ACM Symposium on Solid Modeling and Applications (2004)
G. Elber, N. Patrikalakis, P. Brunet (Editors)

Efficient and Robust Computation of an Approximated Medial Axis

Yuandong Yang Oliver Brock Robert N. Moll

Laboratory for Perceptual Robotics
Department of Computer Science
University of Massachusetts Amherst
Email: {yuandong, oli, moll}@cs.umass.edu

Abstract

The medial axis can be viewed as a compact representation for an arbitrary model; it is an essential geometric structure in many applications. A number of practical algorithms for its computation have been aimed at speeding up its computation and at addressing its instabilities. In this paper we propose a new algorithm to compute the medial axis with arbitrary precision. It exhibits several desirable properties not previously combined in a practical and efficient algorithm. First, it allows for a trade-off between computation time and accuracy, making it well-suited for applications in which an approximation of the medial axis suffices, but computational efficiency is of particular concern. Second, it is output sensitive: the computation complexity of the algorithm does not depend on the size of the representation of a model, but on the size of the representation of the resulting medial axis. Third, the densities of the approximated medial axis points in different areas are adaptive to local free space volumes, based on the assumption that a coarser approximation in wide open area can still suffice the requirements of the applications. We present theoretical results, bounding the error introduced by the approximation process. The algorithm has been implemented and experimental results are presented that illustrate its computational efficiency and robustness.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling

1. Introduction

The medial axis of a solid D is the locus of points inside D, which lie at the centers of all non-intersecting closed discs or balls in D of maximum radius that have at least two contact points with D [5, 28]. Given a medial axis of D and its associated radius function, it is easy to reconstruct the surface of D. Thus the medial axis provides a compact representation of D. The medial axis is helpful in many applications, such as shape analysis [24, 36], robot motion planning [13, 18], image processing [22], computer vision [7, 23], solid modeling [14, 17, 31] and mesh generation [1, 27]. In this paper we present a novel approach to compute an approximated medial axis based on the aforementioned definition.

While various algorithms to compute the exact medial axis of simple polyhedra composed of a few hundreds of triangles exist [21, 26, 28], it is non-trivial to scale them to more complicated models. This is a consequence of the instability and the algebraic complexity of the medial axis as a mathematical structure. A small change of the surface can cause relatively large changes in the medial axis; furthermore, the algebraic representation of the medial axis usually is of higher degree than the surface of the solid. In order to address the instability problem and reduce the computation time, algorithms to approximate the medial axis have been proposed. These will be discussed in detail in Section 2.

There are a number of applications for which computation time is of critical concern, but an exact representation of the medial axis is not needed. For example, robotic motion planning techniques based on the medial axis have been devised [13, 18]. These methods exploit the fact that the medial axis captures points at a maximal distance from obstacles and use it as a heuristic to find positions of the robot that do not collide with the environment [18]. For this application an approximated medial axis, which can be computed efficiently, is very desirable. Furthermore, in narrow and geometrically complex regions, a more detailed representation of the medial axis is of interest, whereas in wide open regions the danger of collision is low and a coarse representation suffices.

In this paper, we propose a novel algorithm to approximate the medial axis. Motivated by applications such as robot motion planning, the proposed algorithm has the following characteristics:

1. The computation of the approximated medial axis can be performed very efficiently. As shown in Section 7, the approximated medial axis of the model of the Stanford Bunny with 69,451 triangles can be computed in only 5.6 seconds on a standard PC. The approximated medial axis of the Happy Buddha model, consisting of over one million triangles, can be computed in 13.4 seconds. This is much faster than existing algorithms with the same input.

2. The algorithm allows the user to trade off speed of computation for accuracy: the more accurate the approximation, the more computation time is required.
3. The algorithm adapts the density of points used to represent the approximated medial axis based on local properties of the solid. Few points are necessary in open areas and a denser set of points is used to represent the approximated medial axis in confined areas.

As a consequence, the proposed approach is well suited for application in which an efficiently computed, approximated medial axis is of value. This has been validated experimentally in the domain of sampling-based motion planning in robotics [34].

2. Related Work

Given the geometric description of a solid, there are various algorithms to compute its medial axis. We classify the approaches found in the literature based on their method of exploring the interior of the solid.

The algorithms in the first category use a tracing approach [21, 26, 28] to compute the medial axis. The algorithm starts from a junction point, which is the point where multiple facets of the medial axis intersect, and traces the seams from the junction point until they meet another junction point or an end point. This procedure is repeated recursively until all parts of the medial axis have been traced. In addition, Culver [9] also uses a spatial subdivision technique to reduce the running time of the algorithm. Choset [8] applies this approach to sensor-based motion planning. His algorithm only uses local distance information gained from the sensors to incrementally build a medial axis representation, as the robot explores its environment. The tracing steps are small and computation speed is limited. Holleman [18] uses a similar approach to determine the medial axis, which is then used as a heuristic for sampling free configuration space in the probabilistic roadmap approach to robot motion planning. Computational considerations limit the applicability of these approaches to polyhedra composed of only a few thousand faces.

The algorithms in the second category represent the free space as voxels. Lam [19] gives a survey of thinning algorithms based on voxel representations and Zhang [35] provides a performance evaluation. The thinning algorithms perform erosion in order to compute an approximated medial axis. Siddiqi [29] assigns each voxel a vector field value corresponding to the vector pointing to the closest point on the surface. A voxel is considered to be on the medial axis if the mean flux of the vector field entering the neighbor voxels is positive. Ragnemalm [25] assigns to each voxel the Euclidean distance to its nearest voxel on the boundary and computes the local directional maxima to determine the approximated medial axis, while Hoff [16] utilizes hardware to compute these, thus speeding up the computation. Vleugels [33] divides voxels recursively if they contain a portion of the medial axis until they reach a minimum size. Foskey [11] combines the advantages of [16] and [33], using hardware to compute distances while adaptively and recursively dividing the voxels. This method computes a simplified medial axis [11], which is a subset of the actual medial axis. This new data structure is more stable but it does not necessarily maintain the connectivity of the medial axis, thereby limiting its applicability.

The algorithms in the third category divide the free space into Voronoi regions based on sample points on the surface of the solid. The resulting Voronoi regions are tiny because a dense sampling of the surface is required for an accurate computation of the medial axis. Both [3] and [10] provide good surveys of the literature of the algorithms in this category. These algorithms are applicable to complicated models, if an appropriate set of sample points can easily be determined.

3. Sampling the Approximated Medial Axis

The main idea of our algorithm is to efficiently generate a small set of partially overlapping maximal spheres to cover almost the entire free space within the solid. These spheres are constructed to intersect features of the medial axis. By sampling points on the surface of the spheres and determining their closest feature in the environment, a set of points on the surface of the sphere that are in proximity to the medial axis can be identified. The points serve as an approximation to the medial axis. The union of these points comprises the approximated medial axis, abbreviated as aMA in the remainder of the paper. Because large open areas can be covered by large spheres, the aMA consists of few points in wide open areas and is denser in geometrically complex regions of the solid. The precision with which the aMA approximates the medial axis can be specified as a parameter of the algorithm. This allows users to consciously trade accuracy for computational efficiency.

3.1. Description of the Algorithm

Each point inside the solid has at least one closest feature on the surface of the solid. The *direction vector* $\vec{v}$ of a point p in the solid D is the unit vector pointing from point p to the closest feature on the surface of D. The *distance* $\delta(p)$ associated with a point p in the solid D is the distance from point p to the closest feature on the surface of D. Note that points on the medial axis must have at least two direction vectors.

The description of the algorithm relies on two primitive operations. The first identifies an initial point m and associated distance $\delta(m)$, such that the resulting sphere of radius $\delta(m)$ around m intersects the medial axis. Given a solid D, a set of points P in the interior of D and their direction vectors, the second primitive identifies, for each point $p \in P$, that point on the medial axis of D which is closest to p. These primitives will be described after we have detailed the algorithm for computing the aMA.

Assume point m lies on the medial axis and is distance $\delta(m)$ away from the closest obstacle. This point is determined using the first primitive. A priority queue Q is initialized to contain the sphere described by point m and radius $\delta(m)$. The set S of spheres describing the free space inside the solid D is initialized to be the empty set.

The largest sphere s is extracted from Q and a set U of uniformly distributed samples that have been generated on its surface. Points in U that are contained in one of the spheres in S are discarded. The second aforementioned primitive is used to determine those points in U that lie closest to the medial axis. These points p_i are added into the aMA and, along with their distances $\delta(p_i)$, into the priority queue Q. The sphere s is added to S. To bound the exploration of free space we introduce an additional requirement for insertion into Q: only those spheres with radii larger than the *expansion threshold*

K_e can be added. We can control computation time and the number of aMA points by changing K_e. These steps are repeated until Q is empty (see also Figure 1).

1. Find point m inside D such that $\delta(m) > K_e$ and the medial axis intersects the sphere of radius $\delta(m)$ around m (see Section 3.2).
2. Sphere set $S := \emptyset$
3. Medial axis point set $M := \emptyset$
4. Priority queue $Q := \{(m, \delta(m))\}$
5. While Q is not empty
   a. Extract sphere $s = (p, \delta(p))$ from Q
   b. Generate n uniformly distributed samples $U = \{u_1, \cdots, u_n\}$ on the surface of s. Discard all $u_i \in U$ for which $\exists s_j \in S$ such that $u_i \in s_j$.
   c. Using U and the direction vectors associated with the $u_i \in U$, determine approximated medial axis points $A = \{a_1, \cdots, a_k\}$ (see Section 3.3).
   d. $Q := Q \cup \{(a_i, \delta(a_i)) \mid a_i \in A \text{ and } \delta(a_i) > K_e\}$
   e. $M := M \cup A$
   f. $S := S \cup \{(p, \delta(p))\}$
6. Connect points in M to generate the aMA (see Section 5)

Figure 1: *The pseudo code of the algorithm. S is the set of spheres describing the interior of the solid D. M is the set of points describing the approximated medial axis. Q is the priority queue of spheres, ordered by radius.*

Figure 2 illustrates our method in a two-dimensional case. Assume o_1 is the first element in Q. A maximal circle centered at o_1 is generated and n samples $p_1, p_2, ..., p_n$ are generated on its circumference (not all samples are shown in the figure). The point pairs (p_1, p_2), (p_3, p_4) and (p_5, p_6) have different direction vectors and the midpoints of these pairs, q_1, q_2 and q_3, are considered to be on the medial axis; they are added to the aMA and to the queue. Since q_3 has the largest radius it is expanded next and the procedure repeats.

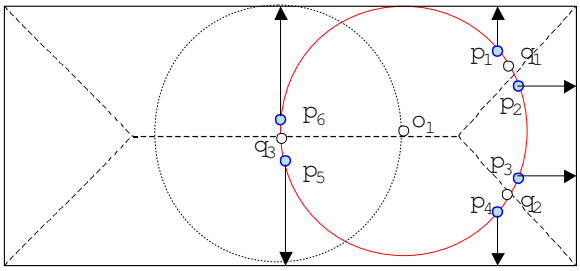

Figure 2: *An illustration of the algorithm. The dashed lines represent the medial axis of the rectangle.*

We now discuss the two primitives used in the description of the algorithm.

3.2. Identifying the Initial Approximated Medial Axis Point

In the description of the algorithm it was assumed that the queue Q is initialized with a point m on the medial axis of the solid D. We use a similar expansion algorithm as the one described above to find m. Starting from a random point p inside D, we generate the maximal sphere with the center at p. If we cannot find a medial axis point of the surface of the sphere (how medial axis points are identified is described in Section 3.3), we take the point p' with the biggest radius as the center of the next sphere. This process is repeated until the sphere of radius $\delta(p')$ around p' intersects the medial axis. Since this procedure converges towards a sphere of locally maximum radius, its center converges towards a point p' on the medial axis and the surrounding sphere thus must intersect the medial axis.

3.3. Identifying Approximated Medial Axis Points

Given a set of uniformly distributed sample points U on the surface of a sphere, we apply the *separation angle criteria* [3, 4, 6, 11] to determine the set containing points of the aMA. If the direction vectors of two adjacent sample points span an angle larger than a threshold θ_t, we take the samples' midpoint as the aMA points. In Section 4 we will bound the error of this approximation.

The separation angle criteria can erroneously place a point on the medial axis (see Figure 3 for an example) [11]. If a reflex vertex on the boundary is the nearest neighbor to both sample points, both direction vectors will point toward that vertex. To identify these cases, we apply the divergence criteria: the direction vectors must point away from each other.

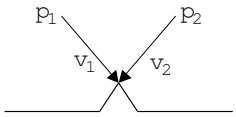

Figure 3: *False positives for the separation angle criteria*

If a sphere only intersects one facet of the aMA, there will be two sets of sample points, each set with a different direction vector, based on classification by the angle criteria. In this case we simply insert the center of the sphere into the aMA. The samples on the surface are superfluous. If a sphere intersects multiple facets of the medial axis, however, we identify adjacent samples with three or more distinct direction vectors and add their midpoint to the aMA. These points are called *critical points*; they designate an edge or a vertex between multiple facets of the aMA. Critical points can be used to approximate the hierarchical generalized Voronoi graph[8].

3.4. Increasing the Accuracy of the Approximation

Our basic algorithm does not sample the interior of the spheres. By restricting sampling to the surface of the sphere, important features of the aMA might be missed. We provide an optional refinement algorithm which can locate aMA points with a maximal number of adjacent aMA facets on the inside of a sphere. Starting from one aMA point, we use a tracing algorithm similar to the one proposed in [18] to look for a point where several aMA facets touch. Our method differs from [18] in that it performs the tracing in 3D, rather than in a plane. The method uniformly samples n points nearby and selects the point with equal or more adjacent aMA facets as the next tracing point. The refinement algorithm stops if the tracing point is outside of the sphere or all points nearby have less MA facets crossing. If the refinement point is close to another critical point, we discard it and stop the algorithm. The aMA points found

in this manner more accurately describe the characteristics of the solid. This refinement considerably increases the computation time of the overall algorithm.

3.5. Discussion

In Section 2, we differentiated three categories of approaches to medial axis computation. These methods were classified according to their methods of free space exploration. All of them generate the (approximated) medial axis either during or after the exploration process. The algorithm proposed here can also be regarded as a tracing approach: it traces the medial axis by sampling on the spheres during expansion. However, the algorithm differs importantly from previous approaches: during the tracing progress, the step size is adjusted based on the local geometrical properties of the free space. By construction, the centers of the spheres lie close to the medial axis and thus free space is explored with near-optimal step sizes. To ensure this property all possible tracing directions are explored at every step. The near-optimal step size is the main reason for the computational efficiency of the proposed approach.

4. Approximation Error

In this section we bound the error made by the proposed method of approximating the medial axis of a solid. We differentiate quantitative errors that result from the finite sampling density of our algorithm, and qualitative errors. The latter are a consequence of the sphere expansion algorithm to explore the free space inside the solid.

When considering the quantitative error, two cases have to be distinguished. If the sampling density on the surface is kept constant, larger spheres will be covered by a larger number of samples than smaller spheres. In this case we compute the absolute error as the distance of a point on the aMA to the closest point on the true medial axis. If the number of samples is kept constant, on the other hand, the sampling density is lower on large spheres, which means that fewer samples are computed in wide open areas. In other words, the computational expense is proportional to the difficulty of the region – a desirable property for an approximation algorithm. For this case we will consider a relative error, which relates the absolute error to the amount of local free space.

4.1. Sample Point Accuracy

Let M be the set of approximated medial axis points for a given solid D attained by our algorithm. For each point p_i in M, there exists a medial axis point t_i which is closest to p_i. The absolute and relative errors for the sample points M of the approximated medial axis, relative to the true medial axis T are

- Absolute error ε_a: $\max_{p_i \in M}\{|p_i - t_i|\}$
- Relative error ε_r: $\frac{\varepsilon_a}{\delta(t_i)}$

Given a set of uniform samples on a unit sphere (circle), we define two points as neighbors of each other if the distance between them is smaller than the *neighbor threshold* d_n. If the closest features of two neighboring points p_1 and p_2 are different, there is a point between them which is closest to both of those features and thus lies on the medial axis. We will consider the midpoint m of the line segment between p_1 and p_2 as a point on the approximated medial axis. Consequently, the maximum distance between points of the approximated medial axis and the real medial axis is given by $\varepsilon_a = \frac{d_n}{2}$. This equation holds in both 2D for circles and 3D for spheres. The *resolution* of the algorithm is then defined by $r_{\min}\frac{d_n}{2}$, where $r_{\min}$ is the radius of the smallest sphere generated by the algorithm described in Section 3.

It is obvious that there is a relation between the number of the samples placed on a sphere and the quality of the approximated medial axis. In the following sections we establish how many samples are needed in order to achieve a desired absolute or relative error in 2D or 3D.

4.1.1. 2D

It is relatively easy to compute the number N of samples needed for a given absolute error ε_a. During the exploration of free space, the algorithm uniformly samples N points on a circle (refer to Figure 2). The angle $\Delta\theta$ between two neighbor sample points (for example, $\angle\, \overrightarrow{q_1p_1}\, \overrightarrow{q_1p_2}$) is $\frac{2\pi}{N}$ and the distance between them is given by

$$|\overrightarrow{p_1p_2}| = 2r\sin\frac{\Delta\theta}{2} = 2r\sin\frac{\pi}{N}.$$

Let d_n equal $|\overrightarrow{p_1p_2}|$. The absolute error is then given by:

$$\varepsilon_a = \frac{d_n}{2} = r\sin\frac{\pi}{N}.$$

Consequently, given an absolute error ε_a the number of samples needed is given as a function of the radius r of the sphere:

$$N = \frac{\pi}{\arcsin\frac{\varepsilon_a}{r}} \tag{1}$$

To determine the relative error ε_r, we have to estimate the radius of a nearby point m on the true medial axis. In Figure 2, assume the closest MA point to q_1 is m. The r in equation 1 is the distance $\delta(q_1)$ and not $\delta(m)$. Although m and $\delta(m)$ are unknown, according to the definition of the medial axis and the triangle inequality, we can estimate $\delta(q_1) \le \delta(m) < \delta(q_1) + \varepsilon_a$ under the condition that m has the same direction vectors as q_1. Thus $\frac{\varepsilon_a}{\delta(q_1)+\varepsilon_a} < \frac{\varepsilon_a}{\delta(m)} \le \frac{\varepsilon_a}{\delta(q_1)}$. If $r \le \delta(q_1)$, $\frac{\varepsilon_a}{\delta(m)} \le \frac{\varepsilon_a}{\delta(q_1)} \le \frac{\varepsilon_a}{r}$, we can select N based on $\frac{\varepsilon_a}{r}$:

$$N = \frac{\pi}{\arcsin\varepsilon_r}$$

Should $r \ge \delta(q_1)$, we determine N based on the following equation:

$$N = \frac{\pi}{\arcsin\frac{\varepsilon_r r}{\delta(q_1)}}$$

Given the number N of desired sample points p_i and the origin o of a sphere with radius r, it is easy to determine their position:

$$p_i = o + r\begin{pmatrix}\cos\frac{2\pi i}{N}\\ \sin\frac{2\pi i}{N}\end{pmatrix} \quad \text{for } i = 1,\cdots,N.$$

4.1.2. 3D

Before we discuss the accuracy of the algorithm applied to a three-dimensional solid, we introduce the *sphere covering problem* [15, 30]: How can n points be placed on a unit sphere so as to minimize the maximum distance of any point on the sphere to the other $n-1$ points? Alternatively, the problem can be defined in

terms of covering the unit sphere with spherical caps [2, 32]. We rely on approaches presented in the literature to determine a uniformly spaced sampling pattern on spheres during the free space exploration phase of the algorithm.

Determining the approximated medial axis point based on two adjacent sample points on a sphere with radius r, the absolute error is given by $\varepsilon_a = r\frac{d_n}{2}$. If more than two samples with different direction vectors are used to determine a point the aMA, the triangle inequality is used to bound the error by $\varepsilon_a < r\, d_n$. Given a desired maximum absolute error, it is easy to determine the maximum allowed distance d between samples. Using the methods for uniform sampling on the unit sphere referenced above, we determine the number of samples N and their locations on the sphere.

The relative error for the three-dimensional case is determined in a similar fashion as in the two-dimensional case.

4.2. Missing Features of the Medial Axis

The aMA points we obtain only represent a subset of the simplified medial axis [11]. The proposed algorithm misses part of the simplified medial axis because it does not expand spheres into the entire free space and because it only considers a small set of points on the surfaces of spheres.

Given an absolute error ε_a, the algorithm will stop if the maximum $\delta(p)$ of all sample points p is smaller than ε_a. For a relative error ε_r, the algorithm will terminate if the radius of all spheres is smaller than the threshold K_e. Obviously, the algorithm can not reach a space with a 'gate' smaller than ε_a or the threshold K_e. The spheres stop spreading when they meet that 'gate' and the free space behind the gate will be missed, including the associated features of the medial axis.

In the case of relative error, the sample points on large spheres are sparser and thus have larger distance d. Consequently, using relative error, the size of the largest gate which is missed depends on the amount of local free space. This is illustrated in Figure 4: p_1 and p_2 are adjacent sample points on the bigger circle and θ_b is the angle between the direction vectors of p_1 and p_2; q_1 and q_2 are the adjacent sample points on the smaller circle and θ_s is the angle between the direction vectors of q_1 and q_2. It is obvious that $|p_1p_2| > |q_1q_2|$ and $\theta_b < \theta_s$. According to the separation angle criteria, the algorithm can find the aMA point on the smaller circle but not on the bigger circle, if $\theta_b < \theta_t < \theta_s$.

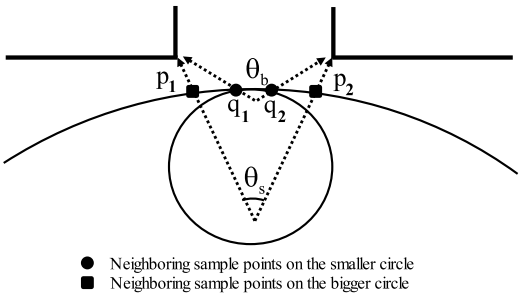

Figure 4: *The algorithm might miss more features when it uses the relative error criterion.*

5. Generating the Approximated Medial Axis

So far we have obtained a set of aMA points by sampling on spheres during the exploration process. For some applications it might suffice to have sampled points on the aMA; other applications might rely on knowing the actual facets of the aMA. We now describe how the adjacency relationship between the sampled aMA points discovered during the free space exploration is exploited to compute facets of the medial axis.

For a two-dimensional solid the medial axis is composed of one-dimensional curves. We connect the aMA points by straight lines based on the parent-child relationship obtained during exploration: if an aMA point is on the surface of a sphere centered at another aMA point, we connect these two aMA points. We know from the algorithm that all aMA points, except for the first one, are the centers of the maximal spheres and on the surfaces of other spheres. So the segments connecting them are completely inside the spheres and far away from the boundary of the model. If the end points of the aMA segments have different direction vectors, we can explore the bisectors of the aMA segments to locate more aMA points in order to refine the approximated medial axis. We can then connect adjacent children to form facets.

For a three-dimensional solid the medial axis consists of facets of dimension two and below. We use the algorithm presented in [18] to determine the faces of the aMA. In relatively simple models, big MA facets may exist that are covered by more than one sphere. We collect aMA points on the same MA facets based on their direction vectors and positions. In order to achieve better approximation and to improve the visual effect, we use the refinement algorithm described in Section 3.4 to find additional aMA points inside the spheres. The refinement algorithm uses a basic tracing method. Performing this optimization will increase the computation significantly.

6. Computational Complexity

There are two factors determining the computational complexity of the proposed algorithm. The most costly operation performed is the distance computation (see also Section 7). The complexity of the model critically impacts the computational complexity of this operation, which generally is assumed to be $O(\log n)$ in practice, where n is the number of faces of the model. In other words, the computational cost of performing a distance computation is determined by the size of the input.

To the extent that the presented algorithm has to input the description of the environment and performs distance computations on this description, its computational cost is dependent on the input size. If we ignore this for a moment, however, and regard distance computation as a constant time operation, we note that the algorithm is output-sensitive, i.e., its computational cost only depends on the size of the output. For every sample point generated on the aMA we can bound the required computation time by a constant. This is a very desirable property, in particular for an algorithm that trades accuracy for efficiency. This property shows that the computational cost is related to the geometric complexity of the solid itself and not its model. In other words, assuming a cube as the solid and still regarding the cost of distance computation as constant, the proposed algorithm would produce the same aMA with the identical number of distance computations and thus with the

same computational cost, irrespective of whether the cube is represented by 12 triangles or by 12 million triangles. To achieve this result we do not have to take the model into account by specifying a desired step size or resolution, we simply specify the desired relative error.

While output sensitivity is a very desirable property for an algorithm of this kind, the dependency on the geometric complexity, rather than on some combinatorial property based on the number of vertices, makes it impossible to classify the proposed method according to well-known complexity classes.

7. Experimental Results

The experiments reported here were performed on a dual-PentiumIII PC with 1024 MB SRAM and a 32MB DDR NVIDIA GeForce2 GTS graphics card. The algorithm was implemented in C++, with Open Inventor as the graphics API. Distance computation was performed using the PQP package [12, 20]. Several models were used, ranging in complexity from 12 to 1,087,718 triangles. Table 1 summarizes the experimental results.

Figure 5 and 6 are the medial axis of the simple models which have relative big facets. In the image different colors represent different aMA facets. Where more than two aMA facets intersect in a line segment, inaccuracies as a result of fixed-resolution sampling can be observed. The facets of the approximated medial axis also do not reach into 'corners' of the space. This is due to the fact that free space exploration only continues as long as spheres have a certain minimum size.

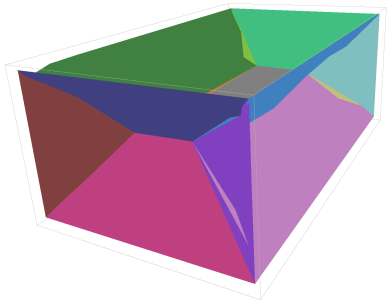

Figure 5: *The medial axis of a box (Box 1, 12 triangles). It takes 3.94 seconds to compute the medial axis. Different colors represent different aMA facets.*

Figure 7 to Figure 10 show more complex models and their approximated medial axes. Our algorithm uses relatively few spheres N_s to cover the free space inside the models and attains a small set of aMA points ($|M| = N_p$). While our method only computes an approximated medial axis, it is significantly faster than previous methods with the same input. For example, at a low resolution the aMA points of the Stanford Bunny, the horse model, and the Happy Buddha can be computed in only 5.61, 12.3 and 13.4 seconds, respectively. Even at higher resolution (see Table 1) the computational cost is significantly lower than previous methods, as known to the authors.

As we described in Section 4, the algorithm may miss some parts

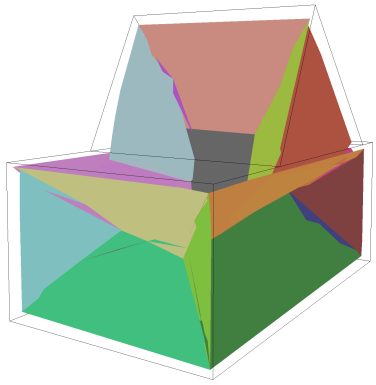

Figure 6: *The medial axis of a simple model (Box 2, 20 triangles). It takes 4.53 seconds to compute the medial axis. Different colors represent different aMA facets.*

of the medial axis when computing its approximation. In order to demonstrate this effect visually, we used spheres with small fixed radius to compute the medial axis of the models; this corresponds to an implementation of a tracing approach. Figure 8 and Figure 9 compare the resulting medial axis with the approximated medial axis computed by the proposed method. It can be seen that most features of the medial axis are preserved. For volumes that are nearly spherical, such as the body of the Stanford Bunny, and are thus captured by a large sphere, some parts of the medial axis are lost. As demonstrated by the aMA in the legs of the horse model, geometrically tight or complex spaces are approximated very well. This comparison visually confirms that the proposed method computes the aMA accurately in tight and complex environments, and determines a coarser approximation in areas that are wide open.

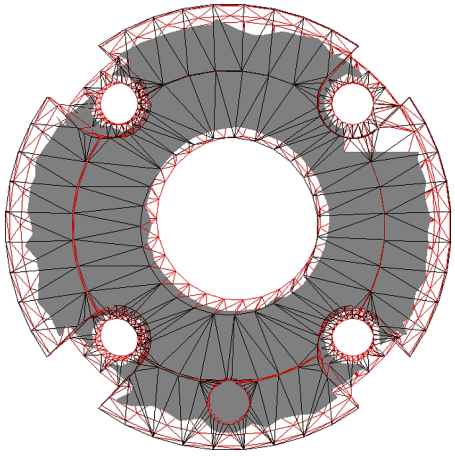

Figure 7: *The approximated medial axis of a flange mount (1,684 triangles) was computed in 3.45 seconds. The approximated medial axis is shown in gray. The surface of the flange mount is shown as a wire frame model.*

The results shown in Table 1 as well as the results shown in Ta-

Model	Δs	N_{sp}	Resolution	N_s	N_p	N_D	T (seconds)
Box1	12	132	0.005646	134	147	7,575	**1.56**
			0.002824	350	363	18,598	**3.94**
Box2	20	132	0.005646	146	169	8,300	**1.84**
			0.002824	382	405	20,335	**4.53**
Flange Mount	1,684	132	0.012	92	172	7,212	**3.45**
			0.006	642	902	63,863	**45.5**
Stanford Bunny	69,451	132	0.00668	91	143	6,564	**5.61**
			0.0033	357	685	20,248	**17.64**
Horse	96,966	132	0.0028	306	400	21,738	**13.3**
Buddha	1,087,718	72	0.006	124	223	3,960	**13.4**
			0.0031	889	1187	29,328	**48.8**
			0.00156	3116	3627	101,918	**174.4**

Table 1: *This table shows experimental results for a variety of models. The aMA was computed for a separation angle of $\frac{\pi}{3}$. Δs is the number of the triangles of the model; N_{sp} is the number of samples on each sphere. Resolution refers to the maximum distance between neighboring points on the smallest sphere generated during the computation of the aMA ($\frac{d_n}{2}$), assuming the model is scaled to fit into a unit cube. The variable N_s designates the number of spheres generated during free space exploration, N_p refers to the number of points used to represent the aMA, N_D indicates the number of distance computations performed during the expansion, and T represents the computation time of the aMA.*

Error criterion	N_{sp}	Resolution	T (seconds)
Relative	132	0.0033	17.64
Relative	72	0.0046	3.343
Absolute	N/A	0.0089	5.156
Absolute	N/A	0.0093	3.531

Table 2: *Computation time for different absolute and relative errors for the Stanford Bunny. During free space exploration, N_{sp} samples were generated on each sphere, determining the resolution of the algorithm; the overall computation time T for the aMA computation is given in seconds.*

ble 2 illustrate how the algorithm allows to trade efficiency for accuracy.

8. Conclusion

A novel approach to computing an approximated medial axis of a solid was presented. It differs from previous approaches in the following respects:

1. It computes an approximated medial axis using adaptive step sizes, resulting in unprecedented computational efficiency.
2. It permits the user to make an explicit trade-off between speed of computation and accuracy by specifying an acceptable relative or absolute error.
3. The algorithm is output sensitive. In other words, the computational cost is not determined by the complexity of the model (considering distance computation a constant time operation for a given model), but rather by the geometric complexity of the solid.
4. The computational effort expended by the algorithm in a particular local region of the solid is proportional to its local geometric complexity.

The algorithm is motivated by the fact that the medial axis of a solid D[5, 28] is the locus of points inside D, which lie at the centers of all closed discs or balls which are maximal in D and have at least two contact points with D. It proceeds by computing these maximal spheres inside D and using their radii as step sizes. Experimental results demonstrate that the proposed method allows for the efficient computation of an approximated medial axis, which (based on visual inspection) preserves the most important features of the true medial axis, while being computationally much more efficient. The algorithm has successfully been applied to a number of benchmark models, consisting of up to one million triangles.

Acknowledgments

The authors would like to acknowledge the creators of the models used in this paper: The flange mount was generated by Ristec, Inc. (Robotic Interface Systems Technologies), the Stanford Bunny was scanned and assembled by Greg Turk, the horse model is courtesy of Cyberware, Inc., and the Happy Buddha was created by Brian Curless.

Figure 8: *The Stanford Bunny (top, 69,451 triangles), its approximated medial axis (middle, computed in 17.64 seconds) and the medial axis computed by expansion of spheres with fixed radii (bottom, computed in 888.6 seconds).*

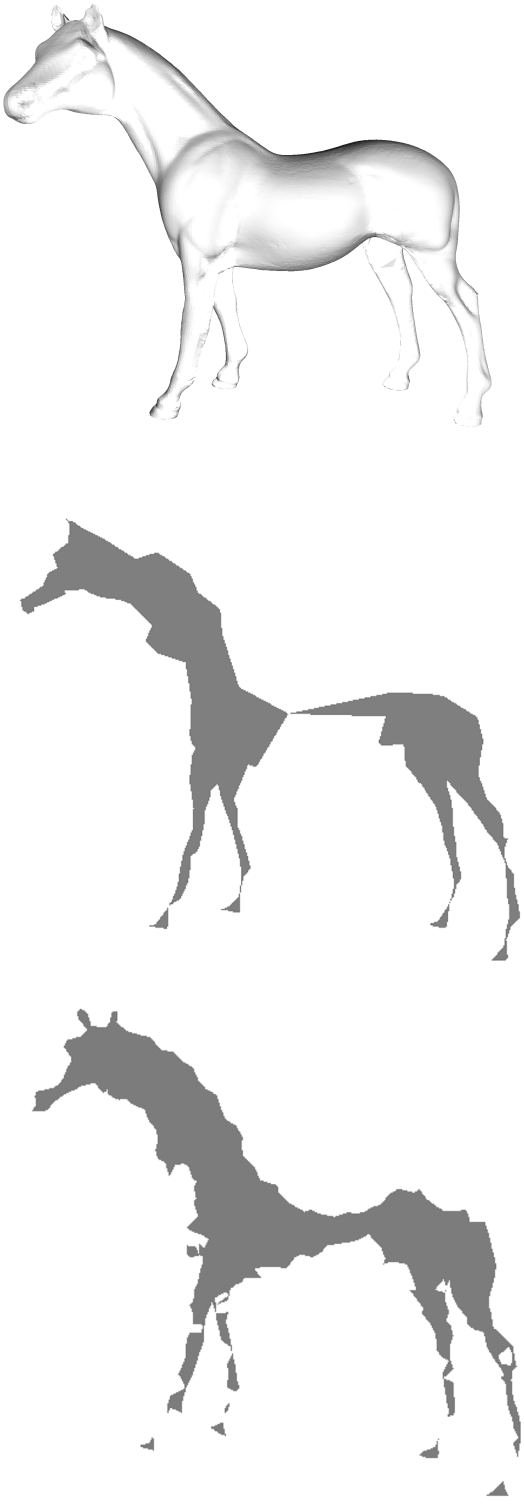

Figure 9: *The horse model (top, 96,966 triangles), its approximated medial axis (middle, computed in 13.3 seconds) and the medial axis computed by expansion of spheres with fixed radii (bottom, computed in 221.3 seconds).*

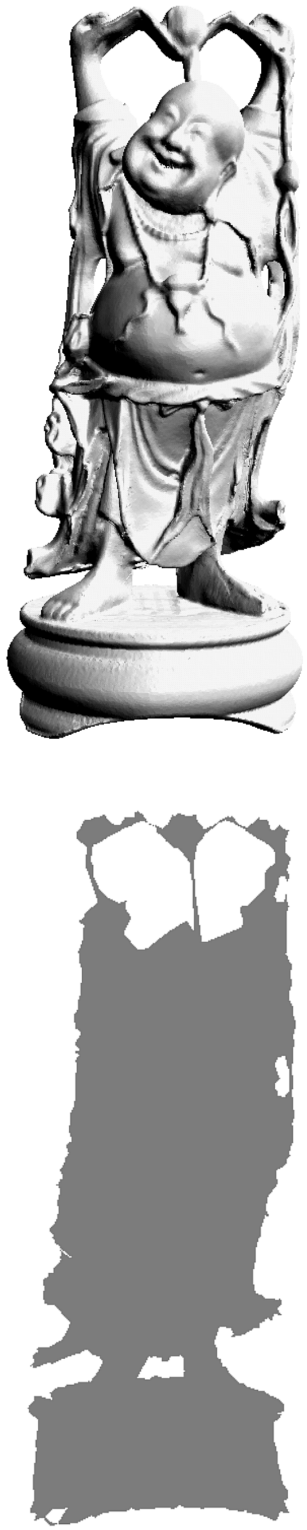

Figure 10: *The Happy Buddha model (top, 1,087,718 triangles) and its approximated medial axis (bottom, computed in 174.4 seconds).*

References

[1] S. Alla, M. Etzion, A. Rappoport, and M. Bercovier. Hexahedral mesh generation using the embedded Voronoi graph. In *7th International Meshing Roundtable*, pages 347–364, 1998. 1

[2] M. Allali. Compression on the digital unit sphere. In *16th Conference on Applied Mathematics*, pages 15–24, Jul. 2001. 5

[3] N. Amenta, S. Choi, and R. K. Kolluri. The power crust, unions of balls, and the medial axis transform. *Computational Geometry*, 19(2-3):127–153, 2001. 2, 3

[4] D. Attali and A. Montanvert. Computing and simplifying 2D and 3D continuous skeletons. *Computer Vision and Image Understanding*, 67(3):261–273, 1997. 3

[5] H. Blum. A transformation for extracting new descriptors of shapes. In Wathen-Dunn W., editor, *Models for the perception of Speech and Visual Form*, pages 362–380. MIT Press, 1967. 1, 7

[6] J. D. Boissonnat and F. Cazals. Smooth surface reconstruction via natural neighbor interpolation of distance function. In *16th Proc. Annu. ACM Symposium on Computer Geometry*, pages 185–203, 2000. 3

[7] S. Bouix and K. Siddiqi. Divergence-based medial surfaces. In *ECCV: European Conference on Computer Vision (1)*, pages 603–618, 2000. 1

[8] H. Choset. Incremental construction of the generalized Voronoi diagram, the generalized Voronoi graph, and the hierarchical generalized Voronoi graph. In *1st CGC Workshop on Computation Geometry*, 1997. 2, 3

[9] T. Culver, J. Keyser, and D. Manocha. Accurate computation of the medial axis of a polyhedron. Technical Report TR98-034, UNC-Chapel Hill, Sep. 1998. 2

[10] T. K. Dey and W. Zhao. Approximate medial axis as a Voronoi subcomplex. In *Proc. 7th ACM Sympos. Solid Modeling Applications*, pages 356–366, 2002. 2

[11] M. Foskey, M. Lin, and D. Manocha. Efficient computation of a simplified medial axis. In *Proc. ACM Symposium on Solid Modeling and Applications*, 2003. 2, 2, 3, 3, 5

[12] S. Gottschalk, M. C. Lin, and D. Manocha. OBBTree: A hierarchical structure for rapid interference detection. *Computer Graphics*, 30(Annual Conference Series):171–180, 1996. 6

[13] L. Guibas, C. Holleman, and L. Kavraki. A probabilistic roadmap planner for flexible objects with a workspace medial axis-based sampling approach. In *IEEE Int. Conf. on Intelligent Robots*, 1999. 1, 1

[14] H. N. Gursoy and N. M. Patrikalakis. Automated interrogation and adaptive subdivision of shape using medial axis transform. *Advances in Engineering Software and Workstations*, 13:287–302, 1991. 1

[15] R. H. Hardin and N. J. A. Sloane. Codes (spherical) and designs (experimental). In A. R. Calderbank, editor, *Different Aspects of Coding Theory*, volume 50, pages 179–206. AMS Series Proceedings Symposia Applied Math, 1995. 4

[16] K. E. Hoff, J. Keyser, M. Lin, D. Manocha, and T. Culver. Fast computation of generalized Voronoi diagrams using graphics hardware. *Computer Graphics*, 33(Annual Conference Series):277–286, 1999. 2, 2

[17] C.M. Hoffmann. How to construct the skeleton of csg objects. In A. Bowyer and J. Davenport, editors, *Proc. of the Fourth IMA Conference, The Mathematics of Surfaces*. Oxford University Press, 1994. 1

[18] C. Holleman and L. E. Kavraki. A framework for using the workspace medial axis in PRM planners. In *Proceedings of the International Conference on Robotics and Automation*, pages 1408–1413, 2000. 1, 1, 1, 2, 3, 3, 5

[19] L. Lam, S.-W. Lee, and C. Y. Suen. Thinning methodologies, a comprehensive survey. *Transactions on Pattern Analysis and Machine Intelligence*, 14(9):869–885, 1992. 2

[20] E. Larsen, S. Gottschalk, M. C. Lin, and D. Manocha. Fast proximity queries with swept sphere volumes. Technical Report TR99-018, Department of Computer Science, University of N. Carolina, Chapel Hill, 1999. 6

[21] V. Milenkovic. Robust construction of the Voronoi diagram of a polyhedron. In *Proc. 5th Canadian Conference on Computational Geometry*, pages 473–478, 1993. 1, 2

[22] C. Niblak, P. Gibbons, and D. Capson. Generating skeletons and centerlines from the distance transform. *CVGIP: Graphical Models and Image Processing*, 54:746–751, 1992. 1

[23] R. L. Ogniewicz. Skeleton-space: A multiscale shape description combining region and boundary information. In *Proc. Computer Vision and Pattern Recognition*, pages 746–751, 1994. 1

[24] S.M. Pizer, B.S. Morse, and D.S. Fritsch. Zoom-invariant vision of figural shape: the mathematics of cores. *Computer Vision and Image Understanding*, 69:55–71, 1998. 1

[25] I. Ragnemalm. Pattern recognition letters. *Pattern Recognition Letters*, 14(11):883–888, Nov. 1993. 2

[26] J. M. Reddy and G. M. Turkiyyah. computation of 3D skeletons using a generalized delaunay triangulation technique. *Computer-Aided Design*, 27:677–694, 1995. 1, 2

[27] D. Sheehy, C.Armstrong, and D. Robinson. Shape description by medial axis construction. *IEEE Trans. Visualization Comput. Graphics*, 2:62–72, 1996. 1

[28] E. C. Sherbrooke, N. M. Patrikalakis, and E. Brisson. Computation of the medial axis transform of 3-D polyhedral. In *Symposium on Solid Modeling and Applications*, pages 187–200, 1995. 1, 1, 2, 7

[29] K. Siddiqi, S. Bouix, A. Tannenbaum, and S. Zucker. The Hamilton-Jacobi skeleton. In *International Conference on Computer Vision (ICCV)*, pages 828–834, 1999. 2

[30] N. J. A. Sloane. http://www.research.att.com/njas/icosahedral.codes/index.html. 4

[31] D. W. Storti, G. M. Turkiyyah, M. A. Ganter, C. T. Lim, and D. M. Stal. Skeleton-based modeling operations on solids. In *Proceedings of the fourth ACM symposium on Solid modeling and applications*, pages 141–154. ACM Press, 1997. 1

[32] T. Sugimoto and M. Tanemura. Random sequential covering of a sphere with identical spherical caps. In *Fifth Interdisciplinary Symmetry Congress and Exhibition, International Society for the Interdisciplinary Study of Symmetry*, Jul. 2001. 5

[33] J. Vleugels and M. Overmars. Approximating generalized Voronoi diagrams in any dimension. Technical Report UU-CS-95-14, Dept. Comput. Sci., Utrecht Univ, 1995. 2, 2

[34] Y. Yang and O. Brock. Adapting the sampling distribution in PRM planners based on an approximated medial axis. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, New Orleans, USA, 2004. 2

[35] Y.Y. Zhang and P.S.P. Wang. Analytical comparison of thinning algorithms. *International Journal of Pattern Recognition and Artificial Intelligence*, 7:1227–1246, 1993. 2

[36] S. Zhu and A. Yuille. Forms: A flexible object recognition and modeling system. *Int. J. Comp. Vision*, 20(3):187–212, 1996. 1