Advances in Engineering Software 52 (2012) 1–9

Contents lists available at SciVerse ScienceDirect

Advances in Engineering Software

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

A fast algorithm for constructing approximate medial axis of polygons, using Steiner points

G. Smogavec*, B. Žalik

University of Maribor, Faculty of Electrical Engineering and Computer Science, Smetanova 17, SI-2000 Maribor, Slovenia

ARTICLE INFO

Article history:
Received 22 September 2011
Received in revised form 30 March 2012
Accepted 23 May 2012
Available online 1 July 2012

Keywords:
Algorithms
Polygons
Medial axis
Constrained Delaunay triangulation
Steiner points
Approximation

ABSTRACT

This paper presents a new algorithm for constructing the approximation of a polygon's medial axis. The algorithm works in three steps. Firstly, constrained Delaunay triangulation of a polygon is done. After that, the obtained triangulation is heuristically corrected by inserting Steiner points. Finally, the obtained triangles are classified into three classes, thus enabling the construction of the polygon's medial axis. As shown by experiments, the obtained approximate polygon medial axis has better properties than those methods based on a Voronoi diagram.

1. Introduction

The medial axis (or a topological skeleton) is a thinner version of a geometric object, which is equidistant from the object's boundary (see Fig. 1). Actually, if a set of maximal inscribed circles of a geometric object is constructed, the circle's centres are located on the object's medial axis (see Fig. 2).

The concept of a medial axis was introduced by Blum as a tool for biological shape recognition [1,2]. Since then, it has been used within various scientific and engineering areas, including geographic information systems [3,4], face recognition [5], path-finding [6–8], image processing [9], computer vision [10,11], collision detection [12], mesh-generation [13], machining applications [14] and others [15,16]. The efficient computation of a medial axis is of vital interest, due to its applicability. The existing solutions can be divided into two groups: namely the exact and the approximate algorithms. The exact algorithms are difficult to implement, they frequently fail to cope with holes and those working with curved boundaries suffer from numeric instability and tend to be slow as stated in [17–19]. Other methods for exact medial axis determination are based on domain decomposition [20]. The main idea is to divide the geometric object into simpler shapes until the medial axis of the domain can be trivially determined. During the merging step, the partial medial axes are joined into the final solution as stated in [21].

Approximation algorithms use the Voronoi diagram [17,22]. The Voronoi diagram is constructed on a set of points, where each Voronoi edge is equidistant to two nearest points. The Voronoi edges meet at the Voronoi vertices, which are equidistant to three (or more) input points. The set of points is obtained by uniform discretization of the geometric object's border. Upon these points, a Voronoi diagram is constructed and the Voronoi vertices are located on the medial axis [19,23] (see Fig. 3). The same approach can be used for polygons, as well as for free-form shapes. However, Brandt showed that Voronoi vertices converge to the medial axis only if the point-density defining the Voronoi diagram approaches infinity, otherwise an error can be expected [24]. An extreme scenario is shown in Fig. 4. If only the polygon vertices are used, the obtained Voronoi diagram has only one point on the medial axis (see Fig. 4a), which is completely different from the correct medial axis shown in Fig. 4b. Although fast algorithms for constructing Voronoi diagrams are known [25–28], a large number of points slows down the construction of a medial axis, as shown in Section 3.

Alternatively, some algorithms use Delaunay triangulation instead of Voronoi diagrams for approximating the medial axis. Recently, an algorithm, based on Delaunay triangulation has been introduced without implementation details [29]. Also the algorithm in [30] uses conformal Delaunay triangulation to compute a medial axis approximation.

This paper considers an algorithm for the approximate medial axis of polygons, based on constrained Delaunay triangulation. The initially-obtained triangles are analysed and, if necessary, a few artificial vertices – Steiner points [31] – are inserted onto the polygon's boundary. In the final step, the approximate medial axis

* Corresponding author. Tel.: +386 22207443; fax: +386 22207272.
E-mail address: gregor.smogavec@uni-mb.si (G. Smogavec).

0965-9978/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.advengsoft.2012.05.006

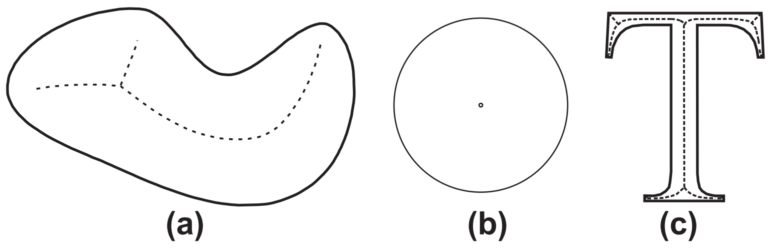

Fig. 1. Medial axes of geometric objects: a free-form shape (a), a circle (b), a character (c).

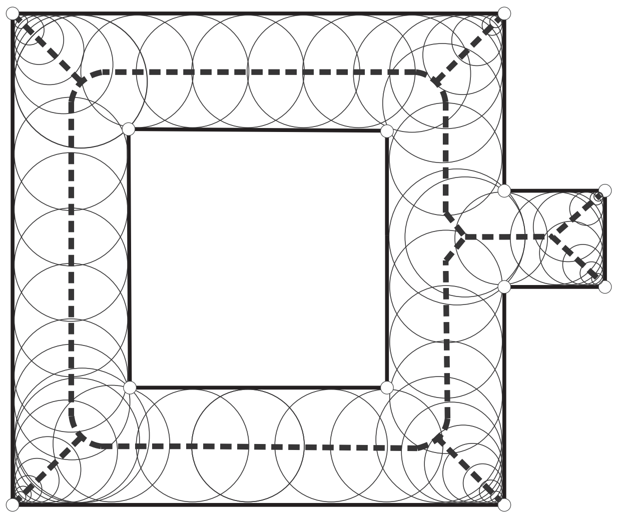

Fig. 2. The centres of maximal polygon's inscribed circles determine the polygon's medial axis (dashed-line).

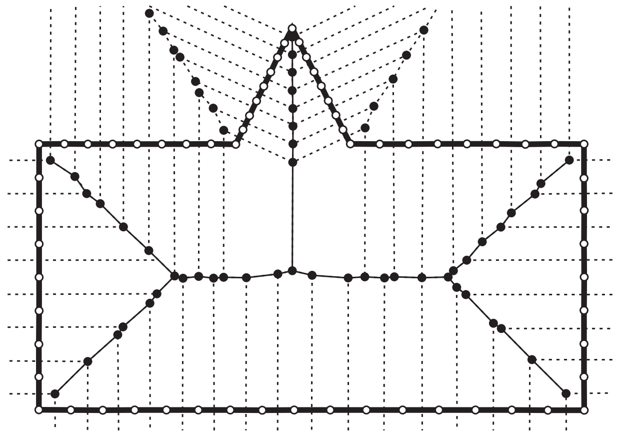

Fig. 3. Approximate medial axis (thinner solid black-line) defined by the Voronoi vertices (filled black circles). The hollow circles show the boundary nodes and the dashed-line represents the Voronoi diagram edges.

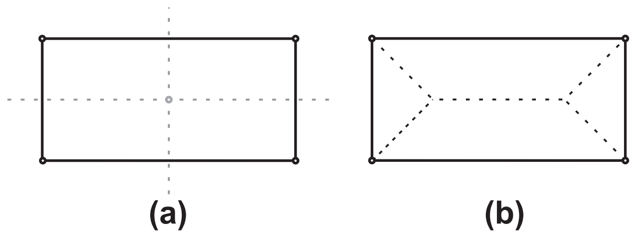

Fig. 4. Voronoi diagram (a), exact medial axis of a polygon (b).

is constructed using modified triangulation. Far less additional points are needed with this approach than with the Voronoi-based approach. According to the experiments, the obtained error against the exact medial axis is small, while the algorithm is fast and simple to implement.

This paper is organised as follows. Section 2 gives a detailed explanation of the algorithm. The results of the experiments are shown in Section 3. Section 4 summarises the paper.

2. The algorithm

As mentioned in the Introduction, approaches for constructing the approximate medial axis are usually based on the Voronoi diagram. The Voronoi diagram is a dual of Delaunay triangulation [31], where the centre of a Delaunay triangle's circumcircle coincides with the Voronoi vertex (see Fig. 5). Consequently, the centres of the Delaunay triangle's circumcircles are also placed on the medial axis.

The constrained Delaunay triangulation [32] of the considered polygon is applied in order to reduce the number of vertices that have to be additionally inserted. The constrained Delaunay triangulation may violate the Delaunay criteria and, therefore, only some of the centres of the triangle's circumcircles may be positioned on the medial axis. For this reason, the obtained polygon triangulation is heuristically modified in our case. The proposed algorithm works in three steps:

- constrained Delaunay triangulation is applied on the polygon,
- the obtained triangulation is locally modified by inserting Steiner points, and
- an approximation of the medial axis is constructed.

After constructing the constrained Delaunay triangulation (the algorithm introduced in [33] is used), the obtained triangles are

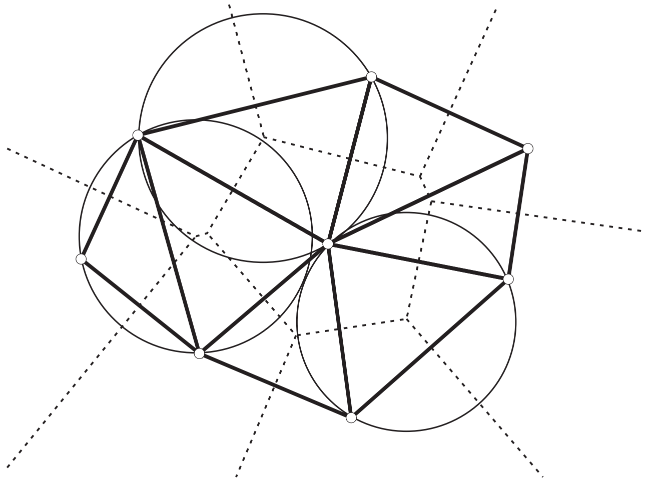

Fig. 5. The Voronoi diagram is a dual of Delaunay triangulation.

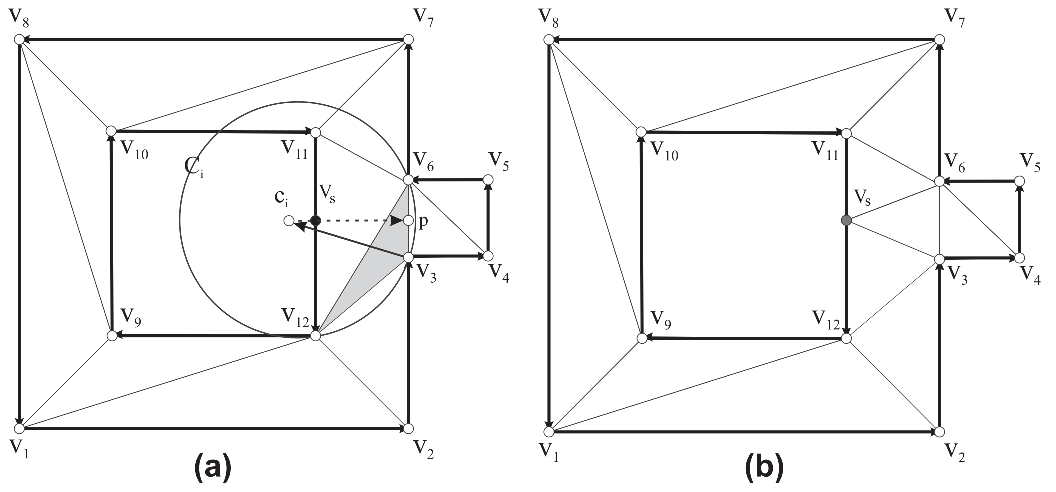

Fig. 6. Inserting a Steiner point to remove an obtuse triangle.

inspected. Special care is given to those triangles with obtuse inner angles and to the triangles rising from the convex polygon vertices.

2.1. Local modification of obtuse triangles having three neighbours

This subsection focuses on those obtuse triangles having three neighbours. An obtuse triangle has the centre $\boldsymbol{c_i}$ of its circumcircle outside the triangle (dark triangle in Fig. 6a). A ray is sent from the obtuse vertex (vertex v_3 in Fig. 6a) towards $\boldsymbol{c_i}$. Only those triangles, intersected by the ray, are inspected (usually, only the neighbouring triangle over the longest edge of the considered obtuse triangle). If the ray does not intersect any polygon edge, $\boldsymbol{c_i}$ is inside the polygon, and the triangle is considered as acceptable. Otherwise, a Steiner point is inserted at the intersected polygon edge as follows: Firstly, a common vertex v_o is found that is on the intersected polygon edge and is a member of the obtuse triangle (v_o is v_{12} in Fig. 6a). Two triangle edges define the considered obtuse angle (edges v_3v_6 and v_3v_{12}). The edge that does not contain vertex v_o (edge v_3v_6 in Fig. 6a) is selected. Its middle point p is calculated and vector $\vec{c_ip}$ is formed. Its intersection with the previously determined polygon edge is calculated. This intersection determines the position of the Steiner point denoted by v_s in Fig. 6a. After that this part of the polygon is re-triangulated, as shown in Fig. 6b.

The triangulation refinement process described in this subsection only modifies those polygon triangles that have three neighbours and their circumcentre outside the polygon. This step excludes any checking of Delaunay criterion.

2.2. Triangles defined by polygon convex vertices

The medial axis always derives from the polygon's convex vertices (see the example in Fig. 4b). Because of this, all the triangles in convex vertices have to be modified. The situation shown in Fig. 7a shows the obtained polygon triangulation. Vertices v_1 and v_3 define two triangles. The continuation describes the process of inserting Steiner points.

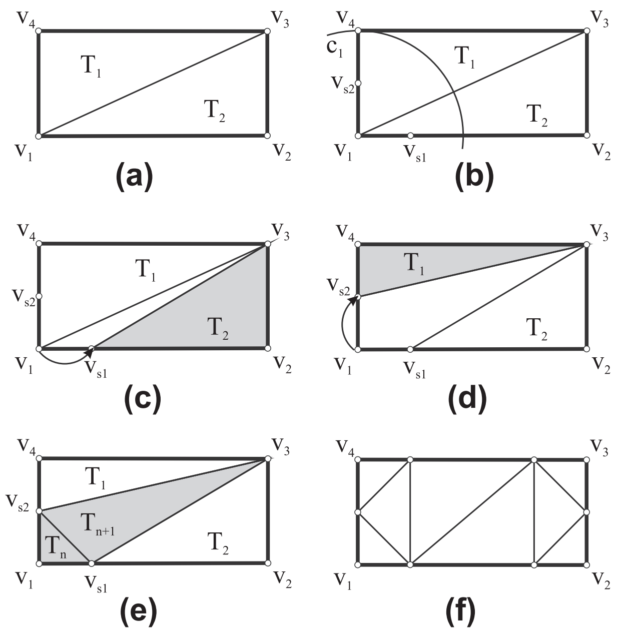

Fig. 7. Correcting triangulation at the polygon convex vertices.

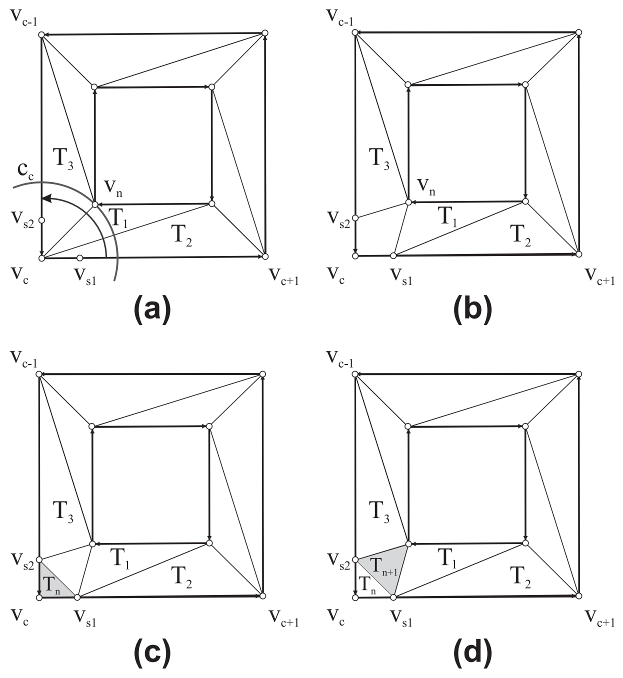

Fig. 8. Correcting triangulation at the polygon convex vertex, when the nearest vertex is not on the same polygon loop.

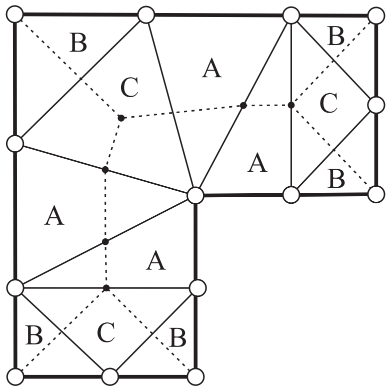

Fig. 9. Triangles are classified as types A, B, and C.

Let us observe Fig. 7b and denote the considered convex vertex with v_c (in Fig. 7b the $v_c = v_1$). Firstly, a list of triangles originating in v_c is formed. These triangles are sorted in a counterclockwise direction around vertex v_c in order to form a triangle strip T_s (in Fig. 7b $T_s = T_2, T_1$). The nearest polygon vertex v_n in regard to vertex v_c is found ($v_n = v_4$), by sequentially calculating the distances to all the triangle vertices in T_s. If two or more vertices have the same minimal distance, the first encountered is selected. The distance $d = |v_c v_n|$ represents the radius of circle c_c, the centre of which is in vertex v_c. Two Steiner points v_{s1} and v_{s2} are inserted at the intersections between circle c_c with halved-radius and polygon edges terminating in vertex v_c (see Fig. 7b). The triangles from T_s are now changed as follows. Firstly, the bisector of $\angle v_{c-1} v_c v_{c+1}$ ($\angle v_4 v_1 v_2$) is determined and after that, the angle at vertex v_c is calculated (for each triangle from T_s). The triangles from T_s are processed sequentially in a counterclockwise direction. Until the cumulative value

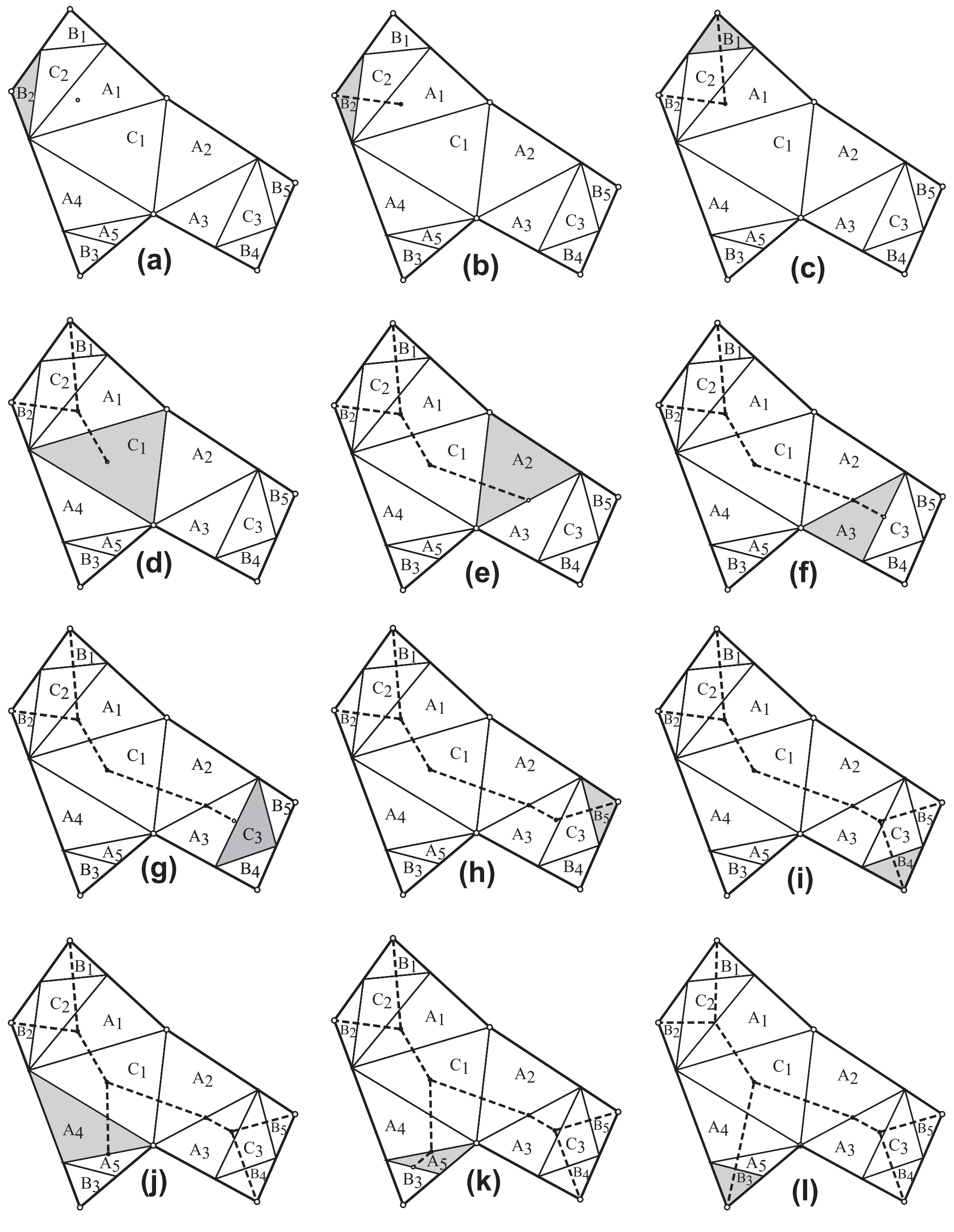

Fig. 10. Constructing an approximation of a polygon's medial axis.

of the angles at vertex v_c is smaller than the half of $\angle v_{c-1}v_cv_{c+1}$, the triangle vertex is moved from vertex v_c to vertex v_{s1} (see Fig. 7c). Otherwise, if the cumulative value of the angles at vertex v_c is larger or equal to the bisector of $\angle v_{c-1}v_cv_{c+1}$, the triangle vertex v_c is moved to vertex v_{s2} (see Fig. 7d).

After these operations, a gap is obtained, which is triangulated by adding two new triangles: $\Delta v_cv_{s1}v_{s2}$ and $\Delta v_{s2}v_{s1}v_3$ (Fig. 7e). Fig. 7f represents the obtained triangulation when all the convex vertices have been processed.

However, it is not necessary that the nearest vertex v_n is on the same polygon-loop as the convex vertex v_c. In this case, the algorithm differs when triangulating the gap. Let us observe the convex vertex v_c in Fig. 8a. A gap is obtained, after sorting the triangles in counterclockwise order ($T_s = T_2$, T_1, T_3), finding the nearest vertex v_n, and then moving the triangles (see Fig. 8b). This gap is triangulated by adding two new triangles $\Delta v_cv_{s1}v_{s2}$ (Fig. 8c) and $\Delta v_nv_{s1}v_{s2}$ (Fig. 8d).

2.3. Creating the approximate medial axis

After an adequate polygon triangulation is obtained, the medial axis is constructed according to the three different types of triangles (see Fig. 9):

- triangles coinciding with one polygon edge, are considered as type A,
- triangles coinciding with two polygon edges, are referred as type B, and
- triangles that do not coincide with any polygon edge, represent type C.

Triangles that coincide with all three polygon edges (i.e. the polygon is triangular) represent a special case, the solution of which is trivial.

The triangles of type C branch the medial axis, triangles of type B derive the medial axis from the polygon's convex vertices, whilst triangles of type A contribute one straight line connecting those two triangle's edges that do not belong to the polygon. The whole process of medial axis construction is done recursively, as explained in the continuation.

The algorithm starts in an arbitrary C-type triangle (such a triangle always exists due to the Steiner point insertion). Firstly, it is checked as to whether the centre of the triangle's circumcircle is inside the triangle. Let us suppose that the centre is inside. In this case the centre represents the vertex on the medial axis and the algorithm recursively visits the neighbouring triangles. If an A-type of triangle is encountered, the last vertex obtained on the medial axis is connected to the bisector of the common edge with the next neighbouring triangle. If a triangle of B-type is met, all the line segments between the considered B-type triangle and the last C-type triangle, generated by A-type triangles, are firstly removed from the solution. After that, the circumcentre of the last C-type triangle

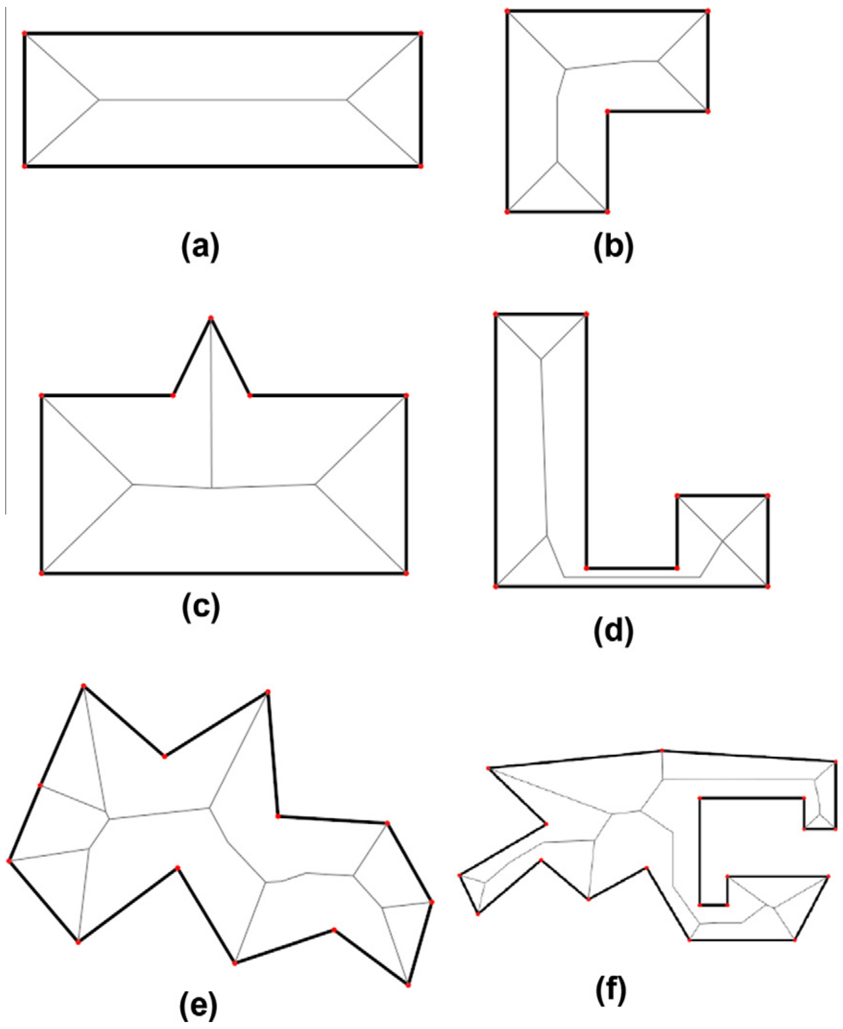

Fig. 11. Examples of approximate medial axis generated by the proposed algorithm. Number of vertices: 4 (a); 6 (b); 7 (c); 8 (d); 13 (e); 19 (f).

is connected to the B-type triangle's top vertex (i.e. with the polygon's convex vertex).

If the centre of the C-type triangle's circumcircle is outside the triangle (the triangle is obtuse), a vector is sent from the triangle's obtuse vertex to the circle's centre. All the triangles of type A, pierced by this vector, are marked as *disabled* and do not participate in the final solution. The algorithm then recursively visits the neighbouring triangles and upon each individual triangle type, adds a contribution to the final MA, as described above.

Let us consider an example and start with triangle C_2 (see Fig. 10a). Its circumcentre is in triangle A_1. Because of this, triangle A_1 is disabled and does not contribute to any further steps of the algorithm. From C_2, triangle B_2 is visited. The last inserted point on the medial axis (the circumcentre of triangle C_2) is connected with the top vertex of B_2 (see Fig. 10b). The algorithm recursively returns to C_2 and turns into triangle B_1 (see Fig. 10c). The same procedure as before is applied, and the algorithm returns to triangle C_2. The remaining neighbour is visited (triangle A_1). However, it has already been marked as disabled, and therefore it is skipped and triangle C_1 is reached (see Fig. 10d). The C_1 circumcentre is connected with the last inserted point on the medial axis, i.e. with the circumcentre of triangle C_2. The algorithm visits triangle A_2. The last inserted medial axis point (circumcentre of C_1) is connected with the midpoint of the edge between A_2 and A_3 (see Fig. 10e). The algorithm proceeds to A_3, where the same procedure is repeated (see Fig. 10f). The next triangle is C_3. Its circumcentre is inside triangle A_3. Because of this A3 is disabled. As triangle A_3 has already been processed, its contribution to the medial axis is removed, and the midpoint of the edge between triangles A_2–A_3 is connected with the circumcentre of triangle C_3 (see Fig. 10g). From C_3, the algorithm moves to triangle B_5. The last inserted point (the circumcentre of triangle C_3) is connected with the top vertex of triangle B_5 (see Fig. 10h). Similarly, triangle B_4 is processed (see Fig. 10i). The recursion returns to triangle C_1. The only unvisited triangle (triangle A_4) is reached. A line segment starting in the circumcentre of triangle C_1 to the midpoint of the edge between triangles A_4 and A_5 (see Fig. 10j) is added to the medial axis. The algorithm enters triangle A_5 and adds a new line segment (see Fig. 10k). The next triangle is triangle B_3. All the contributions of the A-type triangles between B_3 and triangle C_1 are removed and the circumcentre of triangle C_1 is connected with the top vertex of triangle B_3. Recursion terminates and the process of the medial axis is complete (the result is shown in Fig. 10l).

Examples of approximate medial axis construction are shown in Figs. 11–13. It should be noted that this algorithm also solves polygons with holes.

3. Analysis of the algorithm

The proposed approximate algorithm for medial axis construction was analysed regarding time complexity, computational efficiency, and error, caused by the approximation.

3.1. Time-complexity

The algorithm consists of three steps. The constrained Delaunay triangulation, constructed during the first step, has been intensively studied in the past. According to [33] the constrained Delaunay triangulation is solved in $O(n \log n)$ time, where n is the number of vertices. In the second step, constrained Delaunay triangulation has to be improved according to the procedures described

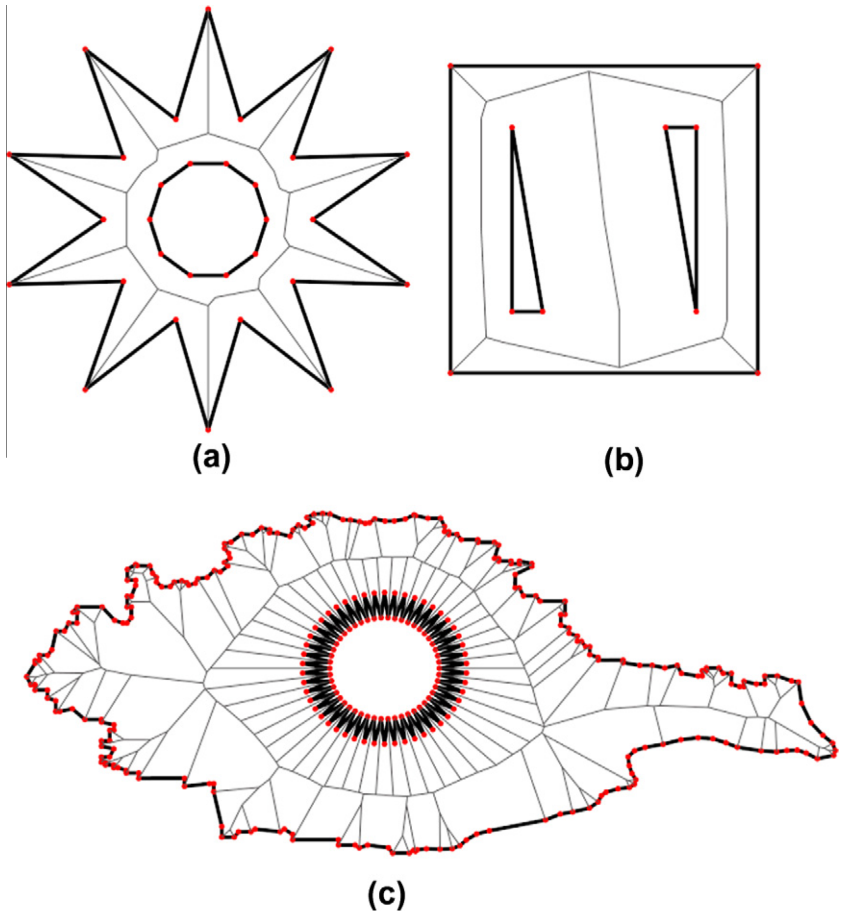

Fig. 12. Examples of approximate medial axis generated by our algorithm on polygons with holes. Number of vertices: 30 (a); 12 (b); 341 (c).

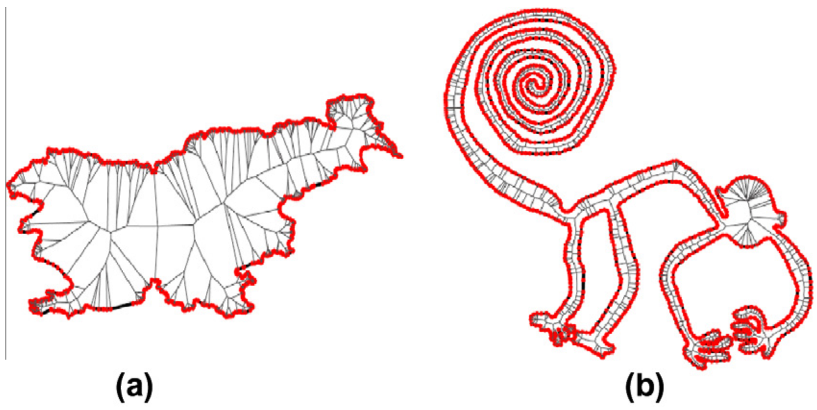
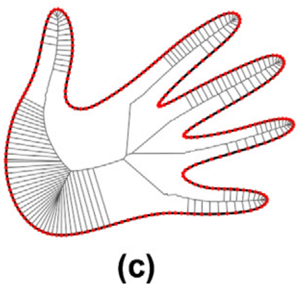

Fig. 13. Examples of approximated medial axis for more complicated polygons. Number of vertices: 997 (a); 9632 (b); 270 (c).

in Section 2. The worse case occurs at the convex polygon because correction is needed in all its vertices. In this case $2n$ new vertices are inserted. As a consequence, $3n - 2$ triangles are obtained in total. All these operations are performed locally and do not include any searches. Therefore, this step is terminated in $O(n)$ time. In the last step, an approximated medial axis is generated. The algorithm visits triangles using established neighbourhood relationships, therefore, this step is also terminated in $O(n)$. The time complexity of the algorithm is therefore $O(n \log n)$ and depends on the used constrained Delaunay triangulation algorithm.

3.2. Spent CPU time

In order to estimate the efficiency of the proposed approximate algorithm (AMA – approximated medial axis), a comparison was made with an exact algorithm (EMA, exact medial axis) obtained at [34] and implemented according to [35]. It should be noted that the used exact algorithm implementation could not process polygons containing holes, therefore, only polygons without them were used. Firstly, the computational efficiencies of both algorithms were compared. Table 1 shows the spent CPU time using polygons (see Fig. 14) of different types and different numbers of points.

As can be seen from Table 1, the proposed algorithm is considerably faster that the referenced algorithm. As explained in the previous section, the presented algorithm firstly constructs constrained Delaunay triangulation. This step is also computationally the most demanding. According to the experiments, the constrained Delaunay algorithm takes between 60% and 95% of the total CPU time (see Table 2).

From the results in Table 1, it can be seen that the actual spent CPU time depends considerably on the polygon type. Let us recall

Table 1
Spent CPU time (ms) of an exact (EMA) and the proposed approximate algorithm (AMA).

No. of vertices	Algorithm	Polygon type		
		Convex	Concave	Star-shaped
100	EMA	3.422	3.235	4.235
	AMA	1.707	2.967	2.705
200	EMA	7.204	15.297	9.657
	AMA	4.054	4.116	3.938
500	EMA	28.504	42.063	23.688
	AMA	7.690	5.135	7.939
1.000	EMA	44.975	63.846	48.642
	AMA	15.966	6.837	11.711
2.000	EMA	96.909	87.159	100.487
	AMA	29.973	12.648	29.515

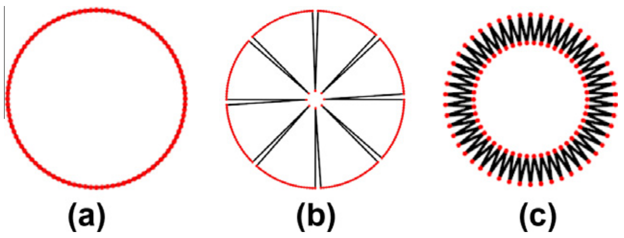

Fig. 14. Input polygons used for the measurement of spent CPU time: convex (a), concave (b) and star-shaped (c).

Table 2
Spent CPU time (ms) for only constrained Delaunay triangulation.

No. of vertices	Polygon type		
	Convex	Concave	Star-shaped
100	1.55	2.75	2.35
200	3.85	3.85	3.05
500	7.05	4.65	4.70
1.000	14.85	5.50	7.15
2.000	27.35	10.15	20.45

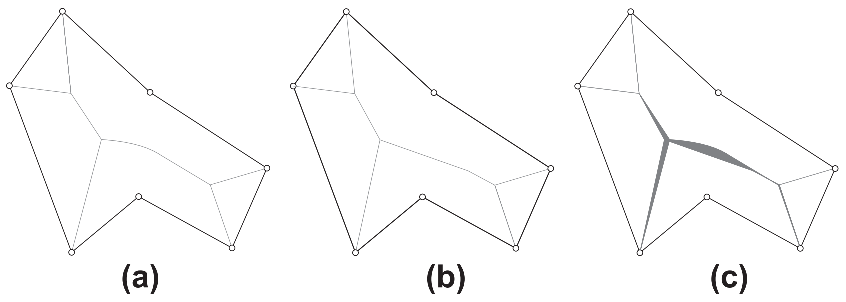

Fig. 15. Example of EMA (a), AMA (b) and the area between AMA and EMA (c).

Table 3
Comparisons between the AMA and Voronoi diagram.

Polygon	No. of init. vertices	AMA			VDA		
		No. of Steiner points	Execution times (ms)	Error (%)	No. of vertices	Execution times (ms)	Error (%)
Fig. 12a	4	6	0.64	0.0	4	0.11	100
					16	0.64	5.59
					64	5.687	0.5
					256	48.812	0.04
Fig. 12b	6	8	0.906	1.09	6	0.188	17.366
					12	0.406	6.38
					24	1.141	1.46
					48	3.718	0.46
Fig. 12c	7	8	0.908	0.12	8	0.219	4.13
					16	0.656	3.43
					32	1.968	1.20
					64	6.422	0.239
Fig. 12d	8	9	1.22	2.76	8	0.265	12.404
					16	0.703	11.548
					32	2.094	5.311
					64	6.766	2.88
Fig. 12e	13	14	1.64	0.69	12	0.485	5.75
					24	1.359	2.73
					48	4.156	1.14
					96	14.797	0.5
Fig. 12f	19	21	1.74	4.31	19	0.763	11.259
					38	2.296	10.11
					76	7.655	4.66
					152	27.835	2.96

that constraint Delaunay triangulation is used in the first part of our algorithm. The sweep-line constraint Delaunay triangulation introduced in [33] is used when the sorting of the vertices is done at the beginning. According to [36], the classical quicksort algorithm exposes different CPU times for different polygon types, which is also reflected in the proposed algorithm.

3.3. Estimation of the approximation quality

The most important information about any approximate algorithm is how good the obtained solution is in regard to the exact solution. In the continuation, the accuracies of the proposed approximation algorithm (AMA) and the Voronoi diagram approximation approach (VDA) were considered, against the exact medial axis (EMA). Implementation of the Voronoi diagram obtained at [37] was used.

In order to estimate the accuracy of our approximation, the area between the exact A_{EMA} (see Fig. 15a) and the approximate A_{AMA} (see Fig. 15b) medial axis was calculated (see Fig. 15c) and then divided by the area A of the whole polygon (Eq. (1)).

$$Error = \frac{A_{EMA} - A_{AMA}}{A} \quad (1)$$

The quality of Voronoi-based approximation depends on the density of a polygon's vertices. The lower the density, the lower the quality of the obtained solution and vice versa. In our experiments, the polygon's edges were split in half until the obtained error became similar to the error generated by our algorithm. However, in this case, the CPU time required by the Voronoi-based algorithm was considerably longer. The results of the experiments are summarised in Table 3. As shown, the error obtained by the proposed algorithm did not pass 6%, in the worse case. In the best case it gave an exact solution for the medial axis (see Fig. 11a).

All the measurements were done on a PC with an Intel Core 2 Duo E6300 1.86 GHz processor, 2 GB RAM and running Windows XP operating system.

4. Conclusion

This paper introduced a new method for constructing the approximated medial axis of simple polygons with or without holes. The algorithm consists of three steps. Firstly, the constrained Delaunay triangulation is constructed. The obtained triangulation is heuristically corrected by inserting Steiner points on some polygon edges. The triangles are then classified into three types. According to the triangle type, the approximated medial axis is constructed using local heuristics. The obtained approximated medial axis is fairly compatible with the exact medial axis. According to the experiments, the presented method outperforms Voronoi-base approaches.

Acknowledgment

This work was supported by the Slovenian Research Agency under the Project P2-0041.

References

[1] Blum H. A transformation for extracting new descriptors of shapes. Models for perception of speech and visual form. MIT-Press; 1967. p. 362–80.
[2] Blum H. Biological shape and visual science (Part 1). J Theor Biol 1973;38(February):205–87.
[3] Lee J. A Spatial Access-oriented implementation of a 3-D GIS topological data model for urban entities. Geoinformatica 2004;8(September):237–64.
[4] Mena JB. Automatic vectorization of segmented road networks by geometrical and topological analysis of high resolution binary images. Knowl-Based Syst 2006;19(July):704–18.
[5] Hallinan P, Gordon G, Yuille A, Giblin P, Mumford D. Two and three dimensional patterns of the face. AK Peters/CRC Press; 1999.
[6] Geraerts R, Overmars MH. Clearance based path optimization for motion planning. Robot Autom 2004;3:2386–92.
[7] Geraerts R, Overmars M. Creating high-quality roadmaps for motion planning in virtual environments. Intell Robots Syst 2006:4355–61.
[8] Guibas L, Holleman R, Kavraki LE. A probabilistic roadmap planner for flexible objects with a workspace medial axis based sampling approach. Intell Robots Syst 1999;1:254–9.

[9] Sonka M, Hlavac V, Boyle R. Image processing, analysis, and machine vision. 2nd ed. Thomson Learning and PT Press; 2003.
[10] Gooch B, Coombe G, Shirley P. Artistic vision: painterly rendering using computer vision techniques. In: Proceedings of the 2nd international symposium on non-photorealistic animation and rendering, France; June 2002.
[11] Sheehy D, Armstrong C, Robinson D. Shape description by medial surface construction. Visual Comput Graph 1996;2:62–72.
[12] Hubbard P. Approximating polyhedral with spheres for time critical collision detection. ACM Trans Graph 1996;15:179–210.
[13] Sheffer A, Etizon M, Rappoport A, Bercovier M. Hexahedral mesh generation using the embedded Voronoi graph. Eng Comput 1999;15(September): 248–262.
[14] Gershon E, Elaine C, Sam D. MATHSM: medial axis transform toward high speed machining of pockets. Comput Aided Des 2005;37(February):241–50.
[15] Gibblin PJ, Kimia BB. A formal classification of 3D medial axis points and their local geometry. Pattern Anal Mach Intell 2004;26(January):238–51.
[16] Teichmann M, Teller S. Assisted articulation of closed polygonal models. In: ACM SIGGRAPH 98 conference; July 1998. p. 254.
[17] Dey TK, Zhao W. Approximate medial axis as a Voronoi subcomplex. Comput Aided Des 2004;36(February):195–202.
[18] Foskey M, Lin MC, Manocha D. Efficient computation of a simplified medial axis. J Comput Inf Sci Eng 2003;3(December):274–85.
[19] Dorado R. Medial axis of a planar region by offset self-intersections. Comput Aided Des 2009;41(December):1050–9.
[20] Choi H, Choi S, Moon H. Mathematical theory of medial axis transform. Pac J Math 1997;181:57–88.
[21] Aichholzer O, Aigner W, Aurenhammer F, Hackl T, Jüttler B, Rabl M. Medial axis computation for planar free-form shapes. Comput Aided Des 2009;41(May): 339–349.
[22] Aurenhammer F, Klein R. Voronoi diagrams. In: Sack J-R, Urrutia J, editors. Handbook of computational geometry; 2000. p. 201–90.
[23] Dey TK, Zhao W. Approximating the medial axis from the Voronoi diagram with a convergence guarantee. Algorithmica 2004;38:179–200.
[24] Brandt JW. Convergence and continuity criteria for discrete approximation of the continuous planar skeletons. CVGIP: Image Understan 1994;59(January): 116–124.
[25] Jin L, Donguk K, Lisen M, Deok-Soo K, Shi-Min H. A sweepline algorithm for Euclidean Voronoi diagram of circles. Comput Aided Des 2006;38(March): 260–272.
[26] Held M, Hubera S. Topology-oriented incremental computation of Voronoi diagrams of circular arcs and straight-line segments. Comput Aided Des 2009;41(May):327–38.
[27] Kalra N, Ferguson D, Stentz A. Incremental reconstruction of generalized Voronoi diagrams on grids. Robot Auton Syst 2009;57(February):123–8.
[28] Fortune SJ. A sweepline algorithm for Voronoi diagrams. Algorithmica 1987;2:153–74.
[29] <http://sog1.me.qub.ac.uk/Research/medial/medial.php> [05.09.11].
[30] Linardakis L, Chrisochoides N. Delaunay decoupling method for parallel guaranteed quality planar mesh refinement. SIAM J Sci Comput 2006;27: 1394–1423.
[31] De Berg M, Van Kreveld M, Overmars M, Schwarzkopf O. Computational geometry, algorithms and applications. Springer-Verlag; 1997.
[32] Maur P, Kolingerova I. The employment of regular triangulation for constrained Delaunay triangulation. Comput Sci Appl 2004;3:198–206.
[33] Domiter V, Žalik B. Sweep-line algorithm for constrained Delaunay triangulation. Int J Geogr Inf Sci 2008;22:449–62.
[34] <http://www.lupinho.net/lupinho.de/gishur.html> [20.03.12].
[35] Lee DT. Medial axis transformation of a planar shape. Pattern Anal Mach Intell 1982;4(July):363–9.
[36] Podgorelec D, Klajnšek G. Acceleration of sweep-line technique by employing smart quicksort. Inf Sci 2005;169(February):383–408.
[37] <http://www.skynet.ie/~sos/mapviewer/voronoi.php> [20.03.12].