 37]; and level set methods [38, 39]. Two recent methods to compute the medial axis for curved planar sets use discrete tracing algorithms along the boundary curves [40] or of the Frenet frames for pairs of boundary curves that bound a closed planar domain [41]. These algorithms do not seem to generalize to 3D space, and their stability appears to be problematic in 2D due to the iterative marching.

A fast method to discretely and approximately compute the generalized Voronoi diagram has been presented in [42], which uses graphics hardware to speed up the computations. For every point of a planar grid, they compute the shortest distance between the point and *all* "sites", which can be points, lines, or polygons. Hence, non-linear sites require linearization. This, in turn, results in a piecewise linear approximation of the distance function. The Voronoi regions and boundaries are then computed based on a simple color buffer algorithm: all points closest to a given site are assigned the same identifying color, and Voronoi boundaries are detected between points that change color. The same 2D algorithm is used to approximate 3D Voronoi diagrams by 2D slices such that each slice is computed independently.

Computing straight skeletons has received much less attention in the literature. Thinning/offseting appears as a natural approach to construct the straight skeleton. However, several published ap-

proaches for computing these skeletons are based on "roofing" – i.e. the principle of roof construction [43, 20, 44], and the fastest algorithms for straight skeleton computation seem to be significantly slower than those for computing $\mathcal{MA}$.

1.3 Scope and Outline

The main goal of this paper is to propose a novel framework for constructing exact or approximate distance functions and the associated skeletons for solid 2-dimensional semi-analytic sets that can be either rigid or undergoing topological deformations. We define our shapes constructively with R-functions that operate on real-valued half-spaces as logic operations, and we show that we can construct skeletons for any planar semi-analytic set with possibly evolving boundaries and undergoing topological changes.

Specifically, we show that our approximate distance functions induce a new type of skeleton – the C-skeleton, which is piecewise linear for polygons, has attractive properties, and generalizes naturally to planar and spatial domains with curved boundaries. Furthermore, we show that the introduction of additional half-spaces in the same formulation provides the *exact* distance function and the medial axis of any semi-analytic planar set as well as the corresponding radius function. In this paper we present the necessary and sufficient additional halfspaces for the case of 2D shapes, and discuss the extensions of this approach to the 3D case. Moreover, we show that our formulation completely avoids performing trimming and pruning of the computed skeletons.

We review in sections 2.1 and 2.2 the main concepts from the theory of R-functions used in this paper, as well as the construction of R-functions for planar semi-analytic sets. The definition of the C-skeleton is given in section 2.3, and the construction of the exact distance function for convex polygons as well as the extraction of the medial axis are presented in section 2.4. The extension to domains with holes, moving or curved boundaries is discussed in sections 2.5 and 2.6, while the topological properties and the sensitivity to noise of the C-skeleton are explored in section 3. Section 4 showcases the capabilities of our approach through several examples. The paper concludes with a summary of the contributions and limitations of this work, and a discussion of how this approach can be extended to 3D domain.

2 From Geometry to Distance Functions to R-Functions and Back

2.1 R-functions as Logic Operators on Real-Valued Half-Spaces

In this section we present the fundamental concepts that we use to develop our framework for computing skeletal representations of a semi-analytic domain Ω. We begin by summarizing the concept of R-functions[1] that we employ to define the approximate or exact distance functions over the given Ω. We then explain the needed ingredients to construct a single implicit function over Ω that represents the distance function to the boundary $\partial\Omega$ of Ω either exactly or approximately.

For any closed subset Ω of $\mathbb{R}^k$, i.e., a set that contains its boundary points, the distance from a point $\mathbf{P} \in \mathbb{R}^k$ to the boundary of Ω is usually defined as

$$d(\mathbf{P}) = \inf_{\mathbf{x}\in\partial\Omega} \|\mathbf{P} - \mathbf{x}\|. \tag{1}$$

Consequently, every continuous function f describes a closed set $f(\mathbf{P}) = 0$ that is a subset of $\mathbb{R}^k$, which further implies that one can construct a C^∞ function that vanishes on the boundary $\partial\Omega$ of

[1] Detailed explanations of key concepts behind the theory of R-functions are found in [26].

Ω. It is known that such shapes are in fact semi-analytic sets of points that can be constructed as a finite Boolean combination of real analytic functions $f_i \geq 0$. In other words, one can construct a C^n function over a semi-analytic subset Ω of $\mathbb{R}^k$ by subdividing the boundary of Ω in primitive halfspaces f_i, followed by a combination of f_i into a single predicate using the standard Boolean logic operators AND, OR or NOT [26].

R-functions have been invented in the 60's by V.L. Rvachev, who called these functions "logically charged functions". These functions provide the means to construct a C^n function over a domain defined by primitive halfspaces. In the context of this paper, the main contribution of the theory of R-functions is to replace these logical operations by real-valued functions, which generates an implicit representation for any semi-analytic set Ω. Specifically, these functions, whose sign is completely determined by the sign of their arguments, and independent of any of their magnitudes, allow us to construct approximate and exact distance functions over a semi-analytic domain (assumed here to be planar).

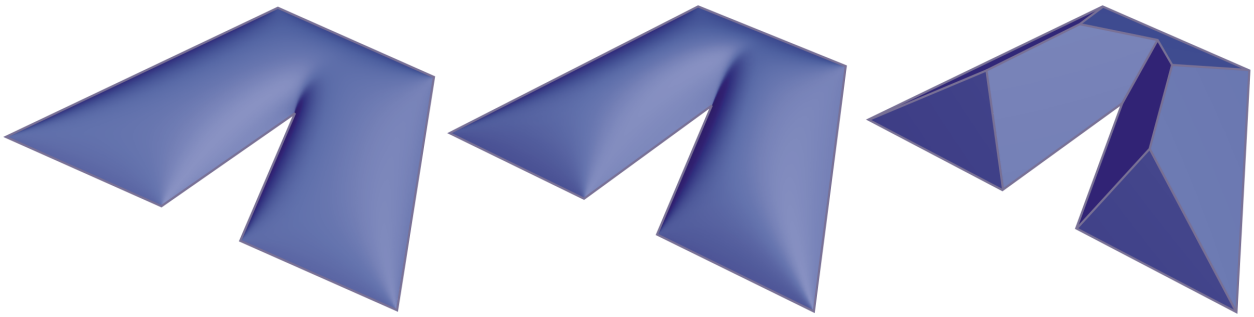


Figure 2: Three implicit representations of the polygonal domain shown in Figure 1 that correspond to (a) $\alpha = 0$; (b) $\alpha = 0.5$; and (c) $\alpha = 1$.

As explained in [26], there are many systems of sufficiently complete R-functions. One such system is known as the $R_1(\Delta)$ system:

$$\begin{array}{rcll} C & \equiv & const & \text{(logical 1) ,} \\ \bar{x} & \equiv & -x & \text{(logical negation } \neg\text{) ,} \\ x_1 \wedge_1 x_2 & \equiv & min(x_1, x_2) & \text{(logical conjunction } \wedge\text{),} \\ x_1 \vee_1 x_2 & \equiv & max(x_1, x_2) & \text{(logical disjunction } \vee\text{),} \end{array}$$

which can be generalized to what is called the principal system of R-functions:

$$R_\alpha(\Delta): \qquad \frac{1}{1+\alpha}\left(x_1 + x_2 \pm \sqrt{x_1^2 + x_2^2 - 2\alpha x_1 x_2},\right) \tag{2}$$

where $(+)$ corresponds to the R-disjunction $(x_1 \wedge_\alpha x_2)$ and $(-)$ to R-conjunction $(x_1 \vee_\alpha x_2)$. By varying the value of α, we obtain different systems of R-functions. In particular, by setting $\alpha = 1$ in equation (2) we obtain the $R_1(\Delta)$ system of R-functions, while a value $\alpha = 0$ in equation (2) would result in the $R_0(\Delta)$ system. Such implicit functions are positive inside the domain[2], negative outside and zero on the boundary – see also Figure 2.

[2]Note that the positive side of any implicit function is subject to convention since $f = 0$ implies $-f = 0$.

Importantly, when expression under the square root in equation (2) becomes zero, the R-functions become non-differentiable. In the next section we show how one can use R-functions to define exact or approximate distance functions for semi-analytic domains. Since the medial axis of a semi-analytic domain corresponds to the points where the distance function is non-differentiable, we will be looking for points of the R-function where the expression under the square root vanishes.

Figure 2 shows the difference between the implicit functions constructed over a polygonal domain with the $R_\alpha(\Delta)$ system for three values of α. One can see that for $\alpha \neq 1$, the resulting functions are differentiable (except at the origin). On the other hand, the implicit function corresponding to $R_1(\Delta)$ shown in Figure 2(c) is made of piecewise planar patches that intersect at piecewise linear edges. In fact, by projecting the edges of the R-function surface on the plane of the polygon we generate a skeleton of the polygon. But how do we construct the function, and which skeleton do we obtain?

2.2 Constructing Boolean Expressions for Polygonal Domains

Conceptually, the problem of constructing a Boolean expression for a domain bounded by halfspaces is the same as that of converting a Boundary Representation (B-rep) into a Constructive Solid Geometry (CSG) representation in Solid Modeling [45]. Many algorithms for performing the B-rep to CSG conversion for polygons are known and efficient [46, 47], but the complexity of the problem quickly escalates with the increase in the complexity of the boundary of the domain. Separating the boundary into primitive pieces is no longer sufficient to construct the Boolean expression of the domain, because additional halfspaces need to be introduced – the so-called separating halfspaces. Determining a sufficient set of separating halfspaces is the critical step of any such algorithm, but the problem is not well understood in general. Solutions exist for planar domains bounded by linear and curved edges that are subsets of convex curves [48], and solids bounded by linear or quadric surfaces [49, 50].

In order to construct the Boolean set representation of a polygonal domain, we have developed a simple algorithm based on the concept of the Convex Deficiency Tree which treats each polygon as its convex hull *minus* a finite number of concavities. Note that the polygon is a closed set, so the subtraction of concavities must necessarily be regularized (i.e., one must use regularized Boolean operations).

Our algorithm uses *three* binary indices for each individual linear halfspace $f_i,\ i = \overline{1,n}$, of the polygonal domain that may have concavity regions (Figure 3). We infer the orientation of each vector defining an edge from the order in which the polygon is prescribed by the user, and we use the cross product while marching in a clock-wise orientation around the polygon boundary to determine whether the current edge is: (1) a concave edge – those edges that are in a concave region of the polygon (edges ②,③ and ④); or a convex edge (edges ①, ⑤ - ⑧). Edges that belong to the last category may or may not be part of the convex hull of the polygon. We differentiate between convex edges that are on the convex hull of the polygon and those that are not. Figure 3 illustrates our indexing procedure for an arbitrary polygonal domain, where we initialize the algorithm with the convex edge ①. Similar algorithms have been thoroughly studied for restricted classes of shapes [47, 48].

We use this edge classification to construct the implicit function (halfspace) corresponding to each edge, such that each halfspace will be positive towards the interior of the polygon. Since we are seeking to establish a distance function for the polygonal domain, the halfspace defined by each linear segment of the polygon's boundary must make an angle of 45^o with the plane containing the polygon. If this condition is satisfied, the value of the function $f_i(\mathbf{x})$ at a point $\mathbf{x} \in \mathbb{E}^2$ will be equal

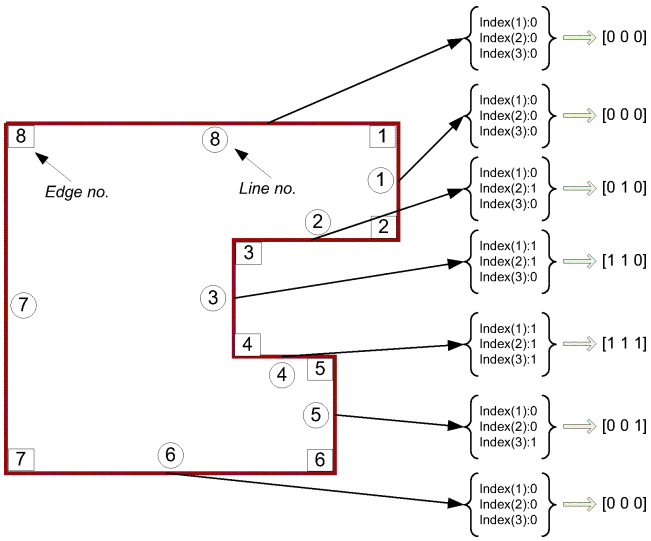


Figure 3: Sample polygon and its edge indices that we use to identify the concavities.

to the shortest Euclidean distance between $\mathbf{x}$ and the line in $\mathbb{E}^2$ that generates the halfspace f_i.

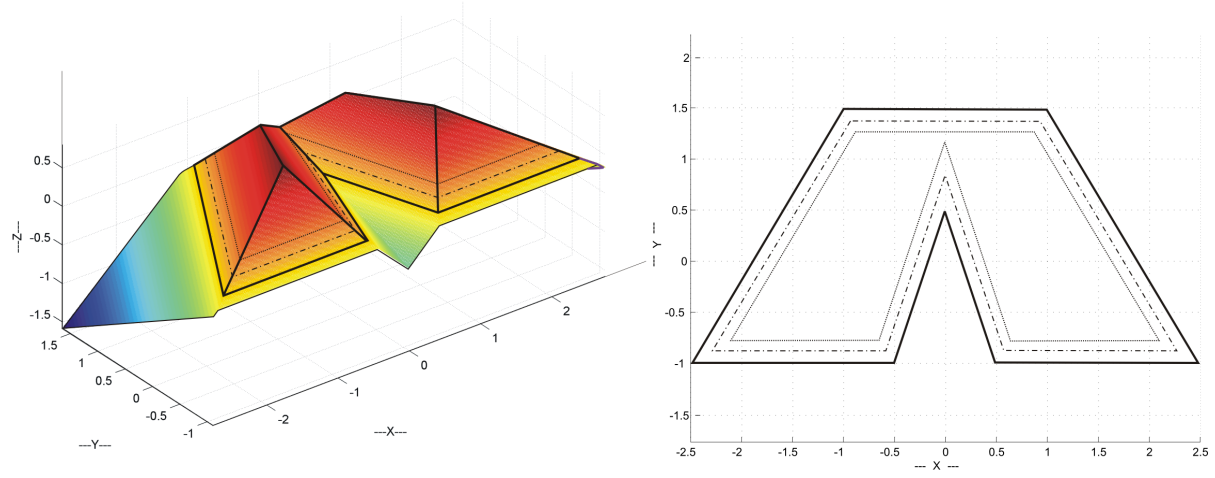
Constructing the Boolean expression defining the R-function for the polygon relies on the indices summarized above to decompose the polygonal domain into polygonal subchains meeting at vertices that are on the convex hull of the polygon. If two halfspaces (or two subchains) of the original polygon meet at a concave vertex they are combined by set union, otherwise they are combined by set intersection. Importantly, this construction can be extended to some other point sets, such as curved polygons [48], 3D polyhedra, and more general 3D solids [51, 50]. Finally, we perform a syntactic substitution to replace the union and intersection with the R-disjunction and R-conjunction given in equation (2), which results in an R-function whose zero level set is the original polygon.

By following this procedure, we obtain an R-function that is the exact distance function for any convex polygon, and an approximate distance function for a concave polygon, as illustrated in Figure 4. Note that such a distance function is obtained from the principal system of R-functions given in equation (2) by setting the value of $\alpha = 1$, i.e., the $R_1(\Delta)$ system. Values of $\alpha < 1$ correspond to R-functions over the same polygonal domain that have established differential properties used in formulating meshfree solutions to boundary value problems [26].

2.3 Ridges and Ravines of the R-function

Observe in Figure 4 that the distance function computed based on the above procedure has a roof-like structure whose edges are convex (ridges) or concave (ravines). The projection of the edges of this R-function on the plane of the polygon is the set of points commonly known as the straight skeleton. Therefore, to compute the straight skeleton, one would need to extract the edges of the R-function and then project them onto the plane of the polygon. There are several approaches that one can employ for the edge extraction step.

One intuitive approach would be to use a solid geometric kernel to construct and combine the halfplanes according to the CSG expression defining the R-function, which would provide the 3D boundary of the solid volume enclosed by the R-function surface and the plane of the polygon. The



(a) (b)

Figure 4: R-function as a distance function for the original convex polygon. The zero level set of the R-function is the polygon itself.

---

advantage of this approach is that one can directly obtain a parametric representation of all the trimmed curve segments comprising the ridges and ravines of the R-function as illustrated in Figure 12. However, this approach cannot be generalized to 3D domains when the R-function exists in a 4D space.

A second approach that is indeed generalizable to 3D domains and 4D R-functions can take advantage of the fact that the resulting $R_1(\Delta)$ R-function is not differentiable at the edges. In order to extract these edges, one can use, the established techniques for edge detection in computer vision that look for intensity changes in images. These techniques rely on the fact an intensity change in an image is accompanied by a large (or peak) first directional derivative, which is equivalent to a zero-crossing in the second directional derivative of the intensity function [52] captured by the Laplace operator. In this paper we used a commonly used finite-difference approximation of the Laplacian [53]

$$\nabla^2 f(x,y) \simeq l_{ij} = f_{i+1,j} + f_{i-1,j} + f_{i,j+1} + f_{i,j-1} - 4f_{i,j}, \tag{3}$$

and a line sweep algorithm to identify points in the interior of the polygon where the R-function is not differentiable.

The straight skeleton contains the projections of both convex and concave edges of the R-function as seen in Figure 5(a). If, instead, we collect only the points obtained from projecting the ridges of the R-function, we obtain a subset of this straight skeleton shown in Figure 5(b) that closely resembles the medial axis of the same domain shown in Figure 5(c). We call this skeleton a *convexized skeleton* or C-skeleton for short. Thus,

Definition 1 *If Ω is a closed, bounded, and regular subset of the Euclidean space $\mathbb{E}^2$, the C-skeleton of Ω is the projection of the convex edges of the R-function onto the plane of the domain.*

There are multiple approaches to detecting the convexity of the edges of the R-function. A good and relatively recent review of the available algorithms for the detection of ridges and valleys in computer vision is presented in [54], and a Laplacian based algorithm for ridge and ravine detection

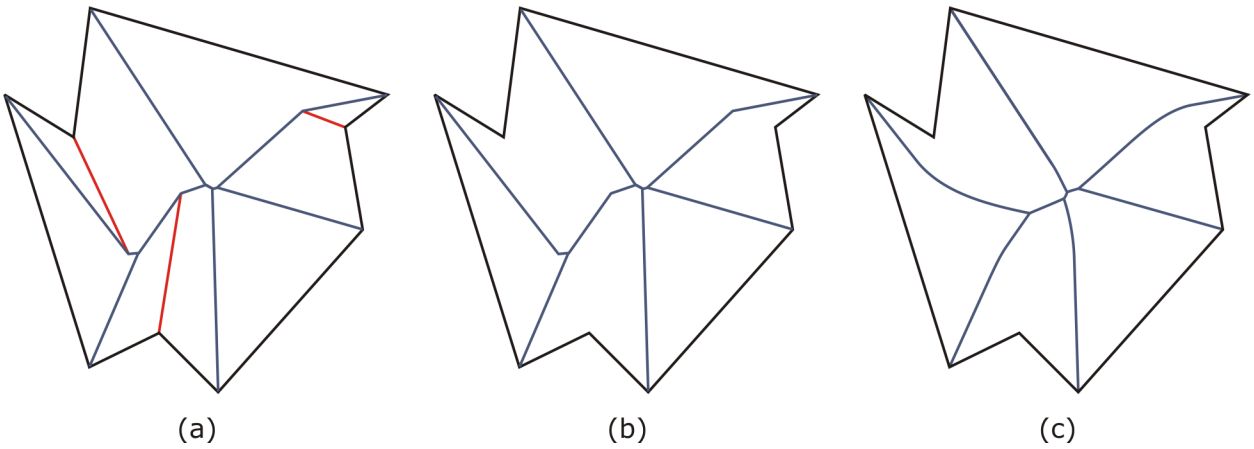


Figure 5: (a) Straight skeleton of a non-convex domain that contains all ridges and ravines of the R-function; (b) its C-skeleton that contains only the convex edges (ridges) of the R-function; (c) the corresponding medial axis of the same domain computed as described in Section 2.4.

---

is discussed in [55]. A straightforward implementation can be achieved if we observe that the discrete Laplacian of equation (3) changes its sign between points that are on ridges of the R-function and points that are on its ravines (see also [55]). Specifically, if we collect all points of the domain for which the discrete Laplacian of equation (3) is non-zero then we obtain the straight skeleton of the domain. On the other hand, by collecting the points of the domain where the discrete Laplacian of the R-function is negative, we obtain its C-skeleton.

One can immediately observe the striking similarity between the C-skeleton and the medial axis of the domain shown in Figure 5: the C-skeleton has the same number of junction (i.e., non-differentiable) points as the straight skeleton of the same domain, and the same number of branches as the medial axis. However, the C-skeleton of polygons is always piecewise linear, so it has a lower geometric complexity than the corresponding medial axis. A preliminary exploration of some of the topological properties of the C-skeleton is described in section 3, and the construction of the R-function for this domain is detailed in appendix 5.

Note that edge detection algorithms that discretize the domain, such as the Laplacian based edge detection discussed above, will inevitably output points of the skeleton. This point cloud requires segmentation and reconstruction of individual curve segments, which can take advantage of a number of commercially available tools and standard CAGD algorithms. While the specific problem of point cloud processing is outside the scope of this paper, it is important to emphasize that our R-function formulation provides an elegant approach to point cloud segmentation. This is so, because the halfspaces generating a given point of the skeleton can be easily identified by simple evaluations of the corresponding implicit functions. On the other hand, implementing the R-function construction in a geometric kernel supporting freeform surfaces as well as standard geometric operations, including surface intersection and projection, will output the medial axis as curve segments[3].

[3] A solid modeling kernel which uses parametric curves and surfaces would output the medial axis as a collection of parametric curves.

2.4 From C-skeleton to Medial Axis

It is not difficult to see that the C-skeleton is identical to the medial axis for convex polygons, but this is not true for arbitrary polygons. Specifically, in the neighborhood of a concave vertex, the medial axis contains curved segments that are subsets of the bisector of a point and a line. This bisector of a point and a line is known to be a parabola which, by definition, is the set of points equally spaced between the point and the line. Farouki and Johnstone showed in [56] that the bisector of a planar rational curve and a point in the same plane is another planar rational curve. Algorithms for computing parametrizations of bisectors encountered for polyhedral domains (which will necessarily be linear or second order curves and surfaces) have been described in [31]. Elber and Kim showed that the bisectors of two rational space curves, of a point and a rational space curve, or of a point and a rational surface are all rational surfaces [57, 58]. More recently, [59] studied algebraic and geometric properties of curve-curve, curve-surface, and surface-surface bisectors, and [60] provided a fast algorithm for computing curve-curve or point-curve bisectors by expressing the bisector of two rational curves as a solution to a linear system.

However, in order to obtain these bisectors in the same space as the original planar domain Ω, we need to add additional halfspaces to the R-function formulation that would generate the correct bisectors. If appropriate additional halfspaces can be *generated*, *trimmed* and *added* automatically to the R-function defining the domain Ω such that the new R-function corresponds to an exact distance function for Ω, then their intersection curves (ridges and ravines of the R-function) in $\mathbb{E}^{k+1}$ Euclidean space will generate the medial axis of Ω by projection onto $\mathbb{E}^k$.

We illustrate in Figure 6(a) a 3D view of the procedure: The parabolic bisector between a point and the line can be obtained by adding at the concave vertex v_i a 3D conical surface c_i whose opening angle is $\frac{\pi}{2}$. By intersecting this conical surface with the distance function to a given line (which is a planar halfspace $f_j > 0$ in 3D) we obtain a conic section. Its projection onto the plane of the polygon is the bisector between the point P and the line $f_j = 0$ as shown in Figure 6(b). This implies that in order to construct the exact distance function of an arbitrary polygonal domain, we need to add an additional conical halfspace c_i at every concave vertex v_i of the polygon. However, a simple addition of these conical halfspaces is not sufficient because the corresponding intersection curves need to be trimmed at points that will generate, by projection, the junction points of the medial axis (see Figure 6(c)). We perform this trimming process by adding two additional planar halfspaces h_i^1 and h_i^2 for each conical halfspace c_i such that the two additional halfspaces are *normal* to the two incident edges e_i and e_{i+1} at vertex v_i, or $h_i^1 \perp e_i$ and $h_i^2 \perp e_{i+1}$. In fact, one can show that the addition of a conical halfspace at every concave vertex v_i of a planar domain (not necessarily polygonal) and of two planar halfspaces that are normal to the two edges incident at v_i are not only necessary, but also sufficient for constructing the exact distance function of the domain bounded by halfspaces f_j.

2.5 Domains with Holes and Moving Boundaries

A planar domain may have one or more holes. One of the main advantages of our approach to compute skeletons based on R-functions is that holes can be easily added to Ω by simply adding (closed) Jordan curves corresponding to the boundary of each hole. Additional conical halfspaces are added for the vertices of the polygon defining the hole by following the procedure described above. The only difference is in where these additional halfspaces are added: a concave vertex of the interior of the hole becomes a convex vertex of the polygonal domain Ω. The R-function expression corresponding to the hole is subtracted from the R-function defining the outer boundary

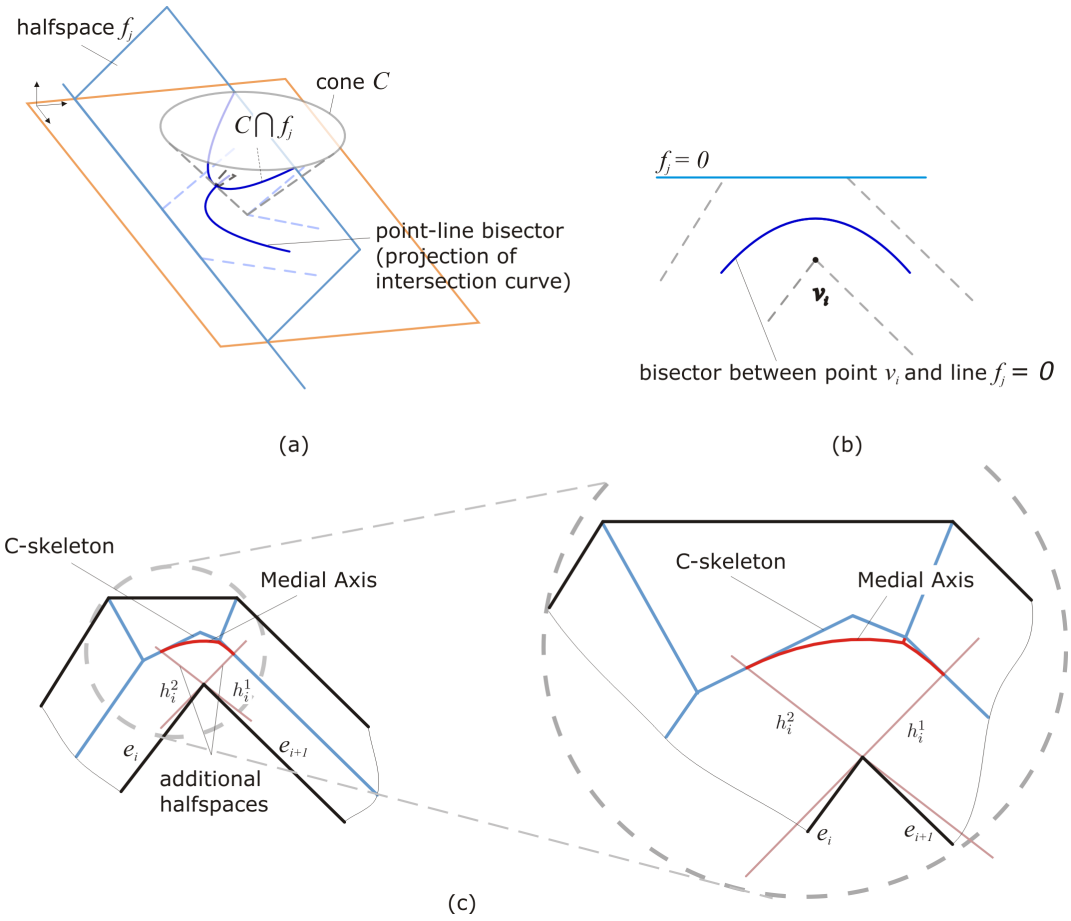


Figure 6: Additional halfspaces are required to construct exact distance functions for concave polygons: (a) the conic section obtained by intersecting the halfspace f_j with the conical halfspace added at the convex vertex v_i; (b) the bisector between a point and a line is obtained by projecting this conic section onto the plane of the domain; (c) trimming the bisector by introducing additional halfspaces h_i^1 and h_i^2 normal to the two incident edges at v_i.

---

of Ω with the R-conjunction (see also section 2.1):

$$f_1 \wedge_1 f_2 = \frac{1}{2}\left(f_1 + f_2 - \sqrt{f_1^2 + f_2^2 - 2f_1 f_2}\right). \tag{4}$$

The connection between R-functions and level sets becomes apparent if one observes that the domain defining the R-function is the zero level set of the R-function itself as shown in Figure 4. Level set methods are known to have the ability to track evolving boundaries with and without topological changes [61], and several optimization problems have been formulated on such higher dimensional surfaces [62, 63]. An important feature of these optimization methods is that holes can be easily relocated to perform parametric optimization [64] or they can be relocated to perform topology optimization [63, 65]. Furthermore, moving boundaries of Ω can be handled as easily, although some deformations may require one to reconsider the additional halfspaces that need to be added at convex vertices that become concave during deformation. One alternative is to add

conical halfspaces at all vertices (both convex and concave), in other words introduce redundancy into the R-function itself, because these extra halfspaces will not play any role in the R-function until the corresponding vertex becomes concave.

2.6 Domains with Curved Boundaries

The two main difficulties in adding curved boundary segments to the polygonal domain are to:

1. construct the distance function corresponding to each curved halfspace;
2. adding additional halfspaces to the formulation that would generate the point-curve bisectors.

The problem of constructing distance functions to given curves and surfaces arises in many applications and has been recently investigated, for example in [66]. This problem can be reformulated as a numerical search for the foot point of a curve or a surface, but the search can be slow, does not always converge and may not be robust [67]. Given these difficulties, we developed a different strategy: rather than searching for the point(s) on a curve that is(are) closest to a given point of the plane in which the curve lies, we

1. enumerate points $\mathbf{P}_c$ on the curve,
2. construct a vector $\mathbf{n_P}$ normal to the curve at each point $\mathbf{P}_c$ on the curve that is coplanar with the curve itself;
3. enumerate points $\mathbf{P}'_c$ along the line defined by the normal vector $\mathbf{n_P}$.
4. set the *known* distance from $\mathbf{P}'_c$ to $\mathbf{P}_c$ as the value of the distance function that we are looking for, that is

$$f(x, y) = d(\mathbf{P}'_c, \mathbf{P}_c)$$

Observe that this procedure: (1) is general, in the sense that it remains valid for any semi-analytic halfspace; (2) it *enumerates* values of the distance function *without requiring a search*; (3) it avoids the problems with the foot-point computations mentioned above; and (4) it is conceptually equivalent to a line that sweeps a halfspace as point $\mathbf{P}_c$ travels along the planar curve. This last point implies that the proposed construction should be followed by a postprocessing step that would eliminate the self-intersections of the sweeping line. Determining such self intersections can be non-trivial in the most general case as discussed in [68, 69], and could significantly impact the performance of the above procedure. Alternatively one could employ visibility tests as in [66] to perform the trimming of the generated points. The final step requires the interpolation of the generated point cloud to obtain a continuous surface approximating the distance function to any desired degree of accuracy.

Figure 7 shows two polygonal domains with one curved boundary segment and their corresponding C-skeletons and medial axes constructed with the procedures outlined above.

It is important to note that adding a conical halfspace c_i at every *concave* vertex v_i and two halfspace h_i^1 and h_i^2 normal to the two incident edges at v_i is sufficient to generate the bisector of a point and any planar curve defined by f_j. One salient feature of our approach is that curve-curve bisectors are embedded automatically in the R-function construction and no additional halfspaces are required.

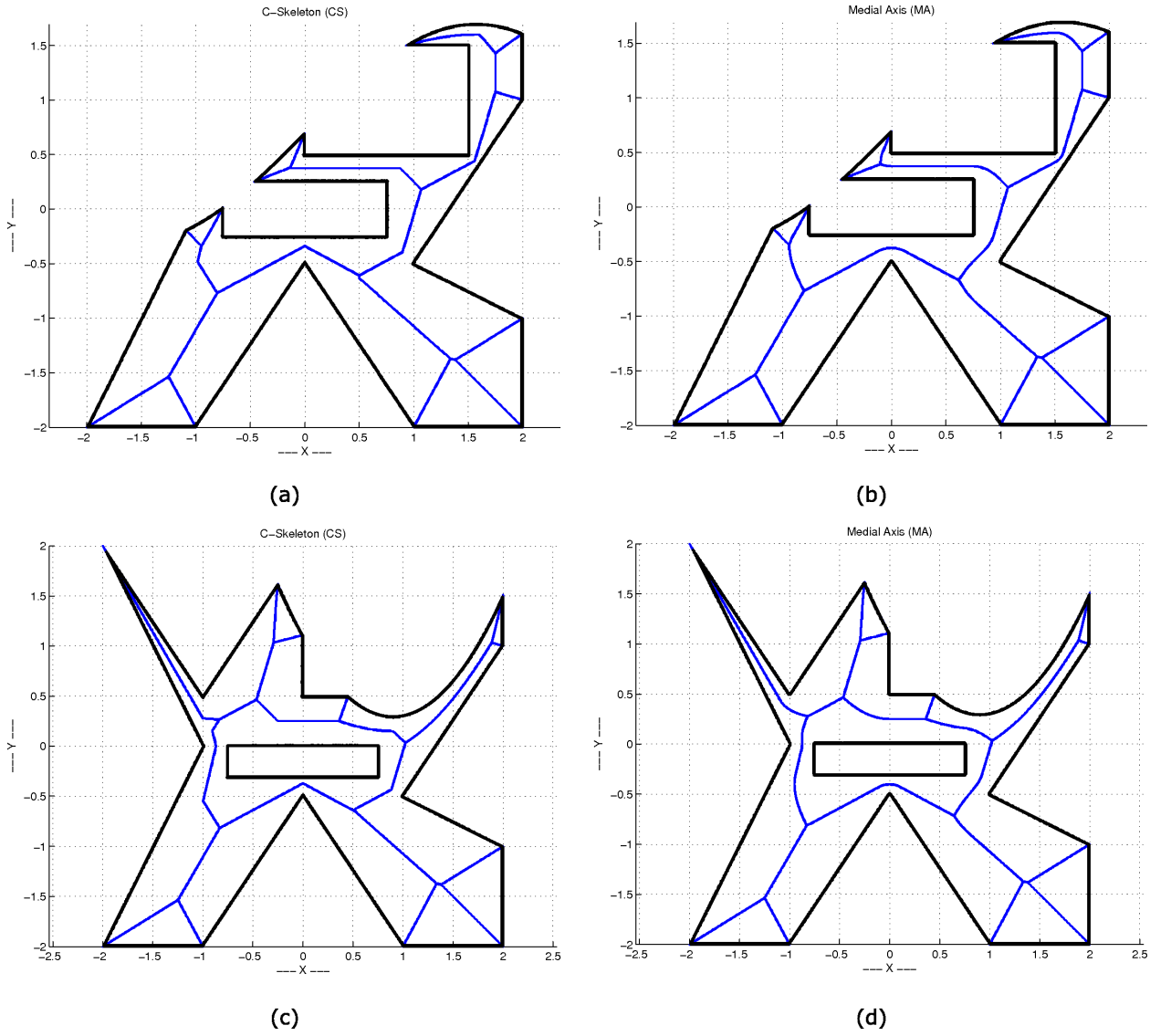


Figure 7: C-skeleton (a) & (c) and the medial axis (b) & (d) for two simple polygons with one curved boundary segment defined by $\sin x + 0.7 - y = 0$ (a) & (b) and $(x - 0.9)^2 + 0.2 - y = 0$ (c) & (d).

---

3 Topology of the C-skeleton and Sensitivity to Noise

It is known that the medial axis has the same connectivity as that of the domain defining the $\mathcal{MA}$ [27]. This however, is not true for a C-skeleton, which can be a disconnected set even if Ω is simply connected. More specifically, the medial axis and the C-skeleton are identical for convex polygonal domains, and therefore the C-skeleton of a convex polygon has the same connectivity as that of its convex domain. However, for non-convex domains the $\mathcal{MA}$ and the C-skeleton are different, and the C-skeleton may or may not be a connected set depending on the geometry and relative positions of the halfspaces defining Ω even if Ω is simply connected. A simple example is shown in Figure 8 illustrating the C-skeleton (a) and the medial axis (b) of a simple non-convex polygon and the

corresponding approximate and exact distance functions. Note that the disconnected C-skeleton becomes a connected medial axis by introducing the additional halfspaces at the concave vertices as described above.

The instability of the medial axis with respect to the Hausdorff distance (i.e., small changes in the boundary of Ω may induce large changes in the MA(Ω)) has been widely studied. This property of the medial axis is the fundamental reason why a pruning step (essentially heuristic) is required when computing the medial axis of a noisy point sample (see [2, 33] for details). One immediate implication of the topological property of the C-skeleton described above is that one could use the disconnected C-skeleton of a noisy point sample to prune the approximated medial axis when the domain Ω is approximated by a finite point set. For the case shown in Figure 8, if the coordinates of the bottom-most point reflect the "noise" in the data, one would prune the vertical segment of the medial axis. However, developing such an algorithm is beyond the scope of this paper.

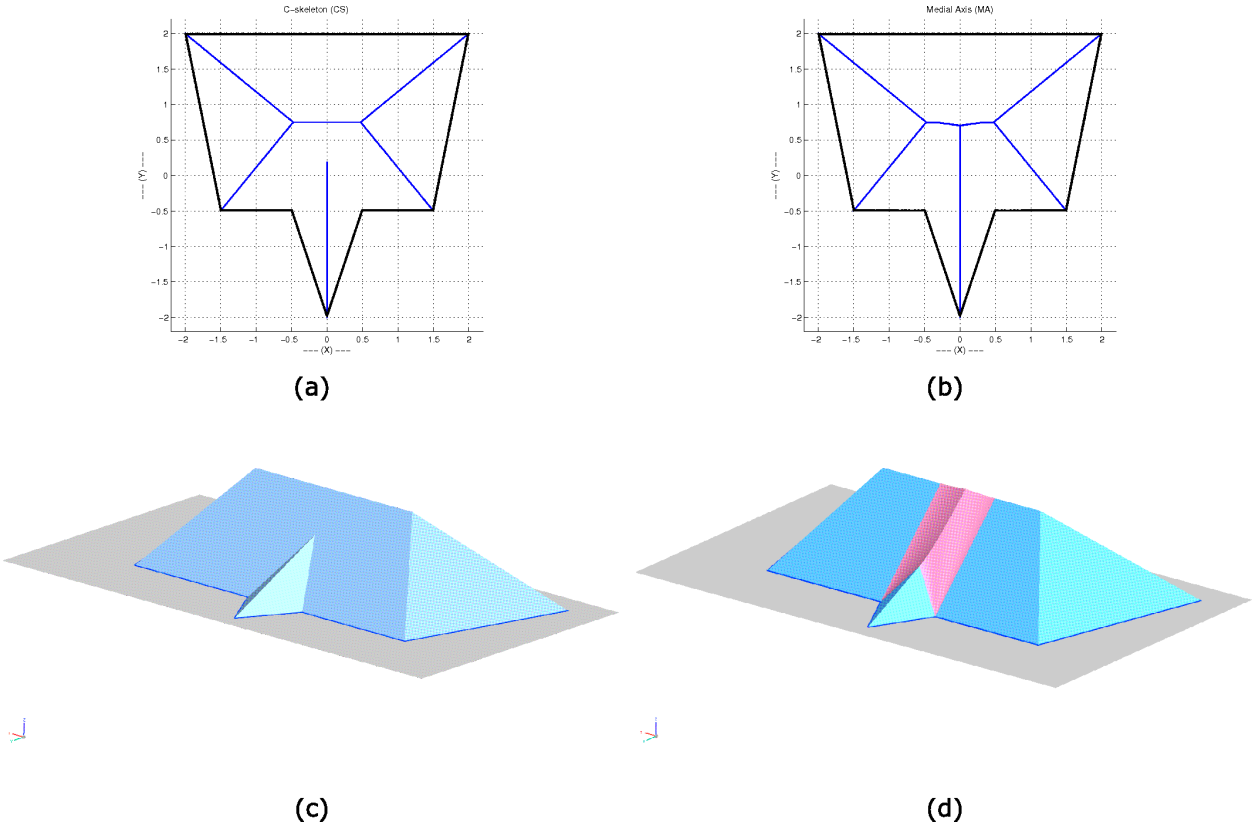


Figure 8: C-skeleton (a) and medial axis (b) of a simple non-convex polygon. The corresponding approximate and exact distance functions are shown in (c) and (d). Note the changes in the distance function generated by the introduction of the additional halfspaces at the two concave vertices of the polygon. These changes transform the (disconnected) C-skeleton into the (connected) medial axis.

4 Examples

In this section we describe several examples that display the capabilities of our approach. As mentioned in section 2.3, one could construct the distance functions within a solid geometric kernel to construct and combine the halfspaces according to the CSG expression defining the R-function. This, in turn, provides the boundary of the solid bounded by the R-function and the plane of the planar domain, as well as a parametric representation of all the trimmed curve segments comprising the ridges and ravines of the R-function. One disadvantage of this approach is that it does not generalize to 3D domains when the R-function exists in a 4D space. Alternatively, one can construct the R-functions programmatically, followed by an edge detection algorithm, and by projection of all or some of these edges onto the space of the domain, as discussed in Section 2.3. Such a numerical procedure can be followed by a fairly simple segmentation of these points to identify the halfspaces that generate a particular point of the skeleton (this segmentation amounts to repeated function evaluations), as well as by a reconstruction step to fit piecewise polynomial curves (and surfaces in 3D) to the segmented point sets. All numerical results provided here have been obtained by implementing the described approach in Matlab, while the last example was constructed in Unigraphics.

The first example shows a non-convex wrench-like polygonal domain with 14 edges and 3 holes. By definition, the C-skeleton of such a domain must only contain straight line segments, but $\mathcal{MA}$ will contain both linear and parabolic curve segments. The Boolean expression is computed in four steps by constructing one R-function for the polygon itself, and three separate R-functions for the three holes. These four R-functions are combined with the appropriate Boolean operators. Finally, ridges are detected and projected to the $x - yY$ plane as described in section 2.3 to obtain the C-skeleton. Converting the C-skeleton into $\mathcal{MA}$ requires the addition of conical halfspaces c_i and trimming halfspaces h_i to the Boolean set expression according to the rules described in Section 2.4. The computed C-skeleton and medial axis are shown in Figure 9.

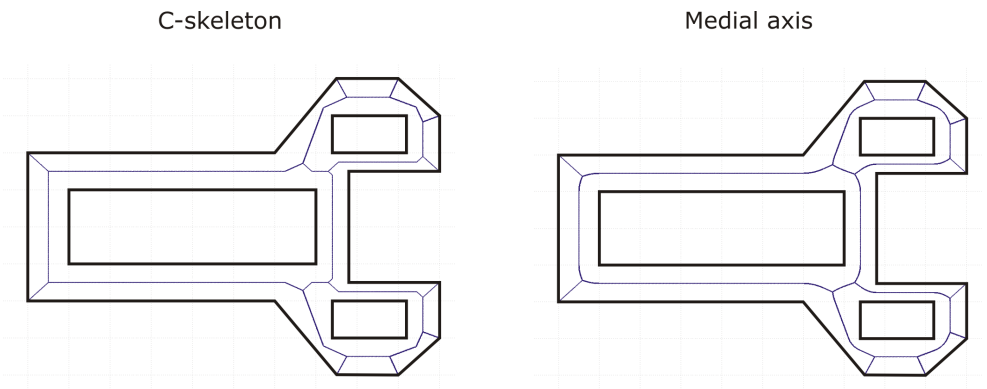


Figure 9: C-skeleton (a) and the medial axis (b) of a planar domain with holes.

In the next example, the domain defined in Figure 9 undergoes topological changes generated by scaling the holes while translating in the positive x direction. Observe that the boundary of the holes will collide with the outer boundary of the polygon and with each other which will generate drastic topological changes as shown in Figure 10. Importantly, the Boolean expression defining the domain does not change, but the C-skeleton and the medial axis *adapts* to the topological

changes. This shows that our approach can track changes in the skeletons induced by changes in the boundary of the domain within the same formulation even when subjected to drastic topological changes. The final domain shown in Figure 10(d) is a non-manifold disconnected planar domain. It is also important to note that each halfspace affects only a subset of the skeleton, and that our approach explicitly provides the correspondence between any particular branch of the skeleton and the halfspaces that generate that particular branch. This can be exploited, in principle, to develop adaptive computations of the changes induced by transformations of the domain boundary that could result in localized computations of the changes in the skeleton shape induced by local modifications in the boundary of the domain.

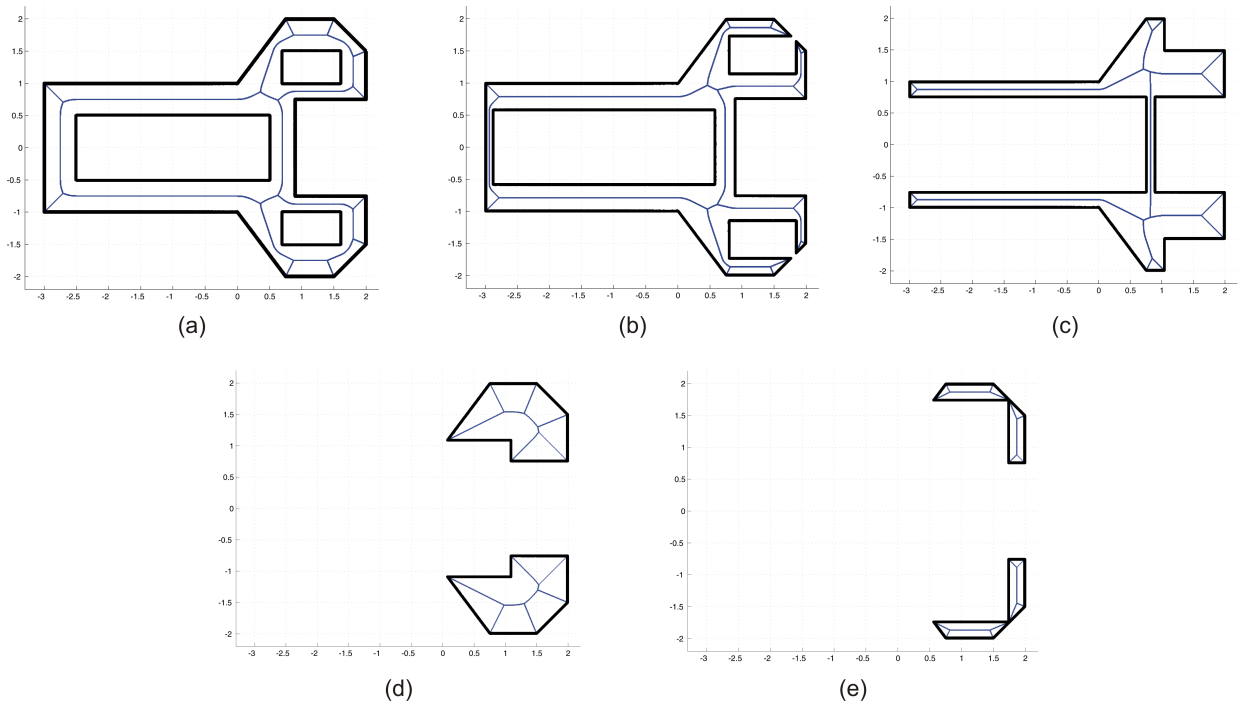


Figure 10: The $\mathcal{MA}$ adapts to topological changes of the boundary of Ω.

The third example illustrates a domain that contains one curved boundary segment and seven holes. Constructing the Boolean set expression for this shape follows essentially the same procedure as the one described above. The only difference is in how the halfspace defined by the curved segment was constructed, which followed the procedure described in section 2.6. The computed C-skeleton and the medial axis of this domain are shown in Figure 11. For this example the curve is defined by

$$ax^3 + by^5 + c = 0 \tag{5}$$

where a, b and c are constants. Clearly the medial axis will contain curved bisecting segments, as shown in Figure 11(a). Unlike previous cases, the curved boundary will introduce curved branches in the C-skeleton. Since the Boolean expression for the C-skeleton will not contain the conical and trimming halfspaces, the corresponding curved segments of the C-skeleton will be different than those of the medial axis.

The final example shown here is a planar domain bounded by piecewise cubic polynomial segments as illustrated in Figure 12(a). The C-skeleton and the corresponding approximate distance

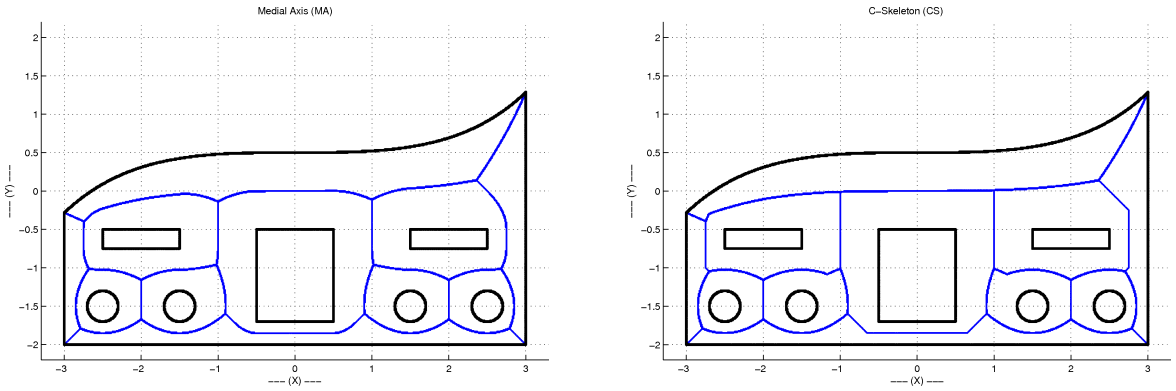


Figure 11: Medial axis and C-skeleton for a sample polygon with one curved segment. The curve is defined by equation $ax^3 + by^5 + c = 0$. The C-skeleton contains both linear and curved branches due to the curved boundary of the domain.

---

function are shown in Figure 12(b) and (d). By introducing a conical halfspace as well as two additional trimming halfspaces h_1 and h_2 (not shown) at the concave vertex, we can construct the exact distance function shown in Figure 12(e) and extract the medial axis as seen in Figure 12(c). Note that, in this example, the approximate and the exact distance functions have been constructed in Unigraphics based on the R-function predicate. The edges of the distance functions have been obtained by intersecting the halfspaces according to this Boolean expression, and are piecewise polynomial curves. Their projection onto the plane of the domain provides the C-skeleton and medial axis of this domain bounded by NURBS curves.

5 Conclusions

We propose a novel and powerful approach to construct either exact or approximate distance functions and the associated skeletal representations for 2-dimensional semi-analytic sets that can be either rigid or undergoing topological deformations. Our distance functions are defined constructively based on R-functions that operate on real-valued half-spaces as logic operations.

In particular, we showed that R-functions can be used to construct approximate distance functions over a domain defined by primitive halfspaces, where the R-functions replace the corresponding logical operations by real-valued functions. In turn, this results in an implicit representation of the approximate distance function for any semi-analytic planar domain. Furthermore, we showed that the constructed approximate distance function corresponds to a new type of skeleton, the C-skeleton, which is piecewise linear for polygons, has attractive properties, and generalizes naturally to planar and spatial domains with curved boundaries. Importantly, the C-skeleton closely resembles the medial axis of a planar domain, but has a lower geometric complexity. Its properties make the C-skeleton very appealing for a variety of applications where medial axis is currently being used.

By adding conical and trimming halfspaces at every concave vertex of the domain, the approximate distance function is transformed into an exact distance function. Furthermore, the medial axis, which corresponds to the ridges of the exact distance function, can be extracted either algebraically – by combining the corresponding halfspaces according to the Boolean expression defined

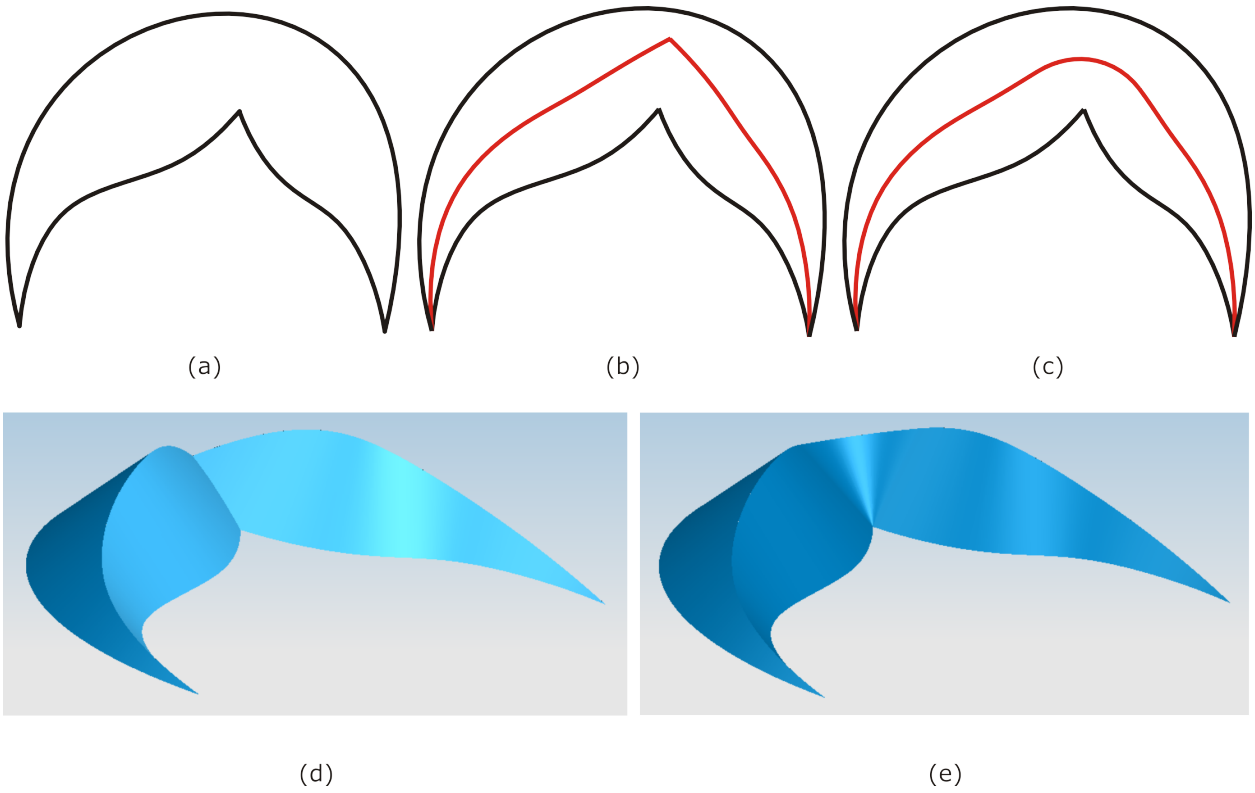


Figure 12: C-skeleton (b) and medial axis (c) of a planar domain bounded by NURBS curves (a). The corresponding approximate and exact distance functions are shown in (d) and (e).

---

by the R-function as illustrated, for example, in Figure 12, or numerically.

Moreover, several other critical features prove the generality and flexibility of our approach. Specifically,

- the proposed C-skeleton is the same as the straight skeleton for convex polygons, but has fewer edges for concave polygons. For piecewise linear shapes, the C-skeleton has the same number of branches as the medial axis, but a lower geometric complexity due to its piecewise linear structure;

- the construction of the exact distance function and medial axis of a planar domain only requires the addition of a conical halfspace as well as of two trimming halfspaces to the Boolean expression defining the approximate distance function for every concave vertex. In fact, if one adds these halfspaces for *every* vertex of the domain (both concave and convex), the halfspaces will automatically contribute to the exact distance function and the corresponding medial axis as long as the corresponding vertex is concave, while these halfspace will have no effect for convex vertices. This is particularly important for domains with evolving boundaries because it eliminates the need to keep track of the convexity of vertices;

- the straight skeleton, C-skeleton and medial axis of a planar domain can be all generated within the *same* formulation. Moreover, due to its Boolean nature, the same formulation can handle both rigid or deformable domains, possibly undergoing drastic topological changes.

This feature becomes particularly useful in those applications where the environment is not fully known *a priori*, such as adaptive motion planning in dynamic environments;

- the C-skeleton can be disconnected even if the domain is simply connected. This important property of the C-skeleton can, in turn, be used to trim the branches of the medial axis that appear due to noise in the data, i.e., the so called *pruning* problem;
- our approach supports multiple representations of the input geometry. In fact, we have proposed a simple and efficient algorithm for constructing an implicit representation of a planar curve (and of its associated halfspace) directly from its parametric representation, which could be a practical alternative to the usual foot-point computations;
- the proposed approach could be implemented in any existing commercial CAD systems and geometric kernel that supports freeform surfaces and standard geometric operations on surfaces.

Finally, our approach extends naturally to 3D. This case would require additional trimmed halfspaces beyond those discussed in this paper, but the generalization of our formulation to 3D appears to be possible because the mathematical foundation remains valid.

Acknowledgments

This work was supported in part by the National Science Foundation grants CMMI-0555937, CAREER award CMMI-0644769, and CMMI-0733107. All 3D examples were created by using Parasolid and Unigraphics, courtesy of Siemens PLM.

Appendix: Constructing the R-function for a Non-convex Polygonal Domain

We revisit here the domain illustrated in Figure 5 and assume that each linear edge is part of a halfspace that represents the exact distance function to the carrier of the line.

The domain shown in Figure 13 has three concave vertices. Since each vertex is shared by two incident edges, we denote by

$$z_1 = (f_2 \cup f_3) \tag{6}$$
$$z_2 = (f_5 \cup f_6) \tag{7}$$
$$z_3 = (f_8 \cup f_9) \tag{8}$$

the halfspaces defined by the incident edges at each of the three concave vertices, whose positive side points towards the interior of the domain. Therefore, the Boolean expression generating the approximate distance function can be written as:

$$f(x, y) = (f_1 \cap f_4 \cap f_7) \cap (z_1 \cap z_2 \cap z_3). \tag{9}$$

The R-function is constructed by performing a syntactic substitution of the Boolean operations with the corresponding R-disjunction and R-conjunction from equation (2), and the resulting function is illustrated in Figure 13(c). The ridges of this approximate distance function are projected onto the plane of the polygon to obtain the C-skeleton shown in Figure 13(a).

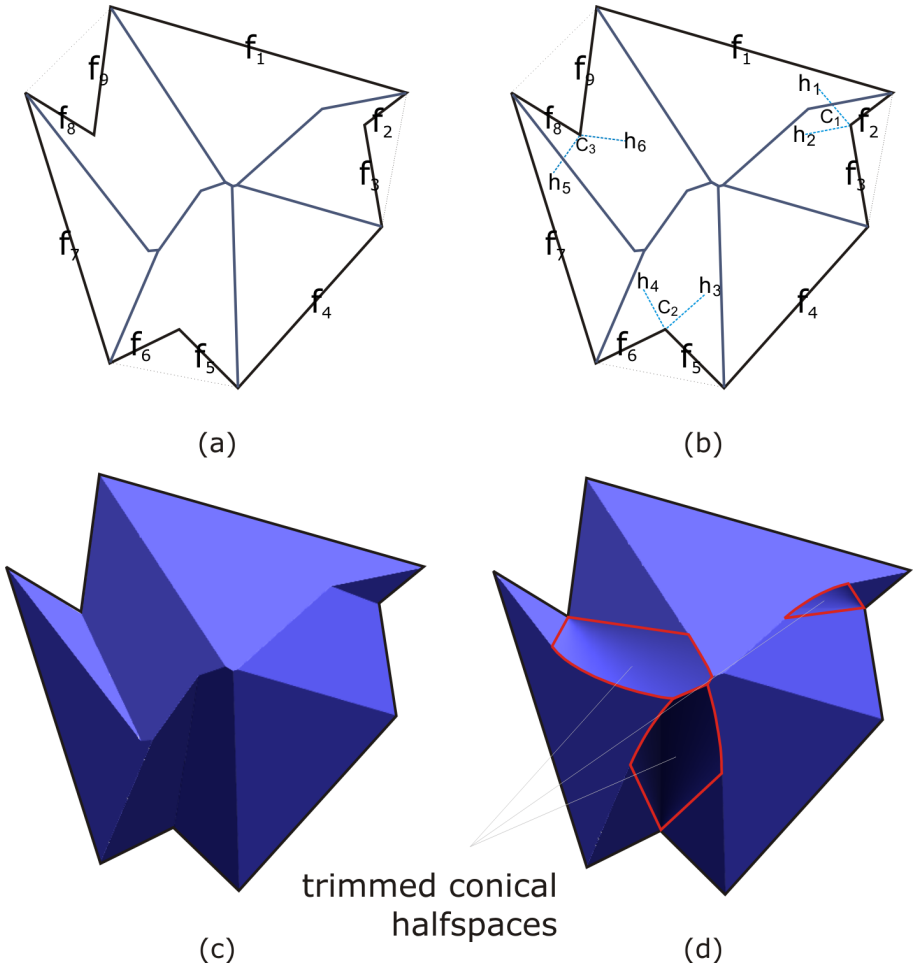


Figure 13: The halfspaces defining a polygonal domain (a) and the corresponding approximate distance function (c); trimmed conical halfspaces are added in (b) and the exact distance function is illustrated in (d) together with the contribution of the trimmed conical halfspaces.

The approximate distance function given in equation (9) can be converted into an exact distance function by adding trimmed conical halfspaces at the concave vertices as follows

$$z_1 \rightarrow z_1 \cup (h_1 \cap h_2 \cap c_1) \tag{10}$$

$$z_2 \rightarrow z_2 \cup (h_3 \cap h_4 \cap c_2) \tag{11}$$

$$z_3 \rightarrow z_3 \cup (h_5 \cap h_6 \cap c_3). \tag{12}$$

The effect of these trimmed conical halfspaces on the resulting distance function is illustrated in Figure 13(d). Therefore, the exact distance function will be given by the same equation (9), but with the new halfspaces $z_i, i = 1, 2, 3$ that contain the trimmed conical halfspaces. The projection of this exact distance function generates the medial axis shown in Figure 5(c).

References

[1] H. Blum. A transformation for extracting new descriptions of shape. In *Models for the Perception of Speech and Visual Form*, pages 362–380, 1967.

[2] Dominique Attali, Jean-Daniel Boissonnat, and Herbert Edelsbrunner. Stability and computation of medial axes — a state-of-the-art report. In Torsten Möller, Bernd Hamann, and Robert Russel, editors, *Mathematical Foundations of Scientific Visualization, Computer Graphics, and Massive Data Exploration*, Mathematics and Visualization. Springer Verlag, 2008.

[3] Amir Shaham, Ariel Shamir, and Daniel Cohen-Or. Medial axis based solid representation. In *SM '04: Proceedings of the ninth ACM symposium on Solid modeling and applications*, pages 37–44, Aire-la-Ville, Switzerland, Switzerland, 2004. Eurographics Association.

[4] M. van Eede, D. Macrini, A. Telea, C. Sminchisescu, and S. S. Dickinson. Canonical skeletons for shape matching. In *ICPR '06: Proceedings of the 18th International Conference on Pattern Recognition*, pages 64–69, Washington, DC, USA, 2006. IEEE Computer Society.

[5] Tyng-Luh Liu, Davi Geiger, and Robert V. Kohn. Representation and self-similarity of shapes. In *ICCV '98: Proceedings of the Sixth International Conference on Computer Vision*, page 1129, Washington, DC, USA, 1998. IEEE Computer Society.

[6] Wooi-Boon Goh. Strategies for shape matching using skeletons. *Comput. Vis. Image Underst.*, 110(3):326–345, 2008.

[7] Thomas B. Sebastian, Philip N. Klein, and Benjamin B. Kimia. Recognition of shapes by editing their shock graphs. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(5):550–571, 2004.

[8] Kaleem Siddiqi, Ali Shokoufandeh, Sven J. Dickinson, and Steven W. Zucker. Shock graphs and shape matching. *Int. J. Comput. Vision*, 35(1):13–32, 1999.

[9] James Damon. Determining the geometry of boundaries of objects from medial data. *Int. J. Comput. Vision*, 63(1):45–64, 2005.

[10] Krishnan Suresh. Automating the CAD/CAE dimensional reduction process. In *SM '03: Proceedings of the eighth ACM symposium on Solid modeling and applications*, pages 76–85, New York, NY, USA, 2003. ACM.

[11] Evan C. Sherbrooke, Nicholas M. Patrikalakis, and Erik Brisson. Computation of the medial axis transform of 3-D polyhedra. In *SMA '95: Proceedings of the third ACM symposium on Solid modeling and applications*, pages 187–200, New York, NY, USA, 1995. ACM.

[12] Stephen M. Pizer, Kaleem Siddiqi, Gabor Székely, James N. Damon, and Steven W. Zucker. Multiscale medial loci and their properties. *Int. J. Comput. Vision*, 55(2-3):155–179, 2003.

[13] Jayant Shah. Gray skeletons and segmentation of shapes. *Comput. Vis. Image Underst.*, 99(1):96–109, 2005.

[14] Benjamin B. Kimia and Amir Tamrakar. The role of propagation and medial geometry in human vision. In *BMCV '02: Proceedings of the Second International Workshop on Biologically Motivated Computer Vision*, pages 219–229, London, UK, 2002. Springer-Verlag.

[15] Fred L. Bookstein. The line skeleton. *Computer Graphics and Image Processing*, 11(2):123–137, oct 1979.

[16] H. Blum and R. Nagel. Shape description using weighted symmetric axis features. *Pattern Recognition*, 10, 1978.

[17] Peter Sampl. Semi-structured mesh generation based on medial axis. In *IMR*, pages 21–32, 2000.

[18] Peter Sampl. Medial axis construction in three dimensions and its application to mesh generation. *Eng. Comput. (Lond.)*, 17(3):234–248, 2001.

[19] William Roshan Quadros, Kenji Shimada, and Steven James Owen. Skeleton-based computational method for generation of 3D finite element mesh sizing function. *Engineering with Computers*, 20:249–264, 2004.

[20] O. Aichholzer, F. Aurenhammer, D. Alberts, and B. Grtner. A novel type of skeleton for polygons. *Journal of Universal Computer Science*, 1(12):752–761, 1995.

[21] Erik D. Demaine, Martin L. Demaine, and Anna Lubiw. Folding and cutting paper. In *JCDCG '98: Revised Papers from the Japanese Conference on Discrete and Computational Geometry*, pages 104–118, London, UK, 2000. Springer-Verlag.

[22] Gill Barequet, Michael T. Goodrich, Aya Levi-Steiner, and Dvir Steiner. Straight-skeleton based contour interpolation. In *SODA '03: Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 119–127, Philadelphia, PA, USA, 2003. Society for Industrial and Applied Mathematics.

[23] Mirela Tănase and Remco C. Veltkamp. Polygon decomposition based on the straight line skeleton. In *SCG '03: Proceedings of the nineteenth annual symposium on Computational Geometry*, pages 58–67, New York, NY, USA, 2003. ACM.

[24] Robert L. Blanding, George M. Turkiyyah, Duane W. Storti, and Mark A. Ganter. Skeleton-based three-dimensional geometric morphing. *Comput. Geom. Theory Appl.*, 15(1-3):129–148, 2000.

[25] F. Chazal and R. Soufflet. Stability and finiteness properties of medial axis and skeleton. *Journal of Dynamical and Control Systems*, 10(2):149–170, April 2004.

[26] Vadim Shapiro. Semi-analytic geometry with R-functions. *Acta Numerica*, 18(3):239–303, 2007.

[27] André Lieutier. Any open bounded subset of R^n has the same homotopy type than its medial axis. In *SM '03: Proceedings of the eighth ACM symposium on Solid modeling and applications*, pages 65–75, New York, NY, USA, 2003. ACM.

[28] Francis Y. Chin, Jack Snoeyink, and Cao An Wang. Finding the medial axis of a simple polygon in linear time. In *ISAAC '95: Proceedings of the 6th International Symposium on Algorithms and Computation*, pages 382–391, London, UK, 1995. Springer-Verlag.

[29] H. I. Choi, S. W. Choi, and H. P. Moon. Mathematical theory of medial axis transform. *Pacific Journal of Mathematics*, 1(181):57–88, 1997.

[30] Rajesh Ramamurthy and Rida T. Farouki. Voronoi diagram and medial axis algorithm for planar domains with curved boundaries — II: detailed algorithm description. *J. Comput. Appl. Math.*, 102(2):253–277, 1999.

[31] Tim Culver, John Keyser, and Dinesh Manocha. Exact computation of the medial axis of a polyhedron. *Comput. Aided Geom. Des.*, 21(1):65–98, 2004.

[32] Jonathan W. Brandt and V. Ralph Algazi. Continuous skeleton computation by voronoi diagram. *CVGIP: Image Underst.*, 55(3):329–338, 1992.

[33] Nina Amenta, Sunghee Choi, and Ravi Krishna Kolluri. The power crust, unions of balls, and the medial axis transform. *Computational Geometry: Theory and Applications*, 19:127–153, 2001.

[34] Tamal K. Dey, Hyuckje Woo, and Wulue Zhao. Approximate medial axis for CAD models. In *SM '03: Proceedings of the eighth ACM symposium on Solid modeling and applications*, pages 280–285, New York, NY, USA, 2003. ACM.

[35] JJ Chou. Voronoi diagrams for planar shapes. *IEEE Computer Graphics and Applications*, 15(2):52–59, 1995.

[36] Kaleem Siddiqi, Sylvain Bouix, Allen Tannenbaum, and Steven W. Zucker. Hamilton-jacobi skeletons. *Int. J. Comput. Vision*, 48(3):215–231, 2002.

[37] Haixia Du and Hong Qin. Medial axis extraction and shape manipulation of solid objects using parabolic PDEs. In *SM '04: Proceedings of the ninth ACM symposium on Solid modeling and applications*, pages 25–35, Aire-la-Ville, Switzerland, Switzerland, 2004. Eurographics Association.

[38] R. Kimmel, D. Shaked, N. Kiryati, and A.M. Bruckstein. Skeletonization via distance maps and level sets. *Computer Vision and Image Understanding*, 62(3):382–391, November 1995.

[39] José Gomes and Olivier Faugeras. Reconciling distance functions and level sets. *Journal of Visual Communication and Image Representation*, 11(2):209–223, 2000.

[40] M. Ramanathan and B. Gurumoorthy. Constructing medial axis transform of planar domains with curved boundaries. *Computer-Aided Design*, 35(7):619–632, 2003.

[41] Lixin Cao and Jian Liu. Computation of medial axis and offset curves of curved boundaries in planar domain. *Comput. Aided Des.*, 40(4):465–475, 2008.

[42] Kenneth E. Hoff, III, John Keyser, Ming Lin, Dinesh Manocha, and Tim Culver. Fast computation of generalized Voronoi diagrams using graphics hardware. In *SIGGRAPH '99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 277–286, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co.

[43] David Eppstein and Jeff Erickson. Raising roofs, crashing cycles, and playing pool: applications of a data structure for finding pairwise interactions. In *SCG '98: Proceedings of the fourteenth annual symposium on Computational geometry*, pages 58–67, New York, NY, USA, 1998. ACM.

[44] Petr Felkel and Stepan Obdrzalek. Straight skeleton implementation. In *Spring Conference on Computer Graphics*, pages 210–218. Lazlo Szimay-Kalos (ed.), 1998.

[45] V. Shapiro. Well-formed set representations of solids. *International Journal of Computational Geometry and Applications*, 9(2):125–150, 1999.

[46] S. B. Tor and A. E. Middleditch. Convex decomposition of simple polygons. *ACM Trans. Graph.*, 3(4):244–265, 1984.

[47] David Dobkin, Leonidas Guibas, John Hershberger, and Jack Snoeyink. An efficient algorithm for finding the CSG representation of a simple polygon. In *SIGGRAPH '88: Proceedings of the 15th annual conference on Computer graphics and interactive techniques*, pages 31–40, New York, NY, USA, 1988. ACM.

[48] Vadim Shapiro. A convex deficiency tree algorithm for curved polygons. *Int. J. Computational Geometry and Applications*, 11(2):215–238, April 2001.

[49] Vadim Shapiro and Donald L. Vossler. Separation for boundary to CSG conversion. *ACM Trans. Graph.*, 12(1):35–55, 1993.

[50] Suzanne F. Buchele and Richard H. Crawford. Three-dimensional halfspace constructive solid geometry tree construction from implicit boundary representations. In *SM '03: Proceedings of the eighth ACM symposium on Solid modeling and applications*, pages 135–144, New York, NY, USA, 2003. ACM.

[51] Vadim Shapiro and Donald L. Vossler. Construction and optimization of CSG representations. *Comput. Aided Des.*, 23(1):4–20, 1991.

[52] D. Marr and E. Hildreth. Theory of Edge Detection. *Proceedings of the Royal Society of London. Series B, Biological Sciences*, pages 187–217, 1980.

[53] Alan C. Bovik. *Handbook of Image and Video Processing (Communications, Networking and Multimedia)*. Academic Press, Inc., Orlando, FL, USA, 2005.

[54] AM López, F. Lumbreras, J. Serrat, and JJ Villanueva. Evaluation of methods for ridge and valley detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(4):327–335, 1999.

[55] Tran Le Hong Du, Duong Anh Duc, and Duong Nguyen Vu. Ridge and valley based face detection. pages 237–243, 2006.

[56] Rida T. Farouki and John K. Johnstone. Computing point/curve and curve/curve bisectors. In *Proceedings of the 5th IMA Conference on the Mathematics of Surfaces*, pages 327–354, New York, NY, USA, 1994. Clarendon Press.

[57] Gershon Elber and Myung-So Kim. The bisector surface of rational space curves. *ACM Trans. Graph.*, 17(1):32–49, 1998.

[58] Gershon Elber and Myung-Soo Kim. Computing rational bisectors. *IEEE Comput. Graph. Appl.*, 19(6):76–81, 1999.

[59] Martin Peternell. Geometric properties of bisector surfaces. *Graphical Models*, 62(3):202–236, May 2000.

[60] X. Chen, R.F. Riesenfeld, and E. Cohen. Complexity Reduction for Symbolic Computation with Rational B-splines. *INTERNATIONAL JOURNAL OF SHAPE MODELING*, 13(1):25, 2007.

[61] J. A. Sethian. *Level Set Methods and Fast Marching Methods*. Cambridge University Press, 1999.

[62] J. A. Sethian and Andreas Wiegmann. Structural boundary design via level set and immersed interface methods. *J. Comput. Phys.*, 163(2):489–528, 2000.

[63] Zhen Luo, Liyong Tong, Michael Yu Wang, and Shengyin Wang. Shape and topology optimization of compliant mechanisms using a parameterization level set method. *J. Comput. Phys.*, 227(1):680–705, 2007.

[64] Jiaqin Chen, Michael Freytag, and Vadim Shapiro. Shape sensitivity of constructive representations. In *SPM '07: Proceedings of the 2007 ACM symposium on Solid and physical modeling*, pages 85–95, New York, NY, USA, 2007. ACM.

[65] Jiaqin Chen, Vadim Shapiro, Krishnan Suresh, and Igor Tsukanov. Shape optimization with topological changes and parametric control. *International Journal for Numerical Methods in Engineering*, 71(3):313–346, 2007.

[66] H. Pottmann and M. Hofer. Geometry of the squared distance function to curves and surfaces. Technical report, Vienna University of Technology, January 2002.

[67] M. Aigner and B. Jüttler. Robust computation of foot points on implicitly defined curves. *Mathematical methods for curves and surfaces: Troms*, pages 1–10, 2005.

[68] X. Chen, E. Cohen, and R.F. Riesenfeld. Tracking Point-Curve Critical Distances. *Lecture Notes in Computer Science*, 4077:87, 2006.

[69] Xianming Chen. An application of singularity theory to robust geometric calculation of intersections among dynamically deforming geometric objects. *University of Utah*, May 2008.