Unit-width curve skeleton generation based on 3D thinning and shortest path finding

Tao Wang

Abstract

3D thinning algorithms are generally faster than any other connectivity-preservation skeletonization methods. However, most 3D thinning algorithms cannot guarantee to generate unit-width skeleton. This document describes an algorithm to generate unit-width skeleton based on 3D thinning and shortest path finding.

1. Introduction

Skeletonization is a useful technique that has potential applications in a wide variety of problems. Skeletonization is also known as skeletonizing or topological skeleton generation. It creates a compact representation (*skeleton*) of the models that may be used for further processing. The skeletons can be defined via the *medial axis transformation* (MAT) [1]. The MAT can be computed by "prairie fire" propagation. Consider an object as a prairie of uniform and dry grass. Suppose that a fire is lit along its border. All fire fronts advance into the object at the same speed. The MAT of the region is the set of points reached by more than one fire front at the same time.

In 3D space, a medial axis consists of the loci of the centers of all inscribed maximal spheres of the 3D model, where these spheres share at least two points with the boundary of the model. A skeleton in 3D consists of a set of 2D surfaces and/or 1D line segments. In many applications, a skeleton with only 1D line segments is more desirable. We call it 3D curve skeleton. In this document, we are interested in extraction of 3D curve skeleton.

Figure 1 (b) shows a 3D box and its skeleton with 2D surfaces and 1D line segments. Figure 1 (c) shows the same 3D box and its curve skeleton with 1D line segments only.

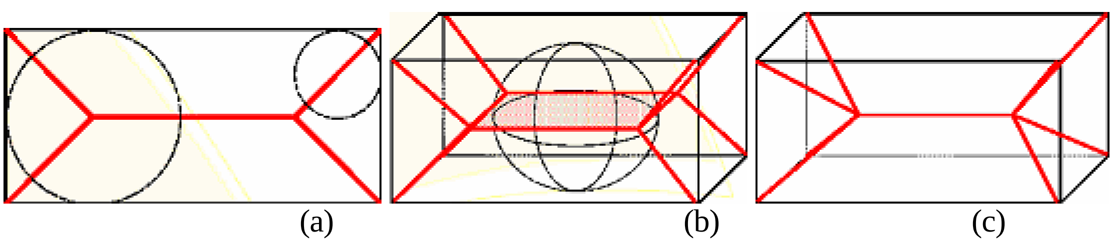
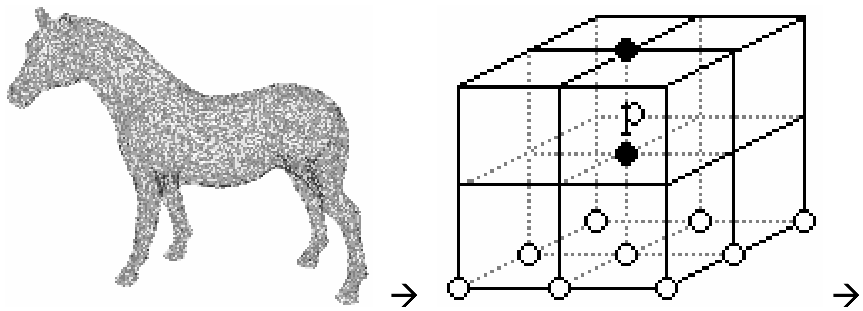


Figure 1: (a) A rectangle and its skeleton in 2D (consisting of 5 line segments), shown with representative maximal circles and contact points; (b) 3D box and its skeleton with 2D surfaces and 1D line segments. (c) 3D box and its curve skeleton with 1D line segments only. Images courtesy of Cornea [2].

2. 3D thinning

There are four kinds of skeletonization techniques in the literature: thinning based algorithms [3-5], general field based algorithms [6-8], Voronoi diagram based algorithm [9-11], and shock graph based algorithm [12-14]. Connectivity-preservation is an important requirement for skeletonization algorithms. In general, thinning based algorithms and Voronoi diagram based algorithms can preserve connectivity of the 3D model. Efficiency is another important requirement for skeletonization algorithms. In general, thinning based algorithms and general field based algorithms are faster than other algorithms. In this document, we are only interested in thinning based algorithm.

A *3D thinning* algorithm applies in a local neighborhood of an object point and iteratively removes object points that satisfy some pre-defined masks to generate skeletons in a *3D binary image*. A 3D binary image is a mapping that assigns the value of 0 or 1 to each point in the 3D space. Points having the value of 1 are called *black* (*object*) points, while 0's are called *white* (*background*) ones. Black points form objects of the binary image. The thinning operation iteratively deletes or removes some object points (that is, changes some black points to white) until only some restrictions prevent further operation. Figure 2 shows the basic idea of a 3D thinning algorithm [15] that generates curve skeletons. In Figure 1 (b), a "•" is used to denote an object point while a "∘" is used to denote a background point. An unmarked point is a "don't care" point, which can represent either an object point or a background point.



(a) (b) (c)

Figure 2: (a) A 3D binary volumetric model; (b) a deleting mask; (c) the curve skeleton, highlighted with red.

3. Unit-width curve skeleton

Comparing to other skeletonization techniques, 3D thinning algorithms have two advantages: connectivity guarantee and efficiency. However, most 3D thinning algorithms [3-5] cannot guarantee to generate unit-width curve skeletons. For instance, in Figure 1 (c), there are some regions on the skeleton swarmed with many points. This is an undesired property for many applications that require curve skeleton as an input. The following definitions are used to formally define unit-width curve skeleton.

Definition 1: The ***degree*** of a point is defined as the number of points in its 26-neighborhood.
Definition 2: An ***end point*** is point that has a degree of 1.
Definition 3: A ***middle point*** is point that has a degree of 2.
Definition 4: A ***joint point*** is point that has a degree of n ($n > 2$) and all the n neighbors are either end points or middle points.
Definition 5: A ***crowded joint point*** is point that has a degree of n ($n > 2$) and at least one of its neighbors are neither end points nor middle points.
Definition 6: 26-connected crowded joint points form a ***crowded region***.
Definition 7: A skeleton is a ***unit-width curve skeleton*** if and only if it has no crowded region.

To create unit-width curve skeleton, all the crowded joint points have to be removed. In [16], an iterative algorithm was used to merge crowded joint points to create unit-width curve skeleton. The basic idea of this algorithm is to merge two crowded joint points with minimal cost. However, this algorithm cannot be applied to create unit-width curve skeletons generated by most 3D thinning algorithms because it works in non-equilateral 3D grid while most 3D thinning algorithms work in equilateral 3D grid (3D image). Moreover, this algorithm is not efficient. It calculates the costs of different merging options and merges the two crowded adjacent joint points with minimal cost in each iteration. If there are E crowded joint points and V edges between them, the complexity of this algorithm is $O(E*V^3)$. One example of this algorithm is shown in Figure 3.

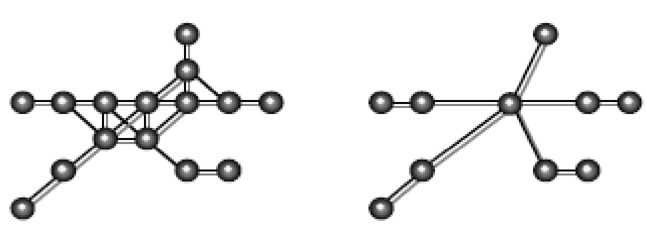


Figure 3: The left graph contains crowded joint points between the curve skeleton as edges. In the right graph, some crowded joint points are merged in order to create unit-width curve skeleton [16].

Sundar *et al* [19] use a clustering method to reduce the number of points on a curve skeleton. The basic idea is to use one point to replace a cluster of points that are within distance $D_{threshold}$ of the point. This algorithm has two drawbacks. First, different clusters have different $D_{threshold}$. How to determine different $D_{threshold}$ for different clusters remains unsolved. Second, in most cases, a cluster is not fully connected. Therefore, this algorithm may disconnect the skeleton by choosing a point that is not connected with other points on the skeleton.

To design an algorithm that works in 3D binary volumetric model and preserves connectivity, we use the Dijkstra shortest path finding algorithm [17-18] to create unit-width curve skeletons. This algorithm has three advantages:

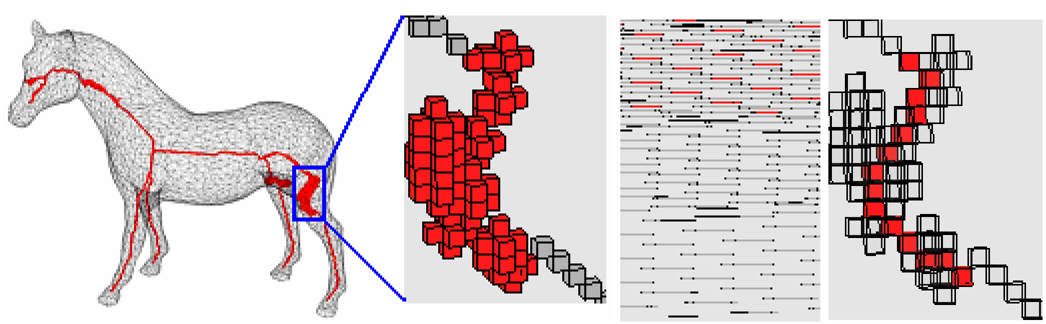
1. It works in 3D binary volumetric models
2. It preserves connectivity
3. It does not need any free parameters such as $D_{threshold}$

The pseudo code of our algorithm is as follows:
Input: non-unit-width curve skeleton *I* in a 3D binary volumetric model
Output: unit-width curve skeleton *O* in the 3D binary volumetric model
Algorithm unit_width_curve_skeleton (*I*)

1. **Initialize:** Initialize output *O* and copy *I* to *O*.
2. **Compute degrees:** For each point on the skeleton *O*, calculate the degree of each point.
3. **Classify points:** According to the degree of each point, mark end points, middle points, joint points, and crowded joint points. If there is no crowded joint points, output *O* and exit.
4. **Locate crowded regions:** Organize crowded joint points into crowded regions. Each crowded region consists of 26-adjacent crowded joint points.
5. **Detect exits:** In each crowded region, find all the end points and middle points that are 26-adjacent to this crowded region, and mark each as "exit".
6. **Determine the centroid:** In each crowded region, determine the centroid. If there are N points (x_1, y_1, z_1), (x_2, y_2, z_2), …, (x_N, y_N, z_N) in a crowded region, the centroid a point in this region and is closest to $(\sum_{i=1}^{N} x_i / N, \sum_{i=1}^{N} y_i / N, \sum_{i=1}^{N} z_i / N)$.
7. **Find shortest path:** In each crowded region, apply Dijkstra shortest path finding algorithm to find shortest path between the centroid and each "exit". Remove the crowded joint points that are not on any of the shortest paths from *O*.
8. **Output:** Output the unit-width curve skeleton *O*.

End algorithm

If there are E crowded joint points and V edges between them, the complexity our new algorithm is $O(E+V^2)$ by taking advantage of the Dijkstra shortest path finding algorithm. Since shortest path found by Dijkstra shortest path finding algorithm is unique, connected and has no circles, the skeleton generated by this approach is unit width. Figure 4 shows the basic idea of this algorithm. More examples of unit-width curve skeleton generated by this algorithm are shown in Figure 5.



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

Figure 4: (a) Non-unit-width curve skeleton (b) a crowded region (c) two exits of this crowded region (d) the shortest path.

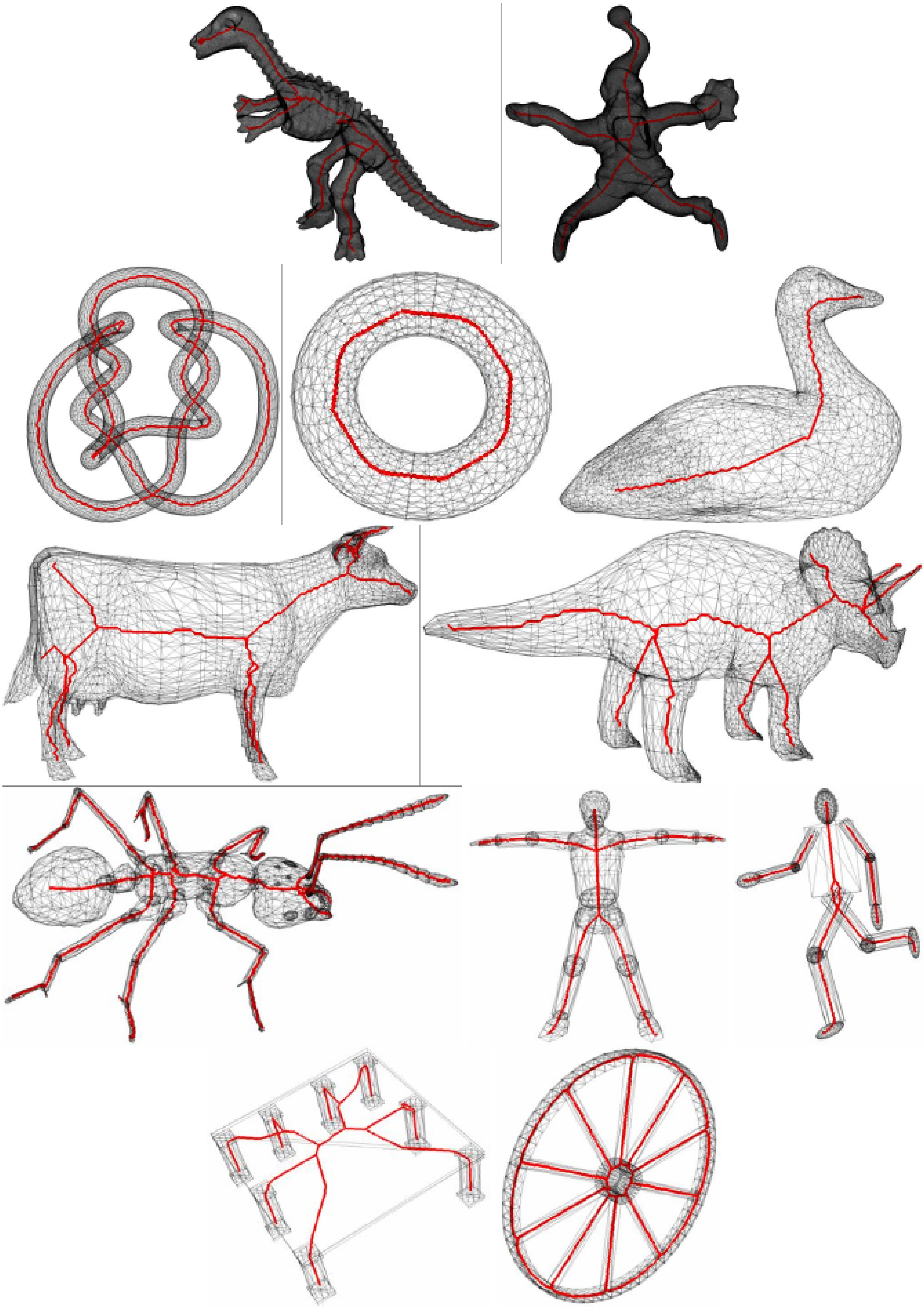


Figure 5: Some examples of unit-width skeletons generate with our algorithm.

References

[1] H. Blum. A transformation for extracting new descriptors of shape, *Models for the Perception of Speech and Visual Form*, pp. 362–380, MIT Press, Cambridge, MA, USA, 1967.

[2] N. D. Cornea. Curve-Skeletons: Properties, Computation And Applications, Ph.D. Thesis, The State University of New Jersey, May 2007.

[3] C. M. Ma, M. Sonka. A fully parallel 3D thinning algorithm and its applications, *Computer Vision and Image Understanding*, 64 (3): pp 420-433, 1996.

[4] K. Palagyi and A. Kuba. A 3D 6-subiteration thinning algorithm for extracting medial lines, *Pattern Recognition Letters*, 19 (7): pp 613-627, 1998.

[5] C. Lohoua and G. Bertrand. A 3D 6-subiteration curve thinning algorithm based on P-simple points, *Discrete Applied Mathematics*, Vol. 151, pp 198–228, 2005.

[6] C. Pudney. Distance-Ordered Homotopic Thinning: A Skeletonization Algorithm for 3D Digital Images, *Computer Vision and Image Understanding*, 72(3):404-413, 1998.

[7] A. Rosenfeld and A. C. Kak, *Digital Picture Processing*, 2nd ed., Vol. 2, Academic Press, New York, 1982.

[8] C. Arcelli and G. S. di Baja, A width independent fast thinning algorithm, *IEEE Trans. Pattern Anal. Mach. Intell.* 7, 1985, 463–474.

[9] R. L. Ogniewicz and M. Ilg. Voronoi Skeletons Theory and Applications, *CVPR* 1992.

[10] R. L. Ogniewicz and O. Kubler. Hierarchic Voronoi Skeletons, *Pattern Recognition*, 28 (3): 343-359 mar 1995.

[11] E. C. Sherbrooke, N. M. Patrikalakis and E. Brisson. An algorithm for the medial axis transform of 3d polyhedral solids, *IEEE Transactions on Visualization and Computer Graphics*, 2 (1): 44-61 Mar 1996.

[12] P. Giblin and B. B. Kimia. A formal classification of 3D medial axis points and their local geometry, *CVPR* 2000.

[13] F. F. Leymarie and B. B. Kimia. The Shock Scaffold for Representing 3D Shape, *LNCS 2059*, pp 216-229, 2001.

[14] F. F. Leymarie and B. B. Kimia. Computation of the Shock Scaffold for Unorganized Point Clouds in 3D, *CVPR'03*.

[15] T. Wang and A. Basu. A note on "A fully parallel 3D thinning algorithm and its applications", *Pattern Recognition Letters*, 28(4): 501-506, 2007.

[16] D. Brunner, G. Brunnett. An extended concept of voxel neighborhoods for correct thinning in mesh segmentation, Spring Conference on Computer Graphics, pp.119-125, 2005.

[17] E. W. Dijkstra: A note on two problems in connexion with graphs. In Numerische Mathematik, 1 (1959), S. 269–271.

[18] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. Introduction to Algorithms, Second Edition. MIT Press and McGraw-Hill, 2001. ISBN 0-262-03293-7. Section 24.3: Dijkstra's algorithm, pp.595–601.

[19] H. Sundar, D. Silver, N. Gagvani, S. Dickinson, Skeleton Based Shape Matching and Retrieval, Shape Modeling International 2003, 130-142.