Extracting medial curves on 3D images

Cherng-Min Ma *, Shu-Yen Wan, Her-Kun Chang

Department of Information Management, College of Management, Chang Gung University, 259 Wen-Hwa 1st Rd., Kwei-Shan, Taoyuan 333, Taiwan, ROC

Received 16 April 2001; received in revised form 26 June 2001

Abstract

This paper proposes a 2-subfield thinning algorithm for extracting medial curves on 3D binary images where the set of all voxels of an image is partitioned into two isometric subsets and the algorithm works alternatively on each subset at a time. The thinning algorithm is proved to preserve topology. An algorithm for verifying the topology soundness of thinning algorithms should be established to simplify the procedure for preparing such proofs. © 2002 Elsevier Science B.V. All rights reserved.

Keywords: Topology; Thinning; 2-Subfield; 3D Image; Medial curve

1. Background

A thinning algorithm is to reduce unnecessary information by peeling objects layer by layer so that the result is sufficient to allow topological analysis on applications of OCR or medical informatics, etc. A thinning algorithm should preserve topology that implies, for example, an object cannot be vanished completely or be split to two or more objects, and no background components can be created or merged. The simplicity of an object voxel is a well-known property in 3D thinning (Kong and Rosenfeld, 1989). For preserving topology, a thinning algorithm can only delete simple object voxels (i.e., change to background voxels). It is difficult to prove that a parallel thinning algorithm preserves topology since a number of object voxels can be deleted in parallel. Properties to simplify such proof procedures were proposed in (Kong and Rosenfeld, 1989; Ma, 1994).

There are two types of skeletons for thinning on 3D images: medial surfaces (Bertrand, 1995) and medial curves. A medial surface is a set of object voxels forming a surface of unit thickness, and a medial curve is a set of object voxels forming a curve of unit width. The paper proposes a 2-subfield topology preserving thinning algorithm on 3D images for extracting medial curves. The algorithm is alternatively applied to the two subfields of a 3D image. See Bertrand and Aktouf (1994) for other approaches of thinning algorithms by dividing a 3D image into several isometric...
subfields. For preserving geometry, the algorithm preserves end voxels. Several test images and their unit-width skeletons extracted by the algorithm are provided. Generally, verifying a thinning algorithm preserving topology is quite challenging. An “algorithm-verification algorithm” should be established to simplify the procedure for preparing such proofs.

2. Basic notations

A voxel of a 3D image is an element in the 3D integral space. A unit edge is a set of two voxels of distance 1 where the two voxels are 6- or directly adjacent; a unit square is a set of four voxels forming a $2 \times 2$ square where any two voxels of distance $\sqrt{2}$ are diagonally adjacent; a unit cube is a set of eight voxels forming a $2 \times 2 \times 2$ cube where any two voxels of distance $\sqrt{3}$ are diametrically adjacent (see Fig. 1). Two voxels of distance 1 or $\sqrt{2}$ are 18-adjacent; two voxels of distance 1, $\sqrt{2}$ or $\sqrt{3}$ are 26-adjacent. The relations of the six directions, namely upper, lower, north, south, east and west, of a 3D coordinate system are shown in Fig. 2. Then, the upper neighbor of $x$ (denoted $u(x)$) is the voxel next to $x$ from the upper direction; $l(x)$, $n(x)$, $s(x)$, $e(x)$, and $w(x)$ are defined similarly. The lower-north neighbor of $x$ (denoted $l(n(x))$ or briefly $ln(x)$) is defined accordingly; other diagonal neighboring voxels of $x$ are denoted similarly. The union of $x$ and all voxels 26-adjacent to $x$ is denoted $N(x)$. Let $N'(x) = N(x) - \{x\}$. Let $N_{18}(x)$ be the union of $x$ and all voxels 18-adjacent to $x$.

The set of voxels of a 3D binary image is partitioned into two subsets, $C$ (object voxels) and $C'$ (background voxels). Two $C$-voxels (voxels in $C$) are adjacent if they are 26-adjacent; two $C'$-voxels (voxels in $C'$), or a $C$-voxel and a $C'$-voxel are adjacent if they are 6-adjacent. A voxel is adjacent to a set of voxels if it is adjacent to any voxel of the set. A $C$ neighbor of a voxel $x$ is a $C$-voxel that is adjacent to $x$. Two $C$-voxels in a subset $X$ of voxels in a 3D image are connected if there exists a sequence of $C$-voxels joining them in $X$ where all two consecutive $C$-voxels in the sequence are 26-adjacent. A $C$-component in $X$ is a connected subset of $C$-voxels in $X$ that is not adjacent to any other $C$-voxel in $X$. The concepts of $C'$-neighbors, connectedness of $C'$-voxels, $C'$-component are defined similarly. A (north) border voxel is a $C$-voxel with a direct (north) $C'$-neighbor. Border voxels from other directions are defined similarly. An end voxel is a $C$-voxel that has only one $C$-neighbor. A non-end voxel is a $C$-voxel that is not an end voxel.

3. The 2-subfield thinning algorithm and its applications

Suppose the set of voxels of a 3D image is partitioned into two isometric subsets (called subfields) so that two 6-adjacent voxels are in different subfields, and two diagonally adjacent voxels are in the same subfield. We propose a 2-subfield thinning algorithm to alternatively delete $C$-voxels in each subfield. Definitions 3.1 and 3.2 are to delete a $C$-voxel with a direct or diagonal $C$-neighbor, respectively.

Definition 3.1. In Fig. 3(a), a $C$-voxel $x$ of a 3D image is $U$-deletable (upper deletable) if all of the following conditions hold (up to the 90°, 180° and 270° rotations on the z-axis):

1. $u(x) \in C'$ and $\beta(x) = l(x) \in C$;

Fig. 1. The configurations of unit edges, unit squares and unit cubes.

Fig. 2. The six orthogonal directions of the 3D space.
2. \( a \in C \) implies \( b \in C \); and
3. \( c \in C \) implies either \( b \in C \) or \( d \in C \).

\( L\alpha \), \( N\beta \), \( S\gamma \), \( E\delta \), \( W\varepsilon \)-deletable \( C \)-voxels are defined similarly. Such a \( C \)-voxel is also called \( \text{ORTH-deletable} \).

**Definition 3.2.** In Fig. 3(b), a \( C \)-voxel \( x \) of a 3D image is \( \text{UE-deletable} \) (upper-east deletable) if all of the following conditions hold:
1. \( \delta(x) = hw(x) \in C \) and all other voxels of \( x \)'s \( 3 \times 3 \) neighborhood containing \( \delta(x) \) are in \( C' \);
2. \( a_i \in C \) implies that at least one voxel in \( \{b_i, d_i, e_i\} \) is a \( C \)-voxel;
3. \( b_i \in C \) or \( e_i \in C \) implies either \( e_i \in C \) or \( f_i \in C \);
4. \( d_i \in C \) or \( g_i \in C \) implies either \( e_i \in C \) or \( h_i \in C \); and
5. \( e_i \in C' \) implies that then there is at least one voxel in \( \{b_i, d_i, f_i, h_i\} \) which is a \( C' \)-voxel, where \( i = 1 \) or 2. A \( \text{UN-}, \hspace{1mm} \text{US-}, \hspace{1mm} \text{USW-}, \hspace{1mm} \text{LN-}, \hspace{1mm} \text{LS-}, \hspace{1mm} \text{LE-}, \hspace{1mm} \text{LW-}, \hspace{1mm} \text{NE-}, \hspace{1mm} \text{NW-}, \hspace{1mm} \text{SE-} \) or \( \text{SW-deletable} \) \( C \)-voxel is defined similarly. Such a \( C \)-voxel is also called \( \text{DIAG-deletable} \).

\( \beta(x) \) in Definition 3.1 is a relative term for \( x \) to be \( \text{ORTH-deletable} \). For example, if \( x \) is \( N \)-deletable, then \( \beta(x) = s(x) \), and if \( x \) is \( E \)-deletable, then \( \beta(x) = w(x) \). Similarly, \( \delta(x) \) in Definition 3.2 is a relative term for \( x \) to be \( \text{DIAG-deletable} \). For example, if \( x \) is \( \text{UN-deletable} \), then \( \delta(x) = ls(x) \), and if \( x \) is \( \text{SE-deletable} \), then \( \delta(x) = mw(x) \). Definition 3.3 is to preserve a \( C \)-voxel with a diagonal \( C \)-neighbor and Definition 3.4 is to delete small branches.

**Definition 3.3.** Suppose \( x \) and \( y \) are two distinct \( C \)-voxels where \( x \) is \( \text{DIAG-deletable} \), and \( y \) is one of the voxels in \( \{ue(x), uw(x), uw(x), us(x), nw(x), sw(x)\} \). Then \( y \) is \( \text{DIAG-preserved} \) if \( x \) and \( y \) are the only two \( C \)-voxels in the unit square containing \( \{x,y\} \).

**Definition 3.4.** A \( C \)-voxel \( x \) is called a \textit{twig voxel} if one of the following conditions holds:
1. \( x \) has only one \( C \)-neighbor \( y \) and
   (a) \( y \) has three or more \( C \)-neighbors (including \( x \)), or
   (b) \( y \) has only two \( C \)-neighbors, \( x \) and \( z \), and \( z \) has three or more \( C \)-neighbors; or
2. \( x \) has only two adjacent \( C \)-neighbors, \( y \) and \( z \), where \( \{x,y,z\} \) is not a \( C \)-component.

For example, each \( x \) in the above figures is a twig voxel where \( x \), \( y \), \( z \) and any voxel marked \( \bullet \) are \( C \)-voxels, and all unmarked voxels are \( C' \)-voxels. Algorithm MedialCurve is applied alternatively to either subfield to delete \( C \)-voxels by Rule 3.5 until no \( C \)-voxels can be deleted. A thinning algorithm is an iterative process where, in each iteration, an operator is applied to delete \( C \)-voxels (i.e., every \( C \)-voxel matches the operator is deleted). Clearly, a thinning algorithm is topology preserving if, in each iteration, the deletion by the operator is topology preserving. The following Rule 3.5 is the operator of Algorithm Medial-Curve.
Rule 3.5. Let \( x \) be a border \( C \)-voxel of a 3D image. Then \( x \) is deleted if any of the following conditions holds:
1. \( x \) is an ORTH-deletable voxel;
2. \( x \) is a DIAG-deletable but not DIAG-preserved voxel; or
3. \( x \) is a twig voxel.

By Rule 3.5, we introduce the structure of the iterative 2-subfield parallel thinning Algorithm MedialCurve as follows:

**Algorithm MedialCurve**
- \( CurrentSubfield = \) the subfield contains the voxel of the coordinate \((0, 0, 0)\);
- repeat
  - delete every non-end \( C \)-voxel satisfying Rule 3.5 in \( CurrentSubfield \);
  - \( CurrentSubfield = \) the alternative subfield;
- until no \( C \)-voxels can be deleted in either subfield.

4. Discussions and conclusion

Testing images are provided in Figs. 4–11 where a cube stands for a voxel. Figs. 4–6, show images of size \( 40 \times 40 \times 40 \) where a letter ‘b’ and objects with holes are presented. Their topology preserving skeletons are provided. Figs. 7 and 8 show images of size \( 128 \times 128 \times 128 \) containing letters ‘A’ and ‘X’. Two small branches are shown in the middle of the skeleton in Fig. 8 that are mainly because when objects are thicker, boundary noises are likely to influence the structure of the skeletons. Fig. 9 shows the application of Algorithm MedialCurve on a special 3D configuration, node, where a node is an intercrossed cycle. Figs. 10 and 11 are normalized images of size \( 256 \times 256 \times 256 \) obtained from real 3D medical images. In Fig. 11, another prospect of the skeleton is provided to show that the skeleton corresponding to the stem of the object does not contain circles.

Algorithm MedialCurve is to generate thin curves on elongated objects. For an object like a flat board, an ideal skeleton should look like a thin surface (i.e., a 1-voxel thick surface). A thin-curve skeleton may look awkward if Algorithm MedialCurve is applied to such a flat-board object. Although Algorithm MedialCurve guarantees the thin-curve skeleton preserves topology of the original flat-board object, the thin-curve skeleton is less meaningful to the original object.

A zig-zag skeleton is a consequent result of a 2-subfield thinning algorithm (see Fig. 6). For example, consider the following two 2D elongated objects of \( 2 \times 10 \) \( C \)-pixels. The first case shows a possible skeleton generated by a 2-subfield thinning algorithm, and the second case shows a possible skeleton generated by a fully parallel thinning algorithm where a pixel marked \( \circ \) is a \( C' \)-pixel that is deleted by the corresponding thinning algorithm, a pixel marked \( \bullet \) is a \( C \)-pixel, and all unmarked pixels are \( C \)-pixels. Clearly, a skeleton generated by a 2-subfield thinning algorithm may not be as smooth as a skeleton generated by a fully parallel thinning algorithm.

A thinning algorithm is applied iteratively to delete \( C \)-voxels from a 3D image. Generally, there is only one deletion operation in each iteration of a fully parallel thinning algorithm where such a deletion removes the outmost layer of a \( C \)-component in parallel (see Ma and Sonka, 1996). A commonly used 6-subiteration thinning algorithm (see Bertrand, 1995; Palágyi and Kuba, 1998)

![Fig. 4. A letter ‘b’ and its skeleton (cubes stand for voxels).](image-url)
contains six subiterations in each iteration where, in each subiteration, the deletion is applied to border voxels from one of the six orthogonal directions. The deletion operation of our 2-subfield thinning algorithm is applied to C-voxels in one of the two subfields.
Fig. 8. A letter ‘X’ and its skeleton (cubes stand for voxels).

Fig. 9. A node and its skeleton (cubes stand for voxels).

Fig. 10. A tree structure object and its skeleton (cubes stand for voxels).
Suppose a fully parallel thinning algorithm takes \( n \) parallel iterations to thin an elongated \( C \)-component down to a unit-width skeleton. Then, in general, a 6-subiteration thinning algorithm takes \( 6 \times n \) parallel deletions for extracting a similar result. For a 2-subfield thinning algorithm, half of the \( C \)-voxels in the outmost layer of the \( C \)-component are deleted in the first parallel deletion, the rest half of \( C \)-voxels in the second outmost layer are deleted in the second parallel deletion, the rest half of \( C \)-voxels in the third outmost layer are deleted in the third parallel deletion, and so on. Generally, a 2-subfield thinning algorithm takes \( n + 1 \) parallel deletions for extracting a similar result.

An 8-subfield thinning algorithm on 3D images was proposed in (Bertrand and Aktouf, 1994). Their algorithm divides the 3D space into eight isometric subfields where two 26-adjacent voxels belong to two distinct subfields. Thus, the \( C \)-voxels...
in a flat unit thickness surface belong to four distinct subfields. Then, in general, it takes 4 parallel deletions for an 8-subfield thinning algorithm to delete a flat unit thickness surface. It is not difficult to see that, in general, the first 4 parallel deletions remove the first layer of an elongated C-component, the next 4 parallel deletions remove the second layer of the C-component, the following 4 parallel deletions remove the third layer of the C-component, and so on. Thus, in general, an 8-subfield thinning algorithm takes $4 \times (n + c)$ parallel deletions for extracting a similar skeleton where $c$ is a constant.

3D thinning is useful in medical image processing and in tracking moving objects. Topology and geometry preservations are major concerns of thinning algorithms. This paper presents a 2-subfield thinning algorithm on 3D images for extracting topology preserving skeletons as medial curves. Generally, verifying a thinning algorithm preserving topology is quite challenging. An algorithm for verifying the topology soundness of thinning algorithms should be established to simplify the procedure for preparing such proofs.

Appendix A. Topology preservation of thinning algorithms

Useful properties for determining the topology soundness of a single C-voxel of a 3D image were proposed in (Malandain and Bertrand, 1992) as follows:

**Definition A.1** (Malandain and Bertrand, 1992). Let $x$ be a C-voxel of a 3D image. Then $x$ is simple if and only if both of the following conditions hold:
1. there is only one C-component in $N^*(x)$;
2. $x$ is adjacent to only one C-component in $N_{18}(x)$.

Other useful properties related to the simplicity of C-voxels were proposed in (Saha and Chaudhuri, 1994; Saha and Rosenfeld, 2000). By Definition A.1, the image after exactly one simple C-voxel is deleted preserves the topology as the original image does. Since a parallel thinning algorithm may delete two or more C-voxels at the same time, Definition A.2 characterizes the simplicity of a set of simple C-voxels. Then we define the meaning of a thinning algorithm to preserve topology in Definition A.3.

**Definition A.2** (Kong, 1995; Ma, 1994). A set $D$ of C-voxels is called simple if $D$ can be ordered as a sequence in which every C-voxel is simple after all its predecessors in the sequence are deleted. A set of C-voxels is called non-simple if it is not simple.

**Definition A.3** (Kong, 1995; Ma, 1994). A thinning algorithm is topology preserving if it never deletes any non-simple set of C-voxels on any 3D image.

By Definition A.3, every possible 3D image should be tested for verifying the topology preservation of a 3D thinning algorithm. Clearly, this approach involves infinitely many images and hence is infeasible. Theorem A.4, expanded from Ronse’s result in (Ronse, 1988), shows that a 3D thinning algorithm can be proved to preserve topology by checking the configurations contained in a unit cube.

**Theorem A.4.** A thinning algorithm on 3D images is topology preserving if all of the following conditions hold:
1. only simple C-voxels can be deleted;
2. only simple sets of C-voxels in a unit square can be deleted; and
3. no C-component contained in a unit cube can be deleted completely.

It is worth developing a computer program that can automatically verify the topology soundness of a parallel thinning algorithm on 3D images based on Theorem A.4. Such a program is definitely helpful for designing topology preserving 3D thinning algorithms.

Appendix B. The topology soundness of medial curve

Before introducing lemmas and propositions, we introduce a notation $K$ as follows. Let $x, y$ be
two distinct voxels. Then \( K(x,y) \) is the union of every unit cube containing both \( x \) and \( y \). If \( x \) is 6-adjacent to \( y \), then \( K(x,y) \) is a set of four unit cubes sharing a common unit edge; if \( x \) is diagonally adjacent to \( y \), then \( K(x,y) \) is a set of two unit cubes sharing a common unit square; if \( x \) is diametrically adjacent to \( y \), then \( K(x,y) \) is a unit cube; if \( x \) is not 26-adjacent to \( y \), then \( K(x,y) \) is an empty set. Let \( X \) be a set of voxels in a 3D image. Denote \( \#_C(X) \) to be the number of \( C \)-components in \( X \). Thus, \( \#_C(K(x,y)) \) is the number of \( C \)-components in \( K(x,y) \). Lemma B.1 is to determine the simplicity of a set of two simple \( C \)-voxels. Then we prove Lemmas B.2–B.4 which shows that Rule 3.5 in either subfield satisfies Theorem A.4(1–3), respectively. Proposition B.5 follows immediately after Lemmas B.2–B.4.

**Lemma B.1.** Let \( x \) and \( y \) be two diagonally adjacent simple \( C \)-voxels in a 3D image. Then \( \{x,y\} \) is simple if and only if \( \#_C(K(x,y) - \{x,y\}) = 1 \).

**Lemma B.2.** The deletion by Rule 3.5 in either subfield satisfies Theorem A.4(1).

**Proof.** Let \( x \) be a \( C \)-voxel in a 3D image that satisfies Rule 3.5. To show this lemma, we need to show that \( x \) is simple by the following cases.

**Case 1.** \( x \) is a twig voxel. If \( x \) satisfies Definition 3.4(1), then \( x \) has exactly one \( C \)-neighbor \( y \) which, by Definition A.1, implies that \( x \) is a simple \( C \)-voxel. If \( x \) satisfies Definition 3.4(2), then \( x \) has exactly two distinct \( C \)-neighbors, \( y \) and \( z \), and \( y \) is adjacent to \( z \) which, by Definition A.1, implies that \( x \) is a simple \( C \)-voxel.

**Case 2.** \( x \) is ORTH-deletable. WLOG (without losing generality) let \( x \) be \( U \)-deletable. By Definition 3.1(1–3), every possible \( C \)-neighbor of \( x \) is 26-connected to \( \beta(x) = l(x) \in C \) in \( N^*(x) \). By Definition 3.1(2), every possible \( C \)-voxel 6-adjacent to \( x \) is 6-connected to \( u(x) \) in \( N_{18}(x) \). Thus, by Definition A.1, every ORTH-deletable \( C \)-voxel is simple.

**Case 3.** \( x \) is DIAG-deletable. WLOG let \( x \) be \( UE \)-deletable. See Fig. 3(b), by Definition 3.2(1–3), every possible \( C \)-neighbor of \( x \) is 26-connected to \( \delta(x) = lw(x) \in C \) in \( N^*(x) \). By Definition 3.2(4), every possible \( C' \)-voxel 6-adjacent to \( x \) is 6-connected to a \( C' \)-voxel in \( N_{18}(x) \). Thus, by Definition A.1, every DIAG-deletable \( C \)-voxel is simple.

**Lemma B.3.** The deletion by Rule 3.5 in either subfield satisfies Theorem A.4(2).

**Proof.** Consider two diagonally adjacent \( C \)-voxels, \( x \) and \( y \), each of which is deleted by Rule 3.5, we need to show that \( \{x,y\} \) is simple. WLOG we consider \( x \) as a twig voxel, ORTH and DIAG-deletable as follows.

**Case 1.** \( x \) is a twig voxel. If \( x \) satisfies Definition 3.4(1), then \( y \) is the only \( C \)-neighbor of \( x \) where \( y \) is non-simple and cannot satisfy Rule 3.5. If \( x \) satisfies Definition 3.4(2), then \( y \) and another \( C \)-voxel, \( z \), are the only two \( C \)-neighbors of \( x \), and \( z \) is the only \( C \)-voxel in \( K(x,y) - \{x,y\} \). Thus, \( \#_C(K(x,y) - \{x,y\}) = 1 \) and, by Lemma B.1, \( \{x,y\} \) is simple.

**Case 2.** \( x \) is ORTH-deletable. WLOG let \( x \) be \( U \)-deletable and \( y \) be one of the voxels \( \{a_1,a_2,c_2\} \) in Fig. 12. Since by Definition 3.1(2), \( a_1 \in C \) implies \( a_2 \in C \), \( \#_C(K(x,a_1) - \{x,a_1\}) = 1 \). Since \( \beta(x) \in C \), \( \#_C(K(x,a_3) - \{x,a_3\}) = 1 \). Since by Definition 3.1(2–3), every possible \( C \)-voxel in \( K(x,c_2) - \{x,c_2\} \) is connected to \( \beta(x) \) in \( K(x,c_2) - \{x,c_2\} \), \( \#_C(K(x,c_2) - \{x,c_2\}) = 1 \). No matter \( y \) is \( a_1 \), \( a_3 \) or \( c_2 \), by Lemma B.1, \( \{x,y\} \) is simple.

**Case 3.** \( x \) is DIAG-deletable. By the above cases, we only need to consider the condition when \( y \) is DIAG-deletable as well. If \( x \) is in \( \{ue(y),um(y), \)

---

**Fig. 12.** \( x \) and one of its diagonally \( C \)-neighbors are deleted where a voxel marked \( o \) or \( \bullet \) is a \( C' \)- or \( C \)-voxel, respectively, and a voxel marked \( \Box \) is a \( C' \)-voxel or a \( C \)-voxel.
Then, by Definition 3.3, $x$ is DIAG-preserved after $y$ is deleted. Otherwise, $x$ is in \{lw(x), ls(x), ln(x), se(x), ne(x)\}, that is, $y$ is in \{ue(y), un(y), uw(y), us(y), nw(y), sw(y)\}. Then, by Definition 3.3, $y$ is DIAG-preserved after $x$ is deleted. Thus, \{x, y\} cannot satisfy Rule 3.5 at the same time. \hfill \Box

**Lemma B.4.** The deletion by Rule 3.5 in either subfield satisfies Theorem A.4(3).

**Proof.** By Lemmas B.2 and B.3, let $X$ be a $C$-component consisting of three or four $C$-voxels contained in a unit cube and in the same subfield. Since every voxel in $X$ has two or more $C$-neighbors and $X$ itself is a $C$-component, no voxel in $X$ can be a twig voxel. Since no voxel in $X$ has a direct $C$-neighbor, no voxel in $X$ can be ORTH-deletable. By Lemma B.4, no two diagonally adjacent $C$-voxels can be DIAG-deleted in parallel. Thus, no $C$-component contained in a unit cube can be deleted by Rule 3.5 in parallel. \hfill \Box

By Lemmas B.2–B.4, we have the following proposition.

**Proposition B.5.** Algorithm MedialCurve preserves topology.

---

**References**


